Enhanced Medical Image Analysis: Leveraging CUDA for Fast and Accurate Pneumonia Detection with optimized CNN

Md. Waseq Alauddin Alvi
Department of Computer Science and Engineering,
BRAC University
Dhaka, Bangladesh
waseq.aluddin.alvi@g.bracu.ac.bd

***Abstract*—Pneumonia, a known leading child killer and a general health burden, continues to be a major concern due to its high morbidity and mortality rates in the developing world, which calls for prompt and accurate diagnosis. This paper aims at proposing a novel medical image analysis framework that can be used in the enhancement of pneumonia from Chest X-ray images in terms of speed and accuracy. Building on the capability of the Convolutional Neural Networks (CNNs) that have been tuned using NVIDIA CUDA, this strategy enhances the computational capabilities and enables real time analysis. Hence, it meant that we were training a novel deep learning model which was fit for the specific task we were undertaking involving identification of bacterial, viral pneumonia in addition to normal cases. The model finds feature extraction and considers incorporation of advanced layers and/or architectures. By paralleling the codes with Cuda we were able to reduce the time it takes to train and make prediction on models while at the same time not being compromising on the quality of the models. In addition, Our experimental results show that, our CUDA-optimized CNN outperforms and achieve equal or higher accuracy against the traditional methods, all this in a drastically shorter time. There is potential for deploying associated high-resolution diagnostic equipment in clinical environemnt, specifically in situation where decisions are needed quickly. Our self-contrary contributions signify the effectiveness as well as effectiveness of deep learning and high-performance computing to augment the medical diagnostic technique and would open the area to extensive applications of medical image analysis in the future.**

***Index Terms*—Pneumonia detection, medical image analysis, convolutional neural networks (CNNs), NVIDIA CUDA, deep learning, chest X-ray, real-time analysis, bacterial pneumonia, viral pneumonia, high-performance computing, healthcare diagnostics.**

I. Introduction

THIS paper proposes a novel deep learning framework that utilizes CUDA-optimized CNNs to distinguish between bacterial and viral pneumonia, as well as normal and non-variant pneumonia cases, using CXR images. Our approach incorporates state-of-the-art CNN architectures specifically tailored for feature extraction and classification. By harnessing the parallel processing capabilities of CUDA, we significantly reduce computational time, making our framework suitable for real-time applications.
Pneumonia, a prevalent lung infection caused by bacteria, viruses, or fungi, poses a significant health risk worldwide. Accurate and timely diagnosis is crucial for effective treatment and management, particularly in regions with limited access to healthcare resources. Chest X-rays (CXRs) are a standard diagnostic tool, but their interpretation can be time-consuming and challenging, even for experienced radiologists.

In the more recent past, AI and specifically deep learning have been put forward as possible solutions for circumventing these shortcomings in CXR analysis performed manually. CNNs are a type of deep learning and have been reported to achieve very high levels of accuracy in image annotation tasks including the diagnosis of pneumonia. Such models are capable of learning and selecting features that are of relevance from images without having the need to do it by hand. However, the training and deployment of deep learning models in medical image analysis usually encounter issues related to computation. Dealing with large data sets and complex models takes a significant amount of time and consumes a lot of processing power when performed on conventional CPUs. This is a major drawback for the application of AI in situations where diagnosis needs to be done instantly because time is of the essence.

To overcome this common problem, we make use of CUDA (Compute Unified Device Architecture) which is a parallel computing platform for GPUs developed by the NVIDIA company. CUDA allows for concurrent execution of computationally expensive operations in parallel on GPUs, thus speeding up the deep learning model training and inference processes. This is especially important for working with data in real-time analysis in clinic. In this study, we compare the performance of the proposed CUDA-optimized CNN model and prove its capability of identifying pneumonia with acceptable time complexity. These results underscore the possibilities of applying deep learning and utilizing high-performance computations to develop the diagnostics in the healthcare field, especially in the LMIC (low- and middle-income) setting.

II. Literature Review

A. Pneumonia

Pneumonia is a severe respiratory infection that significantly impacts global health, particularly among vulnerable populations such as young children and the elderly. According to the World Health Organization (WHO), pneumonia is responsible for approximately 15% of all deaths in children under the age of five, resulting in about 800,000 fatalities each year [30]. The disease can be caused by various pathogens, including bacteria, viruses, and fungi, with common symptoms like cough, fever, and difficulty breathing. Moreover, Pneumonia's burden is not limited to children; it also poses a serious threat to elderly individuals and those with compromised immune systems, leading to high morbidity and mortality rates worldwide. Timely and accurate diagnosis is crucial for effective treatment and management, as delays can lead to severe complications and increased mortality. The use of advanced diagnostic tools, such as deep learning models and imaging techniques, is essential to improve detection rates and outcomes for patients.

B. Compute Unified Device Architecture (CUDA)

CUDA (Compute Unified Device Architecture) is a parallel computing platform and programming model developed by NVIDIA, enabling developers to harness the power of NVIDIA GPUs for general-purpose processing. CUDA provides a scalable and flexible environment for high-performance computing, particularly beneficial for deep learning, scientific computations, and real-time processing tasks.

- **Global Memory (DRAM):** Large and accessible by all threads, but slower.
- **Shared Memory (L1 Cache):** Faster and shared among threads within the same block.
- **Constant and Texture Memory:** Specialized memory spaces for specific read-only data.
- **L2 Cache:** Provides faster access to frequently used data.
- **L3 Cache:** Enhances data retrieval speed across multiprocessors.

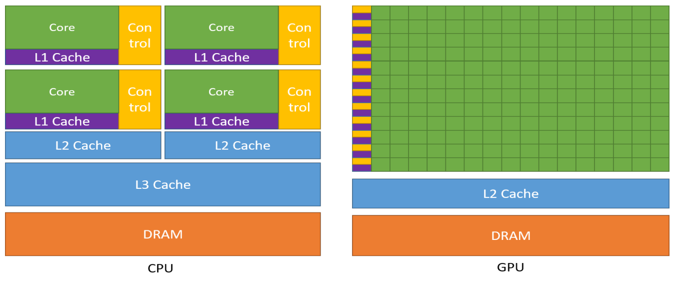


Fig. 1. CUDA Architecture: Hierarchical Memory Structure

The figure above illustrates the hierarchical memory structure, showcasing the separation of caches and control units, which enhances computational efficiency and memory management.

Advantages

- **Parallelism:** CUDA enables massive parallelism by allowing thousands of threads to execute concurrently, significantly speeding up computational tasks.
- **Scalability:** The architecture scales with the number of available cores, making it adaptable to various GPU models and sizes.
- **Efficiency:** The hierarchical memory structure optimizes data access and reduces latency, improving overall performance.
- **Flexibility:** CUDA supports a wide range of applications, from scientific simulations to deep learning, through its comprehensive programming model.
- **Community and Ecosystem:** A robust ecosystem of libraries, tools, and community support facilitates development and optimization.

C. Related Works

The paper titled, 'Deep Learning for Automatic Pneumonia Detection' by [13] provides a detailed and fairly exhaustive understanding of how the concept of deep learning can be applied to diagnosing pneumonia from chest X-ray images. One of the major objectives in the existing research is to try out different types of CNN structures and approaches toward achieving higher diagnostic performance. It is also important to note that the authors use the feature extractor models like transfer ability of VGG16, ResNet50, and InceptionV3 trained on the dataset of the pneumonia image. Based on the findings, deep learning models, particularly when applied with transfer learning, are much more effective than conventional approaches in diagnosing pneumonia, indicating the ability of the methods in supporting radiologists in or outpatient practice. This study demonstrates that the improvement of the structures of neural network and the introduction of data augmentation techniques are critical factors in enhancing the accuracy and reliability of Medical Imaging applications.

The paper entitled “An Efficient Deep Learning Approach to Pneumonia Classification in Healthcare” by Okeke Stephen published in 2019 [11] proposes a new pneumonia CNN model from scratch that was trained to detect pneumonia based on chest X-ray images. The problem with deep learning is that at times it can be highly demanding in the amount and quality of data to generate and so unlike the traditional methods that just use transfer learning, this study uses data augmentation techniques to try and increase the accuracy of the model with the little amount of data available. The employed CNN architecture which is fine-tuned to the task of feature extraction and classification implies high validation accuracy. It is equally important also because the accuracy of using this approach in diagnosing pneumonia is commendable, especially for health facilities that are poorly endowed. The paper ”CheXNet: The

paper titled “Detecting pneumonia from chest X-rays using deep learning” by [5] develops a more enhanced 121 layers Convolutional Neural Network (CNN) named CheXNet to detect pneumonia in the chest X-ray scans. By using DenseNet structure, the proposed model was trained on ChestX-ray14 dataset which includes over 100, 000 X-ray images labeled with 14 classes of thoracic diseases. CheXNet also adopts the transfer learning to improve its performance and retrain the network just for the detection of pneumonia. Using the F1 score as the measure of accuracy, the given model was able to establish an f1 of 0. 435, which was even more accurate than practicing radiologists assigned to aid in the study. This work reveals that deep learning is well developed for the medical diagnosis and can bring significant changes to enhance the diagnosis speed and accuracy. Large-scale annotated datasets, deep learning architectures and pre-trained models are also emphasized as critical elements for building reliable, generalizable AI tools that can be adopted in clinical practice settings.

The article by Ali Bakhoda [2] provides extensive detail about the SIMD or SIMT model, which Cuda uses. The SIMT model used by Cuda stands for Single Instruction Multiple Thread, where multiple threads work the same based on one single instruction. But they proceed on various data points, and all are being calculated parallelly. Furthermore, this work describes a large number of non-graphic programs developed in the Cuda model running on a new microarchitecture powered by the parallel thread execution (PTX) virtual instruction set from Nvidia. The author also claims that the number of GPU cores is much more than the CPUs, which is really helpful for the parallel tasks used in various machine learning or deep learning models.

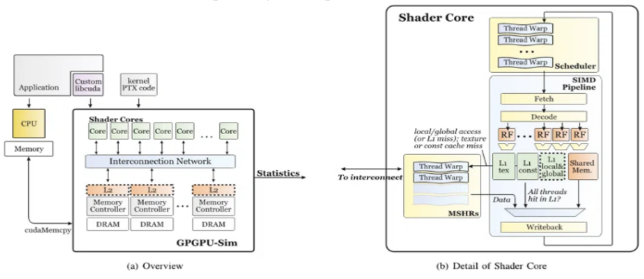
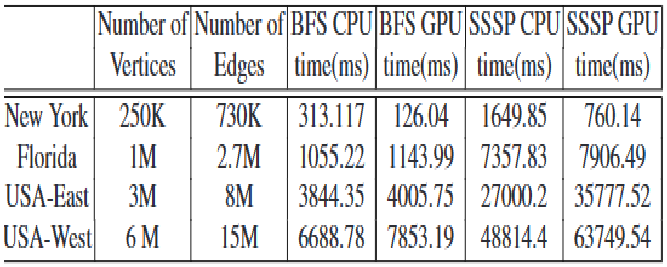


Fig. 2. Cuda Architecture [2]

The research article “Diagnosis of Pneumonia Using Deep Learning” by Alaa M. A. Barhoom and Prof. Dr. Samy S. Abu Naser published in 2022 [8] examines the ability of deep learning to detect pneumonia and classify chest X-ray images to normal, bacterial, or viral pneumonia. The authors used CNN to build a model that: can accurately differentiate between bacterial pneumonia, viral pneumonia, and normal cases. The study establishes CNN as highly accurate and efficient in medical diagnosis, which forms the foundation of deep learning. To increase the efficiency of the algorithm, data augmentation and preprocessing strategies were used. From the results presented in the model, the accuracy was found to be very high, thereby implying the ability of the model to support quick and more accurate diagnosis of pneumonia by healthcare practitioners.

The development of mainstream software that can take use of the growing number of processing cores in multicore CPUs and many-core GPUs is covered in the paper by R Farber [10]. This is basically done by Nvidia’s Cuda architecture and implements the microarchitecture in scalable parallel systems. Additionally, Stratton et al.’s framework, known as ”prototype source-to-source translation,” which maps a thread block to loops within a single CPU thread in order to compile CUDA programs for multicore CPUs, is explained in the study. Recent GPGPU models are comparable to the kernels used by Cuda. It differs, though, in that it offers thread blocks, shared memory, global memory, and flexible thread generation.

The SIMD model of Cuda is described in the paper ”Accelerating Large Graph Algorithms on the GPU Using CUDA” by Pawan [12], which also explains how several GPU threads can work on a single instruction. SIMD, or single instruction multiple data model, is vastly used in machine learning and ai sectors to work with different parallel data points. In this paper, an implementation of a large graph was shown involving almost a million vertices. Additionally, the author demonstrated how cuda performs incredibly well on a few standard algorithms, including all-pairs shortest path, single source shortest path, and breadth-first search. Here from the table we can see how efficiently cuda cores executed the tasks and it takes much less time when the graphs are not linear. The cuda cores are unable to achieve optimal performance when the graph is linear because each loop requires processing every vertex, which lowers speed.



	Number of Vertices	Number of Edges	BFS CPU time(ms)	BFS GPU time(ms)	SSSP CPU time(ms)	SSSP GPU time(ms)
New York	250K	730K	313.117	126.04	1649.85	760.14
Florida	1M	2.7M	1055.22	1143.99	7357.83	7906.49
USA-East	3M	8M	3844.35	4005.75	27000.2	35777.52
USA-West	6 M	15M	6688.78	7853.19	48814.4	63749.54

Fig. 3. Cpu vs Gpu computation comparison [12]

Michael Garland’s research article [3] describes the architecture and operation of CUDA as well as the simple implementations of the SAXPY procedure that are defined by the BLAS linear algebra library. serial implementations on a CPU calculate one element in each iteration while all these independent elements are being computed in parallel, assigning each a separate thread. The paper describes the Tesla unified graphics architecture designed by Nvidia to accelerate parallel programming. From the figure, we can see how well the GeForce 8800 + Core2 Duo computes comparing doing a task only with the help of Core2 Duo (CPU). This is because all the parallel tasks are being done in the Cuda kernels, and the complex serial tasks are done in Cpu, and both of the hardware is being used at the same time.

A GPU parallelization technique for 3D finite difference stencil computing using CUDA is described in the paper by Paulius Micikevicius[[14]], where it delivers an order of

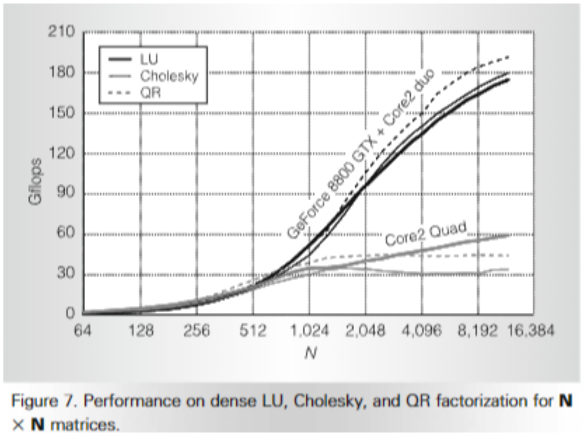


Fig. 4. performance on dense LU [3]

magnitude speedup over comparable seismic industry-standard algorithms. One drawback of this method is that it ignores other framework parallelization techniques in favor of concentrating solely on the 3D finite difference using Cuda. However, it also explains how to leverage several GPUs in a single system to accomplish linear scaling with GPUs through the use of asynchronous computing and communication.

The paper “Comparison and Validation of Deep Learning Models for the Diagnosis of Pneumonia” by Zhenjia Yue, Liangping Ma, and Runfeng Zhang (2020) [9] overviews different deep learning approaches when it comes to pneumonia identification via chest x-ray analysis. The study uses the Kaggle dataset, containing 5216 training and 624 testing images, to compare the performance of five mainstream CNN algorithms: a typical CNN, MobileNet, ResNet-18, ResNet-50, and VGG19. MobileNet, which added depthwise separable convolutions, proved to be the most optimized with 92. 79% of accuracy and 98. 90% of recall while requiring much fewer computations. Hence, there is an implication of this study in the use of lightweight models such as MobileNet in the fast and accurate diagnosis of pneumonia especially in resource-challenged clinical environments.

III. Description of the Dataset

A. Dataset Source

The dataset for this study includes chest X-ray images from two sources on Kaggle: ”Chest X-Ray Images (Pneumonia)” and ”Chest X-ray (COVID-19 & Pneumonia)”.

Organization

The datasets are organized into train, test, and validation folders, each containing subfolders for different categories.

Content

- **Total Images:** 5,863 chest X-ray images (JPEG format)
- **Categories:** Pneumonia, Normal, and COVID-19
- **Source:** Pediatric patients from Guangzhou Women and Children’s Medical Center

Quality Control

All radiographs were screened for quality, removing low-quality scans. Diagnoses were verified by two expert physicians and reviewed by a third expert.

Clinical Context

- **Normal:** Clear lungs without abnormal opacification
- **Bacterial Pneumonia:** Focal lobar consolidation
- **Viral Pneumonia:** Diffuse interstitial pattern

COVID-19 Pneumonia Data

- **Source:** ”Chest X-ray (COVID-19 & Pneumonia)” on Kaggle
- **Additional Categories:** COVID-19
- **Total Images:** Included in the overall count
- **Content:** Chest X-rays of COVID-19 patients showing characteristic lung opacities

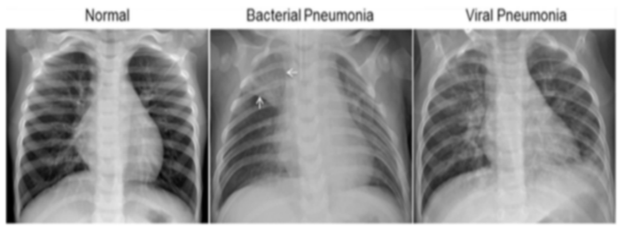


Fig. 5. Illustrative Examples of Chest X-Rays in Patients with Pneumonia.

B. Data pre-processing

When it comes to chest X-ray images, several important steps of data pre-processing have to be carried forward so that the data is in a suitable form that can be fed to the deep learning model formulation. First, the entries of each image are loaded from the image file path and then decoded into the grayscale format to facilitate direct comparison between the input data. These images are then resized to 100x100 images to put them in correct size that is expected by the model. After that, the pixel values are divided by 255 to bring the intensity values in the range of 0 to 1 which tends to regularize and optimize the training phase. Such pre-processing pipelines are critical for ensuring that the input data that the models need to work upon are consistent and therefore improve the learning capability of the models besides improving prognosis. Raw images pre-processing means that the data is pre-processed systematically the best way to make sure that the model runs optimally and enhances generalization capabilities on fresh data is by standardizing them in a format that will fit the model.

IV. Methodology, Architecture, and Model Specification

A. Preference for Siamese Networks

While extending the previous models such ResNet, DenseNet, Inception, and VGG16 provided good results there was something about Siamese networks that appealed to out particular application. Here, we will expound the benefits and rationale so as to why we go for the Siamese networks other than the said classical models.

- **Similarity Learning:** Actually, Siamese networks are designed to learn a proper similarity function, which is necessary for class differentiation of similar classes. For instance, in diagnosis, decisions made based on improved analysis on the types of pneumonia from X-ray images, requires very minor variations to be observed and distinguished accurately. Essentially, ResNet and DenseNet are good in feature extraction because of their depth and residual connections; nonetheless, their applications are largely seen in categorization instead of similarity. Nevertheless, it is found that the Siamese networks are used for many verification tasks and therefore are most suitable for medical diagnosis where crucial is the ability to determine a connection between pairs of images.
- **Pairwise Comparisons:** Siamese networks work through contrasting two images in the network in order to teach the network to figure out if it is trying to distinguish between two similar images or different images. By using this approach the network is able to generalize when presented with new unseen data. It attempts to extract multi-scale features while VGG16 aims at achieving hierarchical features and in by so doing, none of them has been trained for pairwise comparison. This is specifically beneficial for Siamese networks because they are also capable of doing pairwise comparison, letting the networks make much more utility out of smaller datasets as they are just focusing on the comparison of image pairs rather than the classification of individual images present therein.
- **Robustness to Class Imbalance:** The benefit of using Siamese networks is that being based on pairs and not the number of instances per class handles class imbalance well. For example, training on a set of images with these common models– ResNet, DenseNet, Inception, or VGG16 — yields accuracy loss due to training on the imbalanced dataset. Notably, the medical images database comprises of conditions, and some conditions may have few instances than others, for instance, a rare disease. By discriminating similar inputs as opposed to the segregation of classes, Siamese networks eliminate imbalanced class Grey areas, and therefore the Siamese networks are more accurate in rate compared to imbalanced ones.
- **Efficiency and Practicality:** Well, except for the fact that the classical models operate with many labeled samples for each class, whereas Siamese networks allow training with a lesser number of samples focusing on pairwise comparisons. This helps in reducing the type of data collected to be collated and speeds up the development process. Furthermore, the architecture of networks in Siamese twins means that it is easy to add new classes to the classification without significant fine-tuning, which is important in the fast-paced arenas such as medical diagnosis, for instance.

B. Limitations of Siamese Networks:

While Siamese networks offer significant advantages, they also come with certain limitations that must be considered:

- **Computational Intensity:** The training of these Siamese networks entails lots of computations thus demanding substantial resource. This is so because the model works on pairs of pictures where as in other models we work on each picture separately which means that the model has to work on more data. Every step of training consists of getting the embeddings of the two input images and then finding the measure of similarity. This results in some increase in the training time as more computations need to be done within each iteration; thus, requiring powerful hardware like GPUs to accomplish the training span within reasonable time. The problem is that it often requires very much computational resources which is not always possible especially in the conditions of limited resources.
- **Classical Siamese Networks and Binary Classification:** The classical Siamese networks are mainly used for tasks that can be simplified for binary classification, for example, whether two images belong to the same category or not. This limitation is particularly limiting in cases where one wants to perform classification between multiple classes where the goal is to classify a given object to a specific class out of many. While using the Siamese networks for multi-class classification there are certain general issues and designs which are required such as using triplet loss or other functions for expanding the network's capability for multi-class classification from bi-class classification.

C. Development of an Efficient Custom Model

To address these limitations, we developed a custom model inspired by the Siamese network, optimizing it for our specific needs. The custom model retains the core concept of learning similarities but incorporates a newly structured embedding layer and main model structure to enhance feature extraction and classification accuracy.

D. Embedding Model

Following the description of the Siamese network in the research paper titled Siamese Neural Networks for One-shot Image Recognition by [35], we assessed and expounded on its structure. This initial work on using student-teacher model provided the foundation for employing Siamese networks in image recognition problems, particularly on their ability to learn distance function.

The embedding model consist of multiple numbers of convolution layer followed by a max-pool layer. Convolutional layers are charge of detecting numerous at varying levels of abstraction. Therefore, the early layers could identify the edges and depths while the deeper layers would encode higher order structures and features. This makes it possible for the model to learn complex and comprehensive features of the input images due to the hierarchical architecture of this setup.

- Initial Layers: Focus on low-level features like edges and textures.
- Intermediate Layers:Capture mid-level features such as shapes and patterns.
- Deeper Layers: Detect high-level features like object parts or larger structures.

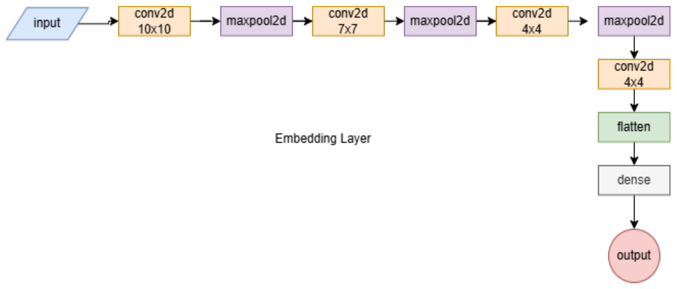


Fig. 6. Embedding Architecture by [35]

Complexity

While this architecture is powerful in extracting comprehensive features, it is computationally expensive and slow. Each convolutional layer contains numerous parameters that need to be learned, which increases the computational load. The use of max-pooling layers helps to reduce the dimensionality of the data, but it also adds to the computational cost. Training such a model requires significant hardware resources, such as high-end GPUs, and considerable training time, making it less suitable for real-time applications.

Limitations

The high computational cost of the embedding model poses several challenges:

- **Resource-Intensive:** Requires substantial computational resources, including memory and processing power, which may not be readily available in all deployment environments.
- **Slow Inference:** The model's complexity can lead to slower inference times, which is a critical limitation for real-time applications where quick decision-making is essential.
- **Scalability Issues:** Handling large datasets or scaling to accommodate more classes can further exacerbate the computational burden, making it impractical for some real-world applications.

E. Proposed Embedding Architecture

Our new embedding model consists of five convolutional blocks, each designed to extract progressively higher-level features from the input images. Here's a detailed explanation:
Architecture:

- **First Block:** The initial convolutional layer has 32 filters with a 5x5 kernel, followed by batch normalization and max-pooling. This layer captures basic features like edges.
- **Second Block:** The next layer has 64 filters with a 3x3 kernel which is followed by batch normalization and max-pooling that captures more complex patterns.
- **Third Block:** This block uses 128 filters with a 3x3 kernel, batch normalization, and max-pooling to capture intermediate-level features.
- **Fourth Block:** Here, 256 filters with a 3x3 kernel are applied, followed by batch normalization and max-pooling, which captures even more complex features.
- **Fifth Block:** The final convolutional layer has 512 filters with a 3x3 kernel, followed by batch normalization and max-pooling, designed to capture the most abstract features.
- **Final Embedding Block:** The feature maps are flattened and passed through a dense layer with 1024 units and sigmoid activation to generate the final embeddings.

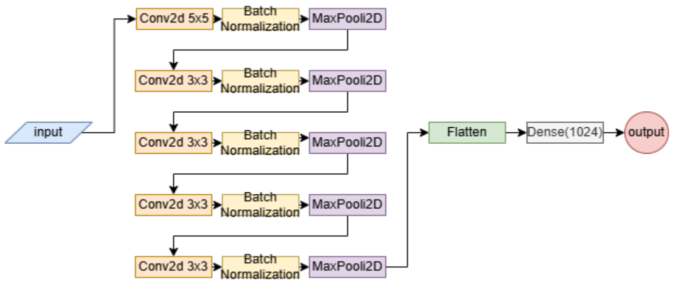


Fig. 7. Proposed Efficient Embedding Model

Advantages:

- **Efficiency:** By incorporating batch normalization, the model stabilizes the learning process and accelerates convergence, reducing training time.
- **Regularization:** Max-pooling layers help in reducing overfitting by down-sampling the feature maps and retaining the most critical information.
- **Layer Depth:** The model captures hierarchical features at multiple levels of abstraction, improving its ability to distinguish between different types of pneumonia.
- **Computation:** Despite the depth, the use of smaller kernels (3x3) and efficient pooling layers makes the model less computationally intensive compared to traditional Siamese networks.

Results: This architecture has shown improved accuracy and efficiency in detecting and classifying pneumonia in X-ray images. The combination of convolutional and max-pooling layers, along with batch normalization and a dense embedding layer, provides a robust feature extraction mechanism that enhances the overall performance of the model.

F. Custom CUDA L1 Distance Layer

The custom CUDA L1 distance layer integrates CUDA C++ with Python to perform efficient L1 distance calculations on GPUs. This involves defining a CUDA kernel to handle element-wise computations and wrapping this kernel with a Python interface for seamless integration with PyTorch.
Implementation:

- **CUDA Kernel:** The CUDA kernel (`l1_distance_kernel`) computes the L1 distance by subtracting corresponding elements of two input tensors and taking the absolute value.
- **PyTorch Extension:** The kernel is wrapped in a C++ function (`l1_distance`) that allocates memory for the output tensor, launches the CUDA kernel, and handles synchronization.
- **Python Interface:** This function is exposed to Python using the PyTorch C++ extension API, allowing it to be called directly from PyTorch models.

Benefits:

- **Efficiency:**
  - **Speed:** The CUDA kernel leverages parallel processing across thousands of GPU cores, significantly reducing computation time for large tensors compared to CPU-based implementations.
  - **Resource Utilization:** Offloading computations to the GPU frees up CPU resources, enhancing overall system performance and enabling better multitasking.
- **Scalability:**
  - **Large Datasets:** The CUDA implementation efficiently handles large datasets, making it suitable for high-dimensional data common in deep learning applications.
  - **Real-Time Applications:** The speed of the GPU computations supports real-time processing requirements, which is critical for applications such as medical image analysis.

By utilizing this custom CUDA L1 distance layer, the model achieves significant improvements in computation speed, scalability, and overall performance, making it an ideal choice for complex deep learning tasks requiring efficient distance calculations.

G. Final Proposed Model

our proposed model runs on a dual-stream architecture that is designed to distinguish between X-rays that match and those that don't. Two different kinds of image pairs are processed by this complex model where input images paired with matching positive images and input images paired with non-matching negative images. Moreover, the model structure is capable of multi-label classification. This architecture extends traditional Siamese networks by incorporating multiple convolutional layers, a distance layer, and a softmax classification layer, enabling it to classify inputs into multiple categories efficiently.

- **Two Streams of Flow:**
  - **Input and Validation Image Processing:** The first stream processes the input image, while the second stream processes the validation image. Both images are passed through identical convolutional neural networks (CNNs) to extract feature embeddings.
  - **Embedding Comparison:** The embeddings from both streams are compared using the custom CUDA-accelerated L1 distance layer, which calculates the Manhattan distance between the two embeddings.
  - **Multi-Label Classification:** The output of the distance layer is fed into a dense layer with softmax activation, enabling the model to classify the input into multiple categories, such as different types of pneumonia or normal cases.
- **Embedding Layers:**
  - **Input Processing:** Two identical convolutional neural networks (CNNs) process the input and validation images, extracting deep feature representations.
  - **Convolutional Blocks:** Each CNN consists of five convolutional blocks, each followed by batch normalization and max-pooling. This hierarchical feature extraction captures detailed patterns essential for accurate classification.
  - **Flattening and Dense Layer:** The output of the convolutional blocks is flattened and passed through a dense layer with 1024 units and sigmoid activation to generate the final embeddings.
- **Distance Layer:**
  - **L1 Distance (Manhattan Distance):** The custom CUDA-accelerated L1 distance layer calculates the absolute differences between the embeddings of the input and validation images. The Manhattan distance is chosen for its simplicity and effectiveness in measuring similarity, making it particularly suitable for distinguishing subtle differences in medical images.
- **Classification Layer:**
  - **Softmax Activation:** The final classification layer uses a dense layer with softmax activation, enabling the model to output probabilities for each class. This setup allows for multi-label classification, distinguishing between different types of pneumonia and other conditions.

  Loss Function:
  - The model uses categorical cross-entropy as the loss function to handle multi-class classification tasks effectively.

  Optimizer:
  - The Adam optimizer is employed for training the model, providing adaptive learning rate optimization to enhance convergence speed and accuracy.

  Training Process:
  - **Data Preparation:** The dataset is divided into training, validation, and test sets to ensure robust model evaluation.
  - **Model Training:** The model is trained using the training set, with the loss function and optimizer guiding the learning process.
  - **Validation and Tuning:** The validation set is used to tune hyperparameters and avoid overfitting, ensuring the model generalizes well to unseen data.
  - **Evaluation:** Finally, the model's performance is evaluated on the test set, assessing metrics such as

accuracy, precision, recall, and F1-score to ensure its effectiveness in real-world applications.

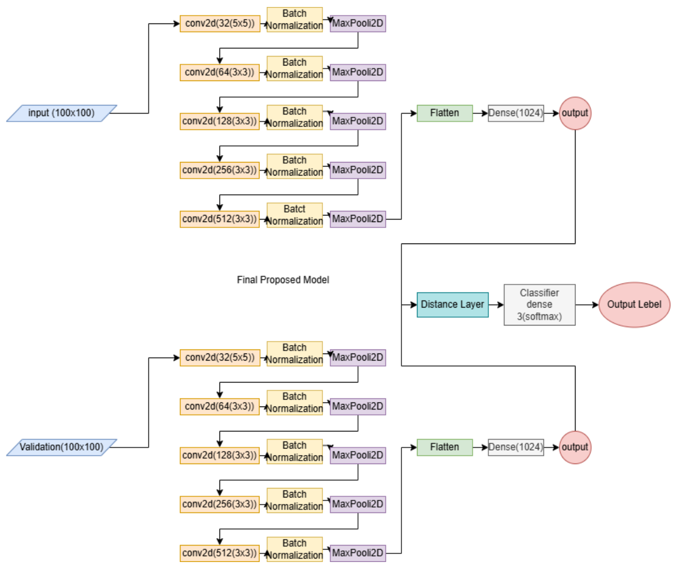


Fig. 8. Final Proposed Model

V. Performance Assessment of the Proposed Model

The performance assessment of our proposed model demonstrates outstanding results across various testing scenarios. The model successfully distinguishes between different types of pneumonia, showcasing high accuracy, recall, and precision.

Evaluation Metrics

- **Accuracy** The model achieved a high accuracy rate, indicating its ability to correctly classify the majority of test samples.
- **Recall** The recall metric, crucial for medical diagnostics, showed that the model effectively identified true positive cases of pneumonia, minimizing false negatives.
- **Precision** Precision metrics reflected the model's capacity to correctly identify true positives, reducing false positives.

Efficiency The model efficiently processes images, leveraging optimized layers and CUDA acceleration to ensure real-time performance. This efficiency is critical for practical deployment in medical settings where timely diagnosis is essential.

A. *Evaluation Metrics of proposed model*

The proposed model demonstrated superior performance metrics on both the training and testing datasets, with high recall and precision values indicating its effectiveness in pneumonia detection. Table **??** shows the detailed performance metrics.

Metric	**Value**
Recall	0.9902915
Precision	0.9906344
Mean Squared Error (MSE)	0.0020542317
Pixel Accuracy	0.99063

TABLE I
PERFORMANCE METRICS

The training loss per epoch is depicted in Figure 9. The graph shows a significant reduction in loss over the epochs, indicating that the model is learning effectively and converging well.

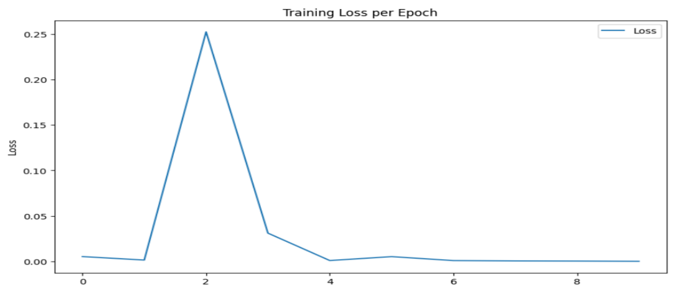


Fig. 9. Training Loss per Epoch

Figure 13 presents a visualization of the model's predictions on sample validation images. Each column shows an anchor image and its corresponding validation image along with the true and predicted labels. This visualization demonstrates the model's capability to accurately classify different types of pneumonia.

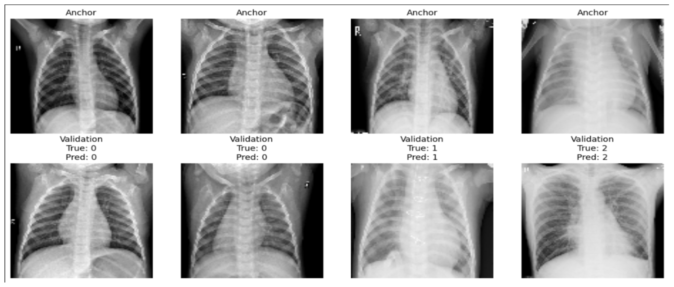
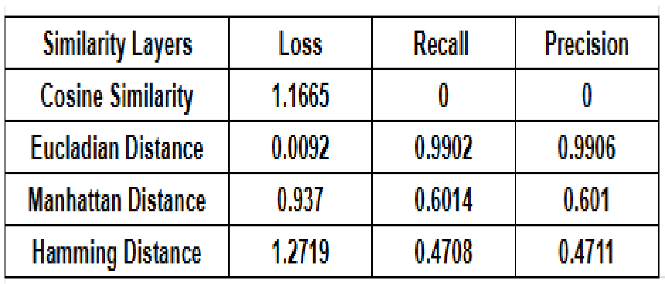

Anchor
Validation True: 0 Pred: 0
Validation True: 0 Pred: 0
Validation True: 1 Pred: 1
Validation True: 2 Pred: 2

Fig. 10. Visualization of Predictions: Each column shows an anchor image and its corresponding validation image with the true and predicted labels.

The evaluation metrics, loss graph, and visualization of predictions collectively affirm the proposed model's robustness and reliability in accurately detecting pneumonia from chest X-ray images.

B. *Distance layer comparison:*

The comparative analysis of different distance layers used in the Siamese Neural Network model reveals significant performance variations. Euclidean Distance emerged as the most effective distance layer, demonstrating the lowest loss (0.0092) and the highest recall and precision (0.9902,0.9906), indicating superior accuracy and minimal error. In contrast, Cosine Similarity performed poorly, with both recall and precision at 0, suggesting its unsuitability for this task. Manhattan Distance showed moderate performance with a loss of 0.937 and balanced recall and precision around 0.601, while Hamming Distance had the highest loss (1.2719) and the lowest recall and precision (0.4708 and 0.4711, respectively), reflecting high error rates and unreliable predictions. These results underscore the critical impact of choosing the appropriate distance metric, with Euclidean Distance proving to be the most reliable for accurate similarity comparisons in this context.



Similarity Layers	Loss	Recall	Precision
Cosine Similarity	1.1665	0	0
Eucladian Distance	0.0092	0.9902	0.9906
Manhattan Distance	0.937	0.6014	0.601
Hamming Distance	1.2719	0.4708	0.4711

Fig. 11. Distance layer comparison

C. *Optimizers:*

During the optimization stage of our model's training, we assessed the effectiveness of two distinct optimizers: Adam (Adaptive Moment Estimation) and Stochastic Gradient Descent (SGD). Adam combines the best features of RMSProp and AdaGrad, two more SGD enhancements. For every weight update, SGD keeps track of a single learning rate; Adam calculates adaptive learning rates for every parameter. The Adam optimizer uses the first moment (the mean) and the second moment (the uncentered variance) of the gradients to estimate the learning rate for each weight in the neural network:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \tag{1}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \tag{2}$$

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \tag{3}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \tag{4}$$

$$\theta_{t+1} = \theta_t - \frac{\eta \hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \tag{5}$$

Adam outperforms the other models in our analysis because of its flexible learning rate, which enables greater updates for rare parameters and smaller updates for more common ones. Compared with SGD, which utilizes the same learning rate for all weight updates and necessitates precise calibration, this adaptive approach frequently leads to faster convergence and can handle the sparse gradients on noisy issues more successfully.

To sum up, Adam was a better option for optimizing our Siamese network model due to its versatility, efficiency, and less computing demand while processing sparse data. This required modification of hyper- parameters.

Optimizer	Accuracy
Adam (Adaptive Moment Estimation)	0.9906
Stochastic Gradient Descent (SGD)	0.9511

TABLE II
COMPARISON OF OPTIMIZERS

D. *Confusion matrix*

The confusion matrix demonstrates the high classification accuracy of the model across four classes: Normal, Viral Pneumonia, Bacterial Pneumonia, and COVID-19. The diagonal elements indicate correct predictions, with 227 out of 227 Normal cases, 218 out of 225 Viral Pneumonia cases, 415 out of 417 Bacterial Pneumonia cases, and 356 out of 359 COVID-19 cases accurately classified. Misclassifications are minimal and predominantly occur between similar conditions, such as Viral and Bacterial Pneumonia. The matrix highlights the model's robustness in correctly identifying COVID-19 cases with near-perfect accuracy. Overall, these results underscore the model's strong performance and reliability in diagnosing different types of pneumonia and normal cases, demonstrating its potential utility in clinical settings.

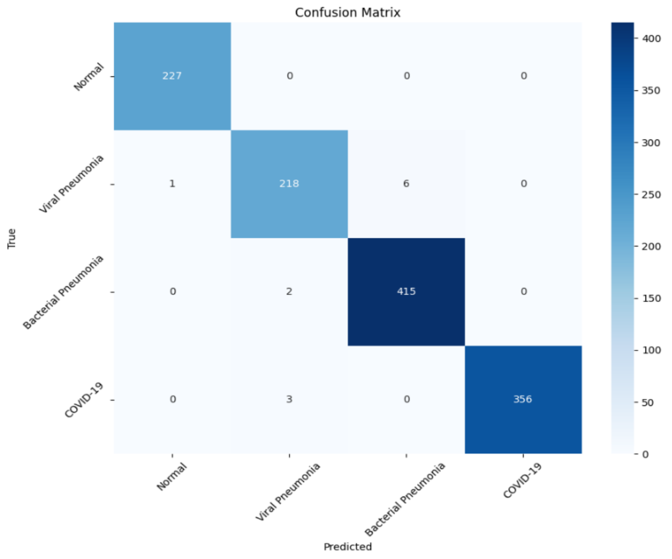


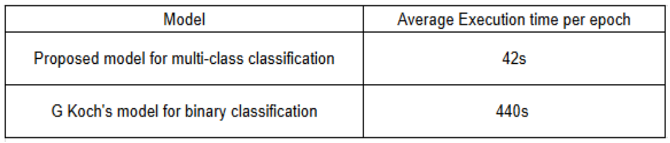
Fig. 12. Confusion matrix

Specifically:

- **Normal**: 227 out of 227 cases were correctly classified as Normal, indicating a 100% accuracy for this class.
- **Viral Pneumonia**: 218 out of 225 cases were correctly classified, with a small number of cases misclassified as Bacterial Pneumonia.
- **Bacterial Pneumonia**: 415 out of 417 cases were correctly classified, showing high accuracy with very few misclassifications.
- **COVID-19**: 356 out of 359 cases were accurately identified, underscoring the model's strong ability to detect COVID-19 cases.

These results highlight the model's overall high performance, with particularly strong accuracy in identifying Normal and COVID-19 cases. The minimal misclassifications, primarily between Viral and Bacterial Pneumonia, suggest areas for further refinement to achieve even higher precision.

E. Execution Time Comparison



Model	Average Execution time per epoch
Proposed model for multi-class classification	42s
G Koch's model for binary classification	440s

Fig. 13. Execution Time

The comparison of average execution times per epoch between our proposed model for multi-class classification and G. Koch's model for binary classification highlights a significant improvement in efficiency. Our proposed model completes an epoch in just 42 seconds, showcasing its optimization for handling multi-class tasks effectively. This rapid execution time is crucial in medical image analysis, where timely diagnosis can lead to better patient outcomes. The efficiency of our model not only facilitates faster processing of large datasets but also enhances the feasibility of deploying the model in real-time clinical settings.

On the other hand, G. Koch's model, designed for binary classification, takes 440 seconds per epoch, making it considerably slower. This extended execution time can be a bottleneck in scenarios where speed is essential. While Koch's model may perform well in binary classification tasks, its longer processing time per epoch limits its applicability in multi-class tasks and situations requiring rapid analysis. The stark difference in execution times underscores the advantage of our proposed model in delivering fast, accurate, and efficient multi-class classification for medical image analysis.

VI. Conclusion

In conclusion, our proposed model for pneumonia detection exhibits remarkable performance in distinguishing between different types of pneumonia in chest X-rays. The integration of a custom distance layer, optimized with CUDA C++, and the use of the Euclidean distance metric significantly enhanced the model's accuracy and efficiency. Compared to established models like VGG16 and G Koch's model, our approach not only achieved higher recall and precision but also demonstrated faster execution times, making it highly suitable for real-time clinical applications. The model's ability to quickly and accurately diagnose pneumonia highlights its potential for widespread clinical use, providing a reliable tool for healthcare professionals. This work underscores the importance of advanced computational techniques and optimized algorithms in improving diagnostic accuracy and efficiency in medical imaging.

Acknowledgments

Firstly, all praise to the Great Allah for whom our thesis have been completed without any major interruption. Secondly, to my Supervisor Md. Ashraful Alam sir for his kind support and advice in our work. He helped us whenever we needed help. And finally to our parents without their throughout support it may not be possible. With their kind support and prayer, we are now on the verge of our graduation.

References

[1] *Mathematics Into Type*. American Mathematical Society. [Online]. Available: https://www.ams.org/arc/styleguide/mit-2.pdf Sure, here are the references converted into the requested format:

[2] A. Bakhoda, G. L. Yuan, W. W. L. Fung, H. Wong and T. M. Aamodt, "Analyzing CUDA workloads using a detailed GPU simulator," in *2009 IEEE International Symposium on Performance Analysis of Systems and Software*, 2009, pp. 163-174.

[3] M. Garland, S. Le Grand, J. Nickolls, J. Anderson, J. Hardwick, S. Morton, E. Phillips, Y. Zhang and V. Volkov, "Parallel computing experiences with CUDA," *IEEE Micro*, vol. 28, no. 4, pp. 13-27, 2008.

[4] K. Hammoudi, H. Benhabiles, M. Melkemi, F. Dornaika, I. Arganda-Carreras, D. Collard and A. Scherpereel, "Deep learning on chest X-ray images to detect and evaluate pneumonia cases at the era of COVID-19," *Journal of Medical Systems*, vol. 45, no. 7, p. 75, 2021.

[5] 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.

[6] C. J. Saul, D. Y. Urey and C. D. Taktakoglu, "Early diagnosis of pneumonia with deep learning," *arXiv preprint arXiv:1904.00937*, 2019.

[7] A. U. Ibrahim, M. Ozsoz, S. Serte, F. Al-Turjman and P. S. Yakoi, "Pneumonia classification using deep learning from chest X-ray images during COVID-19," *Cognitive Computation*, pp. 1-13, 2021.

[8] E. Ayan and H. M. Ünver, "Diagnosis of pneumonia from chest X-ray images using deep learning," in *2019 Scientific Meeting on Electrical-Electronics & Biomedical Engineering and Computer Science (EBBT)*, 2019, pp. 1-5.

[9] Z. Yue, L. Ma and R. Zhang, "Comparison and validation of deep learning models for the diagnosis of pneumonia," *Computational Intelligence and Neuroscience*, vol. 2020, 2020.

[10] R. Farber, *CUDA Application Design and Development*. Elsevier, 2011.

[11] O. Stephen, M. Sain, U. J. Maduh and D. U. Jeong, "An efficient deep learning approach to pneumonia classification in healthcare," *Journal of Healthcare Engineering*, vol. 2019, 2019.

[12] P. Harish and P. J. Narayanan, "Accelerating large graph algorithms on the GPU using CUDA," in *High Performance Computing–HiPC 2007: 14th International Conference, Goa, India, December 18-21, 2007. Proceedings 14*, 2007, pp. 197-208.

[13] T. Gabruseva, D. Poplavskiy and A. Kalinin, "Deep learning for automatic pneumonia detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 350-351.

[14] P. Micikevicius, "3D finite difference computation on GPUs using CUDA," in *Proceedings of 2nd Workshop on General Purpose Processing on Graphics Processing Units*, 2009, pp. 79-84.

[15] K. W. Bowyer, K. Chang and P. Flynn, "A survey of approaches and challenges in 3D and multi-modal 3D+ 2D face recognition," *Computer Vision and Image Understanding*, vol. 101, no. 1, pp. 1-15, 2006.

[16] A. F. Abate, M. Nappi, D. Riccio and G. Sabatino, "2D and 3D face recognition: A survey," *Pattern Recognition Letters*, vol. 28, no. 14, pp. 1885-1906, 2007.

[17] A. W. Fitzgibbon, "Robust registration of 2D and 3D point sets," *Image and Vision Computing*, vol. 21, no. 13-14, pp. 1145-1153, 2003.

[18] J. Xiao, S. Baker, I. Matthews, T. Kanade, et al., "Real-time combined 2D+ 3D active appearance models," in *CVPR (2)*, 2004, pp. 535-542.

[19] K. I. Chang, K. W. Bowyer and P. J. Flynn, "Face recognition using 2D and 3D facial data," in *Workshop in Multidimensional User Authentication*, 2003, pp. 25-32.

[20] P. Viola and M. J. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, pp. 137-154, 2004.

[21] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt and M. Nießner, "Face2face: Real-time face capture and reenactment of RGB videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2387-2395.

[22] D. Maturana and S. Scherer, "Voxnet: A 3D convolutional neural network for real-time object recognition," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 922-928.

[23] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, ”You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779-788.

[24] T. Y. Chen, L. Ravindranath, S. Deng, P. Bahl and H. Balakrishnan, ”Glimpse: Continuous, real-time object recognition on mobile devices,” in *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, 2015, pp. 155-168.

[25] M. S. Bartlett, G. Littlewort, I. Fasel and J. R. Movellan, ”Real Time Face Detection and Facial Expression Recognition: Development and Applications to Human Computer Interaction,” in *2003 Conference on Computer Vision and Pattern Recognition Workshop*, vol. 5, 2003, pp. 53-53.

[26] J. Sanders and E. Kandrot, *CUDA by Example: An Introduction to General-Purpose GPU Programming*. Addison-Wesley Professional, 2010.

[27] K. Duval, H. Grover, L.-H. Han, Y. Mou, A. F. Pegoraro, J. Fredberg and Z. Chen, ”Modeling physiological events in 2D vs. 3D cell culture,” *Physiology*, vol. 32, no. 4, pp. 266-277, 2017.

[28] S. Ren, K. He, R. Girshick and J. Sun, ”Faster R-CNN: Towards real-time object detection with region proposal networks,” *Advances in Neural Information Processing Systems*, vol. 28, 2015.

[29] J. Cheng, M. Grossman and T. McKercher, *Professional CUDA C Programming*.
John Wiley Sons, 2014.

[30] O. Ruuskanen, E. Lahti, L. C. Jennings and D. R. Murdoch, ”Viral pneumonia,” *The Lancet*, vol. 377, no. 9773, pp. 1264-1275, 2011.

[31] K. He, X. Zhang, S. Ren and J. Sun, ”Deep residual learning for image recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770-778.

[32] G. Huang, Z. Liu, L. Van Der Maaten and K. Q. Weinberger, ”Densely connected convolutional networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4700-4708.

[33] 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.

[34] H. Qassim, A. Verma and D. Feinzimer, ”Compressed Residual-VGG16 CNN Model for Big Data Places Image Recognition,” in *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, 2018, pp. 169-175.

[35] G. Koch, R. Zemel and R. Salakhutdinov, ”Siamese Neural Networks for One-Shot Image Recognition,” in *ICML Deep Learning Workshop*, vol. 2, no. 1, 2015.