ΟΡΑΣΗ ΥΠΟΛΟΓΙΣΤΩΝ - ΕΡΓΑΣΙΑ 4η

Report//ΚΟΚΚΟΡΟΣ ΙΩΑΝΝΗΣ 57090

ΘΕΜΑ

Ζητείται να υλοποιηθεί αρχιτεκτονική συνελικτικού δικτύου σε Python με τη χρήση της βιβλιοθήκης Keras-Tensorflow,που θα αφορά στο πρόβλημα της ταξινόμησης πολλαπλών κλάσεων (multi-classclassification). Οι υποβολές των υλοποιήσεων θα γίνονται με τη μορφή Jupyter Notebook ενώ για την ανάπτυξη και δοκιμή θα χρησιμοποιηθεί η πλατφόρμα Google Colab. Στα πλαίσια της εργασίας θα πρέπει να υλοποιηθούν και να υποβληθούν ΔΥΟ αρχιτεκτονικές:

1.Ένα ΜΗ προ-εκπαιδευμένο δίκτυο το οποίο θα δημιουργηθεί αποκλειστικά για το τρέχον πρόβλημα ταξινόμησης, με τη διαδικασία που έχει αποτυπωθεί κατά το εργαστήριο.

2.Ένα προ-εκπαιδευμένο δίκτυο της επιλογής σας, που θα συνοδεύεται και από την αντίστοιχη ανάλυση του δικτύου που επιλέχθηκε. Στον παρακάτω σύνδεσμο υπάρχει η λίστα με τα προ-εκπαιδευμένα δίκτυα που είναι διαθέσιμα στο Keras, καθώς και σύνδεσμοι προς τα αντίστοιχα papers και παραδείγματα: Keras Applications

Κάθε μια από τις αρχιτεκτονικές θα πρέπει να συνοδεύεται από πλήρη περιγραφή της αρχιτεκτονικής και των επιπέδων που χρησιμοποιούνται. Επιπλέον, θα πρέπει να παρέχεται ποσοτική εκτίμηση της επίδοσής τους στο σύνολο δοκιμής, ως το ποσοστό επιτυχίας ταξινόμησης (accuracy). Τέλος, θα πρέπει να περιγράφεται η διαδικασία εκπαίδευσης (εποχές, batchsize, callbacks, προ-επεξεργασία , data augmentation, learning rate schedule).

Μέρος 1

Για το μη εκπαιδευμένο δίκτυο μετά από δοκιμές χρησιμοποίησα την αρχιτεκτονική conv-conv-conv-pool- conv-conv-conv-pool- conv-conv-conv-pool με padding=same για σταθερή διατήρηση του μεγέθους.

Η βέλτιστη απόδοση ήταν η παρακάτω:

```
Found 2149 images belonging to 34 classes.
215/215 [==============================] - 1s 4ms/step - loss: 0.2462 - accuracy: 0.9372
```

Έγινε σταδιακή αύξηση του Batch_Size από 20 μέχρι 100 με καλύτερα αποτελέσματα έχοντας την τιμή 80 η οποία και προτιμήθηκε.

Προτιμήθηκαν 70 εποχές καθώς οι 30 ήταν πολύ λίγες για αποτελεσματική προπόνηση του δικτύου και οι 100 υπερ-αρκετές αφού σε κανένα τεστ δεν ξεπεράστηκαν οι 80, επίσης έχουμε patience=15 στο early_stop_callback το οποίο μετά από δοκιμές κρίθηκε το ιδανικό μέγεθος. Steps_per_epoch=30 ώστε να χωρέσουν όσο το δυνατόν περισσότερα δείγματα σε κάθε εποχή.

Προ-επεξεργασία πραγματοποιήθηκε κατά την διάρκεια του ImageDataGenerator όπου έγινε rescale των train_dir και validation_dir ώστε να περιέχουν τιμές μεταξύ 0 και 1.

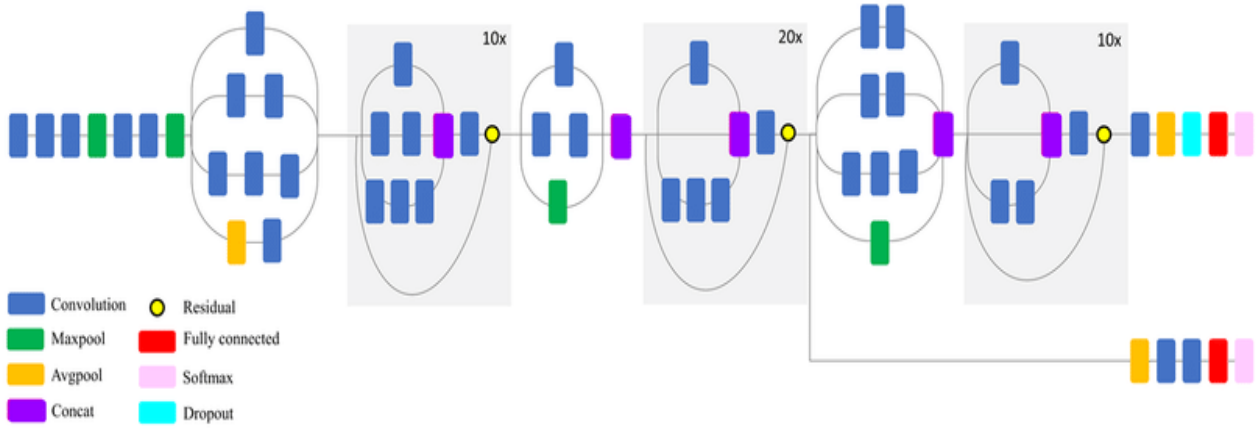
Χρησιμοποιήθηκαν τα παρακάτω augmentations με τις εξής τιμές:

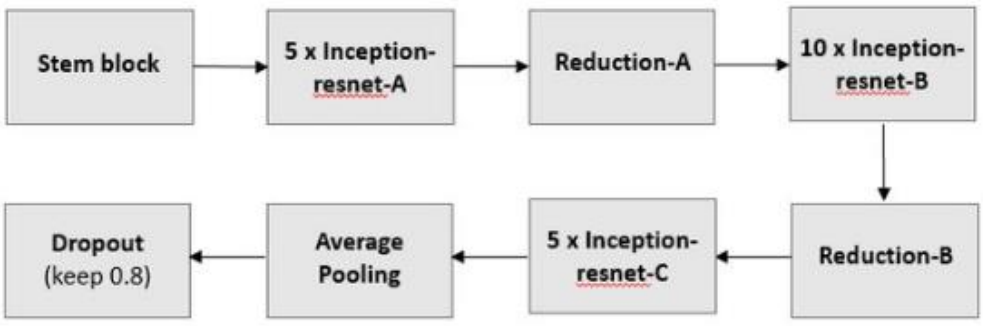
`rotation_range=90`: Περιστρέφει τυχαία το δείγμα από 0 έως 90 μοίρες.
`zoom_range=0.15`: Ζουμάρει τυχαία το δείγμα από 0 μέχρι 0.15.
`width_shift_range=shift`: Μετακινεί τυχαία το δείγμα στον άξονα χ κατά μια τιμή shift, προτιμήθηκε η τιμή 0.03 αφού έδωσε τα βέλτιστα αποτελέσματα.
`height_shift_range=shift`: Μετακινεί τυχαία το δείγμα στον άξονα y κατά μια τιμή shift, προτιμήθηκε η τιμή 0.03 αφού έδωσε τα βέλτιστα αποτελέσματα.
Έγινε προσπάθεια να προστεθούν περεταίρω augmentations αλλά είχαν αρνητική επίδραση στο ποσοστό επιτυχίας οπότε και απορρίφθηκαν.

Τέλος προτιμήθηκαν τα ModelCheckpoint και EarlyStopping callbacks καθώς κρίθηκε ότι η λειτουργία τους επαρκή για την αποτελεσματική προπόνηση του δικτύου.

Μέρος 2

Για το εκπαιδευμένο δίκτυο χρησιμοποιήθηκε η αρχιτεκτονική του InceptionResNetV2 το οποίο έχει την εξής μορφή:





The basic architecture of Inception-Resnet-v2.

https://medium.com/@zahraelhamraoui1997/inceptionresnetv2-simple-introduction-9a2000edcdb6

Το συγκεκριμένο δίκτυο έχει 164 layers βάθος και μπορεί να κατηγοριοποιήσει τα αντικείμενα σε 1000 κατηγορίες.

Η βέλτιστη απόδοση ήταν η παρακάτω:

Πρόσθεσα ένα layer dropout=0.5 για αποφυγή του overfitting.
Batch_size=80 γιατί τα αποτελέσματα μετά τα τεστ ήταν ικανοποιητικά.
Το προ-εκπαιδευμένο δίκτυο διαθέτει 30 εποχές ενώ σε κανένα τεστ δεν ολοκλήρωσε και τις 30 ,patience = 5 στο early_stop_callback.

Προ-επεξεργασία πραγματοποιήθηκε κατά την διάρκεια του ImageDataGenerator όπου έγινε rescale των train_dir και validation_dir ώστε να περιέχουν τιμές μεταξύ 0 και 1.

Χρησιμοποιήθηκαν τα παρακάτω augmentations με τις εξής τιμές:

`rotation_range=90`: Περιστρέφει τυχαία το δείγμα από 0 έως 90 μοίρες.
`zoom_range=0.3`: Ζουμάρει τυχαία το δείγμα από 0 μέχρι 0.15.
`width_shift_range=shift`: Μετακινεί τυχαία το δείγμα στον άξονα x κατά μια τιμή shift, προτιμήθηκε η τιμή 0.03 αφού έδωσε τα βέλτιστα αποτελέσματα.
`height_shift_range=shift`: Μετακινεί τυχαία το δείγμα στον άξονα y κατά μια τιμή shift, προτιμήθηκε η τιμή 0.03 αφού έδωσε τα βέλτιστα αποτελέσματα.
`horizontal flip=True`: Τυχαία flips στον οριζόντιο άξονα
`vertical flip=True`: Τυχαία flips στον κάθετο άξονα

Τέλος προτιμήθηκαν τα ModelCheckpoint και EarlyStopping callbacks καθώς κρίθηκε ότι η λειτουργία τους επαρκή για την αποτελεσματική προπόνηση του δικτύου.

Παρατηρήσεις
Σημαντική η διαφορά στον χρόνο εκτέλεσης μεταξύ του μη προ-εκπαιδευμένου με το προ-εκπαιδευμένο δίκτυο τόσο στην εκπαίδευση όσο και στο τεστ