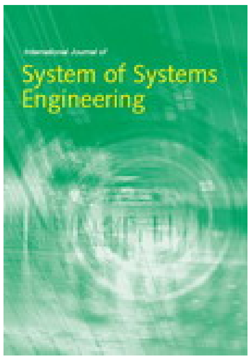


International Journal of System of Systems Engineering

ISSN online: 1748-068X - ISSN print: 1748-0671
https://www.inderscience.com/ijsse

Enhancing the performance assessment of network-based and machine learning for module availability estimation

Aqeel Luaibi Challoob, Abdullah Hasan Hussein

DOI: 10.1504/IJSSE.2024.10054530

Article History:

Received:	13 October 2022
Last revised:	28 November 2022
Accepted:	30 November 2022
Published online:	09 January 2024

Enhancing the performance assessment of network-based and machine learning for module availability estimation

Aqeel Luaibi Challoob*
and Abdullah Hasan Hussein

Department of Computer Techniques Engineering,
Imam Al-Kadhum College (IKC),
Baghdad, 10001, Iraq
Email: aqeelluaibi@alkadhum-col.edu.iq
Email: abdullahhussein87@alkadhum-col.edu.iq
*Corresponding author

Abstract: Interpreting network telemetry data is difficult. Size and volume are network assets. Production rises. ML predicts traffic trends to help decision-making. Classification and monitoring enable data science, sensor fusion, diagnostic devices, and vulnerability assessment. Complex domains have algorithms. Researchers have not found a fast, reliable way to categorise a dataset. Most literature evaluates classifiers' accuracy and falsification rate. Classification constraints include model development time, false positive rate (FPR), and precision. AI can estimate network complexity. New technology expands and complicates network messages. First, send facts. Only key nodes send messages in conventional opportunistic networks. Overusing key nodes reduces network life. We provide energy-efficient message-based routing. We assess message relevance and node energy during forwarding. It fixes energy-hungry nodes and prioritises vital signals. We replace the cache when it's full. This hinders mobility aid. This study employs machine learning to improve traditional mobility management. It presents a realistic technique using path-based forwarding architectures to identify network links. Instead of destination-based routing, delivery path information is transmitted and advanced using a mandatory access test.

Keywords: energy efficient; sensor network; machine learning; network security; data aggregation; module availability estimation; SwitchML; PyTorch; TensorFlow; network infrastructure.

Reference to this paper should be made as follows: Challoob, A.L. and Hussein, A.H. (2024) 'Enhancing the performance assessment of network-based and machine learning for module availability estimation', *Int. J. System of Systems Engineering*, Vol. 14, No. 1, pp.1–22.

Biographical notes: Aqeel Luaibi Challoob received a BSc degree in Software Engineering from the Imam Ja'afer Alsadeq University, Baghdad, Iraq, in 2011. He received his Master's Degree in Computer Science. Information Technology. Volodymyr Dahl East Ukrainian National University, Luhansk, Ukraine, in 2013 and the PhD degree from National Research Tomsk State University, Tomsk, Russia, in 2020. He is currently working with the Department of Computer Techniques Engineering at Imam Al-Kadhum College (IKC). His research interests include communication networking and machine learning.

Abdullah Hasan Hussein has completed PhD from the department of information technology management at Kazan Federal University in Russia. He has been awarded a Master's Degree in Computer Science and Information Technology from Malaysia. His research interests include ML, AI, communication networking and information technology.

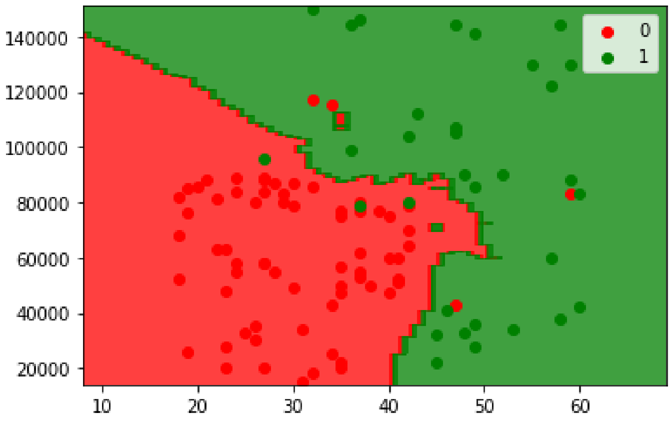
1 Introduction

The extraordinary success of machine learning (ML) resolution today results from the capability to construct increasingly complex models on progressively larger datasets. ML practitioners use distributed training to manage the resulting increase in training time (Cheon et al., 2017; Chilimbi et al., 2014). A workload that is increasingly network-bound is distributed training. It is still computationally demanding, to be clear. But because of GPUs (Jeon et al., 2019) and other hardware accelerators (Jia et al., 2018; Jiang et al., 2020; Jin et al., 2018; Keskar et al., 2016; Kim et al., 2019), computer performance has increased 62 times during the past seven years (Dang et al., 2015, 2016; Deng et al., 2021; Devlin et al., 2018; Harper and Konstan, 2015). This pace has proven difficult for cloud network deployments to match (Klenk et al., 2020; Lao et al., 2021) tilting the compute-to-communication ratio in favour of communication. Can a new kind of accelerator lessen the network bottleneck. We show that an in-network aggregation primitive, implemented using programmable switch hardware (Li et al., 2019; Liu et al., 2020), can speed up distributed ML workloads. By aggregating data during synchronisation periods, throughput is increased, latency is decreased, and training time is shortened (Abdolmaleky et al., 2017).

Developing an in-network consolidation primitive using configurable switches is fraught with various challenges. First, both on-chip memory and per-packet processing power are constrained. So that the switch can carry out its main duty of transmitting packets, we must restrict the number of resources we use. In contrast, a customisable switch's functional units work with decimal points (El-Gamal et al., 2020). The in-network aggregation primitive, unlike conventional unicast or multicast communication patterns, is an all-to-all primitive. As a result, methods for managing workers and locating and addressing packet loss are required for in-network grouping. SwitchML interfaces with distributed machine learning frameworks like PyTorch and TensorFlow to speed up communication and facilitate effective deep learning (Sharma et al., 2021). A. Only one toggle is the limiting factor in terms of scalability; however, we draw attention to the fact that programmable switches available on the market can enable up to 64 clusters at 100 Gbps or 256 at 25 Gbps (Tao et al., 2020). This rating is adequate to exceed the mathematical constraints of the technique due to the frequent triple Processor arrangement for each worker (Liu et al., 2016; Luo et al., 2018; Mai et al., 2014; Narayanan et al., 2020; Nemirovski et al., 2009).

A machine can analyse data and form conclusions thanks to machine learning (ML) (Figure 1). It entails using and expanding information acquired by practice, not only through consuming or memorising it. Machine learning (MLprimary) aims to discover and use disguised motifs in 'training' information. Learning trends are applied to data analysis to map unseen data to recognised groupings or classes. As a result, the coding methodology changes, and tasks are automated through the software (Elhadad et al., 2020). ML creates the model or software that complies with the data. Demand for ML has grown recently. Early machine learning techniques lacked flexibility and were rigid (Heidari et al., 2019).

Figure 1 Equipment to gather information from disaster victims to improve assistance (see online version for colours)



Models are trained with label data in deep classification, a type of machine learning. In supervised learning, models find a mapping function to map the input vector to the output variable (Naseri et al., 2018). Monitoring is needed during supervised education to create a model analogous to having a teacher present while a student is learning (Mendonça et al., 2021). Classification and regression are two different types of problems that can be taught (Metwaly et al., 2014). In reality, supervised learning could be used to assess risk, categorise photos, and find fraud and spam, among other things. Regression and classification are both subject to guided learning. Regression trees, non-linear regression, Bayesian linear regression, and polynomial regression are well-known examples of classification methods (Elkabbash et al., 2021).

In contrast, decision trees, logistic regression, and support vector machines exemplify classification. Another strategy that uses patterns deduced from unlabelled information is based on learning. Unsupervised learning aims to identify the incoming data's patterns and structures (Jamjoom et al., 2022). No supervision is necessary for unlabelled data. Instead, it goes out and looks for data models by itself. Clustering and Association are two issues that unattended learning may cause. As we have input data, however, no matching outcome data, supervised retraining cannot be immediately applied to extrapolation or classification issues. The goal is to locate a basic data collection and arrange the data in a manageable fashion based on commonalities (Bibi et al., 2022).

Famous algorithms include the A priori algorithm, neural networks, hierarchical clustering, and principal component analysis. Perceptron with several layers. One of the most used classifier functions, classifiers, has proven useful in various applications, including time series, classification, and regression issues. Implementation of the testing phase takes little time. The training phase, however, is generally carried out over a lengthy period. On the other hand, tiny weight values present the least effective qualities within a dataset, while big weight values present the most effective features inside a dataset (Singhal and Sharma, 2022).

One tree classifier that uses this method is the random tree classifier, although the number of trees must be determined before employing it. Each tree represents a single decision. Attributes from a dataset were randomly chosen for each unique tree. As a result, one could think of the random decision tree as a limited set of decision trees. Predicting the full dataset entails migrating the outputs of various decision trees and selecting the winning predicted class based on the overall number of votes. One of the decision tree techniques, the random forest classifier's primary objective, is to improve classifiers for trees based on the forest notion. The referred research created random forest classifiers that could be used to manage dataset noise values and had an acceptable accuracy rate. The classification stage does not involve any.

2 Network restrictions for machine learning development

These strategies are already adaptable to potentially different happenings, ranging from outstanding to dull, thanks to recent advancements in ML. With illustration, computer-aided diagnostics and medical imaging have substantially improved thanks to the application of machine learning (ML) in the healthcare industry. Usually, we commonly make use of technical methods based on ML. For instance, search engines mainly rely on machine learning (ML) for difficult tasks like web crawling, spell checking, query suggestions, and page rank. As more areas of our life become automated, including Smart things and driverless driving, it is evident that ML techniques will play a larger role in many systems that enable decision-making, analysis, and automation (Russakovsky et al., 2015).

A variety of additional variables, in addition to advancements in ML methodologies, have contributed to its comeback (El-Hasnony et al., 2021). The performance of methods depends primarily on information (Patarasuk and Yuan, 2009). Certainly, there is a massive volume of information on ethernet networks, which will only increase in the future thanks to networks such as Iot and its millions of mobile networks (Devlin et al., 2018).

This encourages machine learning (ML), which may be employed to discover hidden and unexpected patterns and learn about and comprehend the processes that produce the data (Zhu et al., 2021). In practice, the concepts mentioned earlier help in a limited way to get beyond the limitations of Internet architecture, but they do not provide a comprehensive answer to the problem. For instance, because they lead to network congestion, they have scalability problems. Poor routing often referred to as routing via the anchor point is the cause of this. Distributed mobility management attempts (Jia et al., 2018) aim to overcome the problems mentioned above and offer a better way to control mobile traffic by moving toward a flatter design using distributed anchoring (Zhou et al., 2016). Even when the anchor functionality is disseminated into the network edges, these solutions execute traffic tunnelling and anchoring in a localised manner, which does not reduce the traffic overhead required to allow mobility. In addition to the limitations outlined above, present systems either provide lossless handover with a potential delay or rapid switching with a probability of packet loss. Forcing loss-sensitive programs to accept considerable delays or, on the other hand, pressuring delay-sensitive applications to accept packet loss will reduce the quality of experience for users. This was demonstrated in (Dang et al., 2016), where experiments were carried out on user impressions of subjective video quality. Through these tests, the effects of packet loss on films with High-Efficiency Video Coding were quantified. It has been established that losses greater than 3% cause consumers to experience unfavourable effects.

Clean-slate strategies for revamping the Internet architecture have been used to address the location-dependent end-host communication architecture. As an illustration, Information-Centric Networking research programs (Keskar et al. (2016) aimed to answer the growing demand for an information-centric communication architecture that encourages information dissemination rather than end-host communication. This "clean-slate" approach to re-building the current Internet architecture has some implementation limitations related to protocol stacks and standardisations, stakeholders and vendors of the current internet, end-user equipment modification, and many other factors, despite its benefits for information dissemination (Nemirovski et al., 2009). Thus, recent research initiatives, such as those in Deng et al. (2021), have focused on disseminating ICN information over legacy IP networks and Software Defined Networking (Puthige et al., 2021).

It is critical to remember that learned machine learning models could be applied to inference on devices with lower processing capability, such as smartphones. Despite these advancements, managing and operating networks is still arduous, and network faults are common and typically brought on by human error (Li et al., 2019). Network problems endanger the financial stability and reputation of network providers. Thus, it is crucial to construct exceptionally durable autonomic networks (Harper and Konstan, 2015). ML faces a distinct set of difficulties notwithstanding the pressing requirement for cognitive functioning in networking operation and administration (Klenk et al., 2020). The first is that no rules can be consistently followed to achieve network homogeneity because every network is different. In contrast, one corporation's social network differs greatly from another's. Because of this, it is possible that patterns that have previously been shown to operate in one kind of network won't work in another (Ashraf et al., 2021).

The network dynamics prevent the adoption of permanent patterns that would help with management and operation because the network is constantly changing. Given the variety of devices linked to the net and the steadily growing number of operating programmers, it is almost impossible to manage network connections manually (Adnan et al., 2021).

3 An overview of machine learning for networking

Fundamentally, during training, compute-intensive phases and model update synchronisation alternate. Workers generate a lot of bandwidth to communicate model updates, whether using an everyone or parameter server, which makes training stall till it's complete. Research has found that interaction is becoming an effective hurdle in distance training (Kim et al., 2019). This alteration is the result of two factors. Advancements cause the first in GPUs and other computer accelerators. For instance, compared to the V100, released just 2.5 years earlier, the NVIDIA A100, which was just unveiled, offers 10x and 20x performance enhancements for blended and hanging algorithms, accordingly (Mai et al., 2014). This rate far outstrips network bandwidth advances; for instance, it took eight years to standardise a 10x increase in Ethernet speeds (Abulkasim and Alotaibi, 2019).

Second, the burden has changed regarding how much communication is done vs. how much processing is done. When does the network start getting crowded? To offer a quantitative response, we characterise the training of 8 typical DNNs on an 8-worker cluster utilising. We monitor events at the network level, allowing us to report on the amount of time spent communicating and how much of that time may overlap with computing (Table 1). Except for three jobs at 10 Gbps, all workloads are at this rate. Remember that four means that using the most modern GPUs with a 100 Gbps network will result in a severe problem with network performance (Table 1). In clustering problems, similar data are grouped while the differences between the groups are widened.

Table 1 Analysis of the collected tomography metrics' prediction abilities when used separately and when integrated with modelling weighted learning algorithm

Prototype	*Assignment*	*Vulnerability*
Parameter 2	0.519	62.2%
Parameter 3	2187	82.4%
Parameter 4	98.151	71.2%
Parameter 5	24.14	72.5%
Parameter 6	2562	96.3%

In contrast, a segmentation or projection task aims to turn a set of novel inputs into a succession of distinct or continuous-valued returns. When statistical correlations in the data are wanted, the issues with rule collecting are separate. The use of ML methods has been shown in various issue domains. The study of data from sizable databases is the focus of the closely related field of data mining (Li et al., 2019). While ML approaches may be useful, data mining issues aim to critically and actively study data – its properties, constants, invariants, time granularity, probabilities, and their transformations. However, ML extends beyond huge data to predict impending events or a sequence of events. In general, ML performs flawlessly when finding answers to queries with significant that out. As a result, as shown in Figure 2:

i specific techniques are developed to find and utilise hidden patterns for problems like I explaining the output as a clustering of data for document clustering

ii the result of coming occurrences for supervised learning

iii assessing the outcome of a series of points for numerical optimisation problems (Jiang et al., 2020).

The issue applies to some co-datasets and outcome functions, even if the illustration shows the data and results in a two-dimensional plane. For instance, clustering may create a non-linear function in a support vector that can distinguish between different datasets. One of these issues that can benefit from using ML is networking difficulties. Depending on the condition of the network, a networking classification issue, such as Refusal (DoS), Login (U2R), Core (R2L), or probing, may be used to anticipate the kind of security assault that will occur. On the other side, a regression problem can be created to forecast when a loss will happen. ML benefits many problem types, but there is a universal method for developing ML-based remedies. The basic elements required to create ML-based solutions are shown in Figure 3. The process of acquiring, creating, or identifying the set of data and the set of classes of interest is known as data gathering. Data density is decreased by applying topic modelling to find distinctive features that improve reliability while utilising less processing capability. Not the worst of everything, ML techniques meticulously examine the intricate internal and external links in data to find a theory for the result.

Figure 2 Perspective of the network infrastructure conceptually (see online version for colours)

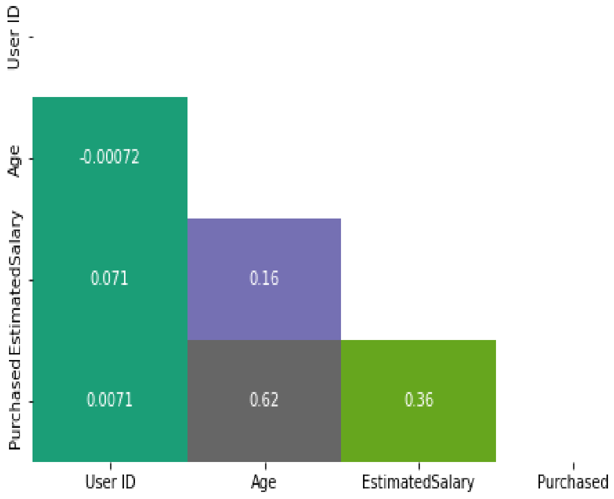
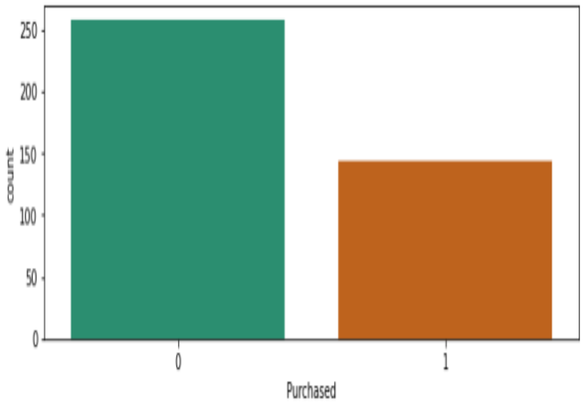


Figure 3 Representation of the proposed system conceptually (see online version for colours)



Nowadays, it is getting harder and harder to tell factual information in machine learning. Before choosing which platform to employ, you must assess the issues you hope to resolve. The tasks carried out manually daily and have a fixed output are the simplest to automate. Before automation, complicated procedures needed more examination. While computer science can undoubtedly aid in automating some processes, not all robotics issues require it. The absence of high-quality data is the main issue facing machine learning. While improving algorithms frequently takes up most developers' time in AI, the data must be high quality for the methods to work as intended. Ideal Machine Learning's archenemies are filthy, noisy, and incomplete data. This problem can be resolved by carefully evaluating and scoping data via information management, integration, and pattern discovery until you obtain clear data. Before you begin, you should complete this. The ability to process enormous amounts of data is necessary for machine learning. Legacy systems often cannot keep up with the strain and break down. Check your system's machine learning capabilities. If possible, provide flexible storage and hardware acceleration. An organisation often has analytics engines in place when upgrading to deep learning. It's difficult to combine new and old Machine Learning approaches. Accurate interpretation and documentation ease implementation. Working with an implementation partner simplifies anomaly identification, predictive analysis, and ensemble modelling. Machine learning and deep analytics are new. Insufficient qualified individuals oversee and develop machine learning analytical content. Data scientists need specialised knowledge and a solid grasp of science, technology, and arithmetic. These workers are in high demand and know their value, so pay them well. Many managed services providers have a list of qualified data scientists and can help with hiring.

4 Parameter estimation

For ML techniques to create an ML model for a specific networking challenge, meaningful data, possibly bias-free, are required. Since typical datasets differ not only from one challenge to another but also from each historical era to the next, data collecting is crucial. Both traditional and online methods can be used for data collection (Kim et al., 2019). It is feasible to obtain historical data for model testing and training offline. But real-time network data gathered during the embedding process can be used to retrain a model and provide feedback. If offline data is pertinent to the addressed networking issue, it can also be acquired from various repositories. Waterloo Netflow Storage (Deng et al., 2021) and the Cornell Information Retrieval in Desks archive are a couple of instances of similar repositories (Dang et al., 2015), Measurement and Analysis on the WIDE Internet (MAWI)Working Group Traffic Archive (Harper and Konstan, 2015), and Information Marketplace for Policy and Analysis of Cyber-risk & Trust (IMPACT) Archive (Chilimbi et al., 2014). Using monitoring and measurement technology can effectively obtain data from both online and offline sources. Concerning factors like data sample rate, monitoring duration, and location, these technologies allow you greater options for data collection.

They frequently employ various network management protocols, including (Chilimbi et al., 2014), Cisco Total (Jiang, et al., 2020), and IP Flow Info Export (Jin et al., 2018). But monitoring comes in two flavours: active and passive (Lao et al., 2021). Active monitoring puts traffic measurements into the network, similar to probe packets, and gathers pertinent data. Passive monitoring, on the other hand, gathers information by monitoring actual internet usage. Injected traffic uses more capacity; hence it is obvious that active monitoring has a higher overhead.

On the other hand, passive monitoring does away with this cost at the expense of additional technologies that comb through network traffic, looking for pertinent data. Data is divided into training, validity and test datasets after being acquired. The training data defines the optimum ML model parameters, such as the weights of connections between neurons in a neural network (NN). On the other hand, the validation set is used to select a theory from a pool of ML models or to determine the proper topology of an ML model.

A testing set is not required if an ML theory and its infrastructure have already been chosen. The chosen model's impartial performance is then assessed using the test set. Consider using hold or k-fold inter as your testing and validation strategy. A portion of the raw dataset is set aside and utilised as a verification set in the holdout approach. In contrast, the dataset is freely divided into k equal sections with the k-fold cross-validation. The training phase uses k distinct subsets, while the confirmation (or testing) method, which is repeated k times, uses the remaining subset. All iterations are averaged to determine the results. Over all-reduce and PS, in-network aggregation provides (Luo et al., 2018) a basic advantage. At its most extreme, the PS technique involves twice so many terminals and network bandwidth to meet this 2U byte sending rate, but at the cost of more resources. In-network aggregation provides 'sub-RTT' lag (Li et al., 2019), which even the competing approaches cannot provide, and does not involve end-host computing to complete aggregate, independent of resource costs, illustrating the benefits from accumulation.

The server's approach to providing customer service depends on the queueing discipline. It reveals if clients are served concurrently or one after the other. It also specifies the sequence in which they will be served, if sequentially. If applicable, it also explains how consumers are divided across the user's capacity. Nodes in a queue may adhere to one of the three service disciplines. A customary queueing system where the person who has waited for the longest gets serviced first. In the alternative method, the consumer who has waited for the longest gets served last. Each consumer receives an equal share of the service capacity. Priority customers receive first-class treatment. The task with the lowest number is the one that will be completed next. The task that requires the least amount of computing is the next one to be completed.

Using current or past data incorporated into metrics predictive models, predictive analytics entails making predictions about unknowable future events. One or more algorithms make up these predictive models. Medical, sales, insurance, transportation, retail, and other industries employ predictive analytics. Every forecast is based on various variables, such as every patient-specific trait. My web server will enter user-inputted waiting lists and anticipated wait times made by a queueing model into its predictive model. This data is examined by the predictive model, which searches for trends to arrive at an accurate prediction. The model will therefore include a regression-based approach.

A trail of network-level events recorded during training is used to assess the effect of communication performance. This trace allows us to capture actual compute durations and virtual memory delays, including latency for boundary events that precede each synchronisation, while simulating various network speeds and calculation patterns. Our trace, in particular, accurately captures the timing of every all-reduce invocation and hence faithfully accounts for possible connection and compute overlap. We contrast the effectiveness of ring all-reduce, the current best practice, and in-network aggregated. The CPU time in batch systems over ringed all-reduce performance is summarised in Table 2. For information exchange models, respectively. The fact that this research delivers a RAR code that is theoretically flawless should be emphasised. The measured development of our real INA solution is higher than those of true Rr solutions since it is challenging to use all bandwidth and remove system overhead expenses in real RAR solutions.

Table 2 Insist on precision in our long-term planning

Prototype	*Not squeezed*	*Rankwise–10%*	*Rankwise–1%*
Parameter 2	0.7213	1.7421	1.4232
Parameter 3	32.62	86.24	82.54
Parameter 4	1.7435	1.7143	84.15
Parameter 5	74.71	71.15	71.53
Parameter 6	67.14	61.57	37.14

By employing codecs, the data volume of system revisions can be decreased. As an additional way to reduce communication expenses. The suggested techniques (Cheon et al., 2017; Kim et al., 2019) include broadcasting only a portion of the elements or quantising gradient elements to decrease their bit-width.

Machine learning's key benefit is its ability to identify patterns in various data. Machine learning programs can analyse vast amounts of heterogeneous data and produce precise predictions. For instance, ML can assist in predicting customer service faults and offer a virtual assistant to direct the scheduler to those failures. Businesses that rely on manual processes and basic management software won't be able to keep up with their more advanced rivals. The fact remains that businessmen are aware of the latest technology and appreciate the immense value they can add to a boom. However, they encounter several difficulties as they use Big Data, Statistics, and Machine Learning to execute them. In addition, the business faces significant issues in managing the negative effects of technology on labour and the availability of experienced workers while maximising the benefits of next-generation applications for product advancement and new product/service offerings. According to Keskar et al. (2016), the management field has increased from Classical management theory to the present. Although the internet has not significantly altered its basic principle, it has become increasingly crucial for company leaders to comprehend the influence of new generations' advances in Machine Learning. For instance, top management can observe significant employee opposition to the changes brought about by new technologies. To create a sense of belonging to ML in such an environment, top managers must put organisational change tactics into practice.

4.1 Toggle framework coupled

To get over the constraints of the constrained computational capacity at switches, SwitchML intelligently divides computation amongst end-hosts and switches. The end hosts are in charge of dependability management and carrying out more complicated computations, while the switch handles integer aggregation.

4.2 Multicast accumulation relying on pools

A switch cannot aggregate full vectors at once since a complete model update would need more data than it could hold. Instead, SwitchML feeds aggregating it through the switch, performing it all at once on a few vector components. A reservoir having a numeric cell state, an abstraction, allows for this. Because end hosts manage analysers in a pool by choosing how they will be deployed, reused, or need more complex failure management, SwitchML has a simple design for something like the switches control plane (Algorithm 1).

Algorithm 1 Modify reasoning

```
1: Establish a new state:
2:     k = frequency of employees
3:     span[m], enumerate[m] := {0}
4: because once getting r(idy, out, variable)
5:     span[r. jdy] span[r. jdy] + r. variable      {+ is the variable sum}
6:     enumerate[r.jdy]++
7:     if enumerate [r.jdy] = k later
8:         r. variable span[r.jdy]
9:         span[p.idx] 0; enumerate[r.jdy]        0
10:        cluster r
11:    also
12:        collapse r
```

4.3 *Resilient to faults procedures*

In order to recover the bit errors with the least amount of overhead, we create efficient algorithms. To address worker or network problems, we use conventional techniques.

4.4 *Leveraging randomised integers to aggregate*

The processing capability of modern switches cannot compete with floating-point operations. Instead, we use a block floating-point-like technique (Liu et al., 2016) to efficiently transform floating-point information to 32-bit integers at end sessions' significantly impacting accuracy percentage.

We will now go over each of these parts individually. We describe a variant of the method wherein the packet losses do not happen to simplify the display. Later, we lift this restriction.

5 Algorithm for flip grouping

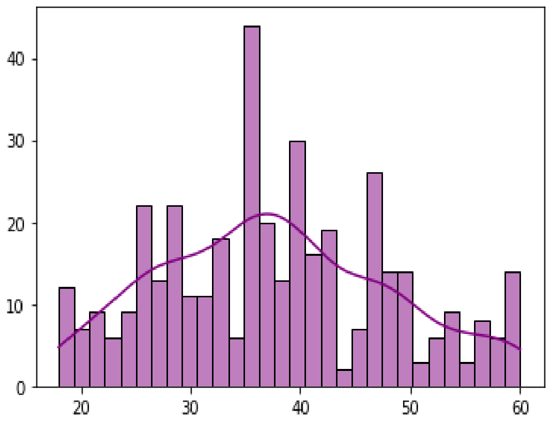
In-switch integer aggregation, the fundamental network primitive offered by SwitchML, is described first. A SwitchML switch offers a pool of s addressable by index integer aggregators. Each pool slot collects a vector of k values, all given simultaneously in a single update packet. The add operator, which is circular and associative and somehow does not rely on the incoming traffic sequence, serves as the aggregate function. Because model changes must be averaged, inclusion is a less ideal method of gathering than average. Identical to the all-reduce technique, the exchange cannot perform the last splits step efficiently; therefore, we delegate it to the router.

Some of the characteristics in the datasets possess extremely erratic intensities, bounds, and measurements. Some algorithms, like the Perceptron, make incorrect predictions due to the dataset's extreme variability. Additionally, a lot of computational resources are needed. Thus, one often employed technique for normalising the spectrum of independent parameters is feature scaling. Some machine learning methods that determine the separation of data must scale features.

When packet loss could result in the Transmission Management Protocol's response, which uses congestion management techniques, substantially reducing the data flow, lossless handover is typically employed for delay-tolerant applications, such as file downloads. If the source runs in lossless handover mode, the handover tunnel establishment follows the same guidelines described above for seamless handover. However, the user plane downlink packets that the source has processed but is still buffering locally due to the UE not having delivered and acknowledged them yet will likewise be transmitted across X2.

The extension header field of these packets contains the requested sequence numbers and is used to transport them all together. They traverse X2 before the packets from the source S1 path arrive, as shown in Figure 4 (Liu et al., 2020). Since crisp handover also requires in-sequence packet delivery, the end of forwarding is handled in the same way as a smooth switchover. Additionally, the target must guarantee that each packet reaches the target side in the proper order. The aggregate primitive's behaviour is demonstrated in Algorithm 1. A packet p contains a vector of k aggregated integers and bears a spam index that identifies which certain extractor should be employed. The slot is instantly made available for usage after resetting the aggregate value and counter.

Figure 4 Fresh revised technique for successful routing (see online version for colours)



The pool-based design is optimised and deals with two significant drawbacks of programmable switch systems. First, since switches only have a limited amount of memory, it is unnecessary to keep a whole model updated at once; instead, bits of the model are aggregated in a streaming form. It is improbable that this will extend significantly in the future due to limits in the design (Jin et al., 2018; Liu et al., 2016). k can be 64 or 256 in our deployment.

5.1 *Method for employee work accumulation*

This system's main benefit is that it obtains worker unanimity on which slots to use for defined criteria without requiring any explicit collaboration among them. Since the translation among modelling changes, slots, and packets is consistent, cooperation is implicit. Additionally, reordering does not affect the scheme since each packet contains the span index and offsets. To identify damage and toss out corrupted packets, employ a straightforward checksum. Just after the first s packets, this frame relay automatically clocks. This is because a slot cannot be utilised again until each worker sends input for the slot's attribute update. When a space is used up, the switch notifies the workers by sending them packets (Algorithm 2).

Algorithm 2 Employee reasoning

```
1: for i in 0: s do
2:    r.jdy   j
3:    l>.°lf  k •i
4:    r.variable Q[r.out : r.out + k]
5:    reply p
6: duplicate
7:    acquire p(r.jdy, out, variable)
8:    A[r.out : r.out +k] r. variable
9:    r.out r.out + k • s
10:   if r.out < size(U) later
11:       r.variable Q[r.out : r.out + k]
12:       send r
13: after A is deficient.
```

5.2 *Managing nodes*

Up until now, we have presumed that transmissions are never lost. Network instability or overload could lead to packet loss. The system would crash with the prior technique if just one packet were lost. The switch cannot complete the necessary parameter aggregations if there is a misplaced shipment on the 'upward' channel of the transition to the employees. This has an additional benefit in addition to eliminating the requirement for multiple shadow copies: the switch no longer needs to keep track of entire numerals. Each worker's timeout discovers packet loss. When this occurs, the employee is left in the dark regarding whether the shift got its previous packet. It communicates the same slot IDX and veers as the original update in either scenario. This element will always contain the status for the precise aggregate in the transfer. The exposed bitmask indicates whether an adjustment has already been made to the slot. The switch considers the most recent packet as a resend statement and responds with a connectionless signal with the outcome if the grouping has already finished for a position. However, the converter still gets an update signalling for the slot. The pooled result of a single slot is only rewritten for reuse when it is confirmed that each staff has obtained it. Slot reuse happens when all of the workers have indicated their intention to go forward by transmitting their updates to the identical slot in the other group. It should be noted that this approach handles the finish of aggregated for even a space idy in one pooled carefully and distinctly validates that each worker has finally received the results from previous aggregating for that column idy in the shadowed copy.

5.3 *The use of bobbing statistics*

Floating-point numbers are frequently used in DNN training, although they are currently not natively supported by programmable switches. We looked at two methods for closing this gap. By definition, floating-point numbers are estimates (Algorithm 3). Over real numbers, SGD and comparable algorithms are defined. In order to create a numerical representation that may be universally used for applications with a wide range of features, floating-point numbers trade off the range, precision, and computational complexity. There are a lot of other approximations, though. An approximation can achieve tolerable accuracy for a given application with less overhead than a traditional floating point. Several illustrations utilise the DNN application's features to lower communication and computation costs. For instance, the mixed-precision (16-/32-bit) TPUs included in NVIDIA Volta and Ampere GPUs (Dang et al., 2015; Harper and Konstan, 2015; Liu et al., 2020) enable training with accuracy comparable to full-precision methods. Other research on gradient exchange for SGD has reduced the number of bits and gradient elements transmitted via fixed-point quantisation, dithering, or scarification (Lao et al., 2021; Kim et al., 2019; Luo et al., 2018).

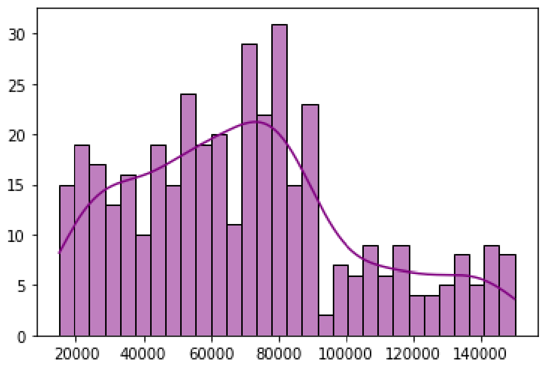
Algorithm 3 Switches logic that recovers from bit errors

```
1: Create a new state:
2:     k = frequency of employees
3:     span[2, m], enumerate[2, m], seen[2, m, k] := {0}
4: because once acquiring r(mjq, ver, jdy, out, variable)
5:     if seen[r.ver r.jdy, r.mjq] = 0 later
6:     seen[r.ver r.jdy, r.mjq]          1
7:     seen[(r.ver+1)%2, r.jdy, r.mjq]          0
8:     enumerate[r.ver p.idx]          (enumerate[r.ver r.jdy]+1)%n
9:     if enumerate[r.ver r.jdy] = 1 later
10:        span[p.ver r.jdy] r.variable
11:    else
12:        span[r.ver r.jdy] span[r.ver r.jdy] + r.variable
13:    if enumerate[r.ver r.jdy] = 0 later
14:        p.vector span[r.ver r.jdy]
15:        cluster r
16:    also
17:        collapse r
18:    also
19:    if enumerate[r.ver r.jdy] = 0 later
20:        p.vector span[r.ver r.jdy]
21:        forward r to r.mjq
22:    also
23:        collapse r
```

Additionally, some researchers have looked into block floating-point representations (Lao et al., 2021; Kim et al., 2019), which share an exponent among some tensor elements to minimise the computation needed to conduct, inspired by block floating-point. Only when aggregating gradients is this representation employed; all other data (weights, activations) continue to be represented in a 32-bit floating-point. Before transferring the matching block of gradients, workers must decide on a universal value of m in order to implement this quantisation of floating-point values. We develop a straightforward look-ahead method: Workers include their local block's maximum gradient (local block $j + 1$) while sending the jth block to slot i. (rounded up to a power of 2).

Through studies, we show that this information quantisation makes it possible for training to reach an equivalent degree of accuracy in an equivalent number of repetitions as a shred of evidence exists to network, and we demonstrated the convergence behaviour. The test accuracy over time is shown in Figure 5. Switch ML's accuracy (about 91–93% in the final five points) is comparable to what TensorFlow training on the same worker arrangement yields, and it fits earlier findings (Aoudni et al., 2022). We also looked into developing a limited kind of 16-bit floating-point, even though the representation mentioned earlier is used throughout the rest of the work. In this variation, the switch converts each incoming packet's 16-bit floating-point value to a fixed-point value before performing aggregation. We could only implement half of the 16-bit floating-point format's dynamic range due to switch resource limits; we anticipate this will cause poor training convergence. On the other hand, our 32-bit integer format offers a low overhead on workers, a strong dynamic range, and little switch resource usage. An advantage for bandwidth would come from a 16-bit format.

Figure 5 The creation and implementation of machine learning models (see online version for colours)



6 Deployment

We develop SwitchML as a group effort, integrating it into TersorFlow via Horovod and PyTorch's Distributed Data Parallel module (Adil et al., 2022). The worker was constructed using Intel DPDK. Our P4 application contains flow control, retransmission, exponent-calculation logic, and aggregation distribution over several ingress pipeline stages. It takes advantage of the traffic manager subsystem to send numerous copies of the result packets. One switch pipeline can handle 64 elements for each packet, while all four switch pipelines can process 256-element (1024-byte) packets.

Recirculation, or sending packets through switch pipelines more than once, is how we accommodate bigger packets. One pipeline is used in our 64-element architecture. Intrinsic looping gates on the microchip offer sufficient bandwidth. We recycle packets through the four switch pipelines to provide 256 elements. For additional recirculation bandwidth, it is necessary to put switch ports into loopback mode, freeing up 16×100 Gbps for use by employees. We recirculate through all the pipelines once a slot is finished to read the findings. The delay grows deterministically with the number of pipelines runs throughput because of the high packet processing cost. By integrating a portion of RDMA within the switch, we address this. As a result of the RDMA NIC's ability to split huge messages into separate packets, employees can offload packet processing. This mode does not use any link-level flow management techniques provided by RoCE. However, it supports several co-conversations and recognises transmission drops without repetition. SwitchML keeps using its current dependability mechanism.

Timeouts compel a client to resend the complete multi-packet communication, while duplicate packets are handled as before. We employ short, multi-packet messages to compromise the cost of retransmission and the advantage of offloading. Forwarding costs additional money, but the common situation is substantially faster, even though we only use one CPU core. RDMA Author Data may transfer immediately between the switch and GPUs thanks to immediate messaging, with client CPUs handling protocol activities. RDMA headers contain SwitchML metadata. To enable packet interleaving, concurrent messages are sent on different queue pairs; during task setup.

7 Results

We provided a rack context for SwitchML's definition. Large-scale ML applications, however, might require more than one rack. SwitchML's framework can hierarchically support numerous racks, creating separate replicas of their switch logic. Mathematica and Microsoft Excel are used for getting results. Each employee has a connection to a switch at the rack's summit, which records information about the workforce. The resulting packet is routed to a tier-1 aggregate switch because this switch gathers updates from several racks rather than disseminating the results to the workers. This can support as many tiers as necessary to provide the appropriate network topology. A root switch finishes the final step in aggregating partial aggregates, and then its activities in the fields an outcome packet downward. The switching distributes the message at each level until something reaches the employees. The governing equations are as follows:

$$T(Y_1,Y_2;k_1,k_2)=\int_{k_i}^{\infty}\varnothing(Y_i \mid h_j;k_1,\mathrm{k})F_j(h_j;k_1,k_2)dh_j \tag{1}$$

$$=\int_{t_i}^{\infty}L[F_{i|j}(h_i \mid h_j):F_i(h_i);k_1,k_2]F_j(h_j;k_1,k_2)\mathrm{d}h_j \tag{2}$$

Additionally, we can provide switches with various processing pipelines thanks to the hierarchical design. In the case of the switches we employ, $p = 16$.

Significantly, the packet loss recovery algorithm described in Section 4.5 functions in the multi-rack case by design. The utilisation of raster graphics and shadowing copies ensures that the change negatively impacted by the interferences is always eventually reached by causing the agglomeration package to be replayed in the channel of the actual network switch. Following equations (1) and (2), we get the network that uses machine learning to express shared information among two nodes:

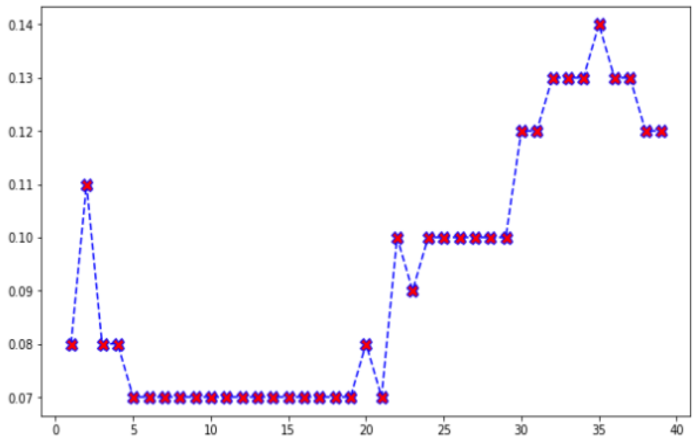
$$T(Y_1,Y_2;k_1,k_2)=K[F_j(h_i;k_1,k_2):F_1(h_1;k_1,k_2)F_2(h_2;k_1,k_2)] \tag{3}$$

$$=R(Y_1,k_1,k_2)+R(Y_2;k_1,k_2)-R(Y_1,Y_2;k_1,k_2) \tag{4}$$

$$=R(Y_1,k_1,k_2)-R(Y_i \mid Y_j;k_1,k_2), i \neq j \tag{5}$$

This is adequate for specialised networks. A congestion control strategy might be required for more widespread use; concurrent effort has been made to design such protocols (Farouk et al., 2020). We consider the following strategy when assessing an algorithm's learning execution time. As previously explained, we employ the walk-forward method, which requires us to run each algorithm multiple times on datasets of varying sizes. The undergoing and finishing times of the execution are measured, and the duration is taken into account to determine the execution time. We make use of the available 'time' function to accomplish this. The average of all an algorithm's execution periods for testing and training determines how long an algorithm will run (Figure 6).

Figure 6 Outline showing how changing the spacing between the end devices has an impact (see online version for colours)



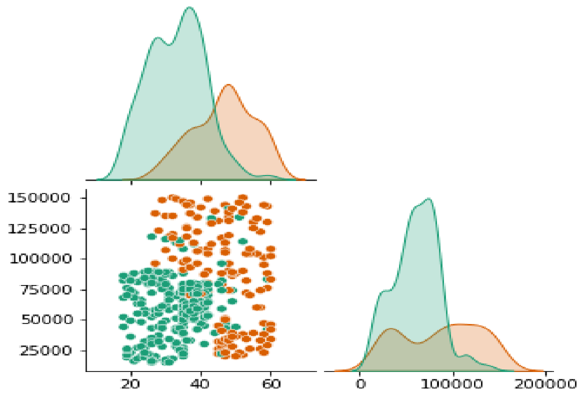
Up to this point, we have described SwitchML as an in-network computing strategy emphasising the mechanisms that provide effective model update aggregation @ line rate on memory-constrained controlled switching circuits. The deployment strategy may be effective in some situations, but we point forth that there may be other uses for our solution. The Switchfly switcher part would work per the chosen net attachment. Then, a data aggregator like this might be installed in racks. It might be directly connected to the previous ToR utilising some 100 Gbps or 400 Gbps ports, or it might be connected to it through a separate network connecting the employee servers to it within the tower. We predict that doing so will lead to equally improved efficiency and more options for distributing setup; a contemporaneous study has been looking into a similar technique on top of a Gigabyte (Abulkasim et al., 2019).

How to handle concurrent reductions using SwitchML in multi-job or multi-tenant settings is a concern (Figure 7). The answer is conceptually straightforward. To ensure accuracy, each project requires its pool of aggregators. As previously said, less than 10% of the switch's capabilities were used for one decrease. Additionally, contemporary switch chips have numerous independent pipelines, each with its resources. So, an admission mechanism would be required to regulate the assignment of tasks to pools.

8 Discussions

The same binary classifier attribute of the sought new was kept in the adequate number of possible samples that the authors could pull from some of these datasets. The suggested system uses extra energy-saving methods and functionally coordinated clustering techniques. This paper examines the benefits and drawbacks of various clustering techniques. Victims can be located, emergencies can be ranked, and flooding flow can be calculated thanks to the WSN nodes' critical sensors and sensors. Consequently, weights have been assigned to these classifiers for various datasets using multi-criteria decision-making. Utilising those scores, the classifiers' rank was also determined. To start, the best classification group has been determined by an intraorganisational study. Second, the best learners for the detection datasets have been identified using an intergroup examination of the best classification group. The authors examined thirteen performance metrics. As a result, the most accurate classifier has been chosen. In contrast to the intraorganisational study, the interethnic analysis ranked the clustering algorithm classifier class as the top classifier group. Guideline classifiers were ranked second.

Figure 7 One decrease uses slightly fewer options than a small percentage of the platform's capability (see online version for colours)



9 Conclusions

Over the past 20 years, machine learning has been employed successfully in numerous networking fields. With a focus on vehicular networks, optimisation, and information assurance, this review offers a thorough body of knowledge on the use of ML approaches in service of system operation and management. After analysing typical literary works, we explore and analyse the applicability of the recommended ML methods in tackling challenges related to the autonomous administration and operation of wireless optical. Professional growth and development will unquestionably need to handle a significant increase in traffic volumes and linked equipment to provide cultural assets for access to information and exchange. It will be more difficult to complete traffic engineering duties, including collision avoidance, traffic prediction, categorisation, and routing, because of the enormous size and quantity of unpredictability. Additionally, this will increase the likelihood of errors and security breaches. The scale of ML-based approaches must be considered in light of the anticipated data volume, instrument count, and like, even though they have shown promise in addressing a variety of technical traffic hurdles. However, most ML-based fault and solid approaches currently in use focus on single-tenant networks. Given the failure and the company is performing the framework required for future networking, it is necessary to improve or alter existing ML algorithms to reflect the idea of poly tenancy in multi-layer infrastructure. In this poll, we look at the issues mentioned earlier and a few other challenges and opportunities. Our findings indicate that additional study is necessary to advance the condition and ultimately attain the major objective of sympathetic connection.

SwitchML reduces communication overheads at a single-rack scale, accelerating DNN training. To effectively synchronise model updates among distributed workers running in parallel at each training iteration, SwitchML leverages in-network aggregation. Since Wireless Sensor Network nodes require less energy than other networks, energy-efficient approaches are frequently used in all networks. The capacity of the nodes' batteries is limited, and how energetic they are influencing how long the network will last. This survey article provides a comprehensive overview of the various energy-effective techniques for extending the life of WSNs. The proposed system employs additional energy-efficient algorithms and operationally centralised clustering approaches. The advantages and disadvantages of various clustering methods are examined in this study. The WSN nodes are equipped with crucial sensors and transducers that help locate victims, rank emergencies, and calculate flood flow. Additionally, they transmit the necessary information to the bs to improve flood relief activities. There has been developed a distributed technique for joint power and rate optimisation. When designing a protocol for wireless sensor networks, two important and competing requirements are network performance and energy usage. In terms of throughput, the suggested approach performs better than conventional sensor message transmission approaches. It was confirmed in the NS3 simulation due to the model's enhanced bandwidth at the same power consumption as the DSDV protocol. The larger throughput makes the connection with both victims possible, increasing the rescue effort's effectiveness and speed.

9.1 Future scope

Transfer learning categorisation and running time analysis was not performed in this work; as a result, future work could implement a transfer learning classification framework that considers time complexity analysis. This study offered a thorough examination with several useful conclusions for designers. The key contribution of this article is the rating of thirty – nine learners on detection datasets after comparison using thirteen confirmed in this study. The current study does, however, have several shortcomings. Additional research is necessary for light of additional datasets and different application domains. To comprehend the specifics of the classifiers, it is also necessary to carry out class-wise function observation, observe the size of the classes, and evaluate the performance of the classifiers using various sample sizes. The classifications' durability and adaptability were not examined. Many additional data can be employed in future studies to evaluate the accuracy of the classifier. Many modern ranking algorithms can be employed as the voting basis to determine the precise ranks of learners. This article discussed some alternative rule-based, decision forest classifiers that may be examined to understand the true performance of a classifier and classifier groups. An ideal classifier can be utilised in conjunction with an appropriate feature selection method to create a reliable detection mechanism.

References

Abdolmaleky, M., Naseri, M., Batle, J., Farouk, A. and Gong, L.H. (2017) 'Red-green-blue multi-channel quantum representation of digital images', *Optik*, Vol. 128, pp.121–132.

Abulkasim, H. and Alotaibi, A. (2019) 'Improvement on 'multiparty quantum key agreement with four-qubit symmetric W state'', *International Journal of Theoretical Physics*, Vol. 58, No. 12, pp.4235–4240.

Abulkasim, H., Alsuqaih, H.N., Hamdan, W.F., Hamad, S., Farouk, A., Mashatan, A. and Ghose, S. (2019) 'Improved dynamic multi-party quantum private comparison for next-generation mobile network', *IEEE Access*, Vol. 7, pp.17917–17926.

Adil, M., Song, H., Ali, J., Jan, M.A., Attique, M., Abbas, S. and Farouk, A. (2022) 'Enhanced-AODV: A robust three phase priority-based traffic load balancing scheme for internet of things', *IEEE Internet of Things Journal*, Vol. 9, No. 16, pp.14426–14437, doi:10.1109/jiot.2021.3072984.

Adnan, R.M., Mostafa, R.R., Islam, A.R.M.T., Gorgij, A.D., Kuriqi, A. and Kisi, O. (2021) 'Improving drought modeling using hybrid random vector functional link methods', *Water*, Vol. 13, No. 23, p.3379.

Aoudni, Y., Donald, C., Farouk, A., Sahay, K.B., Babu, D.V., Tripathi, V. and Dhabliya, D. (2022) 'Cloud security based attack detection using transductive learning integrated with hidden Markov model', *Pattern Recognition Letters*, Vol. 157, pp.16–26.

Ashraf, N.M., Mostafa, R.R., Sakr, R.H. and Rashad, M.Z. (2021) 'Optimizing hyperparameters of deep reinforcement learning for autonomous driving based on whale optimization algorithm', *PloS One*, Vol. 16, No. 6, p.e0252754.

Bibi, A., Attique Khan, M., Younus Javed, M., Tariq, U., Kang, B-G., Nam, Y., Mostafa, R.R. and Sakr, R.H. (2022) 'Skin lesion segmentation and classification using conventional and deep learning-based framework', *Computers, Materials and Continua*, Vol. 71, No. 2, pp.2477–2495.

Cheon, J. H., Kim, A., Kim, M. and Song, Y. (2017) 'Homomorphic encryption for arithmetic of approximate numbers', *Advances in Cryptology – ASIACRYPT 2017*, Springer International Publishing, Cham, pp.409–437.

Chilimbi, T., Suzue, Y., Johnson, A. and Kalyanaraman, K. (2014) *Project Adam: Building an Efficient and Scalable Deep Learning Training System*, OSDI.

Dang, H.T., Sciascia, D., Canini, M., Pedone, F. and Soule, R. (2015) *NetPaxos: Consensus at Network Speed*, SOSR.

Dang, H. T., Canini, M., Pedone, F. and Soulé, R. (2016) 'Paxos made switch-y', *SIGCOMM Comput. Commun. Rev.*, Vol. 46, No. 2.

Deng, W., Pan, J., Zhou, T., Kong, D., Flores, A. and Lin, G. (2021) 'DeepLight: deep lightweight feature interactions for accelerating CTR predictions in ad serving', *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, ACM, New York, NY, USA.

Devlin, J., Chang, M-W., Lee, K. and Toutanova, K. (2018) *Pre-Training of Deep Bidirectional Transformers for Language Understanding*, arXiv (1810)04805.

El-Gamal, A.H., Mostafa, R.R. and Hikal, N.A. (2020) 'Load balancing enhanced technique for static task scheduling in cloud computing environments', *Internet of Things–Applications and Future*, Springer Singapore, Singapore, pp.411–430.

Elhadad, A., Abbas, S., Abulkasim, H. and Hamad, S. (2020) 'Improving the security of multi-party quantum key agreement with five-qubit brown states', *Computer Communications*, Vol. 159, pp.155–160.

El-Hasnony, I.M., Mostafa, R.R., Elhoseny, M. and Barakat, S.I. (2021) 'Leveraging mist and fog for big data analytics inIoTenvironment', *Transactions on Emerging Telecommunications Technologies*, Vol. 32, No. 7, pp.1–16, doi:10.1002/ett.4057.

Elkabbash, E.T., Mostafa, R.R. and Barakat, S.I. (2021) 'Android malware classification based on random vector functional link and artificial jellyfish search optimizer', *PloS One*, Vol. 16, No. 11, p.e0260232.

Farouk, A., Alahmadi, A., Ghose, S. and Mashatan, A. (2020) 'Blockchain platform for industrial healthcare: vision and future opportunities', *Computer Communications*, Vol. 154, pp.223–235.

Harper, F.M. and Konstan, J.A. (2015) 'The MovieLens datasets: history and context', *ACM Trans. Interact. Intell. Syst.*, Vol. 5, No. 4, pp.1–19.

Heidari, S., Abutalib, M.M., Alkhambashi, M., Farouk, A. and Naseri, M. (2019) 'A new general model for quantum image histogram (QIH)', *Quantum Information Processing*, Vol. 18, No. 6, pp.1–20.

Jamjoom, M., Abulkasim, H. and Abbas, S. (2022) 'Lightweight authenticated privacy-preserving secure framework for the Internet of vehicles', *Security and Communication Networks*, pp.1–11, doi:10.1155/2022/6573060.

Jeon, M., Venkataraman, S., Phanishayee, A., Qian, J., Xiao, W. and Yang, F. (2019) *Analysis of Large-Scale Multi-Tenant GPU Clusters for DNN Training Workloads*, arXiv [cs. DC]. Available at. http://arxiv.org/abs/1901.05758

Jia, Z., Zaharia, M. and Aiken, A. (2018) *Beyond Data and Model Parallelism for Deep Neural Networks*, ArXiv [Cs. DC]. Available at http://arxiv.org/abs/1807.05358

Jiang, Y., Zhu, Y., Lan, C., Yi, B., Cui, Y. and Guo, C. (2020) *A Unified Architecture for Accelerating Distributed DNN Training in Heterogeneous GPU/CPU Clusters*, OSDI.

Jin, X., Li, X., Zhang, H., Foster, N., Lee, J., Soule, R. and Stoica, I. (2018) *NetChain: Scale-Free Sub-RTT Coordination (Extended Version)*, arXiv [cs. DC], Available at http://arxiv.org/abs/1802.08236

Keskar, N.S., Mudigere, D., Nocedal, J., Smelyanskiy, M. and Tang, P.T.P. (2016) *On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima*, arXiv [cs. LG], Available at http://arxiv.org/abs/1609.04836

Kim, J., Kim, M., Kang, H. and Lee, K. (2019) *U-GAT-IT: Unsupervised Generative Attentional Networks with Adaptive Layer-Instance Normalization for Image-to-Image Translation*, ICLR.

Klenk, B., Jiang, N., Thorson, G. and Dennison, L. (2020) *An InNetwork Architecture for Accelerating Shared-Memory Multiprocessor Collectives*, ISCA.

Lao, C., Le, Y., Mahajan, K., Chen, Y., Wu, W., Akella, A. and Swift, M. (2021) *ATP: In-Network Aggregation for Multitenant Learning*, NSDI.

Li, Y., Liu, I-J., Yuan, Y., Chen, D., Schwing, A. and Huang, J. (2019) 'Accelerating distributed reinforcement learning with in-switch computing', *Proceedings of the 46th International Symposium on Computer Architecture*, ACM, New York, NY, USA.

Liu, S., Wang, Q., Zhang, J., Lin, Q., Liu, Y., Xu, M. and He, J. (2020) *NetReduce: RDMA-Compatible in-Network Reduction for Distributed DNN Training Acceleration*, arXiv [cs. NI], Available at http://arxiv.org/abs/2009.09736

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C-Y. and Berg, A.C. (2016) 'SSD: single shot MultiBox detector', *Computer Vision – ECCV. 2016*, Springer International Publishing, Cham, pp.21–37.

Luo, L., Nelson, J., Ceze, L., Phanishayee, A. and Krishnamurthy, A. (2018) *PHub: Rack-Scale Parameter Server for Distributed Deep Neural Network Training*.

Mai, L., Rupprecht, L., Alim, A., Costa, P., Migliavacca, M., Pietzuch, P. and Wolf, A.L. (2014) *NetAgg: Using Middle-Boxes for Application-Specific On-Path Aggregation in Data Centres*, CoNEXT.

Mendonça, R.V., Silva, J.C., Rosa, R.L., Saadi, M., Rodriguez, D.Z. and Farouk, A. (2021) 'A lightweight intelligent intrusion detection system for industrial internet of things using deep learning algorithm', *Expert Systems*, p.e12917.

Metwaly, A.F., Rashad, M.Z., Omara, F.A. and Megahed, A.A. (2014) 'Architecture of multicast centralized key management scheme using quantum key distribution and classical symmetric encryption', *The European Physical Journal Special Topics*, Vol. 223, No. 8, pp.1711–1728.

Narayanan, D., Santhanam, K., Kazhamiaka, F., Phanishayee, A. and Zaharia, M. (2020) *Heterogeneity-Aware Cluster Scheduling Policies for Deep Learning Workloads*, ArXiv [Cs. DC], Available at http://arxiv.org/abs/2008.09213

Naseri, M., Abdolmaleky, M., Laref, A., Parandin, F., Celik, T., Farouk, A. and Jalalian, H. (2018) 'A new cryptography algorithm for quantum images', *Optik*, Vol. 171, pp.947–959.

Nemirovski, A., Juditsky, A., Lan, G. and Shapiro, A. (2009) 'Robust stochastic approximation approach to stochastic programming', *SIAM Journal on Optimization: A Publication of the Society for Industrial and Applied Mathematics*, Vol. 19, No. 4, pp.1574–1609, doi:10.1137/070704277.

Patarasuk, P. and Yuan, X. (2009) 'Bandwidth optimal all reduce algorithms for clusters of workstations', *Journal of Parallel and Distributed Computing*, Vol. 69, No. 2, pp.1–24.

Puthige, I., Bansal, K., Chahat Bindra, C., Mahekk Kapur, M., Dilbag Singh, D., Mishra, V.K., Aggarwal, A., Lee, J., Kang, B-G., Nam, Y. and Mostafa, R.R. (2021) 'Safest route detection via danger index calculation and K-means clustering', *Computers, Materials and Continua*, Vol. 69, No. 2, pp.2761–2777.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S. and Fei-Fei, L. (2015) 'ImageNet large scale visual recognition challenge', *International Journal of Computer Vision*, Vol. 115, No. 3, pp.211–252, doi:10.1007/s11263-015-0816-y.

Sharma, D.K., Singh, N.C., Noola, D.A., Doss, A.N. and Sivakumar, J. (2021) 'A review on various cryptographic techniques and algorithms', *Materials Today: Proceedings,* https://doi.org/10.1016/j.matpr.2021.04.583

Singhal, A. and Sharma, D.K. (2022) 'New generalized 'useful' entropies using weighted quasi-linear mean for efficient networking', *Mobile Networks and Applications*, https://doi.org/10. 1007/s11036-021-01858, pp.1–11

Tao, H., Al-Sulttani, A.O., Salih Ameen, A.M., Ali, Z.H., Al-Ansari, N., Salih, S.Q. and Mostafa, R.R. (2020) 'Training and testing data division influence on hybrid machine learning model process: application of river flow forecasting', *Complexity*, pp.1–22.

Zhou, N.R., Liang, X.R., Zhou, Z.H. and Farouk, A. (2016) 'Relay selection scheme for amplify-and-forward cooperative communication system with artificial noise', *Security and Communication Networks*, Vol. 9, No. 11, pp.1398–1404.

Zhu, F., Zhang, C., Zheng, Z. and Farouk, A. (2021) 'Practical network coding technologies and softwarization in wireless networks', *IEEE Internet of Things Journal*, Vol. 8, No. 7, pp.5211–5218.