

Klasifikacija histoloških snimaka crijeva konvolucijskom neuronskom mrežom

Perić, Rujana

Undergraduate thesis / Završni rad

2024

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Electrical Engineering and Computing / Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva**

Permanent link / Trajna poveznica: <https://urn.nsk.hr/urn:nbn:hr:168:033026>

Rights / Prava: [In copyright](#)/[Zaštićeno autorskim pravom.](#)

Download date / Datum preuzimanja: **2025-03-13**



Repository / Repozitorij:

[FER Repository - University of Zagreb Faculty of Electrical Engineering and Computing repozitory](#)



SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1452

**KLASIFIKACIJA HISTOLOŠKIH SNIMAKA CRIJEVA
KONVOLUCIJSKOM NEURONSKOM MREŽOM**

Rujana Perić

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1452

**KLASIFIKACIJA HISTOLOŠKIH SNIMAKA CRIJEVA
KONVOLUCIJSKOM NEURONSKOM MREŽOM**

Rujana Perić

Zagreb, lipanj 2024.

ZAVRŠNI ZADATAK br. 1452

Pristupnica: **Rujana Perić (1191244696)**
Studij: Elektrotehnika i informacijska tehnologija i Računarstvo
Modul: Računarstvo
Mentor: prof. dr. sc. Marko Subašić

Zadatak: **Klasifikacija histoloških snimaka crijeva konvolucijskom neuronskom mrežom**

Opis zadatka:

Ispitati metode za klasifikaciju slika, uz poseban naglasak na klasifikaciju biomedicinskih slika. Prikupiti odgovarajuću bazu histoloških snimaka crijeva visoke rezolucije s poznatom dijagnozom. Implementirati metodu za izdvajanje manjih sličica s odgovarajućim udjelom tkiva u odnosu na pozadinu. Implementirati metodu za klasifikaciju histoloških snimaka crijeva kroz klasifikaciju manjih isječaka iz snimke. Metodu temeljiti na višeslojnoj konvolucijskoj mreži. Ispitati točnost na prikupljenoj bazi slika. Za implementaciju koristiti programski jezik Python.

Rok za predaju rada: 14. lipnja 2024.

Hvala na strpljenju roditeljima i mentoru.

Sadržaj

1. Uvod	3
2. Neuronske mreže	4
2.1. Uvod u neuronske mreže	4
2.2. Duboko učenje u Pythonu	5
3. Priprema dataseta	6
3.1. .NDPI format	6
3.2. Odabir rezolucije slika	7
3.3. Pretprocesiranje slika	8
3.3.1. Detekcija rubova	9
3.3.2. Ekstrahiranje regija interesa	12
3.3.3. Augmentacija podataka	15
4. Organizacija dataseta	17
5. Model konvolucijske neuronske mreže	19
5.1. Set za validaciju	19
5.2. Arhitektura modela konvolucijske neuronske mreže	20
6. Treniranje	23
7. Eksploatacija	24
8. Usporedba s drugim modelima	25
8.1. ResNet arhitektura	25
8.2. EfficientNet arhitektura	26

9. Rezultati i rasprava	28
10. Zaključak	32
Literatura	33
Sažetak	35
Abstract	36

1. Uvod

Klasifikacija medicinskih slika jest pridjeljivanje dijagnoze određenoj slici ovisno o tome koje specifične strukture ili bolesti slika prikazuje. Cilj je koristiti računalo i algoritme dubokog učenja za automatsku klasifikaciju slike. To može uvelike pomoći liječnicima u dijagnosticiranju, planiranju liječenja i praćenju bolesti.

Problem klasifikacije jedan je od glavnih problema u području računalnog vida. Iako postoje razne metode njegovo za rješavanje, najučinkovitijim rješenjem nametnula se izgradnja konvolucijskih neuronskih mreža.

U nastavku ovog rada opisuje se struktura i primjena jednostavne konvolucijske neuronske mreže za klasifikaciju nad ograničenim skupom podataka histoloških snimaka crijeva koje su unaprijed prikupljene i labelirane od strane stručnjaka, te usporedba performansi različitih modela nad istim skupom slika.

Neke snimke prikazuju uzorak patološkog stanja "ulcerozni kolitis" (UC - ulcerative colitis), a neke "kronove bolesti" (CD - Crohn's disease). CD i UC su kronične upalne bolesti crijeva koje karakterizira upala u probavnom sustavu. Liječnici donose odluku dijagnoze bolesti na temelju biopsije (histološke snimke) i kliničkih, laboratorijskih, radiografskih i endoskopskih promatranja [1].

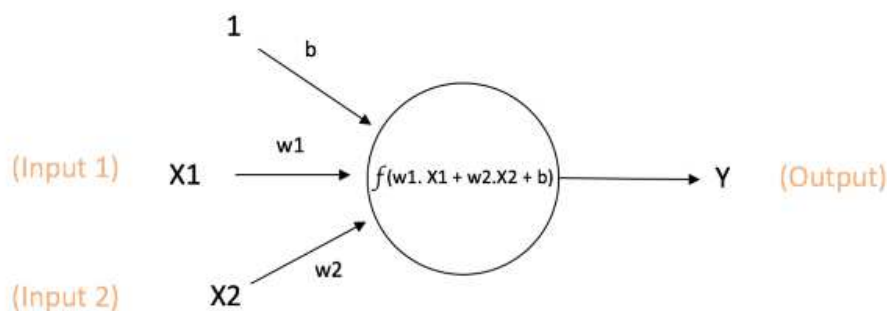
Vidjet ćemo kako konstruirati, istrenirati i koristiti konvolucijsku neuronsku mrežu za klasifikaciju na relativno malenom skupu podataka poput konkretnog skupa biomedicinskih slika, te koliko je takva klasifikacija pouzdana.

2. Neuronske mreže

2.1. Uvod u neuronske mreže

Konvolucijske neuronske mreže (*engl. Convolutional Neural Networks, CNN*) su neuronske mreže koje su bazirane na konvolucijskim operacijama [2].

Neuronske mreže metoda su strojnog učenja i algoritama dubokog učenja s kojima se oponaša način procesuiranja podataka u ljudskom mozgu. Podaci se obrađuju i proslijeđuju kroz neurone (*engl. neurons*), jedinice koje primaju podatak, pridodaju mu neku vrijednost (*težinu, engl. weight*) i proslijeđuju sljedećem neuronu. Neuroni (Slika 2.1.) surađuju kako bi identificirali neki fenomen, prilagođavali težinske vrijednosti fenomena, i došli do zaključka.



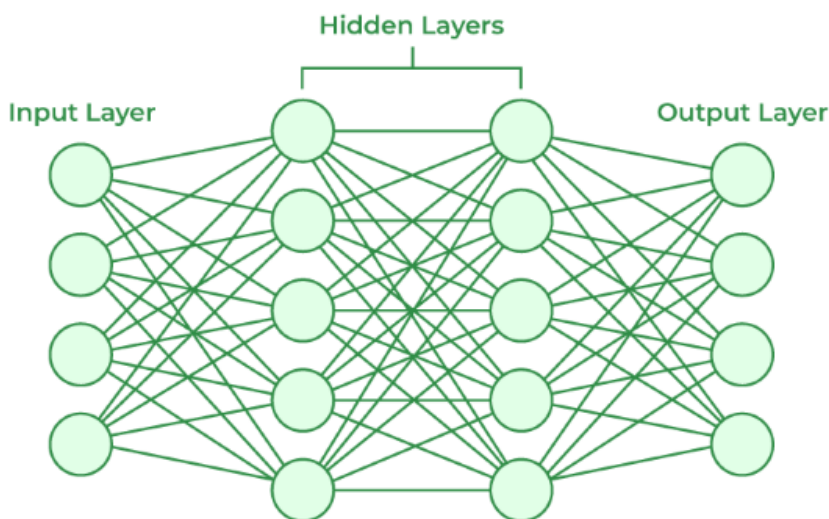
$$\text{Output of neuron} = Y = f(w1.X1 + w2.X2 + b)$$

Slika 2.1. Struktura umjetnog neurona

Svaka se neuronska mreža sastoji od slojeva (*engl. layers*) čvorova (*engl. nodes*), a upravo ti čvorovi su prethodno spomenuti neuroni (2.2.). Ukoliko je izlaz obrađenog ulaza neurona nekog sloja iznad proizvoljno zadane specificirane granične vrijednosti, čvor se aktivira te šalje svoje podatke idućem sloju mreže. Inače, ne šalje nikakve po-

datke.

Za postizanje pouzdane generalizacije i obrade podataka prolaskom kroz čvorove neuronske mreže, mrežu je potrebno istrenirati, tj. ponoviti postupak prilagođavanja težina i dobivanja izlaza mnogo puta nad podacima za koje znamo željeni izlaz. Treniranjem učimo model mreže određenim uzorcima i pouzdanost modela se povećava.



Slika 2.2. Skica strukture neuronske mreže

Najčešća primjena neuronskih mreža je nad klasifikacijskim i regresijskim problemima.

2.2. Duboko učenje u Pythonu

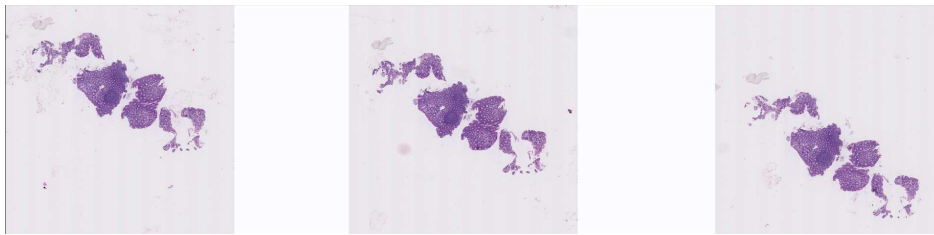
Duboko učenje (*engl. deep learning*) podvrsta je strojnog učenja (*engl. machine learning, ML*) koja koristi višeslojne neuronske mreže. Za potrebe ovog rada korištena je PyTorch biblioteka za duboko učenje.

PyTorch [3] je razvijen od strane Facebookove istraživačke laboratorije AI (FAIR). Ističe se po svojoj fleksibilnosti i intuitivnom radu s tenzorima, omogućavajući dinamičko izračunavanje gradijenata zahvaljujući automatskoj diferencijaciji. Također, nudi odličnu podršku za GPU ubrzanje, što ga čini pogodnim za obuku i implementaciju složenih neuronskih mreža u područjima poput računarskog vida i obrade prirodnog jezika.

3. Priprema dataseta

3.1. .NDPI format

Dataset se sastoji od 78 histoloških snimaka crijeva (Slika 3.1.) u .ndpi formatu labeliranih sa "UC" za ulcerozni kolitis i "CD" za kronovu bolest. Pripadne labele očitane su iz priložene .xlsx datoteke kreirane od strane stručnjaka (liječnika).



Slika 3.1. Histološka snimka crijeva

NDPI format jest slika patološkog uzorka skeniranog pomoću Hamamatsu skenera. NDPI formati mogu se pregledavati pomoću specijaliziranog softvera ili specijalizirane aplikacije.

Kako je NDPI format sličan TIFF formatu, specijalizirani besplatni software [4] služi za konvertiranje .ndpi datoteku u .tiff datoteku bez gubitka rezolucije. Prevođenjem svake datoteke .ndpi formata u .tiff format dobiva se kolekcija sljedećih datoteka:

- slika oznake macro (dodatni podaci)
- slika oznake map (dodatni podaci)
- slika oznake uvećanja x1.25
- slika oznake uvećanja x5
- slika oznake uvećanja x20

3.2. Odabir rezolucije slika

Veličina slike znatno će utjecati na vrijeme potrebno za treniranje i konačnu performansu modela što potvrđuje istraživanje [5]. U primjeni umjetne inteligencije u gastroenterologiji, praksa je koristiti vrlo male rezolucije slika kako bi se uštedjelo na računalnim resursima s cijenom gubitka detalja u slikama.

Iz danog dataseta histoloških snimki gdje su slike oznake uvećanja $\times 1.25$ rezolucije između 3184×6360 i 784×4320 , slike oznake uvećanja $\times 5.0$ rezolucije između 22560×10560 i 19200×9344 i slike oznake uvećanja $\times 20.0$ rezolucije između 109440×41728 i 76800×37376 , odabiremo najmanje moguće slike veličine veće ili jednake veličini pogodnoj za treniranje konvolucijskom neuronskom mrežom (slike oznake uvećanja $\times 1.25$), koje još dodano obrađujemo provodeći postupke predprocesiranja i spremanje u novi, iskoristivi, dataset.

Procesirane slike dimenzija su 64×64 . Ta informacija biti će relevantna za treniranje, testiranje, te izvođenje zaključaka. U budućim istraživanjima je potrebno proširiti saznanja i testirati modele na slikama druge rezolucije jer testiranje samo na danoj rezoluciji predstavlja ograničenje u znanju i razumjevanju modela.

3.3. Pretprocesiranje slika

Stvorena je sljedeća struktura pogodnu za rad s konvolucijskim neuronskim mrežama i PyTorch alatom za duboko učenje:

- **./train** direktorij u kojemu se nalaze slike koje koristimo za treniranje neuronske mreže
- **./test** direktorij u kojemu se nalaze mreži još neviđene slike koje ćemo pokušati ispravno klasificirati pomoću istreniranog modela

Originalni skup podataka prevelik je i pun šuma. Nas zanimaju samo dijelovi slike koji sadrže tkivo na kojem se lijepo razlučuje traženi patološki uzorak. Također, potrebno je paziti i na omjer uzorka naspram pozadine.

Kako bi dobili pogodni dataset razvijeni su algoritmi ekstrahiranja manjih sličica iz originalnog dataseta koji se temelje na sljedećim tehnikama:

- detekcija rubova (*engl. edge detection*)
- ekstrakcija regija interesa (*engl. region of interest - ROI - extraction*)

Naknadno, kao sredstvo kompenzacije za mali skup podataka koristi se augmentacija podataka (*engl. data augmentation*). To je skup metoda transformacija ulaznih podataka radi diverzifikacije skupa podataka. Augmentirane slike su umjetne (*engl. artificial*) te služe kako bi se nadomjestilo skupo prikupljanje i održavanje podataka. Nužno je koristiti augmentacije koje su smislene i koje mogu biti viđene u praksi jer inače model može naučiti na nepostojeće značajke i smanjiti svoje performanse. Korištene augmentacije su:

- rotacija
- promjena razine kontrasta
- zrcaljenje

3.3.1. Detekcija rubova

Detekcija rubova [6] tehnika je procesuiranja slika za alociranje i identificiranje rubova objekata na slici. Koristi se za detektiranje diskontinuiteta u intenzitetu slike. Rubovi objekata obično su regije slike na kojima je zabilježena nagla promjena intenziteta.

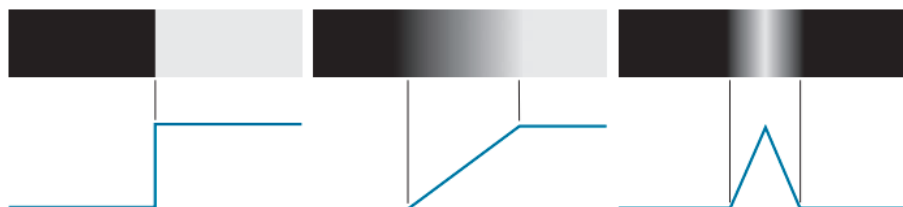
Postoje različite tehnike detektiranja rubova no neke najpopularnije jesu:

- Sobel detekcija rubova
- Canny detekcija rubova
- Laplaceova detekcija rubova
- Prewitt detekcija rubova
- Roberts detekcija rubova

U ovom radu primjenjena je Canny detekcija rubova (Slika 3.2.).



Slika 3.2. Ilustracija produkta postupka detekcije rubova



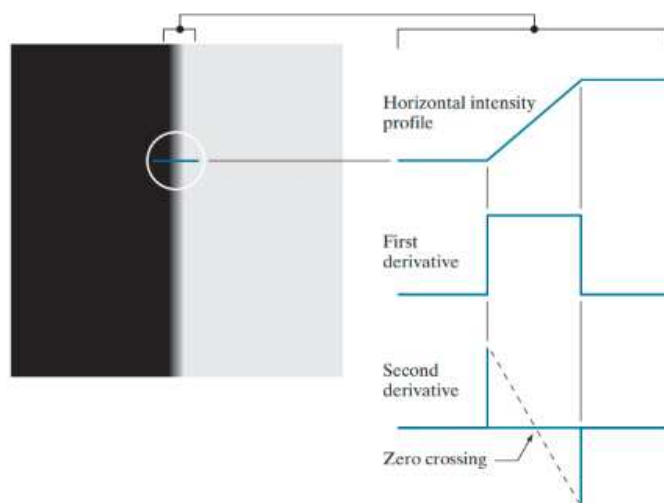
Slika 3.3. S lijeva na desno: korak (*engl. step*) - opisuje naglu promjenu intenziteta, rampa (*engl. ramp*), opisuje postepenu promjenu intenziteta duž neke distance, vrh (*engl. roof*), opisuje maksimum, intenzitet dosiže maksimum te nakon toga ponovno pada

Jedan od glavnih koncepata detekcije rubova su modeli rubova (*engl. edge models*). Modeli rubova (Slika 3.3.) su tehničke konstrukcije koje se koriste za opisivanje različitih

tipova rubova koji se pojavljuju na slikama. Oni kategoriziraju tipove promjena intenziteta.

Intenzitet slike definira se kao svjetlina svakog piksela u crno-bijeloj slici. U slikama u boji intenzitet se može proširiti na intenzitete svakog pojedinog kanala (crvenog, zelenog i plavog, RGB). Prvi korak u detekciji rubova je napraviti crno bijelu sliku.

Modeli rubova detektiraju se primjenom prve i druge derivacije. Prva derivacija mjeri veličinu promjene intenziteta. Računa se tako da se procjenjuje koristeći operatore gradijenta. Druga derivacija mjeri veličinu promjene prve derivacije. Prolaz kroz nulu (*engl. zero-crossing*) druge derivacije rubne su točke, te odgovaraju rubovima objekta na slici. Računa se koristeći Laplaceov operator.



Slika 3.4. Ilustracija uloge prve i druge derivacije u detekciji rubova

Algoritam Canny detekcije rubova sastoji se od sljedećih koraka:

1. Redukcije šumova primjenom Gaussovog zamućenja koja 'zaglađuje' sliku kako bi se izbjegli šumovi visokog intenziteta i vrlo sitni detlji. Jednadžba (3.1) definira Gaussovu funkciju.

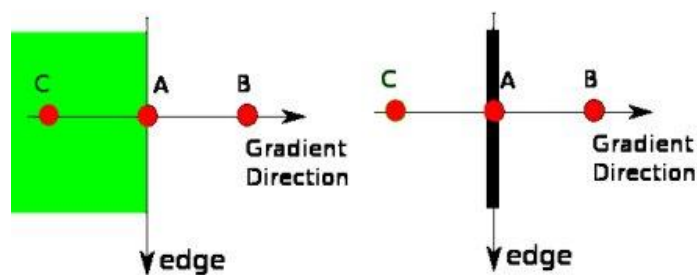
$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (3.1)$$

2. Računanja gradijenta pomoću Sobelove funkcije. Time dobivamo intenzitet slike po x i y osi. Jednadžbe (3.2) (3.3) definira Sobelovu funkciju.

$$G = \sqrt{G_x^2 + G_y^2} \quad (3.2)$$

$$\theta = \arctan\left(\frac{G_y}{G_x}\right) \quad (3.3)$$

3. Potiskivanja, tj. otklanjanje lokalnih maksimuma (Slika 3.5.) koji se ne nalaze u smjeru gradijenta. Ovaj korak stanjuje rubove i uklanja lažne odgovore. Svaki piksel koji nije lokalni maksimum u smjeru gradijenta sigurno nije rub.



Slika 3.5. Ilustracija uloge potiskivanja u Canny detekciji rubova

4. Dvostrukog praga. Rubni pikseli se označavaju pomoću dvostrukog praga. Ovaj korak klasificira rubove u jake, slabe i ne-rubove na temelju dvaju pragova: visokog i niskog.
5. Histereze, što uključuje prelaženje slika kako bi se ustvrdilo koji su slabi rubovi povezani s jakim rubovima. Zadržavaju se samo slabi rubovi povezani s jakim rubovima jer se smatraju pravim rubovima.

Rezultat Canny detekcije nad histološkom snimkom dan je slikom (Slika 3.6.).



Slika 3.6. Histološka snimka provedena kroz detekciju rubova

3.3.2. Ekstrahiranje regija interesa

Regije interesa (*engl. regions of interest - ROI*) su uzorci unutar skupa podataka koji sadrže neko željeno svojstvo. Primjena algoritama koji traže ROI učestale su u svim granama medicinske dijagnostike, računalnom vidu, optičkom prepoznavanju znakova, geografskim informacijskim sustvima.

Regije interesa na slikama iz danog dataseta su regije koje sadrže patološki uzorak. Detekcijom rubova patološkog uzorka napravili smo prvi korak u eliminiranju praznog prostora koji nije od interesa. Naše regije interesa biti će upravo područja između detektiranih rubova, tj., unutrašnjost patološkog uzorka.

Algoritam u nastavku prikazuje programsku logiku u pozadini ekstrahiranja regija interesa, a slika (Slika 3.7.) rezultate provedbe algoritma.

```
def ROI_detection(self, edges, min_size=(64, 64), white_threshold=200,
target_percentage=30, tolerance=5, output_file=None):

    image = Image.open(self.image_path)

    # Convert image to a numpy array
    image_np = np.array(image)
    contours, _ = cv2.findContours(

edges, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)

    # Sort contours by area in descending order and take the two largest
    contours = sorted(contours, key=cv2.contourArea, reverse=True)

    # Draw bounding boxes around contours
    image_with_boxes = image_np.copy()
    for contour in contours:
        x, y, w, h = cv2.boundingRect(contour)
```

```

if w >= min_size[0] and h >= min_size[1]:
    cv2.rectangle(

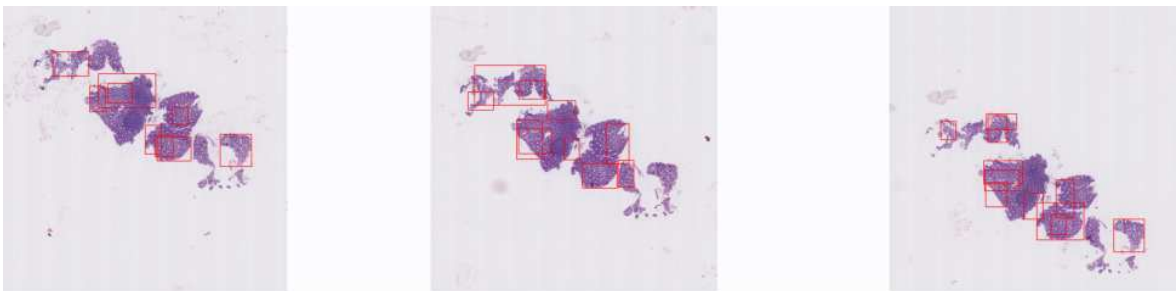
        image_with_boxes, (x, y), (x + w, y + h), (255, 0, 0), 2)

# Extract and resize cropped regions
selected_regions = []
for contour in contours:
    x, y, w, h = cv2.boundingRect(contour)
    if w >= min_size[0] and h >= min_size[1]:
        cropped_region = image_np[y:y+h, x:x+w]
        white_area_percentage =

            calculate_white_area_percentage(cropped_region,

                threshold=white_threshold)
        resized_region = cv2.resize(cropped_region, (64, 64))
        if abs(white_area_percentage - target_percentage) <= tolerance:
            selected_regions.append(resized_region)

```



Slika 3.7. Histološka snimka sa prepoznatim regijama interesa

Za implementaciju metode ekstrakcije regije interesa korištena je OpenCV biblioteka. OpenCV [7] je javna biblioteka za računalni vid, strojno učenje i procesiranje slika.

No, pri kreiranju novih sličica nije dovoljno niti ispravno izrezati regije interesa.

Želimo da naš novi dataset bude homogen s obzirom na sadržaj svake sličice kako bismo omogućili modelu da trenira i uči ono što nas zaista zanima, a to je patogeni uzorak. Ukoliko naš dataset nije homogen model može početi prepoznavati neke druge, nerelevantne uzorke u sličicama i nauči nešto besmisleno i nepostojeće.

Potrebno je definirati funkciju koja selektira regije interesa sa dovoljnom prisutnošću tkiva. Dogovorno je ta prisutnost tkiva 70

Algoritam (Slika 3.9.) prikazuje postupak za selektiranje regija interesa sa prisutnošću tkiva 0.7. To čini pretvaranjem slike u binarni format i definiranjem 'bijelih piksela'. Definirano je da je bijeli piksel piksel praga vrijednosti 200 (piksela vrijednosti veće od 200 su bijeli). Rezultate provedbe algoritma na slici sa označenim regijama interesa prikazani su na slici (Slika 3.8.).



Slika 3.8. Primjer konačno odabranih sličica dimenzija 64×64

```

def calculate_white_area_percentage(image, threshold=200):

    """
    Calculate the percentage of white pixels in the image.

    Parameters:
    image (numpy.ndarray): Input image array.
    threshold (int): Pixel value threshold to consider as white.

    Returns:
    float: Percentage of white pixels.
    """
    # Convert image to binary based on threshold
    _, binary_image = cv2.threshold(image, threshold, 255, cv2.THRESH_BINARY)

    # Calculate percentage of white pixels
    white_pixels = np.sum(binary_image == 255)
    total_pixels = image.size
    percentage_white = (white_pixels / total_pixels) * 100

    return percentage_white

```

Slika 3.9. Algoritam selektiranja regija s određenom prisutnošću tkiva

3.3.3. Augmentacija podataka

Augmentacija podataka (*engl. data augmentation*) je proces umjetnog generiranja novih podataka za obuku modela strojnog učenja.

Modeli zahtjevaju velike i raznolike skupove podataka za početno treniranje kako bi što bolje uhvatili tražene uzorke. No, veliki je problem, pogotovo u medicinskoj analitici, prikupiti dovoljno velik skup podataka. Augmentacija će umjetno povećati skup podataka uvođenjem malih izmjena u izvorne podatke.

Neke od prednosti augmentacije su:

- poboljšanje performansi modela (omogućuje modelu da susretne više različitih značajki i da u budućnosti bolje generalizira dotad neviđene podatke)

- smanjenje ovisnosti o podacima (priprema podataka za modele može biti skupa i dugotrajna, pa augmentacija uvelike smanjuje ovisnos o velikim skupovima podataka)
- ublažavanje prenaučivosti (prenaučivost je nepoželjno ponašanje ML-a gdje model može jako dobro klasificirati podatke za treniranje, ali ne i dosad neviđene podatke jer je pronašao uzorke jako specifične za dani skup)



Slika 3.10. Umjetno stvorene sličice koje imaju male varijacije istog sadržaja

4. Organizacija dataseta

Primarni pristup oragnizaciji dataseta medicinskih slika je takoznavi 'patient-wise split'.

Takav pristup svrstava stakalca pojedinih pacjenata ili u train skupinu ili u test skupinu no svi pacjenti koji imaju više od jednog stakalca nalazit će se u train setu, dok svi pacjenti koji imaju točno jedno stakalce se nalaze u test setu, tj. nad točno tim pacjentima moći će se napraviti predviđanje dijagnoze. Analizu podataka po pacjentima prikazuje dana tablica (Slika 4.1.).

Pacijent	Dijagnoza	Broj snimaka	Skup podataka
3	CD		1 TEST
5	CD		1 TEST
9	CD		1 TEST
11	CD		1 TEST
12	CD		1 TEST
13	UC		1 TEST
16	UC		1 TEST
17	CD		1 TEST
20	UC		1 TEST
21	UC		1 TEST
23	UC		1 TEST
26	UC		1 TEST
28	UC		1 TEST
29	CD		1 TEST
30	UC		1 TEST
32	CD		1 TEST
33	CD		1 TEST
35	UC		1 TEST
37	UC		1 TEST
38	CD		1 TEST
39	CD		1 TEST
40	UC		1 TEST
41	UC		1 TEST
43	UC		1 TEST
4	CD		2 TRAIN
7	CD		2 TRAIN
8	CD		2 TRAIN
18	UC		2 TRAIN
24	UC		2 TRAIN
25	UC		2 TRAIN
27	UC		2 TRAIN
34	CD		2 TRAIN
36	UC		2 TRAIN
1	CD		3 TRAIN
6	CD		3 TRAIN
10	CD		3 TRAIN
14	UC		3 TRAIN
31	UC		3 TRAIN
2	CD		4 TRAIN
15	UC		4 TRAIN
19	CD		4 TRAIN
22	CD		4 TRAIN
42	UC		5 TRAIN

Slika 4.1. Podjela pacijenata u skup za treniranje i skup za testiranje

Iako rijedi u praksi, upravo je taj način bolji za korištenje našeg modela jer se neće moći dogoditi da model pogriješi i za jednog pacjenta odredi dvije dijagnoze (određena stakalca koja se odnose na pacjenta labelira sa 'uc', a određena sa 'cd').

Također, ova metoda podjele eliminira curenje podataka i potiče generalizacijsku vještinu modela na još ne viđenim pacijentima.

Train skup sastoji se od 4.053 sličice (0.6448 od ukupnog broja sličica) 19 različitih pacijenata.

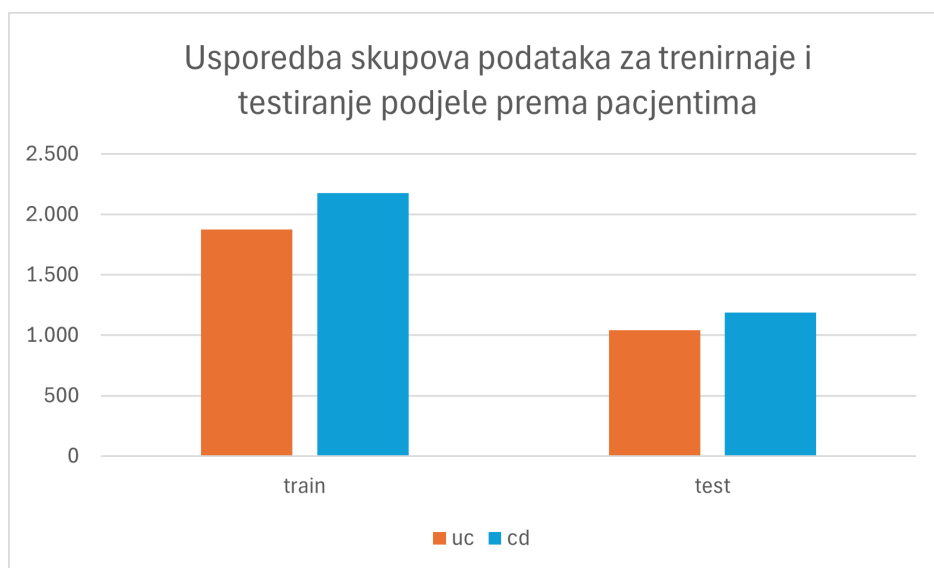
Test skup sastoji se od 2.233 sličice (0.3552 od ukupnog broja sličica) 24 različita pacijenta.

Uočavamo da to približno dobro prati praksu podjele slika u skupove za treniranje i testiranje u omjeru 70:30.

Od slika iz skupa za treniranje 2177, tj. 0.5371, je labelirano sa 'cd' (eng. Chron's disease), a 1876, tj. 0.4629, je labelirano sa 'uc' (eng. Ulcerative colitis). Brojevi primjaka po klasama su približno jednaki.

Od slika iz skupa za testiranje 1190, tj. 0.5329, je labelirano sa 'cd', a 1043, tj. 0.4671, je labelirano sa 'uc'. Brojevi primjaka po klasama su približno jednaki.

Skupovi unutar sebe relativno su uravnoteženi (Slika 4.2.) kada su u pitanju brojevi snimaka svake klase, no postoji razlika od 0.0742 uspoređujući brojeve primjeraka svake od klasa u skupovima train i test.



Slika 4.2. Raspodjela sličica u skupu za treniranje i skupu za testiranje

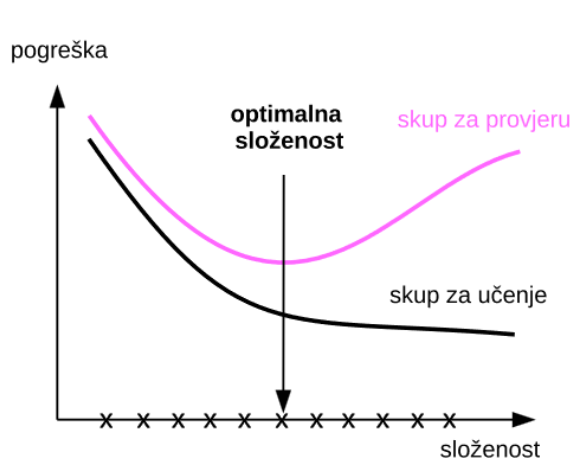
5. Model konvolucijske neuronske mreže

5.1. Set za validaciju

Kako bismo dobili što pouzdaniji model potrebno je dodati set za validaciju. Njega smo konstruirali ekstrakcijom određenog broja slika iz skupa za treniranje.

Set za validaciju [8] služi kako bismo evaluirali performansu modela u toku treniranja i spriječili prenaučenosť. Pojavu koju karakterizira vrlo mala pogreška na setu za treniranje, no velike pogreške na setu za testiranje jer je model 'krivo naučio, pre analizirao' uzorke seta za treniranje.

Što dulje treniramo model to će on biti pouzdaniji, no u jednom trenutku pogreška predviđanja na testu za validaciju počeo će rasti. Tada trebamo prekinuti treniranje, a optimalno stanje modela biti će točka u kojoj test za validaciju ima minimalnu pogrešku, kao što je ilustrirano na slici (Slika 5.1.).



Slika 5.1. Treniranje i pogreške na skupovima za trening i validaciju

5.2. Arhitektura modela konvolucijske neuronske mreže

Konvolucijska neuronska mreža [2] neuronska je mreža koja se primarno koristi za probleme klasifikacije i rješavanje problema računalnog vida primjenjujući principe linearne algebre i matričnog množenja za ekstrakciju i identificiranje značajki iz slika.

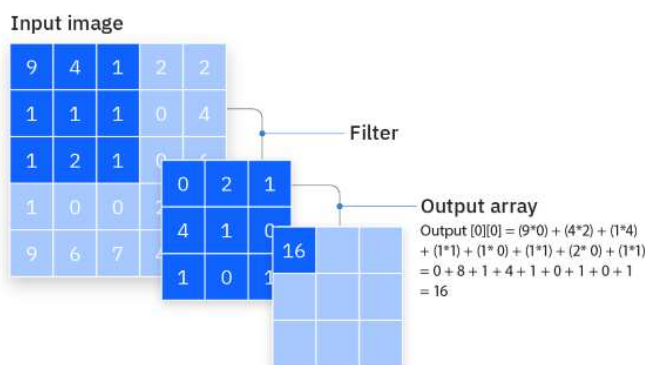
Sastoji se od:

- ulaznog sloja (*engl. input image*)
- jednog ili više konvolucijskih slojeva (*engl. convolutional layer*)
- jednog ili više objedinjujućih slojeva koji slijede konvolucijski sloj (*engl. pooling layer*)
- potpuno povezanih slojeva (*engl. fully-connected, FC, layer*)

Konvolucijski sloj prima ulazne podatke (slike) koji su u obliku matrice. Ukoliko su te slike u boji imat ćemo 3D matricu piksela.

Posjeduje strukturu detektor značajki (*engl. feature detector, filter, kernel*) koji je po svojoj strukturi 2D matrica težina (u praksi obično jednostavna 3×3 matrica). Ona klizi (*engl. stride*) po danom ulaznom podatku, i za svaki segment slike koji pokriva izvršava konvoluciju.

Konvolucija množi elemente matrice koji su na istim pozicijama $m[i][j] * k[i][j]$ i na kraju zbraja sve izmnožene elemente u konačnu skalarnu veličinu kao što je prikazano na slici (Slika 5.2.)



Slika 5.2. Računanje konvolucije zadanog sloja

Izlaz iz konvolucijskog sloja poznat je kao mapa značajki (*engl. feature map, activation map, convolved feature*).

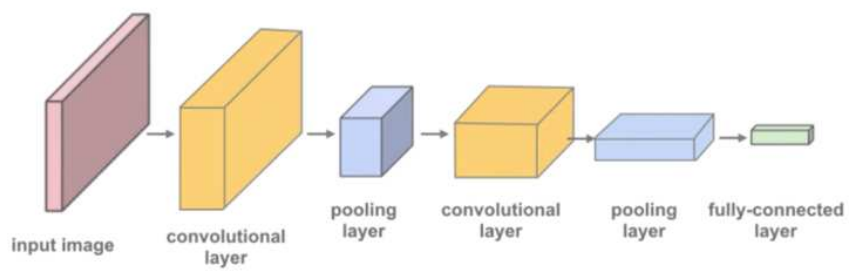
Težine za vrijeme obrade jedne slike su konstantne. Ta značajka naziva se i djeljenje parametara (*engl. parameter sharing*). Težine se prilagođavaju tokom treniranja u procesima "propagacije unazad" (*engl. backpropagation*) ili "smanjenja gradijenta" (*engl. gradient descent*).

Pri konstrukciji i prije treniranja modela potrebno je definirati sljedeće parametre:

- broj detektora značajki (*engl. number of filters*) koji u slučaju slike u boji mora biti 3 (da pokrije sva tri sloja dubine RGB slike)
- pomak (*engl. stride*), distanca koju će u jednom koraku proći detektor značajki
- obrub (*engl. padding*) koji rubove ulazne matrice proširuje i pridjeljuje vrijednost nula, kako bi se osiguralo da izlaz konvolucije bude istih dimenzija kao i ulaz konvolucije
- seriju (*engl. batch*) koju definiramo kao optimlanu količinu slika koja odjednom prolazi kroz mrežu
- broj epoha (*engl. epoch*) gdje jedna epoha predstavlja jedan prolazak (unaprijed i unazad) svih uzoraka iz skupa za treniranje
- stopu učenja (*engl. learning rate*), parametar koji određuje kojim se intenzitetom mijenjaju težine modela prema najmanjem gubitku

Bitno je odabrati vrijednosti parametara [9] tako da proces treniranja daje optimlane rezultate u razumnom vremenu. Optimizacija se postiže uzastopnim isprobavanjem i podešavanjem parametara na konkretnom modelu.

Neuronska mreža konstruirana za potrebe ovog istraživanja sastoji se od dva konvolucijska sloja koji završavaju objedinjujućim slojem i završnog potpuno povezanog sloja. Ilustracija arhitekture mreže vidljiva je na slici (Slika 3.4.).



Slika 5.3. Ilustracija jednostavne konvolucijske neuronske mreže konstruirane za potrebe klasifikacije histoloških snimaka

6. Treniranje

Treniranje [10] je postupak učenja modela rješavanja određenog zadatka na temelju danog skupa podataka za treniranje. U našem slučaju skup podataka sadrži i oznake (*engl.label*) pripadnosti kategorijama pa se stoga radi o nadziranom učenju (*engl.supervised learning*).

Treniranje se sastoji od skupa podataka za učenje, modela, funkcije gubitka (*engl.supervised learning*), optimizatora (*engl.optimizer*) te skupa hiperparametara (*engl.hyperparameter*) za učenje.

Korišteni optimizator je Adam s početnom stopom učenja 0.001.

```
opt_func = torch.optim.Adam
lr = 0.001
```

Model je treniran u 10 epoha, s veličinom serije od 128 sličica.

```
num_epochs = 10
batch_size = 128
```

Postupak treniranja pokreće se funkcijom 'fit'.

```
history = fit(num_epochs, lr, model, train_dl, val_dl, opt_func)
```

7. Eksploatacija

Praćenje općenite pouzdanosti (*engl. accuracy*) mjera je kvalitete modela, no sama primjena modela biti će na dosad neviđenim slikama.

Cilj je koristiti trenirani model na način da ulazni podatak bude originalna snimka, a izlaz labela 'CD' ili 'UC' kao što je prikazano kodom 7.1.

```
def predict_image(img, model):

    # Convert to a batch of 1
    xb = to_device(img.unsqueeze(0), device)

    # Get predictions from model
    yb = model(xb)

    # Pick index with highest probability
    _, preds = torch.max(yb, dim=1)

    # Retrieve the class label
    return dataset.classes[preds[0].item()]

print('Label:', dataset.classes[label], ', Predicted:', predict_image(img, model))
```

Slika 7.1. Funkcija `predict_image` kojom se koristi istrenirani model

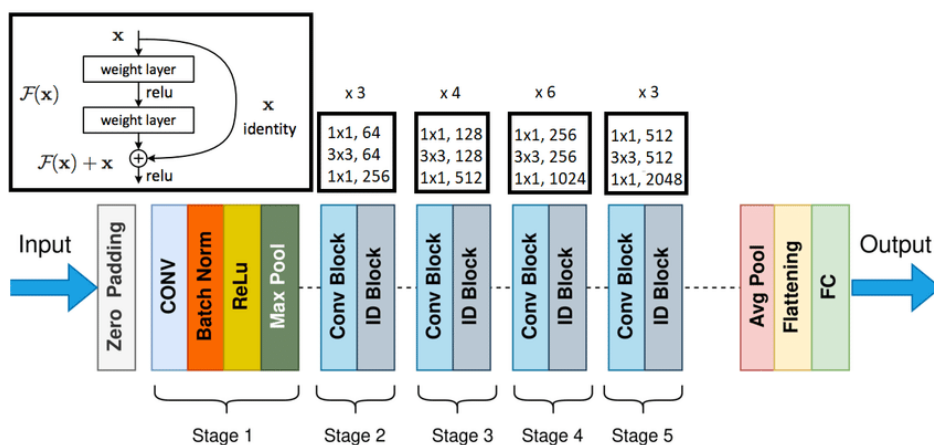
8. Usporedba s drugim modelima

Konstruirani model za potrebe ovog istraživanja vrlo je štur i bazičan, stoga su testirane i performanse popularnih arhitektura ResNet i EfficientNet konvolucijskih neuronskih mreža nad istim datasetom.

8.1. ResNet arhitektura

ResNet je duboka konvolucijska neuronska arhitektura [11] koju karakterizira dubina i korištenje rezidualnih konekcija ilustrirana slikom (Slika 8.1.).

Rezidualne konekcije (*engl. residual connections*) rješavaju problem nestajanja gradijenta u jako dubokim mrežama.



Slika 8.1. Ilustracija ResNet arhitekture

Rezidualni blok je osnovni gradivni blok rezidualne mreže. To je blok uzastopnih višestrukih konvolucijskih slojeva sa rezidualnim konekcijama. Izlaz rezidualnog bloka je $F(x)+x$ gdje $F(x)$ reprezentira izlaz konvolucijskih slojeva bloka, a x ulaz u konvolucijske slojeve (blok).

Funkcija identiteta (*engl. identity function*) omogućuje da mreža uči samo rezidualne, što optimizira duboke mreže i čini ih jednostavnijim za treniranje.

8.2. EfficientNet arhitektura

EfficientNet [12] je arhitektura konvolucijske neuronske mreže koja se razlikuje od drugih arhitektura po načinu skaliranja modela.

Konvolucijske neuronske mreže mogu se skalirati:

- po dubini (dodavanjem novih slojeva)
- po širini (dodavanjem neurona u slojeve)
- promjenom rezolucije slika

EfficientNet kombinira sva tri načina skaliranja istovremeno (Slika 8.2.), uniformno za faktor θ takozvanim složenim skaliranjem (*engl. compound scaling, compound coefficient*) formulom (8.1).

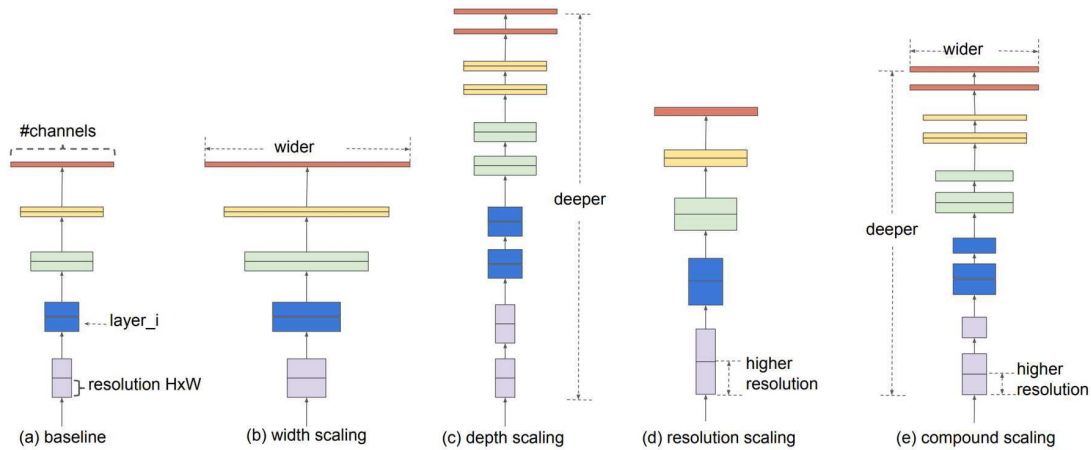
$$\begin{aligned} \text{dubina} : d &= \alpha \phi \\ \text{širina} : w &= \beta \phi \\ \text{rezolucija} : r &= \gamma \phi \\ \alpha \cdot \beta^2 \cdot \gamma^2 &\approx 2 \\ \alpha \geq 1, \beta \geq 1, \gamma \geq 1 \end{aligned}$$

Slika 8.2. Jednadžbe skaliranja

Koeficijent θ je korisnički definiran i određuje koliko se dodatno želi dodijeliti resursa modelu, dok su koeficijenti α , β , γ konstante koje određuju kako će se dodatni resursi raspodijeliti.

FLOPS (*engl. Floating Point Operations*) konvolucijskih operacija je proporcionalan s $\alpha, \beta^2, \gamma^2$.

Većina računanja u konvolucijskim neuronskim mrežama konvolucijske operacije. Možemo aproksimativno izračunati koliko će se FLOPS-a povećati u odnosu na bazni model za dani θ po formuli (8.1).



Slika 8.3. Skaliranje modela. (a) osnovni model, (b) – (d) skaliranje pojedine dimenzije, (e) složeno skaliranje

$$(\alpha \cdot \beta^2 \cdot \gamma^2)\theta \approx 2^\theta \quad (8.1)$$

Bazni model je EfficientNet B0 dok se povećanjem θ dobivaju modeli B1 – B7.

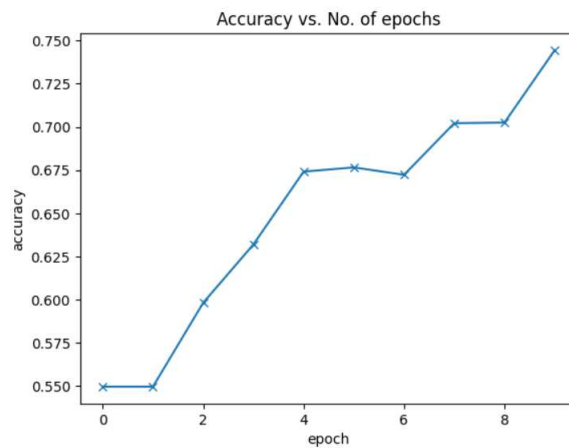
Usporedba performansi varijanti EfficientNet-a i ostalih popularnih modela je vidljiva na slici (Slika 8.4.). Prikazane su performanse EfficientNet modela i nekih popularnih modela na ImageNet skupu podataka. Metrike usporedbe su Top - 1 Acc. i Top - 5 Acc, dok Params i FLOPs prikazuju računalnu zahtjevnost.

Model	Top-1 Acc.	Top-5 Acc.	#Params	Ratio-to-EfficientNet	#FLOPs	Ratio-to-EfficientNet
EfficientNet-B0	77.1%	93.3%	5.3M	1x	0.39B	1x
ResNet-50 (He et al., 2016)	76.0%	93.0%	26M	4.9x	4.1B	11x
DenseNet-169 (Huang et al., 2017)	76.2%	93.2%	14M	2.6x	3.5B	8.9x
EfficientNet-B1	79.1%	94.4%	7.8M	1x	0.70B	1x
ResNet-152 (He et al., 2016)	77.8%	93.8%	60M	7.6x	11B	16x
DenseNet-264 (Huang et al., 2017)	77.9%	93.9%	34M	4.3x	6.0B	8.6x
Inception-v3 (Szegedy et al., 2016)	78.8%	94.4%	24M	3.0x	5.7B	8.1x
Xception (Chollet, 2017)	79.0%	94.5%	23M	3.0x	8.4B	12x
EfficientNet-B2	80.1%	94.9%	9.2M	1x	1.0B	1x
Inception-v4 (Szegedy et al., 2017)	80.0%	95.0%	48M	5.2x	13B	13x
Inception-resnet-v2 (Szegedy et al., 2017)	80.1%	95.1%	56M	6.1x	13B	13x
EfficientNet-B3	81.6%	95.7%	12M	1x	1.8B	1x
ResNeXt-101 (Xie et al., 2017)	80.9%	95.6%	84M	7.0x	32B	18x
PolyNet (Zhang et al., 2017)	81.3%	95.8%	92M	7.7x	35B	19x
EfficientNet-B4	82.9%	96.4%	19M	1x	4.2B	1x
SENet (Hu et al., 2018)	82.7%	96.2%	146M	7.7x	42B	10x
NASNet-A (Zoph et al., 2018)	82.7%	96.2%	89M	4.7x	24B	5.7x
AmoebaNet-A (Real et al., 2019)	82.8%	96.1%	87M	4.6x	23B	5.5x
PNASNet (Liu et al., 2018)	82.9%	96.2%	86M	4.5x	23B	6.0x
EfficientNet-B5	83.6%	96.7%	30M	1x	9.9B	1x
AmoebaNet-C (Cubuk et al., 2019)	83.5%	96.5%	155M	5.2x	41B	4.1x
EfficientNet-B6	84.0%	96.8%	43M	1x	19B	1x
EfficientNet-B7	84.3%	97.0%	66M	1x	37B	1x
GPipe (Huang et al., 2018)	84.3%	97.0%	557M	8.4x	-	-

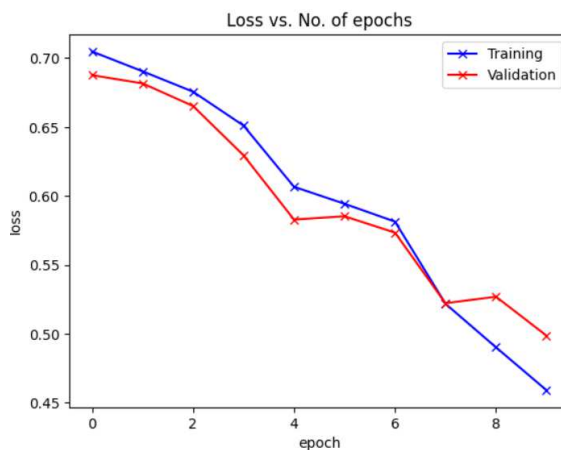
Slika 8.4. Usporedba EfficientNet modela na ImageNet skupu podataka.

9. Rezultati i rasprava

Rezultati treniranja proizvoljno konstruirane konvolucijske mreže sa dva konvolucijska sloja i jednim potpuno povezanim slojem uočljivi su na grafu (9.1.) i grafu (9.2.).



Slika 9.1. Pouzdanost proizvoljno konstruiranog konvolucijskog modela s obzirom na broj epoha na setu za testiranje



Slika 9.2. Vrijednost funkcije gubitka na setu za treniranje i validaciju proizvoljno konstruiranog konvolucijskog modela s obzirom na broj epoha na setovima za treniranje i validaciju

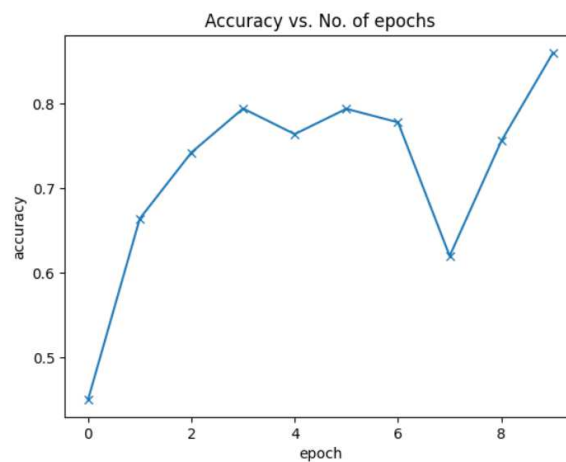
Vrijednost gubitka seta za treniranje kreće s vrijednošću 0.7048 i smanjuje se do vrijednosti 0.4592. Smanjivanje ukazuje na to da model uči s vremenom.

Vrijednost gubitka seta za validaciju kreće s vrijednošću 0.6876 i smanjuje se do vrijednosti 0.4987 s manjim odstupanjima. To ukazuje na to da se model poboljšava u generalizaciji novih podataka.

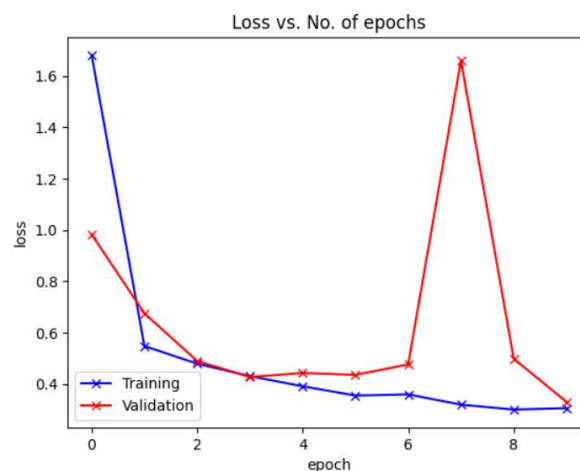
Točnost provjere na setu za testiranje počinje od 0,5497 (54,97%) i povećava se na 0,7444 (74,44%). Ovo povećanje ukazuje na to da model ispravno klasificira više validacijskih uzoraka kako treniranje napreduje.

Konačna točnost od 75% smatra se zadovoljavajućom za početak, no ostavlja nam još mnogo prostora za rad.

Rezultati treniranja konvolucijske mreže sa ResNet18 arhitekturom uočljivi su na grafu (9.3.) i grafu (9.4.).



Slika 9.3. Pouzdanost ResNet modela s obzirom na broj epoha na setu za testiranje



Slika 9.4. Vrijednost funkcije gubitka na setu za treniranje i validaciju ResNet modela s obzirom na broj epoha na setovima za treniranje i validaciju

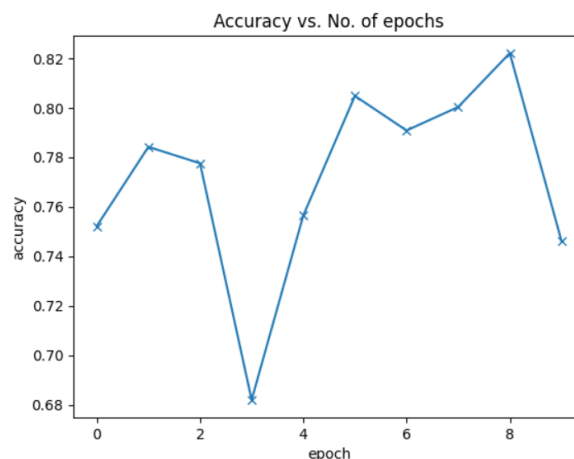
Vrijednost gubitka seta za treniranje kreće s vrijednošću 1.6807 i smanjuje se do vrijednosti 0.2997. Smanjivanje ukazuje na to da model vrlo dobro uči s vremenom.

Vrijednost gubitka seta za validaciju kreće s vrijednošću 0.9821 i smanjuje se do vrijednosti 0.3295 s većim odstupanjem na 7. epohi. To ukazuje na to da se model dobro poboljšava u generalizaciji novih podataka s naznakom da u sedmoj epohi potencijalno dolazi do prenaučivosti te da je model potrebno dalje usavršavati i pronaći bolje hiperparametre.

Točnost provjere na setu za testiranje počinje od 0.4503 (45,03%) i povećava se na 0,8602 (86.02%). Prema ovim podacima sigurni smo da nakon 5. epohe dolazi do prenaučivosti modela. Model se u zadnjoj epohi oporavlja od prenaučivosti, no potrebno je provesti dodatna testiranja kako bi se definiralo ponašanje modela u procesu treniranja na drugačijem broju epoha, što bi ustvrdilo razlog naglog skoka točnosti u zadnjoj epohi.

Točnost modela 86,02% označava da je model vrlo pogodan za treniranje ovakvih skupova podataka, no rezultat treba uzeti s oprezom i provesti daljnja istraživanja.

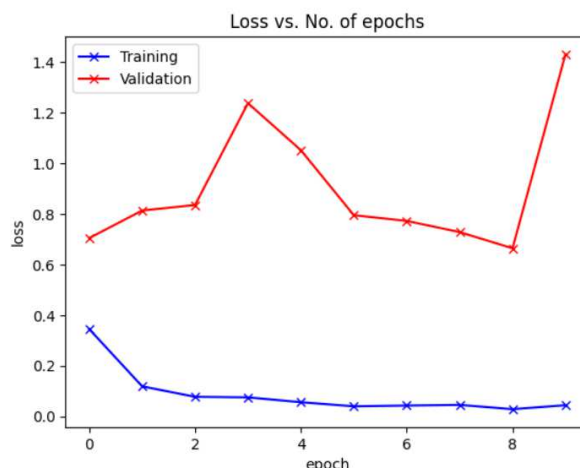
Rezultati treniranja konvolucijske mreže sa EfficientNetB0 arhitekturom uočljivi su na grafu (9.5.) i grafu (9.6.).



Slika 9.5. Pouzdanost EfficientNet modela s obzirom na broj epoha na setu za testiranje

Vrijednost gubitka seta za treniranje kreće s vrijednošću 0.3448 i smanjuje se do vrijednosti 0.0438. Smanjivanje ukazuje na to da model vrlo dobro uči s vremenom.

Vrijednost gubitka seta za validaciju varira i raste već na 3. epohi. To ukazuje na to da u modelu dolazi do prenaučivosti. Model se u kasnijim epohama uspjeva oporaviti



Slika 9.6. Vrijednost funkcije gubitka na setu za treniranje i validaciju EfficientNet modela s obzirom na broj epoha na setovima za treniranje i validaciju

od prenaučivosti, no moguće je da se doseže manja točnost zbog početnih nestabilnosti.

Točnost provjere na setu za testiranje počinje od 0,6820 (68,20%) i povećava se na 0,8220 (82,20%). Sigurni smo da nakon 3. epohe dolazi do prenaučivosti modela. Model se oporavlja od prenaučivosti, i usprkos početnim nečistoćama dolazi do točnosti od 82,20%, što nam govori da je model vrlo pogodan za treniranje ovakvih skupova podataka. Potrebno je provesti dodatna testiranja kako bi se definiralo ponašanje modela u procesu treniranja na drugačijem broju epoha, što bi potvrdilo daljnji pad točnosti i lokalni maksimum od 82,20%.

Rezultat treba uzeti s oprezom i provesti daljnja istraživanja.

10. Zaključak

Uočeno je da najbolje performanse daje model ResNet arhitekture, što je bilo i očekivano ukoliko pogledamo benchmark performansi arhitektura na nekim vrlo popularnim datasetovima (CIFAR-10) [13].

Također je uočeno da najgore performanse daje proizvoljno definirani model zbog svoje trivijalnosti.

S malim skupom podataka teško je donositi odluke o samoj arhitekturi neuronske mreže za dijagnozu bolesti. Česta je pojava da mreža nauči nepostojeće uzorke koji nisu od našeg interesa za dijagnostiku bolesti.

Bolji rezultati bi se potencijalno mogli dobiti poboljšanjem dataseta tako da sličice za treniranje jasno prikazuju patološku tvorevinu unutar tkiva, a ne komadić tkiva za kojeg nismo niti sigurni sadrži li patološku tvorevinu. Također, potrebno je unaprijediti parametre neuronskih mreža koje su korištene za treniranje, kao što su broj epoha, veličina serije, stopa učenja, normalizacija podataka, L1 i L2 regularizacija, dodavanje ispadanja.

Literatura

- [1] K. Geboes, “Histopathology of crohn’s disease and ulcerative colitis”, u *Inflammatory bowel disease*, 2003., str. 210–28.
- [2] IBM, “What are convolutional neural networks?” <https://www.ibm.com/topics/convolutional-neural-networks>, [mrežno; stranica posjećena: srpanj 2024.].
- [3] C. B. D. S. A. B. Suraj Subramanian, Seth Juarez, “Learn the basics”, <https://pytorch.org/>, [mrežno; stranica posjećena: travanj 2024.].
- [4] IMNC, “ndpertools”, <https://www.imnc.in2p3.fr/pagesperso/deroulers/software/ndpertools/>, [mrežno; stranica posjećena: ožujak 2024.].
- [5] S. A. H. P. H. S. P. M. A. R. Vajira Thambawita, Inga Strümke, “Impact of image resolution on deep learning performance in endoscopy image classification: An experimental study using a large dataset of endoscopic images”, 2021. <https://doi.org/10.3390/diagnostics11122183>
- [6] C. Writer, “Edge detection in image processing: An introduction. roboflow”, <https://blog.roboflow.com/edge-detection/>, [mrežno; stranica posjećena: lipanj 2024.].
- [7] OpenCV, “Contours”, https://docs.opencv.org/4.x/d4/d73/tutorial_py_contours_begin.html, [mrežno; stranica posjećena: lipanj 2024.].
- [8] J. Brownlee, “What is the difference between test and validation datasets?” <https://machinelearningmastery.com/difference-test-validation-datasets/>, [mrežno; stranica posjećena: lipanj 2024.].

- [9] I. S. M. T. Eva Tuba, Nebojša Bačanin, “Artificial intelligence: Theory and applications”, u *Convolutional Neural Networks Hyperparameters Tuning*, 2021., str. 65–84.
- [10] F. Hrabar, “Klasifikacija histoloških snimaka crijeva”, str. 26–29, 2024.
- [11] S. R. J. S. Kaiming He, Xiangyu Zhang, “Deep residual learning for image recognition”, <https://paperswithcode.com/paper/deep-residual-learning-for-image-recognition>, [mrežno; stranica posjećena: lipanj 2024.].
- [12] Q. V. L. Mingxing Tan, “Efficientnet: Rethinking model scaling for convolutional neural networks”, <https://arxiv.org/abs/1905.11946v5>, [mrežno; stranica posjećena: lipanj 2024.].
- [13] P. W. Code, “Image classification on cifar-10”, (https://paperswithcode.com/sota/image-classification-on-cifar-10?tag_filter=3%2C5), [mrežno; stranica posjećena: lipanj 2024.].

Sažetak

Klasifikacija histoloških snimaka crijeva konvolucijskom neuronskom mrežom

Rujana Perić

Tema ovog rada je razvoj metoda za klasifikaciju Chronove bolesti i Ulceroznog kolitisa pomoću dubokih neuronskih mreža. Rad objašnjava postupak preobrade malog skupa podataka nepogodnog za klasifikaciju što uključuje pronalaženje regija interesa, ekstrakciju malih sličica regija interesa, uniformiranje količine podloge u odnosu na tkivo, povećavanje dataseta, te podjelu na skup za treniranje i skup za testiranje. Opisuje se arhitektura jednostavne konvolucijske mreže s dva konvolucijska, dva objedinjujuća i jednim potpuno povezanim slojem, proces treniranja i usporedbe performansi jednostavne konvolucijske mreže s ResNet i EfficientNet arhitekturama.

Ključne riječi: umjetna inteligencija; strojno učenje; duboko učenje; računalni vid; umjetne neuronske mreže; duboke neuronske mreže; konvolucijske neuronske mreže; klasifikacija; obrada slika; nadzirano učenje; histološke snimke; upalne bolesti crijeva; Crohnova bolest; Ulcerozni kolitis; tehnologija u medicini; računalno potpomognuta dijagnostika;

Abstract

Classification of histological images of the intestine using convolutional neural network

Rujana Perić

The topic of this thesis is the development of methods for the classification of Crohn's disease and Ulcerative Colitis using deep neural networks. The thesis describes processing a small data set unsuitable for classification, which includes finding regions of interest, extracting small thumbnails of regions of interest, uniforming the amount of substrate in relation to tissue, increasing the dataset, and dividing it into a training set and a test set. The architecture of a simple convolutional network with two convolutional, two unifying and one fully connected layers is described, as well as the process of training and performance comparison of a simple convolutional network with ResNet and EfficientNet architectures.

Keywords: artificial intelligence; machine learning; deep learning; computer vision; artificial neural networks; deep neural networks; convolutional neural networks; classification; image processing; supervised learning; histological recordings; inflammatory bowel diseases; Crohn's disease; Ulcerative colitis; technology in medicine; computer assisted diagnostic;