ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΙΓΑΙΟΥ
ΤΜΗΜΑ ΜΑΘΗΜΑΤΙΚΩΝ
ΠΡΟΓΡΑΜΜΑ ΠΡΟΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ

ΑΝΑΓΝΩΡΙΣΗ ΕΙΚΟΝΩΝ
και
ΤΑΞΙΝΟΜΗΣΗ ΣΤΟΙΧΕΙΩΝ

Πτυχιακή Εργασία του
ΙΩΑΝΝΗ ΓΚΟΥΝΤΟΥΒΑ
Προπτυχιακού φοιτητή του
ΠΠΣ ΜΑΘΗΜΑΤΙΚΩΝ

Επιβλέπων: ΝΙΚΟΛΑΟΣ ΧΑΛΙΔΙΑΣ

ΣΑΜΟΣ, 19 ΦΕΒΡΟΥΑΡΙΟΥ 2024

ΑΝΑΓΝΩΡΙΣΗ ΕΙΚΟΝΩΝ και ΤΑΞΙΝΟΜΗΣΗ ΣΤΟΙΧΕΙΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ ΤΟΥ
ΙΩΑΝΝΗ ΓΚΟΥΝΤΟΥΒΑ

ΤΡΙΜΕΛΗΣ ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ:

1. ΝΙΚΟΛΑΟΣ, ΧΑΛΙΔΙΑΣ
2. ΑΝΔΡΕΑΣ, ΠΑΠΑΣΑΛΟΥΡΟΣ
3. ΠΑΝΑΓΙΩΤΗΣ, ΝΑΣΤΟΥ

ΗΜΕΡΟΜΗΝΙΑ ΕΞΕΤΑΣΗΣ : 19 ΦΕΒΡΟΥΑΡΙΟΥ 2024

ΤΙΤΛΟΣ ΔΙΠΛΩΜΑΤΙΚΗΣ : ΑΝΑΓΝΩΡΙΣΗ ΕΙΚΟΝΩΝ και ΤΑΞΙΝΟΜΗΣΗ ΣΤΟΙΧΕΙΩΝ.
Του Γκούντουβα Γιάννη
Φοιτητή του ΠΠΣ Μαθηματικών

Περίληψη

Τα τελευταία χρόνια τα πολυεπίπεδα τεχνητά νευρωνικά δίκτυα γίνονται ένα εργαλείο με πολλές εφαρμογές στην καθημερινότητα του ανθρώπου, στο επίκεντρο αυτής της τεχνολογικής επανάστασης βρίσκεται η βαθιά μάθηση. Γνωστές έννοιες από τα εφαρμοσμένα και υπολογιστικά μαθηματικά κυρίως από τον απειροστικό λογισμό, την γραμμική άλγεβρα, την βελτιστοποίηση και την θεωρεία προσέγγισης παρέχουν την δυνατότητα σε μία μηχανή να δημιουργήσει για την ίδια μία τεχνητή ευφυΐα με σκοπό την επίλυση προβλημάτων που της ανατίθενται. Αυτή η εργασία περιέχει μία αναφορά στις βασικές ιδέες που αποτελούν τη βάση της βαθιάς μάθησης από την οπτική των εφαρμοσμένων μαθηματικών. Δίνεται έμφαση σε τρία θεμελιώδη ερωτήματα: Τι είναι ένα βαθύ νευρωνικό δίκτυο; Τι είναι η μέθοδος της στοχαστικής κλήσης; Πως είναι ένα εκπαιδευμένο δίκτυο; Παρουσιάζονται με εικόνες οι ιδέες με έναν σύντομο κώδικα στην *Python* που δημιουργεί και εκπαιδεύει ένα δίκτυο. Αναφέρεται επίσης η χρήση λογισμικού τελευταίας τεχνολογίας για προβλήματα ταξινόμησης εικόνων μεγάλης κλίμακας. Στο τέλος της εργασίας, υπάρχουν αναφορές στην τρέχουσα βιβλιογραφία.

Ευχαριστίες

Στους Γονείς μου!

Στον θείο μου τον Σωτήρη!

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή

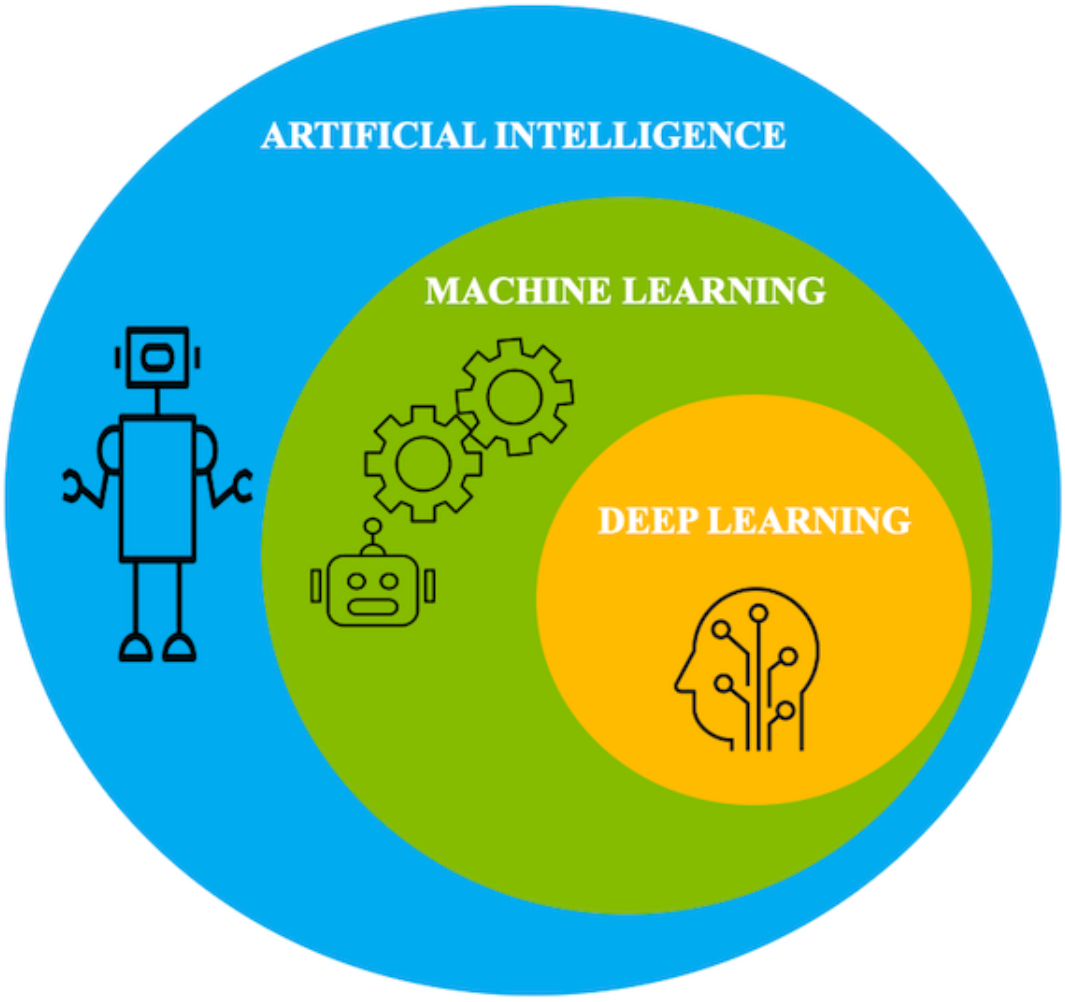
Οι περισσότεροι από εμάς έχουμε συναντήσει τη φράση βαθιά μάθηση. Σχετίζεται από ένα σύνολο εργαλείων που έχουν γίνει εξαιρετικά δημοφιλή σε ένα ευρύ φάσμα εφαρμογών, όπως η αναγνώριση εικόνας, η αναγνώριση ομιλίας, η επεξεργασία φυσικής γλώσσας, τα αυτοοδηγούμενα αυτοκίνητα, η στοχευμένη διαφήμιση και η ανακάλυψη φαρμάκων. Το συγκεκριμένο πεδίο τα τελευταία χρόνια έχει κάνει τεράστια πρόοδο στην επιστήμη των υπολογιστών λόγω της μεγαλύτερης διαθεσιμότητας δεδομένων και της αυξανόμενης ισχύς των υπολογιστών.

Η βαθιά μάθηση κάνει χρήση των νευρωνικών δικτύων για να πετύχει τους στόχους της. Τα νευρωνικά δίκτυα είναι θεωρητικά ικανά να μάθουν οποιαδήποτε μαθηματική συνάρτηση με επαρκή δεδομένα εκπαίδευσης και μερικές μεταβλητές, οι οποίες είναι υπεύθυνες για την απόδοση της λειτουργικότητας των νευρωνικών δικτύων, όπως τα επαναλαμβανόμενα νευρωνικά δίκτυα *RNN* είναι γνωστό ότι είναι πλήρη κατά "*Turing*". Η πληρότητα κατά *Turing* αναφέρεται στο γεγονός ότι ένα νευρωνικό δίκτυο μπορεί να προσομοιώσει οποιαδήποτε συνάρτηση μάθησης, δεδομένου ότι του παρέχονται επαρκή δεδομένα εκπαίδευσης. Το πρόβλημα είναι ότι ο όγκος των δεδομένων που απαιτούνται για την εκμάθηση ακόμα και απλών καθηκόντων είναι συχνά εξαιρετικά μεγάλος, γεγονός που προκαλεί αντίστοιχη αύξηση του χρόνου εκπαίδευσης για την αναγνώριση εικόνων, που είναι ένα απλό έργο για έναν άνθρωπο, μπορεί να είναι της τάξεως μερικών εβδομάδων ακόμη και για σύστημα υψηλής απόδοσης. Επιπλέον υπάρχουν πρακτικά ζητήματα που σχετίζονται με την ευστάθεια της εκπαίδευσης των νευρωνικών δικτύων, τα οποία προκαλούν προβλήματα στην μάθηση του δικτύου ακόμη και σήμερα. Παρόλα αυτά δεδομένου ότι η ισχύ των υπολογιστών αναμένεται να αυξηθεί ραγδαία με την πάροδο του χρόνου και θεμελιωδώς πιο ισχυρά παραδείγματα όπως η κβαντική υπολογιστική ισχύ είναι στον ορίζοντα, το υπολογιστικό ζήτημα μπορεί τελικά να μην είναι και τόσο κρίσιμο όσο φανταζόμαστε.

Τα νευρωνικά δίκτυα αναπτύχθηκαν με σκοπό την προσομοίωση του ανθρώπινου νευρωνικού συστήματος για λειτουργίες της μηχανικής μάθησης, που βασίζονται στην επεξεργασία των υπολογιστικών μονάδων σε ένα μοντέλο εκπαίδευσης με τρόπο παρόμοιο με τους ανθρώπινους νευρώνες. Το ευρύτερο όραμα των νευρωνικών δικτύων είναι η δημιουργία τεχνητής νοημοσύνης με την κατασκευή μηχανών των οποίων η αρχιτεκτονική τους προσομοιώνει τους υπολογισμούς που πραγματοποιούνται στο ανθρώπινο νευρικό σύστημα. Αυτό προφανώς δεν είναι ένα απλό εγχείρημα επειδή η υπολογιστική ισχύς του ταχύτερου υπολογιστή σήμερα είναι ένα μικροσκοπικό κλάσμα της υπολογιστικής

ισχύς του ανθρώπινου εγκεφάλου. Τα νευρωνικά δίκτυα αναπτύχθηκαν αμέσως μετά την έλευση των ηλεκτρονικών υπολογιστών κατά την δεκαετία του '50 και του '60. Ο αλγόριθμος *perceptron* του *Rosenblatt* θεωρήθηκε ως ο ακρογωνιαίος λίθος των νευρωνικών δικτύων, ένα γραμμικό μοντέλο που βασίστηκε στις λογικές πύλες, γεγονός που προκάλεσε έναν ενθουσιασμό σχετικά με την προοπτική της τεχνητής νοημοσύνης. Ωστόσο μετά την αρχική ευφορία, υπήρξε μια περίοδος απογοήτευσης στην οποία η επιτακτική ανάγκη για δεδομένα και η υπολογιστικά εντατική φύση των νευρωνικών δικτύων θεωρήθηκαν εμπόδιο στην χρηστικότητά τους.

Αν και η βιολογική αναλογία των νευρωνικών δικτύων είναι συναρπαστική και μας θυμίζει σενάρια επιστημονικής φαντασίας, η μαθηματική κατανόηση των νευρωνικών δικτύων είναι λιγότερο εντυπωσιακή. Η αφαίρεση του νευρωνικού δικτύου μπορεί να θεωρηθεί ως μια δομοστοιχειωτή προσέγγιση που καθιστά δυνατούς τους αλγόριθμους μάθησης που βασίζονται στην συνεχή βελτιστοποίηση σε ένα υπολογιστικό γράφημα εξαρτήσεων μεταξύ εισόδου και εξόδου.



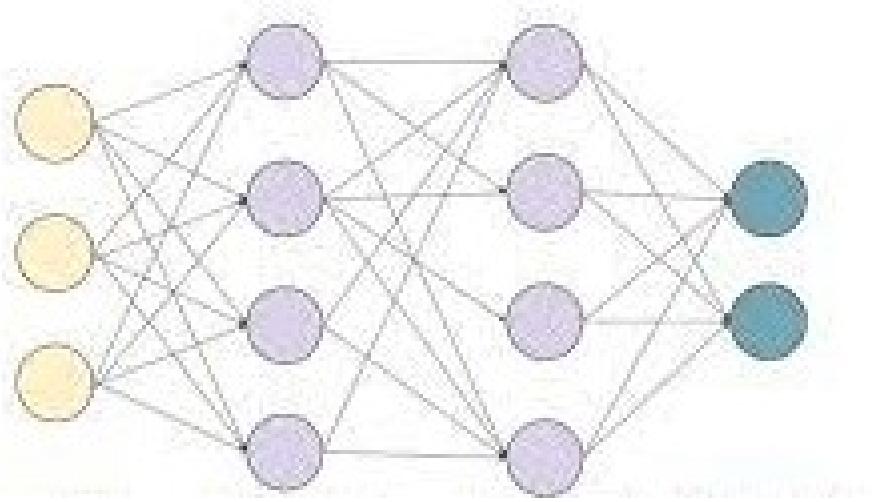
Σχήμα 1.1

1.1 Παράδειγμα Αρχιτεκτονικής Νευρωνικού Δικτύου

Νευρωνικό δίκτυο ονομάζεται ένα κύκλωμα διασυνδεδεμένων νευρώνων. Στην περίπτωση βιολογικών νευρώνων, πρόκειται για ένα τμήμα νευρωνικού ιστού. Στην περίπτωση τεχνιτών νευρώνων, πρόκειται για ένα αφηρημένο αλγοριθμικό κατασκεύασμα το οποίο εμπίπτει στον τομέα της υπολογιστικής νοημοσύνης, όταν στόχος του νευρωνικού δικτύου είναι η επίλυση κάποιου υπολογιστικού προβλήματος ή της υπολογιστικής νευροεπιστήμης, όταν στόχος είναι η υπολογιστική προσομοίωση της λειτουργίας των βιολογικών νευρωνικών δικτύων με βάση κάποιο μαθηματικό μοντέλο.

Η δομή του νευρωνικού δικτύου αποτελείται από απλούς υπολογιστικούς κόμβους τους νευρώνες, οι οποίοι είναι πλήρως διασυνδεδεμένοι μεταξύ τους. Είναι εμπνευσμένη η αρχιτεκτονική του από το κεντρικό νευρικό σύστημα το οποίο προσπαθεί να προσομοιώσει.

Τα νευρωνικά δίκτυα είναι ένας τύπος αλγορίθμου μηχανικής μάθησης και αποτελούν μια μέθοδο στη τεχνητή νοημοσύνη, που διδάσκει στους υπολογιστές να επεξεργάζονται δεδομένα με τρόπο εμπνευσμένο από τη δομή και τη λειτουργία του ανθρώπινου εγκεφάλου. Υπάρχουν δίκτυα ενός και πολλαπλών επιπέδων. Στο δίκτυο ενός επιπέδου, ένα σύνολο εισόδων χαρτογραφείται απευθείας σε μία είσοδο χρησιμοποιώντας μια γενικευμένη παραλλαγή γραμμικής συνάρτησης. Αυτή η απλή υποστασιοποίηση ενός νευρωνικού δικτύου αναφέρεται επίσης ως *perceptron*. Στα νευρωνικά δίκτυα πολλαπλών επιπέδων, οι νευρώνες διατάσσονται σε επίπεδα, όπου τα επίπεδα εισόδου και εξόδου χωρίζονται από μία ομάδα κρυφών επιπέδων. Αυτή η αρχιτεκτονική επιπέδων του νευρωνικού δικτύου αναφέρεται επίσης ως εμπρόσθετο δίκτυο, δηλαδή είναι ένα δίκτυο, όπου μέσα σε αυτό η πληροφορία μεταφέρεται μόνο προς μία κατεύθυνση.



Σχήμα 1.2: Ένα τεχνητό νευρωνικό δίκτυο

Στο Σχήμα 1.2 αναπαριστάται μια τυπική μορφή ενός νευρωνικού δικτύου. Εμφανίζονται επίπεδα εισόδου-εξόδου και δύο κρυφά επίπεδα. Το επίπεδο εισόδου αποτελείται από τρεις αριθμητικές τιμές. Το επίπεδο εξόδου αποτελείται από δύο νευρώνες που θα περιέχουν μια τιμή έκαστος, οι οποίες μπορεί να συμβολίζουν την πιθανότητα δύο κατηγοριών ή δύο εκδοχές λύσεων ενός προβλήματος.

Τα κρυφά επίπεδα νευρώνων περιέχουν τέσσερεις νευρώνες το καθένα και είναι πλήρως συνδεδεμένα στρώματα. Κάθε είσοδος σε ένα νευρώνα είναι βαθμωτή με ένα βάρος, το οποίο επηρεάζει τη συνάρτηση που υπολογίζεται σε αυτή τη μονάδα. Ένα τεχνητό νευρωνικό δίκτυο υπολογίζει μια συνάρτηση των εισόδων μεταδίδοντας τις υπολογιζόμενες τιμές από τους νευρώνες εισόδου στους νευρώνες εξόδου και χρησιμοποιώντας τα βάρη ως ενδιάμεσες παραμέτρους. Η μάθηση γίνεται με την αλλαγή της τιμής των βαρών που συνδέουν τους νευρώνες. Το δίκτυο αποτελείται συνολικά από δέκα νευρώνες με $3\times4 + 4\times4 + 4\times4 + 4\times2 = 52$ βάρη και 10 τιμές μεροληψίας. Κατά την διάρκεια της εκπαίδευσης του δικτύου αυτές οι τιμές των συντελεστών βάρους και μεροληψίας θα μεταβάλλονται με στόχο να παρθεί η βέλτιστη λύση χωρίς την συμμετοχή του ανθρώπινου παράγοντα. Αυτό πραγματοποιείται καθώς μπορούν να εκπαιδευτούν σε μεγάλα σύνολα δεδομένων και να μοντελοποιήσουν περίπλοκες σχέσεις μεταξύ δεδομένων εισόδου και εξόδου, βγάζοντας συμπεράσματα και κάνοντας διάφορες γενικεύσεις.

Ο στόχος αυτής της εργασίας είναι η παρουσίαση της λειτουργίας κατά την οποία μία μηχανή εκπαιδεύεται στην επίλυση ορισμένων προβλημάτων με αποτέλεσμα να δημιουργείται τεχνητή ευφυΐα στο μηχάνημα και ύστερα από αυτήν την εκπαίδευση δεν χρειάζεται την ανθρώπινη παρέμβαση σε ζητήματα για τα οποία έχει περάσει εκπαίδευση. Στο κεφάλαιο 2 παρουσιάζεται η δομή ενός νευρωνικού δικτύου. Στο κεφάλαιο 3 αναλύονται οι μαθηματικές διαδικασίες που ακολουθεί μια μηχανή έτσι ώστε να αποκτήσει λογική κρίση. Στο κεφάλαιο 4 αναφέρεται η αρχιτεκτονική των συνελλικτικών νευρωνικών δικτύων και ο τρόπος λειτουργίας τους. Στο πέμπτο και τελευταίο κεφάλαιο παρουσιάζεται ένα παράδειγμα εκπαίδευσης μίας μηχανής στην ταξινόμηση εικόνων μεταξύ πέντε κλάσεων.

Κεφάλαιο 2

Δομή Νευρωνικών Δικτύων

2.1 *Perceprton*

Ο νευρώνας *Perceptron* ή Αντίληπτρο είναι ένα είδος τεχνητού νευρωνικού δικτύου που εφευρέθηκε το 1957 στο Αεροναυτικό Εργαστήριο του Κόρνελλ από τον *Frank Rosenblatt*. Μπορεί να χαρακτηριστεί ως ένα απλό είδος προς τα εμπρός τροφοδότησης δικτύου, δηλαδή, τροφοδοτούμενο μόνο προς την μία κατεύθυνση νευρωνικό δίκτυο.

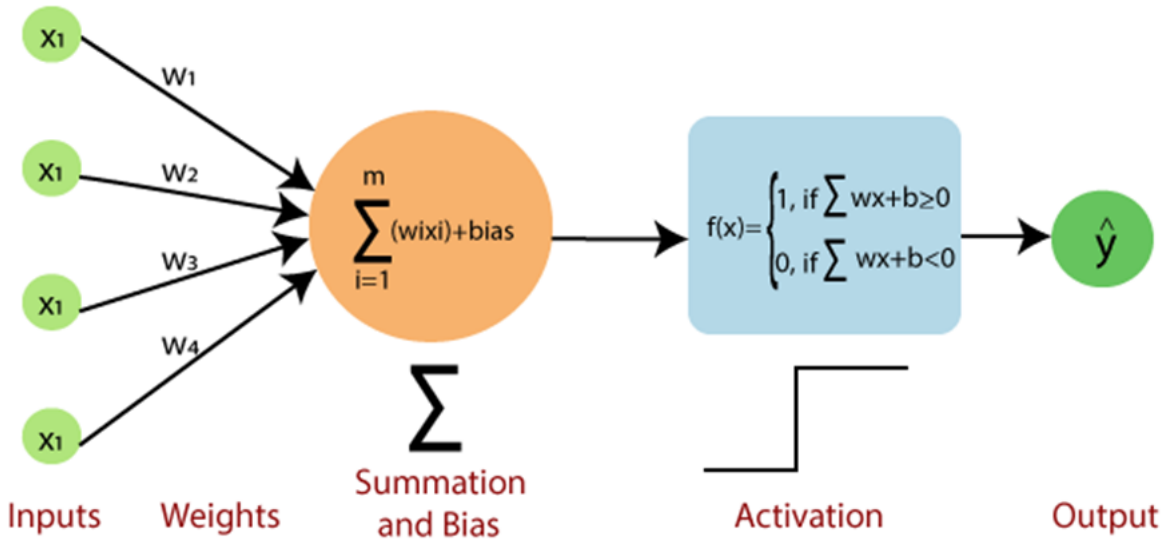
Το απλούστερο νευρωνικό δίκτυο αναφέρεται ως *Perceptron*. Αυτό το νευρωνικό δίκτυο περιέχει μόνο ένα επίπεδο εισόδου και έναν κόμβο εξόδου. Η αρχιτεκτονική του *Perceptron* φαίνεται στο Σχήμα 2.1 στο οποίο ένα επίπεδο εισόδου μεταδίδει τα χαρακτηριστικά στον κόμβο εξόδου. Οι άκρες από την είσοδο στην έξοδο περιέχουν τα βάρη $w_1, ..., w_d$ με τα οποία τα χαρακτηριστικά πολλαπλασιάζονται και προστίθενται στον κόμβο εξόδου. Ο *Perceptron* είναι ένας δυαδικός ταξινομητής δηλαδή μία συνάρτηση η οποία απεικονίζει την είσοδο x(ένα διάνυσμα με πραγματικές τιμές) σε μία τιμή εξόδου $f(x)$ (Μία και μοναδική τιμή).

$$f(x) = \begin{cases} 1 & wx + b > 0 \\ 0 & wx + b \leq 0 \end{cases} \tag{2.1}$$

Το w είναι ένα διάνυσμα από βάρη με πραγματικές τιμές και wx είναι το εσωτερικό γινόμενο μεταξύ των διανυσμάτων w και x. Το b είναι ένας σταθερός όρος που δεν εξαρτάται από καμία τιμή εισόδου και ονομάζεται μεροληψία.

Η τιμή της συνάρτηση $f(x)$ χρησιμοποιείται για να ταξινομήσει το x είτε ως θετικό ή αρνητικό στιγμιότυπο, στην περίπτωση ενός δυαδικού προβλήματος ταξινόμησης. Η μεροληψία χρησιμοποιείται για την μετατόπιση της συνάρτηση ενεργοποίησης ή για να δώσει στον νευρώνα εξόδου ένα βασικό επίπεδο δραστηριότητας. Αν το b είναι αρνητικό τότε ο βεβαρημένος συνδυασμός των εισόδων πρέπει να παράγει μία θετική τιμή μεγαλύτερη του $-b$, έτσι ώστε να αναγκάσει τον νευρώνα που ταξινομεί να έχει τιμή μεγαλύτερη της μεροληψίας 0. Χωρικά η μεροληψία μεταβάλλει την θέση αλλά όχι τον προσανατολισμό του συνόρου απόφασης.

Στο Σχήμα 2.1 παρουσιάζεται η λειτουργία του *Perceptron* ως νευρωνικό δίκτυο.



Σχήμα 2.1: Η βασική αρχιτεκτονική του *Perceptron*

Εφόσον οι είσοδοι τροφοδοτούνται στο δίκτυο άμεσα μέσω των βεβαρημένων συνδέσεων, ο νευρώνας *Perceptron* μπορεί να θεωρηθεί ως ένα απλό είδος νευρωνικού δικτύου προς τα εμπρός τροφοδότησης.

2.2 Νευρώνας

Το βασικό δομικό υλικό των νευρωνικών δικτύων είναι ο νευρώνας. Ο εκάστοτε νευρώνας ενός δικτύου δέχεται μία ή περισσότερες αριθμητικές εισόδους και παράγει μια έξοδο. Αυτή η διαδικασία είναι ένας μαθηματικός μετασχηματισμός κάποιας συνάρτησης ενεργοποίησης. Επίσης ο κάθε νευρώνας χαρακτηρίζεται από τα βάρη που ισοσταθμίζουν τις εισόδους και το κατώφλι/μεροληψία που προστίθεται στον νευρώνα.

Σχήμα 2.2: Αριστερά παρουσιάζεται ένα ανθρώπινο νευρωνικό κύτταρο ενώ δεξιά ένα τεχνητό

Τα βάρη και η μεροληψία είναι παράμετροι εκπαίδευσης, όπου μεταβάλλονται κατά την διάρκεια της εκπαίδευσης του συστήματος, με στόχο την βέλτιστη προσέγγιση μιας πρόβλεψης. Έτσι κατά την διάρκεια της μάθησης ο κάθε νευρώνας μαθαίνει να αναγνωρίζει τα δεδομένα που εκπαιδεύεται, αντίστοιχα με έναν άνθρωπο που λαμβάνει ερεθίσματα συνεχώς.

Ο νευρώνας πήρε το όνομά του από το νευρικό κύτταρο του ανθρώπου αφού λειτουργεί λαμβάνοντας ερεθίσματα και με την σειρά του τα μεταβιβάζει σε όλους τους νευρώνες του επόμενου στρώματος του νευρωνικού δικτύου. Οι νευρώνες οργανώνονται σε στρώματα που είναι πλήρως συνδεδεμένα μεταξύ τους, δηλαδή ο κάθε νευρώνας συνδέεται με όλους του επόμενου στρώματος.

2.3 Συνάρτηση Ενεργοποίησης

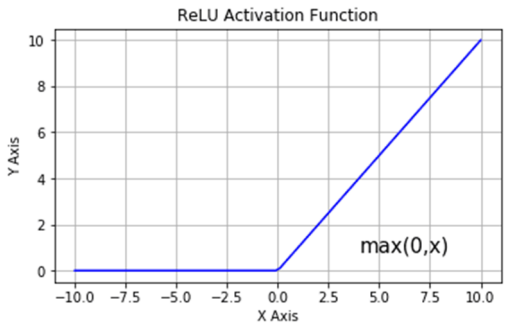
Σε αυτήν την ενότητα αναλύεται η χρήση των συναρτήσεων ενεργοποίησης στα νευρωνικά δίκτυα αλλά και η νευραλγική της λειτουργία στην εκπαίδευση του δικτύου, καθώς χωρίς την κατάλληλη συνάρτηση δεν υπάρχει μάθηση. Η συνάρτηση ενεργοποίησης είναι ουσιαστικά μία συνάρτηση που τροφοδοτεί τον εκάστοτε νευρώνα με πληροφορία προς επεξεργασία από το επίπεδο εισόδου μέχρι και το επίπεδο εξόδου του δικτύου. Ενεργοποιεί, δηλαδή, κάθε νευρώνα με την σειρά του και τον θέτει σε λειτουργία.

Η επιλογή της συνάρτησης ενεργοποίησης είναι ένα κρίσιμο μέρος του σχεδιασμού νευρωνικών δικτύων. Στην περίπτωση του *perceptron* η επιλογή της συνάρτησης ενεργοποίησης προσήμου από το γεγονός ότι ο στόχος της πρόβλεξης είναι δυαδικής κλάσης, όμως είναι πιθανό να υπάρχουν και άλλοι τύποι καταστάσεων όπου μπορούν να προβλεφθούν. Για παράδειγμα αν η μεταβλητή στόχος της εκπαίδευσης είναι πραγματική τότε λογικό είναι να χρησιμοποιήσουμε την συνάρτηση ενεργοποίησης ταυτότητας, ενώ αν είναι δυαδική μπορούμε να χρησιμοποιήσουμε την σιγμοειδή συνάρτηση.

Η ενεργοποίηση του νευρώνα γίνεται υπολογίζοντας με βάση την είσοδο που δέχεται, τα βάρη και την τιμή της μεροληψίας, την έξοδο. Έτσι όταν ένας νευρώνας αναφέρεται ότι ενεργοποιείται σημαίνει ότι έχει κάποια μεγάλη τιμή, σε αντίθετη περίπτωση πολύ μικρής τιμής μέσω της πυροδότησής του παραμένει απενεργοποιημένος.

Επομένως η μαθηματική συνάρτηση ενεργοποίησης μπορεί να πάρει πολλές μορφές ανάλογα με την μεταβλητή στόχο της πρόβλεψεις. Ακολουθούν κάποιες από της πιο διαδεδομένες συναρτήσεις ενεργοποίησης στα νευρωνικά δύκτια.

• Η *ReLU* (*Rectified Linear Unit*) είναι ίσως η πιο δημοφιλής συνάρτηση ενεργοποίησης αυτήν την στιγμή στα νευρωνικά δίκτυα. Παρουσιάστηκε από τον *Hanhloser* το 2000. Επιτρέπει την γρηγορότερη και πιο αποτελεσματική μάθηση σε μεγάλα σύνολα δεδομένων σε σχέση με την σιγμοειδή συνάρτηση ή την συνάρτηση υπερβολικής εφαπτομένης.



Σχήμα 2.3: Η γραφική παράσταση της συνάρτησης ενεργοποίησης Ανορθωμένης Γραμμικής Μονάδας

Αυτό το γεγονός προκύπτει από το ότι η γραμμική μετάβαση στο θετικό μέρος του άξονα των xx' δεν παραμορφώνει την παράγωγο. Συνεπώς η διαδικασία της βελτιστοποίησης της πρόβλεψης πραγματοποιείται χωρίς το πρόβλημα της εξαφάνισης της παραγώγου, το οποίο είναι αρκετά σύνηθες πρόβλημα σε συναρτήσεις ενεργοποίησης. Συγκεκριμένα, όταν η παράγωγος της συνάρτησης ενεργοποίησης είναι μηδέν ο νευρώνας παραμένει σε μια διαρκώς ανενεργή κατάσταση. Αυτή είναι μια μορφή του προβλήματος της εξαφάνισης της παραγώγου.

Σε ορισμένες περιπτώσεις, μεγάλος αριθμός νευρώνων σε ένα δίκτυο μπορεί να αδρανοποιείται σε νεκρές καταστάσεις, μειώνοντας ουσιαστικά την χωρητικότητα του μοντέλου. Αυτό το πρόβλημα προκύπτει συνήθως όταν ο ρυθμός εκμάθησης είναι πολύ υψηλός, δηλαδή όταν το δίκτυο δέχεται πολύ μεγάλο όγκο δεδομένων στην εκπαίδευση σε μικρό χρονικό διάστημα. Υπάρχουν διάφορες παραλλαγές των συναρτήσεων ενεργοποίησης, έτσι ώστε να αποφεύγονται τέτοιου είδους προβλήματα. Ο τύπος της *ReLU* είναι ο εξής:

$$\phi(z) = max(0, z)$$

Τμηματικές παραλλαγές *ReLU*

LeakyReLU

Οι διαρροές *ReLU* επιτρέπουν μια μικρή, θετική κλίση όταν η μονάδα δεν είναι ενεργή, συμβάλλοντας στην άμβλυνση του προβλήματος της εξαφάνισης της κλίσης ή της παραγώγου.

$$f(x) = \begin{cases} x & x > 0 \\ 0.01x & x \leq 0 \end{cases} \tag{2.2}$$

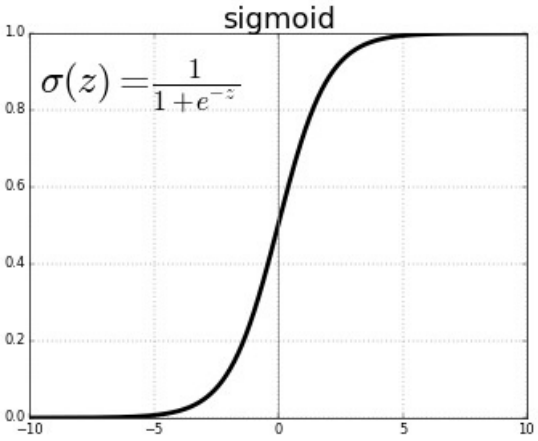
ParametrikReLU

Τα παραμετρικά *ReLUs* προωθούν αυτήν την ιδέα περαιτέρω κάνοντας τον συντελεστή διαρροής σε μια παράμετρο που μαθαίνεται μαζί με τις άλλες παραμέτρους του νευρωνικού δικτύου.

$$f(x) = \begin{cases} x & x > 0 \\ ax & x \leq 0 \end{cases} \tag{2.3}$$

• Η σιγμοειδής συνάρτηση ενεργοποίησης έχει την ιδιότητα να μετατρέπει τους μεγάλους αριθμούς σε 1 και τους μικρούς σε 0. Το όνομα της συνάρτησης οφείλεται στο σχήμα της γραφικής της παράστασης καθώς θυμίζει το λατινικό γράμμα S. Χρησιμοποιείται συχνά σε προβλήματα ταξινόμησης με στόχο 2 πιθανές κλάσεις γιατί κάνει ταχύτερη την εκπαίδευση του δικτύου. Η συγκεκριμένη συνάρτηση όμως επειδή από πολύ νωρίς τα άκρα της συνάρτησης τείνουν στο 0 προκαλείται το πρόβημα της εξαφάνισης της παραγώγου με αποτέλεσμα σε κάποιο μεγάλο δίκτυο να μην μπορεί να τελεσφορήσει η εκπαίδευση. Ο τύπος της σιγμοειδούς συνάρτησης είναι ο εξής:

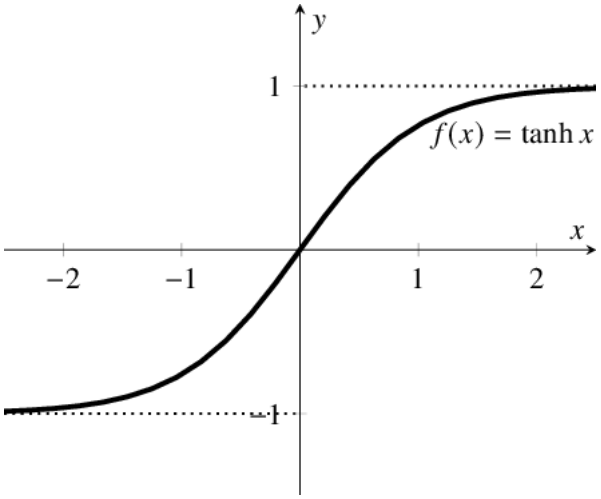
$$\sigma(z) = \frac{1}{1 + e^{-z}}$$



Σχήμα 2.4: Η γραφική παράσταση της σιγμοειδής συνάρτησης ενεργοποίησης

• Η συνάρτηση υπερβολικής εφαπτομένης ενεργοποίησης έχει σχήμα παρόμοιο με εκείνο της σιγμοειδούς συνάρτησης, εκτός από το ότι ορίζεται στο διάστημα [-1,1] και είναι οριζόντια επανακλιμακούμενη και μεταφερόμενη κατακόρυφα. Η συνάρτηση εφαπτομένης και η σιγμοειδής συνάρτηση σχετίζονται ως εξής:

$$\tanh(u) = 2 * \sigma(2u) - 1$$



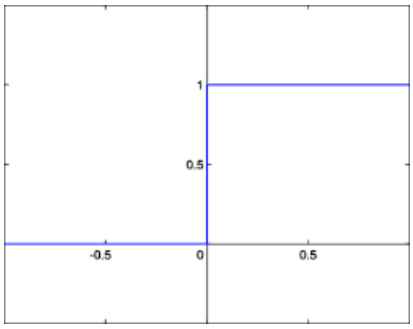
Σχήμα 2.5: Η γραφική παράσταση της υπερβολικής εφαπτομένης συνάρτησης ενεργοποίησης

Πολλές φορές προτιμάται από τη σιγμοειδή όταν θέλουμε οι έξοδοι των υπολογισμών να είναι και θετικές και αρνητικές. Επιπλέον η κεντραρισμένη στο μέσο και μεγαλύτερη βαθμίδα σε σχέση με την σιγμοειδή διευκολύνει την εκπαίδευση. Αυτές οι δύο συναρτήσεις υπήρξαν τα ιστορικά εργαλεία επιλογής για την ενσωμάτωση της μη γραμμικότητας στο νευρωνικό δίκτυο. Τα τελευταία χρόνια όμως έχουν γίνει πιο δημοφιλείς ορισμένες συναρτήσεις γραμμικής ενεργοποίησης σε τμηματικό επίπεδο όπως η συνάρτηση απότομης εφαπτομένης και η *ReLU*. Ο τύπος της υπερβολικής εφαπτομένης είναι ο εξής:

$$\varphi(z) = \frac{e^{2z} - 1}{e^{2z} + 1}$$

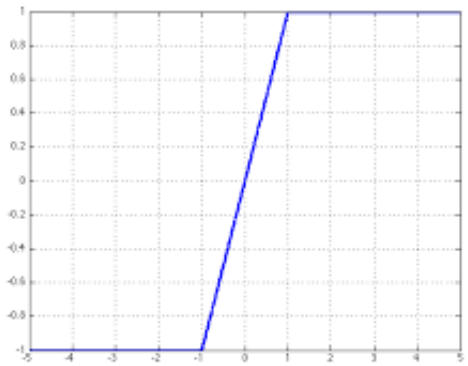
• Η συνάρτηση προσήμου είναι μια συνάρτηση ενεργοποίησης αρκετά κλασική για τα νευρωνικά δίκτυα. Μπορεί να χρησιμοποιηθεί για την απεικόνιση στοιχείων στον χώρο των δεδομένων σε δυαδικές εξόδους σε χρόνο πρόβλεψης. Η μη διαφορισιμότητά της αποτρέπει τη χρήση της για τη δημιουργία της συνάρτησης απώλειας σε χρόνο εκπαίδευσης. Ένα αρκετό καλό παράδειγμα είναι το *perceptron* όπου κατά την εκπαίδευση απαιτεί μόνο γραμμική ενεργοποίηση. Ο τύπος της συνάρτησης προσήμου είναι ο εξής:

$$\varphi(z) = sign(z)$$



Σχήμα 2.6: Η γραφική παράσταση της συνάρτησης ενεργοποίησης προσήμου

• Η συνάρτηση απότομης εφαπτομένης όπως και η *ReLU* έχουν αντικαταστήσει σε μεγάλο βαθμό τις συναρτήσεις ενεργοποίησης της σιγμοειδής και της υπερβολικής εφαπτομένης στα σύγχρονα νευρωνικά δύκτια λόγω της ευκολίας στην εκπαίδευση των νευρωνικών δυκτίων πολλών επιπέδων με αυτές τις συναρτήσεις ενεργοποίησης.



Σχήμα 2.7: Η γραφική παράσταση της απότομης εφαπτομένης συνάρτησης ενεργοποίησης

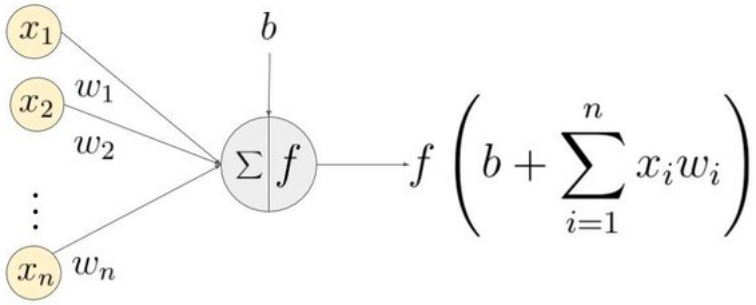
Σε σχέση με την υπερβολική εφαπτομένη είναι φθηνότερη υπολογιστικά και για τιμές μεγαλύτερες από το 1 υφίσταται κορεσμό. Ο τύπος της συνάρτηση απότομης εφαπτομένης είναι ο εξής:

$$\phi(z) = max[min[z, 1], -1]$$

2.4 Βάρη και Μεροληψία

Τα βάρη και η τιμή της μεροληψίας σε κάθε νευρώνα κατέχουν μεγάλο ρόλο στην λειτουργία του νευρωνικού δικτύου. Τα βάρη σε κάθε επίπεδο είναι ένα ενιαίο διάνυσμα διαστάσεων ισότιμο με τον αριθμό των νευρώνων, όπου συμβολίζεται με W και η μεροληψία είναι μία τιμή πόλωσης του εκάστοτε νευρώνα και συμβολίζεται με b. Αυτοί οι συντελεστές είναι υπεύθυνοι για την σωστή λειτουργία της μάθησης μίας μηχανής, δηλαδή είναι οι παράμετροι που θα καθορίσουν την ακρίβεια των προβλέψεων του δικτύου. Όταν ξεκινάει η εκπαίδευση τα βάρη και η μεροληψία αρχικοποιούνται με διάφορες μεθόδους, ίσως η πιο γνωστή είναι η μέθοδος *Kaiming* η οποία είναι μια μέθοδος αρχικοποίησης για τα νευρωνικά δίκτυα όπου λαμβάνει υπόψη τη μη γραμμικότητα των συναρτήσεων ενεργοποίησης όπως οι ενεργοποιήσεις *ReLU*. Μία σωστή μέθοδος αρχικοποίησης θα πρέπει να μην προσφέρει πολύ μικρές ή πολύ μεγάλες τιμές σημάτων εισόδου εκθετικά. Οι παράμετροι αυτοί προστίθενται σε κάθε νευρώνα με τον παρακάτω τρόπο:

$$b + \sum_{i=1}^{n} x_i w_i \tag{2.4}$$



Σχήμα 2.8: Τα βάρη W_i και η μεροληψία b κατά την ενεργοποίηση του νευρώνα

Στο Σχήμα 2.8 παρουσιάζεται ο τρόπος που δέχεται τις τιμές ο κάθε νευρώνας κατά την ενεργοποίησή του .

Κεφάλαιο 3

Εκπαίδευση Νευρωνικού Δικτύου

3.1 Συνάρτηση Κόστους

Στο προηγούμενο κεφάλαιο παρουσιάστηκε η διάδοση προς τα εμπρός μεταβιβάζοντας πληροφορία μόνο από τα επίπεδα εισόδου προς την έξοδο. Σε αυτό το κεφάλαιο λύνονται προβλήματα όπως το πόσο καλό είναι ένα μοντέλο εκπαίδευσης και πόσο προσεγγιστική είναι η πρόβλεψη της μηχανής μετά το πέρας της εκπαίδευσης.

Το σφάλμα μεταξύ δύο τιμών υπολογίζεται από την συνάρτηση απώλειας

$$Cost(\hat{y}, y) = \frac{1}{2}(\hat{y} - y)^2 \tag{3.1}$$

Αυτή είναι μια αρκετά απλή συνάρτηση απώλειας, δεν χρησιμοποιείται σε λογιστική παλινδρόμηση, επειδή το πρόβλημα βελτιστοποίησης είναι μη κυρτό.

Μια καλύτερη συνάρτηση απώλειας για χρήση είναι η εξής:

$$Cost(\hat{y}, y) = -ylog(\hat{y}) - (1 - y)log(1 - \hat{y}) \tag{3.2}$$

Υπάρχουν πολλές επιλογές για τη λειτουργία απώλειας, αφού έχει σημαντικό ρόλο στην εκπαίδευση των νευρωνικών δικτύων. Για παράδειγμα η Σχέση 3.2 αντιπροσωπεύει το σφάλμα μεταξύ δύο τιμών της πρόβεψης $\hat{y}$ και της πραγματικής y. Αν $y = 1$ τότε η Σχέση 3.2 μετατρέπεται:

$$Cost(\hat{y}, y = 1) = -ylog(\hat{y}) \tag{3.3}$$

Αν $y = 0$ τότε η Σχέση 3.2 μετατρέπεται:

$$Cost(\hat{y}, y = 0) = -log(1 - \hat{y}) \tag{3.4}$$

Ο στόχος της εκπαίδευσης είναι η βέλτιστη προσέγγιση της πρόβλεψης, συνεπώς στη συνάρτηση απώλειας είναι χρήσιμο να μεγιστοποιείται η προσέγγιση της συνάρτησης στο 0, δηλαδή, $Cost(\hat{y}, y) \approx 0$. Για παράδειγμα, βλέποντας την τιμή του y από την οπτική της πιθανότητας κατανομής *Bernoulli*, όπου με πιθανότητα $\hat{y}$ μπορεί να πάρει

την τιμή του 1 και το 0 με πιθανότητα $1 - \hat{y}$. Η συνάρτηση μάζας πιθανότητας για μια τυχαία μεταβλητή *Bernoulli* είναι η εξής:

$$P(k|p) = k^p(1 - k)^(1 - p) \tag{3.5}$$

Όπου k είναι μια τιμή συγκεκριμένη και αντιπροσωπεύει τα δυνατά αποτελέσματα της κατανομής και το p το δυνατό αριθμό αποτελεσμάτων προβλέψεων, υπολογίζοντας όμως την τιμή του y η Σχέση 3.5 γίνεται:

$$P(y|\hat{y}, x) = y^{\hat{y}}(1 - y)^{1-\hat{y}} \tag{3.6}$$

Συνεπώς μεγιστοποιώντας την πιθανότητα P, έτσι ώστε να έρθει η ζητούμενη πιθανότητα ή τιμή του y. Υπάρχουν αρκετές επιλογές για την συνάρτηση κόστους. Σημαντικό ρόλο στην επιλογή της παίζει βασικό ρόλο η μορφή της παραγώγου της, καθώς ο αλγόριθμος οπισθοδιάδοσης που αναφέρεται στην επόμενη ενότητα χρησιμοποιεί την παράγωγο της συνάρτησης κόστους. Η συγκεκριμένη συνάρτηση που αναγράφεται στην Σχέση 3.6 έχει αρκετά περίπλοκη παράγωγο. Αν λογαριθμηστούν και τα δύο μέλη της Σχέσης 3.6 τότε έχουμε:

$$log(P(y|\hat{y}, x)) = log(y^{\hat{y}}(1 - y)^{1-\hat{y}}) = \hat{y}logy + (1 - \hat{y})log(1 - y) \tag{3.7}$$

Παρατηρείται ότι από την Σχέση 3.7 το δεξιό της μέρος γίνεται

$$-\mathcal{L}(\hat{y}, y)$$

Η ουσία είναι ότι όταν ελαχιστοποιήσουμε το L μεγιστοποιούμε την πιθανότητα να είναι η τιμή του y ίση με αυτή της πρόβλεψης $\hat{y}$. Άρα η συνάρτηση απώλειας L είναι ίδια με τη συνάρτηση κόστους K.

$$\mathcal{J} = \mathcal{L}(\hat{y}, y)$$

Σε αυτή την περίπτωση, ονομάζεται ο αλγόριθμος βελτιστοποίησης μέθοδος στοχαστικής κλίσης η οποία αναλύεται σε επόμενη ενότητα του κεφαλαίου. Μπορεί να χρησιμοποιηθεί αυτός ο αλγόριθμος και για ένα μέρος των δεδομένων m:

$$\mathcal{J} = \frac{1}{m}\sum_{i=1}^{m}\mathcal{L}(\hat{y}^i, y^i) \tag{3.8}$$

Αν m είναι ίσο με τον αριθμό των παραδειγμάτων εκπαίδευσης στα οποία υπάρχει πρόσβαση, συνήθως ονομάζεται ο αλγόριθμος βελτιστοποίησης κατά παρτίδα *gradient descent*. Αν m είναι μικρότερο από τον αριθμό των παραδειγμάτων εκπαίδευσης, τότε ονομάζεται *mini − batch gradient descent*.

Για απλότητα, σε αυτό το κεφάλαιο θα χρησιμοποιείται η συνάρτηση κόστους J

$$\mathcal{J} = \mathcal{L}(\hat{y}, y)$$

Ωραία, θέτοντας την συνάρτηση κόστους J για να ελαχιστοποιηθεί χρησιμοποιούμε *gradient descent* κάτι που απαιτεί τον υπολογισμό της κλίσης του J ως προς τις παραμέτρους που μπορούν να αλλάξουν όπως τα βάρη και οι μεροληψίες. Ο υπολογισμός αυτών των κλίσεων είναι ο στόχος της αντίστροφης διάδοσης ή του αλγόριθμου οπισθοδιάδοσης.

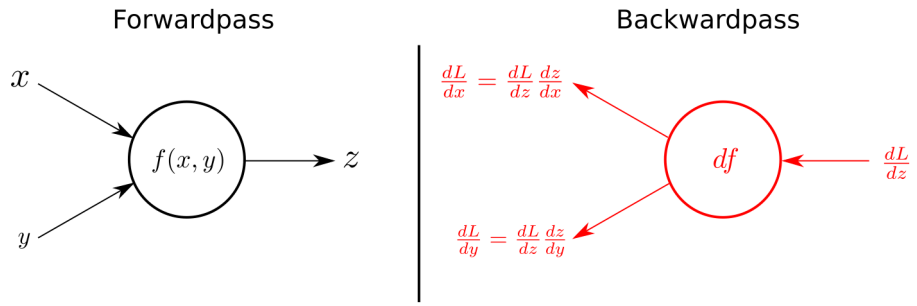
3.2 Αλγόριθμος Οπισθοδιάδοσης

Στο νευρωνικό δίκτυο ενός επιπέδου, η διαδικασία εκπαίδευσης είναι σχετικά απλή επειδή το σφάλμα μπορεί να υπολογιστεί ως άμεση συνάρτηση σύνθεσης των βαρών από προηγούμενα επίπεδα. Ο βαθμός μιας συνάρτησης σύνθεσης υπολογίζεται χρησιμοποιώντας τον αλγόριθμο οπισθοδιάδοσης. Ο αλγόριθμος οπισθοδιάδοσης εκμεταλλεύεται τον κανόνα της αλυσίδας του διαφορικού λογισμού, ο οποίος υπολογίζει τα σφάλματα βαθμίδων όσον αφορά τις αθροίσεις προϊόντων τοπικών βαθμίδων στις διάφορες διαδρομές από έναν κόμβο στην έξοδο. Αν και αυτό το άθροισμα έχει έναν εκθετικό αριθμό στοιχείων (διαδρομών), μπορεί κανείς να τον υπολογίσει αποτελεσματικά χρησιμοποιώντας δυναμικό προγραμματισμό. Ο αλγόριθμος οπισθοδιάδοσης είναι μια άμεση εφαρμογή δυναμικού προγραμματισμού. Περιέχει δύο φάσεις, που αναφέρονται για τον υπολογισμό των τιμών εξόδου και των τοπικών παραγώγων σε διάφορους κόμβους και η φάση προς τα πίσω απαιτείται για τη συσσώρευση των προϊόντων αυτών των τοπικών τιμών σε όλες τις διαδρομές από τον κόμβο στην έξοδο. Με λίγα λόγια η δυνατότητα μάθησης των νευρωνικών δικτύων οφείλεται στις παραμέτρους που καθορίζουν την μάθηση. Κάθε νευρώνας υπολογίζει τα βάρη του και την μεροληψία και είναι ένα μέρος του δικτύου που είναι προς εκπαίδευση. Οι τιμές λοιπόν αυτές των παραμέτρων είναι υπεύθυνες ως προς την σωστή εκπαίδευση του δικτύου επηρεάζοντας την έξοδο σε κάθε σετ προπόνησης. Το κλειδί στην υπόθεση εκπαίδευσης μιας μηχανής είναι ένα σωστό σετ τιμών των παραμέτρων αυτών έτσι ώστε η πρόβλεψη να είναι ακριβής. Αυτή η προσέγγιση των παραμέτρων γίνεται μέσω του αλγόριθμου οπισθοδιάδοσης μέσω επαναλήψεων των δύο φάσεων που περιγράφονται παρακάτω.

Φάση προς τα εμπρός: Σε αυτή τη φάση, οι είσοδοι για μια εκπαίδευση/περίπτωση τροφοδοτούνται στο νευρωνικό δίκτυο. Αυτό έχει ως αποτέλεσμα τους προς τα εμπρός διαδοχικά συνδεδεμένους υπολογισμούς σε όλα τα επίπεδα,χρησιμοποιώντας το τρέχον σύνολο βαρών. Η τελική προβλεπόμενη έξοδος μπορεί να σιγκριθεί με εκείνη της περίπτωσης της εκπαίδευσης και της απώλειας, πρέπει τώρα να υπολογιστεί σε σχέση με τα βάρη σε όλα τα επίπεδα της φάσης προς τα πίσω.

Φάση προς τα πίσω: Ο κύριος στόχος της προς τα πίσω φάσης είναι η μάθηση της βαθμίδας της συνάρτησης απώλειας σε σχέση με τα διαφορετικά βάρη, χρησιμοποιώντας τον κανόνα αλυσίδας του διαφορικού λογισμού. Αυτές οι βαθμίδες χρησιμοποιούνται για την ενημέρωση των βαρών. Δεδομένου ότι αυτές οι βαθμίδες μαθαίνονται στην κατεύθυνση προς τα πίσω, ξεκινώντας από τον κόμβο εξόδου, αυτή η διαδικασία μάθησης αναφέρεται ως προς τα πίσω φάση. Θεωρούμε μια σειρά κρυφών μονάδων($h_1, h_2, ..., h_k$) ακολουθούμενες από έξοδο m, σε σχέση με την οποία υπολογίζεται η συνάρτηση απώλειας L.

Η ολοκλήρωση του αλγορίθμου οπισθοδιάδοσης με την μεταβολή των παραμέτρων σε ένα νευρωνικό δίκτυο πάνω στο σύνολο των διαθέσιμων δεδομένων ονομάζεται εποχή. Κάποιες περιπτώσεις όμως είναι πιο δύσκολη η ανεύρεση των μερικών παραγώγων των



Σχήμα 3.1: Ο κανόνας της αλυσίδας με τον οποίο υπολογίζουμε το σφάλμα L

παραμέτρων λόγω του αυξημένου υπολογιστικού κόστους. Σε αυτήν την περίπτωση το εκπαιδευόμενο δίκτυο δέχεται στην είσοδο πακέτα δεδομένων, *minibatch*. Αυτή η διαδικασία μάθησης είναι η ίδια και η κάθε εποχή ολοκληρώνεται με την ανανέωση των παραμέτρων από τον αλγόριθμο οπισθοδιάδοσης του τελευταίου *minibatch*. Τέλος αυτή η διαδικασία επαναλαμβάνεται εκ νέου για όσες εποχές απαιτεί η μάθηση χρησιμοποιώντας τον κανόνα της αλυσίδας για την προς τα πίσω διάδοση όπως παρουσιάζεται στο Σχήμα 3.1.

Ο αλγόριθμος οπισθοδιάδοσης παρουσιάζει ένα μεγάλο μαθηματικό ενδιαφέρον, καθώς είναι ο μηχανισμός που αναφέρει πόσο γρήγορα μεταβάλλονται τα βάρη αλλά και οι μεροληψίες κατά την διάρκεια της εκπαίδευσης. Στην καρδιά αυτού του αλγορίθμου βρίσκεται η μερική παράγωγος των βαρών και των μεροληψειών του δικτύου, όπως φαίνεται και στο Σχήμα 3.1.

Ο εκάστοτε νευρώνας σε οποιοδήποτε εκπαιδευόμενο νευρωνικό δίκτυο μήκους n παρουσιάζει κάποιο βάρος w_i^n και θα δέχεται μια μεροληψία b_j^n. Συγκεκριμένα η ενεργοποίηση κάθε νευρώνα προς τα πίσω γίνεται ως εξής:

$$\delta_j^i = \frac{\partial \mathcal{J}}{\partial \sigma_j^i} \tag{3.9}$$

Το δ είναι το σφάλμα της εκπαίδευσης και από τη Σχέση 3.9 φαίνεται ότι το σφάλμα δ εξαρτάται ανάλογα με την ταχύτητα της μείωσης της συνάρτησης κόστους και αντιστρόφως ανάλογα από την ταχύτητα της συνάρτησης ενεργοποίησης. Η μορφή του δ δηλαδή θα εξαρτηθεί φυσικά από τη μορφή της συνάρτησης κόστους. Για παράδειγμα η συνάρτηση κόστους είναι η εξής:

$$Cost = \frac{1}{2}\sum_j (y_j - \sigma_j^L)^2$$

Της οποίας η παράγωγος είναι εύκολο να υπολογιστεί. Ξαναγράφοντας την Σχέση 3.9 στην Σχέση 3.10 παρατηρείται ότι η κλίση της συνάρτησης κόστους ορίζεται από ένα διάνυσμα του οποίου τα συστατικά είναι οι μερικές παράγωγοι $\partial \mathcal{J} / \partial \sigma_j^i$.

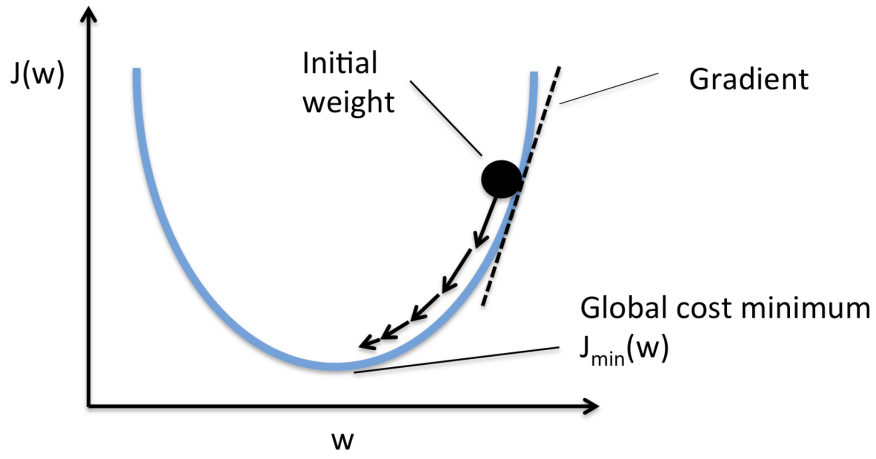
$$\delta_j^i = \nabla C \times \sigma(\acute{z}^L) \tag{3.10}$$

3.3 Στοχαστική Κατάβαση Βαθμίδας

Ο αλγόριθμος οπισθοδιάδοσης είναι υπεύθυνος για τον υπολογισμό της κατεύθυνσης της μεταβολής που πρέπει να δεχτεί κάθε παράμετρος ώστε να ελαχιστοποιηθεί η συνάρτηση κόστους. Η μαθηματική διαδικασία που καθορίζει τις μεταβολές των παραμέτρων ονομάζεται μέθοδος βελτιστοποίησης. Η πιο δημοφιλής μέθοδος που χρησιμοποιείται για αυτήν την λειτουργία είναι η *Stochastic gradient descent* ή (*SGD*).

Αυτή η μέθοδος λοιπόν είναι η διαδικασία μεταβολής των συντελεστών και ενημέρωση αυτών των τιμών μετά το πέρας της κάθε εποχής. Ειδικά όταν η εκπαίδευση συνοδεύεται με *minibatches* η ύπαρξη της μεθόδου *SGD* είναι απαραίτητη αλλά τροποποιημένη λίγο ως προς την λειτουργία της, καθώς ανανεώνονται οι τιμές των συντελεστών με βάση το τελευταίο *minibatch* και όχι με το πέρας της κάθε εποχής.

Το κάθε *minibatch* είναι ένα αντιπροσωπευτικό δείγμα του συνόλου δεδομένων όταν υπάρχει μεγάλη συσχέτιση των δεδομένων που εισάγει ο χρήστης, αλλιώς δεν θα ήταν αποτελεσματική η διαδικασία της εκπαίδευσης. Εν ολίγοις η *SGD* προσεγγίζει σε αρκετά καλό βαθμό την παραλλαγή των συντελεστών χωρίς μεγάλο υπολογιστικό κόστος, κάτι που κάνει ταχύτερη την εκπαίδευση του νευρωνικού δικτύου.



Σχήμα 3.2: Στοχαστική καταβίβαση της παραγώγου

Σίγουρα υπάρχουν και άλλες μέθοδοι βελτιστοποίησης παραλλαγές της *SGD* που χρησιμοποιούνται ευρέως, οι οποίες υπολογίζουν τα βάρη και η μεροληψία. Κάποιες από αυτές είναι η μέθοδος μάθησης *Adam* και η μέθοδος μάθησης *AdaGrad*. Όλες αυτές οι μέθοδοι βελτιστοποίησης είναι υπεύθυνες για τον ρυθμό μάθησης της μηχανής καθώς όπως εμφανίζεται και στο Σχήμα 3.2 στόχος αυτών των μαθηματικών διαδικασιών είναι ο μηδενισμός των παραγώγων των βαρών και των παραγώγων από τις εκάστοτε μεροληψίες με συνέπεια να μειώσουν την συνάρτηση απώλειας.

3.4 Παράδειγμα ταξινόμησης χειρόγραφων ψηφίων του *MNIST*

Η βάση δεδομένων *MNIST*, η οποία σημαίνει"Τροποποιημένη βάσης δεδομένων Εθνικού Ινστιτούτου Προτύπων και Τεχνολογίας", είναι μια μεγάλη βάση δεδομένων με

χειρόγραφα ψηφία. Όπως υποδηλώνει το όνομα. Αυτό το σύνολο δεδομένων δημιουργήθηκε με την τροποποίηση μιας πρότυπης βάσης δεδομένων με χειρόγραφα ψηφία που παρείχε η *MNIST*. Το σύνολο δεδομένων περιέχει 60.000 εικόνες εκπαίδευσης και 10.000 εικόνες δοκιμών. Κάθε εικόνα είναι μια σάρωση ενός χειρόγραφου ψηφίου από 0 έως 9 και οι διάφορές μεταξύ των διαφορετικών εικόνων είναι αποτέλεσμα των διαφορών στο χειρόγραφο αποτύπωμα ψηφίων διαφορετικών ατόμων. Αυτά τα άτομα ήταν υπάλληλοι της *American Census Bureau* και Αμερικάνοι μαθητές λυκείου. Οι αρχικές ασπρόμαυρες εικόνες από το *MNIST* είχαν κανονικοποιηθεί στο μέγεθος, ώστε να ταιριάζουν σε ένα κουτί εικονοστοιχείων 20×0, διατηρώντας παράλληλα την αναλογία τους και κεντραρίστηκαν σε μια εικόνα 28×28, υπολογίζοντας το κέντρο μάζας των εικονοστοιχείων. Οι εικόνες μεταφέρθηκαν για να τοποθετήσουν αυτό το σημείο στο κέντρο του πεδίου 28×28. Κάθε μια από αυτές βρίσκεται στο φάσμα της κλίμακας του γκρίζου. Οι ετικέτες που σχετίζονται με τις εικόνες αντιστοιχούν στις δέκα τιμές ψηφίων. Το μέγεθος του συνόλου δεδομένων είναι μικρό και περιέχει μόνο ένα απλό αντικείμενο που αντιστοιχεί σε ένα ψηφίο.

Ωστόσο, το μικρό μέγεθος αυτών των δεδομένων και η απλότητα είναι επίσης ένα πλεονέκτημα της βάσης αυτής επειδή μπορεί να χρησιμοποιηθεί ως εργαστήριο για γρήγορη δοκιμή αλγορίθμων μηχανικής μάθησης. Επιπλέον, η απλοποίηση του συνόλου δεδομένων λόγω του ότι τα ψηφία είναι κεντραρισμένα καθιστά εύκολη τη χρήση του για τον έλεγχο των αλγορίθμων πέρα από την όραση του υπολογιστή.

Αν και η αναπαράσταση πίνακα κάθε εικόνας είναι κατάλληλη για ένα συνελικτικό νευρωνικό δίκτυο, μπορεί κανείς να την μετατρέψει σε πολυδιάστατη αναπαράσταση διαστάσεων 28×28=784. Αυτή η μετατροπή χάνει ορισμένες από τις χωρικές πληροφορίες στην εικόνα, αλλά αυτή η απώλεια δεν μας καταβάλει ιδιαίτερα, λόγω της σχετικής απλότητάς της.

Τέλος, πρέπει να σημειωθεί ότι η 784-διαστάσεων μη χωρική αναπαράσταση των δεδομένων *MNIST* χρησιμοποιείται για τη δοκιμή όλων των τύπων αλγορίθμων νευρωνικών δικτύων πέρα από τον τομέα της όρασης του υπολογιστή. Παρόλο που η χρήση εικόνων 784-διαστάσεων είναι χρήσιμη για την δοκιμή της γενικής αποτελεσματικότητας των αλγορίθμων μη προσανατολισμένων στην όραση νευρωνικών δικτύων. Για παράδειγμα, τα δεδομένα *MNIST* χρησιμοποιούνται συχνά για τη δοκιμή γενικών αυτομάτων κωδικοποιητών και όχι μόνο των συνελικτικών. Γενικά τείνει να έχει ευρύτερη χρηστικότητα από πολλούς άλλους τύπους συνόλων δεδομένων.

Παρουσιάζεται στο Σχήμα 3.3 και στο Σχήμα 3.4 μια συγκεκριμένη απεικόνιση κώδικα *Python* που αφορά την πίσω διάδοση και την μέθοδο της στοχαστικής κλίσης της παραγώγου. Ειδικότερα στο πρόγραμμα επαναλαμβάνονται μια προς τα εμπρός και μετά προς τα πίσω περάσματα σε κάθε στρώμα του νευρωνικού δικτύου. Τα βάρη και οι μεροληψίες δεν έχουν την ίδια διάσταση σε κάθε στρώμα, δεν είναι βολικό να αποθηκεύονται σε τρισδιάστατο πίνακα. Χρησιμοποιείται, λοιπόν, ένας πίνακας δομής ανάλογος των διαστάσεων του νευρωνικού δικτύου αποθηκεύει τις τιμές των βαρών και των μεροληψιών και στη συνέχεια γίνονται τα περάσματα προς τα εμπρός και προς τα πίσω μέσα στους βρόχους.

Η συνάρτηση κόστους η οποία αξιολογεί την εκπαίδευση της μηχανής είναι κλιμακούμενη συνάρτηση. Επειδή αυτή η συνάρτηση είναι ένθετη, έχει πρόσβαση στις μεταβλητές της κύριας συνάρτησης, ιδίως στα δεδομένα εκπαίδευσης. Επισημαίνεται ότι το κόστος της ένθετης συνάρτησης δεν χρησιμοποιείται απευθείας στα περάσματα προς

```
[ ]  import tensorflow.keras as keras
```

```
[ ]  import tensorflow as tf
```

```
[ ]  mnist = tf.keras.datasets.mnist
     (x_train, y_train),(x_test, y_test) = mnist.load_data()
```

```
[ ]  print(x_train[0])
```

```
[ ]  import matplotlib.pyplot as plt

     plt.imshow(x_train[0],cmap=plt.cm.binary)
     plt.show()
```

```
[ ]  print(y_train[0])
```

```
[ ]  x_train = tf.keras.utils.normalize(x_train, axis=1)
     x_test = tf.keras.utils.normalize(x_test, axis=1)
```

```
[ ]  print(x_train[0])
```

```
[ ]  plt.imshow(x_train[0],cmap=plt.cm.binary)
     plt.show()
```

```
[ ]  model = tf.keras.models.Sequential()
     model.add(tf.keras.layers.Flatten())
     model.add(tf.keras.layers.Dense(128, activation=tf.nn.relu))
     model.add(tf.keras.layers.Dense(128, activation=tf.nn.relu))
     model.add(tf.keras.layers.Dense(10, activation=tf.nn.softmax))
```

Σχήμα 3.3: Πρώτο μέρος του κώδικα

τα εμπρός και προς τα πίσω και καλείται σε κάθε επανάληψη της μεθόδου της στοχαστικής κλίσης ώστε να μπορεί να παρακολουθείται η πρόοδο της εκπαίδευσης. Ενώ η συνάρτηση ενεργοποίησης κάθε νευρώνα είναι η *ReLU* στις εικόνες κώδικα.

```
[ ]  model.compile(optimizer='adam',
                   loss='sparse_categorical_crossentropy',
                   metrics=['accuracy'])

     model.fit(x_train, y_train, epochs=3)
```

```
[ ]  val_loss, val_acc = model.evaluate(x_test, y_test)
     print(val_loss)
     print(val_acc)
```

```
[ ]  model.save('epic_num_reader.model')
```

```
[ ]  new_model = tf.keras.models.load_model('epic_num_reader.model')
```

```
[ ]  predictions = new_model.predict(x_test)
```

```
[ ]  print(predictions)
```

```
[ ]  import numpy as np

     print(np.argmax(predictions[0]))
```

```
[ ]  plt.imshow(x_test[0],cmap=plt.cm.binary)
     plt.show()
```

Σχήμα 3.4: Δεύτερο μέρος του κώδικα

Κεφάλαιο 4

Συνελικτικά Νευρωνικά Δίκτυα

Τα συνελικτικά νευρωνικά δίκτυα είναι δίκτυα βιολογικά εμπνευσμένα που χρησιμοποιούνται στην όραση του υπολογιστή για ταξινόμηση εικόνων και ανίχνευση αντικειμένων. Το βασικό κίνητρο για το συνελικτικό νευρωνικό δίκτυο αποκτήθηκε από την κατανόηση των λειτουργιών του οπτικού φλοιού της γάτας, όπου συγκεκριμένα τμήματα του οπτικού πεδίου φάνηκαν να διεγείρουν συγκεκριμενους νευρώνες. Αυτή η ευρύτερη αρχή χρησιμοποιήθηκε για να σχεδιάσει μια αραιή αρχιτεκτονική για συνελικτικά νευρωνικά δίκτυα. Η πρώτη βασική αρχιτεκτινική βασισμένη σε αυτή τη βιολογική έμπνευση ήταν το *neocognitron*, το οποίο στην συνέχεια γενικεύτηκε στην αρχιτεκτονική *LeNet* – 5. Ένα δίκτυο που τα πρώτα χρόνια δημιουργίας του χρησιμοποιήθηκε από αρκετές τράπεζες για να αναγνωρίζει τους χειρόγραφους αριθμούς στις επιταγές.

Τα συνελικτικά νευρωνικά δίκτυα είναι σχεδιασμένα να δουλεύουν με εισόδους που εξαρτώνται από τοπικές περιοχές στην είσοδο του πλέγματος του δυκτίου. Το πιο προφανές παράδειγμα δεδομένων με δομή πλέγματος είναι μια δυσδιάστατη εικόνα. Αυτού του είδους τα δεδομένα παρουσιάζουν επίσης χωρική εξάρτηση, καθώς γειτονικές περιοχές του χώρου σε μια εικόνα συνήθως έχουν παρεμφερές χρώμα, που δημιουργεί μια τρισδιάτατη είσοδο όγκου. Επομένως τα χαρακτηριστικά σε ένα συνελικτικό νευρωνικό δίκτυο παρουσιάζουν αλληλεξαρτήσεις με βάση τις χωρικές αποστάσεις. Άλλες μορφές χωρικών δεδομένων όπως το κείμενο, χρονικές σειρές και ακολουθίες μπορούν επίσης να θεωρηθούν ως χωρικές περιπτώσεις πλεγματικά δομημένων δεδομένων με διαφορετικού τύπου σχέσεις μεταξύ γειτονικών αντικειμένων. Το μεγαλύτερο ποσοστό των εφαρμογών των συνελικτικών νευρωνικών δικτύων αφορούν δεδομένα εικόνων. Παρόλα αυτά μπορεί κανείς να χρησιμοποιήσει τα δίκτυα αυτά για όλους τους τύπους χρονικών χωρικών και χωρο-χρονικών δεδομένων.

Μια σημαντική ιδιότητα των δεδομένων εικόνας είναι ότι παρουσιάζουν ένα ορισμένο επίπεδο μεταθετικής αμεταβλητότητας κάτι που δεν συμβαίνει σε πολλούς άλλους τύπους δεδομένων με δομή πλέγματος. Για παράδειγμα, ένα μήλο έχει την ίδια ερμηνεία, είτε είναι στο πάνω μέρος της εικόνας είτε στο κάτω. Τα συνελικτικά νευρωνικά δίκτυα τείνουν να δημιουργούν παρόμοιες τιμές χαρακτηριστικών από τοπικές περιοχές με παρόμοια μοτίβα. Ένα πλεονέκτημα των δεδομένων εικόνας είναι ότι οι επιδράσεις συγκεκριμένων εισόδων στις αναπαραστάσεις χαρακτηριστικών μπορούν συχνά να περιγράφουν με διαισθητικό τρόπο.

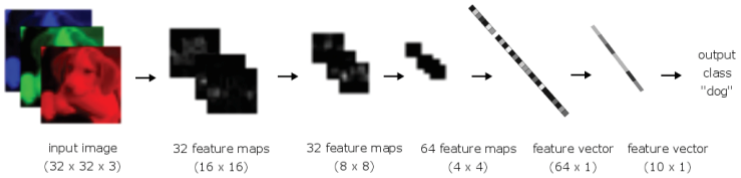
4.1 Η βασική δομή ενός Συνελικτικού Δικτύου

Στα συνελικτικά νευρωνικά δίκτυα, οι καταστάσεις σε κάθε επίπεδο είναι διατεταγμένες σύμφωνα με μια χωρική δομή πλέγματος. Αυτές οι χωρικές σχέσεις κληρονομούνται από το ένα στρώμα στο επόμενο επειδή κάθε τιμή χαρακτηριστικού βασίζεται σε μια τοπική χωρική περιοχή στο προηγούμενο στρώμα. Είναι σημαντικό να διατηρηθούν αυτές οι χωρικές σχέσεις μεταξύ των κελιών του πλέγματος, επειδή η διαδικασία συνέλιξης και ο μετασχηματισμός στο επόμενο στρώμα εξαρτώνται σημαντικά από αυτές τις σχέσεις. Κάθε στρώμα του συνελικτικού δικτύου είναι μια τρισδιάστατη δομή πλέγματος, η οποία έχει ύψος, πλάτος και βάθος. Το βάθος ενός στρώματος σε ένα συνελικτικό νευρωνικό δίκτυο δεν πρέπει να συγχέεται με το βάθος του ίδιου του δικτύου. Η λέξη ¨βάθος¨ αναφέρεται στον αριθμό των καναλιών σε κάθε στρώμα, όπως ο αριθμός των κύριων καναλιών χρώματος στην εικόνα εισόδου ή αριθμός χαρτών χαρακτηριστικών στα κρυφά στρώματα. Αυτή η ορολογία όμως είναι μια ατυχής υπερφόρτωση εννοιολογικά τόσο για τον αριθμό των χαρτών χαρακτιριστικών όσο και για τον αριθμό των στρωμάτων.

Το συνελικτικό νευρωνικό δίκτυο λειτουργεί σαν ένα παραδοσιακό εμπροσθόδοτο νευρωνικό δίκτυο. Η διαφορά τους έγκειται στο γεγονός ότι σε ένα συνελικτικό νευρωνικό δίκτυο οι πράξεις στα στρώματα του είναι χωρικά οργανωμένες με αραιές συνδέσεις μεταξύ των στρωμάτων. Οι τρεις τύποι στρωμάτων που υπάρχουν συνήθως σε ένα συνελικτικό νευρωνικό δίκτυο είναι η συνέλιξη(*convolution*), η ομαδοποίηση(*pooling*) και το (*ReLU*). Η ενεργοποίηση (*ReLU*) δεν διαφέρει από το παραδοσιακό νευρωνικό δίκτυο. Επιπλέον, ένα τελικό σύνολο στρωμάτων είναι συχνά πλήρως συνδεδεμένο και χαρτογραφείται με τρόπο συγκεκριμένο ανάλογα με την εκάστοτε εφαρμογή σε ένα σύνολο κόμβων εξόδου.

Τα δεδομένα εισόδου στο συνελικτικό νευρωνικό δίκτυο είναι οργανωμένα σε μία δυσδιάστατη δομή πλέγματος και αναφερόμαστε στις τιμές των επιμέρους σημείων πλέγματος ως εικονοστοιχεία. Κάθε εικονοστοιχείο αντιστοιχεί σε μία χωρική θέση μέσα στην εικόνα. Ωστόσο για να κωδικοποιήσουμε το ακριβές χρώμα του εικονοστοιχείου, χρειαζόμαστε έναν πολυδιάστατο πίνακα τιμών σε κάθε θέση του δικτύου. Στο σύστημα *RGB*, έχουμε ένταση των τριών βασικών χρωμάτων, που αντιστοιχούν στο κόκκινο, το πράσινο και το μπλε, αντίστοιχα. Επομένως εάν οι χωρικές διαστάσεις μιας εικόνας είναι 32×32 εικονοστοιχεία και το βάθος είναι 3, τότε ο συνολικός αριθμός εικονοστοιχείων στην εικόνα είναι $32\times 32 \times 3$. Αυτό το συγκεκριμένο μέγεθος εικόνας είναι αρκετά σύνηθες και εμφανίζεται σε ένα ευρέως χρησιμοποιούμενο σύνολο δεδομένων για συγκριτική αξιολόγηση, γνωστό ως *CIFAR* − 10. Ένα παράδειγμα αυτής της οργάνωσης παρουσιάζεται στο Σχήμα 4.1. Είναι λογικό να χρησιμοποιούμε αυτήν την τρισδιάστατη δομή για το στρώμα εισόδου, επειδή οι δύο διαστάσεις αντιστοιχούν στις χωρικές σχέσεις και η τρίτη διάσταση αντιστοιχεί στις ανεξάρτητες ιδιότητες σε αυτά τα κανάλια. Για παράδειγμα οι εντάσεις των βασικών χρωμάτων είναι οι ανεξάρτητες ιδιότητες στο πρώτο στρώμα. Στα κρυφά στρώματα αυτές οι ανεξάρτητες ιδιότητες αντιστοιχούν σε διάφορους τύπους σχημάτων που εξάγονται από τοπικές περιοχές της εικόνας. Ας υποθέσουμε ότι η είσοδος στο στρώμα τάξης q έχει μέγεθος $L_q\times B_q\times \delta_q$. Εδώ το L_q αντιστοιχεί στο ύψος ή το μήκος, το B_q αντιστοιχεί στο πλάτος ή εύρος και το d_q είναι το βάθος. Σε όλες σχεδόν τις εφαρμογές με βάση την εικόνα οι τιμές L_q και B_q είναι οι ίδιες.

Για το πρώτο στρώμα (είσοδος), αυτές οι τιμές καθορίζονται από τη φύση των δεδομένων



Σχήμα 4.1: Παράδειγμα σάρωση εικονοστοιχείων από συνελικτικά νευρωνικά δίκτυα

εισόδου και από την προεπεξεργασία τους. Στο παραπάνω παράδειγμα οι τιμές είναι $L_1 = 32, B_1 = 32$ και $d_1 = 3$. Τα επόμενα στρώματα έχουν ακριβώς την ίδια τρισδιάστατη οργάνωση με τη διαφορά ότι κάθε ένα από τα d_q πλέγματα τιμών δύο διαστάσεων για μια συγκεκριμένη είσοδο δεν μπορεί να θεωρείται πλέον ένα πλέγμα μη-επεξεργασμένων εικονοστοιχείων. Επιπλέον η τιμή του d_q είναι πολύ μεγαλύτερη από τρεις κρυμμένες στρώσεις επειδή ο αριθμός των ανεξάρτητων ιδιοτήτων μια συγκεκριμένης περιοχής που σχετίζονται με την ταξινόμηση είναι αρκετά σημαντικός. Για $q > 1$ αυτά τα πλέγματα τιμών αναφέρονται ως χάρτες χαρακτηριστικών ή χάρτες ενεργοποίησης. Αυτές οι τιμές είναι ανάλογες με τις τιμές στα κρυμμένα στρώματα σε ένα εμπρόσθόδοτο δίκτυο.

4.2 Συνέλιξη

Σε ένα συνελικτικό νευρωνικό δίκτυο οι παράμετροι είναι οργανωμένοι σε ομάδες τρισδιάστατων δομικών μονάδων οι οποίες είναι γνωστές ως φίλτρα ή πυρήνες *kernels*. Το φίλτρο είναι συνήθως τετράγωνο ως προς τις διαστάσεις του στο χώρο οι οποίες διαστάσεις είναι συνήθως πολύ μικρότερες από αυτές του στρώματος στο οποίο εφαρμόζεται το φίλτρο. Ωστόσο το βάθος ενός φίλτρου είναι πάντα ίδιο με το βάθος του στρώματος στο οποίο εφαρμόζεται.

Η διαδικασία της συνέλιξης τοποθετεί το φίλτρο σε κάθε πιθανή θέση στην εικόνα, έτσι ώστε το φίλτρο να επικαλύπτει πλήρως την εικόνα και να πραγματοποιεί ένα εσωτερικό γινόμενο μεταξύ των παραμέτρων του φίλτρου και του αντίστοιχου πλέγματος εισόδου. Το εσωτερικό γινόμενο εκτελείται με επεξεργασία των δεδομένων στην αντίστοιχη τρισδιάστατη περιοχή του όγκου εισόδου και του φίλτρου ως διανύσματα διαστάσεων, έτσι ώστε τα στοιχεία και των δύο διανυσμάτων να ταξινομούνται με βάση τις αντίστοιχες θέσεις τους στο πλέγμα.

Για τα επίπεδα της συνέλιξης, ορίζεται μια λειτουργία συνέλιξης, στην οποία χρησιμοποιείται ένα φίλτρο για τη χαρτογράφηση των ενεργοποιήσεων από ένα επίπεδο στο επόμενο. Μια λειτουργία συνέλιξης χρησιμοποιεί ένα τρισδιάστατο φίλτρο βαρών με το ίδιο βάθος με το τρέχον επίπεδο, αλλά με μικρότερο χωρικό εύρος. Το εσωτερικό γινόμενο μεταξύ όλων των βαρών στο φίλτρο, κάθε επιλογή χωρικής περιοχής σε ένα επίπεδο, ορίζει την τιμή κρυφής κατάστασης στο επόμενο επίπεδο. Η λειτουργία μεταξύ του φίλτρου και των χωρικών περιοχών σε ένα επίπεδο εκτελείται σε κάθε δυνατή θέση προκειμένου να οριστεί το επόμενο επίπεδο.

Για παράδειγμα:

$$\begin{pmatrix} 1 & -1 & & & & \\ & 1 & -1 & & & \\ & & 1 & -1 & & \\ & & & 1 & -1 & \\ & & & & 1 & -1 \end{pmatrix} \epsilon R^{5x6}$$

(4.1)

Αυτό παράγει ένα διάνυσμα στο R^5 που αποτελείται από διαφορές μεταξύ γειτονικών τιμών. Σε αυτή την περίπτωση χρησιμοποιούμε ένα φίλτρο $[1, -1]$ και ένα βήμα μήκους 1, το φίλτρο προχωρά κατά μέρος μετά από κάθε χρήση. Ύστερα από κατάλληλες γενικεύσεις αυτού του πίνακα στην περίπτωση εισόδου. Τα διανύσματα που προκύπτουν από δυσδιάστατες εικόνες μπορούν να χρησιμοποιηθούν για την ανίχνευση ακμών σε μια εικόνα επιστρέφοντας μια μεγάλη απόλυτη τιμή όταν υπάρχει απότομη αλλαγή στο γειτονικό εικονοστοιχείο. Η μετακίνηση ενός φίλτρου σε μια εικόνα μπορεί επίσης να αποκαλύψει άλλα χαρακτηριστικά, για παράδειγμα, συγκεκριμένους τύπους καμπυλών ή κηλίδων του ίδιου χρώματος. Έτσι,έχοντας ορίσει ένα φίλτρο μεγέθους και μήκους διασκελισμού,μπορούμε να επιτρέψουμε στη διαδικασία προπόνησης να μάθει τα βάρη στο φίλτρο ως μέσο εξαγωγής χρήσιμης δομής.

Η λέξη συνελικτικός προκύπτει επειδή οι γραμμικοί μετασχηματισμοί που εμπλέκονται μπορούν να γραφτούν με τη μορφή συνέλιξης. Στη μονοδιάστατη περίπτωση, η συνέλιξη του διανύσματος $x \epsilon R^p$ με το φίλτρο $g_{1-p}, g_{2-p}, ..., g_{p-2}, g_{p-1}$ έχει την k συνιστώσα δεδομένη με:

$$y_k = \sum_{n=1}^{p} x_n g_{k-n}$$

Το παράδειγμα στο (4.1) αντιστοιχεί σε ένα φίλτρο με $g_0 = 1, g_{-1} = -1$ και όλα τα άλλα $g_k = 0$.Στην περίπτωση αυτή έχουμε:

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix} = \begin{pmatrix} a & b & c & d & & & & & \\ & & a & b & c & d & & & \\ & & & & a & b & c & d & \\ & & & & & & a & b & c & d \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ 0 \end{pmatrix} \tag{4.2}$$

Εφαρμόζουμε ένα φίλτρο με τέσσερα βάρη, a, b, c και d, χρησιμοποιώντας μήκος διασκελισμού δύο. Επειδή η διάσταση του διανύσματος εισόδου x δεν είναι συμβατή με το μήκος του φίλτρου, έχουμε συμπληρώσει μια επιπλέον μηδενική τιμή.

Στην πράξη, τα δεδομένα εικόνας συνήθως θεωρούνται ως τρισδιάστατο διάνυσμα το καθένα εικονοστοιχείο το οποίο έχει δύο χωρικές συντεταγμένες και μια τιμή κόκκινο/πράσινο/μπλε. Από αυτή την άποψη, το φίλτρο παίρνει τη μορφή ενός μικρού πίνακα που εφαρμόζεται διαδοχικά σε μπαλώματα του διανύσματος εισόδου και η αντίστοιχη πράξη συνέλιξης είναι πολυδιάστατη. Από υπολογιστική άποψη, ένα βασικό πλεονέκτημα του συνελικτικού νευρωνικού δικτύου είναι ότι το διάνυσμα που παράγουν τα βάρη που εμπλέκονται στις διελεύσεις προς τα εμπρός και προς τα πίσω μέσω του δικτύου μπορεί να είναι υπολογιστικά εξαιρετικά αποτελεσματικό χρησιμοποιώντας τεχνικές γρήγορου μετασχηματισμού.

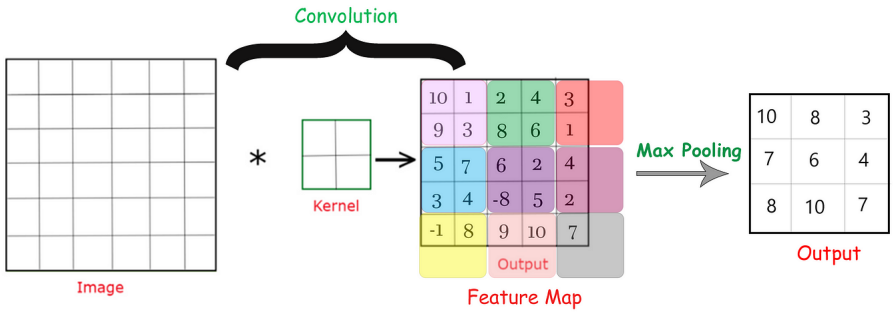
Οι συνδέσεις σε ένα συνελικτικό νευρωνικό δίκτυο είναι πολύ αραιές, επειδή οποιαδήποτε ενεργοποίηση σε ένα συγκεκριμένο επίπεδο είναι συνάρτηση μόνο μιας μικρής χωρικής περιοχής στο προηγούμενο επίπεδο. Όλα τα επίπεδα, εκτός από το τελικό σύνολο των δύο εκ των τριών επιπέδων, διατηρούν τη χωρική τους δομή. Επομένως, είναι δυνατό να απεικονιστούν χωρικά τα τμήματα της εικόνας που επηρεάζουν συγκεκριμένα τμήματα των ενεργοποιήσεων σε ένα επίπεδο. Τα χαρακτηριστικά σε επίπεδα χαμηλότερου επιπέδου συλλαμβάνουν γραμμές ή άλλα πρωτόγονα σχήματα, ενώ τα χαρακτηριστικά σε επίπεδα υψηλότερου επιπέδου καταγράφουν πιο περίπλοκα σχήματα όπως βρόχους. Συνεπώς τα μεταγενέστερα επίπεδα μπορούν να δημιουργήσουν ψηφία συνθέτοντας τα σχήματα σε αυτά τα διαισθητικά χαρακτηριστικά. Αυτό είναι ένα κλασικό παράδειγμα του τρόπου με τον οποίο χρησιμοποιούνται σημασιολογικές γνώσεις σχετικά με συγκεκριμένους τομείς δεδομένων για τον σχεδιασμό έξυπνων αρχιτεκτονικών.

Τα συνελικτικά νευρωνικά δίκτυα είναι ιστορικά από τα πιο επιτυχημένα όλων των τύπων των νευρωνικών δικτύων. Χρησιμοποιούνται ευρέως για την αναγνώριση εικόνων, την ανίχνευση/εντοπισμό αντικειμένων και ακόμη για την επεξεργασία κειμένου. Η απόδοση αυτών των δικτύων έχει υπερβεί πρόσφατα εκείνη των ανθρώπων στο πρόβλημα της ταξινόμησης των εικόνων. Τα συνελικτικά νευρωνικά δίκτυα παρέχουν ένα πολύ καλό παράδειγμα γεγονότος ότι οι επιλογές αρχιτεκτονικού σχεδιασμού σε ένα νευρωνικό δίκτυο θα πρέπει να εκτελούνται με σημασιολογική γνώση σχετικά με το συγκεκριμένο πεδίο δεδομένων.

4.3 Ομαδοποίηση

Η διαδικασία της ομαδοποίησης (*pooling*) είναι αρκετά διαφορετική. Η διαδικασία της ομαδοποίησης λειτουργεί σε μικρές περιοχές του πλέγματος μεγέθους $P_q \times \Pi_q$ σε κάθε στρώμα παράγει ένα άλλο στρώμα με το ίδιο βάθος. Για κάθε τετραγωνική περιοχή μεγέθους $P_q \times \Pi_q$ σε κάθε έναν από τους χάρτες ενεργοποίησης d_q επιστρέφει το μέγιστο αυτών των τιμών. Αυτή η προσέγγιση αναφέρεται ως μέγιστη ομαδοποίηση. Εάν χρησιμοποιείται βήμα με τιμή 1, τότε αυτό θα παράγει ένα νέο στρώμα μεγέθους $(L_q - P_q + 1) \times (B_q - P_q + 1) q d_q$. Ωστόσο είναι πιο συνηθισμένο να χρησιμοποιούμε ένα βήμα $S_q > 1$ στην ομαδοποίηση. Σε αυτές τις περιπτώσεις, το μήκος του νέου στρώματος θα είναι $(L_q - P_q / S_q + 1)$ και το πλάτος θα είναι $(B_q - P_q / S_q + 1)$. Συνεπώς η συγκέντρωση μειώνει δραστικά τις χωρικές διαστάσεις κάθε χάρτη ενεργοποίησης.

Σε αντίθεση με την διαδικασία της συνέλιξης, η ομαδοποίηση γίνεται στο επίπεδο κάθε χάρτη ενεργοποίησης. Μια διαδικασία συνέλιξης χρησιμοποιεί ταυτόχρονα όλους τους χάρτες χαρακτηριστικών d_q σε συνδυασμό με ένα φίλτρο για να παράγει μια ενιαία



Σχήμα 4.2: Παράδειγμα μέγιστης ομαδοποίησης ενός χάρτη ενεργοποίησης μεγέθους 3χ3 με βήματα 1 και 2

τιμή χαρακτηριστικών, ενώ η ομαδοποίηση λειτουργεί ανεξάρτητα σε κάθε χάρτη χαρακτηριστικών για την παραγωγή ενός άλλου χάρτη χαρακτηριστικών. Επομένως η διαδικασία ομαδοποίησης δεν αλλάζει τον αριθμό χαρτών χαρακτηριστικών. Με άλλα λόγια το βάθος του στρώματος που δημιουργείται με την ομαδοποίηση είναι το ίδιο με αυτό του στρώματος στο οποίο εκτελέστηκε η διαδικασία της ομαδοποίησης.

Παράδειγμα ομαδοποίησης με βήματα 1 και 2 φαίνονται στο Σχήμα 4.2. Εδώ χρησιμοποιούμε την ομαδοποίηση για περιοχές 3×3. Το τυπικό μέγεθος d_q της περιοχής πάνω από το οποίο πραγματοποιείται η ομαδοποίηση είναι 2×2. Για βήμα 2 δεν θα υπάρχει αλληλοεπικάλυψη μεταξύ των διαφορετικών περιοχών της ομαδοποίησης και είναι αρκετά σύνηθες η χρήση αυτού του τύπου ρύθμισης. Γενικά, όπως έχει υποδειχθεί είναι επιθυμητό να υπάρχει τουλάχιστον κάποια επικάλυψη μεταξύ των χωρικών μονάδων στις οποίες πραγματοποιείται η ομαδοποίηση, επειδή καθιστά την προσέγγιση λιγότερο πιθανή να υπερκαλυφθεί.

4.4 Αποφυγή Υπερπροσαρμογής

Τα νευρωνικά δίκτυα είναι ευφυείς μαθητές που έχουν επανειλημμένα αποδειχθεί ικανοί να μαθαίνουν πολύπλοκες συναρτήσεις σε πολλούς τομείς. Ωστόσο η μεγάλη δύναμη των νευρωνικών δικτύων είναι επίσης η μεγαλύτερη αδυναμία τους. Τα νευρωνικά δίκτυα απλώς υπερφορτώνουν τα δεδομένα εκπαίδευσης αν δεν φροντίσουμε για τον προσεκτικό σχεδιασμό της μαθησιακής διαδικασίας. Σε πρακτικό επίπεδο, υπερπροσαρμογή σημαίνει ότι ένα νευρωνικό δίκτυο θα παρέχει εξαιρετική απόδοση πρόβλεψης στα δεδομένα εκπαίδευσης πάνω στα οποία είναι χτισμένο, αλλά θα έχει κακή απόδοση πρόβλεψης σε περιπτώσεις δοκιμών που δεν έχει ξαναδεί. Αυτό οφείλεται στο γεγονός ότι η διαδικασία μάθησης συχνά θυμάται τα τυχαία αντικείμενα των δεδομένων εκπαίδευσης που δεν γενικεύονται καλά στα δεδομένα των δοκιμών. Οι ακραίες μορφές υπερπροσαρμογής αναφέρονται ως απομνημόνευση. Μια χρήσιμη παρομοίωση με την λειτουργία των ανθρώπων είναι η σκέψη ενός παιδιού που μπορεί να λύσει όλα τα αναλυτικά προβλήματα για τα οποία έχει δει τις λύσεις αλλά δεν είναι σε θέση να δώσει χρήσιμες λύσεις σε ένα νέο πρόβλημα. Ωστόσο, αν το παιδί εκτίθεται σε λύσεις όλο και περισσότερων διαφορετικών τύπων προβλημάτων, αυτός ή αυτή θα

είναι πιο πιθανό να λύσει ένα νέο πρόβλημα, αποσπώντας την ουσία των μοτίβων που επαναλαμβάνονται στα διάφορα προβλήματα και τις λύσεις του. Από την άλλη πλευρά, μία πρόβλεψη που βασίζεται σε μοτίβα σε ένα μικροσκοπικό σύνολο δεδομένων εκπαίδευσης δύο μηνυμάτων ηλεκτρονικού ταχυδρομείου, θα οδηγήσει σε καλή απόδοση σε αυτά τα μηνύματα, αλλά όχι σε νέα μηνύματα. Η ικανότητα ενός μαθητευόμενου να παρέχει προβλέψεις για περιπτώσεις που δεν έχει δει μέχρι τώρα αναφέρεται ως γενίκευση.

Η γενίκευση είναι μία χρήσιμη πρακτική ιδιότητα και ως εκ τούτου είναι η πανάκεια σε όλες τις εφαρμογές μηχανικής μάθησης. Εξάλλου, εάν τα παραδείγματα εκπαίδευσης έχουν ήδη επισημανθεί, δεν υπάρχει πρακτική χρήση της πρόβλεψης τέτοιων παραδειγμάτων ξανά. Για παράδειγμα σε μία εφαρμογή υπότιτλων εικόνας, πάντα προσπαθεί κανείς να χρησιμοποιήσει τις ετικέτες εικόνων για να μάθει λεζάντες για τις εικόνες που ο μαθητευόμενος βλέπει για πρώτη φορά.

Το επίπεδο της υπερπροσαρμογής εξαρτάται τόσο από την πολυπλοκότητα του μοντέλου όσο και από τον όγκο των διαθέσιμων δεδομένων. Η πολυπλοκότητα του μοντέλου που ορίζεται από ένα νευρωνικό δίκτυο εξαρτάται από τον αριθμό των υποκειμένων παραμέτρων. Οι παράμετροι παρέχουν πρόσθετους βαθμούς ελευθερίας, οι οποίοι μπορούν να χρησιμοποιηθούν για την εξήγηση συγκεκριμένων σημείων δεδομένων εκπαίδευσης χωρίς να γενικεύουν καλά σε σημεία που δεν έχουν ξαναδεί.

Υπάρχουν πολλά ενδεικτικά σημάδια υπερπροσαρμογής. Ένα από αυτά είναι όταν ένα μοντέλο εκπαιδεύεται σε διαφορετικά σύνολα δεδομένων, η ίδια δοκιμαστική περίπτωση μπορεί να λάβει πολύ διαφορετικές προβλέψεις. Αυτό είναι ένα σημάδι ότι η διαδικασία εκπαίδευσης απομνημονεύει τις διαφοροποιήσεις του συγκεκριμένου συνόλου δεδομένων εκπαίδευσης, αντί να μάθει τα μοτίβα που γενικοποιούνται σε περιπτώσεις δοκιμών που δεν έχει ξανασυναντήσει. Κάτι τέτοιο βέβαια συμβαίνει όταν το μοντέλο εκπαίδευσης είναι πολυωνυμικό και όχι απλό γραμμικό.

Εξίσου σημαντικό είναι το χάσμα μεταξύ σφάλματος πρόβλεψης των περιπτώσεων εκπαίδευσης και των περιπτώσεων δοκιμής που δεν έχουμε ξανασυναντήσει να είναι αρκετά μεγάλο. Συνήθως το σφάλμα για το πολυωνυμικό μοντέλο είναι πάντα 0 ενώ για ένα γραμμικό μοντέλο είναι πάντα μη μηδενικό.

Λόγω των μεγάλων χασμάτων μεταξύ εκπαίδευσης και δοκιμαστικού σφάλματος, τα μοντέλα συχνά δοκιμάζονται σε τμήματα των δεδομένων εκπαίδευσης που δεν έχουν ξαναδεί. Αυτά τα τμήματα των δεδομένων δοκιμών που δεν έχουν ξανασυναντήσει εκτυλίσσονται συχνά νωρίς και στη συνέχεια χρησιμοποιούνται για να κάνουν διαφορετικούς τύπους αλγοριθμικών αποφάσεων όπως η ρύθμιση των παραμέτρων εκπαίδευσης. Αυτό το σύνολο σημείων αναφέρεται ως σύνολο επικύρωσης. Η τελική ακρίβεια δοκιμάζεται σε ένα σύνολο από τα σημεία εκτός δείγματος που δεν έχουν χρησιμοποιηθεί για την οικοδόμηση μοντέλου ή για ρύθμιση παραμέτρων. Το σφάλμα στα δεδομένα δοκιμών εκτός δείγματος αναφέρεται επίσης ως σφάλμα γενίκευσης.

Τα νευρωνικά δίκτυα είναι μεγάλα και ενδέχεται να έχουν εκατομμύρια παραμέτρους σε πολύπλοκες εφαρμογές. Παρά τις προκλήσεις αυτές, υπάρχουν πολλά τεχνάσματα που μπορεί κανείς να χρησιμοποιήσει για να εξασφαλίσει ότι η υπερπροσαρμογή δεν αποτελεί πρόβλημα. Η επιλογή της μεθόδου εξαρτάται απ'τη συγκεκριμένη ρύθμιση και τον τύπο του χρησιμοποιούμενου νευρωνικού δικτύου. Οι βασικές μέθοδοι για την αποφυγή υπερπροσαρμογής σε ένα νευρωνικό δίκτυο αναφέρονται παρακάτω:

• Κανονικοποίηση βάσει ποινής: Η κανονικοποίηση βάσει ποινής είναι η πιο κοινή τεχνική που χρησιμοποιείται από τα νευρωνικά δίκτυα για να αποφευχθεί η υπερπροσαρμογή. Η ιδέα στην κανονικοποίηση είναι να δημιουργηθεί μια ποινή ή άλλοι τύποι περιορισμών στις παραμέτρους προκειμένου να ευνοηθούν τα απλούστερα μοντέλα. Για παράδειγμα, στην περίπτωση πολυωνυμικής παλινδρόμησης, ένας πιθανός περιορισμός των παραμέτρων θα ήταν να εξασφαλιστεί ότι το πολύ k διαφορετικές τιμές του w_i είναι μη μηδενικές. Αυτό θα εξασφαλίσει απλούστερα μοντέλα. Ωστόσο δεδομένου ότι είναι δύσκολο να επιβληθούν ρητά τέτοιοι περιορισμοί μια απλούστερη προσέγγιση είναι να επιβληθεί μια πιο ήπια ποινή, όπως η $l \sum_{i=0}^{d} w_i^2$ και να την προσθέσουμε στην συνάρτηση απώλειας. Μια τέτοια προσέγγιση αντιστοιχεί περίπου στον πολλαπλασιασμό κάθε παραμέτρου w_i με ένα πολλαπλασιαστικό συντελεστή αποσύνθεσης $(1\text{-}al)$ πριν από κάθε ενημέρωση με ρυθμό μάθησης a. Εκτός από την επιβολή ποινών στις παραμέτρους του δικτύου, μπορεί κανείς να επιλέξει να δώσει ποινή στις ενεργοποιήσεις των κρυφών μονάδων. Αυτή η προσέγγιση οδηγεί συχνά σε αραιές κρυφές αναπαραστάσεις.

• Έγκαιρη διακοπή: Κατά την έγκαιρη διακοπή η επαναληπτική μέθοδος βελτιστοποίησης τερματίζεται νωρίς, χωρίς να συγκλίνει προς την βέλτιστη λύση στα δεδομένα εκπαίδευσης. Το σημείο διακοπής προσδιορίζεται χρησιμοποιώντας ένα τμήμα των δεδομένων εκπαίδευσης που δεν χρησιμοποιείται για την κατασκευή μοντέλου. Τερματίζουμε όταν αρχίσει να αυξάνεται το σφάλμα στα παρακρατημένα δεδομένα. Παρόλο που αυτή η προσέγγιση δεν είναι η βέλτιστη για τα δεδομένα εκπαίδευσης, φαίνεται ότι αποδίδει καλά στα δεδομένα των δοκιμών επειδή το σημείο διακοπής καθορίζεται με βάση τα παρακρατημένα δεδομένα.

• Κανονικοποίηση L_1: Η χρήση της ποινής τετραγωνικής νόρμας είναι η πιο κοινή προσέγγιση για κανονικοποίηση. Ωστόσο είναι δυνατή η χρήση άλλων τύπων ποινών για τις παραμέτρους. Μία κοινή προσέγγιση είναι η L_1 κανοκοποίηση στην οποία το τετραγωνικό σφάλμα αντικαθίσταται με ποινή για το άθροισμα των απόλυτων τιμών των συντελεστών. Άρα η νέα αντικειμενική συνάρτηση είναι η εξής:

$$L = \sum (y - \hat{y}) + l \sum_{i=0}^{d} |w_i|$$

4.5 Εκπαίδευση Συνελικτικών Δικτύων

Η διαδικασία εκπαίδευσης ενός συνελικτικού νευρωνικού δικτύου χρησιμοποιεί τον αλγόριθμο οπισθοδιάδοσης. Υπάρχουν κυρίως τρεις τύποι στρωμάτων, που αντιστοιχούν στα στρώματα συνέλιξης, *ReLU* και μέγιστης ομαδοποίησης. Το *ReLU* είναι σχετικά απλό για τον αλγόριθμο οπισθοδιάδοσης, επειδή δεν διαφέρει από ένα παραδοσιακό νευρωνικό δίκτυο. Για την μέγιστη ομαδοποίηση χωρίς επικάλυψη μεταξύ των ομάδων, πρέπει να προσδιοριστεί ποια μονάδα είναι η μέγιστη τιμή σε μια ομάδα. Η μερική παράγωγος της απώλειας σε σχέση με την ομαδοποιημένη κατάσταση γυρνά πίσω στην μονάδα με μέγιστη τιμή. Όλες οι καταχωρίσεις, εκτός από τη μέγιστη καταχώρηση στο πλέγμα, θα λάβουν τιμή 0. Για περιπτώσεις στις οποίες οι ομάδες επικαλύπτονται, έστω ότι οι ομάδες είναι $P_1 \ldots P_r$ στις οποίες εμπλέκεται η μονάδα h με τις αντίστοιχες ενεργοποιήσεις $h_1 \ldots h_r$, στο επόμενο στρώμα. Αν η h είναι μέγιστη τιμή στην ομάδα P_i, τότε η κλίση της απώλειας σε σχέση με το h_i γυρνά πίσω στο h. Οι συνεισφορές

των διαφόρων επικαλυπτόμενων ομάδων προστίθενται για να υπολογιστεί η κλίση σε σχέση με τη μονάδα h. Επομένως ο αλγόριθμος οπισθοδιάδοσης μέσω της συνάρτησης μεγιστοποίησης και της συνάρτησης *ReLU* δεν είναι πολύ διαφορετική από εκείνη των παραδοσιακών νευρωνικών δικτύων.

Υπάρχουν πολλές άλλες ευρέως χρησιμοποιούμενες επιλογές και η σχετική απόδοσή τους όπου εξαρτάται από την εφαρμογή. Στη ρύθμιση ταξινόμησης εικόνας είναι σύνηθες να χρησιμοποιείται μια διορθωμένη γραμμική συνάρτηση της *ReLU* ως ενεργοποίηση.

$$\varphi(x) = \begin{cases} 0 & for \quad |x| \leq 0 \\ x & for \quad |x| > 0 \end{cases} \tag{4.3}$$

Στην περίπτωση που τα δεδομένα εκπαίδευσης $(x^i)_{i=1}^N$ προέρχονται από κατηγορίες με ετικέτα K, ως $l_i \epsilon (1, 2, ..., K)$ είναι η δεδομένη ετικέτα για το σημείο δεδομένων x^i. Ως εναλλακτική λύση στη τετραγωνική συνάρτηση κόστους θα μπορούσαμε να χρησιμοποιήσουμε μια προσέγγιση απώλειας καταγραφής *softmax*, ως εξής. Έστω η έξοδος $a^L(x^i) = u^i$ από το δίκτυο να πάρει τη μορφή ενός διανύσματος σε μορφή R^K έτσι ώστε το j συστατικό να είναι μεγάλο όταν πιστεύεται ότι η εικόνα είναι από κατηγορία j. Η λειτουργία *softmax* ενισχύει τα μεγάλα συστατικά και παράγει ένα διάνυσμα θετικών βαρών που αθροίζονται σε ενότητα, η οποία μπορεί να ερμηνευθεί ως πιθανότητες.

$$(u^i)_s \rightarrow \frac{e^{(u_s)^i}}{\sum_{j=1}^k e^{(u_j)^i}}$$

Στόχος είναι τώρα να επιβληθεί στη *softmax* τιμή για το σημείο εκπαίδευσης x^i τέτοια ώστε να είναι όσο το δυνατόν πιο κοντά στην ενότητα στο στοιχείο l_i, το οποίο αντιστοιχεί στη σωστή ετικέτα. Χρησιμοποιώντας λογαριθμικό παρά τετραγωνικό μέτρο του σφάλματος, φτάνουμε στη συνάρτηση κόστους.

$$-\sum_{i=1}^{N} log(\frac{e^{(u_{l_i})^i}}{\sum_{j=1}^k e^{(u_j)^i}}) \tag{4.4}$$

Κεφάλαιο 5

Παράδειγμα Εκπαίδευσης Ταξινόμησης Εικόνων

5.1 Παράδειγμα

Εμφανίζουμε τώρα αποτελέσματα για μια μηχανική μάθηση στην ταξινόμηση εικόνων. Για να το κάνουμε αυτό, βασιζόμαστε σε δεδομένα που είναι διαθέσιμα μέσω του ιστότοπου της *Google*. Χρησιμοποιήσαμε έτοιμα *script* κώδικα *Python* τα οποία είναι διαθέσιμα μέσω του ιστότοπου της *Kaggle*. Συγκεκριμένα, χρησιμοποιούμε την αρχιτεκτονική δικτύου και τις επιλογές παραμέτρων από έτοιμα παραδείγματα του *Kaggle*. Αναφερόμαστε στην τεκμηρίωση και το εκπαιδευτικό υλικό του *Kaggle* για τις λεπτές λεπτομέρειες και επικεντρωνόμαστε εδώ σε μερικά από τα μεγαλύτερα ζητήματα ταξινόμησης εικόνας.

Εξετάζουμε ένα σύνολο εικόνων, καθεμία από τις οποίες εμπίπτει ακριβώς σε μία από τις πέντε κατηγορίες: μαργαρίτες, πρικαλίδες, τριαντάφυλλα, ηλιοτρόπια, τουλίπες. Εμείς χρησιμοποιήσαμε δεδομένα με ετικέτα από τη δωρεάν διαθέσιμη συλλογή *Flowers*. Οι εικόνες 32 επί 32 εικονοστοιχεία, το καθένα με ένα κόκκινο, πράσινο και μπλε εικονοστοιχείο. Έτσι ένα κομμάτι των δεδομένων εκπαίδευσης αποτελείται από 32×32×3 = 3.072 τιμές. Χρησιμοποιούμε ένα σετ εκπαίδευσης από 3670 εικόνες και χρησιμοποιήσαμε 723 εικόνες ως σύνολο επικύρωσης. Έχοντας ολοκληρώσει τη βελτιστοποίηση και την εκπαίδευση του δικτύου, στη συνέχεια αξιολογούμε την απόδοσή του σε μια τυχαία εικόνα από το *Google* μιας κατηγορίας εκ των 5 λουλουδιών. Σημειώνουμε ότι περισσότερο Θα μπορούσαν να χρησιμοποιηθούν περίπλοκες στρατηγικές διασταυρούμενης επικύρωσης, όπου τα ίδια συνολικά δεδομένα χωρίζονται επανειλημμένα σε διακριτά σετ εκπαίδευσης, επικύρωσης και δοκιμών.

Ακολουθώντας την αρχιτεκτονική που χρησιμοποιείται από τα συνελικτικά νευρωνικά δίκτυα, ρυθμίσαμε ένα δίκτυο του οποίου τα επίπεδα χωρίζονται σε πέντε μπλοκ ως εξής. Εδώ περιγράφουμε τις διαστάσεις των εισόδων/εξόδων και των βαρών σε συμπαγή συμβολισμό διανυσμάτων. Φυσικά, τα διανύσματα αυτά θα μπορούσαν να μετασχηματιστούν σε αραιά διανύσματα και πίνακες προκειμένου να ταιριάζουν στο γενικό πλαίσιο των προηγούμενων κεφαλαίων, αλλά πιστεύουμε ότι ο συμβολισμός του διανύσματος είναι φυσικό σε αυτό το πλαίσιο και είναι συνεπής με τη σύνταξη των συνελικτικών νευρωνικών δικτύων.

Block1: Αποτελείται από ένα στρώμα συνέλιξης που ακολουθείται από ένα στρώμα συγκέντρωσης και ενεργοποίησης. Αυτό μετατρέπει την αρχική είσοδο 32×32×3 σε διάσταση 16×16×32. Περισσότερες λεπτομέρειες, το συνελικτικό στρώμα χρησιμοποιεί φίλτρα 5×5 που σαρώνουν επίσης κατά μήκος του δικτύου τρία έγχρωμα κανάλια. Υπάρχουν 32 διαφορετικά φίλτρα, οπότε συνολικά τα βάρη μπορούν να αναπαρασταθούν σε έναν πίνακα 5×5×3×32. Η έξοδος από κάθε φίλτρο μπορεί να περιγραφεί ως χάρτης χαρακτηριστικών. Τα φίλτρα εφαρμόζονται με μοναδιαίο βηματισμό. Με αυτόν τον τρόπο, καθένας από τους 32 χάρτες χαρακτηριστικών έχει διάσταση 32×32. Η μέγιστη συγκέντρωση στη συνέχεια εφαρμόστηκε σε κάθε χάρτη χαρακτηριστικών χρησιμοποιώντας το μήκος βήματος δύο. Αυτό μειώνει την διάσταση των χαρακτηριστικών και αντιστοιχίζεται σε 16×16. Στη συνέχεια χρησιμοποιείται μια ενεργοποίηση *ReLU*.

Block2: Εφαρμόζει συνέλιξη που ακολουθείται από ενεργοποίηση και στη συνέχεια από το στρώμα συγκέντρωσης. Αυτό μειώνει τη διάσταση σε 8×8×32. Πιο αναλυτικά, χρησιμοποιούμε 32 φίλτρα, καθένα από τα οποία είναι 5×5 στις διαστάσεις των χαρτών χαρακτηριστικών και επίσης σαρώνει κατά μήκος και τους 32 χάρτες χαρακτηριστικών. Έτσι τα βάρη θα μπορούσαν να θεωρηθούν ως πίνακας 5×5×32×32. Το μήκος του διασκελισμού είναι ένα, επομένως οι 32 χάρτες χαρακτηριστικών που προκύπτουν εξακολουθούν να υπάρχουν σε διάνυσμα διάστασης 16×16. Μετά την ενεργοποίηση *ReLU*, ένα μέσο επίπεδο συγκέντρωσης διασκελισμού εφαρμόζεται και μειώνει κάθε έναν από τους 32 χάρτες χαρακτηριστικών σε διάσταση 8×8.

Block3: Εφαρμόζει ένα στρώμα συνέλιξης που ακολουθείται από τη συνάρτηση ενεργοποίησης και στη συνέχεια εκτελεί μια λειτουργία ομαδοποίησης με τρόπο που μειώνει τη διάσταση σε 4×4×64. Πιο αναλυτικά εφαρμόζονται 64 φίλτρα. Κάθε φίλτρο είναι 5×5 στις διαστάσεις των χαρτών χαρακτηριστικών και επίσης σαρώνει και τους 32 χάρτες χαρακτηριστικών. Τα βάρη λοιπόν θα μπορούσαν να θεωρηθούν ως πίνακας 5×5×32×64. Ο διασκελισμός έχει μήκος ένα, με αποτέλεσμα χάρτες χαρακτηριστικών διαστάσεων 8×8. Μετά την ενεργοποίηση του *ReLU*, εφαρμόζεται το μέσο στρώμα συγκέντρωσης του βήματος δύο, το οποίο μειώνει το καθένα από 64 χάρτες χαρακτηριστικών σε διάσταση 4×4.

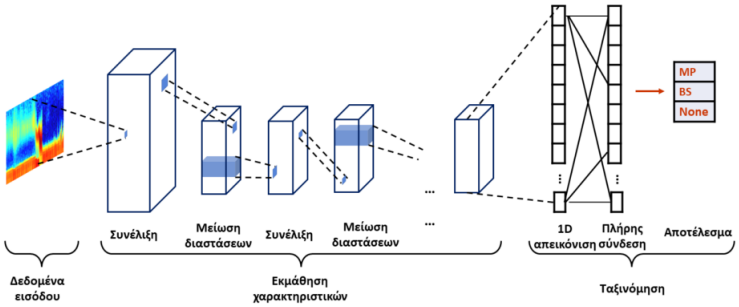
Block4: Δεν χρησιμοποιεί ομαδοποίηση, απλώς συνέλιξη ακολουθούμενη από ενεργοποίηση, που οδηγεί σε διάσταση 1×1×64. Πιο αναλυτικά χρησιμοποιούνται 64 φίλτρα. Κάθε φίλτρο είναι 4×4 στους 64 χάρτες χαρακτηριστικών, επομένως τα βάρη θα μπορούσαν να θεωρηθούν ως πίνακας 4×4×64×64 και κάθε φίλτρο παράγει έναν μόνο αριθμό.

Block5: Δεν περιλαμβάνει συνέλιξη. Χρησιμοποιεί ένα γενικό πλήρως συνδεδεμένο βάρος του τύπου που συζητήθηκε στα κεφάλαια 1 και 2 για να δώσει έξοδο διάστασης 1×1×10. Αυτό αντιστοιχεί σε μια μήτρα βάρους με διάσταση 10×64.

Softmax: Μια τελική λειτουργία *softmax* μετατρέπει καθένα από τα δέκα στοιχεία εξόδου στην περιοχή [0, 1].

Το Σχήμα 5.1 δίνει μια οπτική επισκόπηση της αρχιτεκτονικής του δικτύου.

Η έξοδος μας είναι ένα διάνυσμα πέντε πραγματικών αριθμών. Η συνάρτηση κόστους στο πρόβλημα βελτιστοποίησης παίρνει τη μορφή απώλειας καταγραφής *softmax* με



Σχήμα 5.1: Παρουσιάζονται τα στρώματα του συνελικτικού νευρωνικού δικτύου

$K = 10$. Καθορίζουμε στοχαστική κλίση με ορμή, η οποία χρησιμοποιεί έναν κινούμενο μέσο όρο τρέχουσας και προηγούμενης κλίσης κατευθύνσεις. Χρησιμοποιούμε μικρές παρτίδες μεγέθους 100 άρα *minibatch* = 100 και ορίζουμε έναν σταθερό αριθμό των 10 εποχών. Προκαθορίζουμε το ρυθμό μάθησης για κάθε εποχή: $\eta = 0,05$. Εκτελώντας με κάρτες γραφικών *GPU* με απλή ακρίβεια, οι 10 εποχές μπορούν να ολοκληρωθούν σε λίγα μόνο λεπτά.

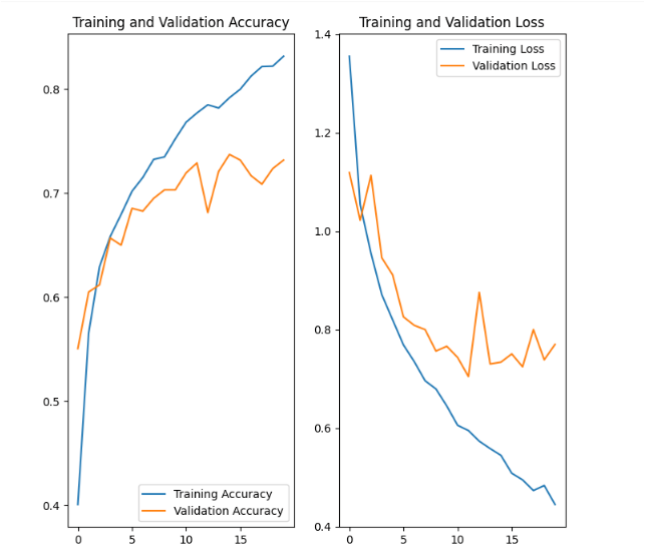
Όπως φαίνεται το διάγραμμα στο Σχήμα 5.2, επειδή η επιτυχία της μάθησης είναι αρκετά χαμηλή παρουσιάζεται πρόβλημα υπερπροσαρμογής στην εκπαίδευση του δικτύου. Για να αντιμετωπίσουμε αυτό το θέμα υπάρχουν αρκετές τεχνικές, όμως στο συγκεκριμένο πρόβλημα εφαρμόζεται η απαλοιφή των νευρώνων, έτσι ώστε οι πληροφορίες όπου δεν είναι και τόσο σπουδαίες για την εκπαίδευση να μην επηρεάζουν τόσο το αποτέλεσμα.

Σχήμα 5.2: Παρουσιάζονται διαγράμματα ακρίβειας για το σφάλμα εκπαίδευσης της μάθησης και για την επικύρωσή της.

Η γραφική παράσταση δεξιά στο Σχήμα 5.2 εξετάζει το ποσοστό των σφαλμάτων που λαμβάνουν χώρα όταν εμείς ταξινομούμε με την υψηλότερη πιθανότητα επιλογής. Ομοίως, η πλοκή στα δεξιά δείχνει το ποσοστό περιπτώσεων όπου η σωστή κατηγορία δεν είναι μεταξύ των πέντε πρώτων.Βλέπουμε από εικόνα 5.2 ότι το σφάλμα επικύρωσης αρχίζει να αυξάνεται σε ένα στάδιο όπου η στοχαστική μέθοδος *SGD* συνεχίζει να

κάνει σημαντικές μειώσεις στο σφάλμα εκπαίδευσης. Αυτό δίνει μια ένδειξη ότι είμαστε υπερβολικοί,μαθαίνοντας αρκετές λεπτές λεπτομέρειες σχετικά με την εκπαίδευση δεδομένων που δεν θα βοηθήσουν το δίκτυο να γενικευτεί σε αόρατα δεδομένα.

Τονίζουμε ότι σε αυτή την περίπτωση όλοι οι νευρώνες ενεργοποιούνται όταν το εκπαιδευμένο δίκτυο εφαρμόζεται στα δεδομένα της δοκιμής. Στο Σχήμα 5.3 απεικονίζουμε τη διαδικασία εκπαίδευσης σε περίπτωση απαλοιφής νευρώνων. Για τη γραφική παράσταση στα αριστερά, το μπλε χρώμα χρησιμοποιείται για να δείξει πώς μειώνεται η αντικειμενική συνάρτηση. Μετά από καθεμία από τις 10 εποχές χρησιμοποιούμε επίσης τα ίδια χρώματα για να υποδείξουμε την τιμή της αντικειμενικής συνάρτησης στα δεδομένα επικύρωσης. Ακριβέστερα, αυτά τα μέτρα σφάλματος υπολογίζονται κατά μέσο όρο οι μεμονωμένες παρτίδες που αποτελούν την εποχή και τα βάρη ενημερώνονται μετά από κάθε μία παρτίδα. Δεδομένου ότι ο γενικός μας στόχος είναι να αντιστοιχίσουμε εικόνες σε μία από τις πέντε κατηγορίες.



Σχήμα 5.3: Όπως και στο Σχήμα 5.2 αλλά στην περίπτωση που χρησιμοποιήθηκε η μέθοδος απαλοιφής και 20 εποχές εκπαίδευσης.

Το Σχήμα 5.3 δείχνει τα ανάλογα αποτελέσματα στην περίπτωση που χρησιμοποιείται η μέθοδος απαλοιφής με 20 εποχές αυτήν την φορά. Εμείς δείξαμε ότι τα σφάλματα εκπαίδευσης είναι σημαντικά μεγαλύτερα από αυτά στο Σχήμα 5.2 και τα σφάλματα επικύρωσης είναι παρόμοιου μεγέθους. Ωστόσο, δύο βασικά χαρακτηριστικά στην μέθοδο απαλοιφής: πρώτη περίπτωση είναι ότι (α) το σφάλμα επικύρωσης είναι κάτω από το σφάλμα εκπαίδευσης και (β) η επικύρωση στο σφάλμα συνεχίζει να μειώνεται σε συγχρονισμό με το σφάλμα εκπαίδευσης και τα δύο δείχνουν ότι η διαδικασία βελτιστοποίησης εξάγει χρήσιμες πληροφορίες για όλες τις εποχές.

Για να δώσουμε μια αίσθηση της δυσκολίας αυτής της εργασίας, η Εικόνα 5.4 δείχνει 9 εικόνες με τυχαία δειγματοληψία από αυτά που ταξινομήθηκαν στο επίπεδο εισόδου του δικτύου και θα γίνει επεξεργασία αυτών χωρίς απαλοιφή κάποιου νευρώνα.

Σχήμα 5.4: Εννέα τυχαίες εικόνες

5.2 Βιβλιοθήκες

Για την επίτευξη της μαθησιακής εργασίας της μηχανής, πάνω στο κομμάτι της όρασης του υπολογιστή εργαστήκαμε με κώδικα *Python*. Το περιβάλλον που γράφτηκε ο κώδικας ήταν το *Collaboration* της *Google* εν συντομία *Google – Collab*. Αυτό λοιπόν το περιβάλλον μας έδωσε την δυνατότητα να εργαστούμε με επαγγελματικά πρότυπα συνδέοντας βάσεις δεδομένων της *Google* με την μορφή του *Collab – Programming*. Τα πλεονεκτήματα αυτής της εργασίας είναι ότι δεν χρησιμοποιήσαμε καθόλου χώρο αποθήκευσης ενός υπολογιστή και τα δεδομένα που επεξεργαστήκαμε αποθηκεύτηκαν στο *Google Drive*, επίσης η εκπαίδευση της μηχανής έγινε χρησιμοποιώντας μια βάση δεδομένων του *Collaboration* της *Google* από όπου κάναμε και χρήση της κάρτας γραφικών *GPU*.

Η επεξεργασία των δεδομένων μας στο πρόγραμμα αλλά και η εκπαίδευση της μηχανής μας έγινε χρησιμοποιώντας βιβλιοθήκες της *Python*, όπως η *Numpy*, η *Matplotlib*, η *PIL*, η *Tensorflow* και η *Keras*. Περιγράφονται αναλυτικά παρακάτω.

Numpy: Η *numpy* είναι μια βιβλιοθήκη για τη γλώσσα προγραμματισμού *Python*, που προσθέτει υποστήριξη για μεγάλους, πολυδιάστατους πίνακες μαζί με μια μεγάλη συλλογή μαθηματικών συναρτήσεων υψηλού επιπέδου για λειτουργία περίπλοκων υπολογισμών ή μετασχηματισμών σε αυτούς τους πίνακες.

PIL: Η *Python Imaging Library* είναι μια δωρεάν και ανοιχτού κώδικα πρόσθετη βιβλιοθήκη για τη γλώσσα προγραμματισμού *Python* που προσθέτει υποστήριξη για άνοιγμα, χειρισμό και αποθήκευση πολλών διαφορετικών μορφών αρχείων εικόνας.

Matplotlib: Η *Matplotlib* είναι μια βιβλιοθήκη σχεδίασης για τη γλώσσα προγραμματισμού *Python* και την αριθμητική της επέκταση μαθηματικών *NumPy*. Παρέχει ένα αντικειμενοστραφή *API* για την ενσωμάτωση σχεδίων σε εφαρμογές χρησιμοποιώντας *kit* εργαλείων *GUI* γενικής χρήσης όπως *Tkinter*, *wxPython*, *Qt* ή *GTK*.

Tensorflow: Η *TensorFlow* είναι μια δωρεάν βιβλιοθήκη λογισμικού ανοιχτού κώδικα για μηχανική μάθηση και τεχνητή νοημοσύνη. Μπορεί να χρησιμοποιηθεί σε μια σειρά εργασιών, αλλά έχει ιδιαίτερη εστίαση στην εκπαίδευση και την εξαγωγή συμπερασμάτων σε βαθιά νευρωνικά δίκτυα.

Keras: Η *Keras* είναι μια βιβλιοθήκη ανοιχτού κώδικα που παρέχει μια διεπαφή *Python* για τεχνητά νευρωνικά δίκτυα. Η *Keras* λειτουργεί ως διεπαφή για τη βιβλιοθήκη *TensorFlow*.

Βιβλιογραφία

[1] Michael Nielsen , *Neural Networks and Deep Learning*, This book is accompanied by the above website. (www.deeplearningbook.org).

[2] Stuart Russell - Peter Norvig ,*Artificial Intelligence A Modern Approach*, Third Edition, This book is accompanied by the above website. (http://people.engr.tamu.edu).

[3] Charu C. Aggarwal , *Neural Networks and Deep Learning*, Fountas.

[4] Catherine F. Higham - Desmond J. Higham , *Deep Learning: An Introduction for Applied Mathematicians SIAM*, 2019 61(4) 860-891.

Παράρτημα Α΄

Παρουσίαση Κώδικα

Εδώ παρατίθονται εικόνες από τον κώδικα του πειράματος που αναφέρθηκε αναλυτικά στο κεφάλαιο 4. Αναλυτικά είναι 5 εικόνες οι οποίες παρουσιάζονται με την σειρά εκτέλεσης των εντολών για την επεξεργασία δεδομένων από εικόνες λουλουδιών που ανοίκουν σε 5 διαφορετικές κατηγορίες φυτών και την εκπαίδευση της μηχανής να μπορεί να ξεχωρίσει τι λουλούδι εμφανίζει η κάθε εικόνα.

```
 #ΒΙΒΛΙΟΘΗΚΕΣ
import matplotlib.pyplot as plt
import numpy as np
import PIL
import tensorflow as tf

from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras.models import Sequential
```

```
#ΚΑΤΕΒΑΣΜΑ ΤΩΝ ΦΩΤΟΓΡΑΦΙΩΝ ΤΩΝ ΛΟΥΛΟΥΔΙΩΝ
import pathlib

dataset_url = "https://storage.googleapis.com/download.tensorflow.org/example_images/flower_photos.tgz"
data_dir = tf.keras.utils.get_file('flower_photos.tar', origin=dataset_url, extract=True)
data_dir = pathlib.Path(data_dir).with_suffix('')
```

```
#ΑΡΙΘΜΟΣ ΕΙΚΟΝΩΝ
image_count = len(list(data_dir.glob('*/*.jpg')))
print(image_count)
```

```
3670
```

```
roses = list(data_dir.glob('roses/*'))
PIL.Image.open(str(roses[0]))
```

```
PIL.Image.open(str(roses[1]))
```

```
tulips = list(data_dir.glob('tulips/*'))
PIL.Image.open(str(tulips[0]))
```

```
PIL.Image.open(str(tulips[1]))
```

```
#ΔΗΜΙΟΥΡΓΕΙΑ DATA SETS
#ΟΡΙΣΜΟΣ ΠΑΡΑΜΕΤΡΩΝ
batch_size = 32
img_height = 180
img_width = 180
```

```
#ΔΙΑΧΩΡΙΣΜΟΣ ΕΙΚΟΝΩΝ ΚΑΤΑ ΤΗΝ ΑΝΑΠΤΥΞΗ ΤΟΥ ΜΕΝΤΕΛΟΥ. ΧΡΗΣΙΜΟΠΟΙΟΥΜΕ ΤΟ 80% ΤΩΝ ΕΙΚΟΝΩΝ ΓΙΑ ΕΣΚΠΑΙΔΕΥΣΗ ΚΑΙ ΤΟ 20% ΓΙΑ ΕΠΙΚΥΡΩΣΗ.
train_ds = tf.keras.utils.image_dataset_from_directory(
  data_dir,
  validation_split=0.2,
  subset="training",
  seed=123,
  image_size=(img_height, img_width),
  batch_size=batch_size)
```

Σχήμα Α΄.1

```
#ΕΔΩ ΕΜΦΑΝΙΖΟΝΤΑΙ 9 ΕΙΚΟΝΕΣ ΤΟΥ DATA SETS
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 10))
for images, labels in train_ds.take(1):
  for i in range(9):
    ax = plt.subplot(3, 3, i + 1)
    plt.imshow(images[i].numpy().astype("uint8"))
    plt.title(class_names[labels[i]])
    plt.axis("off")
```

```
for image_batch, labels_batch in train_ds:
  print(image_batch.shape)
  print(labels_batch.shape)
  break
```

```
#ΔΙΑΜΟΡΦΩΣΗ ΣΥΝΟΛΟΥ ΔΕΔΟΜΕΝΩΝ ΓΙΑ ΑΠΟΔΟΣΗ
AUTOTUNE = tf.data.AUTOTUNE

train_ds = train_ds.cache().shuffle(1000).prefetch(buffer_size=AUTOTUNE)
val_ds = val_ds.cache().prefetch(buffer_size=AUTOTUNE)
```

```
normalization_layer = layers.Rescaling(1./255)
```

```
normalized_ds = train_ds.map(lambda x, y: (normalization_layer(x), y))
image_batch, labels_batch = next(iter(normalized_ds))
first_image = image_batch[0]
print(np.min(first_image), np.max(first_image))
```

```
num_classes = len(class_names)

model = Sequential([
  layers.Rescaling(1./255, input_shape=(img_height, img_width, 3)),
  layers.Conv2D(16, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(32, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(64, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Flatten(),
  layers.Dense(128, activation='relu'),
  layers.Dense(num_classes)
])
```

```
model.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
              metrics=['accuracy'])
```

Σχήμα Α΄.2

```
epochs=10
history = model.fit(
  train_ds,
  validation_data=val_ds,
  epochs=epochs
)
```

```
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']

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

epochs_range = range(epochs)

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()
```

```
#ΑΥΞΗΣΗ ΔΕΔΟΜΕΝΩΝ
data_augmentation = keras.Sequential(
  [
    layers.RandomFlip("horizontal",
                      input_shape=(img_height,
                                   img_width,
                                   3)),
    layers.RandomRotation(0.1),
    layers.RandomZoom(0.1),
  ]
)
```

```
plt.figure(figsize=(10, 10))
for images, _ in train_ds.take(1):
  for i in range(9):
    augmented_images = data_augmentation(images)
    ax = plt.subplot(3, 3, i + 1)
    plt.imshow(augmented_images[0].numpy().astype("uint8"))
    plt.axis("off")
```

Σχήμα Α΄.3

```
model = Sequential([
  data_augmentation,
  layers.Rescaling(1./255),
  layers.Conv2D(16, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(32, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(64, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Dropout(0.2),
  layers.Flatten(),
  layers.Dense(128, activation='relu'),
  layers.Dense(num_classes, name="outputs")
])
```

```
model.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
              metrics=['accuracy'])
```

```
epochs = 20
history = model.fit(
  train_ds,
  validation_data=val_ds,
  epochs=epochs
)
```

```
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']

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

epochs_range = range(epochs)

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()
```

Σχήμα Α΄.4

```
[ ]  #ΠΡΟΒΛΕΨΗ ΜΕ ΝΕΑ ΔΕΔΟΜΕΝΑ ΠΟΥ ΔΕΝ ΑΝΟΙΚΟΥΝ ΣΤΟ DATA SETS
     sunflower_url = "https://storage.googleapis.com/download.tensorflow.org/example_images/592px-Red_sunflower.jpg"
     sunflower_path = tf.keras.utils.get_file('Red_sunflower', origin=sunflower_url)

     img = tf.keras.utils.load_img(
         sunflower_path, target_size=(img_height, img_width)
     )
     img_array = tf.keras.utils.img_to_array(img)
     img_array = tf.expand_dims(img_array, 0) # Create a batch

     predictions = model.predict(img_array)
     score = tf.nn.softmax(predictions[0])

     print(
         "This image most likely belongs to {} with a {:.2f} percent confidence."
         .format(class_names[np.argmax(score)], 100 * np.max(score))
     )
```

Σχήμα Α΄.5