

Spunti di tesi e stage

Alcuni di questi spunti si basano sugli Spiking Neural P Systems (SNPS), un modello introdotto nel paper [14], che contiene una panoramica generale del suo funzionamento; mentre altri riguardano le Spiking Neural Networks (SNN), e in [1] è presente un review recente. Per una descrizione più chiara e concisa, che comprende SNPS, SNN e alcune delle possibili applicazioni, vedere [37].

Al momento è in corso lo sviluppo di un simulatore in grado di eseguire SNPS con anti-spike e white hole (vedi [30]), in questo **link**, alcuni spunti trattano questo codice.

Per un approccio pratico a come lavorare su una rete SNN che analizzi le immagini, si trova un tutorial esaustivo **qui**

Spunti differenti possono condividere gli stessi paper, essendo collegati da tematiche comuni.

1 Federated Learning

Il lavoro in [33] fornisce una spiegazione approfondita del funzionamento del Federated Learning (FL) applicato alle SNN, con esempi pratici, codice e numerosi risultati sperimentali. Il tema è di grande interesse poiché unisce due ambiti emergenti: l'apprendimento distribuito e i modelli neurali ispirati al funzionamento biologico.

Nel lavoro [32], pur non focalizzato sulle SNN, vengono discusse diverse direzioni di ricerca nel campo del green FL, riassunte nella figura 1. Tecniche come la sparsification consentono di ottenere grafi con un numero di archi significativamente inferiore rispetto a quello massimo possibile, risultando quindi applicabili sia alle SNN sia agli SNPS. Nel caso delle SNN, tuttavia, le sinapsi possiedono pesi reali da trasmettere, mentre negli SNPS si tratta di connessioni booleane (presenza o assenza di sinapsi), rendendo il problema di compressione concettualmente diverso.

Ulteriori tecniche, come pruning e quantization, possono essere esplorate per ridurre la complessità dei modelli, e sono tecniche riconducibili al graph compression. Una possibile direzione di tesi consiste quindi nel confrontare le principali tecniche di efficientamento del Federated Learning applicate alle ANN, SNN e SNPS, valutandone i benefici in termini di risparmio computazionale e spaziale, nonché la loro adattabilità ai modelli spiking.

Si può trovare a questo **link** una bozza di SNPS, poi evoluto in **questo**, specificatamente strutturato per essere privacy preserving (ottimo per il FL) e

collegato al lavoro [25], ma è incompleto e non formale, segno che non ci sono molti lavori in questa direzione.

Un'altra direzione di ricerca riguarda la modifica degli algoritmi di apprendimento distribuito tra client e server per le SNN. Tali reti richiedono metodi specifici, come la Backpropagation Through Time (BPTT) o la Batch Normalization Through Time (BNTT), che differiscono profondamente dagli algoritmi utilizzati nelle ANN. Questi metodi devono inoltre essere adattati a scenari realistici in cui:

- i dati non sono indipendenti e identicamente distribuiti (non IID) e le classi risultano sbilanciate;
- le risorse energetiche e computazionali dei dispositivi sono limitate;
- è necessario ridurre i tempi o i costi di addestramento in ambienti distribuiti.

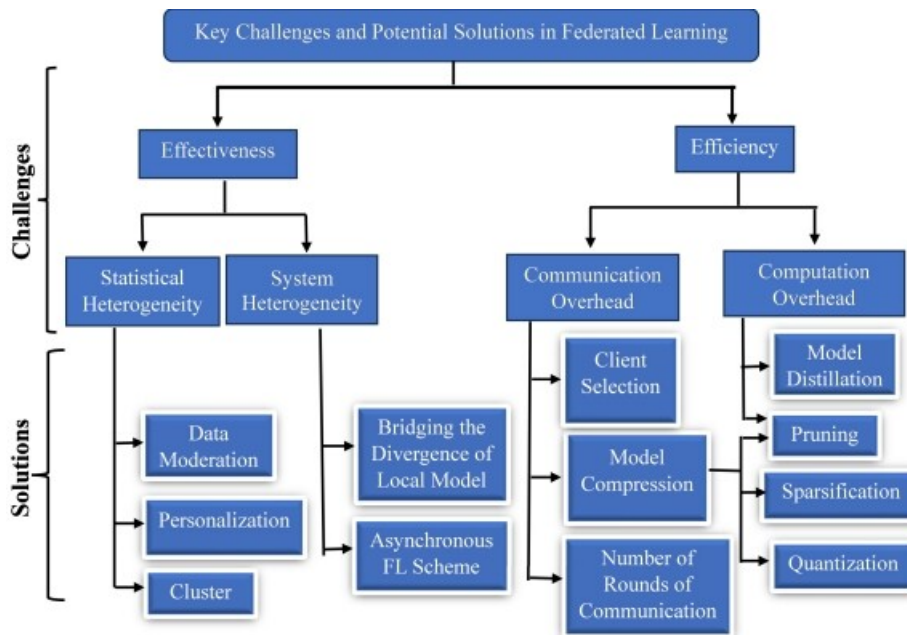


Figure 1: Sfide e possibili soluzioni nel Federated Learning verso un approccio più sostenibile [32].

Esempio di ricerca

- Trovare, nel paper [32] o in altri simili, un ambito dove il FL e le SNN comportano vantaggi. Per esempio nei wearable devices, dove le informazioni devono restare private e il consumo energetico ha grossi vincoli;

- Cercare paper più specifici a riguardo, come [29], e studiare il database utilizzato;
- Implementare un modello equivalente o simile, studiarlo, e intraprendere le direzioni di ricerca proposte dal paper scelto.

2 Tuning della SNPS con algoritmi euristici

Attualmente nel **simulatore** citato è stata realizzata una semplice rete composta da tre strati, ma la sua struttura è completamente modificabile: è possibile variare il numero di strati e di neuroni inclusi, la disposizione delle sinapsi e le regole interne dei neuroni.

L'obiettivo è raggiungere prestazioni più elevate attraverso il tuning dei parametri del sistema.

È inoltre possibile intervenire sulla connettività sinaptica, eliminando o inibendo determinate connessioni per allenare e migliorare il modello. A differenza delle reti neurali artificiali classiche, dove l'ottimizzazione agisce sui pesi sinaptici, negli SNPS sono richiesti meccanismi diversi, e sono attualmente in fase di esplorazione nuove strategie per gestire questa dinamica.

In questo contesto entra in gioco l'uso di algoritmi euristici di ottimizzazione. Sarà quindi necessaria una fase di tuning articolata in tre sottofasi principali.

- Tuning struttura: Il numero di layer e di neuroni per ciascun livello è modificabile, così come la funzione svolta da ogni layer (max/average pooling, convolution, edge detection, fully connected). Il problema è complesso perché si tratta di neuroni con regole di attivazione adattabili, ed è possibile costruire diverse strutture su cui operare. Questo processo può essere affrontato, ad esempio, tramite Tabu Search o mediante una neighborhood function che definisce piccole modifiche alla topologia della rete, come l'aggiunta o la rimozione di neuroni o sinapsi.
- Tuning regole: Il tuning delle regole può essere gestito con Simulated Annealing o tecniche affini per individuare la combinazione ottimale dei parametri. Anche in questo caso lo spazio di ricerca è estremamente ampio: ogni neurone può possedere più regole, e ciascuna regola presenta diversi valori da ottimizzare. Il problema risulta complesso poiché è presente un elevato numero di variabili da considerare.
- Tuning sinapsi: Per descrivere questa fase si immagina un'area di ricerca triangolare, dove a ogni vertice è associata la percentuale di sinapsi inibite (funzionamento negativo), cancellate e mantenute, che somma a 1. Ogni combinazione produce differenti livelli di accuratezza, e l'esplorazione di questo spazio può essere condotta con Particle Swarm Optimization (PSO) o simili.

Il modello è nuovo e ancora poco studiato, il che comporta alcune difficoltà, ma, allo stesso tempo, offre numerose opportunità di miglioramento, apprendi-

mento e ricerca innovativa attraverso l'impiego di tecniche di ottimizzazione non convenzionali.

Vedi [24] per un riassunto generico sulle tecniche di learning in questo modello, e [8] per un survey sulle tecniche di image processing nel membrane computing (include tissue-like oltre che Spiking Neural).

Il lavoro in [41], successivamente esteso in [43], propone un modello in cui più Extended SNPS operano in parallelo, con probabilità di applicazione delle regole modificabili, dando origine a un Optimization SNPS capace di risolvere il Knapsack Problem. Si tratta di un esempio significativo di come i SNPS possano essere adattati e ottimizzati per affrontare un problema teorico specifico, ma con una fase di tuning concreta e ben definita.

Esempio di ricerca

- Analizzare il paper appena citato [41] e la sua estensione [43];
- Ricreare il codice presentato astraendo dal modello a membrane, ovvero simulandolo con matrici o tecniche equivalenti;
- Trovare una direzione di ricerca a esso collegata. Per esempio, testare su un knapsack problem con una funzione di profitto più complessa; o su un diverso problema in NP, oppure differenziare le fasi di exploration e exploitation;
- Adattare il codice, ottenere dei risultati e confrontarli con il lavoro originale in aspetti quali performance, struttura, realismo, complessità del modello.

3 Minimizzazione dei costi energetici

Dopo una fase iniziale di stima accurata dei costi computazionali associati all'esecuzione del sistema, si procederebbe con la definizione della funzione obiettivo e con lo studio di meccanismi volti a ridurla. La struttura della rete può essere modellata come un grafo orientato, in cui i nodi rappresentano i neuroni e gli archi rappresentano le sinapsi. A ciascun arco è associato un peso fisso, corrispondente al costo energetico necessario per la trasmissione di uno spike, mentre a ciascun nodo è attribuito un peso che rappresenta l'energia richiesta per l'applicazione delle regole di firing (potenzialmente maggiore in presenza di più regole). Queste operazioni possono includere il controllo di espressioni regolari, conteggi o manipolazioni di stati interni: è quindi necessario distinguere tra costo computazionale e costo energetico effettivo (in joule), così da poter passare successivamente dal primo al secondo.

L'obiettivo è minimizzare il costo complessivo della rete mantenendo la correttezza funzionale, valutata tramite metriche quali l'accuracy o altre misure di performance comparabili. La procedura di ottimizzazione potrà essere

affrontata mediante tecniche di local search o metaeuristiche su grafo, confrontando le diverse strutture ottenute in termini di costi, benefici e limitazioni.

Una criticità di questa direzione di ricerca è rappresentata dalla mancanza di dispositivi hardware che consentano la realizzazione e l'utilizzo diretto di SNPS, rendendo complessa la stima precisa e affidabile dei costi energetici. Un possibile approccio consiste nell'adattare tali modelli ai chip neuromorfici sviluppati per SNN, come Loihi 2, o in parte minore con SpiNNaker o TrueNorth. Ulteriori informazioni sui lavori pubblicati con il chip Loihi 2 sono disponibili **qui**.

Applicato alle SNN, questo spunto di tesi risulta più esplorato, poiché è ampiamente documentata la riduzione del costo energetico delle SNN rispetto alle ANN. Il lavoro presentato in [31] utilizza un quantization-aware training in una SNN per la classificazione di immagini e ne confronta il consumo energetico con quello di un training classico, illustrando come sia possibile analizzare i consumi di una SNN.

Esempio di ricerca

- Approfondire la conoscenza dei costi energetici delle SNN attraverso lo studio dei principali lavori scientifici che ne descrivono le caratteristiche;
- Analizzare quali layer, funzioni di attivazione e meccanismi delle reti neurali presentano consumi energetici maggiori o minori;
- Individuare una pubblicazione che proponga una rete di complessità medio-bassa e riprodurre l'implementazione secondo quanto descritto dagli autori;
- Ridurre i costi energetici della rete e confrontare i nuovi risultati con quelli precedenti, mirando a minimizzare il degrado delle prestazioni e a massimizzare il beneficio energetico;
- Applicare i metodi adottati a reti analoghe, analizzandone la risposta a modifiche di natura simile.

4 Equivalenze tra modelli

Una possibile direzione di ricerca consiste nello sviluppo di un software in grado di tradurre un modello di SNPS in altri formalismi computazionali. Per esempio in [19] si introducono le Spiking Petri Nets, un algoritmo per passare da queste agli SNPS, e un algoritmo per fare l'opposto. In [3], anche, si traccia un corrispettivo tra i due modelli, e questi sono solo alcuni esempi. In seguito ad un'analisi teorica preliminare, si possono individuare relazioni di equivalenza (o forme meno strette di corrispondenza) tra gli SNPS e modelli a eventi discreti, automi a stati finiti o temporizzati, circuiti logici e altri.

Successivamente si potrebbe ragionare su un traduttore che faccia da ponte tra i modelli scelti. Questa avrebbe un duplice valore: da un lato, contribuirebbe a una comprensione più profonda delle proprietà formali degli SNPS (nonostante

siano già state studiate a fondo), dall'altro permetterebbe di connetterli a modelli con maggiore diffusione e applicabilità pratica. In questo modo, gli SNPS potrebbero diventare un linguaggio di modellazione intermedio, da cui derivare rappresentazioni più facilmente realizzabili o simulabili in ambito hardware o software.

Un riferimento interessante in questa direzione è rappresentato dal lavoro [5], nel quale, pur trattandosi di tissue-like P systems e non di SNPS, è stato utilizzato il linguaggio VHDL per implementare un P system su FPGA, dimostrando la possibilità di una trasposizione del modello in hardware fisico.

Esempio di ricerca

- Individuare un modello teorico affine agli SNPS, eventualmente già correlato ad essi, come nel caso delle Virus Machines descritte in [28];
- Studiare il modello individuato e, ove possibile, stabilire un parallelismo diretto delle componenti che lo compongono con il modello SNPS;
- Analizzare varianti degli SNPS che consentano di colmare eventuali lacune, al fine di applicare la teoria sviluppata finora a tali varianti anche al modello considerato;
- Implementare un codice che consenta la conversione da un modello all'altro nel modo più efficiente possibile, minimizzando il numero di neuroni, regole e sinapsi nello SNPS.

5 Evoluzione dei simulatori

Il simulatore WebSnapse [11], utilizzabile in questo **link**, consente la creazione e l'esecuzione di piccoli SNPS tramite un'interfaccia web intuitiva, risultando particolarmente utile per attività didattiche e di divulgazione. Una possibile direzione di lavoro, più legata ad uno stage, consiste nello sviluppo di un'estensione del simulatore utilizzando il codice fornito in **questo** repository GitHub.

Una possibilità di miglioramento del simulatore consiste nell'introdurre meccanismi di evoluzione delle regole con l'obiettivo di esplorare dinamiche di ottimizzazione dei P system. Si propone quindi l'implementazione di un framework ispirato agli algoritmi genetici, che consenta l'evoluzione automatica di regole e configurazioni di P system, con metriche di fitness legate alla correttezza o all'efficienza del sistema. Si guardi il codice presente in **questo** repository, che propone un algoritmo genetico applicato ai SNPS.

Esempio di ricerca

- Provare il simulatore WebSnapse e analizzarne il codice;
- Informarsi sull'uso dei Genetic Algorithms nel campo dei P systems. Per esempio, in questo paper [6] si utilizza un GA per ridurre la grandezza dei

SNPS, mentre in questo lavoro [9] si utilizza per generare possibili SNPS che soddisfino un determinato compito;

- Scegliere un approccio di ricerca simile e testare vari tipi di algoritmi genetici per migliorare gli SNPS sotto vari aspetti quali le performance, il numero di neuroni e di regole presenti.

6 Ispirazione biologica

Il cervello umano rappresenta un sistema di elaborazione estremamente efficiente, in grado di eseguire in parallelo un'enorme quantità di operazioni con un consumo energetico molto ridotto rispetto ai sistemi computazionali tradizionali. Pur non essendo ottimale in termini di velocità o precisione numerica, eccelle in compiti come il riconoscimento visivo, l'elaborazione linguistica e l'apprendimento associativo, grazie alla natura fortemente distribuita e adattiva delle sue reti neurali. Un esempio è la rapidità con cui il sistema visivo umano riesce a estrarre informazioni spaziali e semantiche da una scena complessa, identificando oggetti, relazioni e contesto con una latenza di pochi millisecondi.

Una possibile direzione di ricerca consiste nel modellare, all'interno del framework degli SNPS o delle SNN, specifici meccanismi biologici come quelli della retina, della corteccia visiva o di altre aree sensoriali. L'obiettivo non è la riproduzione completa di un cervello funzionante, ma la simulazione di processi locali ben compresi, ad esempio il comportamento delle cellule gangliari retiniche o i meccanismi di competizione neuronale che portano al riconoscimento dei bordi [18].

Tali studi richiedono una comprensione, anche basilare, dei principi neurofisiologici coinvolti e la disponibilità ad approfondire la letteratura neuroscientifica. Un lavoro esaustivo che comprende SNN e ispirazione al cervello è [39].

Esempio di ricerca

- Approfondire in dettaglio il funzionamento di un meccanismo visivo umano specifico;
- Simularne il comportamento con un modello spiking;
- Analizzarne l'efficacia e i risultati ottenuti confrontandoli con l'effettivo output della retina.

7 Computazione parallela, discreta e continua

Per aumentare le performance del modello e per avvicinarsi al funzionamento effettivo dei P system, l'uso della GPU è fondamentale. Ogni membrana presente nel P system, infatti, computa in modo indipendente e parallelo, portando vantaggi in termini di efficienza e di fedeltà rispetto al comportamento teorico. Il primo passo sarebbe quindi l'adattamento del simulatore per un approccio

parallelo, sfruttando framework come CUDA, OpenCL o PyTorch per la gestione simultanea delle membrane. Questa prima fase può configurarsi come un possibile progetto di stage, incentrato sulla parte implementativa e di ottimizzazione.

Un'ulteriore direzione di approfondimento riguarda la computazione dipendente dal tempo negli SNN. Nei modelli teorici di Spiking Neural Network il tempo è continuo, e gli spike avvengono in istanti reali; tuttavia, nella pratica di simulazione su hardware digitale, il tempo viene discretizzato in piccoli intervalli (Δt), aggiornando i potenziali di membrana e le sinapsi a ogni step. Questa discretizzazione consente di eseguire la rete in modo sincrono e parallelo, approssimando il comportamento continuo e permettendo l'uso efficiente delle GPU.

Nei modelli SNPS, invece, il tempo è completamente discreto e sincrono: le regole vengono applicate simultaneamente a tutti i neuroni, scandendo una sequenza ben definita di step computazionali. Confrontare questi approcci permette di analizzare due differenti concezioni di tempo computazionale — una continua e biologicamente ispirata, l'altra discreta e formale — valutando vantaggi, limiti e possibilità di integrazione. Più informazioni a riguardo, con un confronto diretto, si possono trovare nel lavoro [24], mentre in questo **GitHub** è presente un punto di partenza nel codice relativo agli SNPS.

Esempio di ricerca

Una possibile tesi potrebbe quindi concentrarsi sull'analisi dell'implementazione parallela della computazione temporale in entrambi i modelli spiking. Gli obiettivi principali includerebbero:

- Implementare un'architettura di SNPS parallela, eseguibile su GPU, con computazione indipendente per membrana. Vedere il lavoro [38] per una definizione con matrici degli SNPS;
- Studiare le differenze nella rappresentazione del tempo tra SNN (tempo continuo) e SNPS (tempo discreto e sincrono), analizzandone l'impatto su performance e correttezza;
- Esplorare modelli ibridi, vedi per esempio i Layered SNPS [40, 21], nei quali vengono introdotti pesi o attivazioni fuzzy, confrontando i vantaggi portati da queste fusioni rispetto ai modelli originali;
- Proporre un framework di confronto che permetta di passare tra diversi paradigmi di computazione spiking, evidenziandone le proprietà emergenti e i possibili vantaggi in termini di parallelismo e modellazione temporale.

8 Differenti modelli di SNPS

Nel simulatore sviluppato è stata implementata unicamente l'estensione basata sugli anti-spike, ma esistono numerose altre varianti che possono essere integrate,

rendendo il sistema più flessibile e adattabile. Diversi tipi di P system, insieme ai relativi simulatori, sono descritti in [2] e, in modo più approfondito, nel paragrafo 2.2 di [24].

L'obiettivo di questa linea di lavoro è consentire la simulazione di differenti varianti del modello all'interno dello stesso framework, in modo da poter effettuare confronti diretti tra di esse. Sarà necessario individuare quali varianti risultano più promettenti e meritevoli di approfondimento, attraverso una fase iniziale di ricerca in letteratura e di analisi del loro potenziale impiego.

In assenza di questa fase di studio preliminare, l'attività potrebbe essere più adatta a uno stage, poiché maggiormente orientata alla componente implementativa del progetto.

Esempio di ricerca

- Approfondire una variante specifica degli SNPS, per esempio i P system con Polarization [35];
- Analizzare come questa variante si possa integrare nel **simulatore** in sviluppo;
- Lavorare nel proprio ramo GitHub, unificando successivamente la variante proposta, e osservandone le funzionalità.

9 Self Organizing Maps

Si propone di estendere il modello degli SNPS introducendo una componente spaziale, in modo che ogni neurone sia associato a delle coordinate (x, y) , o eventualmente (x, y, z) , rappresentando la sua posizione nello spazio. Le sinapsi collegherebbero quindi neuroni fisicamente vicini tra loro, consentendo di simulare strutture biologiche o anatomiche (ad esempio un polmone, un cuore o porzioni di cervello) in maniera spaziale, integrando concetti di graph drawing.

Per rendere il modello dinamico, si potrebbero introdurre regole che permettono ai neuroni di modificare la propria posizione o la lunghezza delle sinapsi in risposta agli stimoli. Una sinapsi più lunga potrebbe rappresentare un collegamento più debole, quindi vicino alla rottura. In tal modo si otterrebbe un sistema in grado di auto-organizzarsi nello spazio in base all'attività interna: questa idea è alla base del concetto delle **Self Organizing Maps** (SOM).

Nel lavoro [20] viene presentata una breve rassegna sulle SOM e sui loro principali utilizzi, con riferimento anche al lavoro [10], dove le SOM vengono applicate al riconoscimento di immagini mediche tramite un software dedicato, oggi parzialmente obsoleto ma comunque rilevante come punto di partenza.

In [12] viene invece mostrato un modello che usa le SNN e integra una SOM per classificare immagini semplici (le cifre del dataset MNIST, 28×28 pixel), fornendo un esempio concreto di come un approccio spiking possa favorire l'auto-organizzazione non supervisionata.

Ci sono quindi due possibili direzioni di ricerca: da un lato, approfondire lo sviluppo delle Spiking SOM (legate alle SNN); dall'altro, esplorare la possibilità di una SNPS con valori posizionali e con regole che ne permettano il movimento, avvicinando così il modello al comportamento auto-organizzante delle SOM.

Esempio di ricerca

- Approfondire la conoscenza del modello con paper come [34], che aggiungono la possibilità di creare e distruggere sinapsi grazie a delle regole;
- Programmare questa caratteristica così da utilizzarla per cambiare dinamicamente la struttura topologica della rete;
- Trovare un'area di ricerca e un dataset corrispondente dove questa caratteristica può essere utile, le aree possono variare dall'assemblaggio di galassie [13] alla selezione di specie rappresentative di comunità ecologiche [22];
- Analizzare il comportamento della rete applicata al problema scelto, confrontandolo con lavori simili.

10 Liquid State Machine

Le Liquid State Machine (LSM), introdotte in [17] e facenti parte del **Reservoir Computing**, sono descritte in modo intuitivo nel sito **linkato**, dove vengono presentati esempi di codice e approcci pratici alla loro implementazione. Si basano interamente sulle SNN e sono progettate per catturare efficacemente informazioni che variano nel tempo, come mostrato nel lavoro [23]. Questo le rende meno adatte a database statici, come le immagini radiografiche, ma molto utili per analizzare sequenze di immagini che mostrano evoluzioni nel tempo, ad esempio la crescita di un tumore o controlli ripetuti sulla stessa regione anatomica.

Sono disponibili diversi codici open source per la loro simulazione, ad esempio **questo** o **questo**, che include anche una spiegazione sintetica del funzionamento.

Un possibile sviluppo consisterebbe nell'approfondire la conoscenza di questo tipo di modello, con l'obiettivo di analizzare nel dettaglio il comportamento della SNN che lo costituisce. Sarà necessario individuare dataset adeguati al compito, ossia variabili nel tempo, e progettare una struttura per la loro classificazione.

Un aspetto interessante è evidenziato nel lavoro [42], che propone una combinazione tra LSM e SOM.

11 Digital Twin

I Digital Twin (DT) non sono semplici simulatori, ma modelli digitali che mantengono un legame continuo e bidirezionale con il sistema reale che rappresen-

tano. Ricevono in tempo reale dati provenienti da sensori o immagini, aggiornando dinamicamente il proprio stato e, a loro volta, possono inviare decisioni o previsioni al sistema fisico. In questo senso, il DT vive insieme al suo corrispettivo reale, invece di limitarsi a riprodurlo staticamente.

Le SNN si adattano naturalmente a questo paradigma, poiché elaborano dati temporali in modo asincrono e risultano intrinsecamente efficienti dal punto di vista energetico. Come discusso in [15], l'integrazione tra neuromorphic computing e Digital Twin Technology (DTT) può migliorare l'elaborazione in tempo reale, l'efficienza energetica e la capacità adattiva dei modelli, aprendo nuove prospettive per sistemi dinamici e scalabili.

Le possibili applicazioni includono la creazione di digital twin biomedici, ad esempio per il monitoraggio dell'attività cardiaca o cerebrale [36], in cui la rete spiking apprende e replica in tempo reale i pattern fisiologici, prevedendone l'evoluzione e individuando eventuali anomalie.

Questa direzione di ricerca, ancora poco esplorata, propone di utilizzare le SNN come cervello dinamico di un Digital Twin, unendo la modellazione temporale con la capacità predittiva e adattiva tipica del neuromorphic computing.

12 Privacy preserving SNPS

Un possibile ambito di ricerca riguarda la costruzione e l'addestramento di reti SNPS in modo che siano robuste e resistenti ad attacchi alla privacy. Questo tema è in parte legato al Federated Learning (FL), ma non completamente sovrapposto. I lavori in [16, 25, 26] affrontano il problema della privacy in reti di tipo spiking, mentre [27] propone una trattazione più dettagliata e chiara delle vulnerabilità e delle possibili soluzioni.

Il codice disponibile su **GitHub**, già citato nella sezione dedicata al FL, simula un attacco alla privacy. Tuttavia, non si tratta di uno SNPS formale: il modello implementato include diverse astrazioni non presenti nella definizione classica del sistema, come le funzioni softmax e argmax, l'uso esplicito di pesi sinaptici e le applicazioni lineari del tipo $x \cdot w + b$, tipiche delle reti neurali artificiali.

Si può quindi affermare che la teoria alla base di questi approcci sia già stata esplorata, ma che manchino ancora implementazioni pratiche coerenti con il modello SNPS. Questo rappresenta una direzione di ricerca promettente, volta a colmare il divario tra formalismo teorico e applicazioni reali nel contesto della privacy-preserving computation.

13 Modelli ibridi

Una possibile direzione di ricerca è l'unione di differenti modelli, oppure la costruzione di versioni ibride che rappresentino un compromesso sensato tra SNN, SNPS e altri. Soluzioni di questo tipo aprono la porta a numerose possibilità, sia come estensioni di modelli esistenti sia come nuove varianti. In ogni

caso, l'approccio è sempre quello di studiarne il comportamento da un punto di vista teorico e matematico, implementarne il funzionamento con un simulatore, e analizzarne le performance. Di seguito vengono illustrate alcune proposte.

- **SNPS con pesi:** Il lavoro [26] ha presentato risultati interessanti nell'ambito della cybersecurity, utilizzando uno SNPS modificato che include pesi in alcuni layer specifici. In [40] si utilizza un modello simile, chiamato Layered SN P System (LSNPS), che è weighted e fuzzy, per la classificazione di immagini. Risulta interessante notare che il codice in **questo link**, citato nella sezione sul FL, è stato fornito proprio dagli autori di questo paper. Si potrebbero analizzare modelli simili, studiandone la struttura in dettaglio e proponendo miglioramenti in termini di efficienza o robustezza del sistema.
- **SNPS with structural plasticity:** Gli SNPS with structural plasticity sono simili al codice sviluppato nel **simulatore**, in cui alcune sinapsi vengono distrutte o inibite durante la fase di training. Questi modelli includono regole dedicate a tale comportamento, e nella loro versione omogenea utilizzano regole identiche per tutti i neuroni. Anche se i lavori principali riguardano linguaggi formali più che applicazioni pratiche, risultano interessanti perché rappresentano in modo rigoroso dei concetti che, nel simulatore, sono stati esplorati in modo informale. Si vedano [4, 7] per ulteriori dettagli.
- **ANN + SNPS:** È possibile progettare modelli ibridi che combinano reti neurali artificiali (ANN) e SNPS, ad esempio utilizzando CNN per l'estrazione delle feature e SNPS per la fase di classificazione, con l'obiettivo di ottenere un maggiore risparmio energetico in questa seconda fase.
- **SNN/SNPS + GAN:** Queste architetture potrebbero essere utili in ambito medico, ad esempio per simulare l'evoluzione di un tumore nel tempo o generare scenari realistici per l'analisi prognostica. Viene proposto di utilizzare le Generative Adversarial Networks (GAN) per creare sequenze temporali che mostrino la progressione di noduli e patologie. Un approccio che integri SNN o SNPS con GAN potrebbe consentire una generazione più biologicamente plausibile dei dati temporali.
- **SOM + GAN:** Utilizzando una SOM-SNN come quella descritta in [12], si potrebbero associare classi di tessuto (ad esempio sano, malato o in fase di degenerazione) al posto delle cifre da 0 a 9. Come mostrato in figura 2, nel caso di MNIST le cifre si dispongono in modo ordinato nello spazio delle feature. In modo analogo, si potrebbe strutturare una SOM-SNN per disporre i neuroni in modo da ricostruire, ad esempio, la forma di un polmone. In questo scenario, la clusterizzazione non servirebbe solo a raggruppare classi simili, ma anche a ricreare una struttura spaziale coerente. Poiché i neuroni sono connessi da sinapsi, l'informazione posizionale rimarrebbe tracciabile, permettendo di valutare quanto la disposizione iniziale si

sia modificata per adattarsi all'immagine appresa. È un'idea complessa e sperimentale, ma potrebbe aprire direzioni interessanti, soprattutto combinata con una GAN che sia in grado di creare immagini realistiche di polmoni, ad esempio.

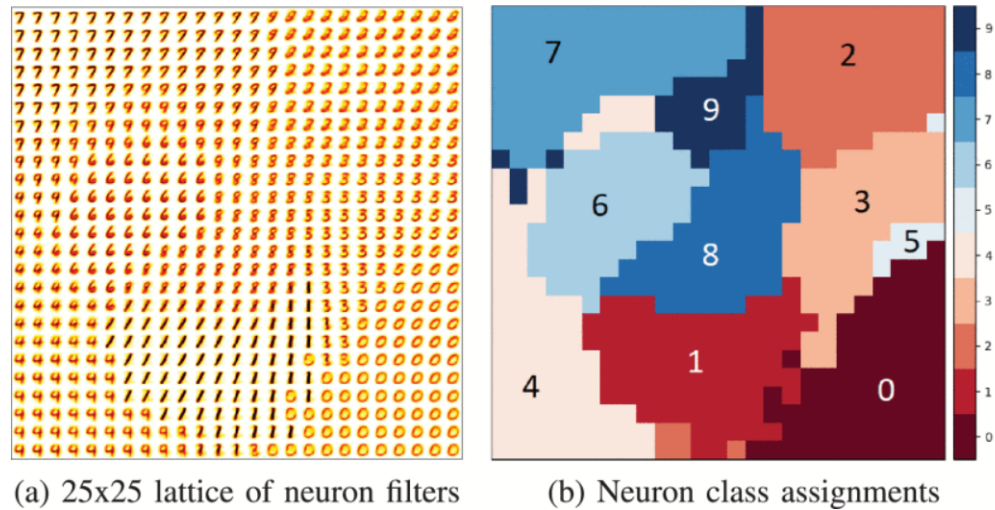


Figure 2: Il grafico in figura è tratto dalla SOM del paper [12], ma si hanno risultati simili anche in [42].

Bibliografia

- [1] Sales G Aribe Jr. “Spiking Neural Networks: The Future of Brain-Inspired Computing”. In: ().
- [2] Francis George C Cabarle. “Thinking about spiking neural P systems: some theories, tools, and research topics”. In: *Journal of Membrane Computing* 6.2 (2024), pp. 148–167.
- [3] Francis George C Cabarle and Henry N Adorna. “On structures and behaviors of spiking neural P systems and Petri nets”. In: *International Conference on Membrane Computing*. Springer. 2012, pp. 145–160.
- [4] Francis George C Cabarle et al. “Spiking neural P systems with structural plasticity”. In: *Neural Computing and Applications* 26.8 (2015), pp. 1905–1917.
- [5] Javier Carnero, Daniel Díaz-Pernil, and Miguel A Gutiérrez-Naranjo. “Designing tissue-like P systems for image segmentation on parallel architectures”. In: *Ninth Brainstorming Week on Membrane Computing 2011* (2011), pp. 43–62.

- [6] Lovely Joy P Casauay et al. “A Framework for Evolving Spiking Neural P Systems.” In: *International Journal of Unconventional Computing* 16 (2021).
- [7] Ren Tristan A de la Cruz et al. “Homogeneous spiking neural P systems with structural plasticity”. In: *Journal of Membrane Computing* 3.1 (2021), pp. 10–21.
- [8] Daniel Díaz-Pernil, Miguel A Gutiérrez-Naranjo, and Hong Peng. “Membrane computing and image processing: a short survey”. In: *Journal of Membrane Computing* 1.1 (2019), pp. 58–73.
- [9] Jianping Dong et al. “Automatic design of spiking neural P systems based on genetic algorithms”. In: *International Journal of Unconventional Computing* 16.2-3 (2021), pp. 201–216.
- [10] L Grajciarova et al. “Biomedical image analysis using self-organizing maps”. In: *Matlab Conference*. 2012.
- [11] Mutya Gulapa et al. “Websnapse reloaded: The next-generation spiking neural p system visual simulator using client-server architecture”. In: *Workshop on Computation: Theory and Practice (WCTP 2023)*. Atlantis Press. 2024, pp. 434–461.
- [12] Hananel Hazan et al. “Unsupervised learning with self-organizing spiking neural networks”. In: *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2018, pp. 1–6.
- [13] Benne W Holwerda et al. “Galaxy and mass assembly (GAMA): Self-Organizing Map application on nearby galaxies”. In: *Monthly Notices of the Royal Astronomical Society* 513.2 (2022), pp. 1972–1984.
- [14] Mihai Ionescu, Gheorghe Păun, and Takashi Yokomori. “Spiking neural P systems”. In: *Fundamenta informaticae* 71.2-3 (2006), pp. 279–308.
- [15] Vijayakumar Kempuraj and C Lakshmi. “A Review of Neuromorphic Computing and Its Potential for Enhancing Digital Twin Technology”. In: *International Conference on Computational Intelligence in Data Science*. Springer. 2025, pp. 224–242.
- [16] Youngeun Kim, Yeshwanth Venkatesha, and Priyadarshini Panda. “Privatesnn: privacy-preserving spiking neural networks”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 36. 1. 2022, pp. 1192–1200.
- [17] Wolfgang Maass, Thomas Natschläger, and Henry Markram. “Real-time computing without stable states: A new framework for neural computation based on perturbations”. In: *Neural computation* 14.11 (2002), pp. 2531–2560.
- [18] Richard H Masland. “The neuronal organization of the retina”. In: *Neuron* 76.2 (2012), pp. 266–280.

- [19] Venkata Padmavati Metta, Kamala Krithivasan, and Deepak Garg. “Spiking Neural P systems and Petri nets”. In: *Proc. of the Int’l Workshop on Machine Intelligence Research*. 2009.
- [20] Dubravko Miljković. “Brief review of self-organizing maps”. In: *2017 40th international convention on information and communication technology, electronics and microelectronics (MIPRO)*. IEEE. 2017, pp. 1061–1066.
- [21] Hai Nan et al. “Competitive Nonlinear Layered Spiking Neural P System for solving classification problems”. In: *Neurocomputing 637* (2025), p. 130036.
- [22] Young-Seuk Park et al. “Application of a self-organizing map to select representative species in multivariate analysis: a case study determining diatom distribution patterns across France”. In: *Ecological Informatics 1.3* (2006), pp. 247–257.
- [23] Meet K Patel et al. “Image Classification using Convolution Liquid State Machines”. In: *2025 3rd International Conference on Sustainable Computing and Data Communication Systems (ICSCDS)*. IEEE. 2025, pp. 2084–2091.
- [24] Prithwineel Paul, Petr Sosik, and Lucie Ciencialova. “A survey on learning models of spiking neural membrane systems and spiking neural networks”. In: *arXiv preprint arXiv:2403.18609* (2024).
- [25] Mihail-Iulian Plesa, Marian Gheorghe, and Florentin Ipate. “Privacy-preserving Linear Computations in Spiking Neural P Systems”. In: *arXiv preprint arXiv:2309.13803* (2023).
- [26] Mihail-Iulian Pleșa et al. “Applications of spiking neural P systems in cybersecurity”. In: *Journal of Membrane Computing 6.4* (2024), pp. 310–317.
- [27] Mihail-Iulian Pleșa, Marian Gheoghe, and Florentin Ipate. “Private Inference on Layered Spiking Neural P Systems”. In: *International Work-Conference on the Interplay Between Natural and Artificial Computation*. Springer. 2024, pp. 163–172.
- [28] Antonio Ramírez-de-Arellano, David Orellana-Martín, and Mario J Pérez-Jiménez. “Bridges between spiking neural membrane systems and virus machines”. In: *International Journal of Neural Systems 34.06* (2024), p. 2450034.
- [29] Yonglin Ren and Gang Zhao. “A Novel Spiking Neural Network for Wearable-Based Human Activity Recognition”. In: *2024 6th International Conference on Frontier Technologies of Information and Computer (ICFTIC)*. IEEE. 2024, pp. 1657–1661.
- [30] Tao Song et al. “Spiking neural P systems with white hole neurons”. In: *IEEE transactions on nanobioscience 15.7* (2016), pp. 666–673.
- [31] Martino Sorbaro et al. “Optimizing the energy consumption of spiking neural networks for neuromorphic applications”. In: *Frontiers in neuroscience 14* (2020), p. 662.

- [32] Dipanwita Thakur et al. “Green federated learning: A new era of green aware AI”. In: *ACM Computing Surveys* 57.8 (2025), pp. 1–36.
- [33] Yeshwanth Venkatesha et al. “Federated learning with spiking neural networks”. In: *IEEE Transactions on Signal Processing* 69 (2021), pp. 6183–6194.
- [34] Xun Wang et al. “On the computational power of spiking neural P systems with self-organization”. In: *Scientific reports* 6.1 (2016), p. 27624.
- [35] Tingfang Wu et al. “Spiking neural P systems with polarizations”. In: *IEEE transactions on neural networks and learning systems* 29.8 (2017), pp. 3349–3360.
- [36] Hui Xiong et al. “The digital twin brain: A bridge between biological and artificial intelligence”. In: *Intelligent Computing* 2 (2023), p. 0055.
- [37] Claudio Zandron. “An Overview on Applications of Spiking Neural Networks and Spiking Neural P Systems”. In: *Languages of Cooperation and Communication: Essays Dedicated to Erzsébet Csuhaj-Varjú to Celebrate Her Scientific Career* (2025), pp. 267–278.
- [38] Xiangxiang Zeng et al. “Matrix representation of spiking neural P systems”. In: *International conference on membrane computing*. Springer, 2010, pp. 377–391.
- [39] Yi Zeng et al. “Braincog: A spiking neural network based, brain-inspired cognitive intelligence engine for brain-inspired ai and brain simulation”. In: *Patterns* 4.8 (2023).
- [40] Gexiang Zhang et al. “A layered spiking neural system for classification problems”. In: *International journal of neural systems* 32.08 (2022), p. 2250023.
- [41] Gexiang Zhang et al. “An optimization spiking neural P system for approximately solving combinatorial optimization problems”. In: *International Journal of Neural Systems* 24.05 (2014), p. 1440006.
- [42] Youdong Zhang et al. “Unsupervised spiking neural network based on liquid state machine and self-organizing map”. In: *Neurocomputing* 620 (2025), p. 129120.
- [43] Ming Zhu et al. “An adaptive optimization spiking neural P system for binary problems”. In: *International Journal of Neural Systems* 31.01 (2021), p. 2050054.