Εθνικο Μετσοβιο Πολυτεχνειο

Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων

Εργαστηριο Συστηματων Τεχνητης Νοημοσυνης και Μαθησης

Techniques to accelerate the adoption of Federated Machine Learning for heterogeneous environments

Diploma Thesis

by

Christos D. Ntokos

Επιβλέπων: Αθανάσιος Βουλόδημος
Επ. Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2023

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Πληροφορικής
Εργαστήριο Συστημάτων Τεχνητής Νοημοσύνης και Μάθησης

Techniques to accelerate the adoption of Federated Machine Learning for heterogeneous environments

Diploma Thesis

by

Christos D. Ntokos

Επιβλέπων: Αθανάσιος Βουλόδημος
Επ. Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 13η Ιουλίου, 2023.

.....	.....	.....
Αθανάσιος Βουλόδημος	Γεώργιος Στάμου	Νικόλαος Δουλάμης
Επ. Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2023

.....

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

Περίληψη

Σε αυτή τη διπλωματική εργασία, πραγματοποιούμε μια εις βάθος εξερεύνηση μιας μεθοδολογίας μηχανικής μάθησης με γνώμονα την διατήρηση απορρήτου, και συγκεκριμένα την Ομοσπονδιακή Μηχανική Μάθηση (ΟΜΜ). Χαρακτηριζόμενο ως πρότυπο κατανεμημένης μηχανικής μάθησης, η ΟΜΜ αντιμετωπίζει τις βασικές προκλήσεις της Μηχανικής Μάθησης (ΜΜ) και των δεδομένων, όπως η ετερογένεια δεδομένων, η προστασία του απορρήτου και η ιδιοκτησία δεδομένων. Παρέχει μια πλατφόρμα που επιτρέπει στους οργανισμούς να συμβάλουν από κοινού στην ανάπτυξη μοντέλων, διατηρώντας πλήρη κυριότητα στα δεδομένα τους. Αυτό το χαρακτηριστικό το καθιστά εξαιρετικά ωφέλιμο σε περιπτώσεις όπου τα δεδομένα είναι είτε ευαίσθητα είτε ογκώδη, καθιστώντας τα ανέφικτα για κεντρική συλλογή.

Αυτή η διπλωματική εργασία στοχεύει να διερευνήσει τις δυνατότητες και τις περιπλοκές της χρήσης της Ομοσπονδιακής Μηχανικής Μάθησης σε πρακτικές εφαρμογές. Χρησιμοποιήσαμε ένα ευρέως αναγνωρισμένο λογισμικό, το Flower, για να δημιουργήσουμε μια λύση που έχει σχεδιαστεί για τον εξορθολογισμό των προσομοιώσεων ΟΜΜ. Έχουμε επίσης εφαρμόσει διαφορετικές τεχνικές χρησιμοποιώντας FedAVG και FEDMA για να αξιολογήσουμε την αποτελεσματικότητα. Επιπλέον, έχουμε εμβαθύνει στην εξέταση της ετερογένειας δεδομένων και υλικού, καθώς και στην αξιολόγηση των πελατών που εγκαταλείπουν και καθυστερούν, για να παρέχουμε μια ολιστική κατανόηση των προκλήσεων και των μεταβλητών που σχετίζονται με αυτήν την κατανεμημένη προσέγγιση μηχανικής μάθησης. Τα αποτελέσματα αυτής της μελέτης ρίχνουν φως στις ευέλικτες εφαρμογές και την πολυπλοκότητα της ΟΜΜ, υπογραμμίζοντας την αξία της για μελλοντική έρευνα και υλοποιήσεις στον πραγματικό κόσμο.

Για να συμπληρώσουμε την έρευνά μας, δημιουργήσαμε ένα φιλικό προς τον χρήστη εργαλείο σχεδιασμένο για να επιταχύνει και να απλοποιήσει την εκτέλεση προσομοιώσεων. Αυτό το εργαλείο ενσωματώνει βασικά ερευνητικά ευρήματα και ανταποκρίνεται αποτελεσματικά στην πολυπλοκότητα της ΟΜΜ, όπως η ετερογένεια δεδομένων και υλικού. Έτσι, στοχεύουμε να συνεισφέρουμε στην έρευνα στην Ομοσπονδιακή Μηχανική Μάθηση, διευκολύνοντας την πρακτική εφαρμογή της σε διάφορα σενάρια πραγματικού κόσμου.

Λέξεις-κλειδιά — Μηχανική μάθηση, Ομοσπονδιακή μηχανική μάθηση, διατήρηση απορρήτου, ετερογένεια δεδομένων, ετερογένεια υλικού, Flower framework, FedAVG, FEDMA

Abstract

In this thesis, we undertake an in-depth exploration of a privacy-conscious machine learning methodology, namely Federated Machine Learning (FML). Characterized as a distributed machine learning paradigm, FML addresses core Artificial Intelligence (AI) and data-related challenges such as data heterogeneity, privacy protection, and data ownership. It provides a platform for organizations to jointly contribute to model development, maintaining full authority over their data. This feature makes it exceptionally beneficial in instances where data is either sensitive or voluminous, rendering it impractical for centralized collection.

This diploma thesis aims to investigate the potential and intricacies of employing Federated Machine Learning (FML) for advanced research and practical applications. We have used a widely recognized framework, Flower, to craft a solution designed to streamline FML simulations. The aspects considered in our study include various parameters such as the number of clients, rounds, and epochs. We have also implemented different techniques using FedAVG and FEDMA to assess their effectiveness within the FML context. Furthermore, we have delved into the examination of data and hardware heterogeneity, as well as the evaluation of client dropouts and strugglers, to provide a holistic understanding of the challenges and variables associated with this distributed machine learning approach. The outcomes of this study shed light on the versatile applications and complexities of FML, underscoring its value for future research and real-world implementations.

To complement our research, we have created a user-friendly tool designed to expedite and simplify the execution of FML simulations. This tool integrates key research findings and effectively navigates the complexities of FML, such as data and hardware heterogeneity, client dropouts, and strugglers. With this development, we aim to stimulate further research in Federated Machine Learning, easing its practical application across diverse real-world scenarios.

Keywords — Machine Learning, Federated Machine Learning, Privacy preservation, Data heterogeneity, Hardware heterogeneity, Flower framework, FedAVG, FEDMA

Ευχαριστίες

Θα ήθελα να εκφράσω τη βαθύτατη εκτίμηση μου στον επιβλέποντα καθηγητή μου, κ. Αθανάσιο Βουλοδήμο, του οποίου η ανεκτίμητη καθοδήγηση και η τεχνογνωσία συνέβαλαν καθοριστικά στη διαμόρφωση αυτής της έρευνας.

Η ειλικρινής μου ευγνωμοσύνη απευθύνεται και στον υποψήφιο διδάκτορά, Σταύρο Συκιώτη. Η διαρκής υπομονή του, τα εποικοδομητικά του σχόλια και η βαθιά γνώση στον τομέα αποτέλεσαν ουσιαστική πυξίδα σε όλο αυτό το ταξίδι.

Θα ήθελα επίσης να ευχαριστήσω επίσης τους κ. Νικόλαο Δουλάμη και κ. Βασίλειο Βεσκούκη για την παροχή των απαραίτητων πόρων και ευκαιριών για την έρευνά μου. Η βοήθεια και η υποστήριξη που έλαβα ήταν καθοριστικής σημασίας για την ολοκλήρωση αυτής της μελέτης.

Τέλος, ευχαριστώ από καρδιάς την οικογένειά μου, τους φίλους και τους συναδέλφους μου, των οποίων η αμέριστη συμπαράσταση και ενθάρρυνση ήταν οι ακλόνητοι σύντροφοί μου σε όλη τη διάρκεια της ακαδημαϊκής μου διαδρομής.

Χρήστος Ντόκος, Ιούλιος 2023

I dedicate this work to my extraordinary mother, whose boundless patience and persistent faith in me have been my guiding lights throughout this journey.

Αφιερώνω αυτό το έργο στην εξαιρετική μητέρα μου, της οποίας η απεριόριστη υπομονή και η επίμονη πίστη σε μένα υπήρξαν τα φώτα καθοδήγησής μου σε αυτό το ταξίδι.

Contents

List of Figures

Chapter 1

Εκτεταμένη Περίληψη στα Ελληνικά

1.1 Εισαγωγή

1.1.1 Η αυξανόμενη επιρροή των μεγάλων δεδομένων και της Τεχνητής Νοημοσύνης

Τα "μεγάλα δεδομένα", που αντιπροσωπεύουν μεγάλα και σύνθετα σύνολα δεδομένων, έχουν συγκεντρώσει αυξημένη προσοχή σε όλους τους κλάδους λόγω των προόδων στις τεχνολογίες αποθήκευσης και διαχείρισης δεδομένων. Τα μεγάλα δεδομένα προσφέρουν πολύτιμες πληροφορίες για τις τάσεις και τις συμπεριφορές σε τομείς όπως η υγεία, η οικονομία και το εμπόριο. Ωστόσο, με την επέκτασή του, προκύπτουν νέες προκλήσεις, ιδιαίτερα προκλήσεις που σχετίζονται με την ιδιωτικότητα.

Παρά τις εκτενείς μελέτες για την προστασία της ιδιωτικής ζωής, το τεράστιο μέγεθος των παραγώμενων δεδομένων καθιστά τις παραδοσιακές κρυπτογραφικές λύσεις λιγότερο αποτελεσματικές. Δεδομένων των περιορισμών χωρητικότητας των συσκευών και της υποκειμενικότητας του ορισμού του απορρήτου, η ανάπτυξη νέων αλγορίθμων και πλαισίων απορρήτου είναι υψίστης σημασίας. Ακόμη και όταν χρησιμοποιούνται τεχνικές ενίσχυσης του απορρήτου όπως η k-anonimity, η t-closeness, η l-diversity, η εξελισσόμενη φύση των επιθέσεων απορρήτου μεγάλων δεδομένων απαιτεί συνεχή έρευνα.

Η Τεχνητή Νοημοσύνη και τα Μεγάλα Δεδομένα μοιράζονται μια στενή σχέση, με την ΤΝ που χρησιμοποιείται συχνά για τη διευκόλυνση της σύλληψης, της δομής και της ανάλυσης των Μεγάλων Δεδομένων. Η ΤΝ, χρησιμοποιώντας μηχανική μάθηση και άλλες τεχνικές, επιτρέπει την αποτελεσματική επεξεργασία και ανάλυση μεγάλων συνόλων δεδομένων, βοηθώντας στον εντοπισμό μοτίβων, σχέσεων και προβλέψεων που θα μπορούσαν να χαθούν από τη μη αυτόματη ανάλυση. Η ικανότητα αυτοματοποίησης της ΤΝ βοηθά επίσης τους οργανισμούς να επικεντρωθούν σε κρίσιμα δεδομένα και να αποκτήσουν γρήγορα γνώσεις.

Η Τεχνητή Νοημοσύνη ενέχει επίσης κινδύνους, όπως η διαιώνιση των προκαταλήψεων, η απώλεια θέσεων εργασίας και οι ανήθικες αποφάσεις. Έτσι, οι ηθικοί προβληματισμοί

έχουν προσελκύσει αυξημένη προσοχή. Παρά τα πιθανά μειονεκτήματα της τεχνητής νοημοσύνης, τα οφέλη της, συμπεριλαμβανομένης της βελτιωμένης υγειονομικής περίθαλψης, των βελτιστοποιημένων συστημάτων μεταφορών και της βελτιωμένης ασφάλειας στον κυβερνοχώρο, είναι σημαντικά. Για να επιτευχθεί ισορροπία, τα συστήματα τεχνητής νοημοσύνης θα πρέπει να αναπτύσσονται και να χρησιμοποιούνται με υπευθυνότητα, διασφαλίζοντας διαφάνεια, επεξηγματικότητα και υπευθυνότητα. Οι ηθικές ανησυχίες περιλαμβάνουν θέματα όπως η έλλειψη εμπιστοσύνης, η ακρίβεια των δεδομένων, το απόρρητο, η διαφάνεια, η μεροληψία, η κακή χρήση προσωπικών δεδομένων και η έλλειψη λογοδοσίας, μεταξύ άλλων.

Στο πλαίσιο της Τεχνητής Νοημοσύνης, είναι σημαντικό να γίνει διάκριση μεταξύ της ασφάλειας δεδομένων και του απορρήτου δεδομένων. Η ασφάλεια δεδομένων εστιάζει στην πρόληψη μη εξουσιοδοτημένης πρόσβασης και παραβιάσεων δεδομένων μέσω μέτρων όπως η κρυπτογράφηση και οι έλεγχοι πρόσβασης. Από την άλλη πλευρά, το απόρρητο των δεδομένων επικεντρώνεται γύρω από το ποιος μπορεί να έχει πρόσβαση σε δεδομένα, την ηθική επεξεργασία πληροφοριών προσωπικής ταυτοποίησης και τον έλεγχο των ατομικών δεδομένων. Ενώ οι ανησυχίες για την ασφάλεια είναι καλά μελετημένες, οι εγγυήσεις απορρήτου έχουν αντιμετωπίσει αρκετές αξιοσημείωτες αποτυχίες.

1.1.2 Αρχιτεκτονικές Μηχανικής Μάθησης

Οι εξελίξεις στην ΤΝ και τη μηχανική μάθηση έχουν ενισχυθεί από ανακαλύψεις σε τομείς όπως το cloud computing και το edge computing. Αυτό επέτρεψε στην μηχανική μάθηση να λειτουργεί αποκεντροποιημένα, όπου παράγονται δεδομένα, επιτρέποντας την αποτελεσματική εξαγωγή συμπερασμάτων στο σημείο προέλευσης. Η κατανεμημένη μηχανική μάθηση και η διασπορά του υπολογισμού σε πολλούς κόμβους, σηματοδοτούν μια μετατόπιση από το κεντρικό μοντέλο, μειώνοντας την καθυστέρηση και ενισχύοντας την επεκτασιμότητα. Ωστόσο, δεν διασφαλίζουν πλήρως το απόρρητο των δεδομένων καθώς εξακολουθεί να απαιτεί κοινή χρήση δεδομένων μεταξύ των κόμβων.

Για την αντιμετώπιση αυτής της ανησυχίας, εισήχθη η Ομοσπονδιακή Μηχανική Μάθηση. Επιτρέπει την εκπαίδευση μοντέλων, τη δοκιμή και την εξαγωγή συμπερασμάτων αποκεντροποιημένα, διατηρώντας τα δεδομένα στην πηγή τους και, ως εκ τούτου, διατηρώντας το απόρρητο από το σχεδιασμό.

1.1.3 Συνεισφορές

Στην εργασία αυτή παρέχεται μια διεξοδική διερεύνηση της Ομοσπονδιακής Μηχανικής Μάθησης, εξηγώντας τις αρχές της, τις βασικές συνεισφορές της στην Μηχανική Μάθηση, αλλά και τις δυνατότητες της με σκοπό την επίλυση ζητημάτων απορρήτου δεδομένων. Αναπτύχθηκε επίσης ένα νέο εργαλείο προσομοίωσης βασισμένο στο λογισμικό Flower, το οποίο προσομοιώνει μια ποικιλία σεναρίων Ομοσπονδιακής Μηχανικής Μάθησης, λαμβάνοντας υπόψη την ετερογένεια των δεδομένων και του υλικού.

1.1.4 Σχετική εργασία

Οι λύσεις Ομοσπονδιακής Μάθησης χωρίζονται ευρέως σε βιβλιοθήκες και σε λειτουργικά εμπορικά συστήματα. Οι βιβλιοθήκες προσφέρουν ένα ευέλικτο σύνολο εργαλείων για το σχεδιασμό, την υλοποίηση και την εκτέλεση αλγορίθμων FL, αλλά απαιτούν εξειδίκευση και πρόσθετη υποδομή για τη διαχείριση και την παρακολούθηση των ροών εργασίας. Αντίθετα, τα συστήματα προσφέρουν ολοκληρωμένες λύσεις που χειρίζονται όλες τις πτυχές της ροής εργασίας [45, 8, 22, 29].

1.2 Μηχανική Μάθηση Διατήρησης Απορρήτου

Η μηχανική εκμάθηση που διαφυλάσσει το απόρρητο είναι ένα υποπεδίο της μηχανικής μάθησης που στοχεύει στην προστασία των ευαίσθητων πληροφοριών εντός των δεδομένων, ενώ επιτρέπει την αποτελεσματική και ακριβή εκπαίδευση και προβλέψεις μοντέλων. Αυτό περιλαμβάνει την ανάπτυξη μεθόδων και τεχνικών που προστατεύουν τα δεδομένα από μη εξουσιοδοτημένη πρόσβαση ή εξαγωγή. Ορισμένες δημοφιλείς μέθοδοι περιλαμβάνουν το διαφορικό απόρρητο [19], το οποίο προσθέτει θόρυβο στα δεδομένα ή στους υπολογισμούς για να διασφαλίσει ότι το αποτέλεσμα ενός αλγορίθμου μηχανικής εκμάθησης δεν αποκαλύπτει συγκεκριμένες λεπτομέρειες για οποιοδήποτε μεμονωμένο σημείο δεδομένων. Μια άλλη προσέγγιση είναι η ομοσπονδιακή εκμάθηση, η οποία διατηρεί δεδομένα σε τοπικές συσκευές και μοιράζεται μόνο ενημερώσεις μοντέλων για συγκέντρωση. Τεχνικές όπως η ομομορφική κρυπτογράφηση [3] και οι αποδείξεις μηδενικής γνώσης χρησιμοποιούνται επίσης για την εκτέλεση υπολογισμών απευθείας σε κρυπτογραφημένα δεδομένα. Η μηχανική εκμάθηση που διαφυλάσσει το απόρρητο είναι ιδιαίτερα σημαντική σε τομείς όπως η υγειονομική περίθαλψη ή τα οικονομικά, όπου η εμπιστευτικότητα των δεδομένων είναι απαραίτητη.

1.2.1 Επισκόπηση Ομοσπονδιακής Μάθησης

Η ομοσπονδιακή εκμάθηση είναι μια τεχνική μηχανικής εκμάθησης που διατηρεί το απόρρητο και επιτρέπει την αποκεντρωμένη εκπαίδευση μοντέλων σε συσκευές, χωρίς να μοιράζονται ακατέργαστα δεδομένα.

Η βασική ομοσπονδιακή αρχιτεκτονική περιλαμβάνει συμμετέχοντες κόμβους (συσκευές που συνεισφέρουν δεδομένα), έναν κεντρικό διακομιστή (συντονίζει τη διαδικασία εκμάθησης), ένα πρωτόκολλο επικοινωνίας (ασφαλίζει την ανταλλαγή δεδομένων) και αλγόριθμους εκμάθησης (βελτιστοποίηση του συστήματος) [40].

1.2.2 Τύποι Ομοσπονδιακής Μάθησης

Η ομοσπονδιακή μάθηση κατηγοριοποιείται σε τύπους πολλαπλών συσκευών και διαφορετικών σιλό με βάση τη φύση και τις δυνατότητες των διαθέσιμων πόρων των συμμετεχόντων οντοτήτων. 1.2.1 Στην ομόσπονδη μάθηση με πολλαπλά σιλό, τα δεδομένα που χωρίζονται σε διαφορετικούς οργανισμούς χρησιμοποιούνται για την εκπαίδευση μοντέλων. Είναι κατάλληλο όταν η συγκέντρωση δεδομένων δεν είναι εφικτή λόγω απορρήτου ή ρυθμιστικούς λόγους.

Η ομοσπονδιακή μάθηση μεταξύ συσκευών, από την άλλη πλευρά, εξυπηρετεί τα δεδομένα που είναι αποθηκευμένα σε συσκευές χρηστών. Κάθε συσκευή εκπαιδεύει το δικό της μοντέλο, στέλνοντας μόνο ενημερώσεις σε έναν κεντρικό διακομιστή, ενισχύοντας έτσι το απόρρητο και την ασφάλεια των δεδομένων.

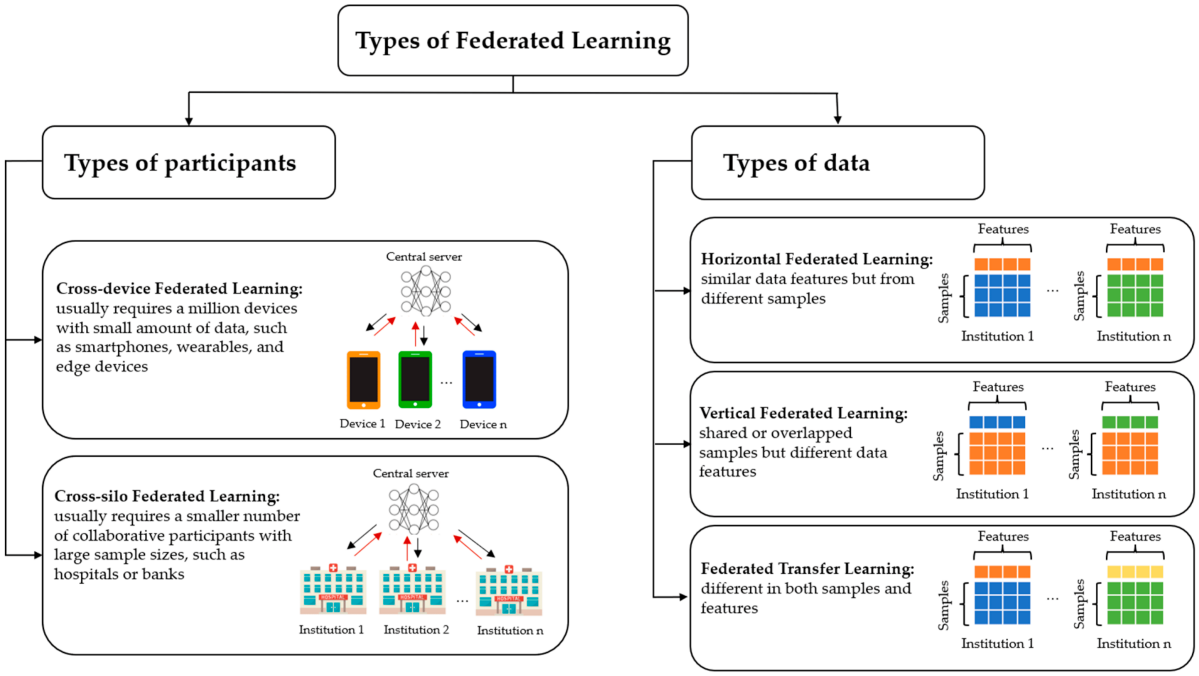


Figure 1.2.1: Τύποι Ομοσπονδιακής Μάθησης

Επιπλέον, η ομοσπονδιακή μάθηση μπορεί να ταξινομηθεί σε κάθετους, οριζόντιους και τύπους μεταφοράς ανάλογα με τη διανομή δεδομένων.

Κάθετη Ομοσπονδιακή Μάθηση: Χρησιμοποιείται όταν διαφορετικές οντότητες έχουν διαφορετικά χαρακτηριστικά για το ίδιο σύνολο δειγμάτων, επιτρέποντας τη συλλογική δημιουργία μοντέλων χωρίς κοινή χρήση πρωτογενών δεδομένων.

Οριζόντια Ομοσπονδιακή Μάθηση: Ισχύει όταν διαφορετικές οντότητες έχουν τα ίδια χαρακτηριστικά για διαφορετικά άτομα, επιτρέποντας τη συμβολή στη μάθηση χωρίς να μοιράζονται προσωπικά δεδομένα.

Μεταφορά Ομοσπονδιακής μάθησης: Περιλαμβάνει μοντέλα εκπαίδευσης σε έναν οργανισμό και μεταφορά σε άλλον για τελειοποίηση, που χρησιμοποιείται συχνά όταν ο οργανισμός-στόχος δεν έχει επαρκή δεδομένα.

1.2.3 Αλγόρθμοι

Ο ομοσπονδιακός μέσος όρος, ή FedAvg [40], είναι ένας αλγόριθμος στην Ομοσπονδιακή Μάθηση για τη δημιουργία μοντέλων μηχανικής εκμάθησης χρησιμοποιώντας δεδομένα σε

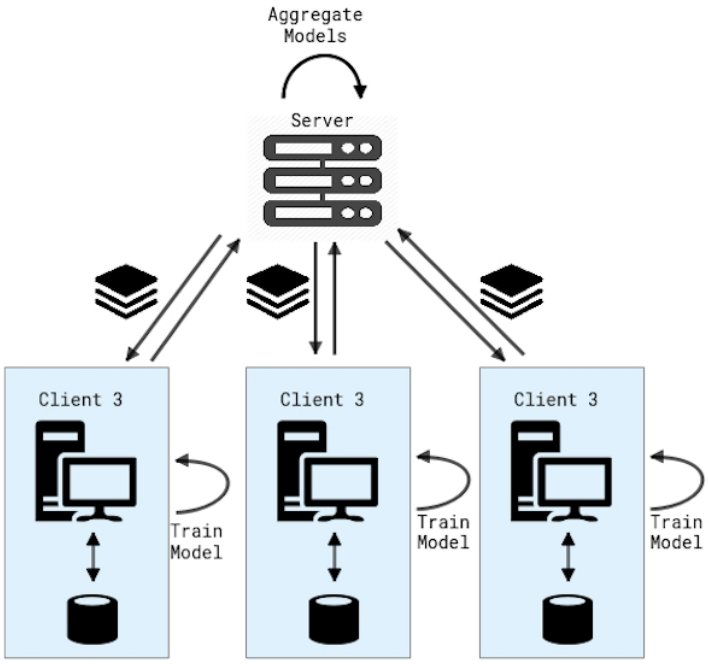


Figure 1.2.2: Ομοσπονδιακή Μηχανικής Μάθηση

απομακρυσμένη συσκευή, αντί για εκπαίδευση που βασίζεται σε κεντρικούς διακομιστές, βελτιώνοντας το απόρρητο των δεδομένων.

Τα βήματα του FedAvg είναι:

1. Ένα καθολικό μοντέλο προετοιμάζεται στον διακομιστή με τυχαία βάρη.
2. Το καθολικό μοντέλο μεταδίδεται σε ένα κλάσμα των συνολικών συσκευών που συμμετέχουν για τοπική εκπαίδευση.
3. Κάθε συσκευή εκπαιδεύει το μοντέλο στα τοπικά της δεδομένα χωρίς να μεταφέρει ακατέργαστα δεδομένα, διατηρώντας το απόρρητο.
4. Μετά την τοπική εκπαίδευση, κάθε συσκευή υπολογίζει μια ενημέρωση του μοντέλου και την στέλνει στον διακομιστή.
5. Ο διακομιστής συγκεντρώνει τις ενημερώσεις, συνήθως με υπολογισμό του μέσου όρου, για να ενημερώσει το καθολικό μοντέλο.
6. Τα βήματα 2-5 επαναλαμβάνονται μέχρι το μοντέλο να συγκλίνει σε ικανοποιητική ακρίβεια.

Ενώ το FedAvg είναι ιδανικό για εφαρμογές όπου το απόρρητο δεδομένων αποτελεί πρόβλημα ή η κεντρική ταξινόμηση δεδομένων δεν είναι πρακτική, εξακολουθεί να αντιμετωπίζει ζητήματα όπως ο χειρισμός δεδομένων που δεν είναι IID και συσκευών που έχουν ποικίλους όγκους δεδομένων. Από την άλλη, ο αλγόριθμος FedMA [54], είναι μια

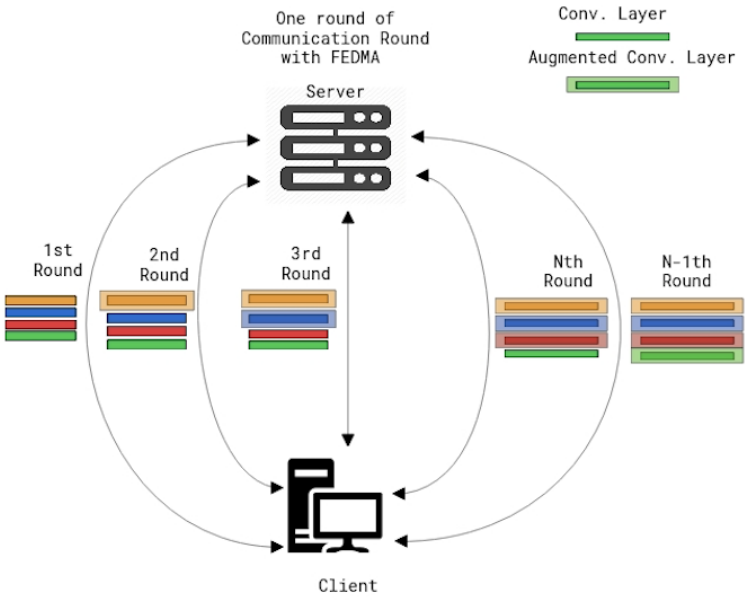


Figure 1.2.3: Γύρος FEDMA

βελτίωση σε σχέση με τον αλγόριθμο FedAvg. Σε αντίθεση με το FedAvg, το FedMA δεν συγχωνεύει αφελώς τις παραμέτρους του μοντέλου, διατηρώντας έτσι μοναδικά χαρακτηριστικά σε δεδομένα που δεν είναι IID.

Η FedMA χρησιμοποιεί την τεχνική "αντίστοιχου μέσου όρου" και δημιουργεί ένα παγκόσμιο κοινόχρηστο μοντέλο, αντιστοιχίζοντας και υπολογίζοντας τον μέσο όρο των κρυφών στοιχείων του μοντέλου, όπως τα κανάλια των CNN, οι κρυφές καταστάσεις των LSTM ή οι νευρώνες των πλήρως συνδεδεμένων επιπέδων. Κάθε γύρος Ομοσπονδιακής Μηχανικής Εκμάθησης στο FEDMA περιλαμβάνει γύρους αντιστοίχισης. Οι συσκευές εκπαιδεύουν το μοντέλο σε τοπικά δεδομένα και το στέλνουν πίσω στον διακομιστή, όπου ένα συγκεκριμένο επίπεδο αντιστοιχίζεται και επαυξάνεται κάθε φορά (Εικόνα 4.2.2). Στη συνέχεια, οι συσκευές εκπαιδεύουν το υπόλοιπο μοντέλο με το επαυξημένο στρώμα παγωμένο. Το τελικό μοντέλο διατηρεί πληροφορίες από το μοντέλο κάθε συσκευή.

Για τη διευκόλυνση του FEDMA, χρησιμοποιείται το πλαίσιο Flower, το οποίο λειτουργεί σε δύο γύρους: αντιστοίχισης και τυπικούς γύρους FML. Ο διακομιστής και οι συσκευές συντονίζονται για να εκπαιδεύσουν, να ταιριάξουν και να αυξήσουν τα μοντέλα σε αυτούς τους γύρους.

1.3 Προκλήσεις Ομοσπονδιακής Μηχανικής Εκμάθησης

Η Ομοσπονδιακή Μηχανική Μάθηση ενσωματώνει πολυπλοκότητες Τεχνητής Νοημοσύνης, οι οποίες γίνονται πιο περίπλοκες λόγω της κατανεμημένης φύσης των συστημάτων.

Περιλαμβάνουν:

1. **Δεδομένα non-IID:** Τα δεδομένα δεν διανέμονται ανεξάρτητα και πανομοιότυπα στις απομακρυσμένες συσκευές.
2. **Ετερογένεια δεδομένων:** Το μέγεθος, η κατανομή και η ποιότητα των δεδομένων μπορεί να διαφέρουν μεταξύ των συσκευές.
3. **Ασύγχρονη επικοινωνία:** Οι συσκευές ενδέχεται να μην είναι πάντα διαθέσιμες ή να έχουν διαφορετικούς υπολογιστικούς πόρους.
4. **Διαφορετική Συμμετοχή συσκευών:** Δεν επιτρέπεται να συμμετέχουν όλες οι συσκευές σε κάθε γύρο.
5. **Data Drift:** Η κατανομή δεδομένων ενδέχεται να αλλάξει με την πάροδο του χρόνου, επηρεάζοντας την ακρίβεια του μοντέλου.
6. **Επιβάρυνση επικοινωνίας:** Οι συχνές ενημερώσεις μοντέλων μεταξύ διακομιστή και συσκευών μπορεί να είναι δαπανηρές.
7. **Αντίπαλοι:** Οι συσκευές ενδέχεται να εισάγουν σκόπιμα εσφαλμένες ενημερώσεις ή κακόβουλα δεδομένα.
8. **Σφάλματα επικοινωνίας και θόρυβος:** Η πραγματική επικοινωνία μεταξύ συσκευών και διακομιστή μπορεί να είναι επιρρεπής σε σφάλματα ή θόρυβο.
9. **Συμπίεση και κβαντοποίηση μοντέλου:** Το περιορισμένο εύρος ζώνης επικοινωνίας μπορεί να απαιτεί συμπίεση και κβαντοποίηση του μοντέλου.
10. **Ετερογενείς αρχιτεκτονικές μοντέλων:** Οι ποικίλοι περιορισμοί ή οι προτιμήσεις υλικού μπορεί να οδηγήσουν σε διαφορετικές αρχιτεκτονικές μοντέλων μεταξύ των συσκευών.
11. **Επιθέσεις απορρήτου:** Παρά τους στόχους απορρήτου της ομοσπονδιακής μάθησης, επιθέσεις μπορούν ακόμα να συμβούν.

1.4 Αρχιτεκτονική Συστήματος

1.4.1 Διαγράμματα

Η υλοποίηση του συστήματος αποσκοπεί στην εύκολη μετάβαση από τοπική μηχανική μάθηση σε διακομιστή και έπειτα σε κατανεμημένο περιβάλλον. Τα διαγράμματα χρησιμοποιούνται για την απεικόνιση της δομής του συστήματος και των αλληλεπιδράσεων των στοιχείων.

Συστατικό Διάγραμμα

Το διάγραμμα 1.4.1 αποκαλύπτει ότι όλα τα στοιχεία λειτουργούν μέσα σε container για επεκτασιμότητα και δυνατότητα τροποποίησης. Τα container FML Manager και FML Client υποδηλώνουν τη σχέση διακομιστή-συσκευής. Η βάση δεδομένων Redis αποθηκεύει κρίσιμα δεδομένα εκπαίδευσης και πληροφορίες παρακολούθησης συστήματος.

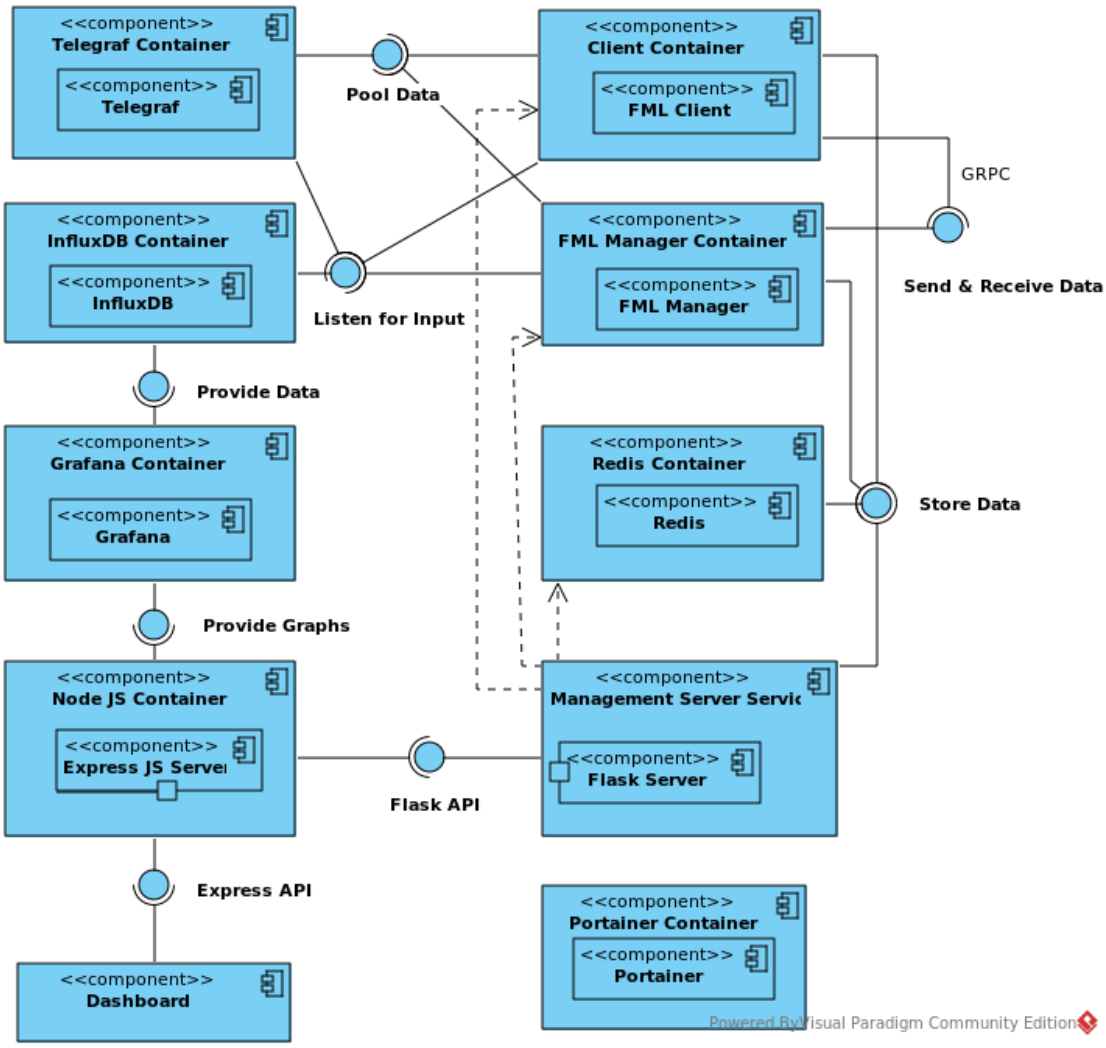


Figure 1.4.1: Συστατικό Διάγραμμα

Η Υπηρεσία Διακομιστή Διαχείρισης περιλαμβάνει ένα Flask API που εκκινεί το διαδικασία, διαχειρίζεται δίκτυα, και τις υπηρεσίες Docker. Το σύστημα περιέχει επίσης ένα στοιχείο παρακολούθησης και μια διεπαφή χρήστη για αναπαραγωγή προσομοίωσης και παρακολούθηση σε πραγματικό χρόνο.

Διαγράμματα υπηρεσιών REST

Για να βελτιώσουμε την επεκτασιμότητα και την ανθεκτικότητα του συστήματος, έχουμε ενσωματώσει API 1.4.2 σε έναν διακομιστή Flask. Αυτά τα API δημιουργούν ζωτικές υπηρεσίες και μιμούνται σενάρια ετερογένειας υλικού στην ομοσπονδιακή μηχανική μάθηση. Τα API μπορούν να εκκινήσουν υπηρεσίες «N», με κάθε λειτουργία «R» αντίγραφα container με ίδιους περιορισμούς πόρων. Αυτό επιτρέπει την εκτεταμένη επεκτασιμότητα και την εξομοίωση ενός πραγματικά ετερογενούς τοπίου υλικού, παρέχοντας πολύτιμες πληρο-

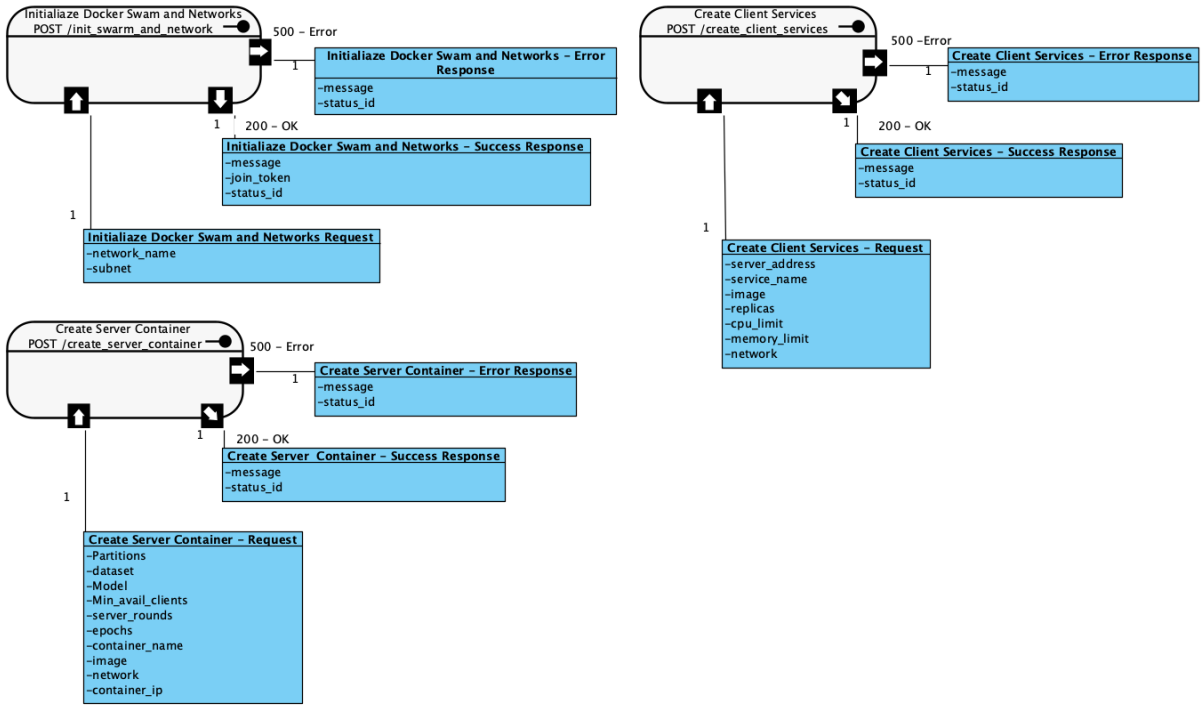


Figure 1.4.2: Διαγράμματα υπηρεσιών REST

φορίες για την απόδοση των ομοσπονδιακών μοντέλων μηχανικής εκμάθησης κάτω από διαφορετικές συνθήκες υλικού.

Διάγραμμα ακολουθίας

Το σχήμα 1.4.3 περιγράφει την ακολουθία του παραδείγματος της Ομοσπονδιακής Μηχανικής Μάθησης. Αρχικά, ένας χρήστης καθορίζει παραμέτρους μέσω φόρμας. Στη συνέχεια, αυτές οι πληροφορίες αποστέλλονται μέσω των API REST στον διακομιστή διαχείρισης, ο οποίος εκκινεί τον διακομιστή FML και ενεργοποιεί της απαραίτητες συσκευές.

Διάγραμμα ανάπτυξης

Το σχήμα 1.4.4 απεικονίζει την αρχιτεκτονική της λύσης μας, που αναπτύσσεται σε έναν μόνο διακομιστή με κάθε στοιχείο να λειτουργεί σε ένα ξεχωριστό container. Αυτή η δομή μιμείται ένα κατανεμημένο σύστημα και χρησιμεύει ως σκαλοπάτι προς μια πιο κατανεμημένη ρύθμιση, όπως αναπαρίσταται στο Σχήμα 1.4.5, που δείχνει τους συσκευές που λειτουργούν από ξεχωριστές συσκευές, κάτι που είναι απαραίτητο για την πραγματική ομοσπονδιακή μάθηση.

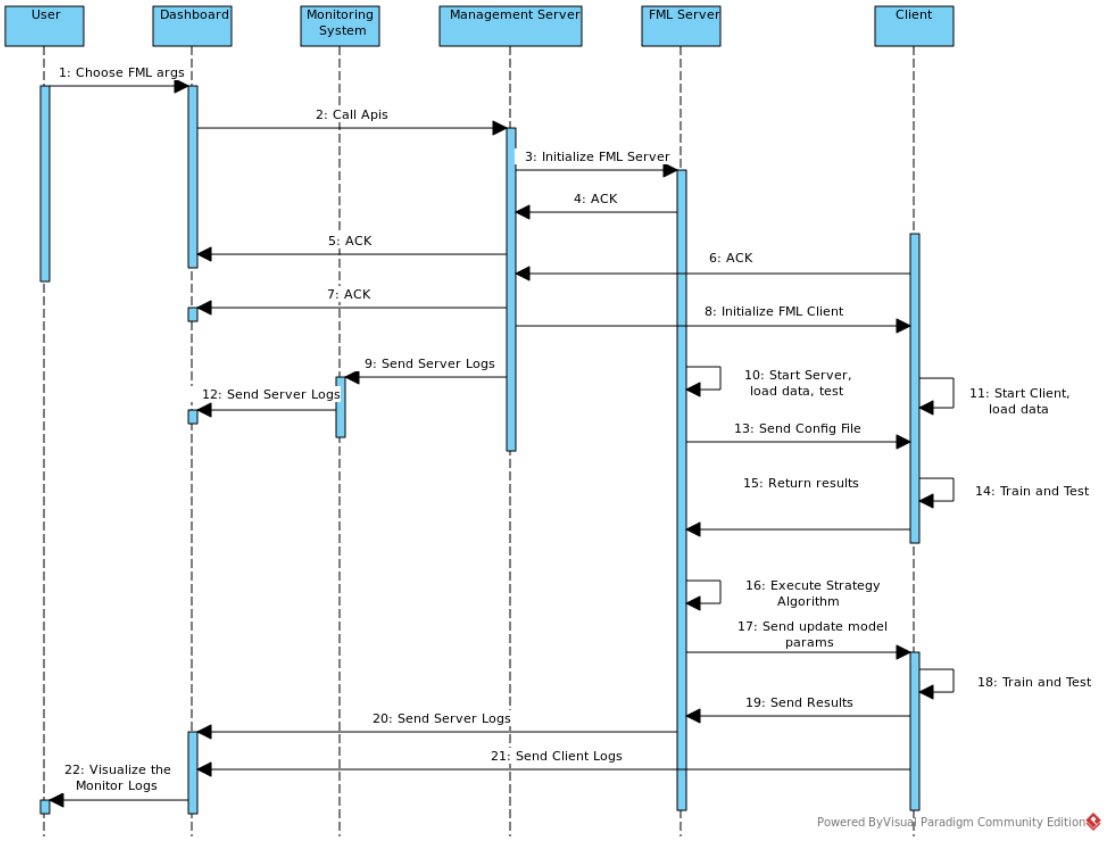


Figure 1.4.3: Διάγραμμα ακολουθίας

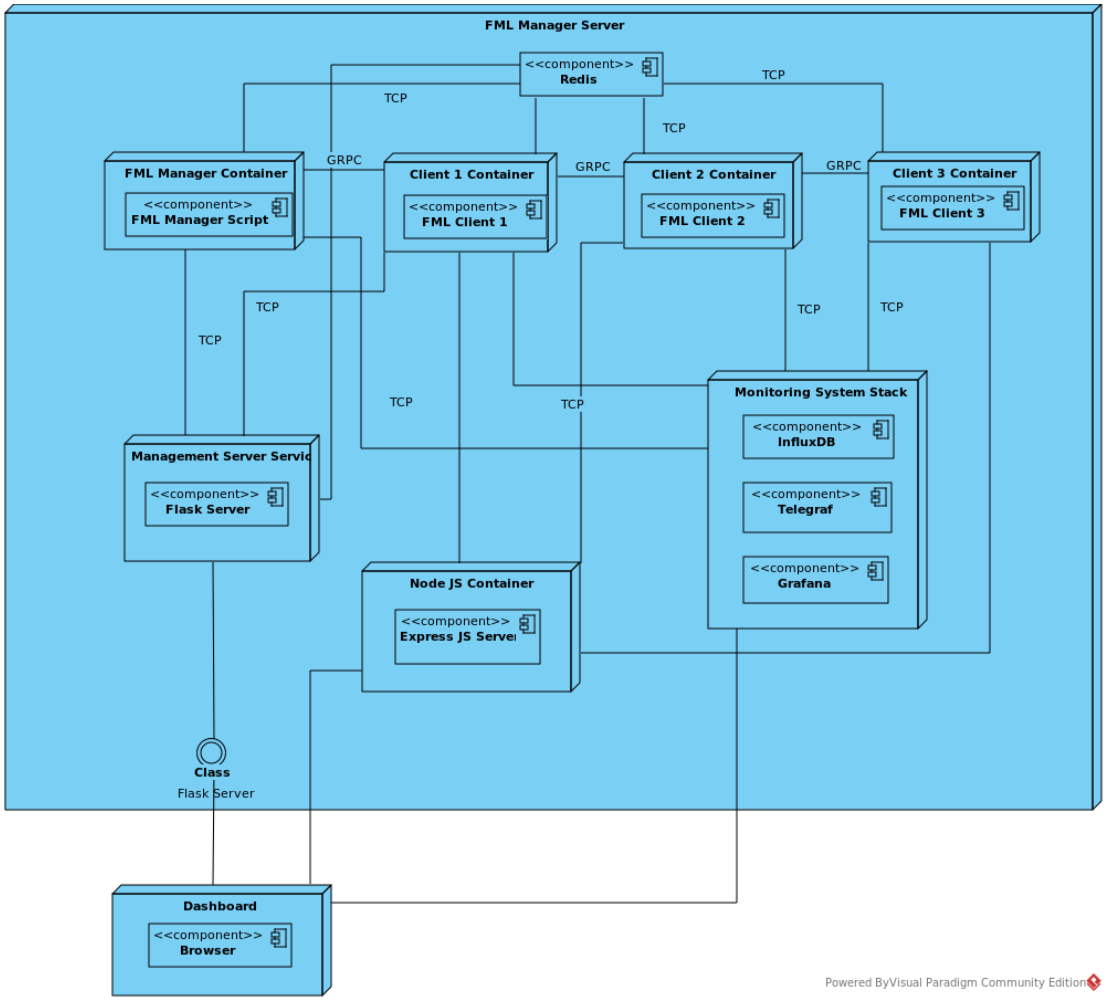


Figure 1.4.4: Διάγραμμα ανάπτυξης σε διακομιστή

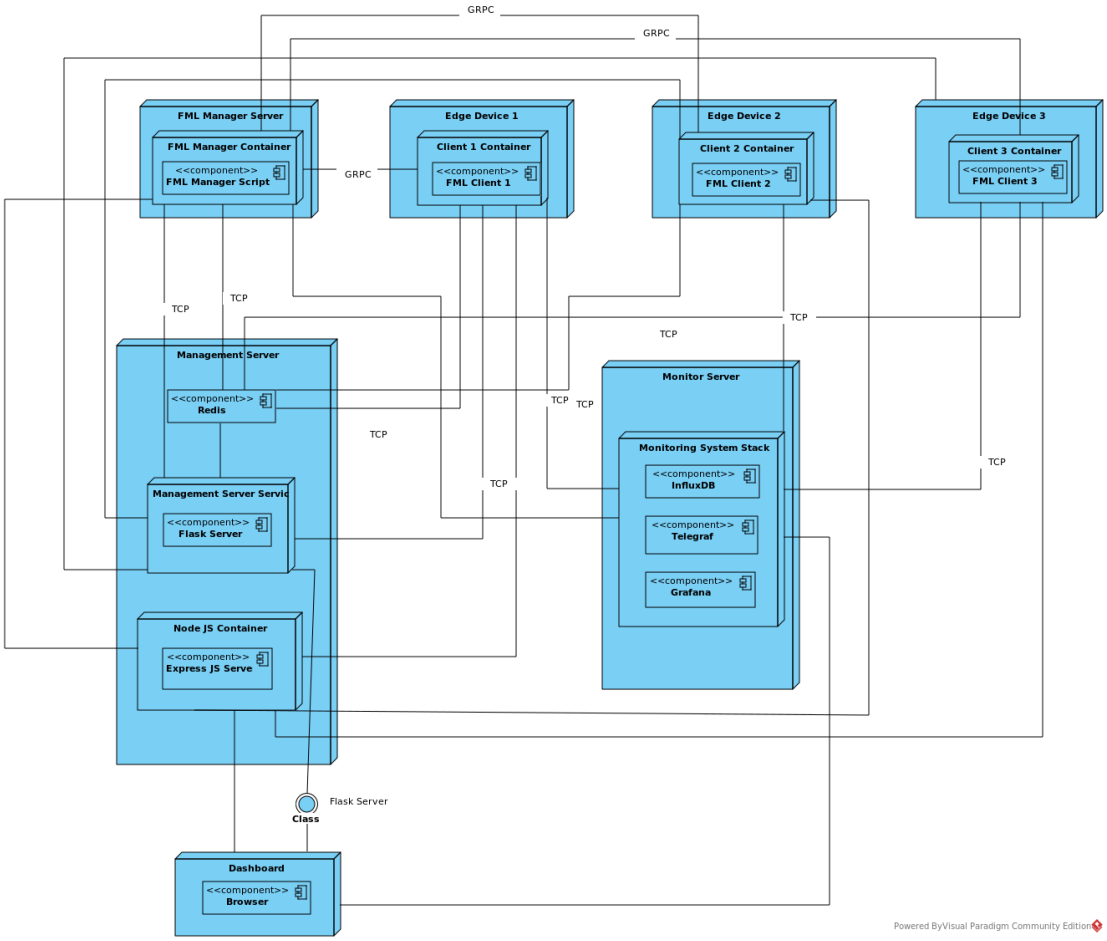


Figure 1.4.5: Αποκεντρωμένο διάγραμμα ανάπτυξης

Διάγραμμα τάξης

Η λύση μας χρησιμοποιεί το Flower Framework ως τη θεμελιώδη αρχιτεκτονική του. Περιλαμβάνει μαθήματα για τη διαχείριση συσκεύων, λειτουργίες διακομιστή, στρατηγικές εκπαίδευσης, καταγραφή δεδομένων και πολλά άλλα. Αυτές οι κλάσεις διευκολύνουν την προσαρμογή, την ευρωστία και τον πλήρη έλεγχο του παραδείγματος μας.

1.4.2 Παρακολούθηση

Το ομοσπονδιακό μας σύστημα εκμάθησης χρησιμοποιεί το Telegraf για συλλογή μετρήσεων, το InfluxDB ως βάση δεδομένων χρονοσειρών και το Grafana για την οπτικοποίηση δεδομένων. Αυτά τα εργαλεία επιτρέπουν την παρακολούθηση σε πραγματικό χρόνο της χρήσης της CPU, της χρήσης μνήμης και των συνδέσεων συσκευών. Τα δεδομένα από container και συσκευές Docker αποθηκεύονται στο InfluxDB και αντιπροσωπεύονται οπτικά στο Grafana, επιτρέποντας την αναγνώριση προβλημάτων σε πραγματικό χρόνο και βελτιώσεις στην απόδοση του συστήματος.

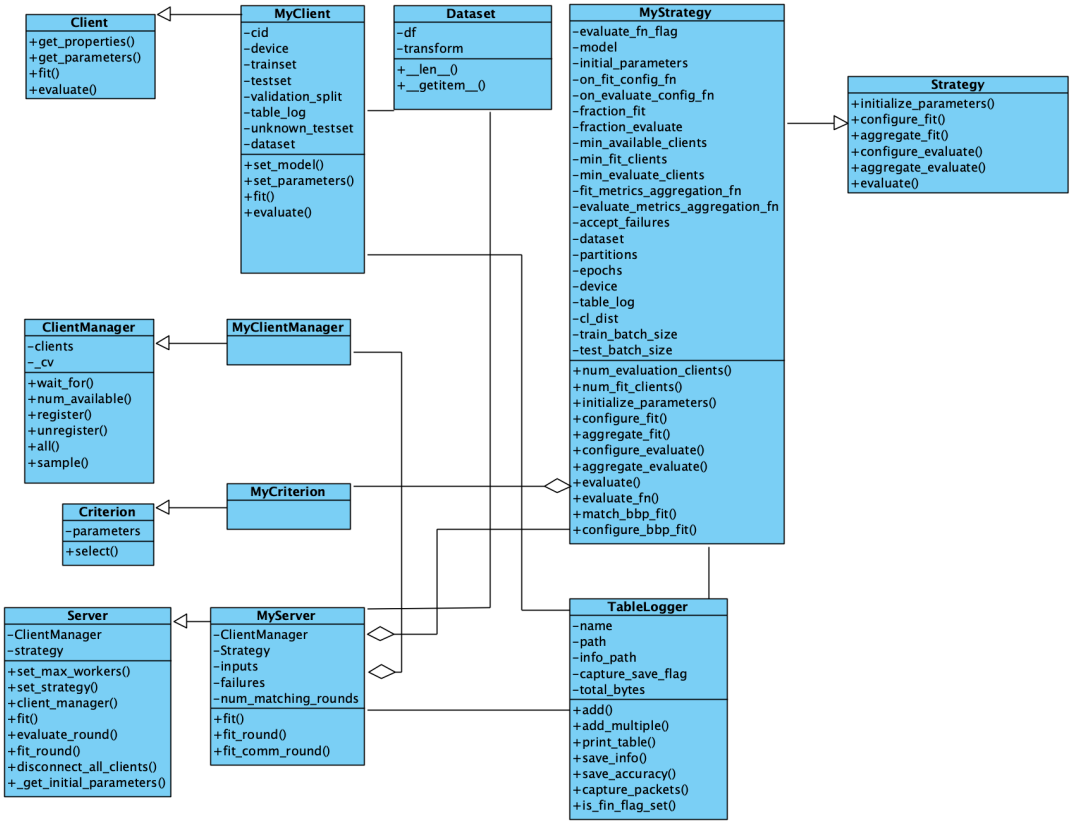


Figure 1.4.6: Διάγραμμα τάξης

1.4.3 Δρόμος προς την αποκεντροποίηση

Η κρίση αναπαραγωγιμότητας στην έρευνα Ομοσπονδιακής Μηχανικής Μάθηση, λόγω της ανεπαρκούς τεκμηρίωσης των στρατηγικών καταμερισμού δεδομένων και των υπερπαραμέτρων, εμποδίζει την επιστημονική πρόοδο. Αυτή η πρόκληση περιπλέκεται περαιτέρω από τη δυσκολία μετάβασης από τα θεωρητικά μοντέλα σε ζωντανές εφαρμογές.

Η λύση μας ενσωματώνει στοιχεία λογισμικού σε έναν πίνακα εργαλείων διεπαφής χρήστη, επιτρέποντας προσαρμόσιμες προσομοιώσεις και παρακολούθηση σε πραγματικό χρόνο της διαδικασίας. Ο πίνακας εργαλείων παρουσιάζει μετρήσεις που περιλαμβάνουν ακρίβεια μοντέλου, απώλεια εκπαίδευσης, χρήση CPU και χρήση μνήμης από συσκεύες και διακομιστές. Αυτό βελτιώνει την κατανόηση της διαδικασίας από τον χρήστη και προωθεί την αποτελεσματικότερη έρευνα.

Ο πίνακας εργαλείων διεπαφής χρήστη περιλαμβάνει μια φόρμα για την επιλογή παραμέτρων, η οποία επιτρέπει στους χρήστες να προσαρμόζουν τις προσομοιώσεις. Η παρακολούθηση σε πραγματικό χρόνο επιτρέπει την παρακολούθηση της ακρίβειας του μοντέλου, της απώλειας εκπαίδευσης και των αποτελεσμάτων δοκιμών σε διάφορα σύνολα δεδομένων.

1.5 Πειραματικά αποτελέσματα

1.5.1 Σύνολα δεδομένων

Το σύνολο δεδομένων MNIST [35] περιέχει 70.000 εικόνες σε κλίμακα του γκρι χειρόγραφων ψηφίων, χωρισμένες σε 60.000 εκπαιδευτικές και 10.000 δοκιμαστικές εικόνες. Κάθε εικόνα 28x28 pixel είναι κανονικοποιημένη και κεντραρισμένη για ομοιομορφία.

Το σύνολο δεδομένων FEMNIST, από το LEAF [14], επεκτείνει το MNIST συμπεριλαμβάνοντας εικόνες χειρόγραφων ψηφίων και χαρακτήρων από 3.550 συγγραφείς. Το σύνολο δεδομένων FEMNIST έχει σχεδιαστεί για αποκεντροποιημένη Μηχανική Μάθηση, με δεδομένα για κάθε συγγραφέα.

1.5.2 Ετερογένεια δεδομένων

Η ετερογένεια των δεδομένων, μια κρίσιμη πτυχή της Ομοσπονδιακής Μηχανικής Μάθησης, ασχολείται με την ποικιλία στο μέγεθος, τη διανομή και την ποιότητα των δεδομένων σε διαφορετικούς κόμβους. Τα δεδομένα του πραγματικού κόσμου είναι συχνά άνισα κατανεμημένα, οδηγώντας σε προκλήσεις που θα μπορούσαν να οδηγήσουν σε υποβέλτιστα ή ανακριβή μοντέλα.

Ο βαθμός διακύμανσης των δεδομένων μπορεί να μετρηθεί για την επιλογή κατάλληλων αλγορίθμων για την μείωση των δυσμενών επιπτώσεων. Αυτό μπορεί επίσης να αξιολογήσει την καταλληλότητα της οργάνωσης ομοσπονδιακής εκμάθησης για έναν συγκεκριμένο τομέα προβλημάτων και να βελτιστοποιήσει την εκπαίδευση.

Στα πειράματά μας, η ετερογένεια των δεδομένων προσομοιώνεται με την προεπεξεργασία συνόλων δεδομένων και την ανάθεση σε κάθε συσκευή διαφορετικών υπολοίπων κλάσεων ή ενός τυχαίου αριθμού παραδειγμάτων.

Για να μετρήσουμε την ανισορροπία δεδομένων, χρησιμοποιούμε τη μέτρηση ετερογένειας διανομής του [60]. Το DH είναι μια μέτρηση μεταξύ 0 και 100%, όπου το 0% σημαίνει πανομοιότυπα παραδείγματα από κάθε κλάση (IID) μεταξύ των συσκευών και το 100% υποδηλώνει κάθε συσκευή που έχει παραδείγματα μόνο από μια κλάση (NON-IID).

Τέλος, η μέτρηση της ομοιογένειας χρησιμοποιείται για την αξιολόγηση της απόδοσης των αλγορίθμων ομαδοποίησης στη μηχανική μάθηση, καταγράφοντας την έκταση που κάθε σύμπλεγμα αποτελείται μόνο από στιγμιότυπα δεδομένων από μια ενιαία κλάση, αξιολογώντας την ποιότητα της ομαδοποίησης. Η βαθμολογία 1 υποδηλώνει τέλεια ομαδοποίηση, ενώ η βαθμολογία 0 σημαίνει αναποτελεσματική ομαδοποίηση.

1.5.3 Ετερογένεια υλικού

Η ετερογένεια του υλικού είναι ένας αποφασιστικός παράγοντας στο τοπίο της ομοσπονδιακής μηχανικής μάθησης, λαμβάνοντας υπόψη τις σημαντικές παραλλαγές στις διαμορφώσεις υλικού σε περιπτώσεις χρήσης cross-silo και cross-device.

Για περιπτώσεις χρήσης cross-silo, τα μοντέλα μηχανικής εκμάθησης χρησιμοποιούν μνήμη υψηλής χωρητικότητας και ισχυρούς CPU που διατίθενται συνήθως σε διακομιστές

ιδρυμάτων ή εταιρικών υπηρεσιών, προσφέροντας αποτελεσματικούς υπολογισμούς. Αντίθετα, οι θήκες χρήσης μεταξύ συσκευών διαθέτουν ένα ευρύ φάσμα υλικού από μικροεπεξεργαστές και συσκευές Raspberry Pi έως κινητά τηλέφωνα, το καθένα με μοναδικές υπολογιστικές δυνατότητες και περιορισμούς, υπογραμμίζοντας την εκτεταμένη ετερογένεια υλικού σε αυτές τις ρυθμίσεις.

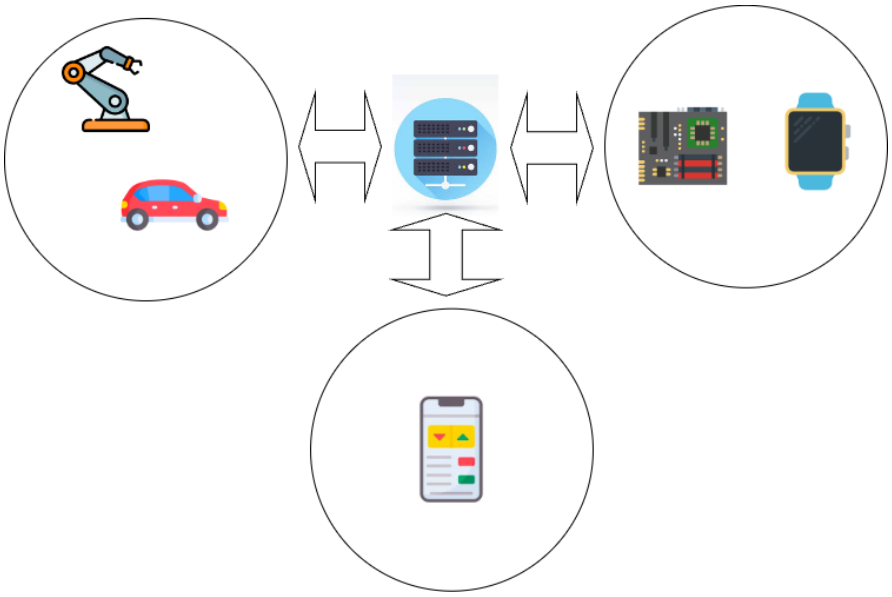


Figure 1.5.1: Ετερογένεια υλικού

Οι τεράστιες διαφορές στις διαμορφώσεις υλικού θέτουν προκλήσεις στην ανάπτυξη ισχυρών και ευέλικτων μοντέλων προσαρμόσιμων σε όλο το φάσμα των υπολογιστικών δυνατοτήτων. Η δομή του κώδικα μας, ευέλικτη και προσαρμόσιμη, βοηθά στην εξομοίωση της ετερογένειας του υλικού στα πειράματά μας, επιτρέποντας τη δυνατότητα διαμόρφωσης πολλών παραμέτρων.

Οι συσκευές μπορούν να κατηγοριοποιηθούν σε κατηγορίες χαμηλής, μεσαίας και υψηλής διαθεσιμότητας με βάση τα χαρακτηριστικά υλικού και τους περιορισμούς πόρων τους. Οι συσκευές χαμηλής διαθεσιμότητας (περιορισμένες σε πόρους) περιλαμβάνουν έξυπνα ρολόγια και ορισμένους μικροεπεξεργαστές με περιορισμένη επεξεργαστική ισχύ και μνήμη.

Για την εξομοίωση της ετερογένειας του υλικού, το σύστημα μας επιτρέπει να προσαρμόσουμε τα υπολογιστικά όρια κάθε συσκευής, προσομοιώνοντας περιβάλλοντα περιορισμένων πόρων που είναι κοινά σε περιπτώσεις μεταξύ συσκευών. Μεταβάλλοντας το μέγιστο όριο επεξεργαστή και το όριο μνήμης, προσομοιώνουμε διάφορα περιβάλλοντα υπολογιστικών πόρων.

Προσομοιώνουμε επίσης σενάρια πραγματικού κόσμου όπου οι συσκεύες συνδέονται και αποσυνδέονται κατά διαστήματα λόγω προβλημάτων όπως διακοπές επικοινωνίας ή διακοπές ρεύματος, ξεκινώντας και σταματώντας τυχαία τα container. Αυτή η ολοκληρωμένη προσέγγιση προσομοίωσης στοχεύει να διερευνήσει, να κατανοήσει και να προτείνει λύσεις σε

προκλήσεις ετερογένειας υλικού, συμβάλλοντας στην ανάπτυξη πιο ισχυρών και προσαρμόσιμων μοντέλων μηχανικής μάθησης.

1.5.4 Μετρικές

Η ακρίβεια, που αντιπροσωπεύει την αναλογία των σωστών προβλέψεων προς τις συνολικές παρατηρήσεις, είναι ένα βασικό μέτρο της απόδοσης του μοντέλου στις ομοσπονδιακές προσομοιώσεις μάθησης. Το εργαλείο μας υποστηρίζει τόσο την κεντρική δοκιμή, η οποία προσφέρει αμερόληπτη αξιολόγηση γενίκευσης, όσο και την αποκεντρωμένη δοκιμή, η οποία δείχνει την απόδοση του μοντέλου σε διαφορετικές διανομές δεδομένων.

Το Κόστος Ομοσπονδιακής Ποιότητας Μοντέλου (CFMQ) είναι μια μέτρηση για την αξιολόγηση της επίδρασης της ετερογένειας του υλικού στην Ομοσπονδιακή Μάθηση. Ενσωματώνει το κόστος επικοινωνίας, το τοπικό κόστος υπολογισμού και την ποικιλομορφία υλικού για να δώσει πληροφορίες για την απόδοση και τις απαιτήσεις πόρων σε διαφορετικά περιβάλλοντα υλικού.

Πολλές παράμετροι συμβάλλουν στο CFMQ όπως οι γύροι επικοινωνίας (R), το ωφέλιμο φορτίο επικοινωνίας (P),ο όρος εξισορρόπησης (α),ο μέσος αριθμός βημάτων τοπικής βελτιστοποίησης (m_i), η μέγιστη μνήμη που καταναλώνεται κατά τη διάρκεια βήμα (ν) και ο αριθμός των συσκευών που συμμετέχουν (K).

Στο παράδειγμά μας, το CFMQ είναι προσαρμοσμένο για να λαμβάνει υπόψη παραμέτρους που σχετίζονται με την συσκευή, όπως το ωφέλιμο φορτίο επικοινωνίας ανά συσκευή (P_i), έναν τροποποιημένο μέσο αριθμό βημάτων τοπικής βελτιστοποίησης ανά συσκευή (m_i) και τη μέγιστη μνήμη που καταναλώνεται κατά τη διάρκεια ενός βήματος ανά συσκευή (v_i). Το νέο CFMQ εκφράζεται ως εξής:

$$\text{CFMQ} = R \cdot \sum_i (P_i + \alpha \cdot m_i \cdot v_i) \quad \text{για όλες τις συσκευές}$$

Επίσης, ο χρόνος εκπαίδευσης ποσοτικοποιεί τη διάρκεια που απαιτείται για να μάθει ένα μοντέλο από τα κατανεμημένα δεδομένα. Περιλαμβάνει το χρόνο που απαιτείται για την εκπαίδευση του μοντέλου σε κάθε κόμβο και για την επικοινωνία και τη συγκέντρωση των ενημερώσεων του μοντέλου. Οι μικρότεροι χρόνοι εκπαίδευσης, που υποδηλώνουν αποτελεσματική μάθηση και καλύτερη επεκτασιμότητα, είναι επιθυμητοί στις περισσότερες πραγματικές καταστάσεις.

1.5.5 Προσομοιώσεις

Τα Συνελικτικα Νευρωνικά Δίκτυα (CNN), αποτελούν μια κατηγορία μοντέλων βαθιάς μάθησης, διαπρέπουν στον χειρισμό δεδομένων που μοιάζουν με πλέγμα, όπως εικόνες. Το κύριο συστατικό, το επίπεδο συνέλιξης, χρησιμοποιεί έναν πυρήνα για να εξάγει συστηματικά χαρακτηριστικά από εικόνες.

Συνήθως, μια μη γραμμική συνάρτηση ενεργοποίησης όπως η ReLU ακολουθεί τα επίπεδα συνέλιξης, αυξάνοντας τις δυνατότητες εκμάθησης. Στη συνέχεια, τα στρώματα συγκέντρωσης μειώνουν το χωρικό μέγεθος, ελέγχοντας τις παραμέτρους και την πολυπλοκότητα, ενώ αποτρέπουν την υπερπροσαρμογή. Η ιεραρχία του δικτύου του επιτρέπει να ανιχνεύει πολύπλοκα μοτίβα ανεξάρτητα από τη θέση τους σε μια εικόνα, επιδεικνύοντας επεκτασιμότητα και αποτελεσματικότητα.

Η βελτιστοποίηση παραμέτρων, ή η εκπαίδευση, στοχεύει στην ελαχιστοποίηση της διαφοράς μεταξύ των εξόδων του μοντέλου και των πραγματικών ετικετών, συνήθως χρησιμοποιώντας backpropagation και gradient descent.

Οι προσομοιώσεις μας περιλαμβάνουν την τροποποίηση παραμέτρων όπως το σύνολο δεδομένων, το μοντέλο, ο βελτιστοποιητής, τα μεγέθη παρτίδας και οι περιορισμοί του συστήματος. Χρησιμοποιούμε κυρίως ένα βασικό μοντέλο CNN με δύο συνελικτικά και τρία πλήρως συνδεδεμένα επίπεδα.

Στη διαδικασία εκπαίδευσής μας, χρησιμοποιούμε τη συνάρτηση Cross-Entropy Loss και τον βελτιστοποιητή Stochastic Gradient Descent, με ρυθμό εκμάθησης 0,001.

Συγκεντρώνουμε βασικές μετρήσεις κατά τη διάρκεια της εκπαίδευσης από την πλευρά της συσκευής, όπως η απώλεια κατα την διάρκεια εκπαίδευσης και η απόδοση του συνόλου στο κομμάτι δεδομένων το οποίο προβλέπεται για τελική δοκιμή, και επίσης δοκιμάζουμε την απόδοση σε δεδομένα τα οποία οι συσκεύες δεν έχουν πρόσβαση. Πραγματοποιούμε πρόσθετες δοκιμές χρησιμοποιώντας το συνολικό σύνολο δεδομένων του διακομιστή, το οποίο μπορεί να διαφέρει ως προς το μέγεθος, την κατανομή και την ετερογένεια. Αυτή η ποικιλία περιβαλλόντων δοκιμών ενισχύει την ευρωστία και την προσαρμοστικότητα του μοντέλου μας σε διαφορετικά περιβάλλοντα δεδομένων.

Αρχικά, εκπαιδεύουμε το μοντέλο μας σε συγκεκριμένα συμπλέγματα χρηστών των δεδομένων από την βάση FEMNIST, επιτυγχάνοντας πάνω από 0,9% ακρίβεια σε παρόμοια στυλ γραφής. Ωστόσο, η ακρίβεια πέφτει όταν δοκιμάζεται σε διαφορετικά στυλ, υπογραμμίζοντας την ανάγκη για μοντέλα που γενικεύονται σε διαφορετικά δεδομένα.

Στην συνέχεια της κεντρικοποιημένης εκπαίδευσης, χρησιμοποιούμε Ομοσπονδιακή Μηχανική Μάθηση με τον αλγόριθμο FEDMA. Η ρύθμιση περιλαμβάνει 7 συσκευές, 10 γύρους και 10 εποχές ανά γύρο. Το προκύπτον μοντέλο, που δημιουργήθηκε με την ενσωμάτωση διαφορετικών τοπικών μοντέλων, προβλέπει με ακρίβεια ακόμη και σε στυλ γραφικού χαρακτήρα του οποίου οι συσκεύες δεν έχουν πρόσβαση 1.5.2.

Το μοντέλο κάθε συσκευής μπορεί να προβλέψει με ακρίβεια τα αποτελέσματα των δεδομένων του, ακόμη και με ακραίες τιμές. Εάν εγγραφεί μία νέα συσκευή, μπορεί να χρησιμοποιήσει το παγκόσμιο μοντέλο για ακριβή εξαγωγή συμπερασμάτων χωρίς πρόσθετη εκπαίδευση, δείχνοντας την αποτελεσματικότητα της Ομοσπονδιάς Μάθησης στον χειρισμό της ετερογένειας των δεδομένων και τη γενίκευση σε διάφορες εισόδους.

Στην συνέχεια χρησιμοποιούμε το σύνολο δεδομένων MNIST με ανισορροπία δεδομένων στο πείραμά μας, με κάθε συσκεύη να έχει 4 κλάσεις. Προσομοιώνουμε την ετερογένεια κλάσεων (DH=0,4) εκχωρώντας έξι από τα δέκα ψηφία/κλάσεις από το σύνολο δεδομένων

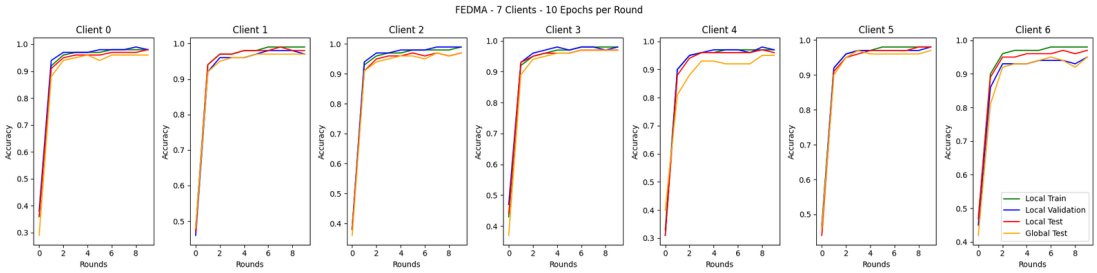


Figure 1.5.2: Αποτελέσματα εκπαίδευσης στο FEDMA με διαφορετικές ομάδες FEMNIST

σε κάθε συσκεύη. Κατά τη διαδικασία FML, οι συσκευές εκπαιδεύουν τα μοντέλα τους και τα στέλνουν στον διακομιστή μετά από κάθε επανάληψη.

Οι ακρίβειες μετά την εκπαίδευση συσκευή δείχνουν υψηλές τιμές για τα σετ εκπαίδευσης και επικύρωσης. Η δοκιμή στα δεδομένα της ίδιας της συσκευής συγκλίνει σταδιακά στην ακρίβεια εκπαίδευσης, ενώ η δοκιμή σε παγκόσμια μη προσβάσιμα δεδομένα αναμένεται να φτάσει το 0,6, αντικατοπτρίζοντας το 60% του συνόλου δεδομένων που διαθέτει κάθε συσκευή.

Η πραγματική αξία της ομοσπονδιακής μηχανικής μάθησης εν μέσω της ετερογένειας της τάξης αποκαλύπτεται με την παρατήρηση της ακρίβειας του καθολικού μοντέλου του διακομιστή σε ολόκληρο το σύνολο δοκιμών του συνόλου δεδομένων 1.5.3.

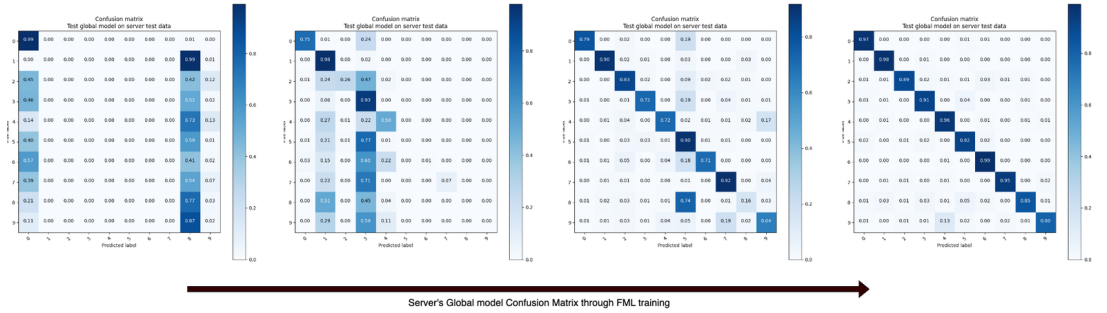


Figure 1.5.3: Πρόοδος μήτρας σύγχυσης σε παγκόσμιο μοντέλο

Συχνά οι συσκευές έχουν ποικίλες χωρητικότητες και μεγέθη συνόλων δεδομένων. Δοκιμάζοντας με διαφορετικά μεγέθη σε κάθε προσομοίωση, αναλύουμε την ακρίβεια. Σημειωτέον, ακόμη και όταν μειώνονται τα δείγματα δεδομένων από 15.000 σε 10.000, το σύστημα εξακολουθεί να συγκλίνει με την ίδια ακρίβεια, σημαντικό όταν οι συσκευές έχουν περιορισμούς πόρων.

Η προσομοίωση διαφορετικών αριθμών συσκευών είναι ζωτικής σημασίας για τη βελτιστοποίηση ενός ομοσπονδιακού συστήματος εκμάθησης. Λιγότερες συσκευές απλοποιούν το σύστημα και βοηθούν στον εντοπισμό σφαλμάτων, ενώ περισσότερες συσκευές δοκιμάζουν τη φόρτωση του συστήματος και αποκαλύπτουν προβλήματα από-

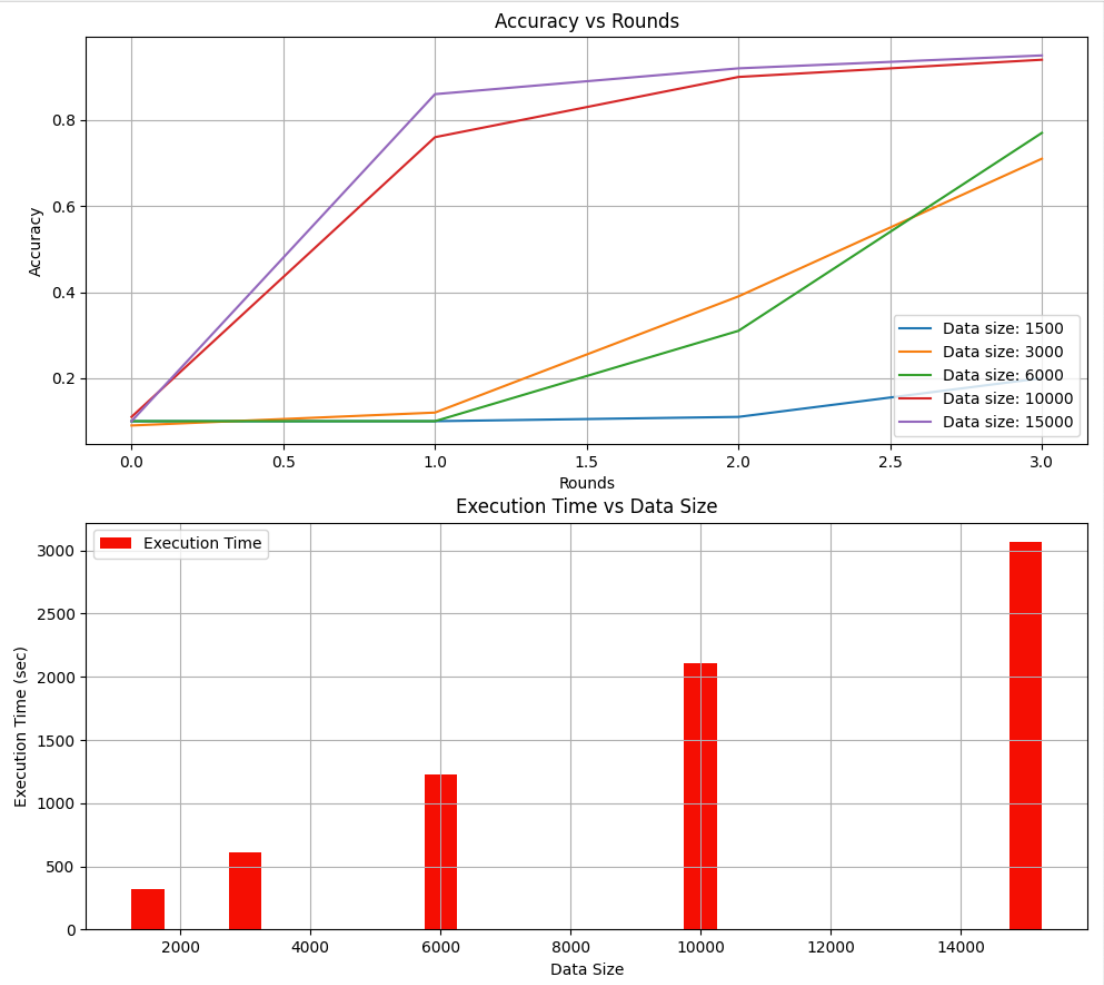


Figure 1.5.4: Ακρίβεια και χρόνος εκτέλεσης σε διαφορετικά μεγέθη δεδομένων

δοσης. Αντιπροσωπεύουν επίσης καλύτερα την ετερογένεια που χαρακτηρίζει την ομοσπονδιακή μάθηση.

Για την προσομοίωση διαφορετικών αριθμών συσκευών, διατηρούμε σταθερές ορισμένες παραμέτρους, όπως μνήμη, CPU, γύροι διακομιστή, εποχές ανά γύρο, σύνολο δεδομένων, και στρατηγική. Μετράμε την ακρίβεια, τον χρόνο και το CFMQ με 3, 5 και 10 συσκευές μετά την εκπαίδευση.

Table 1.1: Χρόνος εκτέλεσης και τιμές CFMQ

Αριθμός Συσκευών	Χρόνος (δευτ.)	CFMQ	Απόδοση διακομιστή μετά από 3 γύρους
3	900	2,177MB	0.43
5	1,193	3,347MB	0.38
10	1,606	7,296MB	0.36

Εξετάσαμε τον αντίκτυπο της ετερογένειας της CPU στην Ομοσπονδιακή Μηχανική Μάθηση. Για να απομονώσουμε αυτό το φαινόμενο, διατηρήσαμε πολλούς παράγοντες σταθερούς στις προσομοιώσεις, όπως το όριο μνήμης, τους κύκλους διακομιστή, τις εποχές ανά γύρο, το μοντέλο νευρωνικού δικτύου, το σύνολο δεδομένων, τις κατατμήσεις δεδομένων και τη στρατηγική.

Ξεκινήσαμε με την κατανομή δύο πυρήνων CPU ανά συσκευή και σταδιακά μειώσαμε έως ότου κάθε μία έλαβε το 1/8 της ισχύς της CPU. Καθώς η ισχύς της CPU ανά συσκευή μειώθηκε, ο χρόνος εκτέλεσης για την εκπαίδευση διπλασιάστηκε. Η μέτρηση CFMQ παρέμεινε περίπου στα 22 * 10^8MB, παρά τις αλλαγές στην κατανομή της CPU. Η ακρίβεια της τελικής δοκιμής παρέμεινε περίπου στο 0,9 μετά από τρεις γύρους διακομιστή. Αυτά τα αποτελέσματα υπογραμμίζουν την προσαρμοστικότητα του συστήματος σε διαφορετικές διαμορφώσεις CPU.

Table 1.2: Αποτελέσματα ετερογένειας CPU

Συσκεύες	CPUs	CPUs/Συσκευή	Χρόνος(δευτ.)	CFMQ	Ακρίβεια Διακομιστή
3	6	2	623.28	2530MB	0.92
3	3	1	1240.51	2300MB	0.93
3	3/2	1/2	2315.45	2240MB	0.92
3	3/4	1/4	5487.26	2210MB	0.91
3	3/8	1/8	12065.69	2010MB	0.92

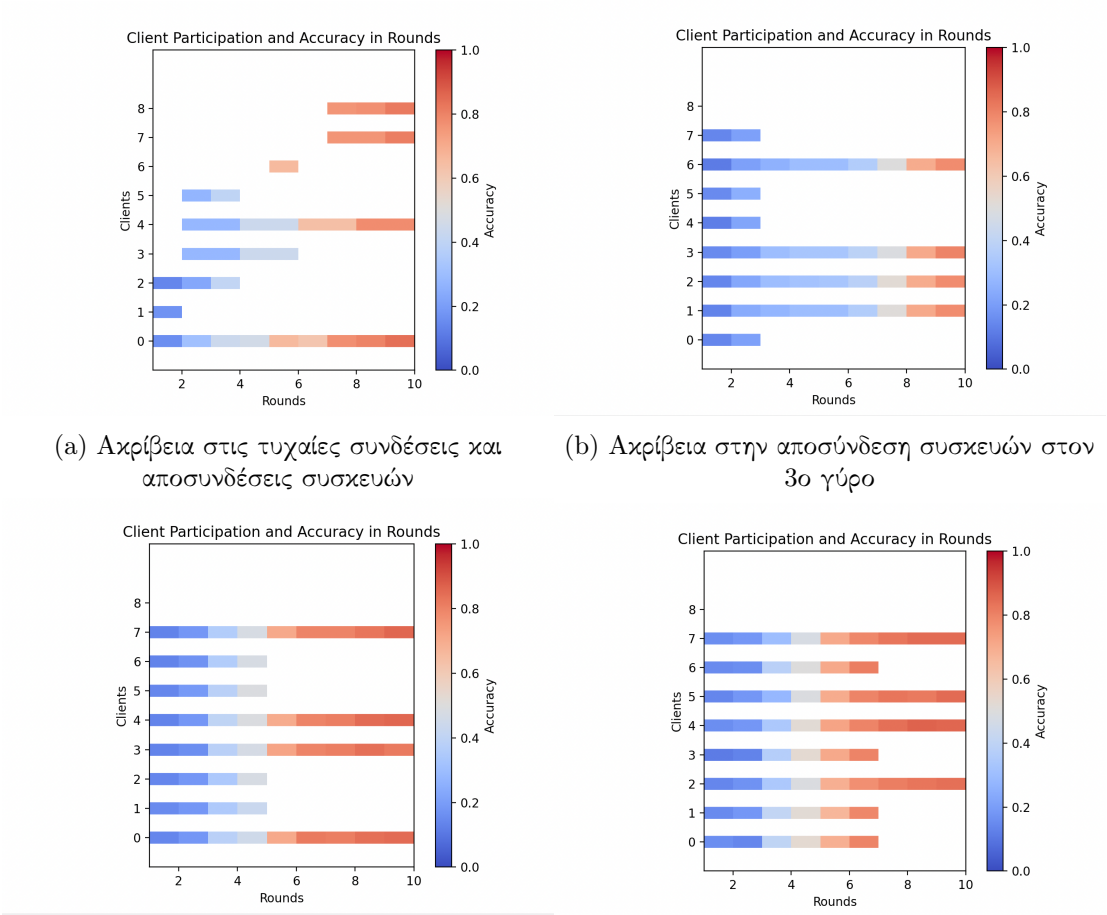
Πραγματοποιήσαμε ένα πείραμα για να κατανοήσουμε τον αντίκτυπο των συσκευών που δεν μπορούν να συμβαδίσουν με τις υπόλοιπες συσκεύες λόγω περιορισμένων πόρων. Δημιουργήσαμε ομάδες συσκευών αιχμής με ίση κατανομή CPU σε κάθε ομάδα, αλλά ποικίλες κατανομές μεταξύ ομάδων, προσομοιώνοντας σενάρια πραγματικής ανισότητας πόρων. Ο στόχος μας είναι να χρησιμοποιήσουμε αυτήν τη ρύθμιση για να κατανοήσουμε καλύτερα τη συμπεριφορά των ομόσπονδων συστημάτων μάθησης παρουσία ατόμων και να βελτιώσουμε την αποτελεσματικότητά τους σε περιβάλλοντα με περιορισμένους πόρους.

	Προσ. 1	**Προσ. 2**	**Προσ. 3**
Σύνολο Συσκευών	4	4	8
Συσκευές - Συστάδα 1	2	2	4
Συσκευές - Συστάδα 2	2	2	4
CPUs/Συσκευή - Συστάδα 1	2	3	3
CPUs/Συσκευή - Συστάδα 2	1	1/2	1/2
Χρόνος(δ)/Γύρος Μ.Ο.-Συστάδα 1	150, 79, 55	79, 47, 32	238, 237, 235
Χρόνος(δ)/Γύρος Μ.Ο.-Συστάδα 2 Χρόνος Προσ.	238, 124, 85 1019s	314, 190, 125 1569s	566, 562, 568 2443s
CFMQ	3027MB	6102MB	13012MB
Τελική Ακρίβεια	0.92	0.92	0.92

Table 1.3

Για την προσομοίωση κατανεμημένων περιβαλλόντων μάθησης στον πραγματικό κόσμο,εισάγουμε κόμβους που αποσυνδέονται κατά διαστήματα. Οι λόγοι για αυτές τις εγκαταλείψεις μπορεί να περιλαμβάνουν ζητήματα συνδεσιμότητας, διαθεσιμότητα συσκευών, περιορισμούς πόρων, προτιμήσεις χρήστη, ενημερώσεις συστήματος ή σφάλματα και λανθάνουσα κατάσταση επικοινωνίας.

Δημιουργήσαμε πολλαπλές ομάδες συσκευών ακμής που η καθεμία αντιμετώπιζε διαφορετικά ποσοστά εγκατάλειψης προκαλώντας αποσυνδέσεις κόμβων. Αυτή η προσομοίωση αστάθειας δικτύου μας βοήθησε να μελετήσουμε τις επιπτώσεις των σποραδικών αποσυνδέσεων στην απόδοση και την ευρωστία του συστήματος.



(a) Ακρίβεια στις τυχαίες συνδέσεις και αποσυνδέσεις συσκευών

(b) Ακρίβεια στην αποσύνδεση συσκευών στον 3ο γύρο

(c) Ακρίβεια στην αποσύνδεση συσκευών στον 6ο γύρο

(d) Ακρίβεια στην αποσύνδεση συσκευών στον 7ο γύρο

Figure 1.5.5: Ακρίβεια στην αποσύνδεση συσκευών

Παρά την κυμαινόμενη συμμετοχή και την εγκατάλειψη των συσκευών, τα ανεξάρτητα και πανομοιότυπα κατανεμημένα δεδομένα εξασφάλισαν τη συνέχιση της βελτίωσης της ακρίβειας του παγκόσμιου μοντέλου. Ακόμη και οι συσκευές που προσχώρησαν καθυστερημένα μπόρεσαν να επωφεληθούν από την προϋπάρχουσα μάθηση, με αποτέλεσμα

την ταχεία μάθηση και την υψηλή αρχική ακρίβεια. Αυτό το πείραμα έδειξε την αποτελεσματικότητα της ομοσπονδιακής μάθησης σε δυναμικά κατανεμημένα περιβάλλοντα.

Η πρόωρη εγκατάλειψη των συσκευών είχε ως αποτέλεσμα την ταχύτερη ολοκλήρωση των κύκλων εκπαίδευσης λόγω μειωμένης πιθανότητας καθυστέρησης. Ωστόσο, η εγκατάλειψη συσκευών που συνέβη αργότερα εξασφάλισε πιο ολοκληρωμένη εκπαίδευση του παγκόσμιου μοντέλου λόγω της αυξημένης ποικιλομορφίας και όγκου δεδομένων. Επομένως, είναι ζωτικής σημασίας να εξισορροπηθεί η συμμετοχή των συσκευών με τις εκτιμήσεις λανθάνοντος χρόνου για τη βέλτιστη ομοσπονδιακή μάθηση.

1.6 Περιπτώσεις χρήσης

1.6.1 Εφαρμογές για κινητά

Η Ομοσπονδιακή μηχανική εκμάθηση (FML) βελτιώνει τις εμπειρίες από κινητές συσκευές, συμπεριλαμβανομένων μοντέλων γλώσσας σε πληκτρολόγια και αναγνώρισης προσώπου ή φωνής, χρησιμοποιώντας εκμάθηση στη συσκευή χωρίς αποστολή δεδομένων σε κεντρικούς διακομιστές [28, 4, 27, 25]. Αυτές οι εφαρμογές λαμβάνουν υπόψη διάφορους παράγοντες όπως οι δυνατότητες υλικού χρήστη, οι συνθήκες δικτύου και η χρήση της μπαταρίας. Η Ομοσπονδιακή Μάθηση έχει δείξει βελτιώσεις σε σχέση με τα παραδοσιακά μοντέλα, λαμβάνοντας υπόψη το απόρρητο, την αποτελεσματικότητα και τους νομικούς περιορισμούς που σχετίζονται με τη χρήση δεδομένων.

1.6.2 Αυτόνομα οχήματα

Η Ομοσπονδιακή Μάθηση ενισχύει την εμπειρία του αυτοκινήτου αυτόνομης οδήγησης μειώνοντας τον λανθάνοντα χρόνο, μειώνοντας τα έξοδα μετάδοσης δεδομένων και βελτιώνοντας το απόρρητο [23, 18, 43]. Αντί για κεντρική εκπαίδευση δεδομένων, επιτρέπει την αποτελεσματική χρήση δεδομένων σε πραγματικό χρόνο από διάφορους αισθητήρες.

1.6.3 Ιατρικά δεδομένα

Η μηχανική εκμάθηση που βασίζεται σε δεδομένα έχει πολλές δυνατότητες στην υγειονομική περίθαλψη, αλλά περιορίζεται από ρυθμιστικούς περιορισμούς [34]. Η Ομοσπονδιακή Μάθηση παρέχει μια λύση διατήρησης της ιδιωτικότητας, επιτρέποντας την ανάλυση δεδομένων χωρίς μεταφορά δεδομένων πέρα από το ίδρυμα όπου είναι αποθηκευμένο. Μπορεί να αντιμετωπίσει προκλήσεις όπως η μεροληψία, η ετερογένεια δεδομένων και το απόρρητο στα ιατρικά δεδομένα, βελτιώνοντας παράλληλα τα αποτελέσματα της υγειονομικής περίθαλψης.

1.7 Συμπέρασμα

1.7.1 Μελλοντικές αναβαθμίσεις

Οι μελλοντικές αναβαθμίσεις για το εργαλείο προσομοίωσης Ομοσπονδιακής Μηχανικής Μάθησης θα επικεντρωθούν στη βελτίωση της αλληλεπίδρασης με τον χρήστη, στην

ενίσχυση των αναλυτικών στοιχείων και της παρακολούθησης, στην επέκταση των επιλογών μοντέλων και στρατηγικής, στην ενσωμάτωση διαφορικού απορρήτου, στην ενεργοποίηση της ανάπτυξης σε συσκευές πραγματικών συσκευών και στην προσομοίωση παραμέτρων πραγματικού κόσμου.

Οι διευρυμένες επιλογές μοντέλων και οι στρατηγικές ομοσπονδιακής μάθησης θα επιτρέψουν στους ερευνητές να επιλέξουν τα πιο κατάλληλα μοντέλα για τις απαιτήσεις τους.

Με την ενσωμάτωση διαφορικών τεχνικών απορρήτου, θα ενισχύσουμε τις διασφαλίσεις απορρήτου του εργαλείου προσομοίωσης, προστατεύοντας ευαίσθητα δεδομένα κατά τη διάρκεια της εκπαίδευσης. Οι δυνατότητες ανάπτυξης για συσκευές πραγματικών άκρων θα διευκολύνουν τον πρακτικό πειραματισμό. Οι προσομοιώσεις που ενσωματώνουν παραμέτρους της συσκευής του πραγματικού κόσμου, όπως η ισχύς σήματος WiFi, τα επίπεδα μπαταρίας και τα πρότυπα χρήσης θα βοηθήσουν στη ρεαλιστική αξιολόγηση.

1.7.2 Επίλογος

Αυτή η μελέτη έχει φωτίσει τις δυνατότητες της Ομοσπονδιακής Μηχανικής Μάθηση στην αντιμετώπιση σημαντικών προκλήσεων δεδομένων, συμπεριλαμβανομένης της ετερογένειας, του απορρήτου και της ιδιοκτησίας. Χρησιμοποιώντας το πλαίσιο ανοιχτού κώδικα Flower, έχουμε αποδείξει την πρακτικότητα και την ευελιξία του σε μια ποικιλία ερευνητικών σεναρίων, αξιολογώντας παράγοντες όπως η ετερογένεια δεδομένων και υλικού και η αποσύνδεση συσκευών.

Τα ευρήματά μας επιβεβαιώνουν την αξία της Ομοσπονδιακής Μηχανικής Μάθηση στον χειρισμό ευαίσθητων και μεγάλης κλίμακας δεδομένων, διατηρώντας παράλληλα το απόρρητο και την ιδιοκτησία. Θέτουν μια σταθερή βάση για μελλοντική έρευνα και ανάπτυξη, υπογραμμίζοντας την ικανότητά του ως προηγμένου εργαλείου για έρευνα και εφαρμογές μηχανικής μάθησης.

Chapter 2

Introduction

2.1 Rise of Big Data

In recent years, the term "Big Data" has become increasingly popular not only within the technology sector but also in other fields. This term refers to the aggregation of enormous quantities of data in a single repository, which has become possible due to advancements in data storage and management technologies.

The collection and analysis of Big Data has become a critical element in many sectors, including healthcare, financial institutions, government agencies, and e-commerce companies, to name a few [10]. These organizations have been at the forefront of data gathering and have utilized Big Data to gain insights into customer behavior, market trends, and other valuable information. As technology continues to evolve, it is expected that the use of Big Data will become increasingly widespread, with many organizations recognizing its potential to drive innovation and growth.

Big Data is also a term used to describe extremely large and complex data sets that cannot be effectively processed or analyzed using traditional data management and processing tools. These data sets include structured data, semi-structured data, and unstructured data. The concept of Big Data is characterized by the "3Vs" of data [33]: volume, variety, velocity and new "Vs" like Veracity, Variability and Low-Value density[10].

2.2 Privacy in the Age of Big Data

Over the past few decades, privacy has been extensively studied, with a focus on topics such as cryptography, communication, and information theory. However, given the enormous size of Big Data, it has become increasingly challenging to effectively apply existing cryptographic solutions. Furthermore, the limited processing and storage capacity of mobile devices makes encryption and decryption infeasible. As a result, conventional cryptographic solutions are no longer suitable for addressing the emerging privacy requirements of Big Data.

The failure of simple anonymization techniques also adds to the challenges in managing privacy in the era of Big Data. The lack of a clear definition of privacy, coupled with its subjective nature, makes it difficult to establish a universal definition that can be applied across all contexts. As the adoption of Big Data continues to grow rapidly, questions about the reliability of existing privacy techniques have emerged. As a consequence, it is important to revisit existing privacy studies in the context of Big Data and develop new algorithms, models, and frameworks to address the challenges of privacy in this new paradigm. This will be crucial for ensuring that privacy concerns are effectively addressed and that individuals' personal data is protected as Big Data continues to play an increasingly important role in various sectors [5].

The increasing use of Big Data has brought about a range of issues and challenges, particularly with regard to privacy attacks and the need for effective counter-techniques. Various techniques, such as k-anonymity [49], t-closeness [36], l-diversity [39], and differential privacy [19], have been developed to address these privacy concerns. However, there is still a need for further research in this area, particularly with regard to developing effective anonymization techniques for unstructured Big Data [5].

2.3 Rise of AI

Indeed, AI and Big Data are closely connected [32], as AI is often used to facilitate the capture, structure, and analysis of Big Data. With the volume, variety, velocity, and veracity of Big Data, it can be challenging for organizations to extract meaningful insights from the data. This is where AI comes in, as it can help automate and streamline the data analysis process, making it easier to identify patterns and trends.

By leveraging machine learning algorithms and other AI techniques, organizations can process and analyze large data sets more efficiently and effectively. AI can help identify key relationships between different data points, predict future outcomes, and provide insights that might not be apparent through manual analysis.

In addition, the use of AI can help organizations overcome some of the challenges associated with Big Data, such as the complexity and diversity of data types. By automating certain aspects of the data analysis process, AI can help organizations focus on the most critical data and gain valuable insights more quickly.

2.4 Ethical Problems of AI

AI has the potential to transform our lives and make them easier in many ways, but it is important to acknowledge that there are also potential drawbacks associated with its use. AI systems can be incredibly powerful, but they can also perpetuate biases and discrimination [6, 17], and they may make decisions that are harmful or unethical. Additionally, there are concerns that the use of AI may result in job loss or other negative impacts on employment. Ethics on AI is a subject that has been gaining increment attention the recent years and a lot of studies focus on this burning subject [7, 48]

Despite these concerns, the benefits of AI cannot be ignored. AI is being used to improve healthcare outcomes, optimize transportation systems, enhance cybersecurity, and more. AI-powered virtual assistants and chatbots are also becoming increasingly popular, providing a convenient way for people to access information and services.

To ensure that the benefits of AI are realized while minimizing the potential drawbacks, it is important that AI systems are developed and used in a responsible and ethical manner. This includes designing AI systems that are transparent, explainable, and accountable, as well as developing regulations and guidelines to ensure that AI is used in a way that aligns with societal values. By taking a thoughtful and responsible approach to AI, we can harness its potential to improve our lives while minimizing the potential risks. Based on [48] there are 39 ethical issues that we can identify and some of them are:

1. Lack of trust
2. Lack of quality data
3. Negative impact on health
4. Problems of integrity
5. Lack of accuracy of data
6. Lack of privacy
7. Lack of transparency
8. Bias and discrimination
9. Unfairness
10. Misuse of personal data
11. Negative impact on democracy
12. Loss of freedom and individual autonomy
13. Contested ownership of data
14. Problems of control and use of data and systems
15. Negative impact on vulnerable groups
16. Lack of accountability and liability
17. Negative impact on the environment
18. Loss of human decision-making

These issues have a close connection to the usage of Machine Learning and researchers and regulations are trying to safeguard humanity in the face of this ever-growing technology. The outcome of these efforts p.e. is the AI Act [21].
The issues of opacity, unpredictability, and the need for large datasets in machine learning techniques based on artificial neural networks contribute to the growing concerns

about AI explainability. The lack of transparency and interpretability in AI decision-making can result in decisions that are difficult to understand, leading to concerns about potential biases, inaccuracies, and unintended consequences. With machine learning algorithms that are opaque and unpredictable, it is difficult for the developer, deployer, or user to know how the system will react to a given set of inputs. Furthermore, the adaptiveness and dynamic nature of these systems mean that past behaviors are not always a perfect predictor of future behavior in identical situations. Therefore, the need for AI explainability is becoming increasingly important to ensure that AI systems are transparent, accountable, and aligned with ethical and societal values. Developing techniques and tools for improving AI explainability is essential to overcome these challenges and ensure the responsible development and deployment of AI systems.

2.5 Privacy Vs Security on AI

In the context of AI, it is important to distinguish between data security and data privacy. While both are essential considerations in ensuring the responsible development and deployment of AI systems, they address different aspects of data protection.

Data security focuses on protecting data against unauthorized access, including when and what can be accessed. This involves implementing measures to prevent data breaches, such as encryption and access controls, and ensuring that AI systems are designed and deployed in a secure manner. Data security can be achieved without compromising privacy, as it is possible to implement strong security measures while still protecting personal data.

Data privacy, on the other hand, focuses on who can access the data and the ability to protect personally identifiable information. This includes ensuring that personal data is collected and processed in a responsible and ethical manner, and that individuals have control over their own data. Data privacy cannot be achieved without adequate data security measures, as protecting personal data requires ensuring that it is kept confidential and secure.

Overall, both data security and data privacy are essential considerations in ensuring the responsible development and deployment of AI systems. By implementing robust security measures and ensuring that personal data is protected and used in a responsible manner, we can help to ensure that the benefits of AI are realized while minimizing the potential risks.

Although there is always a concern for security, it has been well-researched and studied in depth. However, when it comes to privacy guarantees, there are numerous examples of failures that have resulted in significant problems.

2.6 Machine Learning Architectures

Significant advancements in Artificial Intelligence (AI) and Machine Learning (ML) have been driven by breakthroughs not only in these specific areas but also in related fields

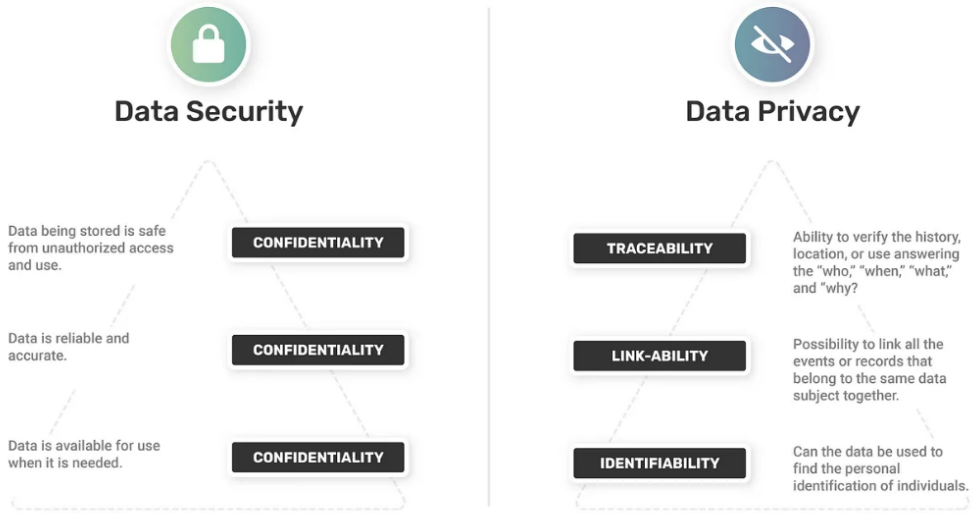


Figure 2.5.1: Privacy vs Security

such as cloud computing, edge computing, and hardware technology. These developments have facilitated AI and ML to operate effectively on the edge, directly where data is produced, streamlining the process of inference at the point of origin.

One approach that has garnered considerable attention in recent years is distributed machine learning, which involves dispersing computation across multiple nodes for efficient processing. This evolution from the traditional centralized model, where all data was collected and processed in a single location, was a major leap forward. Distributed machine learning significantly reduced latency and increased scalability but still fell short in ensuring complete data privacy as it still necessitated data sharing between nodes.

To address this lingering concern, Federated Machine Learning (FML) was introduced [40]. FML represents the latest evolution in this journey [2], presenting an innovative method that allows model training, testing, and inference to be conducted at the edge. By keeping data at its source, it eliminates the need for data transfer, thereby preserving privacy by design and offering a robust solution to protect user data.

FML is part of the broader scope of Privacy-Preserving Machine Learning technologies [16]. By combining the benefits of decentralized processing with enhanced privacy measures, Federated Machine Learning represents the industry's continuous efforts to balance efficient data processing with stringent data privacy and security [52]. The implementation and further development of such technologies signify the ever-growing importance of data privacy in our increasingly interconnected digital world [11].

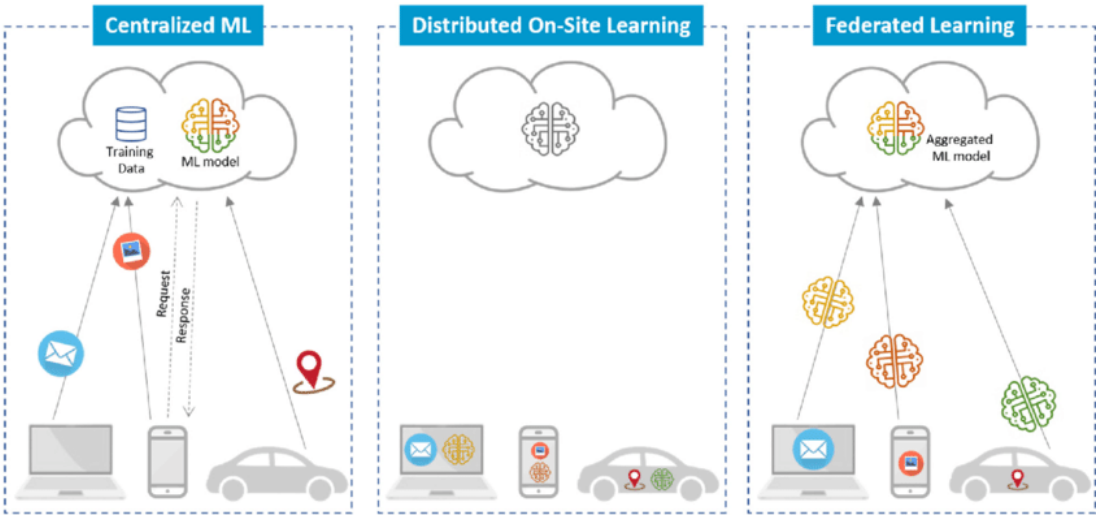


Figure 2.6.1: Centralized, Decentralized and Federated ML

2.7 Contributions

This thesis provides an exhaustive exploration of Federated Machine Learning (FML), outlining its principles, key contributions to Machine Learning (ML), and potential as a solution for data privacy concerns. In this research endeavor, we have also developed an innovative simulation tool based on the Flower framework.

This tool facilitates the simulation of a multitude of FML scenarios, accounting for aspects like data and hardware heterogeneity, as well as the occurrence of stragglers and client dropouts. It features an intuitive user interface that empowers researchers to distribute partitioned datasets among multiple clients with diverse hardware specifications. Equipped with comprehensive monitoring capabilities, this tool enables detailed tracking of a wide range of performance metrics, including accuracy, training loss, network capacity, and CPU performance.

The objective of this study is not only to provide an in-depth understanding of FML but also to bridge the gap between theoretical knowledge and its practical application, shedding light on the expansive potential of FML in addressing real-world challenges.

2.8 Related Work

Federated Learning (FL) solutions in the field of machine learning have been divided broadly into two categories: libraries and end-to-end systems. Libraries such as PySyft [45], and Flower [9] offer a flexible suite of tools to design, implement, and execute FL algorithms. They provide granular control over the FL process but require considerable expertise and additional infrastructure to manage and monitor FL workflows at scale4.

On the other hand, end-to-end systems such as FEDn [22] and FedML [29] provide comprehensive, prepackaged solutions that handle all aspects of the FL workflow. These

include data distribution, model training, updates aggregation, and monitoring, among other features. They are designed to offer an integrated experience, reducing the complexity typically associated with orchestrating FL workflows.

In addition to these systems and libraries, our work also delves into the challenges that arise due to hardware heterogeneity, class heterogeneity, stragglers, and dropouts in the context of federated learning. Each of these factors can significantly impact the effectiveness and efficiency of FL algorithms and require thoughtful strategies to mitigate their effects. We investigate these issues and propose solutions to address them, with a particular emphasis on their relevance in real-world FL deployments.

2.9 Chapter Description

The "Introduction" outlines the context of the study, addressing the rise of big data and AI, the resultant privacy concerns, the ethical problems of AI, and the ongoing struggle between privacy and security in AI. It also briefly introduces the variety of machine learning architectures, underlining the specific contributions of the thesis and associated works in the field. Subsequently, the "Privacy Preserving Machine Learning" section delves into various methodologies for maintaining privacy within machine learning, discussing concepts like differential privacy, homomorphic encryption, and multiparty computation.

The thesis then provides a comprehensive overview of Federated Learning (FL), elaborating on its learning algorithms, its frameworks, and the challenges it poses. In the "System Architecture" section, the focus shifts to the practical aspects of the study. The architecture of the system used in the thesis is presented using UML diagrams, with sections dedicated to system description, monitoring, and transitioning from Jupyter to Federation. In the "Experimental Results" section, the outcomes of the experimental investigations are detailed, including discussions about the datasets used, class/data and hardware heterogeneity, and simulation results. Finally, the thesis explores practical applications or "Use Cases" of the system, in contexts like mobile applications, autonomous vehicles, and medical data management, before concluding with an assessment of the study and potential directions for future work.

Chapter 3

Privacy Preserving Machine Learning

Privacy-preserving machine learning [44] is a subfield of machine learning that aims to protect the sensitive information contained within the data while still enabling efficient and accurate model training and predictions. This involves developing methods and techniques that safeguard the data used in the machine learning process from unauthorized access or extraction. One popular method is differential privacy, which adds noise to the data or computations to ensure that the outcome of a machine learning algorithm does not reveal specific details about any individual data point. Another approach is federated learning, where the data remains on local devices and only model updates are shared and aggregated. Homomorphic encryption, secure multi-party computation, and zero-knowledge proofs are also used to perform computations directly on encrypted data. Privacy-preserving machine learning is crucial in many areas such as healthcare or finance where data confidentiality is paramount.

3.1 Differential Privacy

Differential privacy [19] is a framework designed to enable data analysis without compromising individual privacy. By adding statistical noise to either input or output data, the probability of learning a specific set of parameters remains relatively consistent, even if a single training example in the dataset is altered. This framework ensures privacy by maintaining a privacy budget, where smaller budgets correspond to stronger privacy guarantees. There are two main types of differential privacy: Local and Global. Local differential privacy adds noise to individual data points, providing plausible deniability for each user, while Global differential privacy introduces noise in the output of a dataset, requiring users to trust the party managing the database. Both types provide similar privacy guarantees, with Global differential privacy offering higher accuracy at the expense of requiring trust in a central authority.

The level of privacy provided by a differential privacy mechanism is determined by

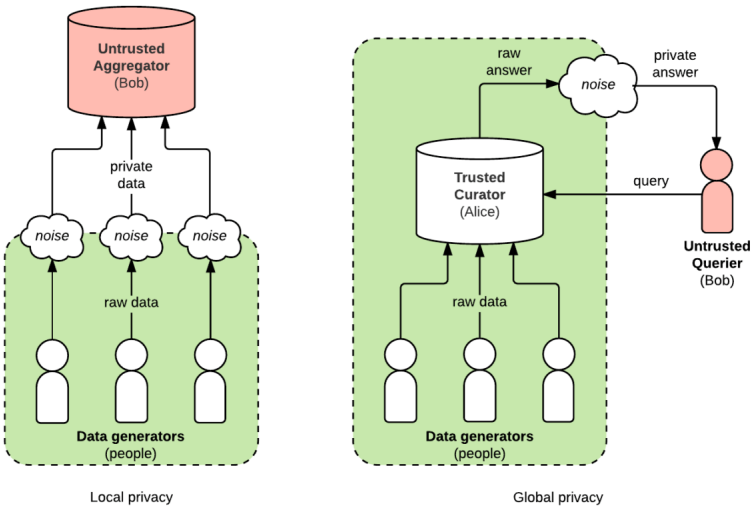


Figure 3.1.1: Differential Privacy with FML

two parameters, epsilon (ϵ) and delta (δ). Epsilon represents the maximum difference between the output probabilities of two neighboring databases, with smaller values indicating stronger privacy guarantees. Delta is the probability of accidental information leakage, with zero leakage corresponding to (ϵ)-differential privacy. The amount of noise added to achieve the desired level of privacy depends on several factors, including the sensitivity of the query, desired epsilon and delta values, and the type of noise to be added (e.g., Gaussian or Laplacian). By carefully selecting these parameters, differential privacy allows for meaningful data analysis while maintaining the privacy of individuals within the dataset. Differential Privacy is able to tackle different privacy attacks like linkage and reconstruction [20].

Definition: Differential Privacy: A randomized algorithm M is (ϵ, δ)-differentially private if for all data sets D and D' differing on at most one row, and any $S \subseteq Range(M)$,

$$\Pr[M(D) \in S] \leq \exp(\epsilon) \times \Pr[K(D') \in S] + \delta$$

One of the early works in machine learning using Differential Privacy can be found in the work by Abadi et al.[1] where they presented a study on differentially private stochastic gradient descent. They introduced the "moments accountant" as a tool to monitor privacy loss when applying the Gaussian mechanism to random subsets of the input data. Differential privacy can be used as the mean to minimize the risk of the AI-Big data recognized problems of bias and discrimination while preserving the model accuracy. With the use of DP the models comply to the strict regulations, ethical requirements and keep inclusiveness and fairness in their predictions [51].

3.2 Homomorphic Encryption

Homomorphic encryption [3] is a form of cryptography that allows computation to be performed on ciphertext, without revealing the underlying plaintext. In simpler terms, it allows computations to be carried out on encrypted data, thereby protecting the privacy of the data. There are two main types of homomorphic encryption: fully homomorphic encryption (FHE) [26] and partially homomorphic encryption (PHE). Also there are more types with different attributes, like Somewhat homomorphic encryption [24], Leveled fully homomorphic encryption [57] FHE allows for arbitrary computations to be performed on ciphertext, while PHE only allows for computations of a specific type (either addition or multiplication) to be performed on ciphertext.
Homomorphic encryption has many applications in areas such as cloud computing [50], secure data storage, and secure data processing. For example, it can be used to allow computation on sensitive data without the need for the data to be decrypted first, thus protecting the privacy of the data.

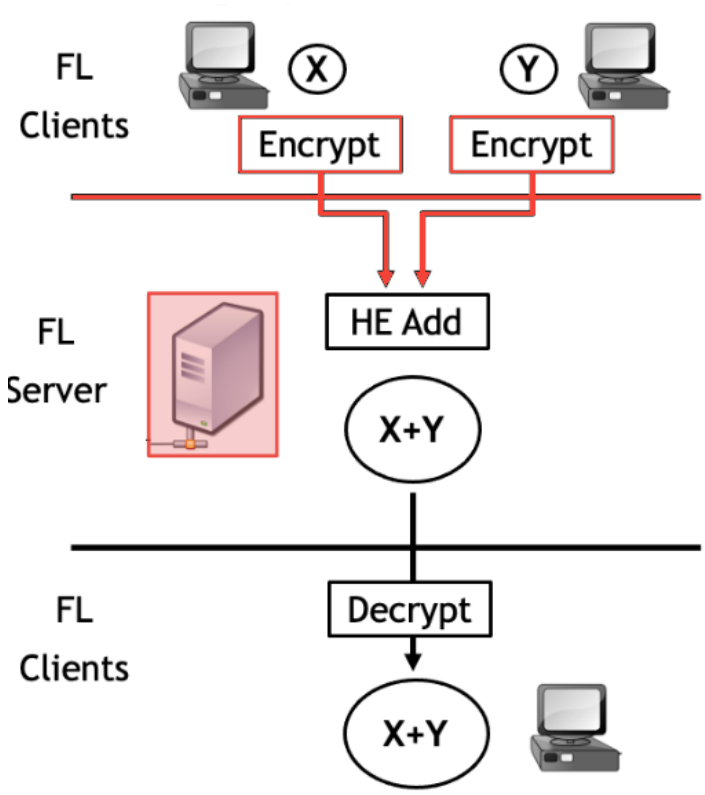


Figure 3.2.1: Homomorphic Encryption with FML

However, homomorphic encryption has some limitations. Firstly, it can be computationally intensive, which can limit its practical use in certain applications. Secondly, it can be vulnerable to attacks that exploit the structure of the encrypted data. Despite these limitations, homomorphic encryption is an active area of research, and it has the

potential to revolutionize the way sensitive data is handled and processed.
The federated machine learning process can potentially expose sensitive information in the updates sent to the server. By using homomorphic encryption, the updates can be encrypted before being sent to the server, thus ensuring that the sensitive information remains private. The central server can then perform computations on the encrypted updates without decrypting them, and send the updated model back to the parties. This is a different approach from the common encryption technics where the encryption happens when the data are stored or when are transmitted. There are several challenges associated with using homomorphic encryption in federated learning, such as the computational overhead and the need for specialized hardware. However, researchers have made significant progress in overcoming these challenges, and homomorphic encryption is seen as a promising technique for enhancing the privacy and security of federated learning.
In this paper [57] purpose a system model for based on the BFV scheme [56] so that the server is aggregating on encrypted data. Also they give the algorithms for model training in the clients where the model is trained normally in a federated machine learning paradigm with the difference of encrypting the data after the fit. In the server, the model aggregation happens on the encrypted data without decrypting doing the holomorphic addition and multiplication that is required. Lastly, the decryption on the client side is taking place where the clients are using their keys in order to access the weights and update them for the next round of local training. In the same work it is shown that the execution times are exponentially greater with the use of encryption but the prediction model performance metrics stay the same.

3.3 Multiparty Computation

Secure multi-party computation (MPC) is a cryptographic technique designed to safeguard digital assets or protect sensitive information by splitting it into multiple parts. The method is characterized by its ability to maintain secrecy since no single participant can access or leak the entire "truth". MPC eliminates the need for trusted third parties to see the data, resolving the tradeoff between data usability and privacy, while offering high accuracy and precision. However, the technique comes with some drawbacks: it requires significant computational overhead due to the need for generating random numbers to ensure secure computation, and it involves high communication costs since it necessitates connectivity and communication between all participants. Thus, while MPC is highly secure and reliable, it can slow down runtime and increase operational costs. Furthermore, MPC can be combined with federated machine learning, enhancing the efficacy and privacy-preserving capabilities of these systems [41, 30, 13].

Chapter 4

Federated learning Overview

By enabling machine learning models to be trained on data that remains decentralized and secure, federated machine learning can help to ensure that AI systems are both privacy-respecting and secure. By leveraging the power of decentralized data, federated machine learning has the potential to unlock new insights and drive innovation while protecting personal data and mitigating the risks associated with centralized data collection and analysis.
The inception of federated learning can be traced back to the seminal paper by McMahan et al [40], which primarily focused on the growing capabilities of modern devices to store, analyze, and process vast amounts of data, as well as to train machine learning models locally. The authors identified several critical challenges associated with such devices, including the sensitivity of the data they contain and the reluctance of users to share this information. Furthermore, the authors recognized the difficulties in transmitting data to external locations, primarily due to concerns related to data quality and volume. In response to these challenges, the concept of a decentralized approach, termed Federated Learning, was proposed. This innovative method aimed to facilitate the analysis and application of machine learning models directly on the devices, without compromising data privacy. In this paper, McMahan demonstrated the potential of this technology in addressing issues related to privacy and robust machine learning. Additionally, they highlighted the challenges posed by heterogeneous data (non-IID) and communication costs. To summarize, this pioneering paper laid the groundwork for Federated Learning by addressing the burgeoning need for decentralized machine learning approaches. The authors acknowledged the potential of modern devices in handling large-scale data and model training, while simultaneously emphasizing the importance of addressing data privacy concerns. The proposed Federated Learning framework not only demonstrated the feasibility of addressing these issues, but also shed light on the challenges of dealing with heterogeneous data and communication costs, paving the way for future advancements in this domain.

The basic Federated architecture consists of several key components:

Participating Nodes: In the FML architecture, nodes refer to individual devices or organizations that contribute data to the learning process. These nodes could be smartphones, IoT devices, or separate entities with their own data repositories like financial and government institutions. The nodes are responsible for local computation and maintaining the privacy of their data. Each node trains a local machine learning model using its own dataset and shares only the model parameters or updates with the central server, rather than raw data, thus preserving data privacy.

Central Server: The central server plays a crucial role in the Federated Machine Learning process. It is responsible for aggregating the model updates received from the participating nodes and coordinating the overall learning process. The server merges the updates from all nodes to create a global model, which is then sent back to the nodes for further refinement. The central server also ensures that the communication between nodes is secure and efficient.

Communication Protocol: An essential aspect of the FML architecture is the communication protocol that facilitates the exchange of information between the participating nodes and the central server. The protocol needs to be secure, reliable, and efficient to ensure the privacy and integrity of the data while minimizing the communication overhead.

Learning Algorithms: The choice of learning algorithms in Federated Machine Learning is an important factor that affects the overall efficiency and effectiveness of the system. FML typically uses distributed optimization algorithms that can be applied iteratively and in parallel across the participating nodes.

4.1 Types of FML

A fundamental characteristic of federated machine learning (FML) revolves around the nature of the participating devices or entities, which dictates whether the approach is cross-device or cross-silo[42]. The magnitude, specifications, and resource capabilities of these entities form the basis for differentiating between these two styles of federated learning.

The difference between cross-silo and cross-device federated learning lies in the nature of their architectures and the specific data privacy challenges they address. In cross-silo federated learning, data is split across different servers or organizations (the silos), and machine learning models are trained on each of these separate silos. This approach is particularly valuable in scenarios where data cannot be pooled together due to regulatory constraints, competitive considerations, or privacy concerns. For example, multiple hospitals might collaborate to train a shared machine learning model without having to disclose individual patient data to one another.

On the other hand, cross-device federated learning is designed for scenarios where data is generated and stored directly on users' devices, such as mobile phones or Internet-

of-Things (IoT) devices. In this setting, each device trains its own machine learning model based on locally stored data and only sends model updates to a central server for aggregation. This approach enables data to stay on the original device, boosting privacy and security. It's particularly useful in consumer-focused applications, where personal data is often sensitive and regulations like GDPR demand strict privacy protections.

In essence, the differences arise due to the contrasting demands of the environments where the data resides, and the unique privacy and logistical challenges posed by each environment.

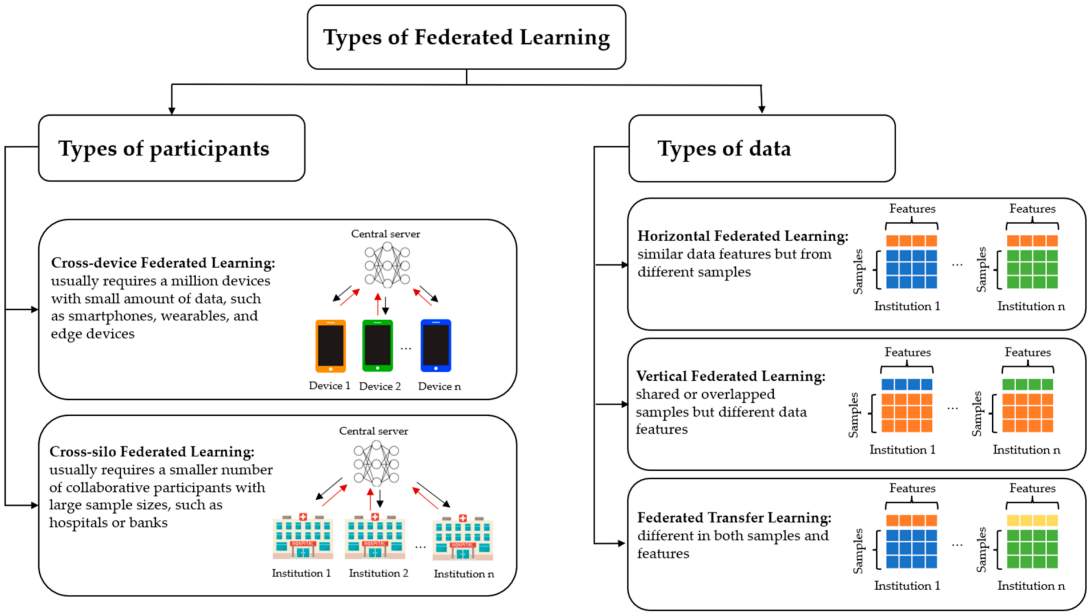


Figure 4.1.1: FML Types

There are also three types of federated learning, depending on how data is distributed.

Vertical Federated Learning [55][59]: This method is used when different entities have different features for the same set of samples. In vertical federated learning, all participating organizations collect different feature data for the same group of users. This form of federated learning is useful when there is a large number of features, and the models that are being build, consider all of these features without having to share raw data between organizations.

An example could be two businesses that interact with the same clients but collect different kinds of information about them. A bank might have financial data about a customer, and a health app might have health data about the same person. In vertical federated learning, these two businesses can collaboratively learn a model without sharing their raw data.

Horizontal Federated Learning [59]: This method is used when different entities have the same features but for different individuals. It means that different parties own information about different individuals but share a similar feature space.

For example, multiple hospitals might have the same type of medical information (like MRI scans) about different patients. Each hospital can contribute to the learning process without directly sharing patient data, protecting individual privacy.

Transfer Federated Learning [46]: In this type, learning models are trained in one organization and then transferred to another where the model can be fine-tuned. This is often used when the target organization has insufficient data or when it is desirable to transfer knowledge from one domain to another.

For instance, a general image recognition model could be trained on a large, diverse dataset at a tech company, then transferred to a hospital where it's fine-tuned to recognize specific types of medical images. The hospital benefits from the initial learning but doesn't need a large dataset of its own to get started

4.2 Learning Algorithms

4.2.1 FedAVG

Federated Averaging, often called as FedAvg, is an algorithm used in Federated Learning. This method was introduced by McMahan et al. [40] as a way to build machine learning models using data that remains on the device where it was created, instead of transferring it to a central server for training. Here is a general description of how the FedAvg algorithm works:

1. Initialize the global model: A global model is initialized on the server. This model's weights are randomly initialized.
2. Broadcast the global model: The global model is then sent to a fraction of the total devices participating in the federated learning process. This fraction may be chosen randomly or based on certain criteria (e.g., devices that are currently active). The model received by the devices is used as a starting point for local training.
3. Local Training: Each of these devices, often referred to as clients, trains the model on their local data. This typically involves several epochs of training on the local dataset. During this phase, the raw data never leaves the individual device, thus preserving privacy.
4. Aggregate updates: Once local training is completed, each client calculates an update to the model (i.e., the difference between the weights of the locally trained model and the original global model), and sends these updates back to the server.
5. Update the global model: The server then aggregates these updates from all the clients. The most common way to do this is by averaging the updates (hence the

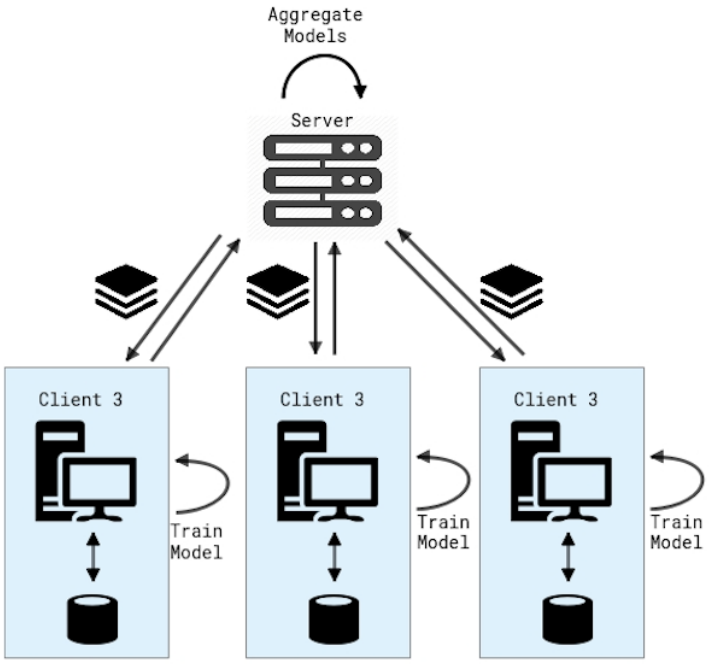


Figure 4.2.1: Federated Machine Learning Setting

name "Federated Averaging"). This averaged update is then used to update the weights of the global model.

6. Repeat the process: Steps 2-5 are repeated for a number of communication rounds until the model converges to a satisfactory level of accuracy.

The goal of FedAvg is to provide a method to train machine learning models that can effectively learn from decentralized data while maintaining privacy, as the raw data never leaves the local device. This makes it useful for applications where data privacy is a concern or when it is impractical to bring all the data to a central server. However, it is worth noting that it still faces challenges such as handling non-IID data, stragglers, and varying amounts of data among clients.

4.2.2 FedMA

The federated matched averaging (FEDMA) algorithm is designed for neural networks CNN and LSTMS and was proposed by IBM [54].
One difference from FedAVG is that the merging of model parameters is not being done in a naive way. More specifically, FedAVG assumes that the neurons of all the client models, explain the features extracted from the training are in the same parameter and overlap. This issue arises due to the permutation invariance of neural network (NN) parameters, where the same NN can have different parameters even if the input is the same.This problem becomes even more visible when the data are Non-IID and fedAVG loses a lot of the unique characteristics of each client by this naive averaging.

Algorithm 1 Federated Averaging (FedAvg)

```
Initialize global model parameters w
for each round t = 1, 2, ..., T do
    Randomly select K clients to participate in training
    for each client k in selected clients do
        Send global model parameters w to client k
        Client k trains model on local data to produce updated parameters w_k
        Client k sends updated parameters w_k back to server
    end for
    w = average of all received parameters w_k from clients
end for
return final global model parameters w =0
```

So FedAVG is a good strategy for benchmarking but losses in accuracy. To tackle this problem, FEDMA algorithm uses a technique called matching average, based on Probabilistic Federated Neural Matching (PFNM)[61].
Its key point is that the global shared model that creates comes from a layer-wise manner by matching and averaging the hidden elements of the model: the channels of the CNNs or the hidden states of the LSTMs, or the neurons of the fully connected layers. The matching of the hidden elements is happening with similar feature extraction signatures.

In the FEDMA algorithm, there are, the federated learning rounds like to ones in FEDAVG and in every FML round there are the matching rounds where each client trains the model on the local data, sends it back to the server where each time a specific layer is matched and augmented as shown in Figure 4.2.2. More in detail, FEDMA matches the neurons of each client model's (FCNN) before averaging them by using the Bayesian non-parametric method to adapt to global model size and to the non-IID data. After the clients receive the augmented layer and train the rest of the model with that model frozen. At the end of the layers, the model is augmented in every layer, holding the information from every client's model.
An FCNN can be formulated as: $\hat{y} = \sigma(\mathbf{x}W_1\Pi)\Pi^T W_2$,

where Π is any $L \times L$ permutation matrix, and L is the number of hidden units.

Suppose we have datasets X_j, X'_j and the weights after training as $\{W_1\Pi_j, \Pi_j^T W_2\}$ and $\{W_1\Pi_{j_0}, \Pi_{j_0}^T W_2\}$. With high probability, we have $\Pi_j \neq \Pi_{j_0}$ and $\frac{W_1\Pi_j + W_1\Pi_{j_0}}{2} \neq W_1\Pi$ for any Π. To tackle this FEDMA needs to undo the permutation.

In order to test FEDMA and transition it in a more federated learning format, we used the Flower framework and the neccessary changes were made in order to achieve this. So from the simple FedAVG strategy we transitioned to a more complex architecture and flow that can be seen at the diagram.

The server initiates the process by configuring the models, datasets, and setting up the requisite SSL configurations. Following this, the relevant classes, namely FEDMA and

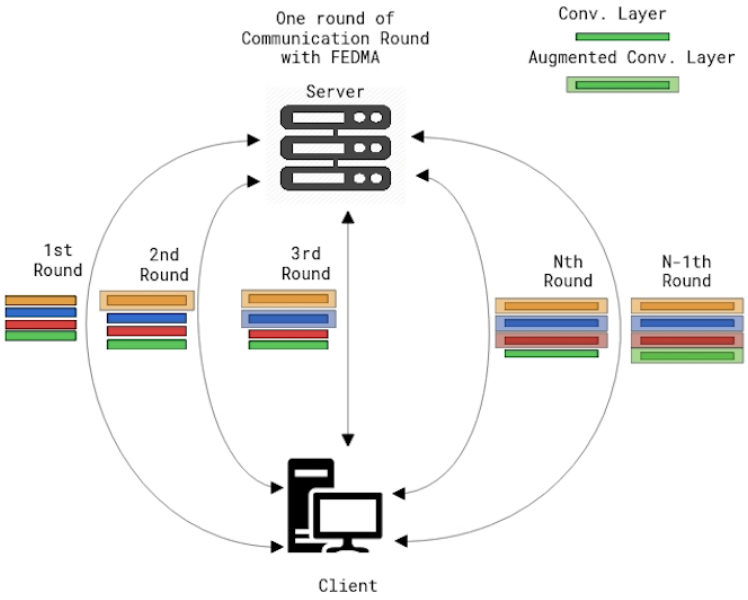


Figure 4.2.2: FEDMA Round

Client Manager, are initialized. The server then commences operation, starting with the first round during which the clients undergo fitting. Concurrently, the Client Manager waits for clients to connect. Once a sufficient number of clients have connected, the server distributes the configuration file to each client.

Parallel to this, the clients boot up, establish a connection with the server, and receive the eagerly-awaited configuration file. Subsequently, the clients proceed to load the datasets and the model.

The Flower Framework is designed to operate in two rounds: the matching round and the standard FML round. The unique aspect of the matching round is the augmentation of the models as previously described, wherein each round trains the subsequent layer before freezing it. During the standard FML round, the models adapt to align the augmented model with the initial one.

On the client side, the model undergoes evaluation and testing. If it's the end of the round on the server, the final model layers are generated or the server performs layer-wise matching, subsequently sending the models back to the clients.

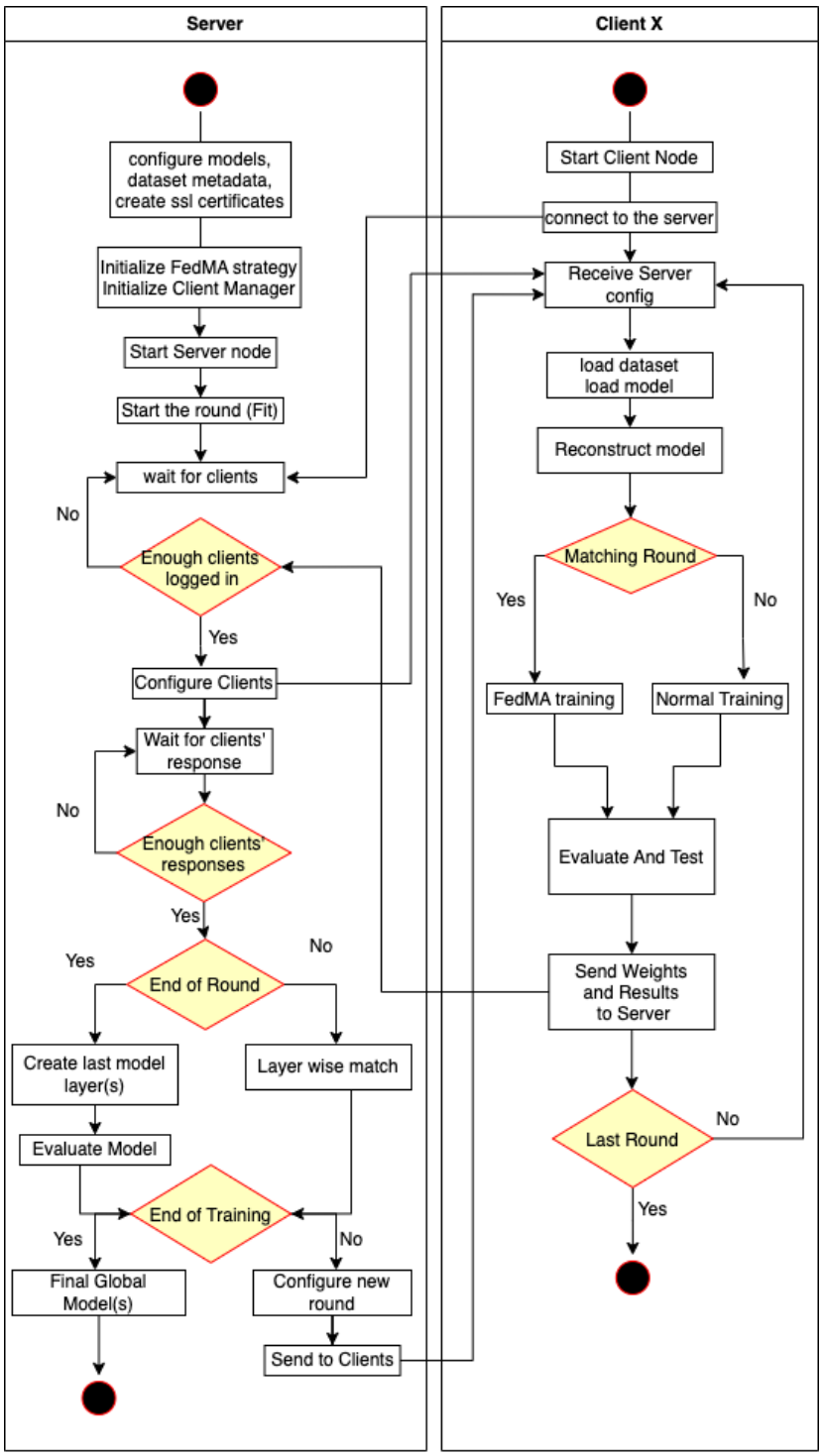


Figure 4.2.3: Sequence of FEDMA training

Algorithm 1: Federated Matched Averaging (FedMA)

Input : local weights of N-layer architectures $\{W_{j,1}, \ldots, W_{j,N}\}_{j=1}^{J}$ from J clients
Output: global weights $\{W_1, \ldots, W_N\}$
$n = 1$;
while $n \leq N$ **do**
  if $n < N$ **then**
    $\{\Pi_j\}_{j=1}^{J}$ = **BBP-MAP**$\left(\{W_{j,n}\}_{j=1}^{J}\right)$; `// call BBP-MAP to solve Eq. 2`
    $W_n = \frac{1}{J} \sum_j W_{j,n} \Pi_j^T$;
  else
    $W_n = \sum_{k=1}^{K} \sum_j p_{jk} W_{jl,n}$ where p_k is fraction of data points with label k on worker j;
  end
  for $j \in \{1, \ldots, J\}$ **do**
    $W_{j,n+1} \leftarrow \Pi_j W_{j,n+1}$; `// permutate the next-layer weights`
    Train $\{W_{j,n+1}, \ldots, W_{j,L}\}$ with W_n frozen;
  end
  $n = n + 1$;
end

Figure 4.2.4: Federated Matched Averaging

4.3 FML Frameworks and Systems

A multitude of frameworks have been developed to facilitate the implementation and deployment of Federated Machine Learning (FML). These frameworks vary greatly in their design, functionality, and focus areas, aiming to tackle different aspects of the challenges in the FML paradigm. Some are designed with a focus on certain machine learning tasks, such as deep learning or gradient-boosted trees, while others prioritize specific computing environments like edge computing or data center setups. Furthermore, they can differ in terms of privacy preservation techniques, such as differential privacy or secure multi-party computation, that are built-in or supported. Additionally, the type of federated learning scenario they cater to - horizontal, vertical, or federated transfer learning - can also be a differentiating factor. Some are designed to be highly scalable and handle a large number of clients, while others emphasize the robustness against dropping out or malicious clients. Interoperability with popular machine learning libraries, usability, and community support can also differ among these frameworks. In the sections that follow, we will delve into a detailed discussion about the most prominent frameworks developed for Federated Machine Learning.

4.3.1 Flower

Flower is a friendly federated learning framework. It is designed to allow machine learning practitioners and researchers to easily implement and experiment with federated learning algorithms. This is particularly useful for training models on decentralized data, where data privacy and security are important, or when data cannot be moved for other reasons.

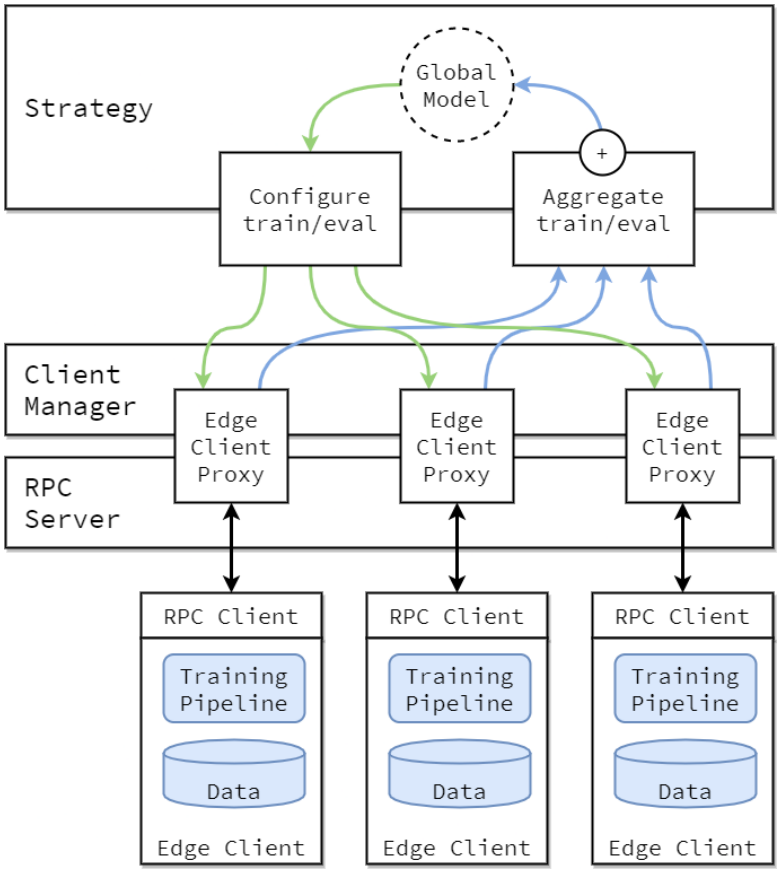


Figure 4.3.1: Flower Framework Architecture

Flower provides an easy-to-use framework for implementing federated learning algorithms. It is designed to be flexible and extensible, allowing the use of different machine learning libraries (e.g., TensorFlow, PyTorch), and is compatible with various programming languages. It provides a server-client architecture, where a central server communicates with multiple clients to orchestrate the federated learning process. The central server coordinates the learning process, sends model updates to clients, and collects updates from them.

4.3.2 FEDn by Scale:

FEDN [22] is a production-scale, hierarchical federated machine learning (FML) framework that offers a flexible tool for research. It is designed to incorporate private data into the federated model without exposing the data, ensuring privacy. The framework supports various aggregation techniques and privacy-enhancing technologies, while focusing on scalability and resilience. Although most FML work is based on neural networks, random forest implementations also exist.

FEDN is structurally similar to distributed optimization for statistical learning, with

no control over data distribution across nodes. The framework balances local training iterations with global synchronization to minimize communication rounds and avoid poor convergence. Inspired by the MapReduce programming model, FEDN aims to be robust, resilient, and highly scalable in real-world applications, accommodating both large numbers of edge clients and large model sizes.

The FEDN architecture supports cross-device and cross-silo scenarios, incorporating horizontal and vertical approaches. Federated averaging (FedAvg) is the most widely used method for horizontal FL, a decentralized version of stochastic gradient descent. FEDN's architecture consists of three main components: Combiners, Reducers, and Clients.

1. Combiner (CB): A stateless gRPC server that coordinates client updates and aggregates model updates from a subset of clients. It operates independently of other combiners in the network and works on one partial model update at a time. This stateless design provides fault-tolerance and horizontal scalability, with the combiner workload scaling linearly with the number of attached clients.

2. Reducer: Responsible for implementing a reducer protocol, which combines partial model updates from combiners into a single global model update for each round. The system supports multiple reducers running in active-passive mode, with one reducer preparing the global model at a time. The reducer workload scales with the number of combiners, independent of the number of clients.

3. Clients (CL): Main workers in the system that run on local sites or devices and access private data. Clients join the network by requesting a combiner assignment from the discovery service. They receive training and validation requests, download the global model from the combiner, execute model updates and validations, and stream results back to the combiner. Clients follow a black-box execution model, allowing for ML-framework agnostic implementation, and do not require any open ingress ports, which is an important consideration in many production environments.

The Controller Round Protocol algorithm and HAProxy for high availability are utilized in FEDN, along with specific metrics tailored for this architecture. In summary, FEDN offers a fault-tolerant, scalable, and flexible federated learning system with stateless Combiners, Reducers, and Clients working together to coordinate and aggregate model updates while preserving privacy.

4.3.3 Protea

Protea [62], designed as a client profiling component within federated systems using the Flower Federated Learning framework, is a forward-thinking tool focused on enhancing simulation capabilities in Federated Learning (FL). The tool offers automatic collection of system-level statistics and resource estimation for each client, which facilitates resource-aware simulation. Guided by principles of usability, flexibility, compatibility, efficiency, and scalability, Protea aims to streamline the design and implementation of

FL frameworks. It ensures higher degrees of realism in FL simulations by considering the specific capabilities of each client, such as computational power, network speed, and data size.
Protea is proficient in monitoring critical metrics such as CPU, RAM, GPU, VRAM, CPU time, and CUDA time. By providing this granular view, Protea aids in accurate client profiling and resource allocation, fostering efficient simulations. The tool shows commendable performance in accelerating simulations with its design successfully augmenting parallelism, leading to significantly faster wall-clock times and improved GPU utilization. The capabilities of Protea extend to handling both homogeneous and heterogeneous configurations of clients, making it a valuable tool for large-scale experiments. Through these functionalities, Protea contributes towards reducing the barriers in Federated Learning research and application, thus encouraging more realistic and scalable FL solutions.

4.3.4 FLINT

FLINT [53], a device-cloud collaborative platform for Federated Learning (FL) integration, is an innovative addition to LinkedIn's established centralized machine learning platform. Its architecture is designed to share common components with the centralized ML platform such as model stores, job scheduling, monitoring, and visualization tools, thus creating a synergy between centralized and federated approaches.
One of the key strengths of FLINT lies in its ability to build upon well-known FL and centralized ML platforms, integrating tools to leverage centralized data and resources. This allows for comprehensive analysis of FL's impact and viability in various contexts. FLINT also introduces a feature catalog that manages both cloud and device-based data, providing a unified view of the data landscape across different sources. It contains an experimental framework designed to optimize model performance and system requirements, facilitating efficient and effective FL implementations. Moreover, FLINT aids decision-making through its workflow that provides an understanding of the constraints, costs, and effectiveness of FL for business needs. It delivers key insights that help decision-makers assess the applicability and potential benefits of implementing FL. FLINT is also attentive to the User Device Availability, considering the compute capabilities of devices, device state, user attributes, and the use of a proxy data generator. This focus ensures that the platform can effectively operate in real-world scenarios where device availability and capabilities can significantly impact the performance of federated learning models.
In essence, FLINT represents a holistic approach to FL integration, blending the strengths of existing ML platforms with the privacy and decentralization benefits of federated learning.

4.4 FML Challenges

In the paradigm of distributed Federated Machine Learning (FML), not only are the fundamental complexities of Artificial Intelligence (AI) and Machine Learning (ML) present, but additional challenges arise due to the distributed nature of the systems

involved. Particularly when factoring in devices and the Internet of Things (IoT), the level of difficulty and intricacy substantially increases.

1. Non-IID Data: In a federated learning setup, data may not be independently and identically distributed (non-IID) across clients.
2. Data heterogeneity: Varying data size, distribution, and quality across nodes
3. Asynchronous Communication: In real-world federated learning, clients might not always be available or have varying computational resources.
4. Varying Client Participation: Not all clients may participate in every round of federated learning.
5. Data Drift: The data distribution may change over time (concept drift), causing the model to become less accurate.
6. Communication Overhead: High communication costs due to the frequent model updates between server and clients.
7. Adversarial Clients: Federated learning may involve clients that intentionally introduce incorrect updates or malicious data.
8. Communication Errors and Noise: In real-world scenarios, communication between clients and the server may experience errors or noise.
9. Asynchronous Communication: Clients might not always be available or have varying computational resources.
10. Model Compression and Quantization: Communication bandwidth may be limited, leading to the need for model compression and quantization.
11. Heterogeneous Model Architectures: Clients may have different model architectures due to varying hardware constraints or preferences.
12. Privacy Attacks: Federated learning aims to protect clients' privacy, but privacy attacks can still occur.

Chapter 5

System Architecture

5.1 UML diagrams

One of the most pressing challenges in the field of Federated Machine Learning (FML) pertains to issues of reproducibility and implementation. Unlike traditional data science or machine learning scenarios, where the requisite code executes on a centralized server with abundant data, FML represents a convergence of distributed systems and machine learning. It's essential to recognize that the bulk of academic research in this area does not adequately reflect the realities of real-world software infrastructure. Moreover, many issues pertinent to FML are frequently overlooked. The approach to implementing an

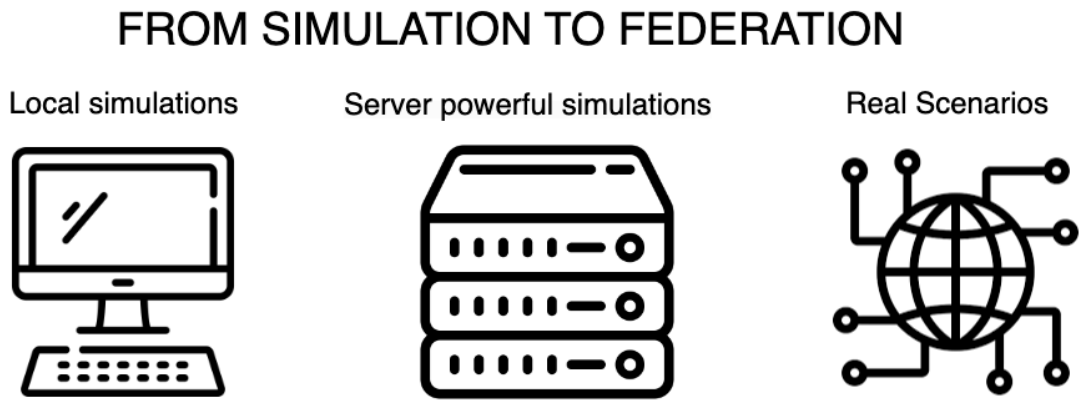


Figure 5.1.1: Simulation to Federation

FML paradigm closely mirrors the established protocol for implementing mainstream machine learning models and use-cases. It begins with scripts that operate on a researcher's personal computer, with FML servers and clients represented as instances of processes communicating within the local machine.

The subsequent phase involves transposing the implementation to an upgraded environ-

ment, such as a server, where memory, CPU, and disk space no longer constitute bottlenecks. The final test of implementation efficacy takes place in a distributed environment, on other large-scale servers (cross-silo) or across multiple edge devices (cross-device).

As part of our initiative, we have developed a robust system, equipped with state-of-the-art components, that can effectively simulate the entire Federated Machine Learning paradigm. This approach allows for a comprehensive understanding of the complexities and challenges inherent to this domain, promoting efficient solutions and more reliable results.

In order to visualise the details of each part of our solution, there are Unified Modeling Language (UML) diagrams below. This schematic representation enables a comprehensive understanding of the complex structure and interrelationships of our system components.

5.1.1 Component Diagram

As depicted in the provided Component diagram 5.1.2, all components operate within containers, ensuring a scalable and modifiable system structure. The fundamental elements of our paradigm, the Federated Machine Learning (FML) Manager and FML Client containers, represent the server-client relationship inherent to any FML use case. The FML server houses scripts responsible for the management of the paradigm, while clients each possess a unique database and maintain connectivity with the manager.
A pivotal component of the system is the Redis database, tasked with storing essential information during the training process. This includes data vital for training, such as dataset partitioning for each client, as well as container-specific information (names, IPs), which is crucial for effective system monitoring.

The Management Server Service incorporates a Flask API designed to initiate the Docker swarm, thereby facilitating seamless client connection and data transmission. This service also oversees the necessary Docker networks, volumes, and services. Its role is crucial in system initialization, scalability to accommodate numerous clients, and maintaining fault tolerance. The monitoring system forms an integral part of a distributed machine learning paradigm. With data creation occurring at the edge, this component manages log transmission to the server, log grouping, and real-time logging.
Lastly, an Express server and client dashboard provide a user-friendly interface for users and researchers. These tools allow for simulation replication, real-time monitoring via a web browser, and more, offering an interactive and dynamic system interaction experience.

5.1.2 Rest Services Class Diagrams

To bolster system scalability and resilience, we have devised a series of application programming interfaces (APIs) within a Flask server. These APIs ensure the inception of vital services required for simulation and emulate various scenarios of hardware heterogeneity in federated machine learning. Three principal POST REST APIs, each serving a unique function, form the core of this system as seen in Figure 5.1.3:

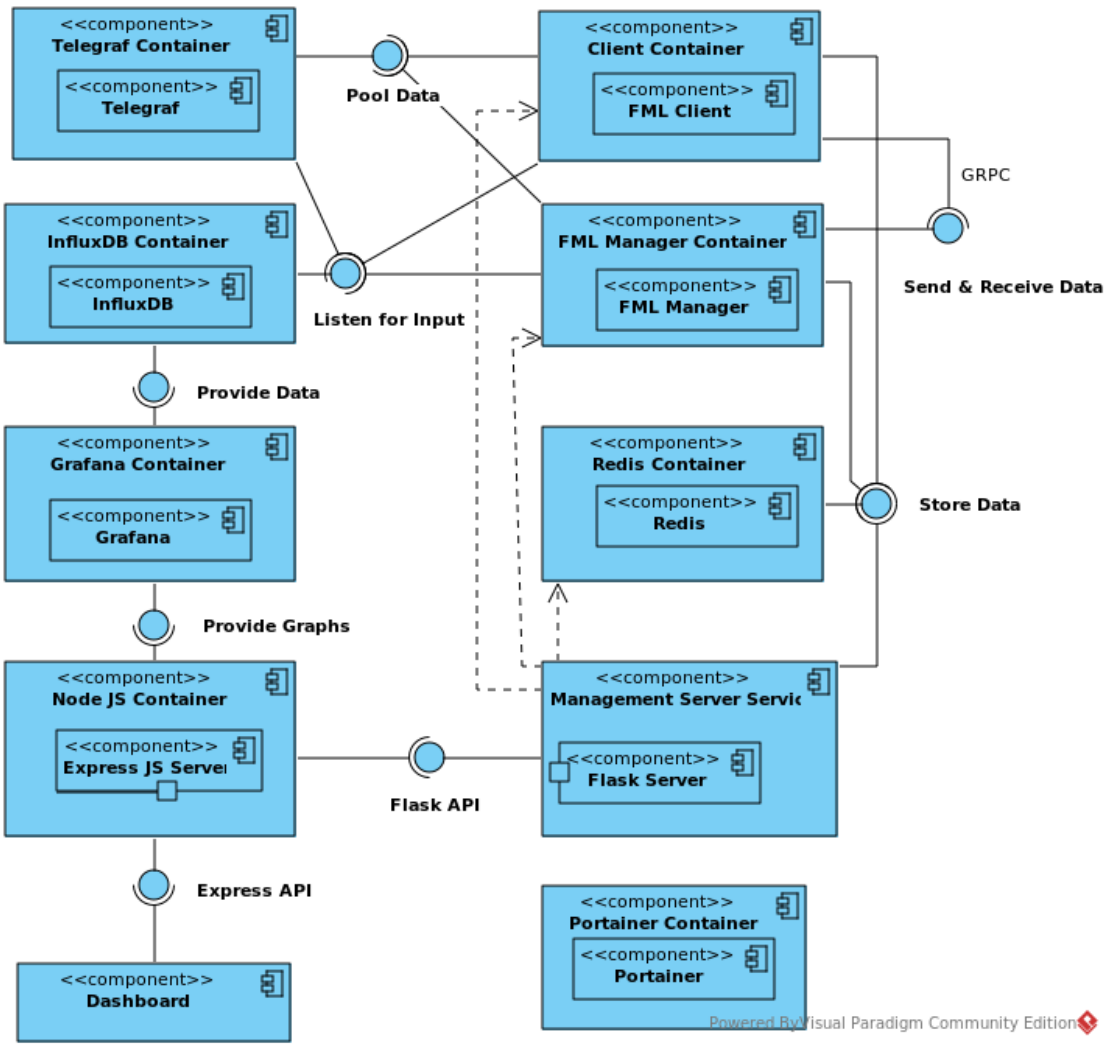


Figure 5.1.2: Component Diagram

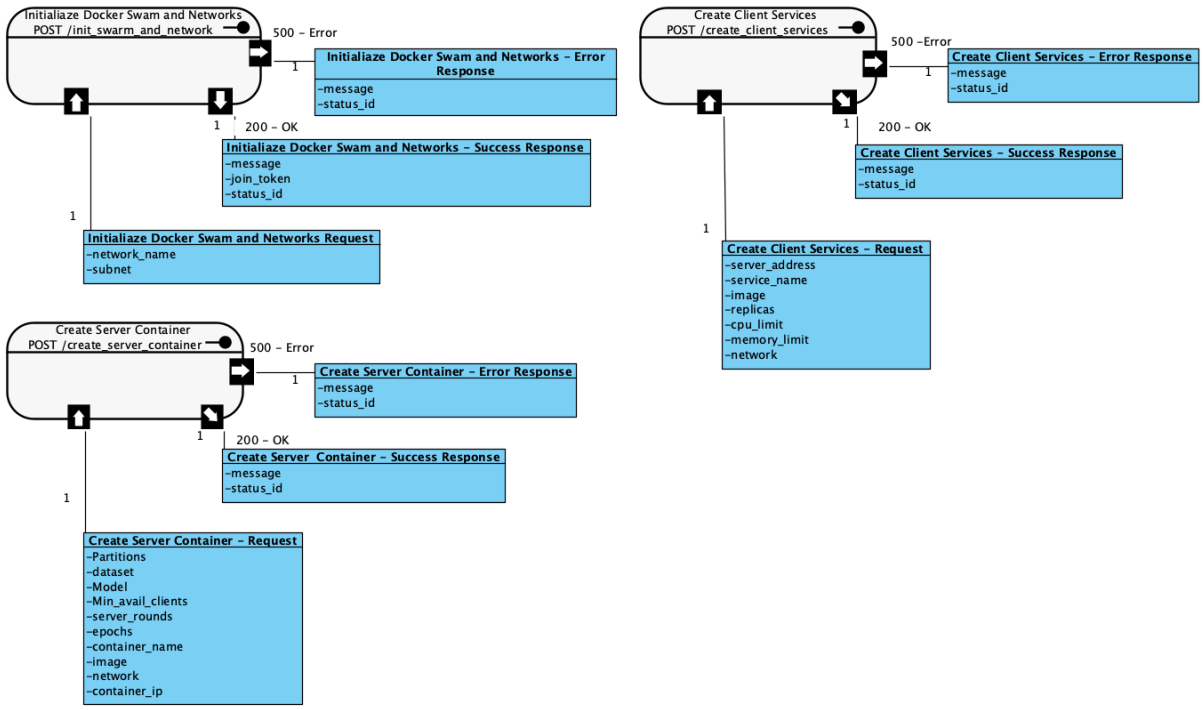


Figure 5.1.3: Rest Services Class Diagrams

1. Initialize Docker Swarm and Networks API:
   (a) Description: This API requires specific input parameters such as the intended network name for creation and the subnet available for container utilization. It delivers a success message, a status ID of 200, and a join token required for clients to connect to the swarm upon successful completion of its tasks.
   (b) Functionality: The API's primary tasks include initializing the Docker Swarm, creating necessary Docker networks, activating the Monitoring System, and launching the Redis DB Container.
   (c) Purpose: The API lays the groundwork for the federated machine learning simulation by setting up the foundational infrastructure. This includes network creation, enabling a monitoring system, and initiating a database for efficient data management.
2. Create Server Container API:
   (a) Description: This API requires various input parameters to start the container that hosts the FML server. Upon successful execution, it returns a status message and ID.
   (b) Functionality: The API's main role is to create the necessary volumes for the FML server and connect the appropriate network.
   (c) Purpose: The API ensures the server's readiness to manage the federated learning workflow effectively. It sets up the server to coordinate the FML tasks, maintain connectivity, and carry out training and testing processes.
3. Create Client Services API:
   (a) Description: This API is designed to fabricate 'N' groups of clients with different configurations, including variable CPU and Memory limits for each container. Upon successful execution, the API returns a status message and ID, thus confirming the successful creation and configuration of client groups.
   (b) Functionality: The API is tasked with generating and managing client containers, allowing them to participate in the federated learning process. This includes the ability to emulate different types of devices, each with its own computational capacity and memory constraints.
   (c) Purpose: The API facilitates the simulation of diverse device clusters, crucial for studying the impact of hardware heterogeneity on federated learning performance. It affords a nuanced understanding of how varying hardware configurations influence the behavior and performance of federated machine learning models.

The APIs enable the launch of 'N' services, where each service operates 'R' replicas of containers with identical resource constraints. This design yields extensive scalability; we can generate an arbitrary number of client devices grouped by their resource configurations.

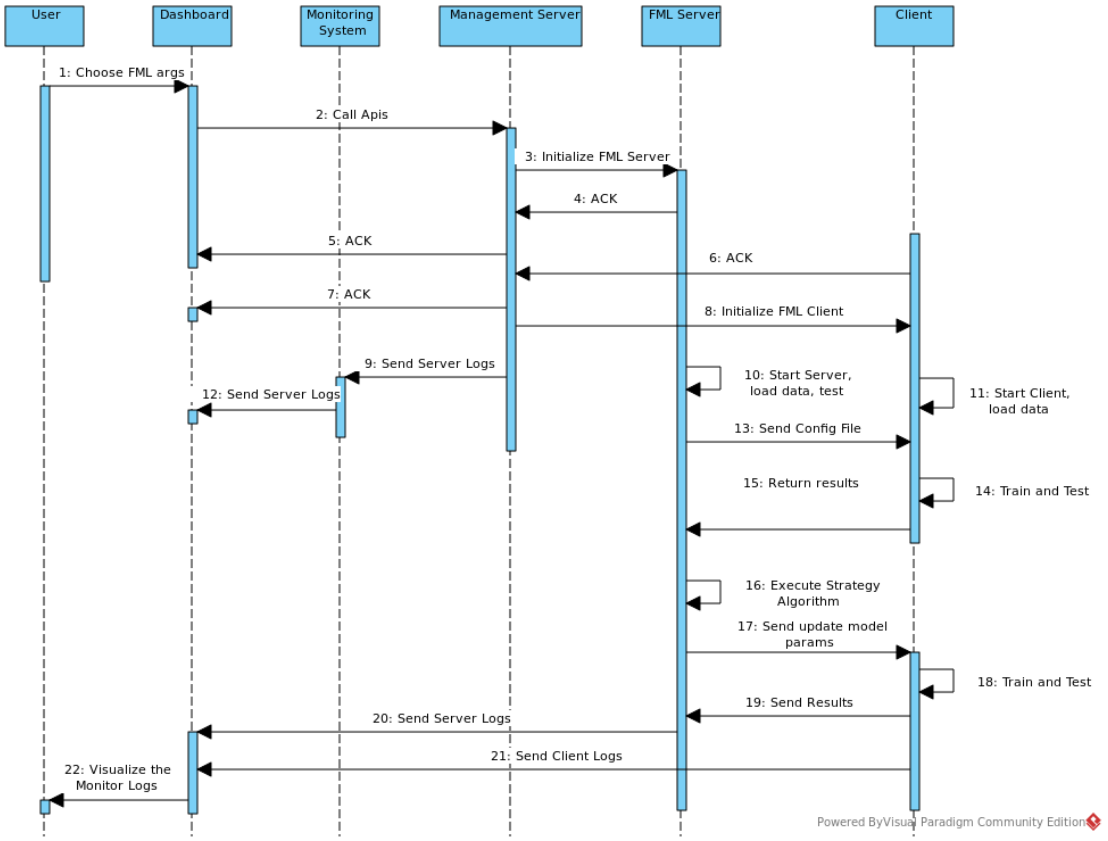


Figure 5.1.4: Sequence Diagram

In order to simulate a broader variety of hardware configurations, we can create an equal number of each resource-constrained container. This decision facilitates the depiction of a truly heterogeneous hardware landscape, encapsulating a multitude of resource configurations. Consequently, our simulation can closely mimic real-world hardware heterogeneity, providing valuable insights into the performance and behaviour of federated machine learning models under diverse hardware conditions.
Overall, these APIs play an indispensable role in the set up of the federated learning environment, facilitating the creation of a swarm, provisioning and configuration of the server container, and simulation of multiple clusters of varied devices. This approach allows us to model a network of multiple clusters of devices, each potentially possessing unique characteristics, thereby enhancing our understanding of the performance dynamics of federated learning in a heterogeneous environment.

5.1.3 Sequence Diagram

The accompanying sequence diagram 5.1.4 provides a comprehensive, high-level overview of our federated machine learning paradigm, detailing the integral components that ensure its effective operation.
Initially, a User—typically a researcher—selects the parameters of the paradigm they

wish to simulate via the Dashboard interface. Following this, the User Interface (UI) communicates with our REST APIs, connecting to the Management Server to instigate the entire federated machine learning procedure.
Subsequently, the Server initiates the Federated Machine Learning (FML) Server, which houses the Flower scripts. It then dispatches the appropriate response, affirming the commencement of the procedure, and ensuring all necessary elements are in place.
Concurrently, the Server activates the specified clients, each of which relays a confirmation indicating their operational readiness.
The diagram further elucidates the unfolding federated machine learning process, illustrating the synchronised operations of the FML Server and Client. The Server and Clients execute their respective procedures, load their data, and the Clients acquire the Server's configuration file. The Clients commence their training and local testing phases while the Server accumulates the Clients' data, consolidates the results, and performs global dataset tests. Following each round, the Server disseminates the global model parameters to the Clients.
Upon the completion of the paradigm, the monitoring system procures logs from these components and presents the results on the Dashboard. The culmination of the procedure yields a display of aggregated data and the generation of corresponding graphical representations.

5.1.4 Deployment Diagram

Presented in Figure 5.1.5 is the deployment diagram delineating the architecture of our Federated Machine Learning (FML) solution. Given the complexity and challenges associated with conducting paradigm simulations in a genuinely distributed system, we have elected to deploy our initial setup within a single server. In this setup, each FML component operates within its own distinct container, thereby simulating the disparate environments that would be encountered in a more distributed context. In this arrangement, containers are utilized as isolated environments wherein each component can operate independently of others. This simulates the condition of a distributed system and allows for individualized management of each component, while still benefiting from the cohesive oversight provided by a singular server.
Despite this, it's essential to note that the aforementioned configuration primarily serves as a stepping stone towards a more realistic and complex implementation. In real-world scenarios, as demonstrated in the Figure 5.1.6, the deployment architecture is significantly more distributed, with clients operating from separate devices.
Each device, effectively functioning as an individual client, interacts independently with the federated machine learning system. This embodies the true essence of federated learning, as it represents a system where multiple entities, or 'clients,' can collectively contribute to, and benefit from, a machine learning model while maintaining data locally
The ability to transition smoothly from a contained, server-based deployment to a fully distributed implementation is a testament to the flexibility and scalability inherent in our federated machine learning solution. As we continue to refine and expand our system, it's anticipated that further complexity and variation will be integrated into our deployment architecture, ensuring our system remains adaptable to a wide array of potential

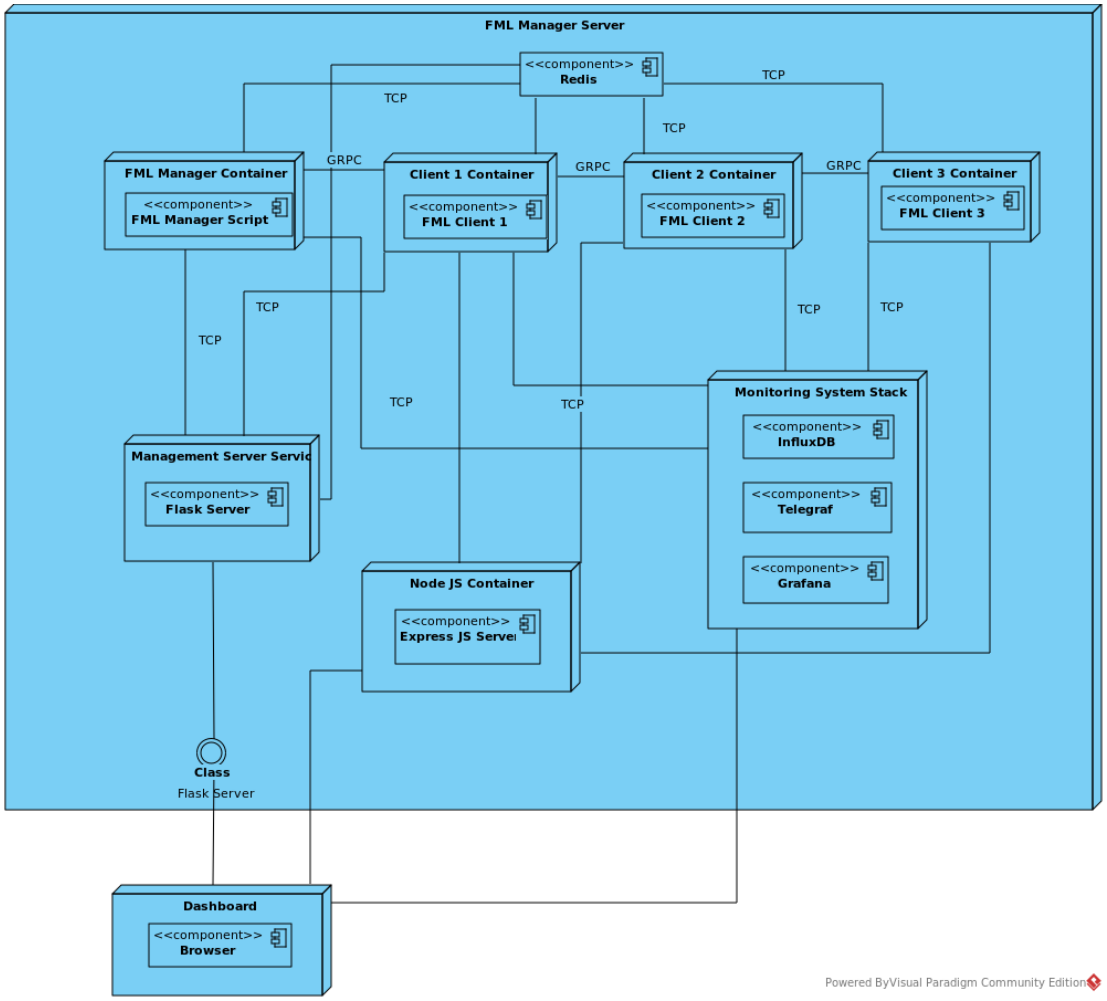


Figure 5.1.5: Deployment Diagram in a Server

applications and scenarios.

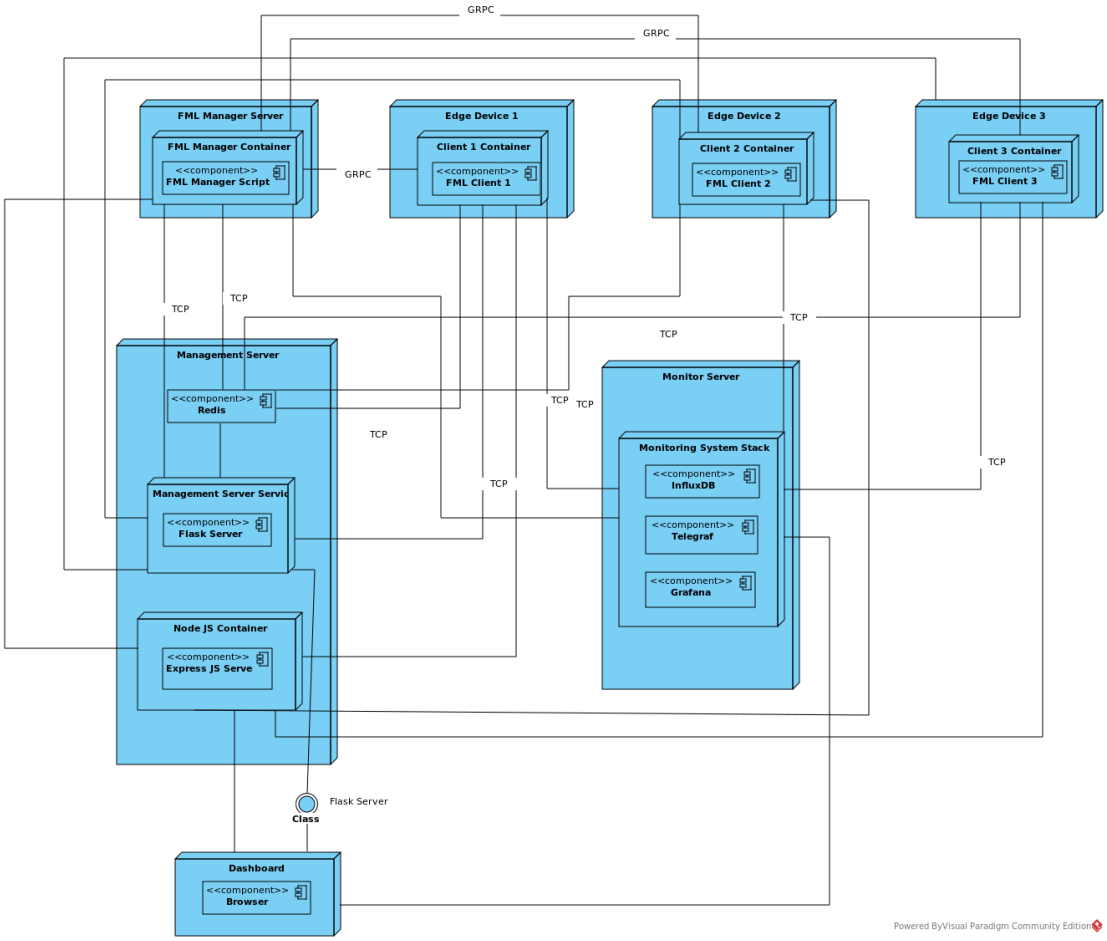


Figure 5.1.6: Deployment Diagram Decentralized

5.1.5 Class Diagram

The Flower Framework serves as the fundamental architecture upon which the Federated Machine Learning (FML) implementation is built. Our solution adopts this blueprint and enhances it by extending several abstract classes, including Server, Strategy, Client-Manager, Criterion, and Client. This approach allows for the complete customization of the FML paradigm according to our needs.

Additionally, we have implemented robustness measures by creating a custom datasets class and a monitor class. The latter is crucial for data accumulation throughout the training process.

The Criterion class is where we establish the code for client selection in each training or evaluation round. The criteria can be as simple as "all clients log in" for straightforward FML paradigms. In cross-device scenarios, it may involve filtering based on the device's

battery status, excluding in-use devices, or even potentially malicious devices.

The ClientManager is responsible for managing low-level connectivity aspects. It monitors clients, updates their statuses, and unregisters them if they become inactive.

The Server class is pivotal in organizing the paradigm from a system's perspective. It manages client log-ins, sets the strategy for the paradigm to follow, and coordinates client fitting and evaluation in either a decentralized or centralized manner. In the context of FEDMA, the Server class has been modified to handle the complexities of FEDMA training. The Client class carries out tasks at the edge. It handles training,

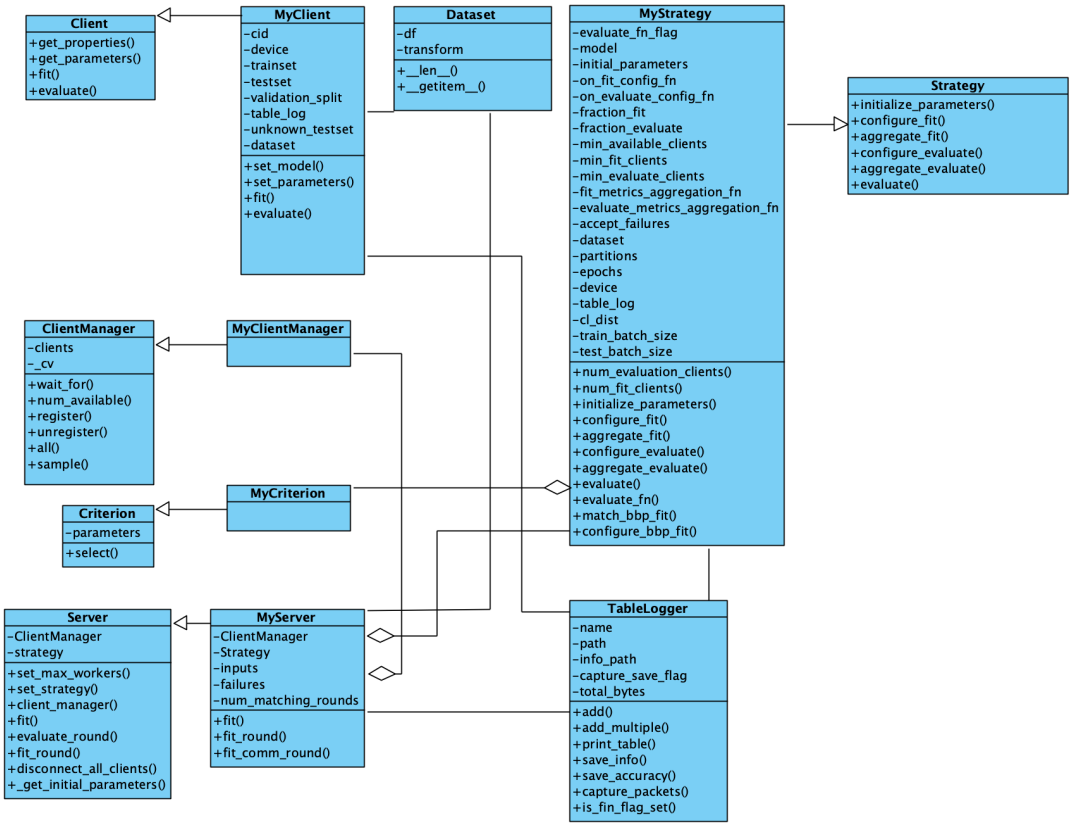


Figure 5.1.7: Class Diagram

testing, and the procedures needed for data loading and logging.

The Strategy class is invoked by the Server to define the trajectory of the FML paradigm. This class configures all clients before and after fitting and evaluation, managing the aggregation of model parameters.

The custom Dataset class facilitates the splitting of datasets into partitions, devising ways to simulate class heterogeneity and unify dataset loading.

The TableLogger class plays a key role in managing the logs produced throughout the FML paradigm. It is tasked with saving these logs and transmitting them to InfluxDB

for real-time presentation.

5.2 System description

The system is running on an Intel(R) Xeon(R) Silver 4208 CPU @ 2.10GHz which supports 64-bit architecture, with 8 CPU cores each with a single thread. This setup is an instance running on a VMware virtualization platform with full virtualization capabilities.
The CPU has a three-level cache with a total size of 256 KiB for L1d (data) cache, 256 KiB for L1i (instruction) cache, 8 MiB for L2 cache, and 88 MiB for L3 cache. Each of the 8 cores has an instance of these caches.
In terms of memory, the system has 62GB of RAM, of which 48GB is currently free. This should be more than sufficient for most computing tasks. The swap memory, which is used when the system runs out of RAM, is 8GB in size and it's almost entirely free.
If the code, along with its associated data, occupies 2.8GB, and the containers for monitoring consume approximately 1GB, then remains a total of around 3.8GB of storage being used.

5.3 Monitoring

In order to gather real-time information on our federated machine learning paradigm, such as the CPU usage, memory usage, and the number of connected devices, we make use of a trio of powerful tools: Telegraf, InfluxDB, and Grafana. Telegraf is an open-source server agent for collecting and reporting metrics. It provides us with real-time data collection of all relevant metrics from our Docker containers and connected devices. Its plugin-driven architecture supports data extraction from a variety of sources, making it a versatile choice for our needs.

InfluxDB is a high-performance time-series database. It stores the data collected by Telegraf, effectively serving as our system's memory. InfluxDB is designed to handle high write and query loads, making it well suited for storing large amounts of time-sensitive data.
Finally, Grafana is an open-source platform for monitoring and visualization. It retrieves data from InfluxDB and displays it in meaningful and interactive graphs and dashboards. This enables us to monitor our system's operation in real-time and facilitates the identification of any potential issues or bottlenecks.
By leveraging these tools, we are able to efficiently monitor our system's performance, swiftly address issues, and improve the overall efficacy of our federated machine learning paradigm.
By leveraging these tools, we are able to efficiently monitor our system's performance, swiftly address issues, and improve the overall efficacy of our federated machine learning paradigm.
For visualization of our simulation results, we employ InfluxDB queries. The data for

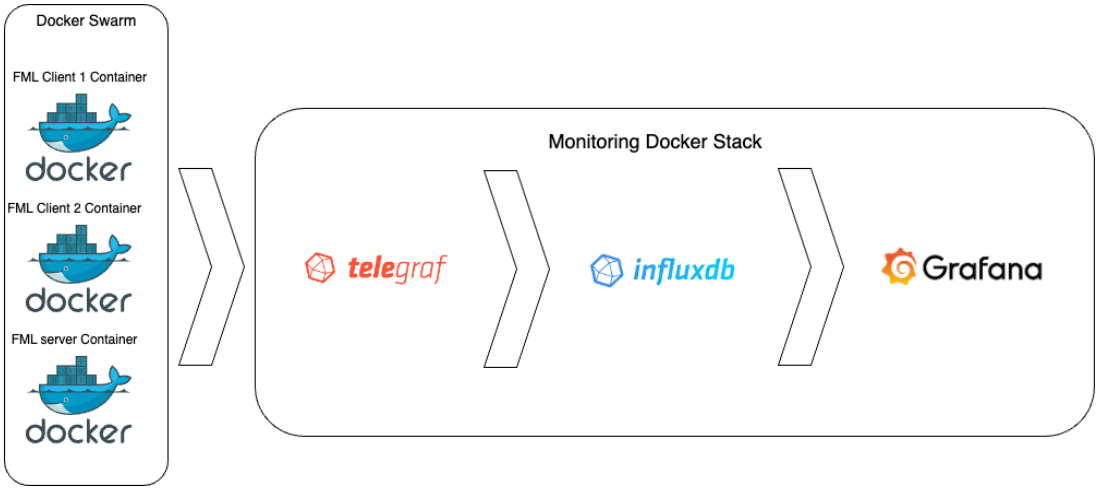


Figure 5.3.1: Monitoring System

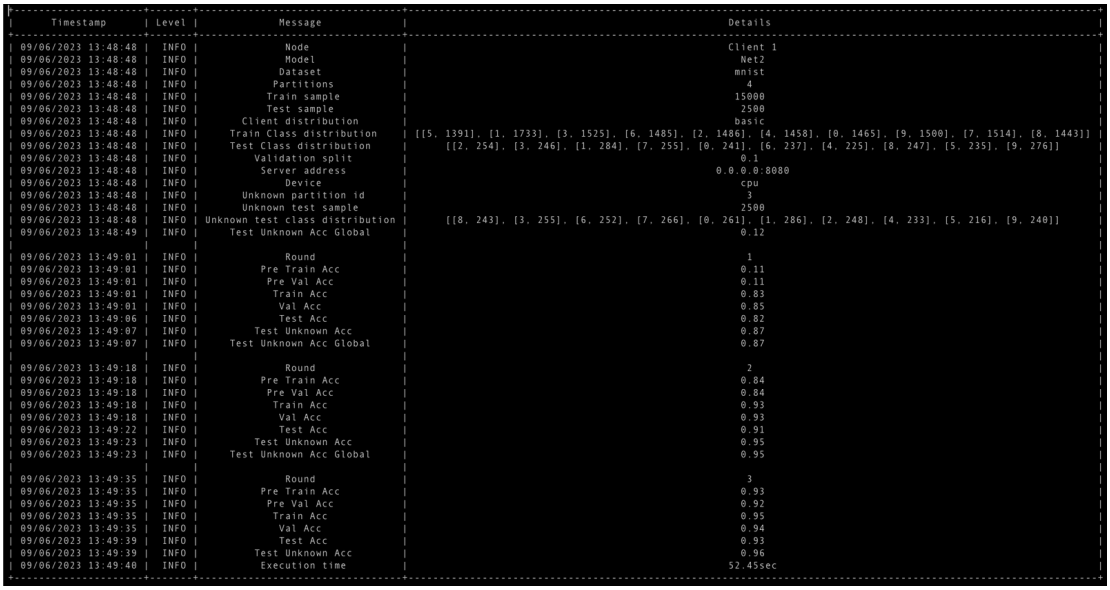
```
from(bucket: "my-bucket")
|> range(start: v.timeRangeStart, stop: v.timeRangeStop)
|> filter(fn: (r) => r["com.docker.swarm.service.name"] == "fml-service-server-1")
|> filter(fn: (r) => r["_field"] == "started_at" or r["_field"] == "usage_percent")
|> filter(fn: (r) => r["_measurement"] == "docker_container_mem")
|> aggregateWindow(every: v.windowPeriod, fn: mean, createEmpty: false)
|> yield(name: "mean")
```

Figure 5.3.2: Influx example query code

these queries is sourced from an InfluxDB database, which is continuously updated via the monitoring system, tracking the metrics of the Docker containers, and our custom logging mechanism within the client system which records and emits logs in real time. The sample query presented demonstrates how we extract meaningful information from the large stream of data we collect. The query performs several functions - it narrows the time scope of the data, filters out irrelevant data, and further refines the data based on specific parameters like service name, specific fields, and measurements related to Docker container memory.

The query 5.3.2 then proceeds to aggregate the filtered data based on a predetermined window period, producing an average (mean) value for each window. The end result of this query execution provides a focused snapshot of memory usage over time for the specified service, which can be crucial for performance monitoring and optimization strategies.

Also throughout the training and testing process of the clients, logs are are kept and saved to be easily read with an easy to read human format like the Figure 5.3.3:



Timestamp	Level	Message	Details
09/06/2023 13:48:48	INFO	Node	Client 1
09/06/2023 13:48:48	INFO	Model	Net2
09/06/2023 13:48:48	INFO	Dataset	mnist
09/06/2023 13:48:48	INFO	Partitions	4
09/06/2023 13:48:48	INFO	Train sample	15000
09/06/2023 13:48:48	INFO	Test sample	2500
09/06/2023 13:48:48	INFO	Client distribution	basic
09/06/2023 13:48:48	INFO	Train Class distribution	[[5, 1391], [1, 1733], [3, 1525], [6, 1485], [2, 1486], [4, 1458], [0, 1465], [9, 1500], [7, 1514], [8, 1443]]
09/06/2023 13:48:48	INFO	Test Class distribution	[[2, 254], [3, 246], [1, 284], [7, 255], [0, 241], [6, 237], [4, 225], [8, 247], [5, 235], [9, 276]]
09/06/2023 13:48:48	INFO	Validation split	0.1
09/06/2023 13:48:48	INFO	Server address	0.0.0.0:8080
09/06/2023 13:48:48	INFO	Device	cpu
09/06/2023 13:48:48	INFO	Unknown partition id	3
09/06/2023 13:48:48	INFO	Unknown test sample	2500
09/06/2023 13:48:48	INFO	Unknown test class distribution	[[8, 243], [3, 255], [6, 252], [7, 266], [0, 261], [1, 286], [2, 248], [4, 233], [5, 216], [9, 240]]
09/06/2023 13:48:49	INFO	Test Unknown Acc Global	0.12
09/06/2023 13:49:01	INFO	Round	1
09/06/2023 13:49:01	INFO	Pre Train Acc	0.11
09/06/2023 13:49:01	INFO	Pre Val Acc	0.11
09/06/2023 13:49:01	INFO	Train Acc	0.83
09/06/2023 13:49:01	INFO	Val Acc	0.85
09/06/2023 13:49:06	INFO	Test Acc	0.82
09/06/2023 13:49:07	INFO	Test Unknown Acc	0.87
09/06/2023 13:49:07	INFO	Test Unknown Acc Global	0.87
09/06/2023 13:49:18	INFO	Round	2
09/06/2023 13:49:18	INFO	Pre Train Acc	0.84
09/06/2023 13:49:18	INFO	Pre Val Acc	0.84
09/06/2023 13:49:18	INFO	Train Acc	0.93
09/06/2023 13:49:18	INFO	Val Acc	0.93
09/06/2023 13:49:22	INFO	Test Acc	0.91
09/06/2023 13:49:23	INFO	Test Unknown Acc	0.95
09/06/2023 13:49:23	INFO	Test Unknown Acc Global	0.95
09/06/2023 13:49:35	INFO	Round	3
09/06/2023 13:49:35	INFO	Pre Train Acc	0.93
09/06/2023 13:49:35	INFO	Pre Val Acc	0.92
09/06/2023 13:49:35	INFO	Train Acc	0.95
09/06/2023 13:49:35	INFO	Val Acc	0.94
09/06/2023 13:49:39	INFO	Test Acc	0.93
09/06/2023 13:49:39	INFO	Test Unknown Acc	0.96
09/06/2023 13:49:40	INFO	Execution time	52.45sec

Figure 5.3.3: Example of logs

5.4 Jupyter To Federation

The field of Federated Machine Learning (FML) is currently facing a reproducibility crisis that hinders its scientific progress. This crisis primarily stems from custom data partitioning practices by researchers, where many hyperparameters are often not fully specified and intricate implementation details are overlooked or omitted in the published reports. As a consequence, replicating and verifying the existing results requires a substantial amount of effort, often involving extensive detective work to infer missing methodological details. This lack of clarity and transparency poses significant challenges to the scientific community attempting to reproduce previous studies' findings or build upon prior work. These issues underline the critical need for more rigorous reporting standards in FML research, including clear and comprehensive documentation of data partitioning strategies, hyperparameters, and other relevant implementation details. Addressing this reproducibility crisis will foster a more robust and cumulative science of FML, accelerating advancements in this promising field.

Furthermore, a significant challenge in contemporary machine learning research is the difficulty of transitioning from theoretical models to live applications, testing models with real-world data, and establishing pipelines to assess the robustness of these models. This issue is even more pronounced in the context of Federated Machine Learning (FML), where the distributed and privacy-preserving aspects introduce additional complexities. Despite recent growth in the field, research is advancing slowly, compounded by issues such as the difficulty of reproducing the code featured in academic papers.

To address these challenges, we have combined the above software components and logic into a User Interface (UI) dashboard that is a form like the Figure 5.4.1. This platform

enables researchers to train, test, monitor, and even customize the FML process.

The dashboard includes a form from which users can select various parameters to alter in an FML paradigm, thereby facilitating customizable simulations. Through our UI, users can monitor, in real-time, model accuracies, training loss, and testing results on diverse datasets, among other metrics. Additional details such as CPU usage, memory usage of the clients and server, and the volume of data transmitted between server and clients can also be tracked. This comprehensive overview enhances the user's understanding and control of the FML process, promoting more effective research and development.

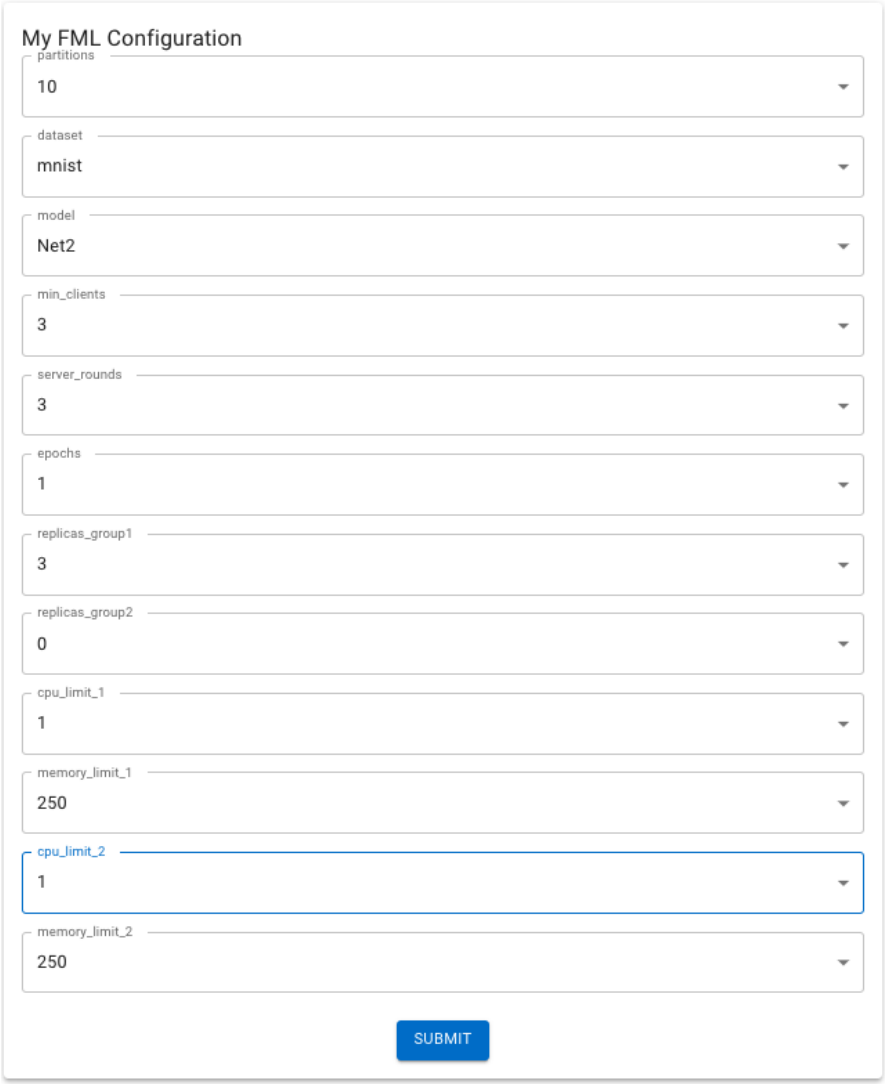


Figure 5.4.1: UI interface

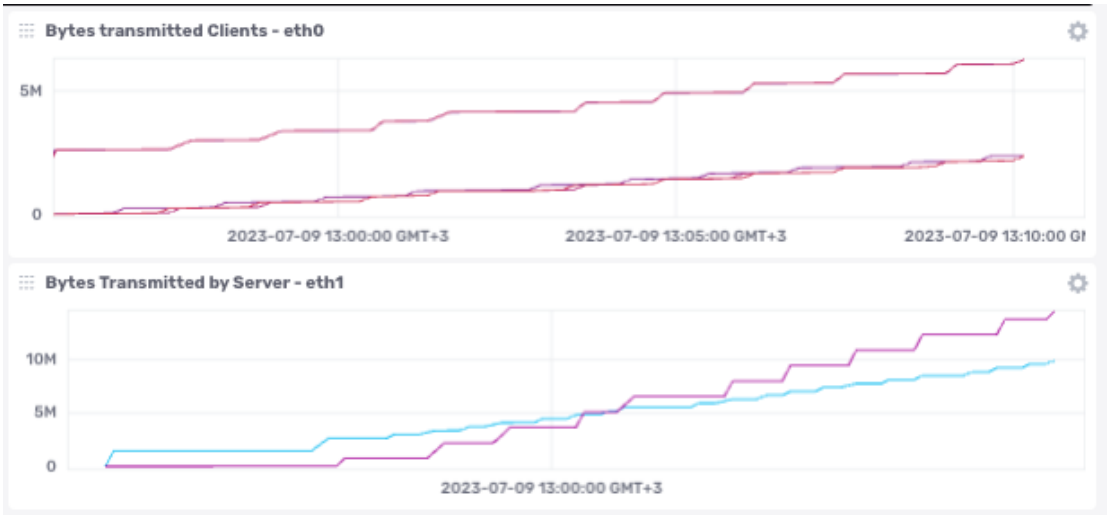


Figure 5.4.2: Real Time Network Monitoring

Figure 5.4.3: Real Time Visualisation of Clients' Accuracy and Loss

Figure 5.4.4: Real Time System Monitoring of Clients

Chapter 6

Experimental results

6.1 Datasets

6.1.1 MNIST

The MNIST dataset [35] is a well-known collection of handwritten digits that is widely used in the field of machine learning for image classification tasks. It contains 70,000 grayscale images of handwritten digits from 0 to 9, with 60,000 images used for training and 10,000 images used for testing. Each image is of size 28x28 pixels, and has been normalized and centered to have a similar size and position.

To standardize the datasets, we use the mean and standard deviation of the training set to adjust each data sample. This is done by subtracting the average value from each sample and dividing it by its standard deviation. For neural networks that are designed to classify images in the MNIST dataset, the input layer size is determined by the number of pixels in each image. The goal of the classification task is to identify which of the ten digits (0 to 9) is present in each input image. Therefore, the size of the output layer in the neural network is set to 10, which corresponds to the ten possible digit classes.

In the simulations conducted, the MNIST dataset was segmented into distinct partitions, with each client receiving a unique partition as shown at 6.1.1. The design ensured that the data assigned to a specific client remained exclusive to that client and was not shared with or viewed by others. Furthermore, the server had the capability to evaluate the global model on a partition that is analogous to those held by the clients, on data unseen by the clients, or even on a client's specific partition. The server could also conduct tests on the entire dataset. This methodology facilitates a comprehensive evaluation of the model's performance under various data conditions while respecting the client-specific data segregation.

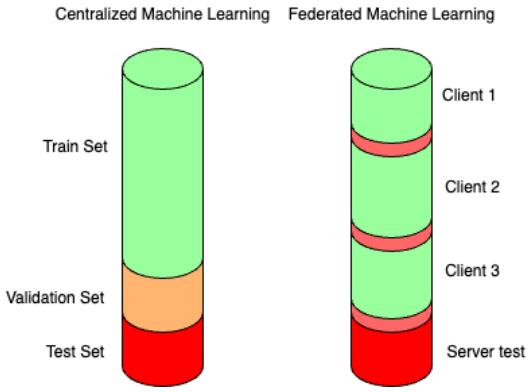


Figure 6.1.1: MNIST Dataset Split

6.1.2 FEMNIST

FEMNIST is a benchmark dataset from the LEAF framework used widely for federated learning [14]. FEMNIST stands for Federated Extended MNIST, and it is an extension of MNIST dataset. The FEMNIST dataset consists of 3,550 writers, with each writer contributing a set of handwritten digit and character images. There are over 80,000 total images, and each image is associated with a specific writer. On average every writer has contributed with 200 images with unbalanced multitude of samples. FEMNIST is unique in that it is designed to simulate a realistic, decentralized setting, where each writer's data is stored locally and only available to them.

In the interest of enhancing the relevance and applicability of the FEMNIST dataset to our specific solution, we initially retain only the numerical samples from the dataset and discard any letters. Subsequently, we calculate the variance of the remaining dataset for each user and segment the users into 10 clusters 6.1.2 to accumulate more data pertaining to the same handwriting style. The goal is to create clusters of users who have similar handwriting styles, which can then be used to assign clients in a federated paradigm. Each client will be assigned a cluster of users with similar handwriting styles, and will train their model on that cluster's data. By doing this, we hope to introduce variability in the handwriting styles across clients, making the task of federated learning more challenging. To demonstrate the heterogeneity of these clusters, we can visualize the variance of handwriting examples within each cluster. This will help us to better understand the impact of data heterogeneity on the performance of the federated learning algorithm.

The revised dataset we employ possesses nearly the same sample quantity as the MNIST dataset in both training and testing phases, approximating around 60,000. Each cluster comprises 60 users, totaling 6000 training sets and 1000 test sets. The disparity in training data across each cluster of users can be exemplified by the following visual representation showcasing 8 numerical characters from a randomly selected group of users as shown at 6.1.3 and 6.1.4.

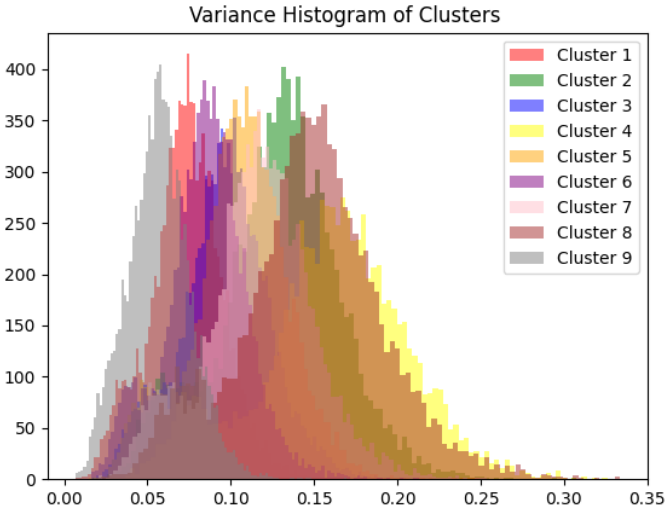


Figure 6.1.2: FEMNIST Users Cluster Groups

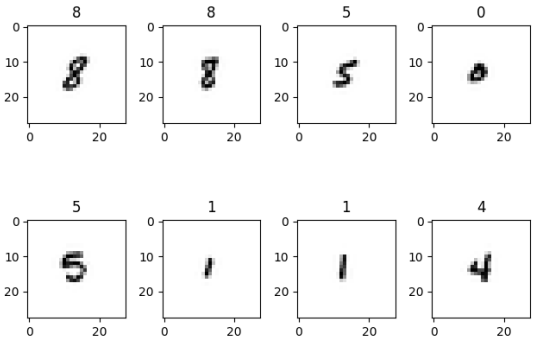


Figure 6.1.3: Group 1 hand writing style

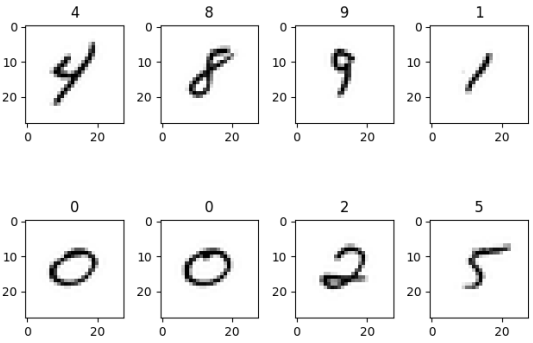


Figure 6.1.4: Group 2 hand writing style

6.2 Class/Data Heterogeneity

As we dive deeper into the world of Federated Machine Learning, one of the most critical aspects to consider is the heterogeneity of data across participating devices or nodes. In real-world scenarios, data is often distributed unevenly among the participants, varying in size, distribution, and quality. The challenge of dealing with such heterogeneity can lead to suboptimal or even incorrect models if not addressed properly.

Ultimately, Federated Machine Learning provides a robust framework for addressing data heterogeneity, allowing us to harness the full potential of diverse datasets while preserving privacy and maintaining the decentralized nature of the learning process. By embracing the heterogeneous nature of data, FML paves the way for more accurate and reliable machine learning models that can be applied to a wide range of domains and applications. Substantial research and progression have occurred in this domain, as evidenced in the study referenced in **??**.

Measuring data heterogeneity can help identify the degree of variation in the data and enable the selection of appropriate federated learning algorithms and strategies to mitigate the negative effects.
By measuring data heterogeneity, one can assess the suitability of the federated learning setup for a particular problem domain and optimize the training process to improve model performance.

The setup is challenging for FL algorithms, because each client only processes data from a single institution, which potentially suffers from more severe domain-shift and overfitting issues compared with a data-centralised training:

Also one of the biggest problems regarding data created "on the edge" is Domain shift. Domain shift refers to the difference in the distribution of data between the source domain (typically, the training dataset) and the target domain (usually, the testing or real-world deployment dataset). In machine learning, a model is trained on a specific

dataset with the assumption that the data distribution in the training set is representative of the data distribution in the real world. However, in practice, the data distribution in the real world can differ from the training data distribution, leading to a decrease in model performance when applied to the target domain. Domain shift can occur for various reasons, such as:

1. Changes in data collection methods
2. Different populations or demographics
3. Temporal changes (e.g., changes in trends, user behavior, or market conditions)
4. Different sensors or devices used for data collection

Addressing domain shift is an important aspect of machine learning research. Techniques such as domain adaptation, transfer learning, and domain-invariant feature learning aim to build models that are more robust to domain shift and can generalize better to different data distributions in the target domain.

In our experiments we simulate data heterogeneity by pre-processing our data sets and assigning each client with different class balances or with random number of examples.

In our experiments we will use four different ways of "splitting" the data sets of each client.

1. Basic (Balanced): Clients have an equal number of images, and the distribution of classes is balanced. This configuration can be achieved by evenly distributing the images among the clients, ensuring that each client receives an equal number of images from each class.
2. Imbalanced: Clients have a random number of images, but the distribution of classes is maintained. This configuration can be created by randomly assigning a different number of images to each client while ensuring that the class distribution remains the same within each client's dataset.
3. Skewed: Each client has the same number of images, but all the images come from a single class. This configuration can be obtained by dividing the images by class and assigning all images of a particular class to a single client, ensuring that each client has images from only one class.
4. Imbalanced and skewed: Clients have a random number of images, and each client has images from multiple classes.
5. Different handwriting in FEMNIST: This configuration can be created by selecting a subset of writers from the FEMNIST dataset, ensuring that each client has images from a different set of writers. This configuration introduces variability in the writing styles among clients, making the task more challenging.

In order to measure the class imbalance we calculate the distribution of each class in each clients' data set using the Distributional Heterogeneity metric of [60].

Where DH is the distributional heterogeneity metric, N is the number of classes, C is the number of clients, and cj is the number of clients that have at least one instance of class j in their local data. The function returns the DH metric.
DH is a metric between 0 and 100%, where DH = 0% means that our experiment has clients with the same number of examples from every class (IID) and DH = 100% where each client has examples from only one class (NON-IID). Some examples are:

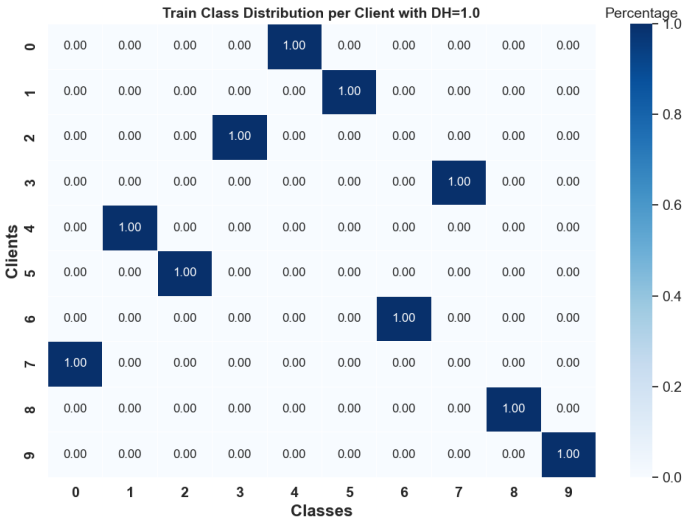


Figure 6.2.1: Distribution of Trainset with DH = 1.0

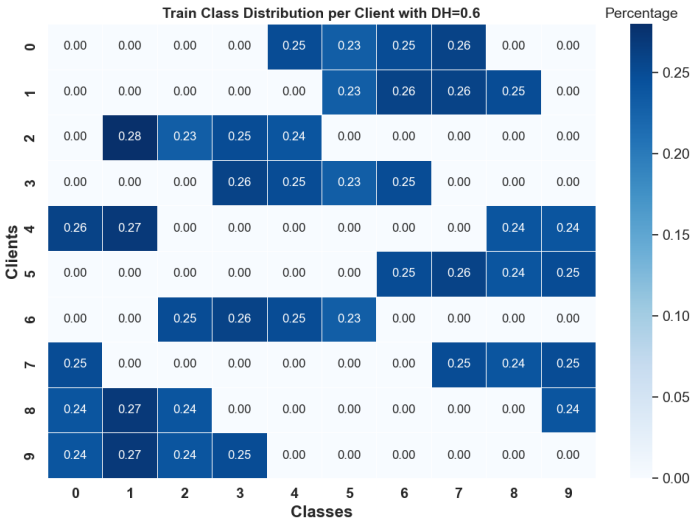


Figure 6.2.2: Distribution of Trainset with DH = 0.6

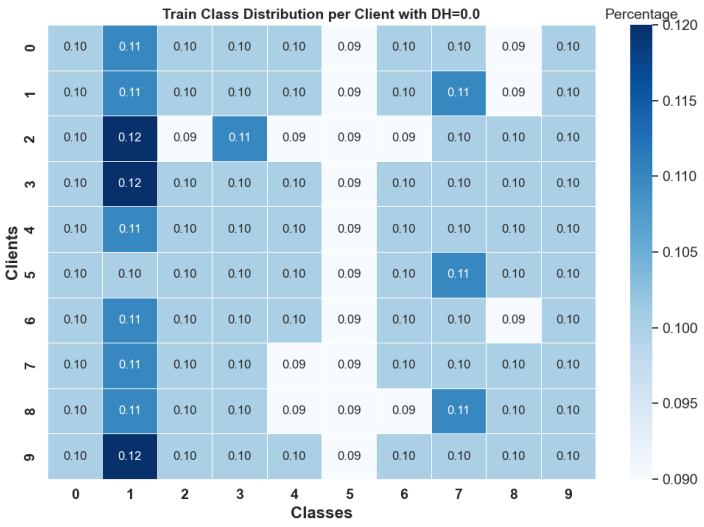


Figure 6.2.3: Distribution of Trainset with DH = 0.0

Homogeneity is a critical measure employed to evaluate the performance of clustering algorithms in machine learning. This metric captures the degree to which each cluster consists only of data instances from a single class, thus assessing the quality of the clustering task.

In an ideal scenario, a homogeneity score of 1 is achieved when each cluster contains data instances belonging to only a single class, implying that perfect clustering has been performed. Conversely, a homogeneity score of 0 indicates the least desirable state, where the class distribution within clusters is not significantly different from the overall class distribution in the dataset, signifying that the clustering has not effectively separated different classes.

6.3 Hardware Heterogeneity

In the realm of federated machine learning, hardware heterogeneity emerges as a critical factor that plays a decisive role in the overall process. The hardware configurations demonstrate substantial variations in cross-silo and cross-device use cases, each presenting unique challenges and opportunities for model building.
For cross-silo use cases, the machine learning models are deployed on-premises, utilizing the resources of corporate or institutional servers. These servers are often equipped with a high capacity of memory and powerful CPUs, along with other sophisticated hardware, making them an efficient choice for such scenarios.
On the contrary, in cross-device use cases, the hardware spectrum expands from microprocessors and Raspberry Pi devices to ubiquitous smartphones. Each device presents unique computational capabilities and constraints, thus underlining the vast

Table 6.1: Summary of Strategies tackling Data Heterogeneity

Work	Description	Advantages	Limitations
McMahan et. al. [40] FedAVG	[itemsep=0pt]Train using SGD Naive parameter averaging on the server	[itemsep=0pt]Preserves data privacy Reduces comm. costs Scalable for many devices Simple to implement	[itemsep=0pt]Struggles with non-IID data Slowed down by stragglers Lacks robustness
Zhao et. al. [63]	[itemsep=0pt]Creating a small globally shared subset of data between all clients The weight divergence is measured using the Earth Mover's Distance (EMD)	[itemsep=0pt]Improved accuracy with globally shared data Effective non-IID handling	[itemsep=0pt]Complex initial set-up Server dataset bias Sensitive data exposure Increased communication costs
Yuan et. al. [60] DisTrans	[itemsep=0pt]Optimizes distributional offsets and models for each client to shift their data distribution and aggregate them on the server. Uses train and test-time distributional transformations with a double input channel model structure.	[itemsep=0pt]Better performance without significant communication overhead. First to utilize test-time transformation to improve federated learning accuracy under data heterogeneity.	[itemsep=0pt]Difficulties in practical implementation due to complexity a Potential overfitting.
Li et. al. [37] MOON	[itemsep=0pt]Corrects the local updates by maximizing the agreement of representation learned by the local model and the global model.	[itemsep=0pt]Efficient learning with fewer communication rounds Scalability Robust to different levels of data heterogeneity Mitigates local update drift.	[itemsep=0pt]Lower accuracy with minimal local epochs. Initial training slowdown with large model-contrastive loss Performance relies on μ parameter.
Zhou .et .al [64] FEDFA	[itemsep=0pt]It introduces a concept called feature anchors to align the extraction of features and classifier updates across clients.	[itemsep=0pt]Effective under combined label/ feature distr. skews Significant advantage in homogeneous data Robustness and scalability.	[itemsep=0pt]Performance decrease under combined skews vulnerable to high local epochs.
Li et. al. [38] FedProx	[itemsep=0pt]Designed to tolerate "stragglers" or slow-performing clients	[itemsep=0pt]Addresses system and statistical heterogeneity Reduced comm. rounds. It tolerates devices dropping in and out of the network Robust to variations in the number of local updates.	[itemsep=0pt]The use of proximal terms in the optimization process increases the computational complexity of each iteration.
Karimireddy et. al. [31] SCAFFOLD	[itemsep=0pt]Addresses the issue of client-drift in local updates by using control variates for variance reduction	[itemsep=0pt]Efficient convergence Robust to data heterogeneity and client sampling Accelerated convergence with similar data.	[itemsep=0pt]Performance decreases by increasing the number of local training epochs.

hardware heterogeneity in these use cases.

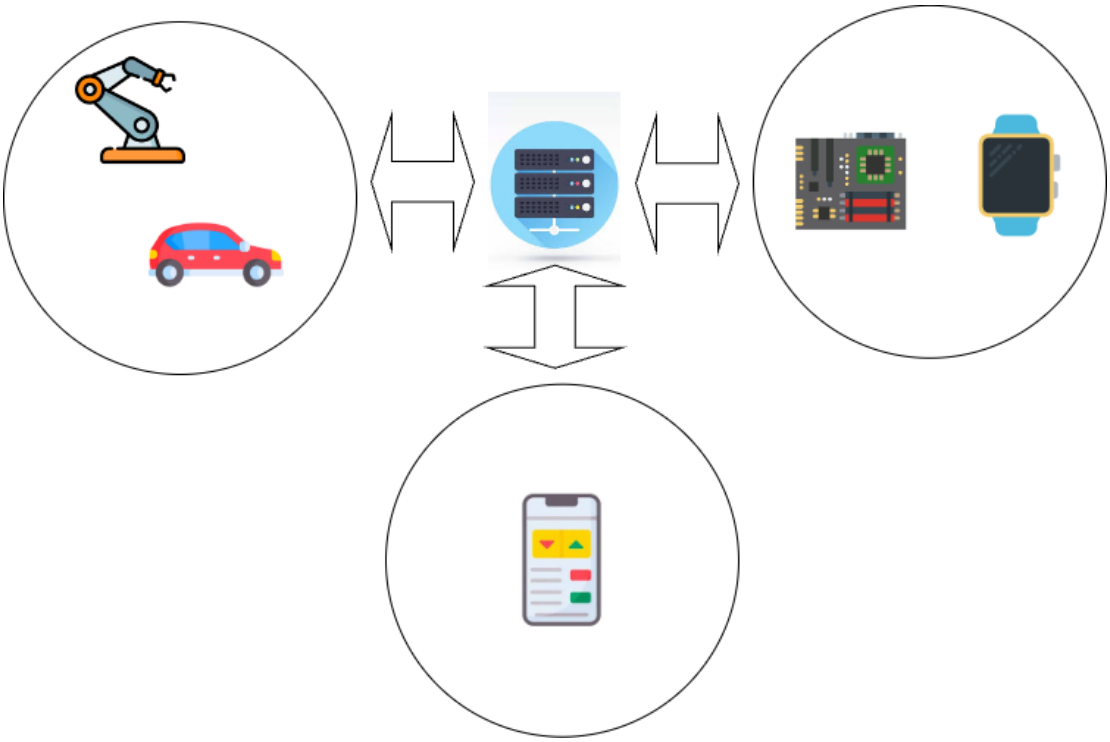


Figure 6.3.1: Hardware heterogeneity

The significant variations in hardware configurations present challenges in developing robust and versatile models that can adapt to the spectrum of computational capabilities.
To effectively tackle this issue and emulate hardware heterogeneity in our experiments, we have structured our code to be flexible and adaptable, governed by several configurable parameters.

Based on their hardware characteristics and resource constraints, some example devices can be clustered into low, medium, and high availability categories:

1. Low availability (resource-constrained):

   (a) Smartwatch: Given their limited processing power and memory, smartwatches fall into the low availability category. They require optimized software that can function efficiently under strict hardware constraints.

   (b) Microprocessor: Depending on the model and use case, some microprocessors can also be categorized as low availability. For instance, the microcontrollers used in small embedded systems are usually very resource-constrained.

2. Medium availability:

(a) Phone (Smartphone): With diverse hardware capabilities, smartphones typically fit in the medium availability category. They have more processing power and memory than smartwatches or low-power microprocessors, but less than high-powered devices like autonomous vehicles or industrial robot arms.

3. High availability:

(a) Car (Connected Cars/Autonomous Vehicles): Cars, specifically autonomous ones, need robust computational capabilities for their complex real-time tasks. They have high-powered CPUs and a considerable amount of memory, making them fall into the high availability category.

(b) Bionic Arm of a Factory (Industrial Robot Arm): These are optimized for specific tasks with real-time requirements. Industrial robot arms generally have high computational power and ample memory to perform complex operations, categorizing them as high availability.

To effectively tackle this issue and emulate hardware heterogeneity in our experiments, we have structured our code to be flexible and adaptable, governed by several configurable parameters.
Leveraging Docker container options, we can adjust the computational limits of each container in our paradigm, thereby simulating resource-constrained environments typically found in cross-device use cases. Our simulation approach specifically alters the maximum CPU limit and memory limit, simulating environments with diverse computational resources.
Furthermore, we manipulate parameters such as batch size, model size, and data size within each device. This adjustment allows us to explore how the federated machine learning paradigm functions when dealing with insufficient models and data sizes.
Lastly, to simulate realistic scenarios where clients, or 'stragglers,' intermittently connect and disconnect - a phenomenon possibly resulting from communication hitches, power disruptions, or other issues - we have devised our experiment to start and stop the containers at random intervals.
Our research strategy provides a comprehensive simulation of various conditions that arise due to hardware heterogeneity in federated machine learning. By doing so, we aim to explore, understand, and propose solutions to the challenges presented in this field, thus contributing to the development of more robust and adaptable machine learning models.

6.4 Metrics

6.4.1 Accuracy

Accuracy is a crucial metric that is used to evaluate the performance of the models in our federated machine learning simulations. Specifically, it is a ratio of the correctly predicted observations to the total observations. It provides a straightforward, understandable measure of a model's performance, offering a quick snapshot of the effec-

tiveness of a prediction. In the context of federated learning, where data is distributed across multiple nodes and the model is trained collectively on this data, accuracy gives an aggregated measure of how well the global model is performing.

The most reliable method to assess this accuracy is by testing the final model that the server constructs, combining all the updates from the client models, on a separate dataset that none of the client models have encountered before. This approach provides an unbiased evaluation of the model's generalization ability and its capacity to make accurate predictions on unseen data.

In addition to this centralized accuracy measurement, our tool also supports a decentralized testing approach. In this setup, a subset of clients is sampled to evaluate the global model on their local data. This decentralized evaluation provides additional insights into the performance of the model across different data distributions present in the federated learning system. It reflects the model's performance in real-world scenarios where predictions are often made on the individual nodes. However, it's important to bear in mind that this decentralized testing could be biased if the local data distributions are not representative of the overall data distribution. Therefore, a balanced combination of centralized and decentralized testing can give a more comprehensive understanding of the model's performance.

6.4.2 CFMQ

Federated Learning has gained prominence as a collaborative training approach for decentralized data, but evaluating its effectiveness in the context of hardware heterogeneity poses significant challenges.
Traditional deep learning metrics, including accuracy, precision, and recall, are commonly used to evaluate FL models. However, these metrics fail to provide a comprehensive assessment of FL's performance when confronted with diverse device characteristics. Consequently, there is a need for a comprehensive metric that accounts for hardware heterogeneity in FL evaluations.
To address this limitation, the Cost of Federated Model Quality (CFMQ) metric is used as a comprehensive measure that incorporates the considerations of communication costs, local computation costs, and their interplay with hardware heterogeneity. By utilizing CFMQ, researchers gain valuable insights into the efficiency and resource requirements of FL in the presence of diverse hardware environments.
While traditional deep learning metrics serve as important evaluation tools, they are insufficient for fully capturing the impact of hardware heterogeneity on the FL paradigm. The CFMQ metric offers a holistic approach to evaluating FL in diverse hardware settings, providing researchers with a comprehensive measure to assess the computational resource requirements and performance of FL systems.
The Cost of Federated Model Quality (CFMQ) is a novel metric that quantifies the computational resources required for a federated learning process. It is a comprehensive measure that takes into account both the communication cost (data transfer between the server and clients) and the local computation cost (processing done on each client). Using CFMQ allows for systematic and comparative assessment of different federated learn-

ing setups, enabling more efficient design and resource allocation. It aids in balancing trade-offs between local computation and data communication, which can significantly impact the efficiency, speed, and feasibility of federated learning processes, especially in large-scale or resource-constrained scenarios.

1. R (Number of rounds): R denotes the quantity of communication iterations between the central server and the associated clients throughout the federated learning procedure.

2. P (Communication Payload): P symbolizes the magnitude of the data (in bytes) reciprocally exchanged between the server and each client during each communication round.

3. α (Balancing term): α is a modifier that adjusts the relative weight of the local computational cost, and can be fine-tuned to match the specific system requirements or constraints.

4. m_i (Average number of local optimization steps): m signifies the average count of computations a client executes during the process, calculated as $m = \frac{eN}{bK}$, where e is the number of local epochs, N is the total number of examples in a round, b is the batch size, and K is the number of clients.

5. ν (Peak memory consumed during a step): ν represents the peak memory consumption during a local computation step on a client.

6. K (Number of clients participating): K denotes the total count of clients participating in the federated learning process.

7. b (Batch size): b refers to the grouping of examples processed collectively during the learning operation.

8. N (Total number of examples in a round): N symbolizes the complete count of examples or data points utilized in each round of learning.

9. e (Number of local epochs): e represents the number of comprehensive passes through the complete dataset made on each client.

Consequently, these parameters are combined to compute the CFMQ, which estimates the necessary computational resources for a federated learning operation. The CFMQ is expressed as:

$$\text{CFMQ} = RK(P + \alpha\mu\nu)\,[\text{bytes}]$$

In order to make this metric more adapted in our paradigm we make the below changes.

1. P_i (Communication Payload per client): P_i symbolizes the magnitude of the data (in bytes) exchanged between the server and each i-th client during each communication round.

2. m_i (Modified Average number of local optimization steps per client): m_i signifies a modified version of μ. It accounts for the average count of computations a client executes during the process, now calculated as:

$$m_i = e \cdot \left(\frac{N_{\text{train}_i} + N_{\text{test}_i}}{B_{\text{sz\_train}_i} + B_{\text{sz\_test}_i}} \right),$$

where e is the number of local epochs, N_{train_i} and N_{test_i} are the total number of training and testing examples for the i-th client in a round, and $B_{\text{sz\_train}_i}$ and $B_{\text{sz\_test}_i}$ are the batch sizes for the i-th client during training and testing phases, respectively.

3. v_i (Peak memory consumed during a step per client): v_i represents the peak memory consumption during a local computation step on the i-th client.

Incorporating these new definitions, the CFMQ is recalculated for each client and then summed over all clients. The new CFMQ is expressed as:

$$\text{CFMQ} = R \cdot \sum_i (P_i + \alpha \cdot m_i \cdot v_i) \quad \text{for all clients}$$

By breaking down the calculation per client, the new CFMQ allows for more flexibility and precision when clients have different parameters. This approach is beneficial when dealing with federated learning in heterogeneous environments, which is often the case in real-world scenarios.

6.4.3 Training time

Training time is another critical factor when assessing the efficiency of our federated machine learning simulations. It quantifies the amount of time taken for a model to learn from the distributed data, with the learning process typically involving multiple iterations or epochs. Shorter training times are generally more desirable, particularly in environments where quick decision making is paramount or where computational resources are limited. It is worth noting that in the context of federated learning, training time not only includes the time taken to train the model on each node but also the time taken to communicate and aggregate the model updates. The training time can provide insights into the scalability of the federated learning system and can help identify bottlenecks, such as network latency, straggler nodes, or computational inefficiencies. As such, optimizing training time without compromising the model's accuracy is a key challenge in federated learning.

6.5 Simulations

6.5.1 Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNNs) are a class of deep learning models, especially apt for processing data with a grid-like structure, such as images. They draw inspiration from the organization of the animal visual cortex and are specifically designed to

automatically and adaptively learn spatial hierarchies of features, ranging from low- to high-level patterns.

CNNs are typically comprised of three core types of layers: convolutional layers, pooling layers, and fully connected layers. Convolution and pooling layers handle feature extraction, while fully connected layers map these extracted features into the final output, like classification or prediction.

The convolution layer is the pivotal component of CNN. It involves a stack of mathematical operations, with convolution being a specialized form of a linear operation. In digital images, pixel values are organized in a two-dimensional (2D) grid, i.e., an array of numbers. A small, optimizable grid of parameters known as a kernel or filter is systematically applied across this grid. The kernel operates as a feature extractor, making CNNs highly efficient for image processing since a particular feature can occur anywhere in the image.

Convolution layers are often followed by non-linear activation functions, like the Rectified Linear Unit (ReLU), which introduces nonlinearity into the model, enhancing its learning capability. Pooling layers, on the other hand, progressively reduce the spatial size of the input, controlling the number of parameters and computational complexity while preventing overfitting.

As the output of one layer feeds into the next, the extracted features hierarchically and progressively become more intricate and complex, effectively capturing the spatial relationships in the input data. This unique architecture of CNNs allows the detection of intricate patterns and features, irrespective of their location in the image, making them remarkably scalable and efficient.

The process of optimizing parameters, such as kernels, is referred to as training. The goal is to minimize the difference between the model's outputs and the ground truth labels. This optimization is usually achieved using algorithms like backpropagation and gradient descent, among others. In essence, CNNs automatically learn the most relevant features directly from the data, reducing the need for manual feature extraction and making them powerful tools for numerous tasks, especially in image and video recognition.

6.5.2 Paradigm

In our simulations, we have the ability to modify a wide array of parameters. From a machine learning perspective, these range from the dataset employed (including its size and class heterogeneity) to the specific model, optimizer, batch sizes, as well as CPU and memory limitations of our clients.

A significant portion of our simulations involve the utilization of a straightforward Convolutional Neural Network (CNN) model. This model is comprised of two convolutional layers and three fully connected layers.

Furthermore, we've incorporated a loss function and an optimization method as essential elements in our training process. Specifically, we used the Cross-Entropy Loss

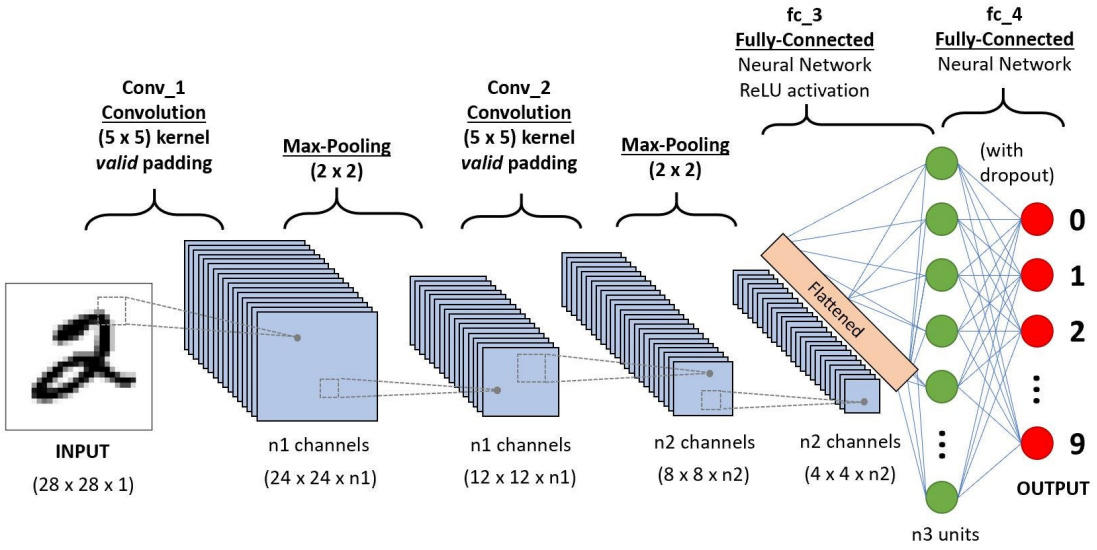


Figure 6.5.1: MNIST CNN model

function (torch.nn.CrossEntropyLoss) and the Stochastic Gradient Descent (SGD) optimizer (torch.optim.SGD), configured with a learning rate of 0.001 and momentum of 0.9. These provide us with a robust foundation for effective network training.

Furthermore, during client-side training, we acquire several crucial metrics. These include training loss, performance on the validation set, and performance on an unseen dataset that mirrors the client's data characteristics. We also conduct testing using the server's global dataset. This dataset may differ significantly in size, distribution, and data heterogeneity compared to the client's data. This diverse array of testing environments helps us ensure the robustness and adaptability of our model across varying data contexts.

Table 6.2: Configuration of the Neural Network Layers

Layer Type	**Configuration**
Convolutional	Input channels: 1, Output channels: 6, Kernel size: 5
Max Pooling	Kernel size: 2, Stride: 2
Convolutional	Input channels: 6, Output channels: 16, Kernel size: 5
Max Pooling	Kernel size: 2, Stride: 2
Fully Connected	Input features: 16*16, Output features: 120
Fully Connected	Input features: 120, Output features: 84
Fully Connected	Input features: 84, Output features: 10

6.5.3 Data heterogeneity

First, we train our model using various clusters of users, resulting in a model that is primarily based on a single handwriting style (or multiple styles with similar characteristics). The validation accuracy in this scenario exceeds 0.9%. However, when the same model is tested on a different handwriting style, the test accuracy significantly drops. Our experiment demonstrates that the model trained on cluster 4 does not perform well on test datasets from other clusters (see Figure 6.5.14), yet it achieves an acceptable level of accuracy when applied to its own handwriting style. This highlights the importance of addressing data heterogeneity and developing models that can generalize across diverse styles in real-world applications.

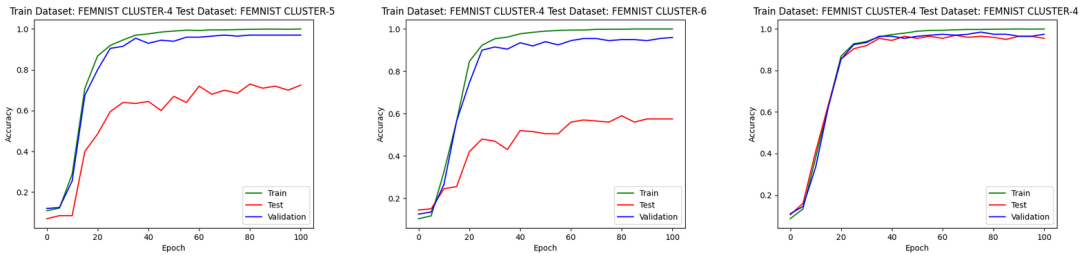


Figure 6.5.2: Accuracies of FEMNIST Group Training

Following the initial training, we employ the same clusters to train our Federated Machine Learning (FML) algorithm of choice using the FEDMA algorithm and the Flower framework, with a setup of 7 clients, 10 rounds, and 10 epochs per round. The results align with our expectations as shown in Figure 6.5.3; clients utilize their own datasets, characterized by unique handwriting styles, to train their local models. The merged model on the server side, however, integrates these diverse local models to form a more comprehensive and generalized global model, enabling accurate inference even on unseen handwriting styles.

Each client possesses a model that can accurately predict outcomes for their own dataset and correctly infer results, even in the presence of outliers within the client's homogeneous dataset. Furthermore, if a new client joins the FML paradigm, it can immediately leverage the global model for accurate inference without the need for additional training. This demonstrates the effectiveness of Federated Learning in addressing data heterogeneity and creating models that can generalize across a wide range of input data.

In this experiment we use the skewed-imbalance dataset of MNIST. In the simulation where we have 4 classes per client, as shown below:

The clients have taken their own partition of the dataset and we can simulate the class heterogeinity with DH=0.4 by assigning data points from only the 6 out of 10 digits /classes from our dataset. As the FML process starts the users train on their data and at each iteration they send their models back to the server. As we can see the results of the accuriacies after training on local clients. The train and validation set are expected to reach high accuracy values and the test on their own data is slowly converging to the

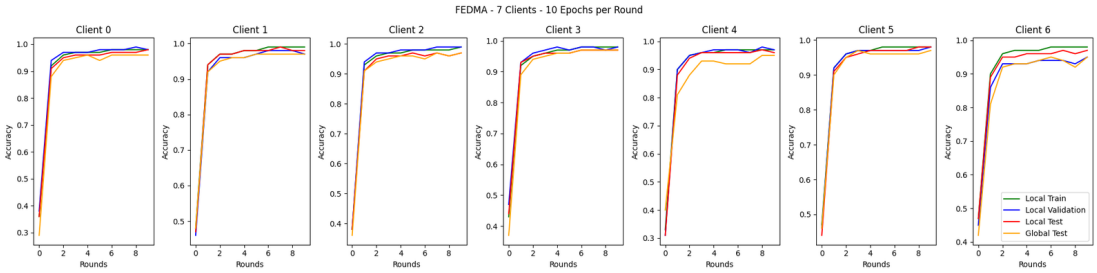


Figure 6.5.3: Results of training on FEDMA with different FEMNIST groups

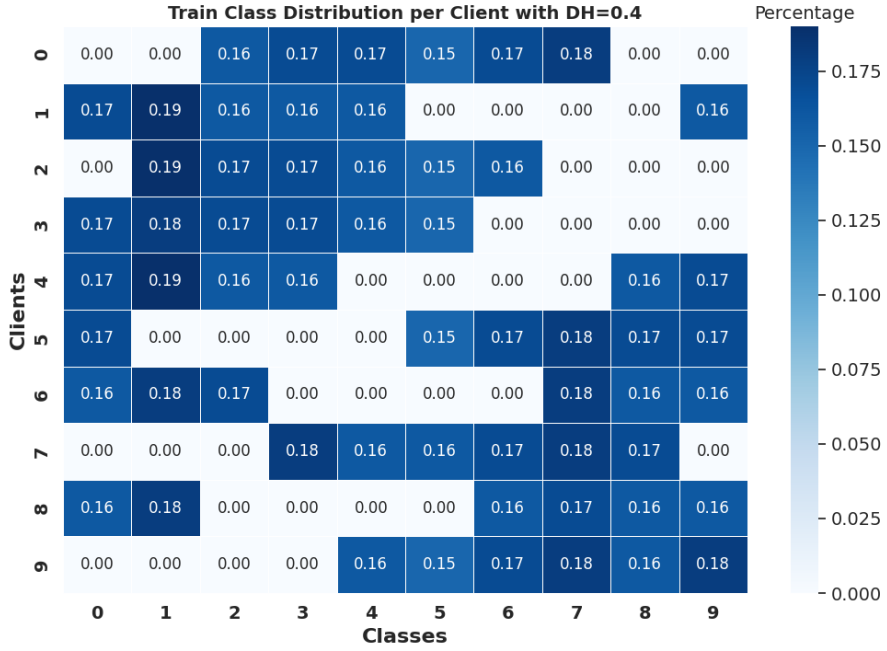


Figure 6.5.4: Data Heterogeneity of Simulation

same level of the train accuracy. The Test on Global Unseen data is expected to reach the 0.6 as each client has the 60

In order to see the real value of federated machine learning behind class heterogeneity is to observe the accuracy of the global model that is created on the server on the test set of the whole dataset.

6.5.4 Data Size Heterogeneity

In practical scenarios, it is observed that edge devices exhibit varying capacities, and their data set sizes differ as well. Given the information provided below, the accuracy of tests with different sizes in each simulation can be analyzed.

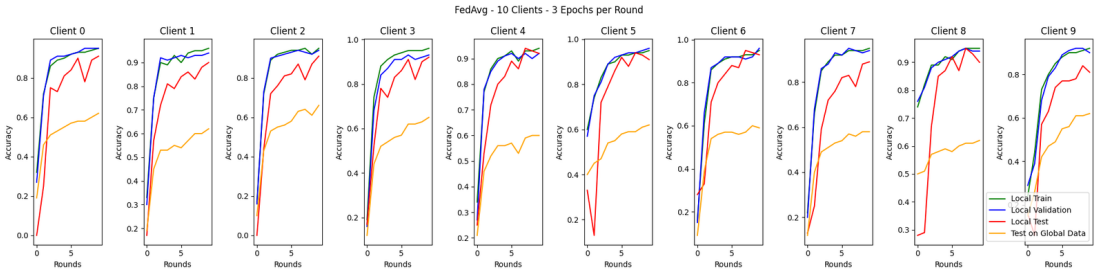


Figure 6.5.5: Clients Accuracies on MNIST

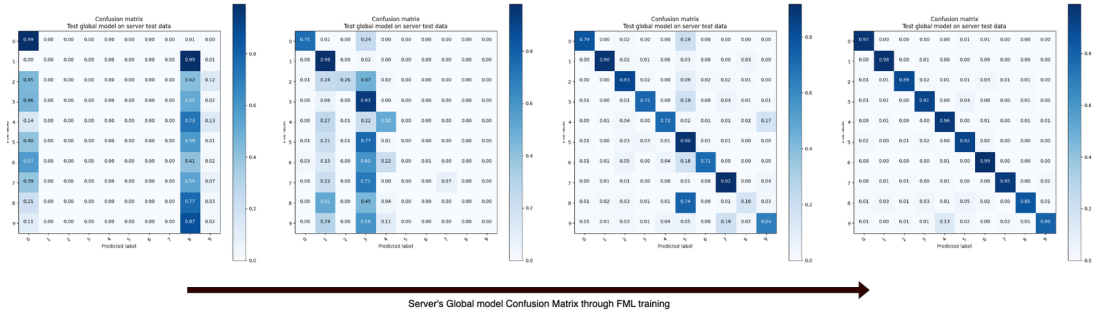


Figure 6.5.6: Progress of confusion matrix on glabal model

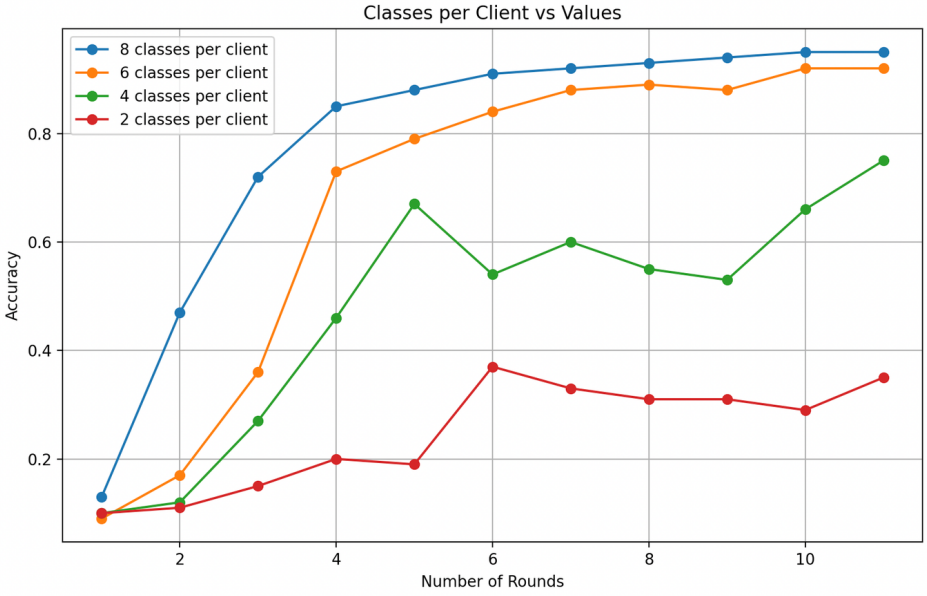


Figure 6.5.7: Accuracy with different data sizes

Three clients are employed, each having the same data size. It is evident that the overall execution time of the entire paradigm increases. However, noteworthy is the fact that the

paradigm achieves convergence to the same accuracy value, specifically when all clients possess a data size of 15,000, as well as when the data size is 10,000. These simulations hold significant relevance in cases where edge devices face resource limitations, thereby allowing the omission of 5,000 data samples.

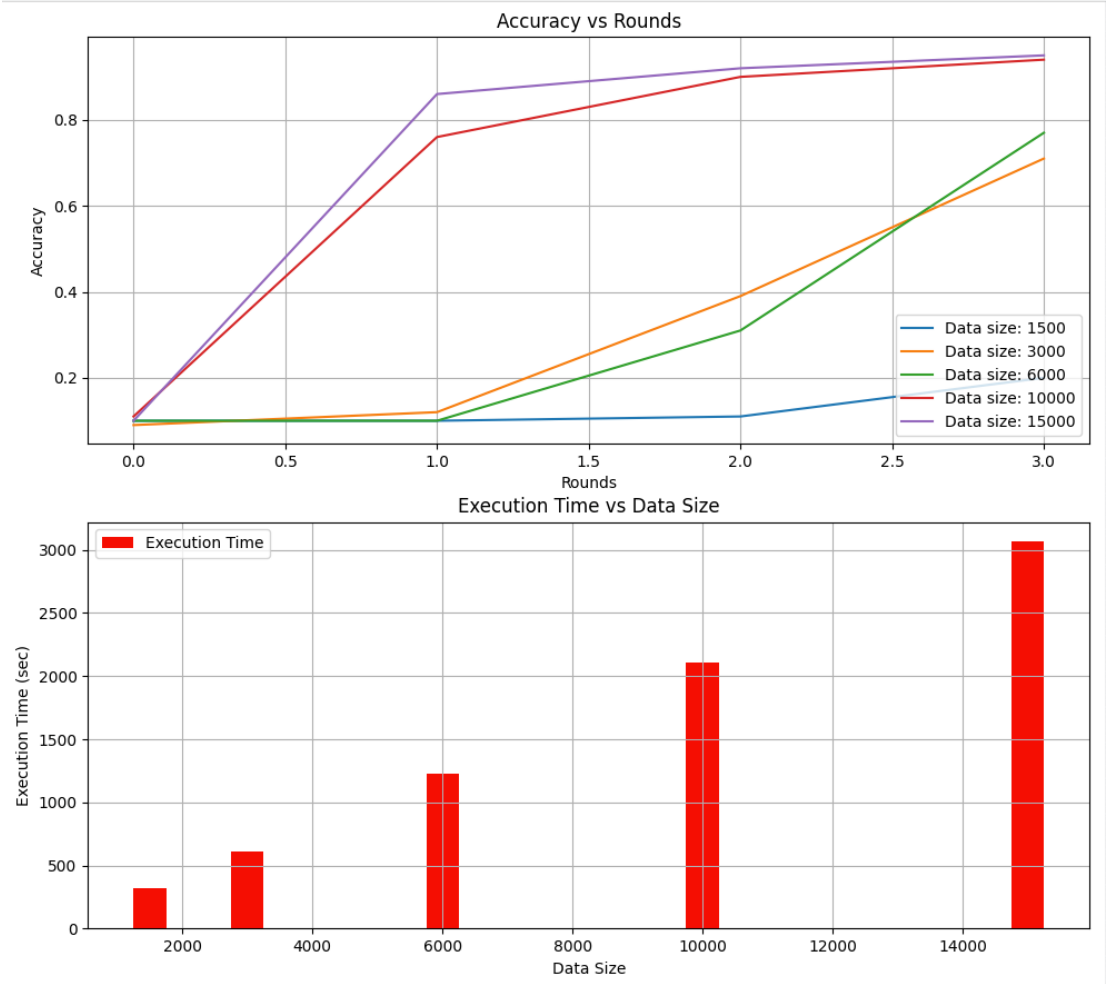


Figure 6.5.8: Accuracy and Execution time on Different Data sizes

6.5.5 Number of Clients

In developing and optimizing a federated learning system, the ability to effectively simulate different configurations is crucial. Simulating the paradigm with the same number of clients as in the real-world deployment enables us to approximate the system's actual behavior, offering valuable insights into its performance, potential bottlenecks, and areas for improvement.

Simulating the system with a smaller number of clients can help to simplify the complexity and make it easier to understand the fundamental behaviors of the system. It

can also be helpful for debugging, as fewer clients can often mean fewer places for things to go wrong. Conversely, simulating with a larger number of clients can allow us to test the system under heavier loads, potentially revealing scalability issues or performance bottlenecks that might not be visible with fewer clients. Additionally, it can better reflect the diversity of the client's local datasets, capturing more of the heterogeneity that is often a key characteristic of federated learning.

So in order to simulate our paradigm with different number of clients we keep the below arguments fixed:

1. Memory: max 250
2. CPU: max 1
3. Server rounds: 3
4. Epochs per round: 1
5. Model: Net2
6. Dataset: Mnist
7. Partitions: 10
8. Strategy: FedAVG

We use 3, 5, and 10 clients and we measure their accuracy, time and CFMQ after the training has completed.

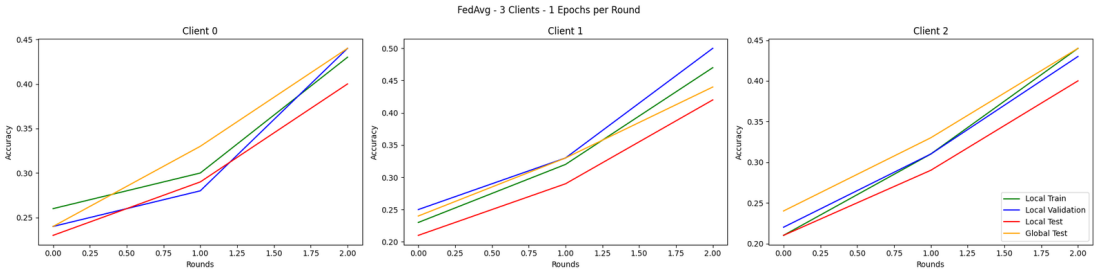


Figure 6.5.9: Accuracy of training on 3 Clients

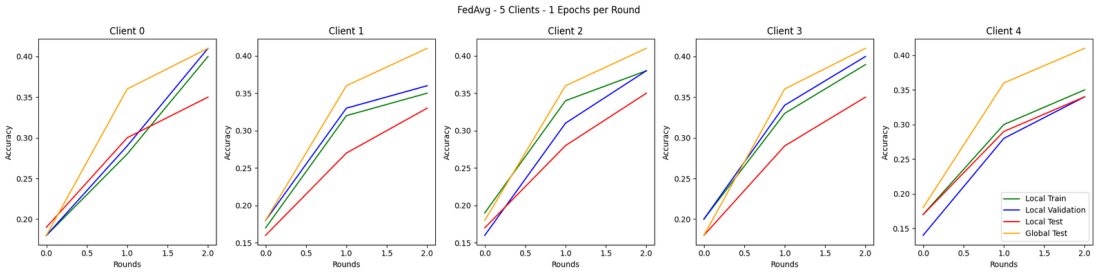


Figure 6.5.10: Accuracy of training on 5 Clients

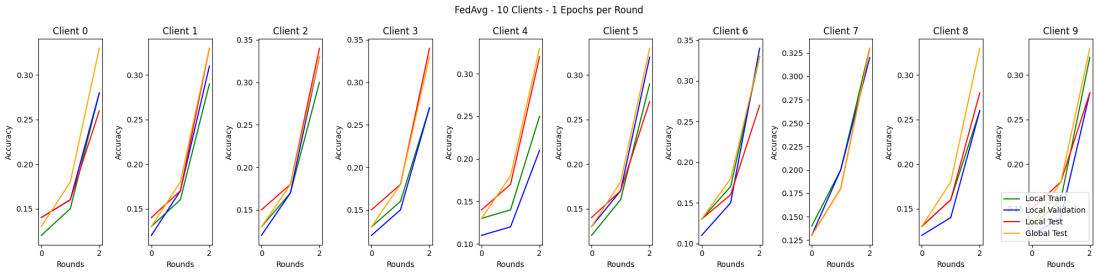


Figure 6.5.11: Accuracy of training on 10 Clients

Table 6.3: Execution Time and CFMQ Values

Number of Clients	**Time (sec)**	**CFMQ**	**Server Accuracy after 3 rounds**
3	900	2,177MB	0.43
5	1,193	3,347MB	0.38
10	1,606	7,296MB	0.36

6.5.6 Impact of CPU heterogeneity

In this study, we explore the effects of CPU heterogeneity within the context of the Federated Machine Learning (FedML) paradigm.

In order to maintain consistency throughout the different simulations and to ensure that the variations in CPU resources are the sole differentiating factors, several key parameters are kept constant across all simulations:

1. Memory: Each simulation is confined to a maximum memory limit of 250 units, thus ensuring that any performance variation cannot be attributed to differential memory utilization.

2. Server Rounds: We restrict the number of server rounds to three for all simulations. This ensures that any observed changes are not due to differences in the number of server rounds.

3. Epochs Per Round: Each server round is structured to include three epochs, a feature that is invariant across all simulations.

4. Model: The neural network model utilized in all simulations is 'Net2'. This standardization prevents any variations in the neural network architecture from influencing the outcomes.

5. Dataset: All simulations make use of the 'Mnist' dataset, eliminating any discrepancies that might arise from using different datasets.

6. Partitions: We uniformly divide the dataset into ten partitions for each simulation. This ensures that the data distribution remains constant across all simulations.

7. Strategy: We adopt the FedAVG strategy in all simulations.

By standardizing these elements across all simulations, we can isolate the impact of CPU heterogeneity on the performance and efficiency of the Federated Machine Learning paradigm.
Initially, we investigate a scenario where each client within the simulation shares an identical CPU configuration, with variations in CPU power only present between different simulations. This means, at the onset, we allocate six out of the available eight CPU cores on our machine to three clients, effectively allocating two CPUs per client. Over time, we progressively reduce the CPUs assigned per client until each receives 1/8th of a CPU, a procedure we refer to as CPU division.
As per our expectations, we observe that with every successive reduction in CPU power per client, the time required to train, test, and conclude the entire FedML paradigm approximately doubles. Despite these alterations in CPU allocation, the Communication-Computation Trade-Off measure, denoted here as CFMQ, remains relatively stable, hovering around a value of $22 * 10^8$ as all the parameters are the same in the FML tests except the Peak Memory usage that adds at the CFMQ metric when more CPUs are used.
Furthermore, we note that the Test Accuracy metric on the server at the conclusion of the simulations is approximately 0.9. This result is achieved after three server rounds, each comprising three epochs per round. These findings demonstrate the resilience of the FedML system in the face of varying computational resources and offer insights into optimizing its performance under different CPU configurations.

Table 6.4: CPU heterogeneity configurations and results

Test	Clients	CPUs	CPUs/Client	Time	CFMQ	Server Test Accuracy
1	3	6	2	623.28sec	2530MB	0.92
2	3	3	1	1240.51sec	2300MB	0.93
3	3	3/2	1/2	2315.45sec	2240MB	0.92
4	3	3/4	1/4	5487.26sec	2210MB	0.91
5	3	3/8	1/8	12065.69sec	2010MB	0.92

6.5.7 Stragglers

To simulate the phenomenon of stragglers - nodes that lag behind in performance due to limited resources - we have designed a nuanced experimental structure. The objective is to explore the impact of non-uniform resource distribution among the federated learning nodes.

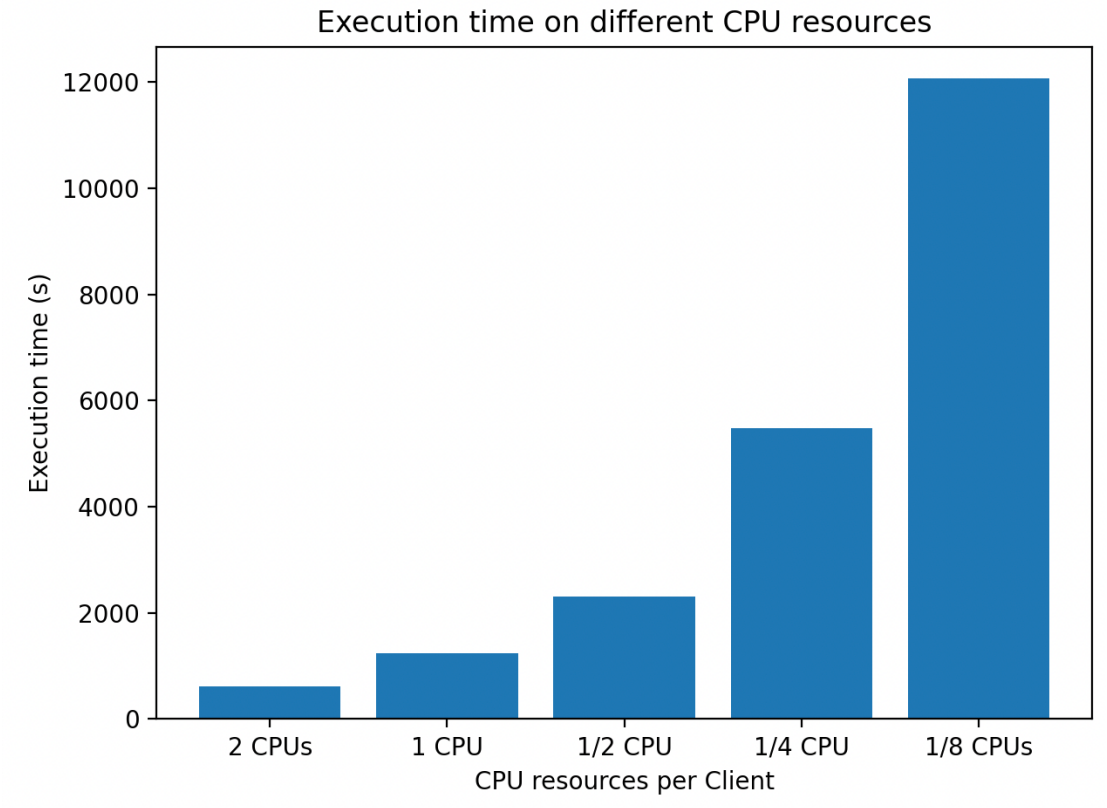


Figure 6.5.12: Execution time on different CPU resources

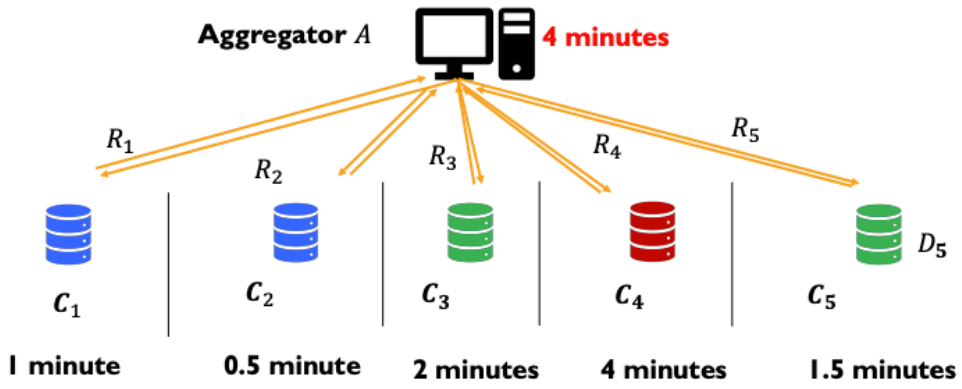


Figure 6.5.13: Visualisation of Stragglers

This is achieved by constructing multiple groups of edge devices, each characterized by uniform CPU allocation within the group. However, the CPU allocation varies between different groups, thus creating a systematic disparity in computational capacity among

them. This setup emulates a real-world scenario where certain nodes, or stragglers, do not function as swiftly as others due to resource constraints.

By carefully orchestrating this resource heterogeneity among the federated learning nodes, we aspire to gain deeper insights into the dynamics of the system under the influence of stragglers. This understanding will enable us to develop more robust and efficient strategies for federated learning in resource-limited settings.

	Test 1	**Test 2**	**Test 3**
Total Clients	4	4	8
Clients - Group 1	2	2	4
Clients - Group 2	2	2	4
CPUs/Client - Group 1	2	3	3
CPUs/Client - Group 2	1	1/2	1/2
Time(s)/Round AVG-Group 1	150, 79, 55	79, 47, 32	238, 237, 235
Time(s)/Round AVG-Group 2	238, 124, 85	314, 190, 125	566, 562, 568
Exec Time Total	1019s	1569s	2443s
CFMQ	3027MB	6102MB	13012MB
Final Accuracy	0.92	0.92	0.92

Table 6.5

6.5.8 Dropouts

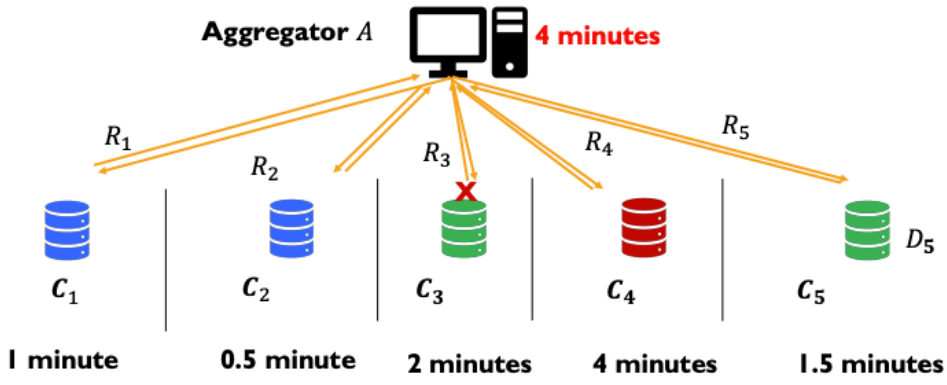


Figure 6.5.14: Visualisation of Client Dropouts

To further simulate the real-world complexities of a federated learning environment, we introduce the concept of 'dropouts' in our experiment. 'Dropouts' refer to the nodes that intermittently disconnect from the network, often without any prior indication, thereby introducing unexpected discontinuities in the learning process.

In cross-device federated learning, where a large number of client devices collaborate to train a machine learning model while keeping their data locally, there can be numerous reasons for client dropouts. These could include:

1. Network Connectivity Issues: Client devices may drop out due to intermittent or unreliable network connections. If a device cannot maintain a stable connection to the server, it may not be able to participate effectively in the training process.
2. Device Availability: Devices may not be continuously available. Users may turn off their devices or disconnect from the network. For example, mobile phones may be switched off, put on airplane mode, or simply run out of battery.
3. Resource Limitations: Devices with low battery power, limited computational resources, or insufficient memory might be unable to participate in the training process and may drop out.
4. User Preferences: Users can often choose whether or not to participate in federated learning processes. If a user decides to stop participation (due to privacy concerns, for instance), their device becomes a dropout.
5. System Updates or Crashes: If a device needs to update its operating system, or if it crashes for some reason, it will also drop out of the federated learning process.
6. High Communication Latency: In some cases, a client might have a slow communication link. If the server waits for updates from all clients, then a slow client can significantly delay the entire training process. To prevent this, the server might set a deadline and proceed with the next round without waiting for slow clients. Thus, clients with high latency might frequently drop out.
7. Data Changes: If the local data on a client device changes significantly (due to new user actions, for instance), the client might be unable to provide a useful model update and may drop out of the learning process.
8. Load Balancing and Client Selection: Sometimes, the server selects only a fraction of available clients for each round to reduce communication overhead or balance the load. So, some clients might appear as "dropouts" in some rounds.

The issue of client dropouts is a widely recognized challenge in Federated Machine Learning (FML), and numerous studies have been conducted aiming to address this problem. Various solutions have been proposed, demonstrating the depth of ongoing research in this area [47], [12] [58] [15].

We establish multiple groups of edge devices, wherein each group experiences a unique dropout rate. This is achieved by artificially inducing disconnections in the nodes at different intervals, thereby creating a diversified range of dropout patterns across groups. This allows us to evaluate the resilience of our federated learning system to sudden and unexpected interruptions.

The design of our experimental framework mirrors the unpredictable network instabilities often encountered in practical deployments. Through this, we aim to understand the

effects of sporadic node disconnections on the performance and robustness of our federated learning system. This understanding will empower us to devise strategies that are capable of handling and adapting to unexpected network interruptions, thereby ensuring sustained learning performance in the face of uncertainties.

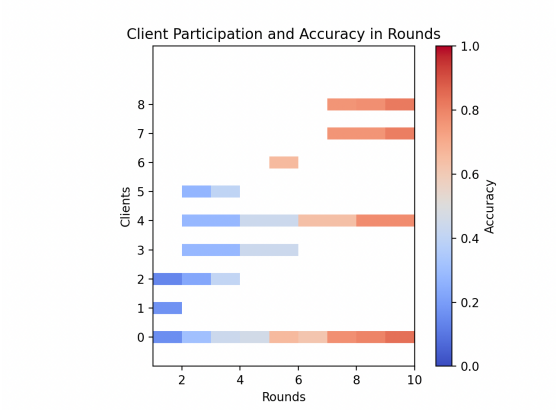
To accomplish this objective, we posit an environment where all nodes are endowed with identical configurations and resources. Our primary focus is to analyze how the system dynamically adapts to fluctuations in the number of active clients. Leveraging the real-time scalability of our APIs, we have the ability to modulate the number of clients in operation, thus facilitating a comprehensive examination of the system's response to varying levels of client engagement.
IID Data on Dropouts Scenario:
In scenarios such as Test 1, we begin with a Federated Machine Learning (FML) paradigm that uses a homogeneous configuration of clients. The paradigm commences with three clients, and throughout its execution, the number of participating clients fluctuates. Despite these variations and multiple client dropouts, the data remains homogeneous and independently and identically distributed (IID), ensuring the accuracy of the global model continues to improve. Each new client, working with a distinct partition of the dataset, inherits the most recent version of the model, contributing to its ongoing refinement through further training. Consequently, the resilience of the model is demonstrated as it successfully accommodates changes in client participation while persistently enhancing accuracy.

In this simulation, we notice a significant trend wherein the clients that initiate midway through the entire learning paradigm display an intriguing behavior. These clients are downloading the parameters of an already trained model and initiating training on their own datasets. Given the Independent and Identically Distributed (IID) nature of the data, it is observed that these clients start off with a high accuracy right from the onset. This implies that even late-joining clients can benefit significantly from the pre-existing learning of the model, resulting in an accelerated learning process and high initial accuracies. This behavior underscores the effectiveness of federated learning in dynamically distributed environments.

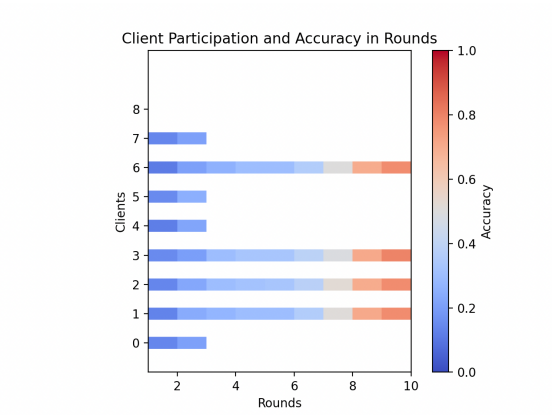
In the realm of cross-device federated machine learning, the issue of client dropouts emerges as a significant challenge. In practical scenarios, various factors can lead to device disconnections. For instance, devices may disconnect due to unstable network conditions, power outages, or the server may deliberately reduce the number of devices sampled to optimize resource utilization. Our observations suggest that early client dropouts result in quicker completion of training rounds. This is because fewer clients participating in fewer rounds translates into reduced potential for latency, thereby accelerating the overall learning process. However, while this can enhance system efficiency, it's important to consider the potential impact on learning quality and model robustness, given the reduced data diversity and volume.

Conversely, when client dropouts occur towards the end of the overall training process, each client participates in more rounds of model training. This extended participation contributes to a more comprehensive training of the global model. The inclusion of more

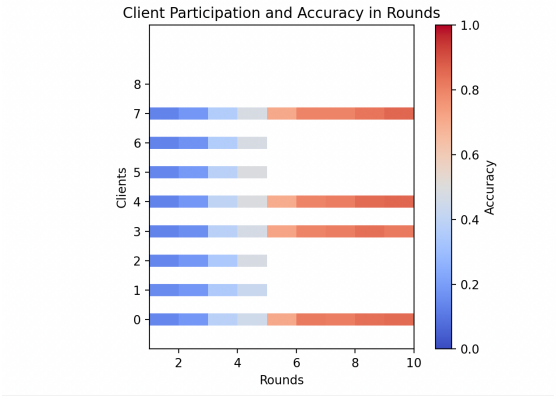
client data across multiple rounds enriches the model by adding diversity and volume to the training data. This results in a model that is better generalized and more robust, highlighting the value of sustained client participation in the federated learning process. However, it's crucial to balance this with the potential increase in latency due to the larger number of rounds and the associated computational and communication overhead.



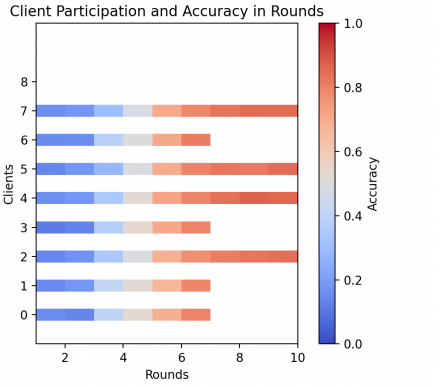
(a) Accuracy on random client logins and logouts



(b) Accuracy on Clients Dropout at Round 3



(c) Accuracy on Clients Dropout at Round 5



(d) Accuracy on Clients Dropout at Round 7

Non-IID Data on Dropouts Scenario:
In the current setup, the dataset is partitioned into ten separate segments, with the data uniformly and randomly distributed among these partitions. Consequently, even if certain clients disconnect midway through the training process without contributing to the overall training, the global model should theoretically continue to improve. This is because new clients joining the network receive a well-refined model, while the clients that have disconnected retain their model up to the point of disconnection. However, the situation might drastically change when we transition to a non-IID (Independent and Identically Distributed) data scenario. For instance, in the extreme case where each client obtains different classes of the MNIST dataset or a combination thereof, the

dynamics of the federated learning process may become more complex. The model's performance could potentially be impacted by the intermittent participation of clients, and the evolution of the global model might not follow the same progressive improvement trajectory as observed in the IID scenario. Therefore, handling these situations will require a careful consideration of the distribution of data among the clients and a well-designed strategy for managing client participation and dropout.

Chapter 7

Use cases

7.1 Mobile applications

In the context of mobile phones, federated machine learning has a wide range of applications.

One of the most early applied use cases is the language models on keyboards where the fml is used in a personalized manner to improve the writing experience of the user without the used words sent to a centralized server for inference. In et. al. [28] the authors train an RNN language model using on-device learning using FedAVG. More in detail the Google Keyboard has 600 language varieties and 1 billion installs as of 2019 and offers autocorrection, word completion and next-word prediction. Also they add that the users, the mobile phones used are running in different hardware capabilities and the communications are not always the best, and the batteries are a problem as the inference takes a lot of their capacity. ALso they use hyperparameter optimization in terms of batch size, number of client epochs and the number of clients per round supporting our claims that these kind of simulations are vital for the usage of fml in real use case scenarios. At the end the show that a language model trained from scratch using federated learning can outperform an identical server trained CIFG model and baseline n-gram model on the keyboard next-word prediction task.

Furthermore, this technology can be used in face detection use cases where the users want to connect with their face for faster login but they hesitate to give their face representations in big companies. the privacy concerns in the face dataset and also the legal restrictions are growing. As the authors suggest at [4] the Automated Face Recognition systems are used widely and have a lot of uses, but to be used they need a big database with different and wide characteristics like race and age. And because is very difficult to buy that kind of data the big companies use the public images from the well known social media websites to train their models without user consent. In order to compact this there are regulation that target this “web scraping“ of faces. In this paper they propose FedFace, that learns an accurate face recognition model from multiple mobile devices in a collaborative manner without sending training face images

outside of the device, using a pretrained face recognition system, CosFace and trying it more only in the case of one face per device.

In the same length voice recognition is a valid use case where the models can always get better to identify the voice of the users and make the use better and the experience whole while keeping their data private. More in depth the speaker verification use case is vital for the usage of devices that need a "wake-up phrase". For this purpose the authors at el. [27] support the idea that a side federated model can contribute to upgrade the already created models while preserving the user privacy. In this paper they use federated machine learning in conjaction with different differential privacy methods like Local and Central DP to ensure the privacy in the inference and they conclude that the fml way in combination with the central training have 6% better results. Also at el. [25] they study on training Automatic Speech Recognition (ASR) models in Federated Learning (FL) settings while trying to make the experiments as close to real world systems while using cross-silo and cross-device scenarios with only 10 and then 2.000 and 4.000 clients.

7.2 Autonomous Vehicles

In this context FML can be used to provide a safer and better self-driving car experience with real-time data and predictions. In the most common machine learning scenarios in the autonomous vehicles, the data that are captures by the various sensors like RADAR and LIDAR and cameras are transmitted for a centralized training, this add latency and transmission overhead and privacy concerns. So there are some efforts to research the implementation of fml in these use case in order to make the autonomous vehicles more robust [23], [18]. In the paper [43] the authors name the integration of the FML paradigm in vehicular networks as Federated Vehicular Network (FVN) which in conjunction with distributed learning and blockchain conclude, as presented, the future of autonomous vehicles.

7.3 Medical Data

Data-driven machine learning has emerged as a promising approach for extracting valuable insights within the healthcare sector, where data is generated and stored in vast quantities. However, due to various regulatory constraints, the full potential of this data remains largely untapped, and public datasets available for use are often limited and inadequate for real-world applications. Furthermore, there are instances where access to medical datasets requires substantial financial investments. These challenges have led to the increased interest in privacy-preserving paradigms, with particular emphasis on federated learning [34]. Although several obstacles remain to be addressed in the implementation of federated machine learning, it presents a viable pathway to unlocking the hidden value of underutilized medical data and ultimately enhancing health outcomes.

Machine learning has found extensive applications in various healthcare domains, such as radiology, pathology, and genomics. For these applications, deep learning models often necessitate a large number of parameters to achieve the accuracy required for

real-world scenarios. One major challenge associated with medical datasets is the generalization of models to previously unseen data. Medical data is often subject to high levels of bias and heterogeneity, which arise due to geopolitical and social factors, as well as hardware-related differences, such as the quality of imaging equipment. Data-driven methodologies necessitate datasets that accurately represent the data distribution relevant to the problem at hand. Comprehensive and inclusive databases are crucial for ensuring generalizability; however, advanced algorithms are often evaluated using carefully curated datasets that originate from a limited number of sources, or even a single source. This presents substantial challenges, as isolated data repositories can introduce sample bias, resulting in demographic imbalances (e.g., gender, age) or technical disparities (e.g., acquisition protocol, equipment manufacturer) that can skew predictions and compromise the accuracy of predictions for specific groups or locations. Regulatory restrictions often limit the use of medical data for research purposes, with patient consent and ethical approval being prerequisites for data utilization [5]. Although data anonymization has been proposed as a means to circumvent these constraints, it has become increasingly apparent that the removal of metadata, such as patient names or dates of birth, is often insufficient to ensure privacy [6]. For instance, it has been demonstrated that a patient's face can be reconstructed from three-dimensional imaging data, such as computed tomography (CT) or magnetic resonance imaging (MRI), and that the human brain itself is as unique as a fingerprint [7]. Consequently, subject identity, age, and gender can be predicted and disclosed [8]. The advantage of federated learning (FL) lies in its ability to protect sensitive training data by obviating the need for data transfer beyond the firewalls of the institutions where it is stored. This approach enables the development of robust models while maintaining patient privacy and adhering to regulatory requirements.

In support of our claims regarding the potential benefits of federated learning and privacy-preserving paradigms in healthcare, we will present several cases of initiatives aimed at combining health data from various sources. These initiatives not only demonstrate the growing interest in utilizing medical data to enhance healthcare outcomes but also highlight the challenges associated with data privacy and regulatory compliance:

1. IBM's acquisition of Merge Healthcare for 1 billion dollars: IBM has completed the acquisition of Merge Healthcare, a provider of medical image handling and processing technology used in over 7,500 U.S. healthcare sites, as well as clinical research institutes and pharmaceutical firms. The Watson Health Cloud is expected to be utilized by these organizations to gain insights from consolidated medical images, electronic health records, and other medical data. This acquisition will enhance IBM's Watson Health business unit and improve its ability to cross-reference medical images against 315 billion data points in the Watson Health Cloud, providing insights that can help clinicians diagnose, treat, and monitor patients more effectively. By comparing new medical images with a patient's medical history and other similar patients, Watson can help healthcare providers pursue more personalized approaches to healthcare.

2. Charter for Safe Havens in Scotland: The Charter for Safe Havens in Scotland is

a set of guidelines created to regulate the use of unconsented data from National Health Service (NHS) patient records for research and statistical purposes. The charter outlines a framework for the establishment of "safe havens" that can handle such data securely and safely, ensuring that the privacy of patients is protected. By providing a secure environment for handling unconsented patient data, the initiative seeks to support the development of new treatments and therapies, as well as advance medical knowledge and understanding.

3. The French Health Data Hub and the German Medical Informatics Initiatives: These two national projects aim to promote data sharing in healthcare while addressing the challenges related to data privacy and security. Both initiatives focus on creating a centralized platform for managing and sharing health data, enabling researchers and clinicians to access valuable information that can contribute to the improvement of healthcare services, the development of new treatments, and the advancement of medical knowledge.

In addition to these examples, further research and initiatives in federated learning and privacy-preserving paradigms will be explored to underscore the growing importance of these approaches in the healthcare sector. By examining the successes and challenges of these cases, we aim to demonstrate the potential of federated learning in unlocking the value of underutilized medical data while maintaining patient privacy and adhering to regulatory requirements.

Federated machine learning (FL) holds significant potential to benefit a wide range of stakeholders within the healthcare industry. For clinicians, augmenting their expertise with expert knowledge from other institutions can ensure a level of diagnostic consistency not attainable today. Patients in remote areas or those with rare diseases can also benefit from the same high-quality, ML-aided diagnosis available in larger hospitals. By maintaining full control and possession of their patient data while contributing to a potentially vast collection of real-world data, hospitals and practices can facilitate research and development of novel algorithms. FL can enhance the accuracy and robustness of healthcare AI, leading to improved patient outcomes and cost reductions, which are essential for the advancement of precision medicine. Manufacturers of healthcare software and hardware can leverage FL to foster the continuous improvement of ML-based systems, potentially unlocking new sources of data and revenue. However, stakeholders must invest in on-premise computing infrastructure or private-cloud service provision and standardize data formats to ensure the seamless training and evaluation of predictive models. Additionally, stakeholders must conduct research on algorithmic strategies for federated training and navigate the complex regulatory frameworks surrounding continual and FL approaches. Despite the inherent challenges, FL has the potential to profoundly impact the healthcare industry by promoting data sharing and collaboration, improving patient outcomes, and advancing medical knowledge and understanding. By addressing the obstacles and capitalizing on the opportunities presented by FL, the healthcare sector can harness the power of underutilized medical data while upholding patient privacy and adhering to regulatory requirements.

Federate Machine learning use cases in healthcare:

1. Identifying clinically similar patients through electronic health records (EHR).
2. Predicting hospitalizations due to cardiac events.
3. Estimating mortality and intensive care unit (ICU) stay durations.
4. Conducting whole-brain segmentation in MRI scans.
5. Performing brain tumor segmentation.
6. Enhancing mammogram assessments.
7. Predicting treatment responses for breast cancer and melanoma patients.
8. Improving tumor boundary detection in brain gliomas, breast tumors, liver tumors, and bone lesions from multiple myeloma patients.
9. Optimizing drug discovery processes.

Patients: FL can ensure high-quality clinical decisions globally, benefiting patients in remote areas and those with rare diseases. Data donors may feel more comfortable knowing that their data remains with their own institution.

Hospitals: By retaining control and possession of patient data, hospitals and practices limit the risk of misuse. However, they must invest in computing infrastructure, private-cloud services, and standardized data formats for seamless ML model training and evaluation.

Researchers and AI developers: Access to a vast collection of real-world data benefits smaller research labs and start-ups. However, they must develop algorithmic strategies for federated training and work without direct access to all training data.

Healthcare providers: FL can support the shift to value-based healthcare and precision medicine by increasing accuracy and robustness of healthcare AI, reducing costs, and improving patient outcomes.

Manufacturers: Healthcare software and hardware manufacturers can benefit from FL by continuously validating and improving their ML-based systems, although this may require significant upgrades to local compute, data storage, networking capabilities, and software.

Clinicians: Clinicians can augment their expertise with expert knowledge from other institutions, ensuring diagnostic consistency. While federated training may yield less biased decisions, it requires compliance with agreements on data structure, annotation, and report protocols.

Chapter 8

Conclusion

8.1 Future Upgrades

In order to further enhance the capabilities and flexibility of our Federated Machine Learning (FML) simulation tool, we propose several future upgrades for consideration:

Improved User Interface (UI): Enhancing the visual appeal and user-friendliness of the tool's UI is a key priority. By adopting modern UI frameworks or libraries, we can create a more responsive design that adapts seamlessly to different screen sizes and devices. A visually appealing and intuitive interface will enhance the user experience and facilitate easier interaction with the tool.

Enhanced Details, Analytics, and Monitoring: To provide researchers with comprehensive insights into the performance of the FML paradigm, we intend to augment the tool with additional features for detailed analytics and monitoring. This includes integrating comprehensive statistics, visualizations, and metrics such as accuracy, loss, convergence speed, communication costs, and other relevant measurements. These enhancements will empower researchers to gain deeper insights into the behavior and effectiveness of their models.

Expansion of Model and Strategy Options: A crucial aspect of FML is the availability of diverse machine learning models and strategies. In our future upgrades, we aim to incorporate a wider range of model options, such as neural networks, decision trees, support vector machines (SVMs), and more. Additionally, we plan to expand the available strategies for federated learning, including federated averaging, hierarchical aggregation, and other advanced techniques. This will enable researchers to select the most appropriate models and strategies for their specific requirements.

Integration of Differential Privacy: Privacy preservation is a critical concern in FML. Therefore, we intend to integrate differential privacy techniques into our tool. By implementing mechanisms such as noise injection, privacy-preserving aggregations, and privacy budget management, we can ensure that participants' sensitive data remains

protected during the training process. This addition will strengthen the privacy guarantees of our FML simulation tool.

Real Edge Device Deployment: The deployment of FML on real edge devices is an important aspect for practical evaluation and experimentation. As a future upgrade, we plan to enable the deployment of the FML paradigm on physical edge devices. This will involve developing functionalities to connect and communicate with real devices, allowing researchers to conduct experiments and evaluate the performance of their models in realistic edge computing scenarios.

Simulations with Added Parameters: WiFi, Battery, In Use: To create more realistic FML simulations, we aim to introduce additional parameters that mimic real-world edge device characteristics. These parameters include variables such as WiFi signal strength, battery levels, and device usage patterns. By incorporating these factors into our simulations, we can analyze the impact of these realistic conditions on the performance of the FML paradigm.

Data Augmentation: Data augmentation is a widely used technique to improve model generalization and robustness. As a future upgrade, we plan to implement data augmentation capabilities within our tool. This will allow researchers to perform various transformations such as rotation, scaling, cropping, and noise addition to augment the diversity and quantity of training data available to each client. The inclusion of data augmentation techniques will enable researchers to explore the impact of augmented data on the FML paradigm.

Model Save Before End: We recognize the importance of model persistence and reusability for researchers. As part of our future upgrades, we aim to incorporate an automatic model-saving feature at predefined checkpoints or before the completion of training. This functionality will ensure that trained models are saved and easily accessible for further analysis, fine-tuning, or deployment purposes.

Customizable Input of Code from Researchers: To cater to the diverse needs and preferences of researchers, we plan to provide a customizable input feature. This feature will allow researchers to input their own custom code for model architectures, preprocessing techniques, or any other specific requirements. By facilitating customization, our FML simulation tool can adapt to different research goals and support the exploration of novel approaches.

By implementing these future upgrades, our FML simulation tool will offer researchers an advanced platform for conducting experiments, analyzing results, and exploring the potential of FML in various scenarios. These enhancements will empower researchers to push the boundaries of federated machine learning and facilitate progress in this rapidly evolving field.

8.2 Epilogue

In conclusion, this study offers a comprehensive exploration of Federated Machine Learning (FML), elucidating its potential to address critical data-related challenges such as heterogeneity, privacy, and ownership. Through the effective utilization of the open-source framework, Flower, we have demonstrated the practicality and adaptability of FML in diverse research scenarios. Our simulations encompassing varying numbers of clients, rounds, and epochs, coupled with the implementation of distinct techniques such as FedAVG and FEDMA, have shown the versatility of FML. Additionally, the examination of data and hardware heterogeneity, as well as the evaluation of client dropouts and strugglers, have provided valuable insights into the complexities inherent to FML. The findings underscore the value and potential of FML as an advanced tool for machine learning research and applications, affirming its capacity to handle sensitive or large-scale data while preserving privacy and ownership rights. Our work, therefore, provides a robust foundation for future studies and developments in the burgeoning field of federated machine learning.

Chapter 9

Bibliography

[1] Abadi, M. et al. "Deep learning with differential privacy". In: *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 2016, pp. 308–318.

[2] AbdulRahman, S. et al. "A survey on federated learning: The journey from centralized to distributed on-site learning and beyond". In: *IEEE Internet of Things Journal* 8.7 (2020), pp. 5476–5497.

[3] Acar, A. et al. "A survey on homomorphic encryption schemes: Theory and implementation". In: *ACM Computing Surveys (Csur)* 51.4 (2018), pp. 1–35.

[4] Aggarwal, D., Zhou, J., and Jain, A. K. "Fedface: Collaborative learning of face recognition model". In: *2021 IEEE International Joint Conference on Biometrics (IJCB)*. IEEE. 2021, pp. 1–8.

[5] Alabdullah, B., Beloff, N., and White, M. "Rise of Big Data–Issues and Challenges". In: *2018 21st Saudi Computer Society National Computer Conference (NCC)*. IEEE. 2018, pp. 1–6.

[6] Barocas, S. and Selbst, A. D. "Big data's disparate impact". In: *California law review* (2016), pp. 671–732.

[7] Berendt, B. "AI for the Common Good?! Pitfalls, challenges, and ethics pentesting". In: *Paladyn, Journal of Behavioral Robotics* 10.1 (2019), pp. 44–65.

[8] Beutel, D. J. et al. "Flower: A friendly federated learning research framework". In: *arXiv preprint arXiv:2007.14390* (2020).

[9] Beutel, D. J. et al. "Flower: A friendly federated learning framework". In: (2022).

[10] Bhadani, A. K. and Jothimani, D. "Big data: challenges, opportunities, and realities". In: *Effective big data management and opportunities for implementation* (2016), pp. 1–24.

[11] Bharati, S. et al. "Federated learning: Applications, challenges and future directions". In: *International Journal of Hybrid Intelligent Systems* 18.1-2 (2022), pp. 19–35.

[12] Bonawitz, K. et al. "Towards federated learning at scale: System design". In: *Proceedings of machine learning and systems* 1 (2019), pp. 374–388.

[13] Byrd, D. and Polychroniadou, A. "Differentially private secure multi-party computation for federated learning in financial applications". In: *Proceedings of the First ACM International Conference on AI in Finance*. 2020, pp. 1–9.

[14] Caldas, S. et al. "Leaf: A benchmark for federated settings". In: *arXiv preprint arXiv:1812.01097* (2018).

[15] Chai, Z. et al. "Towards taming the resource and data heterogeneity in federated learning". In: *2019 USENIX conference on operational machine learning (OpML 19)*. 2019, pp. 19–21.

[16] Chamikara, M. A. P. et al. "Privacy preserving distributed machine learning with federated learning". In: *Computer Communications* 171 (2021), pp. 112–125.

[17] Chouldechova, A. "Fair prediction with disparate impact: A study of bias in recidivism prediction instruments". In: *Big data* 5.2 (2017), pp. 153–163.

[18] Du, Z. et al. "Federated learning for vehicular internet of things: Recent advances and open issues". In: *IEEE Open Journal of the Computer Society* 1 (2020), pp. 45–61.

[19] Dwork, C. "Differential privacy". In: *International colloquium on automata, languages, and programming*. Springer. 2006, pp. 1–12.

[20] Dwork, C., Roth, A., et al. "The algorithmic foundations of differential privacy". In: *Foundations and Trends® in Theoretical Computer Science* 9.3–4 (2014), pp. 211–407.

[21] Ebers, M. et al. "The european commission's proposal for an artificial intelligence act—a critical assessment by members of the robotics and ai law society (rails)". In: *J* 4.4 (2021), pp. 589–603.

[22] Ekmefjord, M. et al. "Scalable federated machine learning with fedn". In: *2022 22nd IEEE International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*. IEEE. 2022, pp. 555–564.

[23] Elbir, A. M. et al. "Federated learning in vehicular networks". In: *2022 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*. IEEE. 2022, pp. 72–77.

[24] Fan, J. and Vercauteren, F. "Somewhat practical fully homomorphic encryption". In: *Cryptology ePrint Archive* (2012).

[25] Gao, Y. et al. "End-to-end speech recognition from federated acoustic models". In: *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2022, pp. 7227–7231.

[26] Gentry, C. *A fully homomorphic encryption scheme*. Stanford university, 2009.

[27] Granqvist, F. et al. "Improving on-device speaker verification using federated learning with privacy". In: *arXiv preprint arXiv:2008.02651* (2020).

[28] Hard, A. et al. "Federated learning for mobile keyboard prediction". In: *arXiv preprint arXiv:1811.03604* (2018).

[29] He, C. et al. "Fedml: A research library and benchmark for federated machine learning". In: *arXiv preprint arXiv:2007.13518* (2020).

[30] Kanagavelu, R. et al. "Two-phase multi-party computation enabled privacy-preserving federated learning". In: *2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID)*. IEEE. 2020, pp. 410–419.

[31] Karimireddy, S. P. et al. “Scaffold: Stochastic controlled averaging for federated learning”. In: *International Conference on Machine Learning*. PMLR. 2020, pp. 5132–5143.

[32] Kulakli, A. and Osmanaj, V. “Global research on big data in relation with artificial intelligence (A bibliometric study: 2008-2019)”. In: (2020).

[33] Laney, D. et al. “3D data management: Controlling data volume, velocity and variety”. In: *META group research note* 6.70 (2001), p. 1.

[34] Langlotz, C. P. et al. “A roadmap for foundational research on artificial intelligence in medical imaging: from the 2018 NIH/RSNA/ACR/The Academy Workshop”. In: *Radiology* 291.3 (2019), pp. 781–791.

[35] LeCun, Y. et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[36] Li, N., Li, T., and Venkatasubramanian, S. “t-closeness: Privacy beyond k-anonymity and l-diversity”. In: *2007 IEEE 23rd international conference on data engineering*. IEEE. 2006, pp. 106–115.

[37] Li, Q., He, B., and Song, D. “Model-contrastive federated learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 10713–10722.

[38] Li, T. et al. “Federated optimization in heterogeneous networks”. In: *Proceedings of Machine learning and systems* 2 (2020), pp. 429–450.

[39] Machanavajjhala, A. et al. “l-diversity: Privacy beyond k-anonymity”. In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 1.1 (2007), 3–es.

[40] McMahan, B. et al. “Communication-efficient learning of deep networks from decentralized data”. In: *Artificial intelligence and statistics*. PMLR. 2017, pp. 1273–1282.

[41] Mugunthan, V. et al. “Smpai: Secure multi-party computation for federated learning”. In: *Proceedings of the NeurIPS 2019 Workshop on Robust AI in Financial Services*. MIT Press Cambridge, MA, USA. 2019, pp. 1–9.

[42] Nguyen, T. X. et al. “Federated learning in ocular imaging: current progress and future direction”. In: *Diagnostics* 12.11 (2022), p. 2835.

[43] Posner, J. et al. “Federated learning in vehicular networks: Opportunities and solutions”. In: *IEEE Network* 35.2 (2021), pp. 152–159.

[44] Al-Rubaie, M. and Chang, J. M. “Privacy-preserving machine learning: Threats and solutions”. In: *IEEE Security & Privacy* 17.2 (2019), pp. 49–58.

[45] Ryffel, T. et al. “A generic framework for privacy preserving deep learning”. In: *arXiv preprint arXiv:1811.04017* (2018).

[46] Saha, S. and Ahmad, T. “Federated transfer learning: Concept and applications”. In: *Intelligenza Artificiale* 15.1 (2021), pp. 35–44.

[47] Shao, J. et al. “Dres-fl: Dropout-resilient secure federated learning for non-iid clients via secret data sharing”. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 10533–10545.

[48] Stahl, B. C. and Stahl, B. C. “Ethical issues of AI”. In: *Artificial Intelligence for a better future: An ecosystem perspective on the ethics of AI and emerging digital technologies* (2021), pp. 35–53.

[49] Sweeney, L. "k-anonymity: A model for protecting privacy". In: *International journal of uncertainty, fuzziness and knowledge-based systems* 10.05 (2002), pp. 557–570.

[50] Tebaa, M., El Hajji, S., and El Ghazi, A. "Homomorphic encryption method applied to Cloud Computing". In: *2012 National Days of Network Security and Systems*. IEEE. 2012, pp. 86–89.

[51] Tran, C., Fioretto, F., and Van Hentenryck, P. "Differentially private and fair deep learning: A lagrangian dual approach". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 11. 2021, pp. 9932–9939.

[52] Truong, N. et al. "Privacy preservation in federated learning: An insightful survey from the GDPR perspective". In: *Computers & Security* 110 (2021), p. 102402.

[53] Wang, E. et al. "FLINT: A Platform for Federated Learning Integration". In: *Proceedings of Machine Learning and Systems* 5 (2023).

[54] Wang, H. et al. "Federated learning with matched averaging". In: *arXiv preprint arXiv:2002.06440* (2020).

[55] Wei, K. et al. "Vertical federated learning: Challenges, methodologies and experiments". In: *arXiv preprint arXiv:2202.04309* (2022).

[56] Wibawa, F. et al. "BFV-Based Homomorphic Encryption for Privacy-Preserving CNN Models". In: *Cryptography* 6.3 (2022), p. 34.

[57] Wibawa, F. et al. "Homomorphic encryption and federated learning based privacy-preserving cnn training: Covid-19 detection use-case". In: *Proceedings of the 2022 European Interdisciplinary Cybersecurity Conference*. 2022, pp. 85–90.

[58] Yang, C. et al. "Characterizing impacts of heterogeneity in federated learning upon large-scale smartphone data". In: *Proceedings of the Web Conference 2021*. 2021, pp. 935–946.

[59] Yang, Q. et al. "Federated machine learning: Concept and applications". In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 10.2 (2019), pp. 1–19.

[60] Yuan, H. et al. "Addressing Heterogeneity in Federated Learning via Distributional Transformation". In: *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXVIII*. Springer. 2022, pp. 179–195.

[61] Yurochkin, M. et al. "Bayesian nonparametric federated learning of neural networks". In: *International conference on machine learning*. PMLR. 2019, pp. 7252–7261.

[62] Zhao, W. et al. "Protea: Client profiling within federated systems using flower". In: *Proceedings of the 1st ACM Workshop on Data Privacy and Federated Learning Technologies for Mobile Edge Network*. 2022, pp. 1–6.

[63] Zhao, Y. et al. "Federated learning with non-iid data". In: *arXiv preprint arXiv:1806.00582* (2018).

[64] Zhou, T., Zhang, J., and Tsang, D. "FedFA: Federated Learning with Feature Anchors to Align Feature and Classifier for Heterogeneous Data". In: *arXiv preprint arXiv:2211.09299* (2022).