An IoT Application Business-Model on top of Cloud and Fog Nodes

Zakaria Maamar[1], Mohammed Al-Khafajiy[2], and Murtada Dohan[3]

Abstract This paper discusses the design of a business model dedicated for IoT applications that would be deployed on top of cloud and fog resources. This business model features 2 constructs, flow (specialized into data and collaboration) and placement (specialized into processing and storage). On the one hand, the flow construct is about who sends what and to whom, who collaborates with whom, and what restrictions exist on what to send, to whom to send, and with whom to collaborate. On the other hand, the placement construct is about what and how to fragment, where to store, and what restrictions exist on what and how to fragment, and where to store. The paper also discusses the development of a system built-upon a deep learning model that recommends how the different flows and placements should be formed. These recommendations consider the technical capabilities of cloud and fog resources as well as the networking topology connecting these resources to things.

1 Introduction

On top of services to provision and products to sell, organizations are known for their cultures and business models. A typical example of a successful organization's culture would be *Google* where openness, innovation, excellence that comes with smartness, hands-on approach, and small-company-family rapport are promoted (`tinyurl.com/y6ld8wn9`). And, a typical example of a successful organization's business model would be *Amazon.com* where millions of customer-enacted e-commerce transactions are successfully performed (`www.garyfox.co/amazon-business-model`).

To remain competitive and sustain growth, organizations also embrace ICTs with focus lately on the Internet-of-Things (IoT). IoT is about making things like sensors and actuators act over the cyber-physical surrounding so, that, contextualized,

[1]Zayed University, UAE · [2]University of Reading, UK · [3]University of Babylon, Iraq & University of Northampton, UK.

smart services are provisioned to users and organizations. According to Gartner (www.gartner.com/newsroom/id/3165317), 6.4 billion connected things were in use in 2016, up 3% from 2015, and will reach 20.8 billion by 2020. Another recent trend in the ICT landscape is coupling IoT with cloud computing and fog computing [15] (fog is *aka* edge). The massive volume of data that things generate, needs to be captured, processed, analyzed, shared, and protected. Data range from vegetables' freshness in warehouses to traffic flows on roads and pollution levels in cities. For many years, cloud has been the technology of choice for exposing resources (traditionally software, platform, and infrastructure) as services (*aaS) to organizations that have to "wrestle" with this massive volume of data. According to Gartner too (goo.gl/m9MQXc), "*by 2021, more than half of global enterprises already using cloud today will adopt an all-in cloud strategy*". However, despite cloud's benefits there exist some concerns about cloud's appropriateness for certain applications (e.g., medical and financial) that have strict non-functional requirements to satisfy in terms of minimizing data latency and protecting sensitive data. Transferring data over public networks to (distant) clouds could take time because of high latency, could be subject to interceptions, alterations, and misuses, and could depend on network availability and reliability. To address data-latency and data-sensitivity concerns, ICT practitioners advocate for fog computing where processing and/or storage facilities are expected to exist "next" (or nearby) to where data is collected minimizing its transfer and avoiding its exposure to unnecessary risks.

In this paper, we examine the appropriateness of developing a business model for IoT applications that would run on top of a cloud/fog infrastructure. This business model should (*i*) identify the data flows between IoT devices, clouds, and fogs, (*ii*) expose how collaboration arises between IoT devices, clouds, and fogs, and (*iii*) track where data processing and storage happen because of the distributed nature of IoT devices, clouds, and fogs. To the best of our knowledge, this is the first step towards defining such a business model in the double context of cloud and fog. Mahmud et al. report the lack of business model for fog environments, only, [10]. The rest of this paper is organized as follows. Section 2 is a brief overview of cloud, fog, and business model. Section 3 details the way we design a business model for IoT applications. Technical details of this design are included in this section, as well. Section 4 concludes the paper and presents some future work.

2 Background

Cloud and fog in brief. Cloud is a popular ICT topic that promotes Anything-as-a-Service (*aaS) operation model. This model adopts pay-per-use pricing and consolidates hardware and software resources into data centers. However, despite cloud's popularity, it does not, unfortunately, suit all applications. Cloud is not recommended for latency-critical and data-sensitive applications due to reasons such as high latency added by network connections to data centers and multi-hops/nodes between end-users and data centers that increase the probability of interceptions.

Fog was first introduced by Satyanarayanan et al. in 2009 [12] and generalized by Cisco Systems in 2014 [3] as a new ICT operation-model. The objective is to make processing, storage, and networking facilities "close" to where data is captured and/or stored. The extension from cloud to fog is not trivial due to their subtle similarities and differences. However, their suitability for certain applications remains an open debate [11]. Real-time applications that require almost immediate action and high data protection, would discard cloud in favor of fog. Varghese et al. mention that by 2020, existing electronic devices will generate 43 trillion gigabytes of data that need to be processed in cloud data-centers [14]. However, this way of operating cannot be sustained for a long time due to frequency and latency of communication between these devices and cloud data-centers. Fog would process data closer to its source so, that, network traffic is reduced and both quality-of-service and quality-of-experience are improved.

Business model in brief. Despite the critical role of business model in any organization's operation, there is not a common definition of what a business model is. According to Geissdoerfer et al., it is "*a simplified representation of the elements of an organisation and the interaction between these elements for the purpose of its systemic analysis, planning, and communication in face of organizational complexity*" [5]. In a 2015 Business Harvard Review report (`hbr.org/2015/01/what-is-a-business-model`), it is mentioned that a business model consists of 2 parts: *"Part one includes all the activities associated with making something: designing it, purchasing raw materials, manufacturing, and so on. Part two includes all the activities associated with selling something: finding and reaching customers, transacting a sale, distributing the product, or delivering the service"*. Last but not least, Li sheds light on a business model's key constructs, namely, value proposition (product offerings of the firm, its market segments and its model of revenue generation), value architecture (how a firm senses, creates, distributes, and captures values), and functional architecture (core activities of a firm, namely, product innovation and commercialization, infrastructure for production and delivery, and customer relations management) [8].

3 Business model for IoT applications

After an overview of the constructs of our proposed IoT application business-model, we details the types of flows and types of placements and then, present the implementation of a system recommending the definition of these flows and placements.

3.1 Overview

Fig. 1 illustrates the constructs we propose to define an IoT application business-model in the context of cloud/fog. The 2 key constructs are flow, specialized into data and collaboration, and placement, specialized into processing and storage.

On the one hand, the flow construct is about who sends what and to whom, who collaborates with whom, and what (time- and location-related) restrictions ex-

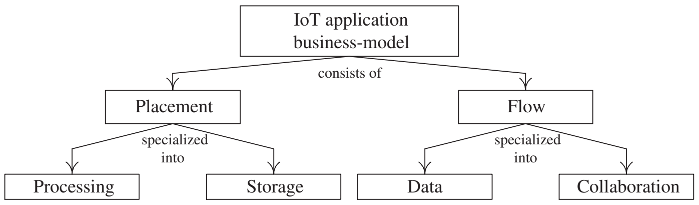


Fig. 1: IoT application business-model's proposed constructs

ist on what to send, to whom to send, and with whom to collaborate. First, the Data Flow (DF) sheds light on how data is transferred from senders to receivers where senders could be things and fogs, and receivers could be fogs and clouds. Second, the Collaboration Flow (CF) sheds light on how, when, and why thing-2-thing, fog-2-fog, and cloud-2-cloud interactions arise. On the other hand, the placement construct is about what and how to fragment, where to store, and what (time- and location-related) restrictions exist on what and how to fragment, and where to store. First, the Processing Placement (PP) sheds light on how an application's business logic is decomposed into fragments and where the fragments are executed over things *versus* fogs *versus* clouds. Second, the Storage Placement (SP) sheds light on where an application's data are spread over things *versus* fogs *versus* clouds.

3.2 Types of flows

Fig. 2 illustrates both the data flows and the collaboration flows that could be part of an IoT application business-model. First, we rely on our previous work on cloud-fog coordination to recommend how data flows should be formed [9, 16]. Second, we partially rely on our previous work on fog-2-fog collaboration to recommend how collaboration flows should be formed [1].

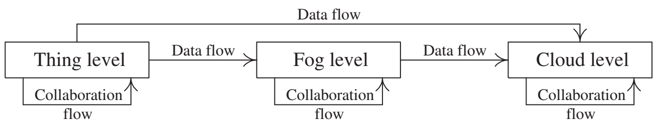


Fig. 2: Data/Collaboration flows in an IoT application business-model

Data flows connect thing and fog together ($DF_{T \to F}$), thing and cloud together ($DF_{T \to C}$), and thing and fog and cloud together ($DF_{T \to F \to C}$[1]). Compared to the work of Thekkummal et al. who identify 2 data flows, from things to clouds and from fogs to clouds where fogs collect data from things [13], we consider a third

[1] Pre-processing data at fogs prior to sending the pre-processed data to clouds.

data flow that is from things to fogs since some data do not need to be sent to clouds. It is worth mentioning that although our 3 specialized data flows could simultaneously exist, we came up in [9] and [16] with 6 criteria whose use would permit to either highly-recommend (HR), recommend (R), or not-recommend (NR) which data flow should be formed for a particular IoT application. These criteria are *frequency* (rate of data transfer from things to fogs/clouds; the frequency could be regular, e.g., every 2 hours, or continuous), *sensitivity* (nature of data exchanged between things and fogs/clouds; highly-sensitive data should not be exposed longer on networks during transfer), *freshness* (how important data exchanged between things and fogs/clouds should be up-to-date, i.e., recent), *time* (delay that results from withholding/processing data at the thing level until its transfer to fogs/clouds), *volume* (amount of data that things produce and send to fogs/clouds), and *criticality* (demands that fogs/clouds express with regard to data of things; low demands could lead to ignoring certain data). In support of these criteria, we made the assumption that, distance-wise, clouds are **far** from things and fogs are **close** to things. In Table 1, we summarize how the aforementioned criteria, taken independently from each other, assist with recommending the formation of specific data flows: T $\rightarrow$ C, T $\rightarrow$ F, and T $\rightarrow$ F $\rightarrow$ C. More details about these recommendations are presented in [9].

Table 1: Recommendations to form data flows when criteria are separated ([9])

Criterion	**Features**	T $\rightarrow$ C	T $\rightarrow$ F	T $\rightarrow$ F $\rightarrow$ C
Frequency	Continuous stream	NR	HR	R
	Regular stream			
	Short gaps	NR	HR	HR
	Long gaps	R	R	R
Sensitivity	High	NR	HR	HR
	Low	R	R	R
Freshness	Highly important	NR	HR	R
	Lowly important	R	R	R
Time	Real-time	NR	HR	HR
	Near real-time	R	HR	HR
	Batch-processing	HR	NR	NR
Volume	High amount	HR	NR	NR
	Low amount	NR	HR	R
Criticality	Highly important	HR	HR	R
	Lowly important	NR	HR	HR

- *Frequency* criterion is dependent on the data stream between things and clouds/fogs. If the stream is continuous (non-stop), then it is highly recommended to involve fogs in all interactions so, that, direct data-transfer to clouds is avoided (i.e., low-latency and low-delay jitter). If the data stream is regular, recommendations will depend on how short *versus* long the gaps are during data transfer.
- *Sensitivity* criterion is about the protection measures to be put in place during data exchange between things and clouds/fogs. If the data is highly sensitive, then it is highly recommended to involve fogs in all interactions so, that, protection is ensured. Otherwise, data could be sent to clouds and fogs.

- *Freshness* criterion is about the data quality to maintain during the exchange between things and clouds/fogs. If the data needs to be highly fresh, then it is highly recommended to involve fogs in all interactions (i.e., subject to be aware of the location of fogs and support to real-time interactions is provided). Otherwise, data could be sent to clouds and/or fogs.
- *Time* criterion is about how soon data is made available for processing. If it is real-time processing, then it is highly recommended to send data to fogs. If it is near real-time (i.e., minutes are acceptable) then it can be sent to clouds and/or fogs. Otherwise, cloud is ideal for data batch-processing.
- *Volume* criterion is about the space constraint over the amount of data collected/produced by things. If this amount is big, it is highly recommended to send data directly to clouds. Otherwise, data could be sent to fogs. In case of a big data amount and data is divisible, then data could be sent over to multiple fogs (i.e., distributed geo-distribution).
- *Criticality* criterion is about ensuring data availability according to fog/cloud demands. If fog/cloud demands are highly important, then it is highly recommended that data should be sent to fog/cloud regardless of the hop number (i.e., geo-distribution).

Contrarily to what we did in [9] where *frequency*, *sensitivity*, *freshness*, *time*, *volume*, and *criticality* criteria were taken independently from each other, we combined them all using a fuzzy logic-based multi-criteria decision making approach [16]. This approach was demonstrated using a healthcare-driven IoT application along with an in-house testbed that featured real sensors (temperature and humidity DHT11) and fog (rPi2) and cloud (Ubidots) platforms. During the experiments, we modified the *frequency* of streaming data (every 3 second, 5 second, 7 second, and randomly) for each of the 3 data flows, $T \rightarrow C$, $T \rightarrow F$, and $T \rightarrow F \rightarrow C$, and the *volume* (around low and high amount) and *criticality* (around low and high important) of the transmitted data. Upon data messages receipt at an end-point whether fog or cloud, we timestamped these messages prior to storing them. Table 2 includes 2 out of 4 scenarios that summarize the experiments we conducted with focus on the recommendations of forming specific data flows: $T \rightarrow C$, $T \rightarrow F$, and $T \rightarrow F \rightarrow C$. More details about these recommendations are presented in [16].

Collaboration flows connect things together (CF_{T2T}), fogs together (CF_{F2F}), and clouds together (CF_{C2C}) using offloading mechanism that would allow to maintain "acceptable" loads over things, fogs, and clouds. It is not mandatory that the 3 specialized collaboration flows simultaneously exist since this would depend on satisfying under-execution IoT-applications' functional and non-functional requirements. For illustration, we demonstrate how a collaboration flow between fogs could be formed based on our previous work on improving fog performance [1]. We expect adopting the same strategy when developing both collaboration flows between things and collaboration flows between clouds.

The ICT community already agrees that fog is not a substitute to cloud but a complement; both should work hand-in-hand [2, 9]. As per Section 2, fog can support,

Table 2: Recommendations to form data flows when criteria are combined ([16])

Scenario #	Criteria	Linguistic values	Recommendations
Scenario 1	*Frequency*	Regular stream (around short and long gaps)*	$\mathsf{T} \rightarrow \mathsf{C}$ is NR; $\mathsf{T} \rightarrow \mathsf{F}$ is R; $\mathsf{T} \rightarrow \mathsf{F} \rightarrow \mathsf{C}$ is R
	Sensitivity	Around low and high*	
	Freshness	Highly important	
	Time	Real time	
	Volume	High amount	
	Criticality	Lowly important	
Scenario 4	*Frequency*	Regular stream long gaps	$\mathsf{T} \rightarrow \mathsf{C}$ is R; $\mathsf{T} \rightarrow \mathsf{F}$ is R; $\mathsf{T} \rightarrow \mathsf{F} \rightarrow \mathsf{C}$ is R
	Sensitivity	Low	
	Freshness	Lowly important	
	Time	Near-real time	
	Volume	High amount	
	Criticality	Around lowly and highly important*	

*: Around Val_1 and Val_2: both Val_1 and Val_2 meet the scenario's requirements.

serve, and facilitate services that cloud does not cater well for their needs and requirements. These services are known for being latency sensitive, geo-distributed (e.g., water-pipe monitoring), mobile with high-speed connectivity (e.g., connected vehicles), and largely distributed (e.g., smart energy distribution). However, despite the benefits of fog computing, it happens that fogs working in silos cannot accommodate these services' processing, storage, and communication requirements. Compared to clouds, fogs mean *less* resources, *less* reliability, and *less* latency [7]. Promoting offloading among fogs could be an option, leading to the formation of collaboration flows between fogs according to their ongoing loads and availabilities of their processing, storage, and communication capabilities. This offloading has been the focus of our work in [1].

In compliance with Fig. 2, things periodically collect and generate data from the cyber-physical surroundings and send them to fogs ($\mathsf{DF}_{\mathsf{T}\rightarrow\mathsf{F}}$) and/or clouds ($\mathsf{DF}_{\mathsf{T}\rightarrow\mathsf{C}}$) for processing/storage as deemed necessary. A fog can serve a certain number of requests instantly or offload some to other fogs in the same domain if this fog is congested, which could delay processing these requests. It is worth noting that the importance of fogs being located between things and clouds, makes fogs more accessible/reachable to both things and clouds. Therefore, fog can be used horizontally (i.e., $\mathsf{CF}_{\mathsf{F2F}}$) and vertically (i.e., $\mathsf{DF}_{\mathsf{T}\rightarrow\mathsf{F}\rightarrow\mathsf{C}}$) in the network to provide the desired services.

By analogy with $\mathsf{CF}_{\mathsf{F2F}}$, $\mathsf{CF}_{\mathsf{T2T}}$ and $\mathsf{CF}_{\mathsf{C2C}}$ could be formed allowing to develop federations of things and federations of clouds, respectively. Communications in all federations could be either planned where links among nodes, i.e., things, fogs, and clouds, are known at design-time and according to a specific business model or ad-hoc where links among nodes are formed on the fly and sometimes opportunistically. $\mathsf{CF}_{\mathsf{T2T}}$ and $\mathsf{CF}_{\mathsf{C2C}}$ would require a coordination model at the thing level and cloud level, respectively, to ensure that a better load balancing among things and among clouds would be achieved. For instance, the decision of a cloud to collaborate with other clouds to support the processing of a received service's request would depend on the response time. Generally, the response time of a task on a cloud will be computed based on the time required to wait in the queue (in case of loaded cloud), the time to process the received task on the cloud, and the response travel-time that includes both transmission and propagation delays. Meantime, since it is a distributed model, the cloud sends requests for collaboration to all neighboring nodes within its domain to examine the possibility of providing a quicker response time, especially for time-sensitive services. It is worth noting that request-and-response times are considered part of service latency.

3.3 Types of placements

Fig. 3 illustrates both the processing placement and the storage placement that could be part of an IoT application business-model. In this figure, appropriate IT facilities run on top of clouds, fogs, and things to support the deployment of these 2 types of placements.

To work out and illustrate the details about processing placement and storage placement, we assume an IoT application that runs on top of a Business Process Management System (BPMS) coupled to a DataBase Management System (DBMS). Therefore, this application's business logic is designed as a set of Business Processes (BPs) that manipulate a set of DataBases (DBs). For the processing placement, we present some initiatives that examine BP fragmentation where each process fragment would be processed on top of either things (though unlikely), fogs, or clouds. For the storage placement, we present some initiatives that examine data fragmentation where each data fragment would be stored in things (though less likely), fogs, or clouds. Due to lack of space, only the processing placement is discussed.

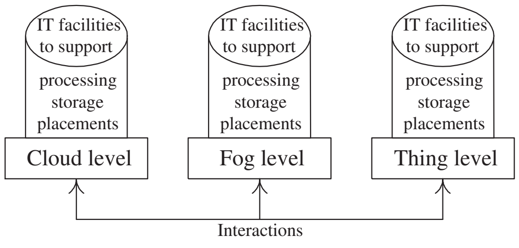


Fig. 3: Processing/Storage placements in an IoT application business-model

Processing placements identify physical locations where an IoT application's BP fragments would be deployed for processing. BP fragmentation has been the subject of many studies shedding light on its rationales, techniques, challenges, benefits, and consequences. Hereafter, we briefly present the works of Cheikhrouhou et al. [4] and Hou et al. [6]. The first two carry out BP fragmentation over clouds/fogs and clouds, respectively, and the last one carries out BP fragmentation in the context of IoT.

Saoussen et al. report that with the continuous advances in ICT and organizations' changing needs, cloud computing has shown some signs of "fatigue" when for instance, real-time applications call for almost zero time-latency. Transferring data to distant clouds is a potential source of delays and opens doors to unwanted interceptions. Luckily, fog computing addresses some clouds' concerns like latency and security. Building upon a previous approach to formally specify and verify cloud resources allocation to BPs using Timed Petri-Net (TPN), Saoussen et al. extended this approach by fragmenting and deploying free-of-violations time-constrained BPs in a mono-cloud and multi-fog context. They resorted to cloud-fog collaboration by verifying at both design-time and run-time where tasks should run (cloud, fog, or either) and where data should be placed (cloud, fog, or either). During BP fragmentation, Saoussen et al. defined $DD(data_i, t_i, t_k, h_i^s, h_k^s)$ that is data dependency between task t_i that produces $data_i$ and task t_k that consumes $data_i$ when t_i is executed in the host h_i^s and t_k is executed in another host h_k^s where a host could be either cloud or fog with respect to Fig. 3.

Hou et al. explore BP fragmentation for distribution purposes with emphasis on IoT-aware BPs. These BPs have to cope with the high volume and velocity of data that IoT generates and hence, need to be transmitted and processed timely. The authors propose a location-based fragmentation approach to partition a BP before applying Kuhn-Munkres algorithm so, that, an optimal deployment of BP fragments is achieved. Fragment collaboration takes place through a topic-based publish/subscribe infrastructure, which allows to reduce network traffic and save process execution-time.

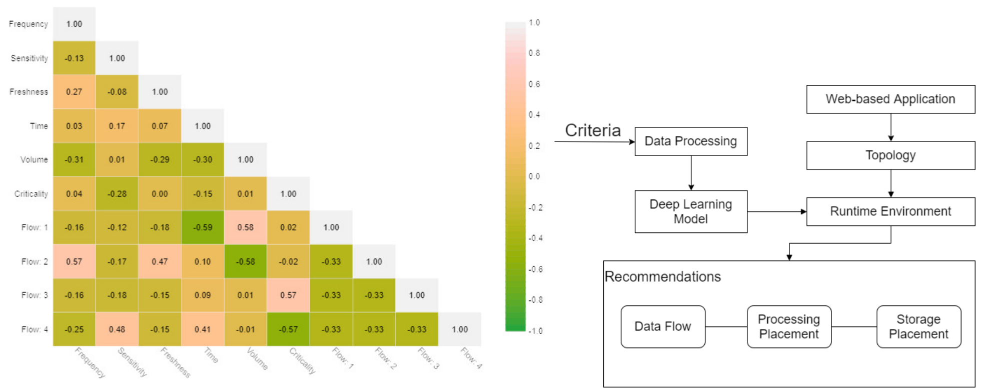
3.4 System implementation

This section demonstrates a Recommender System ($\mathcal{RS}$) that we developed to provide recommendations for data flows and processing placement as part of our IoT application business-model in the context of cloud/fog. The $\mathcal{RS}$ is associated with a Deep Leaning (DL) model that takes as inputs a network topology's constituents (i.e., number of things, number of fog nodes, and number of cloud nodes) and criteria weights (Table 1), and produces as outputs data flows and processing placement recommendations.

Deep learning model. The core of our $\mathcal{RS}$ is a DL model that is built in MATLAB on Intel core i7-6700HQ, CPU 2.60GHz, GPU GTX 1070, and RAM 32GB. This model is a multi-layer classifier that extracts features from the weighted input criteria in order to produce suggestions labeled as highly-recommended.

During the implementation, a sequential development approach (also called $feed\text{-}forward$ nn model) is adopted allowing each layer's output to be transferred to the next layer and so on. The DL model consists of 5 layers distributed over 1 layer for input, 1 layer for output, and 3 hidden layers for transformation having 20, 20, and 15 neurones, respectively. The hidden layers are used to avoid or help in preventing overfitting. It is worth noting that we experimented more/less number of layers and different number of neurons to get the best accuracy and faster processing time when providing recommendations.

The data used for training was generated by using the 6 criteria related to data flow namely, *frequency*, *sensitivity*, *freshness*, *time*, *volume*, and *critically*. Fig. 4a shows the correlation matrix for the trained data based on these criteria as well as the possible data flows. Hence, the data flow classes/labels are Flow 1, Flow 2, Flow 3, and Flow 4 referring to T→C, T→F, T→C and T→F, and finally T→F and T→F→C, consecutively. From Fig. 4a, we note the data independent and not-related to each other, hence there will be no direct/common indications or features between input data and possible output recommendations. Before the training process, a sequence of operations have been applied to input data, such as manipulate the missing data by mean substitution and normalize the data by standard deviation.



(a) (b)

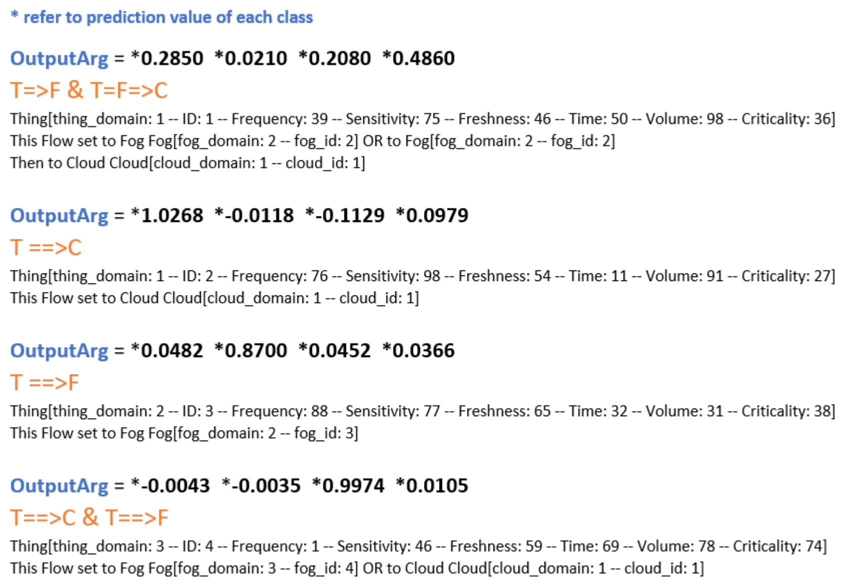
Fig. 4: Correlation matrix and recommendation generation process

Recommendation creation. The process of generating a recommendation is presented in Fig. 4b. A Web-based system has been developed to allow a user to add or design a desired network topology required for an IoT application. Moreover, the IoT application's specifications (based on the 6 criteria presented in Table 1) can also be added to the topology and encapsulated in a JSON message. To this end, the system forwards JSON messages (includes both criteria and topology) to a pre-trained DL model to find the best fitting for data flows and processing placement. It is worth noting that the DL model combines topology and application's criteria (e.g., *frequency*, and *sensitivity*) in one request to recommend the best data flow to find the best routes to.

Results and evaluation. After passing the topology and criteria to the DL model, the $\mathcal{RS}$ proceeds with predicting the best routes for data flows (i.e., T→F, T→C, etc.) and selecting the best potential neighbor node to receive the data. For instance, if there are 2 fog nodes that can be recommended, the $\mathcal{RS}$ will recommend either

Fog$_a$ or Fog$_b$ based on distance (handled by the domains in the network topology) and send rate (e.g., frequency).

The $\mathcal{RS}$ output consists of a predication value for each of the classes representing our data flows, namely Flow 1, Flow 2, Flow 3, and Flow 4. The highest value of a class represents the most recommended class. For instance, for a topology that consists of 1 Cloud node to cover Fog and Thing nodes distributed over 3 $domains$ with $domain_1$ having 1 Fog node and 2 Thing nodes, $domain_2$ having 2 Fog nodes and 1 Thing node, and $domain_3$ having 1 Fog node and 1 Thing node, the output recommendation for this topology is presented in Fig. 5. It is noted that each Thing node has it is own criteria that lead to different predictions/recommendatiosn for the data flow and processing placement. In this figure, the processing placement node is the first node in the recommended flow; for instance, in T$\rightarrow$F$\rightarrow$C flow, the Fog node (i.e., F) is for the processing placement.



```
* refer to prediction value of each class

OutputArg = *0.2850  *0.0210  *0.2080  *0.4860
T=>F & T=F=>C
Thing[thing_domain: 1 -- ID: 1 -- Frequency: 39 -- Sensitivity: 75 -- Freshness: 46 -- Time: 50 -- Volume: 98 -- Criticality: 36]
This Flow set to Fog Fog[fog_domain: 2 -- fog_id: 2] OR to Fog[fog_domain: 2 -- fog_id: 2]
Then to Cloud Cloud[cloud_domain: 1 -- cloud_id: 1]

OutputArg = *1.0268  *-0.0118  *-0.1129  *0.0979
T ==>C
Thing[thing_domain: 1 -- ID: 2 -- Frequency: 76 -- Sensitivity: 98 -- Freshness: 54 -- Time: 11 -- Volume: 91 -- Criticality: 27]
This Flow set to Cloud Cloud[cloud_domain: 1 -- cloud_id: 1]

OutputArg = *0.0482  *0.8700  *0.0452  *0.0366
T ==>F
Thing[thing_domain: 2 -- ID: 3 -- Frequency: 88 -- Sensitivity: 77 -- Freshness: 65 -- Time: 32 -- Volume: 31 -- Criticality: 38]
This Flow set to Fog Fog[fog_domain: 2 -- fog_id: 3]

OutputArg = *-0.0043  *-0.0035  *0.9974  *0.0105
T==>C & T==>F
Thing[thing_domain: 3 -- ID: 4 -- Frequency: 1 -- Sensitivity: 46 -- Freshness: 59 -- Time: 69 -- Volume: 78 -- Criticality: 74]
This Flow set to Fog Fog[fog_domain: 3 -- fog_id: 4] OR to Cloud Cloud[cloud_domain: 1 -- cloud_id: 1]
```

Fig. 5: Recommendation output

4 Conclusion

This paper presented the design of a business model for IoT applications that would be deployed on top of cloud and fog resources. This business model features 2 constructs, flow (specialized into data and collaboration) and placement (specialized into processing and storage). The paper also presented the development of a system built-upon a deep learning model that recommends how the different flows and placements should be formed. These recommendations consider the technical capabilities of cloud and fog resources as well as the networking topology connecting these resources to things. In term of future work, we would like to complete the design of the current system by including the collaboration flow and storage placement and to examine the impact of other criteria like privacy on the collaboration flow.

References

1. Al-Khafajiy, M., Baker, T., Al-Libawy, H., Maamar, Z., Aloqaily, M., Jararweh, Y.: Improving Fog Computing Performance via Fog-2-Fog Collaboration. Future Generation Computer Systems **100** (2019)
2. Al-khafajiy, M., Webster, L., Baker, T., Waraich, A.: Towards Fog Driven IoT Healthcare: Challenges and Framework of Fog Computing in Healthcare. In: Proceedings of ICFNDS'2018. Amman, Jordan (2018)
3. Bonomi, F., Milito, R., Natarajan, P., Zhu, J.: Fog Computing: A Platform for Internet of Things and Analytics. In: Big Data and Internet of Things: A Roadmap for Smart Environments, Studies in Computational Intelligence. Cisco, Springer International Publishing (2014)
4. Cheikhrouhou, S., Kallel, S., Guidara, I., Maamar, Z.: Business Process Specification, Verification, and Deployment in a Mono-Cloud, Multi-Edge Context. Computer Science and Information Systems **17**(1) (2020)
5. Geissdoerfer, M., Savaget, P., Evans, S.: The Cambridge Business Model Innovation Process. In: Proceedings of GCSM'2017. Stellenbosch, South Africa (2017)
6. Hou, S., Zhao, S., Cheng, B., Cheng, Y., Chen, J.: Fragmentation and Optimal Deployment for IoT-aware Business Process. In: Proceedings of SCC'2016. San Francisco, USA (2016)
7. Khebbeb, K., Hameurlain, N., Belalab, F.: A Maude-based Rewriting Approach to Model and Verify Cloud/Fog Self-Adaptation and Orchestration. Journal of Systems Architecture **110** (November 2020)
8. Li, F.: The Digital Transformation of Business Models in the Creative Industries: A Holistic Framework and Emerging Trends. Technovation **92-93** (January 2020)
9. Maamar, Z., Baker, T., Faci, N., Ugljanin, E., Al-Khafajiy, M., Burégio, V.: Towards a Seamless Coordination of Cloud and Fog: Illustration through the Internet-of-Things. In: Proceedings of SAC'2019. Limassol, Cyprus (2019)
10. Mahmud, R., Ramamohanarao, K., Buyya, R.: Application Management in Fog Computing Environments: A Taxonomy, Review and Future Directions. ACM Computing Surveys **53**(4) (July 2020)
11. Nieves, E., Hernández, G., Gil González, A., Rodríguez-González, S., Corchado, J.: Fog Computing Architecture for Personalized Recommendation of Banking Products. Expert Systems with Applications **140** (2020)
12. Satyanarayanan, M., Bahl, P., Cáceres, R., Davies, N.: The Case for VM-based Cloudlets in Mobile Computing. IEEE Pervasive Computing **8**(4) (2009)
13. Thekkummal, N., Jha, D., Puthal, D., James, P., Ranjan, R.: Coordinated Data Flow Control in IoT Networks. In: Proceedings of ALGOCLOUD'2019. Munich, Germany (2019)
14. Varghese, B., Wang, N., Nikolopoulos, D., Buyya, R.: Feasibility of Fog Computing. arXiv preprint arXiv:1701.05451 (2017)
15. Wang, T., Zhang, G., Alam Bhuiyan, M., Liu, A., Jia, W., Xie, M.: A Novel Trust Mechanism based on Fog Computing in Sensor-Cloud System. Future Generation Computer Systems **109** (2020)
16. Yahya, F., Maamar, Z., Boukadi, K.: A Multi-Criteria Decision Making Approach for Cloud-Fog Coordination. In: Proceedings of AINA'2020. Caserta, Italy (2020)