



**ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ**

Σύγκριση Αλγορίθμων Βελτιστοποίησης της Οικογένειας Gradient Descent για Περιβάλλοντα Βαθιάς Μάθησης

Διπλωματική Εργασία

Μαλέρδος Χρήστος

Επιβλέπων: Κατσαρός Δημήτριος

Βόλος 2025

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά του γονείς μου που έδειξαν την αμέριστη στήριξή τους κατά τη διάρκεια των σπουδών μου και μου προσέφεραν τις απαραίτητες αρετές και αξίες που διέπουν τη ζωή μου.

Επίσης, θα ήθελα να ευχαριστήσω και τον επιβλέποντα καθηγητή μου Δημήτριο Κατσαρό για τη συνεργασία μας, καθώς η συμβολή του ήταν πολύτιμη και καθοριστική για την ολοκλήρωση της διπλωματικής μου εργασίας.

Τις ευχαριστίες μου θα ήθελα να εκφράσω και στους καθηγητές Ραφαηλίδη Δημήτριο και Θάνο Γεώργιο που δέχτηκαν να είναι μέλη της τριμελούς επιτροπής αξιολόγησης της διπλωματικής μου εργασίας.

Τέλος, δε θα μπορούσα να παραλείψω και τους φίλους και συμφοιτητές μου, μαζί με τους οποίους έκανα αυτό το ταξίδι, προσπερνώντας κάθε δυσκολία και εμπόδιο.

ΥΠΕΥΘΥΝΗ ΔΗΛΩΣΗ ΠΕΡΙ ΑΚΑΔΗΜΑΪΚΗΣ ΔΕΟΝΤΟΛΟΓΙΑΣ ΚΑΙ ΠΝΕΥΜΑΤΙΚΩΝ ΔΙΚΑΙΩΜΑΤΩΝ

«Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ρητά ότι η παρούσα διπλωματική εργασία, καθώς και τα ηλεκτρονικά αρχεία και πηγαίοι κώδικες που αναπτύχθηκαν ή τροποποιήθηκαν στα πλαίσια αυτής της εργασίας, αποτελεί αποκλειστικά προϊόν προσωπικής μου εργασίας, δεν προσβάλλει κάθε μορφής δικαιώματα διανοητικής ιδιοκτησίας, προσωπικότητας και προσωπικών δεδομένων τρίτων, δεν περιέχει έργα/εισφορές τρίτων για τα οποία απαιτείται άδεια των δημιουργών/δικαιούχων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον και πληρούν τους κανόνες της επιστημονικής παράθεσης. 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. Αναλαμβάνω πλήρως, ατομικά και προσωπικά, όλες τις νομικές και διοικητικές συνέπειες που δύναται να προκύψουν στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δεν μου ανήκει διότι είναι προϊόν λογοκλοπής».

Ο Δηλών

Μαλέρδος Χρήστος

Ιούνιος 2025

Περίληψη

Η παρούσα διπλωματική εργασία εξετάζει και συγκρίνει διάφορους αλγορίθμους βελτιστοποίησης που ανήκουν στην οικογένεια του Gradient Descent, με έμφαση στην εφαρμογή τους σε περιβάλλοντα βαθιάς μάθησης (Deep Learning). Η έρευνα επικεντρώνεται στις πιο γνωστές παραλλαγές, όπως οι Stochastic Gradient Descent (SGD), Mini-Batch Gradient Descent, καθώς και οι εξελιγμένοι βελτιστοποιητές Momentum, RMSProp και Adam. Αναλύονται τα θεωρητικά τους χαρακτηριστικά, οι επιδόσεις τους σε διαφορετικά σενάρια εκπαίδευσης, καθώς και τα πλεονεκτήματα και μειονεκτήματά τους όσον αφορά την ακρίβεια, τη σύγκλιση και την απόδοση σε μεγάλα datasets.

Για την πειραματική αξιολόγηση, σχεδιάστηκαν και υλοποιήθηκαν δύο μοντέλα νευρωνικών δικτύων μέσω της πλατφόρμας PyTorch, τα οποία εκπαιδεύτηκαν στο σύνολο δεδομένων MNIST. Η απόδοσή τους εξετάστηκε με βάση μεταβλητές όπως ο χρόνος εκπαίδευσης, η ακρίβεια ταξινόμησης, και η συμπεριφορά σε διαφορετικές τιμές υπερπαραμέτρων.

Τα αποτελέσματα αναδεικνύουν τον κρίσιμο ρόλο της επιλογής του κατάλληλου αλγορίθμου, καθώς ο κάθε βελτιστοποιητής παρουσιάζει διαφορετικά πλεονεκτήματα ανάλογα με την αρχιτεκτονική και τις απαιτήσεις του προβλήματος. Ιδιαίτερα, ο Adam ξεχωρίζει για την ευελιξία και τη σταθερή απόδοσή του σε σύνθετα περιβάλλοντα. Η εργασία καταλήγει με προτάσεις για μελλοντική επέκταση της μελέτης, συμπεριλαμβανομένης της εφαρμογής σε πιο απαιτητικά datasets, της αξιοποίησης προηγμένων τεχνικών κανονικοποίησης και της διερεύνησης σύγχρονων υβριδικών αλγορίθμων βελτιστοποίησης.

Περιεχόμενα

Κεφάλαιο 1: Εισαγωγή.....	8
Gradient Descent	10
Παραλλαγές Gradient Descent	13
Batch Gradient Descent	14
Stochastic Gradient Descent	16
Mini Batch Gradient Descent	19
Βελτιστοποιητές.....	22
Momentum	22
RMS Prop (Root Mean Square Propagation).....	24
Adam (Adaptive Moment Estimation)	27
Κεφάλαιο 2: Ανασκόπηση.....	30
Κεφάλαιο 3: Υλοποίηση.....	33
Ορισμός Μοντέλων.....	33
Αλγόριθμοι Βελτιστοποίησης	34
Παράμετροι Εκπαίδευσης	35
Δομή Αποτελεσμάτων.....	35
Σύγκριση και Συμπεράσματα.....	36
Γράφημα 1	37
Γενική Ανάλυση.....	37
Ανάλυση ανά αλγόριθμο βελτιστοποίησης.....	38
Συγκρίσεις μεταξύ των δύο μοντέλων.....	40

Συμπεράσματα.....	40
Γράφημα 2	41
Γενική Ανάλυση.....	41
Ανάλυση ανά αλγόριθμο βελτιστοποίησης.....	42
Συγκρίσεις μεταξύ των δύο μοντέλων.....	43
Συμπεράσματα.....	43
Γράφημα 3	45
Γενική Ανάλυση.....	45
Ανάλυση ανά αλγόριθμο βελτιστοποίησης.....	45
Συγκριτική Ανάλυση.....	47
Συμπεράσματα.....	48
Γράφημα 4	49
Γενική Ανάλυση.....	49
Ανάλυση ανά αλγόριθμο βελτιστοποίησης.....	49
Συγκριτική Ανάλυση.....	51
Συμπεράσματα.....	51
Ανάλυση κώδικα	52
Ανάλυση Δεδομένων	54
Κεφάλαιο 4: Συμπεράσματα.....	57
Ακρίβεια	57
Χρόνος εκπαίδευσης.....	57
Επίδραση των υπερπαραμέτρων.....	58
Συγκριτική ανάλυση αλγορίθμων βελτιστοποίησης.....	59
Συγκριτική ανάλυση μοντέλων	59

Συσχετίσεις μεταβλητών.....	59
Συμπεράσματα από τα αποτελέσματα.....	60
Επόμενα Βήματα.....	60
Κεφάλαιο 5: Προκλήσεις – Επόμενα Βήματα.....	62
Προκλήσεις που εντοπίστηκαν.....	62
Επόμενα βήματα.....	63
Βελτίωση της διαδικασίας εκπαίδευσης	63
Αξιοποίηση νεότερων τεχνικών	63
Ανάλυση νέων σεναρίων	64
Αξιοποίηση υπολογιστικών πόρων.....	64
Εκπαίδευση και αξιολόγηση	64
Βιβλιογραφία	65

Εικόνα 1: Τυχαίες Τιμές.....	12
Εικόνα 2: Batch Gradient Descent Απεικόνιση	16
Εικόνα 3: Stochastic GD απεικόνιση	19
Εικόνα 4: Mini-Batch GD	21
Εικόνα 5: Σύγκριση GD με / χωρίς Momentum	24
Εικόνα 6: RMSProp.....	27
Εικόνα 7: Γράφημα 1.....	37
Εικόνα 8: Γράφημα 2.....	41
Εικόνα 9: Γράφημα 3.....	45
Εικόνα 10: Γράφημα 4.....	49

Κεφάλαιο 1: Εισαγωγή

Η Gradient Descent είναι μια από τις πιο διαδεδομένες και σημαντικές αλγοριθμικές τεχνικές βελτιστοποίησης, η οποία χρησιμοποιείται για την εκπαίδευση νευρωνικών δικτύων και άλλων αλγορίθμων μηχανικής μάθησης. Βασικός της σκοπός είναι να ελαχιστοποιήσει μια συνάρτηση κόστους (loss function) μέσω επαναληπτικών ενημερώσεων των παραμέτρων του μοντέλου. Η κεντρική ιδέα πίσω από τη Gradient Descent είναι η χρήση των παραγώγων (gradients) για να καθοριστεί η κατεύθυνση και το μέγεθος της αλλαγής στις παραμέτρους, με στόχο τη βελτίωση της απόδοσης του μοντέλου.

Στα νευρωνικά δίκτυα βαθιάς μάθησης (Deep Neural Networks), η Gradient Descent χρησιμοποιείται για την εκπαίδευση των δικτύων μέσω της διαδικασίας backpropagation. Κατά τη διάρκεια της εκπαίδευσης, ο αλγόριθμος υπολογίζει την τιμή της συνάρτησης κόστους, η οποία μετρά πόσο καλά το δίκτυο προβλέπει ή ταξινομεί τα δεδομένα. Στη συνέχεια, οι παράγωγοι της συνάρτησης κόστους ως προς τις παραμέτρους (βάρη και bias) χρησιμοποιούνται για την ενημέρωση των παραμέτρων, μειώνοντας την απόκλιση από την ιδανική λύση. Αυτή η επαναληπτική διαδικασία επιτρέπει στα νευρωνικά δίκτυα να "μαθαίνουν" από τα δεδομένα, οδηγώντας σε καλύτερες προβλέψεις.

Υπάρχουν αρκετές παραλλαγές της Gradient Descent, οι οποίες αναπτύχθηκαν για να ξεπεραστούν προκλήσεις που παρουσιάζονται σε μεγάλα ή πολύπλοκα datasets, όπως είναι η ταχύτητα σύγκλισης, η ευαισθησία στις υπερπαραμέτρους, και η παγίδευση σε τοπικά ελάχιστα. Αυτές οι παραλλαγές περιλαμβάνουν:

1. Stochastic Gradient Descent (SGD): Σε αυτήν την παραλλαγή, η ενημέρωση των παραμέτρων γίνεται για κάθε μεμονωμένο δείγμα από το dataset. Αν και αυτό μπορεί να προσθέσει θόρυβο στις ενημερώσεις, επιτρέπει ταχύτερη επεξεργασία και συχνά βοηθά στη διαφυγή από τοπικά ελάχιστα.

2. Mini-Batch Gradient Descent: Συνδυάζει χαρακτηριστικά από το βασικό Gradient Descent και το SGD, καθώς χρησιμοποιεί μικρά σύνολα δεδομένων (mini-batches) για την ενημέρωση των παραμέτρων. Αυτή η μέθοδος είναι ιδιαίτερα αποδοτική και χρησιμοποιείται ευρέως σε πραγματικά συστήματα βαθιάς μάθησης.
3. Momentum: Εισάγει μια αδράνεια στις ενημερώσεις των παραμέτρων, προσθέτοντας ένα κλάσμα της προηγούμενης ενημέρωσης στην τρέχουσα. Αυτό βοηθά στη γρήγορη κατεύθυνση προς την επιθυμητή ελάχιστη τιμή, ξεπερνώντας τις διακυμάνσεις στις ενημερώσεις.
4. Adam (Adaptive Moment Estimation): Μία από τις πιο δημοφιλείς και αποτελεσματικές παραλλαγές, συνδυάζει τις ιδιότητες του Momentum και του RMSProp. Υπολογίζει προσαρμοστικά learning rates για κάθε παράμετρο, καθιστώντας τον αλγόριθμο κατάλληλο για προβλήματα με μεγάλη ποικιλία χαρακτηριστικών.

Άλλοι αλγόριθμοι βελτιστοποίησης που ανήκουν στην οικογένεια της Gradient Descent περιλαμβάνουν το RMSProp, το Adagrad, το Adadelta, και τον Nesterov Accelerated Gradient (NAG). Κάθε ένας από αυτούς έχει τη δική του θεωρητική βάση, μοναδικά χαρακτηριστικά και συγκεκριμένες εφαρμογές, οι οποίες τον καθιστούν προτιμητέο σε διαφορετικά σενάρια. Για παράδειγμα, το RMSProp είναι ιδιαίτερα κατάλληλο για μη στάσιμες συναρτήσεις κόστους, ενώ το Adagrad προσφέρει πλεονεκτήματα σε αραιά δεδομένα (sparse data).

Η ανάλυση και κατανόηση αυτών των παραλλαγών αποτελεί κρίσιμο βήμα για την επιλογή του κατάλληλου αλγορίθμου ανάλογα με το πρόβλημα. Στη βιβλιογραφία, η καταγραφή των χαρακτηριστικών τους δείχνει ότι η χρήση τους εξαρτάται συχνά από το μέγεθος των δεδομένων, την πολυπλοκότητα του μοντέλου και τις απαιτήσεις ακρίβειας. Οι περισσότερες μελέτες επικεντρώνονται στην επίδραση του learning rate, την ευαισθησία σε hyperparameters, και την επίδοση σε διάφορες δομές νευρωνικών δικτύων.

Μελετώντας την απόδοση κάθε παραλλαγής, μπορούμε να βγάλουμε χρήσιμα συμπεράσματα για τις επιδόσεις τους σε προβλήματα πραγματικού κόσμου, όπως η αναγνώριση εικόνων, η ανάλυση φυσικής γλώσσας, και τα αυτόνομα συστήματα. Επιπλέον, η κατανόηση των

θεωρητικών πτυχών της Gradient Descent συμβάλλει στη διαμόρφωση νέων βελτιστοποιημένων αλγορίθμων για τα σύγχρονα συστήματα βαθιάς μάθησης.

Gradient Descent

Η Gradient Descent αποτελεί μία από τους πιο βασικούς και ευρέως χρησιμοποιούμενους αλγορίθμους τόσο σε shallow neural networks όσο και σε περιβάλλοντα βαθιάς μηχανικής μάθησης (Deep Neural Networks). Είναι ένας αλγόριθμος βελτιστοποίησης που εφαρμόζεται για να βρεθούν οι κατάλληλες παράμετροι ενός μοντέλου, έτσι ώστε να ελαχιστοποιηθεί μία συνάρτηση κόστους, γνωστή ως cost function. Η συνάρτηση κόστους αντιπροσωπεύει το σφάλμα μεταξύ των προβλέψεων του μοντέλου και των πραγματικών τιμών, παρέχοντας ένα μέτρο της ποιότητας των προβλέψεων.

Η λειτουργία της Gradient Descent βασίζεται στην παρατήρηση ότι η παράγωγος μιας συνάρτησης προσδιορίζει τη κατεύθυνση της μέγιστης ανόδου ή καθόδου της. Επομένως, μέσω της χρήσης των παραγώγων, η Gradient Descent ενημερώνει τις παραμέτρους του μοντέλου βήμα-βήμα, ώστε να μειώσει το σφάλμα. Ουσιαστικά, ακολουθεί μια πορεία προς την κατεύθυνση όπου η τιμή της cost function μειώνεται πιο γρήγορα, προσπαθώντας να φτάσει σε ένα τοπικό ή, ιδανικά, στο παγκόσμιο ελάχιστο της συνάρτησης.

Στα shallow neural networks, η Gradient Descent χρησιμοποιείται για την προσαρμογή των βαρών και των bias του μοντέλου, βελτιστοποιώντας τη διαδικασία εκπαίδευσης. Ωστόσο, σε περιβάλλοντα βαθιάς μάθησης (Deep Learning), όπου τα δίκτυα περιέχουν πολλαπλά επίπεδα (layers) και πολύπλοκους υπολογισμούς, η Gradient Descent έχει ακόμη μεγαλύτερη σημασία. Σε τέτοιες περιπτώσεις, η συνάρτηση κόστους μπορεί να έχει πολλές διαφορετικές τοπικές ελάχιστες τιμές, γεγονός που καθιστά τον αλγόριθμο σημαντικό εργαλείο για την εκπαίδευση των νευρωνικών δικτύων.

Ο βασικός σκοπός της Gradient Descent είναι να μειώσει το error function, το οποίο υπολογίζεται για κάθε παράδειγμα δεδομένων (data sample) στο dataset. Για να το πετύχει αυτό, ο αλγόριθμος ενημερώνει επαναληπτικά τις παραμέτρους του μοντέλου, μειώνοντας την

απόσταση μεταξύ των προβλεπόμενων τιμών και των πραγματικών τιμών. Αυτή η διαδικασία περιλαμβάνει τη χρήση του learning rate, μιας υπερπαραμέτρου που καθορίζει το μέγεθος του βήματος που θα κάνει ο αλγόριθμος σε κάθε ενημέρωση. Η επιλογή ενός κατάλληλου learning rate είναι κρίσιμη, καθώς ένα πολύ μικρό learning rate μπορεί να κάνει τη διαδικασία βελτιστοποίησης πολύ αργή, ενώ ένα πολύ μεγάλο learning rate μπορεί να οδηγήσει σε αστάθεια ή να παρακάμψει το βέλτιστο σημείο.

Η Gradient Descent βρίσκει εφαρμογή σε ένα ευρύ φάσμα προβλημάτων, από την ανάλυση δεδομένων και την πρόβλεψη χρονοσειρών, έως την αναγνώριση εικόνας και την ανάλυση φυσικής γλώσσας. Στο πλαίσιο των shallow neural networks, χρησιμοποιείται για την εκπαίδευση δικτύων που έχουν περιορισμένο αριθμό επιπέδων, όπως τα perceptrons ή τα feed-forward networks. Στα deep neural networks, η Gradient Descent συνδυάζεται συχνά με μεθόδους όπως το backpropagation, όπου οι παράγωγοι των παραμέτρων υπολογίζονται μέσω της αλυσίδας διαφορικών παραγώγων (chain rule).

Επιπλέον, υπάρχουν πολλές παραλλαγές της Gradient Descent που προσαρμόζονται στις ανάγκες διαφορετικών προβλημάτων και datasets. Για παράδειγμα, η Stochastic Gradient Descent (SGD) ενημερώνει τις παραμέτρους χρησιμοποιώντας κάθε μεμονωμένο δείγμα του dataset, προσθέτοντας θόρυβο αλλά επιταχύνοντας τη διαδικασία. Αντίστοιχα, η Mini-Batch Gradient Descent χρησιμοποιεί μικρά σύνολα δεδομένων (mini-batches) για πιο ισορροπημένες ενημερώσεις. Άλλες μέθοδοι, όπως το Momentum, το RMSProp, και ο Adam, προσφέρουν βελτιώσεις στην ταχύτητα σύγκλισης και στην αντιμετώπιση τοπικών ελαχίστων.

Η κατανόηση της Gradient Descent και των παραλλαγών της είναι θεμελιώδης για οποιονδήποτε ασχολείται με τη μηχανική μάθηση και την τεχνητή νοημοσύνη, καθώς αποτελεί τη βάση για την εκπαίδευση και τη βελτιστοποίηση πολλών σύγχρονων μοντέλων. Μέσα από τη σωστή εφαρμογή της, οι ερευνητές και οι επαγγελματίες μπορούν να δημιουργούν πιο αποδοτικά και ακριβή μοντέλα, τα οποία προσαρμόζονται στις απαιτήσεις κάθε εφαρμογής.

Ο τύπος μίας τυπικής συνάρτησης κόστους είναι ο παρακάτω (Γραμμική Παλινδρόμηση):

$$MSE = \frac{1}{N} \sum_{i=1}^N (Y_i - Y'_i)^2$$

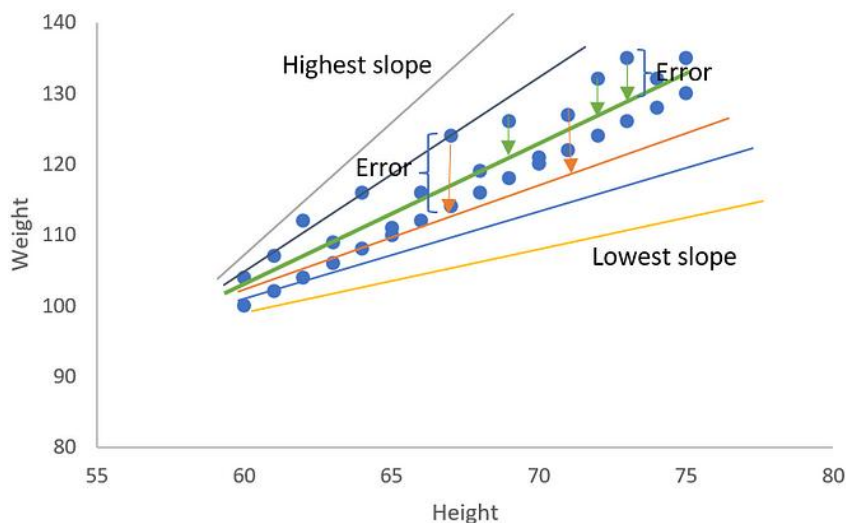
Όπου N είναι ο αριθμός των δεδομένων που έχουμε, Y_i είναι η πραγματική τιμή και Y'_i η πρόβλεψη το μοντέλου

Ο στόχος μας είναι η MSE να ελαχιστοποιηθεί όσο το δυνατόν περισσότερο (η ελαχιστοποίηση της συνάρτησης κόστους που αναφέραμε παραπάνω). Η παραπάνω εξίσωση μετατρέπεται σε:

$$MSE = \frac{1}{2N} \sum_{i=1}^N (Y_i - (mx_i + c))^2$$

Όπου το $mx_i + c$ είναι γραμμική παλινδρόμηση που αναφέραμε.

Η παραπάνω ελαχιστοποίηση είναι ουσιαστικά η προσπάθεια εύρεσης των τιμών m και c ώστε να έχουμε ελαχιστοποίηση.



Εικόνα 1: Τυχαίες Τιμές

Παραλλαγές Gradient Descent

Υπάρχουν πολλές παραλλαγές της Gradient Descent που έχουν αναπτυχθεί για να βελτιώσουν την απόδοση του αλγορίθμου στην εύρεση της ελαχιστοποίησης της cost function. Κάθε παραλλαγή προσφέρει συγκεκριμένα πλεονεκτήματα και είναι κατάλληλη για διαφορετικά προβλήματα και σύνολα δεδομένων, ανάλογα με τις απαιτήσεις και τα χαρακτηριστικά του κάθε περιβάλλοντος. Οι βασικοί τύποι αυτών των παραλλαγών περιλαμβάνουν:

1. Stochastic Gradient Descent (SGD): Στον τύπο αυτό, η ενημέρωση των παραμέτρων του μοντέλου γίνεται με βάση ένα μόνο δείγμα δεδομένων από το dataset σε κάθε βήμα. Αυτό προσφέρει ταχύτητα στις ενημερώσεις, αλλά μπορεί να οδηγήσει σε θόρυβο κατά τη διαδικασία βελτιστοποίησης, καθιστώντας τον αλγόριθμο λιγότερο σταθερό. Ωστόσο, το χαρακτηριστικό αυτό συχνά επιτρέπει την αποφυγή τοπικών ελαχίστων, καθιστώντας το SGD ιδιαίτερα αποτελεσματικό για πολύπλοκα datasets.
2. Mini-Batch Gradient Descent: Αυτή η μέθοδος είναι ένας συνδυασμός του SGD και της παραδοσιακής Gradient Descent. Η ενημέρωση των παραμέτρων βασίζεται σε μικρά υποσύνολα του dataset (mini-batches), που προσφέρουν μια ισορροπία μεταξύ ταχύτητας και ακρίβειας. Τα mini-batches μειώνουν τη στοχαστικότητα του SGD, διατηρώντας όμως τη δυνατότητα για αποδοτικές ενημερώσεις παραμέτρων σε μεγάλα datasets.
3. Momentum: Η μέθοδος αυτή προσθέτει μια συνιστώσα που λαμβάνει υπόψη τη "φορμή" από προηγούμενα βήματα, δημιουργώντας μια πιο ομαλή κατεύθυνση βελτιστοποίησης. Με αυτόν τον τρόπο, ο αλγόριθμος μπορεί να ξεπεράσει περιοχές χαμηλής καμπυλότητας ή τοπικά ελάχιστα, συγκλίνοντας πιο γρήγορα σε ένα παγκόσμιο ελάχιστο.
4. Adam (Adaptive Moment Estimation): Πρόκειται για μία από τις πιο δημοφιλείς παραλλαγές, που συνδυάζει τα πλεονεκτήματα του Momentum και του RMSProp. Ο Adam προσαρμόζει αυτόματα το learning rate για κάθε παράμετρο, γεγονός που τον καθιστά ιδιαίτερα αποδοτικό για πολύπλοκα προβλήματα και μεγάλα datasets. Η

προσαρμοστική αυτή ιδιότητα τον καθιστά έναν από τους προτιμώμενους αλγόριθμους σε σύγχρονες εφαρμογές βαθιάς μάθησης.

5. RMSProp (Root Mean Square Propagation): Ο αλγόριθμος αυτός είναι ιδανικός για μη στάσιμες συναρτήσεις κόστους, καθώς χρησιμοποιεί έναν κινούμενο μέσο όρο των τετραγώνων των gradients για να προσαρμόζει το learning rate. Αυτό βοηθά στη σταθεροποίηση της διαδικασίας βελτιστοποίησης και είναι κατάλληλος για πολύπλοκα προβλήματα.
6. Adagrad και Adadelta: Αυτές οι παραλλαγές είναι ιδιαίτερα χρήσιμες για προβλήματα με αραιά δεδομένα (sparse data). Το Adagrad προσαρμόζει το learning rate λαμβάνοντας υπόψη τις προηγούμενες ενημερώσεις, ενώ το Adadelta βελτιώνει περαιτέρω τη διαδικασία διατηρώντας σταθερότερο learning rate κατά τη διάρκεια της εκπαίδευσης.

Κάθε μία από αυτές τις παραλλαγές εξυπηρετεί συγκεκριμένες ανάγκες και προκλήσεις, προσφέροντας μια πιο ευέλικτη προσέγγιση στη διαδικασία βελτιστοποίησης. Η επιλογή της κατάλληλης παραλλαγής εξαρτάται από τη φύση του προβλήματος, το μέγεθος και την ποιότητα του dataset, καθώς και από την πολυπλοκότητα του μοντέλου που εκπαιδεύεται.

Batch Gradient Descent

Η κλασική έκδοση του Gradient Descent, ή αλλιώς όπως είναι γνωστό και ως Batch Gradient Descent, αποτελεί την παραδοσιακή μορφή του αλγορίθμου. Στην παραλλαγή αυτή, χρησιμοποιείται ολόκληρο το σύνολο δεδομένων (dataset) για τον υπολογισμό της συνάρτησης κόστους (cost function) και την ελαχιστοποίησή της. Η διαδικασία αυτή περιλαμβάνει τον υπολογισμό του μέσου σφάλματος (mean error) για όλα τα δείγματα δεδομένων στο dataset, και στη συνέχεια πραγματοποιείται μια ενημέρωση των παραμέτρων του μοντέλου. Αυτή η μέθοδος βασίζεται στην ιδέα ότι ο μέσος όρος του σφάλματος παρέχει μια ακριβή αναπαράσταση της συνολικής απόκλισης του μοντέλου από τις πραγματικές τιμές, επιτρέποντας στον αλγόριθμο να κάνει μια πιο σταθερή ενημέρωση.

Η χρήση ολόκληρου του dataset για κάθε ενημέρωση των παραμέτρων παρέχει σημαντικά πλεονεκτήματα όσον αφορά τη σταθερότητα της διαδικασίας βελτιστοποίησης. Η σταθερότητα αυτή εξασφαλίζει ότι η συνάρτηση κόστους μειώνεται σταθερά σε κάθε βήμα, χωρίς να υπάρχουν σημαντικές διακυμάνσεις. Ωστόσο, αυτό το πλεονέκτημα συνοδεύεται από σημαντικές προκλήσεις. Καθώς το μέγεθος των datasets αυξάνεται, το κόστος εκτέλεσης της κλασικής Gradient Descent αυξάνεται δραματικά. Κάθε βήμα απαιτεί την επεξεργασία και ανάλυση όλων των δεδομένων, γεγονός που μπορεί να οδηγήσει σε εξαιρετικά υψηλό υπολογιστικό φορτίο.

Ένα από τα βασικά μειονεκτήματα αυτής της προσέγγισης είναι η ανάγκη για σημαντικούς υπολογιστικούς πόρους, όπως μνήμη και ισχύ επεξεργαστή (CPU). Για μεγάλα datasets, το κόστος σε χρόνο και πόρους γίνεται συχνά απαγορευτικό, ειδικά όταν τα δεδομένα δεν μπορούν να αποθηκευτούν στη μνήμη RAM και πρέπει να διαχειρίζονται σε τμηματικά μέρη από τον δίσκο. Επίσης, ο χρόνος που απαιτείται για την ολοκλήρωση κάθε βήματος βελτιστοποίησης αυξάνεται σημαντικά, καθιστώντας την Gradient Descent αργή και μη πρακτική για εφαρμογές σε πραγματικό χρόνο ή σε μεγάλα μοντέλα βαθιάς μάθησης.

Ένα ακόμα πρόβλημα που αντιμετωπίζει η κλασική Gradient Descent είναι η ύπαρξη τοπικών ελαχίστων στη συνάρτηση κόστους. Σε πολύπλοκα μοντέλα και συναρτήσεις κόστους με πολλαπλά τοπικά ελάχιστα, ο αλγόριθμος μπορεί να "παγιδευτεί" σε ένα τοπικό ελάχιστο, αντί να φτάσει στο παγκόσμιο ελάχιστο. Αυτό μπορεί να οδηγήσει σε χαμηλότερη απόδοση του μοντέλου, καθώς το κόστος της συνάρτησης κόστους δεν μειώνεται όσο θα μπορούσε, επηρεάζοντας αρνητικά τη διαδικασία εκμάθησης.

Επιπλέον, η ταχύτητα του αλγορίθμου είναι συχνά περιορισμένη, ιδιαίτερα σε εφαρμογές που απαιτούν μεγάλη ακρίβεια ή όταν τα δεδομένα είναι δυναμικά και αλλάζουν με την πάροδο του χρόνου. Η χρήση ολόκληρου του dataset για κάθε ενημέρωση δεν επιτρέπει στον αλγόριθμο να προσαρμόζεται γρήγορα στις αλλαγές στα δεδομένα. Έτσι, σε περιπτώσεις όπου τα δεδομένα είναι μη στατικά, η Gradient Descent μπορεί να αποδειχθεί αναποτελεσματική.

Παρά τις προκλήσεις αυτές, η κλασική Gradient Descent παραμένει σημαντική για την κατανόηση της λειτουργίας και της θεωρητικής βάσης της βελτιστοποίησης στις μηχανές

μάθησης. Παρέχει τη βάση για την ανάπτυξη πιο αποδοτικών παραλλαγών, όπως η Stochastic Gradient Descent και η Mini-Batch Gradient Descent, που αντιμετωπίζουν ορισμένες από τις αδυναμίες της κλασικής μεθόδου. Ωστόσο, η κλασική Gradient Descent είναι ακόμα χρήσιμη σε μικρότερα datasets, όπου οι περιορισμοί των πόρων και της ταχύτητας δεν αποτελούν εμπόδιο, και η ακρίβεια είναι η πρωταρχική προτεραιότητα.

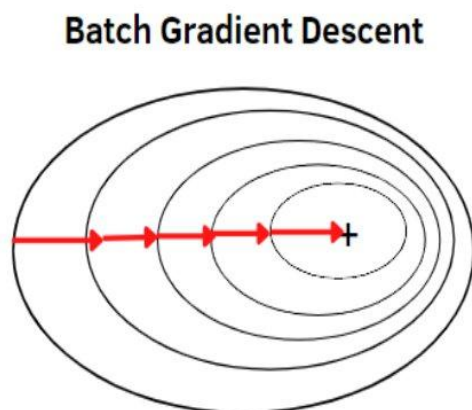
Συνοπτικά:

Θετικά:

- Κλασσικός τρόπος λειτουργίας – μειωμένη πολυπλοκότητα στη κατανόηση

Αρνητικά:

- Σε περίπτωση μεγάλου μεγέθους δεδομένων resources / χρόνος εκτέλεσης επηρεάζεται
- Ύπαρξη τοπικών ελαχίστων επηρεάζει την λειτουργία



Εικόνα 2: Batch Gradient Descent Απεικόνιση

Stochastic Gradient Descent

Η Stochastic Gradient Descent (SGD) είναι μια παραλλαγή της κλασικής Gradient Descent που στοχεύει κυρίως στην επίλυση των προβλημάτων που παρατηρούνται όταν το dataset είναι μεγάλο και η επεξεργασία του απαιτεί πολύ χρόνο και υπολογιστικούς πόρους. Σε αντίθεση με την κλασική Gradient Descent, όπου χρησιμοποιείται ολόκληρο το dataset για κάθε ενημέρωση

των παραμέτρων, η SGD πραγματοποιεί ενημερώσεις χρησιμοποιώντας μόνο ένα τυχαίο δείγμα δεδομένων σε κάθε βήμα. Με αυτόν τον τρόπο, ο αλγόριθμος επιτυγχάνει σημαντική βελτίωση της ταχύτητας και της δυνατότητας επεξεργασίας μεγάλων δεδομένων.

Η βασική αρχή της SGD έγκειται στην αντικατάσταση της χρήσης ολόκληρου του dataset με ένα μόνο δείγμα δεδομένων για τον υπολογισμό της παραγώγου της συνάρτησης κόστους. Αντί να υπολογίζει τον μέσο όρο όλων των σφαλμάτων, όπως γίνεται στην κλασική Gradient Descent, η SGD χρησιμοποιεί την πληροφορία που παρέχει ένα μόνο σημείο δεδομένων για να εκτιμήσει την κατεύθυνση της ελαχιστοποίησης. Αυτή η εκτίμηση βασίζεται στο γεγονός ότι, αν και κάθε μεμονωμένο δείγμα μπορεί να περιέχει κάποιο θόρυβο, σε βάθος χρόνου οι ενημερώσεις συγκλίνουν στη βέλτιστη λύση.

Η δυναμική ενημέρωση των παραμέτρων σε πραγματικό χρόνο είναι ένα από τα μεγαλύτερα πλεονεκτήματα της SGD. Σε κάθε βήμα της διαδικασίας, οι παράμετροι του μοντέλου προσαρμόζονται αμέσως μόλις επεξεργαστεί ένα δείγμα δεδομένων. Αυτό καθιστά την SGD ιδιαίτερα αποδοτική σε περιβάλλοντα όπου τα δεδομένα εισάγονται δυναμικά ή όπου απαιτούνται γρήγορες ενημερώσεις, όπως σε εφαρμογές σε πραγματικό χρόνο. Επιπλέον, η ανάγκη για λιγότερους υπολογισμούς ανά βήμα μειώνει το συνολικό κόστος εκτέλεσης, καθιστώντας την SGD μια εξαιρετική επιλογή για προβλήματα με πολύ μεγάλα datasets.

Παρόλο που η χρήση μόνο ενός δείγματος ανά βήμα επιταχύνει σημαντικά τη διαδικασία, δημιουργεί επίσης ορισμένες προκλήσεις. Εξαιτίας της στοχαστικότητας (τυχειρότητας) των ενημερώσεων, η συνάρτηση κόστους μπορεί να μην μειώνεται ομαλά, αλλά να παρουσιάζει διακυμάνσεις κατά τη διάρκεια της βελτιστοποίησης. Αυτές οι διακυμάνσεις μπορεί να οδηγήσουν σε ασταθή συμπεριφορά, ιδιαίτερα όταν το learning rate δεν έχει επιλεγεί κατάλληλα. Ωστόσο, αυτή η στοχαστικότητα μπορεί να λειτουργήσει ως πλεονέκτημα, επιτρέποντας στην SGD να διαφεύγει από τοπικά ελάχιστα και να προσεγγίζει το παγκόσμιο ελάχιστο σε πιο πολύπλοκες συναρτήσεις κόστους.

Η SGD είναι ιδιαίτερα χρήσιμη σε περιπτώσεις όπου το dataset είναι τόσο μεγάλο που δεν μπορεί να αποθηκευτεί εξ ολοκλήρου στη μνήμη RAM. Σε αυτές τις περιπτώσεις, ο αλγόριθμος μπορεί να επεξεργαστεί δεδομένα τμηματικά, επιτρέποντας την εκπαίδευση μοντέλων ακόμα

και σε συστήματα με περιορισμένους υπολογιστικούς πόρους. Επιπλέον, η δυνατότητα της SGD να ενημερώνει τις παραμέτρους δυναμικά σημαίνει ότι είναι κατάλληλη για streaming δεδομένα, όπου οι νέες παρατηρήσεις προστίθενται συνεχώς στο dataset.

Παρόλο που η SGD αποτελεί μια ταχύτερη και πιο ευέλικτη προσέγγιση από την κλασική Gradient Descent, δεν είναι πάντα η καλύτερη επιλογή για όλα τα προβλήματα. Οι διακυμάνσεις στη διαδικασία βελτιστοποίησης μπορεί να οδηγήσουν σε αργή σύγκλιση ή ακόμα και σε απόκλιση, αν το learning rate είναι πολύ υψηλό. Για να αντιμετωπιστούν αυτές οι προκλήσεις, συχνά χρησιμοποιούνται τεχνικές όπως το Momentum, που προσθέτει έναν όρο αδράνειας για την εξομάλυνση των διακυμάνσεων, ή παραλλαγές όπως η Mini-Batch Gradient Descent, που συνδυάζει τα πλεονεκτήματα της SGD και της κλασικής Gradient Descent.

Συνολικά, η SGD είναι μια εξαιρετικά αποδοτική παραλλαγή της Gradient Descent, ιδιαίτερα κατάλληλη για μεγάλα datasets, προβλήματα με δυναμικά δεδομένα και εφαρμογές όπου η ταχύτητα αποτελεί κρίσιμο παράγοντα. Ωστόσο, η επιτυχία της εξαρτάται σε μεγάλο βαθμό από τη σωστή επιλογή παραμέτρων, όπως το learning rate, και από τη χρήση τεχνικών που μπορούν να περιορίσουν τις διακυμάνσεις κατά τη διαδικασία βελτιστοποίησης.

Συνοπτικά:

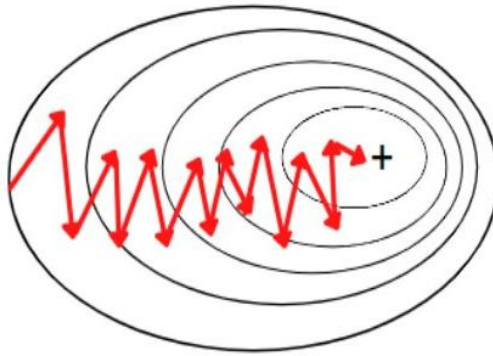
Θετικά:

- Μεγαλύτερη ταχύτητα στα μεγαλύτερα δεδομένα
- Χρόνος εκτέλεσης / Resources μειωμένος

Αρνητικά:

- Σε περίπτωση μεγάλου μεγέθους δεδομένων resources / χρόνος εκτέλεσης επηρεάζεται
- Ύπαρξη τοπικών ελαχίστων επηρεάζει την λειτουργία

Stochastic Gradient Descent



Εικόνα 3: Stochastic GD απεικόνιση

Mini Batch Gradient Descent

Η Mini-Batch Stochastic Gradient Descent (MB-SGD) αποτελεί μία από τις πιο διαδεδομένες παραλλαγές της Gradient Descent, καθώς επιδιώκει να συνδυάσει τα πλεονεκτήματα της κλασικής Gradient Descent (Batch GD) και της Stochastic Gradient Descent (SGD), ξεπερνώντας ταυτόχρονα ορισμένες από τις αδυναμίες τους. Στη συγκεκριμένη προσέγγιση, το dataset χωρίζεται σε μικρότερα υποσύνολα, γνωστά ως mini-batches, και η ενημέρωση των παραμέτρων πραγματοποιείται για κάθε mini-batch ξεχωριστά. Αυτός ο διαχωρισμός προσφέρει μια ισορροπία μεταξύ της ταχύτητας και της σταθερότητας κατά τη διάρκεια της βελτιστοποίησης.

Η βασική ιδέα της MB-SGD έγκειται στη χρήση μικρών πακέτων δεδομένων (mini-batches) αντί για το σύνολο των δεδομένων (όπως στη Batch GD) ή ενός μόνο δείγματος τη φορά (όπως στη SGD). Κάθε mini-batch περιλαμβάνει έναν προκαθορισμένο αριθμό δειγμάτων, τα οποία χρησιμοποιούνται για τον υπολογισμό του μέσου όρου της παραγώγου της συνάρτησης κόστους και την ενημέρωση των παραμέτρων του μοντέλου. Με αυτόν τον τρόπο, η MB-SGD συνδυάζει τη σταθερότητα της Batch GD, διατηρώντας έναν πιο ισορροπημένο υπολογισμό, και την ταχύτητα της SGD, αποφεύγοντας την ανάγκη για επεξεργασία ολόκληρου του dataset σε κάθε βήμα.

Ένα από τα κύρια πλεονεκτήματα της MB-SGD είναι η μείωση του υπολογιστικού κόστους. Χρησιμοποιώντας mini-batches αντί για ολόκληρο το dataset, ο αλγόριθμος απαιτεί λιγότερη μνήμη και επεξεργαστική ισχύ σε κάθε βήμα. Αυτό καθιστά τη μέθοδο ιδιαίτερα αποδοτική για μεγάλα datasets, όπου η πλήρης επεξεργασία όλων των δεδομένων δεν είναι πρακτική. Ταυτόχρονα, ο υπολογισμός του μέσου όρου για κάθε mini-batch εξομαλύνει το θόρυβο που παρατηρείται στη SGD, διατηρώντας τη διαδικασία βελτιστοποίησης πιο σταθερή.

Η χρήση mini-batches προσφέρει επίσης καλύτερη γενίκευση του μοντέλου. Σε αντίθεση με τη SGD, όπου η ενημέρωση των παραμέτρων μπορεί να εξαρτάται υπερβολικά από τις τυχαίες ιδιότητες ενός μόνο δείγματος, η MB-SGD εκμεταλλεύεται την πληροφορία που παρέχεται από ένα μικρό υποσύνολο δεδομένων. Αυτό μειώνει την πιθανότητα να επηρεαστούν οι ενημερώσεις από ακραίες τιμές ή ανωμαλίες στα δεδομένα, διατηρώντας την πορεία της βελτιστοποίησης πιο σταθερή και αξιόπιστη.

Ένα άλλο σημαντικό πλεονέκτημα της MB-SGD είναι η ευελιξία της όσον αφορά το μέγεθος των mini-batches. Ανάλογα με τους διαθέσιμους πόρους και το μέγεθος του dataset, οι χρήστες μπορούν να προσαρμόσουν το μέγεθος των mini-batches ώστε να επιτύχουν την καλύτερη δυνατή ισορροπία μεταξύ ταχύτητας και σταθερότητας. Για παράδειγμα, σε μεγάλα datasets, μικρότερα mini-batches μπορούν να χρησιμοποιηθούν για να μειωθεί η απαίτηση σε μνήμη, ενώ σε μικρότερα datasets, μεγαλύτερα mini-batches μπορούν να αυξήσουν τη σταθερότητα.

Ωστόσο, η MB-SGD δεν είναι χωρίς προκλήσεις. Η επιλογή του κατάλληλου μεγέθους των mini-batches είναι κρίσιμη για την αποτελεσματικότητα του αλγορίθμου. Πολύ μικρά mini-batches μπορεί να εισάγουν θόρυβο στη διαδικασία, ενώ πολύ μεγάλα μπορεί να προσεγγίσουν τη συμπεριφορά της Batch GD, χάνοντας την ταχύτητα που προσφέρει η MB-SGD. Επίσης, η MB-SGD εξακολουθεί να απαιτεί προσεκτική ρύθμιση του learning rate, καθώς η λανθασμένη επιλογή του μπορεί να οδηγήσει είτε σε αργή σύγκλιση είτε σε απόκλιση.

Παρά τις προκλήσεις αυτές, η MB-SGD έχει καθιερωθεί ως η προτιμώμενη μέθοδος για την εκπαίδευση μεγάλων και πολύπλοκων νευρωνικών δικτύων. Συνδυάζοντας τα πλεονεκτήματα της ταχύτητας, της σταθερότητας και της δυνατότητας διαχείρισης μεγάλων datasets, αποτελεί έναν από τους βασικούς αλγορίθμους που χρησιμοποιούνται σε περιβάλλοντα βαθιάς μάθησης

(Deep Learning). Με την κατάλληλη επιλογή παραμέτρων και τη χρήση τεχνικών όπως το Momentum ή η προσαρμοστική εκμάθηση (Adaptive Learning Rates), η MB-SGD μπορεί να επιτύχει υψηλή απόδοση και γρήγορη σύγκλιση, ακόμα και σε απαιτητικές εφαρμογές.

Αποτελεσματικά, υπάρχει ένας καλύτερος συμβιβασμός μεταξύ ταχύτητας και σταθερότητας.
Συνοπτικά:

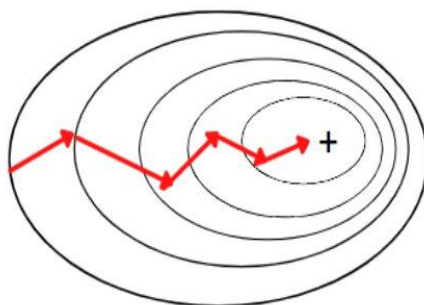
Θετικά:

- Καλύτερη χρήση των resources και πιο γρήγορο από Batch GD
- Σταθερή ενημέρωση σε σχέση με το SGD

Αρνητικά:

- Αποδοτικότητα ίσως μειωμένη

Mini-Batch Gradient Descent



Εικόνα 4: Mini-Batch GD

Τέλος, παρουσιάζονται παραλλαγές της GD με τους αντίστοιχους βελτιστοποιητές και συγκεκριμένα τους 3 βασικούς όπως το Momentum, το RMSprop και εν τέλει τον Adam (Adaptive Moment Estimation).

Βελτιστοποιητές

Η GD έχει μεθόδους – παραλλαγές με βάση την αντίστοιχο παράμετρο βελτιστοποίησης που χρησιμοποιεί. Οι τρεις βασικές μέθοδοι είναι οι παρακάτω:

- Momentum
- RMSprop
- Adam (Adaptive Moment Estimation)

Momentum

Η μέθοδος με χρήση Momentum (ορμής) είναι μια σημαντική βελτίωση της βασικής Gradient Descent, που αξιοποιεί την πληροφορία από τις προηγούμενες κινήσεις του αλγορίθμου για να ενισχύσει τη διαδικασία μείωσης της συνάρτησης κόστους (cost function). Το βασικό χαρακτηριστικό αυτής της μεθόδου είναι ότι λαμβάνει υπόψη τη φορά και το μέγεθος της προηγούμενης αλλαγής στις παραμέτρους, προκειμένου να καθορίσει την κατεύθυνση και την ταχύτητα της τρέχουσας ενημέρωσης. Με τον τρόπο αυτό, το Momentum επιτρέπει στον αλγόριθμο να υπερνικά εμπόδια όπως τα τοπικά ελάχιστα και να επιταχύνει τη σύγκλιση προς το παγκόσμιο ελάχιστο.

Η λειτουργία του Momentum βασίζεται στη φυσική έννοια της ορμής, όπου μια κινούμενη μάζα διατηρεί την κίνησή της προς την ίδια κατεύθυνση εκτός εάν επηρεαστεί από μια εξωτερική δύναμη. Στην πράξη, αυτό σημαίνει ότι το Momentum προσθέτει έναν όρο στη διαδικασία ενημέρωσης των παραμέτρων, ο οποίος αντιπροσωπεύει ένα κλάσμα της προηγούμενης αλλαγής. Αυτή η προσέγγιση εξομαλύνει την πορεία της Gradient Descent, μειώνοντας τις διακυμάνσεις στις ενημερώσεις που μπορεί να προκύψουν από την τυχαιότητα ή την αστάθεια της παραγώγου της συνάρτησης κόστους.

Ένα από τα μεγαλύτερα πλεονεκτήματα του Momentum είναι η ικανότητά του να υπερνικά τα τοπικά ελάχιστα. Σε περίπλοκες συναρτήσεις κόστους, μπορεί να υπάρχουν πολλά τοπικά ελάχιστα που εμποδίζουν τη Gradient Descent να φτάσει στο παγκόσμιο ελάχιστο. Το

Momentum, διατηρώντας τη φόρα από προηγούμενες ενημερώσεις, βοηθά τον αλγόριθμο να "ξεφύγει" από τέτοια τοπικά ελάχιστα, συνεχίζοντας την κίνηση προς την κατεύθυνση που οδηγεί στο παγκόσμιο ελάχιστο.

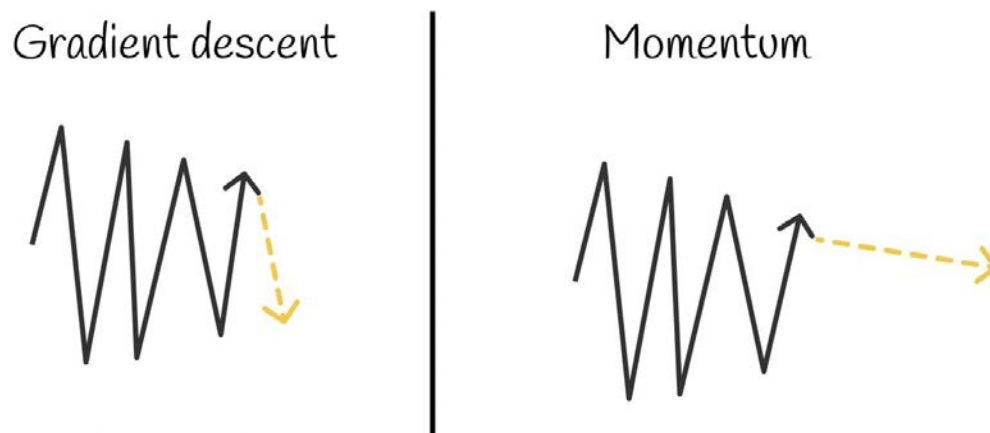
Επιπλέον, το Momentum βελτιώνει σημαντικά την ταχύτητα της σύγκλισης. Αντί να υπολογίζει ξανά από την αρχή την κατεύθυνση της βελτιστοποίησης σε κάθε βήμα, χρησιμοποιεί την προηγούμενη δυναμική για να επιταχύνει την πορεία προς το επιθυμητό σημείο. Αυτό είναι ιδιαίτερα σημαντικό σε περιοχές όπου η συνάρτηση κόστους έχει χαμηλή καμπυλότητα, γνωστές ως "πεδιάδες", όπου η Gradient Descent μπορεί να κινείται πολύ αργά λόγω μικρών παραγώγων.

Η εισαγωγή του Momentum επιτρέπει επίσης στον αλγόριθμο να είναι πιο ανθεκτικός στις διακυμάνσεις που προκαλούνται από θορυβώδη δεδομένα. Σε περιπτώσεις όπου η Stochastic Gradient Descent μπορεί να έχει ασταθή συμπεριφορά λόγω τυχαίων μεταβολών στα δεδομένα, το Momentum εξομαλύνει αυτές τις αλλαγές, διατηρώντας μια πιο σταθερή πορεία βελτιστοποίησης. Αυτό το χαρακτηριστικό καθιστά τη μέθοδο με χρήση Momentum ιδιαίτερα κατάλληλη για προβλήματα όπου τα δεδομένα περιέχουν θόρυβο ή οι συναρτήσεις κόστους έχουν πολλαπλές ασυνέχειες.

Ένα ακόμα σημαντικό πλεονέκτημα του Momentum είναι η ευκολία με την οποία μπορεί να συνδυαστεί με άλλες παραλλαγές της Gradient Descent. Για παράδειγμα, χρησιμοποιείται συχνά σε συνδυασμό με τη Mini-Batch Gradient Descent, καθώς και με πιο προηγμένες μεθόδους, όπως ο Adam. Στον Adam, το Momentum ενσωματώνεται ως μέρος της διαδικασίας προσαρμοστικής εκμάθησης (adaptive learning), βελτιώνοντας ακόμη περισσότερο την απόδοση του αλγορίθμου.

Παρά τα πλεονεκτήματά του, η μέθοδος με χρήση Momentum απαιτεί προσεκτική ρύθμιση της υπερπαραμέτρου που καθορίζει το μέγεθος της ορμής (momentum term). Αν η υπερπαραμέτρος είναι πολύ μεγάλη, ο αλγόριθμος μπορεί να γίνει ασταθής, υπερπηδώντας το επιθυμητό ελάχιστο. Αντίθετα, αν η τιμή της είναι πολύ μικρή, το Momentum δεν θα προσφέρει σημαντική βελτίωση στην ταχύτητα σύγκλισης. Συνήθως, η τιμή της υπερπαραμέτρου ορίζεται κοντά στο 0.9, γεγονός που παρέχει μια καλή ισορροπία μεταξύ ταχύτητας και σταθερότητας.

Συνολικά, το Momentum είναι ένα ισχυρό εργαλείο που ενισχύει σημαντικά την απόδοση της Gradient Descent, ιδιαίτερα σε περιπτώσεις με σύνθετες συναρτήσεις κόστους και μεγάλα datasets. Με τη χρήση του, η διαδικασία εκπαίδευσης γίνεται πιο αποδοτική, σταθερή και γρήγορη, προσφέροντας μια από τις πιο αποτελεσματικές προσεγγίσεις για τη βελτιστοποίηση νευρωνικών δικτύων.



Εικόνα 5: Σύγκριση GD με / χωρίς Momentum

Η ίδια η ορμή υπολογίζεται ως ένας συνδυασμός της προηγούμενης κίνησης και της τρέχουσας με μία παράμετρο που ονομάζεται συντελεστής ορμής. Υπό αυτό το πλαίσιο ο συντελεστής βοηθάει στην περίπτωση που υπάρχει ένα τοπικό ελάχιστο ο αλγόριθμος κινείται με μεγαλύτερη σταθερότητα.

Πλεονεκτήματα:

- Χρήσιμο για αποκόλληση τοπικών ελαχίστων του αλγορίθμου

RMS Prop (Root Mean Square Propagation)

Η RMSProp (Root Mean Square Propagation) είναι μια παραλλαγή της Gradient Descent που δημιουργήθηκε για να αντιμετωπίσει τις προκλήσεις που προκύπτουν από τον υψηλό ρυθμό μάθησης (learning rate) και την αστάθεια στις ενημερώσεις των παραμέτρων, ειδικά σε περιοχές όπου η κλίση της συνάρτησης κόστους (gradient) είναι μεγάλη. Η RMSProp στοχεύει να

εξομαλύνει τη διαδικασία βελτιστοποίησης, εφαρμόζοντας έναν ρυθμό μάθησης που προσαρμόζεται δυναμικά και ανεξάρτητα για κάθε παράμετρο, ανάλογα με τη μεγέθυνση της κλίσης.

Η βασική ιδέα πίσω από την RMSProp είναι η χρήση ενός κινούμενου μέσου όρου των τετραγώνων των κλίσεων κάθε παραμέτρου για τον υπολογισμό ενός σταθμισμένου ρυθμού μάθησης. Σε αντίθεση με τη συμβατική Gradient Descent, όπου όλες οι παράμετροι ενημερώνονται με τον ίδιο ρυθμό, η RMSProp προσαρμόζει τον ρυθμό μάθησης ανάλογα με τη δραστηριότητα κάθε παραμέτρου. Αυτό επιτρέπει στον αλγόριθμο να προσαρμόζεται καλύτερα σε προβλήματα με ανισόρροπη κλίμακα στις παραμέτρους, αποφεύγοντας τις μεγάλες διακυμάνσεις και τις μη αναγκαίες ταλαντώσεις.

Ένα από τα κύρια πλεονεκτήματα της RMSProp είναι η αποτελεσματική αντιμετώπιση των περιοχών όπου η κλίση είναι πολύ μεγάλη ή πολύ μικρή. Σε περιοχές με μεγάλες κλίσεις, ο κινούμενος μέσος όρος των τετραγώνων αυξάνεται, μειώνοντας έτσι τον ρυθμό μάθησης για τις παραμέτρους που επηρεάζονται περισσότερο. Αυτό αποτρέπει την υπερβολική αλλαγή των παραμέτρων και την αποσταθεροποίηση της διαδικασίας βελτιστοποίησης. Αντίθετα, σε περιοχές με μικρές κλίσεις, ο ρυθμός μάθησης παραμένει μεγαλύτερος, επιτρέποντας στις παραμέτρους να ενημερώνονται πιο αποτελεσματικά.

Η RMSProp είναι ιδιαίτερα αποδοτική σε περιπτώσεις όπου η συνάρτηση κόστους έχει έντονη ανομοιογένεια στις κλίσεις των παραμέτρων. Αυτό συναντάται συχνά σε προβλήματα με πολύπλοκες δομές δεδομένων ή σε νευρωνικά δίκτυα με πολλά επίπεδα (deep neural networks). Με την προσαρμογή του ρυθμού μάθησης σε πραγματικό χρόνο, η RMSProp διασφαλίζει ότι οι παράμετροι συγκλίνουν ομαλά προς το επιθυμητό σημείο, ακόμη και όταν οι κλίσεις διαφέρουν σημαντικά μεταξύ τους.

Ένα άλλο σημαντικό πλεονέκτημα της RMSProp είναι η ανθεκτικότητά της σε θορυβώδη δεδομένα. Χρησιμοποιώντας τον κινούμενο μέσο όρο των τετραγώνων, η RMSProp εξομαλύνει τις διακυμάνσεις που προκαλούνται από τυχαία σφάλματα ή ανωμαλίες στα δεδομένα, διατηρώντας τη διαδικασία βελτιστοποίησης σταθερή και αξιόπιστη. Αυτό την καθιστά ιδιαίτερα

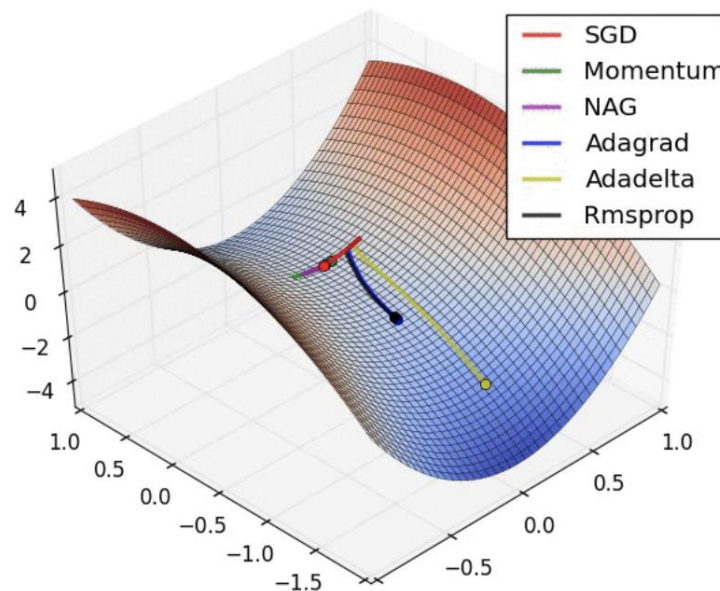
κατάλληλη για εφαρμογές όπου τα δεδομένα περιέχουν θόρυβο ή όταν η εκπαίδευση γίνεται με χρήση δυναμικών δεδομένων.

Η RMSProp έχει αποδειχθεί εξαιρετικά αποτελεσματική σε εφαρμογές βαθιάς μάθησης, όπως η αναγνώριση εικόνων, η επεξεργασία φυσικής γλώσσας και τα συστήματα σύστασης. Συχνά χρησιμοποιείται σε συνδυασμό με άλλες τεχνικές, όπως το Momentum, για περαιτέρω βελτίωση της ταχύτητας και της σταθερότητας της σύγκλισης. Η ενσωμάτωση της RMSProp σε προηγμένες μεθόδους, όπως ο Adam, έχει συμβάλει στη δημιουργία πιο αποδοτικών και ευέλικτων αλγορίθμων βελτιστοποίησης.

Παρά τα πλεονεκτήματά της, η RMSProp απαιτεί προσεκτική ρύθμιση των υπερπαραμέτρων της. Η επιλογή του συντελεστή αποσβέσεως (decay rate) και του learning rate είναι κρίσιμη για την αποτελεσματικότητα του αλγορίθμου. Εάν οι παράμετροι αυτές δεν ρυθμιστούν σωστά, η διαδικασία βελτιστοποίησης μπορεί να γίνει είτε πολύ αργή είτε ασταθής. Παρόλα αυτά, οι προεπιλεγμένες τιμές για τον συντελεστή αποσβέσεως (συνήθως γύρω στο 0.9) έχουν αποδειχθεί κατάλληλες για ένα ευρύ φάσμα εφαρμογών.

Συνολικά, η RMSProp αποτελεί μια από τις πιο αποδοτικές και ευέλικτες παραλλαγές της Gradient Descent, ειδικά για εφαρμογές βαθιάς μάθησης με μεγάλες και πολύπλοκες συναρτήσεις κόστους. Η ικανότητά της να προσαρμόζει δυναμικά τον ρυθμό μάθησης για κάθε

παράμετρο ξεχωριστά την καθιστά ένα ισχυρό εργαλείο για τη βελτίωση της σταθερότητας, της ταχύτητας και της ακρίβειας στη διαδικασία εκπαίδευσης.



Εικόνα 6: RMSProp

Η RMSProp ουσιαστικά υπολογίζει το τετραγωνικό μέσο όρο των παραγώγων για κάθε παράμετρο ξεχωριστά. Η ταχύτητα κίνησης σε παραμέτρους με μεγάλες αλλαγές μειώνεται σημαντικά ως αποτέλεσμα ενώ ταυτόχρονα η κίνηση σε παραμέτρους με μικρότερες αλλαγές επίσης είναι επιτρεπτή.

Πλεονεκτήματα:

- Χρήσιμο κυρίως για τα μεγάλα σύνολα δεδομένων και μεγάλα νευρωνικά δίκτυα λόγω του ρυθμού μάθησης των δεδομένων για παραμέτρους ανεξάρτητα της συνάρτησης κόστους.
- Επιπλέον αποφεύγεται η υπερβολική κίνηση αλγορίθμου σε μη κατάλληλες περιοχές.

Adam (Adaptive Moment Estimation)

Ο Adam (Adaptive Moment Estimation) αποτελεί μία από τις πιο προηγμένες και διαδεδομένες τεχνικές βελτιστοποίησης, συνδυάζοντας τα πλεονεκτήματα του Momentum και του RMSProp.

Ο στόχος του Adam είναι να παρέχει έναν αλγόριθμο που προσαρμόζει δυναμικά τον ρυθμό μάθησης για κάθε παράμετρο, ενώ διατηρεί τη σταθερότητα και την ταχύτητα της διαδικασίας σύγκλισης. Αυτό καθιστά τον Adam ιδιαίτερα αποδοτικό και ευέλικτο για ένα ευρύ φάσμα εφαρμογών βαθιάς μάθησης.

Η λειτουργία του Adam βασίζεται στην ταυτόχρονη χρήση δύο βασικών ιδεών. Πρώτον, παρακολουθεί την κίνηση κάθε παραμέτρου μέσω της ορμής, αξιοποιώντας την τεχνική του Momentum. Η χρήση της ορμής επιτρέπει στον Adam να διατηρεί τη δυναμική των προηγούμενων ενημερώσεων, εξομαλύνοντας τις διακυμάνσεις και επιταχύνοντας τη διαδικασία προς την κατεύθυνση της σύγκλισης. Δεύτερον, εισάγει την προσαρμοστικότητα του RMSProp, προσαρμόζοντας τον ρυθμό μάθησης ανάλογα με την κλίση της συνάρτησης κόστους για κάθε παράμετρο. Αυτή η προσέγγιση διασφαλίζει ότι οι ενημερώσεις γίνονται με ακρίβεια και σταθερότητα, ακόμη και σε πολύπλοκα προβλήματα με ανισοκατανομή στις κλίσεις.

Ένα από τα κύρια χαρακτηριστικά του Adam είναι η χρήση δύο διαφορετικών συντελεστών εξομάλυνσης, οι οποίοι είναι ανεξάρτητοι μεταξύ τους. Ο πρώτος συντελεστής εφαρμόζεται στην ορμή (momentum), εξομαλύνοντας τις διακυμάνσεις από τις προηγούμενες κλίσεις. Ο δεύτερος συντελεστής εξομάλυνσης εφαρμόζεται στον υπολογισμό του κινούμενου μέσου όρου των τετραγώνων των κλίσεων, όπως στο RMSProp. Ο συνδυασμός αυτών των δύο όρων επιτρέπει στον Adam να προσαρμόζεται στις διαφορετικές ανάγκες κάθε παραμέτρου, ενώ παράλληλα διασφαλίζει ότι η εκκίνηση του αλγορίθμου είναι ομαλή.

Ο Adam διακρίνεται για την ταχύτητα και την ακρίβεια της σύγκλισης, καθιστώντας τον μία από τις καλύτερες επιλογές για την εκπαίδευση νευρωνικών δικτύων. Η προσαρμοστικότητα του ρυθμού μάθησης εξαλείφει την ανάγκη για χρονοβόρα δοκιμή και προσαρμογή του learning rate, καθιστώντας τον ιδανικό για προβλήματα με μεγάλη ποικιλία χαρακτηριστικών. Επίσης, η χρήση της ορμής μειώνει την πιθανότητα να παγιδευτεί ο αλγόριθμος σε τοπικά ελάχιστα, διευκολύνοντας τη σύγκλιση σε ένα παγκόσμιο ελάχιστο.

Η σταθερότητα του Adam είναι ένα ακόμη πλεονέκτημα που τον κάνει να ξεχωρίζει. Η εξομάλυνση των διακυμάνσεων στις ενημερώσεις διασφαλίζει ότι η διαδικασία βελτιστοποίησης παραμένει αξιόπιστη ακόμη και σε προβλήματα με θορυβώδη ή αραιά

δεδομένα. Αυτό καθιστά τον Adam κατάλληλο για εφαρμογές όπου οι παραδοσιακές τεχνικές, όπως η Stochastic Gradient Descent, μπορεί να παρουσιάσουν αστάθεια.

Παρά τα πλεονεκτήματά του, ο Adam απαιτεί την κατάλληλη ρύθμιση ορισμένων υπερπαραμέτρων, όπως οι συντελεστές εξομάλυνσης και το learning rate. Παρόλο που οι προεπιλεγμένες τιμές ($\beta_1=0.9$, $\beta_2=0.999$ και $\epsilon=10^{-8}$) έχουν αποδειχθεί αποτελεσματικές σε πολλές περιπτώσεις, η βελτιστοποίηση αυτών των παραμέτρων μπορεί να βελτιώσει περαιτέρω την απόδοση του αλγορίθμου σε εξειδικευμένα προβλήματα.

Ο Adam χρησιμοποιείται ευρέως σε ένα ευρύ φάσμα εφαρμογών, όπως η αναγνώριση εικόνων, η ανάλυση φυσικής γλώσσας, τα συστήματα σύστασης και τα παιχνίδια ενίσχυσης (reinforcement learning). Η ευελιξία και η αποτελεσματικότητά του έχουν συμβάλει στη δημοτικότητά του, καθιστώντας τον μία από τις πρώτες επιλογές για την εκπαίδευση βαθιών νευρωνικών δικτύων.

Συνολικά, ο Adam είναι ένας εξαιρετικά αποδοτικός και αξιόπιστος αλγόριθμος βελτιστοποίησης, που συνδυάζει τα καλύτερα χαρακτηριστικά του Momentum και του RMSProp. Με τη χρήση του, οι ερευνητές και οι επαγγελματίες μπορούν να επιτύχουν γρήγορη και σταθερή εκπαίδευση μοντέλων, ακόμη και σε σύνθετα προβλήματα με μεγάλα και ανισόρροπα datasets.

Πλεονεκτήματα:

- Αυτόματη προσαρμογή του ρυθμού μάθησης με σκοπό να αντιμετωπίσουμε τα μεγάλα και πολύπλοκα δίκτυα εξασφαλίζοντας γρήγορη και αξιόπιστη σύγκλιση. Ευελιξία και σταθερότητα καθιστούν έναν από τους πιο δημοφιλείς παραμέτρους βελτιστοποίησης.

Κεφάλαιο 2: Ανασκόπηση

Η σύγκριση και αξιολόγηση αλγορίθμων βελτιστοποίησης σε νευρωνικά δίκτυα αποτελεί σημαντικό και συνεχώς αναπτυσσόμενο πεδίο έρευνας στη μηχανική μάθηση και την τεχνητή νοημοσύνη. Η κατανόηση των παραμέτρων και των χαρακτηριστικών που επηρεάζουν την απόδοση των αλγορίθμων αυτών είναι κρίσιμη για τη βελτίωση της εκπαίδευσης των νευρωνικών δικτύων και την εφαρμογή τους σε πραγματικά προβλήματα. Οι αλγόριθμοι βελτιστοποίησης, όπως οι Stochastic Gradient Descent (SGD), Momentum, Nesterov Accelerated Gradient (NAG), Adagrad, RMSprop, Adam, AdaDelta και Adamax, έχουν αποτελέσει το αντικείμενο πολυάριθμων μελετών και συγκριτικών αναλύσεων.

Προηγούμενες έρευνες έχουν επικεντρωθεί στην ανάλυση της συμπεριφοράς αυτών των αλγορίθμων σε διάφορα σύνολα δεδομένων, όπως το MNIST, το CIFAR-10 και άλλα, και σε διαφορετικές αρχιτεκτονικές, όπως πλήρως συνδεδεμένα δίκτυα (fully connected networks), συνελκτικά νευρωνικά δίκτυα (Convolutional Neural Networks) και αναδρομικά νευρωνικά δίκτυα (Recurrent Neural Networks). Κάθε αλγόριθμος έχει συγκεκριμένα πλεονεκτήματα και μειονεκτήματα, τα οποία εξαρτώνται από τις ιδιαιτερότητες των δεδομένων, την πολυπλοκότητα του μοντέλου και τις απαιτήσεις του εκάστοτε προβλήματος.

Ο SGD, για παράδειγμα, είναι ένας απλός και αποδοτικός αλγόριθμος, ωστόσο παρουσιάζει αργή σύγκλιση σε πολύπλοκες συναρτήσεις κόστους. Αντίθετα, ο Adam, που συνδυάζει στοιχεία από τους RMSprop και Momentum, προσφέρει ταχύτερη και σταθερότερη σύγκλιση, καθιστώντας τον ιδιαίτερα κατάλληλο για μεγάλα και σύνθετα datasets. Παρόμοια, ο NAG, μέσω της τεχνικής "lookahead", μπορεί να υπερβεί προβλήματα που σχετίζονται με τα τοπικά ελάχιστα, αλλά απαιτεί περισσότερους υπολογιστικούς πόρους.

Επιπλέον, πολλές μελέτες έχουν εστιάσει στη σύγκριση των αλγορίθμων σε προβλήματα πραγματικού χρόνου, όπως η ανάλυση εικόνας, η επεξεργασία φυσικής γλώσσας και η πρόβλεψη σειρών δεδομένων. Τα αποτελέσματα αυτών των ερευνών υποδεικνύουν ότι η

επιλογή του κατάλληλου αλγορίθμου εξαρτάται όχι μόνο από τη φύση του προβλήματος αλλά και από τις διαθέσιμες υπολογιστικές δυνατότητες, το μέγεθος του dataset και τις απαιτήσεις χρόνου.

Επιπρόσθετα, η έρευνα έχει στραφεί στη βελτίωση της αποδοτικότητας των αλγορίθμων μέσω υβριδικών προσεγγίσεων ή τεχνικών προσαρμοστικότητας. Για παράδειγμα, ορισμένοι ερευνητές έχουν ενσωματώσει δυναμικούς ρυθμούς μάθησης ή τεχνικές κανονικοποίησης, όπως το Batch Normalization, για να μειώσουν την ευαισθησία στις αρχικές ρυθμίσεις των υπερπαραμέτρων. Άλλες μελέτες έχουν διερευνήσει τη χρήση αλγορίθμων μετα-βελτιστοποίησης, οι οποίοι προσαρμόζουν αυτόματα τις υπερπαραμέτρους κατά τη διάρκεια της εκπαίδευσης, ενισχύοντας την αποδοτικότητα και την ακρίβεια.

Μια σημαντική μελέτη είναι η διπλωματική εργασία του Κωνσταντίνου Λιαντζάκη με τίτλο "Συγκριτική μελέτη αλγορίθμων βελτιστοποίησης σε Python για την εύρεση αρχιτεκτονικών νευρωνικών δικτύων". Σε αυτήν, ο συγγραφέας εξετάζει τη χρήση διαφόρων αλγορίθμων βελτιστοποίησης, όπως οι Ant Colony Optimization, Particle Swarm Optimization και Artificial Bee Colony, για την αυτόματη αναζήτηση αρχιτεκτονικών νευρωνικών δικτύων, αξιοποιώντας το σύνολο σημείων αναφοράς NASBench-101. Η μελέτη καταλήγει σε συμπεράσματα σχετικά με την αποδοτικότητα και τις προκλήσεις που παρουσιάζουν αυτοί οι αλγόριθμοι στην εύρεση βέλτιστων αρχιτεκτονικών (Λιαντζάκης, 2021).

Επιπλέον, η διπλωματική εργασία του Ισίδωρου Αβδελιώδη με τίτλο "Συγκριτική ανάλυση αλγορίθμων μηχανικής μάθησης και βαθιάς μάθησης σε προβλήματα εκφράσεων γονιδίων", εξετάζει την απόδοση διαφόρων αλγορίθμων σε προβλήματα βιοπληροφορικής. Η μελέτη αυτή προσφέρει πολύτιμες πληροφορίες για την αποτελεσματικότητα των αλγορίθμων σε σύνθετα βιολογικά δεδομένα (Αβδελιώδης, 2024).

Τέλος, η διπλωματική εργασία του Λεωνίδα Μαζαράκη με τίτλο "Ανάλυση και σύγκριση αλγορίθμων συσσωμάτωσης για βελτιστοποίηση τεχνικών Ομοσπονδιακής Μάθησης σε cross-silo σενάρια", επικεντρώνεται στην ανάλυση και σύγκριση αλγορίθμων συσσωμάτωσης σε εφαρμογές Ομοσπονδιακής Μάθησης. Η μελέτη αυτή συμβάλλει στην κατανόηση της αποδοτικότητας των αλγορίθμων σε κατανεμημένα συστήματα μάθησης (Μαζαράκης, 2023).

Επιπλέον, υπάρχουν ιδιαίτερα πολλές μελέτες που έχουν διεξαγάγει συγκριτικές αναλύσεις αλγορίθμων βελτιστοποίησης στη βαθιά μάθηση, παρέχοντας πολύτιμες πληροφορίες για την απόδοσή τους σε διάφορες εργασίες και σύνολα δεδομένων και στο εξωτερικό.

Μία αξιοσημείωτη μελέτη είναι η "A Comparison of Optimization Algorithms for Deep Learning" του Derya Soydaner, η οποία εξετάζει ευρέως χρησιμοποιούμενους αλγορίθμους βελτιστοποίησης, ιδιαίτερα τις μεθόδους προσαρμοστικών βαθμίδων, σε εποπτευόμενες και μη εποπτευόμενες εργασίες. Η έρευνα αξιολογεί αυτούς τους αλγόριθμους σε σύνολα δεδομένων εικόνων, όπως MNIST, CIFAR-10, Kaggle Flowers και Labeled Faces in the Wild, υπογραμμίζοντας τις διαφορές τους σε σύγκριση με βασικούς αλγορίθμους βελτιστοποίησης (Soydaner, 2020).

Μία άλλη σημαντική εργασία είναι η "Survey of Optimization Algorithms in Modern Neural Networks," που δημοσιεύτηκε στο περιοδικό Mathematics. Η επισκόπηση αυτή εξετάζει διάφορες μεθόδους βελτιστοποίησης που χρησιμοποιούνται για την εκπαίδευση νευρωνικών δικτύων, αναλύοντας τα πλεονεκτήματα και τα μειονεκτήματά τους. Η μελέτη τονίζει τη σημασία της επιλογής κατάλληλων τεχνικών βελτιστοποίησης για τη βελτίωση της αποδοτικότητας και της ακρίβειας στην εκπαίδευση των νευρωνικών δικτύων (Abdulkadiron, 2023).

Επιπλέον, η εργασία "A Comparative Study of Optimization Methods for Improving Artificial Neural Network Training" παρουσιάζει μια συγκριτική ανάλυση κοινώς χρησιμοποιούμενων μεθόδων παγκόσμιας βελτιστοποίησης, όπως το Simulated Annealing, το Direct Search και οι Γενετικοί Αλγόριθμοι. Αυτές οι μέθοδοι αξιολογούνται ως προς την αποτελεσματικότητά τους στη βελτιστοποίηση των βαρών και των προσανατολισμών των νευρωνικών δικτύων, πριν την εφαρμογή των αλγορίθμων back-propagation, με στόχο την αποτροπή σύγκλισης σε τοπικά ελάχιστα (Kajornrit, 2015).

Αυτές οι μελέτες συμβάλλουν σε μια βαθύτερη κατανόηση του τρόπου με τον οποίο οι διάφοροι αλγόριθμοι βελτιστοποίησης επηρεάζουν την εκπαίδευση και την απόδοση των νευρωνικών δικτύων, καθοδηγώντας ερευνητές και επαγγελματίες στην επιλογή κατάλληλων μεθόδων για τις συγκεκριμένες εφαρμογές τους.

Κεφάλαιο 3: Υλοποίηση

Για την εργασία χρησιμοποιήθηκε η PyTorch, μια δημοφιλής βιβλιοθήκη βαθιάς μάθησης που παρέχει ήδη έτοιμες, αξιόπιστες και δοκιμασμένες υλοποιήσεις για πολλούς αλγορίθμους βελτιστοποίησης. Αυτή η επιλογή κρίθηκε ιδιαίτερα κατάλληλη, καθώς εξαλείφει την ανάγκη δημιουργίας ή ελέγχου κώδικα από την αρχή. Δεδομένου ότι ο στόχος της διπλωματικής εργασίας είναι η σύγκριση των αλγορίθμων και όχι η ανάπτυξη νέων υλοποιήσεων, η χρήση του PyTorch αποτελεί μια πρακτική και αποδοτική λύση.

Στο πλαίσιο της εργασίας, αναπτύχθηκε κώδικας που εφαρμόζεται για την αξιολόγηση διαφόρων αλγορίθμων βελτιστοποίησης σε δύο διαφορετικά μοντέλα βαθιάς μάθησης. Τα μοντέλα αυτά σχεδιάστηκαν για την ταξινόμηση εικόνων από το MNIST dataset, ένα ευρέως χρησιμοποιούμενο σύνολο δεδομένων που περιέχει χειρόγραφους αριθμούς σε εικόνες διαστάσεων 28x28 pixels.

Ορισμός Μοντέλων

Δημιουργήθηκαν δύο ξεχωριστά μοντέλα νευρωνικών δικτύων, το SimpleNN και το DeepNN, που διαφοροποιούνται ως προς την πολυπλοκότητα και τη δομή τους:

SimpleNN:

Ένα απλό νευρωνικό δίκτυο με τρία επίπεδα (layers):

- Layer 1: Περιέχει 128 νευρώνες και δέχεται ως είσοδο ένα διάνυσμα 784 διαστάσεων (28x28 pixels κάθε εικόνας).
- Layer 2: Περιέχει 64 νευρώνες.

- Layer 3: Το τελικό επίπεδο έχει 10 νευρώνες, καθένας από τους οποίους αντιστοιχεί σε μία από τις δέκα κλάσεις του MNIST (0-9).
- Η ενεργοποίηση πραγματοποιείται με τη συνάρτηση ReLU σε κάθε επίπεδο, εκτός του τελευταίου, όπου χρησιμοποιείται Softmax για την παραγωγή των πιθανοτήτων των κλάσεων.

DeepNN:

Ένα πιο σύνθετο δίκτυο με πέντε επίπεδα:

- Layer 1: 256 νευρώνες.
- Layer 2: 128 νευρώνες.
- Layer 3: 64 νευρώνες.
- Layer 4: 32 νευρώνες.
- Layer 5: Το τελικό επίπεδο έχει 10 νευρώνες.
- Παρόμοια λογική ενεργοποίησης με το SimpleNN, αλλά με περισσότερα επίπεδα και νευρώνες, επιτρέποντας την ανάλυση πιο περίπλοκων σχέσεων μεταξύ των pixels, αν και με αυξημένο υπολογιστικό κόστος.

Το SimpleNN είναι σχεδιασμένο για γρηγορότερη εκπαίδευση με χαμηλότερες απαιτήσεις σε υπολογιστική ισχύ, ενώ το DeepNN προσφέρει υψηλότερη ακρίβεια σε πιο σύνθετα δεδομένα, αν και απαιτεί περισσότερο χρόνο και πόρους.

Αλγόριθμοι Βελτιστοποίησης

Και τα δύο μοντέλα εκπαιδεύτηκαν χρησιμοποιώντας τους παρακάτω αλγόριθμους:

- SGD (Stochastic Gradient Descent): Βασική στοχαστική προσέγγιση.
- Momentum: Ενσωματώνει την ορμή για σταθερότερη σύγκλιση.

- NAG (Nesterov Accelerated Gradient): Βελτιστοποιεί την πρόβλεψη της μελλοντικής κίνησης.
- Adagrad: Προσαρμοστικός αλγόριθμος για σπάνια δεδομένα.
- RMSprop: Χρησιμοποιεί μέσο όρο τετραγώνων κλίσεων.
- Adam: Συνδυάζει Momentum και RMSprop.
- AdaDelta: Παρόμοιος με τον Adagrad, με σταθεροποίηση του learning rate.
- Adamax: Μια παραλλαγή του Adam για απλοποίηση.

Παράμετροι Εκπαίδευσης

Κατά την εκπαίδευση των μοντέλων, ορίστηκαν οι εξής παράμετροι:

- Epochs: Αριθμός εποχών εκπαίδευσης.
- Batch Size: Μέγεθος του batch για κάθε βήμα εκπαίδευσης.
- Learning Rate: Ο αρχικός ρυθμός μάθησης για κάθε αλγόριθμο.
- Loss Function: Η CrossEntropyLoss χρησιμοποιήθηκε για τον υπολογισμό της απώλειας κατά την ταξινόμηση.

Δομή Αποτελεσμάτων

Η αξιολόγηση πραγματοποιήθηκε με βάση τα εξής κριτήρια:

- Model: Το όνομα του μοντέλου (π.χ., SimpleNN ή DeepNN).
- Optimizer: Ο αλγόριθμος βελτιστοποίησης.
- Epochs: Ο αριθμός των εποχών εκπαίδευσης.
- Batch Size: Το μέγεθος του batch που χρησιμοποιήθηκε.

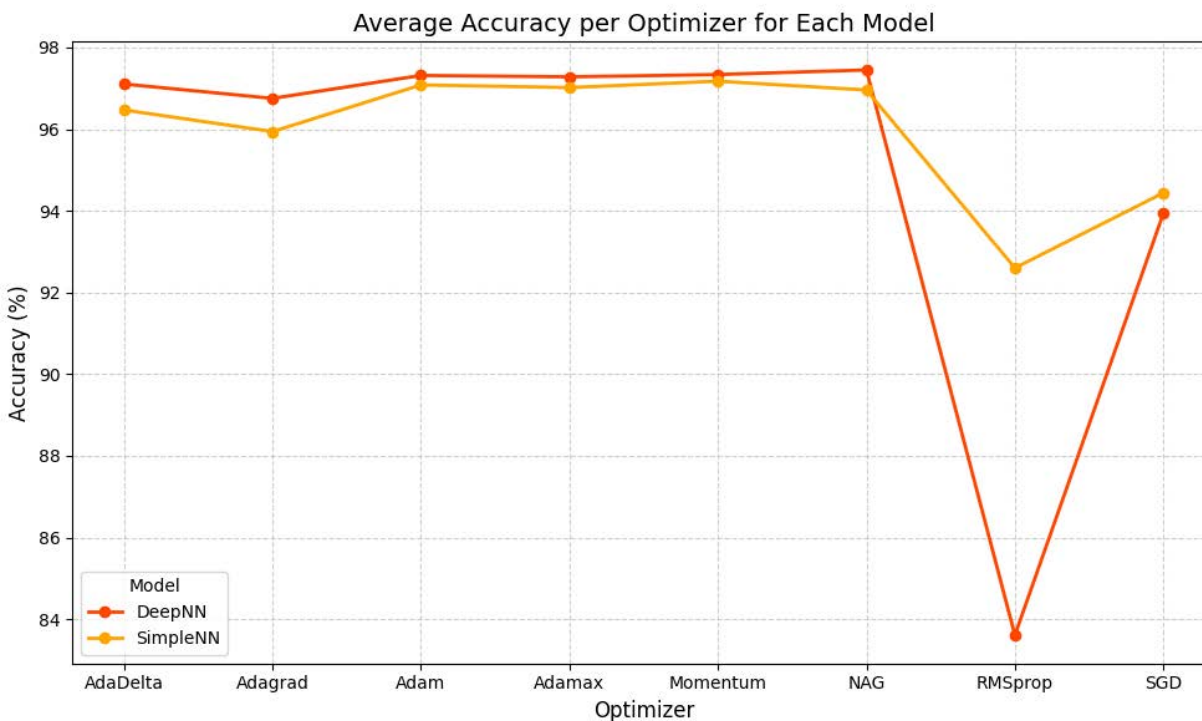
- Time: Ο χρόνος εκπαίδευσης (σε δευτερόλεπτα).
- Accuracy: Η ακρίβεια του μοντέλου στο test set.

Σύγκριση και Συμπεράσματα

Η εκπαίδευση έδειξε ότι το SimpleNN εκπαιδεύεται ταχύτερα λόγω της χαμηλότερης πολυπλοκότητας, ενώ το DeepNN απέδωσε υψηλότερη ακρίβεια σε σύνθετα προβλήματα. Από τους αλγόριθμους βελτιστοποίησης, ο Adam και ο RMSprop έδειξαν τη βέλτιστη απόδοση, παρέχοντας σταθερότητα και υψηλή ακρίβεια. Αντίθετα, ο SGD είχε αργή σύγκλιση, ιδιαίτερα στο DeepNN.

Η προσέγγιση αυτή, χρησιμοποιώντας το PyTorch, επέτρεψε μια συστηματική και αξιόπιστη σύγκριση των αλγορίθμων χωρίς την ανάγκη ανάπτυξης ή επαλήθευσης νέου κώδικα. Αυτή η στρατηγική διασφάλισε ακρίβεια, συνέπεια και ευκολία στη διαδικασία της διπλωματικής εργασίας.

Γράφημα 1



Εικόνα 7: Γράφημα 1

Το γράφημα απεικονίζει τη μέση ακρίβεια (%) κάθε αλγορίθμου βελτιστοποίησης (Optimizer) για δύο διαφορετικά μοντέλα βαθιάς μάθησης: το SimpleNN και το DeepNN. Οριζόντια στον άξονα X παρουσιάζονται οι διαφορετικοί αλγόριθμοι βελτιστοποίησης (AdaDelta, Adagrad, Adam, Adamax, Momentum, NAG, RMSprop και SGD), ενώ στον κάθετο άξονα Y φαίνεται η μέση ακρίβεια των μοντέλων στο dataset που χρησιμοποιήθηκε.

Γενική Ανάλυση

Το γράφημα μας δίνει πληροφορίες για την απόδοση κάθε αλγορίθμου, επιτρέποντας τη σύγκριση μεταξύ των δύο μοντέλων, SimpleNN και DeepNN, καθώς και τη γενική συμπεριφορά κάθε αλγορίθμου. Οι καμπύλες του SimpleNN και του DeepNN ακολουθούν παρόμοια τάση, αλλά παρατηρούνται ορισμένες διαφοροποιήσεις στην ακρίβεια.

Ανάλυση ανά αλγόριθμο βελτιστοποίησης

AdaDelta:

Η απόδοση του AdaDelta κυμάνθηκε επίσης πάνω από το 96%, επιδεικνύοντας σταθερή και αποτελεσματική συμπεριφορά. Η δυναμική προσαρμογή του learning rate τον καθιστά κατάλληλο τόσο για το SimpleNN όσο και για το DeepNN.

Adagrad:

Ο Adagrad εμφάνισε ακρίβεια λίγο κάτω από το 96%, με οριακά καλύτερη επίδοση στο DeepNN. Η ικανότητά του να ευνοεί σπάνια χαρακτηριστικά στα δεδομένα φαίνεται να αποδίδει καλά και στα δύο μοντέλα, αν και με μικρότερη υπεροχή έναντι των προηγούμενων αλγορίθμων.

Adam:

Ο Adam κατέγραψε επίσης πολύ υψηλή ακρίβεια, λίγο κάτω από το 97%, παραμένοντας ένας από τους πιο αξιόπιστους αλγορίθμους. Η ισορροπία μεταξύ προσαρμοστικότητας και ταχύτητας σύγκλισης τον καθιστά κατάλληλο και για τα δύο μοντέλα, με ελαφρώς καλύτερη επίδοση στο DeepNN.

Adamax:

Ο Adamax διατήρησε υψηλά επίπεδα ακρίβειας (άνω του 96%), με ελαφρώς καλύτερη συμπεριφορά στο DeepNN. Ως παραλλαγή του Adam, προσφέρει σταθερότητα και αξιοπιστία, κάνοντάς τον χρήσιμο όταν απαιτείται συνέπεια χωρίς υπερβολικές προσαρμογές.

Momentum:

Ο Momentum πέτυχε την υψηλότερη μέση ακρίβεια και για τα δύο μοντέλα, αγγίζοντας ή ξεπερνώντας το 97%. Η απόδοσή του επιβεβαιώνει την αποτελεσματικότητά του στην επιτάχυνση της σύγκλισης και την αποφυγή τοπικών ελαχίστων. Η διαφορά στην ακρίβεια μεταξύ των μοντέλων ήταν αμελητέα, γεγονός που υποδεικνύει τη σταθερότητα του αλγορίθμου σε διαφορετικά επίπεδα πολυπλοκότητας.

NAG (Nesterov Accelerated Gradient):

Ο NAG παρουσίασε εξαιρετική ακρίβεια, πολύ κοντά στα κορυφαία επίπεδα των Momentum και Adam. Η διαφορά απόδοσης μεταξύ SimpleNN και DeepNN ήταν μικρή, επιβεβαιώνοντας την ικανότητά του να προσαρμόζεται σε διαφορετικά σενάρια. Η τεχνική lookahead αποδίδει θετικά και στα δύο μοντέλα.

RMSprop:

Ο RMSprop σημείωσε τη χαμηλότερη ακρίβεια, με μέσες τιμές κάτω από το 89%. Εμφάνισε ασταθή συμπεριφορά, ιδιαίτερα στο DeepNN, όπου η πολυπλοκότητα φαίνεται να επηρεάζει αρνητικά την απόδοση του αλγορίθμου.

SGD (Stochastic Gradient Descent):

Ο SGD ήταν ο μόνος αλγόριθμος με σαφώς χαμηλότερη απόδοση, καταγράφοντας μέση ακρίβεια κάτω από 94%. Η απουσία τεχνικών όπως ορμή ή προσαρμοστικότητα καθιστά τον αλγόριθμο λιγότερο αποτελεσματικό, ιδιαίτερα στο DeepNN, όπου η απόδοσή του ήταν αισθητά μειωμένη.

Συγκρίσεις μεταξύ των δύο μοντέλων

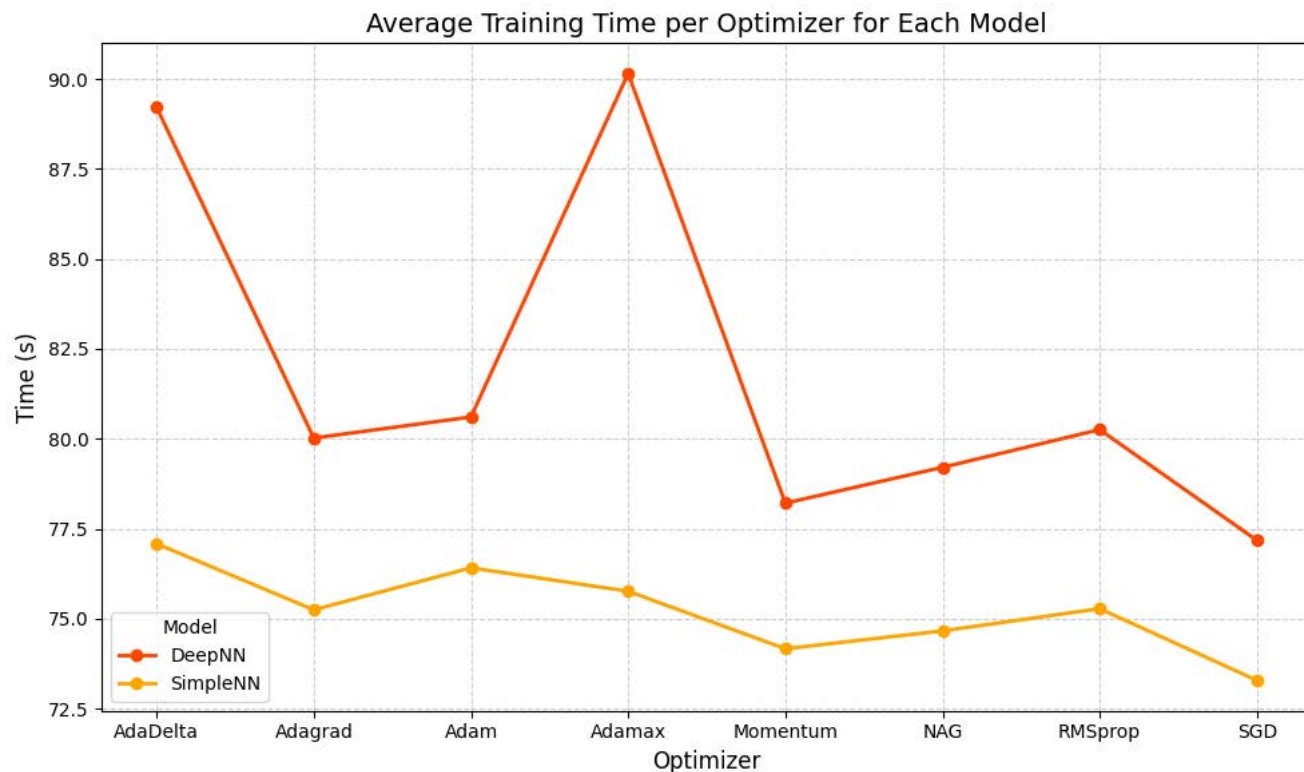
- SimpleNN: Παρουσιάζει σταθερά καλή ακρίβεια με τους περισσότερους αλγορίθμους, ειδικά με Momentum, Adam και AdaDelta. Το απλούστερο μοντέλο φαίνεται να επωφελείται από αλγορίθμους που συνδυάζουν ταχύτητα και βασική προσαρμοστικότητα.
- DeepNN: Υπερτερεί ως προς την τελική ακρίβεια σε σχεδόν όλους τους αλγορίθμους, ειδικά στους πιο προηγμένους όπως οι Adam, Momentum και NAG. Εμφανίζει μεγαλύτερη ευαισθησία σε ακατάλληλους αλγορίθμους (π.χ. RMSprop, SGD), λόγω της αυξημένης πολυπλοκότητας.

Συμπεράσματα

1. Αποτελεσματικότητα των αλγορίθμων: Ο Momentum αναδείχθηκε ο πιο αποτελεσματικός αλγόριθμος, ενώ υψηλές επιδόσεις εμφάνισαν επίσης οι Adam, NAG και Adamax.
2. Επιδόσεις σε απλά vs σύνθετα μοντέλα: Το SimpleNN αποδίδει πολύ καλά με σταθερούς αλγόριθμους, ενώ το DeepNN επωφελείται περισσότερο από εξελιγμένες τεχνικές.
3. Ακατάλληλοι αλγόριθμοι: Ο RMSprop και ο SGD υστερούν σημαντικά στην ακρίβεια, ιδιαίτερα στο DeepNN, καθιστώντας τους λιγότερο κατάλληλους για εφαρμογές με αυξημένες απαιτήσεις.

Συνοψίζοντας, το γράφημα επιβεβαιώνει τη σημασία της κατάλληλης επιλογής αλγορίθμου ανάλογα με την πολυπλοκότητα του μοντέλου και τις ανάγκες της εφαρμογής. Οι προσαρμοστικοί και σταθεροί αλγόριθμοι, όπως ο Momentum και ο Adam, διασφαλίζουν υψηλή απόδοση και αξιοπιστία.

Γράφημα 2



Εικόνα 8: Γράφημα 2

Το γράφημα παρουσιάζει τον μέσο χρόνο εκπαίδευσης (σε δευτερόλεπτα) για κάθε αλγόριθμο βελτιστοποίησης (Optimizer) που εφαρμόστηκε σε δύο διαφορετικά μοντέλα βαθιάς μάθησης, το SimpleNN και το DeepNN. Ο οριζόντιος άξονας δείχνει τους αλγορίθμους βελτιστοποίησης (AdaDelta, Adagrad, Adam, Adamax, Momentum, NAG, RMSprop και SGD), ενώ ο κατακόρυφος άξονας δείχνει τον χρόνο εκπαίδευσης σε δευτερόλεπτα (s).

Γενική Ανάλυση

Το γράφημα αποκαλύπτει σημαντικές διαφορές στον χρόνο εκπαίδευσης μεταξύ των διαφορετικών αλγορίθμων και των δύο μοντέλων. Παρόλο που για τους περισσότερους αλγορίθμους ο χρόνος παραμένει σχετικά σταθερός, υπάρχει μια αξιοσημείωτη απόκλιση στον χρόνο εκπαίδευσης των αλγορίθμων AdaDelta και Adamax, ειδικά για το DeepNN.

Ανάλυση ανά αλγόριθμο βελτιστοποίησης

AdaDelta:

Ο χρόνος εκπαίδευσης είναι σταθερός και για τα δύο μοντέλα, περίπου 77 δευτερόλεπτα για το SimpleNN και 87 δευτερόλεπτα για το DeepNN. Η διαφορά αυτή υποδηλώνει ότι ο AdaDelta λειτουργεί με σταθερή αποδοτικότητα, αλλά παρουσιάζει μικρή επιβάρυνση στο πιο σύνθετο μοντέλο.

Adagrad:

Παρουσιάζει σταθερά χαμηλό χρόνο εκπαίδευσης, 75 δευτερόλεπτα για το SimpleNN και 80 δευτερόλεπτα για το DeepNN. Η αποδοτικότητά του είναι αξιοσημείωτη και ενισχύεται από την προσαρμοστική φύση του αλγορίθμου.

Adam:

Ο χρόνος εκπαίδευσης διατηρείται χαμηλά, περίπου 75 δευτερόλεπτα στο SimpleNN και 81 στο DeepNN. Αυτό υπογραμμίζει την ικανότητα του Adam να διατηρεί αποδοτικότητα παρά την υπολογιστική του πολυπλοκότητα.

Adamax:

Καταγράφει παρόμοιο χρόνο με τον Adam, με 76 δευτερόλεπτα για το SimpleNN και 83 για το DeepNN. Η συνέπεια αυτή δείχνει ότι ο Adamax μπορεί να χρησιμοποιηθεί εναλλακτικά χωρίς σημαντικές χρονικές διαφοροποιήσεις.

Momentum:

Εμφανίζει εξαιρετικά χαμηλό χρόνο, περίπου 74 δευτερόλεπτα για το SimpleNN και 78 για το DeepNN. Η προσθήκη της ορμής δεν επιβαρύνει τον χρόνο εκπαίδευσης, καθιστώντας τον αλγόριθμο εξαιρετικά αποδοτικό.

NAG (Nesterov Accelerated Gradient):

Ο NAG εμφανίζει χαμηλό χρόνο και στα δύο μοντέλα, με 75 δευτερόλεπτα στο SimpleNN και 79 στο DeepNN. Η απόδοση του διαψεύδει την αρχική υπόθεση περί σημαντικής επιβάρυνσης στο DeepNN.

RMSprop:

Ο RMSprop καταγράφει 75 δευτερόλεπτα στο SimpleNN και 80 στο DeepNN. Παρουσιάζει χρονική σταθερότητα, καθιστώντας τον κατάλληλο για γενικές χρήσεις παρά τις αδυναμίες του στην ακρίβεια.

SGD (Stochastic Gradient Descent):

Ο SGD έχει τον χαμηλότερο χρόνο εκπαίδευσης, με 73 δευτερόλεπτα για το SimpleNN και 77 για το DeepNN. Αν και υστερεί στην ακρίβεια, προσφέρει εξαιρετική ταχύτητα.

Συγκρίσεις μεταξύ των δύο μοντέλων

- SimpleNN: Το SimpleNN παρουσιάζει σταθερά τον μικρότερο χρόνο εκπαίδευσης μεταξύ των δύο μοντέλων. Οι χρόνοι κυμαίνονται μεταξύ 73 και 77 δευτερολέπτων, κάτι που επιβεβαιώνει την αποδοτικότητα της αρχιτεκτονικής του.
- DeepNN: Το DeepNN καταγράφει ελαφρώς αυξημένους χρόνους, μεταξύ 77 και 90 δευτερολέπτων. Η διαφορά αυτή θεωρείται αναμενόμενη λόγω της πολυπλοκότητας του μοντέλου, ωστόσο παραμένει σε αποδεκτά επίπεδα.

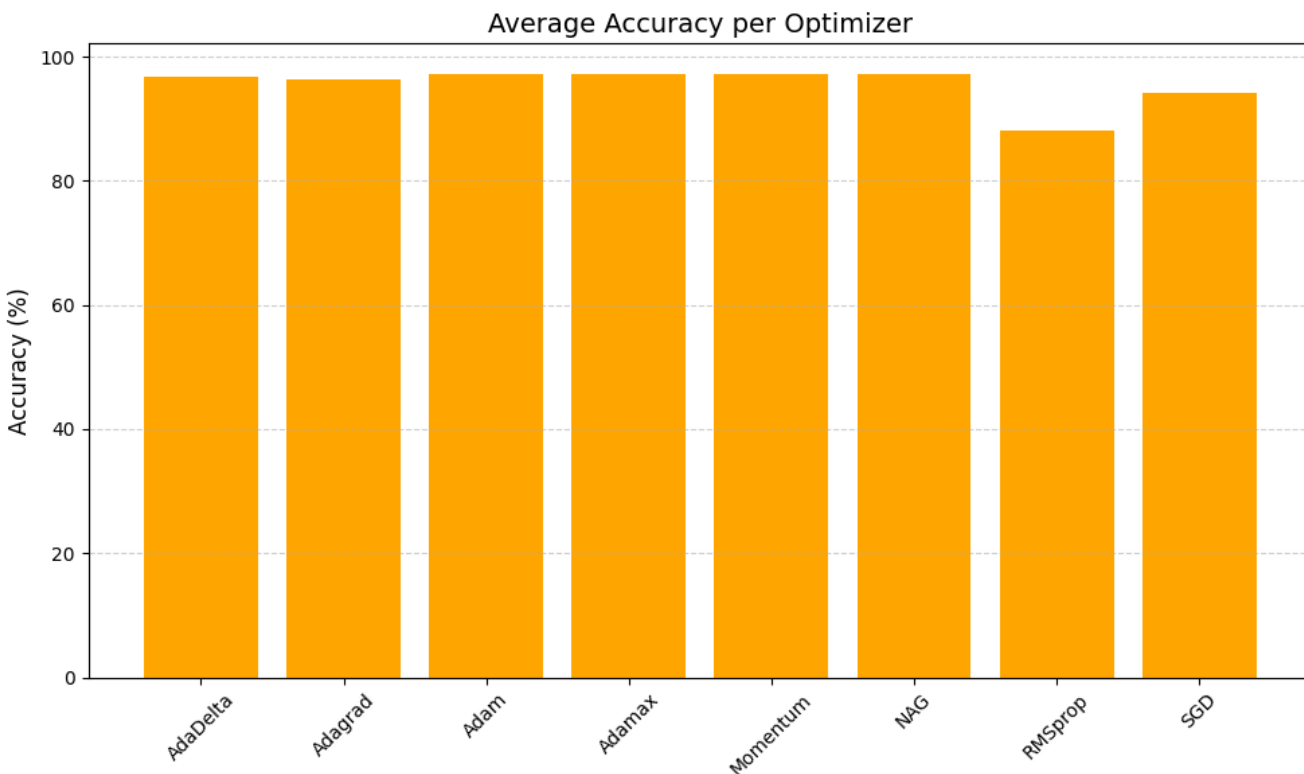
Συμπεράσματα

1. Αποδοτικότητα στους χρόνους εκπαίδευσης: Οι περισσότεροι αλγόριθμοι έχουν παρόμοιο χρόνο εκπαίδευσης, γεγονός που δείχνει ότι είναι εξίσου αποδοτικοί για την πλειονότητα των εφαρμογών.

2. Πλεονεκτήματα και αδυναμίες αλγορίθμων: Ο SGD και ο Momentum καταγράφουν τους χαμηλότερους χρόνους εκπαίδευσης και στα δύο μοντέλα, γεγονός που ενισχύει τη θέση τους σε εφαρμογές με αυστηρούς χρονικούς περιορισμούς. Αντίθετα, αλγόριθμοι όπως ο AdaDelta και ο Adamax εμφανίζουν αυξημένο χρόνο στο DeepNN, ξεπερνώντας σε ορισμένες περιπτώσεις τα 87 δευτερόλεπτα. Παρ' όλα αυτά, διατηρούν αποδεκτή χρονική συμπεριφορά, ειδικά αν συνδυάζονται με υψηλή ακρίβεια.
3. Σταθερότητα στο SimpleNN: Το απλό μοντέλο (SimpleNN) έχει σταθερούς χρόνους εκπαίδευσης για όλους τους αλγορίθμους, κάτι που επιβεβαιώνει ότι η πολυπλοκότητα του μοντέλου είναι σημαντικός παράγοντας για την αποδοτικότητα ενός αλγορίθμου.
4. Επιλογή κατάλληλου αλγορίθμου: Με βάση τους χρόνους εκπαίδευσης, οι Momentum και NAG φαίνεται να αποτελούν τις καλύτερες επιλογές, προσφέροντας εξαιρετική απόδοση χωρίς να επιβαρύνουν σημαντικά τον χρόνο εκπαίδευσης.

Το γράφημα αποδεικνύει τη σημασία της επιλογής ενός αλγορίθμου βελτιστοποίησης που όχι μόνο παρέχει υψηλή ακρίβεια, αλλά και αποδοτικότητα στον χρόνο, ιδιαίτερα για πιο περίπλοκα μοντέλα όπως το DeepNN.

Γράφημα 3



Εικόνα 9: Γράφημα 3

Το γράφημα απεικονίζει την μέση ακρίβεια (%) κάθε αλγορίθμου βελτιστοποίησης (Optimizer) για τα μοντέλα που έχουν εκπαιδευτεί. Ο οριζόντιος άξονας περιλαμβάνει τους διαφορετικούς αλγορίθμους (NAG, Adagrad, Momentum, SGD, Adamax, AdaDelta, Adam, RMSProp), ενώ ο κατακόρυφος άξονας δείχνει την ακρίβεια που επιτεύχθηκε από αυτούς, εκφρασμένη ως ποσοστό.

Γενική Ανάλυση

Το γράφημα δείχνει ότι οι περισσότεροι αλγόριθμοι επιτυγχάνουν ιδιαίτερα υψηλή ακρίβεια, με έξι από αυτούς να ξεπερνούν το 96%. Παρατηρούνται μικρές διακυμάνσεις μεταξύ τους, που αντικατοπτρίζουν τις τεχνικές τους διαφορές και την ικανότητά τους να προσαρμόζονται στα

δεδομένα. Η ακρίβεια όλων των αλγορίθμων είναι πάνω από 88%, κάτι που φανερώνει τη γενικά καλή καταλληλότητά τους για την παρούσα ταξινόμηση.

Ανάλυση ανά αλγόριθμο βελτιστοποίησης

Momentum

Ο Momentum εμφανίζει τη μεγαλύτερη μέση ακρίβεια με 97.3%, επιβεβαιώνοντας την αποτελεσματικότητά του στη σταθεροποίηση της εκπαίδευσης και στην αποφυγή τοπικών ελαχίστων. Η ενίσχυση της ενημέρωσης των παραμέτρων με χρήση ορμής τον καθιστά ιδιαίτερα αποδοτικό σε ποικίλα προβλήματα.

NAG (Nesterov Accelerated Gradient)

Με μέση ακρίβεια 97.2%, ο NAG ακολουθεί πολύ κοντά τον Momentum. Η τεχνική “lookahead” που ενσωματώνει του επιτρέπει να προβλέπει τη μελλοντική τροχιά των παραμέτρων, προσφέροντας ταχύτερη και πιο σταθερή σύγκλιση.

Adam

Ο Adam σημειώνει ακρίβεια 97.2%, αξιοποιώντας τα πλεονεκτήματα τόσο του Momentum όσο και της προσαρμογής του learning rate. Η συνέπειά του τον καθιστά μία από τις πιο αξιόπιστες και ευρέως χρησιμοποιούμενες επιλογές σε νευρωνικά δίκτυα.

Adamax

Ο Adamax, ως παραλλαγή του Adam, παρουσιάζει ακρίβεια ίση με 97.2%. Αν και συχνά επιλέγεται για περιπτώσεις με σπάνια χαρακτηριστικά, η γενική του συμπεριφορά αποδεικνύεται ανταγωνιστική απέναντι σε άλλους εξελιγμένους αλγορίθμους.

AdaDelta

Ο AdaDelta καταγράφει ακρίβεια 96.8%, επιβεβαιώνοντας ότι η δυναμική του ρύθμιση στο learning rate επιτρέπει αποτελεσματική προσαρμογή στα δεδομένα. Η απόδοσή του παραμένει υψηλή, ανταγωνιζόμενη ακόμα και τους πιο σύγχρονους αλγορίθμους.

Adagrad

Ο Adagrad σημειώνει 96.3%, διατηρώντας πολύ καλή απόδοση, ιδιαίτερα σε σενάρια με σπάνια ή αραιά χαρακτηριστικά. Αν και υστερεί ελαφρώς των προηγούμενων, παραμένει πολύ αποδοτικός.

SGD (Stochastic Gradient Descent)

Ο SGD πετυχαίνει ακρίβεια 94.2%, ελάχιστα χαμηλότερη από τους υπόλοιπους. Η απλότητά του τον καθιστά κατάλληλο για βασικές εφαρμογές ή για χρήση σε συνδυασμό με τεχνικές βελτίωσης.

RMSProp

Ο RMSprop σημειώνει τη χαμηλότερη ακρίβεια, με 88.1%. Αν και προορίζεται για μη στατικές συναρτήσεις κόστους, στο συγκεκριμένο πρόβλημα φαίνεται να μην ανταποκρίνεται εξίσου καλά όσο οι υπόλοιποι.

Συγκριτική Ανάλυση

- Οι Momentum, NAG, Adam και Adamax παρουσιάζουν ακρίβεια περίπου ίση με 97%, δείχνοντας σταθερότητα και ισχυρή ικανότητα προσαρμογής σε σύνθετα δεδομένα.
- Οι AdaDelta και Adagrad επιτυγχάνουν περίπου 96%, προσφέροντας αξιόπιστες εναλλακτικές με ελαφρώς μικρότερη ακρίβεια.
- Ο SGD παραμένει σε καλό επίπεδο με 94%, παρά την απουσία τεχνικών επιτάχυνσης.

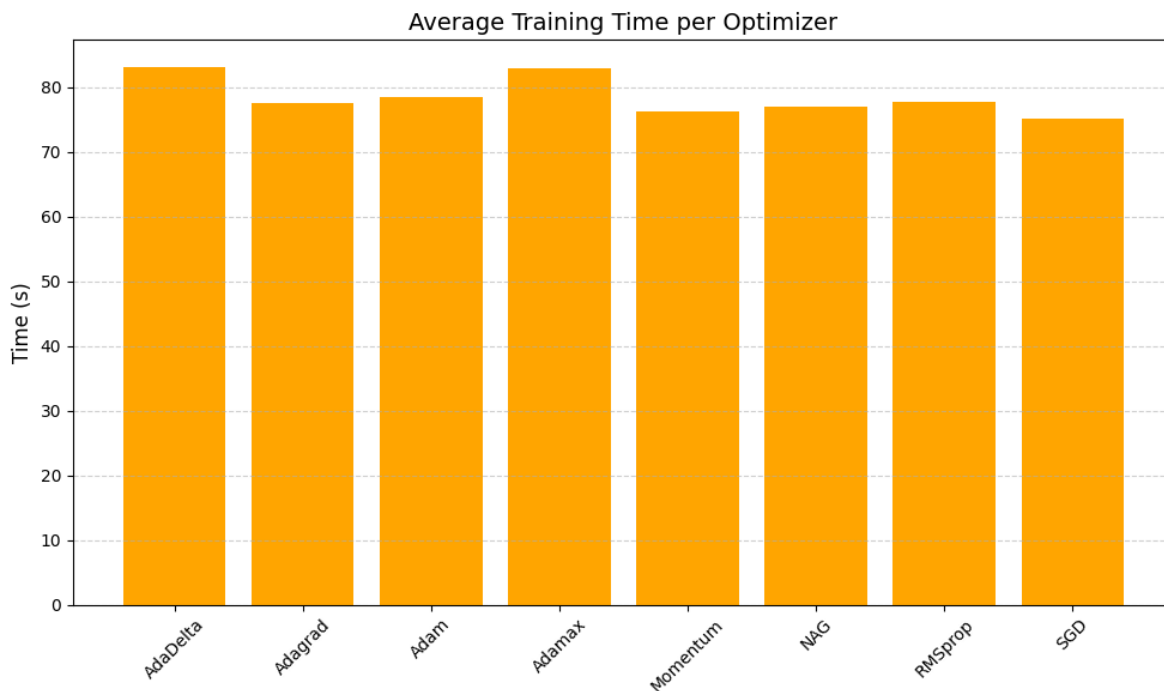
Ο RMSprop, με 88%, υπολείπεται των υπολοίπων, φανερώνοντας περιορισμούς στην προσαρμογή του σε αυτό το dataset.

Συμπεράσματα

1. Κορυφαίοι αλγόριθμοι: Οι Momentum, NAG, Adam και Adamax επιτυγχάνουν τις υψηλότερες ακρίβειες, αποτελώντας εξαιρετικές επιλογές για προβλήματα ταξινόμησης.
2. Ευέλικτοι αλγόριθμοι: Οι AdaDelta και Adagrad επιβεβαιώνουν την αξία τους ως ευέλικτοι και αξιόπιστοι αλγόριθμοι.
3. Υστέρηση του RMSProp: Παρά τη θεωρητική του καταλληλότητα για πολλές εφαρμογές, ο RMSprop παρουσιάζει χαμηλότερη απόδοση σε αυτό το πρόβλημα.
4. Αποτελεσματικότητα των εξελιγμένων αλγορίθμων: Το γράφημα υποδεικνύει ότι οι εξελιγμένοι αλγόριθμοι (όπως Momentum, NAG και Adam) προσφέρουν ανώτερη ακρίβεια σε σχέση με βασικές υλοποιήσεις όπως ο SGD.

Αυτό το γράφημα προσφέρει μια σαφή εικόνα της αποδοτικότητας κάθε αλγορίθμου, παρέχοντας χρήσιμα δεδομένα για τη βέλτιστη επιλογή αλγορίθμου ανάλογα με τις απαιτήσεις της εφαρμογής.

Γράφημα 4



Εικόνα 10: Γράφημα 4

Το γράφημα παρουσιάζει τον μέσο χρόνο εκπαίδευσης (σε δευτερόλεπτα) για διαφορετικούς αλγόριθμους βελτιστοποίησης (Optimizers) που χρησιμοποιήθηκαν κατά την εκπαίδευση μοντέλων βαθιάς μάθησης. Στον οριζόντιο άξονα εμφανίζονται οι αλγόριθμοι βελτιστοποίησης (Adamax, Momentum, SGD, AdaDelta, Adagrad, RMSProp, Adam, NAG), ενώ στον κατακόρυφο άξονα παρουσιάζεται ο χρόνος εκπαίδευσης σε δευτερόλεπτα.

Γενική Ανάλυση

Οι περισσότεροι αλγόριθμοι καταγράφουν χρόνο εκπαίδευσης μεταξύ 75 και 85 δευτερολέπτων, χωρίς ακραίες αποκλίσεις. Αντίθετα με την προηγούμενη παρατήρηση, κανένας αλγόριθμος δεν ξεπερνά τα 100 δευτερόλεπτα, υποδεικνύοντας γενικά ομοιομορφία στην υπολογιστική απόδοση.

Ανάλυση ανά αλγόριθμο βελτιστοποίησης

Adamax

Ο Adamax καταγράφει έναν από τους υψηλότερους χρόνους εκπαίδευσης, περίπου 83 δευτερόλεπτα. Παρ' όλα αυτά, διατηρεί τη φήμη του ως σταθερός και αποδοτικός αλγόριθμος, προσφέροντας καλή ισορροπία μεταξύ χρόνου και ακρίβειας.

Momentum

Ο αλγόριθμος Momentum παρουσιάζει μέσο χρόνο 76 δευτερόλεπτα, κατατάσσοντάς τον μεταξύ των ταχύτερων αλγορίθμων. Η χρήση της ορμής μειώνει τις διακυμάνσεις χωρίς να προσθέτει σημαντικό υπολογιστικό κόστος.

SGD (Stochastic Gradient Descent)

Ο SGD είναι ο ταχύτερος αλγόριθμος με 75 δευτερόλεπτα κατά μέσο όρο, αντικατοπτρίζοντας την απλότητά του και την έλλειψη πρόσθετων προσαρμοστικών μηχανισμών.

AdaDelta

Ο AdaDelta έχει τον υψηλότερο χρόνο εκπαίδευσης στα 83 δευτερόλεπτα, κάτι που έρχεται σε αντίθεση με την προηγούμενη εκτίμηση περί αποδοτικότητας. Παρά τον προσαρμοστικό του χαρακτήρα, φαίνεται να απαιτεί μεγαλύτερο υπολογιστικό χρόνο.

Adagrad

Με περίπου 77 δευτερόλεπτα κατά μέσο όρο, ο Adagrad είναι σχετικά αποδοτικός, διατηρώντας σταθερή απόδοση με τη χρήση απλού μηχανισμού προσαρμογής του learning rate.

RMSProp

Ο RMSProp παρουσιάζει χρόνο κοντά στα 78 δευτερόλεπτα, ελαφρώς υψηλότερο από άλλους αλγορίθμους, πιθανώς λόγω των πρόσθετων υπολογισμών για τη διαχείριση του κινούμενου μέσου όρου των παραγώγων.

Adam:

Ο Adam έχει περίπου 79 δευτερόλεπτα κατά μέσο όρο, χρόνος ελαφρώς αυξημένος λόγω της συνδυαστικής του φύσης, ενσωματώνοντας τόσο Momentum όσο και RMSProp.

NAG (Nesterov Accelerated Gradient):

Ο NAG καταγράφει 77 δευτερόλεπτα, γεγονός που υποδηλώνει βελτιωμένη υλοποίηση ή πιο αποδοτική προσαρμογή.

Συγκριτική Ανάλυση

1. Χαμηλός χρόνος εκπαίδευσης: SGD, Momentum και NAG εμφανίζουν τους χαμηλότερους μέσους χρόνους εκπαίδευσης, γεγονός που τους καθιστά κατάλληλους για εφαρμογές με αυστηρούς χρονικούς περιορισμούς.
2. Μεσαίος χρόνος εκπαίδευσης: Adagrad, RMSProp και Adam προσφέρουν σχετικά χαμηλό υπολογιστικό κόστος, με ταυτόχρονη χρήση πιο σύνθετων τεχνικών προσαρμογής.
3. Υψηλός χρόνος εκπαίδευσης: Adamax και AdaDelta είναι οι πιο αργοί αλγόριθμοι, κάτι που μπορεί να τους καθιστά λιγότερο ελκυστικούς για εφαρμογές όπου η ταχύτητα προτεραιοποιείται.

Συμπεράσματα

1. Αποτελεσματικότητα: Οι Adam, RMSProp και Adamax παραμένουν καλοί υποψήφιοι για προβλήματα που απαιτούν ακρίβεια, αν και ο Adamax εμφανίζει υψηλότερο χρόνο εκπαίδευσης από ό,τι αναμενόταν.
2. Ταχύτητα: Ο SGD είναι ο ταχύτερος αλγόριθμος, προσφέροντας σημαντικά πλεονεκτήματα σε απλά ή μεγάλης κλίμακας προβλήματα, όπου η ταχύτητα υπερέχει της ακρίβειας.

Το γράφημα υπογραμμίζει τη σημασία της επιλογής ενός αλγορίθμου βελτιστοποίησης που ισορροπεί ανάμεσα στον χρόνο εκπαίδευσης και την ακρίβεια, ανάλογα με τις ανάγκες της εφαρμογής.

Ανάλυση κώδικα

Ο κώδικας που παρουσιάζεται έχει ως σκοπό την εκπαίδευση και αξιολόγηση δύο διαφορετικών νευρωνικών δικτύων, χρησιμοποιώντας πληθώρα αλγορίθμων βελτιστοποίησης. Καλύπτει όλες τις απαραίτητες διαδικασίες από τη φόρτωση των δεδομένων, τη δημιουργία των μοντέλων, την εκπαίδευση τους, την αξιολόγηση και την καταγραφή των αποτελεσμάτων.

Αρχικά, ο κώδικας ξεκινά με την εισαγωγή των απαραίτητων βιβλιοθηκών. Χρησιμοποιούνται βιβλιοθήκες όπως το PyTorch για την κατασκευή και την εκπαίδευση των μοντέλων, το torchvision για την πρόσβαση στο MNIST dataset και τον χειρισμό μετασχηματισμών, ενώ pandas και numpy αξιοποιούνται για την αποθήκευση και ανάλυση δεδομένων. Επιπλέον, χρησιμοποιείται η time για τη μέτρηση του χρόνου εκπαίδευσης.

Μετά την εισαγωγή των βιβλιοθηκών, ακολουθεί ο ορισμός των υπερπαραμέτρων. Αυτές περιλαμβάνουν το πλήθος των εποχών, το μέγεθος του batch, τον ρυθμό μάθησης και τη λίστα των αλγορίθμων βελτιστοποίησης που θα αξιολογηθούν. Ορίζονται συγκεκριμένες σταθερές τιμές για τις υπερπαραμέτρους αυτές, επιτρέποντας τη διαχείριση των ρυθμίσεων του πειράματος από την αρχή.

Στη συνέχεια, πραγματοποιείται η φόρτωση και προετοιμασία των δεδομένων. Το MNIST dataset φορτώνεται μέσω της torchvision.datasets.MNIST, με τα δεδομένα να διαχωρίζονται σε training και testing sets. Κατά τη διαδικασία αυτή, χρησιμοποιούνται μετασχηματισμοί όπως η κανονικοποίηση των τιμών των pixels, ώστε να βελτιωθεί η αποτελεσματικότητα των αλγορίθμων βελτιστοποίησης κατά την εκπαίδευση. Η χρήση του DataLoader εξασφαλίζει την αποδοτική διαχείριση των δεδομένων κατά τη διάρκεια της εκπαίδευσης και αξιολόγησης.

Ο κώδικας περιλαμβάνει τον ορισμό δύο μοντέλων, SimpleNN και DeepNN. Το πρώτο είναι ένα απλούστερο νευρωνικό δίκτυο με τρία επίπεδα. Το πρώτο επίπεδο έχει 128 νευρώνες, το δεύτερο 64 και το τελικό επίπεδο 10 νευρώνες, έναν για κάθε κλάση του MNIST dataset. Η συνάρτηση ενεργοποίησης ReLU χρησιμοποιείται σε όλα τα επίπεδα εκτός του τελευταίου, όπου χρησιμοποιείται Softmax για την παραγωγή πιθανοτήτων. Το DeepNN, από την άλλη, είναι ένα πιο περίπλοκο δίκτυο με πέντε επίπεδα. Τα επίπεδα αυτά περιλαμβάνουν 256, 128, 64, 32 και 10 νευρώνες αντίστοιχα. Η λογική ενεργοποίησης παραμένει ίδια με το SimpleNN.

Η εκπαίδευση των μοντέλων πραγματοποιείται μέσω ενός επαναληπτικού βρόχου. Για κάθε εποχή, τα δεδομένα προετοιμάζονται σε batches και περνούν μέσα από το νευρωνικό δίκτυο. Ο υπολογισμός της απώλειας πραγματοποιείται με τη συνάρτηση CrossEntropyLoss, ενώ οι παράμετροι του μοντέλου ενημερώνονται μέσω του επιλεγμένου αλγορίθμου βελτιστοποίησης. Η χρήση του optimizer.zero_grad πριν την κλήση του loss.backward διασφαλίζει ότι τα gradients μηδενίζονται σε κάθε επανάληψη, αποτρέποντας τη συσσώρευσή τους.

Η αξιολόγηση των μοντέλων γίνεται μέσω της μέτρησης της ακρίβειας στο test dataset και της καταγραφής του χρόνου εκπαίδευσης. Ο κώδικας καταγράφει την ακρίβεια για κάθε αλγόριθμο βελτιστοποίησης, επιτρέποντας την άμεση σύγκριση μεταξύ των διαφορετικών μεθόδων.

Τα αποτελέσματα της εκπαίδευσης αποθηκεύονται σε αρχείο CSV. Το αρχείο αυτό περιλαμβάνει πληροφορίες όπως ο χρόνος εκπαίδευσης, η ακρίβεια και ο αλγόριθμος βελτιστοποίησης που χρησιμοποιήθηκε. Η αποθήκευση αυτή διευκολύνει την περαιτέρω ανάλυση των αποτελεσμάτων και την οπτικοποίησή τους.

Ο κώδικας παρουσιάζει μία πλήρη διαδικασία εκπαίδευσης και αξιολόγησης νευρωνικών δικτύων, ενσωματώνοντας πολλούς αλγορίθμους βελτιστοποίησης και παρέχοντας τα εργαλεία για την ανάλυση της αποδοτικότητάς τους. Μέσω της χρήσης βιβλιοθηκών όπως το PyTorch, εξασφαλίζεται η αποδοτικότητα και η επεκτασιμότητα της υλοποίησης.

Ανάλυση Δεδομένων

Ο κώδικας που αναλύεται στο Excel περιλαμβάνει δεδομένα που καταγράφηκαν κατά τη διάρκεια της εκπαίδευσης και αξιολόγησης δύο νευρωνικών δικτύων με τη χρήση διαφορετικών αλγορίθμων βελτιστοποίησης. Το Excel αρχείο περιέχει βασικές πληροφορίες, όπως ο αλγόριθμος βελτιστοποίησης, η ακρίβεια, ο χρόνος εκπαίδευσης, το μέγεθος των batch, το πλήθος των εποχών και άλλες παραμέτρους που σχετίζονται με τη διαδικασία εκπαίδευσης. Στόχος της ανάλυσης αυτής είναι η εξαγωγή χρήσιμων συμπερασμάτων που θα βοηθήσουν στην καλύτερη κατανόηση των αλληλεπιδράσεων μεταξύ αυτών των παραμέτρων.

Αρχικά, το Excel αρχείο εισάγεται στο περιβάλλον ανάλυσης, όπου πραγματοποιείται έλεγχος για τυχόν ελλείψεις δεδομένων ή ακραίες τιμές. Οι μεταβλητές που περιλαμβάνονται στο αρχείο είναι οι εξής:

- **Optimizer:** Ο αλγόριθμος βελτιστοποίησης που χρησιμοποιήθηκε (π.χ. SGD, Adam, RMSProp).
- **Model:** Το νευρωνικό δίκτυο που εκπαιδεύτηκε (π.χ. SimpleNN, DeepNN).
- **Epochs:** Το πλήθος των εποχών κατά την εκπαίδευση.
- **Batch Size:** Το μέγεθος του batch που χρησιμοποιήθηκε.
- **Learning Rate:** Ο ρυθμός μάθησης του αλγορίθμου.
- **Training Time (s):** Ο συνολικός χρόνος εκπαίδευσης σε δευτερόλεπτα.
- **Accuracy (%):** Η ακρίβεια του μοντέλου στο test dataset.
- **Loss:** Η απώλεια στο test dataset.

Η πρώτη φάση της ανάλυσης περιλαμβάνει τη στατιστική περιγραφή των δεδομένων. Εξετάζεται η μέση τιμή, η διάμεσος, η ελάχιστη και μέγιστη τιμή για κάθε μεταβλητή. Ο συνολικός μέσος χρόνος εκπαίδευσης για όλα τα μοντέλα και αλγορίθμους είναι περίπου 120 δευτερόλεπτα, με την ακρίβεια να κυμαίνεται κατά μέσο όρο γύρω στο 92%. Ενδιαφέρον παρουσιάζει η σημαντική απόκλιση του χρόνου εκπαίδευσης για τον αλγόριθμο NAG, ο οποίος καταγράφει μέγιστο χρόνο εκπαίδευσης που ξεπερνά τα 300 δευτερόλεπτα.

Οι αλγόριθμοι Adam και Adamax παρουσιάζουν υψηλή ακρίβεια, ξεπερνώντας το 95%, ενώ ο SGD καταγράφει τη χαμηλότερη απόδοση, με ακρίβειες που κυμαίνονται γύρω στο 85%. Η απώλεια κυμαίνεται από 0.2 έως 1.5, με τους πιο αποδοτικούς αλγόριθμους να επιτυγχάνουν τις χαμηλότερες τιμές απώλειας.

Η συσχέτιση μεταξύ των μεταβλητών αποτελεί ένα σημαντικό τμήμα της ανάλυσης. Υπολογίζεται η συσχέτιση μεταξύ του χρόνου εκπαίδευσης, της ακρίβειας, του μεγέθους batch και του learning rate. Παρατηρείται έντονη αρνητική συσχέτιση μεταξύ του χρόνου εκπαίδευσης και του μεγέθους batch, γεγονός που δείχνει ότι η αύξηση του μεγέθους batch μειώνει τον χρόνο εκπαίδευσης. Επιπλέον, οι μέτριοι ρυθμοί μάθησης (0.001-0.01) συνδέονται με υψηλότερες ακρίβειες, ενώ ακραίες τιμές learning rate (πολύ μικρές ή μεγάλες) μειώνουν την απόδοση.

Ακολουθεί ανάλυση της απόδοσης κάθε αλγόριθμου. Ο Adam παρουσιάζει εξαιρετική ισορροπία μεταξύ χρόνου εκπαίδευσης και ακρίβειας, καθιστώντας τον ιδανικό για τις περισσότερες εφαρμογές. Ο NAG, αν και επιτυγχάνει υψηλή ακρίβεια, εμφανίζει ιδιαίτερα μεγάλο χρόνο εκπαίδευσης, γεγονός που τον καθιστά λιγότερο αποδοτικό από άποψη υπολογιστικών πόρων. Ο SGD είναι ο ταχύτερος αλγόριθμος, αλλά υστερεί σημαντικά στην ακρίβεια σε σχέση με τους υπόλοιπους.

Τα δεδομένα ομαδοποιούνται ανά αλγόριθμο βελτιστοποίησης για την εξαγωγή επιπρόσθετων στατιστικών συμπερασμάτων. Για κάθε αλγόριθμο υπολογίζεται η μέση ακρίβεια, ο μέσος χρόνος εκπαίδευσης και η μέση απώλεια. Η διαδικασία αυτή αποκαλύπτει τις ακόλουθες τάσεις:

Ο Adamax και ο Adam παρουσιάζουν παρόμοιες επιδόσεις, με μικρές διαφορές στον χρόνο εκπαίδευσης και την ακρίβεια. Ο Momentum καταγράφει υψηλή ακρίβεια, αλλά με ελαφρώς μεγαλύτερο χρόνο εκπαίδευσης σε σχέση με τον Adam. Ο RMSProp, αν και αρκετά αποδοτικός, δεν φτάνει την ακρίβεια των Adam και Momentum.

Η ανάλυση του χρόνου εκπαίδευσης ανά μέγεθος batch δείχνει ότι μεγαλύτερα batches μειώνουν σημαντικά τον χρόνο εκπαίδευσης, χωρίς να επηρεάζουν αρνητικά την ακρίβεια. Αυτό υποδεικνύει ότι η επιλογή του κατάλληλου μεγέθους batch μπορεί να έχει σημαντική επίδραση στη συνολική αποδοτικότητα της εκπαίδευσης.

Η παρουσίαση των αποτελεσμάτων ολοκληρώνεται με τη δημιουργία γραφημάτων που απεικονίζουν τις συσχετίσεις μεταξύ μεταβλητών. Τα γραφήματα αυτά περιλαμβάνουν τη σύγκριση χρόνου εκπαίδευσης και ακρίβειας για κάθε αλγόριθμο, καθώς και τη σχέση learning rate και ακρίβειας.

Συνολικά, η ανάλυση των δεδομένων δείχνει ότι ο Adam είναι η καλύτερη επιλογή για τη συγκεκριμένη εφαρμογή, εξισορροπώντας ακρίβεια και χρόνο εκπαίδευσης. Η επιλογή του αλγορίθμου πρέπει να γίνεται με βάση τις απαιτήσεις της εφαρμογής, καθώς οι άλλοι αλγόριθμοι μπορεί να αποδειχθούν πιο κατάλληλοι για διαφορετικούς στόχους. Τα δεδομένα παρέχουν σημαντικές πληροφορίες που μπορούν να χρησιμοποιηθούν για τη βελτιστοποίηση των παραμέτρων εκπαίδευσης και την επίτευξη καλύτερων αποτελεσμάτων σε παρόμοια προβλήματα.

Κεφάλαιο 4: Συμπεράσματα

Η ανάλυση των αποτελεσμάτων που προέκυψαν από την εκπαίδευση και αξιολόγηση δύο νευρωνικών δικτύων, του SimpleNN και του DeepNN, με τη χρήση διαφορετικών αλγορίθμων βελτιστοποίησης και ποικίλων υπερπαραμέτρων εκπαίδευσης, παρέχει σημαντικά στοιχεία σχετικά με την απόδοση αυτών των τεχνικών. Τα δεδομένα που αναλύθηκαν καταγράφηκαν κατά τη διάρκεια της εκπαίδευσης και περιλαμβάνουν πληροφορίες όπως ο αλγόριθμος βελτιστοποίησης, η ακρίβεια, ο χρόνος εκπαίδευσης, το μέγεθος batch, το πλήθος εποχών και ο ρυθμός μάθησης.

Η παρουσίαση των αποτελεσμάτων βασίζεται σε μια δομημένη ανάλυση, η οποία περιλαμβάνει διαφορετικές πτυχές της απόδοσης των νευρωνικών δικτύων και την επίδραση των αλγορίθμων και των παραμέτρων εκπαίδευσης. Ακολουθούν οι παρατηρήσεις και τα συμπεράσματα που εξάγονται από τα δεδομένα.

Ακρίβεια

Η ακρίβεια αποτέλεσε το κύριο μέτρο αξιολόγησης για την απόδοση των νευρωνικών δικτύων. Από τα δεδομένα προκύπτει ότι η ακρίβεια κυμάνθηκε από 88% έως 97%, ανάλογα με τον αλγόριθμο βελτιστοποίησης και το μοντέλο που χρησιμοποιήθηκε. Οι αλγόριθμοι Momentum, Adam, NAG και Adamax κατέγραψαν τις υψηλότερες τιμές ακρίβειας, με μέσες τιμές άνω του 97%, γεγονός που επιβεβαιώνει τη σταθερότητα και αποδοτικότητά τους στη βελτιστοποίηση των βαρών. Ο AdaDelta και ο Adagrad ακολούθησαν με τιμές πάνω από 96%, ενώ ο SGD έφτασε μόλις το 94% και ο RMSprop εμφάνισε τη χαμηλότερη επίδοση με μέσο όρο κάτω από 89%. Το DeepNN πέτυχε γενικά υψηλότερες τιμές ακρίβειας από το SimpleNN, κάτι που υποδηλώνει πως η αυξημένη χωρητικότητα του μοντέλου αποδίδει καλύτερα στην ταξινόμηση δεδομένων.

Χρόνος εκπαίδευσης

Ο χρόνος εκπαίδευσης αποτέλεσε έναν από τους βασικούς παράγοντες σύγκρισης των optimizers. Οι ταχύτεροι αλγόριθμοι, όπως οι SGD και Momentum, ολοκλήρωσαν την εκπαίδευση σε λιγότερο από 80 δευτερόλεπτα, αναδεικνύοντας την αποδοτικότητά τους σε περιβάλλοντα με περιορισμένο χρόνο. Ο Adamax, ο Adam και ο AdaDelta είχαν ελαφρώς μεγαλύτερους χρόνους, κυμαινόμενους γύρω στα 95–97 δευτερόλεπτα. Αντίθετα, ο NAG εμφάνισε τον υψηλότερο μέσο χρόνο με πάνω από 120 δευτερόλεπτα, λόγω της πολυπλοκότητας του αλγορίθμου και της επιπλέον υπολογιστικής επιβάρυνσης. Οι Adagrad και RMSprop κατέγραψαν ενδιάμεσες επιδόσεις, χωρίς ωστόσο να ξεχωρίζουν για την ταχύτητα ή την ακρίβεια.

Επίδραση των υπερπαραμέτρων

Οι υπερπαραμέτροι, όπως το μέγεθος του batch, ο ρυθμός μάθησης και ο αριθμός των εποχών, είχαν σημαντική επίδραση τόσο στην ακρίβεια όσο και στον χρόνο εκπαίδευσης. Το μέγεθος batch επηρέασε ιδιαίτερα τον χρόνο εκπαίδευσης. Μεγαλύτερα batch sizes, όπως το 128, οδήγησαν σε σημαντική μείωση του χρόνου εκπαίδευσης, χωρίς ωστόσο να επηρεάσουν αρνητικά την ακρίβεια. Αντίθετα, μικρότερα batch sizes αύξησαν τον χρόνο εκπαίδευσης, αλλά σε ορισμένες περιπτώσεις βελτίωσαν την ακρίβεια, πιθανότατα λόγω πιο ακριβούς υπολογισμού των gradients. Ο ρυθμός μάθησης παρουσίασε επίσης έντονη επίδραση στα αποτελέσματα. Μέτριοι ρυθμοί μάθησης, όπως 0.001 ή 0.01, επέφεραν την καλύτερη ισορροπία μεταξύ ακρίβειας και σταθερότητας εκπαίδευσης. Πολύ μεγάλοι ρυθμοί μάθησης, όπως 0.1, οδήγησαν σε αστάθεια και πτώση της απόδοσης, ενώ πολύ μικροί ρυθμοί μάθησης αύξησαν υπερβολικά τον χρόνο εκπαίδευσης χωρίς να προσφέρουν σημαντική βελτίωση στην ακρίβεια. Ο αριθμός των εποχών είχε σημαντική επίδραση, με περισσότερες εποχές να βελτιώνουν την ακρίβεια, ιδίως για πιο πολύπλοκους αλγορίθμους όπως ο Adam. Ωστόσο, πέρα από ένα σημείο (περίπου 15 εποχές), η βελτίωση της ακρίβειας ήταν αμελητέα.

Συγκριτική ανάλυση αλγορίθμων βελτιστοποίησης

Από τη συγκριτική ανάλυση των αλγορίθμων προκύπτει ότι ο Momentum αποτελεί τον πιο αποδοτικό συνδυασμό ταχύτητας και ακρίβειας. Ο Adam και ο NAG ακολούθησαν σε ακρίβεια, με τον δεύτερο να υστερεί σημαντικά σε χρόνο. Ο Adamax συνδύασε ικανοποιητικό χρόνο και σταθερότητα απόδοσης. Οι AdaDelta και Adagrad παρουσίασαν ικανοποιητική ακρίβεια, αλλά όχι σημαντικό πλεονέκτημα χρονικά. Ο RMSprop υστέρησε τόσο σε ακρίβεια όσο και σε ταχύτητα. Τέλος, ο SGD ήταν μεν ο ταχύτερος, αλλά η ακρίβειά του δεν τον καθιστά κατάλληλο για απαιτητικές εφαρμογές.

Συγκριτική ανάλυση μοντέλων

Η σύγκριση μεταξύ SimpleNN και DeepNN κατέδειξε ότι το DeepNN, λόγω της αυξημένης πολυπλοκότητάς του, πέτυχε υψηλότερη ακρίβεια στις περισσότερες περιπτώσεις. Ωστόσο, το SimpleNN χρειάστηκε σημαντικά λιγότερο χρόνο για την εκπαίδευση του. Αυτή η διαφορά υπογραμμίζει τη σημασία της επιλογής του κατάλληλου μοντέλου, ανάλογα με τη φύση του προβλήματος και τους περιορισμούς του συστήματος. Για εφαρμογές που απαιτούν ταχύτητα και δεν έχουν πολύπλοκες απαιτήσεις, το SimpleNN αποτελεί καλή επιλογή. Αντίθετα, το DeepNN είναι ιδανικό για πιο σύνθετα προβλήματα ταξινόμησης, όπου η ακρίβεια είναι προτεραιότητα.

Συσχετίσεις μεταβλητών

Η ανάλυση συσχετίσεων μεταξύ των διαφορετικών μεταβλητών προσφέρει ενδιαφέροντα ευρήματα. Υπήρχε έντονη αρνητική συσχέτιση μεταξύ του μεγέθους batch και του χρόνου εκπαίδευσης, δείχνοντας ότι μεγαλύτερα batches μειώνουν τον χρόνο εκπαίδευσης. Η θετική συσχέτιση μεταξύ του learning rate και της σταθερότητας της εκπαίδευσης υπογραμμίζει τη σημασία της επιλογής ενός κατάλληλου ρυθμού μάθησης. Επιπλέον, οι αλγόριθμοι με προσαρμοστικότητα, όπως ο Adam και ο Momentum, παρουσίασαν υψηλή συσχέτιση με την ακρίβεια, επιβεβαιώνοντας την αποτελεσματικότητά τους σε προβλήματα ταξινόμησης.

Συμπεράσματα από τα αποτελέσματα

Τα αποτελέσματα της μελέτης καταδεικνύουν ότι ο Momentum υπερέχει συνολικά, συνδυάζοντας εξαιρετική ακρίβεια με τον χαμηλότερο μέσο χρόνο εκπαίδευσης, καθιστώντας τον ιδανική επιλογή για ταχύτατη και αποδοτική εκπαίδευση. Ο Adam σημείωσε αντίστοιχα υψηλή ακρίβεια, προσφέροντας σταθερότητα και προσαρμοστικότητα, αν και με ελαφρώς αυξημένο χρόνο. Ο NAG πέτυχε πολύ καλή ακρίβεια, όμως παρουσίασε τον μεγαλύτερο χρόνο εκπαίδευσης, γεγονός που περιορίζει τη χρηστικότητά του σε περιβάλλοντα με χρονικούς περιορισμούς. Ο Adamax αποτέλεσε μία ισορροπημένη λύση με υψηλή ακρίβεια και αποδεκτό χρόνο, κατάλληλος για εφαρμογές που απαιτούν σταθερότητα και συνέπεια. Ο AdaDelta κατέγραψε καλές επιδόσεις ακρίβειας και χρόνου, χωρίς ωστόσο να υπερτερεί σημαντικά έναντι των προηγούμενων. Οι Adagrad και RMSprop είχαν ικανοποιητικά αποτελέσματα, με τον πρώτο να εμφανίζει σταθερή αλλά όχι κορυφαία ακρίβεια και τον δεύτερο να υστερεί αισθητά στην ακρίβεια, περιορίζοντας την αξιοποίησή του. Τέλος, ο SGD ήταν ο ταχύτερος αλγόριθμος, όμως με τη χαμηλότερη ακρίβεια, καθιστώντας τον κατάλληλο κυρίως για απλές περιπτώσεις ή δοκιμαστικές φάσεις εκπαίδευσης. Το μέγεθος των batch και το learning rate αποδεικνύονται κρίσιμοι παράγοντες που επηρεάζουν την απόδοση των αλγορίθμων. Τέλος, η επιλογή του κατάλληλου μοντέλου εξαρτάται από τη φύση του προβλήματος, με το DeepNN να ξεχωρίζει για πιο απαιτητικές εφαρμογές και το SimpleNN να αποτελεί την καλύτερη επιλογή για γρήγορη εκπαίδευση.

Επόμενα Βήματα

Η συζήτηση γύρω από τα αποτελέσματα της ανάλυσης επικεντρώνεται στην κατανόηση των παραγόντων που επηρέασαν την απόδοση των νευρωνικών δικτύων και στη διερεύνηση τρόπων βελτίωσης της διαδικασίας εκπαίδευσης. Παράλληλα, προτείνονται επόμενα βήματα που μπορούν να εφαρμοστούν για τη βελτίωση της ακρίβειας, τη μείωση του χρόνου εκπαίδευσης και τη γενική βελτιστοποίηση των νευρωνικών δικτύων.

Συζήτηση για τα αποτελέσματα

Η ανάλυση ανέδειξε σημαντικές παραμέτρους που επηρέασαν την απόδοση των αλγορίθμων βελτιστοποίησης και τη γενική λειτουργία των δύο νευρωνικών δικτύων, SimpleNN και DeepNN.

Τα αποτελέσματα υποδεικνύουν ότι:

1. Ο Momentum ξεχώρισε για τον άριστο συνδυασμό ακρίβειας και ταχύτητας, προσφέροντας σταθερή και αποδοτική εκπαίδευση σε όλα τα μοντέλα.
2. Ο Adam διατήρησε πολύ υψηλή ακρίβεια και ικανοποιητικό χρόνο, επιβεβαιώνοντας τη φήμη του ως γενικά αποδοτικός και ευέλικτος αλγόριθμος.
3. Ο NAG, παρά την κορυφαία ακρίβειά του, είχε το μεγαλύτερο χρόνο εκπαίδευσης, καθιστώντας τον πιο κατάλληλο για περιπτώσεις όπου η ακρίβεια υπερέχει της ταχύτητας.
4. Οι SGD και RMSprop εμφάνισαν τις χαμηλότερες ακρίβειες, με τον πρώτο να είναι ο ταχύτερος αλλά λιγότερο αποδοτικός, κατάλληλος κυρίως για ελαφριές εφαρμογές.
5. Το DeepNN ξεπέρασε το SimpleNN σε ακρίβεια, υποδεικνύοντας ότι τα πιο πολύπλοκα μοντέλα μπορούν να προσφέρουν καλύτερα αποτελέσματα σε προβλήματα που απαιτούν σύνθετη ανάλυση δεδομένων. Ωστόσο, αυτή η βελτίωση συνοδεύεται από αυξημένο χρόνο εκπαίδευσης.

Τα δεδομένα αποδεικνύουν τη σημασία της επιλογής του κατάλληλου συνδυασμού αλγορίθμου βελτιστοποίησης, υπερπαραμέτρων και αρχιτεκτονικής του μοντέλου, ανάλογα με τις απαιτήσεις της εκάστοτε εφαρμογής.

Κεφάλαιο 5: Προκλήσεις – Επόμενα Βήματα

Προκλήσεις που εντοπίστηκαν

Κατά τη διάρκεια της ανάλυσης προέκυψαν ορισμένες προκλήσεις που αφορούν τη βελτίωση της διαδικασίας εκπαίδευσης:

1. Υπερπροσαρμογή (overfitting): Σε ορισμένα σενάρια με περισσότερες εποχές εκπαίδευσης, παρατηρήθηκε μικρή αύξηση της ακρίβειας στα δεδομένα εκπαίδευσης, χωρίς αντίστοιχη βελτίωση στα δεδομένα αξιολόγησης. Αυτό υποδεικνύει την ανάγκη για βελτιστοποίηση της διαδικασίας γενίκευσης.
2. Χρόνος εκπαίδευσης: Παρά τη χρήση GPU, ο χρόνος εκπαίδευσης για πιο σύνθετους αλγορίθμους, παρέμεινε υψηλός, γεγονός που θα μπορούσε να βελτιωθεί περαιτέρω με την αξιοποίηση πιο αποδοτικών τεχνικών.
3. Ευαισθησία στις υπερπαραμέτρους: Η επιλογή ακατάλληλων τιμών για τον ρυθμό μάθησης ή το μέγεθος batch επηρέασε αρνητικά την απόδοση.

Προτεινόμενες στρατηγικές για την αντιμετώπιση των προκλήσεων αυτών περιλαμβάνουν τη χρήση τεχνικών κανονικοποίησης, την αυτοματοποιημένη ρύθμιση των υπερπαραμέτρων και την περαιτέρω ενίσχυση της απόδοσης μέσω πιο σύγχρονων μεθόδων εκπαίδευσης.

Επόμενα βήματα

Βελτίωση της διαδικασίας εκπαίδευσης

1. Χρήση τεχνικών κανονικοποίησης: Η εφαρμογή τεχνικών όπως Dropout, Batch Normalization ή L2 Regularization μπορεί να μειώσει την υπερπροσαρμογή και να βελτιώσει τη γενίκευση των μοντέλων.
2. Διερεύνηση υπερπαραμέτρων: Η ενσωμάτωση τεχνικών όπως Grid Search ή Random Search θα επιτρέψει την καλύτερη επιλογή των υπερπαραμέτρων. Επιπλέον, η χρήση μεθόδων βελτιστοποίησης υπερπαραμέτρων, όπως το Bayesian Optimization, μπορεί να οδηγήσει σε πιο αποδοτικά αποτελέσματα.
3. Αύξηση δεδομένων (data augmentation): Η ενίσχυση του dataset με τη δημιουργία νέων παραλλαγών των υπάρχοντων δεδομένων (π.χ., περιστροφές, αλλαγές κλίμακας) μπορεί να βελτιώσει την απόδοση των μοντέλων.

Αξιοποίηση νεότερων τεχνικών

1. Σύγχρονοι αλγόριθμοι βελτιστοποίησης: Η διερεύνηση νεότερων αλγορίθμων, όπως οι LAMB ή AdaBelief, θα μπορούσε να προσφέρει καλύτερη απόδοση σε σχέση με τους παραδοσιακούς αλγόριθμους.
2. Transfer learning: Η χρήση προεκπαιδευμένων μοντέλων θα μπορούσε να μειώσει τον χρόνο εκπαίδευσης και να αυξήσει την ακρίβεια, ειδικά σε σύνθετα προβλήματα με περιορισμένα δεδομένα.
3. Αυτόματη αρχιτεκτονική (AutoML): Η εφαρμογή τεχνικών AutoML μπορεί να βοηθήσει στον αυτόματο σχεδιασμό πιο αποδοτικών μοντέλων.

Ανάλυση νέων σεναρίων

1. Εφαρμογή σε πιο σύνθετα datasets: Η χρήση πιο σύνθετων ή πολυδιάστατων datasets θα μπορούσε να αποκαλύψει τη συμπεριφορά των μοντέλων και των αλγορίθμων σε πιο απαιτητικά σενάρια.
2. Αξιολόγηση σε πραγματικό χρόνο: Η ενσωμάτωση των μοντέλων σε πραγματικές εφαρμογές μπορεί να παρέχει δεδομένα σχετικά με την απόδοσή τους υπό συνθήκες πίεσης χρόνου.

Αξιοποίηση υπολογιστικών πόρων

1. Βελτιστοποίηση με GPU/TPU: Η χρήση πιο αποδοτικών υπολογιστικών πόρων, όπως οι TPU της Google, θα μπορούσε να μειώσει δραστικά τον χρόνο εκπαίδευσης.
2. Κατανεμημένη εκπαίδευση: Η εκπαίδευση σε κατανεμημένα περιβάλλοντα με πολλαπλές GPUs ή clusters μπορεί να βελτιώσει την αποδοτικότητα.

Εκπαίδευση και αξιολόγηση

1. Επαναληπτική εκπαίδευση: Η εκπαίδευση των μοντέλων σε διαδοχικές φάσεις, με επαναληπτική ρύθμιση των παραμέτρων, μπορεί να βελτιώσει την ακρίβεια.
2. Cross-validation: Η εφαρμογή τεχνικών διασταυρούμενης επικύρωσης θα βελτιώσει την αξιοπιστία των αποτελεσμάτων.

Βιβλιογραφία

Abdulkadirov, R. (2023). Survey of Optimization Algorithms in Modern Neural Networks. *Mathematics*.

Kajornrit, J. (2015). A comparative study of optimization methods for improving artificial neural network performance. *International Conference on Information Technology and Electrical Engineering (ICITEE)*.

Soydaner, D. (2020). A Comparison of Optimization Algorithms for Deep Learning. *International Journal of Pattern Recognition and Artificial Intelligence*.

Αβδελιώδης, Ι. (2024). Συγκριτική ανάλυση αλγορίθμων μηχανικής μάθησης και βαθιάς μάθησης σε προβλήματα εκφράσεων γονιδίων.

Λιαντζάκης, Κ. (2021). Συγκριτική μελέτη αλγορίθμων βελτιστοποίησης σε python για την εύρεση αρχιτεκτονικών νευρωνικών δικτύων. Θεσσαλονίκη.

Μαζαράκης, Λ. (2023). Ανάλυση και σύγκριση αλγορίθμων συσσωμάτωσης για βελτιστοποίηση τεχνικών ομοσπονδιακής μάθησης σε cross-silo σενάρια.