ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ
ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Βαθιά Μάθηση με Βελτιστοποίηση Υπερ-παραμέτρων με την Μέθοδο Σμήνους Σωματιδίων για Πρόβλεψη Χρήσης Πόρων σε Υπολογιστικές Υποδομές των Άκρων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Τίτα Παγουλάτου

Επιβλέπουσα : Θεοδώρα Βαρβαρίγου
Καθηγήτρια Ε.Μ.Π.

Αθήνα, Νοέμβριος 2020

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ
ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Βαθιά Μάθηση με Βελτιστοποίηση Υπερ-παραμέτρων με την Μέθοδο Σμήνους Σωματιδίων για Πρόβλεψη Χρήσης Πόρων σε Υπολογιστικές Υποδομές των Άκρων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Τίτα Παγουλάτου

Επιβλέπουσα : Θεοδώρα Βαρβαρίγου
Καθηγήτρια Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 18η Νοεμβρίου 2020.

.....	.....	.....
Θ. Βαρβαρίγου	Ε. Βαρβαρίγος	Σ. Παπαβασιλείου
Καθ. Ε.Μ.Π	Καθ. Ε.Μ.Π.	Καθ. Ε.Μ.Π.

Αθήνα, Νοέμβριος 2020

.....
Τίτα Παγουλάτου

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Τα τελευταία χρόνια, πολλά συστήματα βασίζονται σε τεχνικές και αλγόριθμους βαθιάς μάθησης και έχουν γίνει ιδιαιτέρως δημοφιλή τόσο στην επιστημονική κοινότητα όσο και σε πολλούς κλάδους της βιομηχανίας, λόγω των εξαιρετικών επιδόσεων των μεθόδων αυτών σε πληθώρα προβλημάτων μηχανικής μάθησης. Ένας μεταευρετικός αλγόριθμος που μπορεί να εφαρμοστεί σε μοντέλα βαθιάς μάθησης είναι ο αλγόριθμος βελτιστοποίησης σμήνους σωματιδίων (Particle Swarm Optimization, PSO) ο οποίος περιλαμβάνει ένα πληθυσμό αυτόνομων "πρακτόρων" που αλληλεπιδρούν μεταξύ τους αλλά και με το περιβάλλον τους με στόχο την εύρεση του ολικού ελάχιστου.

Η παρούσα εργασία διερευνά την πρόβλεψη της χρήσης πόρων σε υπολογιστικές δομές των άκρων με χρήση του αλγόριθμου βελτιστοποίησης σμήνους σωματιδίων που έχει σκοπό την αναζήτηση και εύρεση των βέλτιστων υπερπαραμέτρων που επιλύουν το πρόβλημα αυτό. Στο πλαίσιο αυτό, παρουσιάζονται και αναλύονται τα βασικά δομικά στοιχεία για την οικοδόμηση δικτύων βαθιάς μάθησης. Στη συνέχεια, για την εύρεση των βέλτιστων μη αριθμητικών υπερπαραμέτρων χρησιμοποιούμε Μπεϋζιανό Στατιστικό Μοντέλο. Τα δεδομένα που χρησιμοποιήσαμε αφορούν τη χρήση πόρων μηχανημάτων ακμής Raspberry Pi.

Λέξεις Κλειδιά

Μηχανική Μάθηση, Βαθιά Μάθηση, Μέθοδοι Βελτιστοποίησης, Μέθοδος Σμήνους Σωματιδίων, Υπολογιστική των Άκρων

Abstract

In recent years, many systems are based on deep learning techniques and algorithms and have become particularly popular both in the scientific community and in many industries, due to the excellent performance of these methods in a variety of machine learning problems. A metaheuristic algorithm used in deep-learning models is the Particle Swarm Optimization (PSO) algorithm which includes a population of autonomous "agents" that interact with each other and with their environment in order to find the global minimum.

The present work investigates the prediction of the resource usage in edge computational structures using the Particle Swarm Optimization algorithm that aims to search and find the optimal hyperparameters that solve this problem. In this context, the basic structural elements for building deep learning networks are presented and analyzed. Next, we use the Bayesian Statistical Model to find the optimal nominal hyperparameters. The data we used refer to the resource usage of edge devices Raspberry Pi.

Keywords

Machine Learning, Deep Learning, Optimization Methods, Particle Swarm Optimization, Edge Computing

Ευχαριστίες

Φτάνοντας στο τέλος των προπτυχιακών μου σπουδών, θέλω να ευχαριστήσω την κ. Βαρβαρίγου για την ευκαιρία που μου έδωσε να υλοποιήσω ένα πολύ ενδιαφέρον θέμα Διπλωματικής Εργασίας. Ένα τεράστιο ευχαριστώ οφείλω στον επιβλέποντά μου δρ. Γιάννη Βιόλο, διότι χωρίς την πολύτιμη βοήθεια και στήριξή του σε πρακτικό αλλά και ψυχολογικό επίπεδο, δεν θα μπορούσα να είχα ολοκληρώσει την εργασία. Φυσικά, δεν μπορώ να μην ευχαριστήσω το φίλο μου Στέλιο Τσανάκα, για την βοήθεια που μου παρείχε τους τελευταίους μήνες στην εκπόνηση της εργασίας μου. Επιπλέον, θα ήθελα να ευχαριστήσω τους καθηγητές κ. Βαρβαρίγο και κ. Παπαβασιλείου που παρευρέθησαν στην παρουσίαση της παρούσας διπλωματικής εργασίας και συνέθεσαν την τριμελή επιτροπή.

Τέλος, θέλω να ευχαριστήσω την οικογένειά μου και τους φίλους μου που με υποστήριξαν καθ'όλη την διάρκεια των σπουδών μου και με ενθάρρυναν να στοχεύω ψηλά και να ακολουθώ τα όνειρά μου.

Τη διπλωματική μου εργασία θα ήθελα να την αφιερώσω στην αγαπημένη μου γιαγιά Κωνσταντία.

Τίτα Παγουλάτου,
Αθήνα, Νοέμβριος 2020

Περιεχόμενα

Ευρετήριο Σχημάτων

Ευρετήριο Πινάκων

1. Εισαγωγή

Η Μηχανική Μάθηση (Machine Learning - ML) αποτελεί ίσως τον πιο ραγδαία αναπτυσσόμενο τομέα της Τεχνητής Νοημοσύνης (Artificial Intelligence - AI) καθώς τα τελευταία χρόνια, ειδικά μετά την έλευση της Βαθιάς Μάθησης (Deep Learning), έχει προσφέρει πληθώρα μεθόδων με αξιοσημείωτα αποτελέσματα σε όλες σχεδόν τις εφαρμογές που απαιτούν ευφυΐα.

Τι είναι όμως η τεχνητή νοημοσύνη και η μηχανική μάθηση; Οι περισσότεροι άνθρωποι που δεν είναι σχετικοί με τον τομέα αυτό, φαντάζονται ρομπότ με εξελιγμένες ιδιότητες. Τέτοιες σκέψεις βέβαια, δεν απέχουν και πολύ από την πραγματικότητα, αφού χάρη στις επιστήμες αυτές ήδη από το 2000 το ρομπότ Nomad εξερευνεί απομακρυσμένες περιοχές στην Ανταρκτική αναζητώντας δείγματα μετεωριτών, ενώ το 2009 η Google δημιουργεί το πρώτο αυτο-οδηγούμενο αυτοκίνητο. Ορίζονται, λοιπόν, η τεχνητή νοημοσύνη και η μηχανική μάθηση με τους εξής ορισμούς:

- Ο όρος τεχνητή νοημοσύνη αναφέρεται στην επιστήμη εκείνη η οποία ασχολείται με τη σχεδίαση και την υλοποίηση υπολογιστικών συστημάτων που αναπαράγουν στοιχεία της ανθρώπινης συμπεριφοράς και τα οποία υπονοούν έστω και στοιχειώδη ευφυΐα: μάθηση, προσαρμοστικότητα, εξαγωγή συμπερασμάτων, κατανόηση από συμφραζόμενα, επίλυση προβλημάτων κλπ. Ο Τζον Μακάρθι χαρακτήρισε την τεχνητή νοημοσύνη ως την «επιστήμη και μεθοδολογία της δημιουργίας νοημόνων μηχανών».
- Ο όρος μηχανική μάθηση μπορεί να οριστεί ευρέως ως υπολογιστικές μέθοδοι οι οποίες χρησιμοποιούν εμπειρία για να βελτιώσουν την απόδοση ή να κάνουν ακριβείς προβλέψεις.

Η μηχανική μάθηση αντιμετωπίζει ένα πολύ ευρύ σύνολο πρακτικών εφαρμογών. Στην παρούσα διπλωματική εργασία αποφασίσαμε να επιλύσουμε ένα πρόβλημα μηχανικής μάθησης με τη μέθοδο της παλινδρόμησης σε ένα σύνολο δεδομένων από συσκευές ακμής. Στο πρόβλημα αυτό προτείναμε και υλοποιήσαμε ένα υβριδικό μοντέλο βελτιστοποίησης που χρησιμοποιεί δύο αλγορίθμους βελτιστοποίησης: τον αλγόριθμο βελτιστοποίησης σωματιδίων σμήνους και το μπεϋζιανό αλγόριθμο βελτιστοποίησης με γκαουσιανή διαδικασία.

Δομή Διπλωματικής Εργασίας

Στο 1ο Κεφάλαιο, γίνεται εισαγωγή στην μηχανική μάθηση και επεξηγείται η δομή της διπλωματική εργασίας. Στο 2ο Κεφάλαιο, αναλύονται οι βασικές έννοιες της βαθιάς μηχανικής μάθησης. Στο 3ο Κεφάλαιο, αναλύονται τα βασικά στοιχεία για την επίλυση προβλημάτων χρονοσειρών στην βαθιά μάθηση. Στο 4ο Κεφάλαιο, αναλύονται ο αλγόριθμος βελτιστοποίησης σωματιδίων σμήνους και το μπεϋζιανό στατιστικό μοντέλο. Στο 5ο Κεφάλαιο, εξηγείται η έννοια της υπολογιστικής των άκρων και παρατίθενται τα πλεονεκτήματα της τεχνικής αυτής. Στο 6ο Κεφάλαιο παρουσιάζονται το προτεινόμενο μοντέλο και τα πειραματικά αποτελέσματα της διπλωματικής εργασίας τα οποία συγκρίνονται με συγγενείς εργασίες και τέλος παρουσιάζονται τα συμπεράσματα.

2. Ο επιστημονικός κλάδος της βαθιάς μάθησης

Στην ενότητα αυτή ορίζεται η επιστήμη της βαθιάς μάθησης. Στη συνέχεια γίνεται επεξήγηση και ανάλυση των επιμέρους βασικών στοιχείων που την απαρτίζουν.

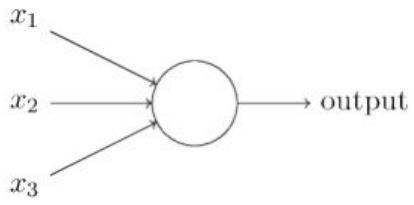
2.1 Εισαγωγή στην βαθιά μάθηση

2.1.1 Βασικές Έννοιες

Η μηχανική μάθηση (Machine Learning - ML) είναι ένα πεδίο της επιστήμης των υπολογιστών που χρησιμοποιεί τεχνικές στατιστικής ώστε να δώσει σε υπολογιστικά συστήματα την δυνατότητα να «μάθουν» από δεδομένα χωρίς να χρειαστεί να ακολουθήσουν κάποιον ντετερμινιστικό αλγόριθμο. Βαθιά Μηχανική Μάθηση (Deep Learning - DL) είναι μία υποκατηγορία της μηχανικής μάθησης. Για να εξηγηθεί όμως, πιο αναλυτικά η βαθιά μάθηση είναι αναγκαίο να ορίσει κανείς τί είναι τα νευρωνικά δίκτυα. Τεχνητό νευρωνικό δίκτυο (Artificial Neural Network - ANN) είναι εκείνο το δίκτυο που περιέχει πολλά πυκνά διασυνδεδεμένα στοιχεία που ονομάζονται νευρώνες ή κόμβοι, τα οποία δεν είναι τίποτα περισσότερο από υπολογιστικά στοιχεία μη γραμμικής φύσης. Το απλούστερο νευρωνικό δίκτυο είναι το single layer perceptron που είναι ένας τύπος τεχνητού νευρώνα που μπορεί να αποφασίσει αν μία είσοδος ανήκει σε μία από τις δύο πιθανές κλάσεις [1]. Ορίζεται, επίσης, επίπεδο ως το γενικό όρο που χαρακτηρίζει μία «συλλογή» νευρώνων που λειτουργούν μαζί - ταυτόχρονα και βρίσκονται στο ίδιο βάθος εντός ενός νευρωνικού δικτύου.

2.1.1.1 Perceptron

Τα perceptrons αναπτύχθηκαν στη δεκαετία του 1950 και 1960 από τον επιστήμονα Frank Rosenblatt, εμπνευσμένο από προηγούμενες εργασίες των Warren McCulloch και Walter Pitts. Πώς λειτουργούν λοιπόν, τα perceptrons; Ένα perceptron παίρνει πολλές δυαδικές εισόδους, $x_1, x_2, \ldots$ και παράγει μία μόνο δυαδική έξοδο [2]:



Εικόνα 2.1 Ένα perceptron

Στο παράδειγμα που παρουσιάζεται στην Εικόνα 2.1 το perceptron έχει τρεις εισόδους, x_1, x_2, x_3. Σε γενικές γραμμές θα μπορούσε να έχει περισσότερες ή λιγότερες εισόδους. Ο Rosenblatt πρότεινε έναν απλό κανόνα για τον υπολογισμό της εξόδου. Εισήγαγε *βάρη,* $w_1, w_2, \ldots$, που είναι πραγματικοί αριθμοί και εκφράζουν τη σημασία των αντίστοιχων εισόδων στην έξοδο. Η έξοδος του νευρώνα, 0 ή 1, καθορίζεται από το αν το σταθμισμένο άθροισμα $\sum_j w_j x_j$ είναι μικρότερο ή μεγαλύτερο από κάποια τιμή κατωφλίου (threshold). Ακριβώς όπως τα βάρη, το κατώφλι είναι ένας πραγματικός αριθμός και αποτελεί παράμετρο του νευρώνα. Πιο συγκεκριμένα, με αλγεβρικούς όρους:

$$output = \begin{cases} 0, & if \sum_j w_j x_j \leq threshold \\ 1, & if \sum_j w_j x_j > threshold \end{cases}$$

Στα όρια $\sum_j w_j x_j$ > threshold και $\sum_j w_j x_j \leq$ threshold μπορούν να γίνουν δύο συμβολικές αλλαγές για να απλοποιηθεί η εξίσωση. Η πρώτη αλλαγή είναι να γραφεί το $\sum_j w_j x_j$ ως εσωτερικό γινόμενο, $w \cdot x = \sum_j w_j x_j$, όπου τα w και x είναι διανύσματα βάρους και εισόδου, αντίστοιχα. Η δεύτερη αλλαγή είναι η μετακίνηση του κατωφλίου στην άλλη πλευρά της ανισότητας και η αντικατάστασή του με αυτό που είναι γνωστό ως bias, b≡ −threshold. Ως εκ τούτου, ο κανόνας perceptron μπορεί να ξαναγραφεί:

$$output = \begin{cases} 0, & if\ \mathrm{w}\cdot x + \mathrm{b} \leq 0 \\ 1, & if\ \mathrm{w}\cdot x + \mathrm{b} > 0 \end{cases}$$

Μεταβάλλοντας τα βάρη και το κατώφλι, μπορούν να παραχθούν πολλά διαφορετικά μοντέλα λήψης αποφάσεων.

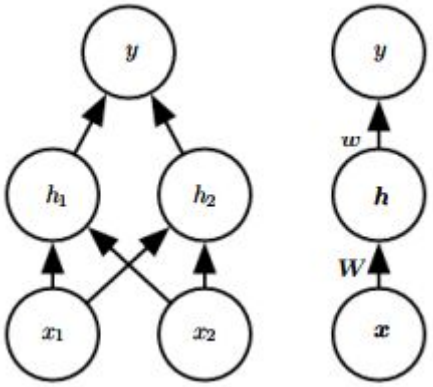
Ωστόσο, τα perceptrons δεν αποτελούν ένα πλήρες μοντέλο ανθρώπινης λήψης αποφάσεων, έχουν αρκετά απλή δομή και δεν χρησιμοποιούνται ευρέως. Παραλλαγές των perceptrons που χρησιμοποιούνται διαφέρουν ως προς τον τύπο εισόδου/εξόδου π.χ. δεν αποδέχονται μόνο 0 ή 1 αλλά χρησιμοποιούν και δεκαδικούς αριθμούς ανάμεσα στο 0 και στο 1. Επίσης, αντί για τη βηματική συνάρτηση που χρησιμοποιήθηκε μέχρι τώρα και όριζε την έξοδο σε 0 ή 1, η έξοδος μπορεί να ορίζεται από άλλες συναρτήσεις και έτσι η συνάρτηση ενεργοποίησης του νευρώνα να είναι π.χ. η σιγμοειδής συνάρτηση $\frac{1}{1+exp(-\sum_j w_j x_j - b)}$.

2.1.1.2 Αρχιτεκτονικές Νευρωνικών Δικτύων

Τα νευρωνικά δίκτυα διακρίνονται με βάση την αρχιτεκτονική τους, τα χαρακτηριστικά των κόμβων τους (nodes) και τους κανόνες εκμάθησης που ακολουθούν. Ο βασικός διαχωρισμός που μπορεί να γίνει στα νευρωνικά δίκτυα είναι σε δίκτυα εμπρόσθιας τροφοδότησης (Feedforward Neural Networks), αναδρομικά νευρωνικά δίκτυα (Recurrent Neural Networks - RNN), καθώς και σε υβριδικά αυτών των δυο. Άλλες δημοφιλείς τοπολογίες των νευρωνικών δικτύων είναι τα δίκτυα πλέγματος (lattice networks), τα πολυεπίπεδα δίκτυα εμπρόσθιας τροφοδότησης με πλευρικές διασυνδέσεις (layered feedforward networks with lateral connections) και τα κυψελοειδή δίκτυα (cellular networks).

2.1.1.2.1 Δίκτυα Εμπρόσθιας Τροφοδότησης

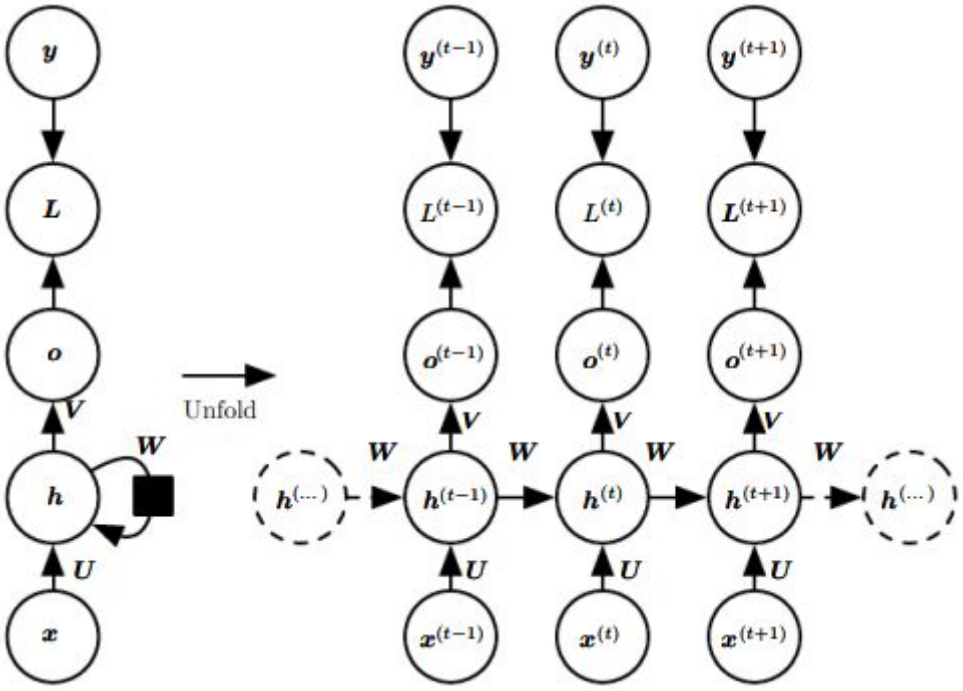
Τα δίκτυα εμπρόσθιας τροφοδότησης (Feedforward Neural Networks) αποτελούνται από ένα ή περισσότερα επίπεδα και οι συνδέσεις μεταξύ των νευρώνων είναι προς μία μόνο κατεύθυνση. Τα επίπεδα αποτελούνται από αυτό της εισόδου, τα εσωτερικά ή αλλιώς κρυφά επίπεδα και το επίπεδο εξόδου. Σε αυτού του είδους τα δίκτυα δεν υπάρχει καμία σύνδεση μεταξύ των νευρώνων που βρίσκονται στο ίδιο επίπεδο καθώς και καμιά ανάδραση μεταξύ των επίπεδων. Σε ένα πλήρως διασυνδεδεμένο δίκτυο εμπρόσθιας τροφοδότησης, όπως αυτό του Σχήματος 2.2, ο κάθε κόμβος ενός επιπέδου συνδέεται με όλους τους κόμβους στο επόμενο επίπεδο.



Εικόνα 2.2 Ένα παράδειγμα δικτύου εμπρόσθιας τροφοδότησης, σχεδιασμένο σε δύο διαφορετικά στυλ.(Αριστερά) Σχεδιάζεται κάθε μονάδα ως κόμβος στο γράφημα. Αυτό το στυλ είναι πολύ σαφές και ξεκάθαρο, αλλά για δίκτυα μεγαλύτερα από αυτό το παράδειγμα μπορεί να καταναλωθεί πολύς χώρος. (Δεξιά) Σχεδιάζεται ένας κόμβος στο γράφημα για κάθε ολόκληρο διάνυσμα που αντιπροσωπεύει τις ενεργοποιήσεις ενός επιπέδου. Αυτό το στυλ είναι πολύ πιο συμπαγές. Δίπλα από τις ακμές σε αυτό το γράφημα γράφεται το όνομα των παραμέτρων που περιγράφουν τη σχέση μεταξύ δύο επιπέδων. Εδώ, υποδεικνύεται ότι ένας πίνακας W περιγράφει την αντιστοίχιση από x έως h και ένας διάνυσμα w περιγράφει την αντιστοίχιση από h έως y.

2.1.1.2.2 Αναδρομικά νευρωνικά δίκτυα

Στα αναδρομικά νευρωνικά δίκτυα (Recurrent Neural Networks - RNN) υπάρχει τουλάχιστον μια ανάδραση, είτε ανάμεσα σε κόμβους του ίδιου επιπέδου, είτε σε κάποιο κόμβο με κόμβο ενός προηγούμενου επιπέδου (Σχήμα 2.3). Αυτά τα δίκτυα θα μελετηθούν διεξοδικότερα σε επόμενη ενότητα.



Εικόνα 2.3 Αναδρομικό νευρωνικό δίκτυο. Το RNN εισάγει τα δεδομένα στους κρυφούς νευρώνες που παραμετροποιούνται από έναν πίνακα μήτρας U, στις κρυφές προς κρυφές αναδρομικές συνδέσεις που παραμετροποιούνται από μια μήτρα βάρους W και στις συνδέσεις κρυφού επιπέδου προς έξοδο που παραμετροποιούνται από μια μήτρα βάρους V. (Αριστερά) Το RNN και η απώλειά (loss) του αντλούνται με αναδρομικές συνδέσεις. (Δεξιά) Το ίδιο αναδρομικό δίκτυο ως ένα υπολογιστικό γράφημα ξεδιπλωμένο στο χρόνο, όπου κάθε κόμβος συνδέεται τώρα με ένα συγκεκριμένο στιγμιότυπο χρόνου.

2.1.1.3 Βαθιά νευρωνικά δίκτυα

Έχοντας, λοιπόν, οριστεί τί είναι τα επίπεδα και πώς λειτουργούν τα νευρωνικά δίκτυα, ορίζονται βαθιά νευρωνικά δίκτυα (deep neural networks) εκείνα τα δίκτυα με δομή πολλών επιπέδων - δύο ή περισσότερων κρυφών επιπέδων. Αντίστοιχα, η προσέγγιση της Τεχνητής Νοημοσύνης που ασχολείται με βαθιά νευρωνικά δίκτυα ονομάζεται Βαθιά Μάθηση (Deep Learning - DL).

Η σύγχρονη βαθιά μάθηση παρέχει ένα ισχυρό πλαίσιο επιβλεπόμενης (supervised) μάθησης. Προσθέτοντας περισσότερα επίπεδα (layers) και περισσότερα στοιχεία σε κάθε layer, ένα βαθύ δίκτυο (deep network) μπορεί να αναπαραστήσει όλο και πιο πολύπλοκες συναρτήσεις. Τα περισσότερα προβλήματα αντιστοίχισης ενός διανύσματος εισόδου σε ένα διάνυσμα εξόδου, τα οποία είναι εύκολα για τον άνθρωπο, μπορούν να επιλυθούν με βαθιά μάθηση, εφόσον εφαρμοστούν αρκετά μεγάλα μοντέλα και υπάρχουν στη διάθεση του προγραμματιστή αρκετά δείγματα εκπαίδευσης.

Ένα βαθύ νευρωνικό δίκτυο για να επιτελέσει το σκοπό του πρέπει να ορίσει διάφορες παραμέτρους του όπως είναι ο αριθμός των επιπέδων, ο αριθμός των νευρώνων σε κάθε επίπεδο, συναρτήσεις ενεργοποίησης, βελτιστοποιητές, παράμετρο απόρριψης κ.ά. Τα κύρια χαρακτηριστικά αυτών των νευρωνικών δικτύων αναλύονται στη συνέχεια του κεφαλαίου.

2.1.2 Τύποι εργασιών Μηχανικής Μάθησης

Η μηχανική μάθηση είναι ως επί το πλείστον ενδιαφέρουσα λόγω των εργασιών που καλείται ο προγραμματιστής να επιτύχει με αυτήν. Από μηχανικής άποψης, η μηχανική μάθηση τού επιτρέπει να αντιμετωπίζει εργασίες που είναι πολύ δύσκολο να επιλυθούν με σταθερά προγράμματα γραμμένα και σχεδιασμένα από ανθρώπους. Από επιστημονική και φιλοσοφική άποψη, η μηχανική μάθηση είναι ενδιαφέρουσα επειδή τού επιτρέπει να κατανοήσει τις αρχές που διέπουν την ευφυή συμπεριφορά. Ευφυή συμπεριφορά ορίζεται ως η ικανότητα εκτέλεσης ορισμένων εργασιών.

Σε αυτόν τον σχετικά επίσημο ορισμό της λέξης «εργασία», η ίδια η διαδικασία της μάθησης δεν είναι η εργασία. Η μάθηση είναι το μέσο για την επίτευξη της ικανότητας εκτέλεσης της εργασίας. Για παράδειγμα, αν προγραμματιστεί ένα ρομπότ να μπορεί να περπατάει, τότε το περπάτημα είναι η εργασία του.

Πολλά είδη εργασιών μπορούν να επιλυθούν με τη μηχανική μάθηση. Μερικές από τις πιο κοινές εργασίες μηχανικής μάθησης περιλαμβάνουν τα ακόλουθα:

- ➢ Ταξινόμηση (Classification): Σε αυτόν τον τύπο εργασιών, ζητείται από το πρόγραμμα του υπολογιστή να καθορίσει σε ποιες από τις k κατηγορίες ανήκει κάποια είσοδος.
- ➢ Μεταγραφή (Transcription): Σε αυτόν τον τύπο εργασίας, ζητείται από το σύστημα μηχανικής μάθησης να παρατηρήσει μία σχετικά μη δομημένη αναπαράσταση κάποιου είδους δεδομένων και να τα μεταγράψει σε διακριτή, μορφή κειμένου.
- ➢ Μετάφραση (Translation): Σε μια εργασία μετάφρασης, η είσοδος αποτελείται ήδη από μια ακολουθία συμβόλων σε κάποια γλώσσα και το πρόγραμμα του υπολογιστή πρέπει να το μετατρέψει σε ακολουθία συμβόλων σε άλλη γλώσσα.
- ➢ Παλινδρόμηση (Regression): Σε αυτόν τον τύπο εργασίας, ζητείται από το πρόγραμμα του υπολογιστή να προβλέψει μια αριθμητική τιμή δεδομένης της εισόδου. Για την επίλυση αυτής της εργασίας, ζητείται από τον αλγόριθμο εκμάθησης να εξάγει μια συνάρτηση $f : R^n \rightarrow R$. Αυτός ο τύπος εργασίας είναι παρόμοιος με την ταξινόμηση, με διαφορά τη μορφή της εξόδου, η οποία είναι διαφορετική. Ένα παράδειγμα μιας εργασίας παλινδρόμησης είναι η πρόβλεψη του αναμενόμενου ποσού που θα απαιτήσει ένας ασφαλισμένος (χρησιμοποιείται για τον καθορισμό ασφάλιστρων) ή η πρόβλεψη των μελλοντικών τιμών των ασφάλιστρων.

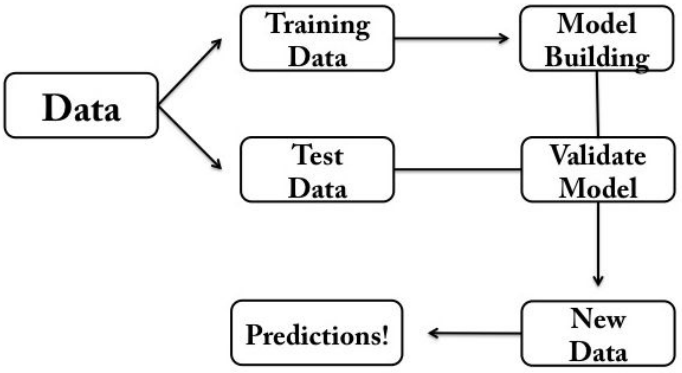
2.1.3 Αλγόριθμος Βαθιάς Μάθησης

Πώς λειτουργεί όμως η βαθιά μάθηση; Μια διαδικασία μηχανικής μάθησης αποτελείται, σε γενικές γραμμές, από τα παρακάτω βήματα:

1. Συλλογή δεδομένων, που θα χρησιμοποιηθούν για την εκπαίδευση και την δοκιμή του μοντέλου.

2. Προετοιμασία δεδομένων. Εν συνεχεία, τα δεδομένα θα πρέπει να «καθαριστούν», δηλαδή να απομακρυνθούν δεδομένα κακής ποιότητας, να ταξινομηθούν σε μια επιθυμητή σειρά και εν τέλει να τροποποιηθούν, ώστε να έχουν μια ενιαία μορφή. Τα δύο αυτά πρώτα βήματα είναι συνήθως χρονοβόρα και πολύ μεγάλης σημασίας, καθώς είναι η βάση πάνω στην οποία αργότερα στήνονται τα μοντέλα πρόβλεψης.
3. Επιλογή μοντέλου μηχανικής μάθησης. Σε αυτό το στάδιο γίνεται η δοκιμή και τέλος η επιλογή των μοντέλων που θα χρησιμοποιηθούν. Τα μοντέλα μηχανικής μάθησης είναι πολλά και δεν υπάρχει σαφής κατηγοριοποίηση για το ποιο μοντέλο ταιριάζει σε ποια εργασία. Ωστόσο υπάρχουν αποδεδειγμένες ομάδες μοντέλων που προτιμώνται για συγκεκριμένες διεργασίες όπως τα συνελικτικά δίκτυα (convolutional neural networks - CNNs) ταιριάζουν σε προβλήματα με τοπολογία πλέγματος. Στην διπλωματική αυτή θα χρησιμοποιηθούν τα αναδρομικά νευρωνικά δίκτυα LSTM και τα δίκτυα εμπρόσθιας τροφοδότησης CNN τα οποία και θα αναλυθούν εκτενέστερα στο 3ο Κεφάλαιο.
4. Εκπαίδευση (training). Το στάδιο αυτό μπορεί από πολλούς να θεωρηθεί η ουσία της μηχανικής μάθησης. Σε αυτό το σημείο ένα κομμάτι (περίπου το 60-80%) από τα καλώς οργανωμένα δεδομένα που έχουν συλλεχθεί στα προηγούμενα στάδια περνούν από το μοντέλο μηχανικής μάθησης με σκοπό αυτό να «εκπαιδευτεί» και να μάθει να αναλύει τέτοιου τύπου δεδομένα.
5. Εκτίμηση (validation). Σε αυτό το σημείο ένα μικρότερο κομμάτι από τα δεδομένα (10-20%) χρησιμοποιείται με σκοπό να δοκιμαστεί αν το εκπαιδευμένο πλέον μοντέλο μπορεί να ανταποκριθεί σε νέες εισόδους και έτσι να γίνουν οι τελικοί υπολογισμοί και οι διορθώσεις ανάλογα με την απόδοση του.
6. Δοκιμή (testing). Σε αυτό το επίπεδο, το τελευταίο κομμάτι των δεδομένων θα χρησιμοποιηθεί ώστε να μετρηθεί η τελική απόδοση του μοντέλου.
7. Βελτίωση μοντέλου. Τις περισσότερες φορές μετά την δοκιμή θα χρειαστεί να γίνουν αλλαγές τόσο στην αρχιτεκτονική του μοντέλου όσο και στις υπερπαραμέτρους μέχρι να υπάρχει το επιθυμητό αποτέλεσμα. Σε αυτή την περίπτωση η διαδικασία επιστρέφει στα αρχικά βήματα και δοκιμάζεται ξανά το καινούριο μοντέλο.

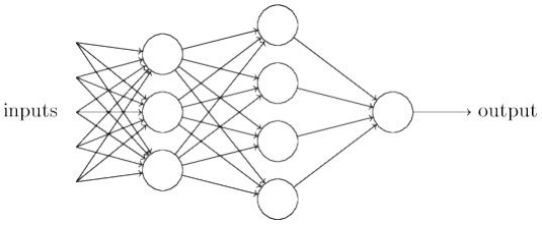
Παρακάτω, η εικόνα 2.4 παρουσιάζει συνοπτικά τη διαδικασία της μηχανικής μάθησης.



Εικόνα 2.4 Πώς λειτουργεί η Μηχανική Μάθηση

2.2 Επίπεδα

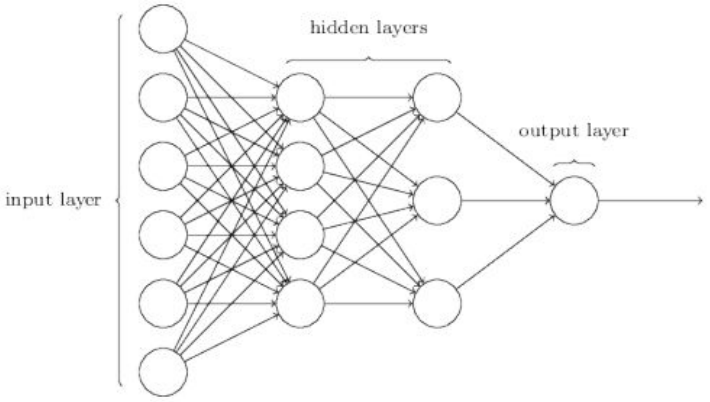
Επίπεδο ή στρώμα (layer), όπως παρουσιάστηκε προηγουμένως, είναι μία συστάδα νευρώνων που λειτουργούν μαζί και βρίσκονται στο ίδιο βάθος εντός ενός νευρωνικού δικτύου. Ένα σύνθετο νευρωνικό δίκτυο αποτελείται από πολλά επίπεδα - στρώματα νευρώνων, καθένας από τους οποίους δημιουργεί συνδέσεις ή, πιο σωστά, συνάψεις με όλους τους επόμενούς του:



Εικόνα 2.5 Ένα απλό νευρωνικό δίκτυο

Στο δίκτυο της Εικόνας 2.5, η πρώτη στήλη του δικτύου λαμβάνει τρεις πολύ απλές αποφάσεις, σταθμίζοντας τα δεδομένα που του δίνονται. Τι γίνεται με τους νευρώνες στο δεύτερο επίπεδο; Καθένα από αυτούς τους νευρώνες παίρνει μια απόφαση ζυγίζοντας τα αποτελέσματα από το πρώτο επίπεδο λήψης αποφάσεων. Με αυτόν τον τρόπο ένας νευρώνας στο δεύτερο στρώμα μπορεί να πάρει μία απόφαση σε πιο περίπλοκο και πιο αφηρημένο επίπεδο από ό,τι οι νευρώνες στο πρώτο επίπεδο. Και ακόμη πιο περίπλοκες αποφάσεις μπορούν να ληφθούν από ένα νευρώνα στο τρίτο. Με αυτόν τον τρόπο, ένα δίκτυο πολλαπλών επιπέδων μπορεί να συμμετάσχει στη λήψη εξελιγμένων αποφάσεων.

Το αριστερότερο στρώμα στο δίκτυο της Εικόνας 2.5 ορίζεται επίπεδο ή στρώμα εισόδου και οι νευρώνες εντός αυτού του στρώματος ονομάζονται νευρώνες εισόδου. Το δεξιότερο στρώμα ορίζεται ως το επίπεδο ή στρώμα εξόδου και περιέχει τους νευρώνες εξόδου ή, σε αυτή την περίπτωση, έναν μόνο νευρώνα εξόδου. Το μεσαίο στρώμα ονομάζεται κρυφό επίπεδο (hidden layer), καθώς οι νευρώνες σε αυτό το στρώμα δεν είναι ούτε είσοδοι ούτε έξοδοι. Το παραπάνω δίκτυο έχει μόνο ένα κρυφό επίπεδο, αλλά ορισμένα δίκτυα έχουν πολλά παραπάνω. Για παράδειγμα, το ακόλουθο δίκτυο τεσσάρων επιπέδων έχει δύο κρυφά επίπεδα:



Εικόνα 2.6 Νευρωνικό δίκτυο με 2 κρυφά επίπεδα

2.2.1 Πολυστρωματικά Νευρωνικά Δίκτυα

Δίκτυα πολλαπλών επιπέδων, όπως της εικόνας 2.6, ονομάζονται πολυεπίπεδα perceptron (Multi-Layer Perceptron - MLP) και είναι τα βασικά βαθιά δίκτυα. Είναι επίσης γνωστά ως βαθιά δίκτυα εμπρόσθιας τροφοδότησης (feedforward deep networks). Αυτά τα δίκτυα πρόκειται ουσιαστικά για παραμετρικές συναρτήσεις που ορίζονται από τη σύνθεση πολλών παραμετρικών συναρτήσεων [3]. Κάθε μία από αυτές τις συναρτήσεις έχει πολλές εισόδους και πολλαπλές εξόδους. Στην ορολογία των νευρωνικών δικτύων, κάθε υπο-συνάρτηση αναφέρεται ως στρώμα (layer) του δικτύου και κάθε βαθμωτή έξοδο μιας από αυτών των συναρτήσεων ως μονάδα. Παρόλο που κάθε μονάδα εφαρμόζει μια σχετικά απλή αντιστοίχιση ή μετασχηματισμό της εισόδου της, η συνάρτηση ολόκληρου του δικτύου μπορεί να γίνει αρκετά περίπλοκη.

Παρόλο που δεν μπορεί να αναλυθεί κάθε αλγόριθμος βαθιάς μάθησης με όρους μιας μεμονωμένης, ντετερμινιστικής συνάρτησης όπως είναι τα εμπρόσθια δίκτυα τροφοδότησης, όλοι αυτοί οι αλγόριθμοι μοιράζονται την ιδιότητα να περιέχουν πολλά επίπεδα πολλών μονάδων. Μπορεί

κανείς να σκεφτεί τον αριθμό των μονάδων σε κάθε επίπεδο ως το πλάτος ενός μοντέλου βαθιάς μάθησης και τον αριθμό των επιπέδων ως το βάθος του. Τα εμπρόσθια δίκτυα τροφοδότησης προσφέρουν ένα εννοιολογικά απλό παράδειγμα ενός αλγορίθμου που αποτυπώνει τα πολλά πλεονεκτήματα που προέρχονται από το γεγονός πως έχουν σημαντικό πλάτος και βάθος. Είναι, επίσης, η βασική τεχνολογία που διέπει τις περισσότερες από τις σύγχρονες εμπορικές εφαρμογές βαθιάς μάθησης σε μεγάλα σύνολα δεδομένων.

2.3 Συναρτήσεις Ενεργοποίησης

Τα βαθιά νευρωνικά δίκτυα έχουν χρησιμοποιηθεί με επιτυχία σε ποικίλους αναπτυσσόμενους τομείς για την επίλυση πραγματικών σύνθετων προβλημάτων με αρχιτεκτονικές βαθιάς μάθησης. Για να μπορέσουν να επιλυθούν με επιτυχία αυτά τα προβλήματα, οι αρχιτεκτονικές βαθιάς μάθησης χρησιμοποιούν συναρτήσεις ενεργοποίησης, για να εκτελέσουν διαφορετικούς υπολογισμούς μεταξύ των κρυφών επιπέδων και των επιπέδων εξόδου [4].

Οι συναρτήσεις ενεργοποίησης (Activation Functions - AF) είναι συναρτήσεις που χρησιμοποιούνται στα νευρωνικά δίκτυα για τον υπολογισμό του σταθμισμένου αθροίσματος εισόδου και bias, το οποίο χρησιμοποιείται για να αποφασιστεί εάν ένας νευρώνας θα πυροδοτηθεί ή όχι. Οι συναρτήσεις ενεργοποίησης επεξεργάζονται τα δεδομένα που παρουσιάζονται μέσω κάποιας επεξεργασίας της κλίσης, συνήθως μέσω του αλγορίθμου κατάβαση κλίσης (gradient descent) και στη συνέχεια παράγουν έξοδο για το νευρωνικό δίκτυο. Οι AF αναφέρονται συχνά και ως συναρτήσεις μεταφοράς (transfer functions) στη βιβλιογραφία.

Όμως, πώς χρησιμοποιούνται οι συναρτήσεις ενεργοποίησης; Το στρώμα εισόδου δέχεται τα δεδομένα για την εκπαίδευση του νευρωνικού δικτύου που διατίθεται σε διάφορες μορφές όπως εικόνες, βίντεο, κείμενα, ομιλία, ήχοι ή αριθμητικά δεδομένα, ενώ τα κρυφά στρώματα αποτελούνται κυρίως από τα συνελικτικά (convolutional layers) και συγκεντρωτικά στρώματα (pooling layers), εκ των οποίων τα συνελικτικά στρώματα ανιχνεύουν από τα προηγούμενα επίπεδα τα τοπικά μοτίβα και χαρακτηριστικά σε δεδομένα, ενώ τα συγκεντρωτικά στρώματα συγχωνεύουν σημασιολογικά παρόμοια χαρακτηριστικά σε ένα. Το στρώμα εξόδου παρουσιάζει τα αποτελέσματα του δικτύου - τα οποία συχνά ελέγχονται από τις συναρτήσεις ενεργοποίησης, πραγματοποιώντας ταξινομήσεις ή προβλέψεις, με σχετικές πιθανότητες.

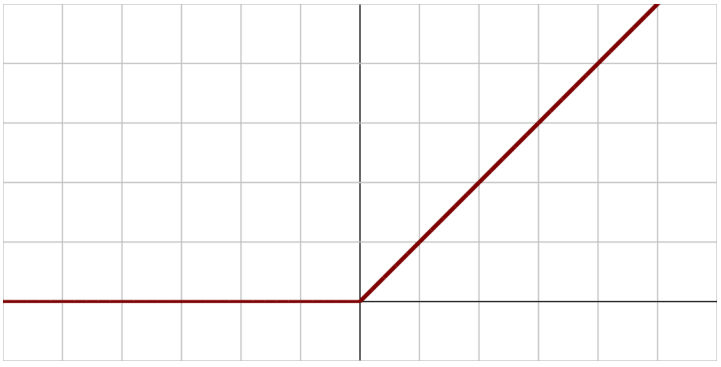
Η θέση μίας συνάρτησης ενεργοποίησης (AF) σε μία δομή δικτύου εξαρτάται από τη λειτουργία του στο δίκτυο, επομένως, όταν η AF τοποθετείται μετά τα κρυφά στρώματα μετατρέπει τις γραμμικές αντιστοιχίσεις σε μη γραμμικές μορφές για διάδοση, ενώ στο στρώμα εξόδου εκτελεί προβλέψεις. Οι βαθιές αρχιτεκτονικές (deep architectures) αποτελούνται από πολλά επίπεδα επεξεργασίας, που περιλαμβάνουν τόσο γραμμικές όσο και μη γραμμικές συναρτήσεις, και εκπαιδεύονται μαζί για την επίλυση μιας δεδομένης εργασίας. Αυτά τα βαθύτερα δίκτυα προσφέρουν καλύτερες επιδόσεις, ακόμα και αν προκύπτουν συνηθισμένα ζητήματα, όπως η εξαφάνιση κλίσης (vanishing gradients) και η εκρηκτική αύξηση κλίσης (exploding gradient), ως αποτέλεσμα των παραγώγων που συνήθως είναι μικρότερο από 1. Με διαδοχικό πολλαπλασιασμό αυτών των παραγώγων, η τιμή γίνεται όλο και μικρότερη και τείνει στο μηδέν και έτσι η κλίση εξαφανίζεται. Κατ' αντιστοιχία, εάν οι τιμές των παραγώγων είναι μεγαλύτερες από 1, ο διαδοχικός πολλαπλασιασμός θα αυξήσει τις τιμές και η κλίση θα τείνει στο άπειρο εκρηγνύοντας έτσι την κλίση. Οι AF, χρησιμοποιώντας διαφορετικές μαθηματικές συναρτήσεις, διατηρούν τις τιμές αυτών των κλίσεων σε συγκεκριμένα όρια. Στη συνέχεια παρουσιάζονται οι πιο γνωστές συναρτήσεις ενεργοποίησης, οι οποίες χρησιμοποιήθηκαν και στα πειράματα της παρούσας διπλωματικής εργασίας. Οι περισσότερες από αυτές συνήθως συνδυάζονται με έναν μετασχηματισμό συγγενών $a = b + Wx$ και εφαρμόζονται στο στοιχείο [4]:

$$h = \varphi(a) \Leftrightarrow h_i = \varphi(a_i) = \varphi(b_i + W_{i,:}x)$$

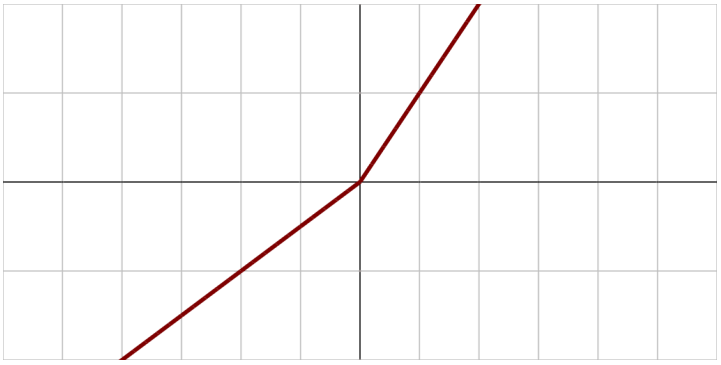
2.3.1 ReLU

Rectifier ή rectified linear unit (ReLU): μετατρέπει την έξοδο του προηγούμενου στρώματος σύμφωνα με τον τύπο $\varphi(a) = max(0, a)$, επίσης γραμμένο και ως $\varphi(a) = (a)^+$. Αυτή είναι μακράν η πιο δημοφιλής συνάρτηση ενεργοποίησης σε μια κρυφή μονάδα στα τρέχοντα δίκτυα εμπρόσθιας

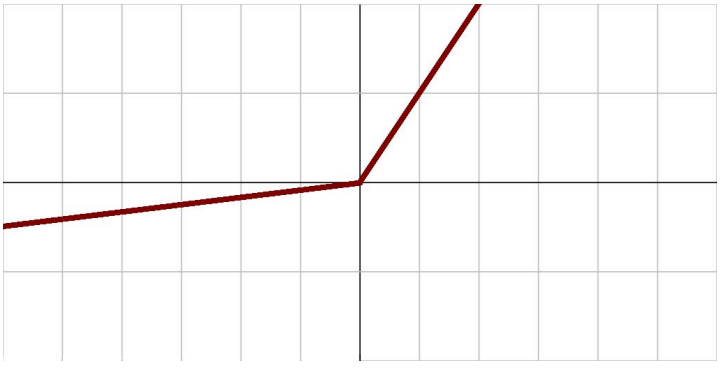
τροφοδότησης. Ορισμένες άλλες αποτελεσματικές παραλλαγές της ReLU βασίζονται στη χρήση μη μηδενικής κλίσης a_i όταν $a < 0$: $h_i = \varphi(a, a_i) = max(0, a) + a_i min(0, a)$, όπου το a_i μπορεί να είναι μια μικρή σταθερά όπως 0,01. Ένα Leaky ReLU διορθώνει το a_i σε μια μικρή τιμή όπως το 0,01, ενώ ένα παραμετρικό ReLU ή PReLU αντιμετωπίζει το a_i ως μια μαθησιακή παράμετρο. Παρακάτω δίνονται οι γραφικές παραστάσεις:



Εικόνα 2.7 ReLU



Εικόνα 2.8 PReLU

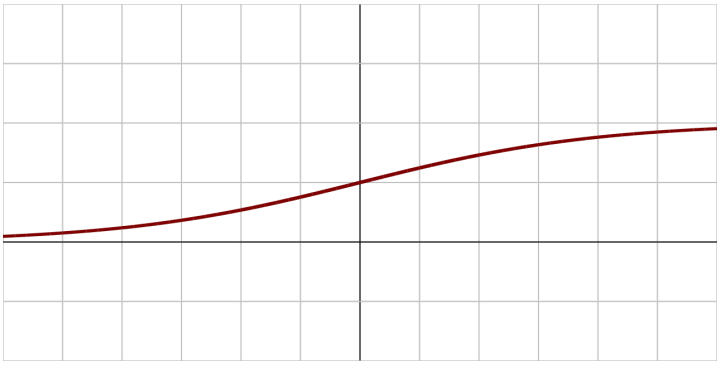


Εικόνα 2.9 Leaky ReLU

2.3.2 Sigmoid

Η Σιγμοειδής ή Logistic ή Sigmoid ή Soft step: μετατρέπει την έξοδο του προηγούμενου στρώματος σύμφωνα με τον τύπο $\varphi(a) = \frac{1}{1+e^{-a}}$ [5]. Η συνάρτηση Sigmoid είναι στην πραγματικότητα μια συνάρτηση $\tanh(a)$ που μεταφέρεται από το εύρος (−1, 1) στο (0, 1). Οι τιμές 0 και 1 λαμβάνονται μόνο για μείον και συν άπειρα, αντίστοιχα. Καθώς οι τιμές εξόδου πλησιάζουν αυτά τα όρια, μειώνεται η παραγωγή αυτής της συνάρτησης [7]. Η Sigmoid χρησιμοποιείται συνήθως για την

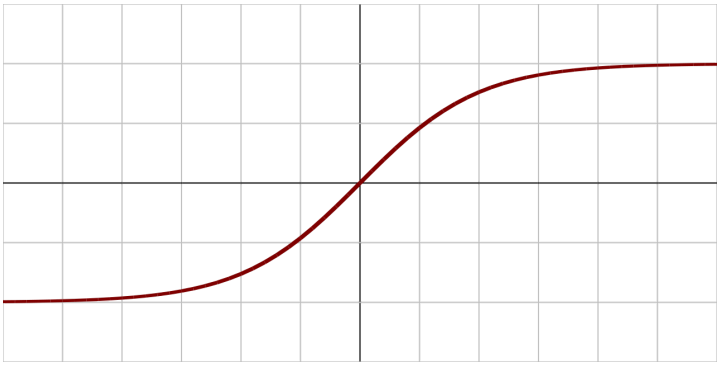
παραγωγή της παραμέτρου φ μιας κατανομής Bernoulli, επειδή το εύρος της ανήκει στο διάστημα (0, 1), το οποίο βρίσκεται εντός του έγκυρου εύρους τιμών της παραμέτρου φ [3].



Εικόνα 2.10 Sigmoid

2.3.3 Tanh

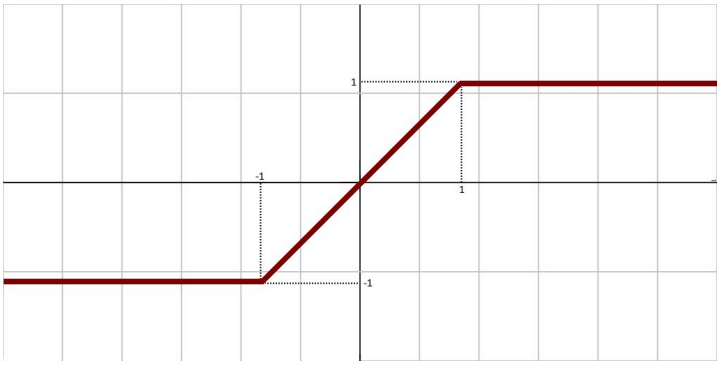
Υπερβολική Εφαπτομένη ή Hyperbolic tangent ή Tanh: μετατρέπει την έξοδο του προηγούμενου στρώματος σύμφωνα με τον τύπο $\varphi(a) = tanh(a)$. Η συνάρτηση TanH είναι μια διπολική εκδοχή της σιγμοειδούς συνάρτησης. Παράγει την βαθμωτή έξοδο σε κλειστό εύρος από -1 έως 1. Οι τιμές εξόδου -1 και 1 λαμβάνονται για μείον και συν άπειρο, αντίστοιχα. Επειδή το διάστημα εξόδου της TanH είναι ευρύτερο, μπορεί να είναι πιο αποτελεσματική στην ταξινόμηση των MLP. Ο Haykin (1994) δήλωσε, επίσης. ότι μια ασύμμετρη συνάρτηση ενεργοποίησης, όπως η Tanh αντί της μη συμμετρικής συνάρτησης Sigmoid, θα μπορούσε να αποδώσει καλύτερα για τα MLP [7].



Εικόνα 2.11 TanH

2.3.4 Hard Tanh

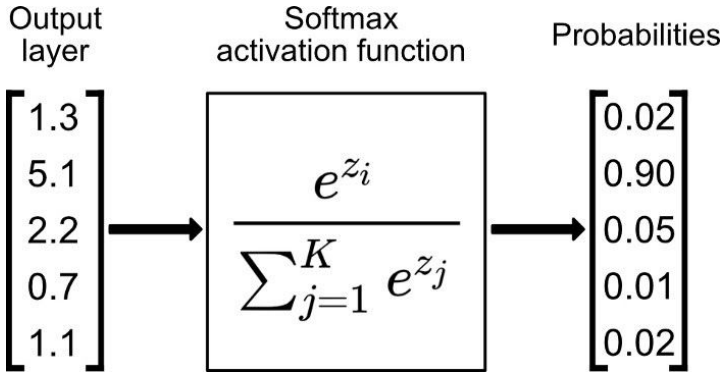
Hard TanH: έχει παρόμοιο σχήμα με την TanH και την ReLU, αλλά σε αντίθεση με την τελευταία, είναι οριοθετημένη, με $\varphi(a) = max(-1, min(1, a))$. Μερικές φορές προτιμάται από τη συνάρτηση TanH, δεδομένου ότι είναι υπολογιστικά φθηνότερη. Ωστόσο, φτάνει σε κορεσμό για μεγέθη a μεγαλύτερα από 1.



Εικόνα 2.12 Hard TanH

2.3.5 Softmax

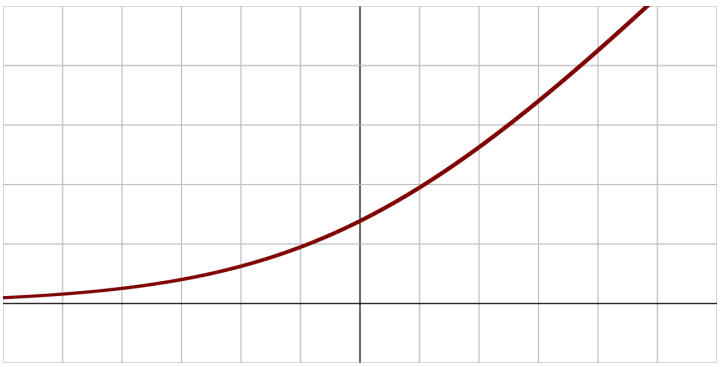
Αυτός είναι ένας μετασχηματισμός από διάνυσμα σε διάνυσμα $\varphi(z) = softmax(z) = \frac{e^{z_i}}{\sum_j e^{z_j}}$ έτσι ώστε $\sum_i \varphi_i(z) = 1$ και $\varphi_i(z) > 0$, με z :διάνυσμα. Για παράδειγμα, η έξοδος της softmax μπορεί να θεωρηθεί ως μια κατανομή πιθανότητας σε ένα πεπερασμένο σύνολο αποτελεσμάτων. Σημειώστε ότι δεν εφαρμόζεται σε επίπεδο στοιχείου αλλά σε ένα σύνολο διανυσμάτων "σκορ". Χρησιμοποιείται ως μη γραμμική έξοδος για την πρόβλεψη διακριτών πιθανοτήτων έναντι των κατηγοριών εξόδου. Παρακάτω δίνεται ένα παράδειγμα για το πώς λειτουργεί η συνάρτηση Softmax:



Εικόνα 2.13 Softmax

2.3.6 Softplus

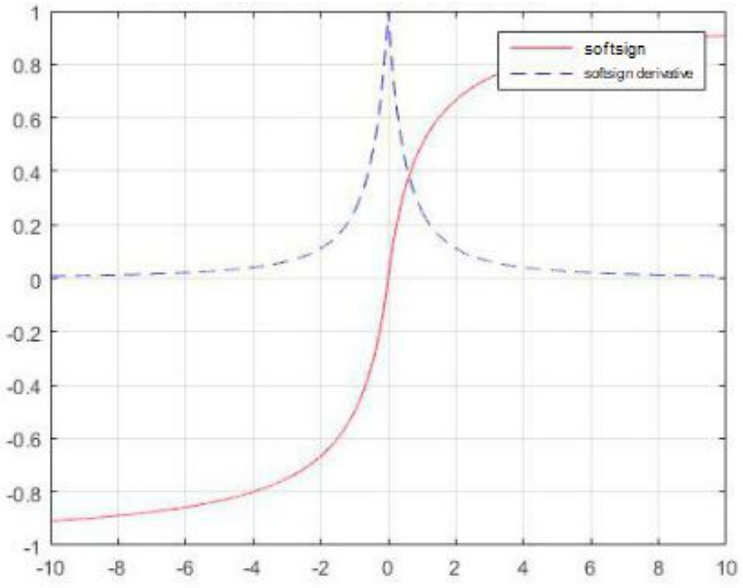
Μετατρέπει την έξοδο του προηγούμενου στρώματος σύμφωνα με τον τύπο $\varphi(a) = \zeta(a) = log(1 + e^a)$. Μπορεί να είναι χρήσιμη για την παραγωγή της παραμέτρου β ή σ μιας κανονικής κατανομής επειδή το εύρος της είναι R^+. Το όνομα της προέρχεται από το γεγονός ότι είναι μια εξομαλυμένη ή «μαλακή» έκδοση του $x^+ = max(0, x)$, δηλαδή της ReLU [3]. Σύμφωνα με το [9], οι Glorot et al. (2011a) σύγκριναν τη softplus και τη ReLU και βρήκαν ότι καλύτερα αποτελέσματα έδινε η δεύτερη, παρά το πολύ παρόμοιο σχήμα και τη μη μηδενική παράγωγο της softplus παντού, σε αντίθεση με την ReLU.



Εικόνα 2.14 Softplus [6]

2.3.7 Softsign

Η συνάρτηση Softsign δίνεται από τον τύπο $\varphi(a) = \frac{a}{|a|+1}$. Η συνάρτηση Softsign είναι μια πολυωνυμική μη γραμμική συνάρτηση ενεργοποίησης. Είναι απλή στον υπολογισμό με ομαλό κορεσμό και μπορεί να μειώσει τον αριθμό των επαναλήψεων, αφού φτάνει εύκολα σε σύγκλιση. Η Softsign συμπιέζει τα δεδομένα στο διάστημα (-1,1), το οποίο είναι παρόμοιο με την υπερβολική εφαπτομένη TanH. Η έξοδος επικεντρώνεται στο 0, όμως επειδή η ασύμπτωτή της είναι πιο ομαλή, και τα δύο σημεία κορεσμού πλησιάζουν αργά στο 0, το εύρος της είναι σχετικά ευρύ, η διαδικασία αρχικοποίησης είναι πιο ισχυρή. Το μεσαίο τμήμα της συνάρτησης Softsign είναι ευρύ, ενώ η περιοχή κοντά στο $x = 0$ λιγότερο. Ο βαθμός μη γραμμικότητας της συνάρτησης είναι υψηλός και είναι εύκολο να οριοθετηθεί το πιο περίπλοκο όριο [10].



Εικόνα 2.15 Softsign

2.4 Βελτιστοποιητές

Σε αυτήν την ενότητα, θα συζητηθεί μία σειρά αλγορίθμων οι οποίοι ονομάζονται Βελτιστοποιητές (Optimizers). Καθένας από αυτούς τους αλγορίθμους επιδιώκει να αντιμετωπίσει την πρόκληση της βελτιστοποίησης των βαθιών μοντέλων κατά την εκπαίδευση των μοντέλων - νευρωνικών δικτύων. Αυτό το επιτυγχάνουν προσαρμόζοντας το ρυθμό μάθησης (learning rate) για κάθε παράμετρο του μοντέλου, έτσι ώστε να ελαχιστοποιηθεί όσο το δυνατόν περισσότερο η συνάρτηση κόστους. Ο ρυθμός μάθησης είναι, δηλαδή, μία υπερπαραμέτρος βελτιστοποίησης που ελέγχει το μέγεθος του βήματος που λαμβάνουν οι παράμετροι προς την κατεύθυνση της κλίσης. Μία

φυσική ερώτηση που προκύπτει όταν κάποιος δημιουργεί ένα μοντέλο βαθιάς μάθησης είναι ποιόν αλγόριθμο θα πρέπει να επιλέξει. Δυστυχώς, προς το παρόν δεν υπάρχει σύγκλιση απόψεων σε αυτό το πρόβλημα. Ο Tom Schaul (2014) παρουσίασε μια πολύτιμη σύγκριση ενός σημαντικού αριθμού αλγορίθμων βελτιστοποίησης (optimizers) σε ένα ευρύ φάσμα μαθησιακών προβλημάτων. Ενώ τα αποτελέσματα υποδηλώνουν ότι η οικογένεια αλγορίθμων που εκπροσωπούνται από τους Optimizers RMSprop και AdaDelta είχε αρκετά ισχυρή απόδοση, δεν έχει εμφανιστεί κανένας καλύτερος αλγόριθμος.

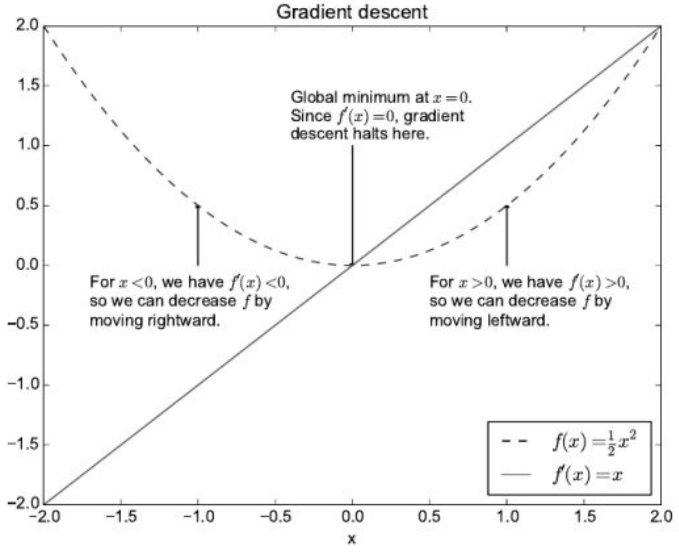
Προς το παρόν, οι πιο δημοφιλείς αλγόριθμοι βελτιστοποίησης που χρησιμοποιούνται ενεργά περιλαμβάνουν SGD, SGD + momentum, RMSprop, RMSprop + momentum, AdaDelta και Adam. Η επιλογή του αλγορίθμου που θα χρησιμοποιηθεί, φαίνεται να εξαρτάται τόσο από την εξοικείωση των χρηστών με τον εκάστοτε αλγόριθμο (για ευκολία συντονισμού υπερπαραμέτρων) όσο και από οποιαδήποτε καθιερωμένη έννοια εξαιρετικών αποδόσεων [3]. Παρακάτω παρουσιάζονται οι βελτιστοποιητές από τους οποίους οι περισσότεροι χρησιμοποιήθηκαν στα πειράματα της παρούσας διπλωματικής εργασίας.

2.4.1 Gradient Descent

Κατάβαση κλίσης (Gradient Descent): Η κατάβαση κλίσης είναι ο πιο βασικός αλγόριθμος με βάση την κλίση που μπορεί να εφαρμοστεί για την εκπαίδευση ενός βαθύ μοντέλου. Ο αλγόριθμος καλείται, επίσης, μερικές φορές κατάβαση κλίσης της παρτίδας (batch gradient descent) ή ντετερμινιστική κλίση κατάβασης (deterministic gradient descent) στη βιβλιογραφία των νευρωνικών δικτύων, επειδή ενημερώνει τις παραμέτρους μόνο αφού έχει δει μια παρτίδα όλων των παραδειγμάτων εκπαίδευσης και η κλίση υπολογίζεται ντετερμινιστικά. Αυτή η μέθοδος περιλαμβάνει την ενημέρωση των παραμέτρων του μοντέλου θ με ένα μικρό βήμα προς την κατεύθυνση της κλίσης της αντικειμενικής συνάρτησης. Για την περίπτωση της επιβλεπόμενης μάθησης με ζεύγη δεδομένων $[x(t), y(t)]$:

$$\theta \leftarrow \theta + \in \nabla_\theta \sum_t L(f(x^{(t)}; \theta), y^{(t)}; \theta)$$

όπου $\in$ είναι ο ρυθμός μάθησης (learning rate). Ακολουθώντας την κλίση με αυτόν τον τρόπο είναι εγγυημένη η μείωση της απώλειας (loss) μόνο εάν το $\in$ είναι μικρότερο από μια τιμή κατωφλίου (μεγαλύτερη από $1/L$, όπου το L είναι η σταθερά Lipschitz ή το μεγαλύτερη δεύτερη παράγωγος προς οποιαδήποτε κατεύθυνση και αυτή η δήλωση ισχύει μόνο εάν η κλίση είναι συνεχής κατά Lipschitz).



Εικόνα 2.16 Μια απεικόνιση του τρόπου με τον οποίο οι παράγωγοι μιας συνάρτησης μπορούν να χρησιμοποιηθούν για να ακολουθήσουν τη συνάρτηση προς το ελάχιστο. Αυτή η τεχνική ονομάζεται κλίση κατάβασης.

Σημειώστε ότι ο ρυθμός εκμάθησης δεν χρειάζεται να μειωθεί προς το 0, στην περίπτωση μαζικής διαβάθμισης (batch gradient). Υπάρχει σύγκλιση με ένα σταθερό $\in$ επειδή οι κλίσεις γίνονται όλο και μικρότερες (πλησιάζοντας το 0) καθώς πλησιάζεται το τοπικό ελάχιστο μιας συνάρτησης της οποίας οι κλίσεις είναι συνεχόμενες κατά Lipschitz [3].

2.4.2 SGD

Stochastic Gradient Descent (SGD) Optimizer: Η βασική εξίσωση για το SGD φαίνεται στην πρώτη εξίσωση. Το SGD χρησιμοποιεί ορμή στην εκ νέων διαστάσεων κλίση που εμφανίζεται στην δεύερη εξίσωση για ενημέρωση των παραμέτρων.

$$\theta \leftarrow \theta - a\nabla_{\theta} J(\theta,\, x_i,\, y_i)$$

$$\theta \leftarrow \theta - (\gamma\theta + a\nabla_{\theta} J(\theta,\, x_i,\, y_i)$$

2.4.3 Nesterov Momentum

Nesterov Momentum Gradient Descent: Εάν η ορμή είναι αρκετά υψηλή, κοντά στο ελάχιστο, η βελτιστοποίηση μπορεί να υπερβεί το ελάχιστο. Ο προηγούμενος αλγόριθμος λαμβάνει υπόψη την τρέχουσα και τις προηγούμενες κλίσεις για την ενημέρωση των βαρών. Ωστόσο, για να γίνει ο αλγόριθμος βελτιστοποίησης πιο ισχυρός, πρέπει να ληφθούν υπόψιν και οι μελλοντικές κλίσεις, για να προσεγγίσουν την κατεύθυνση των κλίσεων. Τα βάρη ενημερώνονται με βάση την εξίσωση:

$$V_t = \lambda V_{t-1} - \eta \frac{\partial C}{\partial w_t}$$

$$\frac{\partial C}{\partial w_t} = \nabla_w C\,(w_t - \lambda V_{t-1}; x^{(i:i+n)}; y^{(i:i+n)})$$

$$w_{t+1} = w_t - V_t$$

όπου V είναι η ταχύτητα και αρχικοποιείται στο 0, το λ χρησιμοποιείται για να επιλέξει τον απαιτούμενο αριθμό πληροφοριών από την προηγούμενη ενημέρωση. Ενώ, η είναι ο ρυθμός μάθησης, w_t τα βάρη στο βήμα t, το n είναι ο αριθμός των σημείων δεδομένων, $C(.)$ είναι η συνάρτηση κόστους, και $\nabla_w C(w_t)$ είναι η κλίση των παραμέτρων βάρους w_t. Η σχέση $w_t - \lambda V_{t-1}$ υπολογίζει τη θέση που μπορεί να προσεγγίσει την επόμενη κλίση, επιτρέποντάς της έτσι να επιβραδύνει εάν απειλεί να υπερβεί το ελάχιστο.

2.4.4 RMSprop

Οι T. Tieleman και G. Hinton [11][12] εισήγαγαν το RMSprop ως ένα νέο εργαλείο βελτιστοποίησης που διαιρεί το ρυθμό μάθησης για ένα βάρος με τον τρέχοντα μέσο όρο των μεγεθών των πρόσφατων κλίσεων για αυτό το βάρος. Η εξίσωση της ορμής για το RMSprop έχει ως εξής:

$$u(t) = au(t-1) - \varepsilon \frac{\partial E}{\partial w}(t)$$

$$\begin{aligned} \Delta w(t) &= u(t) \\ &= au(t-1) - \varepsilon \frac{\partial E}{\partial w}(t) \\ &= a\Delta u(t-1) - \varepsilon \frac{\partial E}{\partial w}(t) \end{aligned}$$

Το RMSProp δεν κάνει διόρθωση της προκατάληψης (bias), η οποία προκαλεί σημαντικά προβλήματα όταν δημιουργείται το πρόβλημα της αραιής κλίσης (sparse gradient).

2.4.5 Adam

Ο Adam είναι ένα άλλο στοχαστικό εργαλείο βελτιστοποίησης κλίσης που χρησιμοποιεί μόνο τις δύο πρώτες ροπές της κλίσης (u και m, που εμφανίζονται στις παρακάτω εξισώσεις) και υπολογίζει τον μέσο όρο γι' αυτά. Μπορεί να χειριστεί το μη στατικό χαρακτήρα της αντικειμενικής συνάρτησης όπως η RMSProp, ξεπερνώντας παράλληλα τον περιορισμό της αραιής κλίσης του RMSProp [13].

$$\theta \leftarrow \theta - \frac{a}{\sqrt{\hat{u}}+\varepsilon}\hat{m}$$

$$g_{i,t} = \nabla_\theta J(\theta_i, x_i, y_i)$$

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_{i,t}$$

$$m_t = \beta_2 u_{t-1} + (1-\beta_2) g_{i,t}^2$$

όπου το m_t είναι η πρώτη ροπή και το u_t δείχνει τη δεύτερη ροπή που και τα δύο υπολογίζονται. Ισχύουν $\hat{m}_t = \frac{m_t}{1-\beta_1^t}$ και $\hat{u}_t = \frac{u_t}{1-\beta_2^t}$.

2.4.6 Adagrad

Το Adagrad αντιμετωπίζεται στο [14] ως μια νέα οικογένεια μεθόδων υπο-κλίσεων που απορροφούν δυναμικά τη γνώση από τη γεωμετρία των δεδομένων για να πραγματοποιήσουν μια πιο ενημερωτική εκπαίδευση με βάση την κλίση. Το Adagrad είναι μια επέκταση του SGD. Στην επανάληψη k, η κλίση ορίζεται ως:

$$G^{(k)} = diag\left[\sum_{i=1}^{k} g^{(i)}(g^{(i)})^T\right]^{1/2}$$

διαγώνιος πίνακας:

$$G_{jj}^{(k)} = \sqrt{\sum_{i=1}^{k}(g^{(i)})^2}$$

κανόνας ενημέρωσης:

$$x^{(k+1)} = argmin_{x\varepsilon X}\{\ \langle \nabla f(x^{(k)}), x\rangle\ + \frac{1}{2a_k}||x - x^{(k)}||^2_{G^{(k)}}\}$$
$$= x^{(k)} - aB^{-1}\nabla f(x^{(k)}),\quad (if\ X = R^n)$$

2.4.7 Adadelta

Το Adadelta, που εισήχθη από τον M.D. Zeiler [11][15], χρησιμοποιεί τον εκθετικά φθίνον μέσο όρο (exponentially decaying average) του g_t ως 2η ροπή της κλίσης. Αυτή η μέθοδος είναι μια βελτιωμένη έκδοση του Adagrad που βασίζεται μόνο σε πρώτης σειράς πληροφορίες. Ο κανόνας ενημέρωσης για την Adadelta είναι:

$$g_{t+1} = \gamma g_t + (1-\gamma)\nabla L(\theta)^2$$

$$x_{t+1} = \gamma x_t + (1-\gamma)u_{t+1}^2$$

$$x_{t+1} = -\frac{\sqrt{x_t+\varepsilon}\delta L(\theta_t)}{\sqrt{g_{t+1}+\varepsilon}}$$

2.4.8 Adamax

Το Adamax [16] είναι μια παραλλαγή του αλγορίθμου Adam, όπου η μη κεντρική διακύμανση τείνει στο άπειρο. Τα βάρη ενημερώνονται με βάση την εξίσωση:

$$w_t^i = w_{t-1}^i - \frac{\eta}{u_t+\varepsilon} \cdot \widehat{m}_t$$

όπου:

$$\widehat{m}_t = \frac{m_t}{\left(1-\beta_1^t\right)}$$

$$u_t = max\left(\beta_2 \cdot u_{t-1}, \left|G_t\right|\right)$$

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

$$G = \nabla_w C\,(w_t)$$

όπου η είναι ρυθμός μάθησης, w_t είναι τα βάρη στο βήμα t, $C\,(.)$ είναι η συνάρτηση κόστους, και $\nabla_w C\,(w_t)$ είναι η κλίση των παραμέτρων βάρους w_t. Το β_i χρησιμοποιείται για να επιλεχθεί ο αριθμός των πληροφοριών που απαιτούνται από την προηγούμενη ενημέρωση, όπου $\beta_i \in [0, 1]$, ενώ το m_t είναι η πρώτη ροπή, u_t είναι η δεύτερη ροπή συνάρτησης.

2.4.9 Nadam

Το Nadam [16] είναι μια επέκταση του αλγορίθμου Adam συνδυάζοντάς τον με το Nesterov momentum gradient descent. Τα βάρη ενημερώνονται με βάση την εξίσωση:

$$w_t^i = w_{t-1}^i - \frac{\eta}{\sqrt{\overline{m_t}+\varepsilon}} \cdot \overline{m_t}$$

όπου:

$$\overline{m_t} = \beta_1^{t+1}\widehat{m}_t + \left(1-\beta_1^t\right)\widehat{g_t}$$

$$\widehat{m}_t = \frac{m_t}{\left(1-\prod_{i=1}^{t}\beta_1^i\right)}$$

$$\widehat{g_t} = \frac{g_t}{\left(1-\prod_{i=1}^{t}\beta_1^i\right)}$$

όπου η είναι ρυθμός μάθησης, w_t είναι τα βάρη στο βήμα t, ο β_i χρησιμοποιείται για την επιλογή του όγκου των πληροφοριών που απαιτούνται από την προηγούμενη ενημέρωση, όπου $\beta_i \varepsilon\ [0, 1]$, το m_t είναι η πρώτη ροπή συνάρτησης.

2.5 Αξιολόγηση προβλέψεων

Η μηχανική μάθηση παίζει κεντρικό ρόλο στη σύγχρονη καθημερινότητα. Είτε εφαρμόζεται μοντελοποίηση τεχνικών πρόβλεψης στην έρευνα είτε σε επιχειρηματικά προβλήματα, πάντα ο στόχος είναι ένας: η παραγωγή «καλών» προβλέψεων. Η εφαρμογή ενός μοντέλου στα δεδομένα εκπαίδευσης είναι ένα πράγμα, αλλά πώς ξέρει κανείς ότι το μοντέλο γενικεύει καλά τα δεδομένα που δεν εμφανίζονται; Πώς ξέρει ότι δεν απομνημονεύει απλώς τα δεδομένα που του τροφοδοτησε ο προγραμματιστής και ότι δεν θα κάνει καλές προβλέψεις για μελλοντικά δείγματα - δείγματα που δεν έχει ξαναδεί; Και πώς επιλέγεται πρώτα-πρώτα ένα καλό μοντέλο;

2.5.1 Μέθοδοι αξιολόγησης προβλέψεων

Η αξιολόγηση ενός μοντέλου σίγουρα δεν είναι το μόνο που απασχολεί τον προγραμματιστή στη μηχανική μάθηση. Πριν χειριστεί οποιαδήποτε δεδομένα, πρέπει να σχεδιάσει και να χρησιμοποιήσει τεχνικές που είναι κατάλληλες για τους σκοπούς του. Σε αυτή την ενότητα, θα εξεταστούν οι κυριότερες τεχνικές [17].

2.5.1.1 Holdout Validation

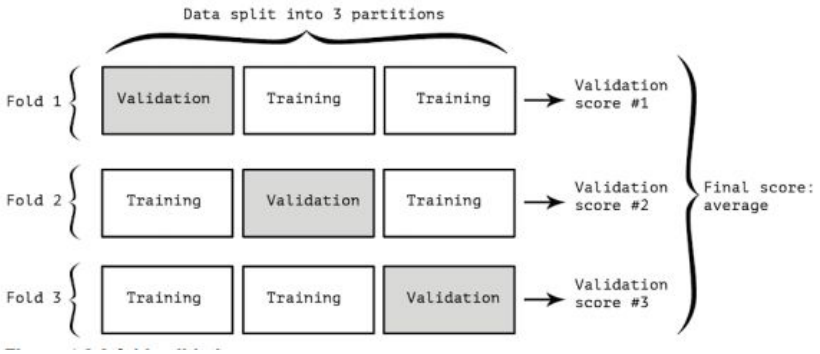
Η μέθοδος holdout είναι αναμφισβήτητα η απλούστερη τεχνική αξιολόγησης προβλέψεων. Μπορεί να συνοψιστεί ως εξής: Αρχικά, τα δεδομένα με τις ετικέτες τους (labeled dataset) χωρίζονται σε δύο μέρη: στο σύνολο εκπαίδευσης (train set) και το σύνολο δοκιμών (test set). Μετά, ένα μοντέλο εφαρμόζεται στο σύνολο εκπαίδευσης και προβλέπει τις ετικέτες του συνόλου δοκιμών. Η ακρίβεια των προβλέψεων, η οποία μπορεί να υπολογιστεί συγκρίνοντας τις προβλεπόμενες ετικέτες με τις πραγματικές ετικέτες του συνόλου δοκιμών, αποτελεί την εκτίμηση για την ακρίβεια της πρόβλεψης του μοντέλου. Εδώ, είναι σημαντικό να σημειωθεί ότι δεν πρέπει να εκπαιδευτεί και να αξιολογηθεί ένα μοντέλο στο ίδιο σύνολο δοκιμών, δεδομένου ότι θα εισήγαγε συνήθως μία πολύ αισιόδοξη προκατάληψη λόγω της υπερεκπαίδευσης (overfitting). Με άλλα λόγια, δεν θα είναι δυνατό να κριθεί αν το μοντέλο απλώς απομνημόνευσε τα δεδομένα του συνόλου δοκιμών, ή αν γενικεύεται καλά σε νέα, αόρατα δεδομένα.

Εικόνα 2.17 Simple hold-out validation split

Συνήθως, ο διαχωρισμός ενός συνόλου δεδομένων σε σύνολο εκπαίδευσης και σύνολο δοκιμών είναι μια απλή διαδικασία τυχαίας δειγματοληψίας. Γίνεται η υπόθεση πως όλα τα δεδομένα χαρακτηρίζονται από την ίδια κατανομή πιθανότητας (σε σχέση με την κατηγορία που ανήκουν). Επιλέγονται τυχαία τα 2/3 αυτών των δειγμάτων για το σύνολο εκπαίδευσης και το 1/3 των δειγμάτων για το σύνολο δοκιμών. Επίσης, συχνές διασπάσεις train/test set είναι 60/40, 70/30 ή 80/20 - ή ακόμη και 90/10 εάν το dataset είναι σχετικά μεγάλο [20].

2.5.1.2 K-fold Validation

Τα δεδομένα χωρίζονται σε K διαμερίσεις ίσου μεγέθους. Για κάθε διαμέριση i, εκπαιδεύεται ένα μοντέλο στις υπόλοιπες διαμερίσεις $K-1$ το οποίο και αξιολογείται στη διαμέριση i. Το τελικό αποτέλεσμα θα είναι οι μέσοι όροι των K αποτελεσμάτων που υπολογίστηκαν. Αυτή η μέθοδος είναι χρήσιμη όταν η απόδοση του μοντέλου εμφανίζει σημαντική διακύμανση με βάση το διαχωρισμό των συνόλου εκπαίδευσης/συνόλου δοκιμών. Όπως στη μέθοδος holdout, και σε αυτή τη μέθοδο κρίνεται αναγκαία η χρήση ενός ξεχωριστού συνόλου επικύρωσης (validation set) για τη βαθμονόμηση του μοντέλου.



Εικόνα 2.18 k-fold validation (k = 3)

2.5.2 Μετρικές Αξιολόγησης Παλινδρόμησης

Για να μπορούν να συγκριθούν τα διάφορα μοντέλα, χρειάζεται ένας ορισμός του μέτρου απόδοσής του μοντέλου, P. Δίνεται ένας πίνακας με n παραδείγματα εισόδου που δεν θα χρησιμοποιηθούν για την εκπαίδευση, αλλά για την αξιολόγηση της απόδοσης του μοντέλου. Επίσης, δίνεται ένα διάνυσμα στόχων παλινδρόμησης που παρέχει τη σωστή τιμή του y για καθένα από αυτά τα παραδείγματα. Επειδή το σύνολο δεδομένων θα χρησιμοποιηθεί μόνο για την αξιολόγηση, επιλέγεται το σύνολο δοκιμών (test set) και οι μετρικές αξιολόγησης ως σφάλμα δοκιμών (test error). Από εδώ και στο εξής, θα αναφέρεται ο πίνακας σχεδίασης των εισόδων ως X^{test} και του διανύσματος των στόχων παλινδρόμησης ως y^{test} [21].

2.5.2.1 MAE

Μέσο Απόλυτο Σφάλμα (Mean Absolute Error MAE): όπου n ο ορίζοντας πρόβλεψης, $\hat{y}_i, y_i$ η προβλεπόμενη ετικέτα και η πραγματική ετικέτα για $i = 1, ..., n$ το σύνολο δοκιμών (test set) αντίστοιχα:

$$MAE = \frac{1}{n}\sum_i \left|\hat{y}_i^{test} - y_i^{test}\right|$$

2.5.2.2 MSE

Μέσο Τετραγωνικό Σφάλμα (Mean Square Error MSE): όπου n ο ορίζοντας πρόβλεψης, $\hat{y}_i, y_i$ η προβλεπόμενη ετικέτα και η πραγματική ετικέτα για $i = 1, ..., n$ το σύνολο δοκιμών (test set) αντίστοιχα:

$$MSE = \frac{1}{n}\sum_i \left(\hat{y}_i^{test} - y_i^{test}\right)^2$$

2.5.2.3 RMSE

Ρίζα του Μέσου Τετραγωνικού Σφάλματος (Root Mean Square Error RMSE): όπου n ο ορίζοντας πρόβλεψης, $\hat{y}_i, y_i$ η προβλεπόμενη ετικέτα και η πραγματική ετικέτα για $i = 1, ..., n$ το σύνολο δοκιμών (test set) αντίστοιχα:

$$RMSE = \sqrt{\frac{1}{n}\sum_i \left(\hat{y}_i^{test} - y_i^{test}\right)^2}$$

2.6 Υπερεκπαίδευση και Υποεκπαίδευση

Κατά την εκπαίδευση (training) ενός μοντέλου μηχανικής μάθησης, ο προγραμματιστής έχει πρόσβαση σε ένα σύνολο εκπαίδευσης και μπορεί να υπολογίσει κάποιο μέτρο σφάλματος σε αυτό το σύνολο που ονομάζεται σφάλμα εκπαίδευσης (training error) το οποίο προσπαθεί να μειώσει. Μέχρι στιγμής, αυτά τα βήματα που περιγράφηκαν είναι απλά ένα πρόβλημα βελτιστοποίησης. Αυτό που διαχωρίζει τη μηχανική μάθηση από τη βελτιστοποίηση είναι ότι στη μηχανική μάθηση πρέπει το σφάλμα γενίκευσης να είναι επίσης χαμηλό. Το σφάλμα γενίκευσης (generalization error) ορίζεται ως η αναμενόμενη τιμή του σφάλματος σε μία νέα είσοδο [3]. Το σφάλμα γενίκευσης ενός μοντέλου μηχανικής εκμάθησης υπολογίζεται συνήθως μετρώντας την απόδοσή του σε ένα σύνολο δοκιμών που συλλέχθηκε ξεχωριστά από το σύνολο εκπαίδευσης.

Πώς μπορεί, όμως, κάποιος να επηρεάσει την απόδοση του συνόλου δοκιμών όταν παρατηρεί μόνο το σύνολο εκπαίδευσης; Το πεδίο της στατιστικής μαθησιακής θεωρίας παρέχει μερικές απαντήσεις. Εάν το σύνολο εκπαίδευσης και το σύνολο δοκιμών συλλέγονται αυθαίρετα, υπάρχουν πράγματι λίγα πράγματα που μπορούν να γίνουν. Εάν επιτρέπεται να γίνουν κάποιες υποθέσεις σχετικά με τον τρόπο συλλογής των συνόλων εκπαίδευσης και δοκιμών, τότε μπορεί να σημειωθεί κάποια πρόοδος.

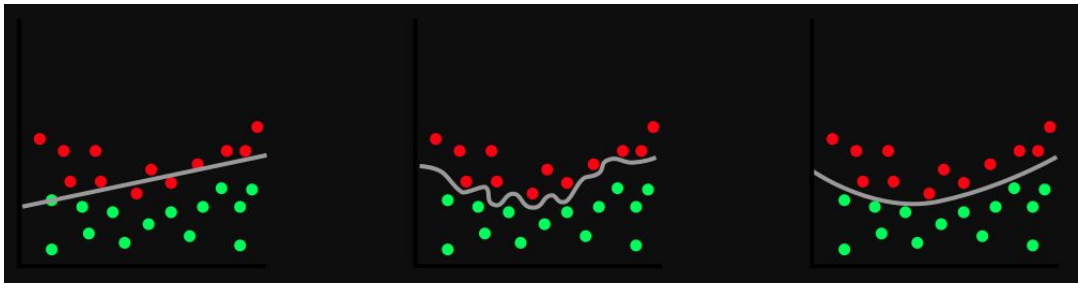
Συνήθως γίνεται ένα σύνολο υποθέσεων γνωστές ως ανεξάρτητες & ταυτόσημες παραδοχές. Αυτές οι υποθέσεις θεωρούν πως τα παραδείγματα σε κάθε σύνολο δεδομένων είναι ανεξάρτητα το ένα από το άλλο, και ότι το σύνολο εκπαίδευσης και το σύνολο δοκιμών κατανέμονται πανομοιότυπα, και προέρχονται από την ίδια κατανομή πιθανότητας.

Μια άμεση σύνδεση που παρατηρείται μεταξύ του σφάλματος εκπαίδευσης και του σφάλματος δοκιμών είναι ότι για ένα τυχαία επιλεγμένο μοντέλο, και τα δύο σφάλματα έχουν την ίδια αναμενόμενη τιμή. Έστω μια κατανομή πιθανότητας $p(x, y)$ η οποία δειγματοληπτεί επανειλημμένα για τη δημιουργία των συνόλων εκπαίδευσης και δοκιμών. Για κάποια σταθερή τιμή w, το αναμενόμενο σφάλμα εκπαίδευσης σε αυτή τη διαδικασία δειγματοληψίας είναι ακριβώς το ίδιο με το αναμενόμενο σφάλμα του συνόλου δοκιμών. Η μόνη διαφορά μεταξύ τους είναι το όνομα που αποδίδεται στο σύνολο δεδομένων που επιλέγεται. Από αυτήν την παρατήρηση, είναι φυσικό να υποθέσει κανείς πως υπάρχει κάποια σχέση μεταξύ του σφάλματος εκπαίδευσης και του σφάλματος δοκιμών υπό αυτές τις υποθέσεις.

Φυσικά, όταν χρησιμοποιείται ένας αλγόριθμος μηχανικής μάθησης, δοκιμάζεται το σύνολο εκπαίδευσης το οποίο χρησιμοποιείται στη συνέχεια για να επιλεχθούν οι παράμετροι που θα μειώσουν το σφάλμα του συνόλου εκπαίδευσης και μετά δοκιμάζεται το σύνολο δοκιμών. Στο πλαίσιο αυτής της διαδικασίας, το αναμενόμενο σφάλμα δοκιμών είναι μεγαλύτερο ή ίσο με την αναμενόμενη τιμή του σφάλματος εκπαίδευσης. Οι παράγοντες που καθορίζουν πόσο καλά θα αποδώσει ένας αλγόριθμος μηχανικής μάθησης είναι η ικανότητά του να:

1. Να κάνει το σφάλμα εκπαίδευσης μικρό.
2. Να μειώσει τη διαφορά μεταξύ του σφάλματος εκπαίδευσης και του σφάλματος δοκιμών.

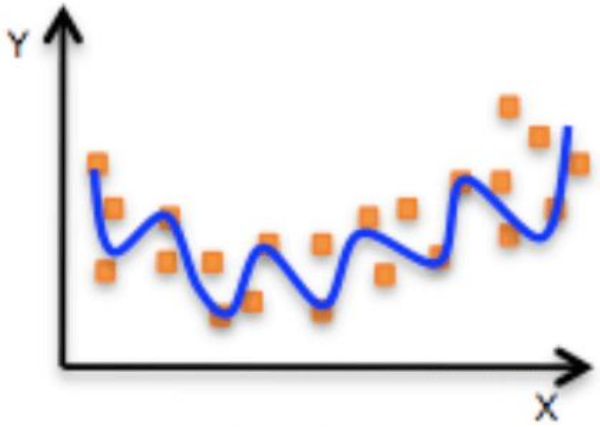
Αυτοί οι δύο παράγοντες αντιστοιχούν στις δύο κεντρικές προκλήσεις της μηχανικής μάθησης: υποεκπαίδευση και υπερεκπαίδευση. Η υποεκπαίδευση συμβαίνει όταν το μοντέλο δεν είναι σε θέση να αποκτήσει μια αρκετά χαμηλή τιμή σφάλματος στο σύνολο εκπαίδευσης. Η υπερεκπαίδευση συμβαίνει όταν το χάσμα μεταξύ του σφάλματος εκπαίδευσης και του σφάλματος δοκιμών είναι πολύ μεγάλο.



Εικόνα 2.19 Παράδειγμα υποεκπαίδευσης, υπερεκπαίδευσης και ισορροπημένου μοντέλου αντίστοιχα

2.6.1 Υπερεκπαίδευση

Πιο συγκεκριμένα, η υπερεκπαίδευση (overfitting) δημιουργείται εξαιτίας της απομνημόνευσης των δεδομένων εκπαίδευσης και συνήθως οδηγεί σε κακή απόδοση στο σύνολο δοκιμών. Αυτό σημαίνει ότι η απόδοση στο σύνολο εκπαίδευσης μπορεί να είναι πολύ καλή, αλλά η απόδοση στο σύνολο δοκιμών να είναι αρκετά κακή. Η απώλεια της γενίκευσης του δικτύου μπορεί να οφείλεται σε πολλά ζητήματα, όπως η χωρητικότητα του δικτύου ή η φύση του ίδιου του συνόλου εκπαίδευσης.



Εικόνα 2.20 Overfitting

Όταν το μοντέλο δεν υπολειτουργεί, τα bias είναι γενικά υψηλά και η διακύμανση (variance) είναι χαμηλή. Το overfitting χαρακτηρίζεται συνήθως από υψηλή διακύμανση και χαμηλά bias . Σε πολλές περιπτώσεις, οι μικρές αυξήσεις των bias οδηγούν σε μεγάλες μειώσεις στη διακύμανση.

2.6.1.1 Αντιμετώπιση Υπερεκπαίδευσης

Πολλά μέτρα έχουν εισαχθεί στη βιβλιογραφία για να ξεπεραστεί η υπερεκπαίδευση (overfitting). Παρακάτω είναι μερικές τεχνικές που βοηθούν στην αντιμετώπιση του overfitting [3][16][20].

2.6.1.1.1 Αύξηση Συνόλου Δεδομένων

Σύμφωνα με το [3], ο καλύτερος τρόπος για να βελτιωθεί η γενίκευση ενός μοντέλου μηχανικής μάθησης είναι να εκπαιδευτεί σε περισσότερα δεδομένα. Φυσικά, στην πράξη, ο όγκος των δεδομένων που μπορεί να υπάρχουν να είναι περιορισμένος. Ένας τρόπος αντιμετώπισης αυτού του προβλήματος είναι η δημιουργία ψεύτικων δεδομένων και η προσθήκη τους στο σύνολο εκπαίδευσης.

Αυτή η προσέγγιση είναι ευκολότερη για προβλήματα ταξινόμησης. Ένας ταξινομητής πρέπει να λάβει μια περίπλοκη, πολυδιάστατη είσοδο x και να την συνοψίσει κατηγορία y. Η αύξηση του συνόλου δεδομένων ήταν μια ιδιαίτερα αποτελεσματική τεχνική για ένα συγκεκριμένο πρόβλημα ταξινόμησης: αναγνώριση αντικειμένων. Οι εικόνες έχουν πολυδιάστατες και περιλαμβάνουν μια τεράστια ποικιλία παραγόντων που μπορούν να παραλλαχθούν, πολλοί από τους οποίους μπορούν εύκολα να προσομοιωθούν. Βέβαια, η προσέγγιση αυτή δεν εφαρμόζεται τόσο εύκολα σε πολλές άλλες εργασίες. Για παράδειγμα, είναι δύσκολο να δημιουργηθούν νέα ψεύτικα δεδομένα για ένα πρόβλημα εκτίμησης πυκνότητας εκτός εάν έχει ήδη λυθεί το πρόβλημα εκτίμησης πυκνότητας.

2.6.1.1.2 Πρόωρη διακοπή

Είναι πιθανόν η πιο συχνά χρησιμοποιούμενη μορφή κανονικοποίησης στη βαθιά μάθηση. Η δημοτικότητά του οφείλεται τόσο στην αποτελεσματικότητα όσο και στην απλότητά του. Η πρόωρη

διακοπή (early stopping) είναι ουσιαστικά ένα προληπτικό μέτρο που χρησιμοποιείται για να αποφευχθεί η υπερεκπαίδευση του δικτύου, και μπορεί να οριστεί ως η διακοπή της εκπαίδευσης του δικτύου όταν η απόδοση στο σύνολο εκπαίδευσης σταματά να βελτιώνεται για έναν προκαθορισμένο αριθμό εποχών.

2.6.1.1.3 Απόρριψη

Η παράμετρος απόρριψης (dropout) είναι άλλη μία πολύ αποτελεσματική και πολύ συχνά χρησιμοποιούμενη τεχνική γενίκευσης για τα νευρωνικά δίκτυα, που αναπτύχθηκε από τον Hinton και τους μαθητές του στο Πανεπιστήμιο του Τορόντο. Η παράμετρος απόρριψης, που εφαρμόζεται σε ένα επίπεδο (layer), λειτουργεί σύμφωνα με μια τυχαία "απόρριψη" (π.χ. ορισμένη στο μηδέν) μιας σειράς χαρακτηριστικών εξόδου του επιπέδου κατά τη διάρκεια της εκπαίδευσης. Έστω ότι ένα επίπεδο κανονικά θα επέστρεφε για ένα δεδομένο δείγμα εισόδου ένα διάνυσμα [0.6, 0.7, 1.3, 0.2, 1.7] κατά τη διάρκεια της εκπαίδευσης. Μετά την εφαρμογή της παραμέτρου απόρριψης, αυτό το διάνυσμα θα έχει μερικές μηδενικές καταχωρίσεις, τυχαία κατανεμημένες. Για παράδειγμα, το διάνυσμα εισόδου μετά την εφαρμογή της παραμέτρου απόρριψης θα μπορούσε να έχει την μορφή [0, 0.7, 1.3, 0, 1.7]. Το "ποσοστό απόρριψης" (dropout rate) συνήθως ορίζεται μεταξύ του 0,2 και 0,5. Κατά τη διάρκεια της εφαρμογής του μοντέλου στο σύνολο δοκιμών, καμία μονάδα δεν απορρίπτεται, και αντίθετα οι τιμές εξόδου του επιπέδου μειώνονται κατά έναν παράγοντα ίσο με το ποσοστό απόρριψης, έτσι ώστε να εξισορροπηθεί το γεγονός ότι περισσότερες μονάδες είναι ενεργές τώρα από ό,τι κατά την διάρκεια της εκπαίδευσης. Περισσότερες λεπτομέρειες σχετικά με την παράμετρο απόρριψης παρουσιάζονται σε επόμενη ενότητα.

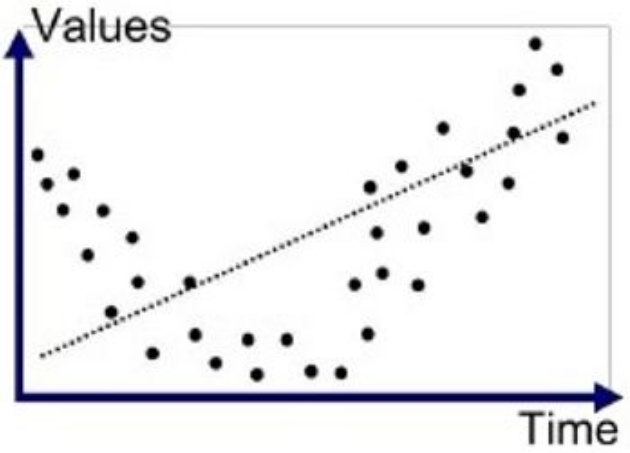
2.6.1.1.4 Κανονικοποίηση Βάρους

Σύμφωνα με την αρχή του ξυραφιού του Occam [20]: δίνοντας δύο εξηγήσεις για ένα πράγμα, η εξήγηση που είναι πιο πιθανό να είναι σωστή είναι η "απλούστερη", αυτή που κάνει τις λιγότερες υποθέσεις. Αυτό ισχύει και για τα μοντέλα των νευρωνικών δικτύων: δεδομένων κάποιου συνόλου εκπαίδευσης και κάποιας αρχιτεκτονικής δικτύου, υπάρχουν πολλοί συνδυασμοί τιμών βάρους (πολλαπλά μοντέλα) που θα μπορούσαν να εξηγήσουν τα δεδομένα και τα απλούστερα μοντέλα είναι αυτά που είναι λιγότερο πιθανό να υπερεκπαιδευτούν σε σχέση με τα πιο πολύπλοκα.

Ένα "απλό μοντέλο" σε αυτό το πλαίσιο είναι ένα μοντέλο όπου η κατανομή των τιμών των παραμέτρων του έχει λιγότερη εντροπία (ή ένα μοντέλο με λιγότερες παραμέτρους συνολικά). Έτσι, ένας κοινός τρόπος για τον περιορισμό της υπερεκπαίδευσης είναι να τεθούν περιορισμοί στην πολυπλοκότητα ενός δικτύου, αναγκάζοντας τα βάρη του να λαμβάνουν μόνο μικρές τιμές, γεγονός που καθιστά την κατανομή των τιμών βάρους πιο «κανονική». Αυτό ονομάζεται "κανονικοποίηση βάρους" (Weight Regularization) και γίνεται με την προσθήκη στη συνάρτηση κόστους του δικτύου ένα κόστος που σχετίζεται με την ύπαρξη μεγάλων βαρών.

2.6.2 Υποεκπαίδευση

Η υποεκπαίδευση (underfitting) συμβαίνει όταν το μοντέλο δεν είναι σε θέση να καταγράψει τη μεταβλητότητα των δεδομένων. Για παράδειγμα, έστω ότι κάποιος εκπαιδεύει έναν γραμμικό ταξινομητή (linear classifier) σε ένα σύνολο δεδομένων που είναι παραβολή. Ο προκύπτων ταξινομητής δεν θα έχει ισχυρή ισχύ ούτε θα μπορεί να χαρτογραφήσει σωστά τα δεδομένα εκπαίδευσης.



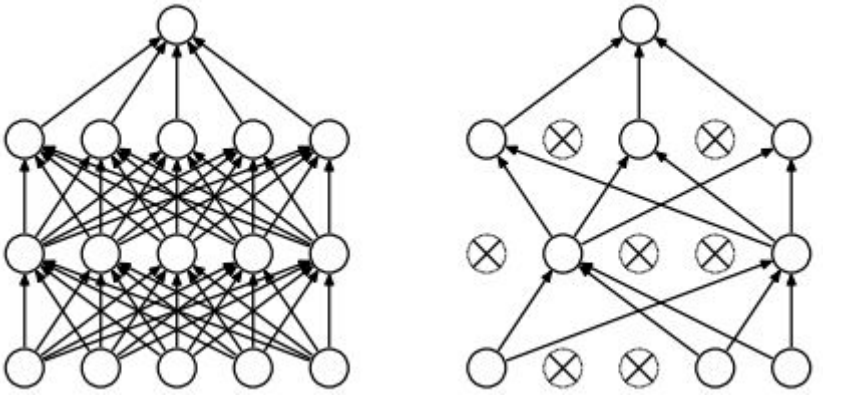
Εικόνα 2.21 Underfitting

2.7 Απόρριψη

Τα βαθιά νευρωνικά δίκτυα με μεγάλο αριθμό παραμέτρων είναι πολύ ισχυρά συστήματα μηχανικής μάθησης. Ωστόσο, το overfitting είναι ένα σοβαρό πρόβλημα σε τέτοια δίκτυα. Δεδομένου ότι τα μεγάλα δίκτυα είναι αργά στη χρήση, καθίσταται δύσκολη η αντιμετώπιση του overfitting και με τον συνδυασμό προβλέψεων πολλών διαφορετικών μεγάλων νευρωνικών δικτύων κατά τη διάρκεια της δοκιμής.

Όπως αναφέρθηκε και στην προηγούμενη ενότητα, η παράμετρος απόρριψης dropout αποτρέπει την υπερεκπαίδευση και παρέχει έναν τρόπο για να συνδυαστούν εκθετικά πολλές διαφορετικές αρχιτεκτονικές νευρωνικών δικτύων με αποτελεσματικό τρόπο. Η βασική ιδέα είναι πως απορρίπτονται τυχαία μονάδες από το νευρωνικό δίκτυο κατά τη διάρκεια της εκπαίδευσης. Με την απομάκρυνση μιας μονάδας, υπονοείται η προσωρινή αφαίρεσή της από το δίκτυο, μαζί με όλες τις εισερχόμενες και εξερχόμενες συνδέσεις της, όπως φαίνεται στην παρακάτω εικόνα.

Η επιλογή των μονάδων που θα "απορριφθούν' είναι τυχαία. Στην απλούστερη περίπτωση, κάθε μονάδα διατηρείται με σταθερή πιθανότητα p ανεξάρτητη από άλλες μονάδες, όπου το p μπορεί να επιλεγεί χρησιμοποιώντας ένα σύνολο εκπαίδευσης ή μπορεί απλώς να οριστεί στο 0,5, το οποίο φαίνεται να είναι σχεδόν βέλτιστο για ένα ευρύ φάσμα δικτύων. Ωστόσο, για τις μονάδες εισόδου, η βέλτιστη πιθανότητα συγκράτησης είναι συνήθως πιο κοντά στο 1 παρά στο 0,5.



(a) Ένα νευρωνικό δίκτυο (b) Μετά την εφαρμογή dropout

Εικόνα 2.22 Αριστερά: Ένα νευρωνικό δίκτυο με 2 κρυφά στρώματα. Δεξιά: Ένα παράδειγμα ενός αραιωμένου δικτύου ύστερα από εφαρμογή της παραμέτρου απόρριψης στο νευρωνικό δίκτυο στα αριστερά

Πιο αναλυτικά, η εφαρμογή της παραμέτρου απόρριψης σε ένα νευρωνικό δίκτυο οδηγεί στη δειγματοληψία ενός αραιωμένου δικτύου σε σχέση με το αρχικό. Το αραιωμένο δίκτυο αποτελείται από τους νευρώνες εκείνους που επιβίωσαν της παραμέτρου απόρριψης, όπως φαίνεται στην Εικόνα 2.21.

Κάθε δίκτυο με n νευρώνες μπορεί να θεωρηθεί ως ένα σύνολο από 2^n πιθανά υποδίκτυα. Αφού τα υποδίκτυα αυτά μοιράζονται τα βάρη, ο συνολικός αριθμός των παραμέτρων παραμένει $O(n^2)$ ή μικρότερος. Για κάθε αναπαράσταση σε κάθε φάση της εκπαίδευσης, ένα νέο υποδίκτυο δειγματοληπτείται και εκπαιδεύεται. Συνεπώς, η εκπαίδευση ενός νευρωνικού δικτύου με παράμετρο απόρριψης καταλήγει στην εκπαίδευση ενός συνόλου 2^n υποδικτύων τα οποία μοιράζονται τα βάρη τους, και το καθένα εκπαιδεύεται πολύ σπάνια, αν όχι καθόλου. Κατά τη διαδικασία δοκιμής (test time), δεν είναι δυνατόν να ληφθεί ο μέσος όρος των προβλέψεων από ένα εκθετικό αριθμό αραιωμένων δικτύων. Ωστόσο, μία πολύ απλή προσέγγιση δουλεύει καλά στην πράξη. Τα βάρη του δικτύου κανονικοποιούνται ανάλογα με την πιθανότητα επιβίωσης που είχαν οι νευρώνες τους κατά την εκπαίδευση. Έτσι, αν ένας νευρώνας επιβίωνε με πιθανότητα p κατά την εκπαίδευση, τα βάρη που ξεκινούν από αυτόν πολλαπλασιάζονται με p κατά τη φάση του ελέγχου του μοντέλου. Αυτό διαβεβαιώνει ότι για κάθε νευρώνα, η αναμενόμενη έξοδος (που ακολουθεί την κατανομή που χρησιμοποιήθηκε για την «απόρριψη» νευρώνων κατά την εκπαίδευση) ταυτίζεται με την πραγματική έξοδο στη φάση της δοκιμής.

Με την κανονικοποίηση αυτή, τα 2^n δίκτυα μπορούν να συνδυαστούν σε ένα ενιαίο δίκτυο το οποίο μπορεί να χρησιμοποιηθεί στη δοκιμή του μοντέλου. Έχει παρατηρηθεί πειραματικά ότι η εκπαίδευση ενός νευρωνικού δικτύου με παράμετρο απόρριψης οδηγεί σε σημαντική μείωση του σφάλματος γενίκευσης (generalization error) σε μια πληθώρα προβλημάτων και με τη χρήση διαφόρων αλγορίθμων για την αντιμετώπισή τους [22].

2.8 Βελτιστοποίηση Υπερπαραμέτρων

Οι περισσότεροι αλγόριθμοι μηχανικής μάθησης, όπως γίνεται αντιληπτό από τις προηγούμενες ενότητες, έχουν πολλές ρυθμίσεις που πρέπει να συντονίσουν με στόχο τον έλεγχο της συμπεριφοράς του αλγορίθμου μάθησης. Αυτές οι ρυθμίσεις ονομάζονται υπερπαραμέτροι. Οι τιμές των υπερπαραμέτρων δεν προσαρμόζονται από τον ίδιο τον αλγόριθμο μάθησης (αν και μπορεί να σχεδιαστεί ένα μετα-μοντέλο όπου θα μαθαίνει τις καλύτερες υπερπαραμέτρους για έναν άλλο μοντέλο, εργασία που υλοποιείται στην παρούσα διπλωματική εργασία).

Εάν σκεφτεί κανείς τον τρόπο με τον οποίο ο χρήστης ενός αλγορίθμου μάθησης αναζητά καλές τιμές υπερπαραμέτρων, συνειδητοποιεί ότι πρόκειται για βελτιστοποίηση: ο χρήστης προσπαθεί να βρει μία τιμή των υπερπαραμέτρων που βελτιστοποιεί μια αντικειμενική συνάρτηση, όπως το σφάλμα επικύρωσης, μερικές φορές υπό περιορισμούς (όπως περιορισμό για το χρόνο εκπαίδευσης ή τη μνήμη). Είναι επομένως δυνατόν, κατ 'αρχήν, να εφαρμοστούν αλγόριθμοι βελτιστοποίησης για να ενθυλακωθεί ένας αλγόριθμος μάθησης που περιλαμβάνει υπερπαραμέτρους και να αποδώσει έναν αλγόριθμο μάθησης χωρίς υπερπαραμέτρους. Αυτές οι διαδικασίες καλούνται αλγόριθμοι βελτιστοποίησης υπερπαραμέτρων. Δυστυχώς, αυτοί οι αλγόριθμοι συνήθως περιλαμβάνουν δικές τους υπερπαραμέτρους, όπως το εύρος τιμών που πρέπει να διερευνηθούν, αλλά αυτές τείνουν να επηρεάζουν πολύ λιγότερο το τελικό αποτέλεσμα [3].

3. Ανάλυσης χρονοσειρών και βαθιά μάθηση

3.1 Εισαγωγή στην Ανάλυση Χρονοσειρών

Τα δεδομένα μπορούν να αναπαρασταθούν σε διαφορετικές μορφές, από τις πιο βασικές, όπως αριθμητικές και ονομαστικές, έως πιο περίπλοκες, όπως ήχος και βίντεο. Ωστόσο, η αποθήκευση των χρονικών πληροφοριών, η οποία επιτρέπει τη χρονολογική οργάνωση των συλλεγόμενων δεδομένων, είναι μία από τις αναπαραστάσεις δεδομένων που έχει προσελκύσει την περισσότερη προσοχή των ερευνητών και οδήγησε στη δημιουργία μεγάλων βάσεων δεδομένων. Ορίζεται χρονοσειρά (time series) μία σειρά από σημεία δεδομένων τοποθετημένα σε μία χρονική σειρά. Συνήθως η χρονοσειρά αποτελείται από σημεία που ισαπέχουν χρονικά και έτσι είναι μία χρονική σειρά διακριτών σημείων.

Η χρονική εξόρυξη δεδομένων είναι μία διαδικασία για την εξαγωγή χρήσιμων πληροφοριών από χρονοσειρές. Η πρόβλεψη είναι μία από τις εργασίες που προβλέπεται να υλοποιείται από τη χρονική εξόρυξη δεδομένων, η οποία και είναι αναγκαία για τη μείωση της μελλοντικής αβεβαιότητας, ιδίως λόγω της αστάθειας ορισμένων φαινομένων. Μερικά παραδείγματα είναι η πρόβλεψη της τιμής κλεισίματος των μετοχών μιας εταιρείας καθημερινά, η πρόβλεψη της ανεργίας για ένα κράτος κάθε τρίμηνο και η πρόβλεψη ενεργειακού φορτίου. Έτσι, οι προβλέψεις αποτελούν ουσιαστικό εργαλείο σε πολλούς τομείς για τη λήψη αποφάσεων ως εμπορικές και χρηματοοικονομικές στρατηγικές ή πολιτικές αποφάσεις, και η κατάλληλη χρήση τους μπορεί να προκαλέσει κοινωνικές και οικονομικές επιπτώσεις [23].

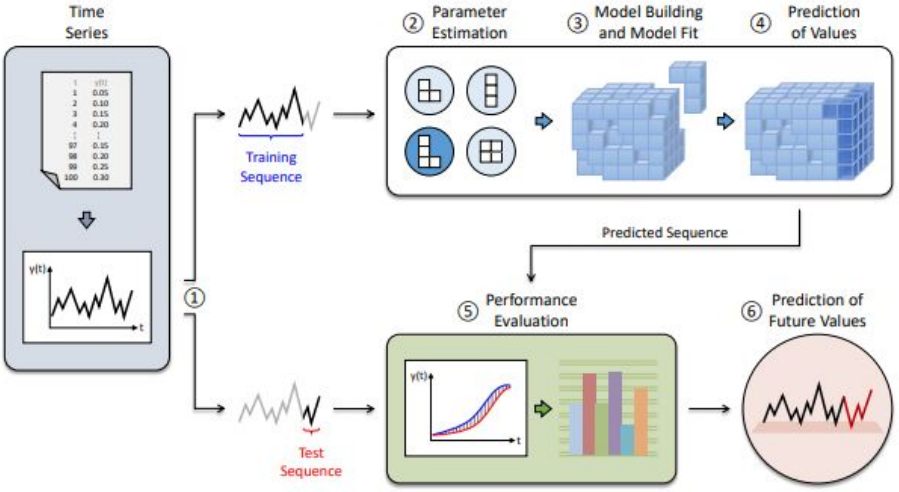
3.2 Ανάλυση χρονοσειρών στην βαθιά μάθηση

3.2.1 Διαδικασία πρόβλεψης χρονοσειρών

Η διαδικασία πρόβλεψης χρονοσειρών καλύπτει έξι βήματα [23]:

1. Το πρώτο βήμα χωρίζει τη χρονοσειρά σε δύο ακολουθίες: μία πριν από τον ορίζοντα πρόβλεψης, η οποία προορίζεται για την εκπαίδευση μοντέλου· και μία άλλη μετά από αυτήν την περίοδο, η οποία χρησιμοποιείται για να αξιολογήσει την ποιότητα του εκπαιδευμένου μοντέλου.
2. Το δεύτερο βήμα επιλέγει τη δομή του μοντέλου πρόβλεψης με βάση τα χαρακτηριστικά των δεδομένων και εκτιμά τις υπερπαραμέτρους χρησιμοποιώντας κάποια τεχνική αναζήτησης. Συνήθως, ο αλγόριθμος που εφαρμόζει αυτήν την τεχνική λαμβάνει ως είσοδο την ακολουθία εκπαίδευσης, η οποία μπορεί να διαιρείται σε υποακολουθίες (δείγματα) για εκπαίδευση (training) και επικύρωση (validation), και ένα σύνολο προκαθορισμένων παραμέτρων. Σε κάθε επανάληψη, ο αλγόριθμος αναζητά τις τιμές των παραμέτρων που ελαχιστοποιούν το προγνωστικό σφάλμα του μοντέλου.
3. Το τρίτο βήμα δημιουργεί το μοντέλο με τις τιμές των παραμέτρων που βρέθηκαν προηγουμένως και τα εφαρμόζει στα δεδομένα της ακολουθίας εκπαίδευσης. Αυτό το μοντέλο στη συνέχεια παρεκτείνεται, στο τέταρτο βήμα, για τις περιόδους της ακολουθίας δοκιμής. Προφανώς, το σφάλμα πρόβλεψης του μοντέλου αντικατοπτρίζει τις επιλεγμένες τιμές για τις παραμέτρους. Τέτοιο σφάλμα μπορεί να ενισχυθεί για μεγάλα χρονικά ορίζοντα.
4. Το τέταρτο βήμα επιλέγει επίσης τη στρατηγική για την πρόβλεψη των τιμών μιας χρονοσειράς αρκετών περιόδων μπροστά (ορίζοντας πρόβλεψης $h > 1$). Η πιο διαισθητική στρατηγική είναι γνωστή ως αναδρομική (multi-step ή recursive), όπου η πρόβλεψη του $h > 1$ διεξάγεται h διαδοχικά φορές λαμβάνοντας υπόψη ένα μοντέλο πρόβλεψης με $h = 1$. Μετά την παρέκταση του μοντέλου, η προβλεπόμενη τιμή ή η αντίστοιχη πραγματική τιμή μπορεί να χρησιμοποιηθεί για τον υπολογισμό της επόμενης πρόβλεψης.
5. Το πέμπτο βήμα συγκρίνει τις προβλεπόμενες τιμές με την ακολουθία δοκιμής για να μετρήσει την ακρίβεια του μοντέλου. Η ανάλυση της απόδοσης είναι απαραίτητη, δεδομένου ότι διαφορετικά μοντέλα μπορεί να έχουν παρόμοιες προσαρμογές, αλλά έχουν ως αποτέλεσμα σημαντικά διαφορετικές τιμές πρόβλεψης.

6. Το έκτο βήμα κάνει προβλέψεις για μελλοντικές περιόδους των χρονοσειρών. Αυτό το βήμα πρέπει να παρακολουθεί το σφάλμα πρόβλεψης μόλις γνωστοποιηθούν οι πραγματικές τιμές της σειράς. Αυτή η παρακολούθηση στοχεύει να δείξει πότε είναι απαραίτητο να ενημερωθεί το μοντέλο με νέα δεδομένα ή να αναπροσαρμοστούν οι παράμετροί του επειδή η διανομή του από τα πιο πρόσφατα δεδομένα διαφέρει από τα παλιά δεδομένα.



Εικόνα 3.1 Η διαδικασία πρόβλεψης χρονοσειρών

3.2.2 Convolutional Neural Networks

Τα συνελικτικά δίκτυα, επίσης γνωστά ως συνελικτικά νευρωνικά δίκτυα (Convolutional Neural Networks - CNN), είναι ένα εξειδικευμένο είδος νευρωνικού δικτύου για την επεξεργασία δεδομένων που έχει μια πλεγματοειδή τοπολογία. Τα παραδείγματα που χρησιμοποιούν CNNs περιλαμβάνουν δεδομένα χρονοσειρών, τα οποία μπορούν να θεωρηθούν ως πλέγμα 1D που λαμβάνει δείγματα σε κανονικά χρονικά διαστήματα και δεδομένα εικόνας, τα οποία μπορούν να θεωρηθούν ως πλέγμα 2D pixel. Τα συνελικτικά δίκτυα είναι εξαιρετικά επιτυχημένα σε πρακτικές εφαρμογές. Το όνομα «συνελικτικό νευρωνικό δίκτυο» υποδηλώνει ότι το δίκτυο χρησιμοποιεί μια μαθηματική λειτουργία που ονομάζεται συνέλιξη. Η συνέλιξη είναι ένα εξειδικευμένο είδος γραμμικής λειτουργίας. Τα συνελικτικά δίκτυα είναι απλά νευρωνικά δίκτυα που χρησιμοποιούν συνέλιξη στη θέση του γενικού πίνακα πολλαπλασιασμού σε τουλάχιστον ένα από τα επίπεδά τους.

Στην πιο γενική του μορφή, η συνέλιξη (convolution) είναι μία λειτουργία σε δύο συναρτήσεις ενός πραγματικού ορίσματος. Έστω πως ένας διαστημικός οργανισμός παρακολουθεί τη θέση ενός διαστημικού σκάφους με έναν αισθητήρα λέιζερ. Ο αισθητήρας λέιζερ παρέχει μία μόνο έξοδο $x(t)$, τη θέση του διαστημοπλοίου στο χρόνο t. Τόσο το x όσο και το t έχουν πραγματική αξία, δηλαδή, είναι δυνατό να δίνεται διαφορετικό αποτέλεσμα θέσης από τον αισθητήρα λέιζερ ανά πάσα στιγμή. Έστω τώρα ότι ο αισθητήρας λέιζερ είναι κάπως θορυβώδης. Για να ληφθεί μια λιγότερο θορυβώδη εκτίμηση της θέσης του διαστημόπλοιου, ο διαστημικός οργανισμός θα πρέπει να μετρήσει μαζί πολλές μετρήσεις. Φυσικά, οι πιο πρόσφατες μετρήσεις είναι πιο σχετικές, οπότε θα πρέπει να οριστεί ένας σταθμισμένος μέσος όρος που δίνει περισσότερο μεγαλύτερο βάρος στις πρόσφατες μετρήσεις. Αυτό μπορεί να πραγματοποιηθεί με μία συνάρτηση στάθμισης $w(a)$, όπου a είναι η ηλικία μιας μέτρησης. Εάν εφαρμοστεί μία τόσο σταθμισμένη μέση διαδικασία κάθε στιγμή, λαμβάνεται μια νέα συνάρτηση s που παρέχει μια ομαλή εκτίμηση της θέσης του διαστημικού σκάφους:

$$s(t) = \int x(a)w(t-a)da$$

Αυτή η διαδικασία ονομάζεται συνέλιξη. Η λειτουργία συνέλιξης χαρακτηρίζεται συνήθως με αστερίσκο:

$$s(t) = (x * w)(t)$$

Στο παραπάνω παράδειγμα, το w πρέπει να είναι μία έγκυρη συνάρτηση πυκνότητας πιθανότητας, ή η έξοδος δεν είναι ένας σταθμισμένος μέσος όρος. Επίσης, το w πρέπει να είναι 0 για όλα τα αρνητικά ορίσματα, διαφορετικά θα κοιτάξει το μέλλον, το οποίο πιθανώς υπερβαίνει τις δυνατότητές του. Σε γενικές γραμμές, η συνέλιξη ορίζεται για οποιεσδήποτε συναρτήσεις για τις οποίες ορίζεται το παραπάνω ολοκλήρωμα και μπορεί να χρησιμοποιηθεί για άλλους σκοπούς εκτός από τη λήψη σταθμισμένων μέσων όρων.

Στην ορολογία του συνελικτικού δικτύου, το πρώτο όρισμα (σε αυτό το παράδειγμα, η συνάρτηση x) στη συνέλιξη αναφέρεται συχνά ως η είσοδος και το δεύτερο όρισμα (σε αυτό το παράδειγμα, η συνάρτηση w) ως πυρήνας (kernel). Η έξοδος αναφέρεται μερικές φορές ως χάρτης χαρακτηριστικών (feature map).

Έστω τώρα ότι τα x και w ορίζονται μόνο σε ακέραιο t, μπορεί να οριστεί η διακριτή συνέλιξη:

$$s(t) = (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a)$$

Στις εφαρμογές μηχανικής μάθησης, η είσοδος είναι συνήθως ένας πολυδιάστατος πίνακας δεδομένων και ο πυρήνας είναι συνήθως ένας πολυδιάστατος πίνακας παραμέτρων που προσαρμόζονται από τον αλγόριθμο μάθησης. Αυτές οι πολυδιάστατες συστοιχίες θα αναφέρονται ως τανυστές (tensors). Επειδή κάθε στοιχείο της εισόδου και του πυρήνα πρέπει να αποθηκεύεται ρητά ξεχωριστά, συνήθως γίνεται η υπόθεση ότι αυτές οι συναρτήσεις είναι μηδενικές παντού, εκτός από το πεπερασμένο σύνολο σημείων για το οποίο αποθηκεύονται οι τιμές. Αυτό σημαίνει ότι στην πράξη μπορεί να εφαρμοστεί το άπειρο σύνολο ως άθροισμα πάνω από έναν πεπερασμένο αριθμό στοιχείων πίνακα.

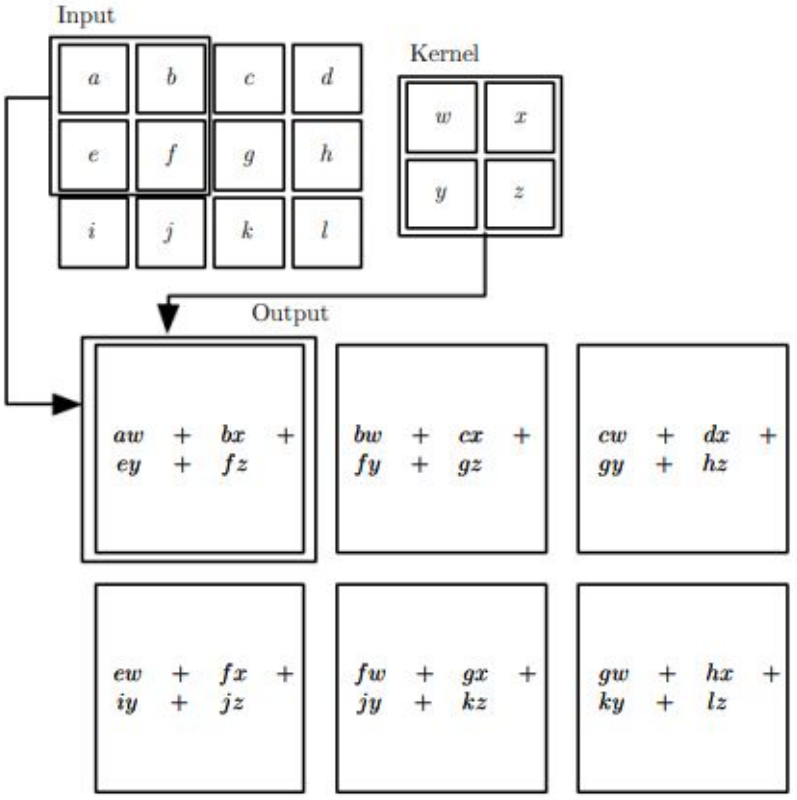
Τέλος, συχνά οι συνελίξεις χρησιμοποιούνται σε περισσότερους από έναν άξονες κάθε φορά. Για παράδειγμα, εάν χρησιμοποιείται μία δισδιάστατη εικόνα I ως είσοδο, πιθανώς να πρέπει να χρησιμοποιηθεί επίσης ένας δισδιάστατος πυρήνας K:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n).$$

Η συνέλιξη ικανοποιεί την αντιμεταθετική ιδιότητα, δηλαδή:

$$S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i-m, j-n)K(m,n).$$

Το παρακάτω σχήμα είναι ένα παράδειγμα συνέλιξης που εφαρμόζεται σε έναν τανυστή 2-D [37].



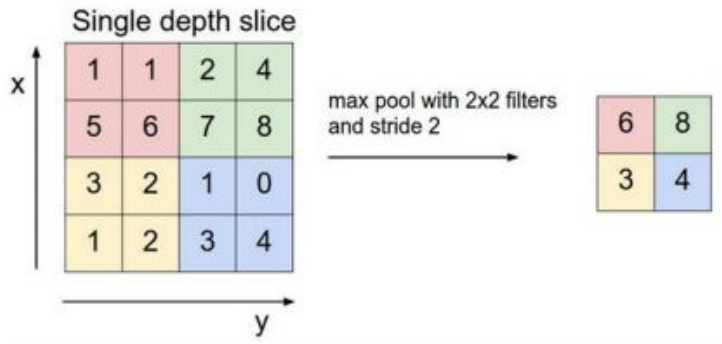
Εικόνα 3.2 Ένα παράδειγμα 2-D συνέλιξης. Σε αυτήν την περίπτωση περιορίζεται η έξοδος μόνο σε θέσεις όπου ο πυρήνας βρίσκεται εντελώς μέσα στην εικόνα, που ονομάζεται «έγκυρη» συνέλιξη σε ορισμένα περιβάλλοντα. Σχεδιάζονται κουτιά με βέλη για να επεξηγηθεί πώς σχηματίζεται το άνω αριστερό στοιχείο του τανυστή εξόδου εφαρμόζοντας τον πυρήνα στην αντίστοιχη άνω αριστερή περιοχή του τανυστή εισόδου.

Υπάρχουν τέσσερις βασικές λειτουργίες σε κάθε CNN, και είναι οι ακόλουθες:

- ❏ Συνέλιξη
- ❏ Με γραμμικότητα (ReLU)
- ❏ Pooling ή υποδειγματοληψία
- ❏ Ταξινόμηση / Παλινδρόμηση (fully connected layers)

Ορίστηκε προηγουμένως ο ορισμός της συνέλιξης (convolution), και πώς χρησιμοποιείται στα πλαίσια ενός αλγορίθμου εκμάθησης. Αναφέρθηκε ότι η είσοδος θα έχει το ρόλο της μίας συνάρτησης και κάποιο φίλτρο ή πυρήνας, που θα διαμορφώνεται από τον αλγόριθμο, θα έχει το ρόλο της δεύτερης. Ο χρήστης παρέχει τη βασική αρχιτεκτονική, όπως το πλήθος των φίλτρων και τη διάσταση του καθενός, και αφήνει τον αλγόριθμο να προσδιορίσει ο ίδιος τις τιμές των παραμέτρων τους, μέσω της εκπαίδευσης. Αντί να ορίζονται εξαρχής φίλτρα που ο χρήστης θεωρεί ότι θα συμβάλλουν στην αναγνώριση μιας εικόνας και στη σωστή ταξινόμησή της, διαμορφώνει το πλαίσιο ώστε αυτά να αυτο-προσδιοριστούν με κριτήριο την ελαχιστοποίηση της συνάρτησης κόστους. Αυτή είναι η βασική ιδέα πίσω από τα Συνελικτικά νευρωνικά δίκτυα. Συχνά η διαδικασία προσδιορισμού των φίλτρων, στα πρώτα στρώματα ενός συνελικτικού δικτύου, αναφέρεται με τον όρο ανίχνευση χαρακτηριστικών (feature detection).

Εφαρμόζοντας υποδειγματοληψία (ή αλλιώς pooling) μειώνεται η διάσταση κάθε feature map διατηρώντας παράλληλα τις σημαντικότερες πληροφορίες. Υπάρχουν διάφοροι τρόποι να το επιτευχθεί αυτό, αλλά ο πιο συχνός είναι το max pooling κατά το οποίο ορίζεται μία γειτονιά από στοιχεία (για παράδειγμα ένα παράθυρο 2x2) από την οποία επικρατεί η μεγαλύτερη τιμή από τα στοιχεία της γειτονιάς αυτής. Ορίζεται, επίσης, το βήμα (stride) για τη μετάβαση από τη μία γειτονιά στην επόμενη. Στην εικόνα φαίνεται ένα παράδειγμα της διαδικασίας max pooling:

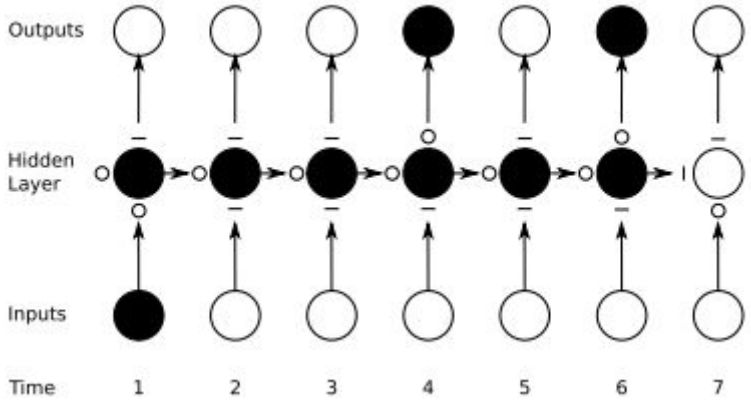


Εικόνα 3.3 Max pooling 2x2 με stride 2

3.2.3 Long Short-Term Memory

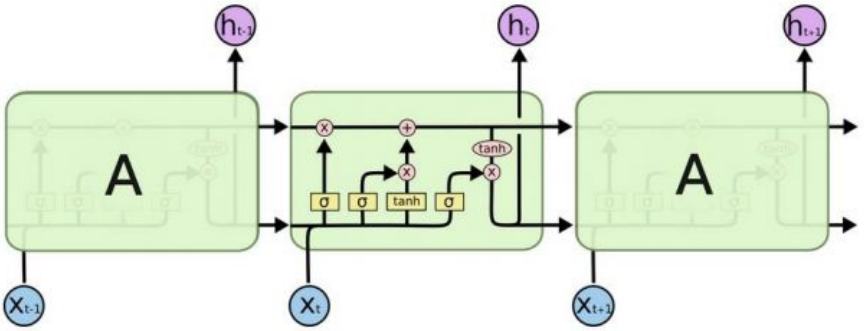
Η χρήση ενός αναδρομικού νευρωνικού δικτύου, είναι μια διαδεδομένη επιλογή για την πρόβλεψη μίας χρονοσειράς. Όπως ειπώθηκε και σε προηγούμενη ενότητα, η δομή των αναδρομικών νευρωνικών δικτύων είναι τέτοια, λόγο των διασυνδέσεων αναδρομής, που έχουν ικανότητα «μνήμης» και αυτό τα καθιστά ειδικά για τα προβλήματα πρόβλεψης χρονοσειράς. Ένα πολύ δημοφιλές νευρωνικό που ανήκει στην κατηγορία των αναδρομικών νευρωνικών δικτύων, είναι το Long Short Term Memory (LSTM).

Πιο συγκεκριμένα, [69] κάθε μονάδα των LSTM αποτελείται από ένα ή περισσότερα κελιά μνήμης που συνδέονται μεταξύ τους και τρία ακόμη στοιχεία, τις πύλες εισόδου, εξόδου και επιλεκτικής συγκράτησης ή πύλη λήθης (forget gate), οι οποίες είναι αντιστοίχως υπεύθυνες για τις λειτουργίες εγγραφής, ανάγνωσης και επαναφοράς των κελιών. Η χρήση αυτών των πυλών διασφαλίζει την αποθήκευση και πρόσβαση στις πληροφορίες ακόμα και με την πάροδο μεγάλων χρονικών περιόδων ή πολλών βημάτων. Στη Εικόνα 3.4 φαίνεται η διατήρηση της πληροφορίας του βήματος 1 με την πάροδο του χρόνου σε ένα LSTM με ένα κρυφό επίπεδο. Ο συμβολισμός “O” και “-” σημαίνει ότι η εκάστοτε πύλη είναι αντίστοιχα ανοιχτή ή κλειστή. Παρατηρείται, λοιπόν, ότι η μονάδα μνήμης είναι σε θέση να συγκρατήσει την πληροφορία του πρώτου βήματος, εφόσον η πύλη εισόδου είναι κλειστή και αυτή της επιλεκτικής συγκράτησης ανοιχτή. Για απλούστευση του παραδείγματος, οι πύλες είναι είτε πλήρως ανοιχτές (1), είτε κλειστές (0).



Εικόνα 3.4 Διατήρηση της πληροφορίας στα δίκτυα LSTM

Στην εικόνα 3.5 φαίνονται τα δομικά στοιχεία των μονάδων μνήμης των LSTM, καθώς και οι συνδέσεις μεταξύ αυτών. Κάθε γραμμή περιέχει ένα διάνυσμα που μεταφέρεται από την έξοδο ενός block στις εισόδους των επόμενων. Οι γραμμές που ενώνονται καταδεικνύουν τις συγχωνεύσεις και αυτές που διακλαδώνονται περιέχουν αντίγραφα της ίδιας πληροφορίας. Οι ροζ κύκλοι αφορούν τις πράξεις μεταξύ των διανυσμάτων και τα κίτρινα πλαίσια είναι διακριτά επίπεδα αναδρομικών δικτύων που χρησιμοποιούνται στην εκπαίδευση των LSTM και περιλαμβάνουν κάποιες συναρτήσεις, όπως η σιγμοειδής και η υπερβολική εφαπτομένη.



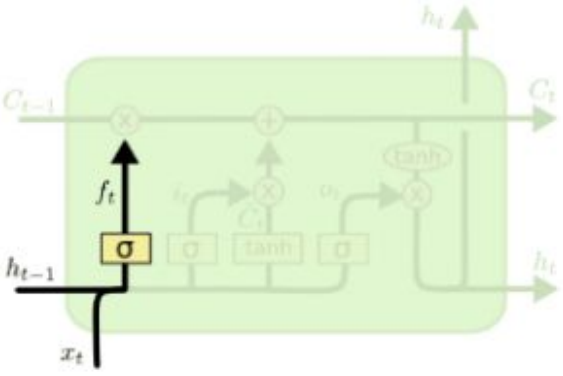
Εικόνα 3.5 Διαγραμματικό ανάπτυγμα του βασικού δομικού στοιχείου του LSTM

Το βασικό στοιχείο των LSTM είναι η οριζόντια γραμμή που φαίνεται να διασχίζει το επάνω μέρος του διαγράμματος και απεικονίζει την κατάσταση των μονάδων της μνήμης τους. Ονομάζεται κύτταρο κατάστασης και μπορεί να τη συγκρίνει κανείς με ένα ιμάντα μεταφοράς που διασχίζει ολόκληρη την αλυσίδα των blocks και δεν έχει παρά μόνο λίγες γραμμικές αλληλεπιδράσεις με τα υπόλοιπα στοιχεία. Είναι πολύ εύκολο, δηλαδή, η περιεχόμενη πληροφορία να περάσει αναλλοίωτη. Η αλληλεπίδραση με τα υπόλοιπα στοιχεία γίνεται μέσω των πυλών που αναφέρθηκε προηγουμένως. Αυτές αποτελούνται από μία σιγμοειδή συνάρτηση, που παίρνει τιμές από 0 εώς 1 και μια πράξη πολλαπλασιασμού ή πρόσθεσης που αναλαμβάνει να προσθέσει την πληροφορία στις ήδη υπάρχουσες της μονάδας μνήμης. Η τιμή της σιγμοειδούς συνάρτησης καθορίζει το ποσοστό της πληροφορίας που θα περάσει για να προστεθεί στη μνήμη, με 0 να σημαίνει ότι δε θα επιτρέψει σε τίποτα να περάσει και με 1 η πληροφορία θα προσχωρήσει αυτούσια.

Το πρώτο βήμα στη λειτουργία του LSTM είναι να "αποφασίσει" ποιο μέρος της πληροφορίας θα αποδεσμεύσει από τη μνήμη. Υπεύθυνο για την απόφαση αυτή είναι η πύλη λήθης του συστήματος, η οποία δέχεται την έξοδο του προηγούμενου επιπέδου h_{t-1} και την είσοδο x_t για να εξάγει μια τιμή 0-1, μέσω της σιγμοειδούς. Η συνάρτηση που διέπει αυτή τη σχέση είναι η παρακάτω:

$$f_t = (W_f[h_{t-1}, x_t] + b_f)$$

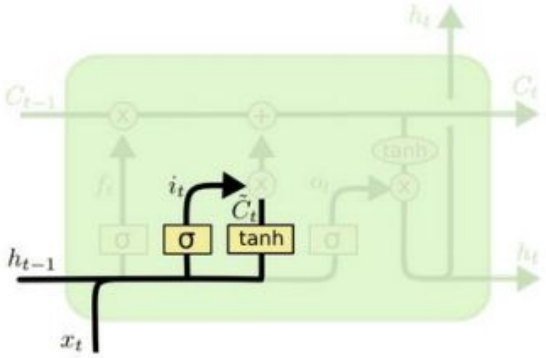
όπου W_f και b_f ο πίνακας των βαρών και το διάνυσμα των bias όρων του σιγμοειδούς στρώματος νευρώνων αντίστοιχα. Τα στοιχεία του δικτύου που είναι υπεύθυνα για αυτή τη λειτουργία φαίνονται στην Εικόνα 3.6.



Εικόνα 3.6 Στρώμα πύλης λήθης του LSTM

Το επόμενο βήμα είναι αυτό που θα πάρει την απόφαση για το ποια στοιχεία της νέας πληροφορίας θα συγκρατηθούν στη μνήμη του δικτύου. Η διαδικασία αυτή απαρτίζεται από δύο βήματα, με το πρώτο να περιλαμβάνει την πύλη εισόδου που ξεχωρίζει ποιες από τις υπάρχουσες

πληροφορίες θα παραμείνουν στη μνήμη και το δεύτερο να δημιουργεί ένα νέο διάνυσμα $\overline{C}_t$ με τις υποψήφιες τιμές που πρόκειται να προστεθούν σε αυτή.



Εικόνα 3.7 Στρώμα πύλης εισόδου και tanh στρώμα του LSTM

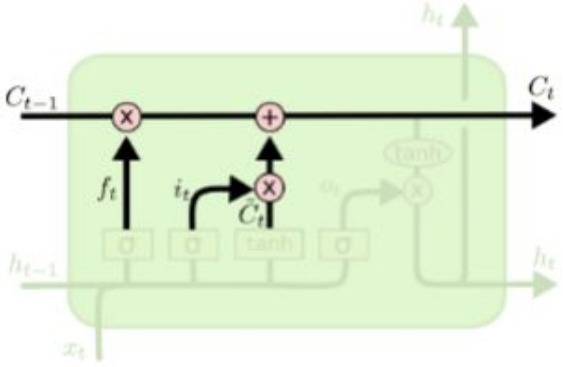
Οι εξισώσεις που περιγράφουν τα δύο στρώματα είναι οι ακόλουθες:

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

$$\overline{C}_t = tanh(W_C[h_{t-1}, x_t] + b_C)$$

Είναι ώρα, λοιπόν, να ανανεωθεί η κατάσταση του κυττάρου από την παλιά, C_{t-1}, στην καινούρια, C_t . Τα προηγούμενα βήματα έχουν ήδη αποφασίσει για το τι θα συμβεί και απομένει η υλοποίησή του.

Πολλαπλασιάζεται η παλιά κατάσταση με f_t , ξεχνώντας τα στοιχεία που αποφασίστηκαν ότι θα ξεχαστούν πρωτύτερα. Έπειτα προστίθεται το γινόμενο $i_t\overline{C}_t$, στις νέες υποψήφιες τιμές δηλαδή κανονικοποιημένες ανάλογα με το πόσο έχει αποφασιστεί από το χρήστη να ανανεώσει κάθε τιμή του κυττάρου κατάστασης.

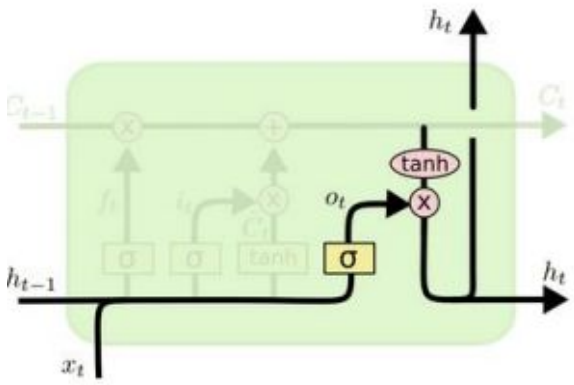


Εικόνα 3.8 Ανανέωση του κυττάρου κατάστασης του LSTM

Η εξίσωση που περιγράφει την ανανέωση του κυττάρου κατάστασης είναι η ακόλουθη:

$$C_t = f_t C_{t-1} + i_t \overline{C}_t$$

Το τελευταίο βήμα αφορά την εξαγωγή του αποτελέσματος h_t, που θα γίνει είσοδος στο επόμενο επίπεδο και αποτελεί μια φιλτραρισμένη εκδοχή της κατάστασης της μνήμης. Αρχικά, γίνεται το πέρασμα της πληροφορίας εισόδου από μια σιγμοειδή συνάρτηση για να προσδιοριστεί ποιο κομμάτι αυτής θα προωθηθεί ως την έξοδο. Έπειτα εισέρχεται το περιεχόμενο της μνήμης μέσω της συνάρτησης της υπερβολικής εφαπτομένης (για να ωθήσει τις τιμές του διανύσματος στο διάστημα -1 έως 1) και πολλαπλασιάζεται με την έξοδο της σιγμοειδούς για να γίνει η προώθηση μόνο των κομματιών που έχουν προσδιοριστεί .



Εικόνα 3.9 Διαμόρφωση της εξόδου του LSTM

Οι εξισώσεις που αναπαριστούν τα παραπάνω είναι οι ακόλουθες:

$$o_t = (W_o[h_{t-1}, x_t] + b_o)$$

$$h_t = o_t tanh(C_t)$$

Όσον αφορά τον αλγόριθμο μάθησης στα LSTM, όπως και στα υπόλοιπα RNN, υπάρχουν πολλές διαθέσιμες επιλογές. Η πιο συνηθισμένη τεχνική είναι ο κατά προσέγγιση υπολογισμός της κλίσης του σφάλματος μέσω συνδυασμού του Real Time Recurrent Learning και του Backpropagation Through Time. Το BPTT χρησιμοποιείται μόνο στους υπολογισμούς που λαμβάνουν χώρα στο πρώτο βήμα της εκπαίδευσης και έπειτα το ρόλο της μνήμης αναλαμβάνουν αποκλειστικά τα memory blocks του LSTM. Το κύριο πλεονέκτημα αυτής της διαδικασίας είναι η σύγχρονη ενημέρωση των βαρών με τη μέθοδο του RTRL, που την καθιστά κατάλληλη για εφαρμογές συνεχούς χρόνου ή για την πρόβλεψη χρονοσειρών. Ωστόσο, είναι δυνατός και ο υπολογισμός της κλίσης χωρίς να χρειαστεί να αφαιρεθεί το κομμάτι του BPTT που παρουσιάζει καλύτερη ακρίβεια και έχει το πλεονέκτημα της ευκολίας αποσφαλμάτωσης σε σχέση με την προηγούμενη τεχνική.

3.2.4 Τεχνικές Εκμάθησης στα Αναδρομικά Νευρωνικά Δίκτυα

Αναμφίβολα, ο πιο δημοφιλής κανόνας εκμάθησης στα νευρωνικά δίκτυα είναι αυτός της οπίσθιας διάδοσης του σφάλματος και έχει κατά συνέπεια επεκταθεί και στα αναδρομικά νευρωνικά δίκτυα με την ονομασία αλγόριθμος οπίσθιας διάδοσης (backpropagation through time - BPTT). Αποτελεί μια γενίκευση του αλγορίθμου ελαχίστων μέσων τετραγώνων (Least Mean Squares - LMS) αλγορίθμου και ονομάζεται, επίσης, γενικευμένος κανόνας δέλτα (generalized delta rule). Χρησιμοποιεί μία τεχνική αναζήτησης της κλίσης (gradient) με στόχο να ελαχιστοποιήσει μια συνάρτηση κόστους που έχει οριστεί για την αξιολόγηση της απόκλισης μεταξύ του επιθυμητού σήματος και της εξόδου του νευρωνικού δικτύου. Η συνάρτηση κόστους που αξιολογείται είναι συνήθως το μέσο τετραγωνικό σφάλμα (mean square error - MSE).

Οπίσθια διάδοση του σφάλματος

Ο αλγόριθμος της οπίσθιας διάδοσης του σφάλματος [24] χρησιμοποιεί, για κάθε ακολουθία εισόδου, την έξοδο του νευρωνικού δικτύου ώστε να τη συγκρίνει με την επιθυμητή ακολουθία εξόδου και έπειτα σχηματίζει το σφάλμα για κάθε κόμβο του επιπέδου εξόδου του δικτύου κάνοντας σύγκριση αυτών των δύο. Ακολούθως, το σφάλμα της εξόδου διαδίδεται προς τα πίσω περνώντας από όλους τους κόμβους του δικτύου σε μία διαδικασία κλειστής επανάληψης. Με βάση αυτό το σφάλμα διορθώνονται τα βάρη του πίνακα W με τη χρήση κάποιου αλγόριθμου μείωσης της κλίσης. Για παράδειγμα ένας δημοφιλής αλγόριθμος μείωσης κλίσης είναι η στοχαστική κατάβαση κλίσης (stochastic gradient descent - SGD) [25]. Προκειμένου να είναι εφικτή η υλοποίηση του αλγορίθμου

οπίσθιας διάδοσης του σφάλματος απαιτείται η χρήση κάποιας συνεχούς, μη γραμμικής, αυξανόμενης μονοτονικά και διαφορίσιμης συνάρτησης ενεργοποίησης (συνηθέστερα σιγμοειδείς συναρτήσεις όπως αυτή της υπερβολικής εφαπτομένης).

Σε αυτό το σημείο θα γίνει μία σύντομη περιγραφή της διαδικασίας που ακολουθεί η εκμάθηση μέσω της οπίσθιας διάδοσης του σφάλματος στα δίκτυα εμπρόσθιας τροφοδότησης. Έστω ένα πολυεπίπεδο perceptron (MLP) με πλήθος k κρυφών επιπέδων. Μαζί με το επίπεδο εισόδου και αυτό της εξόδου υπάρχουν συνολικά $k+2$ επίπεδα στο δίκτυο, τα οποία αριθμούνται από 0 ως $k+1$. Ο αριθμός των κόμβων εισόδου είναι K, των κόμβων εξόδου L και των κόμβων των εσωτερικών επιπέδων 1,…, είναι N. Η τιμή που έχει το βάρος της σύνδεσης του i κόμβου στο m επίπεδο και του κόμβου j στο επίπεδο $m+1$ συμβολίζεται με $w^m(i,j)$. Η ενεργοποίηση του i κόμβου στο m επίπεδο είναι $x^m(i)$, για $m=0$ αποτελεί κόμβο εισόδου και για $m=k+1$ κόμβο εξόδου.

Τα δεδομένα που χρησιμοποιούνται για την εκπαίδευση του δικτύου αποτελούνται από ζεύγη εισόδου-εξόδου.

$$u(n) = \left(x_1^0(n), ..., x_K^0\right)^t$$

$$d(n) = \left(d_1^{k+1}(n), ..., d_L^{k+1}\right)^t$$

όπου $u(n)$ τα δεδομένα εισόδου και $d(n)$ τα δεδομένα εξόδου. Η παράμετρος n αναφέρεται στον αύξοντα αριθμό του δείγματος και όχι στο χρόνο. Ακολουθεί η ενεργοποίηση όλων των κόμβων του δικτύου, εκτός από αυτούς της εισόδου, σύμφωνα με τον τύπο:

$$x_i^{m+1}(n) = f\left(\sum_{j=1}^{N^m} w_{ij}^m x_i(n)\right)$$

Πολλές φορές ο όρος πόλωσης (bias) (θ_j^m), ο οποίος λειτουργεί αθροιστικά στον παραπάνω τύπο, παραλείπεται σε αυτή την προσπάθεια περιγραφής του αλγορίθμου. Με αυτή τη μέθοδο και με δεδομένα τα στοιχεία εισόδου γίνεται η ενεργοποίηση όλων των κόμβων που βρίσκονται στα εσωτερικά επίπεδα, μέχρι τα στοιχεία του τελευταίου επιπέδου (εξόδου) που έχουν τη μορφή:

$$y(n) = \left(x_1^{k+1}(n), ..., x_L^{k+1}\right)^t$$

Ο σκοπός της εκπαίδευσης του δικτύου είναι η εύρεση ενός συνόλου βαρών που να ελαχιστοποιεί το μέσο τετραγωνικό σφάλμα μεταξύ της εξόδου του νευρωνικού δικτύου και της επιθυμητής εξόδου $d(n)$.

$$E = \sum_{n=1}^{T} ||d(n) - y(n)||^2 = \sum_{n=1}^{T} E(n)$$

Αυτό επιτυγχάνεται προσαρμόζοντας ανάλογα τα βάρη των συνδέσεων προς την αντίθετη κατεύθυνση της κλίσης του σφάλματος, με τη χρήση μιας σταθεράς εκμάθησης γ (learning rate) που έχει συνήθως κάποια μικρή τιμή.

$$\frac{\partial \mathrm{E}}{\partial w_{ij}^m} = \sum_{t=1,...,T} \frac{\partial \mathrm{E}(n)}{\partial w_{ij}^m}$$

$$\text{νέο } w_{ij}^m = w_{ij}^m - \gamma \frac{\partial \mathrm{E}}{\partial w_{ij}^m}$$

Οι παραπάνω τύποι αναφέρονται στη μέθοδο της ομαδικής εκμάθησης (batch learning). Η ομαδική εκμάθηση έχει σκοπό τη βελτιστοποίηση (ελαχιστοποίηση) του σφάλματος, όπως αναφέρθηκε παραπάνω, με τις προσαυξήσεις των βαρών να συσσωρεύονται μέχρι να γίνει η τελική τροποποίησή τους στο τέλος κάθε εποχής. Υπάρχει, επίσης και η αυξητική εκμάθηση (incremental learning), όπου η μεταβολή των βαρών γίνεται σταδιακά στο τέλος κάθε βήματος του αλγορίθμου εκμάθησης. Η αυξητική εκμάθηση είναι μια στοχαστική μέθοδος βελτιστοποίησης των παραμέτρων του δικτύου, ενώ η ομαδική εκμάθηση ντετερμινιστική. Στην περίπτωση της ομαδικής εκμάθησης, οι παραπάνω τύποι αντικαθίστανται από τον εξής

$$\text{νέο } w_{ij}^{m} = w_{ij}^{m} - \gamma \frac{\partial \mathrm{E}}{\partial w_{ij}^{m}}$$

Ο αλγόριθμος της οπίσθιας διάδοσης του σφάλματος κάνει επαναληπτικά την εφαρμογή των παραπάνω σχέσεων για κάθε εποχή μέχρι να επιτύχει μείωση του σφάλματος σε τιμή μικρότερη από κάποιο προκαθορισμένο κατώφλι (threshold) ή μέχρι η μεταβολή του σφάλματος να πέσει κάτω από ένα άλλο προκαθορισμένο κατώφλι ή μέχρι να ολοκληρωθεί ο αριθμός των εποχών που έχει οριστεί στην αρχή της εκπαίδευσης. Συνήθως απαιτείται κάποιος αρκετά μεγάλος αριθμός εποχών για να εκπληρωθεί κάποιος από τους παραπάνω περιορισμούς.

Η κάθε εποχή έχει πολυπλοκότητα $O(TM)$, όπου είναι ο συνολικός αριθμός των συνδέσεων μεταξύ των κόμβων του δικτύου. Η προσέγγιση μέσω της μεθόδου κατάβασης κλίσης και του αλγορίθμου οπίσθιας διάδοσης του σφάλματος έχει το μειονέκτημα της αργής σύγκλισης, διότι τυπικά επιλέγεται κάποια μικρή σταθερά εκμάθησης για να αποφευχθεί η αποσταθεροποίηση του δικτύου. Υπάρχουν, όμως, τρόποι για την επίσπευση της σύγκλισης, όπως η χρήση δευτεροβάθμιων τεχνικών κατάβασης κλίσης που εκμεταλλεύονται την καμπυλότητα της κλίσης του σφάλματος, αλλά έχουν πολυπλοκότητα $O(TM^2)$. Ένα ακόμη αρνητικό που έχουν οι τεχνικές κατάβασης κλίσης είναι ότι γίνεται αναζήτηση μόνο του τοπικού ελαχίστου του σφάλματος. Το πρόβλημα αυτό μπορεί να αντιμετωπιστεί με διάφορους τρόπους, όπως με την προσθήκη θορύβου κατά την εκμάθηση του δικτύου, την επανάληψη όλης της διαδικασίας με διαφορετική αρχικοποίηση των βαρών ή με τη χρήση υπαρχόντων πληροφοριών των δεδομένων εισόδου - εξόδου που μπορούν να φανούν χρήσιμες κατά την εκπαίδευση.

3.2.5 Backpropagation Through Time (BPTT)

Ο αλγόριθμος της οπίσθιας διάδοσης του σφάλματος που χρησιμοποιείται στα δίκτυα εμπρόσθιας τροφοδότησης δεν μπορεί να μεταφερθεί ως έχει στα αναδρομικά, επειδή προϋποθέτει την ύπαρξη αποκλειστικά ακυκλικών συνδέσεων μεταξύ των κόμβων του δικτύου. Η λύση δίνεται με την εισαγωγή του αλγόριθμου backpropagation through time (BPTT) [26], που «ξεδιπλώνει» τις συνδέσεις μεταξύ των κόμβων σε ξεχωριστά χρονικά βήματα, δημιουργώντας πανομοιότυπα αντίγραφα και ανακατευθύνει τις συνδέσεις μεταξύ αυτών ώστε να προκύψει ένα δίκτυο εμπρόσθιας τροφοδότησης.

Οι πίνακες των βαρών του συστήματος $w_{ij}^{in}, w_{ij}, w_{ij}^{out}, w_{ij}^{back}$ παραμένουν οι ίδιοι σε όλα τα αντίγραφα των επιπέδων. Τα δεδομένα αποτελούνται από μια χρονοσειρά δειγμάτων εισόδου-εξόδου που έχουν την παρακάτω μορφή:

$$u(n) = \left(u_1(n), ..., u_K(n)\right)^t$$

$$d(n) = (d_1(n), ..., d_L(n))^t$$

Η διαδικασία της εκμάθησης ξεκινάει από το πρώτο επίπεδο και προχωράει σταδιακά στα επόμενα επίπεδα της στοίβας που έχει δημιουργηθεί από το ξεδίπλωμα των επιπέδων στο χρόνο. Σε κάθε αντίγραφο των επιπέδων τη χρονική στιγμή n διαβάζεται η είσοδος $u(n)$, υπολογίζεται το $x(n)$ των

ενδιάμεσων επιπέδων με βάση τα $u(n)$, $x(n-1)$ και $y(n-1)$ (όταν το τελευταίο δεν είναι θ) και τέλος υπολογίζεται η έξοδος $y(n)$. Η συνάρτηση του σφάλματος που ελαχιστοποιείται είναι και πάλι:

$$E = \sum_{n=1}^{T} ||d(n) - y(n)||^2 = \sum_{n=1}^{T} E(n)$$

με τη διαφορά ότι η έννοια του t έχει μετατραπεί από τον αύξοντα αριθμό του δείγματος εκπαίδευσης σε χρονική στιγμή.

Ο αλγόριθμος που ακολουθεί το BPTT έχει ως εξής:

- Είσοδος: η χρονοσειρά των δεδομένων εκπαίδευσης και τα βάρη w_{ij} που αντιστοιχούν στην συγκεκριμένη χρονική στιγμή.
- Έξοδος: τα νέα βάρη των συνδέσεων.

Υπολογιστικά βήματα του αλγορίθμου:

- ➢ Εμπρόσθιο πέρασμα, όπως περιγράφηκε παραπάνω μέχρι την έξοδο $y(n)$.
- ➢ Υπολογισμός από το τέλος προς την αρχή των επιπέδων (για $n = T, ..., 1$) και για την κάθε ενεργοποίηση των κόμβων $x_i(n), y_i(n)$ ενός όρου της διάδοσης του σφάλματος $\delta_i(n)$, ο οποίος δίνεται από τους τύπους:

$$\delta_j(T) = (d_j(T) - y_j(T))\frac{\partial f(u)}{\partial u}|_{u=z_j(n)}$$

για τους κόμβους εξόδου στο χρονικό επίπεδο,

$$\delta_i(T) = \left[\sum_{j=1}^{L} \delta_j(T) w_{ji}^{out}\right] \frac{\partial f(u)}{\partial u}|_{u=z_i(n)}$$

για τους εσωτερικούς κόμβους $x_i(T)$ στο χρονικό επίπεδο,

$$\delta_j(n) = \left[(d_j(n) - y_j(n))\left[\sum_{i=1}^{N} \delta_i(n+1) w_{ji}^{back}\right] \frac{\partial f(u)}{\partial u}|_{u=z_j(n)}\right]$$

για τους κόμβους εξόδου των προηγούμενων χρονικών επιπέδων και για τους εσωτερικούς κόμβους των προηγούμενων χρονικών επιπέδων.

$$\delta_i(n) = \left[\sum_{j=1}^{N} \delta_j(n+1) w_{ji}\right] + \left[\sum_{j=1}^{N} \delta_j(n) w_{ji}^{out}\right] \frac{\partial f(u)}{\partial u}|_{u=z_j(n)}$$

Το $z_i(n)$ είναι η μέγιστη τιμή που μπορούν να πάρουν οι συγκεκριμένοι κόμβοι.

- ➢ Εκ νέου υπολογισμός των βαρών σύμφωνα με τις σχέσεις:

$$\text{νέο } w_{ij} = w_{ij} + \gamma \sum_{n=1}^{T} \delta_i(n) x_j(n-1)$$

υποθέτοντας $x_j(n-1) = 1$ για $n = 1$,

$$\text{νέο } w_{ij}^{in} = w_{ij}^{in} + \gamma \sum_{n=1}^{T} \delta_i(n) u_j(n)$$

$$\text{νέο } w_{ij}^{out} = w_{ij}^{out} + \gamma \sum_{n=1}^{T} \delta_i(n) u_j(n), \text{ αν το j είναι κόμβος της εισόδου}$$

$$w_{ij}^{out} + \gamma \sum_{n=1}^{T} \delta_i(n) x_j(n-1)\text{, αν το j είναι εσωτερικός κόμβος}$$

$$\text{νέο } w_{ij}^{back} = w_{ij}^{back} + \gamma \sum_{n=1}^{T} \delta_i(n) y_j(n-1)$$

υποθέτοντας $y_j(n-1) = 1$ για $n = 1$.

Το θέμα της αργής σύγκλισης που αναφέρθηκε για την οπίσθια διάδοση του σφάλματος στα δίκτυα εμπρόσθιας τροφοδότησης εξακολουθεί να υπάρχει και στο BPTT και η πολυπλοκότητα του αλγορίθμου που περιγράφηκε είναι $O(TN^2)$, με τον αριθμό των εσωτερικών κόμβων. Συνήθως, χρειάζονται πολλές εποχές για να ολοκληρωθεί η διαδικασία της εκπαίδευσης. Η συνεχόμενη εκτέλεση αυτών των εποχών έχει ως αποτέλεσμα τη δημιουργία ενός πολύπλοκου δυναμικού συστήματος που συχνά μπορεί να αποκλίνει της επιθυμητής συμπεριφοράς. Επομένως, είναι πιθανό να δημιουργηθούν διακλαδώσεις όταν οι τιμές αρχικοποίησης των βαρών του δικτύου είναι αρκετά διαφορετικές από τη δυναμική του συστήματος που προσπαθεί να μοντελοποιήσει ο χρήστης.

Αποτέλεσμα αυτών των διακλαδώσεων μπορεί να είναι η αλλοίωση των πληροφοριών της κλίσης και η εκτόξευση του σφάλματος σε μη αποδεκτές (πολύ υψηλές) τιμές, το οποίο στην περίπτωση του BPTT δεν εγγυάται τη σύγκλιση σε κάποια κοντινή περιοχή του ελαχίστου του. Τα προβλήματα αυτά δεν συναντώνται στα δίκτυα εμπρόσθιας τροφοδότησης επειδή μοντελοποιούν μόνο απλές συναρτήσεις και όχι δυναμικά συστήματα.

Τέλος, δεν υπάρχει κάποια συγκεκριμένη τεχνική για να υπερκεραστούν αυτά τα προβλήματα και συνήθως χρειάζονται αρκετά πειράματα και υπολογιστικός χρόνος για να επιτευχθεί ένα ικανοποιητικό αποτέλεσμα. Για τους λόγους αυτούς, η τεχνική του BPTT χρησιμοποιείται σχεδόν αποκλειστικά σε μικρά δίκτυα μεγέθους $3 - 20$ κόμβων ανά επίπεδο και η χρήση μεγαλύτερων δικτύων αποδεικνύεται πολύ δαπανηρή από άποψη χρήσης υπολογιστικού εξοπλισμού και χρόνου.

Ένα ακόμη μειονέκτημα που αφορά το ομαδική εκμάθηση που γίνεται στο BPTT, αλλά και την απλή περίπτωση της οπίσθιας διάδοσης του σφάλματος στα δίκτυα εμπρόσθιας τροφοδότησης είναι ότι η μεταβολή των βαρών γίνεται αποκλειστικά στο τέλος κάθε εποχής, μετά από ένα πλήρες πέρασμα των δεδομένων εκμάθησης. Το γεγονός αυτό καθιστά την τεχνική του της οπίσθιας διάδοσης του σφάλματος μη κατάλληλη για εφαρμογές πραγματικού χρόνου που απαιτούν τη συνεχή ενημέρωση των βαρών.

4.Μέθοδος βελτιστοποίησης σμήνους σωματιδίων

4.1 Εισαγωγή στις μεθόδους βελτιστοποίησης

Για τη δημιουργία ενός μοντέλου βαθιάς μάθησης, ο προγραμματιστής πρέπει να λάβει πολλές φαινομενικά αυθαίρετες αποφάσεις: Πόσα επίπεδα πρέπει να στοιβάχθούν; Πόσες μονάδες ή φίλτρα πρέπει να πηγαίνουν σε κάθε επίπεδο; Πρέπει να χρησιμοποιηθεί το relu ως συνάρτηση ενεργοποίησης ή κάποια διαφορετική συνάρτηση; Πρέπει να χρησιμοποιηθεί το Batch Normalization μετά από ένα δεδομένο επίπεδο; Ποιο ποσοστό απόρριψης πρέπει να χρησιμοποιηθεί; Και ούτω καθεξής. Αυτές οι παράμετροι σε επίπεδο αρχιτεκτονικής ονομάζονται υπερπαραμέτροι (hyperparameters) για να ξεχωρίσουν από τις παραμέτρους ενός μοντέλου, οι οποίες εκπαιδεύονται μέσω backpropagation.

Στην πράξη, οι έμπειροι μηχανικοί και ερευνητές μηχανικής μάθησης μαθαίνουν να χτίζουν με την πάροδο του χρόνου διαισθητικά ως προς το τί λειτουργεί και τί όχι όταν πρόκειται για αυτές τις επιλογές - αναπτύσσουν δεξιότητες συντονισμού υπερπαραμέτρων. Αλλά δεν υπάρχουν επίσημοι κανόνες. Εάν κάποιος θέλει να φτάσει στο όριο του τί μπορεί να επιτευχθεί σε μια δεδομένη εργασία, δεν μπορεί να είναι ικανοποιημένος με αυθαίρετες επιλογές που γίνονται διαισθητικά. Οι αρχικές αποφάσεις είναι σχεδόν πάντα μη βέλτιστες, ακόμα κι αν ο χρήστης διακρίνεται από καλή διαίσθηση. Μπορεί να βελτιώσει τις επιλογές του τροποποιώντας τις με το χέρι και επανεκπαιδεύοντας το μοντέλο επανειλημμένα - σε αυτό οι μηχανικοί και οι ερευνητές της μηχανικής μάθησης περνούν τον περισσότερο χρόνο τους. Αλλά δεν πρέπει να είναι η δουλειά του μηχανικού μηχανικής μάθησης να δοκιμάζει υπερπαραμέτρους - αυτό είναι καλύτερα να το υλοποιεί ένα μηχάνημα μέσω ενός αλγορίθμου.

Γι’ αυτό το λόγο, πρέπει να εξερευνηθεί ο χώρος των πιθανών αποφάσεων αυτόματα, συστηματικά, μεθοδικά. Πρέπει να γίνει αναζήτηση στον χώρο της αρχιτεκτονικής και να βρεθούν οι καλύτερες αρχιτεκτονικές με εμπειρικό τρόπο. Αυτός το υλοποιεί ο τομέας της αυτόματης βελτιστοποίησης υπερπαραμέτρων (automatic hyperparameter optimization): είναι ένας ολόκληρος τομέας έρευνας και είναι ιδιαίτερα σημαντικός.

Η διαδικασία βελτιστοποίησης υπερπαραμέτρων μοιάζει συνήθως ως εξής:

1. Επιλέγεται ένα σύνολο υπερπαραμέτρων.
2. Δημιουργείται το αντίστοιχο μοντέλο.
3. Εφαρμόζεται το μοντέλο στα δεδομένα εκπαίδευσης και μετράται η τελική απόδοση στα δεδομένα επικύρωσης.
4. Επιλέγεται το επόμενο σετ υπερπαραμέτρων που θα δοκιμαστεί (αυτόματα).
5. Επαναλαμβάνεται η παραπάνω διαδικασία.
6. Τελικά, μετριέται η απόδοση στα δεδομένα δοκιμής.

Το κλειδί για αυτήν τη διαδικασία είναι ο αλγόριθμος που χρησιμοποιεί αυτό το ιστορικό της απόδοσης επικύρωσης, λαμβάνοντας υπόψη διάφορους συνδυασμούς υπερπαραμέτρων, για να επιλέξει το επόμενο σύνολο υπερπαραμέτρων που θα αξιολογηθεί. Πολλές διαφορετικές τεχνικές είναι δυνατές: μπεϊζιανή βελτιστοποίηση, γενετικοί αλγόριθμοι, αλγόριθμος σμήνους σωματιδίων, απλή τυχαία αναζήτηση και ούτω καθεξής.

Η εκπαίδευση των βαρών ενός μοντέλου είναι σχετικά εύκολη: υπολογίζεται μια συνάρτηση απώλειας (loss function) σε μια μικρή παρτίδα δεδομένων και, στη συνέχεια, χρησιμοποιείται ο αλγόριθμος Backpropagation για να μετακινηθούν τα βάρη στη σωστή κατεύθυνση. Η ενημέρωση υπερπαραμέτρων, από την άλλη πλευρά, είναι εξαιρετικά δύσκολη. Λαμβάνοντας υπόψιν τα ακόλουθα:

- Ο υπολογισμός του σήματος ανατροφοδότησης μπορεί να είναι εξαιρετικά ακριβός: απαιτεί τη δημιουργία και την εκπαίδευση ενός νέου μοντέλου από το μηδέν στο σύνολο δεδομένων.
- Ο χώρος υπερπαραμέτρων αποτελείται συνήθως από διακριτές αποφάσεις και συνεπώς δεν είναι συνεχής ή διαφοροποιημένος. Ως εκ τούτου, συνήθως ο χρήστης δεν μπορεί να

χρησιμοποιήσει την κατάβαση κλίσης σε χώρο υπερπαραμέτρων. Αντ 'αυτού, πρέπει να βασίζεται σε τεχνικές βελτιστοποίησης χωρίς ντεγκραντέ, οι οποίες φυσικά είναι πολύ λιγότερο αποτελεσματικές από την κατάβαση κλίσης.

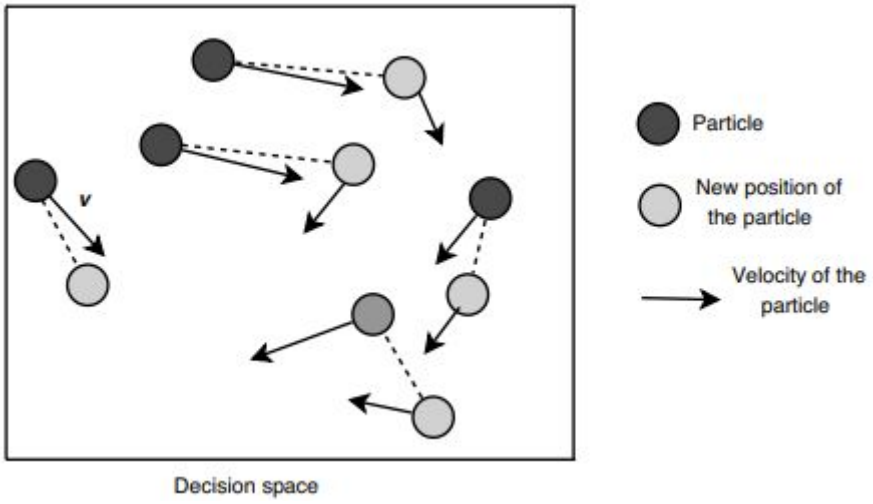
Συνολικά, η βελτιστοποίηση υπερπαραμέτρων είναι μια ισχυρή τεχνική που απαιτείται για να φτάσει κανείς σε υπερσύγχρονα μοντέλα σε οποιαδήποτε εργασία. Μπορεί να σκεφτεί κανείς πως κάποτε, οι άνθρωποι δημιούργησαν τα χαρακτηριστικά που οδήγησαν σε ρηχά μοντέλα μηχανικής μάθησης. Αυτό ήταν το βέλτιστο για εκείνη την εποχή. Τώρα, η βαθιά μάθηση αυτοματοποιεί το καθήκον της ιεραρχικής μηχανικής χαρακτηριστικών - οι συναρτήσεις εκπαιδεύονται χρησιμοποιώντας ένα σήμα ανατροφοδότησης και όχι με χειροκίνητο συντονισμό. [20].

4.2 Μέθοδος βελτιστοποίησης σμήνους σωματιδίων

Η τεχνητή νοημοσύνη (Artificial Intelligence - AI) είναι η νοημοσύνη που υλοποιείται από μηχανές. Ορίζεται ως «η μελέτη και ο σχεδιασμός ευφυών πρακτόρων» [28], όπου ένας έξυπνος πράκτορας αντιπροσωπεύει ένα σύστημα που αντιλαμβάνεται το περιβάλλον του και αναλαμβάνει δράση για να μεγιστοποιήσει την πιθανότητα επιτυχίας του. Η έρευνα για την τεχνητή νοημοσύνη είναι εξαιρετικά τεχνική και εξειδικευμένη και χωρίζεται βαθιά σε υποπεδία που συχνά αποτυγχάνουν να επικοινωνούν μεταξύ τους. Οι δημοφιλείς προσεγγίσεις της AI περιλαμβάνουν σήμερα παραδοσιακές στατιστικές μεθόδους [29], παραδοσιακή συμβολική AI και υπολογιστική νοημοσύνη (Computational Intelligence - CI) [30]. Η CI είναι ένας αρκετά νέος ερευνητικός τομέας. Είναι ένα σύνολο υπολογιστικών μεθοδολογιών και προσεγγίσεων εμπνευσμένων από τη φύση για την αντιμετώπιση σύνθετων πραγματικών προβλημάτων στα οποία οι παραδοσιακές προσεγγίσεις είναι αναποτελεσματικές ή ανέφικτες. Η CI περιλαμβάνει τεχνητά νευρωνικά δίκτυα (ANN), την ασαφή λογική και τον εξελικτικό υπολογισμό (Evolutionary Computation - EC).

Η Νοημοσύνη Σμήνους (Swarm Intelligence - SI) είναι μέρος του EC. Ερευνά τη συλλογική συμπεριφορά αποκεντρωμένων, αυτο-οργανωμένων συστημάτων, φυσικών ή τεχνητών. Η έμπνευση προέρχεται συχνά από τη φύση, ειδικά από τα βιολογικά συστήματα [31]. Πιο συγκεκριμένα, οι Kennedy και Eberhart (1995) εισήγαγαν τη βελτιστοποίηση του σμήνους σωματιδίων που βασίστηκε στην κοινωνική συμπεριφορά του κοπαδιού των ψαριών ή του σμήνους των πτηνών [33].

Η βελτιστοποίηση του σμήνους σωματιδίων (Particle Swarm Optimization - PSO) προέρχεται από τη μίμηση της κοινωνικής συμπεριφοράς του σμήνους πουλιών και αρχικοποιεί έναν πληθυσμό σωματιδίων που προσομοιώνει ένα σμήνος πουλιών.



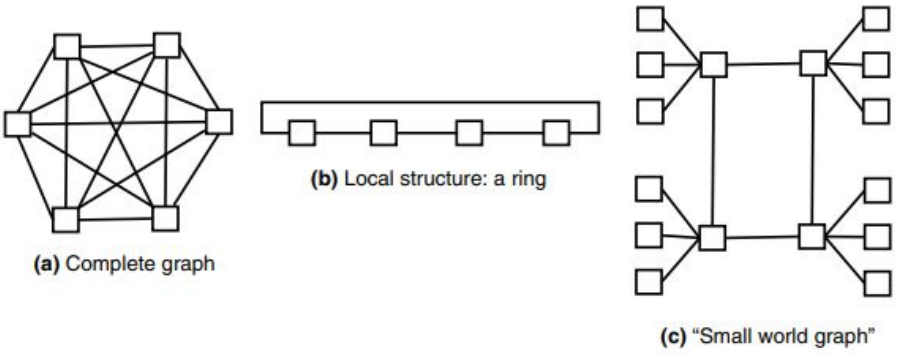
Εικόνα 4.1 Σμήνος σωματιδίων με τις σχετικές θέσεις και ταχύτητες. Σε κάθε επανάληψη, ένα σωματίδιο κινείται από τη μία θέση στην άλλη στον χώρο λήψης αποφάσεων.

Στο βασικό μοντέλο, ένα σμήνος αποτελείται από N σωματίδια που πετούν μέσα σε ένα D -διάστατο χώρο αναζήτησης. Κάθε σωματίδιο i είναι μια υποψήφια λύση στο πρόβλημα και αντιπροσωπεύεται από τον φορέα x_i στο χώρο αποφάσεων. Ένα σωματίδιο έχει τη δική του θέση και ταχύτητα, δηλαδή την κατεύθυνση και το βήμα του σωματιδίου πτήσης (Εικ. 4.1). Η βελτιστοποίηση εκμεταλλεύεται τη συνεργασία - αλληλεπίδραση μεταξύ των σωματιδίων. Η επιτυχία ορισμένων σωματιδίων θα επηρεάσει τη συμπεριφορά των υπόλοιπων σωματιδίων. Κάθε σωματίδιο προσαρμόζει διαδοχικά τη θέση του x_i με στόχο την εύρεση του ολικού βέλτιστου (global best) σύμφωνα με τους ακόλουθους δύο παράγοντες: την καλύτερη θέση που επισκέφθηκε το ίδιο ($pbest_i$) που υποδηλώνεται ως $p_i = (p_{i1}, p_{i2}, ..., p_{iD})$ και την καλύτερη θέση που επισκέπτεται που έχει επισκεφθεί ολικά ολόκληρο σμήνος (gbest) (ή lbest, η καλύτερη θέση για ένα δεδομένο υποσύνολο του σμήνους) που υποδηλώνεται ως $p_g = \left(p_{g1}, p_{g2}, ..., p_{gD}\right)$. Το διάνυσμα $\left(p_g - x_i\right)$ αντιπροσωπεύει τη διαφορά μεταξύ της τρέχουσας θέσης του σωματιδίου i και της καλύτερης θέσης της γειτονιάς του.

Γειτονιά σωματιδίων

Πρέπει να οριστεί μια γειτονιά (Particles Neighborhood) για κάθε σωματίδιο. Αυτή η γειτονιά δηλώνει την κοινωνική επιρροή μεταξύ των σωματιδίων. Υπάρχουν πολλές δυνατότητες για τον ορισμό μιας τέτοιας γειτονιάς. Παραδοσιακά, χρησιμοποιούνται δύο μέθοδοι [37]:

- Μέθοδος gbest: η γειτονιά ορίζεται ως ο συνολικός πληθυσμός σωματιδίων.
- Μέθοδος lbest: Στην τοπική καλύτερη μέθοδο, μια δεδομένη τοπολογία σχετίζεται με το σμήνος. Ως εκ τούτου, η γειτονιά ενός σωματιδίου είναι το σύνολο των άμεσα συνδεδεμένων σωματιδίων. Η γειτονιά μπορεί να είναι κενή στην οποία είναι απομονωμένα τα σωματίδια . Η εικόνα 4.2 δείχνει τρεις διαφορετικές τοπολογίες: πλήρες γράφημα, γράφημα παγκόσμιου δακτυλίου και μικρό παγκόσμιο γράφημα. Αυτό το μοντέλο είναι παρόμοιο με τα μοντέλα κοινωνικής επιστήμης που βασίζονται στην αμοιβαία απομίμηση των μελών του πληθυσμού, όπου μια σταθεροποιημένη διαμόρφωση θα αποτελείται από ομοιογενείς υποπληθυσμούς. Κάθε υποπληθυσμός θα καθορίσει μια κοινωνιομετρική περιοχή όπου θα εμφανιστεί μια «κουλτούρα». Τα άτομα στην ίδια κοινωνιομετρική περιοχή τείνουν να γίνονται παρόμοια και τα άτομα που ανήκουν σε διαφορετικές περιοχές τείνουν να είναι διαφορετικά [38].



Εικόνα 4.2 Γειτονιά σωματιδίων. (α) Μέθοδος gbest στην οποία η γειτονιά είναι ολόκληρος ο πληθυσμός (πλήρες γράφημα). (b) Μέθοδος lbest όπου ένα μη πλήρες γράφημα χρησιμοποιείται για τον καθορισμό της δομής της γειτονιάς (π.χ., ένας δακτύλιος στον οποίο κάθε σωματίδιο έχει δύο γείτονες). (γ) Ενδιάμεση τοπολογία χρησιμοποιώντας ένα μικρό παγκόσμιο γράφημα.

Σύμφωνα με τη γειτονιά που χρησιμοποιείται, ένας ηγέτης (δηλαδή, lbest ή gbest) αντιπροσωπεύει το σωματίδιο που χρησιμοποιείται για να καθοδηγήσει την αναζήτηση ενός σωματιδίου προς καλύτερες περιοχές του χώρου λήψης αποφάσεων.

Ένα σωματίδιο αποτελείται από τρία διανύσματα:

• Το x -vector καταγράφει την τρέχουσα θέση (θέση) του σωματιδίου στο χώρο αναζήτησης.

• Το διάνυσμα p καταγράφει τη θέση της καλύτερης λύσης που έχει βρεθεί μέχρι στιγμής από το σωματίδιο.

• Το διάνυσμα v περιέχει μια κλίση (κατεύθυνση) για την οποία το σωματίδιο θα ταξιδέψει μέσα εάν δεν είναι διαταραγμένο.
• Δύο τιμές φυσικής κατάστασης: Το x-fitness καταγράφει την καταλληλότητα του x-vector και το p-fitness καταγράφει την καταλληλότητα του p-vector.

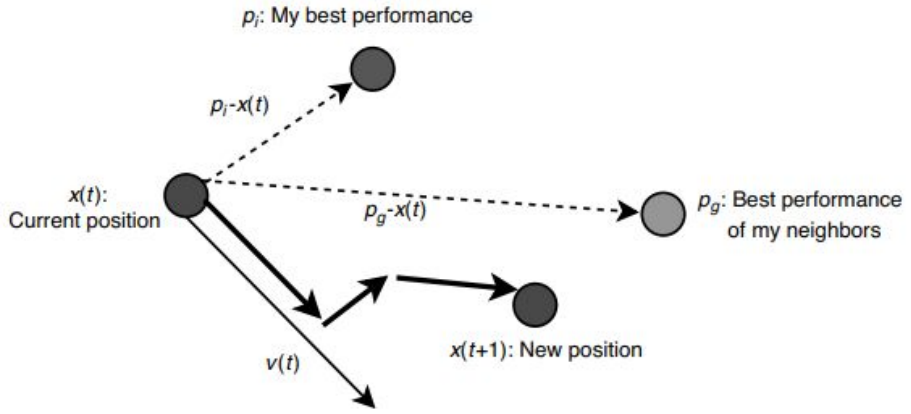
Ένα σμήνος σωματιδίων μπορεί να θεωρηθεί ως ένα κυψελοειδές αυτόματο (cellular automata) όπου οι ενημερώσεις των μεμονωμένων κυττάρων (σωματίδια στο PSO) γίνονται παράλληλα. Κάθε νέα τιμή κελιού εξαρτάται μόνο από την παλιά τιμή του κελιού και της γειτονιάς του και όλα τα κελιά ενημερώνονται χρησιμοποιώντας τους ίδιους κανόνες.

Σε κάθε επανάληψη, κάθε σωματίδιο θα εφαρμόζει τις ακόλουθες λειτουργίες:

• Ενημέρωση της ταχύτητας: Η ταχύτητα που καθορίζει το ποσό της αλλαγής που θα εφαρμοστεί στο σωματίδιο ορίζεται ως:

$$v_i(t) = v_i(t-1) + \rho_1 C_1 \times (p_i - x_i(t-1)) + \rho_2 C_2 \times \left(p_g - x_i(t-1)\right)$$

όπου ρ_1 και ρ_2 είναι δύο τυχαίες μεταβλητές στο εύρος [0, 1]. Οι σταθερές C_1 και C_2 αντιπροσωπεύουν τους μαθησιακούς παράγοντες. Αντιπροσωπεύουν την έλξη που έχει ένα σωματίδιο είτε προς τη δική του επιτυχία και προς την επιτυχία των γειτόνων του αντίστοιχα. Η παράμετρος C_1 είναι ο γνωστικός παράγοντας μάθησης που αντιπροσωπεύει την έλξη που έχει ένα σωματίδιο στην επιτυχία του. Η παράμετρος C_2 είναι ο παράγοντας κοινωνικής μάθησης που αντιπροσωπεύει την έλξη που έχει ένα σωματίδιο προς την επιτυχία των γειτόνων του. Η ταχύτητα καθορίζει την κατεύθυνση και την απόσταση που πρέπει να διανύσει το σωματίδιο (βλ. Εικ. 4.3).



Εικόνα 4.3 Κίνηση σωματιδίου και ενημέρωση ταχύτητας.

Αυτός ο τύπος αντικατοπτρίζει μια θεμελιώδη πτυχή της ανθρώπινης κοινωνικότητας όπου η κοινωνικο-ψυχολογική τάση των ατόμων μιμείται τις επιτυχίες άλλων ατόμων. Ακολουθώντας τον τύπο ενημέρωσης ταχύτητας, ένα σωματίδιο θα περιστρέφεται γύρω από το σημείο που ορίζεται ως ο σταθμισμένος μέσος όρος των p_i και p_g :

$$\frac{\rho_1 p_i + \rho_2 p_g}{\rho_1 + \rho_2}$$

Είναι πολύ κοινό να ορίζεται ένα ανώτερο όριο για την παράμετρο ταχύτητας. Το "Velocity clamping" [34] χρησιμοποιήθηκε ως τρόπος περιορισμού των σωματιδίων που πετούν έξω από το χώρο αναζήτησης. Μια άλλη μέθοδος είναι η στρατηγική «συντελεστής περιορισμού», που πρότειναν οι Clerc και Kennedy [35], ως αποτέλεσμα μιας θεωρητικής ανάλυσης της

δυναμικής του σμήνους, στην οποία οι ταχύτητες περιορίζονται επίσης [33]. Τα στοιχεία του v_i περιορίζονται σε μια μέγιστη τιμή $[-V_{max}, +V_{max}]$, όπως το σύστημα δεν θα εκραγεί λόγω της τυχαιότητας του συστήματος. Εάν η ταχύτητα v_i υπερβαίνει το V_{max} (αντιστοίχως $-V_{max}$), θα επανέλθει σε V_{max} (αντίστοιχα $-V_{max}$).
Στη διαδικασία ενημέρωσης ταχύτητας, ένα βάρος αδράνειας w γενικά προστίθεται στην προηγούμενη ταχύτητα [39]:

$$v_i(t) = \omega \times v_i(t-1) + \rho_1 \times (p_i - x_i(t-1)) + \rho_2 \times \left(p_g - x_i(t-1)\right)$$

Το πρώτο μέρος του τύπου, γνωστό ως «αδράνεια» (inertia), αντιπροσωπεύει την προηγούμενη ταχύτητα, η οποία παρέχει την απαραίτητη ορμή για τα σωματίδια να περιφέρονται στον χώρο αναζήτησης. Το δεύτερο μέρος, γνωστό ως «γνωστικό» συστατικό (cognitive component), αντιπροσωπεύει την ατομική σκέψη κάθε σωματιδίου. Ενθαρρύνει τα σωματίδια να κινηθούν προς τις δικές τους καλύτερες θέσεις που έχουν βρεθεί μέχρι στιγμής. Το τρίτο μέρος, το στοιχείο «συνεργασίας» (cooperation component), αντιπροσωπεύει τη συνεργατική επίδραση των σωματιδίων για την εξεύρεση της παγκόσμιας βέλτιστης λύσης [33]. Το βάρος αδράνειας θα ελέγξει την επίδραση της προηγούμενης ταχύτητας στην τρέχουσα. Για μεγάλες τιμές του βάρους αδράνειας, η επίδραση των προηγούμενων ταχυτήτων θα είναι πολύ μεγαλύτερη. Έτσι, το βάρος αδράνειας αντιπροσωπεύει μια αντιστάθμιση μεταξύ της παγκόσμιας εξερεύνησης (global exploration) και της τοπικής εκμετάλλευσης (local exploitation). Ένα μεγάλο βάρος αδράνειας ενθαρρύνει την παγκόσμια εξερεύνηση (δηλαδή, εντείνει την αναζήτηση σε ολόκληρο το χώρο αναζήτησης), ενώ ένα μικρότερο βάρος αδράνειας ενθαρρύνει την τοπική εκμετάλλευση (δηλαδή, εντείνει την αναζήτηση στην τρέχουσα περιοχή) [36].

• Ενημέρωση θέσης: Κάθε σωματίδιο θα ενημερώσει τις συντεταγμένες του στο χώρο αποφάσεων.

$$x_i(t) = x_i(t-1) + v_i(t)$$

Στη συνέχεια μετακινείται στη νέα θέση.

• Ενημέρωση των καλύτερων σωματιδίων που βρέθηκαν: Κάθε σωματίδιο θα ενημερώσει (ενδεχομένως) την καλύτερη τοπική λύση:

$$\text{Εάν } f(x_i) < pbest_i, \text{ τότε } p_i = x_i$$

Επιπλέον, ενημερώνεται η καλύτερη παγκόσμια λύση του σμήνους:

$$\text{Εάν } f(x_i) < gbest, \text{ τότε } g = x_i$$

Ως εκ τούτου, σε κάθε επανάληψη, κάθε σωματίδιο θα αλλάξει τη θέση του σύμφωνα με τη δική του εμπειρία και αυτή των γειτονικών σωματιδίων.

Ψευδοκώδικας PSO

Έστω $f: R^N \rightarrow R$ είναι η συνάρτηση κόστους που θα ελαχιστοποιηθεί. Η συνάρτηση παίρνει μια υποψήφια λύση ενός διανύσματος p πραγματικών αριθμών και παράγει έναν πραγματικό αριθμό ως έξοδο που δείχνει την τιμή της συνάρτησης κόστους [33]. Η κλίση του f είναι είτε άγνωστη είτε δύσκολο να υπολογιστεί. Ο στόχος είναι να βρεθεί το παγκόσμιο ελάχιστο x^* (Ψευδοκώδικας 1).

Βήμα 1. Αρχικοποίηση

Για κάθε σωματίδιο $i = 1....,N_p$ *, κάνε*

(α)Αρχικοποιήσε τυχαία τη θέση κάθε σωματιδίου με ομοιόμορφη κατανομή ως $x_i(0) \sim U(LB, UB)$*, όπου LB και UB αντιπροσωπεύουν τα άνω και κάτω όρια του χώρου αναζήτησης*

(β) Αρχικοποίηση $pbest_i$ *στην αρχική του θέση:* $pbest_i(0) = p_i(0)$.

(γ) Αρχικοποιήση $gbest$ *στην ελάχιστη τιμή του σμήνους:* $gbest(0) = argminf\left[p_i(0)\right]$

(δ) Αρχικοποιήσε την ταχύτητα: $v_i \sim U(-|UB - LB|, |UB - LB|)$

Βήμα 2. Επαναλάβε έως ότου πληρούνται τα κριτήρια τερματισμού

Για κάθε σωματίδιο $i = 1....,N_p$ *, κάνε*

(α) Διάλεξε τυχαίους αριθμούς: $\rho_1, \rho_2 \sim U(0, 1)$.

(β) Ενημέρωσε την ταχύτητα των σωματιδίων σύμφωνα με τον τύπο:
$v_i(t) = \omega \times v_i(t-1) + \rho_1 \times (p_i - x_i(t-1)) + \rho_2 \times \left(p_g - x_i(t-1)\right)$.

(γ) Ενημέρωσε τη θέση των σωματιδίων σύμφωνα με τον τύπο: $x_i(t) = x_i(t-1) + v_i(t)$

(δ) Εάν $f(x_i(t)) < p_i(t)$*, κάνε*

(i) Ενημέρωσε την καλύτερη θέση του σωματιδίου i: $p_i = x_i(t)$.

(ii) Εάν $f(x_i(t)) < g(t)$*, ενημέρωσε την καλύτερη θέση του σμήνους:* $g = x_i(t)$

(ε) $t \leftarrow (t+1)$

Βήμα 3. Έξοδος $gbest(t)$ *που περιέχει την καλύτερη λύση*

Ψευδοκώδικας 1: Ένα τυπικό PSO.

4.3 Bayesian Optimization

Η Μπεϊζιανή Βελτιστοποίηση (Bayesian Optimization) είναι μια καθιερωμένη στρατηγική για την παγκόσμια βελτιστοποίηση των θορυβωδών, δαπανηρών black-box συναρτήσεων [41], δηλαδή συναρτήσεων των οποίων οι εσωτερικές εκφράσεις και μηχανισμοί δεν είναι γνωστοί. Μεγάλο μέρος της αποτελεσματικότητας πηγάζει από την ικανότητα της Μπεϋζιανής Βελτιστοποίησης να ενσωματώσει προηγούμενη πεποίθηση του προβλήματος για να διευκολύνει τη δειγματοληψία και να ισορροπήσει την εξερεύνηση και εκμετάλλευση του χώρου αναζήτησης. Ονομάζεται μπεϊζιανή επειδή χρησιμοποιεί το περίφημο "Bayes θεώρημα", το οποίο δηλώνει (απλοποιώντας κάπως) ότι η μεταγενέστερη πιθανότητα (posterior probability) ενός μοντέλου (ή θεωρίας ή υπόθεσης) M που έχει αποδεικτικά στοιχεία (ή δεδομένα ή παρατηρήσεις) E είναι ανάλογη (likelihood) με την πιθανότητα του στοιχείου E δεδομένου του M πολλαπλασιασμένη με την προηγούμενη πιθανότητα M [49]:

$$P(M \mid E) \propto P(E \mid M)\, P(M)$$

Μέσα σε αυτήν την απλή εξίσωση είναι το κλειδί για τη βελτιστοποίηση της αντικειμενικής συνάρτησης. Στη Μπεϋζιανή Βελτιστοποίηση, το προηγούμενο (prior) αντιπροσωπεύει την πεποίθησή για το χώρο των πιθανών αντικειμενικών συναρτήσεων. Παρόλο που η συνάρτηση κόστους είναι άγνωστη, είναι λογικό να υποθέσει κανείς ότι υπάρχει προηγούμενη γνώση σχετικά με ορισμένες από τις ιδιότητές της, όπως η ομαλότητα, και αυτό καθιστά ορισμένες πιθανές αντικειμενικές συναρτήσεις πιο εύλογες από άλλες.

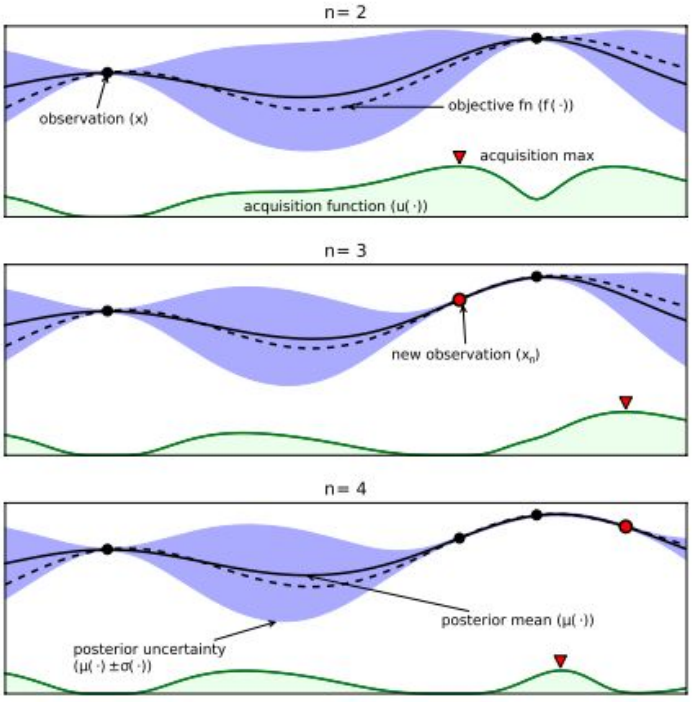
Ας οριστεί το x_i ως το δείγμα i_{th}, και $f(x_i)$ ως την παρατήρηση της αντικειμενικής συνάρτησης στο x_i. Καθώς συγκεντρώνονται παρατηρήσεις $D_{1:t} = \{x_{1:t}, f(x_{1:t})\}$, η προηγούμενη κατανομή συνδυάζεται με τη συνάρτηση πιθανότητας (likelihood function) $P(D_{1:t} \mid f)$. Ουσιαστικά,

δεδομένου ότι υπάρχει η υπόθεση ότι είναι γνωστά τα προηγούμενα, πόσο πιθανά είναι τα δεδομένα που έχει δει κάποιος; Εάν η προηγούμενη πεποίθηση είναι ότι η αντικειμενική συνάρτηση είναι πολύ ομαλή και χωρίς θόρυβο, τα δεδομένα με μεγάλη διακύμανση ή ταλαντώσεις θα πρέπει να θεωρούνται λιγότερο πιθανά από ότι δεδομένα που μόλις αποκλίνουν από το μέσο όρο. Τώρα, μπορούν να τα συνδυαστούν για τη δημιουργία της προηγούμενης κατανομή:

$$P(f \mid D_{1:t}) \propto P(D_{1:t} \mid f)\, P(f)$$

Το προηγούμενο (prior) μέρος συλλαμβάνει τις ενημερωμένες πεποιθήσεις του χρήστη σχετικά με την άγνωστη αντικειμενική συνάρτηση. Κάποιος μπορεί επίσης να ερμηνεύσει αυτό το βήμα της Μπεϋζιανής Βελτιστοποίησης ως εκτίμηση της αντικειμενικής συνάρτησης με μια υποκατάστατη συνάρτηση (surrogate function) (που ονομάζεται επίσης επιφάνεια απόκρισης (response surface)). Οι Γκαουσιανές διαδικασίες (Gaussian processes - GPs) είναι ιδιαίτερα δημοφιλείς υποκατάστατες συναρτήσεις. Ωστόσο, δεδομένου ότι οι GPs κλιμακώνονται κυβικά με τον αριθμό των παρατηρήσεων, είναι δύσκολο να χειριστούν συναρτήσεις των οποίων η βελτιστοποίηση απαιτεί πολλές αξιολογήσεις [40] [45][46].

Για αποτελεσματική δειγματοληψία, η Μπεϊζιανή Βελτιστοποίηση χρησιμοποιεί μια συνάρτηση απόκτησης (acquisition function) για να προσδιορίσει την επόμενη τοποθεσία $x_{t+1} \in A$ που θα δειγματοληπτήσει. Η απόφαση αντιπροσωπεύει μια αυτόματη αντιστάθμιση μεταξύ της εξερεύνησης (όπου η αντικειμενική συνάρτηση είναι πολύ αβέβαιη) και της εκμετάλλευσης (δοκιμάζονται τιμές x σε περιοχές όπου η αντικειμενική συνάρτηση αναμένεται να είναι υψηλή). Αυτή η τεχνική βελτιστοποίησης έχει την ιδιότητα να στοχεύει στην ελαχιστοποίηση του αριθμού των αξιολογήσεων της αντικειμενικής συνάρτησης. Επιπλέον, είναι πιθανό να τα πάει καλά ακόμα και σε περιπτώσεις όπου η αντικειμενική συνάρτηση έχει πολλαπλά τοπικά μέγιστα [48].



Εικόνα 4.4: Απεικόνιση της διαδικασίας Μπεϋζιανής Βελτιστοποίησης σε τρεις επαναλήψεις. Τα γραφήματα δείχνουν τα μέσα και τα διαστήματα εμπιστοσύνης που υπολογίζονται με ένα πιθανό μοντέλο της αντικειμενικής συνάρτησης. Αν και η αντικειμενική συνάρτηση εμφανίζεται, στην πράξη, είναι άγνωστη. Τα οικόπεδα δείχνουν επίσης τις συναρτήσεις απόκτησης στα κάτω σκιασμένα γραφήματα. Η απόκτηση είναι υψηλή όταν το μοντέλο προβλέπει υψηλό στόχο (εκμετάλλευση) και όπου η αβεβαιότητα πρόβλεψης είναι υψηλή (εξερεύνηση). Σημειώστε ότι η περιοχή στα αριστερά παραμένει χωρίς δείγμα, καθώς ενώ έχει υψηλή αβεβαιότητα, προβλέπεται σωστά να προσφέρει μικρή βελτίωση σε σχέση με την υψηλότερη παρατήρηση.

1: Για $n = 1, 2, ...,$ κάνε
2: Επίλεξε νέο x_{n+1} βελτιστοποιώντας τη συνάρτηση απόκτησης
$x_{n+1} = argmax_x a(x; D_n)$
3: Ζήτα από την αντικειμενική συνάρτηση να αποκτήσεις y_{n+1}
4: Αύξησε τα δεδομένα $D_{n+1} = \{D_n, (x_{n+1}, y_{n+1})\}$
5: Ενημέρωσε το στατιστικό μοντέλο
6: τέλος για

Ψευδοκώδικας 2: Μπεϊζιανή Βελτιστοποίηση

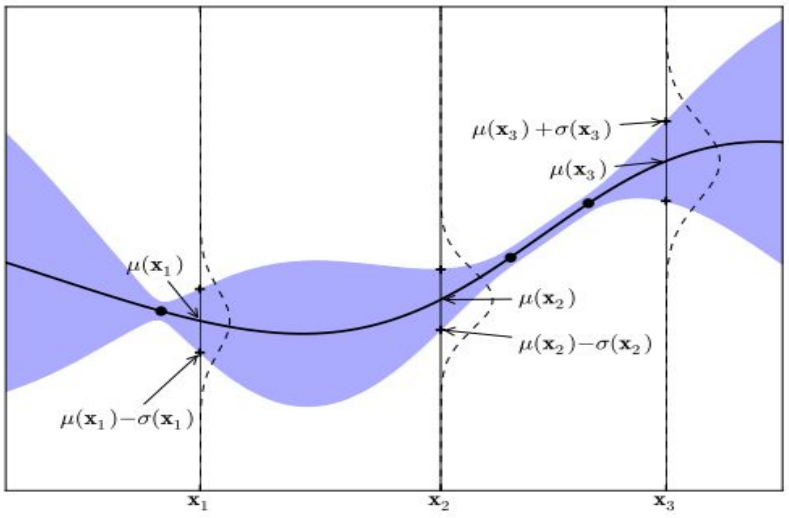
4.3.1 Προηγούμενη Γνώση

Οποιαδήποτε μπεϋζιανή μέθοδος εξαρτάται από μια προηγούμενη κατανομή, εξ ορισμού. Μια μέθοδος Μπεϋζιανής βελτιστοποίησης θα συγκλίνει στο βέλτιστο εάν (i) η συνάρτηση απόκτησης είναι συνεχής και ελαχιστοποιεί τον κίνδυνο (ορίζεται ως η αναμενόμενη απόκλιση από το παγκόσμιο ελάχιστο σε ένα σταθερό σημείο x) και (ii) η υπό όρους διακύμανση συγκλίνει στο μηδέν (ή κατάλληλη θετική ελάχιστη τιμή παρουσία θορύβου) εάν και μόνο εάν η απόσταση από την πλησιέστερη παρατήρηση είναι μηδέν [49][50]. Πολλά μοντέλα θα μπορούσαν να χρησιμοποιηθούν για αυτό το προηγούμενο. Ωστόσο, η διαδικασία Gaussian (GP) προηγείται της Μπεϋζιανής βελτιστοποίησης αφού χρονολογείται τουλάχιστον στο τέλος του 1970. Ο Mockus [1994] έθεσε ρητά το πλαίσιο για τη GP με τον καθορισμό των πρόσθετων «απλών και φυσικών» συνθηκών όπου (iii) η αντικειμενική συνάρτηση είναι συνεχής. (iv) το προηγούμενο (prior) είναι ομοιογενές (v) η βελτιστοποίηση είναι ανεξάρτητη από τις m_{th} διαφορές. Αυτό περιλαμβάνει μια πολύ μεγάλη οικογένεια κοινών εργασιών βελτιστοποίησης και ο Mockus έδειξε ότι η GP είναι κατάλληλη για την εργασία.

Η GP είναι μια επέκταση της πολυμεταβλητής κατανομής Gauss σε μια στοχαστική διαδικασία απεριόριστης διάστασης για την οποία οποιοσδήποτε πεπερασμένος συνδυασμός διαστάσεων θα είναι μια κατανομή Gauss. Ακριβώς όπως μια γκαουσιανή κατανομή είναι μια κατανομή σε μια τυχαία μεταβλητή, που προσδιορίζεται πλήρως από τη μέση τιμή και τη συνδιακύμανσή της, μια GP είναι μια κατανομή σε συναρτήσεις, που καθορίζεται πλήρως από τη μέση συνάρτηση m και τη συνδιακύμανση k:

$$f(x) \sim GP(m(x),\ k(x, x'))$$

Συχνά είναι χρήσιμο να αποφασίζεται διαισθητικά ένα GP ως ανάλογο μιας συνάρτησης, αλλά αντί να επιστρέφει μια βαθμωτή $f(x)$ για ένα αυθαίρετο x, επιστρέφει τη μέση τιμή και τη διακύμανση μιας κανονικής κατανομής (Εικόνα 4.5) έναντι των πιθανών τιμών της f στο x. Οι στοχαστικές διαδικασίες ονομάζονται μερικές φορές «τυχαίες συναρτήσεις», κατ 'αναλογία με τυχαίες μεταβλητές.



Εικόνα 4.5: Απλή Gaussian διαδικασία 1D με τρεις παρατηρήσεις. Η συμπαγής μαύρη γραμμή είναι η μέση πρόβλεψη GP αντικατάστασης της αντικειμενικής συνάρτησης (GP surrogate mean prediction) δεδομένου των δεδομένων, και η σκιασμένη περιοχή δείχνει τη μέση συν και μείον τη διακύμανση. Οι υπέρτιμοι Γκάσοι αντιστοιχούν στη μέση τιμή GP και την τυπική απόκλιση (μ (·) και σ (·)) της πρόβλεψης στα σημεία, $x_{1:3}$.

Για ευκολία, γίνεται η υπόθεση εδώ ότι η προηγούμενη μέση τιμή είναι η συνάρτηση μηδέν $m(x) = 0$. Εναλλακτικά προηγούμενα (priors) για το μέσο όρο μπορούν να βρεθούν. Αυτό δημιουργεί το πιο ενδιαφέρον ερώτημα για τον ορισμό της συνδιακύμανσης συνάρτηση k. Μια πολύ δημοφιλής επιλογή είναι η τετραγωνική εκθετική συνάρτηση:

$$k(x_i, x_j) = exp\left(-\tfrac{1}{2}\|x_i - x_j\|^2\right)$$

Σημειώστε ότι αυτή η συνάρτηση πλησιάζει το 1 καθώς οι τιμές πλησιάζουν μεταξύ τους και το 0 καθώς διαχωρίζονται περισσότερο. Δύο σημεία που βρίσκονται κοντά είναι αναμενόμενο να έχουν πολύ μεγάλη επιρροή μεταξύ τους, ενώ τα απομακρυσμένα σημεία δεν έχουν σχεδόν κανένα.

Εάν επρόκειτο να πραγματοποιηθεί δειγματοληψία από το προηγούμενο, θα έπρεπε να επιλεγεί $\{x_{1:t}\}$ και θα έπρεπε να περαστούν οι τιμές της συνάρτησης σε αυτούς τους δείκτες για να παραχθούν τα ζεύγη $\{x_{1:t}, f_{1:t}\}$, όπου $f_{1:t} = f(x_{1:t})$. Οι τιμές συνάρτησης σχεδιάζονται σύμφωνα με μια κανονική κατανομή πολλαπλών παραλλαγών $N(0, K)$, όπου η μήτρα του πυρήνα δίνεται από:

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \dots & k(\mathbf{x}_1, \mathbf{x}_t) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}_t, \mathbf{x}_1) & \dots & k(\mathbf{x}_t, \mathbf{x}_t) \end{bmatrix}$$

4.3.2 Συναρτήσεις Απόκτησης

Έστω ότι η συνάρτηση $f(x)$ προέρχεται από μια διαδικασία Gauss και ότι οι παρατηρήσεις έχουν τη μορφή $\{x_n, y_n\}_{n=1}^{N}$, όπου $y_n \sim N(f(x_n), v)$ και v είναι η διακύμανση του θορύβου που εισάγεται στις παρατηρήσεις της συνάρτησης. Αυτό το προηγούμενο (prior) και αυτά τα δεδομένα προκαλούν ένα μεταγενέστερο (posterior) στις συναρτήσεις: τη συνάρτηση απόκτησης (acquisition function), η οποία υποδηλώνεται με: $X \to R^+$, και καθορίζει ποιο σημείο στο X πρέπει να αξιολογηθεί στη συνέχεια μέσω ενός $x_{next} = argmax_x a(x)$, όπου έχουν προταθεί διάφορες διαφορετικές συναρτήσεις. Σε γενικές γραμμές, αυτές οι συναρτήσεις απόκτησης εξαρτώνται από τις προηγούμενες παρατηρήσεις, καθώς και από τις υπερπαραμέτρους GP - δηλώνεται αυτή η εξάρτηση ως $a(x; \{x_n, y_n\}, \theta)$. Υπάρχουν πολλές δημοφιλείς επιλογές συναρτήσεων απόκτησης. Σε κάποια διαδικασία Gauss, αυτές οι συναρτήσεις μπορεί να εξαρτώνται από το μοντέλο αποκλειστικά μέσω

της μέσης πρόβλεψης της συνάρτησης $\mu(x; \{x_n, y_n\}, \theta)$ και της συνάρτησης προγνωστικής διακύμανσης $\sigma^2(x; \{x_n, y_n\}, \theta)$. Στην πορεία, θα υποδηλώνεται η καλύτερη τρέχουσα τιμή ως $x_{best} = argmin_{x_n} f(x_n)$ και η συνάρτηση αθροιστικής κατανομής του κανονικού ως $\Phi(\cdot)$ [52].

Πιθανότητα βελτίωσης (Probability of Improvement): Μια διαισθητική στρατηγική είναι η μεγιστοποίηση της πιθανότητας βελτίωσης έναντι της καλύτερης τρέχουσας τιμής [51]. Κάτω από τη GP αυτό μπορεί να υπολογιστεί αναλυτικά ως:

$$a_{PI}(x; \{x_n, y_n\}, \theta) = \Phi(\gamma(x))$$

$$\gamma(x) = \frac{f(x_{best}) - \mu(x; \{x_n, y_n\}, \theta)}{\sigma(x; \{x_n, y_n\}, \theta)}$$

Αναμενόμενη βελτίωση (Expected Improvement - EI): Εναλλακτικά, κάποιος θα μπορούσε να επιλέξει να μεγιστοποιήσει την αναμενόμενη βελτίωση (EI) έναντι του τρέχοντος καλύτερου. Αυτό έχει επίσης κλειστή φόρμα στο πλαίσιο της διαδικασίας Gauss:

$$a_{\mathrm{EI}}(x; \{x_n, y_n\}, \theta) = \sigma(x; \{x_n, y_n\}, \theta)\,(\gamma(x)\,\Phi(\gamma(x)) + N(\gamma(x); 0, 1))$$

Άνω όριο διαστήματος εμπιστοσύνης (GP Upper Confidence Bound): Μια πιο πρόσφατη ιδέα είναι η ιδέα της εκμετάλλευσης χαμηλότερων ορίων εμπιστοσύνης (άνω, όταν εξετάζεται για μεγιστοποίηση) για την κατασκευή συναρτήσεων απόκτησης που ελαχιστοποιούν τη λύπη κατά τη διάρκεια της βελτιστοποίησής τους [3]. Αυτές οι συναρτήσεις απόκτησης έχουν τη μορφή:

$$a_{LCB}(x; \{x_n, y_n\}, \theta) = \mu(x; \{x_n, y_n\}, \theta) - \kappa\sigma(x; \{x_n, y_n\}, \theta)$$

με παράμετρο κ για εξισορρόπηση της εκμετάλλευσης έναντι της εξερεύνησης.

4.4 Δυναμικό και στατικό περιβάλλον βελτιστοποίησης

4.4.1 Πρόβλημα δυναμικής βελτιστοποίησης

Ένα πρόβλημα δυναμικής βελτιστοποίησης (Dynamic optimization problem - DOP) μπορεί να οριστεί διαισθητικά ως μια ακολουθία στατικών προβλημάτων που πρέπει να βελτιστοποιηθούν [53]. Οι δύο κύριες πτυχές της «δυναμικότητας» καθορίζονται από τη συχνότητα και το μέγεθος των περιβαλλοντικών αλλαγών. Οι πρώτες και οι τελευταίες παράμετροι αντιστοιχούν στην ταχύτητα και το βαθμό με τον οποίο το περιβάλλον του προβλήματος αλλάζει, αντίστοιχα. Άλλες πτυχές περιλαμβάνουν την προβλεψιμότητα, την ικανότητα ανίχνευσης και τη χρονική σύνδεση δυναμικών αλλαγών [54][55]. Οι δύο πρώτες πτυχές αντιστοιχούν στο κατά πόσον μια δυναμική αλλαγή μπορεί να προβλεφθεί ή να προκαθοριστεί κατά τη διάρκεια της εκτέλεσης ή όχι, αντίστοιχα, και η τελευταία αντιστοιχεί στο κατά πόσον μια απόφαση που ελήφθη τώρα για την αντιμετώπιση μιας δυναμικής αλλαγής εξαρτάται από προηγούμενες αποφάσεις ή όχι.

Μια περιβαλλοντική αλλαγή μπορεί να περιλαμβάνει την αντικειμενική συνάρτηση (ή συναρτήσεις εάν ένα δυναμικό πρόβλημα πολλαπλών αντικειμένων εξετάζεται [56][57]), μεταβλητές εισόδου, στιγμιότυπα προβλημάτων και περιορισμούς (π.χ. δυναμικά περιορισμένη βελτιστοποίηση [58]). Επισήμως, ένα DOP μπορεί να οριστεί ως εξής:

$$DOP = \text{βελτιστοποίησε την } f(x, t) \text{ υπό τον όρο } X(t) \subseteq S, \quad t \in T$$

όπου S είναι ο χώρος αναζήτησης, t είναι ο χρόνος, $f : S \times T \rightarrow R$ είναι η αντικειμενική συνάρτηση που εκχωρεί μια τιμή (π.χ. R) σε κάθε πιθανή λύση $x \in X$ και $X(t)$ είναι το σύνολο των εφικτών λύσεων $x \in X(t) \subseteq S$ στο χρόνο t [59][60]. Κάθε εφικτή λύση x αποτελείται από μεταβλητές βελτιστοποίησης $x = \{x_1, ..., x_n\}$. Κάθε λύση $x \in X(t)$ έχει ένα σύνολο γειτονικών $N(x) \subseteq X(t)$ όπου $N(x)$ είναι μια συνάρτηση που εκχωρεί μια γειτονιά στο x. Μια τοπική βέλτιστη λύση είναι

μια εφικτή λύση x' με $f(x',t) \leq f(x,t),\ \forall x \in N(x)$ για προβλήματα ελαχιστοποίησης, ή $f(x',t) \geq f(x,t),\ \forall x \in N(x)$ για προβλήματα μεγιστοποίησης, αντίστοιχα.

Το ολικό βέλτιστο είναι μια εφικτή λύση x^* που ελαχιστοποιεί (ή μεγιστοποιεί) την αντικειμενική συνάρτηση $f(x^*,t) \leq f(x,t),\ \forall x \in X(t)$ για προβλήματα ελαχιστοποίησης ($f(x^*,t) \geq f(x,t)\ \forall x \in X(t)$ για προβλήματα μεγιστοποίησης) [61].

Τα κύρια ζητήματα για την επίλυση προβλημάτων δυναμικής βελτιστοποίησης είναι [37]:

• Ανίχνευση της αλλαγής στο περιβάλλον όταν συμβαίνει. Για τα περισσότερα από τα προβλήματα της πραγματικής ζωής, η αλλαγή είναι ομαλή.

• Αντίδραση στην αλλαγή του περιβάλλοντος για να ανιχνευθεί η νέα ολική βέλτιστη λύση. Ως εκ τούτου, η διαδικασία αναζήτησης πρέπει να προσαρμοστεί γρήγορα στην αλλαγή της αντικειμενικής συνάρτησης. Ο στόχος είναι η δυναμική παρακολούθηση της μεταβαλλόμενης βέλτιστης λύσης όσο το δυνατόν πιο κοντά. Η κύρια πρόκληση είναι η επαναχρησιμοποίηση των πληροφοριών από προηγούμενες αναζητήσεις για να προσαρμοστεί το πρόβλημα στην αλλαγή αντί να επιλυθεί ξανά το πρόβλημα από το μηδέν.

4.4.2. Διακριτός και συνεχής χώρος

Υπάρχουν διαφορετικές κατηγορίες προβλημάτων βελτιστοποίησης που διαφέρουν στον ορισμό του χώρου αναζήτησης $X(t)$. Υπάρχουν δύο βασικοί τύποι προβλημάτων οι οποίοι είναι οι εξής:

- Διακριτά προβλήματα βελτιστοποίησης όπου όλες οι μεταβλητές βελτιστοποίησης x_i είναι διακριτές που λαμβάνουν τιμές $x_i \in D_i = \left\{ v_i^1, ..., v_i^{|D_i|} \right\},\ i = 1, ..., n.$
- Συνεχή προβλήματα βελτιστοποίησης όπου όλες οι μεταβλητές βελτιστοποίησης x_i είναι συνεχείς που λαμβάνουν τιμές $x_i \in D_i \subseteq R,\ i = 1, ..., n.$

Η κύρια διαφορά μεταξύ διακριτών και συνεχών προβλημάτων βελτιστοποίησης έγκειται στο ότι τα διακριτά προβλήματα βελτιστοποίησης έχουν άπειρο χώρο αναζήτησης. Πιο συγκεκριμένα, χαρακτηρίζονται από άπειρο σύνολο μεταβλητών τιμών, π.χ. δυαδικό που περιορίζεται στις τιμές 0 και 1 ή αντικείμενα που αντλούνται από άπειρο σύνολο στοιχείων. Διαφορετικά, κάθε μεταβλητή τιμή εντός προβλημάτων συνεχούς βελτιστοποίησης μπορεί να αναλάβει άπειρο αριθμό τιμών, π.χ. πραγματικούς αριθμούς. Δεδομένου ότι οι υπολογιστές έχουν ψηφιακή φυσική κατάσταση, η αναπαράσταση διακριτών μεταβλητών είναι ευθεία προς τα εμπρός. Αντίθετα, η αναπαράσταση συνεχών μεταβλητών απαιτεί την επιβολή ορισμένων περιορισμών, δεδομένου ότι δεν είναι δυνατόν να απεικονιστεί ένας άπειρος αριθμός τιμών [61].

5. Υπολογιστική των άκρων

Η υπολογιστική νέφους (cloud computing) [99], η οποία κυριάρχησε στις συζητήσεις πληροφορικής την τελευταία δεκαετία, έχει μια διπλή πρόταση αξίας. Πρώτον, ο συγκεντρωτισμός εκμεταλλεύεται οικονομίες κλίμακας για τη μείωση του οριακού κόστους διαχείρισης και λειτουργίας του συστήματος. Δεύτερον, οι οργανισμοί μπορούν να αποφύγουν τις κεφαλαιουχικές δαπάνες από τη δημιουργία κέντρου δεδομένων, το οποίο θα καταναλώνει υπολογιστικούς πόρους μέσω διαδικτύου από έναν μεγάλο πάροχο υπηρεσιών. Αυτές οι εκτιμήσεις έχουν οδηγήσει στην ενοποίηση της χωρητικότητας υπολογιστών σε πολλά μεγάλα κέντρα δεδομένων που έχουν εξαπλωθεί σε όλο τον κόσμο. Τα αποδεδειγμένα οικονομικά οφέλη του cloud computing το καθιστούν πιθανό να παραμείνει μόνιμο χαρακτηριστικό του μελλοντικού υπολογιστικού τοπίου.

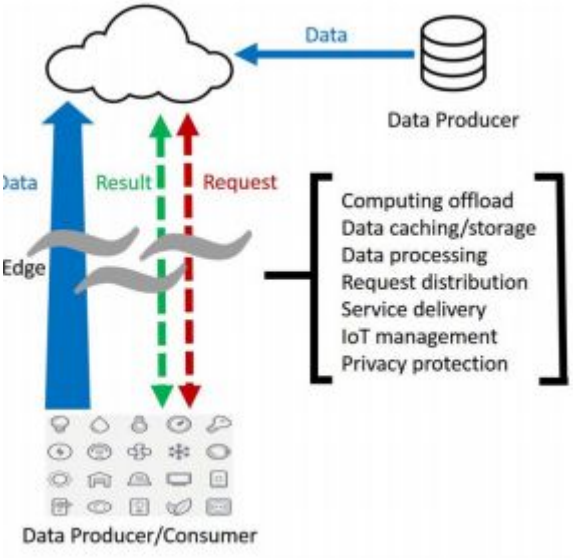
Ωστόσο, οι δυνάμεις που οδηγούν τον συγκεντρωτισμό δεν είναι οι μόνες που λειτουργούν. Οι νέες τεχνολογίες και οι εφαρμογές για φορητούς υπολογιστές και το Διαδίκτυο των πραγμάτων (IoT) οδηγούν τους υπολογιστές προς τη διασπορά. Η υπολογιστική των άκρων είναι ένα νέο παράδειγμα στο οποίο σημαντικοί υπολογιστικοί και αποθηκευτικοί πόροι - που αναφέρονται διαφορετικά ως cloudlets, micro datacenters ή κόμβοι ομίχλης (fog nodes) - τοποθετούνται στην άκρη του Διαδικτύου πολύ κοντά σε κινητές συσκευές ή αισθητήρες [68].

Σε αυτήν την ενότητα, δίνεται επεξηγείται η υπολογιστική των άκρων, και στη συνέχεια παρατίθενται μερικοί λόγοι για τους οποίους η υπολογιστική άκρων είναι αποτελεσματική για ορισμένες υπηρεσίες υπολογιστών.

5.1 Εισαγωγή στην υπολογιστική των άκρων

Η υπολογιστική των άκρων (Edge Computing) [100] αναφέρεται στις τεχνολογίες που επιτρέπουν τον υπολογισμό να εκτελείται στην άκρη του δικτύου, σε δεδομένα καθοδικής πορείας (downstream) για λογαριασμό υπηρεσιών σύννεφου (cloud) και δεδομένα ανοδικής πορείας (upstream) για λογαριασμό υπηρεσιών του διαδικτύου των πραγμάτων (Internet of Things - IoT). Εδώ ορίζεται «άκρη» ως υπολογιστικοί πόροι και πόροι δικτύου κατά μήκος της διαδρομής μεταξύ πηγών δεδομένων και κέντρων δεδομένων συννέφου. Για παράδειγμα, ένα έξυπνο τηλέφωνο είναι η άκρη μεταξύ των πραγμάτων του σώματος και του cloud, μια πύλη σε ένα έξυπνο σπίτι είναι το άκρο μεταξύ των οικιακών πραγμάτων και του cloud, ένα κέντρο μικροδεδομένων και ένα cloudlet [62] είναι το άκρο μεταξύ μιας κινητής συσκευής και cloud. Το σκεπτικό της υπολογιστικής των άκρων είναι ότι ο υπολογισμός πρέπει να συμβαίνει κοντά στις πηγές δεδομένων. Από μια άποψη, η υπολογιστική των άκρων είναι εναλλάξιμη με την υπολογιστική ομίχλης (fog computing) [63], αλλά η υπολογιστική των άκρων εστιάζει περισσότερο προς την πλευρά των πραγμάτων, ενώ η υπολογιστική ομίχλης εστιάζει περισσότερο στην πλευρά της υποδομής. Μπορεί να οραματιστεί κανείς ότι η υπολογιστική των άκρων θα μπορούσε να έχει τόσο μεγάλο αντίκτυπο στην κοινωνία όσο και το cloud computing.

Το Σχήμα 5.1 απεικονίζει τις αμφίδρομες υπολογιστικές ροές στην υπολογιστική των άκρων. Στο παράδειγμα της υπολογιστικής των άκρων, τα πράγματα δεν είναι μόνο οι καταναλωτές δεδομένων, αλλά επίσης λειτουργούν και ως παραγωγοί δεδομένων. Στην άκρη, τα πράγματα δεν μπορούν μόνο να ζητήσουν υπηρεσίες και περιεχόμενο από το σύννεφο, αλλά και να εκτελέσουν τις υπολογιστικές εργασίες από το σύννεφο. Τα άκρα μπορούν να εκτελέσουν υπολογιστική εκφόρτωση (computing offloading), αποθήκευση δεδομένων, προσωρινή αποθήκευση και επεξεργασία, καθώς να διανείμουν αιτήματα και να παραδώσουν υπηρεσίες από το σύννεφο στον χρήστη. Με αυτές τις εργασίες στο δίκτυο, η ίδια η άκρη (edge) πρέπει να είναι καλά σχεδιασμένη για να ικανοποιεί τις απαιτήσεις σε μια υπηρεσία αποτελεσματικά, όπως την αξιοπιστία, την ασφάλεια και την προστασία της ιδιωτικής ζωής [67].



Εικόνα 5.1 Παράδειγμα υπολογιστικής των άκρων

5.2 Πλεονεκτήματα της Υπολογιστικής των Άκρων

Στην υπολογιστική των άκρων πρέπει να τεθεί η υπολογιστική κοντά στην πηγή των δεδομένων. Αυτό έχει πολλά οφέλη σε σύγκριση με το παραδοσιακό πρότυπο υπολογιστικής που βασίζεται στην υπολογιστική νέφους. Εδώ χρησιμοποιούνται αρκετά πρώιμα αποτελέσματα από την ερευνητική κοινότητα για να αναδειχθούν τα πιθανά οφέλη. Οι ερευνητές δημιούργησαν μια δοκιμαστική πλατφόρμα για να εκτελέσουν την εφαρμογή αναγνώρισης προσώπου στο [64] και ο χρόνος απόκρισης μειώνεται από 900 σε 169 ms μετακινώντας τον υπολογισμό από το σύννεφο στην άκρη. Σύμφωνα με το [65], οι Ha et al. χρησιμοποίησαν cloudlets για την εκφόρτωση υπολογιστικών εργασιών για φορετή γνωστική βοήθεια και το αποτέλεσμα δείχνει ότι η βελτίωση του χρόνου απόκρισης κυμαίνεται μεταξύ 80 και 200ms. Επιπλέον, η κατανάλωση ενέργειας θα μπορούσε επίσης να μειωθεί κατά 30% -40% με εκφόρτωση νέφους. Ο κλώνος συννέφου (clone cloud) στο [66] συνδυάζει την κατάτμηση, τη μετεγκατάσταση με τη συγχώνευση και την κατά παραγγελία κατάτμηση μεταξύ κινητού και cloud, και τη μέθοδός τους θα μπορούσε να μειώσει τον χρόνο λειτουργίας και την ενέργεια κατά 20 φορές για δοκιμασμένες εφαρμογές [67].

6. Προτεινόμενο μοντέλο HBPSHPO

6.1 Εισαγωγή

Η εικονικοποίηση των πόρων έχει ριζοσπαστικοποιήσει τον κόσμο των υπολογιστών ειδικά κατά την τελευταία δεκαετία. Η τεχνολογία έχει προχωρήσει πολύ στην αναπαραγωγή, τη μετανάστευση, την αναπαραγωγή πόρων κατά ζήτηση, αλλά υπάρχει ένα βασικό πρόβλημα που παραμένει: η πρόβλεψη χρήσης πόρων. Η ακριβής πρόβλεψη των πόρων που θα απαιτήσει μια υπολογιστική εργασία προκειμένου να εκτελεστεί με επιτυχία παραμένει κύριο ζητούμενο σε κάθε αγωγό υπηρεσίας cloud. Η έλευση της υπολογιστικής ακμών χρησιμεύει μόνο στην ενίσχυση της σημαντικότητας μιας πιθανής λύσης. Οι πόροι ακμής είναι συνήθως περιορισμένης χωρητικότητας και η έξυπνη κατανομή τους σε υπολογιστικές εργασίες είναι υψίστης σημασίας.

Η βιβλιογραφία ανθίζει με προτάσεις που χρησιμοποιούν μεθόδους όπως σημεία αναφοράς, στατιστική ανάλυση, ανάλυση κώδικα και άλλες παρόμοιες προσεγγίσεις. Τον τελευταίο καιρό, η μηχανική μάθηση έχει κερδίσει ένα αυξανόμενο μερίδιο στις προτάσεις. Ακολουθώντας αυτήν την τάση και αξιοποιώντας τις πρόσφατες εξελίξεις στον τομέα της βαθιάς μάθησης, προτείνουμε μια προσέγγιση για την πρόβλεψη του υπολογιστικού φορτίου μιας συγκεκριμένης εργασίας σε πόρους ακμής. Οι μέθοδοι πρόβλεψης που προτείνουμε βασίζονται εξ' ολοκλήρου σε ιστορικά δεδομένα παρακολούθησης, αποσυνδέοντας έτσι την πρόβλεψη από την υλοποίηση της εφαρμογής και τα χαρακτηριστικά των πόρων.

Η βασική προϋπόθεση στην προσέγγισή μας είναι ότι το πρόβλημα της πρόβλεψης της χρήσης πόρων των κόμβων ακμής μπορεί να διαμορφωθεί ως πρόβλημα παλινδρόμησης χρονοσειρών. Τα διαδοχικά δεδομένα ενδέχεται να παρουσιάζουν ομοιότητες ή εξαρτήσεις μεταξύ τους. Τα επαναλαμβανόμενα νευρωνικά δίκτυα (RNN) και τα Convolutional Neural Networks (CNN) είναι σε θέση να αναγνωρίσουν τα παρόμοια μοτίβα δεδομένων και να τα εκμεταλλευτούν προκειμένου να αντιμετωπίσουν αποτελεσματικά το πρόβλημα της χρονικής πρόβλεψης χρήσης πόρων.

Προτείνουμε μια μέθοδο που εκτελεί μια σειρά εργασιών που ξεκινούν με την παρακολούθηση της τρέχουσας χρήσης πόρων των κόμβων ακμής και της αξιοποίησης ιστορικών δεδομένων. Στη συνέχεια, οι παρατηρήσεις παρακολούθησης δέχονται προ-επεξεργασία και τροφοδοτούνται σε ένα CNN παλινδρόμησης πολλαπλών εξόδων. Το πιο δύσκολο έργο σε αυτήν τη ροή εργασίας είναι η εκπαίδευση ενός CNN παλινδρόμησης πολλαπλών εξόδων. Αυτή η πρόκληση περιλαμβάνει την επιλογή της βέλτιστης ή πλησίον της βέλτιστης τοπολογίας του CNN, την εκτίμηση των βαρών και των προκαταλήψεων των συνάψεων του CNN, δηλαδή των παραμέτρων του CNN και πολλών άλλων αποφάσεων αρχιτεκτονικού σχεδιασμού, όπως οι συναρτήσεις ενεργοποίησης και το ποσοστό απόρριψης. Οι αποφάσεις αρχιτεκτονικού σχεδιασμού ονομάζονται υπερ-παράμετροι.

Η εκπαίδευση ξεκινά αναζητώντας τις βέλτιστες αριθμητικές υπερ-παραμέτρους του μοντέλου CNN με τον αλγόριθμο Particle Swarm Optimization (PSO) και τις ονομαστικές υπερ-παραμέτρους χρησιμοποιώντας το Bayesian Optimization (BO). Η προτεινόμενη υβριδική μέθοδος βελτιστοποίησης που συνδυάζει PSO και BO για την εκτίμηση του βέλτιστου μοντέλου CNN έχει τίτλο Hybrid Bayesian Particle Swarm Hyper-parameter Optimization (HBPSHPO). Το HBPSHPO πραγματοποιεί πολλαπλές επαναλήψεις μεταξύ των προτεινόμενων τοπολογιών CNN και τις αξιολογεί βάσει ιστορικών δεδομένων. Όπως θα περιγράψουμε στη θεωρητική ενότητα του κεφαλαίου και τα πειραματικά αποτελέσματα, το HBPSHPO κάνει μια έξυπνη αναζήτηση στις τοπολογίες του CNN συγκλίνοντας γρήγορα και τελικά το εκπαιδευμένο μοντέλο παλινδρόμησης πολλαπλών εξόδων CNN κάνει ακριβείς προβλέψεις για τη χρήση των πόρων για τα επόμενα βήματα.

Προκειμένου να αξιολογηθεί το προτεινόμενο μοντέλο, πραγματοποιήσαμε πραγματικά πειράματα με μια υποδομή υπολογιστικής ακμής που αποτελείται από ένα σύμπλεγμα από πίνακες Raspberry PI που εκτελούν μια εφαρμογή επεξεργασίας φυσικής γλώσσας (NLP). Αναπτύξαμε έναν κώδικα python στις συσκευές ακμής για να παρακολουθήσουμε τη χρήση των πόρων τους και δημιουργήσαμε ένα σύνολο δεδομένων εκπαίδευσης και αξιολόγησης. Στη συνέχεια, πραγματοποιήσαμε πειράματα με το μοντέλο παλινδρόμησης πολλαπλών εξόδων CNN, το οποίο χρησιμοποιεί το μοντέλο HBPSHPO και συγκρίναμε τα αποτελέσματα με άλλους τρόπους

χρησιμοποίησης πόρων και μηχανικής μάθησης μοντέλων πρόβλεψης που είναι διαθέσιμα στη βιβλιογραφία. Τα αποτελέσματα της αξιολόγησης δείχνουν ότι το προτεινόμενο μοντέλο ξεπερνά την προηγούμενη εργασία όσον αφορά την ακρίβεια των προβλέψεων, δηλαδή το ριζικό μέσο τετραγωνικό σφάλμα και το μέσο απόλυτο σφάλμα και επιβεβαιώνουν την εφαρμογή της μεθόδου μας.

Οι δύο σημαντικές συνεισφορές της έρευνάς μας είναι:

- ➢ Η αξιολόγηση της εφαρμοσιμότητας και η ακρίβεια της παλινδρόμησης πολλαπλών εξόδου με CNN ως μέσο πρόβλεψης της χρήσης των πόρων στις υποδομές υπολογιστών ακμής.
- ➢ Η πρόταση ενός καινοτόμου υβριδικού μοντέλου βελτιστοποίησης υπερ-παραμέτρων που συνδυάζει τον αλγόριθμο PSO με τη μέθοδο BO προκειμένου να εκτιμηθεί η βέλτιστη τοπολογία του νευρωνικού δικτύου.

Το υπόλοιπο κεφάλαιο είναι δομημένο ως εξής: Η Ενότητα 6.2 επισημαίνει τη σχετική εργασία στην πρόβλεψη χρήσης πόρων, τη σχετική μηχανική μάθηση και τις τεχνικές βελτιστοποίησης υπερ-παραμέτρων. Το Τμήμα 6.3 εξηγεί τη λειτουργικότητα της πρόβλεψης χρήσης πόρων στη διαχείριση υποδομών υπολογιστών ακμής και παρέχει μια ανάλυση της παλινδρόμησης πολλαπλών εξόδων με CNN και της μεθόδου HBPSHPO. Η Ενότητα 6.4 περιγράφει την πειραματική ρύθμιση και τα αποτελέσματα της αξιολόγησης. Τέλος, η Ενότητα 6.5 ολοκληρώνει την εργασία και προτείνει οδηγίες για μελλοντικές εργασίες.

6.2 Σχετικές Εργασίες

Η πρόβλεψη χρήσης πόρων σχετίζεται συχνά με την «πρόβλεψη φόρτου εργασίας» στο πλαίσιο του cloud computing. Μια τέτοια πρόβλεψη συνήθως συνδέεται πολύ με την εργασία εφαρμογής, κάτι που περιορίζει την εφαρμοσιμότητα σχετικών προσεγγίσεων.

Αυτό ισχύει ιδιαίτερα για κάποια βιβλιογραφία όπου οι συγγραφείς υποθέτουν ότι υπάρχουν μοτίβα φόρτου εργασίας που εμφανίζονται σε διάφορες εργασίες του cloud. Η βασική πρόκληση είναι ο εντοπισμός των μοτίβων [71], ή των μοντέλων φόρτου εργασίας [72] και ο χαρακτηρισμός του φόρτου εργασίας με βάση το αν τα "συμπεριλαμβάνουν" ή όχι. Τέτοια μοντέλα είναι πολύ χρήσιμα ως ορόσημα στις τεχνικές μοντελοποίησης επιδόσεων [73]. Ωστόσο, όπως αναφέρεται στο [74]: «δεδομένης της ευρείας έννοιας του φόρτου εργασίας και των στόχων, της ανεπάρκειας συνόλων δεδομένων φόρτου εργασίας για δημόσια χρήση και της ταχείας εξέλιξης της τεχνολογίας cloud, τέτοια έργα φαίνεται να έχουν διαφορετικούς βαθμούς ωριμότητας» και ως εκ τούτου εμείς επιλέξαμε να μην επικεντρωθούμε σε αυτά.

Η πραγματοποίηση προβλέψεων με βάση τα δεδομένα παρακολούθησης φαίνεται να ανακουφίζει αυτό το πρόβλημα παρέχοντας τη βάση για μεθόδους πρόβλεψης φόρτου εργασίας ανεξάρτητες από εφαρμογές και πόρους. Σε αυτή τη βάση, υπάρχει ένα αυξανόμενο σώμα βιβλιογραφίας που αντιμετωπίζει το πρόβλημα χρησιμοποιώντας μηχανικές μάθησης και στατιστικές αναλύσεις. Εδώ και πολύ καιρό, έχει δοκιμαστεί ένα μεγάλο σύνολο διαμορφώσεων, συμπεριλαμβανομένων των απλών perceptrons [75] και της γραμμικής παλινδρόμησης [74], των μοντέλων ARIMA [76] και της βαθιάς μάθησης με έμφαση στη μείωση της διαστατικότητας του προβλήματος [77].

Το κοινό μειονέκτημα αυτών των προσεγγίσεων είναι ότι πρέπει να αναπτυχθεί ένα νέο μοντέλο για κάθε ζεύγος τύπου εφαρμογών-πόρων. Για να αντιμετωπίσουν το τελευταίο πρόβλημα και να δημιουργήσουν πιο γενικές λύσεις, οι ερευνητές έχουν αναπτύξει μη εποπτευόμενες μαθησιακές προσεγγίσεις όπως η Ανάλυση Κύριων Συστατικών (PCA) για τον εντοπισμό προτύπων χρήσης πόρων σε διαφορετικούς κόμβους.

Η διαχείριση των υποδομών υπολογιστών ακμής περιλαμβάνει ένα σύνολο στρατηγικών προληπτικού προγραμματισμού σε πραγματικό χρόνο [79] για να διασφαλιστεί η ομαλή λειτουργία των εφαρμογών, να διασφαλιστεί η ποιότητα των υπηρεσιών, να ελαχιστοποιηθεί ο υπολογισμός της καθυστέρησης και να βελτιστοποιηθεί η κατανομή εύρους ζώνης, η κατανάλωση ενέργειας και το συνολικό κόστος. Η διακύμανση του υπολογισμού του φόρτου εργασίας, η δυναμικότητα της συμπεριφοράς των χρηστών και η ετερογένεια των κόμβων ακμής ωθούν την ερευνητική κοινότητα να προτείνει ευφυείς και αυτοματοποιημένους μηχανισμούς. Οι μέθοδοι βαθιάς μάθησης έχουν χρησιμοποιηθεί ευρέως τον τελευταίο καιρό για την ενοποίηση των υποδομών ακμής και νέφους με

τα βασικά μοντέλα εκφόρτωσης εργασιών [80], διαχείρισης πόρων [81] και μετεγκατάστασης υπηρεσιών [82] να είναι μόνο μερικά.

Η εκτέλεση εργασιών σε υπολογιστικούς πόρους cloud ή ακμής συνήθως δεν κλιμακώνεται γραμμικά με την εισαγωγή τους, καθιστώντας τα μοντέλα γραμμικής χρήσης πόρων ανεπαρκή και τις μηχανικές λύσεις μάθησης να είναι μια λογική προσέγγιση. Τα νευρωνικά δίκτυα παλινδρόμησης μίας εξόδου [83] έχουν χρησιμοποιηθεί για την πρόβλεψη χρήσης πόρων σε μια δεδομένη εργασία και ιστορικά δεδομένα. Οι παρατηρήσεις χρήσης πόρων μπορούν να αναπαρασταθούν και να αναλυθούν ως δεδομένα χρονοσειρών. Το προηγούμενο απλό μοντέλο εμπρόσθιας τροφοδότησης έχει τον περιορισμό πως δεν μπορεί να αξιοποιήσει τη διαδοχική δομή των πληροφοριών. Το Long Short-Term Memory (LSTM) [84] είναι ένα συγκεκριμένο επίπεδο RNN που αξιοποιεί χρονοσειρές και έχει χρησιμοποιηθεί για τη μοντελοποίηση της σχέσης μεταξύ κατανομής πόρων και χρήσης μνήμης και CPU ενός δεδομένου φόρτου εργασίας.

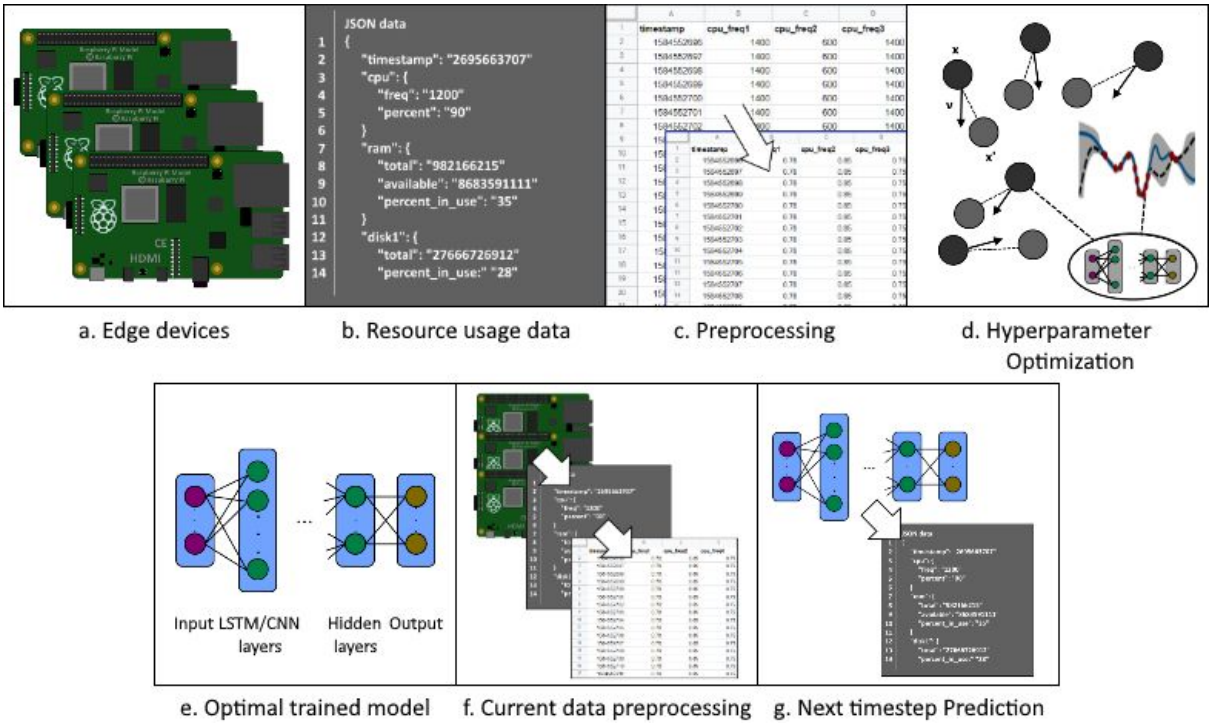
Το μοντέλο LSTM χρησιμοποιεί ένα παράθυρο παρατήρησης σταθερού μεγέθους. Εάν αυτό το παράθυρο είναι σύντομο, τότε δεν μπορεί να καταγράψει κάποιες αλλαγές στις τάσεις χρήσης πόρων. Εάν το παράθυρο σταθερού μεγέθους είναι ασύγκριτα μεγάλο, τότε ένας άσχετος μεγάλος αριθμός παρατηρήσεων μπορεί να αποφέρει ανακριβείς εκτιμήσεις. Για την αντιμετώπιση αυτού του περιορισμού έχει προταθεί μια προσαρμοσμένη επιλογή μεγέθους παραθύρου βασισμένη στη μάθηση [85] για την πρόβλεψη χρήσης πόρων που αξιοποιεί διαφορετικά μεγεθη τάσεων περιόδου. Ένα ελαφρώς διαφορετικό μοντέλο πρόβλεψης χρήσης προσαρμοστικών πόρων δημιουργεί πολλαπλές αναπαραστάσεις δεδομένων που εξάγονται από πολλαπλά μεγέθη παραθύρων και χρησιμοποιεί ένα σύνολο μοντέλων κλασικής μηχανικής μάθησης με προσέγγιση δασών τυχαίας απόφασης [86]. Τα δάση τυχαίας απόφασης ανήκουν στην κατηγορία των εκπαιδευόμενων και των μετα-εκπαιδευόμενων. Οι μέθοδοι συνόλων έχουν επίσης συνδυαστεί με τεχνικές τεχνικών χαρακτηριστικών [87] προκειμένου να παρέχονται ακριβείς προβλέψεις χρήσης πόρων με τρόπο μετα-μάθησης.

Λαμβάνοντας υπόψη τις προαναφερθείσες πρόσφατες εξελίξεις στη διαχείριση πόρων υπολογιστικής ακμής, υπάρχει μεγάλο ενδιαφέρον: (i) πώς να χρησιμοποιηθούν μοντέλα βαθιάς μάθησης ως ακριβείς προβλέψεις παλινδρόμησης. (ii) να αξιοποιηθεί η χρονολογική σειρά των παρατηρήσεων χρήσης πόρων με προσαρμοστικό τρόπο · (iii) να χρησιμοποιηθεί μια προσέγγιση μετα-μάθησης για την αναζήτηση ενός βέλτιστου μοντέλου. Προκειμένου να ικανοποιηθούν αυτές οι τρεις απαιτήσεις, προτείνουμε ένα μοντέλο βαθιάς μάθησης με επίπεδα CNN και ένα καινοτόμο υβριδικό μοντέλο βελτιστοποίησης υπερ-παραμέτρων ως αλγόριθμο μετα-μάθησης.

Το προτεινόμενο μοντέλο μας, από όσο γνωρίζουμε, είναι διαφοροποιημένο και ξεπερνά τη σχετική εργασία για προβλέψεις χρήσης πόρων σε υποδομές υπολογιστών ακμής για δύο λόγους. Πρώτον, χρησιμοποιούμε παλινδρόμηση πολλαπλών εξόδων CNN για να προβλέψουμε τη χρήση πολλαπλών πόρων με ενιαίο τρόπο και να αξιοποιήσουμε τις παρατηρήσεις χρονοσειρών σε ένα προσαρμοστικό μέγεθος παραθύρου. Δεύτερον, προτείνουμε ένα καινοτόμο υβριδικό μοντέλο μετα-μάθησης που συνδυάζει PSO και BO προκειμένου να εκτιμηθεί η προσέγγιση των βέλτιστων ονομαστικών και αριθμητικών υπερ-παραμέτρων βαθιάς μάθησης. Η ανάλυση και η χρήση των επιπέδων CNN, PSO και BO συζητούνται σε ξεχωριστές ενότητες σε αυτό το κεφάλαιο.

6.3. Μια προσέγγιση του Convolutional Neural Network

Η πρόβλεψη της χρήσης πόρων των κόμβων ακμών σε μια υποδομή υπολογιστών ακμής μπορεί να περιλαμβάνει πολλά χαρακτηριστικά από διαφορετικές συσκευές. Κάθε χαρακτηριστικό δεδομένων παρουσιάζει μια διαδοχική δομή και τα χαρακτηριστικά χρήσης πόρων έχουν εγγενείς εξαρτήσεις το ένα με το άλλο. Προκειμένου να αξιοποιήσουμε τα συγκεκριμένα χαρακτηριστικά δεδομένων που παρακολουθούνται από κόμβους ακμής και τις τιμές-στόχους της χρήσης πόρων της επόμενης χρονικής στιγμής, προτείνουμε τη χρήση ενός μονοδιάστατου CNN παλινδρόμησης πολλαπλών εξόδων. Εμείς χρησιμοποιήσαμε παλινδρόμηση πολλαπλών εξόδων επειδή προβλέπουμε για κάθε συσκευή τη χρήση πολλαπλών εξόδων, όπως CPU, μνήμη, χρήση εύρους ζώνης ως βαθμωτές τιμές και είναι ένα μονοδιάστατο συνελικτικό δίκτυο, επειδή το νευρωνικό δίκτυο συλλαμβάνει τις διαδοχικές πληροφορίες.



Εικόνα 6.1 Αγωγός εκπαίδευσης και συμπερασμάτων με παλινδρόμηση πολλαπλών εξόδων CNN με το HBPSHPO

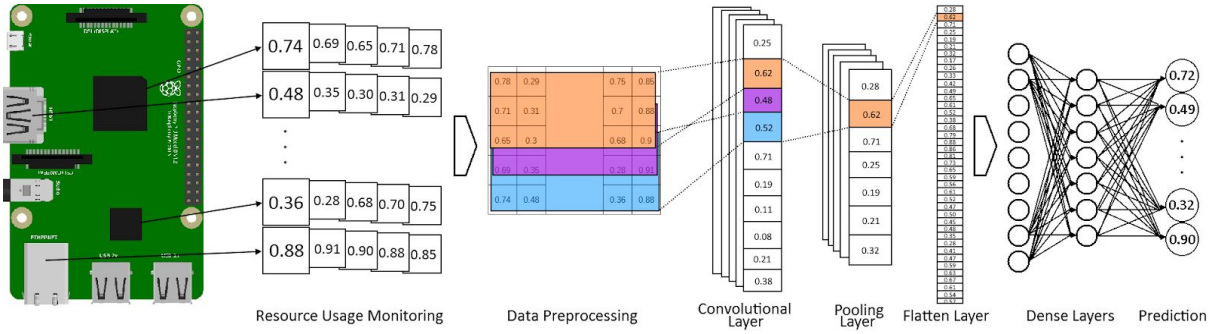
Τα μοντέλα νευρωνικών δικτύων είναι γεμάτα παραμέτρους και υπερ-παραμέτρους. Οι παράμετροι περιλαμβάνουν τα βάρη και τις προκαταλήψεις των συνδέσεων μεταξύ των νευρώνων. Οι παράμετροι υπολογίζονται χρησιμοποιώντας έναν αλγόριθμο μάθησης όπως η κατάβαση κλίσης και οι παραλλαγές του. Οι υπερ-παράμετροι καθορίζουν τη δομή του δικτύου και τον τρόπο εκπαίδευσης αυτού του δικτύου. Η δομή του δικτύου περιλαμβάνει τον αριθμό των στρωμάτων και τον αριθμό των νευρώνων για κάθε στρώμα. Οι αποφάσεις σχετικά με τον τρόπο εκπαίδευσης του δικτύου ενδέχεται να περιλαμβάνουν την επιλογή αλγορίθμου μάθησης, το ρυθμό μάθησης και το μέγεθος της παρτίδας. Η επιλογή των υπερ-παραμέτρων δεν είναι ασήμαντη και μια προσέγγιση δοκιμής και σφάλματος μπορεί να μας οδηγήσει σε μεγάλο αριθμό υπολογιστικών δαπανηρών δοκιμών με μεγάλη αβεβαιότητα για τις επιλεγμένες υπερ-παραμέτρους. Για να κάνουμε μια έξυπνη αναζήτηση στο χώρο των υπερ-παραμέτρων θα χρησιμοποιήσουμε το μοντέλο HBPSHPO που εκτιμά τις ονομαστικές υπερ-παραμέτρους, όπως οι συναρτήσεις ενεργοποίησης με μια προσέγγιση Μπεϋζιανής βελτιστοποίησης και τις αριθμητικές υπερ-παραμέτρους όπως ο αριθμός των επιπέδων με μια προσέγγιση βασισμένη στη βελτιστοποίηση σμήνους σωματιδίων.

Στις επόμενες ενότητες θα περιγράψουμε τα βασικά μέρη του μοντέλου παλινδρόμησης CNN πολλαπλών εξόδων, συμπεριλαμβανομένης της διαδικασίας κανονικοποίησης και μάθησης, το LSTM που επίσης εξετάστηκε ως πιθανή προσέγγιση, αλλά τελικά είχαν χαμηλότερη ακρίβεια από το CNN και το μοντέλο HBPSHPO.

6.3.1 Convolutional Neural Network

Το Convolutional Neural Network (CNN) είναι ένας τύπος νευρωνικού δικτύου που είναι κατάλληλος για την επεξεργασία δεδομένων που έχουν δομή πλέγματος. Η αρχιτεκτονική του αποτελείται συνήθως από ένα ή περισσότερα συνελικτικά στρώματα με στάδια δειγματοληψίας και από ένα ή περισσότερα πλήρως συνδεδεμένα στρώματα όπως συμβαίνει σε κοινά νευρωνικά δίκτυα πολλαπλών επιπέδων. Τα CNN χρησιμοποιούν έναν βελτιστοποιημένο εξαγωγέα χαρακτηριστικών που ονομάζεται πυρήνας, ο οποίος επιτρέπει στα CNN να απομνημονεύουν εύκολα τη χωρική σειρά των χαρακτηριστικών. Το τελευταίο βοηθά τα CNN να αποδίδουν σημαντική ακρίβεια και αποτελεσματικότητα. Η συνέλιξη πραγματοποιείται μεταξύ της μήτρας δεδομένων και της μήτρας του πυρήνα, με το παράθυρο της συνέλιξης να κινεί στοιχεία *n* για κάθε πολλαπλασιασμό,

παράμετρο n που ονομάζεται strides. Μετά το συνελικτικό επίπεδο, μια συνήθης πρακτική είναι η προσθήκη ενός συγκεντρωτικού στρώματος, είτε ενός μέγιστου είτε ενός μέσου όρου. Ο σκοπός του στρώματος συγκέντρωσης είναι να μειώσει τη διάσταση και, συνεπώς, την υπολογιστική ισχύ που απαιτείται για την επεξεργασία των δεδομένων, καθώς και για την καταστολή του θορύβου. Μετά τα στρώματα της συνέλιξης και της συγκέντρωσης, τα δεδομένα ισοπεδώνονται σε ένα, και έτσι μπορούν να χρησιμοποιηθούν ως είσοδος σε ένα πλήρως συνδεδεμένο δίκτυο εμπρόσθιας τροφοδοσίας, το οποίο μπορεί να βοηθήσει στην εκμάθηση μη γραμμικών μοτίβων και χαρακτηριστικών.



Εικόνα 6.2 Convolution Neural Networks για πρόβλεψη χρήσης πόρων

Στην περίπτωση της πρόβλεψης χρήσης πόρων, όπως μπορούμε να δούμε στο Σχήμα 6.2, χρησιμοποιήθηκαν CNN με μονοδιάστατα συνελικτικά στρώματα με μέγιστα συγκεντρωτικά επίπεδα. Τα δεδομένα εισαγωγής περιλαμβάνουν μια χρονοσειρά που δείχνει την ποσοστιαία χρήση κάθε πόρου που διατίθεται στο σύστημά μας, όπως CPU, RAM, εύρος ζώνης. Η ιδέα είναι ότι τα μονοδιάστατα συνελικτικά στρώματα θα βοηθήσουν στην εύρεση μοτίβων που αναφέρονται στη χρήση των πόρων μας, εκτελώντας την επίλυση στην χρονική διάσταση. Συγκεκριμένα, αφού χωρίσουμε τις σειρές δεδομένων του συνόλου δεδομένων μας σε σύνολα εκπαίδευσης και δοκιμών, το μετατρέπουμε σε δείγματα διαστάσεων (n , χαρακτηριστικά). Αυτό τροφοδοτείται στο δίκτυό μας, το οποίο εκτελεί μονοδιάστατες συνελίξεις σε κάθε δείγμα, με την ιδέα ότι η χρήση πόρων ενός συγκεκριμένου χρονικού βήματος έχει πολύ υψηλότερη συσχέτιση με τις πιο πρόσφατες μετρήσεις της χρήσης πόρων σε σύγκριση με προηγούμενες. Με αυτόν τον τρόπο το δίκτυο προσπαθεί να προβλέψει τις τιμές των χαρακτηριστικών του επόμενου χρονικού βήματος χρησιμοποιώντας τις τελευταίες τιμές n του συνόλου δεδομένων.

6.3.2 Long short-term memory

Το Long short-term memory (LSTM) είναι ένας τύπος RNN που περιέχει μπλοκ μνήμης στα αναδρομικά κρυμμένα επίπεδα. Συγκεκριμένα, μια μονάδα LSTM περιλαμβάνει το κελί, την πύλη εισόδου, την πύλη εξόδου και την πύλη λήθης, τα οποία διασφαλίζουν την αποθήκευση και την πρόσβαση σε προηγούμενα δεδομένα, ακόμη και μετά το πέρασμα μεγάλων χρονικών περιόδων.

Χάρη στην ικανότητά του να συλλαμβάνει μακροχρόνιες χρονικές εξαρτήσεις, το LSTM είναι κατάλληλο για προβλήματα χρονοσειρών. Είναι επίσης ικανό να αντιμετωπίσει δύο κοινά προβλήματα των RNN: τη βραχυπρόθεσμη μνήμη (short-term memory problem) και το πρόβλημα της κλισης που εξαφανίζεται (vanishing gradient problem).

Το κύριο μειονέκτημα του LSTM για την πρόβλεψη χρήσης πόρων είναι ότι χρησιμοποιεί παράθυρα παρατήρησης σταθερού μεγέθους που δεν μπορούν να προσαρμοστούν για να καταγράψουν τοπικές τάσεις στις πιο πρόσφατες παρατηρήσεις ή απόμακρες τάσεις στις παλαιότερες ακολουθίες δεδομένων [85].

6.3.3 Κανονικοποίηση

Ένα νευρωνικό δίκτυο πρέπει να είναι σε θέση να γενικεύει σε νέες εισόδους δεδομένων που δεν έχουν ξαναδεί. Μια πρακτική κανονικοποίησης που χρησιμοποιείται συχνά είναι η τεχνική

απόρριψης, στην οποία ένα ποσοστό των αναδρομικών συνδέσεων αποκλείονται από την ενεργοποίηση κατά τη διάρκεια κάθε εκμάθησης. Η πρόωρη διακοπή είναι επίσης μια στρατηγική κανονικοποίησης στην οποία αξιολογούμε το μοντέλο σε κάθε επανάληψη της εκπαίδευσης και όταν η ακρίβεια μειώνεται στο σύνολο δεδομένων δοκιμών, η εκπαίδευση σταματά.

6.3.4 Διαδικασία Μάθησης

Οι παράμετροι ενός νευρωνικού δικτύου μπορούν να βελτιστοποιηθούν ελαχιστοποιώντας μια αντικειμενική συνάρτηση. Η αντικειμενική συνάρτηση εκφράζει τη διαφορά μεταξύ της προβλεπόμενης εξόδου και της πραγματικής εξόδου, όπως το μέσο απόλυτο σφάλμα L1 και το μέσο τετραγωνικό σφάλμα L2. Οι αντικειμενικές συναρτήσεις μπορούν επίσης να συνδυαστούν με τεχνικές βάρους ποινής προκειμένου να κανονικοποιηθεί το μοντέλο. Με βάση την υπάρχουσα βιβλιογραφία, οι πιο ευρέως χρησιμοποιούμενες τεχνικές βελτιστοποίησης για τα CNN-RNN και LSTM-RNN είναι το Root Mean Square Propagation (RMPSrop) και η Adaptive Moment Estimation (Adam) [78]. Το RMSProp και ο Adam είναι και οι δύο προσεγγίσεις στοχαστικής κατάβασης κλίσης με προσαρμοστικό ρυθμό μάθησης. Το RMSProp χρησιμοποιεί την προσέγγιση Momentum, στην οποία η κλίση σε κάθε επανάληψη είναι το άθροισμα της τρέχουσας κλίσης και των προηγούμενων κλίσεων, με αποτέλεσμα τον περιορισμό της ταλάντωσης. Ο Adam χρησιμοποιεί παρόμοια την τεχνική Momentum, αλλά βρίσκει επίσης μια αναλλοίωτη κατεύθυνση κλίσης σε αντίθεση με τις άλλες ταλαντευόμενες κατευθύνσεις, με την πλοήγηση μέσω σημείων σέλας.

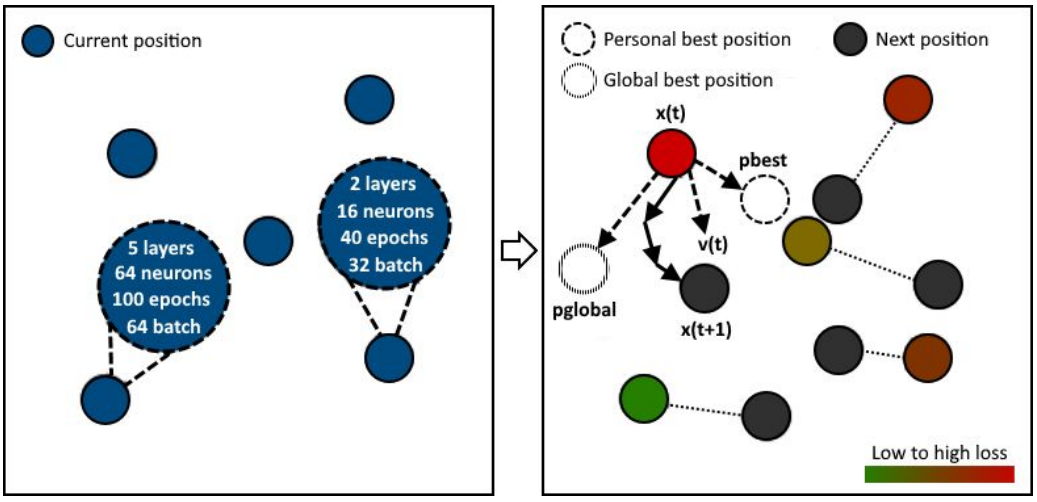
6.3.5 Particle Swarm Optimization

Το Particle Swarm Optimization (PSO) [90] είναι ένας εξελικτικός αλγόριθμος που στοχεύει στον εντοπισμό των βέλτιστων υπερ-παραμέτρων και της ελάχιστης τιμής της αντικειμενικής συνάρτησης. Οι εξελικτικοί αλγόριθμοι είναι μεταευρετικοί αλγόριθμοι βελτιστοποίησης που προσομοιώνουν μηχανισμούς εμπνευσμένους από τη φύση. Το PSO προσπαθεί να μιμηθεί τη συμπεριφορά ενός σμήνους πουλιών ή ενός κοπαδιού ψαριού. Με άλλα λόγια, τα «πουλιά» πετούν μέσω ενός N-διάστατου χώρου αναζήτησης, όπου κάθε διάσταση αντιπροσωπεύει μία μόνο υπερ-παράμετρο και κάθε θέση στον χώρο αναζήτησης αντιστοιχεί σε ένα σύνολο υπερπαραμέτρων από το οποίο μπορούμε να εξαγάγουμε την τιμή σφάλματος της αντικειμενικής λειτουργίας μέσω μιας διαδικασίας εκπαίδευσης και αξιολόγησης νευρωνικών δικτύων. Τα κύρια στοιχεία του αλγορίθμου PSO συνοψίζονται ως εξής.

Πρώτον, το PSO χρειάζεται μια αντικειμενική συνάρτηση για ελαχιστοποίηση. Η αντικειμενική συνάρτηση σε βαθιά νευρωνικά δίκτυα συνήθως περιλαμβάνει τη διαδικασία εκπαίδευσης του δικτύου που επιστρέφει την τιμή απώλειας της μεθόδου αξιολόγησης. Δεύτερον, πρέπει να καθορίσουμε τον αριθμό των πρακτόρων αναζήτησης - σωματιδίων στο σμήνος (μέγεθος σμήνους), τα οποία αντιπροσωπεύουν πιθανές λύσεις του προβλήματος και τον μέγιστο αριθμό επαναλήψεων τις οποίες πρόκειται να "τρέξει" το σμήνος. Οι πράκτορες εξερευνούν το χώρο αναζήτησης σε κάθε επανάληψη προκειμένου να βρουν την ελάχιστη τιμή της αντικειμενικής συνάρτησης, ενημερώνοντας τη θέση τους x και την ταχύτητα v σε κάθε επανάληψη. Κατά τη διαδικασία αναζήτησης, κάθε πράκτορας αποθηκεύει την προσωπική του καλύτερη τιμή p_i και τον συνδυασμό υπερ-παραμέτρων που παρήγαγε αυτή την τιμή. Οι πράκτορες μπορούν να αλληλεπιδρούν μεταξύ τους και έτσι, σε περίπτωση που η προσωπική καλύτερη τιμή ενός πράκτορα είναι η καλύτερη μεταξύ όλων των άλλων πρακτόρων μέχρι αυτό το σημείο, αυτός ο πράκτορας ενημερώνει τους υπόλοιπους πράκτορες και αποθηκεύει την τιμή του ως την παγκόσμια καλύτερη τιμή. Όπως σε οποιοδήποτε πρόβλημα βελτιστοποίησης υπερ-παραμέτρων, δεν υπάρχει αντικειμενικά καλύτερη τιμή μεγέθους σμήνους που πρέπει να επιλέξει κανείς, καθώς το βέλτιστο μέγεθος σμήνους εξαρτάται από το εκάστοτε πρόβλημα. Ωστόσο, η χρήση μεγάλου αριθμού σωματιδίων μπορεί να οδηγήσει σε αργή σύγκλιση χωρίς αισθητή βελτίωση της απόδοσης.

Τρίτον, επιλέγουμε τις υπερ-παραμέτρους του PSO ω, φ_p και φ_g που αντιπροσωπεύουν τον συντελεστή κλιμάκωσης ταχύτητας σωματιδίων, τον παράγοντα κλιμάκωσης για αναζήτηση μακριά από την καλύτερη γνωστή θέση κάθε σωματιδίου και τον παράγοντα κλιμάκωσης για αναζήτηση μακριά από την καλύτερη γνωστή θέση του σμήνους αντίστοιχα. Αυτές οι υπερ-παράμετροι

χρησιμοποιούνται για να αντιμετωπίσουν δύο σημαντικές έννοιες κάθε αλγορίθμου βελτιστοποίησης: την εξερεύνηση και την εκμετάλλευση. Το πρώτο αναφέρεται στην ικανότητα των σωματιδίων να αναζητούν μακριά από την τρέχουσα θέση τους, ενώ το δεύτερο αντιπροσωπεύει την ικανότητα των σωματιδίων να αναζητούν την γύρω περιοχή της τρέχουσας θέσης τους. Συγκεκριμένα, το ω που ονομάζεται επίσης αδράνεια είναι μια θετική σταθερά, συνήθως μεταξύ 0,4 και 0,9, η οποία μειώνεται καθώς εξελίσσεται ο αλγόριθμος. Το ω ορίζει το χαρακτηριστικό της εξερεύνησης καθορίζοντας την επίδραση που είχε η προηγούμενη ταχύτητα των πρακτόρων στην επόμενη θέση. Οι παράμετροι φ_p, φ_g είναι μη αρνητικοί συντελεστές που καθορίζουν το βάρος επιτάχυνσης προς την προσωπική και παγκόσμια καλύτερη λύση αντίστοιχα. Μειώνοντας το φ_p μειώνεται επίσης η εκμετάλλευση του χώρου αναζήτησης, ενώ καθώς μειώνουμε φ_g, μειώνεται η εξερεύνηση. Η ισορροπία μεταξύ εξερεύνησης και εκμετάλλευσης επιτυγχάνεται συντονίζοντας τα ω, φ_p και φ_g κατάλληλα.



Εικόνα 6.3 Τα σωματίδια κάνουν μια έξυπνη αναζήτηση

Ο αλγόριθμος PSO συνοψίζεται ως εξής. Πρώτον, αρχικοποιούμε τυχαία τις θέσεις και τις ταχύτητες των σωματιδίων. Στη συνέχεια, εκτελούμε τη διαδικασία εκπαίδευσης που παράγει την τιμή απώλειας για κάθε σωματίδιο και ενημερώνουμε τα προσωπικά και παγκόσμια καλύτερα αποτελέσματα, εάν είναι απαραίτητο. Στη συνέχεια, ενημερώνουμε την ταχύτητα και τη θέση κάθε σωματιδίου για το επόμενο χρονικό βήμα σύμφωνα με τις εξισώσεις 1 και 2.

$$v_i(t+1) = \omega \times v_i(t) + \varphi_p \times \rho_1 \times (p_i - x_i(t)) + \varphi_g \times \rho_2 \times \left(p_g - x_i(t)\right) \quad \textbf{(1)}$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad \textbf{(2)}$$

όπου τα ρ_1 και ρ_2 είναι τυχαίοι αριθμοί στο εύρος [0,1]. Αφού ολοκληρωθούν οι παραπάνω υπολογισμοί, προχωράμε στη διαδικασία της εκπαίδευσης για την επόμενη επανάληψη. Μόλις ικανοποιηθεί ένα κριτήριο τερματισμού, όπως η επίτευξη του μέγιστου αριθμού επαναλήψεων, ο αλγόριθμος τερματίζει και το p_g είναι το καλύτερο αποτέλεσμα που εντόπισε το PSO.

6.3.6 Bayesian Optimization

Το Bayesian Optimization (BO) [91] είναι μια στατιστική μέθοδος που χρησιμοποιείται για τη βελτιστοποίηση των black-box συναρτήσεων - συναρτήσεις των οποίων οι εσωτερικές εκφράσεις και μηχανισμοί τους δεν είναι γνωστές. Έτσι, το BO είναι κατάλληλο για προβλήματα βελτιστοποίησης με συγκεκριμένους περιορισμούς που δεν θα μπορούσε να αντιμετωπίσει ένας κοινός αλγόριθμος βελτιστοποίησης. Ένας τέτοιος περιορισμός είναι ότι οι παράγωγοι της συνάρτησης είναι άγνωστοι και κατά συνέπεια ο βελτιστοποιητής δεν έχει στοιχεία για την κατεύθυνση της συνάρτησης. Δεύτερον, η βελτιστοποιήσιμη αντικειμενική συνάρτηση μπορεί να

είναι ακριβή στον υπολογισμό, επομένως θα πρέπει να περιορίσουμε τα δείγματα των εισόδων που η συνάρτηση θα υπολογίσει τελικά.

Το BO χρησιμοποιεί δύο διαφορετικούς τύπους συναρτήσεων για να εντοπίσει το παγκόσμιο βέλτιστο, την υποκατάστατη συνάρτηση και τη συνάρτηση απόκτησης. Μια υποκατάστατη συνάρτηση προσπαθεί να προσεγγίσει την αντικειμενική συνάρτηση με δειγματοληψία εισόδων, η οποία μας δίνει τη δυνατότητα να υποθέσουμε με μεγαλύτερη αυτοπεποίθηση πού είναι πιο πιθανό να είναι τα βέλτιστα σημεία. Μια ευρέως χρησιμοποιούμενη υποκατάστατη συνάρτηση είναι η Gaussian Process (GP), η οποία παρέχει προηγούμενες πεποιθήσεις για τη συνάρτηση και είναι σε θέση να αφομοιώσει προηγούμενες πεποιθήσεις σχετικά με την αντικειμενική συνάρτηση.

Η συνάρτηση απόκτησης είναι υπεύθυνη για την πρόταση των επόμενων σημείων δειγματοληψίας που πρέπει να εξεταστούν, εξισορροπώντας ταυτόχρονα την εξερεύνηση και την εκμετάλλευση. Στο BO, η εξερεύνηση περιγράφει τη δυνατότητα αναζήτησης σημείων τα οποία παρουσιάζουν υψηλή αβεβαιότητα να παρουσιάζουν το βέλτιστο σημείο, ενώ η εκμετάλλευση ορίζει τη δυνατότητα της υποκατάστατης συνάρτησης για την εκτίμηση σημείων που βρίσκονται γύρω από αξιολογημένες περιοχές που έχουν παρουσιάσει υψηλές τιμές στην αντικειμενική συνάρτηση. Μια συνάρτηση απόκτησης που προτείνει εξερεύνηση χωρίς εκμετάλλευση, οδηγεί στην εξερεύνηση ολόκληρου του χώρου αναζήτησης, ακόμη και αν βρεθεί ένα καλό σημείο κοντά στο παγκόσμιο βέλτιστο, ενώ η εκμετάλλευση χωρίς εξερεύνηση προκαλεί την εξερεύνηση μιας περιορισμένης περιοχής του πρώτου βέλτιστου σημείου που βρέθηκε.

Ο αλγόριθμος BO περιγράφεται στα ακόλουθα βήματα. Πρώτον, προετοιμάζουμε μια υποκατάστατη συνάρτηση GP πριν από μια συνάρτηση f, υπολογίζουμε $n0$ σημεία όπου βελτιστοποιείται η τρέχουσα προηγούμενη κατανομή και υπολογίζουμε αυτά τα $n0$ σημεία στην αντικειμενική συνάρτηση. Στη συνέχεια, ενημερώνουμε την προηγούμενη κατανομή χρησιμοποιώντας τα ενημερωμένα διαθέσιμα δεδομένα για να εξαγάγουμε τα μεταγενέστερα δεδομένα, τα οποία θα γίνουν οι προηγούμενες πεποιθήσεις στην επόμενη επανάληψη. Επαναλαμβάνουμε την παραπάνω διαδικασία, έως ότου επιτευχθεί ο μέγιστος αριθμός επαναλήψεων και, τέλος, επεξηγούμε την τελική κατανομή GP προκειμένου να εντοπίσουμε το βέλτιστο σε παγκόσμιο επίπεδο.

6.3.7 Hybrid Bayesian Particle Swarm Hyper-parameter Optimization Model

Το HBPSHPO χρησιμοποίησε τόσο PSO όσο και BO με GP για βελτιστοποίηση υπερ-παραμέτρων. Το πρώτο ήταν υπεύθυνο για τις αριθμητικές υπερ-παραμέτρους, όπως μονάδες και επίπεδα, και το δεύτερο για τις ονομαστικές, όπως συναρτήσεις ενεργοποίησης και βελτιστοποιητές. Ο λόγος που προέκυψε ο διαχωρισμός οφείλεται στην αδυναμία του PSO να διαχειριστεί εύκολα τις ονομαστικές υπερ-παραμέτρους, ενώ το BO μπορεί να χειριστεί τόσο αριθμητικές όσο και ονομαστικές υπερ-παραμέτρους. Ο αλγόριθμος του HBPSHPO δίνεται από τον Αλγόριθμο 6.1.

Στο HBPSHPO, οι αριθμητικές υπερ-παράμετροι καθορίζονται από το PSO και περιλαμβάνουν μονάδες, επίπεδα, ρυθμό απόρριψης, παράμετρο αναδρομής, ρυθμό μάθησης, εποχές, μέγεθος παρτίδας και αριθμό επιπέδων LSTM / CNN. Προκειμένου να υπάρχει ένα πλήρες νευρωνικό δίκτυο βαθιάς, η κατασκευή του δικτύου απαιτεί βελτιστοποιητές και συναρτήσεις ενεργοποίησης για κάθε επίπεδο του δικτύου. Αυτή η εργασία υλοποιείται από το BO που λειτουργεί μέσα στον αλγόριθμο PSO και αναζητά το βέλτιστο ονομαστικό σύνολο υπερπαραμέτρων δεδομένου ενός αριθμητικού συνόλου παραμέτρων από το PSO. Μόλις βρεθεί το βέλτιστο ονομαστικό σύνολο υπερπαραμέτρων, το βαθύ δίκτυο εκπαιδεύεται και επιστρέφει την τιμή απώλειας από τη μέθοδο αξιολόγησης.

Αλγόριθμος 6.1: Hybrid Bayesian & PSO Optimization Algorithm

Βήμα 1: Αρχικοποίηση

Για κάθε σωματίδιο $i = 1, ..., M$ κάνε

i) Αρχικοποίησε τυχαία τη θέση κάθε σωματιδίου $x_i(0)$ ανάμεσα στα κάτω και άνω όρια του χώρου αναζήτησης

ii) Αρχικοποίησε τυχαία την ταχύτητα κάθε σωματιδίου $v_i(0)$ μέσα στο εύρος της ταχύτητας

iii) Αρχικοποίησε την καλύτερη θέση κάθε σωματιδίου με την αρχική του θέση: $p_i(0) = x_i(0)$

iv) Αρχικοποίησε την παγκόσμια καλύτερη θέση του σμήνους με την ελάχιστη αρχική θέση ανάμεσα σε όλα τα σωματίδια: $p_g(0) = argmin(x_i(0))$

Βήμα 2: Επανάλαβε μέχρι να ικανοποιηθεί κάποιο κριτήριο τερματισμού

Βήμα 2a: Ανανέωσε της μεταβλητές του PSO

Για κάθε σωματίδιο $i = 1, ..., M$ κάνε

i) Ανανέωσε την ταχύτητα κάθε σωματιδίου σύμφωνα με τον τύπο:
$v_i(t+1) = \omega \times v_i(t) + \rho_1 \times (p_i - x_i(t)) + \rho_2 \times \left(p_g - x_i(t)\right)$

ii) Ανανέωσε τη θέση κάθε σωματίδιου σύμφωνα με τον τύπο:
$x_i(t+1) = x_i(t) + v_i(t+1)$

Βήμα 2b: Bayesian Optimization με GP

i) Εφάρμοσε μια Gaussian Process prior στην f

ii) Παρατήρησε την f σε $n0$ σημεία σύμφωνα με ένα αρχικό πειραματικό σχεδιασμό

iii) Αρχικοποίησε $n = n0$

iv) Επανάλαβε όσο $n \leq N$

a) Ενημερωσε την μεταγενέστερη πιθανοτική κατανομή στην f χρησιμοποιώντας όλα τα διαθέσιμα δεδομένα

b) Έστω x_n είναι το μέγιστο στη συνάρτηση απόκτησης ανάμεσα στα x.

c) Παρατήρησε $y_n = f(x_n, x_i(t+1), v_i(t+1))$.

d) $n \leftarrow n + 1$

Βήμα 2c: Ανανέωσε τις καλύτερες θέσεις του PSO

i)Εάν $y_n < p_i(t)$, τότε

a) Ανανέωσε την καλύτερη θέση του i_{th} σωματιδίου : $p_i = x_i(t)$

b) Εάν $y_n < p_g(t)$, τότε ανανέωσε την καλύτερη θέση του σμήνους: $p_g = x_i(t)$

ii) $t \leftarrow (t+1)$

Βήμα 3: Αποτέλεσμα το p_g που είναι η καλύτερη λύση που βρέθηκε

6.4 Πειραματικά Αποτελέσματα και Συζήτηση

6.4.1 Πειραματική ρύθμιση

Το CNN παλινδρόμησης πολλαπλών εξόδων με το μοντέλο HBPSHPO υλοποιείται με Python 3 χρησιμοποιώντας τα πλαίσια NumPy, pandas, statistics, Scikit-learn, TensorFlow 2, SciPy, PySwarms και Scikit-Optimize. Το περιβάλλον που χρησιμοποιήσαμε είναι το σημειωματάριο Jupyter του Google Colaboratory. Ο πηγαίος κώδικας του πειράματος είναι διαθέσιμος για κάθε είδους αναπαραγωγή και επανεξέταση στο δεύτερο συγγραφέα Αποθήκη GitHub [89].

Το σύνολο δεδομένων κατασκευάστηκε από ένα εργαλείο παρακολούθησης που υλοποιήθηκε σε Python 3 που χρησιμοποιεί τις βιβλιοθήκες psutil [92] και GPUtil [93]. Παρακολουθήσαμε τη χρήση CPU, RAM, Δίσκου I / O και εύρους ζώνης σε πραγματικό χρόνο με μεσοδιάστημα ενός

δευτερολέπτου. Οι κόμβοι ακμών ήταν το Raspberry Pi3 τετραπύρηνο 64-bit ARM Cortex-A53 στα 1.4GHz με λειτουργικό σύστημα Raspbian που είναι μια έκδοση του Debian Linux. Η εν λόγω εφαρμογή ήταν μια ταξινόμηση κειμένου επεξεργασίας φυσικής γλώσσας. Αποφασίσαμε να κάνουμε την ταξινόμηση κειμένου σε ένα προηγμένο περιβάλλον υπολογιστών, τοπικά, κοντά στους κατόχους κειμένου και όχι σε υποδομές υπολογιστικού νέφους για ζητήματα απορρήτου. Ο λόγος για αυτήν την επιλογή είναι ότι οι κάτοχοι του κειμένου δεν συμφώνησαν τα κείμενά τους να μεταφερθούν και να υποβληθούν σε επεξεργασία σε απομακρυσμένους διακομιστές. Προκειμένου να ελέγξουμε την εφαρμογή από απόσταση και να πάρουμε τα σύνολα δεδομένων χρήσης πόρων χρησιμοποιήσαμε το πρωτόκολλο SSH, αλλά δεν είχαμε τα δικαιώματα πρόσβασης στα επεξεργασμένα κείμενα.

Το προτεινόμενο μοντέλο συγκρίνεται με τέσσερις τελευταίας τεχνολογίας προβλέψεις, AUCROP [94], XGBoost [96], Auto-sklearn [95] και LSTM-RNN. Το AUCROP είναι η συντομογραφία του Application and User Context Resource Predictor που είναι ένας προγνωστικός παράγοντας μετα-μοντέλων κατάλληλος για χρήση πόρων που χρησιμοποιεί κοινούς αλγόριθμους μηχανικής μάθησης. Το Auto-sklearn είναι ένα αυτοματοποιημένο μετα-μοντέλο μηχανικής εκμάθησης γενικής χρήσης που χρησιμοποιείται για την προ-επεξεργασία δεδομένων, την παλινδρόμηση και συντονισμό υπερ-παραμέτρων μέσω του αλγορίθμου Bayesian Optimization. Η αποδοτικότητά του οφείλεται στη δυνατότητα αποθήκευσης προηγούμενων εργασιών βελτιστοποίησης που παρέχει την ευκαιρία για την εκπαίδευση δεδομένων με προηγούμενες αποθηκευμένες ρυθμίσεις. Το Auto-sklearn έχει κερδίσει τη διάσημη πρόκληση ChaLearn AutoML, είναι δημοφιλές σε ερευνητικά έγγραφα και θεωρείται ως ένα από τα καλύτερα πλαίσια AutoML από την κοινότητα επιστημόνων δεδομένων.

Το XGBoost είναι μια βιβλιοθήκη λογισμικού ανοιχτού κώδικα που παρέχει ένα πλαίσιο ενίσχυσης της κλίσης και χρησιμοποιείται συχνά για τα δέντρα αποφάσεων που ενισχύονται με κλίση. Η διαθεσιμότητά του σε πολλές γλώσσες προγραμματισμού και η ενσωμάτωσή του στις βιβλιοθήκες της Python, όπως το Scikit-learn, συνέβαλαν στη δημοτικότητά του. Το XGBoost έχει χρησιμοποιηθεί πολύ στις λύσεις Kaggle και KDD Cup. Το μοντέλο LSTM-GA είναι το προηγούμενό μας μοντέλο πρόβλεψης χρήσης πόρων σε συσκευές ακμής που αξιοποιεί γενετικούς αλγόριθμους για την εκτίμηση των υπερ-παραμέτρων το οποίο εκμεταλλεύεται και τροφοδοτεί RNN και συγκεκριμένα επίπεδα LSTM.

Το προτεινόμενο μοντέλο χρησιμοποίησε την τεχνική αξιολόγησης εκτός δείγματος (out-of-sample) που διατηρεί την ακολουθία των παρατηρήσεων. Επομένως, δεν εφαρμόσαμε καμία μέθοδο ανακατεύθυνσης στο πλαίσιο εφαρμογών χρονοσειρών. Διαχωρίσαμε το σύνολο δεδομένων σε δύο μέρη, την ακολουθία εκπαίδευσης που ήταν το 66% των παρατηρήσεων και την ακολουθία δοκιμών που ήταν το υπόλοιπο 34% των δεδομένων. Το προτεινόμενο μοντέλο αξιολογήθηκε με τις μετρικές αξιολόγησης Mean Absolute Error (MAE) και Root Mean Squared Error (RMSE). Καταγράψαμε τον χρόνο εκπαίδευσης και τους χρόνους αποτελεσμάτων για μία μόνο πρόβλεψη και για μια παρτίδα 100 προβλέψεων.

Το HBPSHPO που περιλαμβάνει τους αλγόριθμους PSO και BO πραγματοποίησε μια έξυπνη αναζήτηση στο χώρο αναζήτησης προκειμένου να προσδιορίσει το βέλτιστο μοντέλο CNN παλινδρόμησης πολλαπλών εξόδων. Στα πειράματά μας με το HBPSHPO, συμπεριλάβαμε επίσης υπερ-παραμέτρους για LSTM-RNN και στρώματα εμπρόσθιας τροφοδοτησης. Τα πειραματικά αποτελέσματα του προτεινόμενου μοντέλου μας, AUCROP, XGBoost, Auto-sklearn και Genetic Algorithm with LSTM-RNN συνοψίζονται στον Πίνακα 6.1.

Method	RMSE	MAE	CPU-1(%)		RAM-1(%)		Train. Time	Inference Time	
			RMSE	MAE	RMSE	MAE		Single	Batch
PSO-CNN	0.0631	0.0254	15.932	12.898	1.416	0.587	1692	0.036	0.039
PSO-LSTM	0.0661	0.0276	16.088	12.691	1.479	0.613	1042	0.034	0.035
AUCROP	0.0814	0.0414	17.235	14.009	2.480	1.482	522	0.004	0.011
XGBoost	0.1139	0.0599	16.457	13.569	1.515	0.472	181	0.060	0.010
Auto-sklearn	0.1055	0.0243	52.659	17.856	1.546	0.526	1338	0.263	0.572
GA-LSTM	0.0674	0.0338	16.099	12.838	1.746	0.917	574	0.020	0.024

Πίνακας 6.1 Single-Output & Multi-Output Αξιολόγηση

6.4.2 Αποτελέσματα και Συζήτηση

Η πειραματική μας μελέτη χωρίζεται σε τέσσερα κύρια μέρη. Αρχικά, εξετάζουμε την ακρίβεια των προβλέψεων μοντέλων σε όρους μέσου RMSE και MAE για όλους τους ελεγχόμενους πόρους. Στη συνέχεια, παρέχουμε τη χειρότερη προβλεπόμενη συσκευή, από την υποδομή ακμής, την ακρίβεια των προβλέψεων της CPU και της μνήμης. Στη συνέχεια, εξετάζουμε τον χρόνο εκπαίδευσης, τον χρόνο συμπερασμάτων για μία μόνο πρόβλεψη και το χρόνο συμπερασμάτων για μια παρτίδα 100 προβλέψεων. Τέλος, εξετάζουμε τη σύγκλιση της μεθόδου HBPSHPO.

Στον πίνακα 6.1 μπορούμε να δούμε ότι τα μοντέλα βαθιάς μάθησης, π.χ. PSO-CNN, PSO-LSTM και GA-LSTM έχουν καλύτερη ακρίβεια από τα άλλα μετα-μοντέλα μηχανικής μάθησης σε μέσες προβλέψεις χρήσης πόρων, εκτός από την περίπτωση της χρήσης ενός πόρου RAM, όπου το XGBoost επιτυγχάνει καλύτερη απόδοση MAE. Το PSO-CNN αποδίδει καλά στο μέσο MAE με τιμή 0,0255. Το RMSE που είναι μια μέτρηση που τιμωρεί μεγάλα σφάλματα είναι μεγαλύτερο από το MAE που δηλώνει ότι λίγες συγκεκριμένες προβλέψεις έχουν μεγαλύτερο σφάλμα πρόβλεψης. Η CPU και RAM MAE της συσκευής με το μεγαλύτερο σφάλμα είναι 12,691% και 0,613% αντίστοιχα. Εξάγουμε δύο συμπεράσματα από αυτές τις τιμές. Πρώτον, μπορούμε να δούμε τα όρια πρόβλεψης του μοντέλου χρησιμοποίησης πόρων και δεύτερον να κατανοήσουμε ότι η πρόβλεψη της χρήσης RAM είναι πολύ πιο ακριβής από την CPU. Για να αιτιολογήσουμε το δεύτερο συμπέρασμα, εξετάσαμε το σύνολο δεδομένων χρήσης πόρων και διαπιστώσαμε ότι η χρήση της CPU είχε πιο έντονες διακυμάνσεις από τη μνήμη RAM.

Όσον αφορά τον χρόνο εκπαίδευσης στον πίνακα 6.1 μπορούμε να δούμε ότι δεν είναι το ελάχιστο για τα μοντέλα βαθιάς μάθησης. Οι λόγοι είναι διττοί. Πρώτον, η εκπαίδευση των μοντέλων DL είναι μια πολύ πιο υπολογιστική βαριά διαδικασία λόγω της οπίσθιας διάδοσης και του μεγάλου αριθμού παραμέτρων από την εκπαίδευση των παραδοσιακών μοντέλων μηχανικής μάθησης. Δεύτερον, οι υπερ-παράμετροι των μοντέλων βαθιάς μάθησης είναι πολύ περισσότερες από τις υπερ-παραμέτρους των μοντέλων κλασικής μηχανικής μάθησης. Ωστόσο, μπορούμε να δούμε ότι το μοντέλο HBPSHPO έχει συγκρίσιμους χρόνους εκπαίδευσης με τα άλλα μοντέλα λόγω της αποτελεσματικής έξυπνης αναζήτησης στο χώρο αναζήτησης. Ο ενιαίος και ομαδικός χρόνος συμπερασμάτων σε όλες τις περιπτώσεις είναι πολύ μικρότερος από 600 χιλιοστά του δευτερολέπτου και δεν το θεωρούμε αξιοσημείωτο να το βελτιώσουμε περαιτέρω.

Η μέθοδος HBPSHPO έκανε μια έξυπνη αναζήτηση στο RNN-LSTM, το CNN και τις απλές τοπολογίες νευρωνικού δικτύου εμπρόσθιας τροφοδότησης και κατέληξε στο συμπέρασμα ότι ένα νευρωνικό δίκτυο που ξεκινά με στρώματα CNN και max-pooling, ακολουθείται από ένα flatten επίπεδο, ένα feedforward, ένα dropout και ένα τελευταίο πυκνό στρώμα έχει την καλύτερη ακρίβεια.

Το HBPSHPO επέλεξε τον βελτιστοποιητή Adam. Το γεγονός ότι το ADAM έχει επίσης προταθεί ως ο καλύτερος βελτιστοποιητής σε πολλά μοντέλα CNN [88] επιβεβαιώνει τη δυνατότητα εφαρμογής του HBPSHPO για βέλτιστη επιλογή υπερ-παραμέτρων. Όσον αφορά τη συνολική σύγκριση των μεθόδων βελτιστοποίησης υπερ-παραμέτρων, έχουμε δει ότι η μέθοδος HBPSHPO ξεπερνά τον γενετικό αλγόριθμο. Συγκεκριμένα, εκτελούμε πολλά πειράματα με το ίδιο σύνολο υπερ-παραμέτρων μεταξύ HBPSHPO και GA και το HBPSHPO είχε πάντα καλύτερα αποτελέσματα.

Value	**Units**	**Layers**	**Lookback**	**Dropout**	**Learn. rate**	**Epochs**	**Batch size**	**Num CNN or LSTM**
Min	1	1	1	0	0.001	20	32	1
Max	128	5	5	0.5	0.2	200	1024	2

Πίνακας 6.2: Αριθμητικές υπερπαράμετροι του HBPSHPO

Optimizers	RMSprop	Adam	SGD	Adagrad	Adadelta	Adamax	Nadam
Activation Function	tanh	linear		sigmoid		relu	

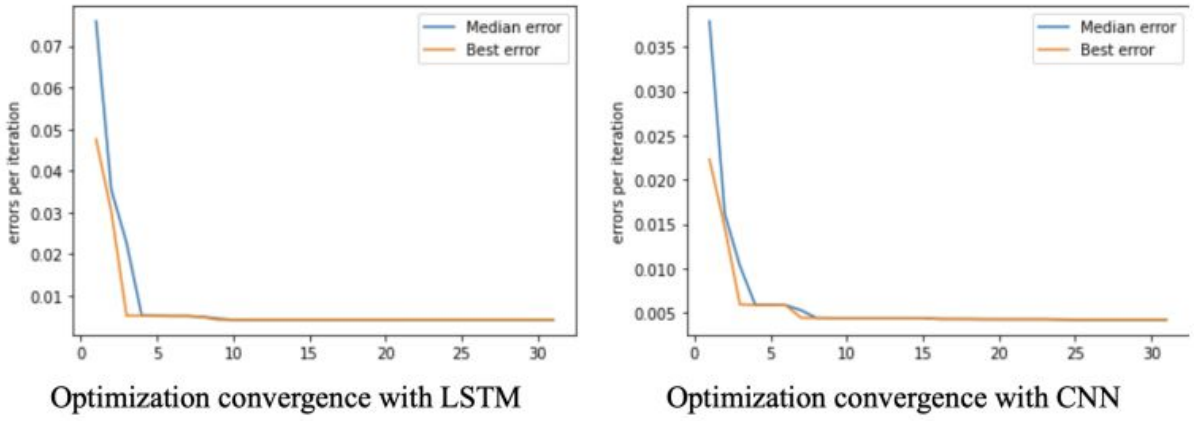
Πίνακας 6.3: Ονομαστικές υπερπαράμετροι του HBPSHPO

Ο χώρος υπόθεσης στον οποίο κινούνται τα σωματίδια του PSO παρέχεται στον Πίνακα 6.2. Το εύρος κάθε υπερ-παραμέτρου παρέχεται από τις σειρές Min και Max. Κάναμε μερικά πειράματα με διαφορετικές διαμορφώσεις του PSO και του BO και καταλήξαμε να ορίσουμε το μέγεθος του σμήνους του PSO σε 15 και την ελάχιστη αλλαγή της καλύτερης αντικειμενικής τιμής του σμήνους πριν από την ολοκλήρωση της αναζήτησης σε 0,0001. Οι υποψήφιες ονομαστικές τιμές του τμήματος BO των υπερ-παραμέτρων HBPSHPO παρέχονται στον πίνακα 6.3 και ορίζουμε την αναμενόμενη βελτίωση ως συνάρτηση απόκτησης.

Μια άλλη επιθυμητή ιδιότητα ενός αλγορίθμου βελτιστοποίησης είναι η ικανότητα σύγκλισης. Το PSO είναι μια μετα-ευρετική προσέγγιση που βασίζεται στη συστηματική πρόοδο της αναζήτησης και της αξιολόγησης των υποψηφίων λύσεων [97]. Η Bayesian βελτιστοποίηση με βάση τη διαδικασία Gaussian συγκλίνει αποδεδειγμένα σε παγκόσμιο βέλτιστο, υποθέτοντας ότι ο πυρήνας είναι γνωστός εκ των προτέρων. Αυτό όμως δεν συμβαίνει στις περισσότερες περιπτώσεις [98] και συγκεκριμένα στις ονομαστικές υπερ-παραμέτρους των νευρωνικών δικτύων. Κατά συνέπεια, η σύγκλιση HBPSHPO δεν έχει μαθηματική απόδειξη. Ωστόσο, η σύγκλιση μπορεί να μετρηθεί πειραματικά με βάση το καλύτερο και διάμεσο σφάλμα των υποψηφίων λύσεων για κάθε γενιά. Το καλύτερο σφάλμα εκφράζει την καλύτερη τιμή απώλειας των υποψήφιων νευρωνικών δικτύων (σωματίδια) σε μία γενιά του σμήνους. Το διάμεσο σφάλμα εκφράζει τη μέση τιμή απώλειας όλων των υποψήφιων νευρωνικών δικτύων (σωματίδια) σε μία γενιά του σμήνους. Σε όλα τα πειράματα, οι τοπολογίες του νευρωνικού δικτύου συγκλίνουν γρήγορα και δεν είχαμε σημαντική βελτίωση στα ποσοστά σφάλματος μετά τις πρώτες δέκα γενιές.

Στο σχήμα 6.4 μπορούμε να δούμε δύο περιπτώσεις όπου στο αριστερό γράφημα δοκιμάσαμε το HBPSHPO με μόνο στρώματα LSTM και εμπρόσθιας τροφοδότησης και στο δεξιό γράφημα δοκιμάσαμε στρώματα CNN, και εμπρόσθιας τροφοδότησης. Το μοντέλο HBPSHPO έχει σχεδιαστεί, ώστε να επιβιώνει πάντα τα καλύτερα σωματίδια από γενιά σε γενιά και να έχει έντονη επίδραση στα άλλα σωματίδια του σμήνους. Έτσι, το καλύτερο σφάλμα θα βελτιώνεται πάντα ή θα παραμένει σταθερό. Όσον αφορά το διάμεσο σφάλμα, βλέπουμε ότι τα καλύτερα σωματίδια επηρεάζουν τα υπόλοιπα σωματίδια του σμήνους και όλα μαζί κινούνται προς πιο ακριβείς τοπολογίες ANN. Και

στις δύο περιπτώσεις βλέπουμε ότι έχουμε μια πολύ γρήγορη σύγκλιση και μετά από τέσσερις έως εννέα γενιές μπορούμε να βρούμε μια σχεδόν βέλτιστη τοπολογία νευρωνικού δικτύου.



Εικόνα 6.4: Σύγκλιση του HBPSHPO Model

6.5. Συμπέρασμα

Σε αυτό το κεφάλαιο, συζητήσαμε πώς η πρόβλεψη της χρήσης πόρων μπορεί να παρέχει χρήσιμες πληροφορίες για πολλές λειτουργίες ενορχήστρωσης υπολογιστών ακμής, όπως εκφόρτωση εργασιών, εξισορρόπηση φόρτου εργασίας, προληπτική αυτόματη κλιμάκωση και ανοχή σφαλμάτων. Η χρήση πόρων είναι δύσκολο να προβλεφθεί εξαιτίας της δυναμικότητας και της ετερογένειας των εργασιών και των κόμβων ακμής. Για την αντιμετώπιση αυτών των προκλήσεων, προτείναμε ένα μοντέλο βαθιάς μάθησης με επίπεδα CNN που αξιοποιεί την ακολουθία των παρατηρήσεων των δεδομένων και μια μέθοδο βελτιστοποίησης υπερ-παραμέτρων που συνδυάζει BO και PSO προκειμένου να πραγματοποιήσει μια έξυπνη αναζήτηση στο χώρο των ονομαστικών και αριθμητικών υπερ-παραμέτρων των μοντέλων βαθιάς μάθησης. Η πειραματική αξιολόγηση έδειξε ότι το προτεινόμενο μοντέλο μας παρέχει πολύ καλές προβλέψεις και ξεπερνά άλλα μοντέλα μηχανικής μάθησης και εκτιμητές χρήσης πόρων.

Οι μελλοντικές κατευθύνσεις αυτής της εργασίας είναι να εφαρμόσουν έναν μηχανισμό εκφόρτωσης εργασιών που χρησιμοποιεί τις προβλέψεις του μοντέλου CNN-HBPSHPO προκειμένου να υπολογίσει ποιοι κόμβοι ακμής επεξεργασίας θα έχουν επαρκείς πόρους διαθέσιμους τις επόμενες χρονικές περιόδους και θα ικανοποιήσουν την ποιότητα των απαιτήσεων υπηρεσίας. Σκοπεύουμε να αξιολογήσουμε την απόδοση του μοντέλου εκφόρτωσης εργασιών με προσομοιωτή άκρων όπως EdgeCloudSim, FogNetSim ++ ή iFogSim και μετά να κάνουμε πειράματα σε πραγματικές υποδομές.

Συντομογραφίες

AF	Activation Function	Συνάρτηση Ενεργοποίησης
AI	Artificial Intelligence	Τεχνητή Νοημοσύνη
ANN	Artificial Neural Network	Τεχνητό Νευρωνικό Δίκτυο
BPTT	Backpropagation Through Time	Οπισθοδιάδοση
CI	Computational Intelligence	Υπολογιστική Νοημοσύνη
DL	Deep Learning	Βαθιά Μάθηση
DOP	Dynamic Optimization Problem	Πρόβλημα Δυναμικής Βελτιστοποίησης
EC	Evolutionary Computation	Εξελικτικός Υπολογισμός
EI	Expected Improvement	Αναμενόμενη Βελτίωση
GP	Gaussian Process	Γκαουσιανή Διαδικασία
HBPSHPO	Hybrid Bayesian Particle Swarm Hyper-parameter Optimization	Υβριδική Μπεϋζιανή Βελτιστοποίηση Σμήνους Σωματιδίων
IoT	Internet of Things	Διαδίκτυο των πραγμάτων
LCB	Lower Confidence Bound	Κάτω Όριο Διαστήματος Εμπιστοσύνης
LSTM	Long Short-Term Memory	
MAE	Mean Absolute Error	Μέσο Απόλυτο Σφάλμα
ML	Machine Learning	Μηχανική Μάθηση
MLP	Multi-Layer Perceptron	Πολυστρωματικό νευρωνικό δίκτυο
MSE	Mean Squared Error	Μέσο Τετραγωνικό Σφάλμα
PI	Probability of Improvement	Πιθανότητα Βελτίωσης
PSO	Particle Swarm Optimization	Βελτιστοποίηση Σμήνους Σωματιδίων
RMSE	Root Mean Squared Error	Ρίζα του Μέσου Τετραγωνικού Σφάλματος
RNN	Recurrent Neural Network	Αναδρομικά Νευρωνικά Δίκτυα
SGD	Stochastic Gradient Descent	Στοχαστική Κατάβαση Κλίσης

SI	Swarm Intelligence	Νοημοσύνη Σμήνους
SVM	Support Vector Machines	Μηχανές Διανυσμάτων Υποστήριξης

Βιβλιογραφικές αναφορές

[1] D. A. Zahner and E. Micheli-Tzanakou, “Networks: Definitions, Methods, Applications,” p. 18, 2000.

[2] M. A. Nielsen, “Neural Networks and Deep Learning,” p. 224, 2015.

[3] Y. Bengio, I. Goodfellow and A. Courville, “Deep Learning,” p.705, MIT Press 2015

[4] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, “Activation Functions: Comparison of trends in Practice and Research for Deep Learning,” *arXiv:1811.03378 [cs]*, p. 20, Nov. 2018.

[5] F. Piekniewski and L. Rybicki, Visual comparison of performance for different activation functions in MLP networks, vol. 4. 2004, p. 2952 vol.4.

[6] “Activation function,” Wikipedia. Sep. 02, 2020, Accessed: Oct. 16, 2020. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Activation_function&oldid=976358403.

[7] C. Özkan and F. S. Erbek, “The Comparison of Activation Functions for Multispectral Landsat TM Image Classification,” *Photogrammetric Engineering & Remote Sensing*, vol. 69, no. 11, pp. 1225–1234, Nov. 2003, doi: 10.14358/PERS.69.11.1225.

[8] D. Radečić, “Softmax Activation Function Explained,” *Medium*, Jun. 18, 2020. https://towardsdatascience.com/softmax-activation-function-explained-a7e1bc3ad60 (accessed Oct. 16, 2020).

[9] X. Glorot, A. Bordes, and Y. Bengio, “Deep Sparse Rectifier Neural Networks,” p. 9.

[10] G. Lin and W. Shen, “Research on convolutional neural network based on improved Relu piecewise activation function,” *Procedia Computer Science*, vol. 131, pp. 977–984, Jan. 2018, doi: 10.1016/j.procs.2018.04.239.

[11] K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown, “Text Classification Algorithms: A Survey,” *Information*, vol. 10, no. 4, Art. no. 4, Apr. 2019, doi: 10.3390/info10040150.

[12] Tieleman, T.; Hinton, G. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. COURSERA Neural Netw. Mach. Learn. 2012, 4, 26–31.

[13] Kingma, D.; Ba, J. Adam: A method for stochastic optimization. arXiv 2014, arXiv:1412.6980.

[14] Duchi, J.; Hazan, E.; Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. J. Mach. Learn. Res. 2011, 12, 2121–2159.

[15] Zeiler, M.D. ADADELTA: An adaptive learning rate method. arXiv 2012, arXiv:1212.5701.

[16] I. Kandel, M. Castelli, and A. Popovič, “Comparative Study of First Order Optimizers for Image Classification Using Convolutional Neural Networks on Histopathology Images,” *Journal of Imaging*, vol. 6, no. 9, Art. no. 9, Sep. 2020, doi: 10.3390/jimaging6090092.

[17] S. Raschka, “Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning,” *arXiv:1811.12808 [cs, stat]*, Dec. 2018, Accessed: Oct. 18, 2020. [Online]. Available: http://arxiv.org/abs/1811.12808.

[18] randerson112358, “Machine Learning Simplified,” *Medium*, May 30, 2020. https://medium.com/@randerson112358/machine-learning-simplified-407caa414386 (accessed Oct. 18, 2020).

[19] H. Allamy, “METHODS TO AVOID OVER-FITTING AND UNDER-FITTING IN SUPERVISED MACHINE LEARNING (COMPARATIVE STUDY),” Dec. 2014.

[20] F. Chollet, *Deep learning with Python*. Shelter Island, New York: Manning Publications Co, 2018.

[21] M. V. Shcherbakov *et al.*, *DOI: 10.5829/idosi.wasj.2013.24.itmies.80032 A Survey of Forecast Error Measures*.

[22] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting,” p. 30.

[23] A. Parmezan, V. Alves de Souza, and G. Batista, “Evaluation of statistical and machine learning models for time series prediction: Identifying the state-of-the-art and the best conditions for the use of each model,” *Information Sciences*, Jan. 2019, doi: 10.1016/j.ins.2019.01.076.

[24] Simon Haykin, Neural Networks: A Comprehensive Foundation, Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition, 1994.

[25] Nikhil Ketkar, Stochastic Gradient Descent, pp. 113–132, Apress, Berkeley, CA, 2017.

[26] P. J. Werbos, “Backpropagation through time: what it does and how to do it”, Proceedings of the IEEE, vol. 78, no. 10, pp. 1550–1560, Oct 1990.

[27] RONALD J. WILLIAMS and DAVID ZIPSER, “Experimental Analysis of the Real-time Recurrent Learning Algorithm”, Connection Science, vol. 1, no. 1, pp. 87–111, 1989.

[28] S. J. Russell and P. Norvig, Artificial Intelligence: A Modern Approach, Prentice Hall, New York, NY, USA, 2010.

[29] B. L. Agarwal, Basic Statistics, New Age International, 2006.

[30] K. E. Voges and N. Pope, Business Applications and Computational Intelligence, Idea Group Publishing, 2006.

[31] V. Kothari, J. Anuradha, S. Shah, and P. Mittal, “A survey on particle swarm optimization in feature selection,” in Global Trends in Information Systems and Software Applications: Proceedings of the 4th International Conference, ObCom 2011, Vellore, TN, India, December 9–11, 2011, Part 2, P. V. Krishna, M. R. Babu, and E. Ariwa, Eds., vol. 270, pp. 192–201, Springer, Berlin, Germany, 2012.

[32] Nakisa, “A SURVEY: PARTICLE SWARM OPTIMIZATION BASED ALGORITHMS TO SOLVE PREMATURE CONVERGENCE PROBLEM,” *Journal of Computer Science*, vol. 10, no. 9, pp. 1758–1765, Sep. 2014, doi: 10.3844/jcssp.2014.1758.1765.

[33] Y. Zhang, S. Wang, and G. Ji, “A Comprehensive Survey on Particle Swarm Optimization Algorithm and Its Applications,” *Mathematical Problems in Engineering*, vol. 2015, pp. 1–38, 2015, doi: 10.1155/2015/931256.

[34] F. Shahzad, S. Masood, and N. K. Khan, “Probabilistic opposition-based particle swarm optimization with velocity clamping,” Knowledge and Information Systems, vol. 39, no. 3, pp. 703– 737, 2014

[35] M. Clerc and J. Kennedy, “The particle swarm-explosion, stability, and convergence in a multidimensional complex space,” IEEE Transactions on Evolutionary Computation, vol. 6, no. 1, pp. 58–73, 2002.

[36] Y. Zhang, S. Wang, and Z. Dong, “Classification of alzheimer disease based on structural magnetic resonance imaging by kernel support vector machine decision tree,” Progress in Electromagnetics Research, vol. 144, pp. 171–184, 2014.

[37] E.-G. Talbi, *Metaheuristics: from design to implementation*. Hoboken, N.J: John Wiley & Sons, 2009.

[38] A. P. Engelbrecht. Computational Intelligence: An Introduction. Wiley, 2002

[39] Y. Shi and R. Eberhart. A modified particle swarm optimizer. In IEEE International Conference on Evolutionary Computation. IEEE Press, Piscataway, NJ, 1998, pp. 69–73

[40] J. Snoek *et al.*, “Scalable Bayesian Optimization Using Deep Neural Networks,” p. 10.

[41] Mockus, J., Tiesis, V., and Zilinskas, A. The application of Bayesian methods for seeking the extremum. Towards Global Optimization, 2, 1978.

[42] Lizotte, D. Practical Bayesian Optimization. PhD thesis, University of Alberta, Edmonton, Alberta, 2008.

[43] Osborne, M. A., Garnett, R., and Roberts, S. J. Gaussian processes for global optimization. In Learning and Intelligent Optimization, 2009

[44] Brochu, E., Brochu, T., and de Freitas, N. A Bayesian interactive optimization approach to procedural animation design. In ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2010.

[45] Jones, D. R. A taxonomy of global optimization methods based on response surfaces. Journal of Global Optimization, 21, 2001.

[46] Osborne, M. A., Garnett, R., and Roberts, S. J. Gaussian processes for global optimization. In Learning and Intelligent Optimization, 2009.

[47] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, “Taking the Human Out of the Loop: A Review of Bayesian Optimization,” Proceedings of the IEEE, vol. 104, no. 1, pp. 148–175, Jan. 2016, doi: 10.1109/JPROC.2015.2494218.

[48] E. Brochu, V. M. Cora, and N. de Freitas, ‘‘A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning,’’ Dept. Comput. Sci., Univ. British Columbia, Vancouver, BC, Canada, Tech. Rep. UBC TR-2009-23, 2009

[49] J. Mockus. The Bayesian approach to global optimization. In R. Drenick and F. Kozin, editors, System Modeling and Optimization, volume 38, pages 473–481. Springer Berlin / Heidelberg, 1982.

[50] J. Mockus. Application of Bayesian approach to numerical methods of global and stochastic optimization. J. Global Optimization, 4(4):347 – 365, 1994.

[51] H. J. Kushner. A new method for locating the maximum point of an arbitrary multipeak curve in the presence of noise. Journal of Basic Engineering, 86, 1964.

[52] J. Snoek, H. Larochelle, and R. P. Adams, “Practical Bayesian Optimization of Machine Learning Algorithms,” in *Advances in Neural Information Processing Systems 25*, F. Pereira,

C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 2951–2959.

[53] M. Mavrovouniotis and S. Yang, "Ant colony optimization with immigrants schemes for the dynamic travelling salesman problem with traffic factors," *Applied Soft Computing*, vol. 13, no. 10, pp. 4023–4037, Oct. 2013, doi: 10.1016/j.asoc.2013.05.022.

[54] P. Bosman, Learning, anticipation and time-deception in evolutionary online dynamic optimization, in: Proceedings of the 2005 Genetic and Evolutionary Computation Conference, ACM Press, New York, NY, 2005, pp. 39–47.

[55] T. Nguyen, X. Yao, Dynamic time-linkage problems revisited, in: M. Giacobini, A.Brabazon, S. Cagnoni, G. Di Caro, A. Ekárt, A. Esparcia-Alcázar, M. Farooq, A.Fink, P. Machado (Eds.), Applications of Evolutionary Computing, Lecture Notesin Computer Science, vol. 5484, Springer Berlin Heidelberg, 2009, pp. 735–744.

[56] M. Helbig, A. Engelbrecht, Dynamic multi-objective optimization using PSO, in:E. Alba, A. Nakib, P. Siarry (Eds.), Metaheuristics for Dynamic Optimization, Studies in Computational Intelligence, vol. 433, Springer Berlin Heidelberg, 2013, pp. 147–188.

[57] M. Farina, K. Deb, P. Amato, Dynamic multiobjective optimization problems: test cases, approximations, and applications, IEEE Trans. Evol. Comput. 8 (5) (2004) 425–442.

[58] T. Nguyen, X. Yao, Continuous dynamic constrained optimization-the challenges, IEEE Trans. Evol. Comput. 16 (6) (2012) 769–786.

[59] C. Cruz, J.R. Gonzalez, D.A. Pelta, Optimization in dynamic environments: a survey on problems, methods and measures, Soft Comput. 15 (7) (2011) 1427–1448.

[60] S. Yang, Y. Jiang, T. Nguyen, Metaheuristics for dynamic combinatorial optimization problems, IMA J. Manag. Math. 24 (4) (2013) 451–480.

[61] M. Mavrovouniotis, C. Li, and S. Yang, "A survey of swarm intelligence for dynamic optimization: Algorithms and applications," *Swarm and Evolutionary Computation*, vol. 33, pp. 1–17, Apr. 2017, doi: 10.1016/j.swevo.2016.12.005.

[62] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for VM-based cloudlets in mobile computing," IEEE Pervasive Comput., vol. 8, no. 4, pp. 14–23, Oct./Dec. 2009.

[63] (2016). OpenFog Architecture Overview. OpenFog Consortium Architecture Working Group. Accessed on Dec. 7, 2016. [Online]. Available: http://www.openfogconsortium.org/wp-content/ uploads/OpenFog-Architecture-Overview-WP-2-2016.pdf

[64] S. Yi, Z. Hao, Z. Qin, and Q. Li, "Fog computing: Platform and applications," in Proc. 3rd IEEE Workshop Hot Topics Web Syst. Technol. (HotWeb), Washington, DC, USA, 2015, pp. 73–78.

[65] K. Ha et al., "Towards wearable cognitive assistance," in Proc. 12th Annu. Int. Conf. Mobile Syst. Appl. Services, Bretton Woods, NH, USA, 2014, pp. 68–81

[66] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "CloneCloud: Elastic execution between mobile device and cloud," in Proc. 6th Conf. Comput. Syst., Salzburg, Austria, 2011, pp. 301–314

[67] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, “Edge Computing: Vision and Challenges,” *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016, doi: 10.1109/JIOT.2016.2579198.

[68] M. Satyanarayanan, “The Emergence of Edge Computing,” *Computer*, vol. 50, no. 1, pp. 30–39, Jan. 2017, doi: 10.1109/MC.2017.9.

[69] Graves A., Supervised Sequence Labelling with Recurrent Neural Networks, SpringerVerlag Berlin Heidelberg, 2012.

[70] “CS231n Convolutional Neural Networks for Visual Recognition.” https://cs231n.github.io/convolutional-networks/ (accessed Oct. 28, 2020).

[71] Chunhong Liu, Chuanchang Liu, Yanlei Shang, Shiping Chen, Bo Cheng, Junliang Chen, ”An adaptive prediction approach based on workload pattern discrimination in the cloud”, Journal of Network and Computer Applications, vol. 80, 2017, pp. 35-44, ISSN 1084-8045, https://doi.org/10.1016/j.jnca.2016.12.017.

[72] Maria Carla Calzarossa, Luisa Massari, and Daniele Tessera. 2016. Workload Characterization: A Survey Revisited. ¡i¿ACM Comput. Surv.¡/i¿ 48, 3, Article 48 (February 2016), 43 pages. DOI:https://doi.org/10.1145/2856127

[73] G. Kousiouris, T. Cucinotta, T. Varvarigou, ”The effects of scheduling, workload type and consolidation scenarios on virtual machine performance and their prediction through optimized artificial neural networks”, Journal of Systems and Software, Volume 84, Issue 8, 2011, pp. 1270-1291, ISSN 0164-1212, https://doi.org/10.1016/j.jss.2011.04.013.

[74] Sadeka Islam, Jacky Keung, Kevin Lee, Anna Liu, ”Empirical prediction models for adaptive resource provisioning in the cloud”, Future Generation Computer Systems, Volume 28, Issue 1, 2012,Pages 155-162, ISSN 0167-739X, https://doi.org/10.1016/j.future.2011.05.027 Predicting Resource Usage in Edge Computing Infrastructures 17

[75] A. Litke, K. Tserpes and T. Varvarigou, ”Computational workload prediction for grid oriented industrial applications: the case of 3D-image rendering,” CCGrid 2005. IEEE International Symposium on Cluster Computing and the Grid, 2005., Cardiff, Wales, UK, 2005, pp. 962-969 Vol. 2, doi: 10.1109/CCGRID.2005.1558665.

[76] R. N. Calheiros, E. Masoumi, R. Ranjan and R. Buyya, ”Workload Prediction Using ARIMA Model and Its Impact on Cloud Applications’ QoS,” in IEEE Transactions on Cloud Computing, vol. 3, no. 4, pp. 449-458, 1 Oct.-Dec. 2015, doi: 10.1109/TCC.2014.2350475.

[77] Q. Zhang, L. T. Yang, Z. Yan, Z. Chen and P. Li, ”An Efficient Deep Learning Model to Predict Cloud Workload for Industry Informatics,” in IEEE Transactions on Industrial Informatics, vol. 14, no. 7, pp. 3170-3178, July 2018, doi: 10.1109/TII.2018.2808910.

[78] J. Tan, P. Dube, X. Meng and L. Zhang, ”Exploiting Resource Usage Patterns for Better Utilization Prediction,” 2011 31st International Conference on Distributed Computing Systems Workshops, Minneapolis, MN, 2011, pp. 14-19, doi: 10.1109/ICDCSW.2011.53.

[79] J. Cao, Q. Zhang, and W. Shi, “Challenges and Opportunities in Edge Computing,” in Edge Computing: A Primer, J. Cao, Q. Zhang, and W. Shi, Eds. Cham: Springer International Publishing, 2018, pp. 59–70.

[80] K. Zhang, Y. Zhu, S. Leng, Y. He, S. Maharjan, and Y. Zhang, “Deep Learning Empowered Task Offloading for Mobile Edge Computing in Urban Informatics,” IEEE Internet of Things Journal, vol. 6, no. 5, pp. 7635–7647, Oct. 2019, doi: 10.1109/JIOT.2019.2903191.

[81] D. Zeng, L. Gu, S. Pan, J. Cai, and S. Guo, “Resource Management at the Network Edge: A Deep Reinforcement Learning Approach,” IEEE Network, vol. 33, no. 3, pp. 26–33, May 2019, doi: 10.1109/MNET.2019.1800386.

[82] Q. Yuan, J. Li, H. Zhou, T. Lin, G. Luo, and X. Shen, “A Joint Service Migration and Mobility Optimization Approach for Vehicular Edge Computing,” IEEE Transactions on Vehicular Technology, vol. 69, no. 8, pp. 9041–9052, Aug. 2020, doi: 10.1109/TVT.2020.2999617.

[83] M. Borkowski, S. Schulte, and C. Hochreiner, “Predicting cloud resource utilization,” in Proceedings of the 9th International Conference on Utility and Cloud Computing, New York, NY, USA, Dec. 2016, pp. 37–42, doi: 10.1145/2996890.2996907.

[84] K. Thonglek, K. Ichikawa, K. Takahashi, H. Iida, and C. Nakasan, “Improving Resource Utilization in Data Centers using an LSTM-based Prediction Model,” in 2019 IEEE International Conference on Cluster Computing (CLUSTER), Sep. 2019, pp. 1–8, doi: 10.1109/CLUSTER.2019.8891022.

[85] S.-R. Baig, W. Iqbal, J. L. Berral, and D. Carrera, “Adaptive sliding windows for improved estimation of data center resource utilization,” Future Generation Computer Systems, vol. 104, pp. 212–224, Mar. 2020, doi: 10.1016/j.future.2019.10.026.

[86] S. Baig, W. Iqbal, J. L. Berral, A. Erradi, and D. Carrera, “Adaptive Prediction Models for Data Center Resources Utilization Estimation,” IEEE Transactions on Network and Service Management, vol. 16, no. 4, pp. 1681–1693, Dec. 2019, doi: 10.1109/TNSM.2019.2932840.

[87] G. Kaur, A. Bala, and I. Chana, “An intelligent regressive ensemble approach for predicting resource usage in cloud computing,” Journal of Parallel and Distributed Computing, vol. 123, pp. 1–12, Jan. 2019, doi: 10.1016/j.jpdc.2018.08.008.

[88] M. Yaqub et al., “State-of-the-Art CNN Optimizer for Brain Tumor Segmentation in Magnetic Resonance Images,” Brain Sci, vol. 10, no. 7, Jul. 2020, doi: 10.3390/brainsci10070427. 18 **** ****** et al.

[89] GitHub, T. Pagoulatou, titapag/HBPSHPO, https://github.com/titapag/HBPSHPO.git.

[90] Y. He, W. J. Ma, and J. P. Zhang, “The Parameters Selection of PSO Algorithm influencing On performance of Fault Diagnosis,” MATEC Web Conf., vol. 63, p. 02019, 2016, doi: 10.1051/matecconf/20166302019.

[91] E. Brochu, V. M. Cora, and N. de Freitas, “A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning,” arXiv:1012.2599 [cs], Dec. 2010

[92] GitHub, G. Rodola’, giampaolo/psutil, https://github.com/giampaolo/psutil.

[93] GitHub, A. K. Mortensen, anderskm/gputil https://github.com/anderskm/gputil.

[94] J. Violos, E. Psomakelis, K. Tserpes, F. Aisopos, and T. Varvarigou, “Leveraging User Mobility and Mobile App Services Behavior for Optimal Edge Resource Utilization,” in

Proceedings of the International Conference on Omni-Layer Intelligent Systems, Crete, Greece, May 2019, pp. 7–12, doi: 10.1145/3312614.3312620.

[95] Feurer, M., Klein, A., Eggensperger, K. , Springenberg, J., Blum, M. , and Hutter, F., "Efficient and Robust Automated Machine Learning," in Advances in Neural Information Processing Systems 28, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 2962–2970.

[96] Chen, T., and Guestrin, C., "XGBoost: A Scalable Tree Boosting System," Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794, Aug. 2016, doi: 10.1145/2939672.2939785.

[97] M. Schmitt and R. Wanka, "Particle swarm optimization almost surely finds local optima," Theoretical Computer Science, vol. 561, pp. 57–72, Jan. 2015, doi: 10.1016/j.tcs.2014.05.017.

[98] F. Berkenkamp, A. P. Schoellig, and A. Krause, "No-Regret Bayesian Optimization with Unknown Hyperparameters," Journal of Machine Learning Research, vol. 20, no. 50, pp. 1–24, 2019.

[99] A. Psychas *et al.*, "Cloud toolkit for Provider assessment, optimized Application Cloudification and deployment on IaaS," *Future Generation Computer Systems*, vol. 109, pp. 657–667, Aug. 2020, doi: 10.1016/j.future.2018.09.016.

[100] J. Altmann *et al.*, "BASMATI: An Architecture for Managing Cloud and Edge Resources for Mobile Users," in *Economics of Grids, Clouds, Systems, and Services*, Cham, 2017, pp. 56–66, doi: 10.1007/978-3-319-68066-8_5.