JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue

JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)

An International Scholarly Open Access, Peer-reviewed, Refereed Journal

A REVIEW OF EDGE MACHINE LEARNING ALGORITHM FOR ARTIFICIAL INTELLIGENCE ENABLE SYSTEM AND IOT DEVICES

Dr. Vineetha KR, Lincy John
Associative Professor, MCA Scholar
Department of MCA,Nehru College of Engineering and Research Centre,Thrissur
vpvprakash@gmail.com
lincyjohnz891@gmail.com

Abstract: In a few years, billions of connected gadgets will be installed in our homes, cities, automobiles, and industries, transforming the globe. Users and the environment will interact with devices with restricted resources. Many of these devices will use machine learning models to decipher the meaning and behavior of sensor data, as well as to make accurate predictions and judgments. The bottleneck will be the high number of linked things, which may cause the network to become congested. As a result, machine learning techniques must be used to include intelligence on end devices. By allowing computations to be done close to data sources, deploying machine learning on such edge devices reduces network congestion. The purpose of this paper is to provide an overview of the field. By allowing computations to be done closeto data sources, deploying machine learning on such edge devices reduces network congestion. The goal of this paper is to give a review of the key strategies for executing machine learning models on low- performance hardware in the Internet of Things paradigm, paving the way for the Internet of Conscious Things. The main purpose of this paper is to define the state of the art and envisage development needs forsystems that apply edge machine learning on Internet of Things devices. Furthermore, an example of edge machine learning implementation on a microcontroller, also known as machine learning on the edge, will be presented.

IndexTerms **- Artificial intelligence; machine learning; Internet of Things; edge devices; deep learning.**

I. INTRODUCTION

In recent years, the Internet of Things (IoT) scenario has attracted a lot of attention. It refers to the software and technology infrastructure that connects the physical world to the Internet. IoT devices often have the low processing power, limited memory, and the ability to generate massive amounts of data. In our homes, cities, automobiles, and industries, low-power and networked systems, mostly sensors, will beused. Due to the growing number of connected devices, only-cloud processing may become difficult, resulting in increased latency, reduced bandwidth, and privacy and reliability issues. Edge computing refers to computations that are performed as close as possible to data sources rather than at remote sites.An effective artificial intelligence (AI) algorithm could be used to make this scenario possible. The implementation of machine learning methods on devices with limited computational power can be utilized to advance the IoT area by enabling edge computing. Fog computing, a similar phrase, provides anarchitecture in which the cloud is stretched to be closer to IoT end devices, reducing latency and enhancing security by executing calculations near the network edge. As a result, the objective of fog computing is to bring the processing phase closer to where the data is generated, with the only distinctionbeing the location of the "intelligence." The processing phase of fog computing takes place on a LAN (local area network) level, in a fog node or IoT gateway. Data is mostly processed on the devices to which the sensors are attached in edge computing (physically very close to the sensors). Because of the lower energy demand associated with data transmission, the closer the sensor is to the user, the better it is in terms of privacy and power consumption. This study focused on machine learning systems on edge devices. Section 2 compares the various machine learning

algorithms that can be used in edge computing. The process of delivering machine learning to the edge is examined in Section 3. Edge server-based architectures are described in Section 4,and wireless standards for AI-enabled IoT devices are introduced in Section 5. Section 6 details the distinctions between the collaborative computation alternatives and offers edge-specific solutions for offloading strategies. Section 7 discusses user privacy and how to safeguard it when uploading data. In ML design, Section 8 describes the edge implementations of the training phase. In Section 9, a machine learning "Hello World"implementation is offered as an example of edge machine learning implementation. Finally, conclusions are reached in section

II. LITERATURE SURVEY

Edge Machine Learning (Edge ML) is one of the most widely discussed technological developments since the Internet of Things (IoT), and for good cause. The advent of IoT resulted in an avalanche of Smart Devices connected to the Cloud, but the network was not yet prepared to handle this surge in demand. Companies ignored crucial difficulties with Cloud computing, such as security, since cloud networks were crowded.

Edge intelligence is a set of connected systems and devices that use artificial intelligence to gather, cache, process, and analyse data in locations close to where it is captured. Edge intelligence aims to improve the quality and speed of data processing while also protecting the data's privacy and security. This topic of research has seen tremendous growth over the last five years, despite its recent emergence (from 2011 to today). We give a complete and comprehensive review of the literature on edge intelligence in this study.

Big data has lately seen a drastic change in data sources from mega-scale cloud data centers to increasingly widespread end devices, such as mobile and IoT devices, as a significant driver for AI development. Big data, such as online shopping records, social media information, and corporate informatics, was traditionally created and housed primarily in massive data centers

Machine learning, particularly deep learning, has become the de facto processing paradigm for intelligently handling these massive data sets. However, the resource-constrained environment of edge devices, owing to their poor computational capabilities and restricted energy budget, makes them a difficult platform for deploying needed data analytics, especially in real-time applications.

With the development of artificial intelligence (AI) in a wide range of disciplines, it is expected that AI-powered edge computing would be able to tackle new issues by fully utilizing the edge big data potential. Edge AI, or edge intelligence, as a result of this new inter-discipline, is attracting a lot of attention. However, research on edge intelligence for IoT is still in its early stages, and both the computer system and artificial intelligence communities would benefit from a dedicated.

III.METHODOLOGIESIES

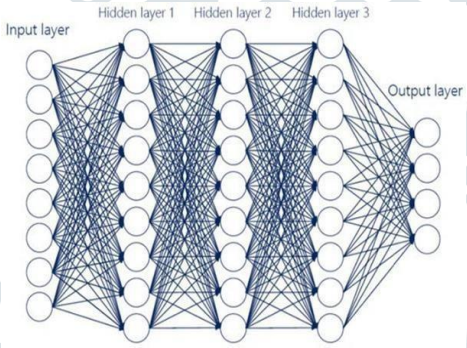
MACHINE LEARNING ALGORITHMS

We now discuss ML algorithms that could be used in resource-constrained settings at the edge of the network. The machine learning algorithms introduced in the next paragraphs are the most used in thepapers that afford the problem of bringing AI into devices with resource-constrained hardware.

Deep Learning

A deep learning model can be thought of as a set of weights and biases together. An optimization function (ADAM optimization algorithm) changes these parameters. commonly employed) based onan objective function (whether the learning is, a loss function or a reward function) respectively, supervised or reinforced) that assesses the model's prediction ability. Following. During the training phase, the AI algorithm finds an underlying pattern in the data and predicts it. A value that is a function of the data in the inputs Various training phases can be distinguished depending on the training phase.

Figure1. Deep Neural Network (DNN) example.



RNN, GAN, K-NN

RNNs (recurrent neural networks) are a specific sort of NN. The output values of a higher layer are used as input for a lower layer in this form of NN. Because of this interconnectedness, one of the layers can be used as state memory. It allows us to model dynamic temporal behavior by providing a temporal sequenceof data as input. This makes them suitable for jobs involving predictive analysis of data sequences, such as handwriting or speech recognition. LSTM (long short-term memory) is a type of RNN. A cell, an input gate, an output gate, and a forget gate make up an LSTM unit. The cell keeps track of the values over time, whilethe gates control the flow of data into it.

The generative adversarial network (GAN) is another type of NN. There are two networks in them: a generator and a discriminator. After learning the data distribution from a training dataset of real data,the first generates data. The second is in charge of distinguishing genuine data from data generated bythe generator.

The K-nearest neighbor's algorithm (K-NN) is a pattern recognition technique that is based on the properties of objects that are close to the one being analyzed. Both classification and regression issues aresolved using this strategy. The most innovative modified version of k-NN is Proto NN, which enables to implementation of the algorithm on hardware-constrained devices. It uses a k-NN algorithm. The primary issues of the training data size (the algorithm creates predictions using the whole datasets), the predictiontime, and the distance metric are all variables in K-NN for edge computation. To address these concerns, Proto

NN uses a smaller training sample that excludes irrelevant information. The dataset is then projected to a low-dimensional matrix, and all data points are jointly learned. Gupta et al. used an ArduinoUno to test Proto NN's performance on 14 datasets and found that it had nearly the same classification accuracy as the state-of-the-art.

Tree based Machine learning Algorithms

Classification and regression problems, which are highly common in the IoT industry, are solved using tree-based ML techniques. However, because of the devices' low resources, the conventional tree methods could not be implemented. Bonsai is a newly developed algorithm. The tree approach is optimized for IoT devices with limited resources, maintaining prediction accuracy while reducing the model size and prediction costs. It first learns a single sparse tree to reduce the size of the model, then makes nonlinear predictions using the internal and leaf nodes. Bonsai eventually learns sparse matrices, projecting all input into a low-dimensional space where the tree can be learned. This enables the technique to be used on small devices such as IoT devices.

The referenced implementation was performed on an Arduino board with an 8-bit ATmega328P microcontroller running at 16 MHz, 2 kB of static random access memory (SRAM), and 32 kB of read-only flash memory, as well as on a BBC Micro: Bit with an ARM architecture 32-bit Cortex running at 16 MHz,16KB of SRAM, and 256 kB of flash memory.

SVM

The SVM is one of the most extensively used ML algorithms at the embedded level. SVM is a supervised learning technique that can be used to solve problems in classification and regression. By establishing an ideal hyperplane that separates all classes, the method distinguishes between two or more classes of data(Figure 2a). The data closest to the support vectors to the hyperplane, which, if removed, would result in the hyperplane's redefinition. As a result, they are regarded as the dataset's most important components. The Hinge loss is usually utilized as the loss function, while the descending gradient approach is employedas the optimization function. Sensors 2020, x FOR REVIEW BY PEERS 6 of 33 accuracies while reducing model size and cost of prediction It first learns a single sparse tree to reduce the size of the model, then makes nonlinear predictions using the internal and leaf nodes. Bonsai eventually learns sparse matrices, projecting all input into a low-dimensional space where the tree can be learned. This enables the technique to be used on small devices such as IoT devices. The implementation in question was carried out on an Arduino board populated with an 8-bitATmega328P microcontroller with 16 MHz operating frequency, 2 kB of Static Random access memory (SRAM), and 32 kB of read-only flash memory and on BBC Micro: B which has an ARM architecture 32-bit Cortex.

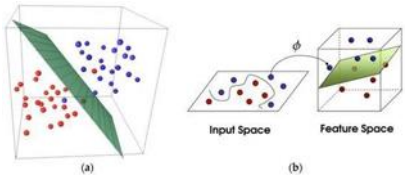


Figure.2 (a) Hyperplane that separates two classes of data, (b) kernel trick.

PUSHING MACHINE LEARNING TO THE FRONTLINE

Architectures

Different architectures for fast-performing model inference have been proposed to meet latency constraints. Thestudy concentrated on three key structures (shown in Figure 3):

(1) on-device computation, in which DNNs are run on the end device;

(1) edge server-based architectures, in which data is transmitted from end devices to edge servers for processing;

(3) joint computation, which includes cloud processing.

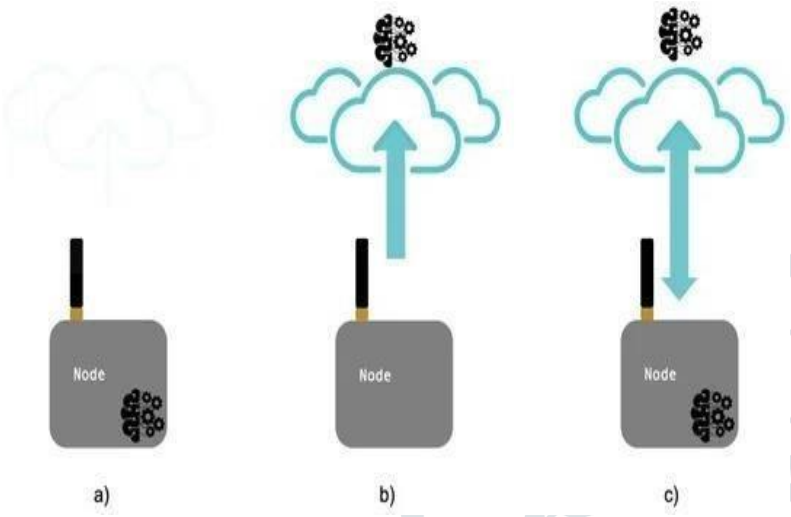


Figure 3. (**a**) On-device computation, (**b**) edge server-based architectures, and (**c**) joint computation.

Model and Hardware

Several research articles have looked into the idea of bringing artificial intelligence to low-resource devices, and efforts have been made to reduce the model's inference time on the device. To put an AImodel on an embedded device, ML developers must consider the appropriate hardware for model creation and compression.

Model Design

In the DNN model, ML developers focus on developing models with fewer parameters, lowering memory and execution latency while maintaining high accuracy. Mobile Nets and Squeeze Net, which were created for computer vision tasks, are two efficient models built primarilyto perform a NN on devices with minimal processing capacity and power.

Model Compression

There are two basic techniques to minimize the network: lower precision (fewer bits per weight) and fewer weights (pruning). With a minor reduction in accuracy, post-training quantization minimizes computational power requirement and energy usage. Note that post-quantization is a technique that is used after the model has been trained, although it can also be used before training. Both solo and combined techniques for quantification and trimming have been considered. These two algorithms formthe foundation of NN compression, which has spawned a slew of new techniques. Deepa T describes a method

for pruning commonly used deep learning structures in IoT devices, and the pruned DNN can bedeployed on the edge right away.

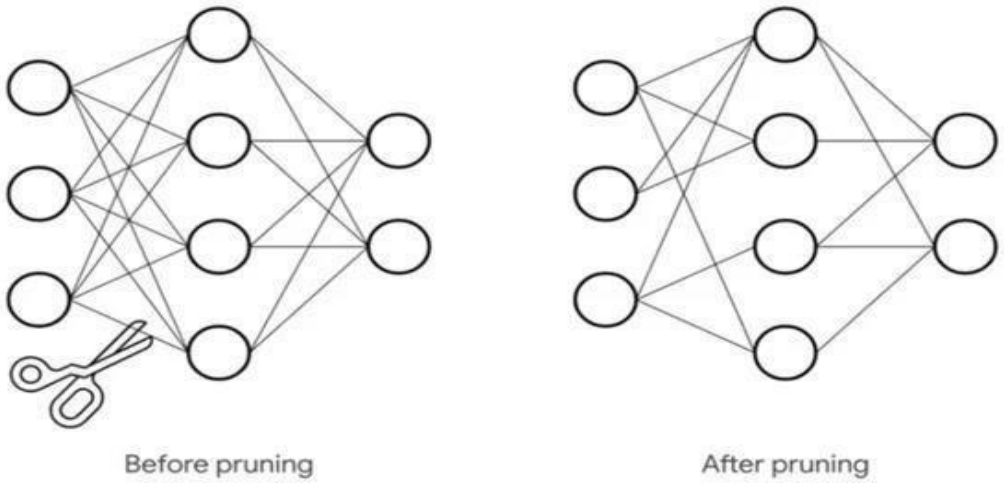


Figure 4. Pruning effect on the network.

SELECTION OF HARDWARE

When running a model on an edge device, the algorithm chosen is critical. However, this must be combined with the best possible hardware. The accuracy, energy consumption, throughput, and cost metrics will be used to select the hardware. The programmability and size of the NN, on the other hand, are intimately related to energy efficiency. By "programmability," we imply the model's ability to react tochanges in the context, such as changing the weights as the scenario changes. The number of layers that the processor must support is what we mean by "NN size." The scenario's magnitude and variety necessitate an increase in computing time. The large size of the NN, in particular, increases the number of data, and programmability necessitates accessing memory, reading the weight value, and modifying it. This usuallyresults in an increase in energy usage. The number of operations required per unit of time is referred toas "throughput," while the quantity of memory required is referred to as "cost."

ARCHITECTURE BASED ON THE EDGE SERVER

Although the strategies discussed in the preceding sections enable us to execute AI algorithms on end devices, installing powerful DNNs on small devices remains difficult (e.g., decision making and real-timeexecution). It may be required to shift computations from end devices to more powerful entities in some cases. Because the edge server is close to the users, it may be the ideal approach for resolving calculation-related issues at optimal times. The simplest approach to make use of the edge server is to offload all computing from end devices to it: The end devices will send their data to a nearby edge server, which will process it and return the results.

Obviously, if data processing is delegated to an edge server, that server will have additional edge devices. This should resolve the issue of shared resources. As a result, the developer should seek out the best trade-offs between accuracy, latency, and other performance indicators like the number of requests fulfilled. A realistic alternative could be to distribute the computation across a hierarchy of edge and cloudservers, with all DNN hyperparameters tuned together. The proposed approach in Mainstream considers asimilar issue, but it uses transfer learning to reduce the computational resources spent by each request.

Transfer learning allows different applications to share the DNN model's lower layers while computinghigher layers unique to each application, lowering the overall amount of data.

WIRELESS STANDARDS FOR IoT DEVICES WITH AI

The communication protocol wastes the majority of the energy in the IoT device. Any superfluous data that is transported, kept, or processed appears to be a potential energy waste. As a result, good algorithms need to be supported with good communication protocols. Developers can employ a variety ofcommunication protocols depending on the situation. This is because we can discriminate between protocols that allow us to send a small quantity of data over long distances with little energy consumptionand protocols that can send a large amount of data over long distances with a lot of energy consumption. Among the different communication methods, we have BLE, an example of which shows the design and optimization of a smart sensor powered by 2.4 GHz RF power and capable of infrared-based motion detection and BLE communication. Bluetooth wireless technology is widely utilized, withBluetooth 5 being introduced, which uses less power and offers mesh topology, allowing large-scale devicenetworks and many-to-many communications.

One of the most common IoT communication standards is ZigBee. It is based on the IEEE 802.15.4 WPAN(wireless personal area network) standard, and its principal application is in wireless sensor networks (smart energy and home automation). Smart cities, agricultural, automotive, and health care are all possible applications. ZigBee runs primarily at 2.4 GHz, although it also supports the ISM bands of 868 MHz and 916 MHz.

Z-Wave, a sub-GHz mesh network technology commonly used for security systems, home automation, and lighting controls is another option. Z-Wave, like Zigbee, is a low-power wireless technology that transmits small amounts of data across short and medium distances..

ANT is a proprietary protocol for low-bit-rate and low-power networks that operates in the 2.4 GHz band.It supports point-to-point, star, tree, and mesh networks, with each channel supporting up to 65,533 nodes. It was first utilized in sports and fitness sensors before being adapted for home automation and industrial use. ANT+ is a standardized layer built on top of the ANT protocol that allows devices to communicate with one another.

Wi-Fi networks (IEEE 802.11) use an access point (AP) as an Internet gateway and provide adequate datacapacity and coverage within buildings. Due to the size and complexity of Wi-Fi and TCP/IP software, as well as the high power consumption that makes it unsuitable for use with battery-powered devices, integrating Wi-Fi connectivity into devices with low computational performance was quite expensive until recently. New gadgets, on the other hand, support Wi-Fi and TCP/IP software while consuming less power.

802.11ax is a more current form of Wi-Fi technology that supports greater transmission speeds and adds power-saving features like TWT, making it more appealing for IoT applications. It also has capabilities thatexpand the range and allow the channel to be partitioned into smaller subchannels to reduce data rates while increasing the number of devices that can be reliably connected to one access point, letting it scale up to thousands of devices.

RFID (radio-frequency identification) is a technique that employs sub-GHz ISM bands to convey signals from devices that do not have batteries. NFC (near-field communication) is a communication protocol for very close proximity. It uses the 13.56 MHz frequency band and is intended to exchange data with anotherNFC device, providing a bidirectional connection. It's only appropriate for niche IoT applications because ofits low data rate and small communication distance.

Lora (long-range) is a low-power wide-area network (LPWAN) technology. It is based on spread spectrummodulation techniques and could be used to enable the Internet of Things. describes a machine learningsystem for edge devices that uses Loral as a low-power communication protocol. Loral's use of machine learning allows it to cut transferred data by 512 times while also extending battery life by three times in that circumstance. The cloud is currently the most popular method of data processing, but the transfer ofmassive amounts of data necessitates frequent device recharging, offsetting the IoT's advantages. Edge processing is also required by IoT devices due to bandwidth, latency, and privacy concerns. Under these circumstances, efficient data reduction and local processing must be combined with long-range, low-bandwidth transmission methods, which Loral may provide.

LTE (long-term evolution), or "4G LTE," is a cellular broadband communication standard based on GSM/EDGE and UMTS/HSPA technologies. LTE began to compete with burgeoning IoT technologies suchas BLE, narrowband Internet of Things (NB-IoT), ZigBee, and Loral after its launch. Although 4G has increased cellular network capabilities, it is still not entirely suitable for IoT applications.

5G networks and standards are expected to address issues that currently plague 4G networks. The fifth generation of mobile, cellular technology, networks, and solutions is referred to as 5G. Although it was not specifically designed for the Internet of Things (IoT), it will be a major driver of IoT growth. The 5G IoT is a new intelligent network based on 5G connectivity that is designed to connect sensing areas (sensors) and processing canters (clouds) using AI algorithms.

The 5G fulfills the needs of the IoT:

- high data rate.
- fine-grained and scalable networks to boost network scalability.
- very low latency; - extended battery life to handle billions of low-power and low-cost IoT devices.

COMPUTATION WITH OTHERS

Although an edge server can speed up DNN processing, it is not necessarily required for edge devices torun DNNs on servers. Three offloading situations will be presented (Figure 6):

(1) partitioned DNN partial offloading (the decision is what fraction of the DNN calculations should beoffloaded)

(2) hierarchical architectures (offloading is done over a combination of edge devices, edge servers, andthe cloud)

(3) distributed computing methodologies (the DNN computation is distributed across multiplepeer devices).

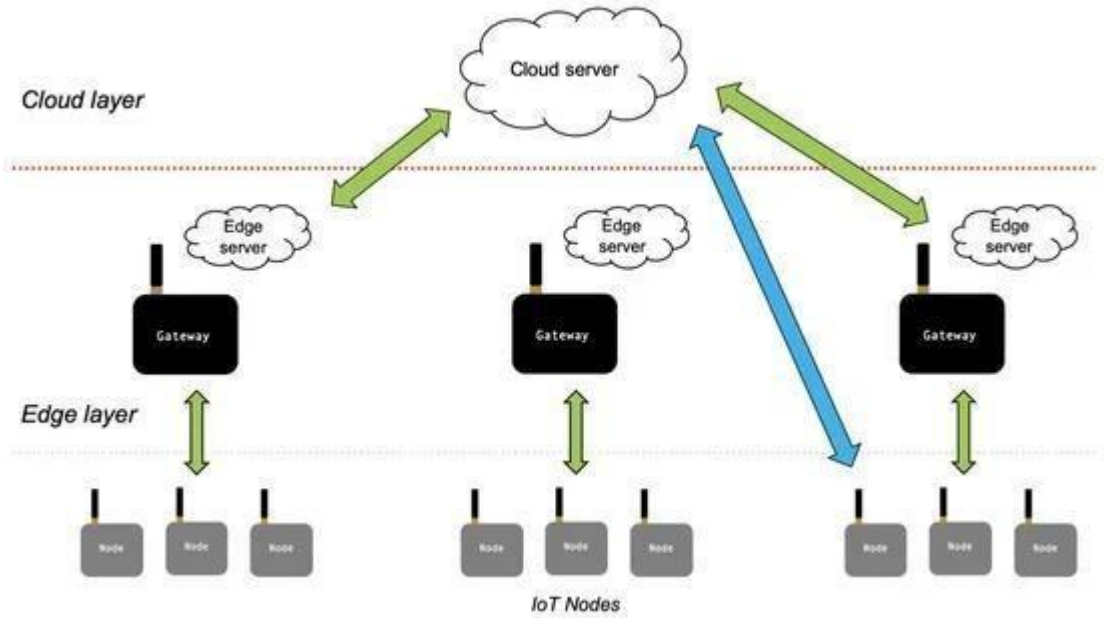


Figure 5. Joint computation among devices, edge, and cloud servers.

Partial Offload

Some layers are computed on the device, while others are computed via the edge server or the cloud in model partitioning schemes. Due to the processing cycles of other edge devices, this strategy may be ableto reduce latency. Indeed, once the first few layers of the DNN model are computed, the intermediate results are relatively modest, and the output may be delivered over the network to an edge server much faster than the original raw data. The moment at which the network must be partitioned is crucial, and one approach that can be employed is Neurosurgeon. It's a lightweight scheduler that automatically divides DNN processing between mobile devices and data centers at the NN layer level. As a result, it determines where to partition the DNN layer-by-layer while taking network conditions into account.

Hierarchical Architectures

The machine learning method can be run on both edge devices and in the cloud. Delegating the computational task to the cloud may result in a delay issue. Instead, by utilizing powerful computing cloud resources, the entire processing time could be reduced. Li et al. for example separated the DNN model intotwo parts: After receiving the input data, the edge server computes the DNN model's initial layers (lower levels), while the cloud computes the DNN model's higher layers. After processing, the cloud returns the final results to the end devices. The cloud assists the edge server with the harder computations in this way. DDNN also enables quick and localized inference at the edge and end devices by utilizing shallow sections of the NN. DDNNs improve data fusion from network sensors, system fault tolerance, and user privacy due to their distributed nature. The fact that the edge server covers a small geographical area is a frequent aspect of edge techniques, therefore the input data and, consequently, the DNN outputs may be comparable.

Distributed Computing

The network is offloaded to more powerful entities such as edge devices or the cloud in a hierarchical situation. DNN computations can be distributed among numerous peer edge devices in a distributed fashion,as in Deep Things. It divides DNN executions among end devices like Raspberry Pi and Android smartphones. The DNN partitioning is determined by the end devices' processing and memory capability.

CHALLENGES

Data is sent via the network (e.g., from end device to edge server or from the edge server to cloud) in bothedge server-based designs and joint computation, and it may contain sensitive information. This may cause privacy concerns. Edge servers, as previously said, work locally in a geographically confined area. As a result, the data's source is essentially known. Although machine learning on edge devices reduces data on the network and thus improves privacy, the system can be improved further with additional approaches such as data noise or cryptographic techniques.

Increase Noise to Data

Adding noise to the samples submitted to the network during inference is one solution. Wang et al. used asmaller DNN to extract features locally on the edge device, introduce noise to the data, and then upload the features to the cloud for additional inference processing by a more powerful DNN. The DNN in the cloud is pre-trained with noisy samples so that the noisy inference samples uploaded from end devices canbe classified with high accuracy at test time. A differential privacy technique underpins this.

Analysis of cryptography

To compute the inference with a high level of privacy, cryptographic approaches might be utilized. Secure computing Another option for secure computation is multiparty computation. Multiple machines collaborate and communicate in multiple rounds to collaboratively calculate a result in insecure multi-party computation. Secure multiparty computation focuses on the privacy of intermediate computation processes, but communication complexity is usually the barrier. Aims to ensure that an inference result is sent to the end device without the edge server discovering anything about the DNN model, and vice versa.Homomorphic encryption, as used in Crypto Nets is one way of safe computation in which communicated data are encrypted and computation can be conducted on the encrypted data. The DNN is turned into Crypto Nets, which are low-degree polynomials that approximate the common activation functions and operations in a DNN, ensuring homomorphic encryption. The compute times of homomorphic encryption, on the other hand, tend to be a bottleneck.

IV TRAINING AND PREPROCESSING

So far, most discussions of edge computing and deep learning have assumed that a deep learning modelhas previously been trained offline on a prebuilt dataset. The training techniques and hardware for the edge field are discussed in this section. Typically, end-device raining data would be transferred to the cloud, which would then execute the training and then return the trained model to the edge devices.

When data is kept at the edge, it is useful for maintaining privacy while simultaneously reducing network bandwidth requirements.

Preparation of algorithm

For training an edge–cloud-based DL model, model parameters and other data must be exchanged between edge devices and cloud servers. However, as the training model grows in size, more data must betransferred between edge devices and servers. A training model's high network connection cost is a bottleneck, necessitating local edge training implementation. A mobile computer system for DNN applications is an example of local networks (MoDNN). MoDNN scans each layer of the DNN model to identify layer types using a pre-trained DNN model. The input layer is separated using the biased one- dimensional partition (BODP) method if the layer is convolutional. In-edge AI is a system that enables improved collaboration between devices and edge nodes by allowing them to exchange learning parameters for better model training and inference. Mobile computing combines deep reinforcement learning techniques with federated learning. Teerapittayanon and colleagues The DDNN was trained on a cloud server among many devices (including edge devices and the cloud), with the most powerful device training

the network. Because DDNNs have several departure locations, training them is tough. To overcome this problem, during backpropagation, the network was jointly trained by combining losses fromeach exit point. Training could also be done on a trimmed model: Chandakkar and his colleagues devised a new architecture for retraining a pruned network on an edge device. On the original data, a complete DNNis trained for an epoch (where an entire dataset is traversed forward and backward through the DNN). A user-defined threshold value is also used to do layer-wise, magnitude-based weight reduction. This method dramatically decreases the computational complexity of a DNN model by eliminating connections,making it appropriate for use on a device with minimal resources. Unfortunately, any trimming procedure diminishes a model's accuracy. To address this problem, this method identifies the indices of the most essential weights for a given feature and prevents these items from being pruned. Finally, because these procedures are repeated, the trimmed DNN network is used to train the following epoch.

Tao and Li proposed a new method termed edge stochastic gradient descent (eSGD) to reduce communication costs while maintaining excellent model accuracy. All edge devices conduct trainingtasks independently with independent data in this technique, and the edge devices' gradient values are communicated to the cloud servers. After receiving the gradients from the end devices, the server averages the gradients to make them uniform. It then uses this average value toupdate the parameters. For the next training step, these revised parameters are delivered back to the edge devices. This is known as parameter synchronization. Unfortunately, this method of gradient selection reduces model accuracy. To maintain a high level of training accuracy, the eSGD employs two mechanisms:

- Important' updating: Only a tiny fraction of the gradient coordinates must be changed after each mini-batch. The server will then update the main gradients determined by the algorithm. This method drastically lowers the cost of communication.
- This mechanism is used to track and accumulate out-of-date residual gradients, which helps to preventthe low convergence rate produced by the prior key updating method.

To 90% of the gradient size of a CNN, the model can be reduced with the eSGD. High gradient decreasing,unfortunately, leads to poor precision. In their experiments, Tao and Li employed the Modified National Institute of Standards and Technology (MNIST) database and reported an accuracy of 91.22 percent with a gradient drop of 50%.

Evaluation of hardware

Updates to the neural network and the computation of sophisticated algorithms cannot be left entirely to microcontrollers. The field-programmable gate array (FPGA) and graphical processing unit (GPU) consume too much power (although the FPGA is still a better choice than the GPU because it is more adaptable and consumes less power), but they are ideal for training NNs or running complex algorithms [186]. GPUs perform MACs in parallel using temporal architectures like SIMD (single instruction multiple data) or SIMT (single instruction multiple threads), and there are software libraries for GPUs that optimize matrix multiplications, such as NVIDIA CUDA® Basic Linear Algebra Subprograms (cuBLAS) and NVIDIA CUDA® Deep Neural Network library (cuDNN). On these platforms, matrix multiplications can be improved further by applying transforms to the data to reduce the number of multiplications. The fast Fourier transform (FFT) is a well-known technique for reducing the number of multiplications from O((N2o) (N2f)) to O(N2olog2No), where the output size is No*No and the filter size is Nf *Nf; however, the benefits of FFTs diminish as the filter size increases. Strassen and Winograd are two other ways. Some highly fascinating devices, such as the Hailo-8 DL from the Hailo firm, have just emerged. The Hailo-8 DL is a CPU designed for high-performance deep learning on end devices with minimal power consumption, size, and cost. It is very fast (26 tera-operations per second) and efficient, as well as highly versatile (reprogrammable). Google created an ASIC (application-specific integrated circuit) for the TensorFlow library TPU (Tensor processing unit), which has a computing capacity of 180 teraflops. These are examples of IA accelerators, such as the NPU, which is a class of microprocessors meant to accelerate artificial neural

networks, automated vision, and machine learning algorithms for robotics, IoT, and other data-driven applications. While hardware DNN accelerators are still relatively new, two design branches have already emerged. The first generation of accelerators focused solely on data flow, disregarding memory energy use. The second attempted to reduce the amount of energy used by memory access. ConvNet Processor (CNP) Neuflow and dynamically configurable (DC) CNN are two types of accelerators that use customized logic to transfer convolution to hardware with higher parallelism and flexibility. The second generation of accelerators focused on memory transfer and data mobility optimization. Researchers are realizing that memory access and data movement are more important than matrix products between layers as modern NNs grow larger. DianNao implements an array of multiply-add unitsto map huge DNNs onto its basic architecture among these accelerators (TPU is included in this class of accelerators). It has a specially designed on-chip buffer to reduce DRAM traffic. By having all weights on-chip, Da DianNao and ShiDianNao avoid DRAM access. The Movidius stick for edge computing is an interesting AI accelerator since it enables adding deep learning capabilities to existing computing platforms simple. It's primarily for edge computer vision workloads and allows CNNs to be used on low-power applications that demand real-time inference. A comprehensive guide on its application is available.

The second attempted to reduce the amount of energy used by memory access. ConvNet Processor (CNP) Neuflow and dynamically configurable (DC) CNN are two types of accelerators that use customized logic to transfer convolution to hardware with higher parallelism and flexibility. The second generation of accelerators focused on memory transfer and data mobility optimization. Researchers are realizing that memory access and data movement are more important than matrix products between layers as modern NNs grow larger. DianNao implements an array of multiply-add unitsto map huge DNNs onto its basic architecture among these accelerators (TPU is included in this class of accelerators). It has a specially designed on-chip buffer to reduce DRAM traffic. By having all weights on-chip, Da DianNao and ShiDianNao avoid DRAM access. The Movidius stick for edge computing is an interesting AI accelerator since it enables adding deep learning capabilities to existing computing platforms simple. It's primarily for edge computer vision workloads and allows CNNs to be used on low-power applications that demand real-time inference. A comprehensive guide on its application is available.

V.CONCLUSION

Machine learning on IoT devices minimizes network congestion by allowing computations to be done closeto data sources, maintaining data privacy while uploading, and lowering battery consumption for continuous wireless transmission to gateways or cloud servers. The goal of this paper was to give a review of the key methodologies for executing machine learning models on low-performance hardware in the Internet of Things paradigm, paving the way for the Internet of Conscious Things. The major purpose of this paper was to define the state of the art and envisage development needsfor systems that utilize edge machine learning on IoT devices.The review compared the ML algorithms that can be implemented in edge computing and focused on ML systems deployed on edge devices. Furthermore, the process of bringing machine learning to the edge was examined in-depth, taking into account edge server-based architectures and joint computation, allowing researchers to consider both the absence (and its impact on privacy and local computational operations) and the presence (and its impact on cloud/edge server communications and remote data transmission power consumption) of data transmission to gateways or servers. The current stage of edge computing development foresees a plethora of solutions capable of meeting a wide range of requirements. It is feasible to define a collection of suitable hardware and software to implement AI-enabled IoT effective solutions based on the requirements (privacy, energy consumption, computational complexity). The study includes an example of edge machine learning implementation, demonstrating the usefulnessand simplicity of the proper edge platform for implementing machine learning.

IV REFERENCES

[1] Atzori, L.; Iera, A.; Morabito, G. The Internet of Things: A survey. Compute. Networks 2010, 54, 2787– 2805. [CrossRef].

[2] Vahid Dastjerdi, A.; Buyya, R. Fog Computing: Helping the Internet of Things Realize. IEEE Compute. Soc. 2016, 49, 112–116. [CrossRef].

[3] Wang, X.; Han, Y.; Wang, C.; Zhao, Q.; Chen, X.; Chen, M. In-edge AI: Intelligentizing mobile edge computing, caching and communication by federated learning. IEEE Netw. 2019, 33, 156–165. [CrossRef]

[4] Savaglio, C.; Ganzha, M.; Paprzycki, M.; Bădică, C.; Ivanovi´c, M.; Fortino, G. Agent-based Internet of Things: State-of-the-art and research challenges. Futur. Gener. Compute. Syst. 2020, 102, 1038–1053. [CrossRef].

[5] Edge Computing—Explore—Google Trends. Available online: https://trends.google.com/trends/explore?date=all&q=edgecomputing (accessed on 5 March 2020).

[6] Scopus Preview - Scopus - Welcome to Scopus. Available online: https://www.scopus.com/ (accessed on 15 March 2020).

[7] Guestrin, C. SVMs, Duality and the Kernel Trick. Mach. Learn. 2006, 10701, 15781.

[8] Lecun, Y.; Bengio, Y.; Hinton, G. Deep learning. Nature 2015, 521, 436–444. [CrossRef].

[9] Borgia, E. The Internet of Things vision: Key features, applications and open issues. Comput. Commun. 2014, 54, 1–31. [CrossRef].

[10] Arshad, B.; Ogie, R.; Barthelemy, J.; Pradhan, B.; Verstaevel, N.; Perez, P. Computer Vision and IoT-Based Sensors in Flood Monitoring and Mapping: A Systematic Review. Sensors 2019, 19, 5012. [CrossRef].