Αυτόματος χρωματισμός ασπρόμαυρου βίντεο με την χρήση ανταγωνιστικών δικτύων

Σάμιος Γεώργιος

Διπλωματική Εργασία

Επιβλέπων: κ. Χριστόφορος Νίκου

κ. Γιώργος Σφήκας

Ιωάννινα, Φεβρουάριος , 2022

ΤΜΗΜΑ ΜΗΧ. Η/Υ & ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

UNIVERSITY OF IOANNINA

Ευχαριστίες

Η παρούσα διπλωματική εργασία με τίτλο **"Αυτόματος χρωματισμός ασπρόμαυρου βίντεο με την χρήση ανταγωνιστικών δικτύων"** εκπονήθηκε στο τμήμα Μηχανικών Ηλεκτρονικών Υπολογιστών και Πληροφορικής του Πανεπιστημίου Ιωαννίνων.

Θα ήθελα να ευχαριστήσω τον επιβλέποντα Καθηγητή μου, Χριστόφορο Νίκου που δέχτηκε την πρότασή μου για το θέμα αυτής της διπλωματικής εργασίας και με βοήθησε να την ολοκληρώσω. Επίσης, θα ήθελα να ευχαριστήσω τον συνεπιβλέποντα καθηγητή μου, Γιώργο Σφήκα ο οποίος με καθοδήγησε σε όλη την διάρκεια εκπόνησης της εργασίας και μου μετέδωσε τις γνώσεις και τον ενθουσιασμό του για την ολοκλήρωση της παρούσας διπλωματικής εργασίας.

Περίληψη

Τα τελευταία χρόνια υπάρχει μια γιγαντιαία ανάπτυξη και εξέλιξη των Νευρωνικών δικτύων και της Υπολογιστικής όρασης που όλο ένα και περισσότερο μας βοηθούν να υλοποιήσουμε πράγματα που χρειάζονται κάποια εκπαίδευση και κάποια ανθρώπινη κρίση, όπως είναι ο χρωματισμός ασπρόμαυρων εικόνων, η ανάλυση ποιότητας εικόνας, η αυτόνομη οδήγηση που γίνεται πράξη από την Tesla. Συγκεκριμένα, η ανάγκη για τον χρωματισμό ασπρόμαυρων εικόνων προέκυψε από την ανάγκη των ανθρώπων είτε να παρακολουθήσουν κινηματογραφικές παραγωγές παλαιότερων εποχών με αληθινό χρώμα, είτε να χρωματιστούν παλιές φωτογραφίες που είχαν στην κατοχή τους.

Σκοπός της παρούσας διπλωματικής εργασίας είναι η ανάλυση μιας από τις αρχιτεκτονικές δικτύων που χρησιμοποιούνται για αυτόν τον σκοπό. Η αρχιτεκτονική αυτή εστιάζει στα Generative Adversarial Network (GAN) ή με άλλα λόγια στα δίκτυα που αποτελούνται από την γεννήτρια (Generator) και τον διακριτή (Discriminator). Η γεννήτρια χρησιμοποιείται για την παραγωγή ψεύτικων εικόνων και ο διακριτής για την διάκριση των ψεύτικων και των αληθινών απεικονίσεων. Για την διαδικασία της κατασκευής και επιλογής της τελικής αρχιτεκτονικής του δικτύου δημιουργήθηκε, εκπαιδεύτηκε και δοκιμάστηκε ένα σύνολο δικτύων σε ένα σύνολο δεδομένων.

Ο κύριος σκοπός ήταν να επιλεχθεί και να διορθωθεί η αρχιτεκτονική με την καλύτερη προσέγγιση στον χρωματισμό από τη μία και τη μικρότερη δυνατή απώλεια, από την άλλη. Η απόφαση για την τελική αρχιτεκτονική πάρθηκε αφού συγκρίθηκαν τα αποτελέσματα για το κάθε δίκτυο ξεχωριστά. Τα εν λόγω αποτελέσματα αναφέρονται σε φωτογραφίες, οι οποίες δεν χρησιμοποιήθηκαν κατά την εκπαίδευσή τους και ενώ τα δίκτυα είχαν ήδη εκπαιδευτεί.

Λέξεις Κλειδιά: Συνελικτικά νευρωνικά δίκτυα, υπολογιστική όραση, χρωματισμός ασπρόμαυρων εικόνων, γεννήτρια, διακριτής, σύνολο δεδομένων

Abstract

In recent years there has been a giant development and radical evolution of Neural Networks and Computational Vision that are helping us more and more to realize things that need training and human judgment or sagacity, such as coloring black and white images, image quality analysis, autonomous driving that was carried out by Tesla. In particular, the need to color black and white images arose from the need of people to either watch all time classic black and white movies in true color or to paint their old pictures.

The purpose of this dissertation is to analyze one of the network architectures used for this exact purpose. This architecture focuses on the Generative Adversarial Network (GAN) or in other words, on the networks consisting of the Generator and the Discriminator. The Generator is used to produce false images and the Discriminator to distinguish between false and true images. For the process of construction and selection of the final network architecture, a set of networks was created, trained, and tested in a data set.

The main purpose was to correct and select the architecture with the best approach on coloring, and the smallest possible loss. The decision for the final architecture was made after comparing the results for each network separately. Those results refer to images which were not used during their own training and while the networks had already been trained.

Keywords: Convolutional neural networks, computational vision, black and white image coloring, generator, discriminator, data set

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή

1.1 Αντικείμενο διπλωματικής

Η ιδέα για την δημιουργία ενός λογισμικού που θα χρωματίζει τις ασπρόμαυρες εικόνες, δεν είναι μία σύγχρονη ανάγκη, αλλά έχει τις ρίζες της αρκετά χρόνια πίσω, τότε που όλες οι ταινίες και οι φωτογραφίες ήταν ασπρόμαυρες και ο κόσμος ήθελε να βάλει περισσότερο χρώμα στην καθημερινότητα που απαθανάτιζε. Χάρις τη βοήθεια της προόδου και της τεχνολογικής ανάπτυξης ο άνθρωπος προχώρησε και βρέθηκε σταδιακά στο σήμερα, οπότε η έγχρωμη εικόνα θεωρείται δεδομένη.

Παράλληλα, η ανάπτυξη ενός λογισμικού για το χρωματισμό ασπρόμαυρων εικόνων συνέβαλε στην ποιοτική βελτίωση της απόδοσης των επαγγελματιών, όχι μόνο στον κλάδο του κινηματογράφου, την αποτελεσματικότητα και την επίδοσή τους. Επί τω πρακτικότερον, απέκτησαν τη δυνατότητα να διεκπεραιώνουν γρηγορότερα έργα που υπό άλλες συνθήκες (προηγούμενες δεκαετίες) χρειάζονταν ώρες ή και μέρες για να ολοκληρωθούν. Εκτός όμως από το ποιοτικότερο αποτέλεσμα είναι πλέον και αισθητά μικρότερη η κατανάλωση πόρων και ενέργειας που απαιτούνται.

Σπουδαίο ρόλο προς την κατεύθυνση για το σχεδιασμό τέτοιου είδους λογισμικών είχε η εμφάνιση της τεχνητής νοημοσύνης και η εδραίωσή της ευρέως στον επιχειρηματικό κι επιστημονικό κόσμο.

1.1 Παράδειγμα χρωματισμού ασπρόμαυρης εικόνας

Στην εν λόγω μελέτη αναλύονται ενδελεχώς οι ήδη υπάρχουσες τεχνολογίες στον τομέα του χρωματισμού εικόνας (βλέπε εικόνα 1.1) και η θεωρία της τεχνητής νοημοσύνης που βρίσκεται πίσω από αυτό το εγχείρημα. Εν τέλει, παρουσιάζεται η πρόκριση μιας πιο απλής εκδοχής από τις υφιστάμενες, η οποία χαρακτηρίζεται ως απλούστερη και αρκετά αποδοτική.

1.2 Στόχος της διπλωματικής

Στόχος της παρούσας διπλωματικής είναι αφενός η παρουσίαση των τεχνολογιών και αρχιτεκτονικών μοντέλων που αφορούν των χρωματισμό ασπρόμαυρων εικόνων και αφετέρου η χρήση του διαθέσιμου θεωρητικού υπόβαθρου για την δημιουργία εξαρχής ενός τέτοιου λογισμικού. Για την πραγματοποίηση του τελικού σκοπού, πρώτο και βασικό βήμα είναι φυσικά η απλοποίηση του όρου αυτού καθαυτού και του προβλήματος του επιχρωματισμού. Συνακολούθως, προκύπτει και η ανάγκη λήψης μιας απόφασης αναφορικά με τη διαδικασία που χρειάζεται να ακολουθεί και τελικά η ουσιαστική κατασκευή του προγράμματος που θα διασφαλίσει πως εισάγοντας μια ασπρόμαυρη φωτογραφία στο δίκτυο, εκείνο θα επιστρέφει μια ρεαλιστικά χρωματισμένη εικόνα.

Η εκπαίδευση αυτή του συστήματος στηρίχθηκε σε έναν κώδικας μέσω του οποίου συγκροτήθηκε μια τεράστια βάση δεδομένων. Οι χιλιάδες εικόνες που βρίσκονται στη βάση συλλέχθηκαν όλες από μία πληθώρα διαφόρων βίντεο που είναι διαθέσιμα, χωρίς περιορισμούς. Εν συνεχεία, οι εικόνες για τις οποίες γίνεται λόγος μετασχηματιστηκαν κατά τέτοιο τρόπο ώστε να έχουν πλέον τις κατάλληλες εκείνες διαστάσεις οι οποίες θα μπορούν να φορτωθούν στο πρόγραμμα. Στο ίδιο σύνολο εκπαιδεύτηκαν τρία (3) έτοιμα δίκτυα, που αντλήθηκαν από το διαδίκτυο και συγκρίθηκαν μεταξύ τους με βάση το σφάλμα και τα αποτελέσματά τους. Τα παραπάνω δίκτυα χρησιμοποιούνται έτοιμα, καθώς σκοπός της μελέτης αυτής δεν είναι σε καμία περίπτωση η δημιουργία μιας αρχιτεκτονικής για το νευρωνικό δίκτυο (εξαιτίας της δυσκολίας του εγχειρήματος).

Τα τελικά εκπαιδευμένα δίκτυα δοκιμάστηκαν με γνώμονα την ικανότητα γενίκευσής τους, δηλαδή με άλλα λόγια ως προς την αληθοφάνεια των τελικών εικόνων (στην περίπτωση αυτή). Αναφορικά με τις τεχνικές εκπαίδευσης πάνω στις οποίες στηρίχθηκε το αποτέλεσμα, αυτές είναι δύο: μια με RGB εικόνες και μια με LAB εικόνες. Ο λόγος που προέκυψε αυτό το δίλημμα παρουσιάζεται στο 3^{ο} κεφάλαιο αναλυτικά. Αν και εκπαιδεύτηκαν όλα τα μοντέλα από κοινού με LAB και με RGB εικόνες, εν τέλει η εκπαίδευση ολοκληρώθηκε με LAB εικόνες.

1.3 Οργάνωση τόμου

- Στο δεύτερο κεφάλαιο παρατίθενται οι ορισμοί της τεχνητής νοημοσύνης, των νευρωνικών δικτύων, αλλά και οι εφαρμογές της πρώτης στην ανθρώπινη καθημερινότητα. Επιπλέον, αναλύεται και ο τύπος δικτύου που χρησιμοποιήθηκε.

- Στο κεφάλαιο που έπεται (τρίτο) γίνεται αναφορά στην καθαυτή υλοποίηση του προγράμματος και τα βήματα γραφής του κώδικα με εικόνες και επεξηγήσεις, για καλύτερη κατανόηση. Ειδικότερα, εμφανίζεται ο τρόπος ανάπτυξης του

dataset και σχεδιασμού των δικτύων γεννήτριας (Generator) και διακριτή (Discriminator).

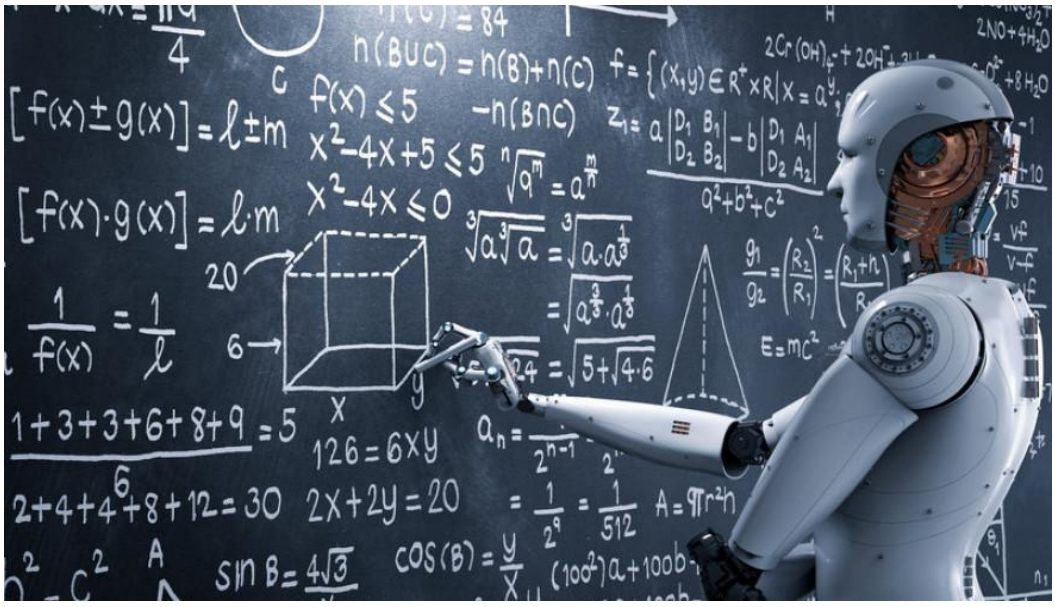
- Στο τελευταίο κεφάλαιο (τέταρτο) δίνεται μια σύνοψη της διπλωματικής εργασίας και του περιεχομένου που αναλύθηκε στα κεφάλαια 2-4. Παρουσιάζονται, επίσης, τα τελικά συμπεράσματα, κατά πόσο επιτεύχθηκαν οι στόχοι της εργασίας, τα προβλήματα που αναδύθηκαν, κάποιους από τους λόγους που το εγχείρημα δεν στέφθηκε από απόλυτη επιτυχία. Το τελικό στοιχείο που αναφέρεται στον αναγνώστη είναι ορισμένες σκέψεις για μελλοντικές επεκτάσεις που θα μπορούσαν να γίνουν προκειμένου να βελτιωθεί το αποτέλεσμα που προκύπτει.

Κεφάλαιο 2

Τεχνητή νοημοσύνη και νευρωνικά δίκτυα

2.1 Εισαγωγή στην τεχνητή νοημοσύνη

Στις αρχές της δεκαετίας του 1990 η ιδέα και η υλοποίηση της ιδέας για την τεχνητή νοημοσύνη των μηχανών αποτέλεσε τη θρυαλλίδα για τη μεγαλύτερη τεχνολογική επανάσταση του αιώνα, αλλάζοντας εκ βάθρων -φυσικά προς το καλύτερο- την ανθρώπινη κοινωνία. Οι αλλαγές που συνοδεύουν την υιοθέτηση της τεχνητής νοημοσύνης εκτείνονται πέρα από τους κλάδους της οικονομίας και της κοινωνίας. Τι είναι όμως πραγματικά η τεχνητή νοημοσύνη και πως πραγματικά αυτή θα συμβάλλει στη βελτίωση της ζωής μας;

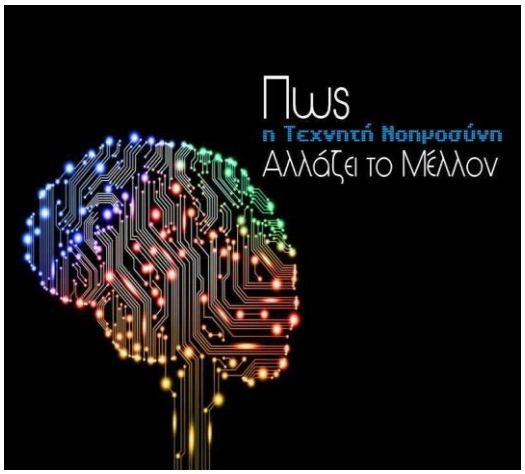


2.1 Εκπαιδευμένο ρομπότ

Ο όρος τεχνητή νοημοσύνη, όπως αυτός δόθηκε από την Ευρωπαϊκή Επιτροπή (Απρίλιος 2021) αναφέρεται στον κλάδο της πληροφορικής, ο οποίος ασχολείται με

την σχεδίαση και την υλοποίηση υπολογιστικών συστημάτων που μιμούνται στοιχεία της ανθρώπινης συμπεριφοράς (βλέπε εικόνα 2.1), τα οποία όμως έχουν την ελάχιστη στοιχειώδη ευφυΐα, μάθηση, προσαρμοστικότητα, εξαγωγή συμπερασμάτων, κατανόηση από συμφραζόμενα, δημιουργικότητα, επίλυση προβλημάτων. Αποτελεί σημείο τομής μεταξύ πολλαπλών επιστημών, εν παραδείγματι της πληροφορικής, της ψυχολογίας, της νευρολογίας, της γλωσσολογίας, της μηχανικής, με στόχο την σύνθεση ευφυούς συμπεριφοράς, με στοιχεία συλλογιστικής σκέψης, μάθησης, και προσαρμογής στο εξωτερικό, μεταβαλλόμενο περιβάλλον [2].

Διακρίνεται στη συμβολική τεχνητή νοημοσύνη, η οποία προσπαθεί να πλησιάσει την ανθρώπινη νοημοσύνη αλγοριθμικά (χρησιμοποιώντας δηλαδή σύμβολα και λογικούς κανόνες υψηλού επιπέδου), και την υποσυμβολική τεχνητή νοημοσύνη, η οποία επιδιώκει να αναπαράξει την ανθρώπινη ευφυΐα χρησιμοποιώντας στοιχειώδη αριθμητικά μοντέλα.



Με την ανάπτυξη της τεχνητής νοημοσύνης και των υπολογιστών δημιουργήθηκαν μηχανές που εκτελούν ενέργειες και λύνουν προβλήματα που έως πρότινος μόνο οι άνθρωποι μπορούσαν να λύσουν. Τα οφέλη της αφορούν τόσο τους ανθρώπους όσο και τον επιχειρηματικό κόσμο.

2.2 Τεχνητή νοημοσύνη

Ως προς την ανθρώπινη καθημερινότητα είναι δεδομένο πως μπορεί να αποφέρει καλύτερη υγειονομική περίθαλψη, ασφαλέστερες μεταφορές, φθηνότερες υπηρεσίες και προϊόντα μεγαλύτερης διάρκειας. Διευκολύνει, επίσης, την πρόσβαση στην ενημέρωση και την εκπαίδευση και συμβάλει σε ένα ασφαλέστερο εργασιακό περιβάλλον, χρησιμοποιώντας μηχανήματα-ρομπότ για την εκτέλεση επικίνδυνων εργασιών. Σε επιχειρηματικό επίπεδο, επιτρέπεται η ανάπτυξη νέων προϊόντων και

υπηρεσιών για την πράσινη οικονομία, η κατασκευή μηχανημάτων για τη γεωργία, την υγεία, τον τουρισμό κτλ [1].

2.2 Εφαρμογές της τεχνητής νοημοσύνης

Αφού αναπτύχθηκε προηγουμένως το θεωρητικό πλαίσιο της έννοιας σε αυτό το σημείο παρουσιάζονται απτά παραδείγματα αυτής στην καθημερινότητα [4]:

- **Υγεία**

  Σε αυτό το πεδίο έχουν εισέλθει τύποι ρομπότ που μπορούν έως και να εκτελέσουν εγχειρήσεις (βλέπε εικόνα 2.3), με θεαματικά αποτελέσματα για τους ασθενείς. Οι επιστήμονες φιλοδοξούν φυσικά να φτάσουν στο σημείο που εξολοκλήρου θα αναλαμβάνουν επεμβάσεις, μειώνοντας ταυτόχρονα τα ανθρώπινα ιατρικά λάθη. Ακόμα, είναι δυνατόν να χρησιμοποιηθούν στην ανάλυση ιατρικών δεδομένων οδηγώντας σε νέες επαναστατικές επιστημονικές ανακαλύψεις.

2.3 Τεχνητή νοημοσύνη στην ιατρική

- **Διαδικτυακές αγορές και διαφήμιση**

  Η τεχνητή νοημοσύνη χρησιμοποιείται ευρέως για την παροχή εξατομικευμένων συστάσεων, για παράδειγμα βάσει προηγούμενων αναζητήσεων και αγορών ή άλλων συμπεριφορών, προτείνει στους χρήστες παρόμοια ή συμπληρωματικά προϊόντα.

- **Διαδικτυακή αναζήτηση**

  Οι μηχανές αναζήτησης παρέχουν αποτελέσματα με τη βοήθεια της μεγάλης ποσότητας δεδομένων που εισάγουν οι χρήστες στο διαδίκτυο.

- **Προσωπικοί ψηφιακοί βοηθοί**

  Τα έξυπνα τηλέφωνα (smartphones) χρησιμοποιούν την τεχνητή νοημοσύνη για την παροχή βελτιστοποιημένων και εξατομικευμένων ρυθμίσεων στους χρήστες τους. Ο εικονικός βοηθός λειτουργεί ως προσωπικός γραμματέας του χρήστη: απαντά σε ερωτήσεις, παρέχει συστάσεις, υπενθυμίζει συναντήσεις. Είναι επίσης ένας ηλεκτρονικός συνομιλητής που προσαρμόζεται στα ατομικά χαρακτηριστικά ενός συγκεκριμένου ατόμου, λαμβάνοντας υπόψη το περιβάλλον του χρήστη, τα ενδιαφέροντα και τις συνήθειές του.

- **Αυτόματες μεταφράσεις**

  Τα λογισμικά αυτόματης μετάφρασης και υποτιτλισμού, που βασίζονται είτε σε γραπτό είτε σε προφορικό λόγο, χρησιμοποιούν την τεχνητή νοημοσύνη για την παροχή και βελτίωση των μεταφράσεων.

- **Κυβερνοασφάλεια**

Τα συστήματα τεχνητής νοημοσύνης μπορούν να συμβάλουν στην αναγνώριση και αντιμετώπιση επιθέσεων και απειλών στον κυβερνοχώρο χάρις τη συνεχόμενη εισροή δεδομένων.

- **Αυτοκίνητα**

  Παρότι τα αυτόνομα οχήματα δεν αποτελούν ακόμα μέρος της καθημερινότητάς μας, τα αυτοκίνητα απαρτίζονται ήδη από ευφυή συστήματα ασφαλείας που κάνουν χρήση τεχνητής νοημοσύνης (βλέπε εικόνα 2.4).

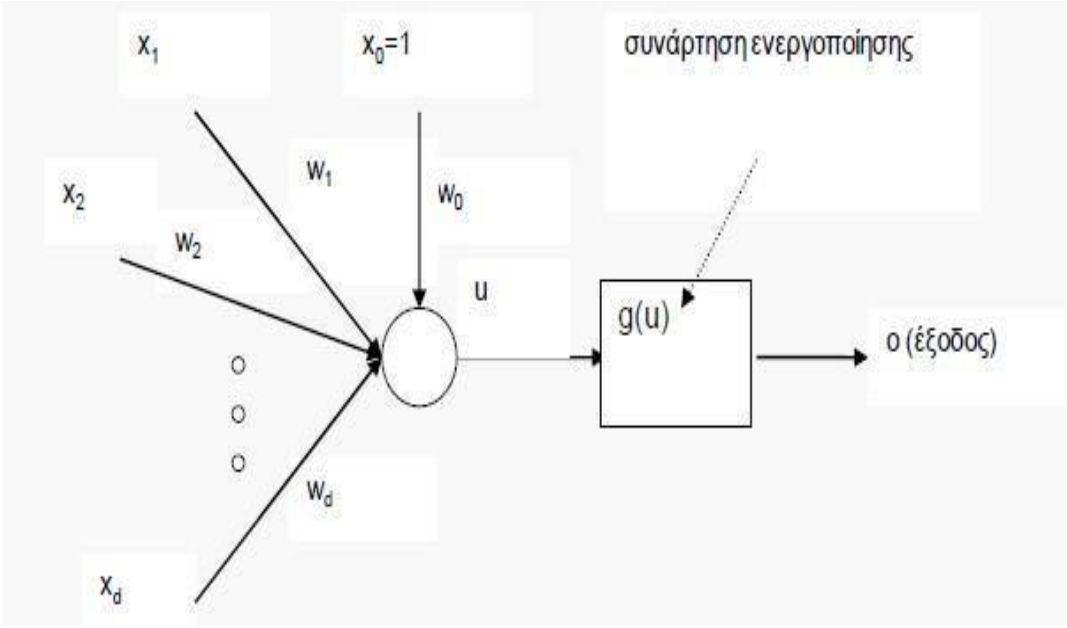
2.4 Αυτόνομο αυτοκίνητο

- **Έξυπνα σπίτια, πόλεις και υποδομές**

  Οι έξυπνοι θερμοστάτες αναλύουν τη συμπεριφορά μας προκειμένου να αποθηκεύσουν ενέργεια, ενώ οι έξυπνες πόλεις βασίζονται σε ευφυή συστήματα ρύθμισης της κυκλοφορίας για να βελτιώσουν τη συνδεσιμότητα και να μειώσουν την κυκλοφοριακή συμφόρηση.

2.3 Εισαγωγή στα νευρωνικά δίκτυα.

Το νευρωνικό δίκτυο ορίζεται (ιατρική ορολογία) ως ένα κύκλωμα από διασυνδεδεμένους υπολογιστικούς κόμβους που ονομάζονται νευρώνες. Οι νευρώνες είναι τα δομικά στοιχεία του δικτύου. Κάθε τέτοιος κόμβος δέχεται ένα σύνολο αριθμητικών εισόδων από διαφορετικές πηγές (είτε από άλλους νευρώνες, είτε από το περιβάλλον), επιτελεί έναν υπολογισμό με βάση αυτές τις εισόδους και παράγει μία έξοδο (βλέπε εικόνα 2.5).



2.5 Σχέδιο νευρωνικού δικτύου

Η έξοδος είτε κατευθύνεται στο περιβάλλον είτε τροφοδοτείται ως είσοδος σε άλλους νευρώνες του δικτύου. Υπάρχουν τρεις τύποι νευρώνων [6]:

- **Οι νευρώνες εισόδου**

  Οι νευρώνες εισόδου δεν επιτελούν κανέναν υπολογισμό, μεσολαβούν απλώς ανάμεσα στις περιβαλλοντικές εισόδους του δικτύου και στους υπολογιστικούς νευρώνες. Μοναδικό τους έργο είναι να διοχετεύσουν την είσοδο του προβλήματος.

- **Οι υπολογιστικοί νευρώνες ή κρυμμένοι νευρώνες**

  (Οι νευρώνες ανάμεσα από τους νευρώνες εισόδου και εξόδου). Οι υπολογιστικοί νευρώνες πολλαπλασιάζουν κάθε είσοδό τους με το αντίστοιχο συναπτικό βάρος και υπολογίζουν το ολικό άθροισμα των γινομένων. Το άθροισμα αυτό περνάει ως όρισμα στην συνάρτηση ενεργοποίησης που υλοποιεί εσωτερικά κάθε νευρώνας. Η τιμή που θα υπολογίσει η συνάρτηση για κάθε κόμβο αποτελεί και την έξοδο του νευρώνα για τις τρέχουσες τιμές βαρών και εισόδου.

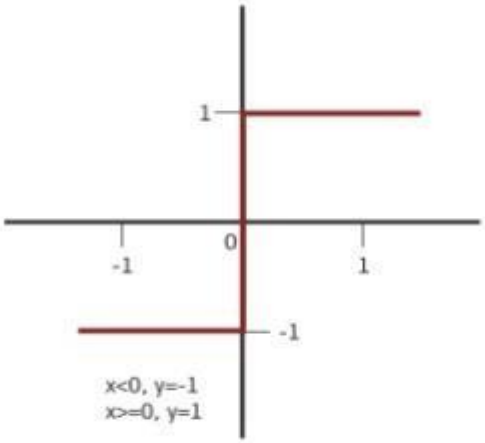
- **Οι νευρώνες εξόδου**

  Οι νευρώνες εξόδου χρησιμοποιούνται για να παρουσιάσουν στο περιβάλλον την απάντηση του δικτύου σε κάποιο πρόβλημα, όπως για παράδειγμα την εκτίμηση της κατηγορίας ενός προβλήματος κατηγοριοποίησης.

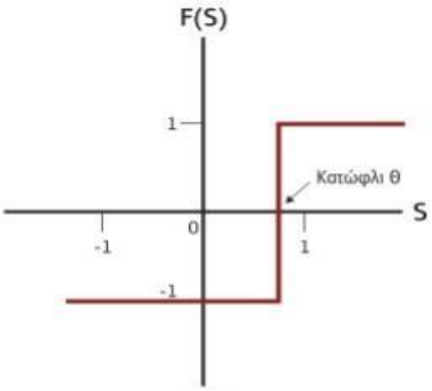
Στην εικόνα 2.5 βλέπουμε ένα τεχνητό νευρώνα με εισόδους τις **x1, x2, x3…xd** και τα αντίστοιχα βάρη τους **w1, w2, w3…wd**. Τα βάρη καθορίζουν τη σημασία κάθε εισόδου και μία συνάρτηση **ενεργοποίησης g(u)** θα καθορίσει την έξοδο του. Η έξοδος, έχοντας μια γραμμική συνάρτηση ενεργοποίησης, θα είναι το άθροισμα του γινομένου κάθε εισόδου με το αντίστοιχο βάρος της, συν μία τιμή πόλωσης Χο=1 που χρησιμοποιείται για την μετατόπιση της καμπύλης [5][6]:

$$U(x) = \sum_{i=1}^{d} w_i x_i + w_0$$

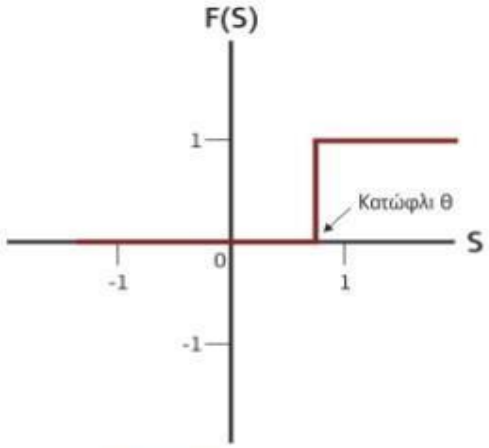
Υπάρχουν 2 ειδών συναρτήσεων μετάβασης οι γραμμικές και οι μη γραμμικές (βλέπε εικόνα 2.6 και 2.7). Οι γραμμικές συναρτήσεις έχουν πολλούς περιορισμούς όταν χρησιμοποιούνται. Κατά την διάρκεια προς εκπαίδευσης δεν μπορούν να επιστρέψουν την έξοδο προς τα πίσω και να ενημερώσουν τα βάρη του δικτύου, διότι η παράγωγος είναι μια σταθερά χωρίς κάποια σχέση με την είσοδο. Μερικά παραδείγματα είναι τα εξής [11]:



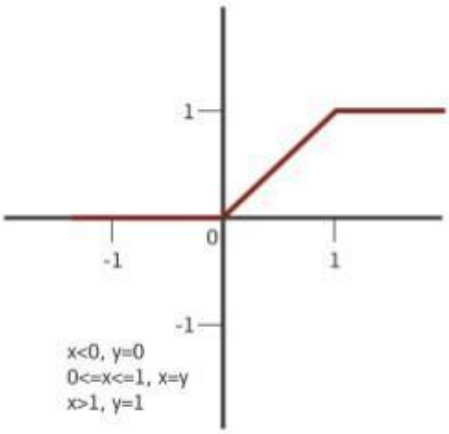
Hard Limiter



Πρόσημου
(έξοδος θετική ή αρνητική πληροφορία, +1, -1)



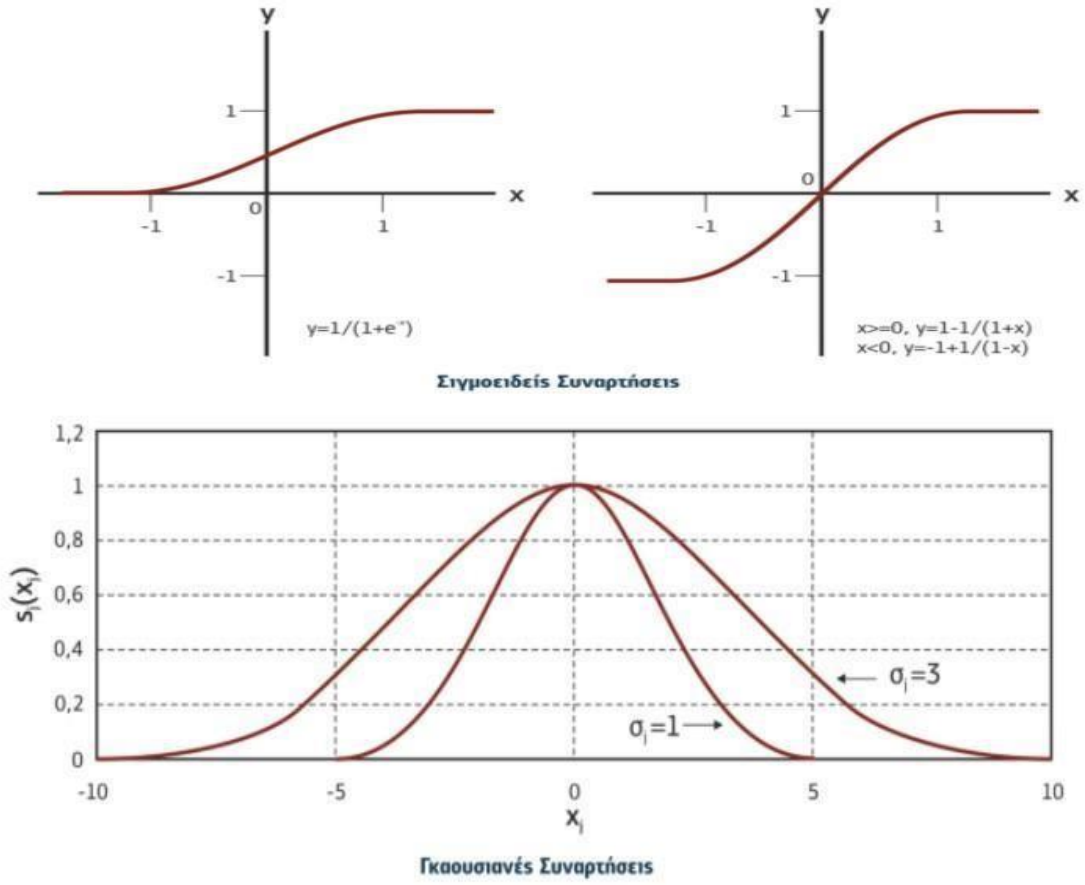
Βηματική (έξοδος 1 ή 0)



Ramping Function

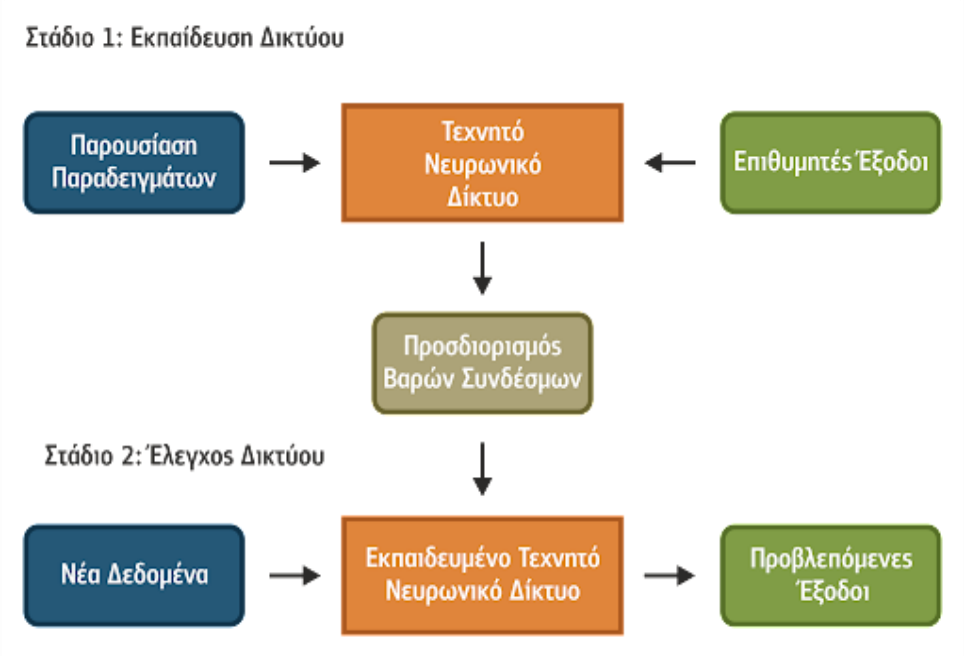
2.6 Γραμμικές συναρτήσεις

Αυτό προς λύνετε με την χρήση μη γραμμικών συναρτήσεων προς οποίες η παράγωγος έχει σχέση με την αρχική είσοδο. Προς είναι η σιγμοειδής και η γκαουσιανές.



2.7 Μη γραμμικές συναρτήσεις

Οι αριθμοί που αποτελούν το διάνυσμα εισόδου και το διάνυσμα εξόδου περιγράφουν χαρακτηριστικά του προς επίλυση προβλήματος. Το κύριο χαρακτηριστικό των νευρωνικών δικτύων είναι η ικανότητα μάθησης. Ως μάθηση νοείται η σταδιακή ικανότητα του δικτύου να επιλύσει το πρόβλημα για το οποίο σχεδιάστηκε. Η μάθηση επιτυγχάνεται μέσα από την εκπαίδευση του δικτύου, μιας επαναληπτικής διαδικασίας προσαρμογής των τιμών των παραμέτρων του δικτύου (τιμές βαρών και πόλωσης) σε κατάλληλες τιμές, ώστε να επιλύεται με σχετική επιτυχία το υπό εξέταση πρόβλημα. Στο τέλος το εκπαιδευμένο νευρωνικό δίκτυο θα ελεγχθεί ως προς την ικανότητα γενίκευσης του, την ικανότητα δηλαδή για ένα παράδειγμα παρόμοιο, αλλά όχι ίδιο, με αυτά που εκπαιδεύτηκε να παραγάγει την επιθυμητή έξοδο (βλέπε εικόνα 2.8).



2.8 Διάγραμμα εκπαίδευσης νευρωνικών δικτύων

Κατηγορίες εκπαίδευσης νευρωνικών δικτύων:

- **Μάθηση με επίβλεψη**

  Σε αυτήν την κατηγορία εποπτεύουμε το σύστημα μάθησης παρέχοντας στο δίκτυο και την είσοδο, αλλά και την επιθυμητή έξοδο για την συγκεκριμένη είσοδο με απώτερο στόχο τη γενίκευση της συνάρτησης αυτής και για εισόδους με άγνωστη έξοδο. Κάποιες από τις μεθόδους, οι οποίες συγκαταλέγονται σε αυτή την κατηγορία είναι η μάθηση με διόρθωση σφάλματος, η στοχαστική μάθηση. Οι αλγόριθμοι οι οποίοι βρίσκονται στην πρώτη κατηγορία, χρησιμοποιούνται για την εύρεση της βέλτιστης σχέσης μεταξύ εισόδων και εξόδων για κάθε ξεχωριστό ζευγάρι προτύπων.

- **Μάθηση χωρίς επίβλεψη:**

  Οι αλγόριθμοι της εν λόγω μάθησης αναφέρονται ως αυτό-οργανωμένοι (selforganized) και είναι διαδικασίες, οι οποίες δεν απαιτούν να είναι παρών ένας επιβλέπων. Βασίζονται, μάλιστα, μόνο σε τοπική πληροφορία καθ' όλη τη διάρκεια της εκπαίδευσης του Τεχνητού Νευρωνικού Δικτύου. Οι συγκεκριμένοι αλγόριθμοι οργανώνουν τα δεδομένα και ανακαλύπτουν τις σημαντικές συλλογικές ιδιότητες.

Πλεονεκτήματα Τεχνητού Νευρωνικού Δικτύου

Το επιστημονικό ενδιαφέρον για τα ΤΝΔ προκύπτει κυρίως από την δυνατότητα τους να επιλύουν δύσκολα και ενδιαφέροντα υπολογιστικά προβλήματα του πραγματικού κόσμου. Η χρήση των ΤΝΔ προσφέρει τις ακόλουθες πολύ χρήσιμες ιδιότητες και δυνατότητες [7]:

- **Αντικατάσταση μαθηματικού μοντέλου**

  Η αξία έγκειται κυρίως στο γεγονός ότι η χρήση τους δε συνεπάγεται τη δημιουργία κάποιου μοντέλου για το περιβάλλον, το οποίο καλούνται να προσομοιώσουν. Όταν η περιγραφή του περιβάλλοντος είναι ιδιαίτερα πολύπλοκη για να περιγράφει με κάποιο μαθηματικό μοντέλο, τότε η χρήση των ΤΝΔ είναι μια καλή λύση. Κατά την εκπαίδευση δίνονται στο ΤΝΔ πρότυπα εισόδου/εξόδου. Στόχος της διαδικασίας εκπαίδευσης είναι να φτάσει το ΤΝΔ σε μια τέτοια κατάσταση όπου για κάθε πρότυπο εκπαίδευσης, η έξοδός του να ταυτίζεται με την επιθυμητή έξοδο. Έτσι, δημιουργείται μια συσχέτιση μεταξύ της εισόδου και της εξόδου, χωρίς όμως τη χρήση κάποιου προκαθορισμένου στατιστικού ή άλλου μοντέλου.

- **Προσαρμογή**

  Έχουν την ικανότητα να μεταβάλλουν την απόκριση τους μεταβάλλοντας ελαφρώς τα βάρη των νευρώνων, ανάλογα με τα πρότυπα εισόδου.

- **Δυνατότητα εύκολης υλοποίησης σε hardware**

  Η απλή δομή των νευρώνων και το γεγονός ότι ένα ΤΝΔ αποτελείται από πολλά ίδια στοιχεία (νευρώνες) καθιστά σχετικά απλή την υλοποίηση των ΤΝΔ σε hardware.

- **Ανεκτικότητα σε σφάλματα**

  Τα ΤΝΔ που έχουν υλοποιηθεί σε υλικό (hardware) έχουν την ιδιότητα της ανεκτικότητας σε σφάλματα, γιατί η απόδοση του συστήματος μειώνεται ομαλά σε περίπτωση λάθους. Ακόμα και αν καταστραφεί ένας νευρώνας, το ΤΝΔ θα συνεχίσει να λειτουργεί με κάπως μειωμένη απόδοση.

- **Ανοχή στο θόρυβο**

  Τα ΤΝΔ παρουσιάζουν ανοχή στο θόρυβο υπό την έννοια ότι αν εκπαιδευτούν για την αναγνώριση προτύπων, τότε σε αυτή την περίπτωση είναι σε γενικές γραμμές εφικτή η ταξινόμηση από το νευρωνικό ακόμα και όταν η είσοδος έχει υποστεί κάποια αλλοίωση.

2.4 Αρχιτεκτονικές νευρωνικών δικτύων

Όπως αναφέρθηκε και νωρίτερα, ένα νευρωνικό δίκτυο αποτελείται από ένα πλήθος νευρώνων που συνδέονται μεταξύ τους. Ο τρόπος με τον οποίο συνδέονται μεταξύ τους οι νευρώνες, καθορίζει την αρχιτεκτονική του δικτύου. Η επιλογή αρχιτεκτονικής, έχει βαρύνουσα σημασία ως προς την απόφαση στη σχεδίαση ενός νευρωνικού δικτύου, διότι παίζει καθοριστικό ρόλο στην ικανότητα του νευρωνικού δικτύου να επιλύει συγκεκριμένα προβλήματα. Ταυτοχρόνως, συνδέεται στενά με τους αλγόριθμους εκπαίδευσης που χρησιμοποιούνται. Γενικά μπορούμε να διακρίνουμε τέσσερις κύριες κλάσεις αρχιτεκτονικών νευρωνικών δικτύων [6]:

1. **Δίκτυα Πρόσθιας Τροφοδότησης Ενός Επιπέδου**

   Ένα νευρωνικό δίκτυο με επίπεδα έχει τους νευρώνες του οργανωμένους σε ομάδες ή αλλιώς επίπεδα. Οι νευρώνες κάθε επιπέδου δε συνδέονται μεταξύ τους, αλλά έχουν συνδέσεις από και προς νευρώνες άλλων επιπέδων ή έχουν συνδέσεις εισόδου και εξόδου του δικτύου. Στην απλούστερη μορφή νευρωνικού δικτύου με επίπεδα που εξετάζουμε εδώ, το δίκτυο αποτελείται από

ένα και μόνο επίπεδο νευρώνων. Το δίκτυο διαθέτει ένα επίπεδο εισόδου, με κόμβους, οι οποίοι συνδέονται ο καθένας με όλους τους νευρώνες του μοναδικού επιπέδου του δικτύου. Οι κόμβοι του επιπέδου εισόδου δε θεωρούνται νευρώνες μιας και δεν εκτελούν κανένα υπολογισμό και χρησιμεύουν μόνο στο να παρέχουν το διάνυσμα εισόδου στο δίκτυο, στην είσοδο κάθε νευρώνα. Ο υπολογισμός γίνεται στο επίπεδο των νευρώνων και το σύνολο των εξόδων των νευρώνων αυτού του επιπέδου αποτελεί το διάνυσμα εξόδου από το δίκτυο. Λόγω της ιδιότητας αυτής της μορφής των δικτύων να μεταβιβάζεται η υπολογιστική διαδικασία από την είσοδο του δικτύου προς την έξοδο, τα δίκτυα αυτά ονομάζονται Δίκτυα Πρόσθιας Τροφοδότησης (Feedforward Networks).

2. Δίκτυα Πρόσθιας Τροφοδότησης Πολλών Επιπέδων

Η δεύτερη κλάση νευρωνικών δικτύων πρόσθιας τροφοδότησης που εξετάζονται διαφέρει από την προηγούμενη ως προς την ύπαρξη περισσότερων από ένα επιπέδων νευρώνων. Τα επιπλέον επίπεδα ονομάζονται κρυφά επίπεδα και οι νευρώνες που τα αποτελούν, κρυφοί νευρώνες. Η λειτουργία τους συνίσταται στο γεγονός ότι παρεμβαίνουν μεταξύ της εξωτερικής εισόδου στο δίκτυο και της εξόδου από αυτό. Προσθέτοντας ένα ή περισσότερα επίπεδα κρυφών νευρώνων σε ένα νευρωνικό δίκτυο, το δίκτυο αποκτά τη δυνατότητα να αφομοιώνει περισσότερες πληροφορίες για τα δεδομένα εισόδου, μέσω των περισσότερων συνάψεων που έχει στη διάθεσή του και των μεγαλύτερης πολυπλοκότητας αλληλεπιδράσεων, μεταξύ των νευρώνων, που δημιουργούνται. Η αξία της δυνατότητας του δικτύου να επεξεργάζεται μεγαλύτερης πολυπλοκότητας δεδομένα, γίνεται φανερή όταν η διάσταση του διανύσματος εισόδου είναι μεγάλη. Η διαδικασία υπολογισμού της εξόδου του δικτύου είναι παρόμοια με αυτή της προηγούμενης κατηγορίας δικτύων. Οι κόμβοι του επιπέδου εισόδου του δικτύου παρέχουν το διάνυσμα εισόδου του δικτύου στους κόμβους του 1ου κρυφού επιπέδου. Εκτελείται ο υπολογισμός στους κόμβους του 1ου επιπέδου και στη συνέχεια η έξοδος του 1ου κρυφού επιπέδου παρέχεται σαν είσοδος στους νευρώνες του 2ου κρυφού επιπέδου.

Αφού εκτελεστεί ο υπολογισμός η έξοδος μεταφέρεται στο επόμενο επίπεδο νευρώνων, και η διαδικασία συνεχίζεται, με το ένα επίπεδο να χρησιμοποιεί σαν είσοδο την έξοδο του προηγούμενου, μέχρι να γίνει ο υπολογισμός της απάντησης του δικτύου από το τελευταίο επίπεδο.

3. Αναδρομικά Δίκτυα

Τα αναδρομικά δίκτυα διαφέρουν από τα δίκτυα προς τα εμπρός τροφοδότησης ως προς το ότι περιέχουν ένα τουλάχιστον βρόχο ανάδρασης (feedback loop). Για παράδειγμα, ένα αναδρομικό νευρωνικό δίκτυο μπορεί να αποτελείται από ένα επίπεδο νευρώνων, καθένας από τους οποίους τροφοδοτεί την έξοδό του ως είσοδο προς όλους τους νευρώνες του δικτύου. Ακόμη υπάρχει δυνατότητα αυτοανάδρασης, δηλαδή ένας νευρώνας μπορεί να δέχεται σαν είσοδο την ίδια του την έξοδο.

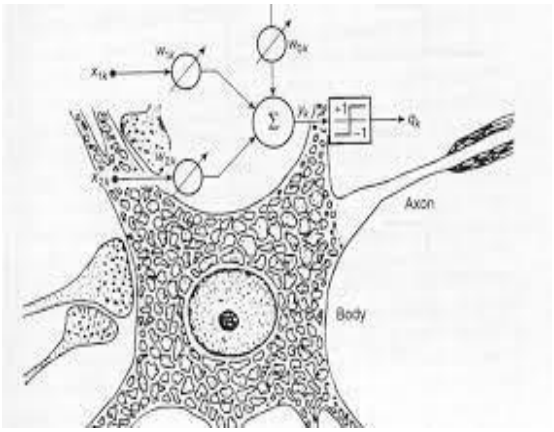
4. Δίκτυα με Δομή Δικτυωτού

Τα νευρωνικά δίκτυα αυτής της μορφής αποτελούνται από συστοιχίες νευρώνων, καθένας από τους οποίους συνδέεται με τους κόμβους εισόδου στο δίκτυο. Το δίκτυο αυτό μπορεί να είναι μονοδιάστατο, δισδιάστατο ή μεγαλύτερης διάστασης, ανάλογα με τη διάσταση του χώρου στον οποίο ανήκει το γράφημα που περιγράφει την αρχιτεκτονική του. Στην ουσία τα νευρωνικά δίκτυα αυτής της μορφής είναι δίκτυα προς τα εμπρός τροφοδότησης με τους νευρώνες εξόδου να είναι διατεταγμένοι σε γραμμές και στήλες, λαμβάνοντας υπόψη και τη διάσταση.

2.5 Εφαρμογές των νευρωνικών δικτύων

Τα νευρωνικά δίκτυα είναι εφαρμόσιμα σχεδόν σε κάθε κατάσταση στην οποία ισχύει μια σχέση μεταξύ μεταβλητών πρόβλεψης και προβλεπόμενες μεταβλητές. Παρακάτω θα παρουσιάσουμε μερικές εφαρμογές των νευρωνικων δικτύων [8][9].

- **Ιατρική διάγνωση**



2.9 Νευρωνικά στην ιατρική

Ένα ευρύ φάσμα ιατρικά συσχετιζόμενων ενδείξεων, όπως ο συνδυασμός της καρδιακής συχνότητας, τα επίπεδα των διαφόρων ουσιών στο αίμα, ο ρυθμός της αναπνοής μπορούν να παρακολουθηθούν. Η εκδήλωση μιας συγκεκριμένης ιατρικής κατάστασης, γίνεται να συσχετιστεί με ένα πολύπλοκο συνδυασμό μεταβολών σε ένα υποσύνολο μεταβλητών που παρακολουθούνται. Τα νευρωνικά δίκτυα έχουν χρησιμοποιηθεί για την αναγνώριση αυτού του προτύπου πρόβλεψης, ώστε να χορηγηθεί η κατάλληλη θεραπεία.

- **Χρηματοοικονομικές προβλέψεις**

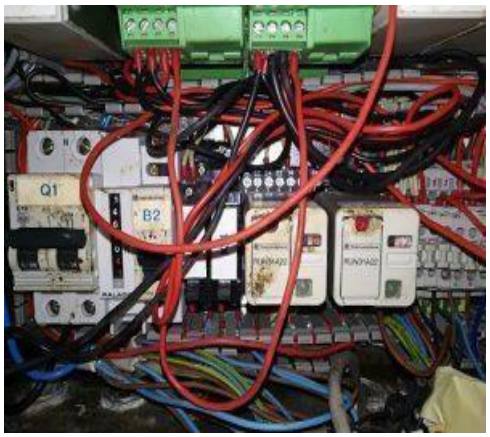
2.10 Πίνακας στατιστικών

Οι διακυμάνσεις των τιμών των μετοχών και των χρηματιστηριακών δεικτών είναι ακόμα ένα παράδειγμα για την χρήση νευρωνικών δικτύων. Τα νευρωνικά δίκτυα χρησιμοποιούνται από πολλούς τεχνικούς αναλυτές, ώστε να κάνουν προβλέψεις σχετικά με τις

τιμές των μετοχών, βασιζόμενοι σε ένα μεγάλο αριθμό παραγόντων, όπως λόγου χάριν, τις προηγούμενες επιδόσεις άλλων αποθεμάτων και διαφόρων οικονομικών δεικτών.

- **Παρακολούθηση της κατάστασης των μηχανημάτων**

  Τα νευρωνικά δίκτυα μπορούν να συμβάλλουν στη μείωση του κόστους της συντήρησης μηχανημάτων. Τα δίκτυα έχουν τη δυνατότητα μέσα από τους ήχους που βγάζουν τα μηχανήματα να αντιλαμβάνονται εάν αυτά λειτουργούν

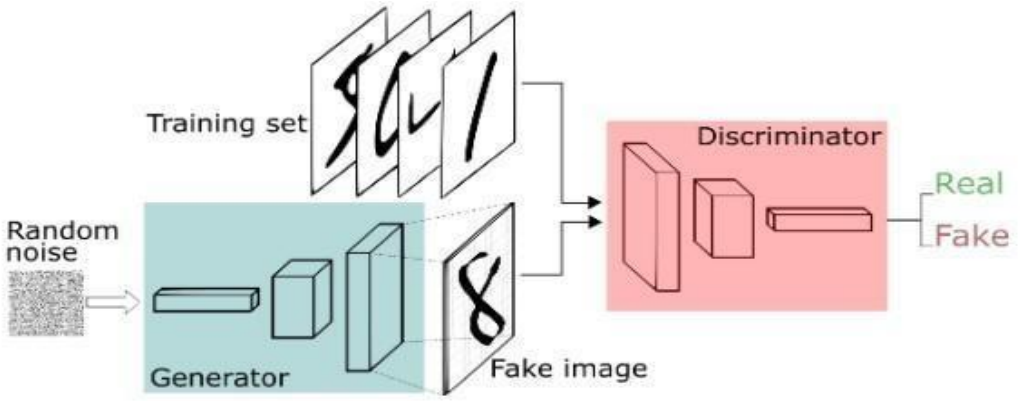


σωστά η εάν είναι στα πρόθυρα εμφάνισης βλάβης. Σε επόμενο βήμα, μάλιστα, μπορούν οι μηχανές να φτάσουν σε σημείο που θα ειδοποιούν τους τεχνικούς για κάποια επικείμενη βλάβη προτού συμβεί, προλαβαίνοντας έτσι δαπάνες και χρονικές καθυστερήσεις.

2.11 Εικόνα κυκλώματος

2.6 Αρχιτεκτονική Generative adversarial networks (GAN´s)

Τα Generative Adversarial Networks (GAN) είναι αλγοριθμικές τεχνικές που χρησιμοποιούν δυο νευρωνικά δίκτυα [3], αντιπαραθέτοντας το ένα με το άλλο προκειμένου να δημιουργούν νέες περιπτώσεις δεδομένων που μπορούν να παρουσιαστούν ως αληθινά δεδομένα. Αυτή η αρχιτεκτονική χρησιμοποιείται ευρέως για την δημιουργία εικόνων, βίντεο, παραγωγή φωνής.



2.12 Αρχιτεκτονική GAN μοντέλου

Για να κατανοηθεί η αρχιτεκτονική (βλέπε εικόνα 2.12) των GAN πρέπει να γνωρίζουμε τον τρόπο που λειτουργούν η γεννήτρια (generator) και ο διακριτής (discriminator). Οι αλγόριθμοι διάκρισης (ή διδακτικοί αλγόριθμοι) προσπαθούν να ταξινομήσουν τα δεδομένα εισόδου, δηλαδή ανάλογα με τα χαρακτηριστικά δεδομένα προβλέπουν μια ετικέτα για την κατηγορία που ανήκουν. Ενώ αντίθετα οι γενετικοί αλγόριθμοι προσπαθούν να μάθουν από τους αλγορίθμους διάκρισης και να παραγάγουν δεδομένα που έχουν τα χαρακτηριστικά της συγκεκριμένης ετικέτας [3].

Πώς λειτουργεί το GAN;

Το ένα νευρωνικό δίκτυο που ονομάζεται γεννήτρια δημιουργεί πλαστά δεδομένα. Τα δεδομένα αυτά μαζί με τα αληθινά περνάνε στο άλλο νευρωνικό δίκτυο ως είσοδοι και ο διακριτής τα αξιολογεί ως προς την αυθεντικότητά τους. Αναλυτικότερα, αποφασίζει εάν κάθε περίπτωση δεδομένων που εξετάζει ανήκει στο πραγματικό σύνολο δεδομένων εκπαίδευσης ή αν αυτά είναι πλαστά. Στη συνέχεια, ο διακριτής ενημερώνεται για να γίνει καλύτερος. Ο διαχωρισμός πραγματικών και ψεύτικων δειγμάτων στον επόμενο γύρο, είναι το σημαντικότερο, καθώς η γεννήτρια ενημερώνεται με βάση την επιτυχία ή την αποτυχία των παραγόμενων δειγμάτων να εξαπατήσουν το διακριτή. Κάθε φορά που ο τελευταίος εντοπίζει ότι η εικόνα είναι ψεύτικη «τιμωρεί» την γεννήτρια μέσω ενός σφάλματος το οποίο ενημερώνει τα βάρη του δικτύου και το κάνει να γίνει πιο καλό στην παραγωγή πλαστών δεδομένων. Έπειτα από τις εποχές εκπαίδευσης του δικτύου και ανάλογα τα πρότυπα εκπαίδευσης σε κάθε

εποχή το δίκτυο καταλήγει να αποκτά μια ικανοποιητική ικανότητα γενίκευσης. Σύμφωνα με αυτήν η γεννήτρια θα φτάσει σε ένα σημείο, όπου τα δείγματα που παράγει θα «ξεγελάνε» τον διακριτή και θα μοιάζουν με αληθινά δείγματα [3].

Εφαρμογές των GAN's

- **Επιστήμη**

  Τα GAN μπορούν να βελτιώσουν αστρονομικές εικόνες και να προσομοιώσουν βαρυτικούς φακούς για έρευνα σκοτεινής ύλης. Τα GAN έχουν επίσης εκπαιδευτεί να προσεγγίζουν με ακρίβεια τα σημεία συμφόρησης σε υπολογιστικά δαπανηρές προσομοιώσεις πειραμάτων διαφόρων σωματιδίων φυσικής [10].

- **Βιντεοπαιχνίδια**

  Τα GAN χρησιμοποιούνται ως μέθοδο αναβάθμισης 2D υφών χαμηλής ανάλυσης σε παλιά βιντεοπαιχνίδια, μετατρέποντάς τα σε 4k ή υψηλότερες αναλύσεις μέσω εκπαίδευσης εικόνας και στη συνέχεια γίνεται δειγματοληψία προς τα κάτω για να ταιριάζουν στη μητρική ανάλυση του παιχνιδιού .

- **Κακοβουλες εφαρμογές**

  Τα GAN μπορούν να χρησιμοποιηθούν για τη δημιουργία μοναδικών, ρεαλιστικών φωτογραφιών προφίλ ανθρώπων που δεν υπάρχουν, προκειμένου να αυτοματοποιήσουν τη δημιουργία πλαστών προφίλ στα κοινωνικά μέσα.

2.7 Τεχνητή νοημοσύνη και νευρωνικά δίκτυα στον χρωματισμό ασπρόμαυρων εικόνων

Σε αυτό το σημείο της μελέτης αναλύεται ο λόγος που αυτή αποτελεί ένα πρόβλημα τεχνητής νοημοσύνης, αλλά και γιατί υπάρχει δυσκολία στην υλοποίησή της.

Grayscale image Final prediction

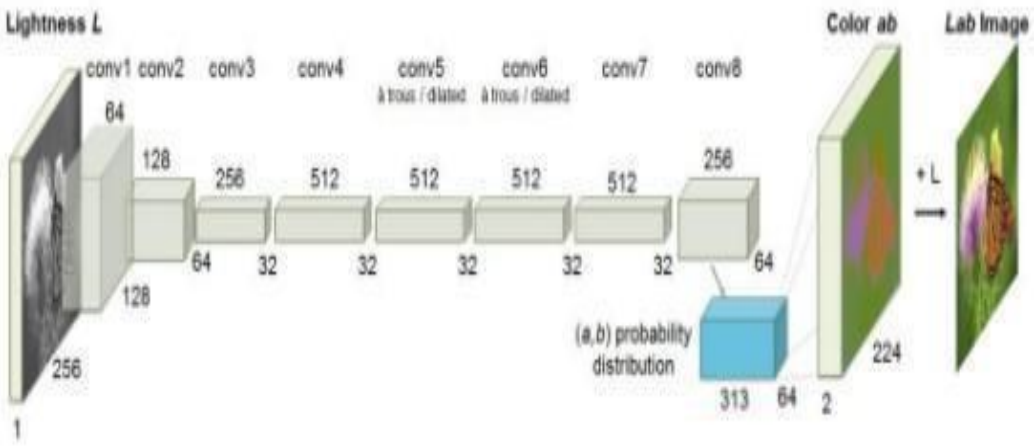
2.13 παράδειγμα χρωματισμού ασπρόμαυρης εικόνας με χρήση νευρωνικών

Αφού σκοπός είναι να φτιαχτεί ένα πρόγραμμα που θα χρωματίζει κάθε ασπρόμαυρη εικόνα (βλέπε εικόνα 2.13), αυτό από μόνο του δημιουργεί δυσκολίες για τα συστήματα τεχνητής νοημοσύνης.

Ο χρωματισμός της εικόνας περιγράφεται ως η διαδικασία εκχώρησης χρωμάτων στα εικονοστοιχεία μιας εικόνας, σε κλίμακα του γκρι. Αυτό ακριβώς γεννά προβλήματα διότι, χωρίς προηγούμενες πληροφορίες σχετικά με την ίδια την εικόνα, υπάρχουν στην πλειοψηφία των περιπτώσεων περισσότεροι από ένας πιθανοί χρωματισμοί. Με άλλα λόγια, τα διαφορετικά χρώματα ενός αντικειμένου δεν μπορούν συνήθως να διακριθούν μεταξύ τους απλά κοιτάζοντας καθαυτό το αντικείμενο, ειδικά σε κλίμακα του γκρι (για το αντικείμενο). Για παράδειγμα, ένας κόκκινος καναπές πιθανόν να έχει το ίδιο παρουσιαστικό με έναν μαύρο καναπέ σε μια εικόνα, σε

κλίμακα του γκρι. Λόγω αυτής της ιδιότητας, ο αυτόματος χρωματισμός της εικόνας είναι κατά γενική ομολογία ένα σχετικά δύσκολο εγχείρημα.

Εν προκειμένω, η ιδέα της εργασίας εφάπτεται στην ανάπτυξη ενός δικτύου (βλέπε εικόνα 2.14) που θα εκπαιδευτεί στα πλαίσια της μάθησης με επίβλεψη. Δηλαδή, μέσα από ένα μεγάλο dataset με έγχρωμες εικόνες θα δοθεί στο δίκτυο μια ασπρόμαυρη εικόνα ως είσοδος και εκείνο θα την ζωγραφίζει ανάλογα με το πόσο καλά έχει εκπαιδευτεί. Αυτή η εικόνα που θα παραχθεί μαζί με την αληθινή έγχρωμη εικόνα θα περνάνε ως είσοδοι σε ένα άλλο δίκτυο που επίσης θα εκπαιδεύεται, ούτως ώστε να διακρίνει τα αληθινά από τα ψεύτικα δείγματα, και όταν θα διακρίνει τα ψεύτικα θα στέλνει ένα σφάλμα στο αρχικό δίκτυο που παράγει τις εικόνες και έτσι θα εκπαιδεύεται συνεχώς προκειμένου να επιτευχθεί ο τελικός σκοπός.



2.14 Παράδειγμα ενός δικτύου με τα επίπεδα του

Κεφάλαιο 3

Υλοποίηση προγράμματος

3.1 Ορισμός του προβλήματος

Με βάση ένα δεδομένο σύνολο έγχρωμων εικόνων με τοπία, ανθρώπους και διάφορες άλλες απεικονίσεις θα πρέπει να δημιουργηθεί ένα απλό πρόγραμμα που θα είναι σε θέση να χρωματίσει ορθά κάθε ασπρόμαυρη φωτογραφία, η οποία όμως δεν έχει χρησιμοποιηθεί κατά την διάρκεια της εκπαίδευσης. Τα βήματα για την κατασκευή του περιγράφονται αναλυτικά βήμα προς βήμα στο κεφάλαιο που ακολουθεί. Επιπροσθέτως, στο ίδιο κεφάλαιο αναλύονται οι σχεδιαστικές επιλογές δικτύων, καθώς και τα αποτελέσματά τους.

3.2 Εγκατάσταση βιβλιοθηκών

Για την κατασκευή του προγράμματος για το οποίο γίνεται λόγος χρειάστηκαν μερικές βιβλιοθήκες, ώστε να υποστηρίξουν επικουρικά τις υπολογιστικές πράξεις και στην κατασκευή του δικτύου. Αναλυτικότερα:

- **Python:** Το πρώτο και βασικότερο βήμα είναι να εγκατασταθεί η python. Η έκδοση (version) της python ελέγχεται με την εντολή «python --version». Εάν εκείνη δεν είναι εγκατεστημένη, τότε αρκεί μία περιήγηση στο επίσημο site της python για να αποκτηθεί η επιθυμητή έκδοση. Στο παρόν project χρησιμοποιείται η έκδοση «3.5.4».

- **Pip:** Το pip είναι ένας «διαχειριστής πακέτων» που χρησιμοποιείται για την εγκατάσταση και την συντήρηση των βιβλιοθηκών της python. Για την εγκατάσταση της pip στα windows αρχικά πληκτρολογείται στο CMD το εξής: «curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py» και στη συνέχεια «python get-pip.py». Έπειτα με την pip help εντολή επιβεβαιώνεται η εγκατάσταση. Σε τελευταίο στάδιο αυτό προστίθενται στις μεταβλητές περιβάλλοντος των windows.

- **Keras:** Είναι μια βιβλιοθήκη ανοικτού κώδικα που χρησιμοποιείται για την δημιουργία νευρωνικών δικτύων. Το Keras λειτουργεί ως διεπαφή για τη βιβλιοθήκη TensorFlow. Σχεδιασμένο για να επιτρέπει γρήγορο πειραματισμό με βαθιά νευρωνικά δίκτυα, εστιάζει στο να είναι φιλικό προς τον χρήστη, αρθρωτό και επεκτάσιμο. Για το κατέβασμά του χρησιμοποιείται «pip install tensorflow==1.8.0», όμως χρειάζεται πρώτα να ελεγχθεί η έκδοση της tensorflow που χρησιμοποιείται, με την εντολή «python -c 'import tensorflow as tf; print(tf.__version__)'». Στην συνέχεια επιλέγεται «pip install keras».

- **Numpy:** Το numpy είναι η βασική βιβλιοθήκη για επιστημονικούς υπολογισμούς και για πράξεις με πίνακες. Η εγκατάσταση της είναι πολύ απλή μέσω της pip, όπου επιλέγεται pip install «numpy».

- **Matplotlib:** Το matplotlib είναι μια βιβλιοθήκη σχεδίασης για την python και αριθμητική επέκταση της numpy. Το Pyplot είναι μια λειτουργική μονάδα Matplotlib που παρέχει μια διεπαφή, και βοηθάει στην δημιουργία σχεδιαγραμμάτων.

3.3 Δημιουργία Dataset

Πρώτο βήμα της εκπαίδευσης του νευρωνικού δικτύου που κατασκευάστηκε ήταν η δημιουργία ενός dataset με πληθώρα έγχρωμων φωτογραφιών που περιέχουν όσο το δυνατόν περισσότερες κατηγορίες είναι εφικτό (πχ. ζώα, ανθρώπους, σπίτια, θάλασσες, ουρανό, αυτοκίνητα κλπ.). Αυτό βοήθησε στο να εκπαιδευτεί το δίκτυο σε ένα πλήθος αντικειμένων, ώστε να υπάρχει ένα ρεαλιστικό αποτέλεσμα σε κάθε φωτογραφία. Έτσι, η αρχική εργασία ήταν η εύρεση και η εγκατάσταση έγχρωμων βίντεο που να περιέχουν όλες τις προαναφερθείσες κατηγορίες, ενώ στη συνέχεια γράφτηκε κώδικας, με σκοπό τη διάκριση και την αποθήκευση κάθε frame του βίντεο στον φάκελο του Dataset.

Παρακάτω παρατίθεται ο σχετικός κώδικας:

```
from os import listdir
from os.path import isfile, join
import cv2
import os
import sys

video=sys.argv[1]
vidcap = cv2.VideoCapture(video)
path='C:[redacted]'
def TakeFrameFromVideo(vidcap,path):
    success,image = vidcap.read()
    count = 0
    while success:
        cv2.imwrite(os.path.join(path , "%d.jpg" % count), image)
        success,image = vidcap.read()
        count += 1
    print("%d" % count)

TakeFrameFromVideo(vidcap,path)
```

3.1 Κώδικας για την δημιουργία του Dataset

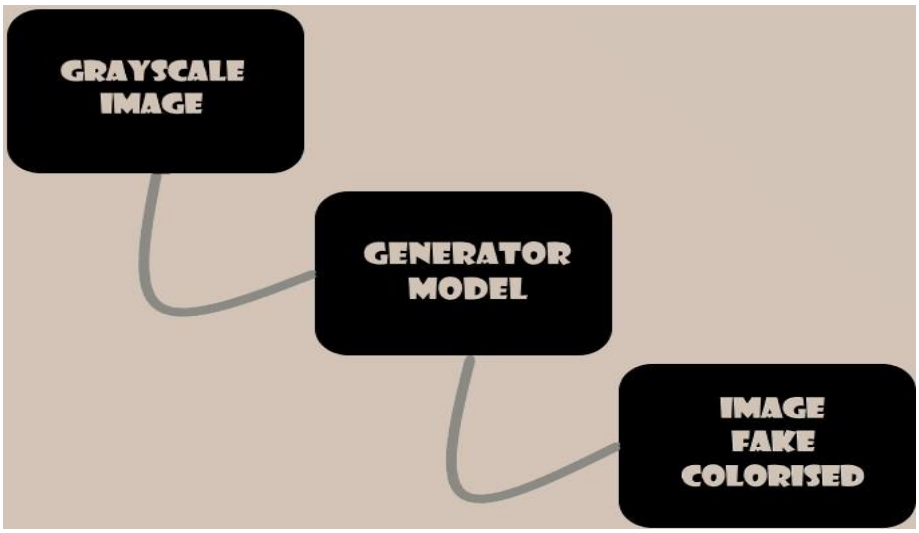
Αφού συγκεντρώθηκαν περίπου 25 χιλιάδες φωτογραφίες από τα frame των βίντεο γράφτηκε κώδικας ώστε να δημιουργηθεί ένας φάκελος train με 15 χιλιάδες φωτογραφίες για την εκπαίδευση του δικτύου, ένας validation με 5 χιλιάδες φωτογραφίες για την επικύρωση, ένας test με 5 χιλιάδες φωτογραφίες για το τεστάρισμα του αποτελέσματος του εκπαιδευόμενου δικτύου.

3.2 Φάκελοι εκπαίδευσης, επικύρωσης και τεσταρίσματος.

3.4 Δημιουργία και εκπαίδευση Generator

Το μοντέλο του generator είναι, όπως έχει ήδη ειπωθεί υπεύθυνο για τη δημιουργία νέων, ψεύτικων, αλλά εύλογων έγχρωμων εικόνων. Αυτό γίνεται δίνοντας μια ασπρόμαυρη εικόνα ως είσοδο στο μοντέλο, μεγέθους 256*256, του generator. Αυτό γίνεται μέχρι ο generator να εκπαιδευτεί κατάλληλα, από το σφάλμα που επιστρέφει ο discriminator, ώστε να χρωματίζει εύλογα τις ασπρόμαυρες εικόνες. Οι εικόνες που θα παράγει δεν θα είναι παρόμοιες, σαφώς, με τις ασπρόμαυρες. Συνεπώς, ως έξοδο θα προκύπτει μια ψεύτικα χρωματισμένη εικόνα που θα χρησιμεύσει παρακάτω για να εκπαιδευτεί ο discriminator σε αληθινές (real) και ψεύτικες (fake) εικόνες και με την σειρά του να επιστρέψει μια τιμή σφάλματος για την εκπαίδευση του generator.

Παρακάτω παρατίθεται μια φωτογραφία που δείχνει πως ακριβώς λειτουργεί ο generator.



3.3 Σχεδιάγραμμα λειτουργίας generator

Για την εκπαίδευση του μοντέλου της γεννήτριας (generator) πρέπει να φορτωθούν κατάλληλα οι εικόνες από τον φάκελο train. Για τον σκοπό αυτό γράφτηκε ο παρακάτω κώδικας οπού τις φορτώνει από το path και τις μετατρέπει σε διαστάσεις 256*256, την ίδια στιγμή επιλέγεται και ο αριθμός των εικόνων που θα φορτωθούν ώστε να συμμετάσχουν στη εκπαίδευση του generator.

```
path = 'C:/Users/GIORGOS/Desktop/networks/train/'

# Normalize images - divide by 255
train_datagen = ImageDataGenerator(rescale=1. / 255)

# Resize images, if needed
train = train_datagen.flow_from_directory(path,
                                          target_size=(256, 256),
                                          batch_size=600,
                                          # shuffle=True,
                                          class_mode=None)
model = define_generator()
gray = []
colorfull = []
n_samples = 1
```

3.4 load images from train folder

Οι αρχικές έγχρωμες εικόνες που φορτώθηκαν με τον παραπάνω κώδικα είναι τύπου RGB και αποθηκεύονται σε έναν πίνακα train και είναι οι εικόνες που

χρησιμοποιούνται στον discriminator ως αληθινές (real). Και όταν εκείνες εμφανίζονται με τον παρακάτω κώδικα μοιάζουν κάπως έτσι:

```
pyplot.imshow(dataset[1][ix])
filename = 'Real_plot_e%03d.png' % (epoch+1)
pyplot.savefig(filename)
pyplot.close()
```

3.5 Original image

```
def select_real_samples(dataset,j,gray):
    # choose image and converted
    lab_image= rgb2lab(dataset[1][j])
    lab_image = (lab_image + 128) / 255
    L=lab_image[:, :, 0]
    lab_image = lab_image.reshape((1, 256, 256, 3))
    gray.append(L)
    # generate 'real' class labels (1)
    y = ones((1, 1))
    return lab_image, y
```

```
pyplot.imshow(real,cmap="gray")
filename = 'Gray_plot_e%03d.png' % (epoch+1)
pyplot.savefig(filename)
pyplot.close()
```

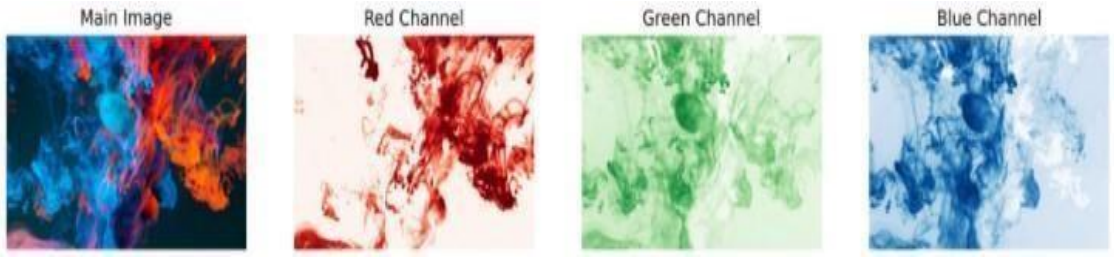
3.6 Grayscale image

Οι ίδιες εικόνες μετατράπηκαν σε ασπρόμαυρες, ώστε να δημιουργηθεί μία είσοδος για το μοντέλο της γεννήτριας (generator) μέσω της συνάρτησης select_real_samples. Μέσα σε αυτήν μετασχηματίστηκε η εικόνα σε LAB από RGB το έγχρωμο κομμάτι της εικόνας αποθηκεύεται και επιστρέφεται από την συνάρτηση και το ασπρόμαυρο κομμάτι επίσης.

Σύγκριση LAB και RGB

Όπως είναι γνωστό, όταν φορτώνεται μια εικόνα, γίνεται η λήψη ενός πίνακα rank3 (ύψος, πλάτος, χρώμα) με τον τελευταίο άξονα να περιέχει τα δεδομένα χρώματος για την εν λόγω εικόνας. Αυτά τα δεδομένα αντιπροσωπεύουν το χρώμα στον χρωματικό χώρο RGB και υπάρχουν 3 αριθμοί για κάθε εικονοστοιχείο που υποδεικνύουν πόσο κόκκινο (R), πράσινο (G) και μπλε (B) είναι το εικονοστοιχείο.

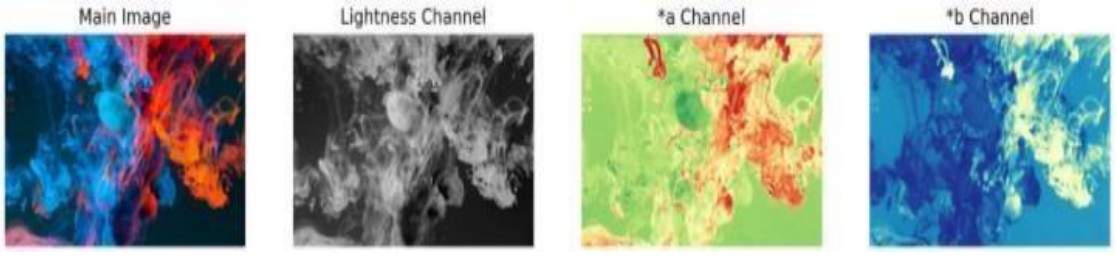
Παρακάτω παρουσιάζεται πως είναι μια αρχική εικόνα RGB και στην συνέχεια πως αντιστοιχίζεται στο κόκκινο το πράσινο και το μπλε κανάλι.



3.7 Αντιστοίχιση καναλιών rgb εικόνας

Στο χρωματικό χώρο L*a*b, υπάρχουν πάλι τρεις αριθμοί για κάθε εικονοστοιχείο, αλλά αυτοί οι αριθμοί έχουν διαφορετική σημασία. Ο πρώτος αριθμός (κανάλι) ***L***, κωδικοποιεί τη φωτεινότητα καθενός εικονοστοιχείου και όταν αυτό απεικονιστεί το κανάλι εμφανίζεται ως ασπρόμαυρη εικόνα. Τα κανάλια ****a και *b*** κωδικοποιούν πόσο **πράσινο-κόκκινο** και **κίτρινο-μπλε** είναι κάθε pixel, αντίστοιχα.

Στην παρακάτω εικόνα παρατίθεται ξεχωριστά κάθε κανάλι χρωματικού χώρου L*a*b.



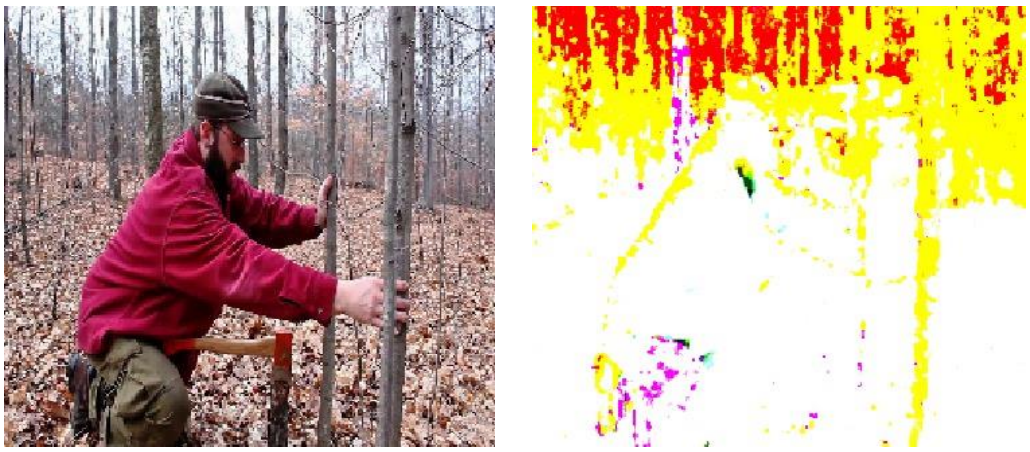
3.8 Αντιστοίχιση καναλιών lab εικόνας

Για την εκπαίδευση ενός μοντέλου για επιχρωματισμό, θα πρέπει να δοθεί σε αυτό μια εικόνα σε κλίμακα του γκρι και να χρωματιστεί. Όταν χρησιμοποιείται το L*a*b, κάθε φορά που δίνεται το κανάλι L στο μοντέλο (που είναι η εικόνα σε κλίμακα του γκρι) σκοπός είναι να προβλέψει τα άλλα δύο κανάλια (*a,*b). Ύστερα, συνδέονται όλα τα κανάλια και προκύπτει εν τέλει η επιθυμητή πολύχρωμη εικόνα.

Εάν χρησιμοποιηθούν RGB, πρέπει πρώτα να μετατραπεί η εικόνα σε κλίμακα του γκρι, να τροφοδοτηθεί κατ' αυτόν τον τρόπο στο μοντέλο και να "ελπίζουμε" ότι το δίκτυο θα προβλέψει 3 αριθμούς, για να βγάλει την εικόνα, γεγονός που από μόνο του είναι μια πολύ πιο δύσκολη και ασταθής διεργασία, αφού υπάρχουν πολύ περισσότεροι δυνατοί συνδυασμοί "3 ψηφίων" σε σύγκριση με δύο αριθμούς.

Για παράδειγμα στο project υπάρχουν 256*256 έγχρωμες εικόνες. Η πρόβλεψη του κάθε καναλιού με 256 επιλογές σημαίνει ότι θα έχουμε για μια RGB εικόνα 256^3, άρα 16.777.216 συνδυασμούς χρωμάτων. Ενώ, εάν μετατρέψουμε τις εικόνες σε lab θα πρέπει να γίνει πρόβλεψη 2 καναλιών του a και b άρα θα έχουμε 256^2 χρωματικούς συνδυασμούς που σημαίνει 65.536 δυνατούς συνδυασμούς [12].

Από τα παραπάνω εξάγεται το συμπέρασμα ότι ο πιο απλός τρόπος να λυθεί ένα τέτοιο πρόβλημα είναι με τη χρήση LAB εικόνων. Αυτή κρίνεται ως η καλύτερη λύση, διότι το δίκτυο θα έχει να εκπαιδευτεί για λιγότερους χρωματικούς συνδυασμούς για κάθε μία εικόνα και εν τέλει θα είναι γρηγορότερη η γενίκευση που επιχειρείται ως τελικός στόχος (βλέπε εικόνα 3.9).



3.9 Σύγκριση rgb και lab εικόνας

Στη συνέχεια και αφού κάθε εικόνα έχει μία αληθινή και μία ασπρόμαυρη χρειάζεται η ασπρόμαυρη να περαστεί από το μοντέλο του generator ώστε να εξεταστεί ο τρόπος που τη χρωματίζει. Η τελική χρωματισμένη εικόνα που θα προκύψει θα έχει παρουσιαστεί σε ολόκληρο το νευρωνικό δίκτυο (GAN) και θα έχει εκπαιδευτεί σε όλο το σύνολο.

Παρακάτω φαίνεται η εικόνα του δίκτυο, το οποίο όμως δεν έχει εκπαιδευτεί πλήρως. Για να παραχθεί η εικόνα είναι απαραίτητη η συνάρτηση generate_fake_samples() η οποία κάνει predict την εικόνα από το μοντέλο. Η εικόνα αυτή μετασχηματίζεται (reshape) σε διαστάσεις 256*256 για να είναι ίδια με την αρχική. Η φωτογραφία είναι όπως αναμένεται, αφού το δίκτυο δεν έχει εκπαιδευτεί.

```
def generate_fake_samples(g_model, image):
    k = g_model.predict(image.reshape(1, 256, 256, 1))
    X = k.reshape(256, 256)
    # create 'fake' class labels (0)
    y = zeros((1, 1))
    return X, y
```

```
img_ = rgb2lab(dataset[1][ix])
img_[:, :, 0]=y_real
img_[:, :, 1]=fake
img_ = lab2rgb(img_)
pyplot.imshow(img_)
filename = 'Predicted_plot_e%03d.png' % (epoch+1)
pyplot.savefig(filename)
pyplot.close()
```

3.10 Generated image

Για την επιλογή του δικτύου δοκιμάστηκαν πολλά έτοιμα μοντέλα σε ένα μικρό αριθμό δεδομένων και επιλέχθηκε εκείνο με την καλύτερη χρωματιστική προσέγγιση. Τα επίπεδα που περιέχει το μοντέλο καθώς και οι έξοδοι που βγάζουν το καθένα από αυτά, αλλά και ο αριθμός των παραμέτρων που δέχονται παρουσιάζονται παρακάτω με την χρήση σχεδιαγράμματος καλώντας την .summary() συνάρτηση του keras.

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 128, 128, 64)	640
batch_normalization_1 (Batch	(None, 128, 128, 64)	256
leaky_re_lu_1 (LeakyReLU)	(None, 128, 128, 64)	0
conv2d_2 (Conv2D)	(None, 64, 64, 128)	73856
batch_normalization_2 (Batch	(None, 64, 64, 128)	512
leaky_re_lu_2 (LeakyReLU)	(None, 64, 64, 128)	0
conv2d_3 (Conv2D)	(None, 32, 32, 256)	295168
batch_normalization_3 (Batch	(None, 32, 32, 256)	1024
leaky_re_lu_3 (LeakyReLU)	(None, 32, 32, 256)	0
conv2d_4 (Conv2D)	(None, 16, 16, 512)	1180160
batch_normalization_4 (Batch	(None, 16, 16, 512)	2048
leaky_re_lu_4 (LeakyReLU)	(None, 16, 16, 512)	0
conv2d_transpose_1 (Conv2DTr	(None, 32, 32, 256)	1179904
batch_normalization_5 (Batch	(None, 32, 32, 256)	1024
activation_1 (Activation)	(None, 32, 32, 256)	0
conv2d_transpose_2 (Conv2DTr	(None, 64, 64, 128)	295040
batch_normalization_6 (Batch	(None, 64, 64, 128)	512
activation_2 (Activation)	(None, 64, 64, 128)	0
conv2d_transpose_3 (Conv2DTr	(None, 128, 128, 64)	73792
batch_normalization_7 (Batch	(None, 128, 128, 64)	256
activation_3 (Activation)	(None, 128, 128, 64)	0
conv2d_transpose_4 (Conv2DTr	(None, 256, 256, 32)	18464
batch_normalization_8 (Batch	(None, 256, 256, 32)	128
activation_4 (Activation)	(None, 256, 256, 32)	0
conv2d_5 (Conv2D)	(None, 256, 256, 2)	578
activation_5 (Activation)	(None, 256, 256, 2)	0

Total params: 3,123,362
Trainable params: 3,120,482
Non-trainable params: 2,880

3.11 Generator model

Το δίκτυο που εν τέλει επιλέχθηκε είναι ένα U-NET, συνελικτικό νευρωνικό, δίκτυο. Η κύρια ιδέα του είναι η συμπλήρωση ενός δικτύου με διαδοχικά επίπεδα, όπου οι λειτουργίες συγκέντρωσης αντικαθίστανται από upsampling. Μια σημαντική τροποποίηση είναι ότι υπάρχει ένας μεγάλος αριθμός χαρακτηριστικών καναλιών στο τμήμα upsampling, τα οποία επιτρέπουν στο δίκτυο να διαδίδει πληροφορίες περιβάλλοντος σε επίπεδα υψηλότερης ανάλυσης [13].

Υπάρχουν πολλές εφαρμογές του **U-Net,** ενδεικτικά:

- Βιοϊατρική
- Σε 3D εκμάθηση πυκνού ογκομετρικού κατακερματισμού
- Μετάφραση εικόνας για εκτίμηση φθορισμού

Ανάλυση των επιπέδων (Layers)

- **Conv2d**

Αυτό το στρώμα (Layer) δημιουργεί έναν πυρήνα συνέλιξης που συστρέφεται με την είσοδο του στρώματος για να παράγει έναν τανυστή εξόδων. Εάν το use_bias είναι Αληθής (True), δημιουργείται ένα διάνυσμα πόλωσης και προστίθεται στις εξόδους. Τέλος, εάν η ενεργοποίηση (activation) δεν είναι none, εφαρμόζεται και στις εξόδους**.**

- **Leaky_relu**

Είναι μια έκδοση διορθωμένης γραμμικής μονάδας που επιτρέπει μια μικρή κλίση όταν η alpha παράμετρος δεν είναι ενεργοποιημένη.

- **Batch Normalization**

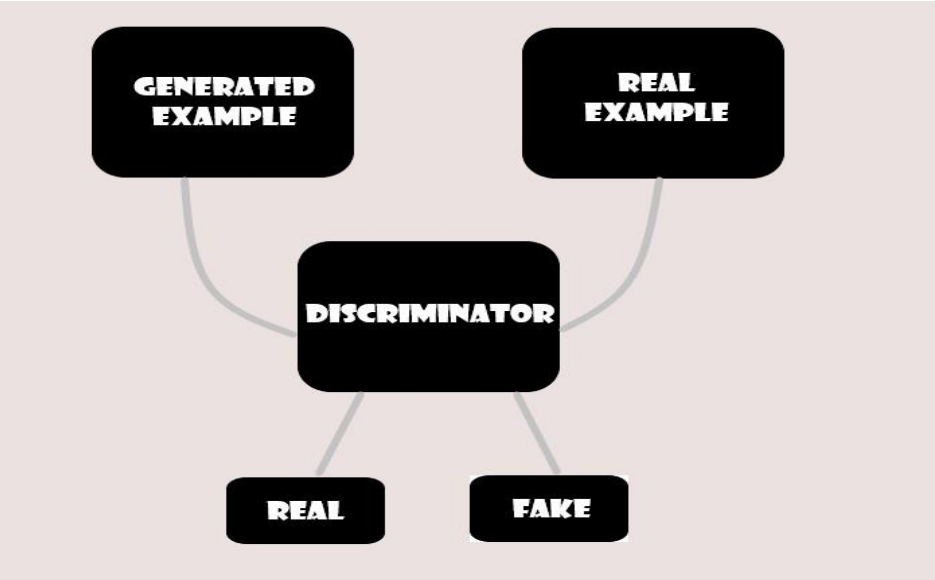
Είναι ένα στρώμα που ομαλοποιεί τις εισόδους. Εφαρμόζει έναν μετασχηματισμό που διατηρεί τη μέση έξοδο κοντά στο 0 και την τυπική απόκλιση εξόδου κοντά στο 1. Είναι σημαντικό ότι λειτουργεί διαφορετικά κατά τη διάρκεια της προπόνησης και κατά τη διάρκεια των συμπερασμάτων. Όσο διαρκεί η εκπαίδευση το επίπεδο ομαλοποιεί την έξοδό του χρησιμοποιώντας τη μέση και την τυπική απόκλιση της τρέχουσας παρτίδας εισόδων. Κατά την εξαγωγή συμπερασμάτων το επίπεδο ομαλοποιεί την έξοδό

του χρησιμοποιώντας έναν κινητό μέσο όρο της μέσης και τυπικής απόκλισης των παρτίδων που έχει δει κατά τη διάρκεια της προπόνησης.

3.6 Δημιουργία και εκπαίδευση Discriminator

Το μοντέλο του discriminator χρησιμοποιείται ώστε να υπάρχει διάκριση ανάμεσα στις πραγματικές (real) και τις ψεύτικες (fake) εικόνες. Το μοντέλο μπορεί να χρησιμοποιήσει οποιαδήποτε αρχιτεκτονική δικτύου κατάλληλη για τον τύπο δεδομένων που ταξινομεί. Τα πραγματικά δεδομένα που καταλήγουν στον discriminator είναι πραγματικές έγχρωμες εικόνες από την βάση δεδομένων (database) που κατασκευάστηκε προηγουμένως, ενώ τα ψεύτικα είναι οι δημιουργημένες εικόνες που παράγει ο generator.

Ο διακριτής συνδέεται με δύο συναρτήσεις απώλειας μια για το discriminator και μια για το generator. Κατά τη διάρκεια της εκπαίδευσης για διακρίσεις, ο διακριτής αγνοεί την απώλεια της γεννήτριας και απλώς χρησιμοποιεί την απώλεια διάκρισης. Η απώλεια της γεννήτριας κατά την εκπαίδευση του μοντέλου χρησιμοποιείται ώστε να εκπαιδευτεί η γεννήτρια. Το σφάλμα μεταφέρεται στην γεννήτρια μέσω του backpropagation (επιστροφή του σφάλματος κατά την πρόσθια τροφοδότηση) και επηρεάζει τα βάρη κατά τέτοιο τρόπο, ούτως ώστε να εκπαιδευτεί σωστά η γεννήτρια και να παράγει τα επιθυμητά αποτελέσματα (βλέπε εικόνα 3.12).



3.12 Discriminator example

Καθόλη τη διάρκεια της εκπαίδευσης:

- Ο διακριτής ταξινομεί τόσο τα πραγματικά δεδομένα όσο και τα πλαστά δεδομένα από τη γεννήτρια.
- Η απώλεια διακρίσεων τιμωρεί τον διακριτή για εσφαλμένη ταξινόμηση του πραγματικού περιστατικού ως πλαστού ή του ψεύτικου περιστατικού ως πραγματικού.
- Ο διακριτής ενημερώνει τα βάρη του μέσω της αντίστροφης διάδοσης από την απώλεια διακρίσεων μέσω του δικτύου διακρίσεων.

Εκπαίδευση Discriminator

Για την εκπαίδευσή του χρησιμοποιείται η συνάρτηση train(), στην οποία σε μια μεταβλητή μέσω τις load_real_samples() φορτώνονται οι εικόνες του dataset που θα χρησιμοποιηθούν για αληθινές (real) εικόνες. Σε μια άλλη μεταβλητή και μέσω της generate_fake_samples() αποθηκεύονται οι παραγόμενες εικόνες. Εν συνεχεία, καλείται η συνάρτηση train_on_batch() του Keras, ώστε να υπολογισθεί το ποσοστό ακρίβειας για τις αληθινές και τις ψεύτικες εικόνες.

```
def train_discriminator(model, dataset, n_iter=20, n_batch=128):
    half_batch = int(n_batch / 2)
    # manually enumerate epochs
    for i in range(n_iter):
        # get randomly selected 'real' samples
        X_real, y_real = select_real_samples(dataset, half_batch)
        print("x_real=",X_real.shape)
        print("y_real=",y_real.shape)
        # update discriminator on real samples
        _, real_acc = model.train_on_batch(X_real, y_real)
        # generate 'fake' examples
        X_fake, y_fake = generate_fake_samples(half_batch)
        # update discriminator on fake samples
        _, fake_acc = model.train_on_batch(X_fake, y_fake)
        # summarize performance
        print('>%d real=%.0f%% fake=%.0f%%' %
              (i+1, real_acc*100, fake_acc*100))
```

3.13 Def train Discriminator

Όσον αφορά την εκπαίδευση του μοντέλου διακρίσεων χρησιμοποιείται η train_on_batch(), και όχι η fit(), διότι επιτρέπει τη ρητή ενημέρωση των βαρών του μοντέλου, βάση μιας συλλογής δειγμάτων που παρέχεται, χωρίς να λαμβάνεται υπόψη το μέγεθος της παρτίδας κάθε εποχής.

Μοντέλο Discriminator

Layer (type)	Output Shape	Param #
conv2d_11 (Conv2D)	(None, 256, 256, 16)	160
conv2d_12 (Conv2D)	(None, 256, 256, 32)	4640
average_pooling2d_1 (Average	(None, 128, 128, 32)	0
dropout_1 (Dropout)	(None, 128, 128, 32)	0
conv2d_13 (Conv2D)	(None, 128, 128, 64)	18496
conv2d_14 (Conv2D)	(None, 128, 128, 64)	36928
average_pooling2d_2 (Average	(None, 64, 64, 64)	0
dropout_2 (Dropout)	(None, 64, 64, 64)	0
flatten_1 (Flatten)	(None, 262144)	0
dense_1 (Dense)	(None, 512)	134218240
leaky_re_lu_10 (LeakyReLU)	(None, 512)	0
batch_normalization_7 (Batch	(None, 512)	2048
dropout_3 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 1)	513
activation_1 (Activation)	(None, 1)	0

Total params: 134,281,025
Trainable params: 134,280,001
Non-trainable params: 1,024

3.14 Discriminator model

● **Average_pooling**

Χαμηλοποιεί την είσοδο κατά μήκος των χωρικών διαστάσεών της (ύψος και πλάτος), λαμβάνοντας παράλληλα τη μέση τιμή πάνω από ένα παράθυρο εισόδου (μεγέθους που καθορίζεται από την μεταβλητή pool_size) για κάθε κανάλι της εισόδου. Το παράθυρο μετατοπίζεται κατά μήκος strides κάθε διάστασης.

● **Dropout**

Το στρώμα Dropout ορίζει τυχαία τις μονάδες εισόδου στο 0 με συχνότητα rate σε κάθε βήμα κατά τη διάρκεια της προπόνησης, κάτι που βοηθά στην αποφυγή υπερβολικής προσαρμογής. Οι είσοδοι που δεν έχουν οριστεί σε 0 κλιμακώνονται κατά 1/(1 - ποσοστό) έτσι ώστε το άθροισμα όλων των εισόδων να παραμένει αμετάβλητο.

● **Flatten**

Ισοπεδώνει την είσοδο. Δεν επηρεάζει το batch_size.

● **Dense**

Σε αυτό το στρώμα υλοποιείται η λειτουργία output= activation(dot(input,kernel)+bias) οπού, activation είναι η συνάρτηση ενεργοποίησης ως προς το στοιχείο που περνά σαν είσοδο (input). Kernel είναι ένας πίνακας βαρών που δημιουργείται από το επίπεδο και bias είναι ένα διάνυσμα πόλωσης που δημιουργείται από το επίπεδο και ισχύει μόνο εάν το “use_bias=True”.

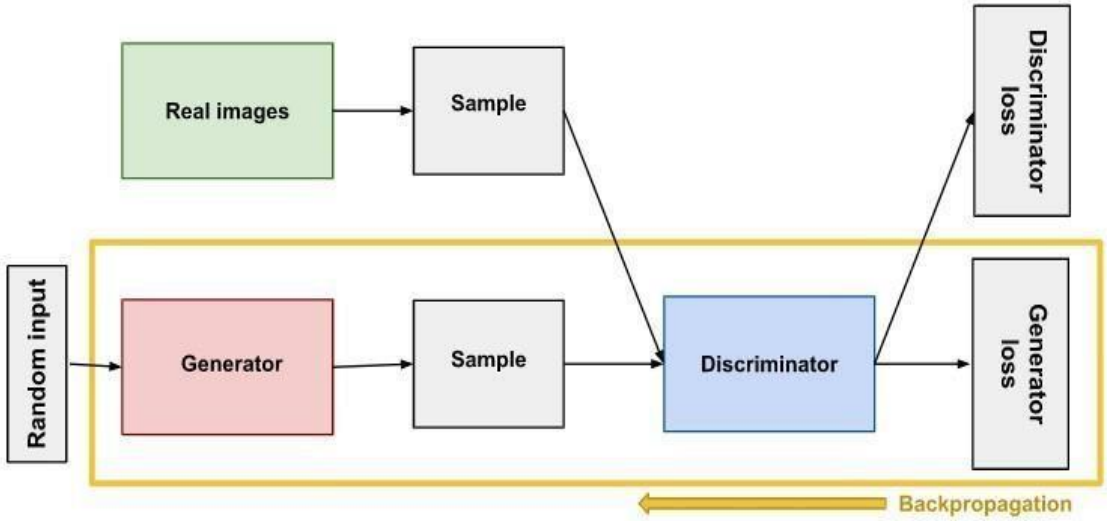
● **Activation**

Εφαρμόζει μια συνάρτηση ενεργοποίησης σε μια έξοδο.

3.7 Κατασκευή του Νευρωνικού δικτύου (GAN)

Η λογική στην κατασκευή του νευρωνικού δικτύου GAN είναι η συνένωση των παραπάνω μοντέλων που παρουσιάστηκαν, της γεννήτριας και του διακριτή. Τελικός σκοπός, όπως έχει πολλάκις τονιστεί είναι η εκπαίδευση του δικτύου, προκειμένου αυτό να επιτελεί την λειτουργία του χρωματισμού ασπρόμαυρων εικόνων. Η γεννήτρια

μαθαίνει να δημιουργεί εύλογα δεδομένα. Οι παραγόμενες εικόνες γίνονται αρνητικά παραδείγματα κατάρτισης για τους διακριτές. Ενώ ο διακριτής μαθαίνει να διακρίνει τα ψεύτικα δεδομένα της γεννήτριας από πραγματικά δεδομένα και τιμωρεί τη γεννήτρια για την παραγωγή λανθασμένων αποτελεσμάτων. Παρακάτω παρουσιάζεται η ολοκληρωμένη αρχιτεκτονική του προγράμματος (δηλαδή ενώ έχουν ενωθεί τα δυο παραπάνω σχεδιαγράμματα που παρουσιάστηκαν).



3.15 GAN model example

Για την δημιουργία του προγράμματος του GAN αρχικά φορτώνεται το dataset με τον τρόπο που παρουσιάστηκε στην ενότητα «Δημιουργία και εκπαίδευση Generator».

```
g_model = define_generator()
g_model.summary()
d_model = define_discriminator()
d_model.summary()
gan_model=define_gan(g_model, d_model)
```

3.16 συνενωση δικτυων

Έπειτα, ορίζονται τα 2 δίκτυα που θα χρησιμοποιηθούν παρακάτω, της γεννήτριας και του διακριτή, για να κατασκευαστεί το δίκτυο. Ο ορισμός και η αρχιτεκτονική της γεννήτριας και του διακριτή ήδη έχουν παρουσιαστεί προηγουμένως. Για την ένωση των 2 δικτύων κατασκευάστηκε μια συνάρτηση define_gan(g_model,d_model), η οποία παίρνει σαν ορίσματα τα δυο δίκτυα.

Αμέσως μετά ορίστηκε ένα καινούργιο δίκτυο μέσω της Sequential() συνάρτηση του Keras και μέσω της add() προστίθενται τα δυο δίκτυα g_model και d_model. Για optimizer χρησιμοποιείται ένας αλγόριθμο βελτιστοποίησης Adam που βασίζεται στην προσαρμοστική εκτίμηση ροπής πρώτης και δεύτερης τάξης. Διατηρεί ένα ενιαίο ποσοστό εκμάθησης (που ονομάζεται άλφα) για όλες τις ενημερώσεις βάρους και το εν λόγω ποσοστό εκμάθησης δεν αλλάζει κατά τη διάρκεια της προπόνησης.

Στην συνέχεια φορτώνονται σε έναν πίνακα dataset όλες οι εικόνες για την εκπαίδευση του δικτύου και για το τεστάρισμα αυτού. Η βασική συνάρτηση που καλείται

```
# define the combined generator and discriminator model, for updating the generator
def define_gan(g_model, d_model):
    # make weights in the discriminator not trainable
    #d_model.trainable = False
    # connect them
    model = Sequential()
    # add generator
    model.add(g_model)
    # add the discriminator
    model.add(d_model)
    # compile model
    opt = Adam(lr=0.0002, beta_1=0.5)
    model.compile(loss='binary_crossentropy', optimizer=opt)
    return model
```

3.17 Συνάρτηση συνένωσης δικτύων define_gan

για να εκτελεστεί το πρόγραμμα είναι η συνάρτηση train(). Η συνάρτηση αυτή παίρνει σαν όρισμα το τελικό δίκτυο, καθώς και ξεχωριστά τη γεννήτρια και το διακριτή, το dataset με τις εικόνες, τον αριθμό των εποχών και των αριθμό των εικόνων που θα χρησιμοποιούνται για την εκπαίδευση του δικτύου.

```
def train(g_model, d_model, gan_model, dataset, n_epochs=1, n_batch=1):
    #bat_per_epo = int(dataset[1].shape[0] / n_batch)
    # manually enumerate epochs
    for i in range(n_epochs):
        # enumerate batches over the training set
        for j in range(n_batch):
            gray=[]
            # get randomly selected 'real' samples
            X_real, y_real = select_real_samples(dataset,n_batch,gray)
            d_loss1, _ = d_model.train_on_batch(X_real, y_real)
            # generate 'fake' examples
            X_fake, y_fake = generate_fake_samples(g_model, gray[0])
            # update discriminator model weights
            d_loss2, _ = d_model.train_on_batch(X_fake.reshape(1,256,256,1), y_fake)
            # create inverted labels for the fake samples
            y_gan = ones((1, 1))
            # update the generator via the discriminator's error
            g_loss = gan_model.train_on_batch(X_fake.reshape(1,256,256,1), y_gan)
            # summarize loss on this batch
            print('>%d, %d/%d, d_real_loss=%.3f, d_fake_loss=%.3f gan_loss=%.3f' %
                (i+1, j+1, n_batch, d_loss1, d_loss2, g_loss))
        # evaluate the model performance, sometimes
        if (i+1) % 1 == 0:
            summarize_performance(i, g_model, d_model, dataset)
```

3.18 Συνάρτηση για την εκπαίδευση των δικτύων

Μέσα σε αυτήν για κάθε εικόνα από το σύνολο εκπαίδευσης, κάθε εποχής, καλείται η select_real_samples() και επιστρέφει την αντίστοιχη πραγματική έγχρωμη εικόνα από RGB σε LAB, την ίδια στιγμή που αποθηκεύει την αντίστοιχη ασπρόμαυρη σε έναν πίνακα gray. Επιπροσθέτως, υπολογίζεται η απώλεια του discriminator για τις real εικόνες.

Σε επόμενο βήμα καλείται η generate_fake_samples() ώστε η γεννήτρια να χρωματίσει και να επιστρέψει την παραγόμενη ψευδή εικόνα. Αντίστοιχα και εδώ υπολογίζεται η απώλεια του διακριτή για τις ψευδές εικόνες. Για τις ψεύτικες εικόνες δημιουργούνται και ανεστραμμένες ετικέτες (tags). Καταληκτικά, υπολογίζεται η απώλεια για όλο το δίκτυο GAN, και κάθε 5 εποχές καλείται την συνάρτηση summarize_performance(), με απώτερο σκοπό να ελεγχθεί η απόδοση του δικτύου, αλλά και ο τρόπος που χρωματίζει μετά το πέρας κάποιων εποχών.

```
def summarize_performance(epoch, g_model, d_model, dataset):
    # prepare real samples
    gray = []
    ix = randint(0, dataset[1].shape[0])
    X_real, y_real = select_real_samples(dataset,ix,gray)
    # evaluate discriminator on real examples
    _, acc_real = d_model.evaluate(X_real, y_real, verbose=0)
    # prepare fake examples
    x_fake, y_fake = generate_fake_samples(g_model, gray[0])
    # evaluate discriminator on fake examples
    _, acc_fake = d_model.evaluate(x_fake.reshape(1,256,256,1), y_fake, verbose=0)
    # summarize discriminator performance
    print('>Accuracy real: %.0f%%, fake: %.0f%%' % (acc_real*100, acc_fake*100))
    # save plot
    save_plot(gray[0],x_fake.reshape(256,256),y_real,ix,epoch)
    # save the generator model tile file
    filename = 'generator_model_%03d.h5' % (epoch+1)
    g_model.save(filename)
```

3.19 Συνάρτηση summarize_perfomance()

```
def save_plot(real,fake, y_real,ix,epoch):
    print("Mphka sthn save plot")
    pyplot.imshow(dataset[1][ix])
    filename = 'Real_plot_e%03d.png' % (epoch+1)
    pyplot.savefig(filename)
    pyplot.close()

    pyplot.imshow(real,cmap="gray")
    filename = 'Gray_plot_e%03d.png' % (epoch+1)
    pyplot.savefig(filename)
    pyplot.close()

    #fake = (fake + [0, 128, 128]) * [100, 255, 255]
    img_ = rgb2lab(dataset[1][ix])
    img_[:, :, 0]=y_real
    img_[:, :, 1]=fake
    pyplot.imshow(img_)
    filename = 'Predicted_lab_plot_e%03d.png' % (epoch+1)
    pyplot.savefig(filename)
    pyplot.close()

    img_ = lab2rgb(img_)
    pyplot.imshow(img_)
    filename = 'Predicted_rgb_plot_e%03d.png' % (epoch+1)
    pyplot.savefig(filename)
    pyplot.close()
```

3.20 συνάρτηση save_plot()

Το βήμα που έπεται αφορά την τυχαία επιλογή ενός αριθμού ανάμεσα στο μηδέν και το μέγεθος του συνολου επικύρωσης, ώστε να επιλεχθεί μια τυχαία εικόνα για να χρωματιστεί. Αμέσως μετά υπολογίζεται η ακρίβεια στη διαπίστωση αληθινών και ψεύτικων εικόνων από το διακριτή και καλείται η συνάρτηση save_plot(). Μέσα σε αυτήν αποθηκεύεται η πραγματική έγχρωμη εικόνα που έχει επιλεχθεί από την αντίστοιχη ασπρόμαυρη που επιστρέφει η

select_real_samples(). Σε αυτό το σημείο στο έγχρωμο μέρος της εικόνας εισάγεται η ψεύτικη εικόνα που παρήγαγε η γεννήτρια, και αυτή είναι η τελική παραγόμενη τύπου LAB εικόνα. Μέσω της lab2rgb την εικόνα μετατρέπεται σε τύπου RGB εικόνα και μετά

αποθηκεύονται και οι δυο, με στόχο να συγκριθούν τα αποτελέσματα. Τελικά, το δίκτυο αποθηκεύεται σε ένα αρχείο .h5 για να χρησιμοποιηθεί αργότερα για κάθε καρέ του βίντεο.

Κεφάλαιο 4

Αξιολόγηση τελικού μοντέλου και παρουσίαση αποτελεσμάτων

Στο κεφάλαιο αυτό θα παρουσιαστούν τα αποτελέσματα του τελικού προγράμματος. Αναμένεται από το πρόγραμμα να λειτουργήσει ομαλά και να έχει αξιοπρεπή αποτελέσματα στα παραδείγματα που θα εκτελεστούν. Κατά τη διάρκεια συγγραφής της παρούσας εργασίας υπήρξαν κάποια προβλήματα που προέκυψαν, χωρίς να είναι εφικτό να επιλυθούν και επηρέασαν σημαντικά το τελικό αποτέλεσμα του προγράμματος και την ικανότητα χρωματισμού της εικόνας. Αυτά, λοιπόν, αναλύονται παρακάτω και προτείνονται ταυτόχρονα ενδεχόμενες λύσεις.

4.1 Παρουσίαση και αξιολόγηση της απόδοσης του μοντέλου

Παρακάτω παρουσιάζονται μερικά παραδείγματα από τα αποτελέσματα σε εικόνες του συνόλου επικύρωσης:

4.1 Παράδειγμα 1 χρωματισμού ασπρόμαυρης εικόνας

4.2 Παράδειγμα 2 χρωματισμού ασπρόμαυρης εικόνας

4.3 Παράδειγμα 3 χρωματισμού ασπρόμαυρης εικόνας

4.4 Παράδειγμα 4 χρωματισμού ασπρόμαυρης εικόνας

4.5 Παράδειγμα 5 χρωματισμού ασπρόμαυρης εικόνας

Όπως φαίνεται στις άνωθεν εικόνες τα αποτελέσματα μετά την εκπαίδευση του δικτύου σε εικόνες του συνόλου επικύρωσης είναι ικανοποιητικά και αρκετά υποσχόμενα για την απόδοση του προγράμματος σε εξολοκλήρου καινούριες φωτογραφίες. Επόμενο βήμα, λοιπόν, είναι η χρήση του προγράμματος για το χρωματισμό των frames από τα βίντεο.

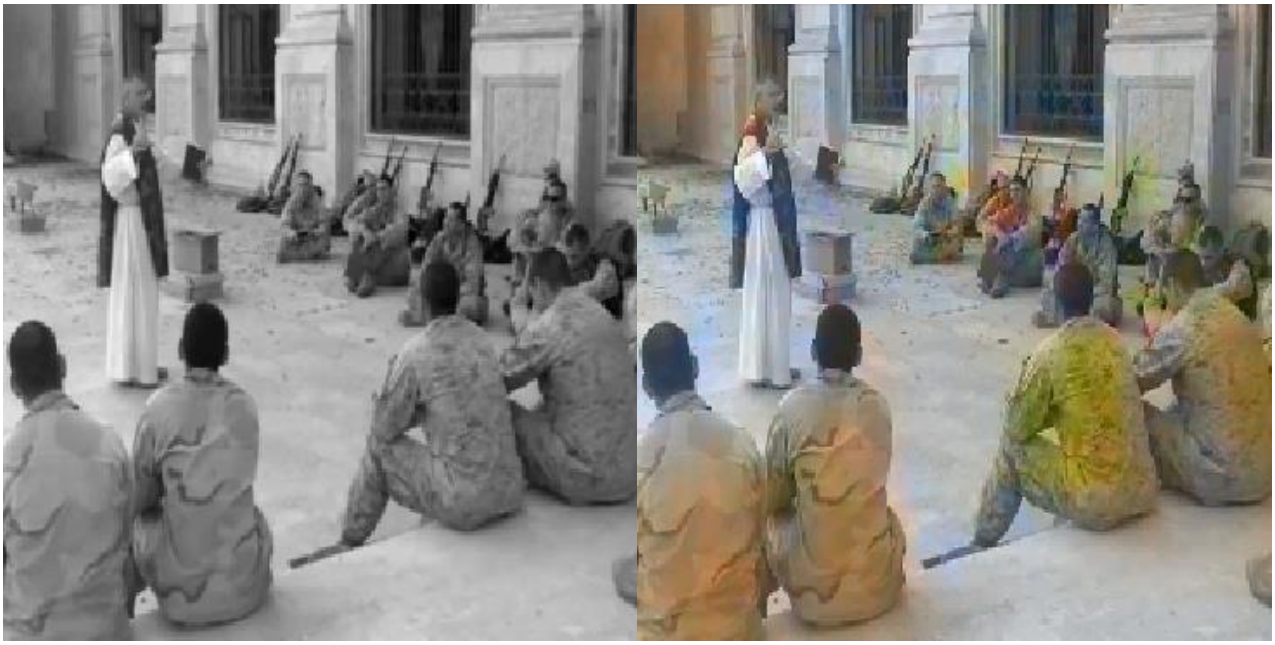
Αρχικά, το βίντεο χωρίζεται σε frames. Κάθε ένα από αυτά τα περνάνε στο πρόγραμμα ως είσοδος και τελικά δίνεται η τελική εικόνα. Με τον τρόπο αυτόν θα διασφαλιστεί ότι το λογισμικό δουλεύει σωστά και το δίκτυο κάνει ορθές προβλέψεις για παραδείγματα εκτός των συνόλων που χρησιμοποιήθηκαν κατά την εκπαιδευση. Εφόσον το οπτικό αποτέλεσμα είναι ικανοποιητικό, όπως ήταν και με τις προηγούμενες εικόνες ξεκινά ο έλεγχος του προγράμματος σε ένα ολόκληρο βίντεο.

Παρακάτω παρουσιάζονται τα αποτελέσματα σε τυχαία frames από ένα συγκεκριμένο βίντεο:

4.6 Παράδειγμα 6 χρωματισμού ασπρόμαυρης εικόνας

4.7 Παράδειγμα 7 χρωματισμού ασπρόμαυρης εικόνας

4.8 Παράδειγμα 8 χρωματισμού ασπρόμαυρης εικόνας



4.9 Παράδειγμα 9 χρωματισμού ασπρόμαυρης εικόνα

Από τα ανωτέρω αποτελέσματα φαίνεται ότι οι προβλέψεις του μοντέλου είναι εξίσου ικανοποιητικές σε τυχαία frames από ένα βίντεο. Αξίζει επίσης να επισημανθεί ότι σε πολλές φωτογραφίες όπου υπάρχουν δέντρα και τα φυτά φαίνεται να παίρνουν έναν εύλογο χρωματισμό, δηλαδή είτε να είναι πράσινα, είτε κίτρινα, είτε να έχουν μια κοντινή απόχρωση αυτών των δυο.

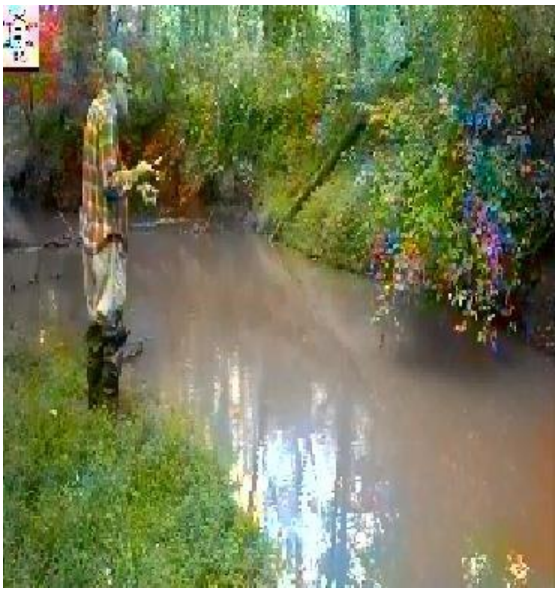
Σε άλλες περιπτώσεις παρατηρήθηκε, επίσης, πολύ ικανοποιητικός χρωματισμός σε δάπεδα και σε πατώματα. Χρωματίζοντάς τα ενίοτε ως ξύλινα, άλλες φορές ως τσιμεντένια. Παρομοίως ρεαλιστικά ήταν και τα αποτελέσματα του μοντέλου σε φωτογραφίες με χιόνι.

Τέλος, σαν σύνολο οι εικόνες θα μπορούσαν να χαρακτηριστούν ρεαλιστικές, με χρώματα λογικά, αναμενόμενα και με συνέπεια, κάνοντας μερικές εξ' αυτών να μπορούν να σταθούν μόνες τους. Ωστόσο, οι ατέλειες που προκύπτουν δεν είναι δυνατόν να αγνοηθούν, αφού ενδέχεται να επηρεάσουν αρνητικά και να αλλοιώσουν το τελικό αποτέλεσμα. Αυτά όμως τα ζητήματα θα αναλυθούν σε επόμενη ενότητα.

Λίγο πριν ολοκληρωθεί αυτό το υποκεφάλαιο είναι απαραίτητο να πραγματοποιηθεί και μια τελευταία σύγκριση ανάμεσα στις αληθινές εικόνες με τις παραγόμενες εικόνες του δικτύου, διότι με αυτό τον τρόπο θα εξαχθεί εν τέλει μία ολοκληρωμένη άποψη.

4.10 Έγχρωμη εικόνα για σύγκριση

4.11 Ασπρόμαυρη εικόνα



4.12 Παραγόμενη εικόνα για σύγκριση

Όπως έχει τονισθεί σε αρκετά σημεία της διπλωματικής, ο επιχρωματισμός μιας ασπρόμαυρης εικόνας είναι μια υποκειμενική διαδικασία, δηλαδή ένα πουκάμισο (όπως στην προκειμένη περίπτωση), μπορεί να σταθεί σε μια εικόνα με οποιοδήποτε χρώμα, χωρίς να αλλάζει κάτι αν αυτό είναι μαύρο ή άσπρο ή κίτρινο. Οπότε, δεν θα γινόταν να θεωρηθεί χρωματιστικό λάθος ή να "κατηγορήσουμε" το δίκτυο για το γεγονός ότι το πουκάμισο στην παραπάνω εικόνα έχει κίτρινο χρώμα.

Υπάρχουν, όμως και περιπτώσεις με αντικείμενα που δεν είναι τόσο αληθοφανή ή ελκυστικά στο ανθρώπινο μάτι να τα αντιληφθεί με διαφορετικά χρώματα, από εκείνα που τα έχει συνηθίσει στην καθημερινή ζωή. Για παράδειγμα το γρασίδι είναι συνηθισμένο να εμφανίζεται στην φύση με πράσινο χρώμα και μερικές φορές με χρυσαφένιο ή καφέ, όταν αλλάζει η εποχή (φθινόπωρο). Εάν όμως εισαχθεί ένα βίντεο στο οποίο το γρασίδι θα απεικονίζεται με μπλε χρώμα, δεν θα ήταν λογικό να αναμένεται από το δίκτυο να δώσει το ίδιο χρώμα, από τη στιγμή που έχει εκπαιδευτεί να λαμβάνει εικόνες με πράσινο γρασίδι. Μάλιστα θα ήταν σφάλμα από μεριάς του χειριστή/ερευνητή να θεωρήσει πως κάτι τέτοιο ελλοχεύει λάθη, αστοχίες και παραλείψεις με την εκπαίδευση του.

Στην παραπάνω εικόνα παρατηρούμε ότι το γρασίδι έχει μια αρμονία με την πραγματικότητα, μιας κι έχει χρωματιστεί ολόκληρο με το σωστό, πράσινο χρώμα, το οποίο μάλιστα σταματάει τη σωστή στιγμή, όταν συναντάει το ποτάμι. Το ποτάμι από την άλλη αναμένεται να έχει ένα γαλάζιο χρώμα, το οποίο δεν εμφανίζεται στη νέα φωτογραφία που προκύπτει. Αυτό αποτελεί ένα λάθος του δικτύου; Όχι, γιατί όπως προαναφέρθηκε από τη μία ο χρωματισμός είναι υποκειμενικός, και από την άλλη στην πραγματική ζωή σε ένα ποτάμι μπορεί να καταλήξουν λάσπες (και να πάρει ένα καφέ χρώμα), βρύα, φύλλα και κλαδιά (και να έχει πράσινο χρώμα), γεγονός που θα επηρεάσει το τελικό του χρώμα. Στο παραπάνω παράδειγμα της παραγόμενης εικόνας το ποτάμι έχει ένα καφέ χρώμα, όμοιο με αυτό που αναμένεται να έχει ένα οποιοδήποτε ποτάμι μετά από μία έντονη βροχή, οπότε το χώμα γίνεται λάσπη.

4.13 Σύγκριση καθαρού ποταμού με λασπωμένο ποτάμι

4.2 Προβλήματα προγράμματος

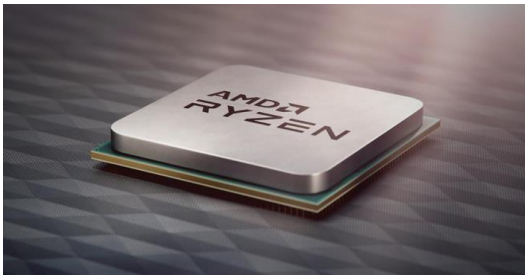
Η κατασκευή του προγράμματος δεν ήταν απόλυτα επιτυχής κι αυτό διότι εμφανίστηκαν αρκετά προβλήματα κατά την εκπαίδευσή του, κάποια εκ των οποίων επηρέασαν την απόδοση του μοντέλου και άλλα τον χρόνο εκπαίδευσης αυτού.

Ένα από τα βασικότερα προβλήματα που δυσκόλεψε στην εκπαίδευση του μοντέλου και ταυτόχρονα επηρέασε σημαντικά το χρόνο εκπαίδευσης -και κατ' επέκταση την απόδοση του δικτύου- ήταν η έλλειψη του κατάλληλου εξοπλισμού, δηλαδή ενός καλού συστήματος υπολογιστή που συμπεριλαμβάνει κάρτα γραφικών με μεγάλη μνήμη.

Ειδικότερα, η εκπαίδευση ενός νευρωνικού δικτύου αποτελεί μια πολύ απαιτητική διαδικασία από τη στιγμή που απαιτεί το πέρασμα κάθε φωτογραφίας από εκατομμύρια νευρώνες, στου οποίους πραγματοποιούνται διάφοροι υπολογισμοί. Για να στεφθεί αυτή με επιτυχία χρειάζεται να υπάρχει ένα πολύ καλό σύστημα υπολογιστή που συμπεριλαμβάνει κάρτα γραφικών με μεγάλη μνήμη., ώστε να τρέχει και να δοκιμάζεται κάθε δίκτυο που κατασκευάζεται σε ένα μεγάλο σύνολο εικόνων και σε μεγάλο αριθμό εποχών.

Κατά τη συγγραφή της παρούσας διπλωματικής και την κατασκευή του μοντέλου ήταν διαθέσιμος ένας σταθερός υπολογιστής, χωρίς κάρτα γραφικών, αλλά με έναν επεξεργαστή που διέθετε ενσωματωμένα γραφικά και η μνήμη ram μόλις 1GB.

VS



4.14 Σύγκριση κάρτας γραφικών με ενσωματωμένα γραφικά σε επεξεργαστή

Οι διαθέσιμες εναλλακτικές που υπήρχαν ήταν να τρέξει και να εκπαιδευτεί το δίκτυο μέσω του Colaboratory. Το Colaboratory αποτελεί ένα προϊόν της Google που επιτρέπει σε οποιονδήποτε χρήστη να γράψει και να εκτελέσει αυθαίρετο κώδικα python μέσω του προγράμματος περιήγησης. Θεωρείται ιδιαίτερα κατάλληλο για μηχανική μάθηση, ανάλυση δεδομένων, τεχνητή νοημοσύνη για την εκπαίδευση των δικτύων τους. Πιο συγκεκριμένα το Colab είναι μια φιλοξενούμενη υπηρεσία φορητών υπολογιστών Jupyter που δεν απαιτεί εγκατάσταση περιβάλλοντος για να πραγματοποιηθεί η χρήση, καθώς αυτή μπορεί να γίνει ανά πάσα στιγμή online από το internet, ενώ παρέχει δωρεάν πρόσβαση σε υπολογιστικούς πόρους συμπεριλαμβανομένων και των GPU.

Το μειονέκτημα που παρουσιάζει το Colab είναι το χρονικό όριο χρήσης της δωρεάν έκδοσης που παρέχει σε κάθε χρήστη, καθώς δεν γίνεται κάθε ένας χρήστης του προγράμματος να έχει απεριόριστο χρόνο όταν χρησιμοποιεί CPU και GPU υπολογιστές, αφού κάτι τέτοιο θα εμπόδιζε τη δοκιμή από άλλους χρήστες των δικών τους προγραμμάτων τους. Συνεπώς, η Google εγγυάται ότι ο χρόνος που θα εκτελεί ο κάθε ένας το πρόγραμμά του θα έχει μέγιστο όριο της 12 ώρες, χωρίς όμως να υπάρχει εγγύηση ότι θα τις φτάσει.

Το πρόγραμμα Colab ήταν ένα από τα βασικά εργαλεία που χρησιμοποιήθηκαν για τη δεδομένη μελέτη. Κάθε φορά, όμως, που γινόταν χρήση αυτού η σύνδεση στο server διακοπτόταν σχεδόν στα μισά της εκπαίδευσης με αποτέλεσμα να μην είναι εφικτό να ληφθεί το τελικό αρχείο με το εκπαιδευμένο δίκτυο.

Μια δεύτερη δυνατή επιλογή ήταν αυτή που μας προσφέρεται από τη σχολή στα πλαίσια της οποίας γίνεται η μελέτη. Υπήρχε, λοιπόν, η δυνατότητα να εκπαιδευτεί το δίκτυο σε ειδικό υπολογιστή που καλύπτει πλήρως όλες τις απαιτήσεις. Το πρόβλημα που αναδύθηκε σε αυτή την περίπτωση ήταν η αδυναμία λήψης συγκεκριμένων εκδόσεων βιβλιοθηκών που χρησιμοποιήθηκαν στο διαθέσιμο (προσωπικό) υπολογιστή, με αποτέλεσμα να μην μπορεί να τρέξει το πρόγραμμα ώστε να εκπαιδεύσουμε το δίκτυο.

Λόγω των παραπάνω ήταν επιτακτική η ανάγκη να μειωθεί ο χρόνος της εκπαίδευσης του δικτύου. Οπότε σε αυτή τη βάση, χρησιμοποιήθηκαν όσο το δυνατόν μικρότερα δίκτυα, ώστε να μειωθούν οι νευρώνες και κατά συνέπεια οι υπολογισμοί που απαιτούνται. Επίσης, χρησιμοποιήθηκαν εικόνες διαστάσεων 128*128, αντί για τις αρχικές διαστάσεις 256*256. Αυτή και μόνο η μετατροπή βοήθησε στην εξοικονόμηση επιπλέον χρόνου, αφού για τις εικόνες των 256 διαστάσεων έχουμε 65.536 pixel ενώ για τις 128 διαστάσεις έχουμε 16.384 pixel. Παρά ταύτα αυτή η κίνηση θυσίασε την ποιότητα των παραγόμενων εικόνων που ήταν σαφώς χαμηλότερη. Ωστόσο, το δίλημμα της επιλογής άξιζε τον κόπο μιας και εν τέλει επιτεύχθηκε η επιθυμητή μείωση του χρόνου που είχε -στην προκειμένη περιπτωση- ζωτική σημασία για την ολοκλήρωση του εγχειρήματος.

4.3 Προτάσεις Βελτίωσης

Για να ξεπεραστεί το πρόβλημα του μεγάλου χρόνου εκπαίδευσης που χρειάζεται το δίκτυο ώστε να εκπαιδευτεί σωστά, πρέπει να τρέχει το δίκτυο σε έναν υπολογιστή με μια “δυνατή” κάρτα γραφικών και με μεγάλη χωρητικότητα μνήμης ram. Με αυτόν τον τρόπο θα επιτευχθεί η κατάλληλη εκπαίδευση του δικτύου σε μεγαλύτερο σύνολο εκπαίδευσης από εκείνο της παρούσας μελέτης, καθώς επίσης και σε περισσότερες εποχές σε πολύ λιγότερο χρόνο από αυτόν που χρειάστηκε.

Μια τελευταία σημαντική πρόταση που θα επηρέαζε το αποτέλεσμα σε καθοριστικό βαθμό είναι η εύρεση ενός πιο μικρού και πιο αποδοτικού δικτύου τόσο για την γεννήτρια όσο και για τον διακριτή. Αποτέλεσμα αυτού θα είναι να γίνονται λιγότεροι και πιο γρήγοροι οι υπολογισμοί των νευρώνων.

4.4 Μελλοντικές επεκτάσεις

Η εργασία αυτή, αν και εκπονείται στα πλαίσια του προπτυχιακού επιπέδου, θεωρείται πως μπορούσε να αποτελέσει το έναυσμα για μελλοντικές έρευνες, επεκτείνοντας το βασικό της περιεχόμενο της προς κάποιες πολύ ενδιαφέρουσες κατευθύνσεις, όπως:

- Επανεκπαίδευση του δικτύου σε μεγαλύτερο σύνολο δεδομένων εφόσον υπάρχει ο κατάλληλος εξοπλισμός που απαιτείται.
- Χρήση μικρότερων δικτύων γεννήτριας και διακριτή.

Βιβλιογραφία

[1] European Parliament- Τεχνητή νοημοσύνη οφέλη και απειλές https://www.europarl.europa.eu/news/el/headlines/society/20200918STO87404/techni ti-noimosuni-eukairies-kai-apeiles

[2] Wikipedia-Τεχνητή νοημοσύνη https://el.wikipedia.org/wiki/%CE%A4%CE%B5%CF%87%CE%BD%CE%B7%CF%84%CE%AE_%CE%BD%CE%BF%CE%B7%CE%BC%CE%BF%CF%83%CF% 8D%CE%BD%CE%B7

[3] Brownlee, J. (2019, July 19). A gentle introduction to generative adversarial networks (Gans). Machine Learning Mastery. Retrieved March 3, 2022, from https://machinelearningmastery.com/what-are-generative-adversarial-networks-gans/

[4] Τεχνητή Νοημοσύνη - Τι είνα και γιατί έχει σημασία. SAS. (n.d.). Retrieved March 3, 2022, from https://www.sas.com/el_gr/insights/analytics/what-is-artificial-intelligence.html

[5] Mehlig, B. (2021). Machine learning with neural networks. https://doi.org/10.1017/9781108860604

[6] J. A. Anderson, An Introduction to Neural Networks, MIT Press, Cambridge (1995). R. Lippmann, An Introduction to Computing with Neural Networks, IEEE ASSP Magazine

[7] Imran, M., & Sultan, M. (2021, November 3). Advantages of neural networks - benefits of AI and Deep Learning. Folio3AI Blog. Retrieved March 3, 2022, from https://www.folio3.ai/blog/advantages-of-neural-networks/

[8] Kaushik, V. (n.d.). 8 applications of Neural Networks. Analytics Steps. Retrieved March 3, 2022, from https://www.analyticssteps.com/blogs/8-applications-neural-networks

[9] Neural networks: Applications in the real world. upGrad blog. (2022, February 22). Retrieved March 3, 2022, from https://www.upgrad.com/blog/neural-networks-applications-in-the-real-world

[10] Fahim, M. A.-N., & Jung, H. Y. (2020). A lightweight GAN network for large scale fingerprint generation. IEEE Access, 1–1. https://doi.org/10.1109/access.2020.2994371

[11] Kumar, K., & Thakur, G. S. (2012). Advanced applications of neural networks and artificial intelligence: A Review. International Journal of Information Technology and Computer Science, 4(6), 57–68. https://doi.org/10.5815/ijitcs.2012.06.08

[12] Kiani, K., Hemmatpour, R., & Rastgoo, R. (2021). Automatic Grayscale Image Colorization using a Deep Hybrid Model. Journal of Artificial Intelligence and Data Mining, 9(3), 321–328. https://doi.org/10.22044/jadm.2021.9957.2131

[13] Kouzouglidis, P., Sfikas, G., & Nikou, C. (2019). Automatic video colorization using 3D conditional generative adversarial networks. Advances in Visual Computing, 209–218. https://doi.org/10.1007/978-3-030-33720-9_16