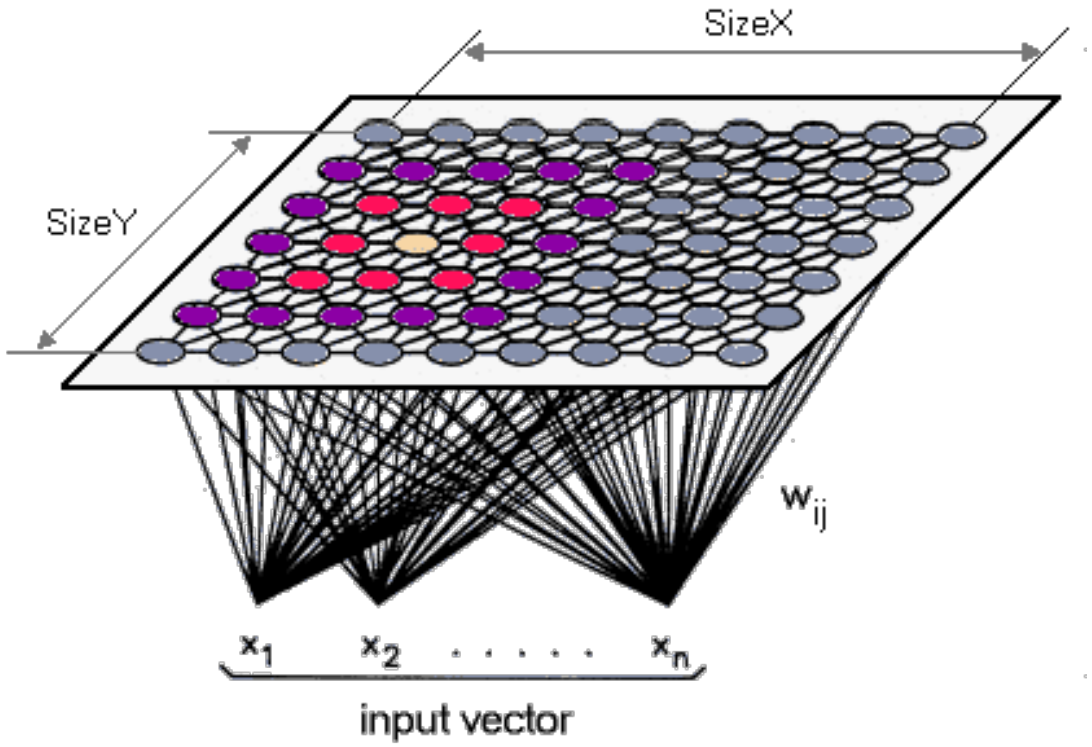
SOM PER IDS

INTRODUZIONE

- Le Self-Organizing Maps (SOM) sono un tipo di rete neurale artificiale inventata da Teuvo Kohonen negli anni 80.
- Sono allenate con un apprendimento **non supervisionato**, questo significa che i dati di training non sono etichettati.
- L'obiettivo è la riduzione della dimensionalità e la visualizzazione dei dati da alta dimensionalità in una mappa
- Si differenziano dalle altre reti neurali, perché si basano su un apprendimento **competitivo.**
- Sono usate per il clustering, il riconoscimento di pattern, estrazione di features e visualizzazione.
- Nel campo della cybersecurity, sono usate nel rilevamento di anomalie, nella classificazione delle minacce e nell'analisi del comportamento di rete.

ARCHITETTURA DELLE SOM



- I neuroni delle SOM sono organizzati in una griglia bidimensionale, nella quale ogni neurone possiede un vettore di pesi, della stessa dimensione del vettore di input. Questa griglia rappresenta lo spazio in cui i dati di input vengono proiettati.
- I vettori di peso collegano ciascun neurone a tutti gli attributi dell'input. L'aggiornamento di questi vettori attraverso l'apprendimento.
- La struttura spaziale della griglia aiuta a visualizzare e comprendere le somiglianze e le differenze all'interno del dataset, con configurazioni rettangolari o esagonali.

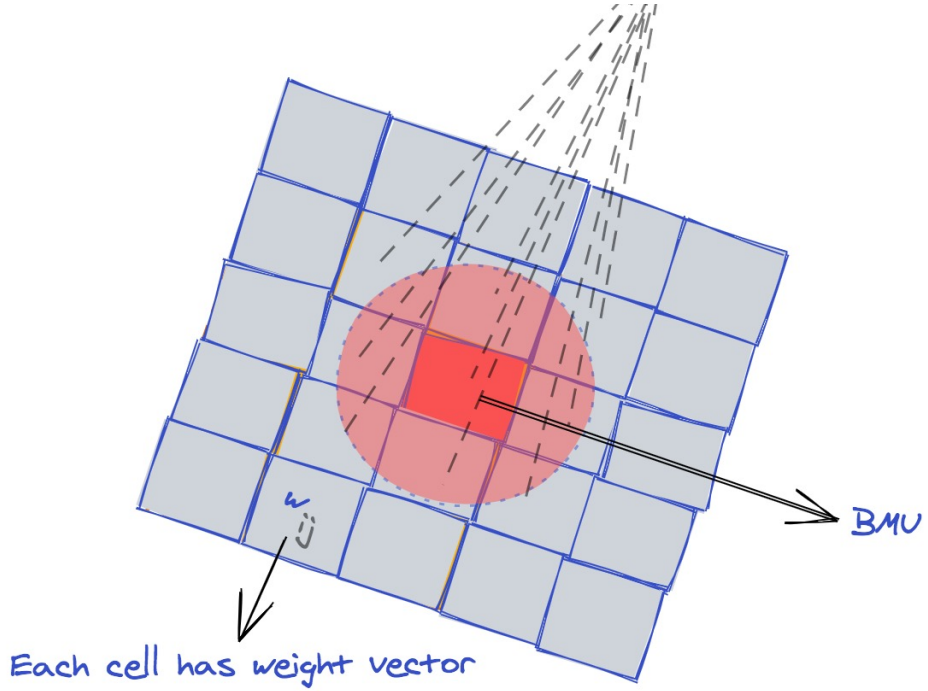
PANORAMICA DEL PROCESSO DI APPRENDIMENTO

- Obiettivo delle SOM: Trasformare dati complessi, ad alta dimensionalità in rappresentazioni semplificate, preservando relazioni topologiche. Permettendo all'utilizzatore una maggiore comprensione.
- Fasi dell'Algoritmo: Il processo di apprendimento si compone di quattro fasi - **Inizializzazione**, **Competizione**, **Cooperazione e Adattamento**.
- L'algoritmo procede iterativamente, presentando sequenze di vettori di input e aggiornando i pesi dei neuroni per riflettere strutture e pattern nei dati.
- L'apprendimento procede fino a che non si osservano più variazioni sostanziali nella mappa, segnale che l'algoritmo ha raggiunto un equilibrio.

FASE DI INIZIALIZZAZIONE

- Scelta degli iperparametri, prima dell'apprendimento: si definiscono il tasso di apprendimento iniziale, il numero di iterazioni, la dimensione della griglia ,la funzione di vicinanza e si determina la metrica da utilizzare per la competizione tra i neuroni.
- Queste scelte influenzano l'efficacia dell'addestramento e la qualità della mappa risultante. Inoltre per quanto riguarda l'inizializzazione del vettore dei pesi vi sono 2 strategie:
- Inizializzazione **casuale**: I vettori dei pesi di ogni neurone nella griglia vengono inizializzati con valori casuali. Questo assicura una partenza non pregiudiziale per il processo di auto-organizzazione.
- Inizializzazione **basata sui dati** : Una buona inizializzazione può accelerare la convergenza dell'algoritmo e migliorare la rappresentatività della mappa.

FASE COMPETIZIONE (1)



- Ogni volta che viene introdotto un nuovo vettore di input alla SOM, inizia una fase cruciale di competizione tra i neuroni. È in questo momento che si decide quale neurone diventerà il rappresentante di quell'input specifico.
- **Rappresentazione dei Dati**: La competizione assicura che ogni input sia rappresentato sulla mappa dalla cella più adatta, preservando le similitudini e differenze tra i diversi input.

FASE COMPETIZIONE (2)

- L'algoritmo delle SOM **identifica** il **Best Matching Unit (BMU)**, basandosi su una metrica come la distanza euclidea (oppure, per es. di Manhattan) tra il vettore di input e i vettori di peso di ciascun neurone. Il neurone con la distanza minima viene eletto come BMU.
- Formula della Distanza Euclidea: $d(\vec{x}, \vec{w}) = \sqrt{\sum_{i=1}^{n}(x_i - w_i)^2}$

  Dove: $\vec{x}$ rappresenta il vettore di input e $\vec{w}$ il vettore di peso del neurone candidato a diventare BMU.
- Il BMU è il neurone che presenta la maggior somiglianza con l'input corrente
- La selezione del BMU è fondamentale perché determina il modo in cui i dati verranno rappresentati sulla mappa.

FASE DI COOPERAZIONE E ADATTAMENTO(1)

- Il Best Matching Unit (BMU), una volta individuato, avvia una fase di cooperazione con i neuroni circostanti. Tale interazione si propaga oltre il BMU, influenzando un'area circostante definita come **neighborhood**.
- Il neighborhood è la zona intorno al BMU dove i neuroni subiscono un aggiustamento dei loro pesi per allinearsi meglio con il vettore di input corrente. L'entità di questa modifica è regolata da un **tasso di apprendimento** variabile, che di norma si riduce progressivamente durante il processo di addestramento.
- La funzione di vicinanza (es. Gaussiana o Triangolo) stabilisce quanto intensamente i pesi dei neuroni nel neighborhood vengono aggiornati in risposta al vettore di input. Questa funzione è più forte per i neuroni immediatamente adiacenti al BMU e diminuisce con l'aumentare della distanza all'interno del vicinato.
- Formule di Aggiustamento: I pesi dei neuroni vicini al BMU vengono aggiornati secondo la formula:
- $w_v(t+1) = w_v(t) + \theta(u, v, t) \cdot \alpha(t) \cdot \big(x(t) - w_v(t)\big)$

$w_v(t)$:vettore peso del neurone v ; $\theta(u, v, t)$:funzione di vicinanza tra il BMU u e il neurone v ;
$\alpha(t)$ tasso di apprendimento, $x(t)$ vettore di input.

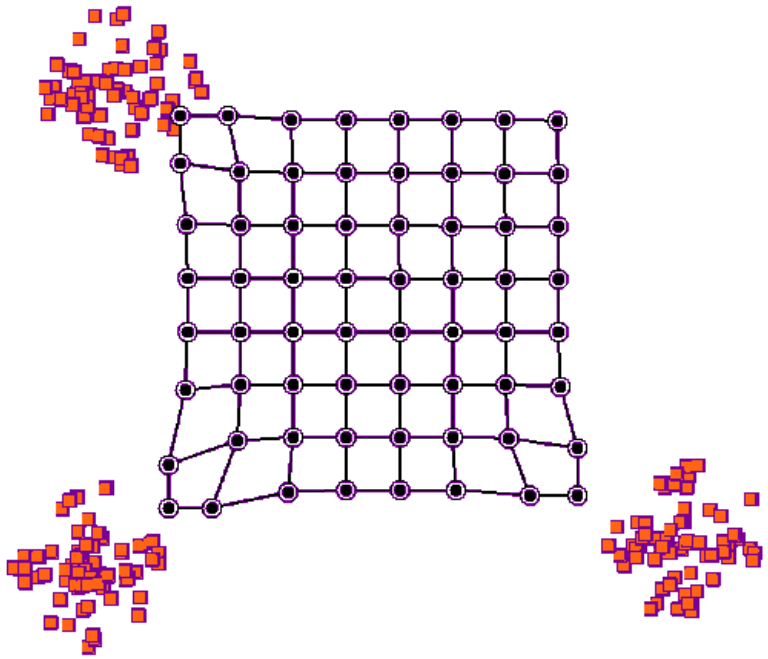
FASE DI COOPERAZIONE E ADATTAMENTO(2)

- L'obiettivo della fase di cooperazione è di rendere la mappa "elastica", permettendo ai neuroni di adattarsi non solo all'input attuale ma anche a input simili che potrebbero essere presentati in futuro.
- Attraverso la cooperazione, la SOM mantiene la topologia dei dati di input. Neuroni che rappresentano input simili si trovano vicini nella mappa.
- La cooperazione è essenziale per l'apprendimento in una SOM. È attraverso questo processo che la mappa impara a organizzarsi in modo che pattern simili di input risultino in neuroni attivati in regioni vicine della mappa.

L'ALGORITMO IN BREVE

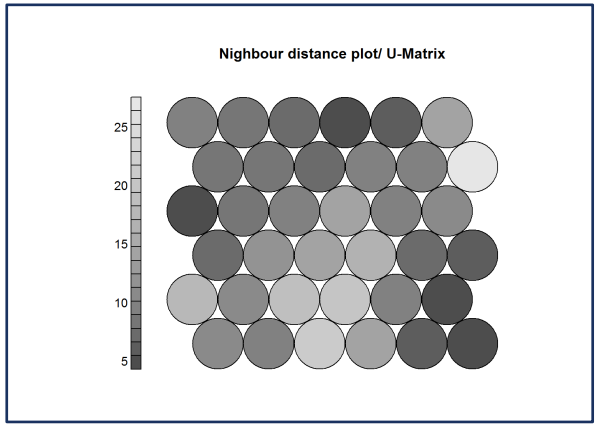
- **Inizializzazione**: Si inizia definendo gli iperparametri e assegnando i pesi iniziali ai neuroni della mappa.
- **Competizione**: la rete è soggetta a una competizione tra i neuroni della mappa per determinare il Best Matching Unit (BMU), ovvero il neurone con i pesi più simili al vettore di input secondo una metrica di distanza.
- **Cooperazione**: i neuroni nel vicinato del neighborhood aumentano la loro somiglianza con il vettore di input. La dimensione del vicinato, definita da una funzione radiale, tende a diminuire nel corso delle iterazioni.
- **Adattamento**: Il BMU e i suoi vicini aggiustano i loro pesi per avvicinarsi ancora di più al vettore di input. Questa fase di adattamento è guidata da un tasso di apprendimento che si riduce nel tempo, rendendo i cambiamenti più fini man mano che l'algoritmo procede.
- Iterazione: Le fasi di Competizione, Cooperazione e Adattamento si ripetono per numerosi cicli. Ogni presentazione di un vettore di input e l'adattamento dei pesi conseguente è chiamato "**epoca**". Con molteplici epoche, la mappa si auto-organizza, fino a che non si raggiunge una convergenza, dove i cambiamenti ai pesi dei neuroni diventano minimi, indicando che la mappa si è stabilizzata e rappresenta in modo adeguato il dominio di input.

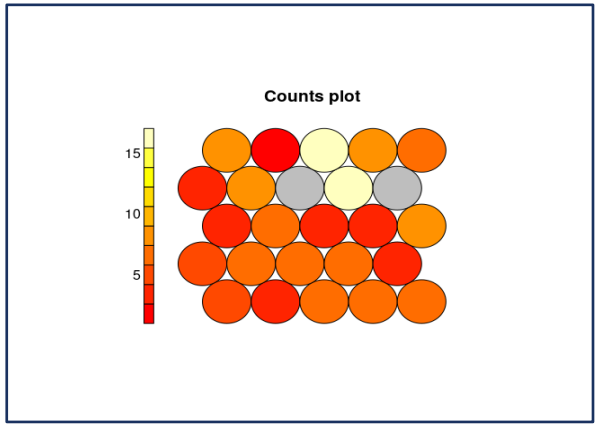
OSSERVAZIONI

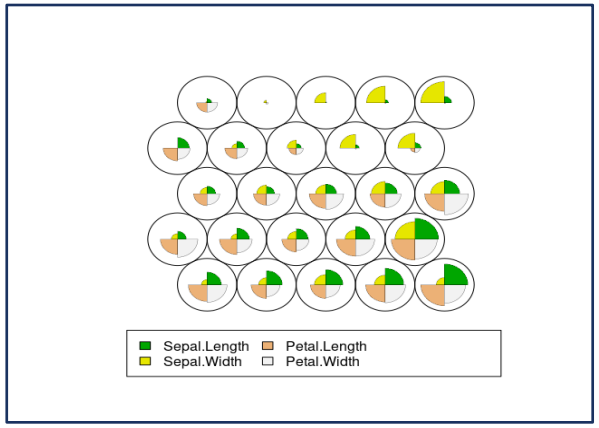


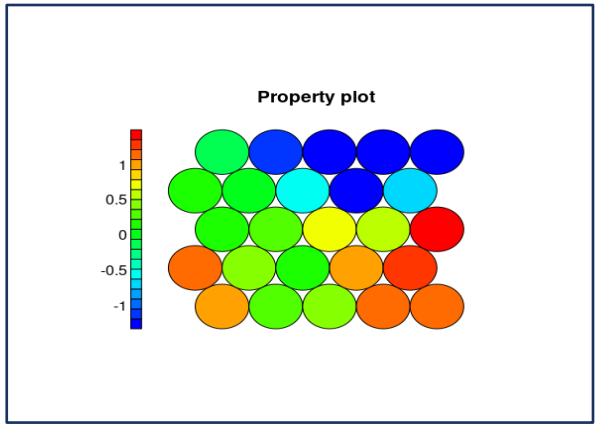
- In seguito al processo di apprendimento, i vettori dei pesi associati a ciascun neurone vengono aggiornati allo scopo di ottimizzare la capacità di ogni neurone di rappresentare in modo più accurato i dati di input.
- La distanza tra i neuroni nella mappa non cambia fisicamente, perché la griglia è fissa.
- L'aggiornamento dei pesi modifica la 'distanza funzionale', che è utilizzata per indicare quanto siano simili le rappresentazioni nel rappresentare un dato input sulla mappa.
- Neuroni che rappresentano dati simili sono considerati vicini in termini di distanza funzionale.

RAPPRESENTAZIONI SOM









- Dopo aver allenato la rete vi sono vari modi per poterla visualizzare, i più famosi sono:
- **Distanza tra Vicini** (**U-Matrix**):rappresenta la distanza funzionale tra ogni nodo e i suoi vicini. Aree con una bassa distanza indicano gruppi di nodi simili.
- **Conteggio dei Nodi:** permette di visualizzare quanti campioni sono mappati su ogni nodo della mappa. Può misurare la qualità della mappa. Nodi vuoti indicano che la dimensione della tua mappa è troppo grande per il numero di campioni.
- **Vettori di Peso dei Nodi/Codebook**: sono collezioni di vettori di peso che rappresentano in maniera normalizzata le variabili di input. Essi sono visualizzati con diagrammi a ventaglio per evidenziare la rilevanza di ogni variabile nei diversi nodi, offrendo un'intuizione sulla distribuzione dei dati nella mappa.
- **Heatmaps:** mostrano la distribuzione di specifiche variabili sulla mappa SOM, aiutando a individuare pattern e aree di interesse mantenendo fissa la posizione dei campioni.

VALUTAZIONE DELLA SOM

- Le principali misure di qualità utilizzate per valutare le SOM sono l'errore di quantizzazione (QE) e l'errore topografico (TE):
- Errore di Quantizzazione (QE): Misura la distanza media tra i punti dati e i nodi della mappa ai quali sono mappati. Valori più bassi indicano una migliore adattabilità del modello. QE è utile per confrontare diverse mappe.

  $QE(M) = \frac{1}{n}\sum_{i=1}^{n}\|\varphi(x_i) - x_i\|$ dove:

  n è il numero di punti dati nei dati di allenamento, x_i rappresenta il *i*-esimo punto dato, $\varphi(x_i)$ è il nodo della mappa SOM a cui il punto x_i è mappato, $\|\varphi(x_i) - x_i\|$ calcola la distanza in base alla metrica scelta per esempio euclidea tra il punto dato e il suo nodo corrispondente nella mappa.

- Errore Topografico (TE): Valuta la capacità della mappa di preservare la struttura topologica dello spazio di input nel suo spazio di output a bassa dimensione. TE si concentra sulle discontinuità locali del mapping, contando come errori i casi in cui i nodi corrispondenti migliori non sono vicini tra loro.

  $\mathrm{T}E(M) = \frac{1}{n}\sum_{i=1}^{n} t(x_i)$ dove:

  $t(x)$ è una funzione che vale 0 se l'unità che meglio corrisponde $(\mu(x))$ e la seconda migliore corrispondente ($\mu_0(x)$) a un punto dato x sono vicine (ad esempio, sono vicini nella mappa), e 1 altrimenti.

IMPLEMENTAZIONI DELLE SOM IN PYTHON

- L'implementazione delle Self-Organizing Maps (SOM) in Python può essere resa accessibile e flessibile attraverso diverse librerie, ciascuna con i suoi punti di forza e casi d'uso ideali.
- **MiniSom** libreria pratica per prototipi veloci e analisi esplorative, con facile configurazione e visualizzazione.
- **Somoclu:** si distingue per gestire grandi dataset su hardware avanzato, offrendo parallelizzazione.
- **SUSI** è adatto per valutazioni approfondite dell'impatto dei modelli di apprendimento.
- La scelta della libreria SOM in Python dipende dalla dimensione dei dati (Somoclu ottimo per grandi set), necessità di visualizzazione (MiniSom e SUSI per dettagli avanzati) e livello dell'utente (MiniSom ideale per principianti).

SCELTA DELLA LIBRERIA

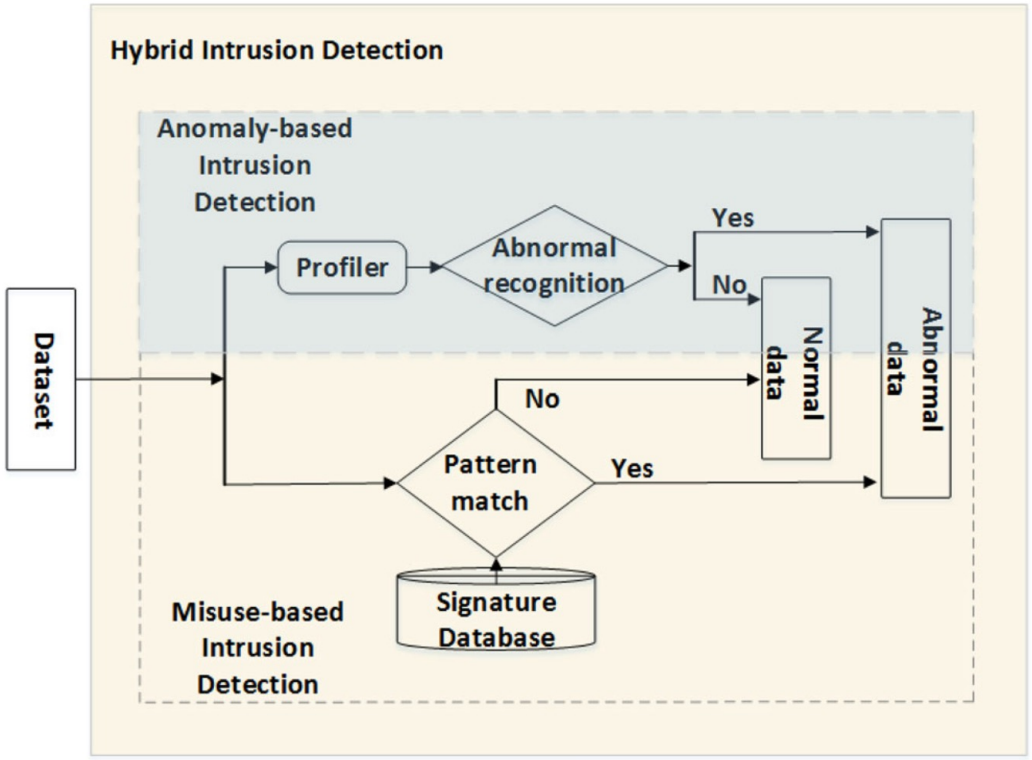
- Ho selezionato la libreria MiniSom per implementare le SOM perché, secondo Snyk.io ha evidenziato ottima manutenzione e sicurezza ottenendo un punteggio di 77/100, superiore a Somoclu (51/100) e a SUSI (63 / 100), considerando aspetti quali sicurezza, popolarità e supporto della comunità. Questo punteggio aiuta a scegliere librerie affidabili e ben supportate. BasicUsage Minisom
- Esempi implementazioni progetti con Minisom:

1. HandwrittenDigits
2. Identifying Breast Cancer Clusters
3. Fraud Detection

SOM APPLICATE ALL'IDS

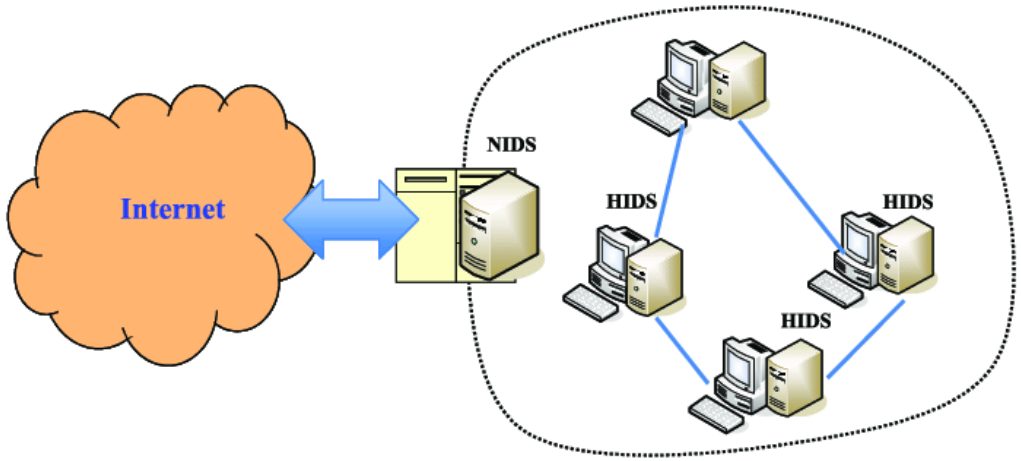
- Le SOM possono essere usate in diversi ambiti come: medicina, marketing, bioinformatica, e analisi finanziaria. Ma noi ci concerteremo sull'utilizzo di questa rete neurale nella Cybersecurity, più nello specifico nell'IDS.
- L'IDS è utilizzato per identificare intrusioni informatiche in reti di computer ha una risposta in tempo reale. Raccoglie informazioni da diversi punti ed esegue analisi per scoprire violazioni di sicurezza.
- Esistono diversi tipi di sistemi di intrusion detection, in base alla loro funzionalità e all'approccio con cui gestiscono intrusioni e anomalie, i due principali sono:
- I sistemi **signature-based** (misuse-based) contengono un database di signature generate che vengono utilizzate per riconoscere entità malevoli esistenti.
- I sistemi **anomaly-based**, mantengono una linea di base del comportamento normale di un sistema, che viene utilizzata per riconoscere se il comportamento di un sistema si discosta in qualche modo da questa linea di base.

SIGNATURE-BASED, ANOMALY-BASED E HYBRID



- La maggior parte degli studi sono stati effettuati su sistemi basati su **anomalie**..
- Ci sono solo alcuni sistemi che possono essere considerati basati su signature nel senso tradizionale. Tutti questi sistemi sono **sistemi ibridi**, che combinano sia tecniche basate su anomalie sia tecniche basate su **signature** al fine di ottenere le migliori capacità di rilevamento possibili.

HIDS E NIDS



- Il secondo gruppo di categorie distingue i sistemi in base al tipo di informazioni che monitorano. Questi sistemi possono essere suddivisi in:
- **HIDS** raccolgono informazioni dal sistema host locale, come log di sistema, file system, comportamento degli utenti, utilizzo di CPU e memoria. Generalmente proteggono il sistema in cui risiedono.
- **NIDS** monitorano il traffico di rete e raccolgono i pacchetti di rete originali con strumenti di cattura pacchetti. Successivamente, analizzano le informazioni dell'intestazione del pacchetto e i payload per rilevare minacce.
- I NIDS eseguono analisi in tempo reale e prevengono attacchi online, mentre gli HIDS conducono analisi post-fatto per prevenire attacchi futuri.

TIPI DI ATTACCHI NELLA RETE

- La maggior parte delle ricerche effettuate che usano le SOM sono basate sul rilevamento di intrusioni nella rete. Ci sono quattro principali categorie di attacchi nella rete. Ogni attacco a una rete può essere classificato in uno di questi gruppi:
- **Denial of Service** (DoS): attacco in cui l'hacker rende le risorse di memoria occupate per servire richieste di rete legittime, negando così agli utenti l'accesso a una macchina, come apache, smurf, Neptune, ping of death, mail bomb, tempesta UDP, ecc.
- **Attacchi da Remoto a Utente** (Remote to User attacks, R2L): un attacco da remoto a utente è un attacco in cui un utente invia pacchetti a una macchina attraverso Internet, senza avere accesso, al fine di esporre le vulnerabilità della macchina e sfruttare i privilegi che un utente locale avrebbe sul computer, ad esempio xlock, guest, xnsnoop, phf, sendmail dictionary, ecc.
- **Attacchi da Utente a Root** (User to Root attacks, U2R): questi attacchi sono sfruttamenti in cui l'hacker inizia nel sistema con un normale account utente e tenta di abusare delle vulnerabilità del sistema al fine di ottenere privilegi di super utente, come perl, xterm.
- **Probing**: un attacco in cui l'hacker scandisce una macchina o un dispositivo di rete per determinare debolezze o vulnerabilità che potrebbero in seguito essere sfruttate per compromettere il sistema. Questa tecnica è comunemente utilizzata nel data mining, ad esempio satan, saint, portsweep, mscan, nmap, ecc.
- Lo scopo dei classificatori negli IDS è di identificare attacchi provenienti da tutti e quattro i gruppi nel modo più accurato possibile.

DATASET KDD CUP'99

S.NO	FEATURE NAME	S.NO	FEATURE NAME
1	Duration	22	Is_guest_login
2	Protocol type	23	Count
3	Service	24	Serror_rate
4	Src_byte	25	Rerror_rate
5	Dst_byte	26	Same_srv_rate
6	Flag	27	Diff_srv_rate
7	Land	28	Srv_count
8	Wrong_fragment	29	Srv_serror_rate
9	Urgent	30	Srv_rerror_rate
10	Hot	31	Srv_diff_host_rate
11	Num_failed_logins	32	Dst_host_count
12	Logged_in	33	Dst_host_srv_count
13	Num_compromised	34	Dst_host_same_srv_count
14	Root_shell	35	Dst_host_diff_srv_count
15	Su_attempted	36	Dst_host_same_src_port_rate
16	Num_root	37	Dst_host_srv_diff_host_rate
17	Num_file_creations	38	Dst_host_serror_rate
18	Num_shells	39	Dst_host_srv_serror_rate
19	Num_access_shells	40	Dst_host_rerror_rate
20	Num_outbound_cmds	41	Dst_host_srv_rerror_rate
21	Is_hot_login		

- La maggior parte delle ricerche sul rilevamento delle intrusioni utilizza il dataset KDD Cup'99. Sebbene il KDD99 presenti alcune carenze, questo dataset è il primo e affidabile set di dati di riferimento.
- Il KDD99 contiene quattro tipi di attacchi, che sono DOS ,R2L,U2R,PROB con cinque milioni di record . KDD99,contiene 42 elementi. L'ultimo elemento è un'etichetta, che indica se i dati erano normali o rappresentavano un attacco.
- KDD99 non riflette le statistiche reali del traffico a causa della sua origine simulata.

DATASET NSL-KDD

- KDD Cup '99 presenta un gran numero di voci ridondanti che rendono difficile categorizzare i record rimanenti.
- È stato suggerito un nuovo dataset NSL-KDD per affrontare queste preoccupazioni. Il dataset NSL-KDD è composto da un piccolo numero di caratteristiche del dataset KDD Cup '99 che non sono ridondanti nel set di training o duplicate nel set di test. Le ragioni convincenti per utilizzare il dataset negli esperimenti sono:
- Rimuove i dati duplicati dal set di training permette ai classificatori di essere più imparziali quando si occupano di record sempre più frequenti.
- I set di training e test contengono un numero sufficiente di istanze per consentire test sull'intero set senza la necessità di selezionare un piccolo pezzo a caso.
- Il numero di record presenti in questo dataset 150.000 record

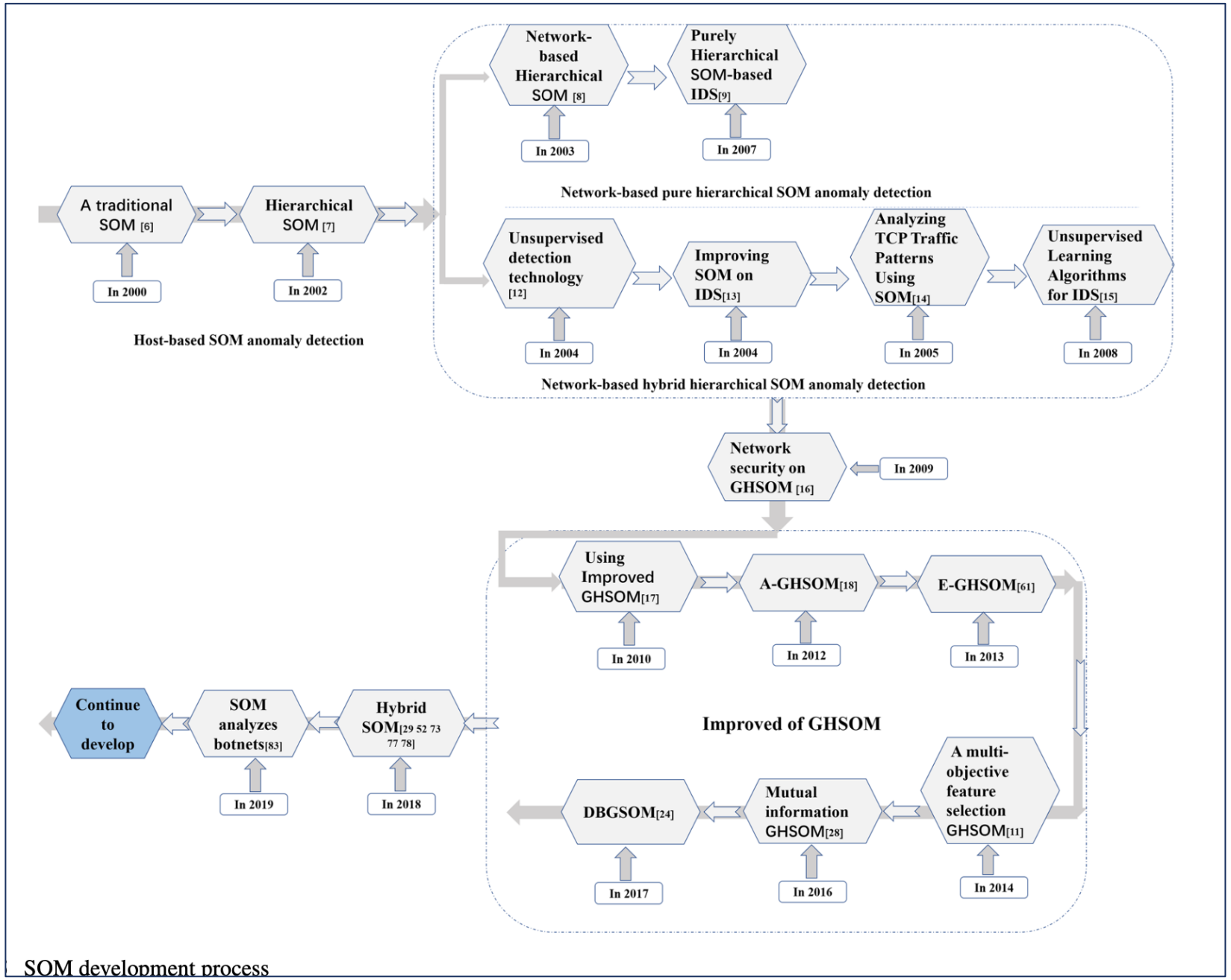
DATASET PIÙ MODERNI

- Nonostante KDD99 e NSL-KDD siano spesso usati nelle ricerche, questi dataset presentano alcune limitazioni, come la mancanza di rappresentazione degli attacchi più recenti.
- Per questo motivo, dataset più recenti come CICIDS2017 e CSE-CIC-IDS2018 hanno preso piede, poiché forniscono una visione più attuale e dettagliata delle minacce alla sicurezza.
- Entrambi hanno milioni di istanze e una ricca varietà di caratteristiche, permettono una modellazione e analisi più profonda.
- Sebbene all'apparenza sembrano essere migliori presentano degli svantaggi:

1. Richiedono più risorse computazionali per l'elaborazione dovuta alla vasta quantità di dati e varietà di attacchi
2. Richiedere una significativa preelaborazione prima di poter utilizzare i dati grezzi efficacemente.

RISULTATI OTTENUTI DALLE SOM E VARIANTI IN IDS

- Facendo riferimento A Survey on the Development of Self-Organizing Maps for Unsupervised Intrusion Detection le SOM presentano dei discreti risultati per quanto riguarda questo ambito, però sono state sviluppate delle varianti delle SOM molto più formati per questo caso d'uso come:
- Le **HSOM** introducono una struttura gerarchica che migliora l'efficienza computazionale e rappresenta i dati in modo più strutturato, consentendo una più facile identificazione dei pattern e delle anomalie nel traffico di rete.
- Le **GHSOM**, invece, estendono le HSOM consentendo alle mappe di crescere dinamicamente a seconda della complessità dei dati, offrendo un modello ancora più adattabile e scalabile per la rilevazione di intrusioni in tempo reale. Le GHSOM, e le sue ottimizzazioni, offrono prestazioni notevolmente migliori nel rilevare intrusioni rispetto all'architettura SOM originale, sottolineando l'importanza dell'innovazione continua nel campo della sicurezza informatica per adattarsi alle minacce in evoluzione.



SOM development process

Methods	DR	FR	Features number	Disadvantage	Advantages
A raw SOM architecture [7]	89%	4.6%	6	The hierarchical relationship of data is not fully mapped. The detection rate is not high and the false alarm rate is high.	Using only the six most basic features, it achieves a detection rate comparable to 41 features.
A 2 or 3 hierarchical SOM architecture [8]	90.4%	1.38%	6 and 41	It takes a lot of experiments to get the best results. Use static hierarchical SOMs architecture.	Different characteristic data correspond to different SOM. Achieve the ideal detection rate with a small number of features.
Growing Hierarchical Self-Organizing Map (GHSOM) [15]	99.99%	3.73%	41	It results in a high false alarm rate, which may be caused by the small amount of attacking sample data and the poor accuracy of SOM mapping.	High detection rate.
Improved GHSOM [16]	96.2%	–	21	The detection rate of U2R and R2L is not high.	Obtain higher detection rate and improve the stability of intrusion detection.
ine an adaptive GHSOM (A-GHSOM) [17]	99.63%	1.8%	41	The classification calculation has high complexity, and there are redundant and irrelevant data, which affects the performance of the classifier.	Adapt online to changing exception detection.
A multi-objective feature selection HSOM [10]	99.6%	4.32%	41	The false positive rate is still relatively high compared to other methods.	Reduce the computational complexity, higher detection rate.
	99.12%	2.24%	25		
I–NGSA–III + GHSOM-pr [26]	99.37%	–	20	The detection rate of U2R and R2L is still not high.	For classes with fewer instances, the classification accuracy is higher. It can achieve higher detection accuracy with lower computational complexity.
GHSOM + mutual information [27]	97.73%	–	41	This method is only applicable to the entire cluster, and there are still some abnormal attacks that cannot be effectively clustered.	The detection rate of U2R and R2L is greatly improved.
DR is Detection rate, FR is False positive rate.					

RETI SOM E VARIANTI VS ALTRE ANN

- Nel campo dell'IDS, le reti Som e le sue varianti sono state confrontate con altre Ann che non si basano su apprendimento competitivo.
- Le Ann non competitive sono risultate più accurate rispetto della miglior versione della Som di circa di 1-3%.
- Le Ann che non si basano su apprendimento competitivo sono considerate delle black box, poichè per l'essere umano risulta difficile comprendere la scelta da parte del modello.
- Per questo motivo nel seguente articolo: Explainable Intrusion Detection Systems Using Competitive Learning Techniques, le Som e varianti sono considerate come una valida scelta per poter implementare l'XIDS(Explainable IDS), permettendo agli esperti di settore una maggiore comprensione della scelta effettuata dal modello, poiché le SOM sono white box. Potendo diventare dei validi tool di consultazione per esperti e analisi.