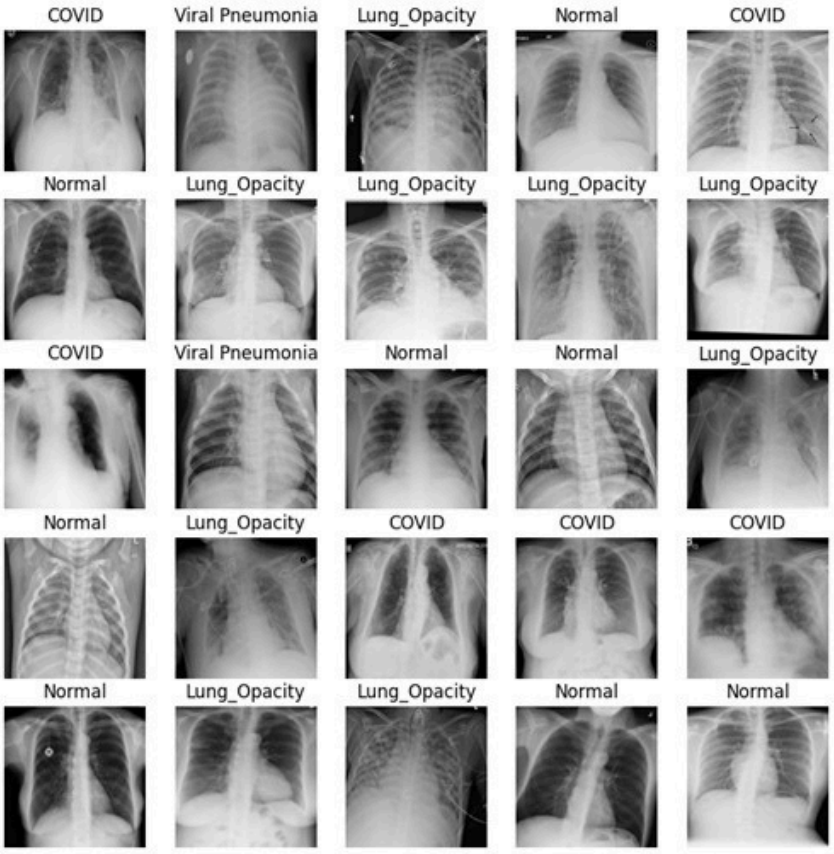
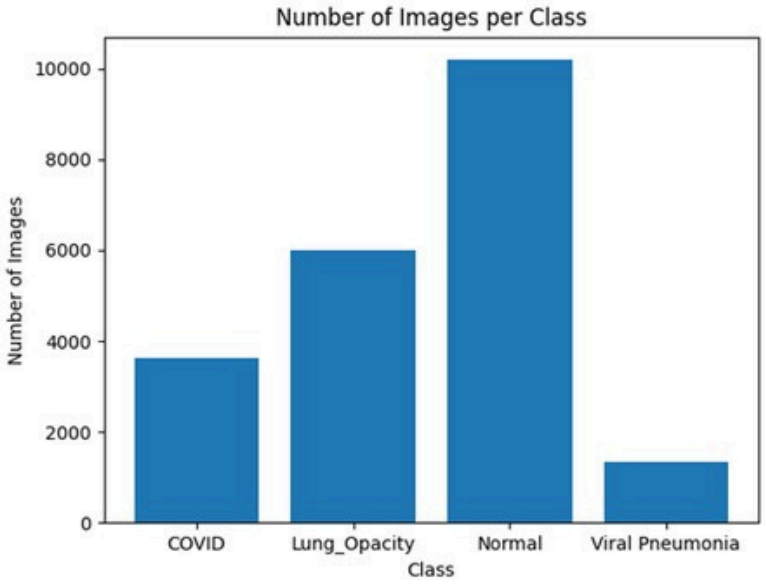
2η Άσκηση: Συνελικτικά Νευρωνικά Δίκτυα

Βασίλειος Κοκκινογένης it2021042

2) Σύνολο δεδομένων

Ζητούμενο 1: Υλοποιήστε την κλάση COVID19Dataset και δημιουργήστε ένα αντικείμενο της με τις εικόνες που κατεβάσατε. Δημιουργήστε επίσης μία λίστα με 25 τυχαίων indexes εικόνων του συνόλου δεδομένων και παρουσιάστε τα χρησιμοποιώντας τη μέθοδο display_batch . Διατρέξτε το σύνολο δεδομένων και υπολογίστε το πλήθος των εικόνων της κάθε κλάσης και απεικονίστε το σε ένα ραβδόγραμμα (bar chart). Σχολιάστε το αποτέλεσμα.





Έχουμε την κλάση ‘Normal’ που έχει τα περισσότερα δείγματα (10192), την κλάση ‘Lung_Opacity’ με 6012 δείγματα, την κλάση ‘COVID’ με 3616 δείγματα και τέλος την κλάση ‘Viral Pneumonia’ με 1345 δείγματα. Η μεγάλη αυτή διαφορά στις ποσότητες των δειγμάτων μεταξύ των κλάσεων υποδηλώνει ότι υπάρχει class imbalance. Αυτό σημαίνει ότι το μοντέλο εκτίθεται σε ορισμένες κλάσεις πολύ πιο συχνά από άλλες, οδηγώντας σε μια προκατειλημμένη διαδικασία μάθησης.

4) Απλό συνελικτικό δίκτυο

Ζητούμενο 2: Ποια είναι η ευστοχία (accuracy) του μοντέλου σας στο σύνολο εκπαίδευσης, στο σύνολο επικύρωσης και στο σύνολο δοκιμής; Δεδομένης της κατανομής των κατηγοριών στο σύνολο δεδομένων, θα χρειαστεί να εξετάσετε λίγο πιο λεπτομερώς τι συμβαίνει στο μοντέλο σας. Με βάση τον πίνακα σύγχυσης, σχολιάστε την επίδοση του μοντέλου σε κάθε κλάση, σε σχέση και με την κατανομή των κλάσεων στο σύνολο δεδομένων. Ποια θεωρείτε ότι είναι η σημασία των αποτελεσμάτων σας στην κλινική πράξη;

Απλό Μοντέλο (Epoch = 5)

```
Training set loss: 0.3189677154077616
Accuracy: 0.8792818332152138
Confusion matrix:
tensor([[1682,  198,  296,   11],
        [ 155, 2875,  568,    5],
        [ 122,  152, 5833,   10],
        [   3,    2,   11,  776]])

Validation set loss: 0.5088557722714712
Accuracy: 0.8192771084337349
Confusion matrix:
tensor([[ 448,  103,  137,    3],
        [  72,  881,  244,    8],
        [  66,   87, 1872,   11],
        [   5,    7,   22,  267]])

Test set loss: 0.5419516284791929,
Accuracy: 0.8110087408457359,
Confusion matrix:
tensor([[ 488,   97,  148,    5],
        [  68,  881,  245,   10],
        [  70,  108, 1844,   17],
        [   8,    5,   19,  220]])

Process finished with exit code 0
```

Απλό Μοντέλο (Epoch = 20)

```
Training set loss: 0.22125055230966764
Accuracy: 0.9170800850460666
Confusion matrix:
tensor([[1605,  398,  184,    0],
        [  23, 3519,   61,    0],
        [  33,  342, 5742,    0],
        [   2,    5,    5,  780]])

Validation set loss: 0.868229127641934
Accuracy: 0.7821875738247106
Confusion matrix:
tensor([[ 336,  196,  157,    2],
        [  30, 1033,  140,    2],
        [  45,  285, 1697,    9],
        [  11,   17,   28,  245]])

Test set loss: 0.9050352004036974,
Accuracy: 0.7710843373493976,
Confusion matrix:
tensor([[ 385,  187,  163,    3],
        [  36, 1017,  148,    3],
        [  52,  321, 1656,   10],
        [   9,   13,   24,  206]])

Process finished with exit code 0
```

Αρχικά έχουμε:

Training Set Accuracy: 91.71%
Validation Set Accuracy: 78.22%
Test Set Accuracy: 77.11%

Δεδομένης της κατανομής των κατηγοριών παρατηρούμε ότι υπάρχει class imbalance, με την κλάση 3 να έχει τα πιο πολλά δείγματα

Στο Training Confusion Matrix έχουμε:

Κλάση 1: Το μοντέλο αποδίδει καλά με 1605 σωστές προβλέψεις, αλλά υπάρχουν 398 εσφαλμένες ταξινομήσεις, κυρίως ως Κλάση 2. Αυτό υποδηλώνει κάποια σύγχυση μεταξύ των δύο κατηγοριών.

Κλάση 2: Αυτή η κατηγορία έχει 3519 σωστές προβλέψεις, αλλά 61 εσφαλμένες ταξινομήσεις, κυρίως ως Κατηγορία 3. Συνολικά, το μοντέλο φαίνεται να έχει πολύ καλή απόδοση εδώ.

Κλάση 3: 5742 σωστές προβλέψεις από 6400, με 342 εσφαλμένες ταξινομήσεις. Αυτό υποδηλώνει ότι η Κλάση 3 αποτελεί εύκολη ταξινόμηση για το μοντέλο. Αυτό ίσως οφείλεται στο γεγονός ότι η κλάση αυτή έχει τα πιο πολλά δείγματα.

Κλάση 4: Αυτή η κατηγορία έχει πολύ υψηλή ακρίβεια με 780 σωστές προβλέψεις, με μόνο 5 εσφαλμένες ταξινομήσεις, υποδεικνύοντας ότι το μοντέλο είναι σίγουρο για την αναγνώριση αυτής της κατηγορίας.

Παρόμοια συμπεριφορά έχουμε και στο Validation και στο Test Confusion Matrix, με μικρότερη όμως ακρίβεια (έχουμε overfitting στα training δεδομένα).
Έχουμε υψηλή ακρίβεια για την κλάση 3, η οποία αφορά την κατηγορία “normal” και στην κλάση 4, η οποία αφορά την κατηγορία “covid”, και μικρότερη ακρίβεια για τις άλλες 2 κλάσεις “lung opacity” και “viral pneumonia”. Αυτή η μικρότερη ακρίβεια για τις άλλες 2 κλάσεις μπορεί να οδηγήσει σε εσφαλμένη διάγνωση ή καθυστέρηση στη θεραπεία του ασθενή. Επιπλέον η απόδοση του μοντέλου στα σύνολα επικύρωσης και δοκιμής δείχνει ότι δυσκολεύεται να γενικευτεί σε νέα, άγνωστα δεδομένα. Σε ένα κλινικό περιβάλλον, αυτό θα μπορούσε να οδηγήσει σε μειωμένη αξιοπιστία του μοντέλου, εφόσον τα δεδομένα είναι πολύ πιθανό να είναι άγνωστα. Για τους λόγους αυτούς δε θεωρώ ότι είναι σημαντικά τα αποτελέσματά του μοντέλου μου στην κλινική πράξη.

5) Συνελικτικό δίκτυο μεγαλύτερου βάθους

Ζητούμενο 3: Συγκρίνετε αυτό το μοντέλο με το προηγούμενο. Παρατηρείτε διαφορά στην επίδοση; Αν ναι, που θεωρείτε ότι οφείλεται;

Βαθύτερο Μοντέλο (Epoch = 5)

```
Training set loss: 0.4791053076784814
Accuracy: 0.8164422395464209
Confusion matrix:
tensor([[1372,  291,  496,   28],
        [ 186, 2902,  462,   53],
        [ 230,  354, 5341,  192],
        [  10,   23,    6,  753]])

Validation set loss: 0.5402290660943558
Accuracy: 0.7880935506732814
Confusion matrix:
tensor([[ 380,  132,  170,    9],
        [  95,  909,  179,   22],
        [  66,  137, 1764,   69],
        [   3,   12,    3,  283]])

Test set loss: 0.5531315785735401,
Accuracy: 0.785731159933853,
Confusion matrix:
tensor([[ 423,   96,  202,   17],
        [  76,  942,  165,   21],
        [  89,  147, 1732,   71],
        [   6,   13,    4,  229]])
```

Βαθύτερο Μοντέλο (Epoch = 20)

```
Training set loss: 0.043882896140891706
Accuracy: 0.9850381919836207
Confusion matrix:
tensor([[2102,   33,   48,    4],
        [   4, 3551,   48,    0],
        [   2,   44, 6065,    6],
        [   0,    1,    0,  791]])

Validation set loss: 0.6588013467948828
Accuracy: 0.8686510748877865
Confusion matrix:
tensor([[ 515,   91,   79,    6],
        [  27, 1017,  159,    2],
        [  22,  132, 1869,   13],
        [   4,    6,   15,  276]])

Test set loss: 0.7219643715157438,
Accuracy: 0.858020316560359,
Confusion matrix:
tensor([[ 553,   93,   84,    8],
        [  33, 1002,  165,    4],
        [  23,  150, 1854,   12],
        [   5,    8,   16,  223]])
```

Η επίδοση του μοντέλου έχει αυξηθεί σε σχέση με το απλό μοντέλο προηγουμένως. Πιο συγκεκριμένα, το training set loss έχει μειωθεί από 0.22 σε 0.04, το validation set loss έχει μειωθεί από 0.86 σε 0.65 και τέλος το test set loss έχει μειωθεί από 0.9 σε 0.72, ενώ αντίστοιχα έχει αυξηθεί και το accuracy. Αυτό πιστεύω ότι οφείλεται στο γεγονός ότι το βαθύτερο μοντέλο έχει περισσότερα layers (convolutional και max-pooling), ενώ στο fully connected layer έχει περισσότερους νευρώνες (1024 αντί για 32). Συνεπώς το βαθύτερο μοντέλο μπορεί να κάνει extract πιο περίπλοκα features απ'ότι το απλό μοντέλο, κάτι που είναι επιθυμητό για περίπλοκα δεδομένα όπως είναι οι εικόνες με διαστάσεις 224x224.

6) Με χρήση προεκπαιδευμένου δικτύου

Ζητούμενο 4: Εκπαιδεύστε το δίκτυο με μέγεθος batch 64 για 5 εποχές με ρυθμό εκμάθησης 10−4 και αξιολογήστε την επίδοσή του (οι υπόλοιπες παράμετροι όπως προηγουμένως). Σχολιάστε τα αποτελέσματα. Πειραματιστείτε επίσης χρησιμοποιώντας το προεκπαιδευμένο μοντέλο ως μηχανισμό εξαγωγής χαρακτηριστικών, με εκπαίδευση μόνο του τελικού επιπέδου ταξινόμησης.

Πλήρες Προεκπαιδευμένο Μοντέλο (Epoch = 5)

```
Training set loss: 0.08020914131773299
Accuracy: 0.9702338766832034
Confusion matrix:
tensor([[2187,    0,    0,    0],
        [  92, 3463,   48,    0],
        [ 155,   69, 5888,    5],
        [   8,    0,    1,  783]])

Validation set loss: 0.3432960783961703
Accuracy: 0.9175525631939523
Confusion matrix:
tensor([[ 686,    2,    3,    0],
        [  62, 1045,   98,    0],
        [  76,   84, 1868,    8],
        [  10,    1,    5,  285]])

Test set loss: 0.3668429777239238,
Accuracy: 0.9073942830144106,
Confusion matrix:
tensor([[ 730,    3,    4,    1],
        [  69, 1033,  102,    0],
        [ 100,   95, 1839,    5],
        [   7,    0,    6,  239]])

Process finished with exit code 0
```

Προεκπαιδευμένο Μοντέλο ως Μηχανισμός Εξαγωγής Χαρακτηριστικών (Epoch = 5)

```
Training set loss: 0.502121039748342
Accuracy: 0.8169147176943067
Confusion matrix:
tensor([[1355,  281,  546,    5],
        [ 165, 2820,  617,    1],
        [ 103,  353, 5599,   62],
        [  27,   16,  149,  600]])

Validation set loss: 0.5118220082993777
Accuracy: 0.8138436097330498
Confusion matrix:
tensor([[ 432,   90,  166,    3],
        [  56,  928,  221,    0],
        [  27,  131, 1863,   15],
        [   8,    8,   63,  222]])

Test set loss: 0.5224546025946455,
Accuracy: 0.8036853295535081,
Confusion matrix:
tensor([[ 442,   97,  195,    4],
        [  64,  919,  221,    0],
        [  27,  139, 1857,   16],
        [  10,    7,   51,  184]])

Process finished with exit code 0
```

Το Πλήρες Προεκπαιδευμένο Μοντέλο (εκπαίδευση σε όλα τα layers) έχει καλύτερες επιδόσεις σε σχέση με το Προεκπαιδευμένο Μοντέλο ως Μηχανισμό Εξαγωγής Χαρακτηριστικών (εκπαίδευση μόνο στο τελευταίο layer).

Σε σχέση με το απλό συνελικτικό δίκτυο (epoch=5), αλλά και με το βαθύ συνελικτικό δίκτυο (epoch=5) το Πλήρες Προεκπαιδευμένο Μοντέλο έχει καλύτερες επιδόσεις, ενώ το Προεκπαιδευμένο Μοντέλο ως Μηχανισμό Εξαγωγής Χαρακτηριστικών έχει παρόμοιες επιδόσεις.

7) Προαιρετικά (Bonus): Συνελικτικό δίκτυο με παραλειπόμενες συνδέσεις

Ζητούμενο bonus: Πειραματιστείτε με τη χρήση του BasicBlock και διάφορες τιμές υπερπαραμέτρων για την ανάπτυξη ενός αποτελεσματικού μοντέλου ταξινόμησης ακτινογραφιών θώρακα για τις δεδομένες παθήσεις. Μπορείτε να εμπνευστείτε από τον τρόπο που οι αρχιτεκτονικές ResNet χρησιμοποιούν παρόμοια μοντέλα για κατηγοριοποίηση εικόνων.

model = ResNet(BasicBlock, [1, 1, 1, 1]) (Epoch = 5)

```
Training set loss: 0.3924774909040114
Accuracy: 0.8476257973068746
Confusion matrix:
tensor([[1744,   90,  352,    1],
        [ 513, 2523,  554,   13],
        [ 193,  100, 5822,    2],
        [  26,   25,   66,  675]])

Validation set loss: 0.48947894716600204
Accuracy: 0.8084101110323647
Confusion matrix:
tensor([[ 508,   49,  133,    1],
        [ 204,  784,  209,    8],
        [  79,   48, 1905,    4],
        [  17,   17,   42,  225]])

Test set loss: 0.5181723988843414,
Accuracy: 0.7944720056697377,
Confusion matrix:
tensor([[ 527,   56,  152,    3],
        [ 210,  757,  224,   13],
        [  82,   66, 1887,    4],
        [  14,   14,   32,  192]])

Process finished with exit code 0
```

model = ResNet(BasicBlock, [2, 2, 2, 2]) (Epoch = 5)

```
Training set loss: 0.3762328636220255
Accuracy: 0.8588077801401686
Confusion matrix:
tensor([[1533,  236,  410,    8],
        [ 211, 2730,  657,    5],
        [ 131,   80, 5895,   11],
        [   4,    1,   39,  748]])

Validation set loss: 0.4430199901309778
Accuracy: 0.8395936687928184
Confusion matrix:
tensor([[ 458,   98,  132,    3],
        [  77,  879,  245,    4],
        [  49,   31, 1951,    5],
        [   7,    5,   23,  266]])

Test set loss: 0.45755246394085436,
Accuracy: 0.8388849515709899,
Confusion matrix:
tensor([[ 494,   94,  146,    4],
        [  65,  893,  240,    6],
        [  50,   47, 1934,    8],
        [   2,    3,   17,  230]])
```

model = ResNet(BasicBlock, [3, 3, 3, 3]) (Epoch = 5)

```
Training set loss: 0.46161699848207327
Accuracy: 0.8254193243562485
Confusion matrix:
tensor([[1196,  176,  803,   12],
        [  77, 2524,  995,    7],
        [  22,   33, 6057,    5],
        [   9,   13,   65,  705]])

Validation set loss: 0.5093751419405892
Accuracy: 0.815024805102764
Confusion matrix:
tensor([[ 350,   83,  256,    2],
        [  29,  832,  341,    3],
        [   5,   17, 2012,    2],
        [   0,    8,   37,  256]])

Test set loss: 0.5461807598764041,
Accuracy: 0.8015591778880227,
Confusion matrix:
tensor([[ 370,   72,  291,    5],
        [  25,  810,  363,    6],
        [   8,   22, 2003,    6],
        [   3,    7,   32,  210]])

Process finished with exit code 0
```

Παρατηρούμε ότι σε σχέση με το συνελικτικό δίκτυο μεγαλύτερου βάθους, το συνελικτικό δίκτυο με παραλειπόμενες συνδέσεις κάνει καλύτερες προβλέψεις στα σύνολα δεδομένων, ωστόσο όχι σε όλες τις περιπτώσεις (π.χ model = ResNet(BasicBlock, [3, 3, 3, 3]))

Όσον αφορά τις διάφορες τιμές υπερπαραμέτρων, παρατηρούμε ότι στα δεδομένα μας, με περισσότερα blocks πετυχαίνουμε καλύτερες επιδόσεις, μέχρι ένα σημείο όμως (μέχρι και το model = ResNet(BasicBlock, [2, 2, 2, 2])), έχοντας μεγαλύτερο computing time.