Θεσσαλονίκη, 04/02/2018

**Υπολογιστική Νοημοσύνη- Συστήματα Εμπνευσμένα από τη Βιολογία**

**Προηγμένα Συστήματα Υπολογιστών & Επικοινωνιών**

Πασιοπούλου Ιωάννα (396)

Ταμπάκη Ειρήνη- Μαρία (401)

**Α. Εισαγωγή- Ορισμοί**

Το Keras είναι ένα υψηλού επιπέδου API νευρωνικών δικτύων, γραμμένο σε Python. Είναι ικανό να τρέχει πάνω από το TensorFlow, το CNTK ή το Theano. Αναπτύχθηκε με έμφαση στον ταχύ πειραματισμό και στην εμπεριστατωμένη έρευνα, παρέχοντας την δυνατότητα εξαγωγής αποτελεσμάτων με την ελάχιστη δυνατή καθυστέρηση.

Το Keras είναι μια βιβλιοθήκη που χρησιμοποιείται για Keras είναι API υψηλού επιπέδου νευρωνικών δικτύων, γραμμένο σε Python και ικανό να τρέχει πάνω από το TensorFlow, το CNTK ή το Theano. Αναπτύχθηκε με έμφαση στη γρήγορη πειραματισμό. Η ικανότητά σας να μεταβείτε από την ιδέα σε αποτέλεσμα με την ελάχιστη δυνατή καθυστέρηση είναι το κλειδί για να κάνετε καλή έρευνα.

To Keras είναι μια βιβλιοθήκη υλοποιημένη για deep learning. Επιτρέπει την εύκολη και γρήγορη δημιουργία πρωτοτύπων. Υποστηρίζει τόσο συνελικτικά δίκτυα όσο και επαναλαμβανόμενα δίκτυα, καθώς και συνδυασμούς των δύο. Λειτουργεί απρόσκοπτα τόσο στην CPU, όσο και στην GPU.

Το TensorFlow ™ είναι μια βιβλιοθήκη λογισμικού ανοιχτού κώδικα για αριθμητικούς υπολογισμούς χρησιμοποιώντας γραφήματα ροής δεδομένων. Οι κόμβοι στο γράφημα αντιπροσωπεύουν μαθηματικές λειτουργίες, ενώ οι άκρες των γραφημάτων αντιπροσωπεύουν τις πολυδιάστατες συστοιχίες δεδομένων (tensors) που επικοινωνούν μεταξύ τους. Η ευέλικτη αρχιτεκτονική επιτρέπει την ανάπτυξη σε μία ή περισσότερες CPU ή GPU σε οποιαδήποτε συσκευή με ένα μόνο API. Το TensorFlow αναπτύχθηκε αρχικά από ερευνητές και μηχανικούς που εργάζονται στον ερευνητικό οργανισμό Machine Intelligence της Google για σκοπούς διερεύνησης της μηχανικής μάθησης και της έρευνας σε νευρωνικά δίκτυα.

**Β. Χρήση Εκπαιδευμένου Συνελικτικού Νευρωνικού Δίκτυου**

Η βιβλιοθήκη Keras διαθέτει εφαρμογές (applications) με μοντέλα βαθιάς μάθησης που διατίθενται μαζί με προ-εκπαιδευμένα βάρη. Αυτά τα μοντέλα μπορούν να χρησιμοποιηθούν για την πρόβλεψη, την εξαγωγή χαρακτηριστικών και την τελειοποίηση. Τα βάρη των προ- εκπαιδευμένων μοντέλων του keras λαμβάνονται αυτόματα κατά την εκμάθηση ενός μοντέλου και αποθηκεύονται στην διεύθυνση ~ /.keras/models/.

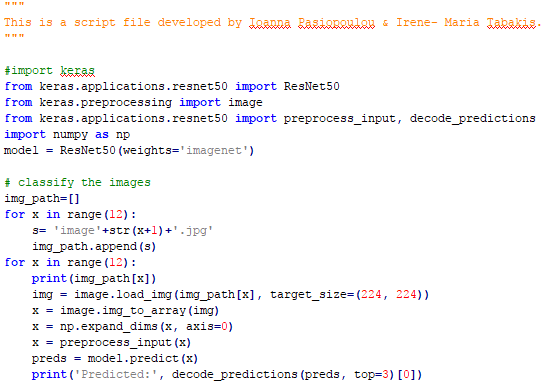
Τα διαθέσιμα μοντέλα του Keras API για ταξινόμηση εικόνων με βάρη εκπαιδευμένα στο ImageNet είναι τα: Xception, VGG16, VGG19, ResNet50, InceptionV3, InceptionResNetV2, MobileNet, DenseNet και NASNet.

Όλες αυτές οι αρχιτεκτονικές (εκτός από το Xception και το MobileNet) είναι συμβατές τόσο με το TensorFlow όσο και με το Theano, ενώ τα μοντέλα Xception και MobileNet είναι διαθέσιμα μόνο στο TensorFlow. Κατά την εκμάθηση η μορφή των δεδομένων είναι τύπου JSON.

Επιλέξαμε να χρησιμοποιήσουμε το ResNet50. Το μοντέλο αυτό δέχεται από προεπιλογή εισόδους εικόνας μεγέθους 224x224. Μπορεί να υλοποιηθεί με μορφή ‘channels\_first’, με δεδομένα σε μορφή (κανάλι, ύψος, πλάτος) ή ‘channels\_last’ με μορφή (ύψος, πλάτος, κανάλι).

Οι εικόνες που επιλέξαμε ήταν τυχαίες.



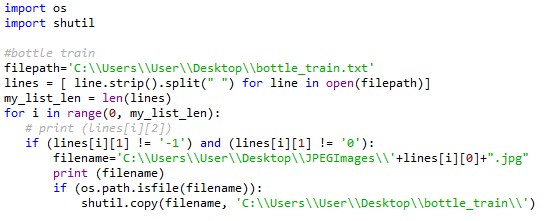


Τα αποτελέσματα που λάβαμε τρέχοντας τον παραπάνω κώδικα για κάθε εικόνα αντίστοιχα είναι τα εξής:



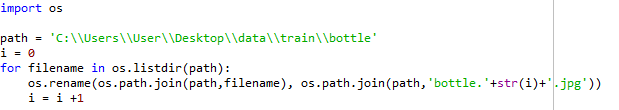
**Γ. Κατασκευή & Εκπαίδευση Συνελικτικού Νευρωνικού Δικτύου Ταξινόμησης Εικόνων**

Το σετ δεδομένων που χρησιμοποιήσαμε είναι το PASCAL VOC 2012. Οι κλάσεις που μας ενδιέφεραν είναι οι bottle και bus. Κατασκευάσαμε λοιπόν ένα script για να ξεχωρίσουμε τις εικόνες που μας ενδιέφεραν. Ενδεικτικά παρατίθεται κώδικας:



Εικόνα : Κώδικας διαχωρισμού εικόνων για εκπαίδευση νευρωνικού κατηγορίας "bottle".

Έπειτα, μετονομάσαμε τις εικόνες ώστε το όνομά τους να έχει την μορφή “category.number.jpg”:



Εικόνα : Κώδικας μετονομασίας εικόνων κατηγορίας "bottle".

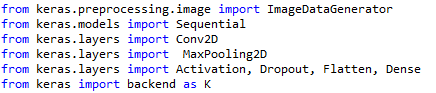
Συνολικά είχαμε για εκπαίδευσης:

* 365 εικόνες της κατηγορίας bottle.
* 213 εικόνες της κατηγορίας bus.

Ενώ για επαλήθευση:

* 341 εικόνες της κατηγορίας bottle.
* 208 εικόνες της κατηγορίας bus.

Για την εκμάθηση του ΝΝ ακολουθήσαμε αυτή τη διαδικασία του feature extraction. Ξεκινήσαμε με το να συμπεριλάβουμε τις κατάλληλες βιβλιοθήκες:



Έπειτα τα βήματα που ακολουθήσαμε είναι τα εξής:

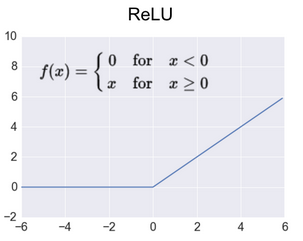
* **Βήμα 1. Συνέλιξη- Convolution**

Με το Sequential του Keras αρχικοποιούμε το ΝΝ με μια σειρά/αλληλουχία στρωμάτων (sequence of layers), δημιουργώντας ένα στιγμιότυπο (instance) της κλάσης (sequential).





Με την αντίστοιχη εντολή εφαρμόζουμε συνέλιξη στο αντικείμενο που δημιουργήσαμε πριν. Εφαρμόζουμε 32 τρισδιάστατα φίλτρα, και οι εικόνες υα έχουν 64x64 ανάλυση και θα είναι έγχρωμες. Η συνάρτηση ενεργοποίηση είναι η relu, συνεχής αλλά όχι παντού παραγωγίσιμη.



*Σημείωση: Μία γραμμή pixel περιμετρικά της εικόνας αφαιρείτε εξ’ αρχής.*

* **Βήμα 2. Pooling**

Για να μικρύνουμε το μέγεθος του της εικόνας, και άρα των κόμβων στα επόμενα στρώματα (layers) εφαρμόζουμε ένα είδος υπο-δειγματοληψίας που ονομάζεται pooling. Μία από τις τεχνικές είναι το Max Pooling που εφαρμόσαμε και εδώ με μέγεθος φίλτρου 2x2 για να μη χάσουμε πολλή πληροφορία, με βήμα 2.



Για μεγαλύτερη μείωση της πολυπλοκότητας επαναλάβαμε τα προηγούμενα βήματα.

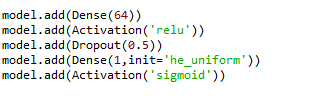
* **Βήμα 3. Flattening**

Εδώ μετατρέπουμε την εικόνα σε διάνυσμα για να εισαχθεί στο νευρωνικό.



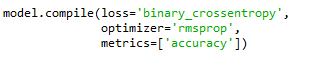
* **Βήμα 4. Εκπαίδευση νευρωνικού**

Προστίθεται το κρυφό στρώμα (hidden layer) με 64 κόμβους και συνάρτηση ενεργοποίησης τη relu.



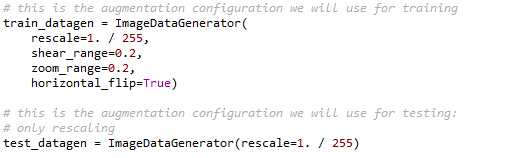
Αρχικοποίηση στρώματος εξόδου. Έχει μία έξοδο καθώς είναι δυαδική και ως συνάρτηση ενεργοποίησης χρησιμοποιεί τη σιγμοειδή.

Τρέξιμο νευρωνικού:

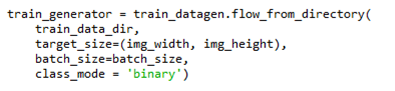


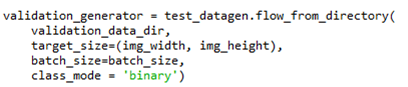
Η παράμετρος optimizer καθορίζει τον αλγόριθμο με τον οποίο θα κάνει βελτιστοποίηση, η loss τη συνάρτηση απώλειας και η μετρική αξιολόγησης την επίδοση του μοντέλου εδώ: accuracy.

Στη συνέχεια επεξεργαζόμαστε τις εικόνες για να τις εισάγουμε στο μοντέλο. Αυτό το κάνουμε για να αποφύγουμε το overfitting.

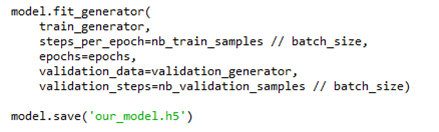


Τέλος, επιλέγουμε το training και το testing set, δηλαδή τις εικόνες που θα φορτώνονται από τον φάκελο μία προς μία και παράγουμε το μοντέλο μας.



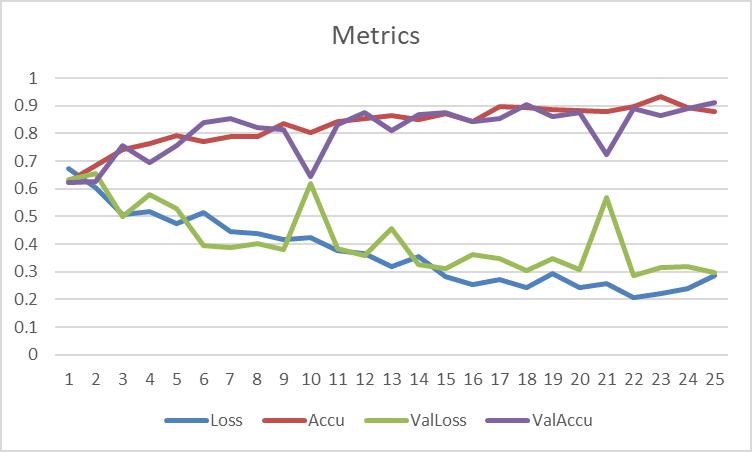


Το μοντέλο με τα βάρη του αποθηκεύεται στο αρχείο ‘our\_model.h5’.



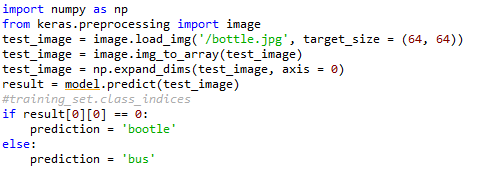
Επιλέξαμε να τρέξουμε το μοντέλο μας με 25 εποχές.

Φτάσαμε να παίρνουμε την τελευταία εποχή έφτασε validation accuracy 90.99% με validation loss 0.2987.



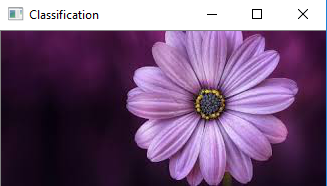
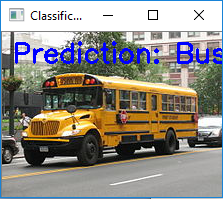
**Δ. Επαλήθευση Ομαλής Λειτουργίας Νευρωνικού Δικτύου**

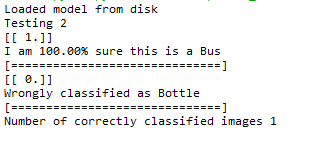
Σε πρώτη φάση θέλαμε δοκιμάσαμε να ταξινομήσουμε εικόνες που να απεικονίζουν μπουκάλι,-α ή λεωφορείο-α, οι οποίες δεν εμπεριέχονταν στις εικόνες εκπαίδευσης και επαλήθευσης.



Στο 100% των εικόνων που δοκιμάσαμε αναγνωρίστηκε το αντίστοιχο προϊόν, ωστόσο αν βάζαμε κάποια εικόνα που να μην απεικονίζει κάποιο από τα δύο το νευρωνικό μας το ταξινομούσε τυχαία. Έτσι προσπαθήσαμε να αναγνωρίσουμε τα αντικείμενα που ταξινομούνται λανθασμένα.







**Ε. Άλλες Δοκιμές**

Κάναμε άλλες 2 δοκιμές. Συγκεντρωτικά αποτελέσματα

|  |  |  |  |
| --- | --- | --- | --- |
| Δοκιμή | Χρόνος Εκτέλεσης | Validation Accuracy | Validation Loss |
| 1η | 12 ώρες | 87,27% | 1,1197 |
| 2η | 2 ώρες | 86,73% | 0,42 |
| 3η  (Παραπάνω υλοποίηση) | 15 λεπτά | 90,99% | 0,2987 |