UNIVERSITÀ
DEGLI STUDI
DI PALERMO

Federated Learning per il Crowdsensing Partecipativo

Tesi di Laurea Magistrale in Ingegneria Informatica

C. Ferdico
Relatore: Prof. Giuseppe Lo Re
Correlatore: Prof. Marco Morana

Federated Learning per il Crowdsensing Partecipativo

Tesi di Laurea di

Cedric Ferdico

Relatore

Ch.mo Prof. Giuseppe Lo Re

Correlatore

Prof. Marco Morana

Sommario

Negli ultimi anni, l'ampia diffusione di dispositivi pervasivi intelligenti, in grado di fornire servizi basati sull'AI, ha incoraggiato la ricerca nella definizione di nuovi paradigmi di *Distributed Learning*. Il *Federated Learning* (FL) è uno degli approcci più recenti che consente ai dispositivi di collaborare per addestrare modelli AI, garantendo privacy e bassi oneri di comunicazione. Nonostante siano stati condotti diversi studi sul FL, manca ancora un'architettura generale e modulare in grado di funzionare bene in diversi scenari. Seguendo questa direzione, questo lavoro di tesi propone un framework di FL versatile la cui validità è valutata considerando uno scenario distribuito di *Human Activity Recognition* in cui i dispositivi personali degli utenti sono impiegati come base dell'infrastruttura di rilevamento. È stata eseguita un'analisi sperimentale per valutare l'efficacia dell'architettura rispetto a un approccio centralizzato proponendo anche un meccanismo di difesa per rendere il sistema più robusto ad una determinata categoria di attacchi. I risultati dimostrano la versatilità e la funzionalità della soluzione proposta.

Indice

Capitolo 1

Introduzione

Nell'arco dell'ultimo decennio, la larga diffusione di dispositivi smart, e la conseguente crescita della quantità di dati prodotti da essi, ha portato alla progettazione di soluzioni di *Artificial Intelligence* (AI) e *Machine Learning* (ML) che richiedono, tra gli altri requisiti, una sempre più rilevante potenza computazionale. Nonostante i più semplici modelli di *machine learning* possano essere addestrati con un modesto quantitativo di dati, le applicazioni che sfruttano modelli più complessi arrivano a richiedere Terabyte, o addirittura Petabyte, di dati per l'addestramento. La necessità di immagazzinare questo enorme quantitativo di dati, così come l'esigenza di una sempre crescente potenza di calcolo, si riflette direttamente sulla necessità di distribuire il carico computazionale delle operazioni di AI su un insieme di elaboratori interconnessi attraverso la Rete.

Il *Distributed Learning* (DL) è un paradigma recentemente introdotto dalla comunità scientifica proprio per affrontare i problemi relativi alla elaborazione di grandi quantità di dati dal punto di vista dei modelli di ML. L'idea è

quella di distribuire una parte del processo di apprendimento su cluster di dispositivi, mentre una singola *Entità Centrale* si occupa del miglioramento del modello globale sfruttando le informazioni ricevute dai client. Tra gli innumerevoli benefici, alcuni dei più rilevanti sono sicuramente la gestione di grandi volumi di dati per l'apprendimento, la suddivisione del carico del processo di apprendimento su macchine diverse, e la possibilità di realizzare algoritmi di apprendimento su dati distribuiti. Tali vantaggi trovano immediata applicabilità in un ampio ventaglio di scenari applicativi, dall'ambito finanziario, a quello industriale, fino al sanitario [1].

Nonostante ciò, la protezione dei dati sensibili non è un requisito a cui tale paradigma pone particolare attenzione. Il DL stabilisce infatti che per aggiornare i modelli di ML è necessario usare i dati condivisi dagli utenti, senza però tenere in conto che i dati possono non sempre essere disponibili a causa di restrizioni imposte dallo scenario applicativo. Per comprendere al meglio questo problema, si consideri un'applicazione nel contesto *healthcare* in cui i modelli di *machine learning* sono addestrati su dati prodotti da dispositivi personali degli utenti per il monitoraggio del loro stato di salute: per esempio, un ente sanitario vuole monitorare le attività dei suoi pazienti per valutare se questi compiano una sufficiente quantità di attività fisica nell'arco della giornata. Nella vita reale i dati raccolti dai pazienti devono essere conservati ed elaborati all'interno dell'ente sanitario stesso, altrimenti si avrebbero chiari problemi di privacy e sicurezza. Tuttavia, la condivisione di tali dati con una terza parte permetterebbe un netto miglioramento dei modelli di *machine learning* che potrebbero, a quel punto, contare su un enorme quantitativo di informazioni [2]. Per superare questi limiti, si ricorre generalmente alla

definizione di meccanismi che se da un lato garantiscono un certo grado di sicurezza, dall'altro introducono un maggiore costo dal punto di vista del carico computazionale. Si intuisce quindi che la sicurezza dei dati, insieme alla distribuzione dei modelli *machine learning*, rappresenta un parametro fondamentale a cui porre particolare attenzione in fase di progettazione di un'architettura distribuita.

Recentemente, il *Federated Learning* (FL) è stato proposto con lo scopo di superare i limiti del Distributed Learning, pur mantenendone le caratteristiche principali. Tecnicamente il FL prevede un approccio di apprendimento, collaborativo e distribuito, che consente l'addestramento di modelli di AI senza condividere i dati locali raccolti dai dispositivi di acquisizione [3]. Questa peculiarità è di fondamentale importanza in quanto rende possibile la collaborazione tra diverse organizzazioni, altrimenti impossibile [4]. Normalmente, infatti, organizzazioni che operano nello stesso settore, concorrenti o meno che siano, potrebbero potenzialmente trarre vantaggio da una collaborazione nella realizzazione di modelli, sfruttando le conoscenze acquisite singolarmente. Questa potenziale collaborazione sarebbe impossibile se non esistesse un paradigma come quello del FL che permette agli enti/organizzazioni di cooperare senza un'effettiva condivisione dei loro dati, locali e personali.

Oltre agli scenari cooperativi appena descritti, esistono applicazioni di *distributed computing*, normalmente vincolate dai limiti dovuti alla condivisione dei dati, e di tutto ciò che ne comporta, in cui il paradigma federato può apportare diversi benefici. Il Distributed Human Activity Recognition (DHAR) è sicuramente uno dei contesti più interessanti in cui valutare come il FL possa fornire un significante miglioramento alle performance del sistema. La

salvaguardia della privacy assicurata da questo paradigma, unita alle buone performance ed alla riduzione degli oneri di comunicazione durante le fasi di addestramento, rappresenta la risoluzione perfetta ai problemi per questo tipo di scenario applicativo. Questo nuovo paradigma, infatti, potrebbe essere applicato per gestire automaticamente le impostazioni dei dispositivi personali degli utenti, in base al rilevamento delle loro attività, oppure per permettere ad un ente (ad es. uno Smart Campus) di fornire servizi migliori o adeguarli dinamicamente in base al comportamento attuale della comunità. Questi scenari seppur simili, sono molto diversi tra loro poiché uno viene istanziato sul contesto dell'utente, mentre l'altro nel contesto di uno *smart environment*. Per ciascuno di essi sarebbe infatti necessario progettare un'architettura ad-hoc che se da un lato soddisfa pienamente tutti i requisiti, dall'altro la rende difficilmente applicabile in contesti diversi da quella per cui è stata pensata. Per questo motivo nasce la necessità di realizzare architetture modulari applicabili al paradigma FL, altamente versatili, e in grado di essere istanziate in molteplici applicazioni attraverso delle semplici modifiche al funzionamento dei moduli che la compongono.

In fase di progettazione, è anche necessario considerare che, a causa della sua natura intrinseca e dell'elevato grado di modularità, gli approcci basati sul FL sono particolarmente soggetto ad attacchi di sicurezza informatica che mirano a peggiorare le performance del sistema, o consentire agli attaccanti di raggiungere scopi personali ben precisi. Tra le diverse minacce, quelle che si focalizzano sull'integrità dei dati sono le più diffuse a causa dell'elevata facilità con cui è possibile compromettere le informazioni condivise dai client. Infatti, in tali sistemi, niente e nessuno assicura che i partecipanti fornisca-

no dati di qualità per l'affinamento del modello. Nel caso più generale, la qualità dei dati potrebbe essere compromessa da normali malfunzionamenti e/o guasti che possono interessare i dispositivi di raccolta dei dati. Nel caso peggiore, invece, la compromissione dei dati potrebbe essere opera di un'entità malevola che *intenzionalmente* mira a peggiorare le performance del sistema, il tutto solo per raggiungere i suoi scopi [5]. A causa dei fattori sopra elencati, è necessario quindi definire degli algoritmi e delle metodologie che abbiano l'obiettivo di contrastare questi comportamenti indesiderati andando a mitigare, se non a risolvere, gli effetti causati da questi ai danni del sistema.
Alla luce di queste osservazioni, nell'ambito di questo lavoro di tesi, gli obiettivi preposti sono i seguenti:

- Definizione di un'architettura modulare basata sul paradigma del Federated Learning per lo scenario del Distributed Human Activity Recognition (DHAR). L'architettura proposta potrà essere impiegata in un'ampia gamma di scenari applicativi, richiedendo semplicemente di modificare il funzionamento delle componenti, o le entità coinvolte. Nello scenario considerato, i dati vengono raccolti dai dispositivi personali degli utenti, mentre le *entità federate* sono responsabili per l'effettiva fase di riconoscimento delle attività, ed aggiornamento dei modelli locali. L'*entità centrale* remota è infine responsabile del mantenimento di una visione complessiva e coerente dell'intero set di attività e si occuperà dell'aggiornamento del modello globale di ML.

- Definizione e progettazione di un algoritmo di difesa contro attacchi che minano l'integrità dei dati raccolti dai dispositivi personali degli utenti.

In particolare, è stato esaminato un attacco di *Label Flipping* in cui l'attaccante scambia la *label* di una classe con un'altra con l'obiettivo di far classificare erroneamente una certa attività al sistema. L'algoritmo di difesa, eseguito dall'entità centrale, mira ad analizzare i contributi ricevuti dalle entità federate ed eliminare quelli ritenuti malevoli.

Il resto del documento è organizzato nel seguente modo: all'interno del capitolo 2 verrà introdotto il *Distributed Learning* andando, in seguito, ad approfondire il *Federated Learning* (FL), dalle sue declinazioni sino alle varianti degli algoritmi di aggregazione dei contributi. Successivamente, verrà introdotto lo *Human Activity Recognition* e come questo si lega al paradigma del FL. A conclusione del capitolo, verrà affrontata un'analisi del materiale presente in letteratura riguardo le minacce a cui è soggetto il FL.
Nel capitolo 3 verrà introdotto il sistema proposto partendo dallo schema architetturale, con una descrizione di alto livello. Sarà quindi introdotta ed approfondita l'architettura modulare proposta indicando le funzioni svolte da ogni modulo. In conclusione, l'architettura proposta verrà istanziata nello scenario HAR andando ad affinare il comportamento dei moduli in base alle esigenze del contesto.
Nello sviluppo del capitolo 4 verrà affrontato il tema della sicurezza andando a configurare lo scenario di attacco secondo uno specifico *Threat Model*, che sarà approfondito ed esposto minuziosamente. In seguito alla definizione dello scenario di attacco verrà esposto l'algoritmo di difesa proposto in questo lavoro di tesi.
Più avanti nell'elaborato, all'interno del capitolo 5, verranno definite le metriche secondo le quali verrà valutato il raggiungimento degli obiettivi. Verrà

dimostrato, nel corso della valutazione delle prestazioni, che l'architettura proposta risulta essere una soluzione dalle buone performance, versatile e sicura, sia nel trattamento dei dati che dal punto di vista dell'affidabilità. Successivamente verranno mostrati gli effetti dell'attacco condotto, ed il funzionamento dell'algoritmo di difesa proposto. In conclusione a questo capitolo verranno mostrati i limiti della soluzione proposta.

Infine, nel corso del capitolo 6 verranno tratte le conclusioni del lavoro svolto andando ad analizzare i limiti dell'algoritmo proposto e proponendo degli sviluppi futuri per migliorarlo. Parte di questo lavoro di tesi è pubblicato in [6].

Capitolo 2

Distributed Machine Learning

Il Federated Learning (FL) è strettamente correlato con il Distributed Learning (DL), un sistema distribuito tradizionale che prevede la spartizione dell'onere computazionale così come dello storage dei dati su diversi client, ma se ne discosta sotto diversi aspetti. Uno scenario di DL mira a connettere diversi dispositivi tramite la rete internet sotto il controllo di un server centrale, in questo modo ogni elemento interconnesso svolge parte differente del lavoro per raggiungere un obiettivo comune, con il principale scopo della distribuzione concentrato nell'accelerazione dell'elaborazione. Differentemente, il FL pone il focus sulla costruzione di un sistema collaborativo senza alcuna distribuzione di dati [7]. Infatti questo paradigma è stato ideato nell'ottica di soddisfare i requisiti del calcolo distribuito dedicando un ruolo di primaria importanza alla salvaguardia della privacy.

Nello sviluppo del presente capitolo si tratterà del *Federated Learning*, approfondendo le diverse tipologie in cui viene differenziato in letteratura; a proseguire il focus sarà spostato nei confronti dell'algoritmo *FedAvg* e le sue

varianti.
Successivamente sarà approfondito il mondo dello Human Activity Recognition andando a vedere come il FL si amalgama con questo scenario. In conclusione verrà effettuata una disamina degli attacchi a cui sono suscettibili i sistemi di FL e su come questi vengano affrontati in letteratura.

2.1 Federated Learning

Il Federated Learning (FL) consiste in un processo di apprendimento distribuito in cui un *Entità Centrale* (Central Entity, CE) e un gruppo di *Entità Federate* (Federated Entity, FE) collaborano per l'addestramento e l'affinamento di un modello globale. La collaborazione tra questi attori può avvenire secondo diversi schemi, tra cui i più comuni sono due: uno prevede che tutto parta dalle entità federate (A), l'altro stabilisce il punto di partenza nell'entità centrale (B).
In termini generali, l'attività di apprendimento distribuito parte dall'Entità Centrale che, sfruttando un set di dati iniziale, addestra e condivide alle Entità Federate un modello globale, chiamato *Modello Federato*. Dopo aver ricevuto tale modello, queste cominceranno a collezionare dati e a processarli secondo lo scenario considerato. Successivamente, il raffinamento del modello federato verrà eseguito tramite un processo collaborativo tra la CE e le FE: tale attività può essere svolta al momento del ricevimento del modello federato (se ogni entità federata possiede già dei dati) o dopo che ogni entità federata avrà raccolto dei nuovi dati. Le condizioni sotto cui avviene l'aggiornamento del modello devono essere concordate con l'entità centrale.

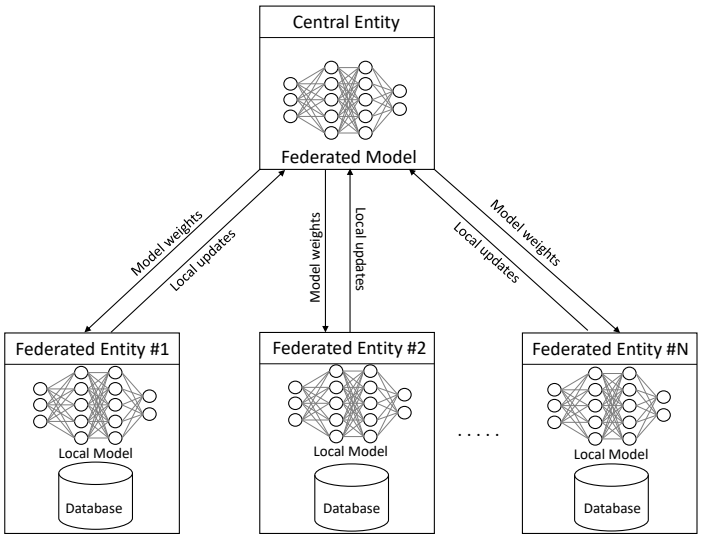

Figura 2.1: Un'architettura di Federated Learning generica.

Quello descritto precedentemente (e mostrato in Figura 2.1) è ciò che avviene nello schema (B). A carattere informativo, in caso si utilizzi lo schema (A) le prime due operazioni non vengono effettuati, inoltre, è necessario che tutte le entità federate condividano tra esse almeno la struttura del modello.
Analizzando più a fondo le caratteristiche del FL, questo risulta essere un particolare paradigma di distributed machine learning che si differenzia dagli altri approcci almeno su tre punti chiave [8]:

1. Il FL non permette il trasferimento di dati "grezzi", mentre altri metodi non prevedono restrizioni a riguardo. Dato che i dati appartengono a differenti proprietari, gli approcci basati sul FL, e quindi caratterizzati da questa restrizione, rispettano i requisiti definiti dai vari regolamenti legislativi sulla protezione dei dati personali (ad esempio il GDPR [9]). Nello specifico, l'espressione del consenso (GDPR, art.6) e il princi-

pio di minimizzazione dei dati (GDPR, art.5), limitano la raccolta dei dati e la loro conservazione solamente a ciò che è espressamente consentito dall'utente e a ciò che è assolutamente necessario ai fini delle elaborazioni.

2. Il FL sfrutta le risorse derivanti dall'elaborazione distribuita in dispositivi che potenzialmente sono situati in regioni geografiche e/o organizzazioni differenti. In contrapposizione, gli altri approcci solitamente impiegano solo le risorse di un singolo server o un cluster di server all'interno di una singola regione geografica e/o appartenenti ad una singola organizzazione. Per questo motivo, una delle caratteristiche chiave del FL è quella di rendere possibile la collaborazione tra più organizzazioni.

3. Generalmente, gli approcci di apprendimento distribuito prestano poca attenzione ai problemi di sicurezza relativi alla protezione dei dati, cosa che invece il FL vede come uno dei suoi punti di forza. Il FL, infatti, assicura la riservatezza e la sicurezza dei dati utilizzati nel processo, anche perché un'eventuale perdita di dati comporterebbe non solo danni dal punto di vista reputazionale ma anche da quello economico [10]. Inoltre, nell'applicazione di questo particolare paradigma di apprendimento distribuito, è possibile adottare ulteriori pratiche di sicurezza (quali la cifratura) qualora lo scenario applicativo lo necessitasse [11].

Volendo riassumere le tre caratteristiche generali di questo paradigma, queste sono:

- **Universalità nelle collaborazioni inter-organizzazioni**: non essendo necessario alcun tipo di condivisione di dati tra i partecipanti al

FL, potenzialmente, ogni organizzazione potrebbe collaborare con ogni altra (ammesso che abbia senso la collaborazione) in modo da creare valore per tutti coloro che partecipano senza dover condividere alcun dato.

- **Applicabilità in scenari Non-IID**: nel FL possono collaborare migliaia di entità e non è detto che tutte abbiano una distribuzione dei dati identica ed indipendente. Uscire dalla assunzione IID non causa alcun problema al sistema federato (a differenza del DL che opera principalmente in scenari IID).

- **Status equo per ogni nodo**: essendo uno scenario collaborativo, ogni partecipante gode dello stesso status con lo scopo di raggiungere la prosperità comune.

In sostanza, l'applicazione del paradigma FL permette ad un ampio numero di dispositivi, chiamati *Entità Federate* e che posseggono dati riguardo delle osservazioni, di addestrare localmente e collaborativamente un unico modello di machine learning senza alcuna condivisione dei propri dati "grezzi". A questo scopo un server centrale, che si occupa di coordinare il tutto, aggrega i contributi di tutte le entità partecipanti in un modello aggiornato, che successivamente ricondivide con tutti i client che hanno partecipato, affinché ognuno di essi possa beneficiare dell'esperienza di apprendimento degli altri.

2.1.1 Tipologie di Federated Learning

Secondo lo studio condotto in [12], gli approcci che sfruttano il paradigma del FL possono essere suddivisi in tre categorie differenti, ovvero "Federated Lear-

ning orizzontale", il "Federated Learning verticale" ed il "Federated Transfer Learning". Tutte le suddette varianti verranno approfondite di seguito.

Federated Learning Orizzontale

Attualmente, gli algoritmi FL esistenti sono stati finalizzati in particolar modo nell'ambito delle applicazioni in cui fossero coinvolti dispositivi intelligenti o comunque appartenenti al mondo dell'Internet delle cose (IOT). In questi scenari, di solito, il FL potrebbe essere classificato come "orizzontale" in quanto i dati possono differire significativamente nello spazio campionario ma, allo stesso tempo, hanno uno spazio delle feature simile, se non uguale. Infatti, volendo sintetizzare una definizione: *si parla di Federated Learning "orizzontale" quando è presente una certa sovrapposizione tra la feature dei dati diffusi su vari nodi, nonostante questi siano abbastanza differenti all'interno dello spazio campionario.*

In alcune applicazioni però, potrebbe risultare impossibile adattare i dati raccolti ad una comune rappresentazione delle feature o potrebbe verificarsi la mancanza di dati etichettati, una situazione simile renderebbe impraticabile la via del FL. Per affrontare questi problemi, in [13] è stato introdotto il "FL orizzontale eterogeneo gerarchico" (HHHFL). Sostanzialmente, la soluzione proposta in questo studio consiste nell'adottare una rappresentazione delle dominio del problema tramite il più grande sottoinsieme comune. Questo approccio rende possibile realizzare una rete federata anche fra entità che rappresentano il dominio del problema tramite un insieme di feature simile, come mostrato in Figura 2.2.

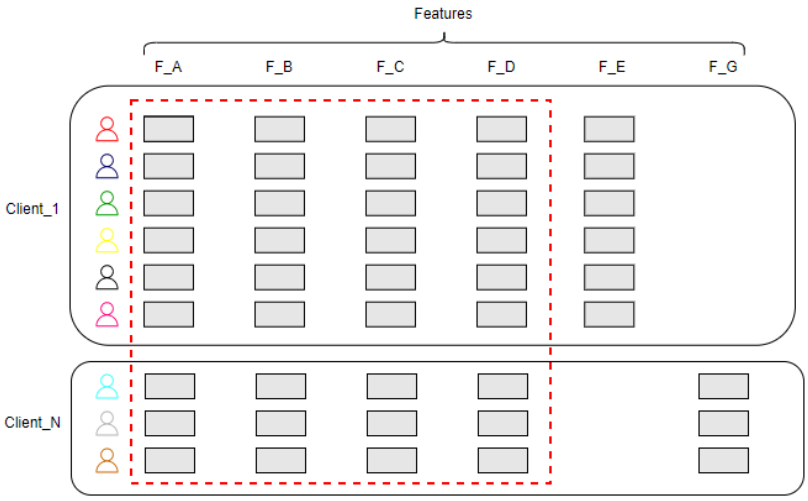

Figura 2.2: Esempio di FL orizzontale.

Federated Learning Verticale

Il FL verticale è adatto per quei casi in cui i dati sono partizionati in direzione verticale in base alla dimensione del vettore delle feature. Tutte le parti detengono dati omogenei, il che significa che presentano una sovrapposizione parziale dello spazio degli identificativi (ID) dei campioni mentre differiscono nello spazio delle feature.

Si prenda ad esempio il caso in cui un istituto medico intenda identificare malattie come il diabete mellito in modo predittivo. Secondo la ricerca, le persone che soffrono di alta pressione sanguigna e di obesità possono essere soggette a sviluppare il diabete di tipo 2 [14]. Pertanto, l'analisi predittiva potrebbe essere basata alla luce di alcuni parametri approssimativi, come l'età dei pazienti, il peso e l'anamnesi. Tuttavia, se un giovane non presenta obesità o ipertensione ma assume più calorie del dovuto e non pratica attività fisica, anch'esso è da considerare incline al potenziale insorgere della

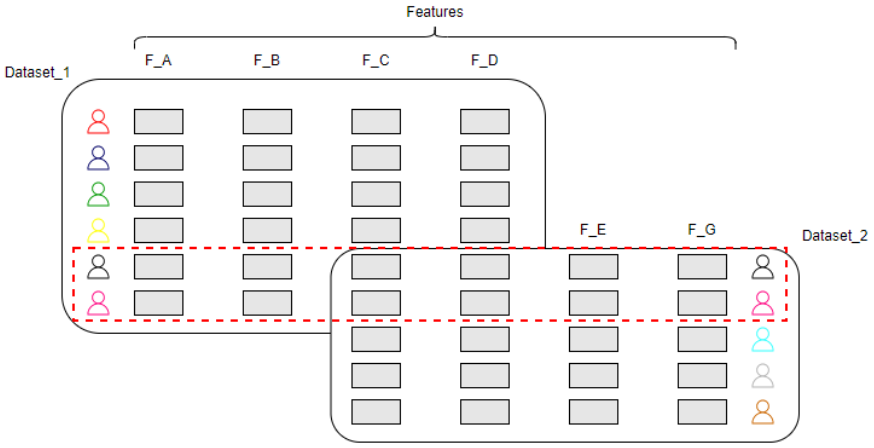

Figura 2.3: Esempio di FL verticale.

malattia. In questo caso, non sarebbe possibile prevedere con efficacia la sua predisposizione con il modello basato su dati approssimativi, a causa della mancanza di informazioni.

Lo sviluppo del FL verticale ha reso possibile lo sviluppo di sistemi che integrassero le informazioni provenienti da più organizzazioni in un unico modello. Per risolvere il problema dell'esempio esposto precedentemente, basi pensare alla collaborazione che potrebbe avvenire tra alcune aziende, che detengono set di dati presi da smartphone con applicazioni che svolgono la funzione di contapassi o monitorano la dieta, e l'istituto medico in questione. In quel caso, il modello addestrato sui dati personali delle due organizzazioni riuscirebbe ad essere esaustivo per la predizione della malattia in questione. Un esempio di FL verticale è riportato in Figura 2.3.

Federated Transfer Learning

A differenza degli scenari di FL orizzontale e FL verticale, in alcuni casi i dati possono non condividere né lo spazio campionario né lo spazio delle feature. Pertanto, il problema principale che viene a sorgere è la mancanza di dati etichettati con conseguente scarsa qualità di questi ultimi. Il transfer learning consente di spostare la conoscenza di un dominio (cioè il dominio di origine) in un altro dominio (cioè il dominio di destinazione) per ottenere risultati di apprendimento migliori. Per ovviare a questo problema, in [15] è stato proposto il Federated Transfer Learning (FTL). Il FTL nasce come generalizzazione del FL, con lo scopo di realizzare un paradigma più versatile da impiegare quando i partecipanti presentano obiettivi comuni ma con piccole intersezioni nella rappresentazione del dominio del problema.
Tuttavia, la ricerca in FTL non è ancora matura, quindi c'è ancora molto spazio di crescita per renderla più flessibile con diverse strutture di dati ed implementazioni.

2.2 *FedAvg* e varianti

Il Federated Averaging, riferito anche come *FedAvg*, è un algoritmo di aggregazione eseguito dal CE, con lo scopo ottenere il modello globale aggiornato a partire dai contributi dei FE. L'esecuzione dell'algoritmo *FedAvg* prevede che, in primo luogo, il CE inizializzi i parametri del modello tramite un addestramento svolto sui dei dati "di partenza" in suo possesso e lo condivida con tutti i FE partecipanti al processo di apprendimento federato. A seguire, ognuno dei partecipanti, una volta ricevuto il modello globale, esegue

l'apprendimento localmente minimizzando la seguente funzione di loss locale

$$f_c(W) = \frac{1}{|D_c|} \sum_{i \in D_c} l(W; x_i, y_i), \tag{2.1}$$

dove $l(W; x_i, y_i)$ rappresenta la loss function per ogni campione dei dati x_i; W rappresenta la matrice dei pesi tra i neuroni; D_c rappresenta il dataset locale.

La minimizzazione della funzione di loss avviene tramite l'algoritmo di *Stochastic Gradient Descent* (SGD) sui dataset che ognuno di essi possiede. Dopo che i contributi mandati da tutti i partecipanti vengono ricevuti, il CE minimizza la funzione di loss globale (riportata nell'equazione 2.2) attraverso la media pesata dei parametri ricevuti da ogni FE.

$$F(W) = \sum_{c=1}^{C} \frac{|D_c|}{S} f_c(W) \tag{2.2}$$

Dove $S = \sum_{c=1}^{C} |D_c|$, ovvero la somma delle dimensioni dei dataset locali.

Calcolato il modello globale aggiornato, il CE lo ridistribuisce a tutti i partecipanti per il prossimo round di addestramento.

Il processo di addestramento federato continua fino a quando una certa condizione non è soddisfatta: fino alla convergenza della funzione globale di loss, fino al raggiungimento di un certo livello di accuratezza desiderato o semplicemente per un numero definito di round.

La complessità dell'algoritmo *FedAvg* è trattata ed analizzata in [16], studio in cui gli autori dimostrano teoricamente che questo algoritmo converge all'ottimo globale ad una velocità di $O(1/T)$ per problemi fortemente convessi

e smooth, dove T è il numero di iterazioni di *Stochastic Gradient Descent*. Con il susseguirsi dei lavori di ricerca a riguardo, sono state proposte diverse varianti di FedAvg per adattare l'aggregazione degli aggiornamenti del modello in diversi scenari di apprendimento federato. Segue una breve introduzione di queste ultime:

- *FedProx*: il FedProx [17] è una variante di FedAvg che include un componente per rappresentare l'eterogeneità statistica tra i partecipanti all'apprendimento federato. L'idea principale di Fedprox è quella di aggiungere un termine prossimale al sotto-problema dell'addestramento locale su ciascuno di essi. Questo termine obbliga gli aggiornamenti locali ad essere più "vicini" al modello globale iniziale con l'obiettivo di limitare la loro influenza. Similmente a quanto avviene in FedAvg, in FedProx, tutti i dispositivi sono pesati allo stesso modo nella fase di aggregazione globale, in modo tale che le differenze nelle capacità del dispositivo (ad es. Hardware) non vengano prese in considerazione.

- *FedPAQ*: il FedPAQ [18] è una tecnica di aggregazione orientata alla comunicazione che suggerisce di effettuare una media periodica degli aggiornamenti del modello locale. In particolare, invece di sincronizzare gli aggiornamenti del modello con il server ad ogni iterazione, FedPAQ consente ai partecipanti di eseguire più aggiornamenti locali sul modello prima di condividerli, tramite la loro media, con il server. Per ridurre ulteriormente l'overhead di comunicazione, FedPAQ consente anche una partecipazione parziale dei partecipanti, in cui, ad ogni iterazione, solo un sottogruppo di questi viene scelto per partecipare all'addestramento sulla base di diversi fattori come (1) il collegamento

di quest'ultimo ad una rete wireless gratuita, (2) l'eventuale inattiva e (3) raggiungibilità. Similmente a FedAvg, il nuovo modello globale in FedPAQ è calcolato come la media dei modelli locali.

- *Turbo-Aggregate*: il Turbo-Aggregate [19] è una tecnica di aggregazione, orientata alla comunicazione e alla sicurezza, che basa l'aggregazione dei model update su una strategia circolare multi-gruppo. Nello specifico, i partecipanti sono divisi in più gruppi e, ad ogni iterazione, gli appartenenti ad un gruppo trasmettono gli aggiornamenti del modello aggregato di tutti i client dei gruppi precedenti e di quelli locali del gruppo corrente ai client del gruppo successivo. Turbo-Aggregate include anche un componente di sicurezza che impiega un meccanismo di condivisione segreto additivo, la cui idea principale è quella di aggiungere un po' di casualità a ciascun modello locale. Questa componente casuale è destinata a scomparire con l'aggregazione dei modelli locali. Turbo-Aggregate è abbastanza adatto per topologie wireless, in cui le condizioni di rete e la disponibilità dell'utente possono variare rapidamente. Tuttavia, il meccanismo di aggregazione sicura incorporato in Turbo-Aggregate, sebbene efficace nella gestione dell'abbandono degli utenti, non si adatta bene all'aggiunzione di nuovi utenti alla rete.

- *FedMA*: il FedMA [20] è una tecnica di aggregazione orientata alla statistica che tiene conto dell'invarianza alla permutazione dei neuroni della rete neurale prima di effettuare l'aggregazione. L'obiettivo è consentire l'adattamento globale delle dimensioni del modello. FedMA utilizza un meccanismo bayesiano, non parametrico, che consente di adattare la

dimensione del modello centrale all'eterogeneità della distribuzione dei dati. Tuttavia, il FedMA può essere vulnerabile all'attacco di poisoning del modello, in cui un avversario può facilmente ingannare il sistema per espandere il modello globale al fine di accogliere qualsiasi modello locale malevolo.

- *HierFAVG*: HierFAVG [21] è una tecnica di aggregazione orientata alla comunicazione che mira a promuovere l'aggregazione parziale del modello a livello di edge server. Nello specifico, HierFAVG si basa su un'architettura gerarchica client-edge-cloud in base alla quale ogni edge server può aggregare gli aggiornamenti del modello dei propri partecipanti. Successivamente, dopo un numero fisso di aggregazioni, i modelli aggregati a livello di edge vengono inoltrati a un server cloud per un'aggregazione globale. Tale struttura multi-livello consente uno scambio di modelli più efficiente sull'architettura edge-cloud esistente. Tuttavia, HierFAVG è ancora vulnerabile ai problemi di ritardi nelle comunicazioni e/o abbandono dei dispositivi finali.

2.3 Human Activity Recognition

Negli ultimi anni l'attività di ricerca dedicata allo Human Activity Recognition (HAR) sta guadagnando una notevole attenzione in quanto esiste una moltitudine di scenari applicativi in cui un processo di HAR può essere adottato con successo: gestione della mobilità urbana, definizione di ambienti intelligenti oppure di tecnologie di *Assisted Living* [22].

Generalmente, è possibile effettuare il riconoscimento delle attività umane sfruttando due tipi di diversi di dati di input: dati visivi (immagini) o dati sensoriali (provenienti da letture di accelerometro e giroscopio). L'idea di base, su cui pone le fondamenta lo HAR effettuato tramite le immagini, è quella di descrivere l'utente attraverso delle sagome che permettono di estrarre informazioni riguardo i suoi movimenti. A questo scopo possono essere impiegate immagini di ogni tipo, da quelle RGB a quelle di profondità, mandandole in input ad un modello di machine learning [23] attraverso il quale sarà effettuato, dopo la fase di addestramento, il riconoscimento delle attività.

Seguendo questa metodologia sono state sviluppate e proposte diverse applicazioni, specialmente all'interno del contesto dei sistemi ambientali intelligenti, ad es., nella gestione automatica delle risorse energetiche o nella sorveglianza intelligente [24]. Nonostante la chiara utilità, il riconoscimento delle attività umane basato sulle immagini soffre di importanti limitazioni che ne ridimensionano il possibile impiego. Nello specifico, l'utilizzabilità di questo approccio è circoscritta ad ambienti al chiuso e risulta essere particolarmente pesante in termini di carico computazionale [25].

Dati gli evidenti limiti del precedente approccio al problema HAR, sono state proposte delle tecniche alternative basate sui sensori. In questo contesto, le attività umane possono essere considerate intuitivamente come una sequenza di schemi ricorrenti nei dati "grezzi" acquisiti tramite i dispositivi indossati dall'utente, come smartwatch o smartband , etc., oppure tramite i sensori integrati negli smartphone. Nel corso degli anni in letteratura sono stati presentati molti algoritmi per lo HAR e, generalmente, si differenzia-

no principalmente sul tipo (ed il numero) di sensori utilizzati oppure sulla composizione del vettore delle feature estratte dalle letture sensoriali.
Per esempio, in [26] gli autori presentano un framework per lo HAR che lavora utilizzando i dati triassiali raccolti per mezzo di accelerometro e giroscopio, sensori integrati negli smartphone. I dati temporali generati da questi sensori sono stati prima di tutto raccolti in finestre temporali, la cui dimensione è stata adattata sperimentalmente; dopodiché, così raccolti, i dati sono stati analizzati con lo scopo di modellare le attività tramite un vettore delle feature 30-dimensionale ed infine le attività sono state classificate tramite un approccio di machine learning.
Un altro esempio interessante presentato in letteratura è quello che è stato implementato nelle Google Activity Recognition API per Android [1]. Tuttavia, questo strumento rappresenta una "black box" per l'utilizzatore in quanto gli sviluppatori non forniscono alcun metodo per comprendere quali feature vengano impiegate nel processo di riconoscimento delle attività o quale modello venga utilizzato per la classificazione.
Tutti gli approcci discussi precedentemente soffrono di due problematiche derivanti da (i) la crescente domanda di modelli e servizi intelligenti sempre più performanti e (ii) le preoccupazioni degli utenti in merito alla condivisione di dati sensibili (ad esempio, le attività svolte) con terze parti.
Col passare del tempo, e con i successivi lavori di ricerca, il primo problema è stato mitigato proponendo soluzioni scalabili ed efficienti in termini di tempo che sfruttano i paradigmi di *Cloud Computing*, *Fog Computing* o *Edge Computing* per fornire servizi HAR. Mentre il *Cloud Computing* po-

[1] `https://developers.google.com/location-context/activity-recognition`

trebbe rappresentare una soluzione praticabile per spostare il pesante carico computazionale verso il cloud, la sua applicabilità nelle applicazioni in tempo reale risulta essere fortemente limitata a causa del continuo trasferimento di dati che deve avvenire da/verso il cloud. Per questo motivo, dunque, è stata analizzata approfonditamente l'applicabilità in scenari HAR dei restanti paradigmi, di *Fog Computing* e *Edge Computing*. In [27] viene proposta un'architettura basata su un'architettura fog per il riconoscimento di attività umane complesse. In questo studio, le letture dei sensori vengono processati il più vicino possibile alla fonte di dati in modo che sia possibile soddisfare il vincolo dell'elaborazione in tempo reale. In particolare, il modello di machine learning viene eseguito su entità computazionalmente potenti all'interno della rete (per esempio entità Fog) a causa della complessità di questo specifico processo di riconoscimento delle attività, ovvero, una combinazione di Clustering, Support Vector Machine (SVM) e modelli Markoviani nascosti (HMM).

L'idea di distribuire il peso dell'elaborazione tra dispositivi intermedi (di fog) o ai dispositivi remoti (il cloud) alleggerisce sicuramente la crescente domanda di servizi più potenti, ma non affronta problemi di privacy o sicurezza. Infatti, sia per la loro natura distribuita che per l'elevato grado di modularità, i sistemi edge-fog-cloud computing sono particolarmente soggetti a cyber-attacchi alla sicurezza che possono essere eseguiti contro ogni elemento dell'infrastruttura [28].

Con lo scopo di trovare una soluzione ai problemi di sicurezza e distribuzione dell'onere computazionale, sono stati mossi i primi passi nell'applicazione del *paradigma federato* nell'ambito dello HAR. Molti ambiti hanno potuto

trarre beneficio dall'applicazione di tale paradigma, spaziando dal dominio finanziario [29] a quello del monitoraggio [30] fino a comprendere la sfera della sanità [31]. Sono proprio gli applicativi sviluppati all'interno di quest'ultimo dominio a fare pesantemente leva sul riconoscimento delle attività umane, come mezzo per fornire un servizio all'utente finale. Per questo motivo HAR sta assumendo un ruolo di grande interesse per la comunità di ricerca nell'ambito dell'apprendimento federato. Per questa ragione lo HAR sta acquisendo sempre più interesse per la comunità di ricerca nell'ambito dell'apprendimento federato assumendo, col passare del tempo, un ruolo di centrale importanza.

Tra gli studi riportati in letteratura, in [32] gli autori hanno valutato le performance dell'applicazione di una *Softmax regression* ed una rete neurale profonda per eseguire il riconoscimento delle attività umane. I risultati prodotti da questo lavoro hanno dimostrato che il FL può effettivamente apportare un miglioramento significativo, consentendo di ottenere un'accuratezza accettabile preservando la privacy.

In [2], invece, viene proposto e discusso un framework chiamato FedHealth. Questo sfrutta un paradigma di Federated Transfer Learning con lo scopo di fornire un servizio di assistenza sanitaria tramite dispositivi indossabili (ad es. smartband, smartwatch, ecc). Inoltre, attraverso il Federated Learning e la cifratura omomorfica. FedHealth effettua un'operazione di aggregazione dei dati provenienti da differenti organizzazioni con lo scopo di sintetizzare modelli di machine learning molto solidi, preservando perfettamente la privacy degli utenti. Successivamente alla realizzazione del modello sul cloud, FedHealth utilizza dei metodi di trasfer learning per realizzare un modello di

apprendimento personalizzato per ogni organizzazione partecipante.
Seguendo tutte le informazioni che è possibile trovare in letteratura, il Federated Learning mostra chiari vantaggi teorici rispetto al classico apprendimento centralizzato dalla prospettiva del calcolo. Ma poco si sa su come questi vantaggi vengano effettivamente raggiunti nella pratica e su come effettivamente tali approcci di apprendimento si comportino [33]. Motivati da queste ragioni, proponiamo un'architettura FL valutando come potrebbe funzionare in uno scenario HAR.

2.4 Minacce

A causa della sua natura intrinseca, il Federated Learning è bersaglio di una varietà di attacchi che mirano a manipolare il processo di apprendimento collaborativo [34]. Gli attacchi realizzabili ad un sistema di FL possono essere categorizzati in due macro-classi [5], ovvero *attacchi mirati* e *attacchi non mirati*. Gli Attacchi non mirati (spesso indicati come "Byzantine") vengono eseguiti con lo scopo di deteriorare le prestazioni del sistema o di causare qualche sorta di fallimento nel processo di addestramento, generalmente senza prendere di mira nessun partecipante in particolare o campioni di dati specifici. Dall'altra parte, gli attacchi mirati hanno chiari obiettivi di attacco e possono impiegare diverse tecniche sofisticate per raggiungerli.
Di seguito, verranno approfondite entrambe le categorie di attacchi e saranno discusse le tecniche esistenti che vengono proposte contrastare ciascuna di esse.

2.4.1 Attacchi "Non Mirati"

Conosciuti anche sotto il nome di "Byzantine Attack", sono quella classe di attacchi che cercano di degradare le prestazioni del modello di apprendimento nel suo complesso senza prendere di mira alcun client in particolare e senza avere come bersaglio nessuna istanza di dati specifica. Gli attacchi bizantini sono stati ampiamente studiati nel contesto dell'apprendimento federato e, come risultato di questi studi, sono stati proposti una varietà di approcci per analizzarli e, da questi, sono stati proposti opportuni meccanismi di difesa. Questi approcci possono essere classificati in tre tecniche principali, vale a dire, *analisi di sicurezza*, *analisi statistiche* ed *autoencoders*. Di seguito saranno illustrate ciascuna di queste tecniche e saranno discussi esempi di soluzioni proposte seguendo i suddetti approcci.

- *analisi di sicurezza*: Gli approcci che rientrano in questa categoria cercano di analizzare il comportamento degli attaccanti bizantini, discutono l'impatto delle loro azioni ed esaminare l'efficienza di alcune strategie di difesa esistenti contro di loro. Per esempio, in [35], gli autori discutono la vulnerabilità del FL verso una classe di attacchi di *poisoning* denominata *model poisoning*. Questo attacco può essere eseguito utilizzando la tecnica della *sostituzione del modello* in cui uno o un gruppo di attaccanti collusi tentano di sostituire il modello globale condiviso con un modello dannoso, facendo sì che il risultato dell'aggregazione classifichi erroneamente gli input futuri. Gli autori discutono anche delle potenziali strategie di difesa contro tali tipi di attacchi e sostengono che le soluzioni proposte possono essere facilmente aggirate dagli attaccanti. La pericolosità degli attacchi di *model poisoning*

deriva quindi dalla loro capacità di causare classificazioni erronee degli input senza manipolare gli input stessi. In [36], gli autori indagano sugli attacchi bizantini applicati al machine learning distribuito. In primo luogo dimostrano che un singolo attaccante bizantino può costringere il server di aggregazione a scegliere qualsiasi vettore di parametri arbitrario, anche uno piuttosto lontano dagli altri vettori. Successivamente, mostrano che una regola di aggregazione basata sulla distanza al quadrato (cioè che seleziona il vettore dei parametri che minimizza la somma del quadrato distanze da ogni altro vettore) è efficiente nel contrastare un singolo attaccante bizantino, ma diventa inefficiente quando due o più attaccanti colludono insieme. In conclusione, i metodi di analisi, utilizzati per contrastare gli attaccanti bizantini, non sono del tutto efficaci e i risultati positivi sono circoscritti a determinate circostanze.

- *analisi statistica*: L'analisi statistica consiste in un insieme di metodologie e tecniche per raccogliere, esaminare, analizzare e trarre conclusioni dai dati. Nel contesto della sicurezza del FL, sono stati utilizzati metodi statistici per esaminare gli aggiornamenti locali generati dai partecipanti, analizzando la loro somiglianza/dissomiglianza e traendo conclusioni appropriate sull'esistenza/inesistenza di attacchi bizantini. Ad esempio, gli autori di [37] propongono un approccio di FL resiliente agli attacchi bizantini chiamato *Adaptive Federated Averaging*. La soluzione ivi proposta sfrutta un modello Markoviano nascosto per identificare ed eliminare gli aggiornamenti locali malevoli ad ogni iterazione. Questo processo è ottenuto valutando la somiglianza tra i singoli

aggiornamenti e il risultato prodotto dall'aggregazione. I partecipanti considerati malevoli vengono quindi bloccati dal sistema di FL per migliorare le iterazioni future. Gli autori di [38] studiano uno scenario in cui i client bizantini alterano arbitrariamente i messaggi contenenti gli aggiornamenti locali (generati da loro stessi) prima di inviarli al server centrale. Propongono quindi un metodo chiamato *Representational Similarity Analysis* (RSA), che è una variante del metodo di discesa stocastica lungo il gradiente, ma resiliente all'attacco bizantino considerato. Gli autori dimostrano che il metodo proposto converge a una soluzione quasi ottimale a con una velocità di convergenza $O(1/k)$.

- *Autoencoder*: Un autoencoder è una classe speciale di rete neurale che riproduce i valori di input nei valori di output [39]. L'interesse principale degli autoencoder è suscitato dagli strati centrali nascosti piuttosto che dallo strato di output. Questo tipo di algoritmo appartiene al mondo dell'apprendimento non supervisionato e non richiede variabili target. Negli autoencoder, il numero di neuroni negli strati nascosti (intermedi) è inferiore rispetto a quello dello strato di input, il che significa che gli strati nascosti estrapolano le informazioni essenziali dai valori di input, apprendendo la maggior parte delle informazioni salienti dei dati ed ignorando i rumori. Le principali applicazioni degli autoencoder sono da individuare nella riduzione della dimensionalità dell'input e nella riduzione del rumore. Rispetto all'approccio convenzionale alla riduzione della dimensionalità, rappresentato dall'analisi delle componenti principali (PCA) [40] che svolge le operazioni utilizzando l'algebra lineare, gli autoencoder eccellono nell'affrontare situazioni caratterizzate

da problemi **non lineari** e **complessi**. In parole semplici, la missione di un autoencoder è codificare i dati in una versione dimensionalmente ridotta (fase di compressione) e quindi decodificarli nuovamente per rigenerare l'input (fase di decompressione). Mentre svolge quest'attività, l'autoencoder apprende le caratteristiche dei dati "normali", li comprime in dimensioni più piccole, ed infine le decodifica nuovamente nell'input commettendo una certa quantità di errore. Pertanto, quando gli vengono mandati in input dati "anomali", un autoencoder non riesce ad eseguire la fase di decompressione correttamente e, di conseguenza, genera un grande errore, dato che è addestrato a riprodurre esclusivamente dati "normali". Pertanto, al fine di utilizzare autoencoder per il rilevamento di anomalie, è sufficiente calcolare l'errore (per lo più il *Mean Square Error*) dell'uscita rispetto a quello dell'ingresso e confrontare questo errore con una soglia predefinita. Ispirati da questa idea, gli autori di [41] propongono un metodo di rilevamento degli attaccanti bizantini che sfrutta un modello di rilevamento delle anomalie basato su un autoencoder pre-addestrato. Più in dettaglio, viene utilizzato un modello di autoencoder pre-addestrato che viene eseguito a livello del CE per riconoscere gli aggiornamenti locali anomali e identificare i relativi mittenti. Di conseguenza, viene assegnato un punteggio di reputazione a ciascun partecipante in base al suo "comportamento". Questo punteggio viene utilizzato per pesare il contributo di ciascun partecipante durante il calcolo del modello globale aggiornato.

2.4.2 Attacchi "Mirati"

A differenza degli attacchi bizantini che mirano a deteriorare le prestazioni complessive del sistema federato, gli attacchi mirati hanno obiettivi malevoli specifici [42], [43] e impiegano tecniche più sofisticate per raggiungerli [44]. Tecnicamente, in questi attacchi si cerca di modificare il comportamento del modello su alcune particolari istanze di dati, mantenendo inalterate le prestazioni sul resto delle istanze. Gli attacchi mirati al FL possono essere classificati in due tipi: attacchi di *data poisoning* e attacchi di *model poisoning*. Gli attacchi di data poisoning mirano a iniettare dati dannosi nel set di dati di addestramento prima dell'inizio del processo di apprendimento. Si ritiene che il processo di apprendimento rimanga valido nel caso in cui si verifichi un attacco di *data poisoning*. Nel FL, poiché un utente malintenzionato ha il controllo solo sui dati del proprio dispositivo, gli attacchi di data poisoning hanno un successo limitato e di conseguenza hanno ricevuto meno attenzione dalla comunità che si occupa di sicurezza.

Dall'altra parte, gli attacchi di *model poisoning* mirano a intaccare il modello di apprendimento stesso invece di compromettere i dati. In particolare, in questo caso gli attaccanti cercano di fare in modo che il modello globale addestrato in modo collaborativo classifichi erroneamente una raccolta di input scelti con elevata sicurezza, garantendo al contempo una buona convergenza sui set di convalida e test, per evitare di essere rilevati. Tecnicamente parlando, tali attaccanti producono i loro aggiornamenti ottimizzandoli per un obiettivo malevolo con lo scopo di produrre una classificazione errata ma mirata, cercando anche di negare l'effetto combinato degli aggiornamenti leciti forniti dai partecipanti benigni.

In letteratura sono state proposte diverse tecniche per contrastare gli attacchi mirati nell'apprendimento federato. Anche in questo caso, queste tecniche possono essere classificate in due categorie principali, ovvero analisi della sicurezza e analisi statistiche. Di seguito, verranno introdotti entrambi gli approcci e saranno discusse le soluzioni che sono state proposte tramite l'adozione di ciascuno di essi.

- Security Analysis: Gli approcci che rientrano in questa categoria cercano di analizzare il comportamento degli attaccanti, discuterne le conseguenze ed esaminare l'efficienza di alcune delle strategie di difesa esistenti nel contrastarli. Ad esempio, gli autori di [45] analizzano un insieme di strategie che consentono di effettuare e contrastare attacchi mirati di model poisoning sul FL. In primo luogo, viene studiata l'esplicita strategia di *boosting* in cui i partecipanti malintenzionati cercano di negare l'effetto combinato dei partecipanti benigni. Successivamente vengono esaminate le nozioni di *stealth*, come le statistiche di aggiornamento dei pesi, da cui gli attaccanti possono trarre vantaggio per migliorare le possibilità di successo degli attacchi. Quindi, viene proposto un problema di minimizzazione alternata che tiene conto sia del model poisoning che delle metriche stealth per consentire agli aggressori di sfuggire al rilevamento.

- Statistics: Come nel caso degli attacchi bizantini, negli attacchi mirati vengono utilizzati metodi statistici per analizzare gli aggiornamenti locali generati dai partecipanti e studiarne la somiglianza/dissomiglianza per rilevare potenziali attacchi. Ad esempio, in [46], gli autori discutono di un attacco di poisoning basato su sybil applicato al FL e propongono

un sistema di difesa chiamato **FoolsGold**. L'attacco considerato può assumere due forme. Il primo è attraverso la sostituzione delle etichette, in cui le etichette degli esempi di addestramento relativi a una particolare classe vengono sostituite mantenendo inalterate le feature dei dati. Il secondo è attraverso il backdoor, in cui singole feature o piccoli insiemi di dati di addestramento vengono manomesse tramite pattern nascosti e di conseguenza rietichettate. L'idea principale di FoolsGold nasce dall'osservazione che un attaccante, insieme ai suoi sybil, che manomette un modello condiviso invierà aggiornamenti verso un singolo obiettivo malevolo specifico, mostrando così un comportamento simile che di solito è diverso da quello dei partecipanti "onesti". Ispirati da questa intuizione, gli autori propongono di adattare l'influenza degli aggiornamenti dei partecipanti nei confronti dell'apprendimento sulla base della somiglianza dei loro contributi. Più specificamente, l'influenza dei gradienti unici viene mantenuta, mentre quella dei partecipanti che inviano costantemente aggiornamenti dall'aspetto simile viene ridotta. Questa soluzione però presenta diversi problemi, come la riduzione della rilevanza nell'apprendimento di tutti quei partecipanti benevoli che mandano update simili premiando, al contrario, i partecipanti malevoli che utilizzano pochi sybil. Inoltre, se per qualche motivo uno dei partecipanti dovesse avere incosapevolmente i sensori rotti, questo andrebbe per una funzione obiettivo tutta sua e, un quanto unica, sarebbe premiato immeritatamente.

Gli autori di [47] sono interessati a rilevare attacchi mirati il cui obiettivo è modificare il comportamento del modello su alcune istanze di

dati. A tal fine, hanno proposto una tecnica di rilevamento delle anomalie spettrali che sfrutta gli incorporamenti a bassa dimensione degli aggiornamenti del modello per identificare ed eliminare quelli dannosi. L'intuizione è che, in uno spazio delle caratteristiche latenti a bassa dimensione, le caratteristiche primarie degli aggiornamenti anormali del modello sarebbero radicalmente diverse da quelle normali.

Capitolo 3

Sistema proposto

Uno degli obiettivi principali di questo lavoro di tesi è presentare un'architettura generale che possa essere adottata nel contesto del *Distributed Human Activity Recognition* (DHAR). Una soluzione semplice potrebbe essere quella di sfruttare la capacità di alimentazione dei dispositivi personali degli utenti, ad esempio smartphone, per effettuare l'elaborazione dei dati e l'affinamento del modello, svolgendo la funzione di FE. Tuttavia, in uno scenario altamente distribuito e multiutente, risulta più efficiente e meno intrusivo [48] utilizzare un singolo nodo con un livello di prestazioni più elevato per svolgere questo compito: un esempio potrebbe essere un personal computer che elabori ed integri i dati da più dispositivi indossati da una comunità di utenti. Si potrebbe anche considerare una situazione diversa in cui, ad esempio, il sistema HAR sfrutti direttamente le informazioni raccolte dagli smartphone degli utenti. In questo caso, questi dispositivi sarebbero logicamente posizionati nello strato inferiore dell'architettura, mentre lo strato intermedio potrebbe essere costituito da altri tipi di unità.

La generalità dell'architettura che viene proposta in questo lavoro permette di utilizzare qualsiasi tipo di dispositivo nello strato intermedio che abbia una potenza di calcolo sufficiente per eseguire l'affinamento del modello di ML ed inviare il proprio contributo al CE. Questa caratteristica rende l'architettura impiegabile in diversi scenari:

- *Gestione automatica delle impostazioni dei dispositivi personali.* In questo scenario, il riconoscimento delle attività umane può essere sfruttato per gestire automaticamente le impostazioni dei dispositivi personali secondo le preferenze dell'utente. Premesso ciò, uno dei *sensing device* potrebbe essere lo smartphone dell'utente che, classificando l'attività attualmente svolta dallo stesso, potrebbe modificare le proprie impostazioni. In questo contesto i FE potrebbero essere i fornitori del servizio che, insieme ad altri provider di servizi basati sullo HAR, potrebbero collaborare per generare un modello di gran lunga più prestante. Un esempio pratico vedrebbe lo smartphone abilitare la modalità silenziosa nel momento in cui rileva che l'utente stia eseguendo attività relative al proprio relax, come mostrato *dormire*, *fare yoga*, e così via.

- *Adattamento della fornitura di servizi per gli Smart Campus* [49]–[51]. In questo secondo possibile scenario, il riconoscimento delle attività umane può essere impiegato per adattare dinamicamente i servizi all'interno di una comunità, come uno *Smart Campus*. In questo contesto, i dispositivi personali degli utenti possono svolgere le funzioni dei *sensing device* mentre quelle dei FE sarebbero svolte dai diversi campus che collaborano per fornire servizi sempre più performanti [52],

[53]. Un esempio pratico potrebbe essere quello della gestione del servizio di navetta all'interno del campus, aumentandone o diminuendone la frequenza in base al livello di congestione di passeggeri alle fermate. Nello specifico, il riconoscimento delle attività, combinato ad altre informazioni di contesto (come la geo-localizzazione), potrebbe consentire di rilevare un elevato livello di congestione alle fermate della navetta; quindi, il gestore del servizio allertato dal sistema può agire tempestivamente facendone circolare di più. In questo modo la coda di attesa viene smaltita e gli utenti subiranno tempi di attesa ridotti.

Gli scenari descritti vedono nel primo caso un servizio fornito al singolo utente, nell'altro un servizio offerto ad una comunità. Questi rappresentano solo due esempi delle tante possibili applicazioni [54] che sfruttano il riconoscimento delle attività umane per fornire un servizio intelligente agli utenti.

Nonostante il funzionamento dei due scenari illustrati sia relativamente simile, la realizzazione dei due progetti richiederebbe la concretizzazione di due architetture ad-hoc diverse tra loro. Le differenti necessità che caratterizzano i due scenari guidano al bisogno della realizzazione di un'architettura modulare, e quanto più generica, che possa essere impiegabile in ogni contesto tramite un semplice lavoro di istanziazione che preveda la definizione delle funzioni dei moduli secondo l'utilità dello scenario in questione.

Uno degli obiettivi di questo elaborato consiste proprio nella sintetizzazione di un'architettura modulare che risolva questa necessità. A seguire, la suddetta architettura verrà descritta nel dettaglio, così come la sua istanziazione nello scenario HAR.

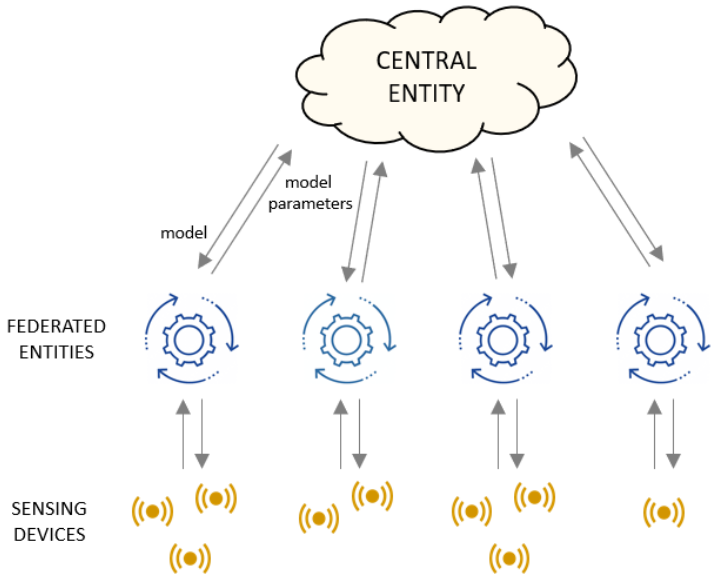

Figura 3.1: Architettura di Federated Learning per il DHAR.

3.1 L'Architettura Modulare

L'architettura modulare proposta in questo elaborato è implementata sulla topologia gerarchica illustrata in Fig 3.1.

Partendo dal livello più basso, i dispositivi di sensing (SDs) distribuiti sono responsabili della raccolta dei dati "grezzi" e dell'esecuzione, qualora fosse richiesto, di semplici operazioni di preprocessing. A questo punto, i dati raccolti vengono condivisi con le entità poste al livello superiore tramite i tradizionali protocolli di rete. Dato che il sistema elabora i dati grezzi nell'ambito della federazione, le informazioni a questo livello non vengono crittografate, in questo modo è possibile soddisfare i vincoli computazionali dei dispositivi intelligenti adoperati, qualora ce ne fossero.

Al livello superiore, i *Federated Entity* (FE, ovvero Entità Federate) sono pro-

gettati per eseguire analisi più approfondite sui dati ottenuti dai dispositivi al livello più basso. In particolare, l'obiettivo principale di un FE è perfezionare il suo modello interno, estrarre gli aggiornamenti del modello locale e condividere queste informazioni con il livello superiore. Altri compiti ugualmente fondamentali svolti dalle FE riguardano la raccolta, la gestione dei nuovi dati (attraverso i quali il modello locale viene aggiornato) e la vera e propria classificazione sfruttata dal sistema per la fornitura dei servizi. Le entità a questo livello possono essere rappresentate da aziende o organizzazioni che vogliono collaborare per il raggiungimento di un obiettivo comune senza, però, alcuna intenzione di condividere i propri dati sensibili con i partner.
Le informazioni prodotte in output dal lavoro di ogni FE vengono inviate al *Central Entity* (CE, ovvero Entità centrale). Il CE è l'attore del sistema responsabile della fase di aggregazione dei parametri, estratti dai modelli locali, provenienti da tutti i device che sono posizionati nello strato sottostante. Il fine dell'operazione di aggregazione è quello realizzare, ad ogni iterazione, l'aggiornamento del modello globale tramite i contributi di ogni partecipante. Il risultato di questo processo è poi rimandato indietro alle FE allo scopo di aggiornare il loro comportamento, rendendo l'intero sistema consistente.
Nella fase di progettazione dell'architettura è stato scelto un approccio modulare al fine di rendere la sua implementazione semplice, chiara, immediata e, soprattutto, più versatile possibile. I moduli principali di cui si compone ogni *federated entity* e che costituiscono il *central entity* sono mostrati dettagliatamente in Figura 3.2.
Il CE rappresenta il cuore dell'intera architettura: questo è dovuto al fatto che il suo ruolo principale è quello di coordinare ogni FE. Da un punto di

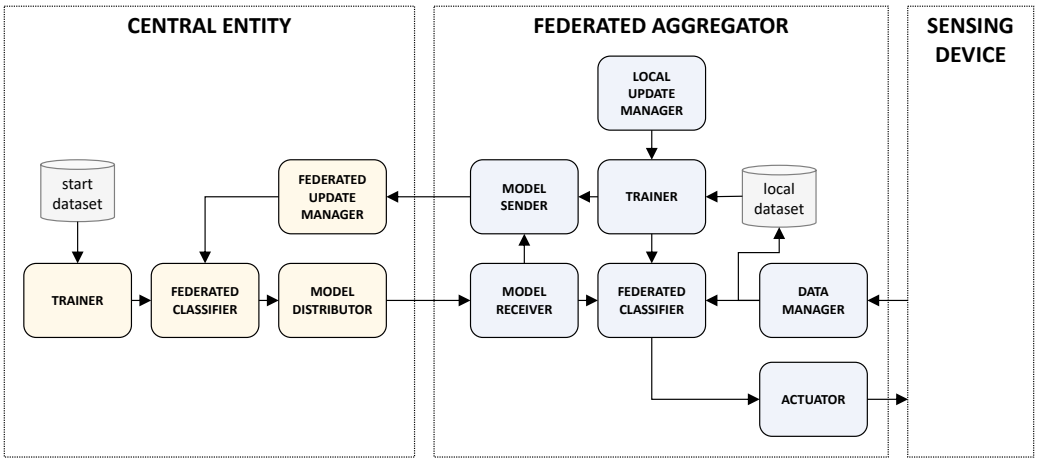

Figura 3.2: Architettura modulare proposta per il Federated Learning.

vista generico, questo attore del sistema è considerabile come il fornitore del servizio ed è responsabile di effettuare una fase di addestramento preliminare su dei dati "di partenza" contenuti all'interno del suo *Start Dataset*. Durante la suddetta fase di addestramento viene messo a punto il ***Federated Classifier*** al fine di soddisfare gli obiettivi specifici dell'applicazione. Infine, i prodotti finali risultanti da questo processo (il modello con i suoi parametri) vengono condivisi con i FE sottostanti attraverso il ***Model Distributor***.

A questo punto, grazie al lavoro svolto dal ***Model Receiver***, ogni FE riceve tutte le informazioni inviate dal CE, estrapolando da queste il proprio ***Federated Classifier***. Il ***Model Receiver*** potrebbe anche inoltrare le informazioni ottenute dal CE al *Model Sender*: questa operazione potrebbe tornare utile in tutte quelle applicazioni che hanno la necessità di calcolare alcuni parametri, ad es., *crossover point* oppure *regret* [55], al fine di valutare la propria partecipazione al Federated Learning.

Il classificatore impiegato nei FE è lo stesso che viene mandato dal CE, ma

è quello contenuto nel FE che, nella pratica, effettua la vera e propria classificazione. Il risultato del processo di classificazione viene mandato come input all'***Actuator***, questo modulo è il responsabile dell'interazione con il mondo esterno. Dipendentemente dal contesto applicativo del sistema, infatti, il lavoro dell'***Actuator*** può essere di natura diversa: in uno scenario di riconoscimento delle attività umane, in cui l'obiettivo è fornire un servizio all'utente finale, questo modulo potrebbe, ad esempio, attivare una funzionalità o selezionare una modalità di funzionamento nel dispositivo di sensing.
Un'altra cosa di primaria importanza da notare, tra le funzioni svolte da ogni FE, è il supporto fornito all'aggiornamento del modello globale localizzato presso il CE. A questo proposito, il ***Data Manager*** svolge un ruolo cruciale: questo, infatti, per prima cosa si occupa di raccogliere i nuovi dati grezzi dal dispositivo di sensing e, in secondo luogo, effettua l'estrazione delle feature da i nuovi dati raccolti. I nuovi dati raccolti ed elaborati sono poi inviati al *Federated Classifier*, per effettuare la classificazione, ed al *Local Dataset* per la loro archiviazione.
Infine, andando oltre i moduli ***Model Sender*** e ***Trainer***, che funzionano in modo simile alle loro controparti nel CE, ogni FE è dotato di un modulo aggiuntivo chiamato ***Local Update Manager***. Questo consiste in un modulo che gestisce l'attivazione del modulo ***Trainer*** e che quindi manovra l'aggiornamento del modello locale. Questo viene attivato ogni qual volta che una determinata condizione viene soddisfatta, ad esempio, potrebbe essere attivato qualora venga raccolta una quantità considerevole di nuovi dati o qualora sia trascorso un certo periodo di tempo.
Quando viene avviata la fase di addestramento federato, viene contestual-

mente attivata in ogni FE la fase miglioramento del modello locale. Ad ogni "round" dell'addestramento federato, ogni FE compie il miglioramento del modello locale ed invia al CE il proprio *update* di round. Gli aggiornamenti locali di ogni FE vengono ricevuti dal CE tramite il modulo federated update manager che si occupa, inoltre di gestirli. Infatti, come già anticipato, il CE ha il ruolo chiave di aggregare i contributi ricevuti dai FE, processo che può essere effettuato tramite diversi algoritmi, ad esempio, il **Federated Average** [56] (*FedAvg*) ed altre sue varianti che saranno approfondite a seguire.

In sintesi, una volta raccolti tutti i contributi locali dei FE, viene eseguito il processo di aggregazione che fornisce in output il modello globale aggiornato che viene distribuito indietro a tutti i FE per un il successivo round di addestramento, ciclicamente.

3.1.1 Istanziazione in HAR

OMISSISS

Capitolo 4

Sicurezza del Sistema

Nella definizione del contesto in cui agirà il sistema risulta essere necessario specificare quale sia lo scenario di attacco preso in considerazione e, di conseguenza, il meccanismo di difesa da attuare per contrastarlo. Per questo motivo, nella stesura di questo capitolo saranno affrontati ed espletati il *Threat Model* e l'algoritmo di difesa di cui sarà dotato il sistema per evitare che l'attacco vada a buon fine.

4.1 Threat Model

OMISSISS

4.2 Algoritmo di difesa

OMISSISS

Capitolo 5

Valutazione sperimentale

La validazione del lavoro svolto nell'ambito della stesura della presente tesi deve necessariamente passare da una fase di valutazione sperimentale. Per questo motivo, in questo capitolo verranno descritte prima di tutto la metriche impiegate nel processo di valutazione, descrivendole sia da un punto di vista matematico che semantico.

Successivamente si passerà alla vera e propria descrizione degli esperimenti svolti, con un piccolo approfondimento sul dataset impiegato. Gli esperimenti principali sono stati svolti con tre obiettivi chiari: il primo mira a validare l'architettura proposta, confrontando il sistema basato sul *Federated Learning* con la sua controparte centralizzata; il secondo ha l'obiettivo di mostrare i risultati dell'attacco effettuato ed i suoi effetti nei confronti del sistema; il terzo ha lo scopo di validare la soluzione proposta nella mitigazione degli effetti causati dall'attacco in corso.

Un ulteriore esperimento verrà infine discusso in cui vengono testati i limiti della soluzione proposta, evidenziando le sue debolezze.

5.1 Metriche di valutazione

OMISSISS

5.2 Valutazione delle prestazioni

OMISSISS

5.2.1 Confronto con il sistema centralizzato

OMISSISS

5.3 Valutazione della sicurezza

OMISSISS

5.3.1 Difesa

OMISSISS

5.3.2 Ulteriori esperimenti

OMISSISS

Capitolo 6

Conclusioni e Sviluppi Futuri

In questo elaborato è stata proposta un'architettura modulare da adottare nella realizzazione di sistemi basati sul Federated Learning. Tra i requisiti fondamentali, che ci si è proposti, dell'architettura in questione assume un ruolo chiave quello della **versatilità**. Il primo obiettivo di questo lavoro, infatti, è stato incentrato sulla realizzazione di un'architettura che possa essere impiegata in ogni scenario che richieda l'utilizzo del paradigma del Federated Learning, con un piccolo lavoro di personalizzazione sulle funzioni dei singoli moduli. Gli esperimenti eseguiti hanno avuto lo scopo di confrontare le prestazioni della soluzione proposta con un approccio centralizzato, dimostrando risultati di gran lunga migliori.

Successivamente l'architettura proposta è stata istanziata in uno scenario di Human Activity Recognition, compito svolto attraverso i dispositivi smart degli utenti. Il processo di HAR è stato basato su un'architettura di FL in cui *Sensing Devices*, *Federated Entity* e un *Central Entity* cooperano, a tre diversi livelli logici, per raccogliere dati sensoriali, eseguire il riconosci-

mento e migliorare continuamente il modello condiviso. Il raggiungimento di questo obiettivo è stato comprovato dallo svolgimento di diversi test, che oltre a testimoniarne la fattibilità ne hanno anche convalidato il successo in termini di performance. Per questo, la nostra soluzione costituisce un buon compromesso tra privacy, prestazioni e versatilità.

Una volta realizzata l'architettura e istanziata nel compito di HAR con successo, il passo successivo è stato quello di attaccare questo sistema. Tra i tutti i possibili è stato scelto di procedere con un attacco sybil di label flipping poiché rappresenta un buon compromesso tra *fattibilità* (intesa come facilità di esecuzione), *efficacia* e *probabilità di incidenza*. Come dimostrato dagli esperimenti svolti, l'attacco è stato portato a termine con successo e i danni riportati al sistema sono stati largamente rilevabili.

A questo punto il passo successivo è stato quello di realizzare un meccanismo di difesa capace di individuare la minaccia in corso ed eliminare dal processo di apprendimento tutti i partecipanti malevoli. Gli esperimenti condotti a concentrazione variabile di dispositivi malevoli sono stati svolti per capire come il sistema reagisse alla più o meno invasiva presenza dell'attaccante. I risultati prodotti dai suddetti esperimenti sono stati a dir poco ottimi: non solo l'algoritmo di difesa è riuscito a discriminare i contributi benigni da quelli maligni, ma in più è risultato evidente che l'invasività dell'attaccante è irrilevante. Infatti, qualunque sia stata la concentrazione di sybil (ammesso che esista almeno un partecipante benigno) l'algoritmo di difesa è stato sempre in grado di individuarli ed eliminarli tutti dal processo di apprendimento. Il raggiungimento di questo obiettivo ottimale e osservabile dai valori riportati nella tabella **??**.

Infine, al di fuori degli obiettivi proposti nella stesura di questo elaborato, sono stati testati i limiti della soluzione proposta andando a fare in modo che il sistema fosse contemporaneamente attaccato da due attaccanti con obiettivi diversi. La sottomissione a questo scenario critico ha esposto il limite di questa soluzione, infatti, l'algoritmo di difesa non è risultato in grado di discriminare tra contributi benigni e maligni, bensì riesce a categorizzare i contributi per obiettivo. Per questo motivo i due cluster definiti nell'algoritmo di difesa sono risultati insufficienti alla protezione del sistema in questo caso.

Quest'ultimo esperimento condotto ha aperto le porte alle idee per i possibili sviluppi futuri dell'algoritmo di difesa: infatti, appurato che il suo punto debole sia il numero di cluster scelti in cui suddividere i contributi è sufficiente che questo aumenti per essere in grado di difendersi da un qualsiasi numero di attaccanti ognuno con i propri obiettivi.

Oltre a questo approccio, si potrebbero valutare tecniche molto complesse, allo stato dell'arte, in grado di rendere il sistema robusto ad a eventuali modifiche [57].

Elenco delle figure

Elenco delle tabelle

Elenco degli Algoritmi

Bibliografia

[1] D. Peteiro-Barral e B. Guijarro-Berdiñas, «A survey of methods for distributed machine learning,» *Progress in Artificial Intelligence*, vol. 2, n. 1, pp. 1–11, 2013, ISSN: 2192-6360. DOI: 10.1007/s13748-012-0035-5.

[2] Y. Chen, X. Qin, J. Wang, C. Yu e W. Gao, «FedHealth: A Federated Transfer Learning Framework for Wearable Healthcare,» *IEEE Intelligent Systems*, vol. 35, n. 4, pp. 83–93, 2020. DOI: 10.1109/MIS.2020.2988604.

[3] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh e D. Bacon, «Federated learning: Strategies for improving communication efficiency,» *arXiv preprint arXiv:1610.05492*, 2016.

[4] I. Kholod, E. Yanaki, D. Fomichev et al., «Open-source federated learning frameworks for IoT: A comparative review and analysis,» *Sensors*, vol. 21, n. 1, p. 167, 2020.

[5] O. A. Wahab, A. Mourad, H. Otrok e T. Taleb, «Federated machine learning: Survey, multi-level classification, desirable criteria and future directions in communication and networking systems,» *IEEE*

Communications Surveys & Tutorials, vol. 23, n. 2, pp. 1342–1397, 2021.

[6] F. Concone, C. Ferdico, G. Lo Re e M. Morana, «A Federated Learning Approach for Distributed Human Activity Recognition,» in *2022 IEEE International Conference on Smart Computing (SMARTCOMP)*, 2022, pp. 269–274.

[7] B. McMahan, E. Moore, D. Ramage, S. Hampson e B. A. y. Arcas, «Communication-Efficient Learning of Deep Networks from Decentralized Data,» in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, A. Singh e J. Zhu, cur., ser. Proceedings of Machine Learning Research, vol. 54, PMLR, apr. 2017, pp. 1273–1282.

[8] J. Liu, J. Huang, Y. Zhou et al., «From distributed machine learning to federated learning: a survey,» *Knowledge and Information Systems*, vol. 64, n. 4, pp. 885–917, 1 apr. 2022, ISSN: 0219-3116. DOI: 10.1007/s10115-022-01664-x.

[9] *REGULATION (EU) 2016/679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL*, European Commission, 6 nov. 2016.

[10] S. A., «Google is fined $57 million under Europe's data privacy law.,» *New York Times*, 28 feb. 2021.

[11] H. Zhu, H. Zhang e Y. Jin, «From federated learning to federated neural architecture search: a survey,» *Complex & Intelligent Systems*, vol. 7, n. 2, pp. 639–657, 1 apr. 2021, ISSN: 2198-6053. DOI: 10.1007/s40747-020-00247-z.

[12] Q. Yang, Y. Liu, T. Chen e Y. Tong, «Federated Machine Learning: Concept and Applications,» vol. 10, n. 2, gen. 2019, ISSN: 2157-6904. DOI: 10.1145/3298981.

[13] D. Gao, C. Ju, X. Wei, Y. Liu, T. Chen e Q. Yang, *HHHFL: Hierarchical Heterogeneous Horizontal Federated Learning for Electroencephalography*, 2019. DOI: https://doi.org/10.48550/arxiv.1909.05784.

[14] S. Lee, M. E. Lacy, M. Jankowich, A. Correa e W.-C. Wu, «Association between obesity phenotypes of insulin resistance and risk of type 2 diabetes in African Americans: The Jackson Heart Study,» *Journal of Clinical & Translational Endocrinology*, vol. 19, p. 100 210, 2020, ISSN: 2214-6237. DOI: https://doi.org/10.1016/j.jcte.2019.100210.

[15] Y. Liu, Y. Kang, C. Xing, T. Chen e Q. Yang, «A Secure Federated Transfer Learning Framework,» *IEEE Intelligent Systems*, vol. 35, n. 4, pp. 70–82, 2020. DOI: 10.1109/MIS.2020.2988525.

[16] X. Li, K. Huang, W. Yang, S. Wang e Z. Zhang, «On the convergence of fedavg on non-iid data,» *arXiv preprint arXiv:1907.02189*, 2019.

[17] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar e V. Smith, «Federated optimization in heterogeneous networks,» *Proceedings of Machine Learning and Systems*, vol. 2, pp. 429–450, 2020.

[18] A. Reisizadeh, A. Mokhtari, H. Hassani, A. Jadbabaie e R. Pedarsani, «Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization,» in *International Conference on Artificial Intelligence and Statistics*, PMLR, 2020, pp. 2021–2031.

[19] J. So, B. Güler e A. S. Avestimehr, «Turbo-aggregate: Breaking the quadratic aggregation barrier in secure federated learning,» *IEEE Journal on Selected Areas in Information Theory*, vol. 2, n. 1, pp. 479–489, 2021.

[20] H. Wang, M. Yurochkin, Y. Sun, D. Papailiopoulos e Y. Khazaeni, «Federated learning with matched averaging,» *arXiv preprint arXiv:2002.06440*, 2020.

[21] L. Liu, J. Zhang, S. Song e K. B. Letaief, «Client-edge-cloud hierarchical federated learning,» in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, IEEE, 2020, pp. 1–6.

[22] N. Gupta, S. K. Gupta, R. K. Pathak, V. Jain, P. Rashidi e J. S. Suri, «Human activity recognition in artificial intelligence framework: a narrative review,» *Artificial Intelligence Review*, gen. 2022, ISSN: 1573-7462. DOI: 10.1007/s10462-021-10116-x.

[23] S. Gaglio, G. Lo Re e M. Morana, «Human Activity Recognition Process Using 3-D Posture Data,» *IEEE Transactions on Human-Machine Systems*, vol. 45, n. 5, pp. 586–597, 2015. DOI: 10.1109/THMS.2014.2377111.

[24] M. A. Khan, K. Javed, S. A. Khan et al., «Human action recognition using fusion of multiview and deep features: an application to video surveillance,» *Multimedia Tools and Applications*, mar. 2020, ISSN: 1573-7721. DOI: 10.1007/s11042-020-08806-9.

[25] S. Zhang, Z. Wei, J. Nie, L. Huang, S. Wang e Z. Li, «A Review on Human Activity Recognition Using Vision-Based Method,» *Journal of*

Healthcare Engineering, vol. 2017, p. 3 090 343, lug. 2017, ISSN: 2040-2295. DOI: 10.1155/2017/3090343.

[26] F. Concone, S. Gaglio, G. Lo Re e M. Morana, «Smartphone Data Analysis for Human Activity Recognition,» in *AI*IA 2017 Advances in Artificial Intelligence*, F. Esposito, R. Basili, S. Ferilli e F. A. Lisi, cur., Cham: Springer International Publishing, 2017, pp. 58–71, ISBN: 978-3-319-70169-1.

[27] F. Concone, G. Lo Re e M. Morana, «A Fog-Based Application for Human Activity Recognition Using Personal Smart Devices,» *ACM Trans. Internet Technol.*, vol. 19, n. 2, mar. 2019, ISSN: 1533-5399. DOI: 10.1145/3266142.

[28] F. Concone, G. Lo Re e M. Morana, «SMCP: a Secure Mobile Crowdsensing Protocol for fog-based applications,» *Human-centric Computing and Information Sciences*, vol. 10, n. 1, p. 28, lug. 2020, ISSN: 2192-1962. DOI: 10.1186/s13673-020-00232-y.

[29] A. Abdallah, M. A. Maarof e A. Zainal, «Fraud detection system: A survey,» *Journal of Network and Computer Applications*, vol. 68, pp. 90–113, 2016, ISSN: 1084-8045. DOI: https://doi.org/10.1016/j.jnca.2016.04.007.

[30] V. Agate, F. Concone e P. Ferraro, «A Resilient Smart Architecture for Road Surface Condition Monitoring,» in *Innovations in Smart Cities Applications Volume 5*, M. Ben Ahmed, A. A. Boudhir, İ. R. Karaș, V. Jain e S. Mellouli, cur., Cham: Springer International Publishing, 2022, pp. 199–209, ISBN: 978-3-030-94191-8.

[31] J. Xu, B. S. Glicksberg, C. Su, P. Walker, J. Bian e F. Wang, «Federated Learning for Healthcare Informatics,» *Journal of Healthcare Informatics Research*, vol. 5, n. 1, pp. 1–19, mar. 2021, ISSN: 2509-498X. DOI: 10.1007/s41666-020-00082-4.

[32] K. Sozinov, V. Vlassov e S. Girdzijauskas, «Human Activity Recognition Using Federated Learning,» in *2018 IEEE Intl Conf on Parallel Distributed Processing with Applications, Ubiquitous Computing Communications, Big Data Cloud Computing, Social Computing Networking, Sustainable Computing Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*, 2018, pp. 1103–1111. DOI: 10.1109/BDCloud.2018.00164.

[33] S. Ek, F. Portet, P. Lalanda e G. Vega, «Evaluation of Federated Learning Aggregation Algorithms: Application to Human Activity Recognition,» in *Adjunct Proceedings of the 2020 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2020 ACM International Symposium on Wearable Computers*, ser. UbiComp-ISWC '20, Virtual Event, Mexico: Association for Computing Machinery, 2020, 638–643, ISBN: 9781450380768. DOI: 10.1145/3410530.3414321.

[34] S. A. Rahman, H. Tout, C. Talhi e A. Mourad, «Internet of things intrusion detection: Centralized, on-device, or federated learning?» *IEEE Network*, vol. 34, n. 6, pp. 310–317, 2020.

[35] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin e V. Shmatikov, «How to backdoor federated learning,» in *International Conference on Artificial Intelligence and Statistics*, PMLR, 2020, pp. 2938–2948.

[36] P. Blanchard, E. M. El Mhamdi, R. Guerraoui e J. Stainer, «Machine learning with adversaries: Byzantine tolerant gradient descent,» *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[37] L. Muñoz-González, K. T. Co e E. C. Lupu, «Byzantine-robust federated machine learning through adaptive model averaging,» *arXiv preprint arXiv:1909.05125*, 2019.

[38] L. Li, W. Xu, T. Chen, G. B. Giannakis e Q. Ling, «RSA: Byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets,» in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 1544–1551.

[39] A. De Paola, S. Favaloro, S. Gaglio, G. Lo Re e M. Morana, «Malware Detection through Low-level Features and Stacked Denoising Autoencoders..»

[40] T. Yu, X. Wang e A. Shami, «Recursive principal component analysis-based data outlier detection and sensor data aggregation in IoT systems,» *IEEE Internet of Things Journal*, vol. 4, n. 6, pp. 2207–2216, 2017.

[41] S. Li, Y. Cheng, Y. Liu, W. Wang e T. Chen, «Abnormal client behavior detection in federated learning,» *arXiv preprint arXiv:1910.09933*, 2019.

[42] F. Concone, G. Lo Re, M. Morana e C. Ruocco, «Assisted Labeling for Spam Account Detection on Twitter,» in *2019 IEEE International Conference on Smart Computing (SMARTCOMP)*, 2019, pp. 359–366. DOI: 10.1109/SMARTCOMP.2019.00073.

[43] F. Concone, G. Lo Re, M. Morana e C. Ruocco, «Twitter Spam Account Detection by Effective Labeling.,» in *ITASEC*, 2019.

[44] F. Concone, F. De Vita, A. Pratap, D. Bruneo, G. Lo Re e S. K. Das, «A Novel Recruitment Policy to Defend against Sybils in Vehicular Crowdsourcing,» in *2021 IEEE International Conference on Smart Computing (SMARTCOMP)*, 2021, pp. 105–112. DOI: 10.1109/SMARTCOMP52413.2021.00035.

[45] A. N. Bhagoji, S. Chakraborty, P. Mittal e S. Calo, «Analyzing federated learning through an adversarial lens,» in *International Conference on Machine Learning*, PMLR, 2019, pp. 634–643.

[46] C. Fung, C. J. Yoon e I. Beschastnikh, «Mitigating sybils in federated learning poisoning,» *arXiv preprint arXiv:1808.04866*, 2018.

[47] S. Li, Y. Cheng, W. Wang, Y. Liu e T. Chen, «Learning to detect malicious clients for robust federated learning,» *arXiv preprint arXiv:2002.00211*, 2020.

[48] P. Cottone, G. Lo Re, G. Maida e M. Morana, «Motion sensors for activity recognition in an ambient-intelligence scenario,» in *2013 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, 2013, pp. 646–651. DOI: 10.1109/PerComW.2013.6529573.

[49] V. Agate, F. Concone e P. Ferraro, «WiP: Smart Services for an Augmented Campus,» in *2018 IEEE International Conference on Smart Computing (SMARTCOMP)*, 2018, pp. 276–278. DOI: 10.1109/SMARTCOMP.2018.00056.

[50] F. Concone, P. Ferraro e G. Lo Re, «Towards a Smart Campus Through Participatory Sensing,» in *2018 IEEE International Conference on Smart Computing (SMARTCOMP)*, 2018, pp. 393–398. DOI: 10.1109/SMARTCOMP.2018.00035.

[51] S. Gaglio, G. Lo Re, M. Morana e C. Ruocco, «Smart Assistance for Students and People Living in a Campus,» in *2019 IEEE International Conference on Smart Computing (SMARTCOMP)*, 2019, pp. 132–137. DOI: 10.1109/SMARTCOMP.2019.00042.

[52] A. De Paola, P. Ferraro, G. Lo Re, M. Morana e M. Ortolani, «A fog-based hybrid intelligent system for energy saving in smart buildings,» *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, n. 7, pp. 2793–2807, 2020.

[53] A. De Paola, G. Lo Re, M. Morana e M. Ortolani, «SmartBuildings: an AmI system for energy efficiency,» in *2015 Sustainable Internet and ICT for Sustainability (SustainIT)*, 2015, pp. 1–7. DOI: 10.1109/SustainIT.2015.7101372.

[54] A. De Paola, S. Gaglio, A. Giammanco, G. Lo Re e M. Morana, «A multi-agent system for itinerary suggestion in smart environments,» *CAAI Transactions on Intelligence Technology*, vol. 6, n. 4, pp. 377–393, 2021. DOI: https://doi.org/10.1049/cit2.12056. eprint:

https://ietresearch.onlinelibrary.wiley.com/doi/pdf/10.1049/cit2.12056.

[55] S. Zehtabian, S. Khodadadeh, L. Bölöni e D. Turgut, «Privacy-Preserving Learning of Human Activity Predictors in Smart Environments,» in *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, 2021, pp. 1–10. DOI: 10.1109/INFOCOM42981.2021.9488681.

[56] H. B. McMahan, E. Moore, D. Ramage, S. Hampson e B. A. y. Arcas, «Communication-Efficient Learning of Deep Networks from Decentralized Data,» 2016. DOI: 10.48550/ARXIV.1602.05629.

[57] V. Agate, A. De Paola, G. Lo Re e M. Morana, «A Simulation Software for the Evaluation of Vulnerabilities in Reputation Management Systems,» *ACM Trans. Comput. Syst.*, vol. 37, n. 1–4, 2021, ISSN: 0734-2071. DOI: 10.1145/3458510.