International Journal of Science and Research (IJSR)
ISSN: 2319-7064
SJIF (2022): 7.942

An Optimized IoT-Enabled Big Data Analytics Architecture for Edge-Cloud Computing Using Deep Learning

Bharathi K.

Assistant Professor, Department of computer science and engineering, Vivekananda College of Engineering and Technology
Email: bharathi2k2[at]gmail.com

Abstract: ***Large numbers of connected devices including sensors, mobile, wearable, and other Internet of Things (IoT) devices, is creating large scale of data that are moving across the network. To carry out machine learning (ML), IoT data are typically transferred to the cloud or another centralized system for storage and processing; however, this causes latencies and increases network traffic. Edge computing has the potential to remedy those issues by moving computation closer to the network edge and data sources. On the other hand, edge computing is limited in terms of computational power, and thus, is not well-suited for ML tasks. Consequently, proposed work aims to combine edge and cloud computing for IoT data analytics by taking advantage of edge nodes to reduce data transfer. In order to process data close to the source, sensors are grouped according to locations, and feature learning is performed on the close by edge node. For comparison reasons, similarity-based processing is also considered. Feature learning is carried out with deep learning- the encoder part of the trained Auto-encoder is placed on the edge and the decoder part is placed on the cloud. The evaluation was performed on the task of human activity recognition from sensor data. When sliding windows are used in the preparation step, data can be reduced on the edge up to 80% without significant loss in accuracy.***

Keywords: Auto encoders (AEs), data reduction, deep learning (DL), edge computing (EC), human activity recognition (HAR), Internet of Things (IoT)

1.Introduction

CISCO estimates that the number of connected devices will exceed 28 billion by the year 2022, up from 18 billion in 2017 [1]. More than half of those devices, over 14.6 billion, will be machine-to-machine connections. The number of connected devices, together with a flood of data they generate, will increase network traffic. According to Cisco, by the year2022, global annual internet traffic will reach 4.8 zettabytes. Although this will come with positive impacts including new applications/services and increased use of the existing ones, it will also increase demand on network bandwidth and put pressure on the already strained communication infrastructure.

The Internet of Things (IoT) provides a platform for devices to connect to the Internet and other devices and enables devices to collect data about their environment. IoT supports smart systems such as smart cities, smart health care, smart transportation, and smart energy; however, the realization of these smart systems relies on the ability to analyze these data [2]. On the other hand, most IoT edge devices, such as sensors, do not have computation abilities to perform complex data analytics computations and, therefore, have been primarily responsible for monitoring the environment and transmitting data to a more powerful system, often the cloud or a centralized system, for storage and processing [3].

Consequently, a typical IoT data analytics involve sending data to the cloud, analyzing it on the cloud, and subsequently delivering results to another device. For example, process monitoring data from a smart factory may be transferred to a data center thousands of miles away where they are stored and processed; then, the results are sent back to the same factory for process optimization. This workflow increases not only network traffic, but also data transfer latencies. However, data analytics computation cannot be performed solely on the connected devices as they have limited computation resources.

Combining edge with cloud computing has the potential to reduce IoT network traffic and associated latencies while still supporting complex data analytics tasks. Edge computing (EC) pushes the computation away from the cloud or a centralized system and to the edges of the network and sources of data, and thus, reduces network traffic and latencies. Although EC has been recognized as a powerful approach for tasks such as mobile task offloading [4] and content delivery [5], its use for data analytic has remained limited [6], [7].

In recent years, deep learning (DL) has demonstrated success in a variety of domains including image classification [8] and human activity recognition (HAR) [9]. In the IoT context, DL ability to carry out representation learning and to transform data into hierarchical abstract representations is beneficial for IoT data analytics because it can enable learning the good features. A type of DL, a deep autoencoder (AE) is a NNs trained to learn data encoding in an unsupervised manner. Together with encodings, the reconstruction is learned enabling AE to restore its inputs from the reduced encodings, possibly with some loss of information.

Proposed work investigates combining edge and cloud computing with IoT data analytics. The main contributions are the reduction of network traffic and latencies for machine learning (ML) tasks by employing

the edge nodes and the evaluation of the degree of data reduction that can be achieved on the edge without a significant impact on the ML task accuracy. Edge nodes act as intermediaries between IoT devices and the cloud reducing the quantity of data sent to the cloud. The encoder part of the trained AE is employed on the edge to create data encodings that are sent to the cloud. ML task on the cloud is carried out in two ways-directly with encoded data, or the original data are first restored with the decoder part of the AE and then used for the ML task.

As IoT data can originate from different sensors and locations, this study explores feature learning from all data fused together, from data grouped by their source locations, and from data grouped according to sensor similarities. The evaluation scenario involves HAR from sensors including accelerometers and gyroscopes mounted on different parts of the human body.

The remainder of the paper is organized as follows. Section II provides the background; Section III discusses related work, Sections IV present edge-cloud ML model. Finally, Section V concludes this article.

2.Background

This section introduces EC and discusses DL-based dimensionality reduction.

A. Edge Computing

Centralized infrastructure systems, such as the cloud, store data, execute business logic, and perform data analytics tasks far away from the end users and data sources. They offer great advantages of immense computation capability, high scalability and reliability, pay-as-you-go billing model, and low initial cost. However, with zettabyte-sized traffic and the explosion of connected devices, transferring all data to the cloud for processing is not practical or even feasible. EC has emerged as a way of dealing with these challenges by pushing computation to the edges of the network and sources of data [6].

Fog computing shares many characteristics with EC. Occasionally, the terms “fog” and “edge” are used interchangeably [10]; but EC focuses more on nodes closer to IoT devices, whereas fog can include any resource located anywhere between the end device and the cloud. In this work, the term “edge” to refer to the end devices themselves, such as sensor nodes and smart phones, as well as computation nodes located close to the network edge such as edge servers.

The key idea behind the EC is to reduce the network traffic by bringing computation closer to the data sources. It has been investigated extensively in mobile computing: mobile EC offloads computation and data storage to the edge (e.g., base stations) to reduce network latencies, improve user experience, reduce battery consumption, and introduce location-awareness [11]. EC is especially suitable for applications with ultralow latency requirements and for content delivery and caching [12].

Even though EC provides advantages of reduced traffic, computing resources available on the edge are not comparable to those present in the cloud. Thus, computationally intensive tasks, such as ML, are not well-suited for edge devices. Nevertheless, the edge can supplement the cloud computing and perform part of the computation, consequently reducing network traffic and latencies.

B. Deep Learning

DL is a class of ML approaches in which the models are composed of multiple computational layers responsible for learning data representations with different levels of abstraction [13]. Due to its representation capabilities, ability to learn complex models and diversity of architectures [2], DL has been successful in many domains including various vision tasks, natural language processing, and speech recognition.

AEs are a subcategory of DL approaches used for learning data representations (encodings) in an unsupervised way. Essentially, an AE is a neural network (NN) that learns to reconstruct its inputs; bottleneck NN layers prevent it from merely copying the input to output and force it to learn data representations. An AE consists of encoder and decoder, each one possibly composed of several stacked layers. The encoder part of the network is responsible for reducing dimensionality (encoding); therefore, the number of neurons typically reduces starting from the input layer to the last encoder layer. In contract, the decoder part is responsible for reconstructing the input signal from encoded values, and thus, typically consists of layers with a gradually increasing number of neurons. AE can be used for noise removal and anomaly detection, but they often serve as a preprocessing step for another ML task [2]. Once an AE is trained, the encoder network can be used for dimensionality reduction by taking encoder outputs (encodings) as inputs to another ML model.

In this article, the encoder part of the trained AE is deployed on the edge to reduce dimensionality before the data are sent to the cloud. Moreover, the decoder is employed on the cloud to reconstruct the original signal.

3.Related Work

EC has been gaining popularity, especially with applications that require fast response time and those with limited bandwidth because it locates computation close to data sources. Applications of EC including smart street lamps [14], face identification [15], smart manufacturing [16], and vehicular networks [17] have demonstrated great success and prompted further investigations.

Wang et al. [18] presented a survey on mobile edge networks focusing on computing-related issues, edge offloading, and communication techniques for edge-based computing. The use cases highlighted in their study include IoT, connected vehicles, content delivery, and big data analysis. At the same time, Wang et al. [18] identified real-time analytics as one of the open challenges. Similarly, Abbas et al. [12] surveyed mobile EC and also

identified big data analytics as a future research direction. While Wang et al. [18] and Abbas et al. [12] examined mobile EC, El-Sayed et al. [6] focused on IoT applications of EC. They compared the characteristics of cloud, multi cloud, fog, and EC, and identified low bandwidth utilization and latencies as the main EC advantages. Mao et al. [19] see EC as a key enabling technology for realizing the IoT vision and, similar to Wang et al. [18] and Abbas et al. [12], recognize data analytics as one of the future research directions in EC. The discussed surveys [6], [12], [18], [19] note the potential ofEC in data analytics and point out the importance of EC in IoT for handling the rapid increase of the number of connected devices. Proposed work contributes to employing EC for data analytics by combining edge and cloud computing for the delivery of ML applications.

Smart cities are one of the commonly discussed use cases and applications of EC. Mohammad et al. [20] examined possibilities of service-oriented middleware for cloud and fog enabled smart city services. They did not discuss specific smart city services but focused on the middleware. Their experiments demonstrated the benefits of EC in terms of response time. Tang et al. [21] presented a hierarchical fog computing architecture for the support of connected devices in smart cities. In addition to the hierarchy of fog nodes, the proposed model includes the cloud as the top layer. The evaluation was performed on an event detection task in a smart pipeline monitoring system: the preliminary results demonstrated the feasibility of the proposed architecture. Similar to Mohammad et al. [20] and Tang et al. [21], proposed work also employs both edge/fog and cloud, but differs from theirs in that it also includes an extensive evaluation of the presented edge-cloud architecture.

Another example of the cloud-fog approach is the work of Wang et al. [22] on combining fog and cloud computing for real-time traffic management. In their system, vehicles act as edge nodes, and roadside units take the role of cloudlets and communicate between vehicles and the cloud. Their study [22] focuses on message passing and processing while this work deals with ML for IoT. He et al. [23] proposed a multitier fog computing model for large-scale IoT data analytics in Smart Cities. They evaluated the proposed model on the classification tasks- the results demonstrated that fogs can improve the performance of smart city services. While the work of He et al. [23] deals with fog architecture, proposed study takes advantage of both edge and cloud for the ML task.

There are also applications of fog computing in health care. Rahmani et al. [24] presented a fog-assisted architecture for smart e-Health which embeds intelligence between sensors and the cloud. In their study, fog nodes are quite powerful and thus able to handle data filtering, compression, fusion, and analysis with only minimal data sent to the cloud. This study, on the other hand, still performs a large part of the computation on the cloud. Ritrovato et al. [25] also dealt with health care and proposed an EC anomaly detection system for streaming data. While they are concerned with stream processing algorithms, this study deals with ML algorithms.

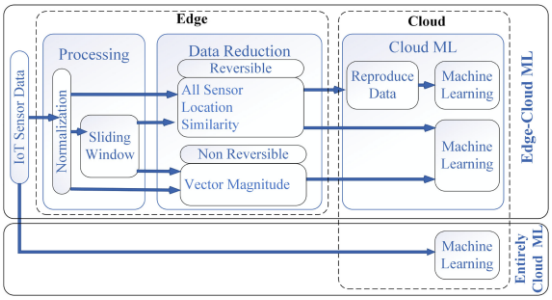
The main difference between proposed work and the reviewed studies is that this work combines the edge and cloud for ML tasks. Several studies also employed edge-cloud architectures [20], [22], [24], but for non-ML tasks. For ML tasks, the processing on the edge must accommodate final ML computation on the cloud while still reducing network traffic. The most similar work to this is the work of Tang et al. [21] as they also consider ML and feature extraction on the edge; however, they only extract the signal's mean and variance, which limits application scenarios, while proposed work use a generic approach based on AEs. Moreover, here also evaluate the degree of feature reduction that can be carried out without significantly impacting ML accuracy.

As this study evaluates the presented approach on sensor based (HAR, it is important to mention a few works from this category. Given the fact that DL has been quite successful and extensively used for HAR [26], the work of Wang et al. [26] surveyed DL approaches for activity recognition; they highlighted the importance of model selection and the significance of preprocessing including the sliding window technique. Zdravevski et al. [27] specifically focused on feature engineering for HAR: they first extracted a large number of features (3232 for MHEALTH dataset), and then reduced them by combining different feature reduction techniques. Ferrari et al. [28] investigated personalization of HAR models and also applied sliding windows and feature generation. Li et al. [29] are concerned with recognizing transitions between activities; they first extracted 118 features and created fixed size segments. Next, each segment was analyzed to determine if there was a change of activity within the segment.

These HAR studies do not employ EC, nor are they concerned with network traffic. As the sliding window technique improves accuracy, it is commonly used in HAR studies [26]– [28]; While others are interested in the sliding window technique effect on accuracy, here investigate its impact on data reduction on the edge. AE and principal component analysis (PCA) have been used to improve HAR accuracy [9], but our study uses those techniques to reduce network traffic.

4.Edge-Cloud Ml System Model

The overall architecture of the edge-cloud ML model is depicted in figure and details of each of the three main components, preprocessing, data reduction, and Cloud ML, are described in the following sections.



A. Preprocessing

The edge-cloud ML process starts with data from IoT sensors being passed to the edge for preprocessing in contrast to traditional ML where the data are sent directly to the cloud. The preprocessing always includes normalization while the sliding window technique is optional, and its impact is evaluated in the experiments.

1) Normalization: To avoid dominance of features with large values and to improve training convergence, the data are normalized using standardization (z-score). In place of standardization, min–max scaling could be used, but standardization was selected because of its ability to handle outliers. Each feature is rescaled to have zero mean and unit variance as given by

$$\hat{x} = \frac{x - \mu}{\sigma} \quad (1)$$

where x is the original feature value, μ and σ are that feature mean and standard deviation, and ˆx is the normalized value.

2) Sliding Window: At this point, one data sample consists of several readings, potentially from different sensors and different locations, for the same time step t. For the time series data, the windows sliding technique is applied to help the model capture time-dependencies or to prepare data in the format needed by the ML algorithms [30]. In HAR, the sliding window has achieved great success [30]. This study investigates if features can be reduced after the sliding window technique is applied and examines the effect on the degree of data reduction. With the sliding window of length l, the first sample consists of all readings for the first l time steps; with the f number of features, this result in the l × f matrix for each sample. Next, the window slides for k steps and the second sample consists of the readings from the time step k to k + l. The window keeps sliding to create the remaining samples. In this work, the sliding step k = 1 is used as this results in a higher number of samples for training and allows the input to capture shifts in temporal patterns. Once the system is trained, different sliding steps can be applied depending on the specifics of the use case and data transfer constraints.

B. Data Reduction

Data reduction happens on the edge in order to reduce the quantity of data sent to the cloud. Feature reduction challenges include selecting the technique and the number of features. In the centralized systems, these decisions are primarily driven by the ML accuracy while in the edge-cloud environment; network traffic also needs to be considered. Even though AEs are well suited for feature learning on the edge; it remains a challenge to determine the maximum feature reduction and the corresponding traffic reduction while maintaining the accuracy of the ML tasks. Data reduction can be carried out directly with normalized data or with samples created by the sliding window technique. Regardless of whether the sliding window technique is used or not, the data reduction approach is the same. Two categories of approaches are considered-reversible and nonreversible.

1) Reversible: Reversible approaches are the approaches that reduce data with an ability to reproduce the original data from the reduced representations. With these approaches data reduction executes on the edge, reduced data are sent over the network, and on the cloud, ML can be performed directly on the reduced data, or the original data can be reproduced first. Here, focus on AEs as, once the AE is trained, the encoder can reduce data on the edge and decoder can restore the original data on the cloud. AE performance is compared to data reduction with PCA [31]. When PCA is applied to reduce dimensions, original data can be reconstructed using the eigenvectors. For both, AE and PCA, the accuracy of reconstructed data depends on the degree of dimensionality reduction.

As illustrated in Fig.1, for both reversible techniques, AE and PCA, three different scenarios are considered-all sensors, location-based, and similarity-based scenarios. In the all sensors approach, all the data are considered together, and data reduction is performed on the merged data from all sensors. The location-based scenario considers data reduction based on a group of colocated sensors. For example, in location 1, there are four different sensors, and their data are sent to the edge node E1 because of its close proximity to those sensors. Similarly, the location 2 sensors send their data to E2, and so on. The idea is to keep the edge part of the processing as close as possible to the sources of data and reduce the distance data needs to travel before reduction. The data that arrive at one node are reduced together on that node; consequently, there is one AE for each of the edge nodes.

The similarity-based scenario sensors based on their similarity. For example, all gyroscopes could represent one group and all accelerometers another one. This will result in more homogeneous data groups, but it can also increase the distance of sensors from the edge nodes. As with the location-based scenario, there is one AE for each of the edge nodes.

2) Nonreversible: Nonreversible approaches include those without the way of reproducing the original data after the data have been reduced. Here, considering the vector magnitude which is suitable for sensors that measure values in multidimensional space such as accelerometers and gyroscopes. The dimensionality is reduced as follows:

$d = \text{sqrt}\,(x^2 + y^2 + z^{2)}$ (2) Here x, y, z are the measurements in Euler coordinates and d is the vector magnitude. Consequently, vector magnitude reduces dimensions in 3: 1 ratio.

C. Cloud ML

The reduced data are sent from the edge to the cloud for further ML processing. As illustrated in Fig., there are two possible ways to carry out the ML task. The first option is to reproduce original data and use these reproduced data for the ML task. This is possible only if the reversible technique was used for data reduction.

The second option is to carry out the ML task directly on the reduced data, which works for both reversible and nonreversible reduction techniques. As the reproduced data have a larger number of features, training ML model for such data is more computationally expensive than training ML model for the reduced number of features.

5.Conclusion

Traditionally, ML with IoT data was done by transferring data to the cloud or another centralized system for storage and processing. With the explosion of connected devices, this would lead to increased latencies and it would put a strain on communication networks. Proposed work explored merging edge and cloud computing for ML with IoT data with the objective of reducing network traffic and latencies. Three scenarios were examined-all sensors together consider all the data at once, location-based scenario groups data according to the IoT device locations, and similarity based scenario groups data according to the similarities of sensors. The evaluation was carrying out on the HAR task considering two nonreversible approaches, AE and PCA, and one nonreversible approach, vector magnitude.

References

[1] CISCO, Cisco Global Cloud Index: Forecast and Methodology, 2016–2021. Accessed: Dec.27, 2019. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/globalcloud-index-gci/white-paper-c11-738085. html
[2] A. L'heureux, K. Grolinger, H. F. Elyamany, and M. A. M. Capretz, "Machine learning with Big Data: Challenges and approaches," IEEE Access, vol.5, pp.777–797, 2017.
[3] H. Cai, B. Xu, L. Jiang, and A. V. Vasilakos, "IoT-based big data storage systems in cloud computing: Perspectives and challenges," IEEE Int. Things J., vol.4, no.1, pp.75–87, Feb.2017.
[4] M. Chen and Y. Hao, "Task offloading for mobile edge computing in software defined ultra-dense network," IEEE J. Sel. Areas Commun., vol.36, no.3, pp.587–597, Mar.2018.
[5] R. Roman, J. Lopez, and M. Mambo, "Mobile edge computing, Fog et al.: A survey and analysis of security threats and challenges," IEEE Commun. Mag., vol.78, no.2, pp.680–698, Jan.2018.
[6] H. El-Sayed et al., "Edge of things: The big picture on the integration of edge, IoT and the cloud in a distributed computing environment," IEEE Access, vol.6, pp.1706–1717, 2018.
[7] A. Kumari, S. Tanwar, S. Tyagi, N. Kumar, R. M. Parizi, and K. R. Choo, "Fog data analytics: A taxonomy and process model," J. Netw. Comput. Appl., vol.128, pp.90–104, 2019.
[8] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in Proc Conf. Comput. Vision Pattern. Recognit., 2018, pp.8697–8710.
[9] H. F. Nweke, Y. W. Teh, M. A. A.-G., and U. R. Alo, "Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: State of the art and research challenges," Expert Syst. Appl., vol.105, pp.233–261, 2018.
[10] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," IEEE Int. Things J., vol.3, no.5, pp.637–646, Oct.2016.
[11] A. Ahmed and E. Ahmed, "A survey on mobile edge computing," in Proc.10th Int. Conf. Intell. Syst. Control, 2016, pp.1–8.
[12] N. Abbas, A. Zhang, Y. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," IEEE Int. Things J., vol.5, no.1, pp.450–465, Feb.2018.
[13] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol.521, no.7553, pp.436–444, 2015.
[14] G. G. Jia, G. G. Han, A. Li, and J. Du, "SSL: Smart street lamp based on fog computing for smarter cities," IEEE Trans. Ind. Informat., vol.14, no.11, pp.4995–5004, Nov.2018.
[15] P. Hu, H. Ning, T. Qiu, Y. Zhang, and X. Luo, "Fog computing based face identification and resolution scheme in internet of things," IEEE Trans. Ind. Informat., vol.13, no.4, pp.1910–1920, Aug.2017.
[16] L. Li, K. Ota, and M. Dong, "Deep learning for smart industry: Efficient manufacture inspection system with fog computing," IEEE Trans. Ind. Informat., vol.14, no.10, pp.4665–4673, Oct.2018.
[17] Y. Wang, K. Wang, H. Huang, T. Miyazaki, and S. Guo, "Traffic and computation co-offloading with reinforcement learning in fog computing for industrial applications," IEEE Trans. Ind. Informat., vol.15, no.2, pp.976–986, Feb.2019.
[18] S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang, and W. Wang, "Asurvey on mobile edge networks: Convergence of computing, caching and communications," IEEE Access, vol.5, pp.6757–6779, 2017.
[19] Y. Mao, C. You, J. Zhang, K. Huang, andK. B. Letaief, "Asurvey on mobile edge computing: The communication perspective," IEEE Commun. Surv. Tut., vol.19, no.4, pp.2322–2358, Fourth quarter 2017.
[20] N. Mohamed, J. Al-Jaroodi, I. Jawhar, S. Lazarova-Molnar, and S. Mahmoud, "Smartcityware: A service-oriented middleware for cloud and fog enabled smart city services," IEEE Access, vol.5, pp.17 576–17 588, 2017.
[21] B. Tang, Z. Chen, G. Hefferman, S. Pei, T. Wei, H. He, and Q. Yang, "Incorporating intelligence in fog computing for big data analysis in smart cities," IEEE Trans. Ind. Informat., vol.13, no.5, pp.2140–2150, Oct.2017.
[22] X. Wang, Z. Ning, and L. Wang, "Offloading in internet of vehicles: A fogenabled real-time traffic management system," IEEE Trans. Ind. Informat., vol.14, no.10, pp.4568–4578, Oct.2018.
[23] J. He, J. Wei, K. Chen, Z. Tang, Y. Zhou, and Y. Zhang, "Multi-tier fog computing with large-scale IoT data analytics for smart cities," IEEE Internet Things J., vol.5, no.5, pp.677–686, Apr.2018.
[24] A. M. Rahmani et al., "Exploiting smart e-health gateways at the edge of healthcare Internet-of-things:

A fog computing approach," Future Gener. Comput. Syst., vol.78, no.2, pp.641–658, 2018.
[25] P. Ritrovato, F. Xhafa, and A. Giordano, "Edge and cluster computing as enabling infrastructure for internet of medical things," in Proc. IEEE 32ndInt. Conf. Adv. Inf. Netw. Appl., 2018, pp.717–723.
[26] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, "Deep learning for sensor based activity recognition: A survey," Pattern Recognit. Lett., vol.119, pp.3–11, 2019.
[27] E. Zdravevski et al., "Improving activity recognition accuracy in ambient assisted living systems by automated feature engineering," IEEE Access, vol.5, pp.5262–5280, 2017.
[28] A. Ferrari, D. Micucci, M. Mobilio, and P. Napoletano, "On the personalization of classification models for human activity recognition," IEEE Access, vol.8, pp.32 066–32 079, 2020.
[29] J.-H. Li, L. Tian, H. Wang, Y. An, K. Wang, and L. Yu, "Segmentation and recognition of basic and transitional activities for continuous physical human activity," IEEE Access, vol.7, pp.42 565–42 576, 2019.
[30] M. N. Fekri, A. M. Ghosh, and K. Grolinger, "Generating energy data for machine learning with recurrent generative adversarial networks," Energies, vol.13, no.1, 2020, Art. no.130.
[31] N. Kambhatla and T. K. Leen, "Dimension reduction by local principal component analysis," Neural Comput., vol.9, no.7, pp.1493–1516, 1997.
[32] C. Banos et al., "Design, implementation and validation of a novel open framework for agile development of mobile health applications," Biomed. Eng. Online, vol.14, no.2, 2015, Art. no. S6