Abdullah Al Noman Chowdhury

Machine Learning and Statistical Methods for GNSS Spoofing Detection: A Systematic Review and Deep Reinforcement Learning Simulation

School of Technology and Innovations
Master's Thesis
Sustainable and Autonomous systems

Vaasa 2025

UNIVERSITY OF VAASA
School of Technology and Innovations
Author: Abdullah Al Noman Chowdhury
Title of the thesis: Machine Learning and Statistical Methods for GNSS Spoofing Detection: A Systematic Review and Deep Reinforcement Learning Simulation
Degree: Master of Science in Computing Sciences
Discipline: Sustainable and Autonomous Systems
Supervisor: Mohammed Elmusrati
Instructor and Second Evaluator: Heidi Kuusniemi
Instructor: Elham Ahmadi
Year: 2025 **Pages:** 126

ABSTRACT:

Global Navigation Systems (GNSS) is essential for global navigation, positioning, and timing across sectors such as transportation, telecommunications, and critical infrastructure. However, the weak, unencrypted nature of GNSS signals makes them highly vulnerable to spoofing attacks, posing serious threats to system reliability and safety. This thesis begins with an introduction to reinforcement learning (RL), providing foundational context on Markov decision processes, model-free and model-based RL, and deep Q-network (DQN) algorithms. Building upon this, it presents a detailed technical review of Global Navigation Satellite System (GNSS) signal architecture, positioning mechanisms, and spoofing vulnerabilities. The widespread reliance on unencrypted, low-power satellite signals renders GNSS systems highly susceptible to spoofing and jamming attacks.

To understand the current state of research, a systematic literature review (SLR) was conducted on AI-based GNSS spoofing and jamming detection strategies. The review identified that while machine learning and deep learning techniques are extensively used, reinforcement learning remains underrepresented. Most prior works focused on binary classification or simulation-based datasets, with limited attention to real-world data.

This thesis also develops and evaluates a multiclass spoofing detection framework using a DQN agent trained on the TEXBAT dataset. Nine spoofing-sensitive features were extracted across eight PRNs per snapshot, forming a 9-dimensional input vector. A stateless OpenAI Gym environment was designed to classify each signal instance as either clean or one of six spoofing scenarios. The model was implemented using Stable-Baselines3 with a Multilayer Perceptron Policy architecture and trained over 80,000 timesteps with ε-greedy exploration and experience replay.

On the test set of 28,391 samples, the model achieved an overall accuracy of 87.6%. Classification accuracy was 90% for ds1, 97% for ds2, 95% for ds3, and 98% for ds4, with F1-scores ranging from 0.922 to 0.979. For more sophisticated attacks, ds7 (sparse stealth) was correctly classified in 81% of cases (F1 = 0.725), and ds8 (replay attack) achieved 71% accuracy (F1 = 0.763). The clean signal class was identified with 81% accuracy, 0.920 precision, and 0.810 recall. These results demonstrate the effectiveness of reinforcement learning in detecting diverse GNSS spoofing attacks using real world tracking features.

KEYWORDS: GNSS Spoofing & Jamming, Deep Reinforcement Learning, Systematic Literature Review, Deep Q-Network (DQN), TEXBAT Dataset, Multi-Class Classification, GNSS Signal Vulnerabilities, Stable-Baselines3.

Contents

Figures

Tables

Abbreviations

RL	Reinforcement Learning
GNSS	Global Navigation Satellite Systems
DQN	Deep Q-Network
TEXBAT	Texas Spoofing Test Battery
PNT	Positioning, Navigation, and Timing
MEO	Medium Earth Orbit
AI	Artificial Intelligence
ML	Machine Learning

DL	Deep Learning
PRN	Pseudorandom Noise
CPS	Cyber-Physical Systems
IoT	Internet of Things
MDP	Markov Decision Process
MCTS	Monte Carlo Tree Search
MPC	Model Predictive Control
MBPO	Model-Based Policy Optimization
MB-MPO	Model-Based Maximum Posteriori Policy Optimization
CNN	Convolutional Neural Networks
TD	Temporal Difference
PVT	Position, Velocity, and Time
GPS	Global Positioning System
FDMA	Frequency Division Multiple Access
CDMA	Code Division Multiple Access
BPSK	Binary Phase Shift Keying
QPSK	Quadrature Phase Shift Keying
TDOA	Time-Difference-of-Arrival
SBAS	Satellite-Based Augmentation Systems
DGNSS	Differential Global Navigation Satellite Systems
EGNOS	European Geostationary Navigation Overlay Service
BDSBAS	BeiDou Satellite-Based Augmentation Systems
SDCM	System for Differential Corrections and Monitoring
SDR	Software-Defined Radio
SNR	Signal-to-Noise Ratio
SCER	Security Code Estimation and Replay
SLR	Systematic Literature Review
DT	Decision Trees
RF	Random Forest
SVM	Support Vector Machines

GLONASS	Global Navigation Satellite System
NN	Neural Network
MLP	Multilayer Perceptron
DNN	Deep Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
BiRNN	Bidirectional Recurrent Neural Network
AFRL	Adaptive Federated Reinforcement Learning
UAV	Unmanned Aerial Vehicle
MSPE	Mean Squared Prediction Error
STFT	Short Time Fourier Transform

1 Introduction

This thesis begins by introducing the theoretical foundations and relevance of Reinforcement Learning (RL) for dynamic detection tasks such as Global Navigation Satellite Systems (GNSS) spoofing classification. It then presents a systematic literature review of AI-based methods for GNSS spoofing and jamming detection, highlighting the dominance of supervised techniques and the underutilization of RL approaches. Building on these insights, the thesis develops a Deep Q-Network (DQN)-based spoofing detection framework trained on real-world scenarios from the Texas Spoofing Test Battery (TEXBAT) dataset.

1.1 Background and Motivation

Global Navigation Satellite Systems (GNSS) are an essential part of our daily lives, operating by giving positioning, navigation, and timing (PNT) services to much of our technology. GPS (United States), Galileo (Europe), GLONASS (Russia), and BeiDou (China) are systems that relay signals via satellites that are used in a variety of functions that include air transportation and maritime activities, power grid synchronization, financial exchange, and the recently expanding autonomous vehicles sector. In the recent decades, the number of applications of GNSS has expanded tremendously due to its superior precision, global coverage, and the fact that civilian users can access it freely.

These satellite systems constitute Medium Earth Orbit (MEO) and constantly transmit timing and navigation signal on specified radio frequencies. These signals allow GNSS receivers on the ground to calculate their location and the time of day by measuring the duration that the signals take to reach them at least four different satellites. A major issue however is that when these signals get to the Earth they are very weak, typically between −130 and −160 dBW, which means they are vulnerable to radio frequency interference (RFI)(Wang et al., 2021). To add to that, since the majority of civilian GNSS signals are unencrypted and openly published, an attacker can recreate them with relatively cheap hardware and software-defined radios. Such a combination of low signal

strength and open specifications makes GNSS systems exceptionally vulnerable to deliberate interference in the form of spoofing and jamming.

Rising Interference Incidents in recent years, numerous incidents have demonstrated that GNSS jamming and spoofing are not merely theoretical concerns but active problems, especially in regions marked by geopolitical tension. A striking example comes from northern Europe, where Finland and the broader Nordic region have experienced a surge in GNSS interference incidents during 2023–2024. According to Finland's Transport and Communications Agency (Traficom), reported GPS disturbances affecting aviation in Finland skyrocketed to about 2,800 incidents in 2024, up from only 200 incidents in 2023(*USU*, 2025a). These disruptions have frequently forced pilots to abandon satellite-based landing procedures at least three eastern Finnish airports had to revert to old-fashioned radio navigation systems to guide landings, and on multiple occasions flights were unable to land due to jamming and had to return to their origin(*USU*, 2025b).

Similar trends have been observed across the broader Nordic skies. In northern Scandinavia (encompassing areas of Finland, Norway, and Sweden), the frequency of GNSS jamming reported by civil aviation has steadily climbed airliners experienced jamming on 18 days in 2021, 122 days in 2022, and as many as 294 days in 2023(Braun et al., 2025). By late 2023 and into 2024, interference was being recorded nearly every day in parts of northern Scandinavia. These real-world cases underscore the pressing need for effective GNSS interference detection and mitigation. What was once considered a niche security issue is now a frequent hazard that can disrupt transportation and pose risks to safety. In the Finnish and Baltic context, GNSS jamming has been used as a form of electronic warfare or geopolitical signaling, revealing how GNSS vulnerabilities can be exploited at scale. The imperative is clear robust methods to detect, and ultimately thwart, spoofing and jamming must be developed and deployed to protect the integrity of GNSS reliant systems in an increasingly adversarial signal environment.

1.2 Problem Statement

Although the critical role of Global Navigation Satellite Systems (GNSS) in modern infrastructure is well recognized, the development of robust and deployable spoofing detection mechanisms remains a significant research and engineering challenge. Research efforts to detect and mitigate GNSS spoofing have accelerated in recent years, particularly through the application of Artificial Intelligence (AI) techniques. Machine Learning (ML) and Deep Learning (DL) have been employed to recognize signal anomalies, classify attack types, and enhance detection accuracy. However, Reinforcement Learning (RL) a framework that enables agents to learn decision policies through interaction with an environment remains largely underutilized in this context.

Accordingly, this research addresses these deficiencies through a dual approach. First, it contributes a structured review of the AI/ML/RL and Statistical landscape in GNSS interference detection, mapping out methodologies, datasets, and performance metrics used across studies, while identifying key limitations. Second, it proposes and implements a novel spoofing detection framework based on Deep Q-Networks (DQN), trained on real-world spoofing scenarios from the TEXBAT dataset. The model leverages raw GNSS tracking features across multiple PRNs to classify spoofing types in a multi-class setting. Together, the literature review and practical implementation demonstrate both the current challenges and the emerging feasibility of applying RL methods to GNSS spoofing detection in safety-critical systems.

1.3 Research Objectives

This thesis aims to bridge this gap by combining a systematic investigation of the existing literature with the development of a RL-based spoofing detection framework. The specific research objectives are as follows:

1. To introduce the theoretical foundations and practical relevance of Reinforcement Learning, outlining its key principles, learning architectures, and

applicability to signal classification and detection problems. This includes a detailed discussion of the Deep Q-Network (DQN) algorithm as a model-free method suitable for adaptive spoofing detection tasks.

2. To conduct a systematic literature review (SLR) of peer-reviewed studies published between 2015 and 2025 that employ AI, Machine Learning (ML), or RL for GNSS spoofing and jamming detection or mitigation. The review addresses four research questions:

    a) What AI/ML/RL and other techniques are applied to GNSS spoofing or jamming detection.
    b) What specific attack types (e.g., spoofing, jamming, hybrid) are targeted?
    c) What datasets, and evaluation metrics are used across studies?
    d) What are the key trends, limitations in this domain?

3. To analyze the current research landscape and identify underrepresented areas, with a focus on the scarcity of RL-based methods. The SLR findings show that while supervised ML and DL dominate, RL has only been applied in two studies, highlighting a significant gap in the development of autonomous, policy-driven detection models.

4. To develop RL-based GNSS spoofing detection framework, implementing a Deep Q-Network trained on real-world spoofing scenarios from the TEXBAT dataset. The model is designed to classify multiple spoofing types based on raw GNSS tracking features extracted from multiple PRNs. This contribution demonstrates RL's potential for adaptive spoofing recognition in complex, noisy environments.

5. To contribute a replicable proof-of-concept that highlights the feasibility of applying RL to GNSS interference detection and lays the groundwork for future

research into real-time detection and active mitigation strategies in safety-critical infrastructure systems.

1.4 Thesis Structure

This thesis is organized into seven chapters, each building sequentially to address the research objectives of exploring reinforcement learning for GNSS spoofing detection through both a comprehensive literature review and the implementation of a practical case study.

Chapter 1 introduces the topic of the research and begins by motivation of the topic through the recent real-world cases of GNSS interference in northern Europe, especially in Finland. It gives the statement of the problem, develops the objectives of the research and gives a thesaurus of the expected structure of the thesis.

Chapter 2 provides a theoretical foundation on Reinforcement Learning. It introduces the taxonomy of ML approaches supervised, unsupervised, and reinforcement learning and elaborates on the core concepts of RL including Markov Decision Processes (MDPs), environment modeling, agent interaction, and reward functions. The chapter presents the mathematical formulation of RL, key algorithmic categories, and highlights deep reinforcement learning (DRL) and DQN as pivotal methods.

Chapter 3 gives the technical background required on Global Navigation Satellite Systems (GNSS), their system architecture, the nature of signals and augmentation mechanisms. It also explores GNSS vulnerabilities, (specifically jamming and spoofing) and elaborates on the danger to critical infrastructure. The chapter ends with spoofing activities in Finland and the Nordic region in general as an example of the operational need of secure GNSS.

Chapter 4 presents the systematic literature review (SLR) on existing AI/ML/RL-based methods for GNSS spoofing and jamming detection and mitigation. It details the

research questions, search strategy, inclusion/exclusion criteria, and quality assessment process. The review findings are synthesized to identify methodological trends, commonly used datasets and metrics.

Chapter 5 discusses the literature review findings in depth. It categorizes the identified studies by machine learning techniques, including classical models, neural networks, ensemble methods, reinforcement learning approaches, and statistical techniques. The chapter also analyzes the studies based on the type of GNSS threats addressed spoofing, jamming, or both and concludes by summarizing trends and limitations observed in the research landscape.

Chapter 6 introduces the RL-based simulation developed for GNSS spoofing detection. It begins with a description of the TEXBAT dataset and the spoofing scenarios it contains. The chapter then outlines the selected feature set, preprocessing steps, and data partitioning. A custom RL environment is designed and integrated with Stable-Baselines3, including observation & action space definitions, reward mechanism, and stateless environment formulation. The DQN model architecture, training setup, and evaluation strategy are described in detail. It also presents the evaluation of the trained RL model using a comprehensive set of multi-class classification metrics. The performance is analyzed through confusion matrices and per-class precision, recall, and F1-scores. The results are discussed in the context of the model's ability to generalize across spoofing types and the practical feasibility of RL in GNSS interference detection.

Chapter 7 concludes the thesis by summarizing the key contributions, reflecting on how the research objectives were addressed, and identifying limitations and potential avenues for future work. Emphasis is placed on the practical implications of the findings and the prospective role of RL in enhancing GNSS security in real world deployments.

2 Introduction to Reinforcement Learning

2.1 Overview of Machine Learning

The development of advanced algorithms has greatly accelerated toward accomplishing Artificial intelligent in this era of advancement in computing technology. The field of Artificial intelligence is the development of machines able to imitate certain facets of human thought and decision making. Machine learning (ML) is the well-known branch of AI which does not rely on established rules instead they improve effectiveness as time passes through learning from experiences, that's how they focuses on the evolution of autonomous systems (Naeem et al., 2020).One of the subfield of AI is Machine Learning(ML), which concerns with ''the question of how to develop software agents that improve automatically with experience'' (Naeem et al., 2020).

We can divide Machine learning into mainly three categories which includes, Supervised learning, Unsupervised learning and Reinforcement learning. A standard taxonomy of machine learning has shown in figure 1.

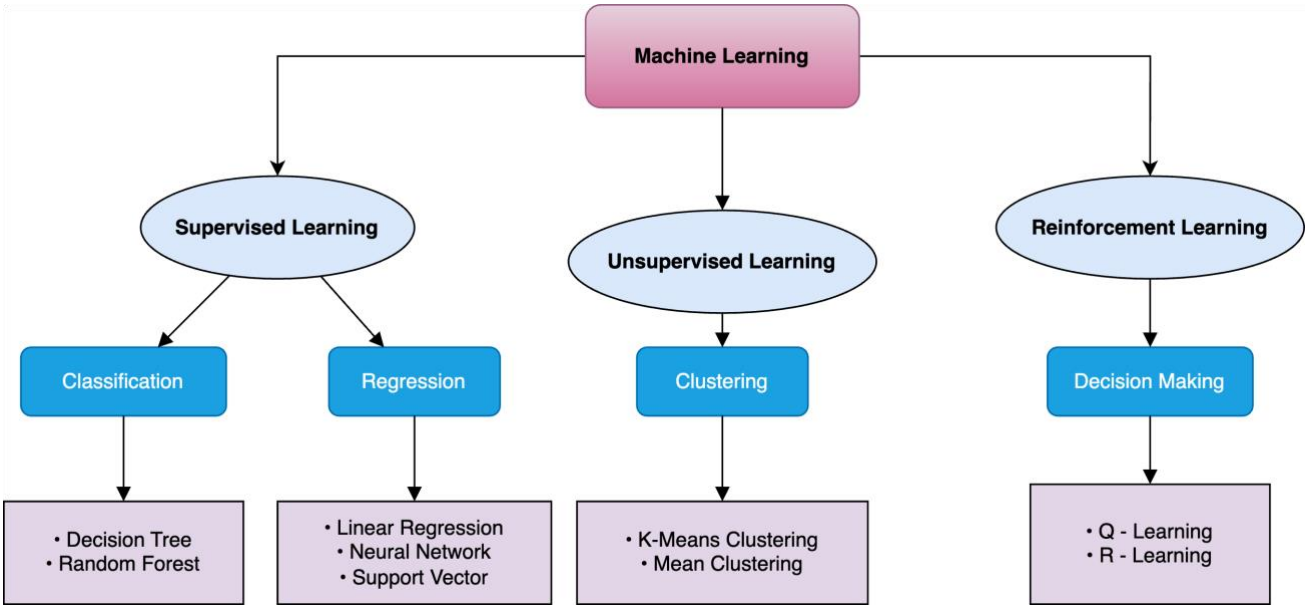

Figure 1. Standard taxonomy of machine learning

Supervised learning involves in learning from training data of labelled examples are given by an authority on the topic assuming the position of external supervisor in the learning process aiming to provide learning agent the understanding to generalize its replies to actual circumstances which is not included in the training set. To discover hidden pattern and information in a dataset without any mistake being made, Unsupervised learning takes control.

Learning from experience is the fundamental concept of a reinforcement learning agent. Reinforcement learning agents learn from their prior failures and successes, just like people do. Basically, the main objective is to maximize a reward signal while learning how to relate states to actions (Morales et al., 1 C.E.). The reward encourages the good actions while the penalty avoids the bad ones.

$$J_{\{k,T\}} = \sum_{i=0}^{T} \gamma^i r_{k+i} \qquad 1$$

Where γ is the discount factor balancing immediate and future rewards, r_{k+i} represents the reward at each step.

2.2 Framework of Reinforcement Learning

Reinforcement Learning is a branch of Machine Learning that studies how agent should behave in a given environment to maximize an idea known as cumulative rewards. RL implies training an agent through trial and error through its interaction with the environment. Instead of training on labelled datasets, RL agents learn by interacting with their surroundings.

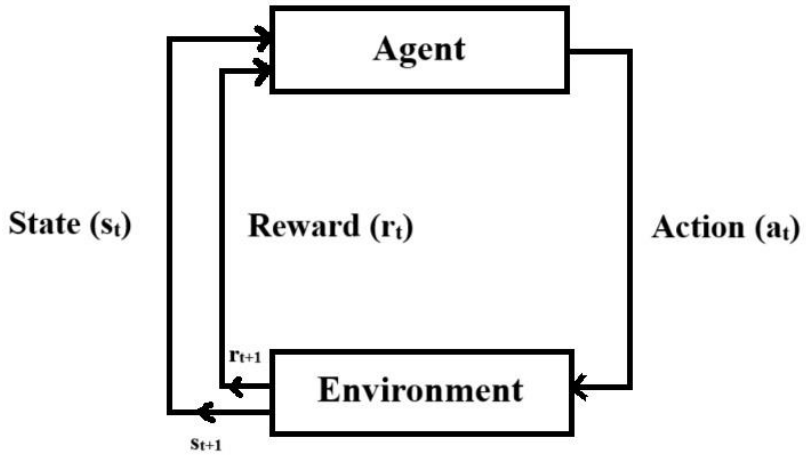

Figure 2. Reinforcement learning framework.

The agent performs an action a_t at each time step s_t, receives an immediate reward and moves to the next state. This process continues until the agent reaches a terminal state, after which the environment resets and a new episode start(Morales et al., 1 C.E.). Because of this, RL is especially well suited for decision making tasks in which the agent must learn long term strategies rather than short term strategies.

This segment explained reinforcement learning's conceptual base through its position in machine learning and its fundamental learning principles and its singularities among other machine learning approaches. The following sections analyze the mathematical methods alongside major algorithmic categories together with contemporary advancements in reinforcement learning research.

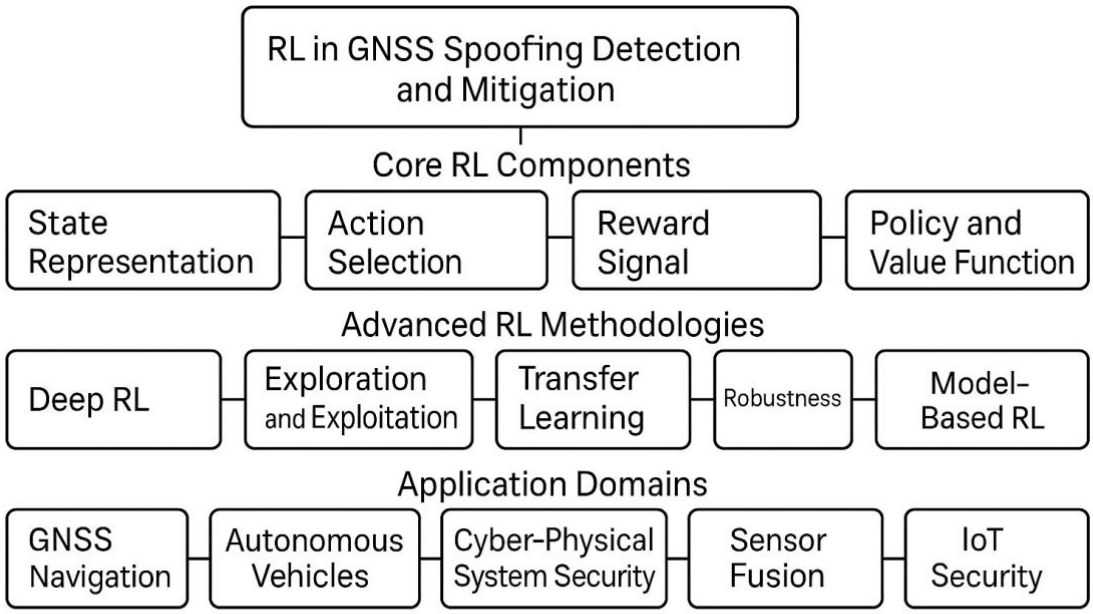

Figure 3. RL core concept in relation with GNSS spoofing detection.

The diagram above illustrates a three-layer conceptual framework for applying Reinforcement Learning (RL) in GNSS spoofing detection and mitigation. This framework highlights the foundational RL concepts, advanced methodological approaches, and the application domains pertinent to achieving secure and resilient navigation systems.

The first layer represents the core RL components:

- State Representation involves encoding GNSS signals and extracted features that describe the agent's perception of its environment.
- Action Selection encompasses the decisions available to the agent, such as choosing detection or mitigation strategies in response to potential spoofing threats.
- Reward Signal is formulated based on detection accuracy and penalties for false alarms or missed attacks, thereby guiding the agent toward optimal behavior.
- Policy and Value Function refer to the internal mechanisms through which the agent determines its actions and estimates future expected rewards.

The second layer introduces advanced RL methodologies that support robust learning in adversarial settings. These include deep reinforcement learning, which leverages neural networks for complex pattern recognition; exploration and exploitation strategies to ensure both adequate learning and optimal performance; transfer learning for applying knowledge across scenarios; robustness measures to counter adversarial attacks; and model-based RL approaches that incorporate environmental modeling for improved planning.

The third layer identifies the key domains where these RL methods are applied, such as GNSS navigation, autonomous vehicles, cybersecurity within cyber-physical systems (CPS), sensor fusion for integrating multiple data sources, and internet of things (IoT) security. By structuring the RL framework in this manner, the diagram clarifies the logical progression from fundamental RL theory to practical deployment in critical navigation and security infrastructures.

2.3 Markov Decision Process (MDP)

The Markov Decision Process (MDP) is the given name of a set of procedures where an agent chooses actions in a typical reinforcement learning case in a discrete time step to maximize cumulative rewards while interacting with an uncertain and fully observable environment. The MDP framework bases its operations on the Markov property that states the future state depends exclusively on present state and actions and ignores earlier states. The prediction of next state occurs independently from all preceding sequence of events. The environment exists in a stationary state because its rules and dynamics maintain consistency throughout time. Chess serves as a perfect illustration of a game where the rules keep unchanged while the best move depends uniquely on the present board layout independent of previous move histories (Naeem et al., 2020). A Markov Decision Process (MDP) offers a mathematical framework for simulating decision-making in circumstances where decisions have some degree of control over the outcome and some degree of randomness. MDP is defined by a set tuple (S,A,P,R, γ)

where,

- S is a finite set of states that representing every possible configuration of the environment.
- A is a finite set of actions that are available to the agent.
- P is the state transition probability function, which satisfy the Markov property:

$$P\left(s_{\{t+1\}} \mid s_1, a_1, s_2, a_2, \ldots, s_t, a_t\right) = P\left(s_{\{t+1\}} \mid s_t, a_t\right) \quad [2]$$

- The reward function $\mathrm{R}: \mathrm{S} \times \mathrm{A} \rightarrow \mathrm{R}$, where $R(s, a)$ defines the direct reward which follows the execution of action a in state s.
- γ $[0,1]$ is the discount factor, which quantifies the importance of future rewards relative to immediate rewards.

a. Environment

Agent operations in reinforcement learning require a formal definition of the environment through which the agent interacts. The development of formal specifications requires defining complete action regulations for agents and specifying environmental states while detailing reward-penalty structures from agent decisions. The RL environment defines the outside system which agents utilize for interaction. Building this environment needs a clear definition of how states shift when agents select their actions while specifying the reward or penalty system which feeds back to agents. The environment operates by receiving both the agent's current state and selected action then generates the following state together with its reward value. An environment operates through different forms that include virtual worlds such as games and useful applications including medical diagnosis systems and domains where agent actions directly modify results. The environment functions as an active setting that enables assessment and improvement of both agent policies and learning processes (Naeem et al., 2020).

b. States

States are the set of every possible state of the environment which agent can observe.

$$S = \{s_1, , s_2, , \dots , , s_n\} \quad 3$$

Every domain of possibilities where an agent could exist within its environment constitutes a state. Within different contexts states take various forms for controllers as specific readings and for robots as particular spatial locations including entire rooms. A state exists in different forms which include coordinates or numbers or letter-based representations. The number of states in an environment varies based on its complexity level from limited to infinite. The terminal states within a system serve as final markers that terminate processes by blocking additional actions or transitions for the agent after reaching them.

c. Actions

$$A = \{a_0, a_1, \dots \dots , a_n\} \quad 4$$

Anything that a robot or an agent can think of or is permitted to do in a given environment is considered an action or series of actions. In a grid environment, for instance, an agent can move left, right, down, up, or remain in the same position. An agent can travel left or right in an environment with a single horizontal line. Generally speaking, an agent can choose any course of action that is feasible in a certain situation.

d. Transition probabilities

How the agents' actions change in the environment is the transition function. The transition function $T(s, a, s\text{^}') = P(s' \mid s, a)$ states that T is the probability of being up in the state s' given that the agent at the moment in state s and taken action a.

e. Reward function (R)

Each state is assigned a specific numerical value by R(s) which represents the immediate feedback an agent receives upon executing an action and reaching that state. The agent's behavior is guided by this feedback, in that the higher the reward, the more desirable the state, while the lower the reward, the less desirable the state. In the case it's positive, then the agent will try to be in the favorable situations and in the negative the agent will try to avoid the unfavorable situations. The agent's choice of actions and thus the learned policy depends in turn on the nature and magnitude of the rewards. The reward structure can be even subtle and yet it can significantly change the behavior of the agent. The return which is the total reward accumulated over time is expressed by

$$\boldsymbol{R_t = r_{\{t+1\}} + r_{\{t+2\}} + r_{\{t+3\}} + \cdots + r_{\{\infty\}}} \qquad \mathbf{5}$$

f. Discounted cumulative function

At each timestep k the reward is obtained $\rho(\mathrm{x_k})$ is based on its state x_k . The total return of an agent's experience over time can be quantified as the sum of discounted rewards. This captures the notion that immediate rewards are more valuable than future rewards, which are weighted by a discount factor $\gamma \in [0,1]$. The discounted cumulative reward can be formulated in two ways depending on the time horizon:

Finite Horizon Return:

$$V(\xi) = \rho(x_0) + \gamma\rho(x_1) + \gamma^2\rho(x_2) + \cdots + \gamma^N\rho(x_N) \qquad 6$$

Here, $\xi = (\mathrm{x_0}, \mathrm{x_1}, \ldots, \mathrm{x_N})$ is a trajectory of $N + 1$ step.

Infinite Horizon Return:

$$V(x) = \sum_{k=0}^{\infty} \gamma^k \rho(x)_k \qquad 7$$

This infinite horizon formulation is often used in theoretical analysis and in tasks that continue indefinitely. In both cases,

γ is the discount factor that determines how much future rewards are worth compared to immediate ones, and $\rho(x)_k$ denotes the reward at state x_k.

g. Policy

Decision making strategy of an agent in each possible state is called a policy whose choice of action is defined by a policy. In other words, formally it is a mapping from the set of all states to the set of possible actions. The goal of the optimal policy is to make an agent maximize its expected cumulative reward over time. A policy is different from a plan, as it does not prescribe a specific sequence of steps to get to a predetermined outcome but gives guidance as to what action will be the best to take at any point in any encountered state for the agent. The reason for that distinction is due to the Markov property that implies that the optimal choice is solely determined by the current state. As a result, an agent's policy is an approach on how to interact with the environment choosing the best action in each environment(Naeem et al., 2020). In general, the policy optimal problem is solved by using the Bellmann equation, which will be discussed in the following section.

H. Bellman Equation

Richard Bellman created the Bellman equation that forms a base in both Markov Decision Processes (MDP) and reinforcement learning in 1953. Computing state utilities and values requires this relationship because it establishes how present state value connects to future state expectancy. The Bellman equation presents a recursive system for estimating future expected reward values within each state. The formula can be written as follows:

$$V(x) = \rho(x) + \gamma \max_a \sum_{x'} \mathcal{T}(x, a, x')V(x') \qquad 8$$

Here, $V(x)$ denotes the value (utility) of state x, $\rho(x)$ is the immediate reward obtained in state x, $\gamma \in [0,1]$ is the discount factor that determines the importance of future rewards, and $\mathcal{T}(x, a, x')$ represents the transition probability of reaching state x' from state x after taking action a. The equation recursively incorporates both the immediate reward and the maximum expected value of possible next states, making it essential for identifying optimal policies in reinforcement learning frameworks(Mnih et al., 2016b).

2.4 Types of Reinforcement Learning Algorithms

Reinforcement learning techniques exist as model-based and model-free approaches which form the two principal methodological categories. Reinforcement learning methods such as Q-learning and Deep Q-Networks (DQN) are part of model-free approaches which directly update policies or value functions based on experience rather than building accurate environmental transitions models. Model-based RL approaches create a specific environmental model which they then utilize for policy optimization and planning tasks(Arulkumaran et al., 2017). Figure 4 illustrates the taxonomy of principal RL algorithm families discussed below.

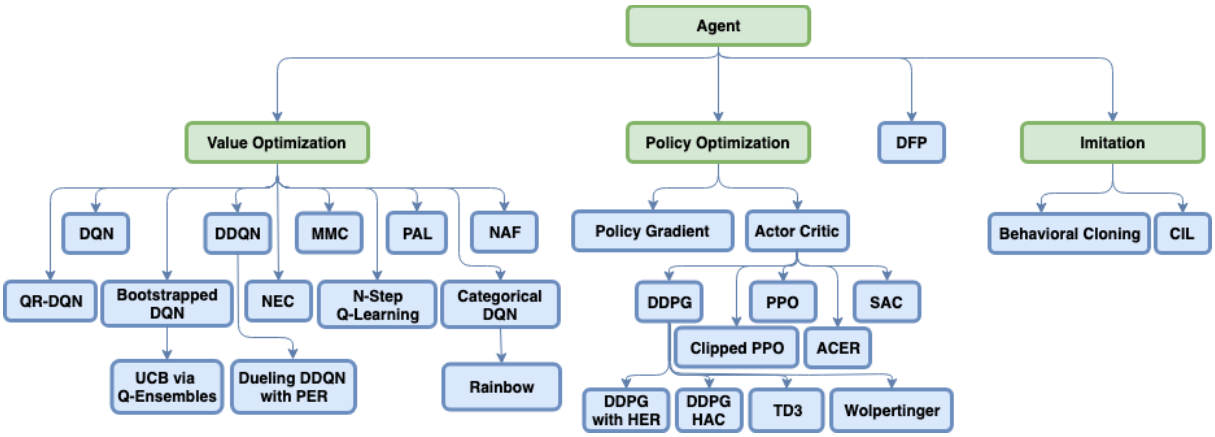

Figure 4. Taxonomy of major RL algorithm families

This diagram provides a hierarchical overview of the primary algorithmic families in reinforcement learning. At the highest level, all algorithms are unified under the concept of the agent. The taxonomy branches into four main categories, Value Optimization, Policy Optimization, Direct Future Prediction (DFP), and Imitation Learning.

2.4.1 Model-Free Reinforcement Learning

Model-free RL algorithms (shown in figure 5) empower agents to learn optimal policies exclusively through direct experience, eschewing the need for an explicit model of the environment's dynamics. These methods adjust policy or value estimates based on observed transitions and rewards, without simulating or predicting future events(Kahil et al., 2025). The most widely adopted model-free techniques include value-based approaches notably Q-learning and SARSA. Q-learning, as an off-policy algorithm, updates its value estimates using the maximum potential reward achievable from the next state, which often leads to more exploratory behavior (CHRISTOPHER J.C.H. WATKINS & PETER DAYAN, 1992). Conversely, SARSA, being on-policy, updates values based on the agent’s actual actions, generally resulting in more conservative policies. Such value-based methods are typically represented via a Q-table, which tabulates the expected returns for each state-action pair (Sutton & Barto, 2020). Actor-critic methods combine the strengths of value-based and policy-based approaches by utilizing two components, the actor (policy) and the critic (value function). These algorithms achieve robust performance in continuous or high-dimensional settings (Arulkumaran et al., 2017).

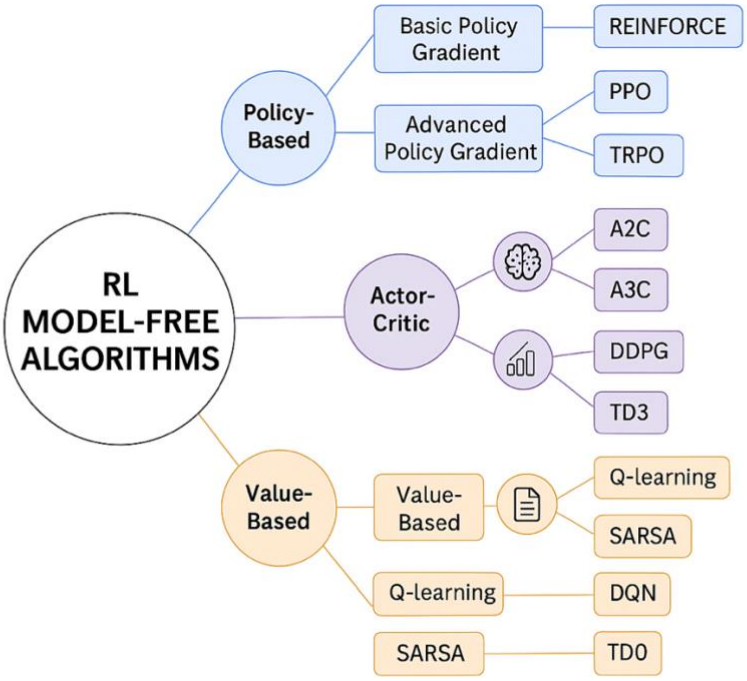

Figure 5. Model-Free Reinforcement Learning Algorithms: Value-Based, Policy-Based, and Actor-Critic Methods.

2.4.1.1 Actor-Critic Methods

Actor-critic methods represent a hybrid reinforcement learning algorithm class that integrates value and policy-based approaches illustrated in figure 6. In these frameworks, the "actor" selects actions based on a policy, while the "critic" assesses the value function to provide feedback on the quality of those actions. By combining the strengths of both paradigms, actor-critic algorithms can efficiently handle high-dimensional or continuous state and action spaces and often exhibit improved convergence properties (Arulkumaran et al., 2017)

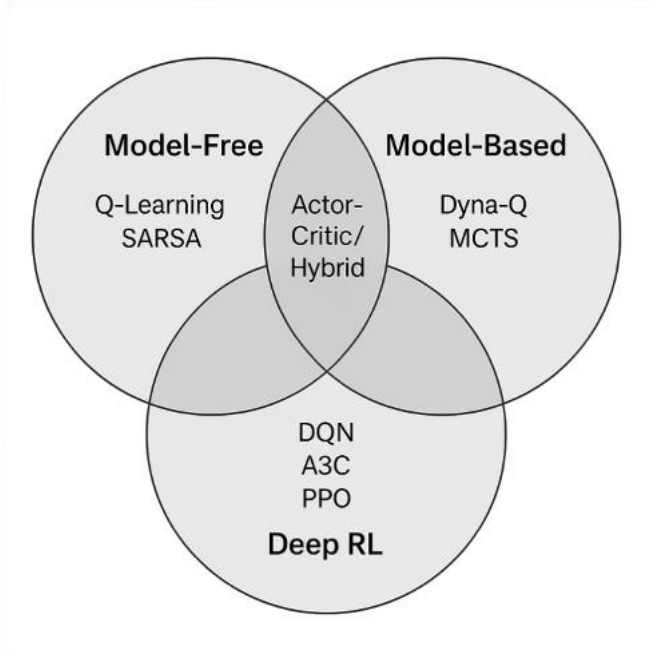

Figure 6. Actor-critic Interception

2.4.2 Model-Based Reinforcement Learning

Model-based RL incorporates or learns a model that captures the environment's transition probabilities and reward structure, enabling the agent to plan by simulating possible future outcomes (Deisenroth et al., 2015). This model allows for more sample-efficient learning and strategic foresight. Notable hybrid approaches, such as Dyna-Q, combine real experience with simulated trajectories generated from the model to accelerate learning (Sutton & Barto, 2020). Planning methods like Monte Carlo Tree Search (MCTS) exemplify the utility of model-based strategies in environments were searching through possible action sequences is tractable and informative such as in strategic board games (Silver et al., 2016).

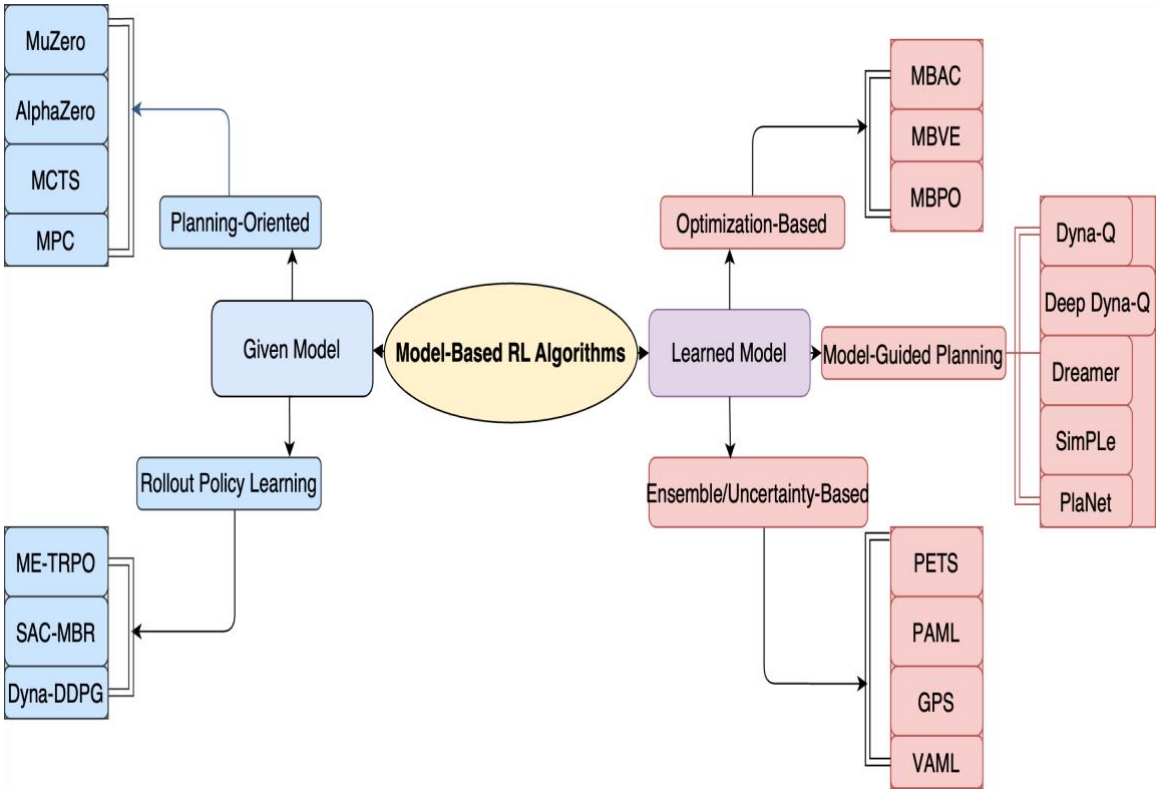

Figure 7. Taxonomy of RL Model-Based Algorithms

Planning-Based Approaches:

Algorithms like Monte Carlo Tree Search (MCTS), Model Predictive Control (MPC), and Value Iteration utilize a model to simulate future states. They optimize action selection by evaluating the outcomes of different action sequences. These methods are especially effective in areas where the model can be queried efficiently, such as in games and robotics(Świechowski et al., 2023).

Policy-Based Model-Based Methods:

The policy-based category includes algorithms like Guided Policy Search, Model-Based Policy Optimization (MBPO) (Janner et al., 2021), and Model-Based Maximum a Posteriori Policy Optimization (MB-MPO) (Janner et al., 2021). These methods directly optimise policies using an environmental model, often combining planning and learning to enhance sample efficiency.

Hybrid/Integrated Methods:

Algorithms such as Dyna-Q, World Models, Dreamer, PETS, SimPLe, and PlaNet merge the advantages of both model-based and model-free techniques. They use simulated experiences generated from the model to supplement real experiences, thereby accelerating learning and enhancing performance in data-scarce environments (Sutton & Barto, 2018)

2.4.3 Deep Reinforcement Learning

Reinforcement learning traditionally uses tabular methods to estimate value functions or policies, storing a value for each state (or state-action pair) in a table. This approach works for simple tasks, but it breaks down in large or continuous state-action spaces. The reason is that the number of possible states grows combinatorially with state variables, making a complete table infeasible(Long & Han, 2023). In high-dimensional or continuous domains, the agent would never visit enough states to fill the table or generalize to new states. In fact, attempting to estimate a separate value for each state without generalization is totally impractical because the value function would be estimated separately for each state without any generalization (Mnih et al., n.d.) . As a result, classical tabular RL fails to scale up it requires function approximation to generalize from seen states to unseen ones.

2.4.3.1 Function Approximation and the Power of Deep Learning

To handle large state spaces, RL algorithms turn to function approximation, where a parameterized function (such as a neural network) represents the value function or policy. Function approximation allows the agent to generalize knowledge gained from one state can impact values of similar states. This is essential for high-dimensional problems(Mnih et al., 2013). Early approaches used linear function approximators or simple non-linear approximators, but modern RL has increasingly adopted deep neural networks as function approximators(Long & Han, 2023). Deep learning provides powerful representation learning, enabling RL agents to automatically learn features

from raw inputs rather than relying on hand crafted state representations. By learning compact, multi-level features, deep RL methods can cope with the curse of dimensionality far better than tabular methods(Mnih et al., 2013). For example, convolutional neural networks (CNNs) can extract spatial features from pixel inputs, allowing an agent to learn directly from raw images(Mnih et al., 2013). In general, deep neural networks can approximate complex nonlinear value or policy functions in very large state spaces, making it possible to tackle problems (video games, robotics, Go, etc.) that were previously intractable for RL(Long & Han, 2023). This synergy of reinforcement learning with deep learning often called deep reinforcement learning (DRL) has led to dramatic successes. DRL agents can learn to play video games from pixels, control robots from camera inputs, and even defeat world champions in Go. These achievements were enabled by deep neural networks' ability to generalize and learn hierarchical features, which allows RL agents to handle the complexity of real-world observation spaces(Mnih et al., 2013). In summary, deep learning serves as a powerful function approximator in RL, addressing the limitations of tabular methods and unlocking previously unattainable applications.

2.4.3.2 Key Deep Reinforcement Learning Algorithms

Several milestone algorithms have demonstrated the effectiveness of deep learning in RL. Among them, Asynchronous Advantage Actor-Critic (A3C) and Proximal Policy Optimization (PPO) are particularly influential. A3C (Mnih et al., 2016a) introduced a paradigm of launching multiple parallel actor-learners to stabilize training. Each actor interacts with its own environment copy, accumulating gradients for a central model. By updating asynchronously, A3C decorrelates updates in time and eliminates the need for a replay buffer (which was crucial in earlier methods like DQN). The diverse experience collected in parallel serves a role similar to experience replay, stabilizing learning without storing large buffers. This approach greatly improved training efficiency (nearly linear speed-up with number of threads) and enabled on-policy algorithms (like actor-critic) to train deep networks stably. A3C achieved state-of-the-art results on a variety of tasks

and demonstrated the strength of using deep networks for both policy (actor) and value (critic) functions in an RL agent (Mnih et al., 2016b).

PPO (Schulman et al., 2017) is another widely-used deep RL algorithm, known for its balance of simplicity and performance. PPO is a policy gradient method that improves training stability by restricting the magnitude of policy updates. It uses either a clipped objective or a penalty on the KL-divergence to ensure the new policy does not deviate too far from the old policy during each update. This avoids catastrophic drops in performance that can occur with large policy changes. The authors describe it as having "the stability and reliability of trust-region methods but much simpler to implement, requiring only a few lines of code change to a vanilla policy gradient," while achieving better overall performance than previous methods(Schulman et al., 2017). PPO's simplicity and robustness have made it a go-to algorithm for many practical applications of deep RL, from games to robotics.

These algorithms (A3C, PPO, and others) underscore the importance of deep learning, the use of neural network function approximators in RL is what allows these methods to scale to complex environments and learn directly from high-dimensional inputs. Next, we focus on one of the breakthrough developments in deep RL the Deep Q-Network which was pivotal in demonstrating the power of deep learning in an RL context.

2.4.4 Deep Q-Networks (DQN)

One of the earliest and most influential successes in deep RL was the Deep Q-Network (DQN) algorithm, introduced by (Mnih et al., 2013). DQN combined Q-learning a classic off-policy RL algorithm with deep neural networks, enabling an agent to learn to play Atari 2600 video games directly from raw pixel inputs. This was a breakthrough because it demonstrated, for the first time, that an RL agent using a deep CNN could achieve human-level performance on many challenging tasks with high-dimensional state spaces. In their Nature paper, Mnih et al. (2013) showed that a single DQN agent, with the same network architecture and hyperparameters, could learn to play a suite of 49 Atari games, outperforming all previous RL methods on the majority of those games and even

surpassing human expert scores on several games. This result galvanized interest in deep RL as it illustrated the raw power of deep neural networks for representation learning in complex environments.

How DQN Works? DQN is essentially a Q-learning algorithm that uses a neural network to approximate the Q-value function $Q(s,a)$. In classical Q-learning, one would update a table of Q-values using the Bellman equation,

$$Q_{\text{tabular}}(s,a) \leftarrow Q_{\text{tabular}}(s,a) + \alpha\left[r + \gamma \max_{a'} Q_{\text{tabular}}(s',a') - Q_{\text{tabular}}(s,a)\right] \quad 8$$

where α is the learning rate. DQN replaces the table with a deep neural network $\mathrm{Q(s,a;\theta)}$ parameterized by weights θ. The network takes the state s as input (for Atari, this is typically a stack of the last four video frames, to incorporate time information) and outputs a vector of Q-values, one for each possible action(Mnih et al., 2015).

During training, Deep Q-Networks (DQN) use stochastic gradient descent to minimize the temporal difference (TD) error derived from the Bellman equation. The core objective is to reduce the discrepancy between the Q-network's prediction and a target Q-value obtained from the agent's interaction with the environment. This is formalized by the following loss function at each iteration i,

$$\mathcal{L}_i(\theta_i) = E_{(s,a,r,s')\sim D}\left[\left(y_i - Q(s,a;\theta_i)\right)^2\right] \quad 9$$

where y is the target value for the Q-network. The target $y_i$is given by the Bellman update (using the reward r plus discounted next-state value) and is treated as a constant for the sake of computing gradients (Mnih et al., 2013). A common choice (used in DQN) is,

If s is a terminal state (episode ended), then

$$\mathrm{y = r} \quad 10$$

Otherwise:

$$y = r + \gamma \max_{a'} Q(s', a'; \theta^{-}) \quad 11$$

Here γ is the discount factor, and $\theta^{-}$are the parameters of a target network, a fixed copy of the Q-network(*DDS Notes - PU BCA Study Materials*, n.d.). In words, the target y is the estimated return (reward plus discounted future value) if the agent takes action a in state s and then follows the best possible actions thereafter. The network's prediction $Q(s, a; \theta)$ is updated toward this target. The gradient of the loss with respect to parameters θ corresponds to the familiar temporal-difference (TD) $\delta = y - Q(s, a; \theta)$, which is backpropagated through the network to adjust the weights. By iterating this process over many episodes, the network learns to approximate the true $Q*(s, a)$.

2.4.4.1 Stabilizing Training Experience Replay and Target Networks

Early attempts to train neural networks with Q-learning were unstable. To address this, DQN introduced two key techniques that significantly improved stability.

a. Experience Reply

Rather than updating the network using consecutive (and highly correlated) states, DQN stores past experiences (s, a, r, s') in a replay buffer *D* During training, it samples mini batches randomly from this buffer. This breaks temporal correlations, making learning more stable and efficient by allowing experiences to be reused. Although DQN uses uniform sampling, later improvements like prioritized experience replay focus on more informative transitions(Mnih et al., 2015; Schulman et al., 2017).

b. Target Network

DQN maintains a separate target network $Q(s, a; \theta^{-})$, which is a copy of the main Q-network. It's updated only every *C* steps by copying the weights from the main network. This helps stabilize the target values $y = r + \gamma \max_{a'} Q(s', a'; \theta^{-})$, which would otherwise shift too rapidly during training. Without this, the model risks divergence due to unstable targets a problem known as the "deadly triad" in RL(Mnih et al., 2015). Together,

experience replay and the target network are essential for stable and successful deep reinforcement learning with DQN.

DQN made a major breakthrough by learning to play dozens of Atari 2600 games at human level performance, using nothing but raw pixel inputs and game scores. This has been a powerful demonstration of deep reinforcement learning, the same algorithm and network architecture could learn to play very different games such as Pong, Breakout, and Space Invaders without any game specific tuning. In many cases, it matched or even outperformed professional human testers. This showed that deep neural networks could effectively act as function approximators in reinforcement learning, handling huge state spaces and learning directly from high dimensional inputs like images.(Mnih et al., 2015).

The success of DQN sparked a wave of research in deep reinforcement learning. It led to important improvements such as Double DQN (to reduce overestimation), Dueling Networks (to separate value and advantage), and Prioritized Experience Replay (to learn more efficiently). It also inspired the development of new algorithms beyond the value-based approach.

DQN highlighted why deep learning is so valuable in reinforcement learning. By using a deep convolutional network, the agent could automatically extract meaningful features for decision-making like the ball and paddle in Pong without needing handcrafted input features. Deep function approximation has since become a core principle in reinforcement learning, especially for problems with large or continuous state and action spaces. More advanced algorithms like A3C and PPO have built on this idea, showing that with the right network architectures and training methods, deep learning can enable agents to master complex environments that were once out of reach.

Reinforcement learning includes various algorithmic frameworks that encompass model-based and model-free approaches and deep RL techniques alongside hybrid methods with specific application domains. Such a defined approach gives researchers a complete picture of basic RL algorithm families while helping them find suitable methods for their research issues and practical application requirements. In the next chapter The

Global Navigation Satellite System (GNSS), its importance and challenges has been discussed.

3 GNSS Importance and Challenges

The Global Navigation Satellite System (GNSS) refers to generic satellite positioning systems which deliver global positioning under autonomous and geo-spatial conditions. GPS serves as the principle GNSS operation alongside three other space-based positioning systems managed by Russia as GLONASS and the European Union as Galileo and China as BeiDou Japan as QZSS, and India as NaviC shown in the figure 8 below(Jin et al., 2022). GNSS receivers use multiple systems that let users receive precise continuous measurement of position, velocity, and time (PVT) information throughout areas close to and on Earth(*GNSS Receivers General Introduction - Navipedia*, n.d.).

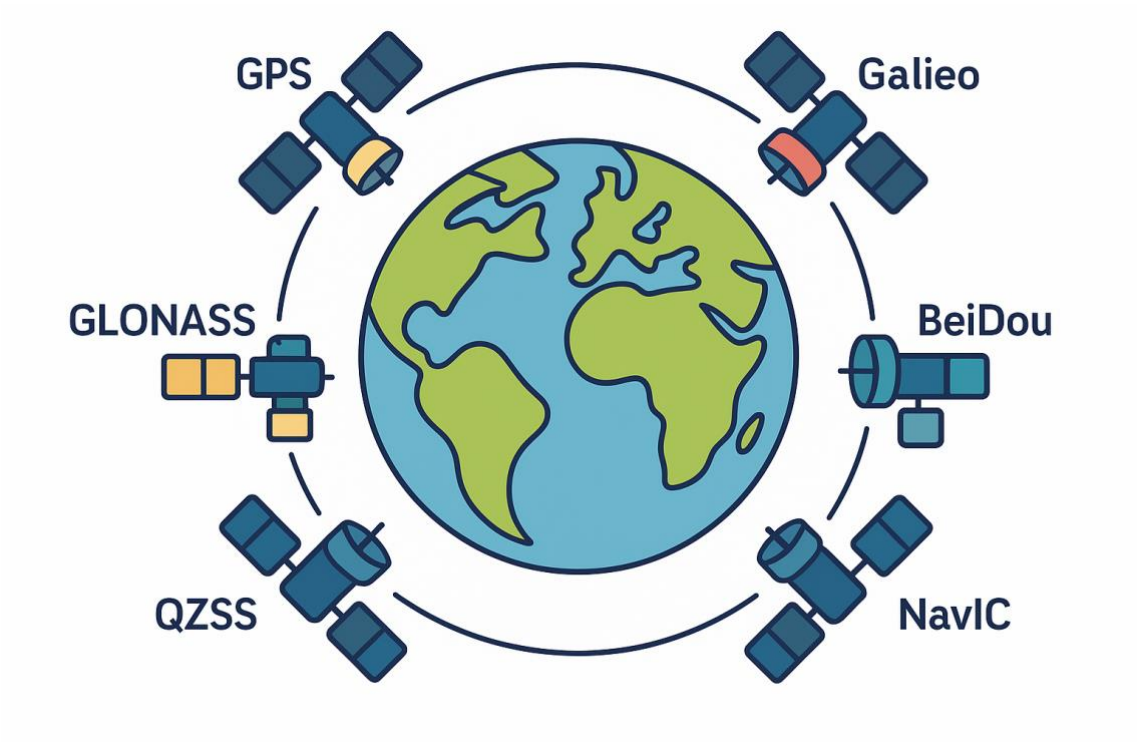

Figure 8. Global Navigation Satellite Systems

All Global Navigation Satellite Systems (GNSS), regardless of the provider, share a fundamentally similar design structure. Each satellite transmits three crucial types of information, a ranging signal for calculating position, velocity, and time (PVT), precise ephemeris data that indicates the satellite's real-time position and an almanac that contains the orbital parameters and operational status of all satellites within the constellation. The built-in interoperability capability of GNSS receivers allows them to use signals from all four satellite constellations simultaneously which improves signal accessibility during continuous operation. Receivers can prevent signal outages when positioned in environments with obstructed views through the use of data from visible satellites from various systems (Schmidt et al., 2016).

The primary GNSS systems have developed unique orbital arrangements to guarantee space-based satellite visibility around the whole planet. The different satellite constellations and orbital planes and altitude heights emerged through design choices and historical development as well as geographical coverage requirements of each GPS system. GPS operated by the United States utilizes thirty-one spacecraft distributed into six Medium Earth Orbit planes to maintain a reliable position service across the globe as presented in Table 1. Russia operates GLONASS with 26 satellites through three MEO orbital planes that have been arranged for better coverage in northern regions. The Galileo system operated by the European Union consists of 27 operational satellites organized in three satellite planes for delivering precise positioning services to civilians. The BeiDou system of China combines MEO orbit with IGSO and GEO satellites to serve worldwide and regional users (Jin et al., 2022). The combined orbital arrangements provide uninterrupted positioning navigation and timing services for the ongoing support of billions of worldwide users (U.S. Government, GPS.gov, 2025).

Table 1. Global GNSS Constellations as of April 2025

Systems	**Region**	**Operational Satellites**	**Orbital Configuration**
GPS	United States	31	6 orbital planes containing 4–6 satellites (MEO)
GLONASS	Russia	26	3 planes × 8 satellites per plane (MEO)
Galileo	EU	27	3 planes × 9 satellites per plane (MEO)
BeiDou	China	45	Combination of MEO, IGSO, and GEO satellites

The GPS uses 31 operational satellites (from Table 1) which position themselves across six Medium Earth Orbit (MEO) orbital planes as of April 2025 (*GPS.Gov: Space Segment*, n.d.). The satellite network exceeds basic requirements of 24 satellites to ensure worldwide coastline and backup coverage. The U.S. Space Force operates this enlarged coverage area to boost operational functionality and system availability. These satellites complete an orbit approximately every 12 hours and ensure global coverage. Each satellite continuously broadcasts navigation messages that include the current timestamp and the precise ephemeris data of the satellite's position in space, allowing receivers to determine the time and location of transmission (Li et al., 2019). More about general GNSS signal characteristics has been discussed in the section below.

3.1 GNSS Signal Characteristic

All Global Navigation Satellite Systems (GNSS) use L-band electromagnetic signals to transmit frequency ranges spanning from 1.1 to 1.6 GHz for their operation. Digital information for positioning, navigation and timing applications along with different modulation methods transmit through these signals from various satellite systems (Schmidt et al., 2016).

Binary Phase Shift Keying (BPSK) serves as the preferred modulation technique that operates on GNSS signals from GPS and GLONASS L1. Binary data transmission occurs through BPSK methodology which causes the carrier wave to experience a full 180-degree phase shift for each bit carried(*Spread Spectrum and Code Modulation of L1 GPS Carrier | GEOG 862: GPS and GNSS for Geospatial Professionals*, n.d.). BeiDou-2 implements Quadrature Phase Shift Keying (QPSK) to achieve four different phase states for transmitting two bits during each phase transition as explained in. Galileo employs multiplexed BOC (MBOC) modulation within its E1 signal while using a specific form of Binary Offset Carrier technology for better spectral separation and GPS L1 signal interoperability (Schmidt et al., 2016a)

GNSS signals transmit their two orthogonal channels as in-phase (I) and quadrature-phase (Q). The system uses two phase-shifted components that transmit separate forms of information. The civilian Coarse/Acquisition (C/A) code travels through the I channel of the GPS L1 signal operating at 1575.42 MHz, The Q channel distributes encrypted military P(Y) code which differs from the I channel information(Schmidt et al., 2016a).

GNSS signal structure uses unique pseudorandom noise (PRN) codes on each satellite to enable receivers to identify different signals transmitted from a single frequency. The L1 C/A code of GPS operates with a 1.023 Mbps chip rate that produces 1023-chip sequences completing transmission within a one-millisecond interval. Military signals like GPS L2 W-code need numerous days to finish their transmissions because they possess exceptionally lengthy PRN sequences. The navigation message containing orbital

parameters, timing data and almanac information is sent at 50 bits per second(Schmidt et al., 2016a).

GNSS systems like GPS, Galileo, BeiDou, and others use the signal access method Code Division Multiple Access to receive signals simultaneously. FDMA operates as GLONASS's traditional access method due to its deployment of distinct frequencies for each satellite broadcast. Modernization within GLONASS brought CDMA signals to enhance reception with other global satellite navigation systems (Schmidt et al., 2016a)

GNSS signals originating from altitudes from 19,100 km GLONASS satellites to 35,786 km BeiDou geostationary satellites experience atmospheric-induced distortions while travelling to Earth. High solar activity inside the ionosphere delivers signal delays that reach up to 300 nanoseconds which would result in 90-meter position errors when left uncorrected(Dana, 1997) .

GNSS modernization programs use new signal structures for both accuracy improvement and system reliability enhancement purposes. GPS incorporates the L5 signal which broadcasts BPSK(10) modulation on 1176.45 MHz frequency to act as an essential safety feature (*GPS Signal Plan - Navipedia*, n.d.). The E5a and E5b signals introduced by Galileo operate with AltBOC modulation to achieve wideband and high-precision performance(*Galileo Signal Plan - Navipedia*, n.d.). BeiDou-3 serves users with two new frequencies named B1C and B2a that enhance both global connectivity and multiple signal correction methods (*BeiDou Signal Plan - Navipedia*, n.d.)

3.2 GNSS Position & Time Solutions

Users obtain positioning data alongside timing data through GNSS systems which determine travel times of radio signals between orbiting satellites and receivers located on Earth. GNSS-based navigation relies on the pseudo range measurement as its fundamental metric which provides an estimated distance between satellite and receiver by assessing time delays of signal transmission and reception. The time delay

responsible for this measurement includes the simple geometric distance together with receiver clock bias and atmospheric interference and relativistic effects (Schmidt et al., 2016a).

Each GNSS satellite broadcasts a signal that includes a highly accurate timestamp, enabled by onboard atomic clocks. However, since the receiver's internal clock is typically less stable, it introduces an unknown time offset T. The geometric distance from satellite i, located at coordinates (x_i, y_i, z_i), to a receiver at position $(\mathrm{x}, \mathrm{y}, \mathrm{z})$ is calculated as:

$$p_i = \sqrt{(x_i - x)^2 + (y_i - y)^2 + (z_i - z)^2} + cT \qquad 12$$

where c is the speed of light. This equation defines the pseudorange p_i, which includes both the true range and the time bias-induced error. Since there are four unknowns (three spatial coordinates and one clock bias), the receiver must acquire signals from a minimum of four satellites to compute a unique solution. These nonlinear equations are typically solved using iterative least squares or extended Kalman filtering techniques, depending on the receiver type and application (Schmidt et al., 2016b)

To improve computational efficiency and minimize the impact of clock bias, it is common to subtract one pseudorange equation from the others, thereby eliminating the common clock offset term cT. This transformation yields a set of range difference equations, each defining a hyperboloid in three-dimensional space. The intersection of these hyperboloids determines the receiver's location. This method, known as hyperbolic multilateration, is conceptually similar to time-difference-of-arrival (TDOA) systems used in terrestrial navigation and tracking(Schmidt et al., 2016a).

Once the receiver's spatial coordinates have been resolved, the local clock can be synchronized to GNSS system time. This process involves recomputing the time of signal travel using the known satellite transmission time t_n and satellite position (x_n, y_n, z_n),

compared against the computed receiver location. The receiver's local time t_l is then derived as:

$$t_l = t_n + \frac{1}{c}\sqrt{(x_n - x)^2 + (y_n - y)^2 + (z_n - z)^2} \quad 13$$

For stationary receivers such as those used in timing infrastructure or network synchronization, it is not necessary to solve the navigation problem in real-time continuously. Instead, a fixed position solution can be averaged over many epochs to enter a position hold mode, wherein subsequent timing computations rely on a known spatial reference. In contrast, mobile receivers must solve both time and position repeatedly, often at rates of 1–10 Hz or higher, depending on motion dynamics and application requirements(Schmidt et al., 2016a).

The accuracy of GNSS timing solutions depends on multiple system-level and environmental factors. These include the signal bandwidth, multipath interference, ionospheric conditions, and whether the receiver supports dual-frequency operation for ionospheric correction. Manufacturer specifications vary, but typical real-world timing accuracies range from ±4 to ±100 nanoseconds. For example, GPS generally provides timing precision better than 40 ns, while GLONASS-M satellites achieve ≤8 ns, and BeiDou-2 is rated below 20 ns. High-end commercial receivers can reach timing stability in the range of ±4–15 ns when operating under optimal conditions with clear sky visibility and dual-frequency input(Schmidt et al., 2016a).

Ultimately, the ability to accurately resolve both position and time underpins not only navigational applications but also time-critical infrastructure systems. GNSS receivers are now foundational in financial transactions, mobile telecommunications, electrical grid synchronization, and scientific research, where nanosecond-level timing precision is often required.

3.3 Augmentation Systems

There are many things that can introduce errors into Global Navigation Satellite Systems (GNSS), including drifting satellite clocks, inaccurate ephemeris data and noise in the atmosphere. To solve these problems and increase the accuracy, reliability and integrity of signals, systems have been built called augmentation systems(Sabatini et al., 2017) . Two fundamental augmentation systems exist for GNSS Satellite-Based Augmentation Systems (SBAS) and Differential GNSS (DGNSS).

Satellite-Based Augmentation Systems (SBAS)(Figure 9) operate by collecting real-time GNSS data at precisely located ground reference stations. These measurements compute correction information, which is then transmitted to users via geostationary satellites on dedicated augmentation frequencies. SBAS significantly improves the performance of standard GNSS signals and supports applications requiring high integrity and precision, such as civil aviation (Yoon et al., 2020).

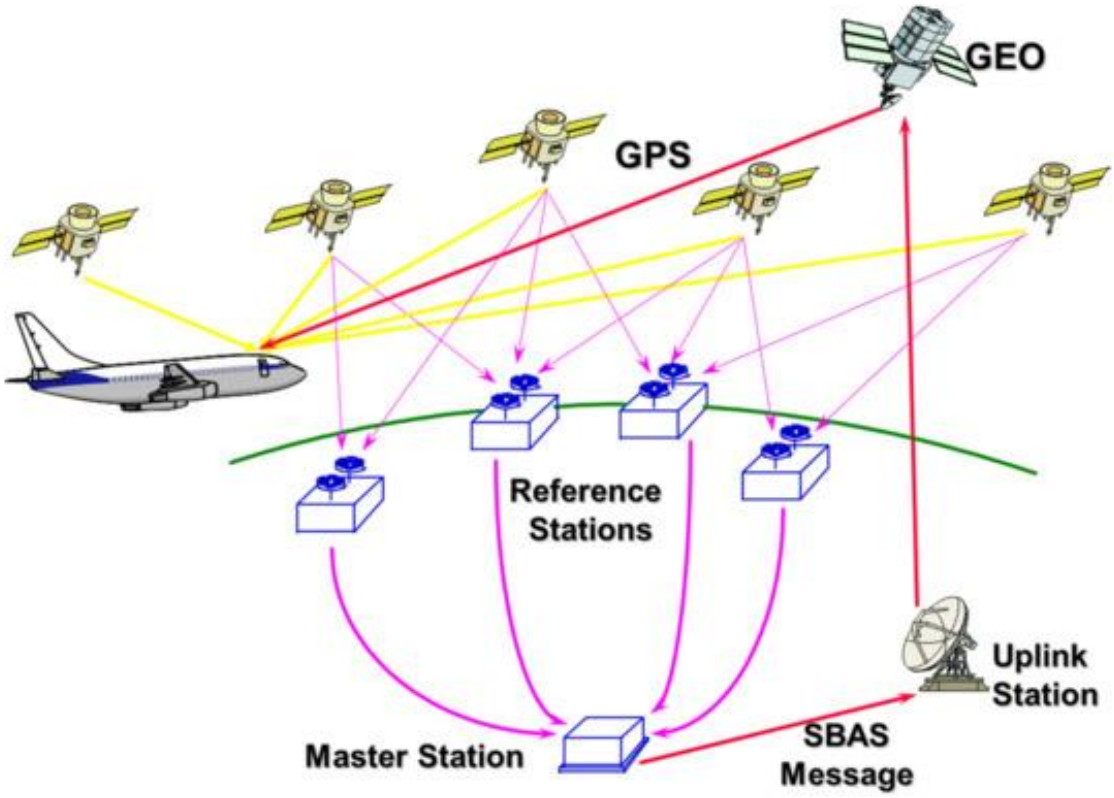

Figure 9. Satellite-Based Augmentation System (SBAS) Architecture(Yoon et al., 2020b)

Several SBAS systems have gone operational across specific global regions. WAAS operates under United States jurisdiction to offer GPS correction capabilities over North America. Through European Geostationary Navigation Overlay Service (EGNOS) which operates under the European Union continues to improve signal quality across all of Europe. MSAS (Japan) and GAGAN (India) offer similar services in their respective regions. The BeiDou system within China operates BDSBAS which delivers corrections through two geostationary satellites for users across Asia-Pacific territory. Russia operates the System for Differential Corrections and Monitoring (SDCM) together with SouthPAN as a new joint program between Australia and New Zealand to augment GLONASS coverage in Australia (Sabatini et al., 2017; Yoon et al., 2020).

SBAS technology offers position precision of 1–2 meters but its correction signals remain restricted to the ground infrastructure coverage area together with satellite visibility range. SBAS providers must establish interoperable systems and specific implementations to match regional needs because of this system restriction(Yoon et al., 2020).

Differential GNSS (DGNSS) distributes real-time corrections through terrestrial radio beacons or satellite links from a network of land-based stations that have their exact positions recorded. DGNSS calculates and shares with receivers the comparison of true position and the location obtained from satellites. The system delivers precise positioning results of 10 to 15 centimeters when DGNSS operates under optimal circumstances(Weng et al., 2020). The operational reach of this system spans only the areas that have access to its signal correction network. Augmentation systems serve as essential components which make it possible for high-precision GNSS applications to function in aviation and maritime sectors and geodetic surveying and autonomous systems. Worldwide precise satellite navigation service demands grow because of their continuous development.

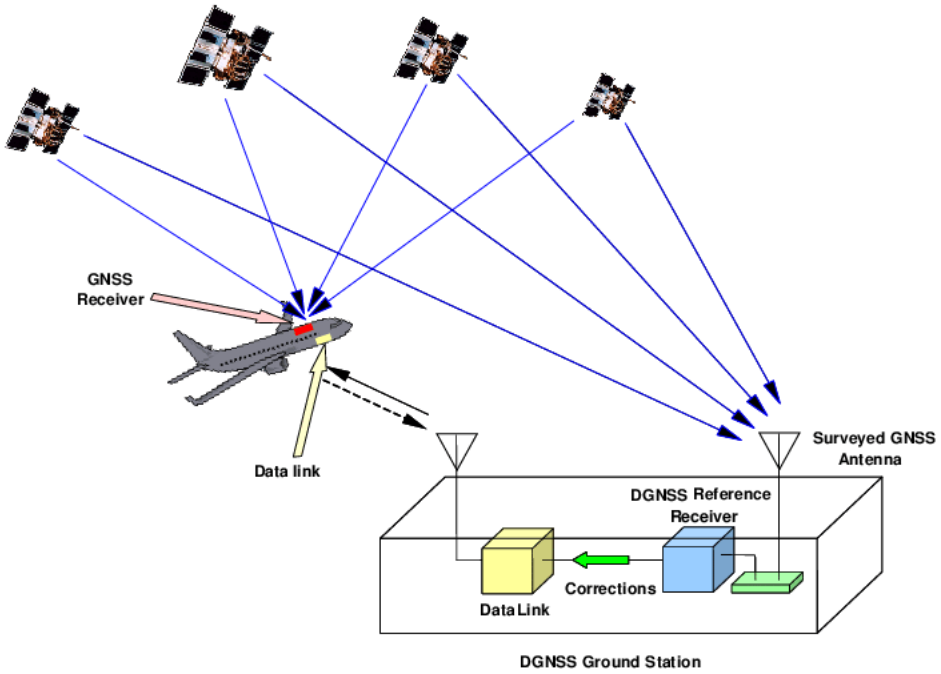

Figure 10: Typical Differential GNSS (DGNSS) system architecture (Sabatini, Moore, & Ramasamy, 2017)

Figure 10 illustrates a typical DGNSS system architecture, depicting the interaction between GNSS satellites, ground-based reference stations, correction data transmission, and user receivers. The diagram effectively demonstrates how correction data is generated and applied to enhance positional accuracy.

3.4 GNSS Vulnerabilities and Threats

Global Navigation Satellite Systems (GNSS) function as the central infrastructure that supports modern navigation, timing, and positioning activities. GNSS technology is widely utilized throughout society but faces inherent signal-level challenges caused by signal transmission characteristics and the openness of civilian GNSS platforms. The

vulnerabilities of GNSS systems primarily appear through signal degradation combined with misdirection resulting from deliberate interference using methods such as jamming and spoofing.

3.4.1 Weakness and Open Access Architechure

Global Navigation Satellite System satellites emit spread spectrum signals from satellites operating at Medium Earth Orbit (MEO) and maintain a standard altitude range between 19,100 kilometers for GLONASS and 23,222 kilometers for GPS and Galileo(Schmidt et al., 2016a). GNSS ground stations receive satellite signals with power levels ranging from -130 dBW to -160 dBW because of excessive propagation loss over such extended distances. The baseband signal received by a GNSS receiver in an interference-free scenario is modeled as:

$$r(t) = a(t) \cdot c(t - \tau) \cdot \cos(2\pi f_c t + \phi) + n(t) \qquad 14$$

where $a(t)$ denotes the time-varying amplitude, $c(t - \tau)$ the delayed PRN code, f_c the carrier frequency, ϕ the phase offset, and $n(t) \sim \mathcal{N}(0, \sigma^2)$ denotes additive white Gaussian noise.

The openness of civilian GNSS services further exacerbates vulnerability. Most civil GNSS signals (e.g., GPS L1 C/A, Galileo E1) are publicly documented and unencrypted, which facilitates unauthorized reconstruction or manipulation by adversarial actors. Such transparency, while essential for global accessibility, enables adversaries to replicate signal waveforms using software-defined radio (SDR) platforms and transmit counterfeit signals to mislead or disable GNSS receivers (Radoš et al., 2024).

3.4.2 Jamming

GNSS jamming refers to the intentional transmission of high-power signals within GNSS frequency bands to degrade or entirely block legitimate satellite signal acquisition. Given the already weak nature of GNSS signals at the Earth's surface, even low-power jammers can significantly raise the local noise floor, often exceeding the receiver's acquisition threshold and causing a complete loss of lock (Morales Ferre et al., 2019). Figure 11 shows a classic GNSS jamming system composed of a ground jammer, GNSS satellites and a GNSS receiver on a vehicle. The satellites send positioning signals to the receiver, but the jammer sends a strong radio frequency interference (denoted by red concentric arcs) that interferes with the satellite signals being received by the receiver.

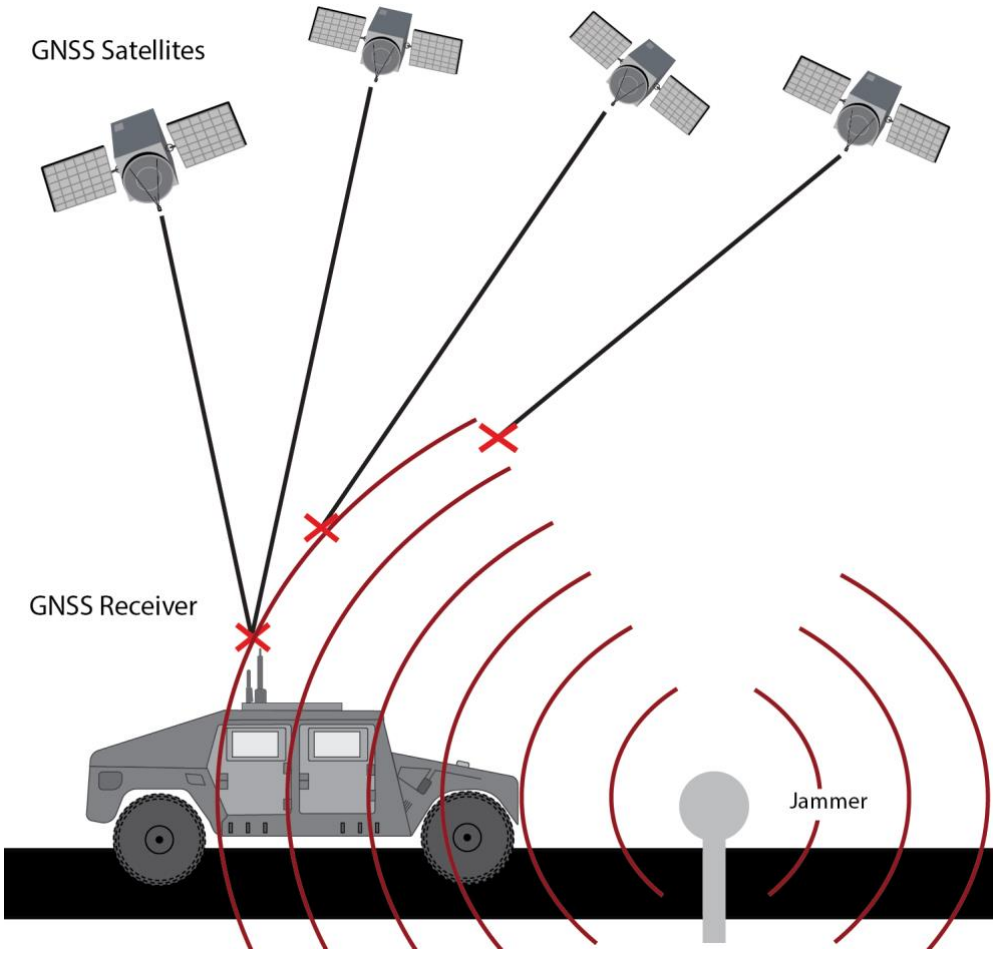

Figure 11. GNSS Jamming conditions (*VectorNav*, n.d.)

Formally, the composite received signal under jamming is expressed as:

$$r(t) = s_{\text{GNSS}}(t) + j(t) + n(t) \quad 15$$

Where $s_{"GNSS}$" (t) denotes the authentic GNSS signal, $j(t)$ is the jamming waveform, and $n(t) \sim \mathcal{N}(0, \sigma^2)$ is additive white Gaussian noise (Schmidt et al., 2016). In practical scenarios, the power of the jamming signal satisfies:

Continuous wave (CW) jamming:

$$j(t) = A_j \cos\left(2\pi f_j t\right) \quad 16$$

Swept frequency (chirp) jamming:

$$j(t) = A_j \cos(2\pi(f_0 + kt)t) \quad 17$$

where A_j represents the jamming signal amplitude, f_j the jamming frequency, f_0 the starting frequency of the chirp, and k the chirp sweep rate. These waveforms severely reduce the carrier-to-noise density ratio C/N_0, which is a critical parameter for satellite signal tracking (Radoš et al., 2024);(Psiaki & Humphreys, 2016).

3.4.3 Spoofing

Unlike jamming, which primarily aims to block GNSS service by overwhelming the receiver's input signal-to-noise ratio (SNR), spoofing is a more insidious form of interference. It involves the deliberate transmission of GNSS-like signals that closely mimic authentic satellite signals in structure, timing, and modulation, but are engineered to mislead the receiver into estimating an incorrect position, velocity, or time (PVT) (Psiaki & Humphreys, 2016).

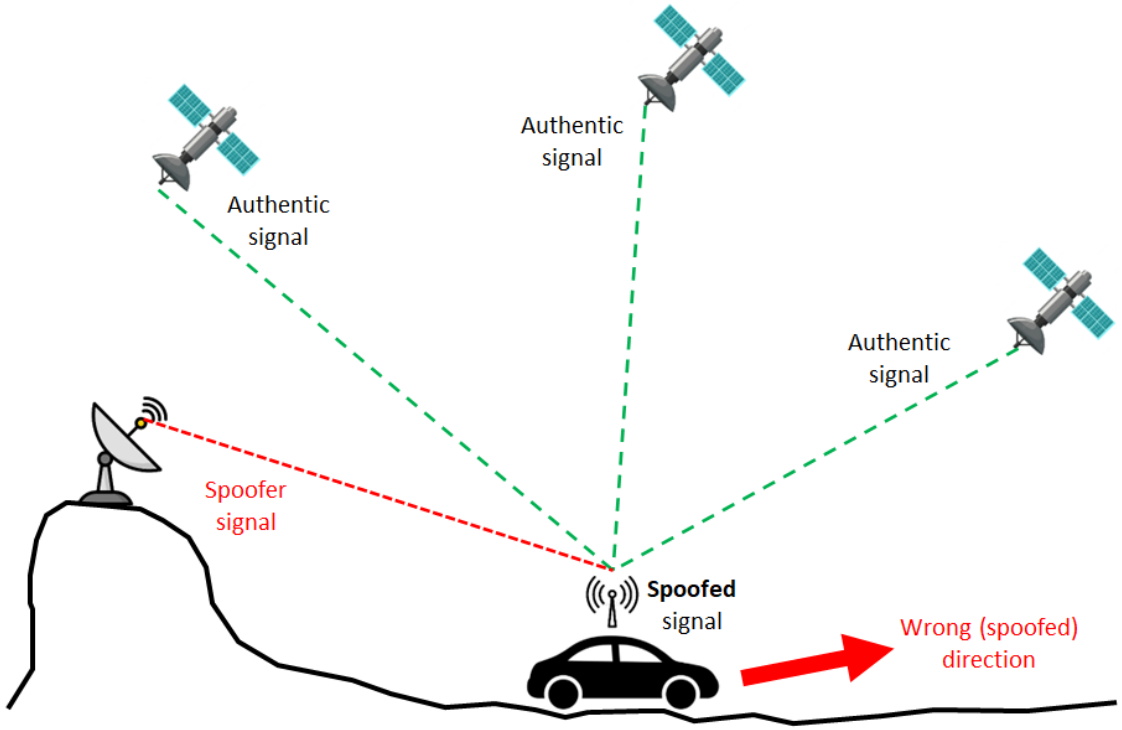

Figure 12. Diagram showing the mechanism of a spoofing attack. The spoofer deceives the GPS receiver in the vehicle, leading to an incorrect navigation solution that misguides the car's direction (*GNSS Spoofing*, 2022)

The spoofing attack is illustrated in Figure 12. A spoofer sends a fake signal to a victim or target receiver at a similar strength (without first using a jamming attack). After that, real GNSS signals from satellites blended with this spoof signal.

3.4.3.1 Spoofing Power Threshold and Principles

Spoofing attacks succeed by exploiting the low power margin of GNSS signals at Earth's surface, typically ranging between –160 dBW and –153 dBW under normal propagation conditions. Since these values span nearly a fivefold power range, receivers are already designed to handle considerable variability(Schmidt et al., 2016b). As Shepard and Humphreys (2011) demonstrated, a counterfeit signal needs only to exceed the authentic signal power by a modest ratio around 1.1 to take control of the receiver's

correlation process. This threshold is easily met by a moderately powered transmitter operating nearby. The spoofed signal can be mathematically modeled as:

$$s_{\text{sp}}(t) = a(t) \cdot c\big(t - \tau(t)\big) \cdot \cos\big(2\pi f_c t + \phi(t)\big) \quad 18$$

Where:

- $a(t)$ represents the time-varying amplitude of the spoofed signal, which may be adjusted to exceed the power of authentic GNSS signals without raising suspicion.
- $c\big(t - \tau(t)\big)$ denotes the delayed pseudorandom noise (PRN) code sequence used to modulate the signal, simulating the structure of a legitimate satellite transmission.
- f_c is the carrier frequency of the spoofed signal, typically aligned with standard GNSS frequency bands (e.g., L1 at 1575.42 MHz);
- $\phi(t)$ corresponds to the time-varying carrier phase, ensuring temporal coherence with authentic signals.
- $\tau(t)$ represents the spoofing-induced code delay, which shifts the perceived arrival time of the signal at the receiver.

This spoofed waveform is carefully engineered to replicate the structure of genuine GNSS signals, including their spectral, temporal, and modulation characteristics. By doing so, it becomes difficult for conventional receivers to distinguish the counterfeit signal from legitimate satellite broadcasts at the physical layer, thereby enabling deception in position, velocity, and timing estimates.(Schmidt et al., 2016a)

3.4.3.2 Composite Received Signal Under Spoofing

Under a spoofing scenario, the composite signal received at the GNSS antenna can be modeled as:

$$r(t) = s_{\text{GNSS}}(t) + s_{\text{sp}}(t) + n(t) \quad 19$$

Here, $s_{"GNSS}$" (t) denotes the authentic satellite signal, $s_{"sp}$" (t) represents the spoofed signal generated by the attacker, and $n(t) \sim \mathcal{N}(0, \sigma^2)$ models the additive white Gaussian noise component inherent in the receiver's radio environment(Psiaki & Humphreys, 2016; Schmidt et al., 2016b).

Spoofing becomes effective when the counterfeit signal exhibits a higher received power than the authentic one. In such cases, the receiver's acquisition process locks onto the spoofed signal due to stronger correlation peaks. This can be expressed as:

$$\left|s_{\mathrm{sp}}(t)\right| > \left|s_{\mathrm{GNSS}}(t)\right| \qquad 20$$

When this condition is met, the spoofed signal dominates the correlation process, causing the receiver to synchronize with and track the counterfeit signal instead of the legitimate satellite signal. This mechanism forms the foundational principle of GNSS signal takeover in practical spoofing attacks(Schmidt et al., 2016a).

3.4.3.3 Spoofing Classifications

GNSS spoofing can be classified based on the techniques used to generate and transmit deceptive signals, each with distinct characteristics and challenges. Meaconing is the most straightforward form of GNSS spoofing, where an attacker intercept authentic GNSS signals and re-broadcasts them after a brief delay. However, this approach is notably challenging for encrypted military signals, such as the GPS P(Y) code, which use significantly longer pseudorandom noise (PRN) sequences. Receivers equipped with secure military signals can easily detect any misalignment in the PRN sequence, as their internal clocks allow them to identify out-of-phase signals (Humphreys, 2013a)(Jung et al., n.d.). Moreover, the GPS P(Y) signal is transmitted at a power level below the noise floor, making accurate retransmission difficult without first estimating the secret W code a process achievable only through advanced "semicodeless" techniques(Jung et al., n.d.). The mathematical representation of a meaconed signal is given by:

$$r_m(t) = a(t) \times c\big(t - (\tau + \Delta\tau)\big)\cos(2\pi f_c t + \varphi) + n(t) \quad 21$$

From above equation, $r_m(t)$ is the received meaconed signal. $a(t)$ is the amplitude of the received signal, $c\big(t - (\tau + \Delta\tau)\big)$ is the delayed PRN code, with τ representing the natural propagation delay and $\Delta\tau$ being delay introduced by the attacker, f_c is the carrier frequency, φ is the phase offset, $n(t) \sim \mathcal{N}(0, \sigma^2)$ is addditive Gaussian noise.
This model highlights that meaconing is characterized by a uniform delay $\Delta\tau$ across all signals, making it a straightforward form of deception. However, this uniform delay may be detectable if the target device is time-sensitive, such as a GNSS-based clock.

In contrast, civilian GNSS signals do not pose the same difficulty. The timing relationships between the received signals remain unchanged during the meaconing process, allowing the target receiver to calculate a position based on the meaconer's location instead of the authentic satellite's (Papadimitratos & Jovanovic, 2008). The timing solution also aligns with that of the meaconer, adjusted by the delay introduced during signal retransmission(Wesson, Kyle D. et al., 2012).

Despite this, meaconing is less effective against timing devices. Such devices maintain a precise internal clock and can easily detect the sudden time shift caused by the delay in the retransmitted signal. This abrupt change conflicts with the device's expected time, triggering an alert. Alternatively, an attacker could execute meaconing without a noticeable delay by pre-calculating the GNSS signal values and synchronizing them precisely with the legitimate signals (Wesson, Kyle D. et al., 2012)

Security Code Estimation and Replay (SCER), also known as selective delay is an advanced variation of meaconing where an attacker selectively rebroadcasts specific satellite signals after a calculated delay. This technique allows the attacker to manipulate the target receiver's position and/or timing solutions. By introducing selective delays, the attacker can ensure that the affected receiver calculates a false position without triggering a sudden time shift, making the attack less noticeable. This method is

discussed in greater detail by(Papadimitratos & Jovanovic, 2008). The received signal at the target receiver can be modeled as:

$$r_{\text{SCER}}(t) = \sum_{i=1}^{N} a_i(t) \cdot c_i\big(t - (\tau_i + \Delta\tau_i)\big) \cdot \cos(2\pi f_{ci} t + \phi_i) + n(t) \qquad 22$$

Where:

N is the number of affected satellite signals.

$a_i(t)$ is the amplitude of the $i^{"th"}$ manipulated signal.

$c_i\big(t - (\tau_i + \Delta\tau_i)\big)$ is the PRN code of the "the " $i^{"th"}$ signal, delayed by $(\tau_i + \Delta\tau_i)$

τ_i is the natural propagation delay of the $i^{"th"}$ signal.

$\Delta\tau_i$ is the attacker's selectively introduced delay for the $i^{"th"}$ signal.

f_{ci} is the carrier frequency of the i^{th} signal.

ϕ_i is the phase offset of the i^{th} signal.

$n(t) \sim \mathcal{N}(0, \sigma^2)$ is the additive white Gaussian noise.

Unlike meaconing, where all signals are delayed uniformly, SCER allows each signal i to have an independent delay $\Delta\tau_i$. This selective delay enables the attacker to manipulate the position solution without affecting the timing solution. The attacker can craft a false position without triggering a sudden time jump, making detection harder.

Spoofing attacks can be further categorized by their complexity and the attacker's technical capabilities. According to(Humphreys et al., 2008), spoofing techniques can be classified into three main categories as described below.

Simplistic Spoofing approach involves the transmission of arbitrary GNSS-like signals without any attempt to synchronize with authentic satellite signals. This kind of attacks are easily detectable due to the evident discrepancies between the false and genuine sign. Intermediate Spoofing attacks maintain synchronization with legitimate GNSS

signals, making them harder to detect. The attacker generates counterfeit signals that align in frequency and timing with authentic satellite broadcasts, ensuring that the receiver's correlation process locks onto the deceptive signal. Sophisticated Spoofing category includes highly advanced methods, such as the use of multiple phase-locked intermediate spoofers. Such methods can manipulate the target receiver's position or timing without causing any obvious anomalies.

A notable example of sophisticated spoofing is the use of limpet spoofers. These are miniature spoofing devices that can be physically attached to the target device, such as a vehicle or timing instrument. By maintaining proximity to the target receiver, limpet spoofers can overcome many of the practical challenges of traditional spoofing attacks. However, their deployment requires direct physical access to the target receiver and advanced miniaturization, which remains a significant technical challenge.

3.4.4 Spoofing Challenges in Finland and the Nordic Region

Finland is facing more incidents of real GNSS interference, such as jamming and spoofing, because of its location at the borders of northern airspace and near the Baltic Sea. Such situations place serious hazards on air travel, safety at sea, and crucially important systems.

Numbers shows that there were over 1,200 cases of GPS interference in 2024, while only 239 were reported the year before. In eastern Finland, operation difficulties were reported, since commercial flights were either redirected or relied on alternative systems after GPS signals were unavailable. To solve the problem, those three airports, which are Joensuu, Savonlinna, and Lappeenranta, switched back to using radio-based navigation. According to Yle News, almost every airport in Finland among 14 affected in total has had to switch to alternative landing systems due to GNSS interference. The report cites around 2,800 GPS interference incidents in 2024, compared to just 200 incidents in 2023(*USU*, 2025c)

Likewise, issues in the maritime domain have been very similar. In the Gulf of Finland, Finnish Coast Guard recognized two types of GNSS disturbances affecting cargo vessels:, losing signals and showing incorrect positions. The occurrence of these anomalies made people worry about ships being safely guided and about the integrity of AIS data (Kauranen & Kauranen, 2024a).

As a result, countries have stepped up their readiness to deal with emergency contagion. Traficom assured people in a public safety bulletin that aviation is still safe because of extra safety features and new operation rules (*Aviation Prepared for GPS Interference*, 2024). Even so, the increasing and unrelenting disruption of GNSS shows that it is essential to have detectors that can recognize both unmodified and tampered signal environments almost instantly, could be through machine learning.

As these events show, GNSS spoofing is now a real threat that needs to be dealt with urgently and carefully watched by experts. In response to this need, the following section presents a Systematic Literature Review (SLR) of recent research focused on GNSS spoofing and jamming detection and mitigation. The review critically examines academic and industry efforts that leverage machine learning (ML), deep learning (DL), and reinforcement learning (RL) approaches, among others, to improve situational awareness and signal integrity in contested GNSS environments.

4 Existing Research on GNSS Spoofing & Jamming Detection and Mitigation

As the reliability of GNSS becomes increasingly critical in safety and infrastructure dependent applications, the threats posed by spoofing and jamming have gained growing attention from the research community. A broad range of Ai, ML, DL, and RL techniques have been explored to detect and mitigate these threats.

This chapter presents a systematic literature review (SLR) of recent research on AI/ML/RL based approaches for GNSS spoofing and jamming detection and mitigation. The review was conducted according to the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) guidelines. The chapter is structured as follows; Section 4.1 describes the methodology adopted for the review. Section 4.2 outlines the research questions that guided the selection and analysis of studies. Section 4.3 details the search strategy, while Sections 4.4 and 4.5 explain the screening, selection, quality assessment, and data extraction process. The results and thematic discussion are presented in Chapter 5.

4.1 Methods

The process that is adopted in this systematical literature review including planning, conduction and reporting has shown in the figure 13 below which has been in line with PRISMA to guarantee clarity, methodological rigor, and reproducibility.

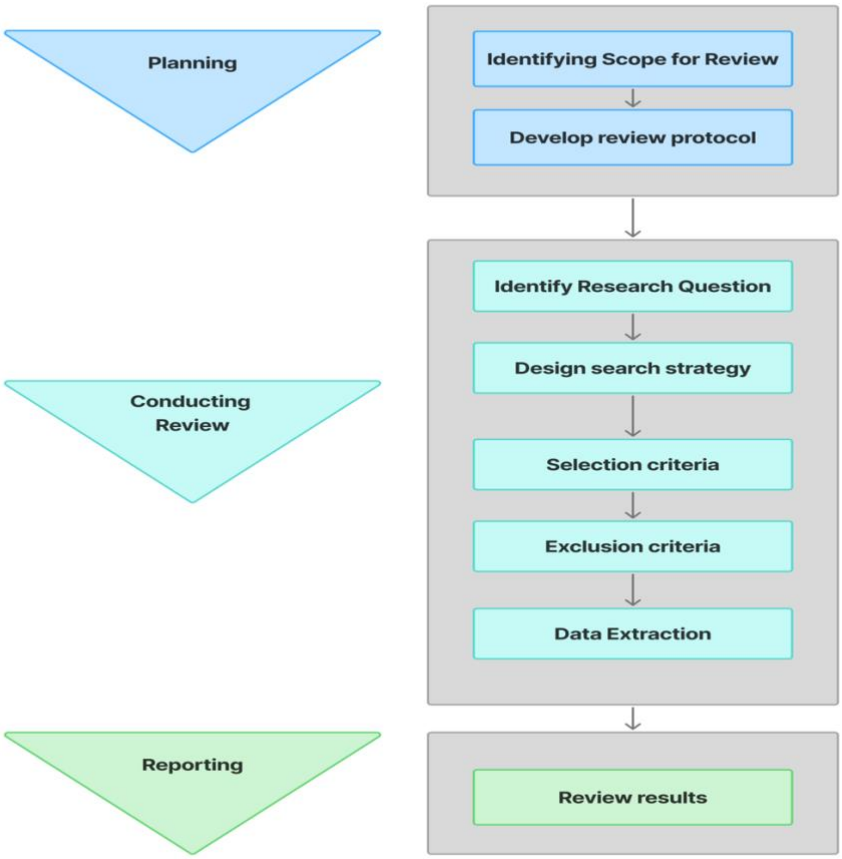

Figure 13 Systematic review process flowchart

From the flowchart the main three phase of the review process has shown which include planning, conduction report and results. First phase includes Identifying scope for review and develop review protocol. The second phase include Identifying research questions, design search strategy, selection criteria, exclusion criteria, data extraction and data synthesis. And the last phase includes reporting, reviewing results.

4.2 Research Questions (RQ)

This study focused on existing GNSS spoofing and jamming detection and mitigation studies using Al, ML, RL, and other techniques. Research questions are aligned with the scope of the existing research, the techniques that have been used, and the outcome of those techniques.

Table 2 Research Questions

RQs	Questions to classify studies	Motivation
RQ1	What techniques have been applied to detect or mitigate GNSS spoofing and jamming threats?	Identify the range of AI, ML, and RL methods utilized for GNSS spoofing and jamming detection and mitigation. This helps in mapping the techniques used and assessing their diversity.
RQ2	What types of attacks are most addressed in the literature?	Determine the types of GNSS spoofing and jamming attacks that are predominantly explored, enabling a better understanding of the focus of existing research.
RQ3	What are the commonly used datasets or Realtime testbeds utilized for evaluating AI/ML/RL techniques in spoofing and jamming.	Understand the data sources and test environments used for evaluating the effectiveness of AI/ML/RL techniques in GNSS security.
RQ4	What are the evaluation metrics and how do the methods compare in terms of effectiveness, robustness and accuracy?	Evaluate the performance of the proposed techniques based on key performance indicators,

RQs	Questions to classify studies	Motivation
		ensuring a clear view of their effectiveness.

Table 2 represents the study research questions (RQs) that have been used to conduct this systematic literature review (SLR) about the utilization of Artificial Intelligence (AI), Machine Learning (ML), and Reinforcement Learning (RL) methods in detecting and mitigating GNSS spoofing and jamming threats. The main goal of these research questions is to provide a structured view of where we stand regarding research on this topic at present times. RQ1 is concerned with analyzing the array of AI/ML/RL methods which have been utilized for GNSS security and summarizing the methodological landscape. This is useful in seeing what the types of techniques are most preferred, and the extent of variety pursued in the literature. RQ2 also investigates types of GNSS spoofing as well as jamming attacks as covered in the selected studies. By classifying types of the attack, this question gives an idea about the main threat scenarios, which were intensively studied by researchers, which allows us to reveal the most pressing problems in the field. RQ3 is concerned with studies of datasets and test environments in these studies. This question is crucial as the quality, type, and relevance of the datasets are directly indicative of the validity and generalizability of the proposed solutions. It also permits understanding if the researchers use simulated data, real world testbeds or hybrid solutions. Finally, the effectiveness of the proposed techniques is addressed by RQ4 which works on performance indicators such as accuracy, robustness, real-time ability, and validation methods. This enables a critical appraisal of the strengths and weaknesses of current approaches and a premise for identifying gaps in the literature and recommending directions for future research. All together, these research questions can act as an overarching framework for the SLR, to ensure a proper systematic and objective review of available studies in the area of GNSS spoofing and jamming detection and mitigation using AI/ML/RL approaches.

4.3 Search Strategy

4.3.1 Literature Resources

To provide a full coverage of recent and current research in the field of GNSS spoofing and jamming detection and mitigation with AI/ML/RL and other techniques, a systematic search was performed in two major and well-renowned academic databases.

- IEEE Xplore: It is well known for a very large set of quality research articles, conference proceedings and technical standards in the field of computer science, engineering and technology.
- Scopus: A multidisciplinary database providing opportunities to access numerous peer-reviewed scientific journals, proceedings of conferences and scientific literature.

These chosen databases were selected due to their importance to research in these fields and to their vast coverage in engineering, artificial intelligence and signal processing. To make sure that the most recent progress made in this field would be included, the search was limited to articles only published between the years 2016 and 2025. Only peer reviewed journal articles and conference papers were considered to keep the quality and credibility of the review.

4.3.2 Search Terms (Keywords)

The search process utilized a systematic approach, such as employing Boolean operators and carefully selected keywords designed explicitly illustrated in figure 14, to ensure comprehensive and relevant retrieval of publications. Keywords involved words related to GNSS technology, threats (spoofing and jamming), advanced computational approaches (AI, ML, RL). The search query run on both Scopus and IEEE Xplore as a search query has shown in the figure below.

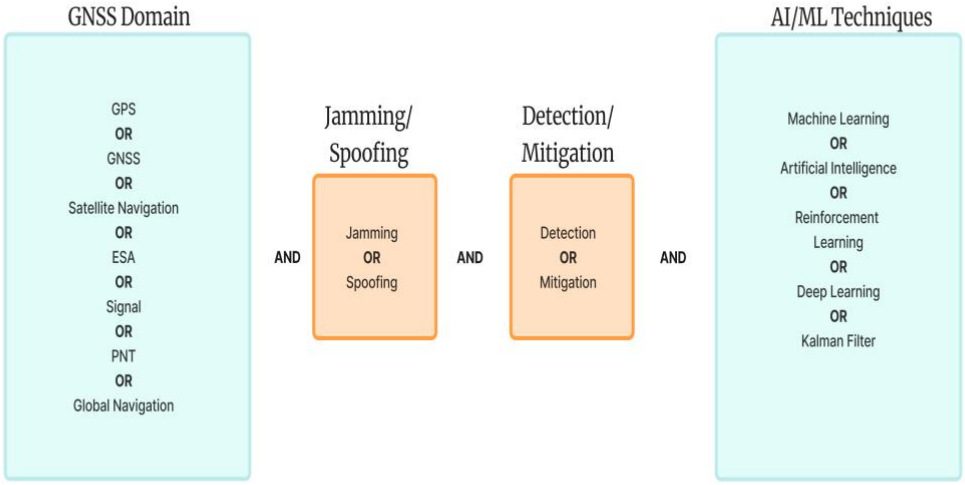

Figure 14 Paper Search String

In particular, the search had three main areas of concentration discussed below,

Application Domain: To retrieve studies in connection with global navigation satellite systems and the security with these systems, the search terms implied use of keywords such as GPS, GNSS, ESA, Signal, and Satellite Navigation. These terminologies guaranteed that the search targeted research on satellite-based positioning and navigation technologies, which are susceptible to a wide range of interference.

Threat Domain: To limit the search to studies that discuss intentional threats to GNSS systems the keywords “Spoofing” and “Jamming” were added. The use of such terms narrowed the search for research that seeks to investigate attacks that undermine the integrity and reliability of GNSS signals, the criticality to the scope of this review.

Computational Techniques Domain: The search also sought out studies using advanced computational methods for detection and mitigation of GNSS threats. The search was conducted using keywords, including “AI”, “Machine learning”, “Artificial intelligence”,

“Reinforcement learning”, “Deep learning”, and “Kalman Filter” to capture quite a large variety of approaches to artificial intelligence and machine learning.

To narrow down the search further and stay relevance to the technical focus of this review, the search results were deliberately limited to unrelated subject as such as neuroscience, agriculture, business, arts, and medicine. This was done using the exclusion capacities of chosen databases (the IEEE Xplore and Scopus) that restricted access to irrelevant studies. Only peer-reviewed journal articles and conference papers were used in the search to keep the studies reviewed within quality and credible parameters. This approach ensured that only peer reviewed studies that were properly done were taken for consideration. Research with affiliations from China were excluded from the search specifically. The choice was made to keep an international viewpoint in the review to prevent possible bias problems. This organized and meticulously refined method of searching meant that the final set of studies were very relevant, specific and of high quality. The application of Boolean operators and selective use of keywords facilitated an even mix of research reports while leaving out irrelevant or of poor-quality articles.

4.4 Search Process & Selection

To ensure the use of Ai and machine learning techniques in GNSS jamming and spoofing this systematic literature review (SLR) was carried out in line with the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA), thus promoted transparency and methodological rigour, as well as reproducibility. The process was broken down into four separate phases. Identification, Screening, Eligibility, and Inclusion illustrated in figure 15.

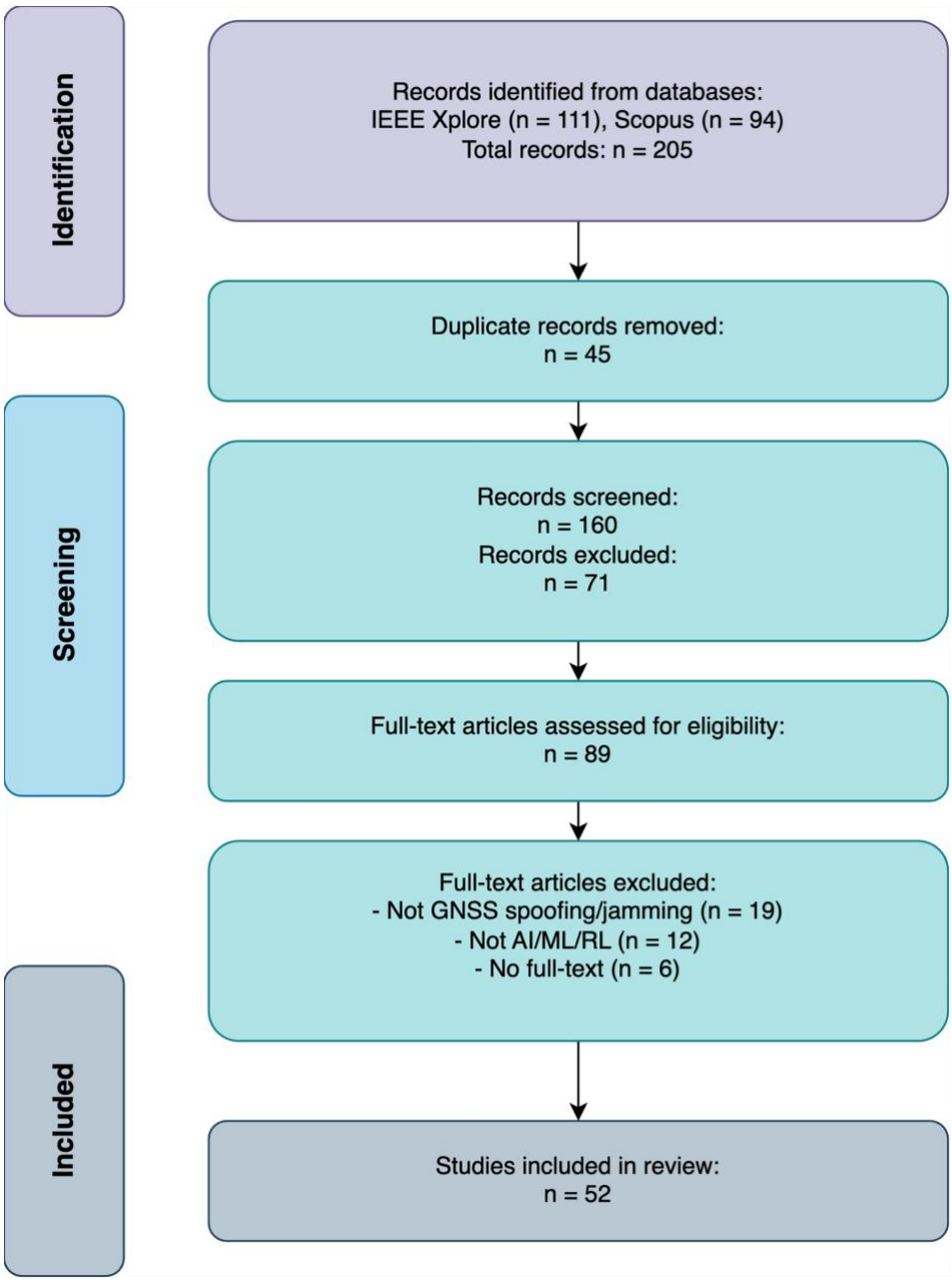

Figure 15. PRISMA Flow Diagram of Study Selection Process for GNSS Spoofing and Jamming Detection Review

Identification: The search process began with the formulation of a comprehensive search query, designed to capture all relevant studies on GNSS spoofing and jamming detection or mitigation using Artificial Intelligence (AI), Machine Learning (ML), and Reinforcement Learning (RL). The query was executed separately on two major academic

databases. This search initially retrieved a total of 205 studies, with 111 from IEEE Xplore and 94 from Scopus.

Screening: The retrieved studies were imported into the Zotero reference management tool for efficient management and deduplication. Removal of duplicate entries using Zotero's automatic duplicate detection (based on title, DOI, and publication metadata). A manual cross-check was performed to ensure accurate duplicate removal. After duplicate removal, 160 unique studies remained. These studies were then subjected to a preliminary screening of titles and abstracts to ensure relevance to the review's objectives.

Eligibility: A detail title and full text abstract review was perfomed on the 89 remaining studies, applying the inclusion and exclusion criteria.

4.4.1 Inclusion and exclusion criteria

In inclusion criteria, to include only the relevant studies and ensure high quality below criteria were applied:

- Articles that explicitly addressing GNSS spoofing and/or jamming detection or mitigation.
- Application of AI, ML, RL, or hybrid computational techniques.
- Published in English between 2016 and 2025.
- Peer-reviewed journal articles or conference papers. Except one preprint paper was included.

In exclusion criteria, to simplify the receiving process of irrelevant studies usage, the following criteria have been applied.

- Research exclusively focused on GNSS positioning (PNT) without addressing spoofing or jamming.
- Studies with ML techniques used with signals, 5G, UWB where no GNSS jamming or spoofing are absent.

- Articles without accessible full texts or abstracts.
- Review papers, editorials, letters, or book chapters.

By following the above procedures, a total of 52 studies relevant to the issue were identified for being incorporated in the SLR process (all papers included are in appendix). Lastly there were quality assessment criteria applied.

4.4.2 Quality Assessment

A rigorous quality assessment procedure was followed to ensure that only strong, methodologically sound studies were in scope for this systematic review. Important assessment criteria presented in table 3, corresponding to the research questions were determined and implemented, detailed questions are shown in the table below.

Clarity of AI/ML/RL Techniques: Studies were rated based on whether they clearly defined the AI, ML, or RL approaches employed for GNSS spoofing or jamming identification, and countermeasures, reporting such details as the network architecture, training mechanism and algorithm selection. This directly supported Research Question 1 (RQ1).

Classification of the spoofing or jamming attacks studied in each research work. Each study was evaluated in terms of clarity on how many types of GNSS spoofing or jamming events they described at the signal level, meaconing, replay, or brute-force jamming. This aspect was associated with Research Question 2 (RQ2).

Dataset or Testbed Specification: Researchers assessed the transparency of reporting data sources (real-world, simulated or available to the public) and the arrangement of the testbeds in the studies. This criterion supported Research Question 3 (RQ3).

Performance Evaluation Metrics: The evaluation tested whether studies explained performance measures such as accuracy, precision, recall, F1-score, robustness in

adequate detail, and contain comparative analysis to a baseline method, guiding Research Question 4 (RQ4).

Scoring was done on a scale of 0-1 for each criterion (0=Not Met, 0.5= Partially Met, 1=Fully Met) and the greatest attainable score was 4.0. All studies with values under 2.0 were considered of inadequate quality and excluded from the final review.

Table 3 Summary of Quality Assessment Criteria

Question	Quality Assessment Criteria (QAC)	YES	Partial	No
Q1	Does the study directly address GNSS spoofing or jamming?			
Q2	Are the AI/ML/RL techniques clearly described?			
Q3	Does the study provide clear performance metrics (e.g., accuracy, precision)?			
Q4	Are the datasets clearly described?			

This structured assessment ensured that only robust, clearly reported, and methodologically sound studies were included in the review. This assessment is based on the paper output not measuring the actual strength of the studies.

4.5 Data Extraction and Data Synthesis

A structured data extraction form was developed to ensure consistency and transparency across the reviewed studies. A summary of the extraction fields is presented in table 4. This form was used to systematically extract and record relevant details from each included paper. The collected data facilitated both quantitative and qualitative synthesis, enabling trend analysis, categorization, and comparative evaluation across diverse AI-based GNSS spoofing and jamming solutions.

Table 4. Data Extraction Template Used in the Systematic Literature Review

Column No.	Field Name
1	ID
2	DOI
3	Title
4	Authors
5	Techniques
6	Spoofing/Jamming Type
7	Dataset/ Testbed
8	Validation Metrics
9	Validation Method
10	Accuracy
11	Study Application
12	Mitigation/ Detection
13	References
14	Paper Type
15	Year

5 Literature Review's Findings and Discussions

This section presents and discusses the findings this literature review. We begin with a general overview of the final selected studies, followed by detailed analyses aligned with each research question (RQ1–RQ4). Each subsection interprets the results regarding the specific research question within the broader context of GNSS spoofing and jamming security. Where relevant, references to related works contextualize and support the observations.

A major factor in understanding the development of research in every domain is looking at its timeline. Thus, the research years of every selected study were studied to discover the rising trend in the use of AI, ML, DL, RL and other techniques used in detecting or dealing with GNSS spoofing and jamming.

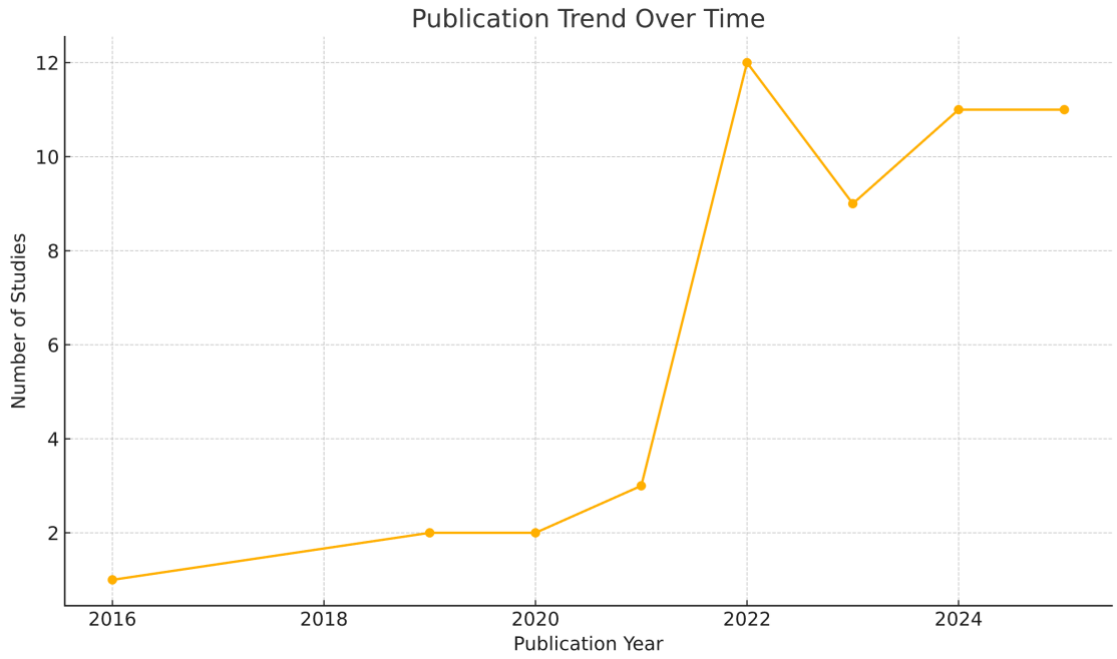

Figure 16. Publication Trend Overtime

The dataset spans from 2016 to 2025 a period of ten years, encompassing 52 studies. As illustrated in Figure 16, early contributions were minimal, with one study published in 2016, followed by two studies each in 2019 and 2020, and a modest increase to three studies in 2021. This initial phase reflects a period of low activity and exploratory

research, where the problem domain was still emerging within the AI and GNSS communities.

In 2022, research has shown growth with 12 publications, showing that the field was beginning to expand rapidly. The trend unfolded when 9 studies were conducted in 2023, followed by another peak in 2024 and 2025, each with 11 studies. Because of an increase in incidents, greater awareness of security issues, and the growing use of GNSS in things like autonomous vehicles, drones, and critical infrastructure, the GNSS spoofing and jamming issue gained notice from 2021 upward.

This trend indicates a strong and sustained growth in the field, reflecting a shift from theoretical or simulated studies to more practical and data-driven implementations. The steep increase also coincides with wider availability of datasets (e.g., TEXBAT), simulation environments, and open-source AI frameworks, which have lowered entry barriers for experimental validation and comparative analysis. This pattern underlines the increasing importance of AI-enhanced GNSS security mechanisms and reinforces the relevance of this SLR in synthesizing the latest progress in the domain.

Out of the 52 final studies a total 43 papers (84.3%) were published in peer-reviewed journals. 6 papers (11.8%) were published in conference proceedings, and 2 papers (3.9%) were published as preprints which has shown in figure 17.

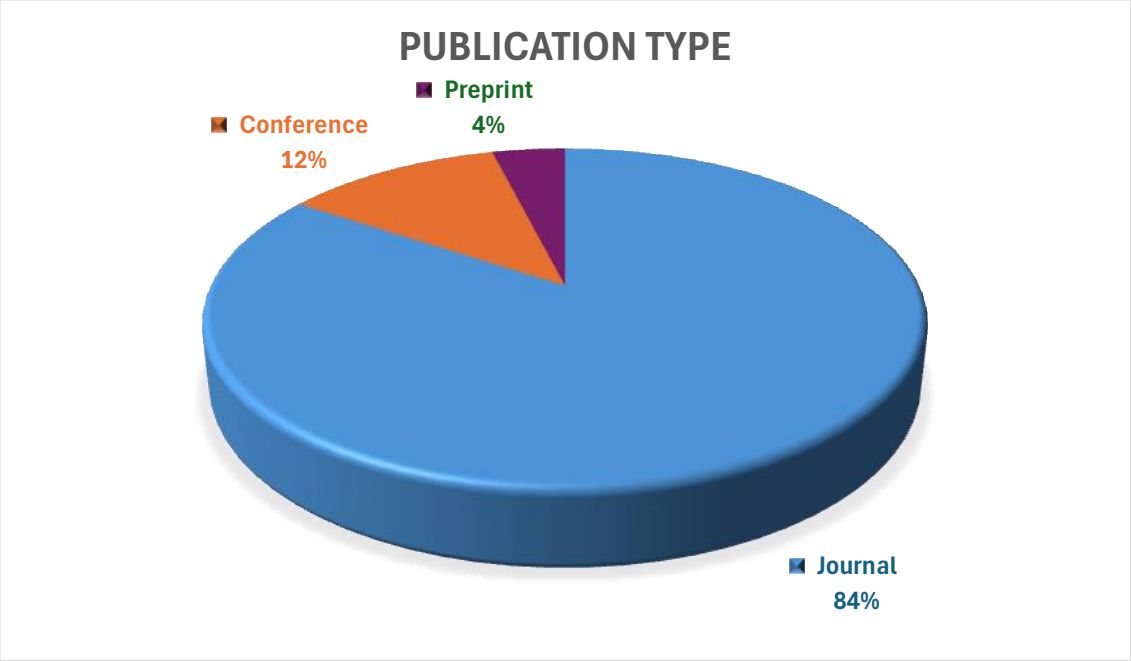

Figure 17 Publication type that are included in these studies

In terms of methodological orientation, nearly all studies followed an experimental research approach, typically involving the training and evaluation of models using either simulated or real-world GNSS spoofing and jamming datasets. A majority of the reviewed papers reported the use of validation metrics such as accuracy, precision, recall, and F1-score to assess the performance of AI-based detection systems. However, only a small subset of papers conducted hardware in the loop or real-time testing, highlighting a persistent gap between simulation-based validation and deployment-ready solutions. The dataset diversity also varied. While some studies employed publicly available datasets like TEXBAT or Spirent-based testbeds, others relied on proprietary or ad hoc simulation environments. This heterogeneity in data sources may impact the generalizability and comparability of results across the literature.

All included studies met a minimum quality threshold defined during the review protocol, with each paper being assessed on criteria such as methodological clarity, dataset specification, and reproducibility. Consequently, the final dataset is considered representative of high-quality, focused research in AI-driven GNSS threat detection.

5.1 Types of Machine Learning Techniques

From this review we've identified a diverse array of artificial intelligence (AI) techniques employed for the detection and mitigation of GNSS spoofing and jamming threats illustrated in Figure 18. These methods were categorized into five primary groups as Classical machine learning (ML), deep learning (DL), reinforcement learning (RL), ensemble/hybrid approaches, and statistical models. This classification reflects the computational strategies and learning paradigms utilized across the reviewed literature.

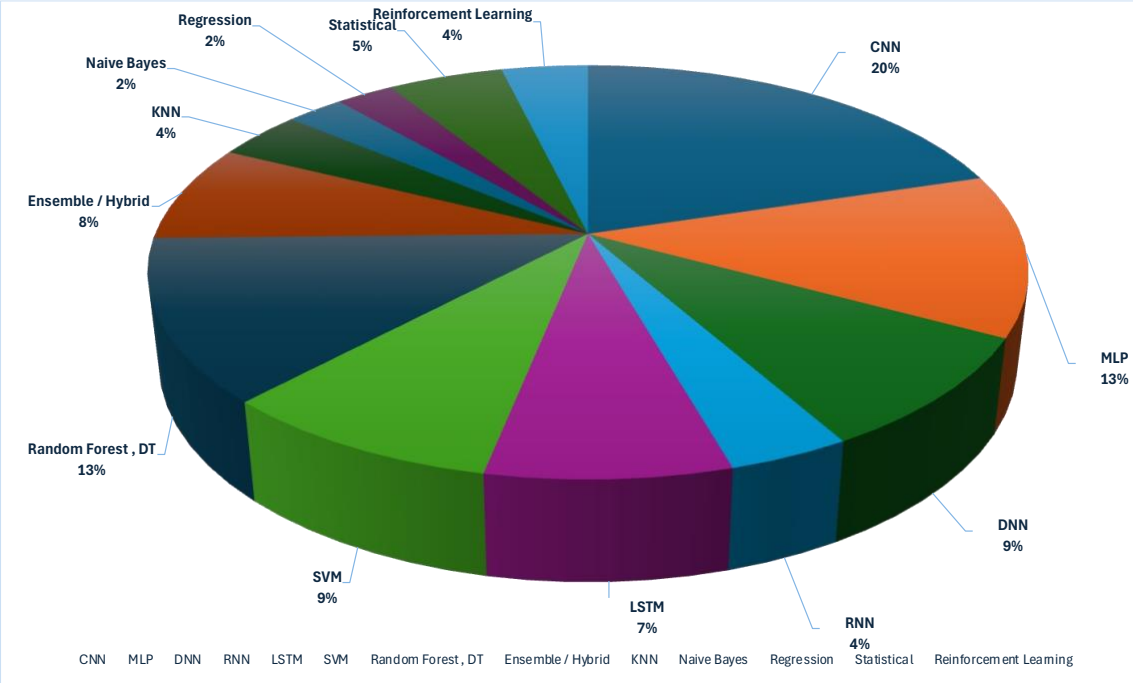

Figure 18 Summary of mostly utilized techniques in GNSS jamming and spoofing

5.1.1 Classical Machine Learning Techniques

Traditional ML techniques remain prevalent in GNSS security research due to their simplicity, efficiency, and ease of deployment. Among these, decision trees (DT), random forests (RF), and support vector machines (SVM) were the most applied. Decision trees were used in 6 studies, often as part of ensemble classifiers due to their interpretability and low computational cost. For example, (I. E. Mehr & F. Dovis, 2025)[P3] leveraged DTs for low-complexity classification of interference signals. Random forests appeared in 10

papers and were favored for their robustness and resistance to overfitting, particularly in high-dimensional feature spaces. Studies such as (H. Farkhari et al., 2024) [P4] and (Nayfeh et al., 2023) [P48] demonstrated their applicability in both spoofing and jamming classification tasks. SVMs were employed in 7 studies, including (Filippou et al., 2023) [P5] and (Ali et al., 2024) [P39], primarily for binary classification problems such as spoofed versus authentic GNSS signals. Although effective, SVMs were more commonly used in earlier studies with smaller datasets or limited computational resources.

Less frequent were K-nearest neighbor (KNN, 3 papers), Naive Bayes (NB, 2 papers), and regression-based models (2 papers). These were generally used for benchmarking or in lightweight detection systems where computational simplicity was a design constraint.

5.1.2 Neural Network-Based Techniques (Deep Learning)

Deep learning techniques, particularly neural networks (NNs), constituted most AI-based approaches in GNSS spoofing and jamming detection. These models were valued for their ability to autonomously learn hierarchical features from complex input data, such as time-domain signals or spectrograms. Convolutional neural networks (CNNs) were the most dominant architecture, appearing in 16 studies. CNNs were primarily applied to transformed GNSS signals such as short-time Fourier transforms (STFT) or spectrograms for spoofing/jamming classification. Notable implementations include (Korium et al., 2024) [P29] and (Ghanbarzadeh et al., 2024) [P49], where CNNs achieved high accuracy on both simulated and real datasets. Multilayer perceptrons (MLPs) and deep neural networks (DNNs) were reported in 10 and 7 studies, respectively. While often used interchangeably, some authors distinguished DNNs as deeper versions of MLPs. (Dang et al., 2021) [P12–P14] employed MLP-based ensembles for GPS spoofing detection, whereas (H. Farkhari et al., 2024)[P4] utilized DNNs for multi-class jamming detection. Recurrent neural networks (RNNs) and long short-term memory (LSTM) networks were less frequent but crucial for time-series analysis. LSTM networks, used in 6 studies, offered superior capability in learning temporal dependencies in GNSS signal behavior,

particularly in dynamic spoofing contexts (e.g., (Alanazi, 2024)[P44], (Badar et al., 2025) [P15]). A single study [P44] implemented bidirectional RNNs (BiRNNs), highlighting their potential for future applications. Transfer learning models such as ResNet, AlexNet, and EfficientNet were found in 4 papers, usually in conjunction with spectrogram inputs. These models benefited from pre-trained weights on large-scale datasets and were repurposed for GNSS security tasks (e.g., (Li et al., 2025)[P31], (Rijnsdorp, 2023) [P40]). It is noteworthy that some studies used general terms such as "ANN" or "DNN" without detailed architectural specifications. Where possible, these were reclassified based on the described model structure.

5.1.3 Ensemble and Hybrid Methods

Ensemble and hybrid models have emerged as effective solutions that combine the strengths of multiple algorithms. These models were applied in 14 of the reviewed studies and often yielded improved robustness and generalization performance. Gradient boosting frameworks such as XGBoost and LightGBM were used in 6 papers, often to complement or compare against neural models. Hybrid combinations such as CNN+LSTM or RF+MLP were observed in 5 studies, offering enhanced feature extraction and temporal learning capabilities. Examples include the work of (Badar et al., 2025)[P15] and (Alhoraibi et al., 2024)[P47], where multimodal fusion contributed to better detection accuracy. Voting and stacking ensembles were implemented in 3 papers, typically to combine outputs from diverse classifiers like SVM, RF and MLP to reduce variance across test scenarios.

5.1.4 Reinforcement Learning Techniques

Reinforcement learning (RL), though limited in occurrence, showed promising applicability in adaptive and real-time GNSS defense strategies. RL was explicitly applied in two studies. (Dasgupta et al., 2022)[P7] employed deep Q-networks (DQN) to train an agent for spoofing mitigation in a dynamic GNSS environment. Additionally, (N. I. Mowla, 2020) [P10] proposed an Adaptive Federated Reinforcement Learning (AFRL) framework

tailored for jamming attack defense in Flying Ad Hoc Networks (FANETs), which are decentralized communication networks composed of Unmanned Aerial Vehicles (UAVs). Addressing the unique vulnerabilities of FANETs such as high mobility, sparse node density, and communication constraints, the authors developed a model-free Q-learning strategy combined with an adaptive epsilon-greedy exploration-exploitation policy. Their approach is guided by an on-device federated jamming detection mechanism, allowing each UAV to independently detect jamming and locally learn optimal spatial retreat actions. Instead of transmitting raw data, UAVs periodically share model updates to refine a global policy collaboratively, reducing overhead and preserving privacy. Simulation results on both the CRAWDAD dataset and a custom ns-3 simulated FANET jamming dataset demonstrated that AFRL significantly reduced jammer location hop counts and improved detection accuracy by 39.9% over fully distributed mechanisms. This work illustrates the feasibility and effectiveness of integrating reinforcement learning with federated detection systems in dynamic, adversarial FANET environments where centralized models and traditional approaches fall short. These studies suggest that RL can provide situational awareness and policy optimization in environments where static models underperform.

5.1.5 Statistical Techniques

Statistical methods played a supporting role across several studies, particularly in signal preprocessing, baseline detection, or model augmentation. Kalman filtering, threshold-based detection, and rule-based decision logic appeared in seven studies. For instance, (Mao, 2016)[P26] applied an RNN-Kalman hybrid for interference mitigation, while (Viana et al., 2022) [P45] used statistical decomposition techniques alongside deep models for jamming detection. Although not often used as standalone classifiers, statistical models contributed essential components for signal smoothing, anomaly detection, and decision logic. Their lightweight nature makes them suitable for embedded GNSS receivers and real-time mitigation systems.

The majority of recent GNSS spoofing and jamming research has gravitated toward deep learning techniques, particularly CNNs and MLPs, owing to their capacity to learn rich feature representations from complex signal data. Classical ML methods like SVMs and random forests remain relevant due to their simplicity and interpretability. Ensemble and hybrid models are increasingly favored for their robustness, while reinforcement learning is emerging as a promising direction for adaptive, policy-based mitigation. Statistical models continue to play a foundational role in preprocessing and lightweight detection.

5.2 GNSS Threat Studies by Type

5.2.1 Spoofing-Focused Studies

GNSS spoofing presents a critical vulnerability in satellite-based navigation systems. It entails transmitting counterfeit signals to mislead receivers into computing false position, velocity, or timing estimates. Figure 19 shows that among 52 studies selected for this systematic review, 37 papers (72.5%) directly addressed spoofing detection and/or mitigation. This section synthesizes their technical contributions, datasets, evaluation strategies, and application domains.

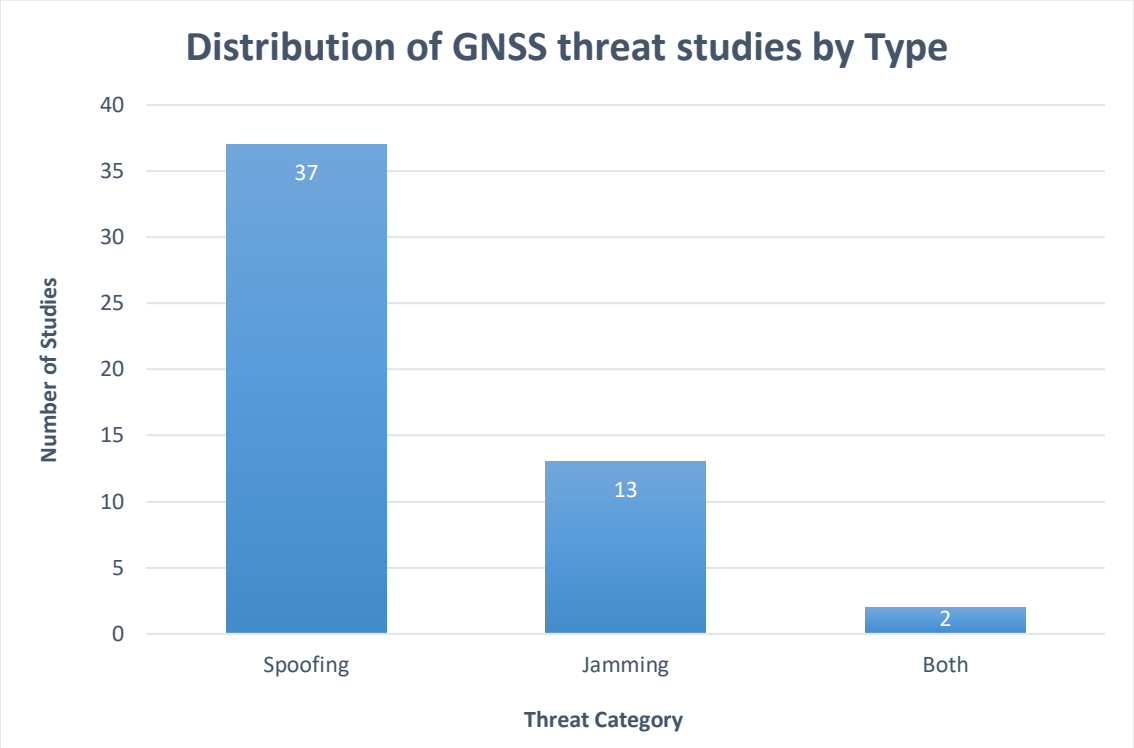

Figure 19 Distribution of selected studies by GNSS threat type (Spoofing, Jamming, Both).

5.2.1.1 Technical Methodologies

Spoofing detection systems have evolved from lightweight machine learning classifiers to sophisticated deep learning and hybrid architectures. Traditional ML models remain popular for their simplicity and suitability for embedded systems. [P13] employed a Multi-Layer Perceptron (MLP) trained on features such as CN_0 and Doppler shift, achieving up to 93% accuracy depending on the number of base stations used (Dang et al., 2021). (Nayfeh et al., 2023) [P48] combined MLP and statistical features to create a low-power detection model, reaching over 93% accuracy using real spoofing data.

Deep learning models especially Convolutional Neural Networks (CNNs) dominate recent literature due to their ability to process raw or spectrogram transformed GNSS data. (Badar et al., 2025) [P15] proposed DeepSpoofNet, a CNN-BiLSTM hybrid trained with ANOVA-based feature selection, which achieved an F1 score of 0.99 on simulated UAV

scenarios. Mehr and Dovis [P3] similarly used CNNs to classify spoofing attacks with 96% accuracy based on signal images.

Afroozeh et al. [P12] introduced Antiference, which used ResNet50 with fingerprinting to distinguish spoofing types. Validated on real and simulated datasets, their system reported 100% classification accuracy across all jamming and spoofing classes(Afroozeh et al., 2023).

Recurrent neural networks were explored for time-series spoofing detection. Alanazi [P44] employed LSTM-GRU structures on spectrogram sequences and reached high classification performance (>95%) without requiring manual feature extraction. Alhoraibi et al. [P47] integrated CNN, LSTM, and GAN modules, enabling spoofing detection and synthetic data generation, with accuracy exceeding 98%. Few-shot learning was proposed by Ott et al. [P34], who used a ResNet18 triplet-loss framework to generalize spoofing detection to new environments. GANSAT by Roy et al. [P23] employed GANs for spoof simulation and CNNs for detection, yielding 94.5% accuracy and enhanced robustness.

Other models included the fingerprint-based system by Oligeri et al. [P38] and the lightweight CNN ensemble by Ali et al. [P39], both demonstrating >98% accuracy on spoofed GNSS datasets.

5.2.1.2 Validation Metrics and Evaluation Strategies

Across studies, accuracy was the predominant evaluation metric. Models such as DeepSpoofNet [P15], Antiference [P12], and Ott et al. [P34] all reported accuracy values ≥96%. Several studies, including Alhoraibi et al. [P47] and Alanazi [P44], incorporated precision, recall, and F1-score to provide a fuller picture of classification performance. Afroozeh et al. [P12] uniquely measured performance in terms of false alarm rate (FAR) and C/N_o improvement, highlighting signal quality impacts. (Ott et al., 2025)[P34] emphasized AUC and generalization ability in few-shot scenarios, which is crucial for real-

world deployment. Meanwhile, Dang et al. [P13] reported latency (20 ms inference time), critical for embedded applications.

Despite these improvements, very few studies reported performance under adversarial spoofing, or evaluated computational efficiency in constrained environments.

5.2.1.3 Dataset Types and Sources

The datasets employed over spoofing focused studies varied significantly in source, format, and realism, influencing the generalizability and validity of the proposed models. Many of the studies specifically 18 papers, including [P1], [P5], [P8], [P9], [P13], [P14], [P15], [P17], [P18], [P19], [P20], [P25], [P29], [P36], [P37], [P42], [P44], and [P45 utilized purely simulated datasets. These were typically generated using software-defined radios or GNSS signal emulators. Simulated datasets offer the advantage of full control over attack conditions and signal parameters, enabling the construction of tailored spoofing scenarios and systematic evaluations. However, they often lack the environmental variability and signal imperfections that characterize real-world spoofing conditions, which may limit their effectiveness in representing operational environments.

Conversely, nine papers including [P6], [P7], [P12], [P22], [P23], [P38], [P43], [P47], and [P48] utilized real-world GNSS spoofing datasets. These were collected through over-the-air transmission or controlled replay of spoofed signals using custom testbeds. Such datasets provide essential realism by incorporating actual signal propagation conditions, multipath effects, and hardware-induced noise, thereby increasing the ecological validity of the evaluation.

In a subset of four studies [P2] (Dang et al., 2023), [P33] (Nabi et al., 2024), [P35] (Shang et al., 2021), and [P39] (Ali et al., 2024) researchers employed the TEXBAT dataset. Developed by the University of Texas at Austin, TEXBAT is a benchmark real-world dataset that offers labeled GNSS spoofing recordings captured in field conditions using sophisticated hardware spoofers. These recordings are particularly valuable due to their

authenticity, consistent formatting, and public availability, which make them suitable for reproducible experimentation and model comparison under controlled yet realistic attack scenarios.

A hybrid approach was adopted in several studies, including [P21], [P24], [P28], [P30], and [P31], wherein simulated data was supplemented with real-world traces to improve model generalization. For example, Ott et al. [P34] blended simulated spoofing samples with empirical recordings to train a ResNet18-based few-shot model capable of detecting previously unseen spoofing attacks. This fusion strategy offered the benefits of controlled training environments while ensuring the model's adaptability to operational GNSS signal dynamics.

5.2.1.4 Application Domains

Spoofing detection systems reviewed in this study were applied across a wide spectrum of GNSS dependent domains. A prominent application area was unmanned aerial vehicle (UAV) navigation. In papers such as [P15] by Badar et al., [P44] by Alanazi, [P47] by Alhoraibi et al., and [P48] by Nayfeh et al., the focus was on designing lightweight and real-time spoofing detection architectures that could operate onboard UAV platforms. These systems prioritized low latency inference and computational efficiency to ensure flight safety in contested radio frequency environments.

In addition to UAVs, embedded and mobile platforms received significant attention. Dang et al. [P13] and Cibecchini et al. [P49] specifically targeted resource constrained GNSS receivers, demonstrating spoofing detection systems capable of operating on Raspberry Pi and other low-power computing environments. These implementations emphasized short inference times, compact memory footprints, and compatibility with consumer grade GNSS chips.

Another critical application domain was GNSS timing infrastructure, where spoofing attacks can compromise time synchronization in power grids, telecom networks, and

financial systems. In this context, Shang et al. [P35] investigated the mitigation of time synchronization attacks using a hybrid machine learning model and Kalman filter fusion, showing how AI-based classifiers can enhance the reliability of GNSS time references in security critical infrastructure.

Lastly, several studies including [P1], [P3], [P12], and [P33] proposed general purpose spoofing detection frameworks aimed at GNSS signal authentication across a variety of receiver types. These models were designed for adaptability, supporting both stationary and mobile applications and potentially integrating with firmware-level security modules in commercial GNSS devices.

5.2.2 Jamming Focused Studies

GNSS jamming involves the deliberate emission of radio frequency interference to block or degrade satellite signal reception. Unlike spoofing, which manipulates position or time estimates, jamming simply denies service by overpowering authentic signals. Fig.12 illustrates 13 studies exclusively addressed jamming detection or mitigation. These studies utilized a diverse array of machine learning and deep learning techniques, from convolutional and recurrent neural networks to ensemble classifiers and reinforcement learning algorithms.

5.2.2.1 Technical Methodologies

Convolutional neural networks (CNNs) were a popular choice for classifying jamming signals, particularly in the time-frequency domain. Mehr et al. (2025) [P3] developed a CNN-based model that used spectrogram transformations of GNSS signals to distinguish between jamming and non-jamming conditions. Their method achieved 99.69% accuracy with a ResNet architecture, along with fast inference times and strong performance on confusion matrix metrics (I. E. Mehr & F. Dovis, 2025)

(W. El-Shafai et al., 2025) [P11] designed an ensemble classifier that combined CNNs with decision tree-based learners Random Forest and Extra Trees to improve classification of various jamming types. Their architecture was trained on real-world GNSS jamming data and yielded 99.81% accuracy, outperforming individual classifiers on both precision and F1-score metrics.

Gradient boosting algorithms were explored in the vehicular context by Kumar et al. [P16], who used CatBoost a fast, scalable decision tree ensemble for jamming detection in Internet of Vehicles (IoV) networks. Their system, validated in a simulated MATLAB/Simulink VANET environment, achieved 99.91% accuracy while maintaining low false detection rates and demonstrating resilience under high mobility scenarios.

Recurrent neural networks were applied by Mao(2016) [P26], who proposed an RNN-based architecture enhanced with an unscented Kalman filter (UKF) for signal filtering. The model was evaluated on derivative-fused GPS signal data under dynamic jamming conditions, achieving significant signal-to-noise ratio (SNR) improvements over baseline filters +8.25 dB over LMS and +7.2 dB over traditional Kalman filtering methods (Mao, 2016).

In a novel approach, Mowla et al. [P10] implemented Adaptive Federated Reinforcement Learning (AFRL) for anti-jamming in satellite IoT networks. The system used Q-learning agents distributed across edge devices to learn jamming avoidance strategies cooperatively. Their method achieved 85.98% jamming detection accuracy and delivered up to 82.01% average performance improvement in convergence speed and energy consumption compared to centralized RL strategies (Mowla et al., 2020).

5.2.2.2 Evaluation Strategies and Dataset Use

Across these studies, evaluation metrics included accuracy, precision, recall, F1-score, mean squared prediction error (MSPE), cumulative reward, and SNR improvement. While CNN-based models such as those by Mehr and Dovis [P3] and El-Shafai et al. [P11]

focused on classification metrics, models like Mao [P26] emphasized signal quality recovery metrics, highlighting the diversity of goals in jamming research.

Dataset quality and authenticity varied across the studies. El-Shafai et al. [P11] used real jamming data from recorded GNSS interference, enhancing ecological validity. Mowla et al. [P10], Kumar et al. [P16], and Mehr and Dovis [P3] relied on simulated jamming scenarios, including multi-path environments and SDR-generated waveforms. Mao [P26] validated on real GPS datasets processed for interference simulation, making his results particularly robust.

5.2.2.3 Application Domains

Several application contexts were targeted. Kumar et al. [P16] focused on vehicular networks, where the system's rapid detection of RF interference is crucial to avoid communication and navigation blackouts. Mowla et al. [P10] addressed satellite-IoT networks with distributed RL agents, suitable for space-constrained and delay-sensitive communication nodes. El-Shafai et al. [P11] and Mehr and Dovis [P3] proposed general-purpose jamming classifiers for integration into GNSS receivers or navigation firmware. Mao's work [P26], with its emphasis on signal recovery and filtering, fits well within GNSS positioning engines in both mobile and embedded contexts.

The reviewed jamming studies present a technically diverse set of detection and mitigation strategies tailored to a range of operational contexts, from vehicular networks to satellite-IoT systems. With models based on CNNs, RNNs, CatBoost, and Q-learning, the field has achieved high accuracy levels under both synthetic and real interference conditions.

5.2.3 Studies Addressing Both Spoofing and Jamming

While most GNSS security research treats spoofing and jamming as separate threat domains, a limited number of studies aim to identify both within a unified detection

framework. This is critical for operational resilience, as adversaries may alternate or combine these tactics to maximize disruption. Fig.12 illustrates, two studies Afroozeh et al. (2023) [P12] and Ghanbarzadeh et al. (2024) [P50] addressed both spoofing and jamming.

Afroozeh et al. [P12] introduced Antiference, an integrated GNSS interference identification system designed for robust classification of spoofing and jamming signals. The framework leverages a ResNet50 deep convolutional neural network trained on spectrogram representations of GNSS signal captures. The model performs end to end learning, automatically extracting features from time-frequency data without manual engineering. Antiference incorporates a fingerprinting mechanism that generates neural activation-based signatures from known signal types, enabling detection of previously unseen interference by comparing test samples to an evolving internal database. The training data includes both real-world jamming recordings, collected using a custom testbed in Austria and simulated spoofing signals, processed into spectrograms using Short Time Fourier Transform (STFT). The model achieved perfect classification (100%) for common jamming types such as AM, FM, CW, and AGW, and performed strongly on more complex chirp-based signals. Spoofing detection was validated using structured classification and similarity matching, and the authors reported low false alarm rates during real-time experimentation (Afroozeh et al., 2023).

Ghanbarzadeh et al. [P50], on the other hand, proposed a multi-model comparison framework that evaluated both classical machine learning and deep learning classifiers for GNSS spoofing and jamming detection. Their system analyzed two real-world datasets: one containing GPS spoofing attacks on UAV scenarios, and another comprising jamming attacks transformed into 2D spectrogram images. For spoofing, the authors tested models including XGBoost, Random Forest, and SVM, as well as CNN and ResNet18 deep learning variants. Spoofing detection involved multi-level classification across four spoofing intensity categories, while jamming classification targeted six classes of RF interference. ResNet18, when pretrained on ImageNet and fine-tuned on

GNSS spectrograms, achieved the highest performance among all tested models, with jamming classification accuracy reaching 98.93% and spoofing accuracy exceeding 94%. The study emphasized robust cross-validation, SMOTE balancing, and extensive visual performance diagnostics including ROC curves and confusion matrices. Unlike Antiference, which relies on adaptive fingerprinting, this approach focused on hard classification and comparative benchmarking of traditional and modern classifiers across the same input space (Ghanbarzadeh, Soleimani, & Soleimani, 2024).

Together, these two studies exemplify complementary approaches to multi-threat GNSS security. Afroozeh et al. [P12] advanced an evolutionary, fingerprint-based detection architecture geared toward deployment in dynamic and adversarial environments. In contrast, Ghanbarzadeh et al. [P50] emphasized model benchmarking and explainability, offering a comparative analysis of classical vs. deep learning methods on labeled, real-world datasets. Both studies turned raw GNSS signals into spectrograms (which are like colorful graphs of how the signal changes over time and frequency). Then they trained deep learning models to analyze those spectrograms and detect whether the signal was normal, spoofed, or jammed. This shows that looking at both time and frequency together is a powerful way to find interference.

6 Reinforcement Learning Based Spoofing Detection

The overall methodological pipeline for GNSS spoofing detection using Reinforcement Learning is illustrated in Figure 20. The process encompasses dataset preparation, feature engineering, custom environment design, agent training, and model evaluation using a Deep Q-Network (DQN). Each block represents a key stage in the development and deployment of the spoofing detection system.

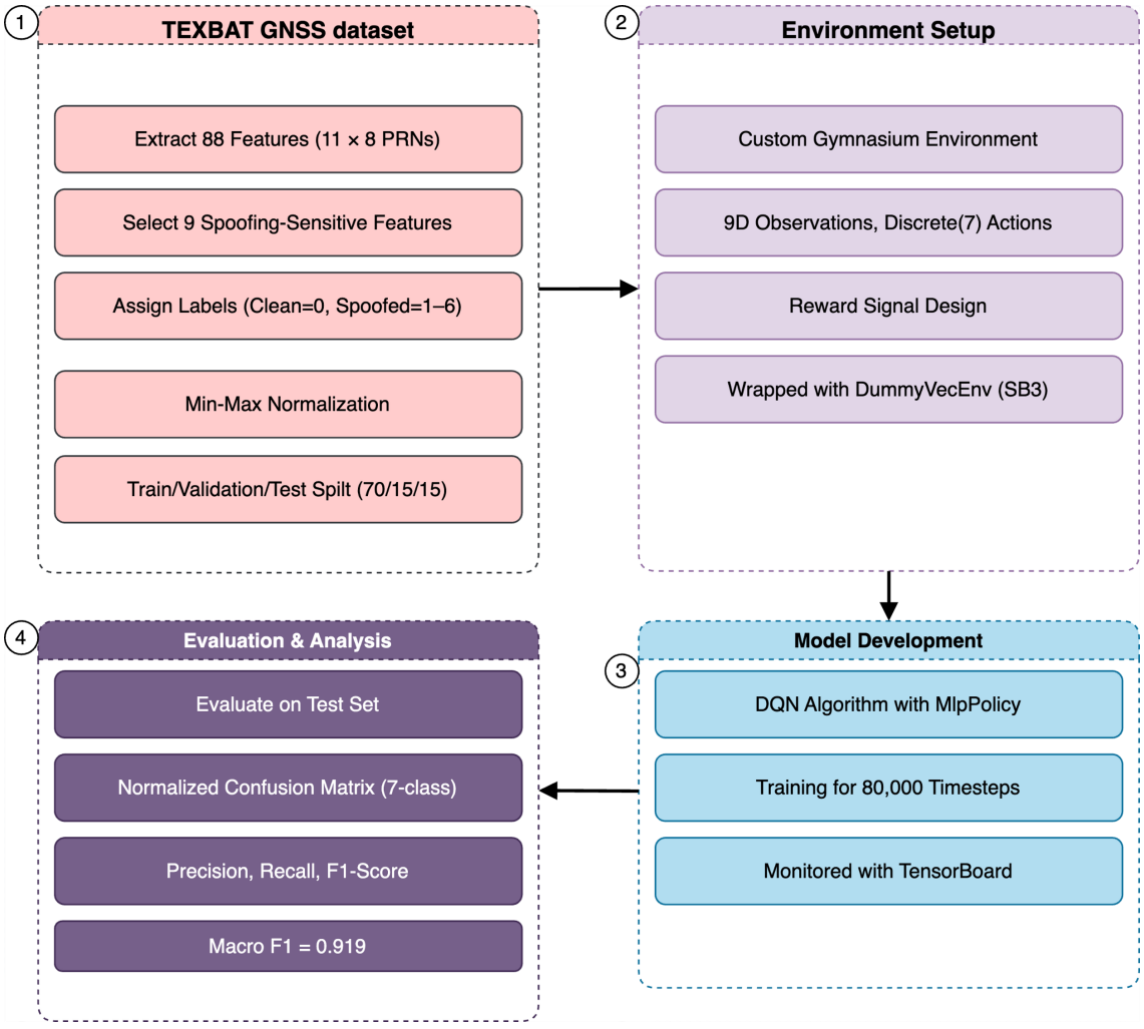

Figure 20. Reinforcement Learning-based GNSS Spoofing Detection Methodology

6.1 Dataset description

The dataset used in this research is the Texas Spoofing Test Battery (TEXBAT) provided by Radio navigation Laboratory at the University of Texas at Austin(*TEXBAT – Radionavigation Laboratory*, n.d.). The dataset consists of GNSS signal recordings of high-fidelity based on both valid and spoofed situations. Furthermore, recordings cover different scenarios that reflect real-world spoofing attacks faced in static and dynamic receiver settings which provides potential to apply Reinforcement Learning to build learning-based spoofing detection mechanism which is the aim of this thesis. The original CSV files obtained from the GitHub repository provided raw GNSS tracking data without column headers. As part of the preprocessing pipeline, meaningful column names were manually assigned to each block of 11 features corresponding to individual PRNs. These column names such as acq_doppler_hz, prompt_i, cn0_db_hz, and pseudorange_m were inferred based on accompanying documentation and feature definitions provided in the GitHub repository. This step was crucial for ensuring consistent feature referencing and enabling subsequent selection of spoofing-relevant tracking metrics(*GitHub - Seki5405/Gnss_spoof_detector*, n.d.).

6.2 Spoofing Scenarios

In order to implement Reinforcement Learning based GNSS Spoofing Detection System, the static datasets of TEXBAT which consist of cleanStatic, cleanDynamic, ds1, ds2, ds3, ds4, ds7, and ds8 and each dataset is based on different spoofing scenario. Here, static means fixed antenna recording and dynamic means antenna mounted on a moving object (e.g. drone or vehicle). In this research, static datasets have used to build spoofing detection system to get better performance across 6 different datasets of static including ds1, ds2, ds3, ds4, ds7, ds8, where each consists of different nature of spoofing attacks.

Table 5. Characteristics of TEXBAT Dataset

Scenario	Dataset	Receiver Type	Spoofing Type	Description	Used in Thesis
1	ds1	Static	None	Static Switch	Yes
2	ds2	Static	Time	Static Overpowered Time Push	Yes
3	ds3	Static	Time	Static Matched-powered Time Push	Yes
4	ds4	Static	Position	Static Match-Powered Position Push	Yes
5	ds5	Dynamic	Time	Dynamic Overpowered Time Push	No
6	ds6	Dynamic	Position	Matched-powered Position Push	No
7	ds7	Static	Stealth/Phase	Static Sparse Stealth	Yes
8	ds8	Static	Replay/Phase	Static Replay Attack	Yes

In table 5, time-push spoofing represents delayed satellite signals which causes noise to be added in clock estimate of receiver which affect the PVT time in dataset. Next, position-push spoofing adds offset to code-phase on all satellites equally that shifts the computed position.

Furthermore, power-matching spoofing attack makes the amplitude of signal much higher than authentic signal. Whereas matched-power means signal is blended at same amplitude as the authentic signal which makes this attach very hard to detect compared to over-powered spoof mentioned earlier.

In addition, ds7 dataset contains most challenging spoofing attacks because spoofer aligns amplitude and carrier phase to avoid sudden change in Doppler or C/N_o and then slowly shifts code-phase to generate clock offset without threshold detectors being triggered. Finally, ds8 dataset consists of signals in which extra sophistication is added based on real-time navigation-bit estimation, covering the loophole of data-bit which detectors look out for.

6.3 Feature Set

Observation Features (*GitHub - Seki5405/Gnss_spoof_detector*, n.d.)
Below are the descriptions of observation features in TEXBAT dataset which define what receiver observes at a higher-level.

Carrier_Doppler_hz: This represents the Doppler shift of carrier signal that is measured in hertz. The Doppler shift occurs because of relative motion between the receiver and the satellite.
Carrier_phase_cycles: This represents the accumulated phase of carrier signal that is measured in cycles.
Flag_valid_pseudorange: A Boolean indicator which shows whether the pseudorange is valid or not. True means authentic and false means potential threat due to spoofing or interference in signal.
Pseudo-Random Noise number (PRN): An identifier for each satellite that allows receiver to identify signals from different satellites.
Pseudorange_m: The distance between the receiver and the satellite that is measured in meters.
RX_time: The timestamp which shows when the signal was captured by receiver.
Time of Week at current symbol in seconds (TOW_at_current_symbol_s): It consists of Time of Week measured in seconds, corresponding to current navigation data symbol.

Tracking Features: Here are the descriptions of tracking features that are part of signal tracking loop, reflecting how well the receiver is locked onto satellite signals.

CN0_SNV_dB_Hz: This consists of Carrier-to-Noise density ratio that indicates the quality of signal, means higher the value, stronger is the signal.

Pseudo-Random Noise number (PRN): An identifier for each satellite that allows receiver to identify signals from different satellites.

PRN_start_sample_count: The sample count where tracking of particular PRN starts in dataset.

Prompt_I: The component of signal called In-phase (I), a core measurement used in signal correlation.

Prompt_Q: The component of signal called quadrature-phase (Q), another core measurement used in signal correlation.

abs_E / abs_L / abs_P / abs_VE / abs_VL: The absolute values of early, late, prompt, very early, and very late correlator outputs, altogether these values signify how well the receiver is locked onto the signal.

acc_carrier_phase_rad: Accumulated carrier phase that shows total phase shift observed over time.

carr_error_filt_hz: It consists of filtered carrier frequency error.

carr_error_hz: It consists of raw carrier frequency error.

carrier_doppler_hz: Latest estimate of Doppler frequency shift.

carrier_doppler_rate_hz: Rate of change of Doppler frequency in hertz which indicate how fast or slow the Doppler is changing.

carrier_lock_test: Metric to test if carrier loop is still locked.

code_error_chips: It consists of Code phase error in chips

code_error_filt_chips: It consists of Filtered code error.

code_freq_chips: It consists of code frequency in chips.

code_freq_rate_chips: Rate of change of code frequency in chips.

6.4 Analysis of TEXBAT Data and Data Preprocessing

The raw data used in this study originates from the Texas Spoofing Test Battery (TEXBAT) and preprocessed into csv file by (*GitHub - Seki5405/Gnss_spoof_detector*, n.d.) a benchmark dataset for spoofing detection research. Each CSV file contains GNSS tracking information organized into 8 blocks of 11 features, corresponding to 8 tracked PRNs per receiver snapshot. This yields a total of 88 columns per row, later parsed into structured features. The base 11 features extracted per PRN are shown in the figure 21.

```
# Column names for each PRN block (11 features per PRN)
columns = ['channel', 'prn', 'acq_doppler_hz', 'acq_doppler_step', 'fs',
           'prompt_i', 'prompt_q', 'cn0_db_hz', 'carrier_doppler_hz', 'pseudorange_m', 'rx_time']
```

Figure 21. Features per PRN

For this visual analysis, the following six features were selected for closer visual inspection, as they have shown consistent disruption in the presence of spoofing: acq_doppler_hz, prompt_i and prompt_q, cn0_db_hz, carrier_doppler_hz, pseudorange_m.

Each feature was plotted against rx_time, which represents the receiver timestamp in milliseconds. The clean dataset (cs.csv) was used as a baseline for normal signal behavior, while spoofed datasets were used to highlight spoofing-induced anomalies. Clean signals are plotted in blue, while spoofed signals appear in red.

1. prompt_i – Clean vs. Spoofed (cs.csv vs. ds2.csv)

The in-phase correlation output, prompt_i, represents the signal power aligned with the local replica. In clean GNSS conditions, this value oscillates smoothly within a predictable range. We observe sharper transitions and irregular spikes in the spoofed dataset, especially at the beginning of the tracking period has shown in Figure 22. These abrupt variations are consistent with signal injection, where the spoofed signal may fail to maintain realistic correlation coherence.

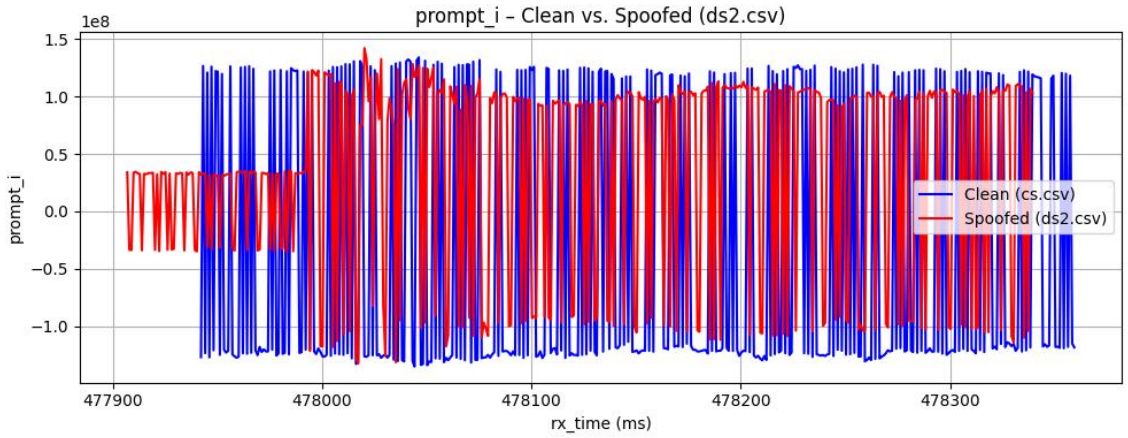

Figure 22. prompt_i Clean vs Spoofed

2. prompt_q – Clean vs. Spoofed (cs.csv vs. ds2.csv)

The quadrature correlation value, prompt_q, ideally centers around zero and reflects the orthogonal noise component. Figure 23 shows in the spoofed dataset, prompt_q shows bursty, high-amplitude fluctuations that do not appear in the clean data. These signal anomalies suggest inconsistencies in the spoofed signal's phase or power, possibly due to improper carrier alignment.

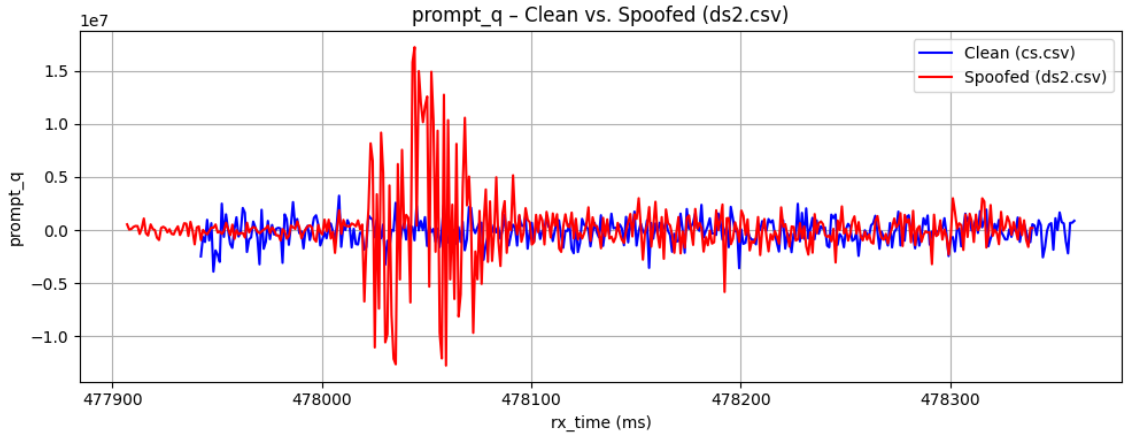

Figure 23. prompt_q Clean vs Spoofed

Carrier-to-Noise Ratio (cn0_db_hz) – cs.csv vs. ds3.csv

The C/N_0 ratio is a direct metric of signal quality. The clean signal shows consistent power levels within expected bounds. From Figure 24, In ds3.csv, a noticeable degradation is observed, with dips in C/N_0 values, including a sharp drop below 40 dB-Hz a threshold typically associated with signal degradation or loss of lock. Such behavior may arise when spoofers fail to maintain consistent signal power or cause interference. These disruptions may also occur when a spoofer abruptly replaces authentic signals without proper power matching.

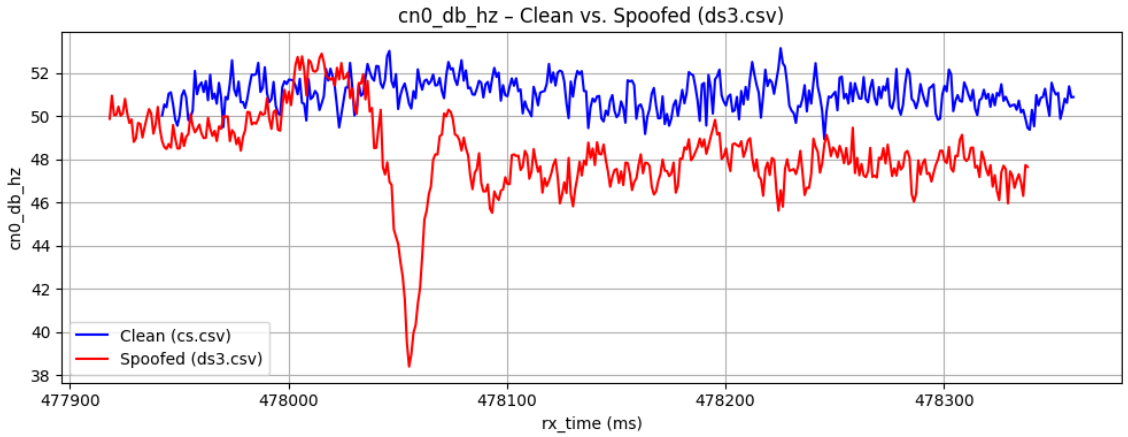

Figure 24. Carrier-to-Noise Ratio Clean vs Spoofed

Carrier Doppler Shift (carrier_doppler_hz) – cs.csv vs. ds4.csv

Carrier Doppler measures fine grained frequency shifts due to relative motion. Clean data displays a smooth, gradually changing Doppler profile. However, the spoofed trace shows abrupt discontinuities and discontinuous jumps, which are physically implausible given orbital mechanics shown in figure 25. These inconsistencies indicate an unsynchronized or stationary spoofing source.

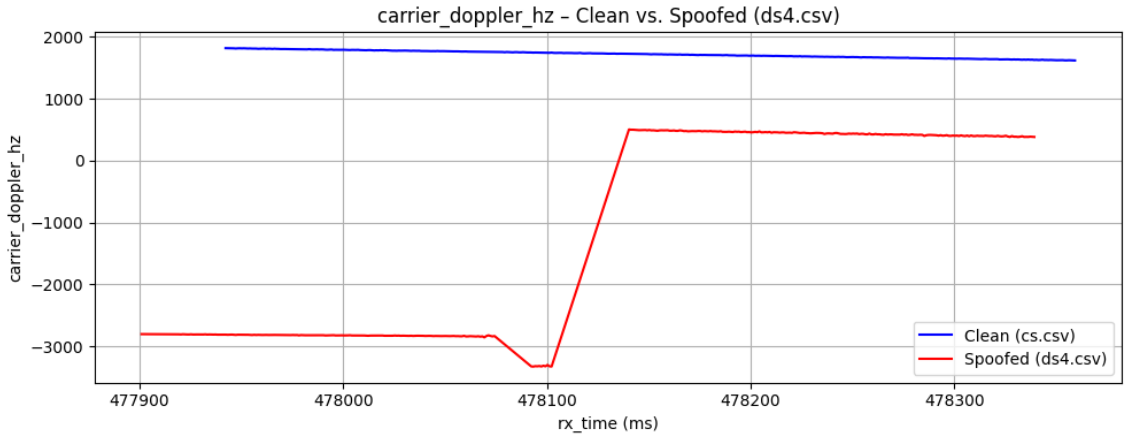

Figure 25. Carrier Doppler Shift Clean vs Spoofed

Pseudorange (pseudorange_m) – cs.csv vs. ds7.csv

Pseudorange represents the estimated satellite-to-receiver distance, a critical input to positioning. Figure 26 shows in the spoofed data, observe a sudden jump in range at the beginning of the tracking interval, where the spoofed signal exhibits an unrealistically large offset from the clean baseline. This kind of behavior is indicative of time delay manipulation, often used in spoofing to shift the receiver's perceived position.

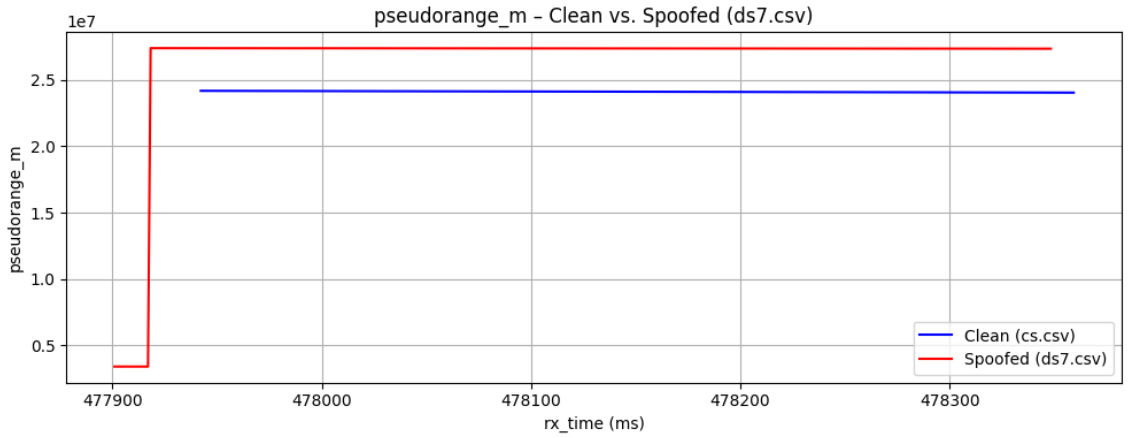

Figure 26. pseudorange_m - Clean vs Spoofed

These visual comparisons provide compelling evidence that spoofing introduces significant distortions in key GNSS signal features. Each selected feature shows a characteristic pattern of degradation, ranging from abrupt transitions to unnatural

offsets, which are largely absent in the clean data. These observations validate the hypothesis that meaningful spoofing signatures exist in raw tracking features and motivate their use in learning-based spoofing detection frameworks.

6.4.1 Feature Extraction Strategy

To transform this raw format into a usable structure for learning-based classification, a systematic PRN-wise feature extraction was conducted. The first 100 rows were removed from each file to eliminate transient or unstable initialization values. Each file was then partitioned into 8 blocks, corresponding to PRNs 0 through 7. Within each block, the following 9 raw GNSS signal tracking features were retained for downstream processing which has shown in the table 6.

Table 6. Selected features

Features	Description
acq_doppler_hz	Doppler frequency estimate during acquisition (Hz). Indicates initial satellite relative velocity.
acq_doppler_step	Step size used during Doppler search in acquisition. In this dataset, it is constant.
prompt_i	In-phase prompt correlation output. Reflects how well the receiver tracks the incoming signal.
prompt_q	Quadrature-phase prompt correlation output. Orthogonal component of the signal used for tracking.
Cn0_db_hz	Carrier-to-noise density ratio (dB-Hz). Measures signal quality.
carrier_doppler_hz	Doppler shift estimate during carrier tracking (Hz). Reflects satellite-receiver relative motion.
pseudorange_m	Estimated pseudorange between satellite and receiver (meters). Used for position computation.
rx_time	Receiver timestamp (milliseconds) when the observation was made. Encodes temporal progression.
fs	Sampling frequency of the GNSS receiver front-end (Hz). Indicates signal capture rate.

These features were selected due to their relevance in representing signal dynamics, correlation behaviour, and timing inconsistencies factors known to be affected by spoofing attacks. For each PRN block, the selected features were extracted and labeled according to the corresponding spoofing condition (0 for clean; 1–6 for spoofed scenarios). This yielded a large corpus of individual PRN samples, each represented by a 9-dimensional feature vector and a class label. The merged data was stored in a Pandas DataFrame for convenient manipulation and inspection.

6.4.2 Normalization and Dataset Partitioning

To standardize the feature scales and facilitate effective convergence during training, Min–Max normalization was applied to rescale all nine features into the [0,1] interval. This transformation is defined as:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{\min}} \qquad 23$$

where x denotes the original feature value, and x_{min} d x_{max} e the minimum and maximum values of that feature across the entire dataset. This step was crucial to ensure that features with large numerical ranges (e.g., prompt_i, pseudorange_m) did not dominate the learning process, and that all dimensions were treated equitably by the neural network during reinforcement learning.

Table 7. Min–Max Scaling Summary of GNSS Tracking Features

Feature	Min (Original)	Max (Original)	Min (Scaled)	Max (Scaled)
acq_doppler_hz	-3500.0	2750.0	0.0	1.0
acq_doppler_step	0.0	0.0	0.0	0.0
fs	25000000.0	25000000.0	0.0	0.0
prompt_i	-181320896.0	179767472.0	0.0	1.0
prompt_q	-23515810.0	22283530.0	0.0	1.0

Feature	Min (Original)	Max (Original)	Min (Scaled)	Max (Scaled)
cn0_db_hz	29.43359	56.457352	0.0	1.0
carrier_doppler_hz	-3354.626403	2612.930195	0.0	1.0
pseudorange_m	-1153282.431191	29746500.471437	0.0	1.0
rx_time	477912.98	478359.22	0.0	1.0

Table 7 illustrate the feature value distributions before and after normalization, respectively. As expected, the normalization process compresses all features into a common numerical range while preserving their distributional shape and modality.

6.4.3 Splitting into Train, Validation, and Test Sets

To assess model performance fairly and prevent overfitting, the dataset was split into three subsets using stratified sampling:

- 70% for training
- 15% for validation
- 15% for final testing

This was implemented in two successive stages using train_test_split() from the scikit-learn library. First, the normalized dataset was split into an 85% temporary subset and a 15% held-out test set. Then, the 85% temporary set was further divided using a test_size=0.1765 to ensure the correct final proportions of 70/15/15.

```
# Train, Validation, Test Split (70/15/15)
X_temp, X_test, y_temp, y_test = train_test_split(X_scaled, y, test_size=0.15, stratify=y, random_state=42)
X_train, X_val, y_train, y_val = train_test_split(X_temp, y_temp, test_size=0.1765, stratify=y_temp, random_state=42)
```

Figure 27. Train, Test and Validation spilt

The stratified approach preserved the class distribution across all subsets, ensuring that each spoofing condition remained proportionally represented throughout the training

and evaluation phases. In the next section Reinforcement Learning Environment Designing explained.

6.5 Reinforcement Learning Environment Design

To enable autonomous GNSS spoofing detection, a custom reinforcement learning (RL) environment was developed that frames the spoofing classification task as a single-step, stateless interaction problem. The goal of the agent is to classify individual GNSS signal observations each corresponding to a satellite PRN at a given time step into one of seven classes (one clean and six spoofed). This design adheres to the OpenAI Gym interface standards and ensures seamless integration with Stable-Baselines3 for DQN-based learning.

6.5.1 Environment Overview

The environment was implemented as a subclass of gym.Env and encapsulates the core dynamics of the spoofing detection task. Each interaction is modeled as a single-step episode, where the agent observes a 9-dimensional feature vector extracted from a specific PRN signal instance and selects a corresponding class label from a finite action space. The environment responds with a scalar reward and immediately terminates the episode.

Internally, the environment is parameterized by a preprocessed feature matrix $\boldsymbol{X} \in R^{n \times d}$ and a label vector $\boldsymbol{y} \in \{0,1, \ldots, 6\}^n$ where n is the number of samples, and $d = 9$ is the number of normalized tracking features details in **section 6.4.1.** Each row of X corresponds to a unique PRN time instance, and each label in $\boldsymbol{y}$ denotes the spoofing class, with 0 representing clean signals (cs.csv) and values 1 to 6 denoting spoofed datasets (ds1–ds4, ds7, ds8).

6.5.2 Observation and Action Spaces

The observation space is defined as a continuous 9-dimensional box constrained within the normalized interval [0,1]:

$$S = [0,1]^9 \tag{23}$$

This space captures the scaled GNSS signal tracking features used to characterize each PRN snapshot. The nine features used are acq_doppler_hz, acq_doppler_step, fs, prompt_i, prompt_q, cn0_db_hz, carrier_doppler_hz, pseudorange_m, and rx_time.

The action space is a discrete set of seven classes:

$$A = \{0,1,2,3,4,5,6\} \tag{24}$$

Here, each action represents the agent's classification of the observed signal as belonging to one of the spoofing scenarios or clean. This setup effectively transforms the multi-class spoofing detection problem into a decision-making task suitable for discrete action-value learning.

6.5.3 Reset Function

The reset() method is responsible for initializing a new episode. It randomly samples an index $i \sim \mathcal{U}(1,n)$ retrieves the corresponding observation vector $\boldsymbol{s_t} = \boldsymbol{X_i}$ and returns it to the agent. A random seed can be specified for deterministic sampling during evaluation. The method ensures that each episode is independent of the previous one:

$$reset(\quad): \quad \text{Sample } i \sim \mathcal{U}(1,n), \quad \boldsymbol{s_t} = \boldsymbol{X_i} \tag{25}$$

This approach guarantees statistical independence across episodes, allowing the DQN to learn from a diverse set of GNSS signal conditions.

6.5.4 Step Function and Reward Mechanism

The step(action) function compares the agent's selected action $a_t \in \mathcal{A}$ with the true class label y_i. If the prediction is correct $(a_t = y_t)$, the agent receives a reward of +1; otherwise, it is penalized with −1. Each episode is terminated immediately after the action is taken:

$$r(s_t, a_t) = \begin{cases} +1, & if\ a_t = y_{\mathrm{i}} \\ -1, & otherwise \end{cases} \quad (26)$$

This reward structure is simple yet effective for training a Deep Q-Network (DQN), which must infer class boundaries based on subtle statistical differences in the spoofing sensitive features (e.g., cn0_db_hz, prompt_i, carrier_doppler_hz).

The environment also returns an info dictionary containing debugging information such as the ground truth label, agent action, and classification correctness. The return signature of the step function is consistent with Gymnasium's newer API.

```
FUNCTION step(action):
    label ← ground truth label at current index
    correct ← 1 IF action equals label ELSE 0
    reward ← +1 IF correct = 1 ELSE -1
    done ← TRUE  // since each sample is one step
    info ← dictionary containing:
        "label"  → label
        "action" → action
        "correct" → correct (1 or 0)

    observation ← feature vector at current index

    RETURN (observation, reward, done, False, info)
```

In this function

- done = True: because each sample is treated as a one-step episode
- truncated = False: episodes do not end due to time limits
- info = {"label": y_i, "action": a_t, "correct": $(a_t = y_i)$}

6.5.5 Justification of Stateless Formulation

The stateless, single-step formulation is deliberate. By eliminating state transitions and memory, the environment forces the DQN agent to rely entirely on spatial (rather than temporal) properties of the feature vectors. This isolates the classification task and evaluates whether spoofing can be detected from instantaneous GNSS signal snapshots. This design is particularly well-suited to real-time GNSS signal monitoring applications, where each time step must be evaluated independently due to dynamic spoofing threats. Moreover, it simplifies training and improves convergence stability in high-dimensional feature spaces.

In future work (see Chapter 7), this design may be extended to include stateful temporal transitions or sequential sampling strategies, allowing the agent to exploit temporal context for more nuanced spoofing detection.

6.5.6 Integration with Stable-Baselines3

To facilitate compatibility with RL training frameworks, the environment was wrapped using DummyVecEnv, which allows vectorized batch interaction with Stable-Baselines3 algorithms. This vectorization enables efficient gradient updates and streamlined logging via TensorBoard. The environment was trained using the DQN agent with MlpPolicy, further described in Section 6.6.

6.6 Deep Q-Network (DQN) Model Architecture and Training

To perform GNSS spoofing classification based on raw signal features, a Deep Q-Network (DQN) was implemented using the Stable-Baselines3 framework. The DQN agent was

trained to classify each individual GNSS signal snapshot into one of seven spoofing scenarios, including clean and six spoofed classes. This section details the DQN model architecture, training pipeline, hyperparameter configuration, and learning strategy used in this study.

6.6.1 Model Architecture

The neural network architecture adopted for the DQN agent is based on the default MlpPolicy provided by Stable-Baselines3. It consists of a feedforward multi-layer perceptron (MLP) comprising two hidden layers. The network receives as input a 9-dimensional normalized GNSS signal feature vector, as described in Section 4.2. Each hidden layer contains 64 neurons, and the ReLU activation function is applied after each linear transformation. The final output layer consists of seven units corresponding to the Q-values of each discrete action (i.e., spoofing class label).

The architecture of the online and target Q-networks is identical and can be described as follows:

$$\begin{gathered} x \in R^{9} \quad (\text{normalized input vector}) \\ h_1 = \mathrm{ReLU}(W_1 x + b_1), \quad W_1 \in R^{64\times 9} \\ h_2 = \mathrm{ReLU}(W_2 h_1 + b_2), \quad W_2 \in R^{64\times 64} \\ Q(x,\cdot) = W_3 h_2 + b_3, \quad Q \in R^{7} \end{gathered} \tag{27}$$

Here, $\mathrm{Q(x, a)}$ denotes the estimated Q-value for action a given the input feature vector x. This structure enables the agent to approximate a mapping from GNSS feature vectors to spoofing class decisions and is trained using Q-learning with experience replay and target networks finds more in Chapter 2.4.4.

6.6.2 Environment Initialization and Vectorization

Although the environment is stateless meaning there are no transitions between time steps the reinforcement learning framework is still applicable because each observation-action-reward triplet is independently useful for learning a Q-function.

The training, validation, and testing environments were instantiated separately using the DummyVecEnv wrapper to allow batched execution and compatibility with Stable-Baselines3. A fixed random seed (SEED = 42) was applied to ensure deterministic behavior during data sampling and weight initialization, enabling reproducible results.

6.6.3 DQN Agent Configuration

The DQN model was configured with the following hyperparameters, based on empirical tuning and prior research best practices has shown in table 8 below.

Table 8. DQN Agent Configuration

Hyperparameter	**Value**
Policy architecture	MlpPolicy
Learning rate	1e – 3
Discount factor ($\boldsymbol{\gamma}$)	0.99
Reply buffer size	10,000
Learning starts 500 steps	500 steps
Batch size	64
Train frequency	Every step
Target update interval	500 steps
Exploration fraction	0.3
Initial $\boldsymbol{\varepsilon}$ (epsilon)	1.0
Final $\boldsymbol{\varepsilon}$ (epsilon)	0.01
Total training timestamps	80,000

These values from table 8 ensure a balance between sample efficiency and stability of training. The use of a large replay buffer and soft exploration decay helps the agent avoid overfitting early samples while maintaining sufficient coverage of the state-action space.

6.6.4 Exploration and Exploitation

Despite the stateless nature of the custom environment where each input is treated as an independent classification problem, the concepts of exploration and exploitation remain relevant and integral to the learning process.

Exploration occurs via the ε-greedy policy, where the agent initially selects actions (class labels) at random to sample a diverse range of outcomes. This randomness ensures that the agent does not prematurely converge to suboptimal decisions. Exploitation is realized as the agent begins to select actions with the highest estimated Q-values, thereby favoring decisions it has learned to associate with correct classifications.

In this one-step environment, exploitation does not refer to learning long-term reward strategies, but rather to the immediate classification of GNSS signal snapshots based on learned feature patterns. This design is especially well suited to real-time spoofing detection, where decisions must be made per-sample without reliance on temporal memory.

6.6.5 Model Training and Evaluation Strategy

Model training was performed over 80,000 timesteps using the training environment (train_env). During training, periodic evaluation was conducted using the EvalCallback mechanism from Stable-Baselines3. Every 5,000 timesteps, the agent's performance was evaluated on the validation environment (val_env), and the best-performing model (based on average episode reward) was saved to disk. The following configuration was used:

```
eval_callback = EvalCallback(
    val_env,
    best_model_save_path='./best_model/',
    log_path='./dqn_gnss_tensorboard/',
    eval_freq=5000,
    deterministic=True,
    render=False
)
```

All training metrics, including loss values, episode rewards, and exploration statistics, were logged to TensorBoard in real time via the tensorboard_log parameter. This allowed for visual monitoring of training convergence and policy stability.

The final model, as well as the best validation performing checkpoint, were saved for subsequent evaluation on the held-out test dataset. Performance metrics including accuracy, precision, recall, F1-score, and confusion matrices are discussed in detail in the next chapter.

6.7 Results and Evaluation

To assess the performance of the trained Deep Q-Network (DQN) model for GNSS spoofing classification, a comprehensive evaluation was conducted using a held-out test dataset. This section presents the evaluation methodology, performance metrics used, and interpretation of the model's classification results across all spoofing scenarios.

6.7.1 Evaluation Strategy

After training for 80,000 timesteps, the model with the best validation reward was selected using EvalCallback. This model was then evaluated on the independent test set, which comprised 15% of the total dataset and was not used during training or validation. The evaluation was performed in deterministic mode to ensure consistent policy behavior and reproducible predictions.

The trained agent predicted a class label (ranging from 0 to 6) for each GNSS feature vector in the test set. These predictions were compared to the true labels to generate the final performance metrics.

6.7.2 Perfomance Metrics

The evaluation of the trained Deep Q-Network (DQN) model was conducted using a comprehensive set of performance metrics commonly employed in multi-class

classification tasks. These metrics include the normalized confusion matrix, overall classification accuracy, and per-class as well as macro averaged values of precision, recall, and F1-score. All computations were performed using the scikit-learn evaluation suite.

Confusion Matrix: To examine the class wise behavior of the model, a normalized confusion matrix was constructed. In this matrix, each row corresponds to the true class labels, while each column represents the predicted class labels. Normalization was applied row-wise such that each entry $\widehat{C_{i,j}}$ indicates the proportion of instances of true class i that were predicted as class j defined as:

$$\widehat{C_{i,j}} = \frac{C_{i,j}}{\sum_{j=1}^{K} C_{i,j}}, \quad \forall i,j \in \{0,1,\dots,K-1\} \tag{28}$$

where $K = 7$ denotes the number of classes, and $C_{i,j}$ is the raw count of predictions. Diagonal elements $\widehat{C_{i,j}}$ reflect the correct classification rate for each class, while off-diagonal elements quantify misclassification tendencies. This visualization is particularly informative in identifying classes that are frequently confused with others, especially in the presence of signal similarities between spoofing types.

Overall Accuracy: The overall accuracy of the model was calculated as the ratio of correct predictions to the total number of samples in the test set:

$$\text{Accuracy} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}\{\hat{y}_i = y_i\} \tag{29}$$

where $\hat{y}_i$ is the predicted label for the i-th test sample, y_i is the corresponding ground truth, N is the total number of test samples, and $1\{\cdot\}$ is the indicator function returning 1 if the condition is true and 0 otherwise. Accuracy provides an overall measure of predictive success but may obscure deficiencies in class-specific performance when class imbalance is present.

Precision, Recall, and F1-Score: To obtain a more nuanced understanding of model performance across individual spoofing scenarios, precision, recall, and F1-score were computed for each class as well as in macro-averaged form. All reported metrics were generated on the test set, which was strictly held out during training and validation phases, ensuring an unbiased assessment of the generalization performance of the proposed model.

6.8 Results and Discussion

The final performance of the Deep Q-Network (DQN) model was evaluated on an independent test set comprising 28,391 samples. This dataset, representing clean and spoofed GNSS signal observations, was strictly held out during training and validation to ensure an unbiased estimation of generalization capability. The evaluation results are presented in terms of per-class and macro-averaged classification metrics, along with graphical visualizations for interpretability.

6.8.1 Confusion Matrix Analysis

Figure 28 illustrates the normalized confusion matrix generated from the test set predictions. Each element of the matrix represents the proportion of instances belonging to true class i that were predicted as class j , normalized over the row. The diagonal entries correspond to correct classifications, while off-diagonal entries indicate misclassifications.

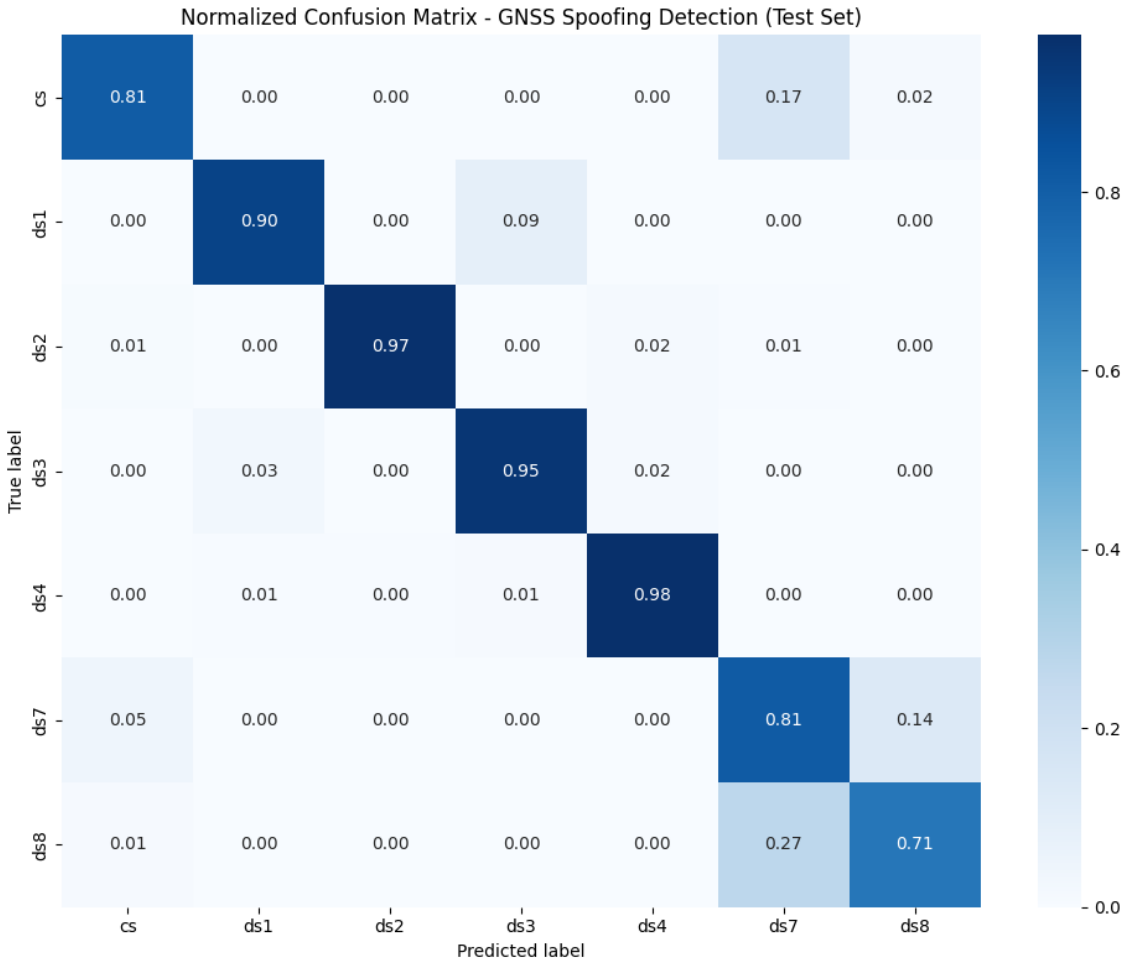

Figure 28. Normalized Confusion Matrix – GNSS Spoofing Detection (Test Set)

The clean signal class (cs) was correctly classified in 81% of the cases. However, 17% of clean instances were misclassified as class ds7, suggesting that the stealth spoofing strategy employed in ds7 closely mimics authentic signal behavior. All four classical spoofing scenarios ds1, ds2, ds3, and ds4 exhibited strong classification performance, with true positive rates of 90%, 97%, 95%, and 98%, respectively. These results indicate the model's ability to detect power-matched and time-push spoofing attacks with high precision.

The performance on ds7 (sparse stealth spoofing) and ds8 (replay/phase spoofing) revealed greater challenges. Class ds7 was correctly identified in 81% of the cases but also demonstrated non-negligible confusion with cs (5%) and ds8 (14%). Similarly, ds8

was predicted correctly 71% of the time, while 27% of its samples were misclassified as ds7. These confusions reflect the subtle temporal and phase alignment characteristics of these advanced spoofing strategies.

6.8.2 Class-wise Precision, Recall, and F1-Score

To quantify the performance of the DQN classifier across individual spoofing classes, per-class precision, recall, and F1-score were computed and are visualized in Figure 28.

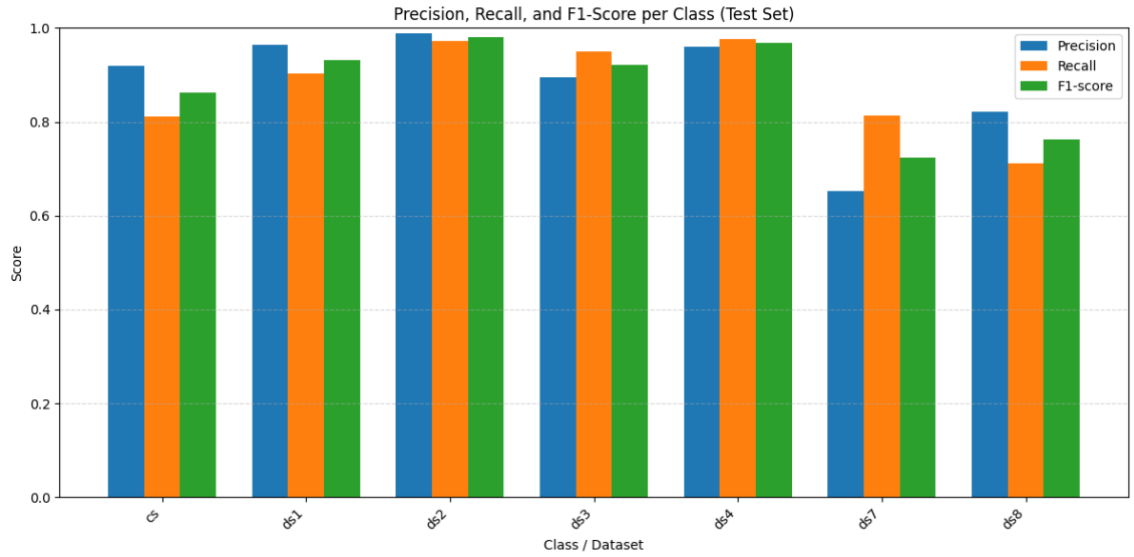

Figure 29. Precision, Recall, and F1-Score per Class (Test Set)

The model demonstrated consistently high predictive accuracy for the classical spoofing scenarios represented by datasets ds1 through ds4. For instance, ds1 achieved a precision of 0.963 and a recall of 0.903, while ds2 recorded the highest scores across all metrics, with precision and recall values of 0.988 and 0.971, respectively. Similarly, ds3 produced a precision of 0.895 and recall of 0.950, and ds4 exhibited balanced performance with precision at 0.960 and recall at 0.976. These results indicate that the model is highly effective in detecting GNSS spoofing attacks characterized by observable signal distortions or power-level discrepancies.

In contrast, the classifier's performance was comparatively weaker for more sophisticated attack types. The sparse stealth spoofing scenario (ds7) resulted in a significantly lower precision of 0.653 and a recall of 0.814, yielding an F1-score of 0.725. This reduction in performance reflects the difficulty in identifying stealthy signal manipulations that closely emulate authentic GNSS behavior. Similarly, the replay/phase-based spoofing represented by ds8 achieved a precision of 0.821 and recall of 0.712, corresponding to an F1-score of 0.763. These metrics suggest that while the model captures some of the distinguishing features of advanced spoofing techniques, certain subtle patterns remain challenging to detect.

The clean signal class (cs) yielded a precision of 0.920 and a recall of 0.810. Although the model was generally successful in recognizing authentic GNSS data, the reduced recall indicates a non-trivial rate of false negatives, likely due to misclassification of clean observations as stealthy spoofing signals particularly those in ds7. This outcome highlights the complexity of accurately distinguishing between genuine and minimally altered GNSS signals in real-time conditions.

7 Conclusions and Future Work

This thesis has examined the characteristics of GNSS signals and their susceptibility to spoofing, conducted a systematic literature review (SLR) of existing detection methods, and developed a Deep Q-Network (DQN) classifier for spoofing detection as well as in depth study on Reinforcement learning Algorithm. As described in Chapter 3, GNSS satellite signals are extremely weak at the receiver (ranging from –130 to –160 dBW) and are typically unencrypted. This makes them highly vulnerable to spoofing by adversaries using low-cost software-defined radios to transmit counterfeit signals. Spoofing can occur in the form of simple power-enhanced attacks or more complex strategies such as stealth and replay, all of which can distort the position or timing solution of the receiver. Recent disruptions in northern Europe, including a sharp increase in reported spoofing incidents in Finland between 2023 and 2024, reinforce the need for robust and intelligent spoofing detection techniques.

At the beginning at chapter 2 described introduction to reinforcement learning (RL). Followed by chapter 3, it presents a detailed technical review of Global Navigation Satellite System (GNSS) signal architecture, positioning mechanisms, and spoofing vulnerabilities.

The systematic literature review in Chapters 4 and 5 revealed that signal-monitoring and classification approaches continue to dominate the field. Traditional metrics such as carrier-to-noise ratio (C/N_o), pseudorange deviations, and Doppler shifts are frequently used in combination with machine learning methods. Classical ML algorithms like support vector machines, decision trees, and random forests are widely employed due to their simplicity and low computational cost. More recently, deep learning models especially convolutional neural networks (CNNs) operating on raw tracking data or spectrogram representations have gained traction, with hybrid models (e.g., CNN+LSTM, RF+MLP) showing promising performance improvements. However, reinforcement learning (RL) remains underrepresented in the literature. Our SLR identified only two

relevant studies using RL techniques, one employing DQN for spoof detection and another applying federated Q-learning for jamming detection in UAV networks. While many studies report high detection accuracy, few evaluate under adversarial or real-time constraints. Overall, the SLR contributed a structured mapping of GNSS spoofing detection approaches, highlighting existing strengths while identifying critical research gaps such as limited use of temporal dynamics, reliance on simulated data, and lack of integrated mitigation strategies.

In the simulation (Chapter 6), a multi-class DQN detection framework was developed and tested using the publicly available TEXBAT spoofing dataset. Raw GNSS tracking features were extracted and normalized across eight PRNs per sample, forming a 9-dimensional input vector for each epoch. A custom OpenAI Gym environment was designed, where each agent episode consists of a single, stateless observation, a snapshot of normalized features. The DQN agent is required to immediately classify the input into one of seven classes (clean or six spoofing scenarios). The model used the Stable-Baselines3 (SB3) MlpPolicy architecture with two hidden layers of 64 ReLU units, trained for 80,000 timesteps using ε-greedy exploration, experience replay, and checkpoint validation. Hyperparameters were selected to optimize stability and convergence, and random sampling ensured balanced exposure to different spoofing scenarios during training.

Evaluation results on a held-out test set of 28,391 samples showed strong performance for overt spoofing classes (ds1–ds4), with precision and recall values exceeding 0.89 in most cases. For example, the ds2 scenario achieved 0.988 precision and 0.971 recall. However, performance declined for more subtle attacks. The stealthy ds7 scenario attained 0.653 precision and 0.814 recall, while ds8 (phase-shift replay attack) achieved 0.821 precision and 0.712 recall. Additionally, the clean class had a recall of only 0.810 due to frequent misclassification as ds7, highlighting the similarity between legitimate and stealth spoofed signals. Confusion matrix analysis confirmed significant overlap

between stealthy classes and the clean class, consistent with prior findings that stealth spoofing is the most difficult to detect.

Several limitations of the current work were identified. The stateless design of the environment limits the agent's ability to exploit temporal dependencies in signal behavior, which could be critical for detecting gradually evolving spoofing strategies. The model also processes individual satellite epochs independently, whereas in practical settings, a receiver integrates observations across multiple satellites and over time. Furthermore, only the static scenarios of the TEXBAT dataset were utilized, the dynamic scenario ("live" spoofed receiver motion) was not incorporated, which could provide additional insights into real-time spoofing behavior. The use of a single dataset and absence of mitigation strategies further constrain the generalizability of the results.

Future work should address these limitations. Incorporating the dynamic scenario from TEXBAT would allow the model to generalize to time-varying spoofing conditions. Reformulating the RL environment into a multi-step or recurrent framework (e.g., using LSTM or GRU layers) would enable the agent to leverage temporal patterns and state transitions. Exploring policy-driven and actor critic driven RL frameworks such as PPO and A3C that not only detect but also recommend or execute mitigation strategies such as switching to alternate sensors, holding position, or down-weighting suspicious signals represents another promising direction. Lastly, adapting the model for deployment in embedded systems and evaluating performance under real-time constraints would move the solution closer to practical field applications.

In conclusion, this thesis demonstrated that reinforcement learning specifically a DQN classifier offers a promising direction for GNSS spoofing detection. The systematic review identified existing trends and direction in the literature, and the case study validated the approach through detailed feature extraction, model design, and robust evaluation. Moving forward, enhanced data diversity, temporal modeling, and integration with real-world mitigation strategies will be critical to advancing GNSS spoofing resilience.

Reference

Afroozeh, S., Bejach, V., Bokan, U., Bos, A., Ober, B., & Bartl, S. (2023). Antiference: New Concept for Evolutive Mitigation of RFI to GNSS †. *Engineering Proceedings*, *54*(1). Scopus. https://doi.org/10.3390/ENC2023-15451

Alanazi, A. (2024). SSRL-UAVs: A Self-Supervised Deep Representation Learning Approach for GPS Spoofing Attack Detection in Small Unmanned Aerial Vehicles. *Drones*, *8*(9). Scopus. https://doi.org/10.3390/drones8090515

Alhoraibi et al. (2024). Detection of GPS Spoofing Attacks in UAVs Based on Adversarial Machine Learning Model. *Journal*. https://doi.org/10.3390/s24186156

Ali, A. S., Lunardi, W. T., Singh, G., Bariah, L., Baddeley, M., Lopez, M. A., Giacalone, J.-P., & Muhaidat, S. (2024). RF Jamming Dataset: A Wireless Spectral Scan Approach for Malicious Interference Detection. *IEEE Communications Magazine*, *62*(11), 114–120. Scopus. https://doi.org/10.1109/MCOM.003.2300483

Arulkumaran, K., Deisenroth, M. P., Brundage, M., & Bharath, A. A. (2017). Deep Reinforcement Learning: A Brief Survey. *IEEE Signal Processing Magazine*, *34*(6), 26–38. https://doi.org/10.1109/MSP.2017.2743240

Aviation prepared for GPS interference: It is safe to fly in and to Finland. (2024, May 3). Traficom. https://www.traficom.fi/en/news/aviation-prepared-gps-interference-it-safe-fly-and-finland

Badar, A. U. R., Mahmood, D., Iqbal, A., Kim, S. W., Akleylek, S., Cengiz, K., & Nauman, A. (2025). DeepSpoofNet: A framework for securing UAVs against GPS spoofing

attacks. *PeerJ Computer Science*, *11*. Scopus. https://doi.org/10.7717/peerj-cs.2714

BeiDou Signal Plan—Navipedia. (n.d.). Retrieved June 11, 2025, from https://gssc.esa.int/navipedia/index.php/BeiDou_Signal_Plan

Braun, B., Montenbruck, O., Markgraf, M., Hörschgen-Eggers, M., & Kirchhartz, R. (2025). GNSS Jamming Observed on Sounding Rocket Flights from Northern Scandinavia. *European Navigation Conference 2024*, 55. https://doi.org/10.3390/engproc2025088055

CHRISTOPHER J.C.H. WATKINS & PETER DAYAN. (1992). *Q-Learning*. *Volume 8, pages 279–292,*. https://doi.org/10.1023/A:1022676722315

Dana, P. H. (1997). *Global Positioning System (GPS) Time Dissemination for Real-Time Applications*. *12*. https://doi.org/10.1023/A:1007906014916

Dang, Y., Benzaid, C., Yang, B., & Taleb, T. (2021). *Deep Learning for GPS Spoofing Detection in Cellular-Enabled UAV Systems*. 501–506. Scopus. https://doi.org/10.1109/NaNA53684.2021.00093

Dasgupta, S., Ghosh, T., & Rahman, M. (2022). A Reinforcement Learning Approach for Global Navigation Satellite System Spoofing Attack Detection in Autonomous Vehicles. *Transportation Research Record*, *2676*(12), 318–330. Scopus. https://doi.org/10.1177/03611981221095509

DDS Notes—PU BCA Study Materials. (n.d.). Retrieved June 2, 2025, from https://dds.com.np/docs/machine learning/reinforcement learning/deep-q-networks

Deisenroth, M. P., Fox, D., & Rasmussen, C. E. (2015). Gaussian Processes for Data-Efficient Learning in Robotics and Control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *37*(2), 408–423. https://doi.org/10.1109/TPAMI.2013.218

Filippou, S., Achilleos, A., Zukhraf, S. Z., Laoudias, C., Malialis, K., Michael, M. K., & Ellinas, G. (2023). *A Machine Learning Approach for Detecting GPS Location Spoofing Attacks in Autonomous Vehicles*. *2023-June*. Scopus. https://doi.org/10.1109/VTC2023-Spring57618.2023.10200857

Galileo Signal Plan—Navipedia. (n.d.). Retrieved June 11, 2025, from https://gssc.esa.int/navipedia/index.php/Galileo_Signal_Plan

Ghanbarzadeh et al. (2024). *GNSS/GPS Spoofing and Jamming Identification Using Machine Learning and Deep Learning*. Preprint. https://doi.org/10.48550/arXiv.2501.02352

GitHub—Seki5405/gnss_spoof_detector. (n.d.). Retrieved June 3, 2025, from https://github.com/seki5405/gnss_spoof_detector

GNSS Receivers General Introduction—Navipedia. (n.d.). Retrieved June 11, 2025, from https://gssc.esa.int/navipedia/index.php/GNSS_Receivers_General_Introduction?utm_source=chatgpt.com

GNSS spoofing. (2022, July 18). Wahyudin Syam. https://www.wasyresearch.com/gnss-spoofing-a-fatal-attack-on-gnss-system-that-is-difficult-to-detect/

GPS Signal Plan—Navipedia. (n.d.). Retrieved June 11, 2025, from https://gssc.esa.int/navipedia/index.php/GPS_Signal_Plan

GPS.gov: Space Segment. (n.d.). Retrieved May 26, 2025, from https://www.gps.gov/systems/gps/space/

H. Farkhari, J. Viana, S. Kahvazadeh, P. Sebastião, V. P. G. Jimenez, & R. Dinis. (2024). A Hybrid Approach to Reliable Jamming Identification in UAV Communications Using Combined DNNs and ML Algorithms. *IEEE Access*, *12*, 178898–178908. https://doi.org/10.1109/ACCESS.2024.3504729

Humphreys, T. E., Ledvina, B. M., Tech, V., Psiaki, M. L., O'Hanlon, B. W., & Kintner, P. M. (2008). *Assessing the Spoofing Threat: Development of a Portable GPS Civilian Spoofer*.

I. E. Mehr & F. Dovis. (2025). A Deep Neural Network Approach for Classification of GNSS Interference and Jamming. *Journal*. https://doi.org/10.1109/TAES.2024.3462662

Janner, M., Fu, J., Zhang, M., & Levine, S. (2021). *When to Trust Your Model: Model-Based Policy Optimization* (No. arXiv:1906.08253). arXiv. https://doi.org/10.48550/arXiv.1906.08253

Jin, S., Wang, Q., & Dardanelli, G. (2022). A Review on Multi-GNSS for Earth Observation and Emerging Applications. *Remote Sensing*, *14*(16), 3930. https://doi.org/10.3390/rs14163930

Jung, H., Psiaki, M. L., & Powell, S. P. (n.d.). *Kalman-Filter-Based Semi-Codeless Tracking of Weak Dual-Frequency GPS Signals*.

Kahil, H., Sharma, S., Välisuo, P., & Elmusrati, M. (2025). Reinforcement learning for data center energy efficiency optimization: A systematic literature review and research roadmap. *Applied Energy*, *389*, 125734. https://doi.org/10.1016/j.apenergy.2025.125734

Kauranen, A., & Kauranen, A. (2024, October 31). Finland detects satellite navigation jamming and spoofing in Baltic Sea. *Reuters*. https://www.reuters.com/world/europe/finland-detects-satellite-navigation-jamming-spoofing-baltic-sea-2024-10-31/

Korium, M. S., Saber, M., Ahmed, A. M., Narayanan, A., & Nardelli, P. H. J. (2024). Image-based intrusion detection system for GPS spoofing cyberattacks in unmanned aerial vehicles. *Ad Hoc Networks*, *163*. Scopus. https://doi.org/10.1016/j.adhoc.2024.103597

Li et al. (2025). Performance Testing and Analysis of a New GNSS Spoofing Detection Method in Different Spoofing Scenarios. *Journal*. https://doi.org/10.1109/ACCESS.2025.3553475

Long, J., & Han, J. (2023). Reinforcement Learning with Function Approximation: From Linear to Nonlinear. *Journal of Machine Learning*, *2*(3), 161–193. https://doi.org/10.4208/jml.230105

Mao, W.-L. (2016). GPS interference mitigation using derivative-free Kalman Filter-based RNN. *Radioengineering*, *25*(3), 518–526. Scopus. https://doi.org/10.13164/re.2016.0518

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T. P., Harley, T., Silver, D., & Kavukcuoglu, K. (2016a). *Asynchronous Methods for Deep Reinforcement Learning* (No. arXiv:1602.01783). arXiv. https://doi.org/10.48550/arXiv.1602.01783

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T. P., Harley, T., Silver, D., & Kavukcuoglu, K. (2016b). *Asynchronous Methods for Deep Reinforcement*

Learning (No. arXiv:1602.01783). arXiv. https://doi.org/10.48550/arXiv.1602.01783

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. (2013). *Playing Atari with Deep Reinforcement Learning* (No. arXiv:1312.5602). arXiv. https://doi.org/10.48550/arXiv.1312.5602

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533. https://doi.org/10.1038/nature14236

Mnih, V., Martin Riedmiller, Wierstra, D., Ioannis Antonoglou, Graves, A., Silver, D., & Kavukcuoglu, K. (n.d.). *Playing Atari with Deep Reinforcement Learning*. Ar5iv. Retrieved June 2, 2025, from https://ar5iv.labs.arxiv.org/html/1312.5602

Morales, E. F., Zaragoza, J. H., Morales, E. F., & Zaragoza, J. H. (1 C.E., January 1). *An Introduction to Reinforcement Learning* (introduction-reinforcement-learning) [Chapter]. Https://Services.Igi-Global.Com/Resolvedoi/Resolve.Aspx?Doi=10.4018/978-1-60960-165-2.Ch004; IGI Global Scientific Publishing. https://doi.org/10.4018/978-1-60960-165-2.ch004

Morales Ferre, R., Richter, P., De La Fuente, A., & Simona Lohan, E. (2019). In-lab validation of jammer detection and direction finding algorithms for GNSS. *2019 International Conference on Localization and GNSS (ICL-GNSS)*, 1–6. https://doi.org/10.1109/ICL-GNSS.2019.8752944

N. I. Mowla, N. H. T., I. Doh, K. Chae. (2020). AFRL: Adaptive federated reinforcement learning for intelligent jamming defense in FANET. *Journal of Communications and Networks*. https://doi.org/10.1109/JCN.2020.000015

Naeem, M., Rizvi, S. T. H., & Coronato, A. (2020). A Gentle Introduction to Reinforcement Learning and its Application in Different Fields. *IEEE Access*, *8*, 209320–209344. https://doi.org/10.1109/ACCESS.2020.3038605

Nayfeh et al. (2023). Machine Learning Modeling of GPS Features with Applications to UAV Location Spoofing Detection and Classification. *Journal*. https://doi.org/10.1016/j.cose.2022.103085

Ott et al. (2025). Metric-Based Few-Shot Learning With Triplet Selection for Adaptive GNSS Interference Classification. *Journal*. https://doi.org/10.1109/JISPIN.2025.3562140

Papadimitratos, P., & Jovanovic, A. (2008). GNSS-based Positioning: Attacks and countermeasures. *MILCOM 2008 - 2008 IEEE Military Communications Conference*, 1–7. https://doi.org/10.1109/MILCOM.2008.4753512

Psiaki, M. L., & Humphreys, T. E. (2016). GNSS Spoofing and Detection. *Proceedings of the IEEE*, *104*(6), 1258–1270. https://doi.org/10.1109/JPROC.2016.2526658

Radoš, K., Brkić, M., & Begušić, D. (2024). Recent Advances on Jamming and Spoofing Detection in GNSS. *Sensors*, *24*(13), 4210. https://doi.org/10.3390/s24134210

Rijnsdorp, J., van Zwol, A. ,. Snijders, M. (2023). Satellite Navigation Signal Interference Detection and Machine Learning-Based Classification Techniques towards Product Implementation †. *Engineering Proceedings*. https://doi.org/10.3390/ENC2023-15449

Sabatini, R., Moore, T., & Ramasamy, S. (2017). Global navigation satellite systems performance analysis and augmentation strategies in aviation. *Progress in Aerospace Sciences*, *95*, 45–98. https://doi.org/10.1016/j.paerosci.2017.10.002

Schmidt, D., Radke, K., Camtepe, S., Foo, E., & Ren, M. (2016a). A Survey and Analysis of the GNSS Spoofing Threat and Countermeasures. *ACM Computing Surveys*, *48*(4), 1–31. https://doi.org/10.1145/2897166

Schmidt, D., Radke, K., Camtepe, S., Foo, E., & Ren, M. (2016b). A Survey and Analysis of the GNSS Spoofing Threat and Countermeasures. *ACM Computing Surveys*, *48*(4), 1–31. https://doi.org/10.1145/2897166

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). *Proximal Policy Optimization Algorithms* (No. arXiv:1707.06347). arXiv. https://doi.org/10.48550/arXiv.1707.06347

Shepard, D. (n.d.). *Characterization of Receiver Response to Spoofing Attacks*.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., & Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, *529*(7587), 484–489. https://doi.org/10.1038/nature16961

Spread Spectrum and Code Modulation of L1 GPS Carrier | GEOG 862: GPS and GNSS for Geospatial Professionals. (n.d.). Retrieved June 11, 2025, from https://www.e-education.psu.edu/geog862/node/1753

Sutton, R. S., & Barto, A. (2020). *Reinforcement learning: An introduction* (Second edition). The MIT Press.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction* (Second edition). The MIT Press.

Świechowski, M., Godlewski, K., Sawicki, B., & Mańdziuk, J. (2023). Monte Carlo Tree Search: A Review of Recent Modifications and Applications. *Artificial Intelligence Review*, *56*(3), 2497–2562. https://doi.org/10.1007/s10462-022-10228-y

TEXBAT – Radionavigation Laboratory. (n.d.). Retrieved June 13, 2025, from https://radionavlab.ae.utexas.edu/texbat/

USU: GPS interference affecting almost all Finnish airports. (2025a, January 14). News. https://yle.fi/a/74-20136751

USU: GPS interference affecting almost all Finnish airports. (2025b, January 14). News. https://yle.fi/a/74-20136751

USU: GPS interference affecting almost all Finnish airports. (2025c, January 14). News. https://yle.fi/a/74-20136751

VectorNav. (n.d.). Retrieved June 11, 2025, from https://www.vectornav.com/resources/inertial-navigation-primer/alternative-navigation/altnav-gnsschallenged

Viana, J., Farkhari, H., Campos, L. M., Sebastiao, P., Cercas, F., Bernardo, L., & Dinis, R. (2022). *Two methods for Jamming Identification in UAV Networks using New Synthetic Dataset*. *2022-June*. Scopus. https://doi.org/10.1109/VTC2022-Spring54318.2022.9860816

W. El-Shafai, A. T. Azar, & S. Ahmed. (2025). AI-Driven Ensemble Classifier for Jamming Attack Detection in VANETs to Enhance Security in Smart Cities. *IEEE Access*, *13*, 50687–50713. https://doi.org/10.1109/ACCESS.2025.3552544

Wang, W., Aguilar Sanchez, I., Caparra, G., McKeown, A., Whitworth, T., & Lohan, E. S. (2021). A Survey of Spoofer Detection Techniques via Radio Frequency Fingerprinting with Focus on the GNSS Pre-Correlation Sampled Data. *Sensors*, *21*(9), 3012. https://doi.org/10.3390/s21093012

Weng, D., Gan, X., Chen, W., Ji, S., & Lu, Y. (2020). A New DGNSS Positioning Infrastructure for Android Smartphones. *Sensors*, *20*(2), 487. https://doi.org/10.3390/s20020487

Wesson, Kyle D., Shepard, Daniel P., & Humphreys, Todd E. (2012, January). *Straight Talk on Anti-Spoofing: Securing the Future of PNT*. GPS World. https://radionavlab.ae.utexas.edu/images/stories/files/papers/antiSpoofStraightTalk_Wesson.pdf

Yoon, H., Seok, H., Lim, C., & Park, B. (2020). An Online SBAS Service to Improve Drone Navigation Performance in High-Elevation Masked Areas. *Sensors*, *20*(11), 3047. https://doi.org/10.3390/s20113047

Appendices

Appendix 1 Summary Table of 52 Reviewed Papers on GNSS Spoofing and Jamming Detection and Mitigation

ID	DOI	Title	Author	Techniques	Spoofing&Jamming Type	Dataset/Testbed	Validation Metrics	Validation Method	Accuracy	Study Application	Mitigation/detection	Ref	Paper Type	Year
P1	10.1109/TMLCN.2023.3316150	3D Radio Map-Based GPS Spoofing Detection and Mitigation for Cellular-Connected UAVs	Dang, Karakoc, Norshahida, & Jamin	MLP, CNN, and RNN	spoofing	Simulated Data	Detection accuracy, training time, spoofing margin sensitivity, mitigation error	Simulated testbed using 3D city map and wireless signal modeling	No explicit accuracy mentioned	GPS spoofing detection and mitigation in urban environments for cellular-connected UAVs	Both (detection with ML, mitigation with particle filter)	(Dang, Karakoc, Norshahida, & Jamin, 2023)	Journal	2023
P2	10.1109/TMLCN.2024.3386649	A Deep Learning Based Induced GNSS Spoof Detection Framework	A. Iqbal; M. N. Aman; B. Sikdar	Deep Learning (VAE + GAN)	spoofing	Real TEXBAT Data	Detection rate, false positives, robustness	Experimental evaluation on all 7 TEXBAT scenarios using deep neural network classifiers and representation learning detectors	Up to 99% detection accuracy for simple and intermediate spoofing. ~93% detection accuracy for sophisticated spoofing. Compared with classical ML models, which dropped to 44.1% for DS-7.	GNSS spoofing detection in static and dynamic receiver scenarios using RF + tracking module features	Detection	(Iqbal, Aman, & Sikdar, 2024)	Journal	2024
P3	10.1109/TAES.2024.3462662	A Deep Neural Network Approach for Classification of GNSS Interference and Jamming	I. E. Mehr; F. Dovis	CNN	Jamming	Simulated	Accuracy, F1-score, inference time, confusion matrix	Monte Carlo simulation with various signal/interference levels and combinations	ResNet with STFT achieved 99.69% accuracy and inference time 3.99 ms (GPU)	terrestrial & LEO-based scenarios	Detection	(Mehr & Dovis, 2025)	Journal	2025
P4	10.1109/ACCESS.2024.3504726	A Hybrid Approach to Reliable Jamming Identification in UAV Communications Using	H. Farkhari et al.	DNN + XGBoost	jamming	Simulated	accuracy	Comparison with baseline DNNs and DNN+XGB			Detection	(Farkhari et al., 2024)	Journal	2024
P5	10.1109/VTC2023-Spring57618.2023.10200857	A Machine Learning Approach for Detecting GPS Location Spoofing Attacks in Autonomous Vehicles	S. Filippou et al.	ML-based anomaly detection (LOF, iForest, OC-SVM, Autoencoder)	spoofing	CARLA simulator	Precision: 97.61%, Recall: 99.31%, Specificity: 97.56%	Simulation-based evaluation using train/val/test splits; compared against threshold-based method (TAD)	98.45% (F1-score)	autonomous vehicle navigation		(Filippou et al., 2023)	Conference	2023
P6	10.1109/ACCESS.2019.2939141	A New Threat for Pseudorange-Based RAIM: Adversarial Attacks on GNSS Positioning	Y. Sun, L. Fu	Fast Gradient Sign Method (FGSM)	spoofing	Real	Simulation + real data, testing spoofed positioning vs. RTK baseline; test statistic (Ts) under RAIM threshold	Simulation + real data, testing spoofed positioning vs. RTK baseline; test statistic (Ts) under RAIM threshold	Not specified		Detection	(Sun & Fu, 2019)	Journal	2019
P7	10.1177/03611981221096509	A Reinforcement Learning Approach for Global Navigation Satellite System Spoofing Attack Detection in	S. Dasgupta et al.	Deep Reinforcement Learning (DQN), using a DNN for Q-value approximation	spoofing	RealWorld/Honda Driving dataset	accuracy, precision, recall	Nine simulated attack scenarios; confusion matrix and precision-recall evaluation	90.99%–100%	autonomous vehicle navigation	Detection	Dasgupta et al., 2022	Journal	2022
P8	10.1109/TVT.2022.3174406	A Slowly Varying Spoofing Algorithm Avoiding Tightly-Coupled GNSS/IMU With	Y. Gao, G. Li	NO AI, mathematical modeling and spoofing control algorithm	spoofing	Simulated	Not Specified	imulated GNSS/IMU spoofing experiment with RAIM and parameter rationality checks		GNSS/IMU systems	Detection	(Gao & Li, 2022)	Journal	2022
P9		Adaptive Resilience Navigation Filter for Detecting and Mitigating Multi-Spoofing Attacks in Range-Based Localization Systems using	A. Venturini et al.	Not AI-based, uses statistical hypothesis testing with an Adaptive Resilience Navigation Filter (ARNF)	spoofing	Simulated Data	PD, PFA, ROC curves	Monte Carlo simulation vs EKF baselines	Comparable to Clairvoyant EKF; low estimation error under multi-spoofer attacks	Multi-spoofing detection & mitigation in vehicles	Detection and Mitigation	(Venturini et al., 2025)	Journal	2025
P10	10.1109/ICN.2020.000015	AFRL: Adaptive federated reinforcement learning for intelligent jamming defense in FANET	N. I. Mowla et al.	Federated Learning + Q-learning (adaptive RL)	jamming	Simulated	Accuracy, Cumulative Reward, Convergence Iterations	Simulation (ns-3), 10 UAV scenarios, convergence vs. Dijkstra/random baseline	80.88% detection accuracy; 82.01% avg. gain over baseline	UAVs in FANET	Detection and Mitigation	(Mowla et al., 2020)	Journal	2020
P11	10.1109/ACCESS.2025.3552544	AI-Driven Ensemble Classifier for Jamming Attack Detection in VANETs to Enhance Security	W. El-Shafai et al.	Ensemble (RF + ET + CNN)	jamming	Real	Accuracy, Precision, Recall, F1-score, AUC	10-fold cross-validation, confusion matrix, ROC curve	99.81	V2V/V2I networks for smart cities	Detection	(El-Shafai et al., 2025)	Journal	2025
P12	10.3390/ENC2023-15451	Antiference: New Concept for Evolutive Mitigation of RFI to GNSS	S. Alrasouli et al.	ML – ResNet50 + fingerprinting	spoofing, jamming	Real+Simulated	accuracy, False Alarm Rate (FAR), C/N_0 improvement	Testbed with COTS receivers; compared against Roc02 performance	100		Detection and Mitigation	(Alrasouli et al., 2023)	Journal	2023
P13	10.1109/NAS51552.2021.00083	Deep Learning for GPS Spoofing Detection in Cellular-Enabled UAV Systems	Y. Dang et al.	ML – MLP (Multi-Layer Perceptron)	spoofing	Simulated	Accuracy, Mean Squared Error (MSE)	ython/TensorFlow simulation across 3 network scenarios; hyperparameter tuning	93% (3 BSs), 80% (1 BS)	UAV navigation	Detection	(Dang et al., 2021)	Conference	2021
P14	10.1109/JIOT.2022.3195320	Deep-Ensemble-Learning-Based GPS Spoofing Detection for Cellular-Connected UAVs	Y. Dang et al.	ML – MLP ensemble (deep learning)	spoofing	Simulated	Accuracy, Binary Cross-Entropy Loss	Python/TensorFlow simulation, multi-MLP + ML meta-model ensemble	>97% (2 BSs), >83% (1 BS)	UAVs	Detection	(Dang et al., 2022)	Journal	2022
P15	10.7717/peerj-cs.2714	DeepSpoofNet: a framework for securing UAVs against GPS spoofing attacks	Badar et al.	ML – CNN-BiLSTM hybrid with ANOVA feature selection and SMOTE balancing	spoofing	Simulated	Accuracy, Precision, Recall, F1 Score	Comparative analysis of feature selection methods (PCA, PCC, chi-squared test, ANOVA) with deep learning models (ANN, CNN, CNN-BiLSTM); evaluated using performance metrics	99.35%	UAVs	Detection	(Badar et al., 2025)	Journal	2025
P16	10.1109/ACCESS.2019.2934632	Delimitated Anti Jammer Scheme for Internet of Vehicle: Machine Learning Based Security Approach	S. Kumar et al.	ML – CatBoost (gradient boosting on decision trees)	jamming	Simulated IoV/VANET environment (MATLAB/Simulink)	Accuracy, Precision, Recall, F1-score, PLR, PDR, SNR, MSE	Simulations comparing with XGBoost, AdaBoost, ANN, SOM	99.91	Internet of Vehicles	Detection	(Kumar et al., 2019)	Journal	2019
P17	10.1186/s13634-023-01103-1	Detecting GNSS spoofing using deep learning	P. Borhani-Darian et al.	ML – CNN (image-based spoofing classifier with GMM clustering)	spoofing	Simulated	Accuracy	Parallel CNN classifiers on CAF sub-images + GMM clustering for multi-spoofer estimation	CNN: 96.6%, MLP: 91.3%		Detection			
P18	10.14569/IJACSA.2024.0151146	Detecting GPS Spoofing Attacks Using Corrected Low-Cost INS Data with an LSTM Network	M. Aftatah, K. Zebbara	ML – LSTM + Modified M-of-N statistical method	Spoofing	Simulated	RMSE (acceleration/gyro), detection rate (8/10 spoof zones)	Simulated dynamic scenario with 10 spoofed intervals and statistical thresholding	80		Detection	(Aftatah & Zebbara, 2024)	Journal	2024
P19	10.1109/ACCESS.2021.3089847	Detecting Signal Spoofing Attack in UAVs Using Machine Learning Models	A. Shafique et al.	ML – SVM (various kernels) + K fold + hard/soft voting	spoofing	Simulated	accuracy, precision, recall, f1-score	Multiple SVM variants with K=5, 10, 15, 20 voting ensemble	98.7% accuracy, 1.0 precision, 0.97 recall, 0.98 F1-score	UAV systems	Detection	(Shafique et al., 2021)	Journal	2021
P20	10.1109/ACCESS.2023.3262996	Detection and Orientation of GNSS Spoofing Based on Positioning Solutions of Three Receivers	S. Chen et al.	Kalman Filter, Adaptive Delta Gradient Descent (ADGD), CKF	spoofing	Simulated Data	Detection rate, false alarm rate, RMSE of orientation angles	Simulation and field test scenarios with navigation signal repeater	100% detection rate; 0.1s response time; avg. orientation error: 0.55° (static), 1.33° (dynamic)	GNSS recivers	Detection	(Chen et al., 2023)	Journal	2023
P21	10.1109/JIOT.2024.3519633	DroneGuard: An Explainable and Efficient Machine Learning Framework for Intrusion Detection in Drone Networks	V. U. Ihekoronye et al.	ML – Decision Tree, RF, AdaBoost, GNB, LR + SHAP (XAI)	spoofing	Real GPS spoofing dataset (AV-GPS), simulated DoS dataset (WSN-DS)	Accuracy, Precision, Recall, F1-score	Comparative ML evaluation + XAI interpretability (SHAP)	99.52% (DT model)	Detecting GPS spoofing & DoS attacks in UAV networks	Detection	(Ihekoronye et al., 2025)	Journal	2025
P22	10.3390/info16020115	Enhancing UAV Security Against GPS Spoofing Attacks Through a Genetic Algorithm-Driven Deep Learning Framework	A. Al-Sabbagh et al.	DL – LSTM optimized with Genetic Algorithm (GA)	spoofing	Real-world Data	Accuracy, Precision, Recall, F1-score, Loss	GA tuning vs baseline LSTM, LSTM-GA, PCA-CNN-LSTM, GA-XGBoost	93.12% accuracy, F1-score 93.39%, Precision 93.93%, Recall 93.12%	Detection of GPS spoofing in UAVs across basic, intermediate, and sophisticated attacks	Detection	(Al-Sabbagh et al., 2025)	Journal	2025
P23	10.1109/ACCESS.2022.3169470	GANSAT: A GAN and SAtellite Constellation Fingerprint-Based Framework for GPS	D. Roy et al.	GANs, ML, Neural Network,	spoofing	Real	Accuracy (spoofing + localization)	Real-world spoofing via SDR + GAN/CNN model comparisons	99.5, 100	Detecting GPS spoofing & estimating position in GPS-degraded	Detection	(Roy et al., 2022)	Journal	2022
P24	10.1109/ITSC55140.2022.9922109	GNSS Anti-spoofing Detection based on Gaussian Mixture Model Machine Learning	Z. Feng et al.	ML – GMM (unsupervised clustering)	spoofing	Real GPS data from NASA RINEX + simulated spoofing (1–3 satellites, 30–200m error)	Accuracy, RMSE, Mean Deviation, Variance, Outlier Count	GMM clustering convergence rule vs. K-Means; PCA projection for visualization	91.6% (1 spoofed), 86.1% (2), 77.1% (3)	pseudo-range-based position solutions	Detection	(Feng et al., 2022)	Conference	2022
P25	10.1109/ACCESS.2022.3160047	GNSS Spoofing Detection and Mitigation in Multireceiver Configuration via Tracklets and Spoofer Localization	B. Pardhasaradhi et al.	Non-AI – Statistical detection (GLRT), Extended Kalman Filter (EKF), Iterative LS	spoofing	Simulated Data	Position RMSE (PRMSE), GLRT decision metric, detection/misclassification time	Simulation with 4 receivers, 4 satellites, repeatable spoofing scenario	PRMSE reduced from 15–20m (no mitigation) to 2–4m (with method)	GNSS receivers using spatial info	Detection + Mitigation	(Pardhasaradhi et al., 2022)	Journal	2022
P26	10.13164/re.2016.0518	GPS interference mitigation using derivative-free Kalman Filter-based RNN	(Mao, 2016)	DL – Recurrent Neural Network (RNN) + Unscented Kalman Filter (UKF)	Jamming	Simulated Data	SNR improvement, MSPE (mean squared prediction error)	MATLAB-based simulations over 100 runs, 5 algorithms compared	Avg. SNR improvement +8.26 dB (vs LMS), +7.2 dB (vs RLS), +5.29 dB (vs DNA), +1.14 dB (vs	for various narrowband jamming environments	Mitigation	(Mao, 2016)	Journal	2016
P27	10.33012/navi.574	GPS Spoofing Mitigation and Timing Risk Analysis in Networked Phasor Measurement Units	S. Bhamidipati & G. Gao	Stochastic Reachability + Distributed Kalman Filter (SR-DKF)	spoofing, signal-level	Not Specified	Timing risk (probabilistic), total vector error (TVE), stochastic set estimation	Simulated attacks (meaconing, signal-level, coordinated spoofing); multi-node testbed	Not specified	Mitigating GPS spoofing in electrical grid PMU timing systems	Mitigation	(Bhamidipati & Gao, 2023)	Journal	2023
P28	10.3390/drones6010008	GPS-Spoofing Attack Detection Technology for UAVs Based on Kullback-Leibler Divergence	E. Basan et al.	Mathematical – Kullback-Leibler Divergence (unsupervised anomaly detection)	spoofing	Real UAV flight logs with simulated spoofing using HackRF	Accuracy, Type I/II errors, likelihood, standard deviation	Real UAV experiments and simulations under spoofed and normal flight scenarios	Detection accuracy: 98%, Type II error: 9%, Type I: 0%, delay: 5–30s	Real-time GPS spoofing detection in UAVs via cyber-physical parameter divergence	Detection	Basan et al., 2022)	Journal	2022
P29	10.1016/j.adhoc.2024.103087	Image-based intrusion detection system for GPS spoofing cyberattacks in unmanned aerial vehicles	Korium et al., 2024	Deep Learning - Convolutional Neural Networks (CNNs) with Principal Component	spoofing	Simulated	Accuracy, detection time, latency	Extensive numerical experiments using benchmark datasets; comparison with existing models	99.25% on unseen data, 100% on training data	Real-time detection of GPS spoofing cyberattacks in Unmanned Aerial	Detection	(Korium et al., 2024)	Journal	2024
P30	10.3389/fcomp.2025.1546282	Integrating artificial intelligence for improved security of IoT-drones through cyber-physical attack	Hakeem et al.	DL – MobileNet (Transfer Learning) + Chi-Square Feature Selection	spoofing	Real + Simulated GPS data	Accuracy, Precision, Recall, F1 Score, 5-fold CV	Experiments using MobileNet vs. other DL/transfer learning models; real and spoofed GPS signal records	Accuracy, Precision, Recall, F1 Score, 5-fold CV	GPS spoofing detection in lightweight drones under cyber-physical constraints	Detection	(Hakeem et al., 2025)	Journal	2025
P31	10.7717/peerj-cs.2419	Intelligent algorithmic framework for detection and mitigation of BeiDou spoofing	Tariq, U.	Autoencoder + LSTM + ARIMA + ABE (cryptography)	spoofing	Real + simulated BeiDou spoofing signals	Accuracy, precision, recall, F1-score	10-fold experiments using real-world signal + simulation	98.4% accuracy, 95.2% F1	Spoofing detection & mitigation in vehicular GNSS (BeiDou-based	Both	(Tariq, 2024)	Journal	2024
P32	10.1109/ACCESS.2022.3150620	Jamming Detection and Classification in OFDM-Based UAVs via Feature- and Spectrogram-Tailored Machine Learning	Y. Li et al.	ML (RF, KNN, DT) and DL (CNN: AlexNet, ResNet-50, EfficientNet)	Jamming	Real SDR-based jamming data	Accuracy, Precision, Recall, F1-Score, False Alarm Rate (FAR)	SDR hardware testing on drone using multiple jammer locations + 10-fold CV	CNN: 99.79%, ML-RF: 92.2%, False Alarm: CNN 0.03%, RF 1.35%	Detecting and classifying RF jamming attacks on UAVs	Detection	(Li et al., 2022)	Journal	2022
P33	10.7717/peerj-cs.2399	Leveraging machine learning for the detection of structured interference in Global Navigation Satellite Systems	Nabi et al.	Logistic Regression, SVM, KNN, Decision Tree	spoofing, replay	Real GNSS spoofing dataset from TEXBAT (ds8 scenario, SCER attack)	Accuracy, F1-score, Precision, Recall, ROC curve	6-fold cross-validation using GNSS-SDR outputs	91.08% accuracy, 94% F1 score (LogReg, SVM)	Detecting structured GNSS spoofing attacks (SCER) via receiver-side tracking features	Detection	(Nabi et al., 2024)	Journal	2024
P34	10.1109/ISPIN.2025.3562140	Metric-Based Few-Shot Learning With Triplet Selection for Adaptive GNSS Interference	Ott et al.	ML – CNN (ResNet18) + Few-Shot Learning (Prototypical Networks)	jamming	Real-world Data	accuracy, Precision, Recall, F2-score	Benchmark across 3 datasets, 10-fold runs, real-world class permutation tests	97.66% accuracy	classification of unseen GNSS jammer types under real and	detection	(Ott et al., 2025)	Journal	2025
P35	10.1109/ACCESS.2022.3187993	Mitigation of GNSS Time Synchronization Attacks in a Multicorrelator Receiver	Shang, X. et al.	Not AI/ML - signal processing with Kalman Filter	Spoofing (TSA	TEXBAT (real spoofing) + Monte Carlo simulations	RMS timing error, Monte Carlo simulations, real dataset testing	Simulated + real spoofing (TEXBAT), REKF used for robustness	Timing error reduced from 600 m to 16 m (53.4 ns RMS)	GNSS timing security in critical systems (e.g., power grid, telecom)	Mitigation	(Shang et al., 2022)	Journal	2022
P36	10.1109/ACCESS.2021.3064383	Navigation in GPS Spoofed Environment Using M-Best Positioning Algorithm and Data Association	Pardhasaradhi et al.	No ML - Uses M-Best Positioning + Kalman Filter + Data Association (NN & PDA)	spoofing	Simulated GPS spoofing scenarios	Position RMSE, Track Swap (TS) rate	Simulated spoofed pseudoranges, Monte Carlo tests	uccessfully tracks true trajectory under spoofing; RMSE < 400 m in complex spoofing	Mitigating spoofing effects on autonomous vehicle navigation	Mitigation	(Pardhasaradhi et al., 2021)	Journal	2021
P37	10.33012/navi.629	Optimal INS Monitor for GNSS Spoofer Tracking Error Detection	Kujur, B.; Khanafseh, S.; Pervan, B.	None – uses Kalman Filter-based monitoring	spoofing	Simulated	Missed detection rate, simulation comparisons, experimental validation	KF-based statistical monitors (CI, CPI), simulation + real data	detects spoofing with <1% missed detection for 10 cm tracking error in <1 min	aircraft navigation	detection	(Kujur, Khanafseh, & Pervan, 2024)	Journal	2024
P38	10.1109/TIFS.2022.3219287	PAST-AI: Physical-Layer Authentication of Satellite Transmitters via Deep Learning	G. Oligeri; S. Sciancalepore; S. Raponi; R. D. Pietro	ML – CNN (ResNet-18), Autoencoders	spoofing	Real I-Q samples (100M+) from IRIDIUM LEO satellites (589 hrs)	Hit/miss rate, AUC, mean squared error, ROC curve	Real dataset (IRIDIUM), training/validation/testing on I-Q image slices	0.8–1.0 accuracy (AUC ~1.0), depending on scenario	Satellite transmitter authentication	detection	(Oligeri et al., 2023)	journal	2023
P39	10.1109/ACCESS.2025.3553475	Performance Testing and Analysis of a New GNSS Spoofing Detection Method in Different Spoofing Scenarios	Li et al.	ML – Optimized Random Forest (RF)	spoofing	TEXBAT dataset +	Accuracy, precision, recall, F1-score, AUC, false alarm rate	ML comparison across RF, SVM, DT, KNN with confusion matrix and ROC	Up to 99.96% accuracy; >97% across spoofing scenarios	Detection of GNSS spoofing in static and mixed spoofing scenarios	detection	(Li et al., 2025)	Journal	2025
P40	10.1109/MCOM.003.2300483	RF Jamming Dataset: A Wireless Spectral Scan Approach for Malicious Interference Detection	Ali, A.S.; Lunardi, W.T.; Singh, G.; Bariah, L.; Baddeley, M.;	ML – MLP, SVM, Random Forest, XGBoost, LightGBM	jamming	Real-world Data	Accuracy, F1-score, Precision, Recall, inference speed, 10-fold CV	Real jamming testbed, various distances/powers, multi-class classifier tuning	100% accuracy (RAF, SVM), 99% accuracy, 96.33% F1-score (RAF)	Detection and classification of RF jamming and interference using spectral features	detection	(Ali et al., 2024)	Journal	2024
P41	10.3390/ENC2023-15449	Satellite Navigation Signal Interference Detection and Machine Learning-Based Classification Techniques towards Product Implementation †	Rijnsdorp, J.; van Zwol, A.; Snijders, M.	ML – CNN (EfficientNet-B5 wi	jamming	Simulated GNSS + jamming signals from Orolia GSG-8 simulator (GPS L1, Galileo E1, SBAS L1)	Confusion matrix, detection latency, qualitative spectrum classification	Over-the-cable SDR testing with clean and various jammer signals		Handheld GNSS jammer detection/classification module for critical users	Detection + Classification	(Rijnsdorp et al., 2023)	Conference proceeding	2023
P42	10.1109/ACCESS.2020.2992119	SCER Spoofing Attacks on the Galileo Open Service and Machine Learning Techniques for End-User Protection	F. Gallardo, A. P. Yuste	ML (Random Forest, AdaBoost, SVM)	spoofing, replay	Simulated	K-fold, F1-score, confusion matrix	Controlled simulation environment, multiple C/N_0 levels, ML classifier benchmarking	98.48	Galileo OS-NMA for civilian receivers	detection	Gallardo & Yuste (2020)	Conference	2020
P43	10.14722/ndss.2022.23071	SemperFi: Anti-spoofing GPS Receiver for UAVs	Sathaye et al.	None ML, Extended Kalman Filter (EKF), Successive Interference Cancellation (SIC)	spoofing	Real-world Data	Recovery time, position error, spoof suppression performance	Experimental testing on real UAVs, GPS SDR platform, simulated and real spoofing traces		UAV navigation resilience	Mitigation	(Sathaye et al., 2022)	Conference	2022
P44	10.3390/drones8090515	SSRL-UAVs: A Self-Supervised Deep Representation Learning Approach for GPS Spoofing	Alanazi, A.	SSRL + DL (LSTM-GRU, LSTM-RNN, DNN)	spoofing	Simulated	Accuracy, precision, recall, F1-score, ROC, confusion matrix	Multilabel classification on GPS spoofing dataset; SSRL + transfer learning	99.9% (train), 70.0% (validation	UAVs under limited label conditions	Detection	(Alanazi, 2024)	Journal	2024
P45	10.33012/navi.616	Stochastic Reachability-Based GPS Spoofing Detection with Chimera Signal Enhancement	Mina, Kanhere, Kousik, & Gao, 2023	Uses stochastic reachability and SR-based Kalman filtering (no ML)	spoofing	Simulated Data	Correct Authentication Rate (CAR), Missed Detection Rate (MDR), Monte Carlo simulations	SR-based Kalman and extended Kalman filters with 1000-run simulations	99	Continuous spoofing detection between authentication intervals in GPS L1C	Detection	(Mina, Kanhere, Kousik, & Gao, 2023)	Journal	2023
P46	10.1109/VTC2022-Spring54318.2022.9860616	Two methods for Jamming Identification in UAV Networks using New Synthetic Dataset	Viana et al.	Statistical Time Series Decomposition + Deep Neural Network (DNN)	jamming	Simulated Data	Accuracy, detection rate	Simulation based evaluation with varying jammer positions and powers	Statistical method: 84.38% detection rate at 30 m jammer distance. DNN: ~99.99% accuracy for jamming power >? and jammer distance <200 m	Jamming detection in UAV networks	Detection	(Viana et al., 2022)	Preprint (arXiv)	2022
P47	https://doi.org/10.3390/s24186156	Detection of GPS Spoofing Attacks in UAVs Based on Adversarial Machine Learning Model	Alhoraibi et al.	CNN + LSTM with GAN-based Adversarial Training	Spoofing	Real-world GPS spoofing dataset with 510,530 samples	Accuracy, precision, recall, F1-score, loss; confusion matrix; AUC (ROC); K-fold cross-validation	Real dataset + synthetic adversarial samples	98% detection accuracy (GAN-enhanced CNN); 80% CNN base model; 71% LSTM	Detecting GPS spoofing in UAVs using adversarial ML for enhanced physical-layer security	Detection	(Alhoraibi et al., 2024)	Journal	2024
P48	https://doi.org/10.1016/j.cose.2022.103080	Machine Learning Modeling of GPS Features with Applications to UAV Location Spoofing Detection and Classification	Nayfeh et al.	ML – Random Forest, KNN, MLP, Logistic Regression, SVM, Decision Tree, Naive Bayes	Spoofing	Real	Detection Rate, Precision, Recall, F1-score, False Alarm Rate, Misdetection Rate	Location-dependent vs. independent classifiers trained on real spoofed data using 10-fold cross-validation	Up to 96.67% detection rate (Logistic Regression), MDR: 4.26%, FAR: 1.08%	Real-time detection and classification of GPS spoofing in UAVs without hardware modification		(Nayfeh et al., 2023)	Journal	2023
P49	doi.org/10.3390/RI7010014	A Lightweight AI-Based Approach for Drone Jamming Detection	Cibecchini, Chiti, & Pierucci	ML – Lightweight Neural Network	Jamming	Simulated	Accuracy, computational load, responsiveness	Simulated scenarios and AI model testing	>97% detection accuracy	jamming attacks on drones using AI-based lightweight classification	Detection	(Cibecchini, Chiti, & Pierucci, 2025)	Journal	2025
P50	doi.org/10.48550/arXiv.2501.02352	GNSS/GPS Spoofing and Jamming Identification Using Machine Learning and Deep Learning	Ghanbarzadeh et al.	ML & DL – XGBoost, Random Forest, SVM, CNN, ResNet18, etc.	Both	Real	Accuracy, Precision, Recall, F1-score, ROC-AUC, Confusion Matrix	Extensive evaluation on real datasets with cross-validation, hyperparameter tuning, and augmentation	XGBoost: 94.44%, Decision Tree: 94.33%, Random Forest: 94.22%	spoofing and jamming in GNSS for UAVs and autonomous vehicles using ML/DL	Detection	(Ghanbarzadeh, Soleimani, & Soleimani, 2024)	Preprint	2024
P51	doi.org/10.1016/j.csa.2025.100094	Multimodal Deep Neural Network for UAV GPS Jamming Attack Detection	Abdollahpour, F., & Vakilinia, O.	DL – Multimodal (MLP + CNN)	Jamming	Simulated	Accuracy, Precision, Recall, F1-score, Confusion Matrix	Multiclass classification with tabular features and image-based spectrograms, fusion via concatenation	MLP: 98.75%, CNN: 94.68%, Multimodal DL: 99% accuracy	Detecting and classifying multiple types of GPS jamming attacks in UAVs	Detection	(Abdollahpour & Vakilinia, 2025)	Journal	2025
P52	doi.org/10.1109/JSEN.2022.3174019	A New Induced GNSS Spoofing Detection Method Based on Weighted Second-Order Central Moment	Zhou, W., Lv, Z., Deng, X., & Ke, Y.	Not ML based, statistical method using signal processing	Spoofing	Real	Detection Probability (Pd), False Alarm Rate (FAR), ROC Curve	Signal Quality Monitoring (SQM) metrics and comparison with benchmark methods (Ratio, Delta, ELP metrics)	WSCM metric showed 24.15% higher detection probability than baseline at 10% false alarm rate	Detecting induced spoofing in GNSS receivers via signal symmetry distortion	Detection	(Zhou, Lv, Deng, & Ke, 2022)	Journal	2022