# A Practical way to Handle Service Migration of ML-based Applications in Industrial Analytics

Domenico Scotece†, Claudio Fiandrino⋆, Luca Foschini†
†Department of Computer Science and Engineering, University of Bologna, Bologna, Italy
⋆IMDEA Networks Institute, Madrid, Spain
Email:†{domenico.scotece, luca.foschini}@unibo.it
⋆claudio.fiandrino@imdea.org

***Abstract*—Nowadays, Machine learning (ML) plays a significant role in Industrial Analytics. It enables predictive analytics, and helps uncovering essential insights to transform industries. As a result, real-time data analytics has become an essential requirement for industrial engineering jobs. Edge computing enables local intelligence and real-time analytics that are key for industry processes to take autonomous decisions locally at the edge of the network. However, outages in edge datacenters can jeopardize the whole plant security. In this paper, we proposed a practical approach to effectively handling service and data migration of ML-based applications in Industrial Analytics scenarios in the presence of a lack of computing resources at the edge. We argue that in this context the value of data is inversely proportional to their age and is very important to work with fresher data. In this paper, we describe our architectural approach for service and data handoff and show a predictive diagnostics case study deployed in an edge-enabled IIoT infrastructure. We evaluate our proposed approach in terms of drop of accuracy in a well-known edge computing emulator, i.e., openLEON. The experimental results show the benefit of our solution with respect to standard techniques.**

***Index Terms*—Edge computing, Computation Offloading, Service Migration, Industrial Analytics**

## I. Introduction

The Industrial Internet of Things (IIoT) aims at connecting industrial assets and machines (i.e., things) to enterprise information systems, business processes, and people who operate and use them. Advanced analytics is at the core of this next-generation level of integration and, when applied to machine and process data, provides new insights and intelligence to optimize decision-making significantly and enable intelligent operations leading to transformational business outcomes and social value [1]. Machine learning (ML) plays a significant role in Industrial Analytics, enabling predictive analytics, and uncovering essential insights to transform industries. With the technological advances of computing and communication technologies, ML enables data analytics on massive quantities of data such as those produced by an IIoT-based system and can use the extracted knowledge (e.g., trained models, uncover patterns) to aid real-time decision-making in complex situations. Fault detection and isolation in industrial processes [2], real-time quality monitoring in additive manufacturing [3], and automatic fruit classification [4] are examples of using ML in IIoT-based Industry 4.0 systems.

978-1-6654-3540-6/22 

In the recent years, edge computing has been considered an important enabler for the industry scenario [5]. A number of new applications have emerged all calling for stringent requirements such as intensive computation and tight latency conditions imposed by Industrial Analytics scenario. Edge computing is key to meet such stringent constraints by pooling computing resources closer to the end user and not in the cloud. Such concept resulted into various paradigms, including fog computing, mist computing [6], and Multi-Access Edge Computing (MEC) [7]. The latter was standardized by the European Telecommunications Standards Institute (ETSI) [8] and is specifically tailored as an enabler for the 5th Generation (5G) mobile networks and beyond.

In this paper, we consider a sub-problem of the service and data migration. Specifically, we tackle the problem of moving computation and data of ML processes upon failures in an edge datacenter. Especially, we considered the migration of ML tasks for pre-trained models in order to provide intelligence services for the Industry scenario. Unlike in our previous work that proposed efficient service migration strategies [9], in this work we focused on the accuracy of the ML tasks during handoff. The proposed solution has the following primary innovation elements and features.

- *First*, we present the system architecture that enables ML-based tasks migration applied to shop floor data retrieved from processes, resources, and products for process optimization, quality inspection, and preventive diagnostics.
- *Second*, in our migration model data is moved in a reactive way to guarantee that fresher data are moved during the handoff process. This allows reaching a satisfactory level of accuracy because working with the latest data is very important in this scenario [10].
- *Third*, ML-based services and pre-trained models are proactively migrated to neighbor edge datacenters thanks to standard orchestrator tools like Kubernetes.
- *Fourth*, the handoff triggering works in a proactive way and could be based on different metrics such as memory or CPU consumption.
- *Finally*, we present a real use case for predictive diagnostics based on an open-source dataset for a chemical detection platform which is considered a hot topic for IIoT applications. To demonstrate the benefits of the proposed service and data migration system for ML-based tasks we quantitatively evaluate the accuracy of the processing predictive diagnostics during

the handoff in the case of the edge datacenter outage.

In a nutshell, the flow of the paper is as follows. First, we provide our motivation for this study and related works (Section II). Then, we present our service and data handoff system that are applicable to tasks that are sensitive to fresher data like predictive maintenance in IIoT scenarios (Section III) and its implementation details (Section IV). Moreover, we benchmark the proposed migration system with openLEON emulator [11] (Section V). Finally, we summarize and conclude our work (Section VI).

## II. Related Work and Motivation

This section provides background information for the involved technologies and paradigms and briefly introduces the research directions of the literature in the areas of edge computing service migration at the edge. Finally, we provide motivations for service and data migration in the Industry Analysis scenario.

### A. Related Work

**Service Relocation**. In the last decade, with the rise of the edge computing paradigm, the problem of service relocation has been studied in the literature from different angles. The aim of service placement is to reduce service access delay, reduce communication delay, and achieve less network latency [12]. Work in [13] studied the service relocation issue with the goal to optimize the Quality-of-Service (QoS). Different approach in [14] that addressed the same problem with the aim at reducing energy consumption. For a most recent and very comprehensive survey on service placement please refer at [15].

Our work is related to the service relocation problem specifically designed for ML-based services. In recent years, the use of ML at the edge of the network led to the advent of Edge Intelligence (EI) [16]. The idea behind EI is to design models that cost less in terms of resources because edge infrastructures have limited resource capabilities [17]. Few works have been started to explore service relocation in this field. The work proposed in [18] investigated the tradeoff between accuracy and latency for offloading decisions regarding deep learning services for ensuring optimal QoS. Other works investigated the service placement problem specifically for EI models. Zehong *et al.* [19] studied the problem of optimally placing EI services with the objective of optimizing energy consumption and service completion time. Our work departs from the literature in that we focus on EI service migration where an intelligence service has to be relocated at new host due to a handoff. To the best of our knowledge, this is the first work to do so.

**ML and Service Migration**. Here we discuss the works related to exploit ML to optimize service migration and how to migrate services in ML pipelines.

ML can be used to optimize service migration in its entirety, including placement or selecting the best migration algorithm [20]. Several works have been studied in the literature for predicting node failure using ML techniques. Farahnakian *et al.* [21] proposed a Linear Regression ML technique to migrate virtual machines (VMs) by predicting future CPU usage of physical nodes based on usage history. Furthermore, they proactively trigger migration in case of foreseen resource shortage and Service Level Agreement violations. Other works used Decision Tree Learning to predict future host states, for instance, the work proposed by Li *et al.* [22] leverages a classifier based on a Binary Decision Tree to perform VM migration and reduce energy consumption. On the contrary, Jeong *et al.* [23] analyzed application logs to detect failure events. Specifically, they proposed a proactive migration algorithm for virtual Evolved Packet Core based on a pre-trained model for failure prediction. Deep Learning was used to train the model and Long short-term memory to predict failures in advance and proactively trigger the migration. As described, several works have been proposed in the literature that leverage ML for service migration issues. However, to the best of our knowledge, none of them have started focusing on the migration of ML-based tasks that should be treated differently compared to the standard services. Moreover, ML techniques also should be focused on analyzing service runtime and data in order to help detect content that should be transmitted early or as later as possible.

### B. Motivation

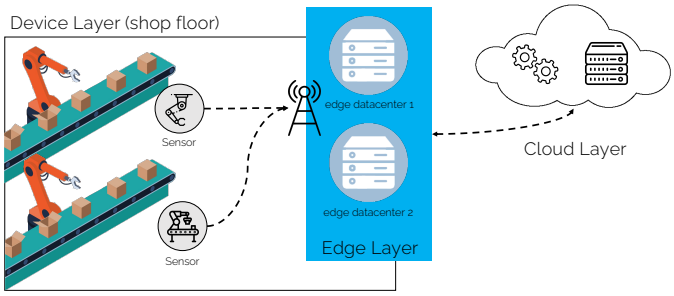



Figure 1. An example of edge-enabled architecture for Industrial Analytics

We illustrate the need for service and data migration in Industrial Analytics scenario with the example shown in Fig. 1. Devices need to run heavy computations and decide to execute ML tasks at the edge layer by leveraging computing resources that it provides. Note that the edge datacenters are all at the same level behind the same 5G base station. The proximity between mobile devices and edge datacenters allows for low latency and better reliability, especially, in the IIoT environment. Specifically, ML tasks and/or online training models need 99,9999% of service availability to properly work [24]. After some time, edge datacenter1 consumes all resources (i.e., CPU and memory) and is no longer able to provide services. In order to guarantee a reasonable level of accuracy, the infrastructure should be able to transparently migrate service/data components from the edge datacenter1 to the new edge datacenter2, which is considered more unloaded.

In this work, we focused on the Industrial Predictive analytics scenario. Specifically Predictive analytics identify expected behaviors or outcomes of machines and systems

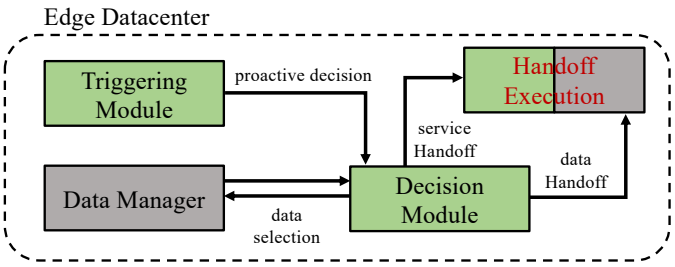



Figure 2. Service and Data Handoff System Architecture

based on predictive modeling using statistical and ML techniques, e.g. capacity demand/usage prediction, material/energy consumption prediction, and component/system wear and fault predictions [1]. Moreover, the analytics results can be applied automatically to the machines and systems, or used to support human decisions to enhance human understanding and generate confidence in a decision. Industrial analytics has unique challenges as an engine driving IIoT transformation because the results can alter the operation and safety of things in the physical world. These effects may be undesirable or harmful, inadvertently affecting the safety of people or damaging property and the environment.

Back to our use case example, production or assembly lines send real-time data streams to the edge datacenters and receive feedback from them. When an alert appears, the operator may recalibrate key parameters of the machines. A problem may occur if an edge datacenter is going to fail, for instance, of the exhaustion of the CPU. The problem that we want to tackle is the re-location of ML tasks to guarantee a high level of accuracy. This means trying to lose as fewer data as possible during the service interruption.

## III. Service and Data Handoff Architecture

This section introduces our solution for service and data migration tailored for Industrial Analytics scenarios and edge computing systems. Our work focuses on the innovation potential of a large set of industrial scenarios that make use of edge-enabled ML-based services. That is typically the case for predictive maintenance and production optimization in manufacturing and for management optimization of large scale facilities.

Fig. 2 presents the architecture of our AI/ML Service and Data Handoff (AMSDH) for the industrial scenario that consists of a set of components that are deployed at the service layer and enable our handoff process. Our proposed architecture consists of four modules: i) a trigger module that takes decisions regarding the moment when migration should be triggered; ii) a decision module that applies migration strategies when the handoff is started; iii) an execution module that executes previous strategies; iv) a data manager module that manages data for ML-based tasks (i.e., predictive maintenance and production optimization).

**Triggering Module**. The goal of this component is to select the best time window to perform the handoff according to several metrics. As outlined in the previous section, we claim the importance of timely detecting an edge datacenter outage. Indeed, this module works in a proactive way in order to trigger the Decision Module for service migration purpose. Several metrics could be taken into consideration for monitoring, for instance, the CPU consumption of edge datacenters. Specifically, when the CPU consumption is above a given threshold, the handoff is started.

**Decision Module**. This module contains a set of strategies that are used to determine service and data mobility. Specifically, it proactively selects neighbor edge datacenters to provide the service functionalities of the ML task. Therefore, data have to be moved in a reactive way in order to guarantee better accuracy. Indeed, this module selects proper data (i.e., fresher data) to move to the new edge datacenter.

**Execution Module**. This module executes the handoff process. Moreover, it offers a set of common APIs that enable the interactions between involved distributed entities and the handoff system. However, it works in two different fashion: in a proactive way for the service migration concern, and in a reactive way for the data migration concern.

**Data Manager**. This module manages the data part of the ML task. Specifically, it manages the trained model and the historical data generated by the ML application for industrial purposes. Moreover, it selects the appropriate data structure to manage the historical data such as a circular buffer or a FIFO queue. Finally, it creates a smart backup of the data, that are considered essential for the working principle of the ML task, that are moved during the service handoff without losing precision.

## IV. Use Case & Implementation Details

This section presents a real case study in order to show the gain in terms of accuracy for ML tasks in the case of handoff. Specifically, this use case aims at leveraging the intelligence at the edge of a chemical detection platform by performing inference tasks for predictive maintenance. To do this we used a public dataset that represents a chemical detection platform [1]. Specifically, the dataset contains information about CO concentration (ppm), humidity (% r.h.), temperature (ºC), flow rate (mL/min), heater voltage (V), and the resistance of 14 different gas sensors. In addition, the dataset is organized of 4 million instances separated into 13 text files, where each file contains 300 000 rows. Finally, the sample frequency is 3.5 Hz.

To emulate the use case, in our testbed, we use the flow rate (mL/min) value due to its non-constant trend (it presents spikes). The simulated sensors read data from the dataset and send a unique packet composed of 50 samples to the edge server for processing. The packet send frequency is $1s$. The edge processing calculates the predicted next value at time $t+1$. The value as being interpreted as the next average value of the flow rate. To do this, the inference task at the edge leverages the historical data value of the flow rate measurements. Then, the predicted value is further processed at the edge for decision.

[1] Available at: https://archive.ics.uci.edu/ml/datasets/Gas+sensor+array+temperature+modulation

Specifically, if the predicted value is in a dangerous range an alarm signal is sent back from the edge to the industry plant.

### A. Implementation Details

As stated before, our testbed consists of User Equipment (UE) that acts as a group of sensors and two edge datacenters (EDCs). The UE is connected to the base station (eNB) and sends data to the EDCs for processing Ml-based tasks such as inference and decision tasks. Furthermore, the UE communicates with EDCs via 4G LTE wireless communication. Finally, the pre-trained model is distributed among the EDCs and is always available.

The sensor uses a part of the dataset, in particular a single csv file that contains 300 000 rows. Specifically we created a python script named *sensors.py* that reads data from the open-source csv file and sends 50 samples every second.

sensors.py

```
file_name = './gas-sensor-file.csv'
df = pd.read_csv(file_name, delimiter=',')
number_of_rows = len(df.index)
n_packets = int(number_of_rows / line) + 1
for i in range(n_packets):
    df1 = df.iloc[i*line:(i+1)*line]
    in_file_name = 'row' + str(i) + '.zip'
    df1.to_pickle(in_file_name, compression='zip')
    rep = client.sendpacket(in_file_name)
    time.sleep(1)
```

On the other side, the EDC retrieves data from sensors and performs inference and decision tasks. To predict the next value, the EDC leverages a pre-trained model formatted in a *h5* file format. Then, it saves historical data in a FIFO queue composed of 10 different files. In the following snippet code *inference.py*, we highlighted the major features of the inference task.

inference.py

```
model = load_model('model/gas-sensor-model.h5')
# make a prediction
yhat = model.predict(test_X)
arr_yhat=[]

for i in range(0,len(yhat)):
    arr_yhat.append(yhat[i])

inv_arr = []
for i in range (0,len(arr_yhat)):
    inv_arr.append(arr_yhat[i]*std + mean)

fifo_file_name = 'o' + str(fifo_file) + '.txt'
fifo_file = fifo_file + 1
o_file = open('knowledge/'+fifo_file_name,'w')
for row in inv_arr:
    np.savetxt(o_file, row)
```

In parallel, we presented the significant parts of the design of our AMSDH solution. Specifically we designed the triggering and the decision modules to work in a proactive way except for the data migration.

triggering module

```
#!/bin/bash
while true
do
    CPU=$(sar 1 5 | sed 's/^.* //')
    CPU=$( printf "%.0f" $CPU )

    if [ "$CPU" -lt 19 ]; then
        echo "CPU usage is high! Handoff"
        sh ./handoff.sh
        break
    fi
done
```

As can be appreciated from the above code, the triggering module triggers the handoff when the remaining CPU percentage is below 19%. The subsequent script *handoff.sh* is part of the Decision Module that proactively migrates the client from EDC1 to EDC2 but only reactively moves data among them. To instantiated the service to the new EDC2, we leveraged Docker Container technology and Kubernetes (Kube). The latter is one of the key enabler for ensuring proactive service migration. Specifically with Kube is possible to instantiate a Pod (a new Docker Container instance) to a new node (i.e., host).

Data Handoff execution

```
#EDC1 part
dirpath = 'knowledge/'
shutil.make_archive('fifo', 'zip', root_dir=dirpath)
client = lib.FileClient(EDC2_IP)
rep = client.upload("handoff.zip")

#EDC2 part
def decompress(filename):
    dir = "knowledge/"
    format = "zip"
    shutil.unpack_archive(filename, dir, format)
    return 1
```

Finally, to guarantee the data migration, we proposed two different python scripts that work in a reactive way. Specifically they start execution only when the handoff is started. The first part of the script (named EDC1 part), creates a zip archive composed of the entire FIFO queue and sends it to the EDC2 via gRPC network protocol. We claim that if we move data as late as possible, we can ensure a high level of accuracy of the ML task. This because we guarantee the ML task to work with fresher data. On the contrary, the second part of the script, decompresses the FIFO queue and restart the data gathering and the ML task at the EDC2.

## V. Experimental Results

This section provides an evaluation of the AMSDH solution discussed in previous sections. We first discuss the experimental setup of openLEON [11] and next we discuss the set of comprehensive results.

### A. Experimental Environment

Fig. 3 shows the openLEON setup. To run the data center network emulated with Containernet [25] and the core network (`srsEPC` application from srsLTE version 19.0.6) we use a laptop equipped with an Intel i7-4600U processor at 2.1 GHz, 8 GB RAM and Linux Ubuntu 16.04 LTS. Then, to run the the BS application (`srsENB` application from srsLTE version 19.0.6), we use a desktop computer equipped with an Intel i7-6700 processor running at 3.4 GHz, 16 GB RAM and Linux Ubuntu 16.04 LTS. The physical BS is an Ettus B210 USRP connected with USB 3.0 to the desktop computer. The UE as well uses a

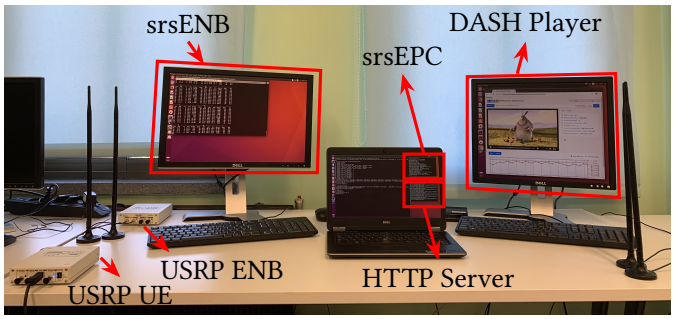



Figure 3. The openLEON platform. In this example setup, the UE consumes a video obtained from a server in the edge. For our experiments, we set the stream direction the other way round.

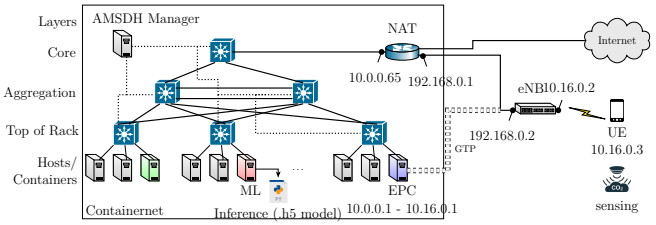



Figure 4. The architecture topology and the AMSDH application

Ettus B210 USRP and it is connected to a laptop with an Intel i7-4600U processor up to 2.1 GHz and 8 GB of RAM that executes the UE application (`srsUE` application from srsLTE version 19.0.6). All results obtained in the emulation are an average of about 30 runs and have exhibited a limited variance, i.e., below 5%.

To assess the accuracy level of our proposed migration system, we devise an Industry 4.0 application aligned with one of the Industry Analytics use cases, such as predictive maintenance. Specifically, we focus a scenario where a group of sensors in a factory are connected to the 5G infrastructure to perform decentralized decision-making. To emulate this, we have created a virtualized network with the Containernet simulation tool which creates the network topology and the edge datacenters. Fig. 4 shows the architecture topology that we implement in openLEON for preliminary tests: no migration is enforced here. The figure provides an intuition of the network setup that is required to operate such a edge scenario. Our AMSDH system is external to the EDCs and manages the two simulated EDCs. For the predictive maintenance application, we used python and its well know ecosystem for ML purposes.

### *B. Experimental Results*

With the experiments, we analyzed the gain in terms of accuracy of a ML-based task in Industry scenario by using an approach based on AMSDH during service and data migration. Specifically, we measured the average prediction error in presence of peaks during a handoff. We first analyzed values in the dataset in order to study its trend and check the number of peaks and they related values. A summary of important metrics of the dataset is presented in Table I. Furthermore, Fig. 5 represents a visual view of the trend of the flow rate metric in the dataset. In that figure, we can appreciate that in

Table I
DATASET ANALYSIS

| **flow rate (mL/min)** | | |
|---|---|---|
| Average Value | Greatest Peak | Greatest Prediction Deviation |
| 239 | 200 | 40 |

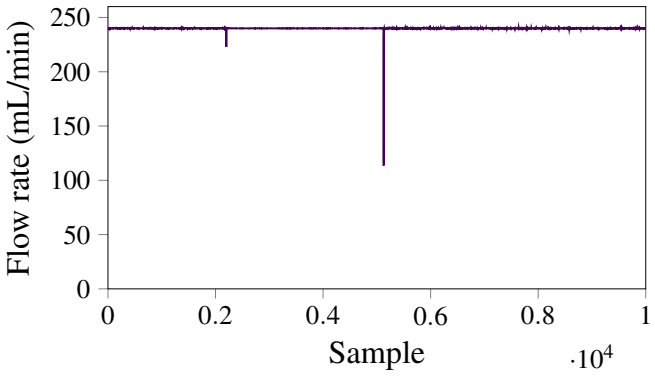



Figure 5. A visual view of the trend of 10000 samples about flow rate metric in the analyzed dataset

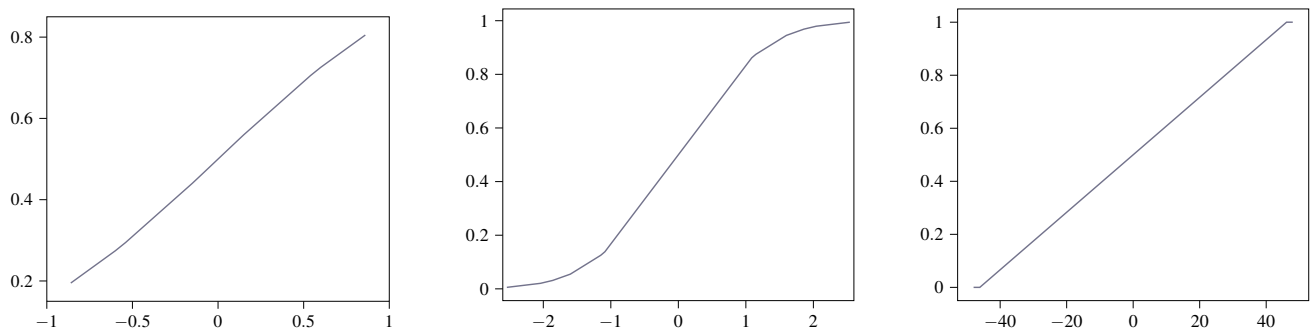
the analyzed dataset there are few peaks, precisely, two peaks in around 10000 samples. However, this definitely represents a problem if we lose one of these detection for our prediction maintenance system.

Then, we measured the gain in accuracy by comparing our AMSDH solution with two different cases such as a standard service migration solution and no migration solution. In order to calculate the accuracy drop, we evaluated the inference value in presence of peaks in the dataset during a handoff. Figure 6 shows the average drop in accuracy of the three different cases in form of a cumulative distribution function. The first case, Fig. 6a, shows that our proposed AMSDH solution in the worst case loses an absolute value average of 0.8 from the real value. The second case, Fig. 6b, loses an average of 1.5 from the real value. Finally, the last case, Fig. 6c, loses an average of 40 from the real value.

## VI. CONCLUSIONS

The paper proposed a Service and Data Handoff system capable of handling migration of ML-based tasks. Specifically, its architecture leverages standard solution for service migration such as Kubernetes. Then, we evaluated the proposed solution in the openLEON emulator under conditions of limited resources at the edge nodes. The results show that the proposed solution can reduce the loss of accuracy during the service migration compared to standard solutions.

Encouraged by these preliminary results, we are now exploring new research directions. On the one hand, are working on mathematical models to manage efficiently the handoff process for ML-based tasks. On the other hand, we are assessing the performance of the proposed migration mechanisms in wide-scale scenarios by carrying out service and data migration experiments over real testbed environments.




(a) CDF of loss of accuracy for AMSDH

(b) CDF of loss of accuracy for standard service migration

(c) CDF of loss of accuracy for no handoff

Figure 6. Comparison of average loss in accuracy in case of handoff in presence of peaks

## ACKNOWLEDGMENT

Dr. Fiandrino's work is supported by the Juan de la Cierva grant from the Spanish Ministry of Science and Innovation (IJC2019-039885-I).

## REFERENCES

[1] I. I. Consortium, "Industrial Analytics: the Engine Driving the IIoT Revolution," https://www.iiconsortium.org/pdf/Industrial_Analytics-the_engine_driving_IIoT_revolution_20170321_FINAL.pdf, 2021, [Online; accessed 30-April-2022].

[2] R. Iqbal, T. Maniak, F. Doctor, and C. Karyotis, "Fault detection and isolation in industrial processes using deep learning approaches," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 5, pp. 3077–3084, 2019.

[3] S. A. Shevchik, G. Masinelli, C. Kenel, C. Leinenbach *et al.*, "Deep learning for in situ and real-time quality monitoring in additive manufacturing using acoustic emission," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 9, pp. 5194–5203, 2019.

[4] M. S. Hossain, M. Al-Hammadi, and G. Muhammad, "Automatic fruit classification using deep learning for industrial applications," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 2, pp. 1027–1034, 2019.

[5] Z. Zhou, X. Chen, E. Li, L. Zeng *et al.*, "Edge intelligence: Paving the last mile of artificial intelligence with Edge computing," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1738–1762, 2019.

[6] L. Lin, X. Liao, H. Jin, and P. Li, "Computation offloading toward edge computing," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1584–1607, 2019.

[7] T. Taleb, K. Samdanis, B. Mada, H. Flinck *et al.*, "On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration," *IEEE Communications Surveys Tutorials*, vol. 19, no. 3, pp. 1657–1681, 2017.

[8] F. Giust, G. Verin, K. Antevski, J. Chou *et al.*, "MEC deployments in 4G and evolution towards 5G," Feb 2018, ETSI White Paper.

[9] D. Scotece, C. Fiandrino, and L. Foschini, "On the efficiency of service and data handoff protocols in edge computing systems," in *2021 IEEE Global Communications Conference (GLOBECOM)*, 2021, pp. 1–6.

[10] C. Sönmez, S. Baghaee, A. Ergişi, and E. Uysal-Biyikoglu, "Age-of-information in practice: Status age measured over tcp/ip connections through wifi, ethernet and lte," in *Proc. of IEEE BlackSeaCom*, 2018, pp. 1–5.

[11] C. Fiandrino, A. B. Pizarro, P. J. Mateo, C. A. Ramiro *et al.*, "openLEON: An end-to-end emulation platform from the edge data center to the mobile user," *Computer Communications*, vol. 148, pp. 17 – 26, 2019.

[12] J. T. Piao and J. Yan, "A network-aware virtual machine placement and migration approach in cloud computing," in *2010 Ninth International Conference on Grid and Cloud Computing*, 2010, pp. 87–92.

[13] B. Gao, Z. Zhou, F. Liu, and F. Xu, "Winning at the starting line: Joint network selection and service placement for mobile edge computing," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 2019, pp. 1459–1467.

[14] S.-H. Wang, P. P.-W. Huang, C. H.-P. Wen, and L.-C. Wang, "Eqvmp: Energy-efficient and qos-aware virtual machine placement for software defined datacenter networks," in *The International Conference on Information Networking 2014 (ICOIN2014)*, 2014, pp. 220–225.

[15] F. A. Salaht, F. Desprez, and A. Lebre, "An overview of service placement problem in fog and edge computing," *ACM Comput. Surv.*, vol. 53, no. 3, jun 2020. [Online]. Available: https://doi.org/10.1145/3391196

[16] D. Xu, T. Li, Y. Li, X. Su *et al.*, "A survey on edge intelligence," *ArXiv*, vol. abs/2003.12172, 2020.

[17] X. Wang, Y. Han, V. C. M. Leung, D. Niyato *et al.*, "Convergence of edge computing and deep learning: A comprehensive survey," *IEEE Communications Surveys Tutorials*, vol. 22, no. 2, pp. 869–904, 2020.

[18] X. Zhao, M. Hosseinzadeh, N. Hudson, H. Khamfroush *et al.*, "Improving the accuracy-latency trade-off of edge-cloud computation offloading for deep learning services," in *2020 IEEE Globecom Workshops (GC Wkshps*, 2020, pp. 1–6.

[19] Z. Lin, S. Bi, and Y.-J. A. Zhang, "Optimizing ai service placement and resource allocation in mobile edge intelligence systems," *IEEE Transactions on Wireless Communications*, vol. 20, pp. 7257–7271, 2021.

[20] M. Masdari and H. Khezri, "Efficient VM migrations using forecasting techniques in cloud computing: a comprehensive review," *Cluster Computing*, vol. 23, no. 4, pp. 2629–2658, 2020. [Online]. Available: https://doi.org/10.1007/s10586-019-03032-x

[21] F. Farahnakian, P. Liljeberg, and J. Plosila, "Lircup: Linear regression based cpu usage prediction algorithm for live migration of virtual machines in data centers," in *2013 39th Euromicro Conference on Software Engineering and Advanced Applications*, 2013, pp. 357–364.

[22] L. Li, J. Dong, D. Zuo, and J. Wu, "Sla-aware and energy-efficient vm consolidation in cloud data centers using robust linear regression prediction model," *IEEE Access*, vol. 7, pp. 9490–9500, 2019.

[23] S. Jeong, N. Van Tu, J.-H. Yoo, and J. W.-K. Hong, "Proactive live migration for virtual network functions using machine learning," in *2021 17th International Conference on Network and Service Management (CNSM)*, 2021, pp. 335–339.

[24] 5G-PPP. White paper: 5g and the factories of the future. [Online; accessed 30-April-2022]. [Online]. Available: https://5g-ppp.eu/wp-content/uploads/2014/02/5G-PPP-White-Paper-on-Factories-of-the-Future-Vertical-Sector.pdf

[25] M. Peuster, J. Kampmeyer, and H. Karl, "Containernet 2.0: A rapid prototyping platform for hybrid service function chains," in *Proc. of IEEE NetSoft*, 2018, pp. 335–337.