

Med-ConvNeXT: A Multi-Class Edge-Compatible Image Classification Model for Respiratory Disease

Parvez Belim¹, Nirav Bhatt¹, Amit Lathigara¹, and Vikas Kumar Sabata²

¹Department of Computer Engineering, School of Engineering, RK University, Rajkot, India

²KIIT University, Odisha, India

Corresponding author: Parvez Belim. (e-mail: parvez.belim@rku.ac.in), **Author(s) Email:** Nirav Bhatt (e-mail: nirav.bhatt@rku.ac.in), Amit Lathigara (e-mail: amit.lathigara@rku.ac.in), Vikas Kumar Sabata (e-mail: vikas_sabat@kiit.ac.in)

Abstract Respiratory abnormality detection in chest X-ray images can be assisted and automated via machine learning models. Many of those models are very effective in detecting abnormalities, but they require large computational and memory resources. Running these models on edge devices often leads to delays in inference or crashes, as most of the devices have memory constraints. Therefore, a more edge-aware image classification model called Med-ConvNeXT is developed by keeping both memory and performance in mind. The proposed model introduces a kernel reparameterization technique called receptive kernel fusion, which combines useful information from kernels ($1 \times 1, 3 \times 3, 5 \times 5$) to calculate a receptive 5×5 kernel. The kernel calculated by the above method can extract both lower- and higher-level features without increasing FLOPs or the network's parameter count. To further reduce the number of parameters and FLOPs, a novel channel pruning method, called Memory-based Channel Pruning, is introduced. This method compares the classification performance of different kernels and ranks them to eliminate the least important kernels. This reduced the total channel size from 96 to 64 in the first stage. Combining the proposed kernel reparameterization and channel pruning technique reduced the parameter count of the ConvNeXT model by up to 54.7% while achieving a classification accuracy of 96.53% for X-ray images. The proposed model achieved 93.7% sensitivity and 95.9% specificity with a parameter count of 12.96 million. The proposed model was implemented on Raspberry Pi 5 hardware and got an inference latency of 985ms using a 118 MB memory footprint. This performance is achieved with an input image dimension of a 512×512 X-ray image. The proposed model balances both image classification performance and edge implementation efficiency. The proposed model can achieve better performance on edge devices without the need for expensive server and cloud infrastructure.

Keywords ConvNeXT, Kernel reparameterization, medical image classification, Channel pruning, CNN, edge application.

1. Introduction

The most common and effective form of early-level screening is chest radiography. Recent estimates indicate that more than 2 billion screenings are performed worldwide each year. This screening method is used to detect major respiratory disorders such as pneumonia, tuberculosis, and lung cancer. Early detection using these methods can increase survival rates and reduce complications during treatment. Increasing the number of X-ray machines can reduce the severity of disorders in later stages of life. However, the main barriers to high adoption are the need for expensive X-ray machines and the difficulty of detecting early symptoms in patients. This gap in real-time adoption can be addressed by introducing new deep learning algorithms for diagnosis that can

enhance the decision-making capability of radiologists. The next problem we address is the need for expensive hardware and the portability issue associated with existing techniques. Most existing deep learning models require high-speed FLOPs and memory for inference, which can be achieved by connecting to a central server or by using hardware with an expensive GPU and memory [1]. This increases the need for models that can be implemented on edge devices with better classification accuracy [2],[3],[4],[5].

The computational cost of the deep learning algorithm depends entirely on the model's architecture. A Vision transformer model, trained on a medical dataset, has a complexity proportional to $O(H \times W)^2$, where the complexity depends on the size of the input image. This introduces quadratic scaling, which makes

it more computationally expensive for edge-device applications. At the same time, convolutional models scale linearly with spatial dimensions, with computational complexity directly proportional to the square of the kernel size (k^2). Increasing the input size of an image increases the overall computational cost of the ViT-based architectures. The main disadvantage of ViT-based algorithms is their pre-emptive assumption that all spatially related pixels contain similar [6],[7],[8] information, even when their spatial positions change in the image. This can be achieved through extensive training on many input datasets, which can be a major disadvantage for novel respiratory diseases. Smaller models, such as ViT-small, do not suffer from overfitting but have lower classification performance than ViT-Base, which does, particularly on small datasets. This renders ViT-based algorithms unsuitable for small-edge-device classification applications [9]. Existing small models often struggle when image complexity increases, making them incompatible with real-time edge applications.

Computationally efficient models with high classification accuracy include convolutional models such as EfficientNet, ResNet, MobileNet, and DenseNet. These models have a strong positional understanding of images with faster learning and lower computational costs during inference. Among those models, one edge-adaptive model is MobileNet, which performs at 0.3 GFLOPs. This algorithm has a significant advantage in terms of size and resource usage, but it often lacks the necessary accuracy (93.5%) for sensitive applications such as medical image classification [10]. Smaller models such as this often struggle with multiclass classification and with dense feature datasets. Very deep networks such as ResNet and EfficientNet achieve good performance on medical image datasets, but their large size makes them unsuitable for edge applications. The DenseNet algorithm's feature-concatenation capability, along with its small number of parameters (~8 million), makes it suitable for edge applications. However, these models often suffer from memory allocation issues on edge devices, and FLOPs are not the primary bottleneck for memory allocation. The main objective is to increase efficiency through powerful math instead of high hardware spending. ConvNeXT models have good performance in image classification, but making them edge-compatible is the biggest challenge. This creates a good opportunity for an edge-compatible ConvNeXT model version. [11],[12],[13],[14],[15]

ConvNeXT is a new architecture for convolutional neural networks trained by modernizing traditional principles of CNNs with architectural ideas inspired by Vision Transformers. It adapts several components with regard to traditional CNNs: the design of the network blocks, size of convolutional kernels,

normalization strategy, activation functions, and stage configuration. In this way, ConvNeXT achieves competitive image-classification performance without discarding the benefits of convolutional architectures. Todi et al. Some research has applied ConvNeXT-based architectures and found they may generalise better than previous modern models [16]. A visual comparison of representative architectures based on parameter count and inference latency is presented in Fig. 1, providing insight into the efficiency-performance trade-offs considered during model design. [11]

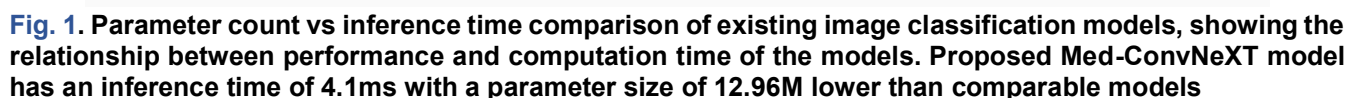
To reduce memory utilization and FLOPs, the proposed model uses fewer channels and smaller kernel sizes than existing ConvNeXT models, such as ConvNeXT-tiny and ConvNeXT-small. Instead of a 7x7 convolution in each stage, we use a 5x5 kernel to extract features, reducing the number of parameters required by the model. Improving kernel scaling can make the model more accurate while reducing the parameter count. The main problem in performing image classification on edge devices is the ability to load weights into registers; all edge devices have limited memory space. Therefore, reducing memory bandwidth can improve inference time on edge devices [17],[18],[19],[20].

II. Related Work

After the introduction of the Vision Transformer algorithm, there is a common belief that ViT outperforms image-processing models with a CNN backbone. Liu et al. (2022) improved the performance of CNN models by modifying the architecture and training framework, building on ViT training. The model replaced 3 x 3 kernels with 7 x 7 kernels, increasing the model's overall feature-extraction capability. The main changes are the use of LayerNorm instead of BatchNorm, the Adam optimizer, and an inverted bottleneck [6]. Wang et al. (2023) used a larger kernel size with fewer channels to improve efficiency. To improve classification performance, the proposed model fuses surface-level and deep features to increase information concentration per channel [21]. Reducing channel sizes can be a better strategy for reducing model size without compromising performance. ConvNeXT models achieve better classification performance than other CNN architectures, but to make them suitable for edge-device applications, inference time, memory utilization, and FLOPs need to be optimized. Zhao, et al. (2024) introduced RepNeXT, a new architecture based on ConvNeXT, which uses multiple kernels during training to extract different features [22]. This kernel representation is then combined into a single 3x3 kernel for easier inference, thereby making the model lighter for edge devices. This type of kernel rearrangement can affect optimization; therefore, a

Ding et al. (2019) found that using a single kernel shape in a CNN often misses directional features, which are important for high-accuracy classification. To overcome this problem, the authors combined 1×3 horizontal and (3×1) vertical kernels to generate an effective 3×3 kernel for classification. This

reparameterization method outperforms the classic VGG-based CNN architectures [9]. This type of model training procedure, with increasing complexity during training and decreasing complexity during inference, has been widely adopted [10]. CNN models often struggle with limited input context; this can be addressed by implementing transformer-based attention over image pixels.



Wang et al. (2023) used fully connected layers in a transformer architecture to estimate the mixing matrix for reparameterization. This type of model merging through channel mixer (FFN) layers can increase the performance and accuracy [23]. The major problem with ViT-based architectures lies in memory utilisation, which makes these models unsuitable for edge applications even when they have lower FLOP requirements and parameter sizes. For better kernel reparameterization, there is a need for a more effective mixing framework that combines the feature-extraction potential of kernels of different dimensions [24]. Channel pruning is an important strategy to lower the total FLOP operations and memory usage in an edge device. This reduces the number of channels in the model, thereby reducing the overall parameter size. The first introduction of channel pruning was performed by He, et al. (2017), in which Lasso regression analysis was performed on the model. The most important channels in the model were then selected for training

[24]. Any optimization across independent layers exponentially increases the model's complexity and error as the network becomes deeper. Instead of independently determining the channel's relevance after training, a new mechanism that uses batch-norm scaling factors was introduced. Liu, Z. et al. (2017) introduced a new loss function that reduces sparsity. Based on these values, the channels are ranked during training [25]. This sparsity-based method often neglects channels that contain the most semantic information, relying instead on simple weight values, thereby degrading the model's classification performance. The model can be pruned to multiple layers by understanding dependencies between channels. Fang et al. (2023) introduced a method for calculating channel dependencies by forming a graph-based dependency group. These values are then used to determine the importance of each channel before pruning [26]. The graph-based decision-making framework depends on external scores, and incorrect

variable prediction can lead to irregular pruning. Therefore, our model proposes a memory-based pruning method that calculates channel importance scores. Channel-wise attention mechanisms, such as Squeeze-and-Excitation networks, can recalibrate feature responses and improve feature representation [27]. Transformer-based medical image classification has also been explored for limited-data settings [28].

There is a huge volume of existing research that has been conducted to make image classification models lightweight for edge device applications. Korkmaz, O. E. (2025) implemented five different depth-wise convolution variants on edge devices to compare the performance [29]. The results show that many of the classification models suffer from memory performance instead of inference time and accuracy. The edge application results indicate that a high parameter count and high memory traffic, resulting from more complex input image dimensions, can impact memory usage. The research uses a 64*64 input image dimension, but when applied with a 512*512 input image, the memory allocation performance of the model can be impacted [18]. Therefore, the study should focus on increasing the pixel size to 512*512 to understand the performance of the model. Therefore, a more compact and effective image classification model needs to be trained for edge deployment.

A. Research Gap

The main problem with existing models is the trade-off between classification accuracy and edge deployment potential. Instead of pruning the channels to reduce parameters, a model-performance-based channel-pruning method is proposed that removes channels that contribute less to the classification. The kernel reparameterization, instead of reducing the kernel dimensions, is a kernel fusion method that is introduced to combine the capability of kernels with both small and large kernels. The proposed model architecture and training procedures are discussed below.

III. Method

The motivation for this change comes from recognizing that the useful information in medical images responsible for a particular diagnosis constitutes only a small percentage of the original image. An effective channel pruning technique can reduce memory usage during inference. The proposed neural network architecture is designed to optimize memory rather than FLOPs, as most modern devices can effectively handle multiple floating-point operations. The main problem lies in memory optimization. The proposed architecture uses a depth-wise convolutional layer for kernel selection, similar to the ConvNeXt architecture. To improve the model's inference efficiency, we introduce a memory-aware reparameterization (MAR) technique. This technique consists of multiple branches used to extract non-separable pathological features

from the input image data. The main objective of this study was to enable the model to focus on the most important regions in the medical image for classification. This reduces memory retrieval and FLOP operations during inference and makes the model faster and more accurate.

To achieve this, we use both channel and spatial pruning to reduce the number of channels and input pixels during inference. The proposed methodology can be applied to an image classification task in which the accurate classification of different structures and shapes is necessary. Existing models, such as ConvNeXt, use learnable convolution kernels that differ for each channel at each stage. The learnable kernel in existing methods can cause overfitting, and the proposed method introduces an adaptive kernel-freezing method based on training error values. Instead of selecting the kernel size to identify the independent components for each channel, the proposed methodology allows an adaptive kernel selection method that automatically trains the model to choose the kernel. The proposed method modifies four parts of the architecture: the kernel geometry level, the spatial activation level, the partial freezing level, and the training and validation levels.

A. Pre-Processing

All the raw input images are converted to a uniform format for healthy training, so the images are resized to 512 × 512 dimensions. Resizing the image ensures that all images have the same spatial footprint across all training epochs. Resizing can be represented using the equation Eq. (1) [2].

$$X = F_r(H, W, Z) \quad (1)$$

Here, X is the resized image, F_r the resizing operator, H and W the target dimensions, and Z the input image. The resized image is then z-normalised from pixel values [0,255] to [0,1] for better training performance. In Eq. (2) [2], X represents the resized image and μ, σ are the mean and standard deviation values of the image.

$$I = \frac{X - \mu}{\sigma} \quad (2)$$

The processed images are then split into training, testing, and validation sets for training the model.

B. Kernel Reparameterization

The proposed algorithm works by understanding the importance of small objects or nodules in an image. The previous ConvNeXt architectures normally use fixed kernel sizes of 7, which increases the receptive field of the model to capture more information from input images. Medical images, which contain more sensitive information, require kernels at multiple scales to extract useful information. The classification model must extract small pulmonary micronodules into large nodules without missing any valuable information. Existing methods, such as a Mixture-of-Experts (MoE) approach

that combines information from multiple kernels, can increase overall FLOPs and memory operations, thereby increasing inference time and overall load on edge devices. Novel reparameterization methods can achieve this goal [30],[31],[32],[33].

The computational cost of complex classification tasks is linearly correlated with the kernel size; therefore, reducing the kernel size can be a suitable architectural change to improve inference time and cost. To identify similar classes during training, temperature values (τ) were used, which retained information from different classes during training. Existing methods use a form of feature matching between layers, which can be counterintuitive in our architecture. The main reason for this is a change in the number of channels due to channel pruning, which changes the architecture and affects feature matching. Therefore, instead of feature matching, attention matching between layers was implemented [34],[32],[33],[34]. The kernel reparameterization technique uses a kernel of three sizes: 1×1 , 3×3 and 5×5 kernels. These kernels are used simultaneously during training and the attention map (A_k) values of three kernel outputs are estimated using their corresponding feature maps (F_{c_k}), where k is the kernel type. Then the score S_k for each kernel, a calculation will be made, which can be represented by a formula in Eq. (3) [30].

$$S_k = \frac{\sum_{i,j} A(i,j)T(i,j)}{\sum_{i,j} A(i,j)} \quad (3)$$

where, $A(i,j)$ is the predicted attention map and $T(i,j)$ is the true bounding box value from training annotations. Then a SoftMax value is applied to the score values S_k to normalise the weight values. Then, zero padding is applied to the weights of 1×1 and 3×3 kernels, and a combined weight value is calculated using the fusion function formula Eq. (4) [32], Eq. (5) [32].

$$W_f = \sum_{k=1,3,5} w_k \bar{W}_k \quad (4)$$

Here, W_f is the fused kernel, w_k the branch weight, and $\bar{W}_k$ the padded kernel.

$$b_f = \sum_{k=1,3,5} w_k b_k \quad (5)$$

The fusion formulation is consistent with established structural reparameterization methods [32]. The resultant weight function is used to merge the weights of all kernel values into a resultant kernel of size 5×5 . This resultant kernel has the local small feature extraction capability of the 3×3 kernel and the resultant 5×5 kernel has a similar receptive field. The resultant values are then used for training operations in convolution layers Eq. (6) [32].

$$F = I \cdot W_f + b_f \quad (6)$$

Here, b_f is the fused bias, W_f the branch weight, b_k the branch bias, F the output, and I the input. The

training is performed in three parallel branches, and the per-epoch loss is used to estimate the degree of feature mixing from the kernels.

C. Channel Pruning

In addition to reducing the number of memory-call operations, the proposed architecture employs an adaptive channel pruning method to reduce channel usage. In image classification inference, a small 224×224 image can trigger a memory allocation of over 150 MB, thereby increasing computation cost and inference time on edge devices. ASP reduces the number of [35],[36],[37],[38] input pixels processed at each stage of the ConvNeXt algorithm by removing image regions that contribute the least to the model's classification output. Channel pruning must be performed effectively to avoid losing useful information from the input signal during inference, which would reduce performance. The existing ConvNeXt model has 96 channels at the first stage; therefore, a more optimized number of channels is selected using the proposed method. The effective number of channels was selected to balance the total pruning percentage and the total reduction in FLOPs, as shown in Fig. 2.

D. Channel importance score

The importance score for each kernel was measured by calculating the gradient of the loss ΔL_T using the total loss function above. The kernel channel score for the channel i can be represented using the equation Eq. (7) [35],[37].

$$K_i = \frac{1}{N} \sum_N \left(\frac{1}{H \cdot W} \sum (\Delta L_T \cdot F_c(h, w)) \right)_{max} \quad (7)$$

where (h, w) are the dimensions of feature maps, F_c is the feature activation function, and N is the number of channels. The kernel's scores are stored at each epoch, and the trend in kernel importance is measured to identify the most important kernels in the model. K_i will indicate the importance score for each kernel at each stage of the training. The loss function (L_T) also helps ensure that the proposed kernel values remain stable across epochs, reducing unnecessary variance in classification performance. Instead of removing the channels that show a very low score at the start of the training, a memory function is used to evaluate the trajectory of change in channel score. The channels that score consistently over a period of epochs will normally be selected for pruning. The memory function can be represented using the formula in Eq. (8) [38].

$$V = \alpha V + (1 - \alpha) K_i \quad \alpha = 0.9 \quad (8)$$

where, V holds the memory of the last 10 epochs, and based on that memory value, the memory will be updated using channel scores, K_i is the current importance score for channel i , and $\alpha=0.9$ controls the contribution of the previous memory. The memory is updated throughout training to reduce unstable early pruning decisions [38].

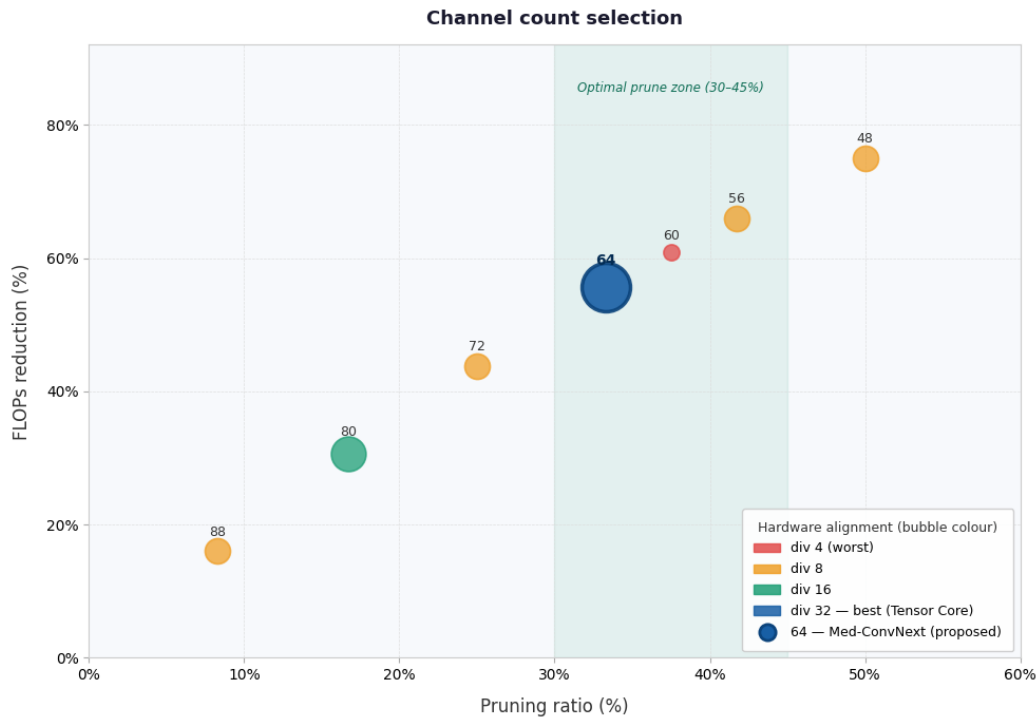


Fig. 2. Selection of the number of channels without affecting the performance and computation time of the model. To find performance only first stage convolution layer is considered to find the optimal channel count. With 64 kernels the FLOP is reduced upto 54.7%.

This memory-based performance analysis method can prevent improper removal of kernels, thereby avoiding poor performance in the initial stages of training. Channel pruning, which removes the least important kernels, occurs only after the importance scores are stabilized, thereby ensuring that no important kernels are deleted. The channel-importance and memory formulations follow established pruning and optimization approaches[35],[38]. Then, Perturbation Sensitivity (PS) values are estimated by turning off a few channels with low K_i scores, and the total change in classification accuracy is noted. The sensitivity values measure the difference in accuracy of the model before (ACC_{i+1}) and after removing the channels (ACC_{i+1}) Eq. (9) [35].

$$PS_i = ACC_i - ACC_{i+1} \quad (9)$$

These values are combined to estimate the final K_i score, which is used to prune unimportant channels. The number of channels to be removed can be set during training. In our proposed architecture, we chose 64 channels as the optimal starting number, but reducing the number of channels below this range often degrades the performance of the trained model. Therefore, optimal channel selection is important; careful selection reduces the parameter count from 28.58 million to 12.96 million, showing the potential of the Med-ConvNeXT model in edge applications.

E. Attention-based gradient loss function

The loss functions (L_T) are calculated based on the model's stability and correctness during training. The proposed loss function is a combination of the existing Binary Cross-Entropy (BCE) loss (L_C) function, the attention loss function (L_A) and adaptive weight values (λ_t). The proposed loss function is represented as follows: [30],[39]

$$L_A = \frac{1}{H \cdot W} \sum A(i, j) - T(i, j) \quad (10)$$

$$L_C = -\frac{1}{N} \sum y_{c,n} \log P_{n,c} \quad (11)$$

$$L_T = L_C + \lambda_t L_A \quad (12)$$

where classification loss (L_C) is calculated using the predicted probability ($P_{n,c}$) and ground truth values ($y_{c,n}$). Attention loss (L_A) is the difference between attention value ($A(i, j)$) and true value ($T(i, j)$), $T(i, j)$ is the target attention value; H and W are the spatial dimensions; λ_t is the adaptive attention-loss weight, $H \cdot W$ are spatial dimensions. The dynamic weight function reflects the model's ability to align with the most important features for classification. The attention weight function enables the model to focus on the most important regions during training, thereby increasing its stability by controlling the loss values. These values depend on the Attention Index (AI), which penalizes the model when its attention moves beyond the desired locations. The AI value was measured by calculating the deviation in the model's attention after predicting the output during

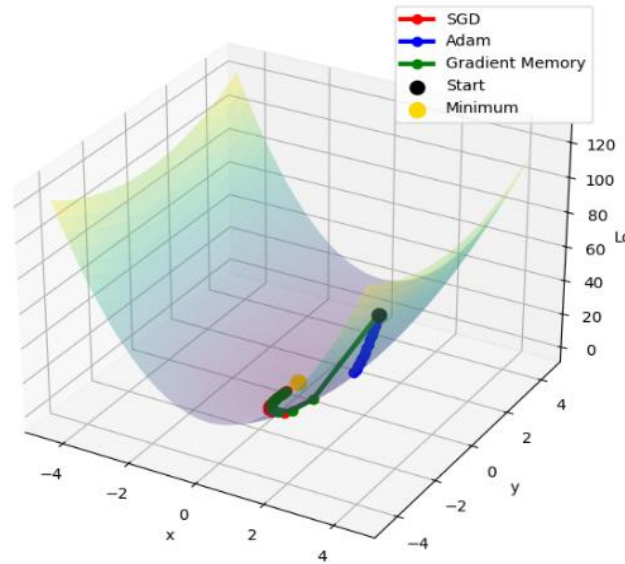


Fig. 3. Comparison between different optimization methods in finding the optimization results after every epoch. The figure represents the trajectory of different loss functions and memory function seems to have smooth trajectory and achieved global minimum faster

each epoch. The epoch value is measured by assessing the alignment between the predicted attention and the original training data, as well as the level of certainty shown when selecting the region, which can be quantified using the compactness score. The value of the attention index can be represented by the following formula Eq. (13) [30],[37].

$$AI = \left(\frac{\text{Alignment} \times \text{certainty}}{1 + V_{GD}} \right) \quad (13)$$

The attention index (AI) indicates the deviation of the predicted attention from the actual attention in the spatial dimension, and the certainty score quantifies the concentration of attention in the predicted region, thereby estimating the overall certainty of the prediction. The value of the gradient direction variance (V_{GD}) indicates sudden changes in gradient values and helps prevent gradient oscillations during training. Gradient values are meant to guide training in the proper direction when the attention score changes suddenly, and to prevent overfitting and underfitting on the given dataset. The gradient weight (λ_t) value is represented by the following equation Eq. (14): [39]

$$\lambda_t = \lambda_{base} + AI \quad (14)$$

Here, λ_t is the adaptive loss weight, λ_{base} the base weight, and AI the attention index. The proposed method uses a memory function that captures the gradient trajectory and updates the weights with greater stability than previous methods, such as Adam [38] and ASAM [37]. The individual layer weights were updated using the chain rule. Attention loss is applied to data with bounding box annotations, enabling the model to train only on correct attention targets. The

gradient loss function comparison between different loss functions is shown in Fig. 3.

The training procedure of the proposed Med-ConvNeXT model is summarized in Algorithm 1. The algorithm begins with a 512x512 input image, followed by iterations over the dataset, the convolutional block, and the chosen kernel sizes of 1×1 , 3×3 , and 5×5 . As the model trains, attention scores and kernel importance scores are determined, after which the kernel weights are padded and fused for the final kernel representation. After this, the attention-based loss function is applied, and the gradient-related weight is computed to compute the total loss. The channel pruning step involves the calculation of importance scores of the channels and iterative update of the ranking values for maintaining the desired number of channels. The architecture diagram of the proposed Med-ConvNeXT model training is shown in Fig. 4 below. The pre-processing stage of the model will convert the images to a uniform dimension of 512x512 and split the images and bounding boxes into separate sections for training. In the next phase of training, both channel pruning and kernel reparameterization act in parallel, as both processes share feature maps and loss value. Kernel reparameterization shares feature maps with channel pruning to find the most important channels, and the attention loss values during channel pruning are sent to kernel reparameterization to train the kernels. The calculated channels are applied to each stage of the ConvNeXT model and training is continued. This training process reduces the number of channels and increases the performance of kernels. Existing ConvNeXT models have predefined kernel

Algorithm. 1. The entire algorithm flow of proposed Med-ConvNeXT model is shown below:

Algorithm: Med-ConvNeXT model training using memory-based kernel reparameterization and channel pruning, and adaptive loss function

Input: 512*512 image, epoch, learning rate, annotated bounding box

Output: Numerical value for probability of each class

Representations: Dataset->D; Epoch->E; Kernel->K={1×1, 3×3, 5×5}; Targets->C; Temperature->r; Memory->V; Input->x; Bounding box->T

#Kernel reparameterization

for epoch (e) = 1 to E;

for Data=1 to D;

for conv_block=1 to N;

for kernel k->K;

$A \leftarrow \text{Atten}(F_c^k) \leftarrow \text{Conv}(x)$ #Compute attention score

$S_k = \frac{\sum_{i,j} A(i,j)T(i,j)}{\sum_{i,j} A(i,j)}$ #Compute kernel importance score

end for

$w_k = \text{softmax}(\frac{S_k}{\tau})$ # Compute individual weights in kernel k

Add padding to 1 × 1 and 3 × 3 kernels for merging

$W_f = \sum_{k=1,3,5} w_k \bar{W}_k$ #merge weights of kernel

end for

Attention-based loss

Compute $L_{ce} < -\text{crossentropy}$; $L_{A_e} < -\text{attention_loss}$

Compute $\lambda_t < \text{-gradient weight}$

Calculate $L_T < L_C - \lambda_t L_A$ #Total loss function

#Channel pruning

for channel i=1 to n;

$K_i = \frac{1}{N} \sum_N \left(\frac{1}{H*W} \sum (\Delta L_T \cdot F_C(h, w)) \right)_{\max}$ #Calculate importance score for each channel.

$V_i < -\text{Channel rank}$

$V = 0.9V_i + 0.1K_i$ #Update rank values after each iteration

If $V_i > n$ where n is desired number of channels

$k_i < -\text{prune the channel}$

end for

end for

return Θ^* # final weight

sizes for convolution layers, often ignoring regions in the image that require varied kernel sizes. This gap can be rectified using Med-ConNeXT models, which employ a kernel fusion operation that merges weights across multiple kernel sizes, leading to better generalization. The kernel fusion operation is also used in methods such as RepVGG and RepNeXT, but they don't incorporate pathology-aware attention-based weighting. This adaptive weighting approach in the Med-ConvNeXT model ensures better feature extraction. Lightweight models like MobileNet and EfficientNet have fewer parameters but are not optimized for memory allocation, which is the biggest bottleneck for edge device implementation. The

introduction of memory-based channel pruning and an attention-guided loss function reduces both memory usage and parameter size, making the model ready for edge deployment.

IV. Result

A. Dataset specifications

The performance of the proposed Med-ConvNeXT model is evaluated on the publicly available VinDr-CXR dataset. The dataset comprised 15,000 radiographic images from the VinDr-CXR chest X-ray dataset collected from two hospitals in Vietnam. There are two folders in the original dataset: test and train. The study only used the train folder and neglected the test folder

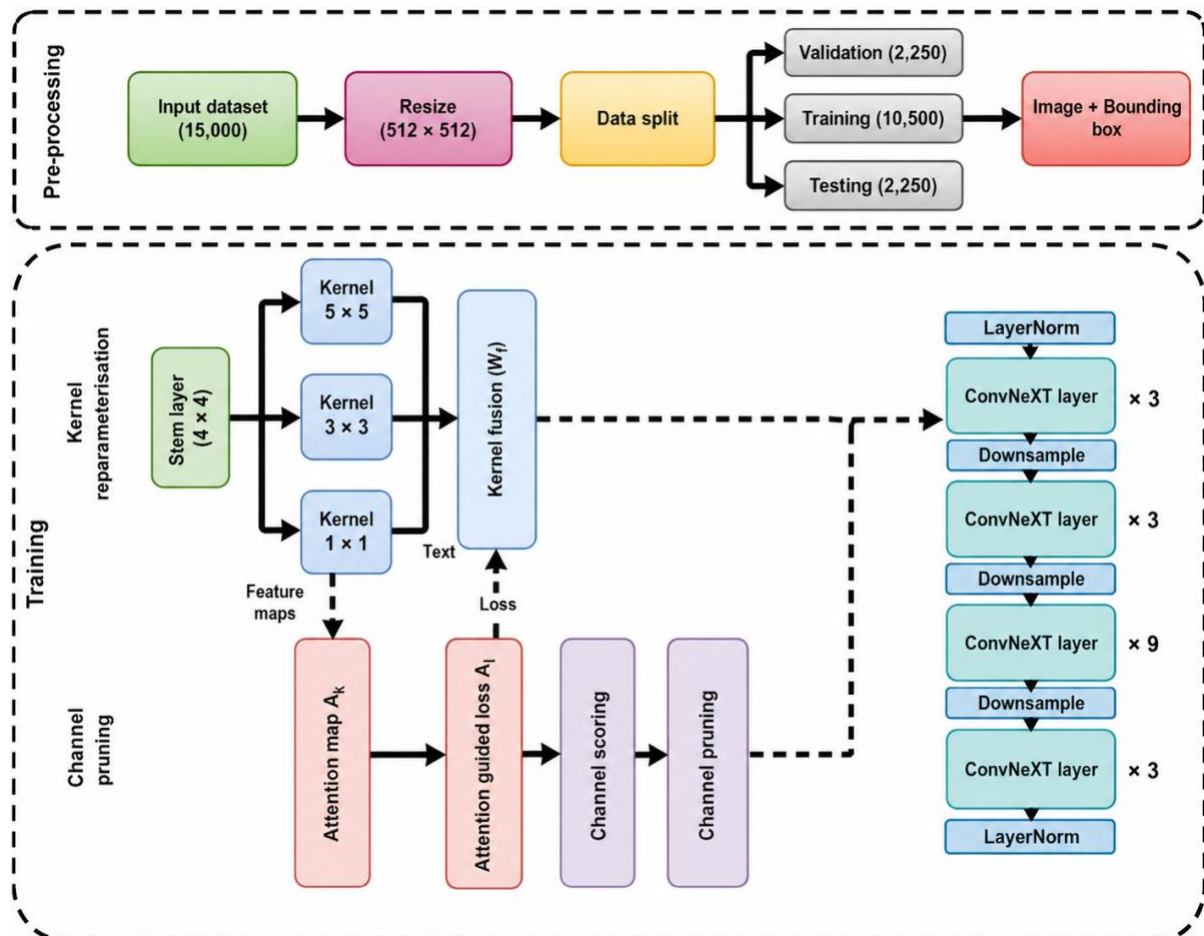


Fig. 4. Architecture diagram of proposed Med-ConvNeXT model explaining both the pre-processing and training phase. The first block represents how the input image is resized to 512 *512 and the dataset is split into training, validation, and test groups. In the second block, the working of kernel reparameterization and channel pruning and how it integrates with the ConvNeXT backbone model

as it doesn't have any bounding box annotations. The dataset contains train (10,500 images), validation (2,250 images), and test (2,250 images) folders. Therefore, the train dataset is divided into three folders: train, validation, and test. The images are pre-processed to JPEG images for implementation on edge devices. The dataset was split into training (70%), validation (15%), and testing (15%) subsets to represent all classes equally, as detailed in Table 1. There is a class imbalance in the original datasets themselves, as more common diagnoses such as no findings and lung opacity have a higher number of samples than other findings. The dataset split was done carefully to make sure there is no data replication happening between training, test, or validation sets. The validation dataset was used for parameter tuning, and the test dataset was used for measuring the model's performance. The test folder in the original dataset is excluded because we need to check the bounding box attention loss performance of the model,

which is not present in the original dataset. There is no patient-level split while preparing the dataset.

The images in the dataset are annotated with bounding boxes by 3 different radiologists and consolidated together during training. The training doesn't include patients' demographic data or medical history. It includes only the image and an annotation indicating the type of abnormality detected in that region. There are a few annotations mentioning the types of abnormalities, including Aortic enlargement, atelectasis, Consolidation, Calcification, Cardiomegaly, Infiltration, Interstitial Lung Disease, Lung Opacity, nodular mass, and lesions. Pleural effusion, Pneumothorax, Pulmonary fibrosis, and Pleural thickening.

The model is trained using bounding boxes annotated by a radiologist to generate attention loss. In case of the absence of bounding box annotations, the model is trained using classification loss, and attention loss will be zero for the normal dataset. Optimizations

are done for data with annotations; all the data without any annotation is left idle during that stage.

The dataset was initially in DICOM format, which was then pre-processed to JPEG and trained on an NVIDIA A100 GPU using Google Colab. The training uses 50 epochs and a learning rate of 1×10^{-4} . Then the trained model is tested on a Raspberry Pi device for edge compatibility. A simple representation of the

(19)[40]. TP, TN, FP, and FN denote true-positive, true-negative, false-positive, and false-negative counts, respectively. The reported multi-class values are macro-averaged unless stated otherwise.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (15)$$

$$Sensitivity = \frac{TP}{TP+FN} \quad (16)$$

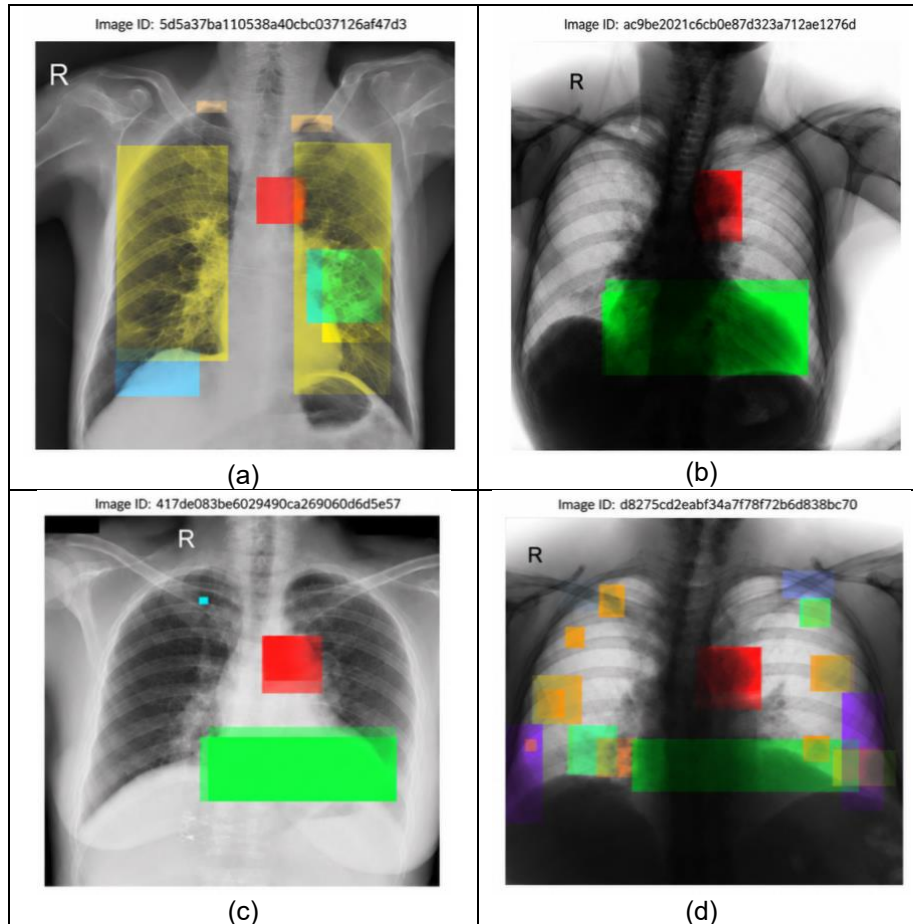


Fig. 5. Regions of Interest in the dataset marked by the radiologist. The bounding box represents different classes diagnosed by radiologist and annotated (a) Aortic enlargement, (b) Cardiomegaly, (c) Pleural effusion, and (d) Pulmonary fibrosis

region of interest, as marked by doctors, is shown in Fig. 5 above. The red region indicates highly important areas, suggesting a lack of focused regions used for image classification. The main goal of the proposed method is to focus on the marked regions when classifying the image, rather than looking at the image randomly. The training and validation performances of the proposed model can be understood using three metrics: sensitivity, specificity, and accuracy. These measures are defined from the confusion-matrix counts and are widely used for multi-class classifier evaluation [40]. In addition to accuracy, sensitivity and specificity values provide insight into the model's ability to identify false positives and false negatives. For Eqs. (15) –

$$Specificity = \frac{TN}{TN+FP} \quad (17)$$

$$Precision = \frac{TP}{TP+FP} \quad (18)$$

$$F1 = \frac{2(Precision)(Sensitivity)}{Precision+Sensitivity} \quad (19)$$

These standard performance measures follow established multi-class classification evaluation practice[40].

At each stage, the number of channels was reduced by up to 54.7%, significantly reducing the number of parameters and FLOP operations per inference. The reduced number of channels also reduced the number of parameters from 28.58 million to 12.96 million for an input image size of 512 x 512. This is a 54.7% reduction

in parameter count without a reduction in the model's classification performance. A stage-wise comparison

Table 1. Detailed Dataset Split Distributions and training specifications

Dataset Split	Images (n)	Percentage (%)
Training	10,500	70%
Validation	2,250	15%
Testing	2,250	15%
Input image size	512 × 512	
Learning rate	0.0001	
Weight decay	0.0001	
Batch size	2	
Random seed runs	1	
Total epoch	50	

between ConvNeXT-Tiny, the smallest model in that architecture, and the Med-ConvNeXT model reduces the parameter count to 12.96 million by reducing the channel numbers. The channel pruning reduces the parameter count at each stage of the model, as shown in Table 2. The comparative analysis of the parameters and FLOP counts of different models with respect to their inference time is shown in the table below. Compared with other models, Med-ConvNeXT models have a very low number of parameters and FLOPs for the same input dimensions, as shown in Table 3. These reduced parameters and FLOPs are reflected in the shorter inference time. This classification is done on a GPU with INT8 precision mode with the same input specs for all convolution models. This reduced inference runtime is also associated with low memory

usage, making the proposed model highly suitable for edge-device applications. This reduced FLOP count shows the effectiveness of the proposed channel pruning method. The reduced FLOP operations and parameter count also reduce the inference time by 47.4%, making the Med-ConvNeXT model faster than its architecture-similar models. When compared to smaller models, the only model that has fewer FLOPs and faster inference is the MobileNetV2 model, which achieves inference at 3.8ms compared to Med-ConvNeXT; it is 7.32% faster. This may be due to the lower parameter count in the model. The next model that achieved similar inference time even with a high parameter count is the ResNet-50 model. Compared to other ConvNeXT models, the proposed Med-ConvNeXT model is faster. The performance of the proposed model in classifying different image classes was compared in the following image. The macro-level sensitivity (93.7%) and specificity (95.9%) values show positive signs, but the model has lower precision (77.2%) due to class imbalance in the dataset. The model has an F1 score of 86%, which is strong for multi-class image classification. The performance of the proposed model was consistent across all classes, making the model suitable for testing on edge devices for image classification. The performance of the proposed model per class is shown in Fig. 6 below. The classification performance is further visualised using a confusion matrix using true class prediction result values. The classes with high training samples have high prediction accuracy, while models with fewer samples have very minimal prediction accuracy. The input image datasets that are most confused are the Lung opacity, infiltration, and consolidation classes. These are the classes with the opacity problem.

Table 2. Number of parameters optimized in the proposed model

	ConvNeXT-Tiny			Med-ConvNeXT		
	Dimensions	Channels	Parameters (M)	Dimensions	Channels	Parameters (M)
Stem			0.0049			0.0033
Stage 1 Downsample (Stage 1)	56×56	96	0.2376	56×56	64	0.1092
Stage 2 Downsample (Stage 2)	28×28	192	0.0741	28×28	128	0.033
Stage 3 Downsample (Stage 3)	14×14	384	0.9176	14×14	256	0.4151
Stage 4	7×7	768	0.2957	7×7	512	0.1316
Head			10.8138			4.8499
			1.1812			0.5253
			14.2871			6.379
			0.7705			0.514
			28.5825			12.9604

Overall, the model performs with better accuracy and sensitivity for most of the classes, making it an effective

insensitive towards class imbalance, and it is shown in Fig. 8 below. The curve demonstrates that classes with

Table 3. Inference time comparison between models

Model	Number of Parameters (In Millions)	Floating Point Operations (G)	Inference time (ms) (INT8)
MobileNetV2	3.5	0.3	3.8
EfficientNet-B0	5.29	0.39	6
DenseNet-121	7.98	2.87	6.2
ResNet-50	25.56	4.1	5.2
ResNet-101	44.55	7.8	7.8
ConvNeXt-Tiny	28.59	4.46	5.5
ConvNeXt-Small	50.22	8.7	7.8
Med-ConvNeXt	12.96	2.31	4.1

classifier model. The confusion matrix for our proposed model is shown in Fig. 7 below. To understand the performance of the model, the per-class ROC curve is estimated using the true positive rate and false positive rate by changing the threshold values in increments. The mean AUC value from all the classifiers is 0.982, which is very high compared to most classifiers. The value of

fewer training samples often have reduced classification performance. The class-specific sensitivity values show that the problem of lower precision is due to lower training data availability.

B. Edge implementation

The proposed model is implemented on Raspberry Pi 5 hardware, and the total inference time is calculated. The

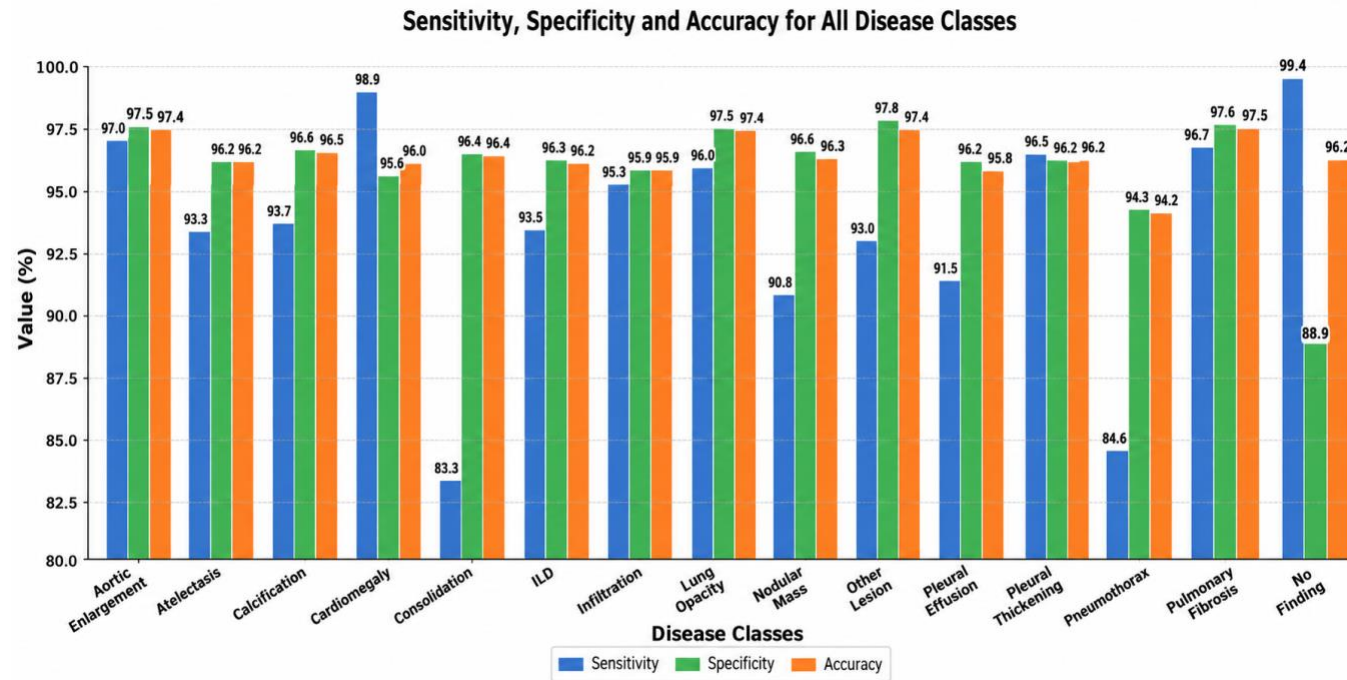


Fig. 6. Multi-class classification performance of the proposed model for each class, showing a consistent classification performance for all the classes. The model has better classification performance with macro-sensitivity of 93.7% and macro specificity of 95.9% calculated for test dataset

AUC is low for classes with a minimum sample size, and the average classifier performance is above most existing classifier models. The ROC curve is highly

inputs are quantized to INT8 using Eq. (20)[20]. Here, q is the quantized integer output, x is the original input value, s is the scale factor, z is the zero-point,

r denotes the rounding operation, and clip limits the result to the supported integer range. The quantization equation follows the standard integer-arithmetic inference approach in Eq. (20) [20].

$$q = clip \left[r \left(\frac{x}{s} \right) + z \right] \tag{20}$$

The proposed model uses an image of size 512*512, which is 64 times larger than previous techniques that implemented convolutional models on a Pi device. The latency and inference time are calculated by a time counter, and memory usage is calculated by analysing RSS usage. The power is measured using an external log meter, and energy is calculated using the same power multiplied by time principle. The inference time of depth-wise convolutional models is only 20.2ms, which seems lower than our proposed model's inference time of 985ms, which is almost a second of time. The processing time doesn't include image loading and pre-processing steps of the model with an estimated memory footprint of 118MB. Still, the number of pixels for our proposed model is very high. The proposed model results are obtained after pre-processing the input into JPEG format. The proposed model achieves an inference time of 985ms with a power consumption of 7.8W.

C. Ablation study

The performance of the model by skipping novel parts of the proposed training architecture is shown in Table 4 below. The results show increased accuracy (97.43%) when channel pruning is removed from training, as a higher number of input channels increases the model's performance. The same performance increase is observed for the model without kernel reparameterization (97.24%). The difference in the number of parameters between kernel reparameterization and channel pruning is very large, showing the importance of channel pruning. When compared to ConvNeXT-Tiny fine-tuned baseline models, our proposed model seems to perform effectively. The reason for the drop in accuracy is because of the trade-off we made in reducing the size and FLOP usage of the model so the minute drop in efficiency is justified. This reduction in accuracy can be due to the loss of many channels during channel pruning. The accuracy drops drastically when we go to the ADAM optimizer instead of the proposed attention-based loss function, showing the efficiency of our proposed loss function.

Table 4. Performance comparison of the proposed model in various training setups

Model	Channel pruning	Kernel Reparameterization	Attention-based loss function	Accuracy (%)	Params (M)	Inference time (ms)
Full Med-ConvNeXT	✓	✓	✓	96.53	12.96	4.1
Med-ConvNext without channel pruning	✗	✓	✓	97.43	28.58	8.7
Med-ConvNext without kernel reparameterization	✓	✗	✓	97.24	12.96	4.57
Without the proposed loss function	✓	✓	✗	95.5	12.96	4.1

V. Discussion

The proposed model achieves an accuracy percentage of 96.53%, macro sensitivity and specificity of 93.7% and 95.9%. These results indicate that the proposed model has fewer false positives and false negatives, which makes it a suitable candidate for image classification. The proposed pruning and kernel reparameterization maintained diagnostic capability

while reducing the parameter count, which can be seen in the macro-F1 score (86%) and Mean AUC (0.982) values. The main limitations found were lower macro precision (77.2%), indicating that classes with a lower number of training data suffer from high false positives. This is due to high data imbalance; a smaller change in false positives can have a high impact on macro precision. The classes with high training data such as

normal (Se=99.43%, Sp=88.32%), lung opacity (Se=96.00%, Sp=97.51%), aortic enlargement (Se=96.96%, Sp=97.49%), pleural thickening (Se=96.58%, Sp=96.17%) and pulmonary fibrosis (Se=96.73%, Sp=97.56%) have higher sensitivity and specificity values. While classes with limited training data and high annotation overlap possibly suffer in performance. Those classes include consolidation (Se=83.33%, Sp=96.64%), Nodule/Mass (Se=90.83%,

Wang et al. [4] presented a method based on a CNN on the ChestX-ray8 dataset and achieved an AUC of 0.8141. Rajpurkar et al. [3] suggested CheXNeXt to detect 14 thoracic diseases and achieved an AUC of 0.862 for atelectasis. Irvin et al. [5] proposed CheXpert, which classifies multiple conditions from chest X-rays using uncertainty labels, whereas Johnson et al. [19] suggested the MIMIC-CXR database. Kundu et al. [40] employed a combination of GoogLeNet, ResNet-18,

Table 5. Comparison of previous studies on chest X-ray-based disease detection

Author(s), Year	Method	Limitations	Output
Wang et al., 2017 [4]	CNN-based classification and localisation of ChestX-ray8	Issue with image level annotation labels; performance is variable	Detection of thoracic diseases; AUC of up to 0.8141
Rajpurkar et al., 2018 [3]	CheXNeXt with DenseNet to detect 14 different thoracic diseases	Limited data for evaluation; single institution data	Detection of diseases; AUC 0.862 for atelectasis
Irvin et al., 2019 [5]	CheXpert with Deep Learning with Uncertainty Labels	Multi-label setting; Not suitable for edge deployment	Multi-label classification of chest x-rays
Johnson et al., 2019 [19]	MIMIC-CXR dataset with chest radiographs and free text reports for automated analysis	Large dataset; labels are derived from radiology reports; image quality and positioning vary	377,110 chest x-rays of 227,835 studies for automated analysis of chest radiographs research
Kundu et al., 2021 [40]	GoogLeNet, ResNet-18, and DenseNet-121 Ensemble	Mainly centred on detection of pneumonia	Accuracy of 98.8

Sp=96.62%), Pneumothorax (Se=84.62%, [6],[11],[12],[13],[14],[15],[20] Sp=94.23%), and Atelectasis (Se=93.33%, Sp=96.22%) which shown lower classification performance. These classes have similar features compared to other findings, making the classification harder for novel classes. When compared with other edge-compatible models, the proposed Med-ConvNeXT model has classification efficiency with lower resource utilization. The dataset that is used for training (VinDr-CXR) is imbalanced, which results in lower precision (77.2%). The method doesn't consider noise in the dataset that is associated with real-time X-ray input acquired from different input sensors. This will result in lower accuracy during real-time implementation. Previous research has applied deep learning methods in automated chest X-ray disease diagnosis using various datasets and model structures. Such research reveals the capabilities of deep learning in disease detection but also shows some difficulties connected with data, labeling, and classification. A comparative analysis of such approaches is given in Table 5.

and DenseNet-121 and detected pneumonia with 98.8% accuracy. While these methods prove the efficacy of deep learning methods on chest X-rays, differences in datasets, tasks, and evaluation processes hinder direct comparison between results. Unlike other methods, Med-ConvNeXT aims not only for multi-class classification but also for computational and memory efficiency. The major limiting factor in our proposed method is dependence on the bounding box annotation used in training. This limits the implementation of the model on different datasets. The attention-based loss function doesn't work for global annotation methods. This training process can be trained only using the dataset with clear bounding-box annotations. [2],[5],[20].

VI. Conclusion

The proposed Med-ConvNeXT architecture, suitable for medical image classification, was specifically designed for edge-device applications. This is achieved by reducing the number of FLOPs and memory utilization, using techniques such as Kernel Fusion and Memory-Based channel pruning. The integration of

both techniques into the existing ConvNeXT architecture reduced the model parameter count by 54.7% compared to the ConvNeXT-Tiny model, a smaller model introduced earlier. The proposed method also reduces the total number of memory operations, making the model highly capable of integrating into edge devices. The proposed model was trained on 14-class labels annotated by trained medical professionals. The proposed model achieves an accuracy of 96.53%, with a mean AUC of 0.982 and a weighted F1 score of 91.91%. This performance was achieved with a reduction in parameter size, making the proposed Med-ConvNeXT model faster than other variants. The model achieved better performance when implemented on a Raspberry Pi device than many lightweight models like MobileNet and EfficientNet. The combination of high classification accuracy, high sensitivity, and fast inference makes the proposed model highly effective for image classification. The proposed attention-based channel-pruning method effectively eliminates unnecessary channels without degrading the classification model's performance. The kernel reparameterization method proposed in this study also reduces inference time and total FLOPs, thereby reducing the model's size while preserving its feature-extraction capability. Future work should focus on real-time implementation of the proposed model using raw input data.

The proposed model is very limited to training on well-annotated medical image datasets, and the performance of the model on a real-world novel dataset has not yet been validated, as the performance of the model on annotation-less data is very low. Therefore, in the future, the model should be trained on both types of datasets to make it effective for edge applications. The dataset that is used to train our model is not balanced, as some classes have very limited training data compared to other classes. This imbalance needs to be rectified for better model performance. The model should be considered a research prototype, and in future research, the model should be tested with real-time clinical data to evaluate its validity.

Declarations

Ethical Approval

This study used the publicly available VinDr-CXR chest X-ray dataset. All data were anonymised by the dataset providers before public release, and no direct involvement of human participants occurred in this research. As the study involved secondary analysis of a publicly available anonymized dataset with no direct participant interaction, additional institutional ethical approval was not required for this study.

Consent for Participation

Not Needed

Consent for Publications

Not Needed

Data Availability

The training and validation were done using the VinDr-CXR chest X-ray dataset, which is publicly available from the PhysioNet repository. To download the dataset, users are required to accept a Data Usage Agreement (DUA) called PhysioNet Credentialed Health Data License 1.5.0 (<https://physionet.org/content/vindr-cxr/view-license/1.0.0/>). By accepting the DUA, users agree that they will not share the data and that the dataset can be used for scientific research and educational purposes only and will not attempt to re-identify any patients, institutions, or hospitals.

Code Availability

Code repository not publicly available.

Declaration of Generative AI and AI-Assisted Technologies

The authors declare that no generative artificial intelligence or AI-assisted technologies were used in the preparation of this manuscript.

Author Contribution

Parvez Belim: Methodology, Software, Validation, Formal Analysis, Investigation, Resources, Data Curation, Writing – Original Draft, Writing – Review & Editing.

Nirav Bhatt: Conceptualization, Methodology, Software, Validation, Formal Analysis, Investigation, Resources, Data Curation, Writing – Original Draft, Writing – Review & Editing, Project Administration, Funding Acquisition.

Amit Lathigara: Conceptualization, Methodology, Investigation, Visualization, Supervision, Project Administration.

Vikas Kumar Sabata: Validation, Investigation, Formal Analysis, Writing – Review & Editing, Supervision.

All authors reviewed and approved the final version of the manuscript and agreed to be accountable for all aspects of the work.

Funding

This research was supported under the FIST program of the Department of Science and Technology (DST), Government of India, undertaken by the Department of Computer Engineering, RK University, Rajkot, under Grant No. **SR/FST/ET-I/2023/1282 (TPN: 90428)**.

Competing Interests

The authors declare no competing interests.

Acknowledgment

The authors sincerely thank the leadership of RK University for their continuous support and

encouragement throughout this research. Special thanks are extended to the Vice-Chancellor and the Director of Engineering for their valuable guidance and support in facilitating this study.

References

- [1] Vincent, Darian, G., & Surantha, N. (2025). Performance evaluation of convolutional neural network (CNN) for skin cancer detection on edge computing devices. *Applied Sciences*, 15(6), 3077. <https://doi.org/10.3390/app15063077>.
- [2] Nguyen, H. Q., Lam, K., Le, L. T., et al. (2022). VinDr-CXR: An open dataset of chest X-rays with radiologist's annotations. *Scientific Data*, 9, 429. <https://doi.org/10.1038/s41597-022-01498-w>.
- [3] Rajpurkar, P., Irvin, J., Zhu, K., et al. (2018). Deep learning for chest radiograph diagnosis: A retrospective comparison of the CheXNeXt algorithm to practicing radiologists. *PLoS Medicine*, 15(11), e1002686. <https://doi.org/10.1371/journal.pmed.1002686>.
- [4] Wang, X., Peng, Y., Lu, L., et al. (2017). ChestX-ray8: Hospital-scale chest X-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2097–2106. <https://doi.org/10.1109/CVPR.2017.369>.
- [5] Irvin, J., Rajpurkar, P., Ko, M., et al. (2019). CheXpert: A large chest radiograph dataset with uncertainty labels and expert comparison. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01), 590–597. <https://doi.org/10.1609/aaai.v33i01.3301590>.
- [6] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A ConvNet for the 2020s," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2022, pp. 11976–11986, doi: 10.1109/CVPR52688.2022.01167
- [7] Dosovitskiy, A., Beyer, L., Kolesnikov, A., et al. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. *International Conference on Learning Representations (ICLR)*. <https://doi.org/10.48550/arXiv.2010.11929>.
- [8] Liu, Z., Lin, Y., Cao, Y., et al. (2021). SwinTransformer: Hierarchical vision transformer using shifted windows. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 10012–10022. <https://doi.org/10.1109/ICCV48922.2021.00986>.
- [9] Ding, X., Guo, Y., Ding, G., & Han, J. (2019). ACNet: Strengthening the Kernel Skeletons for Powerful CNN via Asymmetric Convolution Blocks. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 1911–1920. <https://doi.org/10.1109/iccv.2019.00200>.
- [10] Ding, X., Zhang, X., Ma, N., Han, J., Ding, G., & Sun, J. (2021). RepVGG: Making VGG-Style ConvNets Great Again. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 13733–13742. <https://doi.org/10.1109/CVPR46437.2021.01352>.
- [11] Chollet, F. (2017). Xception: Deep Learning With Depthwise Separable Convolutions. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1800–1807. <https://doi.org/10.1109/CVPR.2017.195>.
- [12] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). MobileNetV2: Inverted residuals and linear bottlenecks. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 4510–4520. <https://doi.org/10.1109/CVPR.2018.00474>.
- [13] Tan, M., & Le, Q. V. (2019). EfficientNet: Rethinking model scaling for convolutional neural networks. *Proceedings of the 36th International Conference on Machine Learning*, 97, 6105–6114. <https://doi.org/10.48550/arXiv.1905.11946>.
- [14] Huang, G., Liu, Z., van der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 4700–4708. <https://doi.org/10.1109/CVPR.2017.243>.
- [15] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778. <https://doi.org/10.1109/CVPR.2016.90>.
- [16] Todi, A., Narula, N., Sharma, M., & Gupta, U. (2023). ConvNext: A Contemporary Architecture for Convolutional Neural Networks for Image Classification. *Proceedings of the 2023 3rd International Conference on Innovative Sustainable Computational Technologies (CISCT)*, 1–6. <https://doi.org/10.1109/CISCT57197.2023.10351320>.
- [17] Abdusalomov, A., Mirzakhilov, S., Umirzakova, S., Buriboev, A. S., Meliboev, A., Muminov, B., & Jeon, H. S. (2025). Accessible AI diagnostics and lightweight brain tumor detection on medical

- edge devices. *Bioengineering*, 12(1), 62. <https://doi.org/10.3390/bioengineering12010062>
- [18] Novac, P., Hacene, G. B., Pegatoquet, A., Miramond, B., & Gripon, V. (2021). Quantization and deployment of deep neural networks on microcontrollers. *Sensors*, 21(9), 2984. <https://doi.org/10.3390/s21092984>.
- [19] Johnson, A. E. W., Pollard, T. J., Berkowitz, S. J., et al. (2019). MIMIC-CXR: A de-identified publicly available database of chest radiographs with free-text reports. *Scientific Data*, 6, 317. <https://doi.org/10.1038/s41597-019-0322-0>.
- [20] Jacob, B., Kligys, S., Chen, B., et al. (2018). Quantization and training of neural networks for efficient integer-arithmetic-only inference. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2704–2713. <https://doi.org/10.1109/CVPR.2018.00286>.
- [21] Wang, C., Wang, X., Zhai, J., & Shao, S. (2023). A Lightweight convolutional medical segmentation algorithm based on ConvNeXt to improve UNet. *Journal of Intelligent & Fuzzy Systems*, 46(2), 4077–4092. <https://doi.org/10.3233/jifs-233232>.
- [22] Yu, W., Zhou, P., Yan, S., & Wang, X. (2024). InceptionNeXt: When Inception Meets ConvNeXt. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 5672–5683. <https://doi.org/10.1109/CVPR52733.2024.00542>.
- [23] Wang, A., Chen, H., Lin, Z., Han, J., & Ding, G. (2024). RepViT: Revisiting Mobile CNN From ViT Perspective. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 15909–15920. <https://doi.org/10.1109/CVPR52733.2024.01506>.
- [24] He, Y., Zhang, X., & Sun, J. (2017). Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE international conference on computer vision*, (pp. 1389-1397). <https://doi.org/10.1109/ICCV.2017.155>
- [25] Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., & Zhang, C. (2017). Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (pp. 2736-2744). <https://doi.org/10.1109/ICCV.2017.298>
- [26] Fang, G., Ma, X., Song, M., Mi, M. B., & Wang, X. (2023). DepGraph: Towards Any Structural Pruning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 16091–16101. <https://doi.org/10.1109/CVPR52729.2023.01544>
- [27] Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-Excitation Networks. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 7132–7141. <https://doi.org/10.1109/CVPR.2018.00745>.
- [28] Sevinc, A., Ucan, M., & Kaya, B. (2025). A Distillation Approach to Transformer-Based Medical Image Classification with Limited Data. *Diagnostics*, 15(7), 929. <https://doi.org/10.3390/diagnostics15070929>.
- [29] Korkmaz, O. E. (2025). Revisiting convolutional design for efficient CNN architectures in edge-aware applications. *Scientific Reports*, 15, 44351. <https://doi.org/10.1038/s41598-025-27856-3>
- [30] Selvaraju, R. R., Cogswell, M., Das, A., et al. (2017). Grad-CAM: Visual explanations from deep networks via gradient-based localization. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 618–626. <https://doi.org/10.1109/ICCV.2017.74>.
- [31] Zeiler, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. *European Conference on Computer Vision (ECCV)*, 818–833. https://doi.org/10.1007/978-3-319-10590-1_53.
- [32] Ding, X., Zhang, X., Han, J., & Ding, G. (2021). Diverse Branch Block: Building a Convolution as an Inception-like Unit. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 10886–10895. <https://doi.org/10.1109/CVPR46437.2021.01074>.
- [33] Vasu, P. K. A., Gabriel, J., Zhu, J., Tuzel, O., & Ranjan, A. (2023). FastViT: A Fast Hybrid Vision Transformer Using Structural Reparameterization. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 5785–5795. <https://doi.org/10.1109/ICCV51070.2023.00532>.
- [34] Ding, X., Zhang, X., Han, J., & Ding, G. (2022). Scaling Up Your Kernels to 31x31: Revisiting Large Kernel Design in CNNs. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 11963–11975. <https://doi.org/10.1109/CVPR52688.2022.01166>.
- [35] Li, H., Kadav, A., Durdanovic, I., Samet, H., & Graf, H. P. (2017). Pruning filters for efficient ConvNets. *International Conference on Learning Representations (ICLR)*.

<https://doi.org/10.48550/arXiv.1608.08710>.

- [36] Luo, J.-H., Wu, J., & Lin, W. (2017). ThiNet: A filter level pruning method for deep neural network compression. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 5068–5076. <https://doi.org/10.1109/ICCV.2017.541>.
- [37] Yi, M., Zhang, H., Chen, W., Ma, Z.-M., & Liu, T.-Y. (2019). BN-invariant sharpness regularizes the training model to better generalization. *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI)*, 4164–4170. <https://doi.org/10.24963/ijcai.2019/578>.
- [38] Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. *International Conference on Learning Representations (ICLR)*. <https://doi.org/10.48550/arXiv.1412.6980>.
- [39] Kendall, A., Gal, Y., & Cipolla, R. (2018). Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 7482–7491. <https://doi.org/10.1109/CVPR.2018.00781>.
- [40] Kundu, R., Das, R., Geem, Z. W., Han, G., & Sarkar, R. (2021). Pneumonia detection in chest X-ray images using an ensemble of deep learning models. *PLoS ONE*, 16(9), e0256630. <https://doi.org/10.1371/journal.pone.0256630>.

Author Biography

Parvez Belim got his Master of Computer Applications (MCA) degree from Gujarat Technological University, India in 2012. Currently, he is a Research Scholar at the Department of Computer Engineering, RK University, Rajkot, Gujarat. His work emphasizes the creation and services of intelligent computational models, mainly



in the areas of data mining, deep learning, and machine learning. Parvez is an energetic researcher and is involved in various academic activities. He has the privilege to publish his peer-reviewed work at top events such as the Artificial Intelligence in Machine Vision (AIMV) conference going to be held in 2025 and the International Conference on Transformations in Engineering Education (ICTIEE) slated for 2024. Apart from this, he loves the computer science field and shares the same interest through various academic development-related activities and is committed to

bringing about the change. He can be contacted at email: belimparvez1989@gmail.com

Nirav Bhatt, Dr. Nirav Bhatt, Professor of Computer



Science and Engineering, School of Engineering, RK University. Over the course of his 18-year career in technical education, he has contributed to teaching, research, and institutional development. His research work encompasses various latest emerging areas including publications in reputed journals. His efforts have been acknowledged through various competitive research grants, including government-supported funding aimed at modernizing technical education. He plays an influential role in shaping academic standards by serving as a member of the editorial boards, review panels, technical committees, etc.



Amit Lathigara Dr. Amit

Lathigara is a distinguished academician and Dean, Faculty of Technology at RK University, with over 20+ years of experience in higher education. He has made significant contributions to research, with high-impact publications and extensive guidance in his field. As an expert, he has represented RK University in numerous national and international academic and research workshops. He has received various national and international research grants. He has received multiple awards, including the Faculty Fellow Award in 2018, and holds an ING.PAED. IGIP certification as an international engineering educator.



Dr. Vikas Kumar Sabata is

associated with Kalinga Institute of Industrial Technology (KIIT) Deemed to be University, Bhubaneswar, Odisha, India, where he serves as Joint Director – International Relations. He is actively involved in strengthening KIIT's international collaborations, academic partnerships, and global engagement initiatives. His responsibilities include supporting international academic relations, facilitating collaborations with universities and institutions across the world, and contributing to initiatives for international students and global academic mobility. He has represented KIIT in various national and international academic

engagements and institutional collaborations. His work focuses on promoting internationalization, fostering academic partnerships, and enhancing global opportunities for students and the university.