ALMA MATER STUDIORUM – UNIVERSITÀ DI BOLOGNA

Corso di Laurea Triennale in Ingegneria e Scienze Informatiche

Uso del Machine Learning per rilevare siti web generati da DGA

Tesi di laurea in:
Programmazione a Oggetti

Relatore
Prof. Mirko Viroli

Candidato
Simone Collorà

Correlatore
Dott. Gianluca Aguzzi

I Sessione di Laurea
Anno Accademico 2024-2025

Sommario

Max 2000 characters, strict.

Indice

Elenco delle figure

Capitolo 1

Introduzione

La sicurezza informatica è un argomento di crescente importanza nel mondo moderno. Con il passare del tempo, i sistemi di protezione sono diventati sempre più sofisticati e potenti ma, allo stesso tempo, anche gli hackers hanno sviluppato tecniche sempre più avanzate per eludere i sistemi di protezione. Tra queste vi è sicuramente l'uso di Botnets dei Command and Control (C&C) servers. I C&C sono dei server che manipolano i computer infetti da malwares, i Botnets, permettendo all'attaccante di eseguire codice malevolo da remoto. Il malware, però, deve conoscere un indirizzo IP o un dominio per contattare il server. L'attaccante potrebbe inserire in modo bruto l'indirizzo IP del server nel codice del malware, ma questo metodo è facilmente rilevabile e bloccabile. Gli hackers, quindi, preferiscono utilizzare dei domini generati in modo pseudo casuale per nascondere i loro server chiamati Domain Generation Algorithm (DGA) servers. Negli ultimi anni, sono stati sviluppati vari metodi di Machine Learning per rilevare questi domini.

Struttura della tesi Il lavoro di tesi si dividerà in 4 capitoli. Nel primo capitolo verrano descritti concetti di base riguardanti DGA e Botnets, e Machine Learning con una breve descrizione delle reti neurali e di alcuni algoritmi usati. Nel secondo capitolo verrà descritto il progetto, i suoi obiettivi e i risultati ottenuti. Nell'ultimo capitolo verranno discusse le conclusioni e i possibili sviluppi futuri.

Capitolo 2

Background

Di seguito sono descritti i concetti base su cui si basa il progetto.

2.1 Botnets e DGA

2.1.1 Botnets

I Botnets sono reti di computer infetti da malware, chiamati bot, che possono essere controllati da un attaccante, il botmaster. La vita di un botnet nella maggior parte dei casi è divisa in 4 fasi:

1. **Infezione e propagazione**: Questo è il primo passaggio. L'attaccante cerca di infettare un computer tramite vari metodi come email con link malevoli o Peer to Peer (P2P) sharing. Una volta infettato un dispositivo, il malware cerca di infettare altri dispositivi nella rete.

2. **Rallying**: i bots cercano di contattare per la prima volta il server C&C per far capire all'attaccante che l'attacco è andato a buon fine.

3. **Commands and Reports**: il malware esegue le istruzioni ricevute dal server C&C e invia i risultati al botmaster. I bots ascoltano continuamente il server C&C o si connettono ad esso periodicamente. Appena ricevono un comando lo eseguono, inviano i risultati al botmaster e aspettano un nuovo comando.

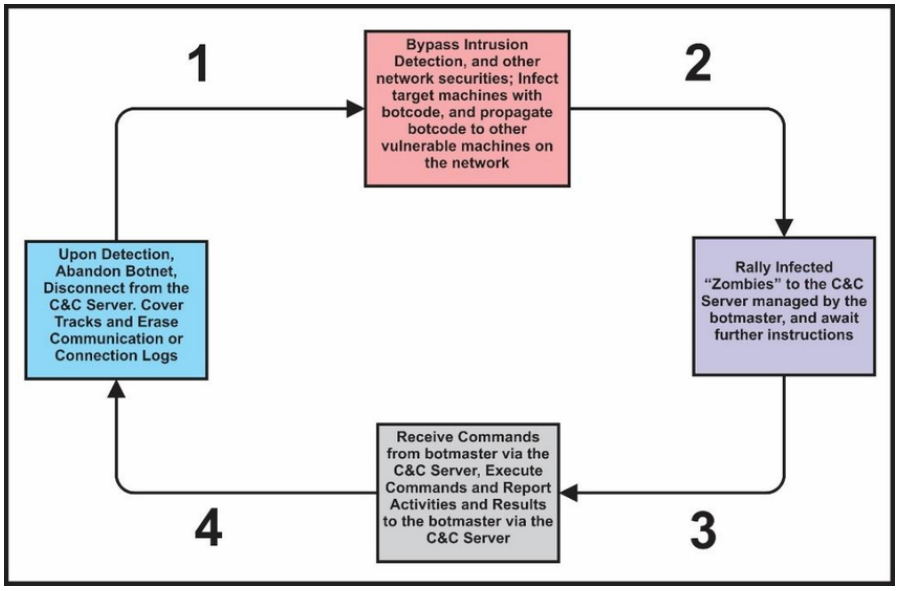

Figura 2.1: Ciclo di vita di un botnet [6]

4. **Abbandono**: Quando un bot non è più utile o utilizzabile, il botmaster può decidere di abbandonarlo. Il botnet, invece, sarà completamente distrutto quando tutti i bot saranno abbandonati o bloccati dalla vittima o quando il C&C server verrà bloccato

2.1.2 Command and Control

Il meccanismo del C&C crea un canale di comunicazione tra il botmaster e i bot. Questo è essenziale per il funzionamento del botnet. Ci sono tre tipi di server C&C:

- **Centralizzati**: In questo tipo di server, il botmaster controlla tutti i bot tramite un server centrale. Questo è il metodo più semplice e veloce per controllare i bot ma è anche il più vulnerabile. Se il server centrale viene bloccato, tutti i bot non possono più ricevere comandi. Questo a sua volta è diviso in due categorie:
  - **IRC** : Internet Relay Chat (IRC) è un sistema di chat usato per comunicare tra i bot e il botmaster in tempo reale. Questo era più usato

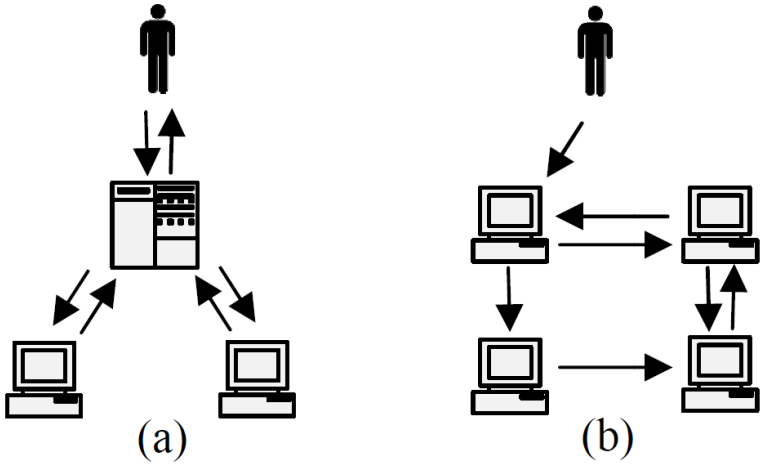

Figura 2.2: Esempio di server C&C. (a) centralizzato, (b) decentralizzato [5]

nella prima generazione di botnet. I bot si connettono al server IRC e aspettano i comandi dal botmaster. I bot seguono un approccio PUSH ovvero quando un bot si connette ad un determinato canale, esso rimane connesso.

- **HTTP**: Il più usato. Con questa tecnica, i bot usano un URL o IP per contatattare il server C&C. Qua invece i bot seguono un approccio PULL. I bot si connettono al server C&C periodicamente e controllano se ci sono nuovi comandi. Questo processo va ad intervalli regolari definiti dal botmaster.

- **Decentralizzati**: Questo tipo di C&C è basato su un sistema P2P senza un server centrale. In questo modo, computer infetti fanno sia da bot che da server C&C. Questo metodo è più difficile da rilevare ma anche più complesso da implementare [8].

2.1.3 Domain Generation Algorithm

I DGA sono algoritmi che generano migliaia di domini in modo pseudo casuale. Prima viene scelto un seed, di solito la data odierna o anche le previsioni meteo [1] e, tramite un algoritmo di hashing, vengono generati i domini. Questi domini

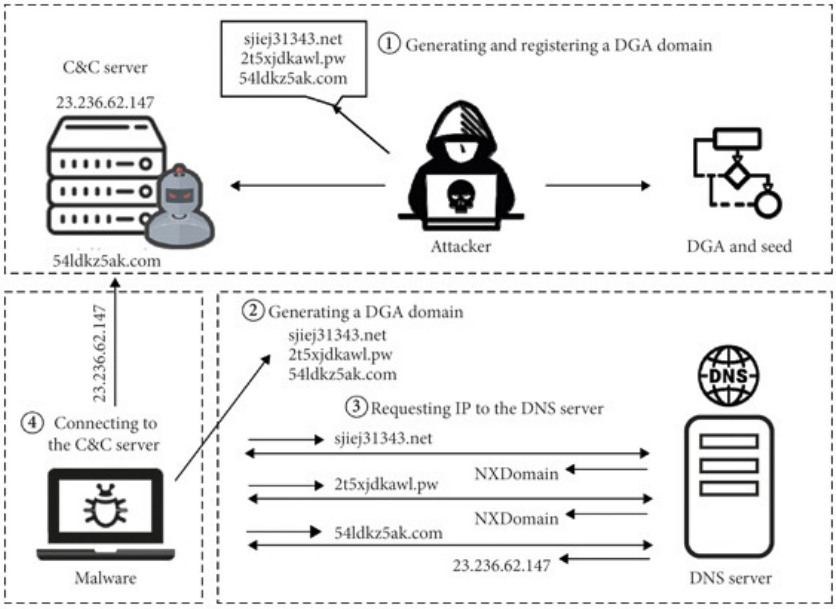

Figura 2.3: Esempio del funzionamento di un DGA [4]

vengono poi utilizzati per contattare i server C&C. Non tutti i domini generati però sono registrati. Il computer infetto, tramite i DNS locali, cercherà di tradurre un dominio in un indirizzo IP. Se non riesce a contattarlo con un determinato dominio, proverà con il successivo finché non troverà un dominio valido che permetterà al malware di comunicare con il server C&C [2]. In questo modo, diventa più difficile per i sistemi di protezione rilevare e bloccare i loro attacchi. Si potrebbe pensare di bloccare direttamente i domini tramite una blacklist ma questo metodo risulta inefficace poiché vengono generati migliaia di domini continuamente. Si pensi che Conficker C, un famoso malware che utilizza DGA, è in grado di generare fino a 50.000 domini pseudo casuali al giorno [3].

Un altro modo per contrastare ciò potrebbe essere quello di fare reverse engineering del DGA per capire quale seed viene utilizzato per generare i domini. Questo però risulta lento e dispendioso e possibilmente inefficace [4].

Per contrastare i DGA, sono stati sviluppati vari metodi di Machine Learning in grado di rilevare i domini generati. Questi metodi hanno due lati poisitivi:

- Non richiedono un lungo processo di reverse engineering.

- Essendo l'AI una blackbox, è molto difficile per gli hackers eseguire un reverse engineering del modello.

2.2 Machine Learning

Il Machine Learning è una branca dell'informatica che punta a far ragionare le macchine come gli esseri umani, ovvero a svolgere compiti autonomamente senza essere programmati esplicitamente e migliorando le loro prestazioni con l'esperienza e i dati.

2.2.1 Reti Neurali

Una **Rete Neurale** o in inglese **Artificial Neural Network** (ANN) è un modello matematico che mira a simulare il funzionamento del cervello umano [10]. Il cervello umano è composto da miliardi di neuroni che comunicano tra di loro tramite sinapsi. Con le reti neurali artificiali, il funzionamento è analogo. A livello matematico, un neurone artificiale è composto principalmente da due componenti:

- **Pesi**: i pesi(weight in inglese) sono valori numerici che aiutano ogni nodo della rete neurale a determinare l'importanza di un input. Usando i pesi, il neurone può decidere se un input è importante o meno.

- **Funzione di attivazione**: la funzione di attivazione del neurone è la funzione che in input prende la sommatoria dei dati pesati con i pesi descritti in precedenza e produce un output che verrà poi inviato ad altri neuroni come input. Alcune delle funzioni di attivazione più comuni sono la funzione sigmoide e la funzione ReLU.

A sua volta una rete neurale è composta da vari strati:

- **Input Layer**: il primo strato della rete neurale e quello che riceve l'input esterno. Poiché la rete neurale è un modello matematico, l'input dovrà essere un vettore numerico. Questo vale anche se l'input da esaminare è una stringa di caratteri, come nel nostro caso.

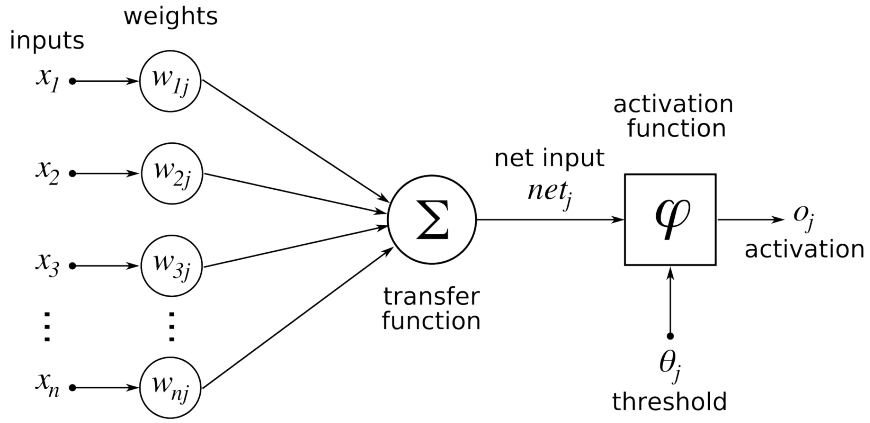

Figura 2.4: Esempio di neurone artificiale [13]

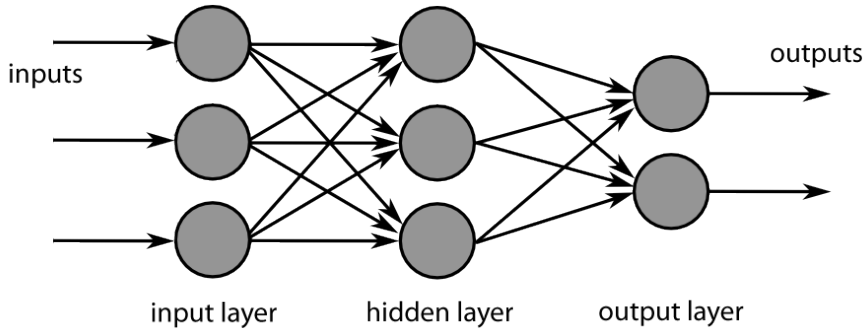

Figura 2.5: Esempio di rete neurale [14]

- **Hidden Layer**: questo è lo strato intermedio della rete neurale in cui avviene il processo di apprendimento. Questo strato elabora gli input ricevuti dallo strato precedente modificando i pesi. Si possono avere anche più hidden layers
- **Output Layer**: Questo è l'ultimo strato della rete neurale. Fornisce i risultati finali ottenuti dalla rete neurale. Questo strato può essere composto anche da un solo neurone che fornisce un output binario (0 o 1) come sarà nel nostro caso(DGA o non DGA)

Abbiamo inoltre vari tipi di reti neurali:

- **Reti Neurali Feedforward**: Una rete Feedforward elabora le informazioni in un solo verso. Non ha né cicli né memoria delle informazioni passate.

Questo tipo di rete è usato principalmente per riconoscimento di pattern o classificazioni

- **Reti Neurali Ricorrenti (RNN)**: A differenza di una rete Feedforward, in una RNN i neuroni possono andare anche a formare dei cicli. Questo permette alla rete di avere una specie di memoria e quindi di ricordare le informazioni passate. Usata per esempio per traduzione automatica o riconoscimento vocale.

- **Reti Neurali Convoluzionali (CNN)**: Questo tipo di rete è usato principalmente per dati strutturati a griglia ad esempio le immagini. Il nome "Convoluzionali" deriva dal fatto che la rete usa un'operazione matematica chiamata proprio convoluzione. Le CNN sono usate soprattuto nell'ambito della visione artificiale per il riconoscimento di oggetti [17].

2.2.2 Sviluppo di una rete neurale

Il primo passo per sviluppare una rete neurale è quello di raccogliere i dati in un dataset. Nel nostro caso, dovendo riconoscere se un dominio è lecito o DGA, il dataset conterrà entrambi i tipi di domini che potranno avere un etichetta, DGA o legit nel nostro caso, o no. Questi come detto in precedenza, dovranno essere convertiti in un vettore numerico. A seconda di come è strutturato il dataset possiamo avere vari tipi di allenamento:

- **Supervised Learning**: È la tecnica più comune per allenare le reti neurali [11]. In questo tipo di apprendimento, il modello viene addestrato su un dataset etichettato.

- **Unsupervised Learning**: In questo tipo di apprendimento, il modello, deve scoprire dei pattern o delle relazioni senza avere nessuna etichetta. Il modello deve trovare degli oggetti che condividono delle caratteristiche simili, chiamati cluster

- **Reinforcement Learning**: In questo tipo di apprendimento, ogni azione ha un effetto nell'ambiente che può essere positivo o negativo.

Si è soliti dividere il dataset in tre parti ovvero training set, validation set e test set. Il **training set** è il dataset su cui viene addestrato il modello, il **validation set** è una parte del training set utile per riconoscere se il modello è in grado di generalizzare su dati nuovi ovvero non va in overfitting o underfitting. il **test set**, invece, è il dataset su cui viene testata la precisione del modello. Per la divisione del dataset non esiste una regola fissa, ma solitamente il training set è più grande del validation set e del test set. Un esempio potrebbe essere 70% training set, 15% validation set e 15% test set. Successivamente viene deciso il tipo di rete neurale da usare, il numero di layers e il numero di neuroni per ognuno di essi. Una rete troppo semplice potrebbe non essere in grado di apprendere i pattern nei dati (**Underfitting**), mentre al contrario, una rete troppo complessa potrebbe imparare troppo bene di dati e non generalizzare su dati nuovi (**Overfitting**). Come detto in precedenza, il validation set ci aiuta a capire se il modello va in overfitting o underfitting. Successivamente vengono inizializzati i pesi di ogni neurone. Questi possono essere inizializzati in modo casuale o con altre tecniche più avanzate ad esempio la **He Initialization** [19] (opzione di default quando si inizializza con attivazione ReLU). Dopo di che si passa alla fase vera e propria di addestramento. Il modello viene iterato per un certo numero di epoche su un batch di dati. Un **batch** è un sottoinsieme del training set dopo il quale il modello aggiorna i propri pesi. Ad esempio abbiamo un training set di 10000 dati e un batch size di 100. Ogni 100 dati il modello calcolerà l'output e la loss function, cioè la funzione che calcola la differenza tra l'output previsto e quello reale, e aggiornerà i pesi. Il numero di epoche è un iperparametro che indica quante volte l'algoritmo di apprendimento deve passare attraverso il dataset di training[20]. Infine viene testato il modello sul test set e, se la precisione è soddisfacente, il modello potrà essere usato.

2.3 Algoritmi di Machine Learning adatti al nostro problema

Ora che abbiamo visto i concetti di base del Machine Learning, delle reti neurali e dei DGA, vediamo alcuni degli algoritmi che possono essere usati per rilevare i

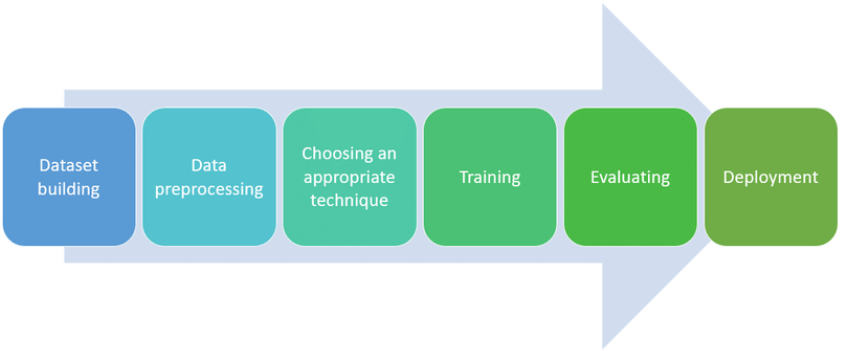

Figura 2.6: Pipeline di un modello di Machine Learning [21]

domini generati da DGA.

2.3.1 Random Forest

Random Forest è un algoritmo esamble (cioè unisce più modelli) di Machine Learning supervisionato che usa un insieme di alberi decisionali per classificare i dati. Un albero decisionale è un modello che usa appunto una struttura ad albero per prendere decisioni. Un esempio può essere quello nella fig. 2.7, in cui, a seconda di determinate caratteristiche, l'albero decide se una persona rischia di avere un attacco cardiaco o meno. Il problema principale del singolo albero decisionale è che è molto suscettibile all'overfitting. Il random forest risolve questo problema aumentando l'accuratezza del modello sia in fase di training che in fase di testing. [23] Random Forest usa un apporccio basato su feature, ovvero vengono selezionate delle caratteristiche del dataset che possono essere utili per la classificazione. Ad esempio, nel nostro caso, le caratteristiche potrebbero essere il numero di vocali, lunghezza del dominio o l'entropia. L'algoritmo funziona nel seguente modo: Per prima cosa vengono creati n alberi decisionali, ognuno contenente una parte casuale del training set. Inoltre ogni albero decisionale viene addestrato su un sottoinsieme casuale di feature. Dopo di che, ogni albero fa la sua previsione, e la previsione viene fatta in base a ciò che la maggior parte degli alberi ha previsto. Qualora si trattasse di un problema di regressione, la previsione finale sarà la media delle previsioni di tutti gli alberi.

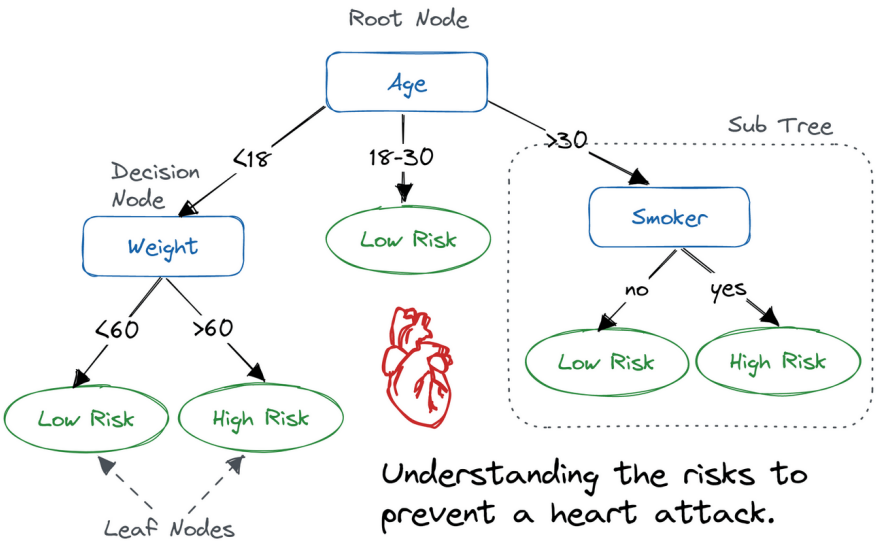

Figura 2.7: Esempio di albero decisionale [22]

2.3.2 Long Short Term Memory (LSTM)

Le LSTM sono un tipo di rete neurale ricorrente (RNN). A differenza del Random Forest, le LSTM usano un approccio featureless e imparano direttamente dai dati. Questo lo rende più adatto con un dataset più grande e complesso.

Capitolo 3

Progetto

3.1 Obiettivi

Il progetto ha come obiettivo quello di sviluppare un sistema di rilevamento di domini generati da DGA tramite l'uso di tecniche di Machine Learning.

Per il dataset di addestramento ho usato un dataset fornito dall'azienda Flashstart che contiene circa 5600 domini generati da DGA. A questo ho aggiunto un dataset di circa 5600 domini benigni presi dal database di Cloudflare aggiornato al 12 maggio 2025[16]

3.2 Some cool topic

Capitolo 4

Contribution

You may also put some code snippet (which is NOT float by default), eg: capitolo 4.

4.1 Fancy formulas here

```
public class HelloWorld {
  public static void main(String[] args) {
    // Prints "Hello, World" to the terminal window.
    System.out.println("Hello, World");
  }
}
```

Bibliografia

[1] R. Sivaguru, C. Choudhary, B. Yu, V. Tymchenko, A. Nascimento, and M. D. Cock, "An evaluation of dga classifiers," in *2018 IEEE International Conference on Big Data (Big Data)*, 2018, pp. 5058–5067.

[2] B. Yu, J. Pan, J. Hu, A. Nascimento, and M. De Cock, "Character level based detection of dga domain names," in *2018 International Joint Conference on Neural Networks (IJCNN)*, 2018, pp. 1–8.

[3] G. Alley-Young, "Conficker worm," in *The Handbook of Homeland Security*. CRC Press, 2023, p. 175.

[4] J. Namgung, S. Son, and Y.-S. Moon, "Efficient deep learning models for dga domain detection," *Security and Communication Networks*, vol. 2021, no. 1, 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1155/2021/8887881

[5] M. Eslahi, R. Salleh, and N. B. Anuar, "Bots and botnets: An overview of characteristics, detection and challenges," in *2012 IEEE International Conference on Control System, Computing and Engineering*, 2012, pp. 349–354.

[6] E. Ogu, N. Vrakas, C. Ogu, and A.-I. B.M., "On the internal workings of botnets: A review," *International Journal of Computer Applications*, vol. 138, pp. 975–8887, 04 2016.

[7] X. Ma, X. Guan, J. Tao, Q. Zheng, Y. Guo, L. Liu, and S. Zhao, "A novel irc botnet detection method based on packet size sequence," in *2010 IEEE International Conference on Communications*, 2010, pp. 1–5.

[8] M. Bailey, E. Cooke, F. Jahanian, Y. Xu, and M. Karir, "A survey of botnet technology and defenses," in *2009 Cybersecurity Applications and Technology Conference for Homeland Security*, 2009, pp. 299–304.

[9] A. L. Samuel, "Some studies in machine learning using the game of checkers," *IBM Journal of Research and Development*, vol. 3, no. 3, pp. 210–229, 1959.

[10] J. Zou, Y. Han, and S.-S. So, "Overview of artificial neural networks," *Artificial neural networks: methods and applications*, pp. 14–22, 2009.

[11] T. O. Ayodele, "Types of machine learning algorithms," *New advances in machine learning*, vol. 3, no. 19-48, pp. 5–1, 2010.

[12] E. Grossi and M. Buscema, "Introduction to artificial neural networks," *European journal of gastroenterology & hepatology*, vol. 19, no. 12, pp. 1046–1054, 2007.

[13] W. Commons, "File:artificialneuronmodel english.png — wikimedia commons, the free media repository," 2024, [Online; accessed 13-maggio-2025]. [Online]. Available: https://commons.wikimedia.org/w/index.php?title=File:ArtificialNeuronModel_english.png&oldid=840034703

[14] ——, "File:multilayerneuralnetwork english.png — wikimedia commons, the free media repository," 2020, [Online; accessed 13-May-2025]. [Online]. Available: https://commons.wikimedia.org/w/index.php?title=File:MultiLayerNeuralNetwork_english.png&oldid=481159061

[15] M. Zakaria, A. Mabrouka, and S. Sarhan, "Artificial neural network: a brief overview," *neural networks*, vol. 1, p. 2, 2014.

[16] Cloudflare. (2025) Top domains - cloudflare radar. Cloudflare Inc. Accessed: 14 May 2025. [Online]. Available: https://radar.cloudflare.com/domains

[17] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[18] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 11 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[20] J. Brownlee, "What is the difference between a batch and an epoch in a neural network," *Machine learning mastery*, vol. 20, no. 1, pp. 1–15, 2018.

[21] B. Hamza, "The comprehensive programming language model," Ph.D. dissertation, École Nationale Supérieure d'Informatique, 07 2021.

[22] "Decision tree classification in python," 2025, accessed: 14 May 2025. [Online]. Available: https://www.datacamp.com/tutorial/decision-tree-classification-python

[23] T. K. Ho, "Random decision forests," in *Proceedings of 3rd International Conference on Document Analysis and Recognition*, vol. 1, 1995, pp. 278–282 vol.1.

Ringraziamenti

Con questo lavoro di tesi, si conclude il mio percorso di studi sicuramente non facile ma che mi ha dato la possibilità di crescere sia a livello personale che professionale.
Ringrazio il mio relatore, il Prof. Mirko Viroli e il mio correlatore Dott. Gianluca Aguzzi per essere stati sempre disponibili e avermi aiutato con il lavoro e Flashstart per avermi dato la possibilità di lavorare su questo progetto.
Ringrazio la mia famiglia per avermi sempre supportato sia moralmente che economicamente. Senza di loro non sarei mai riuscito a raggiungere questo traguardo.
Ringrazio i miei amici in particolare l'associazione Sprite che hanno reso questo percorso più divertente. Ringrazio i bloccati e gli amici di Smash Bros per i momenti di divertimento e avermi dato un hobby per staccare dallo studio.