Зборник радова Факултета техничких наука, Нови Сад

UDK: 004.65:61
DOI: https://doi.org/10.24867/28RB04Miljus

ДЕТЕКЦИЈА СТЕНОЗА И ОКЛУЗИЈА КОРОНАРНИХ АРТЕРИЈА

DETECTION OF STENOSIS AND OCCLUSION OF CORONARY ARTERIES

Сара Миљуш, *Факултет техничких наука, Нови Сад*

Област – БИОМЕДИЦИНСКО ИНЖЕЊЕРСТВО

Кратак садржај – *Рад се заснива на детекцији стеноза и оклузија коронарних артерија срца, као и проналажење њиховог процента зачепљења, уз помоћ алгоритама машинског учења, односно YOLOv4, YOLOv8 алгоритама и Detectron2 библиотеке. База података која је коришћена за израду пројекта састоји се од 3.850 слика коронарних артерија гдје су јасно обиљежене стенозе и/или оклузије.*

Кључне ријечи: *стенозе и оклузије коронарних артерија, коронарна ангиографија, неуралне мреже, дубоко учење, YOLOv4 и YOLOv8 алгоритми*

Abstract – *The paper is based on the detection of stenoses and occlusions of the coronary arteries of the heart, as well as finding their percentage of blockage, using machine learning algorithms, namely YOLOv4, YOLOv8 algorithms, and the Detectron2 library. The dataset used for the project consists of 3.850 images of coronary arteries where stenoses and/or occlusions are clearly marked.*

Keywords: *stenosis and occlusion of coronary arteries, coronary angiography, neural networks, deep learning, YOLOv4 and YOLOv8 algorithms*

1. УВОД

Аортна стеноза је стање срца у којем се аортни вентил срца сужава, спријечавајући његово потпуно отварање. Ако се стеноза погорша, то може довести до потпуног зачепљења (оклузије) артерије. Стенозе и оклузије коронарних артерија дијагностиковане су уз помоћ коронарне ангигорафије, која подразумијева примјену рендгенских зракова зарад приказивања коронарних артерија.

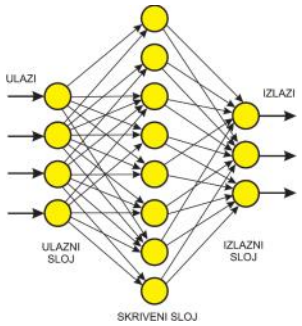
2. НЕУРОНСКЕ МРЕЖЕ И ДУБОКО УЧЕЊЕ

Неуронске мреже представљају сложене рачунарске системе настале повезивањем вјештачких неурона, инспирисане начином функционисања неурона у људском мозгу. Основа сваке неуронске мреже је вјештачки неурон. Неуронска мрежа (слика 1) се састоји од већег броја неурона, који су организовани у два или више слојева.

Први слој је увијек улазни, а последњи излазни. Ако их има више, онда се унутрашњи слојеви зову скривени слојеви.

НАПОМЕНА:
Овај рад проистекао је из мастер рада чији ментор је био др Платон Совиљ, ред. проф.

Начин организације ових слојева и њиховог повезивања чини архитектуру или топологију неуронске мреже.



Слика 1. Начин рада неуронске мреже

Последња развијена грана машинског учења је дубоко учење, која се користи за опис скупа алгоритама заснованих на концепту неуронских мрежа.

Најзначајнија достигнућа дубоког учења су класификације слика, детекција и локализација објеката, као и семантичка сегментација и сегментација инстанци.

Са друге стране, дошло је до невјероватног напретка у рјешавању проблема превођења текста, као и препознавања и синтезе говора, што је омогућило ефикасну гласовну комуникацију са паметним уређајима.

3. *YOLOv4* И *YOLOv8* АЛГОРИТМИ

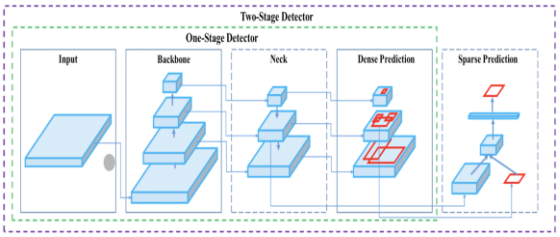
YOLOv4 и *YOLOv8* су алгоритми чија је намјена детекција објеката, широко примјењиван у компјутерској визији. Сама детекција објеката заснива се на детекцији инстанци објеката различитих класа на сликама и видео снимцима. *YOLO* алгоритми се састоје од два дијела: првог, конволутивне неуралне мреже (енг. *CNN*), која се користи за предвиђање координата *bounding box*-а за објекте у сликама и видео снимцима и регресиони модел који се користи за одређивање врсте објеката унутар самог *bounding box*-а.

Архитектура *YOLOv4* алгоритма

YOLOv4 користи једноставну архитектуру која се састоји од неколико компоненти (слика 2). На самом почетку имамо улаз (енг. *Input*) који представља сет слика за тренинг. Ове слике се обрађују у групама које са називају *Batches*, а сама обрада и процесирање врши се на графичкој картици (енг. *GPU*).

Следећи степен алгоритма јесу *Backbone* и *Neck*, компоненте задужене за екстракцију и агрегацију обиљежја. На крају налази се главни дио алгоритма или *Head* који је задужен за саму детекцију/предикцију. Када је у питању *Backbone* алгоритам

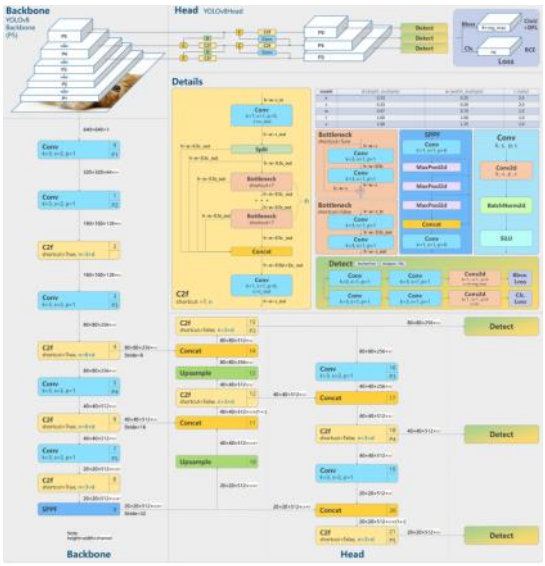
говоримо о конволуцијској неуралној мрежи која представља темељ алгоритма.



Слика 2. Архитектура YOLOv4 алгоритма

Архитектура *YOLOv8* алгоритма

YOLOv8 алгоритам користи мрежу са три гране за детекцију објеката у три различите резолуције како би омогућио брзу и прецизну детекцију објеката у стварном времену. Његову архитектуру (слика 3) чине следећи кораци: улазна слика пролази кроз конволуцијску мрежу како би се извукле карактеристике слике, излаз слике се дијели на три гране, од којих свака одређује карактеристику слике у другој резолуцији, свака грана мреже затим пролази кроз неколико слојева конволуције како би се детектовали објекти у различитим дијеловима слике, након што су објекти детектовани у свакој грани мреже спајају се у јединствену листу детектованих објеката, листа затим пролази кроз поступак *NMS*-а (енг. *Non-Maximum Suppression*) како би се уклонили дупликати и детекције које нису поуздане и коначни резултат је листа објеката која садржи класу објеката, његову тачну позицију на слици и степен поузданости детекције.



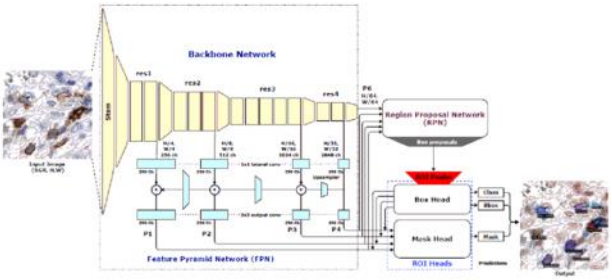
Слика 3. Архитектура YOLOv8 алгоритма

4. *DECETRON2* БИБЛИОТЕКА

Detectron2 је отворена библиотека за дубоко учење коју је развио *Facebook AI Research* за детекцију објеката и сегментацију слика.

Главни кораци код *Decetron2* архитектуре (слика 4):

1. Предпроцесирање: Улазна слика се нормализује на фиксну величину.
2. Слика пролази кроз мрежу за екстракцију карактеристика како би се извукле карактеристике слике.
3. *FPN* (енг. *Feature Proposal Network*): Генерисане карактеристике од стране претходне мреже пролазе кроз мрежу *FPN*, која генерише пирамиду вишескалних мапа обиљежја које се користе за детекцију објеката.
4. *RPN* (енг. *Region Proposal Network*): Генерише предлоге објеката користећи мрежу *RPN*. Ови предлози су скуп граничних оквира који вјероватно садрже објекте.
5. *Rol Align*: Ова операција омогућава мрежи да комбинује карактеристике из региона од интереса различитих величина, без губитка информација.
6. *Rol Head*: Обезбјеђују се ознаке класа и помијерање граничних оквира за сваки *Rol*.
7. Постпроцесирање: Предвиђени оквири се филтрирају помоћу *NMS* алгоритма да би се уклонили преклапајући оквири и задржала само најпоузданија предвиђања.



Слика 4. Архитектура Detectron2 библиотеке

5. БАЗА ПОДАТАКА (енг. *Dataset*)

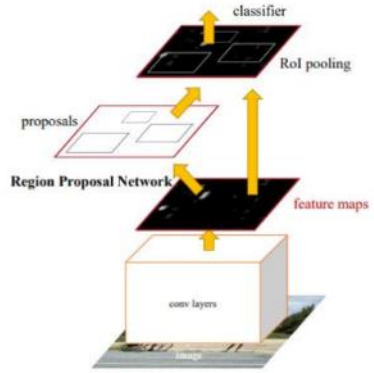
База података која је коришћена за израду задатка састоји се од 3.850 слика коронарних артерија гдје су јасно обиљежене стенозе и/или оклузије, а чини је 11 класа: *stenosis*10, *stenosis*20, *stenosis*30, *stenosis*40, *stenosis*50, *stenosis*60, *stenosis*70, *stenosis*80, *stenosis*90, *stenosis*99 и occlusion100, гдје број поред назива одређене класе представља проценат зачепљења артерије. У скупу за обуку налази се 80% базе, док скуп за тестирање и валидацију садржи по 10% података. Подијела је извршена помоћу *Python* скрипте *process.py*, која генерише три текстуална фајла: *train.txt*, *valid.txt* и *test.txt*, у којима су смјештене путање до слика.

Код *YOLO* алгоритама анотације су представљене у *YOLO* формату, складиштене у текстуалним датотекама које се налазе у директорију као и слике. Анотације код *Detectron2* библиотеке су представљене у *COCO* формату, гдје се чувају у *JSON* датотекама и укључују информације о класама објеката на слици, границама објеката и маскама сегментације ако су доступне.

5.1. Конфигурисање алогитама

Прије самог тренинга неопходно је дефинисати одређене параметре који знатно утичу на квалитет модела као што су број епоха, величина група, стопе учења и архитектура модела. Све ове информације се обично задају путем конфигурацијских датотека које се користе за тренирање модела у *Darknet* библиотеци.

Што се тиче *YOLO* алгоритама првенствено је потребно подесити *batch size*, јер прецизира колико слика прослеђујемо алгоритму у једној итерацији тренинга, а он је у нашем случају он је 64. Наредни хиперпараметар јесте *max batches*, који представља укупан број итерација и у већини случајева се задаје по формули: (број класа*2000) с тим да се не препоручује број мањи од укупног броја слика у скупу за тренинг и не мање од 6000. *Subdevision* може бити 16, 32 или 65 и дијели *batch* на мање дијелове. *Filters* представља (број класа+5)*3, а *Steps* = а,б, гдје је а = 80% од *max batches*, а б = 90% од *max batches*.
Код *Decetron2* кључан је *Faster R-CNN* алгоритам (слика 5). Њему су претходила два модела детектовања објеката *R-CNN* и *Fast R-CNN*.



Слика 5. Архитектура Faster R-CNN-а

5.2. Тренирање и евалуација модела

Када су у питању *YOLO* алгоритми модел је трениран и тестиран уз помоћ *Azure Data Science Virtual Machine* (енг. *DCVM*), виртуелној машини постављеној на *Linux-Ubuntu* 18.04 оперативном систему, са преинсталираним библиотекама и софтвером неопходним за *Data Science*/AI/*Machine Learning*.
Главни параметри који су пропраћени током тренинга су: *Mean Average Precision* (енг. *mAP*), средња просјечна прецизност је метрика која се користи за евалуацију модела детекције објекта и просјечни губитак (енг. *Average Loss*) који представља мјеру квалитета модела и рачуна се као разлика између предвиђених и тачних вриједности за координате *bounding box*-а. Што су губици мањи, алгоритам је квалитетнији.
Команда за покретање тренинга код *YOLOv4* алгоритма приказана је на слици 6.

```
./darknet detector train data/obj.data cfg/yolov4-custom.cfg
yolov4.conv.137 -dont_show -map
```

Слика 6. Команда за покретање обуке код YOLOv4 алгоритма

Гдје се параметар *dont_show* (не приказати) може отклонити, како би се видио графикон напретка *mAP* у односу на итерације. Команда *yolov4.conv.137* се односи на 137 конволутивни слој унутар *YOLOv4* архитектуре. *Obj.data* фајл садржи: број класа, путање до *train.txt, valid.txt, test.txt, obj.names* (смјештена су имена свих класа) и путању до фолдера у којем се смјештају резултати након тренирања. Конфигурациона датотека, *yolov4-custom.cfg*, дефинише архитектуру мреже и остале хиперпараметре.
YOLOv8 садржи исте хиперпараметре као *YOLOv4* које је потребно подесити у зависности од потребе задатка. Направљен је фајл под називом *data* у којем су смјештене путање до слика изабраних за обуку и одговарајуће датотеке са ознакама (координате оквира анотираних стеноза и/или оклузија, ознаке и број класа). Модел који се користио за обуку је *yolov8x.pt*, а комада којом се покреће обука је дата на слици 7.

```
yolo train data=data.yaml model=yolov8x.pt epochs=100 imgsz=512
```

Слика 7. Комада за покретање обуке код YOLOv8 модела

Величина слике подешена је на 512, као и код *YOLOv4* модела. Модел ће бити обучен кроз 100 епоха. Свака епоха представља један пролазак кроз цио скуп података за обуку, са циљем да се побољша перформанса модела током времена.
Што се тиче *Detectron2* модел је трениран и тестиран уз помоћ *X2Go* софтвера, програм отвореног кода који користи протокол *NX Technology*. Ради олакшаног рада, односно компатибилности свих неопходних библиотека које су нам потребне за израду задатка користи се *docker container*. За обуку модела коришћена је *Python* скрипта *trainStenosis.py*, која прима одговарајуће аргументе.

5.3. Тестирање модела

Команда којом тестирамо модел код *YOLOv4* алгоритма приказана је на слици 8, неопходно је прије тестирања модела поставити параметре *batch* и *subdivisions* на јединицу.

```
./darknet detector test data/obj.data cfg/yolov4-custom.cfg
../training/yolov4-custom_best.weights
../../mask_test_images/image1.jpg -thresh 0.3
```

Слика 8. Команда за тестирање YOLOv4 модела

Гдје датотека *yolov4-custom_best.weights* представља тежине најбољег модела обученог на датом скупу података, а *mask_test_images* фајл гдје су смјештене слике за тестирање модела. *Thresh* представља минималну тачност потребну за детекцију објеката.
Тестирање модела код *YOLOv8* алгоритма приказана је на слици 9, гдје *yolov8x_best.pt* представља тежине најбољег обученог модела, *source* је путања до слике коју желимо да тестирамо, а *conf* је параметар који одређује минималну вриједност повјерења коју детектор објекта мора имати да би прихватио детекцију као тачну.

```
yolo predict model=yolov8x_best.pt source='https://ultralytics.com/images/test1.jpg' conf=0.35
```

Слика 9. Команда за тестирање YOLOv8 модела

Тестирање модела код *Detectron2* библиотеке је извршено уз помоћ *Python* скрипте *useStenosis.py*, гдје терминалским позивањем скрипте започиње процес тестирања над задатим сликама.

6. Резултати

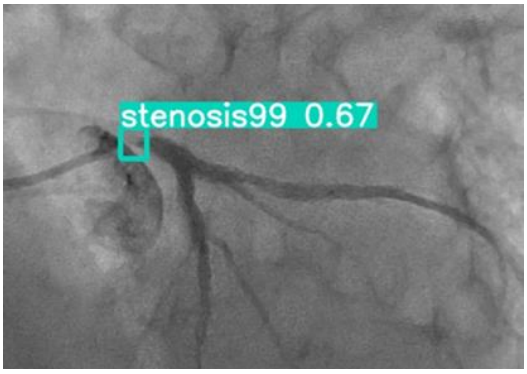
Резултати *YOLO* алгоритама

Након завршеног тренинга, који обично траје око један дан, тестиран је модел. Резултати који су добијени нису били задовољавајући, посебно код пацијената који су имали више стеноза. Проблеми са

којима смо се сусретали: алгоритму је теже било препознати стенозе са мањим процентом зачепљења, алгоритам ништа није препознао, детекција одговарајуће стенозе, али на погрешном мјесту, детекција стенозе на одговарајућем мјесту, али погрешан проценат зачепљења.

Исти кораци су урађени и када је у питању *YOLO8*, добијени су мало бољи резултати, са грешкама које су се понављале. Најчешће се дешавало да алгоритам не препозна ниједну стенозу и/или оклузију. Код квалитетнијих слика обично се дешавало да алгоритам препозна стенозу са већим процентом зачепљења и то на почетку, у главном дијелу артерије.

На слици 11 детектована је стеноза са зачепљењем од 99%, док остале три стенозе са зачепљењем од 50% нису детектоване. Број 0,67 представља вјероватноћу са којом је детектована стеноза.

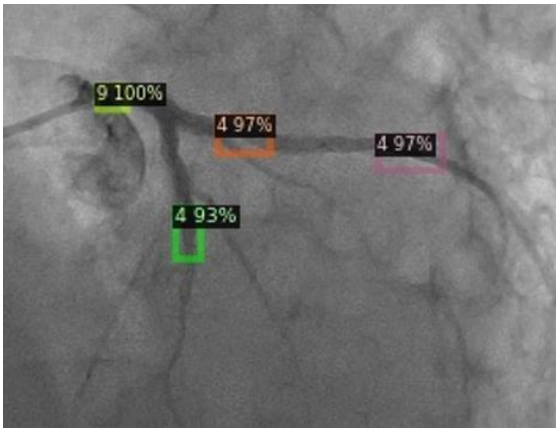


Слика 10. Детекција стенозе99 уз помоч YOLOv8 алгоритма

Резултати *Detectron2* библиотеке

Тренинг код *Detectron2* је трајао дуже у односу на *YOLO* алгоритме, али су добијени знатно бољи резултати. Проблеми са којима смо се сусретали: ни у једном случају није препозната оклизија, детекција стенозе на мјесту на којем је нема, детекција стенозе на одговарајућем мјесту, али погрешан проценат зачепљења, алгоритам не детектује све присутне стенозе, детекција стенозе унутар стенозе.

И код *Detectron2* библиотеке и *YOLO* алгоритама грешке су зависиле и од квалитета слике. На слици 11 представљен је случај у којем је детектована свака стеноза, са великом вјероватноћом детекције.



Слика 13. Детекција стеноза уз помоћ Detectron2 библиотеке

7. ЗАКЉУЧАК

У овом раду је представљена примјена алгоритама дубоког учења за детекцију стеноза и/или оклузија коронарних артерија, као и процента њиховог зачепљења. За постизање одговарајућег циља, упоређене су перформансе три различита алгоритма за детекцију објеката: *YOLOv4*, *YOLOv8* и *Detectron2* библиотеке. У зависности од потребе овог пројекта *Detectron2* библиотека даје најуспјешније резултате. Крајњи исход показује могућност примјене алгоритама машинског учења и у сфери медицинских наука, али је за њихову стварну примјену у пракси потребно додатно побољшати резултате.

Један од кључних изазова, поред осталих проблема, био је тај што алгоритам ниједном није успио препознати оклузију, што указује на потребу за даљим побољшањем алгоритма детекције. Важно је напоменути да алгоритам детекције посматра слике појединачно, док лијекар за откривање стеноза и/или оклузија и њиховог процента зачепљења посматра ангиографски снимак из свих седам углова истовремено.

8. ЛИТЕРАТУРА

1. https://epoteka.rs/blog/aortna-stenoza/
2. https://www.ncbi.nlm.nih.gov/books/NBK560507/
3. https://www.stetoskop.info/bolesti-srca-i-krvnih-sudova-kardiologija/koronarna-angiografija
4. http://solair.eunet.rs/~ilicv/neuro.html
5. file:///C:/Users/milju/Downloads/Milosavljevic_Duboko_ucenje%20.pdf
6. https://blog.roboflow.com/a-thorough-breakdown-of-yolov4/
7. https://blog.roboflow.com/whats-new-in-yolov8/
8. https://medium.com/@hirotoschwert/digging-into-detectron-2-47b2e794fabd
9. https://www.v7labs.com/blog/mean-average-precision
10. https://nardus.mpn.gov.rs/bitstream/handle/123456789/21403/Disertacija_13527.pdf?sequence=1

Кратка биографија:

Сара Миљуш рођена је у Никшићу, 1999. године. Дипломирала је на факултету техничких наука у Новом Саду, на Катедри за електрична мјерења 2022. године.

контакт: miljussara2@gmail.com