A DISSERTATION
SUBMITTED IN FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
IN COMPUTER SCIENCE AND ENGINEERING

Scaling Pneumonia Detection Inference Based on Reconfigurable AI-Enabled System

by

Jiangkun Wang

September 2023

The thesis titled

Scaling Pneumonia Detection Inference Based on Reconfigurable AI-Enabled System

by

JIANGKUN WANG

is reviewed and approved by:

Chief referee

Professor	Date 31/07/2023	
Abderazek Ben Abdallah	Ben Abdallah Abderazek	[illegible]
Professor	Date 22/8/2023	
Hiroshi Saito	SAITO Hiroshi	齋藤
Senior Associate Professor	Date 22/8/2023	
Yuichi Okuyama	[illegible]	[illegible]
Associate Professor	Date 22/08/2023	
Daisuke Suzuki	Daisuke ~~[illegible]~~ Suzuki	鈴木
Associate Professor	Date 22/08/2023	
Khanh Nam Dang	DANG Nam Khanh	ダン

THE UNIVERSITY OF AIZU

2023

DECLARATION: PLAGARISM

I, Jiangkun Wang, declare that:

1. The research reported in this dissertation, except where otherwise indicated, is my original research.
2. This dissertation has not been submitted for any degree or examination at any other university.
3. This dissertation does not contain any other persons' data, pictures, raphs or other information, unless specifically acknowledged as being sourced from other persons.
4. This dissertation does not contain other persons' writing, unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then:
    a. Their words have been re-written but the general information attributed tot them has been referenced
    b. Where their exact words have been used, then their writing has been placed in italics and inside quotation marks, and referenced.
5. This dissertation does not contain text, graphics or tables copied and pasted from the Internet, unless specifically acknowledged, and the source being detailed in the thesis and in the References sections.

Signed: Jiangkun Wang
Date: May 20, 2023

Contents

List of Figures

List of Tables

List of Abbreviations

AI	Artificial Intelligence
AIRBiS	AI-Enabled Real-Time Bio-System
ANN	Artificial Neural Network
AXI	Advanced eXtensible Interface
BRAM	Block Random-Access Memory
CADx	Computer-Aided Diagnosis
CLB	Configurable Logic Block
CNN	Convolutional Neural Network
COVID-19	Coronavirus Disease 2019
CPU	Central Processing Unit
DL	Deep Learning
DMA	Direct Memory Access
DNN	Deep Neural Network
DSP	Digital Signal Processing
FF	Flip-Flop
FLOPs	Floating-Point Operations
FPGA	Field Programmable Gate Array
GOPS	Giga Operations Per Second
GPU	Graphic Processing Unit
GUI	Graphical User Interface
IID	Independent and Identically Distributed
IoT	Internet of Things

LUT	Look-Up Tables
MACC	Multiply-Accumulate Operations
PL	Programmable Logic
PS	Processing System
RAM	Random Access Memory
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
COVID-19	Coronavirus Disease 2019
CXR	Chest X-ray
FF	Flip Flop
FPGA	Field-Programmable Gate Array
FT-3DR	Fault-Tolerant 3-Dimensional Router
FTSP	Fault-Tolerant Shortest Path
KMCR	K-means-based Multicast Routing Algorithm
NoC	Network-on-Chip
SNN	Spiking Neural Network
SNPC	Spiking Neuron Processing Core
WCET	Worst-Case Execution Time

List of Symbols

$Input_{width}$	Width of the input image
$Input_{height}$	Height of the input image
$Input_{channel}$	Number of channels in the input image (e.g., 3 for RGB images)
N_{conv}	Number of convolutional layers
K_i	Number of filters in the i^{th} convolutional layer
F_i	Size (width/height) of filters in the i^{th} convolutional layer
$Stride_i$	Stride size for the i^{th} convolutional layer, determining the step size for sliding the filter across the input
P_i	Pool size for the i^{th} max pooling layer, indicating the dimension of the square region over which the maximum value is taken
N_f	Size of the 1D vector obtained after flattening the output of the last pooling layer
M	Number of fully connected layers
N_j	Number of neurons in the j^{th} fully connected layer
TP	True positive
FP	False positive
TN	True negative
FN	False negative
N_{client}	Total number of client nodes in the network
i	A specific client node, where $1 \leq i \leq N_{client}$
M_i	Updated model of client i after a round of collaborative learning using its local dataset

P_{attack}	Percentage of poisoning attacks in the network. This indicates that P_{attack} percent of clients are malicious and provide manipulated models to the server
M_i	Model update of client i after training
M_g	Global model from the last round of collaborative learning
M_{server}	Server model
M^i_{local}	Local model of client i
N_{client}	Total number of clients
α	Fraction parameter indicating the percentage of models to be selected
M_{selected}	List containing the selected local models for aggregation
P_{server}	One-dimensional array storing all parameters of the server model
P^i_{local}	One-dimensional array storing all parameters of the local model M^i_{local}
Sim_i	Minkowski similarity between the model update M_i of client i and the global model M_g from the last round. This score reflects the quality of the local model
$Sim_{\text{server},i}$	Similarity between the server model and the local model of client i using the Minkowski distance
$Sim_{i,j}$	Similarity measure between the parameters of local models P^i_{local} and P^j_{local}, using the Minkowski distance
$V^l_j(t)$	Membrane potential of a LIF neuron j in layer l at a time-step t
$V^l_j(t-1)$	Membrane potential of a LIF neuron j in layer l at the previous time-step $t-1$
w_{ij}	Synaptic weight from neuron i to j
λ	Leak current value that causes decay in the membrane potential
$x^{l-1}_i(t-1)$	Pre-synaptic spike from the previous layer $l-1$ at the previous time-step $t-1$
θ	Threshold value for neuron spiking
s	Output spike; equals 1 if a spike is released and 0 otherwise
XYZ_s	Source node address
N	Set of all clients

Acknowledgment

I would like to express my utmost gratitude to my supervisor, Prof. Abderazek Ben Abdallah, for his unwavering patience, support, encouragement, and guidance throughout my Ph.D. program. His vast knowledge, invaluable advice, and meticulous instructions have contributed significantly to my personal and academic growth. I am also deeply grateful to Prof. Hiroshi Saito, Prof. Yuichi Okuyama, Prof. Daisuke Suzuki, and Prof. Khanh Nam Dang for their generous contributions to revising my thesis. Their thoughtful and detailed feedback played a crucial role in completing this work.

Special thanks to Dr. Mark and Dr. Wang for taking the time to discuss my research and answer my questions. I am also grateful to the other members of the Ben Abdallah & Dang Laboratory and my friends at the University of Aizu for their encouraging words and motivational messages. Their unwavering support has motivated me to work harder and become a better researcher and person. Moreover, I appreciate the entire staff of the University of Aizu for their support and help in making my study experience and settling into Aizuwakamatsu seamless.

Lastly, I would like to express my infinite love to my parents and family for their endless support and unconditional love. Their guidance and prayers have inspired me to aim for greater heights. I am also grateful to my siblings, mentors, and friends for their support and kind words that have helped me through difficult times.

Jiangkun Wang,

September 2023,

Aizuwakamatsu, Japan

Scaling Pneumonia Detection Inference Based on Reconfigurable AI-Enabled System

Abstract

The success of deep learning in expanding the boundaries of artificial intelligence has led to the widespread use of AI-enabled systems in addressing various challenges in different fields. In healthcare, edge computing platforms are commonly used for deep learning to address security and latency challenges despite these platforms often being resource-constrained. However, conventional artificial neural networks used in deep learning systems are computationally complex, require high power, and have low energy efficiency, making them unsuitable for edge computing platforms. In critical applications such as bio-medicine, the reliability of deep learning systems must be considered when designing them.

To address these challenges, we propose a reconfigurable 3D-NoC-based neuromorphic system for biomedical applications based on a fault-tolerant spike routing scheme. By combining the spatio-temporal nature of information processing of spiking neural networks with fault-tolerant 3D-NoC hardware, our system achieves excellent multi-objective performance accuracy while maintaining low latency and low power consumption. Our performance evaluation results over X-ray images for pneumonia detection show that our proposed system achieves 88.43% detection accuracy over the collected test data and could be accelerated to achieve 4.6% better inference latency than the conventional artificial neural network-based system while consuming 32% less power. Moreover, our proposed system maintains high accuracy for up to 30% inter-neuron communication faults with increased latency.

In addition to healthcare, AI systems are being used in various fields to address different challenges. The demands of AI applications such as deep learning, particularly in machine learning, have led to the need for specific technical architectures to meet the challenges of cloud computing. In healthcare, the ongoing efforts to combat the COVID-19 pandemic have increased the use of diagnosis and detection systems based on edge computing platforms, which are resource-constrained. To achieve high-performance computing and deep learning inference for pneumonia detection in chest X-ray images, an efficient edge computing platform is required.

Our work presents methods and architectures for scaling deep learning inference for pneumonia detection based on a reconfigurable FPGA cluster platform (AIRBiS). Thanks to our proposed hardware design, we were able to optimize and satisfy design constraints such as low latency and high energy efficiency. The performance evaluation results show that the proposed AIRBiS system achieves 95.2% detection accuracy of pneumonia over the collected test data with the computer-aided diagnosis scenario. Furthermore, for rapid batch detection, the detection could be accelerated by 0.023 s. The system inference acceleration is 13 times more energy-efficient than conventional approaches on average.

AI に基づいた再構成可能なシステムによるスケーリング可能な肺炎検出推論

概要

人工知能の境界を広げるディープラーニングの成功により、さまざまな分野のさまざまな課題に対処するために AI 対応システムが広く使用されるようになりました。ヘルスケアでは、エッジコンピューティングプラットフォームは、多くの場合、リソースに制約があるにもかかわらず、セキュリティとレイテンシの問題に対処するためにディープラーニングに使用されます。しかし、深層学習システムで使用される従来の人工ニューラルネットワークは、計算が複雑で、大電力を必要とし、エネルギー効率が低いため、エッジコンピューティングプラットフォームには適していません。生物医学などの重要なアプリケーションでは、設計時にディープラーニングシステムの信頼性を考慮する必要があります。

これらの課題に対処するために、フォールトトレラントスパイクルーティングスキームに基づく生物医学アプリケーション用の再構成可能な 3D-NoC ベースのニューロモルフィックシステムを提案します。スパイキングニューラルネットワークの情報処理の時空間的性質をフォールトトレラントな 3D-NoC ハードウェアと組み合わせることで、システムは低レイテンシと低消費電力を維持しながら、優れた多目的パフォーマンス精度を実現します。肺炎検出のための X 線画像に対するパフォーマンス評価の結果は、提案されたシステムが収集されたテストデータに対して 88.43% の検出精度を達成し、従来の人工ニューラルネットワークベースのシステムよりも 4.6% 優れた推論

レイテンシーを達成するために加速できることを示しています。消費電力が 32% 減少します。さらに、提案されたシステムは、最大 30% のニューロン間通信障害に対して高い精度を維持し、遅延が増加します。

ヘルスケアに加えて、AI システムはさまざまな分野でさまざまな課題に対処するために使用されています。特に機械学習におけるディープラーニングなどの AI アプリケーションの要求により、クラウドコンピューティングの課題に対応するための特定の技術アーキテクチャが必要になっています。ヘルスケアでは、COVID-19 パンデミックと闘うための継続的な取り組みにより、リソースに制約のあるエッジコンピューティングプラットフォームに基づく診断および検出システムの使用が増加しています。胸部 X 線画像で肺炎を検出するための高性能コンピューティングとディープラーニング推論を実現するには、効率的なエッジコンピューティングプラットフォームが必要です。

私たちの仕事は、再構成可能な FPGA クラスタープラットフォーム (AIRBiS) に基づいて、肺炎検出のためのディープラーニング推論をスケーリングするための方法とアーキテクチャを提示します。提案されたハードウェア設計のおかげで、低レイテンシーや高エネルギー効率などの設計上の制約を最適化して満たすことができました。性能評価結果は、提案された AIRBiS システムが、コンピュータ支援診断シナリオで収集されたテストデータに対して 95.2% の肺炎の検出精度を達成することを示しています。さらに、迅速なバッチ検出の場合、検出を 0.023 秒加速できます。システム推論の加速は、平均して従来のアプローチよりも 13 倍エネルギー効率が高くなります。

Chapter 1

Introduction

In the evolving landscape of medical imaging and artificial intelligence, understanding the intricacies and addressing prevailing challenges has become imperative. This chapter delineates the research background that anchors this study, presenting the motivation that spurred this deep dive into scaling pneumonia detection inference on reconfigurable AI-enabled systems. Three pivotal contributions of this dissertation are highlighted: a privacy-preserving method using the Client Selection Secure Collaborative Learning (CSSCL) algorithm, an energy-efficient Reconfigurable system (AIRBiS) leveraging an FPGA cluster for pneumonia detection in chest X-ray images, and an Event-Driven energy-efficient inference method based on a fault-tolerant spike routing mechanism. Recognizing the imperfections, the limitations of this work are also candidly addressed. Conclusively, an overview of the dissertation structure and content is provided to navigate the reader through the ensuing chapters.

1.1 Background

As the contemporary era heralds a transformative shift towards digitalization, Artificial Intelligence (AI) stands at the forefront, revolutionizing myriad sectors. Central to this narrative is the proliferation of AI in the biomedical domain, an arena where the confluence of AI and healthcare promises heightened diagnostic capabilities and improved patient outcomes. This section delves into the foundational principles of AI, emphasizing its integration into biomedical systems. A focused exploration on Deep Neural Networks offers insights into their intricate workings and applications. Further, the narrative contrasts the paradigms of cloud-based and edge-based biomedical systems, delineating their respective advantages and challenges. Concluding this background elucidation, the chapter homes in on the specific arena of AI-based pneumonia detection, illustrating the pivotal role AI plays in transforming medical imaging diagnostics.

Artificial Intelligence (AI)

Artificial Intelligence technology has been applied to several fields, such as speech recognition, natural language processing, and biomedicine. In biomedicine, particularly, the advancements in AI-enabled by deep learning are transforming the potentials of medical research, disease detection and diagnosis, and medical record analysis. By integrating AI in biomedicine, AI-based biomedical systems, such as those in medical imaging, enable the detection of patterns and anomalies in medical images, improving the accuracy of disease detection and diagnosis.

AI-based Biomedical Systems

Traditionally, AI-based biomedical systems are often deployed on the cloud due to the significant computing requirements (Power, cost, computing capability) of the deep learning models on which they are built. These deep learning models are based on conventional artificial neural networks (ANN), which have demonstrated impressive performance in biomedical applications. Their increasingly complex architecture and sizes (Deeper Neural Networks) over the years have led to a significant increase in their computational complexity and power consumption which has become a concern.

Deep Neural Networks

With abundant layers that rack up trillions of parameters [20], the voracious appetite for the computing power of conventional deep learning models has become a concern. As an alternative approach, a spiking neural network (SNN) holds great potential for addressing this challenge.

Cloud-Based Biomedical Systems

The cloud computing platform on which some AI-based biomedical systems are deployed provides excellent performance. However, they are faced with availability, security, privacy, and latency concerns which critical time-driven biomedical systems cannot compromise on.

Edge-Based Biomedical Systems

To address some of the challenges of the cloud-based platform, accelerated edge-based platforms have gained attention as an appealing solution to the availability, security, privacy, and latency concerns of cloud-based platforms. Edge devices, such as IoT devices, have been shown great potential for biomedical data processing and analysis, e.g., pneumonia detection.

AI-based Pneumonia Detection

The pneumonia pandemic has been progressively tackled using deep learning (DL), among other approaches for detection/diagnosis. This detection approach is carried out by analyzing chest X-rays and CT images of infected persons using deep learning techniques to detect viral pneumonia. Most proposed pneumonia detection/diagnosis systems employ conventional artificial neural networks (ANNs), which require huge amounts of computing power and energy due to their computational complexity. Consequently, they are a less suitable deep learning approach for pneumonia detection/diagnosis and monitoring on the edge. As an alternative, the Spiking Neural Network (SNN), which has been shown to be less computationally complex with energy-efficient event-driven computation, can be employed with a neuromorphic approach to achieve energy-efficient real-time pneumonia detection/diagnosis on edge.

1.2 Research Motivation

In the labyrinth of modern technological advancements, the imperative for safeguarding data privacy and ensuring robust security, especially in healthcare applications, has never been more pronounced. The migration of computational tasks to the edge promises efficiency and real-time processing yet concurrently introduces unique challenges in terms of scalability. Furthermore, the potent synergy between AI and medical diagnostics, particularly in pneumonia detection, beckons enhanced methodologies that amalgamate accuracy with efficiency. This section delineates the driving forces behind this research, emphasizing the imperatives of privacy preservation, the need for a scalable edge-based platform, and the pursuit of refining AI-based pneumonia detection methods.

Privacy-preserving and Security

As shown in Figure 1.1, AI-based biomedical systems require the medical images to be shared with the cloud server, which raises privacy concerns. Also, as these medical images will be shared over the network, the possibility of data leakage becomes a security concern.

Scaling Edge-based Platform

With biomedical systems increasingly being deployed on the edge, the processing of time-driven application data has been accelerated to achieve fast decision-making in a secure, reliable, and energy-efficient manner. Traditional edge computing platforms are often resource-constrained, with limited processing power,

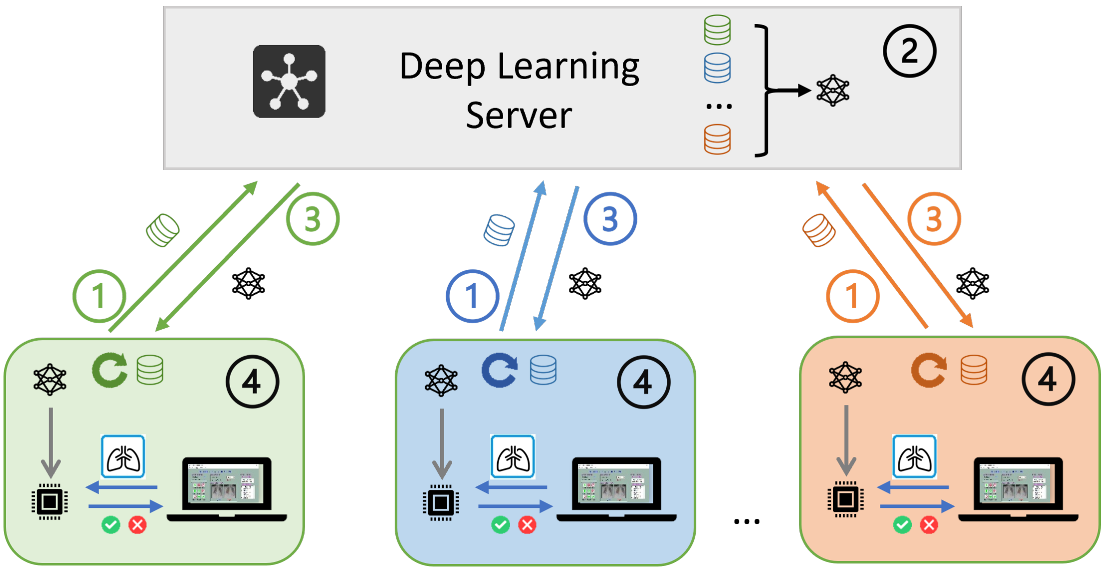
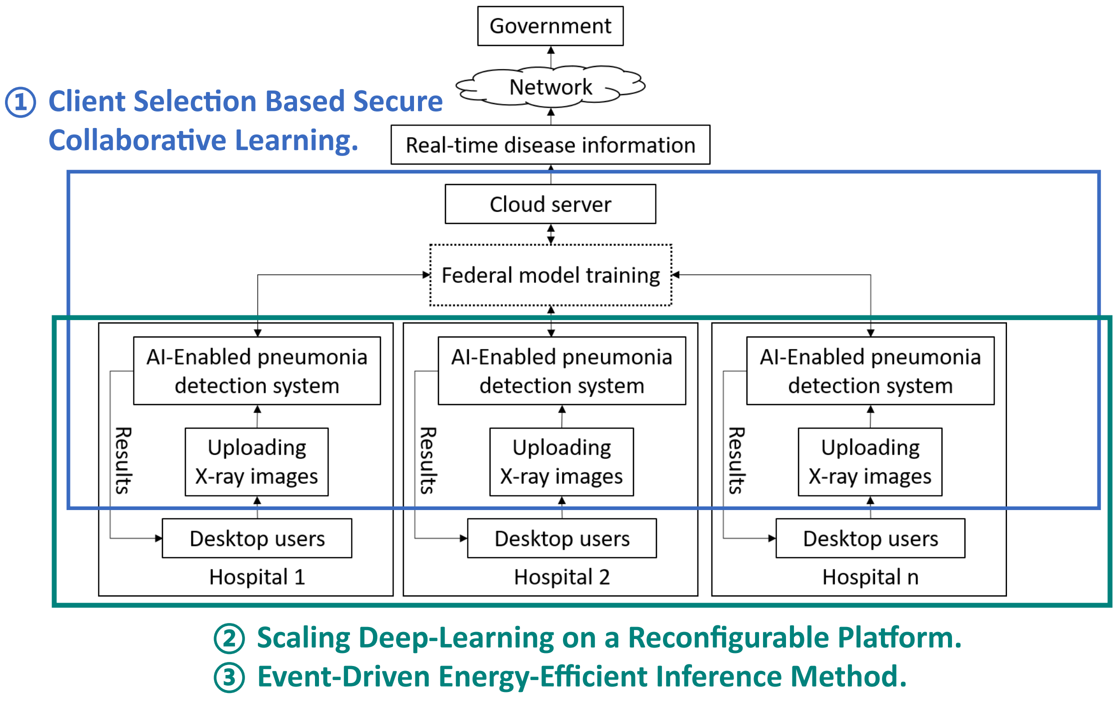


Figure 1.1: Conventional deep learning training approach for pneumonia detection. AI-based biomedical systems require medical images to be shared with the cloud server, which raises privacy concerns. Also, as these medical images will be shared over the network, the possibility of data leakage becomes a security concern.

memory, and scalability. Therefore, processing high-precision biomedical data like lung X-ray images required to detect viral pneumonia has become a challenge. In such scenarios, addressing the need to scale edge-based AI-enabled platforms has become imperative.

AI-based Pneumonia Detection Method

Most proposed pneumonia detection/diagnosis systems employ conventional artificial neural networks (ANNs). Unfortunately, conventional ANNs bring a major burden in terms of computing power and energy consumption due to their complexity. As a result, they may not be the most suitable deep learning approach for pneumonia detection/diagnosis and monitoring at the edge. To address this issue, alternative approaches based on spiking neural network computing algorithms should be explored. Spike-based computing algorithms can reduce the complexity, computing power, and energy consumption of ANNs significantly. Furthermore, they can provide a more robust and efficient solution for edge-based pneumonia detection and monitoring. Thus, spike-based pneumonia detection should be explored as an alternative to conventional ANNs on the edge.



Government
Network
① Client Selection Based Secure Collaborative Learning.
Real-time disease information
Cloud server
Federal model training
AI-Enabled pneumonia detection system
Results
Uploading X-ray images
Desktop users
Hospital 1
Hospital 2
Hospital n
② Scaling Deep-Learning on a Reconfigurable Platform.
③ Event-Driven Energy-Efficient Inference Method.

Figure 1.2: The Proposed System Organization. The AI-enabled COVID-19 detection system is provided to each client hospital. Desktop users upload X-ray images to the system using the user interface and get results. The statistics result is gathered for the government via the network automatically.

1.3 Contribution

This dissertation presents algorithms and architecture for scaling pneumonia detection inference on reconfigurable AI-enabled systems, including three significant contributions:

- A robust privacy-preserving method based on a Client Selection Secure Collaborative Learning (CSSCL) algorithm. This method is based on a distributed learning architecture, which enables only the weights of a shared deep learning model to be communicated without compromising the privacy of the data. The architecture ensures that data is not shared between different data owners but processed locally.

- A self-contained energy-efficient Reconfigurable system (AIRBiS) by leveraging an FPGA cluster for pneumonia detection in chest X-ray images. The FPGA cluster helps to parallelize the inference process of X-ray images to achieve fast and accurate pneumonia detection.

- Event-driven energy-efficient inference method based on fault-tolerant spike routing mechanism. The event-driven energy-efficient reconfigurable neuromorphic biomedical system is based on a fault-tolerant scalable 3D-NoC-based neuromorphic processor. Its components include convolution cores, spiking neuron processing cores (SNPCs), and a fault-tolerant 3-dimensional router (FT-3DR) arranged in 2D mesh topologies and stacked to form a 3D mesh architecture. The detection method achieves less power consumption while requiring less time for inference with minor classification accuracy degradation than the ANN-based approach.

The AIRBiS system integrates three pivotal contributions. To achieve security and privacy-preserving, the first contribution, a robust privacy-preserving method based on a Client Selection Secure Collaborative Learning (CSSCL) algorithm, is proposed. However, the computing capability on the edge side is still limited. To address this challenge, the second contribution is raised: a self-contained energy-efficient Reconfigurable system (AIRBiS) by leveraging an FPGA cluster for pneumonia detection in chest X-ray images. So far, I have trained the AIRBiS-Net model with collaborative learning and deployed it to the scaling reconfigurable FPGA cluster. To further reduce power consumption and improve inference speed, an Event-Driven Energy-Efficient Inference Method Based on a Fault-Tolerant Spike Routing Mechanism is presented. Collectively, as illustrated in Figure 1.2, these three contributions transform AIRBiS into a cohesive platform characterized by enhanced accuracy, heightened security, superior adaptability, and energy efficiency.

1.4 Limitations of This Research

This dissertation presents the design and implementation of a scalable pneumonia detection inference system, utilizing a reconfigurable AI-enabled system that exploits the power of secure collaborative learning and FPGA cluster-based reconfigurable systems. While the system has demonstrated promising outcomes, several limitations and opportunities for future exploration are noted.

The first limitation resides in the robust privacy-preserving method based on a client-selection secure collaborative learning algorithm. While the semi-decentralized fault-tolerant architecture proposed here avoids some security issues associated with centralized aggregation nodes and demonstrates higher efficiency than a fully decentralized blockchain-based scheme, it neglects the exploration of emerging sharding blockchain technology. As the sharding approach in blockchain could possibly enhance the efficiency of the system, future work may explore adaptive system architectures that would automatically adjust the degree of decentralization in response to the application's security level and system scale, striking an optimal balance between efficiency and security.

The second constraint pertains to the implementation of the self-contained energy-efficient reconfigurable system by leveraging an FPGA cluster for pneumonia detection in chest X-ray images. Despite its advantages in providing high energy utilization efficiency and adaptability to task-specific requirements, the high cost of the utilized Zynq UltraScale+ MPSoC ZCU102 Evaluation Kit presents a significant drawback. This device, while suitable for rapid verification, is not ideal for direct deployment in practical scenarios due to its expense. Future work should focus on customizing hardware resources based on task requirements to reduce costs when deploying the system in actual use cases.

The final limitation concerns the proposed pneumonia detection solution based on the Spiking Neural Network (SNN) and 3D-NoC-based neuromorphic processor. The event-driven energy-efficient reconfigurable neuromorphic biomedical system, though more power-efficient and faster in inference time with minor degradation in classification accuracy compared to the Artificial Neural Network (ANN)-based approach, is not without its drawbacks. Currently, the training efficiency of SNN models lags behind that of ANNs, necessitating the development of novel SNN training methods to further enhance detection accuracy and training speed.

In conclusion, while this dissertation has laid substantial groundwork in advancing the scalability of pneumonia detection inference using reconfigurable AI-enabled systems, it presents several limitations that indicate possible directions for future research and development. Future work will need to address the identified limitations to further enhance the efficiency, security, cost-effectiveness, and training speed of the proposed system.

1.5 Dissertation Overview

The rest of the dissertation is organized as follows:

- Chapter 2 covered the technical aspects of deep-learning algorithms, architectures, inference, neuromorphic computing, and hardware acceleration, introducing essential optimization methods and performance metrics for efficient deep-learning system design and deployment.

- Chapter 3 delved into various approaches for computer-aided pneumonia detection, spanning software to hardware systems. The COVID-19 pandemic highlighted the importance of these advancements. Emphasizing deep-learning inference acceleration and scaling research, we glimpse future trends in this area. As technology and disease understanding evolve, it's crucial that detection systems advance to address similar health challenges ahead.

- In Chapter 4, we introduce the Client Selection Secure Collaborative Learning (CSSCL) algorithm, a robust method that emphasizes privacy preservation. By leveraging a distributed learning architecture, this method ensures data privacy by sharing only model weights and processing data locally.

- Chapter 5 delves into the use of an FPGA cluster within the AIRBiS, an energy-efficient Reconfigurable system tailored for rapid and precise pneumonia detection from chest X-ray images.
- Chapter 6 unveils an event-driven, energy-efficient inference approach built on a fault-tolerant spike routing mechanism. This method, operating on a neuromorphic system with a scalable 3D-NoC processor, promises faster inference and reduced power usage while maintaining competitive accuracy levels relative to traditional ANN methods.
- Finally, Chapter 7 ends this dissertation with the conclusion, discussion, and plan for future work.

Chapter 2

Technical Background

The remarkable advancements in the field of artificial intelligence (AI) have revolutionized numerous domains, including healthcare, finance, and transportation. At the core of these advancements lies the development of deep learning techniques, which have been instrumental in overcoming the limitations of traditional machine learning approaches. The ability of deep learning algorithms to learn complex patterns from vast amounts of data has significantly improved the performance of AI systems, enabling them to tackle a wide range of challenging tasks.

As the demand for AI systems continues to grow, so does the need to understand the underlying principles, techniques, and challenges associated with the design and implementation of deep learning algorithms and architectures. Furthermore, with the increasing adoption of edge computing platforms, it is crucial to explore efficient methods for deploying deep learning models on resource-constrained devices while maintaining high performance and low energy consumption.

This introductory section will provide an overview of the core concepts and applications associated with deep learning, with a particular emphasis on convolutional neural networks (CNNs). The discussion will encompass the utility of CNNs in image recognition, medical image diagnosis, and the role of transfer learning in enhancing model performance and training efficiency. The subsequent sections will delve into the specificities of these topics, elaborating on the intricacies of CNNs, their application in medical image diagnosis, and the implementation of transfer learning techniques for improved outcomes in medical image diagnosis tasks.

2.1 Deep-Learning Algorithms and Architectures

Deep-learning algorithms and architectures are the foundation of modern artificial intelligence, enabling machines to learn complex patterns and representations from large amounts of data. In this section, we will discuss various deep-learning architectures, focusing on convolutional neural networks (CNNs) and their application in image recognition and medical image diagnosis. Additionally, we will explore transfer learning and its impact on model performance and training efficiency.

2.1.1 Convolutional Neural Networks (CNNs)

Convolutional neural networks (CNNs) are a type of deep-learning architecture specifically designed for processing grid-like data, such as images, by using convolutional layers to capture local spatial information [21]. CNNs have achieved state-of-the-art performance in various computer vision tasks, including image classification, object detection, and semantic segmentation [22, 23, 24]. Their ability to learn hierarchical feature representations from raw pixel data makes them particularly suitable for medical image diagnosis tasks, such as pneumonia detection from chest X-ray images.

2.1.2 Medical Image Diagnosis with CNNs

CNNs have been successfully applied to various medical image diagnosis tasks, including the detection and classification of diseases from radiology, pathology, and ophthalmology images [25, 26, 27]. By leveraging the powerful feature extraction capabilities of CNNs, researchers have been able to develop models that achieve performance comparable to or even surpass human experts in some medical image diagnosis tasks. For instance, Authors in [28] proposed CheXNet, a CNN-based model that achieved high accuracy in detecting pneumonia and other thoracic diseases from chest X-ray images, demonstrating the potential of deep learning in automating medical image diagnosis.

2.1.3 Transfer Learning

Transfer learning is a technique that enables the use of pre-trained deep-learning models as feature extractors or initializations for training new models on related tasks with limited data (Pan and Yang, 2010). This approach has proven effective in improving model performance and reducing training time, especially in domains where labelled data is scarce, such as medical image diagnosis. One common transfer learning strategy involves fine-tuning a pre-trained CNN, such as VGG, ResNet, or Inception, on a target task, allowing the model to learn task-specific features and representations from the limited available data [29].

2.1.4 Applications of Transfer Learning in Medical Image Diagnosis

The application of transfer learning in medical image diagnosis has led to significant improvements in model performance and generalization. For example, authors in [30] demonstrated that fine-tuning pre-trained CNNs on various medical imaging tasks, such as detecting lung nodules and interstitial lung diseases from chest X-ray and CT images, resulted in models with superior performance compared to those trained from scratch. Similarly, authors in [31] applied transfer learning to the task of detecting white matter hyperintensities in brain MRI images, showing that a pre-trained CNN could achieve state-of-the-art performance with relatively small training datasets.

2.1.5 Optimization Techniques

Optimization techniques are crucial for enhancing the performance of deep-learning algorithms. In this subsection, we will explore popular optimization methods, such as Stochastic Gradient Descent (SGD) [32], Adaptive Moment Estimation (Adam) [33], and RMSprop [34], detailing their importance in adjusting model parameters and minimizing the loss function. Furthermore, we will discuss regularization techniques, such as L1 and L2 regularization and dropout [35], which prevent overfitting and improve model generalization.

In summary, this section on deep-learning algorithms and architectures provides an overview of the fundamental concepts and techniques underlying CNNs, optimization methods, and their applications in image recognition and medical image diagnosis. By understanding the principles and methodologies of deep learning, researchers and practitioners can develop innovative AI systems that address the challenges and opportunities associated with the deployment of deep learning models in healthcare and other domains. This comprehensive understanding includes the importance of transfer learning for leveraging pre-trained models, as well as the role of optimization techniques in improving model performance and preventing overfitting.

2.2 Deep-Learning Inference

Deep-learning inference refers to the process of using trained deep-learning models to make predictions or classifications based on new, unseen data. This process is critical for deploying deep learning models in real-world applications. In this section, we will discuss various aspects of deep learning inference, including model quantization, pruning, and compression techniques, as well as their impact on model performance, accuracy, and resource utilization. We will also provide references to high-level research that has contributed significantly to these areas.

2.2.1 Model Quantization

Model quantization is a technique used to reduce the memory footprint and computational complexity of deep learning models by representing their weights and biases with a lower precision data type, such as int8 or int16, instead of the commonly used 32-bit floating-point numbers [36, 37]. This reduction in precision can lead to significant memory savings and faster inference times, with minimal impact on model accuracy. Research by [38] on incremental network quantization demonstrates the effectiveness of this approach in reducing model size and computational complexity.

2.2.2 Model Pruning

Model pruning is another technique used to reduce the computational complexity and memory requirements of deep learning models by removing redundant or less important neurons, connections, or entire layers from the network [39]. This approach can result in significant improvements in model efficiency, especially for large networks with a high degree of redundancy. In their research, Molchanov et al. [40] proposed a method called Variational Dropout Sparsification, which uses Bayesian optimization to prune deep learning models while preserving their accuracy.

2.2.3 Model Compression

Model compression aims to reduce the size of deep learning models by exploiting redundancies in their structure or using knowledge distillation techniques to transfer the learned knowledge from a large, complex model (teacher) to a smaller, simpler model (student) [41, 42]. The compressed models require less memory and computational resources while retaining a comparable level of accuracy to the original models. Research by [43] on deep model compression and acceleration demonstrated the feasibility of this approach for various deep learning architectures, including CNNs and RNNs.

2.2.4 Inference on Edge Devices

Deploying deep learning models on edge devices, such as smartphones, IoT devices, and embedded systems, presents unique challenges due to the resource constraints of these platforms. Researchers have proposed various techniques to optimize deep learning inference on edge devices, including efficient network architectures [44], binary and ternary neural networks [45, 46], and hardware-aware optimization techniques [47]. These approaches aim to strike a balance between model accuracy, computational complexity, and resource utilization to enable efficient deep-learning inference on edge devices.

In conclusion, this section on deep-learning inference covers the essential techniques and strategies used to optimize deep-learning models for deployment in various applications, particularly on resource-constrained edge devices. By understanding and implementing these approaches, researchers and practitioners can effectively address the challenges associated with deep learning inference and develop efficient, accurate models for a wide range of real-world scenarios.

2.3 Neuromorphic Computing

Neuromorphic computing refers to the design of artificial neural networks and computing systems that emulate the structure and function of biological neural networks found in the human brain. These systems offer unique advantages such as low power consumption, real-time processing capabilities, and adaptability, making them suitable for a variety of applications, particularly those involving resource-constrained edge devices. In this section, we will discuss the fundamental concepts of neuromorphic computing, including spiking neural networks, neuromorphic hardware, and learning mechanisms.

2.3.1 Spiking Neural Networks (SNNs)

Spiking neural networks (SNNs) are a type of artificial neural network that incorporates the temporal dynamics of biological neurons, using spikes or action potentials to transmit information between neurons [48]. Unlike traditional artificial neural networks, which rely on continuous values, SNNs use discrete events, making them more biologically plausible and energy-efficient. Research in [49] provides an overview of SNNs, their properties, and their potential applications in neuromorphic computing.

2.3.2 Neuromorphic Hardware

Neuromorphic hardware refers to specialized computing devices designed to implement neuromorphic computing systems efficiently. These devices often leverage specialized analogue or digital circuits to mimic the behaviour of biological neurons and synapses, resulting in low power consumption and high computational efficiency. Prominent examples of neuromorphic hardware include IBM's TrueNorth [50] and Intel's Loihi [51]. These chips demonstrate the potential of neuromorphic hardware in providing energy-efficient solutions for a wide range of AI applications.

2.3.3 Learning Mechanisms in Neuromorphic Systems

Learning mechanisms in neuromorphic systems play a crucial role in enabling these systems to adapt and process information in a manner similar to biological neural networks. One of the most widely studied learning mechanisms in neuromorphic systems is spike-timing-dependent plasticity (STDP), which adjusts the strength of synaptic connections based on the relative timing of presynaptic and postsynaptic spikes [52]. Research in [53] on unsupervised learning in SNNs using STDP demonstrates the potential of this learning mechanism in facilitating the development of self-organizing neuromorphic systems.

2.3.4 Neuromorphic Computing Applications

Neuromorphic computing has been applied to various domains, including computer vision, robotics, and sensor networks, due to its inherent advantages in energy efficiency, adaptability, and real-time processing capabilities. For instance, research in [54] on neuromorphic vision systems highlights the potential of SNNs and neuromorphic hardware in enabling low-power, high-speed visual processing for applications such as autonomous vehicles and drones. Additionally, authors in [55] discuss the potential of neuromorphic computing in the field of robotics, with a focus on biologically inspired control mechanisms and sensory processing.

In conclusion, this section on neuromorphic computing provides an overview of the fundamental concepts, techniques, and applications of this rapidly evolving field. By understanding the principles and methodologies underlying neuromorphic computing, researchers and practitioners can leverage its unique advantages to develop innovative, energy-efficient AI systems that address the challenges and opportunities associated with the deployment of deep learning models on resource-constrained edge devices.

2.4 Hardware Acceleration

Hardware acceleration is the process of leveraging specialized hardware components to boost the performance of certain computational tasks, particularly those associated with deep-learning algorithms. As the demand for faster computation and processing in the field of deep learning grows, the significance of hardware acceleration has become more pronounced. In this section, we will examine various hardware acceleration platforms, including Graphics Processing Units (GPUs), Field-Programmable Gate Arrays (FPGAs), and Application-Specific Integrated Circuits (ASICs), discussing their benefits and drawbacks in terms of computational power, energy efficiency, and flexibility.

2.4.1 Graphics Processing Units (GPUs)

Graphics Processing Units (GPUs) are specialized hardware accelerators initially designed for rendering graphics in video games and other visualization tasks. However, due to their massively parallel architecture and high computational capabilities, GPUs have been widely adopted for accelerating deep-learning algorithms, significantly reducing training and inference times [56].

GPU Architecture and Parallelism

Modern GPUs consist of thousands of small processing cores, which are organized into multiple streaming multiprocessors (SMs). These cores are capable of executing many threads concurrently, making GPUs suitable for the parallel nature of matrix and vector operations that are common in deep-learning algorithms. The CUDA (Compute Unified Device Architecture) programming model, developed by NVIDIA, allows researchers and developers to leverage the computational power of GPUs by writing parallelized code, facilitating the efficient execution of deep-learning workloads.

GPU-accelerated Deep Learning Frameworks

To harness the power of GPUs for deep learning, several deep-learning frameworks have integrated GPU support, such as TensorFlow [57], PyTorch [58], and Caffe [59]. These frameworks enable researchers and practitioners to develop and train deep-learning models on GPUs without the need for low-level programming, thereby accelerating model training and reducing development time.

GPUs in Medical Image Analysis

The application of GPU-accelerated deep learning has led to significant improvements in medical image analysis tasks. For instance, authors in [60] employed a GPU to accelerate the training of a 3D U-Net for segmenting brain tumours from MRI scans, achieving a 24-fold speedup compared to CPU-based training. Additionally, Authors in [61] utilized a GPU to train a CNN for detecting interstitial lung diseases from CT images, demonstrating a 40-fold reduction in training time compared to a CPU implementation.

In summary, GPUs play a vital role in deep-learning hardware acceleration, providing significant speedups in training and inference times for various tasks, including medical image analysis. The massively parallel architecture of GPUs, coupled with GPU-accelerated deep-learning frameworks, enables researchers and practitioners to develop and deploy efficient AI systems for healthcare and other domains.

2.4.2 Field-Programmable Gate Arrays (FPGAs)

Field-Programmable Gate Arrays (FPGAs) are reconfigurable digital integrated circuits that can be tailored to specific computational tasks, offering high performance, low power consumption, and flexibility in comparison to fixed-function hardware, such as GPUs and CPUs [62]. FPGAs have become increasingly popular for accelerating deep-learning algorithms due to their ability to implement custom parallel architectures that are optimized for specific tasks [63].

FPGA Architecture and Reconfigurability

FPGAs consist of programmable logic blocks, memory blocks, and configurable interconnects, allowing users to define custom digital circuits and data paths that can be tailored to specific computational needs [64]. This reconfigurability enables the implementation of task-specific hardware accelerators that can achieve high performance and energy efficiency by exploiting the inherent parallelism and pipelining in deep-learning algorithms [65].

FPGA-based Deep Learning Frameworks and Tools

Several deep-learning frameworks and tools have been developed to facilitate the deployment of deep-learning models on FPGAs, such as the Xilinx Deep Learning Processor Unit (DPU) [66] and the Intel OpenVINO toolkit [67, 68]. These frameworks and tools provide high-level programming interfaces that abstract the underlying FPGA hardware, enabling researchers and developers to implement and optimize deep-learning models on FPGAs without requiring extensive knowledge of hardware design.

FPGAs in Medical Image Analysis

In medical imaging, where speed, accuracy, and adaptability are of utmost importance, FPGAs have demonstrated significant potential. FPGAs can execute many operations in parallel, making them especially effective for tasks like image processing, where each pixel can be processed simultaneously. They can be reprogrammed to cater to specific algorithms or tasks, making them adaptable to the ever-evolving field of medical image analysis. In contrast to traditional CPUs and even GPUs, FPGAs often consume less power, making them suitable for mobile or portable imaging devices. Developers can design custom hardware circuits specifically optimized for a particular image analysis algorithm, ensuring peak performance. FPGAs can be used in ultrasound, MRI, and other imaging techniques where real-time data processing is essential. Techniques like noise reduction, contrast enhancement, and edge detection can be accelerated using FPGAs. For modalities like CT and MRI, FPGAs can facilitate fast 3D reconstructions of the scanned anatomy.

In summary, FPGAs offer a promising alternative to GPUs and CPUs for deep-learning hardware acceleration, providing flexibility, high performance, and energy efficiency in various applications, including medical image analysis. By leveraging FPGA-based deep-learning frameworks and tools, researchers and practitioners can develop and deploy custom hardware accelerators that are optimized for specific tasks and requirements, enabling the efficient implementation of AI systems in healthcare and other domains.

2.4.3 Application-Specific Integrated Circuits (ASICs)

Application-Specific Integrated Circuits (ASICs) are custom-designed hardware components that are optimized for a specific application, offering superior performance and energy efficiency compared to general-purpose processors like CPUs, GPUs, and FPGAs [69]. In the context of deep learning, ASICs have emerged as an attractive option for accelerating both training and inference tasks, as they can be tailored to the unique requirements of deep-learning algorithms [70].

ASIC Design and Customization

ASICs are designed from the ground up to perform a specific function, utilizing custom digital circuits and data paths that are optimized for a particular application. This customization enables ASICs to achieve high levels of performance and energy efficiency by exploiting the inherent parallelism and pipelining in deep-learning algorithms, as well as implementing specialized memory hierarchies and data formats [71].

ASIC-based Deep Learning Accelerators

Several ASIC-based deep learning accelerators have been developed in recent years, demonstrating the potential of ASICs for high-performance and energy-efficient deep learning. For example, Google's Tensor Processing Unit (TPU) is a custom ASIC designed to accelerate both training and inference tasks for TensorFlow models, offering significant improvements in performance and power efficiency compared to GPUs [71]. Another example is the Graphcore Intelligence Processing Unit (IPU), an ASIC specifically designed for machine learning workloads, delivering high performance and low power consumption for a wide range of deep-learning tasks [72].

ASICs in Medical Image Analysis

The use of ASICs for accelerating deep-learning algorithms has led to substantial advancements in medical image analysis. For instance, the Cambricon-X chip, an ASIC designed for edge-based deep learning, was utilized for real-time lung nodule detection in CT scans, achieving a detection accuracy of 92.3% and an inference speed of 0.64 seconds per image [73].

Challenges and Trade-offs

Despite their performance and energy efficiency advantages, ASICs face several challenges and trade-offs. Designing and fabricating custom ASICs can be expensive and time-consuming, limiting their applicability for small-scale projects and rapid prototyping.

In summary, ASICs offer considerable potential for accelerating deep-learning algorithms in various applications, including medical image analysis. By leveraging custom digital circuits and data paths, ASICs can achieve high performance and energy efficiency, making them an attractive option for implementing AI systems in healthcare and other domains. However, the cost and inflexibility of ASICs may limit their applicability in certain scenarios, making it essential to carefully consider the trade-offs between performance, energy efficiency, cost, and flexibility when selecting an appropriate hardware acceleration platform.

While each hardware acceleration platform comes with its own set of advantages, they also present specific challenges. GPUs, for example, may consume more power than ASICs but offer more general-purpose flexibility. FPGAs, although reconfigurable, might not always match the raw computational speed of GPUs. ASICs, on the other hand, deliver high performance for specific tasks but lack the versatility to adapt to varying computational requirements. In conclusion, choosing the right hardware acceleration platform hinges on understanding the specific demands and constraints of the deep-learning application in question. Each platform offers unique benefits but also comes with its own limitations, which must be carefully weighed before making a decision.

2.4.4 Performance Evaluation Metrics

Evaluating the performance of hardware acceleration platforms is crucial for determining their suitability for specific deep-learning applications. In this subsection, we will explore common performance evaluation metrics, such as throughput, latency, power consumption, and energy efficiency, and discuss their relevance in comparing and selecting appropriate hardware platforms for deep-learning inference tasks. Throughput refers to the number of tasks completed per unit of time, and it is an important metric for assessing the computational capabilities of a platform. Latency, on the other hand, measures the time taken for a single task to be completed, reflecting the responsiveness of a system. Low latency is crucial for real-time applications, such as autonomous vehicles or medical diagnostics.

Power consumption is another critical metric, especially for edge computing devices and battery-powered systems. It indicates the amount of electrical power required by a platform to execute a given task. Energy efficiency, a related metric, measures the ratio of the useful work performed by a system to the energy consumed during that process. High energy efficiency is essential for reducing operational costs and minimizing the environmental impact of computing systems.

Summary of Technical Background

In conclusion, this chapter has provided an overview of the technical background related to deep-learning algorithms and architectures, deep-learning inference, neuromorphic computing, and hardware acceleration. It has also introduced various optimization techniques and performance evaluation metrics that are essential for the design and deployment of efficient and effective deep-learning systems.

Chapter 3

Literature Survey

In recent years, the field of medical imaging has witnessed a surge in technological advancements, particularly in computer-aided detection systems. One of the areas that has garnered significant attention is the detection of pneumonia, especially given the urgency brought about by the COVID-19 pandemic. Such systems, underpinned by software, hardware, and algorithmic optimizations, aim to provide quicker, more accurate diagnoses, thereby aiding healthcare professionals in timely interventions. This chapter delves into various approaches to computer-aided pneumonia detection, exploring both software and hardware-based methodologies. Moreover, we shed light on the current state-of-the-art techniques to accelerate deep-learning inference, a pivotal component in modern detection systems, and delve into the latest research that focuses on scaling such inferences to meet the burgeoning demands. In the rest of this chapter, we survey some of these works as summarized in Figure 3.1, which shows a summary of COVID-19 detection methods on various computing platforms.

AI-based pneumonia detection solutions can be classified into software-based solutions and hardware-based solutions, which have low latency, low cost, and low power consumption. Software-based solutions contain non-DNN-based methods and DNN-based methods. The Non-DNN-based methods are simple and straightforward. However, they have limited capability and performance and are difficult for generalization. DNN-based methods include traditional CNN and collaborative learning. Generally, the DNN-based methods achieve high accuracy. However, the method requires high power consumption. Also, it has potential security issues for both data and models. As for the hardware-based solution, they can be classified into the embedded accelerator and reconfigurable AI chip. The embedded accelerator contains Google Edge TPU and Nvidia Jetson, which have low latency, low cost, and low power consumption. However, they have drawbacks like limited capability and performance, the difficulty of generalization, and adaptability. For a reconfigurable AI chip, it performs more adaptability, is flexible, and requires low power consumption. However, it is complicated in design and requires high equipment costs.

3.1 Non-DNN-Based Approaches on Pneumonia Detection

Mosley [2] delved into the RT-PCR and serological detection methods for COVID-19. These methods have exerted significant strain on medical systems, presenting both cost and accuracy challenges. Bell [3] provided a comprehensive discussion on COVID-19's origin, diagnosis, treatment, and other facets. They highlighted that while clinical manifestations vary extensively, symptoms remain individual-specific. They noted that while chest CT scans are not the primary recommendation for infection detection, they can be beneficial in spotting complications. This was further elucidated by Tian et al. [74], who explored the diagnostic value and consistency between chest CT scans and RT-PCR testing methods in the context of COVID-19.

In their research, Aydin et al. [75] investigated the correlation between indeterminate lesions of COVID-19 pneumonia detected on computed tomography and RT-PCR test results. They found that peripheral indeterminate lesions are often indicative of COVID-19 pneumonia. Furthermore, when indeterminate lesions like tree-in-bud pattern, acinar nodules, and limited consolidation area are observed, other diagnoses should be considered, even if ground-glass opacities are also present. Their findings provide valuable insights that can contribute to a more nuanced understanding and diagnosis of COVID-19 pneumonia based on computed tomography imaging.

Zhou et al. [76] advanced the field with the proposal of many-to-one distribution learning and K-nearest neighbour smoothing (KNNS) methods. Their aim was to elevate the precision of a single model while mitigating label uncertainty and ambiguity issues.

3.2 DNN-Based Approaches on Pneumonia Detection

Asraf et al. [77] surveyed the role of deep learning in combating COVID-19, covering areas such as protein structure prediction, drug discovery acceleration, and infection detection using medical imagery. In contrast, Nakamura et al. [78] focused on assembling a dataset and assessing the accuracy of various convolutional neural networks (CNNs). Despite their promising findings, the study's approach was constrained by a limited dataset size, and the neural network model was not easily scalable.

Mustafa and Rahimi Azghadi [79] overviewed the advancements in AutoML technology, emphasizing its potential in healthcare. This paves the way for the AI community to leverage automated learning for medical notes, thereby decreasing the heavy dependence on human expertise during machine learning training. Latif et al. [80] emphasized the importance of automated methods for COVID-19 detection and diagnosis in light of ongoing pandemic challenges. They demonstrated that combining deep learning algorithms for feature extraction with proven classifiers can yield accurate results from chest CT scans. Their approach achieved an impressive 99.9% accuracy, surpassing previous studies. The authors also noted the potential of their technology to address both the current pandemic and future health emergencies.

In their study, Zhou et al. [81] introduced two contrastive abnormal attention models, notably enhancing the detection accuracy of chest X-ray images. Their novel designs include a dual-weighting graph convolution neural network and a left-right lung contrastive network.

The study conducted by Shiri et al. [82] presents a deep learning (DL) algorithm called COLI-Net, designed to detect and segment whole lung and COVID-19 pneumonia infectious lesions using chest computed tomography (CT) images. The researchers established the efficacy of their approach through a multi-centre study involving a significant number of CT examinations (Shiri et al. 2022). The remarkable segmentation accuracy indicated by Dice coefficients for both lung and lesion segmentation signifies the potential of this method for a fast, consistent, and robust framework for lesion detection and quantification, reducing the risk of human error. Moreover, the research demonstrates the significant utility of transfer learning for enriching the identification of specific COVID-19 pneumonia features from clinical studies. The applicability of the developed AI model across diverse populations and stages of COVID-19 from multiple centres around the globe underscores its potential for automated progression/regression assessment of pneumonia lesions in follow-up studies, providing essential diagnostic and prognostic metrics.

In their work, Ieracitano et al. [83] introduced a novel approach that combines fuzzy logic and deep learning techniques to effectively differentiate between chest X-ray (CXR) images of COVID-19 patients and those with interstitial pneumonia not related to COVID-19. Their model, named CovNNet, leverages the fusion of CXR images and fuzzy images generated by a fuzzy edge detection algorithm to extract relevant features for the task, demonstrating a superior classification performance with an accuracy rate reaching up to 81%. The authors underscored the utility of the approach in the context of the ongoing COVID-19 pandemic, providing a tool that could assist in early diagnosis and effective patient triaging, alluding to its potential for integration into existing clinical decision support systems.

3.3 Hardware-Based Approaches on Pneumonia Detection

While there's limited exploration of hardware-based approaches in computer-aided COVID-19 detection, Rahman et al. [84] introduced a distributed learning framework for COVID-19 detection utilizing a 5G network at the edge. They deployed several FPGA boards for edge nodes. Despite the promising detection accuracy of their frameworks, their primary emphasis was on the benefits of 5G in this context, assessing its performance, complexity, and power consumption.

FPGAs have shown potential with convolutional neural network (CNN) workloads due to their streaming characteristics and compatibility with reconfigurable architectures. Abdelouahab et al. [85] studied various CNN parallelization techniques, providing insights into FPGA-based CNN inference accelerators, workload computations, memory access patterns, and performance optimizations across different layers. Zhao et al. [14] put forth a re-configurable framework for training CNNs, employing streaming data paths with adjustable parameterized modules to adapt during training.

Ashraf et al. [86] designed a smart edge surveillance framework that leverages personal fever metrics, heart rate, and radiological features for COVID-19 detection. Using FPGA boards for edge nodes, they focused on a communication chain to identify infected individuals, although deep learning wasn't incorporated. Fu et al. [87] developed an FPGA-based accelerator for agent-based epidemic modeling to predict COVID-19 spread patterns and intervention effectiveness. While the model exhibited high acceleration, its primary purpose wasn't COVID-19 detection or diagnosis.

Several studies have focused on leveraging accelerated DL systems on edge platforms for swift COVID-19 diagnosis and detection. Goel et al. [88] introduced ComputeCOVID19+, a CT-based framework for COVID-19 diagnosis and monitoring, optimized for a variety of platforms such as multi-core CPUs, GPUs, and FPGAs. Shen et al. [89] put forth an algorithm for COVID-19 detection that combines deep features with discrete social learning particle swarm optimization. They employed a pre-trained ResNet18 to derive features from CXR images, which were subsequently refined and classified using the swarm optimization technique and a support vector. Additionally, a unique normalization technique, grounded in CNN, was suggested by Yaman et al. [90]. This method, tailored for FPGA implementation, aids in the preliminary assessment of COVID-19.

3.4 Privacy and Security Concerns on Pneumonia Detection

There has been considerable recent effort to ensure security and privacy in collaborative learning. While techniques like differential privacy [91, 92, 93], cryptographic methods [94], and trusted execution environments [95, 96, 97] have been proposed, many encounter high computational costs or diminished learning performance. Federated learning, introduced by McMahan et al.[6], offers a promising framework that trains on distributed nodes or clients, keeping raw data localized, thereby addressing many privacy concerns. Yet, it's not without challenges, especially pertaining to system design and model security[98].

A noteworthy contribution came from Wang and Abdallah [99], who presented the FL-QLMS approach. It showcased resistance against client-side attacks, holding its performance even when many models were compromised. However, their study primarily targeted fully connected networks, overlooking the widely-used convolutional neural networks (CNNs). This narrow lens potentially limits the broader applicability and generalizability of their method.

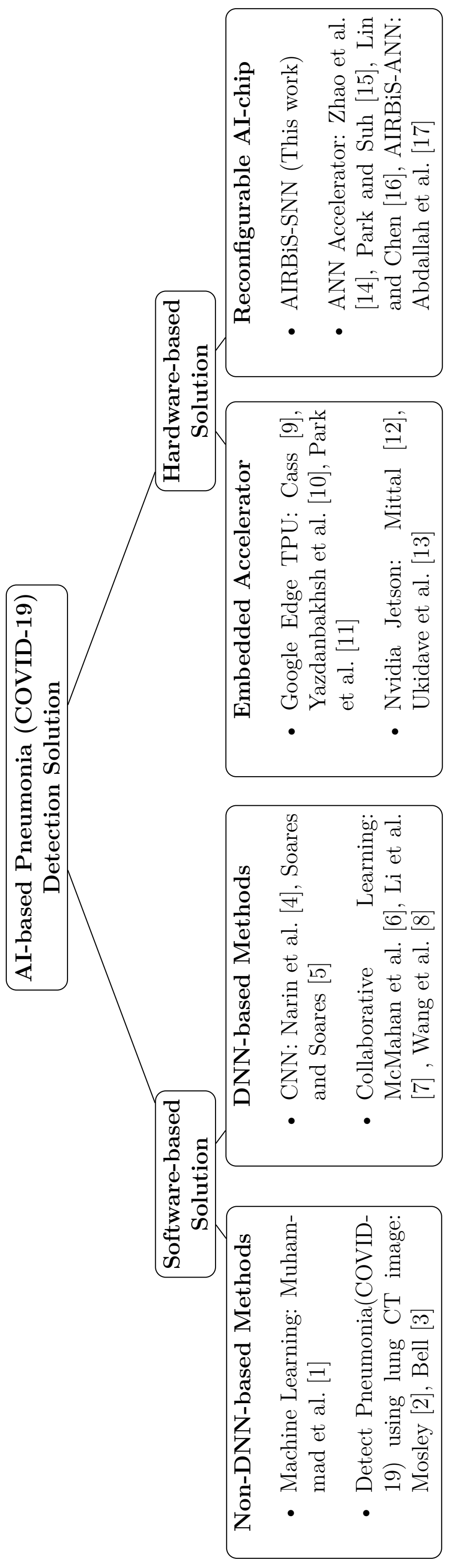

AI-based Pneumonia (COVID-19) Detection Solution
Software-based Solution
Hardware-based Solution
Non-DNN-based Methods
• Machine Learning: Muhammad et al. [1]
• Detect Pneumonia(COVID-19) using lung CT image: Mosley [2], Bell [3]
DNN-based Methods
• CNN: Narin et al. [4], Soares and Soares [5]
• Collaborative Learning: McMahan et al. [6], Li et al. [7] , Wang et al. [8]
Embedded Accelerator
• Google Edge TPU: Cass [9], Yazdanbakhsh et al. [10], Park et al. [11]
• Nvidia Jetson: Mittal [12], Ukidave et al. [13]
Reconfigurable AI-chip
• AIRBiS-SNN (This work)
• ANN Accelerator: Zhao et al. [14], Park and Suh [15], Lin and Chen [16], AIRBiS-ANN: Abdallah et al. [17]

Figure 3.1: Summary of COVID-19 detection methods on various computing platforms [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17].

McMahan et al. [6] introduced a federated learning method for deep networks. Instead of uploading sensitive data, it employs local training and model sharing, proving especially robust against imbalanced and non-IID data. This approach also reduces the communication rounds required by synchronous stochastic gradient descent by up to 100 times. Lu et al. [100] combined blockchain and collaborative learning to ensure secure data sharing in IoT, though data privacy issues remained unresolved. In the study by Khan et al. [101], a DNN architecture called CoreNet was developed for COVID-19 detection from chest X-ray images. Despite showing potential, its generalization was hampered by limited COVID-19 training data and data imbalance.

3.5 Scaling Deep Learning Inference Acceleration on Pneumonia Detection

The complexity of DNNs has grown over time, sometimes expanding to the petascale. While this scale isn't an issue for cloud computing applications [102], it can be problematic for scenarios prioritizing low latency and security due to computational and power constraints. Addressing this, efforts have shifted toward optimizing hardware platforms for edge inference.

In this context, Wei et al. in [103] introduced a systolic array architecture optimized for FPGA. This design links processing elements in a mesh topology, minimizing global data transfers and enhancing matrix and convolution operations. However, its scalability is somewhat restricted due to its fixed shape.

Separately, Ito et al. in [104] presented an interconnection network using 24 FPGA boards and static time division multiplexing. While their approach yielded promising outcomes, manual optimization was essential for each task, making slight changes impactful to overall performance.

Summary of Literature Survey

Throughout this chapter, we've ventured through a comprehensive journey exploring the multifaceted approaches to computer-aided pneumonia detection. From software-centric methodologies to hardware-optimized systems, it's evident that the quest for improved detection mechanisms is multifarious. The urgency of the COVID-19 pandemic has further underscored the significance of these advancements. By understanding the intricacies of deep-learning inference acceleration and the ongoing research to scale these inferences, we gain insights into the potential future trajectories in this domain. As technology continues to evolve and as our understanding of diseases deepens, it's imperative for these detection systems to remain at the forefront of innovation, ensuring that we are better equipped to tackle health crises of similar magnitudes in the future.

Chapter 4

Robust Privacy-Preserving Method Based on a Client Selection Secure Collaborative Learning Algorithm

4.1 AI-Enabled Pneumonia Detection

Artificial Intelligence (AI) has been progressively studied across various sectors, including healthcare, where prompt anomaly detection is crucial for patient monitoring [105, 106]. The onset of COVID-19 at the close of 2019, caused by the SARS-CoV-2 virus, has heavily impacted global communities. Many affected individuals face delayed treatment due to challenges in timely diagnosis.

The conventional diagnostic system, as depicted in Figure 4.1, requires patients to visit hospitals, undergo X-ray scans, and await individual evaluations by doctors. This process often leads to overcrowded facilities, overworked medical professionals, and delayed reporting to decision-makers. Recent advancements in machine learning, both in software and hardware, have been pivotal in the fight against COVID-19. Numerous studies affirm the efficacy of AI-driven COVID-19

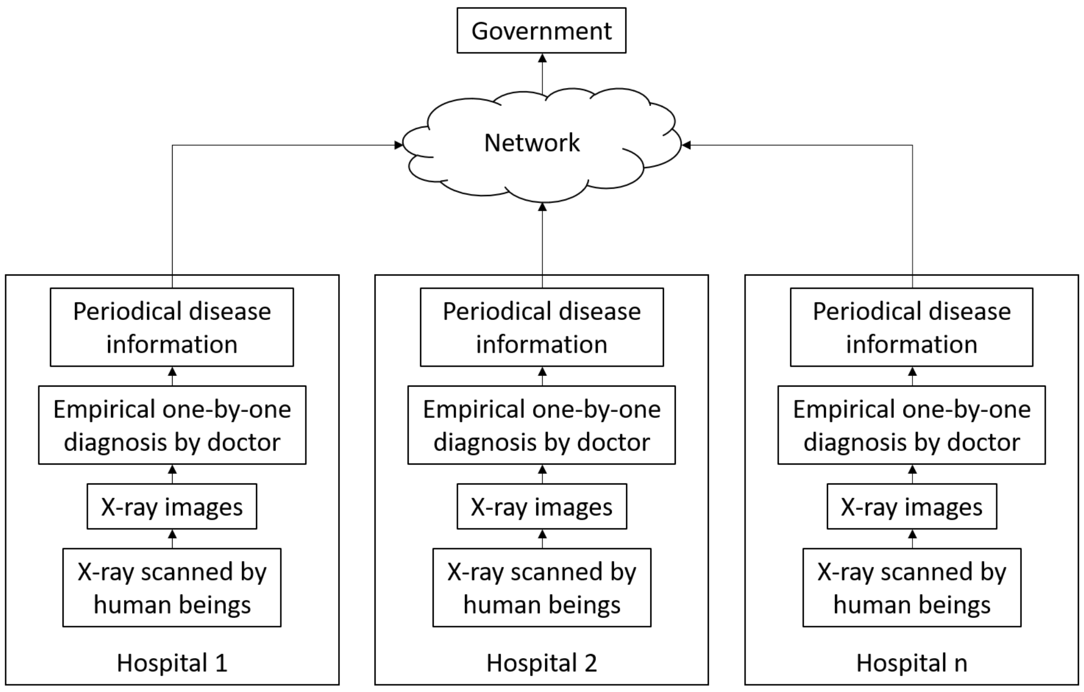

Government
Network
Periodical disease information
Empirical one-by-one diagnosis by doctor
X-ray images
X-ray scanned by human beings
Hospital 1
Hospital 2
Hospital n

Figure 4.1: The Conventional System Organization. The COVID-19 diagnosis is conducted in each hospital. The X-ray image is scanned one by one by humans. The empirical diagnosis of scanned X-ray images is conducted one by one by doctors. The disease information from each hospital is sent to the government by doctors via the network.

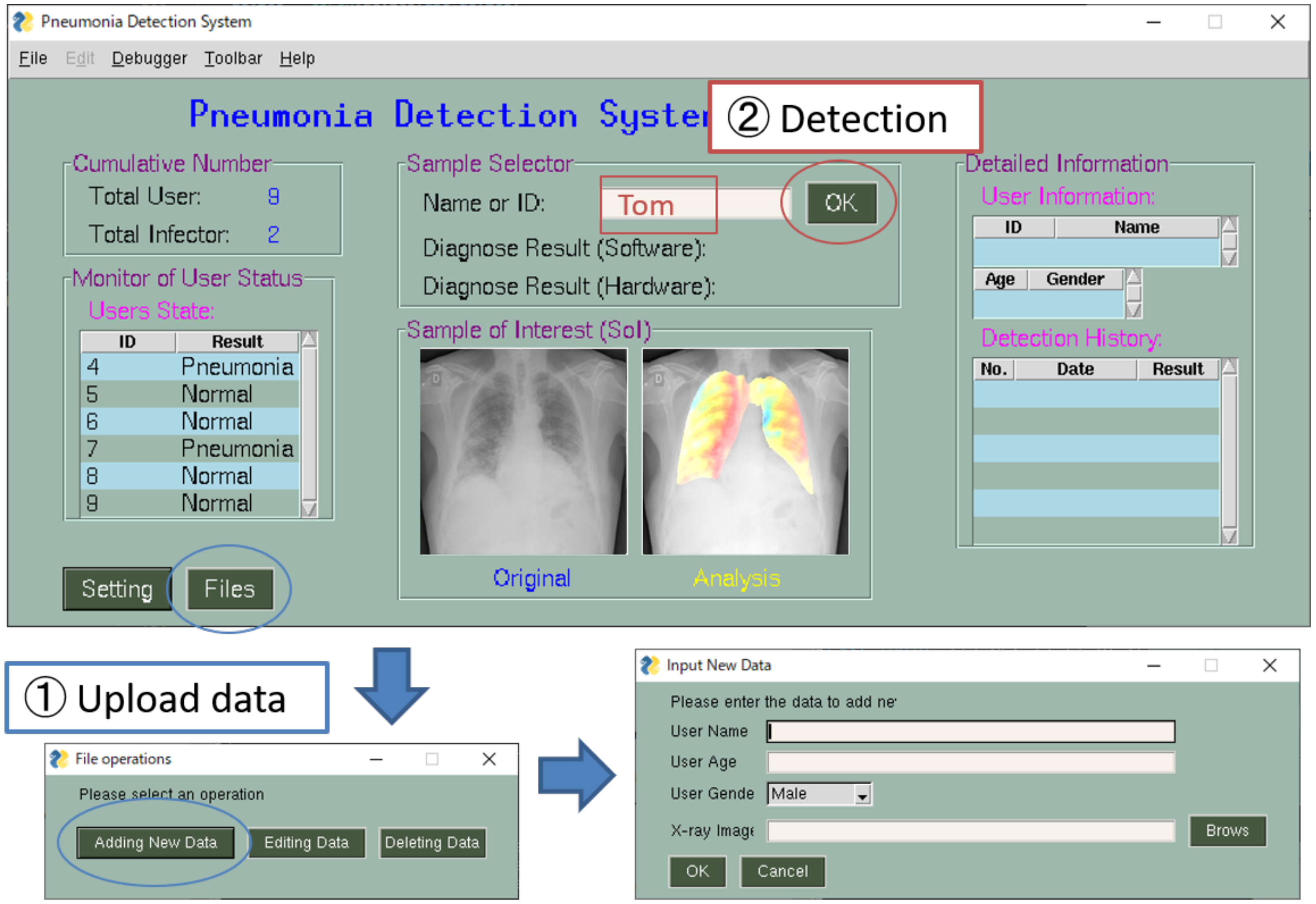
diagnoses. However, there is an evident gap for an integrated AI system that ensures swift, precise diagnostics along with real-time analytics and timely outcomes.

[74] explored the diagnostic capabilities of chest CT scans in comparison to RT-PCR test methods for COVID-19. The findings indicate that while chest CT scans can be employed for detection, their high cost and radiation exposure make them less ideal for frequent use. In their study, Asraf et al. [77] delve into the applications of deep learning in battling COVID-19, particularly in drug discovery and infection detection through medical imagery. Similarly, Narin et al. [4] utilized binary classification on chest X-ray data, including those from COVID-19 patients, achieving high accuracy rates.

While significant progress has been made using deep learning in the fight against COVID-19, the global challenge underscores the importance of swift, accurate, and real-time diagnostic tools. The ideal system would enable doctors to focus solely on patient care. The rise of artificial intelligence (AI) has been monumental in numerous fields, especially in biomedicine, where quick anomaly detection is critical. The current global challenge posed by COVID-19, a disease caused by the SARS-CoV-2 virus, amplifies the need for efficient detection systems. Drawing inspiration from Ahmed et al. [107], who designed a health monitoring system, we introduced the AI-enabled Real-time Biomedical System (AIRBiS) [17]. This system, powered by a deep neural network, efficiently detects conditions like pneumonia with a user-friendly interface, making it a potential tool in the ongoing battle against COVID-19.

The conventional RT-PCR test, commonly used for COVID-19 diagnosis, has a sensitivity range of 60% to 97%[108]. Yet, owing to variations in disease presentation among patients, this sensitivity can dip to between 60%-71%[109], leading to a notable number of false negatives. This is concerning as both symptomatic COVID-19 patients and asymptomatic carriers might still spread the virus if not accurately detected. To enhance diagnostic accuracy, some have turned to analyzing lung X-ray images, achieving an accuracy of 80-90% [110]. However, the manual nature of this method, requiring individual patient assessments, becomes increasingly untenable as patient numbers surge, resulting in diagnostic delays and inefficiencies. In response to these challenges, the system [111, 112, 78, 113] depicted in Figure 1.2 has been proposed to offer a more coordinated and efficient approach.

The AIRBiS system simplifies diagnostic processes. Users upload their chest X-ray images to the system's GUI, illustrated in Figure 4.2, and promptly receive a reliable diagnosis. This enables doctors to easily monitor real-time diagnosis and review the diagnosis procedure for specific samples.



Cumulative Number
Total User: 9
Total Infector: 2

Name or ID: Tom
Diagnose Result (Software):
Diagnose Result (Hardware):

Monitor of User Status
Users State:

ID	Result
4	Pneumonia
5	Normal
6	Normal
7	Pneumonia
8	Normal
9	Normal

ID	Name

Age	Gender

No.	Date	Result

Please select an operation

Please enter the data to add ne
User Name
User Age
User Gende Male
X-ray Image

Figure 4.2: System user interface with the function of showing the X-ray image case, the sample of interest, the diagnosis results, the result statistic, user information database, and connecting to edge inference accelerator.

4.2 AIRBiSNet Algorithm

To assess the efficacy of AIRBiS, we sourced a lung X-ray image dataset from Kaggle [114, 115, 116]. Notably, there's an imbalance between COVID-19 and non-COVID image counts, necessitating augmentation. The relevant code is available in Listing 4.1.

```python
from keras.preprocessing.image import load_img, img_to_array
from keras.preprocessing.image import ImageDataGenerator
import numpy as np

for i, file in enumerate(files):
    img = load_img(file)
    x = img_to_array(img)
    x = np.expand_dims(x, axis=0)

    # ImageDataGenerator
    datagen = ImageDataGenerator(
        #rotate or shear
        rotation_range=5    #[−5,5]
        #shear_range = 0.1
    )
    # Generate images
    g = datagen.flow(x, batch_size=1, save_to_dir=output_dir, save_prefix='
    aug_'+ str(i), save_format='png')
    for ___ in range(2):
        batch = g.next()
```

Listing 4.1: The code of data augmentation (Python).

Our CNN-based detection method, as shown in Figure 4.3, takes in grayscale or RGB X-ray images. This CNN has three convolution layers with a 3×3 kernel

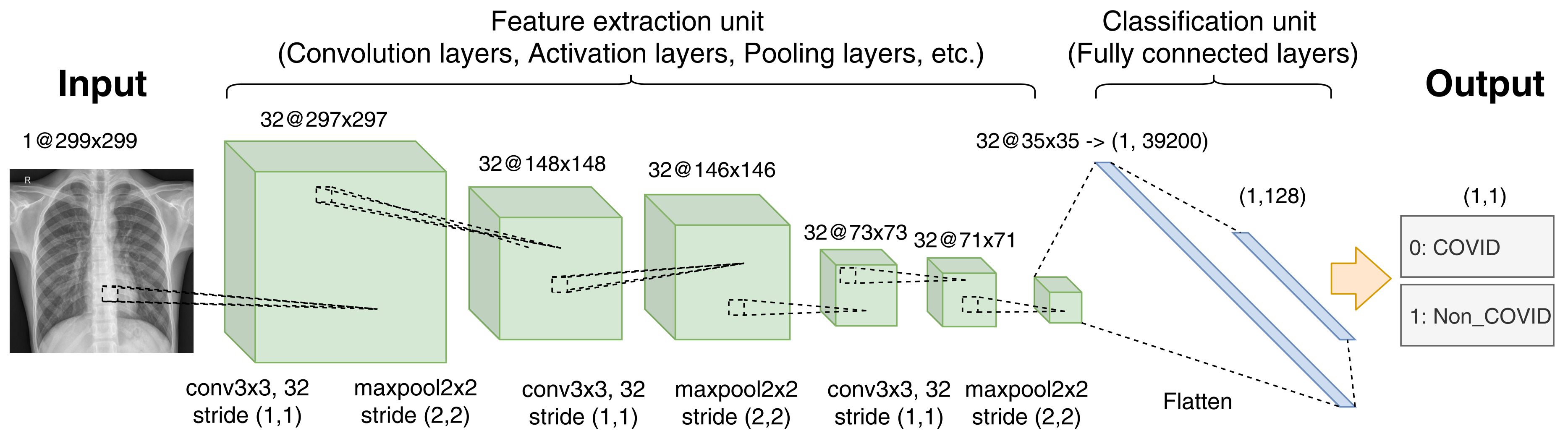


Figure 4.3: AIRBiS diagnosis flow. Input case from the user interface (UI), detect with AIRBiSNet, output detection result, and return to UI.

size and 32 biases, each followed by a Max pooling layer. After flattening, two fully connected layers lead to a final layer that predicts infection status. We utilize the $softmax$ classifier for class probabilities [78] and base our training on the cross-entropy loss, detailed in Equation 4.1.

$$E\left(\boldsymbol{w}_0, \boldsymbol{b}_o\right) = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{y}_i \times \log\left(\boldsymbol{y}_i'\left(\boldsymbol{w}_o, \boldsymbol{b}_o\right)\right) \tag{4.1}$$

Given the network parameters $\boldsymbol{w}_o$ and $\boldsymbol{b}_o$, and using $\boldsymbol{y}$ to represent the real label and $\boldsymbol{y}'$ for the predicted label, we optimize parameters and make classification decisions by minimizing the loss function through the stochastic gradient descent and back-propagation algorithms.

The proposed AIRBiSNet is potentially susceptible to overfitting. To mitigate this, a dropout layer has been incorporated. Dropout enhances performance by occasionally turning off neurons during training [117]. As depicted in Figure 4.4, the dotted line represents the validation loss without dropout, which tends to increase. In contrast, the solid line, representing the AIRBiSNet with dropout, demonstrates a more favourable validation loss, suggesting effective overfitting suppression.

AIRBiSNet 1 is a Convolutional Neural Network (CNN) designed to classify lung X-ray images. The core components and functionalities are as follows:

- **Input:** Image of size $Input_{width} \times Input_{height} \times Input_{channel}$.
- **CNN:** Contains multiple convolutional layers with ReLU activations, interspersed with pooling layers. This structure helps in downsampling the image and extracting pertinent features.
- **Flattening:** Post-CNN, the output is transformed into a 1D vector of length N_f.
- **Fully Connected Layers:** These process the 1D vector, and their depth and parameters can be user-specified.

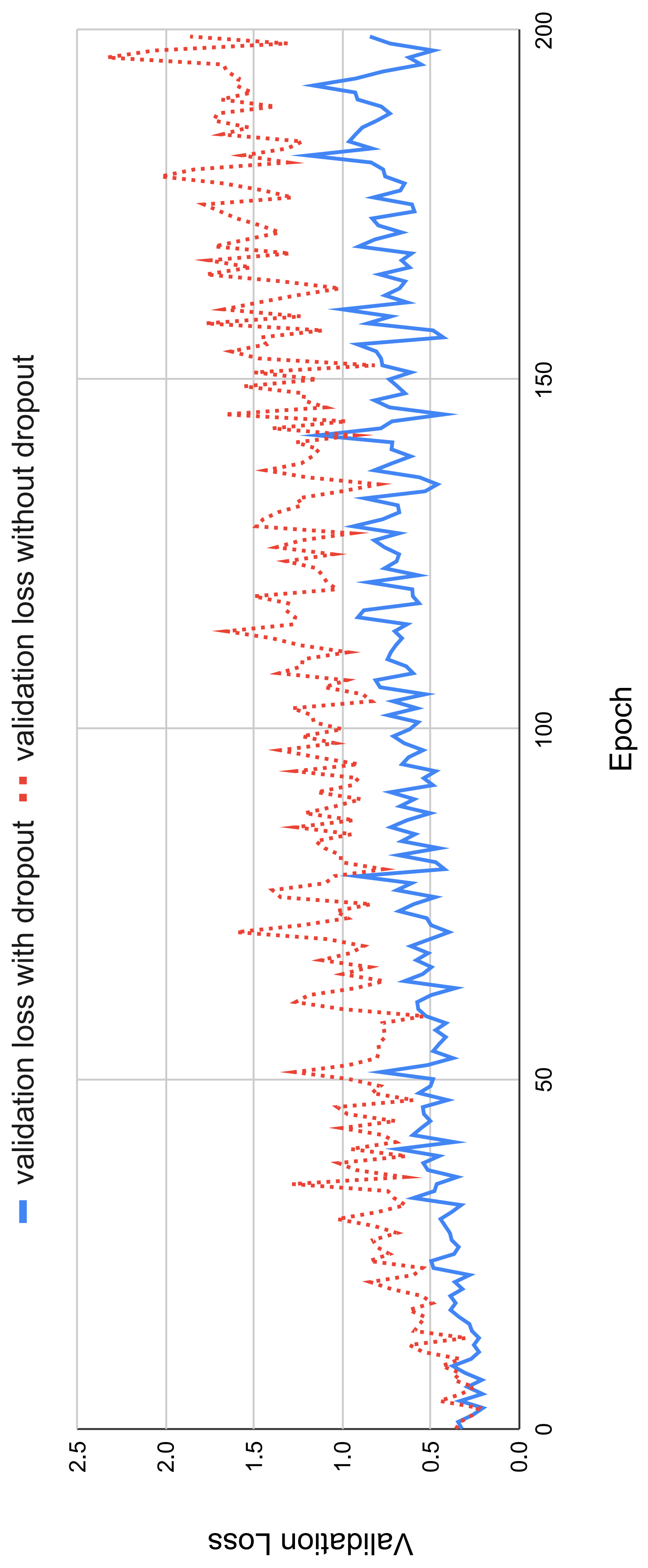


Figure 4.4: Increase in the loss value during CNN training (over-fitting) and suppression of over-fitting by drop-out.

Algorithm 1 AIRBiSNet for Pneumonia Detection

```
1: Initialize weights and biases for the convolutional layers and fully connected layers
2: Input image of size $Input_{width} \times Input_{height} \times Input_{channel}$
3: Apply a series of convolutional layers with ReLU activation and pooling layers to downsample feature maps, e.g.:
4: for $i = 1$ to $N_{conv}$ do
5:     Convolutional layer with $K_i$ filters of size $F_i \times F_i$ and stride $Stride_i$
6:     ReLU activation
7:     Max pooling layer with pool size $P_i \times P_i$ and stride $P_i$
8: end for
9: Flatten the output of the last pooling layer into a 1D vector of size $N_f$
10: Apply one or more fully connected layers with ReLU activation, e.g.:
11: for $j = 1$ to $M$ do
12:     Fully connected layer with $N_j$ neurons
13:     ReLU activation
14: end for
15: Apply a final fully connected layer with softmax activation to get class probabilities
16: Output predicted class label
```

- **Output Layer:** Utilizes softmax activation to give class probabilities. The class with the highest probability is chosen as the output.

The algorithm is calibrated using labelled chest X-ray images. During this phase, the weights and biases are optimized to enhance prediction accuracy. Once trained, AIRBiSNet can be deployed for various medical imaging tasks, with its primary function being the classification of lung X-ray images into infected or healthy.

For model training, we employed the lung X-ray image dataset from Kaggle [114, 115]. This dataset is a standard reference for comparing X-ray image classification via deep learning models. Details about the dataset are elaborated in Table 4.2. The collection comprises 21,111 X-ray images labelled as either COVID-19 (3,570 images) or non-COVID. The Non-COVID category encompasses Lung_Opacity (6,012 images), Viral_Pneumonia (1,338 images), and Normal (10,191 images).

The AIRBiSNet learning programs are implemented using Python, TensorFlow, and PyTorch. The AIRBiS pneumonia detection model was trained on a

Table 4.1: Pneumonia (COVID-19) detection accuracy with input of 256x256 (px).

Predict / Real	**Infected**	**Non-COVID**	**Sum**	**Accuracy**
Infected	(TP) 640	(FN)60	700	91.4%
Non-COVID	(FP) 7	(TN) 693	700	99.0%
Lung_Opacity	4	246	250	98.4%
Viral_Pneumonia	3	47	50	94.0%
Normal	0	400	400	100.0%
Correct Classification	640	693	1400	**95.2%**

TP: True positive. TN: True negative.
FP: False positive. FN: False negative.

deep learning server that runs Ubuntu 18.04.6 LTS (Bionic Beaver), with Linux Kernel 5.4.0-77-generic, 64GB DDR4 RAM, and NVIDIA GeForce RTX 2060 SUPER Graphics Card (8GB GDDR6 RAM).

Table 4.2: AIRBiS chest X-ray image dataset specification.

Label	**Class**	**Train**	**Test**
COVID	COVID	2870	700
	COVID (Augmented)	14,349	-
Non-COVID	Normal	9791	400
	Lung_Opacity	5762	250
	Viral_Pneumonia	1288	50
Sum		34,060	1400

Table 4.1 displays the confusion matrix for the prediction outcomes, indicating an overall accuracy of 95.2%. The class-specific accuracies are 91.4% for COVID and 99.0% for Non-COVID. AIRBiS's performance was benchmarked against several state-of-the-art studies, all employing CNNs on COVID-19 datasets as given by Equation 5.5. The achieved accuracy of 95.2% outperforms [118] (Abbas et al., 2021), [119] (Che Azemin et al., 2020), and [120] (Lin and Lee, 2020). A potential reason could be the dataset size and the imbalance in COVID-19 image counts in these studies. While [121] (Ohata et al., 2020), [122] (Apostolopoulos and Mpe-

siana, 2020), and [123] (Jain et al., 2021) reported higher accuracy, their smaller datasets might lead to reduced accuracy when applied to larger datasets.

The real-time capabilities were assessed by recording the inference time per X-ray image, detailed in Figure 4.6. Figure 4.5 juxtaposes original X-ray images with their Grad-CAM-based heatmaps, highlighting the regions where the AIRBiS neural network focuses during its analysis.

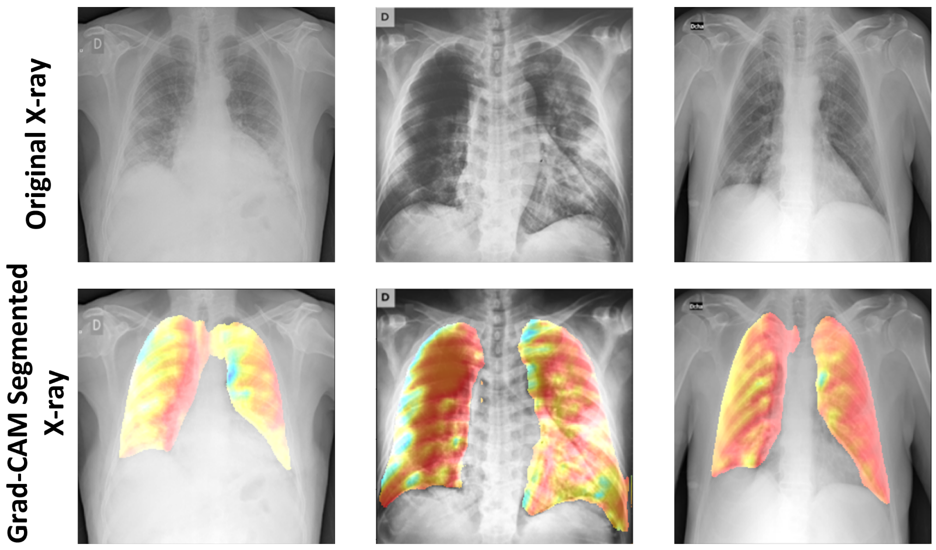


Figure 4.5: Grad-CAM-based visualization of X-ray images.

4.3 Distributed Collaborative Machine Learning Approach

Distributed collaborative learning's framework is depicted in Figure 4.7. Training network models using datasets from multiple hospitals can enhance diagnostic accuracy. Yet, patient data confidentiality often prevents some hospitals from sharing their actual data. The collaborative learning architecture counters this by facilitating the aggregation of a global model without requiring the sharing of real datasets. Consequently, patient data privacy remains intact.

In this setup, each hospital's client (be it a computer or an AI chip cluster) locally trains a neural network model using its data. Post training, the server

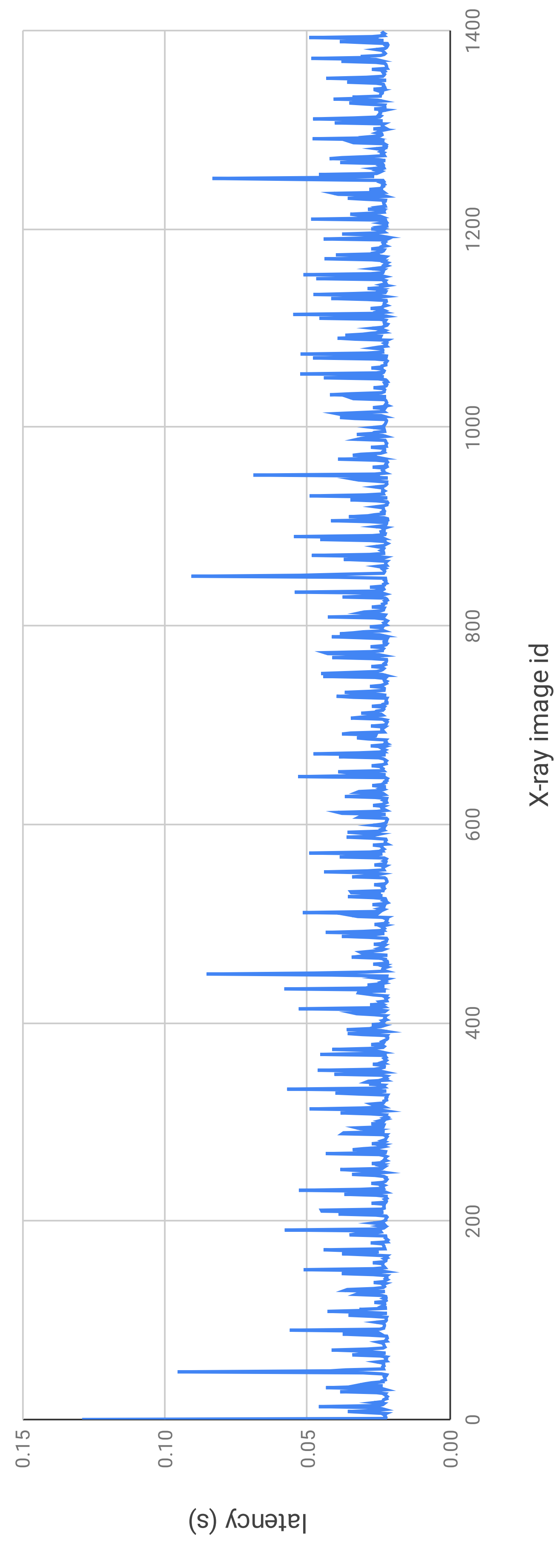


Figure 4.6: AIRBiS system latency.

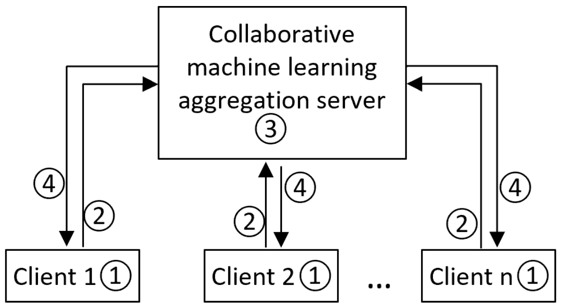


Figure 4.7: Robust collaborative learning for privacy preservation in the proposed system.

combines these local models to form a global one, which is then shared with all edge clients for local model updates. This process is iteratively conducted in several rounds until the model converges. Here's a summary of the main steps for each round:

1. Each hospital's client trains its local model based on its dataset. Within these models, the gradient $\nabla \boldsymbol{g_L}$ is computed as:

$$\nabla \boldsymbol{g_L} = \frac{\boldsymbol{\delta} E\left(\boldsymbol{W}\right)}{\boldsymbol{\delta} W} \tag{4.2}$$

2. These local gradients are then sent to the central server.

3. The server combines the received gradients to establish a global gradient $\nabla \boldsymbol{g_G}$, represented by:

$$\nabla \boldsymbol{g_G} = \frac{1}{\boldsymbol{n}} \sum_{\boldsymbol{i}=1}^{\boldsymbol{n}} \nabla \boldsymbol{g_L^i} \tag{4.3}$$

4. Edge AI clients, upon obtaining the global gradient from the server, adjust their parameters using:

$$\boldsymbol{W}^{r+1} = \boldsymbol{W}^{r} - \eta \nabla \boldsymbol{g_G} \tag{4.4}$$

$$\boldsymbol{b}^{r+1} = \boldsymbol{b}^{r} - \eta \nabla \boldsymbol{g_G} \tag{4.5}$$

Here, $\boldsymbol{W}^r$ and $\boldsymbol{b}^r$ represent the weights and bias during the r^{th} training round, while η signifies the learning rate.

The proposed neural network detection model is CNN-based and is described in Figure 5.5. The X-ray images are fed to the CNN for feature extraction and the final layer outputs probabilities for each class. The input is X-ray images which are either in grayscale or RGB. The CNN model also comprises three three convolution layers with a kernel size of 3×3, 32 biases, and $ReLU$ activation function. A Max pooling layer is used after every convolution layer, two fully connected layers are used after the flatten operation, and the final layer outputs probabilities that are assigned to two classes (Infected and Non-infected). Two dropout layers lie between the max-pooling and fully connected layers to suppress the network from over-fitting. Dropout in the neural network improves the performance by randomly disabling some neurons' activation in each layer during training. This method reduces loss due to over-training during learning. At the output layer, the $softmax$ activation function is used to estimate the probability of output classes [112]. Finally, the loss function cross-entropy is described in Equation 4.6:

$$E\left(\boldsymbol{w}_0, \boldsymbol{b}_o\right) = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{y}_i \times \log\left(\boldsymbol{y}'_i\left(\boldsymbol{w}_o, \boldsymbol{b}_o\right)\right) \tag{4.6}$$

Given the network parameters $\boldsymbol{w}_o$ and $\boldsymbol{b}_o$, and using $\boldsymbol{y}$ to represent the real label and $\boldsymbol{y}'$ for the predicted label, we optimize parameters and make classification decisions by minimizing the loss function through the stochastic gradient descent and back-propagation algorithms. The CNN model, which takes inputs of 256×256 (px) grayscale images, is detailed in Figure 5.5.

The proposed AIRBiSNet is potentially susceptible to overfitting. To mitigate this, a dropout layer has been incorporated. Dropout enhances performance by occasionally turning off neurons during training [117]. As depicted in Figure 4.4, the dotted line represents the validation loss without dropout, which tends to increase. In contrast, the solid line, representing the AIRBiSNet with dropout, demonstrates a more favourable validation loss, suggesting effective overfitting suppression.

To gauge the efficacy of our proposed method, we executed experiments across three distinct scenarios.

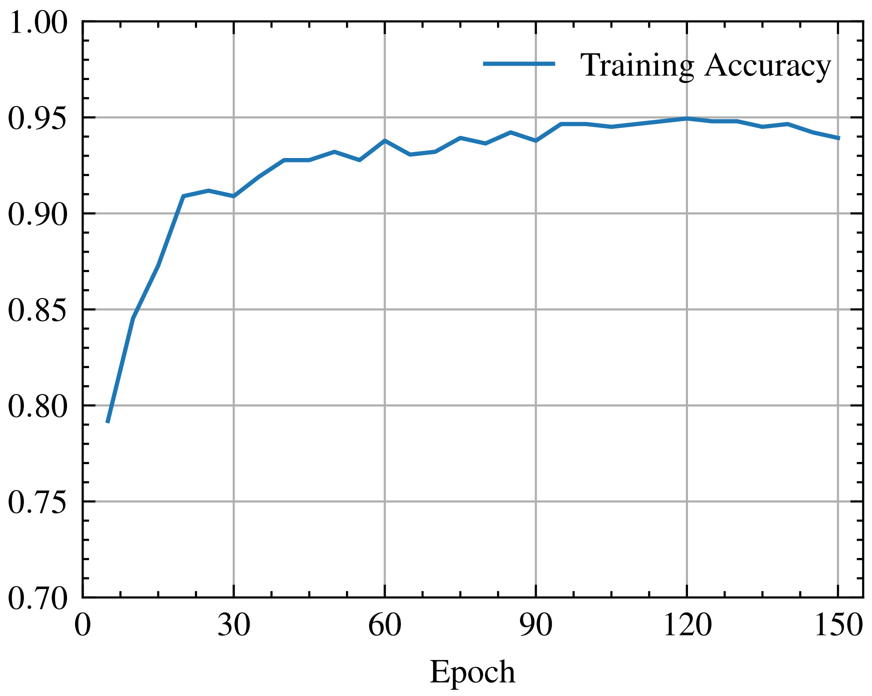


Figure 4.8: AIRBiS collaborative learning accuracy over an uneven data distribution case.

Every scenario encompassed 30 rounds, with each round consisting of five epochs. The training dataset, detailed in Table 4.2, was segmented into ten parts.

The inaugural scenario utilized a solitary learning node that processed all training datasets for conventional centralized training. In the second scenario, ten nodes were employed, each handling an evenly divided subset of the training data. This setup assessed the influence of training node counts and dataset dimensions. The third scenario mirrored the node count of the second but acknowledged the existence of domain disparities across clients, typically found in real-world instances, making its results more indicative.

Centralized training achieved an accuracy of 95.2%, as highlighted in Table 4.3. The outcome of the second scenario can also be viewed in the same table. As the node count and data volume amplify, the resultant model's precision incrementally elevates, nearing the centralized training's efficiency. The training trajectory

Table 4.3: Accuracy of AIRBiS privacy-preserving collaborative learning method over various clients.

Concentrated learning	**Client and dataset**	Learn from all (100%) dataset				
	Accuracy (%)	0.952				
Distributed collaborative learning	**Client number**	1	2	3	4	5
	Ratio	10%	20%	30%	40%	50%
	Accuracy (%)	0.882	0.899	0.896	0.916	0.915
	Client number	6	7	8	9	10
	Ratio	60%	70%	80%	90%	100%
	Accuracy (%)	0.928	0.939	0.942	0.949	0.949

for the third scenario, characterized by an uneven data split, culminated in an accuracy of 94.7%, as depicted in Figure 4.8.

While centralized training mandates data amalgamation, triggering potential privacy concerns, distributed collaborative learning sidesteps this by merging weights from disparate learning nodes.

4.4 A Robust Client Selection Based Secure Collaborative Learning Algorithm

Pneumonia, a global health concern, especially affects vulnerable groups like children and the elderly. Its timely detection and treatment are vital. Yet, traditional diagnosis, leaning heavily on clinical assessments and imaging interpretations, can be error-prone and inefficient. Recent advancements in AI, especially deep learning, offer a transformative solution. These systems can discern intricate patterns in medical images, possibly catching early-stage pneumonia that human observers might miss. However, while AI holds promise for improved healthcare, it also poses privacy risks. Many AI applications involve sending patient data to centralized servers for model training, exposing potential security vulnerabilities. Collaborative learning, which supports distributed data analysis without revealing raw data, seemed like a promising solution.

However, as illustrated in Figure 1.1, collaborative learning isn't immune to challenges. It's at risk from model poisoning attacks, where malicious contributors introduce deceptive model updates. Such disruptions could drastically affect learning accuracy, a grave concern in healthcare where incorrect diagnoses can be disastrous.

Our proposed solution is the Client Selection Secure Collaborative Learning (CSSCL) algorithm, an extension of our earlier AIRBiS work [124, 112, 125]. CSSCL is designed to counter model poisoning by employing the Minkowski similarity metric. This measure evaluates local models, identifying and sidelining those that appear manipulated. This mechanism ensures data protection and robust learning in AI-based medical systems, addressing the significant privacy and security hurdles facing collaborative learning's broader application in healthcare. Distinctly, our Client Selection Secure Collaborative Learning (CSSCL) algorithm provides a comprehensive solution. Beyond addressing prevailing security and privacy concerns in AI-driven biomedical systems, it employs the Minkowski similarity to evaluate and eliminate potentially tainted local models. Doing so, it not only maintains the integrity of the aggregated model but also fortifies data and model privacy.

4.4.1 Problem Formulation

In a collaborative network comprised of N_{client} client nodes, each node possesses its own local dataset. In a typical learning cycle, every active client i refines its local model parameters using its dataset, producing an updated model M_i. These models are then transmitted to the central server for aggregation, and the consolidated model is distributed back to the clients.

However, potential threats arise when certain clients, constituting P_{attack} percent of the network, dispatch adulterated models during a poisoning attack. If the server doesn't recognize these tainted updates, the aggregated model's efficacy diminishes.

Our objective is to devise a strategy to detect and discard these compromised models, optimizing the final model's performance.

4.4.2 Client Selection Based Secure Collaborative Learning

The Client Selection based Secure Collaborative Learning (CSSCL) algorithm we propose refines the conventional collaborative learning method. As depicted in Figure 4.9, our enhancement involves a specialized step to discern and exclude tampered models. Here are the primary components of the CSSCL procedure:

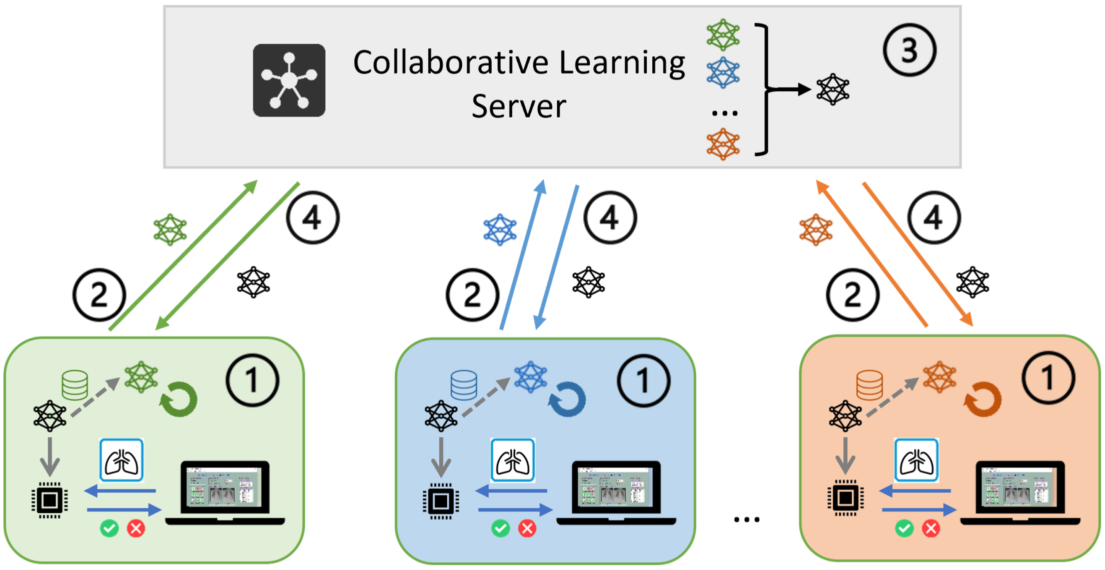


Figure 4.9: Collaborative learning approach for pneumonia detection. This is the overview of the collaborative learning approach, which can be briefly divided into four steps. First, each client trains a local model. Following that, local models are uploaded to the server. Then, on the server side, local models are aggregated into a global model. Finally, each client downloads the global model from the server.

- Client Training: During the initial phase of a collaborative learning round, each client node i refines its local model based on its dataset to generate an update M_i.

- Client Selection: Post-training, every client determines the Minkowski similarity Sim_i between its updated model M_i and the previous global model M_g. This metric provides an insight into the reliability of the model; a higher score suggests a more trustworthy model. Both the similarity score Sim_i and the model update M_i are transmitted to the server.

- Model Aggregation: The server, upon obtaining the updates, classifies the clients using their similarity scores. Only the models from the top Q percent of clients (where Q is an adjustable parameter) are amalgamated to construct a new global model. The server excludes models from the remaining clients due to their potential as poisoning threats.

The CSSCL method, with its selective approach, significantly diminishes the risk of poisoning attacks on the final model. The use of the Minkowski similarity metric ensures accurate differentiation between authentic and manipulated models.

4.4.3 CSSCL with Server Model

Refer to Algorithm 2 and Flowchart 4.10 for detailed visualization. The process begins by initializing and refining the local models. These models are then relayed to the server, which evaluates the similarity between the central model and each local version. Models that closely align with the server model are chosen for the aggregation phase, culminating in the creation of a consolidated global model.

Algorithm 2 outlines the Client Selection Based Secure Collaborative Learning using a server model, M_{server}. This algorithm aims to collate a subset of the most relevant local models, M^i_{local}, for aggregation, chosen from the entire pool of client models.

For the given total number of clients, N_{client}, and a fraction parameter, α, the process begins by initializing an empty list, $M_{selected}$, to hold the chosen local models. The server model's parameters are represented as a one-dimensional array, P_{server}, and each local model's parameters as P^i_{local}.

During the algorithm's main loop, which iterates over each client $i \in N$, it computes the similarity between P_{server} and P^i_{local} using the Minkowski distance,

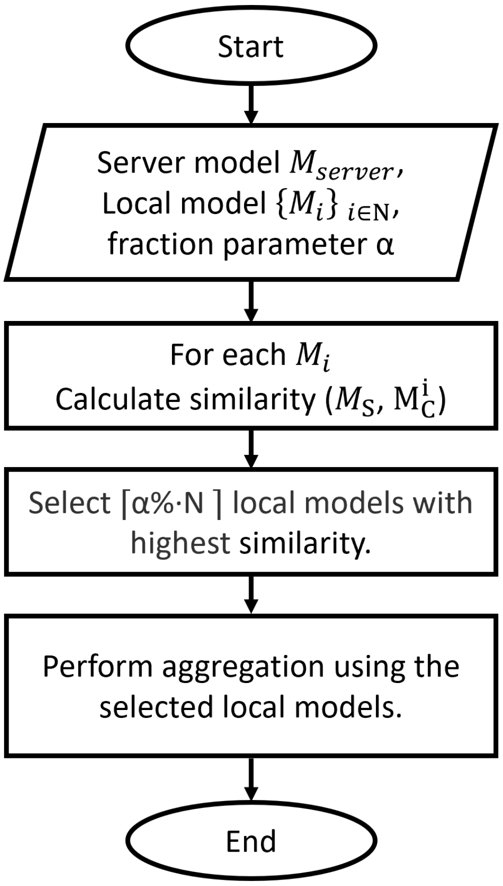

Start
Server model M_{server},
Local model $\{M_i\}_{i\in N}$,
fraction parameter α
For each M_i
Calculate similarity (M_S, M_C^i)
Select ⌈α%·N ⌉ local models with highest similarity.
Perform aggregation using the selected local models.
End

Figure 4.10: Client selection based secure collaborative learning with server model.

which is stored in $Sim_{server,i}$.

Once the loop concludes, the top $\lceil \alpha\% \cdot N \rceil$ models, those with the highest similarity scores to the server model, are added to $M_{selected}$. This curated list is the algorithm's final output, ensuring aggregation efficiency by emphasizing only the most pertinent models.

Minkowski distances generalize various distance metrics. For two points P_1 and P_2, the Minkowski distance is computed as:

$$D_{minkowski}(P_1, P_2) = \left(\sum_{i=1}^{n} (x_i - y_i)^r \right)^{1/r} \tag{4.7}$$

If $r = 1$, the Minkowski distance is converted to a Manhattan distance; if $r = 2$, the Minkowski distance is converted to a Euclidean distance; if $r = \infty$, the Minkowski distance is converted to a Chebyshev distance.

Algorithm 2 Client Selection Based Secure Collaborative Learning Algorithm (with server model)

```
Require: Server model $M_{\text{server}}$, local models $\{M^i_{\text{local}}\}_{i \in N}$, the total number of clients $N_{client}$, and fraction parameter $\alpha$
Ensure: List of selected models for aggregation
1: Initialize an empty list $\{M_{selected}\}$, which is used to store the selected local models
2: Store all parameters of $M_{\text{server}}$ as a one-dimensional array, denoted by $P_{\text{server}}$
3: Store all parameters of each $M^i_{\text{local}}$ as a one-dimensional array, denoted by $P^i_{\text{local}}$
4: for each $i \in N$ do
5:     Calculate the similarity between $P_{\text{server}}$ and $P^i_{\text{local}}$ using the Minkowski distance, denoted by $Sim_{\text{server},i}$
6: end for
7: Select $\lceil \alpha\% \cdot N \rceil$ models with highest similarity and store them to the list $M_{\text{selected}}$
8: return $M_{\text{selected}}$
```

4.4.4 CSSCL without Server Model

In the absence of a pre-existing server model, Algorithm 3 presents a client selection based secure collaborative learning approach. Refer to Flowchart 4.11 for a visual overview.

1. Initialization and Training: Local models are initialized and trained.

2. Model Upload: Trained local models are uploaded to the server.

3. Model Selection: The algorithm computes the similarity between each pair of local models using their parameters, P^i_{local}, represented as a one-dimensional array. Afterwards, it calculates the average similarity for each model. Those with the highest average similarity are earmarked for aggregation.

4. Aggregation: The selected models are combined to form a global model.

Within the algorithm, two primary loops iterate over every client pair ($i, j \in N$). In these loops, the Minkowski distance gauges the similarity between local model parameters, denoted as $Sim_{i,j}$. After determining the average similarity for each model, the top $\lceil \alpha\% \cdot N \rceil$ models are added to the list, $M_{selected}$.

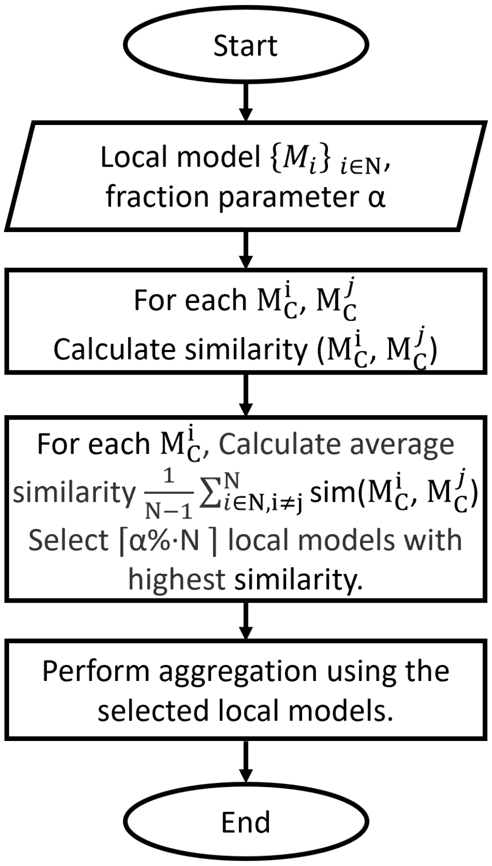

Start
Local model $\{M_i\}_{i \in N}$, fraction parameter α
For each M_C^i, M_C^j Calculate similarity (M_C^i, M_C^j)
For each M_C^i, Calculate average similarity $\frac{1}{N-1}\sum_{i \in N, i \neq j}^{N} sim(M_C^i, M_C^j)$ Select ⌈α%·N⌉ local models with highest similarity.
Perform aggregation using the selected local models.
End

Figure 4.11: Client selection based secure collaborative learning without server model.

The output, $M_{selected}$, consists of models that, on average, are most representative of the entire dataset. This ensures a strategic and efficient aggregation process, aiming to bolster collaborative learning outcomes.

4.5 Evaluation

Evaluation forms an essential part of any research methodology in the realm of machine learning and artificial intelligence. It is through this process that we are able to measure and validate the performance and robustness of the proposed models and algorithms. In the following subsections, we delve into the specifics of our evaluation methodology, describe the implementation tools and benchmarks we used for the experiments, and present the results we obtained from the evaluation process.

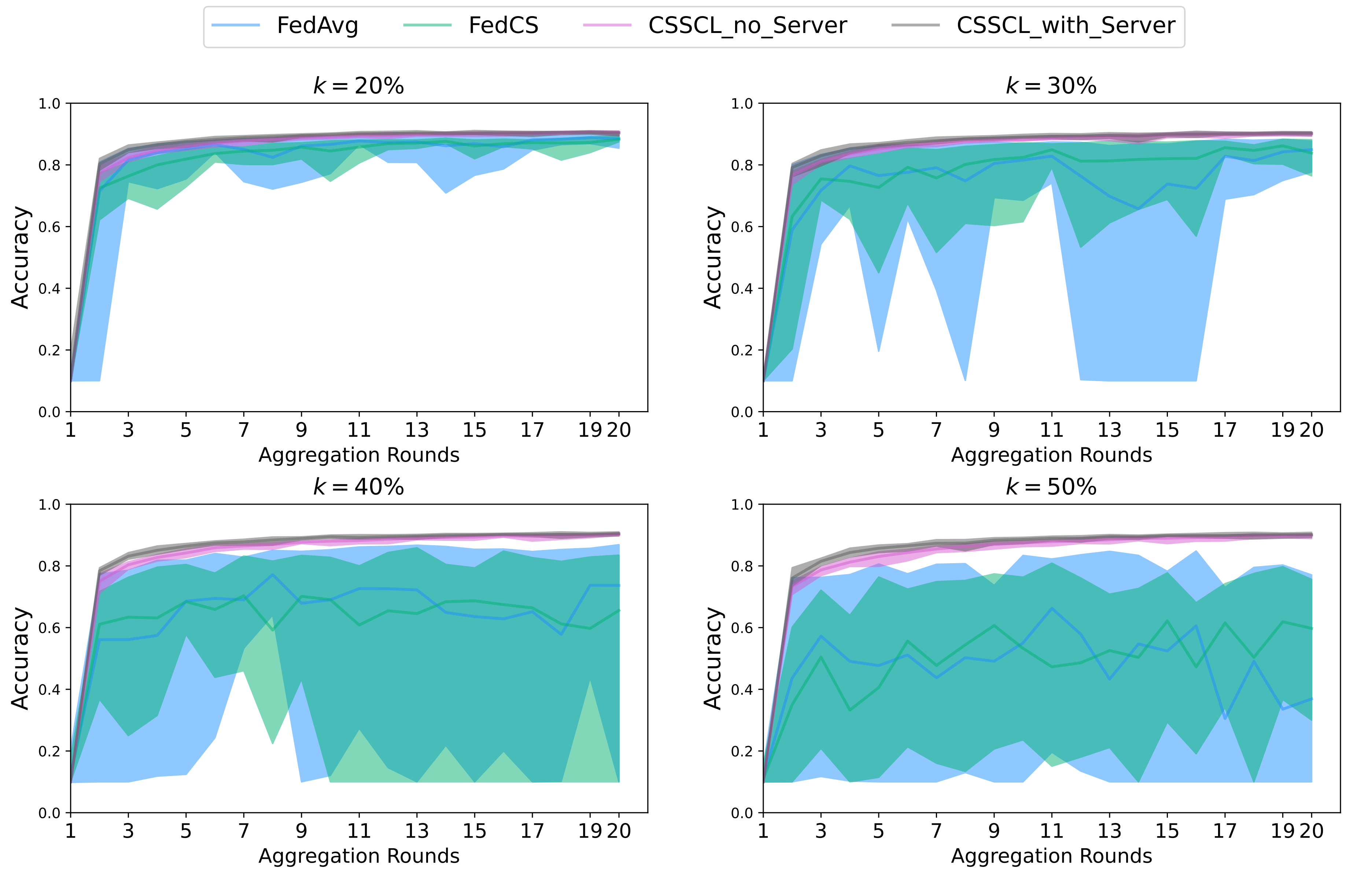
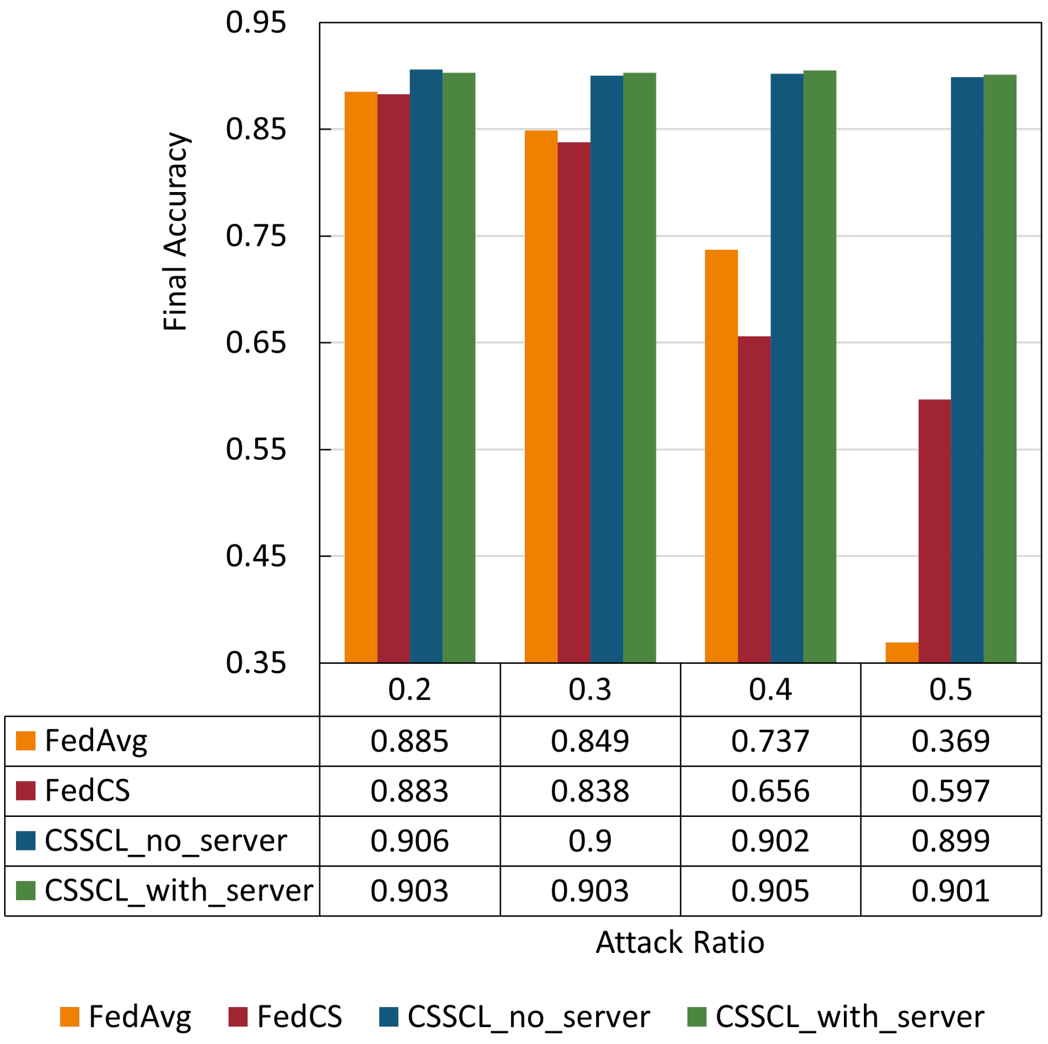


Figure 4.12: Comparison among FedAvg, FedCS, and the proposed CSSCL algorithm under client attacks. The experiment considers variations in the intensity of the attack. The shaded areas signify performance variability. The result of the proposed CSSCL approach is in red and grey colour. The x-axis denotes the number of training rounds, and the y-axis denotes the accuracy.



	0.2	0.3	0.4	0.5
🟧 FedAvg	0.885	0.849	0.737	0.369
🟥 FedCS	0.883	0.838	0.656	0.597
🟦 CSSCL_no_server	0.906	0.9	0.902	0.899
🟩 CSSCL_with_server	0.903	0.903	0.905	0.901

Figure 4.13: Comparison of final accuracy among FedAvg, FedCS, and the proposed CSSCL algorithm under client attacks. The experiment considers variations in the intensity of the attack. When 20 to 50 percent of clients are attacked, the proposed CSSCL method achieves significantly better results. In addition, when the number of attacks increases to 50 percent. The performance of the proposed algorithm is still robust. While other approaches behave even worse.

Algorithm 3 Client Selection Based Secure Collaborative Learning Algorithm (without server model)

```
Require: Local models $\{M^i_{\text{local}}\}_{i \in N}$, the total number of clients $N_{client}$, and fraction parameter $\alpha$
Ensure: List of selected models for aggregation
1: Initialize an empty list $\{M_{selected}\}$, which is used to store the selected local models
2: Store all parameters of each $M^i_{\text{local}}$ as a one-dimensional array, denoted by $P^i_{\text{local}}$
3: for each $i \in N$ do
4:     for each $j \in N$ do
5:         Calculate the similarity between $P^i_{\text{local}}$ and $P^j_{\text{local}}$ using the Minkowski distance, denoted by $Sim_{i,j}$
6:     end for
       Calculate the average similarity of $M^i_{\text{local}}$
7: end for
8: Select $\lceil \alpha\% \cdot N \rceil$ models with the highest average similarity and store them to the list $M_{\text{selected}}$
9: return $M_{\text{selected}}$
```

4.5.1 Evaluation Methodology

Evaluation is crucial in machine learning and artificial intelligence research, enabling the measurement and validation of model and algorithm performance. This section delves into our specific evaluation methods, tools, benchmarks, and the results obtained.

Our primary metric was classification accuracy, which represents the proportion of correctly identified results (true positives and negatives) out of all cases. High accuracy in our study indicates the system's ability to correctly detect pneumonia from chest X-ray images and effectively categorize Fashion MNIST dataset items. The experiments were conducted using Python 3.10.10 and TensorFlow 2.11.

We used two benchmarks to assess our method:

- *Fashion MNIST Classification:* Fashion MNIST comprises 28x28 grayscale images of 70,000 Zalando fashion items spanning 10 categories, with 7,000 images in each. Split into 60,000 training and 10,000 testing images. It offers complexity beyond MNIST while remaining compact for swift experiments.

- *Pneumonia Detection from Chest X-Ray Images:* This biomedical set contains X-ray images tagged as ‘COVID-19’, ‘Lung_Opacity’, ‘Viral_Pneumonia’, or ‘Normal’. It let us assess our method in a real-world healthcare context, emphasizing the significance of timely and precise pneumonia diagnosis.

These benchmarks shed light on our AIRBiS collaborative learning method’s adaptability. Fashion MNIST assessed its generalizability, while pneumonia detection emphasized its relevance to vital healthcare tasks.

4.5.2 Evalaution Results

Using the Fashion MNIST and lung X-ray datasets, we tested our CSSCL algorithm’s capabilities. Our collaborative learning setup had 30 client nodes, simulating poisoning attacks at rates from 20% to 50%.

We gauged the CSSCL’s efficacy against other collaborative techniques like FedAvg and FedCS, considering the ultimate model’s training accuracy as the key metric. Remarkably, in every attack setting, the CSSCL surpassed its counterparts in training accuracy, as displayed in Figure 4.12.

For instance, with a 20% attack rate, CSSCL enhanced training accuracy by about 2% relative to alternative methods. A more pronounced edge was evident as the attack rate climbed. At 50%, there was roughly a 30% leap in training accuracy with the CSSCL, as depicted in Figure 4.13.

Furthermore, when focused on pneumonia detection, CSSCL’s performance was on par with top-tier methods. Indicating resilience against poisoning attacks without compromising learning quality, CSSCL emerges as an exciting prospect for secure, efficient collaborative learning, especially in biomedical contexts.

Chapter Summary

This chapter proposed a Client Selection Secure Collaborative Learning (CSSCL) algorithm to address the problem of model poisoning attacks in collaborative learning. The algorithm uses Minkowski similarity as a metric to assess the quality of local models and select the top-performing models for aggregation. Our empirical evaluations show that the CSSCL algorithm significantly improves the final training accuracy in various poisoning attack scenarios, demonstrating its robustness against such attacks. Furthermore, we demonstrated the effectiveness of the CSSCL algorithm in a specific application scenario, namely pneumonia detection. The results show that the algorithm achieves high detection accuracy while preserving data and model privacy. Overall, the CSSCL algorithm provides an effective and secure solution for collaborative learning in AI-enabled biomedical systems. Future work can extend this approach to other application scenarios and explore more sophisticated methods for client selection and model aggregation to further enhance the robustness and performance of collaborative learning systems.

Chapter 5

Self-Contained Energy-Efficient Reconfigurable System by Leveraging an FPGA Cluster for Pneumonia Detection in Chest X-Ray Images

5.1 Hardware-Based Pneumonia Detection

AI in Healthcare

Recent advancements in Artificial Intelligence (AI) applications showcase the potential of hardware-software co-design principles across various fields [106, 126]. Particularly in healthcare, Deep Learning (DL) models have proven invaluable in anomaly detection during patient monitoring, lung ultrasonography classification, and notably in COVID-19 pandemic research efforts [127, 128, 129]. While the ubiquity of cloud computing platforms provides the computational power for DL models [130], they sometimes fall short in the real-time, secure analysis of medical data due to latency and security challenges [131, 132]. Edge computing emerges as a solution, catering to the low-latency, privacy, and security demands of modern healthcare applications [133, 134, 135].

DL on the Edge

Amid the COVID-19 pandemic, the focus on DL-based systems for rapid diagnosis on edge platforms is paramount. However, these platforms, being resource-constrained, grapple with the high computational, memory, and power needs of DL applications [136]. Strategies to optimize Deep Neural Networks (DNNs) for edge inference aim to strike a balance between reduced power, compactness and maintained accuracy [137, 138]. Field Programmable Gate Arrays (FPGAs) emerge as promising candidates for edge deployment due to their adaptability, cost-effectiveness, and energy efficiency [139]. Nevertheless, with the increasing complexity of DNNs, ensuring their deployment on single FPGAs becomes challenging. For scalable edge inference, a holistic platform is needed that harmonizes performance, accuracy, and efficiency.

COVID-19 Pandemic Outlook

By August 2022, the world witnessed over 578 million COVID-19 cases and more than 6 million fatalities [140]. Efficient and rapid diagnostic methods stand central to managing and potentially curtailing the pandemic.

The standard method for COVID-19 detection is the Reverse Transcription Polymerase Chain Reaction (RT-PCR), which identifies the genetic materials of the SARS-CoV-2 virus in upper respiratory specimens from patients. Though the test's sensitivity is cited to range from 60% to 97% [108], variability among patients can lead to a lower range of 60%-71% [109], causing a notable rate of false negatives. While RT-PCR has the capacity for batch testing, each sample requires manual collection from the individual being tested.

An alternative approach involves analyzing lung X-ray images, which achieves accuracy levels between 80-90% [110]. This method demands doctors to meticu-

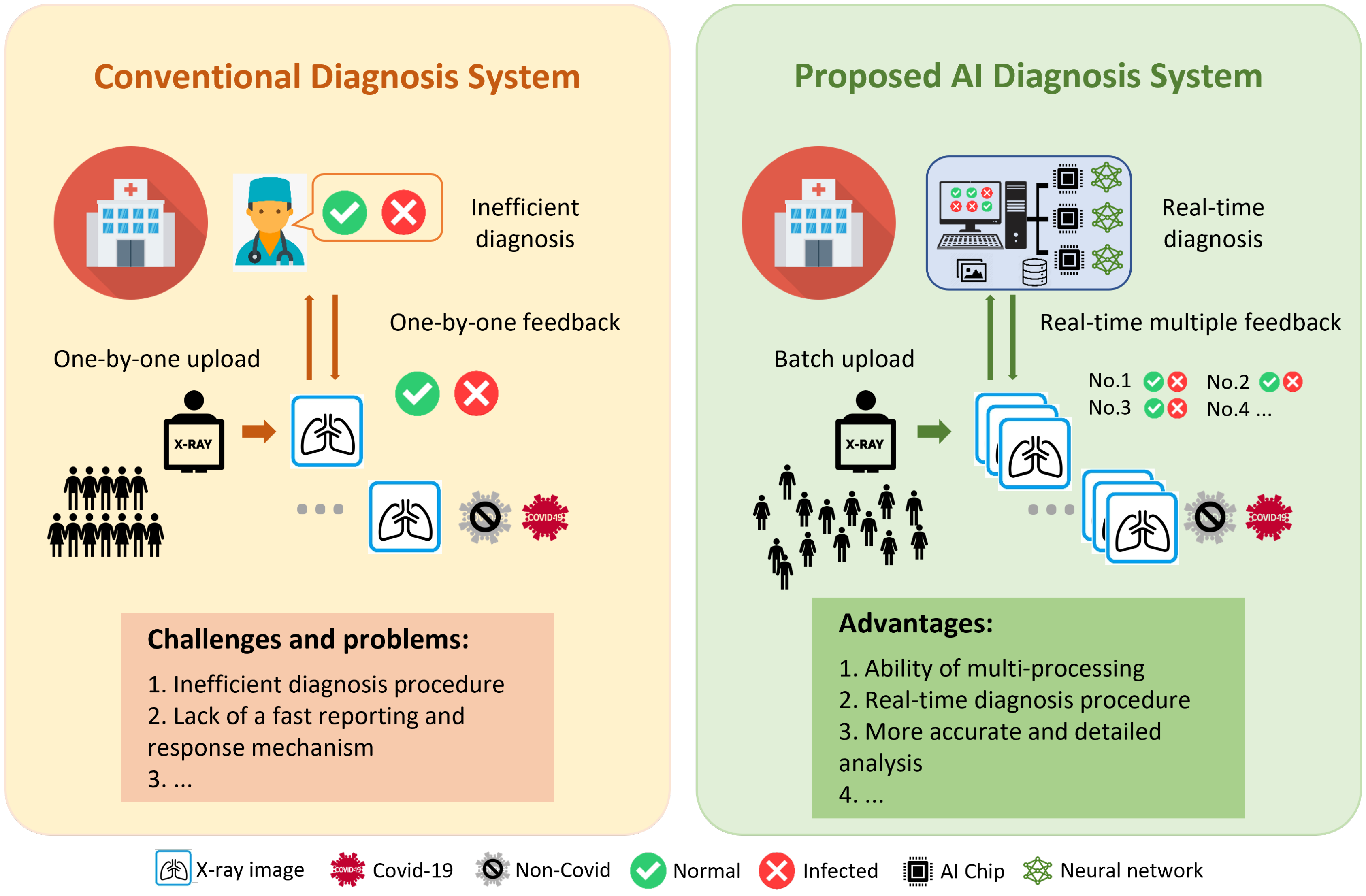

Conventional Diagnosis System
Inefficient diagnosis
One-by-one feedback
One-by-one upload
X-RAY
Challenges and problems:
1. Inefficient diagnosis procedure
2. Lack of a fast reporting and response mechanism
3. ...
Proposed AI Diagnosis System
Real-time diagnosis
Real-time multiple feedback
Batch upload
X-RAY
No.1
No.2
No.3
No.4 ...
Advantages:
1. Ability of multi-processing
2. Real-time diagnosis procedure
3. More accurate and detailed analysis
4. ...
X-ray image
Covid-19
Non-Covid
Normal
Infected
AI Chip
Neural network

Figure 5.1: Conventional and proposed pneumonia detection/ diagnosis systems. In the proposed AIRBiS, a batch of X-ray images is uploaded, and a detection procedure is provided in FPGAs with high-speed computation and low power consumption. The feedback is provided using an interactive user interface with more accurate and detailed analysis in hospitals.

lously assess lung X-rays and integrate them with clinical presentations to diagnose. But given the surging COVID-19 cases, this method struggles with timely reporting and response. Beyond this, ensuring patient privacy and security during this process remains paramount.

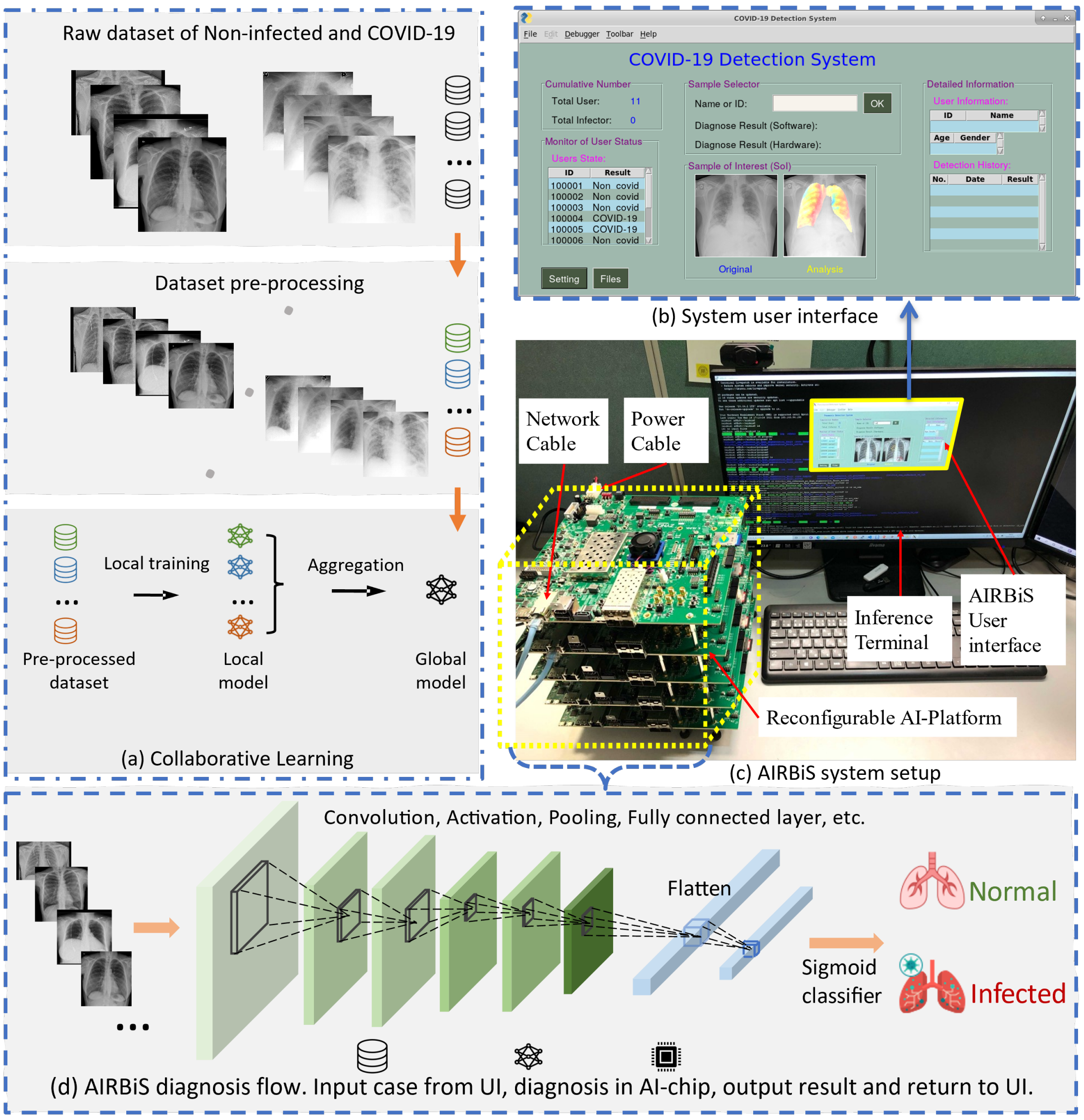
Computer-aided diagnosis systems, especially those built on DNNs, have emerged as promising solutions [2, 77]. Yet, many medical establishments lack the infrastructure for large-scale deployments of these systems. Moreover, traditional biomedical information security measures often don't align with distributed learning mechanisms. This disparity hampers the collective improvement of diagnostic models across institutions and the ability to swiftly relay test results and coordinate with government bodies. These challenges have spurred clinicians and researchers to seek alternative or supplementary methods for enhancing COVID-19 detection.

Furthermore, in scenarios with rampant infections, features like rapid batch inference and low-power consumption are vital for the expansive implementation of computer-aided diagnosis tools. Training these tools often demands centralized data, which can jeopardize patient privacy. Securing patient consent is not only cumbersome but could also spur legal complications.

Our research introduces AIRBiS, a pioneering hardware-software co-design. This system employs an FPGA cluster and a 3-dimensional (3D) network on chip interconnect, aiming to deliver low-latency, energy-efficient DL inference at the edge. Additionally, we present the AIRBiSNet algorithm, a unique DL method for pneumonia detection via chest X-ray images. This work, to our knowledge, is the inaugural attempt at harnessing FPGA for COVID-19 detection.

5.2 Inference Acceleration

Artificial Intelligence (AI) offers powerful solutions to pressing issues across various domains, notably in healthcare. Here, the emerging demands of deep



ID	Result
100001	Non covid
100002	Non covid
100003	Non covid
100004	COVID-19
100005	COVID-19
100006	Non covid

Figure 5.2: AIRBiS Architecture Overview: (a) The raw dataset, comprising Non-infected and COVID-19 samples, is pre-processed. It undergoes separate training on clients and then aggregates to produce a global model. (b) The system user interface showcases the X-ray image case, the sample of interest, diagnosis outcomes, result statistics, and user database. It also provides connectivity to the edge inference accelerator. (c) The AIRBiS setup encompasses a monitor that displays the system interface, a standard keyboard mouse set for operations, and a Zynq Ultra Scale+ MPSoC ZCU102 FPGA cluster facilitating edge inference acceleration. (d) The diagnosis flow of AIRBiS involves inputting a case via the UI, processing the diagnosis in the AI chip, and finally, outputting and presenting the diagnosis result back on the UI.

learning applications necessitate platforms that effectively address latency, security, and power consumption challenges. In this context, we introduce AIRBiS, a reconfigurable self-contained hardware platform tailored for scaling deep learning inference in detecting pneumonia through chest X-ray images.

The conventional diagnostic approach, depicted in Figure 5.1, involves patients visiting hospitals, undergoing X-rays, and awaiting physician interpretations. This method often results in overcrowded facilities, overworked professionals, and delayed results. Our solution streamlines this process: X-ray images are digitally captured and directly uploaded to the local AIRBiS system for automated detection and diagnosis. Physicians only intervene when necessary, alleviating undue burdens.

AIRBiS's computational core, detailed in Figure 5.3, hinges on a reconfigurable AI platform. Comprising five Zynq Ultra Scale+ MPSoC ZCU102 FPGA boards, it offers a robust, scalable architecture interconnected via secure Ethernet. This arrangement allows for the integration of remote nodes into larger clusters as computational needs escalate, ensuring continued operation even if an active node encounters issues.

Physicians interact with AIRBiS through a user interface, also shown in Figure 5.3, that presents comprehensive infection statistics, detailed diagnosis results, and samples of interest (SOI). We further offer a companion application for host PCs, facilitating a seamless connection to the AIRBiS edge inference accelerator. This interface showcases the X-ray image, the SOI, diagnosis outcomes, and corresponding statistics.

For extensive datasets, while GPUs enhance the neural network's inference time and are typically employed on high-performance cloud servers, they fall short in energy efficiency due to their elevated power demands. In contrast, a configurable AI chip system conserves bandwidth, diminishes application latency, and is more power-efficient. We employ an FPGA-based cluster in our approach to optimize inference scaling and further reduce power usage.

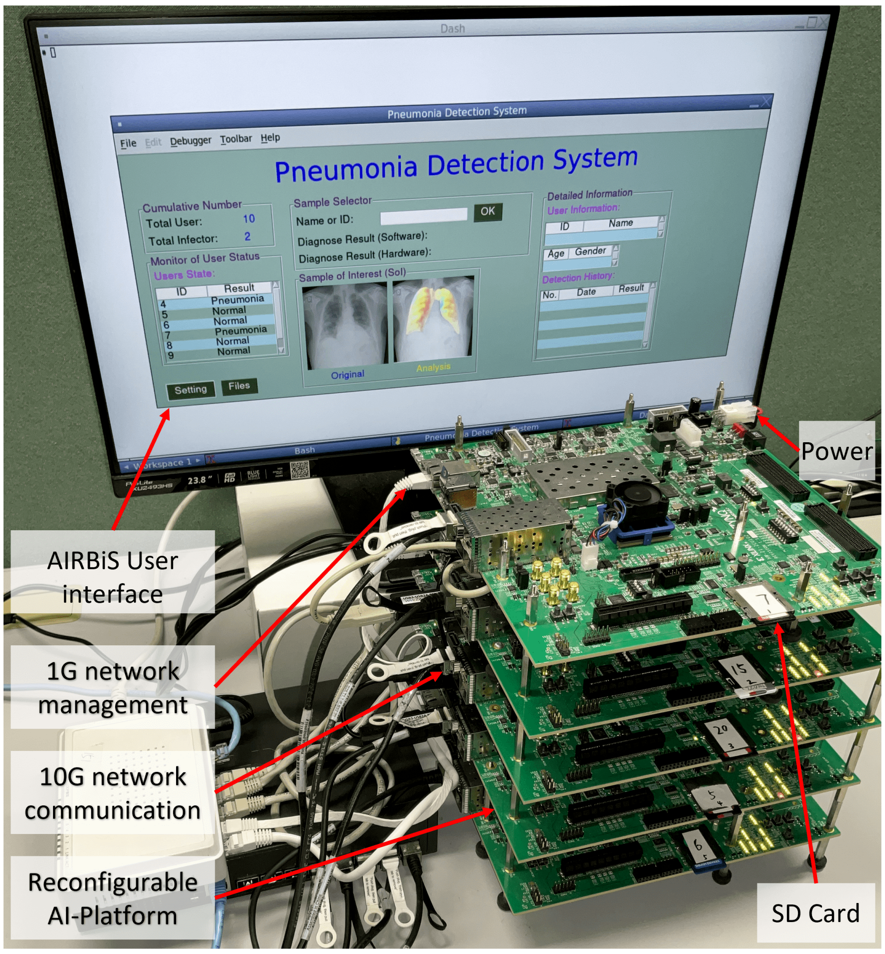

Pneumonia Detection System
AIRBiS User interface
1G network management
10G network communication
Reconfigurable AI-Platform
Power
SD Card

Figure 5.3: AIRBiS scaling and reconfigurable FPGA-cluster-based deep-learning inference platform.

5.2.1 AIRBiSNet Complexity

We analyzed the performance of the CNN model, focusing on its individual components. The FLOPs (FLoating-point OPerations) metric measures the multiply-add operations in convolutional neural networks. The cumulative time complexity of a CNN, primarily dictated by its convolutional layers, is represented in Equation 5.1.

$$\textbf{Time}_{CNN} \sim O\left(\sum_{l=1}^{D} M_l^2 \cdot K_l^2 \cdot C_{l-1} \cdot C_l\right) \tag{5.1}$$

Where:

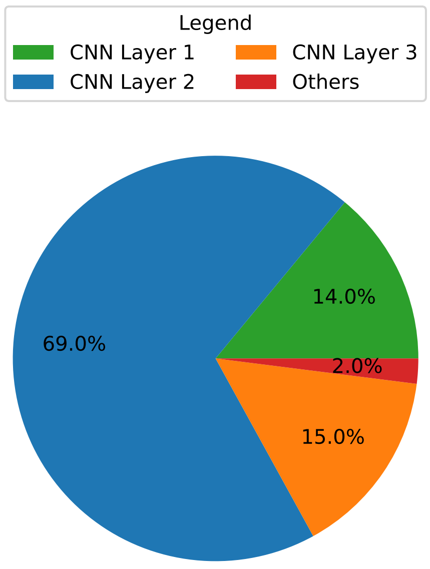
- D denotes the total convolutional layers in the neural network.
- l refers to the lth convolutional layer.
- C_l stands for the number of output channels of layer l.
- C_{in}, for layer l, is the output channels of the $(l-1)$th layer.

The execution times of FC (Fully Connected) and pooling layers, which contribute 5-10% to the total computation, are excluded from this equation. Memory utilization in the CNN is governed by the model's parameters and each layer's output feature map, detailed in Equation 5.2. Here, K represents the convolution kernel size, C is the channel count, and M is the output feature map size.

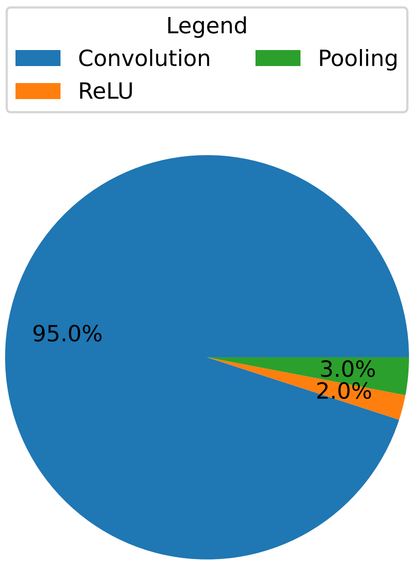
$$\textbf{Space}_{CNN} \sim O\left(\sum_{l=1}^{D} K_l^2 \cdot C_{l-1} \cdot C_l + \sum_{l=1}^{D} M^2 \cdot C_l\right) \tag{5.2}$$

We analyzed the execution time and memory utilization of neural networks, focusing on the AIRBiSNet and its edge FPGA-based low-power inference cluster. In the AIRBiS inference, as depicted in Figure 5.5, there are three CNN layers interspersed with operations like model loading, flattening, forwarding, and sigmoid. Each layer incorporates Convolution, ReLU, and Pooling computations.

The time distribution of AIRBiS neural network inference is shown in Figure 5.4(a). The execution times for the first three layers are 14%, 69%, and 15%,



(a) The profiling results of CNN Layer1, CNN Layer2, CNN Layer3, and others (including flattening, forward and sigmoid.)



(b) The profiling results of the Convolution, Pooling and ReLU.

Figure 5.4: Profile AIRBiS inference program with the average time cost of computation.

respectively, with additional operations consuming 2%. Notably, the second layer, given its vast input data, is the most time-consuming.

The breakdown of operations in the CNN layers, as detailed in Figure 5.4(b), reveals that Convolution consumes 95% of the time, followed by Pooling at 3%, and ReLU at 2%. The dominance of the convolution operation in computation time is attributed to its extensive multiplication operations [141].

5.2.2 Quantization Optimization

While floating-point numbers offer higher precision, their storage and computation requirements lead to extended inference times, especially as network complexity increases. To address the delay, model compression via quantization becomes imperative [142]. Quantization involves mapping floating-point numbers into fixed-point number space using a scale quantization transformation. Considering a weight data set ranging from (X_{min}, X_{max}), it can be mapped to a fixed-point data set's maximum range using an appropriate scale and offset.

$$r = S(q - Z) \tag{5.3}$$

Quantization optimization, as shown in previous studies, reduces inference time while maintaining accuracy [143]. Specifically, weight quantization optimizes memory usage and FPGA resource consumption, requiring fewer DSP blocks, LUTs, and FFs when weights are quantized into smaller bit sizes.

In the AIRBiS inference accelerator design, we employ the Xilinx Fixed-Point functions to quantize FP32 weights into signed INT8. With a 0-point offset Z and a quantization range q of [-127, 127], the scale size S for AIRBiSNet, given its weight bounds of 1.6 and -0.9, is determined to be 79. This means a weight of 1.1 translates to a quantized value of 127, while -0.8 quantizes to -71.

During convolution computation, this scale factor can be leveraged for post-processing, converting MAC operations into fixed-point operations. Thus, the convolution can be succinctly represented as:

$$y(k, l, n) = \Delta_x \Delta_w \operatorname{conv}(w_Q(k, l, m, n) - z_w, x_Q(k, l, m) - z_x) \tag{5.4}$$

where k is the output feature map, l denotes the input feature map, and (m, n) indicates the filter size.

5.3 Scaling Deep-Learning Pneumonia Detection Approach

We introduce two architectures for mapping neural networks onto FPGA clusters to optimize performance and power consumption: (a) non-pipelined inference and (b) pipelined inference.

The non-pipelined inference, as visualized in Figure 5.5 (a), offers flexibility and scalability, adjusting according to system load and ensuring fault tolerance. Conversely, the pipelined inference, depicted in Figure 5.5 (b), is apt for larger DNNs.

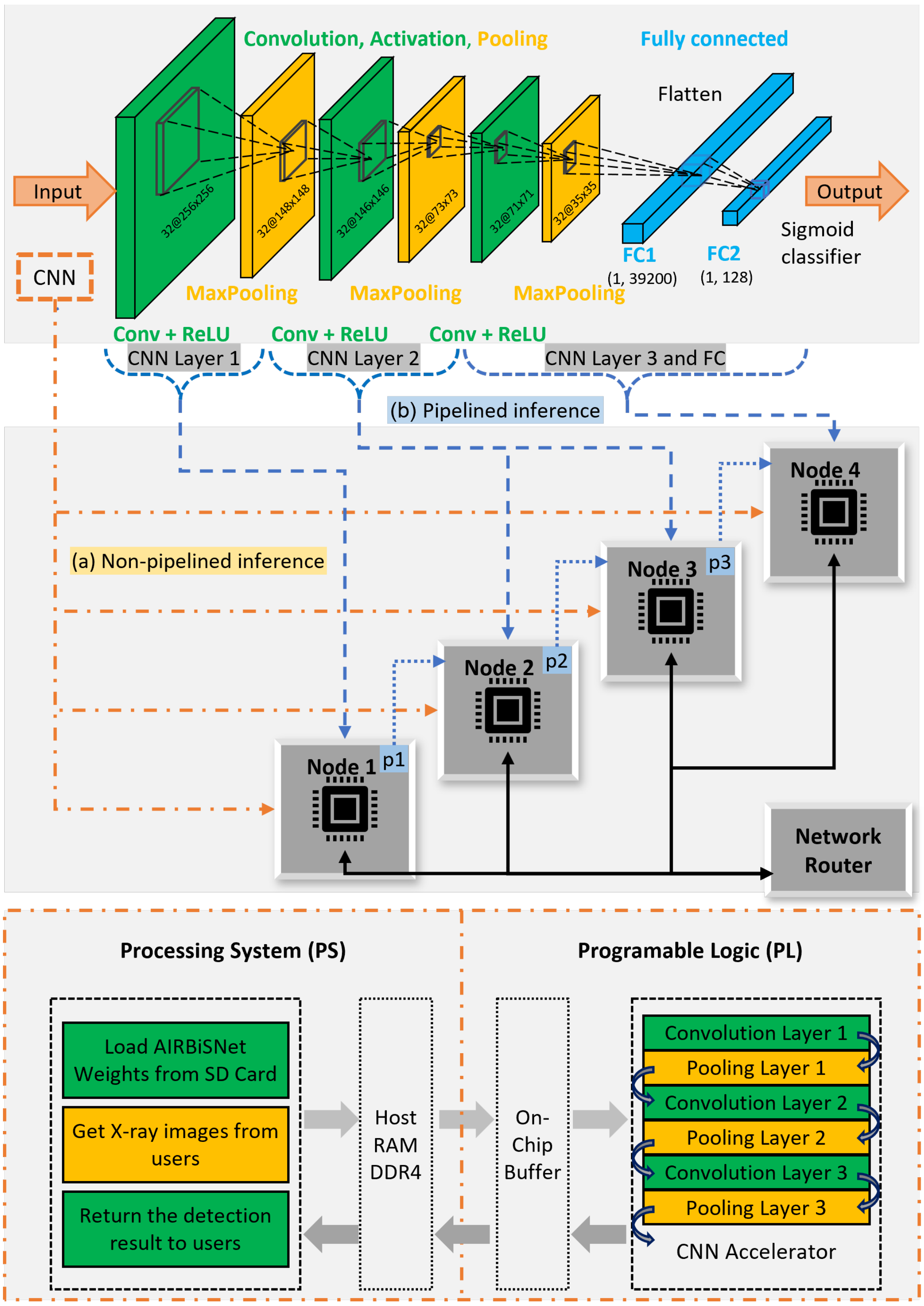


Figure 5.5: AIRBiS parallel inference and collaborative learning architecture. (a) Non-pipelined inference. (b) Pipelined inference.

Dividing the network across nodes of the logic cluster bolsters high-precision inferences, minimizes latency, and conserves power. Moreover, our proposed clusters support edge collaborative learning. Without uploading raw data, they enhance detection precision and robustness, ensuring privacy and energy efficiency.

Figure 5.5 illustrates our hardware system, grounded in a scalable, reconfigurable AI chip. This chip clusters FPGA development boards, specifically the Zynq UltraScale+ MPSoC ZCU102 Evaluation Kit. Within the AI chip, an ARM processor controls the Programmable Logic (PL) to set convolution layer parameters dynamically.

System initialization is achieved by loading weights from external memory into an on-chip buffer via DDR4 RAM. Users' X-ray image data are similarly transferred to this buffer using Direct Memory Access (DMA) and subsequently processed by the AIRBiSNet. The host processor's role is largely limited to input data processing and enhancing speed. With direct DMA transfers, data throughput is maximized, and on-chip FPGA buffers reduce frequent memory access times, especially with bulky data sets.

The DMA not only transfers the trained model and input images but is also initiated by the ARM processor to expedite transfers. The accelerator's pipeline design ensures efficient data processing, allowing simultaneous handling of new and previous data. In this setup, the convolutional layer primarily accelerates tasks within the PL while the PS oversees task scheduling, data management, and interface communication.

The algorithm "Scaling Deep-Learning Pneumonia Detection Inference on a Reconfigurable Self-Contained Platform" specializes in pneumonia detection using deep learning. Designed to classify lung X-ray images through a convolutional neural network (CNN) model, it is demonstrated in Figure 5.6.

This CNN model, consisting of three convolutional layers and two fully connected layers, undergoes training via federated learning before deployment on an FPGA cluster. The algorithm adaptively selects inference strategies based on the

Algorithm 4 Scaling Deep-Learning Pneumonia Detection Inference on a Reconfigurable Self-Contained Platform. (Take a three-layer convolutional neural network as an example.)

```
 1: Train pneumonia detection model using a CNN through collaborative learning

 2: Deploy the model onto a reconfigurable FPGA cluster
 3: Input a lung X-ray image and check its size
 4: if image size < limited size then
 5:     Perform non-pipelined inference using all 4 FPGA nodes in parallel
 6: else
 7:     Node 1 performs the first convolutional layer
 8:     Nodes 2 and 3 perform the second convolutional layer in parallel
 9:     Node 4 performs the third convolutional layer and the following fully con-
        nected layers
10:     Compute the final result by passing through the pipeline
11: end if
12: Output the detection result “infected” or “normal”
```

input image size. For images smaller than 128×128, all four FPGA nodes work simultaneously in a non-pipelined manner. However, for images sized 128×128 and above, the network requires a pipelined approach. In this setup, Node 1 processes the 1st convolutional layer, Nodes 2 and 3 jointly tackle the 2nd layer, and Node 4 is responsible for the 3rd convolutional layer and the subsequent fully connected layers. The final detection is labelled as either “infected” or “normal.”

Among the algorithm’s strengths is its use of collaborative learning, which trains on distributed datasets ensuring privacy. Furthermore, the FPGA cluster deployment optimizes computational efficiency and cost-effectiveness. Its adaptive nature, based on input image size, further enhances performance.

Using the ZYNQ architecture, the ARM processor in the Processing System (PS) initializes the Programmable Logic (PL) through the control bus, setting up the convolution layer parameters at runtime, as depicted in Figure 5.7. The accelerator operates within the PL, utilizing Direct Memory Access (DMA)—a FIFO-based data transfer method between the PL and PS. This arrangement enables the host processor to focus on calculations, thereby boosting computational speed. Direct hardware-based data transfer facilitates the rapid movement of large data volumes. Moreover, the on-chip buffer, fed by the DMA, minimizes memory

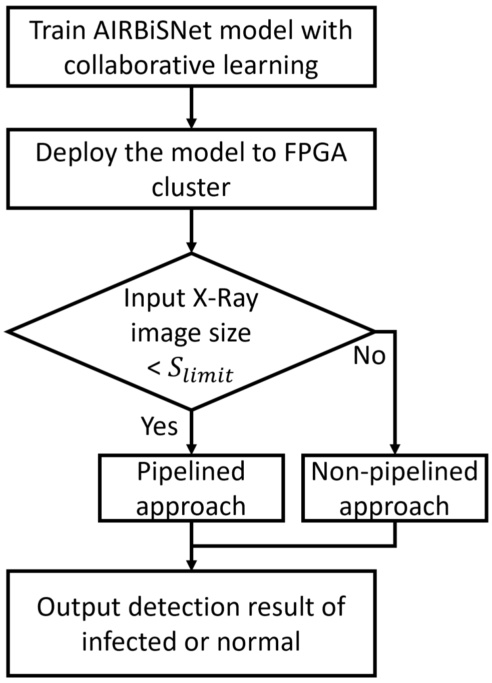

Train AIRBiSNet model with collaborative learning
Deploy the model to FPGA cluster
Input X-Ray image size < S_{limit}
No
Yes
Pipelined approach
Non-pipelined approach
Output detection result of infected or normal

Figure 5.6: The flowchart of pipeline-based inference parallelization algorithm.

access frequency, thus enhancing processing time, particularly for substantial data sets.

The RISC pipeline involves five phases: input reading, instruction decoding, execution, memory access, and output writing[144]. This pipeline design in the accelerator ensures continuous data processing, accommodating new data even as the prior data undergoes processing.

5.3.1 Partitioning CNN Models Across Multiple FPGAs

We present two partitioning strategies for the three-layered CNN model used in our study: (1) horizontal partitioning, with each layer mapped to a distinct FPGA, and (2) vertical partitioning, deploying three FPGAs for each layer. Our choice was the horizontal approach due to its inherent parallel processing capabilities, facilitating high-level synthesis without squandering the FPGA circuit area.

While vertical partitioning in CNN coding leads to data replication across FPGAs, requiring subsequent data consolidation, horizontal partitioning eliminates such necessities. By allocating a unique FPGA to each layer, data consolidation is

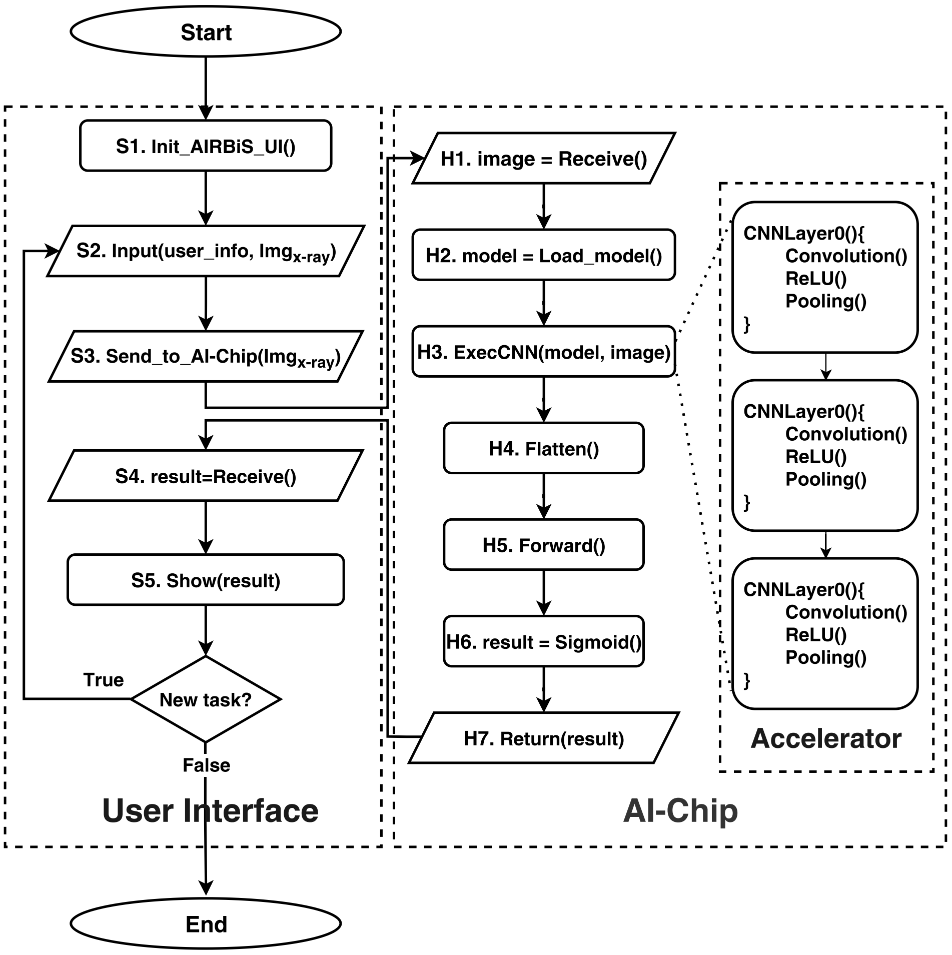

Start
S1. Init_AIRBiS_UI()
S2. Input(user_info, Img$_{x\text{-}ray}$)
S3. Send_to_AI-Chip(Img$_{x\text{-}ray}$)
S4. result=Receive()
S5. Show(result)
New task?
True
False
User Interface
H1. image = Receive()
H2. model = Load_model()
H3. ExecCNN(model, image)
H4. Flatten()
H5. Forward()
H6. result = Sigmoid()
H7. Return(result)
AI-Chip
CNNLayer0(){ Convolution() ReLU() Pooling() }
CNNLayer0(){ Convolution() ReLU() Pooling() }
CNNLayer0(){ Convolution() ReLU() Pooling() }
Accelerator
End

Figure 5.7: Flow chart of user inference and edge diagnosis in AIRBiS. In S3, the test image is sent to the FPGA. In H7, the diagnosis result is sent to the user interface.

rendered redundant, thus maximizing circuit area efficiency without unnecessary data duplication.

5.3.2 Inter-FPGA Data Transmission and Reception

Our CNN workflow employs UDP communication for inter-layer data transmission. Utilizing the standard UDP library in C++, we ensured efficient communication between FPGAs. Due to the local nature of this communication, we experienced no packet loss.

In setting up the server side of UDP, a socket is initially created, mirroring the client side's process. The server then configures its IP address and port for data reception and initializes variables with the 'memset()' function to ensure successful communication. The operating system is informed of the data reception location using the 'bind()' function. To receive data, the 'recvfrom()' function is called. Given the UDP's data transmission limit, data is segmented and received in designated quantities.

5.4 Evaluation

5.4.1 Evaluation Methodology

To evaluate our proposed system, we used the X-ray image dataset outlined in Table 4.2, selecting 700 images from each label for testing. The AIRBiSNet underwent training both conventionally and using collaborative learning techniques on servers equipped with 64GB DDR4 RAM and NVIDIA GeForce RTX 2060 GPU. For collaborative learning, one CPU machine served as the aggregator server, while several others acted as learning nodes.

We implemented the inference FPGA cluster with Xilinx's EDA suite, and the classification times on our system, CPU, and GPU were measured and compared, as shown in Figure 5.11. Power consumption data was gathered from a power meter, referenced studies [145], and software simulations.

5.4.2 AIRBiS Detection Accuracy

After training the model over 500 epochs with a batch size of 16, Table 4.1 displays the detection accuracy. Additionally, Figure 5.8 shows the AIRBiSNet's training progress in terms of loss and accuracy, indicating marked improvement as training continues.

$$\text{Accuracy} = \frac{TP + TN}{TP + FN + FP + TN} \times 100\% \tag{5.5}$$

$$\text{Precision} = \frac{TP}{TP + FP} \times 100\% \tag{5.6}$$

$$\text{Sensitivity(Recall)} = \frac{TP}{TP + FN} \times 100\% \tag{5.7}$$

$$\text{Specificity} = \frac{TN}{TN + FP} \times 100\% \tag{5.8}$$

$$\text{FPR} = \frac{FP}{FP + TN} \times 100\% \tag{5.9}$$

$$\text{F1-score} = 2 \times \frac{Sensitivity \times Precision}{Sensitivity + Precision} \times 100\% \tag{5.10}$$

In Table 5.1, we contrast the performance of AIRBiS with other leading studies using Equation 5.5. Like our approach, these studies applied CNN techniques to a COVID-19 dataset. Most references, such as [146, 147, 148, 149, 118, 150, 115], utilized under 2000 lung X-ray images with a mere 1000 of them being COVID-19 instances. Such an imbalance can compromise a CNN model's robustness. Even Lin's work in [120], which employed 15,478 images, had only 473 COVID-19 samples, indicating a significant imbalance. Our system offers a more balanced dataset, as emphasized by [151].

With an impressive 95.2% accuracy, AIRBiS outperforms the results presented in multiple studies [146, 147, 148, 149, 118, 150, 115]. This is likely due to the smaller, imbalanced datasets they employed. Although Rahman in [115] reached 96.29% accuracy using ChexNet [28] with enhanced images, our unaltered images achieved comparable accuracy. This underlines AIRBiS's efficiency, eliminating the need for intensive pre-processing.

Our system's optimization involved various experiments, with feedback from feature visualization. We explored multiple data augmentation techniques and adopted well-known CNN optimizations like the Adam optimizer, batch normalization, and dropout. Crucially, our AIRBiSNet (256 × 256) uses just 3.7 million parameters, marking about a 53% reduction from Rahman's approach [115], as

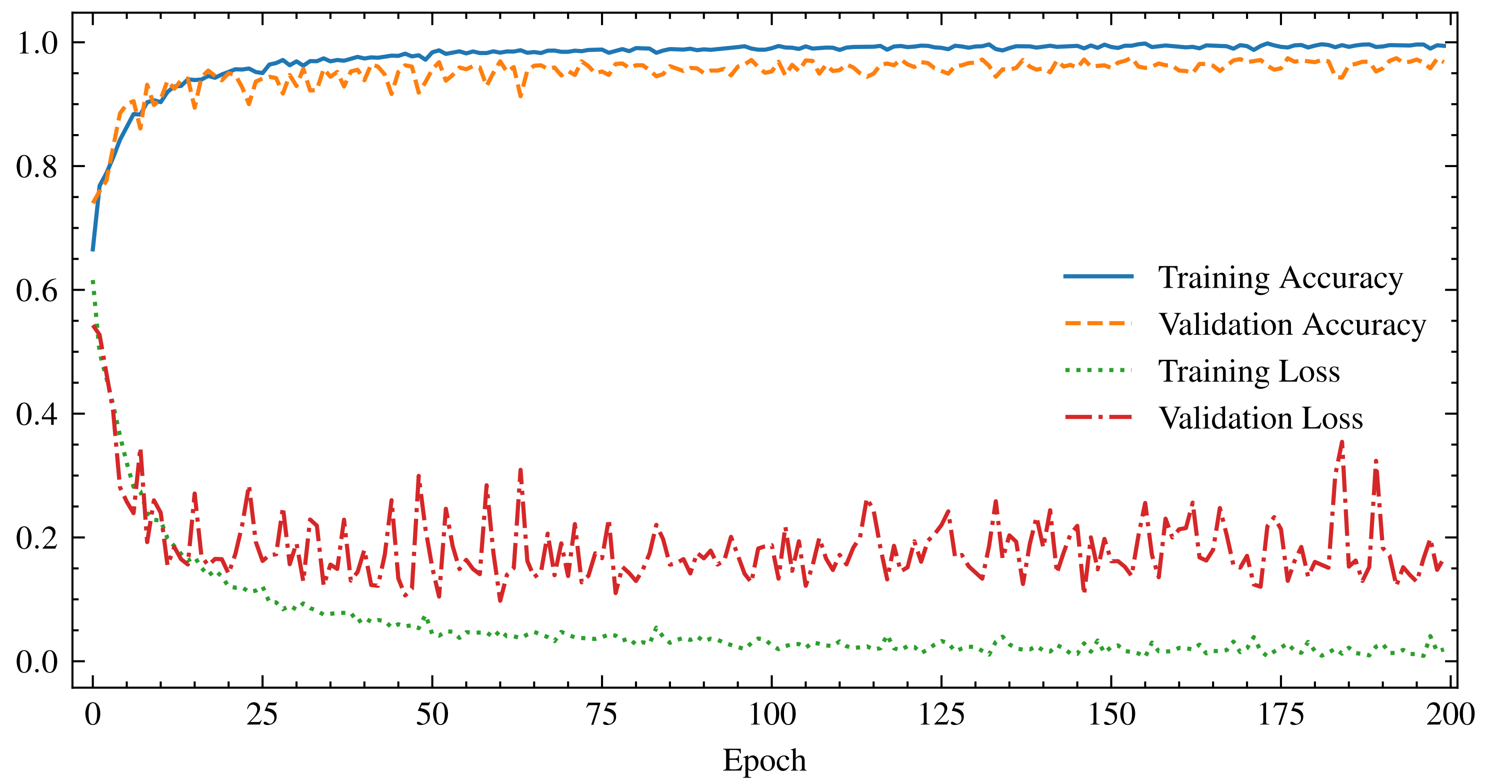


Figure 5.8: AIRBiSNet learning loss and accuracy.

Table 5.1: Comparison of the method, dataset, preprocessing, and accuracy of this work and other state-of-the-art works.

Work	Method	Dataset (Images)	Preprocessing	Accuracy (%)
[119]	CNN (ResNet101)	490 COVID-19 images, 5,942 Non-COVID.	-	71.9
[146]	ResNet50	368 COVID-19, 127 Other Pneumonia.	Segmentation, multi-view fusion, image resize.	76
[147]	Inception-V3	90 COVID-19, 54 Other Pneumonia.	Different re-sampling algorithms	88
[148]	Decaps architecture	349 COVID-19, 397 Non-COVID.	GAN, resize	87.6
[149]	COVIDX-Net (ResNetV2, VGG19, InceptionResNetV2, ChexNet, Xception, MobileNetV2, InceptionV3, DenseNet201)	25 COVID-19, 25 Normal	Rescaling	90
[120]	ResNet50	473 COVID-19, 15,005 Non-COVID.	Contrast Limited Adaptive Histogram Equalization	93
[118]	CNN (DeTraC)	949 COVID-19, 819 Non-COVID.	Data augmentation	93.1
[150]	VGG16, transfer learning with ImageNet	415 COVID-19, 5179 other Pneumonia and 2880 Normal.	Histogram equalization and bilateral filter.	94,5
[115]	ChexNet, Resnet50, InceptionV3, Resnet18, DenseNet201, Resnet101, for detection and modified Unet for segmentation.	3,616 COVID-19, 18,479 Non-COVID.	Segmentation, Histeq, data enhancement, Gamma.	96.29
This work	**AIRBiSNet**	**17,920 COVID-19 (3,571 original), 17,540 Non-COVID.**	**Resize , data augmentation**	**95.2**

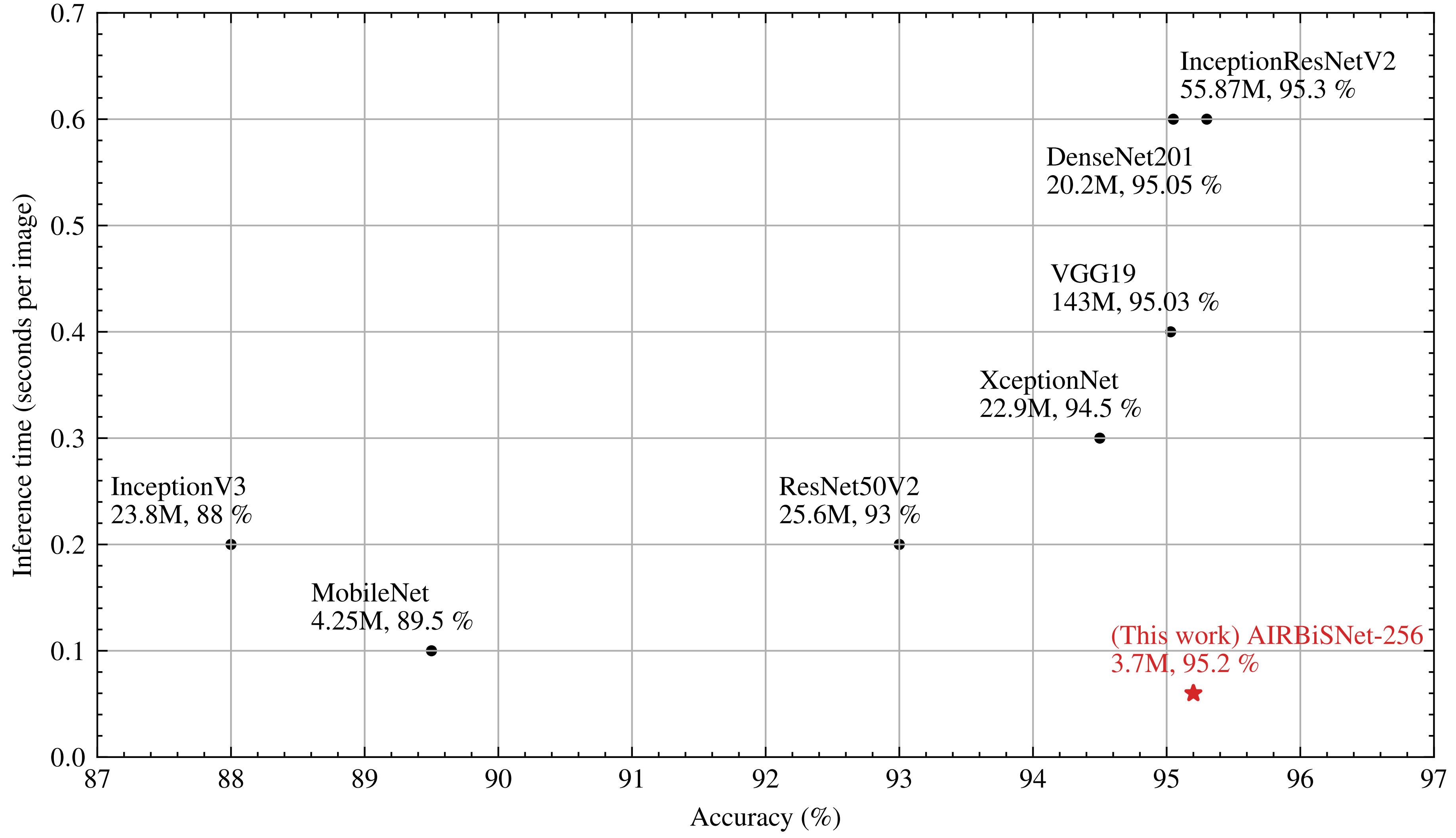


Figure 5.9: Comparison of the accuracy and inference time of the AIRBiSNet and other state-of-the-art ANNs.

illustrated in Figure 5.9.

Figure 5.9 showcases how our method measures up against other advanced methods in terms of accuracy and speed, including DenseNet201 [115], Inception-ResNetV2 [152], and MobileNet [44], among others.

In Figure 5.1, black dots represent various neural network models. The red pentagram showcases AIRBiSNet's performance using two input image sizes: 256x256 and 64x64.

AIRBiSNet, with its refined architecture, delivers superior inference performance, outstripping other large networks by over tenfold compared to ChexNet [115]. Its classification accuracy for the 256x256 version is 95.2%, closely following ChexNet's 96.29%.

AIRBiSNet-64 boosts the inference speed, meeting the high-speed detection needs in different pneumonia detection scenarios. Although it's 3X faster than AIRBiSNet-256, it compromises detection accuracy by just 3%. This faster version holds significance for swift end-side detections.

5.4.3 Inference Time and Power Consumption

Figure 5.10 presents a scatter plot of test data IDs versus the inference time (in milliseconds) of the AIRBiS pneumonia detection system. Linear regression analysis on this data yields an overlaid fitted line, whose equation is provided. The nearly flat slope suggests consistent inference times across test datasets, emphasizing AIRBiS's reliability. The plot also contains confidence interval bands around the regression line, indicating the expected range for most data points. A highlighted point, "The worst-case execution time (WCET)", marks the lengthiest inference time among the 690 tests, revealing AIRBiS's upper time limit in these scenarios.

AIRBiS aims to expedite pneumonia (specifically, COVID-19) detection using a reconfigurable FPGA cluster, ensuring both speed and low power consumption. Weight quantization optimization further refines its performance. The acceleration

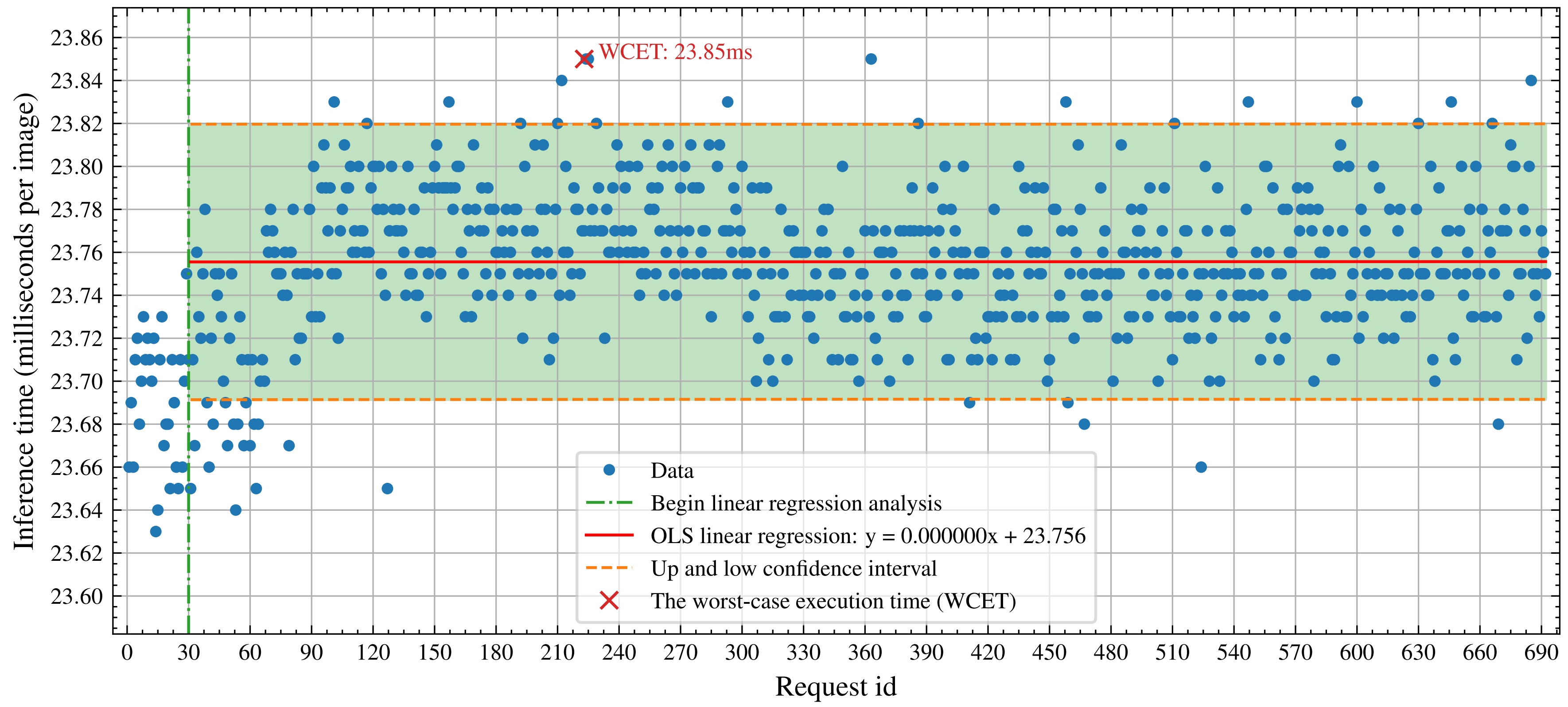


Figure 5.10: Stability of AIRBiS Pneumonia Detection System's Inference Time. It presents a scatter plot of the AIRBiS pneumonia detection system's inference time against test data, with a nearly horizontal regression line indicating consistent performance. The graph also includes confidence intervals and highlights the longest inference time, labelled as 'WCET'.

was realized on a Zynq UltraScale+ MPSoC ZCU102 FPGA cluster, employing 8-bit quantized neural networks.

Figure 5.11 contrasts the energy and power consumption across different platforms: GPU, Desktop CPU, ARM CPU, and AIRBiS-HW, using varying input sizes: 64x64x1, 128x128x1, 256x256x1, and 128x128x3. The findings highlight the FPGA-based platform's energy efficiency: it surpasses the GPU by 1.8 to 13 times, the desktop CPU by 2.6 to 5.2 times, and the ARM CPU by 2.2 to 2.6 times.

Table 5.2 outlines the hardware resource allocation for the 256x256x1 AIRBiS-HW design on the Xilinx ZCU102 FPGA board. The design occupies roughly 19.9% of the lookup table (LUT) and predominantly utilizes the board's configurable logic blocks (CLB) with a minimal RAM footprint. The breakdown includes 90.4% of BRAM, and 9.7% of FF, with an overall average FPGA resource usage of about 11.6%. This utilization is deemed satisfactory for the intended application.

Table 5.2: Hardware complexity of AIRBiS inference accelerator.

Resource	Utilization	Available	Utilization rate (%)
LUT	54585	274080	19.9
LUTRAM	3668	144000	2.5
FF	53035	548160	9.7
BRAM	824	912	90.4
DSP	35	2520	1.4
BUFG	4	404	1.0
MMCM	1	4	25

Chapter Summary

In this chapter, we presented a forward-thinking hardware-software co-design for a deep learning-based pneumonia detection system (AIRBiS) [125, 153, 141] in chest X-ray images. AIRBiS is based on a high-performance, low-power, reconfigurable FPGA cluster for inference, a robust collaborative learning mechanism for privacy-preserving, and an interactive user interface for effective operation and monitoring. We were also able to optimize and satisfy design constraints like

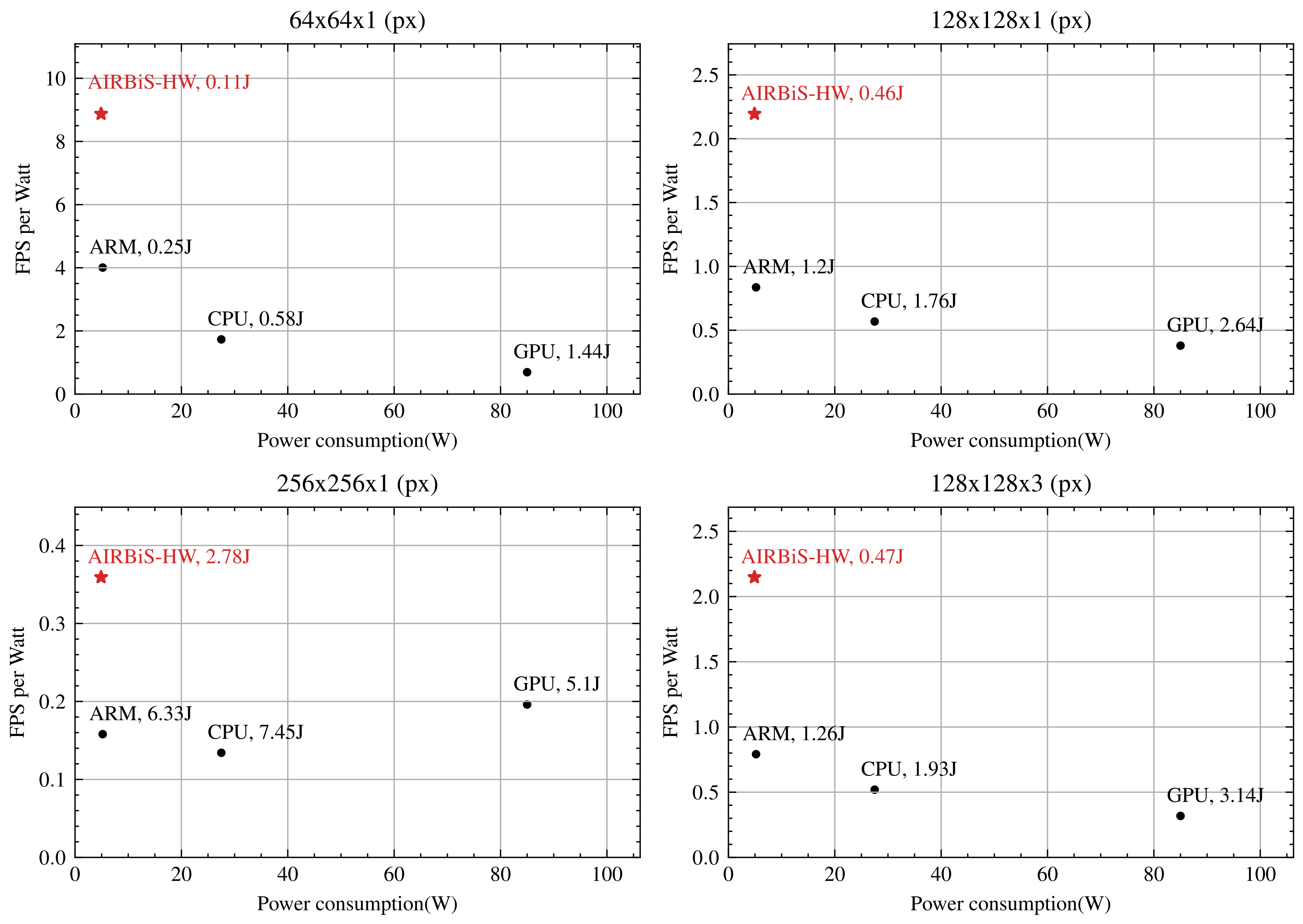


Figure 5.11: AIRBiS Inference power consumption (W) and fps per watt (FPS/Watt) comparison with different computing platforms when giving different input sizes.

power consumption. Moreover, the proposed privacy-preserving learning method, which uses data from geographically dispersed hospitals, improves training accuracy without disclosing data privacy. The performance evaluation results show that the proposed AIRBiS system achieves 95.2% detection accuracy of pneumonia (i.e., COVID-19) over the collected test data with the computer-aided diagnosis scenario. In the rapid batch detection scenario, the detection could be accelerated to 0.023 s. In addition, the system inference acceleration is 13 times more energy-efficient than GPUs, 5.2 times more than CPUs, and 2.6 times that of ARM CPUs.

Chapter 6

Event-Driven Energy-Efficient Inference Method Based on Fault-Tolerant Spike Routing Mechanism

6.1 SNN-Based Pneumonia Detection

The ascent of deep learning in the realm of artificial intelligence has propelled AI-enabled systems to tackle diverse challenges across numerous sectors. Notably, in healthcare, edge computing platforms utilize deep learning to mitigate security and latency issues, even amidst their inherent resource limitations. Conventional artificial neural networks, the foundation for most deep learning systems, present challenges due to their computational demands, power consumption, and diminished energy efficiency, making them suboptimal for edge deployment. Given the critical nature of some applications, such as biomedicine, ensuring their reliability becomes paramount during design phases. In the biomedical context, integrating the spatio-temporal processing attributes of spiking neural networks with a fault-tolerant 3-dimensional network on chip (3D-NoC) can strike a balance between accuracy, latency, and power efficiency. Accordingly, this chapter introduces a reconfigurable 3D-NoC-based neuromorphic system tailored for biomedical purposes that leverages a fault-tolerant spike routing mechanism. Performance evaluation on X-ray images, particularly for COVID-19 detection, reveals that this system boasts an 88.43% accuracy rate. Moreover, it exhibits a 4.6% superior inference latency compared to its ANN-based counterpart and consumes 32% less power. Impressively, even with up to 30% inter-neuron communication faults, the system retains high accuracy, albeit with increased latency.

Artificial intelligence (AI) is making significant strides in various domains, with healthcare being one of the primary beneficiaries. Deep learning (DL) models in healthcare aid in tasks such as patient health monitoring anomaly detection [106], lung ultrasonography classification [127], and COVID-19 detection and diagnosis [128, 129].

Many opt to deploy these DL models on cloud platforms [130]. However, when real-time analysis and data security are paramount [154, 155, 156], cloud platforms may not always be ideal due to concerns about latency [132] and security [131]. Edge computing offers an alternative, emphasizing low-latency [134], privacy [132], and security [135]. Yet, edge devices grapple with the resource demands of deep neural networks (DNNs) [136]. Conventional ANNs, despite their efficacy, are resource-intensive [157], challenging their deployment on edge platforms.

Enter the spiking neural network (SNN). Mimicking the brain's event-driven neuronal communication, SNNs, like the leaky-integrate-and-fire (LIF) model, can offer energy-efficient, fast, and real-time processing suitable for edge applications [158]. Unlike the resource-heavy multiply-accumulate (MAC) operations in ANNs, SNNs leverage binary spike events and accumulations, as visualized in Figure 6.1. When implemented on hardware platforms like the field-programmable gate array (FPGA) [139], renowned for its reconfigurability and efficiency, SNNs excel in parallelism and speed, making them ideal for DL tasks on edge devices.

The COVID-19 pandemic, which emerged at the close of 2019, has compelled

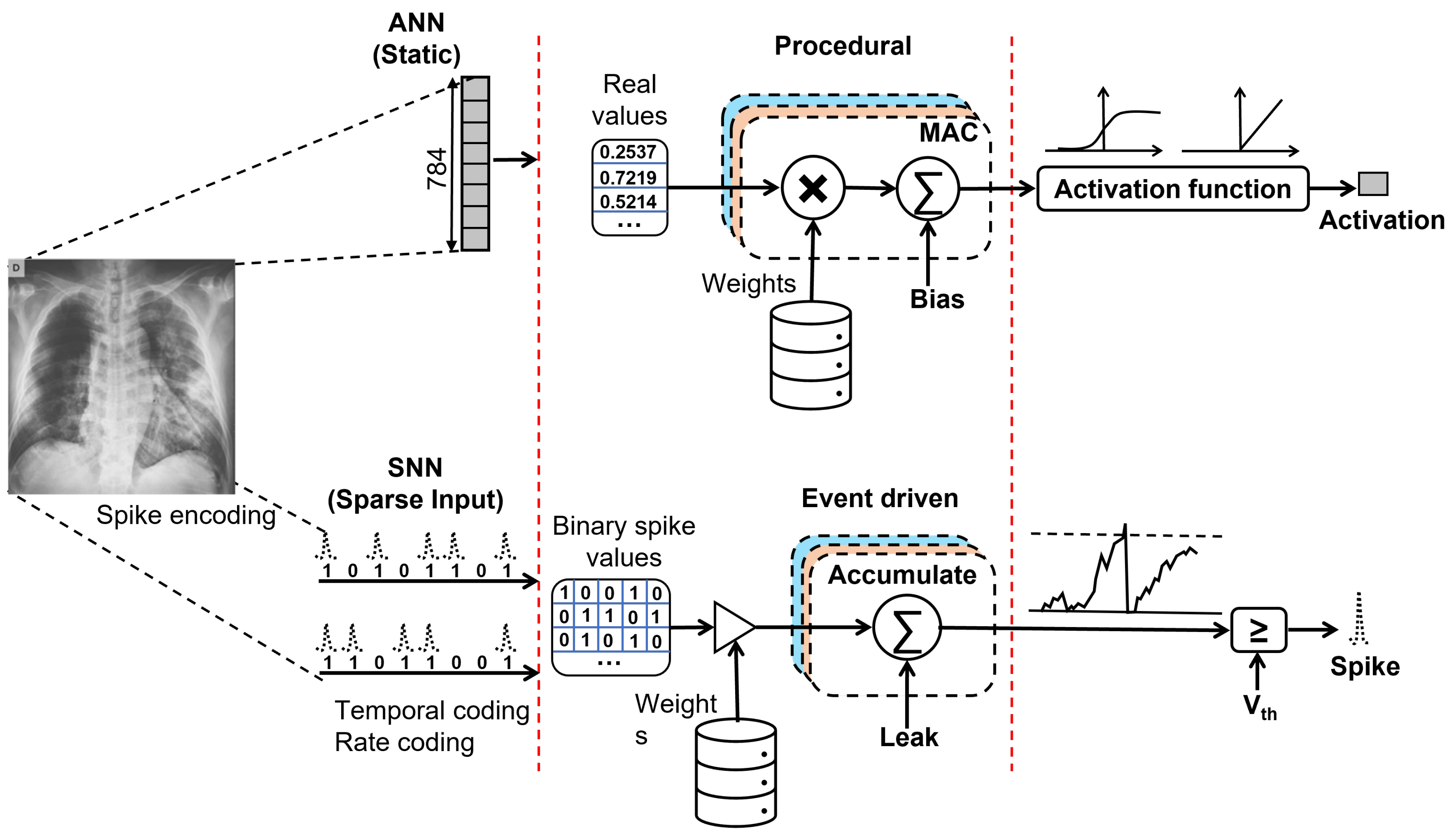


Figure 6.1: Comparison of computational between conventional ANNs and SNNs [18].

countries to invest heavily in diverse fields, notably bio-medicine, to counteract its impact. One focal area of research has been the deployment of DL-based systems on edge computing platforms for timely COVID-19 diagnosis. Specifically, deep learning models, such as SNNs, have been explored to create reconfigurable 3D-NoC-based neuromorphic systems tailored for edge platform-based pneumonia (COVID-19) detection.

However, the inherent computational intricacy of deep learning models makes them a formidable challenge for edge platforms. SNNs, drawing inspiration from the brain's efficient information encoding, present a solution. But as the size of such brain-like models expands, maintaining effective neuron communication proves challenging [159, 160]. Merging neuromorphic systems with a 3D network-on-chip (NoC) framework can pave the way for scalable, energy-conscious architectures apt for biomedical endeavours.

As COVID-19 spread globally, it became a pervasive disruptor. By August 2022, over 578 million cases were reported worldwide, leading to over 6 million fatalities [140]. A key defence against this spread is efficient and quick diagnosis. The prevalent RT-PCR technology, which tests for the virus's genetic material, varies in its sensitivity, ranging from 60% to 97% [108]. However, factors like patient variation can decrease this sensitivity to 60–71% [109]. Moreover, both symptomatic individuals and silent carriers can sometimes test negative, posing a potential health risk.

Testing for COVID-19 can be batch-processed, but sample collection remains manual. A diagnostic alternative involves analyzing patient lung X-ray images, boasting an accuracy between 80 to 90% [110]. However, this method requires doctors to scrutinize individual images and merge findings with patient symptoms for a full diagnosis. Given the burgeoning case numbers, this approach struggles with promptness, report generation, and maintaining patient confidentiality—key factors in treatment. As a solution, there's a turn towards computer-aided diagnosis systems underpinned by deep neural networks (DNNs) [77, 2]. Still, many medical establishments lack the power-efficient infrastructure for broad-scale diagnostic activities. The typical data security measures in medicine also hamper distributed learning processes, thereby complicating the pooling of diagnostic models from various institutions to refine accuracy and pace. This limitation affects real-time report generation and collaborations with governance. To circumvent these obstacles, specialists are exploring supplementary avenues to bolster COVID-19 detection accuracy.

DNNs, characterized by multiple densely interconnected layers, display impressive variability, making them prime candidates for accurate inference. Yet, rooted in traditional ANNs, their deployment on edge computing platforms poses challenges [161]. Hence, it's pivotal to factor in efficient resource allocation [157], green AI aspirations [162], and minimal power consumption for optimal inference [163] when applying DNNs.

In earlier contributions, we introduced an AI-enabled Real-time Biomedical System (AIRBiS-1) [17, 112], designed for COVID-19 detection and health surveillance. This system integrates a dynamic AI chip designed for efficient inference and privacy-centered collaborative learning coupled with an intuitive user interface. While AIRBiS-1 exhibited superior accuracy using traditional ANNs, its suitability for edge deployments is curbed by ANNs' computational demands and energy usage. Addressing this, we proposed an alternative detection approach for chest x-ray images using software-based neuromorphic spiking neural networks [164]. The present work builds upon our former studies, introducing a reconfigurable 3D-NoC-based neuromorphic system tailored for biomedical endeavours and benchmarking it against X-ray imagery for COVID-19 diagnosis.

The proposed reconfigurable neuromorphic biomedical system is depicted in Figure 6.2, an evolution of our previously presented 3D-NoC-based neuromorphic processor [19]. This system encompasses convolution cores, spiking neuron processing cores (SNPCs), and a fault-tolerant 3-dimensional router (FT-3DR). Assembled in 2D mesh topologies, these components collectively shape a 3D mesh architecture. The convolution cores are dedicated to feature extraction from X-ray images to identify pneumonia, whereas the SNPC interprets these features to finalize the pneumonia diagnosis. The FT-3DR mediates the communication between these cores. Subsequent in this section, we delve deeper into the components' specifics.

6.2 Convolution Core

Figure 6.3 elucidates the convolution core's blueprint, segmented into a convolution unit and a pooling unit. Within the convolution unit, one can find a strider, kernel memory, and a controller, whereas the pooling unit comprises a max pool unit, a max-storing register, and a spike-generation threshold derived from the accumulated max feature. Initially, a CXR image undergoes spike encoding and is relayed to the convolution unit. Here, the strider processes the spike-encoded image over several time increments in its 2D form and then linearizes it. The strider subsequently elects pixel values from the linear spike-encoded image, aligning them with the convolution filter's dimensions. These elected values are organized into an array, which parallels a flattened kernel in length. Subsequently, a one-hot operation identifies the spiked pixels, which then guide kernel memory address queries. These addresses yield specific kernel weights fed into integrators. Following integration, potential values get logged into respective feature map registers. This routine continues until all image strides have been assessed, yielding kernel-specific potential feature maps. The system then pivots to the max pooling operation. The convolution core's operations are all under the aegis of its control unit.

The pooling operation involves the strider using the size of the pooling window to process the potential feature map. For each window, the highest potential value is combined with the potential register. If this combined value surpasses a given threshold, a spike is generated and mapped onto the resultant spike feature map. If

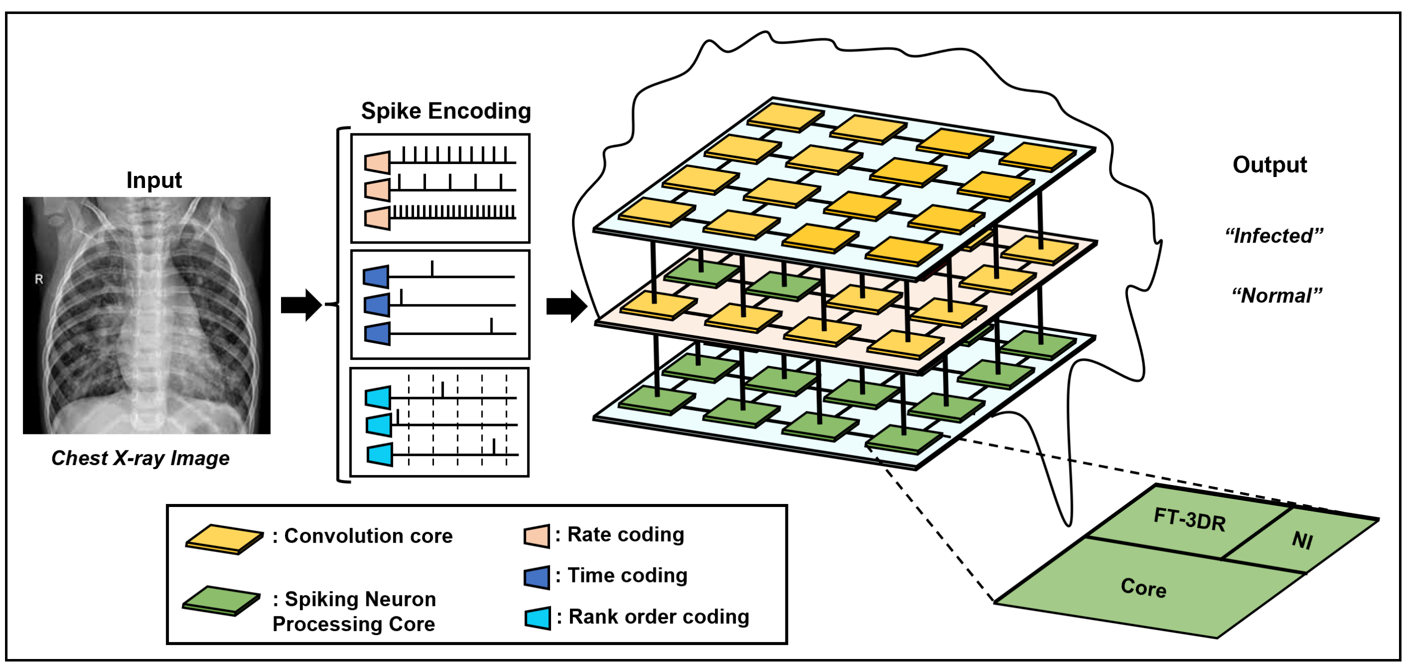


Figure 6.2: High-level view of the reconfigurable-reliable neuromorphic-based biomedical system.

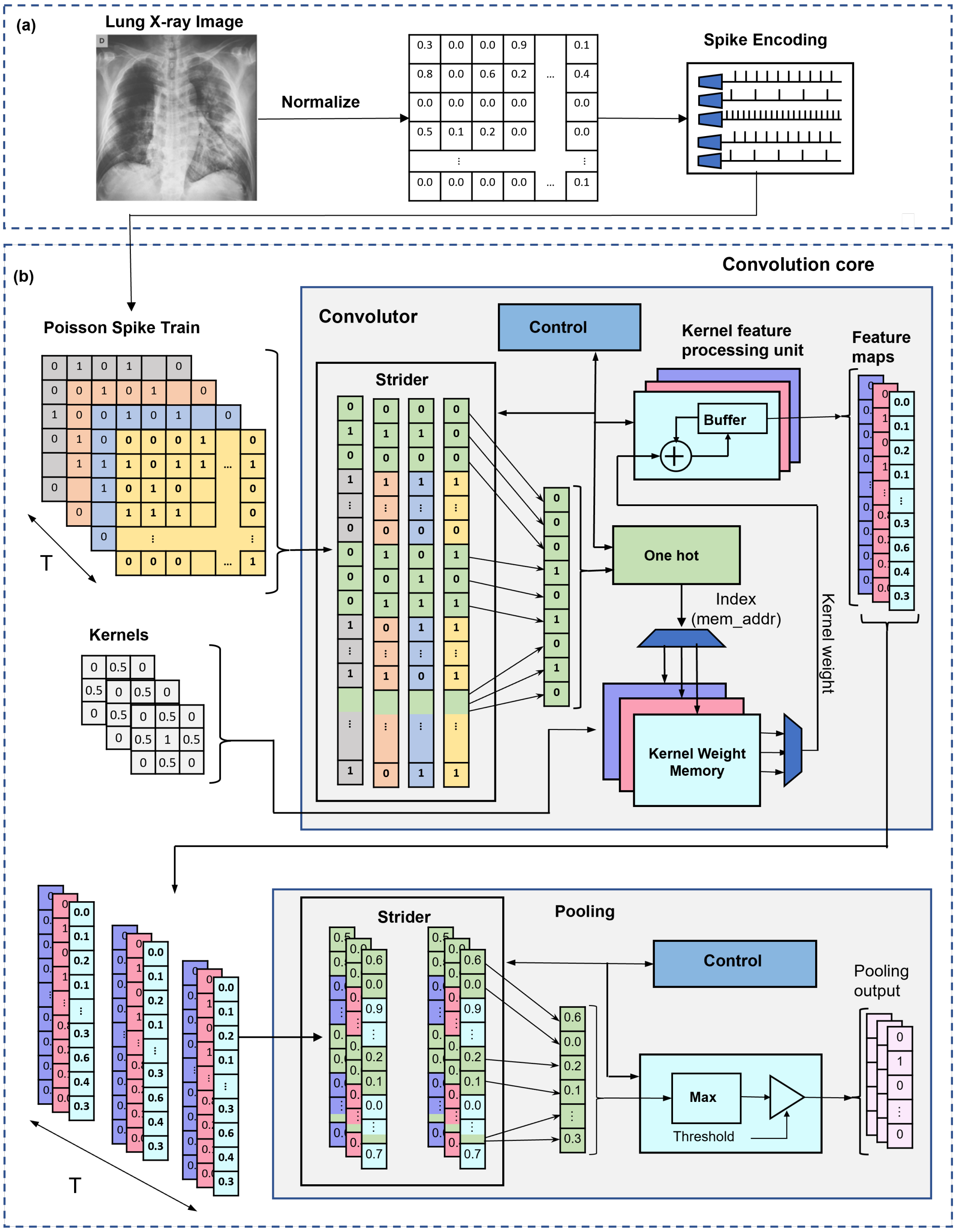


Figure 6.3: Convolution core architecture consisting of a convolutor and a pooling unit.

not, no spike is produced. This process runs concurrently for all potential feature maps from the convolution. If subsequent layers are convolutional, the pooling layer's spike feature map is relayed to another convolution core, continuing until a connection is established with the spiking neuron processing core.

6.3 Spiking Neuron Processing Core

Figure 6.4 illustrates the SNPC, building on our prior designs [19, 165]. Key components include an array of leaky integrate and fire (LIF) neurons, a synapse crossbar, synapse memory encapsulating synaptic interconnections among LIF neurons, and a controlling unit. Upon receiving a spike, the spike feature map, originating from the pooling layer, is directed to the synapse crossbar. Here, the one-hot mechanism identifies spike presence. Detected spikes guide memory address retrieval from the synapse memory, subsequently fetching the relevant synapse weights. These weights are routed to their corresponding post-synaptic neurons, which accumulate and apply a decay operation on them. The resultant value is stored in a designated register.

Neurons emulate neural membrane leak currents by receiving a predefined leak value, decaying the membrane potential when activated. Once this accumulated potential surpasses a certain threshold, an output spike is initiated. The membrane potential then resets, marking the refractory phase's commencement. During this phase, the neuron doesn't aggregate weighted input spikes. As the refractory count diminishes over time steps, reaching zero allows the neuron to resume its accumulation activity. The mechanism for weighted presynaptic spike accumulation, as outlined in [166], follows:

$$V_j^l(t) = V_j^l(t-1) + \sum_i w_{ij}{}^* x_i^{l-1}(t-1) - \lambda \tag{6.1}$$

where V_j^l is the membrane potential of a LIF neuron j in layer l at a time-step t. w_{ij} is the synaptic weight from neuron i to j, λ is the leak, and x_i^{l-1} is a

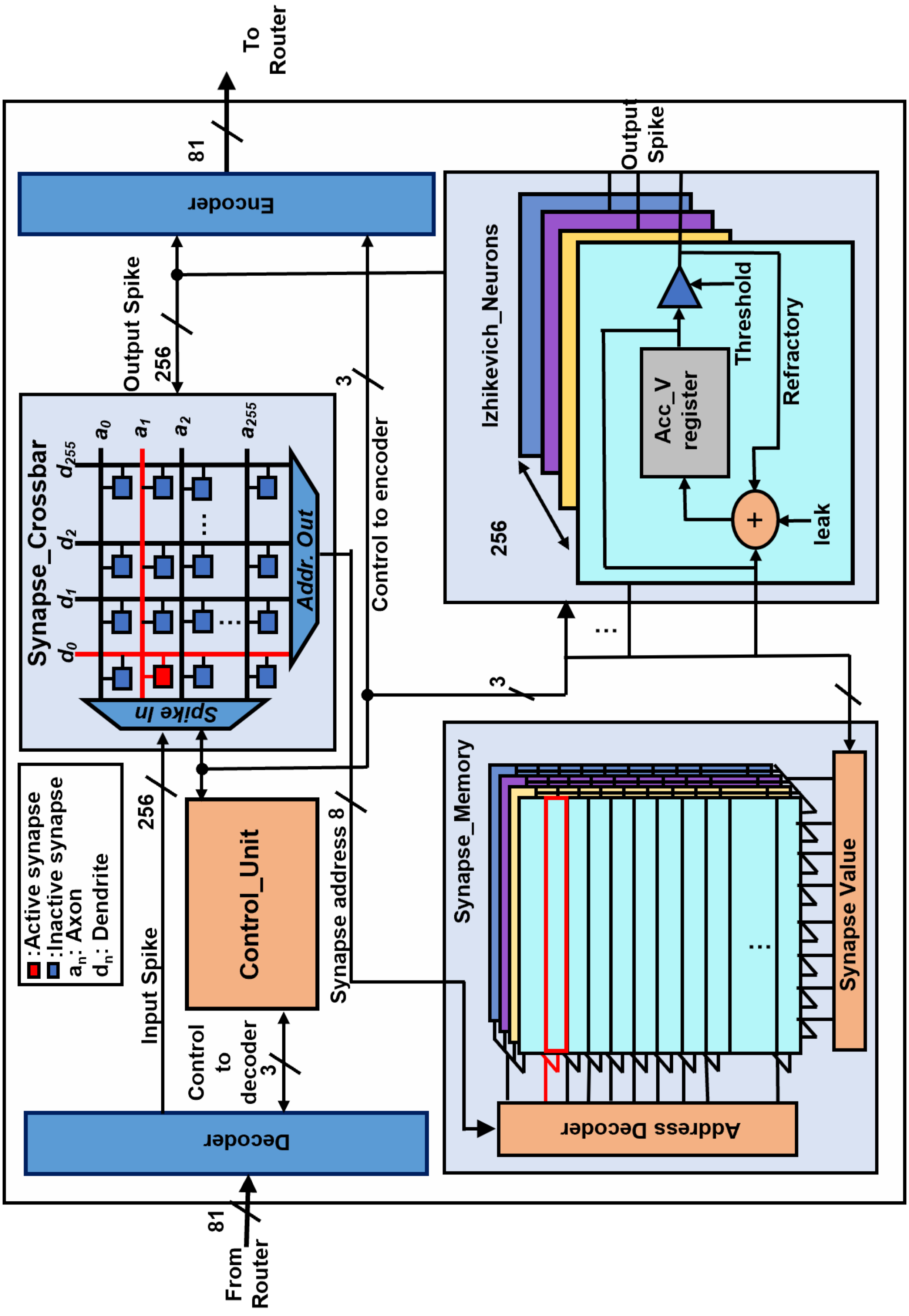


Figure 6.4: Architecture of spiking neuron processing core [19].

pre-synaptic spike from the previous layer $l-1$.

Upon integrating input spikes, if the value of V_j^l at a particular timestep surpasses the threshold θ, a spike ($s=1$) is emitted, causing the neuron to reset. This behaviour is detailed in [165].

$$s = \begin{cases} 1, \text{if } V_j^l > \theta \\ 0, \text{if } V_j^l < \theta \end{cases} \tag{6.2}$$

6.4 Spike Packet Format

The spike packet format, as the name implies, is crucial in determining how the signal information is organized and transmitted across the neuron network. It can be considered a protocol of information exchange, where each packet encapsulates the necessary data payload of a neural spike.

This protocol is crucial because it determines how information is decoded by the receiving neuron. If the receiver fails to interpret the packet correctly, it will lead to an unsuccessful or even damaging transmission. Therefore, the format of such a spike packet becomes a vital aspect of the overall neuron network communication system.

The spike packet primarily consists of fields such as the source ID, destination ID, spike time, and the spike event itself. The source and destination IDs are indicative of the originating and target neurons, respectively. The spike time can be interpreted as a timestamp, enabling temporal sequencing of the signals. The spike event field encapsulates the actual neural spike information. It's important to note that this configuration may be tailored according to specific research needs or applications.

The modular design of the spike packet format is flexible, allowing for the integration of additional fields or the modification of existing ones, thus enhancing the functionality and adaptability of the network. This flexibility is essential when the network needs to adapt to different operational conditions or experimental setups.

As mentioned earlier, the design of the spike packet format used in this study originates from the results of our previous research [167]. In that work, we established the basic structure and identified the fundamental parameters of a spike packet. The current design builds on that foundational work by incorporating additional features and optimizing the format for more complex neural networking tasks.

The transmission of a signal, or a 'spike packet', from a firing neuron to the target neuron tile via the network necessitates an understanding of the format of such a packet. This is detailed in Table 6.1. Notably, the packet format can be customized to cater to different requirements.

Table 6.1: Spike packet format.

Length	**Field Name**	**Meaning**
2-bits	Type	'00': configuration; '11': spike
3-bits	Fault_Flag	Flag for fault-tolerant spike routing algorithm
9-bits	XYZ_s	Source node address
6-bits	Timestamp	The fired time
8-bits	Neuro ID	Identifier of the fired neuron

1. *Type*: This element acts as the packet header, signalling whether the packet serves a configuration or spike purpose. A '00' type designates the system configuration used, where the packet body carries configuration data, as described in Section 6.3. A '11' type, on the other hand, designates a spike packet. Other types are reserved for future use.

2. *Fault_Flag*: Predominantly used in conjunction with the fault-tolerant multicast routing algorithm, this flag indicates whether the packet has followed the primary or backup path. By default, $fault\_flag = 0$, and it is initialized at the start of the backup path (at nodes referred to as 'father' or 'grandfather'). The $fault\_flag$ value is then decremented by one with every hop on the backup path until it reaches zero at the end node or the 'son'.

3. XYZ_s: This is the address of the origin neuron tile, utilized for spike routing to ensure successful packet delivery to the destination tile. Upon the packet's arrival at a router's input buffer, the address is extracted for routing computation, for instance, for consulting the routing table.

4. *Timestamp*: In a spiking neural network, the generation time of a spike encodes information. Each spike packet from the source neuron includes the spike's Timestamp. This data is decoded into the precise time slot of the input spike, which plays a fundamental role in Hebbian-based learning rules. For instance, in the Spike Timing Dependent Plasticity (STDP) learning rule, the arrival time of the input spike is compared with the spike time of the post-synaptic neuron to modify the synaptic strength.

5. *Neuron ID*: Serving as the identifier of the pre-synaptic neuron, the Neuron ID comes into play after the spike packet is delivered to the destination neuron tile via a XYZ_s. It helps identify post-synaptic neurons in the SNPC decoder process. The unique address for each neuron within the entire system is achieved by the combination of XYZ_s and Neuron ID.

6.5 Fault-Tolerant Multicast 3D Router

Figure 6.5 delineates the fault-tolerant multicast 3D router responsible for guiding spikes/packets throughout the system, drawing its foundation from [168, 169]. The router boasts seven pairs of I/O ports: one links to the local core, four to adjacent routers in cardinal directions via intra-layer links, and two connect vertically through TSVs to the nearest layers. This router processes multicast packets through four stages:

1. Buffer Writing (BW): Incoming packets are stored in the input buffer of the port.

2. Routing Calculation (RC): The packet's source address is determined, fur-

ther assisting in deriving the X, Y, or Z-dimensional address.

3. Switch Allocation: Facilitated by a matrix-arbiter scheduler, this stage determines the appropriate port for the succeeding router or local SNPC.

4. Crossbar Traversal (CT): The packet proceeds through the crossbar to the designated output port.

With the help of the fault-tolerant shortest path k-means-based multicast routing algorithm (FTSP-KMCR) [169], the router not only efficiently dispatches packets between cores but also circumvents permanent communication link faults by utilizing alternative routes.

6.6 Evaluation

This section delves into the evaluation methodology of the 3D-NoC-based neuromorphic pneumonia detection system, presents the detection accuracy on the chest X-ray dataset, assesses hardware complexity and fault-tolerant performance, and contrasts these results with current research, focusing on accuracy, inference time, and power consumption.

6.6.1 Evaluation Methodology

Utilizing the snnTorch simulation framework [170], the proposed system is trained off-chip via backpropagation spike-driven learning with rate spike encoding. This encoding converts X-ray images into time-varying spikes of varied frequencies. By influencing the SNN's final layer, correct classes are motivated to spike more, while incorrect ones do less. The gradient navigates backwards from the loss to all synaptic weights, cumulating before weight adaptation. To promote spiking and strong gradients during misclassifications, the target is applied to the membrane potential.

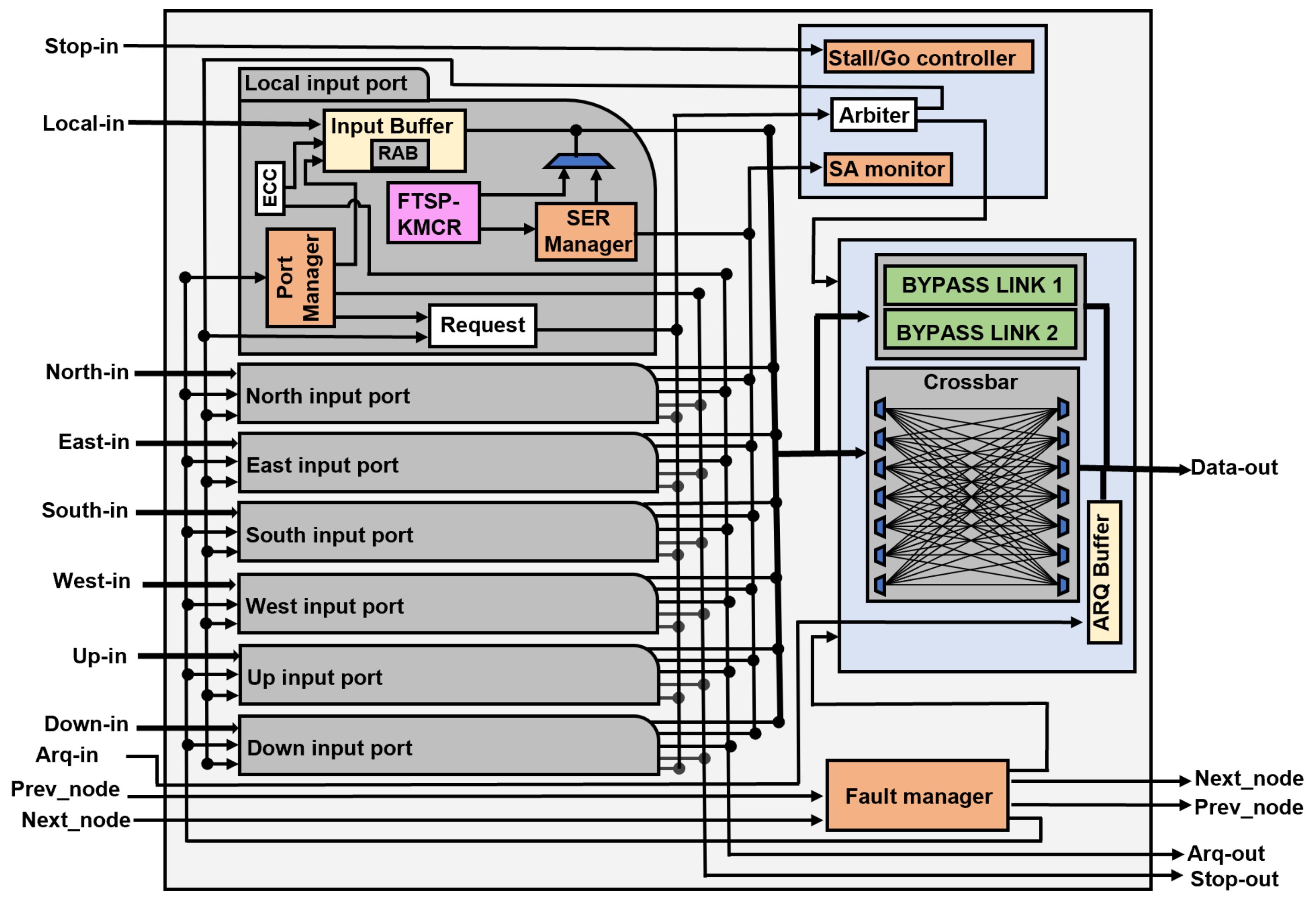


Figure 6.5: Architecture of fault-tolerant 3D router [19].

The hardware implementation is achieved on a Xilinx Zynq UltraScale+ MPSoC ZCU102 board using Verilog HDL, Xilinx's EDA suite (Vivado), and Cadence EDA tools. For ASIC implementation, resources include the NANGATE 45 nm open-cell library [171], OpenRAM [172] for system memory, and FreePDK3D45's TSV [173].

The SNN, detailed in Table 6.2, is trained on the Kaggle lung X-ray image dataset [114, 115], characterized in Table 4.2. This dataset comprises 34,060 training images, with over half being COVID-positive and 42% augmented for dataset enhancement. Post-training, on-chip inference utilized the trained weights. Our evaluation metrics encompass accuracy, average inference time, and fault tolerance using a testing dataset of 1400 images, outlined in Table 4.2. The design's power consumption, area, and fault tolerance are further compared with AIRBiS-1 [112] and other contemporaneous works.

Table 6.2: Structure of SNN diagnosis/detection model for 64 × 64 input images.

Layer	**Output Shape**	**Parameters**
Conv2D_1	(16, 62, 62)	160
MaxPooling2D_2	(16, 31, 31)	0
Leaky_3	(16, 31, 31)	0
Conv2D_4	(32, 29, 29)	4640
MaxPooling2D_5	(32, 14, 14)	0
Leaky_6	(32, 14, 14)	0
Flatten_7	12,546	0
Linear_8	2	12,546
Leaky_9	(2)(2)	0
Total parameters		17,346
Trainable parameters		17,346
Non-trainable parameters		0

6.6.2 Detection Accuracy

The SNN model outlined in Table 6.2 includes two convolution layers, each paired with max-pooling and leaky layers, followed by a linear layer. The first convolution layer processes encoded lung X-ray images, extracting features that are then reduced in size by the max pool layer. These features are transformed into spikes in the leaky layer based on accumulated values and thresholds. The process is reiterated in the second convolution layer. The flattened spike features from this layer proceed to the linear layer for classification into 'normal' or 'infected'.

For inference, the SNN model integrates into a $3 \times 3 \times 3$ NoC design, following a layer-layer mapping depicted in Figure 6.6. The first convolution layer (having 16 single-channel kernels) and associated operations are mapped to the first system layer, distributing two kernels across seven cores and one kernel on the remaining two. The second convolution layer with its 32 kernels (16 channels each) corresponds to the system's second layer, apportioning four kernels across seven cores and two kernels on the final two. The classification layer, consisting of two LIF neurons, is directed to the third layer's SNPC cores. Achieving 88.43% accuracy for 25 timesteps per image, the SNN model falls slightly short when compared to the ANN-based AIRBiS-1's 94.4% accuracy, as expressed in Equation (5.5) [112].

6.6.3 Hardware Complexity

Table 6.3 presents the FPGA resource utilization of both the proposed and ANN-based (AIRBiS-1) systems. The proposed system consumed 9.9% of the lookup table (LUT), 1.28% LUTRAM, 6.7% flip flop (FF), and 4.45% BUFG. This is considerably less than the ANN-based system, attributed to the proposed system's binary input values reducing bandwidth and storage needs. Additionally, the proposed system does not require multiplication operations, unlike the ANN-based system.

The convolution core of the proposed system occupies an area of 0.0748 mm^2, slightly less than the ANN-based system's 0.0755 mm^2. The reduction arises from

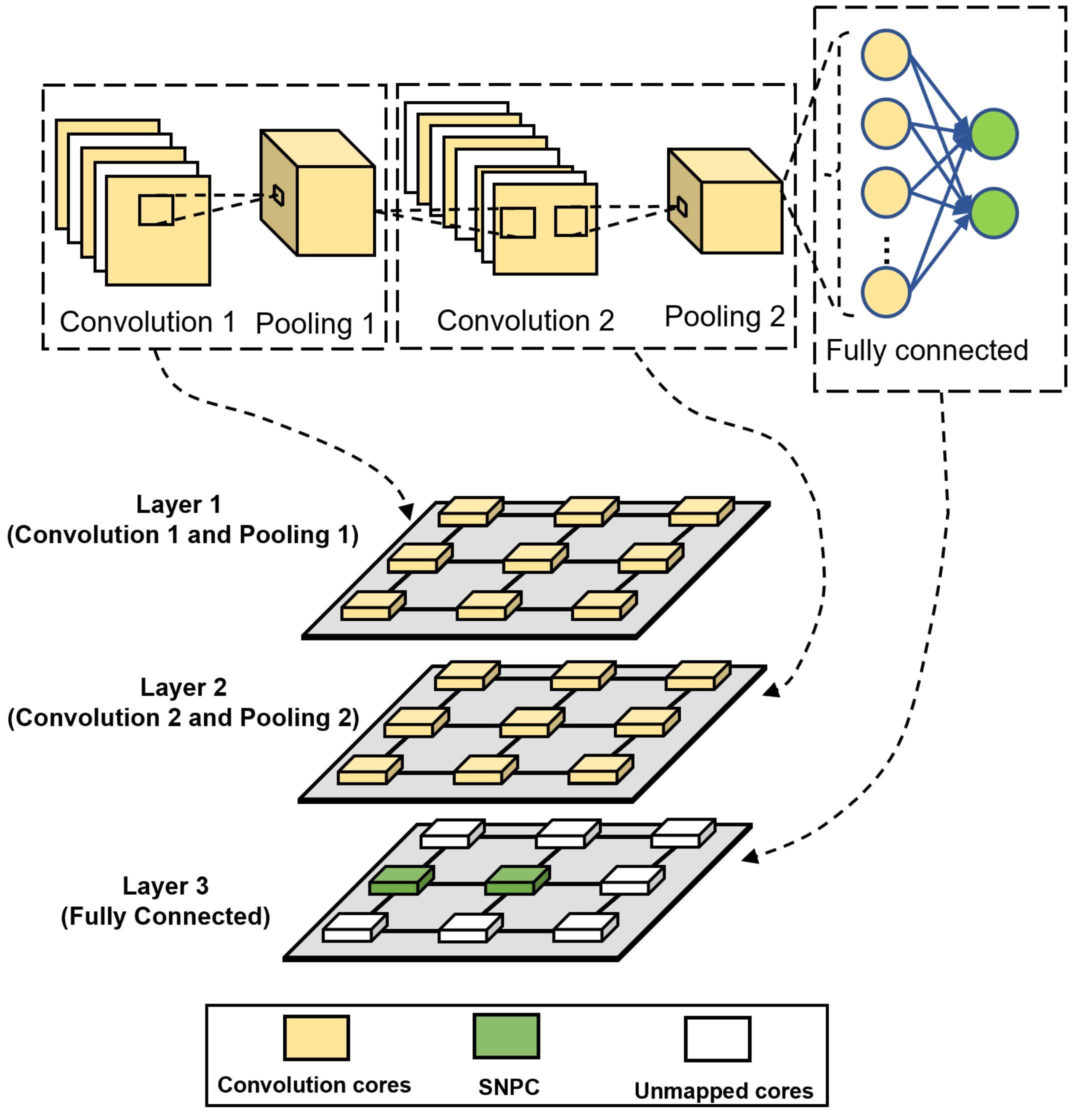


Figure 6.6: Layer-based mapping of the SNN model on the proposed system.

Table 6.3: FPGA resource utilization for the proposed neuromorphic system and the ANN-based system.

Resource	Utilization		Available	Utilization(%)	
	ANN	SNN		ANN	SNN
LUT	54,585	27,288	274,080	19.9	9.9
LUTRAM	3668	2048	144,000	2.5	1.28
FF	53,035	37,098	548,160	9.7	6.77
BRAM	824	0	912	90.4	0
DSP	35	0	2520	1.4	0
BUFG	4	18	404	1.0	4.45
MMCM	1	0	4	25	0

the lack of multiplication operations and the use of binary input in the proposed system. Correspondingly, its power consumption (0.007 mW) is also lower than the ANN-based system (0.011 mW).

The proposed system's layers are interconnected via SNPC, while the ANN-based system employs a multiply-accumulate (MAC) core. For equitable assessment, the MAC core incorporates 256 units, aligning with the 256 LIF neurons in the SNPC. The proposed system exhibits both a smaller area (1.325 mm^2) and reduced power consumption (0.007 mW) than the ANN-based system (1.434 mm^2 and 0.011 mW, respectively), a consequence of the ANN system's more intricate MAC unit.

Classifying a single X-ray image at 100 MHz, the proposed system averages 41 ms. As illustrated in Figure 6.7, the time distribution is 35.9% for the first convolution layer, 7.9% for the second, 6.6% for the SNPC, and 49.6% for data reception. However, Figure 6.8a reveals that fewer timesteps result in diminished accuracy, with no noticeable gains beyond 25 timesteps. In comparison, the ANN-based system demands about 43 ms for a similar task, indicating a slightly lengthier classification duration.

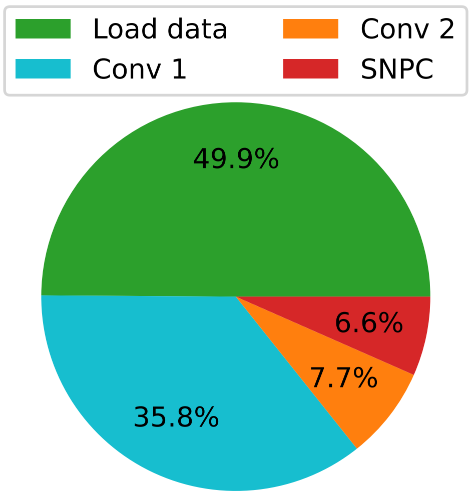
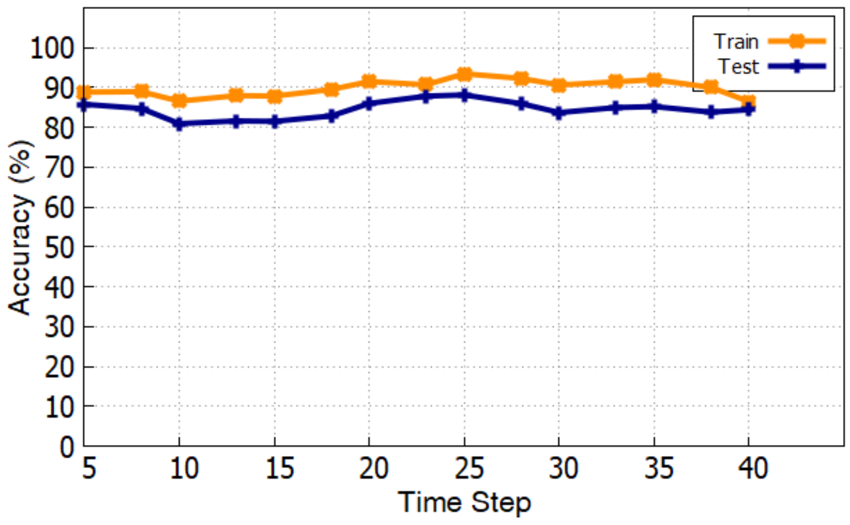
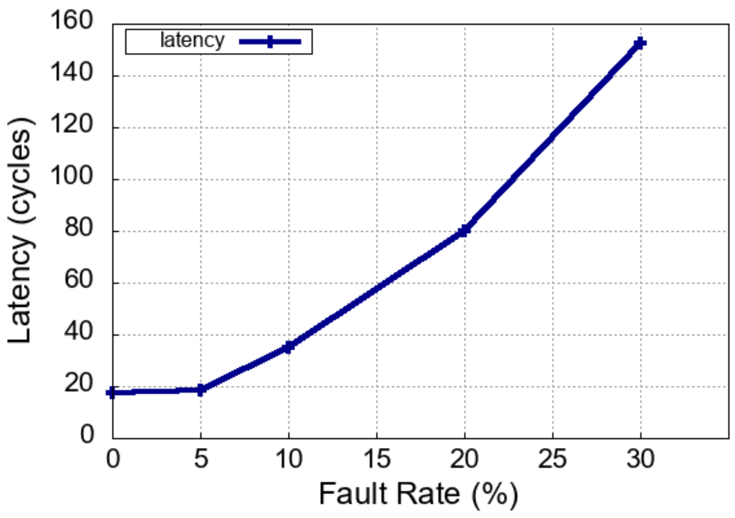


Figure 6.7: Average classification time complexity.

Figure 6.8b presents the fault-tolerant evaluation of our system, simulated with a spike injection rate of 9. At this rate, any decrease would cause spike packet



(a) Detection accuracy



(b) Detection latency

Figure 6.8: Pneumonia detection on the proposed system with various simulation time steps and fault rates. (**a**) Detection accuracy over various time steps. (**b**) Spike communication latency over various fault rates.

drops, as packets from all nine layer cores would converge in a brief cycle. The system retains detection accuracy even with a 30% communication link fault rate. Although the spike communication latency spikes with fault rates from 5% to 30% —due to increased packets rerouting around faulty links—the throughput remains consistent across varying fault rates. Beyond a 30% fault rate, packet loss becomes prevalent, with insufficient alternative paths.

6.6.4 Comparison with Existing Works

Table 6.4 contrasts the results from our system with several contemporaneous research efforts. The study in [174] utilized an SNN with 256×256 X-ray images and reported 78% accuracy. In [164], a system using 64×64 X-ray images via SNN achieved 80.7% accuracy. However, both these systems lag behind our model's accuracy of 88.43%. The reduced accuracy in these works might stem from their smaller SNN architectures. Also, both focused on software implementations and did not detail power metrics. The research in [119] employed 224×224 X-ray images with an ANN model, yielding inferior accuracy compared to our system. Our earlier effort, AIRBiS-1 [112]—comparable to the ANN-based model we scrutinized—used an analogous dataset and posted a higher accuracy of 94.4%. Nevertheless, our current system offers advantages in power and area efficiency and reduced inference time, as delineated in Section 6.6.3.

Chapter Summary

This chapter introduces a 3D-NoC-based neuromorphic system tailored for pneumonia (specifically, COVID-19) detection in chest X-ray images [124, 164]. The system capitalizes on event-driven information processing of SNNs for energy efficiency and incorporates a fault-tolerant spike routing scheme to enhance resilience. Evaluations indicate a 32% power reduction and a 4.6% faster inference time than ANN-based models, albeit with a slight dip in classification accuracy. In

Table 6.4: Comparison with the existing studies of the neuromorphic system.

Works	**Model**	**Platform**	**Dataset**	**Image Size**	**Accuracy (%)**
Fukuchi et al. [164]	SNN	Software	X-ray	64×64	80.7
Kamal et al. [174]	SNN	Software	X-ray	256×256	78
Che Azemin et al. [119]	ANN	Software	X-ray	224×224	71.9
Wang et al. [112]	ANN	FPGA	X-ray	256×256	94.4
This work	SNN	FPGA	X-ray	64×64	88.43

comparison to similar systems, ours boasts superior accuracy and can handle up to 30% inter-neuron communication faults, albeit with increased latency. The growing emphasis on edge learning—driven by evolving user/environment needs, task demands, and privacy considerations—necessitates neuromorphic systems with on-chip learning capabilities. Although our current design lacks this feature, it presents a direction for subsequent research, which should delve into algorithms promoting efficient on-chip learning. Designing neuromorphic systems involves harnessing the unique spatio-temporal and sparse activation traits of SNNs. The spike encoding scheme, pivotal in determining system metrics like accuracy, latency, and power consumption, must be judiciously selected [175]. The ideal choice varies based on the primary objective, be it accuracy, noise resilience, compression, or robustness. Thus, future endeavours will also probe into alternative spike encoding techniques to further refine system performance.

Chapter 7

Summary and Future Work

In this concluding chapter, we take a moment to reflect upon the journey of this doctoral dissertation. We revisit the foundational background that ignited the inception of this research, recalling the primary motivations that drove our inquiries and summarizing the pivotal contributions we have made in this domain. As with any rigorous scholarly endeavour, our work has its limitations, which we candidly acknowledge and discuss, all while charting out avenues for future explorations. Beyond the theoretical, we underscore the real-world applicability of our findings, elucidating the broader societal implications and the transformative potential our research holds.

7.1 Summary

The pneumonia pandemic has seen the application of deep learning techniques for detection, mainly through the analysis of chest X-rays and CT images of infected individuals. However, conventional Artificial Neural Networks (ANNs), often used in pneumonia detection systems, require substantial computing power and energy, making them less ideal for edge-based detection and monitoring. As a more effective alternative, Spiking Neural Networks (SNNs) can be leveraged for real-time pneumonia detection on the edge, thanks to their lower computational complexity and energy-efficient design.

AI-based biomedical systems raise privacy and security concerns due to the need for sharing medical images with cloud servers and potential data leakage over the network. The challenge lies in processing high-precision biomedical data like lung X-ray images on resource-constrained edge platforms, requiring the scaling of these AI-enabled platforms for efficiency and reliability. Conventional Artificial Neural Networks (ANNs) are energy and computation-intensive and not ideal for edge-based pneumonia detection. Hence, the exploration of spike-based computing algorithms as an alternative is essential for more robust and efficient solutions.

This dissertation presents algorithms and architecture for scaling pneumonia detection inference on reconfigurable AI-enabled system, including three significant contributions:

- A Client Selection Secure Collaborative Learning (CSSCL) algorithm offers a privacy-preserving method through a distributed learning architecture. By sharing only model weights and processing data locally, it maintains data privacy.
- The AIRBiS system, an energy-efficient reconfigurable tool, employs an FPGA cluster for swift and precise pneumonia detection from chest X-ray images by parallelizing the inference process.
- An event-driven, energy-efficient inference approach utilizes a fault-tolerant spike routing mechanism. Integrated within a neuromorphic system featuring a scalable 3D-NoC processor and 3D mesh architecture, this method ensures quicker inference and minimized power usage, showing only slight accuracy differences when juxtaposed with traditional ANN methods.

7.2 Future Work

This dissertation describes a scaling pneumonia detection system using a robust privacy-preserving method. However, there are limitations and opportunities for improvement.

- Security Concerns: The proposed collaborative learning algorithms provide robust client selection, but there is a security issue of model leakage during transmission. A possible solution is blockchain technology, which enhances the security of the overall system.
- Cost Constraints: The current prototype system involves high-end hardware for inference chest X-ray images, which has led to cost concerns. Future work needs to address the customization of hardware, meeting specific needs while reducing the cost.
- Efficiency of Training: The current solution employs a Spiking Neural Network (SNN), which, despite being more energy-efficient and faster, lags behind in terms of training efficiency when compared to the ANN-based approach. Consequently, future studies should focus on developing improved training methods for SNNs.

7.3 Significance and Contributions to the Society

The significance of this research lies in its creation of a powerful AI-based pneumonia detection system powered by FPGA for secure learning and easy adaptability. Its potential contributions to society are manifold:

- Image Classification: Capable of handling complex tasks and processing large volumes of images concurrently, this system offers detailed results, enhancing image understanding and interpretation.
- Medical Diagnostics: Particularly effective with high-quality images, potentially leading to more accurate diagnoses.
- Edge Computing: Processing complex tasks near the data source, the system improves speed and security. Its scalability makes it adaptable to vari-

ous applications, from IoT devices to autonomous vehicles, underscoring its wide-ranging potential impact.

Overall, This dissertation presents a powerful pneumonia detection system. This system uses AI and FPGA, allowing it to learn securely and be deployed flexibly. While this dissertation has made important progress in pneumonia detection, it's clear there are areas that need improvement for greater efficiency, security, cost-effectiveness, and training speed. The approaches we have developed are not confined to the laboratory research. They could be applied to real-world applications in specific fields like healthcare, technology, etc., offering innovative solutions to challenges that resonate with our everyday lives.

List of Publications

Major Journals

1. **Jiangkun Wang**, Ogbodo Mark Ikechukwu, Khanh N. Dang, and Abderazek Ben Abdallah. "Spike-Event X-ray Image Classification for 3D-NoC-Based Neuromorphic Pneumonia Detection", Electronics, vol. 11, no. 24, p. 4157, 2022. doi: 10.3390/electronics11244157.

Major conferences

1. **Jiangkun Wang**, Miyuka Nakamura, and Abderazek Ben Abdallah, "Efficient AI-Enabled Pneumonia Detection in Chest X-ray Images", in 2022 IEEE 4th Global Conference on Life Sciences and Technologies (LifeTech), Osaka, Japan, March 7-9, 2022, Proceedings, pp. 470-474. IEEE, 2022. doi: 10.1109/LifeTech53646.2022.9754850.

2. **Jiangkun Wang**, Khanh N. Dang, and Abderazek Ben Abdallah. "Scaling Deep-Learning Pneumonia Detection Inference on a Reconfigurable Self-Contained Hardware Platform". in 2023 IEEE 6th International Conference on Electronics Technology (ICET), Chengdu, China, May 12-15, 2023, Proceedings, 1116-1121. IEEE, 2023. (**Best Student Paper Award**)

3. **Jiangkun Wang**, Zhishang Wang, and Abderazek Ben Abdallah. "A Robust Client Selection Based Secure Collaborative Learning Algorithm for Pneumonia Detection". in 2023 IEEE 6th International Conference on Knowl-

edge Innovation and Invention (ICKII), Sapporo, Hokkaido, Japan, August 11-13, 2023, Proceedings. IEEE, 2023.

Presentations

1. Oral presentation "Efficient AI-Enabled Pneumonia Detection in Chest X-ray Images" in 2022 IEEE 4th Global Conference on Life Sciences and Technologies on March 8, 2022, at Osaka, Japan.

2. Oral presentation "Scaling Deep-Learning Pneumonia Detection Inference on a Reconfigurable Self-Contained Hardware Platform" in the 2023 International Conference on Electronics Technology (ICET) on May 13, 2023, in Chengdu, China. (**Best Student Paper Award**)

3. Oral presentation "A Robust Client Selection Based Secure Collaborative Learning Algorithm for Pneumonia Detection" in 2023 IEEE 6th International Conference on Knowledge Innovation and Invention (ICKII) on August 12, 2023, at Sapporo, Hokkaido, Japan.

References

[1] L. Muhammad, E. A. Algehyne, S. S. Usman, A. Ahmad, C. Chakraborty, and I. Mohammed, "Supervised machine learning models for prediction of covid-19 infection using epidemiology dataset," *SN Computer Science*, vol. 2, no. 1, pp. 1–13, 2020, doi: 10.1007/s42979-020-00394-7.

[2] M. Mosley, *Covid-19: what you need to know about the coronavirus and the race for the vaccine.* Short Books Ltd, 2020, oCLC: 1180152280.

[3] D. J. Bell, "COVID-19 | Radiology Reference Article | Radiopaedia.org," 2020. [Online]. Available: https://radiopaedia.org/articles/covid-19-4?lang=us

[4] A. Narin, C. Kaya, and Z. Pamuk, "Automatic detection of coronavirus disease (covid-19) using x-ray images and deep convolutional neural networks," *Pattern Analysis and Applications*, vol. 24, pp. 1207–1220, 2021, doi: 10.1007/s10044-021-00984-y.

[5] L. P. Soares and C. P. Soares, "Automatic detection of covid-19 cases on x-ray images using convolutional neural networks," *arXiv preprint arXiv:2007.05494*, 2020, doi: 10.48550/arXiv.2007.05494.

[6] H. B. McMahan, E. Moore, D. Ramage, S. Hampson *et al.*, "Communication-efficient learning of deep networks from decentralized data," *arXiv preprint arXiv:1602.05629*, 2016, doi: 10.48550/arXiv.1602.05629.

[7] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Chal-

lenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020, doi: 10.1109/MSP.2020.2975749.

[8] Z. Wang, M. Ogbodo, H. Huang, C. Qiu, M. Hisada, and A. B. Abdallah, "Aebis: Ai-enabled blockchain-based electric vehicle integration system for power management in smart grid platform," *IEEE Access*, vol. 8, pp. 226 409–226 421, 2020.

[9] S. Cass, "Taking ai to the edge: Google's tpu now comes in a maker-friendly package," *IEEE Spectrum*, vol. 56, no. 5, pp. 16–17, 2019, doi: 10.1109/MSPEC.2019.8701189.

[10] A. Yazdanbakhsh, K. Seshadri, B. Akin, J. Laudon, and R. Narayanaswami, "An evaluation of edge tpu accelerators for convolutional neural networks," *arXiv preprint arXiv:2102.10423*, 2021, doi: 10.48550/arXiv.2102.10423.

[11] K. Park, W. Jang, W. Lee, K. Nam, K. Seong, K. Chai, and W.-S. Li, "Real-time mask detection on google edge tpu," *arXiv preprint arXiv:2010.04427*, 2020, doi: 10.48550/arXiv.2010.04427.

[12] S. Mittal, "A survey on optimized implementation of deep learning models on the nvidia jetson platform," *Journal of Systems Architecture*, vol. 97, pp. 428–442, 2019, doi: 10.1016/j.sysarc.2019.01.011.

[13] Y. Ukidave, D. Kaeli, U. Gupta, and K. Keville, "Performance of the nvidia jetson tk1 in hpc," in *2015 IEEE International Conference on Cluster Computing.* IEEE, 2015, pp. 533–534, doi: 10.1109/CLUSTER.2015.147.

[14] W. Zhao, H. Fu, W. Luk, T. Yu, S. Wang, B. Feng, Y. Ma, and G. Yang, "F-cnn: An fpga-based framework for training convolutional neural networks," in *2016 IEEE 27th International Conference on Application-specific Systems, Architectures and Processors (ASAP).* IEEE, 2016, pp. 107–114, doi: 10.1109/ASAP.2016.7760779.

[15] S. Park and T. Suh, “Speculative backpropagation for cnn parallel training,” *IEEE Access*, 2020, doi: 10.1109/ACCESS.2020.3040849.

[16] B.-Y. Lin and C.-S. Chen, “Two parallel deep convolutional neural networks for pedestrian detection,” in *2015 International Conference on Image and Vision Computing New Zealand (IVCNZ).* IEEE, 2015, pp. 1–6, doi: 10.1109/IVCNZ.2015.7761510.

[17] A. B. Abdallah, H. Huang, N. K. Dang, and J. Song, “Ai processor,” Nov. 2020, japanese Patent Application Laid-Open No 2020-194733.

[18] O. M. Ikechukwu, “On the design of adaptive digital neuromorphic system,” Ph.D. dissertation, University of Aizu, 2022, doi: 10.15016/00000204.

[19] O. M. Ikechukwu, K. N. Dang, and A. B. Abdallah, “On the design of a fault-tolerant scalable three dimensional noc-based digital neuromorphic system with on-chip learning,” *IEEE Access*, vol. 9, pp. 64 331–64 345, 2021, doi: 10.1109/ACCESS.2021.3071089.

[20] D. Lepikhin, H. Lee, Y. Xu, D. Chen, O. Firat, Y. Huang, M. Krikun, N. Shazeer, and Z. Chen, “Gshard: Scaling giant models with conditional computation and automatic sharding,” *arXiv preprint arXiv:2006.16668*, 2020, doi: 10.48550/arXiv.2006.16668.

[21] Y. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, “Efficient backprop,” in *Neural Networks: Tricks of the Trade, This Book is an Outgrowth of a 1996 NIPS Workshop.* Berlin, Heidelberg: Springer-Verlag, 1998, p. 9‑50, doi: 10.1007/978-3-642-35289-8_3.

[22] A. Krizhevsky, G. Hinton *et al.*, “Learning multiple layers of features from tiny images,” pp. 32–33, 2009. [Online]. Available: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf

[23] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014, doi: 10.48550/arXiv.1409.1556.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*. Springer, 2016, pp. 630–645, doi: 10.1007/978-3-319-46493-0_38.

[25] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. Van Der Laak, B. Van Ginneken, and C. I. Sánchez, "A survey on deep learning in medical image analysis," *Medical image analysis*, vol. 42, pp. 60–88, 2017, doi: 10.1016/j.media.2017.07.005.

[26] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun, "Dermatologist-level classification of skin cancer with deep neural networks," *nature*, vol. 542, no. 7639, pp. 115–118, 2017, doi: 10.1038/nature21056.

[27] V. Gulshan, L. Peng, M. Coram, M. C. Stumpe, D. Wu, A. Narayanaswamy, S. Venugopalan, K. Widner, T. Madams, J. Cuadros *et al.*, "Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs," *Jama*, vol. 316, no. 22, pp. 2402–2410, 2016, doi: 10.1001/jama.2016.17216.

[28] P. Rajpurkar, J. Irvin, K. Zhu, B. Yang, H. Mehta, T. Duan, D. Ding, A. Bagul, C. Langlotz, K. Shpanskaya *et al.*, "Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning," *arXiv preprint arXiv:1711.05225*, 2017, doi: 10.48550/arXiv.1711.05225.

[29] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" *Advances in neural information processing systems*, vol. 27, 2014, doi: 10.48550/arXiv.1411.1792.

[30] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, and J. Liang, "Convolutional neural networks for medical image analysis: Full training or fine tuning?" *IEEE transactions on medical imaging*, vol. 35, no. 5, pp. 1299–1312, 2016, doi: 10.1109/tmi.2016.2535302.

[31] M. Ghafoorian, A. Mehrtash, T. Kapur, N. Karssemeijer, E. Marchiori, M. Pesteie, C. R. Guttmann, F.-E. de Leeuw, C. M. Tempany, B. Van Ginneken *et al.*, "Transfer learning for domain adaptation in mri: Application in brain lesion segmentation," in *Medical Image Computing and Computer Assisted Intervention- MICCAI 2017: 20th International Conference, Quebec City, QC, Canada, September 11-13, 2017, Proceedings, Part III 20.* Springer, 2017, pp. 516–524, doi: 10.1007/978-3-319-66179-7_59.

[32] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010: 19th International Conference on Computational StatisticsParis France, August 22-27, 2010 Keynote, Invited and Contributed Papers.* Springer, 2010, pp. 177–186, doi: 10.1007/978-3-7908-2604-3_16.

[33] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014, doi: 10.48550/arXiv.1412.6980.

[34] T. Tieleman, G. Hinton *et al.*, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural networks for machine learning*, vol. 4, no. 2, pp. 26–31, 2012.

[35] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[36] M. Courbariaux, Y. Bengio, and J.-P. David, "Binaryconnect: Training deep neural networks with binary weights during propagations," *Ad-*

vances in neural information processing systems, vol. 28, 2015, doi: 10.48550/arXiv.1511.00363.

[37] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Quantized neural networks: Training neural networks with low precision weights and activations," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 6869–6898, 2017, doi: 10.48550/arXiv.1609.07061.

[38] A. Zhou, A. Yao, Y. Guo, L. Xu, and Y. Chen, "Incremental network quantization: Towards lossless cnns with low-precision weights," *arXiv preprint arXiv:1702.03044*, 2017, doi: 10.48550/arXiv.1702.03044.

[39] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015, doi: 10.48550/arXiv.1510.00149.

[40] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, "Pruning convolutional neural networks for resource efficient inference," *arXiv preprint arXiv:1611.06440*, 2016, doi: 10.48550/arXiv.1611.06440.

[41] C. Buciluǎ, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2006, pp. 535–541, doi: 10.1145/1150402.1150464.

[42] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015, doi: 10.48550/arXiv.1503.02531.

[43] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "A survey of model compression and acceleration for deep neural networks," *arXiv preprint arXiv:1710.09282*, 2017, doi: 10.48550/arXiv.1710.09282.

[44] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural

networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017, doi: 10.48550/arXiv.1704.04861.

[45] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks," *Advances in neural information processing systems*, vol. 29, 2016, doi: 10.48550/arXiv.1602.02505.

[46] F. Li, B. Zhang, and B. Liu, "Ternary weight networks," *arXiv preprint arXiv:1605.04711*, 2016.

[47] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, 2017, doi: 10.48550/arXiv.1703.09039.

[48] W. Maass, "Networks of spiking neurons: the third generation of neural network models," *Neural networks*, vol. 10, no. 9, pp. 1659–1671, 1997, doi: https://doi.org/10.1016/S0893-6080(97)00011-7.

[49] M. Pfeiffer and T. Pfeil, "Deep learning with spiking neurons: Opportunities and challenges," *Frontiers in neuroscience*, vol. 12, p. 774, 2018, doi: 10.3389/fnins.2018.00774.

[50] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, 2014, doi: 10.1126/science.1254642.

[51] M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain *et al.*, "Loihi: A neuromorphic manycore processor with on-chip learning," *Ieee Micro*, vol. 38, no. 1, pp. 82–99, 2018, doi: 10.1109/MM.2018.112130359.

[52] G.-q. Bi and M.-m. Poo, "Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic

cell type," *Journal of neuroscience*, vol. 18, no. 24, pp. 10 464–10 472, 1998, doi: 10.1523/JNEUROSCI.18-24-10464.1998.

[53] P. U. Diehl and M. Cook, "Unsupervised learning of digit recognition using spike-timing-dependent plasticity," *Frontiers in computational neuroscience*, vol. 9, p. 99, 2015, doi: 10.3389/fncom.2015.00099.

[54] G. Orchard, A. Jayawant, G. K. Cohen, and N. Thakor, "Converting static image datasets to spiking neuromorphic datasets using saccades," *Frontiers in neuroscience*, vol. 9, p. 437, 2015, doi: 10.3389/fnins.2015.00437.

[55] S. B. Furber, F. Galluppi, S. Temple, and L. A. Plana, "The spinnaker project," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 652–665, 2014, doi: 10.1109/JPROC.2014.2304638.

[56] R. Raina, A. Madhavan, and A. Y. Ng, "Large-scale deep unsupervised learning using graphics processors," in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 873–880, doi: 10.1145/1553374.1553486.

[57] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: a system for large-scale machine learning," in *Osdi*, ser. OSDI'16, vol. 16, Savannah, GA, USA, 2016, pp. 265–283, doi: 10.48550/arXiv.1605.08695.

[58] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019, doi: 10.48550/arXiv.1912.01703.

[59] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast

feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*, 2014, pp. 675–678, doi: 10.1145/2647868.2654889.

[60] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3d u-net: learning dense volumetric segmentation from sparse annotation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2016: 19th International Conference, Athens, Greece, October 17-21, 2016, Proceedings, Part II 19*. Springer, 2016, pp. 424–432, doi: 10.1007/978-3-319-46723-8_49.

[61] S. Christodoulidis, M. Anthimopoulos, L. Ebner, A. Christe, and S. Mougiakakou, "Multisource transfer learning with convolutional neural networks for lung pattern analysis," *IEEE journal of biomedical and health informatics*, vol. 21, no. 1, pp. 76–84, 2016, doi: 10.1109/JBHI.2016.2636929.

[62] K. Compton and S. Hauck, "Reconfigurable computing: a survey of systems and software," *ACM Computing Surveys (csuR)*, vol. 34, no. 2, pp. 171–210, 2002, doi: 10.1145/508352.508353.

[63] S. I. Venieris and C.-S. Bouganis, "fpgaconvnet: A framework for mapping convolutional neural networks on fpgas," in *2016 IEEE 24th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*. IEEE, 2016, pp. 40–47, doi: 10.1109/FCCM.2016.22.

[64] A. DeHon, R. Huang, and J. Wawrzynek, "Hardware-assisted fast routing," in *Proceedings. 10th Annual IEEE Symposium on Field-Programmable Custom Computing Machines*. IEEE, 2002, pp. 205–215, doi: 10.1109/FPGA.2002.1106675.

[65] C. Zhang, P. Li, G. Sun, Y. Guan, B. Xiao, and J. Cong, "Optimizing fpga-based accelerator design for deep convolutional neural networks," in *Proceedings of the 2015 ACM/SIGDA international sym-*

posium on field-programmable gate arrays, 2015, pp. 161–170, doi: 10.1145/2684746.2689060.

[66] J. Zhu, L. Wang, H. Liu, S. Tian, Q. Deng, and J. Li, “An efficient task assignment framework to accelerate dpu-based convolutional neural network inference on fpgas,” *IEEE Access*, vol. 8, pp. 83 224–83 237, 2020, doi: 10.1109/ACCESS.2020.2988311.

[67] S. Bernabé, C. González, A. Fernández, and U. Bhangale, “Portability and acceleration of deep learning inferences to detect rapid earthquake damage from vhr remote sensing images using intel openvino toolkit,” *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 14, pp. 6906–6915, 2021, doi: 10.1109/JSTARS.2021.3075961.

[68] R. Dagli and S. Eken, “Deploying a smart queuing system on edge with intel openvino toolkit,” *Soft Computing*, vol. 25, no. 15, pp. 10 103–10 115, 2021, doi: 10.1007/s00500-021-05891-2.

[69] T. J. Todman, G. A. Constantinides, S. J. Wilton, O. Mencer, W. Luk, and P. Y. Cheung, “Reconfigurable computing: architectures and design methods,” *IEE Proceedings-Computers and Digital Techniques*, vol. 152, no. 2, pp. 193–207, 2005, doi: 10.1049/ip-cdt:20045086.

[70] Y. Chen, T. Luo, S. Liu, S. Zhang, L. He, J. Wang, L. Li, T. Chen, Z. Xu, N. Sun *et al.*, “Dadiannao: A machine-learning supercomputer,” in *2014 47th Annual IEEE/ACM International Symposium on Microarchitecture*. IEEE, 2014, pp. 609–622, doi: 10.1109/MICRO.2014.58.

[71] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers *et al.*, “In-datacenter performance analysis of a tensor processing unit,” in *Proceedings of the 44th annual international symposium on computer architecture*, 2017, pp. 1–12, doi: 10.48550/arXiv.1704.04760.

[72] G. Ltd, “Graphcore’s staff, investors and mission.” [Online]. Available: https://www.graphcore.ai/about

[73] S. Zhang, Z. Du, L. Zhang, H. Lan, S. Liu, L. Li, Q. Guo, T. Chen, and Y. Chen, “Cambricon-x: An accelerator for sparse neural networks,” in *2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. Taipei, Taiwan: IEEE, 2016, pp. 1–12, doi: 10.1109/MICRO.2016.7783723.

[74] Y. Tian, Y. Gong, P. Liu, S. Wang, X. Xu, X. Wang, and Y. Huang, “Infection prevention strategy in operating room during coronavirus disease 2019 (covid-19) outbreak,” *Chinese Medical Sciences Journal*, vol. 35, no. 2, pp. 114–120, 2020, doi: 10.24920/003739.

[75] N. Aydin, Ç. Cihan, T. Us, Y. Öz, Ç. Öztunalı, Ş. Yılmaz, F. Alataş, S. Erginel, E. Kurt, G. Ak *et al.*, “Correlation of indeterminate lesions of covid-19 pneumonia detected on computed tomography with rt-pcr results,” *Current Medical Imaging*, vol. 18, no. 8, pp. 862–868, 2022, doi: 10.2174/1573405618666220111095357.

[76] Y. Zhou, L. Huang, T. Zhou, and L. Shao, “Many-to-one distribution learning and k-nearest neighbor smoothing for thoracic disease identification,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 1, 2021, pp. 768–776, doi: 10.1609/aaai.v35i1.16158.

[77] A. Asraf, M. Z. Islam, M. R. Haque, and M. M. Islam, “Deep learning applications to combat novel coronavirus (covid-19) pandemic,” *SN Computer Science*, vol. 1, pp. 1–7, 2020, doi: 10.1007/s42979-020-00383-w.

[78] M. Nakamura, J. Wang, S. Phea, and A. B. Abdallah, “Comprehensive study of coronavirus disease 2019 (covid-19) classification based on deep convolution neural networks,” in *SHS Web of Conferences*, vol. 102.

Aizuwakamatsu, Japan, January 27-30: EDP Sciences, 2021, p. 04007, doi: 10.1051/shsconf/202110204007.

[79] A. Mustafa and M. Rahimi Azghadi, “Automated machine learning for healthcare and clinical notes analysis,” *Computers*, vol. 10, no. 2, p. 24, 2021, doi: 10.3390/computers10020024.

[80] G. Latif, H. Morsy, A. Hassan, and J. Alghazo, “Novel coronavirus and common pneumonia detection from ct scans using deep learning-based extracted features,” *Viruses*, vol. 14, no. 8, p. 1667, 2022, doi: 10.3390/v14081667.

[81] Y. Zhou, T. Zhou, T. Zhou, H. Fu, J. Liu, and L. Shao, “Contrast-attentive thoracic disease recognition with dual-weighting graph reasoning,” *IEEE Transactions on Medical Imaging*, vol. 40, no. 4, pp. 1196–1206, 2021, doi: 10.1109/TMI.2021.3049498.

[82] I. Shiri, H. Arabi, Y. Salimi, A. Sanaat, A. Akhavanallaf, G. Hajianfar, D. Askari, S. Moradi, Z. Mansouri, M. Pakbin *et al.*, “Coli-net: Deep learning-assisted fully automated covid-19 lung and infection pneumonia lesion detection and segmentation from chest computed tomography images,” *International journal of imaging systems and technology*, vol. 32, no. 1, pp. 12–25, 2022, doi: 10.1002/ima.22672.

[83] C. Ieracitano, N. Mammone, M. Versaci, G. Varone, A.-R. Ali, A. Armentano, G. Calabrese, A. Ferrarelli, L. Turano, C. Tebala *et al.*, “A fuzzy-enhanced deep learning approach for early detection of covid-19 pneumonia from portable chest x-ray images,” *Neurocomputing*, vol. 481, pp. 202–215, 2022, doi: 10.1016/j.neucom.2022.01.055.

[84] M. A. Rahman, M. S. Hossain, N. A. Alrajeh, and N. Guizani, “B5g and explainable deep learning assisted healthcare vertical at the edge: Covid-i9 perspective,” *IEEE Network*, vol. 34, no. 4, pp. 98–105, 2020, doi: 10.1109/MNET.011.2000353.

[85] K. Abdelouahab, M. Pelcat, J. Serot, and F. Berry, "Accelerating cnn inference on fpgas: A survey," *arXiv preprint arXiv:1806.01683*, 2018, doi: 10.1109/MNET.011.2000353.

[86] M. U. Ashraf, A. Hannan, S. M. Cheema, Z. Ali, A. Alofi *et al.*, "Detection and tracking contagion using iot-edge technologies: Confronting covid-19 pandemic," in *2020 international conference on electrical, communication, and computer engineering (ICECCE).* IEEE, 2020, pp. 1–6, doi: 10.1109/ICECCE49384.2020.9179284.

[87] L. Fu, C. Guo, and W. Luk, "Fpga-accelerated agent-based simulation for covid-19," in *2021 IEEE 3rd International Conference on Artificial Intelligence Circuits and Systems (AICAS).* IEEE, 2021, pp. 1–4, doi: 10.1109/AICAS51828.2021.9458570.

[88] G. Goel, A. Gondhalekar, J. Qi, Z. Zhang, G. Cao, and W. Feng, "Computecovid19+: Accelerating covid-19 diagnosis and monitoring via high-performance deep learning on ct images," in *50th International Conference on Parallel Processing*, 2021, pp. 1–11, doi: 10.1145/3472456.3473523.

[89] C. Shen, K. Zhang, and J. Tang, "A covid-19 detection algorithm using deep features and discrete social learning particle swarm optimization for edge computing devices," *ACM Transactions on Internet Technology (TOIT)*, vol. 22, no. 3, pp. 1–17, 2021, doi: 10.1145/3453170.

[90] S. Yaman, B. Karakaya, and Y. Erol, "A novel normalization algorithm to facilitate pre-assessment of covid-19 disease by improving accuracy of cnn and its fpga implementation," *Evolving Systems*, vol. 14, no. 4, pp. 581–591, 2023, doi: 10.1007/s12530-022-09419-3.

[91] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. Quek, and H. V. Poor, "Federated learning with differential privacy: Algorithms

and performance analysis," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3454–3469, 2020, doi: 10.1109/TIFS.2020.2988575.

[92] S. Truex, L. Liu, K.-H. Chow, M. E. Gursoy, and W. Wei, "Ldp-fed: Federated learning with local differential privacy," in *Proceedings of the Third ACM International Workshop on Edge Systems, Analytics and Networking*, 2020, pp. 61–66, doi: 10.1145/3378679.3394533.

[93] R. Hu, Y. Guo, H. Li, Q. Pei, and Y. Gong, "Personalized federated learning with differential privacy," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9530–9539, 2020, doi: 10.1109/JIOT.2020.2991416.

[94] M. Mansouri, M. Onen, W. B. Jaballah, and M. Conti, "Sok: Secure aggregation based on cryptographic schemes for federated learning," *Proc. Priv. Enhancing Technol*, no. 1, pp. 140–157, 2023.

[95] A. Mondal, Y. More, R. H. Rooparaghunath, and D. Gupta, "Poster: Flatee: Federated learning across trusted execution environments," in *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2021, pp. 707–709, doi: 10.1109/EuroSP51992.2021.00054.

[96] F. Mo, H. Haddadi, K. Katevas, E. Marin, D. Perino, and N. Kourtellis, "Ppfl: privacy-preserving federated learning with trusted execution environments," in *Proceedings of the 19th annual international conference on mobile systems, applications, and services*, 2021, pp. 94–108, doi: 10.1145/3458864.3466628.

[97] Y. Chen, F. Luo, T. Li, T. Xiang, Z. Liu, and J. Li, "A training-integrity privacy-preserving federated learning scheme with trusted execution environment," *Information Sciences*, vol. 522, pp. 69–79, 2020, doi: 10.1016/j.ins.2020.02.037.

[98] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving com-

munication efficiency," *arXiv preprint arXiv:1610.05492*, 2016, doi: 10.48550/arXiv.1610.05492.

[99] Z. Wang and A. B. Abdallah, "A robust multi-stage power consumption prediction method in a semi-decentralized network of electric vehicles," *IEEE Access*, vol. 10, pp. 37 082–37 096, 2022, doi: 10.1109/ACCESS.2022.3163455.

[100] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Blockchain and federated learning for privacy-preserved data sharing in industrial iot," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 4177–4186, 2019, doi: 10.1109/TII.2019.2942190.

[101] A. I. Khan, J. L. Shah, and M. M. Bhat, "Coronet: A deep neural network for detection and diagnosis of covid-19 from chest x-ray images," *Computer methods and programs in biomedicine*, vol. 196, p. 105581, 2020, doi: 10.1109/10.1016/j.cmpb.2020.105581.

[102] X. Xu, Y. Ding, S. X. Hu, M. Niemier, J. Cong, Y. Hu, and Y. Shi, "Scaling for edge inference of deep neural networks," *Nature Electronics*, vol. 1, no. 4, pp. 216–222, 2018, doi: 10.1038/s41928-018-0059-3.

[103] X. Wei, C. H. Yu, P. Zhang, Y. Chen, Y. Wang, H. Hu, Y. Liang, and J. Cong, "Automated systolic array architecture synthesis for high throughput cnn inference on fpgas," in *Proceedings of the 54th Annual Design Automation Conference 2017*, 2017, pp. 1–6, doi: 10.1145/3061639.3062207.

[104] K. Ito, K. Iizuka, K. Hironaka, Y. Hu, M. Koibuchi, and H. Amano, "Improving the performance of circuit-switched interconnection network for a multi-fpga system," *IEICE TRANSACTIONS on Information and Systems*, vol. 104, no. 12, pp. 2029–2039, 2021, doi: 10.1587/transinf.2021PAP0002.

[105] P. Lakhani and B. Sundaram, "Deep learning at chest radiography: automated classification of pulmonary tuberculosis by using convolutional neural

networks," *Radiology*, vol. 284, no. 2, pp. 574–582, 2017, doi: 10.1148/radiol.2017162326.

[106] T. H. Vu, R. Murakami, Y. Okuyama, and A. B. Abdallah, "Efficient optimization and hardware acceleration of cnns towards the design of a scalable neuro inspired architecture in hardware," in *2018 IEEE international conference on big data and smart computing (BigComp)*. IEEE, 2018, pp. 326–332, doi: 10.1109/BigComp.2018.00055.

[107] A. B. Ahmed, Y. Kimezawa, and A. B. Abdallah, "Hardware/software prototyping of dependable real-time system for elderly health monitoring," in *2013 World Congress on Computer and Information Technology (WCCIT)*. IEEE, 2013, pp. 1–6, doi: 10.1109/WCCIT.2013.6618696.

[108] T. Ai, Z. Yang, H. Hou, C. Zhan, C. Chen, W. Lv, Q. Tao, Z. Sun, and L. Xia, "Correlation of chest ct and rt-pcr testing for coronavirus disease 2019 (covid-19) in china: a report of 1014 cases," *Radiology*, vol. 296, no. 2, pp. E32–E40, 2020, doi: 10.1148/radiol.2020200642.

[109] Y. Fang, H. Zhang, J. Xie, M. Lin, L. Ying, P. Pang, and W. Ji, "Sensitivity of chest ct for covid-19: comparison to rt-pcr," *Radiology*, vol. 296, no. 2, pp. E115–E117, 2020, doi: 10.1148/radiol.2020200432.

[110] S. A. Harmon, T. H. Sanford, S. Xu, E. B. Turkbey, H. Roth, Z. Xu, D. Yang, A. Myronenko, V. Anderson, A. Amalou *et al.*, "Artificial intelligence for the detection of covid-19 pneumonia on chest ct using multinational datasets," *Nature communications*, vol. 11, no. 1, p. 4080, 2020, doi: 10.1038/s41467-020-17971-2.

[111] J. Wang, Z. Wang, and A. B. Abdallah, "A robust client selection based secure collaborative learning algorithm for pneumonia detection," in *2023 IEEE 6th International Conference on Knowledge Innovation and Invention (ICKII)*. IEEE, 2023.

[112] J. Wang, M. Nakamura, and A. B. Abdallah, "Efficient ai-enabled pneumonia detection in chest x-ray images," in *2022 IEEE 4th Global Conference on Life Sciences and Technologies (LifeTech), Osaka, Japan.* Osaka, Japan, March 7-9: IEEE, 2022, pp. 470–474, doi: 10.1109/LifeTech53646.2022.9754850.

[113] S. Phea, Z. Wang, J. Wang, and A. B. Abdallah, "Optimization and implementation of a collaborative learning algorithm for an ai-enabled real-time biomedical system," in *SHS Web of Conferences*, vol. 102. Aizuwakamatsu, Japan, January 27-30: EDP Sciences, 2021, p. 04017, doi: 10.1051/shsconf/202110204017.

[114] M. E. Chowdhury, T. Rahman, A. Khandakar, R. Mazhar, M. A. Kadir, Z. B. Mahbub, K. R. Islam, M. S. Khan, A. Iqbal, N. Al Emadi *et al.*, "Can ai help in screening viral and covid-19 pneumonia?" *Ieee Access*, vol. 8, pp. 132 665–132 676, 2020, doi: 10.1109/ACCESS.2020.3010287.

[115] T. Rahman, A. Khandakar, Y. Qiblawey, A. Tahir, S. Kiranyaz, S. B. A. Kashem, M. T. Islam, S. Al Maadeed, S. M. Zughaier, M. S. Khan *et al.*, "Exploring the effect of image enhancement techniques on covid-19 detection using chest x-ray images," *Computers in biology and medicine*, vol. 132, p. 104319, 2021, doi: 10.1016/j.compbiomed.2021.104319.

[116] "COVID-19 Radiography Database," 2021. [Online]. Available: https://kaggle.com/tawsifurrahman/covid19-radiography-database

[117] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for lvcsr using rectified linear units and dropout," in *2013 IEEE international conference on acoustics, speech and signal processing.* IEEE, 2013, pp. 8609–8613, doi: 10.1109/ICASSP.2013.6639346.

[118] A. Abbas, M. M. Abdelsamea, and M. M. Gaber, "Classification of covid-19 in chest x-ray images using detrac deep convolutional neural network,"

Applied Intelligence, vol. 51, no. 2, pp. 854–864, 2021, doi: 10.1007/s10489-020-01829-7.

[119] M. Z. Che Azemin, R. Hassan, M. I. Mohd Tamrin, and M. A. Md Ali, “Covid-19 deep learning prediction model using publicly available radiologist-adjudicated chest x-ray images as training data: preliminary findings,” *International Journal of Biomedical Imaging*, vol. 2020, 2020, doi: 10.1155/2020/8828855.

[120] T.-C. Lin and H.-C. Lee, “Covid-19 chest radiography images analysis based on integration of image preprocess, guided grad-cam, machine learning and risk management,” in *Proceedings of the 4th International Conference on Medical and Health Informatics*, 2020, pp. 281–288, doi: 10.1145/3418094.3418096.

[121] E. F. Ohata, G. M. Bezerra, J. V. S. das Chagas, A. V. L. Neto, A. B. Albuquerque, V. H. C. de Albuquerque, and P. P. Reboucas Filho, “Automatic detection of covid-19 infection using chest x-ray images through transfer learning,” *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 1, pp. 239–248, 2020, doi: 10.1109/JAS.2020.1003393.

[122] I. D. Apostolopoulos and T. A. Mpesiana, “Covid-19: automatic detection from x-ray images utilizing transfer learning with convolutional neural networks,” *Physical and Engineering Sciences in Medicine*, vol. 43, no. 2, pp. 635–640, 2020, doi: 10.1007/s13246-020-00865-4.

[123] R. Jain, M. Gupta, S. Taneja, and D. J. Hemanth, “Deep learning based detection and analysis of covid-19 on chest x-ray images,” *Applied Intelligence*, vol. 51, no. 3, pp. 1690–1700, 2021, doi: 10.1007/s10489-020-01902-1.

[124] J. Wang, O. M. Ikechukwu, K. N. Dang, and A. B. Abdallah, “Spike-event x-ray image classification for 3d-NoC-based neuromorphic pneumonia detec-

tion," *Electronics*, vol. 11, no. 24, p. 4157, Dec 2022, doi: 10.3390/electronics11244157.

[125] J. Wang, K. N. Dang, and A. B. Abdallah, "Scaling deep-learning pneumonia detection inference on a reconfigurable self-contained hardware platform," in *2023 IEEE 6th International Conference on Electronics Technology (ICET).* Chengdu, China, May 12-15: IEEE, 2023, pp. 1116–1121, doi: 10.1109/ICET58434.2023.10212016.

[126] B. E. Bejnordi, M. Veta, P. J. Van Diest, B. Van Ginneken, N. Karssemeijer, G. Litjens, J. A. Van Der Laak, M. Hermsen, Q. F. Manson, M. Balkenhol *et al.*, "Diagnostic assessment of deep learning algorithms for detection of lymph node metastases in women with breast cancer," *Jama*, vol. 318, no. 22, pp. 2199–2210, 2017, doi: 10.1001/jama.2017.14585.

[127] S. Roy, W. Menapace, S. Oei, B. Luijten, E. Fini, C. Saltori, I. Huijben, N. Chennakeshava, F. Mento, A. Sentelli *et al.*, "Deep learning for classification and localization of covid-19 markers in point-of-care lung ultrasound," *IEEE transactions on medical imaging*, vol. 39, no. 8, pp. 2676–2687, 2020, doi: 10.1109/TMI.2020.2994459.

[128] M. Yaseliani, A. Z. Hamadani, A. I. Maghsoodi, and A. Mosavi, "Pneumonia detection proposing a hybrid deep convolutional neural network based on two parallel visual geometry group architectures and machine learning classifiers," *IEEE Access*, vol. 10, pp. 62 110–62 128, 2022, doi: 10.1109/ACCESS.2022.3182498.

[129] M. Yamac, M. Ahishali, A. Degerli, S. Kiranyaz, M. E. Chowdhury, and M. Gabbouj, "Convolutional sparse support estimator-based covid-19 recognition from x-ray images," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 5, pp. 1810–1820, 2021, doi: 10.1109/TNNLS.2021.3070467.

[130] W. Lie, B. Jiang, and W. Zhao, "Obstetric imaging diagnostic platform based on cloud computing technology under the background of smart medical big data and deep learning," *IEEE Access*, vol. 8, pp. 78 265–78 278, 2020, doi: 10.1109/ACCESS.2020.2988563.

[131] B. Alouffi, M. Hasnain, A. Alharbi, W. Alosaimi, H. Alyami, and M. Ayaz, "A systematic literature review on cloud computing security: threats and mitigation strategies," *IEEE Access*, vol. 9, pp. 57 792–57 807, 2021, doi: 10.1109/ACCESS.2021.3073203.

[132] F. Wang, M. Zhang, X. Wang, X. Ma, and J. Liu, "Deep learning for edge computing applications: A state-of-the-art survey," *IEEE Access*, vol. 8, pp. 58 322–58 336, 2020, doi: 10.1109/ACCESS.2020.2982411.

[133] J. Chen and X. Ran, "Deep learning with edge computing: A review," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1655–1674, 2019, doi: 10.1109/JPROC.2019.2921977.

[134] X. Wang, Y. Han, V. C. Leung, D. Niyato, X. Yan, and X. Chen, "Convergence of edge computing and deep learning: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 2, pp. 869–904, 2020, doi: 10.1109/COMST.2020.2970550.

[135] K. Cao, S. Hu, Y. Shi, A. W. Colombo, S. Karnouskos, and X. Li, "A survey on edge and edge-cloud computing assisted cyber-physical systems," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 11, pp. 7806–7819, 2021, doi: 10.1109/TII.2021.3073066.

[136] M. Zhang, F. Zhang, N. D. Lane, Y. Shu, X. Zeng, B. Fang, S. Yan, and H. Xu, "Deep learning in the era of edge computing: Challenges and opportunities," *Fog Computing: Theory and Practice*, pp. 67–78, 2020, doi: 10.1002/9781119551713.ch3.

[137] Q. Liu, J. Lai, and J. Gao, "An efficient channel-aware sparse binarized neural networks inference accelerator," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 3, pp. 1637–1641, 2021, doi: 10.1109/TCSII.2021.3119369.

[138] K. Huang, S. Chen, B. Li, L. Claesen, H. Yao, J. Chen, X. Jiang, Z. Liu, and D. Xiong, "Acceleration-aware fine-grained channel pruning for deep neural networks via residual gating," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 6, pp. 1902–1915, 2021, doi: 10.1109/TCAD.2021.3093835.

[139] C. Xu, S. Jiang, G. Luo, G. Sun, N. An, G. Huang, and X. Liu, "The case for fpga-based edge computing," *IEEE Transactions on Mobile Computing*, vol. 21, no. 7, pp. 2610–2619, 2020, doi: 10.1109/TMC.2020.3041781.

[140] "Coronavirus Update (Live) from COVID-19 Virus Pandemic - Worldometer." [Online]. Available: https://www.worldometers.info/coronavirus/

[141] O. Yuuki, J. Wang, O. M. Ikechukwu, and A. B. Abdallah, "Hardware acceleration of convolution neural network for ai-enabled realtime biomedical system," in *SHS Web of Conferences*, vol. 102. Aizuwakamatsu, Japan, January 27-30: EDP Sciences, 2021, p. 04019, doi: 10.1051/shsconf/202110204019.

[142] R. Krishnamoorthi, "Quantizing deep convolutional networks for efficient inference: A whitepaper," *arXiv preprint arXiv:1806.08342*, 2018, doi: 10.48550/arXiv.1806.08342.

[143] Y. Fu, E. Wu, A. Sirasao, S. Attia, K. Khan, and R. Wittig, "Deep learning with int8 optimization on xilinx devices," *White Paper*, 2016.

[144] J. Gray, "Grvi phalanx: A massively parallel risc-v fpga accelerator accelerator," in *2016 IEEE 24th Annual International Symposium on Field-*

Programmable Custom Computing Machines (FCCM). IEEE, 2016, pp. 17–20, doi: 10.1109/FCCM.2016.12.

[145] C. N. Hesse, "Analysis and comparison of performance and power consumption of neural networks on cpu, gpu, tpu and fpga," 2021.

[146] X. Wu, H. Hui, M. Niu, L. Li, L. Wang, B. He, X. Yang, L. Li, H. Li, J. Tian *et al.*, "Deep learning-based multi-view fusion model for screening 2019 novel coronavirus pneumonia: a multicentre study," *European Journal of Radiology*, vol. 128, p. 109041, 2020, doi: 10.1016/j.ejrad.2020.109041.

[147] R. M. Pereira, D. Bertolini, L. O. Teixeira, C. N. Silla Jr, and Y. M. Costa, "Covid-19 identification in chest x-ray images on flat and hierarchical classification scenarios," *Computer methods and programs in biomedicine*, vol. 194, p. 105532, 2020, doi: 10.1016/j.cmpb.2020.105532.

[148] A. Mobiny, P. A. Cicalese, S. Zare, P. Yuan, M. Abavisani, C. C. Wu, J. Ahuja, P. M. de Groot, and H. Van Nguyen, "Radiologist-level covid-19 detection using ct scans with detail-oriented capsule networks," *arXiv preprint arXiv:2004.07407*, 2020, doi: 10.48550/arXiv.2004.07407.

[149] E. E.-D. Hemdan, M. A. Shouman, and M. E. Karar, "Covidx-net: A framework of deep learning classifiers to diagnose covid-19 in x-ray images," *arXiv preprint arXiv:2003.11055*, 2020, doi: 10.48550/arXiv.2003.11055.

[150] M. Heidari, S. Mirniaharikandehei, A. Z. Khuzani, G. Danala, Y. Qiu, and B. Zheng, "Improving the performance of cnn to predict the likelihood of covid-19 using chest x-ray images with preprocessing algorithms," *International journal of medical informatics*, vol. 144, p. 104284, 2020, doi: 10.1016/j.ijmedinf.2020.104284.

[151] F. Bao, Y. Deng, Y. Kong, Z. Ren, J. Suo, and Q. Dai, "Learning deep landmarks for imbalanced classification," *IEEE transactions on neural net-*

works and learning systems, vol. 31, no. 8, pp. 2691–2704, 2019, doi: 10.1109/TNNLS.2019.2927647.

[152] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826, doi: 10.1109/CVPR.2016.308.

[153] O. Yuuki, J. Wang, T. Fukuchi, and A. B. Abdallah, “Parallelization and hardware mapping of deep neural network on reconfigurable platform for ai-enabled biomedical system,” in *SHS Web of Conferences*, vol. 139. Aizuwakamatsu, Japan, January 25-28: EDP Sciences, 2022, p. 03005, doi: 10.1051/shsconf/202213903005.

[154] I. Ahmed, G. Jeon, and F. Piccialli, “A deep-learning-based smart healthcare system for patient’ s discomfort detection at the edge of internet of things,” *IEEE Internet of Things Journal*, vol. 8, no. 13, pp. 10 318–10 326, 2021, doi: 10.1109/JIOT.2021.3052067.

[155] M. Almutairi, L. A. Gabralla, S. Abubakar, and H. Chiroma, “Detecting elderly behaviors based on deep learning for healthcare: recent advances, methods, real-world applications and challenges,” *IEEE Access*, vol. 10, pp. 69 802–69 821, 2022, doi: 10.1109/ACCESS.2022.3186701.

[156] X. Zhou, W. Liang, I. Kevin, K. Wang, H. Wang, L. T. Yang, and Q. Jin, “Deep-learning-enhanced human activity recognition for internet of healthcare things,” *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6429–6438, 2020, doi: 10.1109/JIOT.2020.2985082.

[157] E. Strubell, A. Ganesh, and A. McCallum, “Energy and policy considerations for deep learning in nlp,” *arXiv preprint arXiv:1906.02243*, 2019, doi: 10.48550/arXiv.1906.02243.

[158] M. Sorbaro, Q. Liu, M. Bortone, and S. Sheik, “Optimizing the energy consumption of spiking neural networks for neuromorphic applications,” *Frontiers in neuroscience*, vol. 14, p. 662, 2020, doi: 10.3389/fnins.2020.00662.

[159] P. Wijesinghe, A. Ankit, A. Sengupta, and K. Roy, “An all-memristor deep spiking neural computing system: A step toward realizing the low-power stochastic brain,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 5, pp. 345–358, 2018, doi: 10.1109/TETCI.2018.2829924.

[160] Q. Xia and J. J. Yang, “Memristive crossbar arrays for brain-inspired computing,” *Nature materials*, vol. 18, no. 4, pp. 309–323, 2019, doi: 10.1038/s41563-019-0291-x.

[161] A. Goel, C. Tung, Y.-H. Lu, and G. K. Thiruvathukal, “A survey of methods for low-power deep learning and computer vision,” in *2020 IEEE 6th World Forum on Internet of Things (WF-IoT).* IEEE, 2020, pp. 1–6, doi: 10.1109/WF-IoT48130.2020.9221198.

[162] R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni, “Green ai,” *Communications of the ACM*, vol. 63, no. 12, pp. 54–63, 2020, doi: 10.1145/3381831.

[163] S. Alyamkin, M. Ardi, A. C. Berg, A. Brighton, B. Chen, Y. Chen, H.-P. Cheng, Z. Fan, C. Feng, B. Fu *et al.*, “Low-power computer vision: Status, challenges, and opportunities,” *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 2, pp. 411–421, 2019, doi: 10.1109/JETCAS.2019.2911899.

[164] T. Fukuchi, M. I. Ogbodo, J. Wang, K. N. Dang, and A. Ben Abdallah, “Efficient pneumonia detection method and implementation in chest x-ray images based on a neuromorphic spiking neural network,” in *Conference on Computational Collective Intelligence Technologies and Applications*, ser.

Lecture notes in computer science. Hammamet, Tunisia, September 28-30: Springer, 2022, pp. 311–321, doi: 10.1007/978-3-031-16014-1_25.

[165] M. Ogbodo, T. Vu, K. Dang, and A. Abdallah, “Light-weight spiking neuron processing core for large-scale 3d-noc based spiking neural network processing systems,” in *2020 IEEE international conference on big data and smart computing (BigComp)*. IEEE, 2020, pp. 133–139, doi: 10.1109/BigComp48618.2020.00-86.

[166] A. Ben Abdallah and K. N. Dang, “Toward robust cognitive 3d brain-inspired cross-paradigm system,” *Frontiers in Neuroscience*, vol. 15, p. 690208, 2021, doi: 10.3389/fnins.2021.690208.

[167] V. H. The, “Algorithms and architectures for spiking neuromorphic systems,” Ph.D. dissertation, University of Aizu, 2019, doi: 10.15016/00000168.

[168] T. H. Vu, Y. Okuyama, and A. B. Abdallah, “Comprehensive analytic performance assessment and k-means based multicast routing algorithm and architecture for 3d-noc of spiking neurons,” *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 15, no. 4, pp. 1–28, 2019, doi: abs/10.1145/3340963.

[169] T. H. Vu, O. M. Ikechukwu, and A. B. Abdallah, “Fault-tolerant spike routing algorithm and architecture for three dimensional noc-based neuromorphic systems,” *IEEE Access*, vol. 7, pp. 90 436–90 452, 2019, doi: 10.1109/ACCESS.2019.2925085.

[170] J. K. Eshraghian, M. Ward, E. Neftci, X. Wang, G. Lenz, G. Dwivedi, M. Bennamoun, D. S. Jeong, and W. D. Lu, “Training spiking neural networks using lessons from deep learning,” *arXiv preprint arXiv:2109.12894*, 2021, doi: 10.48550/arXiv.2109.12894.

[171] N. Inc., “Nangate open cell library 45 nm,” (Accessed 23.02.2021). [Online]. Available: http://www.nangate.com/

[172] M. R. Guthaus, J. E. Stine, S. Ataei, B. Chen, B. Wu, and M. Sarwar, "Openram: An open-source memory compiler," in *2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD).* IEEE, 2016, pp. 1–6, doi: 10.1145/2966986.2980098.

[173] N. E. D. Automation, "Freepdk3d45 3d-ic process design kit," (Accessed 23.02.2021). [Online]. Available: http://www.eda.ncsu.edu/wiki/FreePDK3D45:Contents

[174] M. S. Kamal, L. Chowdhury, N. Dey, S. J. Fong, and K. Santosh, "Explainable ai to analyze outcomes of spike neural network in covid-19 chest x-rays," in *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC).* IEEE, 2021, pp. 3408–3415, doi: 10.1109/SMC52423.2021.9658745.

[175] W. Guo, M. E. Fouda, A. M. Eltawil, and K. N. Salama, "Neural coding in spiking neural networks: A comparative study for robust neuromorphic systems," *Frontiers in Neuroscience*, vol. 15, p. 638474, 2021, doi: 10.3389/fnins.2021.638474.