A Systematic Mapping Study on Self-Adaptation in SDN based IoT Networks

Imran Hafeez

Master's thesis

Supervisor: Adnan Ashraf

The Faculty of Science and Engineering

Åbo Akademi University

Abstract

Background: The Internet of Things (IoT) has led to widespread adoption of software-defined networking (SDN) to manage IoT networks. However, challenges like scalability, performance, and security necessitates self-adaptation capabilities in SDN-based IoT networks.

Objective: This study aimed to systematically map research on self-adaptation approaches, issues, and metrics in SDN-based IoT networks.

Method: Following systematic mapping guidelines, an extensive literature search was conducted across scientific databases. After screening, 32 relevant studies were selected for analysis. Data extraction and synthesis was performed to identify adaptation approaches, publication trends, and key issues.

Results: Machine learning and deep learning are the prevailing methods for adaptation. Most research findings have been disseminated through academic journal articles, with the highest number of 11 studies published in 2020, followed by a gradual decrease. These studies primarily address significant challenges, including scalability, congestion, energy efficiency, service quality, and security.

Conclusion: This mapping study offers a current overview of research in the field and identifies areas where further investigation is needed, which can serve as a roadmap for future research. Expanding the focus beyond SDN-IoT, exploring novel adaptation approaches, and creating reusable frameworks are suggested as promising research avenues. These insights provide a basis for advancing research on self-adaptive SDN for IoT.

Table of Contents

List of Figures

List of Tables

List of Acronyms

- IoT - Internet of Things
- SDN - Software-Defined Networking
- SMS - Systematic Mapping Study
- MAPE-K - Monitor, Analyze, Plan, Execute, Knowledge
- PICOC - Population, Intervention, Comparison, Outcomes, Context
- RQ - Research Question
- ML - Machine Learning
- DL - Deep Learning
- MADQNs - Multi-Agent Deep Q-Networks
- MTD - Moving Target Defense
- AQRA - Application-Aware QoS Routing Algorithm
- FNNN - Fuzzy Normalized Neural Network
- QoS - Quality of Service
- RCT - Randomized Controlled Trial
- SLR - Systematic Literature Review
- API - Application Programming Interface
- ONOS - Open Network Operating System
- ODL - OpenDaylight
- CDPI - SDN Control-Data Plane Interface
- NBI - Northbound Interface
- DNNs - Deep Neural Networks
- CNNs - Convolutional Neural Networks
- RNNs - Recurrent Neural Networks
- GANs - Generative Adversarial Network

1 Introduction

The rapid expansion of embedded systems and their computing and communicating capabilities has produced a new era of Internet technology, commonly referred to as the *Internet of Things (IoT).* Coined by *Kevin Aston in 1999, the Internet of Things (IoT)* is a network of "things" that possess sensing capabilities and limited computational power and can communicate the data they sense using standard protocols. [1, p. 3]. This network of smart objects provides a large amount of the data that can be used for further processing and analysis. Essentially, IoT is a network of intelligent objects with sensing and computational capabilities. [1, p. 3].

A software-defined network (SDN) architecture is commonly used as the primary communication backbone for many Internet of Things (IoT) systems. This technology allows IoT networks to automatically adjust themselves using a software. Imagine it as a system that keeps an eye on IoT devices, figures out if something is going wrong, and then makes changes to fix the problem in real-time. It's like having a smart system that can constantly check, think, and make improvements as needed [2].

SDN makes it possible to have a centralized control plane that manages network resources and routes traffic. This makes it easier to manage IoT networks, which often have several devices and sensors that are connected to each other. However, as the *number of devices and sensors in an IoT network grows*, the network becomes more complicated. This results in problems such as *network congestion, latency, and downtime*, which can be tackled using self-adaptation strategies. In short, including self-adaptability is a smart option in many software-heavy systems. Adaptation methods consist of fixed, pre-defined actions discovered based on domain knowledge or they consist of novel behaviors or entities introduced at runtime [3].

1.1 Need for a systematic mapping analysis

Above mentioned problems can hinder the performance of SDN-based IoT networks and how self-adaptation strategies can be used to remedy these issues. Self-adaptation has been the subject of research for quite some time [1]–[3]. In the context of self-adaptation in SDN-based IoT networks, a systematic mapping analysis can help identify current research trends, the most often employed self-adaptation strategies, and open research issues.

A systematic mapping study (SMS) is a rigorous method for identifying, categorizing and synthesizing research on a particular topic. The purpose of an SMS is to provide an overview of existing research and to identify gaps and opportunities for future research.

In the case of self-adaptation in Software-Defined Networking (SDN)-based Internet of Things (IoT) networks, an SMS can be useful for several reasons:

1.1.1 Identifying research trends

An SMS can help identify the current state of research on self-adaptation in SDN-based IoT networks. This can help researchers and practitioners to understand the most active research areas, the most common research methods used, and the most promising avenues for future research.

1.1.2 Identifying gaps

An SMS can also help identify gaps in existing research. For example, it may reveal areas where there is very little research or where existing research is not sufficient to address important challenges.

1.1.3 Identifying challenges

An SMS can help identify challenges and issues that need to be addressed to enable effective self-adaptation in SDN-based IoT networks. This can include technical challenges related to network architecture and protocols, as well as broader challenges related to governance, privacy, and security.

1.1.4 Facilitating forthcoming research

An SMS can provide a foundation for future research on self-adaptation in SDN-based IoT networks. By identifying research gaps and challenges, it can help guide future research efforts to address important questions and advance the state-of-the-art.

Generally, a systematic mapping study on self-adaptation in SDN-based IoT networks can be a valuable tool for researchers and practitioners looking to understand the current state of research, identify gaps and challenges, and guide future research efforts.

The study's findings are used to shed light on the current status of self-adaptation in SDN-based IoT networks and to answer the questions, which are mentioned in Section 3.4**.** The findings can be of interest to researchers and practitioners who are interested in self-adaptation in SDN-based IoT networks and can aid in identifying the most significant research problems and gaps in the field. This detailed mapping study adds to the knowledge of self-adaptation in SDN-based IoT networks and lays the groundwork for future research in this field.

The remaining chapters are organized as follows: *Chapter 2* delves into the fundamental aspects of the subject, while *Chapter 3* outlines the study's design. *Chapter 4* provides insights into the execution of the systematic mapping study, and *Chapter 5* unveils the findings. *Chapter 6* engages in the discussion and gap analysis, while *Chapter 7* evaluates potential threats to validity. Finally, *Chapter 8* offers a concise summary and draws conclusions for the thesis.

2 Background

In this chapter, the foundational concepts of the thesis are described.

2.1 Self-adaptation

Self-adaptation in computer engineering is the capacity of a system to dynamically modify its behavior and/or structure in response to changes in its environment or internal state. Self-adaptive systems are designed to be more robust and resilient, capable of withstanding unanticipated events or changes, and able to optimize their performance over time.

According to [4, p. 1], modern *software-intensive systems* are expected to function under uncertain conditions and without interruption. Changes in the operational environment, fluctuations in the availability of resources, and differences in user objectives are examples of potential sources of uncertainty. Traditionally, such ambiguities are the responsibility of system operators. However, these management tasks can be complex, error-prone, and costly. The purpose of self-adaptation is to allow the system to collect additional data about uncertainties during operation, so that it can manage itself based on high-level goals. The system uses additional data to resolve uncertainties, and based on its objectives, reconfigures or adjusts itself to accommodate changing conditions.

Self-adaptation in computer engineering can be achieved through various methods, including:

- Algorithms for machine learning that enable the system to learn from experience and make data-driven decisions to adapt to changing conditions.
- Rule-based systems that define a set of rules for the system to adhere to and reconfigure them as needed to achieve the desired result.
- Evolutionary algorithms that mimic natural selection to optimize the performance and behavior of a system over time.
- Continuous monitoring of the system's output and adjusting its input to achieve a desired set of goals through feedback control loops.

Self-adaptive systems are used in a wide range of places, from self-driving cars to smart homes, where they can improve the performance, dependability, and safety of the system.

2.1.1 Management system model of adaptive systems

The model consists of four parts, as stated in [4, p. 5].

- Environment
- Managed system
- Feedback loop
- Adaptation Goals

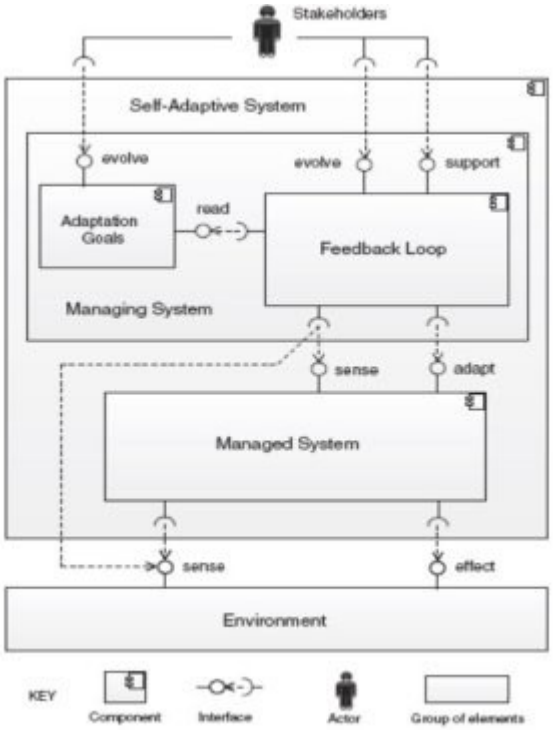

Figure 2.1: Conceptual model of self-adaptive system. Adapted from [1, p. 6]

2.1.2 Environment

In self-adaptive systems, the environment refers to the external world, which can consist of users, physical and virtual elements [4, p. 5]. The external environment can be sensed using sensors and can be modified using effects. The difference

between an environment and a self-adaptive system is how much control the system has over itself [4, p. 6].

2.1.3 Managed system

According to [4, p. 7], the word "managed system" has been used interchangeably with terms like "managed element," "system layer," "core function," "base-level system," and "controllable plant." It is the application software that provides the intended functions and services to the system's users. This software is designed and developed to meet the specific needs and goals of the system and is responsible for processing data and performing user-requested tasks.

2.1.4 Feedback loop

Feedback loop play a crucial role in self-adaptive systems, as they allow the system to monitor and adjust its behavior in real time. The feedback loop of a self-adaptive system typically involves the following steps:

- Monitor: Monitors the system and collects data about its execution context and internal state. This provides the up-to-date information needed for adaptation.
- Analyze: Analyzes the data collected by the Monitor and determines if the system needs to adapt. It identifies possible adaptation strategies.
- Plan: Formulates an adaptation plan to achieve the adaptation strategy identified by the Analyze component. It determines the actions needed to adapt.
- Execute: Executes the adaptation plan formulated by the Plan component. It applies the changes to the system.
- Knowledge: Represents the shared knowledge between the components like system models, goals, rules, etc. This knowledge base is used throughout the adaptation process.

The continuous feedback loop between these components allows a self-adaptive system to monitor itself and dynamically adapt its behavior at

runtime in response to changes. The MAPE-K loop enables autonomous and robust adaptation.

The feedback loop of a self-adaptive system can be either reactive or proactive. Reactive systems respond to changes in their environment, whereas proactive systems anticipate changes and take action to prevent problems before they occur.

2.1.5 Adaptation goals

Adaptation objectives are used to define the state of the intended system in response to environmental changes and can be functional or non-functional. Typically, adaptation objectives are specified by the system designers, but can also be dynamically modified based on the environment. The primary purpose of adaptation goals is to provide a systematic and efficient method for self-adaptive systems to continuously monitor, analyze, and modify their behavior to ensure the system's efficacy in dynamic environments.

2.2 Internet of things

The Internet of Things (IoT) is a topic that has been covered by various sources, including Forbes [5] and Cisco [6]. According to Forbes [5], IoT refers to the interconnection of devices, objects, and machinery through the Internet, generating and sharing data with each other. This connectivity and data exchange can result in operational efficiency and valuable insights in industries such as healthcare, transportation, and manufacturing. Cisco [6] describes the IoT as an interconnected network of devices that can be used to collect and analyze data to improve decision making and create innovative solutions. Forbes [5] further elaborates that IoT can revolutionize several industries by improving efficiencies, reducing costs, and enhancing the overall customer experience. However, both sources highlight the importance of addressing the security, privacy, and ethical concerns associated with the IoT, as the data generated by these interconnected devices can be sensitive and personal.

2.2.1 Evolution of IoT

The technologies that created the foundation for connected systems by attaining easy integration into daily life, widespread public acceptance, and significant advantages through the use of connected solutions can be considered the founding solutions for the development of the Internet of Things (IoT) [7]. The Figure shown below shows a series of technological advancements.

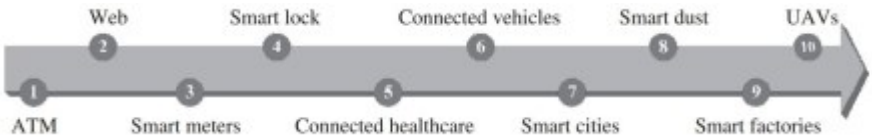

Figure 2.2: Evolution of IoT. Adapted from [2, p. 80]

2.2.2 IoT and complex interdependence amongst technologies

As mentioned earlier, IOT has evolved over a period of a few decades and its foundation rests on the complex interdependence of technologies. The IoT paradigm can be divided into four planes, as *services, local connection, global connectivity, and processing* [7]. This division can be seen in the Figure 2.3 below.

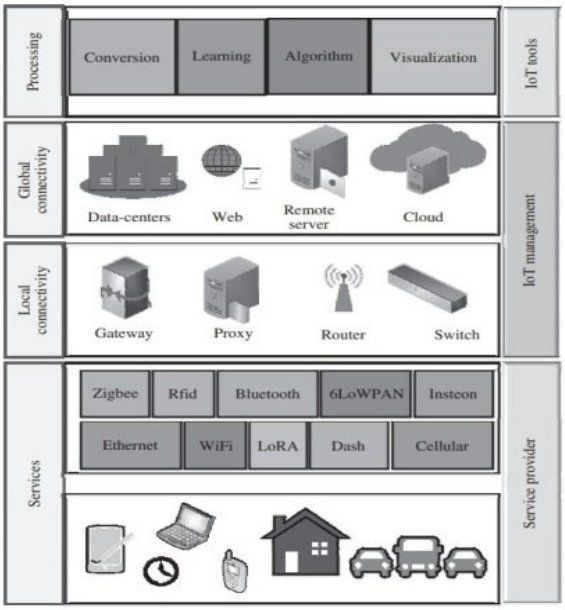

Figure 2.3: IoT interdependence. Adapted from [2, p. 85]

The service plane is composed of two parts:

1. Devices or things
2. Low-power connectivity

According to [7], this connectivity layer provides services that combine connectivity for things and low-power devices. Low-power connectivity is responsible for linking local implementation devices, such as wearables, laptops, smartphones, household appliances, smart eyewear, factory machinery, vending machines, automobiles, UAVs, and robotics. In contrast, the majority of contemporary technologies are wireless and frequently programmable. The range of these connectivity technologies is quite constrained, and they are responsible for the connectivity between Internet of Things devices and the nearest hub or gateway to the Internet. Local connectivity is responsible for delivering internet

connections to numerous IoT deployments based on their physical location, application domains, or service providers.

2.2.3 IoT networking components

Five broad categories of IoT networking components are generally described in the literature. However, the components described here are the main components that play a role when establishing any IoT network, divided into six types: 1) IoT nodes, 2) IoT routers, 3) IoT LAN, 4) IoT WAN, 5) IoT gateways, and 6) IoT proxy [7, p. 87]. The Figure below shows the typical implementation of the IoT network.

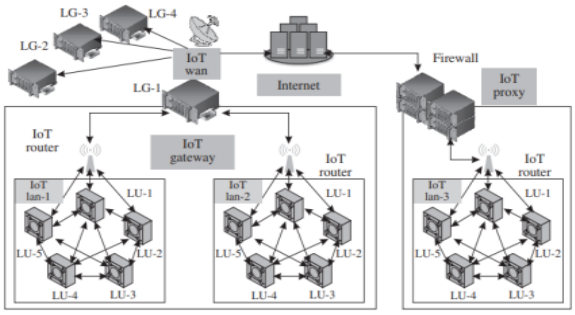

Figure 2.4: IoT networking components

1. IoT Node: An IoT node is a physical object equipped with internet connectivity and data transmission capabilities. Typically, it consists of a small sensor or actuator designed to collect and transmit data to other interconnected devices.
2. IoT Router: An IoT router is a networking device used to link IoT nodes to other networks or the Internet. With functions such as security, bandwidth management, and network monitoring, it acts as the focal point for data transmission and routing. A network of IoT devices connected to a local network, such as a home or office network, is known as an IoT LAN (local area network). Within a small space, it enables devices to exchange data and communicate with one another.
3. A network of IoT devices connected to a local network, such as a home or office network, is known as an IoT LAN (local area network). Within a small distance, it allows devices to exchange data and communicate with each other.
4. Wide Area Network (IoT WAN): An IoT WAN (Internet of Things Wide Area Network) is a network that connects IoT devices over a larger geographic area, such as a city or region. For data transmission over longer distances, cellular or satellite networks are typically used.
5. IoT Gateway: An IoT Gateway is a device that links IoT devices to other networks or the Internet. It typically connects local IoT networks to the Internet and has features such as protocol translation, data filtering, and security.

6. IoT proxy: A software component known as IoT proxy sits between IoT devices and other network resources, such as servers or databases. Devices can communicate securely and effectively with other resources because they intercept and process requests and responses.

2.3 Software-defined networks

According to VMware [8], SDN (Software-Defined Networking) is a network architecture that separates the *control and data forwarding functions of traditional networking equipment, such as routers and switches.* In an SDN environment, the network control is decoupled from the forwarding hardware and placed in a centralized software-based controller, allowing network administrators to program and manage the network from a single location. This architecture enables organizations to create a more agile and efficient network that can quickly adapt to changing business needs. Additionally, SDN facilitates the creation of network overlays, virtualizing the network infrastructure, and providing logical separation of traffic for increased security and performance. VMware further explains that SDN solutions can be deployed in various network types, including data centers, wide area, and even wireless networks, providing greater flexibility and scalability.

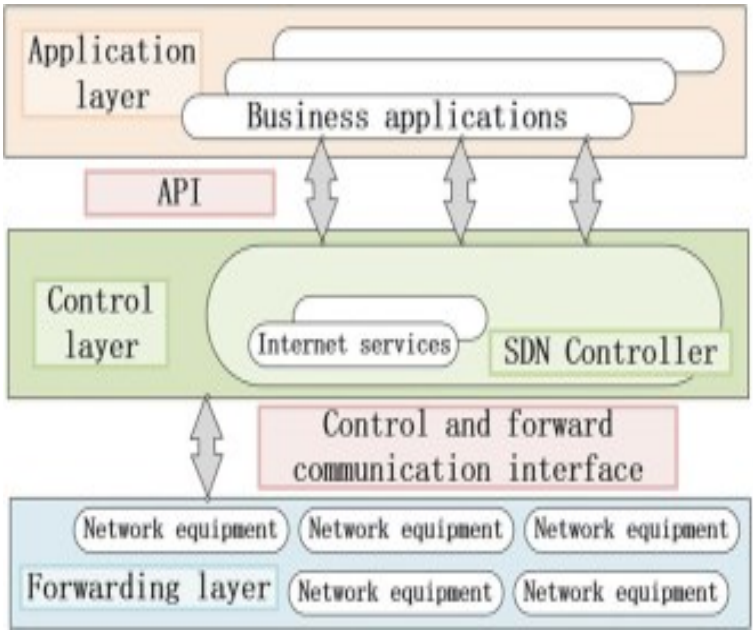

Figure 2.5: SDN block diagram

As can be seen in the Figure 2.5, the control plane exerts its control through the control-forward communication interface. The controller and network equipment are responsible for managing data traffic resulting from communication between terminals, and the network device generates a forwarding Table to determine how to process the traffic [9, p. 47].

2.3.1 Advantages and core technology

The SDN architecture comprises four planes: data plane, control plane, application plane, and management plane; and two interfaces: the SDN control data plane interface (CDPI) and the SDN northbound interface (NBI). The data plane includes network elements with SDN data paths, which are responsible for forwarding and processing data. The control plane includes the SDN controller, which is responsible for converting SDN application requests and providing an abstract model of the network. The application plane includes SDN applications that can interact with the controller through the NBI. The management plane is responsible for static tasks such as configuring network elements and defining the scope of SDN applications. The CDPI is the interface between the control and data planes, controlling forwarding behaviors and providing performance inquiry and event notifications. The NBI provides an abstract network view and allows applications to control the network's behavior.

3 Study design

According to [10], a systematic mapping study "provides a structure of the type of published research reports and results by categorizing them and frequently provides a visual summary, a map of its findings." This systematic mapping study can be broken down into three well-defined phases: planning, conducting, and reporting. The phases are depicted in the Figure given below, and the subsequent Chapters describe each phase in detail.

3.1 Planning

The planning phase of a systematic mapping study consists of the following steps: (i) identifying the need for the study, (ii) defining the research objectives and questions, (iii) defining the protocol to be followed to conduct the study, and (iv) reviewing the protocol and, if necessary, returning to step (iii).

3.2 Conducting

The following are the steps for the conducting phase.

3.2.1 Search and selection

The search and selection process involve (1) the application of a designated search string to chosen digital libraries, (2) followed by the merging and removal of impurities. (3) Selection criteria are then applied to identify primary studies for inclusion. After completion of these steps, the final set of primary studies is acquired. A detailed account of this stage is presented in Chapter 4.1.

3.2.2 Data extraction

The process involves iterating through a set of primary studies selected and extracting relevant information based on the research questions.

3.2.3 Data synthesis

The extracted data were summarized and analyzed. The results of the data synthesis will be used to answer the research questions.

3.3 Results

This Chapter describes the concluding aspect of the systematic mapping study, in which the extracted data and mapping study are discussed, and the key findings are reported.

3.4 Research goals and questions

The purpose of this systematic mapping study is to *identify, classify and evaluate* the current literature related to *"self-adaptation in SDN-based IoT networks"* and present the results of the research questions in a structured and systematic manner.

In accordance with the guidelines [11], the research questions are founded on the Population, Intervention, Comparison, Outcomes, and Context (PICOC) criteria.

- **POPULATION:** SDN-based IoT network
- **INTERVENTION:** Self-adaptation in SDN-based IoT networks
- **COMPARISON:** Not applicable.
- **OUTCOMES:** A classification of primary studies reflecting the current state-of-the-art of self-adaptation in SDN-based IoT networks
- **CONTEXT:** SDN-based IoT networks

RQ1. What approaches (methods, algorithms, techniques, frameworks, schemes, and protocols) exist for self-adaptation in SDN-based IoT networks?

Rationale: By understanding the existing approaches, researchers can identify gaps in the literature and propose new techniques to improve the efficacy of self-adaptation in SDN-based Internet of Things (IoT) networks.

RQ2. What are the publication trends?

Rationale: Reason for this question to identify the existing state of research on self-adaptation in the domain of SDN based IoT networks over the years.

RQ3. What problems/issues are addressed in the literature?

Rationale: This question is important and relevant because it can help researchers Figure out the most important problems/issues and gaps in the study. By knowing about these problems/issues and gaps, researchers can suggest new study ideas and solutions that can solve the problems/issues, fill in the gaps, and move the field forward.

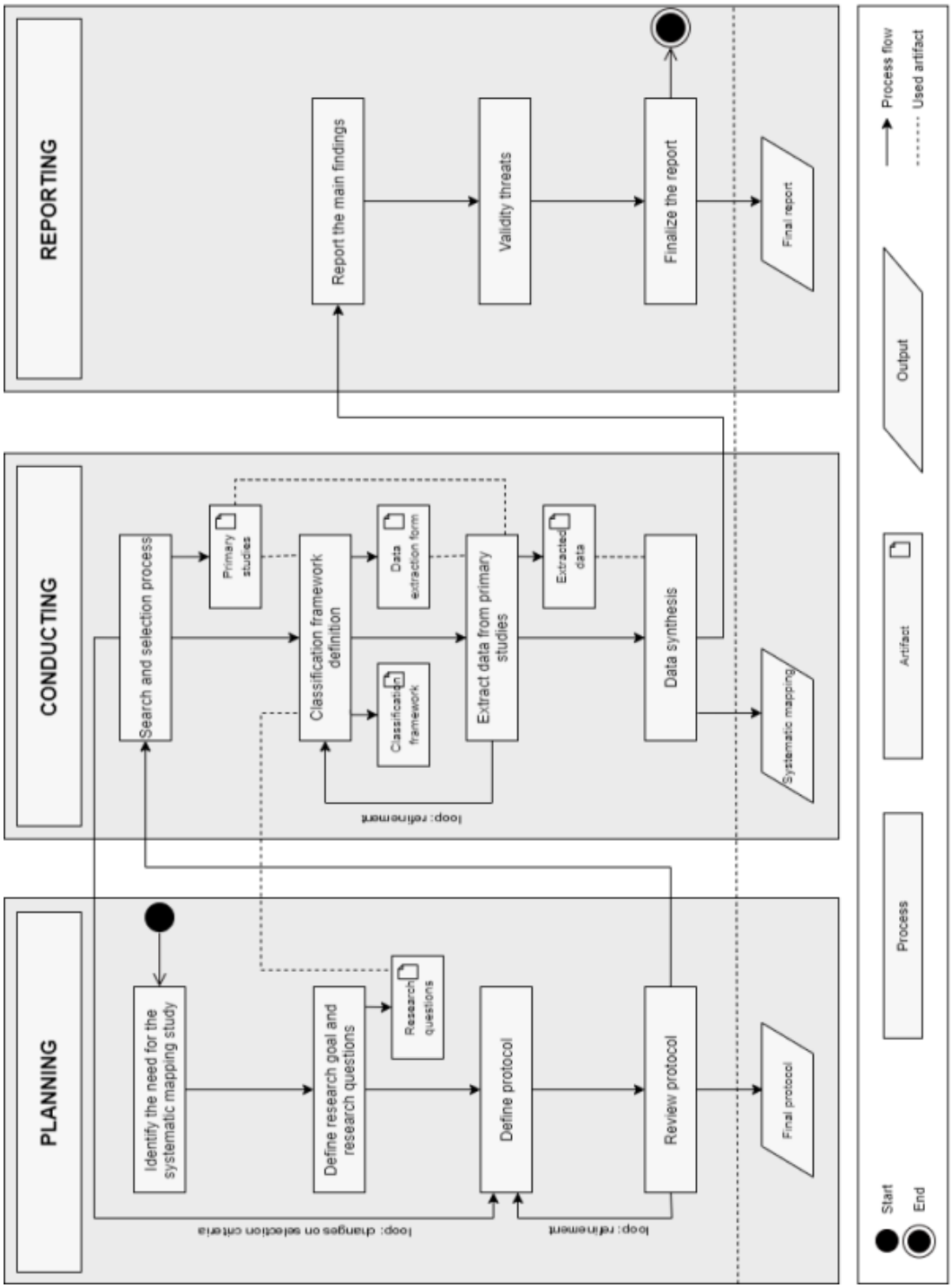

Figure 3.1: Systematic mapping study process

4 Conducting the systematic mapping study

4.1 Search and selection process

In a systematic mapping study, the search and selection procedures are a critical, multi-stage process that provides comprehensive coverage of the investigated topic. It must be carefully documented so that other researchers can verify the procedures and results.

The first step in this process is conducting an initial search of the chosen databases and libraries. Next, articles are merged, duplicates removed, and relevance assessed using the selection criteria. Irrelevant articles are further excluded by reviewing the abstracts and titles.

The subsequent step is known as "data extraction," during which relevant information is collected to form the basis of the study. In the final step, we synthesize the collected data. The illustration below depicts this process.

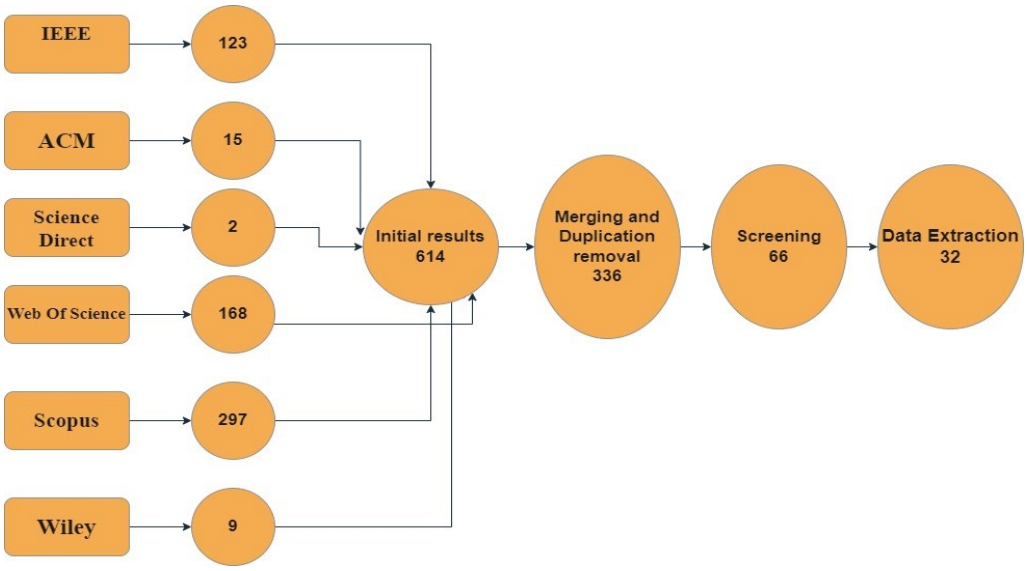

Figure 4.1: Search and selection procedure

4.1.1 Initial search

To obtain an initial set of primary studies, we used the advanced command search feature, and the libraries/databases used were *IEEE Xplore, ACM, Science Direct, Web of Science, Scopus* and *Wiley*. All these libraries/databases are well established and host a wide spectrum of peer-reviewed publications. The search was performed by considering only the abstract of studies and the Figure above shows the initial results.

((self-* OR adapt*) AND ("Software defined network" OR SDN) AND (“Internet of things” OR IOT))

The general form of the search string used was as follows:

(*) wildcard. The general form of the string was customized for ScienceDirect, which is given below.

((self-adaptation OR self-adaptive OR self-adapting) AND ("Software defined network" OR SDN) AND (“Internet of things” OR IOT))

We ran an automatic search on the selected libraries: IEEE Xplore, Scopus, ACM, Science Direct, Wiley Online Library, and Web of Science. We have introduced the generic search string using the *PICOC* criteria. A different search string was proposed for each library and is given below in Table 4.1**.** Automatic search on the selected libraries gave 614 initial primary studies.

Table 4.1: Search string

Digital Library	Search String
IEEE Xplore	("Abstract":self* OR "Abstract":adapt*) AND ("Abstract":"Software defined network" OR "Abstract":SDN) AND ("Abstract":"Internet of things" OR "Abstract":IOT)
ACM	((self\-* OR adapt*) AND ("Software defined network" OR SDN) AND ("Internet of things" OR IOT))
Science Direct	((self-adaptation OR self-adaptive OR self-adapting) AND ("Software defined network" OR SDN) AND (“Internet of things” OR IOT))
Scopus	ABS (((self-* OR adapt*) AND ("Software defined network" OR sdn) AND ("Internet of things" OR IOT))) AND (LIMIT-TO (SUBJAREA , "COMP"))
Wiley Online	"self-* OR adapt*" in Abstract and ""software defined network" OR SDN" in Abstract and ""internet of things" OR IOT" in Abstract
Web of Science	((self-* OR adapt*) AND ("Software defined network" OR SDN) AND (“Internet of things” OR IOT)) (Abstract)

4.1.2 Title and abstract screening

Throughout the merging phase, we simultaneously conducted a screening process for titles and abstracts to filter out studies that didn't meet our search criteria. While some studies could be disregarded based on their titles alone, most required assessing their abstracts too. We carefully examined keywords and sentences to determine if the study addressed the desired topic. Studies that didn't cover the desired topic were excluded, while those that seemed relevant or potentially relevant based on the title and abstract screening were selected for further consideration. This resulted in the removal of 270 studies, leaving us with 32

studies to proceed with in the next phase. Our rigorous screening process enabled us to narrow down the selection effectively.

4.1.3 Merging and impurity removal

Digital libraries/databases often contain duplicate publications, which can be addressed using reference managers such as *Zotero, EndNote*, etc. These reference managers remove duplicates by checking the title, authors, and publication year. If there are two different versions of the same study, the older version is also removed. In addition, digital databases/libraries contain a wide range of publication types such as conference proceedings, textbooks, magazine articles, etc. that are not peer-reviewed research documents. These non-research publications are excluded as well.

4.1.4 Application of selection criteria

To determine the relevance of potentially relevant studies for our mapping study, an assessment is necessary, as not all studies returned by the initial search can be utilized. To filter primary studies and only include those significantly relevant to the research questions, selection criteria are used. During this stage, potentially relevant studies are filtered based on inclusion and exclusion criteria, which examine the title, abstract, and keywords. If a study meets all the inclusion criteria and none of the exclusion criteria, it is added to the collection of primary studies. If it does not meet the criteria, it is excluded. If a study cannot be clearly excluded based on the given criteria, a thorough full-text investigation is conducted to make a final decision.

The inclusion criteria for primary studies are:

- Discusses self-adaptation.
- Presents an approach, method, algorithm, technique, or framework.
- Related to software-defined networks.
- In the context of IoT networks.
- Written in English.

- Published in a peer-reviewed journal, conference, or workshop related to computer science, computer engineering, or software engineering.

Furthermore, if multiple papers present the same test automation approach, only the most recent one will be included.

The exclusion criteria for primary studies are:

- Not related to self-adaptation.
- Does not present an approach, method, algorithm, technique, or framework.
- Not related to software-defined networks.
- Not in the context of IoT networks.
- Not written in English.
- Not published in a peer-reviewed journal, conference, or workshop related to computer science, computer engineering, or software engineering.

Only the most recent paper presenting the same self-adaptation approach will be included. If the selected papers meet all the criteria, they will be chosen for data extraction; otherwise, they will be excluded as irrelevant.

4.2 Data extraction

Data extraction is a critical step in a systematic mapping study. It involves the systematic and structured collection of the data from the identified studies that meet the inclusion criteria. Data extraction Table is given below. The Table is partitioned into two sections: general information and self-adaptation in SDN-based IoT networks. The data extraction Table was used to collect necessary details, including the extractor's name, extraction date, and a unique identifier. Furthermore, study particulars, such as title, authors' names, publication year, and source (conference, journal or workshop), were extracted. Second part of the Table was used to collect data regarding the research questions presented earlier.

Table 4.2: Data collection

Data Item	**Value**	**Additional notes**
General		
Data extractor's name		
Data extraction date		
Study identifier		
Bibliographic reference (title, authors, year, journal/conference/workshop name)		
Author affiliations and countries		
Publication type (journal, conference, or workshop)		
Self-adaptation in SDN-based IoT Networks		
RQ1: Approaches(methods, algorithms, techniques, frameworks)?		
RQ2: Publication trends for studies covering self-adaptation in SDN-based IoT networks?		
RQ3: Problems/issues addressed in the literature?		

4.3 Data synthesis

Following the data extraction, the subsequent step is data synthesis, whereby the extracted information is systematically evaluated and condensed in a manner that

is illustrative. There are two main phases of the data synthesis process: vertical analysis and horizontal analysis.

4.3.1 Vertical analysis

During vertical analysis, we scrutinize data extracted from primary studies to gather information about each parameter in our classification framework. The line of argument synthesis method is applied, where we first analyze each study separately to document its main features based on the classification framework, and then analyze the set of studies to identify potential patterns and trends.

4.3.2 Horizontal analysis

During horizontal analysis, we carefully examine the data collected from the primary studies to find any connections between the different categories assigned to each research question. It helps researchers identify patterns and differences, make cross-sectional comparisons, and test hypotheses. Data visualization and benchmarking are common uses. However, it's important to combine horizontal analysis with other methods for a comprehensive understanding of the research topic.

5 Results

In this chapter, we present the results of the vertical and horizontal analysis conducted on the extracted data. The complete list of studies, from which the data has been extracted, can be found in the appendix.

Vertical analysis aims to provide quantitative insights relevant to the research questions. It involves comparing various elements or categories within a dataset to determine their relative proportions or changes over time. Essentially, vertical analysis examines the vertical arrangement of the data by assessing the relative significance or distribution of different factors or variables within a particular context.

5.1 Result analysis of RQ1

Our first research question is, “What approaches (methods, algorithms, techniques, frameworks, schemes, and protocols) exist for self-adaptation in SDN-based IoT networks?”

As can be observed in the Figure below, there are a variety of approaches present in the literature. Some approaches were utilized in multiple studies, while others were specific to individual studies. Certain studies did not explicitly specify a particular approach and instead referred to a broader framework or model. Several approaches were used only once, such as the ADS scheme, while machine learning and deep learning were employed in several studies. The Figure 5.1, given below, displays the occurrences of different approaches in each study.

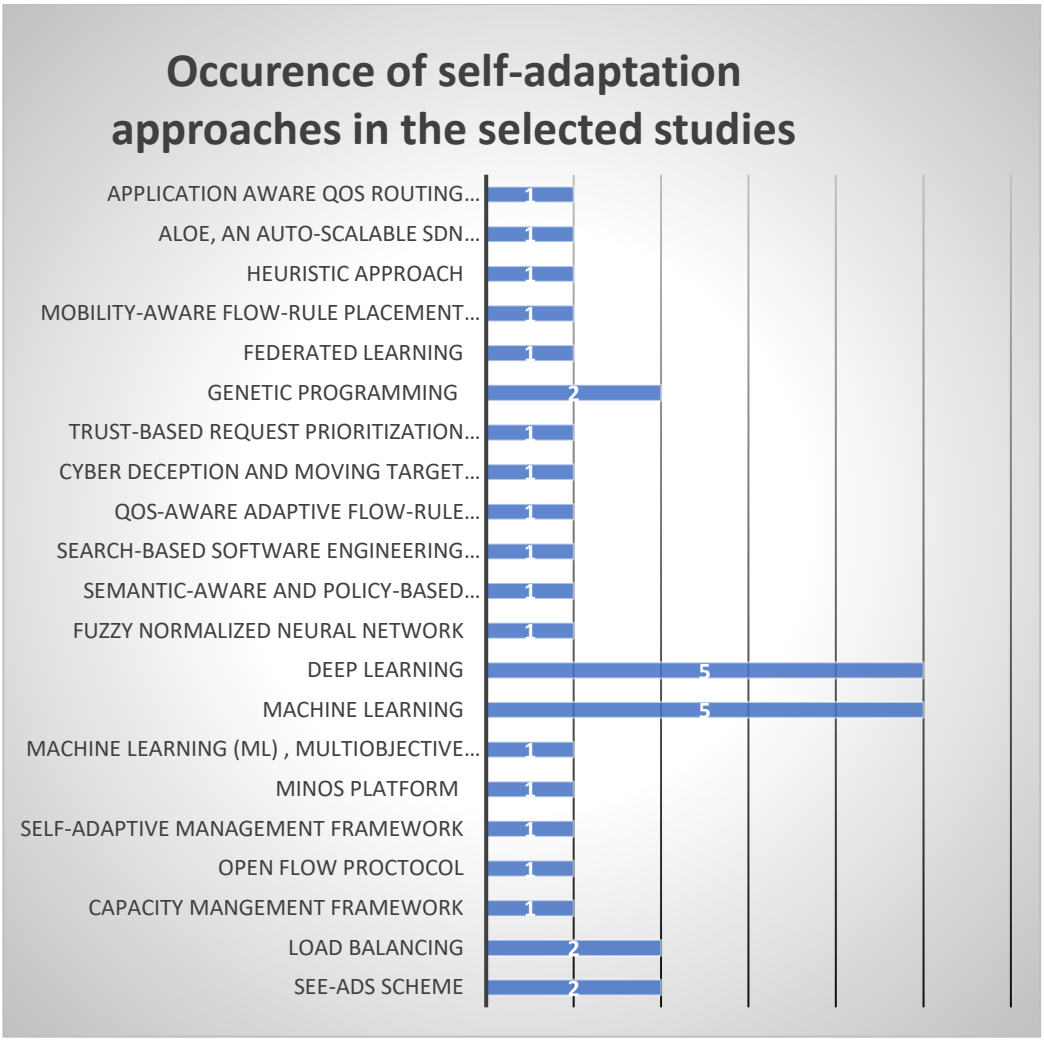

Figure 5.1: Self-adaptation approaches

As can be seen in the Figure 5.1, given above, that most used techniques are based on machine learning and deep learning. Machine learning-based techniques are discussed in these studies *S9, S15, S22, S23, S29* and deep learning-based techniques are discussed in these studies *S10, S11, S14, S28, S3.* In the Table below, the individual approaches are mentioned, along with the studies in which they appeared and the number of times they appeared.

Table 5.1: Self-adaptation approaches

Types of approaches	**Study Ids**	**Number of Occurrences**
SEE-ADS Scheme	S1, S25	2
Load balancing (based on M/M/1 queue and open flow protocol)	S2, S26	2
Capacity management framework (Game theory)	S3	1
Self-adaptive management framework (ONOS and ODL)	S6	1
MINOS platform	S5	1
Machine learning (ML), multi-objective optimization (MOO)-based techniques	S7	1
Machine learning	S9, S15, S22, S23, S29, S7	5
Deep learning	S10, S11, S14, S28, S32	5
Fuzzy normalized neural network	S24	1
Semantic-aware and policy-based security orchestration framework (SAPSOF)	S12	1
Search-based software engineering technique	S13	1
QoS-aware adaptive flow-rule aggregation scheme	S16	1
Cyber deception and moving target defence (MTD)	S17	1

techniques		
FlowJustifier based on Trust-based request prioritization algorithm	S18	1
Genetic programming	S19, S30	2
Multi-agent deep Q-networks (MADQNs)	S20	1
Mobi-flow (Mobility-aware flow-rule) placement scheme)	S21	1
HSPC-SDN approach	S22	1
Aloe, an auto scalable SDN orchestration framework	S28	1
Application aware QoS routing algorithm (AQRA)	S31	1

5.2 Definition of approaches

In this section we will provide a brief introduction to the approaches mentioned above.

5.2.1 Scalable and energy-efficient anomaly detection scheme

This has been developed as a novel solution to address the challenges encountered in anomaly detection within large-scale systems, with particular emphasis on energy efficiency. Anomaly detection holds immense significance across domains such as cybersecurity, industrial automation, and network monitoring, as it assumes a critical role in identifying aberrant behaviors or events crucial for ensuring the dependability and security of systems.

5.2.2 Load balancing

It is a crucial aspect of network management and can be achieved by leveraging the M/M/1 queue model and the OpenFlow protocol. The *M/M/1 queue*

characterizes the length of a queue within a system that operates with a single server. In this system, incoming jobs follow a Poisson process for arrivals, and the service times for each job are exponentially distributed. The *M/M/1* queue is denoted in Kendall's notation and represents the most basic form of queueing models[3]. It is highly valuable for research purposes as it allows for the derivation of closed-form expressions for various important metrics. Additionally, an extension of this model, known as the *M/M/c queue*, involves multiple servers to further explore more complex scenarios in queueing theory.

5.2.3 OpenFlow protocol

The OpenFlow protocol is a networking technology that separates the control and data planes and provides innovation and flexibility in network management and operations[4]. In the *OpenFlow* architecture, the control plane is centralized, allowing network administrators to have a global view and programmable control over the network devices [13]. This separation allows researchers and network operators to experiment with new protocols, services, and network configurations without disrupting the underlying infrastructure [13]. *OpenFlow* facilitates the development of innovative network architectures by providing a standardized protocol for communication between the control and data planes, allowing the dynamic control of network flows and enabling the deployment of new network services [13].

5.2.4 Game theory

This is a branch of applied mathematics that studies how decision-makers interact with each other in various scenarios, trying to understand the strategic behavior behind their actions[5]. In the context of Self-Adaptive Software-Defined Networking-based Internet of Things networks, Game Theory provides a powerful tool to model and analyze complex behaviors among different entities, such as devices or services, competing for resources while aiming to optimize their performance. By considering game-theoretical concepts like *Nash Equilibria* or *Pareto optimality*, researchers can design SDN-based IoT systems capable of adapting themselves efficiently and effectively under changing conditions [6].

5.2.5 Self-adaptive management framework

A *self-adaptive management framework* is a system designed to autonomously control and manage distributed networks, such as Software-Defined Networking (SDN) controllers in the Internet of Things (IoT) environment [7]. The primary goal of this framework is to enable dynamic, flexible, and self-adaptive network controllers that can automatically respond to business demands and changes in the underlying network topology [16]. Two popular SDN controllers used in a self-adaptive management framework are *ONOS (Open Network Operating System) and ODL (OpenDayLight)* [16].

ONOS (Open Network Operating System) and ODL (OpenDaylight) are open-source platforms for managing and controlling network infrastructure in Software-Defined Networking (SDN) environments. The focus of ONOS is on scalability and resilience, offering a modular architecture and supporting various network applications and protocols. ODL, on the other hand, prioritizes flexibility and programmability, providing rich APIs and plugins for developers to build customized network applications. These platforms contribute to the advancement of SDN by enabling open, interoperable, and innovative networking solutions.

5.2.6 MINOS

The MINOS platform is a software-defined, multiprotocol IoT networking solution designed to address challenges related to elasticity, heterogeneity, and mobility in IoT environments [8]. It dynamically deploys and configures networking protocols based on application requirements and changing network conditions. The architecture comprises three interconnected planes: the Data Communication Plane, the Control Plane, and the Application Plane. Within this platform, experiments were conducted to compare the performance of two supported protocols, Adaptable-RPL and CORAL-SDN, with the standard RPL in a smart city network scenario [17]. The results demonstrate that dynamic protocol configuration through MINOS enhances network performance and adaptability.

5.2.7 Multi-objective optimization (MOO) techniques

The *SDN-Enabled Adaptive and Reliable Communication Approach*, focuses on enhancing the Quality of Service (QoS) in IoT applications by combining Software-Defined Networking (SDN), Machine Learning (ML), and *Multi-objective Optimization (MOO)* techniques[9]. The approach predicts link reliability using an ML-based k-nearest neighbors algorithm and employs NSGA-II for finding Pareto-optimal communication paths balancing end-to-end delay and path reliability. An adaptive decision mechanism in the SDN controller selects suitable paths for various IoT applications (real-time, delay-sensitive, or task offloading) based on packet types. Experiment results show the approach's effectiveness in optimizing QoS for IoT applications.

5.2.8 Machine learning

As can be seen in the Table above several studies *(S9, S15, S22, S23, S29, S7)* are using machine learning for self-adaptation. *Machine learning (ML)* is a subfield of artificial intelligence (AI) that revolves around the creation and development of algorithms and models. Its primary goal is to empower computer systems to learn, make predictions, and make decisions without relying on explicit programming. This is achieved through the utilization of mathematical models and algorithms, enabling computers to effectively analyze and interpret vast volumes of the data. By doing so, machine learning allows computers to identify patterns within the data and subsequently generate predictions or take appropriate actions.

ML applications in SDN-IoT include network traffic prediction, anomaly detection, QoS optimization, resource management, and network fault prediction/self-healing. ML models analyze historical and real-time data to accurately predict traffic patterns, detect anomalies, optimize QoS parameters, allocate resources intelligently, and predict network faults [10]. These ML techniques enhance network performance, resource utilization, security, and reliability.

5.2.9 Deep learning

Deep learning, a subset of machine learning, can be applied in SDN-based IoT networks for self-adaptation. Deep learning models excel at analyzing complex patterns and large-scale data. They can be used for traffic classification, anomaly detection, predictive analytics, resource management, fault detection and recovery, and energy optimization [11]. Deep learning enables the network to dynamically adapt and optimize based on changing conditions. However, it requires significant computational resources and labeled data for training, and its decision-making process can be difficult to interpret [12]. Overall, deep learning shows promise in enhancing performance, security, and resource efficiency in SDN-based IoT networks.

Deep learning offers several approaches and methods for self-adaptation in SDN-based IoT networks:

- Deep Reinforcement Learning combines deep learning with reinforcement learning for adaptive decision-making based on feedback and rewards. It optimizes resource allocation, routing decisions, and QoS management.
- Deep Neural Networks (DNNs) handle complex patterns and large-scale data. They are used for traffic prediction, anomaly detection, fault diagnosis, and QoS optimization[13].
- Convolutional Neural Networks (CNNs) analyze images and find applications in image-based anomaly detection, video surveillance, and security-related self-adaptation tasks.
- Recurrent Neural Networks (RNNs) analyze sequence data and are used for time-series analysis, such as network traffic prediction, fault detection, and event-based self-adaptation.
- Generative Adversarial Networks (GANs) generate synthetic data to augment training datasets, enhancing robustness and diversity of deep learning models. They are used for data generation in scenarios with limited real-time data.

- Autoencoders perform data representation and dimensionality reduction. They support anomaly detection, network optimization, and feature extraction in SDN-based IoT networks.

The selection of the approach relies on the particular self-adaptation task, the data accessible, and the requirements of the network. Ongoing efforts by researchers involve investigating and creating novel techniques to tackle the difficulties encountered in this domain.

5.2.10 Fuzzy normalized neural network

In the context of SDN-based networks, *a fuzzy normalized neural network (FNNN)* combines fuzzy logic and neural network techniques. The FNNN takes input variables, applies fuzzy logic membership functions, normalizes the inputs, and uses a neural network architecture for decision-making [14]. It learns from labeled data to establish relationships between inputs and adaptation actions. The FNNN's fuzzy logic and learning capabilities enable it to handle uncertainty and complexity in the network. It autonomously makes adaptive decisions, such as resource allocation or routing optimizations, to enhance network performance, security, and resource utilization. Careful consideration must be given to network architecture, training data, fuzzy logic rules, normalization techniques, and learning algorithms during the design and implementation of the FNNN.

5.2.11 Semantic-aware and policy-based security orchestration framework

The semantic-aware and policy-based security orchestration framework (SAPSOF) is a solution specifically developed for managing security in scenarios involving software defined networking (SDN), network function virtualization (NFV)-aware systems, and the Internet of Things (IoT) [15]. Its primary objective is to achieve autonomous and conflict-free security orchestration while optimizing the allocation of virtual Security functions (VSF) and service function chaining (SFC) [16]. By leveraging Semantic technologies, the framework can comprehend and process the underlying semantics of IoT system models and management policies. This capability enables dynamic detection and resolution of conflicts during the orchestration process. The framework comprises several key

components, including Semantic-aware Orchestration, Knowledge-based Inference Engine, and Policy Interpreter. These components work collaboratively to effectively manage and ensure the consistency of security orchestration policies. Additionally, the framework considers factors such as Quality of Service (QoS), available resources, and security conditions to make informed decisions. Ultimately, this framework streamlines security management by applying semantic understanding and policy-based approaches, enabling efficient security orchestration in complex SDN, NFV, and IoT environments.

5.2.12 Search-based software engineering (SBSE)

(SBSE) is a subfield of software engineering that utilizes search algorithms and optimization techniques to tackle various software engineering challenges. When applied to Software-Defined Networking (SDN), SBSE can optimize and automate different aspects of SDN, such as network management, configuration, and resource allocation.

SBSE algorithms can automatically generate optimal network topologies for SDN, which can minimize congestion, maximize bandwidth utilization, or reduce network latency by exploring different routing strategies [17]. Resource allocation in SDN can also be optimized using SBSE techniques, which considers factors such as traffic patterns, bandwidth requirements, and Quality of Service (QoS) constraints, to suggest resource allocation strategies that maximize network efficiency and satisfy application-specific requirements [18].

Furthermore, *SBSE algorithms* can assist in determining the optimal placement of network functions in an SDN infrastructure, which can minimize communication delays and maximize system performance by analyzing factors such as traffic patterns, latency requirements, and resource availability [19]. SBSE can also be employed to search for optimal configuration parameters for SDN controllers, considering several factors such as network topology, traffic patterns, and performance objectives, which can lead to improved controller performance, and network efficiency [20].

SBSE also includes Pareto search which has been commonly adopted in the presence of multiple objectives to be optimized in Search-Based Software Engineering (SBSE). It searches for a good approximation of the problem's Pareto optimal solutions, from which the stakeholders choose the most preferred solution according to their preferences [21].

5.2.13 QoS-aware adaptive flow-rule aggregation scheme

A QoS-aware adaptive flow-rule aggregation scheme is a mechanism designed to enhance network performance and quality of service (QoS) in software-defined networking (SDN) environments. This scheme aims to optimize the efficiency of flow rule processing in SDN controllers by dynamically aggregating similar flow rules into a single rule, thereby reducing the overall number of rules that need to be processed [22]. By leveraging intelligent algorithms and QoS metrics, such as bandwidth requirements, latency constraints, and packet loss tolerance, the scheme intelligently groups and merges flow rules that share common QoS characteristics [22]. This adaptive approach ensures that network resources are utilized efficiently while maintaining the desired QoS levels, ultimately improving network performance and reducing the processing overhead in SDN controllers.

5.2.14 Cyber deception and moving target defense

In the domain of SDN networks, *cyber deception and moving target defense (MTD)* techniques are employed to enhance network security and mitigate cyber-attacks. Traditional network defense techniques, based on static configurations and services, are insufficient in countering sophisticated attacks [23]. MTD, however, offers an intelligent and proactive countermeasure by constantly reconfiguring the underlying systems, making it difficult for attackers to exploit vulnerabilities[32] [33]. By dynamically changing potentially vulnerable points and system parameters, MTD confuses attackers and nullifies their reconnaissance activities, thus reducing the effectiveness of cyber-attacks [24]. MTD can be effectively implemented in SDN environments, which provide flexibility and centralized control over network operations [34] [35]. The use of artificial intelligence techniques, such as Bayesian attack graph analysis, can further enhance the

effectiveness of MTD in decision-making and security risk assessment[24]. Additionally, MTD can be integrated with other tools and countermeasures already deployed in the network without affecting their behavior [27]. The emergence of technologies like NFV and FPGA programmable acceleration cards has enabled the implementation of more sophisticated MTD techniques [28]. Furthermore, combining MTD with cyber deception methods, such as signal games, can improve the effectiveness of network defense [29]. Overall, MTD techniques in SDN networks provide proactive and dynamic defense mechanisms to protect against evolving cyber threats and ensure network security [30] .

5.2.15 FlowJustifier

Sarwar et al. *(S18)* [31] proposed a technique for mitigating DDoS attacks on SDN controllers in the IoT environment using trust management. In the study, it is argued that software-defined network (SDN) controllers are susceptible to Distributed Denial of Service (DDoS) attacks. These attacks involve overwhelming the controller with an excessive number of new data flows, causing network failures for legitimate users. This vulnerability is even more concerning in the context of the Internet of Things (IoT) because IoT networks are typically open and more susceptible to such attacks.

The paper presents *"FlowJustifier", a request prioritization algorithm* based on a trust return value list, which assigns confidence values to users based on their network activities and prioritizes their requests accordingly. The algorithm aims to reduce the load on the controller and make the defense against attacks more effective. The paper evaluates the performance of *FlowJustifier* using simulations and compares it with other existing techniques. The results show that *FlowJustifier* can achieve higher throughput, lower latency, and lower packet loss rate than other techniques.

5.2.16 Genetic programming

Genetic programming (GP) is a technique that uses evolutionary algorithms to automatically generate computer programs that can solve a given problem. GP can

be applied to the domain of self-adaptation in SDN-based IoT networks to learn and update the data-forwarding logic of the network based on the changing conditions and requirements. One paper [32] delves into the concept that rather than generating distinct adaptation plans, the goal should be to alter the underlying logic and code of the operational system. This modification enables the system to acquire the ability to autonomously prevent future anomalies without relying on frequent self-adaptation triggers.

The paper suggests a way to make IoT networks smarter by using a technique called Genetic Programming (GP). This technique helps the network constantly improve and change the way it manages and forwards data in Software-Defined Networking (SDN)-based IoT networks. The paper evaluates the performance of the solution using synthetic and industrial data, and shows that it can achieve higher throughput, lower latency, lower packet loss rate, and lower adaptation frequency than other techniques.

5.2.17 Multi-agent deep Q-networks (MADQNs)

Multi-agent deep Q-networks (MADQNs) have been proposed as a solution to address various challenges in software-defined IoT networks. MADQNs leverage deep reinforcement learning techniques, specifically *deep Q-networks (DQNs)*, to optimize resource allocation policies, enable computation offloading decisions, and facilitate self-learning softwarization capabilities [33]. In these networks, DQNs estimate the expected long-term rewards of potential actions based on observed network conditions and resource states [27]. The deep reinforcement learning approach enables the MADQN agents to learn optimal policies through experience interacting with the IoT network environment. By using DQNs, the MADQN agents can continuously update their decision-making policies to maximize cumulative future rewards. This data-driven approach facilitates adaptive resource management and control in complex, dynamic software-defined IoT networks.

5.2.18 Mobi-flow

According to the study [34], *Mobi-flow* is a mobility-aware adaptive flow-rule placement scheme. It aims to maximize the performance of software-defined access networks (SDANs) by predicting the future locations of end-users and dynamically adapting flow-rules based on these predictions. The scheme consists of a path estimator that predicts user locations and a flow-manager that determines the optimal placement of flow-rules at access points. Mobi-Flow utilizes an integer linear programming approach to determine the optimal number of access points. The scheme does not introduce any client-side changes and can be integrated with existing SDN architectures. Its practical applications include IoT environments with both static and mobile users.

5.2.19 HSPC-SDN

HSPC-SDN stands for Heuristic Driven Self-Configuring Proactive Controller for QoS-Centric Software Defined Network. It is a controller designed for QoS-centric software defined networks (SDN). The HSPC controller utilizes heuristic-driven techniques to perform various tasks such as risk assessment, path selection, and fault recovery in the network. It considers factors such as dynamic link-quality information, cumulative congestion degree, probability of successful transmission, and link quality change index to make informed decisions [35]. Additionally, the HSPC controller applies genetic algorithms to perform disjoint multiple forwarding cum failure recovery path selection, ensuring fault-tolerant communication with QoS guarantees [35]. The HSPC controller leverages network availability information, minimal distance, and strictly no-shared component criteria to select multiple disjoint forwarding and recovery paths. Overall, the HSPC controller aims to enhance the reliability, performance, and fault tolerance of SDN networks.

5.2.20 Aloe

Aloe is an auto scalable Software-Defined Networking (SDN) orchestration framework developed specifically for Internet of Things (IoT) applications and

services. Instead of relying on large service-grade SDN controller applications, Aloe utilizes multiple lightweight controller instances called microcontrollers (μC) [36]. This distributed controller approach reduces performance bottlenecks and minimizes flow-setup delay. Aloe operates on Commercial Off-The-Shelf (COTS) hardware and open-source software, ensuring cost-effectiveness and accessibility. Its architecture and modules are interoperable and robust, allowing for customized configurations based on application requirements. The framework includes a set of Application Programming Interfaces (APIs) for autonomous and language-independent application deployment. Aloe emphasizes availability and fault-tolerance, even in the face of network partitions or failures. Performance testing of Aloe showcases its efficiency and functionality, demonstrating significant improvements compared to other frameworks.

5.2.21 AQRA

The Application-Aware Quality of Service (QoS) Routing Algorithm (AQRA) is a routing algorithm that aims to optimize QoS in SDN-based IoT networks. AQRA takes into consideration the network performance, which directly impacts the Quality of Experience (QoE) for users [32]. Traditional routing strategies may not possess QoS awareness and may fail to meet the QoS requirements of emerging network applications and services [33]. AQRA tackles this challenge by addressing the multi-constrained nature of QoS routing, which is a complex problem in routing research [34]. The algorithm incorporates concepts from machine learning, such as reinforcement learning (RL), to achieve adaptive routing and enhance QoS [32].

5.3 Result analysis of RQ2

"What are the publication trends for studies covering self-adaptation in SDN-based IoT networks?"

Chart below showing the number of studies published in each year.

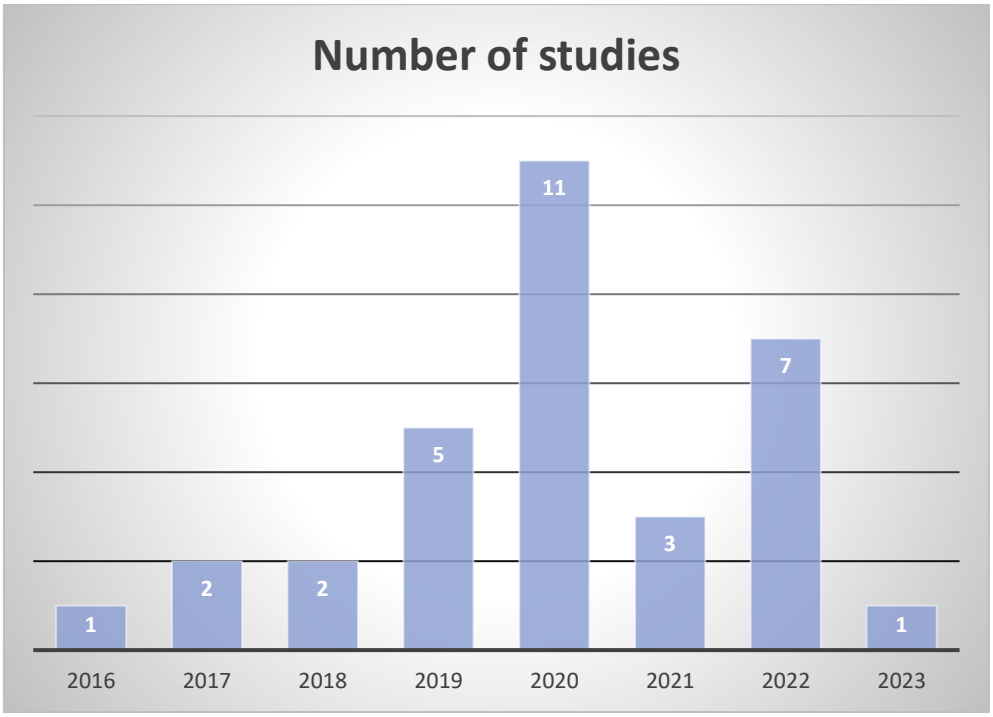

Figure 5.2: Years of publication

It can be observed that most studies were published in the year 2020, with fewer studies appearing in subsequent years. This decline in the number of studies can be attributed to our focus on SDN-based IoT networks. In 2006 [37], Rohr et al. published a classification scheme for self-adaptation research, which served as one of the pioneering taxonomies in the field. A classification scheme that organizes self-adaptation based on various dimensions. These dimensions include origin, activation, system layer, controller distribution, and operation. By considering these factors, the classification scheme provides a structured framework for categorizing different approaches and understanding the diverse aspects of self-adaptation in a systematic manner. If we search based on these factors, more publications can be seen.

In computer engineering, various types of publications are used to share research and scholarly work. These include research papers, conference papers, journal articles, technical reports, books, and theses. Research papers and journal articles present original research with rigorous peer review, while conference papers focus on specific themes. Technical reports document research or projects, books cover a

wide range of topics, and theses/dissertations are written by graduate students. The choice of publication depends on the nature of the work and the target audience. Figure below shows the type of publication.

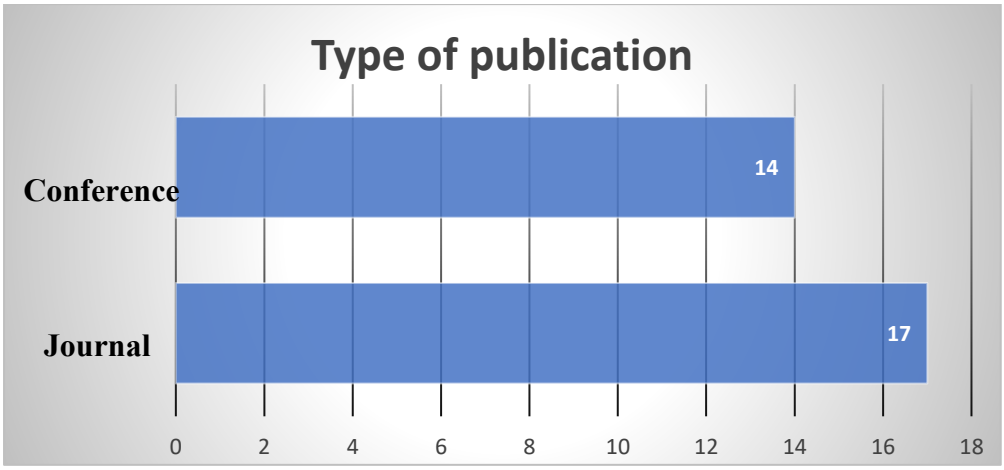

Figure 5.1: Publication type

5.4 Result analysis of RQ3

In this section, the research question "What problems/issues are addressed in the literature?" is addressed. While going through the selected publications, several issues have been encountered which are given below in the Table.

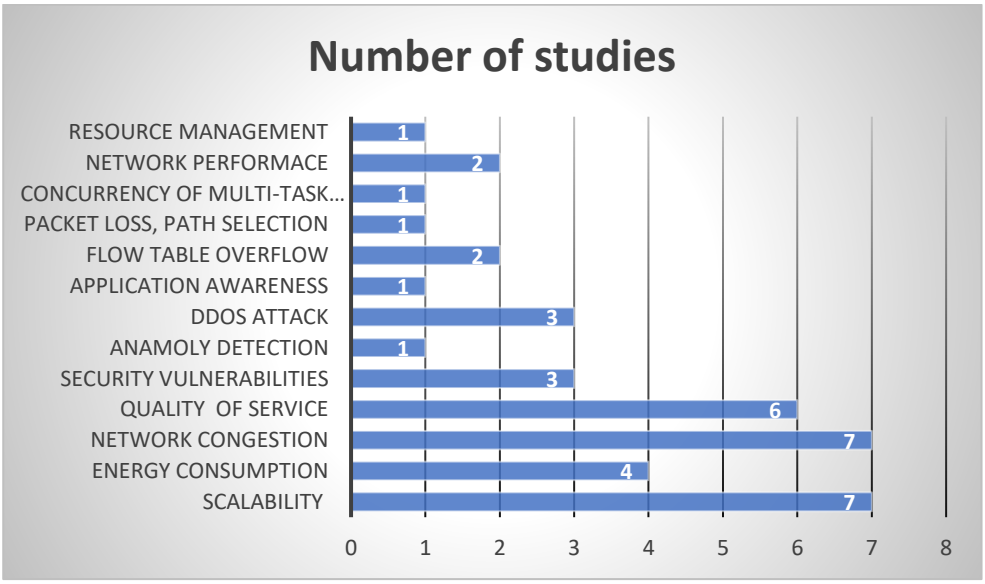

Figure 5.2: Number of studies

Table 5.2: Issues addressed in studies.

Issues	Study ID	Number of studies
Scalability	S1,S21,S25,S26,S27,S5,S6	7
Energy consumption	S2,S25,S29,30	4
Network congestion	S3,S12,S18,S19, S24, S31, S32	7
Quality of service	S3,S7,S15,S22,S26,S31	6
Security vulnerabilities	S8,S11,S16	3
Anomaly detection	S14	1
DDoS attack	S10,S17,S23	3
Application awareness	S28	1
Flow Table overflow	S15,S32	2
Packet loss, path selection	S13	1
Concurrency of multi-task application in real-time	S4	1
Network performance	S20, S9	2
Resource management	S3	1

5.5 Definition of issues

5.5.1 Scalability

Scalability is a major challenge in SDN-based IoT networks due to the massive number of devices and huge volumes of the data traffic [38], [39]. The centralized control plane of SDN architectures faces scalability issues in terms of flow setup rate, flow Table size, and control overhead as the network grows [38]. Potential solutions involve hierarchical controller designs, clustering, optimized monitoring and routing algorithms, and localizing traffic management [39]. Enhancing the scalability of SDN control and data planes is an active research area to enable future massive IoT deployments.

5.5.2 Energy consumption

SDN-based IoT networks face several issues related to *energy consumption.* Firstly, IoT devices often operate on limited power sources, requiring careful management to ensure prolonged device lifetime [40]. Secondly, as the network scales with increasing device and traffic numbers, scalability becomes a challenge, necessitating efficient energy management techniques. Dynamic network conditions, including device mobility and network reconfiguration, also need to be addressed while maintaining energy efficiency. Additionally, network protocol overhead can contribute to increased energy consumption, requiring optimization and reduction of unnecessary control traffic. Balancing energy efficiency with low-latency communication poses another challenge, as energy-saving techniques may introduce latency and slower response times [41]. Moreover, security considerations must be considered, ensuring that energy-saving measures do not compromise network security. The heterogeneity of IoT devices further complicates energy management, requiring compatibility with SDN-based infrastructure.

5.5.3 Network congestion

Network congestion refers to a situation where the demand for network resources exceeds the available capacity, leading to a degradation in network performance. It occurs when there is a high volume of the data traffic or when network resources, such as links or routers, become overloaded. Network congestion can result in increased latency, packet loss, reduced throughput, and overall degradation of network quality. Resolving network congestion is crucial to ensure efficient and reliable data transmission in communication networks. Congestion in SDN-based IoT networks can be addressed through various techniques, such as intelligent network management systems that integrate AI modules to guarantee QoS and QoE based on delay, loss rate, and jitter [42].

5.5.4 Quality of service

Providing *Quality of service (QoS)* guarantees is important in SDN-based IoT networks to meet the diverse requirements of IoT applications [43], [44]. The centralized control and global network view of SDN can enable efficient QoS management through intelligent traffic engineering, priority-based routing, and dynamic resource allocation [43]. However, existing SDN architectures lack standard interfaces and mechanisms for QoS control [53]. Key research efforts focus on extending SDN to integrate QoS configuration protocols like OpenFlow for priority queueing, rate limiting, and traffic policing on network devices [52]. Developing QoS-aware routing algorithms and controllers can also improve service differentiation in SDN-based IoT networks [53].

5.5.5 Security vulnerabilities

Security vulnerabilities refer to weaknesses or flaws in a system, network, or application that can be exploited by attackers to gain unauthorized access, disrupt operations, or compromise the confidentiality, integrity, or availability of the data or resources. These vulnerabilities can exist due to design flaws, programming errors, misconfigurations, or outdated software components. Several studies (S8, S11, S16) have identified various security vulnerabilities, including software vulnerabilities, configuration vulnerabilities, hardware vulnerabilities, social engineering vulnerabilities, and network vulnerabilities.

5.5.6 Anomaly detection

Anomaly detection is an important technique for identifying potential security threats and attacks in Internet of Things (IoT) environments. By analyzing network traffic patterns, anomaly detection can detect when unusual or unexpected activity occurs that deviates from normal behavior [45]. This enables early detection of issues like malware infections, network intrusions, and abuse of vulnerabilities before significant damage occurs.

5.5.7 Distributed denial of service (DDos)

(DDoS) attacks pose a major security threat in SDN-based IoT networks due to the centralized control plane [38]. DDoS attacks can overwhelm SDN controllers by flooding them with invalid requests and traffic flows. This disrupts the controller's ability to install flow rules and manage the network. Potential mitigation techniques include controller load balancing, fast attack detection systems, flow rule optimization, and machine learning for automated DDoS protection [47]. Further research is needed to develop adaptive and robust DDoS defense mechanisms tailored for SDN-based IoT environments.

5.5.8 Application awareness

Application awareness is an important capability in SDN-based IoT networks to enable tailored traffic management and resource optimization [43]. The centralized control plane of SDN allows controllers to identify flows and analyze traffic based on the requirements of IoT applications. This application-level visibility facilitates policy-based network control, dynamic QoS provisioning, and improved security [52]. Key enablers include application fingerprinting, deep packet inspection, machine learning for traffic classification, and open APIs for communicating application requirements to the SDN control layer. Further research on application aware SDN architectures and algorithms can enhance quality of service and efficiency in IoT networks.

5.5.9 Flow Table overflow

Flow Table overflow is a key scalability challenge in SDN-based IoT networks [38]. SDN switches maintain flow Tables containing forwarding rules installed by the controller. However, massive volumes of granular flows from high-density IoT devices can overwhelm switch flow Tables. This leads to frequent flow evictions, requests for new rules, and degraded network performance [47]. Potential solutions include optimizing flow rule wildcards, hierarchical flow management, compressing/deduplicating rules, and using fast caching structures like TCAMs.

More research is required to develop efficient flow Table management mechanisms to prevent overflow as IoT scale increases.

5.5.10 Packet loss

Packet loss is a critical concern in SDN-based IoT networks due to constraints such as unreliable wireless links and resource limitations [39]. Intelligent *path selection* algorithms enabled by the centralized network view of SDN can mitigate packet loss. The controller can dynamically calculate optimal paths based on metrics such as link quality, congestion, and latency. Machine learning techniques can also help predict network conditions and proactively re-route flows via reliable paths [48]. However, factors like control plane latency and outdated network views can degrade the performance of path selection schemes. Further research on responsive and predictive path optimization mechanisms is important to minimize packet loss in software-defined IoT networks.

5.5.11 Concurrent multi-task applications

Supporting *concurrent multi-task applications* is important in SDN-based IoT networks. The diversity of IoT use cases leads to multiple applications with varied requirements executing simultaneously. SDN controllers must manage concurrency issues like race conditions, deadlock, and resource contention when installing flow rules for concurrent apps [46]. Potential solutions involve scheduling mechanisms, policy conflict resolution, and intent-based abstractions to automatically handle concurrency. Machine learning can also help predict application behaviors and patterns to optimize concurrent executions. Further research is needed to develop SDN controller platforms that provide robust concurrency support for multi-task IoT environments.

5.5.12 Network performance

Optimizing network performance is critical in SDN-based IoT networks due to massive traffic volumes and stringent application requirements [43]. SDN's centralized control enables global visibility and programmability to monitor performance and dynamically optimize resources. Controllers can exploit real-time

network insights to efficiently route flows, balance loads, mitigate congestion, and provision QoS to enhance performance [52]. Machine learning can also help predict network demands and proactively allocate resources. However, control plane latency and overhead should be minimized to prevent degrading performance. Further research on high-performance traffic engineering and resource optimization mechanisms tailored for IoT is needed.

5.5.13 Resource management

Efficient *resource management* is critical in SDN-based IoT networks due to massive device densities and workloads [47]. Key issues include radio resource allocation in wireless domains, optimizing forwarding rules and flow Tables in switches, controller load balancing, and coordinated data/control plane resource provisioning [56]. The centralized network view of SDN allows holistic monitoring and abstraction to dynamically optimize resource utilization. However, control plane latency and synchronization issues can degrade efficiency. Further research on fast and adaptive resource control algorithms tailored to IoT environments is required to address these resource management challenges.

6 Discussion and gap analysis

This systematic mapping study was conducted to uncover valuable insights regarding self-adaptation in SDN based IoT networks. This chapter aims to describe and interpret the significance of our findings by discussing the results obtained from the vertical and horizontal analysis introduced in the previous section.

Regarding *self-adaptation approaches (RQ1)*, no single approach encompasses all aspects, there are a few approaches *(machine learning and deep learning approach)* that are more commonly utilized than others. These approaches are mostly utilized for scalability and security issues. Other approaches are used only once so we conclude that certain approaches have gained more popularity than others and application of approaches depends on the context.

Regarding *publication trends (RQ2),* the majority of selected studies, 17 studies were published as research articles and the rest were published at conferences. In 2020, 11 publications were published but then we can see a drop in publications. The drop is due to the criteria we have selected for finding the publications. But if we follow the classification scheme [37], more publications can be seen.

Regarding *issues, mentioned in the literature (RQ3),* it was noted that scalability and network congestion were leading the issues. Which makes sense because IoT networks can consists of hundreds or even thousands of resource-constrained end devices that frequently send small packets of the data to the network. This can easily overwhelm the control and data planes of traditional SDN architectures that are designed for more traditional network traffic patterns. Network congestion is closely related to scalability. With huge numbers of devices transmitting data, even small data flows can add up and congest the links in the network.

Other notable issues were energy consumption, quality of service and security vulnerabilities. Energy consumption and quality of service are closely tied to scalability and network congestion. As IoT networks scale to large numbers of devices, energy consumption increases proportionally with more data

transmissions. On the other hand, Network congestion directly leads to poor quality of service in terms of latency, packet loss and reliability. Congestion avoidance mechanisms like traffic engineering help provide QoS but require energy-intensive control and computation.

The gap analysis revealed several limitations and open research questions that present opportunities to significantly advance the research on self-adaptation in SDN-based IoT networks. A major gap is the restricted scope focusing only on SDN-IoT networks. Expanding the scope to include related research areas could provide broader insights into adaptation approaches and issues. Additionally, the heavy emphasis on technical network-level challenges indicates a need for further studies on higher-level business and human factors that influence adaptation requirements. There is also a lack of comparative quality evaluation between different self-adaptation techniques in terms of metrics like overhead, complexity, reliability, and optimality. The dominance of context-specific solutions demonstrates a need for more generalized and reusable adaptation frameworks that can be tailored to diverse use cases. Emerging network paradigms like 6G and edge computing have not been investigated and require dedicated research on their self-adaptation needs. The declining publication trend post 2020 highlights a shortage of recent studies that need to be addressed. Other gaps involve assessing user satisfaction, business impact, interactions between multiple coexisting adaptations, autonomous learning capabilities, and comparing centralized versus distributed control tradeoffs. Overall, addressing these knowledge gaps presents immense opportunities to significantly strengthen and propel research on self-adaptation in SDN-based IoT networks. Focused efforts on these limitations can both deepen understanding of core issues and broaden scope into new domains, ultimately advancing the state-of-the-art.

Bubble plot in Figure 6.1 shows the relationship between different kinds of approaches and issues and gaps in research on self-adaptation in SDN based IoT networks. The size of the bubbles indicates the prevalence of each approach and issue.

Several insights can be drawn which are given below:

- Machine learning (ML) and deep learning (DL) dominate as the most widely applied approaches. However, other techniques like genetic algorithms, game theory, etc. are not as extensively explored. There is room for more comparative evaluation and novel applications of these less prevalent approaches.
- Scalability and network congestion stand out as the most frequently addressed issues.
- Every method appears to be dedicated to a specific range of concerns. For instance, ML/DL prioritize scalability and security. Broadening the utilization of methods such as ML to encompass a wider range of issues such as anomaly detection, network congestion can generate practical solutions.
- As can be seen that several approaches and issues have minimal overlap, indicating potential opportunities. By combining the strengths of ML/DL and GA (genetic algorithm), it is possible to create algorithms that are more powerful and versatile than either technique on its own. For example, ML/DL algorithms can be used to learn the fitness function for a GA, which can help the GA to find better solutions more quickly. Similarly, GAs can be used to generate new candidate solutions for ML/DL algorithms to explore.
- Few approaches have addressed application awareness, flow Table overflow, and concurrent applications in SDN-IoT networks. Dedicated efforts to adapt SDN-IoT networks to address these neglected issues will advance the field. These issues are important challenges that need to be addressed in order to make SDN-IoT networks more reliable and efficient. By dedicating efforts to adapt SDN-IoT networks to address these issues, we can advance the field and make SDN-IoT networks a more viable solution for a wider range of applications.
- More approaches can be extended to tackle important issues like anomaly detection and security vulnerabilities. All intersections in the plot which do not have a bubble are also gaps. It means, there is the possibility to explore that area and determine whether it can generate useful solutions. For

example, open flow protocol can be studied further in the context of issues such as energy consumption, DDos attack or quality of service.

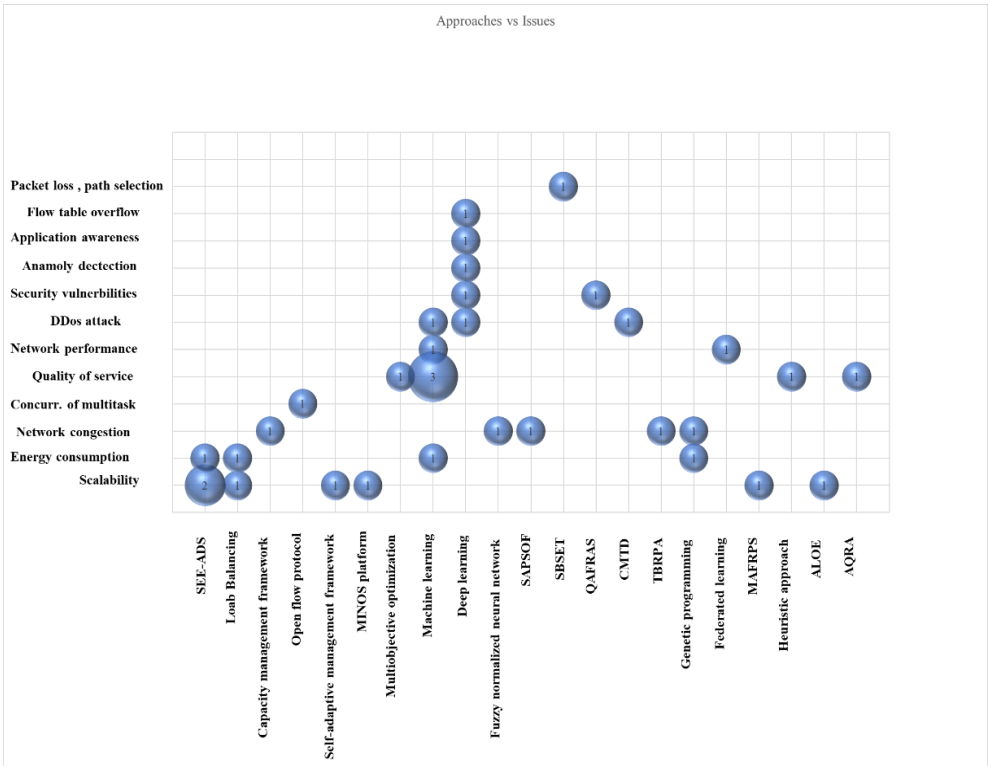

Figure 6.1: Bubble plot

7 Threats to validity

In this chapter, we explore the potential factors that could impact the validity of our study and discuss the strategies we implemented to minimize their effects.

7.1 External validity

External validity is concerned with the extent to which the final results and outcomes of a study can be applied or generalized to other contexts [48]. The generalizability of our study could be compromised by a significant threat, which involves a set of primary studies that may not adequately represent the research on our chosen topic. To address this concern, we performed an automated search on 6 comprehensive software engineering databases, namely IEEE Xplore Digital Library, SCOPUS, ACM, Science Direct, Web of Science and Wiley, which are widely recognized for their extensive coverage. By utilizing these databases, we aimed to enhance the representativeness and inclusiveness of our study's primary studies. The external validity of our study could also be at risk due to the exclusion of primary studies conducted in languages other than English. However, given that English is widely regarded as the standard language for scientific papers, this potential threat is considered negligible.

7.2 Internal validity

Internal validity in a study refers to the extent to which the design is influenced by external variables [48]. To address biases and enhance internal validity, we diligently developed and validated a comprehensive study protocol that adheres to recommended guidelines provided in [49], [50] and [48] .

7.3 Construct validity

Construct validity pertains to the validity of primary studies and the extracted data in relation to the defined research question [48]. It evaluates whether the selected studies and the data obtained from them align with and accurately address the research question at hand. A significant concern is the potential lack of representativeness in the selected primary studies, which may not adequately

encompass the population defined by the research questions. To address this concern, we took measures to mitigate it. We developed a search string by incorporating the PICOC criteria and relevant terms extracted from the research question. This approach ensured the alignment of our search with the research question and increased confidence in the validity of the search strategy.

In addition, we conducted preliminary searches on the 6 electronic databases and refined the search string based on the analysis of a set of sample studies. Subsequently, all relevant studies were screened using clear and deterministic selection criteria.

7.4 Conclusion validity

In our study on self-adaptation in SDN-based IoT networks, ensuring the trustworthiness of our conclusions is essential. We face several potential challenges to the validity of our findings. First, our search strategy was quite limited, confined to a specific set of digital libraries and a narrow search query focused on SDN-IoT networks. Expanding our search could have unearthed more relevant literature, strengthening our conclusions. Additionally, our focus on peer-reviewed academic publications in selected venues may have introduced publication bias, potentially missing relevant research in other formats.

The data analysis we conducted was relatively elementary, focusing primarily on frequencies and trends. To enhance our findings, it would have been beneficial to utilize more robust quantitative and statistical approaches, along with a variety of synthesis techniques. Additionally, it is important to note that our study's scope was confined to self-adaptation within the SDN-IoT context, which restricts its generalizability.

Furthermore, the subjective nature of study selection could introduce bias, highlighting the potential benefit of involving multiple reviewers. Although we closely followed established systematic mapping guidelines [48], enhancing our research by expanding the search, considering various publication types, conducting more robust data analysis, and involving additional reviewers could

further validate our conclusions. Replicating the study over time is also recommended to ensure the conclusions remain robust. In summary, while our systematic approach provides a reasonable level of confidence in our conclusions within the defined scope, addressing these identified limitations can enhance their validity.

8 Conclusion

This thesis presents a systematic mapping study on self-adaptation in SDN-based IoT networks with the goal of identifying and classifying the approaches/methods, issues and related metrics. To fulfill this goal, we answered the following questions:

- What approaches (methods, algorithms, techniques, frameworks, schemes, and protocols) exist for self-adaptation in SDN-based IoT networks?
- What are the publication trends for studies covering self-adaptation in SDN-based IoT networks?
- What problems/issues are addressed in the literature?

To conduct this study, we followed the guidelines on systematic reviews proposed in [49] and [48]. The initial set of potentially relevant studies consisted of 614 publications. Through a rigorous, well-documented process, 32 primary studies were selected for the final set. The results present a picture of the current situation. The results of this systematic mapping study paint a picture of the current research landscape on self-adaptation in SDN-based IoT networks.

Regarding the approaches explored (RQ1), the findings reveal that machine learning and deep learning are the most popular techniques at present. However, a variety of other approaches have also been investigated, indicating that research is still actively exploring different methods rather than converging on a single dominant approach. The choice of approach seems to depend largely on the specific context and adaptation task.

Looking at publication trends (RQ2), we see that most studies were published as journal articles, with a peak in 2020 followed by a decline in 2021-2022. This drop-off is likely attributable to the relatively narrow focus on SDN-based IoT networks. Expanding the scope beyond this specificity could potentially uncover more publication activity.

Finally, the key issues addressed (RQ3) demonstrate that scalability and network congestion are among the most pressing challenges driving research in this area.

However, concerns related to energy consumption, quality of service, and security vulnerabilities are also prevalent motivators for developing self-adaptive solutions.

Here are some notes and recommendations for future researchers based on this systematic mapping study of self-adaptation in SDN-based IoT networks:

- Expand the scope of the search beyond just SDN-based IoT networks to uncover more of the literature on self-adaptation approaches and issues. This specificity limited publications found after 2020.
- Explore emerging adaptation approaches like evolutionary algorithms, swarm intelligence, etc. that were not prevalent in the mapped studies. As research evolves, new techniques beyond ML/DL may gain prominence.
- Investigate adaptations for emerging network paradigms like 6G, which will bring new requirements and challenges beyond 5G IoT. Study how current adaptations extend to these next-gen networks.
- Examine adaptations optimized for edge computing architectures as more processing moves to the edge in IoT networks. Edge-specific issues like localization, low latency, mobility etc. need to be considered.
- Develop more standardized adaptation frameworks that can be tailored for different use cases. Current solutions are largely customized for specific contexts. Reusable frameworks can accelerate research.
- Leverage knowledge from MAPE-K feedback loops for autonomous control. Integrate monitoring, analysis, planning and execution capabilities for robust adaptations.
- Evaluate tradeoffs between distributed vs centralized control when designing adaptive SDN architectures. Distributed intelligence can aid scalability and fault tolerance.
- Study interactions between adaptation mechanisms when multiple co-exist in a system. Ensure different controllers' actions are coordinated and consistent.

I hope these recommendations provide some useful directions for advancing research on self-adaptation in SDN-based IoT networks. Findings from this mapping study can serve as a baseline for future works.

References

[1] D. Weyns, *An introduction to self-adaptive systems: a contemporary software engineering perspective*. Hoboken: Wiley, 2021.

[2] S. Misra, A. Mukherjee, and A. Roy, *Introduction to IoT*, 1st ed. Cambridge University Press, 2021. doi: 10.1017/9781108913560.

[3] J. R. Sturgul, *Mine design: examples using simulation*. Littleton, CO: Society for Mining, Metallurgy, and Exploration, 2000.

[4] N. McKeown *et al.*, 'OpenFlow: enabling innovation in campus networks', *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Mar. 2008, doi: 10.1145/1355734.1355746.

[5] C. Chi, Y. Wang, X. Tong, M. Siddula, and Z. Cai, 'Game Theory in Internet of Things: A Survey', *IEEE Internet Things J.*, vol. 9, no. 14, pp. 12125–12146, Jul. 2022, doi: 10.1109/JIOT.2021.3133669.

[6] L. Huang, G. Jia, W. Fang, W. Chen, and W. Zhang, 'Towards Security Joint Trust and Game Theory for Maximizing Utility: Challenges and Countermeasures', *Sensors*, vol. 20, no. 1, p. 221, Dec. 2019, doi: 10.3390/s20010221.

[7] I. Bedhief, M. Kassar, T. Aguili, L. Foschini, and P. Bellavista, 'Self-Adaptive Management of SDN Distributed Controllers for Highly Dynamic IoT Networks', in *International Wireless Communications and Mobile Computing Conference*, 2019, pp. 2098–2104.

[8] T. Theodorou, G. Violettas, P. Valsamas, S. Petridou, and L. Mamatas, 'A Multi-Protocol Software-Defined Networking Solution for the Internet of Things', *IEEE Commun Mag*, vol. 57, no. 10, pp. 42–48, Oct. 2019, doi: 10.1109/MCOM.001.1900056.

[9] A. Akbar, M. Ibrar, M. A. Jan, A. K. Bashir, and L. Wang, 'SDN-Enabled Adaptive and Reliable Communication in IoT-Fog Environment Using Machine Learning and Multiobjective Optimization', *IEEE INTERNET THINGS J.*, vol. 8, no. 5, pp. 3057–3065, Mar. 2021, doi: 10.1109/JIOT.2020.3038768.

[10] Z. Kazhmaganbetova, S. Imangaliyev, and A. Sharipbay, 'Machine Learning for the Communication Optimization in Distributed Systems', *Int. J. Eng. Technol.*, vol. 7, no. 4.1, p. 47, Sep. 2018, doi: 10.14419/ijet.v7i4.1.19491.

[11] S. Sengupta *et al.*, 'A review of deep learning with special emphasis on architectures, applications and recent trends', *Knowl.-Based Syst.*, vol. 194, p. 105596, Apr. 2020, doi: 10.1016/j.knosys.2020.105596.

[12] Q.-V. Pham *et al.*, 'A Survey of Multi-Access Edge Computing in 5G and Beyond: Fundamentals, Technology Integration, and State-of-the-Art', 2019, doi: 10.48550/ARXIV.1906.08452.

[13] L. Aversano, M. L. Bernardi, M. Cimitile, R. Pecori, and L. Veltri, 'Effective Anomaly Detection Using Deep Learning in IoT Systems', *Wirel. Commun. Mob. Comput.*, vol. 2021, pp. 1–14, Oct. 2021, doi: 10.1155/2021/9054336.

[14] Z. Zhang, 'An Efficient Neuro-Fuzzy-Genetic Data Mining Framework Based on Computational Intelligence', in *2009 Ninth International*

Conference on Hybrid Intelligent Systems, Shenyang, China: IEEE, 2009, pp. 178–183. doi: 10.1109/HIS.2009.148.

[15] A. Molina Zarca, M. Bagaa, J. Bernal Bernabe, T. Taleb, and A. F. Skarmeta, 'Semantic-Aware Security Orchestration in SDN/NFV-Enabled IoT Systems', *Sensors*, vol. 20, no. 13, Jul. 2020, doi: 10.3390/s20133622.

[16] A. H. Shamsan and A. R. Faridi, 'A conceptual architecture for integrating software defined network and network virtualization with internet of things', *Int. J. Electr. Comput. Eng. IJECE*, vol. 12, no. 6, p. 6777, Dec. 2022, doi: 10.11591/ijece.v12i6.pp6777-6784.

[17] J. Ye, X. Cheng, J. Zhu, L. Feng, and L. Song, 'A DDoS Attack Detection Method Based on SVM in Software Defined Network', *Secur. Commun. Netw.*, vol. 2018, pp. 1–8, 2018, doi: 10.1155/2018/9804061.

[18] F. J. Moreno-Muro, C. San-Nicolas-Martinez, M. Garrich, P. Pavon-Marino, O. G. De Dios, and R. L. Da Silva, 'Latency-Aware Optimization of Service Chain Allocation with Joint VNF Instantiation and SDN Metro Network Control', in *2018 European Conference on Optical Communication (ECOC)*, Rome: IEEE, Sep. 2018, pp. 1–3. doi: 10.1109/ECOC.2018.8535492.

[19] M. Harman, P. McMinn, J. T. De Souza, and S. Yoo, 'Search Based Software Engineering: Techniques, Taxonomy, Tutorial', in *Empirical Software Engineering and Verification*, B. Meyer and M. Nordio, Eds., in Lecture Notes in Computer Science, vol. 7007. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 1–59. doi: 10.1007/978-3-642-25231-0_1.

[20] Z. Jia, Y. Sun, and Q. Liu, 'OPSDN: an enhanced SDN simulation framework for OPNETModeler', *J. Open Source Softw.*, vol. 8, no. 83, p. 4815, Mar. 2023, doi: 10.21105/joss.04815.

[21] T. Chen and M. Li, 'The Weights can be Harmful: Pareto Search versus Weighted Search in Multi-Objective Search-Based Software Engineering', 2022, doi: 10.48550/ARXIV.2202.03728.

[22] S. Tariq and M. Bassiouni, 'QAMO-SDN: QoS aware Multipath TCP for software defined optical networks', in *2015 12th Annual IEEE Consumer Communications and Networking Conference (CCNC)*, Las Vegas, NV, USA: IEEE, Jan. 2015, pp. 485–491. doi: 10.1109/CCNC.2015.7158023.

[23] S. Sengupta, A. Chowdhary, A. Sabur, A. Alshamrani, D. Huang, and S. Kambhampati, 'A Survey of Moving Target Defenses for Network Security', *IEEE Commun. Surv. Tutor.*, vol. 22, no. 3, pp. 1909–1941, 2020, doi: 10.1109/COMST.2020.2982955.

[24] H. Kim *et al.*, 'Time-Based Moving Target Defense Using Bayesian Attack Graph Analysis', *IEEE Access*, vol. 11, pp. 40511–40524, 2023, doi: 10.1109/ACCESS.2023.3269018.

[25] A. Javadpour, F. Ja'fari, T. Taleb, M. Shojafar, and B. Yang, 'SCEMA: An SDN-Oriented Cost-Effective Edge-Based MTD Approach', *IEEE Trans. Inf. Forensics Secur.*, vol. 18, pp. 667–682, 2023, doi: 10.1109/TIFS.2022.3220939.

[26] S. Yoon, J.-H. Cho, D. S. Kim, T. J. Moore, F. Free-Nelson, and H. Lim, 'Attack Graph-Based Moving Target Defense in Software-Defined Networks', *IEEE Trans. Netw. Serv. Manag.*, vol. 17, no. 3, pp. 1653–1668, Sep. 2020, doi: 10.1109/TNSM.2020.2987085.

[27] S. Chiba, L. Guillen, S. Izumi, T. Abe, and T. Suganuma, 'An SDN-Based Moving Target Defense as a Countermeasure to Prevent Network Scans', *IEICE Trans. Commun.*, vol. E105.B, no. 11, pp. 1400–1407, Nov. 2022, doi: 10.1587/transcom.2021TMP0020.

[28] Ł. Jalowski, M. Zmuda, and M. Rawski, 'A Survey on Moving Target Defense for Networks: A Practical View', *Electronics*, vol. 11, no. 18, p. 2886, Sep. 2022, doi: 10.3390/electronics11182886.

[29] C. Gao, Y. Wang, and X. Xiong, 'A Cyber Deception Defense Method Based on Signal Game to Deal with Network Intrusion', *Secur. Commun. Netw.*, vol. 2022, pp. 1–17, Mar. 2022, doi: 10.1155/2022/3949292.

[30] M. F. Hyder, . Waseemullah, and M. U. Farooq, 'Towards Countering the Insider Reconnaissance Using a Combination of Shuffling and Diversity Moving Target Defense Techniques', *Eng. Technol. Appl. Sci. Res.*, vol. 11, no. 6, pp. 7745–7749, Dec. 2021, doi: 10.48084/etasr.4417.

[31] M. A. Sarwar, M. Hussain, M. U. Anwar, and M. Ahmad, 'FlowJustifier: An Optimized Trust-Based Request Prioritization Approach for Mitigation of SDN Controller DDoS Attacks in the IoT Paradigm', in *ICFNDS '19*, New York, NY, USA: Association for Computing Machinery, 2019. doi: 10.1145/3341325.3342037.

[32] J. Li, S. Nejati, and M. Sabetzadeh, 'Learning Self-adaptations for IoT Networks: A Genetic Programming Approach'. arXiv, May 09, 2022. Accessed: Mar. 16, 2023. [Online]. Available: http://arxiv.org/abs/2205.04352

[33] S. Bera, S. Misra, and M. S. Obaidat, 'Mobi-Flow: Mobility-Aware Adaptive Flow-Rule Placement in Software-Defined Access Network', *IEEE Trans. Mob. Comput.*, vol. 18, no. 8, pp. 1831–1842, Aug. 2019, doi: 10.1109/TMC.2018.2868932.

[34] S. S and S. N, 'HSPC-SDN: Heuristic Driven Self-Configuring Proactive Controller for QoS-Centric Software Defined Network', In Review, preprint, Aug. 2022. doi: 10.21203/rs.3.rs-1520988/v1.

[35] R. S-Julián, I. Lacalle, R. Vaño, F. Boronat, and C. E. Palau, 'Self-* Capabilities of Cloud-Edge Nodes: A Research Review', *Sensors*, vol. 23, no. 6, p. 2931, Mar. 2023, doi: 10.3390/s23062931.

[36] 'Rohr, Matthias & Giesecke, Simon & Hasselbring, Wilhelm. (2006). A Classification Scheme for Self-adaptation Research.'.

[37] S. Scott-Hayward, S. Natarajan, and S. Sezer, 'A Survey of Security in Software Defined Networks', *IEEE Commun. Surv. Tutor.*, vol. 18, no. 1, pp. 623–654, 2016, doi: 10.1109/COMST.2015.2453114.

[38] B. A. A. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, and T. Turletti, 'A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks', *IEEE Commun. Surv. Tutor.*, vol. 16, no. 3, pp. 1617–1634, 2014, doi: 10.1109/SURV.2014.012214.00180.

[39] M. Skuta, D. Macko, and K. Jelemenska, 'Automation of Dynamic Power Management in FPGA-Based Energy-Constrained Systems', *IEEE Access*, vol. 8, pp. 165894–165903, 2020, doi: 10.1109/ACCESS.2020.3022955.

[40] D. Paikaray, D. Chhabra, S. Sharma, S. Goswami, S. H K, and Prof. G. Jethava, 'Energy Efficiency Based Load Balancing Optimization Routing

Protocol In 5G Wireless Communication Networks', *Int. J. Commun. Netw. Inf. Secur. IJCNIS*, vol. 14, no. 3, pp. 187–198, Dec. 2022, doi: 10.17762/ijcnis.v14i3.5605.

[41] A. Nauman, Y. A. Qadri, M. Amjad, Y. B. Zikria, M. K. Afzal, and S. W. Kim, 'Multimedia Internet of Things: A Comprehensive Survey', *IEEE Access*, vol. 8, pp. 8202–8250, 2020, doi: 10.1109/ACCESS.2020.2964280.

[42] N. Feamster, J. Rexford, and E. Zegura, 'The road to SDN: an intellectual history of programmable networks', *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 2, pp. 87–98, Apr. 2014, doi: 10.1145/2602204.2602219.

[43] S. Jain *et al.*, 'B4: experience with a globally-deployed software defined wan', *ACM SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, pp. 3–14, Sep. 2013, doi: 10.1145/2534169.2486019.

[44] H. Sedjelmaci, S. M. Senouci, and M. Al-Bahri, 'A lightweight anomaly detection technique for low-resource IoT devices: A game-theoretic methodology', in *2016 IEEE International Conference on Communications (ICC)*, Kuala Lumpur, Malaysia: IEEE, May 2016, pp. 1–6. doi: 10.1109/ICC.2016.7510811.

[45] D. Kreutz, F. M. V. Ramos, P. Esteves Verissimo, C. Esteve Rothenberg, S. Azodolmolky, and S. Uhlig, 'Software-Defined Networking: A Comprehensive Survey', *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, Jan. 2015, doi: 10.1109/JPROC.2014.2371999.

[46] X. Huang, R. Yu, J. Kang, Z. Xia, and Y. Zhang, 'Software Defined Networking for Energy Harvesting Internet of Things', *IEEE Internet Things J.*, vol. 5, no. 3, pp. 1389–1399, Jun. 2018, doi: 10.1109/JIOT.2018.2799936.

[47] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. doi: 10.1007/978-3-642-29044-2.

[48] K. Petersen, S. Vakkalanka, and L. Kuzniarz, 'Guidelines for conducting systematic mapping studies in software engineering: An update', *Inf. Softw. Technol.*, vol. 64, pp. 1–18, Aug. 2015, doi: 10.1016/j.infsof.2015.03.007.

[49] B. Kitchenham and S. Stuart, 'Guidelines for performing Systematic Literature Reviews in Software Engineering'.

Appendix 1: Selected studies

[S1]. B. Wang, Y. Sun, and X. Xu, 'A Scalable and Energy-Efficient Anomaly Detection Scheme in Wireless SDN-Based mMTC Networks for IoT', *IEEE Internet Things J.*, vol. 8, no. 3, pp. 1388–1405, Feb. 2021, doi: 10.1109/JIOT.2020.3011521.

[S2]. E. Hajian, M. R. Khayyambashi, and N. Movahhedinia, 'A Mechanism for Load Balancing Routing and Virtualization Based on SDWSN for IoT Applications', *IEEE Access*, vol. 10, pp. 37457–37476, 2022, doi: 10.1109/ACCESS.2022.3164693.

[S3]. S. O. Aliyu, F. Chen, Y. He, and H. Yang, 'A Game-Theoretic Based QoS-Aware Capacity Management for Real-Time EdgeIoT Applications', in *2017 IEEE International Conference on Software Quality, Reliability and Security (QRS)*, Prague, Czech Republic: IEEE, Jul. 2017, pp. 386–397. doi: 10.1109/QRS.2017.48.

[S4]. M. M. Raikar, M. S M, and M. M. Mulla, 'Software Defined Internet of Things using lightweight protocol', *Procedia Computer Science*, vol. 171, pp. 1409–1418, 2020, doi: 10.1016/j.procs.2020.04.151.

[S5]. T. Theodorou, G. Violettas, P. Valsamas, S. Petridou, and L. Mamatas, 'A Multi-Protocol Software-Defined Networking Solution for the Internet of Things', *IEEE Commun. Mag.*, vol. 57, no. 10, pp. 42–48, Oct. 2019, doi: 10.1109/MCOM.001.1900056.

[S6]. I. Bedhief, M. Kassar, T. Aguili, L. Foschini, and P. Bellavista, 'Self-Adaptive Management of SDN Distributed Controllers for Highly Dynamic IoT Networks', in *2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC)*, Tangier, Morocco: IEEE, Jun. 2019, pp. 2098–2104. doi: 10.1109/IWCMC.2019.8766349.

[S7]. A. Akbar, M. Ibrar, M. A. Jan, A. K. Bashir, and L. Wang, 'SDN-Enabled Adaptive and Reliable Communication in IoT-Fog Environment Using Machine Learning and Multiobjective

Optimization', *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3057–3065, Mar. 2021, doi: 10.1109/JIOT.2020.3038768.

[S8]. H. Narayanankutty, 'Self-Adapting Model-Based SDSec For IoT Networks Using Machine Learning', in *2021 IEEE 18th International Conference on Software Architecture Companion (ICSA-C)*, Stuttgart, Germany: IEEE, Mar. 2021, pp. 92–93. doi: 10.1109/ICSA-C52384.2021.00023.

[S9]. W. A. Alonazi, H. Hamdi, N. A. Azim, and A. A. A. El-Aziz, 'SDN Architecture for Smart Homes Security with Machine Learning and Deep Learning', *IJACSA*, vol. 13, no. 10, 2022, doi: 10.14569/IJACSA.2022.01310108.

[S10]. R. M. A. Ujjan, Z. Pervez, K. Dahal, A. K. Bashir, R. Mumtaz, and J. González, 'Towards sFlow and adaptive polling sampling for deep learning based DDoS detection in SDN', *Future Generation Computer Systems*, vol. 111, pp. 763–779, Oct. 2020, doi: 10.1016/j.future.2019.10.015.

[S11]. A. Molina Zarca, M. Bagaa, J. Bernal Bernabe, T. Taleb, and A. F. Skarmeta, 'Semantic-Aware Security Orchestration in SDN/NFV-Enabled IoT Systems', *Sensors*, vol. 20, no. 13, p. 3622, Jun. 2020, doi: 10.3390/s20133622.

[S12]. S. Y. Shin, S. Nejati, M. Sabetzadeh, L. C. Briand, C. Arora, and F. Zimmer, 'Dynamic adaptation of software-defined networks for IoT systems: a search-based approach', in *Proceedings of the IEEE/ACM 15th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, Seoul Republic of Korea: ACM, Jun. 2020, pp. 137–148. doi: 10.1145/3387939.3391603.

[S13]. Zabeehullah, F. Arif, and Y. Abbas, 'DLIRS: Deep Learning based Intelligent Routing in Software Defined IoT', in *2022 19th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*, Islamabad, Pakistan: IEEE, Aug. 2022, pp. 237–242. doi: 10.1109/IBCAST54850.2022.9990473.

[S14]. S. S. Bhunia and M. Gurusamy, 'Dynamic attack detection and mitigation in IoT using SDN', in *2017 27th International*

Telecommunication Networks and Applications Conference (ITNAC), Melbourne, VIC: IEEE, Nov. 2017, pp. 1–6. doi: 10.1109/ATNAC.2017.8215418.

[S15]. N. Saha, S. Misra, and S. Bera, 'QoS-Aware Adaptive Flow-Rule Aggregation in Software-Defined IoT', in *2018 IEEE Global Communications Conference (GLOBECOM)*, Abu Dhabi, United Arab Emirates: IEEE, Dec. 2018, pp. 206–212. doi: 10.1109/GLOCOM.2018.8647471.

[S16]. W. Liu, M. Ge, and D. S. Kim, 'Integrated Proactive Defense for Software Defined Internet of Things under Multi-Target Attacks', in *2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID)*, Melbourne, Australia: IEEE, May 2020, pp. 767–774. doi: 10.1109/CCGrid49817.2020.00-12.

[S17]. M. A. Sarwar, M. Hussain, M. U. Anwar, and M. Ahmad, 'FlowJustifier: An optimized trust-based request prioritization approach for mitigation of SDN controller DDoS attacks in the IoT paradigm', in *Proceedings of the 3rd International Conference on Future Networks and Distributed Systems*, Paris France: ACM, Jul. 2019, pp. 1–9. doi: 10.1145/3341325.3342037.

[S18]. J. Li, S. Nejati, and M. Sabetzadeh, 'Learning Self-adaptations for IoT Networks: A Genetic Programming Approach', 2022, doi: 10.48550/ARXIV.2205.04352.

[S19]. P. Tam, S. Math, A. Lee, and S. Kim, 'Multi-Agent Deep Q-Networks for Efficient Edge Federated Learning Communications in Software-Defined IoT', *Computers, Materials & Continua*, vol. 71, no. 2, pp. 3319–3335, 2022, doi: 10.32604/cmc.2022.023215.

[S20]. S. Bera, S. Misra, and M. S. Obaidat, 'Mobility-Aware Flow-Table Implementation in Software-Defined IoT', in *2016 IEEE Global Communications Conference (GLOBECOM)*, Washington, DC, USA: IEEE, Dec. 2016, pp. 1–6. doi: 10.1109/GLOCOM.2016.7841995.

[S21]. F. Kandah, I. Ozcelik, and B. Huber, 'MARS: Machine learning based AdapTable and Robust Network Management for Software-defined Networks', in *2020 10th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, NV, USA: IEEE, Jan. 2020, pp. 0586–0591. doi: 10.1109/CCWC47524.2020.9031241.

[S22]. S. Sharathkumar and N. Sreenath, 'HSPC-SDN: Heuristic Driven Self-Configuring Proactive Controller for QoS-Centric Software Defined Network', *IJCDS*, vol. 13, no. 1, pp. 203–222, Jan. 2023, doi: 10.12785/ijcds/130117.

[S23]. Aslam, M. et al. (2022) Adaptive machine learning based distributed denial-of-services attacks detection and mitigation system for SDN-enabled IOT, MDPI. Available at: https://www.mdpi.com/1424-8220/22/7/2697 (Accessed: 30 August 2023).

[S24]. F. Naeem, G. Srivastava, and M. Tariq, 'A Software Defined Network Based Fuzzy Normalized Neural Adaptive Multipath Congestion Control for the Internet of Things', *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 4, pp. 2155–2164, Oct. 2020, doi: 10.1109/TNSE.2020.2991106.

[S25]. B. Wang, Y. Sun, and X. Xu, 'A Scalable and Energy-Efficient Anomaly Detection Scheme in Wireless SDN-Based mMTC Networks for IoT', *IEEE Internet Things J.*, vol. 8, no. 3, pp. 1388–1405, Feb. 2021, doi: 10.1109/JIOT.2020.3011521.

[S26]. Z. Min, H. Sun, S. Bao, A. S. Gokhale, and S. S. Gokhale, 'A Self-Adaptive Load Balancing Approach for Software-Defined Networks in IoT', in *2021 IEEE International Conference on Autonomic Computing and Self-Organizing Systems (ACSOS)*, Washington, DC, USA: IEEE, Sep. 2021, pp. 11–20. doi: 10.1109/ACSOS52086.2021.00034.

[S27]. S. Chattopadhyay, S. Chatterjee, S. Nandi, and S. Chakraborty, 'Aloe: Fault-Tolerant Network Management and Orchestration Framework for IoT Applications', *IEEE Trans. Netw.*

Serv. Manage., vol. 17, no. 4, pp. 2396–2409, Dec. 2020, doi: 10.1109/TNSM.2020.3008426.

[S28]. N. Hu, F. Luan, X. Tian, and C. Wu, 'A Novel SDN-Based Application-Awareness Mechanism by Using Deep Learning', *IEEE Access*, vol. 8, pp. 160921–160930, 2020, doi: 10.1109/ACCESS.2020.3021185.

[S29]. F. Rahman, Md. S. Satu, Md. Ashaduzzaman, Md. I. Khan, and S. Roy, 'A Power-efficient Framework for Software-defined IoT Ecosystem using Machine Learning', in *2020 2nd International Conference on Sustainable Technologies for Industry 4.0 (STI)*, Dhaka, Bangladesh: IEEE, Dec. 2020, pp. 1–8. doi: 10.1109/STI50764.2020.9350443.

[S30]. S. Salehi, H. Farbeh, and A. Rokhsari, 'An adaptive data coding scheme for energy consumption reduction in SDN-based Internet of Things', *Computer Networks*, vol. 221, p. 109528, Feb. 2023, doi: 10.1016/j.comnet.2022.109528.

[S31]. G.-C. Deng and K. Wang, 'An Application-aware QoS Routing Algorithm for SDN-based IoT Networking', in *2018 IEEE Symposium on Computers and Communications (ISCC)*, Natal: IEEE, Jun. 2018, pp. 00186–00191. doi: 10.1109/ISCC.2018.8538551.

[S32]. T. G. Nguyen, T. V. Phan, D. T. Hoang, H. H. Nguyen, and D. T. Le, 'DeepPlace: Deep reinforcement learning for adaptive flow rule placement in Software-Defined IoT Networks', *Computer Communications*, vol. 181, pp. 156–163, Jan. 2022, doi: 10.1016/j.comcom.2021.10.006.