Data-centric Edge-AI: A Symbolic Representation Use Case

Shashikant Ilager[1], Vincenzo De Maio[1], Ivan Lujic[2], and Ivona Brandic[1]
[1]*Vienna University of Technology (TU Wien), Austria*
{shashikant.ilager, vincenzo.maio, ivona.brandic}@tuwien.ac.at
[2] *Ericsson Nikola Tesla, Croatia*
ivan.lujic@ericsson.com

***Abstract*—Today's machine learning pipelines are primarily executed in the cloud, from data storage to data processing, model training, and deployment. However, machine learning is moving to edge devices, creating the demand for AI applications at the edge, known as Edge-AI. Traditional data management practices applied in the cloud are proving to be inefficient for Edge-AI, due to resource and energy constraints of edge devices and real-time requirements of applications. This paper identifies the challenges associated with data processing for Edge-AI. We then discuss methods for efficient data processing at the edge, leading to data-centric Edge-AI. As a use case scenario, we discuss the symbolic representation of time series data and explain how it could help save the cost of data storage and processing in developing Edge-AI applications.**

***Index Terms*—Edge-AI, Data-centric Edge, IoT data, Symbolic representation of data, Big data**

I. Introduction

The Internet of Things (IoT) enables different types of physical devices to embed with sensors and actuators and exchange data with smart systems over the Internet. The rapid growth in IoT system deployments produces a huge amount of data, known as Big Data. Traditionally, IoT data is transmitted and stored in a centralized cloud to derive insights and develop smart applications. However, this cloud-centric IoT has become infeasible for time-critical applications for multiple reasons [8], [13], [42]. First, modern applications require a sub-millisecond response to their requests, and the cloud-centric model fails to provide a faster response due to their high network latency. Second, it is expensive to transfer IoT data to remote cloud [35] as it consumes critical bandwidth of the backbone network and creates network congestion [9], [10]. Consequently, edge computing promises to solve these issues by delivering computing, storage, and network resources across cloud boundaries [7], at the network edge. This paradigm shift is powering the development of real-time machine-learning-based applications, known as Edge-AI [13], [15].

Edge-AI enables extracting information from IoT data streams using various techniques such as machine learning, artificial intelligence, and visualization. Edge-AI provides solutions such as anomaly detection, prediction, optimization, and decision-making, which would enable the development of real-time smart systems [36] with several benefits, such as (1) Edge-AI enables data security and privacy since user data do not need to be transferred to a geographically different location or to different ownership; (2) machine learning models such as federated learning [45], and personalized models [3] can be trained locally and collaboratively to preserve data privacy; (3) Edge-AI facilitates (near) real-time analytics and applications, such as autonomous vehicles, and AR/VR systems, which demand faster data processing and response; (4) Edge-AI can assist in reducing energy consumption since it processes data locally, saving energy consumption of data communication [40].

Cloud-native AI assumes an infinite amount of centralized resources to store, process, train and deploy the ML models for run-time inference. Consequently, Edge-AI depends on resources in the geographical vicinity, where the data is generated and consumed, i.e., at the network edge. Due to this hyper-distribution of resources and requirement of cost-effectiveness, edge nodes are designed as much smaller systems to only handle necessary processing tasks in the proximity of IoT systems [33], [42]. The edge nodes can contain all software components of an ordinary cloud data center, but they are resource constrained. The edge nodes can vary from embedded devices (e.g., gateways and everyday smart objects with limited computing capabilities), stand-alone devices (e.g., Raspberry Pis, cloudlet servers) to micro data centers (e.g., co-located data centers and container data centers [1]) according to the application requirements [17]. While edge nodes can provide support for time-critical Edge-AI applications, they cannot scale due to limited computational power and storage capacity [8], [13].

Several efforts have been made to build efficient data processing techniques at the edge, such as reducing network traffic and improving data storage [27], adapting a-posteriori data reduction of data streams [33], raw data compression [44], and energy reduction measures using prediction-based schemes [11]. However, there are many challenges that still exist for efficient edge data processing and the development of machine learning pipelines for Edge-AI. We require *data-centric Edge-AI approaches* to (1) cope with the velocity and volume of data generated, (2) support applications within the resource constraints of edge, (3) efficiently utilize the edge devices [20], [29], and (4) deliver high-quality services to end users with minimal cost. Therefore, in this paper, we investigate edge data-centric processing and identify its associated challenges.

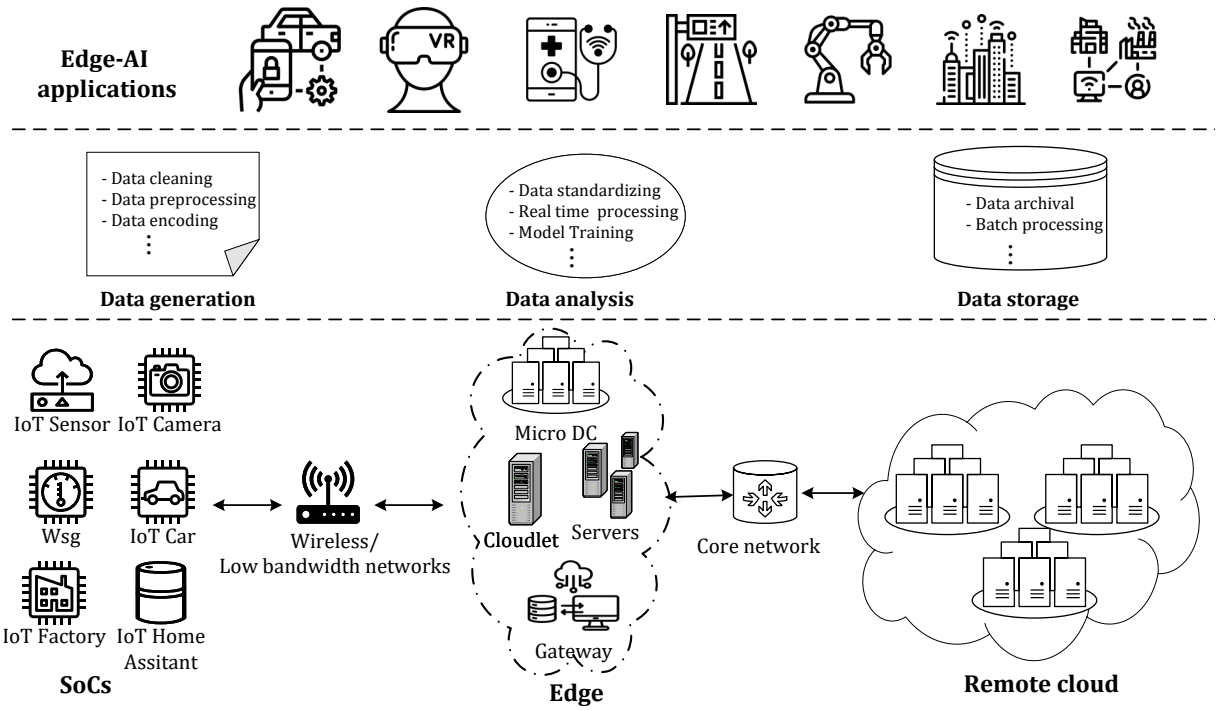

Fig. 1. An overview of Edge-AI.

Afterward, we provide potential methods for efficient data processing to support Edge-AI. Finally, as a use case, we describe the symbolic representation of time series data to reduce data size with minimal or no loss of accuracy, which is suitable for Edge-AI applications.

The rest of the paper is organized as follows. Section II provides background details, and Section III describes the challenges of edge data processing. In Section IV, we discuss the potential methods and future directions to develop better data processing systems for Edge-AI. In Section V, we discuss the symbolic representation use case. Section VI provides related work, and finally, we derive conclusions in Section VII.

II. Background: Data-Centric Edge-AI

Batch-processing has been the basis for designing and building database environments for many years. This involves extracting, transforming, loading, storing, and accessing data for building ML models and developing analytic applications. Batch processing suits the needs of many scientific and enterprise applications, where data is collected for a long time and stored and processed centrally. However, the advent of IoT and edge computing is enabling *stream-processing*, where the data is produced and consumed in real-time by ML pipelines and analytic applications. Many new database environments and stream processing systems have been built recently, such as Apache Kafka, Spark, and Flink [38], to handle the continuous stream of data. However, stream processing frameworks are still designed for centralized cloud environments and do not fully address the challenges of managing the large volume of streaming data at the edge.

The overview of typical Edge-AI applications supported by edge computing is depicted in Fig. 1. IoT sensors are usually implanted over **System-on-Chip** (SoCs) computing devices that provide crucial computational resources. Still, SoCs have limited computational capacities and energy budgets, therefore are not suitable for data management. However, they support lightweight data processing tasks such as data cleaning, preprocessing, and encoding. Also, they rely on higher-layer resources such as edge and cloud for complex data processing tasks. Edge computing provides resources at the network edge, usually at a one-hop distance from the sensors, to enable a real-time computing experience for user applications. They give access to computational resources where data stream could be consumed in real-time and complex analytical tasks could be performed such as data standardization and model training. The remote Cloud provides elastic and inexpensive resources for non-critical tasks such as data archival and batch processing, facilitating non-interactive applications. This multi-tier computing model is creating a computing continuum from the extreme edge (processing the data at the data source itself, in our case, at SoCs in Fig. 1) to the remote cloud, enabling truly distributed real-time computing where the data stream is processed online on heterogeneous distributed resources.

The characteristics of hardware resources, cost, reliability, and energy budget vastly differ among the resources across the computing tiers. Consequently, the computing continuum is creating completely different challenges for data management that are unseen in the past. In the next section, we identify such

unique challenges associated with edge data management and discuss them in detail.

III. Challenges of Data-Centric Edge-AI

A. Data size

IoT sensors generate data in short periodic intervals, leading to massive amounts of data generated within a short period of time. It not only increases storage and processing cost across the computing continuum but also puts tremendous stress on the core backbone networks. While compression could reduce the size of data considerably, computational latency induced by compression and decompression tasks affects real-time applications. Moreover, compression leads to a loss of data accuracy, leading to inaccurate models in Edge-AI. As identified in [5], 80% of energy at SoCs or extreme edge resources are spent only on transferring the data to the nearest processing server. Therefore, it is necessary to develop data size reduction techniques that are suitable for edge environments, requiring methods to deal with the exponential growth of the data and assist Edge-AI applications with minimal resource consumption.

B. Strict real-time requirements

Edge-AI applications support many real-time and near-real-time critical applications. For *real-time applications*, response times must be guaranteed within a specific deadline strictly. On the other hand, in *near real-time applications*, a soft deadline is expected for completing a data processing task. For instance, a real-time traffic monitoring system might use sensor data to detect high traffic volumes and update a map to show congestion or detect blind spot objects in traffic intersections to avoid potential accidents [29]. Similarly, a VR gaming application expects near-real-time processing, and an excessive latency will degrade the quality of experience of users. Developing such systems needs to ingest, preprocess, store, and analyze the data in real-time at high volumes. While existing platforms support real-time requirements of applications [2], [31], they still fail to provide the required reliability and are resource inefficient.

C. Adaptive data processing for dynamic IoT/edge environment

IoT devices and sensors generate data in a distributed heterogeneous environment, which leads to multiple issues, such as (1) a high volume of data generated with irregular velocity, (2) a change in the quality of data generated over time, and (3) demand for highly flexible computing, storage and network resources. In addition, Edge-AI applications are continuously exposed to fluctuating workloads. Therefore, we need to store and retrieve data from edge devices and sensors in a way that meets the performance requirements of different Edge-AI applications. Simple solutions like over-provisioning resources for peak demand would greatly waste resources and increase the cost of application service. On the other hand, under-provisioning would affect the application requirements. Therefore, we require scalable and cost-efficient methods to meet the dynamic situations at the edge.

D. Incomplete and incorrect data

Machine learning pipelines require large amounts of data to train a good-quality model. Data incompleteness is a natural phenomenon in the IoT due to multiple factors, including (1) temporary failure of sensor nodes, (2) network connectivity issues, and (3) measurement errors. Traditionally, missing and incorrect data is either completely removed or data imputation techniques are used to fill in the missing data and correct the data. Existing data imputation techniques depend on statistical characteristics such as mean and median mode for missing items [18] or pattern and correlation identification [22], [47]. Such data imputation techniques perform better when only a small percentage of data is missing and are only feasible for numerical time series data. However, modern IoT sensors are generating not only time series data but also complex data structures with categorical and multi-media data, which is extremely difficult to recover or correct from simple statistical tools. Moreover, we could expect a large volume of missing data as the norm in edge data since IoT sensors are deployed in unreliable environments [39]. Therefore, we require new methods and techniques to handle missing and incorrect data at the edge.

E. Energy

Energy is the main bottleneck across the computing continuum for developing Edge-AI applications sustainably. First, at the extreme edge, SoC sensor nodes spend a significant amount of available energy on the communication subsystem [5]. Hence, reducing the required communication in resource-constrained SoCs is essential for the efficacy of Edge-AI applications. Second, at the edge, the limited power budget is still a big issue for efficient data processing [23], where edge devices are often powered through a limited power supply (e.g., batteries); therefore, energy efficiency is absolutely necessary at the edge. On the other hand, cloud nodes usually have sufficient energy available at their disposal. However, massive energy consumption in the cloud leads to higher service costs and negatively impacts the environment due to its CO_2 footprint. Thus, one single solution would not address all energy issues that we have across the whole computing continuum. We require new data processing methods and platforms that are not only application-aware but also energy-aware in managing data pipelines across computing-continuum.

Summary: Addressing the aforementioned challenges associated with edge data management impacts decision-making through Edge-AI. Efficient data management in Edge-AI affects crucial application and business domains, whether it is related to energy efficiency, reducing traffic accidents or improving air quality, and building Industry 4.0 applications, among others. In the next section, we discuss the potential future directions for edge data processing in relation to the aforementioned challenges.

IV. Future Directions for Data-centric Edge-AI

A. Data size

Edge-AI requires intelligent data size reduction techniques other than raw-data compression. In that regard, Symbolic Representation (SR) of data could be potentially used for many classes of ML applications without needing to reconstruct the compressed data [2]. SR algorithms convert a time series numerical IoT data into a reduced string with a specific length. It makes an approximation of the input data by dividing the time series into a certain number of segments in which data for each segment can be represented by one specific value, i.e., the average of its data points—the original length of the raw data sequence to a reduced string, with a specific alphabet size. In the context of an Edge-AI, this helps to reduce data dimension, size, and network bandwidth usage, save edge storage, and improve analytics features [19].

Nevertheless, many modern Edge-AI applications are built upon diverse and complex types of IoT data, such as categorical and multi-media, with different data formats. In such cases, it becomes important to develop application-specific solutions to deal with data size. For example, decreasing frame resolution in video results in negligible loss of model accuracy [24]; moreover, we do not require high-resolution video frames to train the accurate ML models. Thus, dynamic adaptation of application-specific configurations will lead to a massive reduction in data size.

B. Strict real-time requirements

Strict real-time data processing is an essential component of Edge-AI due to time-sensitive requirements for decision-making in emerging applications. Therefore, it needs completely new low-latency data processing methods. Latency is mainly introduced by two components, i.e., *computational latency* and *network latency*. Tackling computational latency would require techniques such as smart placement of tasks on edge nodes with accelerators or nodes with higher computing power [43]. In a real setting, achieving this constraint becomes challenging due to shared environments and dynamic workloads. Similarly, MLOps such as model quantization and neural network pruning [46] become extremely important at the edge to reduce the computational complexity of machine learning models.

Network latency can be decreased by setting up hyper-distributed edge resources closer to application access points [9]. However, this may not always be feasible due to the cost of infrastructure setup and the practical limitations of geographical locations. In addition, many Edge-AI applications have to deal with user mobility and mobile access points (e.g., autonomous vehicles, drones). Thus, resource provisioning and scheduling approaches should be mobility-aware to meet the network latency requirements. For such scenarios, we envision that spare computing resources in autonomous vehicles, IoT edge devices, and smart systems could be dynamically leased in the near future, resulting in *"data centers on wheels"* paradigm [41], that enables extreme-edge and dynamic mobile computing. However, to realize such a computing paradigm, secure and cost-efficient resource-sharing platforms are required. Unlike the cloud, which has established interoperability standards and matured virtualization technology for secure resource sharing, the edge still requires the development of new standards, lightweight virtualization techniques, and software stacks.

C. Adaptive data processing for dynamic IoT/Edge

Edge-AI applications are subject to constantly changing workloads. Flexible data processing techniques should be developed, such as adaptive sensing [37] and approximate computing [48]. Adaptive sensing needs to consider the requirements of applications and should only produce and process new data only when required. For example, temperature sensor readings do not drastically change most days. Instead of a fixed sensing interval, adapting for a dynamic delayed interval based on the change in actual data could lead to resource efficiency without affecting applications [4], [30]. Similarly, approximate computing in Edge-AI focuses on less precision computation with much lower computational latency. For example, model training and inference with 16-bit computation results in a reduction of multiple magnitudes of computational latency [46] with a little loss in accuracy.

D. Incomplete and incorrect Data

The problem of missing and incomplete data in Edge-AI can be dealt through data-driven methods, going beyond current statistical tools. Particularly, generative AI [32], [34] has been proven to be a feasible method for data imputation [21], [25]. Moreover, federated generative models are able to impute the missing data, learning from the data distribution from other sensors in the network. In addition, generative methods are also able to create new synthetic data sets required for training the machine learning models [49]. At the same time, generative AI has many applications in different areas ranging from creating AI-generated art and a principal role in the development of large language models, such as ChatGPT. However, in general, generative AI suffers from the problem of hallucinating [6], i.e., the generation of plausible outputs which are factually incorrect or unrelated to the given context. Consequently, using generative AI for incomplete or incorrect IoT data might result in completely unseen data distribution compared to real data, and models trained on such data could fit into non-real settings. Therefore, methods to verify whether newly constructed data reflects the actual environments or measurements, avoiding inherent biases of AI models, is of research interest to many.

E. Energy

We do require different energy efficiency measures in each layer of the computing continuum. Since SoCs major energy consumption factor is its communication module, we should focus on developing intelligent data-transfer techniques between SoCs and edge. For instance, transferring data only when a new measurement has a deviation from a recent past

measurement [5]. Alternatively, not transferring the actual data, rather employing predictive models at the edge to estimate the sensor data, and only transfer locally trained models to the edge in a periodic manner or when model drift happens with certain thresholds [14]. In addition to energy-aware approaches in managing continuum resources using traditional methods such as energy-aware scheduling and task placements [20], we require different workload distribution strategies in Edge-AI. Since Edge-AI application components are deployed on distributed infrastructure, workload distribution techniques such as split computing [50] shall be leveraged, where neural networks are dynamically partitioned and deployed across a computing continuum based on the energy budget and latency requirements.

F. Emerging hardware architectures, software paradigms, and future application requirements

While data processing systems should support new data flows and techniques, they should also consider the requirements of the future new Edge-AI applications. There is a paradigm shift in software engineering, where monolithic software applications are decomposed into micro-service-based applications due to their flexibility in developing and maintaining software applications. The application software systems have quickly adapted to this new paradigm, data processing systems are still heavily dependent on traditional monolithic architectures (e.g., Hadoop, Spark), which are resource hungry and fail to match the requirements of Edge-AI. In addition, future Edge-AI applications would spend the majority of their computational cycles on ML model training and inference where different specialized accelerators are necessary [20]. Currently, most edge accelerators (e.g., NVIDIA Jetson Nano, Google Coral device) are designed for inference and many powerful accelerators in the cloud are used for training large ML models. Thus, we require hardware-software co-design to for benefiting from heterogeneous resources across the computing continuum without too many manual configurations and system tuning.

V. Use case: Symbolic Representation of IoT Data at Edge

The development of IoT-assisted smart applications such as smart wearables, which monitor and track vital signs of patients, or smart meters, that balance energy demand and supply in smart grids, generate a large amount of time-series data. Such time-series data are usually transferred to nearby processing devices (e.g., an edge node) to be analyzed. This data transfer can create congestion and consume crucial network bandwidth resources, decreasing the quality of service (QoS) for latency-sensitive smart applications. Moreover, raw data storage on the edge is expensive and infeasible due to limited storage capacity. Consequently, reducing the data size at the source and reconstructing it at the remote edge node (when required) could reduce the network and storage cost. Symbolic Representation (SR) techniques are promising methods for reducing time-series data size while maintaining the semantics of the data [26].

Unlike common raw data compression methods, the symbolically converted data can still be used to directly perform data mining tasks such as pattern matching, substring search, motif discovery, and time series prediction, which are commonly used techniques in IoT applications [16]. Nevertheless, if required, it is still possible to reconstruct the original data at run time with minimal reconstruction error.

Symbolic representation of data: A SR algorithm transforms time series data into a string using finite alphabet size, representing a time series of length n into the string with arbitrary length k ($<< n$). Let us consider a time series $T = [t_0, t_1, ..., t_N] \in \mathbb{R}^{N+1}$ converted into a symbolic representation $S = [s_1, s_2, ..., s_n] \in A_n$, where each s_j is an element of an alphabet $A = a_1, a_2, ..., a_k$ of k symbols [16]. The sequence S should be of considerably lower dimension than the original time series T, that is $n << N$, and it should only use a small number of meaningful symbols, that is $k << n$. This intermediate representation must also allow an approximate reconstruction of the original time series, with (1) a minimal and controllable error and (2) the shape of the reconstruction suitably close to the original time series data.

Adaptive Brownian Bridge-based Aggregation (ABBA) [16] is one of the SR algorithms that converts time series data into symbols. A sample illustration of how time series data is converted into symbolic representation is shown in Fig. 2. Here, the black line on the left side figure represents the original data, and the rightmost side shows symbolically represented data. An SR process involves two parts, namely, (1) the original data is split into segments, either adaptive or user-defined interval numbers and (2) segments are mapped to symbols. In the case of ABBA, segments are found adaptively (left), and relatively similar segments are clustered together (middle), and each cluster is mapped to a symbol from the alphabet. A tolerance hyperparameter tol sets boundaries for the allowed reconstruction error, where a lower value results in a lower reconstruction error, but also a lower compression rate with more symbols. In this example, 230 data points are converted to a word of just 7 symbols (rightmost part of Fig. 2). A similar inverse approach will be applied during the reconstruction of the data. However, many challenges arise when using such algorithms in online and resource-constrained edge environments, as described in the next subsection.

A. Design requirements of symbolic representation of data at edge

The state-of-the-art SR algorithms are designed for centralized batch processing systems and perform an offline conversion. The existing SR algorithms have limited applicability for edge environments because of the following design requirements:

1) **Online**: Compression at the Edge should be continuous, i.e., data should be compressed immediately after being

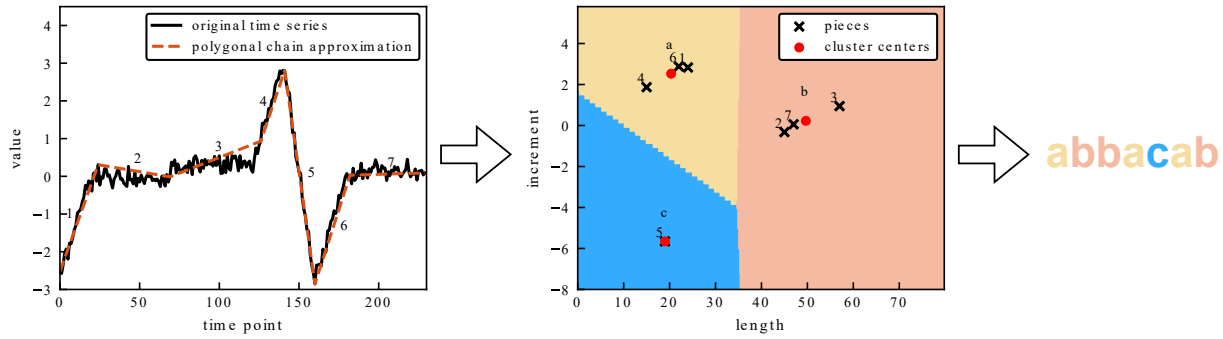

Fig. 2. An illustration of symbolic representation using ABBA [16], [19]. (i) Creating linear pieces using a polygonal chain (left side); (ii) Clustering of pieces (middle); (iii) Symbolizing (right side), i.e., *abbacab*.

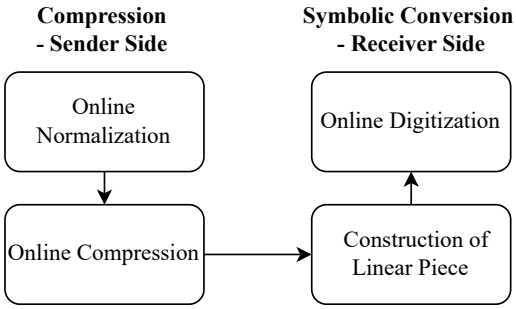

Fig. 3. SymED Components

received since IoT data is produced continuously. Existing SR algorithms are designed to work with batch data, assuming the entire dataset is available prior.

2) **Adaptive:** A SR algorithm should be adaptive (unlike current methods that have fixed parameters such as window and alphabet size), allowing flexibility to configure user-defined parameters, such as rate of compression and error rate in reconstruction based on application and resource constraints.
3) **Distributed:** A SR should be distributed in edge environments since data sources (IoT sensors) and data-consuming nodes are different. The data source does not have enough computational and network capabilities to perform all the steps involved in SR and store the data locally.

In the next subsection, we describe the adaptive SR method suitable for the edge environment.

B. SymEd: Symbolic representation of data at Edge

We discuss our idea of leveraging symbolic representation for data-centric Edge-AI. We discuss SymED [1], Symbolic Edge Data representation method [19], i.e., an online, adaptive, and distributed approach for symbolic representation of streaming data on edge. SymED is based on the Adaptive Brownian Bridge-based Aggregation (ABBA) [16]. Here, we assume low-powered IoT devices do initial data compression (senders), and the more robust edge devices do the symbolic conversion (receivers). The goal is to enable distributed symbolic representation where raw data communication and storage usage are limited in IoT-edge environments. Fig. 3 shows the SymED components. A sender (IoT node) normalizes and compresses all incoming data. A receiver (edge node) collects transmitted data to construct linear pieces (line segments), converts them to symbols in the digitization phase, and optionally reconstructs pieces or symbols again. This method reduces the number of transmitted bytes between IoT and edge devices by efficiently distributing computational tasks of the symbolic representation algorithm between the sender (IoT) and receiver (edge devices). The detailed algorithmic steps of each SymED component (Fig. 3), and its implementation is in [19].

The experiments conducted on 24 datasets from UCR time series classification archive [12] demonstrate that SymEd can significantly reduce the raw data size with minimal computational latency. In summary, SymED achieves on average 9.5% on compression rate and dimension reduction rate, with a mean online reconstruction error of 13.25, while taking a mean time of 42ms to compute a symbol. With a slight overhead in compression performance and computational efficiency compared to offline ABBA, online SymED enables real-time symbolic conversion while improving reconstruction accuracy and adapting to the data stream distribution.

VI. Related Work

The concept of edge computing and its role in IoT has been discussed by [7], which introduces the main concepts of edge/fog architectures and identifies the main challenges. A first tentative in the standardization of edge computing has been described by works such as [17]. Research efforts related to edge data management focus mostly on storage management [27] and in the recovery of time series data, by means of forecasting methods [28] and imputation [22]. However, most of these works are focused on the analysis of time series data and do not consider challenges related to Edge-AI applications.

Edge-AI has been discussed in works such as [50]. Different use cases for Edge-AI have been proposed in recent years,

[1]The presented use case idea is published in [19]; more details about it can be found in the referred paper online.

such as vehicular traffic safety [29], and environmental monitoring [3]. Challenges of guaranteeing efficient data streaming at the edge are discussed in works such as [38], [40].

In this work, we focus on identifying challenges and research opportunities of data-intensive Edge-AI applications, whose goal is to reduce data size while guaranteeing high accuracy of Edge-AI models, and identify a use case for data compression techniques, starting from our seminal work in symbolic data representation at the edge [19].

VII. Conclusions

AI-based applications are becoming pervasive and moving from centralized cloud deployments to the network edge, leading to Edge-AI. Edge-AI applications have to deal with a continuous stream of IoT data and support the latency requirements of IoT applications. To efficiently utilize edge resources and process streaming data, we require completely new approaches, i.e., a data-centric view of how data, applications, and resources are managed, considering the capabilities and limitations of edge environments. In this paper, we have discussed the challenges associated with the data-centric Edge-AI and identified the potential future directions describing different methods and techniques that we can apply to solve the identified challenges. As a use case, we presented the adaptive symbolic representation of IoT data to reduce the streaming data size and support in developing ML applications.

Acknowledgements

This work has been partially funded through the Rucon project (Runtime Control in Multi Clouds), Austrian Science Fund (FWF): Y904-N31 START-Programm 2015, by the CHIST-ERA grant CHIST-ERA-19-CES-005, Austrian Science Fund (FWF), Standalone Project Transprecise Edge Computing (Triton), Austrian Science Fund (FWF): P 36870-N, and by Flagship Project HPQC (High Performance Integrated Quantum Computing) # 897481 Austrian Research Promotion Agency (FFG).

References

[1] What is a containerized data center: Pros and cons. https://community.fs.com/blog/what-is-a-containerized-data-center-pros-and-cons.html (2022), [Online; accessed 05-May-2023]

[2] Abbas, N., Asim, M., Tariq, N., Baker, T., Abbas, S.: A mechanism for securing iot-enabled applications at the fog layer. Journal of Sensor and Actuator Networks **8**(1), 16 (2019)

[3] Ahmad, S., Aral, A.: Fedcd: Personalized federated learning via collaborative distillation. In: 2022 IEEE/ACM 15th International Conference on Utility and Cloud Computing (UCC). pp. 189–194. IEEE (2022)

[4] Almeida, F., Assunção, M.D., Barbosa, J., Blanco, V., Brandic, I., Da Costa, G., Dolz, M.F., Elster, A.C., Jarus, M., Karatza, H.D., Lefèvre, L., Mavridis, I., Oleksiak, A., Orgerie, A.C., Pierson, J.M.: Energy monitoring as an essential building block towards sustainable ultrascale systems. Sustainable Computing: Informatics and Systems **17**, 27–42 (2018). https://doi.org/https://doi.org/10.1016/j.suscom.2017.10.013, https://www.sciencedirect.com/science/article/pii/S2210537916301536

[5] Anastasi, G., Conti, M., Francesco, M.D., Passarella, A.: Energy conservation in wireless sensor networks: A survey. Ad Hoc Networks **7**(3), 537–568 (May 2009). https://doi.org/10.1016/j.adhoc.2008.06.003

[6] Bang, Y., Cahyawijaya, S., Lee, N., Dai, W., Su, D., Wilie, B., Lovenia, H., Ji, Z., Yu, T., Chung, W., Do, Q.V., Xu, Y., Fung, P.: A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity (2023)

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

[8] Buyya, R., Srirama, S.N., Casale, G., Calheiros, R., Simmhan, Y., Varghese, B., Gelenbe, E., Javadi, B., Vaquero, L.M., Netto, M.A., et al.: A manifesto for future generation cloud computing: Research directions for the next decade. arXiv preprint arXiv:1711.09123 (2017)

[9] Charyyev, B., Arslan, E., Gunes, M.H.: Latency comparison of cloud datacenters and edge servers. In: GLOBECOM 2020 - 2020 IEEE Global Communications Conference. pp. 1–6 (2020). https://doi.org/10.1109/GLOBECOM42002.2020.9322406

[10] Chen, Y.K.: Challenges and opportunities of internet of things. In: Design Automation Conference (ASP-DAC), 2012 17th Asia and South Pacific. pp. 383–388. Citeseer (2012)

[11] Cui, E., Yang, D., Zhang, H., Gidlund, M.: Improving power stability of energy harvesting devices with edge computing-assisted time fair energy allocation. IEEE Transactions on Green Communications and Networking **5**(1), 540–551 (2020)

[12] Dau, H.A., Bagnall, A., Kamgar, K., Yeh, C.C.M., Zhu, Y., Gharghabi, S., Ratanamahatana, C.A., Keogh, E.: The ucr time series archive. IEEE/CAA Journal of Automatica Sinica **6**(6), 1293–1305 (2019)

[13] Deng, S., Zhao, H., Fang, W., Yin, J., Dustdar, S., Zomaya, A.Y.: Edge intelligence: The confluence of edge computing and artificial intelligence. IEEE Internet of Things Journal **7**(8), 7457–7469 (2020). https://doi.org/10.1109/JIOT.2020.2984887

[14] Dias, G.M., Bellalta, B., Oechsner, S.: The impact of dual prediction schemes on the reduction of the number of transmissions in sensor networks. Computer Communications **112**, 58–72 (Nov 2017). https://doi.org/10.1016/j.comcom.2017.08.002

[15] Ding, A.Y., Peltonen, E., Meuser, T., Aral, A., Becker, C., Dustdar, S., Hiessl, T., Kranzlmüller, D., Liyanage, M., Maghsudi, S., et al.: Roadmap for edge ai: A dagstuhl perspective (2022)

[16] Elsworth, S., Güttel, S.: Abba: adaptive brownian bridge-based symbolic aggregation of time series. Data Mining and Knowledge Discovery **34**(4), 1175–1200 (2020)

[17] Group, O.C.A.W., et al.: Openfog reference architecture for fog computing. OPFRA001 **20817**, 162 (2017)

[18] Hadeed, S.J., O'Rourke, M.K., Burgess, J.L., Harris, R.B., Canales, R.A.: Imputation methods for addressing missing data in short-term monitoring of air pollutants. Science of The Total Environment **730**, 139140 (2020)

[19] Hofstätter, D., Ilager, S., Lujic, I., Brandic, I.: Symed: Adaptive and online symbolic representation of data on the edge. In: Proceedings of the 29th International European Conference on Parallel and Distributed Computing (Europar). Springer (2023)

[20] Ilager, S., Muralidhar, R., Buyya, R.: Artificial intelligence (ai)-centric management of resources in modern distributed computing systems. CoRR **abs/2006.05075** (2020), https://arxiv.org/abs/2006.05075

[21] Jiang, H., Wan, C., Yang, K., Ding, Y., Xue, S.: Continuous missing data imputation with incomplete dataset by generative adversarial networks–based unsupervised learning for long-term bridge health monitoring. Structural Health Monitoring **21**(3), 1093–1109 (2022)

[22] Junger, W., de Leon, A.P.: Imputation of missing data in time series for air pollutants. Atmospheric Environment **102**, 96–104 (2015)

[23] Khan, M.A., Algarni, F., Quasim, M.T.: Decentralised internet of things. Decentralised Internet of Things: A Blockchain Perspective pp. 3–20 (2020)

[24] Khochare, A., Krishnan, A., Simmhan, Y.: A scalable platform for distributed object tracking across a many-camera network. IEEE Transactions on Parallel and Distributed Systems **32**(6), 1479–1493 (2021). https://doi.org/10.1109/TPDS.2021.3049450

[25] Kuppannagari, S.R., Fu, Y., Chueng, C.M., Prasanna, V.K.: Spatio-temporal missing data imputation for smart power grids. In: Proceedings of the Twelfth ACM International Conference on Future Energy Systems. pp. 458–465 (2021)

[26] Lin, J., Keogh, E., Wei, L., Lonardi, S.: Experiencing sax: a novel symbolic representation of time series. Data Mining and knowledge discovery **15**(2), 107–144 (2007)

[27] Lujic, I., De Maio, V., Brandic, I.: Efficient edge storage management based on near real-time forecasts. In: 2017 IEEE 1st International Conference on Fog and Edge Computing (ICFEC). pp. 21–30. IEEE (2017)

[28] Lujic, I., De Maio, V., Brandic, I.: Adaptive recovery of incomplete datasets for edge analytics. In: Fog and Edge Computing (ICFEC), 2018 IEEE 2nd International Conference on. pp. 1–10. IEEE (2018)

[29] Lujic, I., Maio, V.D., Pollhammer, K., Bodrozic, I., Lasic, J., Brandic, I.: Increasing traffic safety with real-time edge analytics and 5g. In: Proceedings of the 4th International Workshop on Edge Systems, Analytics and Networking. p. 19–24. EdgeSys '21, Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3434770.3459732, https://doi.org/10.1145/3434770.3459732

[30] Mastelic, T., Brandic, I.: Data velocity scaling via dynamic monitoring frequency on ultrascale infrastructures. In: 2015 IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom). pp. 422–425. IEEE (2015)

[31] Montella, R., Ruggieri, M., Kosta, S.: A fast, secure, reliable, and resilient data transfer framework for pervasive iot applications. In: Conference on Computer Communications Workshops. pp. 710–715. IEEE (2018)

[32] Noy, S., Zhang, W.: Experimental evidence on the productivity effects of generative artificial intelligence. Available at SSRN 4375283 (2023)

[33] Papageorgiou, A., Cheng, B., Kovacs, E.: Real-time data reduction at the network edge of internet-of-things systems. In: 2015 11th International Conference on Network and Service Management (CNSM). pp. 284–291. IEEE (2015)

[34] Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., Chen, M.: Hierarchical text-conditional image generation with clip latents (2022)

[35] Ranjan, R.: Streaming big data processing in datacenter clouds. IEEE Cloud Computing **1**(1), 78–83 (2014)

[36] Schulz, P., Matthe, M., Klessig, H., Simsek, M., Fettweis, G., Ansari, J., Ashraf, S.A., Almeroth, B., Voigt, J., Riedel, I., et al.: Latency critical iot applications in 5g: Perspective on the design of radio interface and network architecture. IEEE Communications Magazine **55**(2), 70–78 (2017)

[37] Sekine, M., Ikada, S.: Adaptive cooperative distributed compressed sensing for edge devices: a multiagent deep reinforcement learning approach. In: 2021 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops). pp. 585–591. IEEE (2021)

[38] Shahverdi, E., Awad, A., Sakr, S.: Big stream processing systems: an experimental evaluation. In: 2019 IEEE 35th International Conference on Data Engineering Workshops (ICDEW). pp. 53–60. IEEE (2019)

[39] Shi, W., Cao, J., Zhang, Q., Li, Y., Xu, L.: Edge computing: Vision and challenges. IEEE internet of things journal **3**(5), 637–646 (2016)

[40] Shi, Y., Yang, K., Jiang, T., Zhang, J., Letaief, K.B.: Communication-efficient edge ai: Algorithms and systems. IEEE Communications Surveys & Tutorials **22**(4), 2167–2191 (2020)

[41] Sudhakar, S., Sze, V., Karaman, S.: Data centers on wheels: Emissions from computing onboard autonomous vehicles. IEEE Micro **43**(1), 29–39 (2023). https://doi.org/10.1109/MM.2022.3219803

[42] Sun, X., Ansari, N.: Edgeiot: Mobile edge computing for the internet of things. IEEE Communications Magazine **54**(12), 22–29 (2016)

[43] Tuli, S., Ilager, S., Ramamohanarao, K., Buyya, R.: Dynamic scheduling for stochastic edge-cloud computing environments using a3c learning and residual recurrent neural networks. IEEE Transactions on Mobile Computing **21**(3), 940–954 (2022). https://doi.org/10.1109/TMC.2020.3017079

[44] Wang, J.B., Zhang, J., Ding, C., Zhang, H., Lin, M., Wang, J.: Joint optimization of transmission bandwidth allocation and data compression for mobile-edge computing systems. IEEE Communications Letters **24**(10), 2245–2249 (2020)

[45] Wang, S., Tuor, T., Salonidis, T., Leung, K.K., Makaya, C., He, T., Chan, K.: Adaptive federated learning in resource constrained edge computing systems. IEEE journal on selected areas in communications **37**(6), 1205–1221 (2019)

[46] Wang, T., Wang, K., Cai, H., Lin, J., Liu, Z., Wang, H., Lin, Y., Han, S.: Apq: Joint search for network architecture, pruning and quantization policy. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2078–2087 (2020)

[47] Wellenzohn, K., Böhlen, M.H., Dignös, A., Gamper, J., Mitterer, H.: Continuous imputation of missing values in streams of pattern-determining time series. In: EDBT. pp. 330–341 (2017)

[48] Wen, Z., Quoc, D.L., Bhatotia, P., Chen, R., Lee, M.: Approxiot: Approximate analytics for edge computing. In: 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS). pp. 411–421 (2018). https://doi.org/10.1109/ICDCS.2018.00048

[49] Zhang, C., Kuppannagari, S.R., Kannan, R., Prasanna, V.K.: Generative adversarial network for synthetic time series data generation in smart grids. In: 2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm). pp. 1–6 (2018). https://doi.org/10.1109/SmartGridComm.2018.8587464

[50] Zhao, Z., Barijough, K.M., Gerstlauer, A.: Deepthings: Distributed adaptive deep learning inference on resource-constrained iot edge clusters. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **37**(11), 2348–2359 (2018). https://doi.org/10.1109/TCAD.2018.2858384