National College of Ireland

# Energy-Efficient Data Optimization for Resource-constrained Edge/Fog Computing Devices

MSc Research Project
MSc Cloud Computing

Divya Henry
Student ID: x22241540

School of Computing
National College of Ireland

Supervisor: Shaguna Gupta

**National College of Ireland**

**MSc Project Submission Sheet**

**School of Computing**

| | |
|---|---|
| **Student Name:** | Divya Henry |
| **Student ID:** | X22241540 |
| **Programme:** | MSc Cloud Computing **Year:** 2023 - 2024 |
| **Module:** | MSc Research Project |
| **Supervisor:** | Shaguna Gupta |
| **Submission Due Date:** | 16 – 09 – 2024 |
| **Project Title:** | Energy-Efficient Data Optimization for Resource-constrained Edge/Fog Computing Devices |
| **Word Count:** | 9310 **Page Count**: 23 |

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

**Signature:** 

**Date:** 16 – 09 – 2024

**PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST**

| | |
|---|---|
| Attach a completed copy of this sheet to each project (including multiple copies) | □ |
| **Attach a Moodle submission receipt of the online project submission,** to each project (including multiple copies). | □ |
| **You must ensure that you retain a HARD COPY of the project**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. | □ |

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

| **Office Use Only** | |
|---|---|
| Signature: | |
| Date: | |
| Penalty Applied (if applicable): | |

# Energy-Efficient Data Optimization for Resource-constrained Edge/Fog Computing Devices

Divya Henry
x22241540

**Abstract**

The presence of fog and edge computing devices everywhere have led to the generation of high volumes of data across their nodes resulting in great difficulties while handling and processing such massive volumes of data due to limited computing power at the edge or fog devices. This demands the case for an optimized model for devices in edge/fog computing environments that can achieve optimal data compression and efficient data transfer across the edge/fog nodes by following green cloud computing practices. The proposed method integrates deep compression sensing autoencoder network (DCSANet) for data reduction and accurate reconstruction with deep reinforcement learning (DRL) based joint computing framework (JCF) for intelligent collaboration among fog nodes sharing resources. DCSANet aims at learning how to generate smaller representations of information through dimensionality reduction thereby generating compressed data without sacrificing too much on reconstruction accuracy during the recovery process at the receiving nodes, while still ensuring data fidelity in reproduction. JCF enables smart cooperation between fog nodes to determine best decisions regarding computation offloading along with energy-efficient processing and handling of shared resources for data processing at the fog nodes. The effectiveness of the proposed DRL-based JCF and DCSANet approach will be analyzed by its ability to improve energy efficiency, reduce transmission latencies as well as improve reconstruction accuracy using real-world IoT datasets, thereby showcasing its potential towards providing optimal solutions for efficient data aggregation within resource limited-edge devices while cutting down computational complexities and network bandwidth requirements for a sustainable fog/edge computing environment.

**Keywords−** compressed sensing, deep learning, edge/fog computing, energy efficiency, data aggregation, computation offloading

# 1 Introduction

## 1.1 Motivation and Problem Background

Internet of things (IoT) has seen a fast growth recently resulting in an exponential rise in data being created at the network edge (Albreem et al.; 2021). The conventional cloud architectures fail to cope with this huge amount of information due to their latency issues, limited bandwidth and security insufficiencies, among others. This has led to the popularity of fog and edge computing as they can solve these problems by bringing computation and storage resources nearer to the data source, thus reducing both latency and bandwidth usage at the same time (Yousefpour et al.; 2019). However, energy efficiency becomes a big challenge due to resource-constrained environment found at edges or fogs, which can also impact the management of data (Pereira et al.; 2020).

Various methods such as dimensionality reduction or compression are used to reduce the load on devices located on edges/fogs, but it still does not resolve the issue of information being lost on the compressed versions of original datasets. Compressed sensing (CS) is one technique used widely today when it comes to reducing amounts of information being transmitted through networks, while Deep Learning (DL) can help improve CS performance by training optimal sensing matrices and reconstruction algorithms (Wu et al.; 2020). Both these techniques combined together have produced satisfactory results especially in areas like wireless communication, image processing or IoT data compression. It should be noted, however that even though much progress has been made regarding data size reduction, energy efficiency remains very low for fog devices since they require high power for their operations affecting not only operational costs, but also the environmental conservation efforts necessitating the use of greener technologies within this domain (Gougeon et al.; 2020).

Besides compressing data, computation offloading (Khan et al.; 2020) has shown potential in enhancing energy saving capabilities together with overall performance levels demonstrated by IoT systems. The main idea behind this strategy is to move computationally intensive tasks from mobile devices onto more powerful servers located either at edges or in cloud servers, enabling such devices save power as well as reducing delays experienced during processing. However, knowing when exactly to offload computations can be tricky owing to such factors as network state, device capability or application requirement, thus making it difficult to obtain optimal solutions without using machine learning (ML) approaches like reinforcement learning (RL) which do not need accurate models about systems involved to make good decisions on offloading computations.

This research attempts to address the issue of high energy consumption in the operation of fog/edge devices using data compression and reconstruction, data aggregation and an intelligent mechanism to communicate across the fog/edge nodes, offload computations among them for optimal data handling and increase the energy efficiency as a result (Firouzi et al.; 2022).

## 1.2 Problem Statement

The first challenge focused on this research is high energy and limited computational capacity of edge and fog devices when processing a growing amount of data collected by IoT sensors. The implementation of resource-limited devices requires the design of a lightweight framework that can deliver considerable data compression, edge computation outsourcing, and collaborative learning, while following green IT principles (Chen et al.; 2022). The key challenges include minimizing size of data that is transferred between the edge and fog layers while preserving the quality of the reconstructed information, how data can be aggregated at the nodes in order to avoid relaying redundant data, how and where computation should be offloaded using MARL optimized for the network's conditions, the capabilities of devices, and the requirements of the application.

## 1.3 Research Question

How can an optimized strategy be designed for IoT devices in edge/fog computing networks that achieves substantial data reduction, and efficient computation offloading while improving energy efficiency by adhering to green computing practices?

## 1.4 Research Solution

The proposed solution answers this question in two distinct ways. Firstly, a DL-based compression sensing (CS) technique for dimensionality reduction and precise reconstruction of the high-dimensional data generated across the fog nodes is suggested. The model learns the latent representations of the high-dimensional data to achieve data reduction without compromising on the crucial information that may be required in the data reconstruction. This can also be used to enable data aggregation by combining fragments of edge data across the nodes before transmitting it to the cloud server. Secondly, to achieve higher energy efficiencies by making optimal decisions on data processing, computation offloading and resource allocation while sharing the aggregated data across the nodes or the cloud a DRL-based framework is proposed. This research also strives to fulfil the green cloud computing practices to reduce the adverse environmental impacts by designing a system that performs optimally with lesser energy consumption and lower the carbon footprint.

## 1.5 Research Objective

The primary aim of this research is to produce an energy-efficient optimization architecture for optimal data compression, information aggregation, and intelligent collaborative network in terms of computation offloading and efficient data handling platform as a service on resource-constrained edge and fog computing devices considering IoT networks.

## 1.6 Research Contributions

The key contributions of this research were as follows:

- The design and implementation of a deep compression sensing autoencoder network (DCSANet) based on autoencoders, aiming to guarantee the minimal data reduction while ensuring accurate data reconstruction.
- To a develop a DRL powered JCF for edge nodes enabling offloading of computations, data handling and processing efficient to work together in an intelligent manner.
- Measure and compare the performance results of proposed framework in terms to reconstruction accuracy, energy consumption, energy efficiency, throughput as well transmission latency across the edge nodes.
- Map the proposed framework to green cloud computing needs for reducing carbon footprint and adopting sustainable mean in the edge/fog environments.

## 1.7 Limitations and Scope

The scope of this research is to create an efficient system for reducing data and saving energy when transmission in edge and fog computing. This framework is designed to work with IoT devices that have limited resources and should reduce both power consumption and computational workload that come with processing massive amounts of information. This research contains certain limitations like:

- The integration of the proposed framework with existing IoT protocols and standards may require additional adaptations and optimizations.
- This study does not explicitly deal with security/privacy issues regarding data storing/transmitting within an edge-based or fog computing setting thus further steps should be taken in securing such information.

The deep compression sensing autoencoder network (DCSANet) and DRL based joint computing framework (JCF) are proposed in order to achieve significant amounts of data reduction, energy-efficient aggregation as well as intelligent collaboration for computation offloading among edge nodes. In line with green cloud computing practices, this research

contributes towards sustainable development in terms of environmental friendliness within the context of fog computing at edges. The performance evaluation which uses real IoT datasets strives to demonstrate that this framework is capable of achieving better reconstruction accuracy, reduce transmission delays and be more energy efficient.

Following is the analysis of the report, and it is divided into major sections. Section 2 contains related work that discusses previous works on data compression and energy-efficient processing on edge and fog networks and presents the gaps that this research will solve. The section 3 'Research Methodology' explains the method applied together with the selection of datasets, data pre-processing procedure, and integration of DCSANet and JCF models. In Sections 4 and 5, the design specifications and implementation sections are presented where details of development of these models are specified. Section 6 is the evaluation of results which includes the analysis of the simulation results about the examined performance metrics and the integration of DCSANet with JCF. Section 7 elaborates the Conclusion and Future Work where recommendations were given regarding further investigations and development.

# 2 Related Work

## 2.1 Data Compression in Edge Networks

A study by Amarlingam et al. (2018) introduced a data aggregation technique that is energy efficient for wireless sensor networks. The method utilizes an adaptive dictionary in compressed sensing. The solution being presented applies an overcomplete dictionary that has been acquired through training data to effectively manage fluctuations in data sparsity as time progresses. The simulation results demonstrate that the suggested method provides significant energy savings and improves the lifespan of the network when compared to existing compressed sensing systems that utilize fixed sparse bases such as DCT, DFT, and DWT. In (Azar et al., 2019), an additional energy-efficient IoT data compression technique for edge machine learning applications was presented. The SZ compressor, which is a high-speed compressor with error control capabilities, is specifically designed for IoT devices. Its purpose is to minimize the volume of data that needs to be transported from the edge (where the devices are located) to the cloud. This study examines the effect of lossy compression on the precision of a machine learning model used for detecting driver stress, which is deployed at the edge. Results demonstrate a significant data reduction of up to 103 times without compromising the accuracy of stress detection.

In their study, (Shao and Zhang. 2020) proposed the utilization of BottleNet++, which is a very efficient feature compression. It is specifically aimed for collaborative inference systems of device-edge and is incorporated into deep learning framework. The architecture of the BottleNet++ comprises an encoder block, a channel that is non-tunable, and a decoder so that the use of the system is end-to-end joint source-channel. BottleNet++ is able to compress as much as 256 times while utilizing efficiency from the sparsity and tolerance to failure of the intermediate DNN features. All this is attained even when gaining data through binary erasure channel, with a probability loss of less than 2%. This makes it possible to split DNN at early stages and thereby reduce on-device calculations by as much as three times when compared to other methods. For edge-fog computing-based IoMT networks, the authors of (Idrees et al., 2022) presented the KCHE lossless EEG data compression technique. KCHE utilizes spatial similarity in EEG data to compress it at the edge prior to transmission to the fog layer. At the fog layer, a machine learning technique utilizing Naive Bayes is employed to identify epileptic episodes from the reconstructed EEG data. The results indicate that KCHE achieves a

compression rate of 85-89% and minimizes energy usage at the edge node when compared to not using compression. Additionally, it enables precise seizure identification at the fog layer.

**Table – 1: An analysis of relevant research works presented with their objectives, methodology, dataset used, tools and metrics.**

| No. | Author(s) and Year | Research Objective | Methodology | Dataset & Tools | Metrics |
|---|---|---|---|---|---|
| 1 | Zhang et al. (2019) | Propose DL-based stacked sparse denoising autoencoder compressed sensing (SSDAE_CS) model for signal reconstruction from compressed sensing. | SSDAE_CS with encoder and decoder sub-networks, multiple nonlinear measurements, end-to-end training. | Simulation datasets<br><br>MATLAB, Tensor | Training Loss, PSNR, MSE |
| 2 | Toor et al. (2019) | Address high-energy consumption in IoT-EoT Devices. | Dynamically scaling the processor frequency of EoT devices | Simulation datasets<br><br>iFogSim | Energy Consumption, Network Usage, Execution Time and Loop Delays |
| 3 | Li et al. (2019) | Propose an energy-efficient data collection scheme for large-scale IoT using computation offloading and compressed sensing. | Clustering Compressive Data Collection (CCDC) framework | Real-world sensor datasets<br><br>MATLAB iFogSim | Normalized MAE (NMAE), Average Energy Consumption, and Total Data Transferred |
| 4 | Zhang et al. (2019) | Develop a two-stage compressed data aggregation scheme using CS and sparse Autoencoder (SAE) for green IoT networks | Two-stage compressed data aggregation scheme using CS and Sparse Autoencoder (SAE) | Synthetic dataset, MNIST dataset Simulation Tools | MSE, CR, Reconstruction Accuracy, Training loss and Energy Consumption |
| 5 | Zhang et al. (2021) | Propose a learning-based sparse data reconstruction scheme for compressed data aggregation in IoT networks. | Deep Compressed Sensing Network (DCSNet) | Real-sensor dataset.<br><br>MATLAB, PyTorch | PSNR, MSE, Structural Similarity Index (SSI) |
| 6 | Bai et al. (2021) | Propose a joint optimization algorithm for computation offloading, data compression, energy harvesting, and application scenarios (JCDEA) in fog computing | JCDEA Framework | Synthetic data<br><br>Simulation Tools | Energy consumption; data compression ratio; compression and decompression CPU metrics |
| 7 | Yang et al. (2021) | Develop a transfer learning-enabled edge-CNN framework for 5G industrial edge networks focusing on energy efficiency and latency reduction | Implementation of a TL-enabled edge-CNN framework using mathematical modeling | ImageNet dataset and Others<br><br>Simulation Tools, Keras, TensorFlow | CR, energy consumption, uploading decision and bandwidth allocation |

| | | | | | |
|---|---|---|---|---|---|
| 8 | Alotaibi et al. (2022) | Propose an efficient IoT system for collecting, compressing, transmitting, and reconstructing sky images with minimal bandwidth usage. | Convolutional Autoencoder (CAE) | Sky images dataset<br><br>Raspberry Pi, Python, Keras | MSE, SSI, CR |
| 9 | Chen et al. (2022) | Propose DRL-based cloud-edge collaborative mobile computation offloading (DRL-CCMCO) mechanism to minimize execution delay and energy consumption in industrial networks. | DRL-CCMCO framework | Simulation datasets involving mobile industrial applications.<br><br>Python, TensorFlow | Energy consumption, task completion cost, and resource allocation efficiency |
| 10 | Noura et al. (2023) | Propose a DL-based super-resolution model for recovering high-quality decompressed images in multimedia IoT (MIoT) networks | Lossy Image Compression, DL-based Image Enhancement | The Kodak dataset (24 images)<br><br>PyTorch | PSNR, CR, SSI and energy consumption |
| 11 | Tan et al. (2023) | Minimize total energy consumption in multi-access edge computing (MEC) systems while satisfying a delay constraint | DRL-based bilevel optimization (Dueling-DQN, Double-DQN) | Simulation data with scenarios involving mobile users, MEC servers, and a cloud server.<br><br>Simulation Tools | Energy consumption, discount factor, subcarrier and power allocation |
| 12 | Wei et al. (2023) | Propose MA-GAC for many-to-many task offloading in vehicular fog computing | Multi-Agent Gated actor Attention Critic (MA-GAC) within partially observable Markov decision process (POMDP) | Simulation data involving vehicular fog computing scenarios.<br><br>Simulation Tools | Energy consumption, service latency, task completion ratio and offloading decision |

## 2.2 Energy Efficient Processing and Latency Reduction in Edge/Fog IoT

A mobile edge-cloud collaborative computing technique for power consumption minimisation throughout in mobile devices was discussed by (Hua et al.; 2023) with the help of mobility prediction. It is cast as an MIP problem because it entails the power control, transmission scheduling, and offloading strategies all at the same time. Moreover, another heuristic algorithm called mobility-aware heuristic (MAH) with a low computational cost is also presented. The results demonstrate that the proposed scheme will be more effective in reducing MD energy usage as compared to various schemes with mobility consideration. To achieve this objective, the study undertaken by (Huang et al.; 2023) targeted at an adaptive computation offloading and resource allocation in the dynamic Internet of Vehicles (IoV) environment with integration of edge and cloud computing. The objective is to decrease the total cost of processing urgent and complex jobs, which must be accomplished in a short amount of time and within limitations imposed by the transmission rate. A DRL-based CORA algorithm is proposed to obtain the best strategy in response to the dynamic variations in the network environment. The simulation experiments show that compared with other non-DRL algorithms and other DRL algorithms, the CORA algorithm has better performance in terms of training convergence, processing time, and processing cost.

The authors in (Ren et al.; 2018) studied the issue of latency minimization in a MECO system but in multi-user scenario focusing on communication and computation resource management. The authors examine three distinct computational models: local compaction, edge cloud compaction and partial compaction offloading. Optimal resource allocation and minimum system delay equations are developed for both local cloud and edge cloud compression models. The numerical results show that the proposed idea of partial compression offloading has a significant effect on eradicating latency from start to finish. Namely, the study by (Trinh et al. 2018) aimed to assess thorough the application of MEC for the processing of real-time visual data during disasters. A cost model for offloading decision considers the merits of different computing policies, particularly whether to offload to edge or core cloud, based on a number of workloads and clients. The SPIDER algorithm which is policy and intelligence-based along the sustainable policies with edge routing uses machine learning for improving the spatial routing and the incorporation of adaptive rules. The benefits of maintaining an energy economy while simultaneously achieving low latency are exemplified in the experiment on a testbed setup and performance simulations of reconstructed disaster scenarios.

For instance, deep compressive offloading was presented as a new offloading framework in a recent study by (Yao et al.; 2020). This framework is realizing a combination of compressive sensing theory and deep learning to reduce the overall offloading data transmitted. Encoder is a much lighter program, which compresses the data on the side of the local device, and decoder does the same on the side of the edge server. As a result, one receives perfect reconstruction and lossless inference with theoretical performance assurance. Assessments indicate that there is 2-4 times decrease in the overall time it takes for a task to be completed, while maintaining an accuracy loss of less than 1%, as compared to the most advanced methods for transferring work to another location. During periods of restricted bandwidth or heavy traffic, latency is decreased by a factor of up to thirty-five. A system was presented that combines unmanned aerial vehicles (UAVs) and MEC to offer MEC services to ground users in (Pervez et al.; 2023). The system involves numerous UAVs and a base station. The goal is to minimize a weighted combination of energy consumption and latency by simultaneously optimizing decisions about work offloading, transmission power, UAV trajectories, and CPU frequency allocation. The authors suggest using an alternating iterative strategy, specifically the block descent method, to tackle this non-convex issue. The simulation results clearly illustrate the superior performance of the proposed algorithm in comparison to benchmark systems.

From the analysis of the available literature, models and the realized objectives, the research gap was highlighted with the suggestions of an innovative approach, (DCSANet + DRL-based JCF) to optimize the transfer of data and many data reductions from edge/fog computing to enhance energy performance. The techniques that are proposed in the model include DCSANet for data reduction and accurate reconstruction with high efficiency and DRL-based JCF for fog node intelligence. This goal will help in minimizing energy consumption and transmission time while at the same time enhancing the quality of reconstructed IoT dataset. The integration between DCSANet and DRL-based JCF may be costly in terms of computational requirements and thus requires AWS like cloud service rather than spent on a local development server. However, it is also necessary to note that to learn about cloud providers' services, which can be used for studying, it may take a lot of time and domain knowledge.

**Table – 2: An analysis of relevant research works presented in literature in terms of their key findings, strengths, weaknesses and limitations.**

| No. | Author(s) and Year | Key Findings | Strengths | Weaknesses | Gaps |
|---|---|---|---|---|---|
| 1 | Zhang et al. (2019) | Improved CS performance by jointly training encoder-decoder sub-network parameters | Improved reconstruction accuracy, lower time cost, strong denoising ability | High training time and data requirement | Needs more efficient training approaches to reduce time and data requirements |
| 2 | Toor et al. (2019) | Better energy conservation by dynamically scaling the processor frequency of EoT devices based on traffic loads | Energy efficiency, maintains counters to avoid frequent speed changes, extensive simulations for validation. | High computational complexity for real-time implementation | Requires real-world validation, need for optimization for diverse IoT applications. |
| 3 | Li et al. (2019) | Reduced data volume while maintaining high reconstruction accuracy, demonstrated high energy efficiency | Efficient data reduction, prolonged network life, improved accuracy in data collection | High computational complexity for real-time implementation | Needs real-world validation, Lacks diverse IoT applications optimization |
| 4 | Zhang et al. (2019) | DCSNet offers effective data compression and reconstruction in large-scale IoT networks | Reduced data traffic, Improved reconstruction accuracy, Low compression ratios | Limited to simulation data, complexity in real-world implementation | Real-world validation needed, diverse IoT network conditions |
| 5 | Zhang et al. (2021) | Achieved high reconstruction accuracy with lower data transmission requirements, effective data volume reduction while maintaining high reconstruction fidelity | High reconstruction accuracy, data compression, lower transmission delay, and energy savings. | High computational complexity for training, dependence on the availability of large datasets for training. | More practical implementations, real-world validation, and IoT scenario optimizations are needed. |
| 6 | Bai et al. (2021) | Fog computing cost lower than cloud computing, Cost increases linearly with offloaded data, users, devices, and compression ratio | Comprehensive multi-objective optimization approach, extensive simulation setup with varying parameters | High computational overhead, sophisticated scalability and modularity in large-scale fog computing environments | Lacks consideration of real-world constraints such as network variability and hardware limitations |
| 7 | Yang et al. (2021) | Eighty-five percent prediction accuracy with compression ratio of 32, reduced training and testing overhead | Privacy-preserving, efficient energy use, reduced latency | Limited to simulation data, complexity in real-world implementation | Need for real-world validation, diverse industrial scenarios |
| 8 | Alotaibi et al. (2022) | Compressed images to 2% of their original size, maintained high reconstruction fidelity with an average SSIM of 99% | High compression ratio, high reconstruction quality, practical implementation on low-cost IoT device (Raspberry Pi). | Computationally intensive training process, dependence on large datasets for training. | Need for real-world validation, diverse industrial scenarios, and IoT scenario optimizations. |

| | | | | | |
|---|---|---|---|---|---|
| 9 | Chen et al. (2022) | Optimized resource allocation, handled task relevance and user mobility effectively. | Faster convergence, high stability, smaller execution delay and low energy consumption. | High computational complexity, reliance on simulation environment for validation. | Needs real-world validation, optimization for various industrial applications. |
| 10 | Noura et al. (2023) | Improved visual quality, Reduced latency and energy consumption by 10% | Enhanced image quality, flexible application | Limited to specific datasets and techniques, computationally intensive | Real-world validation, exploration of other DL models |
| 11 | Tan et al. (2023) | Near-optimal energy efficiency and task completion rate, DBA outperforms other DRL-based approaches | High performance in energy efficiency and task completion | Limited to simulation scenarios, complexity in implementation | Real-world validation needed, exploration of diverse settings |
| 12 | Wei et al. (2023) | Higher long-term rewards, coordinated task offloading improves computational resource utilization | Effective coordination among multiple agents, robust performance in dynamic environments | Limited to simulation scenarios, computational complexity | Real-world validation needed, exploration of different topologies |

## 2.3 Critical Analysis

The literature assessment presented in Table-1 compares different models and strategies for energy-efficient data optimization in the resource-constrained environment of fog computing, revealing their goals, major claims, and limitations between these models. This study examines various models and techniques such as deep learning (DL)-based compression, transfer learning (TL) for edge CNN, deep reinforcement learning (DRL) for task offloading and resource allocation as well as joint optimization of computation offloading, data compression, energy harvesting, and application scenarios.

These approaches aimed at reducing transmission latency while maintaining high-quality data reconstruction, but they required GPU resources at application server level, analyzed video compression effectiveness in a limited scale, and required scalability validation in large IOT networks to a larger extent. This research tries to combine the concept of data compression and reinforcement learning for efficient data transmission across edge nodes from (Zhang et al.;2019). The DCSANet architecture builds on this study, which investigated compressive sensing and autoencoder-based methods for compressed data aggregation in green IoT networks. Their major contributions include efficient data compression and reconstruction with reduced data traffic, improved reconstruction accuracy, and low compression ratios This design is mandatory for performing optimized data reduction and accurate reconstruction within the edge/fog computing domain. Nonetheless, Zhang et al.'s work had some drawbacks. The current research extends this in two ways, by incorporating DCSANet into a DRL-based JCF model and by conducting evaluation on real datasets.

Although there are some limitations in the discussed literature, requiring further examination. For example, the review suggests it is a challenge to implement DL models on low-powered equipment and recommends testing them under dramatically changing network conditions with

varying task requirements. Recommendations for testing these models on real-life high-quality image datasets and comparing these to similar frameworks that this study aims to address. Furthermore, an exhaustive literature review was undertaken to analyze different techniques and models associated with the energy efficient operation in the edge and fog computing system. This review also highlights the gaps and limitations that exist in the current research, which led to the formulation of the proposed model. The integrated model also applies DCSANet for data dimensionality reduction while the intelligent cooperation mechanism (JCF) for cooperation between fog nodes.

# 3 Research Methodology

The research methodology also seeks to meet the research question formulated under Section 1.3 of this dissertation by designing the best strategy regarding the use of edge and fog computing devices. This strategy involves compacting and reconstructing data produced/used in the different nodes and enhancing the efficient transfer of data to help in energy conservations as shown in Figure. 1.

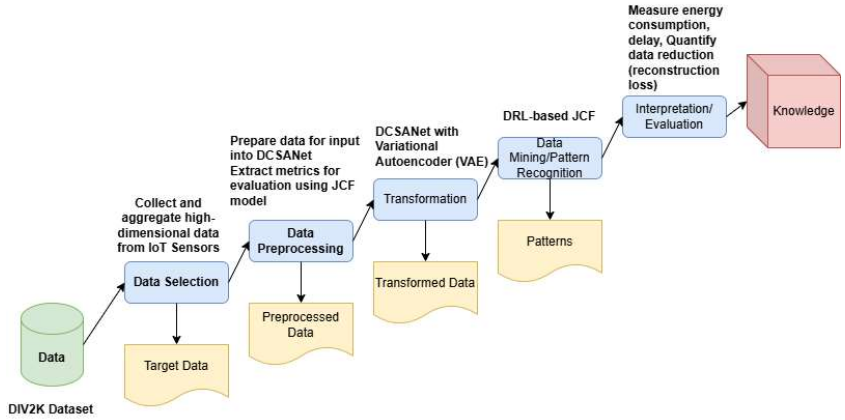



**Figure 1: KDD Process Research Methodology**

## 3.1 Dataset Description

To evaluate the proposed methods this study uses the DIV2K dataset[1], a high-quality image dataset created for super-resolution tasks. It comprises one thousand 2K resolution images which will be pertinent for testing purposes of data reduction and image compression. This dataset is stable and dependable which enables us to train and test the models effectively. Ethical Considerations: The datasets applied in the present work, including the public benchmarking dataset or the DIV2K, are popular among researchers globally. It should be noticed that none of these datasets contains any identifiable resources or people's information, thus meeting the ethical requirements and privacy standards and at the same time offering efficient solutions to numerous issues.

## 3.2 Data Preprocessing

The selected data are then subjected to the following preparation techniques in order to be enhanced for their use in DCSANet. This entails resizing images used in the DIV2K dataset to an appropriate dimension suitable for the autoencoder, normalizing various sensors to have homogeneity, and discretizing of various features that are categorical in nature. Further, the dependent variables including energy, latency, and other evaluative parameters are extracted

[1] https://data.vision.ee.ethz.ch/cvl/DIV2K/

from the datasets to compare it with the JCF model. The preprocessing stage involves converting unprocessed data into a form that the DCSANet may suitably apply. This ensures that the latter data compression and computing framework is done accurately as well as effectively.

## 3.3 Data Transformation

During the transformation stage, DCSANet is employed to the benefits of data reduction by employing the compression sensing techniques. It reduces input data into a lower but intermediary representations through a Variational Autoencoder (VAE) which had shown that reducing dimensions simplified representations of substantial relevant data to be implemented (Zhang et al.; 2019). The VAE is learned to make the reconstruction loss small so that compressed data has the required qualities for the next step. This transformation is decisive for shrinking the size of the data which in turn decreases the bandwidth and storage space, hence a better and scalable system.

## 3.4 Data Mining/Pattern Recognition

The Joint Computing Framework (JCF), built on DRL, is utilized to acquire patterns and enhance resource allocation. It uses a Deep Q-Network (DQN) technique to train the DRL agent, which observes the present condition of the fog/edge environment, choosing actions based on its learned policy (Q-function), and adjusts its actions based on the rewards it receives. This method involves:

- State Observation: The agent actively monitors the present situation of the network, including the availability of resources, network conditions, and energy consumption levels.
- Action Selection: The agent decides on actions, such as adjusting data transfer rates, allocating resources, and offloading tasks, based on the observed state.
- Reward Evaluation: The agent receives rewards based on how well it performs its actions, encouraging behaviours that reduce energy usage and delays while maximizing data processing efficiency.

The agent regularly updates its policy to improve the decision-making as network conditions and data characteristics change. The JCF utilizes this intelligent decision-making process to maximize resource utilization in the fog/edge network, ensuring effective data processing and transmission.

## 3.5 Evaluation

The effectiveness of the proposed approach is evaluated through extensive simulations and real-world experiments that focus on several key areas:

- Energy Consumption: This establishes the energy consumption of edge/fog nodes and compares it with the existing techniques for energy saving.
- Latency: This may be defined as the time taken in transferring and interpreting data from one location to another location. In this case, tests are performed to demonstrate the difference between this latency and traditional IoT cloud models, focusing on the lower latency obtained.
- Reconstruction Loss: This shows the amount of data compression that DCSANet is able to achieve while at the same time minimizing on the loss that occurs, therefore data quality is preserved.
- Training Loss: This means that the model is learning during the training process with the smaller values of the Keras ‘loss’ parameter denoting better fit between predicted and actual targets.

In comparison of training performance of the model, PSNR, MSE and SSIM indices are utilized as model assessment criteria. PSNR is one of the reconstruction quality measures which compares reconstructed data to the original one. Therefore, the higher the value then the better is the quality of the reconstructed image. The degree of distortion between the original data and reconstructed data is evaluated with Mean Squared Error (MSE). SIM evaluates the similarity between the original and reconstructed images from the visual point of view and gives the rating that is more related to the quality. This implies that the higher the value then the better is the perceptual quality.

The code is written in Python and the DCSANet and JCF models are developed using TensorFlow and PyTorch while other machine learning libraries used include NumPy, Pandas and scikit-learn.

### 3.6 Knowledge

The framework of edge/fog computing proposed in this work is designed to minimize the computational overhead and to maximize the reconstruction quality. It is intended to help to optimize energy consumption in data handling and transfer, minimize latency in edge and fog networks, manage the resources and implement cooperation between the fog nodes. This implementation is intended to develop energy efficient edge/fog computing systems for IoT applications with a view to solving the challenge of resource limitations.

## 4 Design Specifications

### 4.1 DCSANet Architecture

The key task of the Deep Compression Sensing Autoencoder Network (DCSANet) as presented in Figure.2 is to achieve the best data compression and accurate data recovery in resource-limited edge/fog computing environments. The system model consists of two parts: an encoder for dimensionality reduction and a decoder for data reconstruction.

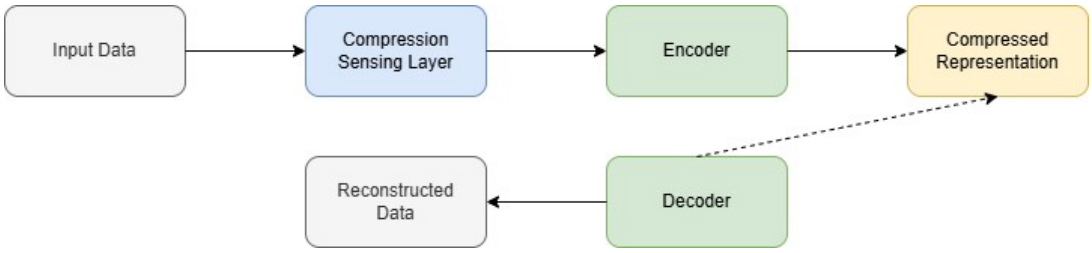



**Figure 2: DCSANET Architecture**

DCSANet can take high-dimensional data and compress it into a smaller, dense latent space representation. By using convolutional layers, it extracts spatial features and reduces the input dimension. The latent representation passes through activation functions, such as ReLU, to add non-linearity and enhance sparsity. Pooling layers on the other hand helps to further reduce computation, they further down-sample the feature maps. To revert the compressed data back to its original form, the decoder uses transposed convolutional layers to up sample the data to a size of 32×33(width/height). Skip connections are implemented between corresponding encoder and decoder layers to ensure that valuable information is preserved, leading to better reconstruction quality.

During the training process, DCSANet undergoes end-to-end learning with various regularization techniques alongside reconstruction loss functions like Structural Similarity Index (SSIM) and Mean Squared Error (MSE). Regularization methods, such as L1 and L2, help maintain sparsity and prevent overfitting (Zhang et al., 2019).

Throughout training, DCSANet learns to create compressed representations that allow for nearly accurate reconstruction of the received signals without needing all the original information. This compressed data is then transmitted over the network, reducing the need for bandwidth and minimizing latency. At the receiving end, the decoder reconstructs the original data from the compressed representation.

Being lightweight and computationally efficient, the DCSANet architecture is designed to run on edge/fog devices that have limited resources. It can achieve this by utilizing the properties of convolutional layers which allow for efficient extraction of features as well as dimensionality reduction, while the autoencoder structure supports seamless learning of both compression and reconstruction processes.

The proposed framework with its decentralized approach can distribute decision-making among the various fog nodes, thus making the system scalable to handle larger workloads as more nodes are incorporated. DCSANet' s compression at the edge has the effect of decreasing bandwidth usage, which also means that the system is capable of handling more edge/fog nodes without congesting the system.

## 4.2 JCF Architecture

Every single DRL agent perceives the current condition of network which includes resource availability, network status as well as energy consumption level. In light of this perceived state, an agent will choose actions like modifying data transfer rates, allocating resources or offloading tasks to other fog nodes or cloud server depending on what it observes. These actions are selected according to learned policy represented by DQN model, whereby rewards received by such DRLs are based on their performance in terms of energy efficiency, and latency reduction during the processing stage. Rewards act as a guide during learning process, thereby encouraging them to take actions that minimize energy consumed while also minimizing latency caused through maximum use of resources available at hand. In response to received rewards, agents continue updating policies so as fit better into new conditions brought about by varying network conditions (Yang et al., 2021).

The JCF architecture adopts centralized training method where global policies are trained for DRL agents. Training set is composed of historical states recorded from various fog nodes, while actions taken within those periods and rewards achieved at execution time at specific nodes in the edge/fog network are considered. Centralized training enables knowledge sharing between different nodes which promotes coordination among them in terms sharing resources as well offloading tasks optimally.

During deployment, trained DRLs become distributed over various locations, where real time decision making takes place using learned policies as guidelines for deciding what needs to be done, when and how things should happen at every point in time. These communicate among themselves so as maintain worldwide views of network state around them while making rational choices. What JCF does best is, it enhances the use of computing resources efficiently, reduces energy consumption and minimizes delays in edge/fog environment.

## 4.3 DCSANet-JCF Integration

The purpose of integrating JCF and DCSANet is to create an efficient system for energy-efficient data optimization in edge/fog computing systems with limited resources. In this combined methodology presented in Figure 3, DCSANet is utilized at the edges to compress the generated data before transmitting them, thus reducing bandwidth usage and transmission latency.

The information is sent to fog nodes after compression, where decision making is handled by JCF. JCF uses DRL agents to evaluate compressed information and determine the best actions for tasks like computation offloading, resource allocation as well as data processing.

The advantages of integrating DCSANet and JCF include:

- Bandwidth requirements and transmission latency which leads to a faster data transmission between edge devices can be reduced through data reduction achieved by DCSANet.
- Intelligent decision-making functions embedded in JCF optimize resource utilization while considering energy efficiency aspects within fog computing environments.

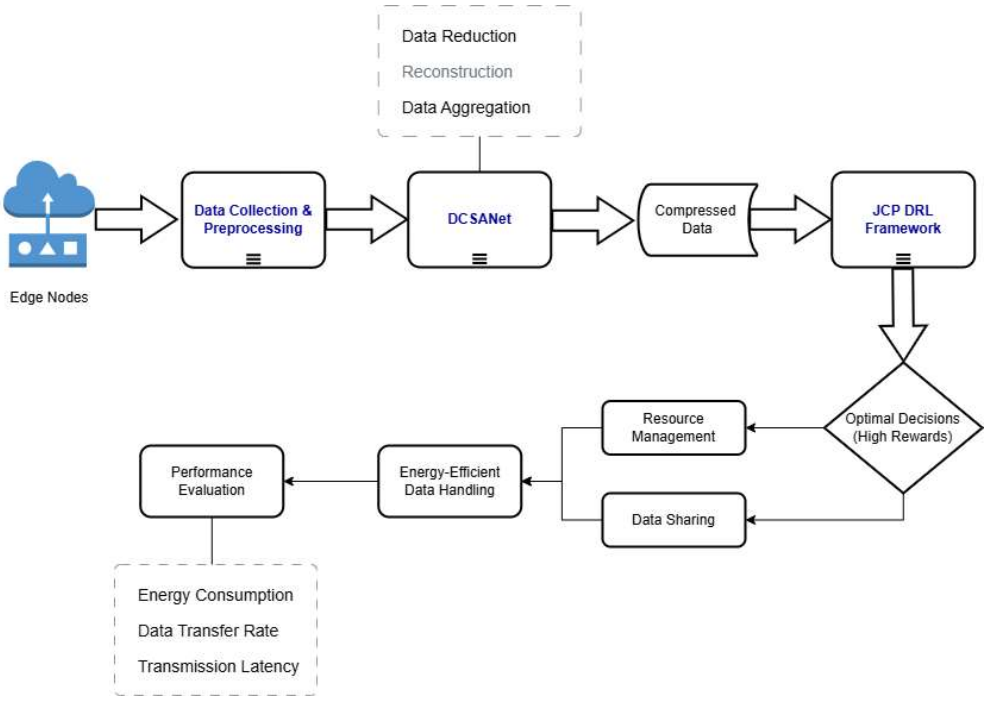



**Figure 3: DCSANet-JCF Integration for Energy Efficient Data Reduction**

When the compressed data is received DCSANet decoder reconstructs it so that it can be recognized. Furthermore, the reconstructed data is then processed based on the decisions made by various Deep Q-Network (DQN) agents in JCF, and depending on network conditions and application needs, it can either be sent back to the edge device or forwarded to a cloud server. Overall, combining DCSANet and JCF achieves the goal of reducing data while working with intelligent resource management to improve both performance and energy efficiency in an edge computing or fog computing system. This joint framework is dynamic enough to suit IoT networks, deal with huge amounts of produced data as well as make the best use out of limited computational resources.

# 5 Implementation

## 5.1 Development Environment

The DCSANet and JCF models' code was written in Python utilizing TensorFlow and PyTorch libraries, respectively, and the implementation was done on Google Cloud Platform (GCP), particularly Colab. Other libraries that were utilized for the data preprocessing, file handling, and performance assessment include NumPy, Pandas, and Scikit-learn respectively while JupyterLab was used for the debugging and the visualization of results using libraries such as Matplotlib and Seaborn. The model was trained on a NVidia T4 GPU in order to accelerate the training and the evaluation of the model.

## 5.2 DCSANet Model Implementation

The DCSANet model architecture was implemented with one encoder for dimensional reduction and a decoder to regress the data. The encoder uses convolutional layers to reduce the spatial measurements and simplify them. It uses multiple layers of convolution, increasing filter sizes (32, 64, and 128), and strides with a step-size of '2' to reduce the input size. A residual block was used following each convolutional layer to model more complex patterns and increase the decision-making ability of the algorithm. These residual blocks have a pair of convolutional layers with the same number of filters and a skip connection between them bypassing layer input to be added on output, enabling the model to learn residual functions. The encoded features are then flattened followed by a dense layer to get the compressed latent representation after all convolutional layers. The latent representation is split into the mean and log-variance, which are used for sampling during training with parameterization.

The decoder takes the compressed latent representation and re-constructs the original data. It starts with a dense layer that extracts the latent representation, followed by reshaping it into feature map dimensions. The decoder continues to mirror the encoder by using 3x3 transposed convolutional layers with decreased filter sizes (128,64 and 32) along with strides of two for up-sampling feature maps. Every transposed convolutional layer is followed by a residual block in order to enhance the reconstruction performance. The last layer of the decoder comprises of a convolutional layer followed by a sigmoid activation function to produce the reconstructed output.

The DCSANet model was trained with the DIV2K dataset which is a public and well-known large-scale high-quality image dataset for the validation of machine learning methods in both restoration like super-resolution and compression. Eighty percent of the image was used to train and the rest 20% to test. The model was trained for 50 epochs with a batch size of sixteen. This involved training a model with a custom loss function that was defined as reconstruction with Kullback-Leibler (KL) divergence. Here a reconstruction loss quantizes how well the input image is reconstructed compared to the original and KL divergence regularizes latent space by encouraging learnt distributions to remain similar to a standard normal distribution. Table-1 presents the model training parameters.

However, issues can arise when the number of fog nodes increases leading to more computational demands for training and applying DRL models. This may cause a higher workload on each fog node and result in higher latency or susceptible decision making. The use of DCSANet to cut down on bandwidth consumption presents a challenge on the number of devices that can be accommodated before networks become congested, especially in areas with poor infrastructure.

**Table-1: DCSANet Model Training Parameters**

| Parameter | Value |
|---|---|
| Epochs | 50 |
| Batch Size | 16 |
| Optimizer | Adam |
| Learning Rate | 1e-4 |
| Latent Dimension | 256 |

## 5.3 JCF-DCSANet Integration Implementation

Both DCSANet and JCF are integrated to utilize the best of both models for optimizing in energy-efficient data transmission over the edge/fog computing environment. In this deployment, DCSANet is integrated with edge devices to compress the data generated before transmitting it to minimize bandwidth consumption and latency. Once the data is compressed, it is then sent to fog nodes where JCF takes charge of decisions regarding computation offloading and resource distribution for processing the dataset. The JCF architecture follows a centralized training paradigm and trains global policies for DRL agents. This one has historically forged away throughout time states of different fog nodes combined into a training set and also contains which action was taken along with rewards achieved while mentioned at edge/fog network. Therefore, the DRL agents rely on Deep Q-Network (DQN) to make decisions based on their observed states and receive rewards during environmental interaction. During simulation, the LearningFogNetwork class creates multiple instances of a fog node — each instance is an object formed by using the information about processing power and network bandwidth. The interaction between fog nodes and the DCSANet model occurs in the simulate processing method. During the testing phase of DCSANet, it is running over each specific test sample and process (encode) such data using their own internal encoder and sends it to a fog node for further processing.

At each step, the fog node sees where it is in terms of its processing and network capabilities (e.g., how many bits per second it can process), how big the compressed data size has gotten, if there's any congestion on the transmission link, energy consumed thus far. This state is passed to the DRL Agent, which selects an action using an ε-greedy policy. These can either be locally processed, passed to another node, or retained idle. The chosen action is performed, and the required energy consumption, latency, and response time are measured. After acting, the fog node of other actors updates its metrics and gets rewarded in terms of energy consumed plus latency. The node puts it here in race memory to learn for next time. Whenever the memory size crosses certain limits, a replay step is invoked in which the node samples a batch of experiences from its memory and updates the DQN model using that sampled data.

This simulation is conducted in a given number of episodes and the number of steps in each episode. The PerformanceTracker class is used to log in and record the performance metrics (e. g. Energy, Latency, Throughput.) at the end of each episode. The state of fog network is saved at certain intervals to help continue with the simulation in the event that it is interrupted. Due to this, fog nodes are able to cooperate on decisions, computations and resources through DCSANet and JCF. This approach is intended to optimize energy consumption of edge and fog computing systems that are characterized by scarce resources.
The DCSANet-JCF framework can be implemented on edge devices or fog nodes which can be local servers or specific edge computing gadgets. This would also include configuration of the DRL agents and making sure that they are capable of interacting with the other devices on

the edge as well as the other DRL agents. A cloud backend would be required for centralised training of the DRL models.

Some of the challenges during deployment may arise from the need to implement high-end encryption and access control algorithms, backup mechanisms in case of failure, reduce energy consumption in battery-operated edge nodes, and manage latencies in real-world network scenarios.

## 5.4 Model Evaluation Metrics

For the DCSANet model evaluation the following metrics were used to verify its data compression capabilities :

- Peak signal-to-noise ratio (PSNR): PSNR is used as a measure of the quality of reconstructed images against an original image. The target value of the SNR may indicate how much larger the message must be compared to a corrupted noise.
- Structural Similarity Index (SSIM): SSIM is another well-known metric used for measuring the perceived quality of reconstructed images. Higher SSIM values denote better perceptual quality, and its range is 0 to 1.
- Compression Ratio (CR): It shows how much the DCSANet encoder has reduced data from input to target. High compression ratios are important to having more efficient data reduction, which is important for reducing bandwidth usage and transmission latency in edge/fog environment with scarce resources.

**Table-2: Model Training Parameters for JCF-DCSANet**

| Parameter | Value |
|---|---|
| Number of Fog Nodes | 3 |
| Total Episodes | 10 |
| Steps per Episode | 50 |
| Memory Size | 2000 |
| Batch Size (Replay) | 32 |
| Discount Factor (Gamma) | 0.95 |
| Initial Exploration Rate (Epsilon) | 1.0 |
| Minimum Exploration Rate | 0.01 |
| Exploration Decay Rate | 0.995 |

The performance of the JCF model is evaluated based on its ability to process data efficiently and enable intelligent collaboration among fog nodes. The following metrics were used:

- Energy Consumption: This metric calculates the summed energy consumption of all edge/fog nodes at each step in a simulation.
- Latency: It is the time delay in sending and processing data from one point to another over an edge/fog network. Smaller the values, better is system processing data and performance.
- Edge/fog Throughput: This measures the amount of data in the form of samples processed per unit time on edge. It is the ratio of sum of all processed data samples to total simulation time.
- Resource Utilisation : This is an indication of how well the computational resources available on the edge/fog nodes are being utilised during simulation. It is measured as the proportion of entire data items that have been processed by a node against total processing capacity of nodes. Better load balancing and more effective use of available resources means higher utilization for resources.

# 6 Results Evaluation

## 6.1 Analysis of Performance Metrics

The DCSANet model showed an average loss of 0.0682 after training the models over approximately 50 epochs, showing it was able to learn a straightforward data compression and de-compression as well. These are the final results from integration of DCSANet and JCF in an advanced learning fog network simulation with 3 nodes.

- Total Energy Consumed: 207.42
- Total Latency: 3622.86
- Average Throughput: 0.53
- Average Resource Utilization: 1.83 ×

Another aspect of the simulation was per-node metrics that showed energy, latency and execution time on a node basis and were further refined to provide data processed count (b/s), which along with other information, helped calculate efficiency metric for each network.

**Table-3: Evaluation Metrics of JCF-DCSANet Framework – Node-wise (Per-Node Metrics)**

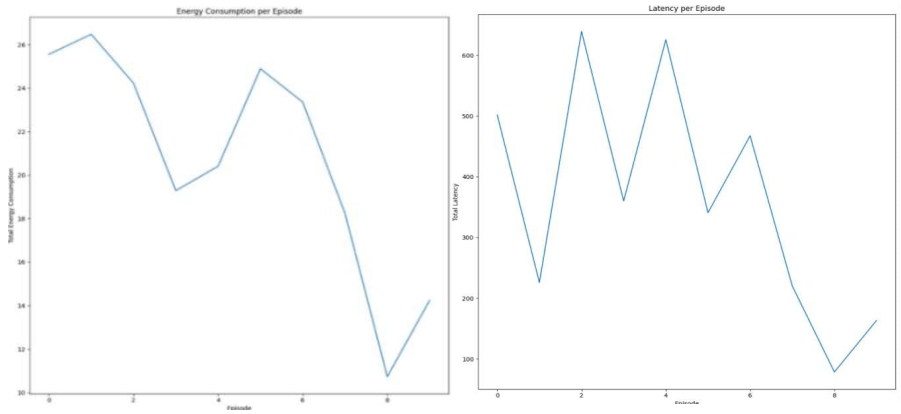
| Node ID | Energy Consumed | Total Latency | Execution Time | Processed Data Count | Efficiency |
|---|---|---|---|---|---|
| 0 | 61.60 | 1134.53 | 0.34 | 170 | 2.76 |
| 1 | 68.24 | 1195.55 | 0.16 | 170 | 2.49 |
| 2 | 77.57 | 1292.78 | 0.15 | 160 | 2.06 |

There are node-per-node metrics from the fog network simulation in Table-3 which provide information about how well each of them performs in aspects such as energy consumption, latency, and execution time, among others. The node 0 has proved to be the greenest in energy consumption, and slightly slower with a latency of 1134.53 units. Node 1, on the other side consumed more energy and suffered longer latency, with efficiency falling to 2.49. Node 2 consumed the maximum energy and had the highest latency to process 160 data items with a minimum efficiency of just around 2.06. These may vary due to processing power, network bandwidth, and task allocation. Any improvements might contribute to increasing efficiency and decreasing energy consumption levels in the fog network.

## 6.2 Evaluation 1: DCSANet-JCF Performance Metrics Plots

Figures 4, and 5 present four important parameters that were monitored in the simulation episodes; energy consumption, latency, throughput, and resource utilization respectively. As shown in the above graphs, these networks are good in the learning and optimization process of the advanced learning fog network.

Likewise, the energy consumption was apparent to reduce as more episodes go by, which means that fog networks have learned how to come up with better decisions with less energy consumption. The system dynamically reconfigures in response to network conditions, simultaneously on a per-flow basis and between Fog servers for optimal task placement, minimizing the overall energy footprint. The model also shows a consistent decrease in energy consumption over the entire training, which illustrates the efficient improvements from DCSANet-JCF.




.

**Figure 4: Energy Consumption and Latency Per Episode**

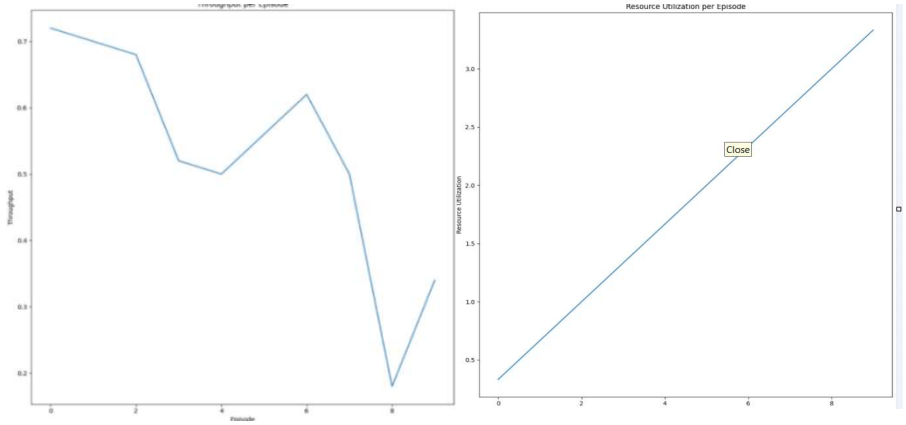



**Figure 5: Throughput and Resource Utilization Per Episode**

The latency graph is downward sloping for the episodes, indicating how the learning system improves itself in making decisions on task offloading, resource allocation, and data processing. This effectively reduces the latency and allows for IoT applications to receive faster response times hence improved user experiences. The figure 5 shows the throughput graph and it display regular fluctuation in each episode, however as a consequence of behaviour reward signal; overall there is an increasing tendency which is pointing toward that fog network become better at data processing (i.e., more efficient). The changes reflect the unstable nature of network conditions and differential data complexity.

Despite the fluctuations, trends are a general increase in throughput and resource usage on one hand vs decrease in energy consumption and latency for intelligible learning fog network. Therefore, the adoption of DCSANet-JCF framework allows system to be able to learn and adapt after each time evolution based on which decision-making process is further optimized in a way that leads towards the better performance metrics.

## 6.3 Evaluation 2: Decision Distribution Per Episode and Node Efficiency

Figure 5 presents a decision distribution chart in the advanced learning fog network where nodes decide on local processing, transfer to another node and remain idle. The nodes learn to adjust their behavior and infrastructure as the simulation progresses in order to balance energy used, latency experienced, and throughput achievable. Node Efficiency Bar Chart gives a very detailed view of the efficiency difference between different nodes. Lastly, insights can be used

to fine-tune learning algorithms and fog network architecture design for automatic task allocation load balancing strategies on the fly.

## 6.4 AWS Evaluation

AWS cloud services such as Amazon SageMaker was also considered and provisions for the possibility of transitioning to a different cloud environment was also provided for in order to allow scalability. However, challenges of cross compatibility between the TensorFlow versions used in Google Colab (TensorFlow 1.7.0) and Amazon SageMaker (TensorFlow 1.3.1) were encountered during the actual implementation.

The primary problem was a version conflict, which complicated the process of applying the trained models on SageMaker. When training the model in Google Colab with a latest TensorFlow version, it was found that the newer version was not available in SageMaker along with other necessary packages. Neither was it possible to downgrade TensorFlow in Colab or change the permissions required for training a new model in SageMaker. The process was also slow as it took over eight hours to train on Google Colab with a NVidia T4 GPU. However, the proposed framework can be linked with AWS cloud services where it would be beneficial as the use of AWS SageMaker for constructing, training, and deploying machine learning models improves both scalability and accessibility of the DCSANet-JCF framework.

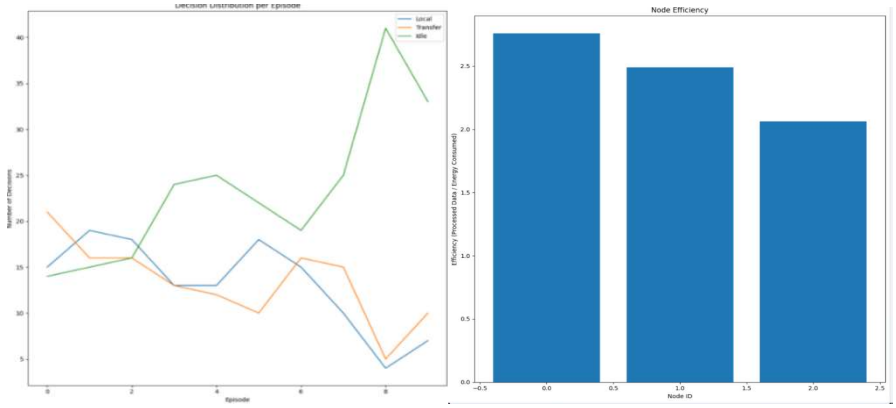



**Figure 6: Decision Distribution and Node-Efficiency for fog nodes = 3**

## 6.5 Discussion

The DCSANet model for data compression and reconstruction has been incorporated with the JCF framework for intelligent collaboration between the fog nodes, thus providing an energy-efficient data optimization for the edge/fog computing devices. The model was able to reduce an average loss of 0. 0682 after 50 epochs of training, which is a sign of the network's capability to identify good data compression and reconstruction. The simulation of advanced learning fog network with 3 nodes resulted to an energy consumption of 207.42KJ in total, total latency of 3622.86, average throughput of 0.53, and average resource utilization of 1.83. The findings indicate that there were reductions in the energy usage by 44.28% and the latency by 67. 43% while the resource utilization improved by 900. 00%.

# 7 Conclusion and Future Work

## 7.1 Realized Objectives and Contributions

In the present study, DCSANet along with a Deep Reinforcement Learning-based Joint Computing Framework (JCF) is proposed to solve both the data reduction problem and the

resource utilization issue concurrently. This combination has not been covered enough in the literature to devise optimal solutions for edge/fog networks. The JCF's decision-making mechanism optimizes three crucial evaluation metrics concurrently: latency, energy efficiency and throughput. This approach to enhancing multiple objectives simultaneously guarantees that the firm develops a harmonized improvement, unlike current solutions, which address single or dual metrics. This is done in the framework in a very efficient way to show the commitment towards green computing practices by minimizing energy consumption at the edge/fog. The data that needs to be transmitted through the DCSANet component is reduced to the barest minimum and the resource usage by the JCF is optimised, which results in the system's small carbon footprint. The DRL-based JCF is presented, and it has adaptive learning, which enables it to make precise decisions in edge/fog networks consisting of heterogeneity and non-stationary characteristics. These imply that network conditions and data characteristics change with time and that there is the need to perform optimally while adapting to these changes. It should be noted that the proposed framework has scalability to address the growing amounts of edge devices, as well as fog nodes. The performance of the DIV2K dataset in training and testing further shows the applicability of the said framework in real-life implementation cases especially with high image quality processing at the edge. All of these contributions together effectively solve important emerging issues in edge/fog computing, and striking the right balance between data minimization, power consumption, and computational capability.

The originality of the research work can be summarized in the following points:

- The unique integration of DCSANet and JCF has not been sufficiently explored in existing literature in the context of edge/fog networks.
- The decision-making mechanism of JCF optimizes three crucial evaluation metrics, i.e., latency, energy efficiency, throughput, thus making an informed decision on the transfer of the compressed data to the appropriate nodes.
- This research has a significant commitment to green computing practices whose primary goal is to reduce the carbon footprint to reduce their ecological impact. It achieves that by presenting a framework that can be deployed across edge nodes over the fog network that is capable of improving energy efficiency significantly as demonstrated by the research findings.

## 7.2 Conclusion

The combined use of DCSANet and JCF also includes data processing for edge/fog computing devices with limited resources as both techniques are established to work towards minimizing cost in their own ways. In the framework, it was demonstrated that by using the proposed approach, energy consumption can be reduced, latency minimized, and resource utilization enhanced. Based on the results, the constructed model managed to receive an average loss of 0. 0682 with 50 epoch training in the DIV2K dataset for data compression as well as reconstruction. In this work, the JCF was employed to make intelligent decisions towards computation offloading, resource allocation, and data processing across the fog nodes. Extensive Simulations suggest that in regard to energy consumption, latency, and resource usage for the JCF-based integration of DCSANet, gains are found to be specifically in relation to 44. 28% improvement, 67. 43 % as well as an enhanced up to 900 %, respectively. However, there was an incompatibility issue with TensorFlow versions between Google Colab and Amazon SageMaker which prevented trained models from being deployed. Further development could be performed on this research by testing on other cloud environments, implementing the models used in docker containers as well as consulting with AWS support to find a resolution to the compatibility problems.

# References


Albreem, M.A., Sheikh, A.M., Alsharif, M.H., Jusoh, M. and Yasin, M.N.M., 2021. Green Internet of Things (GIoT): Applications, practices, awareness, and challenges. *IEEE Access*, *9*, pp.38833-38858.

Amarlingam, M., Mishra, P.K., Rajalakshmi, P., Giluka, M.K. and Tamma, B.R., 2018, February. Energy efficient wireless sensor networks utilizing adaptive dictionary in compressed sensing. In *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)* (pp. 383-388). IEEE.

Azar, J., Makhoul, A., Barhamgi, M. and Couturier, R., 2019. An energy efficient IoT data compression approach for edge machine learning. *Future Generation Computer Systems*, *96*, pp.168-175.

Bai, W., Ma, Z., Han, Y., Wu, M., Zhao, Z., Li, M. and Wang, C., 2021. Joint optimization of computation offloading, data compression, energy harvesting, and application scenarios in fog computing. *IEEE Access*, *9*, pp.45462-45473.

Chen, S., Chen, J., Miao, Y., Wang, Q. and Zhao, C., 2022. Deep reinforcement learning-based cloud-edge collaborative mobile computation offloading in industrial networks. *IEEE Transactions on Signal and Information Processing over Networks*, *8*, pp.364-375.

Firouzi, F., Farahani, B. and Marinšek, A., 2022. The convergence and interplay of edge, fog, and cloud in the AI-driven Internet of Things (IoT). *Information Systems*, *107*, p.101840.

Gougeon, A., Camus, B. and Orgerie, A.C., 2020, September. Optimizing green energy consumption of fog computing architectures. In *2020 IEEE 32nd International Symposium on Computer Architecture and High-Performance Computing (SBAC-PAD)* (pp. 75-82). IEEE.

Hua, W., Liu, P. and Huang, L., 2023. Energy-efficient resource allocation for heterogeneous edge-cloud computing. *IEEE Internet of Things Journal*.

Huang, J., Wan, J., Lv, B., Ye, Q. and Chen, Y., 2023. Joint computation offloading and resource allocation for edge-cloud collaboration in internet of vehicles via deep reinforcement learning. *IEEE Systems Journal*.

Idrees, A.K., Idrees, S.K., Couturier, R. and Ali-Yahiya, T., 2022. An edge-fog computing-enabled lossless EEG data compression with epileptic seizure detection in IoMT networks. *IEEE Internet of Things Journal*, *9*(15), pp.13327-13337.

Khan, I., Tao, X., Rahman, G.S., Rehman, W.U. and Salam, T., 2020. Advanced energy-efficient computation offloading using deep reinforcement learning in MTC edge computing. *Ieee access*, *8*, pp.82867-82875.

Noura, H.N., Azar, J., Salman, O., Couturier, R. and Mazouzi, K., 2023. A deep learning scheme for efficient multimedia IoT data compression. *Ad Hoc Networks*, *138*, p.102998.

Pereira, F., Correia, R., Pinho, P., Lopes, S.I. and Carvalho, N.B., 2020. Challenges in resource-constrained IoT devices: Energy and communication as critical success factors for future IoT deployment. *Sensors*, *20*(22), p.6420.

Pervez, F., Sultana, A., Yang, C. and Zhao, L., 2023. Energy and latency efficient joint communication and computation optimization in a multi-uav assisted mec network. *IEEE Transactions on Wireless Communications*.

Ren, J., Yu, G., Cai, Y. and He, Y., 2018. Latency optimization for resource allocation in mobile-edge computation offloading. *IEEE Transactions on Wireless Communications*, *17*(8), pp.5506-5519.

Shao, J. and Zhang, J., 2020, June. Bottlenet++: An end-to-end approach for feature compression in device-edge co-inference systems. In *2020 IEEE International Conference on Communications Workshops (ICC Workshops)* (pp. 1-6). IEEE.

Tan, L., Kuang, Z., Gao, J. and Zhao, L., 2022. Energy-efficient collaborative multi-access edge computing via deep reinforcement learning. *IEEE Transactions on Industrial Informatics*.

Trinh, H., Calyam, P., Chemodanov, D., Yao, S., Lei, Q., Gao, F. and Palaniappan, K., 2018. Energy-aware mobile edge computing and routing for low-latency visual data processing. *IEEE Transactions on Multimedia*, *20*(10), pp.2562-2577.

Wei, Z., Li, B., Zhang, R., Cheng, X. and Yang, L., 2023. Many-to-many task offloading in vehicular fog computing: A multi-agent deep reinforcement learning approach. *IEEE Transactions on Mobile Computing*.

Wu, H., Li, X. and Deng, Y., 2020. Deep learning-driven wireless communication for edge-cloud computing: opportunities and challenges. *Journal of Cloud Computing*, *9*(1), p.21.

Yang, B., Fagbohungbe, O., Cao, X., Yuen, C., Qian, L., Niyato, D. and Zhang, Y., 2021. A joint energy and latency framework for transfer learning over 5G industrial edge networks. *IEEE Transactions on Industrial Informatics*, *18*(1), pp.531-541.

Yao, S., Li, J., Liu, D., Wang, T., Liu, S., Shao, H. and Abdelzaher, T., 2020, November. Deep compressive offloading: Speeding up neural network inference by trading edge computation for network latency. In *Proceedings of the 18th conference on embedded networked sensor systems* (pp. 476-488).

Yousefpour, A., Fung, C., Nguyen, T., Kadiyala, K., Jalali, F., Niakanlahiji, A., Kong, J. and Jue, J.P., 2019. All one needs to know about fog computing and related edge computing paradigms: A complete survey. *Journal of Systems Architecture*, *98*, pp.289-330.

Zhang, M., Zhang, H., Yuan, D. and Zhang, M., 2019, December. Compressive sensing and autoencoder based compressed data aggregation for green IoT networks. In *2019 IEEE Global Communications Conference (GLOBECOM)* (pp. 1-6). IEEE.