Universitat de Girona
Escola Politècnica Superior

Treball Final de Màster

Estudi: Màster en Ciència de Dades

Títol: Classificació de troballes en radiografies de tòrax

Document: Memòria

Alumne: Pau Olivés Tarrés

Tutor: Robert Marti Marly
Departament: Arquitectura i tecnologia de computadors
Àrea: Arquitectura i tecnologia de computadors

Convocatòria (mes/any): Setembre 2022

Universitat de Girona
Escola Politècnica Superior

TREBALL FINAL DE MÀSTER

Classificació de troballes en radiografies de tòrax

Autor:
Pau OLIVÉS TARRÉS

Setembre 2022

Màster en Ciència de Dades

Tutors:
Robert MARTI MARLY

Índex

Índex de figures

Índex de taules

Capítol 1

Introducció

El COVID-19 és un virus que ha tingut un impacte a nivell mundial durant aquests últims dos anys amb un índex de mortalitat elevat [of Medicine 022]. Si fem una cerca d'articles relacionats amb el COVID-19 dins la pàgina de la World Health Organization [Organization 022], trobem que des del 2020 s'han publicat més de 675.000 articles. Tot i ser un tema relativament recent, la comunitat de Intel·ligència Artificial ja ha dedicat molts esforços en contribuir de múltiples maneres per ajudar a fer front al virus.

Ja s'ha vist en el passat que les xarxes neuronals convolucionals tenen la capacitat de treballar amb imatges mèdiques per tal d'assistir amb el diagnòstic o classificació de malalties i també de detectar troballes dins aquestes imatges [Liu 2018]. Pel que fa al diagnòstic del COVID-19, ja existeixen proves relativament econòmiques i fàcils de realitzar com el PCR [Corman 2020], però aquest virus sol causar lesions en els pulmons [Gattinoni 2021] i aquestes no son tan fàcils de detectar, ja que són d'aspecte similar a altres pneumònies bacterianes i virals. És per això que l'interès principal de la comunitat de Intel·ligència Artificial es centra principalment en la creació de models que, donada una radiografia de tòrax, sigui capaç de detectar si hi ha lesions o no i a on es troben aquestes.

Des del grup de recerca de ViCOROB es proposa un projecte de crear models de xarxes neuronals convolucionals que siguin capaces de detectar lesions en els pulmons causades pel COVID-19 a partir d'imatges de radiografies de tòrax amb l'objectiu d'ajudar als radiòlegs a diagnosticar pneumònies. Aquest projecte esta format per diverses parts les quals consisteixen d'una classificació del tipus de pneumònia present a la imatge, una detecció de la regió dels pulmons on hi ha les lesions i una tècnica de preprocés innovadora que es basa en els passos que segueixen els radiòlegs a l'hora de diagnosticar pneumònia en radiografies.

L'objectiu d'aquest treball de final de màster és contribuir en aquest projecte enfocant-nos en la part de classificació. Per tal de fer això s'ha proposat crear models de xarxes convolucionals utilitzant Transfer Learning d'arquitectures ja existents. Apart d'això, s'han proposat diversos experiments de classificació de radiografies per tal de veure quins donen millors resultats i a la vegada comparar les diferents arquitectures a cada experiment.

Les dades utilitzades en aquest treball provenen de la competició de Kaggle SIIM-FISABIO-RSNA COVID-19 Detection [Lakhani 2021], la qual ens ofereix una bona quantitat de dades per fer tant una classificació entre quatre classes

com una detecció de lesions. Cal indicar que, tot i que estem utilitzant les dades de la competició, l'objectiu no és en cap moment participar, ja que la competició va acabar l'Agost del 2021. Tampoc hem pogut fer enviaments de resultats a la competició, ja que aquest requereix tant una classificació com una detecció i aquest treball es centra només en la classificació.

CAPÍTOL 2

Estat de l'art

L'estudi més similar a aquest projecte és [Chouhan 2020] , en el qual utilitzen dades extretes del Guangzhou Women and Children's Medical Center. Les dades estan formades per radiografies de tòrax igual que les nostres, amb la diferència que les etiquetes que tenen son Normal, Pneumònia Bacteriana i Pneumònia Viral, però a l'article simplifiquen aquestes etiquetes com a Normal i Pneumònia. Els experiments que realitzen són fent servir Transfer Learning amb arquitectures de CNN populars. En aquest cas fan servir AlexNet, ResNet18, Inception V3, DenseNet121, GoogLeNet i finalment fan un ensamblatge de tots el models. Els resultats que obtenen són positius, ja que superen els resultats que hi havia anteriorment sobre el dataset que utilitzen arribant fins a 96.36% d'accuracy en el test amb el model ensamblatge, el qual diuen que és el que els hi dona millor resultat amb bastant de marge.

Un article molt recent que fa una recopilació dels mètodes utilitzats per detectar pneumònia en imatges és [Kareem 2022]. Al llarg de l'article parla de varis models de Machine Learning, però en els que es centra més són les CNN i en molts casos menciona que s'han fet servir arquitectures com ResNet18, VGG16 o DenseNet121. Al final de l'article, proposen un model de Federated Learning on hi ha un servidor local i varis dispositius individuals. En aquesta proposta, en lloc de tenir les dades centralitzades en un sol dataset, proposen que cada dispositiu individual tingui les seves dades per ell sol i el servidor passi el model de Machine Learning al dispositius perquè aquests l'entrenin. Un cop entrenat, el model torna al servidor local, el qual el passa a un altre dispositiu perquè continuï l'entrenament amb les seves dadas. Aquest procés continuaria constantment i s'aniria repetint de tant en tant per tal de tenir el model actualitzat amb noves dades que puguin entrar. Aquesta proposta permetria tenir un model adaptat a varis datasets i les dades quedarien privades, ja que el servidor local només envia i rep el model entrenat, no les dades.

Com que les dades que agafem provenen d'una competició de Kaggle que ja ha acabat, podem veure com han plantejat el problema els participants que han obtingut major puntuació. Això ho trobem a [Howard 022], on s'han recollit informes fets per varis participants de la competició, entre els quals hi ha els que han obtingut puntuacions més elevades.

Si mirem l'explicació del guanyador de la competició veiem que han preentrenat un codificador amb els conjunt de dades de chexpert i chest14, des-

prés han entrenat els models per classificar Normal/Pneumonia amb el conjunt de dades rsna pneumonia i finalment han carregat els pesos d'aquests models per entrenar amb les dades de la competició. També utilitzen varies tècniques de Data Augmentation i afegeixen un pesos a la Loss de les classes perquè el model es centri més en aprender de les classes amb menys exemples. Els models que fan servir són SeResnet152-Unet 512, EfficientnetB5-DeeplabV3+ 512, EfficientnetB6-Linknet 448, EfficientnetB7-Unet++ 512 i al final fan un ensamblatge dels quatre models. La mida d'entrada que fan servir pels models és 512x512 i al final fan un comentari de que han intentar utilizar d'entrada 1024x1024, però que els resultats no milloraven.

La resta de grups que han quedat en posicions elevades han realitzat treballs similars on pre-entrenen varis models utilitzant conjunts de dades externs, principalment el chexpert. Després realitzen Data Augmentation i acaben fent un ensamblatge de múltiples models per fer la classificació. Alguns grups afegeixen una cinquena classe a la que anomenen None i alguns agafen només una sola imatge per estudi. Un cas curiós és el grup que ha quedat en segona posició, els quals diuen que no han utilitzat cap conjunt de dades extern, per tant podem suposar que han entrenat els models només amb les dades de la competició.

Els resultats dels diferents grups que han participat a la competició els podem trobar a [Kaggle 022], on observem que els grups que han quedat entre els deu millors tenen una puntuació de entre 0.624 i 0.635.

CAPÍTOL 3

Preliminars

3.1 Arquitectures de Machine Learning

En aquest projecte s'han utilitzat varies arquitectures de xarxes neuronals convolucionals, ja que ens interesa fer una comparativa dels resultats que dona cada una sobre les nostres dades.

3.1.1 EfficientNetB0

El primer model utilitzat ha sigut la EfficientNetB0 [Tan 2019]. Aquesta arquitectura ha estat molt útil a l'hora de realitzar els primers experiments, ja que és un model que escala molt bé i té un cost de computació més baix que altres models.

3.1.2 VGG16

El següent model és el VGG16 [Simonyan 2014]. Aquesta arquitectura és relativament senzilla i consisteix en repetir varis blocs de capes convolucionals abans de fer una capa de pooling. En el seu moment va aconseguir unes millores importants en els resultats de les xarxes neuronals per imatges.

3.1.3 ResNet50

Un altre model utilitzat ha sigut la ResNet50 [He 2016]. Aquesta arquitectura és una de les més utilitzades a l'hora de realitzar Transfer Learning i es basa en les arquitectures de VGG, però afegint un component residual que conecta els resultats d'una capa amb capes més profundes apart de la que ve directament després.

3.1.4 Inception V3

L'arquitectura de Inception V3 [Szegedy 2015] és diferent a les altres, ja que es basa en la teoría de que si les xarxes es fan molt profundes augmenta la possibilitat de tenir overfitting. Per tant, aquesta xarxa és ample en lloc de profunda.

Això ho aconsegueix fent que dins d'una sola capa hi hagi filtres de convolució i pooling de diferents mides, a diferencia del les altres arquitectures que dins d'una sola capa tenen tots els filtres iguals.

3.1.5 Xception

La següent arquitectura és la Xception [Chollet 2016]. Aquesta és molt similar a la Inception V3, però la porta a l'extrem. Mentre que la Inception fa servir un sol filtre de convolució 1x1 a cada capa, la Xception ho aplica després de cada filtre de cada capa.

3.1.6 DenseNet 121

L'última arquitectura utilitzada és la DenseNet 121 [Huang 2017]. Aquesta xarxa és molt profunda i el problema que es troben aquests tipus d'arquitectura és que a l'hora de modificar els pesos en l'entrenament, les últimes capes reben majors modificacions, mentre que les primeres reben modificacions pràcticament nul·les. Això ho soluciona fent que les sortides d'una capa, no es connectin només amb la següent, si no que també es connecta amb capes més profundes.

3.1.7 Ensamblatge

Un cop tenim tots els models entrenats, podem realitzar un ensamblatge. Això consisteix en agafar tots els models que tinguin resultats similars i ajuntar-los. Agafem les prediccions de cada model i ho convertim en una votació per veure quina és la predicció final de l'ensamblatge.

Aquesta tècnica és molt útil perquè permet que si els models fallen en classes diferents, a l'hora de votar guanyarà la classe amb més vots, que suposadament és la correcte. El requeriment és que tots els models que formen part de l'ensamblatge tinguin resultats similars, ja que si n'hi ha un que tingui resultats inferiors, farà que disminueixi el resultat final del conjunt.

3.2 Material

El conjunt de dades utilitzat prové de la competició de Kaggle SIIM-FISABIO-RSNA COVID-19 Detection [Lakhani 2021]. Aquest conjunt conté un total de 6334 imatges de train i 1263 imatges de test, juntament amb anotacions d'experts en forma de fitxer CSV amb tota la informació necessària per identificar la imatge i les anotacions clíniques. Aquests fitxers només contenen informació

Study level	id Negative for Pneumonia Typical Appearance Indeterminate Appearance Atypical Appearance
Image level	id boxes label StudyInstanceUID

Taula 3.1: Anotacions del conjunt de dades donades en els fitxers CSV

sobre les imatges de train, ja que les de tests són les que s'utilitzen per generar els resultats de la competició.

Per aquest projecte no s'han utilitzat les dades de test ja que, com que ens hem enfocat en la classificació, no podem enviar les prediccions del test a la competició per veure els resultats. En lloc d'això, hem agafat només les dades d'entrenament i les hem partit utilitzant un 80% per entrenar i un 20% per validar.

A la taula 3.1 podem observar les capçaleres que trobem en els fitxers CSV de les dades. Dins del fitxer a nivell d'estudi tenim l'identificador de l'estudi (referint-se a un pacient) i una codificació One-hot depenent de si el pacient ha donat negatiu en pneumònia o, en cas de ser positiu, si és una pneumònia d'aparença típica, atípica o indeterminada. Per altre banda, el fitxer a nivell d'imatge hi tenim l'identificador de la imatge, les regions d'interès on l'expert ha detectat les lesions, les etiquetes de les regions i l'identificador de l'estudi al que pertany la imatge.

Al fer un primer cop d'ull a les dades d'aquests dos fitxers, podem veure que en el de nivell d'imatge, les columnes boxes i labels són bastant redundants, ja que a boxes en donen les coordenades (x, y) en pixels de l'origen de les regions juntament amb l'amplada i l'alçada d'aquesta, mentre que a la columna labels ens dóna les coordenades de la regió en format (x1, y1, x2, y2). En cas de no tenir regió d'interès, la columna boxes ho representa amb un NaN i la columna labels amb un None.

Les imatges es troben dins de carpetes corresponents a estudis on dins d'un estudi hi poden haver una o varies imatges. Aquestes imatges estan en format DICOM (Digital Imaging and Communications in Medicine), el qual ens proporciona una matriu de píxels corresponents a l'imatge de la radiografia juntament amb informació sobre el pacient i la imatge. La majoria d'aquesta informació proporcionada pel fitxer DICOM està anonimitzada per motiu de protecció de dades del pacient i no ens és molt útil, per tant ens podem centrar en agafar

només la matriu de píxels.

	id	boxes	label	StudyInstanceUID
0	000a312787f2_image	[{'x': 789.28836, 'y': 582.43035, 'width': 102...	opacity 1 789.28836 582.43035 1815.94498 2499....	5776db0cec75
1	000c3a3f293f_image	NaN	none 1 0 0 1 1	ff0879eb20ed
2	0012ff7358bc_image	[{'x': 677.42216, 'y': 197.97662, 'width': 867...	opacity 1 677.42216 197.97662 1545.21983 1197....	9d514ce429a7
3	001398f4ff4f_image	[{'x': 2729, 'y': 2181.33331, 'width': 948.000...	opacity 1 2729 2181.33331 3677.00012 2785.33331	28dddc8559b2
4	001bd15d1891_image	[{'x': 623.23328, 'y': 1050, 'width': 714, 'he...	opacity 1 623.23328 1050 1337.23328 2156 opaci...	dfd9fdd85a3e

Figura 3.1: Exemple de les primeres files del CSV a nivell d'imatge

	id	Negative for Pneumonia	Typical Appearance	Indeterminate Appearance	Atypical Appearance
0	00086460a852_study	0	1	0	0
1	000c9c05fd14_study	0	0	0	1
2	00292f8c37bd_study	1	0	0	0
3	005057b3f880_study	1	0	0	0
4	0051d9b12e72_study	0	0	0	1

Figura 3.2: Exemple de les primeres files del CSV a nivell d'estudi

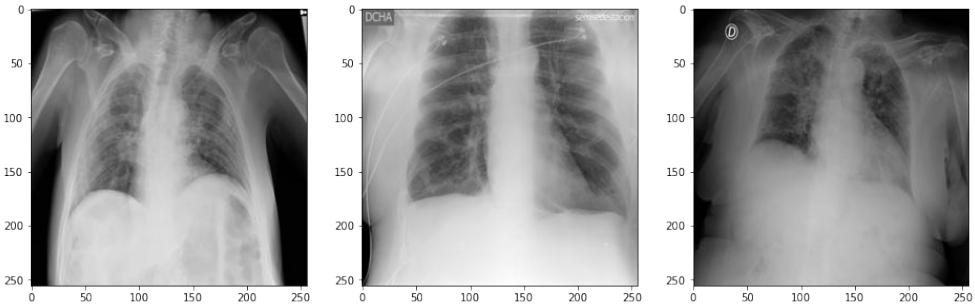

Figura 3.3: Exemple d'imatges

CAPÍTOL 4

Planificació i Metodologia

Aquest projecte està emmarcat dins l'àmbit de recerca i no segueix cap metodologia de desenvolupament de software. Tot i així, fa falta tenir una planificació establerta per saber quins són els objectius i com organitzar el temps disponible. En aquest cas, el treball proposat forma part d'un projecte d'un abast més gran que podem veure a la figura 4.1.

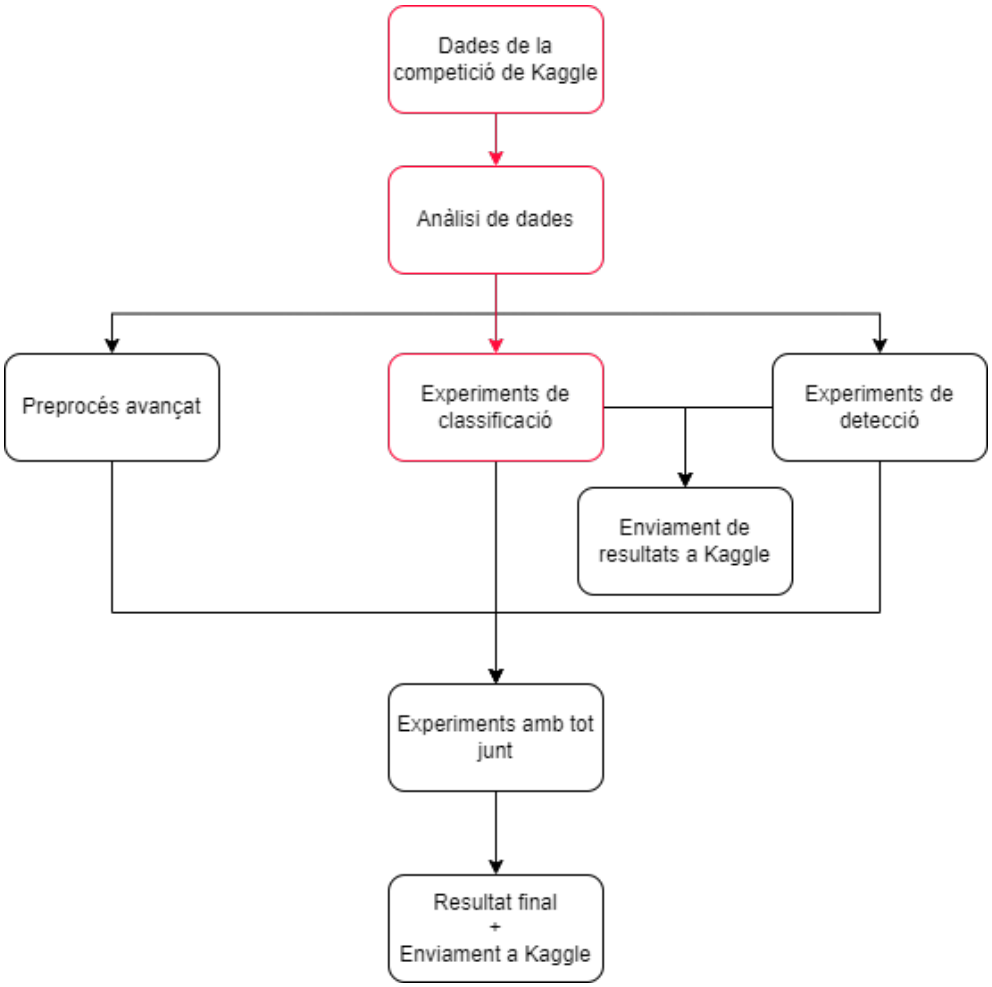

Figura 4.1: Diagrama de la planificació total del projecte

El projecte parteix del conjunt de dades provinent de la competició de Kaggle [Lakhani 2021] i, igual que tots els projectes en ciència de dades, comença

per analitzar les dades. Un cop tenim el coneixement necessari de les dades, el projecte es divideix en tres parts. La primera consisteix en partir d'un model de Machine Learning creat per ViCOROB el qual detecta la regió on hi ha els pulmons i amplia la imatge en aquesta zona, eliminant tot el que hi ha al voltant. La idea és modificar aquest model per tal de que aprengui a detectar els dos pulmons per separat. Això ens interessa per poder simular el procés que realitzen els radiòlegs per tal de detectar pneumònia en les radiografies de tòrax. El procés consisteix en mirar regions concretes dels pulmons que permeten identificar quin tipus de pneumònia hi ha en cas de que hi sigui present.

La segona part consisteix en fer múltiples experiments de classificació amb varies arquitectures de CNN com la ResNet o la Inception per tal de comparar quines s'adapten millor a les dades. Aquesta part pot tardar més o menys depenent de quins siguin els resultats que obtinguem i com variïn amb els diversos experiments i comparacions que realitzem.

La tercera part és similar a la segona, però en lloc de classificació es centra en la detecció de les zones d'interès on es troben les lesions. En aquest cas s'utilitzarien models com la R-CNN o l'algorisme de YOLO i, igual que amb la classificació, s'hi pot dedicar més o menys temps depenent dels resultats obtinguts i les possibles variacions d'experiments que puguem provar.

Amb la classificació i la detecció fetes, podem ajuntar els resultats i fer un enviament a Kaggle per veure en quina posició hauríem quedat. Això ens pot ajudar a orientar-nos de si anem en bona direcció o no.

La part final del projecte és ajuntats totes les parts anteriors en una sola, utilitzar les imatges preprocessades que hem generat i realitzar la classificació i detecció sobre elles per generar un resultat final. Aquest resultat el tornarem a enviar a la competició per veure com hem quedat respecte el primer enviament i la resta de participants.

Aquest treball va començar per analitzar tant les dades com els notebooks públics que hi ha per la competició. Al fer això ens vam adonar que la majoria de notebooks que hi havia públics es centraven principalment en l'anàlisi de les dades i la detecció fent servir l'algorisme de YOLO. Per tant, vam decidir agafar les branques de classificació i preprocés de imatges. Primer vam prioritzar la part de classificació, ja que ens serviria per tenir una base i veure com el preprocés de les dades afecta al resultat. Al final els experiments de classificació ens van requerir més temps del esperat i quan tocava passar a la part del preprocés ja teniem el temps bastant limitat. Per tant, vam decidir en lloc de crear un nou model que agafi els pulmons per separat, agafar el model ja existent i passarlo per les nostres dades per generar imatges on només hi hagi la zona dels pulmons. Amb això, vam aprofitar el temps restant per fer varis experiments de classificació sobre aquestes noves imatges.

CAPÍTOL 5

Contribució Metodològica

5.1 Anàlisi de dades

El primer pas d'aquest projecte és fer un anàlisi de les dades. Ja hem vist una mica l'estructura de les dades a 3.2, però hem d'aprofundir més per veure si totes segueixen l'estructura bé o si tenim possibles outliers. Comencem analitzant els fitxers CSV, ja que aquests són els que contenen la informació necessària per contextualitzar les imatges i per fer la classificació.

Al obrir els fitxers podem veure que tenen un diferent nombre de columnes. El fitxer a nivell d'estudi té 6054 files mentre que el de nivell d'imatges en té 6334. Això és degut a que un estudi pot tenir varies imatges associades, ja sigui perquè s'han pres en diferents instants de temps o perquè les han fet diferents radiòlegs.

El fitxer a nivell d'estudi conté les classes que farem servir a l'hora de classificar amb el model de machine learning. Si contem quants estudis hi ha de cada classe ens trobem lo següent:

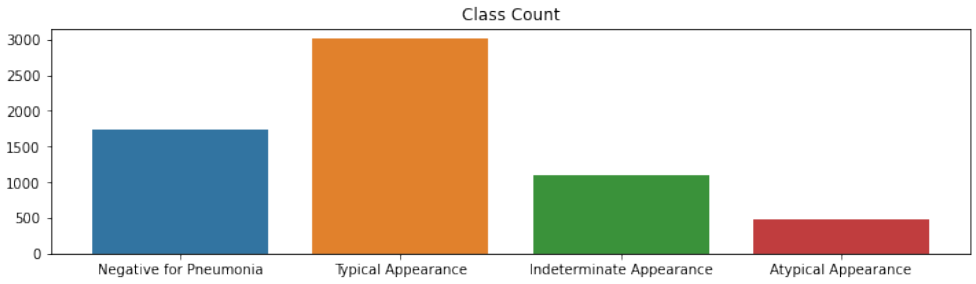

Figura 5.1: Nombre de instàncies pertanyents a cada classe

Observem a la figura 5.1 com quasi la meitat de les instàncies pertanyen a la classe d'aparença típica amb un total de 3007 instàncies. La segona classe més representada és la de negatiu en pneumònia amb 1736 instàncies, seguida per aparença indeterminada amb 1108 instàncies i finalment aparença atípica amb 483 instàncies. Les classes estan clarament desbalancejades i això pot portar problemes a l'hora d'entrenar els models de machine learning.

L'altre fitxer que ens dóna informació és el de nivell d'imatge. Com ja hem vist abans, aquest fitxer conté uns identificadors per poder relacionar les imatges

amb els estudis i dos variables redundants que ens marquen les regions on es troben les lesions. Com que en aquest projecte ens centrem en la classificació de pneumònia, la informació de les regions no les farem servir a l'apartat de machine learning. Tot i això, veure com es relacionen amb els estudis sí que ens pot ajudar a entendre millor les dades. Per això, el que fem és una unió dels dos fitxers utilitzant l'identificador d'estudi per relacionar la informació de les imatges amb la informació dels estudis.

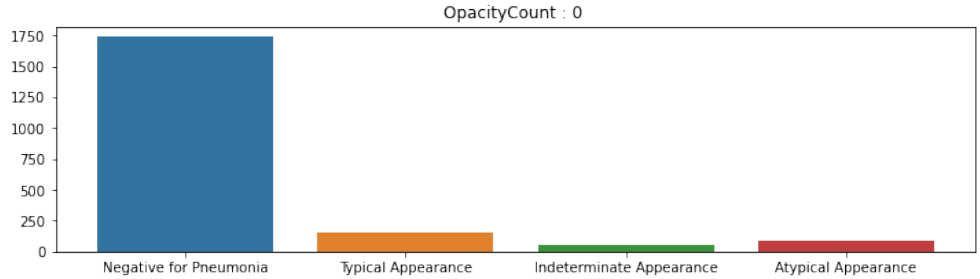

Figura 5.2: Nombre de instàncies sense cap regió pertanyents a cada classe

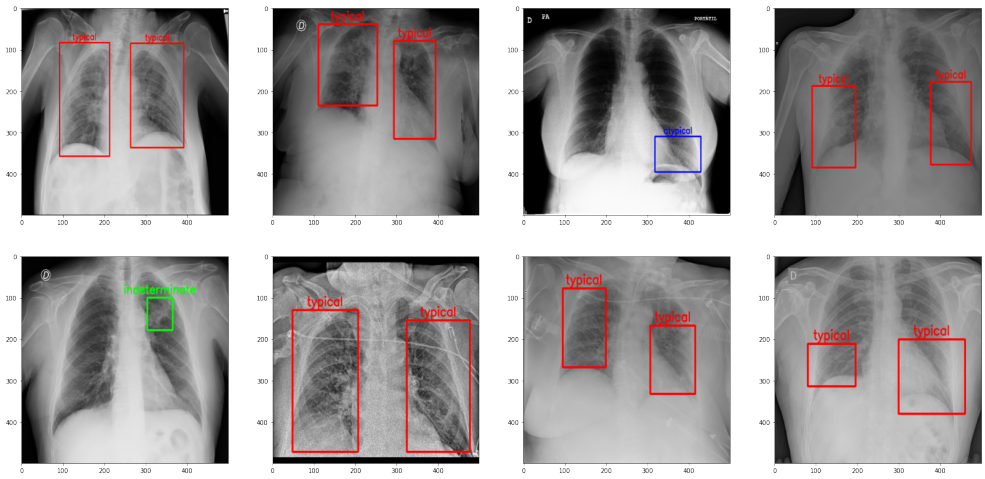

Figura 5.3: Exemples d'imatges amb les regions i les classes a les que pertanyen

Un cop tenim tota la informació junta en un sol fitxer, podem observar més coses com el nombre de regions associades a cada classe o la classe a la que pertanyen les regions de cada imatge com es pot veure a la figura 5.3. Curiosament, si mirem les imatges que no tenen cap regió, ens trobem que la gran majoria són d'estudis de classe negatiu com és d'esperar, però també n'hi ha que pertanyen a estudis d'altres classes. Com observem a la figura 5.2, tenim 153 imatges d'estudis d'aparença típica sense regions, 59 d'aparença indeterminada i 92 d'aparença atípica. El dataset no ens dóna una explicació de perquè hi ha aquests casos, per tant no sabem si són outliers o descuits per part dels experts.

Veient que podem asociar una classe a cada regió, una idea interessant és crear un mapa de calor on es mostri les zones de les imatges on hi ha més densitats de regions. Per fer això, partim d'una imatge en negre i per cada imatge d'una classe augmentem la intensitat de tots els píxels que estiguin dins de les regions d'aquella imatge. Al final normalitzem aquestes intensitats, ja que, com que cada classe té un diferent nombre d'imatges, les classes amb més instàncies tindrien més intensitat que les classes amb menys instàncies. Si els resultats del mapa de calor ens mostren que cada classe té una major densitat de regions en diferents zones de la imatge, ens ajudaria molt a poder adaptar els models a que es fixin en zones concretes per diferenciar les classes.

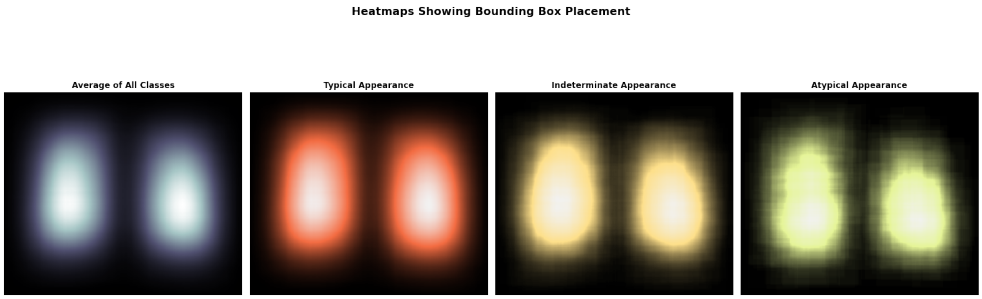

Figura 5.4: Mapa de calor de les regions de cada classe

El resultat d'aquest procés el veiem a la figura 5.4, on es pot observar que les densitats de regions són molt similars a cada classe. La principal diferència és que, amb el procés de normalització d'intensitats, les classes amb menor nombre d'instàncies es veu més pixelades a les zones de menys densitat. Al no haver-hi gaire diferència en les posicions de les regions per cada classe, no podrem utilitzar aquesta informació per tal d'adaptar els models de machine learning.

5.2 Preprocés de dades

Tot i que el preprocés de les dades no ha sigut la part principal del treball, s'ha realitzat un tractament a les dades per tal d'alleugerir el cost de computació i veure si amb una tècnica innovadora milloren els resultats.

Les imatge del nostre conjunt de dades no tenen una mida fixe i solen tenir una resolució molt alta on la majoria són de dimensió 2000x2000 o major. Això és un problema, ja que els models esperen que totes les dades tinguin les mateixes mides. En el nostre cas, els models que es fan servir estan pre-entrenats amb dades de mida 224x224, per tant, un del preprocessos que realitzarem serà escalar totes les imatges a aquesta mida. Apart d'això, també escalarem les dades a 512x512 per verificar si la mida de les imatges afecta molt als resultats.

La tècnica innovadora utilitzada ha estat desenvolupada pel grup de recerca ViCOROB de la Universitat de Girona. Aquesta tècnica consisteix en utilitzar un model d'aprenentatge automàtic que, donada una imatge de radiografia de tòrax, detecti els pulmons i retalli la imatge per tal de que només es vegin aquests. Aquest model ja està entrenat amb altres dades i només cal executar-lo sobre les nostres. Els resultats es poden veure a la figura 5.5.

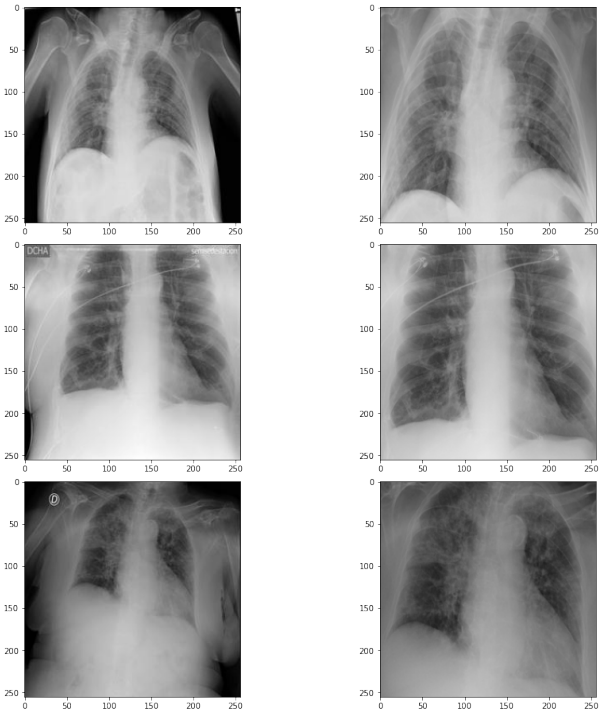

Figura 5.5: Comparació d'imatges abans i després del preprocés

L'expectativa és que al ampliar la zona del pulmons, la qual és on es troben les lesions, estem eliminant tot el soroll del voltant i el model es pot centrar més en les zones d'interès.

5.3 Data Augmentation

La tècnica de Data Augmentation consisteix en modificar les imatges d'entrenament per tal d'augmentar les nostres dades. Existeixen dos variants d'aquesta tècnica les quals tenen els seus avantatges i inconvenients.

La primera variant consisteix en agafar les imatges originals i fer petites modificacions que normalment consisteixen en rotacions, translacions, canvis d'intensitat, fer zoom, voltejat i, en alguns casos, escalat. Això permet que el

model s'adapti a diferents situacions que no siguin sempre ideals i, si un cop entrenat, li entra alguna imatge amb soroll, pugui resoldre millor la situació.

La segona variant és similar a la primera, però en lloc de substituir les imatges originals per les modificades, el que fa és afegir les imatges modificades com a noves dades del conjunt d'entrenament. Aquesta variant és útil sobretot quan es té poques dades per entrenar el model, ja que n'estem generant de noves a partir de les originals. Al ser imatges modificades, no hauria d'afectar a l'overfitting i, normalment, aconsegueix disminuir-lo. La desavantatge d'aquesta variant és que, com que estem generant més dades, augmenta el cost de computació i el temps requerit per entrenar el model.

Per aquest projecte, s'ha decidit utilitzar la primera variant, així permetem que el model s'adapti a imatges no ideals sense augmentar el cost de computació, que en el nostre cas és limitat. Les modificacions aplicades a les nostres imatges són: rotació, translació tant vertical com horitzontal, zoom i canvis d'intensitat. Aquestes modificacions són aleatòries dins un rang petit i diferents per cada imatge. Hem exclòs les modificacions de voltejat perquè el pulmó esquerra i dret tenen certes diferències i no es poden canviar de posició i l'escalat perquè els nostres models esperen imatges de mida fixa, per tant no la podem modificar aleatòriament.

Per aconseguir això, hem utilitzat l'objecte ImageDataGenerator de Keras, el qual ens permet realitzar totes aquestes petites modificacions i, a més a més, ens permet organitzar les imatges i separar-les en conjunt d'entrenament i validació.

5.4 Tranfer Learning / Fine Tuning

Aquest treball es centra en fer una comparativa en com diferents models de xarxes neurals aprenen de les nostres dades. Els models que s'han utilitzat (3.1) són arquitectures pre-entrenades amb el conjunt de dades ImageNet [Deng 2009]. Això ho fem perquè per entrenar un model de Deep Learning com els utilitzats, és necessari tenir molta capacitat de computació. Per tant, és usual agafar models pre-entrenats amb conjunts de dades de grans dimensions, ja que normalment no es poden trobar conjunts d'aquestes dimensions del camp d'estudi, en aquest cas, radiografies de tòrax.

La majoria de proves que s'han fet durant el treball han sigut fent servir només Tranfer Learning. Aquest mètode consisteix en agafar el models pre-entrenats com a base i fer-lo servir per extreure característiques. Després s'afegeix una o varies capes que realitzaran la classificació. Durant l'entrenament amb les nostres dades, el model base estarà congelat, el que significa que els seus pesos no variaran, l'únic que entrenem són les capes que hem afegit que ens retornen el resultat de la classificació.

En tots els models base que hem agafat, les capes que hem afegit al final són una fully-connected de 256 neurones i 50% de dropout que processarà les característiques extretes del model base i una capa amb 4 neurones que realitzarà la classificació.

Una prova que s'ha realitzat ha sigut fer servir la tècnica de Fine Tuning. Aquesta tècnica consisteix en aplicar Transfer Learning, però un cop hem entrenat les capes de classificació, es descongela el model base i es torna a entrenar tot el model amb un learning rate molt baix per acabar d'adaptar els pesos de la xarxa a les dades. En el projecte no s'han realitzat moltes proves fent servir Fine Tuning, ja que requereix de un alt cost de computació per entrenar i tendeix a l'overfitting.

5.5 Experiments

Per tal d'entrenar els models, hem realitzat multiples experiments utilitzant dades tractades amb diferents preprocessos o diferents classificacions.

5.5.1 Primer entrenament

El primer experiment ha sigut fer un entrenament amb les imatges sense cap preprocess apart de l'escalat de dimensions a 224x224. En aquest cas, intentem classificar directament les 4 classes amb un sol model.

La mida de batch que hem fet servir per tots els models és 64 i hem posat les imatges a mode rgb, ja que alguns models esperen una entrada de tres dimensions. Les nostres imatge són a escala de grisos, per tant per aconseguir el mode rgb simplement es triplica la imatge original per representar els tres canals.

Al principi vam fer un entrenament amb 10 epochs, però vam veure que els resultats del model s'estabilitzaven bastants ràpid i no feia falta un entrenament llarg. Això segurament és perquè estem fent servir Transfer Learning i només estem entrenant les últimes capes del model que realitzen la classificació. Finalment vam decidir quedar-nos amb un entrenament de 5 epochs per cada model, el qual és suficient per veure si un model té potencial de millorar i no és tan llarg com per consumir més temps de computació del necessari.

Aquest experiment ens permet tenir una base per no només comparar els models entre si, si no que també podem comparar aquest experiment amb els següents per veure si millorem els resultats o no. Al ser un experiment bastant bàsic, no hem tingut problemes amb el cost de computació.

5.5.2 Classificació binaria

Amb el primer experiment vam veure que els models solen classificar les imatges de validació com Negatiu o Aparença Típica i ignoraven les altres dos classes. Veient això, vam pensar que podria ser bona idea fer una classificació en dos pases. Primer faríem un model que classifiqui entre Negatiu i Positiu i després, en cas de ser Positiu, entrenaríem un segon model que en digui a quina de les tres classes de positiu correspon la imatge.

Aquest segon experiment és igual que el primer amb la diferència que fem una classificació binaria entre Positiu i Negatiu en lloc de les quatre classes originals. La idea era fer també el segon model que classifiqués entre les tres classes de Positiu, però vam decidir que és millor començar només per la classificació binaria, ja que si aquesta no ho fa bé, no té sentit aplicar el segon model.

Inicialment vam veure que els resultats eren bons perquè teníem a la propia avaluació del model una accuracy elevada comparada amb el primer experiment, però al inspeccionar la matriu de confusió dels resultats vam veure que això és degut a que les classes no estan balancejades i que la classificació és bastant errònia. Per tant, vam decidir no continuar amb el segon pas del model i dedicar el recursos a altres experiments.

5.5.3 Entrenament amb Fine Tuning

Com ja s'ha mencionat a la secció anterior, un dels experiments realitzats ha sigut fent servir Fine Tuning. Al principi l'haviem plantejat com el primer experiment, però apart de realitzar les primeres 5 epochs amb el model base congelat, també afegiríem un segon entrenament de 5 epochs amb el model base descongelat i un learning rate reduit de 1e-5. Al veure que durant la primera part de l'entrenament els models no milloraven i que durant la segona part si que tenien potencial de millora, vam decidir canviar el nombre d'epochs amb el model congelat a 3 i les del model descongelat a 15 per observar com evolucionen els models. Una excepció realitzada ha estat el model de EfficientNetB0, el qual és el que mostrava més señal de poder millorar i vam realitzar un entrenament amb 30 epochs del model descongelat.

El primer problema que ens vam trobar al fer això és que al descongelar el model base, el nombre de pesos a entrenar augmenta en gran quantitat i la memòria de la màquina utilitzada per l'entrenament no és capaç de suportar això juntament amb les imatges. Per tant, vam haver de disminuir la mida de batch de 64 a 16. Això ens va permetre poder entrenar un model, però al intentar entrenar el següent ens tornem a trobar que ens quedem sense memòria, ja que tenim dos models carregats al mateix temps. La solució final que vam trobar és entrenar els models un a un i reiniciar l'entorn després de cada entrenament,

així alliberem la memoria pel següent model.

Els resultats d'aquest experiment mostren que alguns models tenen potencial per millorar i obtenir millors resultats que fent servir només Transfer Learning, però la lleugera millora obtinguda no justifica l'elevat cost i temps de computació necessari per realitzar Fine Tuning. Per tant, vam decidir descartar aquesta tècnica d'entrenament i passar al següent experiment.

5.5.4 Entrenament amb dades preprocessades

Com hem comentat a la secció 5.2, hem preprocessat les imatges de tal manera que la regió dels pulmons està ampliada i eliminem tot el soroll del voltant.

Aquest experiment és bàsicament el mateix que el primer canviant les dades utilitzades de les originals a les preprocessades. Això ens servirà per veure si aquest mètode de preprocess de dades millora els resultats o no.

Apart d'això, també hem fet una classificació binaria amb aquestes dades per veure si millorem el resultat del segon experiment.

5.5.5 Entrenament amb dades escalades

L'últim entrenament que hem pogut fer és utilitzant dades de mida 512x512 en lloc de 224x224. Al augmentar la mida de les imatges ens hem tornat a trobar amb el problema de falta de memòria que teníem amb el Fine Tuning i hem tingut que tornar a reduir el batch size a 16 i entrenar els models un a un. A més a més, augmentar la mida de les imatges repercuteix exponencialment en el temps d'entrenament, fent que entrenar models en aquest experiment fos molt costós tant computacionalment com de temps.

Al entrenar els primers models i veure que no només estaven tardant massa, si no que també teníem overfitting dels models, vam decidir para l'experiment i no continuar-lo.

CAPÍTOL 6

Resultats

A la secció 5.5 hem vist el plantejament dels experiments realitzats durant el projecte i en aquest capítol veurem els resultats d'aquests experiments i comentarem que podem aprendre d'ells i com han influït en les decisions preses en els següents experiments.

6.1 Detalls de l'implementació

Tots els experiments s'han implementat en Python 3.10 i utilitzant Jupyter Notebook. S'ha fet servir Anaconda per instal·lar les llibreries i la llibreria que hem fet servir per implementar els models és la versió GPU de Keras.

Tot el codi s'ha executat en local en un ordinador amb Windows 10, procesador Intel Core i7, 16GB de RAM i GPU Nvidia GeForce RTX 2060.

6.2 Discrepàncies entre evaluate i predict

Inicialment per avaluar els models feiem servir la funció model.evaluate() de Keras amb el conjunt de validació per tal de saber quina accuracy tenim. Al veure que teníem bons resultats vam decidir fer un model.predict() també de Keras amb el mateix conjunt i comparar les prediccions amb les etiquetes reals per tal de crear una matriu de confusió. Al fer això vam veure que els resultats obtinguts amb el predict són diferents que els obtinguts amb l'evaluate. La diferencia en casi tots el casos es de aproximadament 20%. No sabem a què és deguda aquesta diferència i amb una recerca exhaustiva d'informació en la documentació de Keras i buscar casos similars, tot indica a que el resultat hauria de ser el mateix, tot i que nosaltres hem trobat aquesta diferència constantment en tots els experiments.

6.3 Resultats del primer entrenament

El que ens interessa del primer experiment és poder executar-lo ràpid per tenir una base i poder veure com son els resultats i com els podríem millorar. Al

acabar l'entrenament dels models podem mirar com ha evolucionat l'accuracy al llarg de les epochs i obtenim la figura 6.1.

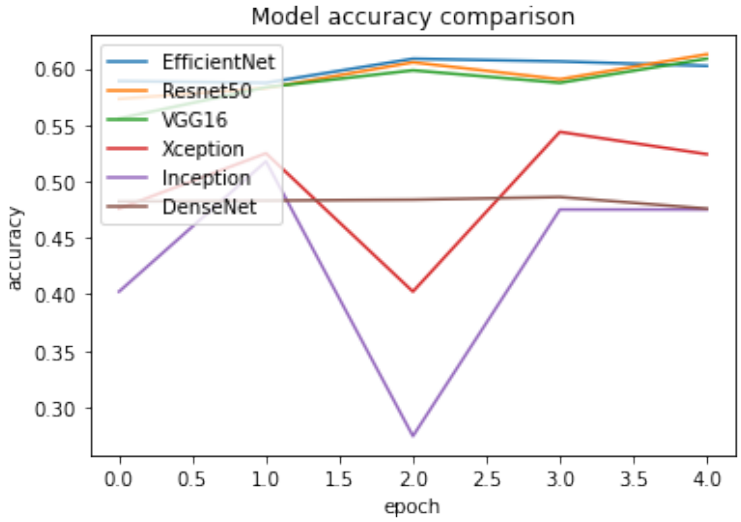

Figura 6.1: Comparació de l'evolució de les accuracies dels models durant el primer entrenament

Observem que els models amb millor accuracy han sigut la EfficientNetB0, la ResNet50 i la VGG16 acabant totes al voltant del 60% d'accuracy. En canvi, observem que el model Xception està entre el 50 i 55% i els models Inception i DenseNet estan entre 45 i 50%.

És important mirar que no tenim overfitting i per això en interessa mirar la comparació dels resultats de validació (els quals Keras anomena test) respecte els d'entrenament. Si mirem per exemple el de la ResNet50, veiem la figura 6.2.

Podem observar com la trajectòria tant de validació com d'entrenament pugen cap adalt. En cas de tenir overfitting veuríem com l'accuracy d'entrenament puja mentre que la de validació baixa. La resta de models tenen unes gràfiques similars, lo qual ens indica que l'entrenament ha anat bé i no hem obtingut overfitting.

Finalment, ens interessa veure les matrius de confusió de cada model per observar com s'estan comportant els resultats. Aquí és on generem les prediccions dels models i les comparem amb les etiquetes reals. Al fer això observem que les accuracies dels models no es corresponen a les accuracies que hem vist abans en els gràfics. D'aquesta forma obtenim la taula 6.1.

Podem observar com els models de Inception i DenseNet que abans estaven per sota, ara són els que tenen millors resultats. El problema es que si mirem les matrius de confusió d'aquests dos models ens surt la taula 6.2, on veiem que totes les prediccions són la mateixa: Aparença Típica. El resultat d'aquests

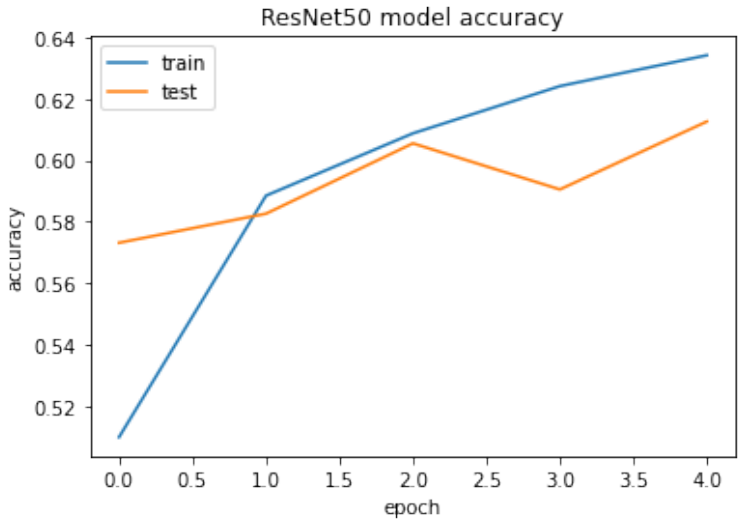

Figura 6.2: Comparació de l'evolució de les accuracies de validació i entrenament del model ResNet50 durant l'entrenament

Model	EfficientNetB0	ResNet50	VGG16	Xception	Inception V3	DenseNet121
Accuracy	44%	40%	40%	37%	47%	48%

Taula 6.1: Accuracies dels models en el primer experiment

models es correspon a la quantitat d'imatges d'aquesta classe que representa un 48% de les nostres dades. Curiosament, aquests són els únics resultats on l'evaluate i el predict són iguals.

Predit / Real	**Atípica**	**Indeterminada**	**Negatiu**	**Típica**
Atípica	0	0	0	96
Indeterminada	0	0	0	221
Negatiu	0	0	0	347
Típica	0	0	0	601

Taula 6.2: Matriu de confusió del model DenseNet121

Al veure això, hem descartat aquests models a l'hora de fer el model d'ensamblatge, el qual hem fet ajuntat els models de EfficientNetB0, ResNet50 i VGG16, ja que són els models amb resultats més similars. Al fer el procés de predicció de l'ensamblatge, obtenim que la accuracy és del 41%, el qual no representa una millora respecte el model EfficientNetB0, però si respecte els altres dos.

6.4 Resultats classificació binaria

En aquest experiment ens interessa sobretot mirar si estem classificant bé els positius i negatius, ja que això determinarà si seguim aquest experiment classificant els tres tipus de positius o si el deixem de banda i passem a un altre experiment.

Al fer l'entrenament dels models obtenim la figura 6.3. Observem com en aquest cas, les accuracies del primers tres models i la DenseNet121 ronden el 76-80% d'accuracy, mentre que la Inception i Xception estan solapades al voltant de 73% i no milloren.

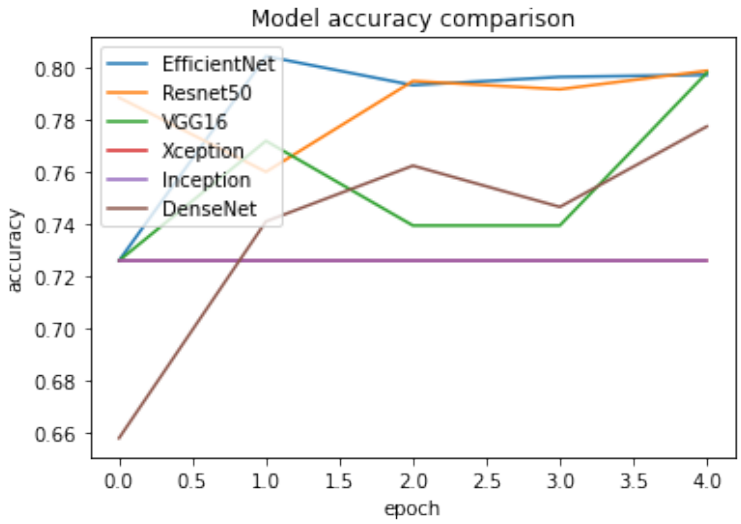

Figura 6.3: Comparació de l'evolució de les accuracies dels models durant l'entrenament de la classificació binaria

Si observem les gràfiques individuals de cada model com hem fet en l'experiment anterior, observem com cap dels models té overfitting.

En el moment de fer la predicció, obtenim els resultats de la taula 6.3. Tornem a observar com els resultats la Inception i Xception es corresponen als del evaluate. Això torna a ser perquè classifiquen totes les imatges com a positives i aquesta classe representa un 73% de les nostres dades. Per altre banda, observem com aquesta vegada el model amb millor accuracy ha sigut la DenseNet121, per tant l'inclourem a l'hora de fer l'ensamblatge apart dels primeres tres.

El resultat del model d'ensamblatge ens dona una accuracy del 63%, la qual es inferior a la DenseNet121, però moderadament major a les altres tres.

Tot i que sembla que amb això obtenim millors resultats que el primer experiment, si observem la precision i el recall de la clase negativa veiem que ronden

el 25%. Això vol dir que, tot i que els models sí que estan intentant classificar entre positius i negatius, estan fallant molt sobretot de la classe negativa. Per ara no val gaire la pena intentar fer la classificació entre les tres classes de positius, ja que segurament els resultats baixarien molt i serien pitjors que el primer experiment, però sí que pot valer la pena en un futur intentar millorar aquesta classificació binaria i continuar amb aquest experiment.

Model	EfficientNetB0	ResNet50	VGG16	Xception	Inception V3	DenseNet121
Accuracy	60%	57%	57%	73%	73%	65%

Taula 6.3: Accuracies dels models en la classificació binaria

6.5 Resultats de l'entrenament amb Fine Tuning

Com hem mencionat a la subsecció 5.5.3, en aquest experiment no hem pogut entrenar tots els models en una sola execució per limitació de recursos i no s'ha pogut generar un gràfic comparant l'evolució de les accuracies de tots els models durant l'entrenament. El que sí que podem mirar són els gràfics individuals de cada model i el que observem és que, com es veu a la figura 6.4, l'accuracy d'entrenament puja mentre que la de validació es queda estable. Això indica que el model té overfitting.

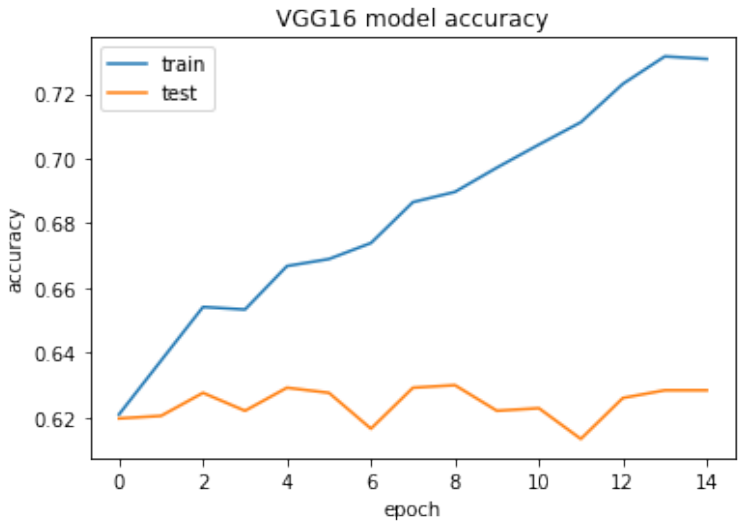

Figura 6.4: Comparació de l'evolució de les accuracies de validació i entrenament del model VGG16 durant l'entrenament

Aquesta situació passa amb tots els models a excepció de la EfficientNetB0, la qual sembla que tant l'accuracy d'entrenament com la de validació augmenten

a la vegada. És per això que vam decidir fer un entrenament excepcional de 30 epochs per veure com evolucionen els resultats. Al fer això obtenim la figura 6.5, la qual sembla indicar que podem arribar a obtenir accuracies superiors a les d'experiments anteriors.

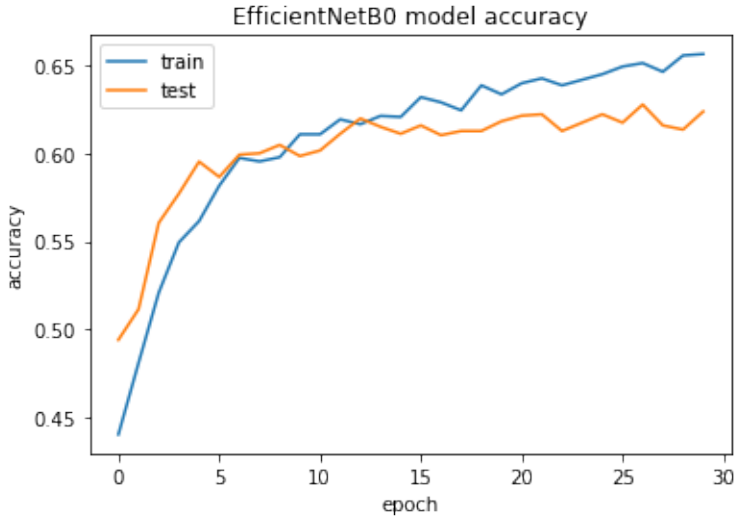

Figura 6.5: Comparació de l'evolució de les accuracies de validació i entrenament del model EfficientNetB0 durant l'entrenament

Finalment, al fer les prediccions obtenim els resultats de la taula 6.4. Observem com les accuracies són inferiors al primer experiment, però si mirem les matrius de confusió, veiem que les prediccions no es centren només en Negatiu i Aparença Típica, sinó que també intenten classificar les altres dues classes. Això és una bona senyal i, si aconseguissim reduir l'overfitting, tindríem una gran possibilitat de millora. Malauradament, com hem comentat abans, aquest experiment té un gran cost de computació i hi ha altres experiments que volem realitzar, per tant l'hem deixat temporalment aparcat.

Model	EfficientNetB0	ResNet50	VGG16	Xception	Inception V3	DenseNet121
Accuracy	42%	37%	40%	40%	38%	39%

Taula 6.4: Accuracies dels models en l'entrenament amb Fine Tuning

6.6 Resultats de l'entrenament amb dades preprocessades

Aquest experiment és com el primer, però utilitzant les dades preprocessades per veure si aquestes ajuden a millorar el resultat. El resultat de l'entrenament del

Model	EfficientNetB0	ResNet50	VGG16	Xception	Inception V3	DenseNet121
Accuracy	42%	40%	38%	48%	33%	31%

Taula 6.5: Accuracies dels models en l'entrenament amb dades preprocessades

model el veiem a la figura 6.6 i observem com, al igual que el primer experiment, els tres primers models estan per sobre dels altres amb una accuracy del voltant de 55-60%.

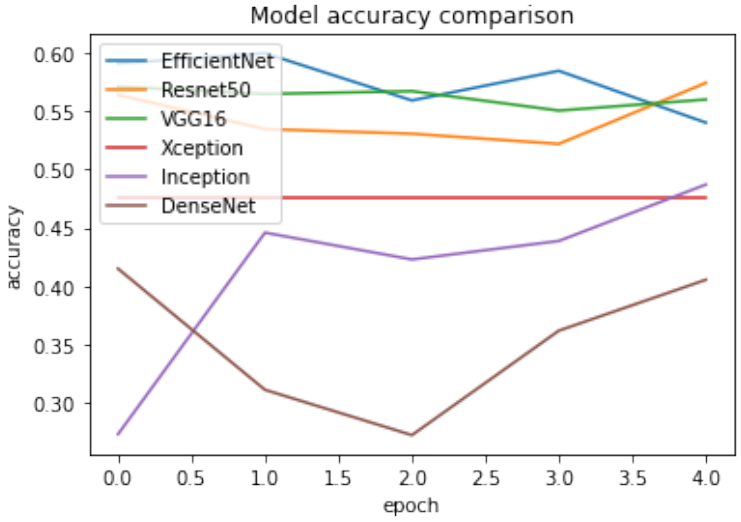

Figura 6.6: Comparació de l'evolució de les accuracies dels models durant l'entrenament amb les dades preprocessades

Si mirem les prediccions obtenim la taula 6.5, on tornem a observar resultats molt similars al primer experiment. També observem que el model de Xception ens ho classifica tot com la classe més popular, aconseguint així una accuracy més alta, però que no ens serveix de res.

En aquest cas tornem a fer un model d'ensamblatge amb els primers tres models i obtenim una accuracy de 40%.

A més a més, hem realitzat aquest experiment sobre aquestes dades aplicant la classificació binaria. El resultat de l'entrenament en aquest cas el veiem a la figura 6.7, on observem que en aquest cas els dos models amb millor resultat han sigut la EfficientNetB0 i la VGG16 amb accuracies del voltant de 78-80%. En aquest cas, també ens trobem al mirar les gràfiques individuals de cada mode que tots els models a excepció d'aquests dos mencionats tenen overfitting.

Al mirar les prediccions obtenim la taula 6.6 on podem destacar que les accuracies del dos models que no han tingut overfitting són més elevades que en l'experiment de classificació binaria sense utilitzar dades preprocessades. Al

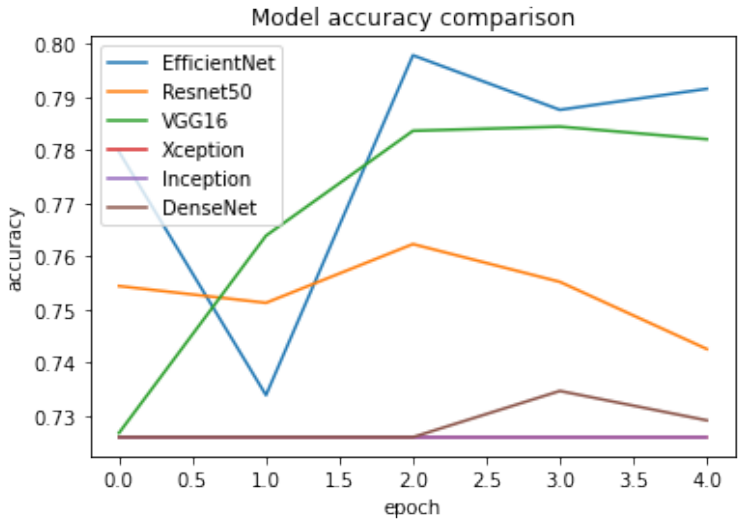

Figura 6.7: Comparació de l'evolució de les accuracies dels models durant l'entrenament amb les dades preprocessades i classificació binaria

Model	EfficientNetB0	ResNet50	VGG16	Xception	Inception V3	DenseNet121
Accuracy	62%	64%	68%	73%	73%	67%

Taula 6.6: Accuracies dels models en l'entrenament amb dades preprocessades i classificació binaria

tenir només dos models utils no podem crear un ensamblatge, però seria interessant buscar alguna manera de reduir l'overfitting dels altres models per veure si realment obtenir bons resultats amb tots o la majoria.

Al acabar aquest experiment ja ens estàvem quedant sense temps i vam decidir intentar realitzar un últim experiment.

6.7 Resultats de l'entrenament amb dades escalades

En aquest últim experiment hem utilitzat les dades escalades a 512x512. Com era d'esperar, això va provocar que el temps d'entrenament augmentés en gran quantitat, fent que cada epoch tardi hores en acabes. Apart d'això ens vam tornar a trobar amb el problema de que la màquina no té memòria suficient per suportar totes les imatges, fins i tot després de reduir el batch size a 16.

Al final, com que no ens quedava gaire temps, en aquest experiment només hem entrenat dos models que són la EfficientNetB0 i la ResNet50. Els resultats es poden veure a les figures 6.8 i 6.9 i observem clarament que hi ha overfitting

desde les primeres epochs, per tant vam decidir aturar l'experiment, ja que té un cost de computació massa elevat i els resultats no són bons.

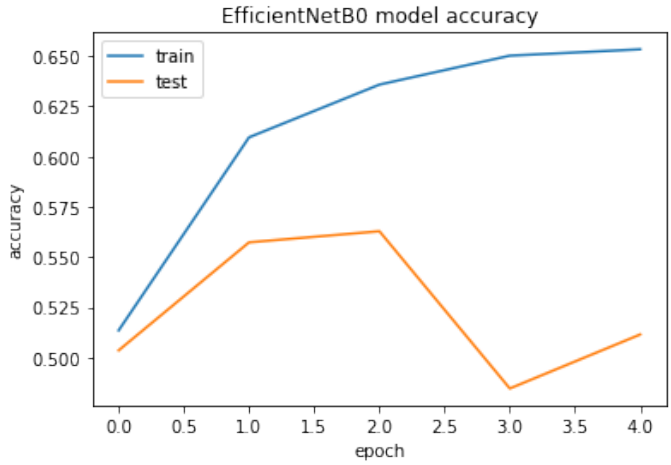

Figura 6.8: Comparació de l'evolució de les accuracies de validació i entrenament del model EfficientNetB0 durant l'entrenament

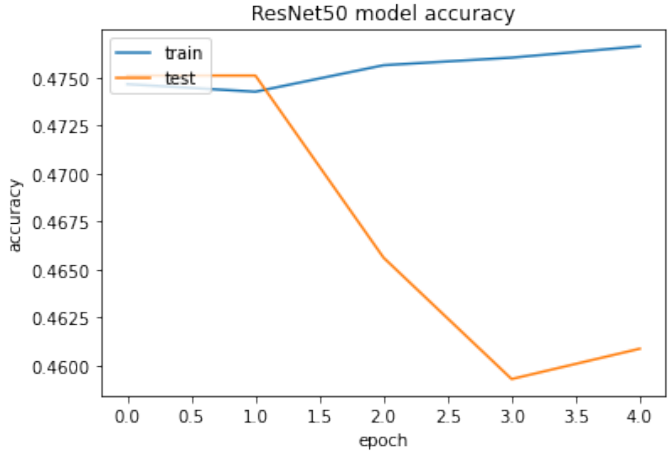

Figura 6.9: Comparació de l'evolució de les accuracies de validació i entrenament del model ResNet50 durant l'entrenament

CAPÍTOL 7

Conclusions i treball futur

En aquest projecte s'han realitzat varis experiments comparant com diversos models de Transfer Learning actuen en diferents situacions sobre un conjunt de dades d'imatges de radiografies de tòrax. Els resultats mostren que, de les arquitectures que s'han provat, que millor s'adapten a aquestes dades són la EfficientNetB0, la ResNet50 i la VGG16. La DenseNet121 ha donat resultats similar a les primeres tres en alguns experiments i en altres experiments s'ha sobreajustat i ha donat mals resultats. La Inception V3 i la Xception són les que pitjors resultats han donat, ja que en casi tots els experiments han etiquetat totes les imatges de validació com una sola classe, però en l'experiment del Fine Tuning han mostrat possibilitat de classificar entre les quatre classes.

Apart de la comparació de models entre ells, també s'ha creat models ensamblatge ajuntant el que tenien resultats similars, però aquests no han aconseguit millorar els resultats dels millors models de cada experiment.

Dels experiments realitzats, hem trobat que els models en general tenen un gran biaix per les classes Negatiu i Aparença Típica. L'excepció ha sigut l'experiment amb Fine Tuning, on sembla que cap dels models té biaix per les classes i les accuracies obtingudes no són molt inferiors a la resta d'experiments.

També hem observat com el preprocessar les imatges ampliant la zona dels pulmons no ha afectat a la classificació de quatre classes, pero si que ha millorat el resultat d'alguns models per la classificació binaria. Un possible projecte futur pot ser intentar modificar el preprocés per tal de mirar si millora el resultat d'alguna manera.

L'experiment on hem augmentat la mida de les imatges no ha sortit bé, ja que, no només ha consumit un gran temps de computació durant l'entrenament, sinó que també ha resultat ser el que té més overfitting i menor accuracy de validació.

El principal problema que ens hem trobat sobretot en els últims experiments és la falta de recursos de computació. Això ha afectat principalment a l'experiment amb Fine Tuning i l'experiment on hem escalat les imatges a una mida major. Pot valer la pena invertir més recursos per tal de realitzar un Data Augmentation que generi més dades en lloc de modificar i substituir les que ja tenim per tal de veure si aconseguim reduir l'overfitting del models al fer Fine Tuning. Apart d'això, es podria pre-entrenar les dades amb conjunt de dades com el chexpert en lloc de fer servir ImageNet.

Per un treball futur, el següent pas del projecte seria realitzar experiments de la detecció de regions a les imatges partint dels algorismes de R-CNN i YOLO. Això ens permetrà ajuntar la classificació amb la detecció per tal de fer un primer enviament a la competició de Kaggle i veure en quina posició quedem.

Alternativament, podríem realitzar el preprocés avançat com a següent pas del projecte i veure com aquest afecta d'alguna forma als resultats de la classificació. El problema de fer això és que seguiriem sense poder fer un enviament al Kaggle, per tant, lo ideal seria fer el preprocés després dels experiments de detecció. D'aquesta manera tindríem un primer enviament a la competició com a base i veuríem millor com els canvis que realitzem afecten al resultat final.

Bibliografia

[Chollet 2016] François Chollet. *Xception: Deep Learning with Depthwise Separable Convolutions*. arXiv:1610.02357 [cs.CV], 2016. (Cited on page 6.)

[Chouhan 2020] V. Chouhan, S.K. Singh, A. Khamparia, D. Gupta, P. Tiwari, C. Moreira, R. Damaševičius and V.H.C. de Albuquerque. *A Novel Transfer Learning Based Approach for Pneumonia Detection in Chest X-ray Images*. Appl. Sci. 2020, 10, 559. https://doi.org/10.3390/app10020559, 2020. (Cited on page 3.)

[Corman 2020] V.M. Corman, Landt O., Kaiser M., Molenkamp R. and et al. *Detection of 2019 novel coronavirus (2019-nCoV) by real-time RT-PCR*. Euro Surveill. 2020;25(3):pii=2000045. https://doi.org/10.2807/1560-7917.ES.2020.25.3.2000045, 2020. (Cited on page 1.)

[Deng 2009] J. Deng, W. Dong, R. Socher, L.J. Li, K. Li and L. Fei-Fei. *Imagenet: A large-scale hierarchical image database*. 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255, 2009. (Cited on page 15.)

[Gattinoni 2021] L. Gattinoni, S. Gattarello, I. Steinberg, M. Busana, P. Palermo, S. Lazzari, F. Romitti, M. Quintel, K. Meissner, J.J. Marini, D. Chiumello and L. Camporota. *COVID-19 pneumonia: pathophysiology and management*. European Respiratory Review 2021 30: 210138; DOI: 10.1183/16000617.0138-2021, 2021. (Cited on page 1.)

[He 2016] K. He, X. Zhang, S. Ren and J. Sun. *Deep residual learning for image recognition*. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778, 2016. (Cited on page 5.)

[Howard 022] Addison Howard. *Leaderboard Finalized - Congratulations to the Winners; Recap*, (Accessed: June 2022). Available at https://www.kaggle.com/competitions/siim-covid19-detection/discussion/266057. (Cited on page 3.)

[Huang 2017] G. Huang, Z. Liu, L. Van Der Maaten and K.Q. Weinberger. *Densely connected convolutional networks*. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4700–4708, 2017. (Cited on page 6.)

[Kaggle 022] Kaggle. *SIIM-FISABIO-RSNA COVID-19 Detection: Leaderboard*, (Accessed: May 2022). Available at https://www.kaggle.com/competitions/siim-covid19-detection/leaderboard. (Cited on page 4.)

[Kareem 2022] A. Kareem, H. Liu and P. Sant. *Review on Pneumonia Image Detection: A Machine Learning Approach*. Hum-Cent Intell Syst 2, 31–43 (2022). https://doi.org/10.1007/s44230-022-00002-2, 2022. (Cited on page 3.)

[Lakhani 2021] P. Lakhani, J. Mongan, C. Singhal, Q. Zhou, K.P. Andriole, W.F. Auffermann, P. Prasanna, T. Pham, M. Peterson, P.J. Bergquist, T.S. Cook, S.F. Ferraciolli, G.C. de Antonio Corradi, M. Takahashi, S.S. Workman, M. Parekh, S. Kamel, J.H. Galant, A. Mas-Sanchez, E.C. Benítez, M. Sánchez-Valverde, L. Jaques, M. Panadero, M. Vidal, M. Culiáñez-Casas, D.M. Angulo-Gonzalez, S.G. Langer, M. de la Iglesia Vaya and Shih G. *The 2021 SIIM-FISABIO-RSNA Machine Learning COVID-19 Challenge: Annotation and Standard Exam Classification of COVID-19 Chest Radiographs*. OSF Preprints. Available from: osf.io/532ek, 2021. (Cited on pages 1, 6 and 9.)

[Liu 2018] N. Liu, L. Wan, Y. Zhang, T. Zhou, H. Huo and Fang T. *Exploiting convolutional neural networks with deeply local description for remote sensing image classification*. IEEE Access, vol. 6, pp. 11215-11228, 2018, doi: 10.1109/ACCESS.2018.2798799, 2018. (Cited on page 1.)

[of Medicine 022] Johns Hopkins University of Medicine. *Coronavirus Resource Centre.*, (Accessed: August 2022). Available at https://coronavirus.jhu.edu/. (Cited on page 1.)

[Organization 022] World Health Organization. *Global literature on coronavirus disease*, (Accessed: August 2022). Available at https://search.bvsalud.org/global-literature-on-novel-coronavirus-2019-ncov/. (Cited on page 1.)

[Simonyan 2014] K. Simonyan and A. Zisserman. *Very deep convolutional networks for large-scale image recognition*. arXiv preprint arXiv:1409.1556, 2014. (Cited on page 5.)

[Szegedy 2015] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna. *Rethinking the Inception Architecture for Computer Vision*. arXiv:1512.00567 [cs.CV], 2015. (Cited on page 5.)

[Tan 2019] M. Tan and Q. Le. *Efficientnet: Rethinking model scaling for convolutional neural networks*. International Conference on Machine Learning, pp. 6105–6114. PMLR, 2019. (Cited on page 5.)