Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων Και Μηχανικων Υπολογιστων
Τομεας Ευφυων Συστηματων, Περιεχομενου και Αλληλεπιδρασης

Κατηγοριοποίηση σε εφαρμογές χειρόγραφων χαρακτήρων με λίγα παραδείγματα

Διπλωματικη Εργασια

του

ΠΑΝΑΓΙΩΤΗ ΓΕΩΡΓΙΑΔΗ

Επιβλέπων : Γιώργος Στάμου
Αναπληρωτής Καθηγητής Ε.Μ.Π.

Συμμετοχή : Παρασκευή Τζούβελη
στην επίβλεψη Εργαστηριακό Διδακτικό Προσωπικό Ε.Μ.Π.

Αθήνα, Οκτώβριος 2019

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Υψηλών Συστημάτων, Περιεχομένου και Αλληλεπίδρασης

Κατηγοριοποίηση σε εφαρμογές χειρόγραφων χαρακτήρων με λίγα παραδείγματα

Διπλωματικη Εργασια

του

ΠΑΝΑΓΙΩΤΗ ΓΕΩΡΓΙΑΔΗ

Επιβλέπων : Γιώργος Στάμου
Αναπληρωτής Καθηγητής Ε.Μ.Π.

Συμμετοχή στην επίβλεψη : Παρασκευή Τζούβελη
Εργαστηριακό Διδακτικό Προσωπικό Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 29η Οκτωβρίου 2019.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Στάμου Γεώργιος	Σταφυλοπάτης Ανδρέας-Γεώργιος	Νικόλαος Παπασπύρου
Αν. Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2019

(Υπογραφή)

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

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου κ. Γεώργιο Στάμου για την ευκαιρία που μου έδωσε να εκπονήσω τη διπλωματική μου εργασία στο Εργαστήριο Ευφυών Συστημάτων, Περιεχομένου και Αλληλεπίδρασης. Επίσης θα ήθελα να ευχαριστήσω την κα. Παρασκευή Τζούβελη για την συνεχή καθοδήγησή της καθόλη την διάρκεια συγγραφής της διπλωματικής αυτής εργασίας.

Ακόμα θα ήθελα να ευχαριστήσω τον αδερφό μου και όλους όσοι στάθηκαν δίπλα μου αυτά τα χρόνια.

Περίληψη

Η ολοένα και αυξανόμενη χρήση των νευρωνικών δικτυών είναι αποτέλεσμα της εκπληκτικής ικανότητας τους να γενικεύουν και να μαθαίνουν καλές αναπαραστάσεις των δεδομένων. Ωστόσο η επιδόση τους είναι συνάρτηση του πλήθους των παραδειγμάτων που διαθέτουν για την εκπαιδευσή τους.

Η εργασία αυτή εκτελεί ανάλυση των βασικών χαρακτηριστικών των δικτύων που χρησιμοποιούνται σε περιπτώσεις με πολύ λίγα παραδείγματα, όπου οι παραδοσιακές μορφές νευρωνικών δικτύων αποτυγχάνουν. Στο πλαίσιο αυτής της εργασίας συγκρίνουμε τις αρχιτεκτονικές στο πλαίσιο του One-shot learning, δηλαδή στην ικανότητα τους να μάθουν μια καινούργια κλάση, με μόνο ένα παράδειγμα από την κλάση αυτή. Συγκεκριμένα αναλύονται οι κυρίαρχες αρχιτεκτονικές που έχουν χρησιμοποιηθεί στο ευρέως διαδεδομένο Omniglot Dataset. Οι αρχιτεκτονικές συγκρίνονται τόσο ως προς την ικανότητα τους να διακρίνουν τους χαρακτήρες όσο και ως προς τις παραμέτρους που χρησιμοποιούν και τον χρόνο εκπαίδευσης.

Στη συνέχεια υλοποιούνται ορισμένες από τις τεχνικές αυτές χρησιμοποιώντας το ίδιο συνελικτικό δίκτυο ως εξαγωγέα χαρακτηριστικών, το οποίο συνεπάγεται ότι έχουν σχεδόν το ίδιο σύνολο παραμέτρων μάθησης για να επιλύσουν το πρόβλημα (όποιες επιπλεόν παράμετροι χρησιμοποιούνται για την επίλυση του προβλήματος αναφέρονται). Οι μέθοδοι εκπαιδεύονται και αξιολογούνται σε χαρακτήρες από την ίδια αλφάβητο αλλά και από διαφορετικές, συγκρίνοντας τα αποτελέσματα όλων των διαφορετικών περιπτώσεων. Παράλληλα μελετώνται τεχνικές που μπορούν να χρησιμοποιηθούν για να βελτιώσουν την επίδοση των συστημάτων και προτείνονται κατευθύνσεις για μελλοντική έρευνα.

Λέξεις Κλειδιά

Μηχανική Μάθηση, Συνελικτικό Νευρωνικό Δίκτυο, Βαθιά Μάθηση, Όραση Υπολογιστών

Abstract

The continuous increase in the usage of neural network is a result of their remarkable ability to generalize and learn good data representations. However their performance is correlated with the amount of data that are used to train them.

This thesis analyzes the basic characteristics of architectures that are used in cases where the available data are insufficient to train conventional neural networks. In this context we compare architectures in the task of One-shot Learning, that challenges their ability to learn new classes with just a single example from that class. Specifically, the state-of-the-art architectures that have been used for the well-know Omniglot Dataset are analyzed. The architectures are compared according to their ability to distinct the characters but also with respect to the amount of training parameters and time of training.

Next, we implement a subset of these architectures using the same underlying convolutional neural network as feature extractor, and therefore the compared networks use approximately the same number of learning parameters (the architectures that use extra are noted). The different methods are trained and evaluated with characters from the same alphabet but also between different alphabets, comparing the results for all possible cases. Additionally, we explore techniques that can increase the performance of these methods and propose some paths for future research.

Keywords

Machine Learning, Deep Learning, Low-shot Learning, Convolutional Neural Network, Computer Vision

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κεφάλαιο 1

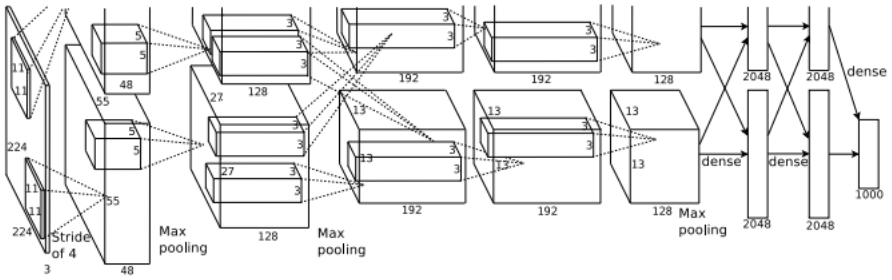
Εισαγωγή

1.1 Σύντομη ιστορική αναδρομή

Η εξέλιξη της Τεχνητής Νοημοσύνης αποτελεί αυτήν τη στιγμή την κινητήριο δύναμη των τεχνολογικών καινοτομιών στον πλανήτη. Αν και το ερώτημα αν οι υπολογιστές σκέφτονται αποτέλεσε αντικείμενο για θερμούς διαλόγους, κανένας δεν μπορεί να αμφισβητήσει ότι ο κόσμος αλλάζει και μάλιστα με ταχύτατους ρυθμούς. Το 1997 στην δεύτερη σειρά παρτίδων του παγκόσμιου πρωταθλητή στο σκάκι Garry Kasparov εναντίον του υπερυπολογιστή της IBM Deep Blue, για πρώτη φορά ο υπολογιστής κερδίζει υπό πραγματικές συνθήκες αγωνιστικού σκακιού. Αυτήν δεν ήταν σαφώς η αρχή της τεχνητής νοημοσύνης, αλλά ήταν η πρώτη φορά όπου ένα τόσο απτό αποτέλεσμα έγινε γνωστό παγκοσμίως, με τη βοήθεια βέβαια και της προβολής που του δόθηκε. Οι επιτυχίες δεν σταμάτησαν και πλέον οι εφαρμογές της Τεχνητής Νοημοσύνης έχουν εξαπλωθεί σε ένα ευρύ φάσμα που περιλαμβάνει από chat bots μέχρι αυτοκινούμενα αμάξια.

Η Τεχνητή Νοημοσύνη είναι ένας ευρύς κλάδος του οποίου ένα πεδίο είναι η Βαθιά Μάθηση. Η Βαθιά Μάθηση αν και σήμερα αποτελεί ένα πολύ δημοφιλές αντικείμενο μελέτης της επιστημονικής κοινότητας, αυτό σε καμία περίπτωση δεν μπορεί να ειπωθεί και για το παρελθόν της. Αυτό βέβαια γίνεται αντιληπτό και από το γεγονός ότι ο κλάδος αυτός εμφανίζεται με διαφορετικό όνομα ανά περιόδους. Το 1943 οι McCulloch και Pits ανέπτυξαν τον McCulloch-Pits νευρώνα, του οποίου η λειτουργία θύμιζε έναν βιολογικό νευρώνα, ο οποίος χρησιμοποιήθηκε για τον διαχωρισμό δύο κατηγοριών, του οποίου τα βάρη τα έθετε ένας άνθρωπος χειριστής. Στην συνέχεια το 1950 ο Rosenblatt ανέπτυξε τον αισθητήρα (perceptron) ο οποίος ήταν επίσης ένα γραμμικό μοντέλο που όμως μάθαινε τα βάρη μόνος του, μέσω παραδειγμάτων από την κάθε κατηγορία, ενώ το 1960 οι Widrow και Hoff με το adaptive linear element (ADALINE), μάθαιναν να προβλέπουν τις τιμές μιας συνάρτησης από τα δεδομένα. Έτσι ολοκληρώνεται η πρώτη περίοδος έρευνας της Βαθιάς Μάθησης γνωστή και ώς cybernetics. Έπειτα από μια αδρανή περίοδο, η Βαθιά Μάθηση επανέρχεται ως συνδετισμός (connectionism), εμπνευσμένη από τη γνωσιακή επιστήμη. Ο Hinton το 1986 εισήγαγε την ιδέα των κατανεμημένων αναπαραστάσεων (distributed representations), όπου κάθε οντότητα αναπαριστάται από πολλά χαρακτηριστικά και κάθε χαρακτηριστικό εμπλέκεται στην αναπαράσταση πολλών διαφορετικών οντοτήτων. Ορόσημο για τη Βαθιά Μάθηση υπήρξε η εφαρμογή του αλγορίθμου της οπισθοδιάδοσης (backpropagation) για την εκπαίδευση νευρωνικών δικτύων το 1986 με τις δουλειές των Rumelhart, Hinton και Williams , ενώ η σύγχρονη μορφή του αλγορίθμου προήλθε από τον LeCun το 1987. Η περίοδος αυτή ολοκληρώνεται με την ανακάλυψη των Δικτύων Μακράς Βραχείας Μνήμης (LSTM) τα οποία συνέβαλαν στην μοντελοποίηση ακολουθιών μέσω νευρωνικών δικτύων. Παρόλο που επικράτησε μια περίοδος όπου η πλειοψηφία της επιστημονικής κοινότητας δεν ασχολούταν με το κομμάτι της

Βαθιάς Μάθησης, οι LeCun, Bengio και Hinton με την δουλεία τους στο πλαίσιο του Canadian Institute for Advanced Research (CIFAR) συνέχισαν την έρευνα πάνω σε αυτόν τον κλάδο. Μάλιστα για την προσπάθεια τους αυτή, βραβεύτηκαν το 2018 με Turing Award. Το τρίτο κύμα, το οποίο ονομάστηκε και Βαθιά Μάθηση, ξεκίνησε με τη δουλεία του Hinton το 2006 πάνω στην γρήγορη εκμάθηση των deep belief network. Ακολούθησαν αντίστοιχες ανακαλύψεις άλλων επιστημόνων του CIFAR πάνω στην εκπαίδευση και την γενίκευση των νευρωνικών δικτύων. Παράλληλα οι δυνατότητες της παράλληλης επεξεργασίας συνεχίζουν να βελτιώνονται και έχουμε πλέον GPUs με σημαντικές υπολογιστικές δυνατότητες. Το 2012 ο Krizhevsky με το AlexNet βελτίωσε θεαματικά το λάθος στο διαγωνισμό του ImageNet Large Scale Visual Recognition Challenge (ILSVRC) χρησιμοποιώντας ένα βαθύ νευρωνικό δίκτυο. Από εκεί και ύστερα τα βαθιά νευρωνικά δίκτυα επανήλθαν στο προσκήνιο και πλέον χιλιάδες επιστήμονες στον κόσμο ασχολούνται με τη μελέτη τους.



Σχήμα 1.1: Παράδειγμα της αρχιτεκτονικής του AlexNet [11].

1.2 Few-shot K-way Learning

Ένα νευρωνικό δίκτυο καλείται να μάθει από κάποια παραδείγματα και να προσαρμόσει τα βάρη του κατάλληλα, προκειμένου να είναι εφτικτός ο διαχωρισμός των κλάσεων στις οποίες ανήκουν τα παραδείγματα. Η διαδικασία κατά την οποία το δίκτυο μαθαίνει να διαχωρίζει τα παραδείγματα καλείται εκπαίδευση. Στη συνέχεια ακολουθεί το στάδιο αξιολόγησης (evaluation). Στο πλαίσιο της Επιβλεπόμενης Μάθησης (Supervised Learning) οι ετικέτες (labels), οι οποίες προσδιορίζουν την κλάση στην οποία ανήκει το κάθε παράδειγμα, είναι διαθέσιμες στο στάδιο της εκπαίδευσης. Το Few-shot Learning είναι ένα πλαίσιο εντός του οποίου το δίκτυο καλείται να μάθει γρήγορα και από λίγα παραδείγματα. Κατά την εκπαίδευση, δίνεται περιορισμένος αριθμός παραδειγμάτων από διάφορες κλάσεις με τις ετικέτες τους και το δίκτυο καλείται να μάθει γενικά χαρακτηριστικά για το πρόβλημα, όπως χαρακτηριστικά τα οποία έχουν κοινά παραδείγματα της ίδιας κλάσης. Μια αντίθεση με τη διαδικασία μάθησης των παραδοσιακών νευρωνικών δικτύων είναι ότι δεν αρκεί το δίκτυο να μάθει καλές αναπαραστάσεις για τις κλάσεις της εκπαίδευσης, καθώς οι κλάσεις αξιολόγησης είναι διαφορετικές και δεν παρουσιάζονται στην εκπαίδευση, αλλά είναι επιθυμητό να μάθει χαρακτηριστικά τα οποία διαχωρίζουν τις κλάσεις. Το στάδιο αξιολόγησης αποτελείται από **επεισόδια** που έχουν την εξής μορφή:

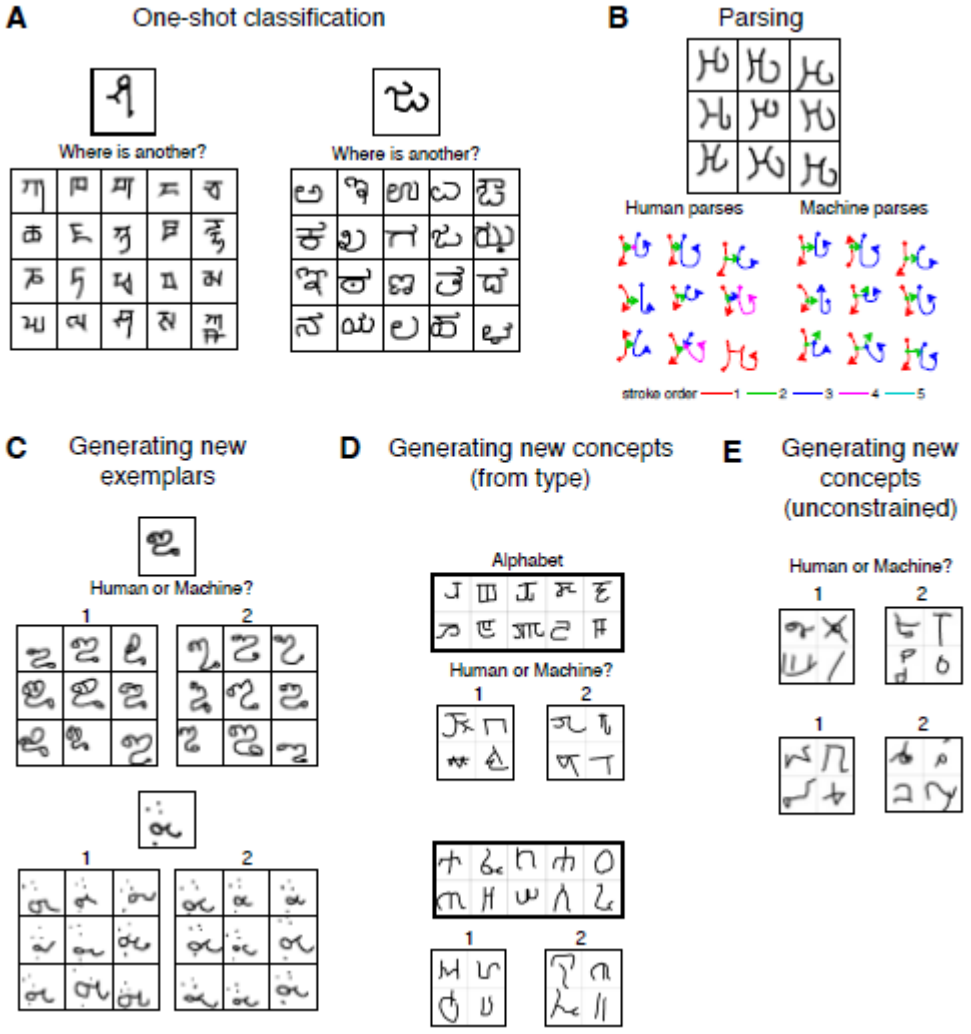
- Στάδιο 1: Δίνονται N παραδείγματα, ίσα με τον αριθμό του Few-shot, δηλαδή αν έχουμε One-shot ένα, Five-shot πέντε κ.ο.κ. από K κλάσεις, τις οποίες δεν έχει δει το δίκτυο στο στάδιο της εκπαίδευσης. Από αυτό το δείγμα, το οποίο καλείται και Σύνολο Υποστήριξης (Support Set), το δίκτυο καλείται να προσαρμοστεί στις κλάσεις.

- Στάδιο 2: Παρουσιάζονται τυχαία παραδείγματα από τις κλάσεις αυτές, διαφορετικά από

το προηγούμενο στάδιο, τα οποία κατηγοριοποιεί το δίκτυο. Το σύνολο των παραδειγμάτων αυτού του σταδίου είναι γνωστό ως Σύνολο Ερωτημάτων (Query Set)

Σημειώνεται ότι η παραπάνω διαδικασία (επεισόδιο) επαναλαμβάνεται πολλές φορές με τυχαίες κλάσεις και παραδείγματα κάθε φορά, τα οποία δειγματοληπτούνται από το σύνολο αξιολόγησης (στο σχήμα 1.2 φαίνεται ένα παράδειγμα με χειρόγραφους χαρακτήρες από το Omniglot Dataset). Αν και ο άνθρωπος φαίνεται να έχει τη δυνατότητα να ανταπεξέλθει σε αυτήν τη διαδικασία [12], τα παραδοσιακά νευρωνικά δίκτυα απαιτούν πολλά παραδείγματα παραπάνω για να γενικεύσουν αποτελεσματικά, πετυχαίνοντας τον αντίστοιχο βαθμό απόδοσης.

Όπως γίνεται άμεσα αντιληπτό από την περιγραφή της διαδικασίας αξιολόγησης, μεγαλώνοντας τον αριθμό των κλάσεων K των επεισοδίων, η εργασία γίνεται δυσκολότερη, γιατί το δίκτυο πρέπει να αποφανθεί ανάμεσα σε περισσότερες κλάσεις για ένα παράδειγμα. Το αντίθετο ισχύει για το N, δηλαδή το Zero-shot είναι σαφώς δυσκολότερο από το One-shot, το οποίο είναι δυσκολότερο από το Five-shot κ.ο.κ. καθώς περισσότερα παραδείγματα με τις ετικέτες τους γίνονται διαθέσιμα από την κάθε κλάση στο 1ο στάδιο του επεισοδίου. Ιδιαίτερο ενδιαφέρον παρουσιάζουν οι περιπτώσεις των One-shot και Zero-shot Learning όπου κατά το στάδιο της αξιολόγησης εμφανίζεται ένα μόνο παράδειγμα από τις υποψήφιες κλάσεις, και μόνο μετα-δεδομένα (meta-data) για την περίπτωση του Zero-shot αντίστοιχα. Στο πλαίσιο αυτής της διπλωματικής επικεντρωνόμαστε στην περίπτωση του One-shot Learning για την υλοποίηση και σύγκριση των δικτύων.



Σχήμα 1.2: Ένα επεισόδιο από το Omniglot dataset [13].

1.3 Στόχοι και Συνεισφορές της Εργασίας

Ο στόχος της εργασίας αυτής είναι η μελέτη και σύγκριση των κυριότερων αρχιτεκτονικών που χρησιμοποιούνται στη βιβλιογραφία για να αντιμετωπίσουν το πρόβλημα του Few-shot Learning . Οι μέθοδοι που μελετώνται ανήκουν στην κατηγορία των μεθόδων διάκρισης (discriminative methods), δηλαδή ασχολούνται με το κομμάτι της κατηγοριοποίησης αποκλειστικά και όχι με την παραγωγή νέων παραδειγμάτων. Το πλαίσιο της σύγκρισης είναι το Omniglot Dataset (2015) των Lake, Salakhutdinov και Tenenbaum [12], [13].

Αν και πολλές από αυτές τις μεθόδους έχουν εξεταστεί στο πλαίσιο αυτού του συνόλου δεδομένων, η σύγκριση γίνεται σε διαφορετικά σύνολα εκπαίδευσης και διαφορετικές εργασίες (task). Η συνεισφορά αυτής της εργασίας εξετάζει τις μεθόδους αυτές στο ίδιο περιβάλλον, χρησιμοποιώντας την ίδια αρχιτεκτονική για το συνελικτικό νευρωνικό δίκτυο που χρησιμοποιείται ως εξαγωγέας χαρακτηριστικών (feature extractor), συγκρίνοντας παράλληλα χαρακτηριστικά όπως ο χρόνος εκπαίδευσης, η μνήμη που χρησιμοποιούν και η δυσκολία υλοποίησης.

1.4 Διάρθρωση της Εργασίας

Η εργασία αυτή είναι οργανωμένη σε 8 κεφάλαια

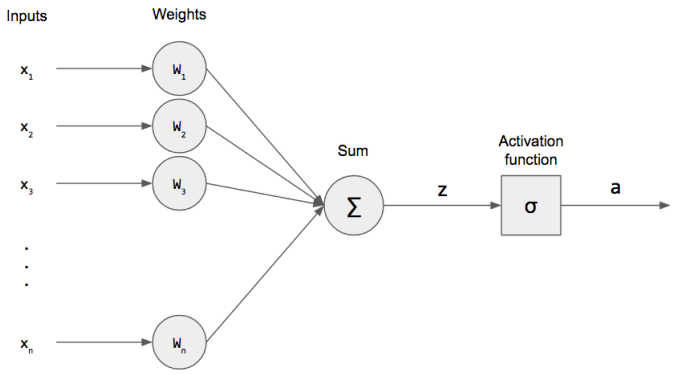
- Στο Κεφάλαιο 2 γίνεται ανασκόπηση των κυριότερων αναφορών της βιβλιογραφίας σχετικά με τα συνελικτικά νευρωνικά δίκτυα και παρουσιάζονται κάποια χαρακτηριστικά του Few-shot Learning.
- Στο Κεφάλαιο 3 παρουσιάζονται προσεγγίσεις βασισμένες σε μετρικές για την επίλυση του προβλήματος.
- Στο Κεφάλαιο 4 παρουσιάζονται αλγόριθμοι βελτιστοποίησης για την επίλυση του προβλήματος.
- Στο Κεφάλαιο 5 παρουσιάζονται αρχιτεκτονικές που βασίζονται στην μνήμη.
- Στο Κεφάλαιο 6 δίνονται τα βασικά στοιχεία της υλοποίησης και τα πειραματικά αποτελέσματα.
- Στο Κεφάλαιο 7 γίνεται σύνοψη των κυριότερων αποτελεσμάτων και συνεισφορών της εργασίας και προτείνονται πιθανές κατευθύνσεις για μελλοντική έρευνα οι οποίες προκύπτουν από την εργασία αυτή.
- Στο Κεφάλαιο 8 δίνονται τα πιο αντιπροσωπευτικά κομμάτια κώδικα της κάθε υλοποιησης.

Κεφάλαιο 2

Ανασκόπηση των συνελικτικών νευρωνικών δικτύων και του Few-shot Learning

2.1 Συνελικτικά νευρωνικά δίκτυα

Τα συνελικτικά νευρωνικά δίκτυα αποτελούν μια μορφή νευρωνικού δικτύου που όμως έχουν την δυνατότητα να εκμεταλλευτούν την χωρική συσχέτιση των δεδομένων εισόδου [15]. Ένα παραδοσιακό νευρωνικό δίκτυο αποτελείται από ένα σύνολο μονάδων, γνωστών ως αισθητήρων (perceptron), όπου δρουν ως ένας γραμμικός μετασχηματισμός πάνω στο διάνυσμα εισόδου ακολουθούμενο από μια συνάρτηση ενεργοποίησης, η οποία είναι μη γραμμική.



Σχήμα 2.1: Παράδειγμα ενός perceptron [19].

Δεν υπάρχει περιορισμός για το βάθος του νευρωνικού δικτύου, το οποίο ορίζεται ως ο αριθμός των αισθητήρων που στοιβάζονται, και υπάρχει η τάση να αυξάνεται το βάθος με το πέρασμα του χρόνου. Η έξοδος ενός νευρωνικού δικτύου με 2 επίπεδα παράγει την έξοδο:

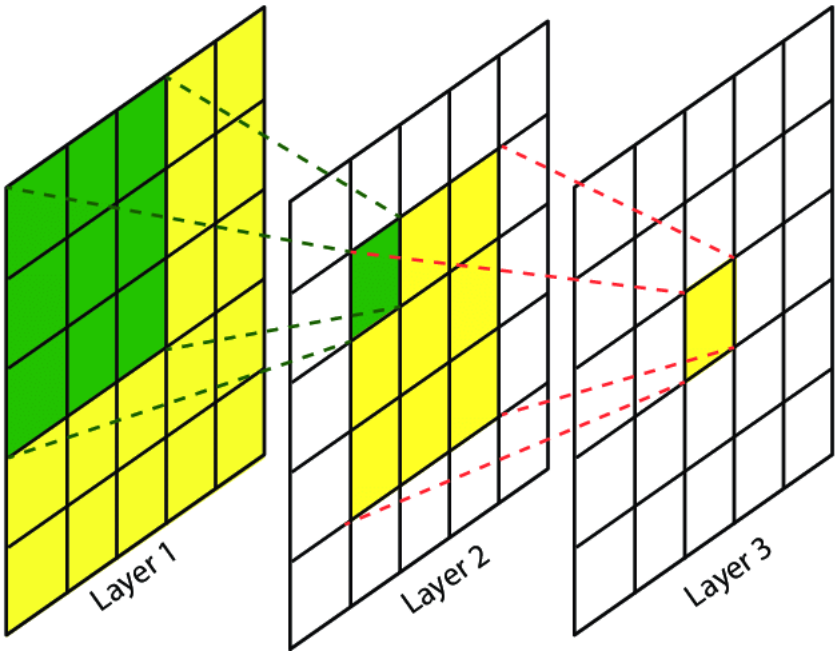
$$y_k = \sigma\left(\sum_{j=0}^{M} w_{kj}^{(2)} \sigma\left(\sum_{i=0}^{D} w_{ji}^{(1)} x_i\right)\right) \quad k = 1, 2, ..., K \tag{2.1}$$

Ένα συνελικτικό νευρωνικό δίκτυο δέχεται πρότυπα τα οποία αναπαρίστανται με τοπολογίες πλέγματος, όπως εικόνες που μπορεί να είναι 2 ή 3 διαστάσεων, ανάλογα αν είναι

ασπρόμαυρες ή έγχρωμες. Η διαφορά τους σε σχέση με τα νευρωνικά δίκτυα είναι ότι χρησιμοποιούν τη διακριτή συνέλιξη αντί του πολλαπλασιασμού πινάκων σε τουλάχιστον ένα από τα επίπεδά τους. Η πράξη της διακριτής συνέλιξης για μια δισδιάστατη εικόνα I με έναν δισδιάστατο πυρήνα K ως είσοδο ορίζεται ως:

$$S_{ij} = (\boldsymbol{I} * \boldsymbol{K})_{ij} = \sum_m (\sum_n (I_{mn} K_{(i-m)(j-n)}))$$

Ο πυρήνας K ονομάζεται και φίλτρο και κάθε συνελικτικό επίπεδο αποτελείται από ένα σύνολο από φίλτρα, οι παράμετροι των οποίων μαθαίνονται κατά την διαδικασία της εκπαίδευσης. Ένα βασικό χαρακτηριστικό των πυρήνων που χρησιμοποιούνται είναι ότι η διάσταση τους είναι σημαντικά μικρότερη από την είσοδο, και έτσι πετυχαίνουν αραιές συνδέσεις και λιγότερες παραμέτρους εκμάθησης. Αυτό μειώνει σημαντικά το υπολογιστικό κόστος για τον υπολογισμό της εξόδου του συνελικτικού νευρωνικού δικτύου. Σε ένα συνελικτικό δίκτυο τα βαθύτερα επίπεδα έχουν μεγαλύτερο προσπελάσιμο πεδίο (receptive field) με αποτέλεσμα να αλληλεπιδρούν με μεγαλύτερο κομμάτι της εισόδου (Σχήμα 2.2). Επομένως αυτά τα επίπεδα μαθαίνουν πιο υψηλού επιπέδου (high level) χαρακτηριστικά σε σχέση με τα προηγούμενα, καθιστώντας τα συνελικτικά δίκτυα πολύ αποτελεσματικά. Κάποιες από τις εφαρμογές τους σε εικόνες είναι οι κατηγοριοποίηση [22], παραγωγή [5], νοηματική τμηματοποίηση [26] κ.α.



Σχήμα 2.2: Στοιβάζοντας 3 συνελικτικά επίπεδα 3×3 επιτυγχάνεται ένα προσπελάσιμο πεδίο 7×7 στο επίπεδο 3.

Ένα τυπικό επίπεδο ενός συνελικτικού δικτύου συνήθως ανήκει σε μια από τις παρακάτω κατηγορίες:

- Συνελικτικό επίπεδο (Convolutional layer): Το επίπεδο αυτό εκτελεί την διακριτή συνέλιξη που αναφέρθηκε προηγουμένως.
- Επίπεδο ενεργοποίησης (Activation layer): Σε αυτό το επίπεδο εφαρμόζεται η συνάρτηση ενεργοποίησης στην έξοδο του προηγούμενου επιπέδου. Οι πιο συνήθεις συναρτήσεις ενεργοποίησης είναι η λογιστική συνάρτηση $f(x) = 1/(1 + \exp(-x))$, η υπερβολική

εφαπτομένη $f(x) = \tanh(x)$ και η συνάρτηση ReLU $f(x) = \max(0, x)$. Από τις συναρτήσεις αυτές, η συνάρτηση ReLU είναι η πιο διαδεδομένη και οδηγεί σε καλύτερη εκπαίδευση για βαθιά δίκτυα [16].

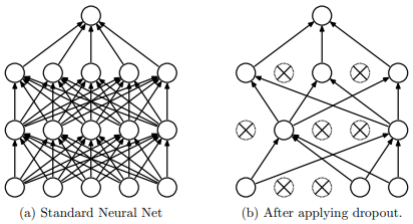
- Επίπεδο τραβήγματος (Pooling layer): Το επίπεδο αυτό συγχωνεύει τα χαρακτηριστικά του διανύσματος του προηγούμενου επιπέδου. Συγκεκριμένα εφαρμόζεται σε ξεχωριστές γειτονίες του διανύσματος χαρακτηριστικών που μπορούν και να επικαλύπτονται, και δίνει στην έξοδο ένα χαρακτηριστικό που προκύπτει από τη γειτονιά αυτή. Συνήθως χρησιμοποιείται μέγιστο τράβηγμα (Max Pooling) όπου δίνει ως έξοδο το μέγιστο της γειτονιάς και μέσο τράβηγμα (Average Pooling) η έξοδος του οποίου είναι ο μέσος όρος της γειτονιάς. Το επίπεδο αυτό συμβάλει στην ανεξαρτητοποίηση από τοπική μετατόπιση (local translation invariance), παράγοντας βελτιωμένα αποτελέσματα σε περιπτώσεις που είναι επιθυμητό να βρεθεί αν η είσοδος διαθέτει κάποιο χαρακτηριστικό, όπως για παράδειγμα να αναγνωριστεί αν υπάρχει ένα αντικείμενο σε μια εικόνα.

Μια συνηθισμένη πρακτική που χρησιμοποιείται είναι να εφαρμόζονται διαδοχικά τα τρία παραπάνω επίπεδα. Το επίπεδο τραβήγματος χρησιμοποιείται κυρίως για τον υποβιβασμό της διάστασης του προηγούμενου επιπέδου (Σχήμα 1.1). Σε πολλές αρχιτεκτονικές κατηγοριοποίησης, το διάνυσμα χαρακτηριστικών του τελευταίου συνελικτικού επιπέδου τροφοδοτείται σε πλήρως συνδεδεμένα επίπεδα (fully connected layers) στα οποία επιτυγχάνεται η διάκριση των παραδειγμάτων στις κλάσεις. Όπως θα δούμε στην συνέχεια πολλές από τις αρχιτεκτονικές που χρησιμοποιούνται για το Few-shot Learning αντί για πλήρως συνδεδεμένα επίπεδα, υπολογίζουν αποστάσεις ή ομοιότητες στα διανύσματα χαρακτηριστικών που προκύπτουν από το τελευταίο συνελικτικό επίπεδο. Ένας από τους λόγους που αυτή η πρακτική παρουσιάζει καλά αποτελέσματα είναι επειδή τα πλήρως συνδεδεμένα επίπεδα εισάγουν ένα μεγάλο πλήθος παραμέτρων, δυσανάλογο με τα συνελικτικά επίπεδα, και εξαιτίας του περιορισμένου μεγέθους του συνόλου των δεδομένων στις εφαρμογές του Few-shot Learning, οι παράμετροι δεν καταφέρνουν να εκπαιδευτούν ικανοποιητικά.

2.2 Σύγχρονες πρακτικές στα συνελικτικά νευρωνικά δίκτυα

Από τη μεγάλη εξάπλωση των νευρωνικών δικτύων που συντελέστηκε το 2012 με το AlexNet [11], έχουν προταθεί πολλές μέθοδοι και διαφορετικές αρχιτεκτονικές για την επίτευξη καλύτερων αποτελεσμάτων. Ένα από τα προβλήματα για το οποίο έχει γίνει εκτεταμένη προσπάθεια να αντιμετωπιστεί είναι η τάση που παρουσιάζουν τα νευρωνικά δίκτυα να υπερεκπαιδεύονται στο σύνολο των δεδομένων που χρησιμοποιήθηκε για την εκπαίδευση τους (overfitting). Μια, ίσως προφανής, λύση είναι η χρησιμοποίηση μεγαλύτερου ή και πιο αντιπροσωπευτικού συνόλου δεδομένων, το οποίο πλησιάζει περισσότερο το σύνολο αξιολόγησης. Αυτή η πρακτική, πέρα του ότι απαιτεί ανθρώπινη εργασία, δεν είναι πάντα εφικτή και είναι ένα από τα βασικά χαρακτηριστικά του προβλήματος του Few-shot Learning. Μια άλλη λύση που προτείνεται από τους Simard et al. [21] είναι ο εμπλουτισμός του συνόλου δεδομένων με την εφαρμογή μετασχηματισμών, όπως αφινικοί μετασχηματισμοί, στα υπάρχοντα δεδομένα (distorted data). Με την εφαρμογή μετασχηματισμών που διατηρούν τη φύση του προβλήματος αναλλοίωτη (transformation invariance), όπως για παράδειγμα η χρήση αφινικού μετασχηματισμού για αναγνώριση ενός αντικειμένου σε μια εικόνα, επιτυγχάνεται καλύτερη απόδοση του νευρωνικού δικτύου [21]. Μια πιο σύγχρονη τεχνική που προτάθηκε από τον Srivastava et al. [8] ονομάζεται Dropout , κατά την οποία μηδενίζονται οι έξοδοι των ενδιάμεσων επιπέδων του δικτύου με πιθανότητα 1/2 κατά την διάρκεια της εκπαίδευσης. Η τεχνική αυτή

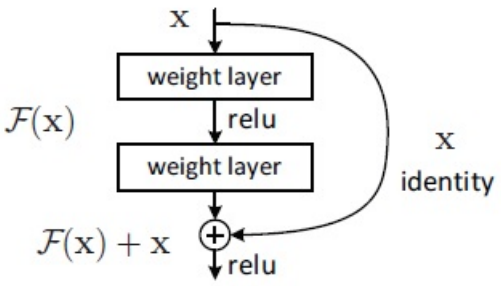
ισοδυναμεί με model averaging από ένα σύνολο αρχιτεκτονικών εκθετικό ως προς το πλήθος των παραμέτρων. Αυτό προκύπτει εύκολα λαμβάνοντας υπόψη ότι σε κάθε εποχή εκπαιδεύεται και ένα δίκτυο με διαφορετική τοπολογία, με τα διαφορετικά δίκτυα να μοιράζονται τις παραμέτρους (parameter sharing). Με αυτό τον τρόπο μειώνεται η συνπροσαρμοστικότητα (co-adaptation) των νευρώνων του δικτύου, οδηγώντας σε καλύτερη γενίκευση [8].



Σχήμα 2.3: Εφαρμογή Dropout κατά την εκπαίδευση. Αριστερά: Η δομή ενός δικτύου με 2 κρυφά επίπεδα. Δεξιά: Ένα thinned νευρωνικό δικτύο που έχει προκύψει τυχαία με την εφαρμογή του Dropout κατά την εκπαίδευση [8].

Ένα άλλο πρόβλημα που παρουσιάζουν τα βαθιά νευρωνικά δίκτυα είναι η εξαφανιζόμενη κλίση (vanishing gradient), η οποία προκύπτει από τις μεθόδους μάθησης που βασίζονται στην κλίση (gradient-based) και το μεγάλο βάθος του δικτύου. Επειδή σύμφωνα με αυτές τις μεθόδους μάθησης τα βάρη του δικτύου ανανεώνονται με βάση τη μερική παράγωγο της συνάρτησης λάθους ως προς το παρόν βάρος, αν η μερική παράγωγος γίνει πολύ μικρή, τείνοντας στο μηδέν, τα βάρη δεν ανανεώνονται. Η μέθοδος της οπισθοδιάδοσης προωθεί την παράγωγο του λάθους από το βαθύτερο επίπεδο προς τα πίσω, χρησιμοποιώντας τον κανόνα της αλυσίδας, όπου οι παράγωγοι του δικτύου πολλαπλασιάζονται καθώς διαδίδονται προς τα πίσω. Χρησιμοποιώντας μια συνάρτηση ενεργοποίησης όπως η λογιστική συνάρτηση, της οποίας το πεδίο ορισμού είναι το $\mathbb{R}$ αλλά το πεδίο τιμών της παραγώγου είναι ένα διάστημα μεταξύ του 0 και 1, η παράγωγος μικραίνει με κάθε επίπεδο καθώς πολλαπλασιάζεται προς τα πίσω. Οπότε όταν έχουμε πολλά επίπεδα και μικρές παραγώγους, τα βάρη των αρχικών επιπέδων δεν θα ενημερωθούν αποτελεσματικά, με συνέπεια η απόδοση του δικτύου να είναι μειωμένη.

Μια λύση για το πρόβλημα αυτό είναι η χρησιμοποίηση συντομότερων συνδέσεων (shortcut connections) για την διάδοση των παραγώγων [7]. Έτσι δημιουργούνται επιμέρους κομμάτια τα οποία ονομάζονται Υπολειπόμενα κομμάτια (Residual blocks) και το δίκτυο που χρησιμοποιεί αυτά τα δομικά στοιχεία καλείται Υπολειπόμενο δίκτυο (ResNet). Το πλεονέκτημα με την χρήση υπολειπόμενων κομματιών γίνεται εμφανές αν πρέπει να διατηρηθεί η είσοδος, δηλαδή να μάθει το δίκτυο την ταυτοτική συνάρτηση $F(\boldsymbol{x}) = \boldsymbol{x}$. Χωρίς τις συντομότερες συνδέσεις το δίκτυο θα έπρεπε μέσω μιας στοίβας μη γραμμικών επιπέδων να παράξει την ταυτοτική συνάρτηση. Αντιθέτως με τις συντομότερες συνδέσεις αρκεί να μάθει την μηδενικη $F(\boldsymbol{x}) = \mathbf{0}$, οπότε η έξοδος θα είναι $H(\boldsymbol{x}) = \boldsymbol{x} + \mathbf{0} = \boldsymbol{x}$.



Σχήμα 2.4: Παράδειγμα ενός Υπολειπόμενου κομματιού με συντομότερες συνδέσεις [7].

Τέλος μια μέθοδος που στοχεύει στην αντιμετώπιση του covariant shift που παρουσιάζουν τα νευρωνικά δίκτυα και παράλληλα συνεισφέρει στην ταχύτερη εκπαίδευση καλείται Κανονικοποίηση Συστάδας (Batch Normalization) και κάνει whitening (μηδενική μέση τιμή και μοναδιαία διασπορά) στις εξόδους του κάθε επιπέδου ενεργοποίησης ως προς το εύρος των τιμών που παίρνουν σε μια συστάδα (batch) από παραδείγματα εισόδου [9]. Η μέθοδος αυτή εισάγει 2 επιπλέον παραμέτρους σε κάθε επίπεδο, οι οποίες εκπαιδεύονται, όπως και το υπόλοιπο δίκτυο, με την μέθοδο της οπισθοδιάδοσης. Οι παράμετροι αυτοί έχουν τη δυνατότητα να αναιρέσουν το whitening, αν οδηγεί σε καλύτερο αποτέλεσμα, και άρα από τη μέθοδο αυτή μπορεί να προκύψει το αρχικό δίκτυο.

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Σχήμα 2.5: Ο αλγόριθμος Κανονικοποίησης Συστάδας [9].

2.3 Few-shot Learning Dataset

Στο προηγούμενο κεφάλαιο περιγράφηκε το πλαίσιο του Few-shot Learning με τις διάφορες παραλαγές του. Ωστόσο, στον κλάδο της Μηχανικής Μάθησης και ιδιαίτερα στη Βαθιά Μηχανική Μάθηση, η διαδικασία ανάπτυξης καινούργιων μεθόδων αρχίζει λίγο διαφορετικά από άλλους κλάδους της Επιστήμης των Υπολογιστών. Ενώ η συνηθισμένη μέθοδος είναι η διατύπωση ενός θεωρητικού και καλά ορισμένου προβλήματος το οποίο στη συνέχεια η επιστημονική κοινότητα καλείται να επιλύσει, στον κλάδο της Μηχανικής Μάθησης πολλές φορές ένα σύνολο δεδομένων, με σαφώς ορισμένες κατατμήσεις εκπαίδευσης και επαλήθευσης γίνεται το κίνητρο και το μέτρο σύγκρισης για τις μεθόδους που ερευνώνται. Έτσι στη βιβλιογραφία όταν συγκρίνονται διαφορετικές μέθοδοι, το μέτρο σύγκρισης τους είναι η επίδοση σε κάποια ευρέως διαδεδομένα και κοινώς αποδεκτά σύνολα δεδομένων για τη συγκεκριμένη εργασία (task). Όσον αφορά το Few-shot Learning και ιδιαίτερα το κομμάτι της κατηγοριοποίησης, έχουν προταθεί και χρησιμοποιηθεί διάφορα σύνολα δεδομένων. Στη συνέχεια παρουσιάζουμε

κάποια από τα πιο ευρέως διαδεδομένα:

- Omniglot Dataset: Το συγκεκριμένο σύνολο δεδομένων αναπτύχθηκε το 2011 από τους Lake, Salakhutdinov, Gross και Tenenbaum και αποτελείται από 1623 χαρακτήρες από 50 διαφορετικά αλφάβητα. Κάθε χαρακτήρας έχει σχεδιαστεί από 20 διαφορετικούς ανθρώπους [12], [13]. Οι χαρακτήρες αυτοί δίνονται σε μορφή ασπρόμαυρης (grayscale) εικόνας διαστάσεως 108×108. Έτσι, θεωρώντας ως κλάσεις τους διαφορετικούς χαρακτήρες, έχουμε 1623 κλάσεις με 20 παραδείγματα από την κάθε κλάση. Όπως θα δούμε στην συνέχεια το πλήθος των παραδειγμάτων είναι πολύ μικρό σε σχέση με τον αριθμό των κλάσεων συγκρινόμενο με άλλα σύνολα δεδομένων. Η αρχική μορφή του Omniglot Dataset είναι χωρισμένη σε ένα σύνολο εκπαίδευσης που αποτελείται από 30 γλώσσες και ένα σύνολο επαλήθευσης που περιέχει τις υπόλοιπες 20. Επίσης, πρόσφατα έχει ενσωματωθεί και η πληροφορία για τις μολυβιές (strokes) που απαρτίζουν τον κάθε χαρακτήρα, οι οποίες δίνονται ως ακολουθίες σημείων. Στο σύνολο δεδομένων αυτό, πέρα από την εργασία της κατηγοριοποίησης, προτάθηκαν επίσης η παραγωγή νέων προτύπων από έναν χαρακτήρα και νέων χαρακτήρων (concepts) είτε κάτω από τον περιορισμό ενός αλφάβητου είτε χωρίς περιορισμό. Τέλος, στις εργασίες έχει προστεθεί και η ανάλυση (parsing) των χαρακτήρων, δηλαδή η εύρεση της σειράς των μολυβιών που τους απαρτίζουν (Σχήμα 1.2).

- Mini-ImageNet Dataset: Το ImageNet σε συνδυασμό με το ImageNet Large Scale Visual Recognition Challenge (ILSVRC) αποτελεί ένα από τα πιο γνωστά στο χώρο της Βαθιάς Μηχανικής Μάθησης. Το Mini-Imagenet προκύπτει από το ILSVRC-12 σύνολο δεδομένων κρατώντας 60.000 έγχρωμες εικόνες, μεγέθους 84×84 χωρισμένες σε 100 κλάσεις με 600 παραδείγματα από την κάθε μια. Στη βιβλιογραφία έχουν προταθεί διαφορετικές διατμήσεις για τις 100 κλάσεις δυσχεραίνοντας τη σύγκριση των αποτελεσμάτων. Μια από τις πιο δημοφιλείς διατμήσεις, εισήγαγαν οι Ravi και Laurochelle [18], όπου χρησιμοποιούνται 64 κλάσεις για την εκπαίδευση, 16 για την επικύρωση (valiation) και 20 για την αξιολόγηση.

- CUB 200 Dataset: Το Caltech-UCSD Birds (CUB) Dataset περιλαμβάνει 11,788 εικόνες από 200 είδη πουλιών. Από αυτές οι 100 κλάσεις χρησιμοποιούνται για την εκπαίδευση, 50 για την επικύρωση και οι υπόλοιπες 50 για την αξιολόγηση.

Σχήμα 2.6: Εικόνες από το CUB200 Dataset [9].

2.4 Meta-learning

Το πρόβλημα του Few-shot Learning μπορεί να εκφραστεί μαθηματικά ως εξής: Έστω $D = \bigcup_i \{(x_i, y_i)\}$ ένα σύνολο από παραδείγματα με τις ετικέτες τους και έστω ότι έχουμε ένα κατηγοριοποιητή (classifier) f_θ με παράμετρους θ, τότε οι βέλτιστες παράμετροι πρέπει να μεγιστοποιούν την πιθανότητα των σωστών κλάσεων για τα παραδείγματα στο D:

$$\theta^* = argmax_\theta \mathbb{E}_{(x,y)\in D}[P(y|x)] \tag{2.2}$$

Μια πολύ βασική αρχή για το Few-show Learning είναι η Μετα-Μάθηση (Meta-Learning), γνωστή και ως "μαθαίνοντας να μαθαίνει" (learning to learn). Σύμφωνα με αυτή την αρχή, ένα μοντέλο Μετα-Μάθησης (meta-learning model) θα πρέπει να εκπαιδεύεται σε μια ποικιλία από εργασίες μάθησης (learning task) και να βελτιστοποιείται για την καλύτερη απόδοση στην κατανομή των εργασιών αυτών. Αυτό σημαίνει:

$$\theta* = argmin_\theta \mathbb{E}_{D\sim P(D)}[\mathcal{L}_\theta(D)] \tag{2.3}$$

Η διαφορά με τη βελτιστοποίηση (optimization) των παραδοσιακών αρχιτεκτονικών είναι ότι οι εργασίες είναι πιο γενικές είτε από απλά παραδείγματα είτε από συστάδες από ένα σύνολο δεδομένων. Ένα τέτοιο παράδειγμα είναι οι εργασίες κατηγοριοποίησης, όπου η κατηγοριοποίηση ενός συνόλου δεδομένων μετράει ως μια εργασία. Τα επεισόδια του Few-shot Learning, που περιγράφηκαν στο προηγούμενο κεφάλαιο, αποτελούν εργασίες κατηγοριοποίησης, όπου κάθε επεισόδιο έχει ένα σύνολο εκπαίδευσης και ένα σύνολο αξιολόγησης. Έτσι, τα μοντέλα που εκπαιδεύονται σε επεισόδια, αποτελούν παραδείγματα Μετα-Μάθησης.

2.5 Διαφορετικές προσεγγίσεις στο Few-shot Learning

Για το πρόβλημα του Few-shot Learning έχουν προταθεί πολλά και διαφορετικά μοντέλα, τα οποία για λόγους ταξινόμησης μπορούν να χωριστούν σε 3 μεγάλες κατηγορίες:

- Βασισμένα σε μετρικές (Metric-Based): τα μοντέλα αυτά μαθαίνουν έναν εξαγωγέα χαρακτηριστικών (feature extractor) και τον συνδυάζουν με μια μέθοδο για την κατηγοριοποίηση στο στάδιο της αξιολόγησης, η οποία συνήθως υπολογίζει αποστάσεις ή ομοιότητες μεταξύ των παραδειγμάτων.

- Βασισμένα σε βελτιστοποίηση (Optimization-Based): στόχος τους είναι να βελτιστοποιήσουν ένα δίκτυο, έτσι ώστε να μπορέσει να μάθει γρήγορα. Ένα παραδαδοσιακό νευρωνικό δίκτυο το οποίο μαθαίνει μέσω της οπισθοδιάδοσης της κλίσης δεν μπορεί να μάθει να γενικεύει αποτελεσματικά από ένα Σύνολο Υποστήριξης, το οποιο διαθέτει ένα πρότυπο από την κάθε κλάση. Ωστόσο, αλλάζοντας τον αλγόριθμο βελτιστοποίησης (optimization algorithm) είναι δυνατό να επιτευχθούν αρχιτεκτονικές που μαθαίνουν με μόνο ένα παράδειγμα. Στην βιβλιογραφία αναφέρονται και ως αρχιτεκτονικές βασισμένες στην κλίση (Gradient-Based), καθώς κάνουν χρήση της κλίσης της συνάρτησης λάθους, τροποποιώντας ή γενικεύοντας την οπισθοδιάδοση.

- Βασισμένα σε μνήμη (Memory-Based): το κύριο χαρακτηριστικό τους είναι η χρήση κάποιας μορφής μνήμης, την οποία χρησιμοποιούν για να αποθηκεύουν εμπειρία (experience), η οποία αποκτήθηκε από προγούμενες εργασίες.

Στη συνέχεια παρουσιάζονται και αναλύονται οι πιο δημοφιλείς μέθοδοι από την κάθε κατηγορία.

Κεφάλαιο 3

Αρχιτεκτονικές βασισμένες σε Μετρικές

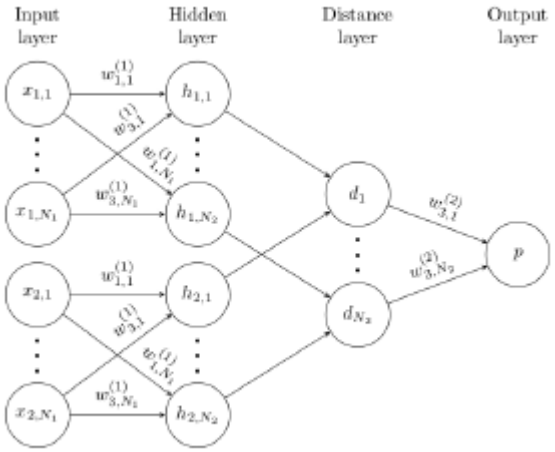
Η πρώτη κατηγορία αρχιτεκτονικών που αναπτύχθηκε για την επίλυση του Few-shot Learning είναι οι αρχιτεκτονικές βασισμένες σε Μετρικές. Οι αρχιτεκτονικές αυτές συσχετίζονται σημαντικά με τα παραδοσιακά συνελικτικά δίκτυα για κατηγοριοποίηση, τα οποία χρησιμοποιούν πλήρως συνδεδεμένα επίπεδα στα βαθύτερα επίπεδα. Συγκεκριμένα, τα συνελικτικά επίπεδα υλοποιούν τον εξαγωγέα χαρακτηριστικών και τα πλήρως συνδεδεμένα ένα κατηγοριοποιητή f_θ που ταξινομεί τα παραδείγματα στις κλάσεις εκπαίδευσης. Ωστόσο, επειδή οι κλάσεις αξιολόγησης είναι διαφορετικές από τις κλάσεις εκπαίδευσης στην περίπτωση του Few-shot Learning, ο ταξινομητής δεν έχει νόημα. Έτσι οι αρχιτεκτονικές επικεντρώνονται στην αντικατάσταση του ταξινομητή με μια μέθοδο, που μπορεί να έχει επιπλέον παραμέτρους (παραμετρική) ή και όχι (μη παραμετρική), η οποία συνήθως υπολογίζει αποστάσεις μεταξύ των παραδειγμάτων. Οι αρχιτεκτονικές συνεχίζουν να χρησιμοποιούν την οπισθοδιάδοση για την εκπαίδευση των παραμέτρων τους, χρησιμοποιώντας ως συνάρτηση λάθους την Cross-entropy Loss η οποία ορίζεται ως:

$$L_{T_i}(f_\theta) = \sum_{x_j, y_j \sim T_i} y_j \log f_\theta(x_j) + (1 - y_j) \log(1 - f_\theta(x_j)) \tag{3.1}$$

Οι αρχιτεκτονικές που θα μελετηθούν σε αυτό το κεφάλαιο είναι οι εξής: Σιαμαία δίκτυα, Δίκτυα Ταιριάσματος και Πρωτότυπα Δίκτυα.

3.1 Σιαμαία νευρωνικά δίκτυα

Τα Σιαμαία νευρωνικά δίκτυα (Siamese Networks) προτάθηκαν από τον LeCun et al. [3] για την εργασία της ταυτοποίησης υπογραφών. Το δίκτυο αυτό αποτελείται από δύο όμοια νευρωνικά δίκτυα, τα οποία λαμβάνουν από ένα παράδειγμα το καθένα και στην συνέχεια ακολουθούνται από ένα επίπεδο που υπολογίζει και σταθμίζει την απόσταση μεταξύ των χαρακτηριστικών τους, η οποία τροφοδοτείται σε μια λογιστική συνάρτηση.



Σχήμα 3.1: Η αρχιτεκτονική του Σιαμαίου δικτύου [10].

Με αυτήν την υλοποίηση το Σιαμαίο νευρωνικό δίκτυο επιλύει την εργασία της επαλήθευσης (verification), δηλαδή αποφαίνεται αν δύο υπογραφές είναι ίδιες ή όχι, υπολογίζοντας τον βαθμό ομοιότητας. Εφόσον η έξοδος προκύπτει από τη λογιστική συνάρτηση, η οποία έχει πεδίο τιμών ένα διάστημα μεταξύ 0 και 1, ο βαθμός ομοιότητας είναι επίσης μεταξύ 0 και 1. Μέσω ενός κατωφλίου (threshold), το οποίο αποτελεί υπερπαράμετρο (hyperparameter), το δίκτυο αποφασίζει για την ομοιότητα ή μη των υπογραφών. Ωστόσο, σε ένα επεισόδιο του Few-shot Learning δεν έχουμε μόνο δύο κλάσεις (είναι όμοια ή όχι), αλλά K. Επίσης διαθέτουμε N παραδείγματα από την κάθε κλάση στο Σύνολο Υποστήριξης. Οι Koch et al. [10] πρότειναν ένα τρόπο επέκτασης των Σιαμαίων δικτύων για το πλαίσιο του One-shot Learning ως εξής: έστω ότι δίνεται μια εικόνα αξιολόγησης x η οποία πρέπει να καταταχθεί σε K κλάσεις, από τις οποίες έχουμε τα πρότυπα x_k με $k = 1, 2, ..., K$. Τότε τροφοδοτούμε στο Σιαμαίο δίκτυο όλα τα ζευγάρια x, x_k και προβλέπουμε:

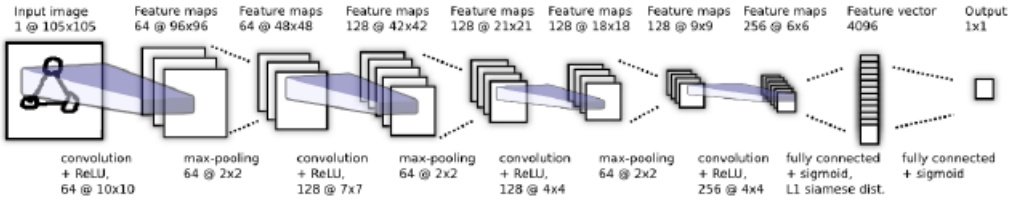
$$C* = argmax_k P_k \tag{3.2}$$

όπου P_k ο βαθμός ομοιότητας που υπολογίζει το δίκτυο για το ζεύγος. Στην περίπτωση του Few-shot Learning, που έχουμε παραπάνω παραδείγματα από την κάθε κλάση ($N > 1$), πάλι παίρνοντας όλα τα δυνατά ζεύγη, προβλέπεται η κλάση του παραδείγματος με τον μεγαλύτερο βαθμό ομοιότητας. Γίνεται άμεσα αντιληπτό ότι όσο μεγαλύτερο είναι το επεισόδιο τόσο πιο αργή είναι η αξιολόγηση, καθώς μεγαλώνει ο αριθμός των δυνατών ζευγών. Μια σημαντική συνεισφορά των Koch et al. είναι ότι άλλαξαν την συνάρτηση ενέργειας μεταξύ των χαρακτηριστικών των παραδειγμάτων που έχει προταθεί από τους LeCun et al. [3] στην $\mathcal{L}_1$ απόσταση, η οποία σταθμίζεται από το τελευταίο επίπεδο (Σχήμα 3.1). Άμεσα προκύπτει ότι το δίκτυο είναι συμμετρικό καθώς η $\mathcal{L}_1$ είναι συμμετρική και τα συνελικτικά νευρωνικά δίκτυα είναι όμοια.

Ως συνάρτηση λάθους χρησιμοποιήθηκε η Binary cross-entropy loss και το δίκτυο εκπαιδεύτηκε με οπισθοδιάδοση σε συστάδες από ζεύγη. Τα ζεύγη επιλέχθηκαν τυχαία, με πιθανότητα 50% τα παραδείγματα να ανήκουν στην ίδια κλάση, όπου όλες οι κλάσεις είχαν την ίδια πιθανότητα να επιλεγούν. Επίσης για καλύτερα αποτελέσματα, επαύξησαν τα δεδομένα (Data augmentation) με τυχαίους μικρούς αφινικούς μετασχηματισμούς (Σχήμα 3.2).

Σχήμα 3.2: Τυχαίοι μικροί αφινικοί μετασχηματισμοί σε δεδομένα του Omniglot [10].

Η αρχιτεκτονική του δικτύου που χρησιμοποίησαν οι Koch et all [10] φαίνεται στο σχήμα 3.3. Όπως αναφέραμε και στην εισαγωγή, για λόγους δίκαιης σύγκρισης χρησιμοποιήσαμε το ίδιο συνελικτικό νευρωνικό δίκτυο (ίδια αρχιτεκτονική), που υλοποιεί τον εξαγωγέα χαρακτηριστικών, με αυτό που χρησιμοποιούν οι Vinyals et al. στα Δίκτυα Ταιριάσματος [25], το οποίο ονομάζεται 4CONV και παρουσιάζεται στο κεφάλαιο 6. Να σημειωθεί ότι στην αρχιτεκτονική τους, οι Koch et al. χρησιμοποιούν μεγαλύτερης διάστασης εξαγωγέα χαρακτηριστικών και αποσύνθεση βαρών (weight decay) που συνεισφέρει στην αποφυγή της υπερεκπαίδευσης. Η υλοποίηση των Σιαμαίων Δικτύων σε PyTorch είναι αρκετά εύκολη καθώς παρουσιάζει πολλά κοινά χαρακτηριστικά με τα παραδοσιακά συνελικτικά δίκτυα. Ο χρόνος εκπαίδευσης τους είναι αυξημένος τόσο σε σχέση με τα παραδοσιακά συνελικτικά δίκτυα, όσο και ως προς τις επόμενες αρχιτεκτονικές που μελετώνται σε αυτό το κεφάλαιο. Αυτό οφείλεται στο ότι πρέπει να δουν πολλά ζεύγη για να αρχίσουν να μαθαίνουν ομοιότητες και διαφορές, σε αντίθεση με τις επόμενες τεχνικές που σε κάθε επανάληψη μελετούν πολλαπλές κλάσεις ταυτόχρονα.



Σχήμα 3.3: Η αρχιτεκτονική που χρησιμοποιήθηκε για τα Σιαμαία Δίκτυα στο Omniglot [10].

3.2 Δίκτυα Ταιριάσματος

Τα Δίκτυα Ταιριάσματος (Matching Networks) αναπτύχθηκαν ειδικά για το Few-Shot Learning από τους Vinyals et al. [25]. Είναι η πρώτη αρχιτεκτονική η οποία εκπαιδεύεται σε επεισόδια από το σύνολο εκπαίδευσης, οπότε η διαδικασία εκπαίδευσης και αξιολόγησης ταυτίζεται. Όπως αναφέρθηκε στο προηγούμενο κεφάλαιο, οι αρχιτεκτονικές των οποίων οι παράμετροι εκπαιδεύονται σε εργασίες, όπως οι γνωσιακές εργασίες των επεισοδίων, κάνουν χρήση της αρχής της Μετα-Μάθησης. Οπότε, κατά την εκπαίδευση, σε κάθε επεισόδιο έχουμε ένα Σύνολο Υποστήριξης $S = \bigcup_{i=1}^{K} \{(x_i, y_i)\}$ και ένα Σύνολο Ερωτημάτων $Q = \bigcup_{i=1}^{K} \{(x_i, y_i)\}$. Έστω $\hat{x}$ ένα παράδειγμα από το σύνολο ερωτημάτων. Ο σκοπός της αρχιτεκτονικής είναι από το Σύνολο S να παράξει έναν κατηγοριοποιητή c_S ο οποίος για κάθε δείγμα $\hat{x}$ θα δώσει ως έξοδο μια κατανομή πιθανοτήτων $\hat{y}$ για τις κλάσεις του επεισοδίου. Πιο αναλυτικά, το δίκτυο υπολογίζει το $P(\hat{y}|\hat{x}, S)$, όπου το P παραμετρικοποιείται από ένα νευρωνικό δίκτυο. Δοθέντος ενός καινούργιου Συνόλου Εκπαίδευσης S', χρησιμοποιείται το παραμετρικό δίκτυο P για να παράξει τις νέες προβλέψεις, δηλαδή $P(\hat{y}'|\hat{x}', S')$ όπου $\hat{x}'$ είναι παράδειγμα από το νέο Σύνολο

Ερωτημάτων Q'. Η πρόβλεψη y' του δικτύου μπορεί να εκφραστεί μαθηματικά ως:

$$\hat{y} = \sum_i (\alpha(\hat{x}, x_i) y_i)$$

όπου το α είναι ένας μηχανισμός προσοχής (attention mechanism).

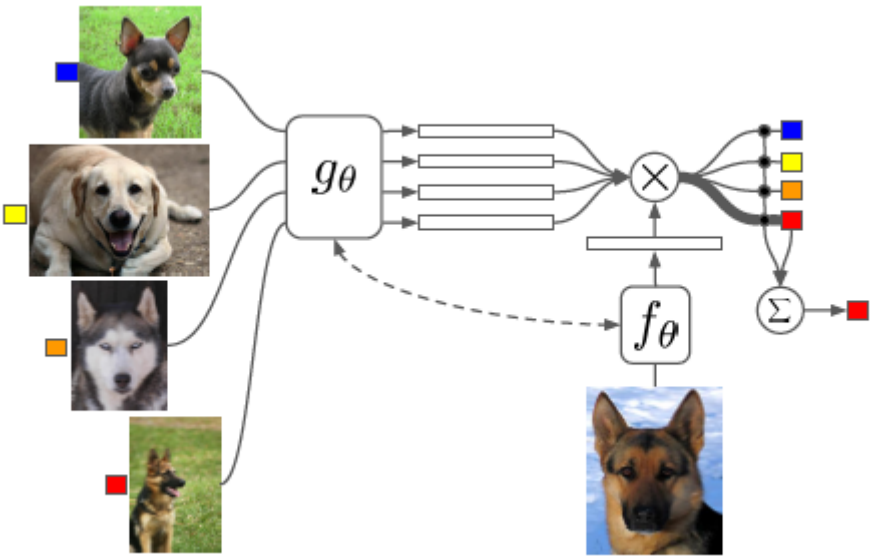
Παρατηρώντας την παραπάνω μορφή της εξόδου $\hat{y}$ βλέπουμε ότι το πλαίσιο αυτό αποτελεί μια γενίκευση κάποιων από τις πιο γνωστώς τεχνικές μηχανικής μάθησης (Machine Learning). Συγκεκριμένα, χρησιμοποιώντας ένα μηχανισμό μάθησης α που είναι μηδεν για τα b μακρυνότερα παραδείγματα x_i από το $\hat{x}$ και μια κατάλληλη σταθερά για τα υπόλοιπα $k - b$, ο μηχανισμός αυτός είναι ισοδύναμος με τη μέθοδο $'k - b'$ Πλησιέστεροι Γείτονες (Nearest Neighbours-NN) [25]. Αντίστοιχα, αν ο α είναι ένας πυρήνας (kernel), τότε ο μηχανισμός γίνεται ισοδύναμος με έναν Kernel Desity Estimator (KDE). Μια άλλη οπτική είναι να θεωρήσουμε τα y_i ως μνήμες που αντιστοιχούν στα x_i, οπότε με τη μέθοδο αυτή ανακτούμε την πληροφορία του σχετικότερου y_i. Γίνεται άμεσα αντιληπτό ότι το πλαίσιο που εισάγουν οι Vinyals et al. είναι γενικό και αρκετά ευέλικτο. Ένας απλός μηχανισμός εστίασης που όμως αποδίδει καλά στο Omniglot Dataset, είναι η χρήση του softmax στις συνημιτονικές αποστάσεις (cosine distance) μεταξύ των παραδειγμάτων, δηλαδή:

$$\alpha(\hat{x}, x_i) = \frac{\exp(-c(f(\hat{x}), g(x_i))}{\sum_{j=1}^{k} \exp(-c(f(\hat{x}), g(x_j))))} \tag{3.3}$$

όπου η συνημιτονική απόσταση ορίζεται ως:

$$c(x, y) = 1 - \frac{xy}{\| x \| \| y \|} \tag{3.4}$$

Οι συναρτήσεις f, g είναι εξαγωγείς χαρακτηριστικών που υλοποιούνται με νευρωνικά δίκτυα. Στην περίπτωσή μας, χρησιμοποιήσαμε το ίδιο νευρωνικό για το Σύνολο Υποστήριξης και το Σύνολο Ερωτημάτων, δηλαδή $f = g$. Στο επιστημονικό άρθρο τους [25], οι συγγραφείς προτείνουν και μια κατηγορία μηχανισμών που ονομάζουν Full Context Embeddings (FCE), των οποίων οι συναρτήσεις f, g επιτελούν πάλι την εξαγωγή χαρακτηριστικών, ωστόσο λαμβάνουν υπόψη τους ολόκληρο το Σύνολο Υποστήριξης, δηλαδή για κάθε δείγμα x_i το διάνυσμα των χαρακτηριστικών δίνεται απο την $g(x_i, S)$ και το ίδιο συμβαίνει για την $f(\hat{x}, S)$. Αν και στο Omniglot τα FCE δεν έδειξαν κάποια βελτίωση στα αποτελέσματα, στο miniImageNet, υλοποιώντας τις f, g με δίκτυα LSTM τα οποία λαμβάνουν υπόψη τους και Σύνολο Υποστήριξης, υπήρξε μια μικρή βελτίωση 2-3%. Για τα πειράματά τους στο Omniglot, οι συγγραφείς χρησιμοποίησαν το δίκτυο 4CONV ως εξαγωγέα χαρακτηριστικών, το οποίο στη συνέχεια χρησιμοποιήθηκε και από πολλές άλλες αρχιτεκτονικές. Τα Δίκτυα Ταιριάσματος, χωρίς τον μηχανισμό του FCE, δεν εισάγουν επιπλέον παραμέτρους εκμάθησης πέρα από τον εξαγωγέα χαρακτηριστικών. Επιπλεόν, λόγω του γεγονότος ότι η εκπαίδευση με την αξιολόγηση ταιριάζουν, χρειάστηκαν λίγα επεισόδια για να αρχίσουν να μαθαίνουν χρήσιμα χαρακτηριστικά για το πρόβλημα. Η υλοποίηση τους είναι επίσης εύκολη, με την προσοχή κυρίως στην επέκταση της εκπαιδευσής τους σε επεισόδια.

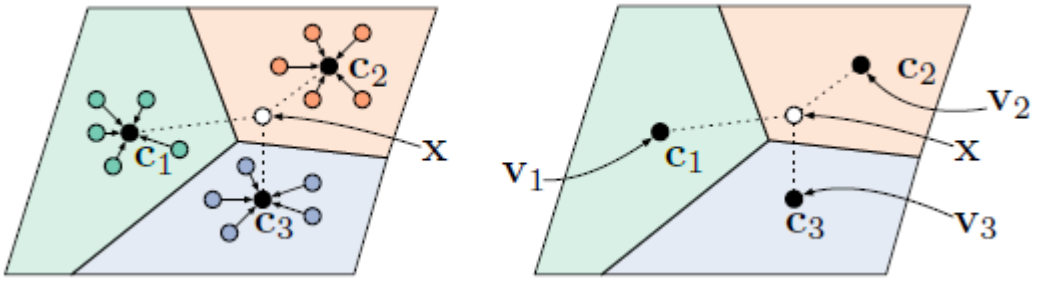


Σχήμα 3.4: Η αρχιτεκτονική του Δικτύου Ταιριάσματος [25].

3.3 Πρωτότυπα Δίκτυα

Τα Πρωτότυπα Δίκτυα (Prototypical Networks) [23] έπονται των Δικτύων Ταιριάσματος και εμπνεύστηκαν από τη δουλειά του Vinyals et al. [25]. Όπως είδαμε, στα Δίκτυα Ταιριάσματος έχει εισαχθεί η εκπαίδευση σε επεισόδια και έχουν προταθεί μηχανισμοί εστίασης. Τα Πρωτότυπα δίκτυα επεκτείνουν τις παραπάνω ιδέες για την περίπτωση που το $N > 1$, δηλαδή σε Σύνολο Υποστήριξης όπου δίνεται παραπάνω από ένα παράδειγμα για την κάθε κλάση. Στην περίπτωση που δίνεται ακριβώς ένα, δηλαδή στο One-shot Learning, οι αρχιτεκτονικές ταυτίζονται, με μόνη διαφορά ότι οι Snell et al. χρησιμοποίησαν την τετραγωνισμένη Ευκλείδια απόσταση $\mathcal{L}_2^2$ αντί της Συνημιτονικής. Η κεντρική ιδέα, η οποία θα αναλυθεί στη συνέχεια, είναι ότι υπάρχει ένας εξαγωγέας χαρακτηριστικών, στον οποίο τα δείγματα της ίδιας κλάσης μαζεύονται γύρω από μια πρωτότυπη αναπαράσταση της κλάσης (prototype). Για να μάθει το παραμετρικό δίκτυο αυτόν τον εξαγωγέα, το δίκτυο υπολογίζει για κάθε κλάση στο Σύνολο Υποστηρίξης την πρωτότυπη αναπαράσταση ως τον μέσο όρο των παραδειγμάτων της κλάσης στο χώρο του εξαγωγέα. Οπότε αν $f_\theta : \mathbb{R}^D \to \mathbb{R}^M$ ένας εξαγωγέας χαρακτηριστικών, όπου D και M οι διάστασεις των χώρων εισόδου και εξόδου αντίστοιχα, τότε για κάθε κλάση k στο Σύνολο Υποστήριξης $S = \bigcup_{i=1}^{N} \{(x_i, y_i)\}$ το πρωτότυπο της κλάσης c_k ορίζεται ως:

$$c_k = \frac{1}{|S_k|} \sum_{(x_i, y_i) \in S_k} f_\theta(x_i) \tag{3.5}$$



Σχήμα 3.5: Τα Πρωτότυπα Δίκτυα στο πλαίσιο του Few-shot (αριστερά) και Zero-shot (δεξιά) [23].

Ο μηχανισμός προσοχής που προτείνεται είναι η χρήση της softmax στις αποστάσεις μεταξύ των παραδειγμάτων του Συνόλου Ερωτημάτων και των πρωτότυπων c_k στον χώρο εξόδου M, δηλαδή:

$$P_\theta(y = k|x) = \frac{\exp(-d(f_\theta(x), c_k))}{\sum_{k'}(\exp(-d(f_\theta(x), c_{k'})))} \tag{3.6}$$

Ο Snell et al. αποδεικνύουν ότι χρησιμοποιώντας μια οικογένεια αποστάσεων γνωστών ως regular Bregman divergences, οι οποίες ορίζονται ως:

$$d_\theta(z, z') = \theta(z) - \theta(z') - (z - z')^T \nabla\theta(z') \tag{3.7}$$

στην οποία ανήκει και η τετραγωνισμένη Ευκλείδια απόσταση, τα Πρωτότυπα δίκτυα είναι ισοδύναμα με τη μέθοδο Mixture Density Estimation (MDE) της μηχανικής μάθησης, χρησιμοποιώντας μια εκθετική οικογένεια κατανομών, δηλαδή η πρόβλεψη του δικτύου μπορεί να εκφραστεί ως:

$$p(y = k|z) = \frac{\pi_k exp(-d_\theta(z, \mu(\phi_k)))}{\sum_{k'} \pi_{k'} exp(-d_\theta(z, \mu(\phi_k)))} \tag{3.8}$$

όπου $z = f_\theta(x)$ και $\mu(\phi_k) = c_k$. Οι συγγραφείς επίσης αποδεικνύουν ότι χρησιμοποιώντας τετραγωνισμένη Ευκλείδια απόσταση, ο κατηγοριοποιητής που προκύπτει είναι ισοδύναμος με ένα γραμμικό κατηγοριοποιητή, όμως αυτό δεν περιορίζει την απόδοση του μοντέλου καθώς η μη γραμμικότητα μπορεί να μαθευτεί από τον εξαγωγέα χαρακτηριστικών που χρησιμοποιείται.

Για την εκπαίδευση στο Omniglot χρησιμοποιήθηκε το δίκτυο 4CONV, όπως στα Δίκτυα Ταιριάσματος. Μια σημαντική παρατήρηση είναι ότι κατά την εκπαίδευση οι Snell et al. δοκίμασαν να αυξήσουν τον αριθμό των κλάσεων K, καθιστώντας το επεισόδιο εκπαίδευσης δυσκολότερο από το επεισόδιο αξιολόγησης. Χρησιμοποιώντας αυτήν την πρακτική παρατήρησαν βελτίωση στην απόδοση του δικτύου. Συγκεκριμένα βρήκαν ότι με $K = 60$ κλάσεις κατά την εκπαίδευση το δίκτυο πετυχαίνει καλύτερα αποτελέσματα. Αντιθέτως, αλλάζοντας τον αριθμό των παραδειγμάτων N από την κάθε κλάση κατά την εκπαίδευση δεν παρατηρήθηκε κάποιο πλεονέκτημα. Όπως και τα Δίκτυα Ταιριάσματος, τα Πρωτότυπα Δίκτυα δεν εισάγουν επιπλέον παραμέτρους. Η διαδικασία εκπαίδευσης και αξιολόγης είναι ακόμα πιο γρήγορη στην περίπτωση που το $N > 1$ σε σχέση με τις άλλες αρχιτεκτονικές, καθώς οι αποστάσεις λαμβάνονται μόνο μεταξύ των πρωτότυπων και του Συνόλου Ερωτημάτων και όχι για όλα τα δυνατά ζεύγη Συνόλου Ερωτημάτων και Συνόλου Υποστήριξης. Η υλοποίηση διαφέρει ως προς τον υπολογισμό των πρωτοτύπων σε σχέση με την προηγούμενη μέθοδο, τα οποία προκύπτουν ως ο μέσος των παραδειγμάτων, οπότε δεν παρουσιάζει ιδιαίτερη δυσκολία.

Algorithm 1 Training episode loss computation for prototypical networks. N is the number of examples in the training set, K is the number of classes in the training set, $N_C \leq K$ is the number of classes per episode, N_S is the number of support examples per class, N_Q is the number of query examples per class. RANDOMSAMPLE(S, N) denotes a set of N elements chosen uniformly at random from set S, without replacement.

Input: Training set $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, where each $y_i \in \{1, \ldots, K\}$. $\mathcal{D}_k$ denotes the subset of $\mathcal{D}$ containing all elements $(\mathbf{x}_i, y_i)$ such that $y_i = k$.
Output: The loss J for a randomly generated training episode.

$V \leftarrow$ RANDOMSAMPLE$(\{1, \ldots, K\}, N_C)$ ▷ Select class indices for episode
for k in $\{1, \ldots, N_C\}$ **do**
  $S_k \leftarrow$ RANDOMSAMPLE$(\mathcal{D}_{V_k}, N_S)$ ▷ Select support examples
  $Q_k \leftarrow$ RANDOMSAMPLE$(\mathcal{D}_{V_k} \setminus S_k, N_Q)$ ▷ Select query examples
  $\mathbf{c}_k \leftarrow \frac{1}{N_C} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_{\phi}(\mathbf{x}_i)$ ▷ Compute prototype from support examples
end for
$J \leftarrow 0$ ▷ Initialize loss
for k in $\{1, \ldots, N_C\}$ **do**
  for $(\mathbf{x}, y)$ in Q_k **do**
    $J \leftarrow J + \frac{1}{N_C N_Q} \left[d(f_{\phi}(\mathbf{x}), \mathbf{c}_k)) + \log \sum_{k'} \exp(-d(f_{\phi}(\mathbf{x}), \mathbf{c}_k)) \right]$ ▷ Update loss
  end for
end for

Σχήμα 3.6: Ο αλγόριθμος εκπαίδευσης των Πρωτότυπων Δικτύων [23].

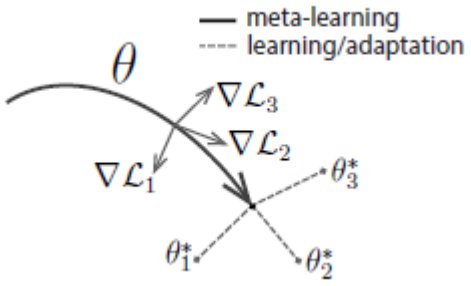
Κεφάλαιο 4

Αλγόριθμοι βασισμένοι σε βελτιστοποίηση για το Few-shot Learning

Είναι η πιο πρόσφατη κατηγορία μεθόδων για το Few-shot Learning, και είναι ιδιαίτερα δημοφιλής συγκεντρώνοντας την προσοχή ενός μεγάλου μέρους της επιστημονικής κοινότητας. Οι αλγόριθμοι αυτοί αντιμετωπίζουν το πρόβλημα του Few-shot Learning από διαφορετική σκοπία σε σχέση με τις προηγούμενες κατηγορίες. Κάποιες από τις λύσεις που προτείνονται είναι εμπνευσμένες από τον κλάδο της Μεταφοράς Μάθησης (Transfer Learning).

4.1 Model-Agnostic Μετα-Μάθηση

Ένας από τους πιο πετυχημένους και παράλληλα απλούς αλγόριθμος βελτιστοποίησης που ανήκει στην κατηγορία αυτή είναι ο Model-Agnostic Meta-Learning (MAML) [4]. Ένα από τα μεγάλα προτερήματά του είναι ότι είναι συμβατός με κάθε μοντέλο που μαθαίνει μέσω της μεθόδου κατάβασης κλίσης (gradient descent). Η κεντρική ιδέα είναι ότι υπάρχουν δύο μοντέλα, ο base-learner και ο meta-learner, ο οποίος εκπαιδεύει τον πρώτο. Τα βάρη του base-learner ενημερώνονται με κατάβαση κλίσης σε γνωστικές εργασίες (learning task) του few-shot προβλήματος, ενώ ο meta-learner εφαρμόζει κατάβαση κλίσης ως προς τα βάρη του base-learner πριν την εφαρμογή της κατάβασης κατά κλίση.



Σχήμα 4.1: Ο αλγόριθμος MAML βρίσκει τις αρχικές παραμέτρους θ που προσαρμόζονται γρήγορα σε νέες εργασίες [4].

Πιο συγκεκριμένα, έχουμε ένα base-model που αντιπροσωπεύεται από μια παραμετρική συνάρτηση f_θ με παραμέτρους θ και ένα task $T_i \sim p(T)$. Μετά την εφαρμογή της κατάβασης

κλίσης έχουμε το νέο διάνυσμα παραμέτρων θ_i':

$$\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{T_i}(f_\theta) \tag{4.1}$$

Για ευκολία θα θεωρήσουμε ότι εκτελούμε μόνο ένα βήμα κατάβασης κλίσης. Ο meta-learner τώρα βελτιστοποιεί με βάση την απόδοση του μοντέλου $f_{\theta'}$ με τις νέες παραμέτρους ως προς τις αρχικές παραμέτρους θ σε εργασίες που δειγματοληπτούνται πάλι από την κατανομή $P(T)$, δηλαδή το meta-objective είναι:

$$\min_\theta \sum_{T_i \sim p(T)} \mathcal{L}_{T_i}(f_{\theta_i'}) = \min_\theta \sum_{T_i \sim p(T)} \mathcal{L}_{T_i}(f_{\theta - \alpha \nabla_\theta \mathcal{L}_{T_i}(f_\theta)}) \tag{4.2}$$

Το meta-optimization υλοποιείται πάλι με SGD και ενημερώνει τις παραμέτρους θ ως εξής:

$$\theta \leftarrow \theta - \beta \nabla_\theta \sum_{T_i \sim p(T)} \mathcal{L}_{T_i}(f_{\theta_i'}) \tag{4.3}$$

Το meta-update του αλγόριθμου MAML περιλαμβάνει gradient μέσα από gradient, ή αλλιώς 2ου βαθμού παραγώγους. Αυτό έχει ως αποτέλεσμα να αυξάνει το υπολογιστικό κοστός σημαντικά, και γι αυτό έχουν προταθεί διάφορες τεχνικές 1ου βαθμού προσέγγισης (approximation) για να επισπεύσουν τον αλγόριθμο. Οι συγγραφείς [4] υλοποίησαν το MAML στο MiniImagenet, αγνοώντας τις δεύτερες παραγώγους, υπολογίζοντας την κλίση ως προς θ' στο meta-update, το οποίο ονόμασαν FOMAML (First Order MAML). Συγκεκριμένα, ο αλγόριθμος MAML βελτιστοποιεί το:

$$\min_\theta \mathbb{E}_{T \sim p(T)}[\mathcal{L}_T(\mathbb{U}_T^k(\theta))] \tag{4.4}$$

όπου $\mathbb{U}_T^k$ η διαδικασία κατά την οποία λαμβάνονται k δείγματα από την εργασία T και ανανεώνεται το θ. Όπως αναφέραμε, το επεισόδιο αποτελείται απο το Σύνολο Υποστηρίξης και το Σύνολο Ερωτημάτων, οπότε η βελτιστοποίηση μπορεί να ξαναγραφτεί:

$$\min_\theta \mathbb{E}_{T \sim p(T)}[\mathcal{L}_{T,Q}(\mathbb{U}_{T,S}(\theta))] \tag{4.5}$$

Έτσι τελικά το MAML μέσω της μεθόδου κλίσης υπολογίζει:

$$g_{MAML} = \mathcal{L}_{T,Q}(U_{T,S}(\theta)) = \mathbb{U}'_{T,S}(\theta) \mathcal{L}'_{T,Q}(\tilde{\theta}) \tag{4.6}$$

όπου $\tilde{\theta} = \mathbb{U}_{T,S}(\theta)$ και $\mathbb{U}'_{T,S}$ ο Ιακωβιανός πίνακας της ανανέωσης $\mathbb{U}_{T,S}$, ενώ το FOMAML θεωρεί τον $\mathbb{U}'_{T,S}$ μοναδιαίο, οπότε υπολογίζει:

$$g_{FOMAML} = \mathcal{L}'_{T,Q}(\tilde{\theta}) \tag{4.7}$$

Algorithm 1 Model-Agnostic Meta-Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ with respect to K examples
6: Compute adapted parameters with gradient descent: $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
7: **end for**
8: Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$
9: **end while**

Σχήμα 4.2: Ο αλγόριθμος εκπαίδευσης του MAML [4].

Μια διαφορετική υλοποίηση με 1ου βαθμού παραγώγους μελέτησαν και ανέλυσαν οι Nichol et al., προτείνοντας μια παραλλαγή του MAML που κάνει χρήση πάλι μόνο της 1ης παραγώγου, την οποία ονόμασαν Reptile . Η διαφορά του με τον FOMAML είναι ότι στο τελευταίο βήμα, αντιμετωπίζει το $\tilde{\phi} - \phi$ ως κλίση και το τροφοδοτεί σε κάποιο προσαμοστικό αλγόριθμο όπως ο Adam.

Algorithm 1 Reptile (serial version)

Initialize ϕ, the vector of initial parameters
for iteration $= 1, 2, \ldots$ **do**
Sample task τ, corresponding to loss L_τ on weight vectors $\widetilde{\phi}$
Compute $\widetilde{\phi} = U_\tau^k(\phi)$, denoting k steps of SGD or Adam
Update $\phi \leftarrow \phi + \epsilon(\widetilde{\phi} - \phi)$
end for

Σχήμα 4.3: Ο αλγόριθμος εκπαίδευσης του Reptile [17].

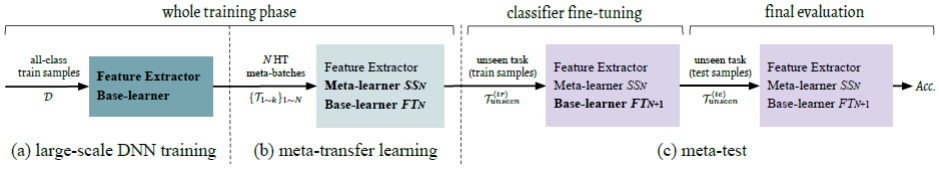
Ο αλγόριθμος MAML και οι παραλλαγές του δεν χρησιμοποιούν επιπλέον παραμέτρους από αυτές του base-learner, ο οποίος αποτελεί τον εξαγωγέα χαρακτηριστικών. Ωστόσο η εκπαίδευση του δικτύου είναι αρκετά πιο αργή από τις προηγούμενες τεχνικές καθώς περιέχει παραγώγους 2ου βαθμού. Για την υλοποίησή του χρειάστηκε να προσομοιωθεί το δίκτυο 4CONV μέσω του nn.functional για να εκμεταλλευτούμε το autograd της PyTorch.

4.2 Meta-Transfer Learning

Η μέθοδος του Meta-Transfer Learning προτάθηκε από τους Sun et al. [24] και είναι ένας συνδυασμός της Μεταφοράς Γνώσης (Transfer Learning) με την Μετα-Μάθηση. Η μέθοδος χρησιμοποιεί ένα βαθύ νευρωνικό δίκτυο ως base learner το οποίο εκπαιδεύεται σε ένα σύνολο δεδομένων μεγάλης κλίμακας, τα βάρη του οποίου στη συνέχεια προσαρμόζονται μέσω Scaling και Shifting (SS), δηλαδή $\alpha X + \beta$. Το γεγονός ότι χρησιμοποιεί ένα προεκπαιδευμένο δίκτυο, αποτελεί μια καλή αρχικοποίηση που προσφέρει ταχύτερη σύγκλιση κατά τη διάρκεια προσαρμογής του, αφού τελικά προσαμόζονται λιγότερες παράμετροι. Ένα άλλο πλεονέκτημα

είναι ότι αποφεύγεται η καταστροφική λήθη (catastrophic forgetting), δηλαδή το δίκτυο να ξεχνάει σημαντική πληροφορία που έχει μάθει με την ανανέωση των βαρών του.

Επίσης, για πιο αποτελεσματική εκπαίδευση προτείνεται μια στρατηγική εκμάθησης, η οποία δίνει προτεραιότητα στα δύσκολα παραδείγματα εκπαίδευσης, την οποία ονομάζουν HT ($HardTask$) meta-batch. Η στρατηγική αυτή χωρίζει την εκπαίδευση σε 2 στάδια, όπου στο 1ο το δίκτυο εκπαιδεύεται σε εργασίες που έχουν παραχθεί τυχαία, ενώ στο 2ο στάδιο δημιουργούνται εργασίες με τις κλάσεις που είχαν τη μικρότερη ακρίβεια στο 1ο στάδιο.



Σχήμα 4.4: Τα 3 στάδια της μεθόδου του MTL [24].

Πιο αναλυτικά, η μέθοδος υλοποιείται ως εξής:

1. Το βαθύ νευρωνικό δίκτυο εκπαιδεύεται στα δεδομένα ενός μεγάλου συνόλου δεδομένων, που αποτελείται από όλες τις κλάσεις ελαχιστοποιώντας μια συνάρτηση λάθους όπως η cross-entropy loss. Έτσι έχουμε έναν εξαγωγέα χαρακτηριστικών (feature extractor) Θ και ένα ταξινομητή θ που εκπαιδεύονται με τη μέθοδο της κατάβασης κλίσης, δηλαδή:

$$[\Theta;\theta] =: [\Theta;\theta] - \alpha\nabla\mathcal{L}_D([\Theta;\theta]) \tag{4.8}$$

   Στη συνέχεια κρατώντας τα βάρη του εξαγωγέα Θ παγωμένα, προστίθενται επιπλεόν παράμετροι σε κάθε νευρώνα, σε κάθε επίπεδο που υλοποιούν το Scale και Shifting που συμβολίζονται με Φ_{S_1}, Φ_{S_2} αντίστοιχα, όπου το Φ_{S_1} αρχικοποιείται με ένα, ενώ το Φ_{S_2} με μηδενικά . Ο ταξινομητής θ που εκπαιδεύτηκε στο προηγούμενο στάδιο για το σύνολο των δεδομένων, δεν λαμβάνεται υπόψη, και αρχικοποιείται ένας καινούργιος ταξινομητής, ο οποίος θα χρησιμοποιείται για την εκπαίδευση σε εργασίες.

2. Ο καινούργιος base-learner που προκύπτει εκπαιδεύεται με meta-batches που περιέχουν εργασίες $T \sim p(T)$, οι οποίες αποτελούνται από ένα κομμάτι εκπαίδευσης T^{tr} και ένα αξιολόγησης T^{te}. Ο ταξινομητής θ εκπαιδεύεται στο κομμάτι εκπαίδευσης με τη μέθοδο της κατάβασης κλίσης με ρυθμό μάθησης β, δηλαδή:

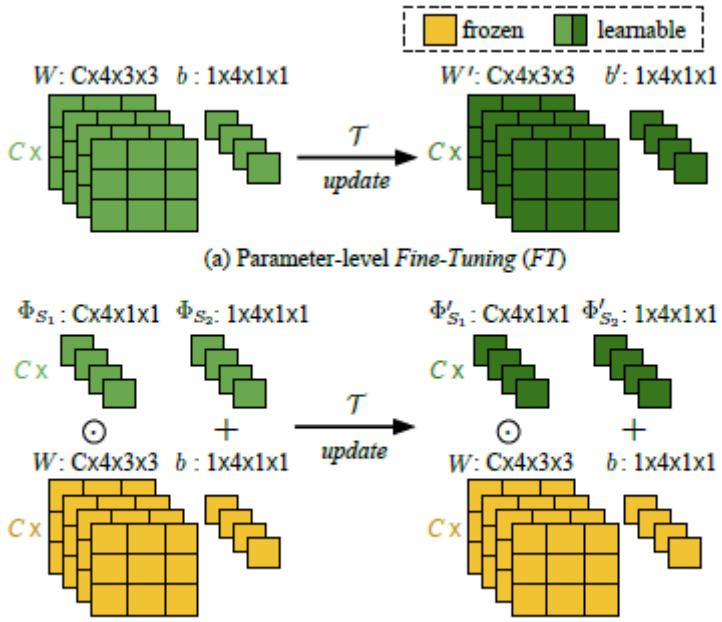
$$\theta' \leftarrow \theta - \beta\nabla_\theta\mathcal{L}_{T^{tr}}([\Theta;\theta], \Phi_{S_{1,2}}) \tag{4.9}$$

3. Αφού εκπαιδευτεί ο νέος ταξινομητής, στο κομμάτι ελέγχου της εργασίας εκπαιδεύονται οι παράμετροι του Scale και Shifting, αλλά και ο ταξινομητής που προέκυψε από το προηγούμενο στάδιο με τη μέθοδο της κατάβασης κλίσης και με ρυθμό γ ως εξής:

$$\Phi_{S_i} =: \Phi_{S_i} - \gamma\nabla_{\Phi_{S_i}}\mathcal{L}_{T^{te}}([\Theta;\theta'], \Phi_{S_{1,2}}) \tag{4.10}$$

$$\theta =: \theta - \gamma\nabla_\theta\mathcal{L}_{T^{te}}([\Theta;\theta'], \Phi_{S_{1,2}}) \tag{4.11}$$

   Μετρώντας την ακρίβεια που επιτυγχάνει το δίκτυο στις κλάσεις του meta-batch, βρίσκονται οι δύσκολες κλάσεις από τις οποίες παράγονται νέες εργασίες και επαναλαμβάνονται η εκπαίδευση του ταξινομητή και των παραμέτρων SS.



Σχήμα 4.5: Η διαφορά μεταξύ fine tunning (FT) και Shift και Scaling (SS). Τα βάρη του αρχικού δικτύου δεν ανανεώνονται με τη μέθοδο SS με αποτέλεσμα το δίκτου να μην ΄ξεχνάει΄ πληροφορία που έχει ήδη μάθει [24].

Λόγω της προεκπαίδευσης που εκτελείται, αυτή η μέθοδος δίνει τη δυνατότητα να εκπαιδευτεί μεγαλύτερο δίκτυο ως εξαγωγές χαρακτηριστικών, με μικρότερη πιθανότητα υπερπροσαρμογής. Οι συγγραφείς χρησιμοποίησαν τη μέθοδο αυτή με το ResNet12 το οποίο αποτελείται από 4 υπολειπόμενα κομμάτια (residual blocks), όπου το κάθε ένα αποτελείται από 3 συνελικτικά επίπεδα με 3×3 πυρήνες και έδειξαν ότι πετυχαίνει μέχρι και 10,8% καλύτερη επίδοση στο miniImageNet, σε σχέση με μεθόδους που χρησιμοποιούν το 4CONV που είναι σημαντικά μικρότερο (θα περιγραφεί στο κεφάλαιο 6).

Κεφάλαιο 5

Αρχιτεκτονικές βασισμένες σε μνήμη

Στο κεφάλαιο αυτό μελετώνται τα νευρωνικά δίκτυα με Επαυξημένη Μνήμη. Αυτή η μέθοδος είναι ίσως η πιο αντιπροσωπευτική της κατηγορίας αυτής, επεκτείνοντας υπάρχουσες αρχιτεκτονικές που χρησιμοποιούν εξωτερικές μνήμες στο Few-shot Learning .

5.1 Νευρωνικά δίκτυα με Επαυξημένη Μνήμη

Τα νευρωνικά δίκτυα με Επαυξημένη Μνήμη (Memory-Augmented Neural Networks-MANNs) βασίζονται σε αρχιτεκτονικές όπως οι Νευρωνικές Μηχανές, οι οποίες αξιοποιούν μια εξωτερική μνήμη για να εγγράφουν και να διαβάζουν αποτελέσματα. Οι αρχιτεκτονικές με εξωτερική μνήμη διαφέρουν από αυτές με εσωτερική, όπως τα LSTM καθώς η εξωτερική μνήμη είναι ευσταθής (stable), επιτρέπει την προσπέλαση κατά στοιχείο (element wise accessible), και το μέγεθος της δεν εξαρτάται από το πλήθος των παραμέτρων του δικτύου.

Οι Νευρωνικές Μηχανές Turing [6] έχουν την δυνατότητα να συμπεράνουν απλούς αλγορίθμους όπως αντιγραφής, ταξινόμησης και συσχετισμένης ανάκλησης (associative recall) μέσω παραδειγμάτων εισόδου και εξόδου. Αποτελούνται από έναν ελεγκτή που αλληλεπιδρά με μια εξωτερική μνήμη μέσω εγγραφών (writes) και αναγνώσεων (reads) οι οποίες αλληλεπιδρούν με ολόκληρη τη μνήμη μέσω ενός μηχανισμού προσοχής (attention mechanism) ο οποίος σταθμίζει τις διαφορετικές περιοχές (memory locations). Συγκεκριμένα, έστω M_t το περιεχόμενο ενός $N \times M$ πίνακα μνήμης την χρονική στιγμή t, όπου N είναι ο αριθμός των θέσεων μνήμης και M είναι το μέγεθος των διανυσμάτων σε κάθε περιοχή. Μια ανάγνωση συνοδεύεται από ένα κανονικοποιημένο διάνυσμα βαρών w_t, όπου το αποτέλεσμα r_t ορίζεται ως:

$$r_t \leftarrow \sum_i w_t(i) M_t(i) \tag{5.1}$$

Ο μηχανισμός εγγραφής ορίζεται σε δύο στάδια, όπου στο πρώτο στάδιο γίνεται η διαγραφή (erase):

$$\tilde{M}_t(i) \leftarrow M_{t-1}(i)[1 - w_t(i)e_t] \tag{5.2}$$

όπου το e_t είναι το διάνυσμα διαγραφής, τα στοιχεία του οποίου βρίσκονται μετάξυ 0 και 1, και στο δεύτερο στάδιο η πρόσθεση (add):

$$M_t(i) \leftarrow \tilde{M}_t(i) + w_t(i)a_t \tag{5.3}$$

όπου το a_t ονομάζεται διάνυσμα προσθήκης. Ένα άμεσο πλεονέκτημα ορίζοντας την ανάγνωση και την εγγραφή σε ολόκληρη τη μνήμη είναι ότι τις καθιστά διαφορίσιμες τόσο ως προς την

μνήμη, όσο και ως προς τα βάρη. Τα βάρη επιτρέπουν τον έλεγχο της περιοχής της μνήμης που τροποποιείται ή διαβάζεται και ρυθμίζονται από τον μηχανισμό προσοχής. Οι Graves et al. [6] πρότειναν 2 μηχανισμούς προσοχής, όπου ο 1ος χρησιμοποείται για προσπέλαση μέσω περιεχομένου (content based addressing) και ο 2ος για προσπέλαση μέσω τοποθεσίας (location based addressing). Η προσπέλαση μέσω περιεχομένου είναι πιο γενική, καθως η μνήμη θα μπορούσε να περιλαμβάνει πληροφορία για τη θέση της πληροφορίας που περιέχει. Έτσι ο μηχανισμός προσοχής για προσπέλαση μέσω περιεχομένων ορίζεται ως:

$$w_t^c(i) \leftarrow \frac{\exp(\beta_t K[k_t, M_t(i)])}{\sum_j \exp(\beta_t K[k_t, M_t(j)])} \tag{5.4}$$

όπου το k_t είναι το διάνυσμα κλειδί (key vector) του οποίου το περιεχόμενο συγκρίνεται με τα στοιχεία της μνήμης $M_t(i)$ και το K είναι ένα μέτρο ομοιότητας (similarity measure). Οι συγγραφείς προτείνουν τη συνημιτονική ομοιότητα (cosine similarity) η οποία ορίζεται ως:

$$K(x, y) = 1 - c(x, y) \tag{5.5}$$

όπου c είναι η Συνημιτονική απόσταση που είδαμε στα Δίκτυα Ταιριάσματος. Το β είναι μια παράμετρος που καθορίζει την ένταση της προσοχής. Για προσπέλαση μέσω τοποθεσίας χρησιμοποιούνται τα ενδιάμεσα βάρη:

$$w_t^g \leftarrow g_t w_t^c + (1 - g_t) w_{t-1} \tag{5.6}$$

όπου g_t είναι μια βαθμωτή πύλη παρεμβολής (scalar interpolation gate) που παίρνει τιμές μεταξύ 0 και 1. Όπως προκύπτει από την παραπάνω εξίσωση για την προσπέλαση για $g_t = 0$ προκύπτει ο μηχανισμός προσπέλασης μέσω περιεχομένου. Αυτά τα βάρη σε συνδυασμό με τις μετατοπίσεις βαρών (shift weighting) s_t που λαμβάνονται από τον ελεγκτή και συνήθως περνούν μέσα από μια συνάρτηση όπως η softmax , χρησιμοποιούνται για να παραχθούν τα τελικά βάρη της προσπέλασης μέσω τοποθεσίας, τα οποία είναι:

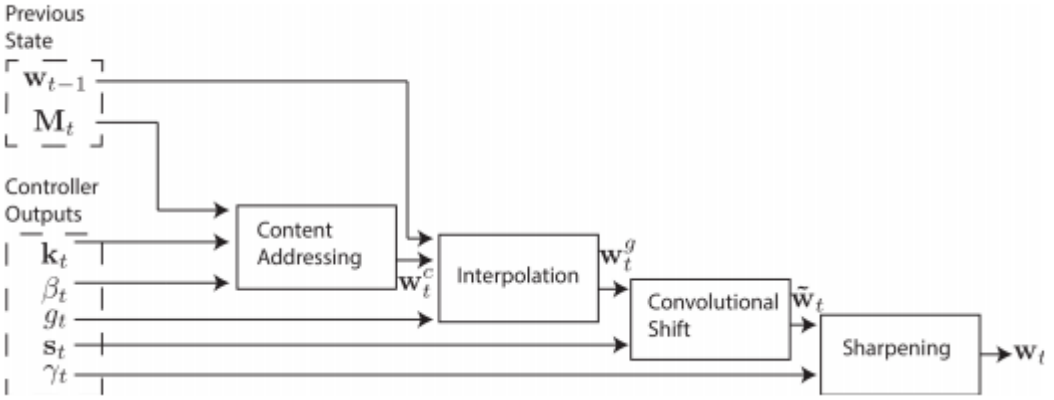
$$\tilde{w}_t(i) \leftarrow \sum_{j=0}^{N-1} w_t^g(j) s_t(i - j) \tag{5.7}$$

Από την παραπάνω εξίσωση γίνεται εμφανές ότι οι μετατοπίσεις βαρών χρησιμοποιούνται για την περιστροφή των w_t^g. Για να οξυνθούν περαιτέρω (sharpening) τα βάρη προσοχής, χρησιμοποιείται η παράμετρος γ_t, οπότε τα τελικά βάρη δίδονται από την εξίσωση:

$$w_t(i) \leftarrow \frac{\tilde{w}_t(i)^{\gamma t}}{\sum_j \tilde{w}_t(j)^{\gamma_t}} \tag{5.8}$$

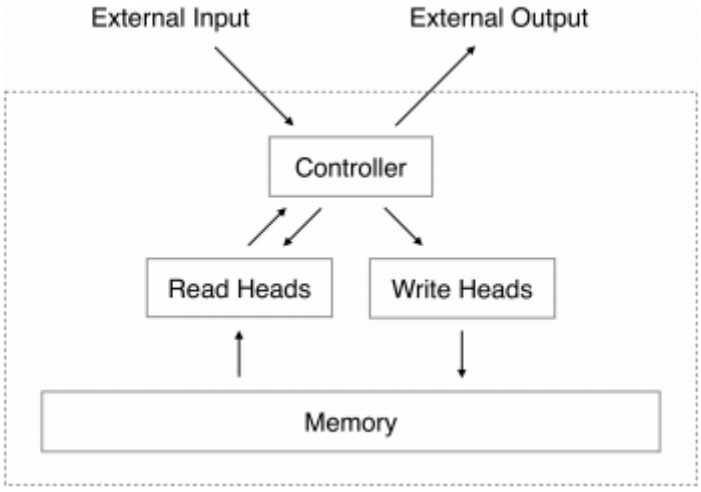
Για να γίνει περισσότερο κατανοητή η διαδικασία της προσπέλασης κατά τοποθεσία, αρκεί να αναλυθεί σε 3 περιπτώσεις:

- Η βαθμωτή πύλη $g_t = 1$ οπότε έχουμε προσπέλαση κατά περιεχόμενο.
- Η βαθμωτή πύλη $g_t = 0$ οπότε τα βάρη από το προηγούμενο βήμα w_{t-1} περιστρέφονται από τις μετατοπίσεις βαρών s_t, και έτσι μπορεί επαναληπτικά να μεταβληθεί η προσοχή σε μια ακολουθία διευθύνσεων.
- Χρησιμοποιούνται και οι 2 τεχνικές, δηλαδή η προσπέλαση μέσω περιεχομένου, μεταβάλεται από τις μετατοπίσεις επιτρέποντας στο μηχανισμό εστίασης να πάει σε μια διεύθυνση δίπλα, αλλά όχι επάνω στη διεύθυνση με το πιο σχετικό περιεχόμενο.



Σχήμα 5.1: Ο μηχανισμός διευθυνσιοδότησης των νευρωνικών μηχανών Turing [6].

Το δεύτερο κομμάτι μιας Νευρωνικής Μηχανής Turing είναι ο ελεγκτής, ο οποίος μπορεί να είναι είτε ένα δίκτυο (LSTM) είτε ένα δίκτυο με προς τα εμπρός τροφοδότηση (feedforward network). Και στις δύο περιπτώσεις οι παράμετροι του δικτύου εκπαιδεύονται με οπισθοδιάδοση. Στα πειράματα τους, οι Graves et al. έδειξαν ότι το δίκτυο έχει την ικανότητα της συσχετισμένης ανάκλησης, η οποία είναι πολυ χρήσιμη για το Few-shot Learning.



Σχήμα 5.2: Αρχιτεκτονική των Νευρωνικών Μηχανών Turing [6].

Στο πλαίσιο του Few-shot Learning ιδιαίτερο ενδιαφέρον παρουσιάζει η προσπέλαση κατά περιεχόμενο, καθώς στο Σύνολο Ερωτημάτων του επεισοδίου, το δίκτυο καλείται να αναγνωρίσει το πιο σχετικό παράδειγμα του Συνόλου Υποστήριξης. Με αυτόν το στόχο, οι Santoro et al. [20] πρότειναν ένα νέο μηχανισμό προσπέλασης όπου αγνοείτο τελείως η δυνατότητα της προσπέλασης κατά τοποθεσίας, και αντικαθίσταται με την λιγότερο πρόσφατη χρησιμοποιηθείσα προσπέλαση (Least Recently Used Accessed -LRUA), σύμφωνα με την οποία κατά την εγγραφή στη μνήμη, οι τοποθεσίες που επιλέγονται είναι είτε η λιγότερο πρόσφατη χρησιμοποιηθείσα θέση μνήμης είτε η πιο πρόσφατη. Αυτό επιτυγχάνεται εισάγοντας τα βάρη

χρήσης (usage weights) w_t^u τα οποία σε κάθε χρονικό βήμα ανανεώνονται ως εξής:

$$w_t^u \leftarrow \gamma w_{t-1}^u + w_t^r + w_t^w \tag{5.9}$$

όπου γ είναι μια παράμετρος (decay) και w_t^r είναι τα βάρη ανάγνωσης. Με βάση αυτά, ορίζονται τα λιγότερο πρόσφατα χρησιμοποιηθείσα βάρη:

$$w_t^{lu}(i) = \begin{cases} 0 & \text{if } w_t^u(i) > m(w_t^u, n) \\ 1 & \text{else} \end{cases} \tag{5.10}$$

Τέλος τα βάρη εγγραφής, τα οποία τελικά χρησιμοποιούνται για την εγγραφή στην μνήμη προκύπτουν ως ένας συνδυασμός των w_t^{lu} και των βαρών ανάγνωσης, δηλαδή:

$$w_t^w \leftarrow \sigma(a) w_{t-1}^r + (1 - \sigma(a)) w_{t-1}^{lu} \tag{5.11}$$

όπου σ η λογιστική συνάρτηση. Η τελική ανανέωση της μνήμης γίνεται ως εξής:

$$M_t(i) \leftarrow M_{t-1}(i) + w_t^w(i) k_t, \forall i \tag{5.12}$$

οπότε η μνήμη ανανεώνεται και μόνο οι λιγότερο χρησιμοποιηθείσες περιοχές διαγράφονται.

Το δίκτυο των Santoro et al. εκπαιδεύτηκε σε 100,000 επεισόδια από 5 κλάσεις (One-shot 5-way), με δεδομένα από το σύνολο εκπαίδευσης. Κατά την αξιολόγηση, οι παράμετροι του ελεκτή παρέμειναν σταθερές με μόνο τη μνήνη να ανανεώνεται για το κάθε παράδειγμα που συναντάει. Μια σημαντική διαφορά αυτής της μεθόδου, σε σχέση με τις προηγούμενες τεχνικές που μελετήσαμε, είναι ότι το επεισόδιο παρουσιάζεται ως ακολουθία στο Δίκτυο με Επαυξημένη Μνήμη. Επειδή ο τρόπος εξέτασης ήταν αρκετά διαφορετικός από αυτόν που είχαν προτείνει ο Lake et al. [12], [13], οι συγγραφείς συνέκριναν τα αποτελέσματά τους με την επίδοση ανθρώπων που ακολούθησαν την ίδια διαδικασία αξιολόγησης, όπου παρατήρησαν ότι η επίδοση του δικτύου υπερτερεί της ανθρώπινης ικανότητας όταν έχουμε τουλάχιστον 15 κλάσεις (χαρακτήρες) στο επεισόδιο.

Κάτι που αξίζει να σημειωθεί είναι ότι αλλάζοντας τον τρόπο κωδικοποίησης των κλάσεων οι συγγραφείς πέτυχαν καλύτερα αποτελέσματα, δίνοντας την ικανότητα στο δίκτυο να ανταπεξέλθει σε επεισόδια με περισσότερες κλάσεις. Συγκεκριμένα, αρχικά χρησιμοποίησαν one-hot encoding, δηλαδή κάθε κλάση αντιστοιχεί σε ένα διάνυσμα που αποτελείται από 0 και μόνο ένα 1. Αλλάζοντας το διάνυσμα σε string, το οποίο αποτελείται από χαρακτήρες, όπου ο κάθε χαρακτήρας είναι one-hot encode, το δίκτυο απέκτησε μια πιο συμπαγή αναπαράσταση για τις ετικέτες παρουσιάζοντας καλύτερη απόδοση. Με one-hot κωδικοποίηση μεγέθους 25, το δίκτυο μπορεί να αναπαραστήσει 25 κλάσεις, ενώ αντίθετα αν κάθε 5-αδα bit, αντιστοιχεί σε έναν χαρακτήρα 'a'-'e' που είναι one-hot κωδικοποιημένος, τότε μπορούν να αναπαρασταθούν $5^5 = 3.125$ κλάσεις.

Κεφάλαιο 6

Υλοποίηση και πειραματική σύγκριση των μεθόδων

6.1 Βασικά στοιχεία υλοποίησης

Όταν οι Lake et al. δημιούργησαν το Omniglot dataset βάζοντας 20 ανθρώπους να σχεδιάσουν τους 1623 χαρακτήρες από 50 διαφορετικά αλφάβητα, χώρισαν το σύνολο Δεδομένων σε μια διάτμιση εκπαίδευσης-αξιολόγησης (train-test split) με συγκεκριμένα αλφάβητα να ανήκαν στο σύνολο εκπαίδευσης και τα υπόλοιπα στο σύνολο αξιολόγησης. Συγκεκριμένα 30 από τα αλφάβητα ανήκαν στο σύνολο εκπαίδευσης και τα υπόλοιπα στο σύνολο αξιολόγησης. Επίσης ως κριτήριο αξιολόγησης πρότειναν την αξιολόγηση εντός του αλφάβητου (within alphabet classification) δηλαδή το επεισόδιο του One-shot learning να αποτελείται από χαρακτήρες από το ίδιο αλφάβητο. Αντίθετα, η κατηγοριοποίηση μεταξύ αλφαβήτων (between alphabet classfication), αποτελείται από χαρακτήρες επιλεγμένους τύχαια από τα διαθέσιμα αλφάβητα. Πολλές από τις αρχιτεκτονικές που μελετάμε χρησιμοποίησαν διαφορετικές διατμήσεις και κατηγοριοποίηση μεταξύ αλφαβήτων, απαλείφοντας την ιεραρχία των αλφάβητων. Είναι προφανές ότι χρησιμοποιώντας μεγαλύτερο σύνολο εκπαίδευσης και μικρότερο σύνολο αξιολόγησης τα αποτελέσματα της μεθόδου, τα οποία συγκρίνονται στο σύνολο αξιολόγησης, θα είναι καλύτερα αφού περιέχει επιπλέον πληροφορία την οποία μπορεί να αξιοποιήσει-μάθει η μέθοδος. Κάτι όχι τόσο προφανές είναι το γεγονός ότι υπάρχει σημαντική διαφορά στην εντός και μεταξύ αλφαβήτων κατηγοριοποίηση. Αν και εκ πρώτης όψεως το εντός πρόβλημα φαίνεται ευκολότερο, καθώς ένα αλφάβητο δεν περιέχει ίδιους χαρακτήρες, σύμφωνα με τα πειραματικά αποτέλεσματα είναι δυσκολότερο, στο πλαίσιο του Omniglot challenge κάτι το οποίο υποστηρίζεται και στο [14]. Περαιτέρω ανάλυση επ΄ αυτού του θέματος θα υπάρξει στο επόμενο κεφάλαιο.

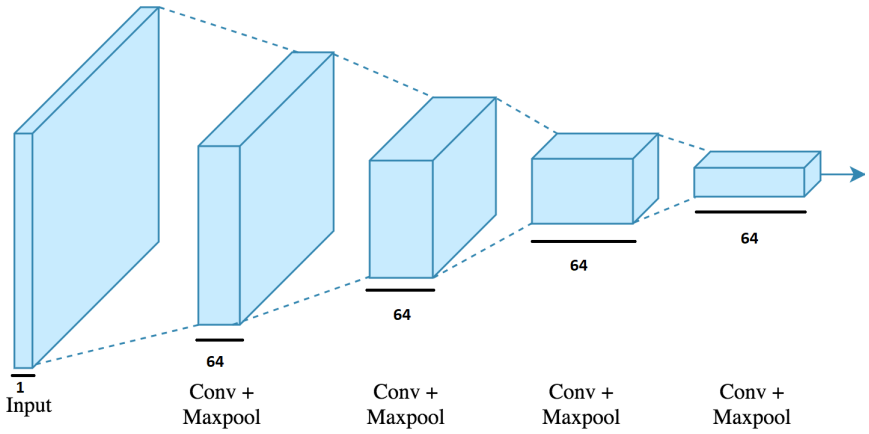
Μια άλλη τεχνική που συναντάται στην βιβλιογραφία για την επίτευξη καλύτερων αποτελεσμάτων, και ιδιαίτερα στο πλαίσιο του Few-shot Learning, είναι η επαύξηση των δεδομένων (Data Augmentation). Στα περισσότερα σύνολα δεδομένων που περιλαμβάνουν εικόνες υλοποιείται με τυχαία crop της εικόνας και αφινικούς μετασχηματισμούς. Ένας αφινικός μετασχηματισμός $f : X \to Y$ μπορεί να εκφραστεί ως σύνθεση μιας μετατόπισης (translation) και ενός γραμμικού μετασχηματισμού, δηλαδή:

$$x \mapsto Mx + b \tag{6.1}$$

όπου το M είναι ένας γραμμικός μετασχηματισμός, το x είναι διάνυσμα στον X και το b είναι διάνυσμα στον Y. Στο πλαίσιο του Omniglot Dataset, το οποίο αποτελείται από εικόνες 108×108 pixel, συναντάμε συνήθως στη βιβλιογραφία περιστροφές και ακέραια πολλαπλάσια

των 90, οι οποίες ωστόσο ορίζονται ως διαφορετικές κλάσεις. Το ίδιο εφαρμόζουν και στο σύνολο εκπαίδευσης πρακτικά αυξάνοντας τα δεδομένα ×4, αυξάνοντας και το variation of data. Η φύση του προβλήματος του Few-shot Learning απαιτεί να βρεθούν αρχιτεκτονικές που αποδίδουν καλά με λίγα παραδείγματα. Έτσι, η χρησιμοποίηση επαύξησης δεδομένων, παρότι με αυτήν επιτυγχάνονται καλύτερα αποτελέσματα, αντιβαίνει στην παραπάνω λογική. Για το λόγο αυτό τις αρχιτεκτονικές που υλοποιήσαμε τις συγκρίναμε στο αρχικό σύνολο δεδομένων και όχι στο επαυξημένο.

Όλες οι αρχιτεκτονικές που συγκρίνουμε, χρησιμοποιούν έναν εξαγωγέα χαρακτηριστικών τον οποίο υλοποιούν χρησιμοποιώντας ένα συνελικτικό νευρωνικό δίκτυο. Στη βιβλιογραφία χρησιμοποιούνται διαφορετικές αρχιτεκτονικές για το συνελικτικό νευρωνικό δίκτυο που επιτελεί την εξαγωγή των χαρακτηριστικών. Αυτό είναι άλλωστε αναμενόμενο, καθώς ορισμένα επιστημονικά άρθρα έχουν μεγάλη χρονική διαφορά μεταξύ τους, και οι μέθοδοι που χρησιμοποιούνται στα συνελικτικά νευρωνικά δίκτυα εξελίσσονται διαρκώς. Για την πειραματική σύγκριση υλοποιήσαμε ένα από τα πιο δημοφιλη συνελικτικά νευρωνικά δικτύα που χρησιμοποιείται για την κατηγοριοποίηση στο Omniglot, το 4CONV το οποίο αποτελείται από 4 επιπέδα (σχήμα 6.2), όπου χρησιμοποιούμε διαδοχικά 3×3 πυρήνες συνέλιξης βάθους 64 χαρακτηριστικών, batch normalization και μέγιστο τράβηγμα (max pooling) 2×2. Έτσι τελικά έχουμε ένα διάνυσμα χαρακτηριστικών μεγέθους 64, το οποίο κάθε αρχιτεκτονική χρησιμοποιεί διαφορετικά. Να επισημάνουμε σε αυτό το σημείο ότι όλες οι αρχιτεκτονικές που συγκρίναμε δεν προσθέτουν επιπλέον παραμέτρους από αυτές του δικτύου, δηλαδή χρησιμοποιούν το ίδιο πλήθος παραμέτρων. Η διαφορά έγκειται στις παραμέτρους που μαθαίνει η κάθε μέθοδος. Αξίζει να σημειωθεί ότι ίδιο πλήθος παραμέτρων δεν σημαίνει και αντίστοιχους χρόνους εκπαίδευσης και αξιολόγησης γιατί αυτό εξαρτάται από την αρχιτεκτονική, όπως για παράδειγμα στο MAML, όπου η κανονική υλοποίηση του χρησιμοποιεί 2ου βαθμού παραγώγους, κάνοντας την εκπαίδευση από το κάθε επεισόδιο πιο αργή. Επίσης σημαντική μείωση στον χρόνο εκπαίδευσης επέφερε το batch normalization, το οποίο αν και προσθέτει παραμέτρους, επιτρέπει πολύ μεγαλύτερους ρυθμούς μάθησης.



Σχήμα 6.1: Η αρχιτεκτονική του συνελικτικού δικτύου που χρησιμοποιήσαμε.

Για τη σύγκριση των μεθόδων χρησιμοποιήθηκαν επεισόδια αξιολόγησης από το Omniglot Dataset, τα οποία δημιουργήθηκαν με τυχαίο τρόπο, με τη διαφορά ότι στα επεισόδια εντός αλφάβητου, οι κλάσεις του επεισοδίου ήταν τυχαίοι χαρακτήρες από το συγκεκριμένο αλφάβητο, ενώ στα εκτός αλφαβήτου, οι χαρακτήρες είχαν δειγματοληφθεί τυχαία από όλα τα αλφάβητα, χωρίς ιεραρχία. Για την κατηγοριοποίηση εντός αλφαβήτου δημιουργήσαμε One-shot 20-way επεισόδια για κάθε αλφάβητο και υπολογίσαμε την ακρίβεια στο συνόλο ερωτημάτων του κάθε

επεισοδίου. Τη διαδικασία αυτή την επαναλάβαμε 3 φορές για κάθε γλώσσα από τις 20, και βρήκαμε το μέσο ποσοστό επιτυχίας. Το ίδιο κάναμε και για την κατηγοριοποίηση μεταξύ των αλφάβητων, με μόνη διαφορά ότι δημιουργήσαμε τυχαία 60 επεισόδια από όλους τους διαθέσιμους χαρακτήρες. Επειδή τα αποτελέσματα παρουσιάζουν διακύμανση ανάλογα με την εκτέλεση, αναφέρουμε τον μέσο όρο επιτυχίας από 10 εκπαιδεύσεις και αξιολογήσεις της ίδιας αρχιτεκτονικής.

6.2 Στοιχεία κώδικα

Υλοποιήσαμε τον κώδικα χρησιμοποιώντας την πλατφόρμα του Anaconda, η οποία είναι open-source και διευκολύνει το machine learning σε Linux και Windows σε Python. Η εκπαίδευση των συνελικτικών δικτύων έγινε αρκετά ευκολότερη και γρηγορότερη χρησιμοποιώντας την βιβλιοθήκη PyTorch, η οποία είναι βασισμένη στην βιβλιοθήκη Torch. Το Torch είναι κατάλληλο για πράξεις μεταξύ tensors, δηλαδή πολυδιάστατων πινάκων, στους οποίους εφαρμόζονται μετασχηματισμοί. Το κύριο προτέρημα αυτής της βιβλιοθήκης είναι ότι επιταχύνει τις πράξεις μεταξύ των tensor χρησιμοποιώντας την GPU, ενώ παράλληλα κάνει τον παράλληλο προγραμματισμό σημαντικά πιο εύκολο. Τέλος η σημαντικότερη συνεισφορά της είναι ότι έχει ενσωματωμένο ένα Autograd module, το οποίο υλοποιεί αυτόματα την οπισθοδιάδοση των κλίσεων. Ορίζοντας τη δομή του δικτύου, δεν χρειάζεται να κατασκευάσουμε τον γράφο των κλίσεων, καθώς το υλοποιεί αυτόματα, και ειδικά σε αρχιτεκτονικές όπως το MAML κάνει τον προγραμματισμό ευκολότερο.

```
def conv_block(in_chan, out_chan):
    return nn.Sequential(
        nn.Conv2d(in_chan, out_chan, 3, padding =1),
        nn.ReLU(),
        nn.BatchNorm2d(out_chan),
        nn.MaxPool2d(kernel_size=2, stride=2)
    )

class ConvNet(nn.Module):
    def __init__(self):
        super(ConvNet, self).__init__()
        self.conv1b = conv_block(1,64)
        self.conv2b = conv_block(64,64)
        self.conv3b = conv_block(64,64)
        self.conv4b = conv_block(64,64)
        ##
        self.fc = nn.Linear(64 * 1* 1, NUM_CL)


    def embedded(self,x):
        x = self.conv1b(x)
        x = self.conv2b(x)
        x = self.conv3b(x)
        x = self.conv4b(x)
        x = x.view(-1, 64*1*1)
        return x

    def forward(self, x):
        x = self.embedded(x)
        x = self.fc(x)
        return x
```

Σχήμα 6.2: Υλοποίηση του Συνελικτικού δικτύου σε PyTorch.

Όπως αναφέρθηκε προηγουμένως, η διάκριση εντός αλφάβητου και μεταξύ αλφάβητων αποτελούν διαφορετικές εργασίες. Για τον σκοπό αυτό, συγκρίναμε τις αρχιτεκτονικές σε όλες τις δυνατές περιπτώσεις, δηλαδή προχωρήσαμε σε ξεχωριστή τόσο εκπαίδευση όσο και αξιολόγηση για την εντός και μεταξύ διάκριση. Έτσι, για κάθε αρχιτεκτονική έχουμε 4 ποσοστά επιτυχίας, για όλους τους συνδυασμούς των περιπτώσεων. Εξαίρεση αποτελούν τα Σιαμαία Δίκτυα, όπου δεν έχει ιδιαίτερο νόημα η εκπαίδευση εντός καθώς με πιθανότητα 50% διαλέγεται τυχαίο ζεύγος από διαφορετικές κλάσεις, και το Simple ConvNet, δηλαδή το παραδοσιακό νευρωνικό συνελικτικό δίκτυο που εκπαιδεύεται με συστάδες.

6.3 Αποτελέσματα

6.3.1 Simple ConvNet

Υλοποιήσαμε το απλό συνελικτικό νευρωνικό δίκτυο (σχήμα 6.2) που αποτελείται από τον εξαγωγέα χαρακτηριστικών που περιγράφηκε προηγουμένως ακολουθούμενο από ένα πλήρως συνδεδεμένο επίπεδο ίσο με τον αριθμό κλάσεων εκπαίδευσης, δηλαδή των διαφορετικών χαρακτήρων στο σύνολο εκπαίδευσης. Η συνάρτηση σφάλματος που ελαχιστοποιεί είναι η cross entropy loss [25]. Μετά την εκπαίδευση, στο στάδιο της αξιολόγησης, εφαρμόσαμε τον εξαγωγέα χαρακτηριστικών και στη συνέχεια πήραμε την Ευκλείδια απόσταση μεταξύ όλων των ζευγών στο Σύνολο Ερωτημάτων και το Σύνολο Υποστήριξης, προβλέποντας την κλάση του προτύπου με την μικρότερη απόσταση. Χρησιμοποιήσαμε πολύ μεγάλο ρυθμό μάθησης, λόγω του batch normalization, με batch size 128 πρότυπα, step size 5 και gamma= 0.1.

Εκπαίδευση/Αξιολόγηση	Πειραματικά αποτελέσματα	Αποτελέσματα στην βιβλιογραφία
ολόκληρο/εντός	74.4%	86.5%
ολόκληρο/μεταξύ	87.8%	-

Πίνακας 6.1: Αποτελέσματα για το Simple ConvNet.

Παρατηρούμε μια απόκλιση στα αποτελέσματα, και αυτό ίσως οφείλεται στο γεγονός ότι η δομή του απλού συνελικτικού δικτύου που χρησιμοποιήθηκε στο [13] δεν αναφέρεται. Στο πλαίσιο της δικιά μας σύγκρισης χρησιμοποιήσαμε την ίδια αρχιτεκτονική για εξαγωγέα χαρακτηριστικών που αναφέρθηκε προηγουμένως. Επίσης η αξιολόγηση του συνελικτικού δικτύου στην βιβλιογραφία έχει γίνει σε 10 αλφάβητα και όχι σε 20.

6.3.2 Σιαμαία Δίκτυα

Τα Σιαμαία είχαν την πιο αργή εκπαίδευση σε σχέση με τις υπόλοιπες αρχιτεκτονικές βασισμένες σε μετρικές. Αυτό οφείλεται στο ότι μαθαίνουν σε ζεύγη παραδειγμάτων, οπότε πρέπει να δουν πάρα πολλά όμοια και διαφορετικά ζεύγη για να εκπαιδευτούν. Κατά την εκπαίδευση ένα ζεύγος είχε πιθανότητα 50% να είναι από την ίδια κλάση και 50% από διαφορετική. Στο πλαίσιο του One-shot 20-way task που μελετάμε, μόνο 1 από τα 20 πρότυπα στο Σύνολο Υποστήριξης ανήκει στην ίδια κλάση με το κάθε παράδειγμα από το Σύνολο Ερωτημάτων.

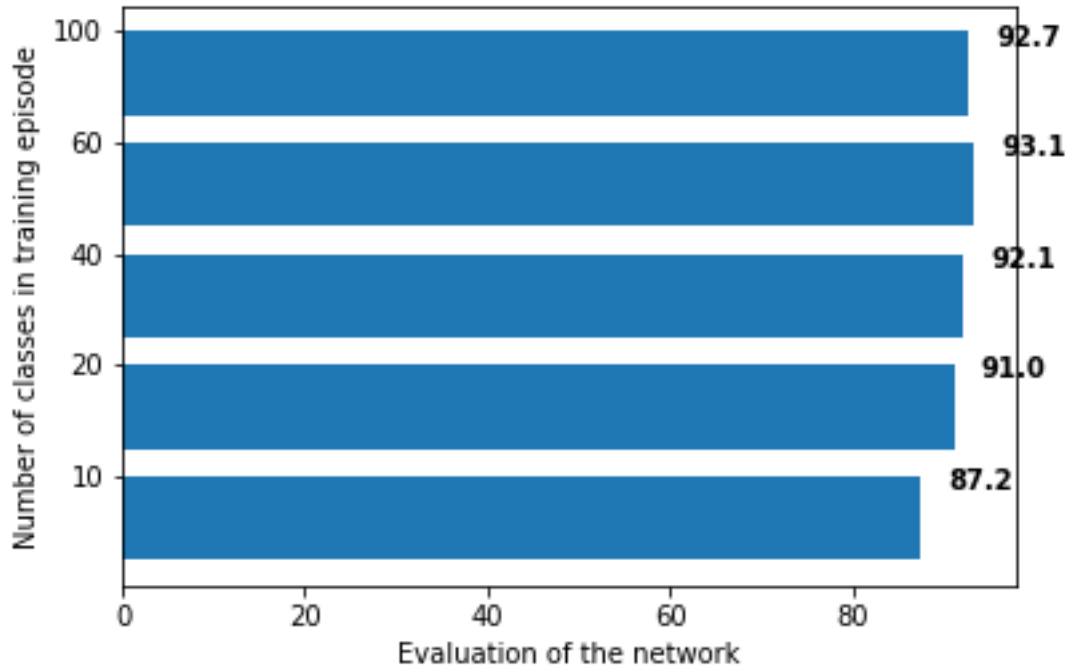
Εκπαίδευση/Αξιολόγηση	Πειραματικά αποτελέσματα	Αποτελέσματα στην βιβλιογραφία
μεταξύ/εντός	70.2%	-%
μεταξύ/μεταξύ	83.2%	91.54%

Πίνακας 6.2: Αποτελέσματα για το Σιαμαίο Δίκτυο.

Στα Σιαμαία Δίκτυα παρατηρήθηκε η μεγαλύτερη διαφορά ανάμεσα στα πειραματικά αποτελέσματα και σε αυτά στη βιβλιογραφία. Και πάλι πρέπει να σημειωθεί ότι η αρχιτεκτονική του εξαγωγέα χαρακτηριστικών είναι πολύ διαφορετική σε σχέση με αυτή που χρησιμοποίησαν οι Koch et al. στο [10]. Όπως φαίνεται στο Σχήμα 3.1, ο χώρος του της εξόδου είναι 4 φορές μεγαλύτερος από αυτόν που χρησιμοποιήσαμε, οπότε δεδομένου ότι χρησιμοποιήθηκε το ίδιο σύνολο για την εκπαίδευση και την αξιολόγηση με αυτό του [10], να εξηγεί εν μέρη την διαφορά. Επίσης η υλοποίηση του [10] περιλαμβάνει weight decay για την αποφυγή της υπερπροσαρμογής.

6.3.3 Δίκτυα Ταιριάσματος

Αυτή είναι η πρώτη αρχιτεκτονική που μελετάμε η οποία εκπαιδεύεται σε επεισόδια, οπότε ο τρόπος εκπαίδευσης ταυτίζεται με τον τρόπο αξιολόγησης, με μόνη διαφόρα να αποτελεί το σύνολο δεδομένων που χρησιμοποιείται την κάθε φορά. Πειραματιστήκαμε με εκπαίδευση εντός/μεταξύ αλφαβήτων, αλλά και με αξιολόγηση εντός/μεταξύ αλφαβήτων. Λάμβάνοντας υπόψη τα συμπεράσματα των Snell et al. [23] πειραματιστήκαμε με μεγαλύτερο K-way κατά την εκπαίδευση. Όπως φαίνεται και στο παρακάτω γράφημα, αυξάνοντας το πλήθος των κλάσεων σε ένα επεισόδιο, που συνεπάγεται και αύξηση της δυσκολίας του, η αρχιτεκτονική μαθαίνει καλύτερες εσωτερικές αναπαραστάσεις για τα δεδομένα.



Σχήμα 6.3: Απόδοση του δικτύου ανάλογα με τον αριθμό των κλάσεων του επεισοδίου κατά την εκπαίδευση. Τα ποσοστά αφορούν εκπαίδευση και αξιολόγηση μεταξύ αλφάβητων.

Εκπαίδευση/Αξιολόγηση	Πειραματικά αποτελέσματα	Αποτελέσματα στην βιβλιογραφία
εντός/εντός	78.5%	-%
μεταξύ/εντός	83.2%	-%
εντός/μεταξύ	91.5%	-%
μεταξύ/μεταξύ	93.1%	93.8%

Πίνακας 6.3: Αποτελέσματα για τα Δίκτυα Ταιριάσματος.

Όπως παρατηρούμε, πετυχαίνουμε περίπου τα ίδια αποτελέσματα με το δίκτυο που χρησιμοποίησαν οι συγγραφείς, με περίπου 20% λιγότερα δεδομένα κατά την εκπαίδευση. Αυτή η διαφορά οφείλεται στην αύξηση του αριθμού των κλάσεων κατά την εκπαίδευση. Για την εκπαίδευση του Δικτύου Ταιριάσματος χρησιμοποιήσαμε 60 κλάσεις και ρυθμό μάθησης 0.1

με μεταβλητό μέγεθος βήματος, μεταξύ 5 και 40.

6.3.4 Πρωτότυπα Δίκτυα

Τα Δίκτυα Ταιριάσματος και τα Πρωτότυπα Δίκτυα διαφέρουν μόνο ως προς την συνάρτηση απόστασης στο πλαίσιο του One-shot Learning. Έτσι χρησιμοποιήσαμε την ίδια αρχιτεκτονική και παραμέτρους με τα Δίκτυα Ταιριάσματος.

Εκπαίδευση/Αξιολόγηση	Πειραματικά αποτελέσματα	Αποτελέσματα στην βιβλιογραφία
εντός/εντός	78.7%	-%
μεταξύ/εντός	83.3%	-%
εντός/μεταξύ	91.7%	-%
μεταξύ/μεταξύ	93.4%	96%

Πίνακας 6.4: Αποτελέσματα για τα Πρωτότυπα Δίκτυα.

6.3.5 MAML

Υλοποιήσαμε τον 1ου βαθμού αλγόριθμο MAML, πάλι με το δίκτυο 4CONV ως εξαγωγέα χαρακτηριστικών. Το πέρασμα της κλίσης (gradient) μέσω της κλίσης έγινε με τη χρήση του torch.nn.functional μέσω του οποίου δίνεται η δυνατότητα να προσομοιώσουμε τις πράξεις του συνελικτικού νευρωνικού δικτύου για να χρησιμοποιήσουμε το Autograd του PyTorch για την οπισθο-διάδοση. Στην ουσία κάνουμε αντιγραφή των βαρών του δικτύου, τα οποία στην συνέχεια χρησιμοποιούνται για να υλοποιηθεί το δίκτυο ως τελεστής (πράξη) στα δεδομένα εισόδου.

Εκπαίδευση/Αξιολόγηση	Πειραματικά αποτελέσματα	Αποτελέσματα στην βιβλιογραφία
μεταξύ/εντός	79.7%	-%
μεταξύ/μεταξύ	90.5%	95.8%

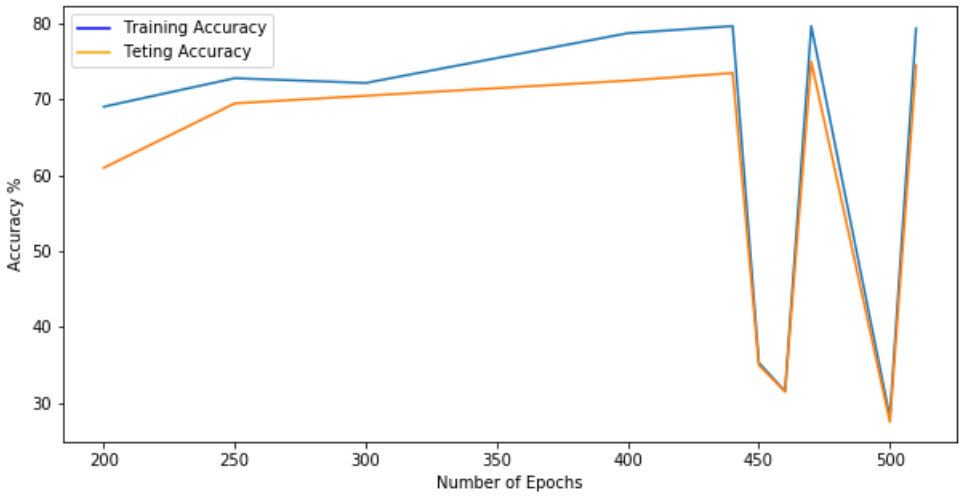
Πίνακας 6.5: Αποτελέσματα για τον αλγόριθμο MAML.

Ο αλγόριθμος MAML είναι ιδιαίτερα ευαίσθητος στις υπερπαραμέτρους που χρησιμοποιούνται, και μπορεί να οδηγηθεί σε αστάθεια [1], κάτι που παρατηρήθηκε και στις δοκιμές μας όπως φαίνεται στο παρακάτω σχήμα. Οι Αντωνίου et al. [1] προτείνουν διάφορες λύσεις για να αποφευχθεί το πρόβλημα αυτό. Μια από τις πιο βασικές είναι η χρησιοποίηση ανόπτησης (annealing) κατά την εκπάιδευση, κάτι το οποίο εκμεταλλεύονται και τα παραδοσιακά νευρωνικά δίκτυα. Ο αλγόριθμος MAML είναι ιδιαίτερα υπηρεπής και στο πρόβλημα της εξαφανιζόμενης κλίσης (vanishing gradient), καθώς σε οι κλίσεις διαδίδονται μέσα από συνεχόμενα βήματα οπισθοδιάδοσης. Χρησιμοποιόντας συντομότερες συνδέσεις (shortcut connections), όπως και στην περίπτωση του Υπολειπόμενου δικτύου, οι κλίσεις μπορούν να διαδοθούν και στα προηγούμενα επίπεδα, με μόνη διαφορά ότι στην περίπτωση του MAML οι κλίσεις διαδίδονται μεταξύ διαφορετικών βημάτων της οπισθοδιάδοσης, δηλαδή:

$$\theta = \theta - \beta \nabla_\theta \sum_{b=1}^{B} \sum_{i=0}^{M} u_i \mathcal{L}_{T_b}(f_{\theta_i^b}) \tag{6.2}$$

όπου B είναι το σύνολο των εργασιων στην συστάδα και M είναι το πλήθος των βημάτων οπισθο-διάδοσης. Τα u_i είναι συντελεστές που σταθμίζουν την συνεισφορά του κάθε βήματος,

οπότε αν έχουμε $u_M = 1$ και $u_i = 0, i < M$ τότε προκύπτει η αρχική εκδοχή του MAML .

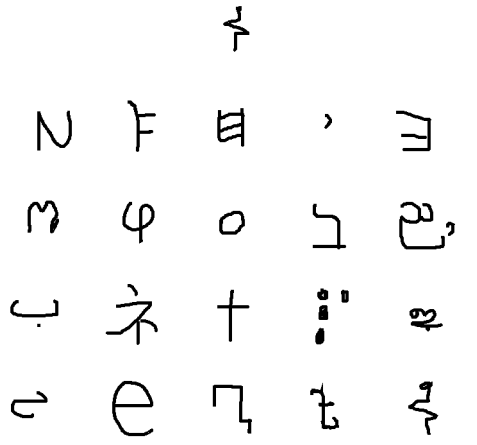


Σχήμα 6.4: Αστάθεια κατά την εκπαίδευση ενός δικτύου MAML.

Κεφάλαιο 7

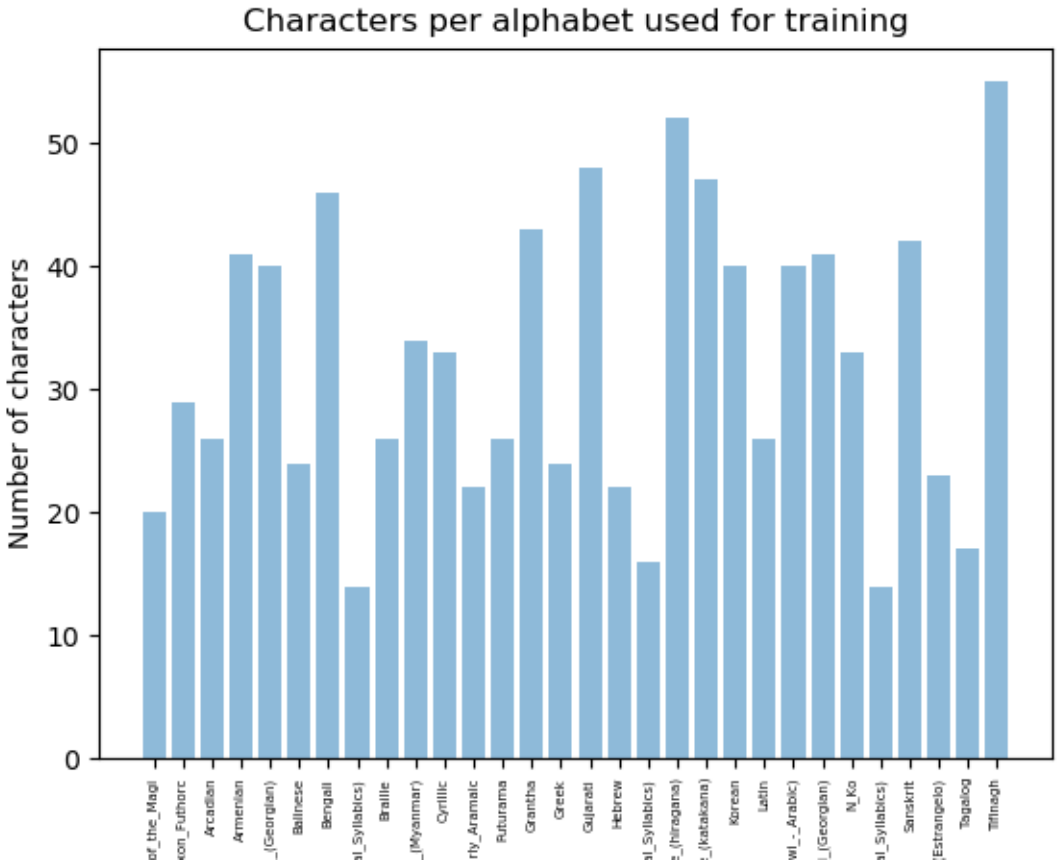
Συμπεράσματα και πιθανές κατευθύνσεις

Στην εργασία αυτή μελετήθηκαν κάποιες από τις κυριάρχες αρχιτεκτονικές για το Few-shot Learning στο πλαίσιο του Omniglot Dataset. Αρκετές από αυτές τις αρχιτεκτονικές είχαν εκπαιδευτεί σε διαφορετικές διαιρέσεις του συνόλου δεδομένων και για διαφορετικές εργασίες (εντός/μεταξύ αλφάβητων). Όπως προκύπτει από την πειραματική σύγκριση οι εργασίες παρουσιάζουν σημαντική διαφορά στη δυσκολία, κάτι που είχε διατυπωθεί και στο [14], χωρίς ωστόσο να γίνει σχετική σύγκριση. Για λόγους δίκαιας σύγκρισης δοκιμάσαμε όλους τους συνδυασμούς εκπαίδευσης και αξιολόγησης εντός/μεταξύ των αλφάβητων και βρήκαμε ότι η εργασία της κατηγοριοποίησης εντός του αλφάβητου παρουσιάζει από 10% έως και 13% μικρότερη ακρίβεια ανάλογα με την αρχιτεκτονική. Αυτό πιθανότατα να οφείλεται στη μεγάλη απόκλιση που παρουσιάζουν οι χαρακτήρες από διαφορετικά αλφάβητα κάνοντας την εργασία διάκρισης μεταξύ αλφάβητων ευκολότερη.

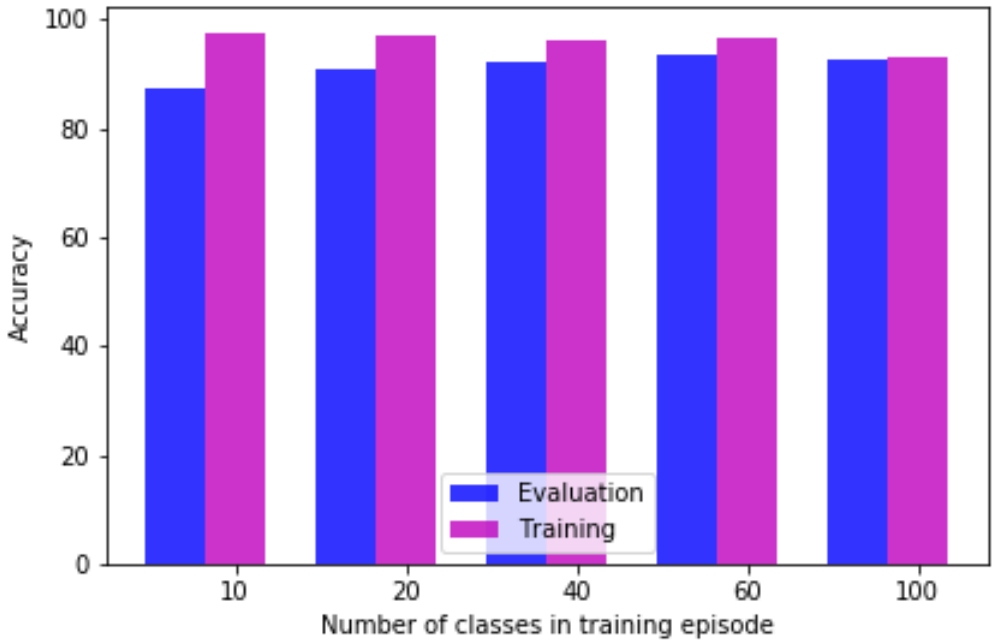


Σχήμα 7.1: Ένα τυχαίο επεισόδιο από εκτός αλφαβήτου εκπαίδευση. Παρατηρούμε σημαντικές διαφορές μεταξύ των γραμμάτων, που διευκολύνουν την αναγνώριση του σωστού ζεύγους.

Ακόμα και για την αξιολόγηση εντός του αλφάβητου, καλύτερο αποτέλεσμα επιφέρει η εκπαίδευση μεταξύ αλφάβητων, όπως φαίνεται από τα αποτελέσματα. Ωστόσο εδώ πρέπει να σημειωθεί ότι στα Δίκτυα Ταιριάσματος και στα Πρωτότυπα Δίκτυα, η εκπαίδευση μεταξύ αλφάβητων έχει γίνει σε δυσκολότερο επεισόδιο 60 κλάσεων. Αντίθετα, το εντός του αλφβήτου επεισόδιο εκπαίδευσης έχει όλους τους δυνατούς χαρακτήρες του αλφάβητου, καθώς όλα τα αλφάβητα εκπαίδευσης είχαν λιγότερους από 60 χαρακτήρες.

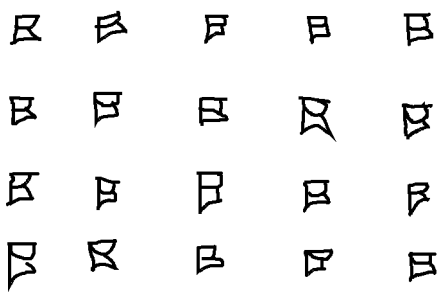


Σχήμα 7.2: Πλήθος χαρακτήρων στο σύνολο εκπαίδευσης ανά αλφάβητο.



Σχήμα 7.3: Σύγκριση της ακρίβειας στην εκπαίδευση και αξιολόγηση, σε συνάρτηση με το πλήθος των κλάσεων. Καθώς ο αριθμός των κλάσεων γίνεται πολύ μεγάλος, το δίκτυο δεν μπορεί να εκπαιδευτεί αποτελεσματικά, και η απόδοση του στην εκπαίδευση μειώνεται.

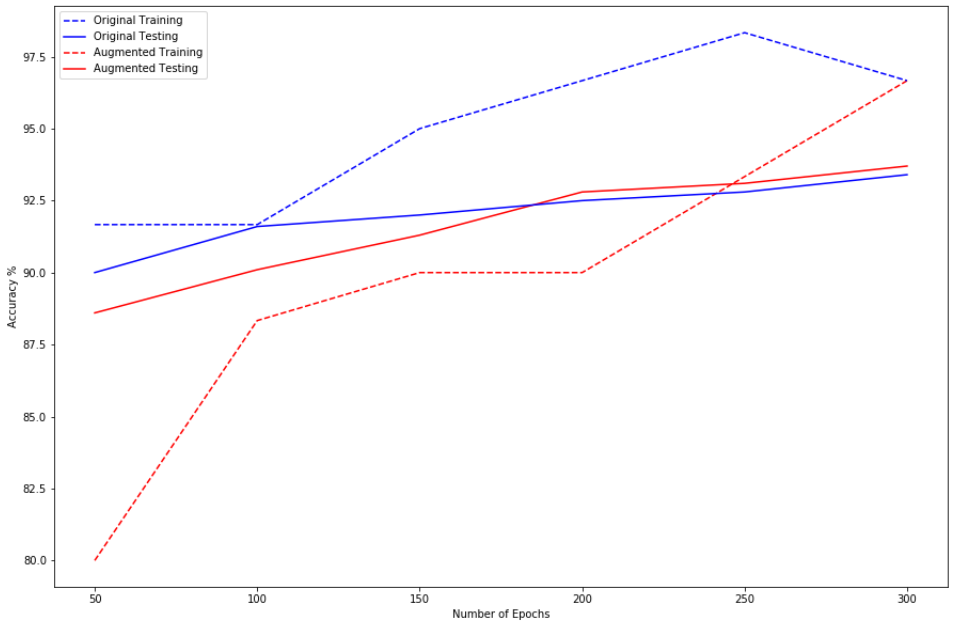
Όπως είδαμε στα Δίκτυα Ταιριάσματος και στα Πρωτότυπα Δίκτυα η αύξηση των κλάσεων κατά την εκπαίδευση, ως ένα συγκεκριμένο βαθμό βοηθάει στην απόδοση του δικτύου. Με λιγότερες κλάσεις ένα επεισόδιο είναι ευκολότερο και επείδη το Few-shot Learning έχει λίγα παραδείγματα από κάθε κλάση, είναι πιο πιθανή η υπερπροσαρμογή. Συνεχίζοντας να αυξάνουμε τις κλάσεις βλέπουμε ότι το δίκτυο φτάνει σε ένα σημείο να αδυνατεί να πετύχει καλή απόδοση στην εκπαίδευση αφού η εργασιά γίνεται πολύ δύσκολη. Όπως και με την στρατηγική Hard Task Meta-Batch κάνοντας την εργασία εκπαίδευσης πιο δύσκολη, το δίκτυο τελικά μαθαίνει καλύτερες αναπαραστάσεις των χαρακτηριστικών (οι συγγραφείς του [24] ονομάζουν την τάση αυτή "grow faster and stronger through hardship"). Ενδιαφέρον θα ήταν να ερευνηθούν άλλοι τρόποι με τους οποίους μπορεί να γίνει η εργασία του Few-shot Learning πιο δύσκολη.



Σχήμα 7.4: Ένας Θιβετιανός χαρακτήρας σχεδιασμένος από 20 διαφορετικούς ανθρώπους.

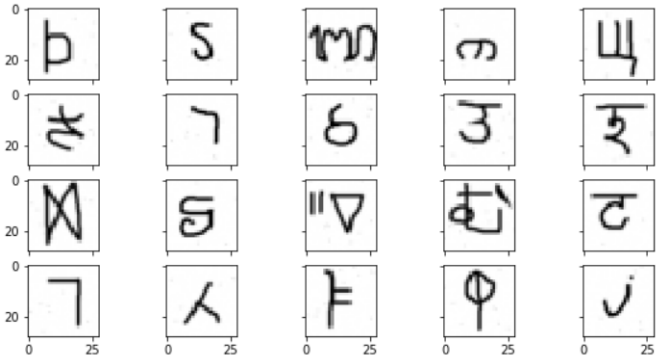
Στην κατεύθυνση αυτή προσθέσαμε τυχαίους αφινικούς μετασχηματισμούς στο κομμάτι

εκπαίδευσης. Σε αντίθεση με ότι έχει χρησιμοποιηθεί στη βιβλιογραφία, οι μετασχηματισμοί αυτοί εισήχθησαν μόνο για το κομμάτι της εκπαίδευσης και τα παραδείγματα δεν αποτελούν διαφορετικές κλάσεις. Στόχος αυτής της προσθήκης είναι το δίκτυο να μην επηρεάζεται από μικρές περιστροφές (rotation invariance) και διαφορές στην κλίμακα (scale invariance) που παρατηρούνται σε διαφορετικούς σχεδιαστές (παράδειγμα στο σχήμα 7.4). Επιτρέψαμε περιστροφή μεταξύ $(-20°, 20°)$ και κλιμάκωση (scaling) στο διάστημα $(0.9, 1.1)$. Αν και το δίκτυο αρχικά παρουσιάζει μικρότερη ακρίβεια στο στάδιο εκπαίδευσης, καθώς ο αριθμός των εποχών μεγαλώνει το δίκτυο επιτυγχάνει την ίδια απόδοση στην εκπαίδευση και υπερτερεί οριακά στην αξιολόγηση (93.7%).

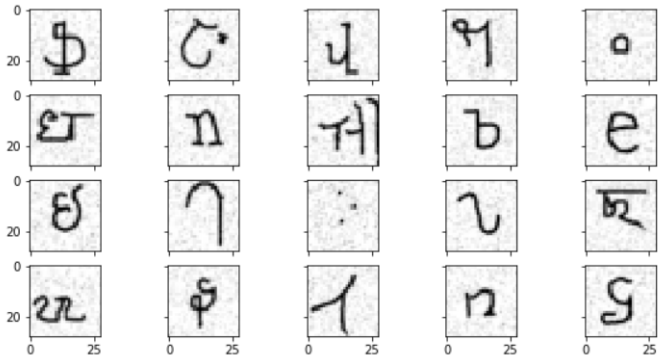


Σχήμα 7.5: Αποτελέσματα προσθέτοντας αφινικούς μετασχηματισμούς στην εκπαίδευση.

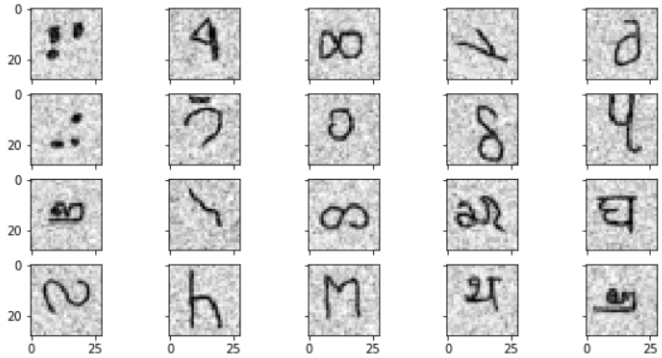
Για να κάνουμε την εκπαίδευση πιο δύσκολη εισάγουμε Γκαουσιανό θόρυβο (Gaussian noise) στις εικόνες. Ο Bishop απέδειξε ότι προσθέτοντας θόρυβο με μικρό πλάτος κατά την εκπαίδευση ισοδυναμεί με την εισαγωγή ενός όρου κανονικοποίησης στην Cross Entropy Loss, αυξάνοντας την ικανότητα γενίκευσης του δικτύου [2]. Τα αποτελέσματα που προκύπτουν από τις μετρήσεις δείχνουν ότι αυτή η μέθοδος είναι λιγότερο αποτελεσματική σε σχέση με τυχαίους αφινικούς μετασχηματισμούς πετυχαίνοντας ακρίβεια 91.5% στην μεταξύ/μεταξύ εκπαίδευση/αξιολόγηση, αρκετά χαμηλότερη από την ακρίβεια δίχως καμία προσθήκη. Στην συνέχεια δοκιμάσαμε να αυξήσουμε προοδευτικά την τυπική απόκλιση του θορύβου, με στόχο την αποφυγή της υπερεκπαίδευσης, κάτι που βελτίωσε τα αποτελέσματα, αλλά εξακολουθεί να υστερεί σε σχέση με τις καθαρές εικόνες. Με βάση αυτά τα πειράματα, παρατηρούμε ότι για την αναγνώριση χαρακτήρων, η εισαγωγή αφινικών μετασχηματισμών βελτιώνει την ικανότητα γενίκευσης, ενώ η προσθήκη θορύβου ωθεί το δίκτυο να μάθει να αντιμετωπίζει τον θόρυβο αντί να εξάγει καλύτερα χαρακτηριστικά για την διάκριση των χαρακτήρων.



Σχήμα 7.6: Ένα τυχαίο σύνολο υποστήριξης πριν την εισαγωγή Γκαουσιανού θορύβου.



Σχήμα 7.7: Ένα τυχαίο σύνολο υποστήριξης μετά την εισαγωγή Γκαουσιανού θορύβου με μέση τιμή 0 και τυπική απόκλιση 0.5.



Σχήμα 7.8: Ένα τυχαίο σύνολο υποστήριξης μετά την εισαγωγή Γκαουσιανού θορύβου με μέση τιμή 0 και τυπική απόκλιση 1. Αυξάνοντας προοδευτικά την τυπική απόκλιση του Γκαουσιανού θορύβου, η εκπαίδευση γίνεται πιο δύσκολη.

Κεφάλαιο 8

Παράρτημα: Κυριότερα κομμάτια κώδικα

Σε αυτό το κεφάλαιο δίνουμε τα σημεία κλειδιά του κώδικα της κάθε υλοποίησης. Στο προηγούμενο κεφάλαιο στο σχήμα 7.3, δώσαμε την υλοποίηση του εξαγωγέα χαρακτηριστικών σε Pytorch.

8.1 Σιαμαία Δίκτυα

Τα Σιαμαία δίκτυα χρησιμοποίουν δύο όμοια Συνελικτικά Δίκτυα, που στην πράξη υλοποιούνται ως ένα δίκτυο, το οποίο εξάγει τα χαρακτηριστικά από το κάθε ζεύγος ξεχωριστά. Στο τέλος η διαφορά στον χώρο εξούδου δίνεται στο πλήρως συνδεδεμένο επίπεδο. Το conv_block είναι το ίδιο που έχει χρησιμοποιηθεί στο Simple ConvNet. Το δίκτυο λαμβάνει ως είσοδο ζεύγη, και η έξοδος y μαζί με την ετικέτα 1 για όμοια ζεύγη ή 0 για διαφορετικά τροφοδείται στην Binary Cross Entropy Loss.

```
class SiamNet(nn.Module):
    def __init__(self):
        super(SiamNet, self).__init__()
        self.conv1b = conv_block(1,64)
        self.conv2b = conv_block(64,64)
        self.conv3b = conv_block(64,64)
        self.conv4b = conv_block(64,64)
        ##
        ## Siamese
        self.fc1=nn.Linear(64,1)
        self.sigmoid = nn.Sigmoid()


    def embedded(self,x):
        x = self.conv1b(x)
        x = self.conv2b(x)
        x = self.conv3b(x)
        x = self.conv4b(x)
        x = x.view(-1, 64*1*1)
        return x

    def forward(self,inp1,inp2):
        y1=self.embedded(inp1)
        y2=self.embedded(inp2)
        y=torch.abs(y1-y2)
        y=self.sigmoid(self.fc1(y))
        return y
```

Σχήμα 8.1: Δομή του Σιαμαίου Δικτύου.

8.2 Δίκτυα Ταιριάσματος

Τα Δίκτυα Ταιριάσματος χρησιμοποιούν τον παραπάνω εξαγωγέα χαρακτηριστικών και στην συνέχεια υπολογίζουν την συνάρτηση λάθους. Όπως φαίνεται και στον παρακάτω κώδικα, χρησιμοποιείται η αρνητική λογαριθμική πιθανοφάνεια για όλες τις αποστάσεις μεταξύ Συνόλου Υποστήριξης και Συνόλου Ερωτημάτων.

```
cos = nn.CosineSimilarity(dim=2, eps=1e-6)
def cosine_dist(x,y):
    n = x.size(0)
    m = y.size(0)
    d = x.size(1)
    x = x.unsqueeze(1).expand(n, m, d)
    y = y.unsqueeze(0).expand(n, m, d)
    dist = 1-cos(x,y)
    return dist

def match_loss(queries, support):
    dists = cosine_dist(queries,support)
    log_p_y = F.log_softmax(-dists,dim=1)
    argmax = torch.argmax(log_p_y,dim=1)
    correct =sum([argmax[i].item()==i for i in range(len(argmax))])
    accu = 100*correct/len(argmax)
    loss = sum([-log_p_y[i][i] for i in range(len(argmax))])
    loss = loss/ (Nc*Nq)
    return loss,accu, correct, len(argmax)

class MatchLoss(Module):
    def __init__(self):
        super(MatchLoss,self).__init__()
    def forward(self,queries, support):
        return match_loss(queries, support)
```

Σχήμα 8.2: Υπολογισμός του λάθους στα δίκτυα Ταιριάσματος.

8.3 Πρωτότυπα Δίκτυα

Για τα Πρωτότυπα Δίκτυα υλοποιήσαμε αποτελεσματικά την τετραγωνισμένη Ευκλείδια απόσταση μεταξύ n διανυσμάτων του Συνόλου Ερωτημάτων και m πρωτοτύπων του Συνόλου Υποστήριξης. Στη συνέχεια πάλι χρησιμοποιείται η αρνητική λογαριθμική πιθανοφάνεια. Πέρα από το λάθος η συνάρτηση επιστρέφει και την αρκίβεια στο επεισόδιο που χρησιμοποιείται για να εντοπίσθει πιθανή υπερ-εκπαίδευση.

```
def euclid_dist(x,y):
    n = x.size(0)
    m = y.size(0)
    d = x.size(1)
    x = x.unsqueeze(1).expand(n, m, d)
    y = y.unsqueeze(0).expand(n, m, d)
    dist = torch.pow(x - y, 2).sum(2)
    return dist #nxm

def prototypical_loss(queries, support):
    dists = euclid_dist(queries,support)
    log_p_y = F.log_softmax(-dists,dim=1)
    argmax = torch.argmax(log_p_y,dim=1)
    correct =sum([argmax[i].item()==i for i in range(len(argmax))])
    accu = 100*correct/len(argmax)
    loss = sum([-log_p_y[i][i] for i in range(len(argmax))])
    loss = loss/ (Nc*Nq)
    return loss,accu, correct, len(argmax)

class PrototypicalLoss(Module):
    def __init__(self):
        super(PrototypicalLoss,self).__init__()
    def forward(self,queries, support):
        return prototypical_loss(queries, support)
```

Σχήμα 8.3: Υπολογισμός του λάθους στα Πρωτότυπα Δίκτυα.

8.4 Αλγόριθμος MAML

Η υλοποίηση του αλγορίθμου MAML διαφέρει σημαντικά από τις προηγούμενες περιπτώσεις. Για την υλοποίηση μέσω PyToch γίνεται χρήση της βιβλιοθήκης nn.functional. Συγκεκριμένα για κάθε επεισόδιο στην μετα-συστάδα, αντιγράφουμε τα βάρη από το base-model και τα ενημερώνουμε για κάθε βήμα κατάβασης πλαγιάς (gradient decent), όπως φαίνεται και στο σχήμα 8.4. Στην συνέχεια εφαρμόζουμε κατάβαση πλαγιάς στο Σύνολο Ερωτημάτων (Query Set), και στην περίπτωση του 2ου βαθμού MAML προωθούμε τα βάρη μέσα από ολόκληρο τον γράφο παραγώγισης. Αντίθετα για την υλοποίηση 1ου βαθμού ορίζουμε hooks τα οποία συνδέουν τις παραγώγους στο αρχικό δίκτυο.

```
## Inner Loop
    fast_net = BaseNet()
    fast_weights = OrderedDict(self.net.named_parameters())  # copy weights
    for j in range(GRAD_STEPS):
        output = fast_net.functional_forward(support, fast_weights)
        loss = fast_net.loss(output, labels)
        gradients = torch.autograd.grad(loss, fast_weights.values(), create_graph= True)
        fast_weights = OrderedDict(
            (name, param - A*grad) for ((name,param),grad) in zip(fast_weights.items(),gradients)
        )
    output = fast_net.functional_forward(query,fast_weights)
    loss = fast_net.loss(output, labels)
    loss.backward(retain_graph=True)
    task_losses.append(loss)
    gradients = torch.autograd.grad(loss, fast_weights.values(), create_graph=True)
    named_grads = {name: g for ((name, _), g) in zip(fast_weights.items(), gradients)}
    task_gradients.append(named_grads)
```

Σχήμα 8.4: Κώδικας του εσωτερικού βρόχου.

```
## Outer Loop
self.net.train()
self.scheduler.step()
self.opt.zero_grad()
meta_loss = torch.stack(task_losses).mean()
if SecondOrder:
    meta_loss.backward()
    self.opt.step()
else:
    sum_task_gradients = {k: torch.stack([grad[k] for grad in task_gradients]).mean(dim=0)
                          for k in task_gradients[0].keys()}
    hooks = []
    for name, param in self.net.named_parameters():
        hooks.append(
            param.register_hook(replace_grad(sum_task_gradients, name))
        )
    outputs = self.net(support)
    labels =  torch.LongTensor(np.random.permutation(NUM_CL))
    loss = self.loss(outputs, labels)
    loss.backward()
    self.opt.step()
    for h in hooks:
        h.remove()
```

Σχήμα 8.5: Κώδικας του εξωτερικού βρόχου.

Βιβλιογραφία

[1] A. Antoniou, H. Edwards, and A. J. Storkey, "How to train your MAML", *CoRR*, vol. abs/1810.09502, 2018.

[2] C. M. Bishop, "Training with noise is equivalent to tikhonov regularization", *Neural Computation*, vol. 7, no. 1, pp. 108–116, 1995.

[3] J. Bromley, J. Bentz, L. Bottou, I. Guyon, Y. Lecun, C. Moore, E. Sackinger, and R. Shah, "Signature verification using a "siamese" time delay neural network", *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 7, p. 25, 1993.

[4] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks", *CoRR*, vol. abs/1703.03400, 2017.

[5] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets", in *Advances in Neural Information Processing Systems 27*, 2014, pp. 2672–2680.

[6] A. Graves, G. Wayne, and I. Danihelka, "Neural turing machines", *CoRR*, vol. abs/1410.5401, 2014.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition", *CoRR*, vol. abs/1512.03385, 2015.

[8] G. Hinton, N.Srivastava, A.Krizhevsky, I.Sutskever, and R.Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors", *CoRR*, vol. abs/1207.0580, 2012.

[9] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift", *CoRR*, vol. abs/1502.03167, 2015.

[10] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition", 2015.

[11] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks", *Neural Information Processing Systems*, vol. 25, 2012.

[12] B. M. Lake, R. Salakhutdinov, J. Gross, and J. B. Tenenbaum, *One shot learning of simple visual concepts*, 2011.

[13] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, "Human-level concept learning through probabilistic program induction", *Science*, vol. 350, no. 6266, pp. 1332–1338, 2015.

[14] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, "The omniglot challenge: A 3-year progress report", *CoRR*, 2019.

[15] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning", *Nature*, vol. 521, pp. 436–44, 2015.

[16] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines vinod nair", vol. 27, 2010, pp. 807–814.

[17] A. Nichol, J. Achiam, and J. Schulman, "On first-order meta-learning algorithms", *CoRR*, vol. abs/1803.02999, 2018.

[18] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning", in *In International Conference on Learning Representations (ICLR)*, 2017.

[19] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain", *Psychological Review*, pp. 65–386, 1958.

[20] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. P. Lillicrap, "One-shot learning with memory-augmented neural networks", *CoRR*, vol. abs/1605.06065, 2016.

[21] P. Y. Simard, D. Steinkraus, and J. C. Platt, "Best practices for convolutional neural networks applied to visual document analysis", in *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.*, 2003, pp. 958–963.

[22] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition", *CoRR*, 2014.

[23] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical networks for few-shot learning", English, *CoRR*, vol. abs/1703.05175, 2017.

[24] Q. Sun, Y. Liu, T. Chua, and B. Schiele, "Meta-transfer learning for few-shot learning", *CoRR*, vol. abs/1812.02391, 2018.

[25] O. Vinyals, C. Blundell, T. P. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning", *CoRR*, 2016.

[26] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions", *CoRR*, vol. abs/1511.07122, 2015.