|  |  |  |
| --- | --- | --- |
|  | Universitatea Politehnică din Timișoara  Facultatea de Automatică și Calculatoare  Specializarea Ingineria Sistemelor |  |

**Identificarea nucleelor anormale cu potențial cancerigen în testul Papanicolau**

|  |  |
| --- | --- |
| Coordonator  ȘL.dr.inf. Cristian Zimbru | Absolvent  Emil Cioplea |

Timișoara

2022

**Cuprins**

[Introducere](#_gjdgxs) 3

[Domeniul proiectului](#_30j0zll) 3

[Specificațiile generale ale aplicației](#_1fob9te) 4

[Obiective](#_3znysh7) 4

[Detalii medicale](#_2et92p0) 5

[Aplicații asemănătoare](#_tyjcwt) 8

[Augmented Reality Microscope for Cancer Detection](#_3dy6vkm) 8

[Breast cancer detection by leveraging Machine Learning](#_4d34og8) 9

[Early detection of breast cancer using svm classifier technique](#_2s8eyo1) 10

[Detection and classification the breast tumors using mask R-CNN on sonograms](#_17dp8vu) 11

[Comparație Aplicații](#_3rdcrjn) 12

[Fundamente teoretice](#_26in1rg) 13

[Rețele neuronale](#_lnxbz9) 13

[CNN ( Convolution Neural Network )](#_35nkun2) 16

[Tensor](#_1ksv4uv) 17

[Rețele de tip SSD ( Single Shot Detector )](#_7kn94k1ddd7n) 17

[Tehnologii folosite](#_44sinio) 18

[Python](#_2jxsxqh) 18

[TensorFlow](#_z337ya) 18

[Tensorflow Zoo](#_3j2qqm3) 18

[Keras](#_1y810tw) 19

[TensorBoard](#_4i7ojhp) 19

[LabelImg](#_2xcytpi) 20

[RobotFlow](#_1ci93xb) 20

[Pașii făcuți pentru antrenarea inteligenței artificiale](#_3whwml4) 21

[Pregătirile necesare pentru antrenare](#_2bn6wsx) 21

[Dezvoltarea de model tip SSD](#_qsh70q) 22

[Utilizarea unei noi arhitecturi de model. (CenterNet)](#_ck66ri6jngc8) 36

[Concluzii](#_1pxezwc) 40

[Viitoare îmbunătățiri](#_49x2ik5) 41

[Bibliografie](#_2p2csry) 42

# Introducere

## Domeniul proiectului

Proiectul va fi orientat pe utilizarea **inteligenței artificiale** pentru detectarea **celulelor cu cancer** din imagini captate la nivel microscopic la probe recoltate de la pacienți.

Cancerul este una dintre cele mai periculoase boli ale vieții moderne, afectând numeroase persoane. Mereu s-a căutat un “remediu al cancerului”, dar nu este atât de ușor, ținând cont de variabilitatea si dinamica acestuia. Cancerul reprezintă multiplicarea incontrolabilă a unui tip de celule, acestea împotrivindu-se procesului fiziologic de apoptoză, distrugere celulară care se întâmplă atunci când unele dintre celulele noastre îmbătrânesc și încep să aibă disfuncții metabolice sau fiziologice. Aceste celule refuză să moară și devin nocive pentru corpul care le găzduiește, profitând de resursele corpului fără să execute cum trebuie funcțiile unei celule, aducând corpul la epuizare.

În pozele cu care se va lucra în cadrul acestui proiect celulele bolnave de cancer se vor diferenția față de celule normale cu următoarele caracteristici vizibile, menționate în articolul[1]:

* Mărimea și formele celulelor. Celule bolnave pot fi neobișnuit de mari sau neobișnuit de mici. Forma unei celule normale este dictată de funcția pe care o îndeplinește. O celulă cu cancer de obicei nu îndeplinește nicio funcție așa că este liberă să aibă orice formă care îi este convenabilă.
* Mărimea și culoarea nucleului. Majoritatea celulelor cancerigene au un nucleu mai negru, hiperpigmentat din cauza condensării de cromatină, material genetic, și cu un nucleu mai mare, ambele fiind cauzate de depozitarea în excess a *ADN*-ului.
* Aranjamentul celulelor reflectă de obicei funcțiile grupării celulare. Celulele cancerigene pentru că de obicei nu respectă funcția nu păstrează arhitectura tisulară care este de așteptat pentru fiecare tip celular.

Criteriile de mai sus pot fi apreciate de ochii antrenați ai doctorilor, dar ideea acestui proiect este să antreneze o rețea neuronală care să fie capabilă să identifice pentru o a doua părere sau în cazul în care există un număr prea mare de mostre pentru a fie analizate individual de către clinician.

Inteligență artificială este corelată de obicei cu următoarele expresii: “*Machine learning*”, “*Neural network*”, “*Deep learning*”. Așa cum sa precizat în [2], într-o conversație normală acești termeni pot fi folosiți interschimbabil, dar totuși fiecare înseamnă ceva diferit.

* Inteligență artificială: Este un termen care este folosit pentru a descrie capacitatea unui dispozitiv de a imita inteligența umană.
* *Machine learning*: Este un subset al inteligenței artificiale. Se urmărește optimizarea/perfecționarea procesului desfășurat de către inteligența artificială.
* *Neural network*: Este un subset al conceptului de machine learning. La un nivel de bază încearcă să imite neuronii pe care îi poți găsi în creierul uman. Este compus din patru componente: *input*, *weights*, *bias* și *output*.
* *Deep learning*: Este un *neural network* care presupune cel puțin 3 straturi de neuroni..

Rețea neuronală convoluțională este o categorie specifică de rețea neuronală care este folosită de obicei pentru *Image/Video recognition*. O imagine este alcătuită din 3 tablouri bidimensionale ( dacă este imagine RGB ), un tablou pentru R, unul pentru G, altul pentru B, fiecare conține valori între 0-255. 0 reprezentând o culoare inexistentă pentru acel pixel, 255 reprezentând o valoare maximă. Din cele 3 tablouri se aplică diferite *kernel*-uri și *pool-ing* pentru a aduce *input*-ul la un simplu *array* care va fi preluat de către rețeaua neuronală.

## Specificațiile generale ale aplicației

Aplicația presupune o rețea neuronală care preia ca input imagini (obținute din baza de date [3]) la nivel microscopic al țesutului pacientului. *Output*-ul va fi o descriere specifică a mostrei, precum și identificarea numărului de celule detectate, dacă s-au detectat celule anormale cu potențial cancerigen. În cazul în care s-au găsit astfel de celule, aplicația va preciza informații utile, precum raportul dintre numărul de celule bolnave și numărul de celule totale.

Se vor depozita imaginile cu probele de țesuturi într-un folder și toate acestea vor fi parcurse pe rând de către rețeaua neuronală. Rezultatele vor fi scrise într-un fișier pentru o citire ușoară și fiecare imagine analizată poate fi inspectată individual după rularea programului.

Implementarea se va realiza în *Python* și va utiliza bibliotecile *TensorFlow* și *Keras*.

## Obiective

* Antrenarea unei rețele neuronale capabile să identifice celulele anormale cu risc cancerigen
* Realizarea unui studiu despre ce trebuie făcut ca o rețea neuronală să funcționeze bine
* Compararea performanțelor a mai multor rețele neuronale

## Detalii medicale

Una dintre cele mai importante probleme de sănătate publică din România o reprezintă cancerul de col uterin. Incidenta este de trei ori mai mare decât media ratei din Uniunea Europeana, România având peste 4000 de femei diagnosticare și o înregistrare a peste 2000 de decese din această cauză în fiecare an. [4]

Cancerul cervical apare la nivelul epiteliul scuamos al colului uterin. Inițial, se prezintă ca o leziune neinvazivă numită neoplazie intraepitelială cervicală (CIN, cervical intraepithelial neoplasia). Această etapă precanceroasă are de obicei un proces evolutiv lung și se caracterizează prin tumori care nu au invadat încă membrana bazală. [5]

Criteriile de benignitate ale unei formațiuni tumorale care se pot identificat pe un frotiu sunt:

1. Cromatina subțire și distribuită uniform
2. Dimensiunea si forma nucleului în limitele fiziologice normale
3. Membranele nucleare uniforme
4. Nucleii normocromatici și intensitatea colorării menținută în toate celulele
5. Mitozele apar de regulă numai la nivelul stratului bazal, astfel că sunt rareori observate în frotiul cervical (reparație/regenerare)

Criteriile de neoplazie prin care se certifica statutul unor formațiuni tumoare sunt:

1. Cromatina cu un aspect grosier, prezintă aglomerări și este distribuită neuniform;
2. Nucleii prezintă devieri în dimensiune și formă în cadrul aceleiași populații de celule;
3. Membranele nucleare au un contur neregulat și prezintă o adâncitură și/sau îngroșare; datorită marginației și aglomerării cromatinei;
4. Nucleii sunt, de obicei, hipercromatici și colorarea variază de la un grup la altul;
5. Multinuclearea are loc frecvent, ceea ce reflectă o fluctuație rapidă a celulelor;
6. Pierderea coeziunii (celulele părăsesc grupurile lor). Nucleii tind să se adune în straturi/grupuri;
7. Figurile mitotice pot fi observate în toate straturile epiteliului.

Infecția cu virusul papilomului uman (HPV), în special tipurile 16 și 18 care sunt oncogene, este o cauză stabilită și frecventă a apariției cancerului de col uterin la femeile active sexual; mai recent, această infecție fost asociată și cu o creștere a incidenței cancerelor capului și gâtului în funcție de locul de inoculare al virusului. [6]

Deși există un vaccin eficient împotriva acestor tulpini oncogene ale virusului papilomului uman, timpul de latență dintre infecție și apariția bolii poate ajunge la ordinul a 40- 50 de anii, ceea ce face ca screening-ul să fie imperios necesar. Diagnosticarea și tratarea eficientă a modificărilor mailgine pre-invazive cu risc ridicat reduce incidența și mortalitatea prin cancer cervical.

Testul de screening al cancerului de col uterin (frotiu citodiagnostic) cunoscut sub numele de Test Papanicolau este ieftin si sigur. Acest test constă în recoltarea unor probe de la nivelul colului uterin, care sunt introduse imediat după prelevare întru-un mediu lichid special. Ulterior, se realizează etalarea într-un singur strat a celulelor dispersate din lichid, colorarea si interpretarea lamei de către un medic anatomopatolog. Acest test necesită un citolog bine instruit pentru a identifica modificările precoce (discarioză și neoplazie intraepitelială cervicală, CIN). Cu toate acestea, progresele în citologia lichidă și testarea concomitentă pentru ADN de HPV pot îmbunătăți rezultatele. [7]

Rezultatele anormale ale testului Papanicolau sunt variate, printre cele mai frecvente din punct de vedere al rate de incidență sunt:

* Celule scuamoase atipice de semnificație nedeterminată (ASC-US), includ anomalii morfologice ale celulelor cervicale de tipul multinuclearității, paracheratozei, discheratozei și hipercheratozei, moficicări ce pot fi observate în cazul infecției cu HPV, dar acestea nu sunt specifice în cazul absenței koilocitozei.
* Leziunei intraepiteliale scuamoase de grad scăzut (LSIL). Aceste modificări sunt situate în treimea inferioară a epiteliului scuamos de acoperire, iar celulele cervicale prezintă modificări care sunt ușor anormale.
* Leziuni intraepiteliale scuamoase de grad înalt (HSIL), constituie modificările epiteliului scuamos situate în două treimi superioare ale grosimii epiteliului scuamos de acoperire. Acest tip de leziuni sugerează modificări mai intense ale colului uterin decât subtipurile prezentate anterior și există o asociere mai frecventă între acest tip de leziune și evoluția celulelor cervicale către precancer, respectiv cancer. Din punct de vedere morfologic, un citolog poate observa pe frotiul cervical celule discariotice imature, cu o maturare epitelială incompletă, în care raportul Nucleu/Citoplasmă este mare, din cauza dimensiunii reduse a citoplasmei și măririi anormale a nucleului.

Cele mai frecvente erori in realizarea testului de frotiu citodiagnostic țin atât de colectarea incorectă a celulelor anormale ale colului uterin, neincluzând leziunile precanceroase. De asemenea, există erori și de interpretare, ce țin de experiența, atenția și nivelul de antrenament al medicului anatomopatolog. [8]

The Clinical Laboratory Improvement Amendments din anul 1988 recomandă o reinterpretare a 10% din probele negative, drept o măsură de precauție. Acest rescreening poate să fie realizat pe o rata de probe negative mult mai mare prin intermediul tehnologiilor computerizate. FDA ( Food and Drug Administration), organismul care se ocupă cu reglementarea alimentației (umană și animală), a suplimentelor dietetice, [medicamentelor](https://ro.wikipedia.org/wiki/Medicament) (umane și animale), cosmeticelor, [dispozitivelor medicale](https://ro.wikipedia.org/w/index.php?title=Dispozitive_medicale&action=edit&redlink=1) (umane și animale) și a dispozitivelor care emit [radiații](https://ro.wikipedia.org/wiki/Radia%C8%9Bie) (sunt incluse dispozitivele non-medicale), biologice, și a produselor pentru [sânge](https://ro.wikipedia.org/wiki/S%C3%A2nge), aprobă astfel doua sisteme: unul care are la bază un algoritm (AutoPap Qc System, TriPath Imaginig, Inc, Redmond, Washington) și unul care se bazează pe o rețea neurală ( PAPNET, Neuromedical Systems, Inc., Suffern, New York).

PAPNET utilizează o rețea neurală computerizată a imaginilor frotiurilor de citodiagnostic pentru a identifica celulele sau grupările celulare care au nevoie de interpretare. Această tehnologie identifică și afișează până la 128 de imagini per slide care au probabilitatea de a conține celule anormale. Un citolog revizuiește aceste imagini și decide dacă este necesară analiza frotiului cu ajutorul unui microscop optic cu lumina.

AutoPap constă într-o tehnologie capabilă să ia decizii pe baza unui algoritm care identifică probele care depășesc un anumit prag de probabilitate a celulelor modificate. Laboratoarele pot seta diferite praguri, corespunzând unui procent de 10%, 15% sau 20% de celule anormale din populația prezentă pe frotiu. În contrast cu rescreening-ul manual, AutoPap selectează o gamă variată de frotiuri pentru citodiagnostic cu cele mai multe modificări celulare, incluzând astfel și probele care au fost omise la interpretarea manuală.

# Aplicații asemănătoare

## Augmented Reality Microscope for Cancer Detection [9]

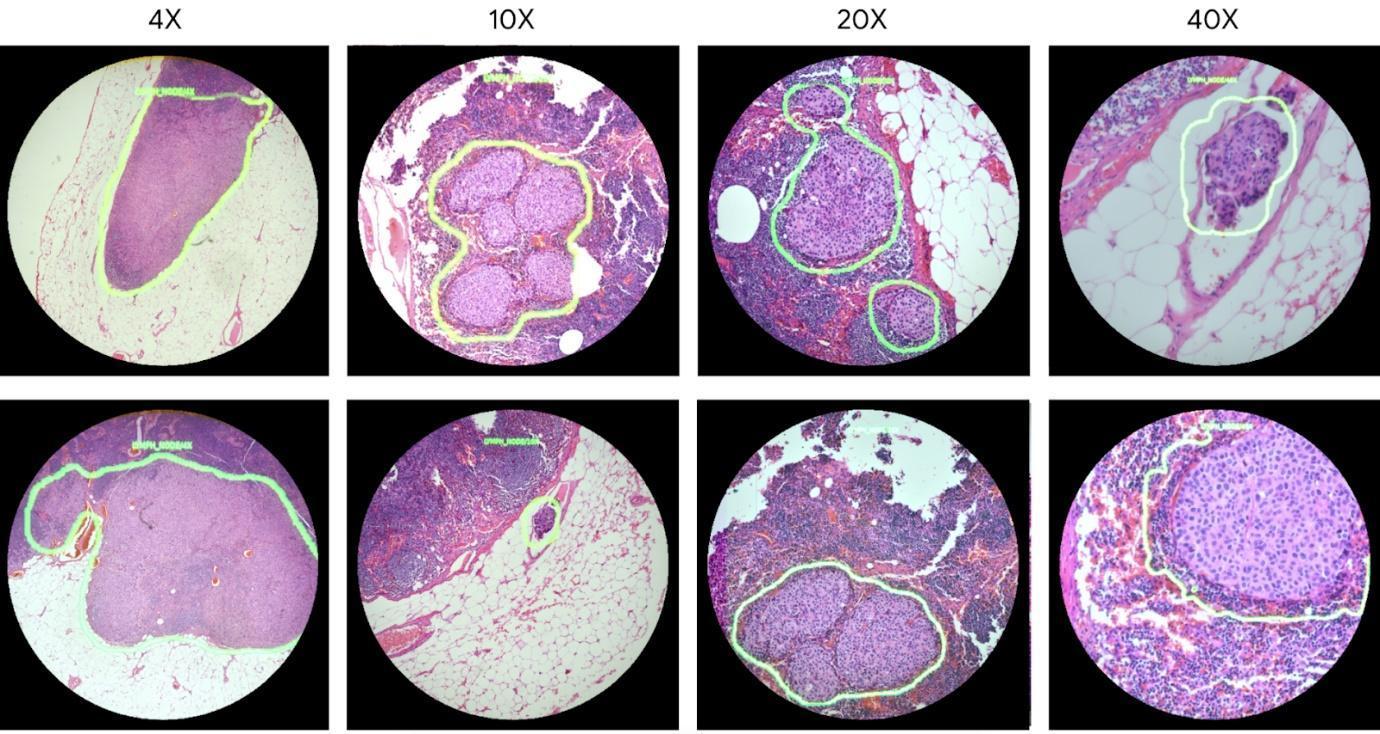


Fig. 2.1 Zone evidențiate cu cancer pentru cancerul la sân și prostată

Un proiect dezvoltat de Google care presupune analizarea imaginilor oferite de către microscop în *real time* (microscop augmentat), are ca scop ușurarea eforturilor depuse de medici, având o a doua părere în detecția cancerului. O cameră digitală implementată în microscop observă același lucru pe care utilizatorul îl vede și trimite imaginile în timp real către o unitate de calculare care aplică rețeaua neuronală artificială. Rezultatul este afișat pe lentila prin care utilizatorul observă celulele.

Microscopul augmentat poate oferi mai multe indicații vizuale decât este arătat în figura 2.1., precum mesaje text, zone de căldură sau animații. Acesta poate rezolva mai multe tipuri de probleme prin intermediul conceptului de *machine learning,* precum detectarea de obiecte, cuantificarea sau clasificarea acestora. Performanțele realizate:

* Pentru analizarea lichidului limfatic a detectat 98% din aria cancerigenă.
* Pentru analizarea cancerului de prostată a detectat 96% din arie.

## Breast cancer detection by leveraging Machine Learning [10]

A close-up of some currency

Description automatically generated with low confidence

Fig. 2.2. Filtrele folosite de DNNs

Această aplicație constă într-un DNNS (*Deep Neural NetworkS*) care să detecteze cancerul la sân cu o precizie mai bună decât metodele populare (se pot observa în [4]). Metodele populare au o acuratețe și o precizie cuprinsă în intervalul 82.5% - 95.6%. Aplicația propusă a ajuns la o precizie de 97.9% și o acuratețe de 97.2%.

Data la care a fost folosită pentru a crea rețelele neuronale conține 8009 de imagini microscopice dintr-o varietate de țesuturi provenite de la 683 pacienți. Imaginile sunt clasificate în tumori grave sau tumori blânde, în funcție de agresivitatea tipului celular multiplicat. Imaginile preprocesate ale tumorilor li s-a aplicat metoda propusă, *Deep Neural Network with Support Value*.

Histograma este o reprezentare precisă a distribuirea valorilor numerice și *Fuzzy clustering,* o metodă care permite ca oricare punct de data să fie găsit într-un *cluster* sau în mai multe clustere.

## Early detection of breast cancer using svm classifier technique [11]

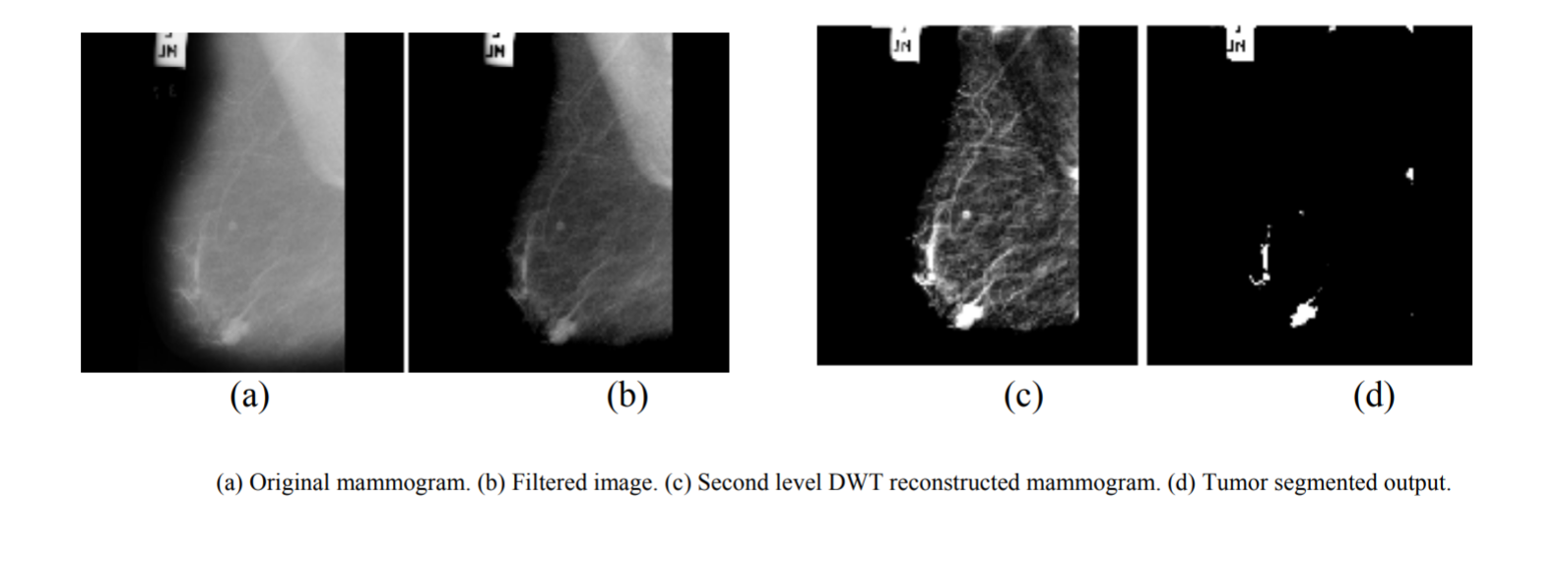


Fig.2.3. Evidențierea tumorilor

Această aplicație lucrează cu un input diferit față de ce am discutat până acum. Așa cum se poate observa în figura 2.3. intrarea constă în imagini provenite de la un mamograf, nu în imagini de frotiu a unor țesuturi biopsiate. Aplicația presupune aplicarea diferitelor filtre și după, analizarea imaginilor fragmentate cu *SVM classifier,* care corespunde cu tematica de identificare a cancerului utilizând *machine learning*.

1. Este imaginea originală scoasă de la mamograf.
2. Imaginea după ce a fost trecută printr-un filtru de netezire Gaussian bazat pe devierea standard.
3. Imaginea după aplicarea celui de-al doilea filtru, un filtru morfologic te tipul *top hat,* un filtru care extrage mici elemente și mici detalii dintr-o imagine. Acest filtru are rolul de a corecta iluminarea distribuită inegal atunci când fundalul este întunecat.
4. Imaginea care o ia ca input metoda propusă.

Imaginile îmbunătățite de la mamograf sunt convertite în imagini binare utilizând mai multe praguri de valori. Selectarea a mai multor praguri de valori este un pas important pentru că împarte imaginea în mai multe regiuni. Imaginile segmentate sunt filtrate din nou cu filtrul de netezire Gaussian pentru a scăpa de zgomot. Extragerea detaliilor este folosită pentru a măsura diferite precum: aria, centroidul, axa majoră a lungimii, axa minoră a lungimii, etc.

Metoda a ajuns la o sensitivitate de 88.75% pentru 50 de imagini provenite de la mamograf.

## Detection and classification the breast tumors using mask R-CNN on sonograms [12]

Această aplicație se bazează pe imagini obținute din scanare cu ultrasunete. A fost făcută această alegere pentru că este o metodă neinvazivă, în comparație cu biopsia care este invazivă.

A picture containing text

Description automatically generated

Fig.2.4 Poza ultrasonică inițială. (B) Poza conturată de radiologist. (C) Poza conturată de aplicație.

Pentru ca aplicația să poată desena conturul s-a folosit o subcategorie a rețelelor neuronale de convoluție numită *Mask R-CNN*. Această subcategorie este folosită pentru a detecta obiecte și generează o mască segmentată pentru fiecare instanță.

Studiul a fost realizat cu ajutorul imaginilor ultrasonice care au fost oferite în mare parte de către *China Medical University Hospital*. Un număr de 80 de cazuri compus din 307 de imagini ultrasonice. Ecografia a fost realizată de radiologisti folosind un ecograf cu ultrasunete GE (General Electric) de tip LOGIQ S8, cu un traductor care folosea frecvențe între 9-12 Mhz. Imaginea originală având o rezolutie de 960x720 pixeli.

Cele 307 de imagini au fost împărțite în 80% ca set de antrenament și 20% ca un set de validare. Aplicația a fost capabilă să identifice cu o precizie de 85% cazurile în care cancerul avea o formă ușoară și cazurile în care avea o formă gravă.

## Comparație Aplicații

Tabel 1 Comparație intre aplicații

|  |  |  |
| --- | --- | --- |
| Funcție | Precizie | Utilitate |
| 1. Augmented microscope | 96%-98% | Analizarea în real-time |
| 2. DNNS supported value | 98% | Analizarea biopsii |
| 3. SVM classifier | 88.75% | Analizarea pozelor de la mamograf |
| 4. R-CNN | 85% | Analizarea ecografiilor |

# Fundamente teoretice

## Rețele neuronale

O rețea neuronală la nivelul fundamental este constituită din mai mulți **perceptroni** interconectați.

Principala caracteristică a rețelelor neuronale este cea **holistică**. „Un sistem dobândește prin structură proprietăți noi față de cele ale obiectelor componente”[13]. Perceptronul este modelul matematic al unui neuron. La nivel quantic un perceptron arată astfel [14]:

Diagram

Description automatically generated

Fig.3.1. Perceptron

Perceptronul va fi activat în funcție de suma produselor dintre greutăți și valorile de intrare corespunzătoare. De exemplu avem: și . Se va calcula output-ul astfel:

= step (0.5 \* -0.5 + 0.2 \* 0.3 + 0.3 \* 1) = step(0.11)  
 => step(0.11) = 1 => out=1, adică neuronul a fost activat.  
În calcularea output-ul se mai ține cont și de bias.   
*Output*-ul unui neuron va fi dat cel mai probabil ca și *input* pentru unul sau mai mulți neuroni, de regulă, pentru neuronii care aparțin *layer*-ul viitor.

Deci o rețea neuronală este o **grupare de neuroni**. Fiecare neuron individual face **operații simple** pentru a da o informație mai departe către alți neuroni. Caracteristica **holistică înseamnă** apariția unui sistem capabil să aibă o performanță complexă, dar totuși sistemul este alcătui din lucruri relativ simple. Step nu este singura funcție folosită pentru a activa un neuron, alte funcții care pot fi folosite sunt: sigmoid, Relu, Tanh (vezi Fig 3.2.).

Chart, line chart

Description automatically generated  
Fig 3.2. Funcții de activare [14]

Alegerea funcții de activare va modifica performanțele rețelei neuronale. Iar fiecare neuron poate să aibă o funcție de activare diferită ( de obicei, toți neuronii de pe același strat au aceeași funcție de activare ). Există 3 tipuri de straturi:

1. *Input layer*
2. *Hidden layer*
3. *Output layer*

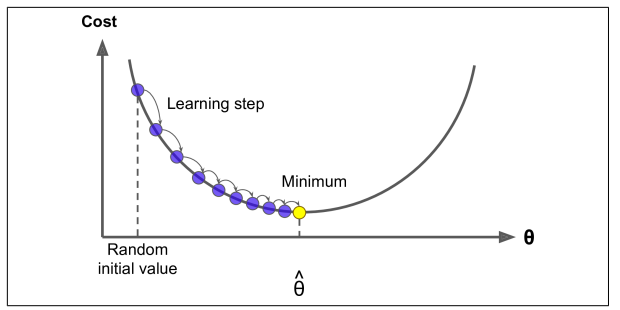
Într-o rețea standard un neuron îți ia input-urile de la *layer*-ul precedent și îl trimite către *layer*-ul următor drept output. Excepții fac *Input layer* și *Output layer*, unde doar primesc input (*output layer*) sau doar trimit output mai departe (i*nput layer*)

Inițial rețeaua neuronală este inițializată cu *bias*-uri și greutăți care au valori *random*, de obicei cuprinse între -1 și 1. Dacă oferim *inpu*t la această rețea neuronală generată la întâmplare se va obține un *output* .... la întâmplare. Aici intră partea de *Machine Learning* se va realiza ajustarea biasurilor și greutăților astfel încât *output*-ul să fie cel dorit pentru *input*-ul dorit. Mai există și alte feluri de *layer*-uri precum cele recursive care sunt folosite pentru aplicații de *speech recognition*.

*Backpropagation* este un algoritm care, în perioada de antrenare, se aplică după o predicție (un *forward pass*) și compara rezultatele rețelei neuronale cu rezultatele dorite. Se folosește o funcție de cost pentru a stabili o metrică pentru cât de departe este rezultatul obținut de cel dorit. În funcție de această metrică se vor ajusta *bias*-urile și greutățile, dacă metrică stabilește că este “pe aproape” de rezultatul dorit ajustările vor fi minore sau chiar nesemnificative. Schimbările sunt realizate în jurul **funcției de cost** care determină cât de bine sau cât de rău rețeaua neuronală se descurcă.

Totul la găsirea unui minim din calculus, doar că aici avem o funcție complexă pe n axe, o axă pentru fiecare input acceptat de rețeaua neuronală. Schimbările aleatorii sunt pentru a ajunge la un minim cu performanțe acceptabile, numit *“gradient descent”*, descent pentru că coboară către unul dintre minime. Cât de repede/încet coboară depinde de *learning rate* setat pe model (cât de rapid se realizează schimbările). *Learning Rate* se modifică pe parcurs, când este aproape de un minim atunci este mic pentru a nu rata acel minim și când este departe de minim atunci este mare. Exista 3 categorii de valori inițiale al *learning rate*-ului:

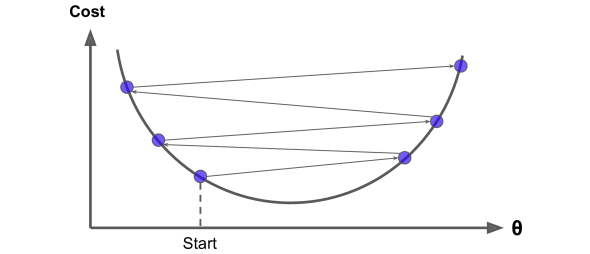
1. *Learning Rate* potrivit care duce la un gradient descent către minim obținut în timp util (Fig.3.1.1).

  
Fig.3.1.1 [14] Exemplu de gradient descent potrivit

1. *Learning Rate* prea mic care duce la un gradient descent către minim obținut într-un timp prea lung sau nu a ajuns la minim în timpul acordat.

  
Fig.3.1.2 [14] Exemplu de gradient descent prea încet

1. *Learning Rate* prea mare care duce la un gradient ascent, ce se îndepărtează de minimul căutat.

  
Fig.3.1.3 [14] Gradient ascendent provocat de *learning rate* prea mare

Bineînțeles, nu toate funcțiile sunt parabole drăguțe precum în figurile de mai sus, pot să aibă urcușuri/coborâșuri abrupte, platouri, multiple minimuri.

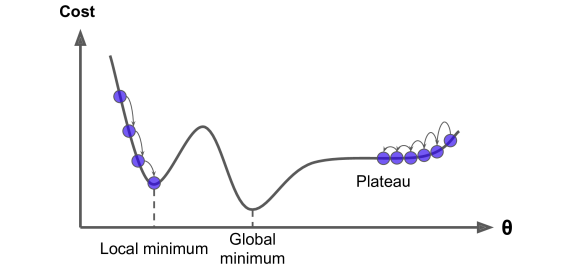
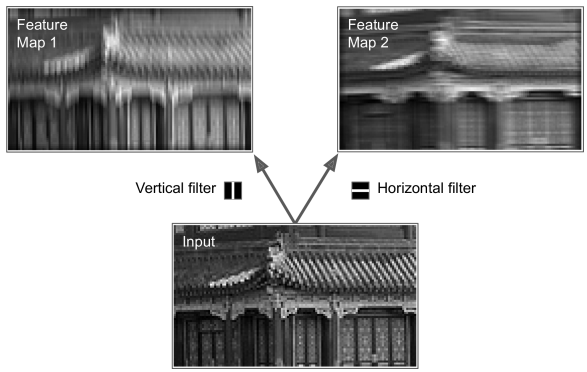
.

Fig.3.1.4 [14] O funcție de cost mai realistică.

### 3.1.1 CNN ( *Convolution Neural Network* )

Este o subclasă de rețele neuronale care se utilizează deseori în analizarea imaginilor, pot fi folosite și în *speech recognition.* Componenta a unui CNN care o diferențiază de rețelele neuronale normal sunt *layer*-urile de convoluție care practic filtrează anumite lucruri. De exemplu, primele layer-uri de convoluție pot să filtreze lucruri relativ ușoare precum linii orizontale și linii verticale ( vezi figura 3.2.1 [9] ).

   
Figura 3.1.1.1 [14] Evidențierea linilor orizontale și verticale folosind *Convolution layer*

Aceste filtre, dacă sunt suprapuse, pot să scoată în evidență lucruri nu doar lucruri simple, ci lucruri complexe precum un nas uman, laba unui câine, botul unei pisici, aripa unei păsări, etc... Aceste informații scoase în evidență de către layer-ele convoluționale ajung la *layer*-urile superioare unde ar trebui să se detecteze omul, câinele, pisica, pasărea sau orice urmărește modelul. Nu se încadrează în *fully connected layer*, adică un neuron nu este conectat cu toți neuronii de pe *layer*-ul inferior, și doar cu câțiva. În cazul de *object detection* o poză care are are dimensiuni de 512x512 ajunge la un număr total de 262.144 de pixeli, însemnând un număr ridicol de mare, de date de intrare. Aici intră în folosință *convolutional layer* care practic filtrează informațiile relevante din toți cei 262.144 pixeli și caută *features.*

### 3.1.2 Tensor

Un tensor poate fi perceput ca o matrice cu un număr variabil de dimensiuni. În capitolul 1 s-a menționat că o imagine poate să fie alcătuită din 3 matrici, una pentru fiecare culoare (RGB). Dacă se folosește ca input o poză în RGB de exemplu cu rezoluția 1920x1080, atunci se va lucra cu un tensor de 1920x1080x3 ( x3 că poza are 3 culori ), unde fiecare element este un byte cu valori între 0 și 255. Dacă se folosește drept input un video care este alcatuit din 182 de frame-uri care folosesc codul RGB cu rezoluția 1280x720 atunci avem un tensor de mărimea 1280x720x3x182 ca și input. Există până și o componentă TPU (*Tensor Processing Unit*) care poate realiza operațiile cu tensori mai rapid decât CPU sau chiar decât un GPU.

### 3.1.3 Rețele de tip SSD ( *Single Shot Detector* ) [15]

SSD-urile sunt rețele neuronale care sunt făcute pentru a detecta rapid și eficient fără a utiliza multe resurse. Acestea realizează detecția fără a realiza operația de propunere de regiune, operație folosită în R-CNN (*Region-based Convolutional Neural Network*).Pentru a compensa în precizie SSD folosesc îmbunătățiri precum factori *multiscale* și default *boxes*. Modelele SSD sunt considerate CNN-uri așa că folosesc totuși features maps și convolution filters pentru a detecta obiecte. Calculează atât cât locația și scorul clasei (siguranța în procent) folosind filtre mici convoluționale. Predicțile provenite de la SSD sunt clasificate ca și potriviri pozitive sau potriviri negative. SSD utilizează numai potriviri pozitive pentru a calcula costul de localizare (amplasarea greșită a cutiei) atunci când siguranța modelului depășește *threshold*-ul necesar pentru a face detecția.

## Tehnologii folosite

### Python [16]

Este un limbaj general cu următoarele proprietății:

* de nivel înalt (ușor de înțeles pentru om)
* interpretat ( nu este compilat, codul nu este generat în limbaj mașină )
* interactiv ( odată ce este rulat, utilizatorul poate adauga în prompt, simbolizat de semnul „ >>> ” sau de „ ... ” alte linii de cod )
* Limbaj Orientat pe Obiecte
* Poate utiliza o gamă diversă de biblioteci precum cele folosite în cadrul acestui proiect, *TensorFlow, scikit-learn*.
* Un timp de execuție mai lung decât limbajele precum C și C++.

### TensorFlow

TensorFlow: Este o bibliotecă *open-source, multi-platform* (Windows, Linux, MAC, mobile) concepută pentru computație numerică, aceasta este eficientă și potrivită, în particular, pentru operațiile necesare dezvoltării rețelelor neuronale. Fiecare operație făcută în cadrul bibliotecii TenserFlow este implementată în limbajul eficient C++. Multe dintre aceste operații au mai multe implementări, numite *kernels.* Fiecare kernel este dedicat unui dispozitiv specific precum CPU, GPU și TPU.

### Tensorflow Zoo [17]

Așa cum îi sugerează numele, este o “grădină zoologică” din care se pot utiliza diverse modele antrenate pe *dataset*-ul COCO 2017. Pentru a avea o idee despre ce presupune fiecare model se folosesc două metrici, mAP *(mean Average Precision*) și *Speed* [ms]. mAP reprezintă precizia modelului. Modelele sunt de la neprecise (20 mAP) și rapide ( 20 ms ) la precise (51.2 mAP) și încete (325 ms). Fiecare model este capabil de a furniza output pentru:

1. Boxes (încadrează obiectul detectat într-o cutie)
2. Keypoints (detectează punctele cheie, folositor pentru a identifica lucruri precum emoția unui om)
3. Masks (delimitează obiectul față de restul mediului înconjurător)

Mai oferă și diferite *script*-uri utile. Un exemplu bun ar fi script-ul model\_main\_tf2.py care în funcție de parametrii oferiți la rularea scriptului poate realiza antrenarea sau evaluarea modelului. Un alt lucru important sunt fișierele pipeline.config care vin personalizate pentru fiecare model. În ele se pot modifica hyperparametri precum funcția de activare, batch size-ul, threshold-ul, learning rate-ul inițial.

### Keras

Este un API de nivel înalt care permite o ușoară construire, antrenare, evaluare, execuție multiple tipuri de rețele neuronale. Pentru a efectua calculele de nivel jos keras se bazează pe o bibliotecă *back-end* pentru realiza calculele dificile. Pe lângă TensorFlow mai sunt, Microsoft Cognitive Toolkit (CNTK) și Theano. S-au dezvoltat mai multe implementări care permit rularea keras pe Apache Apple’s Core ML, MXNet, Typescript (web browser), Java și PlaidML ( pe orice fel de GPU, nu doar pe Nvidia). Tensorflow include automat și Keras.

### TensorBoard

Este un *tool* interactiv care poate fi folosit pentru a vizualiza cum se dezvoltă metricile de performanță a modelului pe parcursul antrenarea acestuia, compararea rezultatelor cu diferite iterații a lui sau chiar cu diferite modele. Pentru a fi folosită este nevoie de generarea anumitor fișiere de tipul *binary log event files*. Acestea conțin informații despre metrici (*clasification\_loss*, *localization\_loss*, *regularization\_loss* și *total\_loss*) atât și detecțiile realizate de model. Aceste detecții pot fi vizualizate în interfața din Fig 3.2.1:

Graphical user interface

Description automatically generated

Fig 3.2.5.1: Interfață TensorBoard

Fiecare detecție este afișată în două panel-uri ale aceleiași poze, dar cu box-uri diferite peste. În partea stângă sunt detecția realizată de modelul antrenat, iar în partea dreaptă sunt detecția ideală din adnotație.

### LabelImg

O aplicație gratuită, de tipul open source pentru a adnota pozele. Este scrisă în python și utilizează software QT pentru a UI. A fost aleasă pentru că este o aplicație comodă, cu capacități bune de zoom pentru o adnotare mulțumitoare. Pozele adnotate au fost salvate în formatul Pascal/YOLO. Totuși nu poate să salveze pozele direct în formatul .tfrecord sau .record așa ca s-a optat pentru folosirea următoarei aplicații (Robotflow).

### RobotFlow

O platformă web care poate fi folosită pentru a realiza computer vision. În cazul acestui proiect această platformă a fost utilizată principal pentru a converti formatul Pascal/VOLO obținut din aplicația LabelImg în fișiere cu extensia .tfrecord, fișierele .tfrecord care sunt utilizate de către fișierul pipeline.config pentru ca pozele să ajungă ca input la model, iar fișierele .xml să ajungă ca output-ul ideal pentru a se calcula costul modelului. Robotflow a mai fost folosit pentru:

* a face resize la poze ( de la 1.376 x 1.020 pixeli până la 420 x 420 pixeli).
* convertirea pozele în grayscale.
* Separarea automată a pozelor de train cu cele de eval.

# Antrenarea rețelei neuronale

## Pregătirile necesare pentru antrenare

Cele două componente esențiale sunt modelul folosit și setul de date (antrenare și evaluare). Hrănirea modelului cu date este realizată prin pipeline.config, în cadrul acestui fișier trebuie setat următoarele lucruri:

* locul (*path*-ul) de unde se iau datele pentru antrenare/evaluare (fișierele .tfrecord)
* locul (*path*-ul) în care se află *label map*
* locul (*path*-ul) unde se află *checkpoint*-ul inițial ( modelul pre-antrenat pe COCO ) pentru a fi adaptat *dataset*-ului nou
* faptul că modelul trebuie să facă *detecțion*, în mod implicit este setat pe *classfication*
* *Batch\_size* implicit este setat pe 64, din cauza limitei impusă de memoria RAM, trebuie redus la 16 ( sau chiar mai puțin ).

După ce a fost configurat fișierul pipeline.config, se apelează un *script* care aparține de tensorflow zoo, numit model\_main\_tf2.py [17]. Acest script poate antrena și după, poate evalua cu o apelare diferită checkpoint-ul modelului respectiv. Diferența dintre antrenare și evaluare este dată de parametri de intrare pe care modelul le utilizează. Parametri de intrare:

* *model\_dir* ( unde se va salva checkpoint)
* *pipeline\_config\_path* ( locația *pipeline*-ului folosit )
* *num\_train\_steps* ( antrenare ) sau *checkpoint\_dir* ( evaluare )

*num\_train\_steps*: câți pași să se realizeze. Ce înseamnă un pas? Efectuarea unui back propagation. Adică parametrul definește prin câte iterații trece rețeaua neuronală până când antrenarea se oprește și salvează checkpoint-ul pentru a putea fi evaluat de apelarea care să aibă ca parametru locul checkpoint-ului respectiv.

Odată evaluat rezultatele vor fi salvate în folder-ul de eval, care pot fi accesate de către tensorBoard pentru a afișa graficele și detecțiile propiu-zise.

## Dezvoltarea iterațiilor modelului de tip SSD

Pentru a antrena modelul (SSD\_MobileNet\_v2\_FPNLite\_320 din tensorflow zoo) prima data cu Obiectivul de a detecta nucleele normale și anormale s-a realizat un data set format din 8 imagini (6 imagini pentru antrenare si 2 pentru testare). Antrenarea modelului a fost realizată la pașii 1.000, 2.000, 3.000, 4.000. Deci sunt 4 puncte pe grafic.

Line chart

Description automatically generated

Fig.4.2.1 Scăderea metricii total loss cu antrenarea modelului

Deși face îmbunătățiri pe măsură ce se își dezvoltă pașii de antrenare, acesta are o pierdere prea mare pentru setul de evaluare, iar pentru setul de antrenare este o pierdere foarte mică. Asta însemnând că modelul învață foarte bine cele 6 poze de antrenament și nu știe cum să analizeze celelalte poze. Soluția posibilă ar fi să adăugăm mai multe poze de antrenament. Acest lucru este explicat de faptul că pe setul de antrenament s-a ajuns pe o pierdere de 0.16 (din care 0.117 este *regularization\_los* pentru a evita fenomenul de overfit, 0.04 din total sunt regularization și classification).

Text

Description automatically generated

Fig.4.2.2 Pierderile din care rezultă metrica *total\_loss*

Așa arată performanța modelului pe una dintre pozele de evaluare (*threshold*-ul de activare este de 0.3, dacă ar fi fost 0.5 cum e setat implicit probabil nu ar fi detectat nimic):

Text

Description automatically generated

Fig.4.2.3 CRC 261 încercare de detecție

Pentru a se antrena mai bine au fost adăugate 3 poze cu un număr mai mare de adnotări (fiecare adnotare fiind un nucleu). Antrenamentul constă în efectuarea a 1.000 de pași, scăzând astfel de la un 0.66 total loss până la 0.28, în comparație cu primul antrenament unde s-a scăzut până la 0.22. Simbolizând faptul că dataset\_1\_v2 este relativ mai dificil decât primul, dar nu chiar foarte dificil. Totuși, nu s-a realizat nicio detecție care să depășească threshold-ul de 0.3. Există posibilitatea ca imaginile din antrenament să fie prea diferite față de imaginile din evaluare (în sensul de culori, cel puțin). Acest lucru evidențiază faptul că există o variabilitate celulară mare, influențată atât de substratul histologic al biopsiei, cât și de colorațiile multiple folosite în practică pentru identificarea celulară. Totuși dacă un om poate evidenția celule individuale, atunci și o inteligență artificială poate. Rezultatul la 2000 de pași:

Graphical user interface, text, application, chat or text message

Description automatically generated

Fig.4.2.4 fenomenul de detecție multiplă observat în CRIC 207 (După 2.000 de steps)

S-au identificat nuclei anormali în această poză, totuși în poza de fig.3 nu a făcut nicio detecție. Se pot observa 4 nuclei anormali detectați, când sunt numai 2. Probabil pozele de antrenament sunt mult prea diferite de poza din fig.3 folosită drept reper de evaluare. După 3.000 de pași:

Text, application

Description automatically generated

Fig.4.2.5 detectarea nucleului normal CRC 207 (după 3.000 de pași)

S-au identificat numărul exact de nuclee anormale, cu un anumit “rateu” pentru nucleul anormal de sus. Și a reușit să identifice un nucleu normal. După 4.000 de pași nu mai detectează nucleul normal, probabil modelul a intrat deja în etapa de *overfitting*, o să fie antrenat la 7.000 de pași pentru a demonstra că extra antrenamentul duce la un efect negativ pentru detectarea nucleelor. După antrenare *Total Loss* a scăzut până la 0.14 din nou, care este “suspicios” de puțin.

Graphical user interface, text, application, chat or text message

Description automatically generated

Fig.4.2.6 modificare de procente la CRC 207(după 7.000 de *steps*)

Modelul a reușit să identifice nucleul normal cu o siguranță mai mare decât cel anormal ( 56% comparat cu 88%). Asta sugerează că apropierea de overfitting poate fi benefic pentru detectarea celulelor normale, care de regulă sunt cele mici.

Pentru a verifica dacă cauza majoră a lipsei de detecție în fig. 3 este culoarea portocalie pe care modelul nu a mai întâlnit-o în setul de antrenament actual se vor preprocesa toate imaginile în alb/negru.   
Antrenarea s-a realizat prima data peste cei 7.000 de pași deja realizați. S-a observat o creștere de *Total\_Loss* de la 0.14 (unde a fost anterior) la 0.18, semnificând că este o mică diferență. Totuși nu a ajutat cu mult la detecție. După aceea s-a antrenat modelul de la 0, cu 4.000 de pași folosind numai poze alb-negru.

A picture containing text, igloo, stone

Description automatically generated

Fig.4.2.7 Încercare de reantrenare pe poze grayscale

După ce modelul a fost antrenat numai cu poze alb-negru, de la forma lui de pretrained cu 7.000 de pași s-a ajuns la concluzia că culoarea imaginii ajută la detecția celulelor. În Fig. 8 se poate observa că metrica de *total\_loss* este mereu mai scăzută (însemnând un model mai bun) atunci când pozele au culoare. Linia gri este modelul antrenat pe grayscale și linia portocalie este cel antrenat pe poze colorate.

Chart, line chart

Description automatically generated

Fig.4.2.8 Modelul bazat pe Culoare (portocaliu) vs Modelul bazat pe alb-negru (gri)

Pentru a ajunge la o detecție decentă pentru cea de a doua poză se vor adaugă poze care să fie cât mai similare cu aceasta.

La 2.000 de pași de antrenare cu noile poze s-a observat că detecțiile nu sunt realizate într-un mod mulțumitor (decât unele dintre nuclee anormale au fost detectate). Pentru a încerca să se remedieze s-a mărit batch\_size la 32. Se remarcă un tipar nociv pentru rezultatul detecției în figura 9.

Text, letter

Description automatically generated

Fig.4.2.9 detectarea eronată a două nuclee învecinate

Nuclee abnormale au o rată de multiplicare mult mai mare, așa că se întâmplă mult mai des ca nuclee anormale să fie aproape suprapuse una peste alta. Modelul observă acest lucru și când vede două nuclee apropiate are tendința să detecteze mai multe decât ar trebui. Acest lucru păcălește și metricile de performanță. Din moment ce suprafața detectată coincide cu o celulă anormală.

Pentru a ieși din acest impas, s-a hotărât ca modelul să:

* ia în considerare doar celulele anormale, care sunt cele mai importante pentru acest proiect.
* Dacă două celule sunt suprapuse sau se ating sa fie adnotate drept una.

Antrenarea modelului cu obiectivul de a identifica doar nucleele anormale a fost realizată un *dataset*, în forma inițială de 21 de imagini (17 utilizate pentru antrenament și 4 pentru evaluare), fiecare conținând multiple nuclee normale cât și anormale, sarcina modelului este să le identifice doar pe cele anormale.

Text, chat or text message

Description automatically generated

Fig.4.2.10 Cric\_381, threshold 0.25

Pentru a evita să detecteze multiple nuclee anormale se va mări threshold-ul de la 0.25 la 0.4. A fost inițial redus pentru a evidenția celulele normale. Această schimbare a necesitat antrenarea de la 0 a rețelei neuronale.

Text

Description automatically generated

Fig.4.2.11 Cric\_381, threshold 0.4

După mărirea threshold-ului metrică de total\_loss a suferit drastic. Dar totuși detecțiile sunt considerabil mai precise. În Cric\_381 a reușit să identifice 3 din 4 nuclee anormale și în cazul Cric\_375 a reușit să detecteze 5 din cele 11 celule. Dezavantajul la un threshold constă în faptul că nuclee care nu se aseamănă cu ce a fost în training set nu sunt detectate ( nucleul mai palid din figura 11). Dezavantajul în cazul detecției nucleelor al unui threshold mic este detecția multiplă a aceluiași nucleu, mai ales când acest nucleu este apropiat de alt nucleu.

Observație: Până acum modelul a ratat câteva nuclee anormale, dar în toate pozele, a identificat cel puțin un nucleu anormal. Acest lucru este un rezultat pozitiv pentru că într-o mare colecție de probe (care nu poate fi analizată de personalul medical imediat), poate să se scoată în evidență cele care probabil au cancer și rapid. Pozele scoase în evidență pot fi rapid aprobate de personalul medical și tratamentul poate să înceapă puțin mai devreme. Evident, ar fi recomandat să fie studiate celelalte mostre cu așteptarea că modelul a ratat o persoană bolnavă.

S-au adăugat mai multe poze care au nuclee asemănătoare cu nucleele din Cric\_275 (Cric\_199, Cric\_206, Cric\_221 și Cric\_222) în *dataset\_abnormal\_v1* (sa denumit *dataset\_abnormal\_v2*) și după antrenare s-a constatat că cele 4 poze au un efect dăunător asupra antrenări atât în metrica de total\_loss cât și vizibil cu ochiul liber. Performanțele celor două modele:

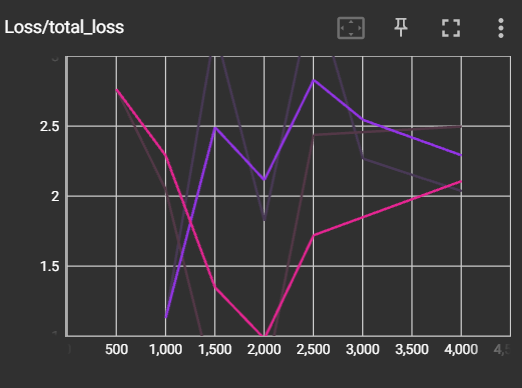


Fig.4.2.12 Comparație între dataset\_abnomal v1 (roz) și v2 (mov)

Chiar dacă pe *dataset\_abnormal\_v2* sa obținut o metrică scăzut la 1.000 de pași nu este validă pentru că are foarte puține detecții în comparație cu alte etape ale antrenări (pe CRIC\_375 nu a detectat nimic) în etapa respectivă și pe CRIC\_381 a detectat decât un nucleeu din cei 4.

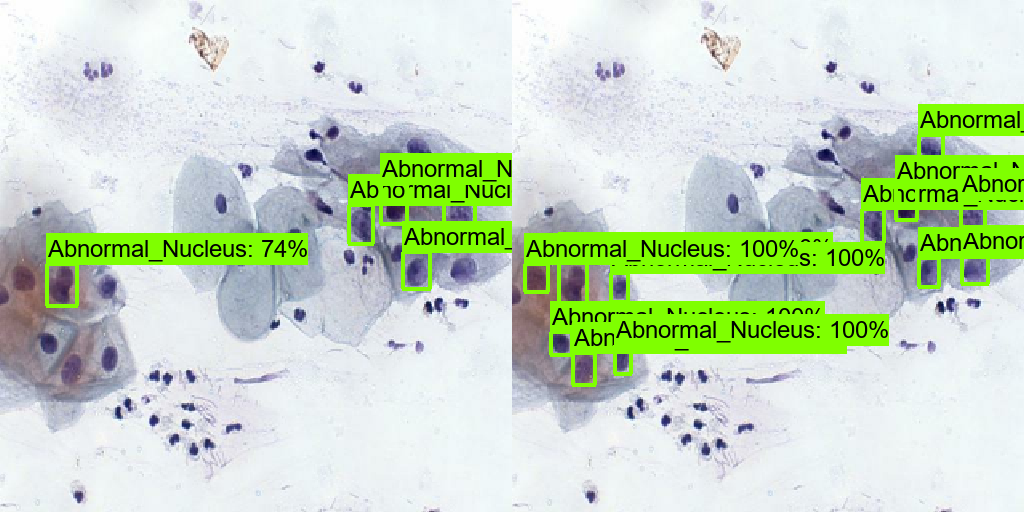
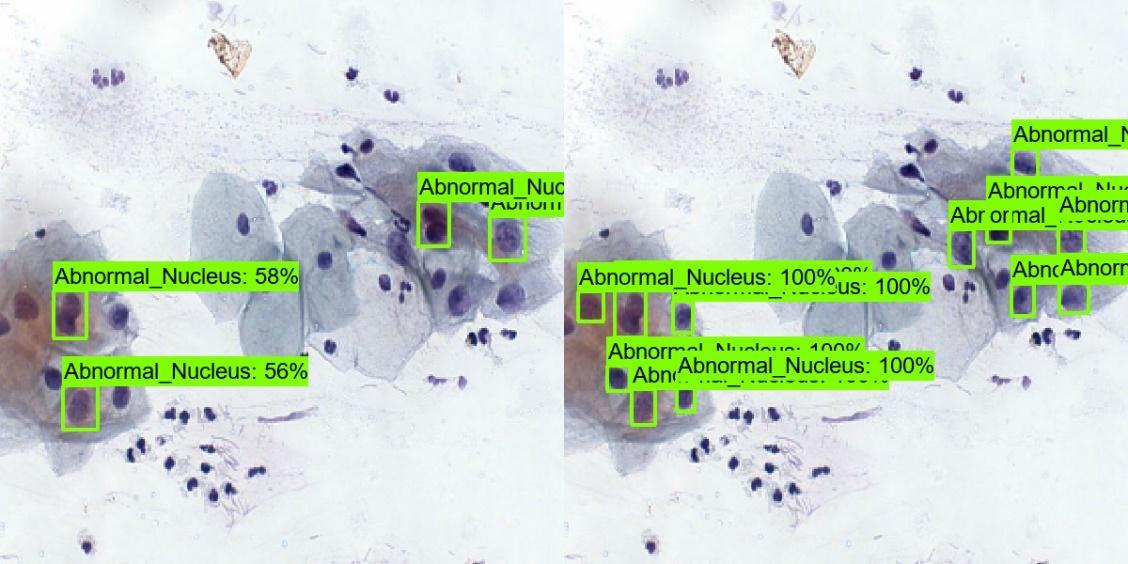


Fig.4.2.13 *dataset\_abnormal\_v2* (4/11) vs Fig.14 *dataset\_abnormal\_v1* (5/11)

În concluzie există si posibilitatea de a avea prea multe date de antrenare în acest caz, poate fi cauzat de complexitatea prea mică a modelului. Acest fapt că mărirea bazei de antrenare sugerează că modelul oarecum “uită” ce a învățat.

Observație: În timpul antrenări modelului pe *dataset\_abnormal\_v2* sa observat că la un moment dat a făcut această detecție eronată:

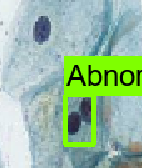


Fig.4.2.15 Detecție eronată

Acest lucru a fost provocat că în *dataset* modelul a fost mințit de principiul “Dacă două celule sunt suprapuse sau se ating sa fie adnotate drept una”. Pentru că acest principiu a fost introdus în scopul de a evita detecțiile multiple a unui singur nucleu, iar acest lucru a fost deja realiza de mărirea threshold-ului la 0,4. Se vor modifica setul de training astfel încât să reflecte cât mai mult realitatea. Adică nucleele suprapuse vor fi considerate individual. În plus, în cazul de față detectarea fals – pozitivă, adică detectarea unui nucleu anormal când nu există unul este mult mai nocivă decât nedetectarea unui nucleu anormal. Pe ideea că persoanele medicale care utilizează un astfel de model mai bine gândesc că în realitate sunt mai multe nuclee anormale decât cele care au fost detectate.

Următorul lucru care ar trebui investigat este performanța modelului pe multiple ramuri de cancer care pot fi găsite pe site-ul CRIC [18]. Până acum modelul a lucrat decât cu imagini care au celule de cancer din ramura ASC-US. De acum se va realiza dataset\_LSIL care va include imagini care conțin nuclee morfologic specifice pentru leziunile de tipul LSIL. Nu ar trebuie să fie un rezultat foarte diferit pentru că celulele care constituie leziunile de tipul LSIL au un nucleu care se remarcă față de cele normale.

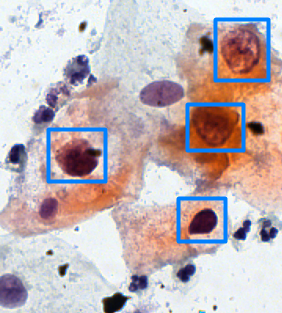
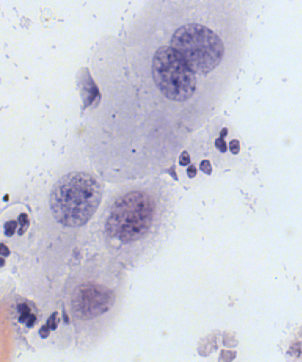


Fig.4.2.16 CRIC\_354 nuclee cu leziunea LSIL (marcat de cutia albastră)

  
Fig.4.2.17 CRIC\_381 nuclee cu leziunea ASC-US

Modelul va identifica doar celule anormale fără să încerce să identifice ramura care este leziunea.

**Crearea unui model pentru a detecția leziunilor din subtipul LSIL**

S-a creat în primul rând un dataset de 11 poze (LSIL\_dataset\_v1). Detecția nu a mers foarte bine, a detectat 1/7 celule din CRIC\_356 și din CRIC\_345 0/4, total\_loss în timpul antrenării a ajuns repede la valoare 0.14 care este foarte puțin. Asta înseamnă că sunt prea puține poze de antrenare (9 este prea puțin în acest caz).

S-au mai adăugat încă 4 poze (LSIL\_dataset\_v2) și s-a obținut următorul rezultat:

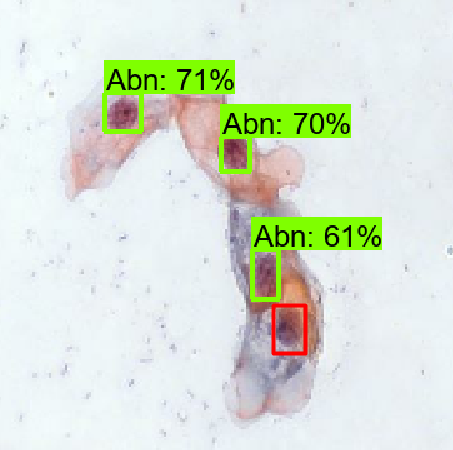


Fig.4.2.18 Detecție realizată pentru CRIC\_356 (3/4 nuclee detectate)

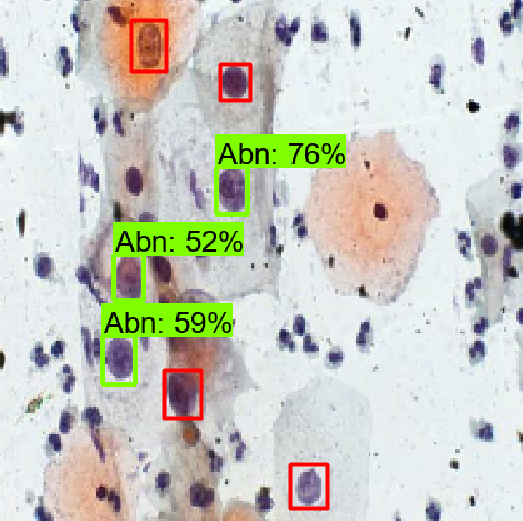


Fig.4.2.19 Detecție realizată pentru CRIC\_345 (3/7 nuclee detectate)

Modelul antrenat cu al doilea set de date a avut o metrică de total\_loss (vezi Fig. 4.2.19) pe pozele de evaluare mai redusă. Deși diferența este de doar de 0.2 din punct de vedere unitar, modelul antrenat cu al doilea set a detectat de 3 ori mai multe nuclee decât cel antrenat pe primul set.

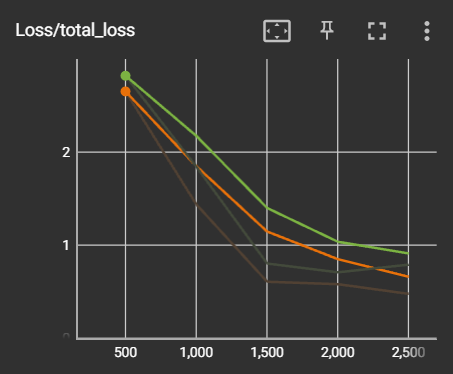


Fig.4.2.20 Total\_Loss pe v2 (portocaliu) și v1 (verde)

În timpul antrenării celei de a doua versiune s-au observat diferite oscilații în metric\_loss (de la 0.20 ajunge la 0.26 ). Asta indică că există cel puțin o imagine în care a găsit ceva diferit față de training set.

S-a mai remarcat că de regulă cea mai bună zonă de antrenare a modelului folosit (SSD\_MobileNet\_v2\_FPNLite\_320) este între 2.000 și 2.500 de *steps*, mai mult de atât se duce spre overfitting si are un efect negativ asupra detecției, dar un efect pozitiv asupra metricilor. Acest lucru poate fi observat în comparație dintre figura 21 și figura 22.

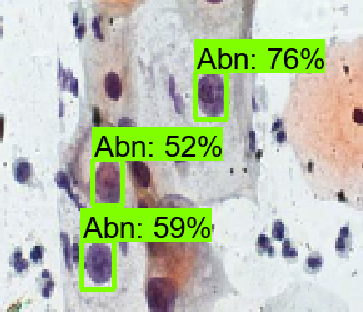


Fig.4.2.21 Detecție după 2.500 de pasi

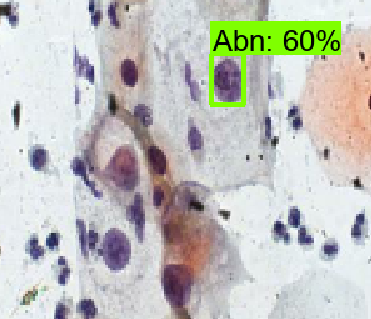


Fig.4.2.22 Detecție după 3.500 de pasi

S-a încercat și adăugarea a încă 4 poze în setul de antrenare și nu s-a mai detectat nimic în imaginea CRIC\_345, iar în CRIC\_356 tot aceleași 3 nuclee au fost detectate doar cu o siguranță redusă de 10%.

**Crearea unui model care poate să detecteze ramurile ASC-US și LSIL.**

Pentru un astfel de model s-a creat un dataset care să includă atât nuclee specific descriptive pentru leziunile de tipul ASC-US și LSIL numit dataset\_hibrid. Compus din 13 poze de antrenare (care este numărul potrivit pentru modelul cu arhitectura folosită) care cuprind atât ambele tipuri de ramuri. Performanța modelului a suferit considerabil, evidențiind faptul că ramurile de leziuni cu potențial precancerigen și cancerigen de tipul ASC-US și LSIL au morfologii diferite și îngreunează detecția realizată de model. S-a întâlnit pentru prima data cazul de *False Positive* în Cric\_203 (Fig. 4.2.23).

Text

Description automatically generated

Fig.4.2.23 Cazul False Pozitive, detectarea unui nucleu inexistent.

Un alt caz interesant rezultată din evaluarea acestui model pentru poza CRIC\_354. Această poză conține ambele ramuri, 4 nuclee specifice leziunilor de tipul LSIL și 8 nuclee care descriu subtipul caracteristic leziunilor cu specific ASC-US. Modelul a detectat 6 din 8 nuclee specifice pentru ASC-US și niciuna pentru ramura LSIL (vezi Fig. 4.2.24). Nucleele marcate cu albastru sunt cazul de false negative pentru ramura LSIL și cele cu roșu sunt cele de false negative pentru ramura de ASC-US.

Application, map

Description automatically generated

Fig.4.2.24 Detecție pe ambele ramuri

Pentru a remedia lipsa de detecție din mostra CRIC\_354 pentru ramura LSIL s-a încercat extragerea din training set pozele irelevante și introducerea unor noi poze asemănătoare, adică să includă din nou ambele ramuri de leziuni cu potențial precancerigen si cancerigen, probabil recoltate de la același pacient. Mai exact au fost introduse mostrele CRIC\_366, CRIC\_364 pentru a ajuta detecția pe CRIC\_354. Acest lucru a îmbunătățit vizibil detecția pentru mostra CRIC\_354, a ratat doar un nucleu specific leziunilor de tipul ASC-US și a detectat 2 dintre cei care ar putea descrie diagnosticul cu LSIL. ( toate acestea sunt evidențiate în Fig. 4.2.25 )

Map

Description automatically generated

Fig.4.2.25 detecție utilizând traing set-ul focusat

O problema care a ieșit în evidență când a fost antrenat modelul utilizând dataset-ul hibrid (atât ramurile LSIL și ASC-US). Este faptul că nucleele și celulele umane sunt foarte diferite de om la om. Acest fapt ridică probleme atât pentru diferitele iterații a modelului antrenat, cât și pentru oameni care nu au analizat multe mostre de testul Papanicolau (PAP). Pentru a realiza o predicție precisă trebuie analizată o mare cantitate de mostre, care sunt chiar de la pacienți diferiți. Este posibil ca modelul utilizat până acum să aibă o arhitectură relativ prea simplă. Acest lucru înseamnă probabil că mărirea setului de antrenare de la 15 la 25 de poze este probabil să ducă la o anumită “uitare” a pozelor deja studiate. Deși în training obține un Total\_loss decent memorând tipare, acele tipare sunt prea abstracte pentru a obține un rezultat dorit. Acest lucru a fost observat la mărirea set-ului de training.

Un alt lucru care poate fi experimentat, este hyperparametru funcții de activare, singura funcție de activare care a fost folosită până acum a fost funcția setată implicit RELU\_6. Este foarte asemănătoare cu RELU normal doar că activarea este limitată la maxim valoare 6 ( de unde și numele de RELU\_6).

Se va schimba funcția RELU\_6 cu: RELU și SWISH

S-a observat că SWISH are un timp de antrenare și de detecție aproape dublu. Acest lucru este provocat de componenta exponențială a acestei funcții, care aduce la un nivel complex atunci când este comparat cu celelalte funcții de activare folosite, RELU/RELU6.

Chart, line chart

Description automatically generated  
Fig.4.2.26 Funcția de activa SWISH desenată cu ajutorul [19]

Funcția de activare SWISH:

Chart, line chart

Description automatically generated

Fig.4.2.27 Compararea metricii de pierdere pentru diferitele funcții de activare

Se observă că RELU se descurcă cel mai deficitar, SWISH cel mai bine iar RELU6 este aproape de SWISH. Dar totuși sunt două motive pentru care nu se justifică folosirea funcții de activare SWISH:

* Timp dublu necesar pentru utilizare și antrenare.
* În ciuda metricii îmbunătățite detecția ratează multe nuclee.

Dacă pentru RELU a fost un Recall de 6/12 (care a avut cea mai proastă performanță din punct de vedere matricial), modelul antrenat cu SWISH care a fost cel mai costisitor a avut un Recall de 5/12 (vezi Fig. 4.2.28).

Map

Description automatically generated Application, map

Description automatically generated

Fig.4.2.28 Detecție pentru SWISH Fig.4.2.29 Detecție pentru RELU

Detecția pentru RELU\_6 a fost deja arătată în Fig.25. Care este cel mai preferat model, detecție bună pentru un timp rezonabil. De acum înainte se va lucra cu funcția de activare RELU\_6.

Modelul SSD\_MobileNet\_v2\_FPNLite\_320 pare promițător, este capabil să aibă un *recall rate* de peste 50%. În plus este mic, se antrenează repede și necesită 0.5 secunde pentru a face o detecție ( cu *hardware*-ul folosit ). În capitolul viitor se va antrena un model mult mai complex pe același set de date pentru a observa avantajele unui alt fel de model.

### 4.3. Utilizarea unei noi arhitecturi de model. (CenterNet)

S-a ales utilizarea arhitecturii *CenterNet HourGlass104* 512x512 care a obținut un mAP (precizie) de 41.9 cu o viteză de 70 [ms] pe COCO dataset. În comparație cu SSD MobileNet care a obținut un mAP 22.2 cu o viteză de [22] ms pe COCO dataset. CenterNet este un tip de arhitectură diferită față de SSD, judecând după performanțele realizate pe COCO *dataset*, sacrifică viteza și alte resurse precum memoria pentru a face predicții cu o precizie cât mai mare. Noul model, *CenterNet HourGlass104* 512x512, a fost antrenat pe același set de date, dataset\_hibrid\_v2.1. După antrenare a identificat chiar prea mult.

Map

Description automatically generated

Fig.4.2.30 Detecție realizată de noul model

Partea bună este că a identificat fiecare nucleu cancerigen. Partea proastă este că a identificat și mai mult de atât. A identifica până și un nucleu de la marginea pozei care este probabil cancerigen, dar nu a fost considerat până acum din cauza poziției lui pe poză. S-a încercat reducerea mărimii fontului în care este scris “Abn: \*procent\*” dar din cauza unui bug nu sa realizat nicio schimbare. Un alt lucru interesant este că rețeaua are nevoie de mult mai puțini pași de antrenare, Fig.4.2.30 este după doar după 1.000 de pași de antrenare, pe când SSD\_mobileNet avea nevoie de 2.000 ca să ajungă în zona bună.

Pentru următorul antrenament se vor schimba: Pentru a observa în mai mult detaliu antrenarea rețelei se va salva din 300 pași în 300 pași (schimbare realizată în scriptul și Mărirea rezoluției imaginilor de la 400x400 pixeli la 700x700 pixeli (care va micșora și fontul)

Map

Description automatically generated

Fig.4.2.31 Detecție la 1.200 de pași

Se observă că modelul este foarte instabil, la 300 a avut o detecție mai bună cu același Recall ( de 9 nuclee ) ca și în Fig.31 dar fără a greși celula normală cu una anormală (Fig. 4.2.31, marcată cu negru). Iar la pașii 600, 900, 1.500 se detecteze prea puține nuclee. Acest lucru indică că *hyperparametr*-ul *learning\_rate* este mult prea ridicat și uneori din întâmplare se ajunge la un punct de minim care să detecteze mulțumitor nucleele. Pentru a testa acest lucru modelul a fost antrenat din nou cu nimic modificat, cea mai bună detecție a fost la pasul 1.500, care confirmă natura aleatorie. Parametrul *learning rate* va fi modificat de la 0.001 la 0.0002. Learning rate nu poate fi mai mic decât 0.001, așa că se va modifica *batch\_size* de la 1 la 4 ( memoria RAM nu permite *batch\_size* mare ). Nu a fost reușită mărirea *batch\_size*-ului din cauza limitării RAM, dar pentru a modifica *learning\_rate* de la valoarea 0.001 de fapt trebuie modificat hyperparametrul *initial\_learning\_rate* din fișierul pipeline.config. Se scade de la 0.001 la 0.0002. Acest lucru necesită antrenarea modelului din nou. Se mai observă că modelul ocupă foarte mult spațiu pe hardisk. Fiecare 300 de pași, adică un checkpoint, ocupă aproximativ 2.5 GigaBytes.

După schimbarea hyperparametrului se ajunge la o scădere de Total\_loss așteptată (vezi Fig. 4.2.32).

Chart, line chart

Description automatically generated

Fig.4.2.32 Total\_loss pentru *learning rate* mic (Verde) vs *learning rate* mare (Galben)

Map

Description automatically generated

Fig.4.2.33 Detecție cu noul *learning rate*

După ce sa mai antrenat la 2.400 sa constatat că modelul îți pierde capacitatea de a detecta nuclee. Deci modelul CenterNet HourGlass104 512x512 este capabil să detecteze mult mai bine nucleele, problema este că începe să detecteze nuclee normale care nu ar trebui sa fie considerate. Acest lucru este grav din cauza faptului că un model într-o situație practică ar lucra în mare parte cu imagini care nu au nicio celulă. Așa că modelul recomandat este SSD\_MobileNet\_320 chiar dacă acesta a fost conceput pentru a realiza detecție în timp real.

# Concluzii

Identificarea nucleelor anormale ale testului Papanicolau utilizând rețele neuronale este realizabilă și posibil utilă. Dacă spitalele ar putea să salveze imagini ale mostrelor la microscop ar putea utiliza un model antrenat pentru a obține rapid o listă imperfectă de diagnostice. Acest lucru ar putea grăbi determinarea dacă un pacient are cancer sau nu. Ar mai putea reduce *workload*-ul personalul medical.

Cele mai bune detecții au fost realizate cu modelul de tip SSD, acest model nu doar că este ușor (nu consumă multe resurse hardware), ci este mai puțin afectat de rezoluția pe care lucrează, s-au stabilit că cei mai buni *hyperparametri* sunt: *Batch\_size* de 16, *Threshold* de 0.4, funcția de activare recomandată este *RELU6*, *dataset*-urile au dat rezultate mai bune când au fost color. În antrenarea modelului de tip *CenterNET* sa observat că este mult mai sigur pe el când identifică un nucleu, atât de sigur pe el încât face detecții eronate. Așa că se preferă modelul de tip SSD nu doar din punct de vedere al costului cât și al performanțelor.

Este foarte important ca modelul să evite cazurile False Pozitive pentru a nu pierde încrederea personalului medical. Dacă detectează nuclee anormale care de fapt nu sunt acolo, atunci încrederea modelului va fi pierdută și personalul medical îl va ignora. Dar dacă modelul găsește doar 10% din nucleele anormale și nu găsește niciodată un nucleu anormal fals, atunci personalul medical îi va lua în serios când realizează o detecție. Cercetând pentru mai departe pentru a concepe un diagnostic final. Un astfel de model ar fi foarte util dacă ar apărea situația de a avea prea multe probe pentru a fi analizat în timp util de către personalul medical.

Aceste principii rămân valabile chiar și pentru celulele de cancer, unde detecția nu este chiar atât de simplă. În cazul testului Papanicolau detecția este relativ simplă pentru că celulele sunt transparente și se poate observa direct nucleele care dau o descriere mult mai bună decât exteriorul celulelor.

## Viitoare îmbunătățiri

1. Un set de date mai extins. Cooperarea cu diverse spitale pentru a obține de mostre.
2. O înțelegere mai bună a noțiunilor medicale pentru a include mai multe feluri de analize cum ar fi biopsii care sunt greu de citit fără ajutor profesional.
3. Crearea unei aplicații care să permită ca un oarecare utilizator să analizeze probe.
4. Crearea modelelor personalizate.
5. Rezolvarea bug-ului care nu permite schimbarea mărimii de la font.
6. Obținerea unui hardware mai bun pentru a antrena modele și mai complexe sau utilizarea unu *batch\_size* mai mare pentru oricare model
7. Segmentarea unei imagini care este de dimensiuni ample în mai multe imaginii care sa fie *input* pentru model

# Bibliografie

[1] “What do doctors look for in biopsy and cytology specimens?” https://www.cancer.org/treatment/understanding-your-diagnosis/tests/testing-biopsy-and-cytology-specimens-for-cancer/what-doctors-look-for.html (accessed Mar. 11, 2022).

[2] E. Kavlakoglu, “AI vs. Machine Learning vs. Deep Learning vs. Neural Networks: What’s the Difference?,” Jan. 19, 2022. https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks (accessed Mar. 12, 2022).

[3] “GDC dataset for cancer biopsy” https://portal.gdc.cancer.gov/ (accessed Jun. 16, 2022).

[4] F. Furtunescu *et al.*, “Cervical Cancer Mortality in Romania: Trends, Regional and Rural–Urban Inequalities, and Policy Implications,” *Medicina (Mex.)*, vol. 58, no. 1, Art. no. 1, Jan. 2022, doi: 10.3390/medicina58010018.

[5] G. F. Sawaya, K. Smith-McCune, and M. Kuppermann, “Cervical Cancer Screening: More Choices in 2019,” *JAMA*, vol. 321, no. 20, pp. 2018–2019, May 2019, doi: 10.1001/jama.2019.4595.

[6] K. Parveen and M. Clark, *Kumar and Clark’s Clinical Medicine*. 2008.

[7] “Accuracy of the Papanicolaou Test in Screening for and Follow-up of Cervical Cytologic Abnormalities: A Systematic Review,” *J. Low. Genit. Tract Dis.*, vol. 5, no. 1, p. 60, Jan. 2001.

[8] L. Peirson, D. Fitzpatrick-Lewis, D. Ciliska, and R. Warren, “Screening for cervical cancer: a systematic review and meta-analysis,” *Syst. Rev.*, vol. 2, no. 1, p. 35, May 2013, doi: 10.1186/2046-4053-2-35.

[9] M. Stumpe and C. Mermel, “An Augmented Reality Microscope for Cancer Detection,” *Google AI Blog*. http://ai.googleblog.com/2018/04/an-augmented-reality-microscope.html (accessed Mar. 16, 2022).

[10] A. R. Vaka, B. Soni, and S. R. K., “Breast cancer detection by leveraging Machine Learning,” *ICT Express*, vol. 6, no. 4, pp. 320–324, Dec. 2020, doi: 10.1016/j.icte.2020.04.009.

[11] Y. I. A. Rejani and D. S. T. Selvi, “EARLY DETECTION OF BREAST CANCER USING SVM CLASSIFIER TECHNIQUE,” p. 4, 2009.

[12] J.-Y. Chiao, K.-Y. Chen, K. Y.-K. Liao, P.-H. Hsieh, G. Zhang, and T.-C. Huang, “Detection and classification the breast tumors using mask R-CNN on sonograms,” *Medicine (Baltimore)*, vol. 98, no. 19, p. e15200, May 2019, doi: 10.1097/MD.0000000000015200.

[13] T.-L. Dragomir, “Teoria sistemelor 1, cursul 1.”

[14] G. Aurelien, *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, 2nd ed. O’Reilly.

[15] J. Hui, “SSD object detection: Single Shot MultiBox Detector for real-time processing,” *Medium*, Dec. 15, 2020. https://jonathan-hui.medium.com/ssd-object-detection-single-shot-multibox-detector-for-real-time-processing-9bd8deac0e06 (accessed Jun. 08, 2022).

[16] “Python Tutorial.” https://www.tutorialspoint.com/python/index.htm (accessed Mar. 25, 2022).

[17] *Welcome to the Model Garden for TensorFlow*. tensorflow, 2022. Accessed: Jun. 16, 2022. [Online]. Available: https://github.com/tensorflow/models/blob/3c77e654cf939214c9adc4e499346a2a22416447/research/object\_detection/g3doc/tf2\_detection\_zoo.md

[18] “CRIC Searchable Image Database.” https://database.cric.com.br/classification (accessed Jun. 16, 2022).

[19] “Desmos | Graphing Calculator,” *Desmos*. https://www.desmos.com/calculator (accessed Jun. 16, 2022).

Text, letter

Description automatically generated