

Binarna klasifikacija resnosti COVID-19 okužbe s pomočjo CT slik prsnega koša

Matic Debeljak

28. avgust 2021

1 Uvod

Pandemija koronavirusne bolezni COVID-19 ni ogrozila le svetovnega zdravja, temveč je močno obremenila celotni zdravstveni sistem . Natančne napovedi resnosti COVID-19 okužbe bi lahko pomagale pri odločitvah o hospitalizaciji in zdravljenju, s čimer bi ublažili obremenitev zdravstvenega sistema. Računalniška tomografija prsnega koša (CT) je neprecenljivo orodje za oceno pljučne prizadetosti. Pri pljučnici, ki nastane zaradi COVID-19 okužbe so najpogostejše kvalitativne CT značilnosti predvsem prisotnost mlečnega stekla (ang. Ground Glass Opacities - GGO) in konsolidacije (zgostitve).

2 Naloga

Na voljo imate učno množico, kjer se nahajajo imena centralnih rezin 60% vseh pacientov, označenih z oznako 0 (blaga okužba) in 1 (resna okužba). S pomočjo opazovanja funkcije izgube med učenjem modela in opazovanjem vrednosti AUC na validacijski množici poskusite najti optimalne hiperparametre, ki vodijo do uteži modela, ki bo na neodvisni testni množici po vašem mnenju dosegel najboljši rezultat. Vse slike imate že procesirane. To naredite za 4 različne modele:

1. Preprost model brez konvolucije (vsako sliko obravnavaj kot vektor dimenzije 512x512)
2. Vaša konvolucijska nevronska mreža.
3. Model z modificirano arhitekturo ResNet18 in ne-naučenimi filtri na bazi ImageNet.

4. Model z modificirano arhitekturo ResNet18 in naučenimi filtri na bazi ImageNet.

Hiperparametri, ki jih pri tem poskusite optimizirati:

- Stopnja učenja - (ang. learning rate) optimizatorja Adam
- Weight decay oz. L2 utež optimizatorja Adam
- Multiplikator za eksponenti urnik spremnjanja uteži

Arhitekturo za modela (1) in (2) napišite sami, modela (3) in (4) pa imate že na voljo. Pri modelih (3) in (4) ne zaklepajte plasti (puštite vse tako kot je) in dovolite, da se vaš model do-uči na vaši učni množici (fine-tuning). Za vsako izmed štirih arhitektur izberite uteži modela, ki se bodo po vašem mnenju najbolje obnesle na testni množici. Ko boste za vsako izmed štirih arhitektur imeli izbrane uteži, na testni množici naredite sledeče:

1. Izračunajte AUC
2. Izberite prag (ang. threshold) in argumentirajte svojo odločitev
3. Izračunajte točnost (ang. accuarcy)
4. Izračunajte vrednost f1
5. Izračunajte konfuzijsko matriko

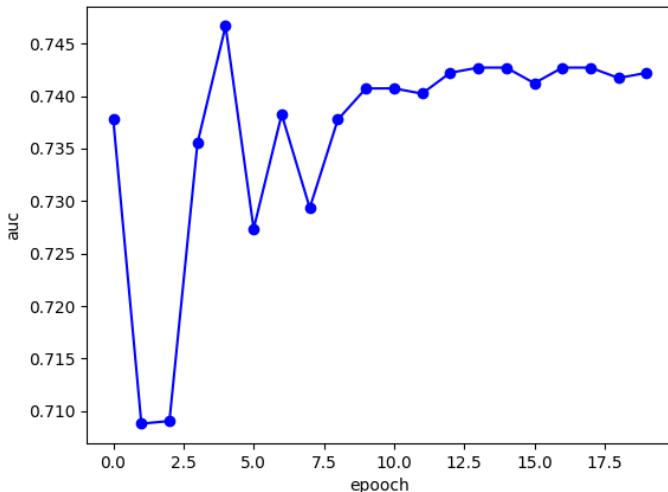
Za vse modele razen prvega, dodaj še:

1. Vizualizirajte in na kratko interpretirajte filtre vhodne konvolucijske plasti.
2. Prikažite, kaj se z vhodno sliko dogaja na različnih globinah mreže.
3. Iz testne množice poiščite nekaj primerov pravilne in napačne klasifikacije ter vizualizirajte relativni doprinos pikslov h končni odločitvi (saliency maps)

3 Rezultati

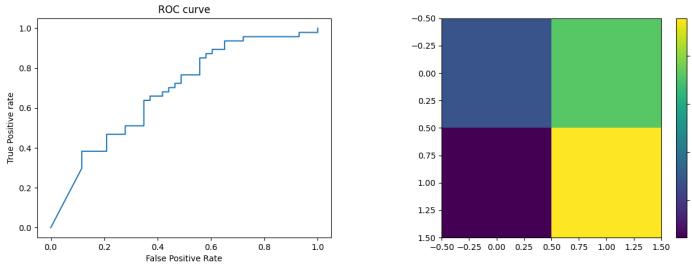
3.1 Vektorski model

Poglejmo si najprej model brez konvolucije, kjer sem sliko "sploščil" v vektor. Vhodna plast je vsebovala 262144 (512×512) nevronov. Sledili sta dve gosto povezani plasti z 10000 in 1000 nevroni, izhodna plast pa je seveda vsebovala en nevron. Aktivacijska funkcija na vseh plasteh je bila relu. Seveda bi si želel uporabiti bolj "kompleksn" mrežo, a žal sem moral ta model poganjati na cpuju (saj je bil za mojo grafično kartico spominksко prezahteven), zato je učenje že tako enostavnega modela precej časovno zahtevno. Zaradi preprostosti mojega modela in dejstva da gosto povezane nevronske mreže niso najbolj primerne za analizo slike (v primerjavi s konvolucijskimi mrežami) od tega modela nisem pričakoval prav blestečih rezultatov. "Optimalne" hiperparametre sem za vse modele določil z "ugibanjem", seveda bi blo bolje uporabiti mrežno iskanje a je bilo to časovno prezahtevno. Za model (1) sem določil: Stopnja učenja = 0.0005, weight decay = 0.00001, multiplicator = 0.97. Na sliki (1) lahko vidimo kako se je tekom učenja modela (1) spremnjala vrednost AUC.



Slika 1: AUC v odvisnosti od epoch za model (1)

Sedaj si poglejmo še kako se je model (1) izkazal na testni množici (AUC = 0.68). To najlažje ocenimo s pomočjo ROC krivulje prikazane na sliki (2) ROC krivulja nam pomaga določiti prag za ločitev med blago in hudo



Slika 2: ROC krivulja (levo) in normalizirana konfuzijska matrika (desno) na testni množici.

okužbo. Želimo čimvečjo uspešnost na pozitivnih pacientih za ceno čimnajše neuspešnosti na negativnih pacientih. Prva ideja je da preposto seštejemo vse narobe napovedane paciente in poskusimo to število minimizirati. Ampak v medicinskih problemih ta rešitev verjetno ni najboljša saj enako kaznujemo da v bolnico sprejmemo zdravo osebo, kot da bolno osebo pošlejmo domov. V resnici pa do večjih zapletov pride če bolno osebo zavrnemo in se tej osebi stanje hitro poslabša. Medtem ko će v bolnico sprejmemo osebo z blago okužbo nas to ne stane prav veliko. A vseno zdravih oseb ne želimo sprejemati v bolnico, saj lahko potencialno zasedejo posteljo za bolno osebo. Zato se splača bolne ljudi, ki so bili označeni za zdrave v naši izbiri bolj "utežiti" kot zdrave ljudi, ki so bili označeni za bolne. Seveda ta utež ne sme biti prevelika, saj bi potem v bolnico sprejeli vse paciente. Zato sem za model (1) izbral threshold = 0.33. Sedaj lahko izračunamo $f_1 = 0.73$ in natačnost = 0.63. Na sliki (2) je prikazana še normalizirana konfuzijska matrika.

3.2 Konvolucijski model

Preden začнем z opisom konvolucijskega modela, bi še povedal, da sem pri modelih (1), (3) in (4) na testni množici za kriterij obolelosti posameznega pacienta vzel največjo napovedano verjetnost (od desetih slik). Pri modelu (2) pa sem vzel poverečeje desetih slik. Bolj intutivna izbira se mi zdi maksimalna izbira, saj je po mojem mnenju bolezen lahko huda tudi če ima pacient "poškodovan" samo en del pljuč, kar se opazi samo na eni od desetih slik, nisem pa zdravnik da bi lahko to trdil. Pri modelu (2) pa sem izbral povrečno oceno saj sem tako dobil bolše rezultate.

Konvolucijski model sem sestavil iz dveh blokov (vsak je vseboval 2d kovolucijo, noramlizacijo, aktivacijsko funkcijo relu in max polling). Sledila je splošcitev v vektor ter ena skrita plast. Podrobnejšo zgradbo vidmio na

sliki (3)

```
class Model_moj(nn.Module):
    def __init__(self):
        super(Model_moj, self).__init__()
        self.conv1 = nn.Conv2d(1, 4, kernel_size=(3,3), stride=(1,1), padding=(1,1), bias=False)
        self.conv2 = nn.Conv2d(4, 4, kernel_size=(2,2), stride=(1,1), padding=(1,1), bias=False)
        self.bn1 = nn.BatchNorm2d(1)
        self.bn2 = nn.BatchNorm2d(4)
        self.fc1 = nn.Linear(65536, 1000)
        self.fc2 = nn.Linear(10000, 1000)
        self.fc3 = nn.Linear(1000, 1)

    def num_flat_features(self, x):
        size = x.size()[1:]
        num_features = 1
        for s in size:
            num_features *= s
        return num_features

    def forward(self, x):
        x = self.conv1(x)
        x = self.bn2(x)
        x = F.relu(x)
        x = F.max_pool2d(x,(2,2))

        x = self.conv2(x)
        x = self.bn2(x)
        x = F.relu(x)
        x = F.max_pool2d(x,(2,2))

        x = x.view(-1, self.num_flat_features(x))

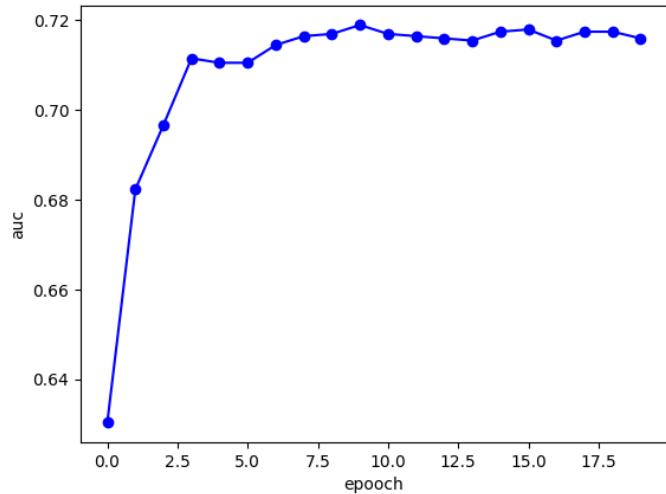
        #print(x.shape)
        x = F.relu(self.fc1(x))

        x = self.fc3(x)

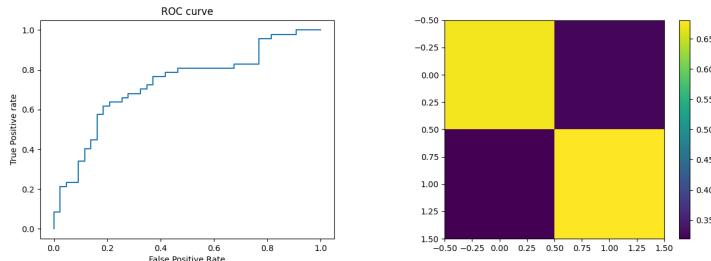
        return x
```

Slika 3: Zgradba konvolucijskega modela.

Konvolucijski model se je kot pričakovano učil bolje kot model (1), spreminjanje AUC tekom učenja je prikazano na sliki (4). Na testni množici je bil $AUC = 0.73$. Ob izbiri praga kot 0.51 lahki izračunamo še $f_1 = 0.69$ in natačnost = 0.68 . V temu modelu sem izbral prag precej višje kot v modelu (1), kar se opazi predvsem na kofuzijski matriki (slika 5), ki je precej bolj uravnovešena kot v modelu (1), kjer je bilo večina veliko več bolnikov uvršečnih v skupini močno bolnih.



Slika 4: AUC v odvisnosti od epoch za model (2).

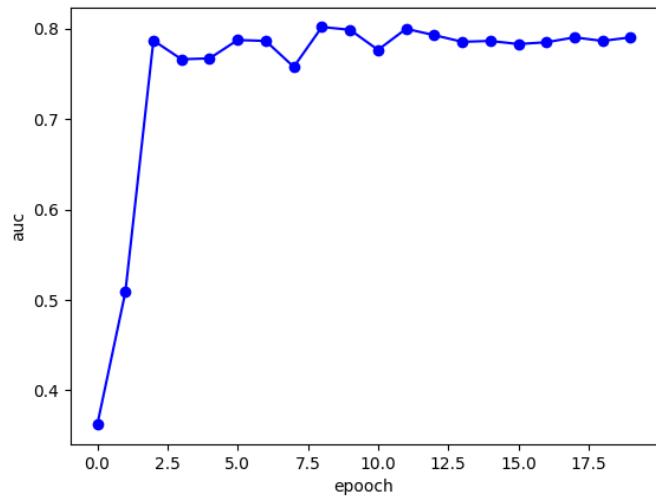


Slika 5: ROC krivulja (levo) in normalizirana konfuzijska matrika (desno) na testni množici.

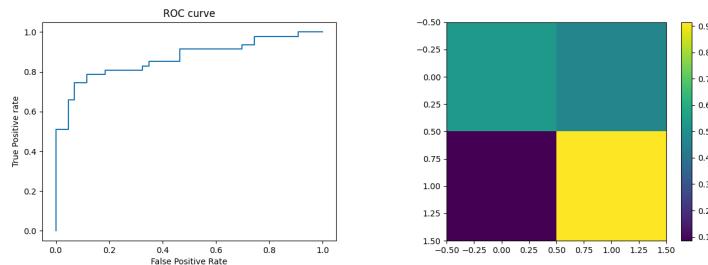
3.3 ResNet18 z ne-naučenimi filterji

Pytorch omogoča zelo preprosto implementacijo kompleksnih konvolucijskih mrež. Mi bomo uporabili mrežo ResNet18 v tem poglavju z ne-naučenimi filterji v naslednjem pa bomo uporabili kar že naučene filtre. Ker je ta mreža precej kompleknejša od modela (2) pričakujemo temu primerno tudi boljše rezultate. Pogljemo si spremjanje AUC med učenjem mreže.

Opazimo, da se je metrika AUC dvignila iz približno 0.7 na 0.8, kar je obetavno za naše napovedi. Na testni množici, pa se je mreža izkazla še bolj saj sem dobili $AUC = 0.87$, ob izbiri praga = 0.24 dobimo vrednosti $f1 = 0.78$.



Slika 6: AUC v odvisnosti od epoch za model (3).

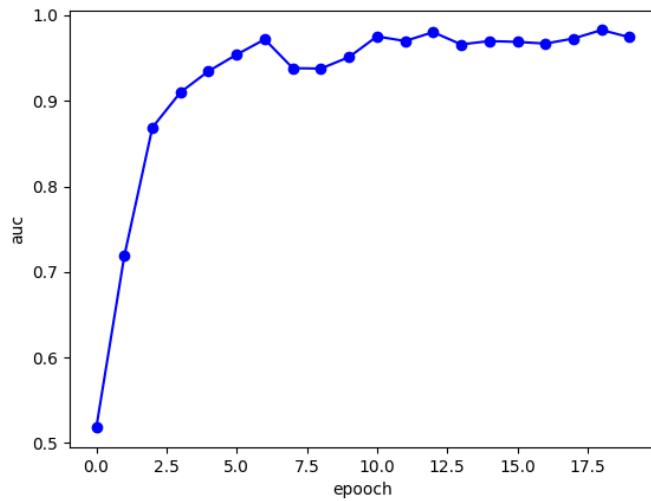


Slika 7: ROC krivulja (levo) in normalizirana konfuzijska matrika (desno) na testni množici.

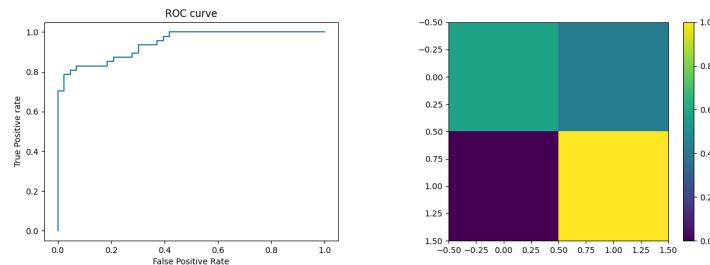
in natančnost = 0.73

3.4 ResNet18 z naučenimi filterji

Pogejmo si še kako deluje model z že naučenimi filterji.



Slika 8: AUC v odvisnosti od epoch za model (4).



Slika 9: ROC krivulja (levo) in normalizirana konfuzijska matrika (desno) na testni množici.

Iz zgornjih grafov lahko vidimo, da model (4) deluje najbolje, to nam pa potrdijo tudi štikle (AUC = 0.94, prag = 0.12, f1 = 0.84, natančnost = 0.8)

3.5 Primerjava modelov

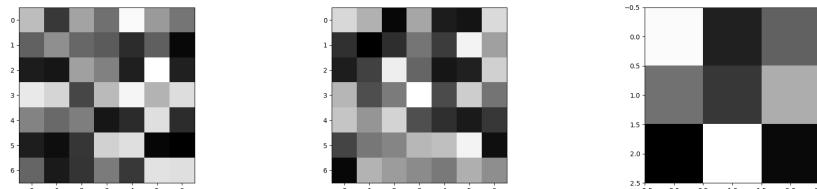
Za lažjo primerjavo bom vse podatke predstavil v tabeli

Model	learning rate	weight decay	multi	AUC	f1	natančnost
Vektor	0.0005	0.00001	0.97	0.68	0.73	0.63
Konvolucijski	0.0005	0.00001	0.97	0.73	0.69	0.68
ResNet18 ne-naučen	0.0004	0.0001	0.97	0.87	0.78	0.73
ResNet18 naučen	0.0002	0.0001	0.95	0.94	0.84	0.80

Kot pričakovano lahko vidimo da sta modela (3) in (4) boljša od ostalih dveh saj sta veliko kompleksnejša. Prav tako je bilo pričakovano da bo model (2) uspešnejši kot model (1). Treba pa ja seveda poudariti, da sta metrike f1 in natančnost odvisni od izbire praga, ki pa ga ni izbiral program ampak jaz. Presenetilo, pa me je, da je model (4) deloval bolje kot model (3), kljbu temu, da bi od modela (3) pričakoval vsaj enako dobro napovedovanje. Verjetno bi to z dovolj dolgim treniranjem in boljšo izbiro hiperparametrov tudi dosegel. Verjetno bi s kakšnimi drugimi hiperparametri izboljšal prav vse modele.

3.6 Vizualizacija filtrov

Zanima nas tudi vizualizacija filtrov, predvsem filterv prvih plasti, saj lahko pričakujemo da bomo v njih opazili določene vzorce.

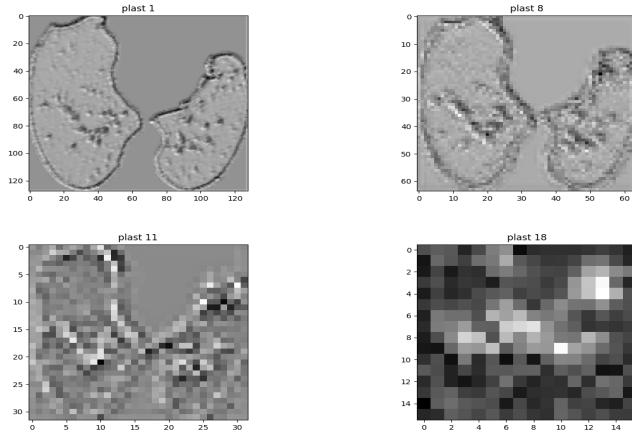


Slika 10: Prikaza najbolj zanimivih filtrov iz prve plasti za model (3), model (4) in model (2) od leve proti desni

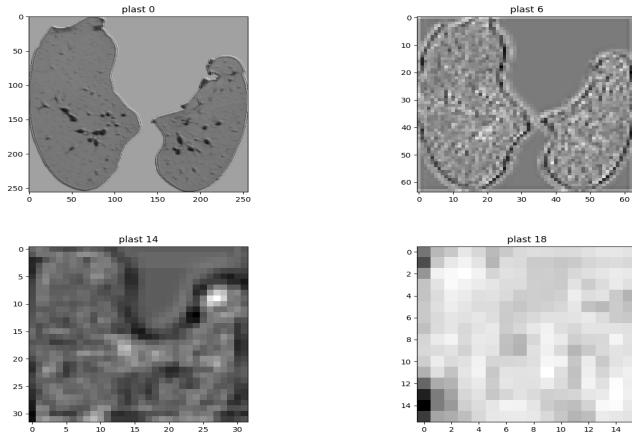
Žal na filtri jaz nisem opazil prav posebnih oblik, ki bi jih lahko analiziral. Z nekaj domišlje lahko na sliki 10 levo opazimo veliko pravokotnih oblik, na sredini črko E, model 3, pa ima filtre velikosti 3x3, kar je verjetno premajhno, da bi opazil kakšno obliko.

3.7 Vizualizacija slike, na različnih globinah mreže

Precej bolj zanimivo je spremjanje spremenjanja slike skozi različne plasti. Pogeljmo si primer za modela (3) in (4):



Slika 11: Prikaz spremnjanja slike skozi različne plasti za model (3)



Slika 12: Prikaz spremnjanja slike skozi različne plasti za model (4)

Vidimo, da lahko v začetnih plasteh zelo lepo vidmo prsni koš, s povendarjenimi deli, z večjo globino pa se slike čedelje bolj "kockajo" in postajajo neprepoznavne.

3.8 Vizualizacija relativnega doprinosa pikslov

Zanima nas tudi na podlagi česa se nevronske mreže odločajo za klasifikacijo v katero od skupin, saj lahko le tako zares preverimo ali se odločajo na podlagi enakih znakov kot zdravniki. Poglejmo si nekaj primerov pravilne klasifikacije v skupino hudo bolnih:

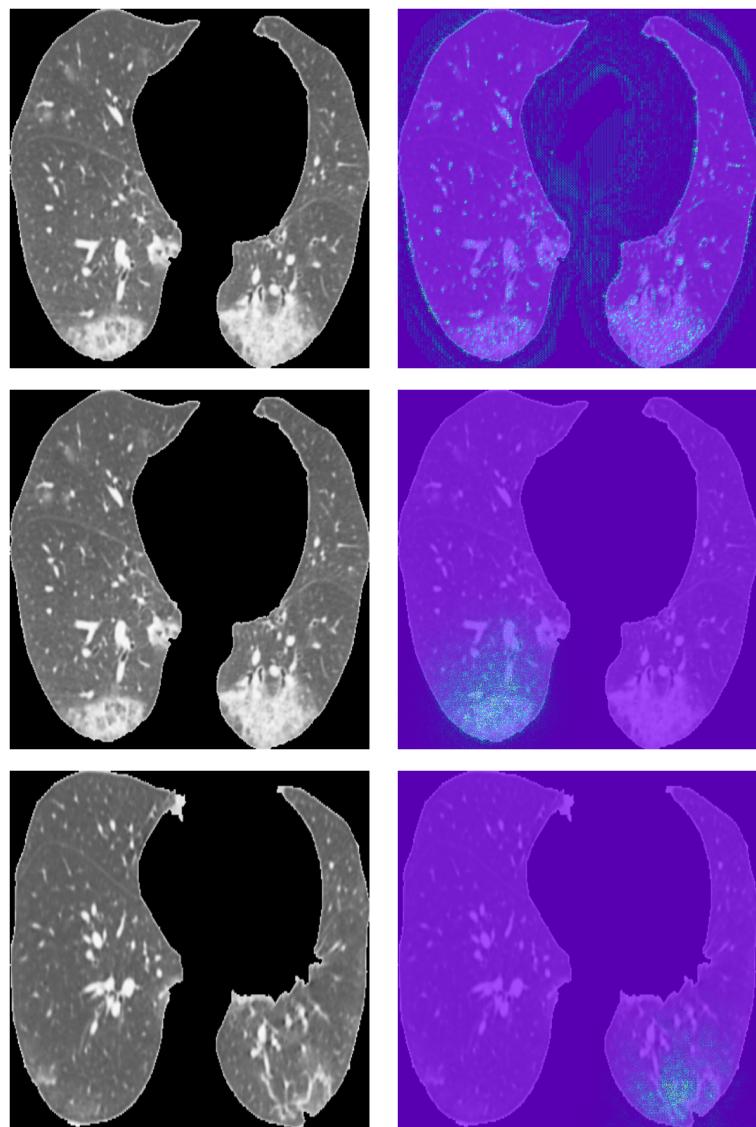
Na sliki 13 opazimo, da so pri modelu (2) pomembni piksli precej razpršeni nekaj jih je celo v zelo nepomembnem območju (črna) medtem ko so pomembni piksli pri modelu (3) in še posebaj (4) zelo skoncentrirani na določene točke v pljučih, ki so verjetno območja mlečnega stekla.

Pogeljmo si še primere napačne klasifikacije, torej primerov bolnikov, ki so bili hudo okuženi, klasificrani pa v skupino z blago okužbo (slika 14).

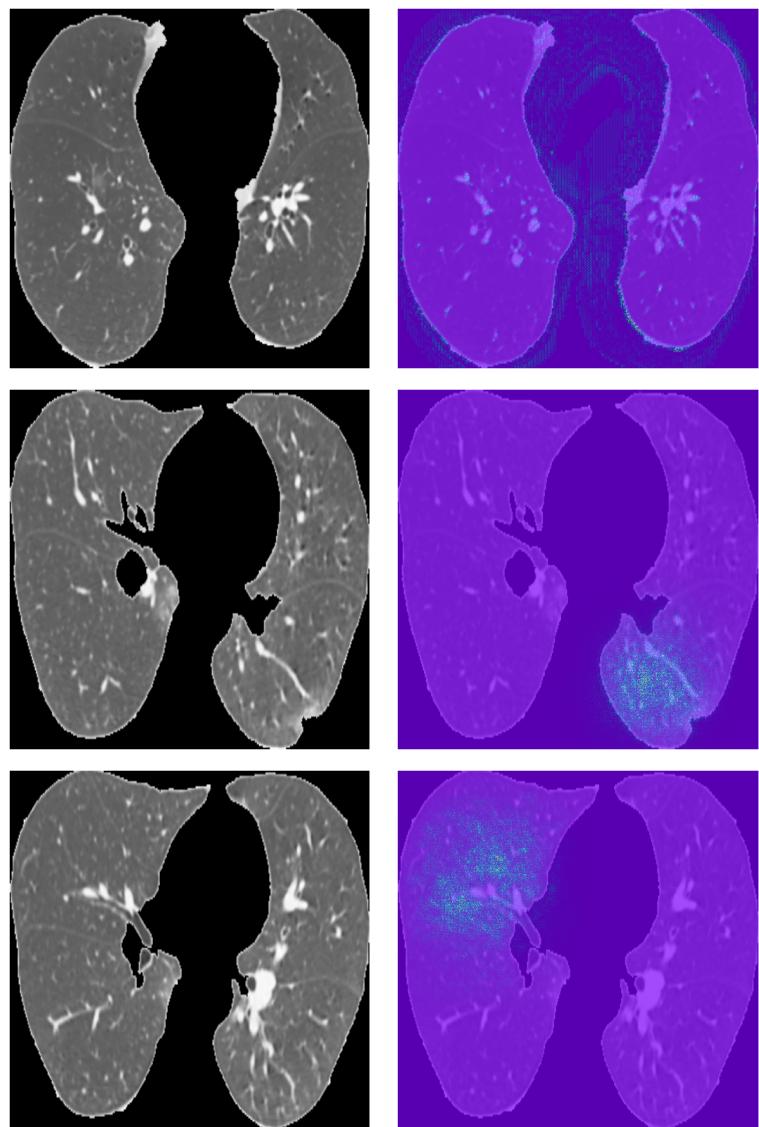
Morda je zanimo pogeldati še skupino slik, v primeru ko nevronska mreža zdravega bolnika označi za bolnega (slika 15)

4 Zaključek

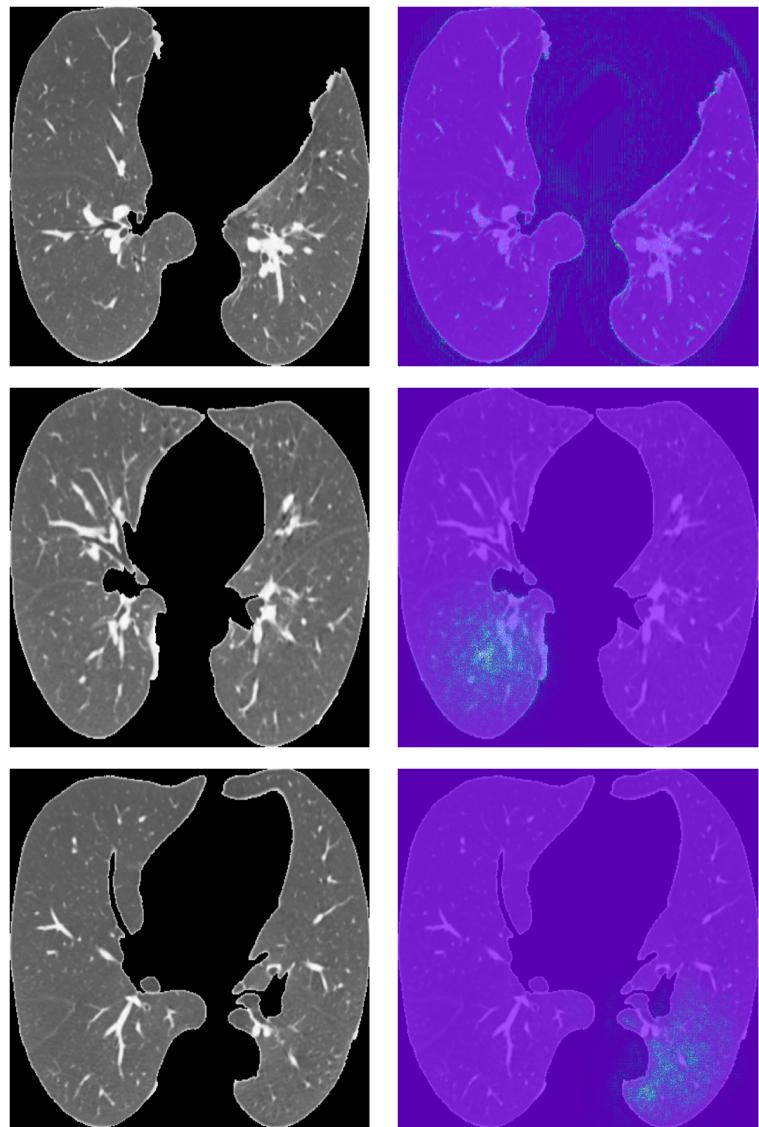
Pri tej nalogi smo se spoznali s konvolucijskimi nevronskimi mrežami, ki so zelo koristne za analizo slik. Prav tako smo uporabili novo knjižnjico za stronjo učenje Pytorch. Naloga je bila zelo aktulana, saj smo sredi epidemije COVID-19 in mi je bila zato precej všeč.



Slika 13: Pravilna kategorizacija v skupino hudo bolni modela (2),(3) in (4) od zgoraj navzdol. Na levi so prikazane originalne slike na desni pa je čez originalne slike prilepljena še ena slika, ki pove kateri piksli so bili najpomebnejši (obarvani svetlo modro) pri odločitvi.



Slika 14: Napačna kategorizacija v skupino blago bolni modela (2),(3) in (4) od zgoraj navzdol. Na levi so prikazane originalne slike na desni pa je čez originalne slike prilepljena še ena slika, ki pove kateri piksli so bili najpomebnejši (obarvani svetlo modro) pri odločitvi.



Slika 15: Napačna kategorizacija v skupino blago hudo modela (2),(3) in (4) od zgoraj navzdol. Na levi so prikazane originalne slike na desni pa je čez originalne slike prilepljena še ena slika, ki pove kateri piksli so bili najpomembnejši (obarvani svetlo modro) pri odločitvi.