Exploring the Effectiveness of Service Decomposition in Fog Computing Architecture for the Internet of Things

Badraddin Alturki, Stephan Reiff-Marganiec, Charith Perera, and Suparna De

Abstract—The Internet of Things (IoT) aims to connect everyday physical objects to the internet. These objects will produce a significant amount of data. The traditional cloud computing architecture aims to process data in the cloud. As a result, a significant amount of data needs to be communicated to the cloud. This creates a number of challenges, such as high communication latency between the devices and the cloud, increased energy consumption of devices during frequent data upload to the cloud, high bandwidth consumption, while making the network busy by sending the data continuously, and less privacy because of less control on the transmitted data to the server. Fog computing has been proposed to counter these weaknesses. Fog computing aims to process data at the edge and substantially eliminate the necessity of sending data to the cloud. However, combining the Service Oriented Architecture (SOA) with the fog computing architecture is still an open challenge. In this paper, we propose to decompose services to create *linked-microservices* (LMS). *Linked-microservices* are services that run on multiple nodes but closely linked to their linked-partners. *Linked-microservices* allow distributing the computation across different computing nodes in the IoT architecture. Using four different types of architectures namely cloud, fog, hybrid and fog+cloud, we explore and demonstrate the effectiveness of service decomposition by applying four experiments to three different type of datasets. Evaluation of the four architectures shows that decomposing services into nodes reduce the data consumption over the network by 10% - 70%. Overall, these results indicate that the importance of decomposing services in the context of fog computing for enhancing the quality of service.

Index Terms—Internet of Things (IoT), Cloud Computing, Fog Computing, Edge Computing, Data Analytics, Distributed Data Analytics, Constraint Awareness.

1 Introduction

Advances in the sensing and data processing capabilities of devices, coupled with that of communication networks is leading to the maturity of the Internet of Things (IoT) paradigm. A number of application domains, such as smart healthcare, smart homes and buildings, now rely on devices as varied as user smartphones, sensor network gateways and network routers for their realization. Most of these applications use the devices for sensing and data pre-processing tasks such as aggregation and filtering, with the majority of the data analysis done in centralized cloud infrastructures, which are located at the core of the communication infrastructure [1]. With the predicted increase in the number of devices (28 billion connected devices by 2021 [2], [3]), which can participate in existing and emergent applications, geographically centralised cloud data centers will find it difficult to support the highly distributed IoT devices without suffering a loss in QoS. The resultant massive flow and exchange of data from the large number of connected devices will also impact on electricity costs and carbon emissions [4], with achieving energy efficiency a significant challenge. With typical IoT applications being highly context-dependent, the resultant short but high-frequency data communication pattern from participating devices will pose a challenge to the bandwidth of the communication and cloud frameworks [5]. The existing computing and communication infrastructure is likely to cause unacceptably high latency in service delivery and network congestion [6], with recent studies showing that cloud servers geographically situated far from user devices affect latency more negatively than those geographically closer [1]. IoT applications that collect sensitive data such as users' private information or location face the challenging decision of whether to store it locally or communicate it to the cloud, since securing the data will incur overheads and subsequently affect performance [7].

There has been a recent interest in moving away from centralized data processing centers to a more distributed fog computing paradigm to bring computing to the edge of the network, closer to user devices [8], [9]. Fog computing is defined as a hierarchically distributed computing paradigm that bridges cloud data centers and IoT devices [10]. The combination of IoT devices and fog computing enables smart environments that can respond to real-time events by combining services offered by multiple, heterogeneous devices. This can be achieved by decomposing the services into linked-microservices which can distribute the data processing as close as possible to the data source. Microservices are defined as independent, tiny autonomous services which function together to complete a task [11]. It is worth noting that if the microservices are linked to each other, then the

B. Alturki and S. Reiff-Marganiec are with the Department of Informatics, University of Leicester, Leicester, UK e-mail:{baba1, srm13}@leicester.ac.uk
C. Perera is with the School of Computer Science and Informatics, Cardiff University, Queen's Buildings, 5 The Parade, Roath, Cardiff, CF24 3AA, UK e-mail:charith.perera@ieee.org
S. De is with the Institute for Communication Systems, University of Surrey, UK, UK e-mail:s.de@surrey.ac.uk

distribution of processes among fog nodes will be admissible in an IoT architecture. In other words, moving the computation from centralized approaches to more distributed ones will be possible, leading to a reduction in data communication cost over the network and reduced data frequency between fog devices and the cloud [1], [12]. Most of such microservices can implement typical machine learning (ML) tasks that can deal with the volatility and heterogeneity of data produced in IoT environments. Data processing using ML techniques in typical IoT-fog applications consists of well-defined steps such as feature extraction, pre-processing and applying relevant algorithms.

Though there have been advances made in fog computing with proposals for reference architectures [13], [14], practical realizations need to tackle the challenge of resource management [1]. A recent study [1] identifies that resource management at the edge of the network requires the following problems to be solved: provisioning fog nodes for executing workloads, managing computing, battery etc. resources on nodes and deploying workloads on nodes. All of these require knowledge and an awareness of the resources available on edge devices as well as constraints related to the services that run on such nodes. While automated service composition for Web services and for pervasive computing environments has been studied extensively [15], existing approaches do not focus on the costs related to time and reliability [16], while also ignoring service constraints or the data computation capabilities of the devices [17]. Thus, there is a need for adapting the service composition approach for IoT domains by considering all the aspects (computation time, reliability and constraints). Moreover, the range of possible data computation capabilities in the IoT devices also needs to be taken into account when distributing service processes among the nodes.

Thus, in this paper, we focus on the research problem of arriving at the best service computation distribution strategy that is cognizant of node constraints and can deliver a reduction in data communication cost for different types of data modalities. For this, we conduct a range of experiments to see how the traditional machine learning algorithms perform in the fog computing domain to highlight the importance of efficiently selecting which services should run on which node.

We acknowledge that theoretical analysis is important, however, we consider such analysis to be out of the scope of this paper. Additionally, our objective is to demonstrate the practical validation of the proposed approach. Our evaluation strategy is similar to the work that has been done by [18].

We have used several machine learning algorithms including Naive Bayes, Logic Regression, K Nearest Neighbours (KNN), Decision Trees, Multi-layer perceptron (MLP) and Support Vector Machine (SVM) to explore and test which algorithm is the best fit for the given data modality. Based on the results, there is no single optimum technique for all types of datasets. Every algorithm has different training time, with some requiring less time or less storage such as Naive Bayes, KNN and Decision trees. However, in terms of executing the persistent features and multi-dimensional datasets, the SVM and MLP algorithms can perform them effectively. The authors in [19] have reviewed many machine learning algorithms and they stated that it is not possible for an algorithm to perform better than others for all given datasets. This paper has surveyed the well known techniques with the focus being to find the key concepts. Therefore, our selection of machine learning algorithms was based on their efficiency in different datasets and being well known techniques.

1.1 Sustainability

Sustainability is important when deploying real-world systems because of several factors such as computation strategy, energy consumption, computation workload and data distribution strategy. The authors in [20] discussed ten crucial characteristics including data analytics, security and privacy, context awareness, mobility and other features to develop sustainable fog computing architectures. A sustainable system aims to optimise trade-offs when selecting the computation strategy, energy consumption and data communication usage. Thus, the proposed infrastructure can help develop sustainable computing architectures as it can enable handling of more computation workload at the network edge by distributing the data computation efficiently, which also has positive implications for energy consumption.

1.2 Fog and Edge Computing

Much of the literature mentions both fog computing and edge computing interchangeably [21], [7], [22], with some papers stating that edge is a synonym of fog computing [23]. Both fog and edge paradigms agree on the concept of moving the computation as much possible from a centralized level to distributed or decentralized levels. However, authors in [24] stated that they differ in terms of the radio access network, with fog computing involving Wireless LAN (WLAN) or cellular networks, but edge computing is usually cellular. In this paper, we have used fog and edge computing interchangeably because of their similarity in moving the computation from centralised clouds to the edge of the network. The proposed linked-microservices decomposition strategy can be extended and applied to a range of edge devices, such as switches and routers if their device and data computation capability information can be obtained, as has been demonstrated in [25].

1.3 Contributions

The range of fog nodes and data modalities (i.e. numerical, text and image) considered in our experiments are drawn from representative IoT-enabled fog computing applications such as crowd surveillance [26], [27], service provision for massive ad-hoc crowds (e.g. 10 million Hajj pilgrims [5]), optimized computation distribution [1], augmented brain-computer interaction game [28], visitor-identification system in smart homes [6] etc. The main contributions of this paper are as follows:

- We explore the importance of decomposing services into linked-microservices in a distributed architecture in fog computing domain for enhancing the QoS and meeting the low latency requirements of IoT-fog applications.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSUSC.2019.2907405, IEEE Transactions on Sustainable Computing

- We explore how different machine learning problems can be efficiently dealt with using service decomposition.
- We propose an efficient approach, which decomposes the services and deploys them as close as possible to the edge of the network. We have conducted a trade-off analysis to demonstrate the usefulness of different service decomposition strategies, with the evaluation of the four possible strategies showing that decomposing service computation over the fog nodes reduces the data consumption over the network by 10% (for text data) - 70% (for numerical data). This reduction in the data flow in turn implies less energy and bandwidth costs for the network, while also enabling reduced overheads for securing the condensed data features.
- We have used different types of data modalities including numerical, text and images using different ML algorithms, since these are the most common modalities of IoT data sources, as shown from our analysis of a variety of IoT-fog applications above. Thus, we believe that this is generalizable for most of the cases in this problem domain.

In this work our objective is to show that splitting the services or decomposing the services into microservices should help in executing the services effectively, we propose as our future work to build a systematic way to divide the workload. For example, we tried different decomposition techniques and the results were different for each decomposition technique depending on the specific use case; for different kind of datasets the results will be different based on the results that we have achieved.

The remainder of this paper is organized as follows: section 2 reviews related works; section 3 presents a motivating scenario that is representative of the problem domain, while section 4 presents our methodology through the different datasets and ML algorithms considered for the experiments. Results are presented in section 5, followed by the corresponding evaluation and discussion in section 6. Pertinent research challenges are discussed in section 7, before we conclude the paper in section 8.

2 Related Works

This section reviews the current state of the art from the two aspects relevant to the problem definition: a. various computation distribution strategies employed in fog computing and the associated architecture implementations and b. service decomposition as has been researched in the wider Web services paradigm.

The architectural aspect related to the location of the data processing is important. Data processing can be applied in various architectures including centralized, decentralized and distributed as follows [39]:

- In a centralized architecture, all sensors' data will be transmitted to the cloud for processing. It is widely accepted that the cloud has unlimited processing power which allows processing large data in an effective manner. Nevertheless, in a real-time case study, the data communication over the network will be huge, which will lead the cloud to be insufficient for efficient data fusion. In addition, this architecture will be more tricky when dealing with image data. The reason is that the data arrival time will be delayed which will adversely affect the result.
- In a decentralized architecture, there are different nodes with diverse computational capabilities in the network, accordingly, there is no central server similar to the centralised system. A node can apply data fusion autonomously to the local data and the data that are obtained from peers or neighbours. One of the major disadvantages of decentralized architecture is the huge cost of communication among peers. In this regard, there might be lack of scalability when increasing the number of nodes.
- In a distributed architecture, sensors data will be processed at the source prior applying data fusion to a particular node which has the capability to fuse the data. This can cope with different problems of the centralised architecture and reduce the high cost of communication among peers in the decentralized architecture. However, this architecture can bring several challenges such as data distribution which needs to meet the flexible requirements under situations that are not expected. In addition, privacy and security can be one of the challenges that distributed architecture can face as the data will be saved in different locations, but security and privacy can be considered as a challenge that most of the architectures face [40].

In recent years, there has been an increase in the amount of literature on distributed architecture in the IoT. One of the attempted proposals is by [29]; they proposed distributed architecture for fog computing for analysing big data in a smart city. They distribute the smartness to the devices in edge and computation at every layer which executes applications that have latency awareness. A fog computing based face resolution framework is proposed in [30] which obtains the information by analysing facial image. There are several features of this framework, including reduced data communication over the network and the response time of resolution, also efficiently solving the issues with bandwidth. A proposal in the distributed analysis by [31] which is a model that combines processing power which is at network level including edge and data centres to process and analyse the data from collection point to a destination. Furthermore, in [32], personal modal training method has been proposed in which the data processing, particularly machine learning is applied to private data in devices that have constraints and raspberry pi was used to test the feasibility of the IoT device in the implementation of such methods. Authors in [1] proposed a framework for managing edge nodes which is called Edge NODE Resource Management (ENORM). In addition, they proposed several techniques that provision edge node resources. They used online game called PokeMon Go-like to check the feasibility of their framework. Their results show that by using ENROM the application latency is reduced between 20 - 80%. In [6] authors proposed an approach (latency aware) to place application modules on fog nodes to make sure that

TABLE 1: Summary of Related Fog Computing Work

Work	Data Modality	Fog Node	Fog Node Functionality	Distribution Strategy	Application
B. Tang, et al. [29]	Real time Temperature sensing data	Parallelized small computing nodes	Data sensing, Pre-processing and data analysis	The workload of Data analysis parallelized between edge nodes, parallel computing mechanism.	Latency aware and location aware
P. Hu, et al. [30]	Three public face databases are used including Georgia Tech, Caltech and BioID	Personal computer and Laptop	Face Detection, Pre-processing, data analysis and computation offloaded from cloud	part of the computation tasks is offloaded from cloud to fog nodes.	Latency aware and network transmission sensitive
A. R. Zamani et al. [31]	simulated data	Virtual Machines with resource limitations	Data Sampling and Camera Aggregator.	Resource federation model. Workload distribution based on value of data. Job Scheduling optimization strategy.	Quality of Service, minimizing the computation cost and time of processing the job. Reduce data transfer time.
Servia-Rodriguez et al. [32]	WISDM dataset [33] (Numerical data), Wikipedia [34] and NIPS [35] datasets (Text data)	Raspberry pi and Personal devices	Supervised and Unsupervised Learning.	Share model (Training model) is distributed from cloud to personal devices to create personal model	Privacy preserving, improving accuracy and maintaining Efficiency.
Ni, Lina, et al. [36]	Simulated data	Linux based Computers	Fog can select satisfying resource from previously allocated resources	resource allocation strategy based on Priced Timed Petri Nets (PTPN)	Aware of task completion price and time.
N. Wang, et al. [1]	Open dataset and iPokeMon game [37].	Resource Allocation, request managing, communication latency monitoring, allocate or deallocate resources to containers.	Linux based Containers	Offloading workloads from cloud to fog.	Latency Aware
R. Mahmud et al. [6]	Simulated data	Heterogenous nodes in modelled environment	Intermediate layer between cloud and IoT devices	Optimization of resources	Latency Aware
D. G. Roy et al. [38]	Experimental data	Laptop, Personal Computer and mobiles.	Cloudlet agent, offloading tasks to mobile phones.	Offloading applications from cloud to multi-cloudlet.	Latency Aware and power consumption is reduced

the service delivery satisfies the deadline for diverse applications. They modelled and evaluated their policy in Fog environment that is simulated in iFogSim [28]. In resource allocation for fog computing, authors in [36] proposed effective resource allocation approach depending on Priced Timed Petri Nets (PTPN) for fog computing influenced by online shopping sales. Furthermore, the users can select the required resources dynamically from already allocated resources. Also, they showed an algorithm that predicts the cost of time and price for finishing jobs relies on PTPN structure.

Table 1 shows a summary of the related works of fog computing, which are reviewed along the following aspects:

- Data Modality: format and modality of the data being used for implementation and validation.
- Fog Node: types of fog nodes that are considered (user phones, low power embedded devices, general purpose computers, high-performance computers etc.)
- Fog node functionality: functions that the fog nodes perform such as data sensing and pre-processing tasks, computation offloaded from cloud.
- Distribution Strategy: strategies used to distribute services, workload or computation to fog nodes from other nodes or from the cloud.
- Application: context aspects considered in the computation/service distribution.

It can be seen that several data modalities and formats are used, ranging over numerical, text and image data. Different types of devices are used as fog devices to perform different functions. The distribution strategy used is typically tied to the application scope and focus, ranging from data analysis parallelization, computation offloading to different optimization strategies.

In web service composition, there is one proposal that focuses on autonomous web service composition by taking care of service constraints [17]. It is important to be aware of the constraints on services, but in the IoT, there are nodes which also have various constraints. This means that there is a need to have a constraint awareness approach for both services and nodes. In addition, [41] investigated the service composition's requirements and the way to obtain a composite service using transport domain as an example. They provided several scenario based approaches to service composition and discussed these. Additionally, authors in [42] proposed a comprehensive device collaboration model which has four layers, namely device, device-oriented web service (doWS), resource and process. This model shows the possibility of integration between devices and web services, also the devices can be considered as active actor because the data is not sent immediately to servers. The authors in [43] proposed a service based model in requirements decomposition, their model process starts with user requirements which are defined as goals, service discovery and discovered services employed to check the feasibility of the preceding decomposition. They decomposed the requirements of three web service composition cases to validate their approach. Another proposal in service decomposition domain by [44] who proposed a greedy algorithm that decomposes interface into interfaces depending on cohesion. This approach mainly focused on improving the cohesion and it was successful in that regards, but other aspects are not considered like coupling between interfaces.

Furthermore, authors in [45] proposed quality of service aware and energy centred service selection algorithm for service composition in the Internet of Things. The idea of the algorithm that it is possible to save energy by decreasing the degree of quality of service while maintaining the ex-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSUSC.2019.2907405, IEEE Transactions on Sustainable Computing

pectation of the user. The algorithm has two phases, the first one performs pre-selecting the services by proposing quality of service degree which meets the user's needs. In the second one, in order to select the best option among selected services, a relative dominance relation has been used for the process of service composition. Additionally, [46] proposed a method to perform data fusion via service composition model of DOHA (Dynamic Open Home-Automation) which is SOA-based middleware in a distributed manner. Every service is liable to get data from outer services using composite processes to control, fuse or create new information. In the implementation, they used DPWS (Device Profile Web Service) which is a framework that develops lightweight service for constrained devices. This framework put restrictions on web service specifications that allow the web services to run on resource-constrained devices, for example, the size of messages. In [47] authors investigated the possibility of building complex services in IoT environment. They showed the SYNAISTHISI IoT adaptable platform that is able to combine services, devices and people with systems. The services in this platform are enriched semantically using ontologies. They developed intelligent meeting room ontology and presented the way that the developer can create a service, that defines how many people are inside an intelligent room. Authors in [48] examined the problems of microservices granularity and how it affects the latency. They simulated the deployment of microservices with two approaches including microservices in one container and microservices divided into several containers. They observed a slight increase in latency for several containers over single container deployments.

Our review of the existing literature shows that most of the proposed models, approaches and architectures do not take into account resource constraints of the IoT devices. We acknowledge that many works have been done on data processing in resources constraint devices, although none of them have proposed service decomposition as a viable solution. The proposals in the service computing domain usually do not focus on deployment of services on nodes by considering the constraints of devices; instead mostly focusing on the quality of services and constraint requirements of services. However, in the case of IoT systems, there will be many constrained devices distributed across the network. This means it is important to decompose the resource-heavy services into smaller micro-linked services which can be distributed to and handled by constrained devices.

3 Problem Analysis

IoT devices have constraints on resources like RAM, CPU and storage and the services that execute on these devices have restrictions expressed in terms of the same resources. Additionally, service execution and distribution also needs to take into account the data computation capabilities (i.e. in terms of the installed library support) of the devices. Therefore, we need to model the data about services to get the knowledge about their restrictions before distributing them across the nodes. For example, the image recognition service needs at least 500MB to be executed, so the IoT devices capability should be powerful enough to execute this service. Therefore, if the device has a limitation in processing power, then it is not possible to execute the service on it. In this case, service decomposition is an important aspect of the IoT architecture due to the involvement of devices with limited hardware capabilities and varying data computation availability, which cannot handle resource intensive tasks. In addition, decomposing services into linked-microservices is an important aspect in terms of service composition in the IoT architecture. This is crucial for effective distribution of services to the nodes in the IoT. The main challenge is to determine which services should be executed on which node in a given IoT architecture, by considering both overall efficiency and feasibility. This is similar to Job shop problem which is one of the most known problems in combinatorial problems [49]. Basically, the idea of the 'job shop' problem, is that there are a group of machines with varying levels of computational power (i.e., given a specific job, j, it could be the case that there are two machines that complete j in different amounts of time, with the quicker one having higher computational power). The problem then asks for an algorithm which produces an optimal assignment of jobs to machines, such that the overall amount of time it takes for all jobs to be completed is minimal. The following scenario illustrates the problem by using a real use case. The scenario is drawn from a recent UAV (unmanned aerial vehicle) crowd surveillance study [26], which looked at energy efficiency achieved by offloading the facial recognition operation to a Mobile Edge-Computing node rather than processing it locally on the UAV (using a Raspberry-Pi for computation). We extend this in our case to include multiple fog nodes (devices) with varying hardware and computation abilities. The scenario demonstrates the significance of deploying the right service on the right node.

3.1 Motivating Scenario: City and Particular Event

There is a major event in a city where people are taking videos and photos. The law enforcement agencies are interested in re-utilizing the captured images to identify criminals (or person of interest) among the crowds in order to anticipate crimes. In Figure 1, there are four types of Nodes (N_i) including mobile phones (N_1), drones (N_2), street lights (N_3) and cloud (N_4). Every node has a different combination of resources (CPU, RAM, energy, storage and network

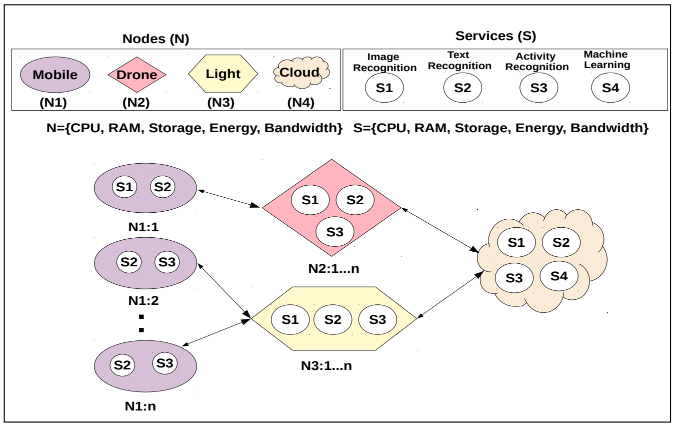


Figure. 1: Scenario: City and Particular Event

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSUSC.2019.2907405, IEEE Transactions on Sustainable Computing

TABLE 2: Parameters

Parameter	Description
s_i, $s_i \in \{ s_1, s_2, s_n \}$	Services
N_i, $N_i \in \{ N_1, N_2, N_n \}$	Nodes

bandwidth) and each node can execute several different services (S_i). Each service requires a specific combination of resources to be executed on a given node. Additionally, the facial recognition service comprises a variety of linked-microservices corresponding to ML tasks, including facial feature extraction, data fusion, data filtering and face detection algorithms. The event goers are taking the photos and videos of the event primarily for their own pleasure, not for helping the law enforcement agencies. However, they may like to help the law enforcement agencies to maintain safety as long as the primary function of their devices in this scenario is not compromised.

The sustainability comes from better energy consumption, less communication means longer duration and more devices can be connected together. The overall architecture should not consume too much energy or communication bandwidth from users' devices. The simplest case of service distribution will send all the raw data to the cloud from individual user devices even though it consumes a lot of network bandwidth. However, if we can do the data transformation in a smartphone, then it consumes less bandwidth and sends only processed data to the cloud. The services associated to the face recognition ML tasks are deployed dynamically in smartphones. It is crucial to consider which responsibilities should be assigned to smartphones. In addition, the nearest lamp or drone will have more computing power than the phones and can handle the more computation intensive tasks than user devices. This reduces the communication cost in two ways: first, the images will be transformed into feature vectors in the mobile, and secondly, further processing will be applied to data when they are sent to the lamps or drones which are temporarily deployed because of the event and connected with the mobiles via Bluetooth which is cheap in terms of communication. Then the transformed data will be sent to the cloud via 3G for further analysis.

This scenario introduces a number of challenges since it involves deploying dynamically composed services during an event in the city. The event happens on a particular day and people are likely to move around while taking photos which introduces unpredictability in their location. The service orchestrator needs to be aware of the resources available on the users phones when sending a request for service computation to them. It is crucial to consider how to compose services like sending data to the drones as mediator. Additionally, what services should be deployed to the drone and mobile is also an important aspect. This scenario illustrates the problems involved in service provisioning on fog nodes (taking into account varying hardware and data processing capabilities) and deploying the service taking into account data communication costs.

4 Methodology

As we discussed in the literature, there are clear trade-offs among the three architectures (centralized, decentralized and distributed). It is possible to find solutions that can maximize the advantages and minimize the disadvantages of each architecture. Our proposed method endeavour to fulfill this, which is presented below. We propose an efficient approach which aims to move the computation from the cloud to the fog as much as possible.

The goal G is to process a service s in an effective and efficient manner.

$$s_i \rightarrow \mathrm{G}(s_i)$$

We begin with decomposing services into set of linked-microservices MS to distribute them among nodes in our architecture.

$$\forall\ s_i,\ s_i \rightarrow \mathrm{MS}(s_i)$$

A good illustration of this is shown in Figure 2, there are three services namely Activity s_1, image s_2 and text s_3 recognition. These services will be decomposed into linked-microservices (MS) before the distribution process. Then, the distribution process will distribute the micro services depending on the constraints on services and nodes of the nodes. Then we apply data processing techniques to sensors' data to extract features and reduce the number of data points in the fog node (FN).

$$MS(s_i) \rightarrow \mathrm{FN}(MS(s_i))$$

This phase is significant because it is not possible to analyze raw time-series data by algorithms of classification effectively. Then, the cloud will receive the extracted features for creating inferences and training purposes.

$$MS(s_i) \rightarrow \mathrm{CN}(MS(s_i))$$

Therefore, to get the results we compose both the fog node and the cloud node to process the service.

$$\mathrm{FN}(MS(s_i))\ ,\ \mathrm{CN}(MS(s_i)) \rightarrow \mathrm{G}(s_i)$$

We conducted experiments for each of the above architectures in order to explore how the hybrid data analytics architecture would be beneficial in a variety of ways.

We have used three types of datasets namely numerical, text and image. The details of each dataset including dataset description, algorithms and process will be discussed below. Each dataset has its own description and algorithms. However, they have a common process in terms of decomposing the services and distributing the computation over the nodes. It is worth discussing the similarities before discussing the details of each experiment.

4.1 Process of three types of experiments

We conducted 4 sets of experiments under each of the three types of experiments which are Numerical, Text and Image data as shown in Figure. 3.

First Experiment: we sent all raw data to the cloud and data transformation methods are applied on the raw

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSUSC.2019.2907405, IEEE Transactions on Sustainable Computing

data to extract features. Then, we applied Machine learning to the altered data based on the extracted features in the cloud as shown in Figure. 3 ({1, 2, 3}: {Numerical, Text Data, Image Data}. (Cloud)). We calculate the accuracy of each algorithm, the amount of data that is sent to the cloud and the execution time.

Second Experiment: we applied data transformation methods to the raw data to create features. Then, we applied analytical algorithms to the modified data based on the extracted features in the fog as shown in Figure. 3 ({1, 2, 3}: {Numerical, Text Data, Image Data}. (Fog)). We check how feasible is the resource constraint device when processing the data and we measure the time of the execution.

Third Experiment: we applied the feature extraction methods to the raw data to extract features in the fog. By applying this the data is minimized as much possible in the fog, then the transformed data will be sent to the cloud for further analysis. as shown in Figure. 3 ({1, 2, 3}: {Numerical, Text Data, Image Data}. (Hybrid)). We calculate the accuracy of each algorithm, the amount of data that is sent to the cloud and the time of execution.

Fourth Experiment: it is similar to the third experiment in terms of applying data aggregation algorithms to the raw data in order to extract features in the fog. However, in this approach, we applied the feature extraction on part of the raw data in fog and the remaining in the cloud. We divided the dataset into two parts randomly, where 70% of the data will be used in fog and the remaining 30% will be sent to the cloud. In this experiment, the statistical measurements in the Fog are presented as S1.1, S2.1, S3.1, S4.1, S5.1 and S6.1, which are applied on the 70% of the raw data. However, in the Cloud the statistical measurements are presented as S1.2, S2.2, S3.2, S4.2, S5.2 and S6.2, which are applied on 30% of the raw data. In numerical data, in the first part, we applied fusion methods on the 70% of the raw data which is 768746 rows that is equal to 35 MB (70% of the file size) in the fog. Then, we sent the transformed data which is equal to 0.84 MB and remaining raw data (30% of the data) which is equal to 15 MB to the cloud for data transformation and then analysis as shown in Figure. 3 (E1: Numerical - 6 Measurements. (Fog+Cloud)). In text data, the first part has 14 newsgroups which are 70% of the data and it will be processed in the fog. The second part has 6 newsgroups which are 30% of the data and it will be transferred to the cloud. Then, we sent the transformed data and remaining raw data to the cloud for further data transformation and analysis as shown in Figure. 3 (E2: Text Data. (Fog+Cloud)). In image data, the first part is training data which have nearly 70% of all images and equals to 17185 images (392 MB file size) and it will be processed in the fog. The second part is testing data which have nearly 30% of all images and equals to 7815 images (178 MB file size) and it will be sent to the cloud. Then, we sent the transformed data and remaining raw data to the cloud for further analysis as shown in Figure. 3 (E3: Image Data. (Fog+Cloud)). In this experiment, we calculate the accuracy of each algorithm, the amount of data that is sent to the cloud and the time of execution.

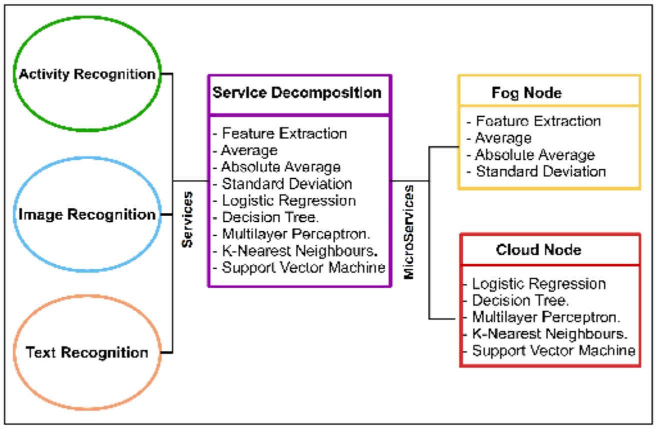


Figure. 2: Service Decomposition

4.2 Experiment 1: Numerical Data with 6 measurements

4.2.1 Dataset Description

The dataset is called WISDM [33] which is accelerometer data that are gathered from volunteers (36 users) who are performing six activities (jogging, walking, descending downstairs, climbing upstairs, sitting and standing). The volunteers held their mobile phones (Android based) when they were doing the six activities for a period of time. The collected data is divided into 10-second chunks. Additionally, 43 features are extracted relying on every 200 readings, in which every reading contains three acceleration values (x, y and z), in the fixed chunks. The transformed data contain 5418 accelerometer traces. In addition, the average traces per volunteer is around 150 and the standard deviation is around 44.

4.2.2 Algorithms

We have used six statistical measurements that are used in [33] as shown in Figure 3: E1: Numerical - 6 Measurements. There are 43 features that are created including the mean (S1), standard deviation (S2), average absolute difference (S3), time between peaks (S4) of every axis , average resultant acceleration of all axis (S5) and binned distribution for each axis (10 equal sized bins and total 30 bins) (S6). After preprocessing the data, five methods of classification (ML) are applied including Naive Bayesian (NB), Logistic Regression (LR), K-Nearest Neighbours (KNN), Decision Tree (DT) and Multilayer Perceptron (MP).

4.3 Experiment 2: Text Data

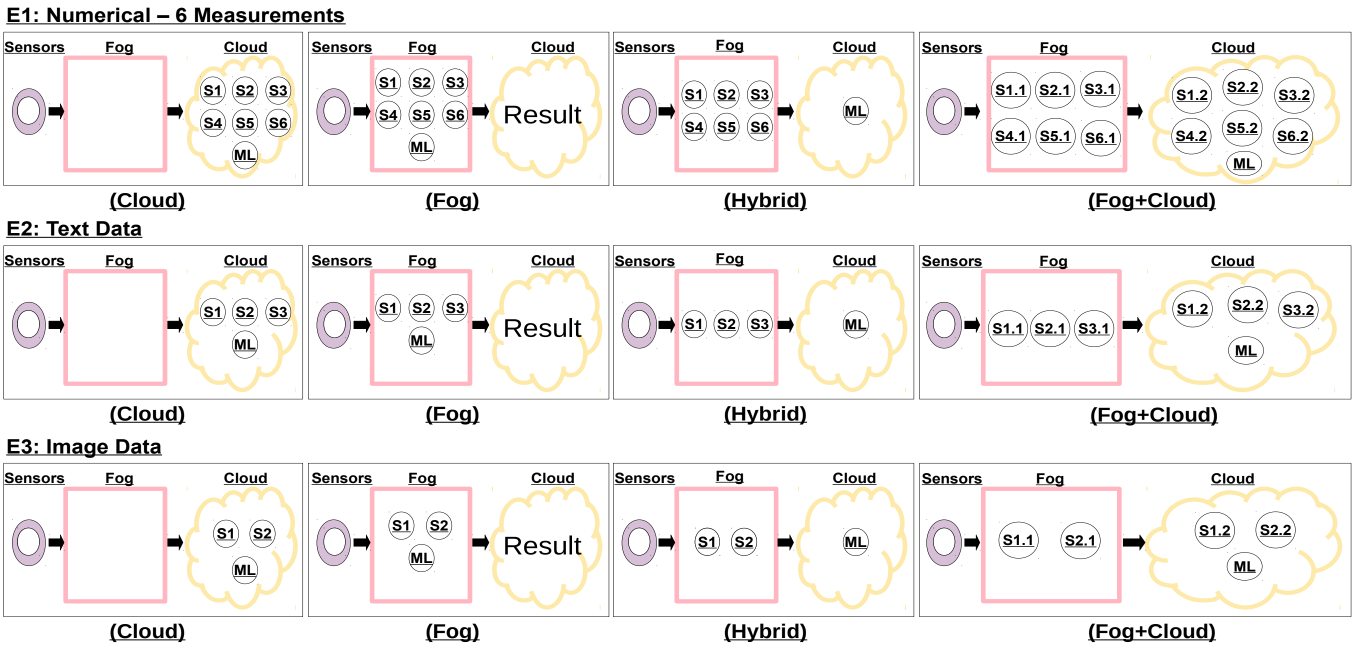
4.3.1 Dataset Description

The dataset is called Twenty Newsgroups [50] which is a set of nearly 20,000 newsgroup files. This dataset is collected for a different purpose, but it has become common data for text analysis in machine learning environment. In addition, the data are divided into 20 newsgroups and each group has a different topic.

4.3.2 Algorithms

To apply analytical algorithms to text data, it is important to convert the text into a numerical feature vector. To extract features, first, we will use Tokenizing text with scikit-learn

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSUSC.2019.2907405, IEEE Transactions on Sustainable Computing



S = Statistical Measurements as Services, ML = Machine Learning Algorithms as Services
(Cloud) = all the services are processed in the cloud, (Fog) = all the services are processed in the fog, (Hybrid) = the statistical measurement and machine learning services are distributed into fog and cloud respectively, (Fog+Cloud) = 70% of statistical measurements services processed in the fog and the rest with machine learning services are processed in the cloud.

Figure. 3: Process of Three Experiments.

(S1) which has text pre-processing and filtering. Second, from occurrences to frequencies (S2) which counts the occurrences and divide the available occurrences of every word by the whole words of the file. These features have a specific name which is term frequencies (tf). Also, downscaling (tf-idf: Term Frequency time inverse document frequency) the weights for words that appear in most of the files have less knowledgeable information than the words that appear in very small part of the file. After extracting features from data, it is possible to apply the classifier (ML) to give a prediction of the category in the post. The first classifier is naive Bayes classifier in scikit-learn which has a variety of the classifiers and the most suitable is a multinomial variant for this dataset. The second classifier is support vector machine (SVM) which can be considered one of the most used algorithms in text classification.

4.4 Experiment 3: Image Data

4.4.1 Dataset Description

Dogs vs. Cats dataset [1] which was a competition from Kaggle, is used and the purpose is to have the classification of cat and dog, so we can know if the picture has a dog or cat. The dataset is divided into two parts including training and testing data. The total number of images in the dataset is 25000 which is equal to 570 MB.

4.4.2 Algorithms

To apply machine learning algorithms, we need to convert the images into a feature vector. We are going to use two methods that take input and produces feature vector as output. First, the $image\_to\_feature\_vector$ (S1) function that takes the image as input and changes the size of the image to stable height and width and the intensity level of RGB is converted into a single set of numerical data. Second, $extract\_color\_histogram$ (S2) function gets an image as input and produces the histogram of colour to describe the image colour classification. Then, we have used k-nearest neighbours algorithm (k-NN) (ML) classifier to give a prediction of the category as either dog or cat.

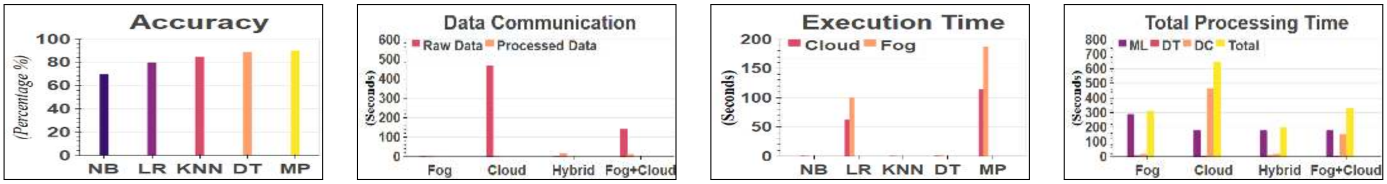
While doing all the three experiments, our expectation was that we can achieve a similar accuracy with the hybrid approach while maintaining the processing time and with considerably minimising data communication over the network.

5 Experimental Set up and Results

As discussed earlier, creating features from raw data such as numerical data, text data and image data, helps in the process of applying analytical algorithms on them. In our experiments, we have used a Raspberry Pi 3 model as the resource-constrained device. The device specification is 1GB RAM, with Raspbian Jessie as the operating system. These experiments can be performed on smartphones as well, but in our experiments we preferred Raspberry pi as it has similar specifications to smartphones, and is much cheaper that smartphones, with the cost also being a factor in IoT environments. Some papers have obtained and process data on smartphones, as in [32], [1], [51]. Furthermore, a Linux based System which has 16GB RAM, is used to mimic the

[1] https://www.kaggle.com/c/dogs-vs-cats

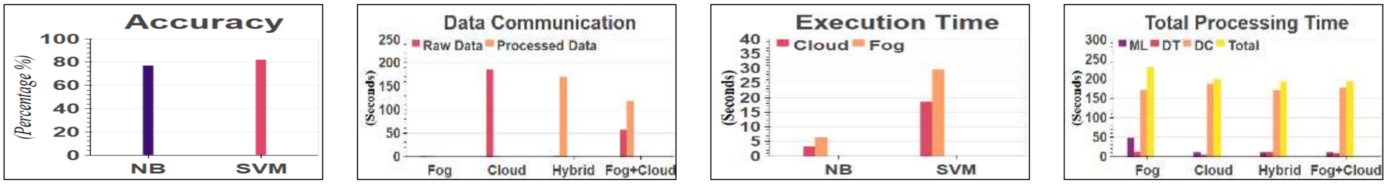
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSUSC.2019.2907405, IEEE Transactions on Sustainable Computing



(a) (b) (c) (d)

NB = Naive Bayes, LR = Logistic Regression, KNN = K Nearest Neighbours, DT = Decision Tree J48, MP = Multilayer Perceptron

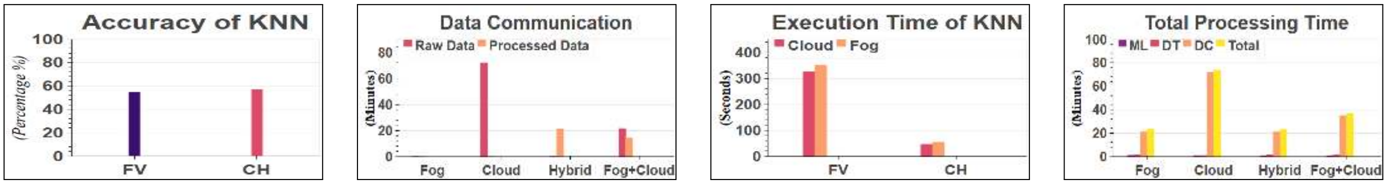
Figure. 4: Experiment 1: Numerical - 6 Measurements



(a) (b) (c) (d)

SVM = Support Vector Machine, NB = Naive Bayes

Figure. 5: Experiment 2: Text Data



(a) (b) (c) (d)

KNN = K Nearest Neighbours, FV = Image to Feature Vector, CH = Extract Color Vector

ML = Machine Learning, DT = Data Transformation, DC = Data Communication

Summary: (a) the accuracy of the analysis algorithms that are applied to the transformed data. (b) the time of data communication between the fog and cloud. (c) the time of execution of the analytical algorithms in both devices the fog and cloud. (d) demonstrates the total processing time for the four architectures.

Figure. 6: Experiment 3: Image Data

device of the cloud. For data aggregation, segmentation and feature extraction, we have used Java and python libraries.

We used python 2.7.12 and 3.5.2 and the weka 3.8 tool for machine learning (classification methods). For the numerical experiments, the weka tool is used and the heap size is adjusted in both cloud and fog. In the cloud environment, the size of heap was set to 8GB, whereas in the fog, the size was 650MB RAM for the numerical data experiment. However, the remaining experiments were done in python. While conducting these experiments, the internet upload speed was 1 Mbps. We used several packages and libraries in python for image analysis including Numpy, argparse, OpenCV packages, scikit-learn library and imutils library. In addition, scikit-learn library and Numpy packages are used for text analysis.

5.1 Experiment 1: Numerical - 6 Measurements

The results show that the highest percentage of accuracy achieved is in multilayer perception. In addition, both algorithms logistic regression and multilayer perceptron have important diversity between the two sides. Clearly, in the execution time, the cloud takes less time than the fog to perform ML algorithms because of its unlimited power. The Total Processing Time graph has three calculations for every architecture including the ML algorithms' execution time, the data transformation's execution time and the time of data communication between the cloud and fog as shown in Figure 4. $Experiment$ 1: Numerical - 6 Measurements. (d). This graph's results have been summarized in Table 3.

5.2 Experiment 2: Text Data

The results show that the highest accuracy is obtained by Support vector machine. There are two bars visible in Figure 5. $Experiment$ 2: Text Data (b): one for the transformed data and the other for the raw data. It is obvious that in the fog only device, the processing happens locally, therefore there is no data communication cost over the network. However, in terms of data communication from fog to cloud, the cloud approach was the highest as all raw data are sent. On the other hand, in the hybrid approach,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSUSC.2019.2907405, IEEE Transactions on Sustainable Computing

the data communication from fog to the cloud is lower than both cloud and fog+cloud approaches as only the transformed data are transmitted. This result emphasizes that it is possible to reduce data communications by pre-processing the data early in the network. The results show us that the two algorithms (Support Vector Machine and Naive Bayes) have considerable diversity between the two sides. It is clear that the cloud takes less time than the fog to execute ML algorithms because of its unlimited power. The Total Processing Time graph has three calculations for every architecture including the ML algorithms' execution time, the data transformation's execution time and the time of data communication between the cloud and fog as shown in Figure 5. $Experiment$ 2: Text Data (d). This graph's results have been summarized in Table 3.

5.3 Experiment 3: Image Data

The results show that applying K-NN classifier on *extract_color_histogram* has higher accuracy percentage than *Image_to_feature_vector*. There are two bars visible in Figure 6. $Experiment$ 3: Image Data. (b): one for the transformed data and the other for the raw data. It is obvious that in the fog only device, the processing happens locally, therefore there is no data communication cost over the network. However, the cloud approach was the highest in terms of data communication from fog to cloud as all raw data are transmitted. On the other hand, the hybrid approach was lower than both cloud and fog+cloud approaches in terms of data communication from fog to the cloud as only transformed data are transmitted . This result emphasizes that it is possible to reduce data communications by pre-processing the data earlier in the network. It is clear from the results that applying K-NN classifier *Image_to_feature_vector* significantly takes more time to execute than *extract_color_histogram*. However, comparing between cloud and fog there is no big difference in execution as in data communication. Obviously, the cloud takes less time than the fog to execute classification algorithms due to its unlimited processing power as shown in Figure 6. $Experiment$ 3: Image Data. (c). The Total Processing Time graph has three calculations for every architecture including the ML algorithms' execution time, the data transformation's execution time and the time of data communication between the cloud and fog as shown in Figure 6. $Experiment$ 3: Image Data. (d). This graph's results have been summarized in Table 3.

6 Evaluation and Discussion

The three approaches Fog, Hybrid and Fog+Cloud approaches have similarities in most cases. whereas, the difference between them is the location of the processing machine learning algorithm. In the Hybrid approach, the data transformation processing is done in the fog node, and then the transformed data are transmitted from fog to cloud to apply classification algorithms to the processed data. However, in the fog approach, the data transformation and machine learning process have been done at the fog node itself. The hybrid approach has an advantage over the fog approach that is utilizing the cloud's processing power for applying more sophisticated algorithms that require further processing power such as machine learning algorithms. Thus, this phase is important for minimising the processing time which has an effect on the total processing time. The results present that the proposed efficient approach is perfect for the given datasets and algorithms. As it can be seen that the results show data communication over the network is effective and gives considerable gains.

The first observation: Data communication over the network. It is widely accepted that when we increase the data size, the data communication over the network will be higher and costly.

Experiment 1: Numerical Data: In this experiment, there were approximately 1 million rows of raw data that are collected from mobile phones which nearly equal to 50MB in terms of data size. On the contrary, after we applied data fusion algorithms on the raw data we extracted features, and the rows of data are reduced to 5418 rows which equals 1.2 MB data size.

Experiment 2: Text Data: In this experiment the raw data were around 20000 newsgroups file which equal to 22.4 MB. After extracting features, the size is decreased but not significantly when we used the method to convert text into a numerical feature vector. Maybe we do not have large savings, but we created features which will allow us to do more analysis.

Experiment 3: Image Data: In this experiment, the raw data were around 570 MB. However, after extracting features from images, the size became of 170 MB. This shows that extracting the features from images in network level can help with significant savings.

According to the experiments, significant savings can be achieved in data communication over the network by applying data transformation earlier in the network. This observation will be more important when the quality and number of sensors rise significantly and therefore the resolution and rate of data will be growing quickly. By applying data fusion to the data in the fog/edge node near to the data source before they are sent to the cloud, we will reduce the data and send only meaningful data. By doing this, the energy consumption in fog devices will be reduced, these devices obtain their internet connection through networks like 3G, 4G or even 5G, therefore the devices' batteries will be lasting longer too.

The second observation: Accuracy of each algorithm. In the results, transformed data can be seen as less accurate than working with the raw data.

Experiment 1: Numerical : Overall, the accuracy loss is between 7 - 25% which is not excessive: the lowest level of accuracy is 75%, however, the highest level accuracy is approximately 93%.

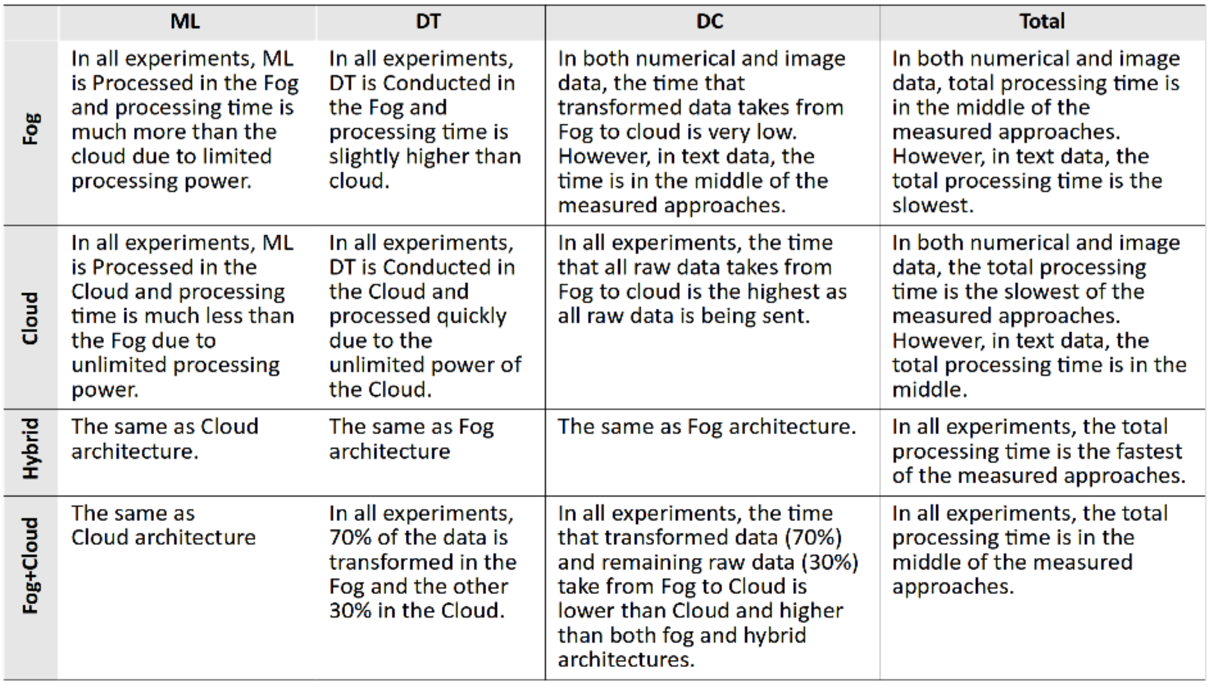
Experiment 2: Text Data: Overall, the accuracy loss is between 17 - 25% which is not extreme: the lowest level of accuracy is 77.3%, however, the highest level accuracy is approximately 82.3%.

Experiment 3: Image Data: Overall, the accuracy loss is around 40% which is greater than both previous experiments, but not significant: the lowest level of accuracy is 54.9%, however, the highest level accuracy is approximately 57.34%.

Clearly, the analytical algorithm that is used has an

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSUSC.2019.2907405, IEEE Transactions on Sustainable Computing

TABLE 3: Total Processing Time of the Three Experiments



	ML	DT	DC	Total
Fog	In all experiments, ML is Processed in the Fog and processing time is much more than the cloud due to limited processing power.	In all experiments, DT is Conducted in the Fog and processing time is slightly higher than cloud.	In both numerical and image data, the time that transformed data takes from Fog to cloud is very low. However, in text data, the time is in the middle of the measured approaches.	In both numerical and image data, total processing time is in the middle of the measured approaches. However, in text data, the total processing time is the slowest.
Cloud	In all experiments, ML is Processed in the Cloud and processing time is much less than the Fog due to unlimited processing power.	In all experiments, DT is Conducted in the Cloud and processed quickly due to the unlimited power of the Cloud.	In all experiments, the time that all raw data takes from Fog to cloud is the highest as all raw data is being sent.	In both numerical and image data, the total processing time is the slowest of the measured approaches. However, in text data, the total processing time is in the middle.
Hybrid	The same as Cloud architecture.	The same as Fog architecture	The same as Fog architecture.	In all experiments, the total processing time is the fastest of the measured approaches.
Fog+Cloud	The same as Cloud architecture	In all experiments, 70% of the data is transformed in the Fog and the other 30% in the Cloud.	In all experiments, the time that transformed data (70%) and remaining raw data (30%) take from Fog to Cloud is lower than Cloud and higher than both fog and hybrid architectures.	In all experiments, the total processing time is in the middle of the measured approaches.

ML = Machine Learning, DT = Data Transformation, DC = Data Communication

influence with trade-offs. For example, the algorithms are optimized for this localized setting as well as the used local processing power that can impact on the accuracy. The correct balance in terms of data transmission, privacy, energy consumption, accuracy and resource cost will need to be identified and our future work will further this area. The results show that the traditional architecture which is based on sending all the data from data source to a single server point has limitations. This is particularly evident when using large data volume with restrictions on time. The experiments show how the data communication over the network can be very expensive while sending large data volume without applying any transformation in advance. Additionally, it shows how effective our hybrid approach is in this regard.

Based on the result, in text data, there was no significant difference in total processing time among the four architectures. The reason for this might be the data transformation algorithms that are used. However, the idea is not just to reduce the size of data, but also to create meaningful data to get more insights. We have used three types of datasets including numerical, text and image. These datasets are not taken from an industrial application. The reason is that there are no publicly available datasets to conduct our experiments. However, the types of data in the used datasets are quite similar to industrial data in terms of numerical, text and image data. The survey in [52] has reviewed over 100 Internet of Things solutions and divided them in the industry marketplace into five classes including smart home, smart wearable, smart environment,smart city and smart enterprise. The datasets that we have used in our experiments fall into some of these classes in terms of data types (numeric, text and image).

For completeness, while working with the image dataset, one of the difficulties faced was installing imutils library [2] and OpenCV library [3] for python. Also, we used Linux based system for our experiments because we faced issues with installing libraries/packages and adapting the environments for the experiments while using other operating systems. Additionally, the accuracy of the image dataset might be a bit low, but it is possible to increase the accuracy by using different methods than KNN such as Convolutional Neural Networks (CNN). Similarly, in both datasets including numerical and text, higher accuracy can be obtained either by tuning the machine learning algorithms or by using different algorithms such as CNN. In addition, the data transformation methods can be utilised or different methods can be used to create features with more insights. These issues can be important aspects of data analytics by finding answers regarding how to increase the accuracy, but it is important to use lightweight solutions to avoid having high execution time. However, in this paper, we focused on exploring the most effective fog computing architectures for the IoT.

7 Requirement Gathering and Research Challenges

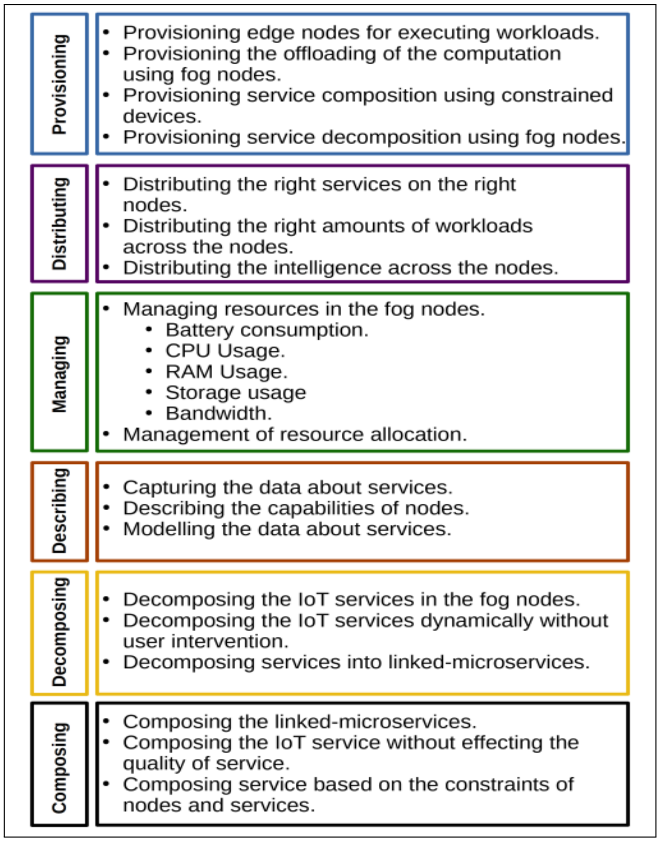
Constraint awareness is an important aspect of the IoT architecture as it will connect a large number of devices with varying computational capabilities, storage, battery power and Internet connectivity. Further, there will be a

[2] https://pypi.python.org/pypi/imutils
[3] https://opencv.org/

variety of services with different requirements (e.g. resource requirements, data requirements, latency requirements). The main challenge is to determine which services should be run on which node in a given IoT architecture, by considering both overall efficiency and feasibility. In addition, the management of resources at the edge of a network is crucial for evaluating the potential of fog computing. However, this is challenging in the IoT for a number of reasons. We present below several research challenges, as shown in Figure 7 that need to be overcome for practical realisations of fog computing in the IoT domain. Personal data store [53] is an example of where addressing these challenges becomes a necessity.

7.1 Provisioning

The first challenge is the provisioning of edge nodes for executing workloads that are offloaded by other fog nodes [54] or from cloud servers [1]. The reduced hardware and processing configurations, heterogeneity of available resources across the range of possible edge nodes and the ”lack of standard protocols for initialising services on a potential edge node” [1] add to the complexity. However, harnessing the capabilities of the diverse resources can contribute to extending the boundaries of a cloud system and provide additional revenue models for network providers, by offering incremental data processing as it moves from the source to its destination [31]. Moreover, matching service execution requirements to the fog nodes' available configurations is also key, necessitating composition or decomposition. For example, consider a service which needs a device configuration of a quad-core processor, 2 GB RAM and 4 GB Storage. Executing this service on most fog nodes is not possible due to their limited processing power. Therefore, there is a need to decompose this service into smaller services, which comes with its own challenges. Similarly, for service composition, it is not possible to provide a composition of two services with high resource demands on fog nodes.



Provisioning
- Provisioning edge nodes for executing workloads.
- Provisioning the offloading of the computation using fog nodes.
- Provisioning service composition using constrained devices.
- Provisioning service decomposition using fog nodes.

Distributing
- Distributing the right services on the right nodes.
- Distributing the right amounts of workloads across the nodes.
- Distributing the intelligence across the nodes.

Managing
- Managing resources in the fog nodes.
  - Battery consumption.
  - CPU Usage.
  - RAM Usage.
  - Storage usage
  - Bandwidth.
- Management of resource allocation.

Describing
- Capturing the data about services.
- Describing the capabilities of nodes.
- Modelling the data about services.

Decomposing
- Decomposing the IoT services in the fog nodes.
- Decomposing the IoT services dynamically without user intervention.
- Decomposing services into linked-microservices.

Composing
- Composing the linked-microservices.
- Composing the IoT service without effecting the quality of service.
- Composing service based on the constraints of nodes and services.

Figure. 7: Research Challenges

7.2 Distribution

The second challenge is distributing the workload on the fog nodes. The lack of industry-standard application containers or Virtual Machines for the diverse edge devices makes it difficult to seamlessly distribute the computation across the edge devices. Moreover, it is not possible to process large workloads on fog nodes as mentioned previously, due to the limited processing power. It is difficult to make a decision about the amount of computation load that can be assigned to a fog node. Moreover, distributing the intelligence across the fog nodes is challenging since most of the neural network, artificial intelligence and machine learning algorithms require high processing power. Current research addresses this issue by implementing different optimization strategies which prioritize different aspects, such as a number of resources without violating application QoS [6], subjective notions of the value of data to the user to decide the location of data processing [31], or prioritising the device's primary function over offloaded workloads [1].

7.3 Resource Management

The third challenge is resource management in the fog nodes. Due to their limited computation power, distributing the workloads to edge nodes is challenging because this needs to be done dynamically with the given limited configuration of resources. Therefore, managing resources like battery consumption, CPU usage, RAM usage, storage usage and bandwidth are difficult in an environment that changes dynamically and unexpectedly which makes the process of resource allocation difficult as well.

7.4 Describing Nodes

The fourth challenge is discovering and describing node capabilities, made especially difficult due to the heterogeneous and volatile nature of the IoT, making it difficult to capture and model data about offered services which have to be done dynamically. While there exist some efforts for a standardised terminology for constrained devices, such as the RFC 7228 [55], there is a need to describe their capabilities dynamically which is challenging and impossible to do manually. Analogously, there is a need to capture and model the data about the services (e.g. RAM usage, CPU usage and others) to orchestrate and allocate them to the right nodes.

7.5 Decomposition and Composition

The fifth challenge is decomposition; as we have shown in this paper, decomposition plays an important role in increasing the quality of service, but decomposing IoT services dynamically and in an automated way is challenging in fog nodes due to the resource constraints and the dynamicity of IoT. In addition, decomposing the services in linked-microservices is challenging because it is difficult to create services that distributed to different nodes and linked to their linked-partners. Similarly, in the composition process, this will be challenging because identifying linked-microservices which are distributed across the network and then composing them is a sophisticated process in an IoT environment. Additionally, this has to be achieved without affecting the quality of service and data, while using constrained devices.

Most of the proposals in service computing domain usually do not give much attention to the nodes in IoT, focussing instead on the quality of service and service constraints requirements. However, in the case of IoT systems, there will be many constrained devices distributed across the network. This means it is important to decompose a resource-heavy service into smaller micro-linked services which can be handled by constrained devices. Then, distributing these micro-linked services to the nodes based on their capabilities is important. As IoT devices have constraints on resources like RAM, CPU and storage and the services have restrictions on the same resources. Therefore, we need to model the data about services to get the knowledge about their restrictions before distributing them across the nodes. For example, in this paper, we conducted an experiment that decomposes the services to linked-microservices and then we distributed the linked-microservices to fog and cloud based on their capabilities manually. This is particularly difficult in IoT systems because of the two key characteristics of IoT systems: the heterogeneity and volatility. Therefore, this description of services and nodes need to be modelled autonomously. Possible solutions to node description can be extending the existing Web Service Description Language (WSDL), creating node description language etc. Additionally, a possible solution to the distribution of services can be artificial intelligence (AI) planning techniques.

8 Conclusion

This paper presents an efficient approach where the raw data is preprocessed in the fog node before being transmitted to the cloud to minimise the data communication over the network. Three types of datasets are used for the experiments including numerical data, text data and image data. Furthermore, we conducted 4 experiments, including 4 architectures namely cloud, fog, hybrid and fog + cloud for each dataset to explore which one is the most effective for the Internet of Things. The results show that the hybrid approach is efficient in terms of minimizing the cost of data communication over the network while maintaining the accuracy. However, fog + cloud approach could be useful to employ in situations where fog device have limited processing capability and cannot perform all the required processing. In such situations, fog + cloud approach would perform better than cloud only approach. In addition, we used the WISDM dataset [33], 20 Newsgroups dataset [50] and Kaggle dogs vs cats dataset [56] to validate our architecture.

Future research will involve distributing services to nodes dynamically in an optimum way. Moreover, research questions that could be asked include which service(s) should be run on which node by being aware of constraints. In addition, using different algorithms to extract different features for greater data transformation. Additionally, further minimizing data communication over the network while maintaining the accuracy. Furthermore, consideration of the positive effect on privacy and evaluation of energy consumption will be elements to consider in future work.

Acknowledgment

Badraddin Alturki's research is funded by the Saudi Arabian Cultural bureau in London and his scholarship is granted by King Abdul Aziz University. Dr De's research is funded by the TagItSmart! collaborative project supported by the European Horizon 2020 programme, contract number: 688061. Dr Perera's work is supported by EPSRC PETRAS 2 (EP/S035362/1).

References

[1] N. Wang, B. Varghese, M. Matthaiou, and D. S. Nikolopoulos, "Enorm: A framework for edge node resource management," *IEEE Transactions on Services Computing*, 2017.

[2] Y. Zhou, S. De, W. Wang, K. Moessner, and M. S. Palaniswami, "Spatial indexing for data searching in mobile sensing environments," *Sensors*, vol. 17, no. 6, 2017. [Online]. Available: http://www.mdpi.com/1424-8220/17/6/1427

[3] "Ericsson mobility report," 2015. [Online]. Available: https://www.ericsson.com/assets/local/mobility-report/documents/2015/ericsson-mobility-report-nov-2015.pdf

[4] N. Kumar, J. J. P. C. Rodrigues, M. Guizani, K. R. Choo, R. Lu, C. Verikoukis, and Z. Zhong, "Achieving energy efficiency and sustainability in edge/fog deployment," *IEEE Communications Magazine*, vol. 56, no. 5, pp. 20–21, May 2018.

[5] M. A. Rahman, M. S. Hossain, E. Hassanain, and G. Muhammad, "Semantic multimedia fog computing and iot environment: Sustainability perspective," *IEEE Communications Magazine*, vol. 56, no. 5, pp. 80–87, May 2018.

[6] R. Mahmud, K. Ramamohanarao, and R. Buyya, "Latency-aware application module management for fog computing environments," 2017.

[7] M. Aazam, S. Zeadally, and K. A. Harras, "Fog computing architecture, evaluation, and future research directions," *IEEE Communications Magazine*, vol. 56, no. 5, pp. 46–52, May 2018.

[8] M. Antonini, M. Vecchio, F. Antonelli, P. Ducange, and C. Perera, "Smart audio sensors in the internet of things edge for anomaly detection," *IEEE Access*, vol. 6, pp. 67 594–67 610, 2018.

[9] B. Alturki, S. Reiff-Marganiec, and C. Perera, "A hybrid approach for data analytics for internet of things," in *Proceedings of the Seventh International Conference on the Internet of Things*, ser. IoT '17. New York, NY, USA: ACM, 2017, pp. 7:1–7:8. [Online]. Available: http://doi.acm.org/10.1145/3131542.3131558

[10] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*. ACM, 2012, pp. 13–16.

[11] S. Newman, *Building microservices: designing fine-grained systems*. " O'Reilly Media, Inc.", 2015.

[12] D. Kaur, G. S. Aujla, N. Kumar, A. Y. Zomaya, C. Perera, and R. Ranjan, "Tensor-based big data management scheme for dimensionality reduction problem in smart grid systems: Sdn perspective," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 10, pp. 1985–1998, Oct 2018.

[13] [Online]. Available: https://www.openfogconsortium.org/ra/

[14] M. ETSI, "Mobile edge computing (mec); framework and reference architecture," *ETSI, DGS MEC*, vol. 3, 2016. [Online]. Available: https://bit.ly/2IxpsME

[15] G. Cassar, P. Barnaghi, W. Wang, S. De, and K. Moessner, "Composition of services in pervasive environments: A divide and conquer approach," in *2013 IEEE Symposium on Computers and Communications (ISCC)*, July 2013, pp. 000 226–000 232.

[16] S. Wang, A. Zhou, M. Yang, L. Sun, C.-H. Hsu *et al.*, "Service composition in cyber-physical-social systems," *IEEE Transactions on Emerging Topics in Computing*, 2017.

[17] P. Wang, Z. Ding, C. Jiang, and M. Zhou, "Constraint-aware approach to web service composition," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 44, no. 6, pp. 770–784, 2014.

[18] M. Satyanarayanan, "The emergence of edge computing," *Computer*, vol. 50, no. 1, pp. 30–39, 2017.

[19] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, "Supervised machine learning: A review of classification techniques," *Emerging artificial intelligence applications in computer engineering*, vol. 160, pp. 3–24, 2007.

[20] C. Perera, Y. Qin, J. C. Estrella, S. Reiff-Marganiec, and A. V. Vasilakos, "Fog computing for sustainable smart cities: A survey," *ACM Computing Surveys (CSUR)*, vol. 50, no. 3, p. 32, 2017.

[21] M. Chiang and T. Zhang, "Fog and iot: An overview of research opportunities," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 854–864, 2016.

[22] K. Habak, C. Shi, E. W. Zegura, K. A. Harras, and M. Ammar, *Elastic Mobile Device Clouds*. John Wiley Sons, Ltd, 2017, ch. 7, pp. 159–188. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119187202.ch7

[23] S. Yi, C. Li, and Q. Li, "A survey of fog computing: concepts, applications and issues," in *Proceedings of the 2015 workshop on mobile big data*. ACM, 2015, pp. 37–42.

[24] A. Munir, P. Kansakar, and S. U. Khan, "Ifciot: Integrated fog cloud iot: A novel architectural paradigm for the future internet of things." *IEEE Consumer Electronics Magazine*, vol. 6, no. 3, pp. 74–82, 2017.

[25] J. Xu, K. Ota, and M. Dong, "Saving energy on the edge: In-memory caching for multi-tier heterogeneous networks," *IEEE Communications Magazine*, vol. 56, no. 5, pp. 102–107, May 2018.

[26] N. H. Motlagh, M. Bagaa, and T. Taleb, "Uav-based iot platform: A crowd surveillance use case," *IEEE Communications Magazine*, vol. 55, no. 2, pp. 128–134, February 2017.

[27] N. H. Motlagh, T. Taleb, and O. Arouk, "Low-altitude unmanned aerial vehicles-based internet of things services: Comprehensive survey and future perspectives," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 899–922, Dec 2016.

[28] H. Gupta, A. Vahid Dastjerdi, S. K. Ghosh, and R. Buyya, "ifogsim: A toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments," *Software: Practice and Experience*, vol. 47, no. 9, pp. 1275–1296, 2017. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/spe.2509

[29] B. Tang, Z. Chen, G. Hefferman, S. Pei, W. Tao, H. He, and Q. Yang, "Incorporating intelligence in fog computing for big data analysis in smart cities," *IEEE Transactions on Industrial Informatics*, 2017.

[30] P. Hu, H. Ning, T. Qiu, Y. Zhang, and X. Luo, "Fog computing-based face identification and resolution scheme in internet of things," *IEEE Transactions on Industrial Informatics*, 2017.

[31] A. R. Zamani, M. Zou, J. Diaz-Montes, I. Petri, O. Rana, A. Anjum, and M. Parashar, "Deadline constrained video analysis via in-transit computational environments," *IEEE Transactions on Services Computing*, 2017.

[32] S. Servia-Rodriguez, L. Wang, J. R. Zhao, R. Mortier, and H. Haddadi, "Personal model training under privacy constraints," *arXiv preprint arXiv:1703.00380*, 2017.

[33] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity recognition using cell phone accelerometers," *ACM SigKDD Explorations Newsletter*, vol. 12, no. 2, pp. 74–82, 2011.

[34] Wikipedia dataset. [Online]. Available: https://dumps.wikimedia.org/enwiki/latest/

[35] Nips: bag of words data set. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/bag+of+words

[36] L. Ni, J. Zhang, C. Jiang, C. Yan, and K. Yu, "Resource allocation strategy in fog computing based on priced timed petri nets," *IEEE Internet of Things Journal*, 2017.

[37] ipokemon. [Online]. Available: https://github.com/Kjuly/iPokeMon

[38] D. G. Roy, D. De, A. Mukherjee, and R. Buyya, "Application-aware cloudlet selection for computation offloading in multi-cloudlet environment," *The Journal of Supercomputing*, vol. 73, no. 4, pp. 1672–1690, 2017.

[39] F. Castanedo, "A review of data fusion techniques," *The Scientific World Journal*, vol. 2013, 2013.

[40] M. Barhamgi, C. Perera, C. Ghedira, and D. Benslimane, "User-centric privacy engineering for the internet of things," *IEEE Cloud Computing*, vol. 5, no. 5, pp. 47–57, Sep. 2018.

[41] P. Stelmach, "Service composition scenarios in the internet of things paradigm," in *Doctoral Conference on Computing, Electrical and Industrial Systems*. Springer, 2013, pp. 53–60.

[42] F. Chen, C. Ren, J. Dong, Q. Wang, J. Li, and B. Shao, "A comprehensive device collaboration model for integrating devices with web services under internet of things," in *2011 IEEE International Conference on Web Services*, July 2011, pp. 742–743.

[43] H. Wang, S. Zhou, and Q. Yu, "Discovering web services to improve requirements decomposition," in *2015 IEEE International Conference on Web Services*, June 2015, pp. 743–746.

[44] D. Athanasopoulos, A. V. Zarras, G. Miskos, V. Issarny, and P. Vassiliadis, "Cohesion-driven decomposition of service interfaces without access to source code," *IEEE Transactions on Services Computing*, vol. 8, no. 4, pp. 550–562, 2015.

[45] M. E. Khanouche, Y. Amirat, A. Chibani, M. Kerkar, and A. Yachir, "Energy-centered and qos-aware services selection for internet of things," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 3, pp. 1256–1269, 2016.

[46] S. Rodríguez-Valenzuela, J. Holgado-Terriza, J. L. Muros-Cobos, and J. M. Gutiérrez-Guerrero, "Data fusion mechanism based on a service composition model for the internet of things," *Actas de las III Jornadas de Computación Empotrada (JCE), Septiembre*, pp. 19–21, 2012.

[47] C. Akasiadis, G. Tzortzis, E. Spyrou, and C. Spyropoulos, "Developing complex services in an iot ecosystem," in *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*, Dec 2015, pp. 52–56.

[48] D. Shadija, M. Rezai, and R. Hill, "Microservices: granularity vs. performance," in *Companion Proceedings of the10th International Conference on Utility and Cloud Computing-UCC'17 Companion*. ACM Press, 2017, pp. 215–220.

[49] R. L. Graham, "Bounds for certain multiprocessing anomalies," *Bell System Technical Journal*, vol. 45, no. 9, pp. 1563–1581, 1966.

[50] K. Lang, "Newsweeder: Learning to filter netnews," in *Proceedings of the Twelfth International Conference on Machine Learning*, 1995, pp. 331–339.

[51] N. Chen, N. Cardozo, and S. Clarke, "Goal-driven service composition in mobile and pervasive computing," *IEEE Transactions on Services Computing*, vol. 11, no. 1, pp. 49–62, Jan 2018.

[52] C. Perera, C. H. Liu, S. Jayawardena, and M. Chen, "A survey on internet of things from industrial market perspective," *IEEE Access*, vol. 2, pp. 1660–1679, 2014.

[53] C. Perera, S. Y. L. Wakenshaw, T. Baarslag, H. Haddadi, A. K. Bandara, R. Mortier, A. Crabtree, I. C. L. Ng, D. McAuley, and J. Crowcroft, "Valorising the iot databox: creating value for everyone," *Transactions on Emerging Telecommunications Technologies*, vol. 28, no. 1, p. e3125, 2017, e3125 ett.3125. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.3125

[54] B. Zhou, A. V. Dastjerdi, R. N. Calheiros, S. N. Srirama, and R. Buyya, "mcloud: A context-aware offloading framework for heterogeneous mobile cloud," *IEEE Transactions on Services Computing*, vol. 10, no. 5, pp. 797–810, Sept 2017.

[55] C. Bormann, M. Ersue, and A. Keranen, "Terminology for constrained-node networks," Internet Requests for Comments, RFC Editor, RFC 7228, May 2014, http://www.rfc-editor.org/rfc/rfc7228.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc7228.txt

[56] Kaggle. Dogs vs. cats dataset. [Online]. Available: https://www.kaggle.com/c/dogs-vs-cats

Badraddin Alturki received the Masters degree in Software Engineering from the University of Leicester, Leicester, U.K. in 2015. He is PhD Candidate in Computer Science at the University of Leicester. His research work focus on distributing services on nodes in fog computing for the Internet of Things. He is a member in both ACM and IEEE

Stephan Reiff-Marganiec is a Senior Lecturer (Associate Professor) in Computer Science at the University of Leicester. He has worked in the computer industry in Germany and Luxembourg and held research positions at the University of Glasgow (while simultaneously reading for a PhD) and the University of Stirling. He co-chaired the 8th and 10th International Conference on Feature Interactions in Telecommunications and Software Systems an was co-Chair of three instances of YR-SOC. Stephan lead work packages in the EU funded projects Leg2Net, Sensoria and inContext focusing on automatic service adaption, context aware service selection, workflows and rule based service composition. Stephan is co-editor of the Handbook of Research on Service- Oriented Systems and Non-Functional Properties and has published in excess of 50 papers in international conferences and journals as well as having served on a large number of programme committees. Stephan was appointed Guest Professor at the China University of Petroleum and was visiting Professor at Lamsade at the University of Dauphine, Paris. He was elected Fellow of the BCS (FBCS) in 2009 and is a member in both ACM and IEEE.

Charith Perera is a Lecturer (Assistant Professor) in Computing Science at Cardiff University, United Kingdom. Dr. Perera received the B.Sc. (Hons) degree in computer science from Staffordshire University, Stoke-on-Trent, UK. in 2009 and the Master of Business Administration (MBA) from the University of Wales, Cardiff, UK. in 2012. He received the Ph.D. degree in computer science with The Australian National University, Canberra, Australia. He completed his post-doctoral at Newcastle University, UK and Open University, UK. Previously, he was with the Information Engineering Laboratory, ICT Centre, CSIRO, Australia. His research interests include Internet of Things, Sensing as a Service, infrastructure and architectures, privacy and security. Dr. Perera is a member of the ACM and IEEE. For more details: charithperera.net

Suparna De is a Senior Research Fellow at the Institute for Communication Systems (ICS), University of Surrey, UK. She obtained her Ph.D. and MSc. (with distinction) degrees in Electronic Engineering from the University of Surrey in 2009 and 2005, respectively. She has been leading technical work areas related to various aspects of service provisioning and data analysis in the Internet of Things domain in several EU projects such as TagItSmart, iKaaS, IoT.est, iCore and IoT-A. Her research has been supported by grants from the EC H2020 and FP7 programs and through DTI, UK-funded programs. Her research interests include large-scale data analytics and latent pattern discovery in Web of Things scenarios, semantic association analysis and knowledge engineering methods. She is a member of both the IEEE and ACM.