ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
& ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΣΥΓΚΡΙΣΗ ΑΛΓΟΡΙΘΜΩΝ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ ΓΙΑ ΤΗΝ ΕΚΠΑΙΔΕΥΣΗ ΝΕΥΡΩΝΙΚΩΝ ΔΙΚΤΥΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

Ναυσικάς – Αναστασίας Περάκη

Επιβλέπων: Νικόλαος Πλόσκας

Αναπληρωτής Καθηγητής

ΚΟΖΑΝΗ/ΟΚΤΩΒΡΙΟΣ/2024

HELLENIC DEMOCRACY
UNIVERSITY OF WESTERN MACEDONIA
SCHOOL OF ENGINEERING
DEPARTMENT OF ELECTRICAL
& COMPUTER ENGINEERING
ece.uowm.gr

COMPARISON OF OPTIMIZATION ALGORITHMS FOR TRAINING NEURAL NETWORKS

THESIS

Nafsika-Anastasia Peraki

SUPERVISOR: Nikolaos Ploskas

Associate Professor

KOZANI/OCTOBER/2024

ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
& ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΔΗΛΩΣΗ ΜΗ ΛΟΓΟΚΛΟΠΗΣ ΚΑΙ ΑΝΑΛΗΨΗΣ ΠΡΟΣΩΠΙΚΗΣ ΕΥΘΥΝΗΣ

Δηλώνω ρητά ότι, σύμφωνα με το άρθρο 8 του Ν. 1599/1986 και τα άρθρα 2,4,6 παρ. 3 του Ν. 1256/1982, η παρούσα Διπλωματική Εργασία με τίτλο "**Σύγκριση αλγορίθμων βελτιστοποίησης για την εκπαίδευση Νευρωνικών Δίκτυων**" καθώς και τα ηλεκτρονικά αρχεία και πηγαίοι κώδικες που αναπτύχθηκαν ή τροποποιήθηκαν στα πλαίσια αυτής της εργασίας και αναφέρονται ρητώς μέσα στο κείμενο που συνοδεύουν, και η οποία έχει εκπονηθεί στο Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Πανεπιστημίου Δυτικής Μακεδονίας, υπό την επίβλεψη του μέλους του Τμήματος κ. Νικόλαου Πλόσκα αποτελεί αποκλειστικά προϊόν προσωπικής εργασίας και δεν προσβάλλει κάθε μορφής πνευματικά δικαιώματα τρίτων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον. Τα σημεία όπου έχω χρησιμοποιήσει ιδέες, κείμενο, αρχεία ή / και πηγές άλλων συγγραφέων, αναφέρονται ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή. Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα. Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τον συγγραφέα και μόνο.

Υπογραφή Φοιτητή:

Περίληψη

Η τεχνητή νοημοσύνη, η μηχανική μάθηση ως υποσύνολό της και ειδικότερα τα νευρωνικά δίκτυα έχουν γνωρίσει μία ραγδαία εξέλιξη τόσο στο θεωρητικό τους υπόβαθρο, όσο και στις εφαρμογές τους τις τελευταίες δεκαετίες. Στην παρούσα διπλωματική εργασία αφού εκτεθούν οι λόγοι που οδήγησαν στην εκπόνησή της, παρουσιάζονται σύντομα η σχέση τεχνητής ευφυίας και μηχανικής μάθησης και το κυριότερο υποσύνολο της δεύτερης, τα νευρωνικά δίκτυα. Χωρίς να υπάρχει σκοπός ταξινόμησης των νευρωνικών δικτύων, οι κυριότερες κατηγορίες τους, αναλύθηκαν, τόσο θεωρητικά όσο και μέσω εφαρμογής τους σε πραγματικά σύνολα δεδομένων, με τη διαδεδομένη και ελεύθερης χρήσης πλατφόρμα Keras, εξειδικευμένη στα νευρωνικά δίκτυα. Τα κυριότερα από αυτά τα νευρωνικά δίκτυα στο πρώτο θεωρητικό τμήμα αυτής της διπλωματικής είναι τα νευρωνικά δίκτυα εμπροσθο-διάδοσης και οπισθο-διάδοσης, του πολυεπίπεδου αναγνωριστή, τα συνελικτικά και τα επαναλαμβανόμενα ή αναδρομικά δίκτυα με διάφορες παραλλαγές. Επίσης, συνοπτικά εκτέθηκαν τα ακτινικά νευρωνικά δίκτυα, τα αρθρωτά νευρωνικά δίκτυα, οι κωδικοποιητές-αποκωδικοποιητές και κάποια άλλα. Οι μέθοδοι βελτιστοποίησης των νευρωνικών δικτύων αποτελούν το δεύτερο κύριο μέρος του θεωρητικού πλαισίου της διπλωματικής. Η Κάθοδος Κλίσης και διάφορες παραλλαγές της όπως αυτή σε μικρές δεσμίδες ή η στοχαστική κάθοδος κλίσης αποτελεί την κυρίαρχη μέθοδο βελτιστοποίησης των νευρωνικών δικτύων. Εξετάστηκαν και άλλες μέθοδοι όπως η Διάδοση Μέσου Τετραγωνικού Σφάλματος Ροπών, η Adam, η Adagrad, η RFTL, η LION και η Loss Scale Optimizer. Ίσως βέβαια το σημαντικότερο τμήμα της παρούσας διπλωματικής να αποτελεί το επόμενο τμήμα της, στο οποίο υλοποιούνται εφαρμογές διαφόρων νευρωνικών δικτύων και μεθόδων βελτιστοποίησης τους, σε πραγματικά δεδομένα, μέσω των επιλογών που παρέχει το Keras. Ακολουθεί αναλυτικός σχολιασμός και σύγκριση των αποτελεσμάτων που πάρθηκαν και τέλος η σύγκριση μεταξύ τους, που οδηγεί σε σχετικά συμπεράσματα για την αποτελεσματικότητα, την καταλληλότητα και τους χρόνους εκτέλεσης κάποιων νευρωνικών δικτύων και των μεθόδων βελτιστοποίησής τους.

Λέξεις Κλειδιά: Νευρωνικά δίκτυα, αλγόριθμοι βελτιστοποίησης νευρωνικών δικτύων, εκπαίδευση, επικύρωση, πρόβλεψη.

Abstract

Artificial intelligence, machine learning as its subset, and specifically neural networks, have seen rapid evolution both in theoretical foundations and applications in recent decades. This thesis first explains the reasons behind its creation, then presents a brief overview of the relationship between artificial intelligence and machine learning, as well as the key subset of the latter, neural networks. Without attempting to classify neural networks comprehensively, the main categories are analyzed both theoretically and through their application to real datasets using the widely popular and open-source platform Keras, which is specialized in neural networks. The main types of neural networks discussed in the first theoretical part of this thesis include feedforward and backpropagation neural networks, the multilayer perceptron, convolutional networks, and recurrent networks with their variations. Additionally, brief mention is made of radial basis function networks, modular neural networks, encoder-decoder architectures, and a few others. The optimization methods for neural networks comprise the second key part of the thesis' theoretical framework. Gradient Descent, along with its variations such as mini-batch gradient descent and stochastic gradient descent, is the dominant optimization method for neural networks. Other methods, such as Momentum RMSprop, Adam, and Adagrad, were also examined, along with similar techniques like FTRL, LION, and Loss Scale Optimizer. Perhaps the most significant part of this thesis is the subsequent section, which implements the application of various neural networks and their optimization methods on real datasets using the tools provided by Keras. Detailed commentary and comparison of the results obtained follow, along with a final comparison that leads to conclusions about the effectiveness, suitability, and execution times of certain neural networks and their optimization methods.

Keywords: Neural networks, Optimizers for neural networks, training, validation, prediction.

Ευχαριστίες

Θα ήθελα να εκφράσω τις ειλικρινείς μου ευχαριστίες σε όλους όσους συνέβαλαν στην ολοκλήρωση αυτής της διπλωματικής εργασίας. Πρώτα και κύρια, θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου, κ. Νικόλαο Πλόσκα, για την πολύτιμη καθοδήγηση, την υπομονή και την αμέριστη υποστήριξή του σε κάθε στάδιο αυτής της διαδικασίας. Οι συμβουλές και οι προτροπές του αποτέλεσαν σημαντική βοήθεια και έπαιξαν καθοριστικό ρόλο στην εξέλιξη και την επιτυχία αυτής της εργασίας. Επίσης, ένα μεγάλο ευχαριστώ αξίζει στην οικογένειά μου, η οποία στάθηκε στο πλευρό μου καθ' όλη τη διάρκεια των σπουδών μου. Η στήριξή τους, τόσο ηθική όσο και πρακτική, μου έδωσε τη δύναμη να συνεχίσω και να ολοκληρώσω τις σπουδές μου με επιτυχία. Ιδιαίτερα, θα ήθελα να ευχαριστήσω τον πατέρα μου, του οποίου η συνεχής πίστη σε εμένα, οι θυσίες και η αμέριστη υποστήριξή του υπήρξαν αναντικατάστατες στη διάρκεια αυτής της πορείας. Χωρίς τη δική του αγάπη και ενθάρρυνση, η ολοκλήρωση αυτής της εργασίας δεν θα ήταν δυνατή.

Περιεχόμενα

Περιεχόμενα Εικόνων

Περιεχόμενα Σχημάτων

1. Εισαγωγή

1.1. Ορισμός του προβλήματος

Από τους αρχαίους χρόνους μέχρι και σήμερα, στις θετικές επιστήμες και όχι αποκλειστικά, τόσο η ανακάλυψη, όσο και η βελτιστοποίηση μεθόδων -σε οποιονδήποτε τομέα- που θα καλυτέρευε την ανθρώπινη ζωή, υπήρξε ένας από τους θεμελιώδης λίθους στην πρόοδο της επιστήμης. Και ενώ στις θετικές κλασικές επιστήμες γενικά, όπως είναι η φυσική και τα μαθηματικά, οι προτεινόμενες ανακαλύψεις, διαφοροποιήσεις και βελτιστοποιήσεις αλγορίθμων, μεθόδων ή/και ολόκληρων μεθοδολογικών αλυσίδων, επιτελούνταν με έναν σχετικά λογικό χρονικό ρυθμό, οι αλματώδης αν όχι τεράστια εξέλιξη των Η/Υ και της πληροφορικής τις τελευταίες δεκαετίες οδήγησε και στην ραγδαία εξέλιξη νέων μεθόδων πληροφορικής-μαθηματικών, όσο και των αντίστοιχων μεθόδων βελτιστοποίησης τους. Η μεγάλη εξέλιξη που επιτελέστηκε στην Μηχανική Μάθηση αποτελεί φωτεινό παράδειγμα της προόδου αυτής. Για να υπάρξουν όσο το δυνατόν ακριβή και αξιόπιστα αποτελέσματα στον τομέα της μηχανικής - βαθιάς μάθησης, είναι απαραίτητο να βελτιστοποιηθούν οι παράμετροι του εκάστοτε προτεινόμενου μοντέλου. Ένας μεγάλος αριθμός αλγορίθμων βελτιστοποίησης έχει δημιουργηθεί για τη βελτίωση των αποτελεσμάτων των δοκιμών και της σύγκλισης ταχύτητας.

Στην παρούσα διπλωματική εργασία, αφού παρουσιαστούν συνοπτικά οι κυριότεροι τύποι των νευρωνικών δικτύων και στην συνέχεια οι πιο σημαντικές μέθοδοι βελτιστοποίησης τους, μελετήθηκε η απόδοση των μεθόδων βελτιστοποίησης των νευρωνικών δικτύων, μέσω εφαρμογών κώδικα στην γλώσσα προγραμματισμού Python.

Ο πλέον αποδοτικός αλγόριθμος βελτιστοποίησης, ο οποίος αποτελεί και το σκοπό της παρούσας εργασίας, αναδύεται μετά από επανειλημμένες εφαρμογές του κώδικα και σε διαφορετικά σύνολα δεδομένων. Φυσικά, τα συμπεράσματα αφορούν τον συγκεκριμένο αριθμό δοκιμών και τα συγκεκριμένα σύνολα δεδομένων και δεν είναι καθολικά αλλά ικανοποιητικά ενδεικτικά.

1.2. Κίνητρα και στόχοι υλοποίησης

Υπάρχουν διάφορες γλώσσες προγραμματισμού οι οποίες έχουν την δυνατότητα αναλυτικής επεξεργασίας και εφαρμογής των περισσότερο ή λιγότερο γνωστών τύπων νευρωνικών δικτύων σε μικρού ή μεγάλου όγκου δεδομένα. Μια από τις πλέον κατάλληλες για εφαρμογές νευρωνικών δικτύων είναι η γλώσσα προγραμματισμού python. Όπως και σε άλλες γλώσσες, αλλά ίσως σε υψηλότερο βαθμό, η γλώσσα python, περιέχει διάφορες βιβλιοθήκες ειδικές για την επεξεργασία και εφαρμογές των νευρωνικών δικτύων. Οι πιο γνωστές από αυτές είναι η TensorFlow, Keras, Pytorch και άλλες. Η βιβλιοθήκη Keras (https://keras.io/) είναι μια Διεπαφή Προγραμματισμού Εφαρμογών (Application Programming Interface - API), γραμμένη σε γλώσσα προγραμματισμού python, που δημιουργήθηκε και αναπτύχθηκε από την Google, για εφαρμογές Νευρωνικών Δικτύων με εύκολο τρόπο. Λόγω της front-end και της μεγάλης ελαφρότητας κατασκευής της, ίσως είναι πιο αργή από άλλες βιβλιοθήκες νευρωνικών δικτύων αλλά είναι ιδιαίτερα φιλική προς τον χρήστη. Δίνει την δυνατότητα εναλλαγής με διάφορες άλλες back-end βιβλιοθήκες, όπως Tensorflow, PlaidML, Theano και άλλες. Στην παρούσα διπλωματική, λόγω των πλεονεκτημάτων αλλά και λόγω της ευρείας διάδοσης και χρήσης της βιβλιοθήκης Keras, αφενός εξετάστηκαν και αναλύθηκαν διάφοροι τύποι νευρωνικών δικτύων που εφαρμόζονται αρκετά στην καθημερινότητα, (αλλά και άλλου τύπου σε θεωρητικό επίπεδο), αφετέρου εκτίθενται σύντομα θεωρητικά, οι περισσότερες μεθοδολογίες βελτιστοποίησης που περιέχει το Keras, αλλά και μερικές ακόμη.

Επιπλέον, η επιλογή της βιβλιοθήκης Keras έγινε διότι αυτή τρέχει επάνω από το Tensorflow το οποίο όντας γραμμένο και σε άλλες γλώσσες όπως π.χ. η C++ (και άλλες), αυξάνει την ταχύτητά της. Μέσω των αρθρωμάτων που περιέχει, δίνει την δυνατότητα στον χρήστη να προγραμματίσει πιο συνεκτικά και σύντομα, καλώντας βιβλιοθήκες που ήδη περιέχονται στο Keras και είναι πολύ κατατοπιστικό στην περίπτωση σφαλμάτων. Διαθέτει μεγάλη ποικιλομορφία επιλογών ανάλογα με τους στόχους του χρήστη.

Τα μέλη της διεθνούς Πανεπιστημιακής και Ερευνητικής κοινότητας που το χρησιμοποιούν είναι πολυάριθμα. Σε σύγκριση με άλλες βιβλιοθήκες νευρωνικών δικτύων η κοινότητα των χρηστών και η ενδεχόμενη βοήθεια που μπορεί να έχει κάποιος είναι σημαντική.

Λειτουργεί κανονικά τόσο σε CPU όσο και σε GPU. Διαθέτει μια πληθώρα τύπων νευρωνικών δικτύων. Και τέλος, είναι δομημένο με αρθρώματα, ώστε να μπορεί ο χρήστης να προσαρμόσει την έρευνά του, και παρουσιάζει εξαιρετική ευελιξία, γεγονός που το καθιστά ιδιαίτερα φιλικό προς τον χρήστη.

Δεν είναι τυχαίο το γεγονός ότι πολλές μεγάλες διεθνείς εταιρείες όπως οι Netflix, Uber, Yelp, κ.λπ., το χρησιμοποιούν εμπορικά, με προϊόντα που δημιούργησαν με τη χρήση της keras, και που χρησιμοποιούνται στο δημόσιο τομέα.

Τελικά, η σαρωτική χρήση των νευρωνικών δικτύων στην παγκόσμια έρευνα και αγορά, η τεράστια έκταση των εφαρμογών τους σε μεγάλες αλλά και καθημερινές εφαρμογές και η αίσθηση ότι συντελούνται ισχυρές αλλαγές στην ανθρώπινη διαβίωση λόγω της τεχνητής νοημοσύνης, της μηχανικής μάθησης και ειδικότερα των νευρωνικών δικτύων, οδήγησε στην εκπόνηση αυτής της διπλωματικής εργασίας.

1.3. Διάρθρωση κειμένου

Το περιεχόμενο αυτής της διπλωματικής, χωρίζεται σε τρία ουσιαστικά τμήματα. Το πρώτο μετά από μια σύντομη εισαγωγή στην τεχνητή νοημοσύνη και τη μηχανική μάθηση, παρουσιάζει μερικά από τα κυριότερα και πλέον χρησιμοποιούμενα είδη νευρωνικών δικτύων, που είναι γνωστά σήμερα. Το δεύτερο τμήμα παρουσιάζει με όσο πιο σαφή και σχετικά σύντομο τρόπο τις κυρίαρχες μεθοδολογίες βελτιστοποίησης των νευρωνικών δικτύων, με οδηγό την παγκόσμια πλατφόρμα Keras που έχει γραφεί στην επικρατούσα γλώσσα προγραμματισμού για νευρωνικά δίκτυα, την python. Στο τρίτο και τέταρτο τμήμα που είναι από τα σημαντικά τμήματα αυτής της διπλωματικής εφαρμόζοναι σχεδόν όλοι οι τύποι βελτιστοποίησης των νευρωνικών δικτύων που προσφέρονται για ελεύθερη χρήση από την πλατφόρμα Keras. Οι μεθοδολογίες αυτές εφαρμόστηκαν σε γνωστά και ελεύθερα για χρήση σύνολα δεδομένων. Τα τέσσερα αυτά βασικά τμήματα της διπλωματικής, πλαισιώνονται από την παρουσίαση των αποτελεσμάτων αυτής της έρευνας, την τοποθέτηση ως προς το περιεχόμενο και τα αποτελέσματα, συμπεράσματα για την συμπεριφορά και την αποδοτικότητα των αλγορίθμων, και τέλος από προτάσεις για μελλοντικές επεκτάσεις, μελέτη και έρευνα σχετικά με τις μεθόδους βελτιστοποίησης των νευρωνικών δικτύων.

2.Τεχνητή Νοημοσύνη και Μηχανική Μάθηση

Σήμερα στο ευρύ κοινό αλλά μερικές φορές και στην παγκόσμια επιστημονική κοινότητα άλλων ειδικοτήτων, υπάρχει η τάση να συγχέονται η Τεχνητή Νοημοσύνη (Artificial Intelligence) και Μηχανική Μάθηση (Machine Learning). Η κυριότερη αιτία για την σύγχυση αυτή είναι ότι η μηχανική μάθηση είναι υποσύνολο της τεχνητής νοημοσύνης, όπως πολύ σωστά εξηγεί ένας διακεκριμένος μηχανικός της διεθνούς εταιρείας υπολογιστών και λογισμικού IBM στο αντίστοιχο βίντεο του YouTube. Οι τομείς έρευνας και εφαρμογής και των δύο αυτών επιστημονικών κλάδων είναι πολλοί και ποικίλοι και σε αρκετές περιπτώσεις συναφείς. Οι πλέον γνωστοί από αυτούς είναι η ανάλυση μεγάλου όγκου δεδομένων, η πρόβλεψη ποσοτικών αλλά και ποιοτικών χαρακτηριστικών και μεταβλητών, οι ψηφιακοί μετασχηματισμοί, η ομαδοποίηση, η συμπίεση δεδομένων και πολλά άλλα. Ωστόσο υπάρχουν και σημαντικές διαφορές μεταξύ αυτών των δύο επιστημονικών κλάδων.

Η τεχνητή νοημοσύνη είναι ένας πολύ γενικός επιστημονικός κλάδος, που εξειδικεύεται στην κατασκευή ηλεκτρονικών υπολογιστών και γενικότερα μηχανών οι οποίες έχουν την ικανότητα να υιοθετούν ανθρώπινες νοητικές και διαδικαστικές λειτουργίες. Οι μηχανές αυτές, κάνουν χρήση ευφυίας ανάλογης με την ανθρώπινη, όπως η τεχνητή όραση και σαν συνέπεια η αντίληψη του εξωτερικού από αυτές κόσμου, οι αναλύσεις της γραπτής ή προφορικής ομιλίας και η δυνατότητα μέσω αυτών για επικοινωνία με ανθρώπους, η ανάλυση των δεδομένων που τις οδηγεί μέχρι και του σημείου να κάνουν προτάσεις για την επίλυση διαφόρων προβλημάτων (Russell J. St., 2005).

Η ίδια η τεχνητή νοημοσύνη θεωρείται σαν αυτόνομη, ωστόσο, αποτελείται από διάφορες επιμέρους τεχνικές και μεθοδολογίες, οι οποίες τις κατευθύνουν, στην συλλογή δεδομένων, στην εκμάθηση από τα δεδομένα και τελικά στην λήψη αποφάσεων, με σκοπό την επίλυση διαφόρων τύπων σύνθετα προβλήματα.

Η μηχανική μάθηση, είναι ένα υποσύνολο της τεχνητής νοημοσύνης. Πρόκειται για ένα σύστημα, το οποίο μαθαίνει από τα εκάστοτε δεδομένα και βελτιώνεται συνεχώς από την επαναλαμβανόμενη και συνεχόμενη μάθηση της (Hyatt, 2018). Χρησιμοποιεί διάφορους αλγόριθμους και γενικότερα μεθοδολογικές αλυσίδες, για να αναλύει μεγάλη ποσότητα δεδομένων και να μαθαίνει από τα δεδομένα αυτά ώστε να μπορεί να παίρνει κάποιες πιο

εμπεριστατωμένες αποφάσεις. Η βελτιστοποίηση της μηχανικής μάθησης συντελείται με το χρόνο και με τη χρήση ολοένα και περισσότερων δεδομένων.

Σε τι έγκειται λοιπόν, η σημαντική διαφορά μεταξύ της τεχνητής νοημοσύνης και της μηχανικής μάθησης;

Η τεχνητή νοημοσύνη είναι ένας επιστημονικός κλάδος με ευρύτερη σύλληψη που δίνει την δυνατότητα σε μια μηχανή ή σε ένα σύστημα να αντιλαμβάνεται να αναλύει τα δεδομένα και να επιλύει προβλήματα όπως ακριβώς οι άνθρωποι.

Η μηχανική μάθηση είναι ένα υποσύνολο της τεχνητής νοημοσύνης, που επιτρέπει στις μηχανές ή σε προγράμματα ηλεκτρονικών υπολογιστών, να εξορύξουν γνώση από δεδομένα και να μαθαίνουν αυτόνομα από αυτά ώστε να επιλύουν σύνθετα προβλήματα.

Επίσης, ενώ η Τεχνητή νοημοσύνη, έχει ως στόχο την δημιουργία μιας μηχανής, κατάλληλα κατασκευασμένης ώστε να μπορεί να μιμηθεί την ανθρώπινη ευφυία, η μηχανική μάθηση, έχει σαν σκοπό να μάθει σε μια μηχανή την διαδικασία που θα εκτελέσει μια ειδική εργασία ή επεξεργασία ανιχνεύοντας και αναλύοντας διάφορα μοτίβα και διάφορους συσχετισμούς μεταξύ των δεδομένων, με τρόπο ώστε να εξάγει αξιόπιστα αποτελέσματα.

Η Τεχνητή νοημοσύνη μπορεί να προσομοιάσει με την ανθρώπινη ευφυία, έτσι ώστε να επιλύει σύνθετα προβλήματα με τρόπο παρόμοιο με τον ανθρώπινο. Μπορεί να χρησιμοποιεί διάφορα τεχνολογικά εργαλεία ώστε να φθάσει στην λήψη αποφάσεων με ανθρώπινο τρόπο. Μπορεί να χειριστεί διαφόρων τύπων δεδομένων, δομημένα και αδόμητα. Τέλος, έχει την δυνατότητα να χρησιμοποιήσει έναν ιεραρχικό τρόπο απόφασης, δηλαδή τα «δένδρα αποφάσεων», έτσι ώστε να μαθαίνει, να αποφασίζει και να βελτιώνεται.Τι χαρακτηρίζει τη μηχανική μάθηση; Η μηχανική μάθηση, έχει καθορισμένο σκοπό και περιορισμένο πεδίο εφαρμογών. Είναι κατάλληλη στην εκπαίδευση αλγορίθμων, προγραμμάτων και μηχανών, από τα εκάστοτε δεδομένα έτσι ώστε να τα αναλύει, και να αποδίδει αξιόπιστα αποτελέσματα (Διαμαντάρας Κ., 2019).

Επίσης χρησιμοποιεί αλγόριθμους και μεθοδολογίες οι οποίοι επιδέχονται βελτιστοποίηση, έτσι ώστε να αυξάνει την ορθότητα των αποτελεσμάτων. Μαθαίνει αυτόνομα, από παλαιότερα δεδομένα και από ειδικούς αλγόριθμους. Χρησιμοποιεί μόνο δομημένα ή ημι-δομημένα δεδομένα. Γενικά και σε ένα μεγάλο τμήμα της βασίζεται σε στατιστικά μοντέλα με σκοπό τη μάθηση και

την βελτίωση της μάθησης της, με σκοπό την εφαρμογή της βελτιωμένης μάθησης σε καινούργια δεδομένα (Mueller J. P., 2016).

Υπάρχουν αρκετοί κοινοί τομείς εφαρμογής της τεχνητής νοημοσύνης και της μηχανικής μάθησης.

Ένας είναι ο τομέας υγείας και των διαφόρων ιατρικών διαδικασιών και τεχνικών. Για παράδειγμα, κάποιος ασθενής μπορεί να πάρει δεδομένα της προσωπικής του κατάστασης της υγείας του, που έχουν αναλυθεί από τη μηχανική μάθηση, η οποία προβλέπει με κάποιο ποσοστό επιτυχίας την ίαση του ασθενούς. Η τελευταία, προτείνει κάποιες εντολές διατροφής και διαβίωσης καθώς και μια ορισμένη δοσολογία φαρμάκων που πρέπει να ακολουθήσει για την θεραπεία του, και τέλος πάντα με την σύμφωνη γνώμη του ασθενούς αυτού και ανώνυμα, αποθηκεύει πληροφορίες για άλλους ασθενείς.

Σημαντικός και σύγχρονος τομέας είναι το ηλεκτρονικό εμπόριο. Η πρόβλεψη της ζήτησης της αγοράς, οι διάφορες προσωπικές προσφορές, καθώς και η εμπειρίες του αγοραστικού κοινού αλλά του Υπουργείου Εμπορίου και του εμπορικού συνόλου των ανθρώπων, βοηθούν στη βελτίωση του ηλεκτρονικού εμπορίου.

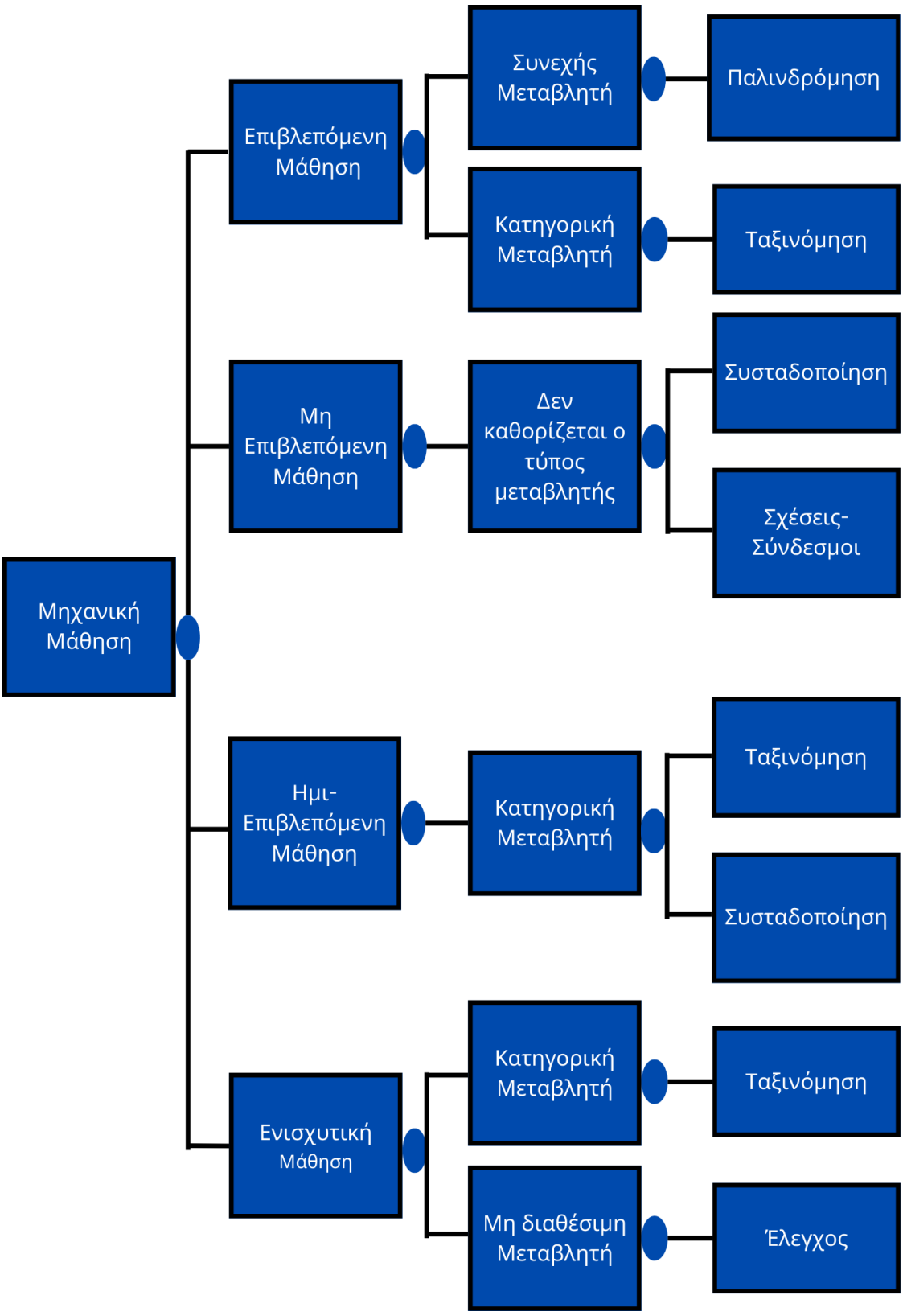
Επίσης είναι και ο τομέας της παραγωγής. Η τεχνητή νοημοσύνη με την χρήση κατάλληλων πληροφοριών από τα ΜΜΕ, την ανάλυση των δεδομένων και την επιχειρηματική αποτελεσματικότητα μπορεί να συντελέσει και στην ποιοτική και στην ποσοτική βελτίωση της παραγωγής.

Είναι και ο τομέας κάποιας Εθνικής ή ακόμη και της Παγκόσμιας οικονομίας. Η πρόβλεψη κινδύνου και η ανάλυση. Η έρευνα οικονομικών απατών και ή αυτόματη συναλλαγή με διάφορους οικονομικούς οργανισμούς, όπως είναι οι τράπεζες, που είναι μερικές από τις υπηρεσίες της τεχνητής νοημοσύνης και της μηχανικής μάθησης.

Άλλος τομέας είναι οι τηλεπικοινωνίες. Ο σχεδιασμός και η δημιουργία Ευφυών δικτύων, η βελτίωση, η συντήρηση και η αυτοματοποίηση τους και η πρόβλεψη ορθολογικής και οικονομικής λειτουργίας τους, είναι τομείς οι οποίοι προάγουν ιδιαίτερα τις τηλεπικοινωνίες και επιτελούνται με την μηχανική μάθηση και την τεχνητή νοημοσύνη.

2.1. Κατηγορίες μηχανικής μάθησης

Στο Σχήμα 1, που δημιουργήθηκε με βάση μια δημοσίευση από τον ιστότοπο “Medium” (Kumar, 2018), φαίνονται οι τέσσερις βασικοί τύποι Μηχανικής Μάθησης, με τους αντίστοιχους τύπους μεταβλητών που αναλύουν και τους αντιπροσωπευτικότερους αλγόριθμους -μεθόδους του κάθε τύπου.



Σχήμα 1-Κλάδοι και Τομείς της Μηχανικής Μάθησης

2.1.1. Επιβλεπόμενη ή Καθοδηγούμενη ή Εποπτευόμενη μάθηση (Supervised Learning)

Στην επιβλεπόμενη μάθηση, οι κατηγορίες στις οποίες θα διαχωρισθούν τα δεδομένα εισόδου με βάση γνωστούς αλγόριθμους, είναι από την αρχή γνωστές, και ο αλγόριθμος πρέπει να μπορεί να αντιστοιχίσει σε κάθε αντικείμενο της εισόδου μια από τις πρότερα ορισμένες κατηγορίες. Για παράδειγμα ένας μεγάλος αριθμός από εικόνες γατών, σκύλων, αλεπούδων, και λύκων δίνονται ως δεδομένα εισόδου. Ο αλγόριθμος τροφοδοτείται για εκπαίδευση και πρέπει να μάθει αναγνωρίζει τα ομοειδή ζώα με βάση ορισμένα κριτήρια. Στην έξοδο παρέχονται σε κλάσεις, όλα τα αντικείμενα εισόδου κατηγοριοποιημένα. Στην κατηγορία αυτή της μηχανικής μάθησης ανήκουν η ταξινόμηση και η παλινδρόμηση. Όλα τα δεδομένα εκπαίδευσης είναι κατηγοριοποιημένα σε κάποια από τις τελικές κλάσεις.

Οι πιο γνωστοί και περισσότερο χρησιμοποιούμενοι αλγόριθμοι είναι οι: Η Γραμμική παλινδρόμηση (Linear Regression), οι Μηχανές Διανυσμάτων Υποστήριξης (Support Vector Machines), τα Νευρωνικά Δίκτυα (Neural Networks), τα Δέντρα Αποφάσεων(Decision Trees), ο Αφελής Ταξινομητής Bayes (Naive Bayes) και η Συσταδοποίηση με βάση τον πλησιέστερο γείτονα(k-Neighbor Clustering) . Οι κύριες χρήσεις της είναι για ταξινομήσεις και για την δημιουργία μοντέλων πρόβλεψης.

Λόγω της σπουδαιότητας και της ευρείας χρηστικότητάς τους, στη συνέχεια θα εκτεθούν σύντομα οι αλγόριθμοι των Μηχανών Διανυσμάτων Υποστήριξης και των Δέντρων Αποφάσεων.

Μηχανές Διανυσμάτων Υποστήριξης – Support Vector Machines (SVM)

Οι Support Vector Machines (SVM), αλγόριθμοι είναι επιβλεπόμενης μάθησης που χρησιμοποιούνται κυρίως για προβλήματα ταξινόμησης, αλλά και για παλινδρόμηση. Ο κύριος στόχος των SVM είναι να εντοπίσουν το βέλτιστο υπερεπίπεδο που διαχωρίζει διαφορετικές κατηγορίες δεδομένων σε έναν πολυδιάστατο χώρο χαρακτηριστικών. Η λειτουργία των SVM ακολουθεί στα εξής βήματα (Deisenroth, 2020):

1) Ο αλγόριθμος εκπαιδεύεται χρησιμοποιώντας ένα σύνολο δεδομένων, όπου κάθε σημείο δεδομένων ανήκει σε μία από τις προκαθορισμένες κατηγορίες. Κατά τη διάρκεια της εκπαίδευσης, ο αλγόριθμος, SVM, προσπαθεί να βρει το υπερεπίπεδο που μεγιστοποιεί την απόσταση-περιθώριο (margin) μεταξύ των κοντινότερων σημείων των διαφορετικών κατηγοριών, γνωστών ως support vectors. Το υπερεπίπεδο είναι η γραμμή ή το επίπεδο που διαχωρίζει τις κατηγορίες. Στην περίπτωση δύο διαστάσεων, το υπερεπίπεδο είναι μια ευθεία γραμμή, ενώ σε τρεις διαστάσεις γίνεται ένα επίπεδο. 2) Για μη γραμμικά διαχωρίσιμα δεδομένα, ο SVM χρησιμοποιεί την τεχνική του πυρήνα (Kernel Trick), η οποία μετατρέπει τα δεδομένα σε υψηλότερη διάσταση, επιτρέποντας τη γραμμική ταξινόμηση σε αυτή τη νέα διάσταση. 3) Αφού εκπαιδευτεί ο αλγόριθμος στα δεδομένα εκπαίδευσης, μπορεί να χρησιμοποιηθεί για την ταξινόμηση νέων σημείων δεδομένων, προσδιορίζοντας σε ποια κατηγορία ανήκουν με βάση τη θέση τους σχετικά με το υπερεπίπεδο. Τα κυριότερα πλεονεκτήματα του αλγορίθμου SVM είναι:

Η αποτελεσματικότητά του σε μεγάλο αριθμό διαστάσεων. Ο SVM είναι ιδιαίτερα αποτελεσματικός όταν τα δεδομένα έχουν πολλές διαστάσεις, καθώς μπορεί να χειριστεί καλά την πολυπλοκότητα. Η ιδιότητα του μέγιστου περιθωρίου (margin). Ο SVM προσπαθεί να μεγιστοποιήσει την απόσταση μεταξύ των κατηγοριών, γεγονός που τον καθιστά ανθεκτικό σε θόρυβο και υπερβολική προσαρμογή. Και τέλος η ευελιξία με τους πυρήνες (Kernels). Η δυνατότητα χρήσης διαφορετικών πυρήνων επιτρέπει στον SVM να προσαρμόζεται σε διάφορους τύπους δεδομένων και προβλημάτων.

Ωστόσο δεν πρέπει να παραλειφθούν και τα μειονεκτήματα του αλγόριθμου SVM, που είναι: Μεγάλη κατανάλωση Υπολογιστικών Πόρων: Η εκπαίδευση ενός SVM μπορεί να είναι χρονοβόρα και απαιτητική σε υπολογιστικούς πόρους, ειδικά για μεγάλα σύνολα δεδομένων. Η επιλογή του κατάλληλου πυρήνα, και των παραμέτρων του μπορεί να είναι δύσκολη και απαιτεί εμπειρία. Σε σύγκριση με άλλους αλγόριθμους όπως για παράδειγμα, με την λογιστική παλινδρόμηση, οι αποφάσεις του SVM μπορεί να είναι λιγότερο διαισθητικές και πιο δύσκολες στην ερμηνεία. Συνοψίζοντας, οι αλγόριθμοι «Support Vector Machines» αποτελούν έναν ισχυρό εργαλείο στη Μηχανική Μάθηση για την ταξινόμηση και την παλινδρόμηση, και παρουσιάζουν σημαντικά πλεονεκτήματα και όχι αμελητέα ελαττώματα. Οι αλγόριθμοι «Support Vector Machine» (SVM) χρησιμοποιούνται κυρίως για ταξινόμηση και παλινδρόμηση. Αναλυτικά, τα βήματα ενός Αλγορίθμου SVM είναι:

1) Η συλλογή των δεδομένων. Ο αλγόριθμος SVM ξεκινά με ένα σύνολο εκπαιδευτικών δεδομένων, όπου κάθε δείγμα έχει χαρακτηριστικά (features) και μια ονοματολογία της κατηγορίας (label) που ανήκει.

2) Ο ορισμός του Υπερεπίπεδου διαχωρισμού των διαφόρων ατόμων σε ομοειδή μεταξύ τους σύνολα και διαφορετικά από τα υπόλοιπα σύνολα, που στοχεύει να βρεθεί το υπερεπίπεδο (hyperplane) που διαχωρίζει τις διάφορες κατηγορίες. Στον δισδιάστατο χώρο, το υπερεπίπεδο είναι μια ευθεία γραμμή, ενώ σε τρεις διαστάσεις είναι ένα επίπεδο.

3) Ο αλγόριθμος SVM, προσπαθεί να βρει το υπερεπίπεδο που μεγιστοποιεί την απόσταση (margin) μεταξύ των κοντινότερων σημείων από κάθε κατηγορία, γνωστά ως support vectors. Αυτά τα σημεία είναι κρίσιμα για τον καθορισμό του υπερεπιπέδου.

Το πρόβλημα μπορεί να διατυπωθεί σε υπολογιστικά βήματα ως εξής:

Ελαχιστοποίηση της συνάρτησης κόστους: $\frac{1}{2\|w\|^2}$, υπό τους περιορισμούς: $y_i(wx_i + b) \geq 1$, για κάθε i, όπου w είναι το διάνυσμα των βαρών, b είναι η σταθερά, x_i είναι τα χαρακτηριστικά του δείγματος και y_i είναι η ονομασία της κατηγορίας (+1 ή -1).

Εφαρμογή της τεχνικής του πυρήνα: Για μη γραμμικά διαχωρίσιμα δεδομένα, ο SVM χρησιμοποιεί την τεχνική του πυρήνα, που επιτρέπει τη μετατροπή των δεδομένων σε υψηλότερη διάσταση. Κοινοί τύποι πυρήνων είναι : Ο γραμμικός πυρήνας (Linear Kernel), ο πολυωνυμικός πυρήνας (Polynomial Kernel), ο πυρήνας συνάρτησης ακτινωτής βάσης (Radial Basis Function – RBF – Kernel)

Τέλος, αφού εκπαιδευτεί το μοντέλο, ο αλγόριθμος SVM μπορεί να χρησιμοποιηθεί για την ταξινόμηση νέων δειγμάτων. Η απόφαση γίνεται με βάση την εξίσωση του υπερεπιπέδου : $f(x) = wx + b$. Αν $f(x)$>0, το δείγμα ταξινομείται στην κατηγορία +1, αλλιώς στην κατηγορία -1.

Υπάρχουν και κάποιες σημαντικές παράμετροι του αλγόριθμου που είναι : Η παράμετρος ποινής C (Penalty Parameter), η οποία ρυθμίζει την ισορροπία μεταξύ της μεγιστοποίησης του περιθωρίου (margin) και της ελαχιστοποίησης των σφαλμάτων ταξινόμησης. Ένα υψηλό C σημαίνει ότι ο SVM θα προσπαθήσει να ταξινομήσει όλα τα δείγματα σωστά, ενώ ένα χαμηλό C επιτρέπει κάποια σφάλματα. Η παράμετρος γ (Gamma), η οποία χρησιμοποιείται στον πυρήνα συνάρτησης ακτινωτής βάσης (RBF kernel) και καθορίζει την επιρροή ενός δείγματος. Ένα υψηλό

γ μπορεί να οδηγήσει σε υπερβολική προσαρμογή, ενώ ένα χαμηλό γ μπορεί να υποδείξει γενικευμένη συμπεριφορά. Γενικότερα, Ο αλγόριθμος SVM είναι μια ισχυρή μέθοδος για την επίλυση προβλημάτων ταξινόμησης και παλινδρόμησης, με τη δυνατότητα να χειρίζεται πολύπλοκες σχέσεις μεταξύ των χαρακτηριστικών μέσω της χρήσης των πυρήνων. Η κατανόηση της λειτουργίας του και των παραμέτρων του είναι κρίσιμη για την αποτελεσματική εφαρμογή του σε διάφορα προβλήματα μηχανικής μάθησης.

Τυχαία Δάση (Δέντρων) – Random (Tree) Forests

Ο όρος «Random Forests» μεταφράζεται στα Ελληνικά ως Τυχαία Δάση. Αυτή η μέθοδος είναι μια τεχνική μηχανικής μάθησης που συνδυάζει πολλαπλά δέντρα αποφάσεων για να βελτιώσει την ορθότητα της ταξινόμησης ή της παλινδρόμησης. Η έννοια αυτή αναφέρεται σε ένα σύνολο αλγορίθμων που χρησιμοποιούν τυχαία δείγματα δεδομένων και χαρακτηριστικών για την κατασκευή των δέντρων, επιτρέποντας έτσι την καλύτερη γενίκευση και ανθεκτικότητα σε θόρυβο στα δεδομένα.

Ο αλγόριθμος των Τυχαίων Δασών, είναι μια δημοφιλής μέθοδος μηχανικής μάθησης που χρησιμοποιείται για ταξινόμηση και παλινδρόμηση. Λειτουργεί με τη δημιουργία πολλαπλών δέντρων αποφάσεων και τη συνδυαστική τους χρήση για την παραγωγή πιο ακριβών προβλέψεων. Ακολουθούν τα βασικά βήματα και οι αρχές λειτουργίας του αλγορίθμου.

Δημιουργία Δέντρων Απόφασης: Με επαναλαμβανόμενη Δειγματοληψία (Bootstrap). Ο αλγόριθμος, Random Forest, δημιουργεί κάθε δέντρο απόφασης χρησιμοποιώντας τυχαία υποσύνολα των δεδομένων εκπαίδευσης μέσω της μεθόδου bootstrap. Αυτό σημαίνει ότι κάθε δέντρο εκπαιδεύεται σε ένα διαφορετικό δείγμα, το οποίο περιλαμβάνει επαναλαμβανόμενα στοιχεία από το αρχικό σύνολο δεδομένων.

Με τυχαία επιλογή χαρακτηριστικών. Κατά την κατασκευή κάθε δέντρου, ο αλγόριθμος επιλέγει τυχαία ένα υποσύνολο χαρακτηριστικών για να επιλέξει τον βέλτιστο διαχωρισμό (split) σε κάθε κόμβο. Αυτή η διαδικασία μειώνει τη συσχέτιση μεταξύ των δέντρων και ενισχύει την ποικιλία στο μοντέλο. Με εκπαίδευση των δέντρων: Κάθε δέντρο αποφάσεων εκπαιδεύεται χρησιμοποιώντας τον αλγόριθμο CART (Classification and Regression Trees), ο οποίος στοχεύει

στην εύρεση του βέλτιστου διαχωρισμού (split) για να διαχωρίσει τις κατηγορίες ή να εκτιμήσει τις τιμές. Οι αποφάσεις στα δέντρα βασίζονται σε διάφορα κριτήρια, όπως η «Gini impurity» ή «μη καθαρότητα Gini», που είναι ένα μέτρο που χρησιμοποιείται στους αλγόριθμους δέντρων αποφάσεων, συμπεριλαμβανομένων των Random Forests, για να αξιολογήσει την ποιότητα ενός διαχωρισμού (split) σε ένα δέντρο απόφασης. Επίσης, η «information gain», ή «κερδισμένη πληροφορία» που υπολογίζει την ποσότητα πληροφορίας που αποκτάται για την κατηγοριοποίηση ενός δείγματος όταν χρησιμοποιείται ένα συγκεκριμένο χαρακτηριστικό για τον διαχωρισμό. Όσο μεγαλύτερη είναι η κερδισμένη πληροφορία, τόσο καλύτερα το χαρακτηριστικό διαχωρίζει τις κατηγορίες για να καθορίσουν ποιες ερωτήσεις θα γίνουν σε κάθε κόμβο. Με συνδυασμό των αποτελεσμάτων, δηλαδή με Πλειοψηφική Ψηφοφορία. Για προβλήματα ταξινόμησης, τα αποτελέσματα όλων των δέντρων συνδυάζονται μέσω πλειοψηφικής ψηφοφορίας. Η κατηγορία που λαμβάνει τις περισσότερες ψήφους από τα δέντρα είναι η τελική πρόβλεψη του Random Forest. Για προβλήματα παλινδρόμησης, οι προβλέψεις των δέντρων συνδυάζονται με τη μορφή του μέσου όρου. Και τέλος, με Αξιολόγηση και Γενίκευση: Ο Random Forest χρησιμοποιεί την τεχνική out-of-bag (OOB) για την αξιολόγηση της απόδοσης του μοντέλου. Ένα ποσοστό του δείγματος δεν χρησιμοποιείται στην εκπαίδευση αλλά στην αξιολόγηση, παρέχοντας μια εκτίμηση της γενίκευσης χωρίς την ανάγκη ξεχωριστού συνόλου δοκιμών. Τα πλεονεκτήματα είναι κυρίως τρία: Η υψηλή ορθότητα και ανθεκτικότητα στον θόρυβο. Η μείωση του κινδύνου υπερβολικής προσαρμογής (overfitting). Και τέλος, η ικανότητα χειρισμού μεγάλων συνόλων δεδομένων με πολλές διαστάσεις.

Αλλά, φυσικά, υπάρχουν και μειονεκτήματα: Μπορεί να είναι λιγότερο ερμηνεύσιμος από άλλα μοντέλα, όπως τα απλά δέντρα αποφάσεων. Επίσης εμφανίζεται, υψηλή υπολογιστική πολυπλοκότητα κατά την εκπαίδευση, ειδικά με μεγάλο αριθμό δέντρων.

Συνολικά, όμως, ο αλγόριθμος Random Forest είναι μια ισχυρή και ευέλικτη μέθοδος που έχει ευρεία εφαρμογή σε διάφορους τομείς, όπως η χρηματοοικονομική ανάλυση, η ιατρική διάγνωση και η πρόβλεψη καταναλωτικών συμπεριφορών.

2.1.2. Μη επιβλεπόμενη ή μη Καθοδηγούμενη ή μη Εποπτευόμενη Μάθηση (Unsupervised Learning)

Οι κλάσεις (το πλήθος) και οι κατηγορίες εξόδου είναι άγνωστες. Τα δεδομένα εισόδου ομαδοποιούνται με βάση κάποιες «αποστάσεις» ή κριτήρια ομοιότητας και διαφοροποίησης (συνήθως τα διαθέσιμα δεδομένα προέρχονται από τον πραγματικό κόσμο). Οι κύριοι αλγόριθμοι της κατηγορίας αυτής είναι αλγόριθμοι ομαδοποίησης και αλγόριθμοι εκμάθησης. Χρησιμοποιείται κυρίως για προβλήματα συσταδοποίησης (clustering), όπως για την ανίχνευση ανωμαλιών-σημείων ολόκληρου του πληθυσμού που δεν κατηγοριοποιούνται σε κάποια κλάση (σε ασυνήθιστες συναλλαγές τραπεζών). Τα μη κατηγοριοποιημένα δεδομένα χρησιμοποιούνται σε μάθηση χωρίς επίβλεψη. Οι πιο γνωστοί αλγόριθμοι είναι: η συσταδοποίηση των k-μέσων (k-means clustering), και οι πίνακες αποστάσεων ή συσχετίσεων (Distance or Correlation Matrices). Χρησιμοποιείται κυρίως στην Περιγραφική Μοντελοποίηση (Descriptive Modelling).

2.1.3. Ημι-εποπτευόμενη μάθηση(Semi-supervised Learning)

Πρόκειται για έναν συνδυασμό της εποπτευόμενης και της μη εποπτευόμενης μάθησης. Όλα τα διαθέσιμα δεδομένα είναι ένας συνδυασμός κατηγοριοποιημένων ή μη δεδομένων. Μια διαφοροποίηση της ημι-εποπτευόμενης μάθησης είναι ο συνδυασμός εφαρμογής αλγορίθμων μη εποπτευόμενης ταξινόμησης οπότε τα δεδομένα κατηγοριοποιούνται και στη συνέχεια εφαρμογής εποπτευόμενης ταξινόμησης για την πιο αξιόπιστη ομαδοποίηση των δεδομένων (η μεθοδολογία αυτή συναντάται και ως υβριδική ταξινόμηση)

2.1.4. Ενισχυτική μάθηση (Reinforced Learning)

Ο αλγόριθμος εκπαίδευσης είναι αλγόριθμος «δοκιμής και λάθους», και στη συνέχεια λαμβάνει μια απόφαση με βάση ορισμένα κριτήρια που τον οδηγούν στο σωστό ή λάθος. Ο αλγόριθμος διδάσκεται και μαθαίνει από τις προηγούμενες δοκιμές, δέχεται με έναν αυτοματοποιημένο τρόπο κάποιες κριτικές και αποφασίζει για την επόμενη κίνηση δοκιμής και λάθους ενισχυμένος και από την προηγούμενη γνώση και από τις κριτικές που έλαβε προηγουμένως.

Οι πιο δημοφιλείς αλγόριθμοι της ενισχυτικής μάθησης είναι Βαθιά Ανταγωνιστικά Δίκτυα (Deep Adversarial Networks) και η Q-Μάθηση (Q-Learning). Κάποιοι τομείς εφαρμογής της ενισχυτικής μάθησης είναι τα αυτό-κινούμενα αυτοκίνητα αλλά και γνωστά επιτραπέζια παιχνίδια με υπολογιστή όπως το σκάκι.

2.2. Νευρωνικά Δίκτυα

2.2.1. Νευρωνικά δίκτυα στη Μηχανική Μάθηση

Τα νευρωνικά δίκτυα παρουσιάζουν πολλά πλεονεκτήματα στις εφαρμογές της Μηχανικής Μάθησης, τα κυριότερα από τα οποία είναι:

Λόγω της αρχικής δημιουργίας τους και των ποικίλλων αρχιτεκτονικών που μπορούν να έχουν, διαθέτουν μεγάλη προσαρμοστικότητα τόσο σε διάφορους τύπους δεδομένων, όσο και σε ετερόμορφες εφαρμογές όπως είναι η επεξεργασία εικόνων διάφορης προέλευσης, χρήσης και σκοπού, (ιατρικές, αναγνώρισης προσώπων ή αντικειμένων, ομαδοποίησης αντικειμένων, κ.ά.) η αναγνώριση και ταξινόμηση φωνών ή ήχων. Επιπλέον, η καταγραφή και επεξεργασία κειμένων από ήχο ή εικόνες, τα καθιστούν ιδιαίτερα ευέλικτα σε μεγάλη γκάμα εφαρμογών.

Διαθέτοντας την δυνατότητα αναγνώρισης και προσδιορισμού ιδιαιτέρων σχέσεων μεταξύ των δεδομένων που επεξεργάζονται, μπορούν να εξάγουν μοτίβα και χαρακτηριστικά που τα καθιστούν ιδιαίτερα ισχυρά στην αξιόπιστη ομαδοποίηση άγνωστων δεδομένων ή στην πρόβλεψη σχέσεων μεταξύ διαφόρων μεταβλητών. Τα νευρωνικά δίκτυα ενδείκνυνται έτσι, για την αναγνώριση εικόνας και την η επεξεργασία φυσικής γλώσσας.

Η παγκόσμια ανάγκη για έγκαιρη, αξιόπιστη και στα πλαίσια του δυνατού, καθολική πληροφόρηση, δημιούργησε ογκώδεις αν όχι τεράστιες βάσεις δεδομένων μεγάλης γκάμας αλλά και μεγάλου όγκου. Τα νευρωνικά δίκτυα χειρίζονται ικανοποιητικά τέτοιου τύπου δεδομένα.

Με την δημιουργία πολλαπλών τύπων εξειδικευμένων νευρωνικών δικτύων, για ομαδοποιημένες ως προς το είδος των δεδομένων που χειρίζονται και τον τελικό σκοπό εφαρμογής τους προσφέρουν αποτελεσματικότητα σε πολύ υψηλά ποσοστά. Τέτοιες, για παράδειγμα, οικογένειες νευρωνικών δικτύων είναι τα Επαναληπτικά ή Ανατροφοδοτούμενα (Recurrent) και τα Συνελικτικά (Convolutional) Νευρωνικά Δίκτυα.

2.3. Κατηγορίες Νευρωνικών Δικτύων

Η διεθνής επιστημονική βιβλιογραφία που αναφέρεται στα νευρωνικά δίκτυα δεν παρέχει σαφή και καθολική άποψη της κατηγοριοποίησης τους. Υπάρχουν διάφορες κατηγοριοποιήσεις σχετικά

με τη δομή τους και την κατασκευή τους ή σχετικά με τη λειτουργία τους ή σχετικά με τη χρηστικότητα τους ή/και συνδυασμούς αυτών. Στην παρούσα διπλωματική εργασία υιοθετήθηκε μία κατηγοριοποιημένη επιλογή των νευρωνικών δικτύων, συνδυάζοντας κάποια από τα προηγούμενα κριτήρια, λαμβάνοντας υπόψη την χρηστικότητά τους.

2.3.1. Η χρήση της βιβλιοθήκης Keras

Το Keras είναι μια βιβλιοθήκη νευρωνικών δικτύων ανοιχτού κώδικα υψηλού επιπέδου που έχει σχεδιαστεί για να είναι φιλική προς το χρήστη, σπονδυλωτή και εύκολη στην επέκταση. Είναι γραμμένο σε Python και υποστηρίζει πολλαπλές μηχανές υπολογισμού νευρωνικών δικτύων -- back-end — αν και το κύριο (και προεπιλεγμένο) back end του είναι το TensorFlow και ο κύριος υποστηρικτής του είναι η Google.

Το Keras τρέχει επίσης πάνω από το λογισμικό Theano. Διαθέτει μια σειρά από αυτόνομες μονάδες που μπορούν να συνδυαστούν, συμπεριλαμβανομένων των νευρωνικών επιπέδων, των συναρτήσεων κόστους, των βελτιστοποιητών, των σχημάτων αρχικοποίησης, των συναρτήσεων ενεργοποίησης και των σχημάτων τακτοποίησης.

Το Keras παρέχει υποστήριξη για ένα ευρύ φάσμα επιλογών ανάπτυξης παραγωγής και ισχυρή υποστήριξη για πολλαπλές GPU και κατανεμημένη εκπαίδευση. Ωστόσο, η υποστήριξη της κοινότητας είναι μικρή και η βιβλιοθήκη χρησιμοποιείται συνήθως για μικρά σύνολα δεδομένων.

2.3.2. Εμπροσθο-τροφοδοτούμενα Νευρωνικά Δίκτυα Feed Forward Neural Networks

Τα Νευρωνικά Δίκτυα Εμπροσθο-διάδοσης είναι από τα πρώτα του είδους και ίσως μια από τις απλούστερες μορφές νευρωνικών δικτύων. Τα δεδομένα αρχικά και καθ’ όλη την διαδικασία επεξεργασίας τους κινούνται μόνο προς μια κατεύθυνση και συγκεκριμένα από αριστερά προς τα δεξιά, από τους κόμβους εισόδου, μέσω των κρυφών κόμβων (εάν υπάρχουν) και προς τους κόμβους εξόδου. Η αρχιτεκτονική τους είναι απλή γιατί δεν υπάρχουν κύκλοι ή βρόχοι στο δίκτυο.

Αποτελούνται από τριών ειδών επίπεδα. Το επίπεδο εισόδου (input layer), τα κρυφά επίπεδα (hidden layers) και το επίπεδο εξόδου (output layer). Τα επίπεδα αποτελούνται από μικρότερες δομικές μονάδες, τους νευρώνες οι οποίοι συνδέονται μεταξύ τους - όπως και τα επίπεδα - με βάρη. Τα βάρη είναι συντελεστές στάθμισης για κάθε μεταβλητή.

Το επίπεδο εισόδου, αποτελείται από νευρώνες που τροφοδοτούνται με τις τιμές εισόδου και τις μεταβιβάζουν στο επόμενο επίπεδο. Οι διάσταση του διανύσματος των δεδομένων εισόδου ορίζει τον αριθμό των νευρώνων στο επίπεδο εισόδου.

Τα κρυφά επίπεδα δεν φαίνονται από την είσοδο ή την έξοδο και αποτελούν την βασική υπολογιστική μονάδα ολόκληρου του νευρωνικού δικτύου. Οι νευρώνες του πρώτου κρυφού επιπέδου λαμβάνουν το σταθμισμένο άθροισμα των νευρώνων εισόδου, και στη συνέχεια, οι νευρώνες κάθε κρυφού επιπέδου, λαμβάνουν το σταθμισμένο άθροισμα των εξόδων από το προηγούμενο επίπεδο, εφαρμόζουν μια συνάρτηση ενεργοποίησης και διοχετεύουν το αποτέλεσμα στο επόμενο επίπεδο. Το δίκτυο μπορεί να έχει κανένα, ένα ή περισσότερα κρυφά επίπεδα.

Το τελικό επίπεδο που λέγεται επίπεδο εξόδου, που παράγει το τελικό αποτέλεσμα – έξοδο έχει σαν αριθμό νευρώνων, τον αριθμό των πιθανών εξόδων που έχει σχεδιαστεί να παράγει το δίκτυο.

Ένα πλήρως συνδεδεμένο δίκτυο, είναι εκείνο που κάθε νευρώνας σε ένα επίπεδο συνδέεται με κάθε νευρώνα στο επόμενο επίπεδο. Η ισχύς της σύνδεσης μεταξύ των νευρώνων συνδέεται με τα βάρη και η μάθηση σε ένα νευρωνικό δίκτυο περιλαμβάνει την ενημέρωση αυτών των βαρών με βάση την διαδοχική μείωση του σφάλματος της εξόδου.

Η λειτουργία ενός νευρωνικού δικτύου Εμπροσθο-διάδοσης – Feedforward, περιλαμβάνει δύο στάδια : της εμπροσθο-διάδοσης και της οπισθο-διάδοσης.

Στο Στάδιο της εμπροσθοδιάδοσης, τα δεδομένα εισόδου τροφοδοτούνται στο δίκτυο και μεταδίδονται προς τα εμπρός μέσω του δικτύου. Σε κάθε κρυφό επίπεδο, το σταθμισμένο άθροισμα των εισόδων από το αρχικό ή από προηγούμενο επίπεδο, υπολογίζεται να εισάγεται σε μια συνάρτηση ενεργοποίησης, η οποία με τη σειρά της εισάγει και τη μη γραμμικότητα στο μοντέλο. Αυτή η διαδικασία συνεχίζεται μέχρι να επιτευχθεί η τροφοδότηση του επιπέδου εξόδου ώστε να πραγματοποιηθεί μια πρόβλεψη.

Στο Στάδιο της οπισθοδιάδοσης, μόλις γίνει μια πρόβλεψη, υπολογίζεται το σφάλμα (διαφορά μεταξύ της προβλεπόμενης εξόδου και της πραγματικής εξόδου). Αυτό το σφάλμα στη συνέχεια διαδίδεται ξανά μέσω του δικτύου προς τα πίσω, και τα βάρη προσαρμόζονται για να ελαχιστοποιηθεί αυτό το σφάλμα. Η διαδικασία προσαρμογής των βαρών γίνεται συνήθως χρησιμοποιώντας έναν αλγόριθμο βελτιστοποίησης καθόδου κλίσης (ή παραλλαγής αυτού).

Οι συναρτήσεις ενεργοποίησης είναι ιδιαίτερα σημαντικές στα νευρωνικά δίκτυα ανατροφοδότησης. Εισάγουν μη γραμμικές ιδιότητες στο δίκτυο, γεγονός που δίνει την δυνατότητα στο μοντέλο να μάθει σύνθετες και πολύπλοκες εσωτερικές δομές. Οι πιο γνωστές συναρτήσεις ενεργοποίησης είναι η σιγμοειδής, η υπερβολική εφαπτομένη (tanh), η Διορθωμένη Γραμμική Μονάδα (ReLU), και η μέθοδος softmax.

Η εκπαίδευση ενός νευρωνικού δικτύου ανατροφοδότησης περιλαμβάνει τη χρήση ενός συνόλου δεδομένων για την αναπροσαρμογή των βαρών των συνδέσεων μεταξύ των νευρώνων. Αυτό πραγματοποιείται μέσω μιας επαναληπτικής διαδικασίας, όπου το σύνολο δεδομένων περνά μέσα από το δίκτυο πολλές φορές και κάθε φορά, τα βάρη ενημερώνονται και διορθώνονται ώστε να μειωθεί το σφάλμα της πρόβλεψης. Αυτή η διαδικασία είναι γνωστή ως πτώση κλίσης (gradient descent) και συνεχίζεται μέχρι το δίκτυο να επιτύχει ικανοποιητικό ποσοστό επιτυχίας στα δεδομένα εκπαίδευσης.

Τα νευρωνικά δίκτυα feedforward χρησιμοποιούνται σε μια ποικιλία εργασιών μηχανικής μάθησης, όπως: Αναγνώριση εικόνας, αναγνώριση προτύπων, παλινδρόμηση, ταξινόμηση και ανάλυση και πρόβλεψη με χρονοσειρές.

Τα νευρωνικά δίκτυα ανατροφοδότησης, αν και συγκαταλέγονται στις πλέον απλές μορφές νευρωνικών δικτύων, μπορούν να μοντελοποιήσουν πολύπλοκες σχέσεις στα δεδομένα και αποτέλεσαν τη βάση για πιο σύνθετες αρχιτεκτονικές νευρωνικών δικτύων.

Ενώ τα νευρωνικά δίκτυα ανατροφοδότησης θεωρούνται ιδιαίτερα αποτελεσματικά και σε μεγάλο βαθμό αξιόπιστα, υπόκεινται σε κάποιες σχετικές δεσμεύσεις και απαιτούν κατάλληλες επιλογές. Μία από τις σημαντικές προϋποθέσεις σωστής απόδοσης του δικτύου είναι η επιλογή του πλήθους των κρυφών επιπέδων και του αριθμού των νευρώνων σε κάθε επίπεδο.

Η υπερπροσαρμογή (overfitting) είναι ένα σύνηθες πρόβλημα. Στην περίπτωση αυτή το δίκτυο έχει υψηλό ποσοστό σωστής εκμάθησης από τα δεδομένα εκπαίδευσης, που όμως εμπεριέχεται και θόρυβος ενδεχόμενα, και σαν αποτέλεσμα δεν είναι ικανοποιητικά αποδοτικό με καινούργια δεδομένα.

Τα νευρωνικά δίκτυα γενικότερα, διαμορφώνουν μια μεγάλη οικογένεια προσεγγιστικών μεθοδολογιών πρόβλεψης και ομαδοποίησης στην μοντελοποίηση που οδηγούν σε εξελιγμένες αρχιτεκτονικές νευρωνικών δικτύων για σύγχρονες εφαρμογές τεχνητής νοημοσύνης.

2.3.3. Οπισθο-τροφοδοτούμενα Νευρωνικά Δίκτυα Back Propagation Neural Networks

Ουσιαστικά τα Οπισθο-τροφοδοτούμενα Νευρωνικά Δίκτυα, ανήκουν στην κατηγορία των Εμπροσθο-τροφοδοτούμενων Νευρωνικών Δικτύων, επειδή εκπαιδεύονται με βάση την αρχιτεκτονική των Εμπροσθο-τροφοδοτούμενων Νευρωνικών Δικτύων.

Η οπισθο-διάδοση είναι μια αλγοριθμική μέθοδος εκπαίδευσης που χρησιμοποιείται για την αναπροσαρμογή των βαρών των Εμπροσθο-τροφοδοτούμενων Νευρωνικών Δικτύων, με στόχο τη μείωση του σφάλματος μεταξύ της προβλεπόμενης και της πραγματικής τιμής εξόδου. Αυτή η διαδικασία περιλαμβάνει δύο στάδια: την προώθηση (forward propagation), όπου υπολογίζεται η έξοδος του δικτύου, και την οπισθο-διάδοση (backward propagation), όπου το σφάλμα μεταφέρεται πίσω από την έξοδο προς τα κρυφά επίπεδα για την διόρθωση και την αναπροσαρμογή των βαρών.

Συνοπτικά, τα Οπισθο-τροφοδοτούμενα Νευρωνικά Δίκτυα είναι ένα υποσύνολο των Εμπροσθο-τροφοδοτούμενων Νευρωνικών Δικτύων, επειδή χρησιμοποιούν την αρχιτεκτονική τους και εφαρμόζουν την τεχνική της backpropagation για την εκπαίδευση και τη βελτίωση της απόδοσής τους.

Η backpropagation -οπισθοδιάδοση- είναι μια αποτελεσματική μέθοδος εκπαίδευσης για τα τεχνητά νευρωνικά δίκτυα, ειδικά για τα feedforward neural networks. Αποτελεί έναν

επαναληπτικό αλγόριθμο που βοηθά στην ελαχιστοποίηση της συνάρτησης κόστους, προσδιορίζοντας ποια βάρη και ποιες αποκλίσεις πρέπει να προσαρμοστούν.

Κατά τη διάρκεια κάθε «εποχής», το μοντέλο μαθαίνει προσαρμόζοντας τα βάρη (weights) και τις αποκλίσεις(bias) για να ελαχιστοποιήσει την συνάρτηση απώλειας (Loss function), κινούμενο προς τα κάτω και προς την κλίση του σφάλματος (gradient descent). Έτσι, περιλαμβάνει τους δύο πιο δημοφιλείς αλγόριθμους βελτιστοποίησης, όπως η κατάβαση κλίσης ή η στοχαστική κατάβαση κλίσης.

Στα νευρωνικά δίκτυα, η "εποχή" αποτελεί μία πλήρη εκτέλεση της διαδικασίας εκπαίδευσης του δικτύου στα δεδομένα εκπαίδευσης. Κατά τη διάρκεια κάθε εποχής, το δίκτυο επεξεργάζεται όλα τα δεδομένα εισόδου και προσαρμόζει τα βάρη των συνδέσεων μεταξύ των νευρώνων με βάση τα σφάλματα που προκύπτουν από τις προβλέψεις του σε σχέση με τις πραγματικές τιμές.

Ο υπολογισμός της κλίσης στον αλγόριθμο backpropagation βοηθά στην ελαχιστοποίηση της συνάρτησης κόστους και μπορεί να υλοποιηθεί χρησιμοποιώντας τον μαθηματικό κανόνα που ονομάζεται κανόνας αλυσίδας από τον διαφορικό λογισμό (παραγώγιση σύνθετης συνάρτησης) για να πλοηγηθεί μέσα από τα πολύπλοκα επίπεδα του νευρωνικού δικτύου.

Ο αλγόριθμος backpropagation εκτελείται με δύο διαφορετικές διελεύσεις:

Εμπροσθοδιάδοση ή Διέλευση προς τα εμπρός (Forward pass). Οι αρχικές τιμές των δεδομένων τροφοδοτούν το επίπεδο εισόδου όπου τους προσάπτονται κάποια βάρη. Στη συνέχεια, τα δεδομένα εισόδου και τα αντίστοιχα βάρη τους μεταβιβάζονται στο πρώτο κρυφό επίπεδο. Το κρυφό επίπεδο εκτελεί τους υπολογισμούς στα δεδομένα που λαμβάνει. Πριν εφαρμοστεί η συνάρτηση ενεργοποίησης, προστίθεται η απόκλιση που συνήθως στο γραμμικό μοντέλο είναι ο σταθερός όρος. Μετά, εφαρμόζεται η συνάρτηση ενεργοποίησης (activation function) στο σταθμισμένο άθροισμα των εισόδων σε κάθε νευρώνα του κρυφού επιπέδου.

Οπισθοδιάδοση ή διέλευση προς τα πίσω (Backward pass). Κατά την οπισθοδιάδοση, το σφάλμα διαδίδεται προς τα πίσω από το επίπεδο εξόδου και προς τα κρυφά επίπεδα. Χρησιμοποιώντας τον

κανόνα αλυσίδας, οι παράγωγοι των βαρών υπολογίζονται ξεκινώντας από το επίπεδο εξόδου και προχωρώντας προς τα πίσω μέσω των κρυφών επιπέδων.

Συνοπτικά, το backpropagation είναι μια αποτελεσματική μέθοδος για τον υπολογισμό των παραγώγων κατά την εκπαίδευση των νευρωνικών δικτύων, ειδικά των νευρωνικών δικτύων εμπροσθοδιάδοσης -feedforward neural networks-. Χρησιμοποιεί τον κανόνα αλυσίδας στην παραγώγιση, για να μεταδώσει τις παραγώγους πίσω μέσω των διαφόρων επιπέδων, οδηγώντας σε αποτελεσματική ενημέρωση των βαρών για την ελαχιστοποίηση του σφάλματος.

Όμως όπως σε κάθε προτεινόμενη μέθοδο, έτσι και στην μεθοδολογία της οπισθοδιάδοσης, υπάρχουν κάποιες εγγενείς αδυναμίες. Κατά καιρούς όμως, έχουν προταθεί μέθοδοι βελτιστοποίησης και ελαχιστοποίησης, αν όχι εξάλειψης των προβλημάτων του είδους αυτού. Οι πιο σημαντικές είναι :

Ευαισθησία σε στον θόρυβο: Ο αλγόριθμος μπορεί να είναι ευαίσθητος σε αλλοιωμένα δεδομένα λόγω θορύβου, που καταλήγει σε μειωμένης απόδοσης εκπαίδευση και την τελικής απόδοσης του νευρωνικού δικτύου.

Εξάρτηση από τα Δεδομένα: Η προβληματική ποιότητα (μη ακριβή – ελλιπή – εσφαλμένα – μη αντιπροσωπευτικά – με θόρυβο) και η πολύ μεγάλη ποσότητα των δεδομένων εισόδου, ενέχουν τον κίνδυνο της κακής εκπαίδευσης και γενίκευσης του μοντέλου.

Τοπικά Ελάχιστα: Η διαδικασία εκπαίδευσης μπορεί να καταλήξει σε τοπικά ελάχιστα της συνάρτησης κόστους, αντί να βρει το καθολικό ελάχιστο (infimum). Με αυτόν τον τρόπο, μειώνεται η ικανότητα του μοντέλου να μαθαίνει ικανοποιητικά από τα δεδομένα.

Απαιτήσεις σε Υπολογιστική Ισχύ. Τα βαθιά νευρωνικά δίκτυα που χρησιμοποιούν οπισθοδιάδοση, απαιτούν σημαντικούς υπολογιστικούς πόρους και χρόνο για την εκπαίδευση τους, που δεν είναι εφικτό σε κάθε περίπτωση.

Αξιόπιστη και Αποτελεσματική Ρύθμιση των Υπερπαραμέτρων: Η αποτελεσματικότητα της backpropagation εξαρτάται από τη σωστή ρύθμιση των υπερπαραμέτρων, όπως το ποσοστό μάθησης και το μέγεθος των δεσμίδων δεδομένων που επιλέγονται (batch). Εκπαίδευση, με σοβαρή απόκλιση από τον επιθυμητό στόχο είναι απόρροια της λάθους επιλογής των υπερπαραμέτρων.

Αυτά τα μειονεκτήματα καθιστούν την οπισθοδιάδοση λιγότερο αποδοτική σε ορισμένες περιπτώσεις, ειδικά όταν τα δεδομένα είναι θορυβώδη ή όταν απαιτείται γρήγορη εκπαίδευση.

Για τη βελτιστοποίηση της μεθόδου backpropagation και την επίτευξη καλύτερης απόδοσης στα νευρωνικά δίκτυα, μπορούν να εφαρμοστούν οι παρακάτω τεχνικές:

Ρύθμιση Υπερπαραμέτρων. Η σωστή ρύθμιση υπερπαραμέτρων όπως το ποσοστό μάθησης (learning rate), το μέγεθος των δεσμίδων που θα επιλεγούν (batches), και η επιλογή του πλήθους των «εποχών», μπορεί να βελτιώσει σημαντικά την απόδοση του μοντέλου. Χρησιμοποιώντας τεχνικές όπως η αναζήτηση πλέγματος (grid search) ή η τυχαία αναζήτηση (random search) μπορεί να βρεθούν οι καλύτερες ρυθμίσεις.

Κανονικοποίηση (Normalization). Η κανονικοποίηση των εισόδων μπορεί να βοηθήσει στην ταχύτερη σύγκλιση του αλγορίθμου. Η χρήση τεχνικών όπως η κανονικοποίηση «min-max» ή η τυπική κανονικοποίηση (standardization) μπορεί να μειώσει τα προβλήματα που σχετίζονται με τις διαφορετικές κλίμακες των χαρακτηριστικών.

Χρήση κατάλληλων Συναρτήσεων Ενεργοποίησης. Η επιλογή ειδικών συναρτήσεων ενεργοποίησης, όπως η ReLU (Rectified Linear Unit) ή οι παραλλαγές της (Leaky ReLU, Parametric ReLU), μπορεί να βελτιώσει την εκπαίδευση και να μειώσει το πρόβλημα της γρήγορης καθόδου και απόσβεσης-εξαφάνισης της κλίσης (vanishing gradient).

Μείωση της Υπερβολικής Προσαρμογής (Overfitting). Η κανονικοποίηση L1/L2, η χρήση dropout και η διαίρεση των δεδομένων σε σύνολα εκπαίδευσης (training sets), σύνολα

επαλήθευσης (validation sets) και σύνολα ελέγχου (test sets), μειώνει την υπερβολική προσαρμογή.

Πτώση (κάθοδος) κλίσης σε μικρές δεσμίδες Mini-Batch Gradient Descent: Ο διαχωρισμός και η τυχαία επιλογή μικρών δεσμίδων δεδομένων (mini-batches) για την εκπαίδευση βελτιώνει την ταχύτητα και την απόδοση της οπισθοδιάδοσης, επειδή υλοποιεί την εκπαίδευση του αλγόριθμου σε μικρότερα μέρη των δεδομένων, απαιτώντας μικρότερο ποσοστό χρήσης της μνήμης και έτσι επιταχύνοντας την επεξεργασία των δεδομένων.

Προσαρμοσμένοι Ρυθμοί Εκμάθησης (Adaptive Learning Rates) . Η χρήση ειδικών μεθόδων βελτιστοποίησης όπως των Adam, RMSprop ή Adagrad, που προσαρμόζουν το ποσοστό μάθησης κατά τη διάρκεια της εκπαίδευσης μπορεί να οδηγήσει σε καλύτερη σύγκλιση και απόδοση των νευρωνικών δικτύων.

Γρήγορη Διακοπή (Early Stopping) : Η παρακολούθηση της απόδοσης του μοντέλου σε ένα σύνολο επικύρωσης (validation set) και η έγκαιρη διακοπή της εκπαίδευσης όταν η απόδοση αρχίσει να μειώνεται, καταλήγει στην αποφυγή της υπερβολικής προσαρμογής.
Αυτές οι στρατηγικές μπορούν να συνδυαστούν για να βελτιώσουν την απόδοση της οπισθοδιάδοσης, και να διασφαλίσουν ότι το νευρωνικό δίκτυο εκπαιδεύεται αποτελεσματικά και αποδοτικά.

2.3.4. Πολυεπίπεδος Αναγνωριστής (Αντίληπτρο) Multilayer Perceptron

Ένας Πολυεπίπεδος Αναγνωριστής (MLP) είναι ένας τύπος τεχνητού νευρωνικού δικτύου που αποτελείται από πολλαπλά επίπεδα νευρώνων, συμπεριλαμβανομένου ενός επιπέδου εισόδου, ενός ή περισσότερων κρυφών επιπέδων και ενός επιπέδου εξόδου (McCulloch, 1943).

Αυτή η αρχιτεκτονική επιτρέπει στους Πολυεπίπεδους Αναγνωριστές (MLPs) να μοντελοποιούν περίπλοκες σχέσεις στα δεδομένα, καθιστώντας τους κατάλληλους για διάφορες εργασίες στη μηχανική μάθηση και την βαθιά μάθηση.

Η δομική τους κατασκευή είναι η ακόλουθη: Πρώτα, διαθέτει ένα Επίπεδο Εισόδου το οποίο δέχεται τα αρχικά δεδομένα. Κάθε νευρώνας σε αυτό το επίπεδο, αντιστοιχεί σε ένα χαρακτηριστικό ή μια παράμετρο του συνόλου δεδομένων.

Στη συνέχεια διαθέτει τα λεγόμενα Κρυφά (για την ακριβή μετάφραση από την Αγγλική ορολογία «κρυμμένα») Επίπεδα. Οι MLPs περιέχουν ένα ή περισσότερα κρυφά επίπεδα όπου πραγματοποιούνται διάφοροι υπολογισμοί. Κάθε νευρώνας σε αυτά τα επίπεδα, επεξεργάζεται τις εισόδους από το προηγούμενο επίπεδα, χρησιμοποιώντας βάρη και εφαρμόζει μια μη γραμμική συνάρτηση ενεργοποίησης (π.χ., sigmoid, ReLU) για να παραγάγει εξόδους που μεταφέρονται στα επόμενα επίπεδα.

Το επίπεδο εξόδου παράγει τις τελικές προβλέψεις ή ταξινομήσεις με βάση τους μετασχηματισμούς που εκτελούνται στα κρυφά επίπεδα. Ο αριθμός των νευρώνων σε αυτό το επίπεδο, αντιστοιχεί στον αριθμό των επιθυμητών εξόδων (π.χ., για δυαδική ταξινόμηση, μπορεί να υπάρχει ένας νευρώνας εξόδου).

Η λειτουργία των MLPs μπορεί να αναλυθεί σε διάφορες διαδικασίες υπολογισμών:

Η πρώτη, είναι η Εμπροσθοδιάδοση, κατά την οποία, τα δεδομένα εισόδου τροφοδοτούνται στο δίκτυο, και κάθε νευρώνας υπολογίζει ένα σταθμισμένο άθροισμα των εισόδων του, ακολουθούμενο από μια συνάρτηση ενεργοποίησης. Αυτή η διαδικασία συνεχίζεται σε όλα τα επίπεδα, μέχρι να παραχθεί η έξοδος.

Η αντίστροφη, είναι η διαδικασία της Οπισθοδιάδοσης η οποία πραγματοποιείται για τη βελτίωση της ορθότητας. Οι MLPs, χρησιμοποιούν την οπισθοδιάδοση, μια μέθοδο για την διόρθωση των βαρών βάσει του σφάλματος της πρόβλεψης. Ο αλγόριθμος υπολογίζει τις παραγώγους της συνάρτησης απώλειας σε σχέση με κάθε βάρος και τις προσαρμόζει ώστε να ελαχιστοποιήσει τα σφάλματα πρόβλεψης κατά την εκπαίδευση.

Η χρηστικότητα των MLPs τους χαρακτηρίζει ως ιδιαίτερα αποτελεσματικούς στις ακόλουθες περιπτώσεις:

Όταν υπάρχουν μη Γραμμικές Σχέσεις. Σε αντίθεση με τους Αναγνωριστές ενός Επιπέδου, οι οποίοι μπορούν να μοντελοποιήσουν μόνο γραμμικά διαχωρίσιμα δεδομένα, οι MLPs μπορούν να μάθουν σύνθετες μη γραμμικές συναρτήσεις λόγω των πολλών επιπέδων τους και των μη γραμμικών συναρτήσεων ενεργοποίησης.

Στην ύπαρξη Πολυμορφίας. Οι MLPs μπορούν να εφαρμοστούν σε διάφορους τομείς, συμπεριλαμβανομένης της αναγνώρισης εικόνας, της επεξεργασίας φυσικής γλώσσας και της πρόβλεψης χρονοσειρών. Είναι ικανοί να χειριστούν τόσο ταξινόμηση όσο και παλινδρόμηση.
Όταν απαιτείται Υψηλή Ορθότητα. Με τη σωστή εκπαίδευση και ρύθμιση, οι MLPs συχνά υπερβαίνουν απλούστερα μοντέλα επιτυγχάνοντας υψηλότερα ποσοστά ορθότητας και καλύτερη γενίκευση σε άγνωστα δεδομένα.

Οι MLPs είναι επίσης, κατάλληλοι για χρήση σε Πολύπλοκα Μοτίβα Δεδομένων. Αφορά, τα δεδομένα που παρουσιάζουν πολύπλοκα μοτίβα και που δεν μπορούν να ταξινομηθούν από γραμμικά μοντέλα.

Στην περίπτωση των Μεγάλου Όγκου Δεδομένων, οι MLPs, λειτουργούν αποτελεσματικά σε μεγάλα σύνολα δεδομένων όπου οι παραδοσιακοί αλγόριθμοι μπορεί να δυσκολεύονται.

Όταν απαιτούνται διεργασίες Επιβλεπόμενης Μάθησης, οι MLPs χρησιμοποιούνται κυρίως σε επιβλεπόμενες διαδικασίες εκμάθησης, όπου είναι διαθέσιμα μερικά ταξινομημένα δεδομένα για εκπαίδευση.

Παρουσιάζουν τα εξής πλεονεκτήματα: Ικανότητα μοντελοποίησης σύνθετων μη γραμμικών σχέσεων, υψηλή ορθότητα όταν εκπαιδεύονται σωστά, και γρήγορους χρόνους πρόβλεψης μετά την εκπαίδευση.

Όμως παρουσιάζουν και τα εξής μειονεκτήματα: Είναι υπολογιστικά απαιτητικοί, απαιτώντας σημαντικούς πόρους για εκπαίδευση, είναι επιρρεπείς σε υπερβολική προσαρμογή αν δεν κανονικοποιηθούν σωστά ή αν εκπαιδευτούν με ανεπαρκή δεδομένα. Τέλος, η απόδοσή τους, εξαρτάται σε μεγάλο βαθμό από τη ρύθμιση υπερπαραμέτρων.

Γενικά, οι Πολυεπίπεδοι Αναγνωριστές συνιστούν ένα θεμελιώδες μοντέλο στη βαθιά μάθηση που έχει ανοίξει το δρόμο για πιο προηγμένες αρχιτεκτονικές. Η ικανότητά τους να μαθαίνουν από περίπλοκα σύνολα δεδομένων τους καθιστούν ισχυρά εργαλεία σε διάφορες εφαρμογές, από την ταξινόμηση εικόνας έως την επεξεργασία φυσικής γλώσσας. Η κατανόηση της δομής τους και της λειτουργίας τους είναι κρίσιμη για την αποτελεσματική αξιοποίηση των δυνατοτήτων τους σε πραγματικά προβλήματα.

2.3.5. Συνελικτικά Νευρωνικά Δίκτυα. Convolutional Neural Networks

Η συνέλιξη είναι μια μαθηματική πράξη, μέρος της επεξεργασία ψηφιακών σημάτων για τον προσδιορισμό της συσχέτισης μεταξύ δύο σημάτων (Anirudha Ghosh, 2020,). Εφόσον οι ψηφιακές εικόνες θεωρούνται ως απόρροια ψηφιακών σημάτων, η συνέλιξη χρησιμοποιείται και για να βρεθεί η συσχέτιση μεταξύ δύο εικόνων. Εάν μία από τις εικόνες είναι μικρότερου μεγέθους, τότε η συνέλιξη υπολογίζει τη συσχέτιση της μικρότερης εικόνας με κάθε δυνατό πλαίσιο ίδιου μεγέθους στην μεγαλύτερη εικόνα. Για ένα διακριτό σήμα x(n) μήκους N και ένα φίλτρο f(n) μήκους M, που ορίζεται μεταξύ των ακεραίων -M/2 και M/2, η συσχέτιση μεταξύ τους ορίζεται σαν :

$$(x * f)[n] = \sum_{-M/2}^{M/2} x[n-a]f[a]$$

Εάν έχουμε μια δισδιάστατη εικόνα X μεγέθους M×M και έναν δισδιάστατο πυρήνα μεγέθους N×N, η πράξη της συνέλιξης συνοψίζεται στην παρακάτω εξίσωση. Η τιμή που προκύπτει στη θέση (i,j) ενός εικονοστοιχείου, είναι το εσωτερικό γινόμενο μεταξύ των τιμών ενός παραθύρου

μεγέθους N×N της εικόνας, με κέντρο το εικονοστοιχείο στη θέση (i,j), και του πυρήνα (φίλτρου) που αναφέρθηκε πριν. Η εξίσωση υπολογισμού της τιμής του κεντρικού εικονοστοιχείου είναι:

$$(x * w)[i,j] = \sum_{-M/2}^{M/2} . \sum_{b=-N/2}^{N/2} x(i-a, j-b)w(a,b)$$

Ένα συνελικτικό νευρωνικό δίκτυο αποτελείται από κάποια μεμονωμένα ή και κάποιες ομάδες συνελικτικών επιπέδων και επιπέδων συνένωσης (ή συγκέντρωσης ή συγχώνευσης) ακολουθούμενα από ένα ή πολλαπλά πλήρως συνδεδεμένα (Fully Connected) επίπεδα και ένα επίπεδο εξόδου. Το συνελικτικό επίπεδο είναι το κυρίαρχο δομικό στοιχείο στην αρχιτεκτονική των συνελικτικών νευρωνικών δικτύων.

Το συνελικτικό ή τα συνελικτικά επίπεδα αποτελούνται από πολλούς πυρήνες (ή φίλτρα) συνέλιξης με δυνατότητα εκμάθησης, που χρησιμοποιούνται για τον υπολογισμό διαφορετικών απεικονίσεων των χαρακτηριστικών. Κάθε μονάδα του επιπέδου απεικόνισης των χαρακτηριστικών συνδέεται με ένα πεδίο αποδοχής στο προηγούμενο επίπεδο. Η νέα απεικόνιση των χαρακτηριστικών παράγεται από την συνέλιξη της εισόδου με τους πυρήνες και την εφαρμογή μη γραμμικής συνάρτησης ενεργοποίησης σε κάθε στοιχείο του συνελικτικού αποτελέσματος. Η ιδιότητα του καταμερισμού των παραμέτρων του συνελικτικού επιπέδου μειώνει την πολυπλοκότητα του μοντέλου (You, 2017).

Το επίπεδο συνένωσης (συγκέντρωσης) ή υπο-δειγματοληψίας παίρνει μια μικρή περιοχή της συνελικτικής εξόδου ως είσοδο και μειώνει τη δειγματοληψία της για να παράγει μία μόνο έξοδο. Υπάρχουν διαφορετικές τεχνικές υπο- δειγματοληψίας όπως για παράδειγμα max pooling, min pooling, average pooling, κ.λπ.

Η συνένωση (pooling) μειώνει τον αριθμό των παραμέτρων που πρέπει να υπολογιστούν καθώς και καθιστά τον μετασχηματισμό του δικτύου αμετάβλητο.

Η συνένωση είναι μια τεχνική που χρησιμοποιείται στα συνελικτικά νευρωνικά δίκτυα (CNN) για τη μείωση των χωρικών διαστάσεων των (επιπέδων) απεικονίσεων των χαρακτηριστικών,

διατηρώντας παράλληλα τις πιο σημαντικές πληροφορίες. Περιλαμβάνει την κατάτμηση της εικόνας των χαρακτηριστικών εισόδου σε μικρές περιοχές και την εφαρμογή της συνέλιξης, με αποτέλεσμα τη μέγιστη ή τη μέση συνένωση, σε κάθε περιοχή για να παραχθεί μια ενιαία τιμή εξόδου για αυτήν την περιοχή.

Η δημιουργία των επιπέδων συνένωσης στα συνελικτικά νευρωνικά δίκτυα επιτυγχάνει την μείωση των διαστάσεων των επιπέδων που υφίστανται την συνένωση έτσι ώστε τα επίπεδα που προκύπτουν, να έχουν μικρότερο αριθμό παραμέτρων και να απαιτούνται λιγότεροι υπολογισμοί στο δίκτυο αφού έχουν μειωθεί οι χωρικές διαστάσεις των απεικονισμένων χαρακτηριστικών.

Επίσης, συντελεί στην διατήρηση (Αμεταβλητότητα) των χαρακτηριστικών, αφού τα επίπεδα συνένωσης καθιστούν το δίκτυο πιο σταθερό σε μικρούς μετασχηματισμούς της εικόνας εισόδου, διατηρώντας τα πιο σημαντικά χαρακτηριστικά, και εξαλείφοντας τα τελείως δευτερεύοντα.

Ένα άλλο σημαντικό όφελος της συνένωσης είναι η μείωση της υπερπροσαρμογής (overfitting), λόγω της ελάττωσης του αριθμού των παραμέτρων (λόγω μείωσης των διαστάσεων) Τα επίπεδα συνένωσης συντελούν στην μείωση της υπερπροσαρμογής και στην δυνατότητα γενίκευσης του μοντέλου.

Υπάρχουν δύο κύριοι τύποι διαδικασίας συνένωσης.

Η μέγιστη (τιμή) συγκέντρωσης (Max Pooling). Αυτή η λειτουργία επιλέγει τη μέγιστη τιμή από κάθε περιοχή συγκέντρωσης. Διατηρεί τα πιο σημαντικά χαρακτηριστικά και απορρίπτει λιγότερο σχετικές πληροφορίες.

Μέση συγκέντρωση. Αυτή η λειτουργία υπολογίζει τη μέση τιμή κάθε περιοχής συγκέντρωσης. Εξομαλύνει τον χάρτη χαρακτηριστικών διατηρώντας τα βασικά χαρακτηριστικά. Μια κοινή διαμόρφωση επιπέδου συνένωσης είναι η χρήση ενός φίλτρου 2x2 με βήμα 2, το οποίο μειώνει τις χωρικές διαστάσεις του χάρτη χαρακτηριστικών κατά 2 σε πλάτος και 2 σε ύψος.

Τα επίπεδα συνένωσης προστίθενται συνήθως μετά από συνελικτικά επίπεδα στην αρχιτεκτονική των συνελικτικών νευρωνικών δικτύων. Βοηθούν στο να γίνει η αναπαράσταση περίπου αμετάβλητη σε μικρές διαφοροποιήσεις της εισόδου, που σημαίνει ότι εάν η είσοδος διαφοροποιηθεί κατά ένα μικρό ποσό, οι τιμές των περισσότερων από τις ενοποιημένες εξόδους δεν αλλάζουν σημαντικά.

Συνοπτικά, τα επίπεδα συγκέντρωσης είναι ένα ουσιαστικό συστατικό των CNN, παρέχοντας μείωση διαστάσεων, αμετάβλητη μετάφραση και μείωση υπερπροσαρμογής, που τελικά βελτιώνει την απόδοση και την αποδοτικότητα του μοντέλου

2.3.6. Λειτουργικά σε Ακτινική Βάση Νευρωνικά Δίκτυα. Radial Basis Functional Neural Networks

Τα Νευρωνικά Δίκτυα Συναρτήσεων Ακτινωτής Βάσης (RBFNN) είναι μια κατηγορία τεχνητών νευρωνικών δικτύων που χρησιμοποιούν τις ακτινωτής βάσης συναρτήσεις (radial basis functions) ως συναρτήσεις ενεργοποίησης. Αυτά τα δίκτυα είναι ιδιαίτερα αποτελεσματικά σε εφαρμογές όπως η προσαρμογή συναρτήσεων, η πρόβλεψη χρονοσειρών και η ταξινόμηση.

Ένα RBFNN συνήθως περιλαμβάνει τρία κύρια επίπεδα:

Επίπεδο Εισόδου: Μεταφέρει τα δεδομένα εισόδου στους νευρώνες του κρυφού επιπέδου.

Κρυφό Επίπεδο: Κάθε νευρώνας σε αυτό το επίπεδο, εφαρμόζει μια ακτινωτή συνάρτηση (συνήθως Γκαουσιανή) στα εισερχόμενα δεδομένα. Η έξοδος του νευρώνα υπολογίζεται με βάση την απόσταση του εισερχόμενου διανύσματος από το κέντρο της ακτινωτής συνάρτησης.

Επίπεδο Εξόδου: Υπολογίζει τη γραμμικά συνδυασμένη συνάρτηση των εξόδων του κρυφού επιπέδου για να παραγάγει την τελική έξοδο του δικτύου.

Η λειτουργία ενός RBFNN πραγματοποιείται ως εξής : Το διάνυσμα εισόδου τροφοδοτείται στο επίπεδο εισόδου. Οι νευρώνες της κρυφού επιπέδου, υπολογίζουν την απόσταση του εισερχόμενου

διανύσματος από τα κέντρα τους, χρησιμοποιώντας μια ακτινωτή συνάρτηση ως συνάρτηση ενεργοποίησης. Το επίπεδο εξόδου υπολογίζει την τελική έξοδο ως γραμμικό συνδυασμό των εξόδων του κρυφού επιπέδου (Soper, 2023).

Τρία είναι τα βασικά Πλεονεκτήματα των RBFNN. Αποτελεσματικότητα στην Προσέγγιση Συναρτήσεων. Τα RBFNN είναι ικανά να προσεγγίζουν οποιαδήποτε συνεχή συνάρτηση, σύμφωνα με το θεώρημα της καθολικής προσέγγισης. Γρήγορη Εκπαίδευση. Η διαδικασία εκπαίδευσης είναι συχνά πιο γρήγορη σε σύγκριση με άλλα είδη νευρωνικών δικτύων, καθώς οι βάσεις μπορούν να καθοριστούν μέσω αλγορίθμων όπως ο K-means. Καλή Απόδοση σε Μη Γραμμικά Προβλήματα. Ικανότητα να χειρίζονται μη γραμμικά προβλήματα και πολύπλοκες σχέσεις μεταξύ μεταβλητών.

Ωστόσο, υπάρχουν και τα εξής μειονεκτήματα. Ανάγκη για Ρύθμιση Παραμέτρων. Απαιτούν προσεκτική ρύθμιση παραμέτρων, όπως οι θέσεις των κέντρων και οι παράμετροι διάχυσης των ακτινωτών συναρτήσεων. Καταλληλόλητα για μεγάλες και πολλές διαστάσεις. Μπορεί να μην είναι κατάλληλα για δεδομένα μεγάλων και πολλών διαστάσεων, επειδή η απόδοσή τους μειώνεται καθώς αυξάνεται ο αριθμός των διαστάσεων. Περιορισμένες Γενικεύσεις. Σε ορισμένες περιπτώσεις, μπορεί να εμφανίζουν προβλήματα γενίκευσης αν δεν ρυθμιστούν σωστά ή αν εκπαιδευτούν με περιορισμένο σύνολο δεδομένων.

Τα RBFNN αποτελούν ένα ισχυρό εργαλείο στη μηχανική μάθηση, με μοναδικές δυνατότητες στην ανάλυση και την πρόβλεψη δεδομένων.

Τα RBFNN, χρησιμοποιούν διάφορα σχήματα εκπαίδευσης για την αποτελεσματική προσαρμογή των παραμέτρων τους.

Τα κύρια σχήματα εκπαίδευσης που χρησιμοποιούνται για τα RBFNN είναι:

Η Εκπαίδευση με K-means. Αυτή η μέθοδος περιλαμβάνει τη χρήση του αλγορίθμου K-means για τον καθορισμό των κέντρων των ακτινωτών συναρτήσεων. Ο αλγόριθμος ομαδοποιεί τα δεδομένα εισόδου σε K ομάδες και το κέντρο κάθε ομάδας χρησιμοποιείται ως κέντρο για τους νευρώνες

του κρυφού επιπέδου. Πρόκειται για μια γρήγορη και απλή στη χρήση μέθοδο, επιτρέποντας την εύκολη προσαρμογή των κέντρων.

Η Ορθογώνια Μέθοδος που περιλαμβάνει τη χρήση ορθογώνιας αποσύνθεσης για την επιλογή των κέντρων και τη βελτιστοποίηση των βαρών. Οι νευρώνες προστίθενται ή αφαιρούνται με βάση τη συμβολή τους στην απόδοση του δικτύου. Η μέθοδος αυτή, βελτιώνει την απόδοση και μειώνει την πολυπλοκότητα του δικτύου, επιτρέποντας καλύτερη γενίκευση.

Η Μέθοδος των Ελαχίστων Τετραγώνων, κατά την οποία, αφού καθοριστούν τα κέντρα, χρησιμοποιείται για την εκπαίδευση των βαρών του επιπέδου εξόδου. Αυτή η μέθοδος ελαχιστοποιεί το σφάλμα μεταξύ της πραγματικής εξόδου και της προβλεπόμενης. Επίσης, η μέθοδος αυτή παρέχει μια απλή και αποτελεσματική προσέγγιση για την εκπαίδευση του δικτύου.

Η Διαδικασία Εκπαίδευσης με Επανάληψη. Σε αυτή την περίπτωση, το δίκτυο εκπαιδεύεται επαναληπτικά προσθέτοντας νέα δεδομένα ή τροποποιώντας τα υπάρχοντα κέντρα και βάρη με βάση την απόδοση του δικτύου. Έχει το πλεονέκτημα ότι, επιτρέπει στο δίκτυο να προσαρμόζεται δυναμικά σε νέα δεδομένα.

Και τέλος οι Μέθοδοι Κανονικοποίησης, οι οποίες σχετίζονται με την κανονικοποίηση των παραμέτρων του δικτύου ώστε να αποφεύγεται η υπερβολική προσαρμογή (overfitting). Αυτές οι μέθοδοι περιλαμβάνουν τεχνικές όπως η κανονικοποίηση L2 ή L1 (Aggarwal, 2018).

Αυτές οι μέθοδοι βοηθούν στη βελτίωση της γενίκευσης του μοντέλου.

Αυτές οι τεχνικές εκπαίδευσης επιτρέπουν στα RBFNN να προσαρμόζονται αποτελεσματικά σε διάφορες εφαρμογές, όπως η ταξινόμηση, η πρόβλεψη χρονοσειρών και η ανάλυση δεδομένων.

2.3.7. Ανατροφοδοτούμενα Νευρωνικά Δίκτυα Recurrent Neural Networks

Τα ανατροφοδοτούμενα νευρωνικά δίκτυα (RNN) είναι μια κατηγορία τεχνητού νευρωνικών δικτύων, ιδιαίτερα κατάλληλα για εφαρμογές σε ακολουθίες, σειρές και λίστες δεδομένων με αλυσιδωτή κατασκευή στην σχηματική τους γραφική εμφάνιση.

Ένα ανατροφοδοτούμενο νευρωνικό δίκτυο (RNN) ανήκει στα μοντέλα βαθιάς μάθησης και εκπαιδεύεται να επεξεργάζεται μια σειριακή είσοδο δεδομένων και να την μετατρέπει σε μια συγκεκριμένη διαδοχική έξοδο δεδομένων. Τέτοια σειριακά δεδομένα μπορεί να είναι λέξεις, προτάσεις ή δεδομένα χρονοσειρών, όπου τα σειριακά στοιχεία συνδέονται μεταξύ τους με διάφορους κανόνες συντακτικούς ή σημειολογικούς. Οι εφαρμογές τους είναι ποικίλες σε σχεδόν όλα τα δεδομένα που εμφανίζουν σειριακή μορφή, όπως ψηφιακά βίντεο, ψηφιακές εικόνες και ψηφιακοί ήχοι αλλά και σε δεδομένα με μορφή χρονοσειρών. Μπορούν να αναλύσουν και αρκετά περίπλοκα δεδομένα όπως τις σχετικά επαναλαμβανόμενες δομές του DNA, τις χρονολογικές συναλλαγές μετοχών με σκοπό τον προσδιορισμό ασυνήθιστων συναλλαγών, την αλληλουχία των υπότιτλων σε βίντεο και φίλμς, την δημιουργία γλωσσικών μοντέλων και πολλά άλλα.

Το βασικό και κύριο χαρακτηριστικό τους είναι ότι διατηρούν δεδομένα και πληροφορίες που επεξεργάστηκαν σε προηγούμενο εγγύτερο χρόνο χάριν των βρόχων ανατροφοδότησης που διαθέτουν (G. Toderici, 2017).

Το κυρίαρχο αναγνωριστικό των RNN είναι τα κρυφά επίπεδα μνήμης τα οποία θυμούνται κάποιες πληροφορίες για μια προηγούμενη σειρά ή ακολουθία. Αυτή η κατάσταση μνήμης θυμάται την προηγούμενη είσοδο στο δίκτυο.

Για παράδειγμα σε μια χρονολογική σειρά κάποιων καιρικών μεταβλητών, όπως για παράδειγμα της θερμοκρασίας, για να προβλεφθεί η επόμενη και η μεθεπόμενη θερμοκρασία πρέπει να γνωρίζει το νευρωνικό δίκτυα και κυρίως να διατηρεί την μνήμη των προηγούμενων θερμοκρασιών

Εκτελώντας την ίδια διεργασίες σε όλες τις εισόδους, χρησιμοποιεί τις ίδιες παραμέτρους για κάθε είσοδο καθώς ή για τα κρυφά επίπεδα για την παραγωγή της εξόδου. Αυτό καθιστά την σύνθεση ως προς τις παραμέτρους πιο απλή στα ανατροφοδοτούμενα νευρωνικά δίκτυα, σε σχέση με κάποια άλλα νευρωνικά δίκτυα.

Η βασική τους μονάδα επεξεργασίας, η οποία ουσιαστικά είναι μια μονάδα ανατροφοδότησης, έχει την ιδιότητα να διατηρεί ένα κρυφό επίπεδο το οποίο έχει καταγεγραμμένες προηγούμενες σχέσεις διότι «θυμάται» κάποιες προηγούμενες εισόδους. Όμως οι εξαρτήσεις αυτές δεν διατηρούν μνήμες από αρκετά επίπεδα πριν στην αλληλουχία των επιπέδων που υπάρχουν και γι' αυτό έχουν δημιουργηθεί τα μοντέλα «Μακροπρόθεσμης-Βραχυπρόθεσμης Μνήμης» (LSTM), ή «Περιφραγμένης Ανατροφοδοτούμενης Μονάδας» (GRU),τα οποία μειώνουν την αδυναμία αυτή.

Τα Ανατροφοδοτούμενα Νευρωνικά Δίκτυα (Α.Ν.Δ.), βασικά έχουν την ίδια φιλοσοφία κατασκευής με τα δίκτυα Βαθιάς Μηχανικής Μάθησης. Ωστόσο, ενώ στην πληθώρα των Βαθιών Νευρωνικών δικτύων, τα βάρη από επίπεδο σε επίπεδο αλλάζουν, εδώ στα Α.Ν.Δ., τα βάρη σε όλο το επίπεδο παραμένουν τα ίδια. Ουσιαστικά υπάρχει επικαιροποίηση των κρυφών επιπέδων «μνήμης» σε κάθε χρονικό βήμα. Δηλαδή η τρέχουσα κατάσταση την χρονική στιγμή t, είναι μια συνάρτηση της κατάστασης την χρονική στιγμή t-1, και της κατάστασης εισόδου x την χρονική στιγμή t. Για τους υπολογισμούς των βαρών και τις διορθωτικές αναπροσαρμογές τους, χρησιμοποιείται η μέθοδος της οπισθοδιάδοσης, μόνον που εδώ εφαρμόζεται προς τα πίσω στον χρόνο. Γι' αυτό και είναι γνωστή ως οπισθοδιάδοση στον χρόνο (Backpropagation Through time – BPTT).

Η μεθοδολογία της οπισθοδιάδοσης εφαρμόζεται προς τα πίσω στον χρόνο στις συναρτήσεις των κρυμμένων επιπέδων και για τα βάρη (τα οποία διατηρούνται τα ίδια). Η ελαχιστοποίηση της συνάρτησης απώλειας(κόστους) έστω "L", πραγματοποιείται μέσω διαδοχικών αλυσιδωτών παραγωγίσεων της συνάρτησης "L", ως προς τις προηγούμενες συναρτήσεις των κρυφών επιπέδων "i", έστω "h_i", και αυτών ως προς τα βάρη από τα οποία εξαρτώνται, έστω "w_i" :

$$\frac{\partial L}{\partial w_i} = \left(\frac{\partial L}{\partial y_{Pred}}\right)\left(\frac{\partial y_{Pred}}{\partial h_i}\right)\left(\frac{\partial h_i}{\partial w_i}\right)$$

όπου y_{pred} η προβλεπόμενη τιμή από το Ν.Δ.

Αυτή είναι γνωστή ως μέθοδος «Οπισθοδιάδοσης στον χρόνο».

Η εκπαίδευση ενός ανατροφοδοτούμενου νευρωνικού δικτύου (RNN), γίνεται με διαδοχικά βήματα. Σε πρώτο χρόνο δίνονται τα δεδομένα εισόδου. Στη συνέχεια υπολογίζονται οι τιμές των τρεχουσών καταστάσεων βάσει των αρχικών και των τιμών των προηγούμενων καταστάσεων. Αφού υπολογισθεί η τιμή κάθε επιπέδου, χρησιμοποιείται ως επίπεδο εισόδου για το επόμενο επίπεδο. Το πλήθος των βημάτων εξαρτάται από το κάθε πρόβλημα, και αφού υπολογισθούν οι τιμές σε όλα τα επίπεδα, υπολογίζεται η τιμή εξόδου και συγκρίνεται με την αναμενόμενη τιμή. Η διαφορά τους (σφάλμα) προωθείται προς τα πίσω επίπεδα στα προηγούμενα χρονικά βήματα και οι διορθώσεις των βαρών πραγματοποιούνται μέσω της οπισθοδιάδοσης μέσω του χρόνου (BPTT).

Τα ανατροφοδοτούμενα νευρωνικά δίκτυα παρουσιάζουν πλεονεκτήματα και μειονεκτήματα. Τα πλεονεκτήματά τους είναι ότι αφενός «θυμούνται» την προηγούμενη κατάσταση μέσα στο χρόνο, αλλά και υπάρχει η δυνατότητά τους να συνδυαστούν με τα συνελικτικά νευρωνικά δίκτυα και έτσι να αποκτήσουν «γνώση», όχι μόνον χρονική, που από κατασκευής τους έχουν αλλά και χωρική χάρις στην αντίστοιχη ιδιότητα της συνέλιξης, που εφαρμόζεται μέσω «χωρικών» φίλτρων.

Στα μειονεκτήματά τους συγκαταλέγονται οι μεγάλες αλλαγές που είναι δυνατόν να συμβούν κατά την εφαρμογή της μεθόδου της καθόδου κλίσης, για την καλύτερη δυνατή προσέγγιση του ελαχίστου της συνάρτησης απώλειας. Αυτές είναι η «έκρηξη» δηλ. η πολύ απότομη και υψηλή αλλαγή της κλίσης προς μεγαλύτερες τιμές ή η «εξαφάνιση» της κλίσης προς πολύ χαμηλές τιμές. Οι συναρτήσεις ενεργοποίησης της υπερβολικής εφαπτομένης (tanh) ή της τροποποιημένης γραμμικής μονάδας (Relu) δεν οδηγούν σε ικανοποιητικά αποτελέσματα, όταν τα δεδομένα σχηματίζουν πολύ μεγάλες αλληλουχίες.

Στην προσπάθεια για να ξεπεραστούν τα προβλήματα υπερβολικά μεγάλης ή υπερβολικά μικρής κλίσης, έχουν προταθεί διάφορες παραλλαγές των Α. Ν. Δ. , οι κυριότερες από τις οποίες είναι τα Νευρωνικά Δίκτυα Αμφίδρομης Κατεύθυνσης (Bidirectional Neural Networks) και τα μοντέλα Μακροπρόθεσμης – Βραχυπρόθεσμης Μνήμης (Long Short Term Memory - L.S.T.M.)

Τα Ν.Δ. αμφίδρομης κατεύθυνσης είναι ιδιαίτερα αποδοτικά στις περιπτώσεις Επεξεργασίας Φυσικών Γλωσσών και Ανάλυσης Χρονολογικών Σειρών.

Τα δίκτυα L.S.T.M. δίκτυα λειτουργούν με βάση την αρχή Ανάγνωση – Εγγραφή – Απάλειψη (από τη μνήμη), και βάσει αυτής εγγράφουν την πλέον χρήσιμη πληροφορία για την πρόβλεψη του τελικού αποτελέσματος, «ξεχνώντας» τις λιγότερο χρήσιμες πληροφορίες.

Οι κατηγορίες των ανατροφοδοτούμενων νευρωνικών δικτύων είναι τέσσερις ανάλογα με το πλήθος των επιπέδων εισόδου και το αντίστοιχο εξόδου του δικτύου. Είναι το ένα σε ένα, ένα σε πολλά, πολλά σε ένα και πολλά σε πολλά. Το ένα σε ένα γνωστό και σαν Βανίλια Νευρωνικό Δίκτυο, έχει την ίδια δομή με ένα απλό νευρωνικό δίκτυο. Το ένα σε πολλά εφαρμόζεται κυρίως όταν δίνεται μια εικόνα και προβλέπεται μια φράση με πολλές λέξεις. Αντίθετα με τον προηγούμενο τύπο το πολλά σε ένα χρησιμοποιείται κυρίως στην ανάλυση συναισθημάτων όπου δοθέντων πολλών λέξεων προβλέπεται το συναίσθημα που προκαλεί. Τέλος το μοντέλο πολλά σε πολλά, χρησιμοποιείται κύρια για την μετάφραση από μια γλώσσα σε μια άλλη όπου δίνεται μια αλληλουχία λέξεων της μιας γλώσσας και παράγεται μια αλληλουχία λέξεων της άλλης γλώσσας.

2.3.8. Νευρωνικά Δίκτυα Μακροπρόθεσμης – Βραχυπρόθεσμης Μνήμης Long Short Term Memory (L.S.T.M. Neural Networks)

Τα νευρωνικά δίκτυα μακράς και βραχείας μνήμης (LSTM) είναι ένας εξειδικευμένος τύπος επαναλαμβανόμενου νευρωνικού δικτύου (RNN) που έχει σχεδιαστεί για να αντιμετωπίσει τα προβλήματα που προκαλούνται στη μηχανική μάθηση από διαδοχικά δεδομένα. Είναι ιδιαίτερα αποτελεσματικά στη διατήρηση πληροφοριών για μεγάλες περιόδους, κάτι που είναι κρίσιμο για εργασίες όπου το σύνολο από προηγούμενες εισόδους επηρεάζει τις τρέχουσες εξόδους.

Η αρχιτεκτονική ενός δικτύου LSTM αποτελείται από κύτταρα μνήμης και τρεις τύπους πυλών που διαχειρίζονται τη ροή της πληροφορίας. Βασικό είναι το Κύτταρο Μνήμης. Αυτό είναι το βασικό στοιχείο που διατηρεί πληροφορίες με την πάροδο του χρόνου. Η Πύλη Λήθης. Καθορίζει ποιες πληροφορίες θα πρέπει να απορριφθούν από το κύτταρο μνήμης. Η Πύλη Εισόδου. Ελέγχει ποιες νέες πληροφορίες προστίθενται στο κύτταρο μνήμης. Και η Πύλη Εξόδου, που ρυθμίζει ποιες πληροφορίες εξάγονται από το κύτταρο μνήμης. Αυτές οι πύλες χρησιμοποιούν σιγμοειδείς (sigmoid) συναρτήσεις ενεργοποίησης για να παράγουν τιμές μεταξύ 0 και 1, υποδεικνύοντας το μέρος της πληροφορίας που θα πρέπει να διατηρηθεί ή να απορριφθεί.

Τα LSTM έχουν σχεδιαστεί για να καταπολεμούν δύο κύρια προβλήματα που αντιμετωπίζουν τα παραδοσιακά RNN: το πρόβλημα εξαφάνισης της κλίσης και το πρόβλημα εκρηκτικής αύξησης της κλίσης. Το πρόβλημα εξαφάνισης της κλίσης, συμβαίνει όταν οι κλίσεις καταλήγουν να γίνονται πολύ μικρές για αποτελεσματική μάθηση κατά τη διάρκεια της οπισθοδιάδοσης, ειδικά σε μεγάλες ακολουθίες. Τα LSTM μετριάζουν το πρόβλημα αυτό, επιτρέποντας στις κλίσεις να ρέουν μέσα από στην δομή των νευρωνικών δικτύων, χωρίς σημαντική μείωση, ωθώντας τες να μαθαίνουν και να εξαρτώνται από μεγαλύτερα χρονικά διαστήματα.

Η εσωτερική δομή των LSTM τους επιτρέπει να διατηρούν μια σταθερή ροή σφάλματος, η οποία βοηθά στην αποτελεσματική μάθηση εξαρτήσεων από μεγάλα χρονικά διαστήματα. Αυτό επιτυγχάνεται μέσω μηχανισμών όπως τα " καρουζέλ σταθερού σφάλματος " που βοηθούν στη διατήρηση των κλίσεων κατά την εκπαίδευση. Το καρουζέλ σταθερού σφάλματος,- Constant Error Carousel (CEC) - είναι μια σημαντική διάταξη που χρησιμοποιείται στα δίκτυα Long Short-Term Memory (LSTM) για να αντιμετωπίσει το πρόβλημα της εξαφάνισης των γραμμικών κλίσεων (vanishing gradients). Οι μονάδες CEC είναι ειδικές μονάδες μέσα στις αρχιτεκτονικές LSTM που διατηρούν μια σταθερή ροή σφάλματος. Χρησιμοποιούν τη ταυτοτική συνάρτηση ως συνάρτηση ενεργοποίησης, πράγμα που σημαίνει ότι η έξοδος είναι ίση με την είσοδο. Αυτή η σχεδίαση επιτρέπει να διατηρείται η κλίση σταθερή και να μην εξαφανίζεται κατά τη διάρκεια της ανατροφοδότησης. Η βασική λειτουργία των CECs είναι ότι επιτρέπουν στα σφάλματα να κυκλοφορούν χωρίς να αλλάζουν μέγεθος, όπως σε ένα καρουσέλ. Αυτό διασφαλίζει ότι σημαντικές πληροφορίες μπορούν να διατηρηθούν σε μεγάλες ακολουθίες. Με την διατήρηση ενός σταθερού σήματος σφάλματος, οι CECs βοηθούν στην αποφυγή του προβλήματος της εξαφάνισης γραμμικών κλίσεων, επιτρέποντας στο LSTM να μάθει από μεγαλύτερες ακολουθίες χωρίς να χάνει κρίσιμες πληροφορίες. Οι CECs μπορούν επίσης να αλληλεπιδρούν με άλλα μέρη του δικτύου, επιτρέποντας προσαρμοστική μάθηση μέσω πυλών που ελέγχουν τη ροή πληροφοριών μέσα και έξω από την μνήμη. Συνοψίζοντας, το καρουζέλ σταθερού σφάλματος, - Constant Error Carousel, είναι ένας καινοτόμος μηχανισμός μέσα στα LSTMs που διευκολύνει τη διατήρηση και την ανατροφοδότηση σφαλμάτων σε εκτεταμένα χρονικά διαστήματα, βελτιώνοντας σημαντικά την ικανότητα του μοντέλου να μαθαίνει από σειριακά δεδομένα.

Τα δίκτυα LSTM είναι ιδιαίτερα κατάλληλα σε εφαρμογές που περιλαμβάνουν διαδοχικά δεδομένα, όπως:

Επεξεργασία Φυσικής Γλώσσας (NLP): Εργασίες όπως η μετάφραση γλώσσας, η παραγωγή κειμένου και η ανάλυση συναισθημάτων έχουν σημαντική πρόοδο από την ικανότητα των LSTM να κατανοούν το πλαίσιο και τις εξαρτήσεις στη γλώσσα (Z. C. Lipton, 2015).

Αναγνώριση Ομιλίας: Τα LSTM μπορούν να επεξεργαστούν σήματα ήχου αποτελεσματικά, καθιστώντας τα κατάλληλα για εφαρμογές μετατροπής ομιλίας σε κείμενο.

Ανάλυση Βίντεο: Μπορούν να αναλύσουν ακολουθίες καρέ βίντεο για να αναγνωρίσουν μοτίβα ή δράσεις με την πάροδο του χρόνου.

Πρόβλεψη Χρονολογικών Σειρών: Κλάδοι της Οικονομίας και της Βιομηχανίας όπως η χρηματοδότηση και η υγειονομική περίθαλψη χρησιμοποιούν τα LSTM για την πρόβλεψη τάσεων βάσει ιστορικών δεδομένων.

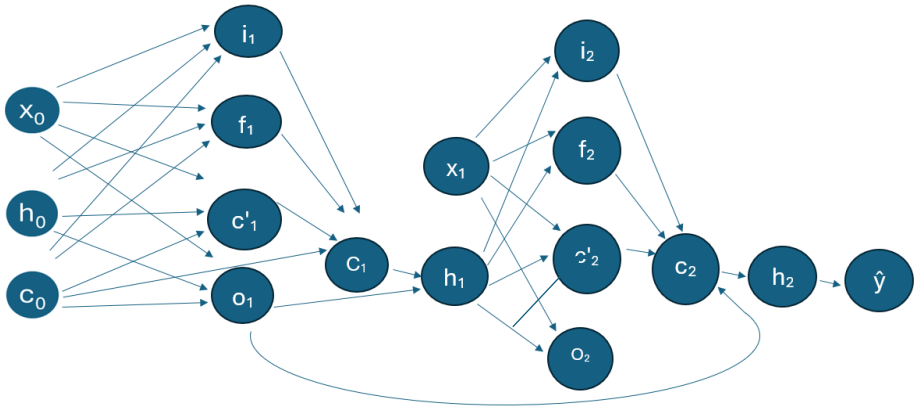
Διαπιστώνεται ότι τα LSTM προτιμώνται σε σχέση με τα παραδοσιακά RNN γιατί έχουν τα παρακάτω πλεονεκτήματα:

Μπορούν να θυμούνται (κατακρατούν) πληροφορίες για χιλιάδες χρονικά βήματα, και έτσι είναι ιδιαίτερα κατάλληλα για μελέτες και έρευνες που απαιτούν κατανόηση του συνόλου του νευρωνικού δικτύου σε μεγάλες ακολουθίες. Τα LSTM μπορούν να προσαρμοστούν σε διάφορα μοντέλα εκπαίδευσης και να συνδυαστούν με άλλες αρχιτεκτονικές, όπως τα Συνελικτικά Νευρωνικά Δίκτυα (CNN), για βελτιωμένη απόδοση σε σύνθετες εργασίες όπως η ανάλυση εικόνας και βίντεο. Και τέλος, ο σχεδιασμός τους τους επιτρέπει να αποφεύγουν κοινές παγίδες που σχετίζονται με την εκπαίδευση RNN, όπως η εξαφάνιση ή η έκρηξη της κλίσης.

Συνοψίζοντας, τα νευρωνικά δίκτυα μακράς και βραχείας μνήμης αντιπροσωπεύουν μια σημαντική πρόοδο στον τομέα των νευρωνικών δικτύων, ιδιαίτερα για εφαρμογές που απαιτούν την επεξεργασία διαδοχικών δεδομένων. Η μοναδική τους αρχιτεκτονική τους επιτρέπει να

μαθαίνουν και να θυμούνται αποτελεσματικά μακροχρόνιες εξαρτήσεις, καθιστώντας τα ανεκτίμητα σε πολλές σύγχρονες εφαρμογές τεχνητής νοημοσύνης.

Στο σχήμα που ακολουθεί (το οποίο αποτελεί μια τροποποίηση του βασικού σχήματος, από το εξαιρετικό αριθμητικό παράδειγμα του ιστότοπου “Statisticalinference” (Statisticalinference). παρουσιάζεται με άλλο τρόπο η ιεραρχική δομή ενός LSTM με δύο διαδοχικές τιμές εισόδου x_0 και x_1.



Σχήμα 2- Δομή ενός LSTM Νευρωνικού Δικτύου με δύο εισόδους

Οι λειτουργίες των Πυλών του παραπάνω Ν.Δ. (και των L.S.T.M. γενικότερα) , είναι:

Η πύλη εισόδου ελέγχει ποιες νέες πληροφορίες θα προστεθούν στην κατάσταση του κυττάρου μνήμης. Παρόμοια με την πύλη λήθης, λαμβάνει επίσης x_t και h_{t-1}, ως εισόδους και εφαρμόζει μια συνάρτηση sigmoid για να αποφασίσει ποιες τιμές θα ενημερωθούν. Κατά περίπτωση, χρησιμοποιεί μια συνάρτηση tanh για τη δημιουργία ενός διανύσματος νέων υποψήφιων τιμών που θα μπορούσαν να προστεθούν στην κατάσταση του κυττάρου.
Η εξίσωση για την πύλη εισόδου είναι:

$$i_t = \sigma \cdot (W_i \cdot [h_{t-1}, x_t] + b_i)$$

όπου :

i_t : η τιμή εξόδου της πύλης εισόδου την χρονική στιγμή t

σ: η σιγμοειδής συνάρτηση

W_i : το διάνυσμα βαρών της πύλης εισόδου

h_{t-1} : η τιμή εξόδου του προηγούμενου επιπέδου

x_t : η τιμή εις εισόδου στην πύλη εισόδου την χρονική στιγμή t

b_i: η απόκλιση (σταθερός όρος – bias) της τιμής της πύλης εισόδου

(ο κάτω δείκτης i σχετίζεται με την αγγλική λέξη input που αντιστοιχεί στα μεγέθη που αναφέρονται στην πύλη εισόδου)

Η πύλη λήθης καθορίζει ποιες πληροφορίες από την προηγούμενη κατάσταση μνήμης θα πρέπει να απορριφθούν. Λαμβάνει δύο εισόδους: την τρέχουσα είσοδο x_t και την προηγούμενη κρυφή κατάσταση h_{t-1}. Οι έξοδοι αυτών των εισόδων επεξεργάζονται μέσω μιας σιγμοειδούς συνάρτησης ενεργοποίησης (sigmoid), η οποία παράγει τιμές μεταξύ 0 και 1. Μια τιμή 0 υποδεικνύει ότι οι πληροφορίες θα πρέπει να ξεχαστούν εντελώς, ενώ μια τιμή 1 σημαίνει ότι θα πρέπει να διατηρηθούν. Η εξίσωση για την πύλη λήθης είναι:

$$f_t = \sigma \cdot \left(W_f \cdot [h_{t-1}, x_t] + b_f\right)$$

όπου :

f_t : η τιμή εξόδου της πύλης λήθης την χρονική στιγμή t

σ: η σιγμοειδής συνάρτηση

W_f: το βάρος που αντιστοιχεί στην τιμή της πύλης λήθης

h_{t-1} : η τιμή εξόδου του προηγούμενου επιπέδου

x_t : η τιμή εισόδου στην πύλη λήθης την χρονική στιγμή t

b_f : η απόκλιση (σταθερός όρος – bias) της τιμής της πύλης λήθης

(ο κάτω δείκτης και η συνάρτηση f σχετίζονται με την αγγλική λέξη forget που αντιστοιχεί στα μεγέθη που αναφέρονται στην πύλη λήθης)

Η τιμή εξόδου f_t της πύλης λήθης είναι δυαδική με τιμή 0 (μηδέν) για απαλοιφή τιμών και 1 για την διατήρησή τους.

Αντίστοιχα οι εξισώσεις για το (αρχικό) κύτταρο μνήμης και την πύλη εξόδου, την χρονική στιγμή t, θα είναι :

$$c_t' = \tanh(W_i[h_{t-1}, x_t] + b_c)$$

$$o_t = \sigma \cdot (W_0 \cdot [h_{t-1}, x_t] + b_0)$$

όπου ο μεν κάτω δείκτης «c» σχετίζεται με την αγγλική λέξη cell και αντιστοιχεί στα μεγέθη που αναφέρονται στο κύτταρο μνήμης, ο δε δείκτης «o», σχετίζεται με την αγγλική λέξη output και αντιστοιχεί στα μεγέθη που αναφέρονται στην τιμή της εξόδου.

Η εξίσωση για το κυρίως κύτταρο μνήμης θα είναι :

$$c_t = i_t \circ c'_t + f_t \circ c_{t-1}$$

όπου:

το σύμβολο «$\circ$» αντιστοιχεί στο γινόμενο πινάκων κατά Hadamard, για πίνακες ιδίου μεγέθους, δηλ. ο πολλαπλασιασμός κάθε στοιχείου του πρώτου με το στοιχείο στην ίδια θέση του δεύτερου πίνακα.

Η εξίσωση του κρυφού επιπέδου h_t , την χρονική στιγμή t, θα είναι :

$$h_t = o_t \circ \tanh(c_t)$$

Η πρώτη τελική πρόβλεψη $\hat{y}$, θα βρεθεί μέσω της εξίσωσης :

$$\hat{y} = W_y \cdot h_t + b_y$$

όπου W_y ο πίνακας βαρών που έχει ορισθεί για την πρόβλεψη, και b_y ο πίνακας των αποκλίσεων που αντιστοιχεί στην πρόβλεψη.

Και τέλος, το Μέσο Τετραγωνικό Σφάλμα (Mean Square Error – M.S.E.) υπολογίζεται από την εξίσωση :

$$E = \frac{1}{2} \cdot (\hat{y} - y_{target})^2$$

όπου y_{target} το επιδιωκόμενο αποτέλεσμα (τιμή στόχος).

Στα επόμενα στάδια της βελτιστοποίησης του νευρωνικού δικτύου, η παραμετροποίηση μέσω της επικαιροποίησης (διόρθωσης των αρχικών βαρών) επιτυγχάνεται με διαδοχικές – διορθωτικές επαναλήψεις προς τα πίσω (από την αρχική φορά του δικτύου) μέσω της οπισθοδιάδοσης ή διάδοσης προς τα πίσω:

Κατά την οπισθοδιάδοση, το σφάλμα διαδίδεται προς τα πίσω από το επίπεδο εξόδου και προς τα κρυφά επίπεδα. Χρησιμοποιώντας στα επόμενα στάδια λειτουργίας του νευρωνικού δικτύου, τον κανόνα της αλυσιδωτής παραγώγισης, υπολογίζονται οι παράγωγοι της συνάρτησης απώλειας ως προς την τιμή που προβλέφθηκε, της τιμής που προβλέφθηκε ως προς τις κρυφές συναρτήσεις, και των κρυφών συναρτήσεων ως προς τα βάρη, ξεκινώντας από το επίπεδο εξόδου και προχωρώντας προς τα πίσω μέσω των κρυφών επιπέδων.

Συνοπτικά, το backpropagation είναι μια αποτελεσματική μέθοδος για τον υπολογισμό των παραγώγων κατά την εκπαίδευση των νευρωνικών δικτύων, οδηγώντας σε αποτελεσματική ενημέρωση των βαρών για την ελαχιστοποίηση του σφάλματος.

Στην εφαρμογή των L.S.T.M. σε πραγματικά δεδομένα, ακολουθούνται διάφορες τακτικές, εκ των οποίων οι πιο κλασικές είναι: Πρώτο βήμα είναι η προεπεξεργασία των δεδομένων, η οποία πρέπει να αποτελείται από την κανονικοποίηση (normalization) των δεδομένων για την βέλτιστη απόδοση του μοντέλου. Ακολουθεί η σωστή διαχείριση και διόρθωση Ελλιπών – Εσφαλμένων – Περιττών Δεδομένων. Η συμπλήρωση η διόρθωση ή η αφαίρεση κάποιων από αυτά. Η εκκίνηση με μια απλή αρχιτεκτονική είναι βασική ώστε να υπάρχει η δυνατότητα αύξησης της πολυπλοκότητας σε περίπτωση που είναι αναγκαίο. Η εκπαίδευση του μοντέλου συνήθως πραγματοποιείται με μικρό ρυθμό εκμάθησης (Learning Rate) και στη συνέχεια με προσαρμογή του ρυθμού, ανάλογα με την απόδοση. Οι δοκιμές με διαφορετικών μεγεθών δειγματικά δεδομένα (δεδομένα δεσμίδων batch size) βοηθούν στο να βρεθεί το βέλτιστο μέγεθος (Cheng J. P., 2016).

Η διασταυρούμενη επικύρωση για την αξιολόγηση της απόδοσης του μοντέλου αποτελεί συνήθη μέθοδο αξιολόγησης του μοντέλου, και η Υπερπαραμετροποίηση (Hyperparameter Tuning), με δοκιμές από διαφορετικές τιμές για τις υπερπαραμέτρους βελτιώνουν την απόδοση. Τέλος κρίνεται απαραίτητη (όπως σχεδόν σε κάθε τύπο νευρωνικών δικτύων), η συνεχής

παρακολούθηση της απόδοσης του μοντέλου σε πραγματικό χρόνο, και η ανανέωση των δεδομένων εκπαίδευσης για την διατήρηση της ενημέρωσης του μοντέλου.

2.3.9. Μοντέλα Ακολουθίας προς Ακολουθία - Sequence to Sequence Models

Τα μοντέλα ακολουθίας προς ακολουθία (Seq2Seq) είναι μια ισχυρή κατηγορία αρχιτεκτονικών μηχανικής μάθησης που χρησιμοποιούνται κυρίως για εργασίες που περιλαμβάνουν διαδοχικά δεδομένα, όπως η γλωσσική μετάφραση, η περίληψη κειμένων και η μοντελοποίηση συνομιλιών. Μετασχηματίζουν μια είσοδο ακολουθίας σε μια έξοδο ακολουθίας, επιτρέποντας επιμήκεις μεταβλητές και στις δύο πλευρές.

Τα μοντέλα Seq2Seq αποτελούνται συνήθως από δύο κύρια στοιχεία: τον κωδικοποιητή που επεξεργάζεται την είσοδο ακολουθίας και συμπιέζει τις πληροφορίες σε ένα διάνυσμα σχετιζόμενων εκφράσεων, σταθερού μεγέθους (ή κρυφή κατάσταση). Αυτό το διάνυσμα περιλαμβάνει τις βασικές πληροφορίες από ολόκληρη την είσοδο ακολουθίας, οι οποίες στη συνέχεια μεταφέρονται στον αποκωδικοποιητή ο οποίος παίρνει το διάνυσμα συμφραζομένων και παράγει την έξοδο ακολουθίας ένα-ένα στοιχείο τη φορά. Προβλέπει κάθε επόμενο στοιχείο με βάση τις προηγούμενες εξόδους του και το διάνυσμα σχετιζόμενων εκφράσεων που παρέχεται από τον κωδικοποιητή. Αυτή η διαδικασία συνεχίζεται μέχρι να παραχθεί ένα ειδικό τέλος ακολουθίας, υποδεικνύοντας ότι η έξοδος είναι ολοκληρωμένη. Πολλά σύγχρονα μοντέλα Seq2Seq ενσωματώνουν έναν μηχανισμό προσοχής, ο οποίος επιτρέπει στον αποκωδικοποιητή να εστιάζει σε διάφορα μέρη της εισόδου ακολουθίας σε κάθε βήμα παραγωγής εξόδου. Αυτό είναι ιδιαίτερα χρήσιμο για μεγαλύτερες ακολουθίες όπου ορισμένα στοιχεία εισόδου μπορεί να είναι πιο σχετικά από άλλα σε διάφορα στάδια αποκωδικοποίησης.

Τα μοντέλα Seq2Seq έχουν βρει εφαρμογές σε διάφορους τομείς όπως 1) η Μηχανική Μετάφραση όπου παρουσιάζουν μεγάλο ποσοστό επιτυχίας στη μετάφραση κειμένου από μια γλώσσα σε άλλη, χαρτογραφώντας ακολουθίες από μια γλώσσα - πηγή σε μια γλώσσα - στόχο, 2) η περίληψη κειμένων, όπου μπορούν να παράγουν συνοπτικές περιλήψεις από μεγαλύτερα κείμενα

κατανοώντας και συμπυκνώνοντας πληροφορίες, 3) Οι πράκτορες συνομιλίας, που χρησιμοποιούνται σε chatbots και εικονικούς βοηθούς και που βοηθούν στη δημιουργία ανθρώπινων απαντήσεων με βάση τις εισόδους των χρηστών. 4) Οι λεζάντες εικόνας, που δημιουργούνται για εικόνες ερμηνεύοντας διαδοχικά δεδομένα από οπτικές εισόδους.

Η ευρεία χρήση των Μοντέλων Seq2Seq αιτιολογείται γιατί τα μοντέλα Seq2Seq μπορούν να χειριστούν εισόδους και εξόδους μεταβλητού μήκους, καθιστώντας τα κατάλληλα για πολλές πραγματικές εφαρμογές. Επίσης, με την αξιοποίηση κωδικοποιητών και αποκωδικοποιητών, αυτά τα μοντέλα συλλαμβάνουν τις σχέσεις συμφραζομένων μέσα στις ακολουθίες, βελτιώνοντας την ικανότητά τους να παράγουν συνεκτικές εξόδους.

Τα Μοντέλα Seq2Seq είναι κατάλληλα για χρήση σε:

Εργασίες Επεξεργασίας Γλώσσας, όπου οι εργασίες περιλαμβάνουν τον μετασχηματισμό μιας ακολουθίας σε άλλη—ιδιαίτερα όταν ασχολούνται με την επεξεργασία φυσικής γλώσσας—, τότε, τα μοντέλα Seq2Seq είναι ιδανικά.

Πολύπλοκα Διαδοχικά Δεδομένα, είναι ιδιαίτερα αποτελεσματικά όταν ασχολούνται με πολύπλοκα δεδομένα όπου οι σχέσεις μεταξύ στοιχείων είναι κρίσιμες, όπως στη λεζάντα βίντεο ή στην αναγνώριση ομιλίας.

Προκλήσεις στην Επεξεργασία Μακρών Εισόδων σε Μοντέλα Ακολουθίας προς Ακολουθία
Η διαχείριση μακρών εισερχόμενων ακολουθιών σε μοντέλα ακολουθίας προς ακολουθία (Seq2Seq) παρουσιάζει αρκετές προκλήσεις που μπορούν να επηρεάσουν την απόδοση και την ορθότητα του μοντέλου.

Μία από τις κύριες δυσκολίες στη χρήση τους, είναι η σύλληψη των μακροχρόνιων εξαρτήσεων στα δεδομένα εισόδου. Τα παραδοσιακά επαναληπτικά νευρωνικά δίκτυ (RNN), συμπεριλαμβανομένων των δικτύων Long Short-Term Memory (LSTM), συχνά αντιμετωπίζουν δυσκολίες σε αυτό λόγω του προβλήματος εξαφάνισης του βαθμού, όπου οι βαθμοί γίνονται πολύ μικροί για να ενημερώσουν αποτελεσματικά τα βάρη για απομακρυσμένα χρονικά βήματα. Αυτό

μπορεί να οδηγήσει σε απώλεια σημαντικών πληροφοριών σχετικών εκφράσεων από νωρίτερα μέρη της ακολουθίας, καθιστώντας δύσκολη τη διατήρηση της συνοχής σε μακρές εισόδους.

Κατά την κωδικοποίηση μακρών ακολουθιών, ο κωδικοποιητής συμπιέζει όλες τις πληροφορίες εισόδου σε ένα ενιαίο διάνυσμα συμφραζομένων. Αυτό μπορεί να έχει ως αποτέλεσμα την απώλεια κρίσιμων λεπτομερειών ή συμφραζομένων, ιδιαίτερα αν η ακολουθία είναι εκτενής και περίπλοκη. Ως συνέπεια, ο αποκωδικοποιητής μπορεί να μην έχει αρκετές πληροφορίες για να παράγει ακριβή αποτελέσματα, οδηγώντας σε λιγότερο βέλτιστη απόδοση.

Τα μοντέλα Seq2Seq εκπαιδεύονται συνήθως χρησιμοποιώντας teacher forcing, όπου η αληθινή έξοδος παρέχεται σε κάθε χρονικό βήμα κατά τη διάρκεια της εκπαίδευσης. Ωστόσο, κατά την εξαγωγή, το μοντέλο παράγει τις δικές του προβλέψεις, οι οποίες μπορεί να διαφέρουν από τα δεδομένα εκπαίδευσης. Αυτή η διαφορά μπορεί να προκαλέσει προκατειλημμένη έκθεση, όπου το μοντέλο έχει άσχημη απόδοση σε μακρές ακολουθίες επειδή δεν έχει μάθει να διορθώνει τα δικά του λάθη κατά τη διάρκεια της παραγωγής. Η επεξεργασία μακρών ακολουθιών αυξάνει σημαντικά τις υπολογιστικές απαιτήσεις. Η διαδικασία αποκωδικοποίησης περιλαμβάνει την αναζήτηση μέσα από έναν μεγάλο συνδυαστικό χώρο για πιθανές εξόδους, γεγονός που μπορεί να είναι χρονοβόρο και απαιτητικό σε πόρους. Αυτή η πολυπλοκότητα μπορεί να εμποδίσει τις εφαρμογές πραγματικού χρόνου και να οδηγήσει σε πιο αργούς χρόνους εξαγωγής.

Υπερ- και Υπο-προσαρμογή

Λόγω του μεγάλου αριθμού παραμέτρων τους, τα μοντέλα Seq2Seq είναι επιρρεπή σε υπερβολική προσαρμογή στα δεδομένα εκπαίδευσης, ειδικά όταν ασχολούνται με μακρές ακολουθίες που περιέχουν θόρυβο ή μη σχετικές πληροφορίες. Αντίθετα, εάν δεν ρυθμιστούν σωστά ή αν τα δεδομένα εκπαίδευσης είναι ανεπαρκή, μπορεί επίσης να εμφανίσουν υπο-προσαρμογή, αποτυγχάνοντας να συλλάβουν τα υποκείμενα μοτίβα σε μεγαλύτερες ακολουθίες.

Η εκπαίδευση αποτελεσματικών μοντέλων Seq2Seq απαιτεί σημαντικές ποσότητες ζευγαρωμένων δεδομένων εισόδου-εξόδου. Σε πολλές περιπτώσεις, η απόκτηση επαρκών ποιοτικών δεδομένων εκπαίδευσης για μακρές ακολουθίες μπορεί να είναι δύσκολη και δαπανηρή, περιορίζοντας την ικανότητα του μοντέλου να γενικεύει καλά σε διαφορετικού τ'υπου δεδομένα.

Για την αντιμετώπιση αυτών των προκλήσεων, χρησιμοποιούνται συνήθως αρκετές στρατηγικές όπως : Οι Μηχανισμοί Προσοχής που επιτρέπουν στο μοντέλο να εστιάζει δυναμικά σε σχετικές περιοχές της εισόδου κατά τη διάρκεια της διαδικασίας αποκωδικοποίησης, βελτιώνοντας την ικανότητά του να διαχειρίζεται μεγαλύτερες εισόδους διατηρώντας σημαντικά συμφραζόμενα σε κάθε βήμα αποκωδικοποίησης. Τα Ιεραρχικά Μοντέλα μπορούν να βοηθήσουν στη σύλληψη διαφορετικών επιπέδων αφαίρεσης σε μεγάλες ακολουθίες, ενισχύοντας την απόδοση με την κατανομή περίπλοκων εισόδων σε διαχειρίσιμα στοιχεία. Οι Τεχνικές Κανονικοποίησης, όπως η εφαρμογή dropout και άλλων μεθόδων κανονικοποίησης, βοηθά στη μετρίαση της υπερβολικής προσαρμογής μειώνοντας την πολυπλοκότητα του μοντέλου και ωθώντας το, σε γενίκευση σε μη ορατά δεδομένα. Και τέλος, η αναζήτηση «Beam Decoding» (αποκωδικοποίησης σε ακτίνα) όπου αντί για μεθόδους αποκωδικοποίησης που επιλέγουν τον πιο πιθανό επόμενο όρο σε κάθε βήμα, η αναζήτηση beam εξετάζει πολλαπλές υποψήφιες ακολουθίες ταυτόχρονα, βελτιώνοντας την ποιότητα της εξόδου εξερευνώντας έναν ευρύτερο χώρο αναζήτησης.

2.3.10. Αρθρωτά Νευρωνικά Δίκτυα Modular Neural Networks (MNNs)

Τα Αρθρωτά νευρωνικά δίκτυα (MNNs) είναι μια προηγμένη προσέγγιση στην τεχνητή νοημοσύνη που ενισχύει τις δυνατότητες των παραδοσιακών νευρωνικών δικτύων, χωρίζοντας σύνθετα προβλήματα σε μικρότερα, πιο διαχειρίσιμα υπο-προβλήματα. Αυτή η μεθοδολογία αντλεί έμπνευση από βιολογικά συστήματα, όπου διαφορετικές περιοχές του εγκεφάλου ειδικεύονται σε διαφορετικά καθήκοντα. Τα MNNs αποτελούνται από πολλαπλά διασυνδεδεμένα νευρωνικά δίκτυα ή "μονάδες", καθένα από τα οποία έχει σχεδιαστεί για να χειρίζεται συγκεκριμένες πτυχές ενός προβλήματος, επιτρέποντας παράλληλη επεξεργασία και μεγαλύτερη ευελιξία.

Η μοναδική προσέγγιση τους στο επιθυμητό αποτέλεσμα, εφαρμόζει την αρχή «διαίρεσης και κατάκτησης», σπάζοντας μεγάλα προβλήματα σε μικρότερα, πιο διαχειρίσιμα μέρη. Κάθε μονάδα μπορεί να εκπαιδευτεί ανεξάρτητα, απλοποιώντας τη διαδικασία μάθησης και μειώνοντας τη συνολική πολυπλοκότητα. Τα MNNs ενθαρρύνουν την ποικιλία μεταξύ των μονάδων,

επιτρέποντας σε διαφορετικούς τύπους νευρωνικών δικτύων να συνεργάζονται. Αυτή η ποικιλία μπορεί να ενισχύσει την απόδοση εκμεταλλευόμενη τα πλεονεκτήματα διαφόρων αρχιτεκτονικών. Οι μονάδες σε ένα MNN μπορούν να λειτουργούν ταυτόχρονα, επιταχύνοντας σημαντικά τους χρόνους εκπαίδευσης σε σύγκριση με κλασσικά νευρωνικά δίκτυα. Κάθε μονάδα επεξεργάζεται την είσοδό της ανεξάρτητα (παράλληλη επεξεργασία) πριν συμβάλει σε μια τελική στρώση λήψης αποφάσεων.

Τα MNNs μπορούν να κατηγοριοποιηθούν με βάση το πόσο στενά ή χαλαρά είναι συνδεδεμένες οι μονάδες τους. MNNS με Σφιχτά Συνδεδεμένες Μονάδες, που σημαίνει ότι οι μονάδες τους, είναι στενά ενσωματωμένες και μοιράζονται πληροφορίες συχνά και MNNS με Χαλαρά Συνδεδεμένες Μονάδες, οι οποίες λειτουργούν πιο ανεξάρτητα, επικοινωνώντας λιγότερο συχνά αλλά εξακολουθώντας να συμβάλλουν σε έναν κοινό στόχο. Αυτή η δομή επιτρέπει ευελιξία στο σχεδιασμό και προσαρμοστικότητα στις μεθόδους εκπαίδευσης.

Τα πλεονεκτήματα των Αρθρωτών Νευρωνικών Δικτύων είναι: Η αποτελεσματικότητα που επιτυγχάνεται με τη μείωση του αριθμού των συνδέσεων και βαρών που πρέπει να διαχειριστούν ανά πάσα στιγμή τα MNNs και που έτσι, μπορούν να εκπαιδευτούν πιο γρήγορα από τα παραδοσιακά νευρωνικά δίκτυα. Η Ανθεκτικότητα που τα MNNs, λόγω της ανεξάρτητης φύσης των μονάδων τους, παρέχουν ανοχή σε σφάλματα. Αν μια μονάδα αποτύχει, οι άλλες μπορούν να συνεχίσουν τη λειτουργία τους χωρίς σημαντική υποβάθμιση της απόδοσης. Η σταδιακή μάθηση κατά την οποία τα MNNs μπορούν να μάθουν νέες εργασίες χωρίς να χρειάζεται επανεκπαίδευση ολόκληρου του συστήματος, καθώς μπορούν να προστεθούν νέες μονάδες ή να τροποποιηθούν υπάρχουσες ανεξάρτητα.

Τα αρθρωτά νευρωνικά δίκτυα είναι ιδιαίτερα χρήσιμα σε περιπτώσεις όπου είναι αναγκαία η σύνθετη επίλυση προβλημάτων που περιλαμβάνουν επικαλυπτόμενα μοτίβα ή απαιτούν πολλές διαφορετικές προσεγγίσεις, και που τα MNNs μπορούν να χειρίζονται υπο-μοτίβα μέσω εξειδικευμένων μονάδων. Επίσης σε περιπτώσεις που είναι αναγκαία η επεξεργασία σε πραγματικό χρόνο, σε εφαρμογές όπως η αναγνώριση εικόνας ή η επεξεργασία φυσικής γλώσσας, διότι μπορούν να εκμεταλλευτούν τις δυνατότητες παράλληλης επεξεργασίας για ταχύτερα

αποτελέσματα. Τέλος για την επίλυση προβλημάτων, που απαιτούν συνεχιζόμενη μάθηση και προσαρμογή, όπως η πρόβλεψη χρηματιστηριακών αγορών ή τα αυτόνομα οχήματα.

Τα Αρθρωτά Νευρωνικά Δίκτυα κρίνονται ιδιαίτερα κατάλληλα για την πρόβλεψη χρηματιστηριακών αγορών, επειδή χρησιμοποιούν διάφορες μονάδες εκπαιδευμένες σε διαφορετικούς δείκτες αγοράς για τη βελτίωση της ορθότητας πρόβλεψης. Επίσης, με την υλοποίηση προσαρμοστικών MNNs που μαθαίνουν από ποικιλία δεδομένων για την ενίσχυση της απόδοσης στην αναγνώριση χειρόγραφων χαρακτήρων. Είναι ιδιαίτερα κατάλληλα για συγχώνευση δεδομένων, συνδυάζοντας αποτελέσματα από πολλαπλές πηγές ή τύπους δεδομένων για τη βελτίωση διαδικασιών λήψης αποφάσεων.

Τα αρθρωτά νευρωνικά δίκτυα αποτελούν μια ισχυρή εξέλιξη στον τομέα της τεχνητής νοημοσύνης, επιτρέποντας πιο αποτελεσματική επεξεργασία και στιβαρές (robust) μηχανισμούς μάθησης. Η ικανότητά τους να αποσυνθέτουν σύνθετες διαδικασίες σε απλούστερες, τα καθιστά ανεκτίμητα σε διάφορες εφαρμογές σε τομείς, όπως η χρηματοδότηση, η υγειονομική περίθαλψη και η τεχνολογία. Καθώς η έρευνα συνεχίζεται, η δυνατότητα των MNNs θα μπορούσε πιθανώς να επεκταθεί περαιτέρω, οδηγώντας σε ακόμα πιο καινοτόμες χρήσεις στην AI.

Η εκπαίδευση αρθρωτών νευρωνικών δικτύων (MNNs) παρουσιάζει αρκετές μοναδικές ιδιαιτερότητες, που μπορούν να επηρεάσουν την αποτελεσματικότητα και την αποδοτικότητά τους. Ο ενσωμάτωση διαφορετικών μονάδων και ο συντονισμός τους ανάγεται στην ανάλυση σύνθετων αλληλεπιδράσεων. Τα MNNs αποτελούνται από πολλές μονάδες που πρέπει να συνεργάζονται ομοιόμορφα. Η εξασφάλιση ότι αυτές οι μονάδες επικοινωνούν αποτελεσματικά και ενσωματώνουν τις εξόδους τους σε μία συνεπή απάντηση μπορεί να είναι περίπλοκη. Συχνά απαιτείται ένα σύστημα ελέγχου για τη διαχείριση αυτής της ενσωμάτωσης, προσθέτοντας ένα ακόμα επίπεδο πολυπλοκότητας στη διαδικασία εκπαίδευσης. Οι διαφόρου τύπου μονάδες μπορεί να εξαρτώνται η μία από την άλλη για ορισμένες εισόδους ή εξόδους, περιπλέκοντας τη διαδικασία εκπαίδευσης καθώς οι αλλαγές σε μία μονάδα μπορεί να επηρεάζουν άλλες.

Αν και τα MNNs μπορούν να μειώσουν τον αριθμό των βαρών σε σύγκριση με κλασσικά δίκτυα, η ανάγκη εκπαίδευσης πολλών μονάδων μπορεί να οδηγήσει σε μεγαλύτερους συνολικούς

χρόνους εκπαίδευσης. Κάθε μονάδα μπορεί να απαιτεί το δικό της σύνολο δεδομένων εκπαίδευσης, κάτι που μπορεί να περιπλέξει τη διαχείριση δεδομένων και να αυξήσει τον χρόνο που απαιτείται για την επίτευξη σύγκλισης.

Η διαδικασία του προς τα πίσω υπολογισμού της κλίσης, στα MNNs είναι πιο περίπλοκη καθώς περιλαμβάνει τον υπολογισμό κλίσεων για πολλές μονάδες. Αυτό μπορεί να εισαγάγει θόρυβο και αστάθεια στην εκπαίδευση, ιδιαίτερα αν δεν είναι όλες οι μονάδες σχετικές για κάθε είσοδο. Επίσης υπάρχουν και οι κίνδυνοι υπερβολικής προσαρμογής και κατάρρευσης κάποιων μονάδων. Οι ατομικές μονάδες μπορεί να υπερ-προσαρμοστούν σε συγκεκριμένα χαρακτηριστικά των δεδομένων εκπαίδευσης, ειδικά αν δεν είναι αρκετά ποικιλόμορφες ή αν τα δεδομένα εκπαίδευσης είναι περιορισμένα. Αυτό μπορεί να οδηγήσει σε κατάσταση όπου χρησιμοποιούνται επανειλημμένα μόνο λίγες μονάδες, μειώνοντας τη συνολική αποτελεσματικότητα του δικτύου. Επίσης, η εξασφάλιση ότι όλες οι μονάδες μαθαίνουν να γενικεύουν καλά χωρίς να καταρρέουν σε λίγες κυρίαρχες διαμορφώσεις είναι εύθραυστη. Τεχνικές κανονικοποίησης μπορεί να χρειαστούν για την επιβολή ποικιλίας μεταξύ των μονάδων, αλλά αυτές μπορούν αντίθετα, να περιπλέκουν τη διαδικασία εκπαίδευσης.

Η συγχώνευση των δεδομένων επίσης, αποτελεί έναν προβληματισμό, όταν συνδυάζονται οι έξοδοι από διαφορετικές μονάδες. Η εξασφάλιση ότι τα δεδομένα συγχωνεύονται κατάλληλα χωρίς απώλεια πληροφοριών ή εισαγωγή ευνοϊκών προϋποθέσεων για κάποια από αυτά, μπορεί να είναι δύσκολη. Αυτό απαιτεί προσεκτικό σχεδιασμό του τρόπου ροής των δεδομένων μεταξύ των μονάδων και του τρόπου συγκέντρωσης των εξόδων. Εξάλλου, η απόφαση σχετικά με το ποια μονάδα θα χειριστεί ποιο μέρος ενός καθήκοντος είναι δύσκολη, ειδικά σε δυναμικά περιβάλλοντα όπου τα καθήκοντα μπορεί να αλλάζουν με την πάροδο του χρόνου. Αυτό απαιτεί έναν προσαρμοστικό μηχανισμό που μπορεί να κατανεμηθεί αποτελεσματικά μεταξύ διαθέσιμων μονάδων.

Τέλος πρέπει να ελεγχθεί το πρόβλημα της κλιμάκωσης. Καθώς προστίθενται περισσότερες μονάδες για την αντιμετώπιση αυξανόμενης πολυπλοκότητας ή πρόσθετων καθηκόντων, η διαχείριση των αλληλεπιδράσεών τους και η διατήρηση της απόδοσης γίνεται ολοένα και πιο

δύσκολη. Η αρχιτεκτονική πρέπει να σχεδιαστεί ώστε να κλιμακώνεται αποτελεσματικά χωρίς σημαντική εισαγωγή υπερβολικού κόστους ή πολυπλοκότητας στην εκπαίδευση.

2.3.11. Νευρωνικά Δίκτυα Kolmogorov – Arnold

Τα Δίκτυα Kolmogorov-Arnold (KANs) είναι μια νέα κατηγορία νευρωνικών δικτύων που εμπνέονται από το θεώρημα αναπαράστασης Kolmogorov-Arnold. Αυτό το θεώρημα επιτρέπει την αποσύνθεση πολύπλοκων πολυδιάστατων συναρτήσεων σε άθροισμα απλών μονοδιάστατων συναρτήσεων. Οι εργασίες των Vladimir Arnold και Andrey Kolmogorov καθιέρωσαν ότι εάν η f είναι μια πολυμεταβλητή συνεχής συνάρτηση, τότε η f μπορεί να γραφτεί ως μια πεπερασμένη σύνθεση συνεχών συναρτήσεων μιας μεμονωμένης μεταβλητής και τη δυαδική πράξη της πρόσθεσης (Vl. Arnold, n.d.) .

Τα KANs εκμεταλλεύονται αυτή την αρχή εφαρμόζοντας μάθηση ενεργοποίησης στα άκρα του δικτύου αντί για τους κόμβους, που αποτελεί μια απόκλιση από τις παραδοσιακές αρχιτεκτονικές νευρωνικών δικτύων όπως οι Πολυεπίπεδοι Αντιληπτές (MLPs) που χρησιμοποιούν στατικές συναρτήσεις ενεργοποίησης στους κόμβους.

Τα κύρια χαρακτηριστικά των KANs είναι:

Η Μάθηση Ενεργοποίησης, όπου, κάθε βάρος σε ένα KAN σχετίζεται με μια συνάρτηση B-spline, επιτρέποντας δυναμική προσαρμογή κατά τη διάρκεια της εκπαίδευσης. Αυτό έρχεται σε αντίθεση με τα MLPs, όπου τα βάρη είναι στατικά και καθορίζονται αποκλειστικά από τη διαδικασία μάθησης. Τα KANs αντικαθιστούν τα παραδοσιακά γραμμικά βάρη με μονοδιάστατες συναρτήσεις, συγκεκριμένα B-splines, οδηγώντας σε πιθανώς πιο αναλυτικά μοντέλα που μπορούν να συλλάβουν πολύπλοκες σχέσεις στα δεδομένα πιο αποτελεσματικά.

Τα KANs επιπλέον, έχουν δείξει ότι προσφέρουν καλύτερη ερμηνευσιμότητα σε σύγκριση με τα MLPs, καθιστώντας τα κατάλληλα για εφαρμογές όπου η ακριβής αντίληψη της συμπεριφοράς του μοντέλου είναι κρίσιμη, όπως στην επιστημονική έρευνα.

Οι κυριότερες εφαρμογές των KANs είναι για:

Επιστημονικούς Υπολογισμούς. Η ικανότητά τους να μοντελοποιούν πολύπλοκα φυσικά φαινόμενα και μαθηματικούς νόμους τα καθιστά πολύτιμα εργαλεία σε τομείς όπως η φυσική και η μηχανική.

Προσαρμογή Δεδομένων: Τα KANs έχουν αποδείξει ανώτερη απόδοση στην προσαρμογή δεδομένων σε σύγκριση με παραδοσιακά MLPs, συχνά απαιτώντας λιγότερους παραμέτρους για να επιτύχουν συγκρίσιμη ορθότητα.

Επίλυση Μερικών Διαφορικών Εξισώσεων (Partial Difference Equations): Μπορούν να χρησιμοποιηθούν για την επίλυση PDEs πιο αποτελεσματικά λόγω της ευέλικτης δομής τους και των ικανοτήτων προσαρμοστικής μάθησης.

Τα KANs εμφανίζουν σημαντικά πλεονεκτήματα όπως:

Υψηλότερη Ορθότητα με Λιγότερες Παραμέτρους: Τα KANs μπορούν να επιτύχουν παρόμοια ή καλύτερη ορθότητα από μεγαλύτερα MLPs ενώ χρησιμοποιούν σημαντικά λιγότερες παραμέτρους, γεγονός που μπορεί να οδηγήσει σε ταχύτερους χρόνους εκπαίδευσης και μειωμένο υπολογιστικό κόστος.

Ενισχυμένη Ερμηνευσιμότητα: Η δομή των KANs επιτρέπει μια πιο διαισθητική οπτικοποίηση και κατανόηση του τρόπου που οι εισροές επηρεάζουν τις εξόδους, καθιστώντας τα κατάλληλα για τομείς που απαιτούν διαφάνεια.

Κλιμάκωση: Τα δίκτυα παρουσιάζουν ταχύτερους νόμους κλιμάκωσης νευρώνων από τα παραδοσιακά MLPs, επιτρέποντάς τους να προσαρμόζονται πιο αποτελεσματικά καθώς αυξάνεται η πολυπλοκότητα των δεδομένων.

Τα βασικά μειονεκτήματα των KANs είναι :

Οι Αργοί Χρόνοι Εκπαίδευσης. Η πολυπλοκότητα των προσεγγίσεων B-spline μπορεί να οδηγήσει σε μεγαλύτερους χρόνους εκπαίδευσης σε σύγκριση με απλές αρχιτεκτονικές MLP. Αυτό μπορεί να εμποδίσει την πρακτική εφαρμογή τους σε σενάρια όπου η γρήγορη ανάπτυξη μοντέλου είναι απαραίτητη.

Η Υπολογιστική Πολυπλοκότητα. Ενώ μπορούν να υπερτερούν σε ορισμένες εργασίες, η αυξημένη πολυπλοκότητα μπορεί να απαιτήσει πιο προηγμένο υλικό ή μεγαλύτερους χρόνους επεξεργασίας, κάνοντάς τα λιγότερο προσβάσιμα για ορισμένες εφαρμογές.

Η Περιορισμένη Γενίκευση. Αν και τα KANs έχουν δείξει αποτελεσματικότητα σε συγκεκριμένες εφαρμογές, οι ικανότητές τους για γενίκευση σε ποικιλία δεδομένων ενδέχεται να είναι περιορισμένες, ακόμη με εκείνες των πιο καθιερωμένων αρχιτεκτονικών MLP, οι οποίες έχουν δοκιμαστεί εκτενώς σε διάφορους τομείς.

Η υλοποίηση των Δικτύων Kolmogorov-Arnold (KANs) παρουσιάζει αρκετούς προβληματισμούς και ιδιαιτερότητες, που οι ερευνητές και οι επαγγελματίες πρέπει να αντιμετωπίσουν. Οι κυριότερες είναι :

Η Σταθερότητα της Εκπαίδευσης των KANs
Η διαδικασία εκπαίδευσης των KANs μπορεί να είναι ασταθής λόγω της εμπλοκής δεύτερης τάξης παραγώγων και αντιστρόφων πινάκων. Αυτές οι μαθηματικές πράξεις είναι ευαίσθητες σε αριθμητικά σφάλματα, γεγονός που μπορεί να οδηγήσει σε δυσκολίες στη σύγκλιση κατά τη διαδικασία βελτιστοποίησης. Αυτή η αστάθεια είναι ιδιαίτερα έντονη σε πολύπλοκα μοντέλα όπου οι αλληλεπιδράσεις μεταξύ πολλών συναρτήσεων B-spline μπορούν να περιπλέξουν τους υπολογισμούς στους βαθμούς που είναι απαραίτητοι για αποτελεσματική μάθηση.

Η Ρύθμιση των Υπερπαραμέτρων

Η αρχιτεκτονική των KANs περιλαμβάνει διάφορες υπερπαραμέτρους, όπως τον αριθμό των B-splines, τη διαμόρφωσή τους και τα σημεία ελέγχου που σχετίζονται με αυτές τις splines. Η εύρεση του βέλτιστου συνόλου υπερπαραμέτρων μπορεί να είναι δύσκολη και συχνά απαιτεί εκτενή πειραματισμό, ο οποίος μπορεί να είναι χρονοβόρος και απαιτητικός ως προς τους πόρους.

Η Υπολογιστική Πολυπλοκότητα

Τα KANs εισάγουν υψηλότερο επίπεδο υπολογιστικής πολυπλοκότητας σε σύγκριση με παραδοσιακά νευρωνικά δίκτυα όπως τα MLPs. Η χρήση B-splines ως συναρτήσεων ενεργοποίησης απαιτεί περισσότερους υπολογιστικούς πόρους τόσο για την εκπαίδευση όσο και για την εξαγωγή συμπερασμάτων. Αυτή η πολυπλοκότητα μπορεί να έχει ως αποτέλεσμα μεγαλύτερους χρόνους εκπαίδευσης και μπορεί να απαιτήσει πιο προηγμένο υλικό, καθιστώντας τα KANs λιγότερο προσβάσιμα για ορισμένες εφαρμογές.

Αν και τα KANs σχεδιάζονται για να ενισχύσουν την ερμηνευσιμότητα μέσω της δομής τους, η κατανόηση του τρόπου που κάθε B-spline συμβάλλει στην συνολική έξοδο του μοντέλου μπορεί ακόμα να είναι περίπλοκη. Οι περίπλοκες αλληλεπιδράσεις μεταξύ πολλών splines μπορεί να θολώσουν τις πληροφορίες σχετικά με τη συμπεριφορά του μοντέλου, καθιστώντας δύσκολο στους χρήστες να εξάγουν σαφείς εξηγήσεις από τις προβλέψεις του μοντέλου.

Η Περιορισμένη Γενίκευση

Ενώ τα KANs έχουν δείξει υποσχέσεις σε συγκεκριμένες εφαρμογές, οι ικανότητές τους για γενίκευση σε ποικιλία δεδομένων ενδέχεται να μην ισούνται ακόμη με εκείνες πιο καθιερωμένων αρχιτεκτονικών νευρωνικών δικτύων. Αυτός ο περιορισμός μπορεί να εμποδίσει την αποτελεσματικότητά τους σε ευρύτερα συμφραζόμενα όπου η μεταβλητότητα των δεδομένων είναι υψηλή.

Τα Προβλήματα Ερμηνευσιμότητας

Αν και τα KANs σχεδιάζονται για να ενισχύσουν την ερμηνευσιμότητα μέσω της δομής τους, η κατανόηση του τρόπου που κάθε B-spline συμβάλλει στην συνολική έξοδο του μοντέλου μπορεί ακόμα να είναι περίπλοκη. Οι περίπλοκες αλληλεπιδράσεις μεταξύ πολλών splines μπορεί να

θολώσουν τις πληροφορίες σχετικά με τη συμπεριφορά του μοντέλου, καθιστώντας δύσκολο στους χρήστες να εξάγουν σαφείς εξηγήσεις από τις προβλέψεις του μοντέλου.

Ενώ τα Δίκτυα Kolmogorov-Arnold προσφέρουν καινοτόμες προσεγγίσεις στην προσέγγιση συναρτήσεων και στη μοντελοποίηση δεδομένων, η υλοποίησή τους δεν είναι χωρίς προκλήσεις. Η αντιμετώπιση ζητημάτων σχετικά με τη σταθερότητα εκπαίδευσης, τις υπολογιστικές απαιτήσεις, τη ρύθμιση υπερπαραμέτρων, τη γενίκευση και την ερμηνευσιμότητα θα είναι κρίσιμη για την πραγματοποίηση της πλήρους δυναμικής τους στις πρακτικές εφαρμογές. Καθώς η έρευνα προχωράει, λύσεις σε αυτές τις προκλήσεις μπορεί να αναδυθούν, ενδυναμώνοντας περαιτέρω την εφαρμογή των KANs σε διάφορους τομείς.

Συμπερασματικά, τα Δίκτυα Kolmogorov-Arnold αντιπροσωπεύουν μια νεότερη τεχνολογία στο σχεδιασμό νευρωνικών δικτύων, δίνοντας έμφαση στην προσαρμοστικότητα και την δυνατότητα ερμηνείας μέσω της μοναδικής τους αρχιτεκτονικής βασισμένης στο θεώρημα αναπαράστασης Kolmogorov-Arnold. Ενώ προσφέρουν σημαντικά πλεονεκτήματα όσον αφορά την ορθότητα και τη διαφάνεια για επιστημονικές εφαρμογές, υπάρχουν προκλήσεις σχετικά με την αποδοτικότητα εκπαίδευσης και τις υπολογιστικές απαιτήσεις. Καθώς η έρευνα συνεχίζεται, οι λύσεις σε αυτές τις προκλήσεις μπορεί να αναδυθούν, ενισχύοντας περαιτέρω την δυνατότητα εφαρμογής των KANs σε διάφορους τομείς.

Ενώ τα Δίκτυα Kolmogorov-Arnold προσφέρουν καινοτόμες προσεγγίσεις στην προσέγγιση συναρτήσεων και στη μοντελοποίηση δεδομένων, η υλοποίησή τους δεν είναι χωρίς δυσκολίες. Η αντιμετώπιση ζητημάτων σχετικά με τη σταθερότητα εκπαίδευσης, τις υπολογιστικές απαιτήσεις, τη ρύθμιση υπερπαραμέτρων, τη γενίκευση και την ερμηνευσιμότητα θα είναι κρίσιμη για την πραγματοποίηση της πλήρους δυναμικής τους στις πρακτικές εφαρμογές. Καθώς η έρευνα προχωράει, λύσεις σε αυτές τις προκλήσεις μπορεί να αναδυθούν, ενδυναμώνοντας περαιτέρω την εφαρμογή των KANs σε διάφορους τομείς.

2.3.12. Κωδικοποιητές – Αποκωδικοποιητές Transformers

Οι κωδικοποιητές-αποκωδικοποιητές (transformers) είναι μια συγκεκριμένη αρχιτεκτονική μέσα στο ευρύτερο πλαίσιο των μοντέλων «transformer», που εισήχθη στην εμβληματική εργασία "Attention Is All You Need" (As. Vaswani, 2017). Αυτή η αρχιτεκτονική έχει δημιουργήσει νέες τάσεις σε διάφορους τομείς της τεχνητής νοημοσύνης (AI), ιδιαίτερα στην επεξεργασία φυσικής γλώσσας (NLP) και στη γενετική AI.

Η αρχιτεκτονική τους, αποτελείται από δύο κύρια στοιχεία: τον κωδικοποιητή και τον αποκωδικοποιητή. Ο ρόλος του κωδικοποιητή είναι να επεξεργάζεται τη σειρά εισόδου και να την μετατρέπει σε μια σειρά συνεχών αναπαραστάσεων ή ενσωματώσεων. Αποτελείται από πολλαπλά επίπεδα που εφαρμόζουν μηχανισμούς εσωτερικού ελέγχου και τροφοδοσίας των νευρωνικών δικτύων. Κάθε επίπεδο βοηθά το μοντέλο να κατανοήσει το πλαίσιο και τις σχέσεις μεταξύ διαφορετικών τόνων (λέξεων ή τμημάτων λέξεων) στη σειρά εισόδου. Ο αποκωδικοποιητής παίρνει τις κωδικοποιημένες αναπαραστάσεις και παράγει μια σειρά εξόδου, ένα τόνο τη φορά. Ενσωματώνει μηχανισμούς που του επιτρέπουν να προσέχει όχι μόνο τις προηγούμενες εξόδους του αλλά και τη κωδικοποιημένη σειρά εισόδου, επιτρέποντάς του να παράγει συνεκτικές και σχετικές εξόδους.

Οι Κωδικοποιητές-Αποκωδικοποιητές λειτουργούν ως εξής: Πρώτα πραγματοποιείται ο «Διαχωρισμός σε Τόνους», που σημαίνει ότι το κείμενο εισόδου διασπάται πρώτα σε μικρότερες μονάδες που ονομάζονται τόνοι. Μετά υλοποιείται η ενσωμάτωση, όπου κάθε τόνος μετατρέπεται σε μια αριθμητική διανυσματική αναπαράσταση μέσω ενός επιπέδου ενσωμάτωσης. Στη συνέχεια ο «Μηχανισμός Αυτοπροσοχής», επιτρέπει στο μοντέλο να ζυγίζει τη σημασία διαφορετικών τόνων σε σχέση με άλλους, ενισχύοντας την ικανότητά του να συλλαμβάνει το πλαίσιο σε μεγάλες σειρές.

Τέλος, ο αποκωδικοποιητής παράγει τους τόνους εξόδου διαδοχικά, χρησιμοποιώντας τους προηγούμενους παραγόμενους τόνους ως μέρος της εισόδου του για τις επόμενες προβλέψεις.

Οι κωδικοποιητές-αποκωδικοποιητές (transformers), χρησιμοποιούνται σε διάφορες εφαρμογές λόγω της ευελιξίας και της αποτελεσματικότητάς τους στην επεξεργασία διαδοχικών δεδομένων. Στην μηχανική μετάφραση, όπου χρησιμοποιούνται ιδιαίτερα για τη μετάφραση κειμένου από μια

γλώσσα σε άλλη, εκμεταλλευόμενοι την ικανότητά τους να κατανοούν το πλαίσιο και τις σημασιολογικές παραλλαγές.

Στην δημιουργία περίληψης κειμένων. Αυτά τα μοντέλα μπορούν να συμπυκνώσουν μεγάλα έγγραφα σε συντομότερες περιλήψεις διατηρώντας τις βασικές πληροφορίες.

Για να παρέχουν απαντήσεις σε ερωτήσεις. Μπορούν να παράγουν απαντήσεις με βάση το νοητικό πλαίσιο μιας ερώτησης, και έτσι είναι κατάλληλοι για "chatbots" και εικονικούς βοηθούς.

Για την δημιουργία κειμένου. Οι κωδικοποιητές-αποκωδικοποιητές μπορούν να δημιουργήσουν συνεκτικές αφηγήσεις ή απαντήσεις με βάση προτροπές, που χρησιμοποιούνται στη δημιουργική γραφή και στη δημιουργία περιεχομένου.

Τα πλεονεκτήματα που οδηγούν στην χρήση των κωδικοποιητών-αποκωδικοποιητών (transformers) είναι:

Η αποτελεσματικότητα στην Εκπαίδευση. Σε αντίθεση με τα παραδοσιακά ανακυκλικά νευρωνικά δίκτυα (RNNs), οι transformers μπορούν να επεξεργάζονται ολόκληρες σειρές ταυτόχρονα, μειώνοντας σημαντικά τον χρόνο εκπαίδευσης.

Η Κλιμάκωση. Είναι ικανοί να χειρίζονται μεγάλα σύνολα δεδομένων, γεγονός που είναι σημαντικό για την εκπαίδευση μεγάλων γλωσσικών μοντέλων (LLMs) που απαιτούν τεράστιες ποσότητες δεδομένων για αποτελεσματική μάθηση.

Η Ευελιξία σε Διάφορους Τομείς. Πέρα από τα NLPs, οι κωδικοποιητές-αποκωδικοποιητές έχουν προσαρμοστεί για χρήση στην υπολογιστική όραση (π.χ., Vision Transformers), στην επεξεργασία ήχου και ακόμη και στην ανάλυση βιολογικών ακολουθιών.
Προκλήσεις και Σκέψεις

Παρά τα πλεονεκτήματά τους, οι κωδικοποιητές-αποκωδικοποιητές transformers έχουν κάποια μειονεκτήματα που τα κυριότερα είναι:

Οι Υπολογιστικοί Πόροι. Απαιτούν σημαντική υπολογιστική ισχύ και μνήμη, ειδικά όταν κλιμακώνονται σε μεγαλύτερα μοντέλα με δισεκατομμύρια παραμέτρους.

Η Ταχύτητα εξαγωγής Συμπερασμάτων. Η πολυπλοκότητα αυτών των μοντέλων μπορεί να οδηγήσει σε πιο αργούς χρόνους κατάληξης συμπερασμάτων, σε σύγκριση με απλούστερες αρχιτεκτονικές, καθιστώντας δύσκολες τις εφαρμογές σε πραγματικό χρόνο.

Και τέλος, οι Απαιτήσεις Τροποποίησης. Ενώ τα προ-εκπαιδευμένα μοντέλα παρέχουν μια σταθερή βάση, η προσαρμογή σε συγκεκριμένες εργασίες είναι συχνά απαραίτητη για την επίτευξη βέλτιστης απόδοσης.

Συμπερασματικά, οι κωδικοποιητές-αποκωδικοποιητές, αντιπροσωπεύουν μια σημαντική πρόοδο στην αρχιτεκτονική AI, επιτρέποντας στις μηχανές να κατανοούν και να παράγουν ανθρώπινα κείμενα με αξιοσημείωτη ορθότητα. Οι εφαρμογές τους εκτείνονται σε πολλούς τομείς, καθιστώντας τους θεμελιώδη τεχνολογία στην ανάπτυξη σύγχρονων AI. Καθώς η έρευνα συνεχίζει να εξελίσσεται, μπορούμε να αναμένουμε περαιτέρω βελτιώσεις που θα επεκτείνουν τις δυνατότητες και την αποτελεσματικότητά τους.

Οι κωδικοποιητές-αποκωδικοποιητές χρησιμοποιούν ένα πλαίσιο κωδικοποίησης-απόκρυψης. Ο κωδικοποιητής επεξεργάζεται ολόκληρη τη σειρά εισόδου ταυτόχρονα χρησιμοποιώντας μηχανισμούς αυτό-προσοχής, ενώ ο αποκωδικοποιητής παράγει τη σειρά εξόδου διαδοχικά, προσέχοντας τόσο τις προηγούμενες εξόδους όσο και την έξοδο του κωδικοποιητή.

Τα Παραδοσιακά Νευρωνικά Δίκτυα, χρησιμοποιούν παραδοσιακές αρχιτεκτονικές όπως τα Αναδρομικά (Επαναληπτικά) Νευρωνικά Δίκτυια (RNNs) και τα Συνελικτικά Νευρωνικά Δίκτυα (CNNs) τα οποία συνήθως επεξεργάζονται δεδομένα διαδοχικά (στην περίπτωση των RNNs) ή μέσω τοπικών υποδοχών (στην περίπτωση των CNNs). Αυτή η διαδοχική επεξεργασία στα RNNs μπορεί να οδηγήσει σε δυσκολίες στη σύλληψη μακροπρόθεσμων εξαρτήσεων λόγω προβλημάτων εξαφάνισης της κλίσης.

Η βασική καινοτομία των transformers είναι ο μηχανισμός αυτό-προσοχής, ο οποίος επιτρέπει σε κάθε τόνο στη σειρά εισόδου να ζυγίζει τη σημασία του σε σχέση με άλλους τόνους. Αυτό επιτρέπει στο μοντέλο να συλλαμβάνει σχέσεις στο πλαίσιο χωρίς αναφορά στις θέσεις τους στη σειρά.

Τα παραδοσιακά μοντέλα συχνά βασίζονται σε σταθερές διαστάσεις παραθύρων συμφραζομένων ή διαδοχικής επεξεργασίας, περιορίζοντας την ικανότητά τους να εξετάζουν όλα τα μέρη της εισόδου ταυτόχρονα. Οι CNNs χρησιμοποιούν φίλτρα σύγκλισης που επικεντρώνονται μόνο σε τοπικά μοτίβα, ενώ οι RNNs επεξεργάζονται εισόδους μία προς μία, γεγονός που μπορεί να εμποδίσει την απόδοση σε εργασίες που απαιτούν παγκόσμιο πλαίσιο.

Οι transformers αντίθετα, μπορούν να επεξεργάζονται όλους τους τόνους μιας σειράς ταυτόχρονα λόγω της παράλληλης αρχιτεκτονικής τους. Αυτό επιταχύνει σημαντικά τους χρόνους εκπαίδευσης σε σύγκριση με τα RNNs, τα οποία πρέπει να επεξεργάζονται τις εισόδους διαδοχικά.

Τα Παραδοσιακά Νευρωνικά Δίκτυα, και ειδικότερα τα RNNs απαιτούν εγγενώς διαδοχική επεξεργασία, καθιστώντας τα πιο αργά στην εκπαίδευση καθώς δεν μπορούν να εκμεταλλευτούν αποτελεσματικά τις δυνατότητες σύγχρονου υλικού για παράλληλη υπολογισμό.

Στους Transformers, αντίθετα, οι τόνοι εισόδου ενσωματώνονται σε διανυσματικές αναπαραστάσεις υψηλής διάστασης και προστίθενται θέσεις ώστε να διατηρείται η πληροφορία τάξης. Αυτό επιτρέπει στους transformers να κατανοούν τη δομή της σειράς ενώ την επεξεργάζονται παράλληλα.

Στα παραδοσιακά Νευρωνικά Δίκτυα, η αναπαράσταση εισόδου συχνά βασίζεται σε απλούστερες τεχνικές ενσωμάτωσης ή άμεσες τιμές pixel (στην περίπτωση των CNNs για δεδομένα εικόνας), οι οποίες μπορεί να μην συλλαμβάνουν πολύπλοκες σχέσεις αποτελεσματικά.

Στους Transformers, ο αποκωδικοποιητής παράγει εξόδους αυτό-ρυθμιστικά, πράγμα που σημαίνει ότι παράγει έναν τόνο τη φορά με βάση προηγούμενους παραγόμενους τόνους και την κωδικοποιημένη είσοδο. Αυτό επιτρέπει τη συνεπή και σχετική παραγωγή κειμένου.

Σε μοντέλα όπως τα RNNs, οι έξοδοι παράγονται βάσει των κρυφών καταστάσεων που μεταφέρουν πληροφορίες από προηγούμενα χρονικά βήματα, γεγονός που μπορεί να οδηγήσει σε δυσκολίες στη διατήρηση μακριών εξαρτήσεων.

Οι Transformers, χρησιμοποιούνται ευρέως σε διάφορους τομείς όπως η επεξεργασία φυσικής γλώσσας (NLP), η υπολογιστική όραση και ακόμη και η επεξεργασία ήχου λόγω της ικανότητάς τους να χειρίζονται αποτελεσματικά διαδοχικά δεδομένα, ενώ τα παραδοσιακά Νευρωνικά Δίκτυα, αν και έχουν σημειώσει επιτυχία σε συγκεκριμένες εργασίες (π.χ., CNNs στην αναγνώριση εικόνας), συχνά δυσκολεύονται με εργασίες που απαιτούν πολύπλοκη κατανόηση ή παραγωγή σειρών.

Στους Transformers, ο μηχανισμός προσοχής τους, τους επιτρέπει να διατηρούν το πλαίσιο πάνω από μεγάλες σειρές χωρίς υποβάθμιση της απόδοσης, κάνοντάς τους ιδανικούς για εργασίες όπως η μετάφραση και η περίληψη, ενώ στα παραδοσιακά Νευρωνικά Δίκτυα, όπως για παράδειγμα, τα RNNs που συχνά αποτυγχάνουν με μεγάλες σειρές λόγω προβλημάτων όπως η εξαφάνιση των κλίσεων, καθιστώντας τα λιγότερο αποτελεσματικά για εργασίες που απαιτούν κατανόηση μακριών εξαρτήσεων.

Οι κωδικοποιητές-αποκωδικοποιητές transformers αντιπροσωπεύουν μια σημαντική πρόοδο στα παραδοσιακά νευρωνικά δίκτυα μέσω της αξιοποίησης μηχανισμών προσοχής, δυνατοτήτων παράλληλης επεξεργασίας και μιας ευέλικτης αρχιτεκτονικής που υπερβαίνει την ικανότητα χειρισμού πολύπλοκων διαδοχικών εργασιών. Η ικανότητά τους να συλλαμβάνουν το πλαίσιο και τις σχέσεις μέσα στα δεδομένα έχει καταστήσει αυτά τα μοντέλα ως την προτιμώμενη επιλογή για πολλές σύγχρονες εφαρμογές AI, ιδιαίτερα στην NLP και στις γεννητικές εργασίες.

2.4. Μέθοδοι Βελτιστοποίησης Νευρωνικών Δικτύων

2.4.1. Κάθοδος Κλίσης – Gradient Descent

Βασικές Αρχές της Μεθόδου Πτώσης Κλίσης (Gradient Descent)

Η μέθοδος πτώσης κλίσης είναι ένας από τους πιο διαδεδομένους αλγορίθμους βελτιστοποίησης που χρησιμοποιείται κυρίως στη μηχανική μάθηση για τη μείωση του σφάλματος μεταξύ των προβλεπόμενων και των πραγματικών αποτελεσμάτων (Han J., 1996). Ο στόχος της είναι να ελαχιστοποιήσει μια συνάρτηση κόστους ή απώλειας μέσω επαναληπτικών ενημερώσεων των παραμέτρων του μοντέλου.

Τα βασικά μεγέθη τα οποία υπεισέρχονται στους υπολογισμούς με την μέθοδο της καθόδου κλίσης είναι: Η συνάρτηση κόστους ή απώλειας (cost / loss function), που μετρά την απόκλιση μεταξύ των προβλεπόμενων και των πραγματικών (επιδιωκόμενων) τιμών. Ο στόχος είναι να ελαχιστοποιηθεί αυτή η συνάρτηση. Η κλίση (gradient) της συνάρτησης κόστους σε μια συγκεκριμένη θέση, που δείχνει την κατεύθυνση της μεγαλύτερης αύξησης της συνάρτησης. Και τέλος, ο ρυθμός μάθησης (Learning Rate), που είναι ο παράγοντας που καθορίζει το μέγεθος των βημάτων που κάνουμε προς τη κατεύθυνση της πτώσης κλίσης (Ruder, 2016).

Η διαδικασία εφαρμογής της μεθόδου της καθόδου κλίσης περιλαμβάνει τα εξής βήματα: Την απόδοση αρχικών τιμών, με τυχαίες αρχικές τιμές για τις παραμέτρους (βάρη και αποκλίσεις) του μοντέλου. Τον υπολογισμό της συνάρτησης κόστους ή απώλειας, για τις αρχικές - τρέχουσες παραμέτρους. Τον υπολογισμό της κλίσης της συνάρτησης κόστους στο τρέχον σημείο (παράμετροι). Αυτό επιτυγχάνεται υπολογίζοντας την πρώτη παράγωγο της συνάρτησης κόστους ως προς τις παραμέτρους. Την ενημέρωση κάθε παραμέτρου (βαρών και αποκλίσεων) με ελάττωση της τρέχουσας τιμής της κατά ένα κλάσμα της κλίσης (κάθοδος κλίσης – Gradient Descent). Το κλάσμα αυτό είναι ο ρυθμός εκμάθησης(learning rate).

$$w = w - \alpha \frac{\partial J(w,b)}{\partial w}$$

$$b = b - \alpha \frac{\partial J(w,b)}{\partial b}$$

Όπου ω είναι τα βάρη και b οι αποκλίσεις, α ο ρυθμός εκμάθησης και J(ω,b) η συνάρτηση απώλειας, οπότε

$\frac{\partial J(w,b)}{\partial w}$ είναι η κλίση (gradient) και $-a\frac{\partial J(w,b)}{\partial w}$ η κάθοδος (πτώση-μείωση) της κλίσης με ένα ρυθμό α.

Οι μερικές παράγωγοι της συνάρτησης απώλειας, ως προς τα βάρη και τις αποκλίσεις είναι :

$$\frac{\partial J(w,b)}{\partial w} = \frac{1}{m}\sum_{i=0}^{m-1}\left(f_{w,b}\left(x^{(i)}\right) - y^{(i)}\right)x^{(i)}$$

$$\frac{\partial J(w,b)}{\partial b} = \frac{1}{m}\sum_{i=0}^{m-1}\left(f_{w,b}\left(x^{(i)}\right) - y^{(i)}\right)$$

όπου $y^{(i)}$ η προβλεπόμενη τιμή και $f_{w,b}\left(x^{(i)}\right)$ η τιμή μετά την εφαρμογή των συναρτήσεων ενεργοποίησης (activation function) στα γραμμικά μοντέλα του δικτύου σε κάθε επίπεδο υπολογισμού. που υπολογίζεται σε κάθε επανάληψη

Επαναλαμβάνουμε συνεχώς τα προηγούμενα βήματα, με αρχικές τις τρέχουσες τιμές των παραμέτρων, μέχρις ότου, η συνάρτηση κόστους να πλησιάσει ένα ελάχιστο ή μέχρι να φτάσουμε σε έναν προκαθορισμένο αριθμό επαναλήψεων.

Ιδιαίτερα σημεία που πρέπει να προσεχθούν είναι:

Η επιλογή Ρυθμού Μάθησης (Learning Rate). Ένας πολύ μεγάλος ρυθμός μπορεί να οδηγήσει σε υπερπήδηση του ελάχιστου, ενώ ένας πολύ μικρός μπορεί να επιβραδύνει πολύ τη διαδικασία.

Τα Τοπικά Ελάχιστα (Local Minima): Η μέθοδος μπορεί να κολλήσει σε τοπικά ελάχιστα, γι' αυτό και οι κλίσεις με θόρυβο από το SGD μπορούν να βοηθήσουν στην αποφυγή τους.

Το Σημείο (Σαμάρι) Saddle: Μπορεί να υπάρξουν σημεία όπου η κλίση είναι μηδενική αλλά δεν είναι ελάχιστο (Duchi, 2011).

Η μέθοδος πτώσης κλίσης είναι ευρέως χρησιμοποιούμενη για την εκπαίδευση μοντέλων μηχανικής μάθησης, αλλά έχει τα δικά της πλεονεκτήματα και μειονεκτήματα.
Τα πλεονεκτήματά της είναι:

Η Αποτελεσματικότητα. Η μέθοδος πτώσης κλίσης είναι ικανή να βρει το ελάχιστο μιας συνάρτησης κόστους σε πολλές περιπτώσεις, ειδικά όταν η συνάρτηση είναι κυρτή. Αυτό την καθιστά ιδανική για προβλήματα βελτιστοποίησης.

Η Ευελιξία. Υπάρχουν διάφορες παραλλαγές της μεθόδου, όπως η Stochastic Gradient Descent (SGD) και η Mini-Batch Gradient Descent, οι οποίες επιτρέπουν την προσαρμογή της μεθόδου ανάλογα με τις ανάγκες του προβλήματος.

Η απλότητα. Ο αλγόριθμος είναι σχετικά απλός να υλοποιηθεί και να κατανοηθεί, γεγονός που διευκολύνει την εφαρμογή του σε διάφορες εφαρμογές μηχανικής μάθησης.

Η γενίκευση. Η GD μπορεί να χρησιμοποιηθεί για την εκπαίδευση πολλών τύπων μοντέλων, συμπεριλαμβανομένων των νευρωνικών δικτύων, κάνοντάς την πολύ χρήσιμη σε διάφορους τομείς της τεχνητής νοημοσύνης.

Τα μειονεκτήματα όμως υπάρχουν και είναι:

Τα Τοπικά Ελάχιστα. Σε μη κυρτές συναρτήσεις κόστους, η μέθοδος μπορεί να κολλήσει σε τοπικά ελάχιστα, αποτυγχάνοντας να βρει το καθολικό ελάχιστο.

Η επιλογή του ρυθμού μάθησης είναι κρίσιμη. Ένας πολύ μεγάλος ρυθμός μπορεί να οδηγήσει σε υπερπήδηση του ελάχιστου, ενώ ένας πολύ μικρός μπορεί να καθυστερήσει ιδιαίτερα τη διαδικασία εκπαίδευσης.

Τα Σημεία Μηδενικής Κλίσης (Saddle points): Η μέθοδος μπορεί να σταματήσει σε σημεία όπου η κλίση είναι μηδενική αλλά δεν είναι ελάχιστο, περιορίζοντας έτσι την πρόοδο του αλγορίθμου.

Η Υπολογιστική Πολυπλοκότητα: Ανάλογα με την παραλλαγή που χρησιμοποιείται (π.χ., Batch vs Stochastic), η υπολογιστική πολυπλοκότητα μπορεί να είναι υψηλή, ειδικά για μεγάλα σύνολα δεδομένων.

Η κατανόηση αυτών των πλεονεκτημάτων και μειονεκτημάτων είναι σημαντική για την αποτελεσματική εφαρμογή της μεθόδου πτώσης κλίσης σε διάφορα προβλήματα μηχανικής μάθησης.

Ωστόσο, η μέθοδος πτώσης κλίσης μπορεί να βελτιωθεί με διάφορες τεχνικές και στρατηγικές που στοχεύουν στην αύξηση της ταχύτητας και της ορθότητας της εκπαίδευσης των μοντέλων. Ακολουθούν μερικές από τις πιο αποτελεσματικές προσεγγίσεις:

Η επιλογή Προσαρμοσμένου Ρυθμού Μάθησης (Adaptive Learning Rate), με τη χρήση προσαρμοστικών αλγορίθμων μάθησης όπως οι Adam, RMSprop και Adagrad. Αυτοί οι αλγόριθμοι προσαρμόζουν τον ρυθμό μάθησης για κάθε παράμετρο, επιτρέποντας ταχύτερη σύγκλιση και αποφυγή υπερπήδησης του ελάχιστου. Η κανονικοποίηση (normalization) των χαρακτηριστικών (features) μπορεί να βοηθήσει στη βελτίωση της ταχύτητας σύγκλισης. Η τυποποίηση (standardization) των δεδομένων μπορεί να μειώσει την επίδραση των εξαιρετικών τιμών (outliers) (Izmailov, 2018).

Η παραλλαγή σε Mini-Batch Gradient Descent, όπου, αντί να χρησιμοποιούνται όλα τα δεδομένα ή μόνο ένα δείγμα, κάνει χρήση μικρών δεσμίδων (mini-batches) και μπορεί να συνδυάσει τα πλεονεκτήματα του Batch και Stochastic Gradient Descent, προσφέροντας πιο σταθερές ενημερώσεις.

Η προσθήκη ενός όρου ‘Ορμής’ (Momentum) στις ενημερώσεις των παραμέτρων μπορεί να βοηθήσει στην επιτάχυνση της διαδικασίας εκπαίδευσης, επιτρέποντας στο μοντέλο να "συγκρατεί" την κατεύθυνση των προηγούμενων ενημερώσεων.

Το momentum αναφέρεται σε μια στρατηγική που προσαρμόζει τις ενημερώσεις των βαρών ενός νευρωνικού δικτύου με βάση την κατεύθυνση και το μέγεθος των προηγούμενων ενημερώσεων. Αντί να βασίζεται μόνο στην τρέχουσα κλίση της συνάρτησης κόστους, το momentum προσθέτει ένα ποσοστό της προηγούμενης ενημέρωσης, δημιουργώντας έτσι μια "ορμή" που βοηθά στην κατεύθυνση της βελτίωσης.

Κατά την εκπαίδευση ενός νευρωνικού δικτύου, τα βάρη ενημερώνονται με βάση την κλίση της συνάρτησης κόστους. Με το momentum, η ενημέρωση γίνεται ως εξής:

$$v_t = \beta v_{t-1} + (1 - \beta)\nabla J(w)$$

$$w = w - \alpha v_t$$

Όπου v_t είναι η τρέχουσα ορμή, β είναι ο παράγοντας momentum, $\nabla J(w)$ είναι η κλίση και w είναι τα βάρη.

Η παρακολούθηση της απόδοσης του μοντέλου σε ένα σύνολο επικύρωσης (validation set) και διακοπή της εκπαίδευσης - Early Stopping - όταν η απόδοση αρχίζει να επιδεινώνεται, αποφεύγοντας έτσι την υπερπροσαρμογή (overfitting).

Η Χρήση ειδικών τεχνικών κανονικοποίησης - Regularization Techniques - όπως L1 ή L2 regularization για την αποφυγή υπερπροσαρμογής και τη βελτίωση της γενίκευσης του μοντέλου (Loshchilov, 2017). Και τέλος, η δοκιμή διαφορετικών μοντέλων και ο πειραματισμός με διαφορετικές αρχιτεκτονικές νευρωνικών δικτύων μπορεί να οδηγήσει σε καλύτερη απόδοση, ανάλογα με το πρόβλημα που αντιμετωπίζεται.

Η εφαρμογή αυτών των στρατηγικών μπορεί να βελτιώσει σημαντικά την απόδοση της μεθόδου πτώσης κλίσης, κάνοντάς την πιο αποτελεσματική στην εκπαίδευση μοντέλων μηχανικής μάθησης.

2.4.2. Στοχαστική Κάθοδος Κλίσης – Stochastic Gradient Descent

Η **Στοχαστική Κάθοδος Κλίσης** (Stochastic Gradient Descent - SGD) είναι μια μέθοδος βελτιστοποίησης που χρησιμοποιείται κυρίως στην εκπαίδευση νευρωνικών δικτύων και άλλων αλγορίθμων μηχανικής μάθησης. Αντί να υπολογίζει την κλίση της συνάρτησης κόστους με βάση ολόκληρο το σύνολο δεδομένων, η SGD χρησιμοποιεί τυχαία επιλεγμένα δείγματα (ή ακόμα και ένα μόνο δείγμα) για να ενημερώσει τις παραμέτρους του μοντέλου. Αυτή η προσέγγιση μειώνει τον υπολογιστικό φόρτο και επιτρέπει ταχύτερες ενημερώσεις των παραμέτρων (Smith, 2018).

Η διαδικασία εφαρμογής της SGD περιλαμβάνει : πρώτα την επιλογή δείγματος, όπου επιλέγεται τυχαία ένα δείγμα από το σύνολο δεδομένων, τον υπολογισμό της κλίσης κατά τον οποίο υπολογίζεται η κλίση της συνάρτησης κόστους για το συγκεκριμένο δείγμα, την ενημέρωση των παραμέτρων στην οποία οι παράμετροι του μοντέλου ενημερώνονται με βάση την κλίση που υπολογίστηκε:

$$w = w - \eta \nabla Q_i(w)$$

όπου w είναι οι παράμετροι, η είναι ο ρυθμός μάθησης και $\nabla Q_i(w)$ είναι η κλίση για το δείγμα i , και τη διαδικασία επανάληψης που επαναλαμβάνεται για όλα τα δείγματα στο σύνολο δεδομένων, πολλές φορές, μέχρι να επιτευχθεί η σύγκλιση.

Τα πλεονεκτήματα της SGD είναι:
Η ταχύτητα, καθώς η ενημέρωση των παραμέτρων γίνεται γρηγορότερα, επειδή δεν απαιτείται ο υπολογισμός της κλίσης για ολόκληρο το σύνολο δεδομένων. Η δυνατότητα Online Learning, επειδή μπορεί να χρησιμοποιηθεί σε σενάρια όπου τα δεδομένα έρχονται διαρκώς, επιτρέποντας την εκπαίδευση κατά τη διάρκεια της λειτουργίας. Η αποτελεσματικότητα σε μεγάλες δεδομένες συλλογές δεδομένων, που την καθιστά πιο αποδοτική σε μεγάλες βάσεις δεδομένων, καθώς μπορεί να ενημερώνει τις παραμέτρους με λιγότερους υπολογισμούς.

Ωστόσο, δεν λείπουν και τα μειονεκτήματα της SGD που είναι: Η SGD μπορεί να συγκλίνει σε τοπικά ελάχιστα αντί για το καθολικό ελάχιστο, ιδιαίτερα αν ο ρυθμός μάθησης δεν είναι κατάλληλα ρυθμισμένος. Οι ενημερώσεις μπορεί να είναι εμφανίσουν περισσότερο θόρυβο, λόγω της τυχαίας επιλογής των δειγμάτων, γεγονός που μπορεί να κάνει τη διαδικασία εκπαίδευσης

λιγότερο σταθερή. Η επιλογή του κατάλληλου ρυθμού μάθησης είναι κρίσιμη και μπορεί να απαιτεί πειραματισμό για να βρεθεί η βέλτιστη τιμή.

Η Στοχαστική Κάθοδος Κλίσης αποτελεί έναν από τους πιο δημοφιλείς αλγορίθμους βελτιστοποίησης στη μηχανική μάθηση και έχει εφαρμογές σε πολλούς τομείς, από την αναγνώριση εικόνας μέχρι την επεξεργασία φυσικής γλώσσας.

Ο αλγόριθμος Robbins-Monro είναι μια μέθοδος στοχαστικής προσέγγισης που χρησιμοποιείται για την εκτίμηση παραμέτρων και την επίλυση προβλημάτων βελτιστοποίησης, όταν οι παρατηρήσεις είναι τυχαίες και η συνάρτηση κόστους δεν είναι άμεσα προσιτή. Αναπτύχθηκε από τους Herbert Robbins και Sutton Monro το 1951 και βασίζεται σε επαναλαμβανόμενες ενημερώσεις που προσαρμόζονται με βάση τυχαία δείγματα.

Η βασική ιδέα του Robbins-Monro είναι να χρησιμοποιεί μια ακολουθία ενημερώσεων της μορφής:

$$\partial_{n+1} = \partial_n - \gamma_n X_n$$

όπου:
∂_n είναι η τρέχουσα εκτίμηση,
γ_n είναι ο ρυθμός μάθησης (step size),
X_n είναι μια τυχαία μεταβλητή που σχετίζεται με την παράμετρο που επιθυμούμε να εκτιμήσουμε.

Η επιλογή του ρυθμού μάθησης είναι κρίσιμη, καθώς πρέπει να μειώνεται με την πάροδο του χρόνου για να διασφαλιστεί η σύγκλιση στην επιθυμητή τιμή.

Η Στοχαστική Κάθοδος Κλίσης (SGD) μπορεί να θεωρηθεί ως μια εφαρμογή του αλγορίθμου Robbins-Monro σε προβλήματα βελτιστοποίησης. Όπως και ο Robbins-Monro, η SGD χρησιμοποιεί τυχαία δείγματα για να υπολογίσει την κλίση της συνάρτησης κόστους και να ενημερώσει τις παραμέτρους του μοντέλου.

Και οι δύο παραπάνω μέθοδοι βασίζονται στην ιδέα ότι μπορούμε να κάνουμε ενημερώσεις χρησιμοποιώντας τυχαία δείγματα αντί για ολόκληρα σύνολα δεδομένων, γεγονός που καθιστά τις διαδικασίες πιο αποδοτικές υπολογιστικά. Στην SGD, η ενημέρωση γίνεται με βάση την κλίση της συνάρτησης κόστους για ένα μόνο δείγμα, ακολουθώντας τη μορφή:

$$w_{n+1} = w_n - \eta \nabla Q_n(w)$$

όπου $\nabla Q_n(w)$, είναι η κλίση για το δείγμα n. Αυτή η διαδικασία είναι παρόμοια με την ενημέρωση του Robbins-Monro, όπου η κλίση υπολογίζεται από τυχαίες παρατηρήσεις.

Η σύγκλιση της SGD έχει αναλυθεί χρησιμοποιώντας θεωρίες που σχετίζονται με τον αλγόριθμο Robbins-Monro. Όταν οι ρυθμοί μάθησης μειώνονται κατάλληλα, η SGD μπορεί να συγκλίνει σχεδόν σίγουρα σε ένα τοπικό ή παγκόσμιο ελάχιστο, ανάλογα με τη φύση της συνάρτησης κόστους. Συνοψίζοντας, ο αλγόριθμος Robbins-Monro παρέχει τη θεωρητική βάση για τη στοχαστική προσέγγιση στη βελτιστοποίηση, ενώ η SGD αποτελεί μια πρακτική εφαρμογή αυτής της θεωρίας σε προβλήματα μηχανικής μάθησης και νευρωνικών δικτύων.

2.4.3. Κάθοδος Κλίσης σε Δεσμίδες - Batch Gradient Descent

Το Batch Gradient Descent είναι ο πιο απλός τύπος. Υπολογίζει το σφάλμα για κάθε παράδειγμα στο σύνολο δεδομένων εκπαίδευσης, ωστόσο, ενημερώνει το μοντέλο μόνο αφού αξιολογηθούν όλα τα παραδείγματα εκπαίδευσης.

Στο Batch Gradient Descent, οι παράμετροι του μοντέλου ενημερώνονται χρησιμοποιώντας τον μέσο όρο των κλίσεων που υπολογίζονται από ολόκληρο το σύνολο εκπαίδευσης. Αυτό σημαίνει ότι για κάθε επανάληψη, εξετάζονται όλα τα παραδείγματα εκπαίδευσης για να υπολογιστεί η κλίση της συνάρτησης απώλειας (Ioffe, 2015).

Τα ιδιαίτερα χαρακτηριστικά της μεθοδολογίας αυτής είναι: αρχικά ότι συγκλίνει ομαλά προς το ελάχιστο της συνάρτησης απώλειας, κάνοντάς το κατάλληλο για κυρτές ή σχετικά ομαλές

επιφάνειες σφάλματος. Όμως, επειδή απαιτεί τον υπολογισμό κλίσεων για όλα τα παραδείγματα εκπαίδευσης σε κάθε βήμα, μπορεί να είναι υπολογιστικά δαπανηρή και αργή, ειδικά με μεγάλα σύνολα δεδομένων. Οι ενημερώσεις των παραμέτρων, παρουσιάζουν λιγότερο θόρυβο, καθώς βασίζονται σε ολόκληρο το σύνολο δεδομένων, οδηγώντας σε σταθερές συγκλίνουσες τιμές.

Κάθοδος Κλίσης σε Μικρές Δεσμίδες – Mini Batch Gradient Descent

Η Mini-Batch Gradient Descent είναι μια παραλλαγή του αλγορίθμου Gradient Descent που συνδυάζει τα πλεονεκτήματα της Batch Gradient Descent και της Stochastic Gradient Descent (SGD). Αυτή η μέθοδος χρησιμοποιεί υποσύνολα - δεσμίδες (mini-batches) του συνόλου δεδομένων για να υπολογίσει την κλίση της συνάρτησης κόστους και να ενημερώσει τις παραμέτρους του μοντέλου.

Η λειτουργία της ακολουθεί τα παρακάτω βήματα:

Αρχικά, το σύνολο δεδομένων χωρίζεται σε μικρότερα υποσύνολα, τα οποία ονομάζονται mini-batches. Συνήθως, το μέγεθος του mini-batch κυμαίνεται από 32 έως 256 δείγματα, αν και μπορεί να διαφέρει ανάλογα με το πρόβλημα.

Αυτή η προσέγγιση επιτρέπει πιο συχνές ενημερώσεις των παραμέτρων σε σύγκριση με την Batch Gradient Descent, ενώ διατηρεί τη σταθερότητα που προσφέρει η χρήση περισσότερων από ένα παραδείγματα ανά ενημέρωση, αποφεύγοντας έτσι την θορυβώδη συμπεριφορά του Stochastic Gradient Descent.

Η Mini-Batch Gradient Descent είναι μια παραλλαγή της μεθόδου Gradient Descent που χρησιμοποιεί μαθηματικούς τύπους για την ενημέρωση των παραμέτρων του μοντέλου. Ακολουθούν οι βασικοί τύποι που χρησιμοποιούνται:

Δημιουργία Mini-Batches: Το σύνολο εκπαίδευσης χωρίζεται σε μικρότερα υποσύνολα (mini-batches).
Υπολογισμός της Συνάρτησης Απώλειας για κάθε mini batch:

Για ένα σύνολο δεδομένων $\{(x_i, y_i)\}_{i=1}^{P}$ με P δείγματα, η συνάρτηση απώλειας J(θ) υπολογίζεται ως εξής:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} L(h_\theta(x_i), y_i)$$

όπου m είναι το μέγεθος της mini-batch και L είναι η συνάρτηση απώλειας (π.χ. μέση τετραγωνική απώλεια).

Υπολογισμός Gradient της συνάρτησης απώλειας L, για κάθε Mini-Batch: Για κάθε mini-batch, υπολογίζεται η κλίση της συνάρτησης απώλειας. Ο υπολογισμός της κλίσης - gradient της συνάρτησης απώλειας ως προς τις παραμέτρους θ γίνεται με τον εξής τύπο:

$$\nabla J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \nabla L(h_\theta(x_i), y_i)$$

με ενημέρωση των παραμέτρων όλων των δεσμίδων Οι παράμετροι ενημερώνονται με βάση τον υπολογισμό του gradient και το learning rate α:

$$\theta = \theta - a\nabla J(\theta)$$

Τα Πλεονεκτήματα της Mini-Batch Gradient Descent είναι σημαντικά.

Σε σύγκριση με την Batch Gradient Descent, η Mini-Batch Gradient Descent επιτρέπει ταχύτερες ενημερώσεις παραμέτρων, καθώς δεν απαιτεί τον υπολογισμό της κλίσης για ολόκληρο το σύνολο δεδομένων σε κάθε επανάληψη.

Σε αντίθεση με την Stochastic Gradient Descent, που μπορεί να έχει μεγάλες διακυμάνσεις λόγω των τυχαίων δειγμάτων, η Mini-Batch παρέχει πιο σταθερές εκτιμήσεις της κλίσης, μειώνοντας τον θόρυβο στις ενημερώσεις.

Η χρήση mini-batches επιτρέπει την εκμετάλλευση των παραλληλίας των σύγχρονων υπολογιστικών συστημάτων και GPU, βελτιώνοντας την απόδοση.

Ωστόσο, η επιλογή κατάλληλου μεγέθους Mini-Batch μπορεί να είναι δύσκολη και να επηρεάζει τη σύγκλιση και την απόδοση του αλγορίθμου. Πολύ μικρά ή πολύ μεγάλα mini-batches μπορεί να οδηγήσουν σε κακή εκπαίδευση.

Επίσης όπως και με άλλες μεθόδους gradient descent, υπάρχει η πιθανότητα να συγκλίνει σε τοπικά ελάχιστα, ειδικά σε μη-κυρτές συναρτήσεις κόστους.

Τέλος, ανάλογα με το μέγεθος του mini-batch και το μέγεθος των δεδομένων, μπορεί να απαιτείται περισσότερη μνήμη για την αποθήκευση των δειγμάτων κατά τη διάρκεια της εκπαίδευσης.
Η Mini-Batch Gradient Descent αποτελεί μια δημοφιλή επιλογή για την εκπαίδευση νευρωνικών δικτύων, καθώς συνδυάζει τα πλεονεκτήματα των άλλων μεθόδων βελτιστοποίησης και προσφέρει μια ισορροπία μεταξύ ταχύτητας και σταθερότητας.

Η επιλογή του μέγιστου αριθμού δειγμάτων για την Mini-Batch Gradient Descent είναι κρίσιμη για την απόδοση και τη σύγκλιση του αλγορίθμου. Ορισμένοι παράγοντες που επηρεάζουν αυτήν την επιλογή περιλαμβάνουν:

Το μέγεθος του συνολικού dataset μπορεί να καθορίσει το εύρος των mini-batches. Σε μικρότερα σύνολα δεδομένων, μπορεί να είναι λογικό να χρησιμοποιούνται μεγαλύτερα mini-batches, ενώ σε μεγαλύτερα σύνολα, μικρότερα mini-batches μπορεί να παρέχουν καλύτερη σταθερότητα και ταχύτητα.

Η διαθέσιμη μνήμη και η υπολογιστική ισχύς επηρεάζουν άμεσα το μέγεθος του mini-batch. Μεγαλύτερα mini-batches απαιτούν περισσότερη μνήμη και πόρους, ενώ μικρότερα mini-batches είναι πιο ελαφριά αλλά μπορεί να επιβραδύνουν τη διαδικασία εκπαίδευσης.

Μεγαλύτερα mini-batches παρέχουν πιο σταθερές εκτιμήσεις της κλίσης, μειώνοντας τον θόρυβο στις ενημερώσεις παραμέτρων. Ωστόσο, πολύ μεγάλα mini-batches μπορεί να οδηγήσουν σε

λιγότερες ενημερώσεις και σε μεγαλύτερη πιθανότητα να κολλήσει ο αλγόριθμος σε τοπικά ελάχιστα.

Ο ρυθμός μάθησης (learning rate) μπορεί να επηρεάσει την επιλογή του μεγέθους του mini-batch. Αν ο ρυθμός μάθησης είναι υψηλός, μικρότερα mini-batches μπορεί να είναι προτιμότερα για να αποφευχθούν μεγάλες διακυμάνσεις στην εκπαίδευση.

Συχνά η επιλογή του μεγέθους του mini-batch γίνεται μέσω πειραματισμού. Δοκιμάζοντας διαφορετικά μεγέθη και παρακολουθώντας τις επιδόσεις του μοντέλου, οι ερευνητές μπορούν να βρουν την καλύτερη ρύθμιση για συγκεκριμένα προβλήματα.

Συνήθως, τα μεγέθη mini-batch κυμαίνονται από 32 έως 256 δείγματα, αν και μπορεί να χρησιμοποιηθούν και άλλες τιμές ανάλογα με τις συνθήκες. Μια καλή αρχική προσέγγιση είναι να ξεκινήσετε με ένα μέγεθος γύρω από 64 ή 128 και στη συνέχεια να προσαρμόσετε ανάλογα με τα αποτελέσματα.

Η σωστή επιλογή του μεγέθους των mini-batches μπορεί να βελτιώσει σημαντικά την απόδοση της εκπαίδευσης ενός νευρωνικού δικτύου, επιτυγχάνοντας καλύτερη σύγκλιση και ταχύτερη εκπαίδευση.

2.4.4. Προσαρμοσμένη Κάθοδος Κλίσης – Adaptive Gradient Descent

Αναφέρθηκε προηγουμένως, ότι η μέθοδος Batch Gradient Descent είναι ο πιο απλός τύπος παραλλαγής της καθόδου κλίσης για την βελτιστοποίηση νευρωνικών δικτύων . Υπολογίζει το σφάλμα για κάθε δεσμίδα στο σύνολο δεδομένων εκπαίδευσης, ωστόσο, ενημερώνει το μοντέλο μόνο αφού αξιολογηθούν όλες οι δεσμίδες εκπαίδευσης.

Επίσης ότι η μέθοδος Stochastic Gradient Descent υπολογίζει το σφάλμα και ενημερώνει το μοντέλο για κάθε παράδειγμα στο σύνολο δεδομένων εκπαίδευσης.

Και η Mini Batch Gradient Descent αντί να εξετάζει όλα τα παραδείγματα, συνοψίζει τον μικρότερο αριθμό παραδειγμάτων με βάση το μέγεθος της παρτίδας και εκτελεί μια ενημέρωση για καθεμία από αυτές τις παρτίδες. Το Stochastic Gradient Descent είναι μια συνηθισμένη μέθοδος βελτιστοποίησης. Είναι εννοιολογικά απλό και συχνά μπορεί να εφαρμοστεί αποτελεσματικά. Ωστόσο, διαθέτει μια παράμετρο (το μέγεθος βήματος) που πρέπει να συντονιστεί χειροκίνητα. Έχουν προταθεί διάφορες επιλογές για την αυτοματοποίηση αυτού του συντονισμού.

Ένα από τα επιτυχημένα σχέδια είναι η μέθοδος AdaGrad. Ενώ οι τυπικές μέθοδοι στοχαστικής καθόδου της κλίσης, ακολουθούν κυρίως ένα προκαθορισμένο διαδικαστικό σχήμα που αγνοεί τα χαρακτηριστικά των δεδομένων που παρατηρούνται, αντίθετα, οι αλγόριθμοι της AdaGrad ενσωματώνουν δυναμικά τη γνώση της γεωμετρίας των δεδομένων που παρατηρήθηκαν σε προηγούμενες επαναλήψεις για να εκτελέσουν πιο ενημερωτική μάθηση με βάση την κλίση. Η AdaGrad έχει κυκλοφορήσει σε δύο εκδόσεις. Διαγώνιο AdaGrad (αυτή η έκδοση είναι αυτή που χρησιμοποιείται στην πράξη), που το κύριο χαρακτηριστικό της είναι να διατηρεί και να προσαρμόζει έναν ρυθμό εκμάθησης ανά διάσταση και η δεύτερη έκδοση που είναι γνωστή ως Full AdaGrad που διατηρεί έναν ρυθμό εκμάθησης ανά κατεύθυνση (π.χ. πλήρης πίνακας PSD). Ο Αλγόριθμος Προσαρμοστικής (Καθόδου) Κλίσης (Adagrad) είναι ένας αλγόριθμος για βελτιστοποίηση που βασίζεται στην κλίση. Ο ρυθμός μάθησης προσαρμόζεται ανάλογα με τις παραμέτρους ενσωματώνοντας γνώσεις από προηγούμενες παρατηρήσεις. Εκτελεί μεγαλύτερες ενημερώσεις (π.χ. υψηλά ποσοστά εκμάθησης) για εκείνες τις παραμέτρους που σχετίζονται με σπάνιες λειτουργίες και μικρότερες ενημερώσεις (δηλαδή χαμηλά ποσοστά εκμάθησης) για παραμέτρους με συχνές λειτουργίες.

Ως αποτέλεσμα, είναι κατάλληλος όταν αντιμετωπίζονται αραιά δεδομένα (NLP ή αναγνώριση εικόνας) Κάθε παράμετρος έχει το δικό της ρυθμό εκμάθησης που βελτιώνει την απόδοση σε προβλήματα με αραιές διαβαθμίσεις.

Τα πλεονεκτήματα της χρήσης της AdaGrad είναι να εξαλείφει την ανάγκη χειροκίνητου συντονισμού του ρυθμού εκμάθησης, η σύγκλιση να είναι ταχύτερη και πιο αξιόπιστη – από την

απλή SGD όταν η κλιμάκωση των βαρών είναι άνιση και τελικά να μην εμφανίζει ιδιαίτερη ευαισθησία στο μέγεθος του βασικού βήματος.

Ο πίνακας PSD (Positive Semi-Definite) στον αλγόριθμο Adagrad είναι ένας σημαντικός παράγοντας που καθιστά δυνατή την προσαρμογή του ρυθμού εκμάθησης (learning rate) για κάθε παράμετρο του μοντέλου ξεχωριστά. Ας εξετάσουμε αναλυτικά τι είναι αυτός ο πίνακας και πώς χρησιμοποιείται στον Adagrad.

Ο πίνακας Positive Semi-Definite (PSD) είναι ένας τετραγωνικός πίνακας G που πληροί την ιδιότητα ότι για κάθε μη μηδενικό διάνυσμα x, ισχύει:

$$x^T G x \geq 0$$

Αυτό σημαίνει ότι όλες οι ιδιοτιμές του πίνακα G είναι μη αρνητικές. Στον αλγόριθμο Adagrad, αυτός ο πίνακας χρησιμοποιείται για να συσσωρεύει τις τετραγωνικές τιμές των gradients και να τις προσαρμόζει ανάλογα τον ρυθμό εκμάθησης για κάθε παράμετρο.

Στον αλγόριθμο Adagrad, ο πίνακας G_t στο χρόνο t είναι ένας διαγώνιος πίνακας που συσσωρεύει τα τετράγωνα των gradients για κάθε παράμετρο μέχρι το βήμα t. Συγκεκριμένα:

$$G_t = diag\left(\sum_{s=1}^{t} g_{s,1}^2, \sum_{s=1}^{t} g_{s,2}^2, \ldots, \sum_{s=1}^{t} g_{s,d}^2\right)$$

Όπου:
G_t είναι ο διαγώνιος πίνακας στο βήμα t.
$g_{s,i}$ είναι το gradient της παραμέτρου i στο βήμα s.
d είναι το συνολικό πλήθος των παραμέτρων του μοντέλου.
Αυτός ο πίνακας είναι Positive Semi-Definite επειδή όλα τα στοιχεία στη διαγώνιο του είναι μη αρνητικά (είναι άθροισμα τετραγώνων).

Ο πίνακας G_t στον αλγόριθμο Adagrad, χρησιμοποιείται για να προσαρμόσει τον ρυθμό εκμάθησης για κάθε παράμετρο. Η ενημέρωση των παραμέτρων γίνεται ως εξής:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{G_t + \epsilon}} \odot g_t$$

Όπου:

η είναι ο αρχικός ρυθμός εκμάθησης.

ϵ είναι ένας μικρός αριθμός για να αποφεύγεται η διαίρεση με το μηδέν.

$\odot$ δηλώνει στοιχειώδη (element-wise) πράξη.

θ_t είναι οι παράμετροι στο βήμα t.

g_t είναι ο gradient στο βήμα t.

Λόγω του ότι ο G_t είναι διαγώνιος και PSD, ο ρυθμός εκμάθησης για κάθε παράμετρο προσαρμόζεται ανάλογα με το ιστορικό των gradients αυτής της παραμέτρου. Παράμετροι που έχουν υψηλά gradients θα έχουν μεγαλύτερη συσσώρευση στον G_t, οδηγώντας σε μικρότερο βήμα ενημέρωσης, ενώ παράμετροι με χαμηλότερα gradients θα ενημερώνονται περισσότερο.

Η χρήση ενός PSD πίνακα εξασφαλίζει ότι οι προσαρμογές του ρυθμού εκμάθησης είναι πάντα θετικές και σταθερές. Αυτό αποτρέπει τη χρήση αρνητικών ρυθμών εκμάθησης και διασφαλίζει ότι οι ενημερώσεις των παραμέτρων είναι κατευθυνόμενες προς τη μείωση της συνάρτησης κόστους.

Παράδειγμα Υπολογισμού του Gt

Ας υποθέσουμε ότι έχουμε δύο παραμέτρους (θ_1, και θ_2) και παρατηρούμε τα ακόλουθα gradients σε κάθε βήμα:

Table 1

Βήμα (t)	Gradient g_t	G_t
1	$[g_{1,1}, g_{1,2}] = [2, 3]$	$\text{diag}(2^2, 3^2) = \text{diag}(4, 9)$
2	$[g_{2,1}, g_{2,2}] = [1, 4]$	$\text{diag}(4 + 1^2, 9 + 4^2) = \text{diag}(5, 25)$
3	$[g_{3,1}, g_{3,2}] = [3, 2]$	$\text{diag}(5 + 3^2, 25 + 2^2) = \text{diag}(14, 29)$

Σε κάθε βήμα, ο πίνακας Grad_g_t συσσωρεύει τα τετράγωνα των κλίσεων -gradients- για κάθε παράμετρο, επιτρέποντας στον Adagrad να προσαρμόζει τον ρυθμό εκμάθησης αντίστοιχα.

Συμπερασματικά, ο πίνακας PSD στον αλγόριθμο Adagrad παίζει καθοριστικό ρόλο στην προσαρμογή των ρυθμών εκμάθησης για κάθε παράμετρο, εξασφαλίζοντας ότι οι ενημερώσεις είναι σταθερές και προσανατολισμένες προς τη βελτιστοποίηση της συνάρτησης κόστους. Επειδή είναι θετικά ημι-ορισμένος ο πίνακας Grad g_t , εγγυάται ότι οι ρυθμοί εκμάθησης παραμένουν θετικοί και ότι οι ενημερώσεις των παραμέτρων είναι σωστά κλιμακωμένες.

2.4.5 Διάδοση Μέσου Τετραγωνικού Σφάλματος Ροπών – Root Mean Square Propagation – RMSProp

Ο RMSProp είναι ένας αλγόριθμος βελτιστοποίησης που χρησιμοποιείται για την προσαρμογή του ρυθμού εκμάθησης κατά τη διάρκεια της εκπαίδευσης ενός μοντέλου. Προτείνεται από τον Geoffrey Hinton και αποτελεί βελτίωση του AdaGrad. Ο RMSProp επιλύει το πρόβλημα της συνεχούς μείωσης του ρυθμού εκμάθησης που συμβαίνει στο AdaGrad, χρησιμοποιώντας έναν εκθετικά σταθμισμένο κινητό μέσο όρο των τετραγωνικών κλίσεων αντί για το αθροιστικό άθροισμα των κλίσεων.

Η βασική ιδέα του RMSProp είναι να διατηρεί έναν κινητό μέσο όρο των τετραγωνικών κλίσεων και να προσαρμόζει τον ρυθμό εκμάθησης ανάλογα με την κλίμακα των κλίσεων σε κάθε διάσταση. Αυτό εξασφαλίζει ότι οι μεγάλες κλίσεις δεν οδηγούν σε υπερβολικά μεγάλες ενημερώσεις των παραμέτρων, ενώ οι μικρές κλίσεις εξακολουθούν να ενημερώνονται αρκετά ώστε να μη σταματήσει η εκπαίδευση.

Ο αλγόριθμος ενημερώνει τις παραμέτρους χρησιμοποιώντας έναν εκθετικά σταθμισμένο κινητό μέσο όρο των τετραγωνικών κλίσεων και προσαρμόζει τις ενημερώσεις των παραμέτρων με βάση αυτόν τον μέσο όρο. Αυτό εξασφαλίζει ότι οι ενημερώσεις των παραμέτρων δεν σταματούν εντελώς, ακόμη και όταν το σύστημα μαθαίνει για μεγάλο χρονικό διάστημα.

Η διαδικασία εφαρμογής του αλγόριθμου RMSprop, είναι:
Πρώτα, ορίζονται ο ρυθμός εκμάθησης (Learning rate) η, ο εκθετικός συντελεστής απόσβεσης β (συνήθως 0.9 που ελέγχει πόσο επηρεάζουν οι παλαιότερες κλίσεις την τρέχουσα ενημέρωση), και μια μικρή σταθερά για αριθμητική σταθερότητα ϵ.

Αρχικά και για κάθε παράμετρο θ, υπολογίζουμε τον εκθετικό κινούμενο μέσο των τετραγωνικών κλίσεων:

$$G_t = \beta G_{t-1} + (1-\beta)\nabla_\theta L(\theta_t)^2$$

όπου: G_t είναι η εκθετική κινούμενη μέση τιμή των τετραγωνικών κλίσεων τη στιγμή *t, και* g_t είναι η κλίση της συνάρτησης απώλειας ως προς την παράμετρο θ τη στιγμή *t.*

Στη συνέχεια οι παράμετροι ενημερώνονται ως εξής:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{G_t + \epsilon}}\nabla_\theta L(\theta_t)$$

όπου: θ είναι η παράμετρος που ενημερώνεται, η είναι ο ρυθμός εκμάθησης(learning rate), G_t είναι ο εκθετικός κινούμενος μέσος των τετραγωνικών κλίσεων και ϵ είναι μια μικρή σταθερά για να αποφεύγεται η διαίρεση με το μηδέν

Αυτή η προσέγγιση επιτρέπει στη μέθοδο RMSprop να προσαρμόζει τον ρυθμό εκμάθησης δυναμικά, βελτιώνοντας τη διαδικασία εκπαίδευσης και επιταχύνοντας τη σύγκλιση.

Σημαντικά είναι τα πλεονεκτήματα του RMSProp:

Σε αντίθεση με το AdaGrad, που συνεχώς μειώνει τον ρυθμό εκμάθησης με την πάροδο του χρόνου, ο RMSProp διατηρεί έναν πιο σταθερό ρυθμό εκμάθησης χρησιμοποιώντας έναν κινητό μέσο όρο των τετραγωνικών κλίσεων.

Ο RMSProp είναι εξαιρετικά ανθεκτικός σε δεδομένα με θόρυβο ή σε δεδομένα που μεταβάλλονται γρήγορα, όπως συμβαίνει σε περιπτώσεις ενισχυτικής μάθησης ή όταν χρησιμοποιούμε μικρές δεσμίδες (mini-batches).

Ο αλγόριθμος προσαρμόζει αυτόματα τον ρυθμό εκμάθησης για κάθε παράμετρο, επιτρέποντας γρήγορες ενημερώσεις στις παραμέτρους που χρειάζονται περισσότερο, ενώ παράλληλα αποφεύγει πολύ μεγάλες ενημερώσεις στις παραμέτρους που έχουν μεγάλες κλίσεις.

Ο RMSProp είναι ιδιαίτερα χρήσιμος σε προβλήματα όπου οι κλίσεις είναι ασταθείς ή με θόρυβο, όπως σε προβλήματα ενισχυτικής μάθησης ή όταν εκπαιδεύουμε με μικρές δεσμίδες δεδομένων.

Ο RMSProp είναι ιδανικός για τη βελτιστοποίηση μοντέλων που εκπαιδεύονται με μικρές δεσμίδες δεδομένων, καθώς είναι σχεδιασμένος να διαχειρίζεται αποτελεσματικά τις ασταθείς κλίσεις που προκύπτουν από τα δεδομένα αυτά.

Σε ασταθείς ή μεταβαλλόμενες κλίσεις, ο RMSProp, είναι ιδιαίτερα κατάλληλος για προβλήματα όπου οι κλίσεις μπορεί να αλλάζουν γρήγορα ή να είναι παρουσιάζουν θόρυβο, όπως κατά την εκπαίδευση σε περιβάλλοντα ενισχυτικής μάθησης ή σε δυναμικά συστήματα.

Τέλος, ο RMSProp λειτουργεί καλά σε μεγάλα προβλήματα βελτιστοποίησης, όπως τα βαθιά νευρωνικά δίκτυα, όπου οι διαφορετικές παράμετροι μπορεί να απαιτούν διαφορετικούς ρυθμούς εκμάθησης για να επιτύχουν καλή σύγκλιση.

2.4.6. Adam

Adam (Adaptive Moment Estimation) είναι ένας δημοφιλής αλγόριθμος βελτιστοποίησης που συνδυάζει τις ιδέες του AdaGrad και του RMSProp. Υπολογίζει έναν προσαρμοσμένο ρυθμό εκμάθησης για κάθε παράμετρο χρησιμοποιώντας σταθμισμένους εκθετικούς κινητούς μέσους όρους των πρώτων και δεύτερων ροπών των κλίσεων. Είναι ευρέως χρησιμοποιούμενος σε προβλήματα βαθιάς μάθησης λόγω της ταχύτητάς του και της δυνατότητάς του να προσαρμόζεται σε αραιά δεδομένα ή κλίσεις με θόρυβο (Reddi, 2019).

Ο αλγόριθμος Adam υπολογίζει προσαρμοσμένους ρυθμούς εκμάθησης για κάθε παράμετρο, χρησιμοποιώντας την πρώτη και τη δεύτερη ροπή των κλίσεων.

Τα βήματα των υπολογισμών που υπάρχουν στον αλγόριθμο Adam, όπως περιγράφονται στο λογισμικό μαθηματικών (και άλλων) υπολογισμών MATLAB.

Δίνονται αρχικές τιμές στις παραμέτρους του μοντέλου (θ), το ρυθμό εκμάθησης (α) και τις υπερπαραμέτρους (β1, β2 και ε).

Υπολογίζονται οι κλίσεις (g) της συνάρτησης απώλειας (L) σε σχέση με τις παραμέτρους του μοντέλου:

$$g^{(t)} = \nabla \mathcal{L}\left(\theta^{(t-1)}\right)$$

Ενημερώνονται οι εκτιμήσεις της πρώτης ροπής (m):

$$m^{(t)} = \beta_1 m^{(t-1)} + (1 - \beta_1) g^{(t)}$$

Ενημερώνονται οι εκτιμήσεις της δεύτερης ροπής (v):

$$v^{(t)} = \beta_2 v^{(t-1)} + (1 - \beta_2)\left(g^{(t)} \odot g^{(t)}\right)$$

Όπου $\odot$ είναι το γινόμενο Hadamard

Διορθώνεται ως $\hat{m}^{(t)}$ η απόκλιση της πρώτης ροπής $m^{(t)}$ και ως $\hat{v}^{(t)}$, της δεύτερης $v^{(t)}$ εκτίμησης της ροπής στην τρέχουσα επανάληψη (t)

$$\hat{m}^{(t)} = \frac{m^{(t)}}{1-\beta_1^t}$$

$$\hat{v}^{(t)} = \frac{v_2^{(t)}}{1-\beta_2^t}$$

Υπολογίζονται οι ρυθμοί προσαρμοσμένης μάθησης $\alpha^{(t)}$

$$\alpha^{(t)} = \frac{\alpha^{(t-1)}\sqrt{1-\beta_2^t}}{1-\beta_1^t}$$

Ενημερώνονται οι παράμετροι του μοντέλου χρησιμοποιώντας τους προσαρμοσμένους ρυθμούς εκμάθησης:

$$\theta^{(t)} = \theta^{(t-1)} - \frac{\alpha^{(t)}\hat{m}^{(t)}}{\sqrt{\hat{v}^{(t)}} + \epsilon}$$

Αυτή είναι μια υλοποίηση MATLAB του αλγόριθμου βελτιστοποίησης Adam όπως περιγράφεται παραπάνω.

Η κυρίαρχη ιδέα στον αλγόριθμο Adam, είναι ότι διατηρεί δύο κινητούς μέσους όρους: έναν για τις κλίσεις (πρώτη ροπή) και ένα για τα τετράγωνα των κλίσεων (δεύτερη ροπή). Στόχος του είναι να προσαρμόσει τον ρυθμό εκμάθησης για κάθε παράμετρο, με βάρη ανάλογα με το πόσο μεταβάλλονται οι κλίσεις με την πάροδο του χρόνου. Οι παράμετροι ενημερώνονται πιο γρήγορα όταν οι κλίσεις είναι σταθερές και πιο αργά όταν οι κλίσεις είναι ασταθείς.

Ο αλγόριθμος Adam συγκεντρώνει διάφορα πλεονεκτήματα που είναι:

Η Γρήγορη Σύγκλιση. Ο Adam τείνει να συγκλίνει γρήγορα σε μεγάλα προβλήματα βελτιστοποίησης, όπως είναι τα νευρωνικά δίκτυα, συνδυάζοντας τα καλύτερα στοιχεία του AdaGrad και του RMSProp (Kingma, 2014).

Η Ανθεκτικότητα σε Κλίσεις που παρουσιάζουν θόρυβο. Χάρη στη χρήση των εκθετικά σταθμισμένων κινητών μέσων όρων, ο Adam είναι ανθεκτικός σε δεδομένα με θόρυβο ή σε προβλήματα όπου οι κλίσεις μεταβάλλονται γρήγορα.

Η Αυτόματη Προσαρμογή του Ρυθμού Εκμάθησης. Ο Adam προσαρμόζει αυτόματα τον ρυθμό εκμάθησης για κάθε παράμετρο, μειώνοντας την ανάγκη για λεπτομερή ρύθμιση των υπερπαραμέτρων.

Η καλή λειτουργία σε Αραιά Δεδομένα, Όπως και το AdaGrad, ο Adam λειτουργεί καλά σε προβλήματα με αραιές κλίσεις, όπως είναι τα προβλήματα επεξεργασίας φυσικής γλώσσας ή συστημάτων προτάσεων.

Ο Adam είναι μια ευέλικτη επιλογή για πολλούς τύπους προβλημάτων και συχνά λειτουργεί καλά σε εφαρμογές βαθιάς μάθησης. Θεωρείται ιδιαίτερα κατάλληλος για δεδομένα μεγάλων διαστάσεων που χρησιμοποιούν μεγάλα νευρωνικά δίκτυα με πολλά επίπεδα και παραμέτρους.

Επίσης, για προβλήματα όπου οι κλίσεις είναι ασταθείς ή με θόρυβο, όπως κατά την εκπαίδευση με μικρές παρτίδες (mini-batches) ή σε ενισχυτική μάθηση. Τέλος, σε περιπτώσεις όπου η επιλογή του σωστού ρυθμού εκμάθησης ή άλλων υπερπαραμέτρων είναι δύσκολη, ο Adam παρέχει ένα σταθερό αποτέλεσμα με λιγότερη ρύθμιση, καθιστώντας τον έτσι, κατάλληλο για αρχικές πειραματικές εφαρμογές.

2.4.7. Adadelta

Το AdaDelta βασίζεται στην ιδέα ότι, αντί να χρησιμοποιούμε το άθροισμα όλων των προηγούμενων τετραγωνικών κλίσεων, μπορούμε να χρησιμοποιήσουμε ένα εκθετικά

σταθμισμένο κινητό μέσο όρο, το οποίο περιορίζει τον υπολογισμό του ιστορικού των κλίσεων σε ένα πιο διαχειρίσιμο εύρος (Zeiler, 2012). Αυτό διατηρεί τις ενημερώσεις των παραμέτρων σταθερές, επιλύοντας το πρόβλημα της υπερβολικής μείωσης του ρυθμού εκμάθησης με την πάροδο του χρόνου, όπως συμβαίνει στο AdaGrad.

Ο κανόνας ενημέρωσης του AdaDelta γίνεται σε δύο βήματα:

Το AdaDelta χρησιμοποιεί έναν εκθετικά σταθμισμένο κινητό μέσο όρο των τετραγωνικών κλίσεων:

$$E[g^2]_t = \rho E[g^2]_{t-1} + (1-\rho) g_t^2$$

Για την ενημέρωση των παραμέτρων στη μέθοδος AdaDelta, υπολογίζεται η ρίζα του κινητού μέσου όρου των τετραγώνων των κλίσεων (πάντα με μικρή προσθετική σταθερά για την περίπτωση που είναι μηδενική) με βάση την παρακάτω εξίσωση :

$$\Delta\theta_t = -\frac{\sqrt{E[\Delta\theta^2]_{t-1} + \epsilon}}{\sqrt{E[g^2]_t + \epsilon}} g_t$$

Τα πλεονεκτήματα του AdaDelta συνοψίζονται παρακάτω:

Δεν υπάρχει ανάγκη για χειροκίνητη επιλογή του ρυθμού εκμάθησης. Σε αντίθεση με άλλους αλγορίθμους όπως το AdaGrad ή το RMSProp, το AdaDelta εξαλείφει την ανάγκη για τη χειροκίνητη ρύθμιση του ρυθμού εκμάθησης, αφού προσαρμόζει αυτόματα την κλίμακα ενημερώσεων με βάση το ιστορικό των προηγούμενων ενημερώσεων.

Οι ενημερώσεις των παραμέτρων είναι σταθερές, χρησιμοποιώντας τον εκθετικά σταθμισμένο κινητό μέσο όρο. Το AdaDelta επιλύει το πρόβλημα της υπερβολικής μείωσης των ενημερώσεων που παρατηρείται στο AdaGrad, κρατώντας τις ενημερώσεις σταθερές κατά τη διάρκεια της εκπαίδευσης.

Ανθεκτικό σε Θορυβώδη Δεδομένα: Το σύστημα κινητού μέσου όρου του AdaDelta είναι αποτελεσματικό στην αντιμετώπιση δεδομένων με θόρυβο ή μη σταθερών συνόλων δεδομένων, όπως αυτά που συναντώνται σε μεγάλα σύνολα δεδομένων ή κατά την εκπαίδευση με μικρές παρτίδες (mini-batches).

Το AdaDelta είναι κατάλληλο για πολλά προβλήματα μηχανικής μάθησης, αλλά ξεχωρίζει ιδιαίτερα στις εξής περιπτώσεις:

Στην εκπαίδευση με μικρές παρτίδες είναι ιδιαίτερα αποτελεσματικό, όπου οι κλίσεις μπορεί να είναι ασταθείς και με θόρυβο.

Σε προβλήματα μεγάλων διαστάσεων, όπου το AdaDelta λειτουργεί καλά όταν η επιλογή ενός ενιαίου ρυθμού εκμάθησης για όλες τις παραμέτρους είναι αναποτελεσματική.

Σε περιπτώσεις όπου η επιλογή ενός κατάλληλου ρυθμού εκμάθησης είναι δύσκολη ή ο ρυθμός εκμάθησης πρέπει να προσαρμόζεται συνεχώς κατά τη διάρκεια της εκπαίδευσης, το AdaDelta μπορεί να είναι μια καλή επιλογή, καθώς εξαλείφει την ανάγκη για χειροκίνητη ρύθμιση του ρυθμού εκμάθησης.

2.4.8. Adamax

Ο αλγόριθμος βελτιστοποίησης Adamax, ο οποίος είναι μια παραλλαγή του Adam, χρησιμοποιεί τη **νόρμα L∞** (ή άπειρη νόρμα) για την προσαρμογή των βαρών κατά τη διαδικασία εκπαίδευσης των νευρωνικών δικτύων. Αυτή η νόρμα υπολογίζει την μέγιστη τιμή των παραμέτρων, προσφέροντας έτσι μεγαλύτερη σταθερότητα σε περιβάλλοντα με υψηλή διακύμανση ή θόρυβο στα δεδομένα. Συγκεκριμένα, ο Adamax προσαρμόζει τα βήματα εκμάθησης με βάση τη μέγιστη τιμή των παραμέτρων που έχουν παρατηρηθεί μέχρι εκείνη τη στιγμή, κάτι που μπορεί να είναι ιδιαίτερα χρήσιμο όταν οι παράμετροι έχουν πολύ διαφορετικές κλίμακες. Αυτό τον καθιστά αποτελεσματικό για διάφορες εφαρμογές σε βαθιά μάθηση και μηχανική μάθηση.

Ο αλγόριθμος Adamax χρησιμοποιεί τη νόρμα L∞ αντί για τη νόρμα L2 , που χρησιμοποιείται στον Adam, για τον υπολογισμό του εκθετικά σταθμισμένου κινητού μέσου όρου των τετραγωνικών κλίσεων. Με αυτόν τον τρόπο, αγνοεί τις ακραίες τιμές στις κλίσεις που μπορούν να οδηγήσουν σε προβλήματα σταθερότητας στις ενημερώσεις των παραμέτρων. Αυτό κάνει τον αλγόριθμο πιο σταθερό και ανθεκτικό σε μεγάλες κλίσεις ή κλίσεις που αλλάζουν δραματικά.
Όπως και ο Adam, ο Adamax διατηρεί ένα κινητό μέσο όρο των πρώτων και δεύτερων ροπών (των κλίσεων και των τετραγωνικών κλίσεων). Ωστόσο, αντί για τη χρήση της νόρμας L2 στα τετραγωνικά μεγέθη, το Adamax χρησιμοποιεί τη νόρμα L∞ για τον υπολογισμό του βήματος ενημέρωσης των παραμέτρων.

Ο αλγόριθμος ενημερώνει την πρώτη ροπή με βάση την κλίση της συνάρτησης απώλειας, όπως γίνεται και στον Adam:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$

Αντί όμως, για την εκθετικά σταθμισμένη δεύτερη ροπή (τετραγωνικές κλίσεις), ο Adamax χρησιμοποιεί την μέγιστη τιμή της κλίσης σε κάθε διάσταση:

$$u_t = \max(\beta_2 u_{t-1}, |g_t|)$$

Όπως και στον Adam, στη συνέχεια, εφαρμόζονται διορθώσεις απόκλισης για την πρώτη ροπή, ώστε να εξασφαλίζεται καλύτερη σύγκλιση στα πρώτα βήματα της εκπαίδευσης:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$$

Η ενημέρωση των παραμέτρων γίνεται ως εξής:

$$\theta_{t+1} = \theta_t - \frac{\eta}{u_t} \hat{m}_t$$

Τα πλεονεκτήματα του Adamax είναι:

Η ανθεκτικότητα σε μεγάλες κλίσεις. Χάρη στη χρήση της νόρμας L∞ , ο Adamax είναι πιο ανθεκτικός σε προβλήματα όπου οι κλίσεις μπορεί να έχουν εξαιρετικά μεγάλες ή ακραίες τιμές.

Αυτό επιτρέπει στον αλγόριθμο να αποφεύγει μεγάλες και απρόβλεπτες αλλαγές κατά την εκπαίδευση.

Η βελτιστοποίηση για μεγάλα μοντέλα. Ο Adamax μπορεί να είναι πιο αποδοτικός από τον Adam σε μεγάλα μοντέλα ή προβλήματα υψηλών διαστάσεων, επειδή αγνοεί τις ακραίες τιμές στις κλίσεις και ενημερώνει τις παραμέτρους με πιο σταθερό ρυθμό.

Η σταθερή ενημέρωση των παραμέτρων. Ο Adamax διατηρεί σταθερές ενημερώσεις των παραμέτρων, χωρίς να απαιτεί λεπτομερή ρύθμιση των υπερπαραμέτρων, γεγονός που τον καθιστά κατάλληλο για πολλά είδη προβλημάτων.

Ο Adamax είναι μια εξαιρετική επιλογή σε περιπτώσεις όπου υπάρχουν μεγάλες ή ακραίες κλίσεις, ή όταν τα δεδομένα ή το μοντέλο έχουν υψηλές διαστάσεις. Όταν υπάρχουν μοντέλα μεγάλης κλίμακας, όπως δίκτυα νευρώνων με πολλαπλά επίπεδα ή μεγάλα σύνολα δεδομένων, ο Adamax μπορεί να προσφέρει σταθερότητα που δεν προσφέρει ο Adam. Επίσης σε περιπτώσεις όπου οι κλίσεις μπορεί να περιέχουν ακραίες ή μεγάλες τιμές, όπως συμβαίνει σε δεδομένα με θόρυβο ή σε περιπτώσεις όπου οι παράμετροι διαφέρουν πολύ μεταξύ τους, ο Adamax παρέχει μεγαλύτερη ανθεκτικότητα και σταθερότητα.

Ο Adamax είναι τέλος, χρήσιμος όταν η δυναμική των κλίσεων αλλάζει δραματικά κατά τη διάρκεια της εκπαίδευσης, όπως συμβαίνει σε προβλήματα ενισχυτικής μάθησης ή άλλες μη στατικές καταστάσεις.

2.4.9. Adafactor

Ο Adafactor προσπαθεί να διαχειριστεί τη χρήση της μνήμης με έναν αποδοτικό τρόπο, αντικαθιστώντας την ανάγκη αποθήκευσης των δευτέρων ροπών (δηλαδή των τετραγωνικών κλίσεων) με μία προσεγγιστική αναπαράσταση που βασίζεται σε δύο διανύσματα. Χρησιμοποιεί έναν μηχανισμό που εκμεταλλεύεται τις ιδιότητες των διαστάσεων του τανυστή των παραμέτρων για να μειώσει τη μνήμη.

Η κύρια διαφορά του από τον Adam είναι ότι δεν αποθηκεύει την πλήρη δευτερεύουσα στιγμή (τετραγωνικές κλίσεις) σε υψηλής διάστασης παραμέτρους. Αντίθετα, χρησιμοποιεί μια προσεγγιστική μέθοδο για να ενημερώσει τις παραμέτρους χωρίς να απαιτεί πολλή μνήμη.

Ο αλγόριθμος διατηρεί έναν εκθετικά σταθμισμένο κινητό μέσο όρο των τετραγωνικών κλίσεων και χρησιμοποιεί την προσεγγιστική μέθοδο για να διαχειριστεί τη μνήμη.

Ο Εκθετικά Σταθμισμένος Μέσος Όρος Κλίσεων, είναι όπως και προηγουμένως :

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$

Ο Adafactor στη συνέχεια, χρησιμοποιεί μια προσεγγιστική μέθοδο για τον υπολογισμό των τετραγωνικών κλίσεων, αντί να αποθηκεύει τη δεύτερη ροπή ολόκληρη. Αντί για πλήρεις πίνακες, χρησιμοποιεί διανύσματα που μειώνουν τη μνήμη και βελτιστοποιούν την απόδοση σε μεγάλες κλίμακες παραμέτρων.

Για τανυστές δύο διαστάσεων, ενημερώνει τα ανεξάρτητα διανύσματα για τις γραμμές και τις στήλες. Για πιο σύνθετες περιπτώσεις, χρησιμοποιεί μια επέκταση αυτής της μεθόδου.

Ο Adafactor ενημερώνει τις παραμέτρους με τρόπο παρόμοιο με τον Adam, αλλά χωρίς να χρειάζεται η πλήρης αποθήκευση των δευτέρων ροπών:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{v_t} + \epsilon} m_t$$

Τα πλεονεκτήματα του Adafactor είναι:

Η μειωμένη κατανάλωση μνήμης. Ο Adafactor είναι ιδανικός για την εκπαίδευση πολύ μεγάλων μοντέλων, επειδή χρησιμοποιεί πολύ λιγότερη μνήμη από τον Adam. Αυτό επιτυγχάνεται χάρη στην προσεγγιστική μέθοδο αποθήκευσης της δεύτερης ροπής.

Η υψηλή αποδοτικότητα σε μεγάλες διαστάσεις. Σε μοντέλα με πολύ υψηλές διαστάσεις (π.χ. μεγάλα γλωσσικά μοντέλα), ο Adafactor εξασφαλίζει ότι η αποθήκευση της δεύτερης ροπής γίνεται με αποδοτικό τρόπο, καθιστώντας τον πολύ αποτελεσματικό σε μεγάλες κλίμακες.

Η πολύ μεγάλη κλίμακα μοντέλων. Ο Adafactor μπορεί να χρησιμοποιηθεί για την εκπαίδευση εξαιρετικά μεγάλων μοντέλων χωρίς τον περιορισμό της μνήμης, κάτι που τον καθιστά κατάλληλο για σύγχρονα βαθιά νευρωνικά δίκτυα.

Ο Adafactor είναι ιδιαίτερα κατάλληλος όταν έχουμε να κάνουμε με μεγάλα μοντέλα και περιορισμένους πόρους μνήμης. Χρησιμοποιείται σε περιπτώσεις με μεγάλα γλωσσικά μοντέλα ή μεγάλα νευρωνικά δίκτυα, όπως τα γλωσσικά μοντέλα Transformer, που απαιτούν τεράστιες ποσότητες μνήμης κατά την εκπαίδευση.

Όταν οι πόροι μνήμης είναι περιορισμένοι, ο Adafactor επιτρέπει την εκπαίδευση χωρίς να απαιτείται τεράστια μνήμη για την αποθήκευση των δευτέρων ροπών.

Τέλος, ο Adafactor είναι κατάλληλος για εφαρμογές σε μεγάλα μοντέλα και σύνολα δεδομένων, όπου η μνήμη είναι ένας σημαντικός περιορισμός και χρειάζεται η αποδοτική διαχείρισή της.

2.4.10. Nesterov Momentum

Η Nesterov Momentum ή Nesterov Accelerated Gradient (NAG), είναι μια τεχνική βελτιστοποίησης που επεκτείνει την κλασική μέθοδο του momentum, προσφέροντας μια πιο "έξυπνη" προσέγγιση για την ενημέρωση των παραμέτρων κατά τη διάρκεια της εκπαίδευσης νευρωνικών δικτύων. Ο αλγόριθμος Nesterov Momentum, που εισήχθη από τον Yurii Nesterov,

επιτρέπει στους αλγορίθμους βελτιστοποίησης να χρησιμοποιούν την πληροφορία της κατεύθυνσης των προηγούμενων βημάτων για να βελτιώσουν τη διαδικασία εκμάθησης.

Ο Nesterov Momentum επιτρέπει στον αλγόριθμο να "προβλέψει" την κατεύθυνση στην οποία θα κινηθούν οι παράμετροι, προτού υπολογίσει την κλίση. Αυτό συμβαίνει με την προσθήκη ενός προγνωστικού βήματος στην ενημέρωση των παραμέτρων, όπου ο αλγόριθμος εκτιμά πού θα βρίσκονται οι παράμετροι στο επόμενο βήμα και στη συνέχεια υπολογίζει την κλίση εκεί.

Αυτή η προσέγγιση δίνει μια πιο ενημερωμένη και ακριβή κατεύθυνση για την ενημέρωση, αποφεύγοντας τα μεγάλα βήματα που μπορεί να προκύψουν με τη χρήση του απλού momentum.

Ακολουθούν οι βασικές εξισώσεις που χρησιμοποιούνται στην Nesterov Momentum, μαζί με τις εξηγήσεις των μεταβλητών και των μεγεθών τους.

Ο υπολογισμός των παραμέτρων στη θέση προβολής:

$$\theta_{\text{lookahead}} = \theta - \gamma v$$

Ο υπολογισμός της κλίσης (gradient) στη θέση προβολής:

$$g = \nabla f(\theta_{\text{lookahead}})$$

Η ενημέρωση της ταχύτητας:

$$v = \gamma v + \alpha g$$

Η ενημέρωση των παραμέτρων:

$$\theta = \theta - v$$

Τα μεγέθη και μεταβλητές που περιέχονται στις παραπάνω εξισώσεις είναι :

$\boldsymbol{\theta}$: Οι παράμετροι του μοντέλου που θέλουμε να βελτιστοποιήσουμε, όπως τα βάρη σε ένα νευρωνικό δίκτυο.
$\boldsymbol{v}$: Η ταχύτητα ή η "μνήμη" του προηγούμενου βήματος ενημέρωσης, η οποία βοηθά στην επιτάχυνση της διαδικασίας εκπαίδευσης.
$\boldsymbol{\alpha}$: Ο ρυθμός εκμάθησης, ο οποίος καθορίζει πόσο θα αλλάξουν οι παράμετροι σε κάθε βήμα.
$\boldsymbol{\gamma}$: Ο παράγοντας momentum, ο οποίος ελέγχει πόσο επηρεάζει η προηγούμενη ταχύτητα την τρέχουσα ενημέρωση.
$\boldsymbol{g}$: Ο βαθμός της συνάρτησης κόστους, ο οποίος δείχνει την κατεύθυνση και το μέγεθος της αλλαγής που πρέπει να γίνει στις παραμέτρους για να μειωθεί το κόστος.

Η Nesterov Momentum διαφέρει από την κλασική μέθοδο momentum διότι υπολογίζει τον βαθμό όχι στη σημερινή θέση των παραμέτρων, αλλά σε μια "προβλεπόμενη" θέση, η οποία είναι το αποτέλεσμα της προηγούμενης ταχύτητας. Αυτή η προσέγγιση επιτρέπει στον αλγόριθμο να είναι πιο ευαίσθητος στις αλλαγές του gradient και να διορθώνει την πορεία του πιο αποτελεσματικά όταν πλησιάζει σε τοπικά ελάχιστα. Η Nesterov Momentum είναι ιδιαίτερα χρήσιμη σε περιβάλλοντα με πολύπλοκες επιφάνειες κόστους, καθώς επιταχύνει τη σύγκλιση και μειώνει την πιθανότητα να "κολλήσει" σε τοπικά ελάχιστα.

Η μέθοδος βελτιστοποίησης Nesterov Momentum παρουσιάζει ως πλεονεκτήματα:

Βελτιωμένη Σύγκλιση. Η τεχνική Nesterov παρέχει ταχύτερη και πιο σταθερή σύγκλιση, καθώς αξιοποιεί τις πληροφορίες της κατεύθυνσης της κλίσης πριν από την ενημέρωση των παραμέτρων.

Αποφυγή Μεγάλων Κινήσεων. Δεδομένου ότι ο Nesterov υπολογίζει την κλίση στο προβλεπόμενο σημείο, είναι λιγότερο πιθανό να γίνουν υπερβολικές ενημερώσεις που μπορεί να προκαλέσουν αστάθεια στη διαδικασία εκπαίδευσης.

Σαφή Κατεύθυνση Ενημέρωσης: Η προσέγγιση του Nesterov βοηθά τους αλγορίθμους βελτιστοποίησης να έχουν μια πιο "έξυπνη" κατεύθυνση ενημέρωσης, κάτι που βελτιώνει τη γενική τους απόδοση.

Ο Nesterov Momentum είναι κατάλληλος για περιπτώσεις όπου η γρήγορη και σταθερή σύγκλιση είναι κρίσιμη. Χρησιμοποιείται κυρίως σε:

Βαθιά Νευρωνικά Δίκτυα. Είναι ιδανικός για την εκπαίδευση βαθιών νευρωνικών δικτύων, όπου οι διακυμάνσεις στις κλίσεις μπορεί να είναι σημαντικές και οι ενημερώσεις πρέπει να είναι σταθερές.

Ασταθή Δεδομένα. Ο Nesterov είναι ιδιαίτερα χρήσιμος σε περιβάλλοντα με δεδομένα που έχουν θόρυβο, ή σε περιπτώσεις όπου οι κλίσεις είναι ασταθείς.

2.4.11. Nadam

Ο Nadam είναι μια εκτεταμένη μορφή του Adam που εφαρμόζει την ιδέα του Nesterov Momentum. Η διαφορά του από τον Adam είναι ότι ο αλγόριθμος λαμβάνει υπόψη την επιτάχυνση πριν από την εκτίμηση της κλίσης, κάτι που επιτρέπει πιο ευέλικτες ενημερώσεις παραμέτρων. Αυτό επιτυγχάνεται προβλέποντας πού θα βρίσκονται οι παράμετροι στο επόμενο βήμα και προσαρμόζοντας ανάλογα την κλίση.

Στο Nesterov Momentum, οι κλίσεις δεν υπολογίζονται μόνο με βάση την τρέχουσα θέση, αλλά και από ένα "προβλεπόμενο" σημείο στο οποίο θα κινηθούν οι παράμετροι. Αυτό δίνει μια πιο "έξυπνη" κατεύθυνση προς τη βέλτιστη λύση, καθώς λαμβάνει υπόψη την τρέχουσα επιτάχυνση.

Ο Nadam χρησιμοποιεί τις ίδιες εξισώσεις για τον υπολογισμό των κινητών μέσων όρων των κλίσεων και των τετραγωνικών κλίσεων όπως και ο Adam, αλλά ενσωματώνει την επιτάχυνση του Nesterov στο βήμα ενημέρωσης των παραμέτρων.

Πρώτη Ροπή (Κινητός Μέσος Όρος Κλίσεων):

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$

Δεύτερη Ροπή (Κινητός Μέσος Όρος Τετραγώνων των Κλίσεων):

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

Οι διορθώσεις απόκλισης για τους κινητούς μέσους όρους παραμένουν παρόμοιες με αυτές του Adam:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t},\ \hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

Η βασική διαφορά του Nadam είναι η προσθήκη της επιτάχυνσης του Nesterov στην ενημέρωση των παραμέτρων. Αντί να χρησιμοποιεί μόνο τον κινητό μέσο όρο των κλίσεων, προσθέτει και μια πρόβλεψη για το επόμενο βήμα:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} (\beta_1 \hat{m}_t + (1 - \beta_1) g_t)$$

Αυτό σημαίνει ότι το Nadam προβλέπει πώς οι παράμετροι θα μετακινηθούν, κάνοντάς το πιο αποδοτικό σε δυναμικές συνθήκες εκμάθησης.

Τα πλεονεκτήματα του Nadam είναι τα ακόλουθα:

Επιτάχυνση με Nesterov Momentum. Το βασικό πλεονέκτημα του Nadam είναι ότι επιτυγχάνει προς ταχύτερη σύγκλιση λόγω της πρόβλεψης της κατεύθυνσης των κλίσεων, όπως συμβαίνει με το Nesterov Momentum.

Γρηγορότερη και Σταθερότερη Σύγκλιση. Χάρις στον συνδυασμό του Adam με την προσέγγιση του Nesterov, ο Nadam τείνει να συγκλίνει πιο γρήγορα και σταθερά, ειδικά σε προβλήματα βαθιάς μάθησης με πολλές παραμέτρους.

Καλύτερη Απόδοση σε Δεδομέναμε θόρυβο. Ο Nadam, όπως και ο Adam, είναι ανθεκτικός σε δεδομένα με θόρυβο ή σε ασταθείς κλίσεις. Ωστόσο, με την προσθήκη του Nesterov, βελτιώνει ακόμα περισσότερο την προσαρμοστικότητα στις αλλαγές των κλίσεων.

Ανθεκτικότητα σε Ασταθείς Κλίσεις. Η τεχνική του Nesterov βοηθά τον αλγόριθμο να αποφύγει μεγάλες και ασταθείς ενημερώσεις, κάνοντάς τον πιο ανθεκτικό σε ακραίες ή ασταθείς κλίσεις.

Ο Nadam είναι κατάλληλος για τις ίδιες περιπτώσεις που χρησιμοποιείται ο Adam, με το πρόσθετο πλεονέκτημα της βελτιωμένης σύγκλισης χάρη στο Nesterov Momentum. Χρησιμοποιείται συνήθως όταν έχουμε:

Βαθιά Νευρωνικά Δίκτυα. Ο Nadam είναι ιδιαίτερα χρήσιμος για την εκπαίδευση βαθιών νευρωνικών δικτύων, όπου η γρήγορη και σταθερή σύγκλιση είναι κρίσιμη.

Ασταθή ή Θορυβώδη Δεδομένα. Όπως και ο Adam, ο Nadam είναι πολύ αποτελεσματικός σε προβλήματα με δεδομένα με θόρυβο ή ασταθείς κλίσεις, αλλά προσφέρει ακόμη πιο σταθερές ενημερώσεις χάρη στο Nesterov.

Προβλήματα Ενισχυτικής Μάθησης. Επειδή η μέθοδος του Nesterov επιτρέπει καλύτερη πρόβλεψη και σταθερότητα, ο Nadam είναι κατάλληλος για περιπτώσεις ενισχυτικής μάθησης ή άλλες δυναμικές συνθήκες όπου οι κλίσεις μπορεί να αλλάζουν γρήγορα.

2.4.12. AdamW

Ο AdamW χρησιμοποιεί τις ίδιες εξισώσεις για την υπολογισμό των κινητών μέσων όρων και της προσαρμογής των ρυθμών εκμάθησης, αλλά η ενημέρωση των παραμέτρων περιλαμβάνει την αποσύνθεση βαρών.

Πρώτη Ροπή (Κινητός Μέσος Όρος Κλίσεων):

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$

Δεύτερη Ροπή (Κινητός Μέσος Όρος Τετραγώνων Κλίσεων):

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

Οι διορθώσεις απόκλισης παραμένουν οι ίδιες όπως και στον Adam:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t},\ \hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

Ενημέρωση Παραμέτρων με Αποδόμηση των Βαρών (Weight Decay), που εφαρμόζεται χωριστά:

$$\theta_{t+1} = \theta_t - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} - \eta \lambda \theta_t$$

Τα πλεονεκτήματα του AdamW είναι:

Η Καλύτερη Γενίκευση. Ο AdamW βελτιώνει τη γενίκευση του μοντέλου μειώνοντας την υπερπροσαρμογή, καθώς η αποδόμηση των βαρών εφαρμόζεται πιο αποτελεσματικά.

Η Ανεξάρτητη Ρύθμιση. Ο AdamW, δίνει τη δυνατότητα για ανεξάρτητη ρύθμιση του ρυθμού εκμάθησης και του παράγοντα αποδόμησης των βαρών, επιτρέποντας μεγαλύτερη ευελιξία στη διαδικασία εκμάθησης.

Η Ταχύτερη Σύγκλιση. Η προσθήκη της αποδόμησης των βαρών οδηγεί σε πιο σταθερές και ταχύτερες ενημερώσεις παραμέτρων, με αποτέλεσμα ταχύτερη σύγκλιση στη βέλτιστη λύση.

Η Βελτιωμένη Απόδοση σε Πραγματικά Δεδομένα. Ο AdamW έχει αποδειχθεί ότι είναι αποτελεσματικός σε διάφορες εφαρμογές βαθιάς μάθησης, βελτιώνοντας την απόδοση σε πραγματικά δεδομένα.

Ο AdamW είναι κατάλληλος για περιπτώσεις όπου απαιτείται αποτελεσματική διαχείριση των παραμέτρων και βελτιωμένη γενίκευση. Χρησιμοποιείται :

Στα Βαθιά Νευρωνικά Δίκτυα. Είναι ιδανικός για την εκπαίδευση βαθιών νευρωνικών δικτύων, όπου οι παράμετροι μπορεί να είναι πολλές και η υπερπροσαρμογή είναι συχνό φαινόμενο.

Στα Δεδομένα μεγάλων διαστάσεων: Σε περιβάλλοντα όπου τα δεδομένα έχουν πολλές διαστάσεις, ο AdamW μπορεί να βοηθήσει στην αποφυγή της υπερπροσαρμογής και στη βελτίωση της απόδοσης.

Σε γενικές εφαρμογές βελτιστοποίησης, επειδή ο AdamW είναι μια καλή επιλογή, όπου απαιτείται αποτελεσματική ενημέρωση παραμέτρων με κατάλληλη ρύθμιση των βαρών.

2.4.13. FTRL (Follow The Regularized Leader)

Η FTRL είναι μια μέθοδος βελτιστοποίησης που χρησιμοποιείται κυρίως σε περιβάλλοντα διαδικτυακής εκμάθησης, όπου οι δεδομένες πληροφορίες γίνονται διαθέσιμες σταδιακά. Ο αλγόριθμος FTRL είναι ιδιαίτερα χρήσιμος για προβλήματα που περιλαμβάνουν μεγάλες διαστάσεις και απαιτούν την αποτελεσματική διαχείριση των παραμέτρων.

Ο FTRL βασίζεται στην ιδέα της παρακολούθησης του καλύτερου (ή "ηγετικού") μοντέλου κατά τη διάρκεια της διαδικασίας εκπαίδευσης, ενσωμαώνοντας τακτικούς περιορισμούς (regularization) που αποτρέπουν την υπερπροσαρμογή. Ο αλγόριθμος ενημερώνει τις παραμέτρους με βάση τις κλίσεις που προκύπτουν από τις απώλειες και την κανονικοποίηση, επιτρέποντας την προσαρμογή των βαρών με ευέλικτο τρόπο.

Ο FTRL χρησιμοποιεί την εξής διαδικασία για την ενημέρωση των παραμέτρων:

Υπολογισμός της Απώλειας. Ο αλγόριθμος υπολογίζει την απώλεια L για κάθε δεδομένο, όπου η απώλεια μπορεί να είναι οποιαδήποτε συνάρτηση απώλειας, όπως η λογιστική απώλεια ή η απώλεια τετραγωνικής απόκλισης.

Ενημέρωση των Παραμέτρων: Οι παράμετροι ενημερώνονται σύμφωνα με τον ακόλουθο τύπο:

$$\theta_{t+1} = \arg\min_{\theta} \left(\sum_{i=1}^{t} \langle g_i, \theta \rangle + R(\theta)\right)$$

Όπου:

θ_{t+1}: Οι παράμετροι του μοντέλου στην επόμενη χρονική στιγμή $t+1$.

g_i : Η κλίση (gradient) της συνάρτησης κόστους για την i-οστή παρατήρηση. Αντιπροσωπεύει την κατεύθυνση και το μέγεθος της αλλαγής που πρέπει να γίνει στις παραμέτρους.

$\langle g_i, \theta \rangle$ :: Το εσωτερικό γινόμενο μεταξύ του gradient g_i και των παραμέτρων θ, που υποδεικνύει την επίδραση της αλλαγής των παραμέτρων στο κόστος.

$R(\theta)$: Η κανονικοποίηση (regularization term), η οποία προστίθεται για να περιορίσει την πολυπλοκότητα του μοντέλου και να αποτρέψει την υπερβολική προσαρμογή (overfitting). Συχνά, η κανονικοποίηση μπορεί να είναι L1 ή L2.

Τα πλεονεκτήματα του FTRL είναι:

Η Αποτελεσματική Διαχείριση Δεδομένων. Ο FTRL είναι ιδιαίτερα αποτελεσματικός σε περιβάλλοντα όπου τα δεδομένα είναι διαθέσιμα σταδιακά, καθώς επιτρέπει την προσαρμογή των παραμέτρων με βάση τις νέες πληροφορίες.

Η Ανθεκτικότητα στην Υπερπροσαρμογή. Η ενσωμάτωση τακτικών περιορισμών (regularization) επιτρέπει στον αλγόριθμο να αποφεύγει την υπερπροσαρμογή και να διατηρεί την απόδοση του μοντέλου.

Η Αναλογική Ενημέρωση. Ο FTRL έχει την ικανότητα να προσαρμόζει γρήγορα τις παραμέτρους σε νέα δεδομένα, κάνοντάς τον ιδανικό για εφαρμογές με συνεχή ροή δεδομένων.

Ο FTRL είναι κατάλληλος για προβλήματα όπου οι πληροφορίες εισάγονται σταδιακά και απαιτούν συνεχείς προσαρμογές. Χρησιμοποιείται συνήθως στις εξής περιπτώσεις:

Διαδικτυακή Εκμάθηση. Είναι ιδανικός για εφαρμογές διαδικτυακής εκμάθησης, όπου τα δεδομένα έρχονται με τη μορφή ροών και οι αποφάσεις πρέπει να ληφθούν σε πραγματικό χρόνο.

Προβλήματα Μεγάλων Διαστάσεων: Ο FTRL αποδεικνύεται αποτελεσματικός σε περιβάλλοντα με πολλές παραμέτρους, όπως είναι η ανάλυση κειμένου και οι συστάσεις προϊόντων.

Προβλήματα Κατηγοριοποίησης και Παλινδρόμησης. Ο FTRL χρησιμοποιείται ευρέως σε προβλήματα κατηγοριοποίησης και παλινδρόμησης, όπου η κανονικοποίηση μπορεί να βοηθήσει στη βελτίωση της γενίκευσης.

2.4.14. Lion (Linearized Optimizer with Nesterov)

Η LION, είναι μια πρόσφατη προσέγγιση βελτιστοποίησης που συνδυάζει στοιχεία από τις παραδοσιακές μεθόδους βελτιστοποίησης και τις σύγχρονες τεχνικές, με στόχο την ταχύτερη και αποτελεσματικότερη εκπαίδευση νευρωνικών δικτύων. Ο LION είναι σχεδιασμένος για να

εκμεταλλεύεται τις πληροφορίες της κλίσης πιο αποδοτικά, επιτρέποντας την ταχύτερη σύγκλιση και τη βελτίωση της απόδοσης.

Ο LION βασίζεται στη γραμμική παραλλαγή του πρώτου επιπέδου, όπου εκμεταλλεύεται τη γραμμική σχέση μεταξύ των παραμέτρων και της απώλειας. Αντί να υπολογίζει τις ενημερώσεις των παραμέτρων απευθείας από την κλίση, ο LION αναλύει την γραμμική συμπεριφορά της συνάρτησης απώλειας γύρω από την τρέχουσα θέση των παραμέτρων και προσαρμόζει τις ενημερώσεις με βάση αυτή την ανάλυση. Ο LION προσδιορίζει γραμμικούς περιορισμούς γύρω από την τρέχουσα θέση των παραμέτρων, προσδιορίζοντας τη συμπεριφορά της συνάρτησης απώλειας.

Η διαδικασία ενημέρωσης στον LION περιλαμβάνει τα εξής βήματα:

Υπολογισμός της Κλίσης. Ο LION υπολογίζει την κλίση της συνάρτησης απώλειας ως προς τις παραμέτρους:

$$g_t = \nabla L(\theta_t)$$

g_t : Η κλίση (gradient) της συνάρτησης κόστους L στην τρέχουσα θέση των παραμέτρων θ_t

Ενημέρωση των παραμέτρων. Οι παράμετροι ενημερώνονται με βάση την γραμμική εκτίμηση της απώλειας:

$$\theta_{t+1} = \theta_t - \eta_t v_t$$

θ_{t+1} : Οι παράμετροι του μοντέλου στην επόμενη χρονική στιγμή.

η_t : Ο ρυθμός εκμάθησης (learning rate) στην t-οστή επανάληψη

v_t : Η ταχύτητα (velocity) που υπολογίζεται με βάση τον προηγούμενο βαθμό και τις παραμέτρους.

Υπολογισμός της ταχύτητας

$$v_t = \beta v_{t-1} + (1 - \beta) g_t$$

β : Ο παράγοντας momentum, που ελέγχει τη συνεισφορά της προηγούμενης ταχύτητας.

v_{t-1} : Η ταχύτητα στην προηγούμενη επανάληψη.

Διόρθωση της ταχύτητας.

$$v_{t+1} = \frac{v_t}{1 - \beta^t}$$

Αυτή η διόρθωση εξασφαλίζει ότι η ταχύτητα δεν είναι προκατειλημμένη στην αρχή της εκπαίδευσης. β είναι ο παράγοντας momentum, ο οποίος ελέγχει πόσο επηρεάζει η προηγούμενη ταχύτητα την τρέχουσα ενημέρωση.

Ο LION εμφανίζει τα εξής πλεονεκτήματα:

Ταχύτερη Σύγκλιση. Ο LION είναι σχεδιασμένος για να επιτυγχάνει ταχύτερη σύγκλιση από τους παραδοσιακούς αλγόριθμους βελτιστοποίησης, καθώς εκμεταλλεύεται τη γραμμική συμπεριφορά της απώλειας.

Αποτελεσματική Διαχείριση Πληροφοριών. Η προσέγγιση του LION επιτρέπει την πιο αποτελεσματική χρήση των πληροφοριών της κλίσης, μειώνοντας την ανάγκη για επαναλαμβανόμενες υπολογισμούς.

Βελτιωμένη Απόδοση σε Μεγάλα Μοντέλα. Ο LION αποδεικνύεται ιδιαίτερα αποτελεσματικός σε μεγάλα νευρωνικά δίκτυα, όπου η καλή διαχείριση των παραμέτρων είναι κρίσιμη.

Ο LION είναι κατάλληλος για περιπτώσεις όπου απαιτείται γρήγορη και αποτελεσματική βελτιστοποίηση. Χρησιμοποιείται σε :

Βαθιά Νευρωνικά Δίκτυα. Είναι ιδανικός για την εκπαίδευση πολύπλοκων μοντέλων βαθιάς μάθησης, όπου η ταχύτητα και η αποτελεσματικότητα είναι κρίσιμες.

Δεδομένα Μεγάλων Διαστάσεων. Ο LION είναι αποτελεσματικός σε περιβάλλοντα με πολλές παραμέτρους, επιτρέποντας τη βελτιστοποίηση των βαρών με ταχύτητα και ορθότητα.

Εφαρμογές με Συνεχή Ροή Δεδομένων. Ο LION είναι χρήσιμος σε περιπτώσεις όπου τα δεδομένα εισάγονται σταδιακά, απαιτώντας γρήγορες προσαρμογές στις παραμέτρους.

2.4.15. Loss Scale Optimizer

Η Loss Scale Optimizer είναι μια τεχνική βελτιστοποίησης που χρησιμοποιείται κυρίως σε εφαρμογές βαθιάς μάθησης για την αντιμετώπιση προβλημάτων με τη σταθερότητα και την ορθότητα της εκπαίδευσης, ειδικά όταν οι αριθμητικές τιμές των παραμέτρων είναι πολύ μικρές. Η κύρια ιδέα πίσω από τον Loss Scale Optimizer είναι η χρήση ενός παράγοντα κλίμακας (scaling factor) για να προσαρμόσει τις τιμές της απώλειας κατά τη διάρκεια της εκπαίδευσης, βελτιώνοντας έτσι τη σταθερότητα των ενημερώσεων των παραμέτρων.

Ο Loss Scale Optimizer επιτρέπει την εκπαίδευση μοντέλων με μικρές κλίσεις, βελτιώνοντας την ορθότητα των υπολογισμών και αποφεύγοντας την απώλεια πληροφορίας κατά τη διαδικασία της πίσω διάδοσης (backpropagation). Χρησιμοποιεί έναν παράγοντα κλίμακας για να "μεγαλώσει" τις τιμές της απώλειας πριν υπολογιστούν οι κλίσεις, επιτρέποντας έτσι καλύτερες και πιο σταθερές ενημερώσεις παραμέτρων.

Η διαδικασία ενημέρωσης με τον Loss Scale Optimizer περιλαμβάνει τα εξής βήματα:

Υπολογισμός της Απώλειας. Η απώλεια υπολογίζεται για το τρέχον δείγμα ή παρτίδα δεδομένων. Ανάλογα με το μοντέλο, αυτή μπορεί να είναι η λογιστική απώλεια, η απώλεια τετραγωνικής απόκλισης ή κάποια άλλη συνάρτηση απώλειας.

Η Εφαρμογή Κατάλληλης Κλίμακας, εφαρμόζεται πριν υπολογιστούν οι κλίσεις :

$$L' = \text{ scale } \times L$$

Οι κλίσεις υπολογίζονται από την απώλεια που έχει υποστεί αλλαγή κλίμακας:

$$g_t = \nabla L'$$

Οι παράμετροι ενημερώνονται με βάση τις κλίσεις που προκύπτουν από την απώλεια της προηγούμενης πράξης:

$$\theta_{t+1} = \theta_t - \eta g_t$$

Η τιμή της κλίμακας μπορεί να προσαρμοστεί κατά τη διάρκεια της εκπαίδευσης, ανάλογα με την κατάσταση των κλίσεων και την απόδοση του μοντέλου. Αν οι κλίσεις είναι πολύ μικρές ή προκαλούν αστάθεια, η κλίμακα μπορεί να αυξηθεί ή να μειωθεί.

Τα πλεονεκτήματα του Loss Scale Optimizer είναι:

Βελτιωμένη Σταθερότητα. Ο Loss Scale Optimizer βελτιώνει τη σταθερότητα κατά την εκπαίδευση, ειδικά σε περιβάλλοντα με μικρές κλίσεις, που μπορεί να προκαλέσουν αστάθεια ή να οδηγήσουν σε κακή απόδοση.

Αύξηση της Ορθότητας. Η χρήση της κλίμακας επιτρέπει καλύτερους υπολογισμούς των κλίσεων, μειώνοντας την πιθανότητα απώλειας πληροφορίας κατά τη διαδικασία της πίσω διάδοσης.
Ευελιξία στη Ρύθμιση. Η δυνατότητα προσαρμογής της κλίμακας κατά τη διάρκεια της εκπαίδευσης επιτρέπει στον αλγόριθμο να προσαρμόζεται στις απαιτήσεις των δεδομένων και της διαδικασίας εκπαίδευσης.

Ο Loss Scale Optimizer είναι κατάλληλος για περιπτώσεις όπου οι κλίσεις των παραμέτρων μπορεί να είναι πολύ μικρές ή να προκαλούν αστάθεια στην εκπαίδευση. Χρησιμοποιείται συνήθως σε περιπτώσεις όπως :

Τα Βαθιά Νευρωνικά Δίκτυα. Είναι ιδανικός για την εκπαίδευση πολύπλοκων μοντέλων βαθιάς μάθησης, όπου οι αριθμητικές τιμές μπορεί να είναι πολύ μικρές.

Τις Εφαρμογές με Στατιστική Ανάλυση. Χρησιμοποιείται σε περιπτώσεις όπου η ακριβής εκτίμηση των παραμέτρων είναι κρίσιμη για την απόδοση του μοντέλου.

Τα Δεδομένα Μεγάλων Διαστάσεων: Σε προβλήματα με πολλές παραμέτρους, η χρήση του Loss Scale Optimizer μπορεί να βοηθήσει στην καλή διαχείριση των ενημερώσεων των παραμέτρων και στη βελτίωση της γενίκευσης.

3. Υλοποίηση

Για την εφαρμογή των μεθόδων βελτιστοποίησης σε πραγματικά δεδομένα, που καλύπτουν του κυριότερους τομείς εφαρμογής των νευρωνικών δικτύων όπως η αναγνώριση εικόνων, η επεξεργασία κειμένου και η παλινδρόμηση, χρησιμοποιήθηκαν γνωστά σύνολα δεδομένων όπως τα MNIST, το οποίο περιέχει χειρόγραφα ψηφία, το CIFAR-10 και το CIFAR-100 που περιέχουν εικόνες από διάφορες κατηγορίες αντικειμένων, το Fashion MNIST με εικόνες ενδυμάτων, και τα σύνολα IMDB και Reuters για ταξινόμηση συναισθημάτων και ειδήσεων αντίστοιχα. Επίσης, χρησιμοποιήθηκε το California Housing για προβλήματα παλινδρόμησης των τιμών κατοικιών.

3.1. Συναρτήσεις Ενεργοποίησης

Στα νευρωνικά δίκτυα που εφαρμόστηκαν στην παρούσα διπλωματική εργασία, οι συναρτήσεις ενεργοποίησης έχουν σημαντικό ρόλο, επειδή εισάγουν την μη γραμμικότητα στο μοντέλο, ώστε να μπορεί να μάθει πολύπλοκες σχέσεις στα δεδομένα. Μετατρέπουν τις εξόδους κάθε νευρώνα σε κλίμακες που το μοντέλο μπορεί να ερμηνεύσει και να χρησιμοποιήσει για την εκπαίδευση. Οι συναρτήσεις ενεργοποίησης που χρησιμοποιήθηκαν σε αυτή την εργασία σε περιπτώσεις όπως η αναγνώριση εικόνων, η επεξεργασία κειμένου και η παλινδρόμηση.

ReLU (Rectified Linear Unit)

Η ReLU είναι μία από τις πιο δημοφιλείς συναρτήσεις ενεργοποίησης για κρυφά επίπεδα με εξίσωση:

$$f(x) = \max(0, x)$$

Όπου:

x είναι η είσοδος στον νευρώνα. Το αποτέλεσμα είναι το ίδιο το x αν είναι θετικό, διαφορετικά είναι 0. Η ReLU είναικατάλληλη για εφαρμογές αναγνώρισης εικόνων, όπως τα σύνολα δεδομένων MNIST, CIFAR-10 και Fashion MNIST, καθώς εισάγει μη γραμμικότητα στα νευρωνικά δίκτυα και επιτρέπει την αποτελεσματική εκπαίδευση βαθιών μοντέλων. Επειδή οι

αρνητικές τιμές αντικαθίστανται με μηδέν, το μοντέλο δεν έχει το πρόβλημα της εξαφάνισης του σφάλματος που έχει για παράδειγμα η sigmoid.

Sigmoid

Η συνάρτηση sigmoid δίνει τιμές εξόδου μεταξύ 0 και 1, κάτι που την καθιστά ιδιαίτερα κατάλληλη για δυαδική ταξινόμηση, όπως στην επεξεργασία κειμένου (π.χ. ταξινόμηση συναισθημάτων στο IMDB). Η εξίσωση της sigmoid είναι:

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

Όπου:

x είναι η είσοδος στον νευρώνα, και το e είναι η σταθερά Euler. Η sigmoid είναι κατάλληλη όταν θέλουμε να μοντελοποιήσουμε πιθανότητες σε δύο κατηγορίες, αλλά δεκτική στο πρόβλημα της εξαφάνισης του σφάλματος (σε σύγκριση με άλλες συναρτήσεις όπως η ReLU).

Tanh (Hyperbolic Tangent)

Η tanh είναι μια βελτιωμένη παραλλαγή της sigmoid, καθώς επιστρέφει τιμές μεταξύ -1 και 1, δίνοντας καλύτερα αποτελέσματα σε προβλήματα όπου οι τιμές εξόδου μπορεί να είναι θετικές ή αρνητικές. Χρησιμοποιείται συχνά στην επεξεργασία κειμένου και σε νευρωνικά δίκτυα που απαιτούν κλίμακες εξόδων με κέντρο το 0. Η εξίσωση της tanh είναι:

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

Όπου:

x είναι η είσοδος στον νευρώνα και το e^{x} αντιστοιχεί στον εκθετικό όρο με βάση το e. Η tanh είναι χρήσιμη σε προβλήματα ταξινόμησης, ειδικά όταν ο μετασχηματισμός των δεδομένων πρέπει να καλύπτει θετικές και αρνητικές τιμές.

Softmax

Η **softmax** είναι μια συνάρτηση που χρησιμοποιείται συχνά στο τελευταίο επίπεδο νευρωνικών δικτύων για ταξινόμηση σε πολλές κλάσεις, όπως για την ταξινόμηση ειδήσεων στο σύνολο δεδομένων Reuters ή για εικόνες σε σύνολα δεδομένων όπως το CIFAR-10 και το CIFAR-100. Η εξίσωση της softmax είναι:

$$f(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$$

Όπου : x_i είναι η είσοδος του νευρώνα για την κατηγορία i, με έξοδο την πιθανότητα που αντιστοιχεί στην κατηγορία i,

Το άθροισμα $\sum_j e^{x_j}$, όλων των εισόδων x_j σαν εκθέτες του e, κανονικοποιεί τις τιμές, ώστε το άθροισμα των εξόδων να είναι 1.

Η softmax μετατρέπει το διάνυσμα εξόδων σε πιθανότητες, και έτσι, είναι κατάλληλη για προβλήματα με πολλές κατηγορίες. Συνίσταται για την ταξινόμηση εικόνων και άλλων σύνθετων δεδομένων.

3.2. Μετρικές

Για την εξαγωγή ποσοτικών συμπερασμάτων, οι κύριες μετρικές που χρησιμοποιήθηκαν είναι accuracy (ορθότητα), precision (ακρίβεια), recall (ανάκληση) και F1-score.

Για την απώλεια χρησιμοποιήθηκαν οι Δυαδική διασταυρούμενη εντροπία (binary cross entropy), Αραιή κατηγορική διασταυρούμενη εντροπία (sparse categorical cross entropy) και Μέσο τετραγωνικό σφάλμα (mean squared error - MSE).

Η ορθότητα (accuracy), εκφράζει την ικανότητα του μοντέλου να προβλέπει σωστά τόσο τις θετικές όσο και τις αρνητικές κατηγορίες.

accuracy = $\frac{tp+tn}{tp+tn+fp+fn}$

όπου :

tp $true\ positive$ ($\theta\varepsilon\tau\iota\kappa\acute{o}\ \pi o\upsilon\ \mu\varepsilon\tau\rho\iota\acute{\varepsilon}\tau\alpha\iota\ \omega\varsigma\ \theta\varepsilon\tau\iota\kappa\acute{o}$),

tn $true\ negative$ ($\alpha\rho\nu\eta\tau\iota\kappa\acute{o}\ \pi o\upsilon\ \mu\varepsilon\tau\rho\iota\acute{\varepsilon}\tau\alpha\iota\ \omega\varsigma\ \alpha\rho\nu\eta\tau\iota\kappa\acute{o}$),

fp $false\ positive$ ($\alpha\rho\nu\eta\tau\iota\kappa\acute{o}\ \pi o\upsilon\ \mu\varepsilon\tau\rho\iota\acute{\varepsilon}\tau\alpha\iota\ \omega\varsigma\ \theta\varepsilon\tau\iota\kappa\acute{o}$),

fn $false\ negative$ ($\theta\varepsilon\tau\iota\kappa\acute{o}\ \pi o\upsilon\ \mu\varepsilon\tau\rho\iota\acute{\varepsilon}\tau\alpha\iota\ \omega\varsigma\ \alpha\rho\nu\eta\tau\iota\kappa\acute{o}$).

Η ακρίβεια (precision) εκφράζει την ικανότητα του ταξινομητή να μην επισημαίνει αρνητικά δείγματα ως θετικά.

precision = $tp/(tp + fp)$,

ενώ η ανάκληση (recall) αναφέρεται στην ικανότητα του ταξινομητή να εντοπίζει όλα τα θετικά δείγματα.

Recall = $tp/(tp + fn)$

Το F1-score είναι ο αρμονικός μέσος της ακρίβεια και της ανάκλησης, με την καλύτερη τιμή να είναι το 1 και τη χειρότερη το 0.

F1 = 2· (precision·recall) / (precision+recall)

3.3. Συναρτήσεις Κόστους

Η Binary Cross entropy (Δυαδική διασταυρούμενη εντροπία), χρησιμοποιείται για δυαδική ταξινόμηση, δηλαδή όταν έχουμε δύο κατηγορίες (π.χ. θετική ή αρνητική κατηγορία). Η εξίσωση της Δυαδικής διασταυρούμενης εντροπίας είναι:

Binary Cross entropy = $-\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1-y_i)\cdot \log(1-\hat{y}_l)]$

όπου:
N είναι ο αριθμός των δειγμάτων,
y_i είναι η πραγματική ετικέτα (0 ή 1) για το δείγμα i,
$\hat{y}_l$ είναι η προβλεπόμενη πιθανότητα για το δείγμα i να ανήκει στην κατηγορία 1.

Η Sparse Categorical Crossentropy (Αραιή κατηγορική διασταυρούμενη εντροπία) χρησιμοποιείται για ταξινόμηση με περισσότερες από δύο κατηγορίες, όπου οι ετικέτες είναι ακέραιοι αριθμοί και όχι δυαδικά κωδικοποιημένες. Η εξίσωση της Αραιής κατηγορικής διασταυρούμενης εντροπίας είναι:

Sparse Categorical Cross entropy = $-\frac{1}{N}\sum_{i=1}^{N}\log(\hat{y}_{i,c})$

όπου:
N είναι ο αριθμός των δειγμάτων,
c είναι η σωστή κατηγορία του δείγματος i,
$\hat{y}_{i,c}$ είναι η προβλεπόμενη πιθανότητα για το δείγμα i να ανήκει στην κατηγορία c.
Mean Squared Error (MSE) (Μέσο τετραγωνικό σφάλμα):

Χρησιμοποιείται κυρίως για προβλήματα παλινδρόμησης. Η εξίσωση του Μέσου τετραγωνικού σφάλματος είναι:

MSE = $\frac{1}{N}\sum_{i=1}^{N}(y_i-\hat{y}_i)^2$

όπου:
N είναι ο αριθμός των δειγμάτων,
y_i είναι η πραγματική τιμή για το δείγμα i,
$\hat{y}_i$ είναι η προβλεπόμενη τιμή για το δείγμα i.

Το MSE μετρά τη μέση τιμή των τετραγώνων των διαφορών μεταξύ των πραγματικών και των προβλεπόμενων τιμών.

3.4. Ανάλυση Κώδικα

Αυτές οι εξισώσεις χρησιμοποιούνται ευρέως για την αξιολόγηση μοντέλων ταξινόμησης και παλινδρόμησης, ανάλογα με τη φύση του προβλήματος.

Για την υλοποίηση του κώδικα που ακολουθεί, χρησιμοποιούνται αρκετές βιβλιοθήκες και εργαλεία της Python, που επιτρέπουν τη δημιουργία, την εκπαίδευση και την αξιολόγηση νευρωνικών δικτύων σε διάφορους τομείς εφαρμογής, όπως η αναγνώριση εικόνων, η επεξεργασία φυσικής γλώσσας και τα προβλήματα παλινδρόμησης.

Η βιβλιοθήκη Numpy παρέχει τη δυνατότητα εκτέλεσης μαθηματικών επεξεργασιών και ιδιαίτερα γραμμικής άλγεβρας με ταχύτητα και ορθότητα. Είναι ιδιαίτερα χρήσιμη για τη διαχείριση τανυστών και διανυσμάτων, συχνό γεγονός στα δεδομένα εκπαίδευσης. Η βιβλιοθήκη Pandas χρησιμοποιείται για την επεξεργασία και ανάλυση δεδομένων μέσω της δομής DataFrame, για τη διαχείριση σύνθετων δεδομένων.

Το TensorFlow και το Keras αποτελούν την βάση για την δημιουργία, εφαρμογή και εκπαίδευση των νευρωνικών δικτύων. Το Keras, χρησιμοποιώντας το Sequential API ως υψηλού επιπέδου Application Programming Interface του TensorFlow, βοηθά στη δημιουργία μοντέλων . Με αυτόν τον τρόπο, εισάγονται διάφορα είδη επιπέδων, όπως τα Dense (πλήρως συνδεδεμένα επίπεδα), Flatten (για την επίπεδη μετατροπή δεδομένων), Conv2D(για συνελικτικά νευρωνικά δίκτυα),και SimpleRNN (για επαναλαμβανόμενα νευρωνικά δίκτυα). Αυτά τα επίπεδα οδηγούν στη δημιουργία νευρωνικών δικτύων που διαχειρίζονται ποικίλα είδη προβλημάτων, όπως ταξινόμηση και παλινδρόμηση.

Ένα σημαντικό κομμάτι της εκπαίδευσης των μοντέλων αφορά τη βελτιστοποίηση των βαρών. Για αυτό τον σκοπό, χρησιμοποιούνται διάφοροι βελτιστοποιητές από το Keras, όπως ο RMSprop, ο SGD, ο Adam, καθώς και σχετικοί με τον τελευταίο, βελτιστοποιητές όπως οι Adadelta, Adamax και Nadam. Οι βελτιστοποιητές αυτοί εφαρμόζουν διαφορετικές μεθοδολογίες για την

αναπροσαρμογή των βαρών και επιταχύνουν τη σύγκλιση του μοντέλου κατά τη διάρκεια της εκπαίδευσης.

Για γραφικά αποτελέσματα και αναπαραστάσεις, χρησιμοποιούνται οι βιβλιοθήκες Matplotlib και Seaborn. Το Matplotlib χρησιμοποιείται για τη δημιουργία γραφημάτων ορθότητας, ενώ το Seaborn προσφέρει εξελιγμένες δυνατότητες απεικόνισης δεδομένων και στατιστικών αναλύσεων καθιστώντας τα αποτελέσματα πιο κατανοητά.

Για την αξιολόγηση των αποτελεσμάτων, χρησιμοποιείται η βιβλιοθήκη Scikit-learn (sklearn), η οποία προσφέρει κατάλληλες συναρτήσεις για αυτά, όπως το classification_report για την ανάλυση ταξινόμησης και το mean_squared_error για προβλήματα παλινδρόμησης. Τα μέτρα αυτά χρησιμοποιούνται για να αξιολογηθεί η ορθότητα και η ποιότητα των προβλέψεων του μοντέλου.

3.4.1. Νευρωνικά Δίκτυα Εμπροσθοδιάδοσης (Feed Forward Neural Networks)

Το σύνολο δεδομένων MNIST

Αφού κληθούν οι απαραίτητες βιβλιοθήκες πρώτα, χρησιμοποιείται το σύνολο δεδομένων MNIST (το οποίο αποτελείται από 70.000 δείγματα εικόνων (28x28 pixels) χειρόγραφων ψηφίων που έχουν κατηγοριοποιηθεί από το 0 έως το 9.). Το σύνολο MNIST διαχωρίζεται σε δύο σύνολα, ένα για εκπαίδευση και ένα για την επαλήθευση. Το σύνολο εκπαίδευσης, όπως υποδηλώνει και το όνομα του, χρησιμοποιείται για την εκπαίδευση του νευρωνικού δικτύου, ώστε να διαχωρίζει το καθένα από τα χειρόγραφα ψηφία του συνόλου. Το σύνολο για δοκιμές χρησιμοποιείται για την αξιολόγηση της απόδοσης του νευρωνικού δικτύου σε νέα, άγνωστα (αόρατα) δεδομένα, που δεν χρησιμοποιήθηκαν κατά την εκπαίδευση του μοντέλου. Η διαδικασία αυτή του διαχωρισμού του συνόλου δεδομένων θα επαναληφθεί για όλα τα σύνολα δεδομένων σε όλες τις αρχιτεκτονικές νευρωνικών δικτύων.

Στην προκαταρκτική επεξεργασία των δεδομένων, κανονικοποιούμε τις τιμές των εικονοστοιχειών του MNIST, έτσι ώστε να μπορεί το νευρωνικό δικτύο να τα επεξεργαστεί ευκολότερα. Η διαδικασία της κανονικοποίησης θα πραγματοποιηθεί σε όλα τα σύνολα δεδομένων και όλους τους τύπους των ΝΔ. που θα χρησιμοποιηθούν.

Αρχιτεκτονική των Ν.Δ. Εμπροσθοδιάσοσης

Όλα τα μοντέλα που χρησιμοποιούνται είναι Σειριακά -Sequential-, εξασφαλίζοντας ότι τα επίπεδα είναι διαδοχικά διατεταγμένα, με είσοδο κάθε επιπέδου την έξοδο από το προηγούμενο επίπεδο. Η αρχιτεκτονική για τη δημιουργία του μοντέλου εμπροσθοδιάδοσης εδώ, περιλαμβάνει ένα επίπεδο προεργασίας (Flatten) και τρία βασικά επίπεδα.

Το επίπεδο Flatten είναι το πρώτο στάδιο στο μοντέλο και μετατρέπει τα δεδομένα εισόδου. Οι εικόνες έχουν διαστάσεις 28x28 pixels και εισέρχονται στο δίκτυο ως δισδιάστατοι πίνακες. Το Flatten επίπεδο μετασχηματίζει αυτές τις δισδιάστατες εικόνες - εισόδου σε ένα μονοδιάστατο διάνυσμα με 784 στοιχεία (28x28=784), για να περάσουν στα επόμενα πλήρως συνδεδεμένα επίπεδα (Dense layers). Το Flatten δεν πραγματοποιεί κανέναν υπολογισμό μάθησης, αλλά απλώς προετοιμάζει τα δεδομένα για το πρώτο επίπεδο επεξεργασίας.

Το πρώτο επίπεδο (Dense layer) επεξεργασίας είναι πλήρως συνδεδεμένο με 128 νευρώνες, είναι το κύριο επίπεδο εισόδου και χρησιμοποιεί τη συνάρτηση ενεργοποίησης ReLU. Ακολουθεί ένα κρυφό επίπεδο (hidden layer) με 64 νευρώνες, που επίσης χρησιμοποιεί τη συνάρτηση ενεργοποίησης ReLU για να διατηρήσει τη μη γραμμικότητα στο δίκτυο και να μάθει πιο πολύπλοκα χαρακτηριστικά από τα δεδομένα. Τέλος, το επίπεδο εξόδου αποτελείται από 10 νευρώνες, καθένας από τους οποίους αντιστοιχεί σε μία κατηγορία χειρόγραφου ψηφίου (0-9). Η συνάρτηση ενεργοποίησης στο επίπεδο εξόδου είναι η softmax, και μετατρέπει τις εξόδους σε πιθανότητες, επιτρέποντας την ταξινόμηση των εικόνων εισόδου σε μία από τις 10 κατηγορίες.

Για την εκπαίδευση του μοντέλου χρησιμοποιείται η συνάρτηση κόστους «sparse_categorical_crossentropy», η οποία είναι κατάλληλη για ταξινόμηση σε πολλές κλάσεις, με ακέραιες ετικέτες. Αυτή η συνάρτηση κόστους συγκρίνει την πραγματική κατηγορία με την κατηγορία που προβλέπει το μοντέλο και υπολογίζει το σφάλμα, το οποίο σφάλμα, ελαχιστοποιείται από το μοντέλο κατά την εκπαίδευση. Η απόδοση του μοντέλου αξιολογείται με βάση τη μετρική accuracy (ορθότητα), η οποία μετρά το ποσοστό των σωστών προβλέψεων που έκανε το μοντέλο στις κατηγορίες των ψηφίων.

Συνοπτικά, το μοντέλο εκπαιδεύεται ώστε να προβλέπει σωστά το ψηφίο που αντιστοιχεί σε κάθε εικόνα, χρησιμοποιώντας τις πιθανότητες ταξινόμησης που παράγει η softmax στο επίπεδο εξόδου

και βελτιστοποιώντας το σφάλμα μέσω της συνάρτησης κόστους sparse_categorical_crossentropy.

Παρακάτω είναι ο αντίστοιχος κώδικας :

```
model=Sequential()

model.add(Flatten(input_shape=(28,28)))

model.add(Dense(128,activation="relu"))

model.add(Dense(64, activation='relu'))

model.add(Dense(10,activation="softmax"))


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

Για κάθε σύνολο δεδομένων και κάθε αρχιτεκτονική μοντέλου, δημιουργείται ένα λεξικό που περιλαμβάνει τους αλγορίθμους βελτιστοποίησης. Κάθε αλγόριθμος εκπαιδεύεται για 25 εποχές, και 20% του συνόλου εκπαίδευσης επιλεγμένο με τυχαίο τρόπο, χρησιμοποιείται για επαλήθευση (validation). Μετά από την εκπαίδευση του κάθε αλγορίθμου, εκτυπώνεται η ορθότητα (accuracy) του μοντέλου. Παράλληλα, εμφανίζονται τα ιστογράμματα για τις μετρικές ακρίβεια (precision), ανάκληση (recall) και F1-score, καθώς και οι γραφικές παραστάσεις που απεικονίζουν την ορθότητα και την απώλεια (loss) τόσο για το σύνολο εκπαίδευσης όσο και για το σύνολο επαλήθευσης.

Αυτή η διαδικασία εφαρμόζεται σε όλα τα σύνολα δεδομένων και τις αρχιτεκτονικές.

Το σύνολο δεδομένων CIFAR-10

Η ίδια δομή κώδικα εφαρμόζεται και στο σύνολο δεδομένων CIFAR-10, το οποίο περιέχει 60.000 έγχρωμες εικόνες μικρού μεγέθους (32x32 pixels) κατανεμημένες σε 10 κατηγορίες. Η διαφορά

σε σχέση με άλλα σύνολα δεδομένων έγκειται στο επίπεδο Flatten, όπου το σχήμα εισόδου (input shape) ορίζεται ως πίνακας (32, 32, 3) για να ταιριάζει με τις διαστάσεις και τα χρώματα των εικόνων του CIFAR-10. Ο κώδικας είναι:

model.add(Flatten(input_shape=(32,32,3)))

Το σύνολο δεδομένων CIFAR-100

Το σύνολο δεδομένων CIFAR-100 έχει ακριβώς την ίδια δομή κώδικα με το CIFAR-10, με τη μόνη διαφορά να βρίσκεται στο επίπεδο εξόδου. Επειδή το CIFAR-100 περιλαμβάνει 100 κατηγορίες, το επίπεδο εξόδου του μοντέλου πρέπει να έχει 100 νευρώνες αντί για 10, προκειμένου να γίνει η ταξινόμηση στις 100 κατηγορίες. Η συνάρτηση ενεργοποίησης είναι και εδώ η «softmax», για να διασφαλιστεί ότι η έξοδος του μοντέλου είναι μια κατανομή πιθανοτήτων ανάμεσα στις 100 κατηγορίες.

Ο αντίστοιχος κώδικας για το επίπεδο εξόδου είναι :

model.add(Dense(100,activation="softmax"))

Το σύνολο δεδομένων Fashion MNIST

Το Fashion MNIST είναι ένα σύνολο δεδομένων, που περιέχει εικόνες ειδών ρουχισμού και υπόδησης, σχεδιασμένο ως αντικαταστάτης του κλασικού MNIST dataset που περιέχει εικόνες χειρόγραφων ψηφίων. Όπως και το MNIST, αποτελείται από 70.000 ασπρόμαυρες εικόνες διαστάσεων 28x28 pixels, χωρισμένες σε 10 κατηγορίες, αλλά αντί για ψηφία, οι εικόνες αντιπροσωπεύουν είδη ρουχισμού και υπόδησης.

Η δομή του κώδικα και η διαδικασία εκπαίδευσης που ακολουθείται για το Fashion MNIST είναι ακριβώς η ίδια με αυτή που χρησιμοποιείται για το MNIST.

Το σύνολο δεδομένων IMDB

Ο κώδικας που ακολουθεί χρησιμοποιεί το IMDB σύνολο δεδομένων, που περιέχει κριτικές ταινιών σε μορφή κειμένου και τις αντίστοιχες ετικέτες θετικότητας ή αρνητικότητας. Το σύνολο δεδομένων έχει ήδη μετατραπεί σε αριθμητικά δεδομένα, όπου οι λέξεις μετατρέπονται σε ακολουθίες αριθμών.

Λόγω της φύσης αυτού του συνόλου δεδομένων, είναι απαραίτητη η προεπεξεργασία για να γίνει διαχειρίσιμο το μέγεθός του. Για να περιοριστεί ο όγκος των δεδομένων, φορτώνονται μόνο οι 10.000 πιο συχνά χρησιμοποιούμενες λέξεις. Αυτό επιτυγχάνεται με την παρακάτω εντολή:

(X_train,y_train),(X_test,y_test)=keras.datasets.imdb.load_data(num_words=10000)

Στη συνέχεια, εφαρμόζεται η τεχνική του padding ώστε όλες οι ακολουθίες αριθμών, οι οποίες αντιπροσωπεύουν τις κριτικές (προτάσεις), να έχουν το ίδιο μήκος. Το μέγιστο μήκος των ακολουθιών ορίζεται στις 200 λέξεις, εξασφαλίζοντας έτσι ότι οι προτάσεις με λιγότερες από 200 λέξεις συμπληρώνονται με μηδενικά (padded), ενώ οι μεγαλύτερες προτάσεις περικόπτονται (truncated) στο απαιτούμενο μήκος.

Ο αντίστοιχος κώδικας είναι :

maxlen = 200 # Maximum length of sequence

X_train = pad_sequences(X_train, maxlen=maxlen)

X_test = pad_sequences(X_test, maxlen=maxlen)

Η δομή είναι η ίδια, με δύο διαφοροποιήσεις. Πρώτα, προστέθηκε το Embedding layer στην αρχή του μοντέλου. Το επίπεδο αυτό μετατρέπει τους δείκτες των λέξεων σε πυκνά διανύσματα σταθερού μεγέθους. Χρησιμοποιεί τις λέξεις που έχουν μετατραπεί σε αριθμητικούς δείκτες από το dataset και δημιουργεί διανύσματα 128 διαστάσεων για κάθε λέξη. Ο αριθμός των μοναδικών λέξεων που λαμβάνονται υπόψη ορίζεται σε 10.000, που αντιστοιχεί στις πιο συχνές λέξεις από το σύνολο δεδομένων.

Η δεύτερη διαφοροποίηση, αφορά το επίπεδο εξόδου, το οποίο τώρα αποτελείται από έναν μόνο νευρώνα που χρησιμοποιεί τη συνάρτηση ενεργοποίησης «sigmoid». Αυτή η συνάρτηση μετατρέπει την έξοδο σε μία τιμή μεταξύ 0 και 1, αντιπροσωπεύοντας την πιθανότητα η κριτική να είναι είτε θετική είτε αρνητική, καθώς το πρόβλημα είναι δυαδικής ταξινόμησης.

Για την εκπαίδευση του μοντέλου, χρησιμοποιείται η συνάρτηση κόστους «binary_crossentropy», η οποία είναι κατάλληλη για δυαδικά προβλήματα ταξινόμησης. Αυτή η συνάρτηση συγκρίνει τις πραγματικές ετικέτες των κριτικών (θετικές ή αρνητικές) με τις προβλέψεις του μοντέλου και υπολογίζει το σφάλμα, το οποίο το μοντέλο προσπαθεί να ελαχιστοποιήσει κατά την εκπαίδευση. Ο κώδικας είναι:

```
model.add(Embedding(input_dim=10000, output_dim=128))

model.add(Flatten())

model.add(Dense(128,activation="relu"))

mod el.add(Dense(64,activation='relu'))

model.add(Dense(1,activation="sigmoid"))

model.compile(loss="binary_crossentropy",optimizer=optimizer,metrics=["accuracy"])
```

Το σύνολο δεδομένων Reuters

Η προκαταρκτική επεξεργασία των δεδομένων είναι παρόποια και για το σύνολο δεδομένων Reuters, το οποίο αποτελείται από ειδησεογραφικά κείμενα που έχουν ταξινομηθεί σε διάφορες θεματικές κατηγορίες. Η αρχιτεκτονική του μοντέλου είναι η ίδια, συμπεριλαμβάνοντας και εδώ ένα επίπεδο που παρεμβάλλεται -embedding layer- το οποίο μετατρέπει λέξεις σε πυκνά διανύσματα. Η κύρια διαφορά έγκειται στο επίπεδο εξόδου, όπου ο αριθμός των νευρώνων αντιστοιχεί στον αριθμό των κατηγοριών (κλάσεων) του συνόλου δεδομένων. Η συνάρτηση ενεργοποίησης που χρησιμοποιείται στο επίπεδο εξόδου είναι η softmax, που δίνει μια κατανομή πιθανοτήτων για κάθε κατηγορία. Τέλος, εδώ, η συνάρτηση απώλειας που εφαρμόζεται είναι η

sparse_categorical_crossentropy, η οποία θεωρείται αρκετά κατάλληλη στις ταξινομήσεις σε πολλές κατηγορίες. Η εκπαίδευση του μοντέλου υλοποιείται με τη συγκεκριμένη συνάρτηση απώλειας και τον κατάλληλο βελτιστοποιητή, και αποδίδει την ορθότητα κατά την αξιολόγηση της απόδοσής του.

Ο κώδικας είναι :

model.add(Dense(num_classes, activation='softmax'))

model.compile(loss="sparse_categorical_crossentropy",optimizer=optimizer,metrics=["accuracy"])

Το σύνολο δεδομένων California Housing

Ο κώδικας που ακολουθεί, χρησιμοποιεί τα δεδομένα California Housing, που περιέχουν πληροφορίες για το μέσο εισόδημα, τον μέσο αριθμό δωματίων κ.λπ., σε δειγματολειπτικές περιοχές της Καλιφόρνιας, και χρησιμοποιείται για πρόβλεψη της αξίας των κατοικιών, με την χρήση της βιβλιοθήκης «sklearn» της python.

housing = fetch_california_housing()

Ακολουθεί, ο αρχικός διαχωρισμός του συνόλου δεδομένων, όπου το X_train_full και y_train δίνουν πληροφορίες για το εισόδημα, τον αριθμό δωματίων σπιτιού κ.α. ενώ τα δεδομένα y_train_full, y_test είναι οι τιμές των σπιτιών.

X_train_full, X_test, y_train_full, y_test = train_test_split(housing.data, housing.target)

Στη συνέχεια υπάρχει και δεύτερος διαχωρισμός, όπου το σύνολο εκπαίδευσης διαχωρίζεται ξανά σε σύνολα εκπαίδευσης και επικύρωσης.

X_train, X_valid, y_train, y_valid = train_test_split(X_train_full, y_train_full)

Μετά, γίνεται η προεπεξεργασια και η κανονικοποίηση των δεδομένων. Εφαρμόζεται ο μετασχηματιστής κλίμακας και κανονικοποιητής - StandardScaler – που κανονικοποιεί τα δεδομένα με μέση τιμή (mean) 0 και τυπική απόκλιση (standard deviation) 1. Χρησιμοποιείται ο συγκεκριμένος μετασχηματιστής , επειδή το σύνολο δεδομένων περιέχει ποικιλόμορφα χαρακτηριστικά , που ευνοεί την προώθηση χαρακτηριστικών με μεγαλύτερες τιμές στην εκπαίδευση του μοντέλου.

scaler = StandardScaler()

X_train = scaler.fit_transform(X_train)

X_test = scaler.transform(X_test)

Η αρχιτεκτονική του μοντέλου περιλαμβάνει ένα επίπεδο εισόδου 64 νευρώνων, ένα ενδιάμεσο επίπεδο 32 νευρώνων, και ένα επίπεδο εξόδου ενός νευρώνα, που χρησιμοποιείται για την πρόβλεψη των τιμών των κατοικιών. Επειδή έχουμε παλινδρόμηση, η αξιολόγηση του μοντέλου, με τη χρήση του μέσου τετραγωνικού σφάλματος (mean squared error), θεωρείται η πλέον κατάλληλη συνάρτηση απώλειας για τέτοιες περιπτώσεις. Ο κώδικας που χρησιμοποιήθηκε είναι:

model = Sequential()

model.add(Dense(64, activation='relu', input_shape=(X_train.shape[1],)))

model.add(Dense(32,activation='relu'))

model.add(Dense(1))

model.compile(loss="mean_squared_error", optimizer=optimizer)

Παρόμοια με προηγουμένως, ένα λεξικό περιλαμβάνει τους αλγόριθμους βελτιστοποίησης. Κάθε ένας αλγόριθμος εκπαιδεύεται για 25 εποχές, και το 20% του συνόλου εκπαίδευσης χρησιμοποιείται για επαλήθευση (validation). Μετά την εκπαίδευση κάθε αλγορίθμου, εκτυπώνεται το μέσο τετραγωνικό σφάλμα (mean squared error), και παρουσιάζονταιτα γραφήματα του μέσου τετραγωνικού σφάλματος για το σύνολο εκπαίδευσης και επαλήθευσης.

3.4.2. Συνελικτικά Νευρωνικά Δίκτυα (Convolutional Neural Networks)

Το σύνολο δεδομένων MNIST

Αρχικά αφού το σύνολο MNIST κανονικοποιείται, μετά και επειδή τα Συνελικτικά Νευρωνικά Δίκτυα, δέχονται δεδομένα εισόδου σε μορφή τετραδιάστατων τανυστών, ώστε να μπορούν να επεξεργαστούν τις εικόνες και να εφαρμόσουν τα φίλτρα τους στις διαστάσεις ύψους και πλάτους ο κώδικας που ακολουθεί, μετασχηματίζει τα σύνολα εκπαίδευσης και δοκιμών σε τετραδιάστατους πίνακες. Η πρώτη διάσταση είναι το πλήθος των εικόνων, οι επόμενες δύο διαστάσεις είναι το ύψος και το πλάτος της κάθε εικόνας (28×28 pixels), και η τελευταία διάσταση είναι ο αριθμός διαύλων μήκους κύματος (στο ορατό μήκος κύματος είναι τα χρώματα), και επειδή οι εικόνες εδώ είναι ασπρόμαυρες είναι ίση με ένα.

X_train = X_train.reshape(60000,28,28,1)

X_test = X_test.reshape(10000,28,28,1)

Η αρχιτεκτονική του Ν.Δ. έχει αρχικά ένα συνελικτικό επίπεδο που εφαρμόζει φίλτρο 3×3 και χρησιμοποιεί την συνάρτηση ενεργοποίησης ReLU, μετά από ένα επίπεδο MaxPooling που χρησιμοποιεί φίλτρο 2×2 για υποδειγματοληψία (subsampling). Κατόπιν, εφαρμόζεται ένα επίπεδο Dropout που περιλαμβάνει με τυχαίο τρόπο το 25% των στοιχείων για να αποφευχθεί η υπερπροσαρμογή (overfitting). Το ίδιο μοτίβο ισχύει σε ένα ακόμη συνελικτικό επίπεδο, αλλά με 64 νευρώνες. Ακολουθεί ένα επίπεδο Flatten, το οποίο προετοιμάζει τα δεδομένα για τα πλήρως συνδεδεμένα επίπεδα. Ένα πυκνό επίπεδο με 64 νευρώνες και συνάρτηση ενεργοποίησης ReLU ακολουθεί, και τέλος, το επίπεδο εξόδου έχει 10 νευρώνες με συνάρτηση ενεργοποίησης την «softmax» για την ταξινόμηση των 10 κατηγοριών.

Ο κώδικας για αυτά τα συνελικτικά Ν.Δ., είναι:

model = Sequential()

model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Dropout(0.25))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Dropout(0.25))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))

model.add(layers.Flatten())

model.add(layers.Dense(64, activation='relu'))

model.add(layers.Dense(10, activation= 'softmax'))

model.compile(loss="sparse_categorical_crossentropy",optimizer=optimizer,metrics=["accuracy"])

Εδώ, δημιουργείται ένα λεξικό που περιλαμβάνει αλγόριθμους βελτιστοποίησης που υπάρχουν στο Keras. Κάθε αλγόριθμος εκπαιδεύεται για 25 εποχές, με 20% του συνόλου εκπαίδευσης να χρησιμοποιείται για επαλήθευση (validation). Μετά την εκπαίδευση του κάθε αλγορίθμου, εκτυπώνεται η ορθότητα (accuracy) του μοντέλου. Παράλληλα, εμφανίζονται τα ιστογράμματα για τις μετρικές ακρίβειας (precision), ανάκλησης (recall) και F1-score, καθώς και τα γραφήματα ορθότητας και απώλειας (loss), για το σύνολο εκπαίδευσης και για το σύνολο επαλήθευσης. Η μεθοδολογία αυτή, εφαρμόζεται σε όλα τα σύνολα δεδομένων και τις αρχιτεκτονικές CNNs, για την αξιολόγηση των μοντέλων.

Το σύνολο δεδομένων CIFAR-10

Στο σύνολο CIFAR-10, όπως και στο MNIST, μετά την εισαγωγή των δεδομένων, πραγματοποιείται η κανονικοποίηση των τιμών τους. Η αρχιτεκτονική του μοντέλου είναι ίδια όπως στα προηγούμενα σύνολα δεδομένων, μόνο που το σχήμα εισόδου (input shape) είναι

32×32×3, όπου οι εικόνες έχουν διαστάσεις 32×32 pixels και 3 κανάλια χρώματος (RGB), επειδή είναι έγχρωμες.

Ο αντίστοιχη εντολή κώδικα για το πρώτο συνελικτικό επίπεδο είναι:

model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(32, 32, 3)))

Το σύνολο δεδομένων CIFAR-100

Το σύνολο δεδομένων CIFAR-100 έχει τον ίδιο κώδικα με το CIFAR-10, με μια διαφορά στο επίπεδο εξόδου. Το επίπεδο εξόδου έχει 100 νευρώνες αντί για 10, προκειμένου να γίνει η ταξινόμηση στις 100 κατηγορίες που έχει το CIFAR-100. Η συνάρτηση ενεργοποίησης είναι και πάλι η softmax, ώστε η έξοδος να είναι μια κατανομή πιθανοτήτων για τις 100 κατηγορίες.

Ο αντίστοιχος κώδικας στο επίπεδο εξόδου είναι :

model.add(layers.Dense(100, activation= 'softmax'))

Το σύνολο δεδομένων Fashion MNIST

Ο κώδικας και η διαδικασία εκπαίδευσης για το Fashion MNIST είναι η ίδια με αυτή του MNIST.

Το σύνολο δεδομένων IMDB

Η διαδικασία εισαγωγής και προεπεξεργασίας δεδομένων είναι η ίδια με τα νευρωνικά δίκτυα εμπροσθοδιάδοσης.

Η αρχιτεκτονική του δικτύου περιλαμβάνει ένα επίπεδο που παρεμβάλεται -Embedding- , για τη μετατροπή των δεικτών λέξεων σε πυκνά διανύσματα σταθερού μεγέθους 128. Στη συνέχεια, τρία συνελικτικά επίπεδα (Conv1D), εφαρμόζουν την πράξη της συνέλιξης, με φίλτρα μεγέθους 7 και χρήση της ReLU. Αυτά είναι συνελικτικά επίπεδα ειδικά για μονοδιάστατα δεδομένα, όπως οι

ακολουθίες κειμένου, και ανιχνεύουν μοτίβα στις ακολουθίες. Μετά από κάθε συνελικτικό επίπεδο, εφαρμόζεται η MaxPooling1D για τη μείωση της διάστασης των χαρακτηριστικών και η Dropout, η οποία αφαιρεί με τυχαία δειγματοληψία, το 25% των νευρώνων για τον κίνδυνο της υπερπροσαρμογής (overfitting).

Ακολουθεί το επίπεδο Flatten που μετατρέπει τα δεδομένα σε μονοδιάστατο διάνυσμα για πλήρη σύνδεση των επόμενων επιπέδων. Το πρώτο πλήρως συνδεδεμένο επίπεδο έχει 64 νευρώνες με συνάρτηση ενεργοποίησης την «ReLU» και το επίπεδο εξόδου, το οποίο αποτελείται από έναν νευρώνα με συνάρτηση ενεργοποίησης την «sigmoid», που θεωρείται κατάλληλο για δυαδική ταξινόμηση.

Η συνάρτηση απώλειας είναι η «binary_crossentropy», κατάλληλη για προβλήματα δυαδικής ταξινόμησης. Συμπερασματικά, η αρχιτεκτονική του μοντέλου είναι προσαρμοσμένη για επεξεργασία ακολουθιών κειμένου και εκτέλεση δυαδικής ταξινόμησης.

Ο αντίστοιχος κώδικας είναι :

```
model.add(Embedding(num_words, 128))
# Convolutional layers adapted for 1D input
model.add(Conv1D(32, 7, activation='relu'))
model.add(MaxPooling1D(2))
model.add(Dropout(0.25))
model.add(Conv1D(64, 7, activation='relu'))
model.add(MaxPooling1D(2))
model.add(Dropout(0.25))
model.add(Conv1D(64, 7, activation='relu'))
model.add(Flatten())
```

model.add(Dense(64, activation='relu'))

model.add(Dense(1, activation='sigmoid')

model.compile(optimizer=optimizer, loss='binary_crossentropy', metrics=['accuracy'])

Το σύνολο δεδομένων Reuters

Το σύνολο δεδομένων Reuters, αποτελείται από ειδησεογραφικά κείμενα κατηγοριοποιημένα σε διάφορες θεματικές ενότητες. Η αρχιτεκτονική του μοντέλου είναι η ίδια, έχοντας ένα embedding layer για τη μετατροπή των λέξεων σε πυκνά διανύσματα. Η διαφορά έγκειται στο επίπεδο εξόδου, με αριθμό νευρώνων ίσο με τον αριθμό των κλάσεων του συνόλου δεδομένων. Η συνάρτηση ενεργοποίησης που χρησιμοποιείται στο επίπεδο εξόδου είναι η «softmax», προκειμένου να δημιουργηθεί μια κατανομή πιθανοτήτων για κάθε κατηγορία. Η συνάρτηση απώλειας είναι η sparse_categorical_crossentropy, κατάλληλη για προβλήματα ταξινόμησης σε πολλές κλάσεις. Η εκπαίδευση του μοντέλου πραγματοποιείται με αυτή τη συνάρτηση απώλειας και τον επιλεγμένο βελτιστοποιητή, μετρώντας την ορθότητα κατά την αξιολόγηση της απόδοσής του.

Ο αντίστοιχος κώδικας είναι ο:

model.add(Dense(num_classes, activation='softmax'))

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

Το σύνολο δεδομένων California Housing

Η διαδικασία είναι ίδια με εκείνη που χρησιμοποιείται στα νευρωνικά δίκτυα εμπροσθοδιάδοσης.

Η αρχιτεκτονική του δικτύου είναι παρόμοια με τα προηγούμενα σύνολα δεδομένων, αλλά στο επίπεδο εξόδου υπάρχει ένας μόνο νευρώνας. Επειδή πρόκειται για πρόβλημα παλινδρόμησης, το μοντέλο προβλέπει μια συνεχή τιμή. Για την αξιολόγηση του μοντέλου, χρησιμοποιείται η

συνάρτηση απώλειας του Μέσου Τετραγωνικού Σφάλματος, -Mean Squared Error (MSE)-, κατάλληλη για προβλήματα παλινδρόμησης. Ο αντίστοιχος κώδικας είναι ο εξής:

```
model = Sequential()

model.add(layers.Conv1D(32, 2, activation='relu', input_shape=(X_train.shape[1], 1)))

model.add(layers.MaxPooling1D(2))

model.add(layers.Conv1D(64, 2, activation='relu'))

model.add(layers.MaxPooling1D(2))

model.add(layers.Flatten())

model.add(layers.Dense(64, activation='relu'))

model.add(layers.Dense(1))  # single output for regression

model.compile(optimizer=optimizer, loss='mse')  # Mean Squared Error for regression
```

Μετά την εκπαίδευση κάθε αλγορίθμου, εκτυπώνεται το μέσο τετραγωνικό σφάλμα (mean squared error), ενώ στο τέλος παρουσιάζονται οι γραφικές παραστάσεις του μέσου τετραγωνικού σφάλματος για το σύνολο εκπαίδευσης και επαλήθευσης.

3.4.3. Αναδρομικά Νευρωνικά Δίκτυα (Recurrent Neural Networks)

Το σύνολο δεδομένων MNIST

Εδώ έχουμε το ίδιο μοτίβο που χρησιμοποιείται στα νευρωνικά δίκτυα εμπροσθοδιάδοσης.

Η αρχιτεκτονική του μοντέλου περιέχει αρχικά ένα απλό αναδρομικό επίπεδο εισόδου (SimpleRNN) με 128 νευρώνες και συνάρτηση ενεργοποίησης την «ReLU», που έχει στη συνέχεια ένα πλήρως συνδεδεμένο επίπεδο με 64 νευρώνες και συνάρτηση ενεργοποίηση επίσης την «ReLU». Το επίπεδο εξόδου αποτελείται από 10 νευρώνες, με συνάρτηση ενεργοποίησης εδώ

την «softmax», για την ταξινόμηση σε 10 κατηγορίες. Η συνάρτηση απώλειας που χρησιμοποιείται είναι η «sparse_categorical_crossentropy», κατάλληλη για ταξινόμηση σε πολλές κλάσεις, ενώ η εκπαίδευση γίνεται με βελτιστοποιητή που επιλέγεται. Ο κώδικας είναι ο εξής:

```
model = Sequential()

model.add(SimpleRNN(128, activation='relu',input_shape=(28,28) ,return_sequences=False))

model.add(Dense(64, activation='relu'))

model.add(Dense(10,activation="softmax")


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

Και εδώ, όπως και προηγουμένως, δημιουργείται ένα λεξικό που περιλαμβάνει τους αλγορίθμους βελτιστοποίησης. Κάθε ένας αλγόριθμος εκπαιδεύεται για 25 εποχές, και το 20% του συνόλου εκπαίδευσης, επιλεγμένο τυχαία, να χρησιμοποιείται για επαλήθευση (validation). Μετά την ολοκλήρωση της εκπαίδευσης κάθε αλγόριθμου, εκτυπώνεται η ορθότητα (accuracy) του μοντέλου. Στη συνέχεια, παρουσιάζονται τα ιστογράμματα για τις μετρικές ακρίβεια (precision), ανάκληση (recall), και F1-score, καθώς και οι γραφικές παραστάσεις που απεικονίζουν την ορθότητα και την απώλεια (loss) τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης. Η ίδια διαδικασία ακολουθείται για όλα τα σύνολα δεδομένων και τις αρχιτεκτονικές RNN, εξασφαλίζοντας τη συνέπεια στην αξιολόγηση των μοντέλων.

Το σύνολο δεδομένων CIFAR-10

Η κλήση και η κανονικοποίηση των δεδομένων είναι η ίδια με τα νευρωνικά δίκτυα εμπροσθοδιάδοσης. Για το συγκεκριμένο σύνολο δεδομένων, οι αρχικές εικόνες με διαστάσεις 32×32 pixels και 3 κανάλια χρώματος (RGB) μετασχηματίζονται σε έναν δισδιάστατο πίνακα. Το μέγεθος των εικόνων μετασχηματίζεται σε (32×32, 3), όπου οι διαστάσεις της εικόνας 32×32 συνενώνονται σε έναν πίνακα με 1024 στοιχεία, ενώ τα 3 κανάλια παραμένουν ως έχουν. Οι νέες

διαστάσεις των δεδομένων εκπαίδευσης και δοκιμών γίνονται (αριθμός εικόνων, 1024, 3). Επιπλέον, τα δεδομένα κανονικοποιούνται διαιρώντας τα με το 255 (8-bit εικόνες), ώστε οι τιμές των pixel να βρίσκονται στο εύρος [0, 1]. Ο κώδικας είναι ο εξής:

*X_train = X_train.reshape(X_train.shape[0], 32*32, 3) # Shape: (50000, 1024, 96)*

X_train=X_train/255

*X_test = X_test.reshape(X_test.shape[0], 32*32, 3)   # Shape: (10000, 1024, 96)*

X_test=X_test/255

Η αρχιτεκτονική του μοντέλου είναι παρόμοια με αυτή που χρησιμοποιείται για το σύνολο δεδομένων MNIST, με διαφορά να βρίσκεται στη μορφή εισαγωγής -«input_shape»-. Επειδή οι εικόνες έχουν μετασχηματιστεί σε διαστάσεις (1024, 3) - όπου 1024 αντιπροσωπεύει τον αριθμό των pixel και 3 τα κανάλια χρώματος (RGB) - η είσοδος του αναδρομικού επιπέδου προσαρμόζεται σε αυτή τη μορφή (μορφή εισαγωγής). Το υπόλοιπο μοντέλο διατηρεί την ίδια δομή με το MNIST.

Ο αντίστοιχος κώδικας για το αναδρομικό επίπεδο έχει ως εξής:

model.add(SimpleRNN(128, activation='relu', input_shape=(1024,3) ,return_sequences=False))

Το σύνολο δεδομένων CIFAR-100

Το σύνολο δεδομένων CIFAR-100 έχει ακριβώς τον ίδιο κώδικα με το CIFAR-10, με διαφορά στο επίπεδο εξόδου. Επειδή το CIFAR-100 περιλαμβάνει 100 κατηγορίες, το επίπεδο εξόδου του μοντέλου πρέπει να έχει 100 νευρώνες αντί για 10. Η συνάρτηση ενεργοποίησης παραμένει η «softmax», για να έχουμε μια κατανομή πιθανοτήτων ανάμεσα στις 100 κατηγορίες.

Ο αντίστοιχος κώδικας για το επίπεδο εξόδου είναι ο εξής:

model.add(layers.Dense(100, activation= 'softmax'))

Το σύνολο δεδομένων Fashion MNIST

Ο κώδικας και η διαδικασία εκπαίδευσης για το Fashion MNIST είναι ακριβώς τα ίδια με το MNIST.

Το σύνολο δεδομένων IMDB

Η διαδικασία κλήσης και προεπεξεργασίας είναι η ίδια με τα νευρωνικά δίκτυα εμπροσθοδιάδοσης.

Η αρχιτεκτονική του μοντέλου είναι επίσης παρόμοια με τα προηγούμενα σύνολα δεδομένων. Απλώς, περιλαμβάνει και ένα επίπεδο «Embedding», για τη μετατροπή των εισερχόμενων λέξεων σε πυκνά διανύσματα. Στο επίπεδο εξόδου υπάρχει ένας νευρώνας με συνάρτηση ενεργοποίησης την «sigmoid», κατάλληλος για δυαδική ταξινόμηση, ενώ η συνάρτηση κόστους που χρησιμοποιείται είναι η «binary_crossentropy», κατάλληλη για προβλήματα δυαδικής ταξινόμησης.

Ο κώδικας έχει την εξής μορφή:

```
model=Sequential()
    model.add(Embedding(input_dim=10000, output_dim=128))
    model.add(Flatten())
    model.add(Dense(128,activation="relu"))
    model.add(Dense(64,activation='relu'))
    model.add(Dense(1,activation="sigmoid"))
    model.compile(loss="binary_crossentropy",optimizer=optimizer,metrics=["accuracy"])
```

Το σύνολο δεδομένων Reuters

Η κλήση και προεπεξεργασία των δεδομένων έχει την ίδια δομή με τα νευρωνικά δίκτυα εμπροσθοδιάδοσης.

Η αρχιτεκτονική του μοντέλου είναι παρόμοια απλά, περιλαμβάνει ένα επίπεδο «Embedding», που μετατρέπει τις εισερχόμενες λέξεις σε πυκνά διανύσματα. Εδώ, στο επίπεδο εισόδου χρησιμοποιείται η συνάρτηση ενεργοποίησης «tanh», κατάλληλη για αναδρομικά νευρωνικά δίκτυα. Το επίπεδο εξόδου έχει τόσους νευρώνες όσες οι κλάσεις του προβλήματος, με συνάρτηση ενεργοποίησης την «softmax», που είναι κατάλληλη για ταξινόμηση σε πολλές κατηγορίες. Η συνάρτηση απώλειας που χρησιμοποιείται είναι η «sparse_categorical_crossentropy», κατάλληλη για προβλήματα με πολλές κατηγορίες.

Ο κώδικας είναι ο εξής:

```
model = Sequential()

model.add(Embedding(num_words, 128))

model.add(SimpleRNN(128, activation='tanh', return_sequences=False))

model.add(Dense(64,activation='relu'))

model.add(Dense(num_classes, activation='softmax')


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

Το σύνολο δεδομένων California-Housing

Η κλήση και η προεπεξεργασία δεδομένων είναι ίδια με τις δύο προηγούμενες αρχιτεκτονικές περίπου. Για να είναι δυνατό, να εισαχθούν τα δεδομένα στο αναδρομικό νευρωνικό δίκτυο, τα σύνολα δεδομένων X_train και X_test πρέπει να αναδιαμορφωθούν σε τρισδιάστατους πίνακες.

Η πρώτη διάσταση θα είναι ο αριθμός των δειγμάτων, η δεύτερη το χρονικό βήμα (timesteps), που εδώ είναι 1, και η τρίτη διάσταση το πλήθος των κατηγοριών για κάθε δείγμα.

Ο αντίστοιχος κώδικας έχει ως εξής:

timesteps = 1

num_features = X_train.shape[1]

 # Reshape to (samples, timesteps, features)

X_train = X_train.reshape((X_train.shape[0], timesteps, num_features))

X_test = X_test.reshape((X_test.shape[0], timesteps, num_features))

Η αρχιτεκτονική του δικτύου παραμένει παρόμοια. Το αναδρομικό επίπεδο (SimpleRNN) λαμβάνει ως είσοδο δεδομένα με μορφή (timesteps, num_features), δηλαδή, η είσοδος του δικτύου περιλαμβάνει μια ακολουθία χρονικών βημάτων με πολλαπλά χαρακτηριστικά. Στο επίπεδο εξόδου, υπάρχει ένας μόνο νευρώνας χωρίς καμία συνάρτηση ενεργοποίησης, επειδή είναι παλινδρόμηση και προβλέπει μια συνεχή τιμή. Η συνάρτηση απώλειας που χρησιμοποιείται είναι η Mean Squared Error (MSE), κατάλληλη για προβλήματα παλινδρόμησης, επειδή μετρά τη μέση διαφορά μεταξύ των προβλεπόμενων τιμών και των πραγματικών δεδομένων. Ο αντίστοιχος κώδικας είναι:

model = Sequential()

model.add(SimpleRNN(128,input_shape=(timesteps,,num_features)activation='relu',return_sequences=False))

model.add(Dense(64,activation='relu'))

model.add(Dense(1,activation="sigmoid"))

model.compile(optimizer=optimizer, loss='mse') # Mean Squared Error for regression

Μετά την εκπαίδευση κάθε αλγορίθμου, εκτυπώνεται το Μέσο Τετραγωνικό Σφάλμα (mean squared error), ενώ στο τέλος εκτυπώνονται οι γραφικές παραστάσεις του μέσου τετραγωνικού σφάλματος για το σύνολο εκπαίδευσης και επαλήθευσης.

4. Υπολογιστική Μελέτη

4.1. Νευρωνικά Δίκτυα Εμπροσθοδιάδοσης (Feed Forward Neural Networks)

Σύνολο δεδομένων MNIST

Στο σύνολο δεδομένων MNIST, παρατηρούμε ότι οι περισσότεροι αλγόριθμοι βελτιστοποίησης επιτυγχάνουν υψηλά επίπεδα ορθότητας, συνήθως πάνω από 88%. Συγκεκριμένα, ο RMSprop φτάνει σε ορθότητα 98.06%, ο SGD 93.03%, ο Adam 97.93%, ο AdamW 97.72%, ο Adadelta 84.42%, ο Adagrad 93.33%, ο Adamax 97.8%, ο Adafactor 9.8%, ο Nadam 97.73%, ο FTRL 11.35%, και ο Lion 86.43%.

Κατά τη διάρκεια της εκπαίδευσης των μοντέλων για 25 εποχές, παρατηρείται σταδιακή αύξηση της ορθότητας και μείωση της απώλειας, τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης, υποδεικνύοντας βελτίωση στην απόδοση των περισσότερων αλγορίθμων.

Στους αλγόριθμους RMSprop, Adam και AdamW, παρατηρείται αύξηση της ορθότητας με κάποιες διακυμάνσεις, ενώ η απώλεια μειώνεται αντίστοιχα με διακυμάνσεις και στα δύο σύνολα. Ωστόσο, στο σύνολο επαλήθευσης η απώλεια εμφανίζει αυξομειώσεις μετά τις πρώτες εποχές, υποδηλώνοντας μια σχετική αστάθεια στη γενίκευση των μοντέλων σε νέα δεδομένα.

Οι αλγόριθμοι SGD και Adagrad παρουσιάζουν πιο ομαλή αύξηση της ορθότητας και στα δύο σύνολα και μείωση της απώλειας με λιγότερες διακυμάνσεις, γεγονός που υποδεικνύει καλή προσαρμογή και σταθερότητα κατά τη γενίκευση σε άγνωστα δεδομένα.

Ο Adadelta παρουσιάζει επίσης σταθερή βελτίωση της ορθότητας και μείωση της απώλειας, παρόλο που η συνολική του απόδοση είναι χαμηλότερη από τους άλλους αλγόριθμους.

Ο Adamax επιδεικνύει ομαλή αύξηση της ορθότητας στο σύνολο εκπαίδευσης, αλλά παρουσιάζει αυξομειώσεις στο σύνολο επαλήθευσης, κάτι που υποδηλώνει αστάθεια στην προσαρμογή σε νέα δεδομένα.

Οι Adafactor και FTRL αντιμετωπίζουν σημαντικά προβλήματα, καθώς εμφανίζουν χαμηλή ορθότητα και σταθερές NaN τιμές στην απώλεια, υποδεικνύοντας ότι δεν μπορούν να εκπαιδευτούν αποτελεσματικά.

Ο Nadam παρουσιάζει αύξηση της ορθότητας με διακυμάνσεις και στα δύο σύνολα, με τη μείωση της απώλειας να είναι εμφανής στο σύνολο εκπαίδευσης, αλλά με αστάθεια στο σύνολο επαλήθευσης, γεγονός που δυσκολεύει τη γενίκευση σε νέα δεδομένα.

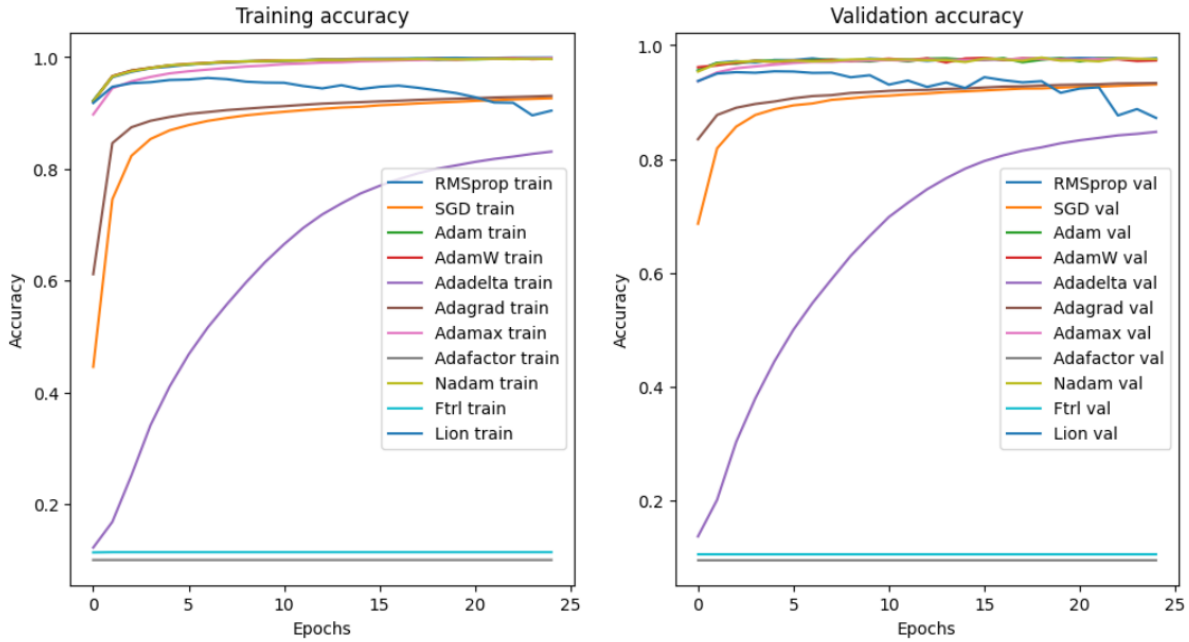
Ο Lion, παρά την αρχικά καλή του απόδοση, εμφανίζει τη μεγαλύτερη αστάθεια, με έντονες αυξομειώσεις τόσο στην ορθότητα όσο και στην απώλεια, καταλήγοντας σε χαμηλότερη απόδοση σε σχέση με άλλους αλγορίθμους.

Γενικά, οι αλγόριθμοι RMSprop, Adam, AdamW, Adamax, και Nadam καταγράφουν τις υψηλότερες επιδόσεις στο σύνολο δεδομένων MNIST, με ακρίβειες πάνω από 97%. Παρά την υψηλή απόδοση, παρουσιάζουν κάποια αστάθεια κατά την προσαρμογή σε νέα δεδομένα, όπως φαίνεται από τις διακυμάνσεις στην απώλεια και την ορθότητα στο σύνολο επαλήθευσης.

Από την άλλη, οι SGD, Adagrad, και Adadelta, αν και έχουν χαμηλότερες επιδόσεις, παρουσιάζουν καλύτερη σταθερότητα και γενίκευση σε άγνωστα δεδομένα, καθιστώντας τους πιο αξιόπιστους για προβλήματα που απαιτούν σταθερή συμπεριφορά.

Ο Lion εμφανίζει τη μεγαλύτερη αστάθεια, με έντονες αυξομειώσεις στην ορθότητα και την απώλεια, γεγονός που υποβαθμίζει την απόδοσή του σε σύγκριση με άλλους αλγόριθμους.

Οι FTRL και Adafactor καταγράφουν τις χαμηλότερες επιδόσεις, εμφανίζοντας χαμηλή ορθότητα και δυσκολία στη βελτίωση, τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης.

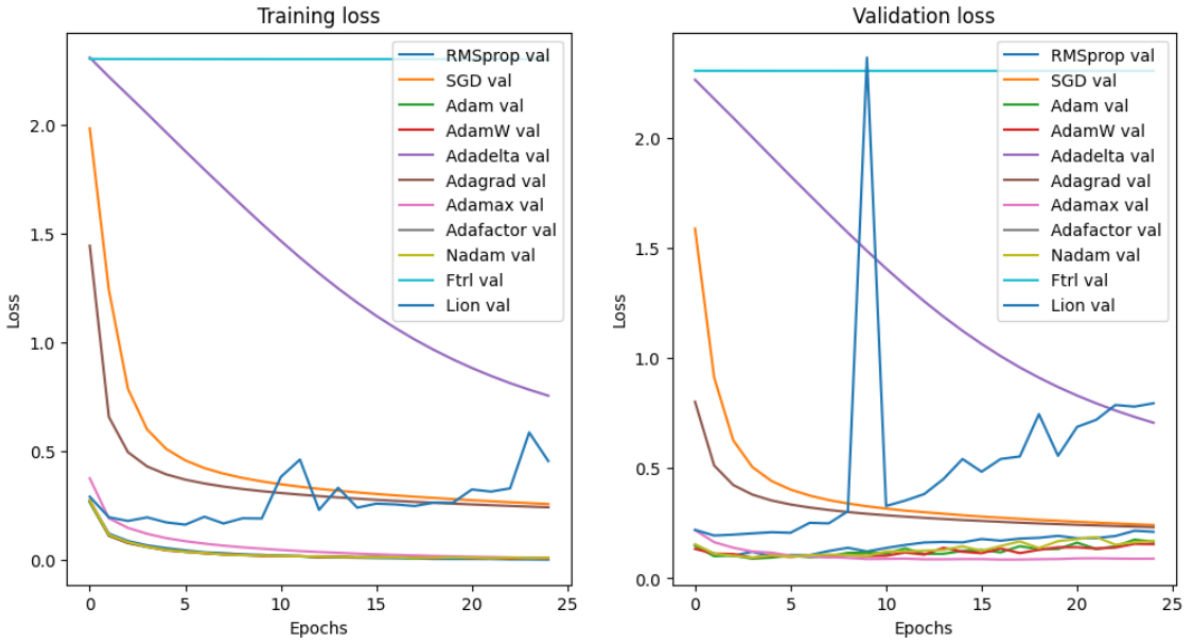


Εικόνα 1-MNIST Accuracy

Οι γραφικές παραστάσεις για την ορθότητα στο σύνολο εκπαίδευσης και επαλήθευσης δείχνουν ότι οι περισσότεροι αλγόριθμοι ακολουθούν μια λογαριθμική πορεία αύξησης, υποδεικνύοντας σταθερή βελτίωση στην απόδοση των μοντέλων.

Ωστόσο, οι αλγόριθμοι Ftrl και Adafactor διαφέρουν, καθώς οι καμπύλες τους παραμένουν σχεδόν παράλληλες με τον άξονα των εποχών, δείχνοντας καθόλου βελτίωση.

Ο Lion, αντίθετα με τους υπόλοιπους, είναι ο μοναδικός αλγόριθμος που παρουσιάζει μείωση στην ορθότητα, γεγονός που υποδηλώνει ότι η απόδοσή του επιδεινώνεται κατά τη διάρκεια της εκπαίδευσης.



Εικόνα 2-MNIST Loss

Στη γραφική παράσταση για την απώλεια του συνόλου εκπαίδευσης, ο αλγόριθμος Ftrl εμφανίζει μια σχεδόν σταθερή γραμμή, παράλληλη με την αρχή των αξόνων, υποδηλώνοντας ελάχιστη βελτίωση στην απόδοση του μοντέλου.

Ο Adadelta παρουσιάζει μια σχεδόν γραμμική μείωση της απώλειας, καταγράφοντας σταθερή πρόοδο.

Ο Lion, αντίθετα, παρουσιάζει μια αυξητική πορεία, γεγονός που δείχνει χειροτέρευση της απόδοσης με την πάροδο των εποχών.

Οι υπόλοιποι αλγόριθμοι, όπως οι Adam, AdamW και RMSprop, παρουσιάζουν εκθετική μείωση της απώλειας και σταδιακά σταθεροποιούνται, με τις καμπύλες τους να γίνονται παράλληλες με τον άξονα των εποχών.

Στη γραφική παράσταση της απώλειας για το σύνολο επαλήθευσης, οι αλγόριθμοι SGD, Adadelta, Adagrad, Adamax και Ftrl ακολουθούν μια παρόμοια πορεία με αυτή της εκπαίδευσης, καταγράφοντας σταθερή μείωση της απώλειας.

Ο Lion παρουσιάζει μια σημαντική αύξηση στην απώλεια, φτάνοντας σε ένα έντονο πικ κοντά στη 10η εποχή, υποδηλώνοντας ασταθή απόδοση.

Οι υπόλοιποι αλγόριθμοι, όπως ο Adam και ο AdamW, επίσης εμφανίζουν μικρή αύξηση της απώλειας.

Συμπερασματικά, ο RMSprop φαίνεται να είναι ο πιο αποδοτικός αλγόριθμος για το MNIST, επιτυγχάνοντας ορθότητα 98.06%, ελαφρώς υψηλότερη από τον Adam, ο οποίος φτάνει στο 97.93%. Παρά την κοντινή τους απόδοση, τα αποτελέσματα υποδεικνύουν ότι ο RMSprop διαχειρίζεται καλύτερα τις προκλήσεις του MNIST και έχει καλύτερη σταθερότητα στη γενίκευση, καθιστώντας τον μια εξαιρετική επιλογή για ταξινομήσεις εικόνων. Ωστόσο, και οι δύο αλγόριθμοι παραμένουν αξιόπιστες επιλογές με μικρές διαφορές στην απόδοση.

Σύνολο δεδομένων CIFAR-10

Σε αντίθεση με το MNIST dataset, το CIFAR-10 παρουσιάζει χαμηλότερες επιδόσεις με ακρίβειες κάτω από 50%. Συγκεκριμένα, ο RMSprop επιτυγχάνει ορθότητα 47.6%, ο SGD 44.71%, ο Adam 46.63%, ο AdamW 47.24%, ο Adadelta 34.46%, ο Adagrad 43.59%, ο Adamax 49.22%, ο Adafactor 10%, ο Nadam 48.21%, ο FTRL 10%, και ο Lion 37.42%.

Κατά την εκπαίδευση των μοντέλων σε 25 εποχές, παρατηρείται σταδιακή αύξηση της ορθότητας και μείωση της απώλειας για τους περισσότερους αλγόριθμους, τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης, υποδεικνύοντας βελτίωση της απόδοσης.

Παρόλα αυτά, αλγόριθμοι όπως οι RMSprop, Adam, Adagrad, Adamax και Nadam παρουσιάζουν διακυμάνσεις, με την ορθότητα και την απώλεια να κυμαίνονται, αν και η γενική τάση παραμένει θετική, με αύξηση της ορθότητας και μείωση της απώλειας.

Ο αλγόριθμος SDG εμφανίζει επίσης αυξομειώσεις στην ορθότητα τόσο στο σύνολο εκπαίδευσης όσο και στο επαλήθευσης, με την απώλεια να μειώνεται παρά τις διακυμάνσεις.

Αντίθετα, ο αλγόριθμος AdamW παρουσιάζει πιο σταθερή βελτίωση, με μικρές διακυμάνσεις στην ορθότητα και ομαλή μείωση της απώλειας και στα δύο σύνολα.

Ο αλγόριθμος Adadelta δείχνει διακυμάνσεις στην ορθότητα στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης η ορθότητα αυξάνεται και η απώλεια μειώνεται με σταθερό ρυθμό.

Από την άλλη πλευρά, ο αλγόριθμος Adafactor εμφανίζει μείωση της ορθότητας με διακυμάνσεις στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης η ορθότητα παραμένει σταθερή, με την απώλεια να καταγράφεται ως NaN και στα δύο σύνολα.

Στον αλγόριθμο Ftrl, η ορθότητα στο σύνολο εκπαίδευσης παρουσιάζει διακυμάνσεις χωρίς συνολική βελτίωση, ενώ στο σύνολο επαλήθευσης μειώνεται και σταθεροποιείται. Η απώλεια σταθεροποιείται επίσης και στα δύο σύνολα.

Ο αλγόριθμος Lion παρουσιάζει διακυμάνσεις στην ορθότητα και την απώλεια, με την ορθότητα να αυξάνεται και την απώλεια να μειώνεται.

Ωστόσο, οι δείκτες precision, recall και F1-score δείχνουν ότι οι περισσότερες κλάσεις έχουν τιμές κοντά στο 0.4, κάτι που υποδηλώνει μέτρια απόδοση στη διάκριση των κλάσεων.

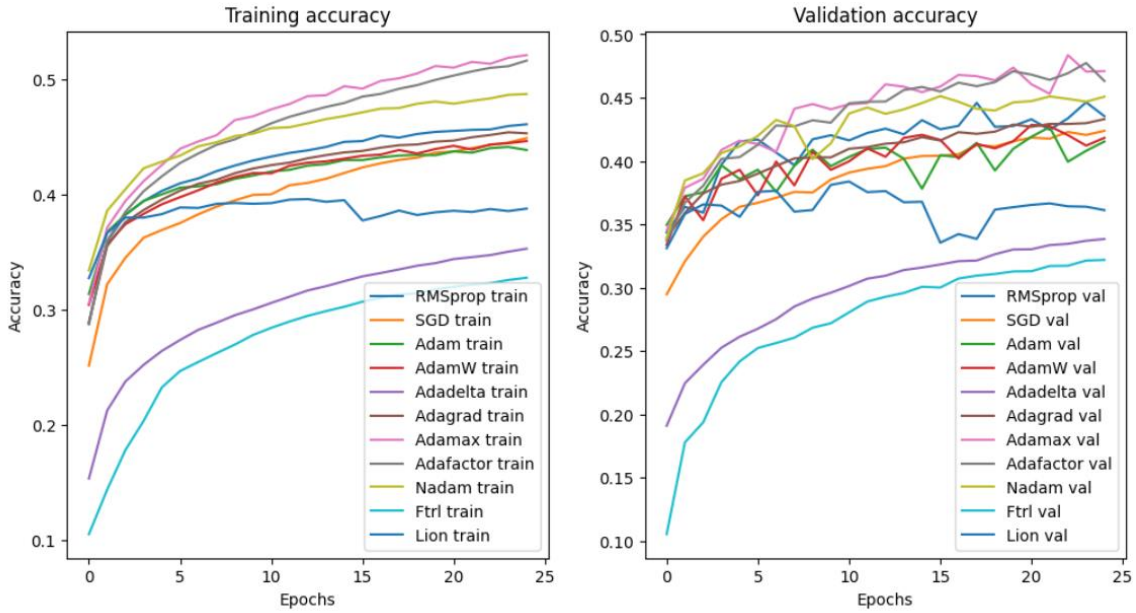
Εξαίρεση αποτελούν οι αλγόριθμοι Adafactor και Ftrl, οι οποίοι αναγνωρίζουν μόνο μία κλάση, επισημαίνοντας κακή απόδοση στην ταξινόμηση.

Γενικά, οι αλγόριθμοι RMSprop, Adam, AdamW, Adamax, και Nadam καταγράφουν τις υψηλότερες επιδόσεις στο σύνολο δεδομένων CIFAR-10, με ακρίβειες κάτω από 50%. Παρά τις σχετικά καλές αποδόσεις, παρουσιάζουν κάποιες διακυμάνσεις στην απώλεια και την ορθότητα στο σύνολο επαλήθευσης, υποδεικνύοντας αστάθεια κατά την προσαρμογή σε νέα δεδομένα. Από την άλλη, οι SGD, Adagrad και Adadelta, αν και έχουν χαμηλότερες επιδόσεις, επιδεικνύουν

καλύτερη σταθερότητα και ικανότητα γενίκευσης σε άγνωστα δεδομένα, καθιστώντας τους πιο αξιόπιστους για προβλήματα που απαιτούν σταθερή συμπεριφορά.

Ο Lion εμφανίζει τη μεγαλύτερη αστάθεια, με έντονες αυξομειώσεις στην ορθότητα και την απώλεια, γεγονός που υποβαθμίζει την απόδοσή του σε σύγκριση με άλλους αλγόριθμους.

Οι FTRL και Adafactor καταγράφουν τις χαμηλότερες επιδόσεις, εμφανίζοντας πολύ χαμηλή ορθότητα και δυσκολία στη βελτίωση, τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης, γεγονός που δείχνει αδυναμία προσαρμογής στο συγκεκριμένο σύνολο δεδομένων.

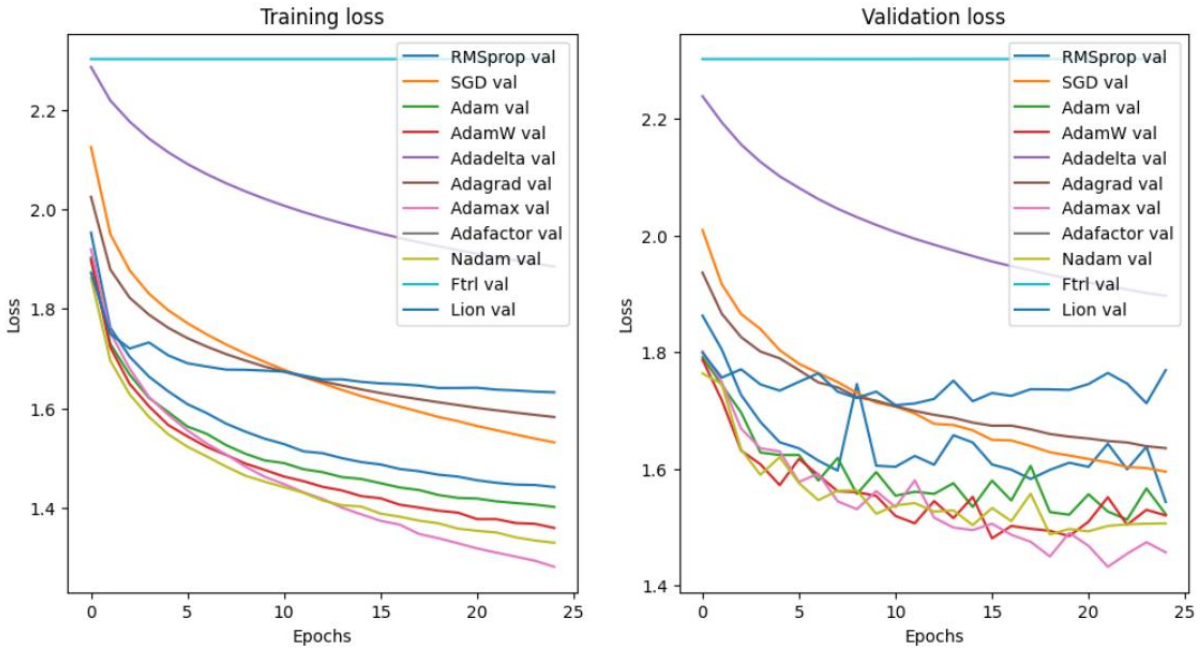


Εικόνα 3- CIFAR-10 Accuracy

Οι γραφικές παραστάσεις για την ορθότητα τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης δείχνουν ότι οι περισσότεροι αλγόριθμοι ακολουθούν μια λογαριθμική πορεία αύξησης, με σταδιακή βελτίωση της απόδοσής τους.

Ειδικότερα, οι αλγόριθμοι όπως οι Adam, AdamW, Lion και άλλοι παρουσιάζουν συνεχή βελτίωση στην ορθότητα, ενώ οι αλγόριθμοι Adafactor και FTRL ξεχωρίζουν, καθώς οι καμπύλες τους παραμένουν σχεδόν παράλληλες προς τον άξονα των εποχών, υποδεικνύοντας ελάχιστη ή καθόλου βελτίωση.

Στο σύνολο επαλήθευσης, παρατηρείται παρόμοια συμπεριφορά, με πιο έντονες διακυμάνσεις σε σύγκριση με το σύνολο εκπαίδευσης, υποδηλώνοντας ότι η ορθότητα σε ορισμένους αλγόριθμους είναι λιγότερο σταθερή κατά τη διάρκεια των επαναλήψεων.



Εικόνα 4- CIFAR-10 Loss

Οι γραφικές παραστάσεις για την απώλεια στο σύνολο εκπαίδευσης και επαλήθευσης δείχνουν παρόμοιες τάσεις με αυτές της ορθότητας.

Στο σύνολο επαλήθευσης, η μείωση της απώλειας είναι γενικά εκθετική, αν και εμφανίζονται πολλές διακυμάνσεις.

Αλγόριθμοι όπως οι SGD, Adagrad και Adadelta επιδεικνύουν μεγαλύτερη σταθερότητα κατά τη διάρκεια της εκπαίδευσης, με λιγότερες αυξομειώσεις στην απώλεια, γεγονός που υποδηλώνει πιο ομαλή απόδοση.

Αντίθετα, ο αλγόριθμος FTRL παραμένει σχεδόν παράλληλος με τον άξονα των εποχών, υποδεικνύοντας ότι η απώλειά του δεν μειώνεται αισθητά.

Τέλος, ο αλγόριθμος Lion είναι ο μόνος που παρουσιάζει αύξηση στην απώλεια, κάτι που φανερώνει πιθανή επιδείνωση της απόδοσής του κατά την επαλήθευση.

Συμπερασματικά, ο Adamax φαίνεται να είναι ο πιο αποδοτικός αλγόριθμος για το CIFAR-10, επιτυγχάνοντας ορθότητα 49.22%, ελαφρώς υψηλότερη από τον Nadam που φτάνει το 48.21%. Παρόλο που αυτές οι ακρίβειες δεν είναι ιδιαίτερα υψηλές σε σύγκριση με άλλες προσεγγίσεις ή σύνολα δεδομένων, τα αποτελέσματα δείχνουν ότι ο Adamax αντιμετωπίζει καλύτερα τις προκλήσεις του CIFAR-10 σε σύγκριση με τους άλλους αλγόριθμους που εξετάστηκαν.

Σύνολο δεδομένων CIFAR-100

Οι επιδόσεις στο CIFAR-100 είναι γενικά χαμηλές, με όλες τις μεθόδους να επιτυγχάνουν ακρίβειες κάτω από το 20%. Ο αλγόριθμος RMSprop καταγράφει ορθότητα 15.9%, ενώ ο SGD επιτυγχάνει 12.16%. Ο Adam παρουσιάζει επίσης χαμηλή απόδοση, με ορθότητα 16.4%, ενώ ο AdamW σημαίνει ελαφρώς καλύτερη απόδοση με 19.99%. Ο Adadelta εμφανίζει τη χαμηλότερη επίδοση, με μόλις 2.99%. Ο Adagrad φτάνει σε ορθότητα 11.88%, ενώ ο Adamax πετυχαίνει 20.29%, το υψηλότερο ποσοστό από όλους τους αλγόριθμους που εξετάστηκαν. Ο Adafactor έχει εξαιρετικά χαμηλή απόδοση με μόλις 1%, όπως και ο FTRL, ενώ ο Nadam καταφέρνει μια πιο αξιοσημείωτη απόδοση με ορθότητα 17.46%. Ο αλγόριθμος Lion παρουσιάζει μια από τις χαμηλότερες επιδόσεις, με ορθότητα μόλις 7.73%. Τα αποτελέσματα αυτά δείχνουν ότι οι αλγόριθμοι βελτιστοποίησης έχουν περιορισμένη αποτελεσματικότητα στο CIFAR-100, καθιστώντας την αντιμετώπιση του συγκεκριμένου συνόλου δεδομένων πιο απαιτητική.

Κατά την εκπαίδευση των μοντέλων για 25 εποχές, παρατηρούνται διακυμάνσεις στην ορθότητα και την απώλεια στα σύνολα εκπαίδευσης και επαλήθευσης με τους αλγόριθμους RMSprop, Adam και AdamW, αν και γενικά παρατηρείται μείωση της απώλειας και αύξηση της ορθότητας.

Ο αλγόριθμος SGD παρουσιάζει ομαλή αύξηση της ορθότητας στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης η ορθότητα παρουσιάζει αυξομειώσεις, με τη συνολική τάση να είναι ανοδική, και η απώλεια μειώνεται ομαλά και στα δύο σύνολα.

Οι αλγόριθμοι Adadelta και Adagrad εμφανίζουν αυξομειώσεις στην ορθότητα του συνόλου εκπαίδευσης και επαλήθευσης, ενώ η απώλεια μειώνεται σταθερά και στα δύο σύνολα.

Ο Adamax παρουσιάζει επίσης ομαλή αύξηση της ορθότητας στο σύνολο εκπαίδευσης, με μικρές διακυμάνσεις στην ορθότητα του συνόλου επαλήθευσης, ενώ η απώλεια μειώνεται σταθερά στο σύνολο εκπαίδευσης και με διακυμάνσεις στο σύνολο επαλήθευσης.

Οι αλγόριθμοι Adafactor και Ftrl εμφανίζουν αυξομειώσεις στην ορθότητα στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης η ορθότητα παραμένει σταθερή, με την απώλεια να παρουσιάζει τιμές NaN και στα δύο σύνολα. Ο αλγόριθμος Nadam παρουσιάζει ομαλή αύξηση της ορθότητας στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης η ορθότητα αυξάνεται με διακυμάνσεις και η απώλεια μειώνεται με διακυμάνσεις και στα δύο σύνολα.

Ωστόσο, ο αλγόριθμος Lion αντιμετωπίζει σοβαρά προβλήματα, καθώς η απώλεια μειώνεται με διακυμάνσεις στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης αυξάνεται ελαφρώς. Η ορθότητα στο σύνολο εκπαίδευσης αυξάνεται με διακυμάνσεις, ενώ στο σύνολο επαλήθευσης μειώνεται σταδιακά μέχρι να σταθεροποιηθεί.

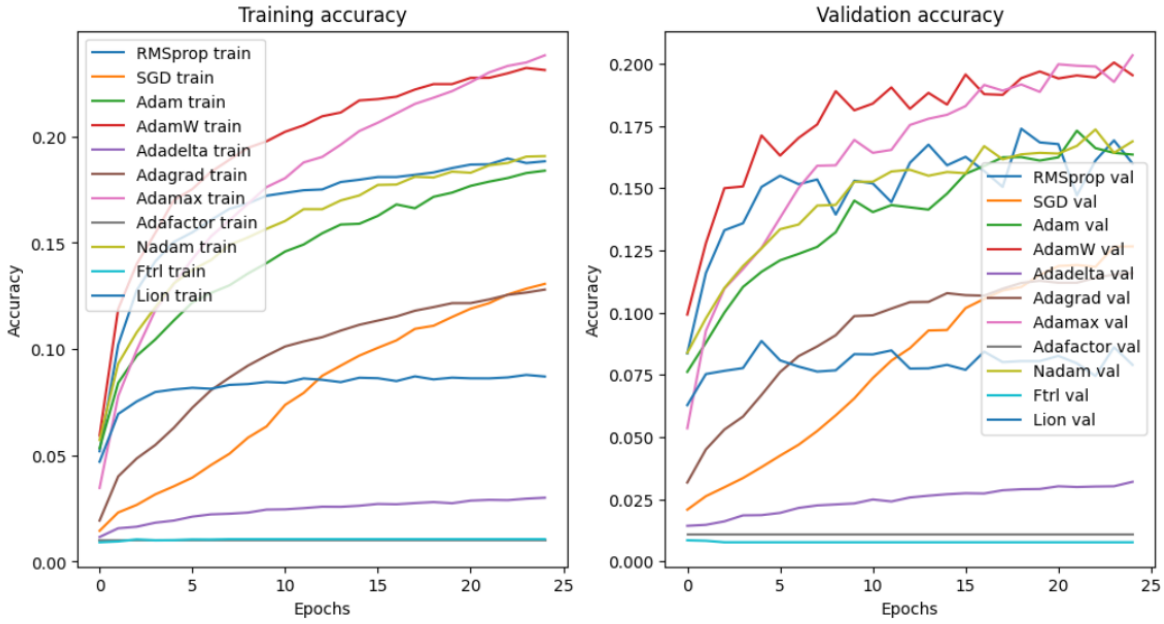
Στα διαγράμματα precision, recall και F1-score, οι περισσότεροι αλγόριθμοι, όπως RMSprop, SGD, Adam, AdamW, Adamax, Adagrad, Nadam και Lion, καταγράφουν χαμηλές επιδόσεις, με την πλειοψηφία των κλάσεων να βρίσκεται κοντά στο 0.2. Ο αλγόριθμος Adafactor και ο Ftrl αναγνωρίζουν μόνο μία κλάση, ενώ ο Adadelta αναγνωρίζει μόνο ορισμένες κλάσεις.

Γενικά, οι αλγόριθμοι Adamax, AdamW, Nadam, Adam και RMSprop καταγράφουν τις υψηλότερες επιδόσεις στο σύνολο δεδομένων CIFAR-100, με ακρίβειες κάτω από 20%. Παρά τις σχετικά καλές αποδόσεις τους, παρουσιάζουν κάποιες διακυμάνσεις στην απώλεια και την ορθότητα στο σύνολο επαλήθευσης, υποδεικνύοντας αστάθεια κατά την προσαρμογή σε νέα δεδομένα.

Από την άλλη, οι SGD, Adagrad και Adadelta, αν και έχουν χαμηλότερες επιδόσεις, επιδεικνύουν καλύτερη σταθερότητα και ικανότητα γενίκευσης σε άγνωστα δεδομένα, καθιστώντας τους πιο αξιόπιστους για προβλήματα που απαιτούν σταθερή συμπεριφορά.

Ο Lion εμφανίζει τη μεγαλύτερη αστάθεια, με έντονες αυξομειώσεις στην ορθότητα και την απώλεια, γεγονός που υποβαθμίζει την απόδοσή του σε σύγκριση με τους υπόλοιπους αλγόριθμους.

Οι FTRL και Adafactor καταγράφουν τις χαμηλότερες επιδόσεις, εμφανίζοντας πολύ χαμηλή ορθότητα και δυσκολία στη βελτίωση, τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης, γεγονός που δείχνει αδυναμία προσαρμογής στο συγκεκριμένο σύνολο δεδομένων.



Εικόνα 5- CIFAR-100 Accuracy

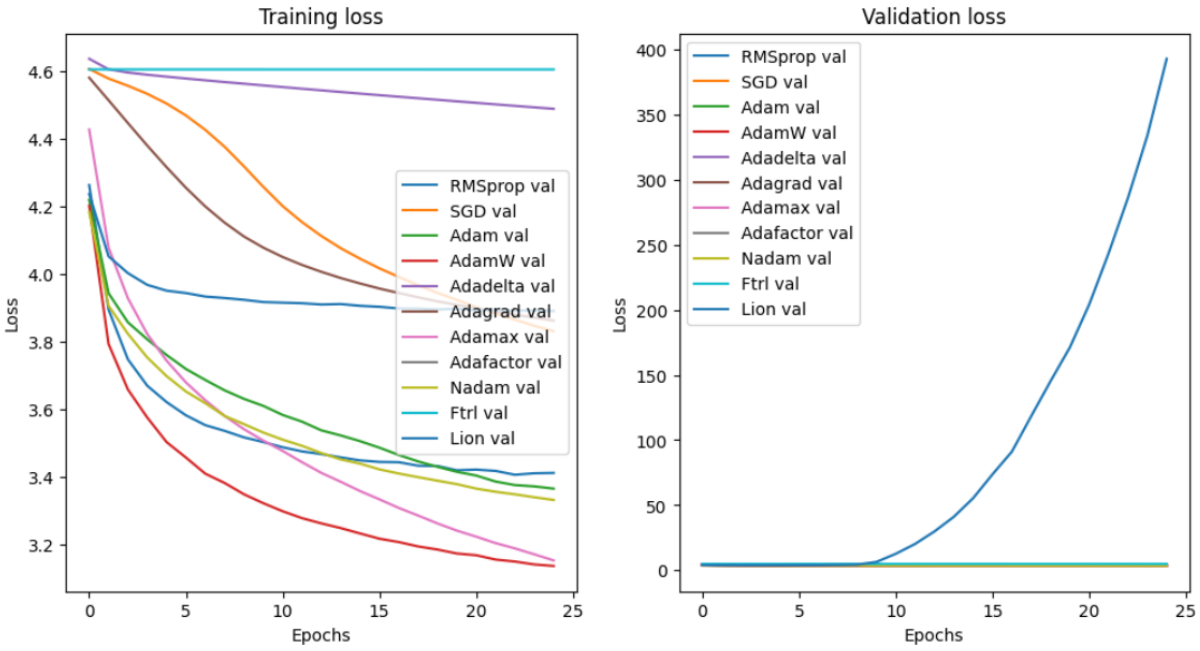
Η γραφική παράσταση της ορθότητας για το σύνολο εκπαίδευσης δείχνει ότι οι περισσότεροι αλγόριθμοι παρουσιάζουν λογαριθμική αύξηση της ορθότητας καθώς προχωρούν οι εποχές. Ωστόσο, οι αλγόριθμοι FTRL και Adafactor εμφανίζουν σχεδόν σταθερή απόδοση, με τις

καμπύλες τους να είναι σχεδόν παράλληλες προς τον άξονα των εποχών, γεγονός που υποδηλώνει ελάχιστη ή μηδενική βελτίωση.

Ο αλγόριθμος Adadelta παρουσιάζει βελτίωση με σταθερό ρυθμό, ακολουθώντας καμπύλη της μορφής $y = ax + b$.

Οι αλγόριθμοι SGD και Adagrad επίσης εμφανίζουν πιο αργή αύξηση σε σύγκριση με τους υπόλοιπους, ενώ ο Lion, παρόλο που αρχικά παρουσιάζει κάποια βελτίωση, στη συνέχεια σταθεροποιείται, με την καμπύλη του να γίνεται σχεδόν παράλληλη προς τον άξονα των εποχών, υποδεικνύοντας στασιμότητα.

Η γραφική παράσταση της ορθότητας για το σύνολο επαλήθευσης ακολουθεί μια παρόμοια πορεία με αυτή του συνόλου εκπαίδευσης, ωστόσο παρατηρούνται περισσότερες αυξομειώσεις. Αυτές οι διακυμάνσεις υποδηλώνουν μεγαλύτερη αστάθεια στη γενίκευση των μοντέλων, υποδεικνύοντας ότι τα μοντέλα δυσκολεύονται να επιτύχουν σταθερά καλή απόδοση σε δεδομένα εκτός του συνόλου εκπαίδευσης. Οι αυξομειώσεις αυτές καταδεικνύουν ότι η διαδικασία εκπαίδευσης μπορεί να μην διασφαλίζει συνεχή βελτίωση, ειδικά όταν αντιμετωπίζει πιο σύνθετα δεδομένα επαλήθευσης, οδηγώντας σε αστάθεια στην απόδοση των μοντέλων.



Εικόνα 6 - CIFAR-100 Loss

Στη γραφική παράσταση της απώλειας κατά την εκπαίδευση, παρατηρείται ότι οι περισσότεροι αλγόριθμοι ακολουθούν μια λογαριθμική μείωση της απώλειας καθώς προχωρούν οι εποχές.

Ο αλγόριθμος FTRL παραμένει σχεδόν σταθερός, με την καμπύλη του να είναι παράλληλη προς τον άξονα των εποχών.

Ο Adadelta, ενώ αρχικά παρουσιάζει λογαριθμική μείωση, στη συνέχεια η καμπύλη του γίνεται ευθεία με μικρή καθοδική κλίση, υποδεικνύοντας επιτάχυνση της μείωσης της απώλειας.

Ο SGD παρουσιάζει μια σταδιακή, σιγμοειδή μείωση, ενώ ο Adagrad φαίνεται να μειώνει την απώλεια πιο αργά σε σχέση με τους υπόλοιπους αλγορίθμους. Ο Lion, αν και αρχικά εμφανίζει λογαριθμική μείωση, τελικά σταθεροποιείται και η καμπύλη του γίνεται παράλληλη προς τον άξονα των εποχών.

Στη γραφική παράσταση της απώλειας στο σύνολο επαλήθευσης, παρατηρούνται μόνο οι καμπύλες των FTRL και Nadam, οι οποίες είναι σχεδόν παράλληλες προς τον άξονα των εποχών, ενώ η καμπύλη του Lion αυξάνεται εκθετικά, φτάνοντας σε πολύ υψηλές τιμές απώλειας. Αυτή η

απότομη αύξηση της απώλειας για τον Lion οδηγεί σε τόσο μεγάλες τιμές που οι καμπύλες των υπόλοιπων αλγορίθμων δεν είναι ευδιάκριτες στη γραφική παράσταση, καθώς η απώλεια του Lion κυριαρχεί. Αυτό το φαινόμενο υποδηλώνει ότι ο Lion αντιμετωπίζει σοβαρά προβλήματα γενίκευσης κατά την επαλήθευση.

Συμπερασματικά, ο Adamax φαίνεται να είναι ο πιο αποδοτικός αλγόριθμος για το CIFAR-100, επιτυγχάνοντας ορθότητα 20.29%, ελαφρώς υψηλότερη από τον AdamW που φτάνει το 19.99%. Παρόλο που αυτές οι ακρίβειες είναι χαμηλές σε σχέση με άλλες προσεγγίσεις και δεδομένα, οι συγκεκριμένες τιμές φαίνεται να υποδεικνύουν ότι ο Adamax αντιμετωπίζει καλύτερα τις προκλήσεις του CIFAR-100 σε σύγκριση με τους άλλους αλγόριθμους που δοκιμάστηκαν.

Σύνολο δεδομένων Fashion MNIST

Στο σύνολο δεδομένων Fashion MNIST, οι περισσότεροι αλγόριθμοι επιτυγχάνουν ορθότητα άνω του 70%. Ο αλγόριθμος Nadam πετυχαίνει την υψηλότερη ορθότητα με 88,7%, ακολουθούμενος από τον Adamax με 88,62%, τον Adam με 88,55%, τον AdamW με 88,37% και τον RMSprop με 87,42%. Ο αλγόριθμος Lion φτάνει στο 84,04%, ο Adagrad στο 83,58% και ο SGD στο 83,35%. Η ορθότητα του Adadelta φτάνει στο 74,54%, ενώ ο FTRL καταγράφει ορθότητα μόλις 10%, παρόμοια με τον Adafactor, ο οποίος παρουσιάζει σημαντικά χαμηλότερη απόδοση με μόλις 10%.

Η εκπαίδευση του μοντέλου για 25 εποχές δείχνει ότι οι περισσότεροι αλγόριθμοι βελτιστοποίησης παρουσιάζουν μείωση της απώλειας στο σύνολο εκπαίδευσης, αλλά αύξηση στο σύνολο επαλήθευσης, ενώ η ορθότητα αυξάνεται σταθερά.

Οι αλγόριθμοι RMSprop, Adam, AdamW, Adamax και Nadam εμφανίζουν διακυμάνσεις στην απώλεια τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης, με συνολική τάση αύξησης της ορθότητας και μείωσης της απώλειας.

Ο SGD και ο Adagrad παρουσιάζουν σταδιακή αύξηση της ορθότητας με κάποιες διακυμάνσεις και στα δύο σύνολα, ενώ η απώλεια μειώνεται με διακυμάνσεις στο σύνολο εκπαίδευσης και ομαλότερα στο σύνολο επαλήθευσης.

Ο Adadelta βελτιώνει σταθερά την ορθότητα τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης, ενώ η απώλεια μειώνεται ομαλά και στα δύο σύνολα.

Αντίθετα, ο Adafactor έχει χαμηλή απόδοση, καθώς αναγνωρίζει μόνο μία κλάση. Η ακρίβειά του αυξάνεται ελαφρώς με διακυμάνσεις στο σύνολο εκπαίδευσης, αλλά παραμένει σταθερή στο σύνολο επαλήθευσης, ενώ η απώλεια είναι NaN και στα δύο σύνολα.

Ο FTRL εμφανίζει αυξομειώσεις στην ορθότητα στο σύνολο εκπαίδευσης και σταθερή απόδοση στο σύνολο επαλήθευσης. Η απώλεια του παραμένει σταθερή και στα δύο σύνολα.

Ο αλγόριθμος Lion παρουσιάζει διακυμάνσεις στην απώλεια τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης, με την τιμή της να αυξάνεται σταδιακά και στα δύο σύνολα. Η ορθότητα του Lion αυξάνεται, αλλά με σημαντικές διακυμάνσεις σε όλη τη διάρκεια της εκπαίδευσης και της επαλήθευσης.
Στα διαγράμματα precision, recall και F1-score, οι περισσότεροι αλγόριθμοι βελτιστοποίησης επιτυγχάνουν τιμές κοντά στο 0.7, υποδηλώνοντας μέτρια έως καλή απόδοση.

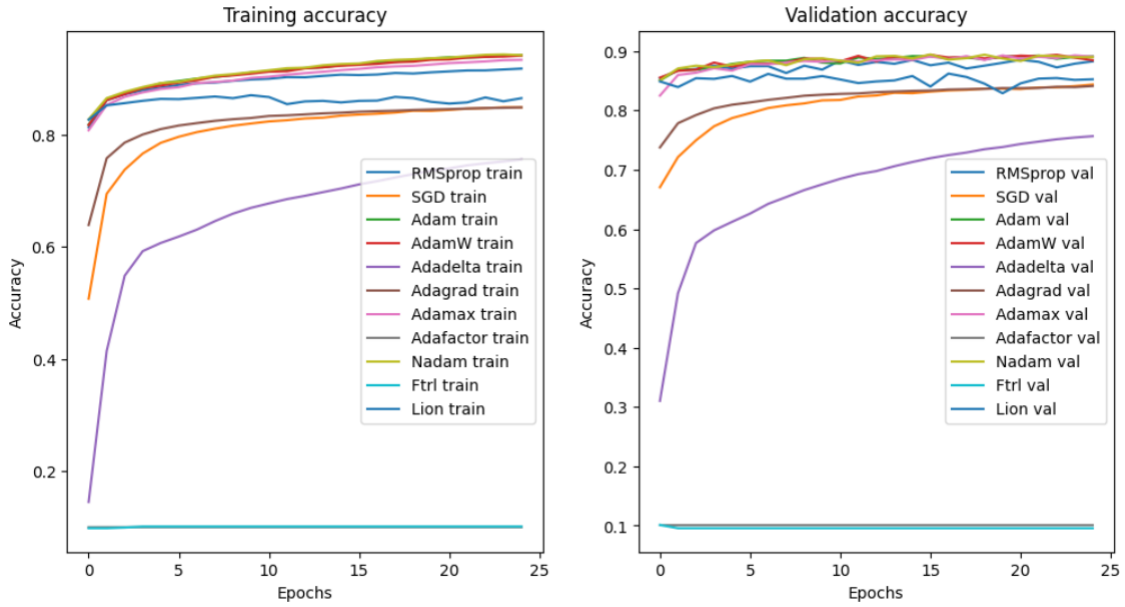
Εξαίρεση αποτελούν οι αλγόριθμοι Adafactor και FTRL, οι οποίοι αναγνωρίζουν μόνο μία κλάση, παρουσιάζοντας εξαιρετικά χαμηλές επιδόσεις.

Γενικά, οι αλγόριθμοι Adamax, Nadam, Adam, AdamW και RMSprop καταγράφουν τις υψηλότερες επιδόσεις στο σύνολο δεδομένων Fashion MNIST, με ακρίβειες άνω του 88%. Παρά τις σχετικά καλές αποδόσεις τους, εμφανίζουν διακυμάνσεις στην απώλεια και την ορθότητα στο σύνολο επαλήθευσης, γεγονός που υποδηλώνει κάποια αστάθεια κατά την προσαρμογή σε νέα δεδομένα.

Από την άλλη πλευρά, οι αλγόριθμοι SGD, Adagrad και Adadelta, αν και καταγράφουν χαμηλότερες επιδόσεις, παρουσιάζουν μεγαλύτερη σταθερότητα και καλύτερη ικανότητα γενίκευσης σε άγνωστα δεδομένα, καθιστώντας τους πιο αξιόπιστους για προβλήματα που απαιτούν σταθερή απόδοση.

Ο αλγόριθμος Lion εμφανίζει τη μεγαλύτερη αστάθεια, με έντονες αυξομειώσεις στην ορθότητα και την απώλεια, γεγονός που υποβαθμίζει την απόδοσή του σε σύγκριση με τους υπόλοιπους αλγόριθμους.

Οι FTRL και Adafactor καταγράφουν τις χαμηλότερες επιδόσεις, εμφανίζοντας πολύ χαμηλή ορθότητα και δυσκολία στη βελτίωση, τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης, γεγονός που υποδηλώνει αδυναμία προσαρμογής στο συγκεκριμένο σύνολο δεδομένων.



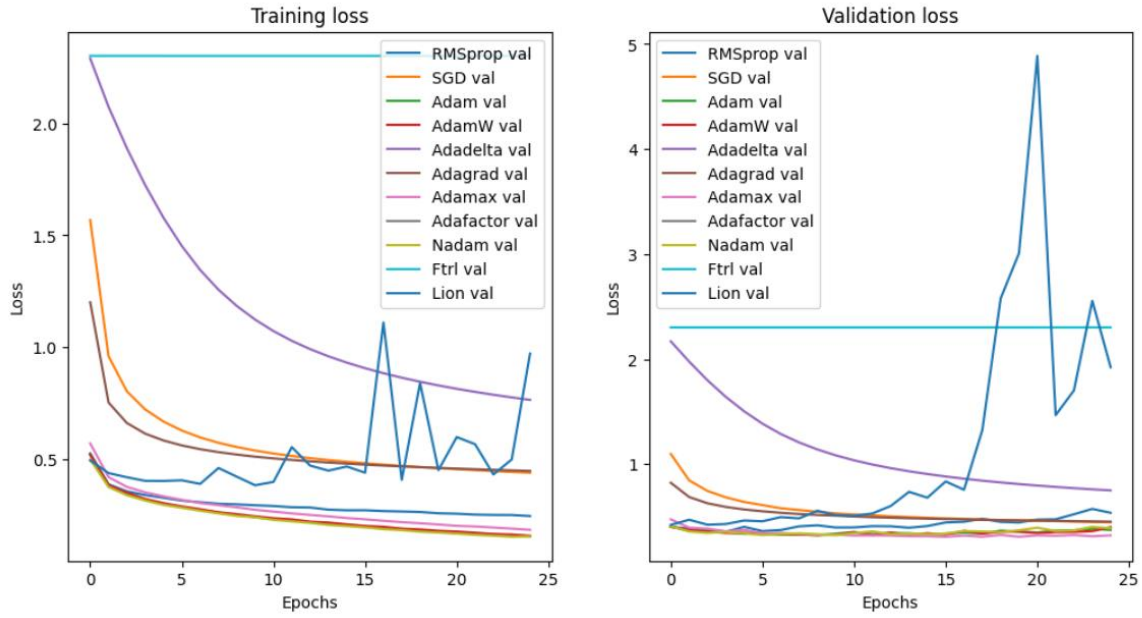
Εικόνα 7- Fashion MNIST Accuracy

Στις γραφικές παραστάσεις της ορθότητας για τα σύνολα εκπαίδευσης και επαλήθευσης, παρατηρείται ότι οι περισσότεροι αλγόριθμοι βελτιστοποίησης ακολουθούν μια λογαριθμική αύξηση, γεγονός που υποδηλώνει σταδιακή βελτίωση της απόδοσης καθώς προχωρά η εκπαίδευση.

Ωστόσο, στο σύνολο επαλήθευσης εμφανίζονται περισσότερες αυξομειώσεις, ιδιαίτερα στους αλγόριθμους RMSprop, Adam, AdamW και Nadam, κάτι που υποδεικνύει αστάθεια στη γενίκευση.

Ο αλγόριθμος Adadelta ξεκινά από σχετικά χαμηλή ορθότητα και σημειώνει σταδιακή βελτίωση.

Παράλληλα, ο Adafactor και ο FTRL παρουσιάζουν σταθερές καμπύλες, παράλληλες με τον άξονα των εποχών, υποδηλώνοντας ότι δεν καταφέρνουν να βελτιώσουν την απόδοσή τους, παραμένοντας στο ίδιο επίπεδο και στα δύο σύνολα. Αυτό οφείλεται στην περιορισμένη ικανότητά τους να αναγνωρίσουν περισσότερες κατηγορίες, καθώς καταγράφουν σταθερά πολύ χαμηλές τιμές ορθότητας, επιβεβαιώνοντας ότι αναγνωρίζουν μόνο μία κλάση.



Εικόνα 8- Fashion MNIST Loss

Οι γραφικές παραστάσεις των αλγορίθμων δείχνουν μια γενική εκθετική μείωση της απώλειας στο σύνολο εκπαίδευσης, υποδεικνύοντας ότι το μοντέλο μαθαίνει σταδιακά κατά τη διάρκεια της εκπαίδευσης.

Οι αλγόριθμοι FTRL και Adafactor παρουσιάζουν σταθερή, σχεδόν παράλληλη με τον άξονα των εποχών πορεία, κάτι που υποδηλώνει μηδαμινή πρόοδο στην εκπαίδευση.

Ο Adadelta εμφανίζει την πιο αργή μείωση της απώλειας σε σχέση με τους άλλους αλγόριθμους, ενώ οι υπόλοιποι αλγόριθμοι όπως ο Adam, AdamW και Nadam καταγράφουν σταθερή μείωση της απώλειας μέχρι να γίνουν σχεδόν παράλληλοι με τον άξονα των εποχών, δείχνοντας ότι η εκπαίδευσή τους σταθεροποιείται. Αντίστοιχη συμπεριφορά παρατηρείται και στο σύνολο επαλήθευσης, όπου οι περισσότεροι αλγόριθμοι ακολουθούν παρόμοιες τάσεις.

Ωστόσο, ο Lion παρουσιάζει σημαντική αύξηση της απώλειας μετά από την 15η εποχή, φτάνοντας σε υψηλές τιμές. Ο Lion εμφανίζει επίσης μεγάλες διακυμάνσεις, με την απώλεια του να αυξάνεται έντονα, ιδιαίτερα γύρω στην 20η εποχή, και παρουσιάζει τις πιο έντονες αυξομειώσεις σε σύγκριση με τους υπόλοιπους αλγόριθμους. Αυτά τα αποτελέσματα δείχνουν την αστάθεια του Lion στο σύνολο επαλήθευσης, καθώς και τη δυσκολία τους να γενικεύσουν σε δεδομένα εκτός εκπαίδευσης.

Συμπερασματικά, ο Nadam φαίνεται να είναι ο πιο αποδοτικός αλγόριθμος για το Fashion MNIST, επιτυγχάνοντας ορθότητα 88.7%, ελαφρώς υψηλότερη από τον Adamax που φτάνει το 88.62%. Παρόλο που αυτές οι ακρίβειες είναι υψηλές σε σχέση με άλλες προσεγγίσεις και δεδομένα, οι συγκεκριμένες τιμές φαίνεται να υποδεικνύουν ότι ο Nadam αντιμετωπίζει καλύτερα τις προκλήσεις του Fashion MNIST σε σύγκριση με τους άλλους αλγόριθμους που δοκιμάστηκαν.

Σύνολο δεδομένων IMDB

Στο σύνολο δεδομένων IMDB, όλοι οι αλγόριθμοι παρουσιάζουν ορθότητα 0.5, αναγνωρίζοντας μόνο τη μία κλάση. Αυτό υποδηλώνει ότι το feed forward neural network αδυνατεί να επεξεργαστεί αποτελεσματικά το συγκεκριμένο σύνολο δεδομένων.

Κατά την εκπαίδευση του μοντέλου για 25 εποχές, παρατηρείται ότι η απώλεια εμφανίζει διακυμάνσεις με γενική τάση αύξησης, ενώ η ορθότητα βελτιώνεται σταδιακά.

Στον RMSprop, η ορθότητα στο σύνολο εκπαίδευσης φτάνει το 1, αλλά στο σύνολο επαλήθευσης παρατηρούνται αυξομειώσεις που καταλήγουν σε τελική μείωση. Η απώλεια στο σύνολο εκπαίδευσης μειώνεται σε πολύ χαμηλές τιμές, κοντά στο μηδέν, ενώ στο σύνολο επαλήθευσης αυξάνεται συνεχώς, ξεπερνώντας το 1.

Ο αλγόριθμος SGD παρουσιάζει διακυμάνσεις και στα δύο σύνολα, με την ορθότητα να αυξάνεται. Η απώλεια μειώνεται σταθερά στο σύνολο εκπαίδευσης, αλλά στο σύνολο επαλήθευσης εμφανίζει διακυμάνσεις.

Οι αλγόριθμοι Adam, AdamW, Adamax και Nadam δείχνουν αύξηση της ορθότητας στο σύνολο εκπαίδευσης, φτάνοντας στο 1, ενώ στο σύνολο επαλήθευσης η ορθότητα παρουσιάζει διακυμάνσεις με τελική μείωση. Η απώλεια στο σύνολο εκπαίδευσης μειώνεται με διακυμάνσεις και φτάνει σχεδόν στο μηδέν, ενώ στο σύνολο επαλήθευσης αυξάνεται με διακυμάνσεις, ξεπερνώντας το 1.

Ο Adadelta παρουσιάζει διακυμάνσεις στην ορθότητα και στα δύο σύνολα, με σταδιακή βελτίωση. Η απώλεια μειώνεται με διακυμάνσεις και στα δύο σύνολα.

Ο Adagrad εμφανίζει επίσης διακυμάνσεις στην ορθότητα, η οποία όμως αυξάνεται σταθερά, ενώ η απώλεια μειώνεται ομαλά και στα δύο σύνολα.

Ο Adafactor ακολουθεί παρόμοια πορεία με τους άλλους αλγόριθμους, με την ορθότητα στο σύνολο εκπαίδευσης να φτάνει το 1 και στο σύνολο επαλήθευσης να παρουσιάζει αυξομειώσεις, με συνολική τάση αύξησης. Η απώλεια στο σύνολο εκπαίδευσης φτάνει σε σχεδόν μηδενικά επίπεδα, ενώ στο σύνολο επαλήθευσης αυξάνεται με διακυμάνσεις.

Ο FTRL παρουσιάζει διακυμάνσεις στο σύνολο εκπαίδευσης με αύξηση της ορθότητας, αλλά στο σύνολο επαλήθευσης η ορθότητα παραμένει σταθερή. Η απώλεια παραμένει σταθερή και στα δύο σύνολα με μικρές διακυμάνσεις.

Ο Lion εμφανίζει διακυμάνσεις στην ορθότητα του συνόλου εκπαίδευσης, ενώ στο σύνολο επαλήθευσης η ορθότητα μειώνεται. Η απώλεια, και στα δύο σύνολα, αυξάνεται με διακυμάνσεις, υποδεικνύοντας αστάθεια.

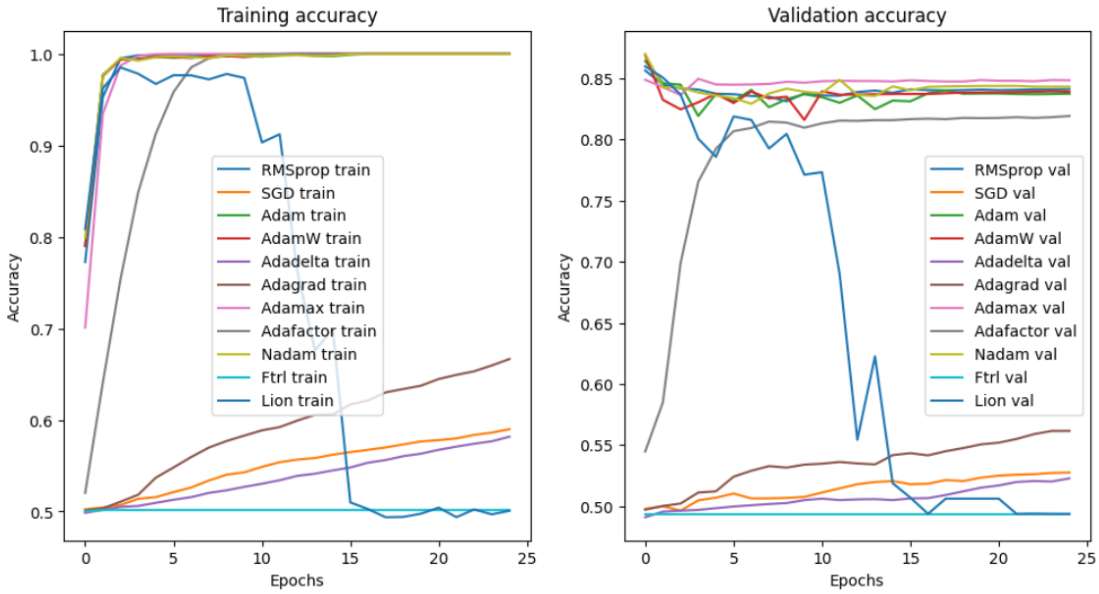
Στα ιστογράμματα των μετρικών, το precision κυμαίνεται κοντά στο 0.5, το recall φτάνει σχεδόν το 1, ενώ το F1-score είναι γύρω στο 0.7, υποδηλώνοντας μέτρια έως καλή απόδοση για τους περισσότερους αλγόριθμους.

Γενικά, φαίνεται ότι κανένας από τους αλγόριθμους δεν αποδίδει ικανοποιητικά στο σύνολο δεδομένων IMDB. Παρά το γεγονός ότι η ορθότητα είναι αρκετά υψηλή τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης, η αυξημένη απώλεια, η οποία αρκετές φορές ξεπερνά το 1, υποδηλώνει την ύπαρξη overfitting. Αυτό σημαίνει ότι τα μοντέλα "μαθαίνουν" υπερβολικά καλά τα δεδομένα εκπαίδευσης, χωρίς να μπορούν να γενικεύσουν καλά στα δεδομένα επαλήθευσης.

Οι αλγόριθμοι RMSprop, Adam, AdamW, Adamax και Nadam δείχνουν αύξηση της ορθότητας στο σύνολο εκπαίδευσης, φτάνοντας το 1, όμως η απόδοσή τους στο σύνολο επαλήθευσης εμφανίζει διακυμάνσεις και τελικά μείωση της ορθότητας, μαζί με αύξηση της απώλειας. Αυτό ενισχύει την ένδειξη για overfitting, καθώς η απώλεια στο σύνολο εκπαίδευσης μειώνεται σε σχεδόν μηδενικά επίπεδα, ενώ στο σύνολο επαλήθευσης αυξάνεται συνεχώς, ξεπερνώντας το 1.

Οι αλγόριθμοι SGD, Adadelta, Adagrad και Adafactor παρουσιάζουν διακυμάνσεις στην ορθότητα και την απώλεια, αλλά επιδεικνύουν μεγαλύτερη σταθερότητα σε σύγκριση με τις εξελιγμένες παραλλαγές του Adam. Ωστόσο, καταλήγουν με χαμηλότερες ακρίβειες, παρόλο που είναι πιο ανθεκτικοί στο overfitting.

Ο Lion εμφανίζει τη μεγαλύτερη αστάθεια, με αυξομειώσεις στην ορθότητα και στα δύο σύνολα, ενώ η απώλεια του αυξάνεται σημαντικά, υποδεικνύοντας προβλήματα στην προσαρμογή του μοντέλου και κακή απόδοση συγκριτικά με άλλους αλγόριθμους.



Εικόνα 9- IMDB Accuracy

Στη γραφική παράσταση της ορθότητας για το σύνολο εκπαίδευσης, οι περισσότεροι αλγόριθμοι εμφανίζουν λογαριθμική αύξηση της ορθότητας με την πρόοδο των εποχών.

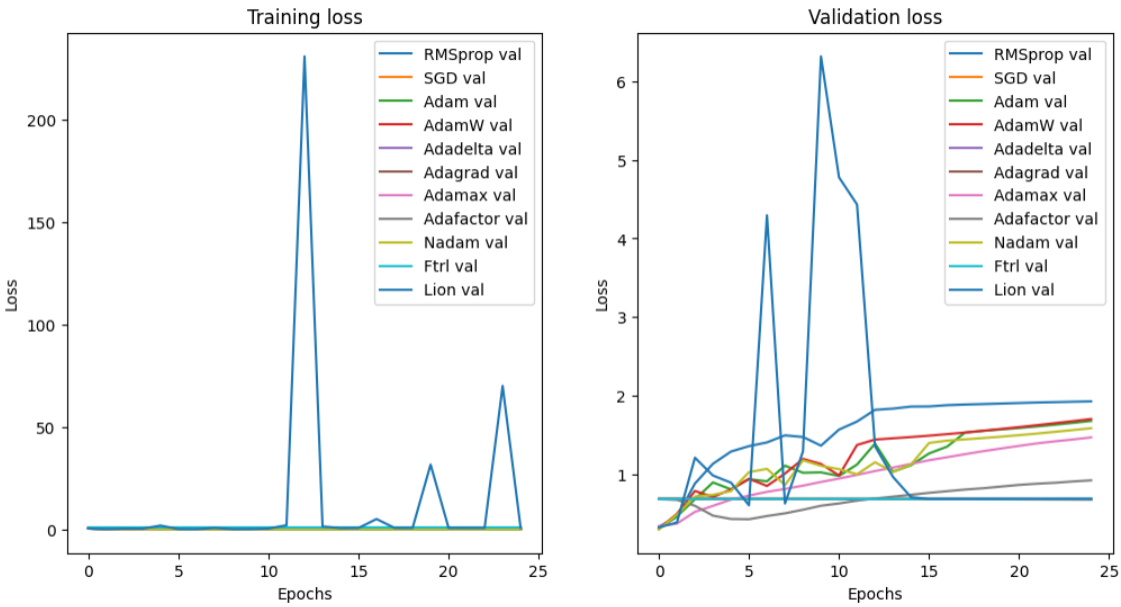
Οι αλγόριθμοι SGD, Adadelta και Adagrad παρουσιάζουν σχεδόν γραμμική αύξηση, με καμπύλες της μορφής y=ax+b.

Ο FTRL παραμένει σχεδόν σταθερός, με την καμπύλη του παράλληλη προς τον άξονα των εποχών, υποδεικνύοντας μηδαμινή πρόοδο, ενώ ο Lion, παρά την αρχική αύξηση, σημειώνει απότομη πτώση της ορθότητας.

Στη γραφική παράσταση της ορθότητας για το σύνολο επαλήθευσης, οι αλγόριθμοι FTRL, Adafactor, Adagrad, Adadelta, SGD και Lion εμφανίζουν παρόμοια συμπεριφορά, με περιορισμένη βελτίωση και παραμονή σε χαμηλά επίπεδα.

Οι υπόλοιποι αλγόριθμοι, όπως RMSprop, Adam, AdamW, Nadam και Adamax, δείχνουν αρχικά πτώση της ορθότητας, ακολουθούμενη από διακυμάνσεις. Ωστόσο, η ακρίβειά τους δεν βελτιώνεται σημαντικά στη συνέχεια, γεγονός που υποδηλώνει ότι τα μοντέλα δεν μπορούν να

γενικεύσουν καλά στα δεδομένα επαλήθευσης και αντιμετωπίζουν δυσκολίες στη σταθερή απόδοση.



Εικόνα 10- IMDB Loss

Στη γραφική παράσταση της απώλειας για το σύνολο εκπαίδευσης, οι αλγόριθμοι FTRL και Nadam παραμένουν σχεδόν σταθεροί, με τις καμπύλες τους να είναι παράλληλες προς τον άξονα των εποχών, υποδεικνύοντας μηδαμινή μεταβολή στην απώλεια.

Ο Lion παρουσιάζει έντονα πικ γύρω από την 10η, 20η και 25η εποχή, με σημαντική αύξηση της απώλειας, γεγονός που καθιστά αδύνατο να αποτυπωθούν οι μικρότερες τιμές απώλειας των άλλων αλγορίθμων, καθώς οι εκρήξεις αυτές κυριαρχούν στην απεικόνιση. Στη γραφική παράσταση της απώλειας για το σύνολο επαλήθευσης, ο Lion παρουσιάζει έντονα πικ στην 5η και 10η εποχή, και στη συνέχεια η απώλεια του γίνεται σχεδόν παράλληλη με τον άξονα των εποχών, υποδηλώνοντας σταθεροποίηση σε υψηλές τιμές.

Ο FTRL παραμένει επίσης παράλληλος με τον άξονα των εποχών, χωρίς σημαντικές μεταβολές στην απώλεια.

Ο Adafactor παρουσιάζει αρχικά μια καμπύλη μείωσης της απώλειας, η οποία στη συνέχεια σταθεροποιείται και αρχίζει να δείχνει μια ελαφριά αυξητική τάση. Ομοίως, ο Adamax παρουσιάζει μια σταθερή καμπύλη με ελαφρά αυξητική πορεία.

Οι υπόλοιποι αλγόριθμοι ακολουθούν παρόμοια συμπεριφορά με διακυμάνσεις, ενώ ο RMSprop εμφανίζει τις περισσότερες αυξομειώσεις στην απώλεια, καταλήγοντας σε σημαντική αύξηση, γεγονός που δείχνει μεγαλύτερη δυσκολία στη γενίκευση του μοντέλου.

Συμπερασματικά, ο αλγόριθμος RMSprop φαίνεται να είναι ο πιο αποδοτικός για το σύνολο δεδομένων IMDB, επιτυγχάνοντας ορθότητα 1 στο σύνολο εκπαίδευσης, αν και αυτό συνοδεύεται από σημαντικά προβλήματα overfitting. Παρόλο που η απόδοσή του στο σύνολο επαλήθευσης παρουσιάζει αυξομειώσεις και η απώλεια αυξάνεται, εξακολουθεί να δείχνει καλύτερη απόδοση σε σχέση με τους υπόλοιπους αλγόριθμους που δοκιμάστηκαν.

Οι εξελιγμένες παραλλαγές του Adam, όπως ο Adam, ο AdamW, ο Adamax και ο Nadam, παρουσιάζουν επίσης καλή απόδοση στο σύνολο εκπαίδευσης, αλλά δυσκολεύονται να γενικεύσουν σωστά στα δεδομένα επαλήθευσης, γεγονός που υποδεικνύει ότι το RMSprop χειρίζεται τις προκλήσεις του IMDB ελαφρώς καλύτερα από τους υπόλοιπους αλγόριθμους, παρά τα προβλήματα γενίκευσης.

Σύνολο δεδομένων Reuters

Στο σύνολο δεδομένων Reuters, η απόδοση των αλγορίθμων βελτιστοποίησης ποικίλλει σημαντικά ως προς την ορθότητα. Ο Nadam ξεχωρίζει με την καλύτερη επίδοση, φτάνοντας το 66.25%, ενώ ακολουθείται στενά από τον Adam με 65.71% και τον AdamW με 66.02%. Ο Adamax και ο RMSprop επιτυγχάνουν επίσης υψηλές τιμές ορθότητας, φτάνοντας το 65.80% και 63.71% αντίστοιχα. Ο Adafactor σημειώνει ικανοποιητική ορθότητα με 63.80%, αν και δεν φτάνει τα επίπεδα των κορυφαίων αλγορίθμων. Από την άλλη πλευρά, οι αλγόριθμοι Adagrad και SGD εμφανίζουν πολύ χαμηλότερες επιδόσεις, με ορθότητα 47.41% και 36.82% αντίστοιχα. Οι αλγόριθμοι Adadelta, FTRL και Lion καταγράφουν ακόμη χαμηλότερες τιμές ορθότητας,

φτάνοντας στο 36.77%, 36.19% και 36.19% αντίστοιχα. Αυτά τα αποτελέσματα υπογραμμίζουν τη σημαντικότητα της επιλογής του κατάλληλου αλγορίθμου βελτιστοποίησης για προβλήματα ταξινόμησης κειμένων, με τις εξελιγμένες παραλλαγές του Adam να υπερτερούν σε απόδοση έναντι των άλλων μεθόδων.

Κατά την εκπαίδευση ενός μοντέλου για 25 εποχές, οι αλγόριθμοι βελτιστοποίησης παρουσίασαν διαφορετικές συμπεριφορές.

Ο RMSprop εμφάνισε σημαντικές αυξομειώσεις τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης. Η ορθότητα αυξήθηκε στο σύνολο εκπαίδευσης, ενώ η απώλεια μειώθηκε, αλλά στο σύνολο επαλήθευσης η απώλεια αυξήθηκε σταθερά.

Ο SGD παρουσίασε διακυμάνσεις και στα δύο σύνολα, με την ορθότητα να αυξάνεται και στο σύνολο εκπαίδευσης η απώλεια να μειώνεται με αυξομειώσεις, ενώ στο σύνολο επαλήθευσης μειώθηκε ομαλά.

Οι αλγόριθμοι Adam, AdamW και Adamax έδειξαν αυξομειώσεις και στα δύο σύνολα, με την ορθότητα να αυξάνεται στο σύνολο εκπαίδευσης και στο σύνολο επαλήθευσης να εμφανίζει διακυμάνσεις. Η απώλεια στο σύνολο εκπαίδευσης μειώθηκε με διακυμάνσεις, ενώ στο σύνολο επαλήθευσης αυξήθηκε με διακυμάνσεις.

Ο Adagrad εμφάνισε διακυμάνσεις στην ορθότητα και στα δύο σύνολα, με σταθερή μείωση της απώλειας στο σύνολο εκπαίδευσης, αλλά στο σύνολο επαλήθευσης αυξήθηκε ομαλά.

Ο Adadelta παρουσίασε σταδιακή αύξηση της ορθότητας με διακυμάνσεις και στα δύο σύνολα, ενώ η απώλεια μειώθηκε ομαλά και στα δύο, αναγνωρίζοντας μόνο δύο κλάσεις.

Ο Adamax εμφάνισε διακυμάνσεις και στα δύο σύνολα, με αύξηση της ορθότητας και μείωση της απώλειας.

Ο Adafactor βελτίωσε την απόδοση με ομαλή αύξηση της ορθότητας και στα δύο σύνολα, ενώ η απώλεια μειώθηκε σταθερά.

Αντίθετα, ο Nadam εμφάνισε πτώση στην ορθότητα με διακυμάνσεις στο σύνολο επαλήθευσης και αυξομειώσεις στο σύνολο εκπαίδευσης, ενώ η απώλεια αυξήθηκε στο σύνολο επαλήθευσης και μειώθηκε στο σύνολο εκπαίδευσης με διακυμάνσεις.

Ο FTRL εμφάνισε αύξηση της ορθότητας με διακυμάνσεις στο σύνολο εκπαίδευσης, διατήρησε σταθερή ορθότητα στο σύνολο επαλήθευσης και μείωσε ομαλά την απώλεια και στα δύο σύνολα, αναγνωρίζοντας μόνο μία κλάση.

Ο Lion παρουσίασε απότομες διακυμάνσεις με μείωση της ορθότητας και σταθεροποίησή της σε χαμηλό επίπεδο στο σύνολο επαλήθευσης, ενώ η απώλεια αυξήθηκε με διακυμάνσεις και στα δύο σύνολα. Συνολικά, οι αλγόριθμοι βελτιστοποίησης επηρέασαν την απόδοση του μοντέλου με διαφορετικούς τρόπους, με κάποιους να δείχνουν πιο σταθερές επιδόσεις από άλλους.

Στα ιστογράμματα των μετρικών, precision, recall και F1-score και για τους αλγόριθμους RMSprop, Adam, Adamax και Nadam, παρατηρούνται διακυμάνσεις στις κλάσεις. Ο Adafactor εμφανίζει επίσης διακυμάνσεις, αν και αναγνωρίζει μόνο μερικές κλάσεις. Οι αλγόριθμοι FTRL και Lion αναγνωρίζουν μόνο μία κλάση, ενώ οι SGD, Adagrad και Adadelta αναγνωρίζουν δύο κλάσεις.

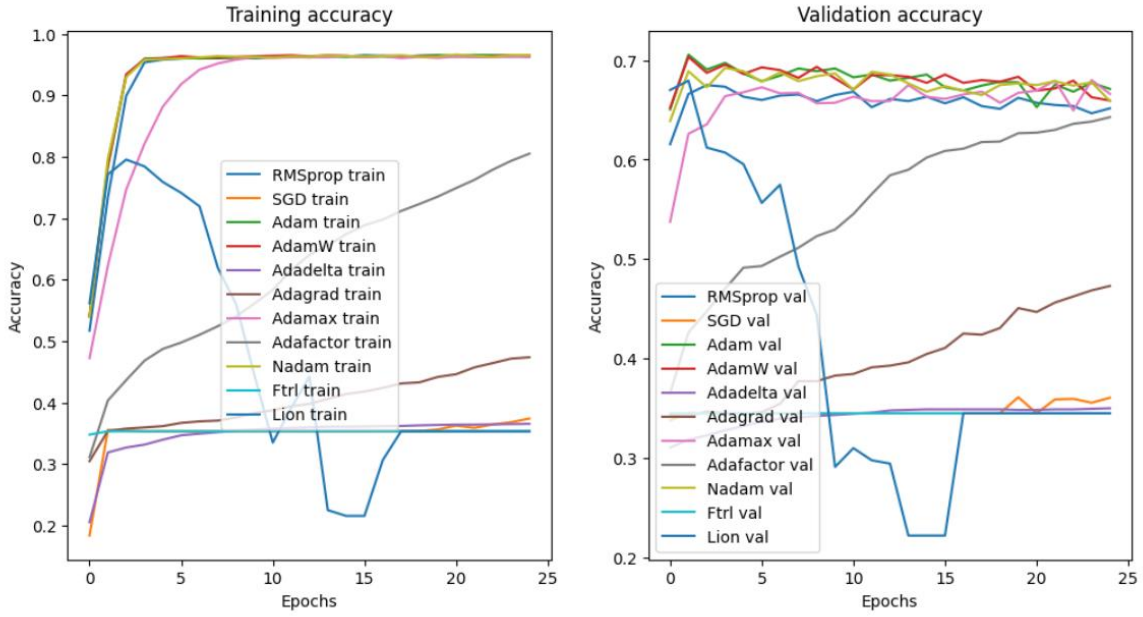
Γενικά, φαίνεται ότι κανένας από τους αλγόριθμους δεν αποδίδει ικανοποιητικά στο σύνολο δεδομένων Reuters. Παρά το γεγονός ότι η ορθότητα στο σύνολο εκπαίδευσης είναι αρκετά υψηλή, τα προβλήματα overfitting καθιστούν τη συμπεριφορά τους στο σύνολο επαλήθευσης μη ικανοποιητική.

Οι αλγόριθμοι Adam, AdamW, Nadam, RMSprop, και Adamax καταγράφουν σημαντική αύξηση στην ορθότητα στο σύνολο εκπαίδευσης, φτάνοντας κοντά στο 1, αλλά στο σύνολο επαλήθευσης παρατηρούνται διακυμάνσεις και πτώση της ορθότητας, ενώ η απώλεια αυξάνεται. Αυτό

υποδεικνύει ότι τα μοντέλα μαθαίνουν υπερβολικά καλά τα δεδομένα εκπαίδευσης, αλλά δεν καταφέρνουν να γενικεύσουν αποτελεσματικά σε νέα δεδομένα.

Αντίθετα, οι αλγόριθμοι SGD, Adadelta, Adagrad και Adafactor επιδεικνύουν πιο σταθερή συμπεριφορά στο σύνολο επαλήθευσης, με την ορθότητα να αυξάνεται και την απώλεια να μειώνεται. Παρόλο που η συνολική τους ορθότητα είναι χαμηλότερη σε σύγκριση με τους πιο εξελιγμένους αλγόριθμους, εμφανίζουν μεγαλύτερη ανθεκτικότητα στο overfitting, γεγονός που τους καθιστά πιο αξιόπιστους για τη γενίκευση σε νέα δεδομένα.

Ο αλγόριθμος Lion παρουσιάζει τη μεγαλύτερη αστάθεια, με έντονες αυξομειώσεις στην ορθότητα και την απώλεια και στα δύο σύνολα, υποδεικνύοντας προβλήματα στη σταθερότητα και τη γενίκευση του μοντέλου, καθιστώντας τον λιγότερο αποδοτικό σε σχέση με τους υπόλοιπους αλγόριθμους.



Εικόνα 11- Reuters Accuracy

Στη γραφική παράσταση της ορθότητας στο σύνολο εκπαίδευσης, οι αλγόριθμοι RMSprop, Adam, AdamW, Adamax, Nadam και Adafactor εμφανίζουν λογαριθμική αύξηση, με την ακρίβειά τους να βελτιώνεται σταδιακά καθώς προχωρά η εκπαίδευση.

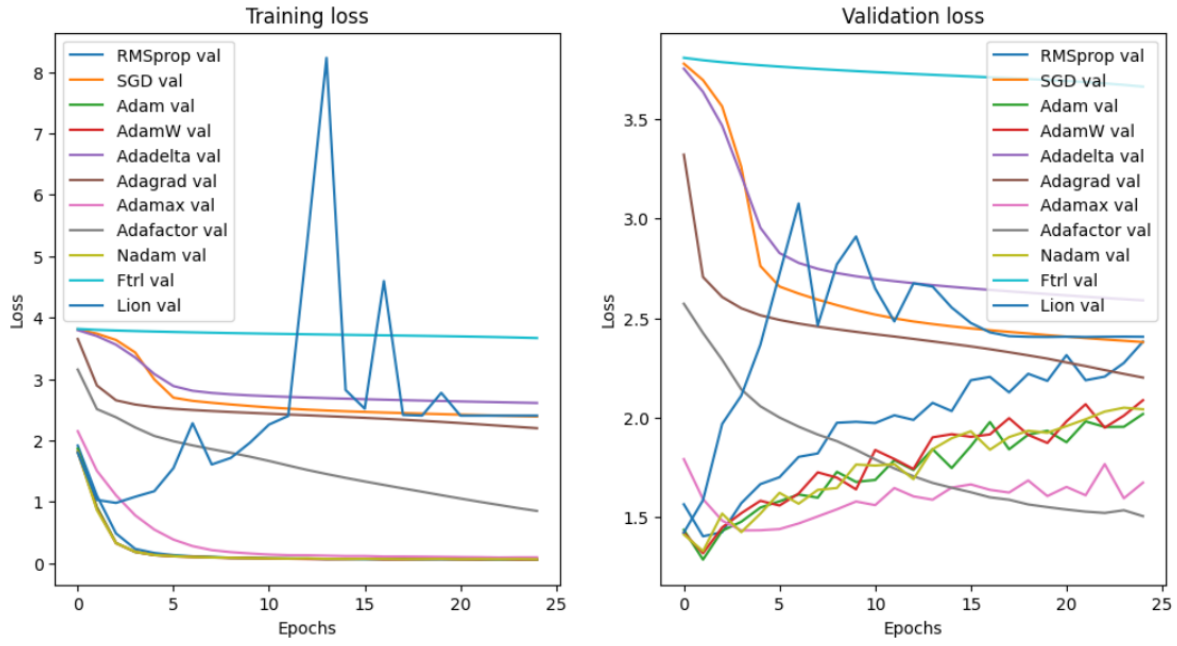
Οι αλγόριθμοι SGD, Adagrad και Adadelta παρουσιάζουν αρχικά λογαριθμική αύξηση, η οποία στη συνέχεια γίνεται γραμμική.

Ο αλγόριθμος Lion παρουσιάζει αρχική αύξηση, αλλά στη συνέχεια ακολουθεί πτωτική πορεία με διακυμάνσεις, καταλήγοντας σε σταθερή πορεία.

Ο αλγόριθμος FTRL παραμένει παράλληλος με τον άξονα των εποχών, χωρίς σημαντική βελτίωση στην ορθότητα. Στη γραφική παράσταση της ορθότητας στο σύνολο επαλήθευσης, παρατηρούνται αυξομειώσεις στην απόδοση των αλγορίθμων.

Οι αλγόριθμοι RMSprop, Adam, AdamW, Adamax, και Nadam παρουσιάζουν αυξομειώσεις στην ορθότητα, ενώ οι SGD, Adagrad,Adadelta και Adafactor εμφανίζουν μικρότερες διακυμάνσεις.

Οι αλγόριθμοι Lion, FTRL (παράλληλος με τον άξονα των εποχών) έχουν διακυμάνσεις, με τη γραμμή τους να παραμένει σταθερή σε χαμηλά επίπεδα.



Εικόνα 12- Reuters Loss

Στη γραφική παράσταση της απώλειας στο σύνολο εκπαίδευσης, ο αλγόριθμος FTRL παραμένει σταθερός, παράλληλος με τον άξονα των εποχών, με απώλεια περίπου ίση με 4.

Οι αλγόριθμοι SGD, Adadelta και Adagrad παρουσιάζουν αρχική μείωση στην απώλεια και στη συνέχεια σταθεροποιούνται, με την πορεία τους να γίνεται παράλληλη με τον άξονα των εποχών σε συγκεκριμένες τιμές.

Ο Lion εμφανίζει έντονες αυξομειώσεις στην απώλεια, η οποία τελικά αυξάνεται και καταλήγει επίσης να ευθυγραμμίζεται παράλληλα με τον άξονα των εποχών.

Ο Adafactor αρχικά παρουσιάζει λογαριθμική μείωση της απώλειας, η οποία στη συνέχεια γίνεται γραμμική.

Οι υπόλοιποι αλγόριθμοι, όπως οι Adam, AdamW, Adamax, Nadam και RMSprop, παρουσιάζουν εκθετική μείωση της απώλειας κατά τη διάρκεια της εκπαίδευσης και τελικά ευθυγραμμίζονται παράλληλα με τον άξονα των εποχών, πλησιάζοντας την τιμή 0.

Στη γραφική παράσταση της απώλειας στο σύνολο επαλήθευσης, ο FTRL παραμένει και πάλι παράλληλος με τον άξονα των εποχών, ενώ οι αλγόριθμοι SGD, Adadelta και Adagrad παρουσιάζουν αρχικά μείωση και στη συνέχεια σταθεροποιούνται, ακολουθώντας μια γραμμή παράλληλη με τον άξονα των εποχών.

Ο Adafactor παρουσιάζει παρόμοια πορεία με τη γραφική παράσταση του συνόλου εκπαίδευσης, με αρχική μείωση της απώλειας. Οι υπόλοιποι αλγόριθμοι, αν και αρχικά παρουσιάζουν λογαριθμική μείωση της απώλειας, εμφανίζουν στη συνέχεια αύξηση με έντονες διακυμάνσεις, υποδεικνύοντας δυσκολίες στη γενίκευση.

Ο πιο αποδοτικός αλγόριθμος είναι ο Nadam, καθώς επιτυγχάνει την υψηλότερη ορθότητα (66.25%), ακολουθούμενος πολύ κοντά από τον AdamW με 66.02% και τον Adamax με 65.80%. Αυτοί οι τρεις αλγόριθμοι ξεχωρίζουν για τη συνολική τους απόδοση, παρουσιάζοντας σταθερές και υψηλές επιδόσεις στην ταξινόμηση κειμένων.

Σύνολο δεδομένων California-Housing

Για την αξιολόγηση της απόδοσης των μοντέλων στο σύνολο δεδομένων για παλινδρόμηση, χρησιμοποιήθηκε το μέσο τετραγωνικό σφάλμα (Mean Squared Error - MSE). Ο αλγόριθμος Lion σημείωσε την καλύτερη απόδοση με το χαμηλότερο MSE, το οποίο ήταν 0.3007. Ακολούθησε ο RMSprop με MSE 0.3740, ο AdamW με 0.3631, ο Nadam με 0.3665 και ο Adam με 0.3830. Οι αλγόριθμοι Adamax και SGD παρουσίασαν υψηλότερες τιμές MSE, με 0.5078 και 0.6141 αντίστοιχα. Οι αλγόριθμοι Adadelta, Adagrad και FTRL εμφάνισαν πολύ υψηλότερα MSE, με τιμές 6.57, 2.97 και 5.17 αντίστοιχα, υποδεικνύοντας σημαντικά χαμηλότερη απόδοση στην παλινδρόμηση σε σύγκριση με τους υπόλοιπους αλγορίθμους.

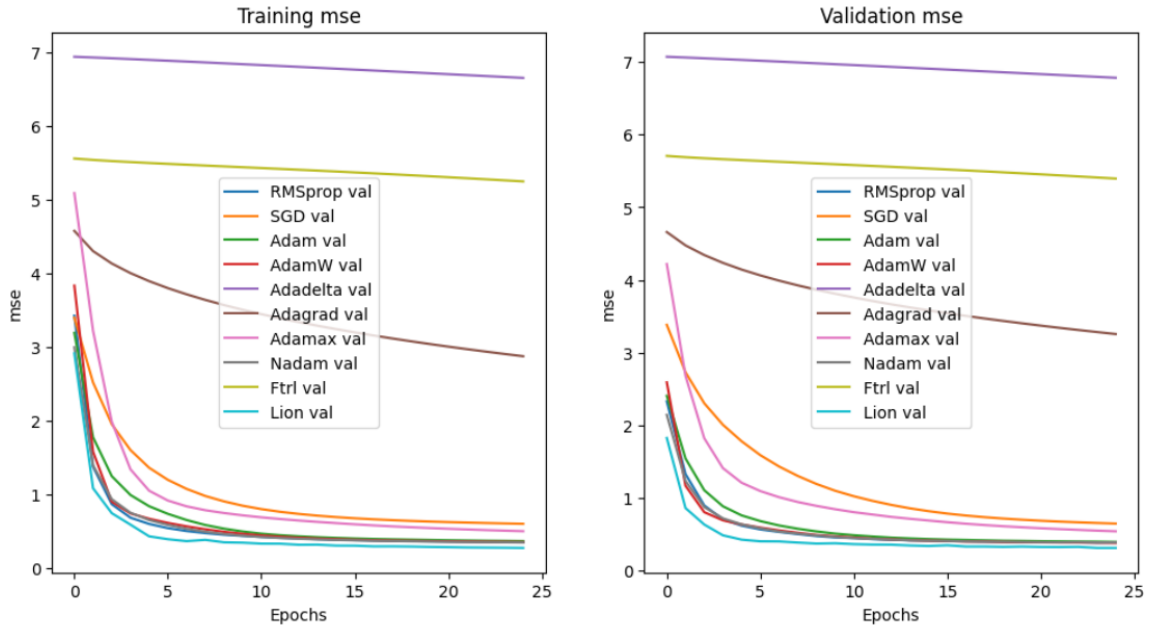
Κατά την εκπαίδευση διάρκειας 25 εποχών, παρατηρείται μείωση του μέσου τετραγωνικού σφάλματος (MSE) στους αλγόριθμους RMSprop, AdamW και Lion, με αυξομειώσεις τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης.

Οι αλγόριθμοι SGD, Adam, Adadelta, Adagrad, Adamax, Nadam και FTRL παρουσιάζουν επίσης μείωση του MSE, αλλά με αυξομειώσεις στο σύνολο εκπαίδευσης, ενώ η μείωση στο σύνολο επαλήθευσης είναι πιο ομαλή. Αυτό δείχνει ότι οι περισσότεροι αλγόριθμοι καταφέρνουν να μειώσουν το σφάλμα τους κατά τη διάρκεια της εκπαίδευσης, αλλά εμφανίζουν διαφορετική σταθερότητα όταν αξιολογούνται σε δεδομένα επαλήθευσης.

Οι περισσότεροι αλγόριθμοι καταγράφουν μια σταθερή μείωση του μέσου τετραγωνικού σφάλματος (MSE), χωρίς σημαντικές διακυμάνσεις, όπως αναφέρθηκε προηγουμένως.
Οι Adadelta, Adagrad και FTRL παραμένουν σε υψηλά επίπεδα MSE, δείχνοντας αδυναμία να προσαρμοστούν στο συγκεκριμένο σύνολο δεδομένων California Housing. Αυτοί οι αλγόριθμοι δεν καταφέρνουν να μειώσουν το σφάλμα τους σημαντικά, γεγονός που σημαίνει ότι δεν καταφέρνουν να βελτιώσουν τη συνολική απόδοση του μοντέλου.

Αντίθετα, οι υπόλοιποι αλγόριθμοι, όπως οι Adam, AdamW, RMSprop, Adamax, και Nadam, ξεκινούν από υψηλότερες τιμές MSE, αλλά παρουσιάζουν σταθερή μείωση, καταλήγοντας σε τιμές κοντά στο μηδέν, γεγονός που υποδεικνύει καλύτερη προσαρμογή στο σύνολο δεδομένων. Αυτό σημαίνει ότι, παρά την αρχική δυσκολία, προσαρμόζονται καλύτερα στο συγκεκριμένο πρόβλημα παλινδρόμησης.

Ο Lion, όπως φαίνεται και από τα διαγράμματα, καταγράφει επίσης σταθερή μείωση του MSE, χωρίς διακυμάνσεις, και φτάνει σε χαμηλά επίπεδα, γεγονός που υποδηλώνει ότι είναι ένας από τους αποδοτικότερους αλγόριθμους για το συγκεκριμένο σύνολο δεδομένων.



Εικόνα 13- California Housing MSE

Στις γραφικές παραστάσεις του MSE για το σύνολο εκπαίδευσης και το σύνολο επαλήθευσης, παρατηρείται εκθετική μείωση στους περισσότερους αλγόριθμους,γεγονός που δείχνει σταθερή βελτίωση στην εκμάθηση του μοντέλου.

Ωστόσο, οι αλγόριθμοι Ftrl και Adadelta παρουσιάζουν σταθερή γραμμική συμπεριφορά, με το MSE να ακολουθεί την πορεία μιας ευθείας γραμμής, υποδηλώνοντας περιορισμένη προσαρμογή στο σύνολο δεδομένων.

Ο πιο αποδοτικός αλγόριθμος είναι ο Lion, καθώς επιτυγχάνει το χαμηλότερο μέσο τετραγωνικό σφάλμα (MSE = 0.3007), γεγονός που υποδεικνύει ότι κάνει τις πιο ακριβείς προβλέψεις από όλους τους αλγόριθμους. Ο RMSprop ακολουθεί με MSE 0.3740, παρουσιάζοντας επίσης υψηλή απόδοση.

4.2. Συνελικτικά Νευρωνικά Δίκτυα (Convolutional Neural Networks)

Σύνολο δεδομένων MNIST

Στο σύνολο δεδομένων MNIST, παρατηρείται υψηλή απόδοση για την πλειοψηφία των αλγορίθμων. Συγκεκριμένα, ο RMSprop επιτυγχάνει ορθότητα 99.32%, ενώ ο SGD φτάνει το 96.96%. Ο Adam και ο Adamax επιτυγχάνουν αντίστοιχα ακρίβειες 99.28% και 99.29%, με τον AdamW και τον Nadam να ακολουθούν με 99.25%. Ο Adadelta παρουσιάζει χαμηλότερη απόδοση με 86.49%, ενώ ο Adagrad φτάνει το 96.53%. Ο Lion καταγράφει ορθότητα 95.31%, ενώ οι αλγόριθμοι Adafactor και Ftrl επιτυγχάνουν χαμηλότερες επιδόσεις με 9.8% και 11.35% αντίστοιχα.

Κατά την εκπαίδευση του μοντέλου για 25 εποχές με διάφορους αλγορίθμους βελτιστοποίησης, παρατηρούνται σημαντικές διαφοροποιήσεις στην απόδοση.

Ο RMSprop δείχνει γενική αύξηση στην ορθότητα τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης, αν και παρατηρούνται διακυμάνσεις, ενώ η απώλεια μειώνεται με αυξομειώσεις.

Ο SGD εμφανίζει βελτίωση με αυξομειούμενη ορθότητα στο σύνολο εκπαίδευσης και το σύνολο επαλήθευσης, ενώ η απώλεια μειώνεται ομαλά και στις δύο περιπτώσεις.

Οι αλγόριθμοι Adam, AdamW και Nadam επιτυγχάνουν υψηλή ορθότητα και χαμηλή απώλεια, αλλά και εδώ υπάρχουν διακυμάνσεις τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης.

Ο Adadelta παρουσιάζει ομαλή αύξηση της ορθότητας στο σύνολο εκπαίδευσης και το σύνολο επαλήθευσης, ενώ η απώλεια μειώνεται σταδιακά, με μικρές διακυμάνσεις στο σύνολο εκπαίδευσης και πιο ομαλή μείωση στο σύνολο επαλήθευσης.

Ο Adagrad δείχνει αύξηση της ορθότητας στο σύνολο εκπαίδευσης με διακυμάνσεις, ενώ στο σύνολο επαλήθευσης η αύξηση είναι πιο σταθερή, με την απώλεια να μειώνεται ομαλά και στις δύο περιπτώσεις.

Ο Adamax εμφανίζει αύξηση της ορθότητας με διακυμάνσεις στο σύνολο εκπαίδευσης και το σύνολο επαλήθευσης, ενώ η απώλεια μειώνεται με αντίστοιχες διακυμάνσεις.

Στον Adafactor, η ορθότητα στο σύνολο εκπαίδευσης αυξάνεται με διακυμάνσεις, ενώ στο σύνολο επαλήθευσης παραμένει σταθερή. Η απώλεια όμως καταλήγει σε τιμές NaN και στις δύο φάσεις.

Ο Ftrl παρουσιάζει διακυμάνσεις στην ορθότητα στο σύνολο εκπαίδευσης με τελική αύξηση, ενώ στο σύνολο επαλήθευσης η ορθότητα παραμένει χαμηλή αλλά σταθερή. Η απώλεια μειώνεται ελάχιστα και με διακυμάνσεις και στις δύο φάσεις, αναγνωρίζοντας μόνο μία κλάση.

Ο Lion εμφανίζει αυξομειώσεις στην ορθότητα, με την ορθότητα στο σύνολο εκπαίδευσης να αυξάνεται και στο σύνολο επαλήθευσης να μειώνεται, ενώ η απώλεια αυξάνεται στο σύνολο επαλήθευσης και μειώνεται στο σύνολο εκπαίδευσης, αμφότερα με διακυμάνσεις.

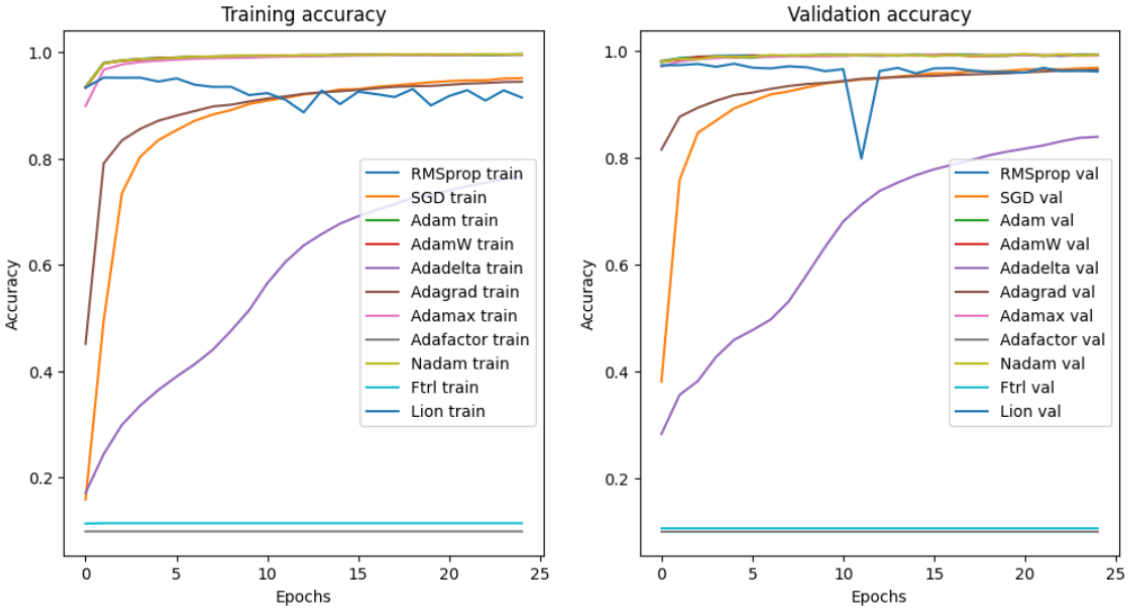
Στα ιστογράμματα των precision, recall και F1-score, οι περισσότερες κλάσεις των αλγορίθμων κινούνται κοντά στο 1, ενώ ο Adadelta βρίσκεται κοντά στο 0.8. Εξαίρεση αποτελούν οι Adafactor και Ftrl, που αναγνωρίζουν μόνο μία κλάση.

Γενικά, οι αλγόριθμοι Adam, AdamW, Nadam, RMSprop και Adamax παρουσιάζουν πολύ καλές επιδόσεις στο σύνολο δεδομένων MNIST με CNNs, με σταθερή μείωση της απώλειας, παρά τις διακυμάνσεις. Ωστόσο, παρότι αποδίδουν πολύ καλά στο σύνολο εκπαίδευσης και επαλήθευσης, οι διακυμάνσεις αυτές υποδηλώνουν μικρή αστάθεια κατά την προσαρμογή.

Από την άλλη πλευρά, οι SGD, Adadelta και Adagrad εμφανίζουν μεγαλύτερη σταθερότητα και ανθεκτικότητα, διατηρώντας καλή γενίκευση χωρίς σημαντικές διακυμάνσεις.

Οι αλγόριθμοι FTRL και Adafactor, ωστόσο, αποδεικνύονται λιγότερο αποδοτικοί, καθώς αναγνωρίζουν μόνο μία κλάση, κάτι που περιορίζει την ικανότητά τους να διακρίνουν μεταξύ των διαφορετικών κατηγοριών του συνόλου δεδομένων.

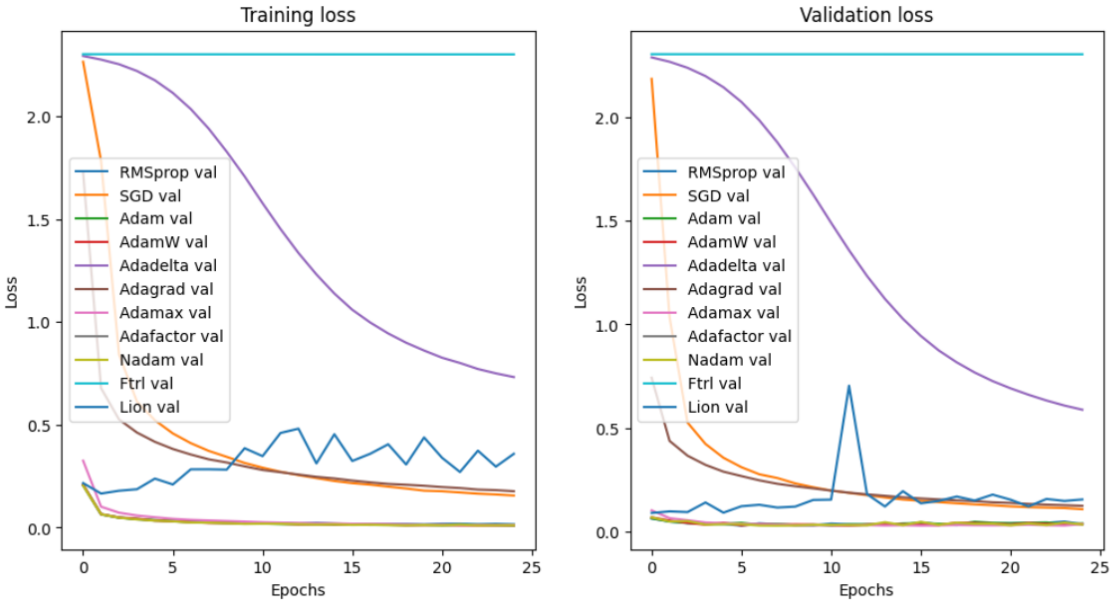
Ο Lion παρουσιάζει έντονες αυξομειώσεις, καθιστώντας τον λιγότερο αποδοτικό σε σύγκριση με τους υπόλοιπους αλγόριθμους.



Εικόνα 14-MNIST Accuracy

Στις γραφικές παραστάσεις της ορθότητας κατά την εκπαίδευση και την επικύρωση, παρατηρείται ότι οι περισσότεροι αλγόριθμοι παρουσιάζουν λογαριθμική αύξηση, υποδεικνύοντας βελτίωση της ορθότητας με την πάροδο του χρόνου.

Ωστόσο, οι αλγόριθμοι Adafactor και Ftrl διατηρούν την ορθότητα τους σταθερή και παράλληλη με τον άξονα των εποχών, κάτι που οφείλεται στο γεγονός ότι αναγνωρίζουν μόνο μία κλάση.



Εικόνα 15- MNIST Loss

Στις γραφικές παραστάσεις της απώλειας κατά την εκπαίδευση και την επικύρωση, οι περισσότεροι αλγόριθμοι παρουσιάζουν μια μικρή μείωση της απώλειας, με τις καμπύλες τους να παραμένουν παράλληλες με τον άξονα των εποχών.

Ο SGD και ο Adagrad ξεχωρίζουν με εκθετική μείωση της απώλειας, ενώ ο Ftrl παραμένει παράλληλος με τον άξονα των εποχών αλλά με υψηλή απώλεια πάνω από 2.

Ο Adadelta, από την άλλη, παρουσιάζει μια σιγμοειδή καμπύλη .

Ο πιο αποδοτικός αλγόριθμος στο σύνολο δεδομένων MNIST είναι ο RMSprop, με ορθότητα 99.32%, ξεπερνώντας όλους τους υπόλοιπους αλγόριθμους. Κατά τη διάρκεια της εκπαίδευσης, εμφανίζει σταθερή βελτίωση στην ορθότητα τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης, ενώ η απώλεια μειώνεται σταδιακά, παρά τις μικρές διακυμάνσεις. Οι αλγόριθμοι Adamax και Adam ακολουθούν πολύ κοντά, με ακρίβειες 99.29% και 99.28% αντίστοιχα, δείχνοντας παρόμοια συμπεριφορά στην εκπαίδευση. Οι AdamW και Nadam βρίσκονται επίσης κοντά, με ακρίβειες γύρω στο 99.25%. Συνολικά, ο RMSprop υπερέχει, προσφέροντας την υψηλότερη ορθότητα και σταθερότητα, γεγονός που τον καθιστά την καλύτερη επιλογή για αυτό το σύνολο δεδομένων.

Σύνολο δεδομένων CIFAR-10

Στο σύνολο δεδομένων CIFAR-10, η πλειονότητα των αλγορίθμων βελτιστοποίησης έχει περιορισμένη απόδοση, με την ορθότητα να κυμαίνεται σε χαμηλά επίπεδα για πολλούς από αυτούς. Συγκεκριμένα, ο RMSprop έχει ορθότητα 71,94%, ο SGD 45,56%, ο Adam 72,32%, ο AdamW 72,4%, ο Adadelta 21,06%, ο Adagrad 42,87%, ο Adamax 72,72%, ο Adafactor 10%, ο Nadam 71,84%, και ο Ftrl, όπως και ο Lion, έχουν επίσης ορθότητα 10%. Αυτό υποδηλώνει ότι, αν και κάποιες μέθοδοι βελτιστοποίησης μπορούν να προσφέρουν αξιόπιστα αποτελέσματα, οι περισσότερες έχουν σημαντικά περιθώρια βελτίωσης ή ενδεχομένως είναι ακατάλληλες για ορισμένες εφαρμογές. Στο συγκεκριμένο σύνολο δεδομένων, ο αλγόριθμος Adamax είναι ο πιο αποδοτικός, επιτυγχάνοντας την υψηλότερη ορθότητα. Ωστόσο, η συνολική απόδοση των αλγορίθμων παραμένει χαμηλή, υποδεικνύοντας ότι τα δεδομένα πιθανώς δεν ταιριάζουν καλά με τις μεθόδους βελτιστοποίησης που χρησιμοποιήθηκαν.

Κατά την εκπαίδευση των αλγορίθμων για 25 εποχές, παρατηρούνται διαφορετικές συμπεριφορές στους αλγορίθμους.

Ο RMSprop εμφανίζει αυξομειώσεις τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης, αλλά συνολικά η ορθότητα αυξάνεται και η απώλεια μειώνεται.

Αντίθετα, οι αλγόριθμοι SGD και Adamax καταγράφουν ομαλή αύξηση της ορθότητας και μείωση της απώλειας στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης παρουσιάζουν μικρές αυξομειώσεις.

Οι αλγόριθμοι Adam, AdamW και Nadam εμφανίζουν συμπεριφορά παρόμοια με τον RMSprop.

Οι αλγόριθμοι Adadelta και Adagrad δείχνουν διακυμάνσεις, με αύξηση της ορθότητας και ομαλή μείωση της απώλειας σε σύνολο εκπαίδευσης και σύνολο επαλήθευσης.

Ο Adafactor παραμένει στάσιμος, με σταθερή ορθότητα στο σύνολο επαλήθευσης και διακυμάνσεις στο σύνολο εκπαίδευσης, όπου παρατηρείται μικρή αύξηση της ορθότητας, ενώ η απώλεια καταλήγει σε NaN.

Ο Ftrl παρουσιάζει διακυμάνσεις στο σύνολο εκπαίδευσης, με μικρή αύξηση της ορθότητας και πτώση στο σύνολο επαλήθευσης, όπου η ορθότητα σταθεροποιείται, ενώ η απώλεια διατηρείται σταθερή.

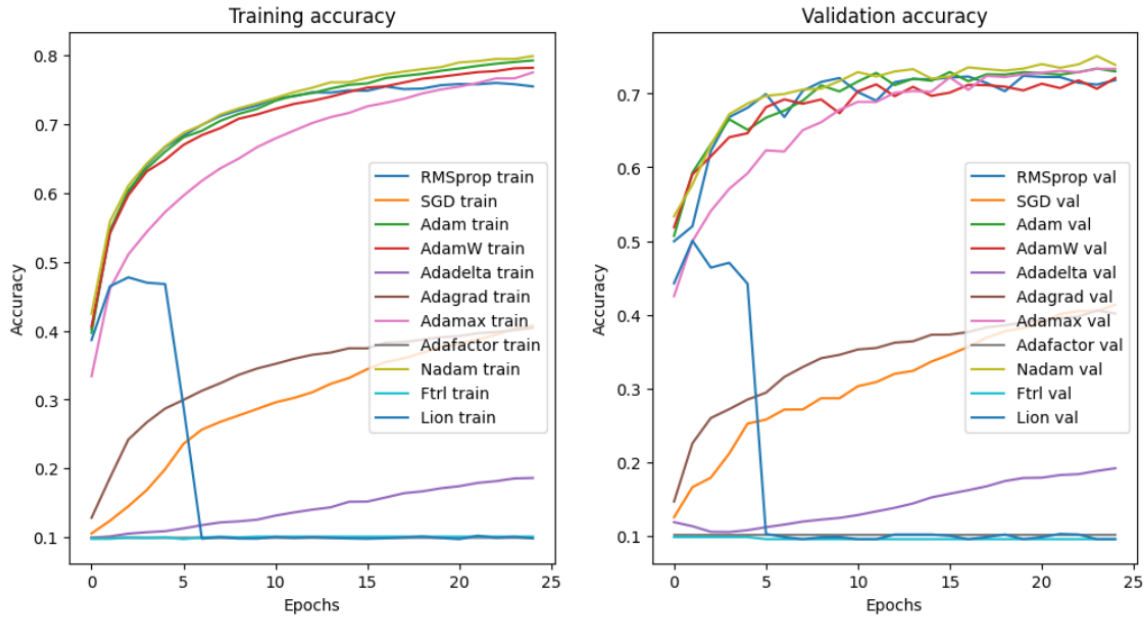
Ο Lion δείχνει αυξομειώσεις, με μείωση της ορθότητας και αύξηση της απώλειας τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης.

Στα ιστογράμματα του precision, recall και F1-score, οι περισσότερες κλάσεις στους RMSprop, Adam, AdamW, Adamax και Nadam έχουν τιμές κοντά στο 0.6. Οι αλγόριθμοι Adafactor, Ftrl και Lion αναγνωρίζουν μόνο μία κλάση, ενώ οι SGD, Adagrad και Adadelta επιτυγχάνουν τιμές κοντά στο 0.2.

Γενικά , στο σύνολο δεδομένων CIFAR-10, οι αλγόριθμοι Adam, AdamW, Adamax, και RMSprop καταγράφουν τις καλύτερες επιδόσεις, με ακρίβειες που κυμαίνονται γύρω στο 72%. Παρά την αρκετά καλή απόδοσή τους, παρατηρούνται διακυμάνσεις στην ορθότητα και την απώλεια στο σύνολο επαλήθευσης, κάτι που υποδηλώνει μικρή αστάθεια κατά την προσαρμογή σε νέα δεδομένα, ενδεχομένως λόγω υπερπροσαρμογής.

Από την άλλη, οι SGD, Adadelta, και Adagrad εμφανίζουν πιο σταθερές και ανθεκτικές συμπεριφορές, αλλά με χαμηλότερες ακρίβειες, γεγονός που τους καθιστά λιγότερο αποδοτικούς για το συγκεκριμένο σύνολο δεδομένων.

Η παρουσία των αλγορίθμων Adafactor, FTRL, και Lion, οι οποίοι αναγνωρίζουν μόνο μία κλάση και καταγράφουν ακρίβειες της τάξης του 10%, δείχνει την ακαταλληλότητά τους για τα χαρακτηριστικά και την πολυπλοκότητα του CIFAR-10.



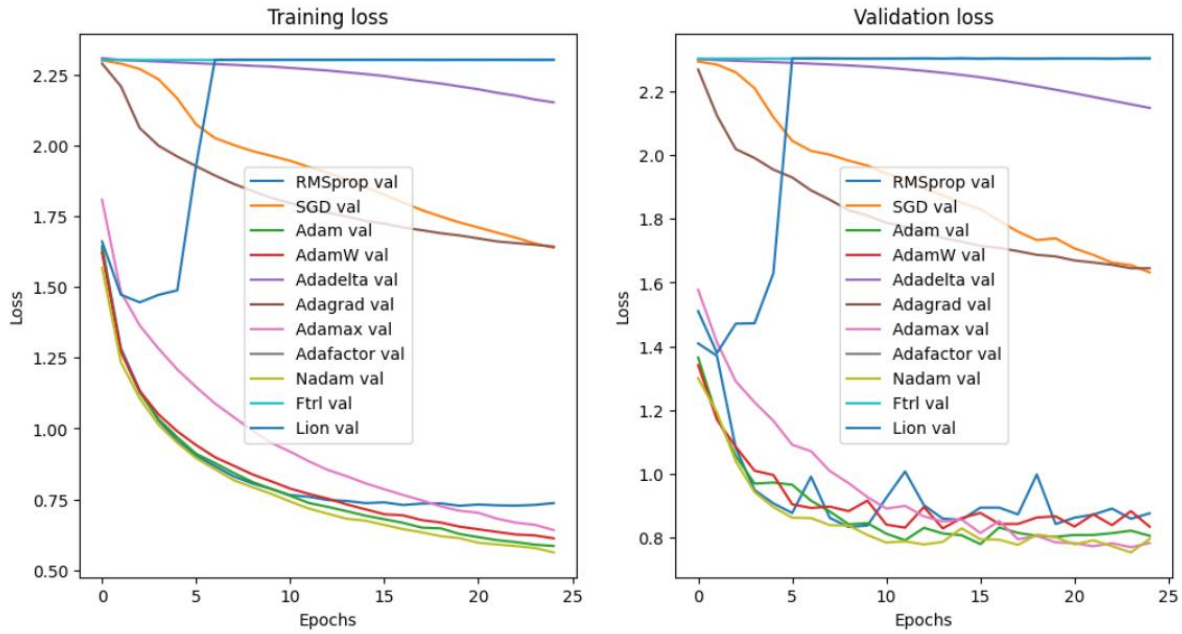
Εικόνα 16- CIFAR-10 Loss

Στις γραφικές παραστάσεις της ορθότητας κατά τη διάρκεια του σύνολο εκπαίδευσης και του σύνολο επαλήθευσης, παρατηρείται ότι οι περισσότεροι αλγόριθμοι ακολουθούν λογαριθμική αύξηση.

Αντίθετα, ο Adadelta διαφοροποιείται, ακολουθώντας μια σχετικά γραμμική πορεία.

Οι Ftrl και Adafactor εμφανίζονται ως γραμμές παράλληλες με την αρχή των αξόνων, υποδεικνύοντας σταθερότητα χωρίς καμία βελτίωση.

Ο Lion, ενώ αρχικά φαίνεται να ακολουθεί την πορεία των περισσότερων αλγορίθμων, τελικά παρουσιάζει μια πτώση, η οποία καταλήγει σε ευθυγράμμιση με τον άξονα των εποχών.



Εικόνα 17- CIFAR-10 Loss

Στις γραφικές παραστάσεις της απώλειας κατά τη διάρκεια του σύνολου εκπαίδευσης και σύνολο επαλήθευσης, οι περισσότεροι αλγόριθμοι παρουσιάζουν εκθετική μείωση με αυξομειώσεις.

Ο Adadelta ακολουθεί μια ομαλή πορεία, με αξιοσημείωτη πτώση μετά την 5η εποχή.

Αντίθετα, οι Ftrl και Adafactor παραμένουν σχεδόν παράλληλοι με τον άξονα των εποχών, διατηρώντας σταθερές τιμές για $y > 2.2$.

Ο Lion, σε αντίθεση με τους άλλους αλγόριθμους, παρουσιάζει αύξηση στην απώλεια, ακολουθούμενη από μια απότομη άνοδο, πριν τελικά ευθυγραμμιστεί με τον Ftrl κοντά στον άξονα των εποχών.

Ο πιο αποδοτικός αλγόριθμος είναι ο Adamax, καθώς επιτυγχάνει την υψηλότερη ορθότητα, φτάνοντας το 72.72%. Αυτό τον καθιστά τον καλύτερο αλγόριθμο σε σχέση με τους άλλους, με μικρές αυξομειώσεις στην απόδοση τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης, ενώ επιτυγχάνει υψηλά αποτελέσματα και στις μετρικές precision, recall και F1-score.

Σύνολο δεδομένων CIFAR-100

Στο σύνολο δεδομένων CIFAR-100, οι περισσότεροι αλγόριθμοι παρουσιάζουν χαμηλή απόδοση. Συγκεκριμένα, ο RMSprop επιτυγχάνει ορθότητα 36,46%, ο SDG μόλις 6,03%, ο Adam 35,03%, ο AdamW 36,6%, ο Adadelta 1,54%, ο Adagrad 7,51%, ο Adamax 38,11%, ο Adafactor 1%, ο Nadam 38,08%, ενώ οι Ftrl και Lion μόλις 1%. Τα αποτελέσματα αυτά δείχνουν ότι το νευρωνικό δίκτυο δεν μπορεί να προσαρμοστεί επαρκώς σε αυτό το σύνολο δεδομένων.

Η εκπαίδευση του μοντέλου σε 25 εποχές έδειξε ότι οι αλγόριθμοι RMSprop και Adamax παρουσιάζουν αυξομειώσεις στην απόδοση στο σύνολο επαλήθευσης, ενώ στο σύνολο εκπαίδευσης παρατηρείται ομαλή αύξηση της ορθότητας και μείωση της απώλειας.

Αντίθετα, οι αλγόριθμοι SGD και Adagrad εμφανίζουν διακυμάνσεις τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης, με αύξηση της ορθότητας και ομαλή μείωση της απώλειας και στα δύο.

Οι αλγόριθμοι Adam, AdamW και Nadam παρουσιάζουν αύξηση της ορθότητας με διακυμάνσεις σε σύνολο εκπαίδευσης και σύνολο επαλήθευσης, ενώ η απώλεια μειώνεται ομαλά στο σύνολο εκπαίδευσης, με αυξομειώσεις στο σύνολο επαλήθευσης.

Ο Adadelta εμφανίζει διακυμάνσεις στην ορθότητα και στα δύο σετ, με τη μείωση της απώλειας να είναι ασταθής στο σύνολο εκπαίδευσης αλλά πιο ομαλή στο σύνολο επαλήθευσης.

Ο Adafactor παρουσιάζει τιμές απώλειας NaN (Not a Number) τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης, με την ορθότητα να μειώνεται ελάχιστα και να παραμένει σταθερή στο σύνολο επαλήθευσης.

Ο αλγόριθμος FTRL επιδεικνύει χαμηλή ορθότητα, με ελάχιστη αύξηση στο σύνολο εκπαίδευσης και σταθεροποίηση στο σύνολο επαλήθευσης, ενώ η απώλεια παρουσιάζει διακυμάνσεις, με ελαφρά μείωση στο σύνολο εκπαίδευσης και μικρή αύξηση στο σύνολο επαλήθευσης.

Ο αλγόριθμος Lion παρουσιάζει έντονες διακυμάνσεις, με απότομη μείωση της ορθότητας και αύξηση της απώλειας σε σύνολο εκπαίδευσης και σύνολο επαλήθευσης.

Στα ιστογράμματα του precision, recall και F1-score, οι αλγόριθμοι RMSprop, Adam, AdamW, Adamax και Nadam εμφανίζουν διακυμάνσεις στις τιμές ανά κλάση, ενώ οι SGD, Adadelta και Adagrad αναγνωρίζουν ορισμένες κλάσεις με διακυμάνσεις ανάμεσά τους. Αντίθετα, οι Adafactor, FTRL και Lion αναγνωρίζουν μόνο μία κλάση, γεγονός που επηρεάζει αρνητικά την απόδοσή τους.

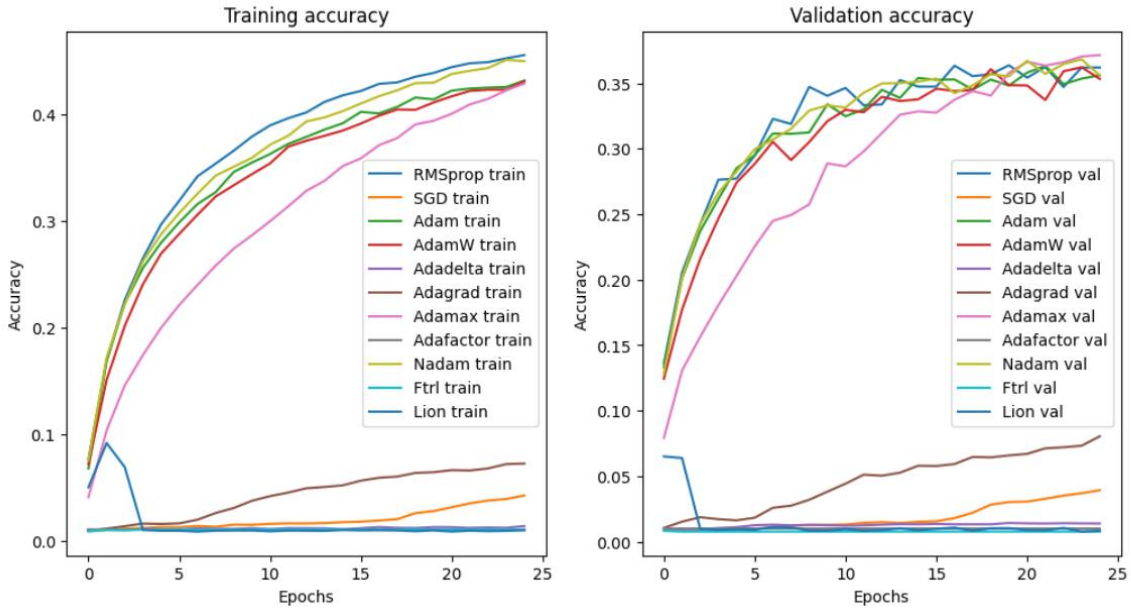
Γενικά, στο σύνολο δεδομένων CIFAR-100, η απόδοση των αλγορίθμων βελτιστοποίησης παραμένει σε χαμηλά επίπεδα, με ακρίβειες κάτω από 40%.

Οι αλγόριθμοι Adam, AdamW, Adamax και RMSprop καταγράφουν τις καλύτερες επιδόσεις, αλλά παρουσιάζουν έντονες διακυμάνσεις στο σύνολο επαλήθευσης, υποδηλώνοντας αδυναμία στη γενίκευση. Παρά την καλή απόδοση στο σύνολο εκπαίδευσης, η αστάθεια αυτή δείχνει δυσκολίες προσαρμογής στα νέα δεδομένα.

Από την άλλη πλευρά, οι SGD και Adagrad εμφανίζουν πιο σταθερή πορεία, αλλά με χαμηλότερες ακρίβειες σε σχέση με τους προηγούμενους αλγόριθμους, κάτι που τους καθιστά λιγότερο αποδοτικούς για το συγκεκριμένο σύνολο δεδομένων.

Τέλος, οι αλγόριθμοι Adafactor, FTRL, και Lion, οι οποίοι αναγνωρίζουν μόνο μία κλάση, καταγράφουν ακρίβειες της τάξης του 1%, υποδεικνύοντας σαφή αδυναμία να αντιμετωπίσουν την πολυπλοκότητα του CIFAR-100.

Συνολικά, οι αλγόριθμοι δείχνουν δυσκολίες στη γενίκευση και την προσαρμογή στο CIFAR-100, με τους πιο αποδοτικούς αλγόριθμους να παρουσιάζουν αστάθεια, ενώ οι πιο σταθεροί εμφανίζουν χαμηλότερες επιδόσεις.



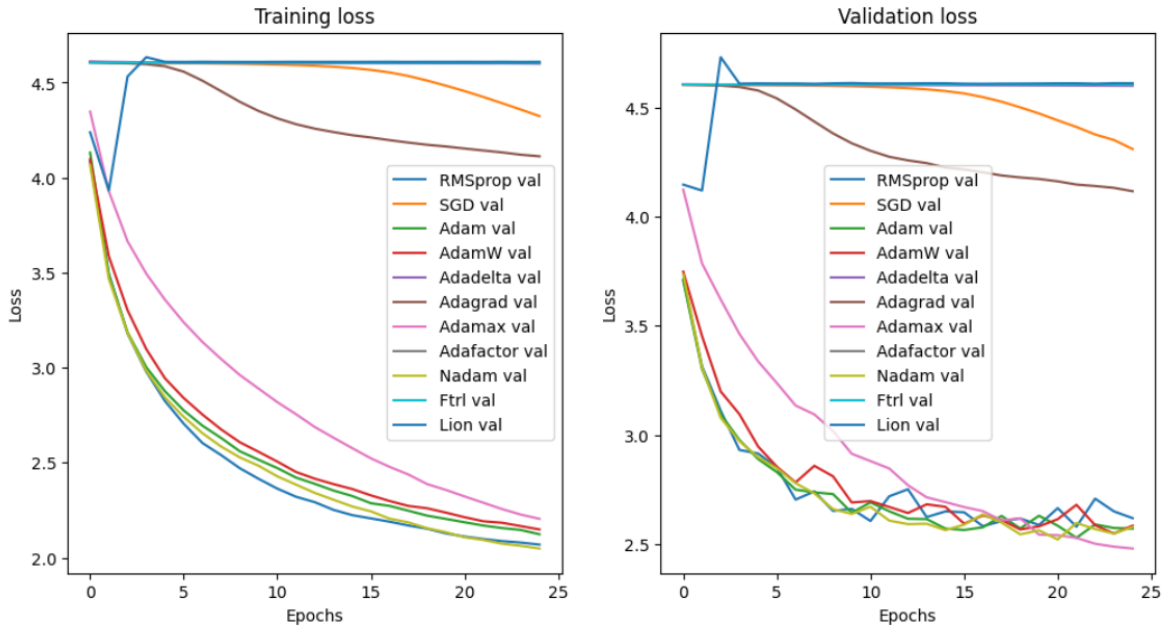
Εικόνα 18- CIFAR-10 Accuracy

Στις γραφικές παραστάσεις της ορθότητας στο σύνολο εκπαίδευσης και επικύρωσης για το σύνολο δεδομένων CIFAR-100, οι περισσότεροι αλγόριθμοι εμφανίζουν λογαριθμική αύξηση στην απόδοση, με το σύνολο επαλήθευσης να παρουσιάζει μεγαλύτερη διακύμανση στις τιμές.

Οι αλγόριθμοι SGD και Adagrad ακολουθούν μια συνεχή αυξητική πορεία στην ορθότητα.

Ο αλγόριθμος FTRL παρουσιάζει σχεδόν επίπεδη απόδοση, παραμένοντας παράλληλος με τον άξονα των εποχών, χωρίς αξιοσημείωτη πρόοδο.

Ο Lion, στην αρχή, δείχνει αύξηση στην απόδοση, αλλά στη συνέχεια η απόδοση πέφτει απότομα και η καμπύλη του γίνεται επίσης παράλληλη με τον άξονα των εποχών, υποδεικνύοντας στασιμότητα.



Εικόνα 19- CIFAR-100 Loss

Στις γραφικές παραστάσεις της ορθότητας στο σύνολο εκπαίδευσης και επικύρωσης, όπως και στο σύνολο δεδομένων CIFAR-100, οι περισσότεροι αλγόριθμοι ακολουθούν εκθετική μείωση της απώλειας, αν και με διακυμάνσεις στις τιμές της ορθότητας επικύρωσης.

Οι αλγόριθμοι SGD και Adagrad παρουσιάζουν μια πτωτική πορεία στην απώλεια.

Ο αλγόριθμος FTRL έχει καμπύλη που είναι σχεδόν παράλληλη με τον άξονα των εποχών, δείχνοντας χωρίς σημαντική πρόοδο για τιμές loss> 4.5.

Ο Lion, αρχικά, δείχνει μείωση στην απώλεια, αλλά στη συνέχεια η απώλεια αυξάνεται απότομα και η καμπύλη του γίνεται παράλληλη με τον άξονα των εποχών.

Ο πιο αποδοτικός αλγόριθμος είναι ο Adamax, με ορθότητα 38.11%. Παρουσιάζει τη μεγαλύτερη ορθότητα σε σύγκριση με τους υπόλοιπους αλγορίθμους, με σχετική σταθερότητα στην

εκπαίδευση και μικρότερες αυξομειώσεις στην ορθότητα και την απώλεια τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης. Ο Nadam, με ορθότητα 38.08%, είναι επίσης πολύ κοντά στην απόδοση του Adamax, αλλά ο Adamax έχει ελαφρώς καλύτερα αποτελέσματα.

Σύνολο δεδομένων Fashion mnist

Στο σύνολο δεδομένων Fashion MNIST, οι περισσότεροι αλγόριθμοι παρουσιάζουν καλή απόδοση. Συγκεκριμένα, ο RMSprop πέτυχε ορθότητα 88.94%, ο SGD 78.15%, ο Adam 91.25%, ο AdamW 91.03%, ο Adadelta 63.61%, ο Adagrad 78.24%, ο Adamax 91.34%, ο Adafactor 10%, ο Nadam 91.2%, ο Ftrl 10% και ο Lion 75.03%. Από τα παραπάνω, οι αλγόριθμοι Adam, AdamW, Adamax και Nadam ξεχωρίζουν με ακρίβειες άνω του 91%, ενώ οι Adafactor και Ftrl είχαν την χαμηλότερη απόδοση, με μόλις 10% ορθότητα.

Κατά την εκπαίδευση του μοντέλου για 25 εποχές, οι αλγόριθμοι RMSprop,Adam, AdamW, Adamax και Nadam παρουσίασαν αυξομειώσεις τόσο σε σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης, με την ορθότητα να αυξάνεται και την απώλεια να μειώνεται.

Ο SGD παρουσιάζει αύξηση της ορθότητας ομαλά στο σύνολο εκπαίδευσης και με μικρή διακύμανση στο σύνολο επαλήθευσης και μείωση της απώλειας με διακυμάνσεις στα σύνολα εκπαίδευσης και επαλήθευσης.

Οι αλγόριθμοι Adadelta και Adagrad εμφάνισαν επίσης αύξηση της ορθότητας ομαλά σε σύνολο εκπαίδευσης και με αυξομειώσεις σε σύνολο επαλήθευσης και μείωση της απώλειας ομαλά τόσο σε σύνολο εκπαίδευσης όσο και σε σύνολο επαλήθευσης .

Ο Adagrad παρουσίασε αύξηση της ορθότητας με διακυμάνσεις στο σύνολο εκπαίδευσης και ομαλά στο σύνολο επαλήθευσης, και ομαλή μείωσης της απώλειας σε σύνολο εκπαίδευσης και σύνολο επαλήθευσης.

Αντίθετα, ο Adafactor παρουσίασε μικρή αύξηση με διακυμάνσεις στο σύνολο εκπαίδευσης και διατήρησε σταθερά χαμηλή ορθότητα στο σύνολο επαλήθευσης και εμφάνισε NaN απώλεια και στα σύνολα εκπαίδευσης και επαλήθευσης , καθώς αναγνωρίζει μόνο μία κλάση.

Παρόμοια, ο Ftrl παρουσίασε μικρή μείωση στην απόδοση στο σύνολο εκπαίδευσης σταθερά χαμηλή απόδοση στο σύνολο επαλήθευσης με σταθερή απώλεια σε σύνολο εκπαίδευσης και σύνολο επαλήθευσης, επίσης αναγνωρίζοντας μόνο μία κλάση.

Ο Lion, από την άλλη, εμφάνισε διακυμάνσεις στην ορθότητα με αύξηση στο σύνολο εκπαίδευσης και μείωση στο σύνολο επαλήθευσης και αύξηση στην απώλεια κατά τη διάρκεια της εκπαίδευσης.

Στα ιστογράμματα των μετρικών precision, recall και F1-score, οι αλγόριθμοι RMSprop, Adam, AdamW, Adamax και Nadam πέτυχαν τιμές άνω του 0.8 για τις περισσότερες κλάσεις, ενώ οι SGD, Adadelta, Adagrad και Lion πέτυχαν τιμές άνω του 0.6.

Τέλος, οι αλγόριθμοι Adafactor και Ftrl αναγνώριζαν μόνο μία κλάση, επιβεβαιώνοντας τη χαμηλή τους απόδοση.

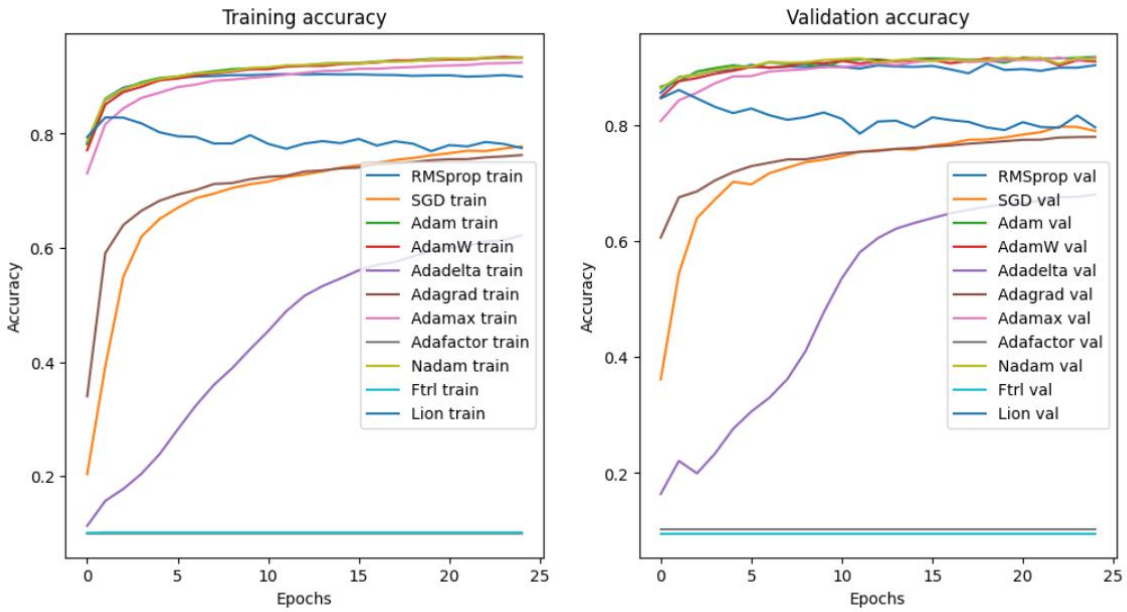
Γενικά, στο σύνολο δεδομένων Fashion MNIST, όπως και στο MNIST, οι περισσότεροι αλγόριθμοι επιτυγχάνουν υψηλές τιμές ορθότητας.

Οι αλγόριθμοι Adam, AdamW, Adamax και RMSprop καταγράφουν τις καλύτερες επιδόσεις, με ακρίβειες πάνω από 88%, παρά τις αυξομειώσεις που παρατηρούνται στο σύνολο επαλήθευσης. Αυτές οι μικρές διακυμάνσεις δεν επηρεάζουν δραματικά τη συνολική τους απόδοση, και οι συγκεκριμένοι αλγόριθμοι φαίνεται να προσαρμόζονται καλύτερα στα δεδομένα.

Οι SGD, Adadelta και Adagrad επίσης παρουσιάζουν καλές αποδόσεις, αν και οι αλγόριθμοι αυτοί συγκλίνουν πιο αργά προς υψηλά επίπεδα ορθότητας, σε σύγκριση με τους Adam, AdamW και Adamax. Η σταθερότητά τους, ωστόσο, στην απώλεια και την ορθότηταδείχνει ότι προσφέρουν μεγαλύτερη ανθεκτικότητα και καλύτερη γενίκευση σε νέα δεδομένα.

Από την άλλη πλευρά, οι αλγόριθμοι Adafactor και FTRL παρουσιάζουν τις χαμηλότερες επιδόσεις, με μόλις 10% ορθότητα, καθώς αναγνωρίζουν μόνο μία κλάση, γεγονός που τους καθιστά λιγότερο αποδοτικούς στο συγκεκριμένο σύνολο δεδομένων.

Ο Lion, αν και πιο ασταθής, κατάφερε να επιτύχει ορθότητα 75.03%, αλλά οι διακυμάνσεις στην απόδοσή του και η συνολική αύξηση της απώλειας τον καθιστούν λιγότερο αξιόπιστο σε σχέση με τους υπόλοιπους αλγόριθμους.



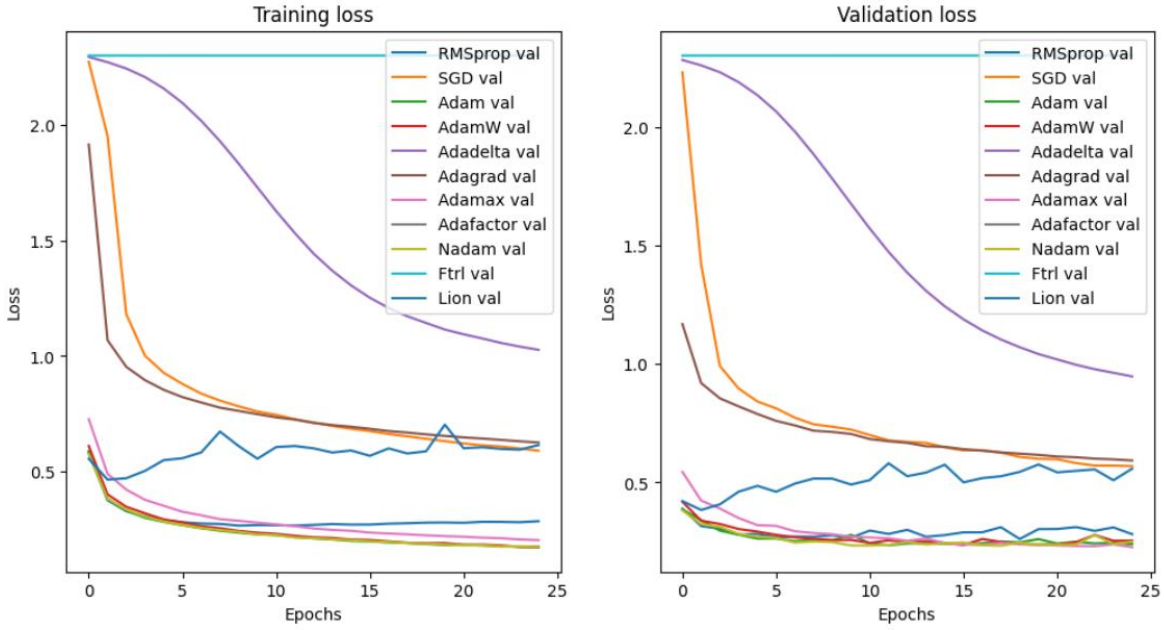
Εικόνα 20- Fashion MNIST- Accuracy

Στις γραφικές παραστάσεις ορθότητας του σύνολο εκπαίδευσης και του σύνολο επαλήθευσης, οι περισσότεροι αλγόριθμοι ακολουθούν εκθετική αύξηση, με την καμπύλη της ορθότητας του συνόλου επαλήθευσης να παρουσιάζει περισσότερες αυξομειώσεις σε σύγκριση με τη ορθότητα του συνόλου εκπαίδευσης.

Αντίθετα, οι αλγόριθμοι Adafactor και Ftrl εμφανίζουν γραμμές παράλληλες προς τον άξονα των εποχών, δείχνοντας σταθερή, χαμηλή απόδοση.

Ο Adadelta προσομοιάζει μια γραμμική σχέση, όπου η ορθότητα αυξάνεται.

Ο Lion, τέλος, παρουσιάζει διάφορες διακυμάνσεις, με συνολική τάση μείωσης στην απόδοση.



Εικόνα 21- Fashion MNIST Loss

Στις γραφικές παραστάσεις της απώλειας του σύνολο εκπαίδευσης και του συνόλου επαλήθευσης, οι περισσότεροι αλγόριθμοι ακολουθούν μια εκθετική μείωση, αν και παρατηρούνται μικρές διακυμάνσεις στις τιμές τους.

Ο Adadelta εμφανίζει μια σιγμοειδή καμπύλη με συνολική πτωτική πορεία.

Ο Ftrl παρουσιάζει μια σχεδόν σταθερή γραμμή, παράλληλη προς την αρχή των αξόνων, με τιμές πάνω από το 2.

Τέλος, ο Lion εμφανίζει πολλές αυξομειώσεις με συνολική τάση αύξησης του loss.

Ο πιο αποδοτικός αλγόριθμος για το σύνολο δεδομένων Fashion MNIST είναι ο Adamax, με ορθότητα 91.34%. Ο Adamax ξεχωρίζει καθώς επιτυγχάνει την υψηλότερη ορθότητα μεταξύ όλων

των αλγορίθμων και παράλληλα παρουσιάζει σταθερότητα στην εκπαίδευση, με μικρότερες αυξομειώσεις στην ορθότητα και την απώλεια σε σύγκριση με άλλους αλγόριθμους.

Σύνολο δεδομένων IMDB

Στο σύνολο δεδομένων IMDB, όλοι οι αλγόριθμοι παρουσιάζουν ορθότητα 0.5, αναγνωρίζοντας μόνο τη μία κλάση. Αυτό υποδηλώνει ότι το convolutional neural network αδυνατεί να επεξεργαστεί αποτελεσματικά το συγκεκριμένο σύνολο δεδομένων.

Κατά την εκπαίδευση του μοντέλου για 25 εποχές, παρατηρήθηκε ότι οι αλγόριθμοι RMSprop, Adam, Ftrl και Lion παρουσίασαν αυξομειώσεις στην απόδοση, με την ορθότητα στο σύνολο επαλήθευσης να αυξάνεται ελαφρώς, ενώ η απώλεια σημείωσε άνοδο.

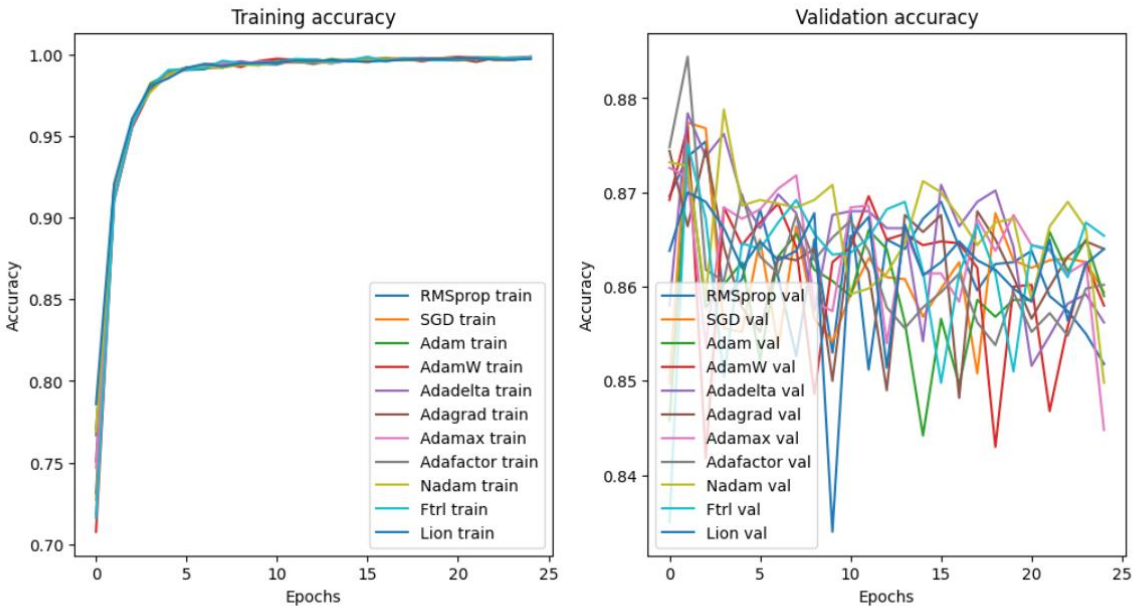
Αντίθετα, οι αλγόριθμοι SGD, AdamW, Adadelta, Adagrad, Adamax, Adafactor και Nadam εμφάνισαν μια μικρή μείωση στην ορθότητα στο σύνολο επαλήθευσης και ταυτόχρονα αύξηση στην απώλεια.

Στο σύνολο εκπαίδευσης, η ορθότητα αυξήθηκε για όλους τους αλγόριθμους, ενώ η απώλεια μειώθηκε, γεγονός που υποδηλώνει ότι το μοντέλο υπερ εκπαιδεύεται (overfitting).

Το precision ήταν κοντά στο 0.5, το recall πλησίασε το 1 και το F1-score κυμάνθηκε περίπου στο 0.7.

Στο σύνολο δεδομένων IMDB, όλοι οι αλγόριθμοι εμφανίζουν παρόμοια συμπεριφορά, παρουσιάζοντας μείωση της ορθότητας στο σύνολο επαλήθευσης και ταυτόχρονα αύξηση της απώλειας. Παρά τις διαφορές τους σε άλλες περιπτώσεις, όπως σε άλλα σύνολα δεδομένων, σε αυτήν την περίπτωση κανένας αλγόριθμος δεν φαίνεται να καταφέρνει να αποδώσει ικανοποιητικά. Η σταθερή τάση μείωσης της ορθότητας και αύξησης της απώλειας υποδηλώνει ένα γενικευμένο πρόβλημα υπερεκπαίδευσης (overfitting) που εμποδίζει τη σωστή γενίκευση στα νέα δεδομένα του συνόλου επαλήθευσης.

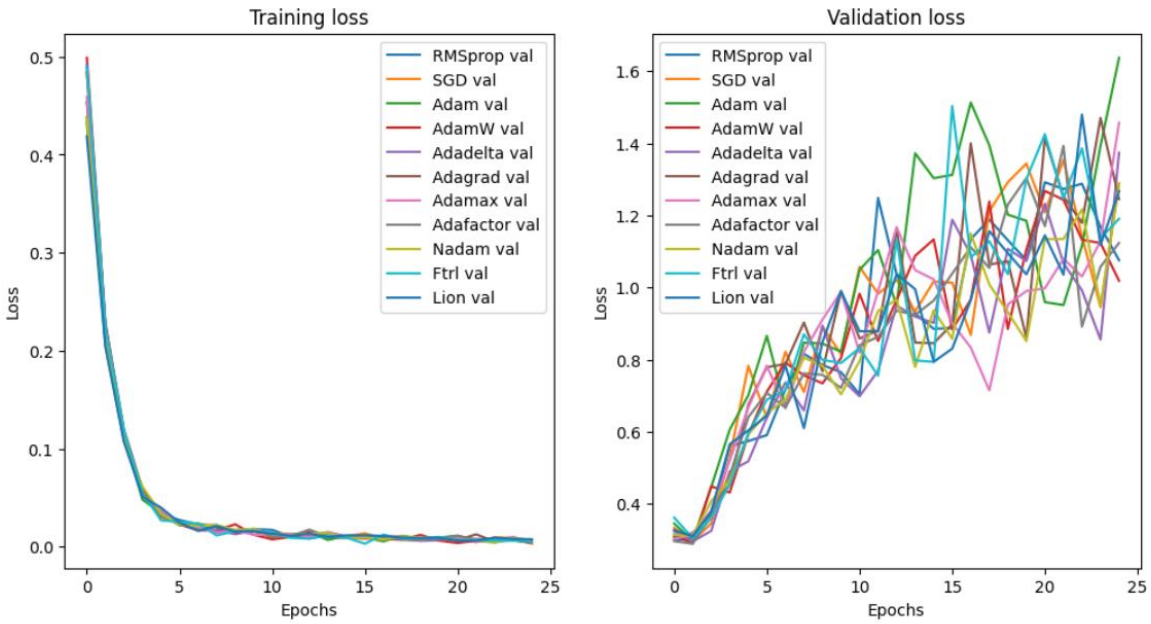
Ακόμα και αλγόριθμοι που συνήθως αποδίδουν καλά, όπως ο RMSprop, Adam, και AdamW, δεν καταφέρνουν να αποφύγουν τη μείωση της ορθότητας. Αυτή η ομοιογενής συμπεριφορά όλων των αλγορίθμων, με κοινά προβλήματα στην επαλήθευση, υποδηλώνει ότι το convolutional neural network (CNN) που χρησιμοποιήθηκε για το IMDB dataset δεν είναι αποτελεσματικό στην επεξεργασία του συγκεκριμένου συνόλου δεδομένων.



Εικόνα 22- IMDB- Accuracy

Στη γραφική παράσταση του σύνολο εκπαίδευσης accuracy, οι αλγόριθμοι παρουσιάζουν λογαριθμική αύξηση, υποδεικνύοντας σταθερή βελτίωση της ορθότητας κατά τη διάρκεια της εκπαίδευσης.

Στη γραφική παράσταση της ακριβειας του συνόλου επαλήθευσης, ενώ αρχικά παρατηρείται μια αυξητική πορεία, μετά το σημείο 0.87 η πορεία γίνεται πτωτική με έντονες διακυμάνσεις. Αυτές οι διακυμάνσεις είναι ιδιαίτερα έντονες στους αλγόριθμους SGD και Nadam.



Εικόνα 23- IMDB-Loss

Στις γραφικές παραστάσεις απώλειας του σύνολο εκπαίδευσης και του συνόλου επαλήθευσης, παρατηρείται αυξητική πορεία με διακυμάνσεις για όλους τους αλγόριθμους.

Οι διακυμάνσεις αυτές είναι ιδιαίτερα έντονες στον αλγόριθμο Adam, ο οποίος παρουσιάζει τις μεγαλύτερες αποκλίσεις σε σχέση με τους υπόλοιπους. Αυτό υποδηλώνει ότι το μοντέλο αντιμετωπίζει δυσκολίες στη γενίκευση και πιθανώς υπερεκπαιδεύεται, καθώς η απώλεια αυξάνεται τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης.

Συνολικά, κανένας αλγόριθμος δεν υπερέχει σταθερά στο σύνολο δεδομένων IMDB, καθώς όλοι εμφανίζουν προβλήματα γενίκευσης και overfitting.

Παρόλο που οι αλγόριθμοι RMSprop και Adam παρουσιάζουν μια ελαφριά βελτίωση στην ορθότητα του σύνολο επαλήθευσης, δεν αποτελούν ιδανική λύση.

Οι υπόλοιποι αλγόριθμοι επίσης αντιμετωπίζουν παρόμοιες δυσκολίες, με αποτέλεσμα να αναγνωρίζουν μόνο μία κλάση και να παρουσιάζουν ορθότητα 0.5.

Αυτό φανερώνει ότι το convolutional neural network (CNN) αδυνατεί να επεξεργαστεί αποτελεσματικά το συγκεκριμένο σύνολο δεδομένων, περιορίζοντας την ικανότητα των αλγορίθμων να γενικεύσουν.

Σύνολο δεδομένων Reuters

Στο σύνολο δεδομένων Reuters, οι περισσότεροι αλγόριθμοι επιτυγχάνουν ορθότητα πάνω από 67%. Συγκεκριμένα, ο RMSprop σημειώνει ορθότητα 68.38%, ο SGD 69.27%, ο Adam 68.65%, ο AdamW 68.87%, ο Adadelta 68.38%, ο Adagrad 69.59%, ο Adamax 67.72%, ο Adafactor 67.14%, ο Nadam 67.94%, ο Ftrl 68.69% και ο Lion 68.78%.

Η εκπαίδευση αλγορίθμων για 25 εποχές με τον Rmsprop παρουσιάζει αυξομειώσεις, όπου η ορθότητα αυξάνεται και η απώλεια μειώνεται στο σύνολο εκπαίδευσης, αλλά στο σύνολο επαλήθευσης παρατηρείται το αντίθετο φαινόμενο.

Το ίδιο μοτίβο παρατηρείται και στους αλγορίθμους SGD, Adam, AdamW, Adadelta, Adagrad, Adamax, Adafactor, Nadam, Ftrl, και Lion, όπου η ορθότητα αυξάνεται στο σύνολο εκπαίδευσης και η απώλεια μειώνεται, ενώ στο σύνολο επαλήθευσης η ορθότητα αυξάνεται με την απώλεια να αυξάνεται σημαντικά, με διακυμάνσεις και στα δύο σετ. Αυτή η συμπεριφορά υποδηλώνει ότι το μοντέλο κάνει overfitting.

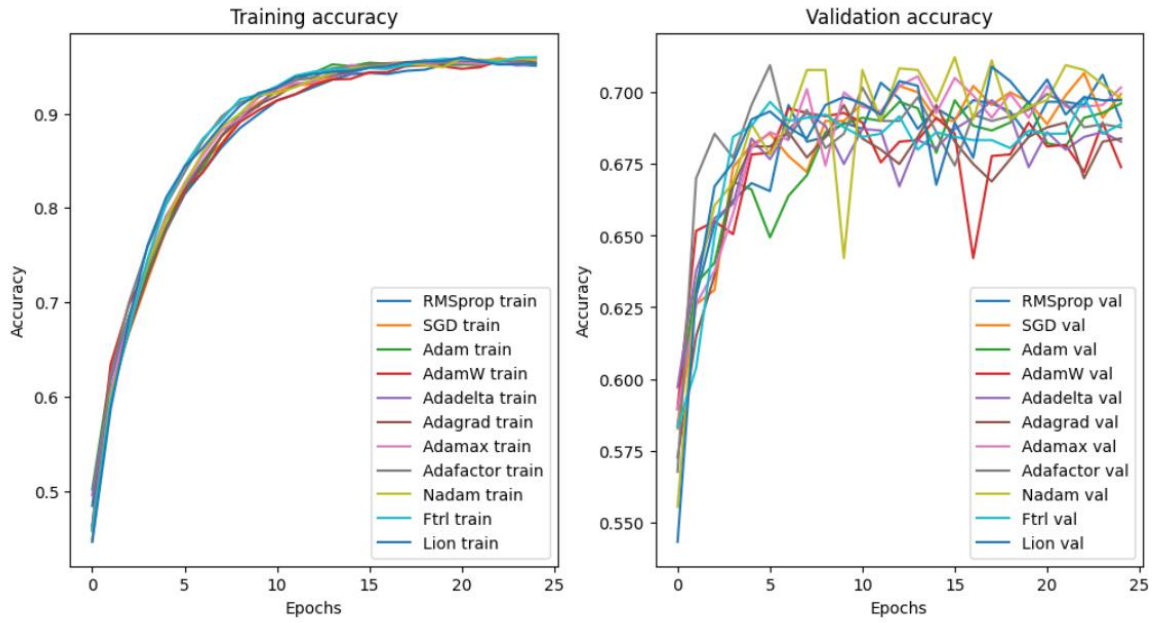
Στα ιστογράμματα των precision, recall και F1-score παρατηρούνται διακυμάνσεις, με τις περισσότερες κλάσεις να ξεπερνούν το 0.4, ενώ κάποιες κλάσεις δεν αναγνωρίζονται καθόλου. Στο σύνολο δεδομένων Reuters, παρατηρείται μια παρόμοια συμπεριφορά με το σύνολο IMDB, αν και οι αποδόσεις είναι ελαφρώς υψηλότερες, φτάνοντας μέχρι και το 70%.

Όλοι οι αλγόριθμοι εμφανίζουν μια γενική μείωση της ορθότητας στο σύνολο επαλήθευσης, ενώ η απώλεια αυξάνεται, κάτι που υποδηλώνει επίσης προβλήματα υπερεκπαίδευσης (overfitting). Παρά τις διαφορές στις αρχικές επιδόσεις, η συμπεριφορά τους παραμένει ομοιόμορφη, με την ορθότητα να μειώνεται σταδιακά κατά την επαλήθευση των δεδομένων.

Οι αλγόριθμοι Adam, AdamW, Nadam, RMSprop και Adamax, που συνήθως αποδίδουν καλά σε άλλα datasets, όπως το MNIST και το Fashion MNIST, δεν καταφέρνουν να αποφύγουν αυτή τη μείωση στην απόδοση.

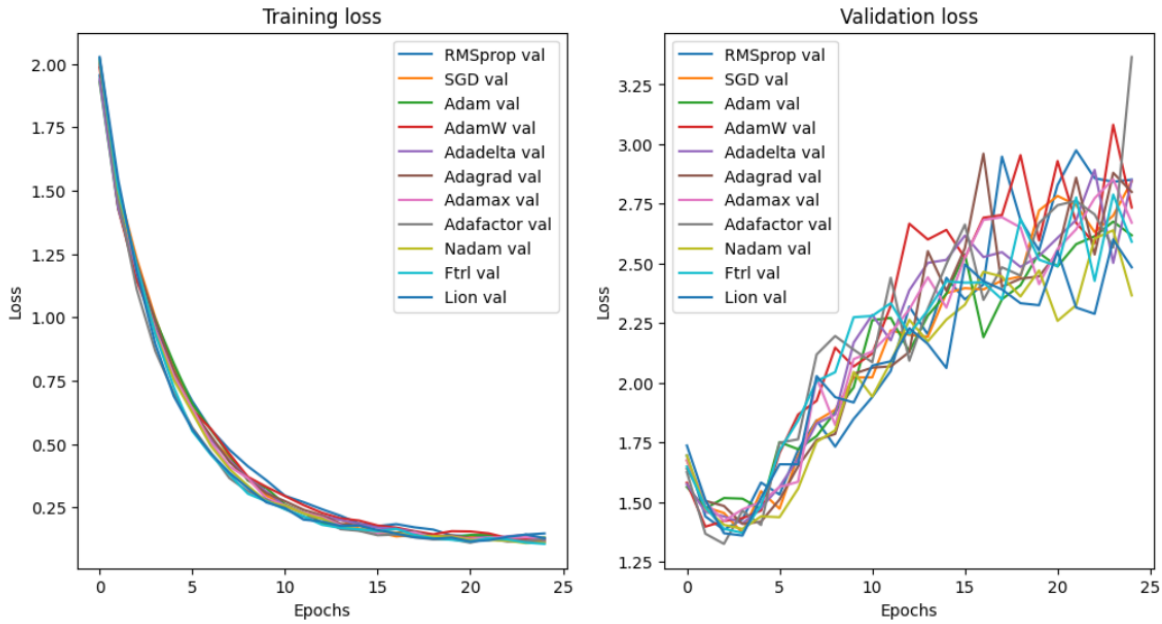
Οι SGD, Adadelta και Adagrad, παρόλο που παρουσιάζουν πιο σταθερή προσέγγιση σε άλλα datasets, εδώ αντιμετωπίζουν παρόμοιες δυσκολίες, με τη συμπεριφορά τους να μην διαφέρει σημαντικά από τους πιο εξελιγμένους αλγόριθμους.

Η κοινή συμπεριφορά όλων των αλγορίθμων στο σύνολο δεδομένων Reuters δείχνει ότι οι μέθοδοι που χρησιμοποιήθηκαν μπορεί να μην είναι οι καταλληλότερες για αυτό το συγκεκριμένο dataset, καθώς όλοι αντιμετωπίζουν δυσκολίες στη γενίκευση των αποτελεσμάτων.



Εικόνα 24- Reuters-Accuracy

Στις γραφικές παραστάσεις της ορθότητας του σύνολο εκπαίδευσης και του συνόλου επαλήθευσης, οι αλγόριθμοι παρουσιάζουν λογαριθμική αύξηση, με την ορθότητα του συνόλου επαλήθευσης να εμφανίζει μεγαλύτερες διακυμάνσεις στις τιμές.



Εικόνα 25- Reuters-Loss

Στις γραφικές παραστάσεις της απώλειας του σύνολο επαλήθευσης, παρατηρείται αρχικά μείωση, αλλά στη συνέχεια η καμπύλη αποκτά παραβολική μορφή που εξελίσσεται σε ευθεία γραμμή με αυξητική τάση και διακυμάνσεις στις τιμές.

Ο αλγόριθμος Adagrad φαίνεται να είναι ο πιο αποδοτικός για το σύνολο δεδομένων Reuters, καθώς επιτυγχάνει την υψηλότερη ορθότητα, με ποσοστό 69.59%.

Παράλληλα, ο αλγόριθμος SGD ακολουθεί με ορθότητα 69.27%, υποδεικνύοντας ότι είναι επίσης αρκετά αποδοτικός.

Ωστόσο, παρατηρείται ότι οι περισσότεροι αλγόριθμοι, συμπεριλαμβανομένων του RMSprop, Adam, AdamW, και άλλων, παρουσιάζουν φαινόμενα overfitting, με την ορθότητα να αυξάνεται στο σύνολο εκπαίδευσης και την απώλεια να μειώνεται, αλλά στο σύνολο επαλήθευσης να παρουσιάζεται αντίθετη τάση (αύξηση της απώλειας). Αυτό σημαίνει ότι ενώ οι αλγόριθμοι αποδίδουν καλά στο σύνολο εκπαίδευσης, δεν γενικεύουν το ίδιο καλά στο σύνολο επαλήθευσης. Συνολικά, με βάση την ορθότητα, ο Adagrad φαίνεται να είναι ο πιο αποδοτικός αλγόριθμος για

το συγκεκριμένο πρόβλημα, παρότι το overfitting παραμένει πρόκληση που πρέπει να αντιμετωπιστεί.

Σύνολο δεδομένων California Housing

Για την αξιολόγηση της απόδοσης των μοντέλων σε ένα σύνολο δεδομένων παλινδρόμησης, χρησιμοποιήθηκε ως μέτρηση το μέσο τετραγωνικό σφάλμα (Mean Square Error - MSE).

Τα αποτελέσματα έδειξαν ότι ο βελτιστοποιητής RMSprop πέτυχε την καλύτερη απόδοση με απώλεια 0.55, ενώ οι υπόλοιποι βελτιστοποιητές παρουσίασαν τις εξής απώλειες: ο Adam είχε MSE 0.56, ο AdamW 0.56, ο Adadelta 0.56, ο Ftrl 0.59, και ο Nadam 0.811. Αντίθετα, ο SGD είχε MSE 1.04, ο Adamax 3.67, ο Lion 5.54, και ο Adagrad 5.97, παρουσιάζοντας χειρότερη απόδοση σε σύγκριση με τους άλλους αλγόριθμους.

Κατά την εκπαίδευση του μοντέλου για 25 επαναλήψεις, οι αλγόριθμοι RMSprop, Adam, AdamW, Nadam και Lion παρουσίασαν μείωση του μέσου τετραγωνικού σφάλματος με διακυμάνσεις τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης.

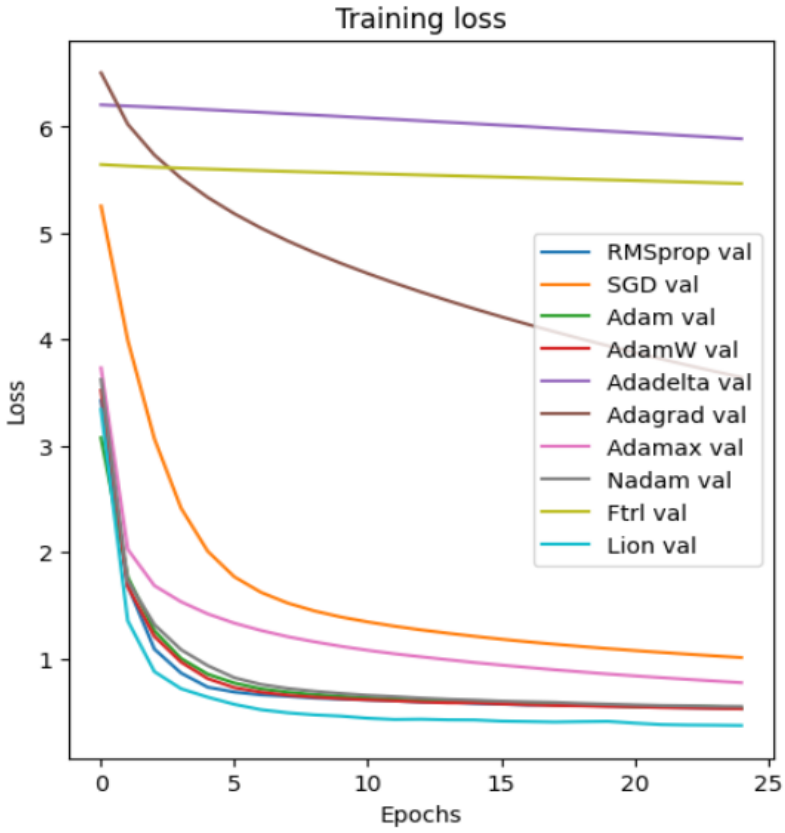
Από την άλλη πλευρά, οι αλγόριθμοι SGD, Adadelta, Adagrad, Adamax και Ftrl παρουσίασαν επίσης μείωση του τετραγωνικού σφάλματος, με διακυμάνσεις κυρίως στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης η μείωση ήταν πιο ομαλή.

Στο σύνολο δεδομένων California Housing, οι περισσότεροι αλγόριθμοι παρουσιάζουν σταθερή μείωση του μέσου τετραγωνικού σφάλματος (MSE), χωρίς ιδιαίτερες διακυμάνσεις.

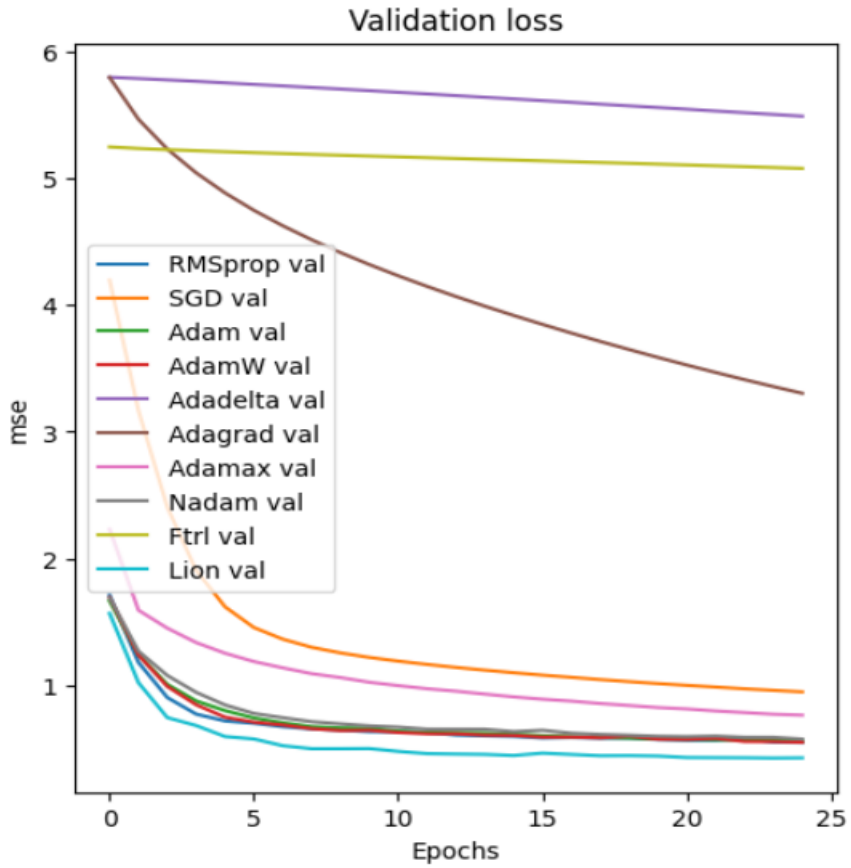
Αλγόριθμοι όπως οι Adam, AdamW, RMSprop, Adamax και Nadam, παρά το γεγονός ότι ξεκινούν με υψηλότερες τιμές MSE, καταφέρνουν να μειώσουν σταδιακά το σφάλμα τους, καταλήγοντας σε χαμηλές τιμές. Αυτή η συμπεριφορά υποδεικνύει ότι προσαρμόζονται αποτελεσματικά στο συγκεκριμένο πρόβλημα παλινδρόμησης, παρέχοντας σταθερή βελτίωση κατά τη διάρκεια της εκπαίδευσης.

Αντίθετα, οι αλγόριθμοι Adadelta, Adagrad και FTRL διατηρούν σταθερά υψηλά επίπεδα MSE, γεγονός που δείχνει ότι δεν καταφέρνουν να προσαρμοστούν ικανοποιητικά στις απαιτήσεις του συνόλου δεδομένων. Αυτό σημαίνει ότι δεν καταφέρνουν να βελτιώσουν τη συνολική απόδοση του μοντέλου, υποδεικνύοντας την ακαταλληλότητά τους για αυτό το πρόβλημα.

Ο αλγόριθμος Lion, όπως καταγράφεται στα διαγράμματα, εμφανίζει επίσης σταθερή μείωση του MSE χωρίς διακυμάνσεις, καταλήγοντας σε χαμηλά επίπεδα. Αυτό υποδηλώνει ότι είναι από τους πιο αποδοτικούς αλγόριθμους για το συγκεκριμένο σύνολο δεδομένων, επιτυγχάνοντας υψηλή ορθότητα στις προβλέψεις του.



Εικόνα 26- Training MSE



Εικόνα 27-Validation MSE

Στις γραφικές παραστάσεις των συνόλων εκπαίδευσης και επαλήθευσης, οι περισσότεροι αλγόριθμοι παρουσιάζουν εκθετική μείωση του σφάλματος.

Ο Adagrad εμφανίζει τη μικρότερη μείωση, ενώ οι αλγόριθμοι Adadelta και Ftrl δείχνουν γραμμική μείωση, με τις καμπύλες τους να είναι σχεδόν παράλληλες προς τον άξονα των εποχών.

Ο πιο αποδοτικός αλγόριθμος για αυτό το σύνολο δεδομένων παλινδρόμησης California-Housing είναι ο RMSprop, καθώς πέτυχε τη χαμηλότερη τιμή του μέσου τετραγωνικού σφάλματος (MSE = 0.55). Οι αλγόριθμοι Adam και AdamW παρουσίασαν παρόμοια απόδοση με το MSE τους να ανέρχεται στο 0.56, κάτι που δείχνει ότι είναι επίσης αρκετά αποδοτικοί. Συνολικά, η χρήση του RMSprop φαίνεται να προσφέρει την καλύτερη ισορροπία μεταξύ μείωσης του σφάλματος και σταθερότητας στις επαναλήψεις, καθιστώντας τον την καλύτερη επιλογή σε αυτό το σύνολο δεδομένων.

4.3. Αναδρομικά Νευρωνικά Δίκτυα (Recurrent Neural Networks)

Σύνολο δεδομένων Mnist

Κατά την εκπαίδευση του μοντέλου για 25 εποχές στο σύνολο δεδομένων MNIST, οι αλγόριθμοι βελτιστοποίησης παρουσιάζουν μια διχασμένη απόδοση: κάποιοι επιτυγχάνουν υψηλή ορθότητα, ενώ άλλοι έχουν πολύ χαμηλή. Συγκεκριμένα, ο RMSprop επιτυγχάνει ορθότητα 94.92%, ο SGD 96.05%, ο Adam 97.26%, ο AdamW 97.61%, ο Adadelta 64.47%, ο Adagrad 92.81%, ο Adamax 98.05%, ο Adafactor 9.8%, ο Nadam 97.65%, ο Ftrl 11.35% και ο Lion 9.8%.

Στο σύνολο εκπαίδευσης και στο σύνολο επαλήθευσης, ο RMSprop παρουσιάζει διακυμάνσεις κατά την αύξηση της ορθότητας, με απότομες αυξομειώσεις στην απώλεια.

Οι SGD, Adam, Adamax και Nadam επιδεικνύουν σταθερή αύξηση στην ορθότητα με μικρές διακυμάνσεις και συνεχή μείωση της απώλειας.

Ο AdamW παρουσιάζει αύξηση της ορθότητας με ομαλές διακυμάνσεις και μείωση της απώλειας με διακυμάνσεις τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης.

Ο Adadelta επιτυγχάνει σταθερή αύξηση στην ορθότητα και μείωση της απώλειας.

Ο Adagrad δείχνει μικρές διακυμάνσεις στην ορθότητα και μείωση της απώλειας με σταθερή απόδοση και στα δύο σύνολα δεδομένων.

Ο Adafactor παρουσιάζει μικρή αύξηση στην ορθότητα κατά την εκπαίδευση, με απώλεια που παραμένει NaN, υποδεικνύοντας αδυναμία μάθησης του μοντέλου.

Ο Ftrl παρουσιάζει μικρή βελτίωση στην ορθότητα στο σύνολο εκπαίδευσης ενώ στο σύνολο επαλήθευσης παραμένει σταθερή με ελαφρά μείωση στην απώλεια.

Ο Lion αρχικά παρουσιάζει υψηλή ορθότητα, η οποία μειώνεται απότομα και η απώλεια καταλήγει σε NaN, γεγονός που δηλώνει αποτυχία εκπαίδευσης.

Όσον αφορά την ανάλυση των metrics (precision, recall και F1-score), οι αλγόριθμοι Adafactor, Ftrl και Lion αναγνωρίζουν μόνο μία κλάση, παρουσιάζοντας χαμηλές επιδόσεις. Οι RMSprop, SGD, Adam, AdamW, Adagrad και Adamax επιτυγχάνουν τιμές κοντά στο 1 σε όλες τις μετρικές.

Ο Adadelta παρουσιάζει σημαντικές διακυμάνσεις μεταξύ των κλάσεων.

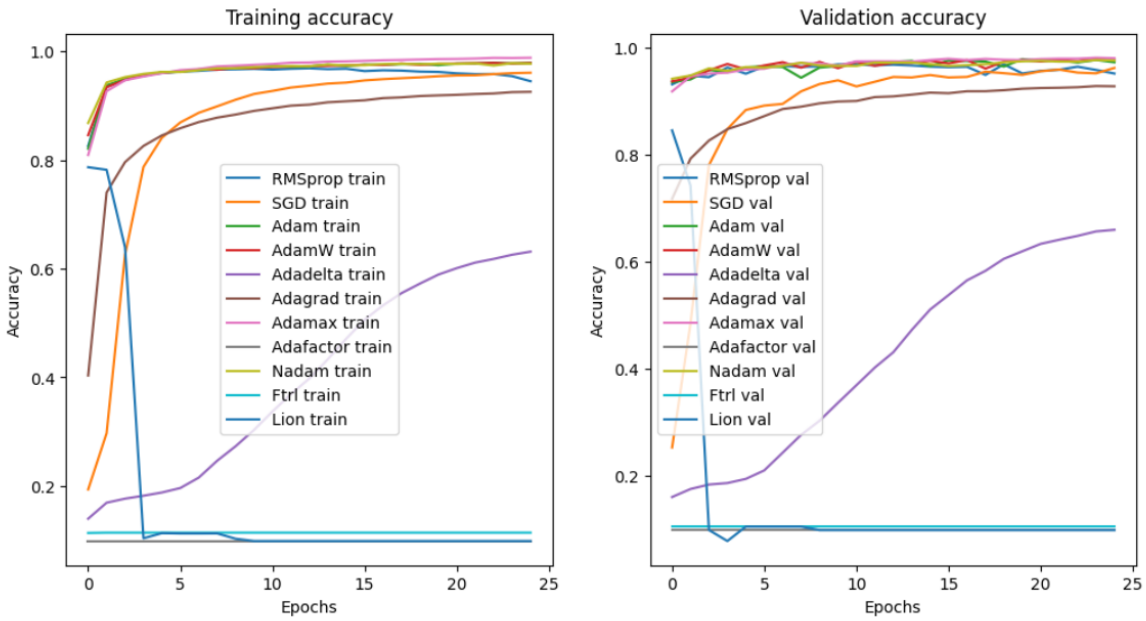
Γενικά, όπως και στις προηγούμενες αρχιτεκτονικές, οι αλγόριθμοι βελτιστοποίησης εμφανίζουν πολύ καλές αποδόσεις, στο σύνολο δεδομένων MNIST.

Οι Adam, AdamW, RMSprop, Adamax και Nadam καταγράφουν τις υψηλότερες ακρίβειες και επιτυγχάνουν μείωση της απώλειας, αν και παρατηρούνται διακυμάνσεις στο σύνολο επαλήθευσης, κάτι που υποδεικνύει πιθανή αστάθεια κατά την προσαρμογή σε νέα δεδομένα. Παρ' όλα αυτά, η συνολική τους απόδοση παραμένει ισχυρή.

Ακολουθούν οι SGD και Adagrad, με χαμηλότερες επιδόσεις, αλλά παρουσιάζουν μεγαλύτερη σταθερότητα, ιδιαίτερα στην ικανότητα γενίκευσης.

Ο Adadelta, με ορθότητα 64.47%, δείχνει δυσκολία στην προσαρμογή στα δεδομένα, έχοντας χαμηλότερη απόδοση συγκριτικά με τους άλλους αλγόριθμους.

Τέλος, οι FTRL, Adafactor, και Lion καταγράφουν τις χαμηλότερες επιδόσεις, με σημαντικές δυσκολίες στη μείωση της απώλειας και προσαρμογής, καθιστώντας τους λιγότερο αποδοτικούς για τη συγκεκριμένη εφαρμογή.



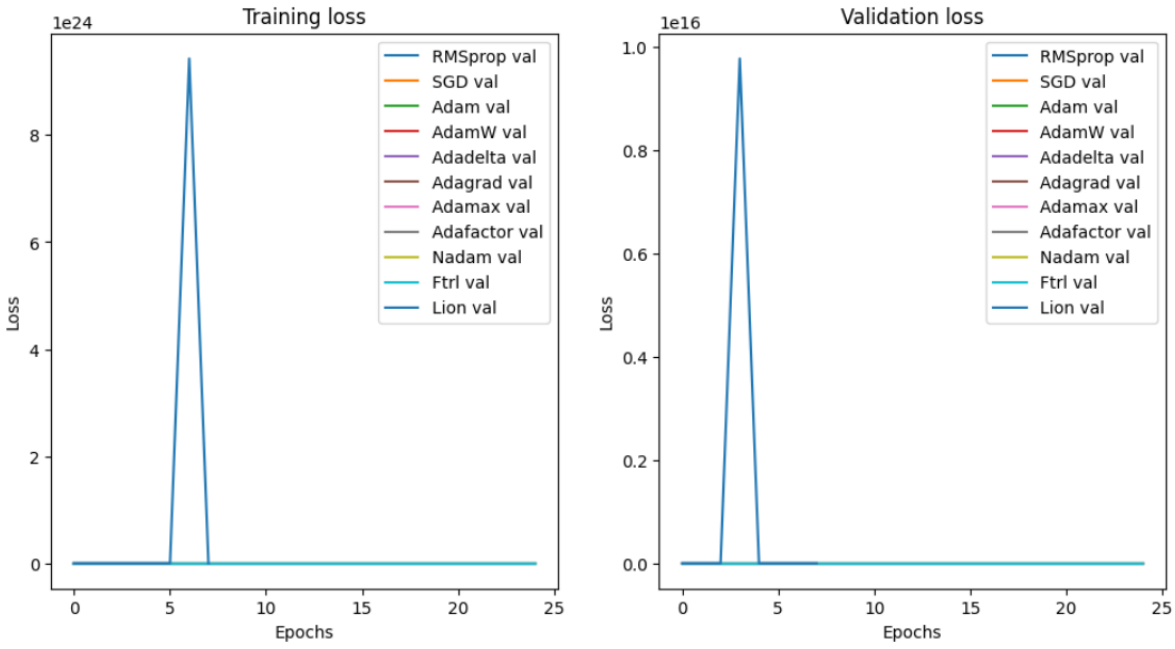
Εικόνα 28- MNIST Accuracy

Στις γραφικές παραστάσεις της ορθότητας του σύνολο εκπαίδευσης και του συνόλου επαλήθευσης, οι αλγόριθμοι Ftrl και Adafactor εμφανίζονται σχεδόν σταθεροί, με καμία ουσιαστική βελτίωση κατά την εκπαίδευση.

Ο Adadelta επιδεικνύει μια σταθερή, σταδιακή αύξηση.

Ο Lion παρουσιάζει απότομη μείωση και στη συνέχεια γίνεται σχεδόν σταθερός.

Οι υπόλοιποι αλγόριθμοι παρουσιάζουν λογαριθμική αύξηση στην ορθότητα, με την απόδοση τους να βελτιώνεται σταθερά.



Εικόνα 29- MNIST Loss

Στις γραφικές παραστάσεις της απώλειας του σύνολο εκπαίδευσης και σύνολο επαλήθευσης, οι αιχμές στην απώλεια οφείλονται κυρίως σε μεμονωμένους αλγόριθμους όπως ο Ftrl και ο Lion, οι οποίοι παρουσιάζουν ασταθή συμπεριφορά και μεγάλες διακυμάνσεις. Αυτό προκαλεί αύξηση του εύρους του άξονα των απωλειών, καθιστώντας δυσδιάκριτους τους αλγόριθμους με μικρές και σταθερές απώλειες.Για να αποτυπωθούν καλύτερα οι μικρές απώλειες των σταθερών αλγορίθμων, είναι σκόπιμο να αποκλειστούν οι αλγόριθμοι με μεγάλες αιχμές ή να χρησιμοποιηθεί λογαριθμική κλίμακα, προκειμένου να μειωθεί το εύρος της κλίμακας των απωλειών.

Ο πιο αποδοτικός αλγόριθμος για το σύνολο δεδομένων MNIST είναι ο Adamax, καθώς πέτυχε την υψηλότερη ορθότητα, η οποία ανέρχεται στο 98.05%.

Οι αλγόριθμοι Nadam και AdamW παρουσίασαν παρόμοια απόδοση, με την ακρίβειά τους να ανέρχεται σε 97.65% και 97.61% αντίστοιχα, υποδεικνύοντας ότι είναι επίσης αρκετά αποδοτικοί. Συνολικά, η χρήση του Adamax φαίνεται να προσφέρει την καλύτερη ισορροπία μεταξύ αύξησης της ορθότητας και σταθερότητας κατά την εκπαίδευση και επαλήθευση, καθιστώντας τον την καλύτερη επιλογή για αυτό το σύνολο δεδομένων.

Σύνολο δεδομένων CIFAR-10

Στο σύνολο δεδομένων CIFAR-10, οι αλγόριθμοι παρουσιάζουν χαμηλή απόδοση, καθώς αναγνωρίζουν μόνο μία κλάση. Συγκεκριμένα, οι ακρίβειες που επιτυγχάνονται είναι οι εξής: RMSprop 10%, SGD 10.01%, Adam 10%, AdamW 10%, Adadelta 11.99%, Adagrad 10%, Adamax 14.69%, Adafactor 10%, Nadam 10%, FTRL 10%, και Lion 10%. Αυτά τα αποτελέσματα υποδηλώνουν ότι οι αλγόριθμοι αποτυγχάνουν να γενικεύσουν σωστά και περιορίζονται σε τυχαία πρόβλεψη μίας μόνο κλάσης.

Κατά την εκπαίδευση του μοντέλου για 25 εποχές, οι αλγόριθμοι βελτιστοποίησης επιδεικνύουν διάφορες συμπεριφορές, ειδικά όσον αφορά την ορθότητα και την απώλεια.

Ο RMSprop, Adam, Adagrad και Nadam παρουσιάζουν αυξομειώσεις στην ορθότητα τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης, με μικρή συνολική αύξηση της τιμής της. Σε αυτούς τους αλγόριθμους, η απώλεια είτε μειώνεται ελάχιστα είτε παραμένει ασταθής, ενώ στο σύνολο επαλήθευσης παρατηρείται αυξομείωση ή στασιμότητα στην τιμή της.

Ο SGD εμφανίζει διακυμάνσεις στην ορθότητα, με πτωτική τάση στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης σταθεροποιείται σε μία συγκεκριμένη τιμή. Η απώλεια μειώνεται σταδιακά, με μικρές διακυμάνσεις σε αμφότερα τα σύνολα.

Ο AdamW καταγράφει αυξομειώσεις στην ορθότητα, με συνολική μείωση της τιμής της. Η απώλεια αυξομειώνεται ελαφρά στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης καταγράφεται αύξηση.

Ο Adadelta παρουσιάζει πτώση της ορθότητας και της απώλειας με διακυμάνσεις, σταθεροποιώντας σταδιακά την τιμή της απώλειας και στα δύο σύνολα.

Ο Adamax εμφανίζει πτωτική τάση στην ορθότητα στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης παραμένει αμετάβλητη. Η απώλεια μειώνεται ελαφρά στο σύνολο εκπαίδευσης και παραμένει σταθερή στο σύνολο επαλήθευσης.

Ο Adafactor εμφανίζει σταθερή ορθότητα στο σύνολο επαλήθευσης, ενώ οι τιμές της απώλειας είναι μη έγκυρες (NaN) τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης.

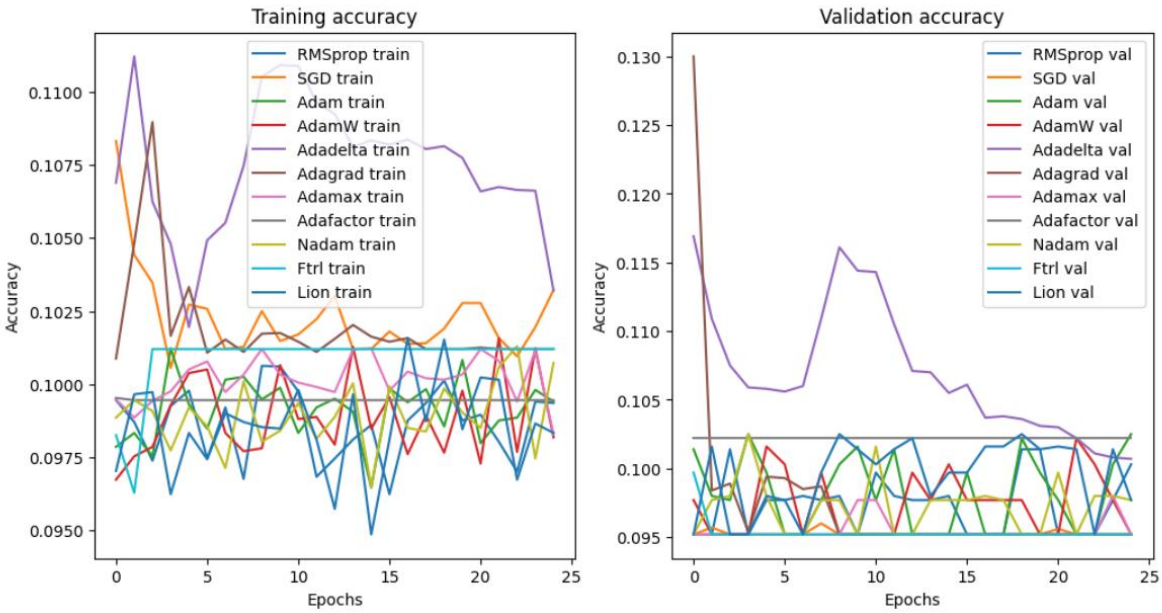
Ο FTRL, παρότι παρουσιάζει άνοδο της ορθότητας με αυξομειώσεις στο σύνολο εκπαίδευσης, καταγράφει μη έγκυρες τιμές απώλειας (NaN) και στα δύο σύνολα.

Ο Lion εμφανίζει αύξηση της ορθότητας με διακυμάνσεις, ενώ η απώλεια αυξάνεται ελαφρώς στο σύνολο εκπαίδευσης και μειώνεται στο σύνολο επαλήθευσης.

Τέλος, όσον αφορά τα γραφήματα για τις μετρικές precision, recall και F1-score, οι περισσότεροι αλγόριθμοι καταφέρνουν να αναγνωρίσουν μόνο μία κλάση, με τον Adadelta να αναγνωρίζει δύο.

Γενικά, στο σύνολο δεδομένων CIFAR-10, είναι εμφανές ότι οι περισσότεροι αλγόριθμοι δυσκολεύονται να γενικεύσουν καλά, παρουσιάζοντας πολύ χαμηλή απόδοση, γύρω στο 10%.

Ο Adadelta, με ορθότητα 10.6%, αναγνωρίζει μόνο τρεις κλάσεις, κάτι που υποδεικνύει αδυναμία να επεξεργαστεί αποτελεσματικά τα δεδομένα. Τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης παρατηρούνται έντονες αυξομειώσεις στην ορθότητα και την απώλεια, δείχνοντας ότι οι αλγόριθμοι δεν καταφέρνουν να αναγνωρίσουν σωστά τις δομές των δεδομένων και δεν προσαρμόζονται καλά στις προκλήσεις που παρουσιάζει το συγκεκριμένο σύνολο.

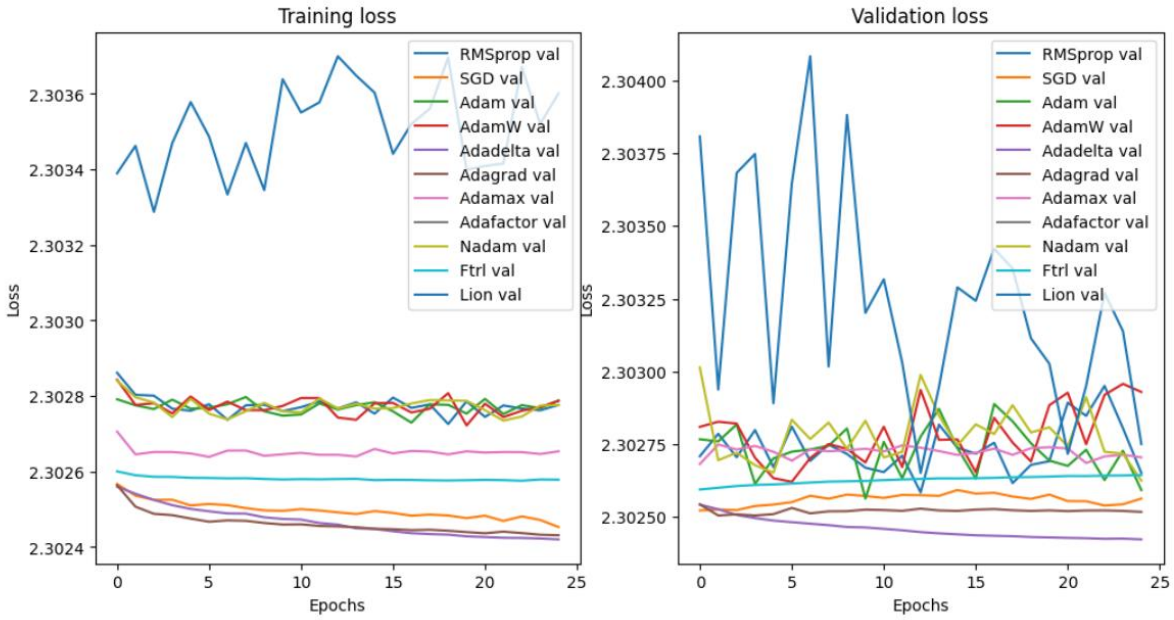


Εικόνα 30- CIFAR-10 - Accuracy

Σύμφωνα με τα διαγράμματα της ορθότητας εκπαίδευσης και επαλήθευσης, οι αλγόριθμοι Ftrl και Adafactor εμφανίζονται σταθεροί και κινούνται σχεδόν παράλληλα με τον άξονα των εποχών.

Από την άλλη, οι αλγόριθμοι Adadelta και Adamax παρουσιάζουν σημαντικές αυξομειώσεις, αλλά συνολικά ακολουθούν ανοδική πορεία, επιτυγχάνοντας τις καλύτερες αποδόσεις σε σχέση με τους υπόλοιπους αλγόριθμους.

Οι υπόλοιποι αλγόριθμοι, όπως οι RMSprop, SGD, Adam και άλλοι, εμφανίζουν επίσης διακυμάνσεις στην ορθότητα, όμως οι τροχιές τους είναι σχετικά πιο σταθερές και γραμμικές, χωρίς να ξεχωρίζουν ιδιαίτερα στις επιδόσεις τους.



Εικόνα 31- CIFAR-10 - Loss

Σύμφωνα με τα διαγράμματα της απώλειας εκπαίδευσης και επαλήθευσης, ο αλγόριθμος Adamax παρουσιάζει τις πιο έντονες διακυμάνσεις, με σημαντικές αυξομειώσεις κατά τη διάρκεια των εποχών, αλλά συνολικά ακολουθεί καθοδική πορεία, σημειώνοντας τη μεγαλύτερη μείωση στην απώλεια.

Από την άλλη πλευρά, οι υπόλοιποι αλγόριθμοι, όπως οι RMSprop, SGD, Adam και άλλοι, παραμένουν σχεδόν σταθεροί, με τις καμπύλες τους να είναι σχεδόν παράλληλες στον άξονα των εποχών υποδηλώνοντας ελάχιστες διακυμάνσεις στην απώλεια κατά τη διάρκεια της εκπαίδευσης και της επαλήθευσης.

Ο πιο αποδοτικός αλγόριθμος για το σύνολο δεδομένων CIFAR-10 είναι ο Adadelta, καθώς πέτυχε την υψηλότερη ορθότητα, η οποία ανέρχεται στο 10.6%.

Σύνολο δεδομένων CIFAR-100

Στο σύνολο δεδομένων CIFAR-100, οι περισσότεροι αλγόριθμοι παρουσιάζουν πολύ χαμηλή απόδοση. Πιο συγκεκριμένα, ο RMSprop έχει απόδοση 1%, ο SGD 1%, ο Adam 1%, ο AdamW 1%, ο Adadelta 0.81%, ο Adagrad 1%, ο Adamax 1%, ο Adafactor 1%, ο Nadam 1%, ο Ftrl 1% και ο Lion 1%.

Κατά την εκπαίδευση του μοντέλου για 25 εποχές, οι αλγόριθμοι παρουσιάζουν διαφορετικές συμπεριφορές.

Ο RMSprop εμφανίζει μικρή αύξηση της ορθότητας με διακυμάνσεις στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης η ορθότητα μειώνεται με διακυμάνσεις, και η απώλεια μειώνεται ελάχιστα στο σύνολο εκπαίδευσης αλλά αυξάνεται στο σύνολο επαλήθευσης.

Ο SGD εμφανίζει αύξηση της ορθότητας με διακυμάνσεις στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης η ορθότητα μειώνεται και σταθεροποιείται σε μία τιμή, με την απώλεια να μειώνεται ομαλά στην εκπαίδευση και να αυξάνεται στο σύνολο επαλήθευσης.

Ο Adam και ο Adamax παρουσιάζουν αύξηση της ορθότητας με διακυμάνσεις στην εκπαίδευση, ενώ στο σύνολο επαλήθευσης η ορθότητα μειώνεται και σταθεροποιείται. Η απώλεια και στους δύο μειώνεται με διακυμάνσεις στο σύνολο εκπαίδευσης και αυξάνεται ελαφρώς στο σύνολο επαλήθευσης.

Ο AdamW εμφανίζει μείωση της ορθότητας με ελάχιστες διακυμάνσεις στην εκπαίδευση και σταθεροποίηση στο σύνολο επαλήθευσης, ενώ η απώλεια διακυμαίνεται στην εκπαίδευση χωρίς ουσιαστική αλλαγή και αυξάνεται στο σύνολο επαλήθευσης.

Ο Adadelta παρουσιάζει μείωση της ορθότητας με διακυμάνσεις και στα δύο σύνολα, ενώ η απώλεια παραμένει σταθερή.

Ο Adagrad παρουσιάζει αύξηση της ορθότητας με διακυμάνσεις στην εκπαίδευση, ενώ η ορθότητα στο σύνολο επαλήθευσης παραμένει σταθερή και η απώλεια μειώνεται ελάχιστα στην εκπαίδευση και αυξάνεται ελάχιστα στο σύνολο επαλήθευσης.

Ο Adafactor εμφανίζει αυξομειώσεις στην ορθότητα στο σύνολο εκπαίδευσης με σταθερή ορθότητα στο σύνολο επαλήθευσης, αλλά η απώλεια έχει τιμή NaN και στα δύο σύνολα.

Ο Nadam παρουσιάζει μείωση της ορθότητας με διακυμάνσεις στην εκπαίδευση και σχεδόν σταθερή ορθότητα στο σύνολο επαλήθευσης, ενώ η απώλεια μειώνεται με διακυμάνσεις στην εκπαίδευση και αυξάνεται στο σύνολο επαλήθευσης.

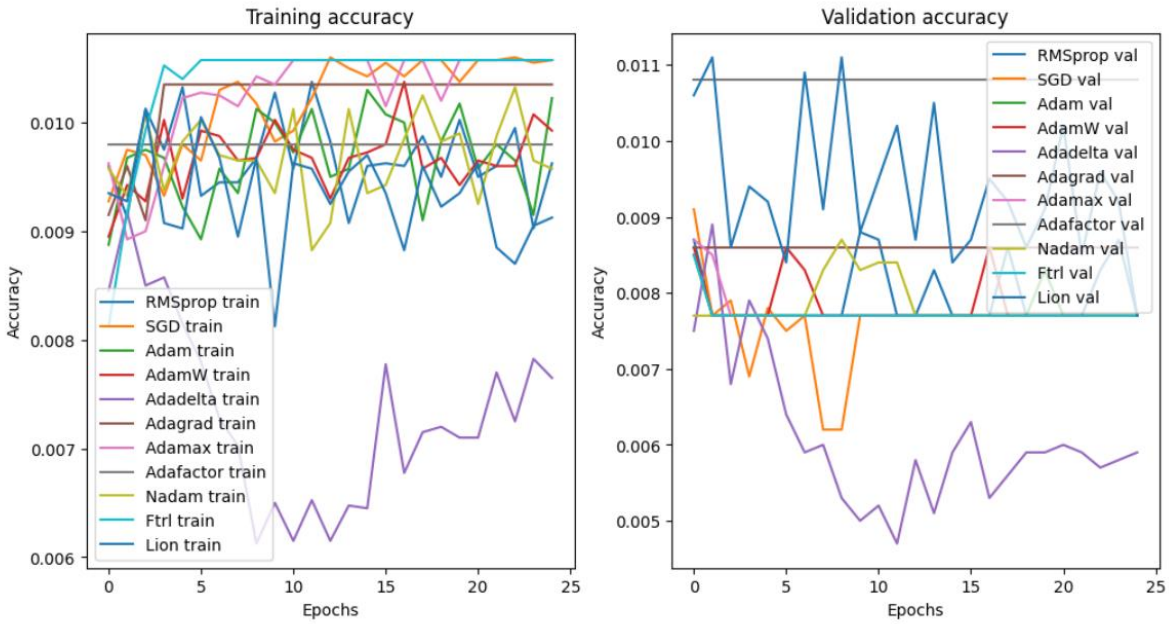
Ο Ftrl παρουσιάζει αύξηση της ορθότητας με διακυμάνσεις στην εκπαίδευση, ενώ η ορθότητα στο σύνολο επαλήθευσης παραμένει σταθερή, με την απώλεια να μειώνεται ελάχιστα στην εκπαίδευση και να αυξάνεται ομαλά στο σύνολο επαλήθευσης.

Ο Lion παρουσιάζει μείωση της ορθότητας με διακυμάνσεις και στα δύο σύνολα, ενώ η απώλεια αυξάνεται με διακυμάνσεις.

Τέλος, στα διαγράμματα για precision, recall και F1-score, οι περισσότεροι αλγόριθμοι αναγνωρίζουν μόνο μία κλάση, με εξαίρεση τον Adadelta, ο οποίος αναγνωρίζει τρεις κλάσεις.

Στο σύνολο δεδομένων CIFAR-100, όπως και στο CIFAR-10, παρατηρείται ότι οι περισσότεροι αλγόριθμοι δυσκολεύονται να γενικεύσουν σωστά, επιτυγχάνοντας πολύ χαμηλή απόδοση, κοντά στο 1%.

Ο Adamax, με την υψηλότερη ορθότητα στο 22.2%, αναγνωρίζει μόνο ορισμένες κλάσεις, γεγονός που υποδεικνύει περιορισμένη δυνατότητα επεξεργασίας και αναγνώρισης των δεδομένων.Τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης, παρατηρούνται έντονες αυξομειώσεις στην ορθότητα και την απώλεια, δείχνοντας ότι οι αλγόριθμοι δεν μπορούν να αναγνωρίσουν επαρκώς τις δομές των δεδομένων και δεν προσαρμόζονται στις απαιτήσεις του συνόλου. Αυτές οι διακυμάνσεις αντικατοπτρίζουν προβλήματα στη σταθερότητα και την ικανότητα γενίκευσης των μοντέλων, τα οποία αποτυγχάνουν να ανταποκριθούν στις πολυπλοκότητες του CIFAR-100.



Εικόνα 32 - CIFAR-100 - Accuracy

Στη γραφική παράσταση της ορθότητας εκπαίδευσης, ο αλγόριθμος Adafactor παραμένει σχεδόν παράλληλος με τον άξονα των εποχών, υποδεικνύοντας μικρή ή καθόλου βελτίωση.

Οι αλγόριθμοι Ftrl και Adagrad αρχικά παρουσιάζουν αύξηση στην ορθότητα, αλλά στη συνέχεια σταθεροποιούνται και γίνονται επίσης παράλληλοι με τον άξονα των εποχών.

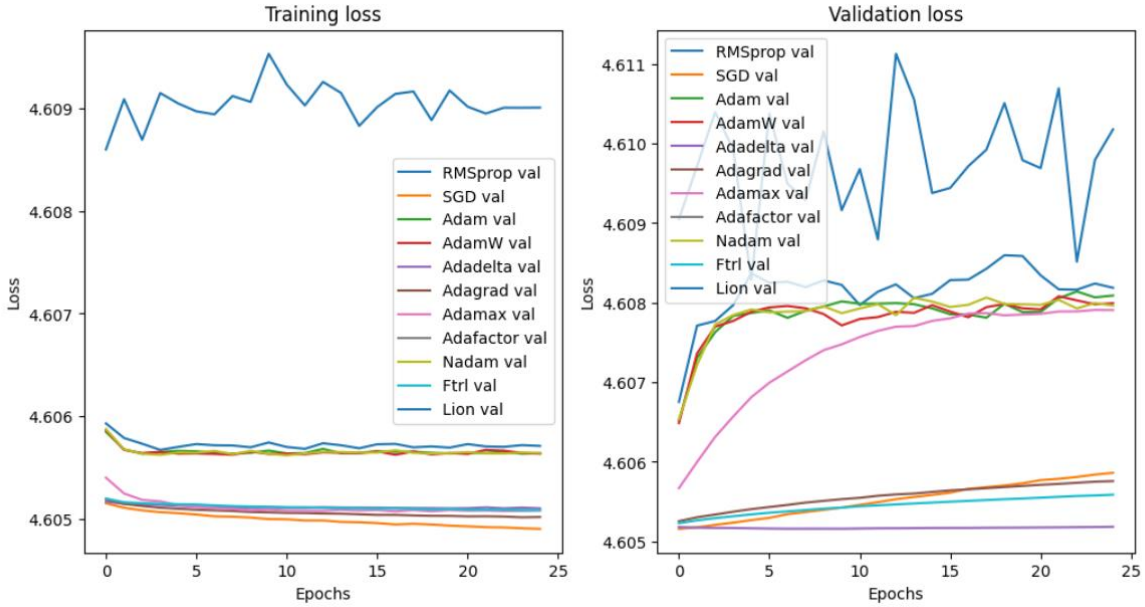
Ο Adadelta εμφανίζει μείωση στην ορθότητα μέχρι την 10η εποχή, μετά την οποία ακολουθεί ανοδική πορεία με έντονες διακυμάνσεις.

Οι υπόλοιποι αλγόριθμοι, όπως οι RMSprop, Adam, και SGD, εμφανίζουν επίσης έντονες διακυμάνσεις καθ' όλη τη διάρκεια της εκπαίδευσης.

Στη γραφική παράσταση της ορθότητας επαλήθευσης, οι αλγόριθμοι Adafactor και Adagrad παραμένουν παράλληλοι με τον άξονα των εποχών, παρουσιάζοντας σταθερή, χαμηλή ορθότητα.

Ο Ftrl παρουσιάζει μείωση στην ορθότητα και στη συνέχεια σταθεροποιείται σε χαμηλό επίπεδο, ενώ οι υπόλοιποι αλγόριθμοι, όπως οι Adam, AdamW, και RMSprop, ακολουθούν μια πορεία με έντονες διακυμάνσεις και γενικά μειούμενη ορθότητα.

Ο Adadelta είναι αυτός που παρουσιάζει τη μεγαλύτερη μείωση στην ορθότητα επαλήθευσης, ειδικά μετά την 10η εποχή.



Εικόνα 33- CIFAR-100 - Loss

Στη γραφική παράσταση της απώλειας εκπαίδευσης, ο αλγόριθμος Lion παρουσιάζει έντονες διακυμάνσεις και ακολουθεί αυξητική πορεία.

Οι υπόλοιποι αλγόριθμοι, όπως οι RMSprop, Adam, και Adagrad, εμφανίζουν μειωτική πορεία και οι καμπύλες τους είναι σχεδόν παράλληλες με τον άξονα των εποχών.

Στη γραφική παράσταση της απώλειας επαλήθευσης, ο Lion συνεχίζει να παρουσιάζει τις πιο έντονες διακυμάνσεις.

Οι RMSprop, Adam, AdamW, Adamax, και Nadam ακολουθούν λογαριθμική αύξηση με διακυμάνσεις, ενώ ο Adadelta παραμένει σχεδόν σταθερός και παράλληλος με την αρχή των αξόνων, χωρίς σημαντική μεταβολή στην απώλεια.

Οι υπόλοιποι αλγόριθμοι, όπως ο SGD,Ftrl και ο Adagrad, εμφανίζουν ομαλή αύξηση της απώλειας, ακολουθώντας μια ευθεία πορεία με μορφή y=ax+b.

Ο πιο αποδοτικός αλγόριθμος για το σύνολο δεδομένων CIFAR-100 είναι ο Adamax με ορθότητα 22.2%

Σύνολο δεδομένων Fashion mnist

Στο σύνολο δεδομένων Fashion MNIST, οι αλγόριθμοι παρουσιάζουν ποικίλες αποδόσεις. Ο RMSprop καταγράφει ορθότητα 79.31%, ενώ οι Adafactor, Ftrl, και Lion εμφανίζουν πολύ χαμηλή ορθότητα, γύρω στο 10%. Ο SGD επιτυγχάνει μια αξιοπρεπή απόδοση με ορθότητα 84.46%, ενώ ο Adagrad φτάνει το 79.75%. Η απόδοση του Adadelta είναι σημαντικά χαμηλότερη, με ορθότητα 56.69%. Αντίθετα, οι αλγόριθμοι Adam, AdamW, Adamax, και Nadam σημειώνουν τις υψηλότερες επιδόσεις, με ορθότητα 86.4%, 86.26%, 87.2%, και 87.08% αντίστοιχα.

Κατά την εκπαίδευση του μοντέλου για 25 εποχές, οι αλγόριθμοι παρουσιάζουν διαφορετική συμπεριφορά.

Ο RMSprop, τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης, παρουσιάζει διακυμάνσεις στην ορθότητα, η οποία τελικά αυξάνεται, ενώ και η απώλεια παρουσιάζει έντονες διακυμάνσεις με αποτέλεσμα να μειώνεται στο σύνολο εκπαίδευσης και να αυξάνεται στο σύνολο επαλήθευσης .

Οι SGD, Adam, AdamW, Adagrad, Adamax, Nadam παρουσιάζουν διακυμάνσεις με αύξηση της ορθότητας και μείωση της απώλειας τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης.

Αντίθετα, ο Adadelta εμφανίζουν σταθερή αύξηση της ορθότητας και μείωση της απώλειας τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης .

Ο Adafactor εμφανίζει NaN στην απώλεια τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης, με την ορθότητα να αυξάνεται ελάχιστα στο σύνολο εκπαίδευσης και στο σύνολο επαλήθευσης και να παραμένει σταθερή.

Ο Ftrl παρουσιάζει μικρή αύξηση στην ορθότητα και ελάχιστη μείωση στην απώλεια στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης η ορθότητα και η απώλεια είναι σταθερή σε μια χαμηλή τιμή.

Ο Lion ξεκινά με υψηλή ορθότητα, η οποία μειώνεται στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης μειώνεται και τελικά παραμένει σταθερή. Η απώλεια σε αυτόν τον αλγόριθμο αυξάνεται δραματικά και τελικά γίνεται NaN στα σύνολα εκπαίδευσης και επαλήθευσης.
Στα ιστογράμματα για precision, recall και F1-score, οι αλγόριθμοι Adafactor, Ftrl και Lion αναγνωρίζουν μόνο μία κλάση.

Ο Adadelta επιτυγχάνει τιμές πάνω από 0.4 για τις περισσότερες κλάσεις, ενώ οι υπόλοιποι αλγόριθμοι επιτυγχάνουν τιμές πάνω από 0.6.

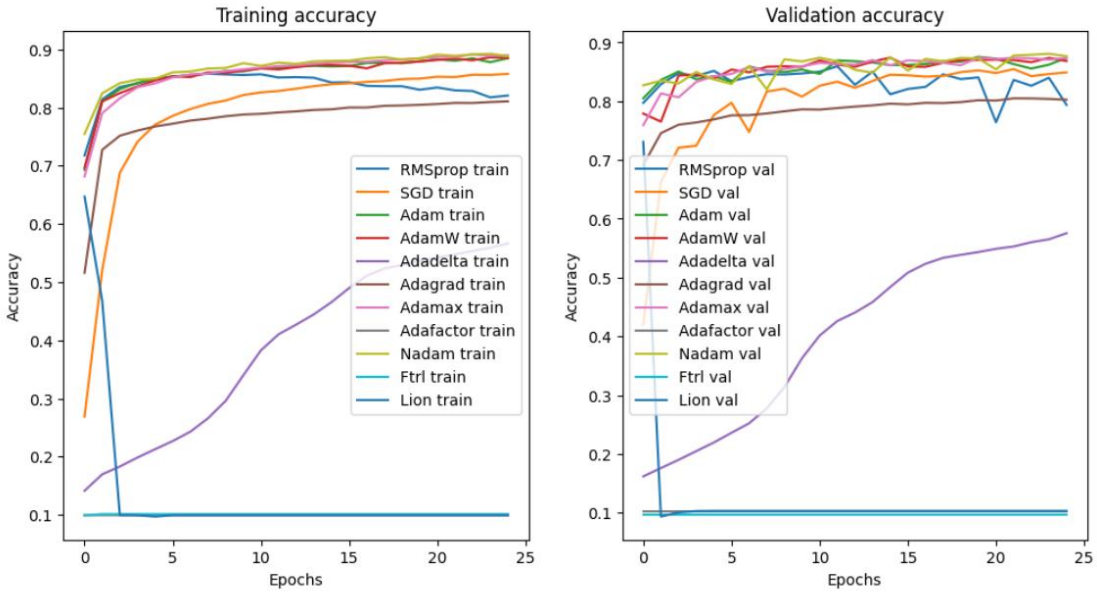
Γενικά, στο σύνολο δεδομένων Fashion MNIST, οι αλγόριθμοι παρουσιάζουν σχετικά καλή απόδοση, με τους Adam, AdamW, Adamax, Nadam και SGD να καταγράφουν τις υψηλότερες ακρίβειες και να μειώνουν την απώλεια. Ωστόσο, παρατηρούνται διακυμάνσεις στο σύνολο επαλήθευσης, γεγονός που υποδηλώνει πιθανή αστάθεια κατά την προσαρμογή σε νέα δεδομένα και τάση για υπερπροσαρμογή (overfitting). Παρόλο που αυτοί οι αλγόριθμοι εμφανίζουν πολύ καλή απόδοση στο σύνολο εκπαίδευσης, η μειωμένη σταθερότητα στο σύνολο επαλήθευσης απαιτεί προσοχή.

Ο Adagrad, αν και επιτυγχάνει χαμηλότερες επιδόσεις, παρουσιάζει μεγαλύτερη σταθερότητα, κυρίως στην ικανότητα γενίκευσης σε νέα δεδομένα, κάτι που τον καθιστά πιο ανθεκτικό στις προκλήσεις της υπερπροσαρμογής.

Από την άλλη, ο RMSprop, παρότι επιτυγχάνει ορθότητα 79.31%, φαίνεται να υπερεκπαιδεύεται, καθώς η ορθότητα μειώνεται και η απώλεια αυξάνεται στο σύνολο επαλήθευσης, γεγονός που δείχνει προβλήματα στη γενίκευση.

Ο Adadelta ακολουθεί, με χαμηλότερη απόδοση σε σύγκριση με τους υπόλοιπους αλγόριθμους, ενώ παρουσιάζει και αυτός διακυμάνσεις στην απόδοσή του.

Τέλος, οι αλγόριθμοι FTRL, Adafactor και Lion καταγράφουν τις χαμηλότερες επιδόσεις, με σημαντικές δυσκολίες στη μείωση της απώλειας και αδυναμία προσαρμογής στα δεδομένα, γεγονός που τους καθιστά λιγότερο αποδοτικούς για τη συγκεκριμένη εφαρμογή.



Εικόνα 34- Fashion MNIST - Accuracy

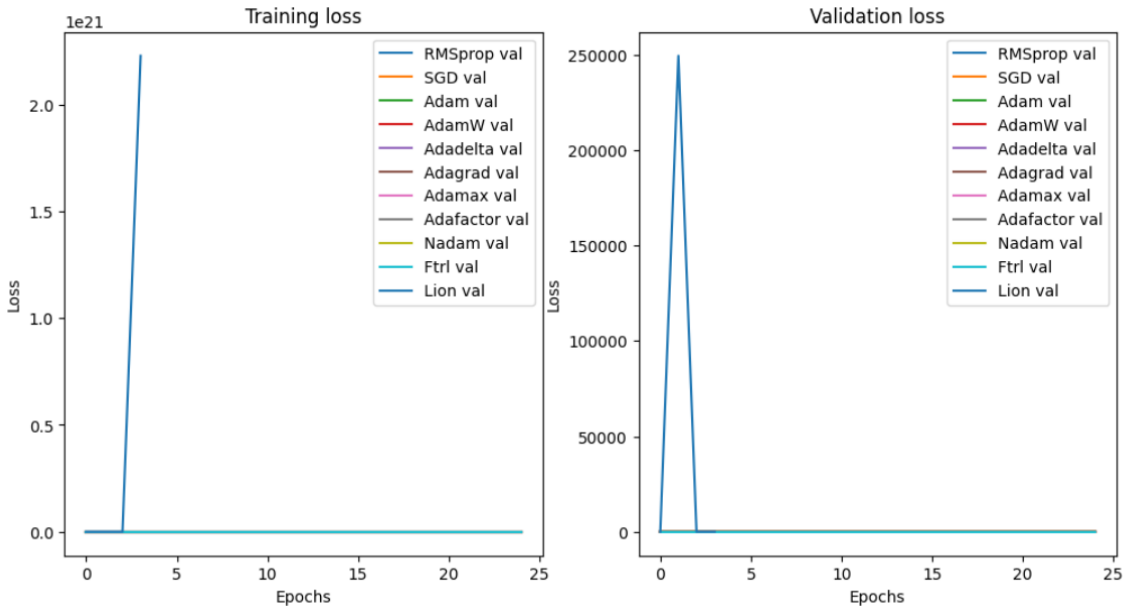
Στις γραφικές παραστάσεις της ορθότητας του σύνολο εκπαίδευσης και του συνόλου επαλήθευσης, οι αλγόριθμοι Adafactor και Ftrl εμφανίζονται ως παράλληλες γραμμές με τον άξονα των εποχών, υποδηλώνοντας σταθερή απόδοση χωρίς βελτίωση.

Ο Adadelta προσομοιάζει μια ευθεία γραμμή, δείχνοντας πολύ μικρή βελτίωση στην ορθότητα.

Ο RMSprop αρχικά ακολουθεί παραβολική καμπύλη με αύξηση αλλά στη συνέχεια πέφτει απότομα και γίνεται παράλληλος με τον άξονα των εποχών.

Ο Lion, σχεδόν αμέσως, πέφτει κατακόρυφα και στη συνέχεια παραμένει παράλληλος με τον άξονα των εποχών.

Αντίθετα, οι υπόλοιποι αλγόριθμοι παρουσιάζουν λογαριθμική αύξηση στην ορθότητα, υποδηλώνοντας συνεχή βελτίωση κατά την εκπαίδευση.



Εικόνα 35 - Fashion MNIST - Loss

Στις γραφικές παραστάσεις της απώλειας στα σύνολα εκπαίδευσης και επαλήθευσης, οι αιχμές στην απώλεια οφείλονται κυρίως σε μεμονωμένους αλγόριθμους όπως ο Ftrl και ο Lion, οι οποίοι παρουσιάζουν ασταθή συμπεριφορά και μεγάλες διακυμάνσεις. Αυτό προκαλεί αύξηση του

εύρους του άξονα των απωλειών, καθιστώντας δυσδιάκριτους τους αλγόριθμους με μικρές και σταθερές απώλειες.

Ο πιο αποδοτικός αλγόριθμος για το σύνολο δεδομένων Fashion MNIST είναι ο Adamax, καθώς πέτυχε την υψηλότερη ορθότητα, η οποία ανέρχεται στο 87.2%.

Οι αλγόριθμοι Nadam και AdamW παρουσίασαν παρόμοια απόδοση, με την ακρίβειά τους να ανέρχεται σε 87.08% και 86.26% αντίστοιχα, υποδεικνύοντας ότι είναι επίσης αρκετά αποδοτικοί. Συνολικά, η χρήση του Adamax φαίνεται να προσφέρει την καλύτερη ισορροπία μεταξύ αύξησης της ορθότητας και σταθερότητας κατά την εκπαίδευση και επαλήθευση, καθιστώντας τον την καλύτερη επιλογή για αυτό το σύνολο δεδομένων.

Σύνολο δεδομένων IMDB

Στο σύνολο δεδομένων IMDB, όλοι οι αλγόριθμοι παρουσιάζουν ορθότητα 0.5, αναγνωρίζοντας μόνο τη μία κλάση. Αυτό υποδηλώνει ότι το convolutional neural network αδυνατεί να επεξεργαστεί αποτελεσματικά το συγκεκριμένο σύνολο δεδομένων.

Κατά την εκπαίδευση του μοντέλου για 25 εποχές, ο RMSprop εμφανίζει έντονες διακυμάνσεις τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης. Παρόλο που η ορθότητα αυξάνεται, η απώλεια παρουσιάζει σημαντικές αυξομειώσεις και στα δύο σύνολα, με την τιμή της τελικά να αυξάνεται.

Ο SGD παρουσιάζει έντονες διακυμάνσεις στην απώλεια κατά την εκπαίδευση, με σταδιακή αύξηση της ορθότητας. Στο σύνολο επαλήθευσης, οι διακυμάνσεις είναι ελάχιστες, με μικρή μείωση της ορθότητας και οριακή μείωση της απώλειας σε αμφότερα τα σύνολα.

Οι βελτιστοποιητές Adam, AdamW και Nadam παρουσιάζουν σταδιακή αύξηση της ορθότητας, με διακυμάνσεις τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης. Η απώλεια

μειώνεται με διακυμάνσεις στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης μειώνεται αρχικά, αλλά στη συνέχεια αυξάνεται, οδηγώντας τελικά σε αύξηση της συνολικής τιμής της.

Ο Adagrad εμφανίζει αύξηση της ορθότητας με διακυμάνσεις και στα δύο σύνολα, ενώ η απώλεια μειώνεται σταθερά.

Ο Adamax παρουσιάζει αύξηση της ορθότητας σε εκπαίδευση και επαλήθευση, με την απώλεια να μειώνεται με διακυμάνσεις στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης αυξάνεται με διακυμάνσεις.

Ο Adafactor εμφανίζει παρόμοια συμπεριφορά, με την ορθότητα να αυξάνεται με διακυμάνσεις και στα δύο σύνολα, ενώ η απώλεια μειώνεται με μικρές αυξομειώσεις στην επαλήθευση και αυξάνεται με έντονες διακυμάνσεις στην εκπαίδευση.

Ο Ftrl παρουσιάζει σταθερή αύξηση της ορθότητας με διακυμάνσεις στο σύνολο εκπαίδευσης και σταθερή απώλεια στο σύνολο επαλήθευσης.

Ο Lion, από την άλλη, εμφανίζει μείωση της ορθότητας κατά την εκπαίδευση, ενώ στο σύνολο επαλήθευσης η ορθότητα μειώνεται και σταθεροποιείται σε συγκεκριμένη τιμή. Η απώλεια αυξάνεται αρχικά και στη συνέχεια παίρνει την τιμή NaN και στα δύο σύνολα.

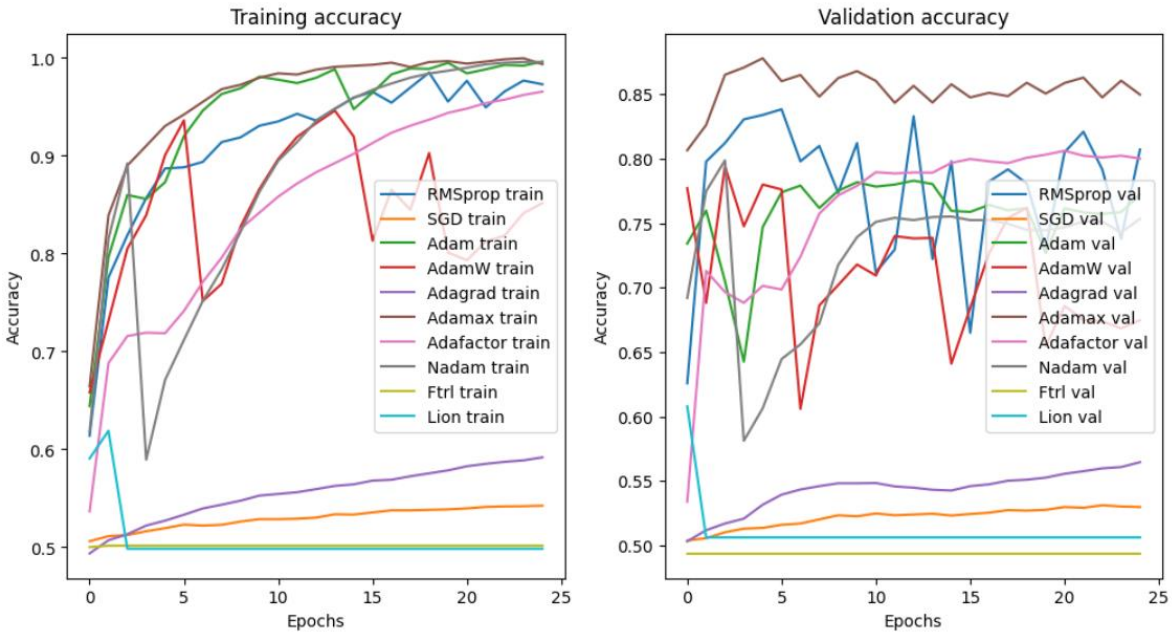
Τέλος, τα ιστογράμματα του precision δείχνουν τιμές κοντά στο 0.5, το recall φτάνει σχεδόν το 1, και το F1-score κυμαίνεται γύρω στο 0.7.

Όπως και στις προηγούμενες αρχιτεκτονικές, έτσι και στα Recurrent Neural Networks (RNNs) στο σύνολο δεδομένων IMDB, παρατηρείται έντονο overfitting, με την ορθότητα να παραμένει στο 0,5 για όλους τους αλγόριθμους, γεγονός που υποδηλώνει αδυναμία των μοντέλων να επεξεργαστούν αποτελεσματικά το σύνολο δεδομένων.

Οι Adam, AdamW, Adamax, Nadam, RMSprop και Adafactor παρουσιάζουν έντονες αυξομειώσεις στο σύνολο επαλήθευσης, επιδεικνύοντας αστάθεια κατά τη γενίκευση. Αυτές οι

διακυμάνσεις, σε συνδυασμό με την σταθερή απόδοση στο 0,5, δείχνουν ότι τα μοντέλα "υπερμαθαίνουν" τα δεδομένα εκπαίδευσης χωρίς να μπορούν να γενικεύσουν σωστά στα δεδομένα επαλήθευσης.

Οι SGD, Adadelta και Adagrad ακολουθούν, παρουσιάζοντας κάπως καλύτερη σταθερότητα, αλλά παραμένουν χαμηλά σε απόδοση, χωρίς να μπορούν να βελτιώσουν την ακρίβειά τους. Τέλος, οι Lion, FTRL και Adafactor σημειώνουν τις χειρότερες επιδόσεις.



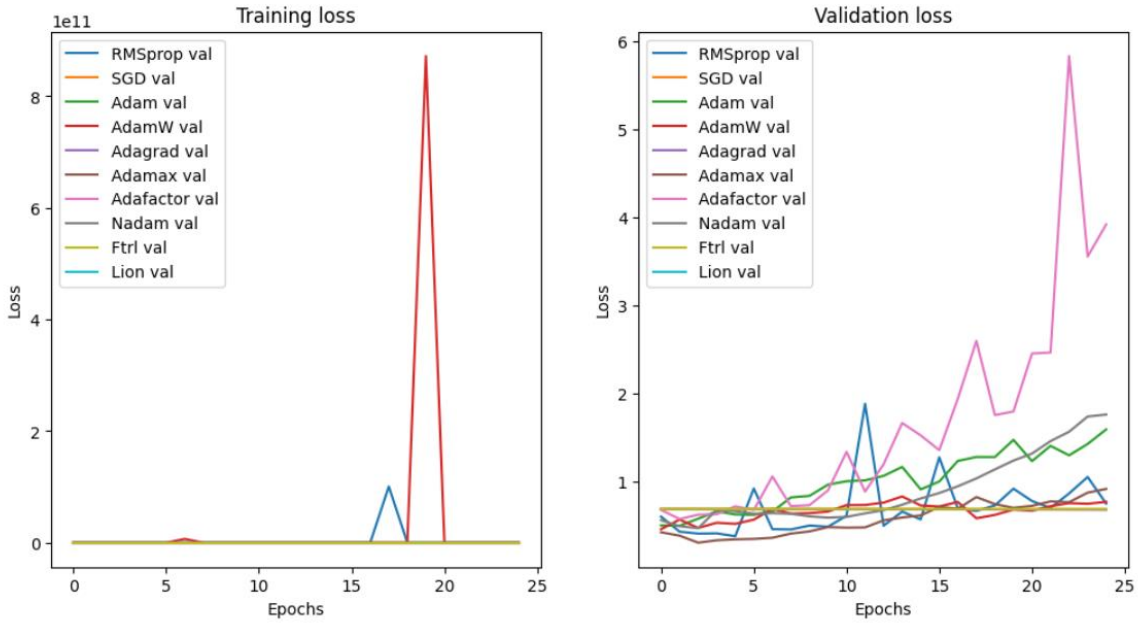
Εικόνα 36- IMDB - Accuracy

Στις γραφικές παραστάσεις της ορθότητας κατά την εκπαίδευση και επαλήθευση, οι βελτιστοποιητές SGD και Adagrad παρουσιάζουν αύξηση που ακολουθεί γραμμική πορεία, μοιάζοντας με την εξίσωση μιας ευθείας $y = ax + b$.

Ο Ftrl παραμένει σταθερός και παράλληλος με τον άξονα των εποχών, χωρίς να παρουσιάζει μεταβολές στην ορθότητα.

Ο Lion αρχικά δείχνει αύξηση της ορθότητας, αλλά στη συνέχεια καταγράφει απότομη πτώση και γίνεται παράλληλος με τον άξονα των εποχών κοντά στο μηδέν.

Αντίθετα, οι υπόλοιποι αλγόριθμοι παρουσιάζουν εκθετική αύξηση στην ορθότητα, συνοδευόμενη από έντονες διακυμάνσεις κατά την πορεία τους.



Εικόνα 37- IMDB - Loss

Στις γραφικές παραστάσεις της απώλειας στα σύνολα εκπαίδευσης και επαλήθευσης, οι αιχμές στην απώλεια οφείλονται κυρίως σε μεμονωμένους αλγόριθμους όπως ο Ftrl και ο Lion, οι οποίοι παρουσιάζουν ασταθή συμπεριφορά και μεγάλες διακυμάνσεις. Αυτό προκαλεί αύξηση του εύρους του άξονα των απωλειών, καθιστώντας δυσδιάκριτους τους αλγόριθμους με μικρές και σταθερές απώλειες.

Στο σύνολο δεδομένων IMDB, οι αλγόριθμοι παρουσιάζουν ορθότητα 0.5, δείχνοντας ότι το μοντέλο αδυνατεί να διακρίνει αποτελεσματικά τις κλάσεις.

Ο RMSprop και ο Lion εμφανίζουν αστάθειες στην απώλεια, ενώ οι SGD και Adagrad έχουν πιο σταθερή, γραμμική αύξηση της ορθότητας.

Οι Adam, AdamW και Nadam σημειώνουν αύξηση της ορθότητας με διακυμάνσεις στην απώλεια, ενώ ο Ftrl παραμένει σταθερός.

Συνολικά, ο Adamax ξεχωρίζει ως ο πιο αποδοτικός αλγόριθμος, προσφέροντας καλή ισορροπία μεταξύ ορθότητας και σταθερότητας.

Σύνολο δεδομένων Reuters

Στον σύνολο δεδομένων Reuters, οι αλγόριθμοι παρουσιάζουν μέτρια έως χαμηλή απόδοση. Αναλυτικότερα, τα ποσοστά απόδοσης των αλγορίθμων έχουν ως εξής: RMSprop 51.46%, SGD 36.19%, Adam 37.71%, AdamW 43.36%, Adadelta 36.19%, Adagrad 36.19%, Adamax 50.35%, Adafactor 49.24%, Nadam 44.07%, Ftrl 36.19% και Lion 37.93%.

Κατά την εκπαίδευση του μοντέλου για 25 εποχές, παρατηρούνται διακυμάνσεις στην απόδοση των διαφόρων αλγορίθμων.

Ο RMSprop παρουσιάζει σταδιακή αύξηση της ορθότητας στο σύνολο εκπαίδευσης με διακυμάνσεις, ενώ και στο σύνολο επαλήθευσης η ορθότητα αυξάνεται με διακυμάνσεις. Η απώλεια μειώνεται και στα δύο σύνολα με παρόμοιες διακυμάνσεις.

Από την άλλη πλευρά, ο SGD εμφανίζει αύξηση της ορθότητας στο σύνολο εκπαίδευσης με διακυμάνσεις, ενώ στο σύνολο επαλήθευσης η ορθότητα παραμένει σταθερή. Η απώλεια μειώνεται και στα δύο σύνολα, επίσης με διακυμάνσεις.

Ο Adam παρουσιάζει αύξηση της ορθότητας στο σύνολο εκπαίδευσης με διακυμάνσεις, ενώ στο σύνολο επαλήθευσης η ορθότητα μειώνεται με αυξομειώσεις. Η απώλεια μειώνεται με διακυμάνσεις στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης αυξάνεται με παρόμοιες διακυμάνσεις.

Ο AdamW εμφανίζει αύξηση της ορθότητας με διακυμάνσεις και στα δύο σύνολα, ενώ η απώλεια μειώνεται στο σύνολο εκπαίδευσης αλλά αυξάνεται στο σύνολο επαλήθευσης.

Παρόμοια συμπεριφορά παρουσιάζει και ο Adadelta, όπου η ορθότητα αυξάνεται με διακυμάνσεις στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης η ορθότητα σταθεροποιείται μετά από μια σταδιακή αύξηση. Η απώλεια μειώνεται και στα δύο σύνολα με διακυμάνσεις.

Ο Adagrad παρουσιάζει αύξηση της ορθότητας στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης η ορθότητα παραμένει σταθερή. Η απώλεια μειώνεται με διακυμάνσεις και στα δύο σύνολα.

Ο Adamax εμφανίζει πιο ομαλή αύξηση της ορθότητας στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης παρατηρούνται αυξομειώσεις στην ορθότητα. Η απώλεια μειώνεται ομαλά στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης αυξάνεται με διακυμάνσεις.

Οι αλγόριθμοι Adafactor και Nadam παρουσιάζουν αύξηση της ορθότητας και μείωση της απώλειας, τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης, με διακυμάνσεις.

Ο Ftrl παρουσιάζει μικρή μείωση της ορθότητας με διακυμάνσεις στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης η ορθότητα παραμένει σταθερή. Η απώλεια μειώνεται σταδιακά και στα δύο σύνολα.

Τέλος, ο Lion παρουσιάζει αύξηση της ορθότητας και μείωση της απώλειας, επίσης με διακυμάνσεις, τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης.

Όσον αφορά τα ιστογράμματα για τα μέτρα precision, recall και F1-score, οι αλγόριθμοι SGD, Adadelta, Adagrad και Ftrl αναγνωρίζουν μόνο μία κλάση.

Ο Lion αναγνωρίζει δύο κλάσεις, ενώ οι υπόλοιποι αλγόριθμοι αναγνωρίζουν κάποιες από τις συνολικές κλάσεις.

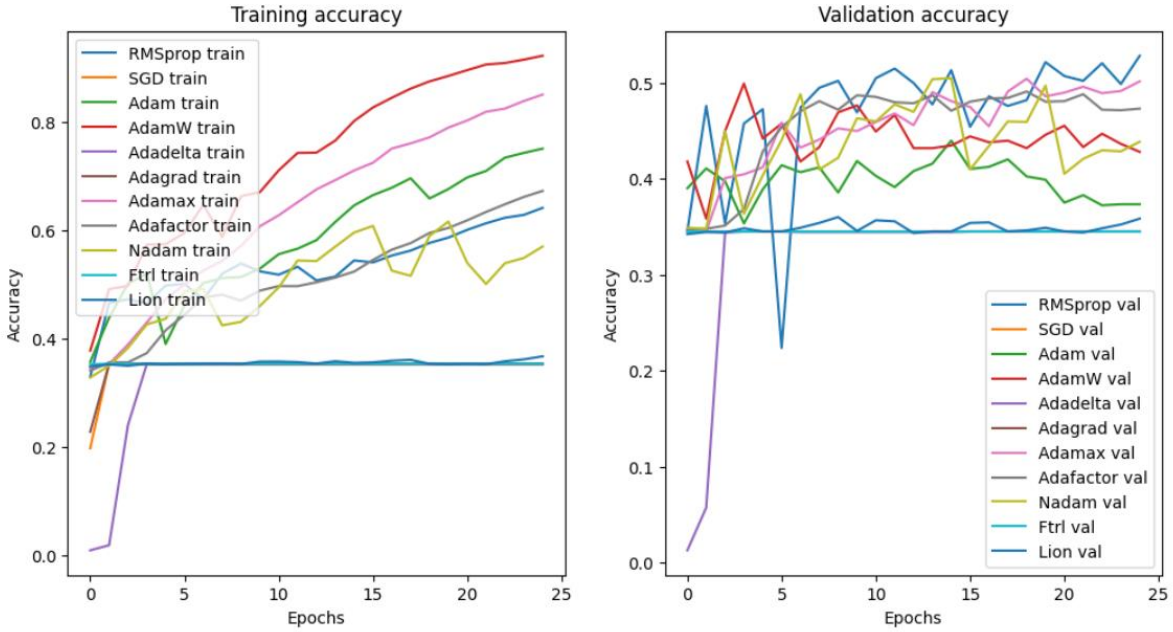
Γενικά, στο σύνολο δεδομένων Reuters, παρατηρείται ότι οι περισσότεροι αλγόριθμοι βελτιστοποίησης δυσκολεύονται να γενικεύσουν καλά, με τις αποδόσεις να κυμαίνονται σε χαμηλότερα επίπεδα από το επιθυμητό.

Οι αλγόριθμοι όπως οι Rmsprop, Adam, AdamW, Nadam και Lion παρουσιάζουν έντονες διακυμάνσεις κατά την προσαρμογή στα δεδομένα επαλήθευσής, γεγονός που υποδηλώνει δυσκολία στη σταθεροποίηση της απόδοσης.

Από την άλλη πλευρά, οι SGD, Adadelta και Adagrad δείχνουν πιο σταθερή συμπεριφορά, αλλά καταγράφουν χαμηλότερη απόδοση, με ακρίβειες γύρω στο 36%.

Επιπλέον, οι αλγόριθμοι Adafactor, Adamax, και Nadam παρουσιάζουν σταδιακή μείωση της απώλειας, ενώ οι FTRL και Lion αντιμετωπίζουν σημαντικές δυσκολίες στη βελτίωση της απόδοσης και την ικανότητα γενίκευσης.

Συνολικά, η χαμηλή γενίκευση και οι έντονες αυξομειώσεις στην ορθότητα και την απώλεια τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο επαλήθευσης δείχνουν ότι οι αλγόριθμοι δεν καταφέρνουν να αναγνωρίσουν σωστά τις δομές των δεδομένων και δυσκολεύονται να προσαρμοστούν στις προκλήσεις που παρουσιάζει το συγκεκριμένο σύνολο.



Εικόνα 38 - Reuters - Accuracy

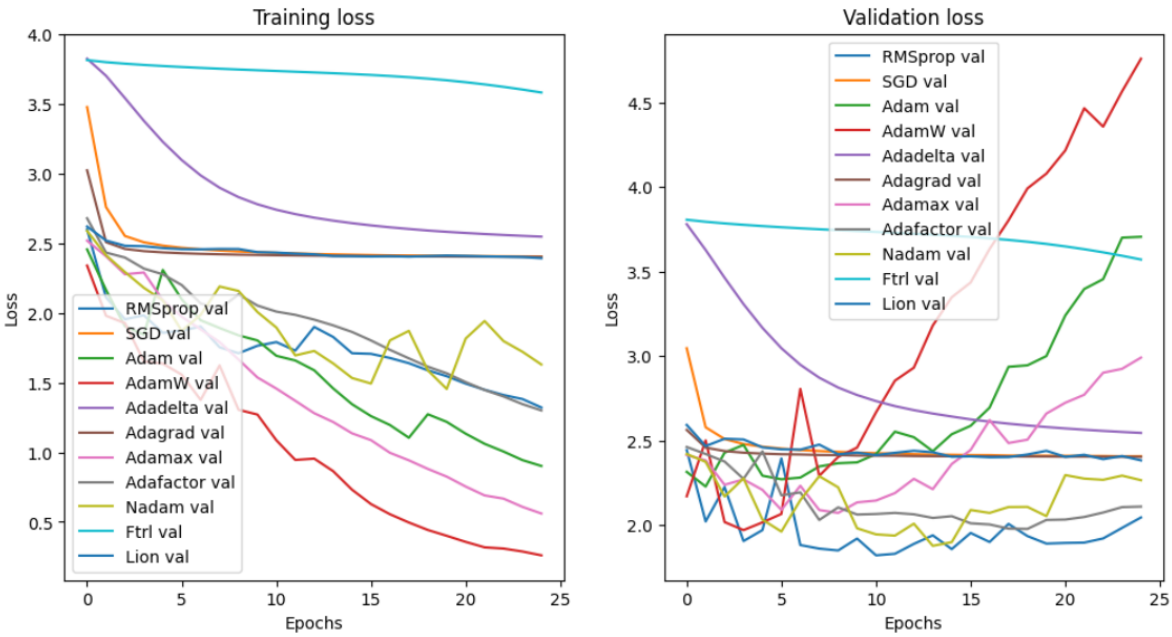
Στη γραφική παράσταση της ορθότητας εκπαίδευσης, ο αλγόριθμος Ftrl εμφανίζει μια σταθερή πορεία παράλληλη με τον άξονα των εποχών, ενώ ο Lion παρουσιάζει μικρές διακυμάνσεις αλλά παραμένει σχεδόν παράλληλος.

Ο Adadelta ξεκινά με μια αυξητική πορεία, αλλά στη συνέχεια η ακρίβειά του σταθεροποιείται και γίνεται παράλληλη με τον άξονα των εποχών.

Οι υπόλοιποι αλγόριθμοι εμφανίζουν λογαριθμική αύξηση της ορθότητας με μικρές διακυμάνσεις.

Στη γραφική παράσταση της ορθότητας επαλήθευσης παρατηρείται παρόμοια συμπεριφορά, αν και οι διακυμάνσεις είναι πιο έντονες.

Ο Adam, μετά από αρχική αύξηση, τελικά μειώνει την ακρίβειά του, ενώ ο RMSprop παρουσιάζει έντονες διακυμάνσεις, ιδιαίτερα στην 5η εποχή, όπου καταγράφεται σημαντική απόκλιση.



Εικόνα 39 - Reuters - Loss

Στη γραφική παράσταση της απώλειας εκπαίδευσης, ο αλγόριθμος Ftrl παρουσιάζει μια γραμμική μείωση, ενώ οι υπόλοιποι αλγόριθμοι μειώνονται εκθετικά.

Οι αλγόριθμοι RMSprop, SGD, Adagrad και Adadelta φαίνεται να σταθεροποιούνται και γίνονται σχεδόν παράλληλοι με τον άξονα των εποχών προς το τέλος της εκπαίδευσης.

Στη γραφική παράσταση της απώλειας επαλήθευσης, ο Ftrl παρουσιάζει επίσης γραμμική μείωση. Ο Adadelta εμφανίζει εκθετική μείωση, ενώ οι RMSprop, SGD και Adagrad αρχικά μειώνουν την απώλεια τους, αλλά στη συνέχεια οι καμπύλες τους γίνονται παράλληλες με τον άξονα των εποχών.

Από την άλλη, οι Adam, AdamW και Adamax παρουσιάζουν αύξηση της απώλειας με διακυμάνσεις, με τον AdamW να καταγράφει τη μεγαλύτερη απώλεια.

Οι υπόλοιποι αλγόριθμοι παρουσιάζουν διακυμάνσεις, μειώνοντας ελάχιστα την απώλειά τους κατά τη διάρκεια των εποχών.

Ο πιο αποδοτικός αλγόριθμος για το σύνολο δεδομένων Reuters είναι ο RMSprop, καθώς πέτυχε το υψηλότερο ποσοστό ορθότητας, το οποίο ανέρχεται σε 51.46%.

Οι αλγόριθμοι Adamax και Adafactor παρουσίασαν παρόμοια απόδοση, με ποσοστά ορθότητας 50.35% και 49.24% αντίστοιχα, υποδεικνύοντας ότι είναι επίσης αρκετά αποδοτικοί.

Συνολικά, η χρήση του RMSprop φαίνεται να προσφέρει την καλύτερη ισορροπία μεταξύ βελτίωσης της ορθότητας και σταθερότητας κατά την εκπαίδευση και επαλήθευση, καθιστώντας τον την καλύτερη επιλογή για αυτό το σύνολο δεδομένων.

Σύνολο δεδομένων California Housing

Για την αξιολόγηση της απόδοσης των μοντέλων στο πρόβλημα παλινδρόμησης, χρησιμοποιήθηκε το μέσο τετραγωνικό σφάλμα (Mean Squared Error - MSE). Συγκεκριμένα, οι τιμές που επιτεύχθηκαν για κάθε αλγόριθμο είναι οι εξής: ο RMSprop, ο Adam, ο AdamW, ο

Adamax, ο Nadam και ο Lion πέτυχαν τον χαμηλότερο MSE, με τιμή 2.43. Ο SGD είχε MSE 2.75, ο Adagrad 3.07, ο Adadelta 3.83 και ο Ftrl 3.74.

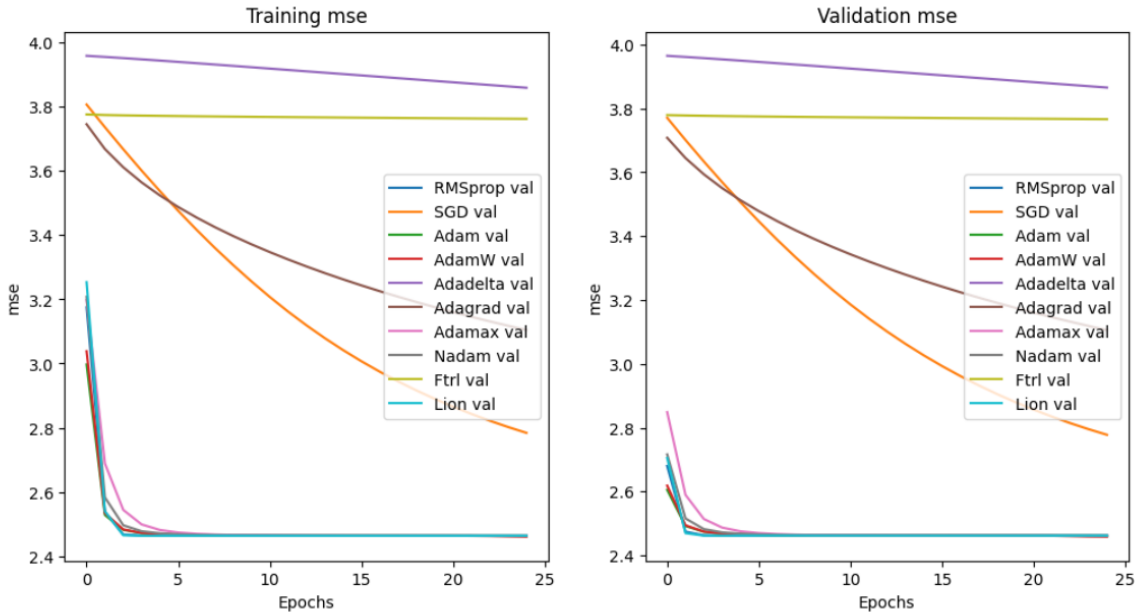
Κατά την εκπαίδευση του μοντέλου για 25 εποχές, η απώλεια για τους αλγόριθμους RMSprop, SGD, Adam, AdamW, Adadelta, Adagrad, Adamax, Nadam, Ftrl, και Lion μειώνεται με κάποιες διακυμάνσεις στο σύνολο εκπαίδευσης, ενώ στο σύνολο επαλήθευσης η μείωση είναι πιο ομαλή.

Στο σύνολο δεδομένων California Housing, οι περισσότεροι αλγόριθμοι παρουσιάζουν σταθερή μείωση του μέσου τετραγωνικού σφάλματος (MSE), χωρίς ιδιαίτερες διακυμάνσεις.

Αλγόριθμοι όπως οι Adam, AdamW, RMSprop, Adamax και Nadam, παρά το γεγονός ότι ξεκινούν με υψηλότερες τιμές MSE, καταφέρνουν να μειώσουν σταδιακά το σφάλμα τους, καταλήγοντας σε χαμηλές τιμές. Αυτή η συμπεριφορά υποδεικνύει ότι προσαρμόζονται αποτελεσματικά στο συγκεκριμένο πρόβλημα παλινδρόμησης, παρέχοντας σταθερή βελτίωση κατά τη διάρκεια της εκπαίδευσης.

Αντίθετα, οι αλγόριθμοι Adadelta και FTRL διατηρούν σταθερά υψηλά επίπεδα MSE, γεγονός που δείχνει ότι δεν καταφέρνουν να προσαρμοστούν ικανοποιητικά στις απαιτήσεις του συνόλου δεδομένων. Αυτό σημαίνει ότι δεν καταφέρνουν να βελτιώσουν τη συνολική απόδοση του μοντέλου, υποδεικνύοντας την ακαταλληλότητά τους για αυτό το πρόβλημα.

Ο αλγόριθμος Lion, όπως καταγράφεται στα διαγράμματα, εμφανίζει επίσης σταθερή μείωση του MSE χωρίς διακυμάνσεις, καταλήγοντας σε χαμηλά επίπεδα. Αυτό υποδηλώνει ότι είναι από τους πιο αποδοτικούς αλγόριθμους για το συγκεκριμένο σύνολο δεδομένων, επιτυγχάνοντας υψηλή ορθότητα στις προβλέψεις του.



Εικόνα 40 - California Housing - MSE

Στις γραφικές παραστάσεις του μέσου τετραγωνικού σφάλματος (MSE) για τα σύνολα εκπαίδευσης και επαλήθευσης, παρατηρούμε ότι ο Adadelta παρουσιάζει σταδιακή γραμμική μείωση της απώλειας, ο Ftrl παραμένει σχεδόν σταθερός και παράλληλος με τον άξονα των εποχών, ενώ οι υπόλοιποι αλγόριθμοι εμφανίζουν λογαριθμική μείωση της απώλειας.

Οι SGD και Adagrad παρουσιάζουν πιο ομαλή καμπύλη με πιο αργή μείωση, σε σύγκριση με τους υπόλοιπους αλγόριθμους που φτάνουν γρήγορα σε χαμηλές τιμές MSE.

Ο πιο αποδοτικός αλγόριθμος για το σύνολο δεδομένων California Housing είναι ο Adamax, καθώς πέτυχε το χαμηλότερο μέσο τετραγωνικό σφάλμα (MSE), το οποίο ανέρχεται σε 2.43.
Οι αλγόριθμοι Nadam και AdamW παρουσίασαν παρόμοια απόδοση, με MSE 2.43 και για τους δύο, υποδεικνύοντας ότι είναι επίσης αρκετά αποδοτικοί.

Συνολικά, η χρήση του Adamax φαίνεται να προσφέρει την καλύτερη ισορροπία μεταξύ μείωσης του σφάλματος και σταθερότητας κατά την εκπαίδευση και επαλήθευση, καθιστώντας τον την καλύτερη επιλογή για αυτό το σύνολο δεδομένων.

4.4. Συμπεράσματα

4.4.1. Σχετικά εφαρμογή των Ν.Δ. Εμπροσθοδιάδοσης(FFNNs)

Σε σύνολα δεδομένων ταξινόμησης όπως το MNIST, το CIFAR-10, το CIFAR-100, και το Fashion MNIST, οι αλγόριθμοι Adam, AdamW, Adamax, και Nadam ξεχώρισαν για τις υψηλότερες αποδόσεις τους, ιδιαίτερα στο MNIST και στο Fashion MNIST, με ακρίβειες άνω του 88%-97%. Ωστόσο, αυτοί οι αλγόριθμοι παρουσίασαν σημάδια overfitting, όπως φαίνεται από τις αυξομειώσεις στην απόδοση στο σύνολο επαλήθευσης, κάτι που έγινε ιδιαίτερα εμφανές στο CIFAR-10 και το CIFAR-100, όπου οι ακρίβειες έπεσαν κάτω από το 50% και 20%, αντίστοιχα. Ο RMSprop βρέθηκε συχνά ανάμεσα στους κορυφαίους αλγόριθμους, παρουσιάζοντας καλή απόδοση, αλλά επίσης αντιμετώπισε παρόμοια προβλήματα αστάθειας στο σύνολο επαλήθευσης. Από την άλλη πλευρά, οι SGD, Adagrad, και Adadelta παρουσίασαν χαμηλότερες επιδόσεις σε ορθότητα, αλλά ήταν πιο ανθεκτικοί στο overfitting και διατήρησαν μεγαλύτερη σταθερότητα στα πιο δύσκολα σύνολα, όπως το CIFAR-100.

Στα δεδομένα παλινδρόμησης, όπως στο California Housing, οι αλγόριθμοι όπως ο Lion, ο RMSprop και οι παραλλαγές του Adam (AdamW, Adamax, Nadam) κατάφεραν να μειώσουν σημαντικά το μέσο τετραγωνικό σφάλμα (MSE), υποδεικνύοντας καλύτερη προσαρμογή σε δεδομένα παλινδρόμησης. Αντίθετα, οι Adadelta, Adagrad, και FTRL δυσκολεύτηκαν να προσαρμοστούν στις απαιτήσεις του συγκεκριμένου συνόλου δεδομένων, με αποτέλεσμα να παρουσιάζουν υψηλότερες τιμές σφάλματος.

Στα δεδομένα ταξινόμησης κειμένων, όπως στο IMDB και το Reuters, οι αλγόριθμοι RMSprop, Adam, AdamW, Adamax και Nadam εμφάνισαν σημαντικά προβλήματα overfitting, καθώς η ορθότητα έφτασε το 1 στο σύνολο εκπαίδευσης, αλλά στο σύνολο επαλήθευσης παρουσίασαν αστάθεια και αύξηση της απώλειας. Εναλλακτικά, οι SGD, Adadelta, Adagrad και Adafactor παρουσίασαν καλύτερη σταθερότητα, αλλά με χαμηλότερες επιδόσεις.

4.4.2. Σχετικά με την εφαρμογή των Συνελικτικών Νευρωνικών Δικτύων (CNNs)

Η συνολική απόδοση των αλγορίθμων βελτιστοποίησης σε διάφορα σύνολα δεδομένων ταξινόμησης με τη χρήση CNNs εμφανίζει ποικιλία στις επιδόσεις και τη σταθερότητα. Στα σύνολα δεδομένων MNIST, CIFAR-10, CIFAR-100 και Fashion MNIST, οι αλγόριθμοι όπως οι Adam, AdamW, Adamax και Nadam παραμένουν ανάμεσα στους πιο αποδοτικούς, καταγράφοντας σταθερά υψηλές ακρίβειες. Ειδικότερα:

Στο σύνολο των δεδομένων «MNIST», οι αλγόριθμοι Adam, AdamW, Adamax, και Nadam επιτυγχάνουν ακρίβειες πάνω από 99%, με τον RMSprop να φθάνει το 99.32%. Παρόλο που παρουσιάζουν μικρές διακυμάνσεις στο σύνολο επαλήθευσης, παραμένουν οι πιο αποδοτικοί εδώ. Οι αλγόριθμοι SGD, Adadelta και Adagrad δείχνουν μεγαλύτερη σταθερότητα και καλή γενίκευση, με επιδόσεις χαμηλότερες.

Στο σύνολο των δεδομένων CIFAR-10: οι αλγόριθμοι Adam, AdamW, Adamax, και RMSprop καταγράφουν τις καλύτερες επιδόσεις, με ακρίβειες γύρω στο 72%. Οι διακυμάνσεις εμφανίζουν overfitting, σε νέα δεδομένα. Αντίθετα, οι SGD, Adadelta και Adagrad παρουσιάζουν λιγότερες διακυμάνσεις, αλλά με ακρίβειες χαμηλότερες.

Η απόδοση όλων των αλγορίθμων στα δεδομένα CIFAR-100, παραμένει χαμηλή, με τις ακρίβειες να κυμαίνονται κάτω από το 40%. Οι αλγόριθμοι Adam, AdamW, Adamax και RMSprop εμφανίζονται και εδώ οι πιο αποδοτικοί με τους Adamax και Nadam να φτάνουν τις υψηλότερες ακρίβειες γύρω στο 38%.

Οι αλγόριθμοι Adam, AdamW, Adamax, και RMSprop, στα δεδομένα MNIST Fashion παρουσιάζουν τις καλύτερες επιδόσεις με ακρίβειες πάνω από 88%. Αυτοί οι αλγόριθμοι προσαρμόζονται καλά στα δεδομένα. Οι SGD, Adadelta και Adagrad ακολουθούν, με πιο σταθερή συμπεριφορά, αν και με χαμηλότερες επιδόσεις. Οι αλγόριθμοι Adafactor, FTRL, και Lion καταγράφουν τις χαμηλότερες επιδόσεις, αναγνωρίζοντας μόνο μία κλάση, με ακρίβειες γύρω στο 10%.

4.4.3. Σχετικά με την εφαρμογή των Ανατροφοδοτούμενων Νευρωνικών Δικτύων (RNNs)

Στα σύνολα δεδομένων CIFAR-10 και CIFAR-100, οι αλγόριθμοι Adam, AdamW, Adamax, και Nadam φτάνουν έως και 72% στο CIFAR-10 και 22.2% στο CIFAR-100, αλλά παρατηρούνται διακυμάνσεις στην απόδοσή τους στο σύνολο επαλήθευσης, με overfitting. Ο Adadelta, με ορθότητα μόλις 10.6% στο CIFAR- 10 και περίπου 1% στο CIFAR-100, δεν κρίνεται κατάλληλος

Ο Adamax ξεχωρίζει και πάλι, ιδιαίτερα στο CIFAR-100, ως ο πιο αποδοτικός αλγόριθμος, με χαμηλή ορθότητα 22.2%.

Στο σύνολο δεδομένων IMDB, παρατηρείται overfitting σε όλους τους αλγόριθμους. Η ορθότητα παραμένει στο 0.5, καθώς οι αλγόριθμοι δεν καταφέρνουν να διακρίνουν μεταξύ των κλάσεων. Οι Adam, AdamW, Nadam, RMSprop, και Adamax σημειώνουν διακυμάνσεις στην ορθότητα και την απώλεια, αλλά η συνολική απόδοση παραμένει χαμηλή, με αδυναμία γενίκευσης.

Παρόμοια προβλήματα εμφανίζονται και στο σύνολο δεδομένων Reuters, με την ορθότητα να φτάνει έως το 70%. Οι αλγόριθμοι Adam, AdamW, Nadam, RMSprop, και Adamax δείχνουν αστάθεια και μειωμένη ορθότητα στο σύνολο επαλήθευσης, γεγονός που υποδηλώνει ότι δεν καταφέρνουν να αποφύγουν το overfitting. Οι SGD, Adadelta, και Adagrad εμφανίζουν πιο σταθερή προσέγγιση, αλλά με χαμηλότερες ακρίβειες. Ο πιο αποδοτικός αλγόριθμος για το Reuters είναι ο RMSprop, με ορθότητα 51.46%.

Στο σύνολο δεδομένων παλινδρόμησης California Housing, οι αλγόριθμοι Adam, AdamW, RMSprop,Adamax, και Nadam επιτυγχάνουν σταθερή μείωση του MSE, καταλήγοντας σε χαμηλά επίπεδα σφάλματος. Ο Lion παρουσιάζει επίσης σταθερή μείωση του MSE, χωρίς ιδιαίτερες διακυμάνσεις, καταλήγοντας να είναι ένας από τους πιο αποδοτικούς αλγόριθμους για αυτό το σύνολο δεδομένων. Ο πιο αποδοτικός αλγόριθμος για το σύνολο δεδομένων California Housing είναι ο Adamax, με τη χαμηλότερη τιμή MSE στα 2.43. Οι αλγόριθμοι Adam, AdamW, Adamax, και Nadam διακρίνονται σταθερά για την απόδοσή τους, αν και παρουσιάζουν αστάθεια στο σύνολο επαλήθευσης λόγω overfitting.

4.4.4. Συνολική Αξιολόγηση

Ο Adamax συχνά ξεχωρίζει ως ο πιο αποδοτικός αλγόριθμος, με καλή ισορροπία μεταξύ ορθότητας και σταθερότητας, ιδιαίτερα σε σύνθετα σύνολα δεδομένων όπως το CIFAR-100 και το California Housing. Από την άλλη, οι SGD, Adadelta, και Adagrad καταγράφουν χαμηλότερες ακρίβειες, αλλά εμφανίζουν μεγαλύτερη ανθεκτικότητα στο overfitting, καθιστώντας τους πιο κατάλληλους για προβλήματα που απαιτούν υψηλή γενίκευση.

Με βάση όλες τις πληροφορίες που συγκεντρώθηκαν από τα διαφορετικά σύνολα δεδομένων και τις αρχιτεκτονικές που αναλύθηκαν (Convolutional Neural Networks, Recurrent Neural Networks, και Fully Connected Feedforward Neural Networks), ο πιο αποδοτικός αλγόριθμος βελτιστοποίησης συνολικά σε αυτήν την έρευνα, είναι ο Adamax.

Στα σύνολα δεδομένων MNIST και Fashion MNIST, ο Adamax σημείωσε εξαιρετικές επιδόσεις, με ακρίβειες άνω του 98% και 91.34% αντίστοιχα, καταγράφοντας σταθερότητα και υψηλή απόδοση σε αυτά τα σύνολα δεδομένων.

Στα σύνολα CIFAR-10 και CIFAR-100, ο Adamax ήταν ξανά ο πιο αποδοτικός αλγόριθμος, με ακρίβειες που έφτασαν το 72.72% και 22.2%, αντίστοιχα, ξεπερνώντας τους άλλους αλγόριθμους σε αυτά τα πιο δύσκολα σύνολα δεδομένων.

Στο σύνολο California Housing (Παλινδρόμηση), ο Adamax κατέγραψε επίσης τις χαμηλότερες τιμές MSE, γεγονός που δείχνει την ικανότητά του να προσαρμόζεται αποτελεσματικά σε προβλήματα παλινδρόμησης.

Στα σύνολα IMDB και Reuters, παρά την ύπαρξη overfitting σε ορισμένα προβλήματα ταξινόμησης κειμένων, ο Adamax εξακολουθούσε να επιτυγχάνει υψηλή απόδοση, ειδικά στο Reuters, όπου ξεχώρισε για την ισορροπία μεταξύ σταθερότητας και ορθότητας.

5. Γενικά Συμπεράσματα

Η Μηχανική Μάθηση ως ένα από τα κύρια υποσύνολα της Τεχνητής Νοημοσύνης διαθέτει ποικίλες μεθοδολογίες για την ανάλυση δεδομένων, την εκμάθηση και την εφαρμογή μεθοδολογιών για την επίλυση ποικίλων προβλημάτων. Με κυρίαρχα τα Νευρωνικά Δίκτυα - Ν.Δ. -, πολλαπλών και εξειδικευμένων τύπων, τα οποία κατηγοριοποιούνται κυρίως ως προς το είδος των δεδομένων που χειρίζονται και τον τελικό σκοπό εφαρμογής τους και τα οποία, ως επί το πλείστον, έχουν αποτελεσματικότητα σε πολύ υψηλά ποσοστά. Τέτοιες, για παράδειγμα, οικογένειες νευρωνικών δικτύων είναι τα Ν.Δ. Εμπροσθοδιάδοσης ή Οπισθοδιάδοσης, τα Επαναληπτικά ή Ανατροφοδοτούμενα (Recurrent) και τα Συνελικτικά (Convolutional) Νευρωνικά Δίκτυα. Από τις διάφορες βιβλιοθήκες για ελεύθερη χρήση η πιο φιλική, ευρέως γνωστή και ικανοποιητικά αποτελεσματική, με προγράμματα σχετικά με διάφορους τύπους νευρωνικών δικτύων είναι η Keras που χρησιμοποιήθηκε και στην παρούσα διπλωματική. Προσφέροντας αρκετούς αλγόριθμους βελτιστοποίησης Ν.Δ., η βιβλιοθήκη Keras κρίθηκε ιδιαίτερα χρήσιμη στην εφαρμογή σχεδόν όλων των τύπων βελτιστοποίησης που διαθέτει από την Κάθοδο Κλίσης - Gradient Descent και διάφορες παραλλαγές της έως την Διάδοση Μέσου Τετραγωνικού Σφάλματος -RMSprop και την Λέων -Lion, με σκοπό την σύγκριση της αποτελεσματικότητάς τους σε εκπαιδευτικό και επίπεδο επαλήθευσης της εφαρμογής των κύριων τύπων των Ν.Δ. Έχοντας σαν γνώμονα, την ad hoc εφαρμογή των διαφόρων κατηγοριών των νευρωνικών δικτύων, ανάλογα με τον τύπο των δεδομένων, χρησιμοποιήθηκαν στην παρούσα εργασία, διάφορες έγκυρες συλλογές ποικίλων τύπων δεδομένων που διατίθενται ελεύθερα, και κρίθηκαν κατάλληλες για τον σκοπό της σύγκρισης των μεθόδων βελτιστοποίησης Ν.Δ. σε ετερογενή σύνολα δεδομένων.

Ως προς τις εφαρμογές των Ν.Δ. και των μεθόδων βελτιστοπόιησής τους που χρησιμοποιήθηκαν στην παρούσα εργασία, το γενικό συμπέρασμα είναι ότι, η απόδοση των αλγορίθμων βελτιστοποίησης στα διάφορα σύνολα δεδομένων υποδεικνύει σημαντικές διαφορές στην αποτελεσματικότητα και σταθερότητά τους, τόσο κατά την εκπαίδευση όσο και κατά την επαλήθευση.

Πιο συγκεκριμένα, και ειδικότερα στα δίκτυα εμπροσθοδιάδοσης, οι παραλλαγές της μεθόδου βελτιστοποίησης Adam (AdamW, Adamax, Nadam) και ο RMSprop ήταν οι πιο αποδοτικοί αλγόριθμοι σε αρκετά από τα σύνολα δεδομένων που χρησιμοποιήθηκαν -παρόλο που παρουσίασαν υπερπροσαρμογή στο CIFAR και τα σύνολα ταξινόμησης κειμένων-, σε ταξινομήσεις εικόνων και παλινδρόμηση. Ωστόσο, αντιμετώπισαν προβλήματα overfitting σε πιο σύνθετα σύνολα δεδομένων όπως. Σε αντίθεση, αλγόριθμοι όπως οι SGD, Adadelta, και Adagrad με χαμηλότερες επιδόσεις σε ορθότητα παρουσιάζουν καλύτερη γενίκευση και σταθερότητα.

Στα Συνελικτικά Δικτυα, σχεδόν παρόμοια με τα δίκτυα εμπροσθοδιάδοσης, οι αλγόριθμοι Adam, AdamW, Adamax, και Nadam δίνουν υψηλές ακρίβειες σε ταξινομήσεις συνόλωνδεδομένων όπως το MNIST και το Fashion MNIST. Σε πιο ειδικά σύνολα όπως το CIFAR-10 και CIFAR-100, παρουσιάζουν προβλήματα γενίκευσης και overfitting. Οι SGD, Adadelta και Adagrad με χαμηλότερες επιδόσεις, έχουν μεγαλύτερη σταθερότητα και ανθεκτικότητα στο overfitting, για σύνθετα σύνολα δεδομένων.

Στα ανατροφοδοτούμενα Νευρωνικά Δίκτυα οι αλγόριθμοι Adam, AdamW, Adamax, και Nadam έχουν ακρίβειες πάνω από 88%, ιδιαίτερα στο MNIST με τον Ο RMSprop να ακολουθεί από κοντά. Όπως και στους δύο προηγούμενους τύπους Ν,Δ., οι αλγόριθμοι SGD, Adagrad, και Adadelta με χαμηλότερες ακρίβειες, προσφέρουν καλύτερη σταθερότητα στην ικανότητα γενίκευσης σε νέα δεδομένα. Ο Adamax αποδεικνύεται ως ο πιο αποδοτικός αλγόριθμος για αυτά τα σύνολα δεδομένων, με ορθότητα 98.05% στο MNIST και 87.2% στο Fashion MNIST.

Γενικότερα κάποιοι αλγόριθμοι, κυρίως της οικογένειας Adam αποδίδουν καλύτερα αλλά σε ορισμένα είδη δεδομένων, όπως τα MNIST και το Fashion MNIST και για τις τρεις αντιπροσωπευτικές κατηγορίες Ν.Δ. (Εμπροσθοδιάδοσης, Συνελικτικά και Ανατροφοδοτούμενα), που εφαρμόστηκαν. Επίσης, ο Adamax υπερέχει σταθερά ως ο πιο αποδοτικός αλγόριθμος σε πολλών τύπων σύνολα δεδομένων και αρχιτεκτονικές. Συνδυάζει την ισορροπία μεταξύ υψηλής ορθότητας, σταθερότητας, και μειωμένης απώλειας σε διάφορους τύπους προβλημάτων, καθιστώντας τον τον πιο αξιόπιστο αλγόριθμο στην παρούσα διπλωματική εργασία.

Βιβλιογραφία

(n.d.). Retrieved from https://keras.io/

Aggarwal, C. C. (2018). *Neural Networks and Deep Learning, A Textbook,*. ©c Springer International Publishing AG, part of Springer Nature .

Anirudha Ghosh, A. S. (2020,). Fundamental Concepts of Convolutional Neural Network. Retrieved from https://www.researchgate.net/publication/337401161_Fundamental_Concepts_of_Convolutional_Neural_Network

As. Vaswani, N. S. (2017). Attention is all you need. *31st Conference on Neural Information Processing Systems (NIPS 2017).*

Cheng J. P., D. L. (2016). Long Short-Term Memory-Networks for Machine Reading. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, (pp. 551, 561). Austin, Texas: Association for Computational Linguistics.

Deisenroth, M. P. (2020). *Mathematics for Machine Learning.* Cambridge University Press.

Duchi, J. H. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research, 12(Jul),*, 2121-2159. Retrieved from jmlr.org/papers/volume12/duchi11a/duchi11a.pdf

G. Toderici, D. V. (2017). Full Resolution Image Compression with Recurrent Neural Networks. *IEEE Conference on Computer Vision and Pattern Recognition.* arXiv:1608.05148v2 [cs.CV].

Han J., M. C. (1996). Optimization of Feedforword Neural Networks. *EngngApplic.Artif.Intell.Vol.9, No.2*, 109-116.

Hyatt, S. (2018). *Machine Learning Fundamentals.* Packt Publishing.

Ioffe, S. &. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *International Conference on Machine Learning*, (pp. 448-456). Retrieved from jmlr.org/proceedings/papers/v37/ioffe15.html

Izmailov, P. P. (2018). Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*. Retrieved from arxiv.org/abs/1803.05407

Jamhuri, M. (2024). *Understanding the Adam Optimization Algorithm.* MATLAB Central File Exchange.

Kingma, D. P. (2014). Adam: A method for stochastic optimization. *arXiv:1412.6980*. Retrieved from arxiv.org/abs/1412.6980

Kumar, C. G. (2018, September 6). *Machine Learning Types and Algorithms.* Retrieved from Medium: https://towardsdatascience.com/machine-learning-types-and-algorithms-d8b79545a6ec

Loshchilov, I. &. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*. Retrieved from arxiv.org/abs/1711.05101

McCulloch, W. P. (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics 5*, 115–133. Retrieved from https://doi.org/10.1007/BF02478259

Mueller J. P., M. L. (2016). *Machine Learning For Dummies.* John Wiley & Sons, Inc.

Reddi, S. J. (2019). On the convergence of adam and beyond. *arXiv:1904.09237. [Link: arxiv.org/abs/1904.09237*.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv:1609.04747*. Retrieved from arxiv.org/abs/1609.04747

Russell J. St., N. P. (2005). *Τεχνητή Νοημοσύνη. Μια σύγχρονη προσέγγιση, 2η Αμερ. Έκδοση.* Επιμ. Ελληνικής έκδοσης : Ρεφανίδης Γ., Εκδόσεις Κλειδάριθμος.

Smith, L. N. (2018). A bayesian perspective on generalization and stochastic gradient descent. *arXiv preprint arXiv:1710.06451*. Retrieved from : arxiv.org/abs/1710.06451]

Soper, D. (2023, September 23). Using an Opportunity Matrix to Select Centers for RBF Neural Networks. (C. S. Information Systems & Decision Sciences Department, Ed.) *Algorithms 16(10), 455*. Retrieved from https://doi.org/10.3390/a16100455

Statisticalinference. (n.d.). A numerical example of LSTMs. Retrieved from https://statisticalinterference.wordpress.com/2017/06/01/lstms-in-even-more-excruciating-detail/

Vl. Arnold, A. K. (n.d.). *https://en.wikipedia.org/wiki/Kolmogorov%E2%80%93Arnold_representation_theorem*. Retrieved from Kolmogorov - Arnold Representation Theorem.

You, Y. G. (2017). Large batch training of convolutional networks. Retrieved from arxiv.org/abs/1708.03888

Z. C. Lipton, J. B. (2015). A Critical Review of Recurrent Neural Networks for Sequence Learning. *arXiv:1506.00019v4 [cs.LG]* . Retrieved June 5, 2015

Zeiler, M. D. (2012). ADADELTA: An adaptive learning rate method. *arXiv:1212.5701*. Retrieved from arxiv.org/abs/1212.5701

Διαμαντάρας Κ., Μ. Δ. (2019). *Μηχανική Μάθηση*. Εκδόσεις Κλειδάριθμος.

Παραρτηματα

Κώδικες

Feedforward Neural Networks

MNIST dataset: https://www.kaggle.com/code/nafsikaperaki/ff-mnist

CIFAR-10 dataset: https://www.kaggle.com/code/nafsikaperaki/ff-cifar10

CIFAR-10 dataset: https://www.kaggle.com/code/nafsikaperaki/ff-cifar100

Fashion MNIST dataset: https://www.kaggle.com/code/nafsikaperaki/ff-fmnist

IMDB dataset: https://www.kaggle.com/code/nafsikaperaki/ff-imdb

Reuters dataset: https://www.kaggle.com/code/nafsikaperaki/ff-reuters

California-Housing dataset: https://www.kaggle.com/code/nafsikaperaki/ff-california-housing

Convolutional Neural Networks

MNIST dataset: https://www.kaggle.com/code/nafsikaperaki/cnn-mnist

CIFAR-10 dataset: https://www.kaggle.com/code/nafsikaperaki/cnn-cifar10

CIFAR-100 dataset: https://www.kaggle.com/code/nafsikaperaki/cnn-cifar100

Fashion MNIST dataset: https://www.kaggle.com/code/nafsikaperaki/cnn-fmnist

IMDB dataset: https://www.kaggle.com/code/nafsikaperaki/cnn-imdb

Reuters dataset: https://www.kaggle.com/code/nafsikaperaki/cnn-reuters

California-Housing dataset: https://www.kaggle.com/code/nafsikaperaki/cnn-california-housing

Reccurent Neural Networks

MNIST dataset: https://www.kaggle.com/code/nafsikaperaki/rnn-mnist

CIFAR-10 dataset: https://www.kaggle.com/code/nafsikaperaki/rnn-cifar10

CIFAR-100 dataset: https://www.kaggle.com/code/nafsikaperaki/rnn-cifar100

Fashion MNIST dataset: https://www.kaggle.com/code/nafsikaperaki/rnn-fmnist

IMDB dataset: https://www.kaggle.com/code/nafsikaperaki/rnn-imdb

Reuters dataset: https://www.kaggle.com/code/nafsikaperaki/rnn-reuters

California-Housing dataset: https://www.kaggle.com/code/nafsikaperaki/rnn-california-housing