Αναφορά 2ου Παραδοτέου – Εφαρμογές Τεχνητής Νοημοσύνης

Για την υλοποίηση του προγράμματος ο κώδικας γράφτηκε στη γλώσσα python τόσο στο Google Colab αλλά και σε προγράμματα περιβάλλοντος python όπως το Pycharm. Στο Google Colab η εργασία χωρίζεται σε πολλαπλά προγράμματα.

Το σημαντικό κομμάτι για την υλοποίηση της εργασίας ήταν να εισάγεται το αρχείο με ορθό τρόπο και χωρίς σφάλματα στον κώδικα. Το βασικό zip αρχείο αποθηκεύτηκε στο ατομικό Google Drive του προγραμματιστή και από εκεί δίνεται πρόσβαση στο καθένα πρόγραμμα του Google Colab. Δίνεται σε κάθε αρχείο, δηλαδή, το δικαίωμα να έχουν πρόσβαση στο αρχείο και να εξάγουν τα δεδομένα του.

Όταν, λοιπόν, το αρχείο θα είναι διαθέσιμο, εξάγουμε τα δεδομένα. Καθ' όλη την διάρκεια της εργασίας και για την υλοποίηση των διαφορετικών μοντέλων, μερικά κομμάτια κώδικα και λειτουργίες παραμένουν ίδιες. Αρχικά, καλούνται οι απαραίτητες βιβλιοθήκες. Το επόμενο βήμα είναι να εξάγουμε τα αντικείμενα του συμπιεσμένου αρχείου, εκτελώντας βασικές εντολές που υπήρχαν στην εκφώνηση, παραμετροποιημένες ώστε να αντιστοιχίζονται στο ατομικό Google Drive. Μόλις, τελειώσει η διαδικασία αποσυμπίεσης, διαθέτουμε και τα δεδομένα αλλά και το directory στο οποίο και βρίσκονται αυτά.

Έπειτα, εκτελείται η συνάρτηση prepare_datasets , όπου αξιοσημείωτο εδώ, είναι πως χρησιμοποιείται δύο φορές η δοθείσα συνάρτηση image_dataset_from_directory, μία για το devel_ds και μία για το test_ds. Ο διαχωρισμός των δεδομένων γίνεται με βάση το ποσοστό που θέλουμε για κάθε σύνολο. Έπειτα από το πρώτο παίρνουμε τα train_ds και val_ds αντίστοιχα με προσαρμοσμένες τις τιμές των πιθανοτήτων τους. Αξίζει να σημειωθεί πως αν για παράδειγμα το κομμάτι του συνόλου εκπαίδευσης αντιστοιχεί στο 0.6 του ολικού συνόλου, τότε όταν χρειάζεται να πάρουμε αυτό από το υποσύνολο devel_ds πρέπει να γίνει και η απαραίτητη μετατροπή ώστε να πάρουμε το 0.6 του ολικού από το devel_ds. Στο τέλος της συνάρτησης, επιστρέφουμε τις απαιτούμενες τιμές στο βασικό πρόγραμμα μαζί με τα ονόματα των κλάσεων που συνοδεύουν τις εικόνες από το COVID-19_Radiography_Dataset. Κάπου εδώ, η αναφορά χωρίζεται σε τμήματα ανάλογα με το ερώτημα και με την ερώτηση που είναι απαραίτητο να απαντηθεί. Επίσης να σημειωθεί πως οι τιμές που εμφανίζονται στις παρακάτω εικόνες πιθανώς να διαφέρουν από αυτές που εμφανίζονται στα αρχεία γιατί, από την στιγμή της συγγραφής, ξανά δοκιμάστηκαν ώστε να τρέχουν ομαλά.

Ερώτημα 3.

Για το πρωταρχικό μοντέλο μετά από πολλαπλά πειράματα, πρώτα με μικρό αριθμό δεδομένων και ύστερα με όλα τα δεδομένα, παρατηρήθηκε μεγάλη διαφορά μεταξύ του accuracy & loss ανάμεσα στα train & val αντίστοιχα. Προκύπτει το φαινόμενο που ονομάζεται over fitting, το οποίο δικαιολογείται από την ύπαρξη

πολλών δεδομένων και λίγων φίλτρων και επιπέδων. Η επεξεργασία είναι αδύναμη για τον φόρτο επεξεργασίας.

Για την αντιμετώπισή του ζητήματος, πρωταρχικά χρησιμοποιήθηκε η τεχνική του Data augmentation (με επίπεδα περιστροφής και random zoom) και ύστερα του Dropout με πιθανότητα 0.2 για ολόκληρο το σύνολο. Το πρώτο βήμα μεταβάλλει σε μικρό βαθμό τα αρχεία των εικόνων μέσω περιστροφών, τυχαίας μεγέθυνσης. Το δεύτερο βήμα της μεθόδου, Dropout, επιλέγει τυχαία εισαγόμενα δεδομένα, με συνολικό βαθμό που να φτάνει το 0.2 του συνολικού συνόλου από όπου τα λαμβάνουμε και τα αντικαθιστά με 0 σε κάθε εποχή.

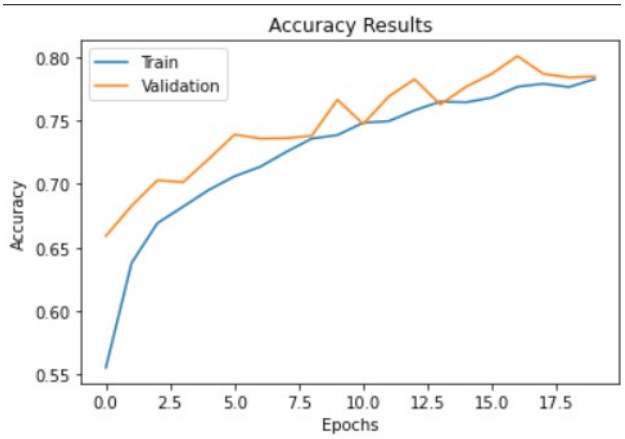
Το επόμενο σημείο είναι η δημιουργία του μοντέλου με βάση τις οδηγίες από την εκφώνηση. Υλοποιείται η τεχνική του Early Stopping όπου, μετά από 3 εποχές στην περίπτωση που δεν υπάρχει βελτίωση της απώλειας στο validation set, το πρόγραμμα θα σταματήσει την εκπαίδευση καθώς δεν σημειώνει σημαντική βελτίωση. Με αυτόν τον τρόπο απαλλάσσονται οι υπολογιστικοί πόροι ή ένα μέρος τους. Επίσης, για την σύνοψη του μοντέλου, χρησιμοποιείται ως συνάρτηση απώλειας η sparse_categorical_crossentropy, για την αυτόματη μετατροπή του σετ σε one-hot, όταν δέχεται σαν είσοδο ακέραιους αριθμούς. Στην περίπτωση που, η συνάρτηση απώλειας ήταν η απλή categorical crossentropy, η μετατροπή θα έπρεπε να γίνει πριν την εισαγωγή των δεδομένων σε αυτήν

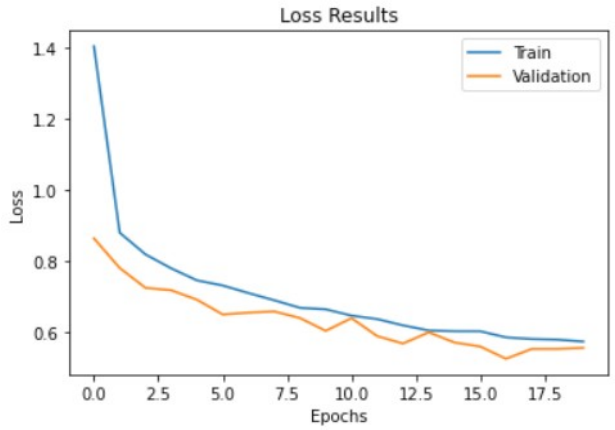
Το πρώτο μοντέλο εκτελείται με μέγιστο δοσμένο αριθμό εποχών τις 20 εποχές. Αφού τερματίσει η εκπαίδευση του μοντέλου, τυπώνονται τα αποτελέσματα σε μορφή διαγράμματος, προκειμένου να ελέγχεται κυρίως το πρόβλημα του over fitting. Για την ολοκλήρωση του πρώτου μοντέλου καλείται η συνάρτηση για την δημιουργία του πίνακα σύγχυσης αλλά και η συνάρτηση αξιολόγησης για το test set.
Ο πίνακας σύγχυσης αποκαλύπτει τα λάθη που κάνει το μοντέλο αναφορικά με την κάθε κατηγορία, ενώ η συνάρτηση αξιολόγησης δείχνει την επίδοση του μοντέλου στο test set. Παρακάτω παρουσιάζεται ενδεικτική δοκιμή του προγράμματος:

```
Epoch 14/20
198/198 [==============================] - 61s 303ms/step - loss: 0.6030 - accuracy: 0.7648 - val_loss: 0.5978 - val_accuracy: 0.7625
Epoch 15/20
198/198 [==============================] - 61s 302ms/step - loss: 0.6009 - accuracy: 0.7643 - val_loss: 0.5693 - val_accuracy: 0.7765
Epoch 16/20
198/198 [==============================] - 61s 302ms/step - loss: 0.6008 - accuracy: 0.7679 - val_loss: 0.5580 - val_accuracy: 0.7867
Epoch 17/20
198/198 [==============================] - 60s 302ms/step - loss: 0.5837 - accuracy: 0.7765 - val_loss: 0.5235 - val_accuracy: 0.8007
Epoch 18/20
198/198 [==============================] - 61s 304ms/step - loss: 0.5790 - accuracy: 0.7789 - val_loss: 0.5512 - val_accuracy: 0.7867
Epoch 19/20
198/198 [==============================] - 60s 302ms/step - loss: 0.5773 - accuracy: 0.7763 - val_loss: 0.5513 - val_accuracy: 0.7839
Epoch 20/20
198/198 [==============================] - 61s 303ms/step - loss: 0.5719 - accuracy: 0.7826 - val_loss: 0.5540 - val_accuracy: 0.7846
```

```
#Confusion matrix to see the errors for each category.
cm , yhat = confusion_matrix(model, test_ds)
print(cm)
#Evaluation function to get the accuracy and loss after test set.
results = model.evaluate(x = test_ds, verbose = 0)
print("Test loss:", results[0])
print("Test accuracy:", results[1])
```

```
tf.Tensor(
[[ 315  232  128    3]
 [  21  998  167   10]
 [  56  268 1722   15]
 [   4   20   26  248]], shape=(4, 4), dtype=int32)
Test loss: 0.5603100061416626
Test accuracy: 0.7755728960037231
```





Αξίζει να σημειωθεί πως χωρίς τα μέτρα για το over fitting το μοντέλο σταματούσε την εκτέλεση μετά από περίπου 7-8 εποχές με μικρότερο αριθμό εποχών τις 5. Η ακρίβεια του συνόλου δοκιμής άγγιζε το 0.98 ή και 0.99 με την ακρίβεια του συνόλου επιβεβαίωσης, ωστόσο να κυμαίνεται σημαντικά χαμηλότερα. Η ίδια απόκλιση ήταν εμφανής και για τη μετρική των απωλειών.

Απάντηση στην ερώτηση 1 : (για το σύνολο δεδομένων δοκιμής)

Παρατηρείται ότι, τα περισσότερα δεδομένα ανήκουν στην ‘Normal’ κατηγορία,με συνολικά περισσότερες από 8000 εικόνες και πάνω από 3000 εικόνες περισσότερες από την αμέσως επόμενη, ‘Lung Opacity’. Τα λιγότερα στοιχεία βρίσκονται στην κατηγορία ‘Viral Pneumonia’ με λιγότερα από 1000. Η τρίτη κατηγορία είναι φανερώς, η Covid-19. Οι τιμές αυτές είναι αποτέλεσμα από τον διαμοιρασμό των δεδομένων στην prepare datasets. Για την διαπίστωση του συμπεράσματος αυτού χρησιμοποιήθηκε κομμάτι κώδικα που δόθηκε στην εκφώνηση, δηλαδή η συνάρτηση confusion matrix αλλά και για την αποτύπωση των στοιχείων δόθηκε ανάλογο κομμάτι.

Απάντηση στην ερώτηση 2): (για την επίδοση του μοντέλου)

Σύμφωνα με τον πίνακα σύγχυσης φαίνεται πως το μοντέλο τα πηγαίνει αρκετά καλά όταν πρόκειται να διακρίνει ένα αντικείμενο για αυτό που είναι. Ωστόσο, εμφανίζει λανθασμένες εκτιμήσεις, όταν δηλώνει μία εικόνα ως κανονική (“Normal”) και εν τέλει προκύπτει ότι ανήκει στην κατηγορία Lung Opacity. Παρόμοια συμπεριφορά παρουσιάζεται και στην περίπτωση όπου προσπαθεί να κατηγοριοποιήσει Covid-19 εικόνες. Οι παραπάνω τιμές, του παραδείγματος για τον πίνακα σύγχυσης, οδηγούν στο συμπέρασμα ότι η ακρίβεια δεν είναι η καλύτερη δυνατή. Η Covid-19 κατηγορία φαίνεται πως είναι και το πρόβλημα αφού οι σωστές εκτιμήσεις είναι 315 με τις λανθασμένες να τις ξεπερνούν. Ποσοστό λάθους πάνω από 50% και συγκεκριμένα 363 εκτιμήσεις λανθασμένες. Επαναλαμβάνεται ότι αυτό είναι απόρροια των μέτρων που έχουν ληφθεί σχετικά με το over fitting καθώς σε συνθήκες όπου δεν υπήρχαν η ακρίβεια είναι αισθητά υψηλότερη. Επομένως, το

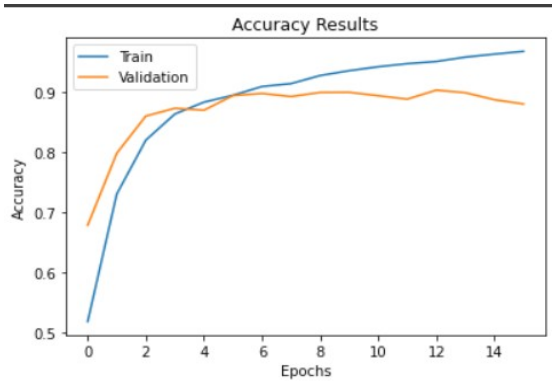
περιθώριο λάθους για αυτήν την υλοποίηση παρότι δεν είναι εξαιρετικά υψηλό, δεν είναι το ιδανικό. Ένας επιπλέον τρόπος να διορθωθεί, θα ήταν η αύξηση των διαθέσιμων δεδομένων εκπαίδευσης, διατηρώντας τις υπάρχουσες υποδομές.

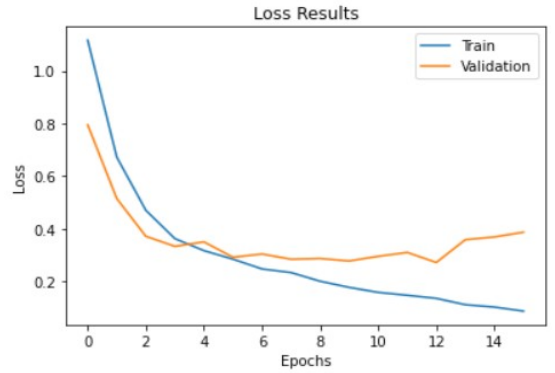
Ερώτημα 4

Δημιουργήθηκε όπως προβλέπεται μία συνάρτηση για την παραγωγή του μοντέλου, με τα ανάλογα συνελεκτικά και άλλα επίπεδα που ζητούνταν. Η έναρξη του προγράμματος, είναι παρόμοια με αυτή του πρώτου μοντέλου. Γίνεται λήψη των δεδομένων, καλούνται οι απαραίτητες βιβλιοθήκες και ύστερα εκτελείται το κομμάτι της επεξεργασίας των δεδομένων (ξανά με την χρήση της συνάρτησης prepare_datasets). Το μοντέλο, δημιουργείται μέσα από την απαραίτητη συνάρτηση που δημιουργήθηκε με το όνομα cnn2. Η συνάρτηση ακολουθεί πιστά τις οδηγίες της εκφώνησης και παράγει ένα μοντέλο με μεγαλύτερη επεξεργαστική ισχύ που αποφέρει καλύτερα αποτελέσματα και χωρίς να εμφανίζει έντονα το φαινόμενο της υπερχείλισης. Ο αριθμός των συνολικών παραμέτρων του μοντέλου είναι μία ένδειξη αυτού. Το δεύτερο μοντέλο έχει στην διάθεσή του 11.000.000 παραμέτρου, πολλές περισσότερες από το πρώτο. Αφού γίνει η σύνοψη του μοντέλου, με τις ίδιες παραμέτρους όπως και για το πρώτο (συνάρτηση βελτιστοποίησης Adam, απώλειας sparse categorical crossentropy,metric accuracy), το μοντέλο εκπαιδεύεται ξανά, με μέγιστο αριθμό εποχών τις 20 εποχές και μέγεθος batch τα 64 αντικείμενα. Παρακάτω παρουσιάζεται ενδεικτική δοκιμή:

```
Epoch 10/20
198/198 [==============================] - 90s 450ms/step - loss: 0.1763 - accuracy: 0.9351 - val_loss: 0.2769 - val_accuracy: 0.8991
Epoch 11/20
198/198 [==============================] - 90s 449ms/step - loss: 0.1573 - accuracy: 0.9416 - val_loss: 0.2946 - val_accuracy: 0.8935
Epoch 12/20
198/198 [==============================] - 90s 451ms/step - loss: 0.1463 - accuracy: 0.9467 - val_loss: 0.3093 - val_accuracy: 0.8878
Epoch 13/20
198/198 [==============================] - 90s 453ms/step - loss: 0.1346 - accuracy: 0.9504 - val_loss: 0.2712 - val_accuracy: 0.9027
Epoch 14/20
198/198 [==============================] - 90s 450ms/step - loss: 0.1104 - accuracy: 0.9576 - val_loss: 0.3579 - val_accuracy: 0.8984
Epoch 15/20
198/198 [==============================] - 90s 452ms/step - loss: 0.1014 - accuracy: 0.9626 - val_loss: 0.3682 - val_accuracy: 0.8868
Epoch 16/20
198/198 [==============================] - 90s 453ms/step - loss: 0.0861 - accuracy: 0.9673 - val_loss: 0.3863 - val_accuracy: 0.8797
```

Το μοντέλο σταμάτησε την εκπαίδευσή του στην 16 εποχή. Σε αυτό το σημείο φαίνεται ευκρινώς η early stopping λειτουργία που υπάρχει και σε αυτό το μοντέλο. Εξίσου σημαντικά, είναι τα αποτελέσματα που παράγει. Οι τιμές γα ακρίβεια / απώλεια των δύο συνόλων έχουν χαμηλή απόκλιση, επομένως το over fitting που έκανε έντονη την παρουσία του στο προηγούμενο μοντέλο, εδώ απουσιάζει. Η ακρίβεια συγκεκριμένα πλησιάζει την μονάδα με κάτω από 0,1 τιμή απώλειας. Οι τιμές αυτές μας κάνουν να πιστεύουμε πως το μοντέλο μπορεί να ελεγχθεί σε πραγματικά, άγνωστα για το ίδιο δεδομένα και να κατηγοριοποιήσει την εικόνα με εξαιρετικά ποσοστά επιτυχίας. Τα θετικά αυτά αποτελέσματα αποτυπώνονται και στις δύο παρακάτω εικόνες:






```
#Confusion matrix to see the errors for each category.
cm2, y_hat2 = confusion_matrix(model2, test_ds)
print(cm2)
#Evaluation function to get the accuracy and loss after test set.
results = model2.evaluate(x = test_ds, verbose = 0)
print("Test loss:", results[0])
print("Test accuracy:", results[1])
```

```
tf.Tensor(
[[ 648   15   13    2]
 [  47  993  155    1]
 [  87  187 1781    6]
 [   4    2   20  272]], shape=(4, 4), dtype=int32)
Test loss: 0.409177303314209
Test accuracy: 0.872667133808136
```

Τα αριστερά διαγράμματα επιβεβαιώνουν την υπόθεση σχετικά με το over fitting. Οι αντίστοιχες τιμές είναι κοντινές και πλησίον του ιδανικού. Ο πίνακας σύγχυσης επίσης αναδεικνύει τις δυνατότητες του μοντέλου. Σε αντιστοίχιση με την πρώτη κατηγορία του πρώτου μοντέλου εδώ για Covid-19 εικόνες, το μοντέλο έκρινε σωστά 648 εικόνες, 30 λάθος. Η ακρίβεια για την συγκεκριμένη κατηγορία διπλασιάστηκε, από το προηγούμενο με τις υπόλοιπες κατηγορίες επίσης σαφώς να βελτιώνονται. Τέλος, η συνάρτηση αξιολόγησης τοποθετεί την ακρίβεια κοντινότερα στο validation κυρίως λόγω των λιγότερων δεδομένων.

Απάντηση στην ερώτηση 3):

Το γεγονός ότι τα δύο μοντέλα παρουσιάζουν διαφορετικά αποτελέσματα, οφείλεται, αποκλειστικά, στα περισσότερα επίπεδα και φίλτρα που απαρτίζουν το δεύτερο μοντέλο. Έχει καλύτερη επίδοση και στα δύο σύνολα (εκπαίδευσης και δοκιμής),αφού παρατηρείται σχεδόν μέγιστη ακρίβεια, με ελάχιστη απώλεια, λόγω του μεγαλύτερου βάθους και επεξεργασίας που διαθέτει. Πιο αναλυτικά, έχει σαφώς περισσότερα συνελεκτικά επίπεδα, με επίσης ιδιαίτερα, μεγαλύτερο αριθμό φίλτρων. Ακόμα, το πλήρως συνδεδεμένο επίπεδο έχει κατά 32 φορές περισσότερους νευρώνες. Παρεπόμενο, είναι, να μειώθηκαν αρκετά οι λάθος εκτιμήσεις των εικόνων και άρα να αυξήθηκε η ακρίβεια.

Ερώτημα 5

Είναι απαραίτητο, να χρησιμοποιηθεί για το ερώτημα ένα προ-εκπαιδευμένο δίκτυο, που θα παράξει μοντέλο με υψηλό αριθμό επιπέδων, μέσω της υλοποιημένης κλάσης EfficientNetB0. Αυτή η κλάση δημιουργεί ένα μοντέλο, το οποίο αποτελείται από πολλαπλά επίπεδα, μεταξύ των οποίων υπάρχουν επίπεδα για batch normalization, dropout, επίπεδα ενεργοποίησης και άθροισης. Η βασική προϋπόθεση που είχε τεθεί, ήταν πως το μοντέλο αυτό θα πρέπει να εκπαιδευτεί σε 5 εποχές και με μικρότερο batch_size. Επιπροσθέτως, το μέγεθος της εικόνας έπρεπε να

μετατραπεί από τον αρχικό συνδυασμό (299,299) σε (224,224) και στα τρία datasets (train_ds, val_ds, test_ds), το οποίο έγινε με τη χρήση της συνάρτησης map σε συνδυασμό με lambda functions. Ύστερα, και πριν καλέσουμε το προ-εκπαιδευμένο δίκτυο, στην συνάρτηση pretrained του κώδικα, δημιουργούμε ένα επίπεδο, για την ολοκλήρωση του resizing των δεδομένων. Δεν τοποθετούνται βάρη μέσα στο δίκτυο και μόλις τελειώσει η λειτουργία της συνάρτησης, επιστρέφεται το μοντέλο.

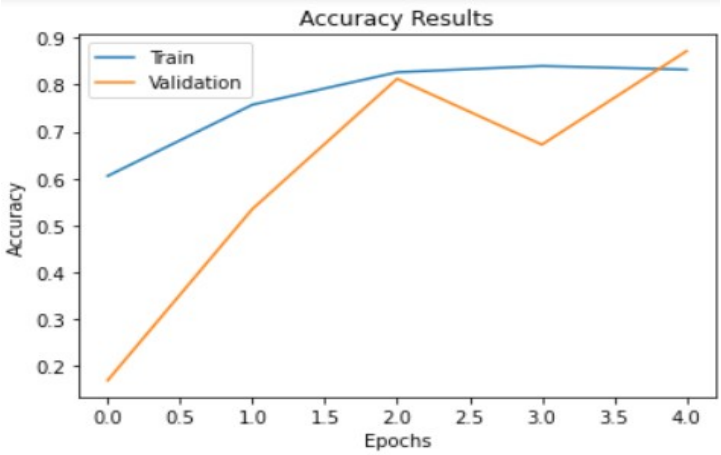
Η έναρξη του προγράμματος και η διαδικασία είναι η ίδια και το κομμάτι της εκπαίδευσης είναι και πάλι παρόμοιο με τα προηγούμενα. Δεν υπήρχε η πρόθεση να προστεθεί κάποιο παραπάνω επίπεδο στο μοντέλο. Άρα δεν υπάρχει το early stopping όπως στα προηγούμενα. Το compile & fit είναι ίδια με τα προηγούμενα και απομένει ο πίνακας σύγχυσης και η αποτύπωσή τους σε διαγράμματα. Ενδεικτική αποτύπωση δοκιμής παρακάτω:

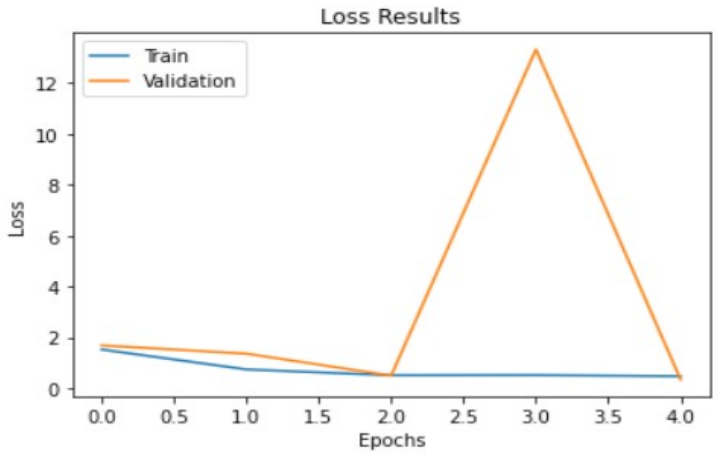
```
Epoch 1/5
198/198 [==============================] - 270s 1s/step - loss: 1.5255 - accuracy: 0.6052 - val_loss: 1.6866 - val_accuracy: 0.1688
Epoch 2/5
198/198 [==============================] - 247s 1s/step - loss: 0.7435 - accuracy: 0.7574 - val_loss: 1.3632 - val_accuracy: 0.5348
Epoch 3/5
198/198 [==============================] - 247s 1s/step - loss: 0.5143 - accuracy: 0.8267 - val_loss: 0.5009 - val_accuracy: 0.8127
Epoch 4/5
198/198 [==============================] - 247s 1s/step - loss: 0.5204 - accuracy: 0.8400 - val_loss: 13.3120 - val_accuracy: 0.6721
Epoch 5/5
198/198 [==============================] - 245s 1s/step - loss: 0.4782 - accuracy: 0.8323 - val_loss: 0.3423 - val_accuracy: 0.8719
```

Εκτελούνται όλες οι εποχές και παράγουν όπως φαίνεται παρόμοιο ίσως και καλύτερο αποτέλεσμα από τα προηγούμενα δύο μοντέλα. Ωστόσο, εμφανίζεται μεγάλη απόκλιση στις απώλειες των δύο συνόλων. Παρατηρούμε πως κατά την αξιολόγηση του συνόλου test, η απώλεια είναι μικρότερη και η ακρίβεια είναι παραπλήσια.

```
#Confusion matrix to see the errors for each category.
cm3, y_hat3 = confusion_matrix(model3, test_ds_3)
print(cm3, y_hat3)
#Evaluation function to get the accuracy and loss after test set.
results = model3.evaluate(x = test_ds_3, verbose = 0)
print("Test loss:", results[0])
print("Test accuracy:", results[1])
```

```
tf.Tensor(
[[ 543   39   91    5]
 [  28  915  253    0]
 [  39   97 1923    2]
 [   2    9   39  248]], shape=(4, 4), dtype=int32) tf.Tensor([2 2 1 ... 2 2 2], shape=(4233,), dtype=int64)
Test loss: 0.3683636784553528
Test accuracy: 0.8573116064071655
```





Ενδιαφέρον παρουσιάζουν και τα διαγράμματα. Η κατακόρυφη αύξηση της απώλειας του validation set προκαλεί εντύπωση, όμως το γενικό αποτέλεσμα που προκύπτει είναι κοντά στην απώλεια του train set. Αυτό, έχει ως αποτέλεσμα να μην είναι παρόν το πρόβλημα over-fitting Τέλος όπως αναφέρθηκε είναι εντυπωσιακό, το γεγονός, πως το μοντέλο πετυχαίνει υψηλή ακρίβεια που αγγίζει το 0.9 και απώλεια κάτω του 0.5, πάντα σε τόσο σύντομο χρονικό διάστημα εκπαίδευσης

Απάντηση στην ερώτηση 4): Το τελευταίο μοντέλο οδηγεί στα εξής συμπεράσματα: τα αποτελέσματα σε ακρίβεια και απώλειες που πετυχαίνει είναι υψηλότερα από αυτά του πρώτου μοντέλου και χωρίς να υπάρχει μέχρι το τέλος και της τελευταίας εποχής, ζήτημα στην υπερκάλυψη. Ο πίνακας σύγχυσης μοιάζει αρκετά σε αυτόν του δεύτερου μοντέλου, με θετικά αποτελέσματα. Τα εξαιρετικά περισσότερα επίπεδα, παρά το γεγονός ότι η επεξεργασία υλοποιείται με λιγότερες παραμέτρους, φαίνεται πως επαρκούν και ξεπερνούν την επίδοση των δύο προηγούμενων. Τα πολλαπλά συνελεκτικά επίπεδα σε συνδυασμό με την κανονικοποίηση των batches και τα συνεχή dropouts παρέχουν ευελιξία και ένα βάθος επεξεργασίας που μπορεί με λίγες επαναλήψεις να φτάσει σε υψηλά αποτελέσματα. Εκεί, που φαίνεται σε μικρό βαθμό η ελάχιστη αστοχία σε σύγκριση με το δεύτερο ειδικά μοντέλο είναι στην πρώτη κατηγορία εικόνων του πίνακα σύγχυσης και επίσης η αστάθεια που φαίνεται στα διαγράμματα δεν είναι το επιθυμητό.

Απάντηση στην ερώτηση 5): Τα περισσότερα σφάλματα εντοπίζονται στις κατηγορίες “Normal “ & “Lung Capacity” αντίστοιχα, μιας και έχουν τις περισσότερες εικόνες. Το μοντέλο έχει λάθος εκτίμηση, όταν υποθέτει ότι η εικόνα είναι φυσιολογική, ενώ στην πραγματικότητα ανήκει στην κατηγορία Lung Capacity. Ωστόσο, και το ανάποδο, φαίνεται πως συμβαίνει καθ’ όλη την διάρκεια. Οι υπόλοιπες δύο κατηγορίες έχουν αναλογικά μικρά ποσοστά σφάλματος, ειδικά η τελευταία κατηγορία “Viral Pneumonia” που έχει στο παραπάνω παράδειγμα 12 εικόνες που το μοντέλο είχε λάθος εκτίμηση και 286 σωστές. Κατά αντιστοιχία με το δεύτερο ερώτημα και εδώ το περιθώριο λάθους είναι φανερά, ελάχιστο.

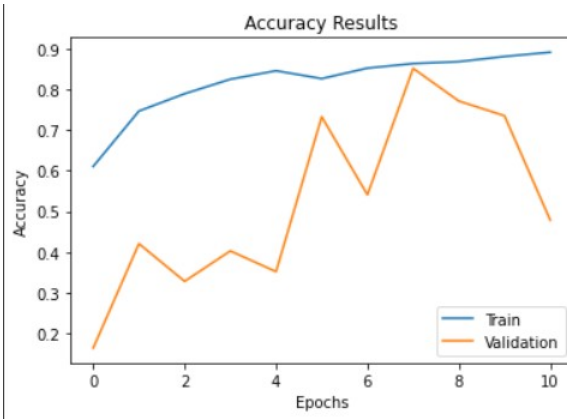
Ερώτημα 6.

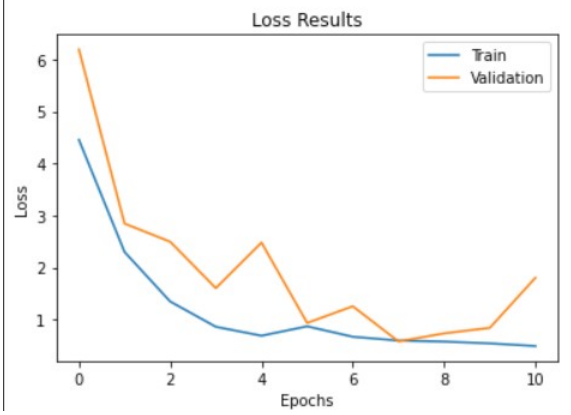
Για το τελευταίο ερώτημα ήταν απαραίτητο να δημιουργηθούν 3 συναρτήσεις. Ο συνδυασμός τους βγάζει ένα ResNet μοντέλο. Οι πρώτες δύο συναρτήσεις, υλοποιήθηκαν αποκλειστικά με τις οδηγίες της εκφώνησης. Έχουν συνολικά 8 και 9 επίπεδα αντίστοιχα. Η τρίτη συνάρτηση που αποβαίνει στο τελευταίο ερώτημα, καλείται να δημιουργήσει ένα Res Net μοντέλο βασιζόμενη στις άλλες δύο. Για την υλοποίηση αυτής της συνάρτησης χρησιμοποιήθηκε κώδικας από το διαδίκτυο. Ο κώδικας ακολουθεί το ResNet – 50 πρότυπο με πολλαπλά επίπεδα με υψηλό αριθμό φίλτρων. Έγινε παραμετροποίηση αυτού για να ταιριάζει στον κώδικα της εργασίας με μικρές αλλαγές. Αφού, λοιπόν, δηλωθούν οι συναρτήσεις αυτές το πρόγραμμα για τελευταία φορά εκτελείται όπως στα προηγούμενα. Φτιάχνονται τα απαραίτητα σύνολα, δημιουργείται το μοντέλο που διαθέτει την early stopping λειτουργία και έχει τον μεγαλύτερο αριθμό παραμέτρων από όλα τα μοντέλα.

Το μοντέλο πετυχαίνει εξαιρετικά καλές επιδόσεις για το train set αλλά όχι για validation και τελικά το test set. Η απώλεια που εμφανίζεται στο δεύτερο σύνολο είναι σημαντικά ασταθής και ξεπερνάει την μονάδα για αριθμό εποχών, με την ακρίβεια να κυμαίνεται σε σχετικά χαμηλές τιμές. Το συμπέρασμα που προκύπτει είναι, πως υπάρχει και μάλιστα έντονα το φαινόμενο του over fitting. Τεχνικές αντιμετώπισης μπορούν να διορθώσουν το φαινόμενο όχι όμως να το εξαλείψουν. Τέλος, παρακάτω παρουσιάζεται μία τελευταία ενδεικτική δοκιμή του τελευταίου μοντέλου.

```
Epoch 1/20
198/198 [==============================] - 103s 443ms/step - loss: 4.4569 - accuracy: 0.6104 - val_loss: 6.1915 - val_accuracy: 0.1645
Epoch 2/20
198/198 [==============================] - 89s 447ms/step - loss: 2.3007 - accuracy: 0.7462 - val_loss: 2.8458 - val_accuracy: 0.4207
Epoch 3/20
198/198 [==============================] - 90s 451ms/step - loss: 1.3487 - accuracy: 0.7887 - val_loss: 2.4969 - val_accuracy: 0.3284
Epoch 4/20
198/198 [==============================] - 90s 451ms/step - loss: 0.8630 - accuracy: 0.8243 - val_loss: 1.6061 - val_accuracy: 0.4029
Epoch 5/20
198/198 [==============================] - 90s 448ms/step - loss: 0.6917 - accuracy: 0.8452 - val_loss: 2.4813 - val_accuracy: 0.3520
Epoch 6/20
198/198 [==============================] - 90s 449ms/step - loss: 0.8722 - accuracy: 0.8258 - val_loss: 0.9369 - val_accuracy: 0.7322
Epoch 7/20
198/198 [==============================] - 89s 446ms/step - loss: 0.6716 - accuracy: 0.8516 - val_loss: 1.2575 - val_accuracy: 0.5407
Epoch 8/20
198/198 [==============================] - 90s 450ms/step - loss: 0.5985 - accuracy: 0.8628 - val_loss: 0.5819 - val_accuracy: 0.8511
Epoch 9/20
198/198 [==============================] - 89s 448ms/step - loss: 0.5787 - accuracy: 0.8675 - val_loss: 0.7365 - val_accuracy: 0.7708
Epoch 10/20
198/198 [==============================] - 90s 450ms/step - loss: 0.5443 - accuracy: 0.8802 - val_loss: 0.8427 - val_accuracy: 0.7346
Epoch 11/20
198/198 [==============================] - 89s 448ms/step - loss: 0.4936 - accuracy: 0.8906 - val_loss: 1.8055 - val_accuracy: 0.4785
```

Το μοντέλο εκτελέστηκε για 11 εποχές. Για ακόμα μία φορά γίνεται χρήση της διαδικασίας του Early Stopping. Η ακρίβεια στο σύνολο εκπαίδευσης είναι ιδιαίτερα ευοίωνη, όμως στις υπόλοιπες μετρικές φαίνεται πως το μοντέλο δυσκολεύεται να παράγει τα προσδοκώμενα αποτελέσματα. Στην εικόνα, παρουσιάζονται και οι έντονες αυξομοιώσεις στην απώλεια του συνόλου επικύρωσης. Σε αυτό το σημείο, είναι σημαντικό να αναφερθεί πως το πρόγραμμα δοκιμάστηκε για συνολικά 20 εποχές με μέγεθος batch 64. Ακόμα, έγινε η δοκιμή για rescaling των δεδομένων με το που εισέρχονται στο ResNet αλλά τα φαινόμενα που αναφέρθηκαν, έγιναν σημαντικά εντονότερα σε σημείο το validation loss να ξεπερνά τις 20 μονάδες.






```
tf.Tensor(
[[ 539   91   44    4]
 [ 468  691   35    2]
 [1062  416  577    6]
 [  22   19   30  227]], shape=(4, 4), dtype=int32)
Test loss: 1.8102318048477173
Test accuracy: 0.48051026463508606
```

Από τα τελευταία διαγράμματα και τον πίνακα σύγχυσης εντοπίζονται οι περιοχές στις οποίες δυσκολεύεται το μοντέλο περισσότερο. Με μία πρώτη ματιά παρατηρείται, πως το μοντέλο δεν καταφέρνει στα τελικά δεδομένα να παράγει υψηλή ακρίβεια, με μέσο όρο να κυμαίνεται στο 0.5. Τα περισσότερα λάθη, καθαρά εμφανίζονται στην κατηγορία “Normal” όπου μονάχα 577 κρίθηκαν σωστά από το μοντέλο. Τέλος στα διαγράμματα και ιδιαιτέρως στο διάγραμμα ακρίβειας φαίνεται η μεγάλη απόκλιση, το φαινόμενο της υπερχείλισης και η έντονη αποσταθεροποίηση των τιμών απώλειας και ακρίβειας.

Τα links για τα Google Colab αρχεία:

1. CNN1: https://colab.research.google.com/drive/1LUvP1bgZ5ZTKLp56xH4lkdMyHHS5ZacB?usp=sharing
2. CNN2: https://colab.research.google.com/drive/1tbRMALe6V95fHZ0h6RLgkV0W-X5FZUu_?usp=sharing
3. Pre_Trained: https://colab.research.google.com/drive/1PxhPCvwGy81q8YbUdcgDu8pEygdxZ268?usp=sharing
4. Skip_Connections: https://colab.research.google.com/drive/16CXUwS1OSFHgmL96V-uyLC83nZsXG484?usp=sharing