



# Leaf disease detection with a convolutional neural network with tensorflow

**Adrià Falgueras Vidal**

Grau d'Enginyeria informàtica

Intel·ligència Artificial

**Joan M. Nuñez Do Rio**

**David Isern Alarcón**

20/05/2020



Aquesta obra està subjecta a una llicència de  
[Reconeixement-NoComercial-CompartirIgual 3.0](#)  
[Espanya de Creative Commons](#)

## FITXA DEL TREBALL FINAL

<b>Títol del treball:</b>	<i>Leaf disease detection with a convolutional neural network using tensorflow</i>
<b>Nom de l'autor:</b>	<i>Adrià Falgueras Vidal</i>
<b>Nom del consultor/a:</b>	<i>Joan M. Nuñez Do Rio</i>
<b>Nom del PRA:</b>	<i>David Isern Alarcón</i>
<b>Data de lliurament (mm/aaaa):</b>	<i>05/2020</i>
<b>Titulació o programa:</b>	<i>Grau d'Enginyeria Informàtica</i>
<b>Àrea del Treball Final:</b>	<i>Intel·ligència artificial</i>
<b>Idioma del treball:</b>	<i>Català</i>
<b>Paraules clau</b>	<i>Leaf disease detection, classification, CNN</i>
<b>Resum del Treball (màxim 250 paraules):</b> Amb la finalitat, context d'aplicació, metodologia, resultats i conclusions del treball	
<p>Degut al creixement exponencial de la població mundial i la previsió que aquesta segueixi en augment durant el present segle, es fa imprescindible buscar mètodes que permetin assegurar que es disposi d'aliments per a tothom. Un punt clau és aconseguir evitar que es malmetin cultius degut a virus i fongs que a vegades són difícils de diferenciar a simple vista.</p> <p>L'objectiu principal de l'estudi és explorar una xarxa neural capaç de reconèixer malalties en cultius a partir de fotografies de les seves fulles. L'estudi es duu a terme a partir de la utilització d'un model base al qual es van afegint tècniques i processos per tal de millorar-lo. També s'utilitza la tècnica de transfer learning amb el model pre-entrenat VGG16. Finalment, es discuteixen tots els resultats obtinguts per extreure conclusions.</p> <p>Per a la creació de la CNN s'utilitzarà la biblioteca de Tensorflow i Keras i s'entrenarà amb el conjunt de dades preexistent de PlantVillage. En aquest, s'ha dividit les imatges en els grups d'entrenament, validació i test en unes proporcions del 78%, 16% i 6% respectivament.</p> <p>Els resultats finals han mostrat que un model que utilitza regularització L2 s'adapta bé a aquest tipus de dades i ha obtingut en el test una precisió del 94% i una exhaustivitat també propera al 94%. Per aquest motiu, es considera que l'objectiu s'ha acomplert satisfactòriament i que per tant és viable crear una CNN per a la detecció de malalties en plantes.</p>	

**Abstract (in English, 250 words or less):**

Because of exponential growth of world population and the prevision that that still growing in the present century, is necessary to find methods which ensure that food is available for everyone. A key point is to prevent crop damage due to viruses and fungi that are sometimes difficult to differentiate with the naked eye.

The main objective of the study is to explore a neural network capable of recognizing diseases in crops from photographs of their leaves. The study is done using a basic model and adding to it techniques and processes in order to improve it. The transfer learning technique with the pre-trained model VGG16 is also used. Finally, all the results obtained to draw conclusions are discussed.

The Tensorflow and Keras library will be used to create CNN and it will be trained with the pre-existing PlantVillage dataset. In this, the images were divided into the training, validation and test groups in proportions of 78%, 16% and 6% respectively.

The final results have shown that a model that uses L2 regularization performs correctly to this type of data and has obtained the test a precision of 94% and also an accuracy close to 94%. For this reason, it is considered that the objective has been satisfactorily achieved and it is feasible to create a CNN to detect diseases in plants.

# Índex

<b>1. INTRODUCCIÓ .....</b>	<b>1</b>
1.1 - CONTEXT I JUSTIFICACIÓ DEL TREBALL .....	1
1.2 - OBJECTIUS DEL TREBALL.....	3
1.2.1 – <i>Generals</i> .....	3
1.2.2 – <i>Específics</i> .....	3
1.3 - ENFOCAMENT I MÈTODE SEGUIT .....	3
1.4 - PLANIFICACIÓ DEL TREBALL.....	5
1.5 - BREU SUMARI DE PRODUCTES OBTINGUTS .....	6
1.6 – ESTRUCTURA .....	6
<b>2. LES XARXES NEURALS CONVOLUCIONALS .....</b>	<b>8</b>
2.1 – INTRODUCCIÓ AL APRENENTATGE COMPUTACIONAL.....	8
2.2 – LES XARXES NEURALS CONVOLUCIONALS.....	8
2.2.1 – <i>Introducció</i> .....	8
2.2.2 – <i>Preprocessament de les imatges</i> .....	10
2.2.3 – <i>La convolució</i> .....	10
2.2.4 – <i>Bias</i> .....	13
2.2.5 – <i>Funció d'activació</i> .....	13
2.2.6 – <i>Pooling</i> .....	14
2.2.7 – <i>Flatten</i> .....	15
2.2.8 – <i>Capes completament connectades (fully Connected)</i> .....	16
<b>3. CONJUNT DE DADES.....</b>	<b>17</b>
<b>4. METODOLOGIA .....</b>	<b>21</b>
4.1 – PREPROCESSAMENT DE LES DADES.....	21
4.2 – ARQUITECTURA .....	22
4.2.1 – <i>Model base</i> .....	22
4.2.2 – <i>Model amb batch normalization</i> .....	23
4.2.3 – <i>Model amb dropout</i> .....	24
4.2.4 – <i>Model VGG16</i> .....	25
4.3 – VALIDACIÓ .....	25
4.3.1 – <i>Conjunts de dades</i> .....	25
4.3.2 – <i>Augment de dades</i> .....	26
4.3.3 – <i>Conceptes</i> .....	27
4.3.4 – <i>Mètriques utilitzades</i> .....	27
<b>5. EXPERIMENTS .....</b>	<b>30</b>
5.1 – MODEL BASE .....	30
5.1.1 – <i>Entrenament</i> .....	30
5.1.2 – <i>Resultats</i> .....	31
5.2 – EXPERIMENT AMB DADES AUGMENTADES .....	32
5.2.1 <i>Entrenament</i> .....	33
5.2.2 – <i>Resultats</i> .....	34
5.3 – EXPERIMENT AMB REGULARITZACIÓ L2 .....	35
5.3.1 – <i>Entrenament</i> .....	35
5.3.2 – <i>Resultats</i> .....	36
5.4 – MODEL AMB BATCH NORMALIZATION .....	38

<i>5.4.1 – Entrenament</i> .....	38
<i>5.4.2 – Resultat</i> .....	39
5.5 – MODEL AMB DROPOUT.....	41
<i>5.5.1 – Entrenament</i> .....	41
<i>5.5.2 – Resultats</i> .....	42
5.6 – MODEL VGG 16.....	44
<i>5.6.1 – Entrenament</i> .....	45
<i>5.6.2 – Resultats</i> .....	46
5.7 – DISCUSSIÓ.....	47
5.8 – FUTURS ESTUDIS.....	52
<b>6. CONCLUSIONS.....</b>	<b>53</b>
<b>7. BIBLIOGRAFIA .....</b>	<b>54</b>

## Llista de figures

IL·LUSTRACIÓ 1 EVOLUCIÓ DE LA POBLACIÓ MUNDIAL (ROSER ET AL., 2020).....	1
IL·LUSTRACIÓ 2 GRÀFIC DE GANTT DE LA PLANIFICACIÓ DEL TREBALL .....	6
IL·LUSTRACIÓ 3 ARQUITECTURA TÍPICA D'UNA XARXA NEURAL CONVOLUCIONAL.....	9
IL·LUSTRACIÓ 4 REPRESENTACIÓ D'UNA NEURONA DURANT UNA CONVOLUCIÓ .....	9
IL·LUSTRACIÓ 5 IMATGE DE FULLA EN FORMAT MATRIU .....	10
IL·LUSTRACIÓ 6 EXEMPLE DE L'OPERACIÓ DE CONVOLUCIÓ.....	11
IL·LUSTRACIÓ 7 EXEMPLE DE CONVOLUCIÓ AMB PADDING .....	11
IL·LUSTRACIÓ 8 MOSTRA DELS PRIMERS 8 MAPES DE CARACTERÍSTIQUES OBTINGUDES EN 5 CONVOLUCIONS AMB EL MODEL DE XARXA NEURAL PRE-ENTRENAT VGG16, CADA IMATGE ÉS RESULTAT D'UN FILTRE DIFERENT (DERTAT, 2017) .....	12
IL·LUSTRACIÓ 9 EXEMPLE DEL FUNCIONAMENT DEL BIAS EN UNA FUNCIÓ DE L'ESTIL $F(x) = mx + c$ ON $c$ EQUIVAL AL BIAS .....	13
IL·LUSTRACIÓ 10 EXEMPLE DE MAXPOOLING AMB FILTRE DE 2x2 I STRIDE 2 .....	15
IL·LUSTRACIÓ 11 PROCÉS DE FLATTENING .....	15
IL·LUSTRACIÓ 12 ÚLTIMES CAPES DE LA XARXA NEURAL CONVOLUCIONAL .....	16
IL·LUSTRACIÓ 13 DATASET PLANTVILLAGE EXEMPLES IL·LUSTRATIUS DE DIFERENTS CLASSES .....	18
IL·LUSTRACIÓ 14 GRÀFIC DE LA DISTRIBUCIÓ DE LES IMATGES DEL CONJUNT D'ENTRENAMENT.....	19
IL·LUSTRACIÓ 15 DISTRIBUCIÓ PER ESPÈCIE EN L'ENTRENAMENT.....	20
IL·LUSTRACIÓ 16 EXEMPLS D'IMATGES ELIMINADES .....	21
IL·LUSTRACIÓ 17 ESKUEMA DE L'ARQUITECTURA DEL MODEL BASE .....	22
IL·LUSTRACIÓ 18 ESKUEMA DE L'ARQUITECTURA AMB BATCH NORMALIZATION ENTRE LES CAPES DE CONVOLUCIÓ I POOLING	23
IL·LUSTRACIÓ 19 ESKUEMA DE L'ARQUITECTURA AMB DROPOUT .....	24
IL·LUSTRACIÓ 20 ARQUITECTURA DEL MODEL VGG16 (UL HASSAN, 2018) .....	25
IL·LUSTRACIÓ 21 RESULTATS D'EXACTITUD I PÈRDUA DURANT L'ENTRENAMENT DEL MODEL BASE .....	31
IL·LUSTRACIÓ 22 Matriu de confusió del model base generat amb les dades de test .....	31
IL·LUSTRACIÓ 23 EXEMPLS DE LES TRANSFORMACIONS D'UNA IMATGE .....	33
IL·LUSTRACIÓ 24 RESULTATS D'EXACTITUD I PÈRDUA DURANT L'ENTRENAMENT DEL MODEL AMB DADES AUGMENTADES.....	34
IL·LUSTRACIÓ 25 Matriu de confusió del model amb dades augmentades generat amb les dades de test .....	34
IL·LUSTRACIÓ 26 RESULTATS D'EXACTITUD I PÈRDUA DURANT L'ENTRENAMENT DEL MODEL AMB REGULARITZACIÓ L2 .....	36
IL·LUSTRACIÓ 27 Matriu de confusió del model amb regularització L2 generat amb les dades de test .....	37
IL·LUSTRACIÓ 28 RESULTATS D'EXACTITUD I PÈRDUA DURANT L'ENTRENAMENT DEL MODEL AMB BATCH NORMALIZATION .....	39
IL·LUSTRACIÓ 29 Matriu de confusió del model amb batch normalization generat amb les dades de test .....	40
IL·LUSTRACIÓ 30 RESULTATS D'EXACTITUD I PÈRDUA DURANT L'ENTRENAMENT DEL MODEL AMB BATCH NORMALIZATION I DROPOUT .....	42
IL·LUSTRACIÓ 31 Matriu de confusió del model amb batch normalization i dropout generat amb les dades de test .....	43
IL·LUSTRACIÓ 32 RESULTATS D'EXACTITUD I PÈRDUA DURANT L'ENTRENAMENT DEL MODEL VGG16 .....	46
IL·LUSTRACIÓ 33 Matriu de confusió del model VGG 16 generat amb les dades de test .....	46
IL·LUSTRACIÓ 34 GRÀFIC QUE MOSTRA LA PRECISSIÓ OBTINGUDA A CADA CLASSE EN CADASCUN DELS EXPERIMENTS REALITZATS .....	48
IL·LUSTRACIÓ 35 GRÀFIC QUE MOSTRA L'EXHAUSTIVITAT OBTINGUDA A CADA CLASSE EN CADASCUN DELS EXPERIMENTS REALITZATS .....	49
IL·LUSTRACIÓ 36 GRÀFIC DE F1-SCORE DE CADA CLASSE PER CADA MODEL.....	50
IL·LUSTRACIÓ 37 MITGES MACRO I WEIGHTED DE L'EXHAUSTIVITAT LA PRECISSIÓ I F1.....	52

### **Llista de Codi**

CODI 1 PERCENTATGES DE DIVISIÓ DE LES IMATGES .....	25
CODI 2 CONFIGURACIÓ DE LES CAPES DEL MODEL VGG 16 .....	45

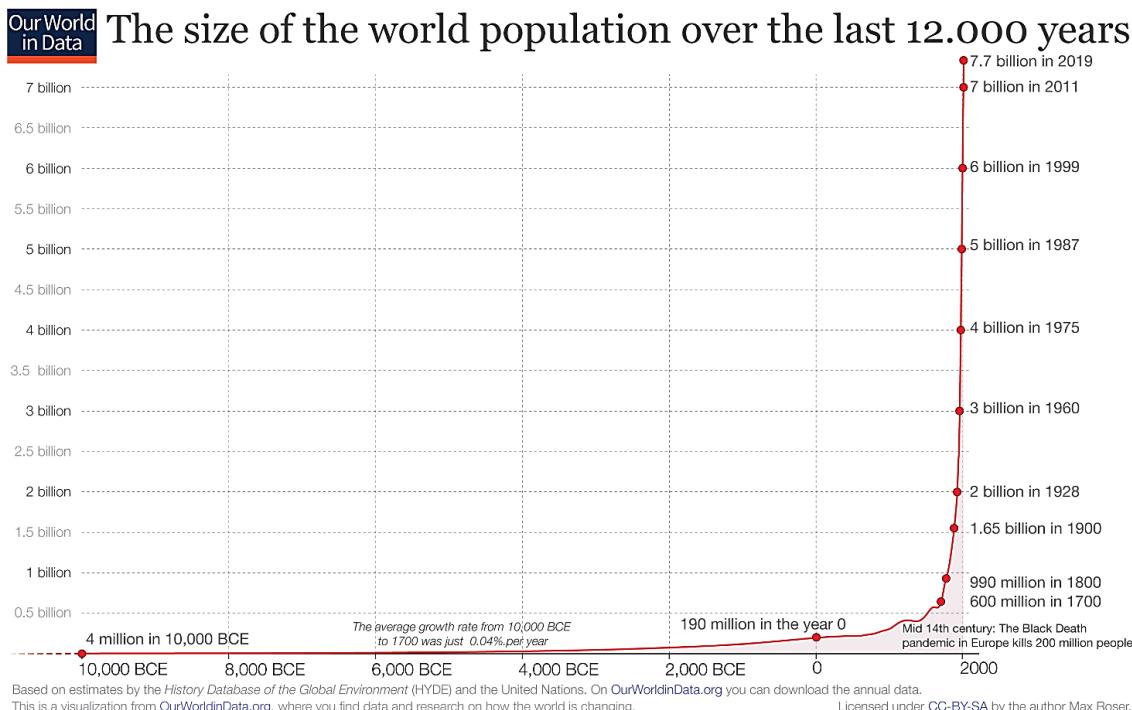
## Llista de taules

TAULA 1 TAULA DE PLANIFICACIÓ .....	5
TAULA 2 GRÀFIQUES I FUNCIONS MATEMÀTIQUES DE LES FUNCIONS D'ACTIVACIÓ (SAGAR SHARMA, 2017) .....	14
TAULA 3 QUANTITAT D'IMATGES PER CLASSE UTILITZADES EN L'ESTUDI.....	19
TAULA 4 TOTAL D'IMATGES ELIMINADES PER CLASSE.....	21
TAULA 5 DISTRIBUCIÓ DE LES IMATGES .....	26
TAULA 6 EXEMPLE DE MATRIU DE CONFUSIÓ.....	28
TAULA 7 EXEMPLE DE TAULA DE RESULTATS DE PRECISIÓ, EXHAUSTIVITAT I F1 I LES MITGES FINALS .....	29
TAULA 8 EXEMPLE DE TAULA DEL SUMATORI DELS ERRORS ACUMULATS EN TOTS ELS EXPERIMENTS .....	29
TAULA 9 TAULA DE PRECISION, RECALL I F1 DEL MODEL BASE.....	32
TAULA 10 TAULA DE PRECISION, RECALL I F1 DEL MODEL AMB DADES AUGMENTADES .....	35
TAULA 11 TAULA DE PRECISION, RECALL I F1 DEL MODEL AMB REGULARITZACIÓ L2 .....	38
TAULA 12 TAULA DE PRECISION, RECALL I F1 DEL MODEL AMB BATCH NORMALIZATION .....	41
TAULA 13 TAULA DE PRECISION, RECALL I F1 DEL MODEL AMB BATCH NORMALIZATION I DROPOUT .....	44
TAULA 14 TAULA DE PRECISION, RECALL I F1 DEL MODEL VGG 16 .....	47
TAULA 15 DESVIACIÓ ACUMULADA ENTRE TOTS ELS MODELS PRESENTATS SEGONS CADA CLASSE .....	51

# 1. Introducció

## 1.1 - Context i justificació del Treball

L'agricultura és un dels pilars de la supervivència humana ja que depenem d'aquesta matèria prima per poder tenir una alimentació saludable. En els últims segles l'explosió demogràfica a nivell mundial fa que ens trobem davant d'un planeta amb una quantitat de població com no s'havia vist. Es calcula que l'any 0 hi havia uns 190 milions de persones al planeta i al 1900 la població era de 1650 milions. Del 1900 a l'actualitat la població mundial ha passat de 1650 milions a 7700 milions, una diferència de 6050 milions en 119 anys provocat per la millora constant de les condicions de vida (il·lustració 1).



Il·lustració 1 Evolució de la població mundial (Roser et al., 2020)

Actualment el ritme de creixement s'està frenant i les previsions són que el percentatge d'increment interanual es redueixi a 0 cap a finals del present segle amb una població estimada d'uns 12 bilions.

En el seu estudi Agrios ja va repassar que l'home ha patit malalties en les plantes des d'èpoques remotes. Fet que queda constatat en l'antic testament, en el qual els míldius es mencionen junt amb la guerra i les malalties humanes, com les grans calamitats dels pobles. Altrament el gran filòsof grec Teofrastro (370-286 a.C.), realment va ser el primer en estudiar i escriure sobre les malalties en arbres, cereals i lleguminoses. (Agrios, 2004)

Actualment la població mundial consumeix milions de tones de menjar cada any. Per tal de suprir aquesta demanda i la que s'anirà generant amb l'augment de població en els pròxims anys, és important poder maximitzar la productivitat de forma eficient.

Buscar solucions per combatre plagues i malalties que puguin donar-se en els cultius per tal de prevenir-los o detectar-los en fases inicials ha de ser un objectiu necessari en les properes dècades. Per això, serà important disposar d'unes de reconeixement de malalties en els nostres cultius i que aquestes siguin de confiança.

Actualment, el procediment de detecció de malalties o plagues en els cultius el fan els propis agricultors, biòlegs, enginyers agrícoles, etc. en base a la seva experiència. Donat que en els últims anys hi ha hagut un gran augment de la capacitat de computació, l'evolució en el camp de la IA s'ha accelerat. Es poden trobar estudis dins l'àmbit de l'agricultura intel·ligent que aprofiten les possibilitats que ofereix l'aprenentatge computacional i l'aprenentatge profund en aquest àmbit. Hi ha estudis per millorar l'eficiència en les irrigacions a partir de les mesures de la humitat del sòl (Goldstein et al., 2018) o sistemes més complexos que afegeixen dades meteorològiques històriques, mesures de pressió atmosfèrica, etc. perquè el sistema sigui capaç de predir el clima (Choudhary et al., 2019).

La literatura reflexa l'interès també en el camp del reconeixement i la detecció de malalties en plantes. Es poden trobar aproximacions on s'ha utilitzat l'algoritme KNN (K-nearest neighbours) i SVM (support vector Machine) com (Kaushal & Bala, 2017) amb valors de precisió entre el 80 i el 90% i (Shivali Sharma et al., 2018) amb una precisió aconseguida del 98%. Les tècniques de transferència de coneixement (*transfer learning*), basades en la utilització de xarxes neurals convolucionals pre-entrenades han estat aplicades en diversos estudis com per exemple (Jadhav et al., 2020). L'estudi es centra en la utilització de les xarxes AlexNet i GoogLeNet utilitzant tècniques de fine tuning per a millorar els resultats i aconsegueix unes precisions del 98.75% amb AlexNet i del 96.25% amb GoogLeNet.

Una de les motivacions del treball és demostrar que es podrien arribar a desenvolupar xarxes neurals convolucionals (*convolutional neural networks, CNN*) que es podrien incorporar posteriorment a petites aplicacions. Aquestes podrien ser utilitzades, per exemple, per a petites i mitjanes empreses agrícoles que no disposin d'un gran poder adquisitiu per obtenir costoses aplicacions. Això els facilitaria el control de plagues dels seus cultius utilitzant el reconeixement d'imatges en fases inicials de la malaltia.

Una de les principals dificultats en el reconeixement de les malalties en les plantes passa pel propi reconeixement de l'espècie de la planta. Aquest punt es considera un tema complicat encara en l'actualitat ja que en la natura hi ha una quantitat molt gran d'espècies i moltes tenen similitud en la forma i color de les seves fulles.

El present estudi explora solucions al problema de la classificació de les espècies i les malalties que aquestes poden tenir, el model hauria de ser capaç de predir quina malaltia en concret té l'espècimen o bé si aquest està sa. L'estudi es duu a terme amb la creació d'una CNN des de zero aplicant diferents proves i comparant els resultats obtinguts. També es duu a terme una prova de *tranfer learning* amb el model pre-entrenat VGG16. L'estudi i la comparativa desenvolupada es realitza fent servir el conjunt de dades conegut com a PlantVillage.

## 1.2 - Objectius del Treball

### 1.2.1 – Generals

- Analitzar dues solucions basades en aprenentatge profund (*deep learning*) per a la classificació de la espècie a la que pertany la planta i si aquesta està sana o presenta alguna malaltia en concret, en funció de la imatge de la fulla. La primera solució és la de generar un model de CNN aplicant diferents proves i comparar-ne els resultats. La segona solució és utilitzar un model pre-entrenat conegut com a VGG16.
- Explorar la viabilitat d'obtenir una confiança mínima del 80%.
- Explorar els temps de computació utilitzant un computador de sobretaula d'ús quotidià a la hora d'entrenar un model de xarxa neural convolucionarial.

### 1.2.2 – Específics

- Revisar la literatura sobre l'aprenentatge profund.
- Revisar la literatura sobre les xarxes neurals convolucionals.
- Fer un anàlisi detallat del conjunt d'imatges.
- Separar el conjunt d'imatges en els subconjunts necessaris.
- Eliminar del conjunt les imatges no aptes.
- Estudiar les versions de Python, Tensorflow i Keras i la compatibilitat entre elles.
- Estudiar les possibles necessitats addicionals per a poder utilitzar la GPU del computador per a realitzar els càlculs.
- Estudiar les llibreries de Tensorflow i Keras consultant la documentació oficial.
- Programar la CNN de partida i entrenar-la per a l'estudi (el model bàsic).
- Explorar modificacions del model base afegint i modificant diferents eines i tècniques habituals (augment de dades, Dropout, etc.).
- Explorar el model pre-entrenat VGG16.
- Revisar, estudiar i comparar tots els resultats obtinguts per discutir-los i extreure conclusions.

## 1.3 - Enfocament i mètode seguit

S'ha dut a terme la realització d'una sèrie de models de xarxes neurals convolucionals capaces de fer les prediccions exposades. Els models creats s'han fet en base a models habituals dins de l'àmbit del reconeixement d'imatges intentant adaptar-los el millor possible al cas d'estudi.

Per desenvolupar els experiments de les CNN s'ha seleccionat Tensorflow i Keras. Tensorflow és una biblioteca de software de codi obert desenvolupada per Google per al aprenentatge automàtic que utilitza gràfics de flux de dades. En aquests gràfics els nodes representen les operacions matemàtiques que s'executen i els arcs representen els tensors (*arrays multidimensionals de dades*). La seva arquitectura facilita l'ús de diverses CPU o GPU per

realitzar càlculs en paral·lel. Per la seva banda Keras també és una llibreria d'alt nivell basada en Python i de codi obert. Pot ser utilitzada sobre Tensorflow, Microsoft Cognitive Toolkit i Theano i està pensada per agilitzar el procés d'experimentació i entrenament de xarxes neuronals. Ens aporta implementacions com les capes o *layers*, funcions d'activació, optimitzadors, etc. Python 3 és el llenguatge de programació sobre el qual es desenvolupa el treball.

També s'ha dut a terme una prova de transfer learning utilitzant un model pre-entrenat conegut com VGG16. Aquest model pre-entrenat, com d'altres dins d'aquest camp, ha estat validat i ha demostrat ser útils per a multitud classificacions diferents, així doncs, s'explora la viabilitat del seu ús en el marc del problema que es vol solucionar en el present estudi.

L'estudi s'ha dut a terme a una màquina local amb les següents característiques:

- 8Gb RAM a 3466MHz
- CPU: AMD Ryzen 5 2600X a 4.2GHz
- GPU: Gigabyte GTX 1070 Ti 8Gb GDDR5

El programa utilitzat per al desenvolupament ha sigut el PyCharm per ser un dels entorns de desenvolupament integrat (IDE) de python més utilitzats i potent.

## 1.4 - Planificació del Treball

A la Taula 1 es presenta la planificació detallada del temps a destinat a cada tasca del projecte:

Procés	Data inici	Dies	Data fi
<b>PAC 1 Pla de treball</b>	<b>3-març</b>	<b>13</b>	<b>16-març</b>
<b>PAC 2 Treball fase 1</b>	<b>17-març</b>	<b>34</b>	<b>20-abr</b>
Instal·lació del programari necessari	17-març	2	19-març
Investigació de què són i com funcionen les CNN	20-març	3	23-març
Aprendentatge sobre tensorflow	24-març	15	8-abr
Revisió de les imatges i preparació	6-abr	4	10-abr
Construcció de la CNN base	10-abr	8	18-abr
Obtenció dels primers resultats	18-abr	3	21-abr
<b>PAC 3 Treball fase 2</b>	<b>21-abr</b>	<b>27</b>	<b>18-mai</b>
Preparació del model amb dades augmentades	21-abr	2	23-abr
Preparació del model amb Batch Normalization	23-abr	2	25-abr
Preparació del model amb regularització L2	25-abr	3	28-abr
Preparació del model amb Dropout	28-abr	3	1-mai
Prova amb transfer learning VGG16	2-mai	4	6-mai
Estudi dels resultats	6-mai	8	14-mai
Comparació entre transfer learning i resultats propis	14-mai	1	15-mai
Extracció de conclusions	15-mai	4	19-mai
Revisió i avanç en la memòria	30-abr	19	19-mai
<b>PAC 4 Redacció memòria</b>	<b>19-mai</b>	<b>15</b>	<b>3-juny</b>
Finalització de la memòria	19-mai	8	27-mai
Revisió general de la memòria	27-mai	4	31-mai
Refinament de la memòria	31-mai	3	3-juny
<b>PAC 5a Elaboració presentació</b>	<b>4-juny</b>	<b>7</b>	<b>11-juny</b>
<b>PAC 5b Defensa pública</b>	<b>15-juny</b>	<b>9</b>	<b>24-juny</b>

Taula 1 Taula de planificació

I en la il·lustració 2 es pot veure en format de gràfic de Gantt:



Il·lustració 2 Gràfic de Gantt de la planificació del treball

## 1.5 - Breu sumari de productes obtinguts

A la finalització del treball s'espera entregar:

- La memòria amb tots els detalls de la investigació i el codi creat per a la preparació de les dades i les xarxes neurals convolucionals.
- Una presentació de l'estudi.

## 1.6 – Estructura

Aquest treball s'ha estructurat en els següents capítols:

El capítol 1 serveix de presentació del treball, s'exposen les motivacions que han conduit a la realització d'aquest treball, quins objectius es perseguien i com s'ha organitzat el temps durant la realització. A més a més, es fa una presentació dels resultats i es mostra com s'estructura el treball.

En el capítol 2 s'explica el funcionament intern de les xarxes neurals convolucionals. Començant per l'estructura general de les capes per entrar posteriorment en més detall en aquestes i entendre quin tractament es dóna a les dades en cada pas.

En el capítol 3 es presenta el conjunt de dades utilitzat, quin és el seu origen, perquè s'han escollit unes determinades imatges, quin és el seu format i quina distribució tenen aquestes en el conjunt. A part també es pot veure una mostra de com són.

El capítol 4 exposa la metodologia aplicada en el treball, el preprocessament de les dades, s'exposa també el conjunt de dades, l'arquitectura dels models utilitzats, com s'ha augmentat les dades i altres conceptes importants i finalment les mètriques utilitzades en l'estudi.

El capítol 5 exposa els resultats dels diferents experiments realitzats amb les CNN i els resultats amb la prova de *transfer learning*. Al final del capítol es fa la discussió de tots els resultats obtinguts.

El capítol 6 exposa les conclusions a les que s'ha arribat després d'haver realitzat tot l'estudi.

## 2. Les xarxes neurals convolucionals

### 2.1 – Introducció al aprenentatge computacional

Des dels anys 50, un subconjunt de la intel·ligència artificial (AI), conegut com a aprenentatge computacional (*Machine Learning*, ML), ha revolucionat multitud de camps en les últimes dècades. L'aprenentatge automàtic fa referència a la capacitat d'un computador per aprendre mitjançant l'adaptació d'algoritmes a certes dades d'entrada en el sistema. Hi ha 4 tècniques d'aprenentatge computacional: supervisat, no supervisat, semi-supervisat i per reforç.

L'aprenentatge supervisat és el que utilitza unes dades inicials, de les quals se'n té el valor correcte que se n'espera, per aprendre d'elles i que permet a l'algoritme aprendre per poder fer prediccions de dades noves. És utilitzat habitualment en problemes de classificació i de regressió.

L'aprenentatge no supervisat no conté el valor correcte que s'esperaria per un conjunt de dades d'entada. L'aprenentatge es basa en l'exploració de les dades per intentar cercar algun patró en elles. Els algoritmes de clustering per buscar grups de dades similars són uns dels més coneguts.

L'aprenentatge semi-supervisat combina elements dels dos anteriors per intentar millorar els resultats obtinguts.

En l'aprenentatge per reforç hi ha la presència d'un agent, que anirà indicant al sistema si la predicció realitzada és o no correcte. Quan la predicció és correcte el sistema té un comportament similar al aprenentatge supervisat, quan no és correcte, s'indica al sistema la quantitat d'error comès per tal que pugui corregir-lo.

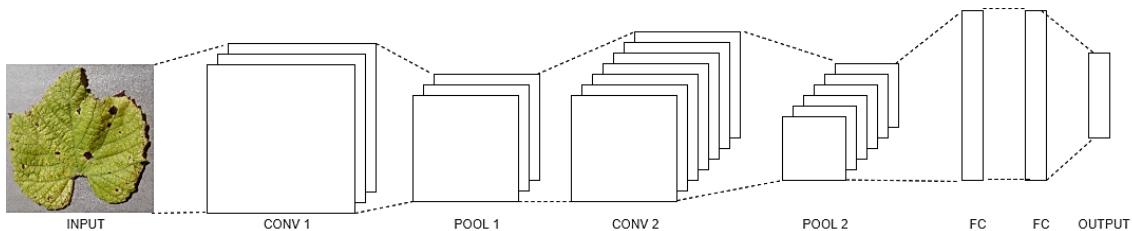
### 2.2 – Les xarxes neurals convolucionals

#### 2.2.1 – Introducció

Les xarxes neurals (*Neural Networks*) són un subcamp del ML i va ser en aquest subcamp el que va generar el *Deep Learning* (DL) (Alom et al., 2018). Dins del DL podem trobar també tècniques d'aprenentatge supervisat, semi-supervisat i no supervisat, dins de les tècniques d'aprenentatge supervisat és on s'ubiquen les xarxes neurals convolucionals (CNN). El primer en proposar l'estruatura de les xarxes neurals convolucionals va ser Fukushima l'any 1988 (Fukushima, 1988).

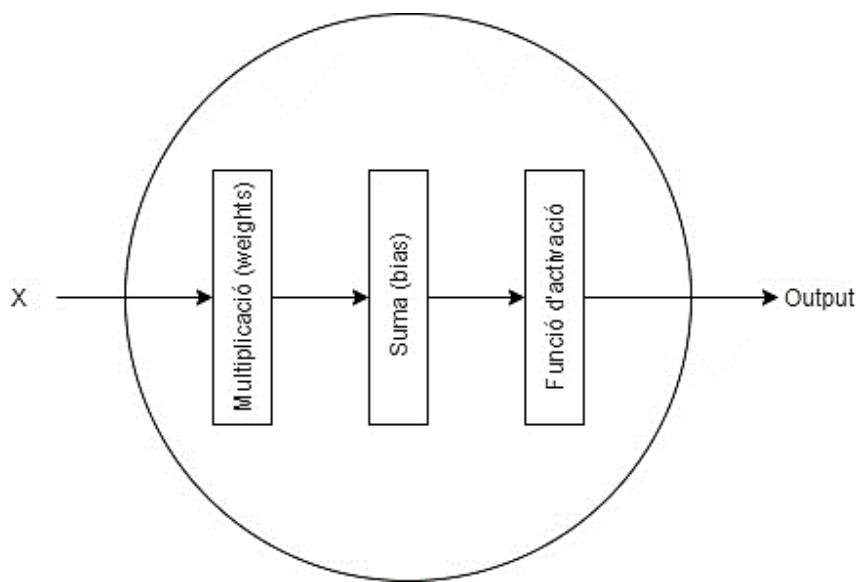
“Les xarxes neurals convolucionals són àmpliament conegudes per ser inspirades en el còrtex visual [...] la informació visual arriba per l'ull i viatja a través d'una sèrie d'estructures cerebrals i arriba al còrtex visual.” (Wang & Raj, 2017). L'àrea del cervell que rep la informació es coneix com a V1 i a questa es va processant i viatjant a diferents àrees del còrtex visual, es comença reconeixent les formes més bàsiques i en cada àrea s'extreu informació més complexa, finalment s'és capaç de reconèixer l'objecte que s'està veient.

Basant-se en aquesta idea, l'arquitectura de les CNN es divideix en 3 segments bàsics: una capa d'entrada o input (que equivaldria a la retina), un conjunt de capes ocultes (serien les conv, pool més les *fully connected* (FC) i que equivaldrien a les àrees on es processa la informació) i la capa de sortida final o output (equivaldria al reconeixement de l'objecte). En la il·lustració 3 es pot veure un exemple de l'arquitectura bàsica de les CNN.



Il·lustració 3 Arquitectura típica d'una xarxa neural convolucional

Cada una d'aquestes capes està formada per un conjunt de neurones sobre les quals es fan els càlculs segons les funcions utilitzades. El procés que es dóna en una neurona segueix l'estruatura mostrada a la il·lustració 4. "Cada neurona rep múltiples entrades, agafa una suma ponderada sobre elles, ho passa a través d'una funció d'activació i respon amb una sortida" (Pokharna, 2016)



Il·lustració 4 Representació d'una neurona durant una convolució

### 2.2.2 – Preprocessament de les imatges

Abans d'aplicar la convolució a la imatge és necessari redimensionar-la a una mida òptima i normalitzar els valors entre 0 i 1. Es pot prendre d'exemple la il·lustració 5 que representa la imatge d'una fulla de 10 per 10 píxels en blanc i negre on s'han normalitzat els valors, aquesta imatge correspondria als valors d'entrada de la CNN. En el cas de les imatges a color, l'entrada estaria formada per tres matrius de 10x10 píxels, una per cada canal de color RGB.

0	0	0	0	1	1	1	1	1	1
0	0	0	1	0	1	0	1	0	1
0	0	1	1	0	1	0	1	1	0
0	1	0	1	0	1	1	0	1	0
0	1	0	1	0	1	1	1	1	0
0	1	0	1	1	0	0	0	1	0
0	1	1	1	1	1	1	1	1	0
0	0	1	0	0	0	0	1	0	0
0	1	0	1	1	1	1	0	0	0
1	1	0	0	0	0	0	0	0	0

Il·lustració 5 Imatge de fulla en format matriu

### 2.2.3 – La convolució

Amb les dades d'entrada apunt es passa a les capes ocultes i es comença aplicant una convolució. En una convolució es poden aplicar variis filtres a elecció de l'usuari, aquests filtres van iterant sobre les dades d'entrada fins a recórrer-les totes i obtenir totes les dades de sortida. En l'exemple de la il·lustració 6, per simplificar només s'utilitzarà un filtre. En verd podem veure la iteració inicial de les operacions que corresponda a:

$$0 * 1 + 0 * 0 + 0 * 1 + 0 * 0 + 0 * 1 + 0 * 0 + 0 * 1 + 0 * 0 + 1 * 1 = 1$$

I en taronja podem veure un exemple d'una altre iteració per obtenir un altre dels valors de sortida que correspon a:

$$1 * 1 + 0 * 0 + 1 * 1 + 1 * 0 + 1 * 1 + 0 * 0 + 1 * 1 + 1 * 0 + 1 * 1 = 5$$

A la realitat habitualment es comença aplicant 32 filters i es va doblant aquesta quantitat fins on decideix l'usuari, és una bona mesura inicial ja que és una estructura freqüent, però no és obligatòria.

0	0	0	0	1	1	1	1	1	1
0	0	0	1	0	1	0	1	0	1
0	0	1	1	0	1	0	1	1	0
0	1	0	1	0	1	1	0	1	0
0	1	0	1	0	1	1	1	1	0
0	1	0	1	1	0	0	0	1	0
0	1	0	1	1	1	1	1	1	0
0	0	1	0	0	0	0	0	1	0
0	1	0	1	1	1	1	0	0	0
1	1	0	0	0	0	0	0	0	0

dades d'entrada

1	0	1
0	1	0
1	0	1

Filtre

1	1	3	3	3	4	4	3
0	4	1	4	2	3	3	3
2	3	2	4	2	5	3	3
1	4	2	3	3	2	4	1
2	4	3	5	3	4	4	3
2	3	3	1	2	2	2	2
1	5	3	4	4	3	4	1
3	1	2	1	1	2	0	0

mapa de característiques

Il·lustració 6 Exemple de l'operació de convolució

Aquests filters contenen els valors dels pesos que s'aplicaran i que es compartiran entre totes les neurones de la mateixa capa. Els pesos inicials els pot definir l'usuari o es poden generar aleatoriament i és la xarxa neural la que, en successives capes, s'encarrega de modificar-los, generant així uns filters diferents en base als resultats obtinguts per intentar millorar la predicción.

Habitualment trobem dos tipus de convolucions, la mostrada en la il·lustració 6 és la que no té padding i s'observa que la matriu resultant té unes dimensions d'una columna i una fila menys que la inicial. L'altre opció és afegir padding com s'observa en la il·lustració 7, això permet que la sortida obtinguda tingui les mateixes dimensions que l'entrada inicial.

0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	0
0	0	0	0	1	0	1	0	1	0	1
0	0	0	1	1	0	1	0	1	1	0
0	0	1	0	1	0	1	1	0	1	0
0	0	1	0	1	0	1	1	0	1	0
0	0	1	0	1	0	1	1	1	1	0
0	0	1	0	1	1	0	0	0	1	0
0	0	1	1	1	1	1	1	1	1	0
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	1	1	1	0	0	0	0
0	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

dades inicials amb padding

1	0	1
0	1	0
1	0	1

filtre

0	0	1	0	3	1	3	1	3
0	1	1	3	3	3	4	4	3
1	0	4	1	4	2	3	3	1
1	2	3	2	4	2	5	3	3
2	1	4	2	3	3	2	4	1
2	2	4	3	5	3	4	4	3
1	2	3	3	1	2	2	2	1
2	1	5	3	4	4	3	4	1
1	3	1	2	1	1	2	0	0
2	1	2	1	2	2	1	1	0

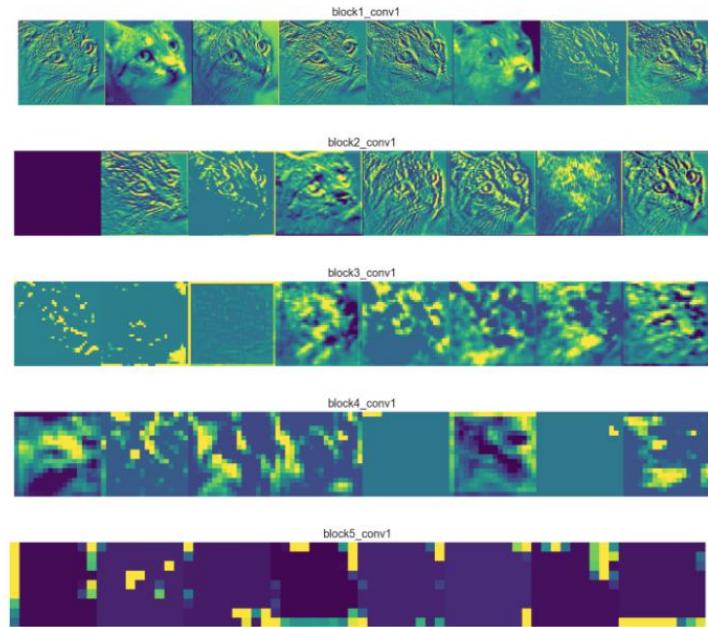
mapa de característiques

Il·lustració 7 Exemple de convolució amb padding

Les convolucions sense padding, al reduir les dimensions es solen utilitzar en substitució de les capes de pooling que es veuran més endavant.

A la hora de decidir com modificar els pesos, la CNN es basa en la funció d'optimització utilitzada per a l'entrenament ja que la seva funció és la de reduir al màxim la pèrdua. Hi ha varis optimitzadors àmpliament utilitzats: gradient descendent, regularització, gradient descendent estocàstic, Adam, etc.

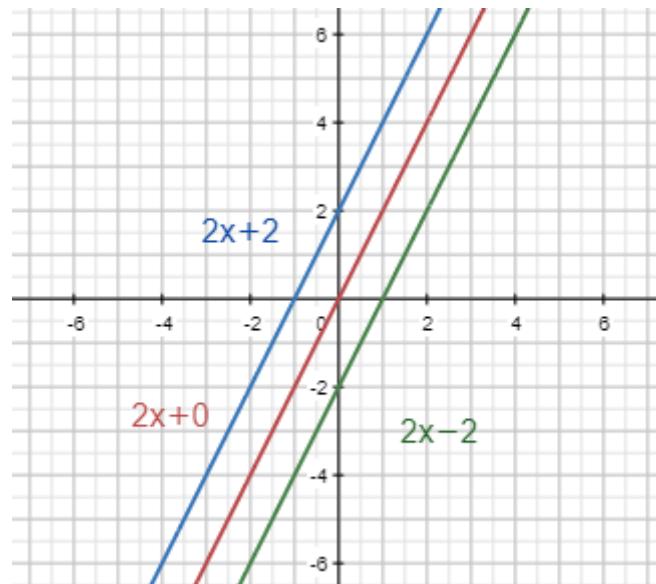
Això permet a la CNN aprendre diferents aspectes de les imatges. Com es pot veure a la il·lustració 8, en les primeres convolucions la xarxa es centra en reconèixer les formes més bàsiques i cada vegada s'especialitza més en detalls, aquestes imatges són els resultats de les convolucions i s'anomenen mapes de característiques.



Il·lustració 8 Mostra dels primers 8 mapes de característiques obtingudes en 5 convolucions amb el model de xarxa neural pre-entrenat VGG16, cada imatge és resultat d'un filtre diferent (Dertat, 2017)

## 2.2.4 – Bias

Un altre pas, com s'ha vist a la il·lustració 4, és la suma del bias, aquest s'utilitza per donar un grau addicional de llibertat a la xarxa permetent a la funció d'activació desplaçar-se a dreta o esquerra com es pot veure en la il·lustració 9. El bias sol ser el mateix per a totes les operacions en un filtre tot i que hi ha models que poden utilitzar per exemple un bias diferent per a cada pes del filtre.



Il·lustració 9 Exemple del funcionament del bias en una funció de l'estil  $f(x) = mx + c$  on  $c$  equival al bias

## 2.2.5 – Funció d'activació

L'últim pas abans d'arribar a la capa de pooling és aplicar la funció d'activació. N'hi ha de dos tipus, les linears i les no linears, aquestes últimes són les més utilitzades en general, alguns exemples són: Sigmoid, TanH, ReLU, softmax, etc. En general la seva funció és la de determinar el valor de sortida del model al mateix temps que pot ajudar-lo a aprendre dades més complexes. Una de les més utilitzades en les xarxes neurals convolucionals és la funció ReLU (Rectified Linear Unit). Com que els filters poden contenir valors negatius, ReLU s'encarrega de transformar els resultat negatius de les operacions a 0 de manera que el resultat final no contindrà valors per sota de zero.

A la taula 2 es pot veure la gràfica, la funció d'activació i la seva derivada, de diferents funcions d'activació.

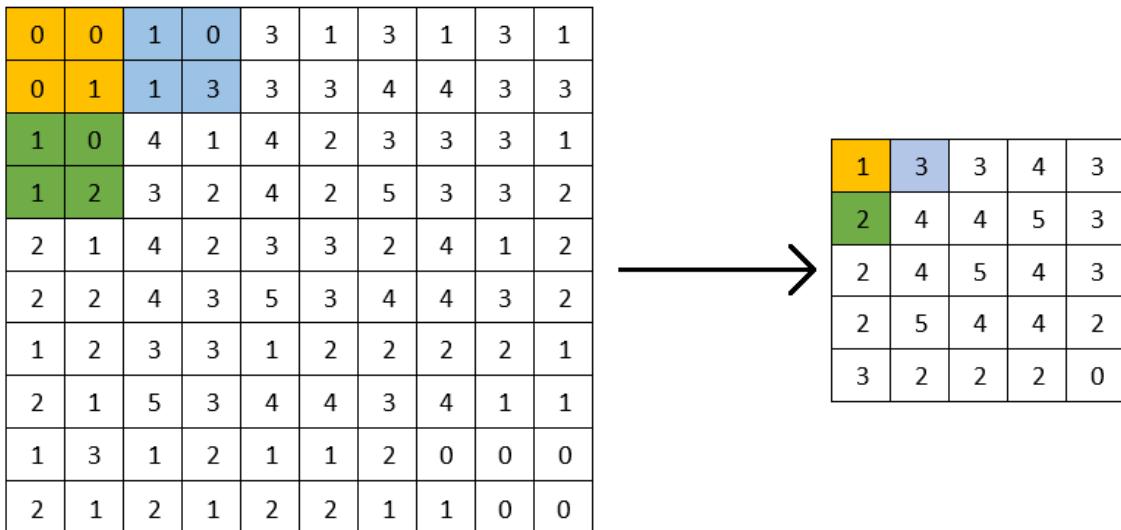
Name	Plot	Equation	Derivative
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (a. k. a Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2 + 1}$
Rectified Linear Unit (ReLU) [2]		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Parameteric Rectified Linear Unit (PReLU) [2]		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Exponential Linear Unit (ELU) [3]		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1 + e^x)$	$f'(x) = \frac{1}{1 + e^{-x}}$

Taula 2 Gràfiques i funcions matemàtiques de les funcions d'activació (Sagar Sharma, 2017)

## 2.2.6 – Pooling

Amb les dades de la capa de convolució i un cop aplicada la funció d'activació, les dades passen a la capa de pooling. Les capes de pooling s'utilitzen per reduir progressivament el nombre de paràmetres ja que es redueixen les dimensions de la matriu d'entrada i per tant també es redueix el temps de computació.

Es pot veure un exemple en la il·lustració 10 on partim del mapa de característiques obtingut a la il·lustració 7 com si ja s'hagués sumat el bias i aplicat la funció d'activació. S'aplica una capa de MaxPooling on a cada iteració el model es queda amb el valor més gran que troba. En l'exemple s'aplica un filtre de 2x2 i un desplaçament de 2. Aquestes característiques són a elecció del usuari, es poden aplicar filtre de 3x3, 4x4, etc. i el desplaçament pot ser de 1 o més tot i que és important que en els salts no quedin files o columnes on no s'appliqui.

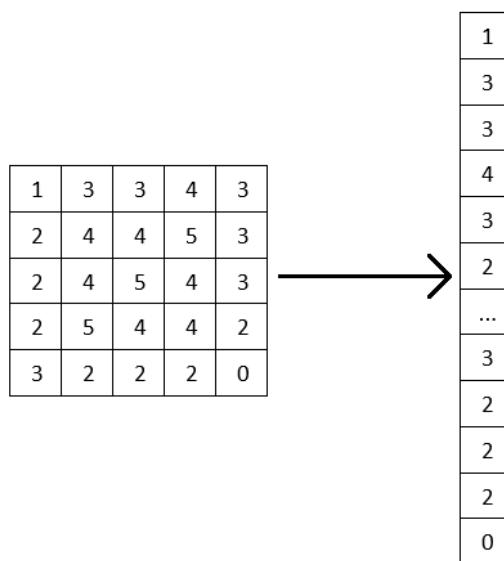


Il·lustració 10 Exemple de MaxPooling amb filtre de 2x2 i stride 2

Hi ha una altre versió de Pooling anomenada AveragePooling que en comptes de seleccionar el valor més gran fa una mitja entre els valors seleccionats pel filtre.

## 2.2.7 – Flatten

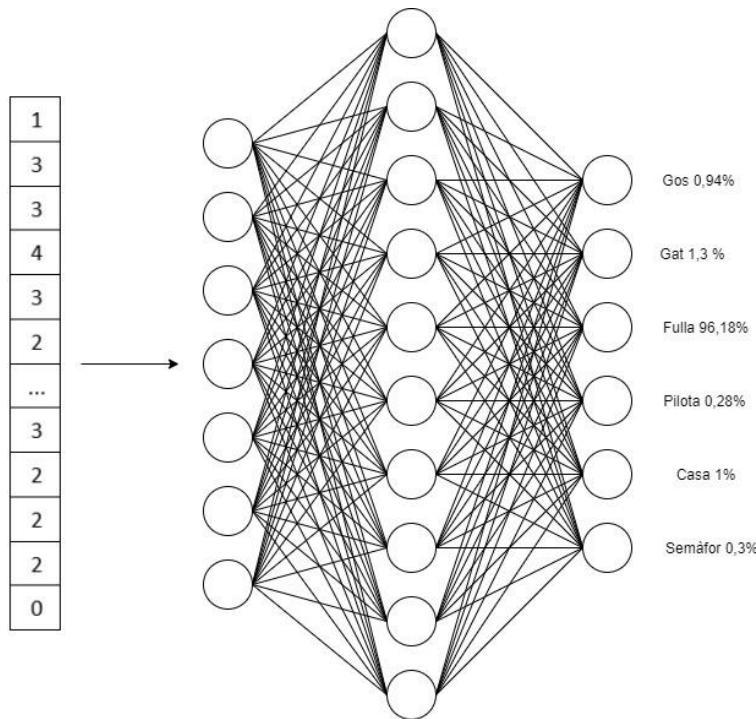
Finalitzades les capes de convolució i pooling s'aplica una funció de *flattening* per obtenir les dades de l'últim pool en una sola dimensió com es pot veure a la il·lustració 11.



Il·lustració 11 Procés de flattening

## 2.2.8 – Capes completament connectades (*fully Connected*)

Amb les dades obtingudes es passa a les capes *fully connected* que en realitat és una xarxa neural artificial clàssica que al final retorna les probabilitats que la imatge analitzada pertanyi a una classe com es pot veure a la il·lustració 12.



Il·lustració 12 Últimes capes de la xarxa neural convolucionarial

### 3. Conjunt de dades

PlantVillage és una unitat d'investigació i recerca de la Universitat de l'estat de Pensilvania que es va proposar com a objectiu donar poder als petits agricultors i treure'l's de la pobresa posant al seu abast noves tecnologies de forma barata i assequible i democratitzant l'accés al coneixement que pot ajudar-los a cultivar més aliments (*PlantVillage*, n.d.).

Aquesta unitat d'investigació va crear un conjunt d'imatges públic que conté fotografies de diferents espècies de cultius i que rep el mateix nom que l'equip (*PlantVillage*). Els cultius fotografiats estan diferenciats entre exemplars sans i exemplars malalts, dividint-los segons la malaltia en concret.

Inicialment el conjunt de fotografies estava format únicament per imatges a color, la versió utilitzada compartida per Ali Abdallah (Ali, 2019) conté, no obstant, les imatges a color, en blanc i negre i segmentades. En el cas que ens ocupa finalment només s'han utilitzat les imatges a color ja que de cara al objectiu que es persegueix estudiar, la viabilitat de la creació d'aplicacions reals, les imatges a color com les que pot fer qualsevol càmera són les més adequades. Les imatges estan disponibles al repositori Kaggle (*Kaggle: Your Home for Data Science*, n.d.).

Les fotografies majoritàriament mostren una única fulla sobre un fons homogeni tot i que segons la imatge pot canviar de tonalitat. Totes tenen unes dimensions de 256x256 píxels i estan en format JPG.

Hi ha 54.305 imatges a color dividides en 38 classes separades en carpetes que contenen imatges d'una espècie i si aquestes corresponen a exemplars sans o si tenen alguna malaltia. Entre aquestes classes trobem un total de 14 espècies diferents (pomera, nabiu, cirerer, blat de moro, raïm, taronger, presseguer, pebrer, patatera, gerd, soja, carbassó, maduixers i tomaqueres). En aquest estudi ens centrarem en la classificació de les classes de cirerer, raïm i tomaquera repartits en un total de 16 classes obtenint 24.096 imatges per al estudi.

S'han escollit aquestes espècies perquè la xarxa neural es planteja com una eina per a petits i mitjans agricultors, que solen estar especialitzats en cultius poc diversificats. Així doncs, no necessitaran eines capaces de reconèixer centenars d'espècies i malalties, sinó que reconeguin bé les que necessiten i diferencien correctament quina malaltia poden tenir. També és una necessitat de hardware ja que a més imatges, més paràmetres s'haurà de tenir en compte i més càlculs haurà de realitzar la CNN per entrenar. Això fa que requereixi més potència de computació si es vol treballar en rangs de temps d'entrenament acceptables. A més a més es corre el risc de no tenir suficient memòria provocant interrupcions en l'execució de l'entrenament que provoquin la pèrdua de dades.

A la il·lustració 13 es mostra exemples de les imatges seleccionades del conjunt de dades per a realitzar el treball.



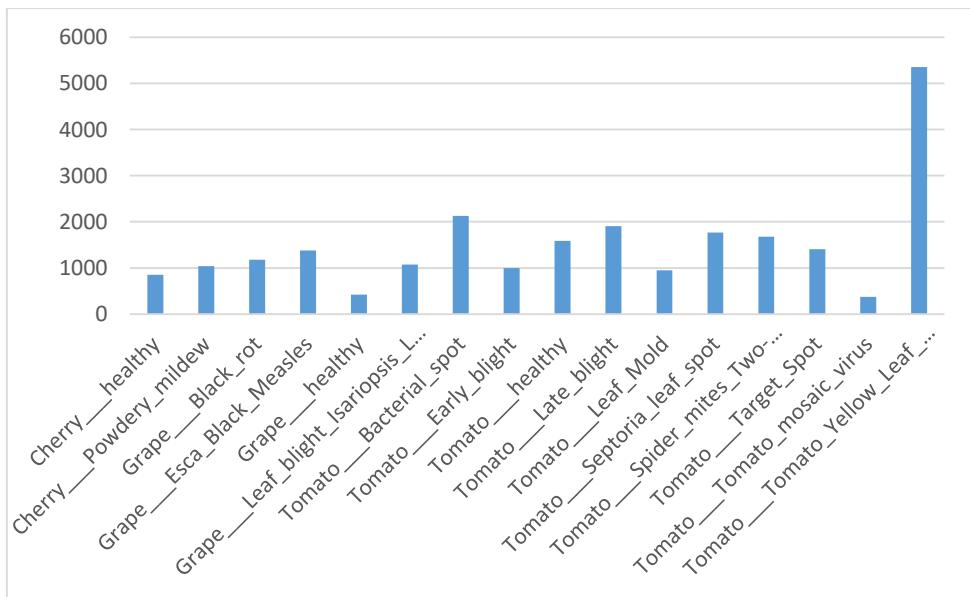
Il·lustració 13 Dataset PlantVillage exemples il·lustratius de diferents classes

A la taula 3 podem veure per una banda a la primera columna el nom de les 16 classes que haurà d'aprendre a classificar la CNN un cop entrenada, aquestes classes estan formades pel nom de l'espècie a la que pertany la imatge i la seva malaltia en cas que en tingui o si esta sana (*healthy*). Per altra banda, tenim la quantitat d'imatges corresponents a cada classe a la segona columna.

Classes	Quantitat d'imatges
Cherry__healthy	852
Cherry__Powdery_mildew	1040
Grape__Black_rot	1178
Grape__Esca_Black_Measles	1381
Grape__healthy	421
Grape__Leaf_blight_Isariopsis_Leaf_Spot	1075
Tomato__Bacterial_spot	2126
Tomato__Early_blight	1000
Tomato__healthy	1589
Tomato__Late_blight	1908
Tomato__Leaf_Mold	951
Tomato__Septoria_leaf_spot	1770
Tomato__Spider_mites_Two-spotted_spider_mite	1675
Tomato__Target_Spot	1403
Tomato__Tomato_mosaic_virus	371
Tomato__Tomato_Yellow_Leaf_Curl_Virus	5356

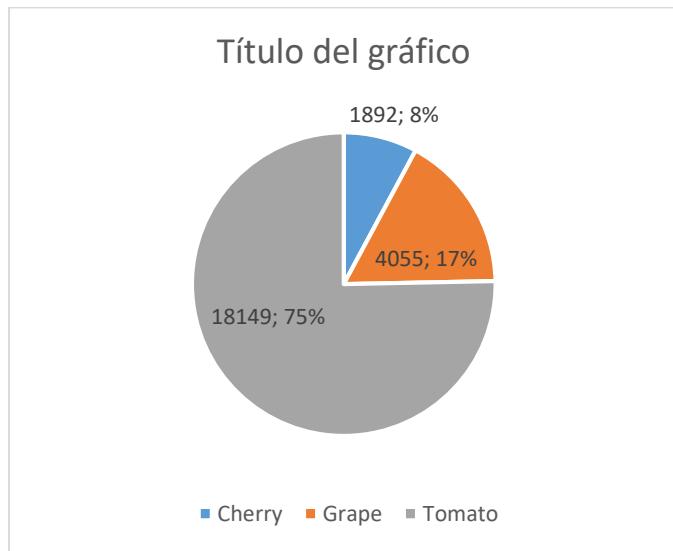
Taula 3 Quantitat d'imatges per classe utilitzades en l'estudi

A la il·lustració 14 podem veure una representació gràfica de la quantitat d'imatges de cada classe del grup d'entrenament. Aquest gràfic mostra de forma visual les dades de la taula 3 i ajuda a comprovar que la distribució del conjunt no està balancejada, per exemple *tomato yellow leaf curl virus* conté més del doble d'elements que la resta i *grape healthy* i *tomato mosaic virus* en contenen menys.



Il·lustració 14 Gràfic de la distribució de les imatges del conjunt d'entrenament

Si s'analitza per espècies, ens trobem una distribució del 8% per a Cirerers, 17% per raïm i 75% per tomaqueries com s'observa a la il·lustració 15.



Il·lustració 15 distribució per espècie en l'entrenament

Aquesta descompensació en les dades, tant des del punt de vista de classe com des del punt de vista d'espècie, s'haurà de tenir present a la hora d'analitzar els resultats i veure si tenen algun efecte en el resultat.

## 4. Metodologia

### 4.1 – Preprocessament de les dades

En primer lloc ha sigut necessari fer una revisió manual de les imatges ja que en algun cas no complien les característiques generals del conjunt. A la il·lustració 16 en podem veure alguns exemples. Observem que, en alguns casos, la imatge de la fulla, en comptes de tenir un fons homogeni es pot observar un paisatge. Un cas concret en comptes de mostrar una fulla d'una tomaquera s'ha fotografiat el tomàquet i altre cop sense el fons homogeni. Un altre cas particular és el que es pot observar en la primera imatge de mostra, aquesta està classificada com a un cirerer saludable i per la imatge no sembla ser un cirerer.



Il·lustració 16 Exemples d'imatges eliminades

El total d'imatges eliminades en cada classe del conjunt es pot veure a la taula 4:

Classes	Qtt
Cherry__healthy	1
Cherry__Powdery_mildew	10
Grape__Black_rot	0
Grape__Esca_Black_Measles	0
Grape__healthy	0
Grape__Leaf_blight_Isariopsis_Leaf_Spot	0
Tomato__Bacterial_spot	0
Tomato__Early_blight	0
Tomato__healthy	1
Tomato__Late_blight	0
Tomato__Leaf_Mold	0
Tomato__Septoria_leaf_spot	0
Tomato__Spider_mites_Two-spotted_spider_mite	0
Tomato__Target_Spot	0
Tomato__Tomato_mosaic_virus	0
Tomato__Tomato_Yellow_Leaf_Curl_Virus	0

Taula 4 Total d'imatges eliminades per classe

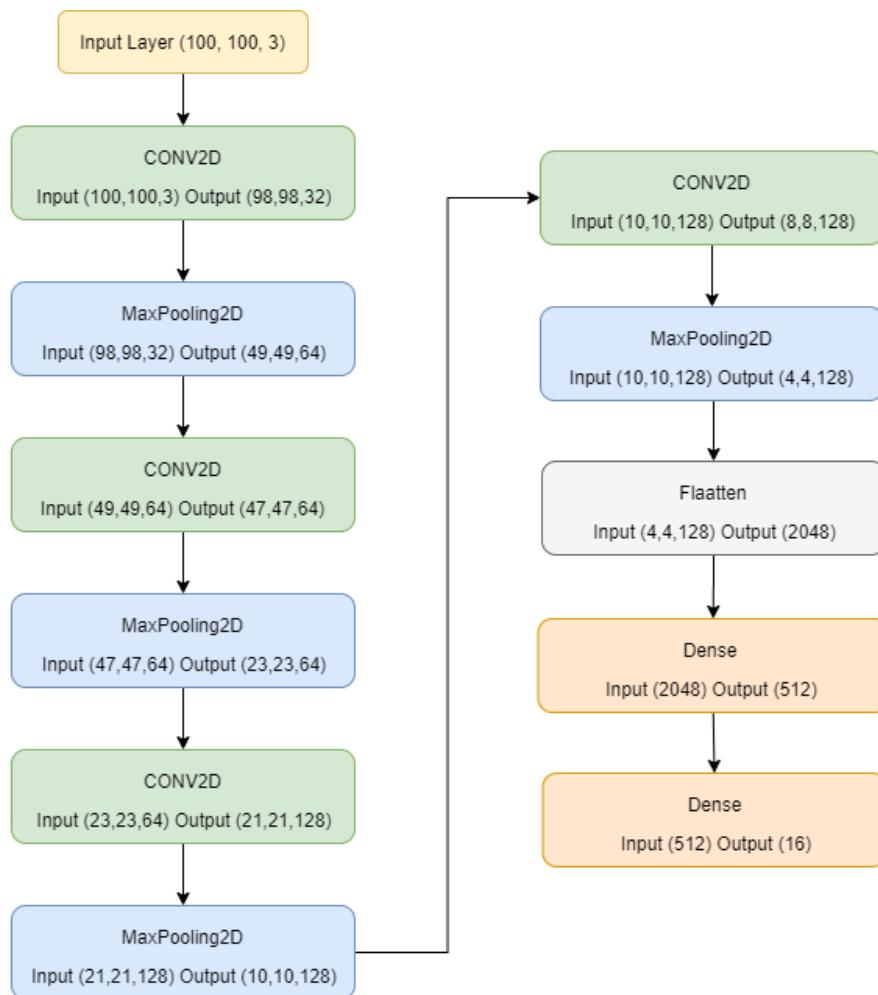
La classe més afectada és la de *cherry powdery mildew* amb 10 imatges de les 12 totals. Aquesta selecció no té una gran rellevància en l'estudi ja que a efectes pràctics, són tant pocs els canvis, que no haurien de tenir impacte en els resultats.

Les imatges utilitzades per a l'entrenament són normalitzades perquè els seus valors estiguin comprèsos entre 0 i 1 en comptes de 0 a 255. Aquestes també es redimensionen a una mida de 100 x 100 píxels per tal de reduir la càrrega al computador. Un cas diferent és la prova de transfer learning amb el model VGG16 que està preparat per imatges de 224 x 224 píxels i s'han deixat amb aquestes dimensions, per contra però, s'ha reduït el batch size a 32 per problemes amb la capacitat de computació.

## 4.2 – Arquitectura

### 4.2.1 – Model base

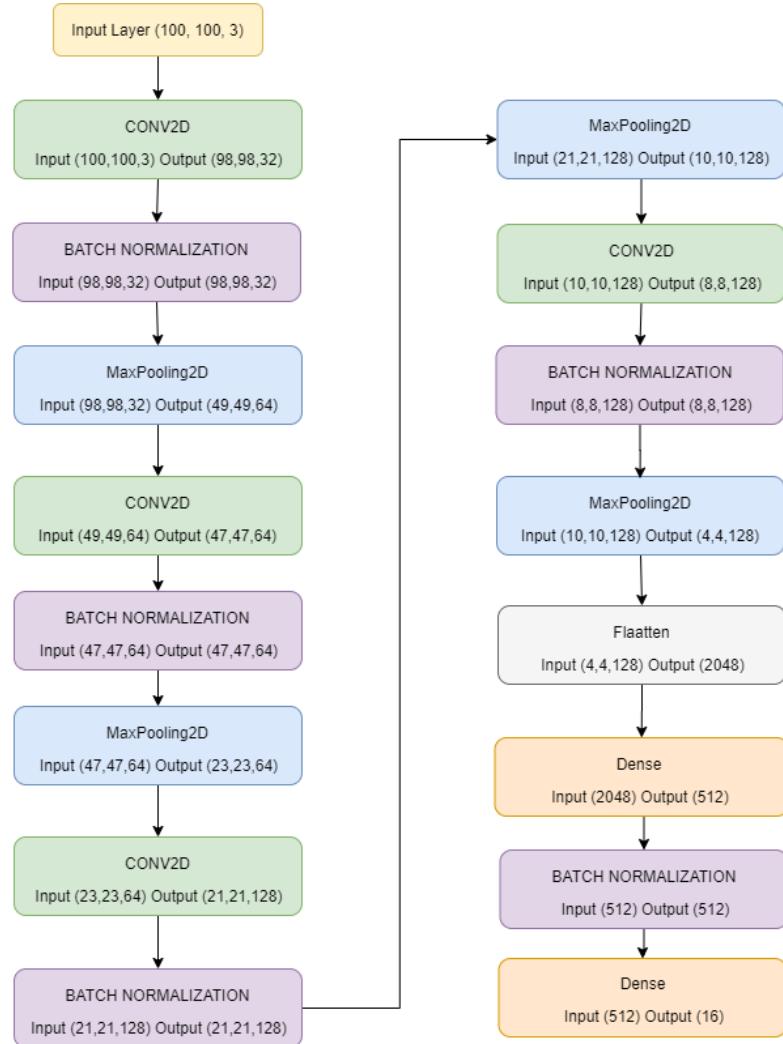
Com a punt de partida s'ha definit un model que serà la base de partida per els altres models. El diagrama del model es pot veure a la *il·lustració 17* i podem veure que està format per quatre capes de convolució 2D utilitzant la funció d'activació *relu* cadascuna d'elles seguida d'una capa de pooling, en aquest cas maxpooling2D. Al final tenim la capa Flatten per compactar els resultats en una matriu d'una dimensió seguida de dues capes Dense que conformen la xarxa neural artificial clàssica amb les capes completament connectades. Aquest model serà utilitzat en tres experiments, el base, el que utilitza dades augmentades i que incorpora regularització L2 en la capa de convolució.



*Il·lustració 17 Esquema de l'arquitectura del model base*

#### 4.2.2 – Model amb batch normalization

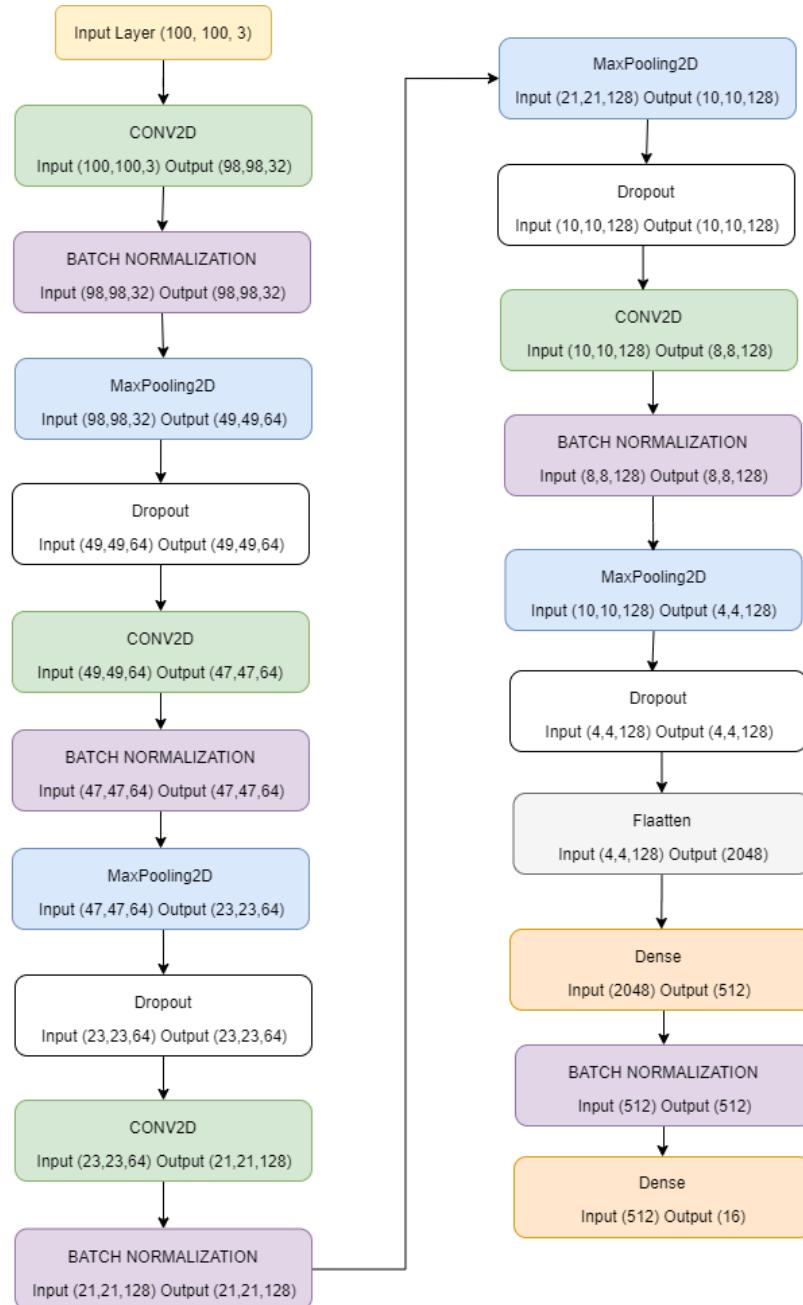
El model amb batch normalization afegeix una capa de batch normalization entre les capes de convolució i les de pooling com es pot veure al diagrama de la il·lustració 18.



Il·lustració 18 Esquema de l'arquitectura amb batch normalization entre les capes de convolució i pooling

#### 4.2.3 – Model amb dropout

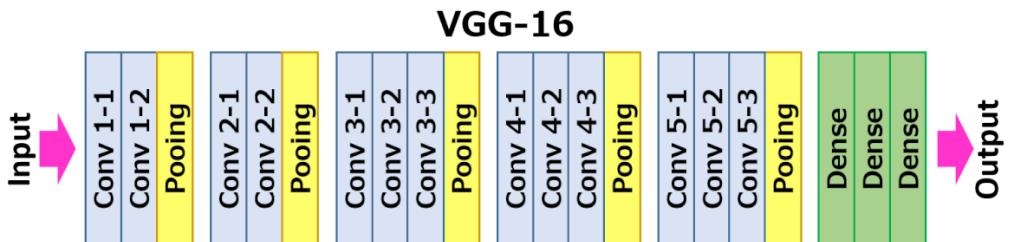
El model amb Dropout, a part del que ja s'ha mencionat, afegeix unes capes de dropout després de cada capa de pooling com es pot veure a la il·lustració 19.



Il·lustració 19 Esquema de l'arquitectura amb dropout

#### 4.2.4 – Model VGG16

El model VGG16 va ser el model guanyador del ImageNet Large Scale Visual Recognition Competition (ILSVRC) del 2014. Aquest model va passar a estar disponible online després de la competició amb els pesos de les seves capes disponibles per a poder ser utilitzat. La seva arquitectura es basa en fer 2 blocs utilitzant 2 capes de convolució i posteriorment una de pooling i després 3 blocs més amb 3 capes de convolució seguida d'una de pooling. Per acabar utilitza tres capes denses com es pot veure a la il·lustració 20.



Il·lustració 20 Arquitectura del model VGG16 (ul Hassan, 2018)

### 4.3 – Validació

#### 4.3.1 – Conjunts de dades

En el treball s'ha dividit la mostra en un 78% d'imatges d'entrenament, un 16% d'imatges de validació i un 6% de test com es pot veure al codi 1.

```
# Calculem el numero de imatges de cada tipus en % per cada cas
numTrain = int((len(x[2])*78)/100)
numValidation = int((len(x[2])*16)/100)
numTest = int((len(x[2])*6)/100)
```

Codi 1 Percentatges de divisió de les imatges

A la taula 5 tenim la quantitat d'imatges corresponents a cada classe segons si pertanyen al grup d'entrenament (train), validació (validation) o test i el percentatge que els correspon, aquest percentatge es manté igual en cada grup.

	TRAIN	VALIDATION	TEST	%
Classes	Qtt	Qtt	Qtt	
Cherry__healthy	665	136	51	3,54%
Cherry__Powdery_mildew	812	166	62	4,32%
Grape__Black_rot	920	188	70	4,89%
Grape__Esca_Black_Measles	1078	221	82	5,73%
Grape__healthy	329	67	25	1,75%
Grape__Leaf_blight_Isariopsis_Leaf_Spot	839	172	64	4,46%
Tomato__Bacterial_spot	1659	340	127	8,82%
Tomato__Early_blight	780	160	60	4,15%
Tomato__healthy	1240	254	95	6,59%
Tomato__Late_blight	1489	305	114	7,92%
Tomato__Leaf_Mold	742	152	57	3,95%
Tomato__Septoria_leaf_spot	1381	283	106	7,34%
Tomato__Spider_mites_Two-spotted_spider_mite	1307	268	100	6,95%
Tomato__Target_Spot	1095	224	84	5,82%
Tomato__Tomato_mosaic_virus	290	59	22	1,54%
Tomato__Tomato_Yellow_Leaf_Curl_Virus	4178	857	321	22,22%
<b>TOTAL</b>	<b>18804</b>	<b>3852</b>	<b>1440</b>	<b>100</b>

Taula 5 Distribució de les imatges

#### 4.3.2 – Augment de dades

L'augment de dades és un conjunt de transformacions que es realitzen a les imatges d'entrenament per a donar més variabilitat al sistema i que aquest sigui capaç de generalitzar millor. Aquestes són les transformacions que s'han fet a les imatges:

- Rotation angle: transforma la imatge rotant-la en un determinat angle, en aquest cas 40 graus. Útil per tal que la CNN sigui capaç de reconèixer una mateixa classe estigui orientada en qualsevol angle.
- Horizontal i vertical flip: Aplica un efecte mirall en qualsevol dels dos eixos a la imatge i serveix perquè la xarxa tingui més models per aprendre.
- Fill mode: Quan es fan les transformacions, es poden quedar zones sense imatge (fons negre), a vegades pot ser interessant, però keras et dóna com a solució aquesta opció. En aquest cas s'ha utilitzat nearest, que mira quin són els colors propers a la zona sense imatge i els extrapola creant un efecte similar al del propi fons de la imatge. Això és especialment útil quan s'utilitzen fons homogenis com és aquest cas.

Transformacions descartades:

- Height i width shift range: desplaça el centre de la imatge en l'eix vertical o l'horizontal. Útil per què la CNN sigui capaç de reconèixer les classes encara que l'element a reconèixer no estigui sempre centrat.
- Shear range: Provoca una distorsió en la imatge .
- Zoom range: Com el seu nom indica fa zoom positiu o negatiu a la imatge, és útil perquè la xarxa aprengui a distingir el mateix objecte tant si està agafat des d'un primer pla com a distància.

Després d'unes proves s'ha arribat a la conclusió que les transformacions descartades no aportaven dades útils al model. Els shift range vertical i horitzontal com el Zoom range, són diferències que ja tenim en el conjunt d'entrenament. No totes les fotografies estan fetes a la mateixa distància exacta ni totes les fulles estan completament centrades. A més a més, pensant en una hipotètica aplicació futura, si un agricultor vol saber quin problema té el seu cultiu, no té massa sentit pensar que pugui analitzar una foto on no es vegi completament bé la fulla pel que també s'ha descartat el shear range.

#### 4.3.3 – Conceptes

**Aprendentatge transferit:** Conegut pel terme anglès *transfer learning* és la tècnica que permet utilitzar xarxes neurals convolucionals prèviament entrenades amb els pesos entre les capes ja definits. Això es tradueix en una reducció del temps d'entrenament. Permet que es puguin utilitzar xarxes entrenades amb milions de paràmetres i capaces de reconèixer entre milers de classes sense entrenar tot aquest conjunt, només s'ha de fer que la xarxa aprengui a reconèixer les dades del estudi desitjat. A més a més, hi ha xarxes ja conegeudes com VGG16 o AlexNet que han demostrat en diversos estudis que poder ajustar-se bé a altres conjunts de dades.

**Refinament:** Conegut com a *fine tuning* és el procés d'explorar diferents valors de diferents paràmetres d'una xarxa neural convolucional. Per exemple, un valor important de les CNN és la taxa d'aprendentatge o *learning rate* i sol ser un dels valors a provar de modificar a veure si la CNN mostra una millora en les prediccions o no.

**Dropout:** És un mètode de regularització. “Durant l'entrenament, alguns dels valors de sortida de les capes són ignorats aleatoriament “drop out”. Això té l'efecte de fer que una capa sembli i es tracti com si fos una capa amb un nombre diferent de nodes i connexions amb la capa anterior. En efecte, cada actualització d'una capa durant l'entrenament es realitza amb una “vista” diferent de la capa configurada. El dropout té l'efecte de fer l'entrenament més sorollós, obligant als nodes dins d'una capa a assumir probabilísticament més o menys responsabilitats per els inputs (Brownlee, 2018).

**Batch normalization:** És una tècnica per normalitzar els valors després de cada convolució. Hi ha experiments que suggereixen que és millor realitzar la normalització després de la capa de pooling i d'altres que suggereixen que és millor abans depenent de les dades a tractar. En qualsevol cas s'ha provat com a efectiu a la hora d'entrenar xarxes neurals convolucionals.

#### 4.3.4 – Mètriques utilitzades

Durant l'entrenament podem veure en cada època com evolucionen els valors de *accuracy* i *loss*, aquests termes fan referència a:

**Accuracy:** la precisió és un mètode per mesurar el rendiment d'un model de classificació que se sol expressar en forma de percentatge. És el recompte de prediccions on el valor pronosticat és igual al valor real.

**Loss:** la pèrdua és una funció que té en compte la probabilitat o incertesa d'una predicció en funció de quina és la diferència entre el valor predit i el real. En aquest cas no és un percentatge sinó la suma dels errors cometuts per cada mostra del conjunt d'entrenament o validació. En xarxes neurals s'utilitza per trobar quins serien els millors pesos per al model.

En les proves de test es generen matrius de confusió de les quals es poden obtenir també els valors de recall, precision i F1-Score, un exemple bàsic de taula de confusió es pot veure a la taula 6.

N = 50	Predicció: SI	Predicció: NO	
Actual: SI	True Positive: 10	False Negative: 3	13
Actual: NO	False Positive: 7	True Negative: 30	37
17	33		

Taula 6 Exemple de matriu de confusió

**Recall:** és la mesura de l'exhaustivitat, ens informe de la quantitat que el model és capaç de predir correctament. Es calcula mitjançant el ràtio entre els elements classificats de forma correcte d'una determinada classe i el nombre total d'elements que haurien d'haver estat identificats però no ho han sigut.

$$R = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

**Precision:** ens permet mesurar la qualitat del model utilitzat. Es mesura mitjançant el ràtio entre els elements classificats de forma correcte d'una determinada classe i el total d'elements classificats d'aquesta classe, incloent els que no són realment.

$$P = \frac{\text{True positive}}{\text{True positive} + \text{False positive}}$$

**F1-Score:** és una mesura que combina la precisió i l'exhaustivitat per mesurar el rendiment del model. Es calcula fent la mitja harmònica entre la precisió i el recall:

$$F1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Junt amb la matriu de confusió també es genera una taula que mostra les mètriques comentades per cada classe, el seu suport (número d'elements de la classe) i al final es pot veure la mitja macro i la mitja tenint en compte el pes de la classe en el conjunt de la mostra per el recall, la precision i l'F1-score. Un exemple es pot veure a la taula 7.

	Precision	Recall	F1-score	Suppor
Class 0	1	1	1	1000
Class 1	0.02	0.02	0.02	5
Accuracy			1	1005
Macro avg	0.51	0.51	0.51	1005
Weighted avg	0.9951	0.9951	0.9951	1005

Taula 7 Exemple de taula de resultats de precisió, exhaustivitat i f1 i les mitges finals

**Weighted avg:** Com indica el seu nom és la mitja de tots els valors d'un tipus (recall, precision o F1) tenint en compte el percentatge que ocupa de la mostra cada classe. En l'exemple, el pes de la classe 0 en el conjunt de dades és del 99.5% ja que 1000 de les 1005 imatges en pertanyen. El pes de la classe 1 correspon al 0.5% restant. La formula en base al exemple i utilitzant precision seria:

$$weightedAvg = \frac{\% \text{ pes(class0)}}{100} * precision(class0) + \frac{\% \text{ pes(class1)}}{100} * precision(class1)$$

**Macro avg:** A diferència del anterior, la mitja macro dóna el mateix pes a cada classe. En l'exemple, com tenim dues classes, ambdues obtindrien un pes del 50%, de manera que la funció per calcular la mitja macro seria la mateixa que l'anterior canviant únicament el percentatge de pes de cada classe.

En conjunts de dades desbalancejats la mesura macro és la que té més penalització, per aquest motiu pot ser un bon indicador que el model no actua bé amb les classes minoritàries.

Per a obtenir una visió global dels errors, en la discussió final es mostra una taula que conté el sumatori dels errors generats en els diferents experiments. La taula 8 és un exemple de la matriu que ens mostra la suma d'errors fets entre tots els models. En color taronja estan representats els errors commesos entre elements de la mateixa espècie i en vermell errors entre espècies diferents.

Cherry A	X	2	4	5	8
Cherry B	1	X	7	4	6
Grape A	2	9	X	5	1
Grape B	5	2	3	X	2
Grape C	4	3	1	1	X
	Cherry A	Cherry B	Grape A	Grape B	Grape C

Taula 8 Exemple de taula del sumatori dels errors acumulats en tots els experiments

# 5. Experiments

## 5.1 – Model base

### 5.1.1 – Entrenament

Configuració inicial:

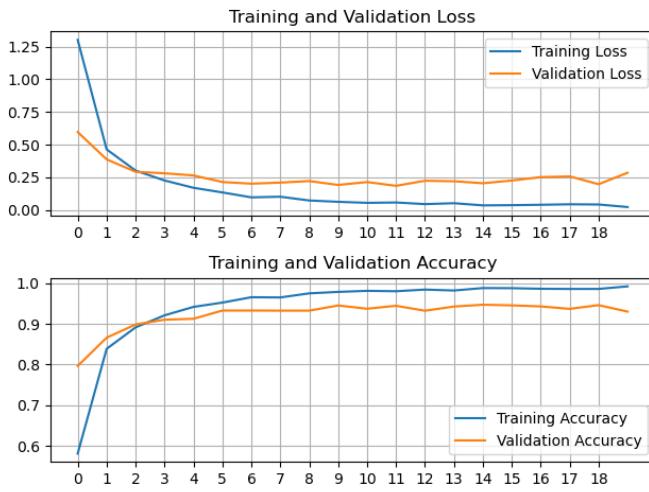
- Transfer Learning: no
- Image shape: 100x100 píxels
- Data augmentation: no
- Èpoques: 20
- Batch size: 64
- Pesos iniciais: aleatoris
- Funció d'optimització: Adam
- Learning rate: 0.001
- Loss function: categorical cross entropy
- Últimes capes: Dense(512, activation="relu"); Dense(16, activation= "none")

A la il·lustració 21 podem veure el comportament de la xarxa neural durant l'entrenament. Durant el transcurs d'aquest, els valors de Loss (gràfic superior) s'han anat reduint gradualment pel grup d'entrenament però no per el grup de validació que ja des de la cinquena època sembla no millorar. A més a més, ja en la tercera època, els valors de pèrdua de la validació són superiors als del entrenament. Això és una senyal que el model pot tendir al sobre acomodament.

L'excés d'acomodament es dona cada vegada que un algoritme s'adapta excessivament al conjunt de dades d'entrenament i llavors funciona malament amb el conjunt de dades de validació (Chicco, 2017). En el cas del reconeixement d'imatge això significa que la xarxa neural ha sigut capaç d'arribar a memoritzar les dades d'entrenament i per tant les prediccions d'aquestes dades tenen una precisió propera al 100%. No obstant, quan ha de classificar imatges noves no és capaç de generalitzar correctament i per tant la precisió és més baixa.

Si ens fixem en l'exactitud (gràfic inferior), el que es busca és que durant el transcurs del entrenament vagi augmentant de valor com passa amb les dades d'entrenament, no obstant altre cop, si ens fixem en les de validació veiem que a la tercera època ja s'obté un valor inferior al d'entrenament i més o menys a la època 5 queda estancat i no millora més.

Les dues evidències juntes reforcen la idea que el model està sobre acomodat.

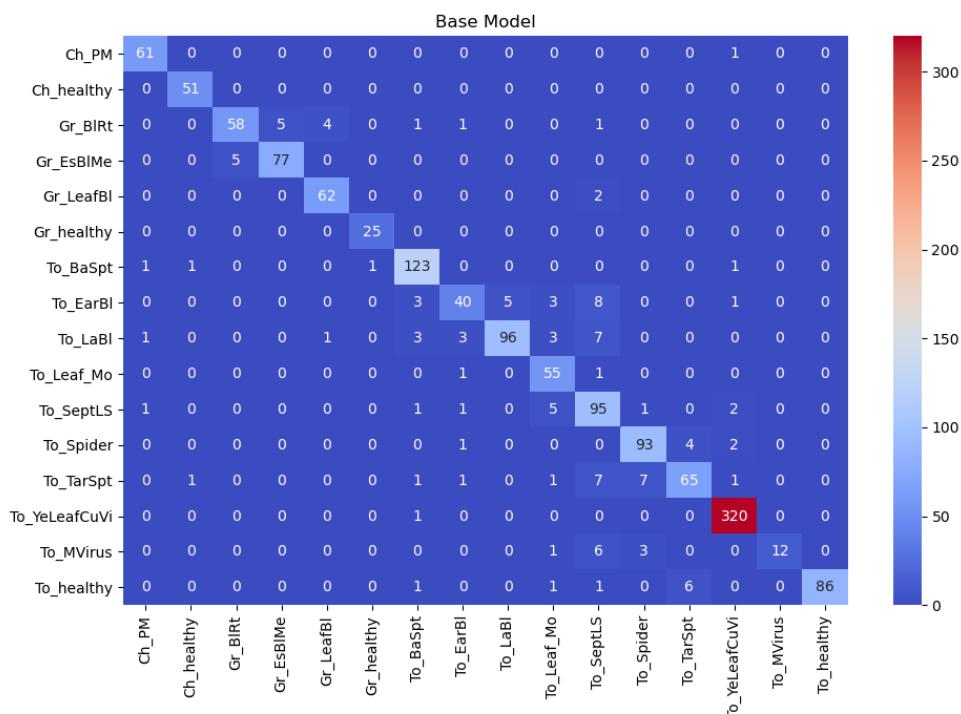


Il·lustració 21 Resultats d'exactitud i pèrdua durant l'entrenament del model base

### 5.1.2 – Resultats

Amb el model entrenat i amb la idea de que pot estar sobre acomodat, s'ha realitzat la validació amb el grup de test. Per a això, s'ha carregat el model base entrenat i s'ha fet que predis les imatges de test.

El resultat és la matriu de confusió de la il·lustració 22. En color blau més fosc trobem les zones amb menys dades i en vermell les zones on hi ha més dades. També es pot veure la diagonal ben diferenciada de la resta. Els valors que hi ha a la diagonal són els que s'han predit correctament i per tant, la idea del sobre acomodament sembla ara menys viable. Crida l'atenció les prediccions fetes a la classe *tomato septoria leaf spot* ja que s'observen 33 prediccions cap a aquesta classe que en realitat corresponen a altres, això li confereix una precisió del 74%.



Il·lustració 22 Matriu de confusió del model base generat amb les dades de test

Analitzant la taula 9 de resultats crida l'atenció algun valor que destaca per ser més baix que la resta, el primer d'ells és la classe *tomato early blight* que té un valor f1 del 0.74. Es veu que només el 83% dels casos predictius a aquesta classe ho són realment per tant tenim que el 17% s'hi ha classificat incorrectament. A més a més, de tots els que haurien d'haver estat classificats en aquesta classe, 60 segons la columna de suport, només ho han fet el 67%, la resta s'ha classificat de forma incorrecta en altres classes.

Un altre cas a destacar és el de *tomato mosaic virus*, que té una presició del 100%, és a dir, tots els que s'ha classificat que pertanyen a aquesta classe, realment hi pertanyen, però per altra banda, si s'alanitza l'exhaustivitat, dels 22 elements que s'haurien d'haver identificat com a *mosaic virus*, només ho han fet el 55%, és a dir 12 dels 22.

	precision	recall	f1	support
Cherry__Powdery_mildew	0,95	0,98	0,97	62
Cherry__healthy	0,96	1	0,98	51
Grape__Black_rot	0,92	0,83	0,87	70
Grape__Esca_Black_Measles	0,94	0,94	0,94	82
Grape__Leaf_blight_Isariopsis_Leaf_Spot	0,93	0,97	0,95	64
Grape__healthy	0,96	1,00	0,98	25
Tomato__Bacterial_spot	0,92	0,97	0,94	127
Tomato__Early_blight	0,83	0,67	0,74	60
Tomato__Late_blight	0,95	0,84	0,89	114
Tomato__Leaf_Mold	0,8	0,96	0,87	57
Tomato__Septoria_leaf_spot	0,74	0,9	0,81	106
Tomato__Spider_mites_Two-spotted_spider_mite	0,89	0,93	0,91	100
Tomato__Target_Spot	0,87	0,77	0,82	84
Tomato__Tomato_Yellow_Leaf_Curl_Virus	0,98	1	0,99	321
Tomato__Tomato_mosaic_virus	1	0,55	0,71	22
Tomato__healthy	1	0,91	0,95	95
accuracy			0,93	1440
macro avg	0,93	0,94	0,93	1440
weighted avg	0,94	0,93	0,93	1440

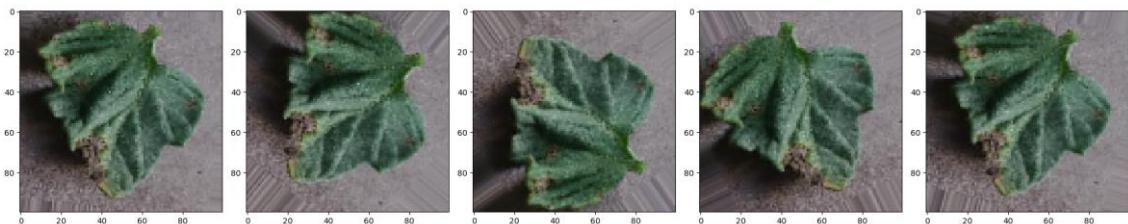
Taula 9 Taula de precision, recall i f1 del model base

Si veiem les mètriques generals però, el model obté valors superiors al 90% tant en presició, en exhaustivitat com en f1. El model és prometedor però s'ha d'analitzar més resultats per comprovar si es poden millorar els errors.

## 5.2 – Experiment amb dades augmentades

En aquest experiment a banda de normalitzar les imatges d'entrada entre 0 i 1 s'ha utilitzat una sèrie d'arguments disponibles a la llibreria Keras que permeten augmentar la variabilitat de les imatges. Les capes per entrenar el model però, són les mateixes que en el model base.

Un exemple de les transformacions que pateixen les imatges el podem veure a la il·lustració 23.



Il·lustració 23 Exemples de les transformacions d'una imatge

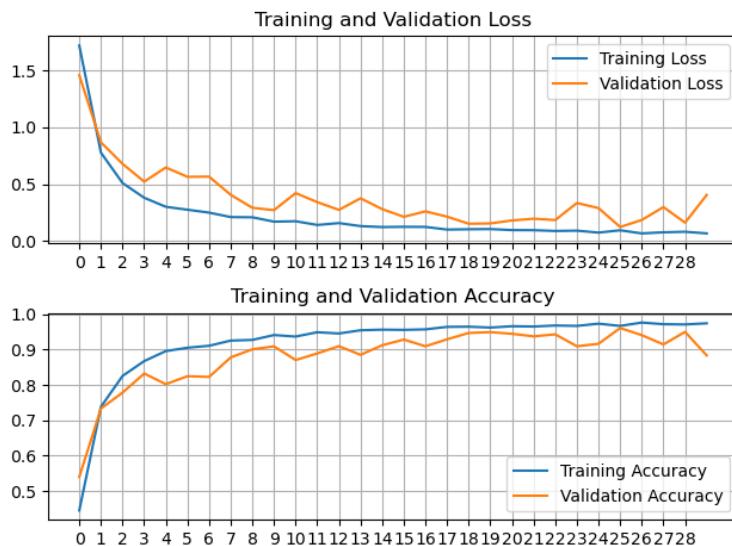
### 5.2.1 Entrenament

Configuració inicial:

- Transfer Learning: no
- Image shape: 100x100 píxels
- Data augmentation: si
- Èpoques: 30
- Batch size: 64
- Pesos iniciais: aleatoris
- Funció d'optimització: Adam
- Learning rate: 0.001
- Loss function: categorical cross entropy
- Últimes capes: Dense(512, activation="relu"); Dense(16, activation= "relu")

En aquest model s'han augmentat les dades i també el nombre d'èpoques per donar més temps a la CNN a aprendre d'elles i millorar els resultats. També s'ha utilitzat la funció d'activació relu a la última capa.

En la il·lustració 24 podem veure l'evolució de l'exactitud i la pèrdua en aquest cas. La pèrdua i l'exactitud de l'entrenament estan arribant al seu límit en les últimes èpoques ja que no hi ha una millora destacable en els resultats obtinguts i sembla que s'està estancant. Pel que fa a les dades de validació s'observa que els resultats obtinguts són més irregulars però de forma general descriuen un comportament similar al entrenament i tenen tendència a descriure la mateixa corba. En les últimes èpoques el comportament mostra una tendència a empitjorar els resultats, això seria indicatiu que s'ha arribat al límit que pot aprendre el model amb aquesta configuració.

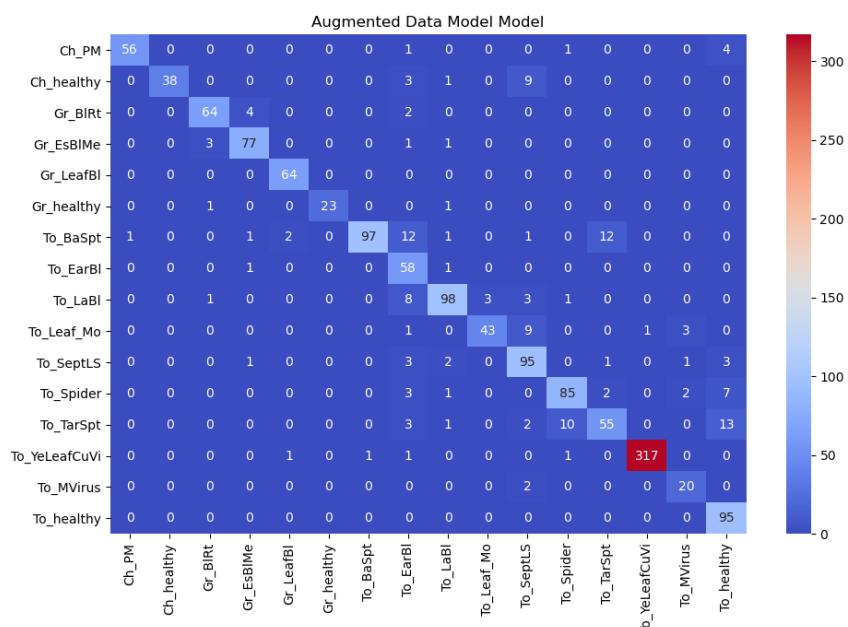


Il·lustració 24 Resultats d'exactitud i pèrdua durant l'entrenament del model amb dades augmentades

### 5.2.2 – Resultats

Si s'analitzen els resultats obtinguts en la il·lustració 25 amb la classificació feta pel model amb les dades de test s'observa una línia diagonal clara però hi ha certes valors que mostren resultats incorrectes. La classe *tomato healthy* té 13 falsos negatius amb *tomato target spot* pel que sembla tenir tendència a la confusió entre aquestes classes. També té 7 falsos negatius amb *tomato spider mites* pel que també mostra una tendència a confondre aquesta classe amb la de la tomaquera saludable.

Tal com succeeix en el cas anterior *tomato bacterial spot* té tendència a ser confós pel *tomato target spot* i per *tomato early blight*.



Il·lustració 25 Matriu de confusió del model amb dades augmentades generat amb les dades de test

Si analitzem les dades des del punt de vista de la precisió i l'exhaustivitat com es veu a la taula 10, veurem que els casos analitzats (tomato target spot i Tomato bacterial spot) mostren els valors més baixos de f1, són els que en general tenen més problemes per a ser classificats correctament.

Les mètriques generals ens mostren que aquest model tindria una efectivitat per sota del 90% en cas de ser utilitzat en un entorn real.

	precision	recall	f1	support
Cherry__Powdery_mildew	0,98	0,9	0,94	62
Cherry__healthy	1	0,75	0,85	51
Grape__Black_rot	0,93	0,91	0,92	70
Grape__Esca_Black_Measles	0,92	0,94	0,93	82
Grape__Leaf_blight_Isariopsis_Leaf_Spot	0,96	1	0,98	64
Grape__healthy	1,00	0,92	0,96	25
Tomato__Bacterial_spot	0,99	0,76	0,86	127
Tomato__Early_blight	0,6	0,97	0,74	60
Tomato__Late_blight	0,92	0,86	0,89	114
Tomato__Leaf_Mold	0,93	0,75	0,83	57
Tomato__Septoria_leaf_spot	0,79	0,9	0,84	106
Tomato__Spider_mites_Two-spotted_spider_mite	0,87	0,85	0,86	100
Tomato__Target_Spot	0,79	0,65	0,71	84
Tomato__Tomato_Yellow_Leaf_Curl_Virus	1	0,99	0,99	321
Tomato__Tomato_mosaic_virus	0,77	0,91	0,83	22
Tomato__healthy	0,78	1	0,88	95
accuracy			0,89	1440
macro avg	0,89	0,88	0,88	1440
weighted avg	0,91	0,89	0,89	1440

Taula 10 Taula de precision, recall i f1 del model amb dades augmentades

### 5.3 – Experiment amb regularització L2

- Transfer Learning: no
- Image shape: 100x100 píxels
- Data augmentation: si
- Èpoques: 20
- Batch size: 64
- Pesos iniciais: aleatoris
- Funció d'optimització: Adam
- Learning rate: 0.001
- Loss function: categorical cross entropy
- Últimes capes: Dense(512, activation="relu"); Dense(16, activation= "none")
- Activity regularizer: L2(0.001)

#### 5.3.1 – Entrenament

S'ha fet 6 proves diferents l'última de les quals és la que aquí s'exposa. Les altres proves tenien les següents característiques:

Prova 1: L2(0.01) + capes de batch normalization (Axis = -1)

Prova 2: L2(0.001) + capes de batch normalization (Axis = -1)

Prova 3: L2(0.01) + capes de batch normalization (Axis = 1)

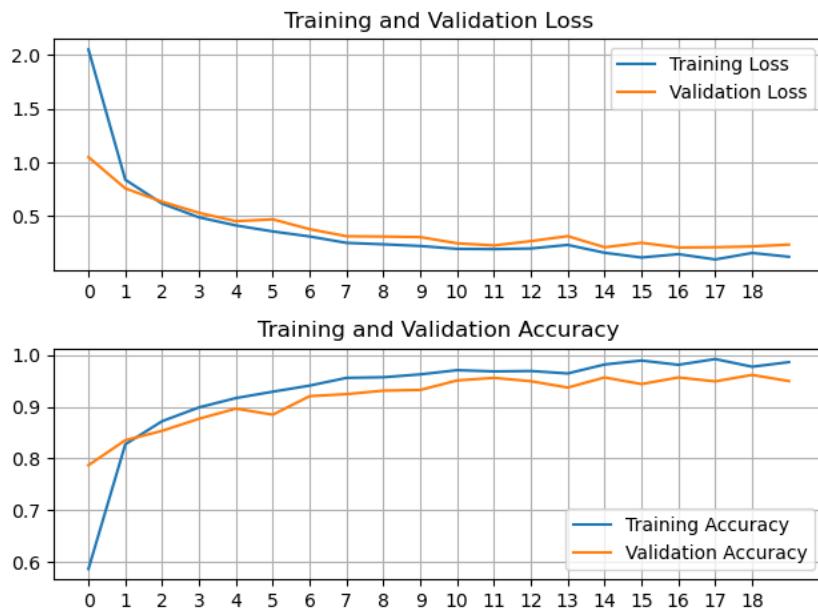
Prova 4: L2(0.001) + capes de batch normalization (Axis = -1)

Prova 5: L2(0.01) sense batch normalization

La resta de característiques es mantenen igual que la configuració inicial ja mostrada. Es va descartar l'ús del batch normalization perquè no actuava correctament amb la regularització L2 i els resultats de validació contenien molt de soroll i eren inestables.

Si analitzem la gràfica de la il·lustració 26 corresponent a la millor prova veurem que les línies d'accuracy i loss tant d'entrenament com de validació descriuen una corba i uns valors molt similars. Això és símptoma d'una xarxa neural ben entrenada. Els valors de pèrdua d'ambdós conjunts descriuen una trajectòria descendent que es va tornant horitzontal progressivament però sense arribar a la horitzontalitat absoluta.

Per una banda els últims valors de pèrdua de l'entrenament són 0,1483; 0,0992; 0,1595 i 0,1241 i els de la validació 0,2103; 0,2128; 0,2199 i 0,2374. Per l'altra banda, els valors d'exactitud són 98,11%, 99,22%, 97,74% i 98,63% per al entrenament i 95,66%, 94,91%, 96,16% i 94,96% per a la validació.



Il·lustració 26 Resultats d'exactitud i pèrdua durant l'entrenament del model amb regularització L2

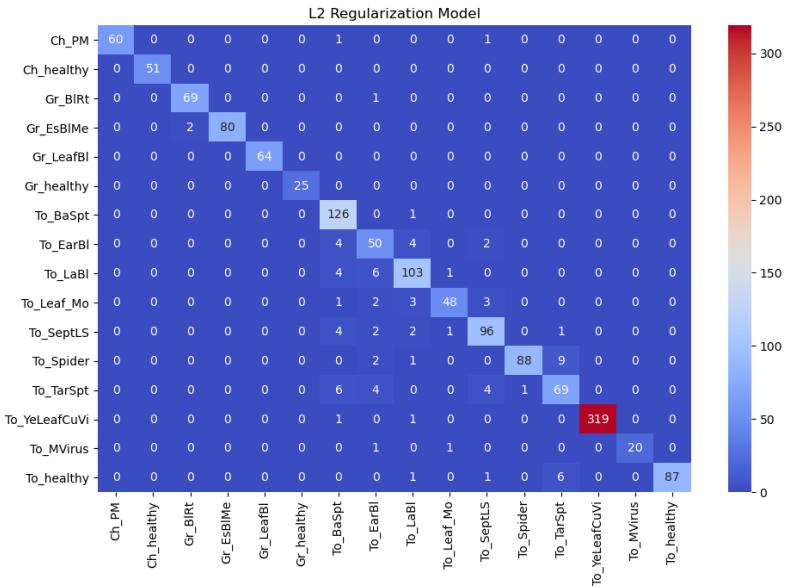
### 5.3.2 – Resultats

Per assegurar que la xarxa funciona correctament apliquem la mateixa prova que en la resta de casos fent que el model entrenat predigui la classe de les imatges de test.

Si s'observa la il·lustració 27 es veu que hi ha una diagonal diferenciada marcant els elements encertats correctament. No s'observa cap anomalia que destaquï considerablement, els casos on es veuen més errades corresponen a la classe *tomato late blight* en la qual 11 de les imatges

no s'han classificat correctament. D'aquestes imatges 6 s'han classificat com a *tomato early blight*, 4 com a *tomato bacterial spot* i 1 com a *tomato leaf mold*.

De *tomato target spot* també s'observen 15 imatges mal classificades. Les primeres 6 s'han classificat com *tomato bacterial spot*, 4 com a *tomato early blight*, 4 més com a *tomato septoria leaf spot* i la restant com a *tomato spider mites*.



Il·lustració 27 Matriu de confusió del model amb regularització L2 generat amb les dades de test

Si ho analitzem amb les dades de precision, recall i f1 de la taula 11 es veu com efectivament *tomato late blight* tot i els problemes que presenta encara té una puntuació de 0.9 en f1, en termes generals no està funcionant tan malament a la hora de classificar en aquesta classe. La precisió és del 89% i l'exhaustivitat del 90%.

Si que s'observen mètriques pitjors per al *tomato early blight* que té un f1 de 0.78. Això es deu a que la seva precisió és del 74% i l'exhaustivitat del 83%. En un primer moment no semblava que això fos així ja que la matriu de confusió aparenta tenir pitjors resultats el *tomato late blight*. Això és així perquè la matriu de calor no té en compte el pes de cada mostra en el total, de *early blight* tenim només 60 imatges de test i de *late blight* 114, això fa que tot i tenir més quantitat d'imatges mal classificades el percentatge final sigui inferior. En el cas de la classe *tomato leaf mold* passa el mateix, sembla tenir un millor comportament a partir de la matriu de confusió però en realitat té un comportament lleugerament pitjor al de *late blight*.

*Tomato target spot* també té un comportament poc precís i poc exhaustiu, un 81% i un 82% respectivament, pel que sembla que aquesta classe genera problemes a la hora de ser classificada.

Les mètriques general mostren que la precisió general estaria al 94% i l'exhaustivitat entre el 93 i el 94%.

		precision	recall	f1-score	support
Cherry	Powdery_mildew	1,00	0,97	0,98	62
Cherry	healthy	1,00	1,00	1,00	51
Grape	Black_rot	0,97	0,99	0,98	70
Grape	Esca_Brown_Rot	1,00	0,98	0,99	82
Grape	Leaf_blight_Isariopsis_Leaf_Spot	1,00	1,00	1,00	64
Grape	healthy	1,00	1,00	1,00	25
Tomato	Bacterial_spot	0,86	0,99	0,92	127
Tomato	Early_blight	0,74	0,83	0,78	60
Tomato	Late_blight	0,89	0,90	0,90	114
Tomato	Leaf_Mold	0,94	0,84	0,89	57
Tomato	Septoria_leaf_spot	0,90	0,91	0,90	106
Tomato	Spider_mites_Two-spotted_spider_mite	0,99	0,88	0,93	100
Tomato	Target_Spot	0,81	0,82	0,82	84
Tomato	Tomato_Yellow_Leaf_Curl_Virus	1,00	0,99	1,00	321
Tomato	Tomato_mosaic_virus	1,00	0,91	0,95	22
Tomato	healthy	1,00	0,92	0,96	95
accuracy				0,94	1440
macro avg		0,94	0,93	0,94	1440
weighted avg		0,94	0,94	0,94	1440

Taula 11 Taula de precision, recall i f1 del model amb regularització L2

## 5.4 – Model amb batch normalization

Aquest model utilitza la segona arquitectura mostrada en l'apartat 4.2.2. La configuració inicial és:

- Transfer Learning: no
- Image shape: 100x100 píxels
- Data augmentation: si
- Èpoques: 20
- Batch size: 64
- Pesos iniciais: aleatoris
- Funció d'optimització: Adam
- Learning rate: 0.001
- Loss function: categorical cross entropy
- Últimes capes: Dense(512, activation="relu"); Dense(16, activation= "none")
- Batch normalization: Axis = 1, momentum = 0.99

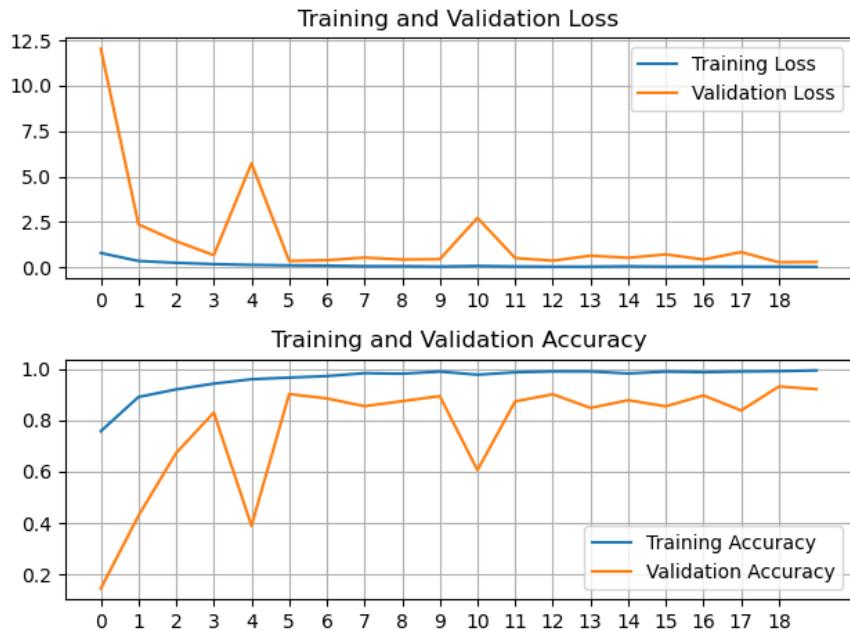
El millor resultat en aquest model s'ha obtingut configurant el paràmetre axis de la capa de batch normalization a 1.

### 5.4.1 – Entrenament

Si observem la il·lustració 28 es pot veure que els valors d'exactitud arriben pràcticament al 100%, de fet les últimes 4 èpoques aconsegueixen uns valors de 98,8%, 99,09%, 99,23% i

99,53%. En aquest punt el model pràcticament no té marge per seguir millorant la seva exactitud. Pel que fa a la pèrdua, el grup d'entrenament es manté pràcticament al mínim de 0 des de la 4 època. En aquest cas, les últimes 4 èpoques aconsegueixen uns valors de 0,0334; 0,0271; 0,0238 i 0,0146, al igual que amb l'exactitud, el model es queda sense marge per seguir aprenent.

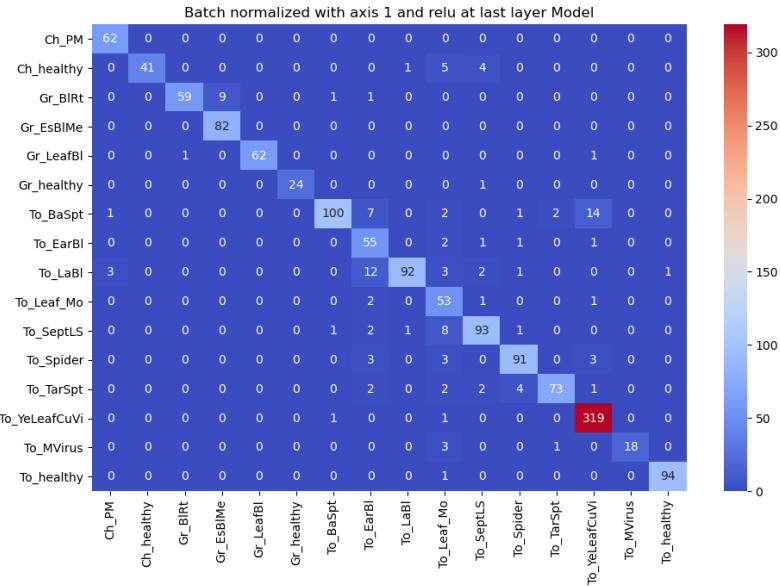
Si comparem aquests valors amb els de validació, veiem que aquest últim grup presenta més soroll que l'anterior, però tret de les èpoques 4 i 10 que marquen pics amb valors clarament pitjors, la resta segueix la tendència del grup d'entrenament. Els últims 4 valors d'exactitud són 89.77%, 83.88%, 93.25% i 92.21% i els de pèrdua 0.4226; 0.8294; 0.2724 i 0.2867. En proves realitzades amb més èpoques no s'obtenen resultats millors, tot apunta a que el model queda estancat en aquest punt.



Il·lustració 28 Resultats d'exactitud i pèrdua durant l'entrenament del model amb batch normalization

#### 5.4.2 – Resultat

Realitzant la prova amb les dades de test utilitzant el model entrenat anterior obtenim la matriu de confusió de la il·lustració 29. Aquesta matriu descriu clarament la línia diagonal que ens indica que en termes generals fa bones prediccions.



Il·lustració 29 Matriu de confusió del model amb batch normalization generat amb les dades de test

Destaquen 14 prediccions fetes com a *tomato yellow leaf curl virus* que en realitat pertanyen a la classe de *tomato bacterial spot*. Aquesta classe té 127 imatges de test 14 de les quals s'han identificat incorrectament en una mateixa classe. Això és un 11% de la classe pel que sembla que té una tendència important a confondre-les.

També destaca la classe *tomato early blight*, s'han predit correctament 55 de les 60 imatges de test que conté donant un recall del 92% com es pot veure a la taula 12, però en canvi, s'han classificat dins d'aquesta classe 12 imatges de *tomato late blight* i 7 de *tomato bacterial spot*. El primer cas equivaldria a un 10.5% de la classe *late blight* mal classificada considerant només la classe *early blight*, d'aquí que el seu valor de recall sigui de només 81%. En el segon cas el 5.5% de les imatges de *tomato bacterial spot* s'han classificat com a *early blight*, però en aquest cas, ja hi havia un 11% mal classificat com hem vist anteriorment, això deixa un mínim del 16.5%. Per aquest motiu, el recall del *bacterial spot* cau fins a un 79%. Aquestes confusions de la CNN també fan que *tomato early blight* tingui una precisió del 65%.

La precisió de *cherry healthy* és del 100% pel que totes les imatges que s'han classificat en aquesta classe són correctes, no obstant, només 41 de les 51 que haurien de ser-ho han estat predites. Les altres es reparteixen en 1 a la classe *tomato late blight*, 5 a la classe *tomato leaf mold* i 4 a la classe *tomato septoria leaf spot* pel que el recall és del 80%.

Les mètriques generals mostren que la precisió oscil·la entre els 92% i el 93% i l'exhaustivitat entre el 90% i el 92%.

		precision	recall	f1-score	support
Cherry	Powdery_mildew	0,94	1	0,97	62
Cherry	healthy	1	0,8	0,89	51
Grape	Black_rot	0,98	0,84	0,91	70
Grape	Esca_Brown_Rot	0,9	1	0,95	82
Grape	Leaf_blight_Isariopsis_Leaf_Spot	1	0,97	0,98	64
Grape	healthy	1,00	0,96	0,98	25
Tomato	Bacterial_spot	0,97	0,79	0,87	127
Tomato	Early_blight	0,65	0,92	0,76	60
Tomato	Late_blight	0,98	0,81	0,88	114
Tomato	Leaf_Mold	0,64	0,93	0,76	57
Tomato	Septoria_leaf_spot	0,89	0,88	0,89	106
Tomato	Spider_mites_Two-spotted_spider_mite	0,92	0,91	0,91	100
Tomato	Target_Spot	0,96	0,87	0,91	84
Tomato	Tomato_Yellow_Leaf_Curl_Virus	0,94	0,99	0,97	321
Tomato	Tomato_mosaic_virus	1	0,82	0,9	22
Tomato	healthy	0,99	0,99	0,99	95
accuracy				0,92	1440
macro avg		0,92	0,9	0,91	1440
weighted avg		0,93	0,92	0,92	1440

Taula 12 Taula de precision, recall i f1 del model amb batch normalization

## 5.5 – Model amb dropout

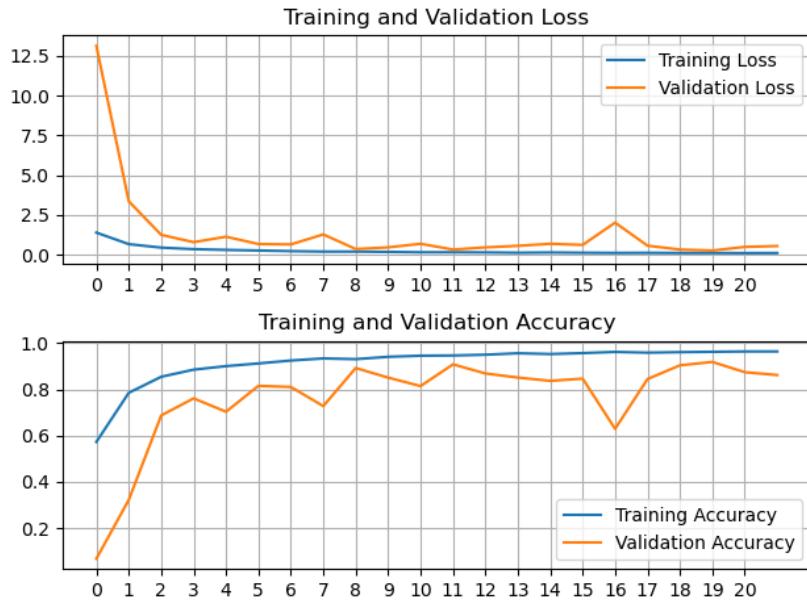
- Transfer Learning: no
- Image shape: 100x100 pixels
- Data augmentation: si
- Èpoques: 22
- Batch size: 64
- Pesos iniciais: aleatoris
- Funció d'optimització: Adam
- Learning rate: 0.001
- Loss function: categorical cross entropy
- Últimes capes: Dense(512, activation="relu"); Dense(16, activation= "relu")
- Batch normalization: Axis = 1, momentum = 0.99
- Dropout: En la primera capa dropout = 0.05, a la resta dropout = 0.35

### 5.5.1 – Entrenament

Per a l'entrenament d'aquest model s'han realitzat fins a 18 proves modificant els valors del dropout i en alguns casos utilitzant el batch normalitzacions i en d'altres no. En general, l'entrenament sempre ha mostrat un comportament similar, la diferència més important s'observa en l'accuracy i el loss de la validació, ja que el batch normalization sembla ajudar a reduir el soroll i a donar més estabilitat.

El resultat de l'entrenament d'aquest model el podem veure a la il·lustració 30. En ella s'observa que de forma general les corbes de validació descriuen una trajectòria similar i propera a la

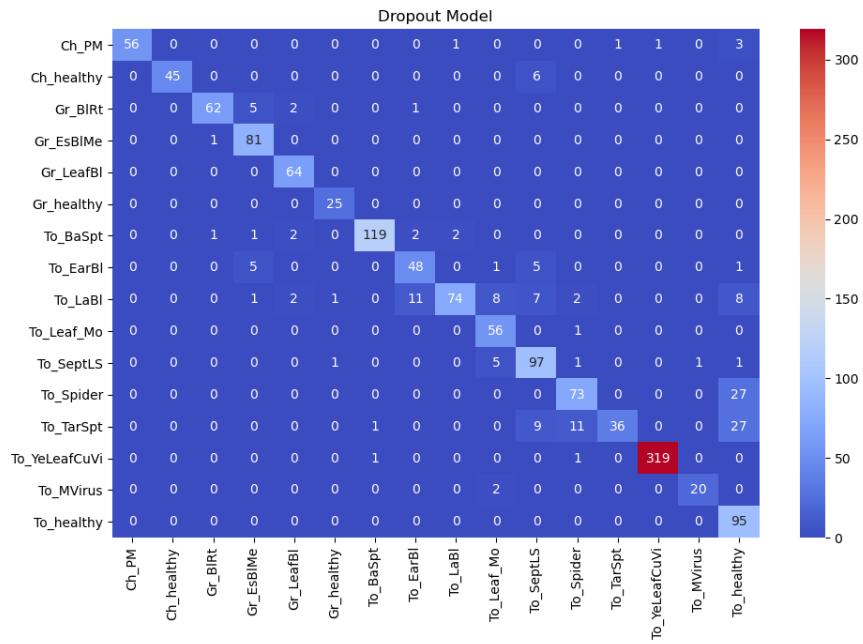
d'entrenament. La excepció la trobem a la època 16 on el gràfic mostra un pic tant al loss com a l'accuracy de la validació. El valors finals d'entrenament són propers a 0 per la pèrdua i properes al 90% per al accuracy.



Il·lustració 30 Resultats d'exactitud i pèrdua durant l'entrenament del model amb batch normalization i dropout

### 5.5.2 – Resultats

Si s'analitzen els resultats obtinguts per a la prova de test que es poden consultar en la il·lustració 31 es pot observar una diagonal diferenciada de la resta de la matriu. El problema més gran que es visualitza ràpidament el trobem a la columna del *tomato healthy* on veiem que se li ha assignat 27 imatges de les classes *tomato spider mites* i *tomato target spot* respectivament i 8 de la de *tomato late blight* pel que sembla que la xarxa neural tendiria a confondre les mostres per tomaques sanes amb facilitat. De *tomato target spot* també s'assignejan incorrectament 11 imatges a la classe *tomato spider mites* i 9 a la de *tomato septoria leaf spot*. De la classe *tomato late blight* podem veure que a part de les 8 imatges mal classificades a *tomato healthy* hi ha 7 més a *tomato septoria leaf spot*, 8 a *tomato leaf mold* i 11 a *tomato early blight*.



Il·lustració 31 Matriu de confusió del model amb batch normalization i dropout generat amb les dades de test

Si inspeccionem les dades amb l'ajuda de la taula 13 s'observa que tant tomato target spot com tomato late blight tenen un recall de 43% i 65% respectivament. És una evidència que la CNN té dificultats per diferenciar bé aquestes classes i les confon massa sovint. De *late blight* s'han aconseguit classificar correctament 74 imatges de 114 i de *target spot* 36 de 84. Per aquest motiu són les classes que mostren un pitjor valor de f1 junt amb *tomato spider mites* que es veu fortament penalitzada per les 27 imatges mal classificades ja descrites.

Les mètriques generals ens mostren una precisió que oscil·la entre el 89% i el 90% i una exhaustivitat del 88%.

	precision	recall	f1-score	support
Cherry__Powdery_mildew	1,00	0,90	0,95	62
Cherry__healthy	1,00	0,88	0,94	51
Grape__Black_rot	0,97	0,89	0,93	70
Grape__Esca_Black_Measles	0,87	0,99	0,93	82
Grape__Leaf_blight_Isariopsis_Leaf_Spot	0,91	1,00	0,96	64
Grape__healthy	0,93	1,00	0,96	25
Tomato__Bacterial_spot	0,98	0,94	0,96	127
Tomato__Early_blight	0,77	0,80	0,79	60
Tomato__Late_blight	0,96	0,65	0,77	114
Tomato__Leaf_Mold	0,78	0,98	0,87	57
Tomato__Septoria_leaf_spot	0,78	0,92	0,84	106
Tomato__Spider_mites_Two-spotted_spider_mite	0,82	0,73	0,77	100
Tomato__Target_Spot	0,97	0,43	0,60	84
Tomato__Tomato_Yellow_Leaf_Curl_Virus	1,00	0,99	1,00	321
Tomato__Tomato_mosaic_virus	0,95	0,91	0,93	22
Tomato__healthy	0,59	1,00	0,74	95
accuracy			0,88	1440
macro avg	0,89	0,88	0,87	1440
weighted avg	0,90	0,88	0,88	1440

Taula 13 Taula de precision, recall i f1 del model amb batch normalization i dropout

## 5.6 – Model VGG 16

El model està preparat per utilitzar imatges de 224 x224 píxels així que en aquest cas s'ha deixat aquest valor per a la dimensió de les imatges. Una altre diferència respecte als altres models és que com les imatges són més grans s'ha redut el batch size per reduir el cost de computació. S'ha utilitzat els pesos pre-entrenats a ImageNet i només s'ha substituït les últimes capes dense que venen preparades per classificar entre 1000 capes diferents per adaptar-les al problema aquí tractat de classificar entre les 16 classes possibles. Per a indicar que no torni a calcular els pesos de les capes que ja ens interessen i que utilitzi les capes finals que ens interessen s'ha fet mitjançant els paràmetres weights = “imagenet” que carrega els pesos ja entrenats, include\_top = False, que descarta les capes dense del model original i definint l'input shape, això es pot veure al codi 4.

També s'observa com s'especifica que les capes originals no es poden entrenar

```

vgg = VGG16(weights="imagenet", include_top=False, input_shape=(224, 224, 3))

for layer in vgg.layers:
    layer.trainable=False

flat1 = Flatten()(vgg.output)
class1 = Dense(512, activation='relu')(flat1)
output = Dense(16, activation='relu')(class1)

model = Model(inputs=vgg.inputs, outputs=output)

```

Codi 2 Configuració de les capes del model VGG 16

Aquesta és la configuració inicial:

- Transfer Learning: si
- Image shape: 224x224 píxels
- Data augmentation: si
- Èpoques: 25
- Batch size: 32
- Pesos inicials: Pre-entrenats a ImageNet
- Funció d'optimització: Adam
- Learning rate: 0.001
- Loss function: categorical cross entropy
- Últimes capes: Dense(1024, activation="relu"); Dense(16, activation= "none")

S'ha realitzat 4 proves amb aquesta arquitectura:

Prova 1: Dense(512, activation="relu"); Dense(16, activation= "none")

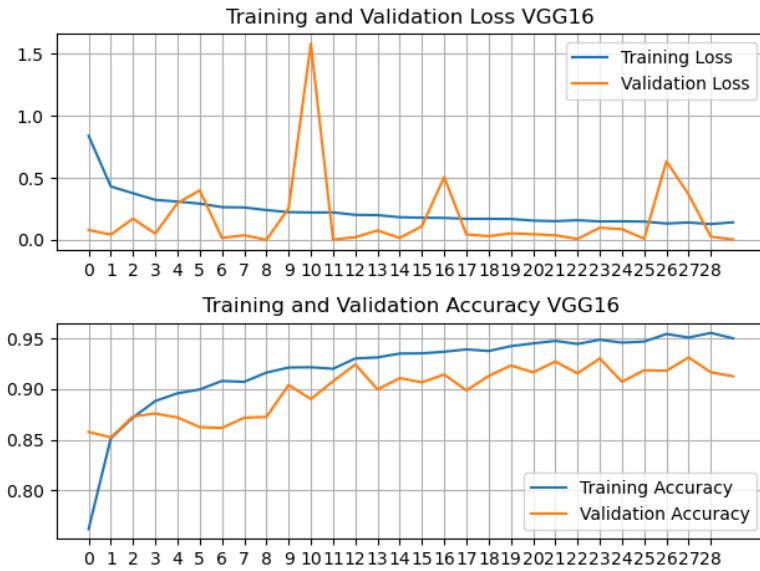
Prova 2: Dense(512, activation="relu"); Dense(16, activation= "relu")

Prova 3: Dense(1024, activation="relu"); Dense(16, activation= "relu")

Els resultats que es mostren són els de la quarta prova amb la configuració inicial descrita.

### 5.6.1 – Entrenament

Si s'observa la il·lustració 32 es veu com els valors d'accuracy d'entrenament han seguit una paràbola ascendent i que en les últimes èpoques s'ha començat a estancar i sembla incapaç de millorar i que la pèrdua ha seguit una paràbola descendent que també s'està estancant en les últimes èpoques. Pel que fa al conjunt de validació, els resultats de la pèrdua mostren molt de soroll amb pics repartits al llarg de tota la història, això fa pensar que el model no acaba de trobar els valors que farien ajustar correctament el model. En quant a la validació, la corba que descriu és millor que la de la pèrdua, no obstant, sembla mantenir-se apartada de la d'entrenament i no que no acaba de trobar els valors dels pesos que facin ajustar millor els resultats.



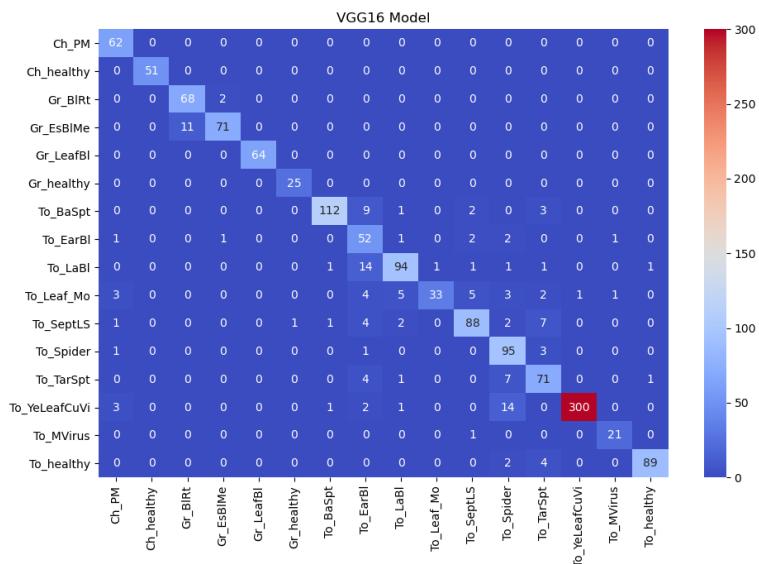
Il·lustració 32 Resultats d'exactitud i pèrdua durant l'entrenament del model VGG16

### 5.6.2 – Resultats

Si s'observa la il·lustració 33 veurem que el model és capaç de definir correctament la diagonal que mostra els resultats correctes. Aquest model mostra un error més gran en la classificació de la classe *tomato yellow leaf curl virus*, 14 de les 321 imatges s'han classificat com *tomato spider mites*, això suposa un 4,36% del total d'imatges de la classe.

També s'observa que la classe *tomato target spot* ha tendit a ser confosa per la de *tomato spider mites* però al mateix temps 7 imatges de la classe *tomato septoria leaf spot* s'han classificat com a *target spot*. També destaquen 14 imatges de la classe *tomato late blight* mal classificades com a *tomato early blight* i 9 imatges *tomato bacterial spot* també classificades com a *early blight*.

Altrament 11 imatges de la classe *grape esca Black measles* també s'han classificat incorrectament com a *grape Black root*.



Il·lustració 33 Matriu de confusió del model VGG 16 generat amb les dades de test

Si s'analitza amb la taula 14 dels resultats de precisió, recall i f1 del model observem que les que obtenen una puntuació més baixa de f1 i que per tant tenen un comportament pitjor en general són les classes *tomato early blight* amb un 0.69, *tomato leaf mold* amb un 0.73 i *tomato target spot* amb un 0.81. La primera té una precisió força baixa, d'un 58% i una exhaustivitat del 87%, mentre que la segona classe, té una exhaustivitat del 58% però una precisió del 97%. Això significa que les que classifica com a *leaf mold* quasi sempre ho són però se'n deixa força de les que hi haurien d'anar ja que les classifica de forma incorrecte. Mentre que en el cas de *tomato early blight* és al contrari, classifica un alt percentatge de les que hi ha d'anar correctament però al mateix temps, posa en la mateixa classe altres imatges que no els pertoca.

Els valors generals mostren una precisió que va del 90% al 91% i una exhaustivitat del 90%.

	precision	recall	f1-score	support
Cherry__Powdery_mildew	0,87	1,00	0,93	62
Cherry__healthy	1,00	1,00	1,00	51
Grape__Black_rot	0,86	0,97	0,91	70
Grape__Esca_Black_Measles	0,96	0,87	0,91	82
Grape__Leaf_blight_Isariopsis_Leaf_Spot	1,00	1,00	1,00	64
Grape__healthy	0,96	1,00	0,98	25
Tomato__Bacterial_spot	0,97	0,88	0,93	127
Tomato__Early_blight	0,58	0,87	0,69	60
Tomato__Late_blight	0,90	0,82	0,86	114
Tomato__Leaf_Mold	0,97	0,58	0,73	57
Tomato__Septoria_leaf_spot	0,89	0,83	0,86	106
Tomato__Spider_mites_Two-spotted_spider_mite	0,75	0,95	0,84	100
Tomato__Target_Spot	0,78	0,85	0,81	84
Tomato__Tomato_Yellow_Leaf_Curl_Virus	1,00	0,93	0,96	321
Tomato__Tomato_mosaic_virus	0,91	0,95	0,93	22
Tomato__healthy	0,98	0,94	0,96	95
accuracy			0,90	1440
macro avg	0,90	0,90	0,89	1440
weighted avg	0,91	0,90	0,90	1440

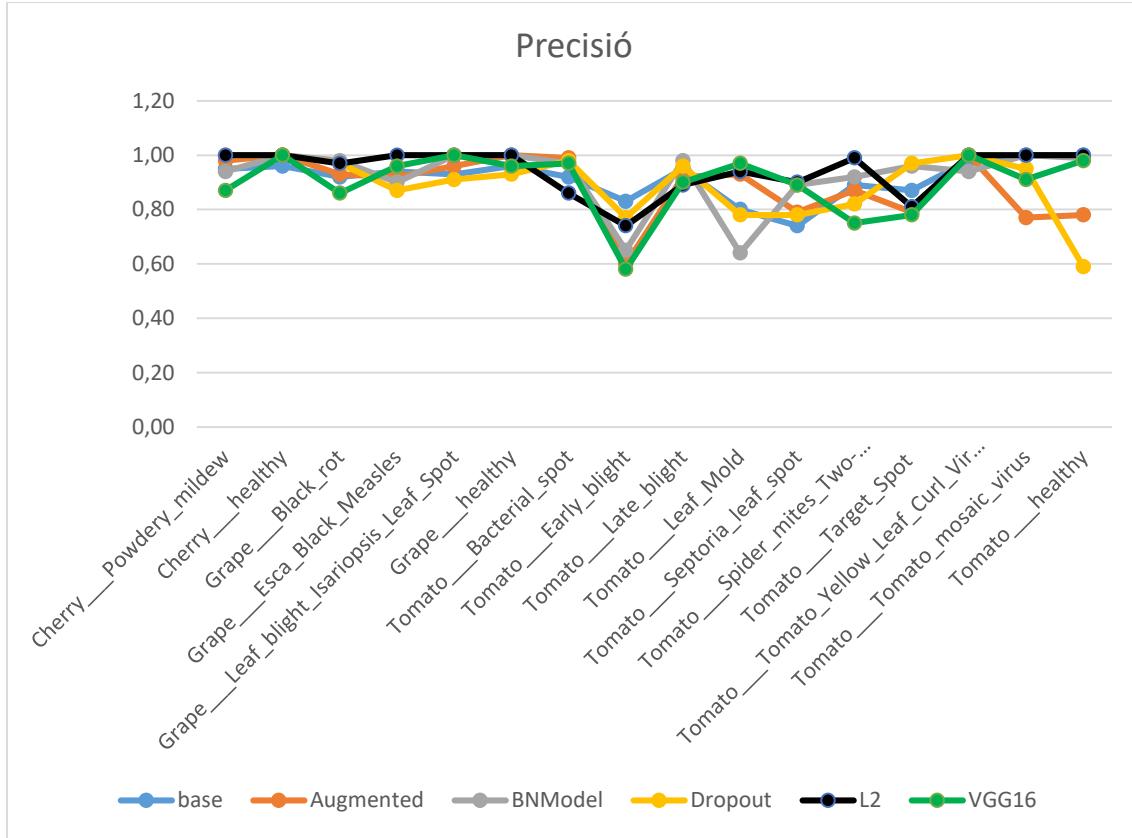
Taula 14 Taula de precision, recall i f1 del model VGG 16

## 5.7 – Discussió

Els resultats de les diferents arquitectures provades han demostrat que totes són capaces de superar el 80% de confiança mínima proposada com a objectiu inicial. D'altra banda s'ha d'observar els valors de precisió i recall ja que com s'ha vist, cada model acostuma a tenir algun problema de confusió amb algunes classes determinades.

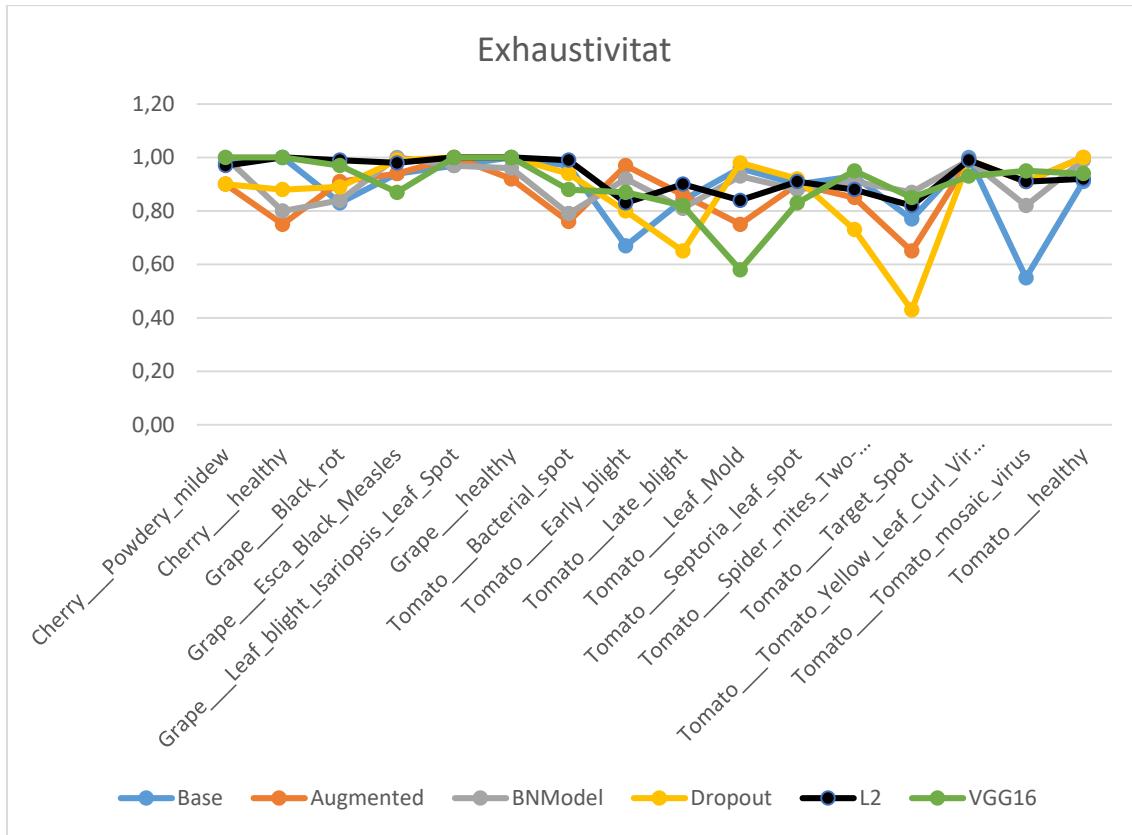
Si s'analitza la il·lustració 34, que ens mostra la precisió obtinguda en cada classe per cada model, veurem que la classe més problemàtica generalment és *tomato early blight*. Això ens indica que els models entrenats solen classificar una considerable quantitat d'imatges d'altres classes incorrectament en aquesta. El model que ha obtingut millor resultat ha sigut el model base seguit del de dropout i poc per sota el model L2. També és pot veure com entre els diferents models hi ha soroll entre les diferents classes de l'espècie tomàquet, és a dir, no hi ha una línia

suau sinó que es percepens pics i valls. Tot apunta a que és on té més problemes per aprendre correctament. Durant l'anàlisi de resultats també s'ha parlat quasi sempre de problemes amb les diferents classes de tomàquet i això ho confirma. De l'anàlisi de la precisió es pot extreure que els models que semblen fer les prediccions més correctes són el model base i el model L2.



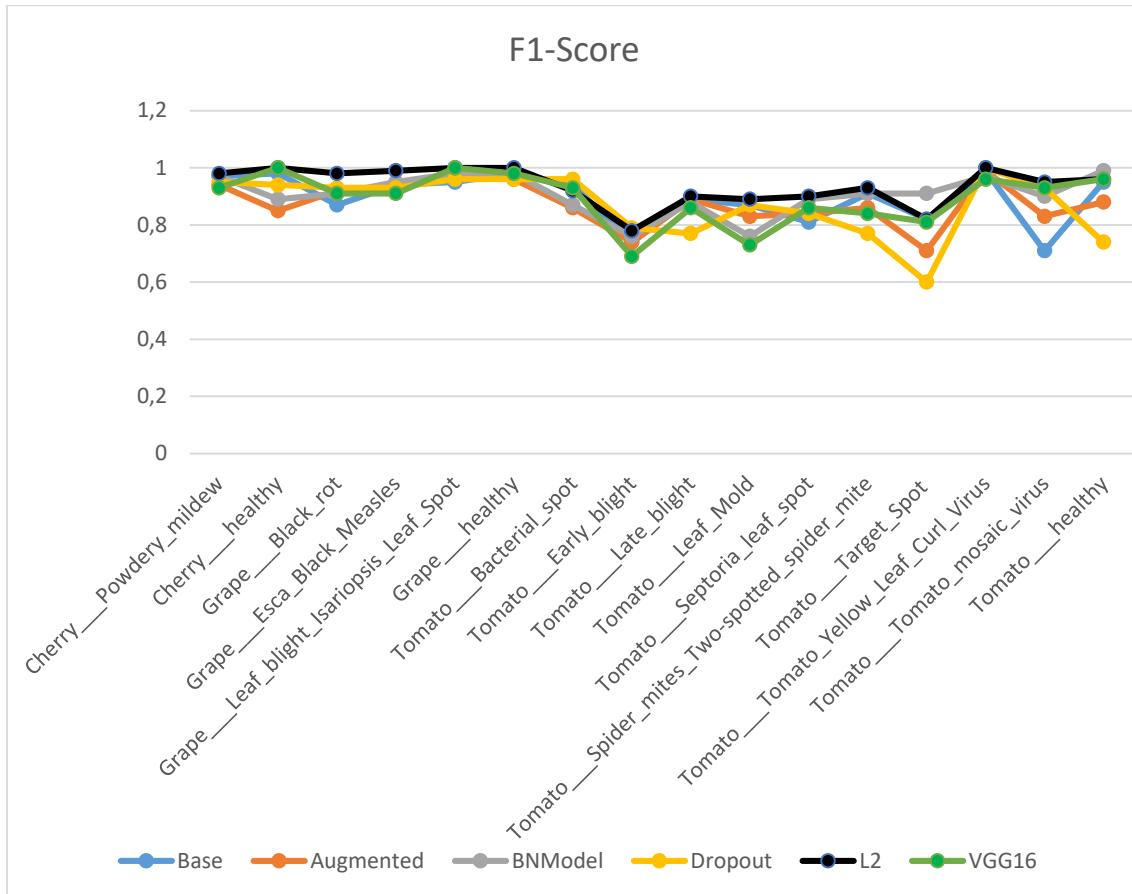
Il·lustració 34 Gràfic que mostra la precisió obtinguda a cada classe en cadascun dels experiments realitzats

Enfocat des del punt de vista de l'exhaustivitat, si analitzem el gràfic resum de la il·lustració 35 se'n desprèn una conclusió similar a l'anterior. En general podem veure que tan l'espècie cirerer com la de cep o vinya tenen una exhaustivitat relativament bona llevat del *cherry healthy* que en el model augmentat i batch normalized obté valors més dolents. En canvi, les diferents classes de l'espècie tomàquet tenen més dificultat i és que com s'ha vist, cada model té alguna tendència a confondre alguna de les classes per les altres. En el gràfic destaca el cas del model dropout amb la classe *tomato target spot* que només ha sigut capaç de predir correctament el 43% de les imatges o el model base amb un 55% amb el *tomato mosaic virus*. Altre vegada, el model L2 sembla el que té els valors més elevats i més constància junt amb el VGG 16 tot i que aquest últim presenta un pic negatiu en *tomato leaf mold* que el perjudica.



Il·lustració 35 Gràfic que mostra l'exhaustivitat obtinguda a cada classe en cadascun dels experiments realitzats

Per últim, si mirem les dades de f1-score en el gràfic de la il·lustració 36 veiem clarament que tots els models empitjoren els resultats quan es tracta de classificar les imatges de fulles de tomaqueries. Destaquen un pic negatiu generalitzat a tots els models en la classe *tomato early blight*. També es veu que el model amb batch normalization i el VGG16 tenen més problemes en la classe *tomato leaf mold* i els models amb dades augmentades i amb dropout presenten els problemes a la classe *tomato target spot*. El model base presenta el pic negatiu a la classe *tomato mosaic virus*. Per tant, analitzant els valors de F1 el model més estable torna a ser el model amb regularització L2.



Il·lustració 36 Gràfic de F1-score de cada classe per cada model

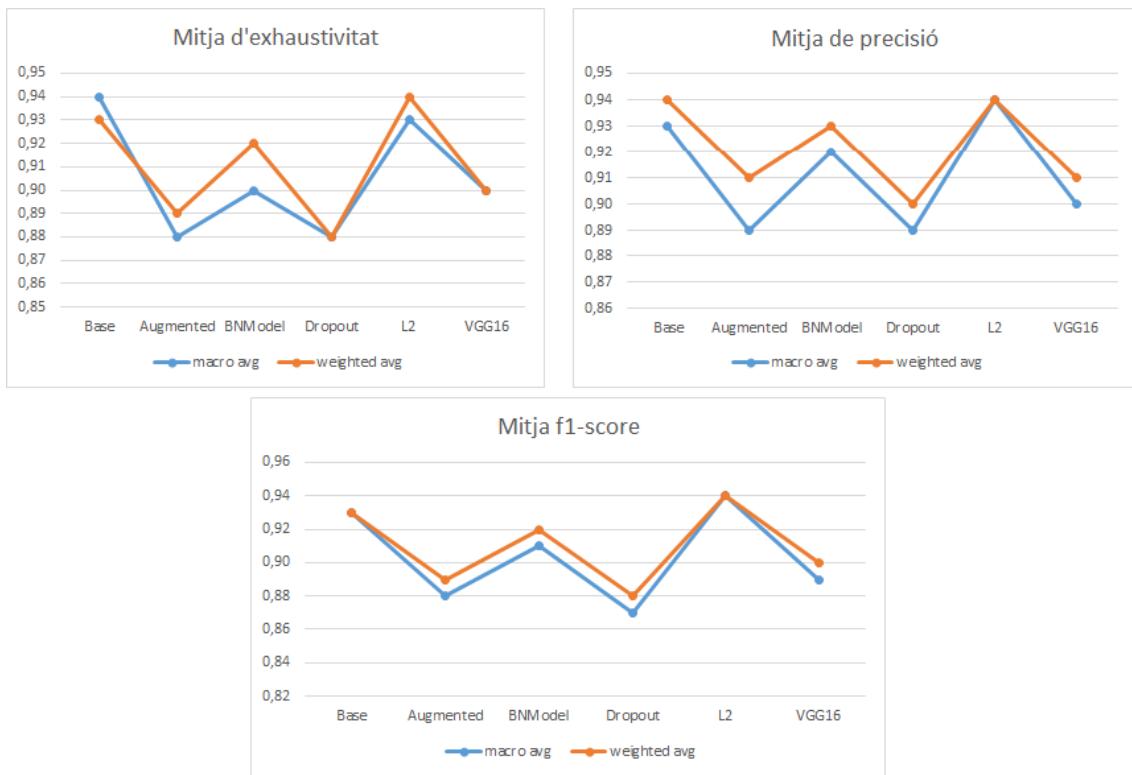
La taula 15 és la matriu que ens mostra la suma d'errors feta entre tots els models. El primer que es pot destacar és que els errors entre espècies diferents es donen menys sovint que els errors dins de la mateixa.

El nombre màxim d'errades el trobem en la classe de *tomato late blight* on fins a un total de 54 ocasions s'ha classificat com a *tomato early blight*, els models generats tenen una tendència elevada a confondre entre aquestes classes. El millor model vist és el que implementa batch normalization amb un 98% de precisió i el L2 amb un 90% d'exhaustivitat. Si utilitzem la mètrica general F1 el millor model és el de L2 amb un 0.90. També hi ha valors dispersos en *tomato target spot* pel que hi ha models que clarament el saben predir millor que d'altres. Un fet curiós és que fins en 19 ocasions els models prediuen incorrectament una fulla de cirerer saludable per la classe *tomato septoria leaf spot*.

Cherry__Powdery_mildew	X							1	1	1		1	1	1	2		7	
Cherry__healthy		X							3	2	5	19						
Grape__Black_rot			X	23	6			2	6			1						
Grape__Esca_Black_Measles			22	X					1	1								
Grape__Leaf_blight_Isariopsis_Leaf_Spot			1		X							2			1			
Grape__healthy			1			X				1	1							
Tomato__Bacterial_spot	3	1	1	2	4	1	X	18	5	2	3	1	14	15				
Tomato__Early_blight	1				7			7	X	11	6	18	3		2	1	1	
Tomato__Late_blight	4		1	1	3	1	8	54	X	19	20	5	1			10		
Tomato__Leaf_Mold	3						1	10	8	X	19	4	2	3	4			
Tomato__Septoria_leaf_spot	2			1		2	6	12	4	19	X	5	9	2	2	4		
Tomato__Spider_mites_Two-spotted_spider_mite	1							10	2	3		X	18	5	2	34		
Tomato__Target_Spot		1						8	14	2	3	24	40	X	2		41	
Tomato__Tomato_Yellow_Leaf_Curl_Virus	3				1			6	3	2	1		16		X			
Tomato__Tomato_mosaic_virus									1		7	9	3	1		X		
Tomato__healthy																X		
	Cherry__Powdery_mildew		Cherry__healthy		Grape__Black_rot		Grape__Esca_Black_Measles		Grape__Leaf_blight_Isariopsis_Leaf_Spot		Grape__healthy							
	Tomato__Bacterial_spot				Tomato__Early_blight				Tomato__Late_blight		Tomato__Leaf_Mold		Tomato__Septoria_leaf_spot		Tomato__Spider_mites_Two-spotted_spider_mite		Tomato__Target_Spot	
															Tomato__Tomato_Yellow_Leaf_Curl_Virus		Tomato__Tomato_mosaic_virus	
																	Tomato__healthy	

Taula 15 Desviació acumulada entre tots els models presentats segons cada classe

Per a fer la selecció final del millor model es presenten els gràfics de les mitges macro i weighted de les mètriques de precisió, exhaustivitat i f1 a la il·lustració 37. On podem veure que pel que fa a l'exhaustivitat, el model base i el model L2 obtenen els mateixos valors, en precisió obté el model L2 els millors valors i en f1-score igualment. El model L2 obté una mitja final del 94% fet que el converteix en el millor model dels provats i un bon model en general.



Il·lustració 37 Mitges macro i weighted de l'exhaustivitat la precisió i f1

Pel que fa a la literatura, es pot destacar la investigació feta per (Jadhav et al., 2020) la qual busca identificar plantes malaltes i sanes de les plantes de soja a partir de les seves fulles. Utilitzen dues xarxes neurals pre-entrenades conegeudes com a AlexNet i GoogleNet amb les quals aconsegueixen una exactitud del 98.75% i del 96.25% respectivament els quals són molt bons resultats. La principal diferència està en què ells només buscaven l'entrenament en una única espècie de planta i és un factor a tenir en compte a la hora d'interpretar els resultats.

Una altre exemple que podem trobar a la literatura és (Mohanty et al., 2016) on s'analitza el conjunt de dades de PlantVillage d'aquest mateix estudi tot i que en el cas de Mohanty et al. Utilitzen el conjunt complet. Fan proves amb AlexNet i GoogleNet com en el cas anterior arribat a aconseguir una exactitud del 99.35% amb GoogleNet.

## 5.8 – Futurs estudis

Una opció per a futurs treballs és la d'acabar de refinjar en la mesura del possible els resultats obtinguts en el present treball i al seu temps crear l'aplicació per a dispositius mòbils que implementi la xarxa neural i aquesta pugui dur a terme les prediccions amb el màxim de fiabilitat possible.

## 6. Conclusions

Els resultats obtinguts mostren la viabilitat d'utilitzar les xarxes neurals convolucionals com a medi per obtenir prediccions sobre la salut dels cultius. Tant els models pre-entrenats disponibles com models desenvolupats des del principi poden arribar a donar molts bons resultats amb el refinament adequat.

Amb un conjunt d'imatges adequat per al entrenament aquestes xarxes han mostrat el seu potencial per a ser implementades en possibles aplicacions per a smartphones que utilitzin les fotografies preses amb el dispositius per a fer la predicció i facilitar la feina dels agricultors.

També s'ha comprovat que és possible realitzar els estudis amb un computador de sobretaula d'ús quotidià tot i que aquest ha de complir uns requisits mínims de hardware perquè el treball sigui viable.

## 7. Bibliografia

- Agrios, G. N. (2004). Fitopatología. *Molecular Plant Pathology*. <https://doi.org/10.1111/mpp.12135>
- Ali, A. (2019, September). *PlantVillage Dataset* / Kaggle. <https://www.kaggle.com/xabdallahali/plantvillage-dataset/kernels>
- Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M. S., Van Esen, B. C., Awwal, A. A. S., & Asari, V. K. (2018). *The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches*. <https://arxiv.org/abs/1803.01164>
- Brownlee, J. (2018, December 3). *A Gentle Introduction to Dropout for Regularizing Deep Neural Networks*. <https://machinelearningmastery.com/dropout-for-regularizing-deep-neural-networks/>
- Chicco, D. (2017). Ten quick tips for machine learning in computational biology. In *BioData Mining*. <https://doi.org/10.1186/s13040-017-0155-3>
- Choudhary, S., Gaurav, V., Singh, A., & Agarwal, S. (2019). Autonomous Crop Irrigation System using Artificial Intelligence. In *International Journal of Engineering and Advanced Technology (IJEAT)* (Issue 8).
- Dertat, A. (2017, November 8). *Applied Deep Learning - Part 4: Convolutional Neural Networks*. <https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2>
- Fukushima, K. (1988). Neocognitron: A hierarchical neural network capable of visual pattern recognition. *Neural Networks*, 1(2), 119–130. [https://doi.org/10.1016/0893-6080\(88\)90014-7](https://doi.org/10.1016/0893-6080(88)90014-7)
- Goldstein, A., Fink, L., Meitin, A., Bohadana, S., Lutenberg, O., & Ravid, G. (2018). Applying machine learning on sensor data for irrigation recommendations: revealing the agronomist's tacit knowledge. *Precision Agriculture*, 19(3), 421–444. <https://doi.org/10.1007/s11119-017-9527-4>
- Jadhav, S. B., Udupi, V. R., & Patil, S. B. (2020). Identification of plant diseases using convolutional neural networks. *International Journal of Information Technology*, 1–10. <https://doi.org/10.1007/s41870-020-00437-5>
- Kaggle: Your Home for Data Science. (n.d.). Retrieved February 25, 2020, from <https://www.kaggle.com/>
- Kaushal, G., & Bala, R. (2017). GLCM and KNN based Algorithm for Plant Disease Detection. *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*. <https://doi.org/10.15662/IJAREEIE.2017.0607036>
- Mohanty, S. P., Hughes, D. P., & Salathé, M. (2016). Using Deep Learning for Image-Based Plant Disease Detection. *Frontiers in Plant Science*, 7(September), 1419. <https://doi.org/10.3389/fpls.2016.01419>
- PlantVillage. (n.d.). Retrieved February 25, 2020, from <https://plantvillage.psu.edu/about-us>
- Pokharna, H. (2016, July 28). *The best explanation of Convolutional Neural Networks on the Internet!* <https://medium.com/technologymadeeasy/the-best-explanation-of-convolutional-neural-networks-on-the-internet-fbb8b1ad5df8>
- Roser, M., Ritchie, H., & Ortiz-Ospina, E. (2020). *World Population Growth - Our World in Data*. Published Online at OurWorldInData.Org. <https://ourworldindata.org/world-population-growth>

growth

Sharma, Sagar. (2017, September 6). *Activation Functions in Neural Networks - Towards Data Science.* <https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6>

Sharma, Shivali, Varinderjit Kaur, E., & Dhillon, N. (2018). PLANT DISEASE CLASSIFICATION WITH KNN-SVM CLASSIFICATION. *International Journal of Advance Engineering and Research Development*, 5.

ul Hassan, M. (2018, November 20). *VGG16 - Convolutional Network for Classification and Detection.* <https://neurohive.io/en/popular-networks/vgg16/>

Wang, H., & Raj, B. (2017). *On the Origin of Deep Learning.* <https://arxiv.org/abs/1702.07800>