Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 175 (2020) 10–19

Procedia Computer Science

www.elsevier.com/locate/procedia

The 17th International Conference on Mobile Systems and Pervasive Computing (MobiSPC)
August 9-12, 2020, Leuven, Belgium

A new Edge Architecture for AI-IoT services deployment

Olivier Debauche[a,b,*], Saïd Mahmoudi[a], Sidi Ahmed Mahmoudi[a], Pierre Manneback[a], Frédéric Lebeau[b,c]

[a] University of Mons - ILIA / Infortech, 20 Place du Parc, Mons 7000, Belgium
[b] University of Liège - GxABT, TERRA, Passage des déportés 2, Gembloux 5030, Belgium
[c] University of Liège - GxABT, BioDynE, Passage des déportés 2, Gembloux 5030, Belgium

Abstract

Artificial intelligence (AI) and Internet of things (IoT) have progressively emerged in all domains of our daily lives. Nowadays, both domains are combined in what is called artificial intelligence of thing (AIoT). With the increase of the amount of data produced by the myriad of connected things and the large wide of data needed to train Artificial Intelligence models, data processing and storage became a real challenge.

Indeed, the amount of data to process, to transfer by network and to treat in the cloud have call into question classical data storage and processing architectures. Also, the large amount of data generated at the edge has increased the speed of data transportation that is becoming the bottleneck for the cloud-based computing paradigms.

The post-cloud approaches using Edge computing allow to improve latency and jitter. These infrastructures manage mechanisms of containerization and orchestration in order to provide automatic and fast deployment and migration of services such as reasoning mechanisms, ontology, and specifically adapted artificial intelligence algorithms.

In this paper we propose a new architecture used to deploy at edge level micro services and adapted artificial intelligence algorithms and models.

© 2020 The Authors. Published by Elsevier B.V.
This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/)
Peer-review under responsibility of the Conference Program Chair.

Keywords: Edge computing; Edge Internet of Things; Edge Artificial Intelligence; Edge A2IoT architecture; Internet of Things; Artificial Intelligence; Edge AI-IoT architecture

1. Introduction

The increase of the amount of data produced by the myriad of connected things, and the amount of data to process, to transfer by network and to treat in the cloud computing question data storage and processing architectures. The large amount of data generated at the edge has increased the speed of data transportation that is becoming

* Corresponding author. Tel.: +32-65-374-059; fax: +32-71-140-095.
E-mail address: olivier.debauche@umons.ac.be

1877-0509 © 2020 The Authors. Published by Elsevier B.V.
This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/)
Peer-review under responsibility of the Conference Program Chairs.
10.1016/j.procs.2020.07.006

the bottleneck for the cloud-based computing paradigms[32]. Different strategies have been proposed to reduce the cloud load and reduce bandwidth requirements by reconciling user processing either on mobile devices or on computers' users. The post-cloud approaches appear after centralized cloud processing which allow to improve latency and jitter for immobile entities. However, they do not provide an adapted answer for mobile devices and local awareness. Edge resources allow to provide faster predictions compared to the use of cloud resources. A compromise between the rational use of the cloud for learning models while not transferring all the data and deployment at the edge level that constitutes a compromise between the amount of data to be transferred, bandwidth and performance.

Different ways of building models are used to process data using artificial intelligence and machine learning algorithms :

(1) The "Abstracted AI" uses machine learning to train models on data sets, processed natural language documents or encoded human expertise and express models in different ways such as inference rule sets, decision trees, neural networks, etc. Models allow then to infer from the sensors data, and guide the operation of the system [3];

(2) The "Centralized Approach" needs to send all produced data to the cloud in order to train the model and performing the inference. This way implies the availability of sufficient bandwidth and low latency. Moreover, it induces other problems such as data privacy, data volume storage, and the cost of network connectivity [3];

(3) The "Edge Approach" is less expensive because its bandwidth is efficient and sends less data to the cloud. In addition, it improves latency, the responsiveness, and the scalability [3].

Container technology offers a lightweight cloud environment to allows the deployment and the execution of applications in an isolated manner with a more density of application in a single host than virtual machine[30]. Applying advanced machine learning and deep learning to IoT data allows to get real-time prediction and provide advance analytics. In this context, edge computing techniques can be used to address the problems of latency, jitter, and to decrease the load processing on the cloud by bringing the processing to the producer or consumer level of the data. Moreover, combining IoT and AI at an edge level allows different applications such as image processing, face recognition, video processing, object segmentation, and object tracking, etc. [25].

Our contribution is an edge heterogeneous architecture which allows to :

1. process data with micro services in order to to annotate, analyze, detect data abnormalities and validate data, etc.
2. select the appropriate, pertinent and required features for training artificial intelligence algorithms and models in the cloud,
3. achieved a predictive and/or prescriptive analysis at the edge level using adapted AI algorithms and models.

We use use Kubernetes containers orchestration to manage dynamically Docker images and select specific node accordingly to algorithms requirements

The remaining of this paper is organized as follow : In section 2, we provide on one hand the summary of the major contribution in the field of study and on the other hand, we describe this contribution in our previous works. The section 3 presents our heterogeneous micro cluster used to process data, and exploit adapted artificial intelligence algorithms for the prediction. In the fourth section, we experiment performances of our architecture on the recognition of plant disease on the basis of the leaf image. The same experimentation is done on the cloud and performances of both approaches are compared. In section 5, we discuss the obtained results with the different artificial intelligence algorithms. Finally in section 6, we draw our conclusion and describe our future work.

2. Literature Review

Many authors have worked on innovative solutions allowing to bring the treatment at the edge of the network.

2.1. Related works

Manco et al. developed an architecture based on Xen hypervisor 4.4, MiniOS and built around the concept of massive consolidation in which an heterogeneous virtualization platform is deployed at different levels of the network (access, aggregation, cloud). This architecture uses minimalistic Virtual Machines (VM) that can be easily and quickly deployed and migrated at various levels of the network under 100 milliseconds. This massively distributed and hierarchical architecture can deploy up to 10,000 specialized virtual machines on a server AMD Opteron 6376@2.3GHz with 128 GB of RAM. Virtual Machines can also be deployed at various points of the network on micro servers such as Cubietruck, Raspberry Pi, Intel NUCs, Intel Edison, AMD Gizmos and can create on-the-fly services [27]. We use inside our architecture docker because containerization is lighter than virtualization in terms of consumption of system resources. In another work, Gazetti et al. proposed a streamed container deployment. In this environment each node is present in a location (located in a same vicinity) and acts as a gateway and interact with the remote cloud infrastructure. The gateway pulls images for the entire of the location and provides information about status of nodes inside that place. The coordination inside the locality is assured by a message broker while a gateway manager operates dynamically the peer-to-peer distribution graph [21]. The proposed solution is efficient even in case of network unavailability with the cloud where the images cannot be brought back. Thus, we store docker images in repositories replicated at different level of the network and externally to be independent of our cloud architecture. For example, Calo et al. proposed an edge deployment system using a framework that implement a set of REST APIs for automating the deployment of Machine learning algorithms on the edge in 4 phases: (1) In the Discover phase, a user upload training data, label them and then test multiple machine learning models on the cloud. At the end, the better models are selected; (2) During the deployment phase, the best models are retrieved, serialized, and packaged as Docker image before storage in repositories shared with edge components; (3) In the Operate phase, Docker containers are deployed and executed on edge device upon request of the user; (4) The retraining phase capture information about the effectiveness of algorithms deployed and transfer them to the cloud in order to allow learning and continuous improvement [3]. We adopt a similar strategy but in our case data are automatically upload on the cloud without intervention from the users. Recently, Kousiouris et al. have proposed a solution based on three modules: (1) Node-Red is used as a UI implementation layer, plugin mechanism with existing IoT management platform, middleware which allows to adapt and modify a data source to another; (2) A semantic service block which map data feeds with semantic concepts; (3) An artificial Intelligence block using TensorFlow which detect deviation of normal behavior by comparison of real data with a prediction model established for an expected duration. They use Docker containerization and Docker Swarm to manage the cluster [26]. Docker Swarm is generally used in development, but it is simple and deploys fastly. However, Kubernetes is preferred to Docker Swarm for production environment because it supports more demand and more complexity, and it is best adapted for production environment.

On the other hand, Teerapittayanon et al. have shown the capability of heterogeneous physical devices located in the cloud, in the edge, and geographically distributed IoT end-devices to train a Distributed Deep Neural Network (DDNN). Some shallow portions of Deep Neural Networks (DNN) are mapped onto a distributed computing hierarchy to allows fast and localized inference [34]. All AI algorithms cannot be adapted in same manner. In an other work Mittal el al. have presented a survey that investigates deep learning algorithms optimizations and their implementation on NVIDIA Jetson: (1) The reduction of the number of frame by filtering and sampling to send only interesting frames; (2) Reducing image resolution; (3) Reduction CNN size using knowledge distillation where a small network is trained to imitate the output of an original larger network; (4) Using lightweight CNNs that trade-offs accuracy for reducing CNN latency and memory demands; (5) Choosing the optimal CNN for a metric such as top-1, top-5 [28]. We use several techniques proposed by theses authors to adapt our AI algorithms for the edge.

2.2. Background

In our previous paper, we have progressively developed and validated a semantic driven and modular cloud centric Lambda Architecture[18] through various uses cases: landslides monitoring [29], bee health [19], smart poultry [10], climatic enclosure[12], irrigation [5], elderly and patient monitoring [13], bird nesting [1], smart campus [4], smart home [8], smart city [9], smart building [16], cattle behavior [7] [15][6][11], phenotyping [14][17], urban gardening [2]. In parallel works, we proposed a Cloud Environment to Deploy Deep Learning Containerized Applications using

Kubernetes (k8s) and Slurm [20]. We studied also the requirement needs of each application and based on this analysis, we configured our Cloud environment to affect training for GPU and inference for CPU. In this paper we extend previous paper with an Edge AIoT (Artificial Intelligence and Internet of Things) Architecture which complete and collaborate with our cloud centric architecture previously.

3. Proposed Method

The proposed edge architecture is deployed on a set on interconnected device composed of three components :

(1) three Nvidia Jetson Nano (472 GFLOPS) with 128-core CUDA Maxwell, a Quad-core ARM A57@1.43 GHz, 4GB 64-bit LPDDR4@25.6 GB/s equipped with an additional M.2 SSD by means of an adapter M.2 Key-E to Mini PCI-E adapter. Jetson Nano use JetPack 4.3 containing (NVIDIA L4T[1], TensorRT[2], cuDNN[3], CUDA[4], OpenCV[5]) and installed on a ScanDisk Extreme 32GB Micro SDHC UHS-3. The aims of Jetson Nano is to train AI models on collected IoT data;

(2) four Odroid N2 with integrates a quad-core ARM Cortex-A73 CPU cluster and a dual core Cortex-A53 cluster, and a new generation Mali-G52 GPU which process data on the Edge;

(3) a 8-port Gigabit Network Switch manages communications between all elements of the heterogeneous cluster (Jetson Nano, Odroid N2).

The coupling of Jetson Nano specifically developed for Artificial Intelligence with Odroid N2 offers better performance than Raspberry Pi 4 and allows to create a heterogeneous cluster. This cluster is used to deploy by means of Kubernetes on one hand micro services (i.e.: to enrich and annotate with specialized data ontology, validate, and detect abnormal conditions with trained on the cloud artificial intelligence models) data collected from IoT, and on the other hand, to deploy and train AI models in order to provide predictive or prescriptive analysis.

Kubernetes (k8s)[6] is an open source container orchestration platform which manage containerized workloads and services at scale. Kubernetes 1.17 used in conjunction with Docker-CE 19.03 to deploy and manage containers in pods. A pod is the smallest unit of deployment in a k8s cluster in which runs one or more containers. The two main types of nodes in the Kubernetes cluster are the master node and worker nodes.

The Fig. 1 provides a high level scheme of our architecture. It is composed from one Odroid N2 4GB master node and six worker nodes (three Odroid N2 4GB and three Nvidia Jetson Nano). Micro services are deployed on Odroid N2 while Jetson Nano are used to train, validate, and deploy AI models.

The **Master Node**, a Odroid N2 is responsible of all application workload deployment, scheduling, and placement decision as well as detecting and managing changes to the state of deployed applications. The master Node is composed of:

- (1) etcd[7] a highly reliable, distributed, and consistent Key-Value Store, written in go, for persisting and maintaining stat information about the various components of the k8s cluster;
- (2) API Server exposes endpoints for all interactions with and in the k8s cluster;
- (3) Scheduler planes in function of application resource requirements and specific affinity constraints which application pod(s) to run on the selected worker node(s) of the k8s cluster;
- (4) Controller Manager composed of :

    1. *Node Controller* responsible for monitoring and detecting the state of health of the worker node(s);

[1] https://developer.nvidia.com/embedded/linux-tegra
[2] https://developer.nvidia.com/tensorrt
[3] https://developer.nvidia.com/cudnn
[4] https://developer.nvidia.com/cuda-zone
[5] https://opencv.org
[6] https://kubernetes.io
[7] https://github.com/etcd-io/etcd

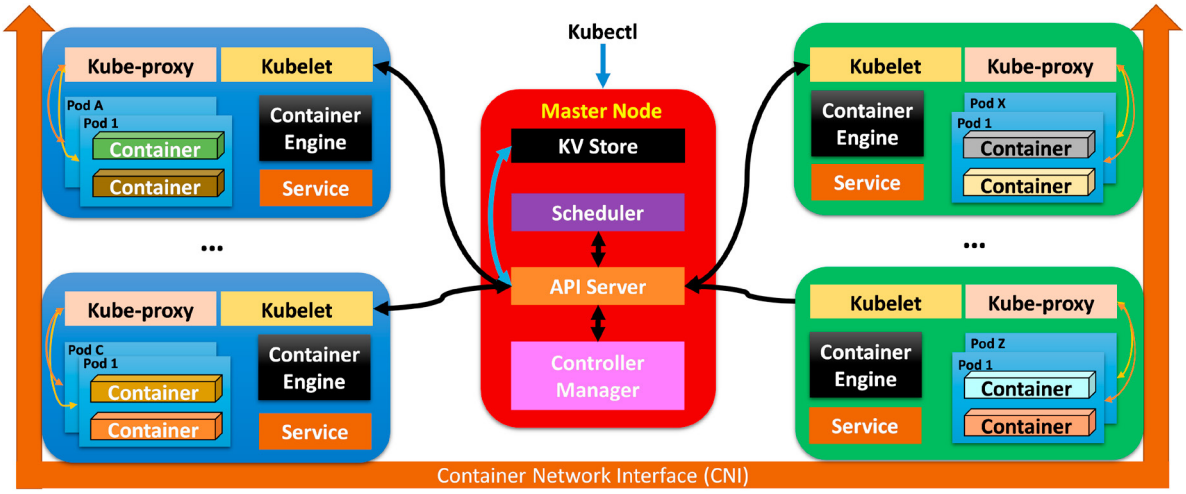


Fig. 1: High Level Scheme of our architecture

2. *ReplicaSet* responsible of maintaining the desired number of pod replicas in the cluster;
3. *Endpoints Controller* responsible for detecting and managing changes to the application service access endpoints.

The master node will also be responsible to assign the application to a chosen node depending of its need. There are two ways to configure our node: The first, by using the pod configuration file to describe which node will be responsible to schedule our application. The second way, by using the command line, and specifying the label of the specific node. Our goals is to provides an automatic system to deploy micro services and neural network dynamically on a cluster place at the edge of the network.

Worker Nodes mix Odroid N2 and Nvidia Jetson Nano are composed of:

- (1) Kubelet, an agent that runs on each worker of k8s. It creates and starts an application pod on the worker and monitors the state of health of worker and all running pods to the master node via the API server;
- (2) Kube-proxy, a network proxy that is an entry point allowing an access to various application service endpoints and routes request to the appropriate pod(s) in the cluster;
- (3) a Container Engine Docker manages the life cycle of containers such as getting the images, starting and stopping containers, etc.

Finally, **kubectl** is a command line tool used by administrators or operators for deploy, scales applications, and manages the k8s cluster.

The Fig. 2 shows the different activities achieved at each level of the data acquisition and processing chain.

Cloud Servers host multiple activities :

- (1) Ontology repository,
- (2) Knowledge Base deduced from ontology by the Semantic Reasoner.
- (3) MQTT Broker used by edge micro cluster to communicate with the cloud server.
- (4) containers Orchestrator that manages containers deployed locally and on edge devices forming the micro cluster.

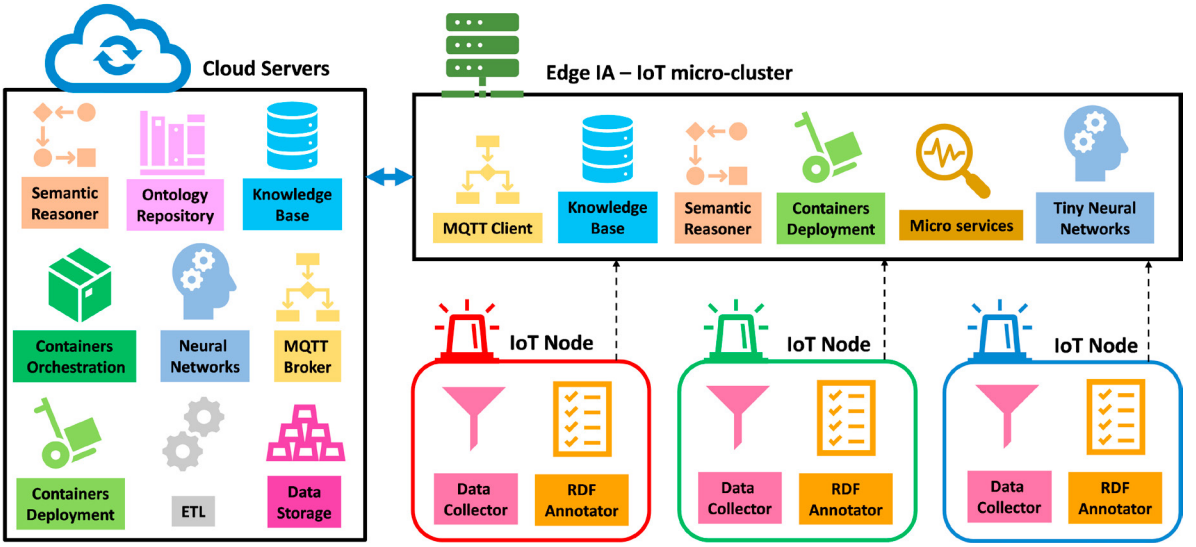


Fig. 2: High Level Scheme of our architecture

- (5) Neural Networks trained from features extracted by tiny neural networks hosted on Jetson Nano;
- (6) Extraction Transformation Load (ETL) process raw data coming from IoT Note before them storage in database.

Edge IA-IoT micro-cluster hosts :

- (1) MQTT client which allows the communication with the Cloud Server;
- (2) Local Knowledge Based associated with a local Semantic Reasoner do other deductions than the cloud;
- (3) Tiny Neural networks are deployed after compression on TensorFlow Lite to predict event, and also used to extract data. In this case, lower Layers of the Distributed Neural Network are extracted from the original model and distributed on Jetson Nano to extract features or fusion sensors data. Resulting data are then transmitted to the cloud where the rest of Neural Network (*NN*) train on the data received from Edge Nodes;
- (4) Micro services are deployed to validate, cure, compress data;
- (5) The container engine manages the local deployment of micro services and Neural Network algorithms.

Finally, **IoT Nodes** achieve data collection from sensors and their annotation with the RDF annotator.

4. Experimentation

In order to demonstrate the usability of our architecture we have achieved a comparison between prediction performance of several AI algorithms between edge and cloud.

The PlantVillage [24] dataset has been used for our experimentations. It contains 54,303 healthy and unhealthy leaf images divided into 38 categories by species and disease. We have trained the raw dataset with MobileNet[22], MobileNet V2 [31], DenseNet121 [23], NasNetMobile [35], EfficientNet-B0 [33], EfficientNet-B1 [33], EfficientNet-B2 [33] in the cloud on servers equipped with a Tesla P100. All models are trained with SGD optimizer with a learning rate (lr) value of 0.0001, an early stopping with a patience of 5 and a minimum delta of 0.001 on the monitored parameter: validation accuracy (val_acc). Trained models have been converted to TensorFlow Lite and containerized to be deployed on Jetson Nano in order to evaluate their performances on the prediction test.

Table 1: Comparison of classifiers trained in the cloud

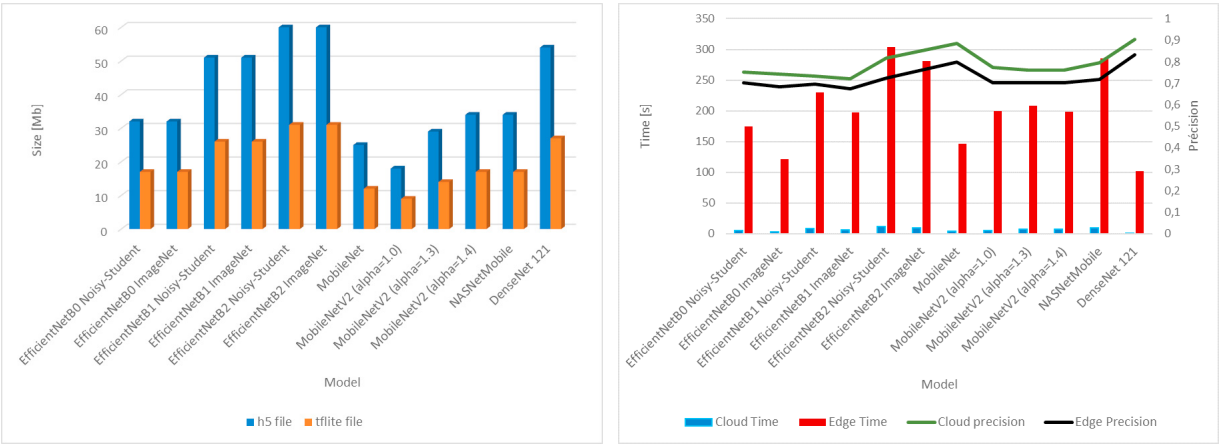
Model	Image Size	Weight	Epochs	h5 Size (Mb)	Total Params	Trainable Params	Non Trainable Params	Loss	Accuracy	Value Loss	Val Accuracy
EfficientNet-B0	224x224	Noisy-Student	32	32	4,098,394	4,056,302	42,092	0.0070	0.9988	9.91e-07	0.9988
EfficientNet-B0	224x224	ImageNet	29	32	4,098,394	4,056,302	42,092	0.0063	0.9991	3.80e-04	0.9984
EfficientNet-B1	240x240	Noisy-Student	32	51	6,624,062	6,561,938	62,124	0.0061	0.9991	1.33e-04	0.9984
EfficientNet-B1	240x240	ImageNet	29	51	6,624,062	6,561,938	62,124	0.0059	0.9991	7.89e-04	0.9980
EfficientNet-B2	260x260	Noisy-Student	29	60	7,822,256	7,754,612	67,644	0.0052	0.9993	1.43e-04	0.9989
EfficientNet-B2	260x260	ImageNet	32	60	7,822,256	7,754,612	67,644	0.0051	0.9993	4.79e-04	0.9989
MobileNet (α=1.0)	224x224	ImageNet	35	25	3,267,966	3,246,002	21,964	0.0041	0.9996	1.01e-04	0.9986
MobileNetV2 (α=1.0)	224x244	ImageNet	17	18	2,306,814	2,272,626	34,188	0,0071	0,9991	1.56e-07	0.9934
MobileNetV2 (α=1.3)	224x244	ImageNet	39	29	3,829,470	3,785,138	44,332	0.0029	0.9996	8.92e-03	0.9979
MobileNetV2 (α=1.4)	224x244	ImageNet	21	34	4,431,998	4,383,986	48,012	0.0078	0.9990	5.01e-04	0.9960
NASNetMobile	224x224	ImageNet	16	34	4,310,034	4,273,220	36,814	0.0094	0.9986	0.0037	0.9962
DenseNet121	224x224	ImageNet	23	54	7,076,606	6,992,882	83,724	0.0063	0.9992	0.1147	0.9982

Table 2: Comparison of classifiers prediction time

Model	Weight	h5 File Size (Mb)	TFLite Size (Mb)	Cloud prediction Time (s)	Cloud Prediction Test	Edge prediction Time (s)	Edge prediction Test
EfficientNet-B0	Noisy-Student	32	17	5.59	0.7515	174.87	0.7031
EfficientNet-B0	ImageNet	32	17	2.93	0.7421	121.11	0.6822
EfficientNet-B1	Noisy-Student	51	26	8.14	0.7335	229.71	0.6975
EfficientNet-B1	ImageNet	51	26	6.44	0.7223	197.61	0.6752
EfficientNet-B2	Noisy-Student	60	31	11.89	0.8161	303.99	0.7234
EfficientNet-B2	ImageNet	60	31	9.64	0.8519	281.54	0.7624
MobileNet (α=1.0)	ImageNet	25	12	4.57	0.8853	146.52	0.7995
MobileNetV2 (α=1.0)	ImageNet	18	9	5.22	0.7737	199.50	0.7012
MobileNetV2 (α=1.3)	ImageNet	29	14	7.16	0.7615	208.14	0.7028
MobileNetV2 (α=1.4)	ImageNet	34	17	7.08	0.7603	198.75	0.7015
NASNetMobile	ImageNet	34	17	9.88	0.7960	285.75	0.7168
DenseNet121	ImageNet	54	27	1.07	0.9031	102.46	0.8311

5. Results and Discussion

The Table 1 presents information related of training of several AI algorithms : EfficientNet B-0, B-1 & B-2 using weight of Imagenet or Noisy Student, and MobileNet, MobileNetV2, NASNetMobile, DenseNet121 with ImageNet weight on the cloud. The analysis of Table 1 shows that there is not significant difference in term of accuracy and val accuracy. The analysis of Table 2 shows that DenseNet121 provides the better accuracy with the minimum time of prediction on the cloud and on the edge. This accuracy can be explained but EfficientNet usually provides better performances on black and white images while MobileNet and MobileNetV2 are shallow models and therefore less efficient in terms of precision.



(a) Comparison of h5 and tflite file size

(b) Comparison of prediction time and accuracy

Fig. 3: Comparison between Cloud and Edge

The Figure 3a shows a comparison between standard trained models (h5) and compressed ones (TFlite) file size. The Figure 3b presents also the validation test time expressed in seconds and precision obtained with h5 model on the cloud and TFlite model on Jetson Nano. Our prediction test has been achieved on 443 images. In the end, Jetson Nano is 28 more slow than the cloud server equipped with a Tesla P100.

6. Conclusion and future works

In this paper, we described a new Edge AIoT architecture collaborating with the cloud able to deploy micro services; in addition to deploy Distributed Neural Network (DNN) or specific AI algorithms.

The proposed architecture was analyzed using Artificial Intelligence algorithms and models that were trained on a cloud server, equipped with Tesla P100 card. Then, a comparative study was realized in terms of models performance (inference time) and precision using a test data-set of 443 images. Experimental results showed that the Jetson Nano is 28 more slow than Tesla P100 and accuracy decrease approximately of 5%.

In future works, we plan to extend our architecture to allows collaboration between many Edge Clusters and the dynamic repartition of load. Indeed, multiple k8s clusters in the vicinity can also federated in one cluster of clusters namely a Kubernetes Cluster Federation (KubeFed) in order to improve the resiliency of the service. Hence, it is possible to federate Edge AIoT cluster by means of a federation application program interface (API) server which interacts with the cluster through the federation controller manager. The federator master exposes the API, schedule, and manage overall cluster. The interaction with the Kubernetes cluster is done through the Federation Controller manager which use the federation API to automatically manages the subordinate Kubernetes cluster by providing an API equivalent to the k8s API.

Acknowledgements

We would especially like to thank Mr Adrinao Guttadauria for his technical support and for setting up all the electronic systems and computing systems necessary for carrying out this research. The authors would linke to thanks also Meryem Elmoulat for the English editing of this paper.

References

[1] Ait Abdelouahid, R., Debauche, O., Mahmoudi, S., Manneback, P., 2020. Smart birds, in: 2020 International Conference on Advanced Communication Technologies and Networking (CommNet), pp. 1–9.

[2] Ait Abdelouhahid, R., Debauche, O., Mahmoudi, S., Marzak, A., Manneback, P., Lebeau, F., 2020. Open phytotron: A new iot device for home gardening, in: 2020 5th International Conference on Cloud Computing Technologies and Applications (Cloudtech), pp. 1–7.

[3] Calo, S.B., Touna, M., Verma, D.C., Cullen, A., 2017. Edge computing architecture for applying ai to iot, in: 2017 IEEE International Conference on Big Data (Big Data), IEEE. pp. 3012–3016.

[4] Debauche, O., Ait Abdelouahid, R., Mahmoudi, S., Manneback, P., 2020. Smart campus, in: 2020 International Conference on Advanced Communication Technologies and Networking (CommNet), pp. 1–9.

[5] Debauche, O., El Moulat, M., Mahmoudi, S., Manneback, P., Lebeau, F., 2018. Irrigation pivot-center connected at low cost for the reduction of crop water requirements, in: 2018 International Conference on Advanced Communication Technologies and Networking (CommNet), pp. 1–9. doi:10.1109/COMMNET.2018.8360259.

[6] Debauche, O., Mahmoudi, S., Andriamandroso, A., Manneback, P., Bindelle, J., Lebeau, F., 2018. Cloud services integration for farm animals' behavior studies based on smartphones as activity sensors. Journal of Ambient Intelligence and Humanized Computing URL: https://doi.org/10.1007/s12652-018-0845-9, doi:10.1007/s12652-018-0845-9.

[7] Debauche, O., Mahmoudi, S., Andriamandroso, A., P., M., J., B., Lebeau, F., 2017. Web-based cattle behavior service for researchers based on the smartphone inertial central. Procedia Computer Science 110, 110 – 116. URL: http://www.sciencedirect.com/science/article/pii/S1877050917313066, doi:https://doi.org/10.1016/j.procs.2017.06.127. 14th International Conference on Mobile Systems and Pervasive Computing (MobiSPC 2017) / 12th International Conference on Future Networks and Communications (FNC 2017) / Affiliated Workshops.

[8] Debauche, O., Mahmoudi, S., Belarbi, M.A., El Adoui, M., Mahmoudi, S.A., 2018a. Internet of things: Learning and practices. application to smart home, in: 2018 International Conference on Advanced Communication Technologies and Networking (CommNet), pp. 1–6. doi:10.1109/COMMNET.2018.8360247.

[9] Debauche, O., Mahmoudi, S., Mahmoudi, S.A., 2018b. Internet of things: learning and practices. application to smart city, in: 2018 4th International Conference on Cloud Computing Technologies and Applications (Cloudtech), pp. 1–7. doi:10.1109/CloudTech.2018.8713337.

[10] Debauche, O., Mahmoudi, S., Mahmoudi, S.A., Manneback, P., Bindelle, J., Lebeau, F., 2020a. Edge computing and artificial intelligence for real-time poultry monitoring. Procedia Computer Science The 17th International Conference on Mobile Systems and Pervasive Computing (MobiSPC) / The 15th International Conference on Future Networks and Communications (FNC 2020) / Affiliated Workshops.

[11] Debauche, O., Mahmoudi, S., Mahmoudi, S.A., Manneback, P., Bindelle, J., Lebeau, F., 2020b. Edge computing for cattle behavior analysis, in: 2020 Second international conference on Embedded Distributed Systems (EDiS), pp. 1–5.

[12] Debauche, O., Mahmoudi, S., Mahmoudi, S.A., Manneback, P., Lebeau, F., 2020c. Edge computing and artificial intelligence semantically driven. application to a climatic enclosure. Procedia Computer Science The 17th International Conference on Mobile Systems and Pervasive Computing (MobiSPC) / The 15th International Conference on Future Networks and Communications (FNC 2020) / Affiliated Workshops.

[13] Debauche, O., Mahmoudi, S., Manneback, P., Assila, A., 2019. Fog iot for health: A new architecture for patients and elderly monitoring. Procedia Computer Science 160, 289 – 297. URL: http://www.sciencedirect.com/science/article/pii/S1877050919317880, doi:https://doi.org/10.1016/j.procs.2019.11.087. the 10th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN-2019) / The 9th International Conference on Current and Future Trends of Information and Communication Technologies in Healthcare (ICTH-2019) / Affiliated Workshops.

[14] Debauche, O., Mahmoudi, S., Manneback, P., Massinon, M., Tadrist, N., Lebeau, F., Mahmoudi, S.A., 2017. Cloud architecture for digital phenotyping and automation, in: 2017 3rd International Conference of Cloud Computing Technologies and Applications (CloudTech), pp. 1–9. doi:10.1109/CloudTech.2017.8284718.

[15] Debauche, O., Mahmoudi, S., Manneback, P., Tadrist, N., Bindelle, J., Lebeau, F., 2017. Improvement of battery life of iphones inertial measurement unit by using edge computing application to cattle behavior, in: 2017 Symposium International sur les Sciences Informatiques et Applications (ISCSA2017), pp. 1–5.

[16] Debauche, O., Mahmoudi, S., Moussaoui, Y., 2020. Internet of things learning: a practical case for smart building automation, in: 2020 5th International Conference on Cloud Computing Technologies and Applications (Cloudtech), pp. 1–7.

[17] Debauche, O., Mahmoudi, S.A., De Cock, N., Mahmoudi, S., Manneback, P., Lebeau, F., 2020. Cloud architecture for plant phenotyping research. Concurrency and Computation: Practice and Experience n/a, e5661. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.5661, doi:10.1002/cpe.5661, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpe.5661. e5661 cpe.5661.

[18] Debauche, O., Mahmoudi, S.A., Mahmoudi, S., Manneback, P., 2018a. Cloud platform using big data and hpc technologies for distributed and parallels treatments. Procedia Computer Science 141, 112–118.

[19] Debauche, O., Moulat, M.E., Mahmoudi, S., Boukraa, S., Manneback, P., Lebeau, F., 2018b. Web monitoring of bee health for researchers and beekeepers based on the internet of things. Procedia Computer Science 130, 991 – 998. URL: http://www.sciencedirect.com/science/article/pii/S1877050918304654, doi:https://doi.org/10.1016/j.procs.2018.04.103. the 9th International Conference on Ambient Systems, Networks and Technologies (ANT 2018) / The 8th International Conference on Sustainable Energy Information Technology (SEIT-2018) / Affiliated Workshops.

[20] Doukha, R., Mahmoudi, S., Zbakh, M., Manneback, P., 2020. Cloud environment to deploy deep learning containerized applications, in: 2020 5th International Conference on Cloud Computing and Artificial Intelligence: Technologies and Applications (CloudTech), pp. 1–6.

[21] Gazzetti, M., Reale, A., Katrinis, K., Corradi, A., 2018. Scalable linux container provisioning in fog and edge computing platforms, in: Heras, D.B., Bougé, L., Mencagli, G., Jeannot, E., Sakellariou, R., Badia, R.M., Barbosa, J.G., Ricci, L., Scott, S.L., Lankes, S., Weidendorfer, J. (Eds.), Euro-Par 2017: Parallel Processing Workshops, Springer International Publishing, Cham. pp. 304–315.

[22] Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H., 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 .

[23] Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q., 2017. Densely connected convolutional networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4700–4708.

[24] Hughes, D., Salathé, M., et al., 2015. An open access repository of images on plant health to enable the development of mobile disease diagnostics. arXiv preprint arXiv:1511.08060 .

[25] Katare, G., Padihar, G., Qureshi, Z., . Challenges in the integration of artificial intelligence and internet of things. International Journal of Systems and Software Engineering 6, 10–15.

[26] Kousiouris, G., Tsarsitalidis, S., Psomakelis, E., Koloniaris, S., Bardaki, C., Tserpes, K., Nikolaidou, M., Anagnostopoulos, D., 2019. A microservice-based framework for integrating iot management platforms, semantic and ai services for supply chain management. ICT Express 5, 141–145.

[27] Manco, F., Martins, J., Yasukata, K., Mendes, J., Kuenzer, S., Huici, F., 2015. The case for the superfluid cloud, in: 7th {USENIX} Workshop on Hot Topics in Cloud Computing (HotCloud 15), pp. 1–6.

[28] Mittal, S., 2019. A survey on optimized implementation of deep learning models on the nvidia jetson platform. Journal of Systems Architecture 97, 428 – 442. URL: http://www.sciencedirect.com/science/article/pii/S1383762118306404, doi:https://doi.org/10.1016/j.sysarc.2019.01.011.

[29] Moulat, M.E., Debauche, O., Mahmoudi, S., Brahim, L.A., Manneback, P., Lebeau, F., 2018. Monitoring system using internet of things for potential landslides. Procedia Computer Science 134, 26 – 34. URL: http://www.sciencedirect.com/science/article/pii/S1877050918311037, doi:https://doi.org/10.1016/j.procs.2018.07.140. the 15th International Conference on Mobile Systems and Pervasive Computing (MobiSPC 2018) / The 13th International Conference on Future Networks and Communications (FNC-2018) / Affiliated Workshops.

[30] Puliafito, C., Mingozzi, E., Longo, F., Puliafito, A., Rana, O., 2019. Fog computing for the internet of things: A survey. ACM Transactions on Internet Technology (TOIT) 19, 1–41.

[31] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C., 2018. Mobilenetv2: Inverted residuals and linear bottlenecks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4510–4520.

[32] Shi, W., Cao, J., Zhang, Q., Li, Y., Xu, L., 2016. Edge computing: Vision and challenges. IEEE Internet of Things Journal 3, 637–646. doi:10.1109/JIOT.2016.2579198.

[33] Tan, M., Le, Q.V., 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. arXiv preprint arXiv:1905.11946 .

[34] Teerapittayanon, S., McDanel, B., Kung, H.T., 2017. Distributed deep neural networks over the cloud, the edge and end devices, in: 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), pp. 328–339. doi:10.1109/ICDCS.2017.226.

[35] Zoph, B., Vasudevan, V., Shlens, J., Le, Q.V., 2018. Learning transferable architectures for scalable image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 8697–8710.