

УНИВЕРЗИТЕТ У БЕОГРАДУ
ЕЛЕКТРОТЕХНИЧКИ ФАКУЛТЕТ

Владимир Р. Јоковић

**АУТОМАТИЗОВАНО ОЦЕЊИВАЊЕ
ПАПИРНИХ ТЕСТОВА
КОРИШЋЕЊЕМ ТЕХНИКА
ВЕШТАЧКЕ ИНТЕЛИГЕНЦИЈЕ**

докторска дисертација

Београд, 2023

UNIVERSITY OF BELGRADE
SCHOOL OF ELECTRICAL ENGINEERING

Vladimir R. Jocović

**AUTOMATED ASSESSMENT
OF PEN AND PAPER TESTS
USING ARTIFICIAL INTELLIGENCE**

Doctoral Dissertation

Belgrade, 2023

Ментори

- проф. др Бошко Николић, доктор електротехничких наука, Универзитет у Београду, Електротехнички факултет
- доцент Саша Стојановић, доктор електротехничких наука, Универзитет у Београду, Електротехнички факултет

Чланови комисије

- ван. проф. др Милош Цветановић, доктор електротехничких наука, Универзитет у Београду, Електротехнички факултет
- проф. др Јелица Протић, доктор електротехничких наука, Универзитет у Београду, Електротехнички факултет
- емеритус проф. др Душан Старчевић, доктор техничких наука, Универзитет у Београду, Факултет организационих наука
- проф. др Игор Тартаља, доктор техничких наука, Универзитет у Београду, Електротехнички факултет
- ван. проф. др Горан Квашчев, доктор електротехничких наука, Универзитет у Београду, Електротехнички факултет

Датум одбране

Предговор

У свету академске посвећености, пут асистента је трновит, испуњен разноврсним изазовима и притисцима. Кроз овај изазовни ход, сетио сам се драгоцене мудрости коју ми је уважени професор и колега Милош Цветановић поделио. Он је говорио о важности посматрања нашег рада из три перспективе, с тим да редослед њиховог навођења не имплицира већу важност једне перспективе у односу на другу:

- Прва перспектива је научни допринос. Када наш труд и рад допринесу новом сазнању, осветљењу непознатог и пружи темељ за нове хоризонте, тада смо створили нешто изванредно.
- Друга перспектива је новчана награда. Ако наш труд доноси не само духовно испуњење, већ и материјалну подршку, онда смо постигли хармонију између наше страсти и прагматичних потреба.
- Трећа перспектива је лично задовољство. Када уживамо у процесу, када сваки корак има своје чари и свака препрека нас изазива да растемо, тада смо пронашли благослов у свом раду.

Данас, док стојим пред овом дисертацијом о аутоматизованом оцењивању папирних тестова путем вештачке интелигенције, сећам се тих речи. Моји напори нису били узалудни. Успео сам да објединим све три перспективе: мој рад је допринео научном пољу, доносећи нову перспективу на традиционално оцењивање тестова; зарадио сам не само новчану награду, већ и дубоко поштовање за свој труд; и што је можда најважније, путовање је било испуњено страшћу и инспирацијом.

Ова дисертација представља више од сложених алгоритама и анализа; она је споменик за све нас који се усудимо да сањамо, радимо и постигнемо. Нека нас ова мудрост подсећа да успех није једноставно дефинисан, већ је моћно ткиво вишеструких компонената. Нека нас инспирише да тражимо те три перспективе и да радимо са страшћу ка стварању, мудрости ка напретку и задовољству ка личном расту. Јер, када се ове компоненте сједине, постиже се изузетно – и то је пут који сви можемо следити.

Захвалница

Желео бих да изразим своју дубоку захвалност према мојим родитељима Радисаву и Десанки за безусловну подршку и љубав коју су ми пружали. Хвала им што су ми омогућили школовање и унапредили моје знање. Захваљујем им на томе што су ме упућивали на различите активности попут енглеског језика, пливања, фолклора и каратеа, како бих се развијао као свестран појединац не само унутар школских зидова, већ и на свим осталим аспектима живота. Њихови драгоцени животни савети су ми много значили, а такође изузетно ценим то што су посвећивали своје време и енергију како бих ја напредовао. Оно што је најважније, увек су своје поступке према мени водили без личних интереса. Такође, неизмерно сам захвалан својој старијој сестри Јовани на подршци и љубави коју ми је пружала. Увек сам се могао ослонити на њу за дискусије о свим аспектима живота, како позитивним тако и изазовним. Наши филозофски разговори су ме дубоко обогатили и увек сам се осећао подржаним. Хвала јој што је увек била присутна као мудар саветник.

Такође, желим да се захвалим и својим учитељима, наставницима и професорима који су ме подучавали на мом путу школовања, као и свим бившим и садашњим члановима Катедре за рачунарску технику и информатику са којима сам сарађивао или данас сарађујем. Наравно, желим да искажем посебну захвалност својим колегама и пријатељима из канцеларије 909 у згради „Лола“: Дражену Драшковићу, Живојину Шуштрану, Марку Мићовићу и Урошу Раденковићу. Њихова несебична помоћ, сати смеха и играња на Сонијевој конзоли су ме испуњавали радошћу. Захваљујем и својим професорима и колегама Бошку Николићу, Саши Стојановићу, Милошу Цветановићу, Горану Квашчеву и осталим резидентима Рачунског центра Електротехничког факултета Универзитета у Београду, где смо проводили бројне опуштене сате у уживању у причи, изврсној храни и у друштву наших четвороножних пријатеља Алфија и Меденка. Целокупна позитивна енергија која је из тога проистекла, допринела је многим важним тренуцима у нашим животима.

Наслов докторске дисертације

Аутоматизовано оцењивање папирних тестова коришћењем техника вештачке интелигенције

Сажетак

Тема истраживања обухвата аутоматизацију процеса оцењивања папирних тестова применом вештачке интелигенције: машинског учења и компјутерске визије. Компјутерска визија се бави усавршавањем способности људског визуелног система и фокусира се на обраду дигиталних слика и видео записа. Иако постоје различите категорије компјутерске визије, већина се бави детекцијом објеката и препознавањем образаца. Развојем алгоритама развијају се специјализовани системи који обављају задатке слично људима, а технолошки напредак омогућава креирање система који у неким аспектима превазилазе људске способности.

Иако постоји много платформи за е-учење и оцењивање као што је *Moodle*, папирни тестови још увек нису напустили употребу. Разлози за то су различити и често су везани за ограничења рачунарских ресурса, што узрокује да се тестирање великог броја кандидата истовремено обавља ручно уз коришћење обичне или хемијске оловке. С обзиром на велики број кандидата и потребу за брзим резултатима, ручно оцењивање представља значајан изазов.

Истраживање се бави анализом често коришћених типова питања на папирним тестовима и њиховом класификацијом на основу суштине коју таква питања испитују. Такође, постојећа решења у овој области се анализирају и категоризују на основу нове класификације питања. Евалуација предложеног система се врши упоређивањем са постојећим системима у истој класи питања.

Циљ истраживања је развој система за аутоматизовано оцењивање папирних тестова уз помоћ вештачке интелигенције: машинског учења и компјутерске визије. Систем прима дигиталне слике страница тестова и примењује алгоритме за детекцију региона од интереса и оцењивање питања. Такође, систем добија конфигурабилну датотеку са структуром теста и на основу тога проверава исправност процеса детекције. Имплементирани систем не само да оцењује питања већ и идентификује нерегуларне тестове узроковане неправилним попуњавањем.

Резултат истраживања доноси значајну олакшицу особљу које се бави оцењивањем папирних тестова. Важности овог истраживања доприноси велики број релевантних референци из области примене вештачке интелигенције у процесу аутоматског оцењивања.

Кључне речи

Вештачка интелигенција, аутоматизовано оцењивање тестова, машинско учење, компјутерска визија, анализа слика, папирни тестови

Научна област

Рачунарска техника и информатика

Ужа научна област

Вештачка интелигенција

Doctoral dissertation title

Automated Assessment of Pen and Paper Tests Using Artificial Intelligence

Abstract

The research subject encompasses the automation of paper test grading through the utilization of artificial intelligence: machine learning and computer vision. Computer vision is concerned with enhancing the capabilities of the human visual system and concentrates on processing digital images and videos. Despite the various categories within computer vision, the majority is centered on detecting objects and recognizing patterns. As algorithms advance, specialized systems are crafted to undertake tasks akin to humans, and technological advancements empower the formation of systems that, in certain aspects, surpass human capabilities.

While numerous e-learning and assessment platforms such as Moodle exist, paper tests have not waned in popularity. Diverse reasons contribute to this, often tied to the constraints of computer resources. Consequently, assessing a significant number of candidates is conducted by hand, utilizing standard or ballpoint pens. Given the volume of applicants and the urgency for prompt outcomes, manual grading poses a considerable challenge.

The study also delves into analyzing frequently employed question types on paper tests and their categorization based on the core concepts these questions examine. Additionally, prevailing solutions within this domain undergo scrutiny and classification through a fresh issue-based system. The evaluation of the proposed system occurs via a comparison against prevailing systems within the same question category.

The research objective is the development of a system for automated paper test assessment through the fusion of artificial intelligence: machine learning and computer vision. The system processes digital images of test pages and employs algorithms for identifying key regions and scoring questions. Furthermore, the system processes a configurable file containing the test's structure, using it to verify the accuracy of the detection process. The actualized system not only assesses questions but also identifies aberrant tests stemming from incorrect responses.

The research outcomes offer notable relief to personnel engaged in paper test grading. The value of this research comes from the wealth of references in the field of AI applied to automated assessment.

Key words

Artificial intelligence, automated test assessment, machine learning, computer vision, image analysis, paper tests

Scientific field

Computer engineering and information technology

Scientific subfield

Artificial intelligence

Садржај

1. Увод	1
2. Класификација најчешће коришћених питања на папирним тестовима	11
2.1. Вишеструки избор	11
2.2. Повезивање	13
2.3. Кратак одговор	16
2.4. Есеј	18
3. Преглед постојећих решења за аутоматизовано оцењивање папирних тестова	20
3.1. TARS	23
3.2. MCTQF	27
3.3. MCG-RAS	30
3.4. GMCQ-FA	33
3.5. Eyegrade	36
3.6. ASSHEP	39
3.7. AGHAS	42
3.8. ARHC	44
3.9. FASAS	48
3.10. ISSHSA	49
3.11. TSAWR	51
3.12. HSAES	53
3.13. AGHNA	55
3.14. AGSLCQ	59
4. Имплементациони детаљи предложеног система	61
4.1. Систем за тестирање кандидата	61
4.2. Подсистем за процесирање папирног теста	64
4.2.1. Препознавање једнодимензионалног бар-кода	70
4.2.2. Препознавање дводимензионалног бар-кода	74
4.2.3. Препознавање граничника региона од интереса	77
4.2.4. Процесирање питања класе „Вишеструки избор“	81
4.2.5. Процесирање питања класе „Повезивање“	89
4.2.6. Процесирање питања класе „Кратак одговор“	100
5. Евалуација имплементираних система	107
5.1. Евалуација препознавања једнодимензионалног бар-кода	107
5.2. Евалуација препознавања дводимензионалног бар-кода	107
5.3. Евалуација процесирања питања класе „Вишеструки избор“	108
5.4. Евалуација процесирања питања класе „Повезивање“	109

5.5. Евалуација процесирања питања класе „Кратак одговор“	111
5.6. Сумарни приказ резултата	112
5.7. Анализа постигнутих резултата	113
6. Закључак	116
Литература	120

1. Увод

Све веће присуство персоналних рачунара, Интернета (енг. *Internet*) и Светске мреже (енг. *World Wide Web*) учинили су знање доступнијим и на дохват руке него икада раније. Последично, то је довело до развоја различитих система за онлајн учење (*Moodle*, на пример), чији је циљ да организују знање и презентују га на структуриранији начин у виду курсева и предавања. Штавише, ове платформе за е-учење почеле су да уграђују функционалности које омогућавају процес испитивања кандидата који су се пријавили за одређени курс [1].

Ове платформе за онлајн учење омогућавају истовремено спровођење процеса испитивања за велики број кандидата. Већина ових платформи не потврђује прави идентитет кандидата током процеса тестирања, осим њихових акредитива за пријаву. Ипак, постоје обећавајући резултати које пружају сигурности системи за константну аутентикацију кандидата у процесу тестирања [2]. Ове системе је могуће интегрисати у платформе за електронско учење и они врше проверу идентитета коришћењем више биометријских улаза као што су слика лица кандидата, глас кандидата, кретање миша и брзина куцања на тастатури итд. Међутим, прикупљање свих ових података може бити проблематично са становишта приватности појединца.

Ипак, већина платформи за електронско учење у свом основном облику није у стању да спрече било какву врсту неакадемске радње коју кандидат може да покуша да почини, као што је решавање теста уз помоћ блиског пријатеља или било које релевантне литературе. У свом основном облику, ове платформе врше онемогућавање основних нежељених радњи, као што је нпр. копирање и лепљење садржаја са неког другог извора података (енг. *copy-paste*) или користе заштиту од скрипти употребом потврдног кода (енг. *captcha*) [3]. Ово не значи да су платформе за онлајн учење корисне искључиво за организовање и презентовање знања и спровођење необавезних тестова за кандидате. Могу се користити за потребе тестирања у учионицама са довољним рачунарским ресурсима и наставницима који контролишу процес тестирања. Штавише, ови системи су у стању да изводе процес оцењивања много брже од људи задужених за ручно оцењивање тестова.

Платформе за електронско учење и испитивање кандидата нуде многоструке предности. Неке од њих су побољшана интерактивност, тренутне повратне информације и поједностављена администрација. Међутим, у многим институцијама које спроводе процесе испитивања великог броја кандидата и даље се користе папирни тестови [4].

Упорно коришћење папирних тестова потиче од неколико фактора. Прво, доступност тестова на папиру омогућава њихову примену у регионима са ограниченим приступом дигиталним уређајима или електричном енергијом. Друго, познавање формата оловке и папира изазива осећај удобности и ублажава анксиозност приликом тестирања код многих појединаца [5]. Треће, не може се занемарити уочена инхерентна сигурност тестова на папиру, у смислу да је тест одштампан за сваког кандидата и не може се користити за било какве нечасне радње у процесу тестирања. Такође, није потребно обезбедити рачунарски систем за сваког кандидата који има приступ ка интернету.

Штавише, исплативост папирних тестова показује се као предност у одређеним контекстима, јер папирни тестови елиминишу потребу за значајним улагањима у дигиталну инфраструктуру и текуће одржавање. Поред тога, свестраност тестова на папиру у прилагођавању различитих формата оцењивања, посебно оних који захтевају замршене цртеже, сложене математичке једначине или писање слободном руком, остаје истакнута предност. Опиљива природа папира олакшава ове задатке и чини их природнијим у односу на њихове дигиталне алтернативе [6].

У светлу горе наведених фактора, папирни тестови настављају да одржавају своју широку употребу у образовним институцијама [7]. Такође, постоји много процеса испитивања који се морају спровести у исто време за све кандидате. Ово се углавном односи на тестирања као што су пријемни испити или тестирања за аплицирање за стипендије, што су масовнији процеси који се ређе дешавају, али и на процесе испитивања у основношколским и средњошколским установама који су релативно мањег обима, али се изводе знатно чешће.

Организатори испитног процеса често имају потешкоће у спровођењу испитног процеса користећи онлајн платформе за учење, јер су већ спутани разним разлозима који се не тичу самих платформи за учење. Ови разлози укључују оскудицу и непоузданост рачунарских ресурса, недостатак простора за вођење процеса тестирања, недостатак особља задуженог за праћење процеса тестирања и велики број пријављених кандидата за полагање теста. Стога и данас није реткост да се испитивања ученика и студената у школама спроводе традиционалним тестовима путем оловке и папира, нарочито у местима без приступа рачунару или у ситуацијама где постоји великим број пријављених кандидата [8].

Након што се процес тестирања заврши, попуњене папирне тестове наставници морају ручно оцењивати, што представља значајне изазове, уводи додатно оптерећење за наставно особље, којем се намећу огромни трошкови, узимајући у обзир да се ови испитни процеси обично спроводе у масовним курсевима и групама са значајним бројем кандидата. Без обзира на то, због великог броја кандидата, задатак евалуације кандидата често постаје монотон, нестимулативан и заморан. Штавише, процес евалуације уводи ниво субјективности који се у великој мери може ублажити коришћењем типова питања који омогућавају прецизне и недвосмислене одговоре. Ипак, неизбежно је да се грешке могу појавити током евалуације тако великог броја листова одговора у самом процесу прегледања папирних тестова [9].

Поред тога што је ручно оцењивање подложно грешкама и одузима много времена, постоје и испитни процеси који захтевају резултате испитивања у најкраћем временском периоду. Дакле, постоји потреба да се процес испитивања олакша и убрза тако што ће се у највећој могућој мери аутоматизовати, што такође помаже у растерећењу наставног особља. На тај начин, не само да би терет био скинут са плећа наставног особља које често сматра да задатак оцењивања папирних тестова није инспиративан, већ би им то такође омогућило додатно време да се посвете изради висококвалитетних тестова [10].

Због тога постоји потреба за развојем софтверских система за аутоматизовано оцењивање папирних тестова. Да би могли да изврше процес оцењивања, ови системи морају да имају дигиталну верзију теста који треба да се оцењује. Додатно, ови системи углавном и користе неку врсту компјутерске визије за обраду слика података [11] представљених у дигиталној верзији теста.

Компјутерска визија је једно од најмоћнијих поља и есенцијални део великог мозаика вештачке интелигенције, нудећи иновативне начине како компјутерски системи могу да „виде“ и „разумеју“ свет око себе [12]. Главни фокус компјутерске визије је да покуша да опонаша софистицирани систем људског вида [13]. На тај начин се омогућава наменским рачунарским системима да одреде и обрађују објекте, линије, текстуре и остале визуелне атрибуте на дигиталним сликама и видео записима са приближно истом прецизношћу као и људи.

Откривање и препознавање објеката на сликама и видео записима представља изазов за истраживаче и инжењере, али и невероватну прилику за примену компјутерске визије, нарочито у спрези са дубоким учењем [14]. Од једноставних објеката, као што су кругови и квадрати, до сложених структура и сцена, као што су људи, аутомобили и зграде, компјутерски системи могу да науче да их разазнају и класификују са прецизношћу која надилази могућности човека. Овакве способности имају дубоке последице у различитим областима, од безбедности и аутономне вожње до медицине и манипулације роботима [15].

Обиље дневно генерисаних визуелних података једна је од главних покретачких снага раста компјутерске визије. У савременом свету, број визуелних података који се дневно генеришу је већи него икад, будући да се скупљају са камера, сензора и других уређаја присутних у окружењу. Велика количина прикупљених података покреће напредак компјутерске визије, потпомагајући развој нових алгоритама и модела који су у могућности да не само анализирају, већ и да утичу на окружење на начин сличан човеку [16].

Постоје различите категорије компјутерске визије, од којих је већина усредсређена на откривање објеката и препознавање образаца [17]. Узимајући у обзир огромна побољшања алгоритама компјутерске визије, стопа тачности идентификације објеката расте. Дакле, наменски системи компјутерске визије постају моћнији од људи у обављању одређених задатака. Штавише, имајући у виду растуће брзине различитих хардверских компоненти које се дешавају током времена, системи засновани на компјутерској визији повећавају способност извршавања задатака већом брзином од људи.

Једна од значајних примена компјутерске визије је њена употреба у аутоматизованом оцењивању папирних тестова. Овај концепт се уклапа у шире људско стремљење да уклони монотоне и временски захтевне задатке из образовног окружења. Коришћењем компјутерске визије, системи могу детектовати и анализирати питања и одговоре на папирним тестовима са брзином и прецизношћу која није могућа ручним приступом.

Ово има дубоке последице на образовни процес, омогућавајући брже и ефикасније оцењивање кандидата и пружајући више времена за образовне активности које захтевају анализу и размишљање. Упркос изазовима који се појављују, као што су сложени обрасци одговора и варијације у писању, истраживачи налазе иновативне начине да примене компјутерску визију у овом контексту и допринесу напретку образовних процеса. Стога је постао очигледан императив да се развију различити типови система који би омогућили аутоматизовано оцењивање папирних тестова [18].

Неки системи користе неуронске мреже да би детектовали области од интереса на дигиталним сликама [19]. С друге стране, постоје системи који се базирају на оријентирним тачкама, ознакама и граничницима који се појављују на папирним тестовима. Они користе фиксне вредности померања између важних елемената, али и узимају у обзир структуру самог теста и примењују ограничења која су наметнута његовом структуром. Додатно, неки системи се ослањају на специфичне алгоритме за препознавање одређених облика који означавају питања или одговоре. Овакав приступ омогућава системима да испоље неки облик интелигентног понашања, сличан оном који би произашао из вештачке интелигенције.

На Електротехничком факултету Универзитета у Београду дужи низ година разматра се питање аутоматског оцењивања папирних тестова. Реализовани системи су се највише користили на предметима који су у вези са основама програмирања. Програмирање 1 представља уводни предмет у свет програмирања, који служи да презентује најбитније концепте у програмирању и архитектури рачунара. Његова сврха је да уведе студенте са мало или нимало искуства у програмирању у свет програмирања. Стога, иако је фокус предмета на програмирању у вишем програмском језику Пајтон (енг. *Python*), почетни део овог курса презентује основне архитектуралне концепте рачунара који су примењиви и у данашњим рачунарима.

У оквиру Катедре за рачунарску технику и информатику на Електротехничком факултету Универзитета у Београд, реализована је домаћа (енг. *in-house*) архитектура рачунара од стране професора овог факултета Јозе Дујмовића. Ова архитектура је заснована на класичној фон Нојмановој архитектури (енг. *Von Neumann architecture*), која се односи на дизајн рачунара који користи јединствену структуру за складиштење у којој се чувају и инструкције и подаци. Поред осмишљавања архитектуре, професор Дујмовић је осмислио и мали

инструкцијски скуп чија је намена управљање овом архитектуром. Овај скуп машинских инструкције представља у ствари симболички машински језик (енг. *assembly language*) за програмирање ове архитектуре. Осмишљену архитектуру је професор Дујмовић назвао Пико компјутер (енг. *pico-Computer*), због малог броја доступних програмских инструкција за управљање хардвером.

Предмет Програмирање 2 представља напреднији наставак предмета Програмирање 1 на Електротехничком факултету Универзитета у Београду и подразумева да студенти већ поседују одређени начин размишљања карактеристичан за програмирање. Фокус предмета је на програмирању у вишем програмском језику *C*, као и на анализи сложености алгоритама. Поред тога уводе се сложенији алгоритамски концепти. Предмет Програмирање 2 сваке године похађа близу 1 000 студената, док предмет Програмирање 1 похађа око 800 студената. Предмете Програмирање 1 и Програмирање 2 током семестра у којима се одржавају прате и курсеви Практикум из програмирања 1, односно Практикум из програмирања 2.

На Електротехничком факултету Универзитета у Београду интензивно се користи *Moodle* платформа за електронско учење [20]. Она подржава могућност организовања градива предмета на факултету коришћењем концепта курса, који је доступан у оквиру ове платформе. Поред тога, платформа може да се користи и у сврхе извођења процеса испитивања и аутоматизованог оцењивања кандидата.

У свом основном режиму рада ова платформа дозвољава да се у тест додају различите врсте питања. За прављење теста и питања, као и организовање питања у оквиру теста није потребно поседовати никакво програмерско знање. Међутим, ова платформа пружа могућност инкорпорирања разноврсних програмских додатака (енг. *plugins*).

Док се неки од програмских додатака за ову платформу могу користити у свом основном облику, одређени програмски додаци могу да се програмирају. Један од таквих је бесплатни програмски додатак отвореног кода *CodeRunner*, који може да се конфигурише одговарајућим скриптама написаним на подржаним програмским језицима. *CodeRunner* може да покреће програмски код који су кандидати послали као одговор на широк спектар програмерских питања на много различитих програмских језика. Намењен је првенствено за коришћење на курсевима рачунарског програмирања, иако се може користити за оцењивање било којег питања за које је одговор текст.

Мотивација за конфигурисање и коришћење овог програмског додатка била је увођење аутоматизације у процес оцењивања програмских кодова написаних од стране студената на предметима рачунарског програмирања на Електротехничком факултету Универзитета у Београду. Разлози за аутоматизацију процеса оцењивања су оправдани. Она доноси растерећење наставног кадра који може ослобођено време да искористи у сврхе академских унапређења курсева.

Такође, аутоматизација процеса смањује грешке настале у процесу ручног оцењивања. Грешке се дешавају услед саме природе посла ручног оцењивања, који је неинспиративан, заморан и мукотрпан. Грешке у процесу ручног оцењивања не дешавају се само на штету студената, већ неретко и на њихову корист. Поред тога, аутоматизација оцењивања доноси већу објективност у процес прегледања.

Најпре је реализовано решење за аутоматизовано оцењивање програмских кодова написаних на програмском језику *C*, конфигурисањем програмског додатка *CodeRunner*. Овај систем је реализован у програмском језику Пајтон и показао је изузетне резултате у тестирањима која су извођена на курсевима Програмирање 2 и Практикум из програмирања 2 на Електротехничком факултету Универзитета у Београду. Систем је испољио велику

скалабилност и прецизност у оцењивању [21]. Слика 1 приказује пример дела кода за конфигурацију програмског додатка *CodeRunner*.

```

63 def findAndCheckOrgDirective ( lines ):
64     ORG_DIRECTIVE = "org";
65
66     orgOccurrences = [ 1 for i, line in enumerate ( lines ) if ( line.lower ( )[:3] == ORG_DIRECTIVE ) ];
67
68     if ( len ( orgOccurrences ) == 0 ):
69         raise ORGException ( "ORG directive is required." );
70
71     if ( len ( orgOccurrences ) > 1 ):
72         raise ORGException ( "Multiple ORG directives are not allowed." );
73
74     orgDirectiveIndex = orgOccurrences[0];
75     orgDirective = lines[orgDirectiveIndex];
76     orgValues = orgDirective.split ( " " )[1:];
77
78     if ( len ( orgValues ) == 0 ):
79         raise ORGException ( "ORG directive must have a value." );
80
81     if ( len ( orgValues ) > 1 ):
82         raise ORGException ( "ORG directive must have only one value." );
83
84     orgValue = orgValues[0];
85
86     try:
87         orgValue = int ( orgValue );
88     except ValueError:
89         raise Exception ( "ORG directive must have an integer value." );
90
91     if ( not ( 0 <= orgValue <= MAX_ADDRESS ) ):
92         raise Exception ( f"ORG directive value must be a valid address between 0 and {MAX_ADDRESS}." );
93
94     return ( orgValue, orgDirectiveIndex );
95

```

Слика 1. Пример дела кода за конфигурацију програмског додатка *CodeRunner*

Увиђајући простора за даља унапређења осталих програмерских курсева, сличан систем је креиран и за оцењивање програмских кодова написаних од стране студената на програмском језику Пајтон. Овај систем је такође реализован у програмском језику Пајтон. Као и претходно поменути систем, и овај систем је показао високу прецизност приликом процеса оцењивања. Коришћење овог система спроводи се на курсевима Програмирање 1 и Практикум из програмирања 1 [22].

Последично, развијен је још један систем за аутоматизовано оцењивање у оквиру *Moodle* платформе за електронско учење. Имплементирани систем испољава способност аутоматизованог оцењивања програмских кодова написаних на симболичком машинском језику за архитектуру Пико компјутер [23]. Овај систем је такође написан на програмском језику Пајтон. Слика 2 приказује пример оваквог питања.

Write a program in an assembly language for picoComputer architecture that calculates the arithmetic mean of an array. The first number in the input data represents the array length, while the rest of the input represent array elements.

For example:

Input	Result
7	8
19	
7	
1	
12	
5	
8	
21	

Answer: (penalty regime: 0 %)

Reset answer

```

1  n = 1
2  arr = 2
3  i = 3
4  mean = 4
5  arrAddr = 100
6
7  org 8
8
9  in n
10 mov arr, 4arrAddr
11 in (arr), n
12 lgt 0, n, end
13 mov i, 0
14 mov mcon, 0
15
16 lgt n, 1, loop_end
17 loop: add mean, mean, (arr)
18 add arr, arr, 1
19 add i, i, 1

```

Proven

Слика 2. Пример *CodeRunner* питања са кодом написаним на симболичком машинском језику за *pC*

Од сва три претходно наведена система, последње наведени систем је било најтеже реализовати, јер је било неопходно написати код који проверава синтаксну, али и семантичку исправност програма писаних на симболичком машинском језику. Поред тога требало је реализовати и генератор кода чијим ће се извршавањем вршити промене стања програмиране машине. С друге стране, системи за аутоматизовано оцењивање програмских кодова написаних на програмским језицима С и Пајтон користе већ готове преводиоце написаног изворног кода и генераторе извршног кода за циљану машину.

Иако се кроз студентских испитивања на предметима Програмирање 1 и Програмирање 2 обавља путем рачунара, одређени делови предмета у погледу испитивања и оцењивања студената и даље се спроводе путем папирних тестова. Ово је неопходно, јер се та врста испитивања мора обавити за све студенте у исто време. Међутим, постоји мали број потребних рачунарских ресурса у лабораторијама, односно доступних слободних термина у њима, да би могли да се сместе сви пријављени студенти на предмет у једном тренутку. Стога, постоји потреба да се аутоматизује оцењивање и овог вида испитивања кандидата.

Ови тестови су задати у форми једног листа папира који садржи текст питања и њихове понуђене одговоре. Питања се налазе на једној страници листа, док се шаблон са понуђеним одговорима питања и упутство за њихово попуњавање налазе на другој страници листа. Овај формат теста се прави коришћењем обичног процесора текстуалног документа као што је Мајкрософтов софтвер Ворд (енг. *Microsoft Office Word*).

Шаблон са понуђеним одговорима дат је у форми правоугаоника који се састоји из три мања правоугаоника распоређених један испод другог у једној колони. У горњем правоугаонику налази се простор за упис идентификационог броја студента (индекса). Индекс је задат у форми шест седмосегментних цифара, које означавају четири цифре броја индекса и две цифре за последње две цифре године уписа студента. Ове податке студент попуњава зацрпљивањем сегмената цифара који одговарају његовом индексу.

Средишњи правоугаоник садржи матрицу понуђених одговора питања. Понуђени одговори су задати у виду матрице (мреже) кругова, при чему се свака врста кругова ове матрице односи на једно питање, а појединачни кругове те врсте на одговоре тог питања. Кандидат попуњава изабране одговоре потпуним зацрпљивањем круга у врсти која се односи на одговарајуће питање и колони која се односи на одговарајући одговор тог питања.

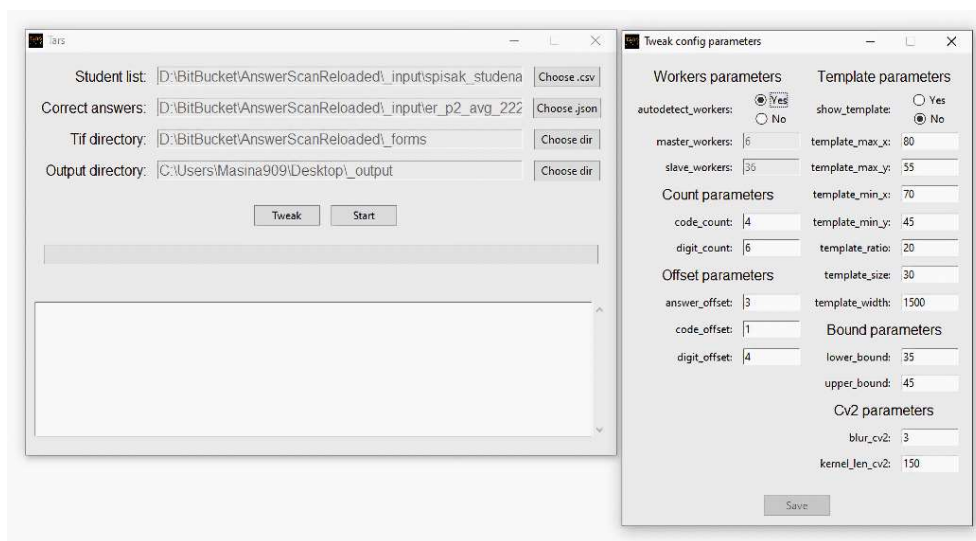
Доњи правоугаоник садржи кодирану варијанту теста. Тест се задаје у више различитих варијанти, при чему се оне разликују у распореду питања и њихових одговора за сваку од доступних варијанти. Редни број варијанте теста кодиран је бинарно, круговима који означавају по један бит вредности варијанте, тако да попуњени круг означава вредност 1, а празни круг вредност 0.

На тесту постоји и један црни квадрат, који се налази у врху простора шаблона теста. Њега студенти треба да попуне, уколико су направили неку грешку у попуњавању шаблона или су променили свој изабрани одговор. На тај начин сигнализирају систему да је након процесирања тог теста потребно ручно прегледати тај тест ради отклањања грешке. Ово се ретко дешава, јер углавном постоји довољан број резервних образаца за замену.

Цео процес спровођења испитивања студената тестовима на папиру реализован је у неколико фаза. Након што се изврши дизајнирање једне варијанте теста, уз помоћ софтвера названог CASE (енг. *Clever Answers Shuffling Environment*) и оригиналног документа насталог у фази дизајна теста врши се генерисање различитих варијанти теста. Као производ ове фазе настају документ са различитим варијантама теста и матрица тачних одговора за сваку од генерисаних варијанти теста.

Затим се варијанте теста у фази штампања теста штампају у копијацији у потребном броју примерака. Након тога се приступа изради теста у току којег кандидати врше попуњавање шаблона теста у фази израде теста. После тога се тако попуњени тестови односе у скенинг центар где се врши њихово скенирање у фази скенирања, како би се добио дигитални облик теста. Појединачни тестови из сваке од сала спајају се у по једну датотеку која се односи на сваку салу.

Затим се над тако добијеним дигиталним обликом попуњених папирних тестова врши процес аутоматског препознавања јединственог идентификатора студента, њихових одговора и варијанте теста. Ово се дешава у фази оцењивања тестова коришћењем софтверског система названог *TARS* (енг. *Test Answers Recognition Software*). Излаз ове фазе представља аотирана варијанта теста заједно са резултатима кандидата. Слика 3 приказује главни прозор овог система.



Слика 3. Главни екран софтверског система *TARS*

Систем поседује способност да назначи да постоје грешке у процесирању теста. Може се десити да се не препозна шаблон на тесту, да се не препозна уписани индекс студента, да се препознати индекс не налази на списку студената, да се не препозна одговор, да се не може са сигурношћу утврдити да ли је одговор изабран, као и да се не може препознати варијанта теста. Стога, у фази прегледа добијених резултата, уколико је то потребно, морају се ручно извршити корекције за ове грешке. Ово углавном захтева мало времена, јер се грешке ретко дешавају и лако се исправљају.

Оба софтверска система, и *CASE* и *TARS* развијени су у оквиру Електротехничког факултета Универзитета у Београду. Софтверски систем *TARS* је послужио као добра полазна тачка за реализацију напреднијег предложеног система за аутоматизовано оцењивање папирних тестова. Такав систем треба да оцењује папирне тестове сложеније структуре са различитим врстама питања. Из наведених разлога спроведена су истраживања која су резултовала у реализацији новог софтверског система, који је превазишао наведене проблеме и представља решење аутоматског оцењивања произвољног папирног теста.

Ово истраживање има значајну примену у области образовања и тестирања, олакшавајући процес оцењивања папирних тестова и узимајући у обзир изазове који иду уз овај посао. Један од најбитнијих аспеката је што овај систем може значајно убрзати поступак оцењивања, ослобађајући прегледаче од великог обима посла и омогућавајући им да се фокусирају на анализу и дубље разумевање резултата тестирања.

Такође, значај овог истраживања потврђује и актуелност области аутоматизованог оцењивања, која и даље привлачи пажњу и има широк потенцијал за развој. Велики број садашњих истраживања и савремених резултата у овом домену сведоче о томе да се наставља рад на побољшању и унапређењу система аутоматизованог оцењивања папирних тестова. Само присуство великог броја референци из ове области у литератури доказује њену релевантност и актуелност.

Полазне хипотезе овог истраживања заснивале су се на вишегодишњем искуству у области аутоматизованог оцењивања папирних тестова:

- Аутоматизација процеса оцењивања папирних тестова може допринети бољим перформансама, квалитету и резултатима самог процеса.
- Могуће је извршити класификацију питања која се могу појавити на папирним тестовима.
- За сваку од идентификованих класа питања могу се утврдити карактеристике које доприносе бољој реализацији процеса аутоматизованог оцењивања папирних тестова.
- За сваку од идентификованих класа питања могуће је реализовати алгоритам користећи технике вештачке интелигенције који ће постићи боље резултате у оцењивању папирних тестова него човек.
- Могуће је реализовати софтверски систем за аутоматизовано оцењивање папирног теста различите структуре и сложености са задовољавајућим перформансама.

На самом почетку је извршена детаљна анализа врсте питања које се користе на папирним тестовима и реализована је класификација уз увођење сопствене номенклатуре. Затим је извршена анализа трансформација слике које је неопходно применити над дигиталним обликом теста. Поред тога, извршена је анализа метода детекције регуларних облика. Додатно, анализирана је детекција невалидно попуњених тестова. Установљена је метрика на основу којих је извршена детаљна евалуација квалитета постојећих система, као и самог реализованог система.

Остварени доприноси овог рада су:

- Анализа распрострањености извођења испитивања кандидата путем папирних тестова.
- Преглед и класификација најчешће коришћених питања на тестовима.
- Преглед и анализа постојећих система за аутоматизовано оцењивање тестова.
- Класификација постојећих система за аутоматизовано оцењивање тестова према унапред утврђеном шаблону.
- Имплементација система за аутоматизовано оцењивање тестова користећи технике вештачке интелигенције.
- Опсежна евалуација имплементираних система над великим скупом папирних тестова у дигиталном облику.
- Поређење добијених резултата са резултатима релевантних истраживања.

Разноврсност тестова нам нуди палету различитих питања – од оних са могућношћу вишеструког избора, преко оних која захтевају спајање концепата који се подударају, до оних питања која изискују кратке одговоре, попуњавање дужим текстом итд. Стога, потребно је прикупити врсте питања која се користе у тестовима, анализирати суштину онога што таква

питања испитују и на основу тога извршити њихову класификацију уз увођење сопствене номенклатуре. Класификацију је, између осталог, неопходно урадити и зато што многе институције које спроводе тестирања кандидата користе суштински исти тип питања, али уводе различиту конвенцију за њихово именовање, те је поред класификације најчешће коришћених питања на тестовима потребно извршити обједињавање једним називом оних типова питања који заправо представљају тип питања исте врсте. Поред увођења класификације питања, биће објашњен и сваки тип питања, као и шта је мотивација за постављање таквог типа питања на тесту. Ово ће бити покривено поглављем број 2 ове докторске дисертације.

Такође, потребно је фокусирати се и на постојећа решења у овој области, а на основу претходно предложене нове класификације питања биће категоризована постојећа решења која пружају могућност аутоматизованог оцењивања тестова. Биће спроведена исцрпна анализа постојећих решења од интереса у области аутоматизованог оцењивања тестова, која за сваки систем подразумева навођење: сврхе система, структуре система, технологија у којима је систем реализован, алгоритама које систем примењује, области вештачке интелигенције које систем користи, уочених предности и недостатака система, категоризације система према способности да детектује и оцењује претходно класификоване типове питања, као и ограничења под којима систем ради и његове коначне евалуације. Ово ће бити покривено поглављем број 3 ове докторске дисертације.

Циљ ове докторске дисертације је предочавање детаља реализованог система за аутоматизовано оцењивање папирних тестова реализацијом сопствених алгоритама који се наслањају на области вештачке интелигенције: машинско учење и компјутерску визију. Систем је реализован у програмском језику који пружа подршку за машинско учење и компјутерску визију и даје најбоље перформансе. Систем као улаз добија слике страница папирних тестова у дигиталном облику над којима најпре примењује алгоритме трансформације слике, а затим комбиновањем сопствених и доступних алгоритама за детекцију региона од интереса и оцењивања врши њихову детекцију и оцењивање одређене класе питања. Такође, систем као улаз добија конфигурабилну датотеку која у себи садржи структуру папирног теста, на основу које систем може извршити проверу успешности процеса детекције. Имплементирани систем, као резултат, поред оцењивања одређене класе питања врши и издвајање свих питања, као и класификацију невалидних тестова проузрокованих неисправним (од стране кандидата) попуњавањем одговора на питања. Циљ система је да пружи могућност поуздане детекције питања и њихових одговора на тесту приложеном систему у дигиталном облику и њиховог оцењивања у што краћем временском року. Ово ће бити покривено поглављем број 4 ове докторске дисертације.

Уз све ово, извршено је истраживање предложеног модела путем евалуације на обимном узорку попуњених папирних тестова, који су систему достављени у дигиталном формату и који су попуњени конвенционалним средствима, као што су обичне или хемијске оловке. Циљ овог истраживања био је да се процени прецизност система у детекцији и оцењивању различитих типова питања које је систем дизајниран да оцењује. Током евалуације проучаване су и ситуације у којима систем није успео да изврши оцењивање са задовољавајућом прецизношћу. Иако су такви случајеви били ретки, од изузетног значаја је разумети зашто су се десили. На тај начин, биће пружен увид у изазове са којима се систем може сусрести. Овакве анализе имају за циљ да унапреде будуће развојне напоре и омогуће још боље резултате у аутоматском оцењивању папирних тестова. Такође, извршено је поређење реализованог система са постојећим системима који припадају истој класи питања, као и са системима који оцењују друге класе питања, у оној мери у којој се наведено поређење може спровести. Додатно, биће установљене метрике на основу којих ће се вршити детаљна евалуација квалитета постојећих система, као и самог реализованог система. Такве метрике могу бити прецизност приликом детекције питања и одговора, као и брзина која је неопходна за обраду једног питања или једне

странице теста. Додатно, одређени системи могу уводити ограничења која могу утицати на формирање тестова, као што је неопходност раздвајања питања и одговора на одвојене обрасце. Сви наведени аспекти реализованог софтверског система ће се разматрати у поглављу број 5 ове докторске дисертације.

На самом крају, на основу добијених резултата биће изведени закључци о успешности истраживања и могућностима даљег унапређења реализованог система. Такође, биће разматране и смернице за даље истраживање у овој области. Последња глава ове докторске дисертације има за циљ да сумира истраживачки рад и да пројектује потенцијалне правце којима се може кренути ка даљем напретку на овом пољу. Ово ће бити покривено поглављем број 6 ове докторске дисертације.

2. Класификација најчешће коришћених питања на папирним тестовима

Папирни тестови обухватају разноврстан низ типова питања, као што су вишеструки избор, тачно-нетачно питања, подвлачења, повезивања, кратак одговор, есеји итд. Набрајање сваке потенцијалне врсте питања која се појављује на папирним тестовима у процесу испитивања представља значајан изазов. Овој сложености додатно доприноси присуство синонимних термина који у суштини означавају исти тип питања.

Такође, неки типови питања се делимично преклапају са неким другим типовима питања или представљају подскуп/надскуп неке друге врсте питања. Међутим, у сврху омогућавања транспарентне и упоредне евалуације имплементираних софтверских система који испољавају могућност аутоматизованог оцењивања папирних тестова, постаје кључно класификовати различите типове питања и консолидовати повезане типове питања у кохезивне класе. У процесу истраживања идентификоване су следеће класе питања чије оцењивање може бити аутоматизовано:

- Вишеструки избор: Кандидати бирају један или више тачних одговора из матричног распореда избора, обично представљених круговима или квадратима.
- Повезивање: Кандидати успостављају везе између повезаних концепата (нпр. спајањем, дописивањем слова, цифара или осталих симбола) или подвлаче специфичне одговоре.
- Кратак одговор: Кандидати дају сажете одговоре у облику једне или неколико речи текста или бројчаних вредности, без потребе за опширним тумачењем писаног садржаја.
- Есеј: Кандидати састављају опширне писмене одговоре, који се састоје од неколико реченица или параграфа, наглашавајући свеобухватно разумевање и артикулацију основних концепата током процеса евалуације.

Ове класе питања су представљене растућим редоследом сложености, с обзиром на технологије потребне за њихово ефикасно решавање. У почетку, имплементирани систем се фокусирао искључиво на аутоматизацију евалуације питања са вишеструким избором одговора. Међутим, каснији развој је омогућио проширење система како би обухватио аутоматизовано оцењивање одговарајућих питања, где су тачни одговори подвучени или се врши дописивање симбола на предвиђеним местима. На самом крају, систем је проширен како би обухватио и аутоматизовано оцењивање питања кратких одговора која укључују нумеричке одговоре.

Предвиђено је да ће се будуће итерације система даље развијати како би обухватиле способности евалуације у свим горе поменутих класама питања. На тај начин би се аутоматизовао цео процес оцењивања папирних тестова. У наставку ће у сваком потпоглављу бити описане идентификоване класе питања.

2.1. Вишеструки избор

Вишеструки избор је један од најчешће коришћених типова питања на папирним тестовима у процесу испитивања. Састоји се од текста питања и делимичне или потпуне изјавне реченице, која се назива основа (енг. *stem*) и више понуђених одговора који се називају избори (енг. *choice*). Избор питања са више одговора састоји се од једног или више тачних и преосталих нетачних одговора који се називају дистрактори [24].

Обично се испред сваког понуђеног одговора налази типски геометријски облик. Овај геометријски облик представља по један од доступних одговора за избор и најчешће је

представљен кругом или квадратом. Слика 4 приказује пример оваквог питања са круговима за означавање понуђених одговора.

4. задатак		101127010301	број бодова: 2
Тема „Горског вијенца” Петра II Петровића Његоша је истрага потурица. Владика Данило се колеба када треба да одлучи да ли се обрачуна са потурицама. Обојте кружић испред тачног одговора.			
<input type="radio"/>	Плаши се славних турских ратника.		
<input type="radio"/>	Свестан је да ће жртве бити огромне.		
<input type="radio"/>	Не жели да ратује против свог народа.		
<input type="radio"/>	Сумња у снагу и храброст своје војске.		
<input type="radio"/>	Плаши се да ће се племена међусобно истребити.		

Слика 4. Пример стандардног питања класе „Вишеструки избор“

У неким ситуацијама, понуђени одговори представљени су табелом. У заглављу табеле са горње и леве стране налази се текст појединачних избора. Одговори су представљени мрежом изабраних типских геометријских облика, као што илуструје Слика 5.

I. задатак		101112010201	број бодова: 1	
Повежи наслов књижевног дела са именом његовог аутора. Име једног аутора је вишак. Обој кружић испред тачног одговора.				
	Подне	Међу својима	Отаџбина	Крвава бајка
Десанка Максимовић	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Јован Дучић	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Ђура Јакшић	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Бранко Радичевић	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Владислав Петковић Дис	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Слика 5. Пример табеларног питања класе „Вишеструки избор“

Штавише, редослед понуђених одговора у питању може се разликовати од једне до друге варијанте истог питања. Овим се обично прибегава у ситуацији када постоји велики број кандидата који полажу тест или су распоређени у згуснутијем распореду седења. Такође, треба напоменути и да су питања тачно-нетачно, која се обично састоје од изјавне реченице, сврстана у ову класу питања, као што илуструје Слика 6.

7. задатак		101212010101	број бодова: 1
Ако је тврдња тачна, обојте кружић у колони Тачно , а ако тврдња није тачна, обојте кружић у колони Нетачно .			
Тврдња	Тачно	Нетачно	
Вук је слово ј узео из старих рукописа.	<input type="radio"/>	<input type="radio"/>	
Вук је објавио три речника српског језика.	<input type="radio"/>	<input type="radio"/>	
Пре Вука ћирилицу је реформисао Саво Мркаљ.	<input type="radio"/>	<input type="radio"/>	
Вук је говорио источнохерцеговачким дијалектом.	<input type="radio"/>	<input type="radio"/>	

Слика 6. Питање типа тачно-нетачно сврстано у класу „Вишеструки избор“

Да би исправно одговорио на овакав тип питања, кандидат треба да изабере одговарајући одговор, тако што ће у потпуности зацртати геометријски облик који се односи на текст изабраног понуђеног одговора. У неким ситуацијама прихвата се и да кандидат изврши уписивање плусева или крстића у геометријски облик или чак и заокруживање око геометријског облика. Као резултат овог начина попуњавања, кандидати могу веома брзо да одговоре на овај тип питања.

Иако је ово врло једноставан тип питања, може се применити када се од испитиваног кандидата захтева одзив знања (присећање, енг. *recall*), примена знања (енг. *application*), анализа градива (енг. *analysis*), али и синтеза знања (енг. *synthesis*). Због тога се ове врсте питања генерално користе када је важно да се тестира широк спектар знања које се испитује [25]. Предности и недостатке оваквог типа питања наводи Табела 1.

Табела 1. Приказ предности и недостатака питања класе „Вишеструки избор“

Предности	Недостаци
Оцењивање од стране оцењивача може да се спроведе релативно брзо.	Ручно оцењивање је и даље много спорије него време потребно арбитарном софтверском решењу. Поред тога, ручно оцењивање је подложно грешкама узрокованим несмотреношћу или умором.
Прецизна формулација питања и одговора.	Присуство кључних речи у питању захтева од кандидата само пуко познавање знања које се испитује.
Понуђени одговори покривају широк спектар знања које се испитује.	Може одузимати превише времена састављачима теста да изврше дизајнирање погрешних одговора.
Одговори се веома брзо могу попунити од стране кандидата.	Може да олакша варање од стране кандидата на тестовима са оваквим типовима питања или да унесе дилему код кандидата да ли је одговор исправно попуњен или не у случају великог броја присутних питања и њихових одговора.

Софтверски системи који решавају овакав тип питања треба да детектују локације геометријских симбола који представљају понуђене одговоре. Након тога треба да утврде да ли су симболи зацртани или заокружени или попуњени на тражени начин. Потом треба да утврде да ли су детектовани одговори у складу са кључем теста. Аутоматско оцењивање оваквог типа питања захтева коришћење алгоритама компјутерске визије за обраду слике и алгоритме за детекцију облика питања [26]. За детекцију облика понуђених одговора, што су често кругови, користе се различите врсте алгоритама [27] [28] [29] [30].

2.2. Повезивање

Повезивање је уобичајена врста питања која се среће на папирним тестовима. Ова врста питања пружа најефикаснији начин да се тестира познавање односа између више понуђених концепата. Обично је питање подељено на два дела, при чему се на једној страни налазе изјавне реченице или термини, док се на другој страни налазе одговарајуће изјавне реченице, термини или избори које треба спојити са концептима на првој страни [31].

Обично се испред термина са једне стране налазе означена места за упис одговора, која могу бити представљена у виду линија. Испред термина са друге стране се могу наћи мала или

велика слова, која треба да се допишу на одговарајуће линије за упис одговора. Пример оваког питања илуструје Слика 7.

7. задатак		101216010101	број бодова: 1
Повежите реч из леве колоне са њеним значењем тако што ћете на линије уписати одговарајућа слова која стоје испред значења.			
7.1.			
<input type="checkbox"/>	неразуман	A. који се не може уразумити	
<input type="checkbox"/>	неразумљив	B. који се не може разумети	
7.2.			
<input type="checkbox"/>	фасцинантан	A. који је очаран	
<input type="checkbox"/>	фасциниран	B. који очарава	
7.3.			
<input type="checkbox"/>	ефективан	A. који се не може уразумити	
<input type="checkbox"/>	ефектан	B. који се не може разумети	

Слика 7. Пример питања класе „Повезивање“ са уписом слова

Број термина са обе стране ове врсте питања не мора да буде исти. У неким случајевима је потребно на линијама за упис одговора дописати арапске или римске бројеве. Пример оваког питања приказује Слика 8.

6. задатак		101211010701	број бодова: 2
Испред функције језика напишите број који стоји испред одговарајућег значења.			
Функција језика:		Значење:	
<input type="checkbox"/>	Акумулативна функција	1. Језик је средство споразумевања међу људима.	
<input type="checkbox"/>	Експресивна функција	2. Језик пружа могућност измештања у простору и времену.	
<input type="checkbox"/>	Комуникативна функција	3. Језик је средство којим се изражавају осећања.	
		4. Језик чува и преноси знања са једне генерације на другу.	

Слика 8. Пример питања класе „Повезивање“ са уписом цифре

У одређеним случајевима питање се састоји само из једног дела. Тада постоји само једна група термина, а повезивање је одређено релацијом између понуђених термина. Пример оваког питања приказује Слика 9.

10. задатак		101226010701	број бодова: 2
Поређајте (књижевно)уметничке епохе хронолошки од најстарије (1) до најмлађе (5) тако што ћете на црте уписати бројеве.			
<input type="checkbox"/>	романтизам		
<input type="checkbox"/>	барок		
<input type="checkbox"/>	класицизам		
<input type="checkbox"/>	реализам		
<input type="checkbox"/>	ренесанса		

Слика 9. Пример питања класе „Повезивање“ са одређивањем редоследа

Понекад се упаривање ових термина може вршити и на супротан начин. Поред тога, у неким ситуацијама питање је дато у табеларном формату, где заглавља табеле која се налазе са две суседне стране табеле садрже понуђене термине, а дописивање симбола је неопходно извршити у матричним пољима. Пример оваког питања показује Слика 10.

15. задатак		101226010703		број бодова: 2		
Повежи наслов књижевног дела са јунаком из тог дела уписом симбола х. Име једног јунака је вишак.						
	Благоје казанџија	Миона	везир Јусуф	Андреас Сам	Малчика	Миле Врабац
Ноћ и магла						
Све ће то народ позлатити						
Прва бразда						
Избирачица						
Мост на Жепи						

Слика 10. Пример питања класе „Повезивање“ са матричним уписом одговора

Питања типа вишеструки избор са предефинисаним просторима за упис одговора, при чему кандидати треба да упишу изабрану цифру, слово или симбол су такође категоризована у овај тип питања. Разлог за ову одлуку је што се са повећањем броја предефинисаних простора за упис одговора, она врло лако могу генерализовати типом питања повезивања. Штавише, такав тип питања је ближе повезан са овим типом питања, него са типом питања вишеструки избор, с обзиром да се у питањима типа вишеструки избор одговори у већини случајева означавају регуларним облицима, као што су на пример квадрати, кругови, ромбови итд.

У одређеним ситуацијама, овај тип питања је задат у виду краћег текста. У датом тексту потребно је подвлачењем одговарајућих делова текста означити изабране одговоре. Пример оваквог питања илуструје Слика 11.

5. задатак		101134010801		број бодова: 1	
Подвуците погрешно написану реч у реченици. Претходне недеље није било састанка, па је председник одељења позвао све ученике на договор.					

Слика 11. Пример питања класе „Повезивање“ са подвлачењем речи

У неким ситуацијама, овај тип питања је задат у виду дужег текста. Тачни одговори, који су представљени текстом који је потребно подвући, често се протежу у више линија текста. Пример оваквог питања приказује Слика 12.

12. задатак		101235010202		број бодова: 1	
Прочитајте одломак из текста <i>Језички знак, њојам, ознака њојма</i> Фердинанда де Сосира, па подвуците део реченице који објашњава општу карактеристику друштвеног закона. Често се говори о законима у лингвистици. Да ли се језичке чињенице збиља сврставају на основу закона, и иако је тако, какве су природе ти закони? Пошто је језик друштвена институција може се <i>a priori</i> мислити да је он регулисан прописима аналогним онима по којим се друштвени закони управљају. Међутим, сваки друштвени закон има две основне карактеристике: императивну и општу. Закон се намеће и захвата све случајеве у одређеном времену и на одређеном простору.					

Слика 12. Пример питања класе „Повезивање“ са подвлачењем већег дела текста

Штавише, у ретким ситуацијама ова врста питања не дефинише места за упис одговора, већ понуђени термини морају бити упарени испрцавањем линија које их повезују. Ипак, овакав тип питања се обично избегава због шуме испресецаних линија које настају као резултат попуњавања одговора од стране кандидата. Ово у значајној мери отежава и ручни начин прегледања овог питања.

Да би исправно одговорио на овај тип питања, кандидат најпре треба да прочита изјавне тврдње које се налазе на једној страни питања. Затим треба да на другој страни питања пронађе један или више понуђених термина који се односе на прочитану тврдњу. На крају треба да споји изабране концепте у складу са назначеним правилима за попуњавање оваквог типа питања.

Као резултат ових правила питања, кандидати могу да одговоре на ову врсту питања поприлично брзо, али нешто спорије у односу на време потребно за одговарање на питања са вишеструким избором. Ова врста питања се обично користи када је потребно на нешто вишем и детаљнијем нивоу тестирати знање кандидата. Предности и недостатке ове врсте питања наводи Табела 2.

Табела 2. Приказ предности и недостатака питања класе „Повезивање“

Предности	Недостаци
Ручно оцењивање од стране оцењивача може да се изврши релативно брзо, али захтева нешто више времена у односу на тип питања вишеструки избор.	Ручно оцењивање је и даље далеко спорије него уз помоћ било ког доступног софтверског решења способног да оцењује ову врсту питања.
Смањује могућност случајног избора одговора у случају непознавања испитиваног градива.	Захтева само познавање односа између података наведених на обе стране овог типа питања.
Покрива велику количину знања које се испитује уз значајно смањен утршак простора које овај тип питања заузима на тесту.	Коришћењем овог типа питања продужава се потребно време за решавање целокупног теста.
Одговори се релативно брзо могу попунити од стране кандидата, али нешто спорије у односу на исту количину испитаног градива користећи тип питања вишеструки избор.	У случају да су правила решавања питања таква да се линије користе за повезивање сродних термина, велики број понуђених термина може увести дилему код кандидата у погледу исправности уписаног одговора.

Софтверски системи који решавају овакав тип питања треба да детектују локацију простора за упис одговора. Након тога треба да утврде где су границе уписаног одговора у оквиру раније детектованог простора. Затим треба да изврше детекцију зацрњености у случају подвлачења одговора, односно одреде припадност уписаног симбола једној од могућих класа, у случају уписивања симбола одговора. Потом треба да утврде да ли су детектовани одговори у складу са кључем теста. Аутоматско оцењивање оваквог типа питања захтева коришћење компјутерске визије [32], алгоритме класификације [33] или неуралне мреже за предикцију написаног симбола [34].

2.3. Кратак одговор

Тип питања кратак одговор се често среће под називима „питање попуњавање“ или „питање допуњавање“. Овај тип питања представља корисно средство за процену знања кандидата и разумевање фундаменталних принципа градива које се испитује. Типично се ово

питање састоји од краћег или нешто дужег текста иза којег следи простор који садржи линију за упис кратког одговора. Пример овог типа питања приказује Слика 13.

3. задатак	101121010801	број бодова: 1
Прочитајте основне биографске податке о писцу, па напишите његово име на линији испод текста.		
Рођен је у Шапцу. Живео је у другој половини XIX века. Творац је психолошког реализма у српској књижевности. По занимању је био лекар, психијатар. Иако је написао само девет приповедака, важи за једног од наших најбољих приповедача. Познат је по приповеткама „Све ће то народ позлатити” и „Први пут с оцем на јутрење”. Умро је у Београду.		
Име и презиме писца: _____		

Слика 13. Пример питања класе „Кратак одговор” са једном линијом за упис одговора

Понекад се овај тип питања састоји од неколико кратких изјава или нешто дужег текста распоређеног у неколико параграфа. Након сваке подцелине следи простор представљен линијама на којима је потребно дописати одговоре. Пример овог типа питања илуструје Слика 14.

3. задатак
У складишту је 3 t робе, при чему је 12,5% неисправно.
а) Колика је маса неисправне робе, у килограмима?
Маса неисправне робе је _____ kg.
б) Од укупне масе неисправне робе 76% може да се рециклира. Колико килограма неисправне робе може да се рециклира?
Може да се рециклира _____ kg неисправне робе.
в) Цена исправне робе на тржишту износи 120 динара по килограму. За један килограм рециклиране неисправне робе добије се 45 динара. Ако би се продала сва исправна роба и рециклирала сва роба која се може рециклирати, колико би укупно динара добили?
Добили би укупно _____ динара.

Слика 14. Пример питања класе „Кратак одговор” са више линија за упис одговора

Поред тога, ова врста питања често може бити прожета вишеструким предефинисаним местима у виду линија на које је потребно дописати текст тачног одговора. Ипак, и у таквим случајевима, тип питања кратак одговор успева да одржи одређени ниво структурираности. На тај начин се кандидату пружа довољна доза флексибилности приликом одговарања на овај тип питања.

Да би исправно одговорио на ову врсту питања, кандидат треба да попуни празна места уписивањем фразе у траженом формату. Кандидат треба да осмисли тачан одговор, а не само да изврши пуко повезивање њему познатих појмова или да препозна неки од понуђених одговора као тачан одговор. Одговор који треба дописати у оваквом типу питања обично није ограничен и може да варира у дужини. Типично је тачан одговор у овој врсти питања представљен бројем, речју или кратком фразом од неколико речи.

Као резултат ових правила питања, кандидати могу да одговарају на ову врсту питања у релативно кратком времену, али значајно спорије у односу на претходно споменуте врсте питања као што су вишеструки избор и повезивање. Ова врста питања се углавном користи када је неопходно тестирати прагове знања кандидата на нешто већем нивоу у односу на претходно наведене врсте питања. Предности и недостатке ове врсте питања наводи Табела 3.

Табела 3. Приказ предности и недостатака питања класе „Кратак одговор“

Предности	Недостаци
Ручно оцењивање од стране човека може да се обави у релативно кратком временском периоду, али је спорије у односу на ручно оцењивање претходно наведених типова питања.	Флексибилност у одговарању на овај тип питања која се тиче редоследа речи одговора, синонимних одговора итд. може представљати значајан напор и за људе задужене за оцењивање оваквог типа питања.
Значајно смањује могућност насумичног погађања тачног одговора.	Да би успешно одговорили на овај тип питања, кандидати треба да запамте релативно малу количину информација.
Питања су донекле структурирана, али кандидатима омогућавају флексибилност у одговарању.	Оцењивање оваквог типа питања је захтевније у поређењу са претходно наведеним типовима питања.
Попуњавање одговора у овом типу питања може се релативно брзо урадити, али значајно спорије у поређењу са претходно описаним типовима питања.	Препознавање нечитко ручно написаног текста понекад може представљати изазов и за људе задужене за прегледање тестова.

Софтверски системи који решавају овакав тип питања треба да детектују локацију простора за упис одговора. Након тога треба да утврде где су границе уписаног одговора у оквиру раније детектованог простора. Затим треба да изврше предикцију уписаног одговора. Потом треба да утврде да ли су детектовани одговори у складу са кључем теста.

За решавање овог типа питања системи не морају да препознају релације између речи, јер је одговор на ово питање у виду једне речи или фразе од неколико речи. Ипак, уколико је то неопходно, могу се користити трансформери [35]. Аутоматско оцењивање оваквог типа питања захтева софистицираније технике вештачке интелигенције, као што је дубоко учење [36].

2.4. Есеј

Ова врста питања омогућава ангажовање аналитичког начина размишљања и критичког мишљења кандидата [37]. Типично се ово питање састоји од нешто дужег текста иза којег следи простор који садржи линије за упис одговора. Пример оваквог типа питања приказује Слика 15.

11. задатак	101227030001	број бодова: 2
У песми „Диоба Јакшића” браћа Дмитар и Богдан Јакшић деле наследство, а на крају се сукобљавају око коња и сокола. Објасните зашто.		
<hr/> <hr/> <hr/>		

Слика 15. Пример питања класе „Есеј“ са вишелинијским одговором

Ова врста питања представља сложенији упит који захтева од кандидата писани текстуални одговор, који може варирати у дужини од неколико реченица до неколико параграфа. На папирним тестовима често се среће у једном од следећих формата одговора: екстензивном, који за циљ има да испитује синтезу, или рестриктивном, који за циљ има да испитује анализу. Понекад питања овог типа садрже једно или више потпитања.

Овај тип питања је значајно мање структуриран у поређењу са претходним типовима питања. Мотивација за избор овог типа питања је да подстиче креативност кандидата и да омогући кандидатима да изразе свој начин разумевања испитиваног знања на значајно

флексibilнији начин. Овај тип питања пружа могућност за интеграцију испитиваног материјала на много иновативнији начин приликом састављања оваквог типа питања.

Да би исправно одговорио на овај тип питања, кандидат треба да напише донекле сложен одговор. Кандидат треба да исказе веома висок ниво знања и размишљања, који укључује анализу, синтезу и евалуацију. Као резултат правила овог типа питања, кандидатима је неопходно више времена да одговоре на овај тип питања.

Овај тип питања се најчешће користи онда када је важно да се тестира способност кандидата да осмишљава свој одговор. Приликом формулације одговора, кандидат свој одговор заснива на сопственом знању и разумевању научног градива. Предности и недостатке овог типа питања наводи Табела 4.

Табела 4. Приказ предности и недостатака питања класе „Есеј“

Предности	Недостаци
Питања поседују значајно умањен ниво структурираности, али омогућавају висок ниво флексibilности кандидатима у писању одговора.	Решавање овог типа питања је временски захтевно за кандидате.
Широка флексibilност кандидата у одговарању на овај тип питања омогућава људима квалификованим за ручно прегледање овог типа питања да постигну дубљи ниво разумевања за резонување кандидата приликом одговарања.	Већи ниво флексibilности омогућен кандидатима при одговарању на овај тип питања намеће већи терет на плећа људи квалификованих за ручно прегледање овог типа питања.
Подстиче кандидате да изврше интеграцију појединачних јединица знања у сложенију целину, као и способност кандидата да надграђује своје већ усвојено знање.	Потребне су још софистицираније технике вештачке интелигенције софтверским системима дизајнираним за оцењивање овог типа питања.
Пружа могућност људима квалификованим за оцењивање овог типа питања да темељно приступе процењивању нивоа разумевања кандидата за испитивано градиво.	Ручно оцењивање од стране људи квалификованих за процену знања је споро и подложно субјективности прегледача.

Софтверски системи који решавају овакав тип питања најпре треба да детектују локацију простора за упис одговора. Затим треба да утврде где су границе уписаног одговора у оквиру раније детектованог простора. Потом треба да изврше предикцију уписаног одговора. Након тога треба да утврде да ли су детектовани одговори у складу са кључем теста.

За решавање овог типа питања системи морају да препознају дубље релације између речи текста одговора. Релације између речи и њихов редослед дају смисао написаном одговору. Ово може бити изазован задатак јер се одговор на питање састоји од параграфа, реченица и речи. Поред тога, потребно је на неки начин детектовати целокупно написани текст на папиру пре саме обраде утврђеног текста. Аутоматско оцењивање оваквог типа питања захтева софистицираније технике вештачке интелигенције, као што су дубоко учење [38] и обрада природног језика [39].

Предложени софтверски систем за аутоматизовано оцењивање папирних тестова не поседује могућност процесирања ове класе питања. Развој система који ће решавати ову врсту проблема је у зачетку и тренутне технологије не омогућавају постизање тачности као код осталих система. Навођење ове класе питања је важно у погледу будућих истраживања и проширења система.

3. Преглед постојећих решења за аутоматизовано оцењивање папирних тестова

Након што је завршена класификација питања и обједињавање сродних типова питања истим типом, фокус истраживања постала је свеобухватна претрага софтверских система који испољавају могућност аутоматизованог оцењивања неких од претходно класификованих типова питања на папирним тестовима. Пре самог процеса исцрпног истраживања требало је идентификовати неколико критеријума који ће усмеравати и уобличити целокупан процес истраживања. Опсежно истраживање ових система спроведено је у оквиру литературе отвореног приступа.

Критеријуми филтрирања идентификованих софтверских система искључивали су системе документоване у радовима објављеним пре више од једне деценије, што обезбеђује релевантност овог истраживања. Критеријуми приоритета у избору софтверских система дали су предност оним решењима чији су радови цитиранији, што указује на њихову истакнутост у овој области. Критеријуми релевантности пронађених софтверских система дали су предност софтверским системима са недавним датумима објављивања, чиме се обезбеђује фокус на недавна унапређења. Такође, нису разматрана решења која нису бесплатно доступна као што су [40] и [41].

На самом почетку процеса аутоматизације оцењивања папирних тестова, најпре су имплементирани системи за аутоматизовано оцењивање искључиво питања са вишеструким избором. Такви системи су захтевали да питања буду дизајнирана тако да су раздвојени обрасци са текстом питања и обрасци са одговорима, који су се налазили на засебним листовима. Међутим, пракса раздвајања текста питања и њихових одговора на посебним формуларима уноси грешке у попуњавању формулара и одвлачи кандидате од суштине самог питања. Стога је пожељна карактеристика система да испољава могућност евалуације тестова са овом врстом питања у којима су текст питања и текст одговора обједињени.

С обзиром да се појавила потражња за аутоматизованим оцењивањем различитих типова питања, осмишљена су нова решења посебно прилагођена сваком појединачном типу питања. Штавише, током истраживања система способних за аутоматизовано оцењивање папирних тестова постало је очигледно да је сваки систем дизајниран са циљем да евалуира тачно један тип питања. Ова специјализација истраживаних система олакшала је идентификацију значајних сличности међу софтверским системима који су способни да аутоматски оцењују исти тип питања, тј. који припадају истој класи. Без обзира на то, чак и међу софтверским системима који припадају истој класи, постоје различити приступи у детекцији питања и оцењивању њихових одговора, чак и када користе идентичне технологије.

Битно је нагласити да, иако су током истраживања идентификовани системи који испољавају могућност аутоматизованог оцењивања питања типа есеј на папирним тестовима, да су они изостављени из овог прегледа. Развој система који ће решавати ову врсту проблема је у зачетку и тренутне технологије не омогућавају постизање тачности као код осталих система. Имајући ово у виду, на тај начин је могуће обавити коректно поређење имплементираног система са осталим идентификованим системима способним за аутоматизовано оцењивање једног од типова питања као и имплементирани систем.

За сваки од одабраних софтверских система најпре је приказано неколико уводних информација. Ове информације укључују назив софтверског система, афилијације аутора софтверског система, тип питања које софтверски систем може да оцењује, језик који се користи на папирним тестовима, методе коришћене за реализацију софтверског система, као и годину документовања рада који описује софтверски систем. Касније ће бити предузета анализа њихове прецизности, перформанси и ограничења о којима ће се расправљати.

Изабрани софтверски системи су развијени од стране аутора чије су афилијације: Катедра за рачунарску технику и информатику, Електротехнички факултет Универзитета у Београду (*TARS*); Факултет за инжењерство, Универзитет „Едит Кован“, Перт, Аустралија (енг. *School of Engineering, Edith Cowan University, Perth, Australia*) (*MCTQF*); Универзитет „Де Ла Сале“, Филипини (енг. *De La Salle University, Philippines*) (*MCG-RAS*); Катедра за електронику и телекомуникације, Кеј Си колеџ за студије инжењерства, менаџмента и истраживања, Тејн, Махараштра, Индија (енг. *Department of Electronics and Telecommunication, K.C. College of Engineering & Management Studies & Research, Thane, Maharashtra, India*) (*AMCG*); Катедра за телематичко инжењерство, Универзитет „Карлос трећи Мадридски“, Мадрид, Шпанија (енг. *Department of Telematic Engineering, University Carlos III of Madrid, Madrid, Spain*) (*Eyegrade*); Факултет за електронику и информационо инжењерство, Универзитет Фошан, Фошан, Кина (енг. *School of Electronic and Information Engineering, Foshan University, Foshan, China*) (*ASSHEP*); Универзитет „Принц Мохамед бин Фад“, Ал Кобар, Саудијска Арабија (енг. *Prince Mohammad bin Fahd University, Al Khobar, Saudi Arabia*) (*AGHAS*); Факултет електротехнике и рачунарства Универзитета у Загребу (*ARHC*); сарадња Токијског универзитета за агрокултуру и технологију, Токио, Јапан и Националног центра за универзитетске пријемне испите, Токио, Јапан (енг. *Tokyo University of Agriculture and Technology, Tokyo, Japan; The National Center for University Entrance Examinations, Tokyo, Japan*); сарадња Факултета за софтвер, Универзитет технологије, Гванџу, Кина и Факултета медицинског информационог инжењерства, Универзитет кинеске медицине у Гванџуу, Гванџу, Кина (енг. *School of Software South China, University of Technology Guangzhou, China; College of Medical Information Engineering, Guangzhou University of Chinese Medicine, Guangzhou, China*) (*ISSHSA*); сарадња Индијског статистичког института, Калкута, Индија, Академије за информационе технологије на Универзитету „Џејмс Кук“, Кернс, Аустралија и Технолошког универзитета у Сиднеју, Аустралија (енг. *Indian Statistical Institute, Kolkata, India; Information Technology Academy, James Cook University, Cairns, Australia; School of Software, University of Technology Sydney, Australia*); Катедра за информатику и инжењерство, „Мар Атанасијус“ колеџ за инжењерство, Котамангалам, Керала, Индија (енг. *Department of Computer Science and Engineering, Mar Athanasius College of Engineering, Kothamangalam, Kerala, India*); Политехнички државни институт у Калифорнији, Сан Луис Обиспо, Калифорнија (енг. *California Polytechnic State University, San Luis Obispo, CA*); и Факултет науке и технологије, Отворени универзитет у Хонг Конгу, Хо Ман Тин, Ковлон, Хонг Конг, Кина (енг. *School of Science and Technology, The Open University of Hong Kong, Ho Man Tin, Kowloon, Hong Kong SAR, China*).

Одабрани софтверски системи су наведени у сортираном поретку. Приликом сортирања изабраних софтверских система користи се критеријум са три кључа. Најпре се врши сортирање по првом кључу, а затим у случају истих вредности првог кључа по другом кључу, а потом у случају истих вредности другог кључа по трећем кључу. Први кључ представља тип питања, при чему су најпре наведени једноставнији типови питања (вишеструки избор, па повезивање, па кратак одговор). Други кључ представља година публикација рада који се тиче реализованог софтверског система, при чему су најпре наведени новији резултати. Трећи кључ представља назив софтверског система, при чему је поредак навођења назива лексикографски растући.

За неке софтверске системе није битно који је коришћени језик на папирном тесту те је у таквим случајевима уместо назива језика наведена коса црта (/). Важно је напоменути да неки аутори нису именovali своје софтверске системе. Стога је и за неке од софтверских система урађено именовање. С друге стране, за неке софтверске системе је урађено преименовање. У оба случаја су њихова нова имена настала на основу података из уводних информација, које приказује Табела 5.

Табела 5. Кратак преглед изабраних софтверских система

Софтверски систем	Класа питања	Језик	Методе	Година
<i>TARS</i> [42]	Вишеструки избор	/	Пајтон; Компјутерска визија.	2022.
<i>MCTQF</i> [43]	Вишеструки избор	/	Матлаб; Компјутерска визија; Прилагођена обрада слике.	2018.
<i>MCG-RAS</i> [44]	Вишеструки избор	/	Октава; Компјутерска визија; Упаривање шаблона; Оптичко препознавање ознака.	2017.
<i>GMCQ-FA</i> [45]	Вишеструки избор	/	Ц шарп (енг. <i>C#</i>) Компјутерска визија.	2016.
<i>Eyegrade</i> [46]	Вишеструки избор	/	Пајтон; Компјутерска визија.	2013.
<i>ASSHEP</i> [47]	Повезивање	Енглески, кинески	Пајтон; Јоло мрежа.	2021.
<i>AGHAS</i> [48]	Повезивање	Енглески	Пајтон; Матлаб; Конволуциона неурална мрежа.	2019.
<i>ARHC</i> [49]	Повезивање	Енглески	Јава; Компјутерска визија.	2014.
<i>FASAS</i> [50]	Кратак одговор	Јапански	Конволуционе неуралне мреже; Трансформери.	2022.
<i>ISSHSA</i> [51]	Кратак одговор	Кинески	Пајтон; Семантичко повезивање; Конволуционе неуралне мреже.	2020.
<i>TSAWR</i> [52]	Кратак одговор	Тајски	Компјутерска визија; Конволуциона неурална мрежа.	2020.
<i>HSAES</i> [53]	Кратак одговор	Енглески	Пајтон; Оптичко препознавање знакова; Обрада природног језика.	2018.
<i>AGHNA</i> [54]	Кратак одговор	Енглески	Пајтон; Компјутерска визија; Конволуциона неурална мрежа.	2017.
<i>AGSLCQ</i> [55]	Кратак одговор	Енглески	Пајтон; Оптичко препознавање знакова; Обрада природног језика.	2015.

Како би се омогућио јаснији и униформнији преглед изабраних софтверских система, осмишљен је шаблон за структурирању приказ њихових најважнијих информација. Сваки од изабраних софтверских система описан је једном од подсекција које следе у наставку. Свака подсекција је форматирана у складу са осмишљеним шаблоном. Структура осмишљеног шаблона састоји се од следећих елемената:

- Резиме – Приказује кратке информације о изабраном софтверском систему. Међу њима спадају афилијације аутора софтверског система, година реализације софтверског система и година публиковања рада. Ове информације неће бити приказане као ставка нумерисане листе, већ ће бити наведене као уводни параграф у сваки од изабраних софтверских система.
- Сврха – Описује мотивацију аутора за израду софтверског система. Приказује циљеве које изабрани софтверски систем треба да испуни.
- Изглед папирног теста – Пружа опис структуре папирног теста који изабрани софтверски систем оцењује. Поред тога, истиче се и да ли изабрани софтверски систем захтева да папирни тест, чије се аутоматизовано оцењивање врши, има раздвојене обрасце за текст питања и упис одговора. Обједињени обрасци представљају већи изазов за системе, јер текст питања уноси шум у процес идентификације одговора кандидата.
- Класе питања – Наводи једну или више претходно идентификованих класа питања које је изабрани софтверски систем способан да оцењује.
- Технологије – Наводи главне технологије које су допринеле развоју изабраног софтверског система.
- Области вештачке интелигенције – Излистава коришћене алгоритме и области вештачке интелигенције искоришћене за развој изабраног софтверског система.
- Структура система – Представља процес спровођења испитивања. Поред тога, даје структурални опис изабраног софтверског система за аутоматизовано оцењивање по његовим компонентама. Додатно, наводи обим података искоришћен за тренирање система, уколико је оно потребно.
- Експерименти – Приказује обављена тестирања изабраног софтверског система. Наводи обим података искоришћен за потребе тестирања система. Износи информације о перформансама добијеним тестирањем система, као што су подаци о тачности у раду и времену потребном за процесирање.
- Евалуација – Укратко наглашава главне уочене предности и недостатке изабраног софтверског система, као и оне које су навели сами аутори.

3.1. TARS

Софтверски систем TARS (енг. *Test Answers Recognition Software*) је направљен на Катедри за рачунарску технику и информатику Електротехничког факултета Универзитета у Београду. Прва верзија овог пројекта имплементирана је 2010. године. Користећи модерне технологије, пројекат је обновљен 2022. године.

- Сврха: Мотивација за развој оваквог система била је растерећење наставног кадра од посла прегледања великог броја папирних тестова. Услед великог броја кандидата и недовољно рачунарских ресурса, испитивање и оцењивање кандидата није било могуће уз помоћ платформи за електронско учење и оцењивање. Стога, приступило се развоју

формата теста и његових питања и софтверског система који ће испољавати могућност да аутоматски оцењује питања на таквом тесту.

- Изглед папирног теста: Папирни тест се састоји из једног листа (формулара) који може да подржи варијабилан број питања и одговора. На једној страници формулара налазе се поставке питања и њихови понуђени одговори. Друга страница формулара подељена је на два дела, при чему се на једном делу налази упутство за попуњавање формулара, а на другом регион од интереса систему који кандидат попуњава. Непопуњени формулар који овај софтверски систем користи приказује Слика 16.

The image shows a blank test form template. At the top, there is a box for identification containing six seven-segment displays. The first display is pre-filled with a square symbol, and the others are empty. Below this is a header row for the answer matrix with labels A, B, C, N, and V. The matrix itself consists of 7 rows and 5 columns of circles. To the left of the matrix are row numbers 1 through 7. Below the matrix is a row of four empty circles. In the top right corner of the form, there is a small square box.

Слика 16. Непопуњени формулар за упис одговора

Регион од интереса се састоји из идентификационе секције, секције за упис одговора, секције за кодирање варијанте формулара и секције за означавање неопходности ручног прегледа формулара. Идентификациона секција се састоји од шест седмосегментних цифара, које представљају четвороцифрен број индекса (при чему је водећа 0 имплицитна и не уписује се ручно) и последње две цифре године уписа кандидата. Секција за упис одговора се састоји од кругова распоређених у квадратну матрицу, при чему свака врста те матрице представља једно питање, док кругови те врсте представљају одговоре на то питање.

Кандидат изабира одговор зацрњивањем оловком круга у одговарајућој врсти и колони. Секција за кодирање варијанте носи информацију о варијанти теста представљеној тим формуларом. Кандидати не врше упис у ову секцију. Варијанта теста се кодира приликом дизајнирања теста тако што је вредност варијанте теста представљена бинарним записом попуњених и празних кругова распоређених у једном реду, при чему попуњени кругови представљају вредност 1, а празни вредност 0.

Регион од интереса садржи и један квадрат чијим зацрњивањем кандидат ставља до знања систему да је овај формулар потребно ручно прегледати. Ово се дешава у ситуацијама када је кандидат направио грешку приликом попуњавања формулара, а није постојао довољан број заменских формулара или је истекло време за замену формулара. У том случају кандидат може поништити свој одговор на неко питање тако што ће у реду матрице који се односи на то питање зацрнити више од једног круга, при чему није неопходно попуњавати квадрат за ручни преглед. Овај квадрат је неопходно зацрнити само, ако је изабран други одговор, а на формулару је већ један био означен.

- Класе питања: Овај софтверски систем је направљен са идејом да детектује и оцењује питања типа вишеструки избор на папирним тестовима.
- Технологије: Целокупан софтверски систем развијен је коришћењем програмског језика Пајтон, његових стандардних библиотека и екстерних библиотека отвореног кода. Систем користи Мајкрософтов производ Ексел (енг. *Excel*) за манипулацију и складиштење добијених резултата.
- Области вештачке интелигенције: Овај софтверски систем користи алгоритме компјутерске визије (као што су адаптивно трешхолдовање, бинаризација, ерозија, дилатација итд.) за препроцесирање слике. Ово је неопходно урадити због шума који је настао у процесу штампања, скенирања или попуњавања теста. За детекцију региона од интереса користе се алгоритми компјутерске визије (претрага контура), док се за потврђивање облика од интереса користе сопствено развијени алгоритми за препознавање облика (кругова, правоугаоника итд.). Такође, развијени су и сопствени алгоритми за прецизно утврђивање степена зацрњености облика. Ови алгоритми се користе за препознавање свих секција у региону од интереса: идентификације кандидата, одговора кандидата, варијанте теста и информације да ли је тест неопходно ручно прегледати.
- Структура система: Целокупан процес испитивања састоји се из неколико делова. Најпре се, користећи компоненту за дизајнирање теста, врши креирање питања и одговора теста, као и генерисање одређеног броја варијанти теста. Као производ ове компоненте настаје тест у дигиталном облику који се умножава у потребном броју примерака у центру за штампање.

Након тога, у процесу израде теста од стране кандидата настаје одређени број попуњених папирних тестова. Затим се они односе у скенинг центар да би се добио дигитални облик сваког од попуњених папирних тестова. Тестови рађени у једној сали представљају појединачне странице вишестраничне дигиталне датотеке.

Ове датотеке представљају улаз у компоненту задужену за детекцију региона од интереса и њихових секција. Најпре се детектује спољни оквир региона од интереса шаблона, као највећи правоугаони облик на самом шаблону. С обзиром да приликом процеса скенирања настају мале ротације формулара, систем детекцијом региона од интереса одређује њихов нагиб у односу на замишљену хоризонталну линију, а затим ротира формулар тако да региони од интереса представљени правоугаоникима буду паралелни замишљеној линији.

Затим се детектују правоугаоници у оквиру региона од интереса шаблона који окружују информације о: идентификацији кандидата, његовим одговорима и варијанти теста. У оквиру идентификационог региона од интереса врши се претрага на 6 седмосегментних цифара и одређује се степен њихове зацрњености. Затим се у региону од интереса са понуђеним одговорима врши претрага на очекивани број кругова одговора, као и

тестирање да ли ови кругови формирају правилну матричну структуру, а потом и одређује степен њихове зацрњености.

Након тога, у региону од интереса са варијантом теста, врши се претрага на кругове којима је варијанта теста кодирана и одређује се степен њихове зацрњености. Детекција свих облика од интереса врши се на попуњеном шаблону и проверава се, укрштањем координата, са подацима добијеним од процеса детекције на непопуњеном шаблону. Затим се идентификовани облици прослеђују компоненти која врши анотацију ротираног формулара утврђеним идентификатором кандидата, тачним и нетачним одговорима, као и варијантом теста на формулару и уписује добијене резултате у табелу са резултатима.

Приликом утврђивања зацрњености користи се правило да се регион зацрњен испод 45 % сматра незацрњеним, изнад 55 % зацрњеним, док опсег између 45 % и 55 % систем сматра „сивом зоном“ коју треба ручно проверити и обележава такве формуларе. Ове параметре је могуће променити кроз опције система или кроз конфигурациону датотеку. Утврђени идентификатор кандидата проверава се са базом кандидата пријављених за процес испитивања, а систем такође води рачуна да ли су детектована два или више истих идентификационих података, као и да ли је детектован неисправан идентификациони податак, при чему се и формулар анотира на посебан начин. Пример анотираног формулара попуњеног од стране кандидата илуструје Слика 17.

Електротехнички факултет Универзитета у Београду
Београд, 29.08.2023.

Упутство

Непопуњеним одговорама нека резултате дисциплованом прилагођеном ступица:

1. На карти су додељени испитивачи испит, лично карта, коришћено и прибор за писање.
2. Забрањено је коришћење општедоступних уређаја и осталих извозних средстава.

Непопуњеним одговорама нека резултате дисциплованом прилагођеном ступица:

3. Обавезно треба попуњавати испитивачки испит или остали извозни систем.
4. На обрасцу треба одмах потписати име и презиме.
5. На обрасцу треба одмах потписати име и презиме.
6. На обрасцу треба одмах потписати име и презиме.
7. На обрасцу треба одмах потписати име и презиме.
8. На обрасцу треба одмах потписати име и презиме.
9. На обрасцу треба одмах потписати име и презиме.
10. На обрасцу треба одмах потписати име и презиме.

На карти је приказано који сегменти треба да се у волуности зацртају за сваку од цифара.

	A	B	C	N	V
1	●	○	○	○	○
2	○	●	○	○	○
3	○	○	●	○	○
4	○	○	○	○	○
5	○	○	○	●	○
6	●	○	○	○	○
7	○	○	○	○	○

Code: 10

Слика 17. Формулар анотиран резултатима процесирања

Поред тога, и у табелу са резултатима уписује се вредност која треба прегледачу да означи да постоји проблем са одређеним формуларом. На анотираном формулару светло-зеленом бојом представљени су одговори који су кандидату у складу са кључем за додељену варијанту шаблона, црвеном бојом су представљени нетачни одговори кандидата, док су плавом бојом представљени тачни одговори за питања на која кандидат није одговорио. Тамно-зеленом бојом представљени су степени зацрњености одговора у процентима и утврђени идентификациони број кандидата.

- Експерименти: Тестирање софтверског система спроведено је над 763 папирна теста попуњена од стране кандидата. Систем је успео коректно да утврди идентификације кандидата у 94.1 % случајева, у 0.2 % случајева није коректно утврдио идентификације

кандидата, док у 5.7 % случајева кандидати нису на исправан начин уписали свој идентификациони број, али је систем коректно утврдио зацрњене сегменте цифара. Стога, може се сматрати да је систем у 99.8% случајева коректно утврдио идентификацију кандидата.

Прецизност система приликом детекције и оцењивања одговора износи 99.3 %. Успешно препознавање кодиране варијанте теста на формулару обављено је у 100 % случајева. Процесирање тестова може да се обавља секвенцијално или у паралели, при чему је за процесирање једног формулара у паралелном режиму рада неопходно 290 милисекунди, што укључује и време потребно за његову анотацију детектованим подацима. Машина коришћена за тестирање поседује 16 гигабајта радне меморије и Интелов процесор *i5-6500 @3.2GHz* са 4 физичка и исто толико виртуелних језгара.

- Евалуација:
 - Предности: Систем испољава изузетно високу прецизност у детекцији и утврђивању зацрњености облика од интереса (изнад 99.3 %). Може да се користи у секвенцијалном или паралелном режиму рада, при чему аутоматски одређује број процеса које користи у паралелизацији. Такође, дозвољава да кандидати у случају грешке означе да формулар треба да буде и ручно прегледан или да пониште свој одговор на питање зацрњивањем два или више круга одговора у реду матрице који представља једно питање. Поред тога, систем дозвољава употребу како обичне, тако и хемијске оловке. Такође, систем је отпоран на ротације попуњеног шаблона настале у процесу скенирања.
 - Недостаци: Формат теста мора бити такав да сва питања и понуђени одговори стану на једну страницу А4 листа. На другој страници на којој се налазе упутство и регион од интереса за попуњавање, то у пракси значи до 20 питања и до 10 понуђених одговора. Такође, систем не дозвољава кандидатима да на истом обрасцу промене свој одговор, већ само да га пониште.


3.2. MCTQF

Софтверски систем MCTQF (енг. *Multiple Choice Test with Quick Feedback*) је развијен од стране Факултета за инжењерство Универзитета „Едит Кован“ у Перту, Аустралији (енг. *School of Engineering, Edith Cowan University, Perth, Australia*). Овај пројекат реализован је током 2018. године.

- Сврха: Мотивација за настанак овог софтверског система била је смањивање трошкова у процесирању тестова коришћењем технологија за обраду слике. Идеја је била да се реализује систем који ће бити јефтинији и бржи у свом раду у односу на скупе машине за оптичко препознавање ознака (енг. *Optical Mark Reader*). Поред тога, овакав систем би ослободио академске раднике од додатног посла ручног прегледања.
- Изглед папирног теста: Папирни тест који овај систем користи представљен је једном А4 страницом која се састоји од 72 питања, при чему свако питање има тачно 5 одговора. Цела страница формулара подељена је на секције које садрже информације о самом испиту, упутство за попуњавање формулара, као и регионе од интереса које кандидат попуњава, а које садрже секцију за идентификацију кандидата и секцију са питањима за упис њихових одговора. Текст питања и њихових одговора не налази се у оквиру овог формулара, већ на посебним листовима. Идентификациона секција се састоји од осам седмосегментних цифара, које представљају јединствени идентификатор кандидата. Секција за упис одговора се састоји од питања распоређених у три колоне од по 24 питања, при чему се 5 одговора обележених латиничним словима

А до Е налази са десне стране броја питања. Кандидат изабира одговор зацрњивањем оловком одговарајућег поља са словом у одговарајућем питању. Испод сваког поља за упис одговора налази се додатно поље, чијим се зацрњивањем поништава одговор одређен тим пољем. На тај начин кандидат може изабрати други одговор од понуђених. Формулар садржи и три попуњена правоугаоника који представљају његове оријентирне тачке. Ово је неопходно из разлога што приликом скенирања формулара готово увек долази до благе ротације формулара која треба бити исправљена. Непопуњени формулар који систем користи илуструје Слика 18.

Answer Sheet



EDITH COWAN

Unit Code:

Unit Title:

Student ID

Student ID Instructions

Highlight the digits above clearly to create your student ID by following the patterns below.

For example, if a student ID is 10324526, this is how it should be written above:

1 0 3 2 4 5 2 6

Surname

First Name

Answer Area Instructions

To answer each question Shade area under the letter as shown in Example 1.

If you would like to change your answer, shade the whole area as shown in Example 2.

How to Answer

Example 1

A B C D E

How to change Answer

Example 2

A B C D E

1	A	B	C	D	E
2	A	B	C	D	E
3	A	B	C	D	E
4	A	B	C	D	E
5	A	B	C	D	E
6	A	B	C	D	E
7	A	B	C	D	E
8	A	B	C	D	E
9	A	B	C	D	E
10	A	B	C	D	E
11	A	B	C	D	E
12	A	B	C	D	E
13	A	B	C	D	E
14	A	B	C	D	E
15	A	B	C	D	E
16	A	B	C	D	E
17	A	B	C	D	E
18	A	B	C	D	E
19	A	B	C	D	E
20	A	B	C	D	E
21	A	B	C	D	E
22	A	B	C	D	E
23	A	B	C	D	E
24	A	B	C	D	E

25	A	B	C	D	E
26	A	B	C	D	E
27	A	B	C	D	E
28	A	B	C	D	E
29	A	B	C	D	E
30	A	B	C	D	E
31	A	B	C	D	E
32	A	B	C	D	E
33	A	B	C	D	E
34	A	B	C	D	E
35	A	B	C	D	E
36	A	B	C	D	E
37	A	B	C	D	E
38	A	B	C	D	E
39	A	B	C	D	E
40	A	B	C	D	E
41	A	B	C	D	E
42	A	B	C	D	E
43	A	B	C	D	E
44	A	B	C	D	E
45	A	B	C	D	E
46	A	B	C	D	E
47	A	B	C	D	E
48	A	B	C	D	E

49	A	B	C	D	E
50	A	B	C	D	E
51	A	B	C	D	E
52	A	B	C	D	E
53	A	B	C	D	E
54	A	B	C	D	E
55	A	B	C	D	E
56	A	B	C	D	E
57	A	B	C	D	E
58	A	B	C	D	E
59	A	B	C	D	E
60	A	B	C	D	E
61	A	B	C	D	E
62	A	B	C	D	E
63	A	B	C	D	E
64	A	B	C	D	E
65	A	B	C	D	E
66	A	B	C	D	E
67	A	B	C	D	E
68	A	B	C	D	E
69	A	B	C	D	E
70	A	B	C	D	E
71	A	B	C	D	E
72	A	B	C	D	E

Слика 18. Непопуњени формулар за упис одговора

- Класе питања: Овај софтверски систем је направљен са идејом да детектује и оцењује питања типа вишеструки избор на папирним тестовима.
- Технологије: Целокупан софтверски систем развијен је коришћењем програмског језика Матлаб и његових стандардних библиотеке. Систем користи Мајкрософтов производ Ексел за манипулацију и складиштење добијених резултата.
- Области вештачке интелигенције: Овај софтверски систем користи алгоритме компјутерске визије (као што су бинаризација, ерозија, дилатација итд.) за детекцију региона од интереса. Поред тога, користи и сопствено развијене алгоритме за препознавање шаблона и одређивање степена зацрњености облика. Ови алгоритми се користе за препознавање идентификације кандидата и његових одговора.
- Структура система: Целокупан процес испитивања састоји се из неколико делова. Најпре се, на основу шаблона теста у дигиталном облику креира потребан број папирних примерака шаблона у центру за штампање. Након тога, у процесу израде теста од стране кандидата настаје одређени број попуњених папирних тестова.

Они се, затим, односе у скенинг центар да би се добио дигитални облик сваког од попуњених папирних тестова. Ове датотеке представљају улаз у компоненту задужену за детекцију региона од интереса и њихових секција. Најпре се врши детекција три оријентирна попуњена правоугаоника, чије су координате са најмањом x , највећом x и највећим x и y вредностима.

Ово је неопходно урадити с обзиром да приликом процеса скенирања настају мале ротације формулара. Након детекције оријентирних правоугаоника одређује се угао нагиба формулара, а затим се и сам формулар ротира за одређени угао. Потом се детектује идентификација кандидата и утврђује степен зацрњености сваког од седам сегмената за сваку од осам цифара идентификације, а након тога се детектују и поља одговора кандидата и утврђује њихов степен зацрњености.

Детекција региона идентификације кандидата и поља одговора врши се транслирањем координата претходно добијеним на непопуњеном референтном шаблону. Утврђена идентификација кандидата проверава се са постојећим вредностима у бази идентификација кандидата. Користи се фиксни праг зацрњености који износи 40 % и није битан начин на који кандидати врше зацрњивање.

Затим се врши анотација формулара утврђеним идентификатором кандидата, као и тачним и нетачним одговорима и уписивање добијених резултате у табелу са резултатима. На анотираном формулару светло-зеленом бојом представљени су одговори који су кандидату у складу са кључем теста, црвеном бојом су представљени нетачни одговори кандидата, док су светло-плавом бојом представљени тачни одговори за питања на које кандидат није одговорио. Анотирани део формулара са одговорима приказује Слика 19.

	A	B	C	D	E
1					
2	A	B	C	D	E
3	A	B	C	D	E
4	A	B	C	D	E
5	A	B	C	D	E
6	A	B	C	D	E
7	A	B	C	D	E
8	A	B	C	D	E

	A	B	C	D	E
1					
2	A	B	C	D	E
3	A	B	C	D	E
4	A	B	C	D	E
5	A	B	C	D	E
6	A	B	C	D	E
7	A	B	C	D	E
8	A	B	C	D	E

Слика 19. Формулар анотиран резултатима процесирања

- Експерименти: Тестирање софтверског система спроведено је над 88 папирних тестова попуњених од стране кандидата. Систем је успешно утврдио идентификације кандидата у 95 % случајева. Прецизност система приликом детекције и оцењивања одговора износи изузетних 100 %.

Процесирање тестова обавља се искључиво секвенцијално, при чему је за процесирање једног формулара потребно 400 милисекунди. Уколико се врши и анотација шаблона, онда време неопходно за процесирање једног теста расте на 2.27 секунди. Машина коришћена за тестирање поседује 8 гигабајта радне меморије и Интелов процесор *i7*, при чему није наведен радни такт процесора нити број физичких или виртуелних језгара. У процесу тестирања утврђено је да је угао ротације шаблона био између 0.5 и 1.5 степени, али је у екстремним ситуацијама износио и 6 степени.

- Евалуација:

- Предности: Систем испољава изузетну прецизност у детекцији и утврђивању зацрњености одговора (100 %), као и велику прецизност у детекцији студентских идентификација (95 %). Поред тога, дозвољава да кандидати у случају избора погрешног одговора прогласе тај одговор за ништаван зацрњивањем поља испод тог одговора и изаберу неки други одговор. Такође, систем је отпоран на ротације попуњеног шаблона настале у процесу скенирања. Систем показује могућност релативно брзог процесирања папирних формулара за које је потребно 0.4 секунде по неанотираном и 2.27 секунди по анотираном формулару на репрезентативној машини.
- Недостаци: Подаци о прецизности система добијени су на изузетно малом скупу података (само 88 тестова). Формат шаблона је стриктан и подржава тачно 5 одговора на највише 72 питања. Иако кандидати имају могућност да пониште раније изабран одговор, не могу никако поново изабрати неки од претходно изабраних одговора. Такође, аутори система су нагласили да би корисничко искуство требало да буде унапређено, како би се избегло коришћење Матлаба у процесу бодовања.

3.3. MCG-RAS

Софтверски систем MCG-RAS (енг. *Multiple Choice Grading using Readily Available Software*) је направљен на Универзитету „Де Ла Сале“ на Филипинима (енг. *De La Salle University, Philippines*). Овај пројекат реализован је током 2017. године.

- Сврха: Мотивација за развој оваквог система била је немогућност коришћења платформи за електронско учење и оцењивање. Поред тога, постојеће машине за оптичко препознавање ознака имају велику набавну цену и нису погодне за коришћење. Стога, приступило се развоју јефтиније софтверске алтернативе која користи камеру или скенер и која пружа већу флексибилност у коришћењу.
- Изглед папирног теста: Папирни тест се састоји из једног листа формулара који се састоји од 110 питања, при чему свако питање има тачно 5 одговора. Питања су нумерисана арапским бројевима, док су понуђени одговори означени са првих 5 слова енглеске абецеде. Шаблон је дизајниран користећи Мајкрософтов Ворд алат, угледајући се на обрасце који се користе у комерцијално доступним решењима.

Страница формулара садржи информације о испиту и секције од интереса за софтверски систем, које кандидат попуњава. То су секција са изабраним одговорима кандидата и секција за упис идентификације кандидата. Текст питања и њихових одговора не налази се у оквиру овог формулара, већ на посебним листовима.

Секција за кодирање варијанте формулара састоји се од 8 кругова поређаних у једном реду и означених са првих 8 великих слова енглеске абецеде. Секција за кодирање варијанте формулара се уписује аутоматски у процесу штампања формулара, а непосредно пре његовог копирања. Идентификациона секција се састоји од матрице кругова која има 10 редова (за цифре од 0 до 9) и 8 колона, која служи за представљање идентификације кандидата дужине 8 цифара.

Секција за упис одговора се састоји од питања распоређених у 4 колоне од по 25 питања и петој непотпуној колони од 10 питања, при чему се 5 кругова (одговора) налази са десне стране броја питања. Кандидат изабира одговор зацрњивањем оловком одговарајућег поља са словом у одговарајућем питању. Формулар садржи и четири попуњена круга већих димензија него сви остали његови елементи. Ова четири попуњена круга представљају оријентирне тачке формулара.

Они се користе као референтне тачке за остале елементе формулара које треба детектовати. Такође, приликом скенирања формулара долази до благе ротације формулара која треба бити исправљена. Стога, оријентирне тачке имају и улогу у одређивању угла нагиба ротираног формулара.

Поред тога, формулар садржи и по један ред и колону од по 25 нешто ситнијих зацрњених кружних облика који служе као оријентирне тачке за понуђене одговоре шаблона. Они су корисни, имајући у виду да се сваки од одговора налази у пресеку врсте и колоне неке две изабране оријентирне тачке. Непопуњени формулар који систем користи показује Слика 20.

Слика 20. Непопуњени формулар за упис одговора

- Класе питања: Овај софтверски систем је направљен са идејом да детектује и оцењује питања типа вишеструки избор на папирним тестовима.
- Технологије: Целокупан софтверски систем развијен је коришћењем програмског језика Октава, његових стандардних библиотека и екстерних библиотека отвореног кода. Систем користи Мајкрософтов производ Ексел за манипулацију и складиштење добијених резултата.
- Области вештачке интелигенције: Овај софтверски систем користи алгоритме компјутерске визије за обраду слике (нпр. бинаризација). Поред тога, користи технике упаривања шаблона за утврђивање облика од интереса. Ове технике се користе за утврђивање локација оријентирних кругова на шаблону, као и кругова који представљају варијанту теста, понуђене одговоре теста и идентификацију кандидата. Такође, користи и сопствено развијене алгоритме за одређивање степена зацрњености облика од интереса.
- Структура система: Целокупан процес испитивања састоји се из неколико делова. Најпре се, на основу шаблона теста у дигиталном облику креира папирни облик шаблона користећи обичан штампач. Затим се он умножава у потребном броју папирних примерака користећи црно-белу фотокопир машину.

Након тога, у процесу израде теста од стране кандидата настаје одређени број попуњених формулара. Они се затим односе у скенинг центар да би се добио дигитални облик сваког од попуњених папирних тестова, при чему је сваки тест представљен по

Најпре се врши детекција четири оријентирна попуњена круга, чије су димензије веће од свих осталих елемената на обрасцу. Ово се врши коришћењем технике упаривања шаблона. Потом се два оријентирна круга користе за корекцију нежељене ротације настале у процесу скенирања.

На сличан начин се одређује и идентификација кандидата, с обзиром да се кругови користе за њено означавање. Потом се шаблон аотира подацима о тачним одговорима. Потом компонента за процесирање упоређује резултате добијене препознавањем са кључем са тачним одговорима за тај тест и уписује добијене резултате у табелу.

Name: _____

Date: DEC 12, 20K Subject/Section: FL 2

Student Number: _____

	a	b	c	d	e	a	b	c	d	e	a	b	c	d	e	a	b	c	d	e	a	b	c	d	e
1																									
2																									
3																									
4																									
5																									
6																									
7																									
8																									
9																									
10																									
11																									
12																									
13																									
14																									
15																									
16																									
17																									
18																									
19																									
20																									
21																									
22																									
23																									
24																									
25																									

Set:

0

- **Експерименти:** Тестирање софтверског система спроведено је над 800 папирних тестова попуњених од стране кандидата. Систем је успео коректно да процесира 790 тестова, што представља прецизност од 98.75 %. Разлози за неуспешно процесирање преосталих 10 тестова (1.25 %) приписују се дефектима насталим у процесу штампања, као и у процесу скенирања.

32

потребно за анотацију формулара детектованим подацима. Нису наведени подаци о машини на којој је покретан систем за процесирање тестова.

- Евалуација:
 - Предности: Систем испољава изузетно високу прецизност у детекцији и утврђивању зацрњености облика од интереса, која износи 98.75 %. Иако појединачно скенирање формулара одузима доста времена, оно може да се обави аутоматски уз помоћ ADF скенера. Систем је отпоран на ротације попуњеног шаблона настале у процесу скенирања. Изабрани систем представља скалабилну и јефтину варијанту система за аутоматизовано оцењивање папирних тестова, која се користи дуги низ година на неколико курсева.
 - Недостаци: Време потребно за процесирање једног формулара је прилично велико и износи 68 секунди. Поред тога, формат теста је фиксан у погледу броја питања и одговора. Такође, систем не дозвољава кандидату да промени свој изабрани одговор нити да га поништи.

3.4. GMCQ-FA

Софтверски систем GMCQ-FA (енг. *Grading Multiple Choice Questions and Feedback Analysis*) је направљен од стране Катедре за електронику и телекомуникације, Кеј Си колеџа за студије инжењерства, менаџмента и истраживања, у Тејну, Махараштри, Индији (енг. *Department of Electronics and Telecommunication, K.C. College of Engineering & Management Studies & Research, Thane, Maharashtra, India*). Овај пројекат реализован је током 2016. године.

- Сврха: Мотивација за развој оваквог система била је омогућавање малим едукационим институцијама да спроведу аутоматизовано оцењивање тестова кандидата, уместо досадашњег ручног прегледања тестова. Таквим институцијама, услед недостатка новчаних средстава, нису на располагању скупе машине за евалуацију тестова препознавањем оптичких ознака. Чак и да је оваква машина доступна у оваквим институцијама, потребни су специјализовани оператери ових машина да би се оне користиле на исправан начин и уз пуну ефикасност. Због тога се приступило развоју софтверског решења које ће поседовати могућност да аутоматски оцењује питања на папирном тесту уз коришћење обичног персоналног рачунара и скенера.
- Изглед папирног теста: Папирни тест се састоји из једног листа формулара који се састоји од 150 питања, при чему свако питање има тачно 4 одговора. Текстови питања и њихових одговора не налазе се у оквиру овог формулара, већ на посебним листовима. Питања су нумерисана арапским бројевима, док су понуђени одговори означени са прва 4 слова енглеске абетецеде.

Страница формулара у заглављу садржи информације о испиту и институцији у оквиру које се испит одржава. У левом делу формулара се налази секција са упис идентификације кандидата, коју кандидат попуњава. Идентификациона секција се састоји од матрице кругова која има 10 редова (за цифре од 0 до 9) и 10 колона и која служи за представљање идентификације кандидата дужине 10 цифара.

Десно од ове секције налази се секција за кодирање варијанте формулара, у коју софтвер аутоматски уписује варијанту у процесу штампања формулара, а непосредно пре његовог копирања. Она се састоји од 10 редова (за цифре од 0 до 9) и 3 колоне, па је могуће означити 1000 различитих варијанти теста. Централни део формулара окупира секција са понуђеним одговорима на питања, које кандидат попуњава.

Секција за упис одговора се састоји од питања распоређених у 6 колона од по 25 питања, при чему се 4 круга (одговора) налазе са десне стране броја питања. Кандидат изабара одговор зацртавањем оловком одговарајућег поља са словом у одговарајућем питању. Непопуњени формулар који систем користи илуструје Слика 22.

Слика 22. Непопуњени формулар за упис одговора

- **Технологије:** Целокупан софтверски систем развијен је коришћењем програмског језика Ц шарп, његових стандардних библиотека и екстерних библиотека отвореног кода. Систем користи Сиквел (енг. *SQL*) језик за манипулацију базом података за манипулацију и складиштење добијених резултата.
- **Класе питања:** Овај софтверски систем је направљен са идејом да детектује и оцењује питања типа вишеструки избор на папирним тестовима.
- **Области вештачке интелигенције:** Овај софтверски систем користи алгоритме компјутерске визије (као што су бинаризација, ерозија, дилатација итд.) за детекцију региона од интереса. Ови алгоритми се користе за препознавање свих секција у региону од интереса: идентификације кандидата, одговора кандидата и варијанте теста. Поред тога, систем користи и процес регистрације слике (енгл. *Image registration*) како би се слика попуњеног формулара трансформисала у исти координатни систем као и слика непопуњеног шаблонског формулара ради исправљања ротације папира настале у процесу скенирања.
- **Структура система:** Целокупан систем састоји се из три дела: софтверске апликације, формулара и скенера. Софтверска апликација омогућава избор једног од неколико врста формулара, чија је измена могућа у погледу различитог начина идентификације

кандидата. Затим се, на основу изабраног шаблона теста у дигиталном облику креира папирни шаблон користећи обичан штампач.

Потом се папирни шаблон умножава у потребном броју примерака користећи црно-белу фотокопир машину. Након тога, у процесу израде теста од стране кандидата настаје одређени број попуњених формулара. Они се, затим, ручно скенирају на обичном скенеру и за сваки шаблон добија се по једна слика која се чува у раније изабраном директоријуму.

Ове датотеке представљају улаз у систем, који врши детекцију региона од интереса, односно идентификације кандидата, варијанте теста и изабраних одговора кандидата. Идентификовани подаци се уписују у базу података. Након тога се они пореде са референтним подацима (нпр. матрицом тачних одговора) из мастер базе података, а затим се добијени резултати чувају у бази са резултатима кандидата.

- Експерименти: Аутори нису навели податке о величини скупа података коришћеног за тестирање, као ни о тачности коју систем постиже у процесу препознавања. Такође, аутори нису навели ни податке о машини на којој је покретан систем за процесирање тестова. Међутим, овај софтверски систем је доступан за коришћење уз претходну регистрацију.

Стога, извршено је тестирање користећи овај систем над скупом од 200 претходно одштампаних тестова са по 25 питања попуњених од стране студената. Систем је успео коректно да процесира 197 тестова, што представља прецизност од 98.5 %. Систем је показао успешност од 99 % у детекцији идентификација кандидата, с обзиром да на 2 теста није успео да детектује идентификацију кандидата. Време потребно за процесирање овог скупа тестова износило је 1520 секунди, што износи 7.6 секунди по једном тесту.

- Евалуација:
 - Предности: У спроведеном експерименту, систем је испољио високу прецизност у препознавању идентификација кандидата која износи 99%, као и високу прецизност у препознавању питања ове класе која износи 98.5 %. Време потребно за процесирање једног обрасца износи 7.6 секунди. Аутори су навели да користећи обичан персонални рачунар, а у зависности од врсте коришћеног скенера, систем омогућава процесирање око 12 000 шаблона у року од 24 сата, што значи да је за процесирање једног попуњеног теста потребно најмање 7.2 секунде, што се поклапа са резултатима спроведеног експеримента.

Систем је отпоран на ротације попуњеног шаблона настале у процесу скенирања. Такође, систем не уводи ограничења по питању средства попуњавања шаблона, односно не прави разлику између обичне и хемијске оловке. Поред тога, датотеке које настају у процесу коришћења система су релативно мале величине. Овај систем представља приуштиву варијанту система за аутоматизовано оцењивање папирних тестова, имајући у виду да захтева само персонални рачунар и обичан скенер.

- Недостаци: Нису наведени подаци о конфигурацији машине која је коришћена за добијање података о времену потребном за процесирање формулара. Поред тога, формат теста је стриктан у погледу броја питања и одговора. Такође, формулар би могао бити аотиран подацима добијеним у процесу детекције облика од интереса. Додатно, систем не дозвољава кандидату да поништи свој раније изабрани одговор нити да га промени.

3.5. Eyegrade

Софтверски систем Eyegrade направљен је на Катедри за телематичко инжењерство Универзитета „Карлос трећи Мадридски“, у Мадрид, Шпанији (енг. *Department of Telematic Engineering, University Carlos III of Madrid, Madrid, Spain*). Овај пројекат реализован је током 2013. године.

- Сврха: Мотивација за развој овог софтверског система била је смањивање трошкова развијањем алата за аутоматизовано оцењивање папирних тестова кандидата који ће бити портабилан, чија ће цена опреме и коришћења бити ниска и који ће показати високе перформансе у погледу брзине рада и прецизности. Ово решење је приуштиво малим и средњим образовним институцијама које не могу да поднесу трошкове скупе опреме као што су системи за оптичко читање ознака и особља задуженог за њихово управљање и одржавање. Такође, ово решење је погодно и за образовне институције у којима нема довољно рачунарских ресурса или особља потребног за обезбеђивање мрежне сигурности приликом таквог начина испитивања. Поред тога, систем је погодан и у сценаријима у којима тестови морају бити пренети на другу локацију ради њиховог оцењивања.
- Изглед папирног теста: Папирни тест се састоји из једног листа формулара који може садржати произвољан број питања и одговора, али који мора имати фиксно одређени формат распореда питања и одговора. Текстови питања и њихових одговора не налазе се у оквиру овог формулара, већ су они смештени на посебним листовима. Питања су нумерисана арапским бројевима, док су понуђени одговори означени словима енглеске абетецеде.

Шаблон је дизајниран користећи Латехов (енг. *LaTeX*) описни језик за припрему докумената, али се може користити и други алат, уколико ће произвести сличан формат распореда питања и одговора. Страница формулара садржи секцију за унос идентификације кандидата, секцију за унос изабраних одговора на питања и секцију у којој се налази кодирана варијанта теста. У идентификациону секцију кандидати руком уписују деветоцифрени идентификациони број, при чему се свака цифра уписује у по један од доступних правоугаоника који чине ту секцију.

Секција за упис одговора се састоји од питања распоређених у колонама од по 10 питања, при чему се празни квадрати (одговори) налазе са десне стране броја питања. Тиме је формиран произвољан број мрежа одговора од по 10 редова (питања). Кандидат изабира одговор уписивањем знака икс у одговарајуће поље које одређује једно питање и један његов одговор.

Кандидат има могућност да поништи свој раније изабрани одговор потпуним зацрњивањем поља тог одговора, али не може поново изабрати раније поништени одговор. Варијанта теста кодирана је на бинарни начин коришћењем зацрњених квадрата распоређених у два реда (за вредност 1 односно 0 у бинарном разреду који сваки квадрат представља). Непопуњени формулар са 20 питања и по 4 одговора у формату који систем користи приказује Слика 23.

ID:

--	--	--	--	--	--	--	--	--

	A	B	C	D
1	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>
2	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>
3	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>
4	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>
5	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>
6	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>
7	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>
8	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>
9	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>
10	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>	<div style="border: 1px solid black; width: 40px; height: 20px;"></div>

Слика 23. Непопуњени формулар за упис одговора

- Класе питања: Овај софтверски систем је направљен са идејом да детектује и оцењује питања типа вишеструки избор на папирним тестовима.
- Технологије: Целокупан софтверски систем развијен је коришћењем програмског језика Пајтон, његових стандардних библиотека и екстерних библиотека отвореног кода. Систем користи CSV (енг. *Comma Separated Values*) формат за манипулацију и складиштење добијених резултата. Систем користи Пајтонов *Pygame* пакет за реализацију корисничког интерфејса.
- Области вештачке интелигенције: Овај софтверски систем користи алгоритме компјутерске визије (као што су адаптивно трешхолдовање, бинаризација, ерозија, дилатација итд.) за препроцесирање слике. Ово је неопходно урадити због шума који је настао у процесу штампања или попуњавања теста. Додатно, ове операције решавају проблем дисторзије слике који може настати због положаја камере према папирном формулару.

За детекцију региона од интереса користе се Хјуове трансформације [56] (енг. *Hough transform*). Поред тога, систем користи и сопствено развијене алгоритме за препознавање и потврђивање облика, као што су матрице одговора и квадрати за кодирање варијанте теста. За одређивање идентификационог броја кандидата користе се технике оптичког препознавања карактера (енг. *OCR, Optical Character Recognition*).

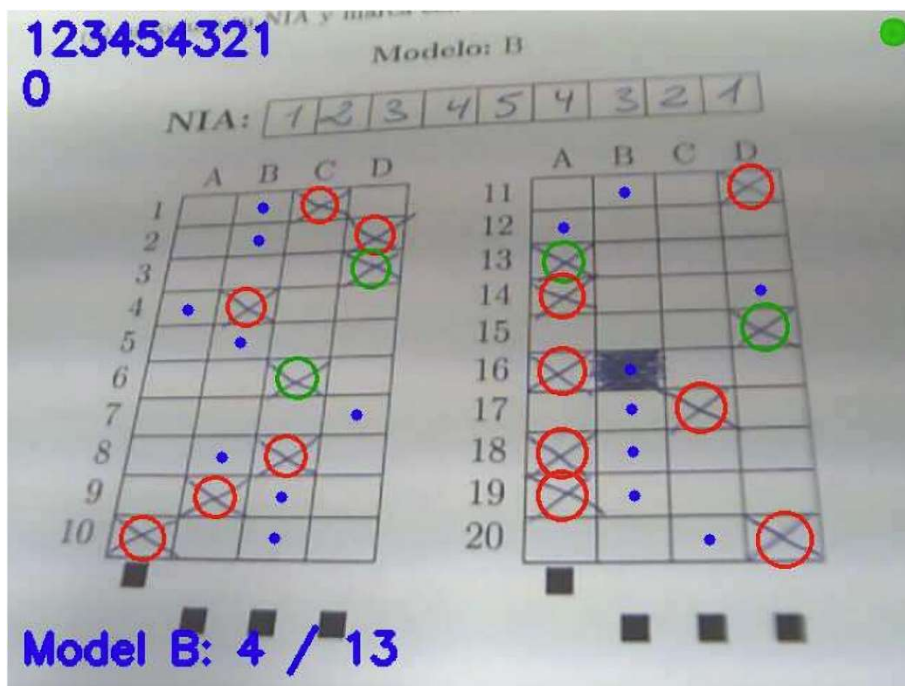
- Структура система: Целокупан процес испитивања састоји се из неколико корака. Најпре се, користећи Латехов описни језик за припрему докумената врши дизајнирање теста у складу са референтним форматом формулара, тако што се додаје мрежа питања и одговора, као и попуњени квадрати за одређивање варијанте теста. Као производ ове компоненте настаје тест у дигиталном облику који се умножава у потребном броју примерака.

Ово умножавање је могуће обавити и коришћењем обичног штампача, али и у центрима задуженим за штампање. Након тога, у процесу израде теста од стране кандидата настаје одређени број попуњених папирних тестова. Затим оператер система приноси камери појединачне попуњене папирне формуларе, тако да цели стану у оквир камере.

За сваки формулар систем хвата континуални ток оквира снимка (енг. *continuous captured frame stream*), који аотира информацијама о идентификационом броју кандидата, детектованим, тачним и нетачним одговорима и варијанти теста. Систем проверава одговоре кандидата тако што упарује шаблон попуњеног симбола икс, чије линије спајају дијагоналне углове квадрата који представља поље одговора. Такође, систем користи фиксну границу да утврди статус зацрњености облика од интереса.

Уколико систем погрешно препозна неки од детектованих података или не изврши детекцију податка који је требало детектовати, оператер система може извршити његову корекцију. Након што оператер потврди да су детектовани подаци и подаци исправљени од стране оператера коректни, за сваки формулар сачува се аотирана слика, док се сви подаци чувају у датотеци у CSV формату. Ова датотека се лако може учитати у систем са циљем измене неке од детектованих информација и њиховог поновног чувања или прегледа од стране оператера система.

На аотираном формулару тамно-зеленим кругом заокружени су одговори који су кандидату у складу са кључем за додељену варијанту шаблона, црвеним кругом су представљени нетачни одговори кандидата, док су плавим попуњеним круговима представљени тачни одговори за питања на која кандидат није одговорио. Тамно-плавом бојом представљени су и утврђени идентификациони број кандидата, који се налази у горњем левом углу, заједно са редним бројем формулара. Истом бојом у доњем левом углу приказани су и детектована варијанта формулара и укупан број тачних и нетачних одговора. Пример аотираног формулара попуњеног од стране кандидата илуструје Слика 24.



Слика 24. Формулар аотиран резултатима процесирања

- Експерименти: Систем је тестиран над 233 папирна теста, при чему је укупно утрошено време обраде износило 2110 секунди. То значи да је у просеку неопходно 9.05 секунди за обраду једног формулара. Систем је испољио изузетно високу прецизност у детекцији и утврђивању зацрњености одговора, која износи 97 %.

Од преосталих 3 % тестова у којима одговори нису успешно детектовани, кандидати нису на исправан начин вршили попуњавање одговора у 2.4 % случајева. У 0.6%

случајева, што износи 29 формулара, одговори су били исправно попуњени, али их систем није исправно детектовао. То значи да је систем ефективно показао 99.4 % успешности у детекцији одговора.

Систем је показао успешност од 85.2 % у детекцији идентификација кандидата, уз коришћење алгоритамског побољшања, којом се детектована идентификација кандидата проверава са базом идентификација кандидата. Без тог побољшања, прецизност у препознавању идентификација кандидата износи само 13.5 %. Такође, изведени су и експерименти у којима су прегледачи ручно прегледали тестове и за то им је било неопходно у просеку 55.3 секунде: од тог времена 41.1 секунду за оцењивање одговора, 12.1 секунду за унос резултата и 2.1 секунду за разврставање теста према његовој варијанти.

- Евалуација:
 - Предности: Систем показује изузетно високу прецизност у детекцији одговора на питања (99.4 %). Поред тога, систем показује високу прецизност у детекцији идентификационих бројева кандидата (85.2 %). Такође, систем показује велику брзину процесања формулара у поређењу са ручним прегледањем (убрзање од 6 пута).

Систем дозвољава креирање шаблона са питањима са произвољним бројем одговора. Такође, систем даје могућност кандидатима да промене свој раније изабрани одговор. Систем представља јефтино и портабилно решење јер, на супрот решењима која користе скенер, користи обичну веб-камеру, која има малу величину и масу и лако се може транспортовати.
 - Недостаци: Систем није отпоран на различите начине попуњавања поља одговора, односно очекује да ће кандидати попуњавати одговоре правилним повлачењем две линије које спајају наспрамне углове поља које се бира као одговор формирајући симбол *икс*. У поређењу са конкурентским решењима сличне архитектуре, систем испољава нешто дуже време обраде формулара. Иако кандидати могу да пониште свој раније изабран одговор, не могу га поново изабрати. Такође, без доступних података о идентификацијама кандидата са којима се могу упарити детектоване вредности о идентификацији кандидата, систем показује само 13.5 % успешности у детекцији идентификација.

3.6. ASSHEP

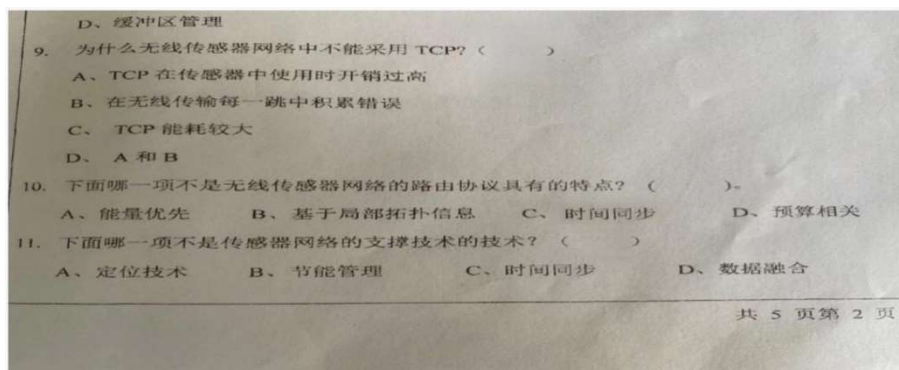
Софтверски систем ASSHEP (енг. *Automatic Scoring System for Handwritten Examination Papers*) је направљен од стране Факултета за електронику и информационо инжењерство Универзитета Фошан, у Фошану, Кини (енг. *School of Electronic and Information Engineering, Foshan University, Foshan, China*). Овај пројекат реализован је током 2021. године.

- Сврха: Мотивација за развој овог софтверског система била је превазилажење проблема субјективности у оцењивању одговора кандидата на папирним тестовима. Такође, поред увођења објективности у процес оцењивања, идеја је била да се наставни кадар растерети заморног посла ручног оцењивања, које је подложно грешкама. Такође, постојала је жеља за превазилажењем ограничења у погледу предефинисаних места за писање одговора од стране кандидата, као и коришћења додатних засебних папира за упис одговора.
- Изглед папирног теста: Не постоје стриктна ограничења приликом формирања папирног теста. Потребно је само да питања буду довољно размакнута, тј. да се

простори које та питања заузимају не преклапају. Понуђени одговори на питања теста налазе се у оквиру простора питања, а не на засебним формуларима.

Кандидати одговарају на питање уписом слова одговора у простор питања. Такође, изабране одговоре на одређено питање кандидати могу да уписују било где у оквиру простора тог питања. Уколико желе да промене свој одговор, кандидати могу поништити стари одговор на различите начине (једном линијом, двома линијама, симболом икс, потпуним прецртавањем итд.) и дописати нови одговор.

Такође, кандидати могу поново дописати свој раније поништен одговор. Кандидати руком уписују свој идентификациони број који се састоји од арапских цифара. Пример изгледа дела папирног теста над којим је систем тестиран илуструје Слика 25.



Слика 25. Пример непопуњеног теста

- Класе питања: Овај софтверски систем је направљен са идејом да детектује и оцењује питања типа повезивање на папирним тестовима. Систем је испробан над тестовима на енглеском и кинеском језику.
- Технологије: Целокупан софтверски систем развијен је коришћењем програмског језика Пајтон, његових стандардних библиотека и екстерних библиотека отвореног кода. Систем користи Мајкрософтов производ Ексел за манипулацију и складиштење добијених резултата.
- Области вештачке интелигенције: Овај софтверски систем користи YOLO [57] (енг. *You Only Look Once*) алгоритам уз коришћење дубоких конволуционих неуралних мрежа. Поред тога, систем користи и алгоритам логистичке регресије за детекцију обухватних правоугаоника.
- Структура система: Целокупан процес испитивања састоји се из неколико делова. Најпре се креира тест у дигиталном облику који се штампа, а затим се копира у потребном броју примерака коришћењем обичног копир апарата или штампача. Након тога, у процесу израде теста од стране кандидата настаје одређени број попуњених папирних тестова.

Затим се на основу сваког од попуњених папирних тестова добија његов дигитални облик, при чему је сваки тест представљен по једном сликом. У свом раду, систем врши процесирање и анотацију слике теста подацима добијеним у процесу детекције региона од интереса. У идентификационој секцији систем најпре препознаје идентификациони број кандидата, који је кандидат уписао оловком.

Након тога, систем проналази оквире простора питања, али их не исцртава. Затим у оквиру сваког питања систем врши детекцију написаних карактера и одређује правоугаонике који ове карактере обухватају, који се и исцртавају око тих карактера.

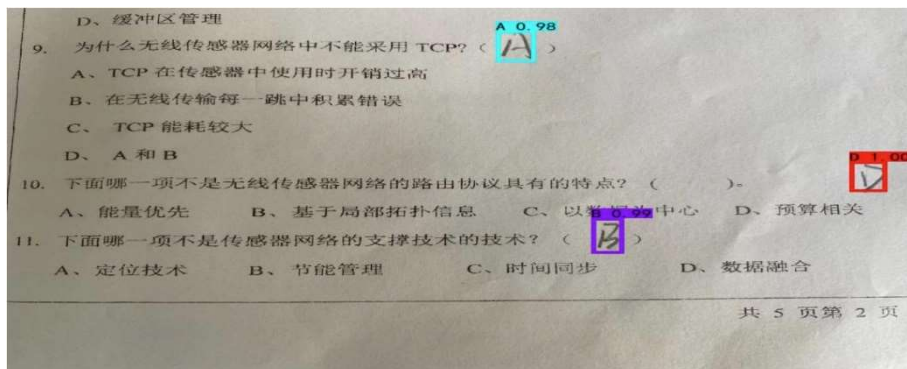
Потом подсистем за предикцију утврђује који карактер се налази у оквиру ових правоугаоника и даје меру сигурности у обављену предикцију.

Након тога се вредности добијене предикцијом проверавају са тачним одговорима на тесту и добијени резултати се уписују у табелу са резултатима. Подсистем за предикцију руком писаних карактера базиран је на YOLO алгоритму. Најпре је потребно на основу сакупљених попуњених папирних тестова пажљиво формирати скупове података, како би могле да се примене технике дубоког учења, а и да би се мрежа боље истренирала.

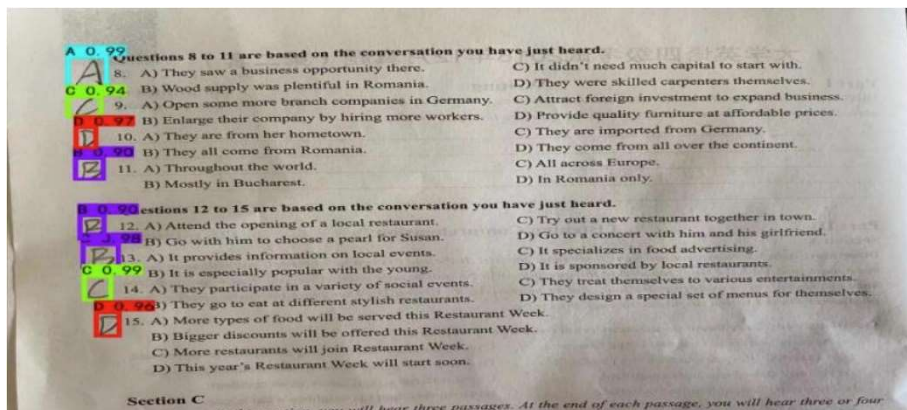
Потом се они деле на тренинг и тест скуп података. Затим се користи анотациони алат да се на сликама тестова из тренинг скупа означе позиције руком уписаних одговора. Потом се из анотационог алата продукује XML датотека, која се конвертује у формат текстуалне датотеке.

Након тога се YOLO алгоритам модификује додавањем категорија одговора на питања у класификацију и врши се тренирање модела на основу тренинг скупа података. Ово доводи до ажурирања тежина YOLO алгоритма. Тренирање се обавља у неколико корака, који укључују предвиђање и класификацију обухватајућих правоугаоника (енг. *bounding boxes classification*), предвиђање унакрсних скала (енг. *cross-scale prediction*) и екстракцију атрибута (енг. *feature extraction*).

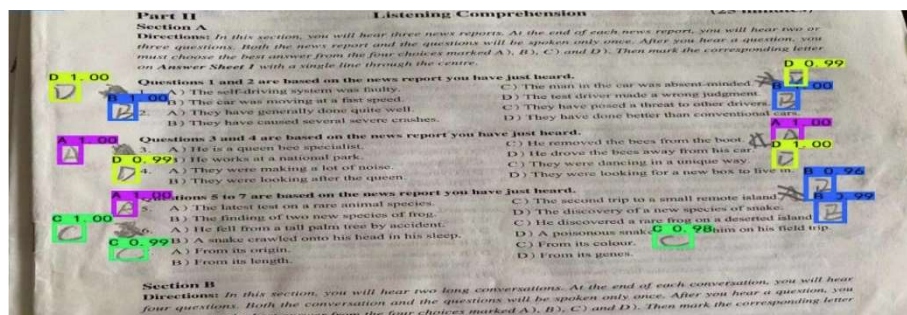
Коришћењем алгоритма логистичке регресије утврђују се локације правоугаоника који обухватају питања на свакој слици. Пример питања са теста на кинеском језику анотираног подацима добијеним у процесу детекције изведеном од стране система приказује Слика 26. Слика 27 илуструје пример питања са теста на енглеском језику анотираног резултатима обраде. Слика 28 показује како се систем понаша у случају прецртавања одговора од стране кандидата.



Слика 26. Тест на кинеском језику анотиран резултатима процесирања



Слика 27. Тест на енглеском језику анотиран резултатима процесирања



Слика 28. Тест на енглеском језику уз прецртавање одговора

- Експерименти: У свом раду, аутори нису навели обим скупа података над којим је извршено тренирање, као ни тестирање система. Такође, аутори нису навели време потребно за обраду једног теста или једног питања на тесту. Поред тога, аутори нису навели ни прецизност коју је систем постигао у процесу детекције података од интереса, већ је само наведено да би унапређења система требало да се тичу и прецизнијих метода идентификације података од интереса.
- Евалуација:
 - Предности: Овај софтверски систем подржава могућност да кандидати на било ком месту у оквиру простора питања пишу своје одговоре. Такође, систем дозвољава и да се раније написани одговор поништи прецртавањем и на тај начин допише неки други одговор. Поред тога, систем не захтева додатне формуларе за упис одговора, већ се текст одговора налази у простору питања, заједно са текстом питања.
 - Недостаци: Потребна су унапређења у погледу коришћења овог система, да би он могао лакше и брже да се покрене. Такође, систему су потребне напредније методе за сегментацију уписаних бројева и карактера, као и ефикаснији модел за њихову предикцију.

3.7. AGHAS

Софтверски систем AGHAS (енг. *Automatic Grading for Handwritten Answer Sheets*) је направљен на Универзитету „Принц Мохамед бин Фад“, у Ал Кобру, Саудијској Арабији (енг. *Prince Mohammad bin Fahd University, Al Khobar, Saudi Arabia*). Овај пројекат реализован је током 2019. године.

- Сврха: Мотивација за развој овог софтверског система била је аутоматизација процеса прегледања папирних тестова, чиме би се смањило време потребно за ручно прегледање ових тестова, као и постигло смањење грешака насталих у процесу ручног прегледања. Грешке које настају у процесу ручног прегледања тестова узроковане су монотонošћу и тривијалношћу самог процеса. Стога, развијено је портабилно решење које користи персонални рачунар, преносиви скенер и софтвер који служи за аутоматско бодовање резултата.
- Изглед папирног теста: Папирни тест се креира уз коришћење обичног процесора текстуалног докумената. У заглављу папирног теста налази се идентификациона секција кандидата у којој се уписују и име и презиме и идентификациони број кандидата. Затим су испод идентификационе секције наведена питања теста.

Одговори на питања налазе се у оквиру простора за питања, заједно са текстом сваког од питања, те нема одвојених формулара за упис одговора. За упис одговора предвиђени су правоугаони облици који се налазе у линији појединачних одговора и поравнати су

уз десну ивицу папира. Сви правоугаони облици за упис одговора су вертикално поравнати тако да се налазе један испод другог.

Кандидати оловком уписују слова енглеске абецеде, односно арапске бројеве, који се налазе са леве стране сваког од појединачних одговора, у њихове одговарајуће правоугаоне облике за упис одговора. Кандидати немају могућност поништавања одговора нити избора неког другог одговора. Пример попуњеног папирног теста који овај систем користи приказује Слика 29.

Name	SARAH AL KHALIDI	ID	201500470
Quiz	1	Marks	

Read carefully

Multiple Choice Questions.

- A number followed by one hundred zeroes is known by what name?
A. Googol B. Megatron C. Gigabit D. Nanamote A
- Which insect inspired the term "Computer bug"?
A. Beetle B. Cockroach C. Fly D. Moth D
- In the "Road Runner and Coyote" cartoons, what famous sound does the Road Runner make?
A. Ping! Ping! B. Beep! Beep! C. Aogaa! Aogaa! D. Vroom! Vroom! B
- Which author has written the book "The Scarlet Letter"?
A. William Shakespeare B. Charles Dickens C. Nathaniel Hawthorne D. Jane Austen C
- "A man is but a product of his thoughts. What he thinks he becomes." These words were said by?
A. Mahatma Gandhi B. Nelson Mandela C. Martin Luther King Jr. D. Dalai Lama A
- The first day of the 21st century was?
A. 1st January 2000 B. 1st January 1901 C. 1st January 2001 D. 1st January 1900 C

Answer either T for True or F for False.

- The color of a slug's blood is red. F
- Musical hacking began with the invention of cassette tape. T
- Fortune cookies were invented in China. F
- A group of unicorns is known as a blessing. T
- The mathematical equivalent to dyslexia is called dyscalculia. T
- The temperature at the tip of a burning cigarette is 900C. T
- The Calculus was discovered before the founding of Harvard University. F
- The first passengers of a hot air balloon were a rooster, a sheep and a duck. T

Match the following.

1. Which was the first country to give women the right to vote?	a. A murder	3
2. The painting "The Starry Night" was a creation of?	b. Leonardo Da Vinci	5
3. What are a group of crows known as?	c. Canada	6
4. What are a group of frogs known as?	d. Vincent Van Gogh	2
5. The painting "Mona Lisa" was a creation of?	e. New Zealand	1
6. Which country's national day is the 1 st of July?	f. An army	4

Слика 29. Пример попуњеног теста

- Класе питања: Овај софтверски систем је направљен са идејом да детектује и оцењује питања типа повезивање на папирним тестовима.
- Технологије: Целокупан софтверски систем развијен је коришћењем програмског језика Пајтон, његових стандардних библиотека и екстерних библиотека отвореног кода. Такође, систем користи и програмски језик Матлаб за сегментацију слика. Поред тога, систем користи Мајкрософтов производ Ексел за манипулацију и складиштење добијених резултата.
- Области вештачке интелигенције: Овај софтверски систем користи технике дубоког учења као што су конволуционе неуралне мреже (енг. *Convolutional Neural Networks*) за предикцију ручно написаних одговора у виду слова и карактера.
- Структура система: Најпре се врши дизајнирање теста коришћењем арбитрарног процесора текстуалног документа, чиме настаје дигитални облик теста. Затим се врши конверзија дигиталног облика теста у папирни уз помоћ обичног штампача. Потом се

врши умножавање папирног облика теста у потребном броју примерака уз коришћење копир машине или штампача.

Након тога се приступа процесу испитивања кандидата у коме настају попуњени папирни тестови. Затим се портабилни скенер користи да сачува попуњени папирни тест у дигиталном облику у Разбери пај (енг. *Raspberry pi*) компоненту. Тамо се врши конверзија слике у *jpeg* формат и бежичним путем шаље на персонални рачунар користећи *SSH* (енг. *Secure Shell Protocol*) сервер.

Персонални рачунар садржи код писан у Матлабу који служи за сегментацију слике, да би се добиле појединачне слике одговора које садрже слова и бројеве писане руком. Сегментација слика одговора врши се упаривањем скенираног облика теста са већ предефинисаним непопуњеним шаблоном теста који садржи локације сегмената одговора на питања. Након тога се исечени сегменти чувају у локалном директоријуму.

Персонални рачунар садржи и моделе дубоке конволуционе неуралне мреже писане на Пајтону, који су раније тренирани и учитавају се по потреби. Они служе за детекцију одговора кандидата на основу раније добијених сегмената одговора, након чега се детектовани одговори упоређују са кључем тачних одговора. Затим се врши бодовање теста кандидата и резултати се уписују у табелу.

- Експерименти: Подаци за тестирање система представљају 250 папирних тестова, од којих се сваки састоји од 20 питања са по једним одговором, што даје укупно 5000 слика одговора. Међутим, нису све слике успешно сегментиране, односно сегментирано је 4871 слика, тј. постигнута је успешност од 97.42 % у процесу сегментације слика одговора. Од тога је 80 % слика искоришћено у процесу тренирања модела, док је преосталих 20 % искоришћено у процесу тестирања.

У процесу предикције руком написаних одговора трениран је велики број модела са различитим хиперпараметрима као што су величина серија (енг. *batch size*) и број епоха. Најбоље се показао модел са величином серије 50 и идентичним бројем епоха, који је досегао 92.86 % успешности у предикцији над тестним скупом података. Ово је остварено по цену нешто дужег времена потребног за тренирање модела.

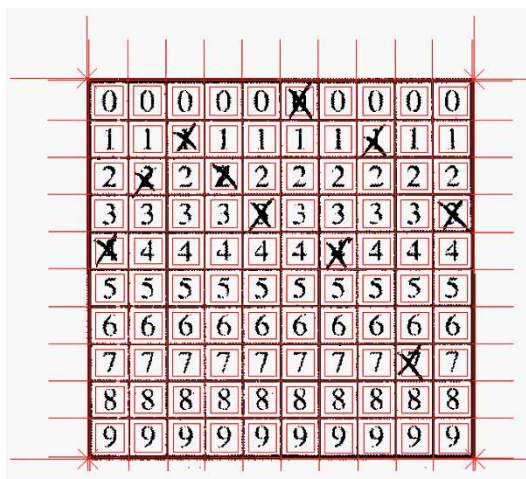
- Евалуација:
 - Предности: Систем је постигао високу прецизност у предикцији руком написаних слова енглеске абетеде и арапских бројева која износи 92.86 %. Такође, систем је постигао високу успешност у погледу сегментације слика одговора на тесту која износи 97.42 %.
 - Недостаци: Систем не пружа кандидатима могућност поништавања одговора или избора раније поништеног одговора. Такође, систем показује неотпорност на ротацију формулара, што је један од узрока неуспешне сегментације слика одговора, јер систем приликом сегментације разматра сваку вредност пиксела шаблона. То значи да ће и релативно мала разлика у оријентацији између шаблона и тестног примерка довести до одбацивања сегмента.

3.8. ARHC

Софтверски систем ARHC (енг. *Automatic Recognition of Handwritten Corrections*) је направљен на Факултету електротехнике и рачунарства Универзитета у Загребу. Овај пројекат је реализован током 2014. године.

- **Сврха:** Мотивација за развој овог софтверског система била је решавање проблема аутоматизованог прегледања папирних тестова у окружењима са великим бројем кандидата. Такође, овим системом је постигнуто убрзање целог процеса прегледања и смањивање броја грешака насталих у процесу ручног прегледања. Поред тога, систем је додатно аутоматизован смањивањем учесталости ситуација у којима је потребна асистенција оператера система. У истраживањима аутора показало се да отприлике 20 % свих тестова захтева некакву ручну интервенцију насталу услед грешке.
- **Изглед папирног теста:** Папирни тест се састоји из једног листа (формулара) који може да подржи до 23 питања и 6 одговора и засебних листова на којима се налазе текстови питања и одговори на та питања. У заглављу формулара налазе се идентификационе информације које кандидат уписује, као што су његово име и презиме. Такође, на сваком тесту налази се и бар-код који кандидат већ поседује од раније, који кандидат лепи, и који ће јединствено одредити кандидата.

У неким ситуацијама, кандидати зацрњују свој десетоцифрен идентификациони број зацрњивањем појединачних квадрата у за то предвиђену матрицу квадрата, која има по 10 врста и 10 колона, за 10 различитих арапских цифара. Одговори на формулару за одговоре су распоређени у матричној структури, при чему су питања представљена појединачним врстама, док непопуњени кругови једне врсте представљају појединачне одговоре на питање представљено том врстом. Кандидат изабира одговор зацрњивањем круга у одговарајућој врсти и колони. Пример зацрњене идентификационе матрице илуструје Слика 30.



Слика 30. Идентификациона матрица у процесу обраде

Међутим, често се јавља ситуација у којој кандидат погрешно зацрни круг због тога што је промашио врсту или колону. Такође, кандидати се често предомисле и имају потребу да промене раније изабрани одговор или да га у потпуности пониште. Због тога је на формулару уведена мрежа квадрата од највише 23 врсте и тачно 3 колоне, која служи да кандидат оловком упише једно од слова енглеске абегеде које одговара новом одговору или цртицу, уколико не жели да одговара на то питање.

Такође, кандидат може да упише и симбол икс, уколико сматра да ниједан од понуђених одговора није тачан. Поред тога, кандидат је дужан и да зацрни круг у колони „Грешка“ и тиме сигнализира систему да је дошло до промене одговора. Аутори система рачунају на највише три исправке, па је број колона ове матрице квадрата тачно 3. Стога, иако је на слици приказан пример теста који би квалификовао овај систем у класу питања за вишеструке одговоре, тај део система није тема у овом потпоглављу, већ део система

задужен за препознавање руком уписаног одговора у предефинисана места. Пример попуњеног папирног формулара теста са бар-код идентификацијом приказује Слика 31.

Digitalna logika (19674)
Međuispit (2013-11-26), AG 2013/2014, zimski semestar

Prezime, ime: _____
Dvorana: B1 (27)
Grupa: A B C D
○ ○ ● ○

Zadaci

Broj	A	B	C	D	E	F	Greška
1	○	○	●	○	○	○	
2	○	○	○	○	○	○	
3	○	○	○	○	●	○	
4	○	○	○	○	○	○	
5	○	○	○	○	○	○	
6	○	○	○	○	○	○	A
7	○	○	○	○	○	○	C
8	○	○	○	○	○	○	
9	○	○	○	○	○	○	
10	○	○	○	○	○	○	
11	○	○	○	○	○	○	
12	○	○	○	○	○	○	
13	○	○	○	○	○	○	
14	○	○	○	○	○	○	
15	○	○	○	○	○	○	
16	○	○	○	○	○	○	
17	○	○	○	○	○	○	
18	○	○	○	○	○	○	-
19	○	○	○	○	○	○	C
20	○	○	○	○	○	○	

Broj A B C D E F Greška

Слика 31. Пример попуњеног формулара

- Класе питања: Овај софтверски систем испољава могућност детекције и оцењивања питања типа повезивање на папирним тестовима.
- Технологије: Целокупан софтверски систем развијен је коришћењем програмског језика Јава, његових стандардних библиотека и екстерних библиотека отвореног кода.
- Области вештачке интелигенције: Овај софтверски систем користи алгоритме компјутерске визије (као што је бинаризација) за препроцесирање слике. За детекцију табеле са руком уписаним одговорима користе се Хјуове трансформације [58]. Поред тога, користе се и сопствено развијени алгоритми за препознавање и потврђивање облика, као што су минимални обухватни правоугаоници око облика од интереса и дистанца ивица (енг. *edge distance*). Такође, користе се и методи случајне шуме (енг. *random forest classifier*) за детекцију припадности класи претходно сегментованог симбола уписаног руком.
- Структура система: Целокупан процес испитивања састоји се из неколико делова. Најпре се креира шаблон теста у дигиталном облику коришћењем обичног процесора текстуалних докумената. Затим се креира папирни примерак шаблона користећи обичан штампач.

Потом се у центру за умножавање на основу папирног примерка направи потребан број штампаних примерака теста. Након тога, у процесу израде теста од стране кандидата

настаје одређени број попуњених папирних тестова. Затим се у процесу скенирања на основу папирних примерака теста добијају скенирани примерци теста.

Они представљају улазне податке за реализовани систем. Над првим примерком скенираног теста оператер система мора да подеси оријентирне параметре у оквиру шаблона који ће се користити у процесирању првог и преосталих тестова. Идентификациона матрица кандидата, уколико је присутна на тесту, ручно се аотира од стране оператера само на првом тесту.

Ово се ради само да би се поставио регион оквирне локације идентификационе матрице. Прецизна локација идентификационе матрице проналази се коришћењем технике линеарне регресије, чиме се добијају спољашње ивице матрице идентификације. Након тога се на једноставан начин одређују квадрати табеле идентификација, поделом табеле на по 10 врста и 10 колона.

Потом се зацрњеност квадрата табеле идентификација одређује утврђивањем односа заступљености црних пиксела у сваком квадрату и експериментално утврђене границе. Уколико је процедура детекције идентификације неуспешна, смањују се маргине ћелије за по 10 % са сваке стране. Да би се детектовала табела оловком уписаних симбола користи се Хјуова трансформација.

Након тога се ћелије табеле ручно уписаних одговора сегментирају. Уз помоћ методе случајних шума врши се предикција оловком написаног симбола. За прецизну детекцију написаног симбола у оквиру квадрата табеле оловком уписаних одговора користи се техника дистанце ивица.

Током процеса аутоматизованог оцењивања оператер система мора бити присутан у случају ручне интервенције. Ово је неопходно због тога што студент може изабрати неколико одговора или у случају да систем није у стању да препозна руком написан одговор кандидата. Након тога се добијени резултати уписују у датотеку са резултатима која се налази у специфицираном директоријуму.

- Експерименти: У процесу тренирања коришћени су руком написани симболи од стране 289 људи, док су симболи написани од стране 15 људи коришћени за тестирање. Сваки човек је 32 пута написао све дозвољене симболе (А, В, С, D, Е, F, X и -). Стога, прикупљено је 264 симбола од сваког човека, односно 76296 тренинг симбола и 3960 тест симбола, рачунајући и симболе заглавља.

Над овим скупом података, користећи класификатор случајних шума и користећи оптимални (уз који је постигнут најбољи резултат) коефицијент сигурности од 0.59, успешно је класификовано 92.8 % симбола, погрешно је класификовано 1.2 % симбола, док је ручну интервенцију захтевало 6.1 % симбола. Уз коришћење оптималног коефицијента разлике сигурности између припадности симбола највероватнијој класи и првој следећој, а који износи 0.37, добијен је сличан резултат од 92.6 % коректних класификација, 1 % погрешних класификација, а 6.4 % је захтевало ручну интервенцију. У детекцији идентификације кандидата коришћен је скуп података од 148 аотираних студентских матрица, при чему је систем успео да коректно детектује све идентификационе бројеве кандидата.

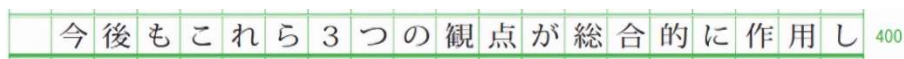
- Евалуација:
 - Предности: Систем је показао високу прецизност у предикцији руком написаних симбола која износи најмање 92.6 %. Поред тога, систем је показао високу прецизност у предикцији идентификација кандидата која износи 100 %. Такође, задржана је раније постигнута брзина обраде папирних формулара.

- Недостаци: Нису наведени подаци о времену неопходном за процесирање једног теста, као ни подаци о конфигурацији машине над којом су експерименти вршени.

3.9. FASAS

Софтверски систем FASAS (енг. *Fully Automated Short Answer Scoring*) је направљен у сарадњи Токијског универзитета за агрокултуру и технологију, у Токију, Јапан и Националног центра за универзитетске пријемне испите, у Токију, Јапан (енг. *Tokyo University of Agriculture and Technology, Tokyo, Japan; The National Center for University Entrance Examinations, Tokyo, Japan*). Пројекат је реализован током 2022. године.

- Сврха: Мотивација за развој овог софтверског система била је елиминисање ручног посла у оцењивању питања типа кратак одговор. Потребно је немало време и труд за конверзију текста написаног руком у електронски облик, како би специјализовани системи за оцењивање одговора који раде над подацима у дигиталном облику могли да оцене те одговоре. Стога, развијен је систем који у потпуности поседује могућност оцењивања руком написаних одговора на питања типа кратак одговор и који аутоматски врши конверзију одговора у текстуални дигитални облик.
- Изглед папирног теста: Папирни тестови које систем користи представљају улазне тестове који се полажу на пријемним испитима за универзитете широм Јапана, како државне, тако и приватне. Они су стандардизовани и састављени од стране Националног центра за универзитетске пријемне испите, независне организације у Јапану. Одговори на питања уписују се на посебном обрасцу за одговоре. Кандидати одговоре на питања типа кратак одговор уписују у мрежу квадрата, тако да се сваки појединачни знак на јапанском језику уписује у посебан квадрат. Пример попуњеног дела теста са приказом места за упис одговора приказује Слика 32.



Слика 32. Пример попуњеног дела теста

- Класе питања: Овај софтверски систем испољава могућност детекције и оцењивања питања типа кратак одговор на папирним тестовима.
- Технологије: За овај софтверски систем нису наведене софтверске технологије које су коришћене.
- Области вештачке интелигенције: Овај софтверски систем користи технике дубоког учења вештачке интелигенције као што су конволуционе неуралне мреже. Такође, користи и технике и моделе обраде природног језика као што су бидирекционо енкодоване репрезентације из трансформера.
- Структура система: Процес дизајнирања теста, његовог умножавања у потребном броју копија, дистрибуирања институцијама које врше испитивање кандидата и добијање дигиталног облика попуњених тестова није у надлежности овог система. Тренирање система за детекцију написаних карактера обављено је над базом знакова писаних на јапанском језику. Она се састоји од 9 скупова података, при чему су појединачни знакови писани у квадратима, слично као и на пријемном испиту на универзитетима у Јапану.

Систем користи склоп састављен од више добро познатих модела конволуционих неуралних мрежа за детекцију знакова написаних на јапанском језику. Како би се овај склоп мрежа тренирао и да не би дошло до преобучавања неуралних мрежа, различите трансформације над сликама (као што су ротирање, скалирање, додавања шума итд.) су

изведене. Такође, због вишесмислености која настаје у детекцији неких од знакова, користи се Н-грам језички модел да изврши исправку погрешно предиктованог знака на основу лингвистичког контекста текста. За оцењивање написаног одговора кандидата користи се класификациони модел са више лабела, који је фино подешен користећи бидирекционо енкодоване репрезентације из трансформера (енг. *BERT, Bidirectional Encoder Representations from Transformers*), који је претходно трениран користећи јапанску верзију странице енциклопедије Википедија.

- Експерименти: Приликом одређивања евалуације система коришћена је метрика у виду QWK (енг. *Quadratic Weighted Kappa*) индекса, што је често коришћена метрика у евалуацији квалитета система за оцењивање питања типа кратак одговор. Што је већа вредност овог индекса, то систем показује боље перформансе. Ова метрика посматра разлике у предикцији између предвиђених одговора и тачних одговора.

У експериментима су коришћени тестови са пријемних испита за упис универзитета у Јапану из 2017. и 2018. године. Оба ова теста садржала су по 6 питања типа кратак одговор. Број одговора кандидата који је процесирао по сваком питању износио је између 58 159 и 67 332.

Тестирање је спроведено над скуповима од по 500, 1000, 5 000, 10 000, 50 000 и са целокупним скупом података, за свако питање овог типа засебно. Очекивано, највећи QWK индекси показани су коришћењем целокупног скупа података. Посматрајући такве случајеве, најмањи QWK индекс по питању износио је 0.86, док је највећи индекс износио 0.94.

- Евалуација:
 - Предности: Експерименти су спроведени над великим бројем реалних података. Изабрани софтверски систем показује веома висок ниво прецизности изражен у QWK индексу у опсегу од 0.86 у најгорем до 0.94 у најбољем случају. Ово показује да је систем упоредив са ручним прегледањем у погледу прецизности. Систем је додатно унапређен могућношћу исправке погрешно детектованог знака на основу лингвистичког контекста.
 - Недостаци: Понекад се дешава да систем не конвергира у процесу учења, иако му се обезбеди велика количина података (реда 50 000 примерака). Такође, нису наведени подаци о перформансама система у погледу брзине његовог рада.

3.10. ISSHSA

Софтверски систем ISSHSA (енг. *Intelligent Scoring System for Handwritten Short Answers*) је направљен у сарадњи Факултета за софтвер у Јужној Кини Универзитета технологије, у Гванџуу, Кини и Факултета медицинског информационог инжењерства Универзитета кинеске медицине у Гванџуу, Кина (енг. *School of Software South China, University of Technology Guangzhou, China; College of Medical Information Engineering, Guangzhou University of Chinese Medicine, Guangzhou, China*). Пројекат је реализован током 2020. године.

- Сврха: Мотивација за развој овог софтверског система била је смањивање субјективности у процесу ручног оцењивања питања типа кратак одговор на папирним тестовима. Такође, идеја је била да се реши проблем слабе ефикасности и прецизности у процесу ручног оцењивања, као и да се смање трошкови код ручног прегледања тестова. Стога, реализован је софтверски систем са циљем аутоматизације процеса оцењивања питања типа кратак одговор на папирним тестовима.

- Изглед папирног теста: Папирни тест је дизајниран у складу са националним стандардизованим форматом папирног теста који се користи у Кини за полагање пријемних испита на факултетима. За свако питање постоји простор за одговоре који је правоугаоног облика. Простор за упис одговора је ограничен и кандидати не би требало да пишу своје одговоре ван овог простора.
- Класе питања: Овај софтверски систем испољава могућност детекције и оцењивања питања типа кратак одговор на папирним тестовима.
- Технологије: Целокупан софтверски систем развијен је коришћењем програмског језика Пајтон, његових стандардних библиотека и екстерних библиотека отвореног кода. Систем користи мобилну апликацију за приказ резултата кандидатима, развијену у програмском језику Пајтон.
- Области вештачке интелигенције: Овај софтверски систем користи алгоритме компјутерске визије (као што су бинаризација, адаптивно трешхолдовање, Гаусова филтрација итд.) за препроцесирање слике. Ово препроцесирање се ради у циљу боље детекције облика од интереса на слици.

Поред тога, користи се и техника детекције ивица (енг. *Canny edge detection*) за утврђивање правоугаоних региона уписа одговора у оквиру питања. За детекцију руком написаног одговора користе се технике дубоког учења као што су конволуционе неуралне мреже, као што је комбиновани модел HCCR-GoogLeNet (енг. *Handwritten Chinese Character Recognition*). Такође, за израчунавање семантичке сличности између два текста користи се модел конволуционе мреже MPCNN (енг. *Max-Pooling Convolutional Neural Network*).

- Структура система: Овај софтверски систем састоји се из специјалног формулара за унос одговора, мобилног интелигентног терминала или скенера који има могућност добијања дигиталног облика на основу попуњеног папирног теста. Поред тога, саставни део система је и персонални рачунар или сервер који комуницира са базом података за смештање резултата. Систем се састоји и из већег броја модула, као што су модул за сегментацију простора одговора, модул за детекцију руком написаног текста, модул за семантичко препознавање и поређење текста.

Процес дизајнирања теста, његовог умножавања у потребном броју копија и дистрибуирања институцијама које врше испитивање кандидата није у надлежности овог система. У процесу функционисања система најпре се коришћењем мобилног интелигентног терминала или скенера врши добијање дигиталног облика попуњеног папирног теста. Затим се дигитални облик теста у виду слике шаље на персонални рачунар или сервер, на коме се налази и систем за аутоматизовано оцењивање тестова.

Потом се коришћењем модула за сегментацију простора одговора врши идентификација простора за одговор. Након тога се коришћењем модула за детекцију руком написаног текста врши идентификација руком написаног текста. На самом крају се врши идентификација речи одговора који се пореде са тачним одговорима.

Тачни одговори су раније учитани у одређени директоријум персоналног рачунара на којем се систем налази. У случају писаног одговора са више речи врши се семантичко поређење идентификованог одговора са тачним одговором. Ово се изводи уз помоћ модула за семантичко препознавање и поређење текста.

- Експерименти: За обучавање модела за детекцију руком писаног одговора на папирном тесту коришћене су базе података CASIA-HWDB1.0 (DB1.0) и CASIA-HWDB1.1 (DB1.1). Скуп података за тестирање преузет је са интернационалног такмичења за

препознавање рукописа написаних на кинеском језику (енг. *ICDAR Chinese handwriting recognition competition; International Conference on Document Analysis and Recognition*) одржаног 2013. године.

Тренинг подаци су измешани и прилагођени коришћеној архитектури конволуционе неуралне мреже. Над тестним подацима постигнута је прецизност у детекцији руком написаног текста и до 99.67 %. За тестирање семантичке сличности и поређења текста коришћени су подаци у виду тројки (први текст, други текст, сличност) прикупљени у периоду од 2012. до 2015. године, чији укупан број износи 23 778 оваквих тројки.

За тренирање модела за препознавање семантичке сличности искоришћено је 90 % ових података, а преосталих 10 % података је искоришћено у сврхе тестирања. Постигнута прецизност модела за препознавање семантичке сличности и поређења текста износила је и до 74 %. Машина коришћена за тестирање поседује 16 гигабајта радне меморије, Интелов процесор *i7-7700K @4.5GHz* са 4 физичка и 8 виртуелних језгара и графичку картицу *GTX 1080Ti*.

- Евалуација:
 - Предности: Прецизност модула за детекцију руком написаног текста износи веома високих 99.67 %. Такође, систем је трениран над великим скупом података, који је представљен корпусом од 1G података прикупљених са енциклопедије Википедија.
 - Недостаци: Прецизност модула за утврђивање семантичке сличности између два текста износи до 74 %. Такође, број корака у обучавању је изузетно велики и износи преко 40 000.

3.11. TSAWR

Софтверски систем TSAWR (енг. *Thai Short Answer Word Recognition*) је направљен у сарадњи Индијског статистичког института, у Калкути, Индији, Академије за информационе технологије на Универзитету „Џејмс Кук“, у Кернсу, Аустралији и Технолошког универзитета у Сиднеју, Аустралији (енг. *Indian Statistical Institute, Kolkata, India; Information Technology Academy, James Cook University, Cairns, Australia; School of Software, University of Technology Sydney, Australia*). Пројекат је реализован током 2020. године.

- Сврха: Мотивација за развој овог софтверског система била је решавање проблема препознавања и оцењивања одговора на кратка питања, односно аутоматизација процеса евалуације одговора на кратка питања. Ово има за циљ да смањи време које је потребно за ручно оцењивање оваквог типа питања. Додатно, систем је намењен образовним институцијама које немају услова за спровођење испитивања кандидата путем рачунара. Такође, систем има могућност препознавања уписаних презимена и имена кандидата. Поред тога, систем испољава могућност препознавања и верификације потписа кандидата, што доказује да неки тест припада баш том кандидату, чији је потпис написан.
- Изглед папирног теста: Папирни тест има такву структуру да се састоји од три компоненте у којима се пише текст. Те компоненте служе за упис презимена и имена кандидата, потписа кандидата и одговора питања типа кратак одговор. Постоји десет питања типа кратак одговор на овом тесту, чији одговори се састоје од пар речи (између једне и три речи). Пример написаних имена и презимена приказује Слика 33. Пример написаних кратких одговора на тесту илуструје Слика 34.

Genuine Name Components		Forged Name Components	
First Name	Last Name	First Name	Last Name
ปอล	ปาน	ปอล	ปาน
ปอล	ปาน	ปอล	ปาน

Слика 33. Руком написана имена и презимена

Word	Handwritten Word Sample 1	Handwritten Word Sample 2	Handwritten Word Sample 3
Information	Information	Information	Information
Input	Input	INPUT	Input
Programming	Programming	Programming	PROGRAMMING
Database	Database	database	Database
Misspelt words	Datababase	Lazee	Languge

Слика 34. Руком написани кратки одговори

- Класе питања: Овај софтверски систем испољава могућност детекције и оцењивања питања типа кратак одговор на папирним тестовима.
- Технологије: За овај софтверски систем нису наведене софтверске технологије које су коришћене.
- Области вештачке интелигенције: Овај софтверски систем користи технике дубоког учења као што су конволуционе неуралне мреже за детекцију презимена и имена и верификацију потписа. Коришћене су технике фракционог максималног груписања (енг. *fractional max pooling*) и нормализација слике базирана на моментима (енг. *image moments-based size normalisation*) за тренирање овог модела. Такође, коришћене су технике компјутерске визије са подржаним операцијама за препроцесирање слике (као што су нпр. дилатација, отварање, затварање итд.).
- Структура система: Процес дизајнирања теста, његовог умножавања у потребном броју копија, дистрибуирања институцијама које врше испитивање кандидата и добијање дигиталног облика папирног теста није у надлежности овог система. Најпре се слика теста подели на квадрате матрице која садржи по 9 врста и 9 колона, при чему ширина и висина сваког квадрата износи по 15 пиксела. За сваки овај квадрат се врши израчунавање атрибута, при чему се за сваки пиксел проверава да ли је део руком написаног одговора.

На основу слика коришћених у процесу тренирања креира се база података од 86 атрибута. Затим се ови исти атрибути издвајају из слика коришћених за тестирање и врши се њихово поређење унакрсном корелацијом (енг. *cross-correlation*) са атрибутима слика коришћених у процесу тренирања. Тестном пикселу се додељује класа највише унакрсне вредности.

За сваку слику време неопходно за класификацију сваког пиксела те слике износи око 90 минута. Након тога се врши дилатација слике да би се елиминисали пиксели који нису били део класификације. Овако обрађена слика се филтрира да би садржала компоненте од интереса, чије су ширина и висина најмање 50 пиксела.

Потом се примењују морфолошке операције затварања и отварања да би се попуниле рупе у компонентама од интереса. Затим се све компоненте од интереса групишу правоугаоним оквиром који ограничава компоненту коришћењем својства хоризонталног преклапања између компоненти. Идентификовани правоугаони оквири

служе да се са оригиналне слике пронађу границе руком написаног текста, односно речи.

- Експерименти: У експериментима су коришћена 3 скупа података формирана подацима добијеним од 100 волонтера. Скуп података за презиме и име кандидата, као и за потпис, формиран је тако што су волонтери 30 пута писали своје презиме и име (на два места) и потпис у за то предвиђена места. Они су ове податке писали по 10 пута заредом, а затим су правили кратку паузу од неколико минута.

Затим су овај поступак понављали 3 пута. Овим је скуп презимена и имена садржао 6 000 примерака, а скуп потписа 3 000 примерака. На основу оригиналних презимена и имена креирано је још по 12 фалсификованих за обе компоненте презимена и имена, па је скуп података са презименима и именима садржао укупно 8 400 примерака. На основу оригиналних потписа креирано је још по 24 потписа за сваког волонтера, па је скуп података са потписима садржао укупно 5 400 примерака.

За утврђивање степена препознавања презимена и имена кандидата, као и његовог потписа, коришћена је метрика прецизности у препознавању само презимена, само имена и комбинованог текста презимена и имена. Систем је испољио прецизност од 99.84 % у процесу утврђивања написаног имена и презимена, прецизност од 99.98 % у детекцији потписа, а прецизност од 7.1 % у верификацији потписа. За утврђивање степена препознавања одговора на питања типа кратки одговор коришћена је Ф-мера (енг. *F-measure*). Систем је показао прецизност од 91.12 % у процесу детекције одговора на кратка питања, а вредност 0.67 за Ф-меру.

- Евалуација:
 - Предности: Систем је показао високу прецизност у детекцији руком написаног презимена и имена кандидата, која износи 99.84 %. Такође, систем је показао високу прецизност у детекцији потписа, која износи 99.98 %. Поред тога, систем је показао прецизност од 91.12 % у детекцији руком написаног одговора на питања типа кратки одговор.
 - Недостаци: Потребан је бољи алгоритам за предикцију написаног одговора на питања типа кратки одговор, јер је Ф-мера износила неимпресивних 0.67. Такође, нису наведене перформансе система у погледу времена потребног за обраду једног теста.

3.12. HSAES

Софтверски систем HSAES (енг. *Handwritten Short Answer Evaluation System*) је направљен на Катедри за информатику и инжењерство „Мар Атанасијус“ колеџа за инжењерство, у Котамангаламу, Керали, Индији (енг. *Department of Computer Science and Engineering, Mar Athanasius College of Engineering, Kothamangalam, Kerala, India*). Пројекат је реализован током 2018. године.

- Сврха: Мотивација за развој овог софтверског система била је смањивање трошкова ангажовања људи неопходних за ручно оцењивање питања типа кратки одговор на папирним тестовима. Поред тога, ручно оцењивање је неефикасно, а додељена оцена зависи од искуства, знања, емоције и енергије прегледача. Стога, често постоји висока девијација у оценама додељеним истим кандидатима од стране различитих прегледача. Чак се дешава да исти тест кандидата исти прегледач може различито оценити, уколико га оцењује више пута. За разрешење ових проблема приступило се развијању аутоматизованог система за оцењивање питања типа кратки одговор.

- Изглед папирног теста: Папирни тест који овај систем користи нема стриктно одређен формат. Ипак, постоји неколико правила која треба да се поштују приликом дизајнирања тестова. Простори за питања треба да буду раздвојени једни од других, односно да се не преклапају. Простор за одговоре треба да се налази у оквиру простора за питање које садржи те одговоре. Кандидати унутар простора за одговоре уписују своје одговоре оловком.
- Класе питања: Овај софтверски систем испољава могућност детекције и оцењивања питања типа кратак одговор на папирним тестовима.
- Технологије: Целокупан софтверски систем развијен је коришћењем програмског језика Пајтон, његових стандардних библиотека и екстерних библиотека отвореног кода.
- Области вештачке интелигенције: Овај софтверски систем користи технике оптичког препознавања знакова за екстракцију речи текста из питања типа кратак одговор. Технике обраде природног језика (енг. *NLP, Natural Language Processing*) користе се у препроцесирајућој фази за издвајање битних делова текста.
- Структура система: Процес дизајнирања теста, његовог умножавања у потребном броју копија и дистрибуирања институцијама које врше испитивање кандидата није у надлежности овог система. Развој овог софтверског система састоји се из неколико фаза. У фази скенирања на основу попуњеног папирног теста добија се његов дигитални облик коришћењем обичног скенера.

Такође, овај модул детектује и издваја одговоре кандидата на питања типа кратак одговор. Као продукт ове фазе настају текстуални фајлови који садрже текст написаног одговора. Креирани скуп података састоји се из идентификованих одговора и бодовања које је додељено одговорима ручним прегледањем одговора.

Одговори се задају као улаз у препроцесирајућу фазу која из текста одговора издваја само битне делове. Препроцесирајућа фаза састоји се од провере граматике, токенизације, елиминације стоп речи, провере антонима и синонима и стемизације. Затим долази фаза учења, која се састоји из тренирања и тестирања модела. У фази тренирања користе се претходно споменути лабелирани скуп података. Поред тога, користи се и кључ, који садржи тачне одговоре, да би се извршило тренирање модела.

Сваки унети одговор и одговарајући тачан одговор мапирају се у векторски простор на основу TF_IDF (енг. *Term frequency - inverse document frequency*) метрике и метрике косинусне сличности. Прва метрика треба да утврди колико је нека реч значајна. На другој метрици базира се одређивање семантичке сличности између тих одговора. У фази тестирања, за одговоре који нису ручно оцењени врши се рачунање метрике сличности. На основу ове метрике одредиће се бодовање за такве одговоре.

- Експерименти: Нису наведени подаци о обиму скупова података који су коришћени за тренирање или за тестирање модела. Такође, нису наведени подаци о прецизности модела.
- Евалуација:
 - Предности: Аутори овог софтверског система наводе високу прецизност у раду система, али нису наведени подаци о тачности ни у апсолутним бројкама нити у процентима.

- Недостаци: Нису наведени подаци о времену потребном за обраду једног папирног теста. Ипак, наглашено је да је потребно смањити тренутно време потребно систему за процесирање папирних тестова. Такође, нису наведени подаци о конфигурацији машине коришћене за тестирање система.

3.13. AGHNA

Софтверски систем AGHNA (енг. *Automated Grading of Handwritten Numerical Answers*) је направљен на Политехничком државном институту у Калифорнији, у Сан Луис Обиспу, Калифорнији (енг. *California Polytechnic State University, San Luis Obispo, CA*). Пројекат је реализован током 2017. године.

- Сврха: Мотивација за развој овог софтверског система била је аутоматизација процеса оцењивања нумеричких одговора на питања на којима се одговор уписује руком уз помоћ оловке. Идеја је била да се реализује систем који ће бити интуитиван за коришћење од стране просечног оцењивача, који може да аутоматски утврди локације одговора, екстрахује их из теста, класификује сваки од њих као тачан или нетачан и додели оцену и објашњење. Такође, систем треба да подржи формат теста који је кандидатима једноставан за коришћење.
- Изглед папирног теста: Папирни тест се састоји од низа нумерисаних питања. Текст питања се налази на посебним листовима, док се простор за упис одговора од стране кандидата налази на посебном обрасцу. Пример попуњеног папирног теста илуструје Слика 35.

Name: _____

Answer Sheet

- Write your name on this answer sheet in the space above.
- The answer to each question is a number. Write the number in the boxes provided, one digit per box. Decimal points should be in their own box. You may leave unused boxes blank.

1.	1	1	1	1	1	1
2.	2	2	2	2	2	2
3.	3	3	3	3	3	3
4.	4	4	4	4	4	4
5.	5	5	5	5	5	5
6.	6	6	6	6	6	6
7.	7	7	7	7	7	7
8.	8	8	8	8	8	8
9.	9	9	9	9	9	9
10.	0	0	0	0	0	0
11.
12.	-	-	-	-	-	-
13.	1	1	2	3	.	6
14.	2	4	6	8	.	7
15.	0	.	7	2	1	.
16.	1	9	2	5	.	3
17.	2	2	.	6	3	
18.	1	1	2	.	2	2
19.	7	6	2	.	0	0
20.	9	3	.	2	7	5
21.	0	.	6	7	5	
22.	9	9	.	9	9	9
23.	2	2	.	3	6	5
24.	1	1	.	3	7	2
25.	2	3	.	0	2	1
26.	-	7	2	.	3	
27.	-	3	2	.	7	6
28.	-	3	7	.	9	0
29.	-	2	3	.	1	6
30.	-	0	.	7	9	2

Слика 35. Пример попуњеног папирног теста

Простор за упис одговора задат је у виду низа квадрата, при чему један низ представља место за упис одговора једног питања. У квадрате за упис одговора кандидати треба да упишу тачно једну арапску цифру, децималну тачку, знак минус или да оставе квадрат празним. Ови квадрати за упис одговора омогућавају каснију лакшу детекцију простора за упис одговора, с обзиром да на формулару формирају правилне структуре. Поред тога, овакав распоред не нарушава једноставност коришћења теста кандидатима, а олакшава систему детекцију облика од интереса.

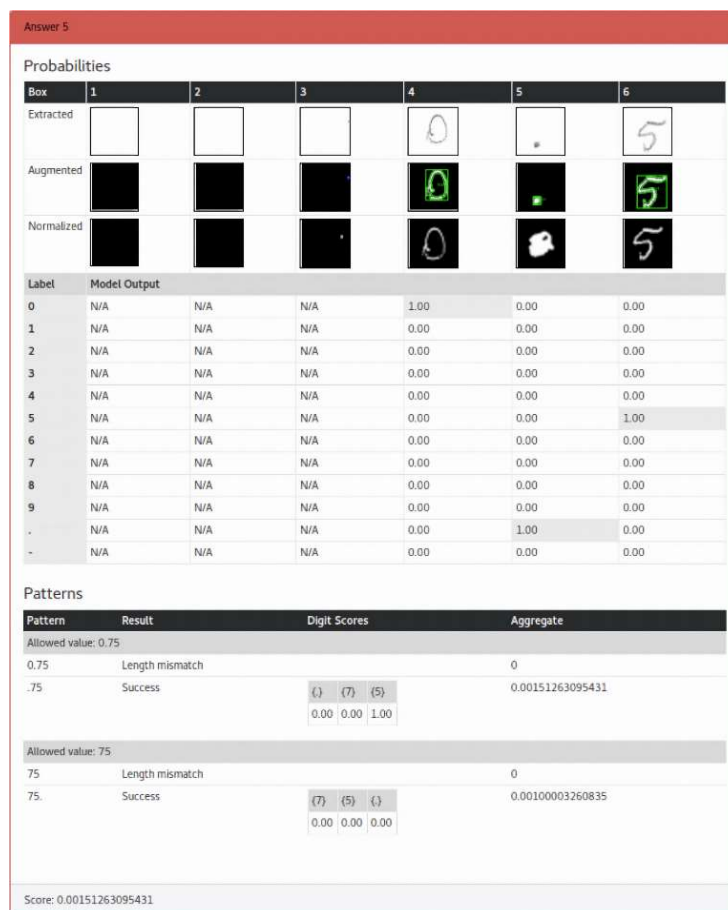
- Класе питања: Овај софтверски систем испољава могућност детекције и оцењивања питања типа кратак одговор на папирним тестовима.
- Технологије: Цео софтверски систем развијен је коришћењем програмског језика Пајтон, његових стандардних библиотека и екстерних библиотека отвореног кода.
- Области вештачке интелигенције: Овај софтверски систем користи алгоритме компјутерске визије (као што су бинаризација, дилатација, затварање, отварање итд.) за препроцесирање слике. Ово препроцесирање се ради у циљу боље детекције облика од интереса на слици. Принципи компјутерске визије као што су детекција контура и њихова апроксимација у правилне полигоне се користе за динамичко утврђивање локација сваког од квадрата за упис цифре одговора. За детекцију руком написаних цифара одговора користе се технике дубоког учења као што су конволуционе неуралне мреже. Модел конволуционе неуралне мреже је био заснован на архитектури дубоке MNIST конволуционе неуралне мреже за експерте (енг. *Deep MNIST for Experts*).
- Структура система: За тренирање модела за предикцију уписаног одговора од стране кандидата коришћен је MNIST скуп података, који садржи за сваку арапску цифру по отприлике 6 500 примерака. С обзиром да систем треба да испољава и могућност препознавања знакова тачке и минуса, ти подаци су додатно добављени. Прикупљено је укупно по 200 различитих примерака за оба ова знака.

Користећи разне технике аугментације података, као што су ротације, обртања, Гаусов блур, скалирање, транслације итд. овај скуп података доведен је на бројку примерака приближно доступну за сваку цифру у MNIST скупу података. Тренирање модела спроведено је у 50 епоха користећи Адамов оптимизатор, како би се минимизовала софтмакс унакрсна ентропија. Поред тога, модел се може додатно фино подесити прикупљањем примерака руком написаних цифара и знакова тачке и минуса од стране кандидата чији ће тестови бити оцењивани.

У ове сврхе довољно је прибавити и до 10 кандидата који ће обезбедити написане примерке ових симбола. При калибрацији модела, мењају се само тежине у финалном потпуно повезаном слоју мреже, док се остале тежине не мењају. Ово се врши да би модел сачувао раније прикупљено знање током оригиналног тренирања модела. Због малог броја додатних примерака, дотренирање модела се може обавити и у 200 епоха.

Целокупан процес испитивања састоји се из неколико делова. Најпре се, користећи обичан процесор текстуалног документа врши креирање питања и одговора теста. Као производ настаје тест у дигиталном облику који се умножава у потребном броју примерака користећи обичан штампач.

Потом, у процесу израде теста од стране кандидата настаје одређени број попуњених папирних тестова. Они се затим скенирају обичним скенер уређајем да би се добио дигитални облик сваког од попуњених папирних тестова. Слика 36 приказује идентификоване одговоре једног кандидата са једног питања теста.



Слика 36. Идентификовани одговори са једног питања ове класе

Користећи компоненту за детекцију региона од интереса која на попуњеном формулару претражује мрежу контура квадратног облика, врши се детекција простора за упис одговора кандидата. Они се затим екстрахују и дају као улаз моделу конволуционе неуралне мреже која је задужена за предикцију симбола уписаних у појединачне квадрате за упис одговора. С обзиром да се исте нумеричке вредности могу уписати на више различитих начина (нпр. 0.75 као .75 или као 0.750), детектоване вредности се проверавају са кључем да би се утврдило да ли је у простору за одговоре записана нека од ових комбинација.

Потом се врши класификација одговора у класу тачан или нетачан и врши се израчунавање целокупне оцене. Систем поседује могућност прављења извештаја о тесту кандидата са подацима о одговорима које је дао кандидат, препознатим одговорима од стране система заједно са мером сигурности у препознате одговоре и тачним и нетачним одговорима на питања. Пример извештаја теста једног кандидата приказује Слика 37.

Summary		
Correct: 11		
Incorrect: 4		
Grade: 11/15 = 73.33333333333333%		
Answer	Score	Correct
1	1.000	True
2	0.999	True
3	0.905	True
4	0.957	True
5	0.002	False
6	0.002	False
7	0.963	True
8	0.857	True
9	1.001	True
10	0.854	True
11	1.001	True
12	1.001	True
13	0.606	True
14	0.000	False
15	0.001	False
16	Not graded	N/A
17	Not graded	N/A
18	Not graded	N/A
19	Not graded	N/A
20	Not graded	N/A
21	Not graded	N/A
22	Not graded	N/A
23	Not graded	N/A
24	Not graded	N/A
25	Not graded	N/A
26	Not graded	N/A

Слика 37. Извештај са теста једног кандидата

- Експерименти: Процес тренирања модела за предикцију руком написаног нумеричког одговора обављен је користећи *GTK 980 Ti* графичку картицу. Цео процес тренирања је трајао око 5 минута. Систем је постигао прецизност у предикцији руком написаног нумеричког одговора од 99.01 % над тестним подацима.

Додатно, систем је тестиран и над подацима добијеним са теста који је садржао 15 питања, а који је полагало 35 кандидата. Систем је додатно калибрисан подацима прикупљеним од кандидата пре процеса тестирања, а који су садржали од стране кандидата написане класе симбола које систем треба да препозна. Сваки одговор који је систем препознао као тачан је и био тачан. Ипак, систем је успео да успешно препозна 95.6 % тачних одговора кандидата као тачне. Стога, ручни прегледач би требало да игра улогу супервизора у случају одговора које је систем препознао као нетачне.

- Евалуација:
 - Предности: Изабрани софтверски систем показује високу прецизност у детекцији руком писаног нумеричког одговора, која износи 100 %, док одзив износи 95.6 %. Ово даје укупну тачност система од 96.7 %. Овај софтверски систем је пројекат отвореног кода (енг. *open source*) и бесплатно је доступан.
 - Недостаци: Систем не прави увек јасну разлику између знака тачке и цифре 0, као и знака минус и цифре 7. Поред тога, за тестирање је коришћен релативно мали скуп података од 35 попуњених тестова кандидата. Такође, нису наведени подаци о времену потребном за процесирање једног попуњеног папирног теста. Иако систем показује висок одзив од 95.6 %, ипак је у случају одговора које је систем препознао као нетачне потребна улога прегледача као супервизора.

3.14. AGSLCQ

Софтверски систем AGSLCQ (енг. *Automated Grading of Short Literal Comprehension Questions*) је направљен од стране Факултета науке и технологије Отвореног универзитета у Хонг Конгу, у Хо Ман Тину, Ковлону, Хонг Конгу, Кини (енг. *School of Science and Technology, The Open University of Hong Kong, Ho Man Tin, Kowloon, Hong Kong SAR, China*). Пројекат је реализован током 2015. године.

- Сврха: Мотивација за развој овог софтверског система била је повећање ефикасности и објективности у процесу оцењивања написаних одговора кандидата на папирним тестовима. Циљ овог система је да оцењује литерарна питања, која имају особину да су њихови одговори представљени једном речју или фразом од неколико речи. Такође, кључне информације за одговарање на оваква питања се обично налазе у тексту питања и акценат приликом испитивања се ставља на разумевање текста питања.
- Изглед папирног теста: Не постоји стриктно одређен формат папирног теста који овај систем користи. Ипак, постоји неколико правила која треба да се поштују приликом дизајнирања тестова. Простори за питања треба да буду раздвојени једни од других, односно да се не преклапају.

Простор за одговоре се налази у оквиру простора за питање које садржи те одговоре. Кандидати унутар простора за одговоре уписују своје одговоре оловком. Такође, питања могу да се постављају користећи само три упитне речи, а то су речи: шта, ко и зашто (нпр. „Шта има већу РН вредност, киселине или базе?“ или „Ко је председник Америке?“). За потребе тестирања овог система коришћени су папирни тестови са 16 питања типа кратки одговор.

- Класе питања: Овај софтверски систем испољава могућност детекције и оцењивања питања типа кратак одговор на папирним тестовима.
- Технологије: Целокупан софтверски систем развијен је коришћењем програмског језика Јава, његових стандардних библиотека и екстерних библиотека отвореног кода. Такође, коришћена је и Стенфордова CoreNLP библиотека.
- Области вештачке интелигенције: Овај софтверски систем користи технике оптичког препознавања знакова за екстракцију речи текста из питања типа кратак одговор. Технике обраде природног језика користе се у препроцесирајућој фази за издвајање битних делова текста.
- Структура система: Циљ изабраног софтверског система је да пронађе меру сличности између одговора кандидата и тачног одговора. Процес мерења ове сличности заснива се на три принципа. Први је да одговор кандидата и тачан одговор морају произићи из текста питања, па ће коришћењем речи и њихових н-грама поређење бити ефикасно. Други принцип је да сваки одговор треба да садржи кључне речи, тако да у случају фразе од неколико речи неке речи носе више значења од других. Трећи принцип је да упитна реч текста питања треба да сигнализира кандидатима информације које је неопходно да препознају из текста питања.

Процес дизајнирања теста, његовог умножавања у потребном броју копија и дистрибуирања институцијама које врше испитивање кандидата није у надлежности овог система. Након што кандидати у процесу израде теста производе попуњене папирне тестове, врши се њихова дигитализација коришћењем обичног скенера. Потом се компонентом система задуженом за екстракцију одговора извлаче речи из одговора кандидата.

Затим се врши препроцесирање одговора, које укључује и лемизацију, тако да се избаце небитне речи. За сваку реч се одређују POS тагови (енг. *part-of-speech*) и одређују се сличности за битне речи, при чему врста есенцијалне речи зависи од упитне речи текста питања. У ове сврхе може се користити неколико шема за поређење, као што је нпр. Џакардов коефицијент (енг. *Jaccard coefficient*).

- Експерименти: Ручно написани одговори кандидата на тип питања кратак одговор прикупљени су из локалне средње школе. Скуп података се састоји од 887 прикупљених одговора кандидата на питања типа кратак одговор. Ови одговори су ручно оцењени од стране 5 учитеља те средње школе.

С обзиром да питања на тесту носе различити број поена, за потребе рада овог система извршена је нормализација бодовања одговора кандидата датог од стране учитеља тако да одговара скали са 3 опсега вредности: $[0 - 0.25]$ (нетачан одговор), $[0.25 - 0.75]$ (парцијално тачан одговор) и $(0.75, 1]$ (тачан одговор). Од 887 прикупљених одговора, 118 је класификовано као парцијално тачан одговор, док је осталих 769 класификовано или као тачан или као нетачан одговор. Систем је укупно 810 одговора класификовао на исти начин као и ручни прегледачи, што представља тачност од 91.3 %.

Разлике у оцењивању јављају се код питања која почињу упитном речју „шта“, јер су кандидати нешто лошије урадили тест, па су прегледачи субјективније и блаже оценили незнање него систем. Код питања која почињу упитном речју „ко“ су прегледачи одбацивали одговоре са додатним речима поред тражене, што систем није радио. Код питања која почињу упитном речју „када“ и систем и ручни прегледачи су били усаглашени.

Оцењивање 100 одговора на обичном лаптоп рачунару, чија конфигурација није наведена, захтевало је нешто мање од 30 секунди. За неке упитне речи нису потребни POS тагови речи одговора, те се оцењивање истог броја одговора може обавити у времену нешто краћем од 10 секунди. Треба узети у обзир да ово мерено време не обухвата режијске трошкове обраде дигиталног облика папирног теста, већ само оцењивање екстрахованог одговора.

- Евалуација:
 - Предности: Систем је испољио високу тачност у свом раду у оцењивању одговора на питања типа кратак одговор која износи 91.3 %. Поред тога, систем је показао завидну брзину у оцењивању одговора која износи између 100 и 300 милисекунди по одговору, при чему време неопходно за оцењивање зависи од типа упитне речи питања (неопходности POS тагова).
 - Недостаци: Систем је тестиран на нешто мањем броју података, што се правда немогућношћу лаког проналаска скупа података која садрже питања овог типа, а да та питања почињу са неком од три наведене упитне речи: шта, ко, када. Такође, предложена класификација је ад хок урађена и сами аутори наводе да би она могла да се замени коришћењем техника машинског учења.

4. Имплементациони детаљи предложеног система

Предложени систем за испитивање кандидата коришћењем папирних тестова састоји се од неколико подсистема. То су подсистем за дизајнирање дигиталног теста, подсистем за зонирање дигиталног теста, подсистем за умножавање дигиталног теста, подсистем за дигитализацију папирног теста, подсистем за аутоматизовано процесирање папирног теста и подсистем за складиштење и манипулацију резултата. Фокус ове докторске дисертације биће на подсистему за аутоматизовано процесирање папирног теста [59], док ће остали подсистеми бити описани у мањој или већој мери у наставку. Такође, у наставку ће бити приказан и дијаграм који приказује како су наведени подсистеми повезани у целокупан систем за испитивање кандидата, као и илустровани разни артефакти настали у самом процесу рада система.

4.1. Систем за тестирање кандидата

Процес испитивања кандидата коришћењем папирних тестова започиње подсистемом за дизајнирање дигиталног теста. Овим системом врши се прављење структуре теста у погледу питања која се на њему налазе, као и њихових одговора. Прва страница теста је и насловна страница и њен део илуструје Слика 38.



Слика 38. Део насловне странице теста

На насловној страници теста (у левом делу њеног заглавља) налази се правоугаоно оивичен простор у којем ће се налазити идентификациона налепница кандидата. Идентификациона налепница представљена је бар-кодом и носи информацију од 18 цифара јединствене идентификације. Ова налепница лепи се од стране дежурног у току трајања испитне активности.

Поред тога, на насловној страници теста (у средини њеног заглавља), а и на свакој наредној страници теста, налази се квадратно оивичен простор у којем се налази идентификатор странице. Идентификатор странице представљен је дводимензионалним бар-кодом и носи информацију о варијанти теста и броју странице у оквиру теста. Ова информација се не лепи налепницом у току трајања теста, већ се у фази дизајнирања теста дводимензионални бар-код умеће на сваку страницу као слика.

Свака страница теста садржи и граничнике корисног садржаја странице. Они се налазе у сва четири угла странице теста. Граничници су облика ћириличног слова Г и ротирани су

тако да се место спајања вертикалне и хоризонталне линије граничника у једну тачку налази најближе углу странице у коме се граничник налази.

Питања у оквиру странице налазе се уоквирена правоугаоницима, при чему се правоугаоници питања налазе у простору оивиченом граничницима странице. Простор за унос одговора на питања се налази у оквиру простора питања. У случају питања са понуђеним одговорима, она се налазе у оквиру правилних табела. Питања могу да се простиру на више страница, при чему се на свакој страници налази засебни правоугаоник за тај део питања. Пример странице теста са питањима показује Слика 39.

The image shows a test page with a header area containing a square logo. Below the header, there are four question blocks, each with a title, ID, points, and a description. The questions are about Old Church Slavonic, a sentence completion, a biography, and a story. Each question has a table of possible answers with radio buttons for selection. The page is framed by a border with the text 'У СВИКИМ ПОЉИМА НЕ ПИСАТИ' (Do not write in the margins) repeated vertically on both sides.

1. задатак 101112010201 број бодова: 1
Прочитајте следећи текст о почецима словенске писмености, па одговорите на питање. Крајем 9. века Ћирило и Методије створили су први књижевни језик свих Словена, који се данас назива старословенски језик. Сматра се да је Ћирило саставио и прво словенско писмо. Како се то писмо зове?
Обојте кружић испред тачног

<input type="radio"/>	глагољица
<input type="radio"/>	ћирилица
<input type="radio"/>	латиница

2. задатак 101121010801 број бодова: 1
У следећој реченици подвуците све прилоге.
Нисам био уморан, па сам отишао горе у своју собу, обукао канут и брзо изашао.

3. задатак 101121010801 број бодова: 1
Прочитајте основне биографске податке о писцу, па напишите његово име на линији испод текста.
Рођен је у Шапцу. Живео је у другој половини XIX века. Творац је психолошког реализма у српској књижевности. По занимању је био лекар, психијатар. Иако је написао само девет приповедака, важи за једног од наших најбољих приповедача. Познат је по приповеткама „Сне ће то народ позлатити“ и „Први пут с оцем на јутрење“. Умро је у Београду.
Име и презиме писца: _____

4. задатак 101127010301 број бодова: 2
Тема „Горског вијенца“ Петра П Петровића Његоша је истрага нотурица. Владика Данило се колеба када треба да одлучи да ли се обрачуна са нотурицама. Обојте кружић испред тачног одговора.

<input type="radio"/>	Плаши се славних турских ратника.
<input type="radio"/>	Свестан је да ће жртве бити огромне.
<input type="radio"/>	Не жели да ратује против свог народа.
<input type="radio"/>	Сумња у снагу и храброст своје војске.
<input type="radio"/>	Плани се да ће се племена међусобно истребити.

Слика 39. Пример странице теста

Након што се заврши процес дизајнирања дигиталног теста, он се штампа у потребном броју папирних примерака користећи подсистем за умножавање дигиталног теста. Штампа се врши на папирима величине А3, тако да се на свакој страници овог папира налазе по две странице теста. Затим се одштампани А3 папири скупљају у тест, који се савија по средини страница формирајући свеску са страницама величине А4. Тако добијени тестови се пакују и шаљу установама које су овлашћене за спровођење тестирања кандидата.

Подсистем за зонирање дигиталног теста ради у паралели са подсистемом за умножавање дигиталног теста. Овај подсистем је реализован као десктоп апликација, чији улаз представља раније дизајнирани дигитални тест. Фазом зонирања дигиталног теста руководи зонер оператер, који је одговоран за означавање и идентификацију горњег левог граничника, који представља референцијалну тачку за одређивање зона свих осталих региона од интереса.

Зонер оператер је дужан да обави и означавање региона за идентификацију кандидата на насловној страни. Поред тога, одговоран је и за означавање региона за идентификацију теста, односно региона са дводимензионалним бар-кодом. Такође, зонер оператер је дужан да

означи регионе свих питања на тесту, као и регионе одговора на та питања. Означавање зона региона од интереса зонаер оператер обавља превлачењем правоугаоника преко региона уз помоћ миша, држећи његов леви тастер притиснутим у процесу превлачења.

Зонирање се обавља на потпуно празном тесту, коме недостају текст питања и одговора, у циљу заштите од цурења информација. Овај процес као свој излаз ствара конфигурациону датотеку за тест, која чува информације о тесту као што су: идентификација теста, број страница, број зона питања и одговора и њихови оквирни положаји итд. Ова датотека представља улаз у подсистем за процесирање папирних тестова, при чему се ове информације у том подсистему користе строго у валидационе сврхе и не транслирају се директно на сваки појединачни тест.

Након тога следи процес тестирања кандидата. Кандидати добијају празан папирни тест који попуњавају уписивањем жељених одговора. Дежурни лепе кандидатима идентификационе налепнице у за то предвиђено место на насловној страници. Излаз ове фазе представљају попуњени папирни тестови од стране кандидата.

Након што се процес тестирања кандидата заврши, попуњени папирни тестови транспортују се у неколико центара у којима се налази подсистем за дигитализацију папирних тестова. Попуњени папирни тестови се исецају специјалним секачима по сивим вертикалним правоугаоним просторима који се налазе дуж обе дуже стране сваке А4 странице теста. Затим се свака страница теста убацује у скенер, који је повезан са подсистемом за дигитализацију папирних тестова.

Подсистем за дигитализацију папирних тестова је реализован као десктоп апликација. Као свој улаз, подсистем прима појединачне дигиталне странице теста од скенера. Он врши склапање страница теста у исправном поретку и од скенера добијене слике страница једног теста склапа у једну вишестраничну датотеку.

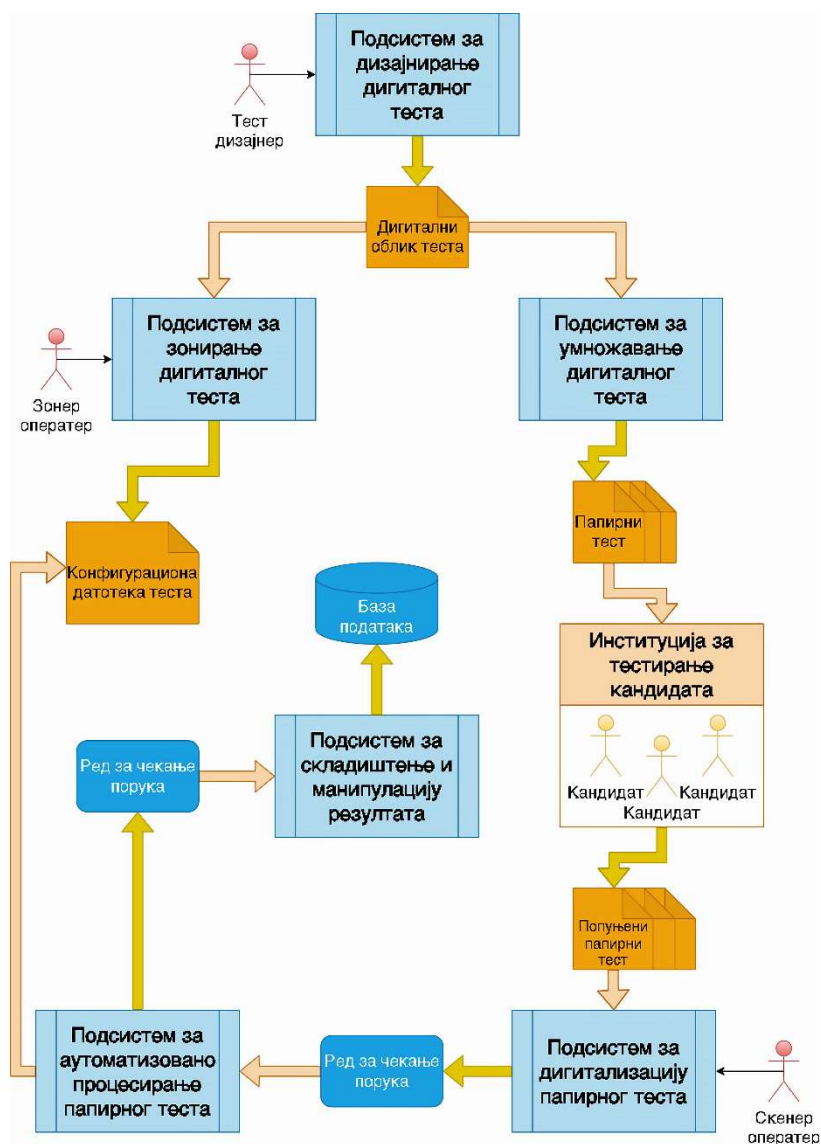
У процесу дигитализације папирних тестова, сваки тест може да се сврста у једну од три категорије: комплетан, некомплетан (недостаје страница) или тест са грешком. Подсистем шаље комплетне и некомплетне тестове у одговарајуће репозиторијуме на серверу који их чува. Иако се и некомплетни тестови шаљу на даље процесирање, биће им потребна додатна ручна интервенција која ће се побринути за недостајуће стране.

Тестови са грешком захтевају ручну интервенцију и тако се означавају у случају било које друге грешке која се може јавити у току обраде теста у овој фази. Подсистем за дигитализацију папирних тестова шаље искључиво комплетне и некомплетне тестове на процесирање користећи концепт реда за чекање порука (енг. *message queue*). Ред за чекање представља спону између овог подсистема и подсистема за процесирање папирних тестова.

Поред тога, овај подсистем периодично шаље и информације о броју послатих комплетних и некомплетних тестова, као и о броју тестова са грешкама које су се десиле. У случају да се неки тест више пута принесе као улаз овог подсистема, за њега ће бити послате вишеструке поруке. Ово има смисла радити, с обзиром да тесту могу недостајати неке странице теста које касније могу бити пронађене.

Подсистем за процесирање папирних тестова чита поруке убачене од стране подсистема за дигитализацију папирних тестова из реда за чекање порука. Затим се врши идентификација кандидата, теста, питања и означених одговора. Детаљи о овом подсистему, који представља аутоматизовано оцењивање папирних тестова, биће наведени у наредним потпоглављима овог поглавља. Сви идентификовани подаци шаљу се као једна порука у ред за чекање који представља спону између овог подсистема и подсистема за складиштење и манипулацију резултата.

На самом крају, подсистем за складиштење и манипулацију резултата чита поруке из реда за чекање убачене од стране подсистема за процесирање папирних тестова. Податке о тестовима, њиховим тачним одговорима и вредностима одговора у броју бодова овај подсистем складишти у релационој бази података. Овај подсистем упоређује добијене одговоре од подсистема за процесирање папирних тестова, укршта их са тачним одговорима складиштеним у бази података, а затим у релационој бази података задуженој за складиштење резултата кандидата смешта одговоре, појединачан и укупан број бодова кандидата. Целокупан дијаграм тока контроле у систему за испитивање кандидата коришћењем папирних тестова илуструје Слика 40.



Слика 40. Дијаграм тока контроле система

4.2. Подсистем за процесирање папирног теста

Целокупан подсистем за процесирање папирног теста реализован је у програмском језику Пајтон као сервис подигнут на једној или више предвиђених машина. Приликом подизања сервиса најпре се читају подаци из конфигурационе датотеке. Ова датотека садржи информације о пријемном и одлазном реду за чекање порука, креденцијале за успостављање конекције са брокером порука (енг. *message broker*) за ове редове за чекање порука, максимално време обраде поруке, информације и креденцијале за приступ и пренос удаљених података

(SMTP, енг. *Simple Message Transfer Protocol*), квалитет скенираног теста (*dpi*, енг. *dots per inch*), број процеса радника (енг. *number of worker processes*) итд.

Након што се прочита конфигурациона датотека, ствара се директоријум за смештање датотека са текстуалним информационим порукама и порукама о грешкама (енг. *log files*). Поред тога, ствара се и директоријум за смештање међурезултата насталих у процесу обраде тестова. При томе, оба ова директоријума ће служити за све тестове који се процесирају. Потом се прави одређен број процеса радника дефинисан конфигурационом датотеком, који се након тога покрећу. Псеудо код 1 илуструје ову обраду.

```
flag = True
if __name__ == '__main__':
    params = {}
    with open('config.json', 'r') as json_file:
        params = json.load(json_file)
        print(json.dumps(params, indent=2))
    recreate_dir('logs')
    recreate_dir('results')
    processes = [Process(target=worker_job) for _ in range(params['NumOfWorkers'])]
    for p in processes:
        p.start()
    for p in processes:
        p.join()
```

Псеудо код 1. Покретање процеса радника

Задатак процеса радника је да најпре успоставе конекцију са брокером порука. То се ради тако што се успоставља ослушкивач конекције, а затим се врши пријављивање на дестинацију. Дестинација у овом случају представља ред за чекање у који ће бити смештене поруке са информацијама о тестовима који долазе од стране подсистема за дигитализацију папирних тестова. Пријављивање на дестинацију врши се и, уколико у неком тренутку дође до прекида конекције са брокером порука. Псеудо код 2 илуструје овај процес.

```
def connect_and_subscribe(connection):
    global params
    try:
        connection.connect(params['StompUsername'], params['StompPassword'], wait=True)
        connection.subscribe(destination=fr'/queue/{params["StompDown"]}',
                              id=os.getpid(), ack='client-individual',
                              headers={'activemq.prefetchSize': 1})
    except Exception as e:
        raise e

def attempt_connection(connection):
    global flag
    flag = True
    while flag:
        try:
            connect_and_subscribe(connection)
            flag = False
        except (Exception,):
            minutes = params['StompRetryMinutes']
            print(f'MQ Server unavailable. Retrying again in {minutes} min...')
            time.sleep(minutes * 60)

def worker_job():
    conn = stomp.Connection(host_and_ports=[(params['StompHost'], params['StompPort'])],
                           heartbeats=(params['StompHeartbeats'], params['StompHeartbeats']))
    conn.set_listener('', Listener(conn))
    attempt_connection(conn)
    global flag
    flag = True
    while flag:
        try:
            time.sleep(params['SleepTime'])
        except (Exception,):
            attempt_connection(conn)
```

Псеудо код 2. Посао процеса радника

Ослушкивач конекције врши неколико корака када се прими порука доспела у ред за чекање порука коју је послао подсистем за дигитализацију папирних тестова. Најпре се из поруке извлачи информација о путањи до датотеке која представља тест и која се налази у директоријуму послатих тестова на серверу, као и информација о путањи до конфигурационе датотеке за тај тест. Затим се успоставља сигурносна конекција са сервером на коме се налази директоријум са до сада послатим тестовима и конфигурацијама тестова користећи *SSH* (енг. *Secure Socket Shell*) протокол. Потом се врши довлачење теста и његове конфигурације копирањем са удаљеног сервера.

Након тога процес радник приступа обради једног теста у дигиталном облику. За обраду су му потребни: конфигурациона датотека теста, датотека теста, као и путање до директоријума са међурезултатима и директоријума са датотекама са информационим порукама и порукама о грешкама. Након завршеног процесирања теста, на сервер се шаљу резултујуће датотеке настале у процесу обраде и информација о статусу обраде. Након што се слање заврши, бришу се сви међурезултати настали на локалној машини током ове обраде, који свакако више нису неопходни. Псеудо код 3 илуструје ову обраду.

Процес радник на почетку за тест који процесира ствара директоријум за смештање међурезултата и директоријум са датотекама са информационим порукама и порукама о грешкама. Затим се из локалног директоријума врши читавање теста представљеног вишестраничном датотеком (енг. *multipage .tiff file*). Након тога се пролази кроз странице датотеке теста са циљем да се свака појединачна страница теста обради у складу са предвиђеном секвенцом корака. Псеудо код 4 илуструје ове кораке.

```

class Listener(stomp.ConnectionListener):
    def __init__(self, connection):
        self.conn = connection

    def on_error(self, frame):
        print(f'Received error: {frame.body}')

    def on_message(self, frame):
        print(f'Received message: {frame.body}')
        message_id = frame.headers['message-id']
        results_folder = json_path = test_name = test_path = ''
        global params
        try:
            ssh = SSHClient()
            ssh.set_missing_host_key_policy(AutoAddPolicy())
            ssh.connect(hostname=params['SftpHostname'], port=params['SftpPort'],
                        username=params['SftpUsername'], password=params['SftpPassword'])
            scp = SCPClient(ssh.get_transport())
            msg_json = json.loads(frame.body)
            if len(sys.argv) > 1 and sys.argv[1] == '1':
                raise Exception()
            test_name = msg_json["Path"][msg_json["Path"].rfind("/") + 1:]
            test_path = msg_json["Path"][:msg_json["Path"].rfind("/") + 1]
            json_path = test_name.split('.')[0] + '.json'
            scp.get(msg_json['Path'])
            scp.get(msg_json['Configuration'], json_path)
            scp.close()
            print(f'Path: {json_path} | Test: {test_name}')
            with open(json_path, 'r') as json_file:
                test_json = json.load(json_file)
            results_folder = f'results{os.sep}{test_name[:test_name.rfind(".")]}'
            args = [test_json, test_name, results_folder,
                    {'name': f'{test_name}',
                     'file': f'logs{os.sep}LOG_{test_name[:test_name.rfind(".")]}.txt'}]
            ret_code = process_test_pages(*args)
            if ret_code not in [4096]:
                scp = SCPClient(ssh.get_transport())
                scp.put(f'{test_name[:test_name.rfind(".")]}.json',
                       f'{test_path}{test_name[:test_name.rfind(".")]}.json')
                for root, dirs, files in os.walk(results_folder):
                    for image in files:
                        scp.put(f'{results_folder}{os.sep}{image}', f'{test_path}{image}')
                        break
                scp.close()
                msg = json.dumps({'path': test_path, 'status': ret_code})
                self.conn.send(body=msg, destination=fr'/queue/{params["StompUp"]}')
            else:
                msg = {'Path': test_path, 'Status': ret_code, 'Error': 'Problem sa testom.'}
                self.conn.send(body=msg, destination=fr'/queue/{params["StompUp"]}')
        except Exception as e:
            ret_code = 8192
            print(f'Exception:{e}')
            msg = {'Path': test_path, 'Status': ret_code, 'Error': 'Problem u komunikaciji.'}
            self.conn.send(body=msg, destination=fr'/queue/{params["StompUp"]}')
        finally:
            global flag
            flag = True
            if len(results_folder) and os.path.isdir(results_folder):
                shutil.rmtree(results_folder)
            if len(test_name) and os.path.isfile(test_name):
                os.remove(test_name)
            if len(json_path) and os.path.isfile(json_path):
                os.remove(json_path)
            self.conn.ack(message_id, os.getpid())
            print('Message send')

    def on_connected(self, frame):
        print('Connected')

    def on_disconnected(self):
        print('Disconnected')
        attempt_connection(self.conn)

```

Псеудо код 3. Обрада поруке из реда за чекање

```

def process_test_pages(test_json, test_file, results_folder, logger):
    test_logger = setup_logger(logger['name'], logger['file'])
    test_start_time = time.time()
    ret_code = 0
    try:
        recreate_dir(results_folder)
        test_tif = load_test_tiff(test_file)
        for page_iter, image in enumerate(test_tif):
            try:
                if not check_page_dim(test_json, image):
                    raise Exception('Bad page dimension!')
                page_key = f'_P_{page_iter + 1}'
                if not page_iter:
                    # FETCHING BARCODE
                    test_json['Bar-code'] = get_bar_code(image, test_json, test_logger)
                    if not test_json['Bar-code']:
                        image = rotate_image(image, 180)
                        test_json['Bar-code'] = get_bar_code(image, test_json, test_logger)
                    if not test_json['Bar-code']:
                        raise Exception('Bar-code not found!')
                    continue
                # FETCHING QR-CODE
                qr, angle = get_qr_code(image, test_json, page_iter + 1, test_logger)
                if not qr:
                    image = rotate_image(image, 180)
                qr, angle = get_qr_code(image, test_json, page_iter + 1, test_logger)
                if not qr:
                    raise Exception('QR not found!')
                test_json['QR_list'].append(qr)
                image = rotate_image(image, angle)
                # FIND QUESTIONS FOR CURRENT PAGE
                questions = get_questions(test_json, page_iter + 1)
                # FIND IMAGE LINES, BORDERS AND POTENTIAL QUESTIONS
                lines, borders, pot_questions = get_zones(test_json, page_image, test_logger)
                # ADJUST QUESTIONS AND ANSWERS ZONES COORDINATES TO TOP LEFT BORDER
                adjust_questions(test_json, borders, questions, pot_questions,
                                page_iter + 1, test_logger)
                # DETECT QUESTIONS' ANSWERS
                for q in questions:
                    for a in get_a_zones(q, page_iter + 1, test_logger):
                        if q.type() is Q_TYPES.MC:
                            ans = get_circles(lines, image, a, test_json, test_logger)
                        elif q.type() is Q_TYPES.M:
                            if a['Zoned'] is None:
                                ans = get_under(test_json, image, a, test_logger)
                            else:
                                ans = get_matches(test_json, image_lines, image, a, test_logger)
                        elif q.type() is Q_TYPES.SA:
                            ans = get_short_answers(image, a, test_logger)
                            ret_code |= update_result(test_json, a, ans, test_logger)
                            save_answer_image(page, a, test_logger)
                            ret_code |= update_result(test_json, q, test_logger)
                            save_question_image(page, q, test_logger)
                        test_logger.log_info(f'Tiff page {page_iter + 1} DONE.')
            except Exception as e:
                save_page_image(page, test_logger)
                test_logger.log_error(f'Page {page_key}: {e}')
                ret_code |= get_error_code(e)
        clean_json_data(test_json)
        with open(f'{test_json["json_file"]}.json', 'w') as file:
            json.dump(test_json, file, ensure_ascii=False)
        test_end_time = time.time()
        test_logger.info(f'Time: {(test_end_time - test_start_time) * 1000} ms')
    except Exception as e:
        test_logger.log_error(f'Test {test_file}: {e}')
        ret_code |= get_error_code(e)
    return ret_code

```

Псеудо код 4. Кораћи обраде једног теста

Најпре се врши провера димензије слике странице да би се утврдило да страница није физички оштећена. Ово се односи на недостајање неког дела странице и таква ситуација се сигнализира грешком. Након тога се врши детекција бар-кодова, при чему се детекција идентификационог једнодимензионалног бар-кода извршава условно само на насловној

страница, а на свим страницама врши се детекција дводимензионалног страничног бар-кода. Потом се слика ротира, уколико је то потребно, за угао њеног нагиба према хоризонталној оси.

Затим се на основу конфигурационе датотеке утврђују оквирне локације региона од интереса за сва питања на тој страници. Потом се на страници врши претрага за граничницима те странице, а у тој обради се добијају и региони потенцијалних питања на тој страници, као и измењена слика странице теста на којој су преостале само хоризонталне и вертикалне линије. Затим се потенцијални региони питања потврђују и усклађују се информације о регионима питања добијене из конфигурационе датотеке теста са информацијама о добијеним регионима потенцијалних питања. Усклађивање пронађених локација региона питања са информацијама из конфигурационе датотеке теста приказује Псеудо код 5.

```
def adjust_questions(config, border, questions, potential_questions, page_num, logger):
    try:
        factor = config['TestDPI']
        for q in questions:
            for zone in q['Zones']:
                if zone['Page'] == page_num:
                    zone_rect = [int(zone['X'] * factor), int(zone['Y'] * factor),
                                int(zone['Width'] * factor), int(zone['Height'] * factor)]
                    zone_rect = [border[0] + zone_rect[0], border[1] + zone_rect[1],
                                zone_rect[2], zone_rect[3]]
                    for pq in potential_questions:
                        if is_overlapped(zone_rect, get_rect(pq), config['overlap_factor']):
                            zone_rect = get_rect(pq)
                    zone['X'], zone['Y'], zone['Width'], zone['Height'] = zone_rect
            for answer in q['Answers']:
                if 'Zone' in answer and answer['Zone'] == page_num:
                    answer_rect = [int(answer['X'] * factor), int(answer['Y'] * factor),
                                   int(answer['Width'] * factor), int(answer['Height'] * factor)]
                    answer_rect = [border[0] + answer_rect[0], border[1] + answer_rect[1],
                                   answer_rect[2], answer_rect[3]]
                    answer['X'], answer['Y'], answer['Width'], answer['Height'] = answer_rect
    except Exception as e:
        logger.log_error(f'Adjust ERROR : {e}')
```

Псеудо код 5. Усклађивање региона питања са информацијама из конфигурационе датотеке

За сва питања на тренутној страници врши се дохватање њихових зона одговора са слике тренутне странице, с обзиром да се питања могу простирати на више страница. За сваку зону одговора се, у зависности од врсте питања коме та зона одговора припада, изводи адекватна обрада која врши и ажурирање конфигурационе датотеке теста. Приликом обраде сваког питања и одговора прави се по једна слика издвајањем зоне питања, односно одговора, са слике странице.

У случају било какве грешке у обради тренутне слике странице теста, чува се цела слика тренутне странице и логују се информације о грешци. Поред тога, ажурира се код грешке, који ова обрада враћа као резултат. У случају било какве опште грешке у раду са тестом, сигнализира се грешка и логују се информације о грешци.

На самом крају се конфигурациона датотека теста пречишћава од ирелевантних информација за резултат процесирања. Резултат обраде представља уписивање измењене конфигурационе датотеке теста у локални директоријум. Овако измењена конфигурациона датотека теста са резултатима процесирања теста биће послата на сервер.

Приликом процеса препознавања врши се претрага на облике од интереса који су обично регуларни облици правилне структуре. Међутим, неколико радњи се одвија пре него што тест буде укључен у подсистем за процесирање тестова:

- Тестови се штампају у штампарским центрима, а изобличење теста може бити узроковано лошом штампарском машином.

- Тестове ручно оловком попуњавају кандидати, додатно доприносећи изобличењу на страницама теста.
- Тестови се физички преносе од локација тестирања до локација на којима се налазе подсистеми за дигитализацију теста и подложни су оштећењу у процесу транспорта.
- Тестови се дигитализују користећи подсистем за дигитализацију папирних тестова, а добијене слике не личе увек у потпуности на оригиналне странице теста због карактеристика скенера који се користе у процесу дигитализације папирних тестова.

Све ове радње доприносе одређеном нивоу нечистоћа и поремећаја на сликама, што не треба занемарити. Руковање сметњама у одређеној мери доприноси побољшању квалитета процеса препознавања. Међутим, с обзиром на број тестова и страница које их чине, додатно пречишћавање слике од шума продужава укупно време обраде. Стога, морфолошке операције које ће се примењивати над сликама страница теста морају бити пажљиво одабране да би се постигао прихватљив компромис између квалитета препознавања и неопходног времена обраде.

4.2.1. Препознавање једнодимензионалног бар-кода

Први корак представља утврђивање идентификационог једнодимензионалног бар-кода кандидата на насловној страници теста. Ипак, може се дискутовати о томе да ли је смисленије да ово буде иницијални корак или детекција дводимензионалног бар-кода. Чак се може размишљати о томе да ли би требало да детекција граничника региона од интереса буде иницијални корак, с обзиром да се и једнодимензионални бар-код и дводимензионални бар-код налазе унутар простора оивиченог граничницима региона од интереса.

Међутим, у процесу добијања дигиталног облика папирног теста може доћи до ненамерне ротације папирне странице теста. Ово значајно може утицати на детекцију граничника региона од интереса. Разлог за ово лежи у природи морфолошких операција које се примењују над сликом странице теста ради детекције граничника, а које ће бити наведене у потпоглављу о детекцији граничника.

Ипак, с обзиром да се идентификациони једнодимензионални бар-код валидно налази само на насловној страници, његова детекција ће бити иницијални корак. Поред тога, у пракси се показало да се бар-код лепи са знатно већом ротацијом, која не одговара ротацији насталој у процесу дигитализације папирног теста. Стога, никакво ротирање слике за угао ротације слике странице у односу на хоризонталну осу неће допринети доброј ротацији слике идентификационог бар-кода.

Због начина рада подсистема за дигитализацију папирног теста, чији скенер папир оријентише у портрет режиму, ненамерна ротација папирне странице теста не може бити блиска 90 степени у смеру казаљке на сату, односно у супротном смеру казаљке на сату. Она може износити или X степени или $180 \pm X$ степени у једном од два претходно наведена смера, где X представља вредност од неколико степени. Стога је претрага идентификационог једнодимензионалног бар-кода иницијално ограничена на врх слике, а уколико та претрага не успе, нова претрага се извршава на дну слике странице теста.

Врх односно дно слике представљају региони слике странице теста који су исте ширине као и ширина слике странице теста, а чија висина износи 20 % висине слике странице теста. Претрага се врши целокупном ширином слике странице теста из разлога што је у пракси утврђено да дежурни лепе идентификационе бар-код налепнице било где при врху теста, а не на за то предвиђеном месту означеном правоугаоником довољне ширине, који се налази у левом делу заглавља теста. Такође, претрага се врши само на очекиваном делу странице теста,

односно у заглављу, а условно на дну странице теста, из разлог што претрага на осталим регионима странице теста нема смисла и захтева много више времена.

Претрага и читавање једнодимензионалних бар-кодова се врши коришћењем Пајтонове библиотеке *pyzbar*, која је намењена за детекцију једнодимензионалних и дводимензионалних бар-кодова. Функција за детекцију једнодимензионалних бар-кодова захтева прослеђивање слике на којој се врши претрага, као и конфигурационе датотеке. Међутим, детекција на сликама које нису довољно препроцесирани не даје увек најбоље резултате. Последице, идентификациони бар-код касније мора бити прочитан ручно, а затим мануелно уписан, што оставља простор за грешку.

Ипак, имајући у виду да претрага идентификационог бар-кода на необрађеним сликама страница теста даје солидне резултате, иницијално се она покреће на необрађеној слици. Таква претрага даје или исправно препознат једнодимензионални бар-код или не детектује никакав бар-код или детектује више од једног бар-кода. У случају да тачно један једнодимензионални бар-код није прочитан, потребно је извршити морфолошке операције које ће извршити елиминацију нечистоћа на сликама страница теста пре покретања поновне претраге.

Основне морфолошке операције над сликом (односно сваким њеним пикселем) називају се ерозија и дилатација. Природа ових операција зависи од структуралног елемента (енг. *kernel*) који се примењује над сликом (конволуција), као и броја итерација. Комбинацијом ових основних операција добијају се сложеније морфолошке трансформације као што су отварање (енг. *opening*), затварање (енг. *closing*), градијент итд.

Ерозија је операција којом се пиксел слике сматра светлим само ако су светли сви пиксели испод структуралног елемента на позицији тог пиксела слике. У супротном, пиксел се сматра тамним, односно врши се његова ерозија. Стога, бели региони на слици нестају, односно дебљина или величина објекта постају мањи. Ова операција је корисна у ситуацијама када је потребно елиминисати мали бели шум или раскинути повезаност између две повезане компоненте беле боје.

Дилатација је операција којом се пиксел слике сматра светлим ако је светао барем један пиксел испод структуралног елемента на позицији тог пиксела слике. У супротном, пиксел се сматра тамним. Стога, бели региони на слици се шире, односно дебљина или величина објекта постају већи. Ова операција је корисна након што се претходно изврши операција ерозије за елиминисање белог шума, јер ће она уз то и смањити величину белог објекта. Применом дилатације бели објекти ће се повећати, али се бели шум неће вратити. Такође, операција је погодна, уколико је потребно успоставити повезаност између две претходно повезане компоненте беле боје.

Отварање је операција која обухвата прво ерозију слике коју прати њена дилатација. Ова операција је корисна у уклањању белог шума. Затварање је операција која обухвата прво дилатацију слике, коју прати њена ерозија. Ова операција је корисна у попуњавању малих отвора у белим објектима.

Приликом скенирања папирног теста добијена је вишестранична *.tiff* датотека при чему су све слике страница теста таквог формата (енг. *greyscale*) да су њихови пиксели интензитета у опсегу од 0 (потпуно таман пиксел) до 255 (потпуно светао пиксел). Затим се врши трансформација којом се сваки пиксел претвара или у потпуно тамни или у потпуно светли пиксел (енг. *binarization*), у зависности од интензитета пиксела и референтне границе (енг. *thresholding*). Ова операција представља бинаризацију слике и служи за уклањање шума, односно филтрирање пиксела веома ниског или веома високог интензитета.

Додатно, користи се аутоматски избор референтне границе коришћењем Оцуовог алгоритма (енг. *Otsu's algorithm*). Поред тога, врши се и инверзија слике, да би светли пиксели

постали тамни, а тамни да би постали светли. Ово је неопходно урадити зато што се подразумева да су објекти сачињени од светлих пиксела и да се налазе на црној позадини.

За конволуционе операције над сликом изабран је правоугаоник као структурални елемент. Разлог за овај избор структуралног елемента лежи у чињеници да су облици од интереса једнодимензионални бар-кодови, који се састоје од вертикалних правоугаоника црне боје на белој позадини, односно белих правоугаоника на црној позадини на инвертованој слици. Додатно, димензије структуралног елемента су изабране тако да се правоугаоници једнодимензионалног бар-кода стопе у један облик који ће лакше бити идентификован.

Да би се ово постигло, над сликом се најпре примењује операција затварања која врши попуњавање отвора у белим објектима. Ово ефективно врши стапање правоугаоника једнодимензионалног бар-кода у један велики правоугаоник. Затим се примењује операција отварања над сликом да би се елиминисао бели шум.

Након ових морфолошких трансформација над сликом може се приступити процесу претраживања контура на слици. Контуре су криве које повезују граничне тачке које имају исту боју односно исти интензитет пиксела. Контура се може посматрати и као линија око самог објекта, односно крива која га ограничава.

Најпре се претраживане контуре филтрирају тако да преостану само оне контуре које испуњавају критеријум теста правоугаоника предвиђене величине. Затим се контуре уређују тако да се тестирају редом са лева на десно. Мотивација за уређивање контура представља то што је процес читавања бар-кодова спорији од процеса уређивања, те је редослед тестирања контура битан. Такође, број контура који се уређује је релативно мали, а и сам критеријум сортирања је лако израчунљив.

Тестирање контура започиње креирањем резултујуће црне шаблонске слике која је истих димензија као и оригинална слика дела странице. Затим се отисак правоугаоника контуре утискује белом бојом на ту слику. Након тога се извршава битска И (енг. *bitwise and*) операција између резултујуће слике и инвертоване оригиналне слике, а добијени резултат се инвертује. На тај начин се врши утискивање правоугаоника контуре који обухвата кандидата за једнодимензионални бар-код са оригиналне слике на потпуно белу слику.

Ова слика се потом прилагођава тако да буде довољне величине да обухвати правоугаону контуру, која представља кандидата за бар-код, ротирану за било који угао. Ово се обезбеђује одређивањем минималног обухватног круга око правоугаоне контуре. Затим се одређује нагибни угао дуже странице правоугаоника према замишљеној хоризонталној линији. Потребно је да се слика ротира тако да дужа страница правоугаоне контуре постане паралелна замишљеној хоризонталној линији.

Ротирање слике обавља се око њеног центра, те га је најпре потребно утврдити. Затим је потребно одредити и афину ротациону матрицу на основу центра слике и угла за који ће се извршити ротација. Након тога се ротација и извршава, при чему се црни пиксели добијени приликом ротације попуњавају белим пикселима.

Резултујућа слика сада представља део са оригиналне слике за који се претпоставља да садржи једнодимензионални бар-код. Она се додатно модификује применом морфолошких операција отварања и затварања да би се попуниле рупе у правоугаоним вертикалним линијама једнодимензионалног бар-кода, као и да би се очистио шум са слике. Након ових трансформација може се извршити процес читавања бар-кода са слике.

Уколико није прочитан тачно један бар-код, врши се декодовање свих бар-кодова и логовање грешке. У супротном случају, покушава се упаривање декодованих података из бар-кода са информацијама из конфигурационе датотеке. Уколико је упаривање успешно,

декодирани бар код се враћа као једини резултат. Препознавање једнодимензионалног бар-кода приказује Псеудо код 6.

```
def get_bar_code(page_image, config, logger):
    try:
        y_end = page_image.shape[0] // 5
        x_start = 0
        x_width = page_image.shape[1]
        bar_image = page_image[:y_end, x_start:x_start + x_width]
        barcodes = [code
                     for code in pyzbar.decode(bar_image, symbols=[ZBarSymbol.CODE128])
                     if satisfies_dimension(code, config['bc'])]
        if (len(barcodes)) != 1:
            bar_image = cv2.threshold(bar_image, 0, 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU)[1]
            morphed_image = cv2.morphologyEx(255 - bar_image, cv2.MORPH_CLOSE,
                                             cv2.getStructuringElement(cv2.MORPH_RECT,
                                                                         (config['rck_x'], config['rck_y'])))
            morphed_image = cv2.morphologyEx(morphed_image, cv2.MORPH_OPEN,
                                             cv2.getStructuringElement(cv2.MORPH_RECT,
                                                                         (config['rok_x'], config['rok_y'])))
            contours = sorted([(cnt, cv2.minAreaRect(cnt))
                              for cnt in filter(
                                  lambda cn:
                                      satisfies_rectangle(cv2.minAreaRect(cn), config['bc']),
                                      cv2.findContours(morphed_image,
                                                         cv2.RETR_LIST, cv2.CHAIN_APPROX_SIMPLE)[0]),
                              key=lambda cnt: cnt[1][0][0] - cnt[1][1][0]])
            for c, mar in contours:
                template = np.zeros(bar_image.shape).astype(bar_image.dtype)
                cv2.fillPoly(template, [c], (255, 255, 255))
                template = cv2.morphologyEx(template, cv2.MORPH_DILATE,
                                             cv2.getStructuringElement(cv2.MORPH_RECT,
                                                                         (config['dk_x'], config['dk_y'])))
                bar_image_cand = 255 - cv2.bitwise_and(template, 255 - bar_image)
                mec = cv2.minEnclosingCircle(c)
                bar_image_cand = get_sufficient_image(bar_image_cand, mec)
                angle = get_angle(mar[-1])
                if angle:
                    image_center = tuple(np.array(bar_image_cand.shape[1::-1]) / 2)
                    rot_mat = cv2.getRotationMatrix2D(image_center, angle, 1.0)
                    bar_image_cand = cv2.warpAffine(bar_image_cand, rot_mat,
                                                    bar_image_cand.shape[1::-1],
                                                    flags=cv2.INTER_LINEAR,
                                                    borderMode=cv2.BORDER_CONSTANT,
                                                    borderValue=(255, 255, 255))
                    bar_image_cand = cv2.morphologyEx(255 - bar_image_cand, cv2.MORPH_OPEN,
                                                       cv2.getStructuringElement(cv2.MORPH_RECT,
                                                                               (config['bc']['ok_x'],
                                                                               config['bc']['ok_y'])))
                    bar_image_cand = cv2.morphologyEx(bar_image_cand, cv2.MORPH_CLOSE,
                                                       cv2.getStructuringElement(cv2.MORPH_RECT,
                                                                               (config['bc']['ck_x'],
                                                                               config['bc']['ck_y'])))
                    barcodes = [code
                                for code in pyzbar.decode(255 - bar_image_cand,
                                                            symbols=[ZBarSymbol.CODE128])
                                if satisfies_dimension(code, config['bc'])]
            if (length := len(barcodes)) != 1:
                print(f'Test {config["test_file"]}')}
                for bc in barcodes:
                    decoded_bc_data = bc.data.decode()
                    print(f'\tDecoded {decoded_bc_data} data.')
                raise Exception(f'Found {length} bar-codes!')
            else:
                decoded_bc_data = barcodes[0].data.decode()
                if check_match_bc(config, decoded_bc_data):
                    print(f'Test {config["test_file"]} | Decoded {decoded_bc_data} data.')
                    return decoded_bc_data
                else:
                    raise Exception(f'Test {config["test_file"]} | BC = {decoded_bc_data}')
    except (Exception,) as e:
        logger.log_error(f'ERROR | {config["test_file"]} | {e}')
    return None
```

Псеудо код 6. Препознавање једнодимензионалног бар-кода

У случају неуспешног упаривања података, логује се грешка. Случајеви са грешком препознају се тако што функција не враћа тачно један читани бар код. У тим ситуацијама бар-кодови се морају ручно уносити од стране оператера. Ипак, ручни унос је и поред верификације подложен грешкама.

Све ове операције које обухватају изоловање контуре кандидата бар-кода, његово ротирање, попуњавање отвора у правоугаоним линијама једнодимензионалног бар-кода насталих у процесима тестирања кандидата и дигитализације теста, као и елиминацију шума са слике извршавају се у сврхе што бољих резултата у процесу читавања бар-кода са слике. Иако може деловати да постоји велики број операција које треба применити над сликом и пре самог процеса читавања бар-кода, већина бар-кодова биће прочитана и на оригиналној неизмењеној слици. То значи да се у тим случајевима ове операције неће ни извршити.

Међутим, у одређеним случајевима, тестови садрже такве бар-кодове за које је неопходно извршити додатне трансформације слике. То значи да су ти бар-кодови лоше налепљени од стране дежурних, оштећени у некој мери у процесу израде теста, његовог транспорта у центре за скенирање или његове дигитализације коришћењем скенера. Стога, правилна ротација бар-кода, као и добро осмишљене операције опоравка могу допринети препознавању и неких од таквих бар-кодова. Пример инвертоване оригиналне слике странице теста (са леве стране) и обрађене слике над којом ће се извршити читавање једнодимензионалног бар-кода илуструје Слика 41.



Слика 41. Пример оригиналног инвертованог и морфолошки обрађеног оригиналног бар-кода

4.2.2. Препознавање дводимензионалног бар-кода

Као што је објашњено у претходном потпоглављу, ненамерна ротација папирне странице теста не може бити блиска 90 степени у смеру казаљке на сату, односно у супротном смеру казаљке на сату. Она може износити или X степени или $180 \pm X$ степени у једном од два претходно наведена смера, где X представља вредност од неколико степени. Стога је претрага страничног дводимензионалног бар-кода иницијално ограничена на вршних 20 % слике странице теста, а уколико та претрага не успе, нова претрага се извршава на доњих 20 % слике странице теста.

За разлику од идентификационог једнодимензионалног бар-кода који дежурни лепе на страници теста, дводимензионални бар-кодови су утиснути у дигитални тест приликом његовог дизајнирања. Стога, могао би да се стекне утисак да за све остале странице теста осим прве није потребно додатно процесирање слике. Разлог за ово представља чињеница да се дводимензионални бар-код сигурно налази на за то предвиђеном месту и да је сигурно оријентисан на добар начин.

Међутим, папирни тестови и даље могу бити деформисани у процесу израде, транспорта или дигитализације. Управо из тог разлога, читавање дводимензионалних бар-кодова на сликама које нису довољно препроцесиране не даје увек најбоље резултате. Ипак, за препроцесирање слике странице теста потребно је знатно мањи број морфолошких операција

него у случају детектовања једнодимензионалних бар-кодова. Поред тога што су дводимензионални бар-кодови одштампани на тесту и не лепе се од стране дежурних, томе доприноси и чињеница да налепнице продукују више одсејаја приликом дигитализације папирних страница теста.

Ипак, ова иницијална претрага у довољном броју случајева пружа доста добре резултате и захтева мање утрошеног времена него примена морфолошких трансформација над сликом. Иницијална претрага се спроводи над вршним делом оригиналне слике странице теста коришћењем Пајтонове библиотеке *pyzbar*. Уколико ова претрага не резултује у тачно једном пронађеном дводимензионалном бар-коду, биће примењена поновна претрага над морфолошки измењеним делом слике странице теста. Ово је, свакако, боље решење, јер се у случају неочитавања дводимензионалног бар-кода и он мора прочитати ручно, а затим је потребно да се прочитани подаци ручно упишу, што оставља простор за грешку.

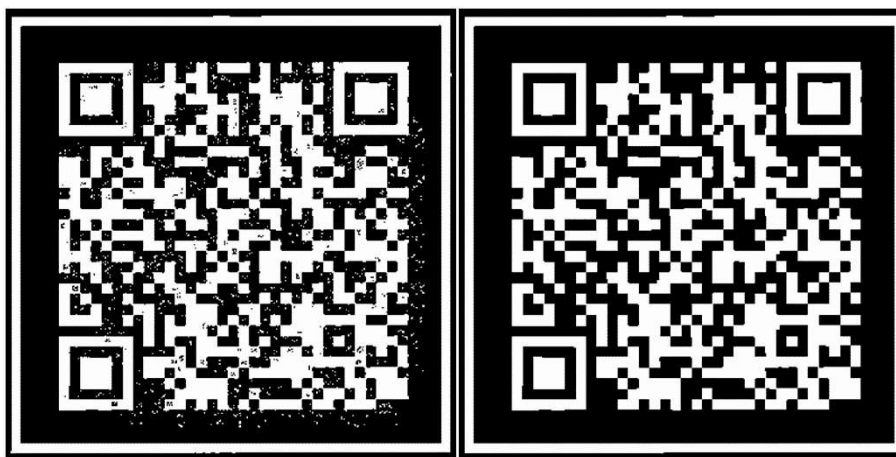
Слика се најпре трансформише операцијом којом се сваки пиксел претвара или у потпуно светли или потпуно тамни пиксел, у зависности од интензитета пиксела и референтне границе. Ова операција представља бинаризацију слике и служи за уклањање шума, односно филтрирање пиксела веома ниског или веома високог интензитета. Додатно, користи се аутоматски избор референтне границе коришћењем Оцуовог алгорита. Поред тога, врши се и инверзија слике, да би светли пиксели постали тамни, а да би тамни пиксели постали светли. Ово је неопходно урадити зато што се подразумева да су објекти сачињени од светлих пиксела и да се налазе на црној позадини.

За конволуционе операције над сликом изабран је квадрат као структурални елемент. Разлог за овај избор структуралног елемента лежи у чињеници да су дводимензионални бар-кодови састављени од црних квадратића који су на неким местима спојени тако да образују линије. У случају инвертоване слике, то су бели квадрати на црној позадини слике.

Најпре се над сликом примењује операција затварања која врши попуњавање отвора у белим објектима. Ово ће довести и до увећања белог шума, али не у довољној мери да би он постао проблематичан и помешан са квадратићима дводимензионалног бар-кода. Управо из тог разлога је изабрана релативно мала димензија структуралног елемента.

Затим се примењује операција отварања над сликом да би се у што већој мери елиминисао бели шум. У овом случају је изабрана нешто већа димензија структуралног елемента него у претходној операцији. Поред елиминисања белог шума, ова операција ће довести и до мањег оштећења облика од интереса, али то оштећење није превише проблематично, јер је претходном операцијом затварања облик од интереса опорављен у некој мери.

На самом крају поново се примењује операција затварања. У овом случају је изабрана нешто мања димензија структуралног елемента у односу на претходну операцију. Тиме се врши опоравак облика од интереса. Пример необрађеног дводимензионалног бар-кода на инвертованој оригиналној слици дела странице теста (леви део слике) и обрађеног дводимензионалног бар-кода са левог дела слике (десни део слике) приказује Слика 42.



Слика 42. Пример оригиналног инвертованог и морфолошки обрађеног оригиналног бар-кода

У случају да се најпре применила операција отварања, она би дефинитивно допринела елиминацији белог шума. Међутим, у неким ситуацијама би довела до нешто већег оштећења облика од интереса. Операција затварања која би је следила не би довела до рестаурације квадратића дводимензионалног бар-кода у случају избора релативно мање димензије структуралног елемента. Међутим, избор веће димензије структуралног елемента довео би до спајања неких од белих квадратића дводимензионалног бар-кода на инвертованој слици.

Након ових морфолошких трансформација над сликом може се приступити поновној претрази дводимензионалних бар-кодова и њиховом читавању са слике. Уколико није прочитан тачно један бар-код, врши се декодовање свих бар-кодова и логовање грешке. У супротном случају, декодовани подаци из бар-кода се упоређују са информацијама из конфигурационе датотеке теста. Провера подразумева потврду да се страница теста налази у оквиру теста у којем и треба да се нађе, као и да се страница налази у добром редоследу у оквиру тог теста. Обе ове информације су кодиране у дводимензионалном бар-коду.

Уколико се подаци поклапају, декодовани бар код се враћа као једини резултат. У случају непоклапања, логује се грешка. Случајеви са грешком препознају се тако што функција не враћа тачно један прочитани бар код. У тим ситуацијама бар-кодови се морају ручно уносити од стране оператера. Ипак, ручни унос је и поред верификације подложен грешкама.

Приликом читавања дводимензионалног бар-кода коришћењем Пајтонове екстерне библиотеке *pyzbar*, као резултат се добија и угао оријентације бар-кода на слици. Ова могућност је доступна имајући у виду да дводимензионални бар-код садржи три већа оријентирна испуњена квадрата. С обзиром да је дводимензионални бар-код уметнут на страницу теста у процесу његовог дизајнирања, он гарантовано има добру оријентацију.

Стога, сваки ненулти нагибни угао који се јави у процесу читавања значи да је страница теста ненамерно ротирана за одређени угао. Ова ненамерна ротација се дешава у процесу дигитализације папирног теста и угао ротације странице је углавном представљен малом вредношћу. Овај угао се користи тако да се страница теста рестаурира у правилно стање, односно да се њене странице поравнају и да краћа и дужа страница буду паралелне са X , односно Y осом, респективно.

Све примењене морфолошке операције над сликом странице теста извршавају се у сврхе што бољих резултата у процесу читавања бар-кода. Иако њихово извршавање одузима време, то извршавање се дешава условно, односно само ако не успе читавање дводимензионалног бар-кода на оригиналној, неизмењеној слици. Функција за детекцију дводимензионалних бар-кодова захтева прослеђивање слике странице теста на којој се врши

претрага, конфигурационе датотеке теста, као и број странице теста. Псеудо код 7 приказује препознавање дводимензионалног бар-кода.

```
def get_qr_code(page_image, config, page_num, logger):
    try:
        y_end = page_image.shape[0] // 5
        x_start = 0
        x_width = page_image.shape[1]
        qr_image = page_image[:y_end, x_start:x_start + x_width]
        qrcodes = [code
                    for code in pyzbar.decode(qr_image, symbols=[ZBarSymbol.QRCODE])
                    if satisfies_dimension(code, config['qr'])]
        if (len(qrcodes)) != 1:
            morphed = cv2.threshold(qr_image, 0, 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU)[1]
            morphed = cv2.morphologyEx(255 - qr_image, cv2.MORPH_CLOSE,
                                      cv2.getStructuringElement(cv2.MORPH_RECT,
                                                                (config['qr']['ck_x'], config['qr']['ck_y'])))
            morphed = cv2.morphologyEx(morphed, cv2.MORPH_OPEN,
                                      cv2.getStructuringElement(cv2.MORPH_RECT,
                                                                (config['qr']['ok_x'], config['qr']['ok_y'])))
            morphed = cv2.morphologyEx(morphed, cv2.MORPH_CLOSE,
                                      cv2.getStructuringElement(cv2.MORPH_RECT,
                                                                (config['qr']['ck_x'], config['qr']['ck_y'])))
            qrcodes = [code
                      for code in pyzbar.decode(morphed, symbols=[ZBarSymbol.QRCODE])
                      if satisfies_dimension(code, config['qr'])]
        if (length := len(qrcodes)) != 1:
            print(f'Test {config["test_file"]} ')
            for qr in qrcodes:
                decoded_qr_data = qr.data.decode()
                print(f'\tDecoded {decoded_qr_data} data.')
            raise Exception(f'Found {length} qr codes!')
        else:
            decoded_qr_data = qrcodes[0].data.decode()
            if check_match_qr(config, decoded_qr_data, page_num):
                print(f'Test {config["test_file"]} | Decoded {decoded_qr_data} data.')
                return decoded_qr_data, qrcodes[0].get_rotation_angle()
            else:
                raise Exception(f'Test {config["test_file"]} | QR = {decoded_qr_data}')
    except (Exception,) as e:
        error_text.append((f'\t[Page 1]', str(e)))
        logger.log_error(f'ERROR | {config["test_file"]} | {e}')
    return None, None
```

Псеудо код 7. Препознавање дводимензионалног бар-кода

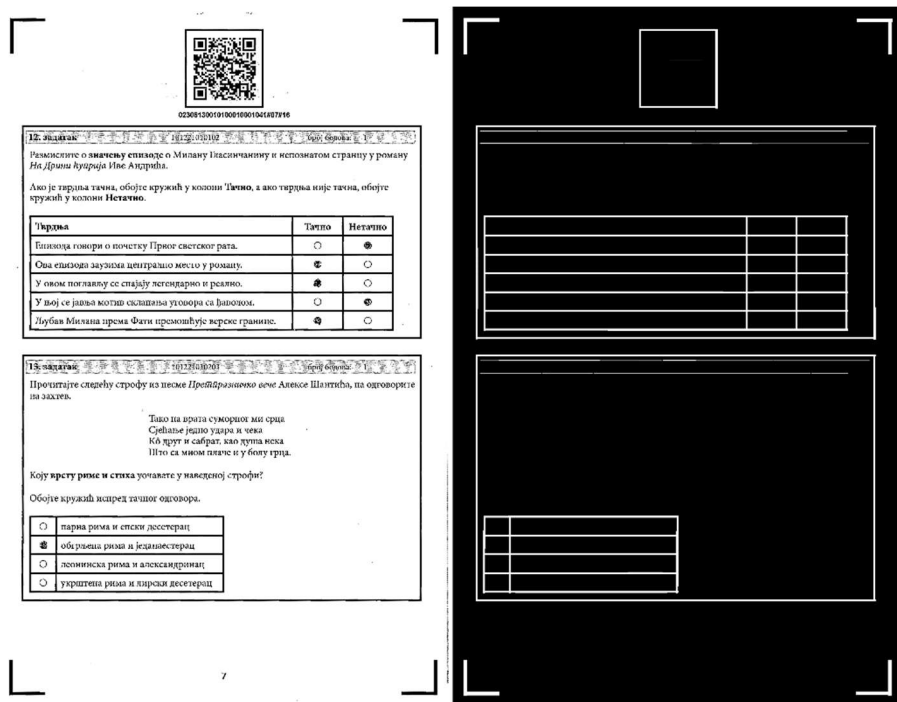
4.2.3. Препознавање граничника региона од интереса

Граничници имају сврху да означе регион од интереса, односно простор на папиру на коме се налазе сви битни подаци на тесту, као што су бар-кодови, питања и њихови одговори. На свакој страници постоји тачно 4 граничника, при чему се сваки налази у по једном углу странице. Граничници су зацрњени облик у виду ћириличног слова Г, који је ротиран у зависности од позиције на страници.

С обзиром да су питања и табеле одговора уоквирени правоугаоним облицима, а да се граничници састоје од хоризонталних и вертикалних линија, може се применити структурални елемент у виду хоризонталног и вертикалног правоугаоника над сликом тако да само облици састављени од оваквих правоугаоника преостану на слици. Поред тога, простори за лепљење идентификационог једnodимензионалног бар-кода, као и простор који садржи странични дводимензионални бар-код су такође правоугаоног облика. Због тога се може поставити питање да ли је најпре требало пронаћи све правоугаоне контуре на слици, а тек након тога приступити детекцији бар-кодова.

Међутим, претрага контура над непроцесираном сликом резултује у великом броју контура, због тога што је на оригиналној слици присутан велики број различитих елемената, који између осталог представљају и текстове питања и њихове одговоре. Они уједно

представљају шум који омета и успорава овај вид претраге. Због тога треба приступити елиминацији великог броја елемената са слике, односно употребити такво маскирање над сликом да преостану само хоризонталне и вертикалне линије. Пример произвољне ненасловне странице теста и слике добијене оваквим маскирањем илуструје Слика 43.



Слика 43. Оригинална и маскирана страница теста са питањима

На слици преостају граничници, правоугаоници питања и табела одговора, правоугаоник простора у коме се налази дводимензионални бар-код и, у случају насловне стране, и правоугаоник простора у коме би требало да се налази идентификациони једнодимензионални бар-код. Уколико би се ово маскирање иницијално применило, то би довело до тога да тражене линије можда и не буду детектоване. Разлог за овакав развој ситуације представља ненамерна ротација слике странице папирног теста која се дешава у процесу дигитализације папирног теста.

Стога, да би овакво маскирање имало смисла, најпре је потребно извршити ротирање слике странице папирног теста тако да она буде добро поравната. Потребни ротациони угао је информација коју управо обезбеђује претрага страничног дводимензионалног бар-кода. Уколико читавање страничног бар-кода не успе, онда ни даља обрада странице нема много смисла. Ротација идентификационог једнодимензионалног бар-кода не игра никакву важну улогу у одређивању нагиба слике странице папирног теста, јер се ова идентификација може залепити на произвољан начин.

Због свега наведеног, у фази детекције граничника и правоугаоника питања већ се може сматрати да је слика добро поравната, односно да претходно наведено маскирање над сликом има смисла. Маскирање се врши правоугаоним структуралним елементом, при чему није битно да ли се прво примењује хоризонтални или вертикални структурални елемент, јер се над добијеним резултатима примењује битска операција ИЛИ (енг. *bitwise or*). Маскирање применом овог структуралног елемента приказује Псеудо код 8.


```
def morph_image_lines(config, image, test_logger):
    test_logger.info('Morphing image...')
    (_, gray_inv) = cv2.threshold(image, 0, 255, cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)
    morphed = cv2.morphologyEx(gray, cv2.MORPH_CLOSE,
                               cv2.getStructuringElement(cv2.MORPH_RECT, (config['ok_x'], config['ok_y'])))
    horizontal_kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (config['line_len'], 1))
    horizontal_lines = cv2.morphologyEx(morphed, cv2.MORPH_OPEN, horizontal_kernel)
    vertical_kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (1, config['line_len']))
    vertical_lines = cv2.morphologyEx(morphed, cv2.MORPH_OPEN, vertical_kernel)
    image_lines = cv2.bitwise_or(horizontal_lines, vertical_lines)
    test_logger.info('Morphing image DONE.')
    return image_lines
```

Псеудо код 8. Маскирање странице теста коришћењем структуралног елемента

Све координате оквирних положаја региона од интереса у конфигурационој датотеци дате су узимајући позицију горњег левог граничника као референтну тачку. Иницијално је кроз алгоритам за детекцију граничника вршена претрага на сва 4 граничника. Првенствена мотивација је била да се проналаском свих граничника допринесе верификацији да страница теста није оштећена.

Међутим, утврђено је да обично у оштећеним страницама теста недостаје барем један граничник, али и да то оштећење није толико велико, због одређеног празног простора који постоји између региона од интереса и граничника. Уколико би се инсистирало на успешној претрази сва 4 граничника, тиме би се велики број страница теста прогласио невалидним и процесирање тих страница се не би обавило. Непроцесираних страница теста морале би ручно да се прегледају.

Ипак, биће покушана претрага свих граничника. Међутим, да би даље процесирање могло да се настави, довољно је успешно утврдити положај барем једног граничника. Након тога је потребно извести транслирање његових координата у горњи леви угао. Ово је, наравно, неопходно урадити, уколико то није управо горњи леви граничник.

Да би могли да се претражују граничници, најпре се морају пронаћи контуре на добијеној слици хоризонталних и вертикалних линија. Приликом претраживања контура специфицира се екстерни начин претраге, тако да преостану само контуре које немају родитељску контуру, односно представљају најспољашњије облике. Након тога се свака контура тестира тако што се исеца квадратни део оригиналне слике странице теста који садржи ту контуру. Потом се тај део слике скалира тако да одговара димензијама претходно учитаних шаблонских слика за сва 4 граничника.

Потврђивање пронађеног граничника обавља се тако што се врши упоређивање дела слике на којој се налази кандидат контура и шаблонских слика граничника. Уколико се изврши потврда горњег левог граничника, претрага осталих граничника није неопходна и она се завршава. У том случају је горњи леви граничник референти граничник, а у случају да он није потврђен, а јесте неки други, извршиће се његово мапирање у горњи леви граничник. У случају да није потврђен ни један граничник, сигнализирати се грешка.

Када се потврди граничник, преостале спољашње контуре представљају потенцијалне регионе питања. Овде се изузимају контуре које ограничавају просторе за бар-кодове. Чак и да се не изузму из разматрања, оне никад неће бити потврђене, јер се не могу укрстити са информацијама о оквирним положајима питања на тој страници.

Ова функција поред граничника враћа и регионе потенцијалних питања који треба да буду потврђени, да се не би касније поново вршила претрага на исте контуре питања. Поред тога, функција враћа и измењену слику на којој су преостале само хоризонталне и вертикалне линије, да се поново не би вршиле исте морфолошке трансформације над сликом. Псеудо код 9 приказује ову обраду.

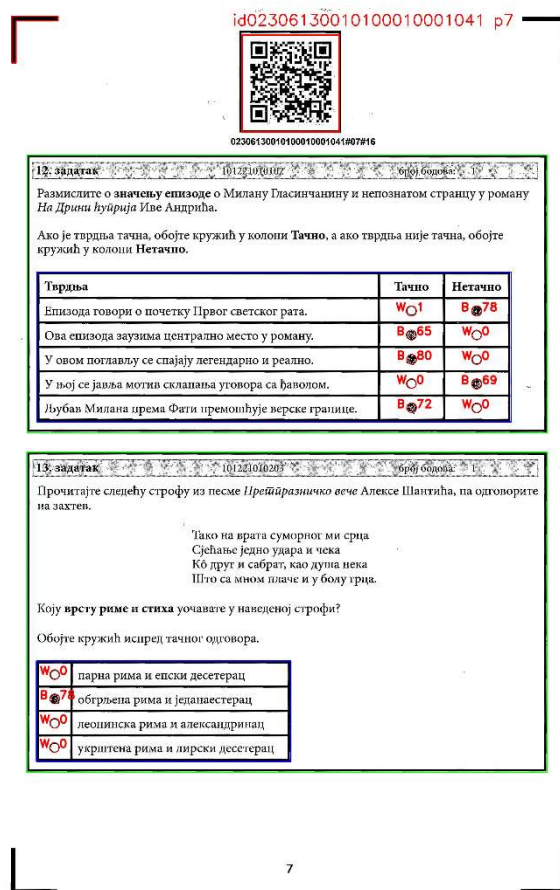

```

def get_zones(config, page_image, test_logger):
    image_lines = morph_image_lines(config, page_image, test_logger)
    contours, hierarchy = cv2.findContours(image_lines,
                                          cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    test_logger.info(f'Contours found: {len(contours)}')
    # Processing borders contours
    border_images = read_border_templates()
    borders_list = []
    for c in contours:
        x, y, w, h = cv2.boundingRect(c)
        dim = w if w < h else h
        region = image_lines[y: y + dim, x: x + dim]
        scaled_image = scale(region, border_images[0].shape[0] / region.shape[0])
        border, kind = match_template(scaled_image, border_images)
        if border:
            borders_list.append((border, kind))
            if kind == 'TL':
                borders_list.insert(0, (border, kind))
                break
    if not len(borders):
        raise Exception('Could not find borders!')
    if borders_list[0][1] != 'TL':
        bor = translate_to_left_border(page_image, borders_list)
    else:
        bor = borders_list[0][0]
    x, y, w, h = bor.bounding_rectangle()
    questions_region = [x, y, page_image.shape[1] - x - 1, page_image.shape[0] - y - 1]
    potential_questions = [c if is_squared(c) and is_inside(questions_region, c)
                           for c in contours]
    return image_lines, bor, potential_questions

```

Псеудо код 9. Дохватање граничника и потенцијалних зона питања

Слика 44 приказује комплетно процесирану страницу теста са идентификованим облицима. На слици се могу видети два питања класе вишеструки избор, која су анотирана приликом процесирања. Црвеном бојом означен је горњи леви граничник на страници, оквири дводимензионалног страничног бар-кода и контуре кругова. Зеленом бојом означени су оквири зона питања. Плавом бојом означени су оквири табеле одговора. Уз контуре кругова понуђених одговора доцртани су подаци о статусу и степену њихове зацрњености.



Слика 44. Комплетно процесирана страница теста

4.2.4. Процесирање питања класе „Вишеструки избор“

Региони одговора на питања типа вишеструки избор налазе се у оквиру региона питања којима припадају. Ови региони одговора су представљени регуларним правоугаоним табелама, које се зонирају од стране зонер оператера у подсистему за зонирање дигиталног теста. Процес утврђивања ових табела најпре подразумева претраживање правоугаоних контура, односно ћелија из којих се табела одговора састоји. Пример питања ове класе са попуњеног теста приказује Слика 45, а претрагу ћелија табеле одговора приказује Псеудо код 10.

12. задатак 101221010102 број бодова: 1		
Размислите о значању епизоде о Милану Гласинчанину и непознатом странцу у роману <i>На Дрини ћурија</i> Иве Андрића.		
Ако је тврдња тачна, обојте кружић у колони Тачно , а ако тврдња није тачна, обојте кружић у колони Нетачно .		
Тврдња	Тачно	Нетачно
Епизода говори о почетку Првог светског рата.	<input type="radio"/>	<input checked="" type="radio"/>
Ова епизода заузима централно место у роману.	<input checked="" type="radio"/>	<input type="radio"/>
У овом поглављу се спајају легендарно и реално.	<input checked="" type="radio"/>	<input type="radio"/>
У њој се јавља мотив склапања уговора са ђаволом.	<input type="radio"/>	<input checked="" type="radio"/>
Љубав Милана према Фати преомоћује верске границе.	<input checked="" type="radio"/>	<input type="radio"/>

Слика 45. Попуњено питање класе „Вишеструки избор“

```

def get_table_cells(image, zone):
    x, y, w, h = zone['X'], zone['Y'], zone['Width'], zone['Height']
    roi = image[y:y + h, x:x + w]
    contours, hierarchy = cv2.findContours(roi, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
    contours_list = sorted([(i, c, h)
                            for i, (c, h) in enumerate(zip(contours, hierarchy[0]))],
                            key=lambda el: cv2.contourArea(el[1]), reverse=True)

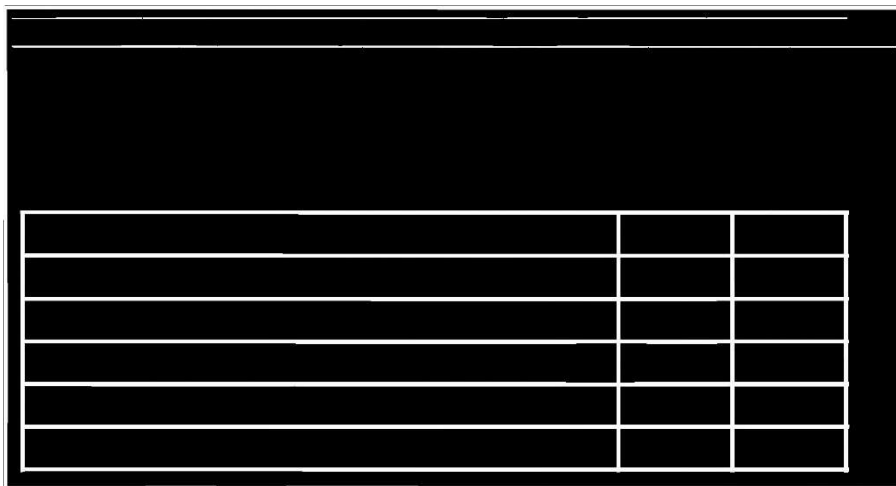
    cells = []
    for e in contours_list:
        if is_square_contour(e[1]) and e[2][3] == contours_list[0][0]:
            cells.append(cv2.boundingRect(e[1]))
    return get_grid_cells(cells, zone['Dim'])
except Exception as e:
    raise e

```

Псеудо код 10. Претрага ћелија табеле одговора

У зависности од начина претраге контура, она поред информација о локацијама контура може пружити још неке корисне информације. Због тога се приликом претраживања контура наводи хијерархијски метод претраге. Овај метод за сваку пронађену контуру даје информације о томе која контура му представља родитеља, односно која контура га садржи, затим информацију о томе да ли контура садржи неке друге контуре, односно да ли има децу, као и информацију о братским контурама које се налазе на истом нивоу хијерархије. Ове информације се кодирају тако што је свакој контури додељен јединствени идентификатор приликом њиховог претраживања.

Претрага контура извршава се над сликом над којом су преостале само хоризонталне и вертикалне линије. То значи да ће на слици преостати и облици правоугаоних табела и њихових ћелија. Због тога се најпре са те слике исеца регион зоне одговора који је раније означен у процесу зонарања, а затим усклађен са информацијама добијеним након проналаска граничника. Пример изолованих хоризонталних и вертикалних линија у региону питања илуструје Слика 46.



Слика 46. Хоризонталне и вертикалне линије преостале у региону питања

Након што се контуре пронађу, оне се сортирају опадајуће према површини облика којег ограничавају. На овај начин је прва контура у листи контура заправо она контура која ограничава табелу одговора са ћелијама. Сада се може проћи кроз преостале контуре и утврдити које контуре за директног претка имају управо ову контуру која ограничава табелу одговора.

Контуре које представљају потенцијалне ћелије табеле морају да прођу и тест којим се утврђује да ли су оне правоугаоног облика. Затим се листа кандидата ћелија табеле прослеђује на проверу која утврђује да ли њихов распоред чини правилну табелу и да ли су димензије

табеле у складу са очекиваним димензијама за ту зону одговора. Уколико се у било ком тренутку обраде деси грешка, она се сигнализира позиваоцу ове обраде. Тестирање да ли ћелије формирају правилну решетку приказује Псеудо код 11.

```
def get_grid_cells(rectangles, dim):
    grid = []
    try:
        for r in rectangles:
            flag = False
            for row in grid:
                if len(row) and same_row_rect(r, row[-1]):
                    flag = True
                    row.append(r)
                    break
            if not flag:
                grid.append([r])
        for row in grid:
            # el[0] is x coordinate
            row.sort(key=lambda el: el[0])
            # el[0] is a rectangle, el[0][1] is y coordinate
            grid.sort(key=lambda el: el[0][1])
        cnt = None
        for i, row in enumerate(grid):
            if cnt is None:
                cnt = len(row)
            elif len(row) != cnt:
                raise Exception(f'Row {i} has {len(row)} cells,'
                                f' initial had {cnt} cells')
        if len(grid) != dim[0] or len(grid[0]) != dim[1]:
            raise Exception(f'Wrong dimension! Expected {dim[0]} x {dim[1]}'
                            f'| Got {len(grid)} x {len(grid[0])}')
        if cnt:
            for j in range(len(grid[0])):
                for i in range(len(grid) - 1):
                    if not same_col_rect(grid[i][j], grid[i + 1][j]):
                        raise Exception(f'Different column {i, j}: {grid[i][j]}'
                                        f' and {i + 1, j}: {grid[i + 1][j]}')
    except Exception as e:
        raise e
    return grid
```

Псеудо код 11. Тест правилне решетке ћелија

Провера се обавља покушајем формирања правилне решетке од прослеђених правоугаоника контура ћелија. Правоугаоници се најпре групишу тако да се формира листа листи, при чему унутрашње листе садрже правоугаонике који се налазе у истом реду. Затим се сваки елемент спољашње листе, који је уједно и листа, уређује у таквом поретку тако да се у оквиру једног реда најпре нађу леви, а потом десни правоугаоници, односно елементи се уређују по вредности X координате растуће. Потом се елементи спољашње листе уређују у таквом поретку тако да се у њој најпре нађу горњи, а потом доњи редови, односно елементи се уређују по вредности Y координате растуће.

Након тога се испитује да ли сваки ред ове решетке има исти број правоугаоника и пријављује се грешка сигнализирањем изузетка, уколико то није случај. Уколико је провера успешна, врши се провера којом се испитује да ли су димензије табеле очекиване за ту зону одговора. Уколико је ова провера неуспешна, пријављује се грешка сигнализирањем изузетка.

Уколико је претходна провера успешна, врши се додатна провера којом се испитује да ли се правоугаоници који се у оквиру сваке врсте налазе под истим редним бројем заправо налазе у истој колони. Уколико то није случај, пријављује се грешка сигнализирањем изузетка. Уколико је провера успешна, решетка правоугаоника се враћа као резултат обраде. Сliku насталу одсецањем табеле са оригиналне слике региона питања илуструје Слика 47.

Тврдња	Тачно	Нетачно
Епизода говори о почетку Првог светског рата.	<input type="radio"/>	<input checked="" type="radio"/>
Ова епизода заузима централно место у роману.	<input checked="" type="radio"/>	<input type="radio"/>
У овом поглављу се спајају легендарно и реално.	<input checked="" type="radio"/>	<input type="radio"/>
У њој се јавља мотив склапања уговора са ђаволом.	<input type="radio"/>	<input checked="" type="radio"/>
Љубав Милана према Фати премашћује верске границе.	<input checked="" type="radio"/>	<input type="radio"/>

Слика 47. Исечени регион питања ове класе питања

Када се успешно утврди постојање табеле правоугаоних ћелија одговора за неко питање, следећи корак представља утврђивање кружних облика у тим ћелијама, које кандидати зацртају у процесу тестирања кандидата. Свака ћелија садржи тачно један круг и он се налази тачно у средини те ћелије. Стога, делује да нема потребе тражити кружне облике у ћелијама, већ да је довољно само проверити степен зацртаности средине сваке ћелије да би се утврдила означеност понуђеног одговора.

Међутим, упутство за попуњавање одговора на тесту настоји да кандидатима објасни да је приликом попуњавања одговора неопходно у потпуности зацртати кругове изабраних понуђених одговора. У случају непоштовања упутства, кандидати би могли да на специфичан начин обележавају своје изабране одговоре (нпр. прецртавањем одговора, њиховим заокруживањем, уписивањем зацртаног симбола квадрата уместо круга итд.). Ово би кандидати могли радити са циљем да нагосте свој идентитет на тесту, при чему би ова информација могла да се употреби у неетичке сврхе.

Стога, претрага кружних облика има за циљ да у одређеној мери утврди постојање исправног начина попуњавања теста, према инструкцијама наведеним у тесту. Процес утврђивања круга спроводи се над сваком кандидат контуром понуђених одговора. Тест кружности контуре обухвата низ од неколико математичких операција које се спроводе над контуром.

За сваку контуру кандидата најпре се рачуна индекс поузданости кружног облика. Овај индекс се рачуна као однос између површине облика ограниченог контуром кандидатом помножене вредностима 4 и Π (π) и квадриране вредности обима облика. За идеалне кругове овај однос износи тачно 1, с обзиром да је формула за површину круга $P = r^2\pi$.

С обзиром да у процесима израде папирног теста, његовог транспорта и дигитализације може доћи до оштећења садржаја папирног теста, добијени резултат се упоређује на вредност која се налази у околини вредности 1. Ова вредност околине је одређена експериментално и налази се у конфигурационој датотеци теста. Уколико ова провера не успе, овде се обрада завршава и враћа се негативан статус.

У случају да претходна провера успе, обрада се наставља да би се додатно валидирала контура. Надаље се израчунавају центар контуре и дохватају се њене тачке, с обзиром да је приликом иницијалног претраживања контуре искоришћен метод који ће апроксимативно, са мањим бројем тачака одредити контуру. Затим се за све тачке контуре рачуна њихова дистанца од утврђеног центра контуре.

Након тога се одређује медијална вредност израчунатих дистанци. Затим се одређује да ли довољан број тачака испуњава својство да се њихова дистанца до центра контуре налази у епсилон околини медијалне дистанце до центра. Ова вредност епсилон околине је одређена експериментално и налази се у конфигурационој датотеци теста.

Уколико довољан број тачака кандидат контуре испуњава овај услов, враћа се позитиван статус. Уколико то није случај, враћа се негативан статус. Псеудо код 12 утврђује да ли је контура кружног облика.

```
def is_circle(candidate, thresh, eps):
    num = 4 * math.PI * get_contour_area(candidate)
    den = candidate.get_perimeter() ** 2
    if not ((1 - thresh) <= num / den <= (1 + thresh)):
        return False
    center, points = candidate.get_moments(), candidate.get_points()
    median = median([get_distance(center, p) for p in points])
    dists = [
        abs(distance(center, p) - median_dist) / median_dist < eps
        for p in c.get_points()
    ]
    if sum(dists) <= len(dists) * thresh:
        return False
    return True
```

Псеудо код 12. Тестирање да ли је кандидат контура кружног облика

Претрага кругова који представљају понуђене одговоре може да започне када се утврди постојање табеле одговора у зони тренутног одговора. Уколико табела одговора није пронађена, емитоваће се изузетак, који ће бити логован у овој обради. Када се успешно пронађе табела одговора на слици хоризонталних и вертикалних линија, наставља се са исецањем зоне одговора са оригиналне слике.

Овај део оригиналне слике са зоном одговора пролази кроз трансформације слике као што су бинаризација уз коришћење аутоматског адаптивног трешхолда и инвертовање. Затим се приступа тражењу контура на оваквој слици, са циљем да се изолују контуре кандидати, које представљају понуђене одговоре у облику круга. Потом се за сваку раније утврђену ћелију табеле одговора проналази кандидат контура која се налази у оквиру ње.

За сваку кандидат контуру се испитује да ли се она налази у средини неке од ћелија табеле одговора и да ли задовољава својство круга. Критеријуми стриктности, односно лабавости теста круга, као и одређивања региона центра наведени су конфигурационом датотеком теста, а добијени су експерименталним путем. Уколико се контура пронађе, проналази се минимални окружујући круг те контуре и она се избацује из листе контура. Такође, за ту ћелију табеле неће више бити тражене кружне контуре.

Затим се пронађени број кругова, односно понуђених одговора, упоређује са стварним бројем понуђених одговора у тој зони одговора. Уколико је број пронађених кругова мањи од очекиваног, емитује се изузетак и логује се грешка да је пронађен недовољан број кругова. У супротном случају, обрада се наставља даље.

Приликом претраживања, може се појавити већи број кругова од очекиваног. Стога се пронађени кругови одговора сортирају према својој површини опадајуће и изабира се онолико највећих кругова колико је очекивани број понуђених одговора у овој зони одговора. Критеријум оваквог изабирања кругова има своје оправдање у томе што приликом дизајнирања теста постоји ограничење да у табели одговора симболи кругова понуђених одговора морају бити већи од кружних симбола или слова који се користе у тексту одговора.

Већи број кругова од очекиваног се може појавити из више разлога. Текст понуђеног одговора може садржати мало или велико слово О, које се може наћи у центру неке од ћелија. Поред тога, кандидати могу доцртати нови круг у мрежи понуђених одговора. Такође, у процесу, папирни тест може бити оштећен и та оштећења се могу манифестовати и у виду мрља кружног облика.

Након тога, одређује се средња вредност полупречника очекиваног броја кругова који су преостали. Одређују се и опсези минималног и максималног одступања величине полупречника круга на основу добијене средње вредности полупречника круга. Затим се из листе преосталих кругова избацују сви кругови чија вредност полупречника не упада у овај опсег вредности.

Уколико је број преосталих кругова мањи од очекиваног броја за ову зону одговора, емитује се изузетак и логује се грешка да не постоји довољни број кругова тражене величине. У супротном, за преостале кругове се покушава формирање правилне решетке која има очекиване димензије наведене у зони овог одговора. Принцип формирања решетке је сличан формирању решетке ћелија табеле одговора. Уколико формирање решетке не успе, емитује се изузетак и логује се грешка. Пример слике табеле одговора на којој се врши претрага кругова понуђених одговора и утврђивање њихове зацрњености илуструје Слика 48.

Тврдња	Тачно	Нетачно
Епизода говори о почетку Првог светског рата.	<input type="radio"/>	<input checked="" type="radio"/>
Ова епизода заузима централно место у роману.	<input checked="" type="radio"/>	<input type="radio"/>
У овом поглављу се спајају легендарно и реално.	<input checked="" type="radio"/>	<input type="radio"/>
У њој се јавља мотив склапања уговора са ђаволом.	<input type="radio"/>	<input checked="" type="radio"/>
Љубав Милана према Фати премошћује верске границе.	<input checked="" type="radio"/>	<input type="radio"/>

Слика 48. Морфолошки измењена слика питања над којом се врши претрага кругова

У случају да је успешно формирана решетка кругова одговора, за сваки круг се одређује степен његове зацрњености. Резултат ове обраде има сличну матричну структуру као и решетка кругова која јој је прослеђена. Питање ове класе анотирано резултатима добијеним у процесу препознавања илуструје Слика 49. Псеудо код 13 приказује дохватање кругова из зоне одговора.

12. задатак	101221010102	број бодова: 1
Размислите о значењу епизоде о Милану Гласинчанину и непознатом странцу у роману <i>На Дрини ћуприја</i> Иве Андрића.		
Ако је тврдња тачна, обојте кружић у колони Тачно , а ако тврдња није тачна, обојте кружић у колони Нетачно .		
Тврдња	Тачно	Нетачно
Епизода говори о почетку Првог светског рата.	W 01	B 78
Ова епизода заузима централно место у роману.	B 65	W 0
У овом поглављу се спајају легендарно и реално.	B 80	W 0
У њој се јавља мотив склапања уговора са ђаволом.	W 0	B 69
Љубав Милана према Фати премошћује верске границе.	B 72	W 0

Слика 49. Питање ове класе анотирано резултатима процесирања

```

def get_circles(config, page, image_lines, answer, logger):
    try:
        table_cells = get_table_cells(image_lines, answer)
        x, y, w, h = answer['X'], answer['Y'], answer['Width'], answer['Height']
        roi = image[y:y + h, x: x + w]
        _, roi = cv2.threshold(roi, 0, 255, cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)
        contours, _ = cv2.findContours(roi, cv2.RETR_LIST, cv2.CHAIN_APPROX_SIMPLE)
        circles = []
        for cell in table_cells:
            for c in contours:
                if is_centered(c, cell, config['cc_min'], config['cc_max'])
                and is_circle(c, config['c_ind_eps'], config['c_eps']):
                    circles.append(cv2.minimum_enclosing_circle(c))
                    contours.remove(c)
                    break
        thresh = get_answers_count(answer)
        if len(circles) < thresh:
            raise NotEnoughCircles()
        circles = sorted(circles, key=lambda c: c[1] ** 2 * math.pi, reverse=True)[:thresh]
        mean_rad = math.mean([c[1] for c in circles])
        min_rad, max_rad = config['c_rad_low'] * mean_rad, config['c_rad_max'] * mean_rad
        circles = [c for c in circles if is_between(c, min_rad, max_rad)]
        if len(circles) < thresh:
            raise NotEnoughCircles()
        circles_grid = get_grid(circles, answer)
        circles_fullnes = get_fulness_status(config, roi, circles_grid)
        return circles_fullnes
    except Exception as e:
        logger.log_error(e)
        return None

```

Псеудо код 13. Дохватање кругова и њихових статуса из зоне одговора

Статус зацрњености кругова одређује се тако што се за сваки круг утврди број црних пиксела који он садржи. Затим се тај број подели са формулом за површину круга да би се израчунао степен зацрњености круга. Израчунати степени зацрњености чувају се у матричној структури која по димензијама одговара прослеђеној матричној структури кругова за коју се утврђује статус зацрњености.

Приликом рачунања степена зацрњености, прати се и ажурира минимални и максимални ниво зацрњености појединачних кругова. Након што се израчуна степен зацрњености за све кругове, минималне и максималне вредности степена зацрњености постају познате. На основу њих је могуће адаптивно, у зависности од начина попуњавања кругова од стране кандидата утврдити статус зацрњености круга.

На основу претходно израчунатих минималних и максималних степена зацрњености израчунавају се максимални горњи праг степена зацрњености до којег се круг сматра непопуњеним и минимални доњи праг степена зацрњености од којег се круг сматра попуњеним. Наравно, приликом израчунавања ових прагова води се рачуна о граничним ситуацијама када кандидат може да попуни све кругове понуђених одговора (што је мало вероватно, али је могуће) или када кандидат не попуни ниједан одговор. Због тога се у сврхе одређивања динамичке адаптивне границе зацрњености користе информације из конфигурационе датотеке теста.

За све вредности степена зацрњености које спадају у опсег вредности ограничен горњим прагом као минималном вредношћу овог опсега и доњим прагом као максималном вредношћу овог опсега, не може се са сигурношћу одредити статус зацрњености. За такве случајеве неопходан је каснији ручни увид прегледача у одговоре ове зоне одговора. Између осталог, из тог разлога се и исецају и чувају све слике зона одговора.

Када се израчунају адаптивне вредности прагова степена зацрњености, приступа се одређивању статуса зацрњености за сваки круг. Статус зацрњености за све кругове чува се у матричној структури која по димензијама одговара решетки кругова понуђених одговора, односно сваки елемент матрице одговара једном кругу одговора и садржи тројку вредности

коју чине круг, његов статус зацрњености и степен зацрњености. Псеудо код 14 одређује статус зацрњености кругова.

```
def get_fullnes_status(config, page, circles_grid):
    filled = []
    low, high = 1, 0
    for row in circles_grid:
        filled.append([])
        for c in row:
            ratio = 1. * count_black_pixels(page, c) / (c.get_radius() ** 2 * math.PI)
            filled[-1].append(ratio)
            low, high = min(low, ratio), max(high, ratio)
        lower = median(0, config['c_trh_low'], low + config['c_trh_low'] * (high - low))
        upper = median(config['c_trh_high'] - config['c_trh_low'], config['c_trh_high'],
                        low + config['c_trh_high'] * (high - low))

    status = []
    for row_c, row_f in zip(circles_grid, filled):
        status.append([])
        for c, f in zip(row_c, row_f):
            status[-1].append((c, None
                                if lower < f < upper
                                else
                                True if f > upper
                                else False)
                               )

    return status
```

Псеудо код 14. Одређивање статуса зацрњености кругова

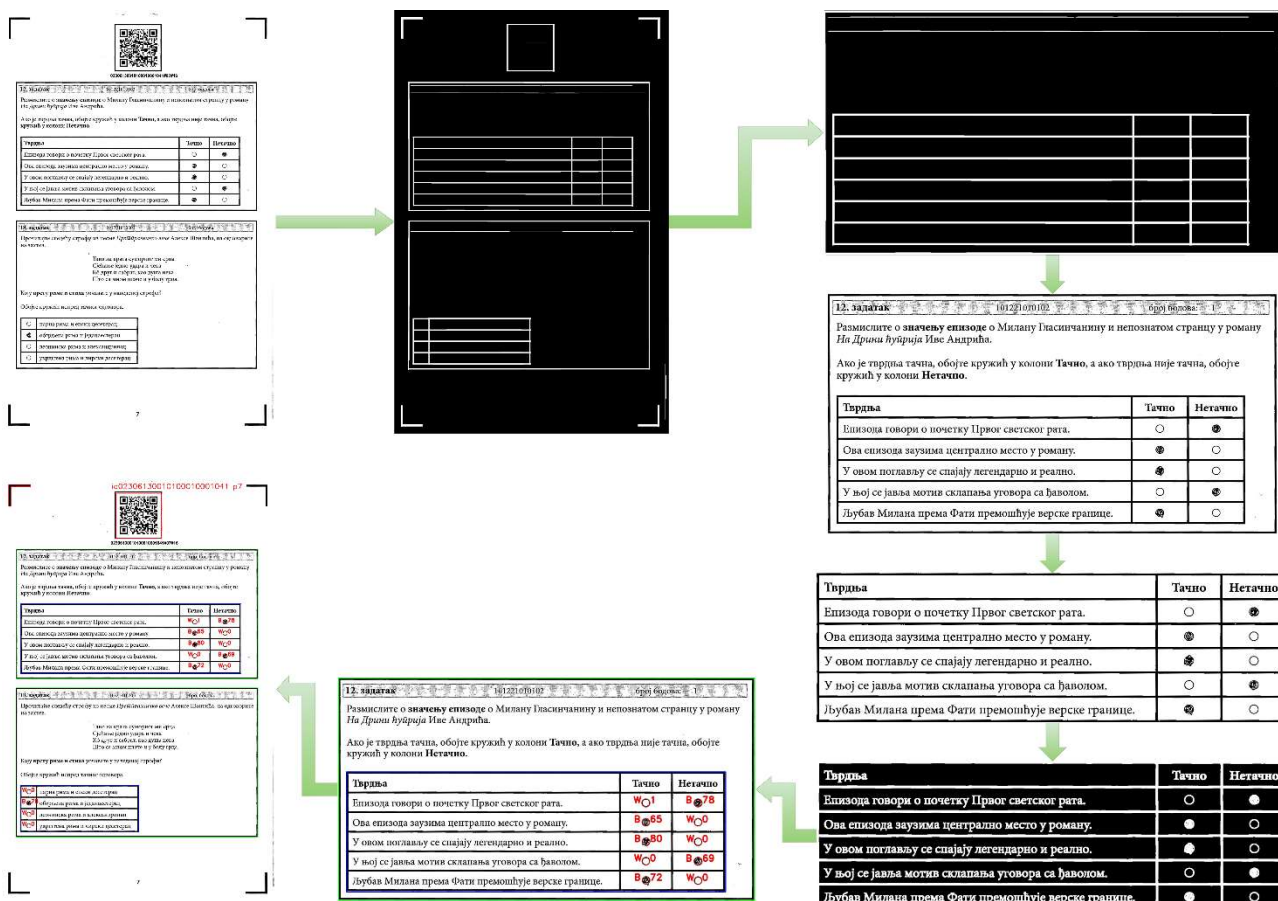
Број црних пиксела који сваки круг садржи одређује се на делу оригиналне слике над којом је раније примењена бинаризација, адаптивно трешхолдовање и инвертовање. Стога, практично се одређује број белих пиксела, уместо црних, јер бели пиксели дефинишу облике на црној позадини инвертоване слике. Део слике се одређује на основу координата центра круга и његовог полупречника и представља квадрат који описује тај круг.

Број затамњених пиксела се одређује проласком кроз све пикселе који чине квадратно исечени део слике одређен у претходном кораку. При томе, броје се само пиксели који немају интензитет 0 и који су удаљени од центра круга, а и самог квадрата, мање од вредности полупречника круга, а добијена вредност представља резултат ове функције. Псеудо код 15 одређује број црних пиксела који сваки круг садржи.

```
def count_black_pixels(image, c):
    (x, y), r = c
    circle_seg = image[y - r:y + r, x - r:x + r]
    cen = [circle_seg.shape[1] // 2, circle_seg.shape[0] // 2]
    result = 0
    for i in range(image.shape[1]):
        for j in range(image.shape[0]):
            result += 1 if (cen[0] - i) ** 2 + (cen[1] - j) ** 2 < r and not image[j][i] else 0
    return result
```

Псеудо код 15. Одређивање броја црних пиксела круга

Слика 50 приказује целокупан след корака за процесирање питања ове класе, укључујући странице теста или њихове сегменте који су настали у том процесу. Зелене стрелице означавају усмереност корака у овом процесу.



Слика 50. Кораци у процесирању питања ове класе

4.2.5. Процесирање питања класе „Повезивање“

Предложени софтверски систем може да процесира питања ове класе која су дата у једном од два формата. Први формат подразумева да се у питању од кандидата захтева да у датом тексту бирају своје одговоре подвлачењем једне или више речи тог текста. У овом случају се регион одговора не зонира од стране зонер оператера у подсистему за зонирање дигиталног теста, већ одговара региону питања којем понуђени одговори припадају.

Најпре се дохвата шаблонска слика региона одговора са текстом на празном тесту, њена бинаризована инвертована варијанта и речи текста које су пронађене у оквиру региона одговора. Затим се са слике странице попуњеног теста одсеца слика коју чини зона одговора овог питања. Након тога се врши усклађивање слике зоне одговора са странице попуњеног теста према шаблонској слици региона одговора тако да оне буду што боље поравнате.

Затим за сваку реч текста за овај одговор треба утврдити да ли је она подвучена или није. Приликом дохватања речи текста за сваку реч су одређене границе простора испод речи који се проверава на зацрпљеност, да би се утврдило да ли је реч подвучена или не. Због тога се са слике поравнате зоне одговора за тестирану реч издваја правоугаони облик који ће се тестирати на степен зацрпљености.

Уколико овај тест врати статус да реч није подвучена, врши се додатна провера да ли је та реч подвучена. Зато се најпре прави структурални елемент правоугаоника, односно хоризонталне линије, чије су димензије одређене конфигурационом датотеком теста. Након тога се слика зоне одговора попуњеног теста модификује морфолошким трансформацијама да би настала нова слика на којој ће преостати само хоризонталне линије.

Са те новодобијене слике хоризонталних линија зоне одговора се за тестирану реч издваја правоугаони облик. Овај правоугаоник је нешто другачијих димензија и налази се на другачијој локацији на слици у односу на претходно изоловани правоугаони простор испод речи. Потом се за издвојени правоугаони облик врши одређивање степена зацрњености.

Уколико прва провера статуса подвучености речи врати резултат да је реч подвучена, он се за ту реч памти. Друга провера подвучености речи је условна и извршава се само, ако прва провера не врати резултат да је реч подвучена. Какав год да је резултат те друге провере, он се памти за ту реч.

Када се за све речи утврди статус подвучености, модификује се структура података која садржи све речи са шаблонске слике зоне одговора тако да преостану само информације од интереса. На крају, ова обрада у случају успеха враћа као резултат све речи са шаблона зоне одговора. У случају да се у току обраде деси било каква невалидна ситуација, емитује се изузетак и логује се грешка. Псеудо код 16 врши процесирање првог формата питања ове класе.

```
def get_under(config, page, answer, test_logger):
    try:
        template_image, templ_img_bin_inv, template_words = get_template_data(config, answer)
        x, y, w, h = answer['X'], answer['Y'], answer['Width'], answer['Height']
        answer_image = page[y:y + h, x:x + w]
        aligned_bin_inv_answer_image = get_aligned_image(config, answer_image, template_image)
        for t_word in template_words:
            x, y, w, h = t_word['rect under']
            under_word_image = aligned_bin_inv_answer_image[y:y + h, x:x + w]
            flag = get_word_status(under_word_image, t_word, config['match_thr1'])
            if not flag:
                horizontal_kernel = cv2.getStructuringElement(cv2.MORPH_RECT,
                                                                (config['line_len'], 1))
                horizontal_lines = cv2.morphologyEx(aligned_bin_inv_answer_image,
                                                    cv2.MORPH_OPEN, horizontal_kernel)
                x, y, w, h = t_word['rect under lines']
                under_word_image = horizontal_lines[y:y + h, x:x + w]
                flag = get_word_status(under_word_image, t_word, config['match_thr2'])
            t_word['status'] = flag
        remove_unnecessary_keys(template_words)
        return template_words
    except Exception as e:
        logger.log_error(e)
        return None
```

Псеудо код 16. Процесирање првог формата питања ове класе

Дохватање шаблонске слике региона одговора празног теста и свих речи текста са те слике које чине понуђени одговор започиње тако што се најпре проверава да ли су већ раније ови подаци учитавани. Уколико то јесте случај, они се враћају као резултат ове обраде. Уколико то није случај, потребно је учитати шаблонску слику региона одговора из локалног директоријума и наставити даље са обрадом.

Затим је потребно искористити технике оптичког препознавања знакова да би се препознале све речи које чине понуђени одговор. У ове сврхе користи се Пајтонова библиотека *pytesseract* при чему се подешава да се траже речи језика и писма наведене конфигурационом датотеком теста. За сваку реч добијају се информације о словима које је чине, поузданости у препознату реч, координате правоугаоника који ту реч окружује, редни број речи у линији текста, број линије текста, редни број линије у параграфу, број параграфа и редни број параграфа у блоку текста.

Добијене речи се филтрирају тако да преостану само оне чија је поузданост већа од оне наведене конфигурационом датотеком и које представљају текст ненулте величине. Уводе се додатне информације које се рачунају на основу претходно добијених информација као што је нпр. информација о томе да ли се реч налази у последњој линији последњег параграфа блока текста. Поред тога, на основу информација добијених из идентификованих речи рачунају се

средње вредности основне линије текста (енг. *baseline*), као и врх наредне линије текста (енг. *top*).

Претходно израчунате вредности користиће се у одређивању координата правоугаоника за које ће се утврђивати степен зацрњености испод речи, односно да ли је нека реч подвучена. У рачунању степена зацрњености простора испод речи користе се две методе. Стога, ове две вредности се морају израчунати за сваку реч понуђеног одговора на шаблонској слици, односно за сваку реч биће формирана два правоугаоника за које ће се одређивати степен зацрњености.

Прва метода ће користити израчунавање над бинарном инвертованом шаблонском сликом, насталом у процесу претходне бинаризације, адаптивног трешхолдовања и инвертовања шаблонске слике региона одговора. Овде се координате правоугаоника за сваку реч рачунају тако да се они простиру од средње вредности базне линије текста која је раније израчуната па до врха речи у следећој линији, односно до краја региона одговора, уколико су у питању речи из последње линије последњег параграфа у блоку текста. Ови правоугаоници се за сваку реч простиру целом ширином те речи.

Друга метода ће користити израчунавање над сликом хоризонталних линија шаблонске слике одговора, насталом у процесу претходне бинаризације, адаптивног трешхолдовања и инвертовања. У ове сврхе се користи структурални елемент у виду правоугаоне линије, чије су димензије дефинисане конфигурационом датотеком. У овој методи се за сваку реч користе правоугаоници из претходне обраде који су проширени по *Y* оси нагоре за одређени померај. Висина ових правоугаоника је проширена за тај померај, при чему положај њихове доње ивице остаје непромењен.

Мотивација за ово проширење и коришћење друге методе огледа се у томе што се у пракси показало да кандидати често реч подвлаче тако што нацртају линију која пролази кроз слова саме речи, односно линија пролази изнад нивоа основне линије текста. Такође, у неким ситуацијама линија пролази тачно по основној линији текста, при чему је тада јако тешко одредити да ли је реч подвучена, нарочито ако се састоји од слова која у изабраном фонту имају зацрњени део основне линије. У таквим ситуацијама проверавање зацрњености простора испод речи коришћењем прве методе неће дати добре резултате.

У оба случаја утврђивање степена зацрњености правоугаоника речи врши се тако што се најпре правоугаоници изделе на одређен број сегмената који је дефинисан конфигурационом датотеком теста. Затим се за сваки сегмент исеца део одговарајуће слике (бинарне инвертоване слике у првој методи, односно бинарне инвертоване слике хоризонталних линија у другој методи) и израчунава број црних пиксела, односно белих на инвертованој слици. Вредности израчунате за све сегменте оба правоугаоника чувају се за сваку реч.

На самом крају се обрађени подаци о шаблонској слици и њеним речима за одговор овог питања чувају у локалном директоријуму. Сваки следећи пут ће се ове вредности дохватати, а потребно их је израчунати само први пут. Ова обрада као резултат враћа шаблонску слику региона одговора питања, њену бинаризовану инвертовану варијанту и речи одговора са шаблонске слике заједно са информацијама за сваку реч. Псеудо код 17 приказује ову обраду.

```

def get_template_data(config, answer):
    if ret := is_loaded_template(config, answer['TemplateName']):
        return ret['templ_img'], ret['data']
    templ_img = get_or_load_template_image(answer['TemplateName'])
    data = ocr.image_to_data(templ_img, output_type='dict', lang=config['language'])
    max_block = max(data['block_num'][i] for i in range(len(data['text'])))
    max_par = max(data['par_num'][i] for i in range(len(data['text'])))
    max_line = max(data['line_num'][i] for i in range(len(data['text'])))
    if data['block_num'][i] == max_block and data['par_num'][i] == max_par
    data = [
        {
            'text': text.strip(),
            'conf': data['conf'][i],
            'rect': [data['left'][i], data['top'][i], data['width'][i], data['height'][i]],
            'block_num': data['block_num'][i],
            'par_num': data['par_num'][i],
            'line_num': data['line_num'][i],
            'last_line': data['line_num'][i] == max_line
                        and data['par_num'][i] == max_par
                        and data['block_num'][i] == max_block
        }
        for i, text in enumerate(data['text'])
        if data['conf'][i] > config['match_conf']
        and data['block_num'][i] > 1 and text.strip() != ''
    ]
    baselines = {
        i: min([int(t['rect'][1]) + t['rect'][3])
                for t in data if (t['block_num'], t['par_num'], t['line_num']) == i])
        for i in set([(t['block_num'], t['par_num'], t['line_num']) for t in data])
    }
    tops = {
        i: max([t['rect'][1]
                for t in data if (t['block_num'], t['par_num'], t['line_num']) == i])
        for i in set([(t['block_num'], t['par_num'], t['line_num']) for t in data])
    }
    baselines_keys = sorted(baselines.keys())
    tops_keys = sorted(tops.keys())
    box_h = min(tops[tops_keys[i]] - baselines[baselines_keys[i - 1]])
    for i in range(1, len(baselines)) if tops_keys[i][0] == baselines_keys[i - 1][0]
    ) if len(baselines) > 1 else int(np.median(min([t['rect'][3] for t in data])))
    horizontal_kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (config['line_len'], 1))
    templ_img_bin_inv = cv2.threshold(templ_img, 0, 255,
                                       cv2.THRESH_BINARY_INV + cv2.THRESH_OTSU)[1]
    horizontal_lines = cv2.morphologyEx(templ_img_bin_inv,
                                       cv2.MORPH_OPEN, horizontal_kernel)
    for text in data:
        text['rect_under'] = [
            text['rect'][0],
            min(baselines[(text['block_num'], text['par_num'], text['line_num'])],
                text['rect'][1] + text['rect'][3]),
            text['rect'][2], box_h
        ]
        if not text['last_line']:
            else templ_img_bin_inv.shape[0] -
                baselines[(text['block_num'], text['par_num'], text['line_num'])] + 1
        ]
        (x, y, w, h) = text['rect_under']
        segments = [int(pos) for pos in np.linspace(x, x + w, num=config['match_seg_cnt'] + 1)]
        text['under_segments'] = [
            cv2.countNonZero(templ_img_bin_inv[y:y + h, segments[i]:segments[i + 1]])
            for i in range(len(segments) - 1)
        ]
        text['rect_under_lines'] = text['rect_under'].copy()
        text['rect_under_lines'][1] -= text['rect_under'][3] // 2
        text['rect_under_lines'][3] += text['rect_under'][3] // 2
        (x, y, w, h) = text['rect_under_lines']
        segments = [int(pos) for pos in np.linspace(x, x + w, num=config['match_seg_cnt'] + 1)]
        text['line_segments'] = [
            cv2.countNonZero(horizontal_lines[y:y + h, segments[i]:segments[i + 1]])
            for i in range(len(segments) - 1)
        ]
    set_loaded_templete(config, answer['TemplateName'])
    return templ_img, templ_img_bin_inv, data

```

Псеудо код 17. Дохватање елемената шаблона

Наравно, може се поставити питање зашто би се уопште и утврђивали статуси подвучености речи на шаблонској слици, када шаблонска слика не би требало да садржи ниједну подвучену реч. Разлог је тај што се велики број слова протеже и испод основне линије текста, што додатно доприноси зацрњености испод основне линије текста. За свако питање ове класе и све речи које могу да се подвуку у њеној зони одговора биће утврђивани статуси подвучености речи, односно биће израчунаване ове информације на сличан начин. Након што се оне буде израчунале, ове информације ће бити упоређиване са раније израчунатим информацијама за сваку реч добијеним над шаблонском сликом зоне одговора овог питања.

Након што се дохвате подаци о шаблонској слици зоне одговора питања и речи које је могуће подвући у тој зони одговора, врши се одсецање дела слике странице попуњеног теста која представља зону одговора. Подаци о прецизном простирању зоне одговора су раније утврђени и поклапају се за зоном питања којој ова зона одговора припада. Стога, на располагању су две слике, при чему једна представља слику зоне одговора питања са шаблонског, непопуњеног, празног теста, а друга представља зону одговора истог тог питања са попуњеног теста.

Ове две слике је потребно ускладити, тако да буду поравнате. Разлог за ово поравнање је што прецизније утврђивање информација о статусу подвучености речи, при чему ће правоугаоници речи са шаблонске слике зоне одговора бити транслирани на слику зоне одговора са попуњеног теста. Поравнање две слике извршава се проналажењем кључних тачака методом заснованом на екстракцији атрибута користећи локалне инваријантне дескрипторе (енг. *keypoints extraction and feature extraction based on local invariant descriptors*).

Имајући у виду да су ове две слике веома сличне и да је нека од њих евентуално незнатно ротирана, за детекцију кључних тачака довољно је коришћење ORB (енг. *Oriented FAST and Rotated BRIEF*) алгоритма. Овај алгоритам представља спој FAST (енг. *Features from Accelerated Segment Test*) детектора кључних тачака и BRIEF (енг. *Binary Robust Independent Elementary Features*) дескриптора, модификованих у сврхе побољшања перформанси. ORB алгоритам користи FAST алгоритам за детекцију кључних тачака, а затим примењује Херисов детектор углова (енг. *Harris corner measure*) да би пронашао најбољих N кључних тачака.

Међутим, FAST алгоритам не израчунава ротацију, те је модификован тако да на основу интензитета пиксела прозора (енг. *patch*) са лоцираним углом у центру израчунава његово тежиште. Оријентација представља правац вектора од ове тачке угла до тежишта. За дескрипторе, ORB користи BRIEF дескрипторе, који пружају лошије перформансе у случају ротације. Стога ORB алгоритам усмерава BRIEF дескрипторе на основу оријентације раније пронађених кључних тачака. Псеудо код 18 илуструје поравнање две слике.

```
def get_aligned_image(config, answer_image, templ_img_bin_inv):
    test_img_bin_inv = cv2.threshold(test_img_gray, 0, 255, cv2.THRESH_BINARY_INV +
cv2.THRESH_OTSU)[1]
    orb_detector = cv2.ORB_create(1000)
    key_point1, des1 = orb_detector.detectAndCompute(test_img_bin_inv, None)
    key_point2, des2 = orb_detector.detectAndCompute(templ_img_bin_inv, None)
    matcher = cv2.BFMatcher(cv2.NORM_HAMMING, crossCheck=True)
    matches = list(matcher.match(des1, des2))
    matches.sort(key=lambda e: e.distance)
    matches = matches[:int(len(matches) * 0.9)]
    no_of_matches = len(matches)
    p1 = np.zeros((no_of_matches, 2))
    p2 = np.zeros((no_of_matches, 2))
    for i in range(len(matches)):
        p1[i, :] = key_point1[matches[i].queryIdx].pt
        p2[i, :] = key_point2[matches[i].trainIdx].pt
    homography, mask = cv2.findHomography(p1, p2, cv2.RANSAC)
    return cv2.warpPerspective(test_img_bin_inv, homography, (templ_img_bin_inv.shape[1],
templ_img_bin_inv.shape[0]))
```

Псеудо код 18. Поравнање тестне слике на основу шаблонске

Најпре је потребно извршити трансформацију слике одговора коришћењем операција као што су бинаризација слике, адаптивно трешхолдовање коришћењем Оцуовог алгоритма и инвертовање слике. Затим се врши конструкција детектора ORB алгоритма са максималним бројем кандидата за разматрање кључних тачака, који је одређен конфигурационом датотеком теста. Након тога се врши детекција кључних тачака и дескриптора над бинарном инвертованом сликом зоне одговора попуњеног теста, као и над шаблонском бинарном инвертованом сликом зоне одговора.

Затим се утврђују поклапања између дескриптора за ове две слике користећи методу грубе силе (енг. *Brute Force*). С обзиром да се користи ORB алгоритам наводи се Хамингова норма (енг. *Hamming Norm*) која даје најбоље резултате. Поред тога, наведена је и унакрсна провера тако да се добију парови најближих дескриптора (x, y) , при чему дескриптор x најбоље одговара дескриптору y и обрнуто. На овај начин обрада резултује у листи конзистентних парова дескриптора. Такође, овај начин утврђивања поклапања дескриптора обично производи најбоље резултате са минималним бројем примерака који одступају, у случају када постоји довољно парова поклапања.

Након тога се листа парова дескриптора уређује растуће према дистанци између два дескриптора. Што је дистанца између два дескриптора мања, то су сличнија два региона слика означена двома кључним тачкама. Од целе листе парова дескриптора задржава се само број парова одређен конфигурационом датотеком теста, при чему се задржавају парови вредности које више личе. У супротном, задржавањем свих парова постоји ризик од увођења шума.

На самом почетку овог процеса алоцира се простор за смештање x и y координата кључних тачака само за парове дескриптора који су преостали. Ово се врши итерирањем кроз листу преосталих парова и индицирањем да се кључне тачке са једне слике мапирају на другу и обрнуто. Затим се на основу израчунатих вредности кључних тачака врши одређивање хомографске матрице користећи *RANSAC* (енг. *RANdom SAmple Consensus*) алгоритам. На самом крају се слика зоне одговора са странице попуњеног теста поравнава према шаблонској слици региона одговора применом перспективних трансформација коришћењем раније израчунате хомографске матрице као трансформационе матрице.

Приликом одређивања статуса подвучености речи врши се сегментација слике испод речи одређена на један од два претходно описана начина. Затим се за сваки сегмент одређује број зацрњених пиксела који они садрже. Након тога се проналази разлика у попуњености између кореспондентних сегмената испод речи са ове слике и испод речи шаблонске слике.

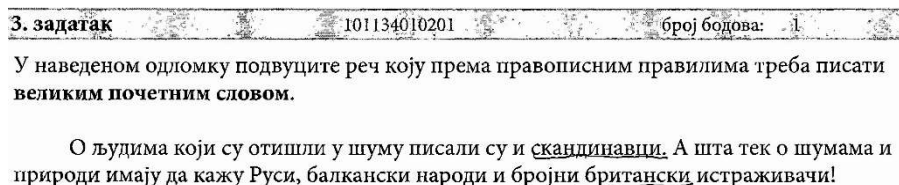
Разлика у попуњености одређује се у односу на дужину сегмента, односно у односу на дужину речи. Затим се за сваки сегмент проверава да ли је број црних пиксела већи од задатог трешхолда и, уколико то јесте случај, сегмент се означава као зацрњен. У случају да је више од половине броја сегмената зацрњено, реч се сматра подвученом, док се у супротном случају реч сматра неподвученом. Псеудо код 19 утврђује да ли је нека реч подвучена.


```
def get_word_status(under_word_image, word, thresh, mode):
    num = len(word[f'{mode}_segments'][i])
    (x, y, w, h) = word[f'rect{mode}']
    segments = [int(pos) for pos in np.linspace(x, x + w, num)]
    filled_segments = [
        cv2.countNonZero(under_word_image[y:y + h, segments[i]:segments[i + 1]])
        for i in range(len(segments) - 1)
    ]
    seg_len = w * 1. / (len(segments) - 1)
    diff = [np.round((filled_segments[i] - word[f'{mode}_segments'][i]) * 1. / seg_len, 2)
            if filled_segments[i] > word[f'{mode}_segments'][i] else 0
            for i in range(len(segments) - 1)]
    change = [diff >= thresh for d in diff]
    return change.count(True) > len(change) * 0.5
```

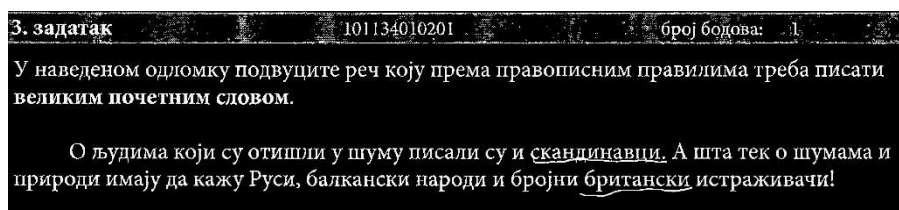
Псеудо код 19. Утврђивање подвучености речи

При утврђивању статуса подвучености речи довољно је да више од половине било којих сегмената буде зацрњено. Иако је могуће да се посматра да ли је више од половине узастопних сегмената зацрњено, односно да они представљају континуалну линију, утврђено је да то није увек случај у реалном сценарију. Одређени кандидати подвлаче реч испрекиданим линијама или линијама које нису праве целом својом дужином.

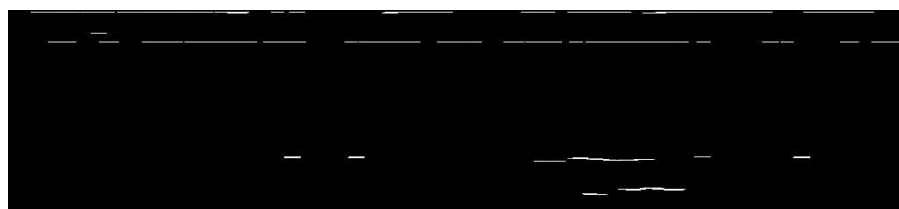
Слика 51 приказује питање класе повезивања са попуњеног теста у којем се подвлаче речи понуђеног текстуалног одговора. Слика 52 приказује бинарну инвертовану и поравнату слику зоне одговора (и питања) са попуњеног папирног теста, која служи за утврђивање зацрњености првом методом. Слика 53 приказује бинарну инвертовану и поравнату слику хоризонталних линија, која служи за утврђивање зацрњености другом методом. Слика 54 је аотирана резултатима добијеним у процесирању ове класе питања.



Слика 51. Зона питања и одговора са попуњеног теста



Слика 52. Слика за утврђивање подвучености речи првом методом



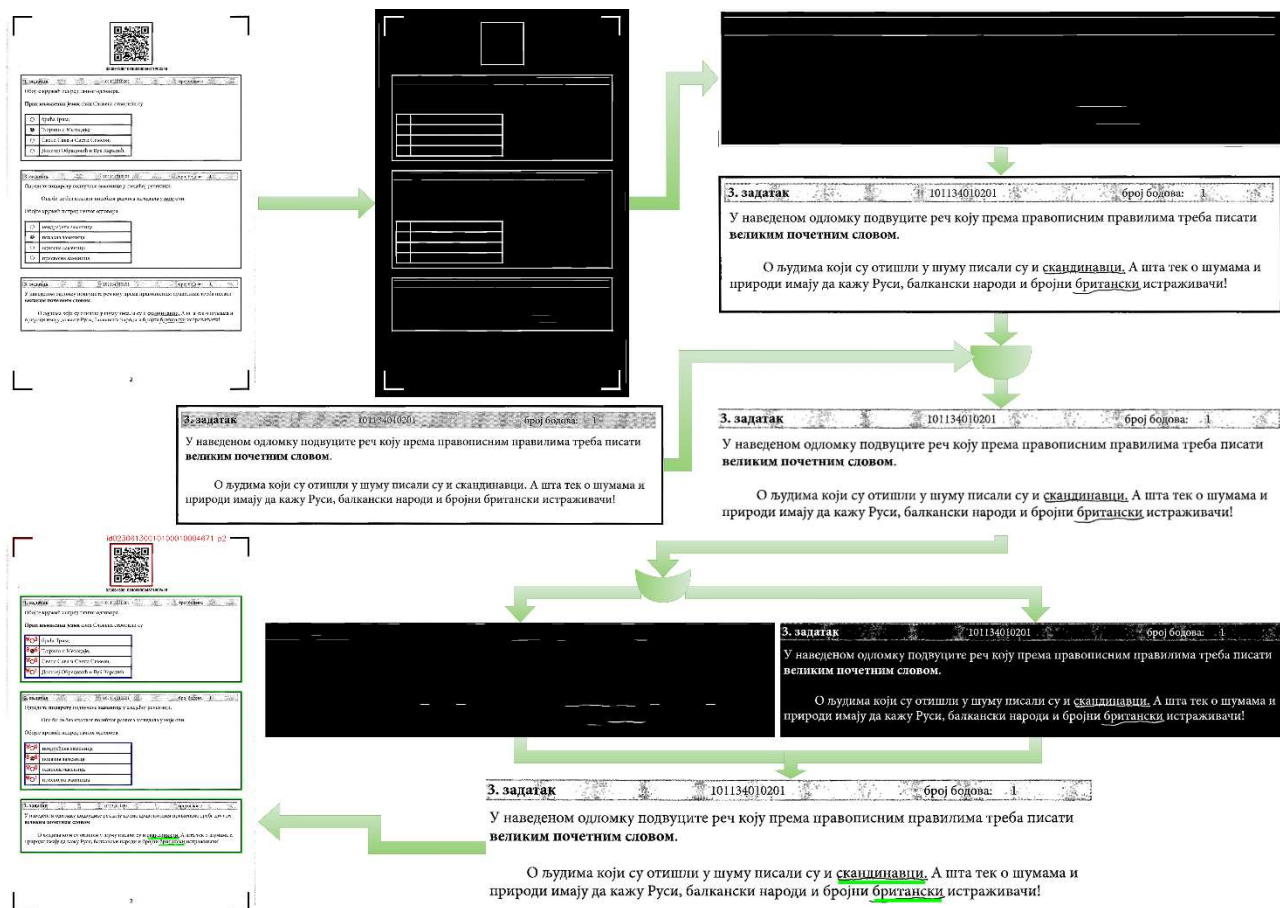
Слика 53. Слика за утврђивање подвучености речи другом методом

У наведеном одломку подвуците реч коју према правописним правилима треба писати великим почетним словом.

О људима који су отишли у шуму писали су и скандинавци. А шта тек о шумама и природи имају да кажу Руси, балкански народи и бројни британски истраживачи!

Слика 54. Анотирано питање резултатима добијем у процесирању

Слика 55 приказује целокупан след корака за процесирање првог формата питања ове класе, укључујући странице теста или њихове сегменте који су настали у том процесу. Зелене стрелице означавају усмереност корака у овом процесу.



Слика 55. Кораци у процесирању првог формата питања ове класе

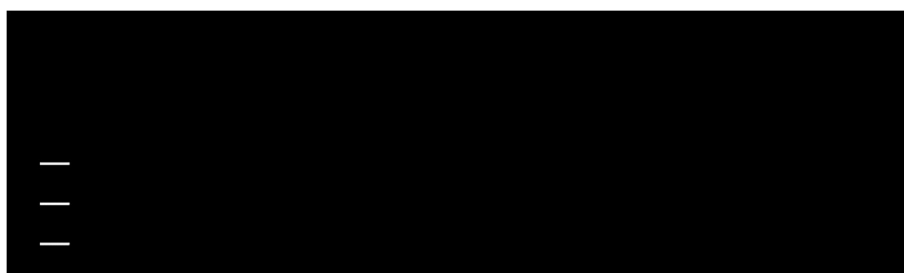
Други формат ове класе питања коју је предложени систем способан да процесира подразумева да су у питању наведени понуђени одговори у две подцелине. У једној подцелини се испред сваког понуђеног одговора налази симбол који треба дописати на једну од одговарајућих линија које се налазе испред понуђених одговора који чине другу подцелину. Друга варијанта овог питања је да постоји само једна подцелина са понуђеним одговорима испред којих се налазе линије за дописивање симбола, чиме се понуђени одговори уређују у поретку дописаних симбола. Пример прве варијанте другог формата питања ове класе илуструје Слика 56.

6. задатак		101211010701	број бодова: 2
Испред функције језика напишите број који стоји испред одговарајућег значења.			
Функција језика:		Значење:	
<u>4</u>	Акумулативна функција	1. Језик је средство споразумевања међу људима.	
<u>3</u>	Експресивна функција	2. Језик пружа могућност измештања у простору и времену.	
<u>1</u>	Комуникативна функција	3. Језик је средство којим се изражавају осећања.	
		4. Језик чува и преноси знања са једне генерације на другу.	

Слика 56. Прва варијанта другог формата питања ове класе

У овом случају се регион одговора зонира од стране зонер оператера у подсистему за зонирање дигиталног теста. Регион одговора налази се у оквиру региона питања којем припада, при чему су понуђени одговори са линијама за дописивање изабраног одговора обухваћени табелом. У фази процесирања одговора овог типа већ је пронађена и потврђена ова табела са одговорима.

Најпре се приступа издвајању слике табеле са странице теста која садржи тренутно процесирано питање овог типа. Затим се над тако добијеном сликом примењују операције као што су бинаризација, адаптивно трешхолдовање коришћењем Оцуовог алгорита и инверзија слике. Потом се приступа формирању правоугаоног структуралног елемента, чије су минималне и максималне димензије наведене конфигурационом датотеком теста, а којим ће се са добијене измењене слике табеле издвојити хоризонталне линије за дописивање изабраног одговора. Ово се врши применом битске операције искључиво ИЛИ (енг. *bitwise xor*) над сликама добијеним применом структуралног елемента минималне и максималне величине. С обзиром да је слика инвертована приликом њене трансформације, примењује се операција отварања. Слика 57 илуструје преостале хоризонталне линије за упис одговора.



Слика 57. Хоризонталне линије за упис одговора

Затим се претражују контуре над добијеном сликом хоризонталних линија. Из добијених контура се издвајају линије за дописивање симбола одговора на основу података из конфигурационе датотеке теста, раније утврђене табеле одговора, информација о самој зони одговора на ово питање, као и релација између контура које су записане у њиховој хијерархији. Када се ове линије за упис одговора пронађу и потврде, оне се уређују тако да се касније обрађују редом одозго на доле.

Потом се дохватљају ћелије табеле одговора у којима се налазе раније потврђене линије за дописивање одговора. Број добијених ћелија треба да буде једнак броју линија за упис одговора, односно треба да одговара информацији записаној у конфигурационој датотеци теста. Такође, добијене ћелије треба да се налазе у истој колони табеле са понуђеним одговорима.

Након тога се врши елиминација линија за упис одговора са инвертоване слике зоне одговора. Најпре се врши копија слике зоне одговора. Затим се на ову слику црним пикселима утискују испуњене линије контура које представљају линије за упис одговора. Овим се линије за упис одговора ефективно бришу са копије инвертоване слике зоне одговора.

Потом се ова слика морфолошки трансформише тако да се исправе ефекти брисања линија за упис одговора, уколико су одговори уписани директно на линији или ако линија пролази кроз њих. Ово се постиже морфолошким операцијама отварања и затварања над сликом, користећи информације о локацијама линија за упис одговора. Ефекте ове операције илуструје Слика 58.

6. задатак		101211010701	број бодова: 2
Испред функције језика напишите број који стоји испред одговарајућег значења.			
Функција језика:		Значење:	
4	Акумулативна функција	1. Језик је средство споразумевања међу људима.	
3	Експресивна функција	2. Језик пружа могућност измештања у простору и времену.	
1	Комуникативна функција	3. Језик је средство којим се изражавају осећања.	
		4. Језик чува и преноси знања са једне генерације на другу.	

Слика 58. Брисање хоризонталних линија за упис одговора и опоравак слике

За предикцију руком написаних симбола на линији одговора којима се ефективно спајају појмови из две подцелине користи се претходно тренирани модел конволуционе неуралне мреже. О овом моделу ће бити речи у наредном потпоглављу, с обзиром да се модел користи за предикцију секвенце записаних симбола. Овај модел се у случају првог коришћења учитава у радну меморију, а затим се сваки наредни пут дохвата, када за то постоји потреба.

Након учитавања модела, за сваку ћелију која садржи линију за упис одговора врши се одсецање дела слике који је ограничен том ћелијом, са копије инвертоване слике одговора са које су ефективно обрисане линије. Кроз ове ћелије табеле једне колоне редом се пролази одозго на доле. Добијене слике се трансформишу тако да одговарају димензијама које модел захтева. Затим се, користећи претходно учитани модел, врши предикција руком написаног симбола. Добијени резултати за сваку ћелију табеле са линијом за упис одговора смештају се у листу резултата. Слика 59 приказује питање ове класе аотирано резултатима предикције.

6. задатак		101211010701	број бодова: 2
Испред функције језика напишите број који стоји испред одговарајућег значења.			
Функција језика:		Значење:	
4	Акумулативна функција	1. Језик је средство споразумевања међу људима.	
3	Експресивна функција	2. Језик пружа могућност измештања у простору и времену.	
1	Комуникативна функција	3. Језик је средство којим се изражавају осећања.	
		4. Језик чува и преноси знања са једне генерације на другу.	

Слика 59. Питање аотирано резултатима процесирања

У случају успешне обраде, добијени резултати се враћају као резултат ове функције. У супротном случају, сигнализира се изузетак и логује се грешка. Псеудо код 20 представља ову обраду.

```

def get_matches(config, page_lines, page, answer, logger):
    try:
        x, y, w, h = answer['X'], answer['Y'], answer['Width'], answer['Height']
        answer_image = page[y:y + h, x:x + w]
        answer_image_bin_inv = cv2.threshold(answer_image, 0, 255,
                                              cv2.THRESH_BINARY_INV + cv2.THRESH_OTSU)[1]
        horizontal_kernel = cv2.getStructuringElement(cv2.MORPH_RECT,
                                                       (config['match_line_len_min'], 1))
        horizontal_lines_min = cv2.morphologyEx(answer_image_bin_inv,
                                                  cv2.MORPH_OPEN, horizontal_kernel)
        horizontal_kernel = cv2.getStructuringElement(cv2.MORPH_RECT,
                                                       (config['match_line_len_max'], 1))
        horizontal_lines_max = cv2.morphologyEx(answer_image_bin_inv,
                                                  cv2.MORPH_OPEN, horizontal_kernel)
        horizontal_lines = cv2.bitwise_xor(horizontal_lines_min, horizontal_lines_max)
        contours, hierarchy = cv2.findContours(horizontal_lines,
                                                cv2.RETR_TREE,
                                                cv2.CHAIN_APPROX_SIMPLE)

        table_cells = get_table_cells(page_lines, answer)
        answer_lines = confirm_answer_lines(contours, hierarchy, config, table_cells, answer)
        answer_lines = sorted(answer_lines,
                               key=lambda line: cv2.bounding_rect(line),
                               reverse=False)

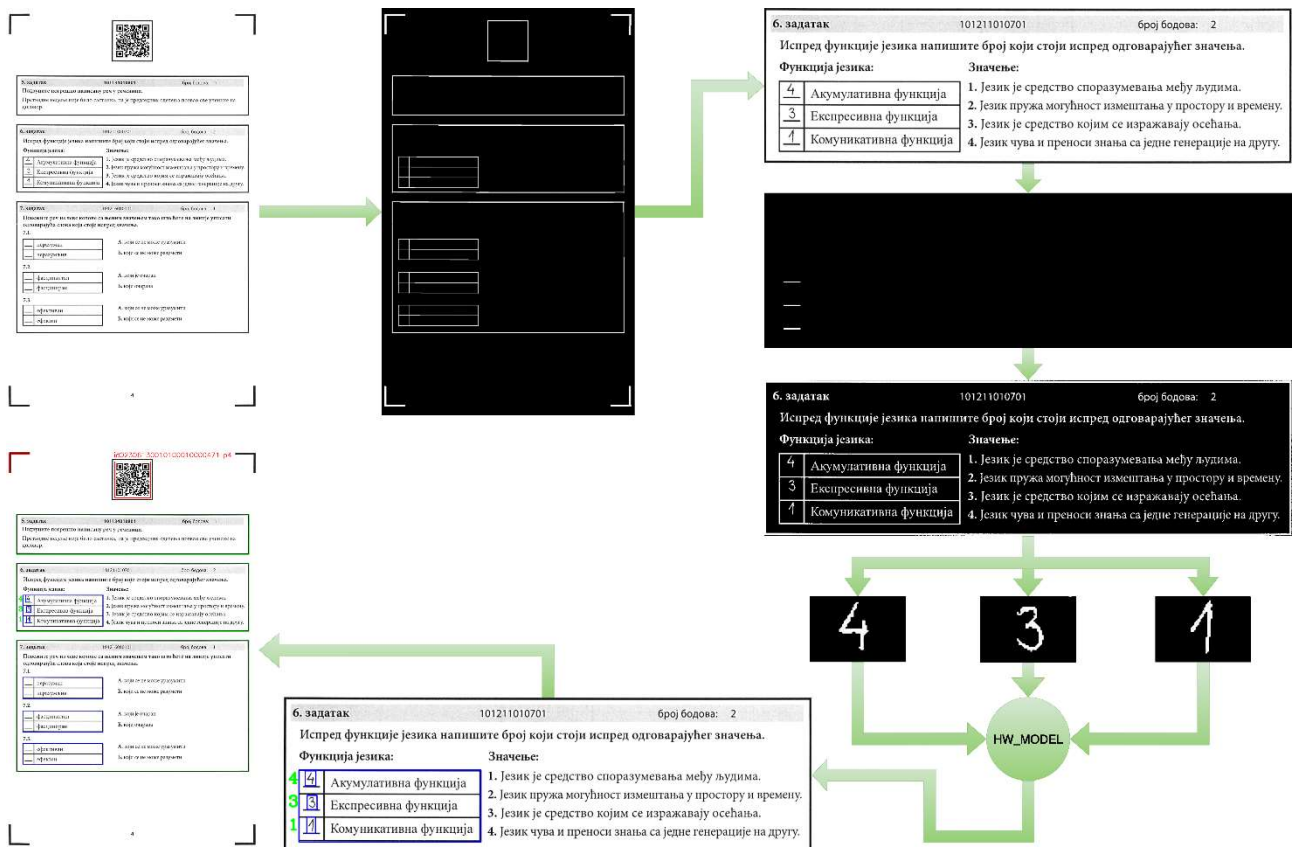
        lines_parent_cells = get_parent_cells(answer_lines, table_cells)
        morphed_image = answer_image_bin_inv.copy()
        for line in answer_lines:
            cv2.draw(answer_image_bin_inv, [line], 0, (0), -1)
        morphed_image = repair_image(morphed_image, answer_lines)
        hw_model = get_or_load_model()
        result = []
        for cell in lines_parent_cells:
            cell_rect = cv2.bounding_rect(cell)
            x, y, w, h = cell_rect
            cell_image = morphed_image[y:y + h, x:x + w]
            transformed_image = hw_model.transform_image(cell_image)
            symbol = hw_model.predict(transformed_image)
            result.append(str(symbol))

        return result
    except Exception as e:
        logger.log_error(e)
    return None

```

Псеудо код 20. Процесирање зона одговора питања ове класе

Слика 60 приказује целокупан след корака за процесирање другог формата питања ове класе, укључујући странице теста или њихове сегменте који су настали у том процесу. Зелене стрелице означавају усмереност корака у овом процесу.



Слика 60. Кораци у процесирању другог формата питања ове класе

4.2.6. Процесирање питања класе „Кратак одговор“

Предложени софтверски систем може да процесира питања ове класе која су дата у таквом формату да постоји текст прожет произвољним бројем линија за упис одговора на којима је потребно да кандидати упишу нумерички одговор са произвољним бројем симбола. Регион одговора зонира се од стране зонер оператера у подсистему за зонирање дигиталног теста и одговара региону питања. Такође, зонирањем се добијају и информације о појединачним просторима са линијама за упис одговора.

На самом почетку се врши издвајање целокупне зоне одговора која одговара зони питања тренутно процесираног питања овог типа. Потом се за тако издвојену слику врше морфолошке трансформације које укључују операције као што су бинаризација, адаптивно трешхолдовање коришћењем Оцуовог алгорита и инверзија слике. Затим се врши креирање правоугаоног линијског структуралног елемента, чије су димензије наведене конфигурационом датотеком теста. Пример питања ове класе илуструје Слика 61.

3. задатак

У складишту је 3 t робе, при чему је 12,5% неисправно.

а) Колика је маса неисправне робе, у килограмима?

Маса неисправне робе је 375 kg.

б) Од укупне масе неисправне робе 76% може да се рециклира. Колико килограма неисправне робе може да се рециклира?

Може да се рециклира 285 kg неисправне робе.

в) Цена исправне робе на тржишту износи 120 динара по килограму. За један килограм рециклиране неисправне робе добије се 45 динара. Ако би се продала сва исправна роба и рециклирала сва роба која се може рециклирати, колико би укупно динара добили?

Добили би укупно 327825 динара.

Слика 61. Пример питања ове класе

Имајући у виду да је раније добијена слика региона одговора инвертована, примењује се конволуциона операција над сликом коришћењем овог структуралног елемента. Она представља операцију отварања и њоме ће на слици зоне одговора преостати само хоризонталне линије. Међу овим линијама налазе се и линије за упис руком написаног одговора. Слика 62 приказује резултат добијен морфолошким трансформацијама слике питања тако да преостану само хоризонталне линије.



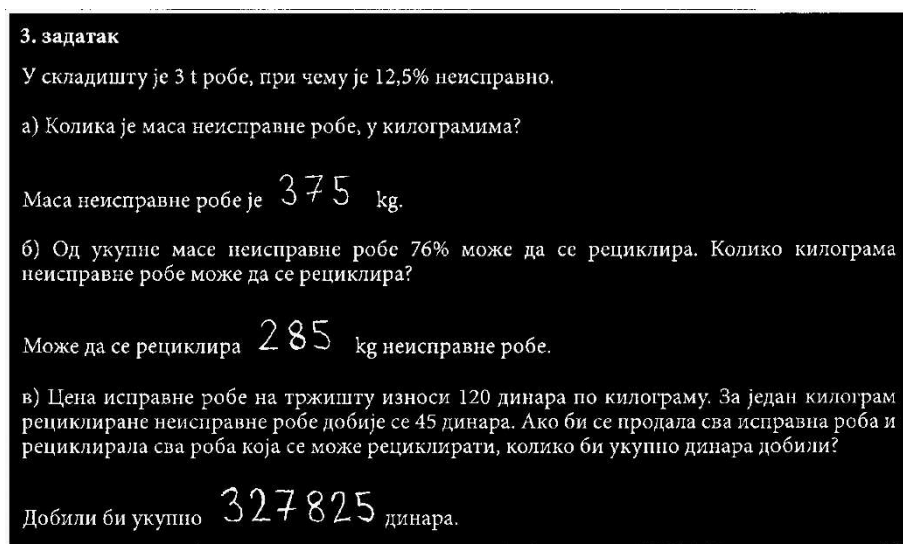
Слика 62. Хоризонталне линије за упис одговора

Након тога се врши претраживање контура линија за упис одговора над сликом хоризонталних линија. Након добијања кандидата хоризонталних линија, врши се укрштање ових информација са информацијама добијеним од подсистема за зонирање дигиталног теста, које се налазе у конфигурационој датотеци теста. Додатно, као испомоћ се користе и информације о релацијама између идентификованих контура записаних у подацима о хијерархији контура. Број резултујућих линија за упис одговора треба да одговара информацијама које се налазе у конфигурационој датотеци теста.

Затим се врши пречишћавање инвертоване слике зоне одговора од линија за упис одговора. Ово се не ради над оригиналном сликом, већ се прави копија инвертоване слике зоне одговора. Применом операција маскирања, над овом копираном сликом врши се утискивање

испуњених линија контура које представљају линије за упис одговора и то коришћењем црних пиксела.

На овај начин се практично са копије инвертоване слике зоне одговора уклањају линије за упис одговора. Након тога се коришћењем морфолошких трансформација слике операцијама отварања и затварања врши репарација штете настале над уписаним одговорима у процесу брисања линија за упис одговора. Ова оштећења уписаних одговора могу настати, уколико су кандидати одговоре уписивали тако да се линије за упис одговора секу са уписаним одговорима. Ефекте морфолошких операција примењених над копијом инвертоване слике зоне одговора илуструје Слика 63.



Слика 63. Брисање хоризонталних линија за упис одговора и опоравак слике

Након тога, врши се дохватање раније тренираног модела за предикцију руком написаног нумеричког одговора. Предикциони модел се учитава само једном, приликом првог захтева за њим, а затим се сваки наредни пут само дохвата по потреби. Овај модел је реализован коришћењем техника дубоких неуралних мрежа.

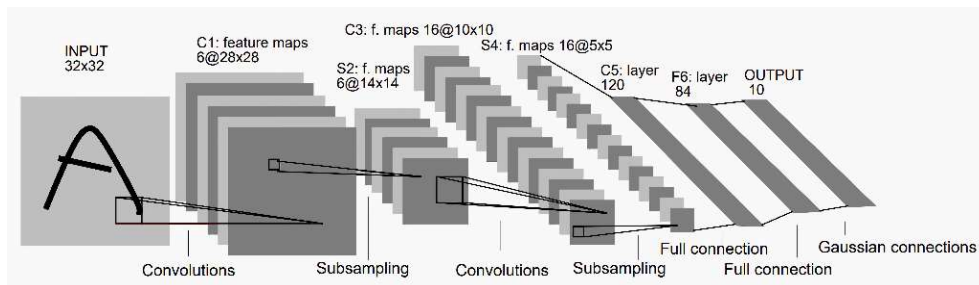
Затим се за сваку линију за упис одговора врши одсецање дела са копије бинарне инвертоване слике одговора, са које су практично обрисане линије одговора, која одговара тој линији. Део који се одсеца одређује се користећи податке о локацији и простирању идентификоване линије за упис одговора и податке из конфигурационе датотеке теста. На таквом делу слике врши се претрага контура како би се добили потенцијални кандидати симбола уписаног одговора.

С обзиром да неки кандидати одређене симболе пишу из више делова, врши се њихово груписање у једну контуру симбола. Ово се обавља на основу података о хијерархијском односу између контура, њиховим локацијама и дистанцама између њих. Издвојене кандидат контуре се уређују са лева на десно да би се обрађивале у поретку писања симбола од стране кандидата.

За сваку идентификовану контуру симбола исеца се слика ограничавајућег правоугаоника око тог симбола. Затим се она трансформише тако да одговара димензијама слике које модел очекује. Користећи претходно учитани модел, врши се предикција руком написаног симбола. У сврхе предикције руком написане секвенце симбола користе се технике дубоког учења.

Уместо да се креира потпуно нови модел неуралне мреже [60], користи се модификована верзија конволуционе неуралне мреже LeNeT-5 [61]. Оригинална верзија ове

конволуционе неуралне мреже је релативно једноставне структуре и састоји се од конволуционог слоја кога прати слој у коме се примењује максимално изабрање (енг. *max-pooling*), при чему овај пар слојева следи још једанпут. Након тога се настављају два потпуно повезана слоја са 10 неурона на излазу из последњег слоја који користи *Softmax* активациону функцију, како би се извршила предикција 10 различитих симбола. Архитектуру оригиналне LeNet-5 [62] мреже илуструје Слика 64.



Слика 64. Архитектура оригиналне LeNet-5 мреже

Иако овај модел пружа прилично добре резултате, он испољава одређен ниво грешке над тренинг (енг. *bias*) и значајну грешку над тест подацима (енг. *variance*). Неке од ових грешака се могу елиминисати тако што ће се направити модел који није недовољно обучен (како би се елиминисала грешка над тренинг подацима) и који није преобучен (како би се елиминисала грешка над тест подацима). Стога, примењене су погодне оптимизације како би се разрешили ови проблеми [63] [64].

Наравно, неки подаци над којима ће се вршити тренирање модела су сувише изобличени да би се и голим оком тешко могло утврдити о ком симболу се ради и то је инхерентна грешка која ће се јавити и коју није могуће избећи. Смањивање грешке варијансе може се постићи увећавањем скупа [65] (енг. *data augmentation*) тренинг података применом различитих трансформационих операција над сликама као што су ротација за не више од 10 степени, увећавање и умањивање слике за не више од 10 процената и транслирање слике у једну од 4 стране – горе, десно, доле и лево, за не више од 10 посто ширине, односно висине слике, у зависности од смера померања. Поред тога, применом L2 регуларизације у одређеним конволуционим слојевима врши се додатно значајније пенализовање функције грешке. Ово ће за последицу имати смањивање тежина које улазе у неуроне и на тај начин ће модел бити донекле једноставнији, како би се избегло преобучавање.

Додатно, коришћењем техника одсецања мреже (енг. *network pruning*), као и техником регуларизације којом се током тренирања врши одбацивање значајнијег процента неурона мреже на случајан начин (енг. *dropout regularization*), модел ће бити у стању да боље генерализује. Примена беч нормализације над активацијама неурона у слојевима мреже (енг. *batch normalization*) довешће до стабилизације мреже, омогућиће слојевима мреже да независно уче у већој мери и обезбедиће бржу конвергенцију у процесу обучавања мреже. Поред тога, употребом варијабилног корака учења (енг. *variable learning rate*), који се динамички прилагођава, омогућава се бржа конвергенција у процесу учења модела, избегава се стагнирање у процесу учења и заробљавање у неком од локалних минимума, а врши се приближавање глобалном минимуму функције грешке.

Смањивање грешке у тренирању модела обезбеђује се додавањем нових конволуционих слојева, као и увећањем броја филтера у конволуционим слојевима [66]. Последично, врши се и увећање броја скривених слојева како би се омогућило прихватање већег обима улазних података због повећања насталог у конволуционим слојевима. Прављењем гушће и дубље мреже са више скривених слојева формира се модел који ће бити способан да боље уочава

комплекснија и значајнија својства на сликама руком написаних симбола чију предикцију треба извршити.

Модел дубоких неуралних мрежа често су комплексни у погледу меморије потребне за чување параметара мреже као што су тежине, појачања (енг. *bias*), активације неурона, који се обично чувају као тридесетдвобитни реални бројеви, што обезбеђује висок ниво прецизности у израчунавањима, а последично високу тачност мреже. Међутим, да би се смањили меморијски захтеви модела, као и трошкови који неминовно настају у процесу различитих израчунавања коришћењем параметара, искоришћена је техника квантизације параметара [67]. Ово се врши тако што се параметри чувају користећи типове података ниже прецизности (нпр. 8-битни цели бројеви), уместо досадашњих типова података више прецизности (нпр. 32-битни цели бројеви) [68].

На овај начин се смањује комплексност модела, који захтева мање меморије, док се и матричне операције могу извршити много брже користећи аритметику над целим бројевима. Квантизација [69] се користи током тренирања неуралне мреже (енг. *quantization-aware training*), тако што се користе квантификоване вредности у фази прослеђивања унапред. На тај начин ће се грешке везане за квантизацију акумулирати у тоталну грешку током тренирања, док ће укупна резултујућа грешка бити мања него да је квантизација извршена након тренирања [70] (енг. *post-training quantization*).

Модификована конволуциона неурална мрежа састоји се од два конволуциона слоја које прате два слоја у којима се примењује максимално изабирање и који користе ReLU (енг. *Rectified Linear Unit*) активацију. Након тога следе још два конволуциона слоја и још један слој у којима се примењује максимално изабирање са ReLU активацијом. Затим следе три потпуно повезана слоја, а на самом крају и излазни слој који садржи 10 неурона, који користе *Softmax* активациону функцију, за предикцију 10 различитих симбола.

Беч нормализација се спроводи након свака два конволуциона слоја, као и након сваког од потпуно повезаних слојева. Регуларизација се примењује након сваког слоја изабирања и након последњег потпуно повезаног слоја. У првом конволуционом слоју примењује се L2 регуларизација.

Модел конволуционе неуралне мреже трениран је кроз 30 епоха користећи варијабилан корак учења над подацима формираним на основу слика руком написаних цифара из MNIST [71] скупа података. Овај скуп података се састоји од 60 000 тренинг примерака и 10 000 тест примерака. Аугментацијом података тренинг скуп је увећан дупло. С обзиром да слике из овог скупа података представљају ништа друго него матрице пиксела различитог интензитета, могуће је спровести нормализацију вредности интензитета пиксела, која ће убрзати процес тренирања модела. Нормализација се спроводи тако што се за сваки пиксел од вредности његовог интензитета одузме средња вредност интензитета свих пиксела свих слика скупа података, а затим се тако добијени резултат подели вредношћу стандардне девијације свих пиксела свих слика из скупа података.

Резултати добијени предикцијом симбола са једне линије за упис одговора спајају се у један текст. За све линије за упис одговора које постоје у оквиру једне зоне одговора питања појединачни резултати смештају се у листу парова резултата, при чему појединачне парове чине подаци о линији за упис одговора и текст уписан на тој линији одговора. Пример слике питања са попуњеног теста аотиране резултатима добијеним у процесирању ове класе питања илуструје Слика 65.

3. задатак

У складишту је 3 т робе, при чему је 12,5% неисправно.

а) Колика је маса неисправне робе, у килограмима?

375

Маса неисправне робе је 375 kg.

б) Од укупне масе неисправне робе 76% може да се рециклира. Колико килограма неисправне робе може да се рециклира?

285

Може да се рециклира 285 kg неисправне робе.

в) Цена исправне робе на тржишту износи 120 динара по килограму. За један килограм рециклиране неисправне робе добије се 45 динара. Ако би се продала сва исправна роба и рециклирала сва роба која се може рециклирати, колико би укупно динара добили?

327825

Добили би укупно 327825 динара.

Слика 65. Питање анотирано резултатима процесирања

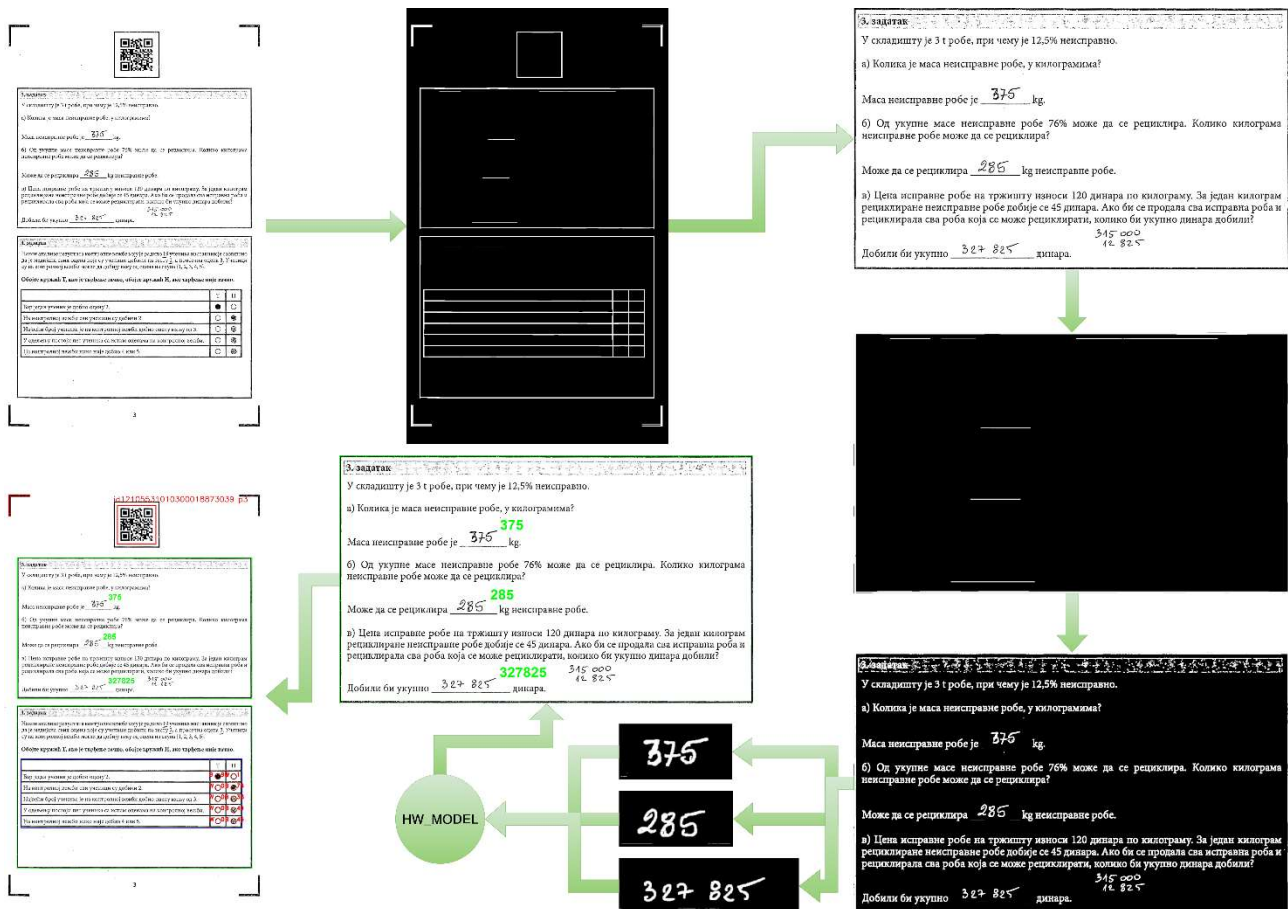
У случају успешне обраде, формирана листа резултата се враћа као резултат ове функције. У супротном случају, сигнализира се изузетак и логује се грешка. Псеудо код 21 представља ову обраду.

```
def get_short_written(page, answer, test_logger):
    try:
        x, y, w, h = answer['X'], answer['Y'], answer['Width'], answer['Height']
        answer_image = page[y:y + h, x:x + w]
        answer_image_bin_inv = cv2.threshold(answer_image, 0, 255,
                                              cv2.THRESH_BINARY_INV + cv2.THRESH_OTSU)[1]
        horizontal_kernel = cv2.getStructuringElement(cv2.MORPH_RECT,
                                                       (config['match_line_len'], 1))
        horizontal_lines = cv2.morphologyEx(answer_image_bin_inv,
                                              cv2.MORPH_OPEN, horizontal_kernel)
        contours, hierarchy = cv2.findContours(horizontal_lines,
                                              cv2.RETR_TREE,
                                              cv2.CHAIN_APPROX_SIMPLE)
        answer_lines = filter_lines_contours(config, contours, hierarchy)
        morphed_image = answer_image_bin_inv.copy()
        for line in answer_lines:
            cv2.draw(answer_image_bin_inv, [line], 0, (0), -1)
        morphed_image = repair_image(morphed_image, answer_lines)
        hw_model = get_or_load_model()
        result = []
        for line in answer_lines:
            line_rect = get_bounding_rect(config, line)
            x, y, w, h = line_rect
            line_answer_image = morphed_image[y:y + h, x:x + w]
            contours, hierarchy = cv2.findContours(line_answer_image,
                                                    cv2.RETR_TREE,
                                                    cv2.CHAIN_APPROX_SIMPLE)
            symbols_contours = obtain_symbols_contours(contours, hierarchy)
            symbols_contours = sorted(symbols_contours,
                                      key=lambda c: cv2.bounding_rect(c)[0],
                                      reverse=False)

            symbols = []
            for c in symbols_contours:
                symbol_rect = cv2.bounding_rect(c)
                sx, sy, sw, sh = symbol_rect
                symbol_image = morphed_image[sy:sy + sh, sx:sx + sw]
                transformed_image = hw_model.transform_image(symbol_image)
                sym = hw_model.predict(transformed_image)
                symbols.append(sym)
            result.append((line, ''.join([str(s) for s in symbols])))
        return result
    except Exception as e:
        logger.log_error(e)
    return None
```

Псеудо код 21. Процесирање зона одговора питања ове класе

Слика 66 приказује целокупан след корака за процесирање питања ове класе, укључујући странице теста или њихове сегменте који су настали у том процесу. Зелене стрелице означавају усмереност корака у овом процесу.



Слика 66. Кораци у процесирању питања ове класе

5. Евалуација имплементираних система

Систем је тестиран над 45 785 папирних тестова које су попуњавали ученици средњих школа широм Републике Србије. Тест се састоји од укупно 16 страница, при чему је прва страница насловна, док је последња страница празна. Укупан број питања на тесту износи 22. Тест садржи 17 питања класе вишеструки избор, 4 питања класе повезивања и 1 питање типа кратак нумерички одговор. Питања су дистрибуирана у оквиру страница теста и не постоје додатни папири за упис одговора, већ се одговори налазе у оквиру региона питања. Региони једног питања могу да се протежу на више узастопних страница, али се воде као засебни региони тог питања.

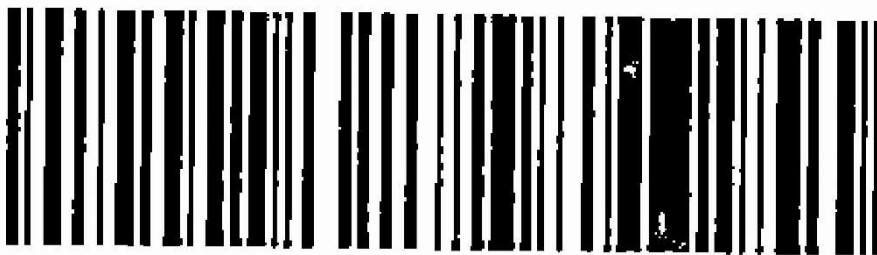
Насловна страница теста не садржи ниједно питање, већ наводи опште информације о тесту и упутство за његово попуњавање. Ова страница је и идентификациона страница теста и кандидата. Последња страница теста је намерно остављена празна и служи као додатни простор кандидату за скицање концепта. Ова страница не подлеже процесу аутоматизованог оцењивања. У сваком од наредних потпоглавља биће наведени резултати добијени у процесу препознавања релевантних информација на тесту.

У процесу тестирања имплементираних система коришћена је машина која поседује 64 гигабајта радне меморије и Интелов процесор *i9-9900 @3.1GHz* са 8 физичких и 16 виртуелних језгара.

5.1. Евалуација препознавања једнодимензионалног бар-кода

Једнодимензионални идентификациони бар-код једнозначно одређује кандидата и налази се на налепници која се лепи од стране дежурног на насловну страницу теста. С обзиром да је целокупни скуп података садржао 45 785 папирних тестова, укупно је очекивано 45 785 једнодимензионалних бар-кодова. Систем је успешно детектовао и декодовао 45 626 једнодимензионалних бар-кодова.

Од преосталих 159 тестова, на укупно 158 тестова дежурни нису залепили налепнице и систем је исправно детектовао да је то случај. На 1 тесту није уопште детектовано постојање једнодимензионалног бар-кода, иако је био залепљен и у исправном формату. Стога је, у случају када је једнодимензионални идентификациони бар-код присутан на насловној страници, успешност у препознавању овог бар-кода изузетно висока и износи 99.99 %. Просечно време потребно за препознавање једнодимензионалног бар-кода износи само 30 мс. Пример непрепознатог једнодимензионалног бар-кода илуструје Слика 67.



Слика 67. Непрепознати једнодимензионални бар-код

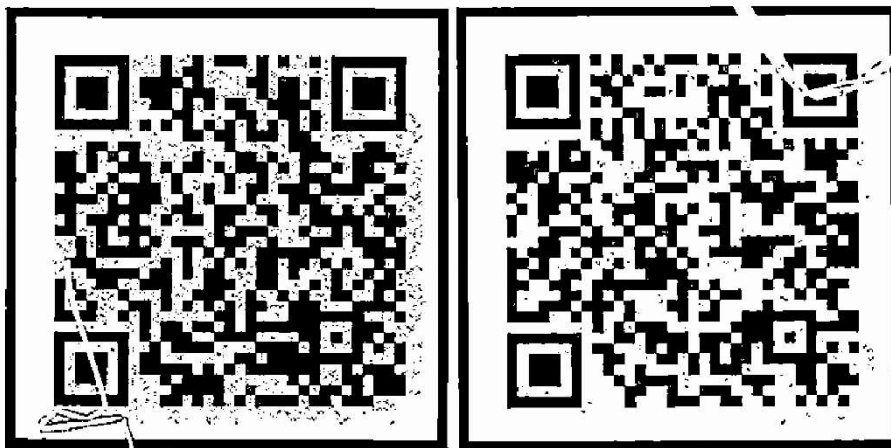
5.2. Евалуација препознавања дводимензионалног бар-кода

Странични дводимензионални бар-код налази се на свакој страници теста и садржи информације о јединственом идентификационом броју теста и броју странице на којој се налази. С обзиром да је целокупни скуп података садржао 45 785 папирних тестова, а да сваки тест садржи 16 страна, укупно је очекивано 732 560 дводимензионалних бар-кодова. Међутим,

утврђено је да је на 117 тестова недостајала барем једна страница, односно да је изгубљена или заборављена негде у процесу од фазе спровођења тестирања кандидата до фазе дигитализације папирног теста.

Укупно је недостајало 149 страница теста, те је очекивани број дводимензионалних бар-кодова једнак 732 411. Систем је успешно детектовао и декодовао 732 403 страничних дводимензионалних бар-кодова. На по једној страници на 8 тестова систем није уопште детектовао постојање дводимензионалног бар-кода.

Стога је успешност у препознавању овог бар-кода изузетно висока и износи 99.99 %. Просечно време потребно за препознавање дводимензионалног бар-кода износи само 20 мс. Пример непознатих дводимензионалних бар-кодова илуструје Слика 68.



Слика 68. Непрепознати дводимензионални бар-кодови

5.3. Евалуација процесирања питања класе „Вишеструки избор“

С обзиром да је тест садржао 17 питања ове класе, а да је целокупни скуп података садржао 45 785 папирних тестова, укупно је очекивано 778 345 питања ове класе. Свако питање ове класе на тесту је садржало тачно један регион одговора, те је укупан очекивани број региона одговора, такође 778 345. Од овог броја региона одговора успешно је детектовано 778 226 региона одговора, што представља успешност од 99.98 %.

Ручним увидом је утврђено да је неуспешној детекцији региона одговора допринео превелики шум настао у процесу од фазе тестирања до фазе дигитализације, те није дошло до успешног потврђивања пронађених региона питања и одговора са информацијама добијеним из конфигурационе датотеке теста. Од укупног броја успешно детектованих региона одговора, успешно је потврђено 777 539 решетки кругова понуђених одговора, што представља успешност од 99.91 %. То значи да за 687 успешно детектованих региона одговора нису успешно потврђене решетки кругова понуђених одговора.

Ручном инспекцијом утврђено је да су разлози за неуспешну детекцију решетки кругова понуђених одговора углавном последица начина попуњавања одговора од стране кандидата, али и утискивање црних линија у процесу дигитализације попуњеног теста који су ометали детекцију контура. Неки кандидати су прецртавали кругове који представљају понуђене одговоре, неки су их заокруживали, неки су формирали описане попуњене квадрате око кругова, док су неки вршили уписивање симбола Х преко кругова. Пример попуњавања одговора од стране кандидата, чији начин попуњавања није у складу са датим упутством, илуструје Слика 69.

7. задатак	101212010101	број бодова: 1
Ако је тврдња тачна, обојте кружић у колони Тачно , а ако тврдња није тачна, обојте кружић у колони Нетачно .		
Тврдња	Тачно	Нетачно
Вук је слово ј узео из старих рукописа.	<input type="radio"/>	<input checked="" type="radio"/>
Вук је објавио три речника српског језика.	<input type="radio"/>	<input checked="" type="radio"/>
Пре Вука ћирилицу је реформисао Саво Мркаљ.	<input checked="" type="radio"/>	<input type="radio"/>
Вук је говорио источнохерцеговачким дијалектом.	<input checked="" type="radio"/>	<input type="radio"/>

7. задатак	101212010101	број бодова: 1
Ако је тврдња тачна, обојте кружић у колони Тачно , а ако тврдња није тачна, обојте кружић у колони Нетачно .		
Тврдња	Тачно	Нетачно
Вук је слово ј узео из старих рукописа.	<input type="radio"/>	<input checked="" type="radio"/>
Вук је објавио три речника српског језика.	<input checked="" type="radio"/>	<input type="radio"/>
Пре Вука ћирилицу је реформисао Саво Мркаљ.	<input checked="" type="radio"/>	<input type="radio"/>
Вук је говорио источнохерцеговачким дијалектом.	<input checked="" type="radio"/>	<input type="radio"/>

7. задатак	101212010101	број бодова: 1
Ако је тврдња тачна, обојте кружић у колони Тачно , а ако тврдња није тачна, обојте кружић у колони Нетачно .		
Тврдња	Тачно	Нетачно
Вук је слово ј узео из старих рукописа.	<input type="radio"/>	<input checked="" type="radio"/>
Вук је објавио три речника српског језика.	<input checked="" type="radio"/>	<input type="radio"/>
Пре Вука ћирилицу је реформисао Саво Мркаљ.	<input type="radio"/>	<input checked="" type="radio"/>
Вук је говорио источнохерцеговачким дијалектом.	<input checked="" type="radio"/>	<input type="radio"/>

Слика 69. Попуњавање одговора од стране кандидата које није у складу са упутством

С обзиром да је успешно потврђено укупно 778 226 региона одговора, израчунато је да би максимални број детектованих кругова понуђених одговора требало да износи 6 584 989. Успешно је детектовано укупно 6 579 847 кругова, што представља успешност од 99.92 % у детекцији кругова понуђених одговора. Добијени проценат успешности одговара проценту успешности потврђивања решетки кругова понуђених одговора.

Ово чињеница сигнализира да детекција решетки кругова не успева јер не успева детекција скоро свих појединачних кругова из те решетки. У супротном, проценат успешности у детекцији кругова би био много већи. Овоме у прилог иде и чињеница да када кандидати невалидно попуњавају тест, онда се на погрешан начин означавају сви понуђени одговори из решетки кругова зоне одговора ове класе питања.

За укупан број потврђених решетки кругова, а који износи 777 539, израчунат је укупан број кругова које оне садрже и он износи 6 579 177. Од тога је за 6 578 729 кругова утврђен статус зацрпљености, што представља успешност од 99.99 %. За утврђивање статуса зацрпљености преосталих 448 кругова неопходна је ручна провера. Просечно време потребно за обраду једне странице са питањима ове класе укључујући и анотацију износи приближно 220 мс.

5.4. Евалуација процесирања питања класе „Повезивање“

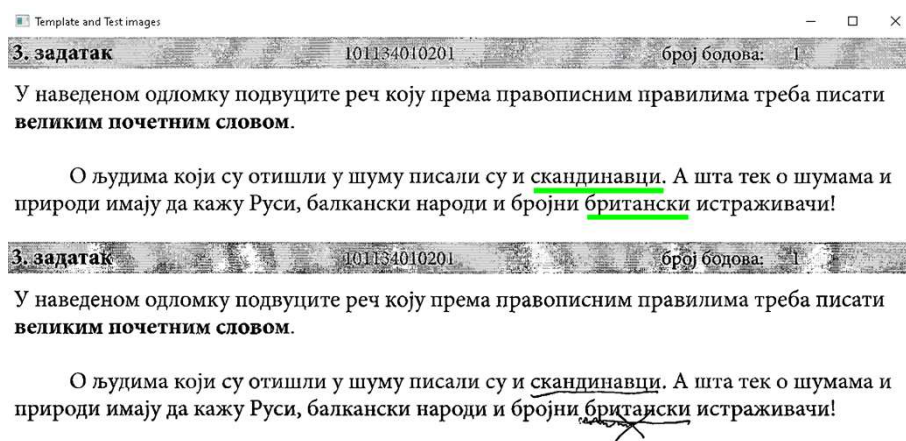
Тест је садржао 4 питања ове класе, при чему су два питања ове класе била у форми подвлачења речи у датом тексту, а два у форми уписивања арапске цифре да би се извршило повезивање одговора. Стога, имајући у виду да је целокупан скуп података садржао 45 785 папирних тестова, укупно је очекивано по 91 570 питања за обе варијанте питања ове класе. Код питања у форми подвлачења се регион одговора поклапа са регионом питања, док је код

питања у форми уписа симбола регион одговора представљен табелом коју је потребно потврдити, слично као и код питања типа вишеструки избор. Свако питање ове класе на тесту је садржало тачно један регион одговора, те укупан очекивани број региона одговора одговара броју региона питања, односно по 91 570.

Од овог броја питања успешно је детектовано 91 562 региона одговора варијанте подвлачења, што представља успешност од 99.99 %, док је у варијанти уписа симбола успешно детектовано 91 546 региона одговора, што представља успешност од 99.97 %. Ручним увидом је утврђено да је неуспешној детекцији региона одговора допринео превелики шум настао у процесу од фазе тестирања до фазе дигитализације, те није дошло до успешног потврђивања пронађених региона питања и одговора са информацијама добијеним из конфигурационе датотеке теста. Виши проценат код варијанте подвлачења оправдава се тиме што се регион одговора у том случају поклапа са регионом питања, док је у случају уписа симбола регион одговора представљен табелом коју треба потврдити.

Од укупно 91 562 успешно потврђених региона одговора варијанте подвлачења, сва подвлачења речи су успешно утврђена на 91 533 оваквих региона, што представља успешност од 99.97 %. У варијанти подвлачења, у једном питању је требало подвући 2 речи, док је у другом требало подвући 7 речи. Стога би укупан број очекиваних речи за свих 91 562 потврђених региона одговора варијанте подвлачења требало да износи 824 058. С обзиром да не успевају сви кандидати да тачно реше сва питања, односно да неки подвлаче и више речи него што треба, неки мање, док неки ни не започињу израду задатка, очекивани број речи износи 792 642. Укупно је детектовано 792 579 речи, што представља успешност од 99.99 %. То значи да су 63 речи биле подвучене, а да систем није детектовао да су подвучене, док је 31 реч била означена као подвучена од стране система, а то није био случај. Просечно време потребно за обраду једне странице са питањима варијанте подвлачења ове класе укључујући и анотацију износи приближно 80 мс.

Ручним увидом у одговоре варијанте подвлачења који нису препознати на успешан начин утврђено је да су кандидати често прецртавали раније подвучену реч са намером да промене свој одговор. Такође, поједини кандидати су користили изузетно слабе и једва приметне линије у подвлачењу што је довело до тога да оне не буду препознате. Пример попуњавања одговора од стране кандидата, који су вршили прецртавање раније подвучене речи, илуструје Слика 70.



Слика 70. Пример прецртавања раније изабране речи које је резултовало у њеном препознавању

Од укупно 91 546 успешно потврђених региона одговора варијанте уписа симбола, сви уписани одговори су успешно утврђени на 88 277 оваквих региона, што представља успешност од 96.43 %. У варијанти уписа симбола, у једном питању је требало уписати 3 цифре, а у другом 5 цифара. Стога би укупан број очекиваних цифара за свих 91 546 потврђених региона одговора

варијанте уписа симбола требало да износи 732 368. С обзиром да не успевају сви кандидати да тачно реше сва питања, односно да неки ни не започињу израду задатка, резултујући број цифара износи 622 512. Успешно је извршена предикција за 616 733 цифре, што представља успешност од 99.07 %. Просечно време потребно за обраду једне странице са питањима варијанте уписа симбола ове класе укључујући и анотацију износи приближно 120 мс. Пример добро препознатог одговора питања ове класе приказује Слика 71.

6. задатак		101211010701	број бодова: 2
Испред функције језика напишите број који стоји испред одговарајућег значења.			
Функција језика:		Значење:	
4	<input checked="" type="checkbox"/> Акумулативна функција	1. Језик је средство споразумевања међу људима.	
3	<input checked="" type="checkbox"/> Експресивна функција	2. Језик пружа могућност измештања у простору и времену.	
1	<input checked="" type="checkbox"/> Комуникативна функција	3. Језик је средство којим се изражавају осећања.	
		4. Језик чува и преноси знања са једне генерације на другу.	

Слика 71. Пример добро препознатих одговора

5.5. Евалуација процесирања питања класе „Кратак одговор“

С обзиром да је тест садржао 1 питање ове класе, а да је целокупни скуп података садржао 45 785 папирних тестова, укупно је очекивано исто толико питања ове класе. Свако питање ове класе на тесту је садржало три региона одговора, те укупан очекивани број региона одговора износи 137 355. Од овог броја региона одговора успешно је детектовано 137 331 региона одговора, што представља успешност од 99.98 %. Ручним увидом је утврђено да је неуспешној детекцији региона одговора допринео превелики шум настао у процесу од фазе тестирања до фазе дигитализације, те није дошло до успешног потврђивања пронађених региона питања и одговора са информацијама добијеним из конфигурационе датотеке теста.

Од укупно 137 331 успешно потврђених региона одговора, сви уписани одговори су успешно утврђени на 131 659 оваквих региона, што представља успешност од 95.87 %. У овом питању, у првом и у другом региону одговора требало је уписати троцифрен број, а у трећем шестоцифрен број, односно укупно 12 цифара. Стога би укупан број очекиваних цифара за свих 137 331 потврђених региона одговора варијанте уписа секвенце симбола требало да износи 1 510 641. Ипак, нису сви кандидати успели на коректан начин да одговоре на ово питање, односно да обезбеде тражени број цифара. Поред тога, неки кандидати нису ни покушавали да реше овај задатак. Због тога, резултујући број цифара износи 1 253 832. Успешно је извршена предикција за 1 238 285 цифара, што представља успешност од 98.76 %. Просечно време потребно за обраду једне странице са питањима ове класе укључујући и анотацију износи приближно 170 мс. Пример добро препознатог одговора питања ове класе илуструје Слика 72, а лоше препознатог одговора Слика 73. Треба напоменути да су ипак две цифре спојене код лоше препознатог одговора.

3. задатак

У складишту је 3 t робе, при чему је 12,5% неисправно.

а) Колика је маса неисправне робе, у килограмима?

Маса неисправне робе је 375 kg.

б) Од укупне масе неисправне робе 76% може да се рециклира. Колико килограма неисправне робе може да се рециклира?

Може да се рециклира 285 kg неисправне робе.

в) Цена исправне робе на тржишту износи 120 динара по килограму. За један килограм рециклиране неисправне робе добије се 45 динара. Ако би се продала сва исправна роба и рециклирала сва роба која се може рециклирати, колико би укупно динара добили?

Добили би укупно 327825 динара.

Слика 72. Пример добро препознатих одговора

3. задатак

У складишту је 3 t робе, при чему је 12,5% неисправно.

а) Колика је маса неисправне робе, у килограмима?

Маса неисправне робе је 375 kg.

б) Од укупне масе неисправне робе 76% може да се рециклира. Колико килограма неисправне робе може да се рециклира?

Може да се рециклира 285 kg неисправне робе.

в) Цена исправне робе на тржишту износи 120 динара по килограму. За један килограм рециклиране неисправне робе добије се 45 динара. Ако би се продала сва исправна роба и рециклирала сва роба која се може рециклирати, колико би укупно динара добили?

Добили би укупно 327825 динара.

Слика 73. Лоше препознавање одговора настало услед спајања цифара приликом писања

5.6. Сумарни приказ резултата

У претходном поглављу, током евалуционог процеса идентификовано је пет кључних ставки које овај софтверски систем треба да детектује: идентификациони код, странични код, питање класе вишеструки избор, питање класе повезивање и питање класе кратак одговор. За сваку од ових ставки наведен је основни елемент који треба детектовати, а то су једнодимензионални бар-код, дводимензионални бар-код, круг, линија/симбол и симбол. Поред тога, избројано је колико ових елемената је потребно детектовати, а затим израчунато колико је елемената успешно детектовано и коју тачност је систем испољио у процесу детекције сваког од тих елемената у контексту ставке у којој се налази. Ове податке на концизан начин наводи Табела 6.

Табела 6. Преглед резултата предложеног система

Ставка	Елемент	Број елемената за детекцију	Број успешно детектованих елемената	Тачност система (%)
Идентификациони код	Једнодимензионални бар-код	45 785	45 626	99.99
Странични код	Дводимензионални бар-код	732 411	732 403	99.99
Питање типа „Вишеструки избор“	Круг	6 584 989	6 579 847	99.92
Питање типа „Повезивање“	Линија (Симбол)	792 642 (622 512)	792 579 (616 733)	99.99 (99.07)
Питање типа „Кратак одговор“	Симбол	1 238 285	1 253 832	98.76

5.7. Анализа постигнутих резултата

Приликом проучавања изабраних софтверских система намењених за аутоматизовано оцењивање папирних тестова уочено је да ниједан од њих не поседује могућност оцењивања папирног теста који садржи различите класе питања као што су вишеструки избор, повезивање и кратак одговор. Утврђено је да је сваки од њих специјализован за оцењивање тачно једне класе питања. Поред тога, већина претходно проучаваних изабраних софтверских решења није прилагођена генерализованом формату папирног теста, односно прилагођена је формату у којем су обрасци са текстом питања и обрасци са одговорима раздвојени.

Додатно, да би изабрани софтверски системи испољавали могућност оцењивања папирног теста генерализованог формата који није стриктан, морале би да се направе значајне измене у изворном коду софтверских решења. Наравно, остаје питање како би се ове измене одразиле на њихове перформансе. Имплементирани систем представља решење које испољава способност да аутоматизовано оцењује папирне тестове генералног формата са различитим класама питања, уз високе перформансе у погледу тачности и брзине у раду система.

Штавише, одлична тачност у раду система последица је коришћења софистицираних алгоритама. Ови алгоритми се користе за утврђивање исправности приликом детекције геометријских облика као што су правоугаоник и круг. Поред тога, коришћење алгорита за адаптивно одређивање прагова зацрњености облика резултује у изузетним резултатима постигнутим у погледу оцењивања питања класе вишеструки избор.

Такође, коришћење разноврсних морфолошких операција над сликом уз примену алгоритама за идентификацију и упаривање кључних тачака над двема сликама доприноси одличној тачности система у оцењивању питања класе повезивање. Осим тога, коришћењем техника дубоког учења, систем постиже високу тачност у предикцији руком написаног одговора, која се испољава при оцењивању варијанте питања класе кратак одговор у којима треба да се руком допише нумерички одговор, односно варијанте питања класе повезивања у којима треба дописати симбол. Овоме додатно доприносе и алгоритми за детекцију, изолацију и брисање хоризонталних линија, као и операције за репарацију слике након тих процеса.

С обзиром да су изабрана софтверска решења за аутоматизовано оцењивање папирних тестова намењена оцењивању специфично једне класе питања, да би могло да се направи коректно поређење, за сваку од класа питања које предложени систем може да процесира биће направљена по једна табела. У оквиру сваке од тих табела, с обзиром да испољава могућност оцењивања све три класе питања (вишеструки избор, повезивање и кратак одговор) као један ред табеле налазиће се и предложени софтверски систем. Остали системи ће бити распоређени у табелама према класи питања коју су способни да оцењују. Табела 7 даје упоредни приказ софтверских система који процесирају класу питања „Вишеструки избор“.

Табела 7. Преглед резултата за класу питања „Вишеструки избор“

Софтверски систем	Величина скупа података	Време потребно за процесирање једне странице теста	Тачност система (%)
Предложени систем	45 785	220 мс	99.91
<i>TARS</i>	763	290 мс	99.30
<i>MCTQF</i>	88	400 мс – 2270 мс	100.00
<i>MCG-RAS</i>	800	68 000 мс	98.75
GMCQ-FA	200	7 600 мс	98.50
<i>Eyegrade</i>	233	9 005 мс	99.40

Предложени систем показује високу тачност у оцењивању питања класе вишеструког избора, која је, осим система МСТQF, виша од свих осталих изабраних система специјализованих за оцењивање питања те класе. Међутим, систем МСТQF је тестиран на значајно мањем скупу података од предложеног система. Такође предложени систем захтева најмање времена за процесирање једне странице са питањима ове класе. Табела 8 даје упоредни приказ софтверских система који процесирају класу питања „Повезивање“.

Табела 8. Преглед резултата за класу питања „Повезивање“

Софтверски систем	Величина скупа података	Време потребно за процесирање једне странице теста	Тачност система (%)
Предложени систем	45 785	70 мс (110 мс)	99.97 (96.43)
<i>ASSHEP</i>	нема података	нема података	нема података
<i>AGHAS</i>	250	нема података	92.86
<i>ARHC</i>	148	нема података	92.60

Предложени систем показује високу тачност у оцењивању питања класе повезивања, која је виша од свих осталих изабраних система специјализованих за оцењивање питања те класе. Конкурентска решења су тестирана на значајно мањем скупу података од предложеног

система. Такође, предложени систем захтева најмање времена за процесирање једне странице са питањима ове класе, с обзиром да конкурентски системи нису навели време потребно за обраду једне странице теста. Табела 9 даје упоредни приказ софтверских система који процесирају класу питања „Кратак одговор“.

Табела 9. Преглед резултата за класу питања „Кратак одговор“

Софтверски систем	Величина скупа података	Време потребно за процесирање једне странице теста	Тачност система
Предложени систем	45 785	170 мс	95.87 %
<i>FASAS</i>	300 000	нема података	0.86 – 0.94 QWK
<i>ISSHSA</i>	40 000	нема података	74 %
<i>TSAWR</i>	8 400	нема података	91.12 %
HSAES	нема података	нема података	нема података
<i>AGHNA</i>	15	нема података	96.7 %
<i>AGSLCQ</i>	887	100 – 300 мс	91.3 %

Предложени систем показује високу тачност у оцењивању питања класе кратког одговора, која је виша од свих осталих изабраних система специјализованих за оцењивање питања те класе. Изузев система *FASAS*, конкурентска решења су тестирана на значајно мањем скупу података од предложеног система. Такође, предложени систем је у сличном рангу са системом *AGSLCQ* по питању захтеваног времена неопходног за процесирање једне странице са питањима ове класе, с обзиром да конкурентски системи нису навели време потребно за обраду једне странице теста.

С обзиром да су одабрани софтверски системи специјализовани за аутоматизовано оцењивање тачно једне класе питања, то би требало да значи да међу одабраним софтверским решењима који оцењују различите класе питања постоје знатне разлике. Међутим, уочене су и сличности које се тичу самог тока процеса аутоматизованог оцењивања, а истакнуте су корацима који чине тај процес. Скоро сви софтверски системи за аутоматизовано оцењивање папирних тестова пролазе кроз сличне фазе: дизајнирање теста, штампање дизајнираног теста, умножавање одштампаног теста, спровођење процеса тестирања кандидата путем папирних тестова, дигитализација попуњених папирних тестова, процесирање папирних тестова и упис и чување добијених резултата. Наравно, варијације леже у имплементационим детаљима сваког од корака, нарочито корака за процесирање питања и одговора теста.

6. Закључак

Испитивање и оцењивање показаног знања кандидата је саставни део посла једног академског радника. Овај део посла је репетитивне природе те је често неинспиративан, мукотрпан и заморан. Поред тога, подложен је субјективности у процесу ручног оцењивања коју прегледач несвесно уноси, а која је узрокована, између осталог и претходно наведеним факторима. Чак и у случају ручног оцењивања питања у којима су одговори или недвосмислено тачни или недвосмислено нетачни, овакво оцењивање је подложно ненамерним грешкама узрокованим претходно наведеним факторима, чиме се оцењивање може извршити на штету или корист оцењиваног кандидата.

Стога се све више користе платформе за електронско учење и испитивање са циљем елиминације субјективности и грешака које уноси прегледач у процесу прегледања. Ове платформе поред пружања једноставног начина за организовање кандидата и предметног градива кроз концепт курсева, обезбеђују и широк спектар питања и могућности за њихово оцењивање. Међутим, за исправно функционисање оне захтевају и особље задужено за њихову администрацију да би се испитни процес спровео на коректан начин и у ограниченом режиму података доступних кандидатима. Ипак, и поред примена сигурносних мера, постоје начини којима се оне могу заобићи ради остваривања циља који није у складу са правилима испитног процеса.

Такође, неки процеси испитивања, као што су испити на факултетима или пријемни испити за упис средњих школа или факултета, морају да се спроведу у исто време за све кандидате, како би се одржао исти критеријум испитивања. Највећи изазов оваквих процеса испитивања је огроман број пријављених кандидата, за које је поред обезбеђивања простора и радног места, потребно обезбедити и еквивалентан број персоналних рачунара на којима ће се изводити процес тестирања кандидата уз коришћење платформи за електронско учење и оцењивање. Ово често представља недостижно решење за многе образовне институције које спроводе испитивање кандидата, најчешће због недостатка материјалних средстава неопходних за подмиривање потреба за рачунарским ресурсима, мрежном инфраструктуром која ће их повезати и особљем које ће вршити њихову администрацију.

Да би се одржали исти услови испитивања кандидата, образовне институције које спроводе испитивање кандидата прибегавају састављању једне поставке теста, а затим процес испитивања спроводе путем папирних тестова. Тест пролази кроз неколико фаза, као што су дизајнирање теста, његово штампање, умножавање у потребном броју примерака, попуњавање од стране кандидата и на самом крају фаза оцењивања. У фази оцењивања потребно је ручно оценити велики број тестова, што има за последицу неке од раније наведених проблема. Због тога су развијана софтверска решења која треба да аутоматски и на објективан начин изврше оцењивање папирних тестова.

Папирни тестови могу садржати палету различитих типова питања, при чему многе институције користе синонимне називе за практично исти тип питања. Стога је извршена класификација најчешће коришћених типова питања у циљу лакше евалуације истраживања, при чему су једном класом обједињени сродни типови питања. За сваку од класа питања утврђених новом класификацијом наведене су форме у којима се питања задају, начин и циљ онога што та питања треба да установе, који начин размишљања треба да активирају код кандидата, као и предности и мане ових класа питања.

Прегледом до сада реализованих софтверских система који испољавају могућност аутоматизованог оцењивања папирних тестова установљено је да је сваки од њих специјализован за евалуацију тачно једне класе питања. Међутим, приликом састављања теста тежи се да он буде што хетерогенији, што се постиже употребом различитих класа питања. Мотивација за хетерогенизацију теста у погледу класа питања који садржи лежи у томе да

свака класа питања има своје предности мане, а да се одређене мане једне класе питања могу надоместити употребом друге класе питања.

Поред тога, установљено је да раздвајање поставки питања и простора за упис одговора на одвојене обрасце дефокусира кандидате у процесу решавања теста. Додатно, овај начин уписивања одговора од стране кандидата је подложен грешкама, које се могу каскадно преносити на наредне одговоре питања. Прегледом до сада реализованих софтверских решења који испољавају могућност аутоматизованог оцењивања папирних тестова установљено је да већи део њих користи одвојене обрасце за поставку питања и уписивање одговора.

Примарни циљ ове докторске дисертације јесте приказивање детаља развијеног софтверског система који пружа могућност аутоматизоване евалуације различитих класа питања и одговора на папирним тестовима који нису у стриктној форми и на којима не постоје раздвојени обрасци за текст питања и уношење одговора, већ су одговори обједињени регионима питања којима припадају. Додатан подстрек у развоју овог софтверског решења представљали су раније развијени алати и системи од стране аутора и његових колега, који су намењени за аутоматизовано оцењивање тестова. Ова решења су развијана у оквиру Катедре за Рачунарску технику и информатику Електротехничког факултета Универзитета у Београду за потребе предмета Програмирање 1 и Програмирање 2 и намењена су оцењивању студентских радова који обухватају изворне кодове написане на симболичком машинском језику, Пајтону, језику C и за испитивање теоријских питања.

Имплементирани софтверски систем показује способност оцењивања питања класе вишеструки избор, у којима кандидати одговоре бирају тако што попуњавају кругове који представљају понуђене одговоре, варијанте класе питања повезивање, у којима кандидати одговоре бирају тако што подвлаче речи у тексту који се састоји од параграфа и реченица, варијанте класе повезивања у којима се одговори спајају уписивањем симбола на за то предвиђене линије и класе кратак одговор, у којима кандидати на за то предвиђеним линијама врше уписивање секвенце симбола. Да би успео да изврши евалуацију папирних тестова са различитим класама питања, систем се ослања на сопствено реализоване алгоритме и технике вештачке интелигенције као што су рачунарска визија и дубоко учење уз коришћење конволуционих неуралних мрежа.

Тестирање предложеног софтверског система спроведено је над великим скупом папирних тестова, који садржи више од 45 хиљада примерка тестова, што даје додатну тежину овом истраживању. Предложени систем је испољио високу тачност у свом раду, која износи 99.99 % у препознавању једнодимензионалних идентификационих и дводимензионалних страничних бар-кодова, 99.92 % у предикцији елемената питања класе вишеструког одговора, 99.99 % и 99.07 % у предикцији елемената класе повезивања и 98.76 % у предикцији елемената класе кратког одговора. Такође, систем је способан да детектује и евидентира различите типове грешака које се могу појавити у току обраде, као и грешака у начину попуњавања одговора од стране кандидата, а које произилазе из непоштовања упутстава за попуњавање одговора теста.

Полазне хипотезе које су доказане овим истраживањем су:

- Аутоматизација процеса оцењивања папирних тестова може допринети бољим перформансама, квалитету и резултатима самог процеса.
- Могуће је извршити класификацију питања која се могу појавити на папирним тестовима.
- За сваку од идентификованих класа питања могу се утврдити карактеристике које доприносе бољој реализацији процеса аутоматизованог оцењивања папирних тестова.

- За сваку од идентификованих класа питања могуће је реализовати алгоритам користећи технике вештачке интелигенције који ће постићи боље резултате у оцењивању папирних тестова него човек.
- Могуће је реализовати софтверски систем за аутоматизовано оцењивање папирног теста различите структуре и сложености са задовољавајућим перформансама.

Такође, остварено је више научних доприноса:

- Анализа распрострањености извођења испитивања кандидата путем папирних тестова.
- Преглед и класификација најчешће коришћених питања на тестовима.
- Преглед и анализа постојећих система за аутоматизовано оцењивање тестова.
- Класификација постојећих система за аутоматизовано оцењивање тестова према унапред утврђеном шаблону.
- Имплементација система за аутоматизовано оцењивање тестова користећи технике вештачке интелигенције.
- Опсежна евалуација имплементираних система над великим скупом папирних тестова у дигиталном облику.
- Поређење добијених резултата са резултатима релевантних истраживања.

Систем врло лако може да се прилагоди и кратким одговорима који се састоје и од осталих знакова писма циљаног језика [72] [73], јер је у ове сврхе потребно пронаћи или формирати скуп података са овим знаковима [74], а онда истренирати модел неуралне мреже за предикцију [75]. Иако систем показује изузетне перформансе у свом раду, било би пожељно да буде робуснији и адаптивнији у ситуацијама када је суочен са субоптималним квалитетом скенирања. Међутим, највећи изазов представља могућност да систем у будућности оцењује питања класе есеја. У том случају није довољна само детекција написаног одговора као у случају питања типа кратког одговора, већ и разумевање његовог значења да би он могао да се упореди са референтним тачним одговором [76]. Да би се овај проблем решио, потребно је користити додатне технике вештачке интелигенције, као што су обрада природног језика [77]. Поред тога, велики изазов би представљало оцењивање питања која захтевају цртање одговора [78]. Такође, било би корисно да систем има могућност да верификује идентитет [79] кандидата који је радио тест користећи његов рукопис [80] или неке друге биометријске атрибуте [81] тако да додатно спречи злоупотребу идентитета у процесу тестирања.

Резултати овог истраживања имају примену у области образовања, односно њеној подобласти испитивања и оцењивања, у образовним институцијама у којима се испитивање спроводи коришћењем папирних тестова услед недостатка материјалних ресурса за набавку рачунара и услуга особља њиховог одржавања. Најбитнији аспект овог истраживања је што предложени систем значајно убрзава процес оцењивања кандидата, доприноси објективности у оцењивању и растеређује наставни кадар од великог обима посла, који захтева значајне временске ресурсе који се могу искористити у сврху унапређења наставног процеса. Треба истаћи да актуелности овог истраживања доприноси и велики број савремених референци у литератури, што потврђује да је ова област и даље од значаја и да постоји потенцијал за њен додатни развој.

Иако је истраживање резултовало у софтверском систему за аутоматизовано оцењивање папирних тестова кандидата, његова примена превазилази академске сврхе и може се искористити и за друге намене. Овај систем би могао да се примени у оцењивању осталих процеса тестирања која се спровode на папирним тестовима, као што су тестови за полагање

познавања саобраћајних прописа у циљу остваривања возачке дозволе. Поред тога, може се користити за добијање резултата великих анкетања која се спроводе путем папирних образаца. Додатно, систем може да се примењује за прикупљање информација о историји болести пацијената кроз медицинске упитнике или друге папирне форме које пацијенти попуњавају у медицинским установама.

Литература

- [1] Z. Stanisavljevic, B. Nikolic, I. Tartalja, and V. Milutinovic, "A classification of eLearning tools based on the applied multimedia," *Multimed Tools Appl*, vol. 74, pp. 3843–3880, 2015.
- [2] G. Fenu, M. Marras, and L. Boratto, "A multi-biometric system for continuous student authentication in e-learning platforms," *Pattern Recognit Lett*, vol. 113, pp. 83–92, 2018.
- [3] M. Kumar, M. K. Jindal, and M. Kumar, "A systematic survey on CAPTCHA recognition: types, creation and breaking techniques," *Archives of Computational Methods in Engineering*, vol. 29, no. 2, pp. 1107–1136, 2022.
- [4] H. Jeong, "A comparative study of scores on computer-based tests and paper-based tests," *Behaviour & Information Technology*, vol. 33, no. 4, pp. 410–422, 2014.
- [5] I. Lewis, B. Watson, and K. M. White, "Internet versus paper-and-pencil survey methods in psychological experiments: Equivalence testing of participant responses to health-related messages," *Aust J Psychol*, vol. 61, no. 2, pp. 107–116, 2009.
- [6] M. E. G. Chamorro, "Cognitive validity evidence of computer-and paper-based writing tests and differences in the impact on EFL test-takers in classroom assessment," *Assessing Writing*, vol. 51, p. 100594, 2022.
- [7] A. Nardi and M. Ranieri, "Comparing paper-based and electronic multiple-choice examinations with personal devices: Impact on students' performance, self-efficacy and satisfaction," *British Journal of Educational Technology*, vol. 50, no. 3, pp. 1495–1506, 2019.
- [8] Ö. Z. Hüseyin and T. Özturan, "Computer-based and paper-based testing: Does the test administration mode influence the reliability and validity of achievement tests?," *Journal of Language and Linguistic Studies*, vol. 14, no. 1, pp. 67–85, 2018.
- [9] T. McClelland and J. Cuevas, "A comparison of computer based testing and paper and pencil testing in mathematics assessment," *The Online Journal of New Horizons in Education*, vol. 10, no. 2, pp. 78–89, 2020.
- [10] S. Candrljic, M. A. Katić, and M. H. Dlab, "Online vs. Paper-based testing: A comparison of test results," in *2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, IEEE, 2014, pp. 657–662.
- [11] S. Xiao, T. Li, and J. Wang, "Optimization methods of video images processing for mobile object recognition," *Multimed Tools Appl*, vol. 79, pp. 17245–17255, 2020.
- [12] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.
- [13] J. Sigut, M. Castro, R. Arnay, and M. Sigut, "OpenCV basics: A mobile application to support the teaching of computer vision concepts," *IEEE Transactions on Education*, vol. 63, no. 4, pp. 328–335, 2020.
- [14] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: A brief review," *Comput Intell Neurosci*, vol. 2018, 2018.

- [15] L. Lopez-Fuentes, J. van de Weijer, M. González-Hidalgo, H. Skinnemoen, and A. D. Bagdanov, "Review on computer vision techniques in emergency situations," *Multimed Tools Appl*, vol. 77, pp. 17069–17107, 2018.
- [16] N. Odeh and C. Direkoglu, "Automated shopping system using computer vision," *Multimed Tools Appl*, vol. 79, no. 41–42, pp. 30151–30161, 2020.
- [17] K. C. Santosh and S. K. Antani, "Recent trends in image processing and pattern recognition," *Multim. Tools Appl.*, vol. 79, no. 47, pp. 34697–34699, 2020.
- [18] A. Bošnjaković, J. Protic, D. Bojić, and I. Tartalja, "Automating the knowledge assessment workflow for large student groups: A development experience," *Int J Eng Educ*, vol. 31, no. 4, pp. 1058–1070, 2015.
- [19] A. Patil and M. Rane, "Convolutional neural networks: an overview and its applications in pattern recognition," *Information and Communication Technology for Intelligent Systems: Proceedings of ICTIS 2020, Volume 1*, pp. 21–30, 2021.
- [20] M. Mišić, V. Jocović, J. Đukić, M. Prodanov, A. Srbljanović, and M. Obradović, "SPROVOĐENJE PRAKTIČNE NASTAVE PROGRAMIRANJA NA ELEKTROTEHNIČKOM FAKULTETU UNIVERZITETA U BEOGRADU U ONLINE OKRUŽENJU," 27. скуп ТРЕНДОВИ РАЗБОЈА: "On-line настава на универзитетима", Факултет техничких наука Универзитета у Новом Саду, Нови Сад, Нови Сад, Србија, pp. 379–382, 2021.
- [21] V. Jocović, J. Đukić, and M. Mišić, "First Experiences with Moodle and Coderunner Platforms in Programming Course," in *Proceedings of the Tenth International Conference on e-Learning, Belgrade Metropolitan University, Belgrade*, 2019, pp. 81–86.
- [22] M. Mišić et al., "PROGRAMSKI JEZIK PYTHON U NASTAVI PROGRAMIRANJA NA ELEKTROTEHNIČKOM FAKULTETU UNIVERZITETA U BEOGRADU-IZAZOVI I REŠENJA," 27. скуп ТРЕНДОВИ РАЗБОЈА: "On-line настава на универзитетима", Факултет техничких наука Универзитета у Новом Саду, Нови Сад, Нови Сад, Србија, pp. 245–248, 2021.
- [23] J. Đukić, V. Jocović, M. Mišić, and M. Tomašević, "Automated grading system for picoComputer assembly codes integrated within E-Learning platform," *PROCEEDINGS of IX International Conference on Electrical, Electronic and Computing Engineering, IcEtran 2022, ETRAN Society, Belgrade, Academic Mind, Belgrade, Novi Pazar, Serbia*, pp. 626–630, 2022.
- [24] C. Loudon and A. Macias-Muñoz, "Item statistics derived from three-option versions of multiple-choice questions are usually as robust as four-or five-option versions: implications for exam design," *Adv Physiol Educ*, vol. 42, no. 4, pp. 565–575, 2018.
- [25] R. E. Tractenberg, M. M. Gushta, S. E. Mulroney, and P. A. Weissinger, "Multiple choice questions can be designed or revised to challenge learners' critical thinking," *Advances in Health Sciences Education*, vol. 18, pp. 945–961, 2013.
- [26] P. Tirpude, P. Girhepunje, S. Sahu, S. Zilpe, and H. Ragite, "Real time object detection using OpenCV-Python," *Int Res J Modernization Eng Technol Sci*, vol. 4, no. 5, pp. 1–6, 2022.

- [27] M. Zhao, X. Jia, and D.-M. Yan, "An occlusion-resistant circle detector using inscribed triangles," *Pattern Recognit*, vol. 109, p. 107588, 2021.
- [28] T.-C. Chen and K.-L. Chung, "An efficient randomized algorithm for detecting circles," *Computer vision and image understanding*, vol. 83, no. 2, pp. 172–191, 2001.
- [29] A. Lopez-Martinez and F. J. Cuevas, "Automatic circle detection on images using the teaching learning based optimization algorithm and gradient analysis," *Applied Intelligence*, vol. 49, pp. 2001–2016, 2019.
- [30] Z. Yao and W. Yi, "Curvature aided Hough transform for circle detection," *Expert Syst Appl*, vol. 51, pp. 26–33, 2016.
- [31] G. Lai, Q. Xie, H. Liu, Y. Yang, and E. Hovy, "Race: Large-scale reading comprehension dataset from examinations," *arXiv preprint arXiv:1704.04683*, 2017.
- [32] D. Golekar, R. Bula, R. Hole, S. Katare, and S. Parab, "Sign language recognition using Python and OpenCV," *Int Res J Modernization Eng Technol Sci*, vol. 4, no. 2, pp. 1–5, 2022.
- [33] J. S. Velasco *et al.*, "Alphanumeric test paper checker through intelligent character recognition using openCV and support vector machine," in *World Congress on Engineering and Technology; Innovation and its Sustainability 2018 1*, Springer, 2020, pp. 119–128.
- [34] N. Bacanin, T. Bezdan, K. Venkatachalam, and F. Al-Turjman, "Optimized convolutional neural network by firefly algorithm for magnetic resonance image classification of glioma brain tumor grade," *J Real Time Image Process*, vol. 18, no. 4, pp. 1085–1098, 2021.
- [35] L. Camus and A. Filighera, "Investigating transformers for automatic short answer grading," in *Artificial Intelligence in Education: 21st International Conference, AIED 2020, Ifrane, Morocco, July 6–10, 2020, Proceedings, Part II 21*, Springer, 2020, pp. 43–48.
- [36] S. González-López, Z. G. Montes-Rosales, A. P. López-Monroy, A. López-López, and J. M. García-Gorrostieta, "Short answer detection for open questions: a sequence labeling approach with deep learning models," *Mathematics*, vol. 10, no. 13, p. 2259, 2022.
- [37] J. G. Borade and L. D. Netak, "Automated grading of essays: a review," in *Intelligent Human Computer Interaction: 12th International Conference, IHCI 2020, Daegu, South Korea, November 24–26, 2020, Proceedings, Part I 12*, Springer, 2021, pp. 238–249.
- [38] K. Taghipour and H. T. Ng, "A neural approach to automated essay scoring," in *Proceedings of the 2016 conference on empirical methods in natural language processing*, 2016, pp. 1882–1891.
- [39] F. Dong and Y. Zhang, "Automatic features for essay scoring—an empirical study," in *Proceedings of the 2016 conference on empirical methods in natural language processing*, 2016, pp. 1072–1077.
- [40] SmartGrade, "SmartGrade - grade multiple choice exams in seconds." [Online]. Available: <https://smartgrade.net/>
- [41] iDoceo Studios, "Home - GradeScanner by iDoceo Studios." [Online]. Available: <https://www.gradescanner.net/>

- [42] A. Bosnjakovic, J. Protic, and I. Tartalja, "Development of a software system for automated test assembly and scoring," in *ICERI2010 Proceedings*, IATED, 2010, pp. 6012–6016.
- [43] M. Alomran and D. Chai, "Automated scoring system for multiple choice test with quick feedback," *International Journal of Information and Education Technology*, vol. 8, no. 8, pp. 538–545, 2018.
- [44] J. A. Catalan, "A framework for automated multiple-choice exam scoring with digital image and assorted processing using readily available software," in *DLSU research congress*, 2017, pp. 1–5.
- [45] S. Patole, A. Pawar, A. Patel, A. Panchal, and R. Joshi, "Automatic system for grading multiple choice questions and feedback analysis," *IEEE International Journal of Technical Research and Applications*, vol. 12, no. 39, pp. 16–19, 2016.
- [46] J. A. Fisteus, A. Pardo, and N. F. García, "Grading multiple choice exams with low-cost and portable computer-vision techniques," *J Sci Educ Technol*, vol. 22, pp. 560–571, 2013.
- [47] M. Lu, W. Zhou, and R. Ji, "Automatic Scoring System for Handwritten Examination Papers Based on YOLO Algorithm," in *Journal of Physics: Conference Series*, IOP Publishing, 2021, p. 12030.
- [48] E. Shaikh, I. Mohiuddin, A. Manzoor, G. Latif, and N. Mohammad, "Automated grading for handwritten answer sheets using convolutional neural networks," in *2019 2nd International conference on new trends in computing sciences (ICTCS)*, Ieee, 2019, pp. 1–6.
- [49] M. Supic, K. Brkic, T. Hrkac, Ž. Mihajlović, and Z. Kalafatić, "Automatic recognition of handwritten corrections for multiple-choice exam answer sheets," in *2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, IEEE, 2014, pp. 1136–1141.
- [50] H. T. Nguyen, C. T. Nguyen, H. Oka, T. Ishioka, and M. Nakagawa, "Fully automatic scoring of handwritten descriptive answers in Japanese language tests," *arXiv preprint arXiv:2201.03215*, 2022.
- [51] Y. Lin, L. Zheng, F. Chen, S. Sun, Z. Lin, and P. Chen, "Design and Implementation of Intelligent Scoring System for Handwritten Short Answer Based on Deep Learning," in *2020 IEEE International Conference on Artificial Intelligence and Information Systems (ICAIS)*, IEEE, 2020, pp. 184–189.
- [52] A. Das, H. Suwanwiwat, U. Pal, and M. Blumenstein, "ICFHR 2020 competition on short answer assessment and Thai student signature and name components recognition and verification (SASIGCOM 2020)," in *2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, IEEE, 2020, pp. 222–227.
- [53] P. J. Sijimol and S. M. Varghese, "Handwritten short answer evaluation system (HSAES)," *Int J Sci Res Sci Technol*, vol. 4, no. 2, pp. 1514–1518, 2018.
- [54] M. T. Brown, "Automated Grading of Handwritten Numerical Answers," 2017.
- [55] A. K.-F. Lui, L.-K. Lee, and H.-W. Lau, "Automated grading of short literal comprehension questions," in *Technology in Education. Technology-Mediated Proactive*

Learning: Second International Conference, ICTE 2015, Hong Kong, China, July 2-4, 2015, Revised Selected Papers 2, Springer, 2015, pp. 251–262.

- [56] A. O. Djekoune, K. Messaoudi, and K. Amara, “Incremental circle hough transform: An improved method for circle detection,” *Optik (Stuttg)*, vol. 133, pp. 17–31, 2017.
- [57] R. Huang, J. Pedoeem, and C. Chen, “YOLO-LITE: a real-time object detection algorithm optimized for non-GPU computers,” in *2018 IEEE international conference on big data (big data)*, IEEE, 2018, pp. 2503–2510.
- [58] A. Manzanera, T. P. Nguyen, and X. Xu, “Line and circle detection using dense one-to-one Hough transforms on greyscale images,” *EURASIP J Image Video Process*, vol. 2016, pp. 1–18, 2016.
- [59] V. Jocovic, M. Marinkovic, S. Stojanovic, and B. Nikolic, “Automated assessment of pen and paper tests using computer vision,” *Multimed Tools Appl*, pp. 1–22, 2023.
- [60] Y. Meir, O. Tevet, Y. Tzach, S. Hodassman, R. D. Gross, and I. Kanter, “Efficient shallow learning as an alternative to deep learning,” *Sci Rep*, vol. 13, no. 1, p. 5423, 2023.
- [61] J. Zhang, X. Yu, X. Lei, and C. Wu, “A novel deep LeNet-5 convolutional neural network model for image recognition,” *Computer Science and Information Systems*, vol. 19, no. 3, pp. 1463–1480, 2022.
- [62] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [63] Z. Tang *et al.*, “Automatic sparse connectivity learning for neural networks,” *IEEE Trans Neural Netw Learn Syst*, 2022.
- [64] S. Lin, X. Ma, S. Ye, G. Yuan, K. Ma, and Y. Wang, “Toward extremely low bit and lossless accuracy in dnns with progressive admm,” *arXiv preprint arXiv:1905.00789*, 2019.
- [65] T. Dasgupta, A. Naskar, L. Dey, and R. Saha, “Augmenting textual qualitative features in deep convolution recurrent neural network for automatic essay scoring,” in *Proceedings of the 5th Workshop on Natural Language Processing Techniques for Educational Applications*, 2018, pp. 93–102.
- [66] N. Bacanin, T. Bezdan, E. Tuba, I. Strumberger, and M. Tuba, “Monarch butterfly optimization based convolutional neural network design,” *Mathematics*, vol. 8, no. 6, p. 936, 2020.
- [67] Q. Huang, “Weight-quantized squeezenet for resource-constrained robot vacuums for indoor obstacle classification,” *AI*, vol. 3, no. 1, pp. 180–193, 2022.
- [68] A. Sher, A. Trusov, E. Limonova, D. Nikolaev, and V. V Arlazarov, “Neuron-by-Neuron Quantization for Efficient Low-Bit QNN Training,” *Mathematics*, vol. 11, no. 9, p. 2112, 2023.
- [69] M. Tasci, A. Istanbulu, S. Kosunalp, T. Iliev, I. Stoyanov, and I. Beloev, “An Efficient Classification of Rice Variety with Quantized Neural Networks,” *Electronics (Basel)*, vol. 12, no. 10, p. 2285, 2023.
- [70] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, and K. Keutzer, “A survey of quantization methods for efficient neural network inference,” in *Low-Power Computer Vision*, Chapman and Hall/CRC, 2022, pp. 291–326.

- [71] L. Deng, "The mnist database of handwritten digit images for machine learning research [best of the web]," *IEEE Signal Process Mag*, vol. 29, no. 6, pp. 141–142, 2012.
- [72] M. Kumar, M. K. Jindal, and M. Kumar, "Distortion, rotation and scale invariant recognition of hollow Hindi characters," *Sāadhanā*, vol. 47, no. 2, p. 92, 2022.
- [73] M. Kumar, M. K. Jindal, and M. Kumar, "Design of innovative CAPTCHA for hindi language," *Neural Comput Appl*, pp. 1–36, 2022.
- [74] U.-V. Marti and H. Bunke, "The IAM-database: an English sentence database for offline handwriting recognition," *International Journal on Document Analysis and Recognition*, vol. 5, pp. 39–46, 2002.
- [75] R. Vaidya, D. Trivedi, S. Satra, and M. Pimpale, "Handwritten character recognition using deep-learning," in *2018 second international conference on inventive communication and computational technologies (ICICCT)*, IEEE, 2018, pp. 772–775.
- [76] A. Shehab, M. Faroun, and M. Rashad, "An automatic Arabic essay grading system based on text similarity Algorithms," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 3, 2018.
- [77] V. Rowtula, S. R. Oota, and C. V Jawahar, "Towards automated evaluation of handwritten assessments," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*, IEEE, 2019, pp. 426–433.
- [78] M. Bansal, M. Kumar, M. Sachdeva, and A. Mittal, "Transfer learning for image classification using VGG19: Caltech-101 image data set," *J Ambient Intell Humaniz Comput*, pp. 1–12, 2021.
- [79] A. Rehman, S. Naz, and M. I. Razzak, "Writer identification using machine learning approaches: a comprehensive review," *Multimed Tools Appl*, vol. 78, pp. 10889–10931, 2019.
- [80] P. Dansena, S. Bag, and R. Pal, "Pen ink discrimination in handwritten documents using statistical and motif texture analysis: A classification based approach," *Multimed Tools Appl*, vol. 81, no. 21, pp. 30881–30909, 2022.
- [81] K. Shaheed, A. Mao, I. Qureshi, Q. Abbas, M. Kumar, and X. Zhang, "Finger-vein presentation attack detection using depthwise separable convolution neural network," *Expert Syst Appl*, vol. 198, p. 116786, 2022.

Биографија

Владимир Јоцовић, дипл. инж. електротехнике и рачунарства, рођен је 25.11.1993. године у Београду. Основну школу „Бранко Радичевић“ завршио је у Београду као ђак генерације и носилац Вукове дипломе. Девету гимназију у Београду (природно-математички смер) завршио је као носилац Вукове дипломе. Током основне и средње школе такмичио се и освајао награде и похвале на градским и републичким такмичењима из математике, физике и српског језика. Поседује две дипломе Кембриџ универзитета за енглески језик (нивои Б2 и Ц1).

Уписао се 2012. године на Електротехнички факултет Универзитета у Београду, студијски програм Софтверско инжењерство, а дипломирао 2016. године са просечном оценом 9.89. Дипломски рад „Едукативни Веб систем за визуелизацију алгоритама претраживања“ под менторством проф. др Бошка Николића одбранио је у септембру 2016. са оценом 10. Током основних студија био је ангажован као демонстратор на више од 10 предмета на Катедри за рачунарску технику и информатику.

Добитник је више стипендија: студентима високошколских установа у Републици Србији, студентима високошколских установа у Београду, стипендија Министарства просвете за изузетно надарене студенте, стипендије за најбоље студенте завршних година основних и мастер академских студија (Доситеја) итд.

Мастер академске студије на Електротехничком факултету Универзитета у Београду, на Модулу за Софтверско инжењерство уписао је у октобру 2016. године, а дипломирао 2018. године са просечном оценом 9.5. Мастер рад „Унапређење софтверског система за процену нивоа звучне изолације различитих конструкција“ под менторством проф. др Бошка Николића одбранио је у септембру 2018. године са оценом 10.

Докторске академске студије на Електротехничком факултету Универзитета у Београду, на Модулу за Софтверско инжењерство уписао је у октобру 2018. године. Положио је све испите са оценом 10 и остварио 120 ЕСПБ. У истраживачком раду усмерио се ка аутоматизацији оцењивања коришћењем области вештачке интелигенције: машинског учења и компјутерске визије.

Од децембра 2016. године запослен је на Електротехничком факултету Универзитета у Београду као сарадник у настави на Катедри за рачунарску технику и информатику, а потом од децембра 2018. године као асистент. Тренутно је ангажован као асистент на 10 предмета који се изводе на студијским програмима Софтверско инжењерство и Рачунарска техника и информатика: Практикум из коришћења рачунара, Увод у рачунарство, Практикум из програмирања 1, Практикум из програмирања 2, Програмирање 1, Програмирање 2, Објектно-оријентисано програмирање 1, Објектно-оријентисано програмирање 2, Интелигентни системи, Интелигентни сервис и системи.

Током мастер и докторских студија објавио је као аутор или коаутор 12 научних радова на домаћим и међународним конференцијама, као и 2 рада у врхунским међународним часописима са импакт фактором (М21 и М22 категорије). Рецензирао је радове на конференцији ETRAN. Коаутор је уџбеника „Збирка решених испитних задатака из Програмирања 2“.

Учесник је на више међународних, научних и комерцијалних пројеката.

Изјава о ауторству

Име и презиме аутора: _____

Број индекса: _____

Изјављујем

да је докторска дисертација под насловом

- резултат сопственог истраживачког рада;
- да дисертација ни у целини ни у деловима није била предложена за стицање друге дипломе према студијским програмима других високошколских установа;
- да су резултати коректно наведени и
- да нисам кршио ауторска права и користио интелектуалну својину других лица.

У Београду,

Потпис аутора

Изјава о истоветности штампане и електронске верзије докторског рада

Име и презиме аутора: _____

Број индекса: _____

Студијски програм: _____

Наслов рада: _____

Број индекса: _____

Изјављујем да је штампана верзија мог докторског рада истоветна електронској верзији коју сам предао ради похрањивања у **Дигиталном репозиторијуму Универзитета у Београду**.

Дозвољавам да се објаве моји лични подаци везани за добијање академског назива доктора наука, као што су име и презиме, година и место рођења и датум одбране рада.

Ови лични подаци могу се објавити на мрежним станицама дигиталне библиотеке, у електронском каталогу и у публикацијама Универзитета у Београду.

У Београду,

Потпис аутора

Изјава о коришћењу

Овлашћујем Универзитетску библиотеку „Светозар Марковић“ да у Дигитални репозиторијум Универзитета у Београду унесе моју докторску дисертацију под насловом:

која је моје ауторско дело.

Дисертацију са свим прилозима предао сам у електронском формату погодном за трајно архивирање.

Моју докторску дисертацију похрањену у Дигиталном репозиторијуму Универзитета у Београду и доступну у отвореном приступу могу да користе сви који поштују одредбе садржане у одабраном типу лиценце Креативне заједнице (*Creative Commons*) за коју сам се одлучио.

1. Ауторство (CC BY)
2. Ауторство – некомерцијално (CC BY-NC)
3. Ауторство – некомерцијално – без прерада (CC BY-NC-ND)
4. Ауторство – некомерцијално – делити под истим условима (CC BY-NC-SA)
5. Ауторство – без прерада (CC BY-ND)
6. Ауторство – делити под истим условима (CC BY-SA)

(Молимо да заокружите само једну од шест понуђених лиценци.

Кратак опис лиценци је саставни део ове изјаве).

У Београду,

Потпис аутора

1. **Ауторство.** Дозвољавање умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце, чак и у комерцијалне сврхе. Ово је најслободнија од свих лиценци.
2. **Ауторство – некомерцијално.** Дозвољавање умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце. Ова лиценца не дозвољава комерцијалну употребу дела.
3. **Ауторство – некомерцијално – без прерада.** Дозвољавање умножавање, дистрибуцију и јавно саопштавање дела, без промена, преобликовања или употребе дела у свом делу, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце. Ова лиценца не дозвољава комерцијалну употребу дела. У односу на све остале лиценце, овом лиценцом се ограничава највећи обим права коришћења дела.
4. **Ауторство – некомерцијално – делити под истим условима.** Дозвољавање умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце и ако се прерада дистрибуира под истом или сличном лиценцом. Ова лиценца не дозвољава комерцијалну употребу дела и прерада.
5. **Ауторство – без прерада.** Дозвољавање умножавање, дистрибуцију и јавно саопштавање дела, без промена, преобликовања или употребе дела у свом делу, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце. Ова лиценца дозвољава комерцијалну употребу дела.
6. **Ауторство – делити под истим условима.** Дозвољавање умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце и ако се прерада дистрибуира под истом или сличном лиценцом. Ова лиценца дозвољава комерцијалну употребу дела и прерада. Слична је софтверским лиценцама, односно лиценцама отвореног кода.