ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΥΠΟΛΟΓΙΣΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ταυτοποίηση γουνοφόρων ζώων με χρήση συνελικτικού νευρωνικού δικτύου βαθιάς μάθησης

Identification of fur animals using a deep convolutional neural network

Έλιος Ντίμο

Α.Μ: 161071

Επιβλέπων Καθηγητής:

Νικόλαος Βασιλάς

Αθήνα, Ιούλιος 2023

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ταυτοποίηση γουνοφόρων ζώων με χρήση συνελικτικού νευρωνικού δικτύου βαθιάς μάθησης

Έλιος Ντίμο

Α.Μ. 711 161071

Επιβλέπων Καθηγητής: **Νικόλαος Βασιλάς**

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την Πέμπτη 13/07/2023

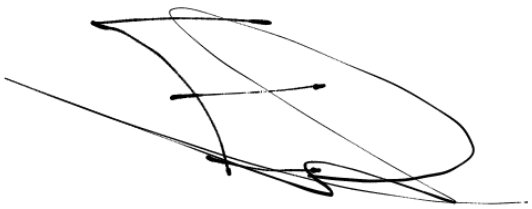
(Υπογραφή)	**(Υπογραφή)**	**(Υπογραφή)**
Nikolaos Vasilas Digitally signed by Nikolaos Vasilas Date: 2023.07.13 16:01:09 +03'00'	Φοίβος-Απόστολος Μυλωνάς - Ψηφιακά υπογεγραμμένο 2023.07.13 17:41:29 +03'00'	Georgios Bardis Digitally signed by Georgios Bardis Date: 2023.07.14 19:03:34 +03'00'
.....	.....	.....
Νικόλαος Βασιλάς	**Φοίβος-Απόστολος Μυλωνάς**	**Γεώργιος Μπαρδής**
Καθηγητής	**Αναπλ. Καθηγητής**	**Επίκ. Καθηγητής**

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Ο κάτωθι υπογεγραμμένος Έλιος Ντίμο του Ανέστη, με αριθμό μητρώου 161071, φοιτητής του Πανεπιστημίου Δυτικής Αττικής της Σχολής Μηχανικών του Τμήματος Μηχανικών Πληροφορικής και Υπολογιστών, δηλώνω υπεύθυνα ότι:

«Είμαι συγγραφέας αυτής της πτυχιακής/διπλωματικής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, οι όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε ακριβώς είτε παραφρασμένες, αναφέρονται στο σύνολό τους, με πλήρη αναφορά στους συγγραφείς, τον εκδοτικό οίκο ή το περιοδικό, συμπεριλαμβανομένων και των πηγών που ενδεχομένως χρησιμοποιήθηκαν από το διαδίκτυο. Επίσης, βεβαιώνω ότι αυτή η εργασία έχει συγγραφεί από μένα αποκλειστικά και αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο δικής μου, όσο και του Ιδρύματος. Παράβαση της ανωτέρω ακαδημαϊκής μου ευθύνης αποτελεί ουσιώδη λόγο για την ανάκληση του πτυχίου μου».

Ο Δηλών



ΠΕΡΙΛΗΨΗ

Η βαθιά μάθηση αποτελεί υποκατηγορία της μηχανικής μάθησης και μέρος της είναι η εκπαίδευση και χρήση συνελικτικών νευρωνικών δικτύων (CNN) για την αναγνώριση και ταυτοποίηση εικόνων. Αναγνωρίζουν μοτίβα και χαρακτηριστικά σε εικόνες με την εφαρμογή πολλαπλών στρωμάτων φίλτρων μεταβιβάζοντας κάθε φορά της πληροφορίες στο επόμενο επίπεδο.

Ο στόχος της παρούσας διπλωματικής εργασίας είναι η δημιουργία ενός μοντέλου που θα αναγνωρίζει μοτίβα γουνοφόρων ζώων σύμφωνα με τα δεδομένα που του δώσαμε για την εκπαίδευση. Συγκεκριμένα έχουμε 30 κλάσεις γουνοφόρων και το μοντέλο θα πρέπει να αναγνωρίζει ζώα που ανήκουν μέσα σε αυτές τις 30 κλάσεις.

Ένα δίκτυο βαθιάς μάθησης έχει την ικανότητα να λαμβάνει αποφάσεις με βάση τα δεδομένα στα οπαία εκπαιδεύεται. Γι' αυτό λοιπόν η συλλογή και ομοιόμορφη κατανομή των δεδομένων αποτελεί ένα απο τα πιο σημαντικά κομμάτια της όλης διαδικασίας. Κάθε κλάση χωρίζεται σε 3 πεδία train, validation και test set με 800, 100 και 100 εικόνες κατά αντιστοιχία.

Για να καταλήξουμε στο τελικό μοντέλο έγιναν πολλοί πειραματισμοί. Δημιουργήθηκαν διάφορα CNN μοντέλα ποικίλων επιπέδων τα οποία χτίστηκαν απο το μηδέν χωρίς όμως να δίνουν τα επιθυμητά αποτελέσματα. Χάρις σε αυτό όμως έγινε καλύτερη κατανόηση του προβλήματος οδηγώντας στην εφαρμογή μια τεχνικής που ονομάζεται μάθηση μεταφοράς που αξιοποιεί τις αρχιτεκτονικές και χαρακτηριστικά που έχουν εξαχθεί από άλλα μοντέλα. Στην προκειμένου περίπτωση έγινε χρήση του VGG16 που μας έδωσε ικανοποιητικά αποτελέσματα.

Λέξεις κλειδιά

Μηχανική Μάθηση, Βαθιά Μάθηση, Συνελικτικά Νευρωνικά Δίκτυα, Ταυτοποίηση, Μάθηση Μεταφοράς

ABSTRACT

Deep learning is a subset of machine learning and part of it is the training and use of convolutional neural networks (CNNs) for image recognition and identification. They recognize patterns and features in images by applying multiple layers of filters, each time passing the information to the next layer.

The objective of this thesis is to create a model that recognizes furbearing animal patterns according to the data we provided for training. Specifically, we have 30 classes of furbearers and the model should recognize animals that belong within these 30 classes.

A deep learning network has the ability to make decisions based on the data it is trained on. Therefore, the collection and uniform distribution of data is one of the most important parts of the whole process. Each class is divided into 3 fields of train, validation and test sets with 800, 100 and 100 images respectively.

To arrive at the final model many experiments were done. Several CNN models of varying levels were created which were built from scratch without giving the desired results. Thanks to this, however, a better understanding of the problem was gained, leading to the application of a technique called transfer learning that exploits architectures and features extracted from other models. In this case we used VGG16 which gave us satisfactory results.

Key Words

Machine Learning, Deep Learning, Convolutional Neural Network, Identification, Transfer learning

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΑΤΑΛΟΓΟΣ ΣΧΗΜΑΤΩΝ

Κεφάλαιο 1 – Εισαγωγή

Η μηχανική μάθηση και συγκεκριμένα η χρήση των CNN γίνεται όλο και πιο συχνή και συνηθισμένη σε διάφορους τομείς στην καθημερινότητα των ανθρώπων. Για να φτάσει σε σημείο να χρησιμοποιείται με ικανοποιητικά αποτελέσματα χρειάστηκαν χρόνια μελέτης και πειραμάτων.

1.1 Ορισμός προβλήματος

Για την δημιουργία μοντέλου αναγνώρισης εικόνων και στην προκειμένου περίπτωση γουνοφόρων ζώων αντιμετωπίζονται πολλά προβλήματα κατά την διάρκεια ανάπτυξης. Ειδικά όταν έχουμε μεγάλο αριθμό κλάσεων η όλη διαδικασία γίνεται ακόμη πιο δύσκολη καθώς έχουμε να κάνουμε με ποικίλα χαρακτηριστικά. Ένα μοντέλο είναι τόσο καλό όσο το ποσοστό ακρίβειας του. Για να φτάσει να έχει υψηλό ποσοστό χρειάζεται σημαντικός χρόνος για να γίνουν τα πειράματα και να ξεπεραστούν τα εμπόδια.

1.2 Προτεινόμενη λύση

Για την αντιμετώπιση των προβλημάτων που προαναφέρθηκαν έχουν δημιουργηθεί μοντέλα που εκπαιδεύονται για διαφορετικά σύνολα δεδομένων και που επαναχρησιμοποιούνται. Έτσι αποφεύγονται τα συνήθη προβλήματα και σε συνδυασμό με την Μάθηση Μεταφοράς καταλήγουμε συντομότερα στο επιθυμητό αποτέλεσμα. Αποφεύγεται η ανάπτυξη μοντέλου από την αρχή και μένει περισσότερος χρόνος για την εκπαίδευση και τα τεστ που κατά κύριο λόγο αποτελούνται από την ρύθμιση παραμέτρων μια μέθοδο για την καλυτέρευση των νευρωνικών δικτύων.

1.3 Συμβολή

Το αντικείμενο της εργασίας είναι η αναγνώριση γουνοφόρων ζώων. Για να καταλήξουμε σε ένα αξιόλογο μοντέλο έγιναν τα εξής βήματα:

- Συλλογή συνόλου δεδομένων από το διαδίκτυο οργάνωση και κατηγοριοποίηση σε train, validation και test sets.
- Παραγωγή πηγαίου κώδικα απο το μηδέν διαφορετικών επιπέδων κάθε φορά.
- Ρύθμιση παραμέτρων συγκρίνοντας κάθε φορά τα αποτελέσματα.
- Μετά την εξάντληση δοκιμών και τέστ κατάληξη στο συμπέρασμα πως η δημιουργία CNN για το συγκεκριμένο σύνολο δεδομένων απο το μηδέν δεν επαρκεί.
- Χρήση μοντέλου VGG16 και αξιοποίηση του με μέθοδο Μάθησης Μεταφοράς.
- Ρύθμιση παραμέτρων και σύγκριση αποτελεσμάτων.
- Τελική αξιολόγηση και επιλογή καταλληλότερου μοντέλου.

1.4 Δομή

Τα κεφάλαια στα οποία είναι χωρισμένη η εργασία είναι:

Κεφάλαιο 1: Εισαγωγή.

Μικρή αναφορά στην αρχή των CNN και βήματα εκπόνησης πειραματικού μέρους.

Κεφάλαιο 2: Μηχανική μάθηση.

Ανάλυση θεωρητικού μέρους της μηχανικής μάθησης.

Κεφάλαιο 3: Βαθιά μάθηση και συνελικτικά νευρωνικά δίκτυα .

Ανάλυση θεωρητικού μέρους της βαθιάς μάθησης και των συνελικτικών νευρωνικών δικτύων.

Κεφάλαιο 4: Ανάλυση παραμέτρων CNN μοντέλου και συνόλου δεδομένων.

Επεξήγηση παραμέτρων ενός τυπικού CNN μοντέλου και οργάνωση συνόλου δεδομένων.

Κεφάλαιο 5: Πειραματικό μέρος 1ο

Ανάλυση αποτελεσμάτων διαφορετικών CNN μοντέλων που φτιάχτηκαν απο το μηδέν.

Κεφάλαιο 6: Μάθηση μεταφοράς.

Ανάλυση θεωρητικού μέρους της μάθησης μεταφοράς.

Κεφάλαιο 7: Πειραματικό μέρος 2ο

Ανάλυση αποτελεσμάτων μοντέλου VGG16 με εφαρμογή μάθησης μεταφοράς.

Κεφάλαιο 8: Συμπεράσματα.

Παρουσιάζονται τα συμπεράσματα που προέκυψαν από όλα τα προηγούμενα κεφάλαια.

Κεφάλαιο 2 – Μηχανική μάθηση.

Η μηχανική μάθηση ανήκει στο πεδίο της τεχνητής νοημοσύνης (AI) και της επιστήμης των υπολογιστών που αντιγράφει και μιμείται τρόπους με τους οποίους μαθαίνει ο άνθρωπος. Η ιδιότητα που την χαρακτηρίζει είναι πως με το πέρασμα του χρόνου βελτιώνεται η ακρίβεια της κάνοντας χρήση αλγορίθμων και συλλέγοντας δεδομένα.

2.1 Ιστορική αναδρομή.

Το 1943 ο Warren McCulloch ένας νευροφυσιολόγος μαζί με τον συνάδελφο του Walter Pitts ο οποίος ήταν μαθηματικός δούλεψαν πάνω σε μια έρευνα χάρις στην οποία η επιστημονική κοινότητα ήρθε για πρώτη φορά σε επαφή με τους νευρώνες και τη λειτουργία τους. Αποτέλεσμα ήταν η γέννηση του πρώτου τεχνητού νευρώνα μέσα απο ένα μοντέλο ηλεκτρικών κυκλωμάτων.

Ο Άρθουρ Σάμιουελ, ερευνητής της IBM και πρωτοπόρος στον τομέα της τεχνητής νοημοσύνης και των ηλεκτρονικών παιχνιδιών επινόησε το 1943 τον όρο μηχανική νοημοσύνη. Το 1952 ο ίδιος ανέπτυξε πρόγραμμα που μάθαινε στον υπολογιστή να παίζει ντάμα ένα απο τα πιο γνωστά παιχνίδια. Το συνώνυμο αυτοδίδακτοι υπολογιστές χρησιμοποιήθηκε επίσης κατά την περίοδο αυτή.

Perceptron ήταν η ονομασία του πρώτου νευρωνικού δικτύου το 1957 με δημιουργό τον Frank Rosenblatt. Αποτελεί την απλούστερη μορφή ενός νευρωνικού δικτύου που χρησιμοποιείται για την ταξινόμηση προτύπων που λέγεται ότι είναι γραμμικά διαχωρίσιμα. Επιπρόσθετα έχει έναν μόνο νευρώνα με ρυθμιζόμενα συναπτικά βάρη και πολώσεις. Για τη ρύθμιση των ελεύθερων παραμέτρων αυτού του νευρωνικού δικτύου χρησιμοποιείται ένας αλγόριθμος ο οποίος εμφανίστηκε για πρώτη φορά σε μια διαδικασία μάθησης που αναπτύχθηκε από τον Rosenblatt για το μοντέλο εγκεφαλικής φύσεως του perceptron.

Τη δεκαετία του 1990 ήταν μια μεταβατική περίοδος για την μηχανική μάθηση καθώς αντί να χρησιμοποιείται γνώση αξιοποιήθηκαν οι τεράστιες πηγές δεδομένων. Το 1997 ο παγκόσμιος πρωταθλητής στο σκάκι γνώρισε την ήττα απο μια μηχανή της IBM εν ονόματι Deep Blue. Πολλές επιχειρήσεις δεν άργησαν να δουν τις δυνατότητες που είχε η μηχανική μάθηση σε πολύπλοκα ζητήματα και υπολογισμούς. Κάποια απο τα πιο πρόσφατα δημιουργήματα είναι: Το Google Brain (2012), ένα βαθύ νευρωνικό δίκτυο που αναγνωρίζει πρόσωπα και αντικείμενα μέσα απο βίντεο και εικόνες. Άρχισε να χρησιμοποιείται σε μεγάλο βαθμό στην πλατφόρμα του YouTube. Το Deep Face (2014), πρόγραμμα του Facebook που καταφέρνει με μεγάλη ακρίβεια να αναγνωρίζει ανθρώπους με τρόπο παρόμοιο αυτό των ανθρώπων.

2.2 Στόχοι

Οι στόχοι της μηχανικής μάθησης στην σύγχρονη εποχή είναι δύο, ο ένας είναι να αξιοποιεί τα μοντέλα που έχουν αναπτυχθεί για να ταξινομεί δεδομένα, ο άλλος είναι χρησιμοποιώντας αυτά τα μοντέλα να κάνει μελλοντικές προβλέψεις δίνοντας επιθυμητά αποτελέσματα.

Η κατανόηση της ανθρώπινης μάθησης και της υπολογιστικής της πτυχής είναι ένας αξιόλογος επιστημονικός στόχος. Τα ανθρώπινα όντα έχουμε από καιρό γοητευτεί από τις ικανότητές μας για ευφυείς συμπεριφορές και προσπαθούμε να κατανοήσουμε τη φύση της νοημοσύνης. Είναι σαφές ότι κεντρικό στοιχείο της νοημοσύνης μας είναι η ικανότητά μας να μαθαίνουμε. Συνεπώς, η σε βάθος κατανόηση της ανθρώπινης διαδικασίας μάθησης είναι ζωτικής σημασίας για την κατανόηση της ανθρώπινης νοημοσύνης. Η μηχανική μάθηση θα μας δώσει την εικόνα των υποκείμενων αρχών της ανθρώπινης μάθησης και αυτό μπορεί να οδηγήσει στην ανακάλυψη αποτελεσματικότερων εκπαιδευτικών τεχνικών. Θα συμβάλει επίσης στο σχεδιασμό συστημάτων μηχανικής μάθησης. Τέλος, είναι επιθυμητό να εξερευνήσουμε εναλλακτικούς μηχανισμούς μάθησης στο χώρο όλων των πιθανών μαθησιακών μεθόδων μάθησης. Δεν θα πρέπει να θεωρείται ότι ο τρόπος με τον οποίο μαθαίνει ο άνθρωπος είναι ο μόνος δυνατός μηχανισμός μάθησης. Αξίζει να διερευνηθούν άλλες μέθοδοι μάθησης οι οποίες μπορεί να είναι πιο αποδοτικές, αποτελεσματικές από ό,τι ανθρώπινη μάθηση.

2.3 Μέθοδοι

Επιβλεπόμενη μηχανική μάθηση.

Η επιβλεπόμενη μάθηση, κάνει κατάλληλη χρήση συνόλων δεδομένων εκπαιδεύοντας πανέξυπνους αλγορίθμους ικανούς να προβλέπουν με ακρίβεια αποτελέσματα και να ταξινομούν δεδομένα. Όταν το μοντέλο τροφοδοτείται απο τα δεδομένα εισόδου γίνεται και προσαρμογή στα βάρη οδηγώντας στην συνολική προσαρμογή του μοντέλου. Αυτό συμβαίνει στο πλαίσιο της διαδικασίας διασταυρούμενης επικύρωσης για να διασφαλιστεί ότι το μοντέλο αποφεύγει την υπερπροσαρμογή ή την υποπροσαρμογή. Οι επιχειρήσεις εφαρμόζουν την επιβλεπόμενη μάθηση σε καθημερινή βάση καθώς ανταποκρίνεται ικανοποιητικά σε ποικίλα προβλήματα του πραγματικού κόσμου με χαρακτηριστικό παράδειγμα ο διαχωρισμός των ανεπιθύμητων και εισερχόμενων μηνυμάτων σε ξεχωριστούς φακέλους. Τα νευρωνικά δίκτυα, το naïve bayes, η γραμμική παλινδρόμηση, η λογιστική παλινδρόμηση, το τυχαίο δάσος και η μηχανή διανυσμάτων υποστήριξης (SVM) είναι κάποιες απο τις μεθόδους της επιβλεπόμενης μηχανικής μάθησης.

Μη εποπτευόμενη μηχανική μάθηση.

Η μη επιβλεπόμενη μάθηση, χρησιμοποιεί αλγορίθμους μηχανικής μάθησης που δεν χρειάζονται ανθρώπινη παρέμβαση. Αυτοί οι αλγόριθμοι ομαδοποιούν δεδομένα ή ανακαλύπτουν κρυμμένα μοτίβα. Λόγω της ικανότητας της συγκεκριμένης μεθόδου να ξεχωρίζει τις διαφορές και ομοιότητες στις πληροφορίες την καθιστούν κατάλληλη για κατηγοριοποίηση πελατών, ανάλυση δεδομένων, έξυπνες στρατηγικές πωλήσεων, αναγνώριση μοτίβων και εικόνων. Επιπλέον είναι αξιόπιστο για τη μείωση του αριθμού των χαρακτηριστικών σε ένα μοντέλο μέσω της διαδικασίας μείωσης της διαστατικότητας. Η

ανάλυση κύριων συνιστωσών (PCA) και η αποσύνθεση μοναδικών τιμών (SVD) είναι δύο κοινές προσεγγίσεις για το σκοπό αυτό. Στην μηχανική μάθηση δίχως επίβλεψη γίνεται χρήση κι άλλων αλγορίθμων που περιλαμβάνουν νευρωνικά δίκτυα, ομαδοποίηση k-means και πιθανοτικές μεθόδους ομαδοποίησης.

Μάθηση με ημιεπίβλεψη.

Η μάθηση με ημιεπίβλεψη βρίσκεται στο κέντρο της επιβλεπόμενης και της μη επιβλεπόμενης μάθησης. Αρχίζει με μικρό σύνολο δεδομένων με ετικέτες που βοηθάνε στην εξαγωγή και αποθήκευση χαρακτηριστικών απο ένα μεγαλύτερο σύνολο δεδομένων χωρίς ετικέτες. Όλα αυτά γίνονται κατά την διάρκεια της εκπαίδευσης ενώ έχει επίσης την δυνατότητα να βοηθήσει τους αλγόριθμους μάθησης με επίβλεψη σε περίπτωση που έχουν έλλειψη επισημασμένων δεδομένων.

2.4 Αλγόριθμοι

Συνήθως χρησιμοποιούνται διάφοροι αλγόριθμοι μηχανικής μάθησης. Αυτοί περιλαμβάνουν:

Νευρωνικά δίκτυα: Τα νευρωνικά δίκτυα προσομοιώνουν τον τρόπο λειτουργίας του ανθρώπινου εγκεφάλου, με έναν τεράστιο αριθμό συνδεδεμένων κόμβων επεξεργασίας. Τα νευρωνικά δίκτυα είναι καλά στην αναγνώριση μοτίβων και διαδραματίζουν σημαντικό ρόλο σε εφαρμογές όπως η μετάφραση φυσικής γλώσσας, η αναγνώριση εικόνων, η αναγνώριση ομιλίας και η δημιουργία εικόνων.

Γραμμική παλινδρόμηση: Αυτός ο αλγόριθμος χρησιμοποιείται για την πρόβλεψη αριθμητικών τιμών, με βάση μια γραμμική σχέση μεταξύ διαφορετικών τιμών. Ένα απλό παράδειγμα της συγκεκριμένης τεχνικής είναι η πρόβλεψη των τιμών των κατοικιών με βάση ιστορικά δεδομένα για την περιοχή.

Λογιστική παλινδρόμηση: Αυτός ο αλγόριθμος μάθησης με επίβλεψη κάνει προβλέψεις για κατηγορικές μεταβλητές απόκρισης, όπως οι απαντήσεις "ναι/όχι" σε ερωτήσεις. Μπορεί να χρησιμοποιηθεί για εφαρμογές όπως η ταξινόμηση ανεπιθύμητης αλληλογραφίας και ο ποιοτικός έλεγχος σε μια γραμμή παραγωγής.

Συσταδοποίηση: Χρησιμοποιώντας μη επιβλεπόμενη μάθηση, οι αλγόριθμοι ομαδοποίησης μπορούν να εντοπίσουν μοτίβα στα δεδομένα ώστε να μπορούν να ομαδοποιηθούν. Οι υπολογιστές μπορούν να βοηθήσουν τους επιστήμονες δεδομένων εντοπίζοντας διαφορές μεταξύ στοιχείων δεδομένων που οι άνθρωποι έχουν παραβλέψει.

Δέντρα αποφάσεων: Τα δέντρα αποφάσεων μπορούν να χρησιμοποιηθούν τόσο για την πρόβλεψη αριθμητικών τιμών (παλινδρόμηση) όσο και για την ταξινόμηση δεδομένων σε κατηγορίες. Τα δέντρα αποφάσεων χρησιμοποιούν μια διακλαδιζόμενη ακολουθία συνδεδεμένων αποφάσεων που μπορούν να αναπαρασταθούν με ένα δενδροειδές διάγραμμα. Ένα από τα πλεονεκτήματα των δέντρων αποφάσεων είναι ότι είναι εύκολο να επικυρωθούν και να ελεγχθούν, σε αντίθεση με το μαύρο κουτί του νευρωνικού δικτύου.

Τυχαία δάση: Σε ένα τυχαίο δάσος, ο αλγόριθμος μηχανικής μάθησης προβλέπει μια τιμή ή μια κατηγορία συνδυάζοντας τα αποτελέσματα από έναν αριθμό δέντρων αποφάσεων.

2.5 Πρακτική χρήση

Μερικά απο τα παραδείγματα μηχανικής μάθησης στην καθημερινότητα:

Αναγνώριση ομιλίας: Η μετάφραση της ανθρώπινης ομιλίας σε γραπτή μορφή αξιοποιώντας την τεχνητή νοημοσύνη NLP. Είναι γνωστή και ως ASR δηλαδή αυτόματη αναγνώριση ομιλίας. Τα σύγχρονα κινητά τηλέφωνα αξιοποιούν αυτές τις δυνατότητες για τη διεξαγωγή φωνητικής αναζήτησης -π.χ. Siri- ή για τη βελτίωση της προσβασιμότητας για την αποστολή κειμένου.

Εξυπηρέτηση πελατών: Τα διαδικτυακά chatbots αντικαθιστούν τους ανθρώπινους πράκτορες κατά μήκος της διαδρομής του πελάτη, αλλάζοντας τον τρόπο προσέγγισης τους μέσα απο το διαδίκτυο. Τα chatbots απαντούν στις πιο συχνές ερωτήσεις (FAQ) σε ότι αφορά προϊόντα ή υπηρεσίες και δίνουν εξατομικευμένες συμβουλές προτείνοντας επιλογές που είναι πιο κοντά στις ανάγκες του πελάτη. Τα παραδείγματα περιλαμβάνουν εικονικούς πράκτορες σε ιστότοπους ηλεκτρονικού εμπορίου, χρησιμοποιώντας bots μηνυμάτων όπως το Slack, το Facebook Messenger και φωνητικούς ή εικονικούς βοηθούς για την εκτέλεση εργασιών.

Όραση υπολογιστών: Χάρις σε αυτή την τεχνολογία τεχνητής νοημοσύνης οι υπολογιστές μπορούν και αντλούν αξιοσημείωτες πληροφορίες από ψηφιακές εικόνες, βίντεο και άλλες οπτικές εισροές και στη συνέχεια να αναλαμβάνουν την κατάλληλη δράση. Με τη βοήθεια των νευρωνικών δικτύων συνελικτικού τύπου, η υπολογιστική όραση εφαρμόζεται στην υγειονομική περίθαλψη για ακτινολογικές απεικονίσεις, στα μέσα κοινωνικής δικτύωσης για την αναγνώριση συνήθως προσώπων σε φωτογραφίες και στην αυτοκινητοβιομηχανία για αυτοκινούμενα αμάξια όπως για παράδειγμα τα Tesla.

Μηχανές συστάσεων: Μελετώντας δεδομένα παρελθούσας καταναλωτικής συμπεριφοράς, οι αλγόριθμοι ΤΝ μπορούν να βοηθήσουν στην ανάδειξη νέων τάσεων δεδομένων ώστε να αναπτυχθούν πιο αποτελεσματικές τεχνικές διασταυρούμενων πωλήσεων. Η προσέγγιση αυτή χρησιμοποιείται από τους διαδικτυακούς λιανοπωλητές για να κάνουν σχετικές συστάσεις προϊόντων στους πελάτες κατά τη διάρκεια της διαδικασίας πληρωμής.

Αυτοματοποιημένη διαπραγμάτευση μετοχών: Οι πλατφόρμες συναλλαγών υψηλής συχνότητας με βάση την ΤΝ πραγματοποιούν πολυάριθμες συναλλαγές την ημέρα χωρίς ανθρώπινη παρέμβαση ειδικά ανεπτυγμένες για τη βελτιστοποίηση χρηματιστηριακών χαρτοφυλακίων.

Ανίχνευση απάτης: Οι τράπεζες και άλλα χρηματοπιστωτικά ιδρύματα μπορούν να χρησιμοποιούν τη μηχανική μάθηση για να εντοπίζουν ύποπτες συναλλαγές. Η μάθηση με επίβλεψη μπορεί να εκπαιδεύσει ένα μοντέλο χρησιμοποιώντας πληροφορίες σχετικά με γνωστές δόλιες συναλλαγές. Η ανίχνευση ανωμαλιών μπορεί να εντοπίσει συναλλαγές που φαίνονται άτυπες και χρήζουν περαιτέρω διερεύνησης.

Κεφάλαιο 3 – Βαθιά μάθηση και συνελικτικά νευρωνικά δίκτυα.

Μέχρι στιγμής, έχουμε μιλήσει για τη μηχανική μάθηση σε γενικές γραμμές. Η βαθιά μάθηση είναι το υποσύνολο της μηχανικής μάθησης που ασχολείται με μοντέλα βασισμένα σε νευρωνικά δίκτυα πολλών επιπέδων. Είναι βαθιά ακριβώς με την έννοια ότι τα μοντέλα της μαθαίνουν πολλά επίπεδα μετασχηματισμών. Αν αυτά τα δίκτυα χρησιμοποιούν τουλάχιστον σε ένα επίπεδο την πράξη της συνέλιξης ονομάζονται συνελικτικά.

3.1 Βαθιά μάθηση

Αναμφισβήτητα, η πιο σημαντική κοινή συνισταμένη των μεθόδων βαθιάς μάθησης είναι η εκπαίδευση από άκρο σε άκρο. Δηλαδή, αντί να συναρμολογούμε ένα σύστημα με βάση στοιχεία που ρυθμίζονται μεμονωμένα, κάποιος κατασκευάζει το σύστημα και στη συνέχεια συντονίζει την απόδοσή του από κοινού. Για παράδειγμα, στην όραση υπολογιστών οι επιστήμονες συνήθιζαν να διαχωρίζουν τη διαδικασία σχεδιασμού χαρακτηριστικών από τη διαδικασία κατασκευής μοντέλων μηχανικής μάθησης. Στο παρελθόν η εφαρμογή της μηχανικής μάθησης σε τέτοια προβλήματα γινόταν με τρόπους μετασχηματισμού των δεδομένων σε κάποια μορφή που να είναι προσιτή σε ρηχά μοντέλα. Δυστυχώς, υπάρχουν μόνο τόσα λίγα που ο άνθρωπος μπορεί να επιτύχει με την εφευρετικότητα σε σύγκριση με μια συνεπή αξιολόγηση πάνω σε εκατομμύρια επιλογές που πραγματοποιούνται αυτόματα από έναν αλγόριθμο.

Όταν ανέλαβε η βαθιά μάθηση, αυτοί οι εξαγωγείς χαρακτηριστικών αντικαταστάθηκαν από αυτόματα ρυθμισμένα φίλτρα, αποδίδοντας ανώτερη ακρίβεια. Έτσι, ένα βασικό πλεονέκτημα της βαθιάς μάθησης είναι ότι αντικαθιστά όχι μόνο τα ρηχά μοντέλα και τις διαδικασίες χαρακτηριστικών αλλά και την απαιτητική διαδικασία σχεδιασμού χαρακτηριστικών. Επιπλέον, με την αντικατάσταση μεγάλου μέρους της προεπεξεργασίας συγκεκριμένου τομέα, η βαθιά μάθηση έχει εξαλείψει πολλά από τα όρια που διαχώριζαν προηγουμένως την όραση υπολογιστών, την ομιλία αναγνώριση, την επεξεργασία φυσικής γλώσσας, την ιατρική πληροφορική και άλλους τομείς εφαρμογών, προσφέροντας ένα ενιαίο σύνολο εργαλείων για την αντιμετώπιση διαφορετικών προβλημάτων.

3.1.1 Πλεονεκτήματα

Ακολουθούν μερικά από τα πλεονεκτήματα της βαθιάς μάθησης:

1. Δεν υπάρχει ανάγκη να επισημανθούν τα δεδομένα

Ένα από τα κύρια πλεονεκτήματα της βαθιάς μάθησης είναι η ικανότητα χειρισμού πολύπλοκων δεδομένων και σχέσεων. Μπορείτε να χρησιμοποιήσετε τη βαθιά μάθηση για να κάνετε λειτουργίες τόσο με δεδομένα με ετικέτες όσο και με δεδομένα χωρίς ετικέτες. Σε περιπτώσεις όπου η επισήμανση των δεδομένων είναι ανέφικτη, τα μοντέλα βαθιάς μάθησης μπορούν να εξάγουν σημαντικές πληροφορίες και μοτίβα. Αξιοποιώντας την εγγενή δομή και τα χαρακτηριστικά που υπάρχουν στα δεδομένα, οι αλγόριθμοι βαθιάς μάθησης μπορούν να αποκαλύψουν κρυφές σχέσεις, να ανακαλύψουν λεπτά μοτίβα και να κάνουν προβλέψεις ή ταξινομήσεις.

Σε πολλές καταστάσεις του πραγματικού κόσμου, η απόκτηση επισημασμένων δεδομένων μπορεί να είναι μια χρονοβόρα και δαπανηρή διαδικασία. Η χειροκίνητη επισήμανση τεράστιων ποσοτήτων δεδομένων, όπως εικόνες, μπορεί να απαιτήσει σημαντική ανθρώπινη προσπάθεια και πόρους.

2. Αποτελεσματική στην παραγωγή αποτελεσμάτων υψηλής ποιότητας

Μόλις διδαχθεί σωστά, ένα μοντέλο βαθιάς μάθησης μπορεί να εκτελέσει χιλιάδες επαναλαμβανόμενες, καθημερινές δραστηριότητες σε κλάσμα του χρόνου. Εκτός εάν η εκπαίδευση περιλαμβάνει ακατέργαστα δεδομένα που δεν αντικατοπτρίζουν το πρόβλημα, η ποιότητα της εργασίας δεν υποβαθμίζεται ποτέ.

3. Δεν υπάρχει ανάγκη για μηχανική των χαρακτηριστικών

Η πρακτική της εξαγωγής χαρακτηριστικών από ακατέργαστα δεδομένα για τον καλύτερο προσδιορισμό του υποκείμενου προβλήματος είναι γνωστή ως μηχανική χαρακτηριστικών. Έχει κρίσιμο ρόλο στη μηχανική μάθηση, καθώς ενισχύει την ακρίβεια του μοντέλου. Η προσέγγιση αυτή μπορεί κατά καιρούς να χρειάζεται εμπειρογνωμοσύνη στον τομέα. Η ικανότητα της βαθιάς μάθησης να αναλαμβάνει μόνη της τη μηχανική των χαρακτηριστικών είναι ένα από τα κύρια πλεονεκτήματά της έναντι των συμβατικών μεθόδων μηχανικής μάθησης.

4. Η σχέση κόστους-αποτελεσματικότητας

Η ανάπτυξη μοντέλων βαθιάς μάθησης μπορεί να είναι δαπανηρή, αλλά, μόλις εκπαιδευτούν, γίνονται εφικτά για τον οργανισμό.

Το κόστος μιας λανθασμένης πρόβλεψης ή ενός ελαττώματος προϊόντος είναι ένα τεράστιο και σοβαρό ζήτημα στις επιχειρήσεις. Συχνά ξεπερνά τα έξοδα ανάπτυξης μοντέλων βαθιάς μάθησης που θα αποφύγουν αυτά τα προβλήματα.

5. Η βαθιά μάθηση υποστηρίζεται τόσο σε παράλληλες όσο και σε κατανεμημένες λειτουργίες

Μπορεί να χρειαστούν ημέρες για να μάθει ένα μοντέλο τις παραμέτρους που το απαρτίζουν. Οι παράλληλοι και οι κατανεμημένοι αλγόριθμοι ανακουφίζουν αυτό το πρόβλημα, επιτρέποντας την εκπαίδευση των μοντέλων βαθιάς μάθησης σημαντικά ταχύτερα.

Ανάλογα με το μέγεθος του συνόλου δεδομένων εκπαίδευσης και την ικανότητα επεξεργασίας GPU, μπορείτε να ολοκληρώσετε την εκπαίδευση σε μια ημέρα με μόλις δύο ή τρεις υπολογιστές ή με 20 υπολογιστές.

6. Επεκτασιμότητα

Η βαθιά μάθηση είναι εξαιρετικά κλιμακούμενη λόγω της ικανότητάς της να αναλύει μεγάλο όγκο δεδομένων και να πραγματοποιεί οικονομικά αποδοτικά πολυάριθμους υπολογισμούς.

Η βαθιά μάθηση είναι επίσης ικανή να χειρίζεται την ενδομεταβλητότητα, πράγμα που σημαίνει ότι μπορεί να διαφοροποιήσει τις μικροσκοπικές διαφορές στα δεδομένα. Αυτό έχει άμεση επίδραση στην παραγωγικότητα, την αρθρωτότητα και τη φορητότητα του μοντέλου.

3.1.2 Μειονεκτήματα

Η βαθιά μάθηση έχει επίσης ορισμένα μειονεκτήματα. Ακολουθούν μερικά από αυτά:

1. Μαζική απαίτηση δεδομένων

Καθώς τα συστήματα βαθιάς μάθησης μαθαίνουν σταδιακά, απαιτούνται τεράστιοι όγκοι δεδομένων για την εκπαίδευσή τους. Για παράδειγμα, για να επιτευχθούν τα επιθυμητά αποτελέσματα από έναν αλγόριθμο αναγνώρισης φωνής, απαιτούνται δεδομένα που καλύπτουν πολυάριθμες γλώσσες, δημογραφικά στοιχεία και χρονικά πλαίσια.

Εταιρείες του τύπου Google και Microsoft έχουν την δυνατότητα να αποθηκεύουν μεγάλες ποσότητες δεδομένων, κάτι που δεν είναι εφικτό για μικρές επιχειρήσεις με στέρεες ιδέες να κάνουν το ίδιο. Επιπλέον, η διαθεσιμότητα δεδομένων για ορισμένους κλάδους μπορεί να είναι περιορισμένη, περιορίζοντας τη βαθιά μάθηση σε αυτόν τον τομέα.

2. Υψηλή ισχύς επεξεργασίας

Ένα άλλο ζήτημα με τη βαθιά μάθηση είναι ότι απαιτεί μεγάλη υπολογιστική ισχύ. Τα σκληρά λογισμικά υψηλής απόδοσης αποτελούνται από πολυπύρηνες μονάδες επεξεργασίας γραφικών που απαιτούν πολλή ηλεκτρική ενέργεια, καθιστώντας τα μια ακριβή επένδυση.

Η βαθιά μάθηση είναι επίσης μια διαδικασία που καταναλώνει μνήμη, οπότε πρέπει να διαθέτουμε αρκετό χώρο στη μνήμη για να αποδίδει καλά το μοντέλο.

3. Δυσκολίες με δεδομένα πραγματικής ζωής

Τα μοντέλα βαθιάς μάθησης που αποδίδουν καλά σε σύνολα δεδομένων αναφοράς ενδέχεται να δυσκολεύονται όταν εφαρμόζονται σε σύνολα δεδομένων του πραγματικού κόσμου.

Σκεφτείτε έναν αλγόριθμο βαθιάς μάθησης που μαθαίνει ότι τα σχολικά λεωφορεία είναι συνήθως κίτρινα. Εάν τα λεωφορεία είναι βαμμένα μπλε, το μοντέλο δυσκολεύεται να εντοπίσει το λεωφορείο. Αποτυγχάνουν να αποδώσουν καλά σε ένα άγνωστο περιβάλλον, όπως κάθε άλλος αλγόριθμος.

4. Πρόβλημα του μαύρου κουτιού

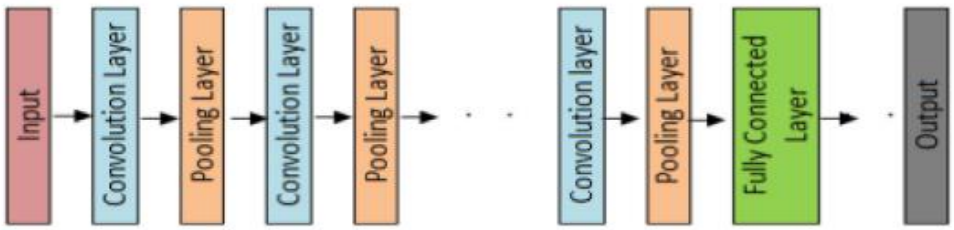
Ένα μαύρο κουτί είναι μια συσκευή ή ένα σύστημα που σας επιτρέπει να βλέπετε την είσοδο/έξοδο αλλά όχι τις ενδιάμεσες λειτουργίες. Οι αλγόριθμοι βαθιάς μάθησης αντιμετωπίζουν επίσης προβλήματα μαύρου κουτιού, γεγονός που καθιστά δύσκολη την αποσφαλμάτωση και την κατανόηση του τρόπου με τον οποίο λαμβάνουν αποφάσεις.

Αφήνει επίσης τους προγραμματιστές άναυδους όταν προσπαθούν να κατανοήσουν γιατί ορισμένες πτυχές αποτυγχάνουν. Γενικά, οι αλγόριθμοι βαθιάς μάθησης κοσκινίζουν εκατομμύρια σημεία δεδομένων για να βρουν μοτίβα και συσχετίσεις που συχνά περνούν απαρατήρητες από τους ανθρώπινους εμπειρογνώμονες.

Ενώ αυτό μπορεί να μην αποτελεί πρόβλημα κατά την εκτέλεση ασήμαντων εργασιών, σε περιπτώσεις όπως η ανίχνευση όγκων, ο γιατρός πρέπει να γνωρίζει γιατί το μοντέλο σημείωσε ορισμένες περιοχές και γιατί δεν το έκανε για άλλες σε μια έκθεση σάρωσης.

3.2 - Συνελικτικά νευρωνικά δίκτυα

Τα συνελικτικά νευρωνικά δίκτυα έχουν σχεδιαστεί για να λειτουργούν με εισόδους με δομή πλέγματος, οι οποίες έχουν ισχυρές χωρικές εξαρτήσεις σε τοπικές περιοχές του πλέγματος. Το πιο προφανές παράδειγμα δεδομένων με δομή πλέγματος είναι μια δισδιάστατη εικόνα. Αυτός ο τύπος δεδομένων παρουσιάζει επίσης χωρικές εξαρτήσεις, επειδή οι γειτονικές χωρικές θέσεις σε μια εικόνα έχουν συχνά παρόμοιες τιμές χρώματος των μεμονωμένων εικονοστοιχείων. Μια πρόσθετη διάσταση αποτυπώνει τα διαφορετικά χρώματα, γεγονός που δημιουργεί ένα τρισδιάστατο όγκο εισόδου. Επομένως, τα χαρακτηριστικά σε ένα νευρωνικό δίκτυο συνελικτικού τύπου έχουν εξαρτήσεις μεταξύ τους με βάση τις χωρικές αποστάσεις. Άλλες μορφές διαδοχικών δεδομένων όπως το κείμενο, οι χρονοσειρές και οι ακολουθίες μπορούν επίσης να θεωρηθούν ειδικές περιπτώσεις δομημένων με πλέγμα δεδομένων με διάφορους τύπους σχέσεων μεταξύ γειτονικών στοιχείων. Η συντριπτική πλειονότητα των εφαρμογών των νευρωνικών δικτύων συμβολής εστιάζει σε δεδομένα εικόνας, αν και μπορεί επίσης να χρησιμοποιηθεί αυτά τα δίκτυα για όλους τους τύπους χρονικών, χωρικών και χωροχρονικών δεδομένων. Το βασικό εννοιολογικό μοντέλο του CNN φαίνεται στο παρακάτω σχήμα 1.

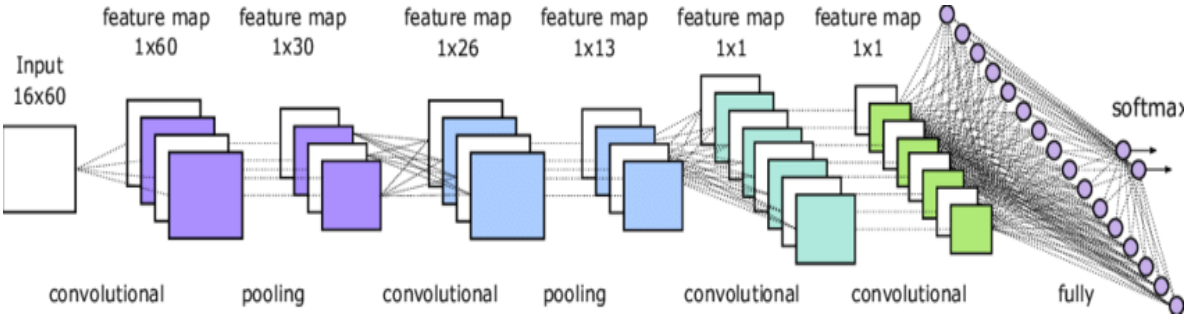


Εικόνα 1 : Βασικό εννοιολογικό μοντέλο του CNN.

Το συνελικτικό νευρωνικό δίκτυο λειτουργεί όπως ένα παραδοσιακό νευρωνικό δίκτυο τροφοδότησης προς τα εμπρός δίκτυο, με τη διαφορά ότι οι λειτουργίες στα στρώματά του είναι χωρικά οργανωμένες με αραιές (και προσεκτικά σχεδιασμένες) συνδέσεις μεταξύ των στρωμάτων. Σύμφωνα και με το παραπάνω σχήμα οι τρεις τύποι στρωμάτων που είναι συνήθως είναι η συνέλιξη, η συγκέντρωση και κάποιος τύπος συνάρτησης. Επιπλέον, ένα τελικό σύνολο στρωμάτων είναι συχνά πλήρως συνδεδεμένο και αντιστοιχίζεται με έναν ειδικό για την εφαρμογή τρόπο σε ένα σύνολο κόμβων εξόδου. Στη συνέχεια, θα περιγράψουμε καθέναν από τους διαφορετικούς τύπους λειτουργιών και στρωμάτων και τον τυπικό τρόπο με τον οποίο τα στρώματα αυτά διαδέχονται το ένα το άλλο σε ένα συνελικτικό νευρωνικό δίκτυο.

3.2.1 – Συνελικτικό στρώμα

Το πρώτο στρώμα (Convolutional Layer) χρησιμοποιείται για την εξαγωγή των διαφόρων χαρακτηριστικών από τις εικόνες εισόδου. Σε αυτό το στρώμα εκτελείται η μαθηματική πράξη της συνέλιξης μεταξύ της εικόνας εισόδου και ενός φίλτρου συγκεκριμένου μεγέθους MxM. Ως έξοδο έχουμε τον χάρτη χαρακτηριστικών, ο οποίος μας δίνει πληροφορίες σχετικά με την εικόνα, όπως είναι οι γωνίες και οι ακμές. Αργότερα, αυτός ο χάρτης χαρακτηριστικών τροφοδοτείται σε άλλα επίπεδα για να μάθει διάφορα άλλα χαρακτηριστικά της εικόνας εισόδου όπως φαίνεται και στην εικόνα του σχήματος 2.



Εικόνα 2: Cnn μοντέλο με χάρτες χαρακτηριστικών.

3.2.2 – Συγκεντρωτικό στρώμα

Το συγκεντρωτικό στρώμα συνήθως εφαρμόζεται μετά το συνελικτικό. Στόχος αυτού του στρώματος είναι να μειώσει το υπολογιστικό κόστος και συγκεκριμένα να μικρύνει το μέγεθος του χάρτη συνελικτικών χαρακτηριστικών ελαττώνοντας τις συνδέσεις μεταξύ τον στρωμάτων. Η παραπάνω διαδικασία γίνεται ανεξάρτητα για κάθε χάρτη χαρακτηριστικών συνήθως με έναν απο τους παρακάτω τρόπους η ακόμα και συνδυασμό τους:

- Max Pooling: το μεγαλύτερο στοιχείο λαμβάνεται από τον χάρτη χαρακτηριστικών.
- Average Pooling: υπολογίζει το μέσο όρο των στοιχείων σε ένα τμήμα εικόνας προκαθορισμένου μεγέθους.

Το συγκεντρωτικό στρώμα χρησιμεύει συνήθως ως γέφυρα μεταξύ του συνελικτικού στρώματος και του πλήρως συνδεδεμένου.

3.2.3 – Συναρτήσεις ενεργοποίησης

Οι συναρτήσεις ενεργοποίησης προσθέτουν μη γραμμικότητα στο δίκτυο ενώ χρησιμοποιούνται για την εκμάθηση και την προσέγγιση κάθε είδους συνεχούς και σύνθετης σχέσης μεταξύ των μεταβλητών του δικτύου.

Υπάρχουν διάφορες ευρέως χρησιμοποιούμενες συναρτήσεις ενεργοποίησης, όπως οι συναρτήσεις ReLU, Softmax, tanH και Sigmoid. Για δυαδική ταξινόμηση προτιμώνται οι συναρτήσεις Sigmoid και Softmax, ενώ για ταξινόμηση πολλαπλών κλάσεων χρησιμοποιείται

γενικά η Softmax. Συμπεραίνουμε λοιπόν ότι οι συναρτήσεις ενεργοποίησης σε ένα μοντέλο CNN καθορίζουν αν ένας νευρώνας πρέπει να ενεργοποιηθεί ή όχι.

Αναλύουμε τις 4 συναρτήσεις ενεργοποίησης που αναφέρθηκαν:

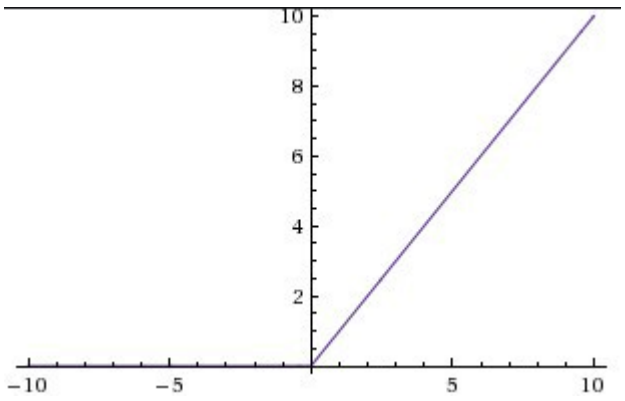
- ReLU

Σημαίνει διορθωμένη γραμμική μονάδα. Είναι η πιο ευρέως χρησιμοποιούμενη συνάρτηση ενεργοποίησης. Εφαρμόζεται κυρίως στα κρυφά στρώματα του νευρωνικού δικτύου ενώ η γραμμική της φύση μας επιτρέπει να κάνουμε οπισθοδιάδοση των σφαλμάτων και να έχουμε πολλαπλά στρώματα νευρώνων που ενεργοποιούνται από τη συνάρτηση ReLU. Η ReLu είναι λιγότερο δαπανηρή υπολογιστικά από την tanh και την sigmoid, καθώς περιλαμβάνει απλούστερες μαθηματικές πράξεις.

Εξίσωση :

relu(x) = max(0,x).

Δίνει μια έξοδο x εάν το x είναι θετικό και 0 διαφορετικά.

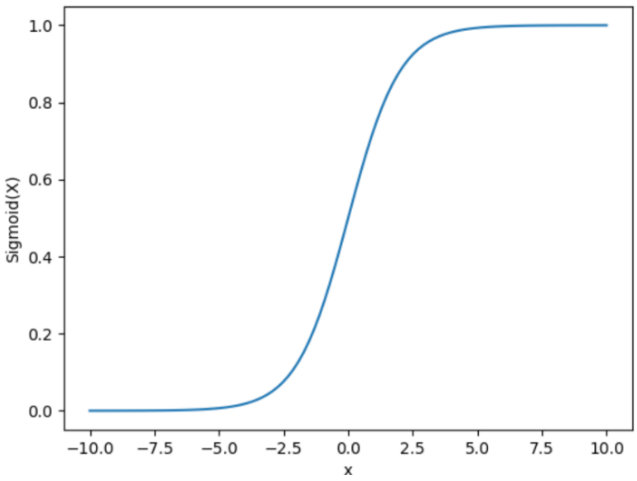


Εικόνα 3: Εικόνα 3: Γραφική παράσταση ReLU.

- Sigmoid

Πρόκειται για μια συνάρτηση η οποία απεικονίζεται ως γράφημα σχήματος "S". Μικρές αλλαγές στο x θα επιφέρουν επίσης μεγάλες αλλαγές στην τιμή του Y. Συνήθως χρησιμοποιείται στο επίπεδο εξόδου μιας δυαδικής ταξινόμησης, όπου το αποτέλεσμα είναι είτε 0 είτε 1, καθώς η τιμή για τη σιγμοειδή συνάρτηση βρίσκεται μεταξύ 0 και 1.

Εξίσωση: sigmoid(x) = 1/(1 + e-x)



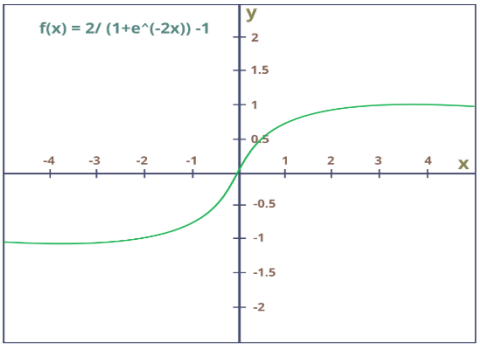
Εικόνα 4: Γραφική παράσταση Sigmoid.

- Tanh

Γνωστή και ως εφαπτομενική υπερβολική συνάρτηση η Tanh λειτουργεί σχεδόν πάντα καλύτερα από τη σιγμοειδή συνάρτηση. Στην πραγματικότητα είναι μαθηματικά μετατοπισμένη έκδοση της σιγμοειδούς συνάρτησης. Και οι δύο είναι παρόμοιες και μπορούν να προκύψουν η μία από την άλλη. Χρησιμοποιείται κυρίως στα κρυφά στρώματα ενός νευρωνικού δικτύου, καθώς οι τιμές της κυμαίνονται μεταξύ -1 και 1, επομένως ο μέσος όρος για το κρυφό στρώμα είναι 0 ή πολύ κοντά σε αυτόν, επομένως βοηθάει στην κεντροποίηση των δεδομένων φέρνοντας τον μέσο όρο κοντά στο 0. Αυτό κάνει τη μάθηση για το επόμενο στρώμα πολύ πιο εύκολη.

Εξίσωση:

$$tanh(x) = \frac{2}{1+e^{-2x}} - 1$$



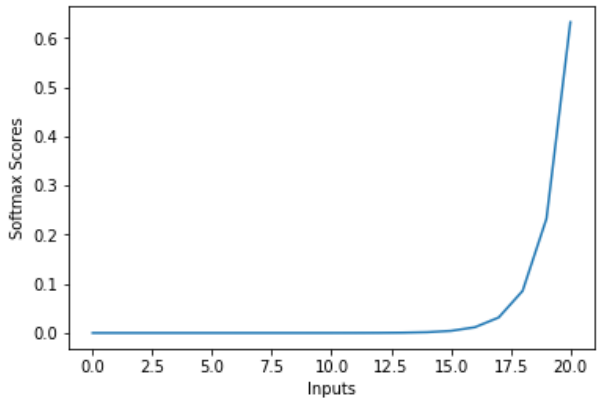
Εικόνα 5: Γραφική παράσταση Tanh.

- Softmax

Είναι κατάλληλη για τον χειρισμό πολλαπλών κλάσεων. Η συνάρτηση softmax συναντάται συνήθως στο επίπεδο εξόδου των προβλημάτων ταξινόμησης εικόνων. Συμπιέζει τις εξόδους για κάθε κλάση μεταξύ 0 και 1 και διαιρεί επίσης με το άθροισμα των εξόδων.

Εξίσωση:

$$\text{softmax}(x) = \frac{e^x}{\sum e^x}$$



Εικόνα 6: Γραφική Παράσταση Softmax.

3.2.4 - Πλήρους συνδεσμολογίας στρώμα.

Το στρώμα FC είναι υπεύθυνο για τη σύνδεση των νευρώνων μεταξύ δύο διαφορετικών στρωμάτων χρησιμοποιώντας βάρη, προκαταλήψεις και νευρώνες. Συνήθως, αυτά τα στρώματα τοποθετούνται πριν από το στρώμα εξόδου, σχηματίζοντας τα τελευταία στρώματα μιας αρχιτεκτονικής CNN. Κατά τη διάρκεια αυτής της διαδικασίας, η εικόνα εισόδου από τα προηγούμενα στρώματα ισοπεδώνεται και μεταδίδεται στο στρώμα FC. Στη συνέχεια, το διάνυσμα περνάει από διάφορα πρόσθετα στρώματα FC, όπου συνήθως εκτελούνται μαθηματικές πράξεις, ξεκινώντας τη διαδικασία ταξινόμησης. Δύο πλήρως συνδεδεμένα στρώματα χρησιμοποιούνται συνήθως για την ενίσχυση της απόδοσης, καθώς μπορούν να αποδώσουν καλύτερα από ένα μόνο συνδεδεμένο στρώμα. Η εφαρμογή αυτών των στρωμάτων σε μια αρχιτεκτονική CNN μπορεί να μειώσει το επίπεδο της απαιτούμενης ανθρώπινης επίβλεψης.

3.3 - Προβλήματα και τρόποι αντιμετώπισης εκπαίδευσης CNN μοντέλου.

Όπως έχει ήδη αναφερθεί αρκετές φορές εμφανίζονται διάφορα πρακτικά ζητήματα που αφορούν την εκπαίδευση και στέκονται εμπόδιο στην αποτελεσματική ανάπτυξη ενός συνελικτικού νευρωνικού δικτύου. Παρακάτω θα αναλύσουμε τα πιο συνήθη και θα δούμε λύσεις που μπορούν να μας διευκολύνουν για την δημιουργία του μοντέλου.

3.3.1 – Προβλήματα.

Μερικά απο τα πιο συχνά προβλήματα που εμφανίζονται κατά την διάρκεια εκπαίδευσης είναι:

Υπερπροσαρμογή:

Η υπερπροσαρμογή συμβαίνει όταν ένα μοντέλο μηχανικής μάθησης παρουσιάζει εξαιρετική απόδοση στα δεδομένα εκπαίδευσης, αλλά έχει κακή απόδοση όταν αντιμετωπίζει νέα δεδομένα (δεδομένα δοκιμής). Σε τέτοιες περιπτώσεις, το μοντέλο μαθαίνει τις συγκεκριμένες λεπτομέρειες και το θόρυβο στα δεδομένα εκπαίδευσης που δεν γενικεύονται στα νέα δεδομένα, επηρεάζοντας έτσι τη συνολική απόδοση του μοντέλου. Η υπερπροσαρμογή συνδέεται συνήθως με μοντέλα που έχουν χαμηλή μεροληψία και υψηλή διακύμανση.

Υποπροσαρμογή:

Ένα μοντέλο με ελλιπή προσαρμογή είναι ένα μοντέλο που δεν έχει μάθει αποτελεσματικά τα υποκείμενα μοτίβα στα δεδομένα εκπαίδευσης και, ως εκ τούτου, έχει κακές επιδόσεις όταν του παρουσιάζονται νέα δεδομένα. Με άλλα λόγια, το μοντέλο αποτυγχάνει να συλλάβει την ουσία των δεδομένων, οδηγώντας σε αναξιόπιστες προβλέψεις. Συνήθως, ένα μοντέλο με ελλιπή προσαρμογή παρουσιάζει κακή απόδοση τόσο στα δεδομένα εκπαίδευσης όσο και στα δεδομένα δοκιμής. Η υποπροσαρμογή συνδέεται με μοντέλα που έχουν υψηλή μεροληψία και χαμηλή διακύμανση.

Εξαφανιζόμενες κλίσεις:

Κατά τη διάρκεια του αλγορίθμου οπισθοδιάδοσης, οι κλίσεις μπορούν να γίνονται όλο και μικρότερες και να προσεγγίζουν το μηδέν καθώς ο αλγόριθμος προχωρά προς το επίπεδο εισόδου, αφήνοντας τα βάρη των αρχικών ή κατώτερων επιπέδων σχεδόν αμετάβλητα. Αυτό μπορεί να αποτρέψει τη σύγκλιση της βαθμωτής καθόδου στο βέλτιστο και αναφέρεται ως πρόβλημα εξαφανιζόμενων κλίσεων.

Εκρηκτικές κλίσεις:

Σε ορισμένες περιπτώσεις, οι κλίσεις μπορεί να γίνονται ολοένα και μεγαλύτερες καθώς ο αλγόριθμος οπισθοδιάδοσης προχωρά, οδηγώντας σε πολύ μεγάλες ενημερώσεις βαρών και προκαλώντας απόκλιση της κάθοδος κλίσης. Αυτό αναφέρεται ως πρόβλημα εκρηκτικής κλίσης.

Ανισορροπία κλάσεων:

Το ζήτημα της ανισορροπίας κλάσεων προκύπτει όταν ορισμένες κλάσεις σε ένα σύνολο δεδομένων έχουν σημαντικά περισσότερες περιπτώσεις από άλλες. Αυτό μπορεί να οδηγήσει στο να μεροληπτούν οι τυπικοί ταξινομητές προς τις μεγαλύτερες κλάσεις και να αποτυγχάνουν να λάβουν επαρκώς υπόψη τους τις μικρότερες.

Ανεπαρκή δεδομένα:

Σε πολλές περιπτώσεις συνηθίζεται να μην υπάρχουν αρκετά δεδομένα για την εκπαίδευση του μοντέλου. Όταν συμβαίνει αυτό το μοντέλο δεν εκπαιδεύεται σωστά η δίνει φτωχά αποτελέσματα.

Αδύναμο μηχάνημα: Για να γίνει η εκπαίδευση χρειάζεται_υπολογιστής με κάρτα γραφικών που μπορεί να ανταπεξέλθει στην διαδικασία. Σε ένα CNN εκτελούνται πολλές πράξεις πινάκων προσθαφαιρέσεις και πολλαπλασιασμοί. Οι GPU είναι ιδιαίτερα κατάλληλες

για την εκτέλεση τέτοιου είδους πράξεων πινάκων, καθώς είναι σχεδιασμένες να εκτελούν πολλούς υπολογισμούς παράλληλα. Αυτό σημαίνει ότι μπορούν να επεξεργάζονται γρήγορα μεγάλες ποσότητες δεδομένων, πράγμα που είναι κρίσιμο για την εκπαίδευση μοντέλων βαθιάς μάθησης. Μια GPU υπερτερεί απο μια CPU σε ότι αφορά την εκπαίδευση CNN. Συμπεραίνουμε πως χωρίς έναν καλό υπολογιστή η δημιουργία CNN μοντέλου μπορεί να γίνει χρονοβόρα και επίπονη με σφάλματα ενώ μπορεί και να μην είναι δυνατή η ολοκλήρωση της εκπαίδευσης.

3.3.2 – Τρόποι αντιμετώπισης.

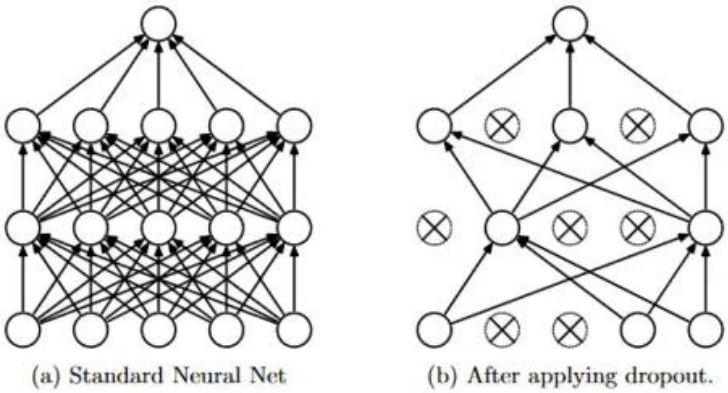
Αντιμετώπιση υπερπροσαρμογής:

Το τυχαίο ανακάτεμα δεδομένων είναι κοινή πρακτική σε αυτόν τον τομέα εκπαίδευσης πριν τα χωρίσουμε για να εξαλειφθούν πιθανές προκαταλήψεις κατά την προετοιμασία τους.

Η αύξηση δεδομένων (data augmentation) για να μειωθεί το χάσμα γενίκευσης (γενίκευση χάσμα: το χάσμα μεταξύ της απόδοσης του μοντέλου στο σετ εκπαίδευσης και του συνόλου δοκιμών απόδοσης μοντέλου).

Η τακτοποίηση (regularization) αναγκάζει το νευρωνικό δίκτυο να γίνει απλούστερο. Βελτιστοποιεί το μοντέλο τιμωρώντας πολύπλοκα μοντέλα, ελαχιστοποιώντας έτσι την απώλεια και την πολυπλοκότητα.

Η τακτοποίηση εγκατάλειψης αγνοεί ένα τυχαίο υποσύνολο των μονάδων σε αυτό το επίπεδο και θέτει το βάρος του στο μηδέν κατά τη διάρκεια αυτής της φάσης εκπαίδευσης. Λόγω του Dropout, δύο νευρώνες μπορεί να μην εμφανίζονται πάντα σε ένα δίκτυο εγκατάλειψης κάθε φορά. Με αυτόν τον τρόπο, η ενημέρωση των βαρών δεν εξαρτάται πλέον από την αλληλεπίδραση άρρητων κόμβων με σταθερές σχέσεις. Η εγκατάλειψη αναγκάζει το δίκτυο να μάθει πιο ισχυρά χαρακτηριστικά, τα οποία υπάρχουν επίσης σε τυχαία υποσύνολα άλλων νευρώνων. Οι εγκαταλείψεις μπορούν να μειώσουν την πολυπλοκότητα του μοντέλου νευρωνικού δικτύου μας, αποτρέποντας έτσι την υπερπροσαρμογή.



Εικόνα 7: Dropout

Αντιμετώπιση υποπροσαρμογής:

Εκπαίδευση για περισσότερες εποχές: Η αύξηση του αριθμού των εποχών εκπαίδευσης μπορεί να δώσει στο μοντέλο περισσότερο χρόνο για να μάθει τα υποκείμενα μοτίβα στα δεδομένα. Ωστόσο, αυτή η προσέγγιση θα πρέπει να χρησιμοποιείται με προσοχή, καθώς η εκπαίδευση για πάρα πολλές εποχές μπορεί να οδηγήσει σε υπερβολική προσαρμογή.

Ένα CNN μπορεί να είναι ανεπαρκές εάν η αρχιτεκτονική του είναι πολύ απλή για να μάθει τα πολύπλοκα χαρακτηριστικά των δεδομένων. Για να διορθωθεί αυτό, μπορούμε να αυξήσουμε την πολυπλοκότητα του μοντέλου προσθέτοντας περισσότερα επίπεδα, αυξάνοντας τον αριθμό των φίλτρων στα συνελικτικά επίπεδα ή αυξάνοντας τον αριθμό των νευρώνων στα πλήρως συνδεδεμένα επίπεδα.

Τροποποίηση του ρυθμού εκμάθησης: Η προσαρμογή του ρυθμού εκμάθησης μπορεί να βοηθήσει στη βελτίωση της απόδοσης ενός CNN. Η μείωση του ρυθμού εκμάθησης μπορεί να επιτρέψει στο μοντέλο να κάνει μικρότερες προσαρμογές στα βάρη και τις μεροληψίες του, οι οποίες μπορούν να βοηθήσουν στην αποφυγή του να κολλήσει στα τοπικά ελάχιστα.

Η αύξηση των δεδομένων (data augmentation), η τακτοποίηση (regularization) και η τακτοποίηση εγκατάλειψης (προσθήκη dropout layer) όπως προαναφέρθηκαν και στην περίπτωση της υπερπροσαρμογής επίσης βοηθάνε.

Αντιμετώπιση εξαφανιζόμενης κλίσης:

Το πρόβλημα της κλίσης εξαφάνισης προκαλείται από την παράγωγο της συνάρτησης ενεργοποίησης που χρησιμοποιείται για τη δημιουργία του νευρωνικού δικτύου. Η απλούστερη λύση στο πρόβλημα είναι η αντικατάσταση της λειτουργίας ενεργοποίησης του δικτύου συνήθως με συνάρτηση ενεργοποίησης ReLU.

Αντιμετώπιση εκρηκτικής κλίσης:

Στα βαθιά νευρωνικά δίκτυα, οι εκρηκτικές κλίσεις μπορούν να αντιμετωπιστούν επανασχεδιάζοντας το δίκτυο ώστε να έχει λιγότερα επίπεδα. Μπορεί επίσης να υπάρχει κάποιο όφελος από τη χρήση μικρότερου μεγέθους παρτίδας (batch size) κατά την εκπαίδευση του δικτύου. Στα επαναλαμβανόμενα νευρωνικά δίκτυα, η ενημέρωση σε λιγότερα προηγούμενα χρονικά βήματα κατά τη διάρκεια της εκπαίδευσης, που ονομάζεται περικομμένη οπισθοδιάδοση στο χρόνο, μπορεί να μειώσει το πρόβλημα της έκρηξης της κλίσης.

Μια άλλη προσέγγιση, εάν εξακολουθούν να εμφανίζονται εκρηκτικές κλίσεις, είναι ο έλεγχος του μεγέθους των βαρών του δικτύου και η εφαρμογή ποινής στη συνάρτηση απώλειας δικτύου για μεγάλες τιμές βάρους. Αυτό ονομάζεται τακτοποίηση βάρους και συχνά μπορεί να χρησιμοποιηθεί ποινή L1 (απόλυτα βάρη) ή L2 (τετράγωνα βάρη).

Αντιμετώπιση ανισορροπίας κλάσεων:

Η υπερδειγματοληψία της τάξης μειοψηφίας με την αναπαραγωγή των δειγμάτων της μπορεί να βοηθήσει στην εξισορρόπηση της κατανομής της τάξης.

Η υποδειγματοληψία της πλειοψηφικής τάξης αφαιρώντας ορισμένα από τα δείγματά της μπορεί επίσης να βοηθήσει στην εξισορρόπηση της κατανομής της κλάσης.

Εκμάθηση με μεταφορά: Η μάθηση με μεταφορά περιλαμβάνει τη χρήση ενός προεκπαιδευμένου μοντέλου εκπαιδευμένου σε ένα μεγάλο, ποικίλο σύνολο δεδομένων για την εξαγωγή χαρακτηριστικών από τις εικόνες εισόδου. Αυτό μπορεί να βοηθήσει στη βελτίωση

της απόδοσης του μοντέλου στην τάξη μειοψηφίας αξιοποιώντας τη γνώση που αποκτήθηκε από το μεγαλύτερο σύνολο δεδομένων.

Αντιμετώπιση ανεπάρκειας δεδομένων:

Όπως έχει προαναφερθεί η αύξηση δεδομένων (data augmentation) αλλά και η μάθηση με μεταφορά μπορούν να βοηθήσουν στο πρόβλημα της ανεπάρκειας δεδομένων.

Αντιμετώπιση αδύναμου μηχανήματος: Ζούμε σε μια εποχή όπου η χρήση του διαδικτύου παρέχει πολλές λύσεις. Σε περίπτωση που δεν έχουμε πρόσβαση σε μηχάνημα με αξιόλογη GPU υπάρχουν ιστοσελίδες που σου παρέχουν αυτή την δυνατότητα. Κάποιες απο τις πιο γνωστές το Google collabs και το Kaggle που κάνουν χρήση GPU με ικανοποιητικά αποτελέσματα και χρόνους.

Κεφάλαιο 4 - Ανάλυση παραμέτρων CNN μοντέλου και συνόλου δεδομένων.

Πριν πάμε στο πειραματικό μέρος θα γίνει ανάλυση των παραμέτρων ενός CNN μοντέλου για καλύτερη κατανόηση του προγράμματος. Επιπλέον θα δούμε με ποιον τρόπο έγινε η οργάνωση του συνόλου δεδομένων που χρησιμοποιήθηκαν.

4.1 Παράμετροι.

Οι παράμετροι σε ένα τυπικό CNN δίκτυο είναι:

target_size: Αναφέρεται συνήθως στο επιθυμητό μέγεθος των εικόνων εισόδου που θα επεξεργαστεί το δίκτυο (ύψος, πλάτος). Θέτοντας την παράμετρο target_size, μπορούμε να διασφαλίσουμε ότι όλες οι εικόνες εισόδου αλλάζουν μέγεθος σε ένα σταθερό μέγεθος πριν τροφοδοτηθούν στο δίκτυο, ανεξάρτητα από το αρχικό τους μέγεθος. Το να έχουμε ένα δίκτυο με σταθερές εικόνες βοηθάει στην αποδοτικότητα του νευρωνικού δικτύου.

batch_size: Το batch_size αναφέρεται στον αριθμό των δειγμάτων ή εικόνων που διαδίδονται μέσω του δικτύου σε ένα πέρασμα προς τα εμπρός/πίσω. Κατά την εκπαίδευση ενός CNN, τα δεδομένα εκπαίδευσης χωρίζονται συνήθως σε μικρότερες παρτίδες και κάθε παρτίδα τροφοδοτείται στο δίκτυο διαδοχικά κατά τη διάρκεια της εκπαίδευσης. Ένα μεγαλύτερο batch_size μπορεί να οδηγήσει σε ταχύτερους χρόνους εκπαίδευσης, επειδή περισσότερα δείγματα επεξεργάζονται παράλληλα, αλλά απαιτεί επίσης περισσότερη μνήμη για την αποθήκευση των κλίσεων κατά τη διάρκεια της οπισθοδιάδοσης. Ένα μικρότερο batch_size μπορεί να οδηγήσει σε πιο αργούς χρόνους εκπαίδευσης, αλλά μπορεί να βοηθήσει το δίκτυο να συγκλίνει σε μια καλύτερη λύση και να γενικεύει καλύτερα σε νέα δεδομένα. Η εύρεση της βέλτιστης τιμής batch_size για ένα συγκεκριμένο CNN απαιτεί πειραματισμό και ρύθμιση.

class_mode: Η class_mode είναι μια παράμετρος που χρησιμοποιείται σε συνδυασμό με την κλάση ImageDataGenerator της βιβλιοθήκης βαθιάς μάθησης Keras για να καθορίσει πώς το CNN θα πρέπει να ερμηνεύει τις ετικέτες των εικόνων εισόδου κατά τη διάρκεια της εκπαίδευσης και της επικύρωσης. Στη συγκεκριμένη περίπτωση που έχουμε 30 κλάσεις

χρησιμοποιούμε την τιμή categorical η οποία χρησιμοποιείται για προβλήματα ταξινόμησης πολλαπλών κλάσεων.

filters: Τα φίλτρα είναι οι παράμετροι που μαθαίνονται κατά τη διάρκεια της εκπαίδευσης για την εξαγωγή σημαντικών χαρακτηριστικών από τα δεδομένα εισόδου. Η παράμετρος φίλτρα σε ένα επίπεδο CNN καθορίζει τον αριθμό των φίλτρων που εφαρμόζονται στα δεδομένα εισόδου. Για παράδειγμα, αν ορίσουμε filters=32 σε ένα επίπεδο CNN, αυτό σημαίνει ότι 32 φίλτρα θα μαθαίνονται κατά τη διάρκεια της εκπαίδευσης και θα εφαρμόζονται στα δεδομένα εισόδου για την εξαγωγή χαρακτηριστικών.

kernel_size: Το μέγεθος των φίλτρων καθορίζεται από την παράμετρο kernel_size. Για παράδειγμα, αν ορίσουμε kernel_size=3,3 σε ένα επίπεδο CNN, αυτό σημαίνει ότι τα φίλτρα θα είναι πίνακες 3x3 που θα είναι συνεπτυγμένοι με τα δεδομένα εισόδου.

padding: Η παράμετρος padding καθορίζει τον τρόπο με τον οποίο τα δεδομένα εισόδου γεμίζονται για να διασφαλιστεί ότι τα δεδομένα εξόδου έχουν τις ίδιες χωρικές διαστάσεις με τα δεδομένα εισόδου. Δύο συνήθεις τρόποι συμπλήρωσης είναι οι 'valid' και 'same'. Στον τρόπο λειτουργίας 'valid', δεν εφαρμόζεται καμία συμπλήρωση και τα δεδομένα εξόδου έχουν μικρότερες χωρικές διαστάσεις από τα δεδομένα εισόδου. Στη λειτουργία "same", προστίθεται συμπλήρωση στα δεδομένα εισόδου για να εξασφαλιστεί ότι τα δεδομένα εξόδου έχουν τις ίδιες χωρικές διαστάσεις με τα δεδομένα εισόδου.

activation: Η παράμετρος ενεργοποίησης καθορίζει τη συνάρτηση ενεργοποίησης που θα εφαρμοστεί στην έξοδο κάθε νευρώνα σε ένα επίπεδο. Η συνάρτηση ενεργοποίησης εισάγει μη γραμμικότητα στο CNN, η οποία είναι σημαντική για να μπορεί το CNN να μαθαίνει σύνθετα και μη γραμμικά μοτίβα στα δεδομένα εισόδου. Οι συναρτήσεις ενεργοποίησης αναλύθηκαν προηγουμένως στο **3.2.3 – Συναρτήσεις ενεργοποίησης.**

units: Η παράμετρος units χρησιμοποιείται συνήθως σε πλήρως συνδεδεμένα επίπεδα για να καθορίσει τον αριθμό των νευρώνων στο επίπεδο. Ένα πλήρως συνδεδεμένο επίπεδο είναι ένα επίπεδο στο οποίο όλοι οι νευρώνες συνδέονται με κάθε νευρώνα στο προηγούμενο επίπεδο. Πριν από το(τα) πλήρως συνδεδεμένο(-α) στρώμα(-τα), η έξοδος των στρωμάτων συνέλιξης συνήθως ισοπεδώνεται σε ένα διάνυσμα 1D (flattening layer), το οποίο στη συνέχεια περνάει από ένα ή περισσότερα πλήρως συνδεδεμένα στρώματα.

optimizer: Στη μηχανική μάθηση, ο βελτιστοποιητής είναι ένας αλγόριθμος που χρησιμοποιείται για την ενημέρωση των παραμέτρων (βάρη και προκαταλήψεις) ενός μοντέλου κατά τη διάρκεια της εκπαίδευσης για την ελαχιστοποίηση της συνάρτησης απώλειας.

Ορισμένοι από τους βελτιστοποιητές που χρησιμοποιούνται συνήθως στα CNN:

- SGD (Stochastic Gradient Descent): Πρόκειται για έναν απλό και ευρέως χρησιμοποιούμενο βελτιστοποιητή που ενημερώνει τις παραμέτρους προς την κατεύθυνση της αρνητικής κλίσης της συνάρτησης απώλειας. Χρησιμοποιείται συχνά ως βασικός βελτιστοποιητής για σύγκριση με πιο προηγμένους βελτιστοποιητές.
- Adam (Adaptive Moment Estimation): Πρόκειται για έναν βελτιστοποιητή που συνδυάζει τις ιδέες τόσο της ορμής όσο και των προσαρμοστικών ρυθμών μάθησης. Προσαρμόζει τον ρυθμό μάθησης για κάθε παράμετρο με βάση το ιστορικό των κλίσεων και της ορμής.
- RMSprop (Root Mean Square Propagation): Πρόκειται για έναν βελτιστοποιητή που χρησιμοποιεί έναν κινητό μέσο όρο των τετραγωνικών

κλίσεων για την κανονικοποίηση του ρυθμού μάθησης. Μειώνει τον ρυθμό μάθησης για παραμέτρους με μεγάλες κλίσεις και αυξάνει τον ρυθμό μάθησης για παραμέτρους με μικρές κλίσεις.

- Adagrad (προσαρμοστική κλίση): Πρόκειται για έναν βελτιστοποιητή που προσαρμόζει τον ρυθμό μάθησης για κάθε παράμετρο με βάση τις πληροφορίες για την ιστορική κλίση. Δίνει μεγαλύτερο ρυθμό μάθησης στις παραμέτρους που είναι σπάνιες και μικρότερο ρυθμό μάθησης στις παραμέτρους που είναι συχνές.

learning_rate: Ο ρυθμός μάθησης καθορίζει πόσο γρήγορα ή αργά ο βελτιστοποιητής προσπαθεί να ελαχιστοποιήσει τη συνάρτηση απώλειας. Εάν ο ρυθμός μάθησης είναι πολύ χαμηλός, το μοντέλο μπορεί να χρειαστεί πολύ χρόνο για να συγκλίνει σε μια καλή λύση, ενώ εάν είναι πολύ υψηλός, το μοντέλο μπορεί να μην συγκλίνει και να υπερβεί το ελάχιστο.

loss: Η παράμετρος απώλειας χρησιμοποιείται για τον ορισμό της αντικειμενικής συνάρτησης την οποία ο βελτιστοποιητής προσπαθεί να ελαχιστοποιήσει κατά τη διάρκεια της εκπαίδευσης. Η συνάρτηση απώλειας μετρά τη διαφορά μεταξύ της προβλεπόμενης εξόδου του μοντέλου και της πραγματικής εξόδου, δεδομένης της εισόδου. Η επιλογή της συνάρτησης απωλειών εξαρτάται από το εκάστοτε πρόβλημα και τον τύπο της εξόδου που παράγει το μοντέλο. Για παράδειγμα, εάν η έξοδος είναι δυαδική (0 ή 1), μπορεί να χρησιμοποιηθεί η συνάρτηση απώλειας δυαδικής διασταυρούμενης εντροπίας. Εάν η έξοδος είναι κατηγορική (μία από πολλές κλάσεις), μπορεί να χρησιμοποιηθεί η συνάρτηση απώλειας κατηγορικής διασταυρούμενης εντροπίας.

4.2 Οργάνωση συνόλου δεδομένων.

Ένα απο τα σημαντικότερα κομμάτια ενός CNN είναι το σύνολο δεδομένων αφού η εκπαίδευση και η εξέλιξη του μοντέλου βασίζεται εκεί. Όπως έχει προαναφερθεί τα δεδομένα μας θα είναι εικόνες απο γουνοφόρα ζώα. Αρχικά κατεβάσαμε δεδομένα απο διαδικτυακές πηγές (kaggle.com , images.cv). Αποφασίστηκε να χρησιμοποιηθούν 30 κλάσεις με 1000 εικόνες η κάθε μια καθώς είναι σημαντικό οι εικόνες να είναι ομοιόμορφα διαμοιρασμένες για καλύτερη απόδοση του δικτύου. Θα μπορούσε οι εικόνες να είναι ακόμη περισσότερες καθώς αυτό θα εξυπηρετούσε στην εκπαίδευση όμως για πολλές κλάσεις δεν υπήρχε παραπάνω διαθέσιμο υλικό κι εφόσον επιθυμούμε τις κλάσεις με ισάριθμο αριθμό περιοριστήκαμε στον συγκεκριμένο αριθμό.

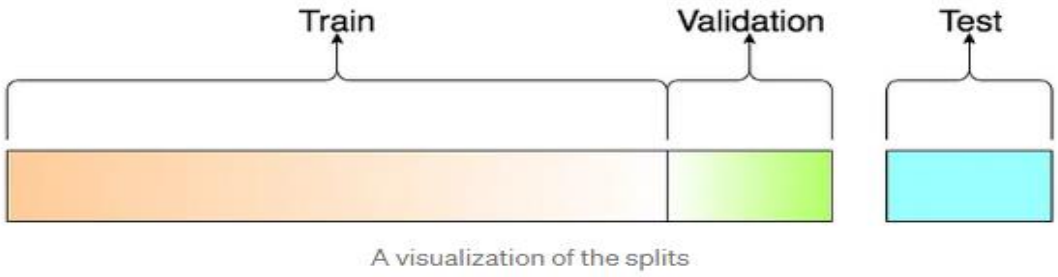
Δεν σταματάμε όμως σε αυτό το σημείο, στη συνέχεια χωρίζουμε την κάθε κατηγορία σε train, validation και test sets και όλα εξυπηρετούν διαφορετικό σκοπό. Αναλύουμε το κάθε σετ:

- Training dataset: Ένα σύνολο δεδομένων εκπαίδευσης αναφέρεται σε ένα σύνολο δειγμάτων δεδομένων που χρησιμοποιούνται για την εκπαίδευση ενός μοντέλου. Είναι το πραγματικό σύνολο δεδομένων που χρησιμοποιείται για την προσαρμογή των παραμέτρων του μοντέλου, όπως τα βάρη και οι προκαταλήψεις στην περίπτωση ενός νευρωνικού δικτύου. Το μοντέλο μαθαίνει από αυτό το σύνολο δεδομένων και προσαρμόζει ανάλογα τις παραμέτρους του.
- Validation dataset: Το σύνολο δεδομένων επικύρωσης είναι ένα δείγμα δεδομένων που χρησιμοποιείται για να παρέχει μια αμερόληπτη αξιολόγηση της απόδοσης ενός μοντέλου που έχει εκπαιδευτεί στο σύνολο δεδομένων εκπαίδευσης. Αυτό το σύνολο δεδομένων χρησιμοποιείται ειδικά για τη λεπτομερή ρύθμιση των

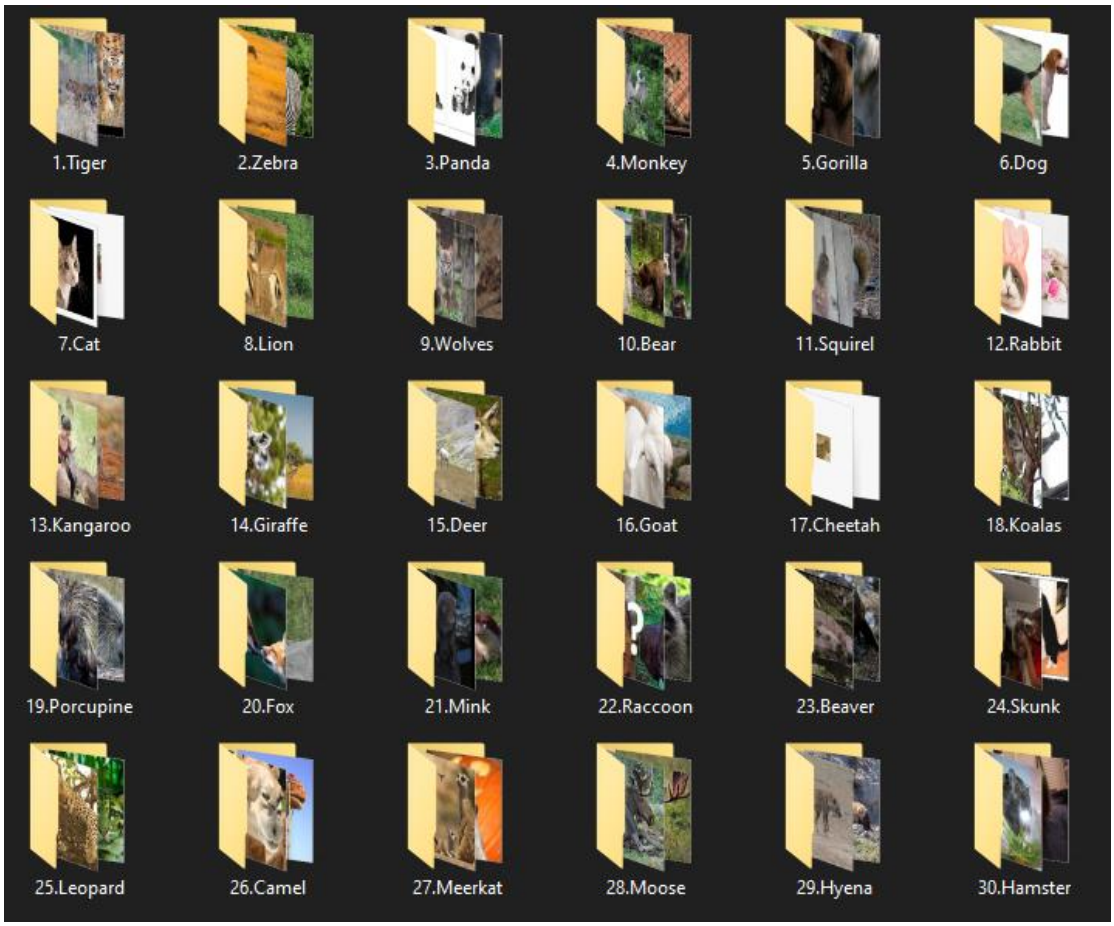
υπερπαραμέτρων του μοντέλου και δεν χρησιμοποιείται για την εκπαίδευση του ίδιου του μοντέλου. Το μοντέλο βλέπει περιστασιακά αυτά τα δεδομένα, αλλά δεν μαθαίνει από αυτά. Αντ' αυτού, τα αποτελέσματα του συνόλου επικύρωσης χρησιμοποιούνται για την ενημέρωση υπερπαραμέτρων υψηλότερου επιπέδου και, ως εκ τούτου, επηρεάζουν έμμεσα το μοντέλο.

- Test dataset: Το σύνολο δεδομένων δοκιμής είναι μια αμερόληπτη αξιολόγηση του τελικού μοντέλου που εκπαιδεύτηκε χρησιμοποιώντας τα σύνολα δεδομένων εκπαίδευσης και επικύρωσης. Χρησιμεύει ως το χρυσό πρότυπο για την αξιολόγηση της απόδοσης του μοντέλου και χρησιμοποιείται μόνο αφού το μοντέλο έχει εκπαιδευτεί πλήρως. Το σύνολο δεδομένων δοκιμής χρησιμοποιείται συνήθως για τη σύγκριση διαφορετικών μοντέλων. Είναι σημαντικό να χρησιμοποιείται ένα καλά επιμελημένο σύνολο δοκιμαστικών δεδομένων που περιλαμβάνει ποικίλα δείγματα που αντιπροσωπεύουν τις διάφορες κλάσεις που θα συναντήσει το μοντέλο σε πραγματικές εφαρμογές. Ενώ το σύνολο δεδομένων επικύρωσης χρησιμοποιείται μερικές φορές ως σύνολο δεδομένων δοκιμής, δεν συνιστάται για βέλτιστη πρακτική.

Ο διαχωρισμός στα σετ που είπαμε εξαρτάται κατά κύριο λόγο από τον συνολικό αριθμό δειγμάτων στα δεδομένα και δεύτερον, από το πραγματικό μοντέλο που εκπαιδεύουμε. Στη συγκεκριμένη περίπτωση μετά απο κάποιες πρώτες προσπάθειες καταλήξαμε σε 800, 100, 100 για train, validation, test set αντίστοιχα. Παρακάτω βλέπουμε μια αναπαράσταση των set και τις κατηγορίες των ζώων που θα χρησιμοποιήσουμε για κάθε set στην συγκεκριμένη διπλωματική.



Εικόνα 8: Διαχωρισμός set δεδομένων.



Εικόνα 9: Κατηγορίες ζώων που θα χρησιμοποιήσουμε για κάθε set.

Στο επόμενο κεφάλαιο θα προχωρήσουμε στην δημιουργία του μοντέλου με το σύνολο δεδομένων που αναφέραμε.

Κεφάλαιο 5 – Πειραματικό μέρος 1ο.

Σε αυτό το μέρος δημιουργήσαμε και εκπαιδεύσαμε 2 διαφορετικά μοντέλα απο το μηδέν. Κάναμε τεστ με τις ίδιες εικόνες σε όλα τα μοντέλα για να δούμε πόσο εύστοχα είναι. Τα σύνολα δεδομένων ανέβηκαν στο google drive κι απο εκεί κάναμε σύνδεση με google collabs το οποίο είναι το εργαλείο με το οποίο δουλέψαμε καθ' όλη την διάρκεια του πειραματικού μέρους.

5.1 Δημιουργία CNN μοντέλου.

Μοντέλο 1:

Σύνδεση με google drive όπου ανεβάσαμε τα δεδομένα.

```
from google.colab import drive
drive.mount('/content/drive')
```

Αντιγραφή και unzip των δεδομένων μας σε φάκελο του google collabs.

```
cp '/content/drive/MyDrive/ColabNotebooks/FurAnimals811.zip'
'/content/FurAnimals811.zip'
!unzip '/content/FurAnimals811.zip' -d '/content/FurAnimals'
```

Εισαγωγή απαραίτητων βιβλιοθηκών.

```
# libaries
import numpy as np # library for working with arrays
import tensorflow as tf # library for deep learning and training deep
neural networks
import pandas as pdb #open-source data manipulation and analysis
library
import cv2 # library for image and video processing, computer vision
algorithms, and machine learning.

from tensorflow import keras #a high-level neural networks API, which
makes it easy to build and train deep learning models
from matplotlib import pyplot as plt # a comprehensive library for
creating static, animated, and interactive visualizations
from keras.preprocessing.image import ImageDataGenerator # library that
can generate batches of tensor image data with real-time data
augmentation.
from keras.layers import Activation, BatchNormalization, Conv2D, Dense,
Dropout, Flatten, MaxPooling2D
from keras.optimizers import Adam
from keras.losses import CategoricalCrossentropy
```

Αποθηκεύουμε σε μεταβλητές το test και validation path.

```
TRAIN = "/content/FurAnimals/Train"
VAL = '/content/FurAnimals/Validate'
```

Μετατροπή των pixel των εικόνων του train set απο εύρος [0,255] σε εύρος [0,1]

```
train_data = ImageDataGenerator(
    rescale = 1. / 255)
```

Δημιουργία τελικού train dataset με ανάλογους παραμέτρους.

```
train_set = train_data.flow_from_directory(TRAIN, target_size=(224,
224), batch_size=32, class_mode='categorical')
```

Όμοια για validation set.

```
val_data = ImageDataGenerator(rescale = 1./255)
val_set = val_data.flow_from_directory(VAL, target_size=(224, 224),
batch_size=32, class_mode='categorical')
```

Δημιουργία μοντέλου και προσθήκη επιπέδων.

```
### build the model ###

#used to create models layer-by-layer
model = tf.keras.models.Sequential()

#This layer creates a convolution kernel that is convolved with the
layer input to produce a tensor of outputs.
model.add(tf.keras.layers.Conv2D(filters=32, kernel_size=(3, 3),
padding='Same', activation='relu', input_shape=[224, 224, 3]))

# maxpool2d calculates the maximum, or largest, value in each patch of
each feature map
model.add(tf.keras.layers.MaxPool2D(pool_size=(2, 2)))

model.add(tf.keras.layers.Conv2D(filters=64, kernel_size=(3, 3),
padding='Same', activation='relu'))
model.add(tf.keras.layers.MaxPool2D(pool_size=(2, 2)))


model.add(tf.keras.layers.Conv2D(filters=128, kernel_size=(3, 3),
padding='Same', activation='relu'))
model.add(tf.keras.layers.MaxPool2D(pool_size=(2, 2)))
```

Τα επίπεδα είναι σε μορφή πίνακα σε αυτό το σημείο γίνεται η μετατροπή απο πολυδιάστατο πίνακα του συνόλου των προηγούμενων επιπέδων σε μονοδιάστατο.

```
model.add(tf.keras.layers.Flatten())
```

Δημιουργία συνδετικού επιπέδου με 512 νευρώνες που ο καθένας συνδέεται με όλους τους νευρώνες του προηγούμενου επιπέδου.

```
model.add(tf.keras.layers.Dense(units=512, activation='relu'))
```

Δημιουργία τελικού συνδετικού επιπέδου με 30 νευρώνες όσες και οι κατηγορίες για κάθε set.

```
model.add(tf.keras.layers.Dense(units=30, activation='softmax'))
```

Εμφάνιση πίνακα μοντέλου που δείχνει όσα έχουμε κάνει μέχρι τώρα.

```
print(model.summary())
```

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 224, 224, 32)      896

 max_pooling2d (MaxPooling2D  (None, 112, 112, 32)     0
 )

 conv2d_1 (Conv2D)           (None, 112, 112, 64)      18496

 max_pooling2d_1 (MaxPooling  (None, 56, 56, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 56, 56, 128)       73856

 max_pooling2d_2 (MaxPooling  (None, 28, 28, 128)      0
 2D)

 flatten (Flatten)           (None, 100352)            0

 dense (Dense)               (None, 512)               51380736

 dense_1 (Dense)             (None, 30)                15390

...
Trainable params: 51,489,374
Non-trainable params: 0
_________________________________________________________________
None
```

Εικόνα 10: Πίνακας μοντέλου 1.

Εφαρμογή Adam optimizer και compile του κώδικα.

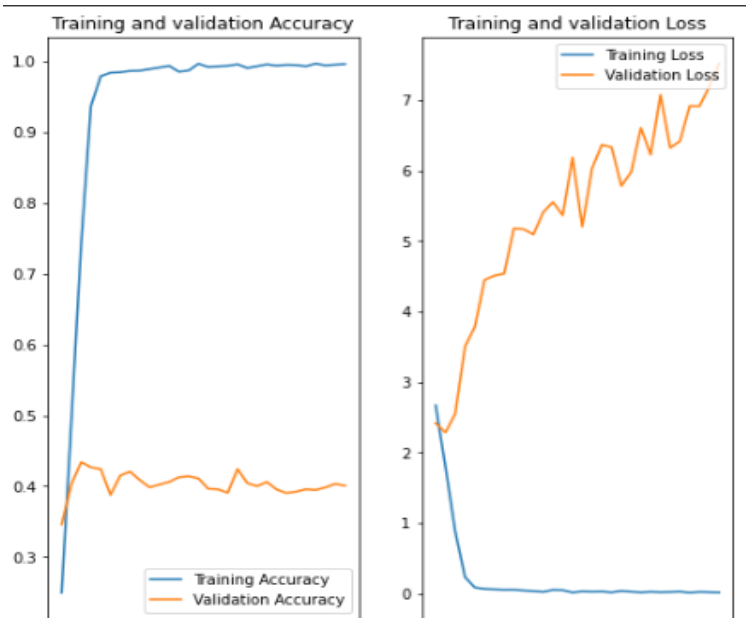
```
adam = tf.keras.optimizers.Adam(learning_rate=0.0005)
model.compile(optimizer=adam , loss='categorical_crossentropy',
metrics=['accuracy'])
```

Η μέθοδος "fit" εκπαιδεύει το μοντέλο στα δεδομένα εκπαίδευσης σε παρτίδες και επικυρώνει το μοντέλο στα δεδομένα επικύρωσης μετά από κάθε εποχή. Συγκεκριμένα το εκπαιδεύσαμε για 30 εποχές.

```
history = model.fit(train_set,
                    steps_per_epoch=800*30/32,
                    validation_data=val_set,
                    validation_steps=100*30/32,
                    batch_size=32, epochs= 30)

#storing lists in variables
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']

loss = history.history['loss']
val_loss = history.history['val_loss']

print(acc)
print(val_acc)
```

Δημιουργία και εμφάνιση γραφήματος.

```
epochs_range = range(30) # creating a sequence of number from 0 to 20

plt.figure(figsize=(8,8))

plt.subplot(1,2,1)
plt.plot(epochs_range, acc, label = "Training Accuracy")
plt.plot(epochs_range, val_acc, label = "Validation Accuracy")
plt.legend(loc='lower right')
plt.title('Training and validation Accuracy')

plt.subplot(1,2,2)
plt.plot(epochs_range, loss ,label = "Training Loss")
plt.plot(epochs_range, val_loss, label = "Validation Loss")
plt.legend(loc='upper right')
plt.title('Training and validation Loss')

plt.show()

#save the model
model.save('/content/drive/MyDrive/ColabNotebooks/cnnV1.h5')
```



Εικόνα 11: Γράφημα Μοντέλου 1.

Αποθηκεύουμε το μοντέλο στο google drive.

```
model.save('/content/drive/MyDrive/ColabNotebooks/cnnV1.h5')
```

Αποθηκεύουμε σε μεταβλητή το test path.

```
TEST = "/content/FurAnimals/Test"
```

Φορτώνουμε το μοντέλο.

```
#load the saved model
model =
tf.keras.models.load_model('/content/drive/MyDrive/ColabNotebooks/cnnV1
.h5')
```

Δημιουργούμε το τελικό test dataset με τους κατάλληλους παραμέτρους.

```
test_datagen = ImageDataGenerator(rescale=1./255)
test_set = test_datagen.flow_from_directory(
        TEST,
        target_size=(224, 224),
        batch_size=32,
        class_mode='categorical')
```

Υπολογισμός ευστοχίας σε άγνωστες εικόνες (test dataset)

```
accu = model.evaluate(test_set, steps=100*30/32)
print("Accuracy: ", accu[1])
```

93/93 [==============================] - 34s 280ms/step - loss: 6.3249 - accuracy: 0.4353
Accuracy: 0.43533334136009216

Παρατηρήσεις μοντέλου 1:

Σύμφωνα με τα αποτελέσματα παρατηρούμε όχι έχουμε πολύ χαμηλό ποσοστό ευστοχίας (0.4353). Επιπλέον στο γράφημα υπάρχει το φαινόμενο της υπερπροσαρμογής απο εκεί και μόνο ξέραμε απο πριν πως το μοντέλο δεν θα αποδώσει καθώς πρέπει. Γι' αυτό προσπαθήσαμε στο μοντέλο 2 να βελτιώσουμε το ποσοστό ακρίβειας κρατώντας ως βάση το μοντέλο 1 με μικροαλλαγές.

Μοντέλο 2:

Το μοντέλο 2 βασίστηκε στο μοντέλο 1 με μικροαλλαγές όμως όπως προσθήκη επιπέδων, δεδομένων (data augmentation), dropout και batch normalization επίπεδα.

Προσθήκη επιπέδων και δεδομένων (data augmentation).

```
# data augmentation and preparation

train_data = ImageDataGenerator(
    rescale = 1. / 255, #convert the pixels in range [0,255] to range
[0,1]
    shear_range = 0.2, #shear the image by 20%
    zoom_range = 0.2, #zoom the image by 20%
    rotation_range=25, #the images will be randomly rotated up to 25
degrees in either direction.
    horizontal_flip = True) #flips both rows and columns of the image
horizontally

train_set = train_data.flow_from_directory(TRAIN, target_size=(224,
224), batch_size=32, class_mode='categorical')

#convert the pixels in range [0,255] to range [0,1]
val_data = ImageDataGenerator(rescale = 1./255)
val_set = val_data.flow_from_directory(VAL, target_size=(224, 224),
batch_size=32, class_mode='categorical')
```

Προσθήκη dropout και batch normalization επιπέδων.

```
### build the model ###

#used to create models layer-by-layer
model = tf.keras.models.Sequential()
#This layer creates a convolution kernel that is convolved with the
layer input to produce a tensor of outputs.
model.add(tf.keras.layers.Conv2D(filters=32, kernel_size=(3, 3),
padding='Same', activation='relu', input_shape=[224, 224, 3]))
```

```
# maxpool2d calculates the maximum, or largest, value in each patch of
each feature map
#The Dropout layer randomly sets input units to 0 with a frequency of
rate (0.4 here) at each step during training time, which helps prevent
overfitting.
#Batch Normalization is a technique used to normalize the inputs of a
neural network layer by adjusting and scaling the activations
model.add(tf.keras.layers.MaxPool2D(pool_size=(2, 2)))
model.add(tf.keras.layers.Dropout(0.4))
model.add(tf.keras.layers.BatchNormalization())
model.add(tf.keras.layers.Conv2D(filters=64, kernel_size=(3, 3),
padding='Same', activation='relu'))
model.add(tf.keras.layers.MaxPool2D(pool_size=(2, 2)))
model.add(tf.keras.layers.Dropout(0.4))
model.add(tf.keras.layers.BatchNormalization())

model.add(tf.keras.layers.Conv2D(filters=128, kernel_size=(3, 3),
padding='Same', activation='relu'))
model.add(tf.keras.layers.MaxPool2D(pool_size=(2, 2)))
model.add(tf.keras.layers.Dropout(0.4))
model.add(tf.keras.layers.BatchNormalization())

model.add(tf.keras.layers.Conv2D(filters=128, kernel_size=(3, 3),
padding='Same', activation='relu'))
model.add(tf.keras.layers.MaxPool2D(pool_size=(2, 2)))
model.add(tf.keras.layers.Dropout(0.4))
model.add(tf.keras.layers.BatchNormalization())

model.add(tf.keras.layers.Conv2D(filters=128, kernel_size=(3, 3),
padding='Same', activation='relu'))
model.add(tf.keras.layers.MaxPool2D(pool_size=(2, 2)))
model.add(tf.keras.layers.Dropout(0.4))
model.add(tf.keras.layers.BatchNormalization())
```

Όμοια στο συνδετικό επίπεδο.

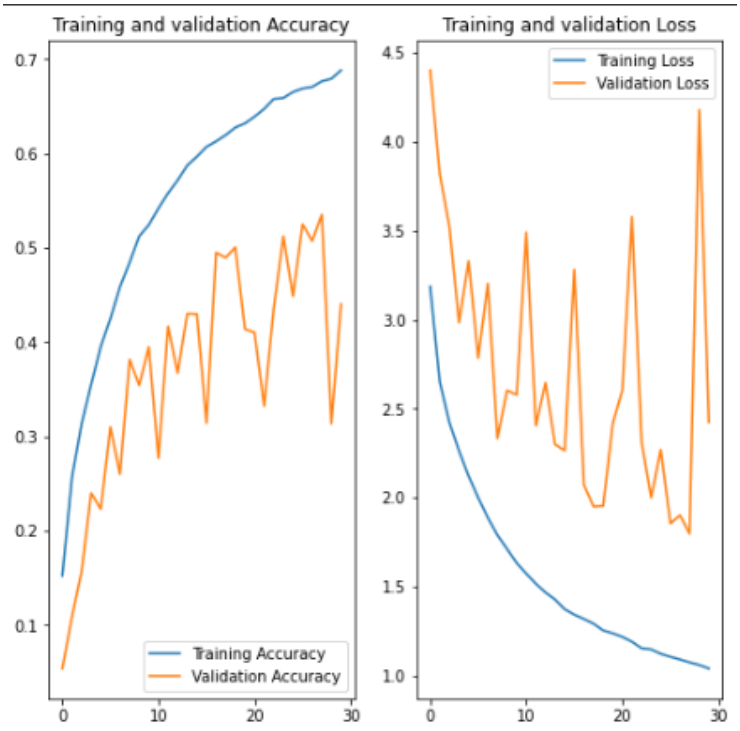
```
#Flatten before dense layer
# converting multi-dimensional array into one dimensional flatten array
or say single dimensional array
model.add(tf.keras.layers.Flatten())
# Dense is the regular deeply connected neural network layer
model.add(tf.keras.layers.Dense(units=512, activation='relu'))
model.add(tf.keras.layers.Dropout(0.2))
model.add(tf.keras.layers.BatchNormalization())
#the last layer
#units=30 animal categories
model.add(tf.keras.layers.Dense(units=30, activation='softmax'))
```

```
print(model.summary())
```

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 224, 224, 32)      896

 max_pooling2d (MaxPooling2D  (None, 112, 112, 32)     0
 )

 dropout (Dropout)           (None, 112, 112, 32)      0

 batch_normalization (BatchN  (None, 112, 112, 32)     128
 ormalization)

 conv2d_1 (Conv2D)           (None, 112, 112, 64)      18496

 max_pooling2d_1 (MaxPooling  (None, 56, 56, 64)       0
 2D)

 dropout_1 (Dropout)         (None, 56, 56, 64)        0

 batch_normalization_1 (Batc  (None, 56, 56, 64)       256
 hNormalization)

 conv2d_2 (Conv2D)           (None, 56, 56, 128)       73856
...
Trainable params: 3,617,566
Non-trainable params: 1,984
_________________________________________________________________
None
```

Εικόνα 12: Πίνακας μοντέλου 2.

Τα υπόλοιπα σημεία του κώδικα παραμένουν ίδια με το μοντέλο 1.



Εικόνα 13: Γράφημα μοντέλου 2.

Υπολογισμός ευστοχίας σε άγνωστες εικόνες (test dataset)

```
accu = model.evaluate(test_set, steps=100*30/32)
print("Accuracy: ", accu[1])
```

93/93 [==============================] - 24s 250ms/step - loss: 1.9591 - accuracy: 0.5043
Accuracy: 0.5043333172798157

Παρατηρήσεις μοντέλου 2:

Αρχικά με την προσθήκη των επιπλέον επιπέδων παρατηρούμε απο τον πίνακα το μοντέλου πως μειώθηκαν οι παράμετροι δηλαδή το δίκτυο μας έγινε πολύ πιο απλοϊκό. Από το γράφημα παρατηρούμε εκρηκτικές κλίσεις ενώ η υπερπροσαρμογή συνεχίζει να υπάρχει αλλά σε μικρότερο βαθμό. Τα dropout και batch Normalization επίπεδα μείωσαν τους χρόνους εκπαίδευσης και όπως βλέπουμε αύξησαν το ποσοστό ακρίβειας (0.5043) κατά 0.07 σε σχέση με το μοντέλο 1. Ακόμα και με όλα τα παραπάνω όμως το ποσοστό ακρίβειας παραμένει αρκετά χαμηλό.

5.2 Δοκιμή μοντέλου με εικόνες (test set).

Σε αυτό το σημείο θα τρέξουμε το μοντέλο για διάφορες εικόνες.

Φορτώνουμε το μοντέλο.

```
#load the saved model
model =
tf.keras.models.load_model('/content/drive/MyDrive/ColabNotebooks/cnnV3
.h5')
```

Διαβάζουμε και εμφανίζουμε την εικόνα που μας ενδιαφέρει.

```
import matplotlib.image as mpimg
img =
mpimg.imread('/content/FurAnimals/Test/11.Squirel/06dd2634cc57924e.jpg'
)
imgplot = plt.imshow(img)
```

Μετατρέπουμε την εικόνες σε τιμές απο 0 έως 1 πίξελ και προσθέτουμε μια ακόμα διάσταση.

```
from keras.preprocessing import image
img =
tf.keras.utils.load_img('/content/FurAnimals/Test/11.Squirel/06dd2634cc
57924e.jpg', target_size=(224, 224))
img = tf.keras.utils.img_to_array(img) / 255.
img = np.expand_dims(img, axis=0)
```

Προβλέπουμε τις πιθανότητες που μπορεί να ανήκει η εικόνα για κάθε κλάση.

```
probs = model.predict(img)
```

Παίρνουμε την κλάση με την μεγαλύτερη πιθανότητα να ανήκει σε εκείνη την κατηγορία η εικόνα.

```
class_index = np.argmax(probs[0])
```

Εμφανίζουμε τα αποτελέσματα και την κλάση με την μεγαλύτερη πιθανότητα.

```
print("Class probabilities: ", probs)
print("Predicted class: ", class_index)
```

```
Class probabilities:  [[1.1161291e-06 4.9466002e-03 9.1536506e-04 6.7903497e-04
1.1629794e-04 4.5616740e-05 2.3795601e-05 8.2000061e-06 8.4419473e-05 8.8003319e-05
 3.5252035e-04 2.2601785e-09 3.7121887e-03 1.9071533e-03 1.8071734e-04
 3.9358361e-04 4.0328082e-06 1.5141042e-04 6.3155005e-03 1.3203512e-05
 6.0739454e-05 6.4532841e-03 7.8730338e-08 2.4613782e-05 1.6862667e-07
 1.1782815e-06 2.5595548e-09 4.9641628e-08 9.6513379e-01 8.3874026e-03]]
Predicted class:  28
```

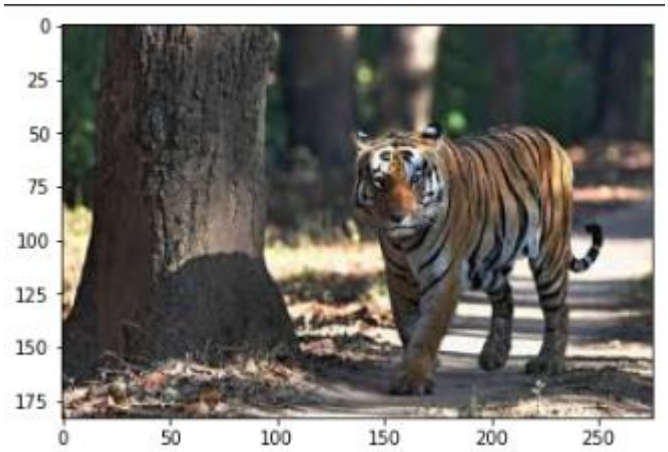
Αντιστοιχούμε τον δείκτη της κλάσης με την μεγαλύτερη πιθανότητα με το ζώο που ανήκει.

```
if class_index == 0:
    print('The animal is a tiger')
elif class_index== 1:
    print('The animal is a bear')
if class_index == 2:
    print('The animal is a squirel')
elif class_index == 3:
    print('The animal is a rabbit')
if class_index == 4:
    print('The animal is a kangaroo')
elif class_index == 5:
    print('The animal is a giraffe')
if class_index == 6:
    print('The animal is a deer')
elif class_index== 7:
    print('The animal is a goat')
```

```
elif class_index == 8:
    print('The animal is a cheetah')
if class_index == 9:
    print('The animal is a koala')
elif class_index == 10:
    print('The animal is a procupine')
elif class_index == 11:
    print('The animal is a zebra')
if class_index == 12:
    print('The animal is a fox')
elif class_index == 13:
    print('The animal is a mink')
if class_index == 14:
    print('The animal is a raccoon')
elif class_index == 15:
    print('The animal is a beaver')
if class_index == 16:
    print('The animal is a skunk')
elif class_index== 17:
    print('The animal is a leopard')
elif class_index == 18:
    print('The animal is a camel')
if class_index == 19:
    print('The animal is a meerkat')
if class_index == 20:
    print('The animal is a moose')
elif class_index == 21:
    print('The animal is a hyena')
if class_index == 22:
    print('The animal is a panda')
elif class_index == 23:
    print('The animal is a hamster')
if class_index == 24:
    print('The animal is a monkey')
elif class_index == 25:
    print('The animal is a gorilla')
if class_index == 26:
    print('The animal is a dog')
elif class_index == 27:
    print('The animal is a cat')
elif class_index == 28:
    print('The animal is a lion')
if class_index == 29:
    print('The animal is a wolf')
```

```
The animal is a lion
```

Στην πρώτη δοκιμή παρατηρούμε λοιπόν πως ενώ του δώσαμε ένα σκίουρο το μοντέλο προέβλεψε πως είναι λιοντάρι. Θα συνεχίσουμε με μερικές ακόμα εικόνες.



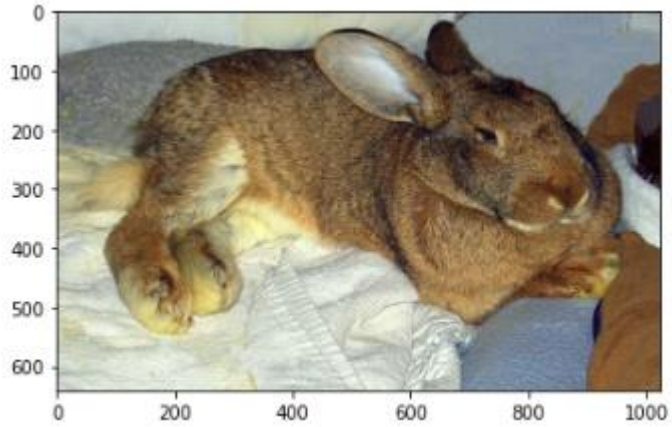

```
Class probabilities:  [[6.25178277e-01 4.70935716e-04 6.52453818e-05 6.03890912e-05
  4.20263736e-03 8.14344792e-04 9.58754739e-04 3.18748903e-06
  4.63298848e-03 1.05216238e-03 2.15181262e-05 9.15920362e-04
  6.72193361e-04 5.01571194e-05 6.96257746e-04 3.76582444e-02
  1.14082148e-04 1.41560987e-01 2.09675962e-03 4.75681736e-05
  6.90743909e-05 5.82130346e-03 2.86602764e-04 1.34690645e-05
  1.94378208e-05 7.34459754e-05 5.57594540e-05 1.44531323e-05
  1.69587880e-01 2.78605008e-03]]
Predicted class:  0
```

```
The animal is a tiger
```

Στη συγκεκριμένη περίπτωση το μοντέλο προβλέπει σωστά το ζώο. Σημαντικό ρόλο παίζει και το γεγονός πως τα χαρακτηριστικά του συγκεκριμένου ζώου φαίνονται με ευκρίνεια στην εικόνα.

```
Class probabilities:  [[1.6909665e-04 8.3372584e-03 7.5564283e-05 1.0315051e-03
8.3173206e-03

 7.5795819e-05 8.3784908e-03 8.4178755e-03 1.1139516e-03 6.9462350e-03
 1.3899287e-02 7.1284873e-04 2.2046673e-03 2.5978583e-01 1.6028235e-03
 4.2368406e-03 5.8295384e-02 5.3158001e-05 1.6618519e-01 3.7649545e-04
 7.9459578e-02 3.1139639e-01 1.0473223e-03 2.7610220e-03 9.1126678e-04
 8.7054906e-04 4.9999723e-04 2.9858376e-05 5.0649259e-02 2.1590367e-03]]
Predicted class:  21


The animal is a hyena
```

Στη συγκεκριμένη περίπτωση το μοντέλο προβλέπει λάθος.




```
Class probabilities:  [[5.16922283e-10 5.95671497e-03 6.43904670e-04 1.49090670e-03

 1.06407423e-02 1.35129972e-04 3.71037777e-06 3.30391886e-05
 6.33037018e-07 2.34733125e-05 4.40662575e-07 2.06586246e-12
 1.29264605e-04 1.85276406e-06 1.17581328e-02 3.72346170e-04
 1.94038530e-05 3.49878101e-05 1.21763663e-03 6.86590829e-06
 2.34233880e-06 7.07484316e-04 9.90564575e-11 4.38288544e-06
 1.29336115e-08 1.13622789e-09 2.19273755e-08 5.97790617e-09
 2.77541187e-02 9.39062417e-01]]
Predicted class:  29

The animal is a wolf
```

Στη συγκεκριμένη περίπτωση το μοντέλο προβλέπει λάθος.

Όπως παρατηρούμε απο τα αποτελέσματα στα τεστ το μοντέλο δεν αποδίδει αποτελεσματικά και κάνει εσφαλμένες προβλέψεις. Όπως προαναφέρθηκε το ποσοστό ευστοχίας ήταν πολύ χαμηλό (0.5043333172798157) οπότε περιμέναμε και τα ανάλογα αποτελέσματα.

Κεφάλαιο 6 – Μάθηση Μεταφοράς.

Η πλειονότητα των μοντέλων μηχανικής μάθησης και τεχνητής νοημοσύνης λειτουργούν με την παραδοχή ότι τα δεδομένα δοκιμής και εκπαίδευσης προέρχονται από τον ίδιο χώρο χαρακτηριστικών ή και κατανομή. Εάν γίνουν οποιεσδήποτε αλλαγές στο χώρο κατανομής, θα χρειαστεί να δημιουργηθεί ένα ολοκαίνουργιο στατιστικό μοντέλο από τη βάση, εκπαιδεύοντάς το με τα νεοαποκτηθέντα δεδομένα. Αυτή η διαδικασία είναι χρονοβόρα και σίγουρα δαπανηρή στις περισσότερες εφαρμογές του πραγματικού κόσμου. Είναι επίσης δύσκολο και σε ορισμένες περιπτώσεις σχεδόν αδύνατο να συλλεχθούν νέα και σχετικά δεδομένα εκπαίδευσης που απαιτούνται για την αναδόμηση του μοντέλου. Κατά τη διάρκεια τέτοιων περιόδων, αν η ανάγκη για την εκ νέου συλλογή των δεδομένων εκπαίδευσης μειωθεί, τότε αυτό θα ήταν μια τεράστια ανακούφιση. Αυτό μπορεί να επιτευχθεί με τη βοήθεια της μάθησης μεταφοράς. Η μάθηση μεταφοράς είναι ουσιαστικά η μεταφορά και αξιοποίηση της γνώσης που αποκτήθηκε από ένα έργο και η αξιοποίησή της για την επίλυση άλλων συναφών έργων.

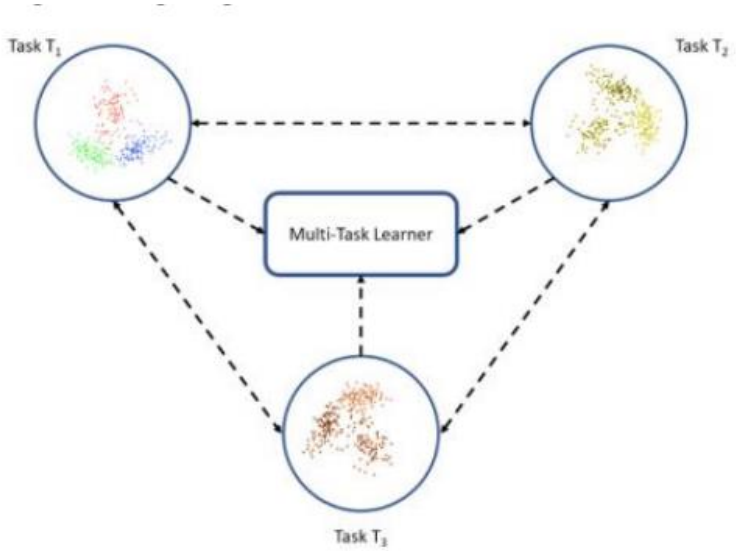
6.1 Τύποι μάθησης μεταφοράς

Η βαθιά μάθηση μεταφοράς, είναι μια γενική έννοια όπου οι πληροφορίες/γνώση από την υπάρχουσα εργασία προέλευσης χρησιμοποιούνται για να βοηθήσουν την εργασία-στόχο, στον ίδιο ή σε διαφορετικό τομέα. Ως εκ τούτου, οι παραλλαγές της περιλαμβάνουν, μεταξύ άλλων, τα εξής:

1. Προσαρμογή τομέα: Μια γνωστή προσέγγιση της Μάθησης Μεταφοράς είναι η προσαρμογή τομέα, η οποία βασίζεται στο σενάριο ότι οι τομείς της εργασίας πηγής και του στόχου κατέχουν διαφορετικές αλλά συναφείς κατανομές. Ένα τέτοιο παράδειγμα είναι το πρόβλημα του φιλτραρίσματος ανεπιθύμητης αλληλογραφίας, όπου ένα μοντέλο από έναν χρήστη εφαρμόζεται σε έναν νέο χρήστη, ο οποίος λαμβάνει ένα διαφορετικό σύνολο ηλεκτρονικών μηνυμάτων.

2. Σύγχυση τομέων: Να αναφέρουμε ότι τα στρώματα σε ένα βαθύ νευρωνικό μοντέλο καταγράφουν διαφορετικό σύνολο χαρακτηριστικών, ορισμένα από τα οποία μπορούν να αναγνωριστούν ως αμετάβλητα. Τέτοια χαρακτηριστικά μπορούν να μεταφερθούν σε διάφορους τομείς, με την προσαρμογή του τομέα του στόχου ώστε να προκληθεί ομοιότητα με αυτόν της πηγής, έτσι ώστε να εφαρμοστεί το μοντέλο που μαθαίνεται, ας πούμε με προεπεξεργασία.

3. Μάθηση πολλαπλών καθηκόντων: Η μάθηση πολλαπλών καθηκόντων αποσκοπεί στη βελτίωση της γενίκευσης με την ταυτόχρονη εκμάθηση πολλαπλών καθηκόντων. Μπορεί να θεωρηθεί ως αντιγραφή των ανθρώπινων μαθησιακών ικανοτήτων, όπου η γνώση μεταφέρεται από μια εργασία σε μια άλλη, όπου οι εργασίες αυτές φαίνονται να σχετίζονται (Εικόνα 14).



Εικόνα 14: Μάθηση πολλαπλών καθηκόντων.

4. Εκμάθηση με μία κίνηση: Ευθυγραμμιζόμενη με τα σενάρια του πραγματικού κόσμου, όπου δεν είναι πάντοτε δυνατή η λήψη δεδομένων με ετικέτες, η μάθηση μίας κίνησης είναι ένας τύπος μάθησης μεταφοράς, όπου η έξοδος συνάγεται χρησιμοποιώντας ένα ή λίγα παραδείγματα εκπαίδευσης.

5. Μάθηση μηδενικού αποτελέσματος: Ακραία παραλλαγή της μάθησης μεταφοράς όπου χρησιμοποιούνται μη επισημασμένα δεδομένα για τη μάθηση μιας εργασίας, ανάλογη με τη μάθηση χωρίς επίβλεψη. Χρησιμοποιεί έξυπνες προσαρμογές κατά τη διάρκεια της εκπαίδευσης του μοντέλου, για την κατανόηση μοτίβων αθέατων δεδομένων. Μια τέτοια εφαρμογή είναι στην επεξεργασία φυσικής γλώσσας, όπου οι μεταφραστές κατασκευάζονται χρησιμοποιώντας δεδομένα χωρίς ετικέτες της γλώσσας-στόχου.

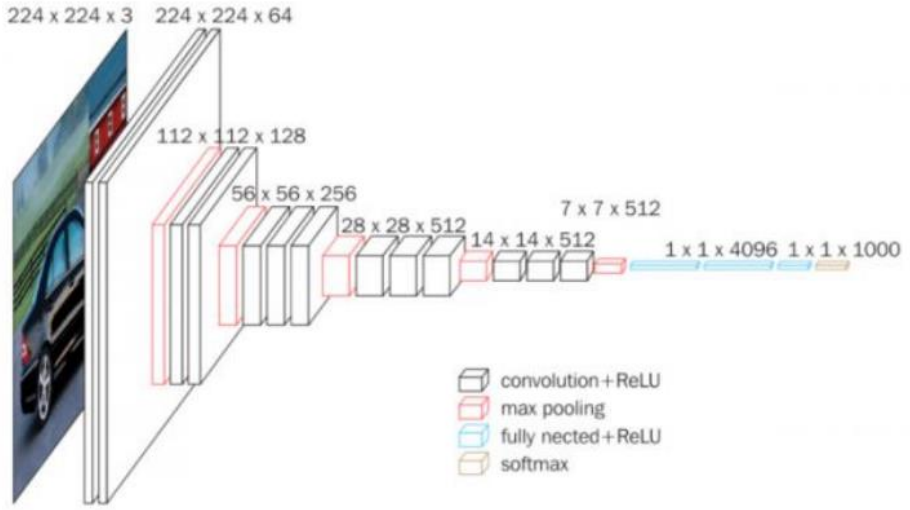
6.2 Κύρια μοντέλα μάθησης μεταφοράς.

Τα προ-εκπαιδευμένα μοντέλα μοιράζονται συνήθως με τη μορφή του εκατομμυρίων παραμέτρων/βαρών που πέτυχε το μοντέλο ενώ εκπαιδεύεται σε μια σταθερή κατάσταση. Υπάρχουν αρκετά από αυτά διαθέσιμα για εφαρμογές στην όραση υπολογιστών, με τα 3 πιο διαδεδομένα να είναι:

1.VGG16

Η ομάδα οπτικής γεωμετρίας, συντομογραφία VGG, που προέκυψε από την ομάδα ερευνητών που την ανέπτυξε και το 16 υποδηλώνει ότι η αρχιτεκτονική της έχει 16 στρώματα βάρους. Είναι ένα βαθύ νευρωνικό δίκτυο συνέλιξης που πέτυχε ακρίβεια 92,7% σε βάση δεδομένων ImageNet με 14 εκατ. εικόνες που ανήκουν σε 1000 κλάσεις.

Αρχιτεκτονική: RGB εικόνας σταθερού μεγέθους 224x224 τροφοδοτείται στη στοίβα 16 στρωμάτων conv. στρωμάτων, όπου χρησιμοποιούνται 3x3 φίλτρα. Χρησιμοποιήθηκε επίσης ένα φίλτρο conv. 1x1 για την εισαγωγή μη γραμμικότητας των καναλιών εισόδου. Το βήμα και το γέμισμα είναι σε ένα εικονοστοιχείο ανά φίλτρο, εξασφαλίζοντας τη διατήρηση τη χωρικής ανάλυσης. Παράθυρα μέγιστης συγκέντρωσης 5-2x2 με βήμα 2 προστίθενται για χωρική συγκέντρωση. 3 πλήρως συνδεδεμένα στρώματα στο στοίβα είναι ως εξής: 4096 κανάλια στα δύο πρώτα, 1000 κανάλια για τις 1000 κλάσεις εικόνας στο τρίτο στρώμα εκτελούν ταξινόμηση και τέλος, η αρχιτεκτονική διαθέτει στρώμα soft-max στο τέλος. Το VGG16 είναι μια βελτίωση σε σχέση με το AlexNet, όπου τα μεγάλα φίλτρα μεγέθους πυρήνα αντικαθίστανται από πολλαπλά φίλτρα 3x3 ως στοίβα.

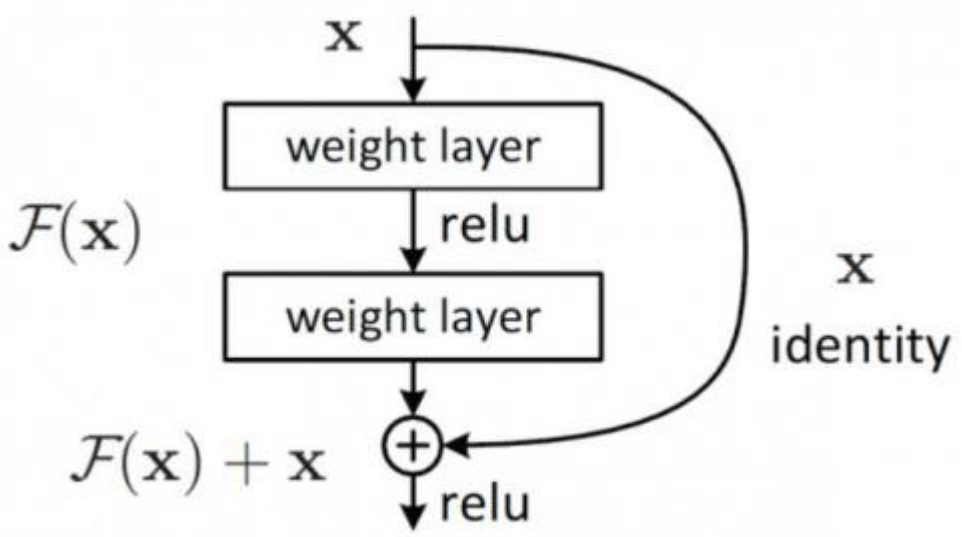


Εικόνα 15: Αρχιτεκτονική μοντέλου μάθησης VGG16.

Το μοντέλο είναι επεξηγήσιμο και επιτυγχάνει επαρκή ακρίβεια κατά τη δοκιμή σε δεδομένα εικόνας για κλασικές εργασίες ταξινόμησης, αλλά λόγω του βάθους του και του μεγάλου αριθμού παραμέτρων βάρους, το μοντέλο είναι μεγάλο σε μέγεθος και έχει υψηλό χρόνο εξαγωγής συμπερασμάτων. Μια από τις παραλλαγές του μοντέλου εισήχθη ως VGG19, όπου το βάθος του δικτύου ωθήθηκε από 16-19 στρώματα βάρους.

2. ResNet

Το μοντέλο αυτό εισάγει ένα πλαίσιο υπολειμματικής μάθησης για τη βελτίωση της εκπαίδευσης των βαθιών νευρωνικών δικτύων, εξ ου και η λέξη ResNet. Για να αναφερθεί η αποδοτικότητα στο βαθύ δίκτυο, το πρόβλημα της υποβάθμισης είναι στοχευμένο. Η ακρίβεια ενός βαθιού δικτύου αυξάνεται με το βάθος, θα κορεστεί και στη συνέχεια υποβαθμίζεται ραγδαία. Το ResNet χρησιμοποιεί υπολειμματική χαρτογράφηση, όπου αντί να επιτρέπει σε κάθε στρώμα να ταιριάζει κάτω απο μια επιθυμητή χαρτογράφηση, το στρώματα αναγκάζονται να προσαρμοστούν σε μια υπολειμματική χαρτογράφηση.



Εικόνα 16: Δομικό μπλοκ του μοντέλου μάθησης μεταφοράς ResNet.

Αρχιτεκτονική: Σε σύγκριση με το VGG, το υπολειμματικό νευρωνικό δίκτυο είναι λιγότερο πολύπλοκο με λιγότερα φίλτρα και μικρότερη ποσότητα εκπαίδευσης. Προστίθεται η σύνδεση συντόμευσης που μετατρέπει το δίκτυο στην υπολειμματική εκδοχή του, εκτελώντας την ταυτότητα χαρτογράφησης, χωρίς συμπλήρωση, επομένως χωρίς πρόσθετους παραμέτρους. Μαθηματικά, η συντόμευση προβολής είναι F(x{W}+x). Το ResNet είναι είτε δύο επίπεδα βάθους, όπως το ResNet18 και το 34, είτε τρία στρώματα βάθους, όπως τα ResNet50, 101 και 152.

3.Inception v3

Προκειμένου να αυξηθεί η αποδοτικότητα και η υπολογιστική ισχύς των δικτύων Inception (v1 κ.λπ.) και την ευκολότερη προσαρμογή του μοντέλου τεχνικές παραγοντοποιημένων συνελίξεων, κανονικοποίησης, μείωση των διαστάσεων και παραλληλισμένοι υπολογισμοί εφαρμόζονται στην v3.

Αρχιτεκτονική: Η αρχιτεκτονική του μοντέλου μπορεί να εξηγηθεί ως εξής:

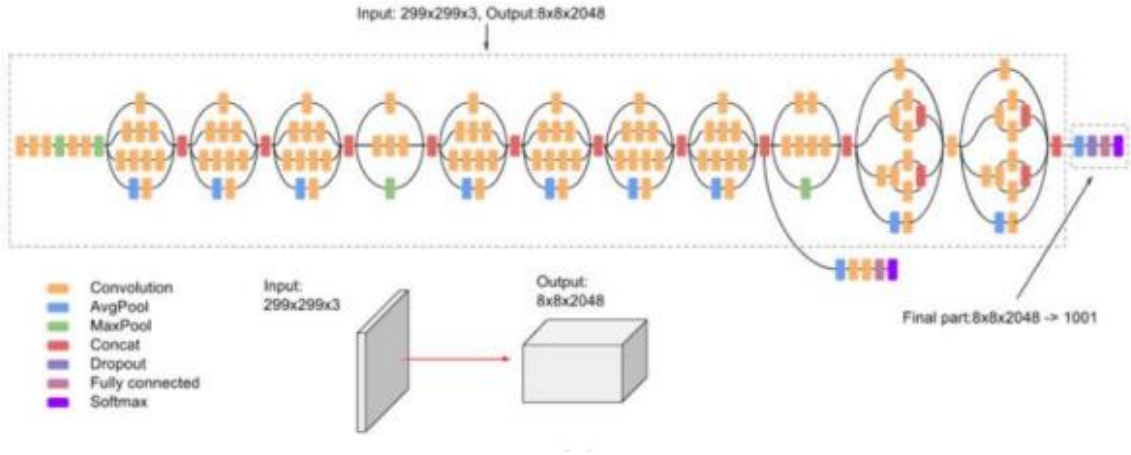
1. Factorized Convolutions - Διατηρεί έναν έλεγχο στο δίκτυο. την αποδοτικότητα και την υπολογιστική αποδοτικότητα μειώνοντας τον αριθμό των παραμέτρων.

2. Μικρότερες συνελίξεις/φίλτρα- Αυτό οδηγεί σε ταχύτερη εκπαίδευση καθώς μειώνεται ο αριθμός των παραμέτρων.

3. Ασυμμετρία- Απλά αντικαθιστώντας τα 3x3 με 1x3 και 3x1 συνέλιξη.

4. Βοηθητικός ταξινομητής - Ο βοηθητικός ταξινομητής λειτουργεί ως κανονικοποιητής, το CNN παρεμβάλλεται μεταξύ των στρωμάτων κατά τη διάρκεια εκπαίδευσης.

5. Μείωση του μεγέθους του πλέγματος- Στη συνέχεια, οι λειτουργίες συγκέντρωσης προστίθενται.



Εικόνα 17: Αρχιτεκτονική μοντέλου μεταφοράς μάθησης Inception v3.

Η μεταφορά γνώσης μπορεί να γίνει με δύο τρόπους: μέσω της εξαγωγής χαρακτηριστικών ή της τελειοποίησης ενός προ-εκπαιδευμένου μοντέλου. Τα προ-εκπαιδευμένα μοντέλα, όπως τα VGG, ResNet και Inception, έχουν χρησιμοποιηθεί ευρέως για τη μάθηση μεταφοράς σε προβλήματα προγνωστικής μοντελοποίησης που αφορούν δεδομένα εικόνας. Αυτά τα μοντέλα αναπτύχθηκαν από ερευνητές που χρησιμοποίησαν αποδοτικούς πόρους GPU και τα διέθεσαν στο διαδίκτυο. Η προσέγγιση αυτή είναι αποτελεσματική επειδή τα μοντέλα αυτά έχουν ήδη εκπαιδευτεί σε ένα μεγάλο σώμα δεδομένων εικόνας με σχετικά μεγάλο αριθμό κλάσεων. Ωστόσο, η μάθηση μεταφοράς δεν εγγυάται πάντα καλύτερη ακρίβεια σε μια συγκεκριμένη εργασία και η αξιολόγησή της μπορεί να γίνει μόνο μετά την ανάπτυξη. Η εκμάθηση μεταφοράς είναι ιδιαίτερα χρήσιμη όταν υπάρχουν περιορισμένα δεδομένα ή όταν το μοντέλο μπορεί να επαναχρησιμοποιηθεί εφικτά για μια συναφή εργασία που έχει πολλά δεδομένα, αποφεύγοντας έτσι την κατασκευή ενός νέου μοντέλου από το μηδέν. Τέλος, κατά την εφαρμογή των εννοιών της μάθησης μεταφοράς, είναι σημαντικό να εξεταστούν τρία ερωτήματα: τι να μεταφερθεί, πότε να μεταφερθεί και πώς να μεταφερθεί.

Κεφάλαιο 7 – Πειραματικό μέρος 2ο.

Το VGG16 είναι το κατάλληλο μοντέλο για την περίπτωση μας καθώς το σύνολο δεδομένων που έχουμε αποτελείται απο εικόνες και δεν είναι αρκετά μεγάλο σε μέγεθος. Παρακάτω λοιπόν αναλύουμε το συγκεκριμένο μοντέλο.

7.1 Δημιουργία CNN μοντέλου με χρήση μοντέλου μάθησης μεταφοράς VGG16.

Σύνδεση με google drive όπου ανεβάσαμε τα δεδομένα.

```
from google.colab import drive
drive.mount('/content/drive')
```

Αντιγραφή και unzip των δεδομένων μας σε φάκελο του google collabs.

```
cp '/content/drive/MyDrive/ColabNotebooks/FurAnimals811.zip'
'/content/FurAnimals811.zip'
!unzip '/content/FurAnimals811.zip' -d '/content/FurAnimals'
```

Εισαγωγή απαραίτητων βιβλιοθηκών.

```
# libaries
import numpy as np # library for working with arrays
import tensorflow as tf # library for deep learning and training deep
neural networks
import pandas as pdb
import seaborn as sn
import cv2
import os

from tensorflow import keras
from matplotlib import pyplot as plt # a comprehensive library for
creating static, animated, and interactive visualizations
from keras.preprocessing.image import ImageDataGenerator # library that
can generate batches of tensor image data with real-time data
augmentation.
from keras.regularizers import l2
from keras.layers import Activation, BatchNormalization, Conv2D, Dense,
Dropout, Flatten, MaxPooling2D
from keras.optimizers import Adam
from keras.losses import CategoricalCrossentropy
from keras.applications.vgg16 import VGG16
```

Προσθήκη επιπέδων και δεδομένων (data augmentation).

```
# data augmentation and preparation

train_data = ImageDataGenerator(
    rescale = 1. / 255, #convert the pixels in range [0,255] to range
[0,1]
    shear_range = 0.2, #shear the image by 20%
    zoom_range = 0.2, #zoom the image by 20%
    rotation_range=25, #the images will be randomly rotated up to 25
degrees in either direction.
    horizontal_flip = True) #flips both rows and columns of the image
horizontally

train_set = train_data.flow_from_directory(TRAIN, target_size=(224,
224), batch_size=32, class_mode='categorical')

#convert the pixels in range [0,255] to range [0,1]
val_data = ImageDataGenerator(rescale = 1./255)
val_set = val_data.flow_from_directory(VAL, target_size=(224, 224),
batch_size=32, class_mode='categorical')
```

Σύνδεση με μοντέλο VGG16.

```
base_model = VGG16(input_shape = (224, 224, 3), # Shape of our images
include_top = False, # Leave out the last fully connected layer
weights = 'imagenet')
```

Αποφυγή εκ νέου εκπαίδευσης των βαρών των επιπέδων του VGG16 καθώς είναι ήδη προεκπεδευμένα σε μεγάλο σύνολο δεδομένων.

```
for layer in base_model.layers:
    layer.trainable = False
```

Μετατροπή σε μονοδιάστατο πίνακα, εφαρμογή batch Normalization και dropout επιπέδων, δημιουργία τελικού συνδετικού επιπέδου.

```
#Flatten before dense layer

# converting multi-dimensional array into one dimensional flatten array
or say single dimensional array
y = base_model.output
y = Flatten()(y)
# Dense is the regular deeply connected neural network layer
y = tf.keras.layers.BatchNormalization()(y)
y = tf.keras.layers.Dense(units=512, activation='relu')(y)
y = tf.keras.layers.Dropout(0.5)(y)
```

Δημιουργία τελικού συνδετικού επιπέδου για τις 30 κατηγορίες ζώων που έχουμε.

```
y = tf.keras.layers.Dense(units=30, activation='softmax')(y)
model = tf.keras.models.Model(base_model.input, y)
print(model.summary())
```

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 224, 224, 3)]     0

 block1_conv1 (Conv2D)       (None, 224, 224, 64)      1792

 block1_conv2 (Conv2D)       (None, 224, 224, 64)      36928

 block1_pool (MaxPooling2D)  (None, 112, 112, 64)      0

 block2_conv1 (Conv2D)       (None, 112, 112, 128)     73856

 block2_conv2 (Conv2D)       (None, 112, 112, 128)     147584

 block2_pool (MaxPooling2D)  (None, 56, 56, 128)       0

 block3_conv1 (Conv2D)       (None, 56, 56, 256)       295168

 block3_conv2 (Conv2D)       (None, 56, 56, 256)       590080

 block3_conv3 (Conv2D)       (None, 56, 56, 256)       590080

 block3_pool (MaxPooling2D)  (None, 28, 28, 256)       0
...
Trainable params: 12,911,134
Non-trainable params: 14,764,864
_________________________________________________________________
None
```

Εικόνα 18: Πίνακας μοντέλου με χρήση VGG16.

Εφαρμογή Adam optimizer και compile του κώδικα.

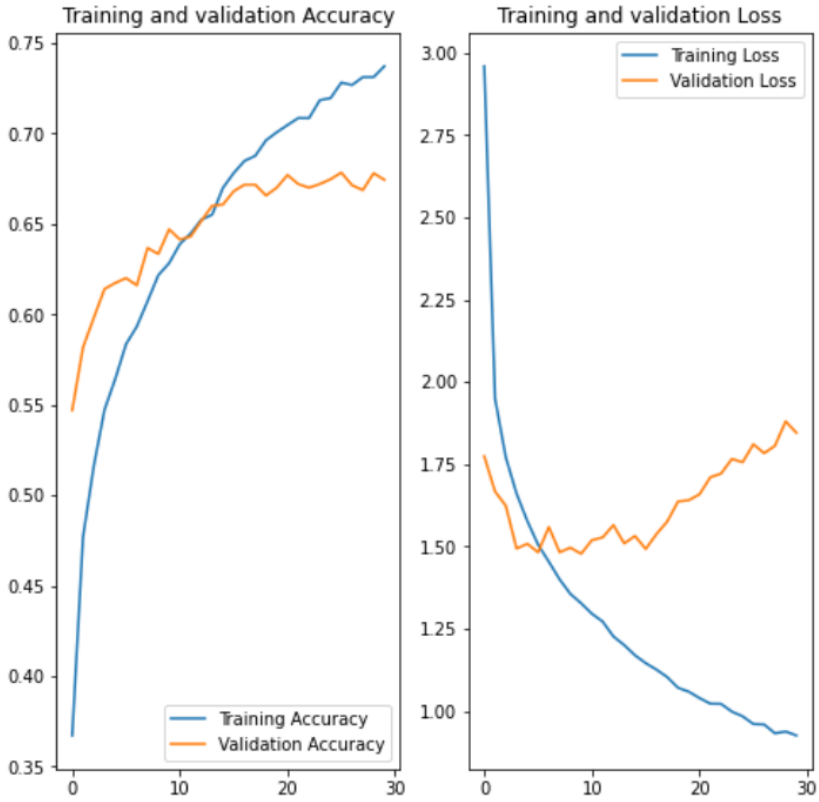
```
adam = tf.keras.optimizers.Adam(learning_rate=0.0005)
model.compile(optimizer=adam , loss='categorical_crossentropy',
metrics=['accuracy'])
```

Η μέθοδος "fit" εκπαιδεύει το μοντέλο στα δεδομένα εκπαίδευσης σε παρτίδες και επικυρώνει το μοντέλο στα δεδομένα επικύρωσης μετά από κάθε εποχή. Συγκεκριμένα το εκπαιδεύσαμε για 30 εποχές.

```
history = model.fit(train_set,
                    steps_per_epoch=800*30/32,
                    validation_data=val_set,
                    validation_steps=100*30/32,
                    batch_size=32, epochs= 30)

#storing lists in variables
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']

loss = history.history['loss']
val_loss = history.history['val_loss']

print(acc)
print(val_acc)
```

Δημιουργία και εμφάνιση γραφήματος, αποθήκευση μοντέλου.

```
#lets continue with displaying the result using pyplot

epochs_range = range(30) # creating a sequence of number from 0 to 30

plt.figure(figsize=(8,8))

plt.subplot(1,2,1)
plt.plot(epochs_range, acc, label = "Training Accuracy")
plt.plot(epochs_range, val_acc, label = "Validation Accuracy")
plt.legend(loc='lower right')
plt.title('Training and validation Accuracy')

plt.subplot(1,2,2)
plt.plot(epochs_range, loss ,label = "Training Loss")
plt.plot(epochs_range, val_loss, label = "Validation Loss")
plt.legend(loc='upper right')
plt.title('Training and validation Loss')

plt.show()

#save the model
model.save('/content/drive/MyDrive/ColabNotebooks/fa10.h5')
```



Εικόνα 19: Γράφημα μοντέλου με χρήση VGG16.

Αποθηκεύουμε σε μεταβλητή το test path.

```
TEST = "/content/FurAnimals/Test"
```

Φορτώνουμε το μοντέλο.

```
#load the saved model
model =
tf.keras.models.load_model('/content/drive/MyDrive/ColabNotebooks/fa10.
h5
```

Δημιουργούμε το τελικό test dataset με τους κατάλληλους παραμέτρους.

```
test_datagen = ImageDataGenerator(rescale=1./255)
test_set = test_datagen.flow_from_directory(
        TEST,
        target_size=(224, 224),
        batch_size=32,
        class_mode='categorical')
```

Υπολογισμός ευστοχίας σε άγνωστες εικόνες (test dataset)

```
accu = model.evaluate(test_set, steps=100*30/32)
print("Accuracy: ", accu[1])
```

93/93 [==============================] - 28s 293ms/step - loss: 1.4667 - accuracy: 0.7103
Accuracy: 0.7103333473205566

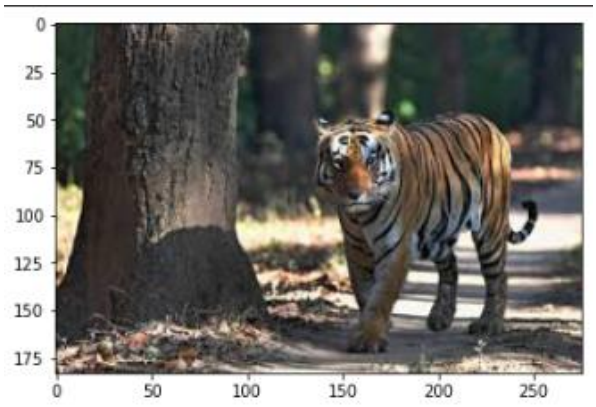
Παρατηρήσεις:

Αρχικά απο το γράφημα παρατηρούμε πως το μοντέλο σταθεροποιήθηκε σε μεγάλο βαθμό. Εξαφανίστηκαν οι εκρηκτικές κλίσεις με μικρή υπερπροσαρμογή που ξεκινάει γύρω στην 15-20 εποχή. Το ποσοστό ευστοχίας αυξήθηκε σημαντικά κατά 0.21 (από 0.50 σε 0.71). Μπορούμε να πούμε λοιπόν με σιγουριά πως η εφαρμογή μάθησης μεταφοράς με VGG16 είχε αξιόλογα και αισθητά αποτελέσματα στο αρχικό μας μοντέλο.

7.2 Δοκιμή VGG16 μοντέλου με εικόνες (test set).

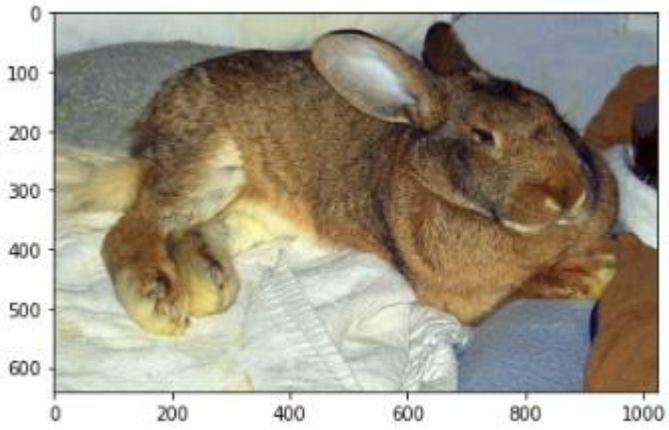
Σε αυτό το σημείο θα συγκρίνουμε τα αποτελέσματα του αρχικού μοντέλου με το VGG16 μοντέλο για τις ίδιες εικόνες. Θα χρησιμοποιήσουμε τον ίδιο κώδικα αλλάζοντας απλά το προορισμό αρχείου όπου θα βάλουμε αυτό του καινούργιου μοντέλου.


```
Class probabilities:  [[5.8230812e-06 7.3414767e-04 9.5903397e-01
1.8946318e-03 2.6749147e-03 1.8648738e-07 1.4806428e-04
1.0867599e-04 1.2182677e-04 9.0829300e-05 1.1156958e-03
3.8812673e-09 2.2212947e-02 8.4865437e-04 1.2001194e-03
5.5216464e-05 1.5851936e-03 1.6895399e-06 3.9857307e-05
2.0459194e-03 9.2732989e-05 2.3821424e-04 1.4000553e-06
3.1622807e-05 4.7528287e-03 5.2696258e-05 1.4216758e-04
2.2660639e-05 2.6482274e-04 4.8249864e-04]]

Predicted class:  2

The animal is a squirel
```




```
Class probabilities:  [[1.0000000e+00 3.1566046e-17 1.7547785e-17
1.8313856e-21 8.9786433e-19 1.5418930e-21 1.1967617e-14
1.8204417e-15 1.2268993e-16 1.3583354e-21 1.9886682e-16
4.1374335e-09 2.1941385e-15 4.4629693e-17 7.0240921e-20
4.5520745e-15 8.9056116e-19 5.1118422e-18 7.5196237e-17
7.3796364e-18 3.9849434e-14 8.4641198e-18 1.8511002e-22
1.3304887e-28 1.0437285e-18 2.0194515e-18 1.6165583e-18
1.4181649e-18 1.0007094e-12 2.9989974e-15]]

Predicted class:  0

The animal is a tiger
```


```
Class probabilities:  [[1.7692576e-03 3.8901162e-01 5.5755105e-02
1.8405234e-02 9.8613957e-03 2.1834304e-03 3.1231588e-03
3.5869043e-02 3.2434758e-02 4.3248909e-04 8.4239891e-04
1.1841920e-04 6.4263113e-02 1.2339752e-02 8.8234218e-03
1.4841373e-02 3.8557999e-02 6.2535185e-04 2.7669752e-02
8.0481526e-03 1.4009739e-01 4.8430782e-02 4.9308734e-04
2.8799535e-03 7.0837518e-04 8.4206881e-03 1.3014696e-02
8.0916501e-04 5.1803071e-02 8.3675655e-03]]

Predicted class:  1

The animal is a bear
```




```
Class probabilities:  [[1.5924286e-06 1.0142602e-03 1.0348315e-04
1.3160423e-03 1.3582525e-03 9.7697080e-07 8.5943109e-05
1.1928321e-02 9.0250516e-08 8.0296741e-06 1.0235061e-06
3.7048118e-09 8.8436072e-06 6.2514802e-05 1.5488293e-05
2.6053400e-04 3.6202004e-04 8.1492729e-08 9.5685291e-01
8.4660962e-05 1.7311996e-04 5.9752361e-05 7.8667955e-07
2.2663014e-02 3.0755888e-05 4.7634516e-05 1.1637831e-03
1.3049890e-05 2.3454749e-03 3.7664719e-05]]

Predicted class:  18

The animal is a camel
```

Το VGG16 μοντέλο αναγνώρισε 3/4 εικόνες ενώ το αρχικό 1/4 απο τις ίδιες εικόνες. Οπότε και στην πράξη το μοντέλο με χρήση VGG16 είναι αποτελεσματικό. Και στις δύο περιπτώσεις δεν αναγνώρισε την εικόνα του λαγού. Οι λόγοι είναι ότι τα χαρακτηριστικά δεν είναι πολύ εμφανή ενώ η γούνα του λαγού στην συγκεκριμένη εικόνα είναι όμοια με αυτή της καμήλας γι' αυτό και την αναγνωρίζει ως καμήλα.

Κεφάλαιο 8 – Συμπεράσματα

Τα συνεπαγωγικά νευρωνικά δίκτυα (CNN) έχουν μεταμορφώσει τον τομέα της όρασης υπολογιστών παρέχοντας κορυφαία αποτελέσματα στην ταξινόμηση εικόνων, την ανίχνευση αντικειμένων. Ωστόσο, η κατασκευή ενός CNN συνοδεύεται από το δικό της σύνολο προκλήσεων. Στην παρούσα διπλωματική συζητήθηκαν τα προβλήματα και οι λύσεις που σχετίζονται με την κατασκευή CNNs.

Μία από τις σημαντικότερες προκλήσεις στην κατασκευή CNN είναι η απόκτηση ενός μεγάλου και ποικίλου συνόλου δεδομένων. Τα επισημασμένα δεδομένα είναι απαραίτητα για την εκμάθηση και τη σωστή γενίκευση των CNN. Ωστόσο, η συλλογή υψηλής ποιότητας επισημασμένων δεδομένων μπορεί να είναι δαπανηρή και χρονοβόρα. Οι ερευνητές έχουν αναπτύξει διάφορες τεχνικές για την αντιμετώπιση αυτής της πρόκλησης, συμπεριλαμβανομένης της επαύξησης δεδομένων, της μάθησης μεταφοράς και της μάθησης με ημι-επίβλεψη.

Μια άλλη πρόκληση στην κατασκευή CNN είναι ο καθορισμός της βέλτιστης αρχιτεκτονικής και των υπερπαραμέτρων. Ένα CNN αποτελείται συνήθως από πολλαπλά επίπεδα, το καθένα με το δικό του σύνολο παραμέτρων, καθιστώντας τον προσδιορισμό της βέλτιστης αρχιτεκτονικής και υπερπαραμέτρων χρονοβόρα διαδικασία. Τεχνικές αντιμετώπισης είναι η αναζήτηση πλέγματος και η τυχαία αναζήτηση.

Η υπερπροσαρμογή, η οποία είναι συχνό φαινόμενο συμβαίνει όταν ένα CNN εκπαιδεύεται πολύ καλά στα δεδομένα εκπαίδευσης, με αποτέλεσμα κακή απόδοση στα δεδομένα δοκιμής. Αναπτύχθηκαν τεχνικές όπως η κανονικοποίηση, η εγκατάλειψη και η πρόωρη διακοπή για τον μετριασμό της υπερπροσαρμογής.

Τέλος έχουμε να κάνουμε και με την αντιμετώπιση της υπολογιστικής πολυπλοκότητας. Τα CNNs απαιτούν μεγάλες ποσότητες μνήμης και επεξεργαστικής ισχύος για την εκπαίδευση και την αξιολόγηση, γεγονός που αποτελεί πρόκληση για ερευνητές με περιορισμένους πόρους. Λύσεις στον συγκεκριμένο τομέα είναι τεχνικές όπως το κλάδεμα, η κβάντιση, η συμπίεση μοντέλων και χρήση διαδικτυακών πόρων.

Παρά τις προκλήσεις αυτές, τα CNN έχουν αποδείξει ότι μπορούν να προωθήσουν τον τομέα της όρασης υπολογιστών και να επιτρέψουν νέες εφαρμογές. Η μελλοντική έρευνα πιθανότατα θα συνεχίσει να διερευνά νέες λύσεις και εξελίξεις στα CNNs.

ΒΙΒΛΙΟΓΡΑΦΙΑ

Pearson Institute of Higher Education. Rosenblatt's Perceptron. Oct 1, 2008
URL: https://www.pearsonhighered.com/assets/samplechapter/0/1/3/1/0131471392.pdf

By Great Learning Team. What is Machine Learning? Defination, Types, Applications, and more. Feb 7, 2023 URL: What is Machine Learning? Definition, Types, Applications (mygreatlearning.com)

Soumya Joshi, Dhirendra Kumar Verma, Gaurav Saxena, Amit Paraye. Issues in Training a Convolutional Neural Network Model for Image Classification. July 2019.
URL:https://www.researchgate.net/publication/334541650_Issues_in_Training_a_Convolutional_Neural_Network_Model_for_Image_Classification#:~:text=Overfitting%2C%20exploding%20gradient%2C%20and%20class,the%20performance%20of%20the%20model.

Jason Brownlee. A Gentle Introduction to Exploding Gradients in Neural Networks. December 18 2017.
URL: https://machinelearningmastery.com/exploding-gradients-in-neural-networks/#:~:text=Exploding%20gradients%20are%20a%20problem,learn%20from%20your%20training%20data.

Sadaf Hossein Javaheri, Babak Teimourpour, in Data Mining Applications with R, 2014
URL: https://www.sciencedirect.com/topics/computer-science/class-imbalance-problem#:~:text=The%20class%20imbalance%20problem%20typically,and%20ignore%20the%20small%20ones.

IBM Technology corporation. What is machine learning?
URL:https://www.ibm.com/topics/machine-learning

Ashwin Joy. Pros and Cons of Deep Learning
URL: https://pythonistaplanet.com/pros-and-cons-of-deep-learning/

ASTON ZHANG, ZACHARY C. LIPTON, MU LI, AND ALEXANDER J. SMOLA. Dive into Deep Learning. July 15, 2019 URL: https://d2l.ai/d2l-en.pdf

C.C. Aggarwal. Neural Networks and Deep Learning: A Textbook. September 2018
URL:http://ndl.ethernet.edu.et/bitstream/123456789/88552/1/2018_Book_NeuralNetworksAndDeepLearning.pdf

Anirudha Ghosh, A.Sufian, Farhana Sultana, Amlan Chakrabarti, Debashis De. Fundamental Concepts of Convolutional Neural Network. January 2020.
URL:https://www.researchgate.net/publication/337401161_Fundamental_Concepts_of_Convolutional_Neural_Network

MK Gurucharan, 2020. Basic CNN Architecture: Explaining 5 Layers of Convolutional Neural Network. July 8 2022. URL: https://www.upgrad.com/blog/basic-cnn-architecture/

Sakshi Tiwari. Activation functions in Neural Networks. 17 Feb, 2023 URL: https://www.geeksforgeeks.org/activation-functions-neural-networks/

By Avijeet Biswal. The Complete Guide on Overfitting and Underfitting in Machine Learning. Feb 20, 2023. URL: https://www.simplilearn.com/tutorials/machine-learning-tutorial/overfitting-and-underfitting

Yash Bohra. The Challenge of Vanishing/Exploding Gradients in Deep Neural Networks. June 18, 2021. URL: https://www.analyticsvidhya.com/blog/2021/06/the-challenge-of-vanishing-exploding-gradients-in-deep-neural-networks/

Abhishek Kushwaha. Solving Class Imbalance problem in CNN. Jan 29, 2019. URL: https://medium.com/x8-the-ai-community/solving-class-imbalance-problem-in-cnn-9c7a5231c478

Jinwen. Tricks to prevent overfitting in CNN model trained on a small dataset. May 23, 2021. URL: https://jinwen17.medium.com/tricks-to-prevent-overfitting-in-cnn-model-trained-on-a-small-dataset-b84f05eb4eb7

Jason Brownlee. A Gentle Introduction to Exploding Gradients in Neural Networks. December 18, 2017. URL: https://machinelearningmastery.com/exploding-gradients-in-neural-networks/

Tina Jacob. Vanishing Gradient Problem, Explained. February 25, 2022. URL: https://www.kdnuggets.com/2022/02/vanishing-gradient-problem.html#:~:text=The%20vanishing%20gradient%20problem%20is,activation%20function%20such%20as%20ReLU.

Benai Kumar. 10 Techniques to deal with Imbalanced Classes in Machine Learning. July 23, 2020. URL: https://www.analyticsvidhya.com/blog/2020/07/10-techniques-to-deal-with-class-imbalance-in-machine-learning/

Ayush Gupta. A Comprehensive Guide on Deep Learning Optimizers. October 7, 2021. URL:https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-on-deep-learning-optimizers/

TARANG SHAH. Train, Validation and Test Sets. 03 DEC 2017. URL:http://tarangshah.com/blog/2017-12-03/train-validation-and-test-sets/

Chandeep Sharma. Transfer Learning and its application in Computer Vision: A Review. March 13, 2022. https://www.researchgate.net/publication/359199617_Transfer_Learning_and_its_application_in_Computer_Vision_A_Review