

A Synopsis on
Design and Development of Context-Aware Offloading
Framework for IoT



JANUARY-2020

Submitted for registration in the degree of
Doctor of Philosophy

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
CHITKARA UNIVERSITY
HIMACHAL PRADESH

Submitted by

Karan Bajaj

PHDENG17049

UNDER THE JOINT SUPERVISION OF

Supervisor

Dr Bhisham Sharma
Associate Professor
Department of Computer Science & Engineering
Chitkara University, Himachal Pradesh

Co-Supervisor

Dr Raman Singh
Assistant Professor
Computer Science & Engineering Department
Thapar Institute of Engineering & Technology
(Deemed to be University), Patiala

TABLE OF CONTENT

Sr. No.	Page No.
<i>List of Figures</i>	<i>ii</i>
<i>List of Tables</i>	<i>iii</i>
<i>List of Abbreviations</i>	<i>iv</i>
1. Introduction.....	1
1.1. Architecture of Internet of Things.....	2
1.1.1 Perception and Sensing Layer.....	3
1.1.2 Transport Layer.....	3
1.1.3 Processing Layer.....	3
1.1.4 Storage Layer.....	4
1.1.5 Application Layer.....	4
1.2. Criteria used in Offloading.....	5
1.2.1. Excessive Computation or Resource Constraint.....	5
1.2.2. To Meet Latency Requirement.....	5
1.2.3. Load Balancing.....	5
1.2.4. Privacy and Security.....	6
2. Literature Review.....	7
3. Justification for Research.....	20
3.1. Motivation.....	20
3.2. Research Gaps.....	21
4. Problem Statement.....	23
4.1. Research Objectives.....	23
4.2. Research Methodology.....	23
4.3. Work Plan.....	26

5. Expected	Outcomes
.....	27
6. References.....	28

List of Figures

Figure1. Basic Architecture of IoT	3
Figure2. Research Methodology	25

List of Tables

Table 1. Work Plan for the Research.....	26
--	----

List of Abbreviations

AAL	Ambient Assisted Living
AJAS	Adaptive Job Allocation Scheduler
ARC	AnyRun Computing
CADA	Context-Aware Decision Algorithm
CE	Cross Entropy
CoSMOS	Context-Sensitive Model for Offloading System
CSOS	Context-Sensitive Offloading System
DEED	Dynamic Energy-Efficient Data Offloading
EES	Energy Efficient Offloading Strategy
EMCO	Evidence-Aware Mobile Computational Offloading
FA	Firefly Algorithm
HAR	Human Activity Recognition
IoT	Internet of Things
IoV	Internet of Vehicles
kNN	k-Nearest Neighbors
MALMOS	Machine Learning-Based Mobile Offloading Scheduler
MDP	Markov Decision Process
MEC	Mobile Edge Computing
NB	Naive Bayes
QoS	Quality-of-Service
SVC	Support Vector Classification
VM	Virtual Machine
WSN	Wireless Sensor Network

1. Introduction

The Internet of Things (IoT) is a type of network connections over the Internet, this network is connected with the objects which communicate with each other through various sensors actuators, and processors communicate with each other to serve a meaningful purpose or to achieve the work that requires a high degree of intelligence with least human intervention [1]. The idea of IoT is not only to bring automation in all sectors of life or to connect all devices but also to make all physical objects intelligent that can connect, communicate with each other and can make the smart decision by themselves.

From the IoT point of view, it is estimated that the total economic impact of IoT will be \$2.7 trillion to \$6.2 trillion per year by 2025, from this the highest impact will be on the health care and manufacturing. After these sectors, the next most influenced areas from IoT will be farming, energy processing, and security. It is calculated that in health-care applications, Internet of Things technology will be having an economic impact of \$1.1 trillion to \$2.5 trillion per year by 2025 [2]. The application domain of IoT is widely spread in all the areas of society and daily life, it serves in all the domains from environmental information, activity information of living organism to the task processing in the industries. In all domains IoT has no existence without Wireless Sensor Network (WSN), it acts as a backbone of IoT, sensors collect the data and communicate them, due to Wide application areas and difference of technology among the devices, it makes the incorporation of IoT with WSN challenging [3].

Based on the study it is found that due to lack of standardization some of the common issues that arise are management of data, communication issues related to a large number of protocols, real-time processing of data, data security, privacy and expansion in existing application termed as scalability [4]. Security is another important aspect in IoT architecture, due to

diversity among the devices and dynamic nature in terms of network and scalability in IoT applications, the existing solutions do not fully satisfy the need of security.

Some applications are delay sensitive due to challenges in the processing of a large amount of data at edge or the cloud level of devices, lead to latency which is not acceptable in some critical applications like health scenarios, transport management etc. Some solutions demand or require high energy, and it's obvious that computation intensive applications require more energy and drain the batteries of devices quickly and lead to expensive solutions [5].

Standardization among architectures and protocols is also an important aspect of IoT architecture due to its lacking, different distributed systems or applications will not be able to communicate and share information among themselves, which negatively affect the interoperability, which is one of the key features of IoT.

1.1. Architecture of Internet of Things

The architecture serves as the most basic and essential block structure for IoT, and it is important in terms of design choices for functional and non-functional requirements in IoT environments to serve the increasing scale and complexity of IoT, they are intended to provide level of abstraction over physical devices and services and support the interoperability among devices. IoT environments contain a high degree of heterogeneity which is both the software and hardware, among the connected objects. These devices have different functionalities, capabilities, characteristics and Internet protocols which also raise the concern of security issues in IoT [3].

There are several key issues faced by IoT like heterogeneity among devices, device management and dynamic discovery, context awareness, data gathering and management, processing of information, scalability, the huge amount of data generation, data security and

integrity, scalability, privacy and dynamic adaptation. Architecture is building blocks to fulfill all the essential requirements and solve the key problems that IoT faces [6].

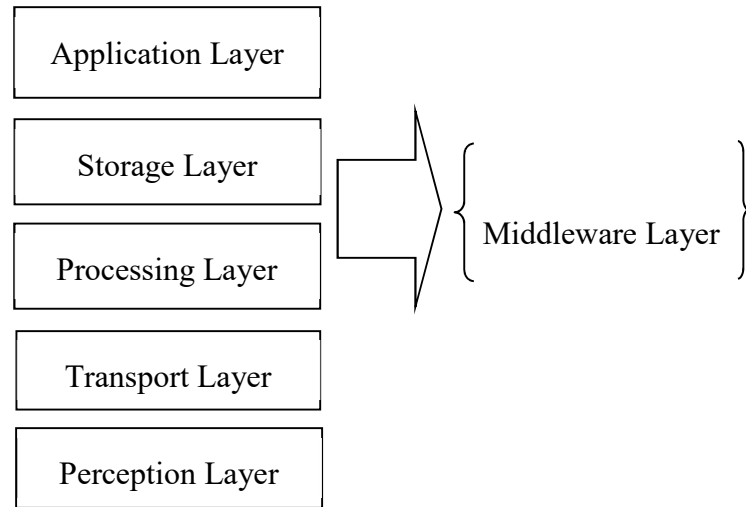


Figure 1. Basic Architecture of IoT

1.1.1. Perception and Sensing Layer

The lowest layer is made up of smart objects integrated with sensors. The sensors enable the interconnection of the physical and digital worlds allowing real-time information to be collected and processed [4].

1.1.2. Transport Layer

This layer is used for transporting data among different devices and objects. In IoT a large amount of data is generated due to a large number of sensors. Therefore the IoT system requires a flexible and high performance network structure that can support different protocols among these devices. As IoT systems provide a large number of services such as high speed transactional services, context-aware applications etc. This layered work with a variety of different technologies and protocols to support communication among the heterogeneous environment [4].

1.1.3. Processing Layer

This layer is also called a middleware layer. A large amount of data coming from the transport layer is analyzed and processed in this layer. This layer uses large numbers of technologies for

analyzing and processing work. Huge databases are used for maintaining the data and edge, femto, fog, and cloud computing are used for processing tasks containing big data [1].

1.1.4. Storage Layer

The temporary storage layer provides storage functionalities such as data replication, distribution, and storage. The request definition component helps to create requests to be sent to the IoT sensors and storage layers.

1.1.5. Application Layer

This layer is the top layer in 3-Layer architecture and provides the application services of the IoT system to the user. Many IoT applications help patients towards their healthy and safe life, engaging more time to spend with their doctors. This service can be in the form of a smart home system, smart city, smart agriculture, or it can be a smart health system [1].

Edge and fog computing and its integration with cloud computing are one of the promising solutions to address many challenges faced by IoT applications and problems related to the services and the limitations of cloud computing. It supports delay-sensitive and context-aware services in the Internet of Things era. Instead of performing data storage and computing in a cluster of clouds, it emphasizes on leveraging the power of local computing and using different types of nearby devices/architectures as edge servers to provide timely and intelligent services.

It can bring many advantages, including highly improved scalability by timely and intelligent service supply and local distributed computing that makes full use of client computing capabilities to meet the requirements of contextual computing. However, to truly realize edge computing in IoT applications, there are still many challenges that need to be addressed, such as how to efficiently distribute and manage data storage and computing, how to make edge and fog computing collaborate with cloud computing for more scalable services, as well as how to secure and preserve the privacy of the whole system.

Computation offloading is a scheme developed to minimize the energy consumption of IoT devices and reduce the latency, also making resource efficient edge/fog computing for intelligent IoT applications. There is need of smart, intelligent and selective offloading scheme to formulate the decision of whether to offload computation, when to offload and where to offload the tasks, like across local devices at edge level, to the fog cloud, or the cloud structure in proximity.

1.2. Criteria used in Offloading

A middleware, such as a smart gateway, monitors the underlying nodes and decides if offloading is needed or not. For making a decision some of the important criteria's that are used for deciding whether or not to offload certain tasks are

1.2.1. Excessive Computation or Resource Constraint

When an application requires computation more than the capability of the native device, certain tasks must be offloaded to a comparative resource rich device [7].

1.2.2. To Meet Latency Requirement

To meet the delay sensitive requirement of certain applications, offloading can be beneficial. The distance can affect delay-sensitive applications. In this case, a node close to the proximity of the receiving node must be involved to offload the task from the distant node and provide the required services with minimum delays [8].

1.2.3. Load Balancing

When one server has reached its capacity of executing tasks, additional tasks will need to be distributed among other servers within the service provider's ecosystem [9]. Similarly, a fog micro, nano data center having multiple servers would distribute tasks to balance the load of the incoming requests.

1.2.4. Privacy and Security

In traditional computing the data that cannot be processed on local devices are offloaded to the cloud, which raises the concern of sensitivity and secrecy of the data or tasks. Offloading at the edge devices, making femto cloud or at the fog cloud may help in securing the privacy and security of data [10]. For example, an enterprise or hospital's data or tasks instead of moving from local machines to a private cloud can now be offloaded to the nearby devices and edge servers. Similarly, personal data from a smartphone may be moved to a personal mobile edge cloud.

2. Literature Review

A systematic review of literature is carried out starting from the basic understanding of context, its role in IoT, how context awareness can help in offloading tasks decision and role of machine learning and deep learning that can aid in recognition of context and taking an offloading decision based on it.

Cuervo et al. [2010]: A code offloading system is proposed, making smartphones last longer with code offload, the system enables fine grained energy-aware offload of mobile code to the infrastructure. It uses the benefits of managed code to reduce the burden on programmers to deal with program partitioning while maximizing the energy benefits of offloading code. Authors present how system partitions programs, how it profiled them, and how it formulated and solved program partitioning as a 0-1 integer linear programming problem. It decides at runtime which methods should be remotely executed, driven by an optimization engine that achieves the best energy savings possible under the mobile device's current connectivity constraints [11].

Kosta et al. [2012]: ThinkAir, a framework that makes it simple for developers to migrate their smartphone applications to the cloud is proposed. ThinkAir exploits the concept of smartphone virtualization in the cloud and provides method-level computation offloading. It focuses on the elasticity and scalability of the cloud and enhances the power of mobile cloud computing by parallelizing method execution using multiple Virtual Machine (VM) images. ThinkAir is evaluated with a range of benchmarks starting from simple micro-benchmarks to more complex applications. It shows that the execution time and energy consumption decrease two orders of magnitude for an N-queens puzzle application and one order of magnitude for face detection and a virus scan application. [12].

Lin et al. [2013]: A decision engine of mobile cloud offloading systems, which decides whether to offload a given method to the cloud servers, is proposed. Authors implement, and evaluate a Context-Aware Decision Algorithm (CADA), to optimize the performance of the mobile devices with various optimization criteria, including short response time and low energy consumption. The CADA algorithm can work with various mobile cloud offloading systems. The evaluation results in the importance of a context-aware decision engine presenting the high prediction accuracy of the CADA algorithm, the performance improvement in both response time and energy consumption [13].

Yürür et al. [2014]: A study to create innovative context-aware applications for recognizing user related social and cognitive activities in any situation and at any location using the increasing computational power of smart phones is proposed. The key idea behind context-aware applications is to encourage users to collect, analyze and share local sensory knowledge in the purpose of a large scale community use by creating a smart network. Many open challenges remain, which are mostly arisen due to the middleware services provided in mobile devices have limited resources in terms of power, memory and bandwidth [14]. Researchers have focused on implementing computationally pervasive systems to create high-level conceptual models to infer activities, and low-level sensory models to extract context from unknown activity patterns.

Energy efficiency is a major restriction imposed on context-aware application developments since the extraction and inference of user relevant sensory data requires continuous sensor operations. A middleware for context-awareness supports the application development task by enhancing the level of abstraction and providing services in dealing with context. The middleware provides basic functionalities such as sensory data acquisition, processing and context recognition [15].

Sukode et al. [2015]: Context has been defined by many researchers whereas the definition of context is given as “Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves.” Here a study is carried out on various approaches related to context-aware systems and self-learning techniques in IoT, also focused on the need for different self-learning techniques to find the openness of IoT environment. Context-aware applications look at the who’s, where’s, when’s and what’s of entities and use this information to determine why the situation is occurring. There are several applications as health care, pervasive games, smartphones, proximate selection, automatic contextual reconfiguration, contextual information and commands, context-triggered actions that require determination of why a situation is occurring [16, 17].

Othman et al. [2015]: Here, authors present a mobile cloud application development model, named MobiByte, to enhance mobile device applications performance, energy efficiency, and execution support. MobiByte is a context-aware application model that uses multiple data offloading techniques to support a wide range of applications. MobiByte is able to offload communications to the cloud and reduce communication overhead on the smartphone [18].

Eom et al. [2015]: Machine Learning-Based Mobile Offloading Scheduler (MALMOS), a novel framework for mobile offloading scheduling based on online machine learning techniques is proposed. MALMOS provides an online training mechanism for the machine learning-based runtime scheduler such that it supports a flexible policy that dynamically adapts scheduling decisions based on the observation of previous offloading decisions and their correctness. To demonstrate its practical applicability, MALMOS integrated with an existing Java-based, offloading-capable code recapturing framework. Quantitative experiments to evaluate the performance and cost for three machine learning algorithms: instance-based

learning, perception, and naive Bayes, with respect to classifier training time, classification time, and scheduling accuracy is performed [19].

Majeed et al. [2016]: To achieve computational efficiency in terms of speed, programming codes that require intensive computational resources can be offloaded to the cloud servers. However, the accuracy of the decision to offload code to cloud server can largely impact the performance of the overall system. Authors propose an accurate decision making system for adaptive and dynamic nature of mobile systems by using support vector machine learning technique for making offloading decision locally or remotely. The proposed system is evaluated with android-based prototype component for experiments considering different internal and external conditions. The proposed system achieves approximately 92% accuracy, leading to the accurate decision, thus improving performance and reducing energy consumption [20].

Ferrari et al. [2016]: The natural candidate for computational offloading is the cloud, but recent results point out the hidden costs of cloud in terms of latency and energy. Strategies that rely on local computing power have been proposed that enable fine-grained energy-aware code offloading from a mobile device to a nearby piece of infrastructure. Authors propose AnyRun Computing (ARC), a system to dynamically select the adequate piece of local computing infrastructure. With ARC, code can run anywhere and be offloaded not only to nearby dedicated devices but also to peer devices [21].

Sezer et al. [2017]: IoT deployments are increasing with accelerating speed, as the field grows in numbers and heterogeneity, one of the major problems in the path to intelligent IoT is understanding “context”, or making sense of the environment, situation, or status using data from sensors, and then acting accordingly in autonomous ways. This is called “context aware computing”, and it requires both sensing and, increasingly learning. Big data is a study area

where the corresponding methodologies for processing large data sets, which cannot be processed through traditional data processing techniques, are examined [22]. IoT converges to Big data to analyze data and make inferences from collected datasets. Machine learning algorithms provide better predictions and decisions with more available data collected from different sources. In addition to machine learning, data mining also supports better predictions and decisions by using appropriate learning algorithms.

Many learning systems and solutions for IoT have been developed with context awareness features. They are mostly designed as rule-based, logic-based, and ontology based solutions, and developed using supervised, unsupervised, and reinforcement algorithms. They can be improved with mixed or hybrid methods, such as rule and ensemble learning algorithms for better performance. Neural networks can be used more extensively with the rapidly growing IoT big data and solutions can be enhanced with better reasoning capabilities. Deep learning has emerged as a revolutionary technique to provide robust solutions in classification and/or prediction problems where traditional machine learning models are failing [23].

Flores et al. [2017]: Authors design and develop an AutoScaler, an essential component for offloading architecture to handle the offloading workload. Also, an offloading simulator is developed to generate dynamic offloading workload of multiple devices. In offloading, a device outsources the processing of a task to a more powerful machine. To calculate the cost of outsourcing, authors had taken into consideration multiple parameters of the system, e.g., network latency, data transfer size, remote server capabilities, etc. The Autoscaler implements a round-robin scheduler to distribute the load among the available servers [24].

Hossain et al. [2018]: Edge computing approach to minimize latency is used. As a major portion of data is generated from the user endpoint, processing this data in the edge can significantly improve the performance. The experiment shows that processing raw IoT data at

the edge devices is effective in terms of latency and provides situational awareness for the decision-makers of a smart city in a seamless fashion. In cloud-based processing, all the sensor data are processed through a remote cloud server, but it is costly in terms of processing and storage to send all data to the cloud. In this aspect, the emerging edge computing can help save a lot of bandwidth and may increase the processing speed.

Here authors contribute by providing an edge computing framework to process situations in an IoT smart city that would help the decision-makers to be situation aware and provide relevant services to city residents. The primary purpose of the proposed framework is to detect the current situation that represents the state of transportation, healthcare, security and other aspects in a smart city. Efficient situation understanding allows the city officials to provide services to the citizens and perform actions accordingly. A prototype implementation of the framework is developed using JDK version 8 and Laravel Framework 5.6. To test the performance, two well known open IoT dataset (CityPulse [25] and City of Chicago [26]) is used. Study shows that processing raw IoT data at the edge devices is effective in terms of latency and provides situational awareness for the decision makers [27].

Flores et al. [2018]: Computational offloading can improve the user experience of mobile applications through improved responsiveness and reduced energy footprint. A fundamental challenge in offloading is to distinguish situations where offloading is beneficial from those where it is not. Crowdsensed evidence traces as a novel mechanism for improving the performance of the offloading system is proposed. Evidence-Aware Mobile Computational Offloading (EMCO) toolkit and platform is designed as a novel solution for computational offloading. EMCO provides better scalability than current cloud platforms, being able to serve a larger number of clients without variations in performance. By outsourcing parts of computationally intensive tasks to dedicated computing infrastructure, devices can reduce the

burden on their resources while benefiting from resources provided by the dedicated infrastructure.

The key novelty in EMCO is the use of crowdsensed evidence traces to characterize the influence of different contextual parameters and other factors on offloading decisions. EMCO models the context where offloading decisions are made is through simple dimensions that are easy to scale and determines optimal dimensions using an analytic process that characterizes the performance of offloading based on contexts captured by the community. Two crowdsensing datasets Carat [28] and NetRadar [29], to demonstrate that it is possible to use crowdsensing to quantify the effect of different factors on offloading performance accurately.

EMCO uses scalable provisioning as a service approach where EMCO is deployed on multiple interconnected servers and responds on-demand to computational requests from any available server. The constructed classifier can then be deployed on the client to make offloading decisions directly without interfacing with the cloud. EMCO provides better scalability than current cloud platforms, being able to serve a larger number of clients without variations in performance [30].

Aazam et al. [2018]: For many of the applications, there is a need of another entity to execute tasks on behalf of the user's device and return the results, this technique often called offloading, where tasks are outsourced. It can occur between IoT nodes, sensors, edge devices, or fog nodes. The large distance between the cloud and end-user devices, it becomes a challenge to support real-time processing and deliver fast response times to end-users. To address this challenge, some middleware is required between the end nodes and the cloud. Various middleware technologies such as mobile edge computing, cloudlets, have been proposed and discussed [31], authors present the various criteria used by recently proposed middleware technologies when tasks are offloaded in the fog computing environment.

In the offloading scenario, a node close to the proximity of the receiving node must be involved to offload the task from the distant node and provide the required services with minimum delays to meet the delay sensitive requirements of applications. Offloading of tasks needs an intelligent system that can make optimal decisions about whether to offload based on the energy tradeoff's and which specific task to be offloaded to the cloud, or to a local fog or femto-cloud. Further, if the task is divided into a micro task then the number of devices to which task allocation will be performed to balance the load [32].

Aazam et al. [2018]: Offloading tasks to co-located edge nodes such as fog, or a femto-cloud is one viable solution to address the issues such as performing compute-intensive tasks, and managing energy consumption. Offloading to a cloud or a fog consumes a different amount of energy and results in different gains in terms of computation. Three machine learning techniques widely used in Human Activity Recognition (HAR) tasks: k-Nearest Neighbors (kNN), Naive Bayes (NB), and Support Vector Classification (SVC) [33] are used. As a benchmark, the publicly available dataset PAMAP2 [34] is used. Communication overheads increase with task offloading and require additional energy consumption. Deciding the number of devices on which the tasks are to be offloaded is an important criterion to be considered before offloading, in proposed work authors has taken one to four devices where one means no offloading and the master device will itself do the computation. The author has used all possibility for allotment of tasks on devices from one to four devices and no intelligent method is there to decide the number of devices where tasks should be offloaded [35].

Nakahara et al. [2018]: Context-Sensitive Model for Offloading System (CoSMOS) a context-aware and self-adaptive offloading decision support model for mobile cloud computing systems. It employs decision-taking estimation based on the application's time execution and energy consumption to decide efficiently when and which application components should be offloaded in order to improve the system's execution. The main objective proposed is to

employ self-adaptive system architecture to reinforce the model's self-adaptive and context-aware capabilities on a mobile cloud computing system to improve cloud support performance. To evaluate the CoSMOS model this work is applied into two mobile applications: an implementation of the N-Queen [36] problem and the BenchImage [37], an image processing application, to emphasize the impact of the estimation model on the application performance. CoSMOS can adjust offloading services according to more adaptive conditions, even if the network conditions are not optimal [38].

Nalepa et al. [2019]: An integration of context-aware systems and affective computing paradigms are done to identify and characterize affective context data, and provide knowledge-based models to identify and interpret effects based on this data. For detection of effective phases use of wearable and mobile devices are made. A data acquisition layer is proposed based on wearable devices able to gather physiological data and integrate it with the mobile context-aware framework. It uses physiological measurements provided by wearable devices and a custom context-aware framework [39].

Kim et al. [2019]: For the computing offloading of IoT devices, diverse job allocation techniques considering performance resources have been studied. An Adaptive Job Allocation Scheduler (AJAS) that adaptively redistributes the jobs allocated to IoT devices based on user behaviour patterns is proposed. The AJAS allocates jobs using the dynamic performance resources, which are idle resources and battery consumption rates of diverse IoT devices. Also, the AJAS measures the battery consumption rate of user applications executed in the IoT device to assess whether the allocated jobs can be processed.

AJAS identifies IoT devices that cannot process jobs and minimizes states in which allocated jobs cannot be processed due to battery exhaustion and delay time due to job reallocation. The

AJAS minimized job reallocation delay time by requesting the allocation of jobs to other nodes when the user's applications are executed while allocated jobs are being processed [40].

Yan et al. [2019]: Data offloading has become main area in both industry and academia, especially for real-time applications. Based on this, authors propose the task reliability model, the energy consumption model, and the device reliability model. From the perspective of optimising energy consumption, the authors proposed an optimal task scheduling model. An innovative Dynamic Energy-Efficient Data (DEED) offloading scheduling algorithm is proposed. The purpose of DEED is to as much as possible reducing the energy consumption while ensuring the task reliability. Authors proposed a heuristic algorithm that can reduce energy consumption while ensuring task reliability. For data offloading, two kinds of offloading methods: end-to-cloud data offloading and end-to-end collaborative data offloading are proposed. Authors propose a dynamic energy-efficient data offloading scheduling algorithm DEED, which can effectively deal with the problem of collaborative data offloading under unstable channel conditions [41].

Adhikari and Gianey [2019]: A cloud data centre consumes a large amount of energy while transmitting and computing the IoT applications, which lead to a high carbon footprint. On the other hand, the fog nodes provide various cloud services at the edge of the network, which can run the IoT applications locally with minimum energy consumption and delay. Due to the limited resource capacity, the fog nodes are not suitable for processing the resource-intensive IoT applications. To address these challenges, sustainable infrastructure in a fog cloud environment for processing delay-intensive and resource-intensive applications with an optimal task offloading strategy is built. The proposed offloading strategy uses Firefly Algorithm (FA) [42] for finding an optimal computing device based on two Quality-of-Service (QoS) parameters such as energy consumption and computational time.

The objective of this strategy is to minimize the computational time and the energy consumption of the IoT applications with minimum delay. Authors have developed new bi-objective Energy Efficient Offloading Strategy (EES) based on firefly technique. The major contribution of the proposed strategy is to find an optimal computational device for each IoT applications. However, the existing multi-objective scheduling strategies deploy the tasks to suitable fog devices without considering the multiple objectives of the IoT applications or tasks [43].

Benedetto et al. [2019]: MobiCOP-IoT allows developers to deploy surrogates on both distant clouds and proximate nodes located on edge. As such, MobiCOP-IoT enabled applications can take advantage of the benefits of mobile edge computing to further enhance their capabilities. MobiCOPIoT offers various features like automatic offloading of arbitrary tasks according to the output of an integrated decision-making engine, a reliable communication model based on an asynchronous communication mechanism, a server component capable of automatic horizontal scaling, and it offers a solid security layer that leverages the robustness of google play services and the security features of the android operating system.

Having rich features but still, the MobiCOPIoT required to manually set whether the platform should operate in cloud or edge mode. It requires specifying the IP of the server manually, automatic server node discovery is currently not supported also MobiCOP-IoT's decision-making engine currently does not take into consideration energy profiles [44].

Chen et al. [2019]: An adaptive framework that supports mobile applications with offloading capabilities in Mobile Edge Computing (MEC) is proposed. A new design pattern to enable an application to be dynamically offloaded among mobile devices, mobile edges, and the cloud, an estimation model, is designed to determine the offloading scheme automatically. In this model, different parts of the application may be executed on different computation nodes.

Finally, an adaptive offloading framework is implemented to support the design pattern and the estimation model. The framework first refactors the application to implement a special design pattern and then analyses the static code of the application to obtain information on movable classes. Thus, it can determine the offloading scheme during runtime according to the device context and enable applications to be offloaded between the device, mobile edges, and the cloud dynamically [45].

Zhao et al. [2019]: Content based offloading mechanism for Mobile Edge Computing system is proposed, the contents are divided into different categories according to their data transmission rate and users with lower data transmission rate contents are considered to have a lower priority, which will be offloaded first. According to the prediction value and user priority, and offloading algorithm based on Cross Entropy method (CE method) is proposed to improve the system performance. The offloading strategy based on the CE method is applied to maximize the system throughput while guaranteeing the data transmission requirements of each user [46].

Alam et al. [2019]: A near-end network solution of computation offloading in mobile edge/fog is proposed. For handling the computation resource demand from the massive mobile devices, a deep Q-learning based autonomic management framework is proposed. The distributed Fog Network Controller (FNC) scavenging the available edge/fog resources, i.e. processing, memory, network to enable edge/fog computation service. The randomness in the availability of resources and numerous options for allocating those resources for offloading computation fits the problem appropriate for modelling through Markov Decision Process (MDP) and solution through reinforcement learning. The proposed model is simulated through MATLAB. A code offloading framework is proposed for computation offloading in mobile fog environment. The offloading method deploys basic blocks, incompatible fog nodes to support parallelism [47].

Junior et al. [2019]: Proposes a Context-Sensitive Offloading System (CSOS) that takes advantage of the main machine-learning reasoning techniques and robust profiling system to provide offloading decisions with high levels of accuracy. Network bandwidth, received signal strength, input data size, and surrogate capabilities, amongst others, play a critical role in deciding whether or not to offload a task. This system implements and evaluates multiple classification algorithms to seek the best offloading decision-making solution that achieves the highest accuracy.

CSOS integrates middleware, machine learning classification algorithms, and a robust profiling system. A decision engine is designed that is highly accurate for the adaptive and dynamic nature of mobile environments by using main classifications algorithms (k-nearest neighbours, rules, decision tree, and Naive Bayes) for the decision-making on whether the computation should be offloaded or not. CSOS adopts a history-based prediction approach where we utilize the past profiled information as a basis for performance inference for future tasks. CSOS follows the standard client–server model [48].

3. Justification for Research

In this section, the motivation behind the work and research gaps which laid the foundation for the problem formulation is discussed.

3.1. Motivation

The main idea of IoT is to bring automation in all sectors of life or to connect all the daily used devices and make all physical objects intelligent that can connect, communicate with each other and can make a smart decision by themselves. From the study, it is estimated that in the future highest impact of IoT will be on healthcare, manufacture, agriculture, energy processing and security [2, 49].

The application domain of IoT is widely spread in all the areas of society and daily life, it serves in all the domains from environmental information, activity information of living organism to the task processing in the industries. IoT is connected with all the domains of society. Therefore, a large amount of data is generated from the device sources, quantifying and characterizing such huge amount of information on a single device is infeasible due to inherent complexity and diversity among the data [30].

When an application requires computation more than the capability of the native device, and to meet the latency requirement of certain delay sensitive applications like healthcare, disaster management etc., certain tasks must be offloaded to a comparatively resource rich device and devices in close proximity to provide the service with minimum delays. In addition to this, when the server reaches the maximum capacity of executing the tasks, additional tasks are required to be distributed in the form of offloading termed as load balancing [7- 9].

Computation offloading is a mechanism developed to minimize the energy consumption, meet the latency requirement of IoT devices. To make resource efficient edge/fog computing for intelligent IoT applications, there is need of smart and selective offloading framework to

formulate the decision of whether to offload computation and where to offload the tasks, across local devices at edge level, fog cloud, or to the cloud structure.

3.2. Research Gaps

There are several challenges in IoT from heterogeneity among the devices, their data to device management and context awareness and processing of information [3]. Offloading of data for IoT applications and devices is not as straightforward as it seems to be, there are several further challenges that are raised with this fusion of technologies. Computing offloading is regarded to be very important to overcome limited computing power and storage capacity and the limitations of built-in batteries [40, 41].

- 1) Learning from the context of data for computational and data offloading is one of the major problems in the path to intelligent IoT, understanding the context, situation, or status and then acting accordingly in autonomous ways [48].
- 2) There are mostly solutions that are designed as rule-based, logic-based, and ontology based using supervised, unsupervised, and reinforcement algorithms. They can be improved with hybrid methods, such as learning algorithms like neural methods and deep learning for better performance [23].
- 3) There is a need for an intelligent system that can make optimal decisions about which specific tasks to be offloaded to the cloud, or femto-cloud. Designing the middleware which can learn from the sensory data, battery behaviour and context inferences through machine learning and processing of the data is the challenge faced. Middleware services provided in the devices have limited resources in terms of power, memory and bandwidth [14, 15, 35].
- 4) Many IoT applications like smart healthcare, Ambient Assisted Living (AAL), virtual reality, augmented reality, intelligent vehicular communication, Tactile Internet, Internet of Vehicles (IoV) etc. require separate entity to execute the tasks on behalf of

the user's device and return the results. The large distance between the cloud and end-user devices, become a challenge to support real-time processing and deliver fast response times to end-users [31, 32].

- 5) Offloading the IoT applications to the cloud consumes a large amount of energy during transmission and computing, because of far-end network and experiences a higher latency and network delay. On the other hand, the fog nodes provide various cloud services at the edge of the network, which can run the IoT applications locally with minimum energy consumption and delay. Due to the limited resource capacity, the fog nodes are not suitable for processing the resource-intensive IoT applications [43, 47].
- 6) Current studies in the field of offloading are more around data centralization and coordination. Edge computing and fog computing are a new area of research and in needs to establish framework to put these concepts into practice [44].
- 7) A fundamental challenge in offloading is to distinguish situations where offloading is beneficial from those where it is counterproductive. For making practical and effective optimal offloading decisions, a wide range of factors can influence the effectiveness of offloading [30].
- 8) Key research questions for offloading computation in fog or edge structures are how to offload computation, which module or process of applications should offload, where to offload the module or process for minimizing the latency of service computing [47].

4. Problem Statement

Computation offloading is a mechanism developed to minimize the energy consumption of devices and reduce the time, a fundamental challenge in offloading is to distinguish situations where offloading is beneficial. Learning from the context of data or understanding the context or situation for computational and data offloading is one of the major problems. For making effective optimal offloading decisions, there is a need for design and development of a context-aware offloading framework for computational intensive IoT applications.

4.1. Research Objectives

Several research gaps have been identified in the previous section. However, the focus of the proposed work will be limited towards the development of an offloading framework for IoT applications and to achieve this, the following objectives need to be fulfilled:

- 1) To study the existing offloading frameworks in the Internet of Things.
- 2) To implement the existing offloading frameworks.
- 3) To propose a new offloading framework using context-aware learning and analyze its computational adaptability.
- 4) To compare the proposed offloading framework with existing frameworks based on various parameters.

4.2. Research Methodology

The methodology of the proposed work is primarily divided into a literature review followed by the development of the proposed framework and comparing it with the existing framework.

- To study existing frameworks and identifies the challenges in them, study solutions available so far and identifying the parameters to make a decision engine for making an offloading decision.

- Identification of tools and technologies towards the formation of a solution. Some of the potential simulators to implement required offloading framework are Weka, Matlab, etc. Besides this framework can be developed with java, python.
- Design and development of a proposed framework. Setup a computational model for a decision engine over IoT framework that will enable an application to be offloaded among the devices, using contextual learning.
- Performance evaluation of the developed framework will be done, the results obtained will be compared with existing solutions based on different parameters like energy efficiency, time.

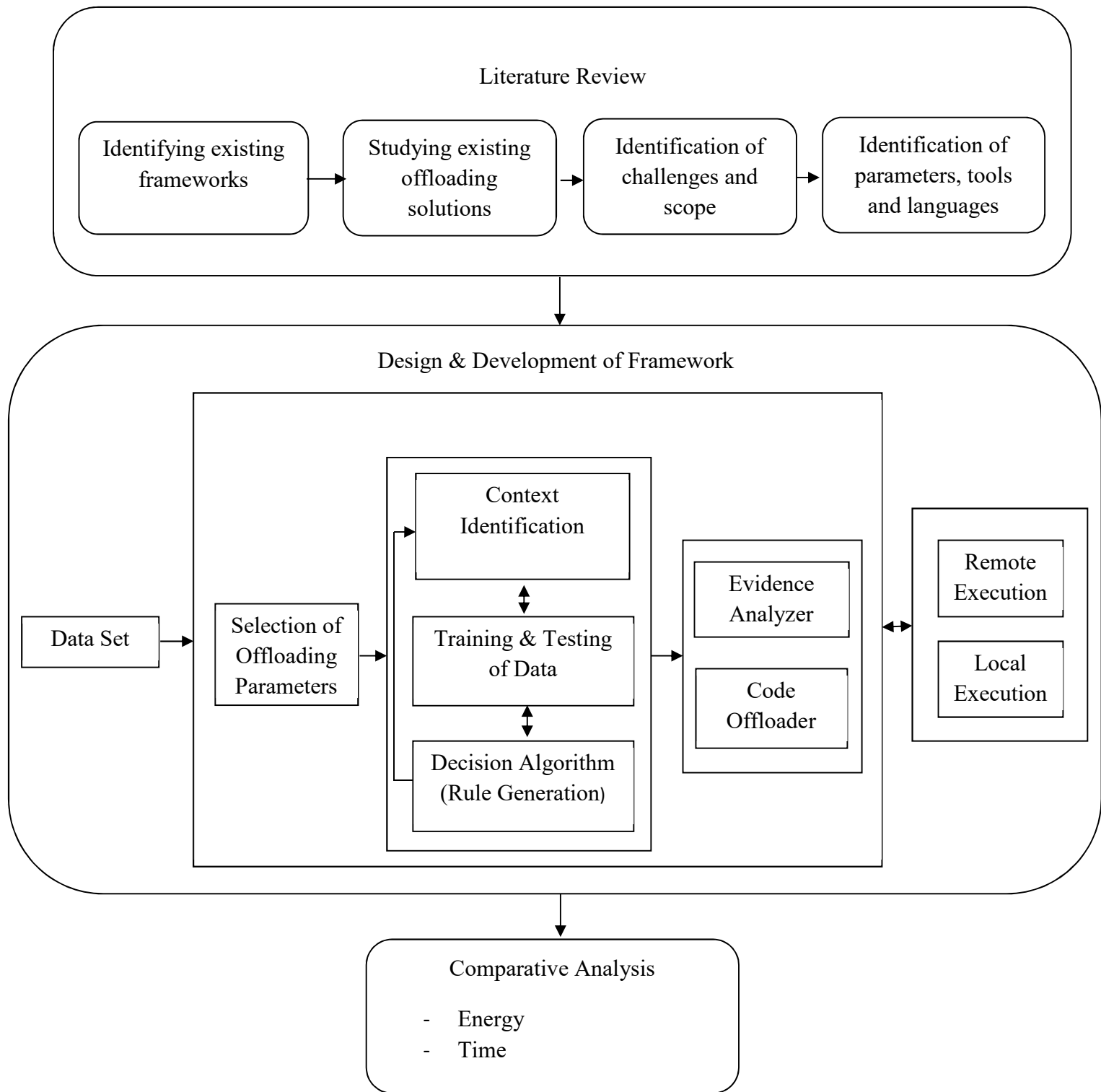




Figure 2. Research Methodology

4.3. Work Plan

Table 1. Work Plan for the Research

Tasks	(January 2018 – December 2018)				(January 2019 – December 2019)				(January 2020 – December 2020)				(January 2021 – December 2021)			
	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
Course work, Submission of Synopsis.																
Identify and study the existing literature of IoT offloading frameworks.																
To study and identify the different data sets and parameters.																
Design & Development of offloading framework.																
Comparison of the proposed framework with existing frameworks in literature.																
Documentation of Thesis.																

	Work Completed		Work In Progress
---	----------------	---	------------------

5. Expected Outcomes

Design and development of an adaptable framework that can make an offloading decision by understanding the context of data. The framework will provide the assumption that will decide that whether offloading will give benefit in the performance of IoT based applications. Decision taking offloading estimation framework will be designed to automatically determine the offloading mechanism according to the context, in which different parts of the applications if required are executed on the edge devices, fog, or to the cloud.

6. References

- [1] Sethi, P. and Sarangi, S.R., 2017. Internet of things: architectures, protocols, and applications. *Journal of Electrical and Computer Engineering*, 2017.vol. 20, pp. 1–25.
- [2] Manyika, J., Chui, M., Bughin, J., Dobbs, R., Bisson, P. and Marrs, A., 2013. *Disruptive technologies: Advances that will transform life, business, and the global economy*. Vol. 180, pp. 17–21. San Francisco, CA: McKinsey Global Institute.
- [3] Giri, A., Dutta, S., Neogy, S., Dahal, K. and Pervez, Z., 2017, October. Internet of things (IoT): a survey on architecture, enabling technologies, applications and challenges. In *Proceedings of the 1st International Conference on Internet of Things and Machine Learning* (pp.1-7). ACM.
- [4] Al-Qaseemi, S. A., Almulhim, H. A., Almulhim, M. F., & Chaudhry, S. R. (2016, December). IoT architecture challenges and issues: Lack of standardization. In *Future Technologies Conference (FTC)* (pp. 731-738). IEEE.
- [5] Li, Y., Björck, F., & Xue, H. (2016, December). Iot architecture enabling dynamic security policies. In *Proceedings of the 4th International Conference on Information and Network Security* (pp. 50-54). ACM.
- [6] Cavalcante, E., Alves, M. P., Batista, T., Delicato, F. C., & Pires, P. F. (2015, May). An analysis of reference architectures for the internet of things. In *Proceedings of the 1st International Workshop on Exploring Component-based Techniques for Constructing Reference Architectures* (pp. 13-16). ACM.
- [7] Elgazar, A., Harras, K., Aazam, M. and Mtibaa, A., 2018, March. Towards intelligent edge storage management: Determining and predicting mobile file popularity. In *6th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)* (pp. 23-28). IEEE.
- [8] Masip-Bruin, X., Marín-Tordera, E., Tashakor, G., Jukan, A. and Ren, G.J., 2016. Foggy clouds and cloudy fogs: a real need for coordinated management of fog-to-cloud computing systems. *IEEE Wireless Communications*, Vol. 23, pp.120-128.
- [9] Huang, C., Lu, R. and Choo, K.K.R., 2017. Vehicular fog computing: architecture, use case, and security and forensic challenges. *IEEE Communications Magazine*, Vol. 55, pp.105-111.

- [10] Aazam, M., Huh, E.N. and St-Hilaire, M., 2018, Towards media inter-cloud standardization—evaluating impact of cloud storage heterogeneity. *Journal of Grid Computing*, 16(3), pp.425-443.
- [11] Cuervo, E., Balasubramanian, A., Cho, D.K., Wolman, A., Saroiu, S., Chandra, R. and Bahl, P., 2010, June. MAUI: making smartphones last longer with code offload. In *Proceedings of the 8th international conference on Mobile systems, applications, and services* (pp. 49-62). ACM.
- [12] Kosta, S., Aucinas, A., Hui, P., Mortier, R. and Zhang, X., 2012, March. Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading. In *2012 Proceedings IEEE Infocom* (pp. 945-953). IEEE.
- [13] Lin, T.Y., Lin, T.A., Hsu, C.H. and King, C.T., 2013, April. Context-aware decision engine for mobile cloud offloading. In *2013 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)* (pp. 111-116). IEEE.
- [14] Wood, A.D., Stankovic, J.A., Virone, G., Selavo, L., He, Z., Cao, Q., Doan, T., Wu, Y., Fang, L. and Stoleru, R., 2008. Context-aware wireless sensor networks for assisted living and residential monitoring. *IEEE network*, 22(4), pp.26-33.
- [15] Yürür, Ö., Liu, C.H., Sheng, Z., Leung, V.C., Moreno, W. and Leung, K.K., 2014. Context-awareness for mobile sensing: A survey and future directions. *IEEE Communications Surveys & Tutorials*, 18(1), pp.68-93.
- [16] Abowd, G.D., Dey, A.K., Brown, P.J., Davies, N., Smith, M. and Steggles, P., 1999, September. Towards a better understanding of context and context-awareness. In *International symposium on handheld and ubiquitous computing* (pp. 304-307). Springer, Berlin, Heidelberg.
- [17] Sukode, S., Gite, S. and Agrawal, H., 2015. Context aware framework in IoT: a survey. *International Journal*, , vol. 4, no. 1, pp. 1–9.
- [18] Othman, M., Khan, A.N., Abid, S.A. and Madani, S.A., 2015, MobiByte: an application development model for mobile cloud computing. *Journal of Grid Computing*, 13(4), pp.605-628.
- [19] Eom, H., Figueiredo, R., Cai, H., Zhang, Y. and Huang, G., 2015, March. Malmos: Machine learning-based mobile offloading scheduler with online training. In *3rd IEEE International Conference on Mobile Cloud Computing, Services, and Engineering* (pp. 51-60). IEEE.

- [20] Majeed, A.A., Khan, A.U.R., UlAmin, R., Muhammad, J. and Ayub, S., 2016, August. Code offloading using support vector machine. In *6th International Conference on Innovative Computing Technology (INTECH)* (pp. 98-103). IEEE.
- [21] Ferrari, A., Giordano, S. and Puccinelli, D., 2016. Reducing your local footprint with anyrun computing. *Computer Communications*, Vol. 81, pp.1-11.
- [22] Zaslavsky, A., Perera, C. and Georgakopoulos, D., 2013. Sensing as a service and big data ‘‘Context aware computing for the Internet of Things: A survey,’’ *IEEE Commun. Surveys Tuts.*, Vol. 16, no. 1, pp. 414–454.
- [23] Sezer, O.B., Dogdu, E. and Ozbayoglu, A.M., 2017. Context-aware computing, learning, and big data in internet of things: a survey. *IEEE Internet of Things Journal*, 5(1), pp.1-27.
- [24] Flores, H., Su, X., Kostakos, V., Ding, A.Y., Nurmi, P., Tarkoma, S., Hui, P. and Li, Y., 2017, March. Large-scale offloading in the Internet of Things. In *IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)* (pp. 479-484).
- [25] Puiu, D., Barnaghi, P., Tönjes, R., Kümper, D., Ali, M.I., Mileo, A., Parreira, J.X., Fischer, M., Kolozali, S., Farajidavar, N. and Gao, F., 2016. Citypulse: Large scale data analytics framework for smart cities. *IEEE Access*, 4, pp.1086-1108.
- [26] C. of Chicago, City of chicago open data, Tech. rep., <https://data.cityofchicago.org/>, last accessed date: 26/12/2019.
- [27] Hossain, S.A., Rahman, M.A. and Hossain, M.A., 2018. Edge computing framework for enabling situation awareness in IoT based smart city. *Journal of Parallel and Distributed Computing*, 122, pp.226-237.
- [28] Oliner, A.J., Iyer, A.P., Stoica, I., Lagerspetz, E. and Tarkoma, S., 2013, November. Carat: Collaborative energy diagnosis for mobile devices. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems* (pp. 1-10). ACM.
- [29] Sonntag, S., Manner, J. and Schulte, L., 2013, May. Netradar-measuring the wireless world. In *2013 11th International Symposium and Workshops on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)* (pp. 29-34). IEEE.
- [30] Flores, H., Hui, P., Nurmi, P., Lagerspetz, E., Tarkoma, S., Manner, J., Kostakos, V., Li, Y. and Su, X., 2018. Evidence-aware mobile computational offloading. *IEEE Transactions on Mobile Computing*, 17(8), pp.1834-1850.

- [31] Jararweh, Y., Doulat, A., AlQudah, O., Ahmed, E., Al-Ayyoub, M. and Benkhelifa, E., 2016, May. The future of mobile cloud computing: Integrating cloudlets and mobile edge computing. In *23rd International conference on telecommunications (ICT)* (pp. 1-5). IEEE.
- [32] Aazam, M., Zeadally, S. and Harras, K.A., 2018. Offloading in fog computing for IoT: Review, enabling technologies, and research opportunities. *Future Generation Computer Systems*, Vol. 87, pp.278-289.
- [33] Lara, O.D. and Labrador, M.A., 2012. A survey on human activity recognition using wearable sensors. *IEEE communications surveys & tutorials*, 15(3), pp.1192-1209.
- [34] Reiss, A. and Stricker, D., 2012, June. Introducing a new benchmarked dataset for activity monitoring. In *16th International Symposium on Wearable Computers* (pp. 108-109). IEEE.
- [35] Aazam, Mohammad, Sherali Zeadally, and Eduardo Feo Flushing., 2018. Task Offloading in Edge Computing and Femto-cloud for Machine Learning-based Healthcare IoT *Computer Networks* Vol. 87, pp. 278-289.
- [36] Gent, I.P., Jefferson, C. and Nightingale, P., 2017. Complexity of n-queens completion. *Journal of Artificial Intelligence Research*, Vol. 59, pp.815-848.
- [37] Costa, P.B., Rego, P.A., Rocha, L.S., Trinta, F.A. and de Souza, J.N., 2015, April. Mpos: A multiplatform offloading system. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing* (pp. 577-584). ACM.
- [38] Nakahara, F.A. and Beder, D.M., 2018. A context-aware and self-adaptive offloading decision support model for mobile cloud computing system. *Journal of Ambient Intelligence and Humanized Computing*, 9(5), pp.1561-1572.
- [39] Nalepa, G.J., Kutt, K. and Bobek, S., 2019. Mobile platform for affective context-aware systems. *Future Generation Computer Systems*, Vol. 92, pp.490-503.
- [40] Kim, H.W., Park, J.H. and Jeong, Y.S., 2019. Adaptive job allocation scheduler based on usage pattern for computing offloading of IoT. *Future Generation Computer Systems*, Vol. 98, pp.18-24.
- [41] Yan, H., Zhang, X., Chen, H., Zhou, Y., Bao, W. and Yang, L.T., 2019. DEED: Dynamic Energy-Efficient Data offloading for IoT applications under unstable channel conditions. *Future Generation Computer Systems*, Vol. 96, pp.425-437.
- [42] Blum, C. and Roli, A., 2003, Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM computing surveys (CSUR)*, 35(3), pp.268-308.

- [43] Adhikari, M. and Gianey, H., 2019. Energy efficient offloading strategy in fog-cloud environment for IoT applications. *Internet of Things*, Vol. 6, pp.34-40.
- [44] Benedetto, J.I., González, L.A., Sanabria, P., Neyem, A. and Navón, J., 2019, Towards a practical framework for code offloading in the Internet of Things. *Future Generation Computer Systems*, Vol. 92, pp.424-437.
- [45] Chen, X., Chen, S., Ma, Y., Liu, B., Zhang, Y. and HUANG, G., 2019, An adaptive offloading framework for Android applications in mobile edge computing. *SCIENCE CHINA Information Sciences*, 62(8), pp.8-21.
- [46] Zhao, X., Yang, K., Chen, Q., Peng, D., Jiang, H., Xu, X. and Shuang, X., 2019. Deep learning based mobile data offloading in mobile edge computing systems. *Future Generation Computer Systems*, 99, pp.346-355.
- [47] Alam, M.G.R., Hassan, M.M., Uddin, M.Z., Almogren, A. and Fortino, G., 2019. Autonomic computation offloading in mobile edge for IoT applications. *Future Generation Computer Systems*, 90, pp.149-157.
- [48] Junior, W., Oliveira, E., Santos, A. and Dias, K., 2019. A context-sensitive offloading system using machine-learning classification algorithms for mobile cloud environment. *Future Generation Computer Systems*, Vol. 90, pp.503-520.
- [49] Nguyen, B. and Simkin, L. (2017), "The internet of things (IoT) and marketing: the state of play, future trends and the implications for marketing", *Journal of Marketing Management*, Vol. 33, pp. 1-6.