

UNIVERZA V LJUBLJANI  
FAKULTETA ZA MATEMATIKO IN FIZIKO  
ODDELEK ZA FIZIKO

PRAKTIKUM STROJNEGA UČENJA V FIZIKI

**5. naloga: Binarna klasifikacija resnosti COVID-19 okužbe s pomočjo  
CT slik prsnega koša**

Žiga Šinigoj, 28222025

Ljubljana, december 2023

# 1 Uvod

Pri okužbi s COVID-19 virusom so eden od bolj prizadetih organov pljuča. V njih se ob okužbi začne tvoriti mlečno steklo. Resnost okužbe je povezana s količino mlečnega stekla v pljučih. Pri hudem poteku se pojavijo vzorci ”norega” tlakovanja in ”haložnaka. V domači nalogi bomo s pomočjo konvolucijskih nevronskih mrež poskušali napovedati resnost okužbe na podlagi CT slik pljuč bolnika. Naloga je sestavljena iz dveh delov, v prvem delu je cilj naloge čim boljše napovedati resnost COVID-19 okužbe z nevronskimi mrežami. V drugem delu naloge pa interpretirati dekompozicijo slike pri potovanju skozi nevronsko mrežo.

Na voljo imamo podatkovni set 226 blago okuženih pljuč in 226 resno okuženih pljuč. Vsaka CT slika je vnaprej normalizirana in segmentirana. Slike so velikosti  $512 \times 512$  pikslov. Za vsakega od pacientov imamo 10 centralnih rezin pljuč v aksialni smeri. V učni množici uporabimo eno centralno rezino na pacienta, v testni in validacijski množici pa 10 rezin na pacienta ter vedno na koncu povprečimo rezultate po 10 rezinah.

## 2 Učenje in testiranje nevronskih mrež

Pri učenju modelov sem poskušal optimizirati hiperparametre: stopnja učenja (lr), weight decay (wd) in multiplikator (mp). Modele sem učil na 20 epohah, saj je v večini primerov prišlo do konvergencije. Ko sem našel ”najboljši” model sem ga učil na 100 ali 50 epohah. Pri optimizaciji hiperparametrov sem izbral nekaj vrednosti hiperparametra in gledal vrednosti funkcije izgube in AUC modela ter tako določil hiperparameter. Najprej sem optimiziral hitrost učenja in z optimalno vrednostjo hitrosti učenja naprej optimiziral weight decay. Po enakem postopku sem optimiziral še multiplikator. V splošnem bi bilo potrebno narediti mrežni sprehod po vseh možnih kombinacijah teh treh parametrov, vendar bi bilo to zelo zamudno.

Pri učenju modelov se mi je večkrat zgodilo, da je bila vrednost AUC modela velika, ampak loss funkcija precej večja kot pri modelih ki so imeli manjšo vrednost AUC. V tem primeru nisem bil najbolj prepričan kateri parameter je bolj pomemben. Mislim, da je pri binarni klasifikaciji bolj pomemben AUC in ne toliko loss funkcija. Vseeno sem izbiral optimalne parametre upoštevajoč vrednost loss funkcije.

### 2.1 DNN

Najprej sem sestavil dve DNN mreži, in sicer DNN1, ki ima arhitekturo

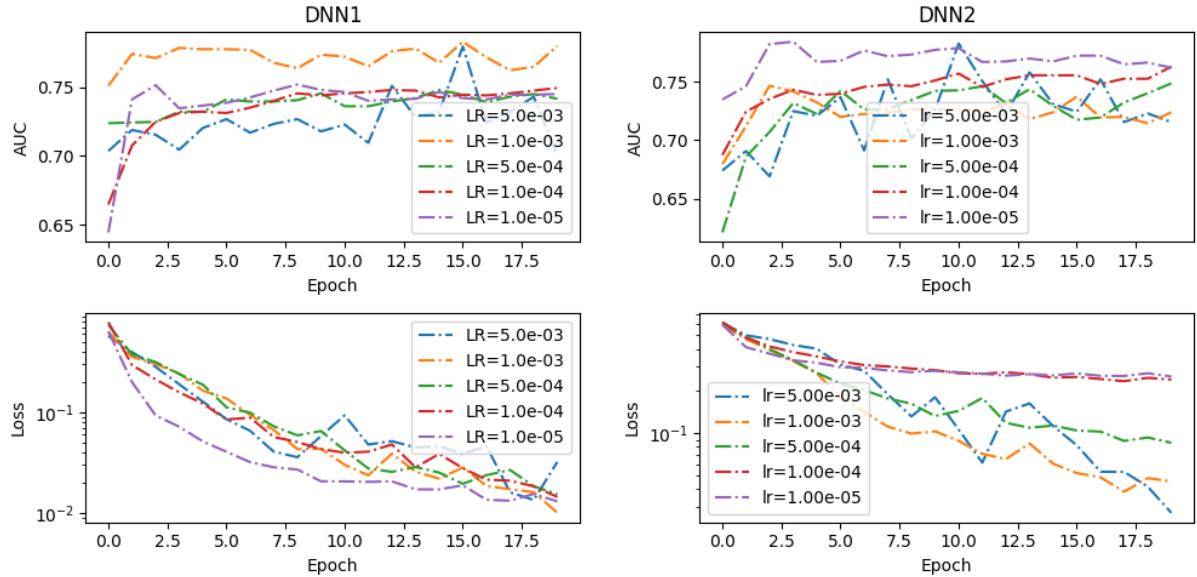
- $\text{Linear}(512^2 \rightarrow 512)$
- $\text{BatchNorm1D}()$
- $\text{ReLU}()$
- $\text{Linear}(512 \rightarrow 1)$

in mrežo DNN2, ki ima arhitekturo

- $\text{Linear}(512^2 \rightarrow 512)$
- $\text{BatchNorm1D}()$
- $\text{ReLU}()$
- $\text{Linear}(512 \rightarrow 256)$
- $\text{BatchNorm1D}()$
- $\text{ReLU}()$
- $\text{Linear}(256 \rightarrow 128)$
- $\text{BatchNorm1D}()$
- $\text{ReLU}()$

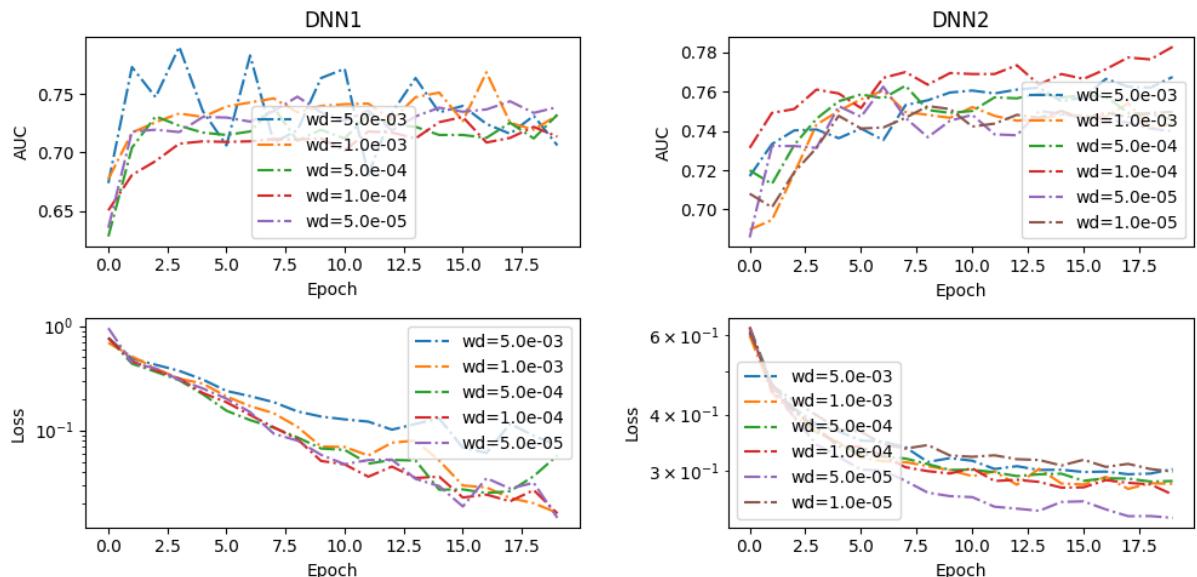
- Linear( $128 \rightarrow 32$ )
- BatchNorm1D()
- ReLu()
- Linear( $32 \rightarrow 1$ )

Učenje modelov pri različnem parametru hitrosti učenja prikazuje slika 1. Za model DNN1 je optimalna hitrost  $lr = 10^{-3}$ , za model DNN2 pa sem izbral  $lr = 10^{-5}$ . Mogoče bi bila dobra tudi vrednost  $10^{-4}$ , saj se zdi da izbrana hitrost učenja pada z epohami.

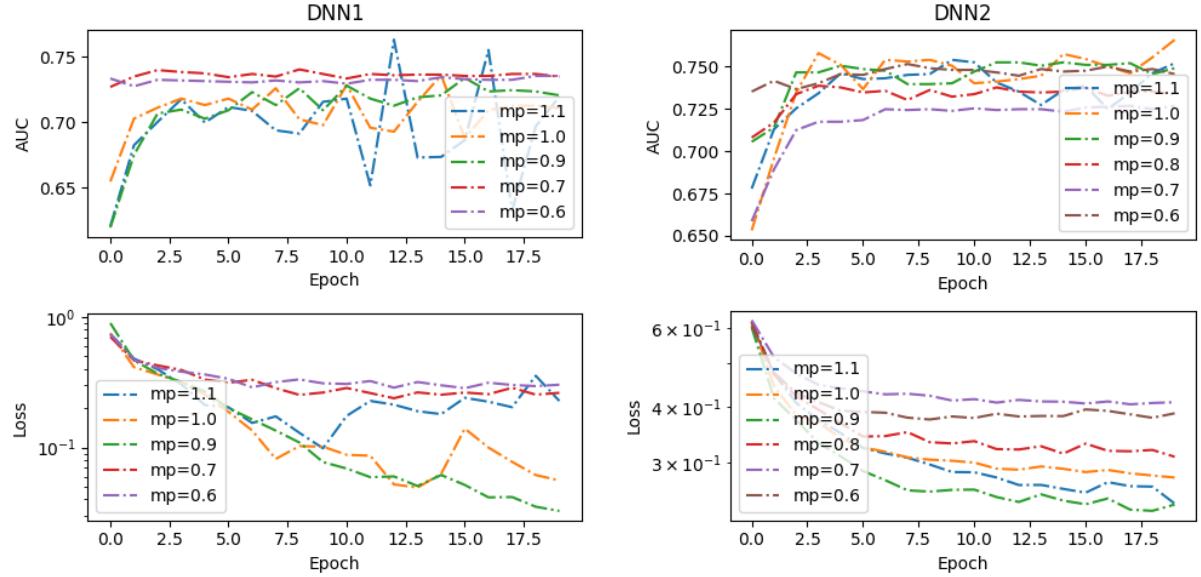


Slika 1: Vrednost AUC in Loss funkcije pri različnih hitrostih učenja.

Z izbrano hitrostjo učenja optimiziramo weight decay (slika 2). Za DNN1 sem izbral vrednost  $wd = 10^{-3}$ , za DNN2 pa  $wd = 10^{-4}$ . Pri optimizaciji multiplikativnega faktorja sem mogoče vzel preveč grobo skalo (slika 3). Za optimalne vrednosti sem izbral DNN1:  $mp = 0.9$  in DNN2:  $mp = 0.9$ .



Slika 2: Vrednost AUC in Loss funkcije pri različnih vrednostih parametra weight decay.



Slika 3: Vrednost AUC in Loss funkcije pri različnih vrednostih multiplikatorja.

## 2.2 CNN

Bolj primerne za obdelavo slik od DNN so konvolucijske nevronske mreže (CNN). Sestavil sem dve arhitekturi, CNN1:

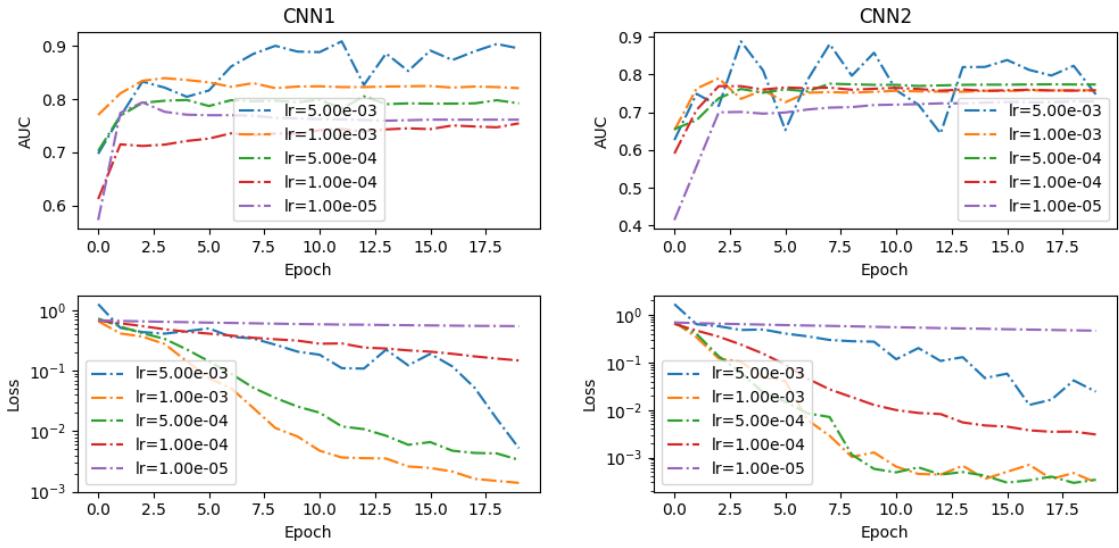
- Conv2d(1,4,kernel\_size = (7, 7), stride = (2, 2), padding = (3, 3))
- BatchNorm2D()
- ReLu()
- Max\_pool2d((2 ,2))
- Conv2d(4,8,kernel\_size = (7, 7), stride = (2, 2), padding = (3, 3))
- BatchNorm2D()
- ReLu()
- Max\_pool2d((2 ,2))
- Conv2d(8,16,kernel\_size = (5, 5), stride = (2, 2), padding = (2, 2))
- BatchNorm2D()
- ReLu()
- Max\_pool2d((2 ,2))
- Conv2d(16,64,kernel\_size = (3, 3), stride = (1, 1), padding = (1, 1))
- BatchNorm2D()
- ReLu()
- Max\_pool2d((2 ,2))
- Linear(1024 → 512)
- ReLu()
- Linear(512 → 1)

Namensko sem zmanjševal velikost jedra, saj večje jedro najprej zajame splošne lastnosti slike, medtem ko manjša jedra omogočajo zaznavo detajlov na sliki. Druga sestavljena arhitektura je malo spremenjena CNN1 in sicer CNN2:

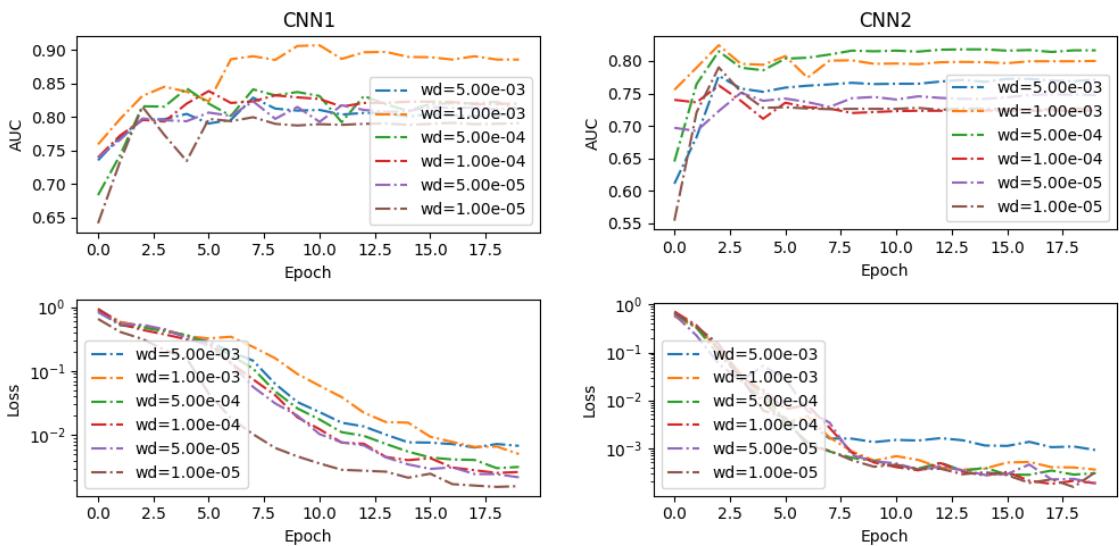
- Conv2d(1,4,kernel\_size = (7, 7), stride = (2, 2), padding = (3, 3))
- BatchNorm2D()
- ReLu()
- Max\_pool2d((2 ,2))
- Conv2d(4,8,kernel\_size = (7, 7), stride = (2, 2), padding = (3, 3))
- BatchNorm2D()
- ReLu()
- Max\_pool2d((2 ,2))
- Conv2d(8,16,kernel\_size = (5, 5), stride = (2, 2), padding = (2, 2))
- BatchNorm2D()
- ReLu()
- Conv2d(16,32,kernel\_size = (5, 5), stride = (2, 2), padding = (2, 2))
- BatchNorm2D()
- ReLu()
- Conv2d(32,64,kernel\_size = (3, 3), stride = (1, 1), padding = (1, 1))
- BatchNorm2D()
- ReLu()
- Max\_pool2d((2 ,2))
- Conv2d(64,128,kernel\_size = (3, 3), stride = (1, 1), padding = (1, 1))
- BatchNorm2D()
- ReLu()
- Conv2d(128,256,kernel\_size = (3, 3), stride = (1, 1), padding = (1, 1))
- BatchNorm2D()
- ReLu()
- Linear(2048 → 512)
- ReLu()
- Linear(512 → 1)

Optimalna hitrost učenja (slika 4) je za CNN1:  $lr = 1e^{-3}$  in za CNN2:  $lr = 0.5e^{-3}$ . Weight decay je optimalen pri vrednosti  $wd = 10^{-3}$  za CNN1 model in  $wd = 0.5 \cdot 10^{-3}$  (slika 5). Za optimalne vrednosti množnikov (slika 6) sem vzel CNN1:  $mp = 0.9$  in CNN2:  $mp = 0.9$ .

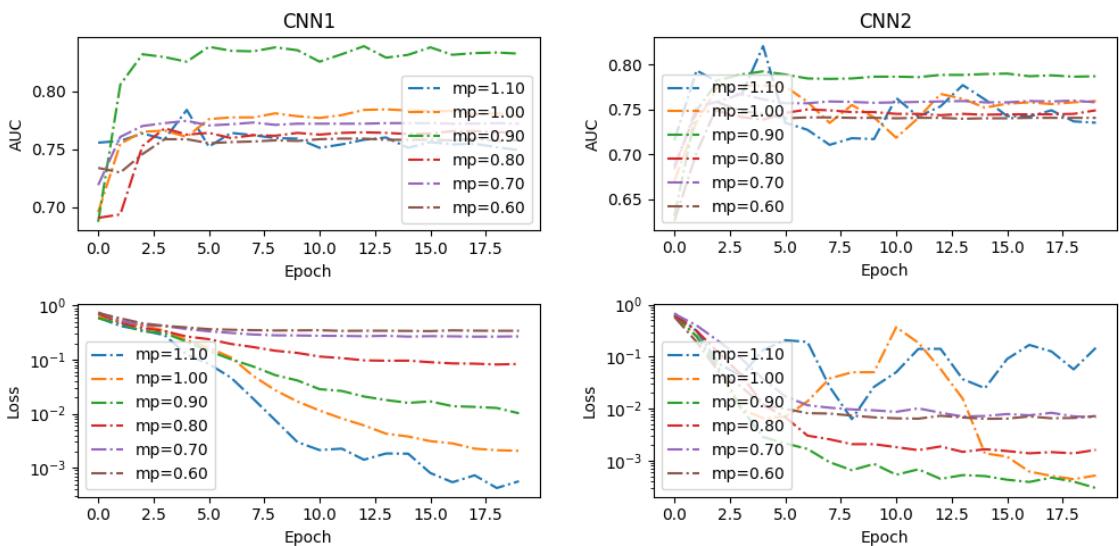
Če sedaj primerjam modele med sabo (slika 7) opazim, da se sta CNN modela boljša od DNN kar je pričakovano. Ker sem ob optimalnih parametrih modele učil 100 epoh bi mogoče pričakovali boljši ali slabši rezultat vendar se zdi, da model hitro konvergira ( $\approx 20$  epoh) in se potem bistveno ne spreminja več. Za nadaljnjo obravnavo bom izbral modela CNN1 in DNN2, kot najboljša iz svoje skupine.



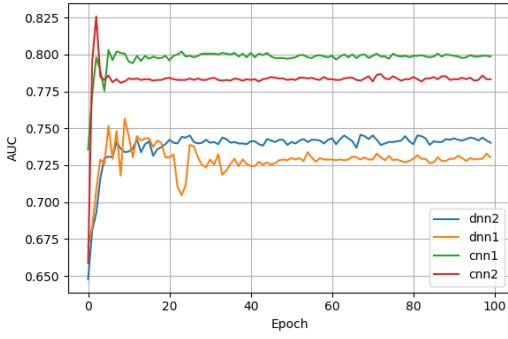
Slika 4: Vrednost AUC in Loss funkcije pri različnih hitrostih učenja.



Slika 5: Vrednost AUC in Loss funkcije pri različnih vrednostih parametra weight decay.



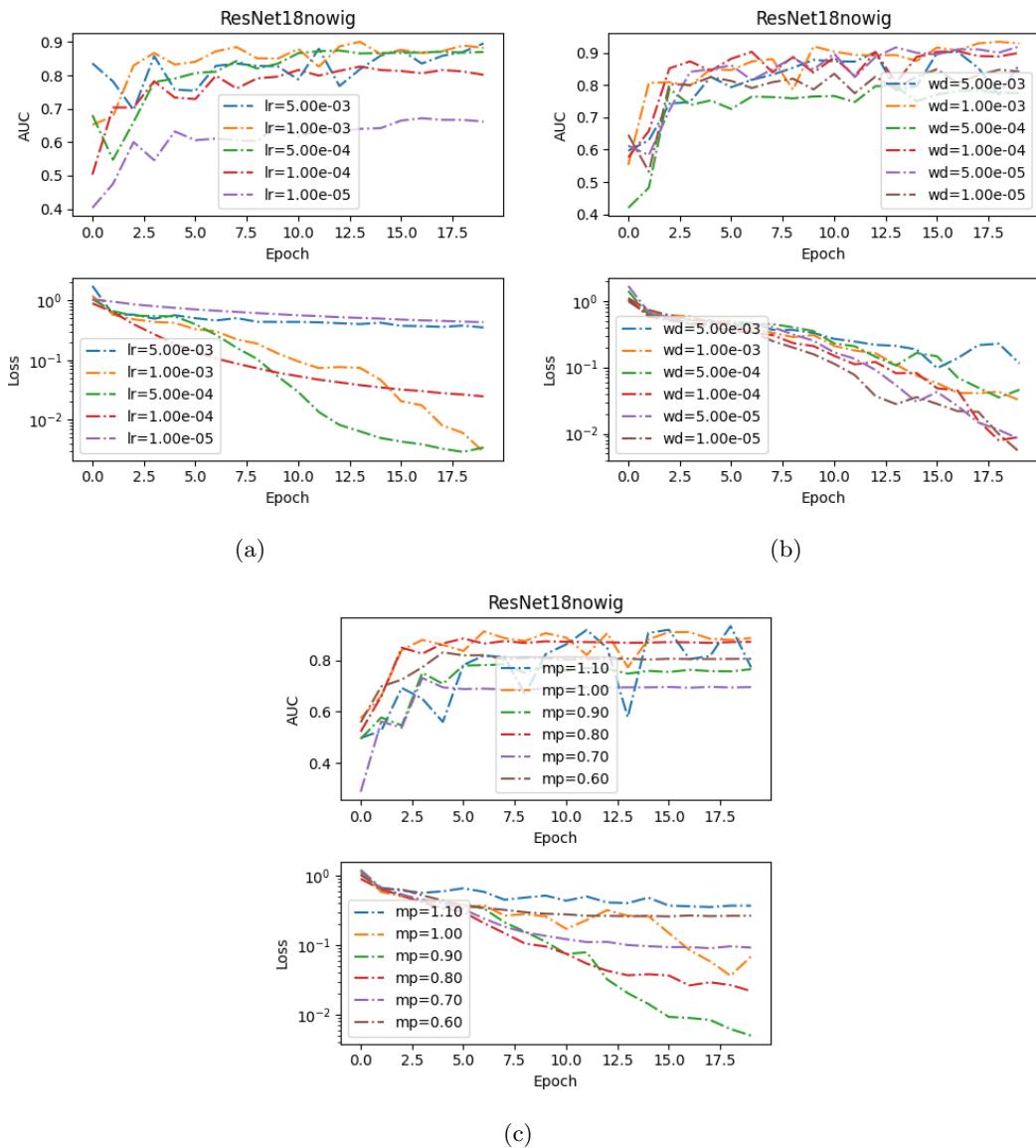
Slika 6: Vrednost AUC in Loss funkcije pri različnih vrednostih multiplikatorja.



Slika 7: AUC za različne ”optimalne” modele.

### 2.3 Netreniran ResNet18

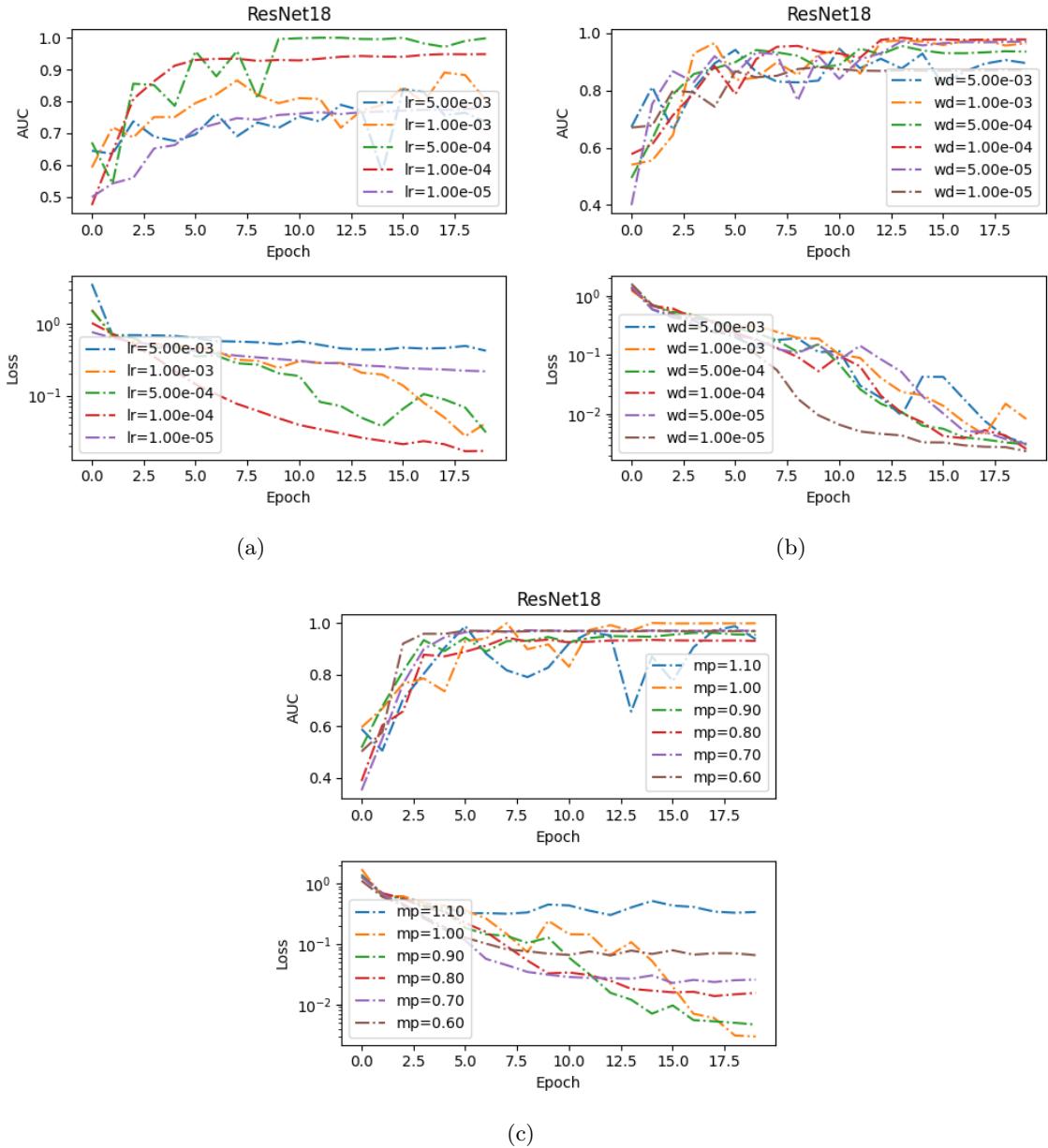
Naslednji obravnavani model je ResNet18 brez predhodno naučenih filtrov. Uporabil sem samo 1 vhoden kanal. Podobno kot za prejšnje modele določimo optimalne parametre (slika 8), ki so  $lr = 10^{-3}$ ,  $wd = 10^{-3}$ ,  $mp = 0.8$ .



Slika 8: a) Vrednost AUC in Loss funkcije pri različnih hitrostih učenja. b) Vrednost AUC in Loss funkcije pri različnih vrednostih parametra weight decay. c) Vrednost AUC in Loss funkcije pri različnih vrednostih parametra weight decay.

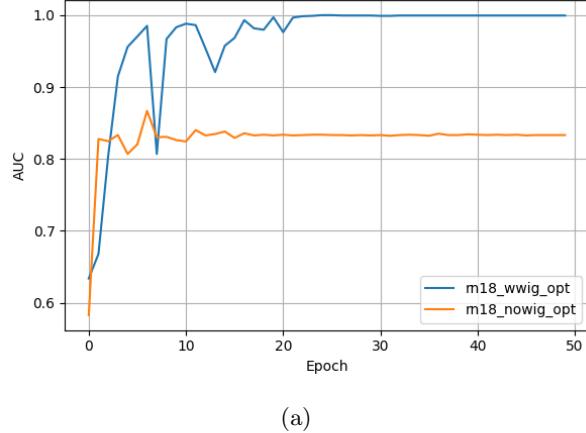
## 2.4 ResNet18

Pri modelu ResNet18 z naučenimi uteži spremenimo samo prvo in zadnjo plast, ker so uteži že naučene sem uporabil kar vse tri vhodne plasti in na vse 3 vhode peljal enako sliko. Tukaj sem učil uteži za prvo plasti, ki je imela 3 vhode (RGB). Kasneje sem se spomnil (pri vizualizaciji), da to ni potrebno saj je prva plast filtrov precej generična. Model, kateremu sem določal uteži na prvih 3 plasteh, imenujem ResNet18, model, kateri ima začetne uteži že naučene sem imenoval ResNet18og. Predpostavil sem tudi, da ima ResNet18og enake optimalne parametre kot ResNet18 (saj nisem imel namena ponavljati iskanja parametrov, ampak sem ga samo vključil v pomembne rezultate). Optimalni parametri so:  $lr = 50^{-4}$ ,  $wd = 10^{-4}$ ,  $mp = 1.0$  (slika 9).



Slika 9: Vrednost AUC in Loss funkcije pri različnih hitrostih učenja. b) Vrednost AUC in Loss funkcije pri različnih vrednostih parametra weight decay. c) Vrednost AUC in Loss funkcije pri različnih vrednostih parametra weight decay.

Primerjavo obeh ResNet18 modelov prikazuje slika 10. Bistevno boljši je model, ki ima prednaučene uteži in 3 uporabljene plasti.



(a)

Slika 10: Validacijska funkcija izgube pri različnem številu plasti-*layer\_num* (brez upoštevanja zadnje plasti).  
a) LSTM many-to-one model. b) LSTM many-to-many model.

## 2.5 Primerjava najboljših modelov

Sedaj ko imamo optimalne parametre najboljših modelov lahko določimo prag. Pri šimetričnih "odločitvah" bi bil prag nevronske mreže 0.5. Ker pa je pri bolezni pomembno, da naš model zazna vse resno okužene, tudi če ob tem zazna kakšnega, ki dejansko ni resno okužen je to boljše kot obraten scenarij. Posledično bo prag večji oz. manjši od 0.5. Za določitev praga sem izbral metriki F1 in G-mean. F1 metrika maksimizira število pravilnih določitev hude okužbe in je definirana kot

$$F1 = \frac{2precision \times recall}{precision + recall} \quad (1)$$

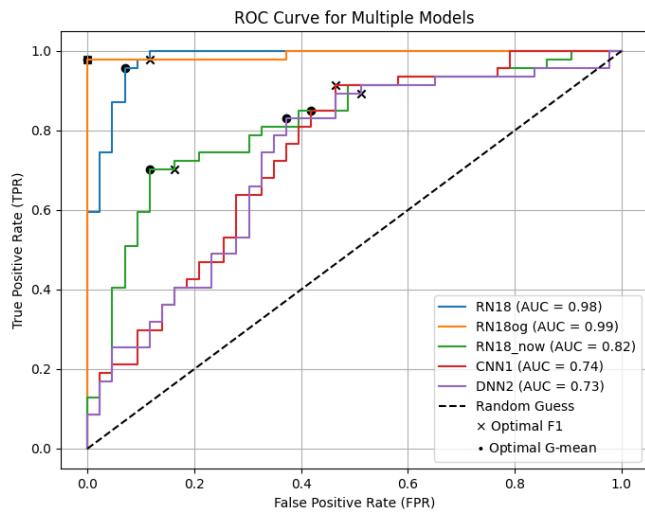
, kjer je  $precision = TP / (TP + FP)$  in  $recall = TP / (TP + FN)$ . Metrika G-mean je definirana kot

$$G - mean = \sqrt{TPR(1 - FPR)} \quad (2)$$

in maksimizira TP in TN. Uspešnosti modelov prikazuje slika 11. Najuspešnejši je seveda model ResNet18 z prednaučenimi utežmi (tudi v prvi plasti), kjer na vhode RGB dobi 3 enake slike. Presenetljivo je precej slabši ResNet18 z nenaučenimi utežmi. Izračunani pragi za modele so prikazani v tabeli 1. Uspešnost modela lahko vidimo tudi iz slike 12, kjer je najmanj uspešen CNN1 model (glede na število FP). Pričakovano je najboljši model ResNet18og. V našem primeru je boljša metrika F1, saj ResNet18 (RN18) doseže boljše rezultate pri threshold vrednosti F1 kot pri G-mean. Relevantne meritve uspešnosti modelov prikazuje slika 14.

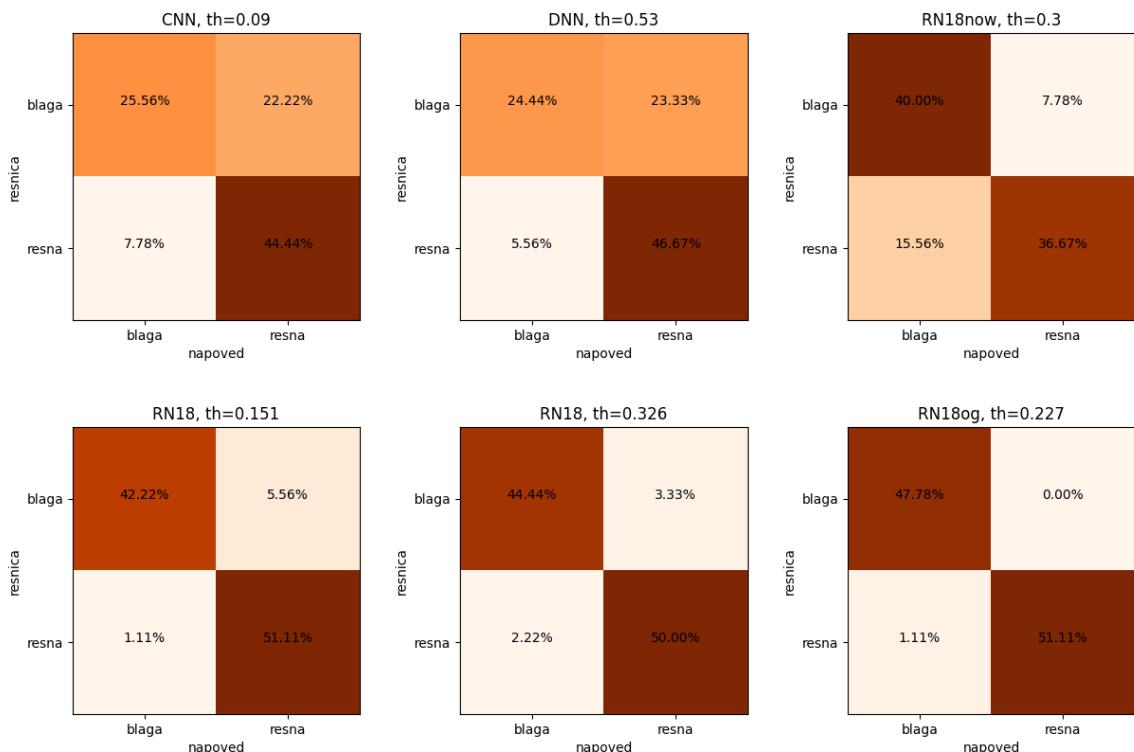
	ResNet18og	ResNet18	ResNet18_nowig	CNN1	DNN1
F1	0.227 1,1	0.351	0.306	0.0827	0.526
G-mean	0.227	0.333	0.306	0.0986	0.557

Tabela 1: Vrednosti metrik F1 in G-mean za različne modele.

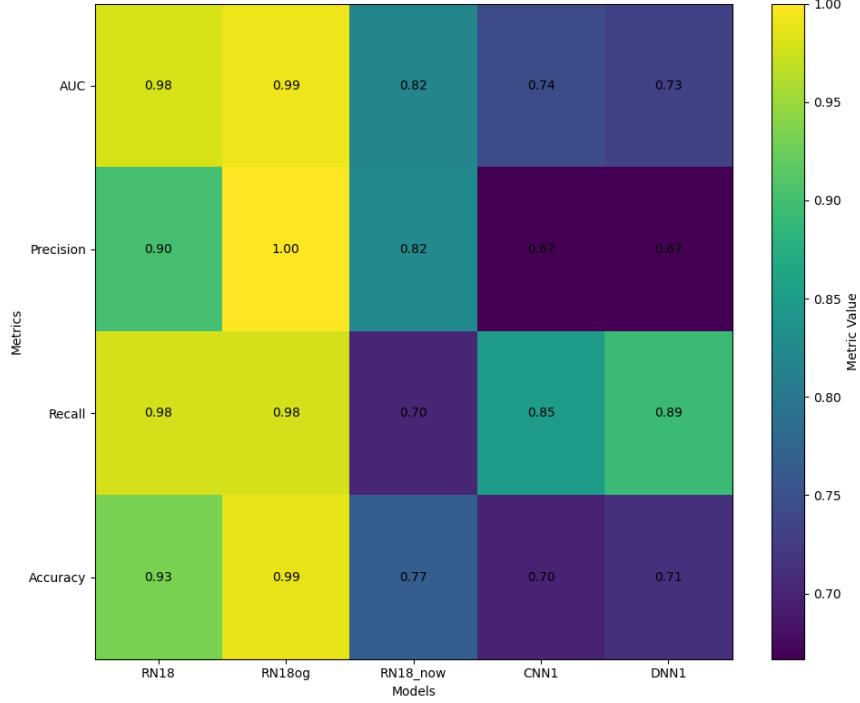


(a)

Slika 11: ROC krivulja za najboljše modele. Križci označujejo prag določen z metriko F1 in krožci označujejo prag določen po metriki G-mean.



Slika 12: Konfuzijske matrike za različne modele. Oznaka  $th$  označuje threshold vrednost.

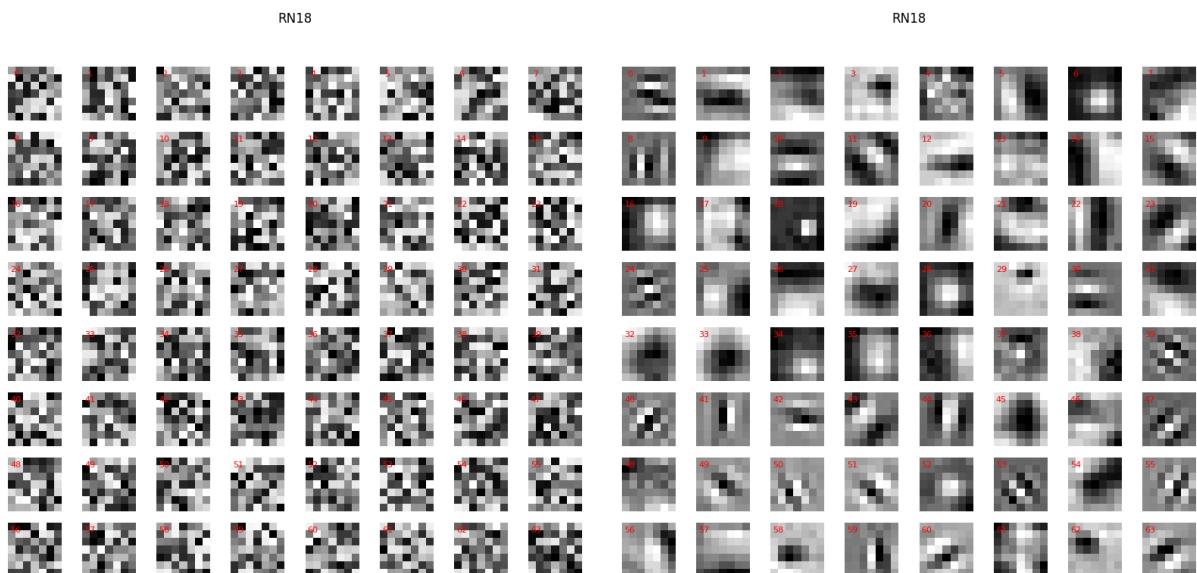


(a)

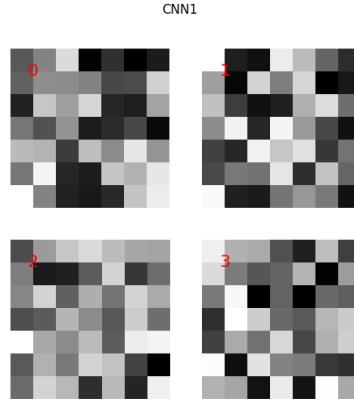
Slika 13: Relevantne meritve uspešnosti modelov.

### 3 Vizualizacija delovanja CNN

V drugem delu naloge nas predvsem zanima kaj se dogaja s sliko ko potuje skoti nevronsko mrežo. Najprej si poglejmo filtre na prvi konvolucijski plasti ResNet18 modela (slika 14). V splošnem pričakujemo, da bodo ti filtri po dovolj temeljitem učenju ustvarili jedra - Fourierove baze, wavelet transformacije, gausso-vska jedra ipd. V primeru nenaучenega modela so filtri razen izjem precej neintuitivni in se precej razlikujejo od filtrov prednaučenega modela (slika 14b). V primeru prednaučenega modela pa je mogoče prepoznati wavelet transformacije npr. 51 filter, fourierove transformacije npr. 57, in gaussovskia jedra npr. 27. Prav tako so neintuitivni filtri začetne plasti CNN1 modela (slika 15).

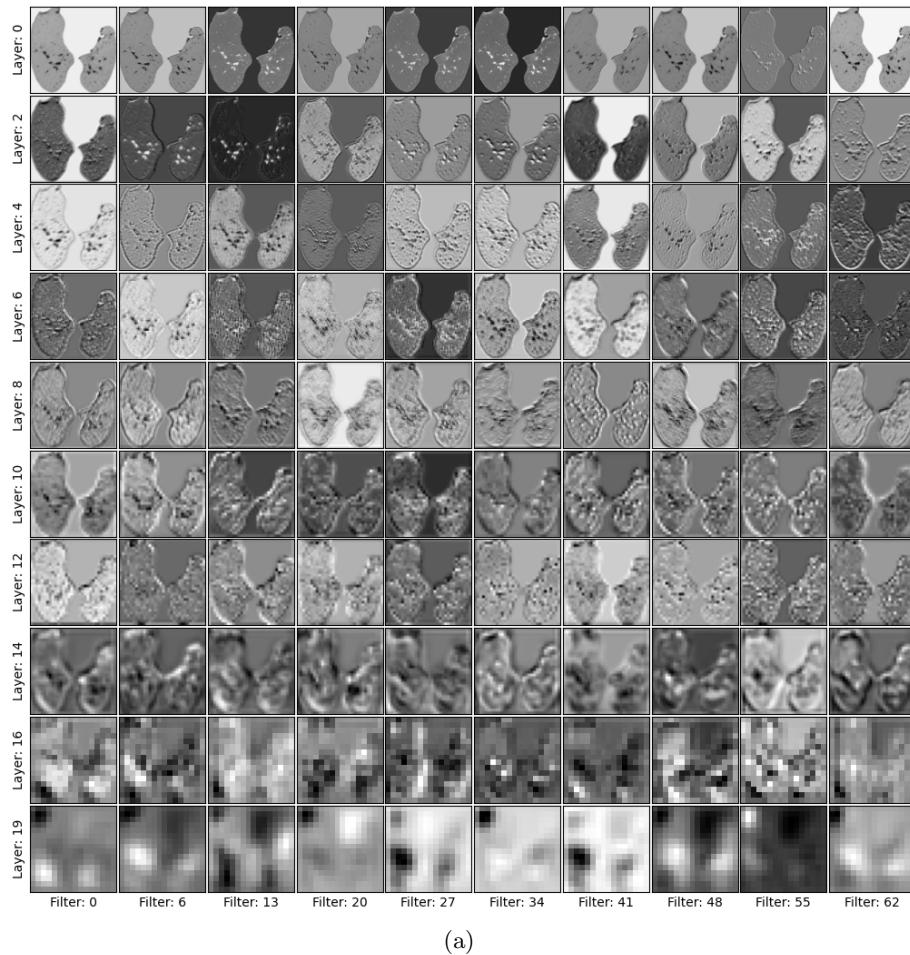


Slika 14: a) Filtri ResNet18 brez prednaučenih uteži v prvi konvolucijski plasti in prvem kanalu. b) Filtri ResNet18og s prednaučenimi utežmi v prvi konvolucijski plasti in prvem kanalu.



Slika 15: a) Filtri CNN1 v prvi konvolucijski plasti.

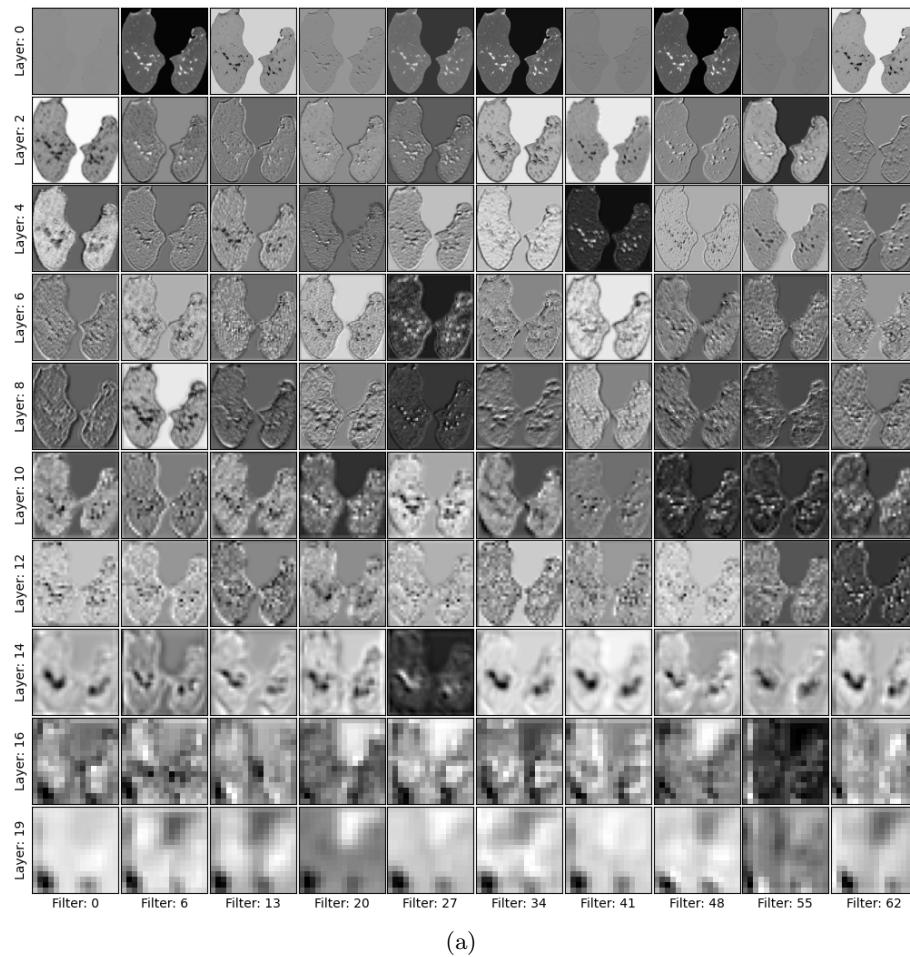
Naslednja stvar, ki je zanimiva, je transformacija slike skozi plast. V ta namen sem prikazal sliko v dani plasti in danem jedru za ResNet18 model in CNN1 model. V začetnih plasteh lahko še razumemo transformacije, model podari robeve in strukturo. V globjih plasteh pa se manjša ostrina slike in ostajajo samo podatki oz. pikli, ki so modelu pomembni. Pri ResNet18 modelu (slika 16, 17) dobimo v predzadnji plasti sliko, ki ima izrazito piko na nekem delu. Model je tam zaznal okužbo in naredil tisti del slike pomemben za klasifikacijo. Pri CNN1 modelu (slika 18) pa so globji filtri precej bolj "razpršeni" in vsebujejo več območij, ki so za model pomembna, ampak niso tako centrirana. Pogledamo si lahko še vhodno in izhodno sliko za dane obravnavane modele (slika 19 in 20). V primeru resnih okužb je precej jasno kaj CNN1 in ResNet18 pogleda.



(a)

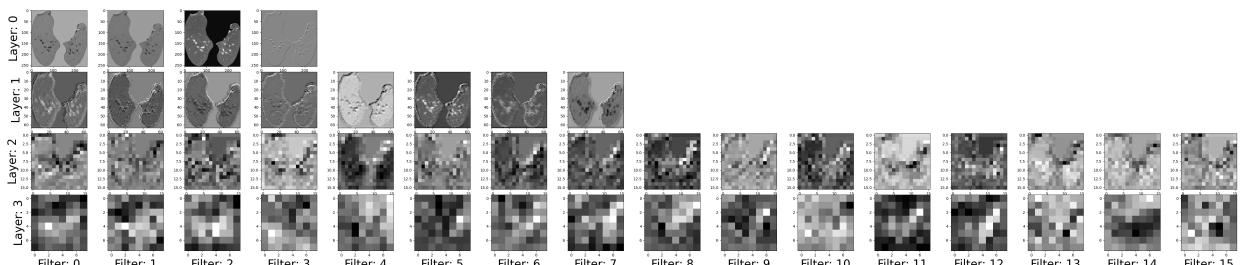
Slika 16: Vhodna slika na različnih delih nevronske mreže. V tem primeru je model ResNet18 s prednaučenimi filtri, a brez prednaučene začetnih in končne plasti uspešno prepozna blago okužbo.

V primeru blagih okužb se zelo dobro odreže ResNet18og, ki dobro vidi blage okužbe. V primeru CNN1 in blage okužbe pa končna slika ne nujno opravičuje odločitve (končna slika nima ozkega območja, kjer bi bila vidna okužba).



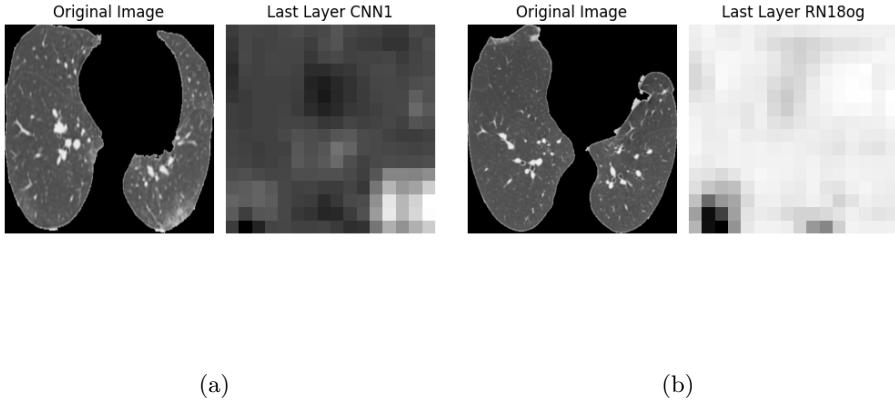
(a)

Slika 17: Vhodna slika na različnih delih nevronske mreže. V tem primeru je model ResNet18 s prednaučenimi filtri (ResNet18og) uspešno prepozna blago okužbo.

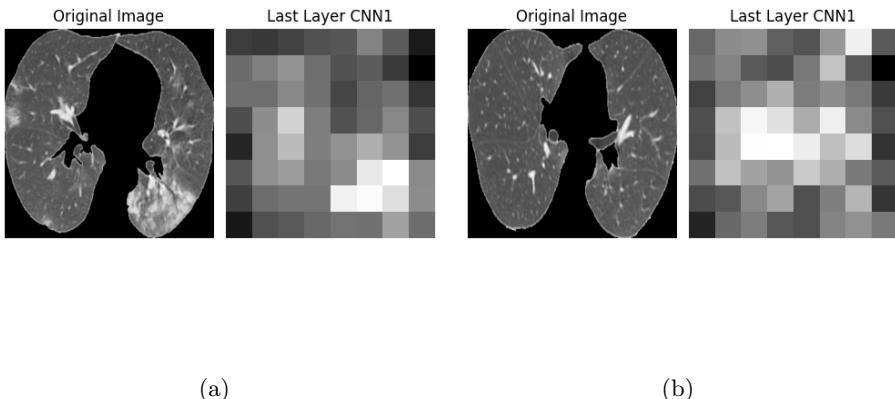


(a)

Slika 18: Vhodna slika na različnih delih nevronske mreže. V tem primeru je model uspešno prepozna blago okužbo.



Slika 19: ResNet18og model. a) Uspešno prepoznanje resna okužba (napaka v opisu na sliki). b) Uspešno prepoznanje blaga okužba.

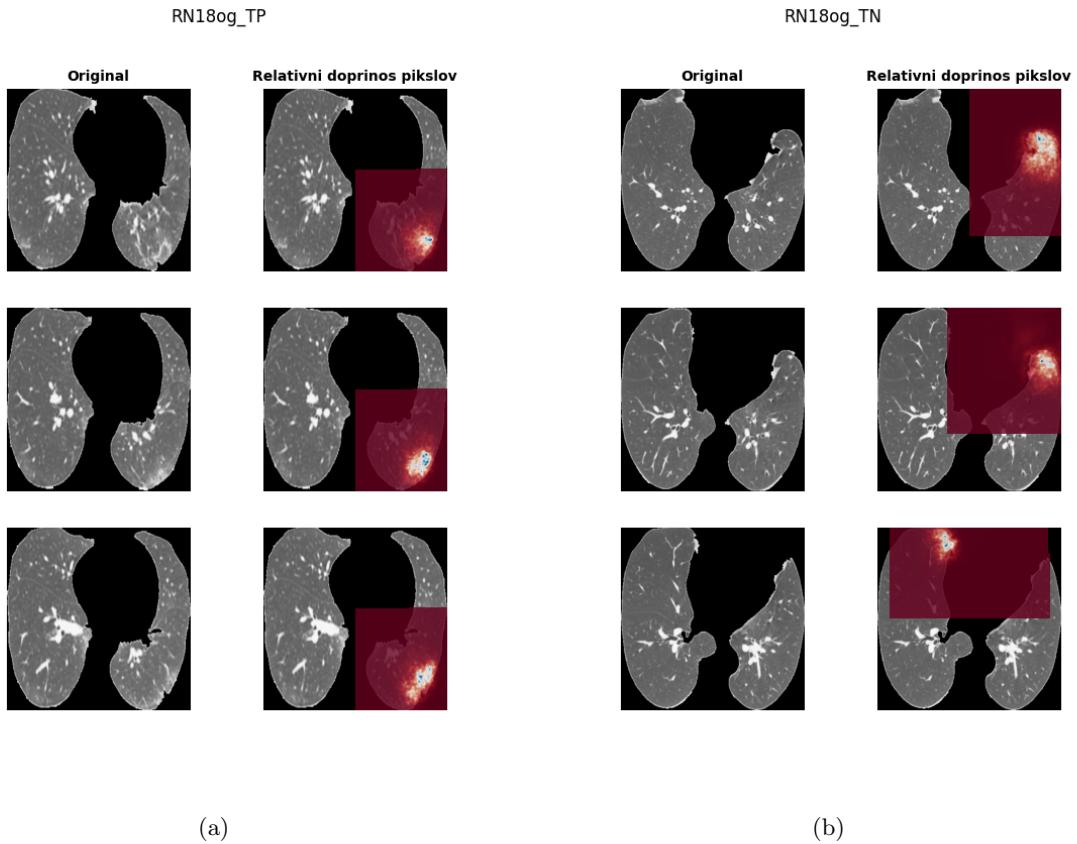


Slika 20: CNN1 model. a) Uspešno prepoznanje resna okužba. b) Uspešno prepoznanje blaga okužba.

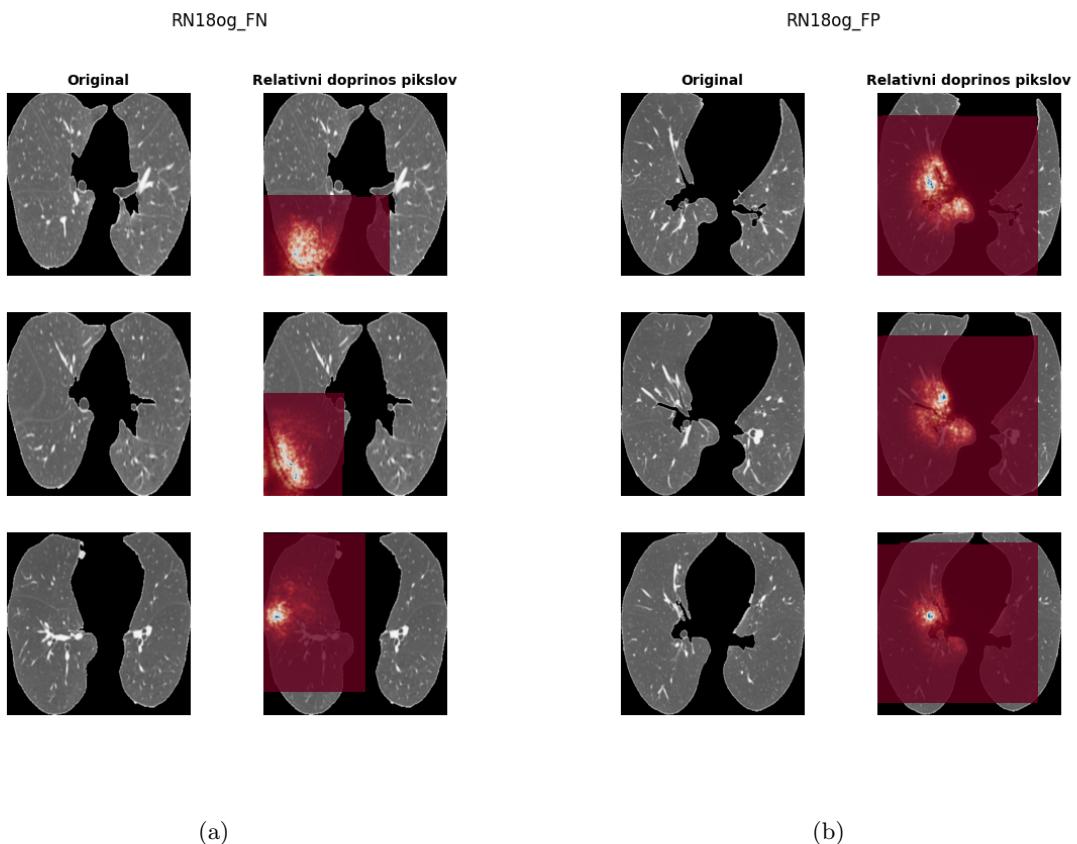
Za vizualizacijo operacij nevronske mreže imamo še metodo map izpostavljenosti, ki nam povedo doprinos posameznih piksov k odločitvi nevronske mreže. Pogledali si bomo mape izpostavljenosti za vse 4 primere (TP, TN, FP, FN). Mape za model ResNet18og prikazujejo slike 21, 22. V primeru TP model dobro prepozna resno okužbo in njegova odločitev je odvisna samo od območja, kjer je okužba. V primeru TN prav tako dobro poišče okuženo območje, se pa njegovo vidno območje poveča. V najbolj FN primeru model podcení resnost okužbe (v tem primeru jaz ne vidim, da bi šlo za močno okužbo). V primeru FP pa model preceni resnost okužbe, prav tako se vidno polje, ki odloča o resnosti okužbe, poveča. V primeru modela CNN1 pa nevronska mreža vedno obravnava celotno sliko in ne samo fokusiran del, kar je lahko v v primeru blagih okužb dobro, ampak v tem primeru je manj jasno po kakšnih kriterijih se je nevronska mreža odločala. V primeru TP nevronska mreža dobro prepozna mlečno steklo (slika 23). Ostale primere prikazujejo slike 23, 24.

## 4 Zaključek

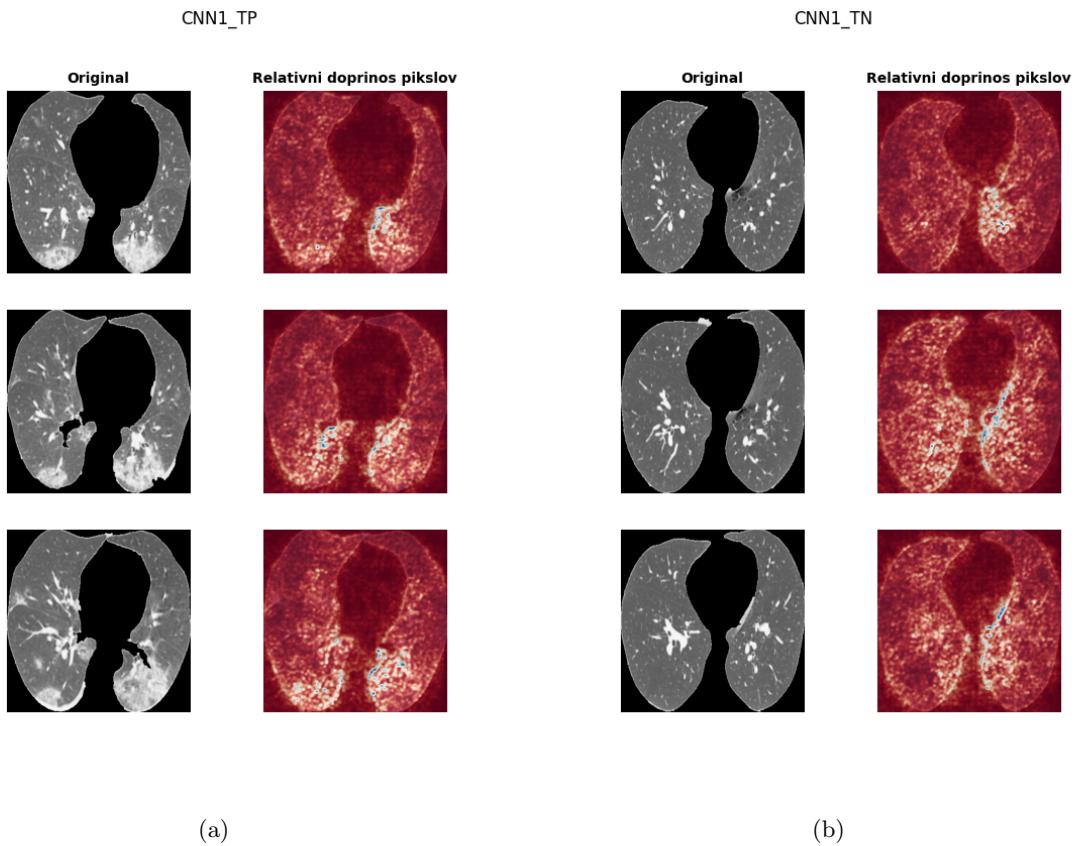
Naloga se je osredotočila na binarno klasifikacijo resnosti okužb COVID-19 na podlagi CT slik prsnega koša s pomočjo konvolucijskih nevronske mrež. Preizkušene so bile različne arhitekture nevronske mrež, vključno z modeli DNN in CNN, ki so bili optimizirani z uporabo parametrov, kot so stopnja učenja, zmanjšanje uteži in multiplikativni faktor. Rezultati so pokazali, da so modeli CNN v splošnem presegli modele DNN pri klasifikaciji resnosti okužb.



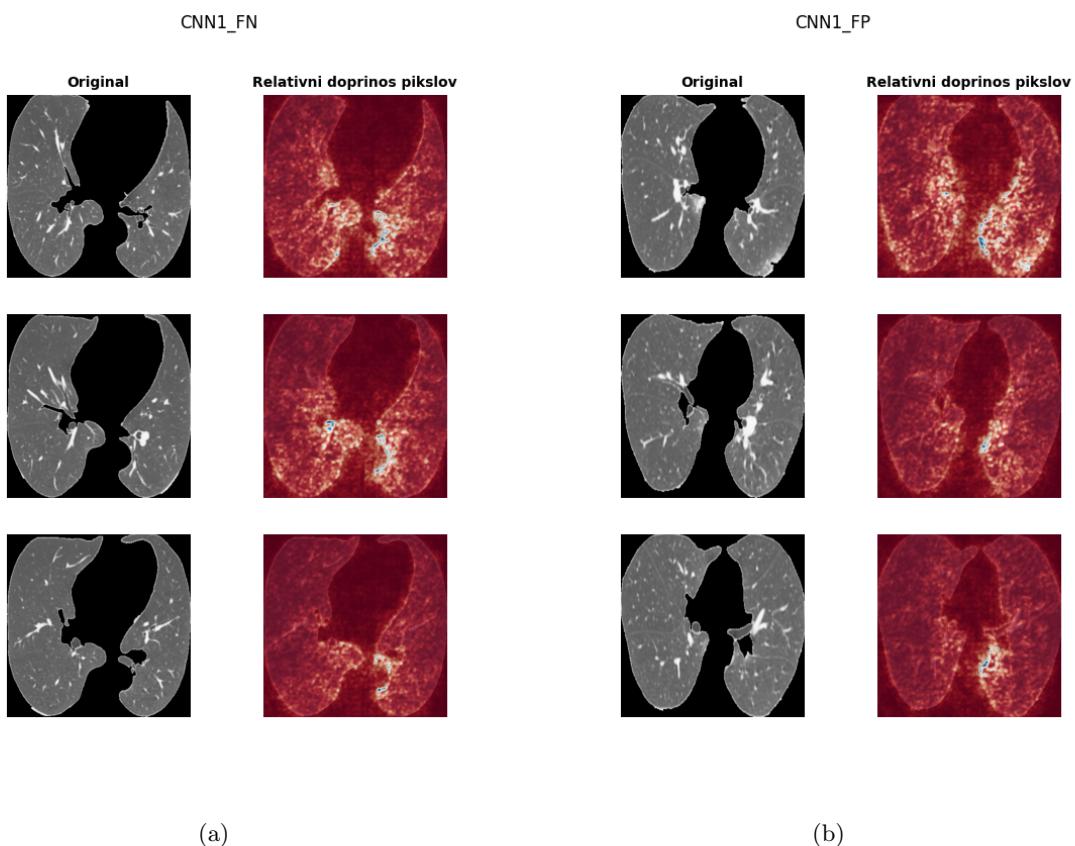
Slika 21: ResNet18og model a) Uspešno prepozna resna okužba b) Uspešno prepozna blaga okužba.



Slika 22: ResNet18og model a) Neuspešno prepozna resna okužba. b) Neuspešno prepozna blaga okužba



Slika 23: CNN1 model a) Uspešno prepoznana resna okužba b) Uspešno prepoznana blaga okužba.



Slika 24: CNN1 a) Neuspešno prepoznana resna okužba. b) Neuspešno prepoznana blaga okužba