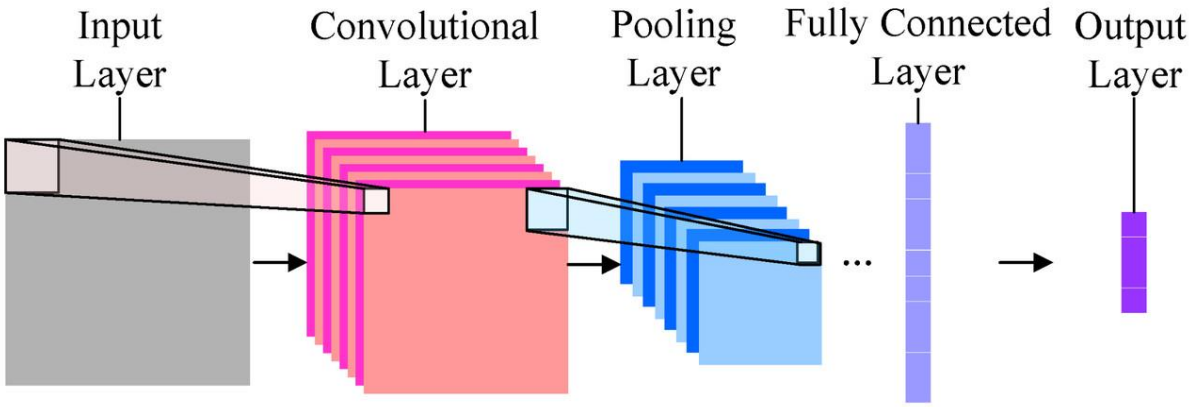
Τελικό Project στην Αναγνώριση προτύπων



Χατζησάββας Χρήστος

Ακαδημαϊκό έτος 2023-2024

ΔΗΜΟΚΡΙΤΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΡΑΚΗΣ
ΤΜΗΜΑ ΗΜ & ΜΥ

Μέρος 1ο:

A) Αρχικά, δημιουργούμε μια συνάρτηση `load_images` για να μπορέσουμε να φορτώσουμε τις εικόνες και να τις φέρουμε σε κατάλληλη μορφή ώστε να δημιουργηθεί το dataset. Αφού γίνει αυτό και δημιουργήσουμε και τα ανάλογα labels (1 αν υπάρχει μάσκα, 0 αν δεν υπάρχει) με την συνάρτηση `train_test_split` κάνουμε τους διαχωρισμούς για train,validation και test set με ποσοστό 60%, 20%, 20% αντίστοιχα.

B) Για την υλοποίηση της άσκησης αυτής θα αξιοποιηθούν τα συνελικτικά νευρωνικά δίκτυα (CNN) καθώς μπορούν πολύ αποδοτικά να εξάγουν χαρακτηριστικά από εικόνες και μαθαίνουν να αναγνωρίζουν πρότυπα. Συνεπώς αποτελούν μια πολύ καλή επιλογή για προβλήματα κατηγοριοποίησης σε εικόνες. Για τον λόγο αυτό θα γίνει μια σύντομη αναφορά στον τρόπο λειτουργίας τους. Ένα συνελικτικό δίκτυο αποτελείται από επίπεδα συνέλιξης τα οποία είναι σύνολα νευρώνων που εκτελούν συνέλιξη των δεδομένων εισόδου με προκαθορισμένα φίλτρα. Τα φίλτρα αυτά διατρέχουν τα δεδομένα εισόδου και επιστρέφουν έναν πίνακα που περιέχει τα εσωτερικά γινόμενα των δεδομένων που διατρέχονται κάθε φορά με το εκάστοτε φίλτρο. Ο πίνακας αυτός είναι ο χάρτης χαρακτηριστικών. Οι διαστάσεις των φίλτρων είναι Ύψος, Πλάτος και Βάθος με το βάθος να ταυτίζεται με αυτό των δεδομένων εισόδου. Τις περισσότερες φορές το μέγεθος του φίλτρου είναι μικρότερο από τις διαστάσεις που έχει η είσοδος. Μία άλλη βασική διεργασία που μπορεί να εκτελέσει ένα επίπεδο πέρα από την διαδικασία της συνέλιξης είναι η υποδειγματοληψία. Η μέθοδος αυτή αποσκοπεί στην μείωση των διαστάσεων εισόδου και εφαρμόζεται μεταξύ 2 συνελικτικών επιπέδων. Αξίζει να σημειωθεί πως συμβάλλει στην μείωση της πιθανότητας το μοντέλο να οδηγηθεί σε υπερπροσαρμογή (overfitting). Έπειτα από τα στάδια της συνέλιξης και της υποδειγματοληψίας γίνεται flatten στα δεδομένα που παράγονται ώστε να μπορέσουν να δοθούν ως είσοδος σε ένα fully connected layer προκειμένου να δημιουργηθεί η έξοδος του μοντέλου. Επιπρόσθετα μπορούν να χρησιμοποιηθούν και άλλες τεχνικές όπως το batch normalization που κάνει μια κανονικοποίηση στα δεδομένα του batch καθώς και το Dropout που απορρίπτει με μια πιθανότητα ένα σύνολο νευρώνων για την αποφυγή και πάλι του overfitting.

Εστιάζοντας τώρα στο μοντέλο που σχεδιάστηκε, αυτό αποτελείται από 2 συνελικτικά layers, 2 επίπεδα υποδειγματοληψίας (pooling) και από ένα fully connected layer που λαμβάνει ως είσοδο το αποτέλεσμα του τελευταίου pooling layer και έχει 2 νευρώνες/εξόδους για τις 2 κλάσεις. Αναλυτικότερα, στο πρώτο συνελικτικό επίπεδο γίνεται συνέλιξη με kernel size 5 και padding 2 ενώ από 3 κανάλια πηγαίνει στα 16. Στο αποτέλεσμα αυτό γίνεται max pooling προκειμένου να μειωθούν οι διαστάσεις με kernel size 2 και stride 2 (υποδιπλασιασμός). Ομοίως, το δεύτερο συνελικτικό επίπεδο, από τα 16 κανάλια πηγαίνει στα 32 διατηρώντας το ίδιο kernel size και padding με το πρώτο. Εφαρμόζουμε και πάλι max pooling 2x2 και οδηγούμαστε σε τελικό μέγεθος 8x8 με 32 κανάλια. Αφού γίνει flatten για να εισαχθεί στο fully connected layer, η έξοδος με τους δύο νευρώνες επιστρέφει το αποτέλεσμα του classification. Τέλος, αξίζει να σημειωθεί πως και στα δύο επίπεδα εφαρμόζεται batch normalization το οποίο εξασφαλίζει σταθερότητα και συμβάλει στην επιτάχυνση της εκπαίδευσης.

Την παραπάνω περιγραφή μπορούμε να την δούμε όπως την επιστρέφει η pytorch με την χρήση της συνάρτησης `parameters()`.

```
CNN(
  (layer1): Sequential(
    (0): Conv2d(3, 16, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2))
    (1): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): ReLU()
    (3): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  )
  (layer2): Sequential(
    (0): Conv2d(16, 32, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2))
    (1): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): ReLU()
    (3): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  )
  (fc): Linear(in_features=2048, out_features=2, bias=True)
  (flatten): Flatten(start_dim=1, end_dim=-1)
)
Number of learnable parameters' sets:  10
torch.Size([16, 3, 5, 5])
torch.Size([16])
torch.Size([16])
torch.Size([16])
torch.Size([32, 16, 5, 5])
torch.Size([32])
torch.Size([32])
torch.Size([32])
torch.Size([2, 2048])
torch.Size([2])
```

Πέρα από τις παραμέτρους του μοντέλου αξίζει να δούμε και το περιεχόμενο του dataset. Πιο συγκεκριμένα όλες οι εικόνες και από τις δύο κλάσεις έχουν μέγεθος 32x32. Ακολουθούν δύο παραδείγματα από εικόνες ένα για κάθε κλάση.

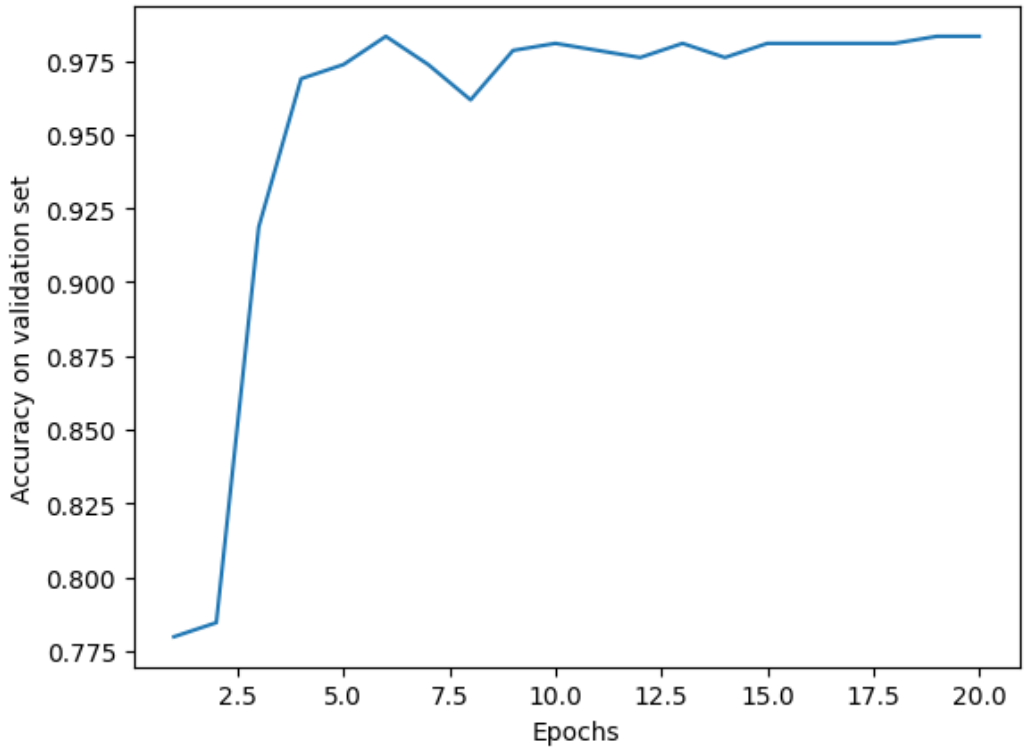
Γ) Στο παραπάνω μοντέλο προκειμένου να επιτευχθεί η καλύτερη ακρίβεια και το μικρότερο σφάλμα ταξινόμησης βασιζόμενοι στα αποτελέσματα του validation set έγιναν διαφορές τροποποιήσεις τόσο στην αρχιτεκτονική του δικτύου όσο και στα hyperparameters. Αρχικά, έγινε δοκιμή στο ίδιο μοντέλο η εκτέλεση χωρίς padding στα συνελικτικά layers και χωρίς καθόλου επίπεδα υποδειγματοληψίας ενώ προστέθηκε ένα ακόμη fully connected layer στο τέλος. Τα αποτελέσματα ήταν παρόμοια με του τελικού μοντέλου ακόμη και με αλλαγή του

kernel size και του stride. Γενικά, ο διαχωρισμός των 2 κλάσεων γίνεται αρκετά εύκολα και οποιοδήποτε **απλό συνελικτικό δίκτυο*** μπορεί να επιτύχει υψηλό ποσοστό ακρίβειας. Επιπλέον, για να καταλήξουμε στο τελικό μοντέλο έγιναν διάφορες παραλλαγές στο learning rate και στα μεγέθη των batch και καταλήξαμε πως οι καλύτερες παράμετροι έπειτα από δοκιμές είναι: learning rate: 0.001 (10^-3), optimizer = Adam, loss function: CrossEntropyLoss (αφού έχουμε πρόβλημα classification) και για τα batch size έχουμε 100, 1, 32 για τα train ,validation, test set αντίστοιχα.

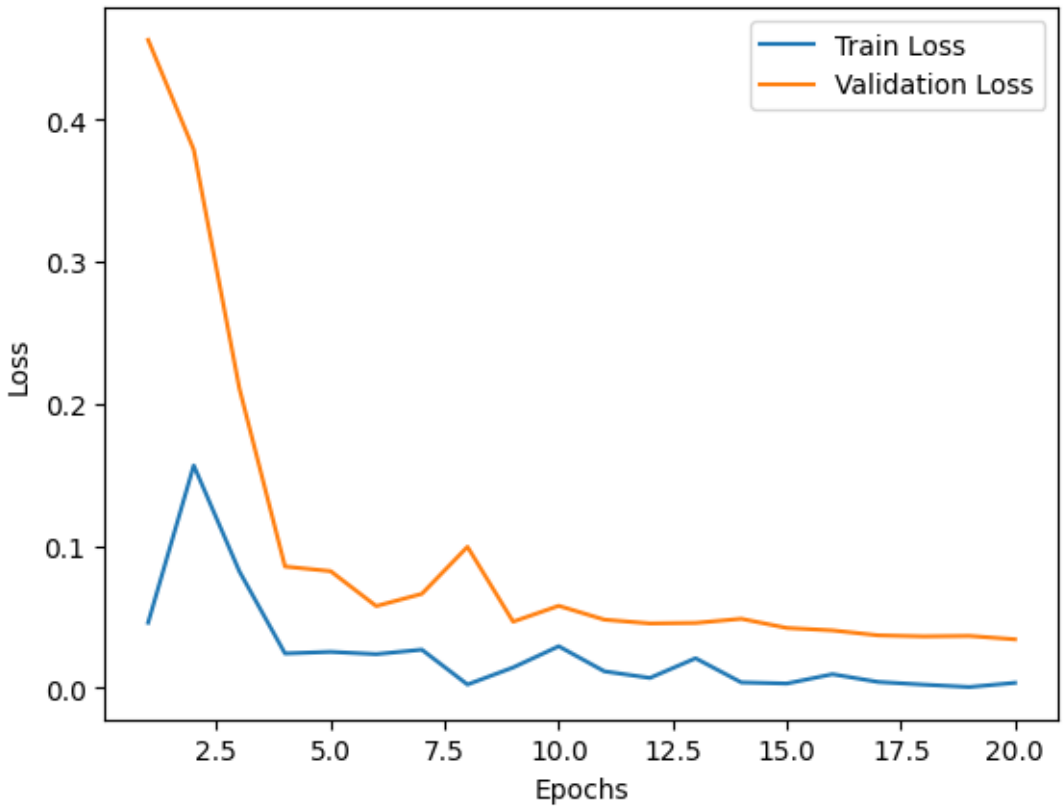
Στο test set παρατηρήθηκε μέσο σφάλμα 0.063 και ακρίβεια 98.6%. Παρατηρούμε επίσης πως το accuracy που πετυχαίνει το μοντέλο στο validation set είναι κοντά σε αυτό που δίνει το μοντέλο στο test set γεγονός που αποτελεί ένδειξη πως το μοντέλο δεν εξειδικεύεται στα δεδομένα εκπαίδευσης και κατορθώνει να κάνει γενίκευση.

```
Test set
Accuracy: 98.6%, Avg loss: 0.063000
```

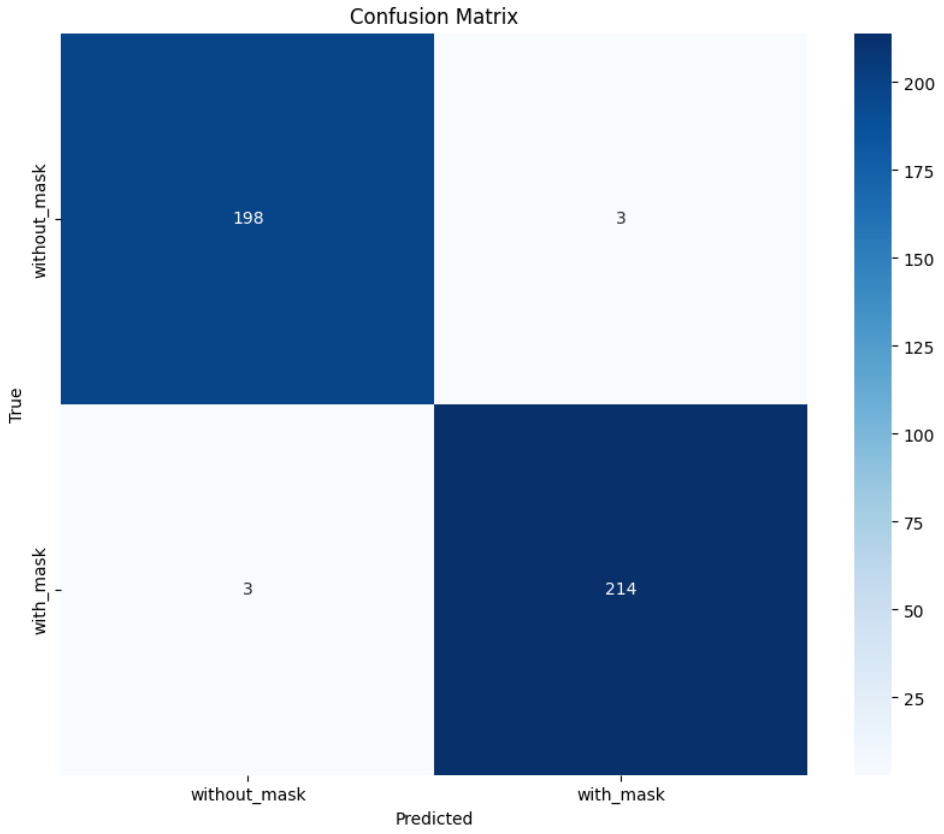
Το διάγραμμα της ακρίβειας για το validation set συναρτήσει των εποχών ακολουθεί:



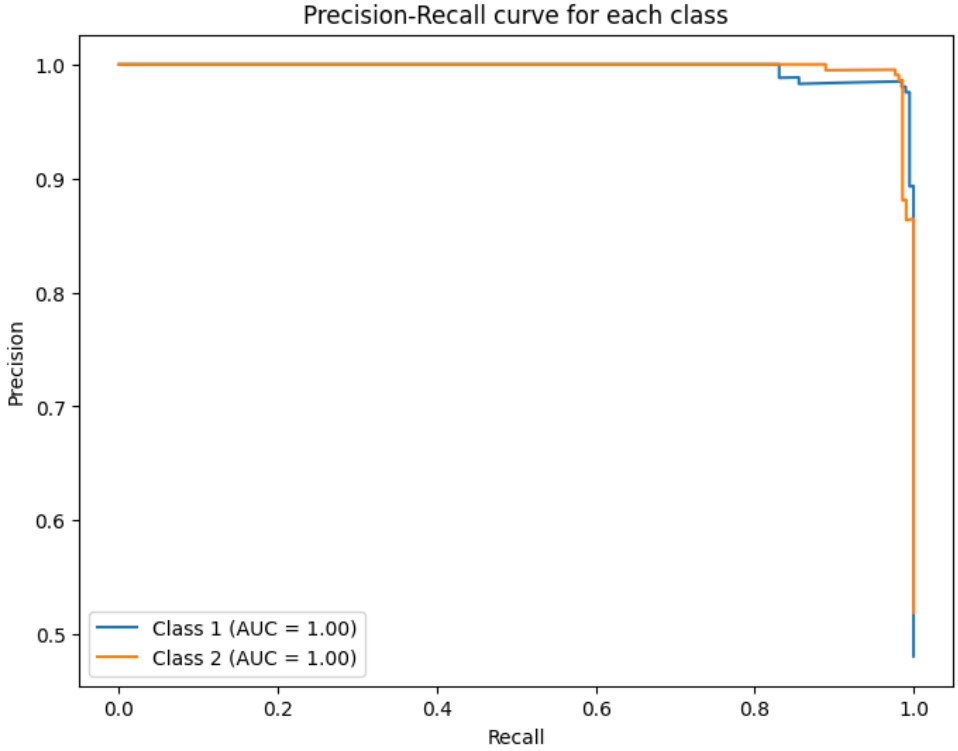
Επιπρόσθετα, γίνεται παράθεση και του διαγράμματος που απεικονίζει την εξέλιξη του loss στην διάρκεια των εποχών για το training και το validation. Παρατηρούμε πως το error και για τις δύο καμπύλες πέφτει έως ένα σημείο και μετά μένει σταθερό διατηρώντας ένα μικρό «κενό» μεταξύ των καμπυλών. Αυτό μας οδηγεί στο συμπέρασμα πως το μοντέλο δεν κάνει overfitting.



Για την καλύτερη οπτικοποίηση των αποτελεσμάτων μπορούμε να κάνουμε plot και τον confusion matrix όπου δείχνει πόσες λάθος προβλέψεις έγιναν για τα δεδομένα του testing. Παρατηρούμε πως 6 συνολικά δείγματα έχουν ταξινομηθεί λανθασμένα από συνολικά 418.



Οι καμπύλες Precision – Recall και η τιμή για το AUC φαίνονται στο παρακάτω διάγραμμα. Η υψηλή ακρίβεια του μοντέλου αιτιολογεί και την μορφή των καμπυλών καθώς ομοιάζουν με την συμπεριφορά ενός ιδανικού ταξινομητή.



* ***Σημείωση:*** Στο συμπιεσμένο αρχείο της εργασίας βρίσκεται και ο κώδικας για ένα δοκιμαστικό δίκτυο της άσκησης αυτής που περιέχει ένα συνελικτικό επίπεδο και 2 fully connected επίπεδα στην έξοδο.

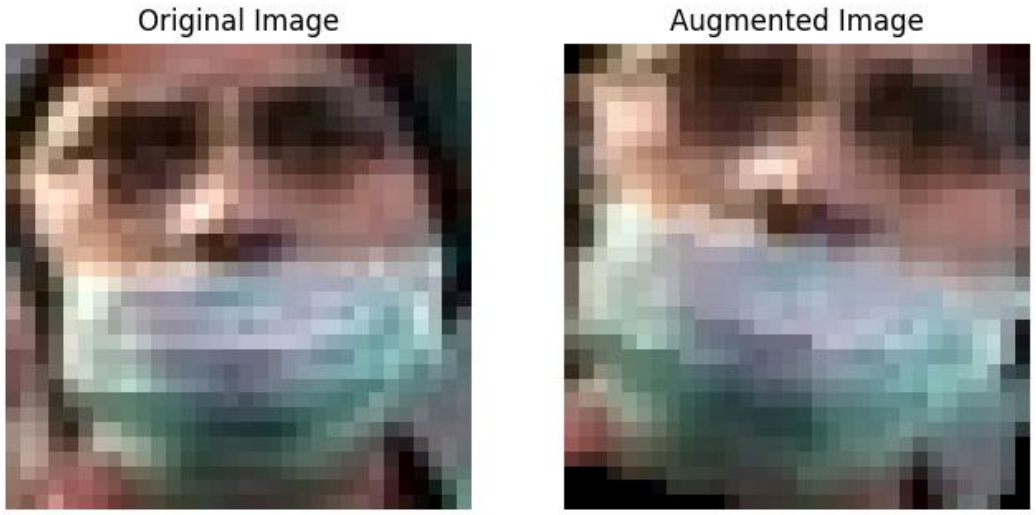
Δ) Δημιουργώντας ένα dataset με τα δεδομένα που έχουν λανθασμένη χρήση μάσκας μπορούμε να δούμε από την εκτέλεση του αντίστοιχου κώδικα ότι το ποσοστό των εικόνων που εντοπίζονται με χρήση μάσκας είναι 85.7%. Το ποσοστό αυτό είναι αρκετά μεγάλο και δεν αντικατοπτρίζει την πραγματική κλάση των δεδομένων.

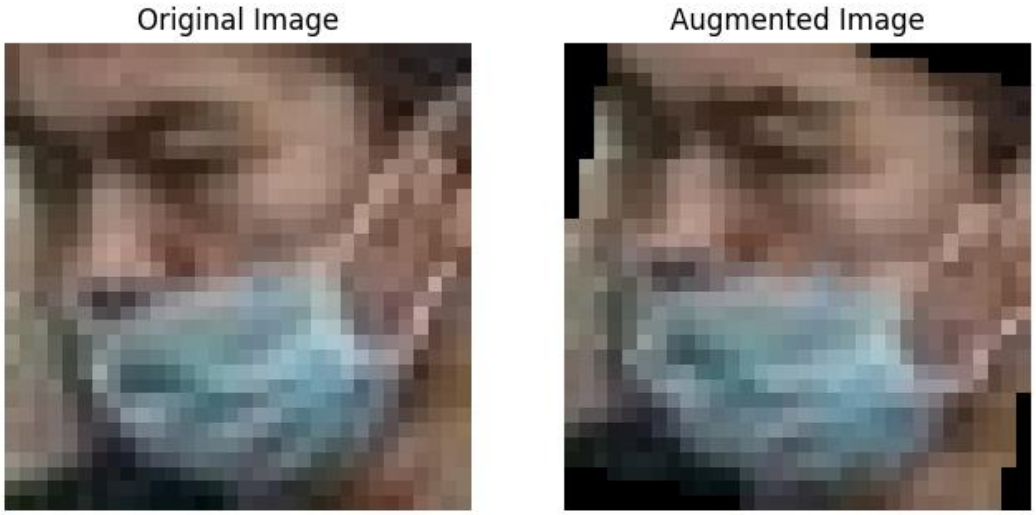
```
Incorrect mask usage recognized as with mask
Accuracy: 85.7%, Avg loss: 0.746954
```

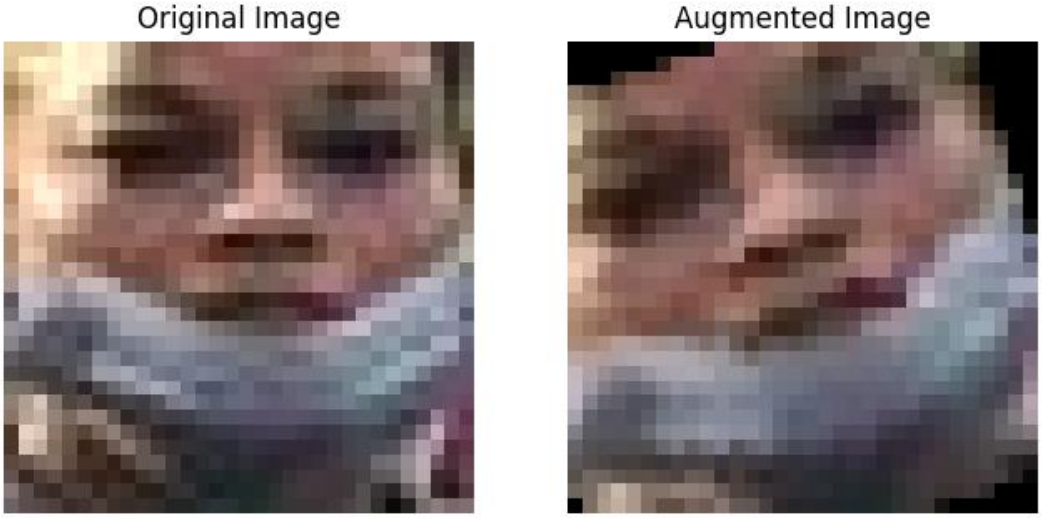
Προκειμένου να αντιμετωπίσουμε αυτό το φαινόμενο μια λύση είναι να κάνουμε data augmentation στις ήδη υπάρχουσες εικόνες χωρίς να δημιουργήσουμε νέες. Πιο συγκεκριμένα εφαρμόζοντας κάποιες τεχνικές επεξεργασίας εικόνας όπως περιστροφή και crop (`RandomRotation`, `RandomResizedCrop`) φαίνεται να μας δίνουν ένα ικανοποιητικό αποτέλεσμα καθώς πλέον το ποσοστό των εικόνων που αναγνωρίζονται με χρήση μάσκας είναι μειωμένο. Έτσι, από την εκτέλεση του κώδικα για το test στις εικόνες αυτές έχουμε:

```
With data augmentation
Accuracy: 48.2%, Avg loss: 0.726437
```

Δεδομένου ότι έχουμε χρησιμοποιήσει τυχαιότητα στην προ-επεξεργασία των εικόνων το αποτέλεσμα ενδέχεται να ποικίλει. Ακολουθούν μερικά παραδείγματα που συγκρίνουν την original με την επεξεργασμένη εικόνα.







Μια διαφορετική προσέγγιση για να επιτευχθεί μείωση του ποσοστού των εικόνων με λανθασμένη χρήση μάσκας που αναγνωρίζονται με χρήση μάσκας είναι η αλλαγή του κατωφλιού της πιθανότητας για το πότε θα θεωρείται ένα δείγμα ως κλάση 0 ή ως κλάση 1. Πιο συγκεκριμένα, πρέπει να γίνει τροποποίηση της συνάρτησης που κάνει το test (`test_loop`) έτσι ώστε αφού υπολογίσει την πιθανότητα να ανήκει σε κάθε μια κλάση να κάνει ανάθεση label όταν αυτή η πιθανότητα είναι μεγαλύτερη από κάποιο threshold . Με τον τρόπο αυτό θα εμφανίζει μεγαλύτερη ευαισθησία στο πότε θα επιλέξει ένα δείγμα να ανήκει στην κλάση 1 (κλάση με χρήση μάσκας).

Εφαρμόζοντας ένα πολύ υψηλό threshold παρατηρούμε πως για το test set το accuracy πέφτει από 98.6% σε 90.4% αλλά μειώνεται και το ποσοστό των δειγμάτων που έχουν λανθασμένη χρήση μάσκας και αναγνωρίζονται με χρήση μάσκας. Αναλυτικότερα, όπως παρατηρούμε στα αποτελέσματα που ακολουθούν, από 85.7% που αναγνωριζόταν εσφαλμένα το ποσοστό έχει μειωθεί στο 55.4% χωρίς να έχουμε μεγάλη απόκλιση στα δεδομένα του test set.

```
Test set accuracy when using the new threshold
Accuracy: 90.4%, Avg loss: 0.066242

Incorrect mask usage recognized as with mask
Accuracy: 55.4%, Avg loss: 0.746954
```

Μέρος 2ο:

Α) Για την άσκηση αυτή, προκειμένου να κατηγοριοποιήσουμε ποια μόρια μπορούν να προσδεθούν ή όχι αποτελεσματικά στον υποδοχέα θα χρησιμοποιηθούν random forests. Η επιλογή αυτή έγινε καθώς το συγκεκριμένο dataset διαθέτει πάρα πολλά χαρακτηριστικά (πολύ περισσότερα από τα δείγματα) και δεν είναι αποδοτική η προσέγγιση με απλό Fully Connected Neural Network. Για τον λόγο αυτό, πριν την εφαρμογή των δεδομένων στο forest πραγματοποιούμε ανάλυση κύριων συνιστωσών (PCA) προκειμένου να μειωθούν οι διαστάσεις διατηρώντας ένα ποσοστό επιθυμητής τυπικής απόκλισης. Αξίζει να σημειωθεί επίσης πως τα random forests μπορούν να διαχειριστούν την ύπαρξη ανισορροπίας μεταξύ των δεδομένων των κλάσεων καθώς στην περίπτωση αυτή τα δείγματα που αντιστοιχούν στην κλάση 1 είναι περισσότερα.

Τα Random Forests αποτελούν ένα σύνολο από δέντρα αποφάσεων (Decision Trees), συνδυάζοντας τα πλεονεκτήματα των δέντρων με την τυχαιότητα για τη βελτίωση και τη γενίκευση του μοντέλου. Κάθε δέντρο εκπαιδεύεται σε ένα τυχαίο υποσύνολο δεδομένων, είτε με επανατοποθέτηση δειγμάτων είτε χωρίς (bootstrapped samples) και χρησιμοποιεί ένα τυχαίο υποσύνολο χαρακτηριστικών ώστε να υπολογίσει την έξοδο Στην συνέχεια, τα αποτελέσματα των δέντρων συνδυάζονται και λαμβάνεται η τελική απόφαση, συνήθως μέσω majority vote όταν πρόκειται για προβλήματα κατηγοριοποίησης. Η ύπαρξη τυχαιότητας στην διαδικασία της εκπαίδευσης αλλά και κατά την διάρκεια των predictions οδηγεί σε μειωμένο κίνδυνο overfitting και καθιστά τα random forests ανθεκτικά σε δεδομένα με θόρυβο.

Αναφορικά με την PCA, είναι μια μέθοδος που έχει ως στόχο την εύρεση της κατεύθυνσης της μέγιστης τυπικής απόκλισης στο dataset. Πιο συγκεκριμένα, κεντράρει τα δεδομένα γύρω από το μέσο και έπειτα βρίσκει τα ιδιοδιανύσματα και τις ιδιοτιμές του πίνακα συνδιασποράς. Τα ιδιοδιανύσματα αναπαριστούν την κατεύθυνση της μέγιστης τυπικής απόκλισης ενώ οι ιδιοτιμές το μέγεθος της τυπικής απόκλισης που περιγράφει το κάθε ιδιοδιάνυσμα. Ο αριθμός των principal components που κρατάμε εξαρτάται από την τιμή της τυπικής απόκλισης που θέλουμε να διατηρήσουμε.

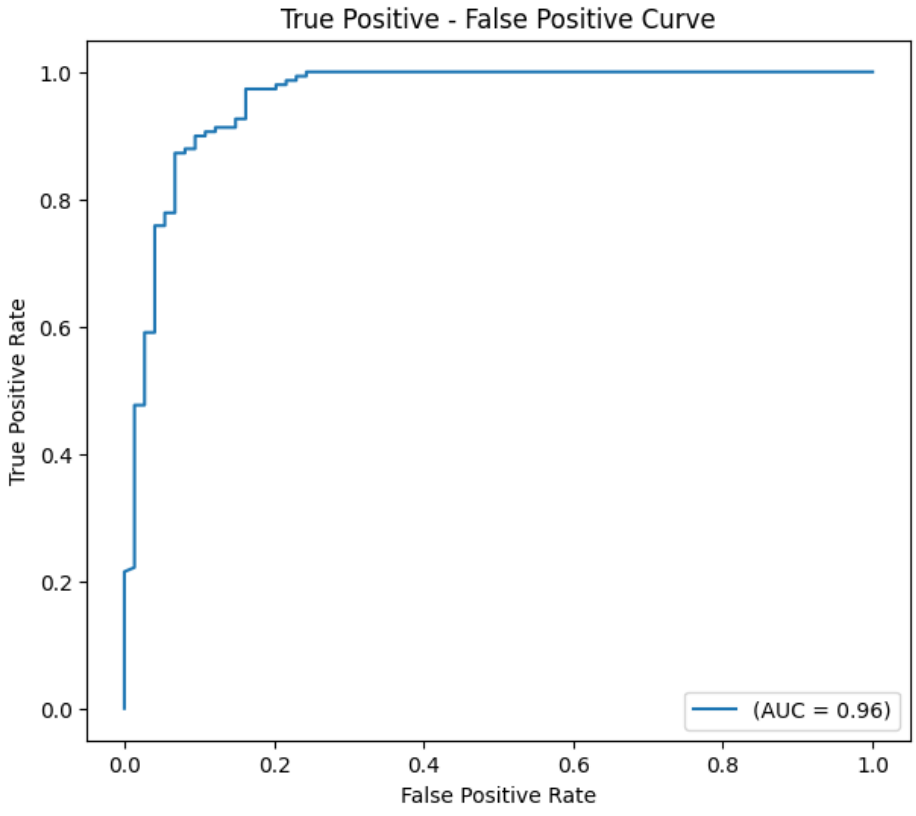
Β) Ξεκινώντας την υλοποίηση του μοντέλου αρχικά εισάγουμε τα δεδομένα και τα αποθηκεύουμε σε πίνακες. Δεδομένου ότι δεν υπάρχει διαφορετικό σύνολο δεδομένων για validation θα χωρίσουμε με την συνάρτηση `train_test_split` από το σύνολο των δεδομένων εκπαίδευσης το 20% για σκοπούς επικύρωσης. Στην συνέχεια θα γίνει κανονικοποίηση στα δεδομένα ώστε να αποκτήσουν μηδενική μέση τιμή και μοναδιαία τυπική απόκλιση. Στην συνέχεια κάνουμε fit και transform τα δεδομένα με την χρήση της PCA ώστε να υπολογιστεί το explained variance και κατά συνέπεια το cumulative explained variance. Με αυτόν τον τρόπο θέτοντας ως παράμετρο το ποσοστό της τυπικής απόκλισης που θέλουμε να διατηρήσουμε βρίσκουμε πόσα components πρέπει να κρατήσουμε. Έχοντας δηλώσει πως πρέπει να διατηρηθεί το 99% της τυπικής απόκλισης η εκτέλεση του κώδικα επιστρέφει πως χρειάζονται 333 principal components.

```
Number of components to retain 99.0% of variance is: 333
```

Αφού έχει ευρεθεί η βασική παράμετρος για την μέθοδο PCA, πρέπει να βρεθούν και οι παράμετροι για το random forest. Για τον σκοπό αυτό, ορίζουμε ένα πλαίσιο παραμέτρων για το πλήθος των δέντρων του forest και το βάθος. Με την χρήση της συνάρτησης `GridSearchCV,` και δίνοντας ως όρισμα τον classifier, το πλαίσιο των hyperparameters, τον αριθμό του KFolds για το cross validation και την μέθοδο μέτρησης της απόδοσης. Το μοντέλο θα αξιοποιήσει τα training δεδομένα προκειμένου να βρει τις καλύτερες παραμέτρους. Έπειτα δημιουργούμε έναν νέο ταξινομητή εφαρμόζοντας τις παραμέτρους αυτές και κάνουμε fit τα δεδομένα εκπαίδευσης. Τέλος, για την αξιολόγηση της απόδοσης του χρησιμοποιούμε το validation set που χωρίστηκε στην αρχή αφού πρώτα κάνουμε normalize τα δεδομένα. Από την εκτέλεση του κώδικα προκύπτει:

```
Best hyperparameters: {'max_depth': 35, 'n_estimators': 250}
Validation accuracy with best Hyperparameters: 0.9192825112107623
Error in validation set:  0.08071748878923768
```

Γ) Δεδομένου ότι η κατανομή κλάσεων στο test set είναι παρόμοια με αυτήν των δεδομένων εκπαίδευσης, το αναμενόμενο σφάλμα στο σύνολο δοκιμής θα είναι περίπου όσο το error στο σύνολο επικύρωσης. Όσον αφορά την καμπύλη TP-FP (True Positive – False Positive), αυτή προκύπτει με την χρήση της συνάρτησης `roc_curve` όπου δίνοντας ως ορίσματα τα labels και το prediction του μοντέλου επιστρέφει τις τιμές του false positive rate και του true positive rate. Επιπρόσθετα, για τον υπολογισμό της τιμής του AUC, δίνοντας ως ορίσματα τις παραπάνω τιμές στην συνάρτηση `auc` επιστρέφεται το ζητούμενο. Η ζητούμενη καμπύλη ακολουθεί.



Παρατηρούμε πως το παραπάνω διάγραμμα είναι πολύ κοντά στο ιδανικό διάγραμμα TP-FP κάτι που υποδηλώνει πως ο ταξινομητής έχει υψηλή ικανότητα διάκρισης μεταξύ των κλάσεων. Επομένως και η τιμή του AUC είναι λογικό να είναι ψηλή καθώς παριστάνει το εμβαδόν της καμπύλης.

Τέλος, για να δημιουργηθεί το αρχείο `test_predictions.csv` δημιουργήθηκε η συνάρτηση `save_predictions` όπου αφού κανονικοποιήσει τα δεδομένα για να γίνει μείωση διαστάσεων υπολογίζει την πιθανότητα κάθε δείγματος να ανήκει σε κάθε μια κλάση. Έπειτα, από την τιμή της πιθανότητας γίνεται εξαγωγή και του label και αυτά αποθηκεύονται σε ένα νέο csv αρχείο όπου κάθε γραμμή περιέχει πρώτα το label και μετά την πιθανότητα πρόσδεσης.

Ακολουθεί ένα παράδειγμα από το πως απεικονίζει το αρχείο τα δεδομένα.

predicted_label	prediction_score
1.0	0.5898
0.0	0.7325
1.0	0.8222
0.0	0.6135
0.0	0.7722
1.0	0.7050
0.0	0.7575
0.0	0.7535
0.0	0.8217
1.0	0.9700
1.0	0.7453
1.0	0.5714
1.0	0.8538
0.0	0.6293
1.0	0.9025
1.0	0.9200
1.0	0.9900
1.0	0.7842
1.0	0.7879
1.0	0.5058
1.0	0.8505
1.0	0.9450
1.0	0.6991
1.0	0.5250
0.0	0.7254