Signal, Image and Video Processing
https://doi.org/10.1007/s11760-024-03379-8

ORIGINAL PAPER

Binarydnet53: a lightweight binarized CNN for monkeypox virus image classification

Debojyoti Biswas[1] · Jelena Tešić[1]

Received: 25 May 2024 / Revised: 6 June 2024 / Accepted: 12 June 2024

Abstract
The recent widespread increase of the Mpox (formerly monkeypox) virus infections in South Asian and African countries has raised concerns among medical professionals regarding the potential emergence of another pandemic in those regions. According to the World Health Organization (WHO) "emergency meeting" on May 20, 2022, there were 82,809 confirmed cases reported in 110 countries. With the number of available test kits surpassing the count of positive/probable cases, there is a pressing need to develop a robust and lightweight classifier model that can alleviate the burden of physical testing kits and expedite the detection process. The existing state-of-the-art primarily focuses on achieving high accuracy in modeling Mpox without considering factors such as modeling suitability, real-time inferencing, and adaptability to resource-constrained CPU-only mobile devices. In this research, we propose a novel lightweight binarized DarkNet53 model, referred to as BinaryDNet53, which is approximately $\sim 20\times$ more computationally efficient and $\sim 2\times$ more power-efficient than the current state-of-the-art. This model demonstrates smooth detection capabilities when deployed on small hand-held or embedded devices. Firstly, we binarize the weights and biases of the DarkNet53 model to prevent high computational costs and memory usage. Next, our work introduces large-margin feature learning and weighted loss calculation to enhance results, particularly on complex samples. We conduct experiments using the latest MSLD v2.0 dataset, showcasing the superiority of the proposed model over state-of-the-art models based on classification and computational metrics, including Watt power consumption, required memory, and GFLOPS.

Keywords Monkeypox classification · Lightweight CNN · Binarized CNN · Power efficient DNN · Deep learning approaches to detect Mpox

1 Introduction

Recent advancements in artificial intelligence (AI) have solved many challenging computer vision problems, ranging from image classification [1], segmentation [2], object detection [3, 4], etc., in consumer, medical, and remote-sensing images. There has been significant improvement in medical science due to AI-powered devices and utilities. Deep Learning is a sub-field of artificial intelligence that can assist us in extracting important features/information from images without human participation. In the field of medical science, different kinds of imaging techniques are primarily used to assist medical professionals and clinics in making diagnoses of a wide variety of diseases, such as brain cancer [5], respiratory conditions like pneumonia [6], as well as the recent COVID-19 virus [7]. Although the infection of COVID-19 is not entirely over yet, the Mpox virus has caused concern in many parts of the world. Mpox (formerly Mpox) is announced as a medical emergency [8] due to its rapid outbreak across 40 countries. Furthermore, the Centers for Disease Control (CDC) in the United States raised the Monkeypox travel alert level to "Level 2" on June 6, 2022 [9] due to its significant outbreak. With the number of available physical test kits surpassing the count of probable cases [10], there is an urgent need to start building lightweight and computationally efficient detection models. A deep learning-based model can help us address problems such as the inadequate number of clinic experts, the high cost of physical kits, and,

✉ Debojyoti Biswas
debojyoti_biswas@txstate.edu

Jelena Tešić
jtesic@txstate.edu

1 Department of Computer Science, Texas State University, 601 University Dr, San Marcos, TX 78666, USA

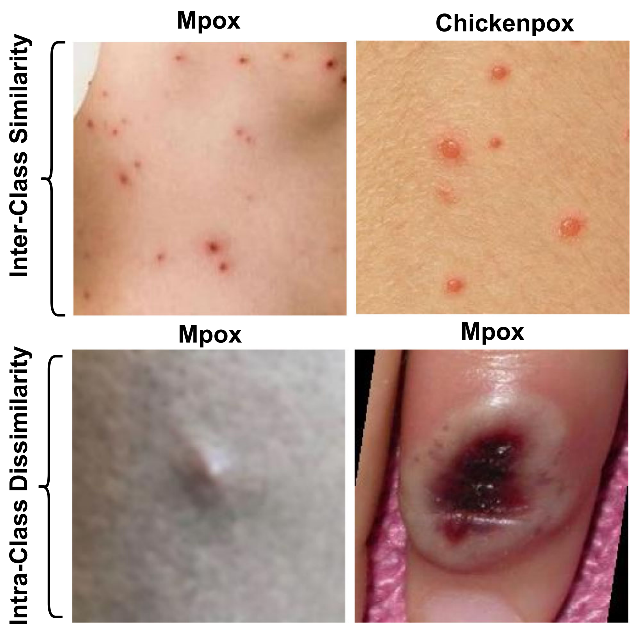


Fig. 1 Example of Sample images of each virus-infected class collected from the MSLD v2.0 dataset

most importantly, the prolonged waiting time involved in the virus detection process.

Recently, Kundu et al. used a vision transformer-based classification model for Mpox classification [1]. This work aimed to evaluate several machine learning and deep learning models toward achieving optimal results. The optimal result in that work came from the transformer and ResNet50 models. However, ResNet50 and Transformer models are heavy in computation, with the ResNet50 requiring around 86.47 Giga Floating-point Instructions Per Second (GFLOPS) and 25.6 Million learnable parameters, making it a poor choice for mobile devices. On the other hand, Islam et al. [11] also used pre-trained ResNet50 and several other feature extractors for Mpox classification. However, they did not perform hard-example mining for samples near the decision boundary. The model is not suitable for datasets with intra-class dissimilarity and inter-class similarity issues (see Fig. 1). Moreover, none of the existing work pays attention to the power and memory usage of the deployed model, which is very crucial for small embedded devices for uninterrupted services.

Contribution

In this paper, we propose a lightweight and computationally efficient model for real-time Mpox detection. The model's low power consumption and faster detection rate make it a strong candidate for real-time virus detection in mobile devices. First, we propose to use a large-margin hybrid network with Maximum margin hyperplane learning to distinguish Mpox from Chickenpox correctly. Second, we propose using the feature vector's saliency to calculate the image's complexity level. It has been shown that the amount of neuron activation is a good indicator of pixel diversity and texture variance [12]. We propose to use this information to calculate the image complexity and put more weight on more complex cases in the loss function. Third, we binarize all weights and biases of the DarkNet53 baseline model, as a CNN with binary weights and biases is $\sim 20\times$ more *computationally efficient* than an equivalent network with single-precision weights. Thus, all calculations use $+, -$ operators only during the forward pass, significantly reducing the computations. The paper's contributions are:

1. Novel framework: We propose a large-margin-based hybrid network to address the high inter-class similarity.

2. Saliency-aware Weighting: In this process, we put more weight on complex sample learning and less on easy examples. The complexity of the sample is calculated from the number of neuron activations. The experiments show that weighted loss calculation significantly improves the actual positive virus detection performance.

3. Binarization of Weights and Biases Inspired by the work of XNOR-Net [13], we convert the traditional DarkNet53 model as a binary CNN model. The Binary version of the DarkNet53 model requires only 8.82 GFLOPS and 0.087 Million learnable parameters, which is significantly less than traditional DenseNet201 and ResNet50 CNN models.

4. Comprehensive Model Analysis Apart from the classification matrics (Accuracy, Precision, and Recall), we include several other important evaluation criteria such as the amount of power (in Watt) needed for the model training and inference, the amount of GFLOPS required for the model computation, and the lastly the size of the model in system memory.

The rest of this article is organized as follows. Section 2 summarizes related work; Sect. 3 introduces the proposed Binary-DarkNet53 method and the pipeline. Section 4 introduces the experimental setup and dataset. Section 5 evaluates the proposed framework, comparing the latest SOTA methods for classification and computational performances. Finally, Sect. 6 summarizes the quantitative findings and outlines future works.

2 Related Work

Convolutional Neural Networks (CNN) have been playing a key role in extracting rich features from different types of images. Some deep CNNs like ResNets [14], DarkNet53 [3], and MobileNetv2 [15] showed promising results on different computer vision tasks. The success of CNN is also evident in the field of medical image classification [6, 11, 16].

The initial use of CNNs in medical science was to classify simple tasks, such as brain tumors or pneumonia, from X-ray and MRI reports. In early 2007, researchers often used the traditional Machine Learning approach for tumor classification. S. Chaplot et al. [17] proposed a novel technique for classifying magnetic resource images of the human brain, which utilizes wavelets to support vector machine and neural system self-organizing maps. Later, Usman et al. [18] proposed a similar wavelet-based multimodal classification of MRI images. Their work extracted feature intensity, intensity differences, local neighborhood, and wavelet texture from multimodal data. However, after the improvement of deep learning networks, CNN became the standard feature extraction method for image data. Afshar et al. [19] developed a capsule network (CapsNet) model to classify brain tumors on the Figshare dataset efficiently. In this work, authors improve the performance by using the context information through the spatial relationships between the tumor and its surrounding tissues. On the other hand, Abiwinanda et al. [20] generated seven distinct CNN variations without segmentation for brain tumor classification and achieved the optimal training and testing accuracy of 98.51% and 84.19% on the Fighsare brain MRI dataset, respectively.

A recent successful application of CNN in medical science is the early detection of COVID-19 patients from Chest X-rays. As the number of Reverse-transcription polymerase chain reaction (RT-PCR) testing kits was in limited supply, it became urgent to develop a rapid screening method that is accurate, fast, and cheap. Toward this goal, Abbas et al. [7] utilized a CNN model known as DeTraC to classify COVID-19 using chest X-ray images. On the other hand, Narin et al. [21] integrated two deeper models for organizing COVID-19 and standard CXR images obtained from the public domain datasets. Moreover, to gain more improved results, Wang et al. [22] proposed an inception migration-learning model for the classification of 453 confirmed cases of COVID-19 with previously diagnosed traditional pneumonia.

The main goal of developing the CNN model for medical image classification is to replace the physical testing kits with a rapid screening DNN model. To deploy the rapid screening DNN model in mobile or embedded devices, it needs to be lightweight and require less power and memory. Toward this goal, several studies have been conducted on developing lightweight classification and segmentation methods for medical image datasets. Gupta et al. [23] propose a novel lightweight deep learning-based model, "Reduced-FireNet," for auto-classification of histopathological images. The goal of the work is to make the classification model suitable for Internet of Medical Things (IoMT) technologies. The proposed method has a very low computational and memory requirement of 0.201 GFLOPS and 0.391 MB, respectively. Deng et al. [24] propose a lightweight U-Net (ELU-Net) with deep skip connections for efficient medical image segmentation. On the other hand, Tuncer et al. [25] propose a novel lightweight CNN called TurkerNet for

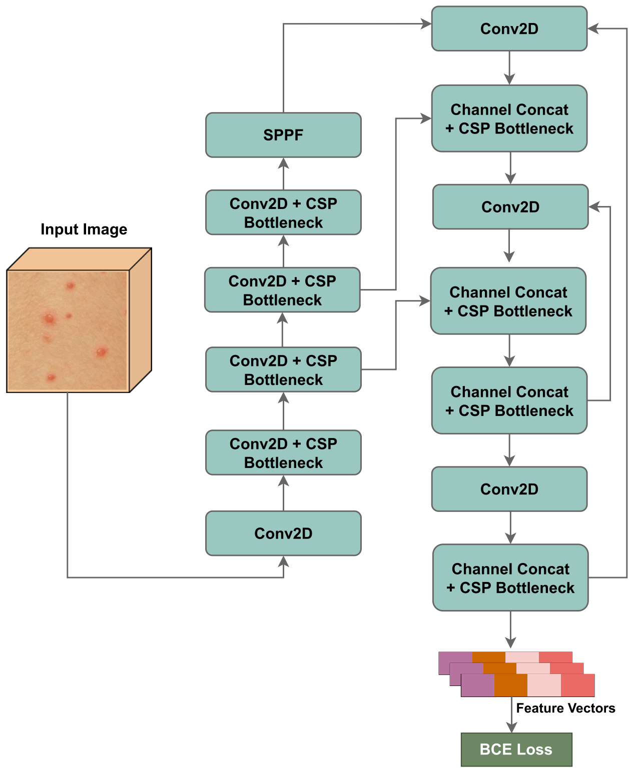


Fig. 2 The architecture of baseline CSP-DarkNet53 Mpox Classifier

skin cancer detection. The goal is to minimize the learnable parameter as much as possible without hurting the classification accuracy. However, none of the work utilized the idea of binarized CNN through binary weights and bias for medical imagery. Furthermore, the above-mentioned works do not report the power consumption requirement for the model inference, which is very crucial for embedded systems.

Due to the success of CNNs in earlier applications, it was the first choice for researchers to perform Mpox classification. Ali et al. [16] used several pre-trained CNN feature extractors such as ResNet50, InceptionV3, and VGG-16 models for the classification of Mpox and other similar diseases on the Mpox Skin Lesion Dataset (MSLD). Like the previous work, Islam et al. [11] also used pre-trained ResNet50, DenseNet121, MobileNet-V2, and several other models to examine the classification performance on Mpox datasets. The latest dataset released by Ahsan et al. [26] caught the researcher's attention due to its completeness covering diverse virus-infected samples. In the work, Ahsan et al. proposed a modified VGG16 model, which achieved an accuracy of 97 ± 1.8% and 88 ± 0.8% for studies one and two, respectively. Finally, Irmak et al. [27] use pre-trained CNN networks, like MobileNetV2 and VGG models, to categorize Mpox infection on the Mpox Skin Image Dataset (MSID) dataset. The optimal result was achieved from the MobileNetv2 model with an accuracy of 91.28%. Bala et

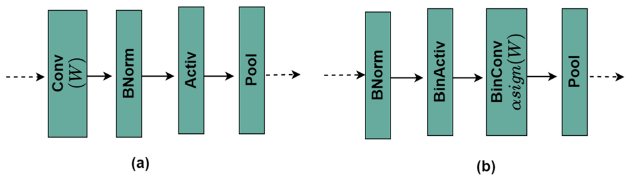


Fig. 3 Here (a) shows a traditional CNN block and (b) shows a Binary weight CNN block

al. propose MonkeyNet [28] with DenseNet201 detector, achieving 93.19% of accuracy on the original test images. However, the computational memory needed for running inference and the energy/ power requirement for deploying the device were not reported. In summary, state-of-the-art relies on only pre-trained CNN networks for classification. They fail to address complex samples where inter-class similarity is an issue. Also, none of these works discusses the feasibility of deploying the models in a mobile device where low-power consumption is in demand.

3 Methodology

The success of deep learning algorithms lies in the fact that they can automatically extract hidden and complex features efficiently from images and videos, which is not often possible by technicians or experts. Several Convolutional Neural Network (CNN) architectures have been proposed for feature extraction, e.g., AlexNet, ResNet, MobileNet, DarkNet, etc. The advances in software and hardware support enabled deep neural network-driven solutions for breast and brain cancer [5], pre-COVID pneumonia detection [6], and efficient automated creation of the MRI report [18]. It is important to note that pixel-level semantic information plays a critical role in medical and healthcare systems because subtle changes in pixel color can change the final decision (see Fig. 1). Keeping that in mind, we carefully design a framework that preserves low-level semantic information efficiently and achieves optimal performance. In the following subsection, we will describe the baseline model we used for our proposed model.

3.1 Baseline Architecture

We propose to use DarkNet53 with the Cross Stage Partial (CSP) module for its proven performance in YOLOv3 architecture. CSP DarkNet53 [3] can detect small pixel information in more detail while preserving spatial information in later layers of CNN [12]. Our baseline architecture is illustrated in Fig. 2, where we have three main parts: 1) Input, 2) Feature Extractor Backbone, and 3) Binary Loss Function. As input, we send original and augmented versions of the Mpox sample from the MSLD v2.0 dataset. The actual size of the data samples is $\sim 224 \times 224$; we downsize the samples to 124×124 pixels before we send the samples to the feature extractor. Next, we perform image normalization using Eq. 1.

$$\text{Normalized Image} = \frac{X - \min(X)}{\max(X) - \min(X)} \tag{1}$$

CSP DarkNet53 backbone starts with a 2D Convolution (Conv2D) operations followed by CSP [29] module blocks (Conv2d+CSP Bottleneck). Adding CSP blocks in DarkNet53 architecture aims to represent complex features effectively by enabling richer gradient combinations with low computational costs. The architecture of the CSP block is illustrated in Fig. 5. Here, we see that the feature map from the base layers gets split into two branches through the channel of the base feature. The former part of the channel blocks is directly connected to the partial transition block; however, the latter part goes through a dense block (such as Residual blocks with bottleneck). Inside the transition block, we first perform some transition on the Residual block output. Then, we concatenate the transition output with the part 1 block output. Finally, another transition is performed on the combined feature and passes to the next block in the architecture.

We have four Conv2D+CSP blocks, one after one, in the bottom-up baseline architecture. Then, we have a Spatial Pyramid Pooling Fast (SPPF) layer following CSP blocks. The SPPF aims to improve the calculation speed by using a single pooling layer instead of multiple pooling layers in the SPP module. After SPPF blocks, we start the top-down architecture by concatenating later layers of CSP blocks with the earlier layers of CSP blocks. The concatenation helps us to retain semantic and spatial features in the final feature vectors.

The output from the last layer of the CSP Bottleneck block is flattened and fed to a fully connected layer to reduce the feature dimension to 1024. The final 1024 dimension feature vector is fed into the sigmoid function, where we predict the class probability from 0 to 1. Finally, the probability scores and the ground truth class labels are provided for the Binary Cross-Entropy (BCE) loss function for calculating the training loss. The Eq. 2 shows the calculation for BCE loss, where y is the true label (0 or 1) and $\hat{y}$ is the predicted probability.

$$\text{BCE}\,(y, \hat{y}) = -\big(y \cdot \log(\hat{y}) + (1 - y) \cdot \log(1 - \hat{y})\big) \tag{2}$$

3.2 Proposed Architecture

Our goal is to outperform State-of-the-art classification performance with as low computational and power consumption as possible for Mpox with proposed lightweight CNN architecture with binarized weights and bias, as illustrated in Fig. 6. ResNets, DenseNets, or DarkNets work fine for classification tasks, but they are computationally heavy and not

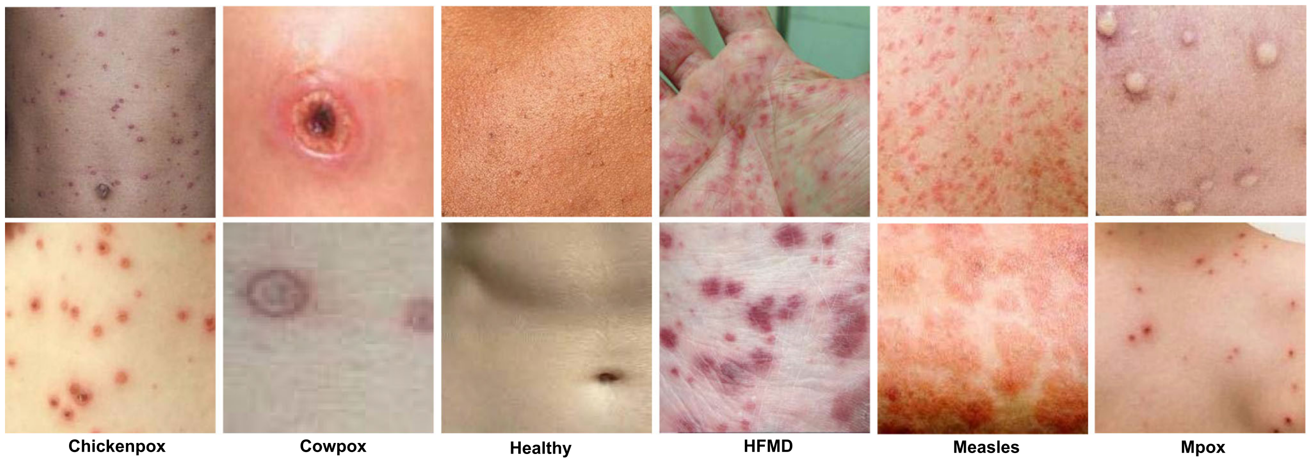


Fig. 4 Example of Sample images of each virus-infected class collected from the MSLD v2.0 dataset

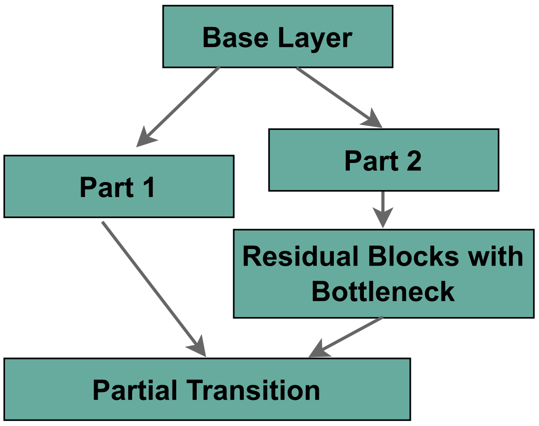


Fig. 5 Cross Stage Partial Network

suiTable for resource-constrained devices [30]. The proposed architecture offers five significant improvements: 1) Binarized weights and bias, 2) Reduction of CSP Blocks, 3) Feature Fusion, 4) Image Complexity Score, and 5) Large Margin SVM classifier.

1. Binary Weights and Bias: Inspired by the work of Rastegari et al. [13], we incorporate binary weight and bias for our baseline CSP DarkNet53 model. As illustrated in Fig. 3(b), we need a binary filter and a scaling factor α to perform binary convolution. If I is the input image, and W is the convolution weight, we can express the convolution operation using the "$(I, W, *)$" expression. A binary filter $B \in \{-1, +1\}^{c \times w \times h}$ and a scaling factor $\alpha \in \mathbb{R}^{+}$ is used to calculate the real-value weight matrix (W) with the following formula, $W \approx \alpha B$. The binary convolution operation is outlined in Eq. 3 using all defined terms.

$$I * W \approx (I \oplus B)\alpha \tag{3}$$

In Eq. 3, the $\oplus$ operation is a convolution operation where no multiplication operation is needed; we rely only on addition and subtraction. It is well-known that multiplication is one of the most computationally expensive operations for calculation. Removing the multiplication operation significantly reduces the computational cost for CNN blocks. Figure 3 shows another term, $B* \approx Sign(W)$, where $B*$ is the weight matrix for binary filter B. We can estimate the binary weight filter by applying logical conditions on weight(W) in Eq. 4.

$$B* \approx sign(W) = \begin{cases} B_i = +1, & \text{if } w_i \geq 0 \\ B_i = -1, & \text{otherwise} \end{cases} \tag{4}$$

2. Reduction of CSP Block: We removed the CSP blocks from the deeper part of the model, focusing on generating high-level features. The high-level feature represents the abstract view of the object at different scales. For healthcare image samples, the challenge is to preserve the pixel-level features because the disease symptoms heavily depend on the color, texture, and shapes. These critical features are mainly generated by CNN's surface layers or low-level features. From Fig. 4, we can see very little diversity at the abstract level, so extracting features at a less abstract level should show optimal performance compared to the baseline model. We also removed several residual connections; it is evident that the scale variation is not very high for Mpox-infected areas in the samples, so it is not very useful to extract multi-scale feature extraction.

3. Feature Fusion helps us to focus more on high-intensity foreground objects and reject the background noise from non-infected skin areas. To perform feature fusion, we first pass CSP block outputs to the Conv2D block to make the feature dimension consistent. Next, we use FC layers with ReLU activation to make the length of each feature the same. We use element-wise multiplication as the feature fusion technique. The fusion operation is $F'_{NX1024} = F1_{NX1024} * F2_{NX1024}$:

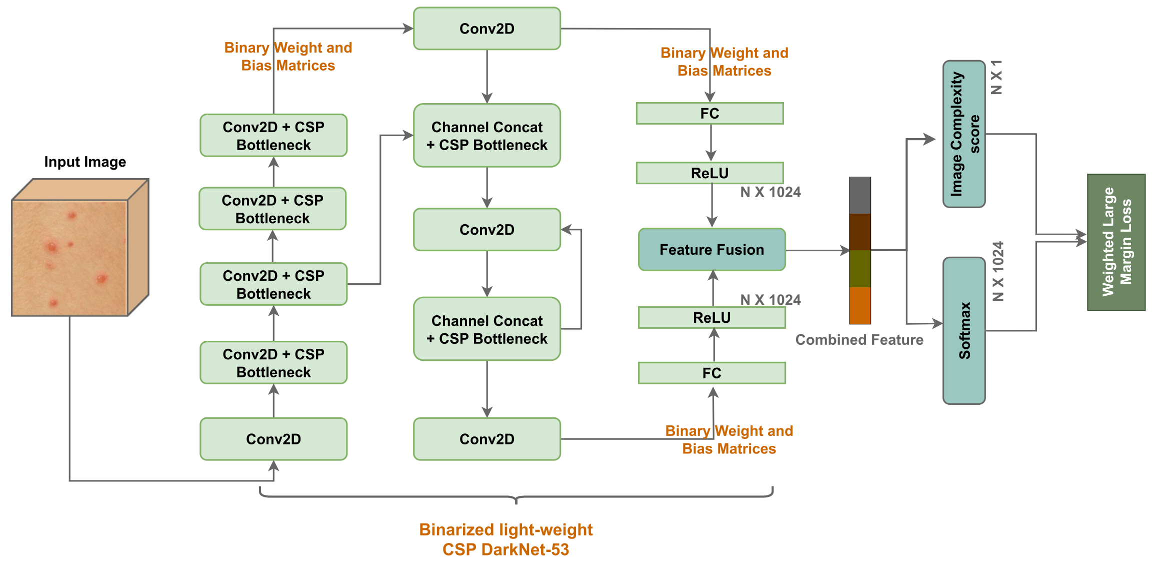


Fig. 6 The proposed BinaryDNet53, a Large-Margin Saliency-aware Binarized Mpox Classifier. Here, the light green blocks represent Binary CNN blocks

F1 and F2 are two outputs from FC layers, $*$ is element-wise multiplication, and F' denotes the output combined feature.

4. Image Complexity Score (ICS): Fig. 1 and Fig. 4 illustrate that Mpox image samples vary in complexity, as the number of infected areas captured in the image varies significantly. The infected area can look very similar to the skin color, making it hard to differentiate the foreground infected area from the skin. We propose to calculate the saliency-aware image complexity score, as introduced in [12, 31], to estimate the complexity of the image. Equation 5 estimates I_{sum}, the sum of all pixel intensity from the image feature vector. Next, to avoid the issue of gradient explosion, we restrict the value of the total image activation within a stable range. The total operation is denoted with ICS in Eq. 5. The ICS for each image is later used in the Large-Margin Loss calculation as a weight for the corresponding image. Here, l_b and u_b are the lower and upper bounds of the ICS value.

$$I_{sum} = \sum_{dim=1}^{D} f_{dim}(I) \Rightarrow$$
$$ICS(I_{sum}) = min(max(I_{sum}, l_b), u_b) \tag{5}$$

5. Weighted Large-Margin Loss is used instead of binary cross-entropy loss for calculating classification error. Figure 1 shows that inter-class similarity exists in our dataset. Many data points lie very close to the decision boundary, and there is a high chance of getting misclassified with a low-margin decision boundary. As shown in Fig. 7, margin-based SVM hinge loss can help us to create large margins near the decision boundary. Figure 7 shows the mechanism for the SVM margin where the samples with a probability near 0.5 are at risk of misclassification. So, the SVM Margin creates space around it to accommodate some errors and keep the margin high for overall improved performance. The loss function can be defined as follows in Eq. 6, where C is the number of classes, N is the mini-Batch Size, P is the Penalty parameter, Δ is the margin parameter, and y is the index $(0 \leq y \leq C-1)$ of the true class label for a sample.

$$M\_Loss_{x,y} = \frac{\sum_{i=0}^{C-1} \max(0, \Delta - x[y] + x[i])^P}{N} \tag{6}$$

Here, x is the 2D logits/probability scores vectors derived from the softmax function. So, the dimension of the vector x is $N \times C$ and $i \neq y$. The above Eq. 6 shows the operation for a single example in a mini-batch of size N.

$$WM\_Loss_i = ICS_i \times M\_Loss_i \tag{7}$$

Next, we multiply the loss for each image w.r.t the corresponding Image Complexity Score ICS for the weighted large-margin loss. The final weighted loss becomes as in Eq. 7, where i is the i^{th} example in the mini-batch.

4 Proof Of Concept Setup

4.1 Dataset

In our experiment, we choose the latest publicly open Mpox dataset known as Mpox Skin Lesion Dataset Version 2.0 (MSLD v2.0) [32, 33]. This dataset is the extended version of MSLD 1.0; in the earlier version, there were only two classes: Mpox and Others. The latest version has six classes: Mpox,

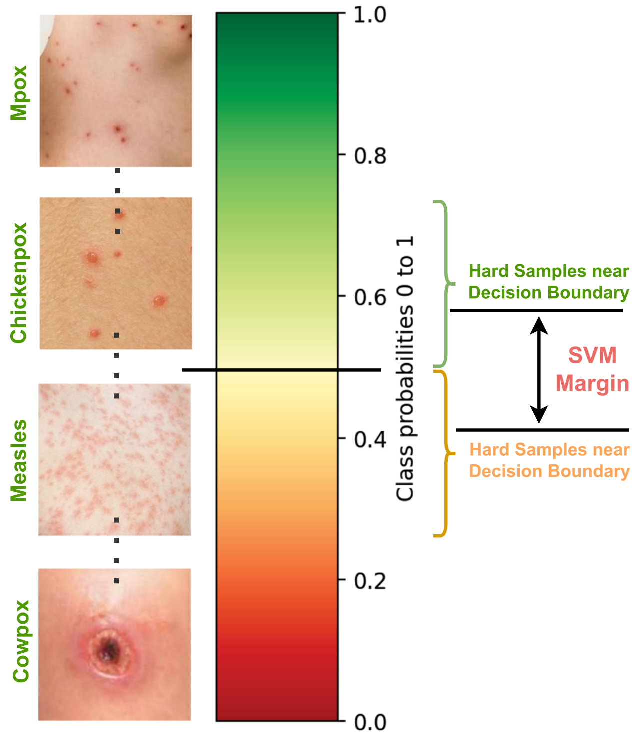

Mpox
Chickenpox
Measles
Cowpox
1.0
0.8
0.6
0.4
0.2
0.0
Class probabilities 0 to 1
Hard Samples near Decision Boundary
SVM Margin
Hard Samples near Decision Boundary

Fig. 7 The Large Margin mechanism for hard sample mining using SVM Classifier

Chickenpox, Measles, Cowpox, Hand-foot-mouth disease, and Healthy. Figure 4 illustrates a few samples from each class. Different augmentation techniques, such as rotation, translation, reflection, brightness jitter, noise, etc., were performed on the original dataset to increase the size of the dataset. The dataset is a suitable candidate for SOTA benchmarking because of its class variety and distribution. The class distribution of the original and augmented dataset is presented in Table 1. The dataset also generalizes to most of the medical image datasets with its inter-class similarity and intra-class dissimilarity challenges (See Fig. 1). Also, the six classes available in the dataset provide diverse feature characteristics that share feature properties with other skin diseases. Our goal is to make a robust and generalized model for medical image classification. To achieve this goal, it is crucial to select a dataset with as much data variety as possible. However, the real samples for infected skins in the dataset are limited. Hence, improper model training without any regularization may lead to overfitting issues. The dataset folder includes images (Train, Test, and Validation) with a proportion of 70:10:20. Each of the Train, Test, and Validation folders was divided into six folders containing six different classes. To make the train-val dataset, we first merged training and validation sets. Then, we followed 75:25 random splitting for training and validation set creation. We used the 20% test set given with the MSLD v2.0 dataset for testing.

Table 1 Instance distribution statistics of the presented Mpox Skin Lesion Dataset (MSLD) v2.0

Infection Class	# of Original Images	# of Augmented Images	# of Unique Patients
Chickenpox	75	3598	62
Cowpox	66	3220	41
Healthy	114	5656	104
HFMD	161	7882	144
Measles	55	2618	46
Mpox	284	14070	143
Total	755	37044	540

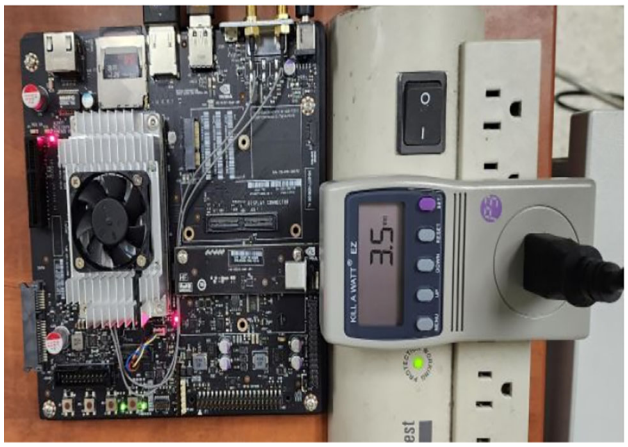


Fig. 8 Nvidia Jatson TX2 power measurement with P4400 p3 Kill-A-Watt meter

4.2 Implementation Details

We have used Python with PyTorch as the deep learning framework to implement the project. First, we create a custom data loader to load all samples from a folder. Next, we perform several transformations on the image, such as resizing, normalization, horizontal flip, and color jitter. Next, we use the Darknet53 as the backbone as it is much lighter than dense residual networks and has been shown to preserve semantic information well from low-level features. We use the pytorch $LeakyReLU$ activation unit and $Conv2d$ for 2D convolution operation throughout the deep layers of the DarkNet53. At the output layer, we use $Softmax$ as the activation. To calculate the ICS, we use the PyTorch sum operation to sum up all neuron activation. Next, we use the torch $Clamp$ method to restrict the ICS within [0.7, 1.2]. Finally, for large-margin loss calculation, we use PyTorch $MultiMarginLoss$, where we pass the softmax class output probability logits and the ground truth class id. The weights and biases were updated using $Adam$ optimizer with a weight decay of 0.003 as a regularization. We also reduce the learning rate by a factor of 0.01 every seven epochs. The model can be deployed on both mobile and small embedded devices. For medical practitioners, the easiest way to give access to the trained image classification model is through mobile applications or computer software. The recent popular method is via mobile applications due to its availability and increased computational power. To deploy a model via the Android application, first, we need to convert the trained TensorFlow model to a mobile-friendly format (TFLite) using the TensorFlow Lite Converter. Next, we need to add the TFLite model as an asset to the Android project. Finally, we can use TensorFlow Lite Interpreter to load and run the model inside the Android application.

4.3 Hyper-parameter Settings

In our proposed network, the load image size was 256×256. To train our model, we have resized all images to 128×128 pixels and set 64 as the mini-batch size in each epoch. The learning rate for training all models was set to 0.003. We found this learning rate provides the most stable and generalized performance for the test set. We also use 25% dropout layers to reduce the overfitting issues during the training phase. Without dropout and batch normalization, the test result outcome was found to be suboptimal to the proposed hyper-parameter settings. We put the $70 : 20$ ratio for the training and validation set. Next, the Margin and Penalty in the $MultiMarginLoss$ loss function were set to 10.0 and 1, respectively. The system specification for experimental devices is presented in Table 2. We have used the workstation presented in Table 2 for all training and fine-tuning. We run the inferences on both the workstation and embedded Jaston TX2; this gives us the difference between power consumption in high GPU and limited GPU devices.

5 Experiments

This work aims to propose a model that can produce optimal performance compared to SOTA, and the model is very lightweight in terms of power consumption, memory consumption, and GFLOPS. Toward this goal, we divide our discussion of experimental results into two parts. In the first part, we focus on classification performance. Here, we use several performance metrics, such as Precision, Recall, and Accuracy, to show the superiority of our model. We also compare our model with several SOTA classification pipelines to prove the consistency. Below in Eq. 8, we define the classification performance measurement metrics.

The calculation of precision (P), recall (R), and accuracy score (AC) is determined through the Eq. 8. True positives (TP) denote instances accurately predicted by the model, false positives (FP) represent instances wrongly predicted as a positive case by the model, and false negatives (FN) indicate instances incorrectly predicted as a negative case by the model and (TN) denotes the correctly predicted negative-cases. Precision (P) represents the proportion of relevant instances recovered by the model, while recall (R) is the fraction of relevant instances correctly identified by the model among all relevant instances. The accuracy score provides an overall measure of correct classification by the model.

$$P = \frac{TP}{TP + FP}, \quad R = \frac{TP}{TP + FN},$$
$$AC = \frac{TP + TN}{TP + TN + FP + FN} \tag{8}$$

On the other hand, we use metrics such as GFLOPS and Num. of Layers, model size, and power consumption (in Watt) for computational expense measurement. Based on this information, we analyze and compare models with our proposed model and discuss its suitability for deployment in embedded resource-constrained devices. Following a previous work [34], we use a P4400 p3 Kill-A-Watt meter (See Fig. 8) for calculating the power consumption. We connect the power meter to the CPU line and take the baseline power consumption reading to run the workstation. We carry several readings and then average the numbers for an accurate and sTable baseline result. Next, we run the model for inference and see how much power consumption increases due to the model inference via GPU and CPU.

5.1 Classification Performance Comparisons:

To verify the challenges in the dataset, we tackle the tasks as 1) Binary Classification and 2) Multi-class Classification problem. To perform the binary classification, we kept Mpox as the positive class and grouped all the classes to create the negative class. In this way, we had 14,354 positive Mpox samples and 22,690 negative samples for training. On the other hand, for multi-class classification, we kept the classes separate and performed training and testing. For SOTA model comparison, we implemented several comparing models following the latest research [1, 16, 28, 33, 35] on this topic. Most of the work used popular classifiers for performance comparisons, such as ResNet50, MobileNetv2, DenseNet121, and Transformer. We use DarkNet53 as the baseline and show the improvements through our Binary-DarkNet53 model.

Binary Classification: Fig. 9 shows the binary classification confusion matrix for the baseline and the proposed model. Here, we can see that the model performs reasonably well on the test dataset, but there are still some FP and FN cases. As we see in 1, there are samples where inter-class similarity and intra-class diversity exist. To solve the problems, we implemented Image Complexity Scores (ICS) and the Large-Margin Loss function. For ICS calculation in Eq. 5,

Table 2 System Specifications

Spec	Workstation	Nvidia Jetson TX2
CPU	Intel® CoreTM i9-11900K	Dual-Core NVIDIA Denver 2 64-Bit
CPU Core	3.50GHzx16	Quad-Core ARM® Cortex®-A57 MPCore
RAM	167GB 128-bit LPDDR4	8GB 128-bit LPDDR4
GPU	GP102 TITAN Xp	256-core NVIDIA Pascal™ GPU

Table 3 Binary Classification performance comparisons with several latest classification models. Here, ICS= Image Complexity Score, LM= Large Margin

Method	Precision	Recall	F1 Score	Accuracy
ResNet50	84.38	86.40	85.38	89.84
DenseNet121	92.00	89.84	90.91	93.73
DarkNet53	86.26	90.40	88.28	91.58
Jaradat et al. [35]	87.40	88.80	88.10	91.78
MonkeyNet [28]	88.84	92.20	90.15	92.10
ViT [1]	92.82	87.33	89.55	93.79
BinaryDNet53 W/O ICS+LM	89.90	92.05	90.96	90.11
BinaryDNet53	91.47	94.40	92.91	**95.05**

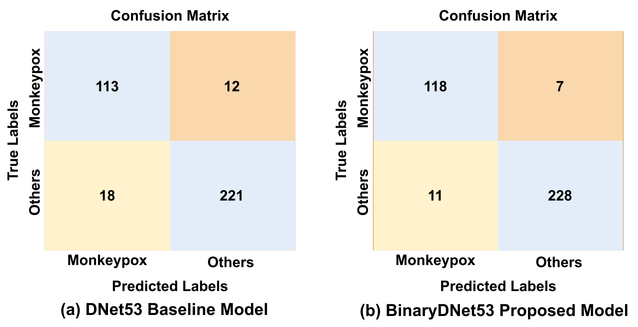


Fig. 9 Binary Classification Confusion matrix for baseline and proposed BinaryDNet53 models. Here, DNet53 = DarkNet53

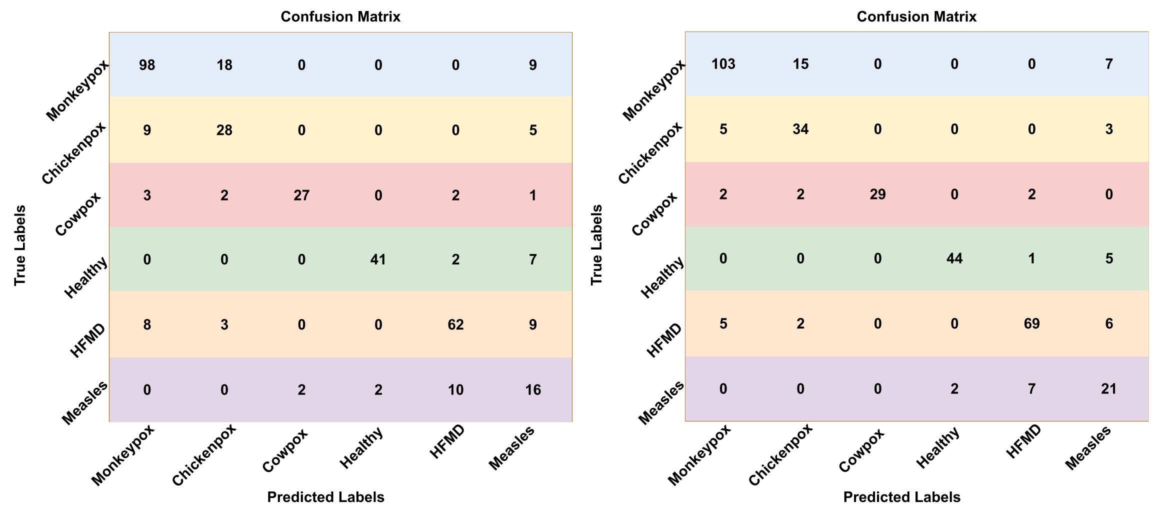
we use Lower Bound = 0.7 and Upper Bound = 1.2 to represent easy and complex examples, respectively. Using our proposed model, we resolved the challenges and recorded optimal results with only 7 FN cases and 11 FP cases. The overall performance was entirely satisfactory, considering the lightweight characteristics of the model.

In Table 3, we compare different SOTA classification methods with our proposed method. For the Mpox classification task, the Recall metric is crucial. If we fail to detect a positive Mpox case correctly, the patient may get released from the hospital and infect more healthy people who are in contact with them. From Table 3, ResNet50 gives the least recall approx. 86%, MonkeyNet [28] and DarkNet53 provide very close recall 92% and 90%, respectively. On the other hand, ViT [1], a vision-transformer model, shows promising performance with 93.79% accuracy, although the model achieves sub-optimal recall (87.33%), which is very crucial for virus-infected disease. Moreover, the computational cost for MonkeyNet, ViT, and Jaradat et al. [35] is significantly higher than the DarkNet53 model. We show our proposed model's performance with and without ICS+LM modules. It is worth noting that the Binarized DarkNet53 performs inferior to the original DarkNet53 implementation. The reason behind this is that, due to binary weights and bias, the model misses some precision values after the decimal points, which makes the learning slightly noisy. However, after incorporating the ICS and LM, we could compensate for the noise and achieve optimal results of approximately 95.05% of accuracy and 94.40% of recall on the test set. Our closest competitor in classification performances was the ViT model, with 93.79% of accuracy and 87.33% of recall.

Table 4 Statistical results from proposed BinaryDNet53 and ViT model

t-Test T-statistic	P-value	chi-square Chi-squared	Test P-value
5.87	0.004	7.03	0.07

To conduct statistical testing, we performed k-fold cross-validations in our validation set. The goal is to prove that the performance metrics are statistically significant, thereby adding robustness to our experimental findings. We performed a t-test and chi-square test between the proposed model and the closest competitor. For paired t-test calculation, we collected accuracies from the 5-fold cross-validations for the proposed BinaryDNet53 and closest SOTA ViT model. From Table 4, we see that Model BinaryDNet53 had a mean accuracy of 93.4% (SD = 1.35%), while Model ViT had a mean accuracy of 91.3% (SD = 1.9%). The paired t-test revealed a statistically significant difference between the model's accuracies ($t(4) = 5.87$, $p = 0.004$). This indicates that Model BinaryDNet53 performs significantly



(a) DNet53 Baseline Model **(b) BinaryDNet53 Proposed Model**

Fig. 10 Multi-class Confusion matrix for baseline and proposed BinaryDNet53 models. Here, DNet53 = DarkNet53

better than Model ViT in terms of accuracy. For calculating the chi-square test, we use the information from the confusion matrices of two candidate models. The result from the chi-square test is presented in Table 4, which showed a significant difference (chi2 = 7.03, p = 0.07). These results indicate that Model BinaryDNet53 performs significantly better than Model ViT.

Multi-Class Classification: The goal of multi-class classification is to observe the effect of inter-class similarity and intra-class dissimilarity issues in the dataset on Recall and precision values. It becomes harder to distinguish some classes correctly, which lowers the True Positive and True Negative rates. We can confirm this fact from Fig. 10, where we can see there are several classes where we have False Positives and False Negatives. We found two groups with some overlapping characteristics. From Fig. 10, we see the class Monekypox and Chickenpox have some inter-class similarity issues. The same goes for HFMD and Measles classes, which lead to reduced Precision and Recall values. We present precision, recall, F1 score, and accuracy in Table 5. In Table 5, we compare the performance from our baseline and proposed model with several of the latest SOTA methods. Where most of the traditional classification model suffers from low recall, DenseNet121 performed reasonably well with a recall of 78.20% and an accuracy of 82.45%. Our baseline model shows quite a competitive performance compared to the traditional models, with a recall of 78.65% and an accuracy of 81.12%. However, our proposed model achieves the optimal result with a recall of 82.46% and an accuracy of 85.78%. The nearest SOTA performance was achieved from the ViT model with recall and accuracy of 80.33% and 83.76%, respectively. It is evident from the performance analysis that the dataset is relatively challenging for the multi-class classification task due to inter-class similarity issues in several diseases.

5.2 Computational Performance Comparisons:

The most crucial aspect of our work is to provide a model with a shallow power consumption requirement and low GFLOPS. Table 6 shows that our proposed model is 20× lightweight than the baseline model in terms of GFLOPS. All other models with decimal weight and bias require significantly more GFLOPS and learnable parameters than the proposed model. The MonkeyNet and DensNet201 require 1163.93 and 929.61 GFLOPS and 20.81 and 6.9 Million parameters, respectively, for a batch size = 64. The ViT model also shows high computational expense regardless of its better performance. ViT model requires 890.61 GFLOPS for 12.6M learnable parameters with a batch size of 64. On the other hand, Jaradat et al. (MobileNet2) were found to be very convincing, with 91.76% accuracy and 52.94 GFLOPS. These heavy models are unsuitable for small devices in real-life scenarios, and the detection rate does not meet real-time requirements.

To calculate the power consumption, we run inference on the workstation and a small embedded device, Nvidia Jetson TX2 (See Fig. 8). The power consumption for our top three performers, DensNet121, Jaradat, et al. (MobileNet2), and ViT on the workstation are 291, 178, and 311 watts, respectively. On the other hand, the embedded devices require 201, 89, and 205 watts, respectively. Our goal was to design a model that will require ≈ 65 watts to use a mobile charger of 65 watts and get continuous throughput without issues. Our BinaryDNet53 can lower the power consumption to 67 watts on the embedded device, which is very close to our

Table 5 Multi-class Classification performance comparisons with several latest classification models. Here, ICS= Image Complexity Score, LM= Large Margin

Method	Precision	Recall	F1 Score	Accuracy
ResNet50	78.19	72.08	76.45	75.77
DenseNet121	81.45	78.20	80.33	82.45
DarkNet53	77.52	75.65	76.39	79.10
Jaradat et al. [35]	77.04	76.11	76.80	78.92
MonkeyNet [28]	79.48	77.85	78.56	80.54
ViT [1]	84.82	80.33	82.73	83.76
BinaryDNet53 W/O ICS+LM	80.70	78.65	79.68	81.12
BinaryDNet53	86.92	82.46	84.20	**85.78**

Table 6 Multi-factor computational cost comparison between our proposed BinaryDNet53 and recent SOTA methods. Power consumption is measured in the **Watt** unit

Method	# of Params	GFLOPS	Power Consumption Jatson TX2	Power Consumption Workstation
ResNet50	25.6M	86.47	110 ± 5	195 ± 5
DenseNet121	6.9M	929.61	201 ± 8	291 ± 8
DarkNet53	36.4M	149.33	121 ± 7	210 ± 5
Jaradat et al. [35]	2.34M	52.94	89 ± 5	178 ± 5
MonkeyNet [28]	20.81M	1164.93	236 ± 8	331 ± 8
ViT [1]	12.6M	890.61	205 ± 8	311 ± 8
BinaryDNet53	**0.37M**	**7.52**	67 ± 2	138 ± 3

goal. Even in the workstation inference, our proposed model shows superiority by consuming the lowest power($\approx$ 139) compared to all SOTA methods. The memory consumption from the SOTA models varied between 140 to 270 MB. Our proposed model requires only 84 MB of space, which is reasonable for any small device.

6 Conclusion

In this study, we proposed a lightweight Mpox classification model for resource-constrained mobile or embedded devices. Here, we show that using the binary weights and bias can make the model significantly lighter. It is to be noted that the binary weights and bias can reject some precision values after decimal points, which should be carefully addressed for smooth learning. It was observed that the classification gets challenging due to intra-class dissimilarity and inter-class similarities in several classes in the MSLD v2.0 dataset. To resolve the issues, we proposed Image Complexity Scoring (ICS) based on saliency information to focus more on complex examples. Our proposed model outperformed all SOTA methods in terms of classification and computational performance. Our BinaryDNet53 achieved 94.40% recall and 95.05% accuracy for the binary classification task and 82.46% recall and 85.78% accuracy for the multi-class classification task. Moreover, we were able to reduce the memory and power Consumption to 84 MB and 67 watts, respectively. Finally, our model is $\approx 20\times$ lightweight in terms of GFLOPS and $\approx 2\times$ power efficient than the baseline model. In the future, we plan to extend our work to a more diverse skin disease classification dataset so that we can provide a one-stop solution for all skin disease detection. Also, we aim to develop a mobile application for our next research project and make it open-source for all practitioners.

Author Contributions D.B. formulated the problem statement, implemented the project, and conducted all experiments and wrote the manuscript. J.T. supervised the project and reviewed the manuscript.

Funding The authors did not receive support from any organization for the submitted work.

Availability of supporting data Data used in this article were obtained from the following Kaggle (dataset link).

Declarations

Conflict of interest The authors declare no Conflict of interest.

Ethical Approval Not applicable.

References

1. Kundu, D., Siddiqi, U.R., Rahman, M.M.: Vision transformer based deep learning model for monkeypox detection. In: 2022 25th International Conference on Computer and Information Technology (ICCIT), pp. 1021–1026 (2022). IEEE
2. Jui, T.D., Bejarano, G.M., Rivas, P.: A machine learning-based segmentation approach for measuring similarity between sign lan-

guages. In: Sign-lang@ LREC 2022, pp. 94–101 (2022). European Language Resources Association (ELRA)
3. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767 (2018)
4. Biswas, D., Tešić, J.: Unsupervised domain adaptation with debiased contrastive learning and support-set guided pseudo labeling for remote sensing images. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. (2024)
5. Xie, Y., Zaccagna, F., Rundo, L., Testa, C., Agati, R., Lodi, R., Manners, D.N., Tonon, C.: Convolutional neural network techniques for brain tumor classification (from 2015 to 2022): Review, challenges, and future perspectives. Diagnostics **12**(8), 1850 (2022)
6. Singh, S., Tripathi, B.: Pneumonia classification using quaternion deep learning. Multimed. Tools Appl. **81**(2), 1743–1764 (2022)
7. Abbas, A., Abdelsamea, M.M., Gaber, M.M.: Classification of covid-19 in chest x-ray images using detrac deep convolutional neural network. Appl. Intell. **51**, 854–864 (2021)
8. Fowotade, A., Fasuyi, T., Bakare, R.: Re-emergence of monkeypox in nigeria: A cause for concern and public enlightenment. Afr. J. Clin. Exp. Microbiol. **19**(4), 307–313 (2018)
9. Kelleher, S.R.: CDC Raises Monkeypox Travel Alert to Level 2. https://www.forbes.com/sites/suzannerowankelleher/2022/06/07/cdc-raises-Monkeypox-travel-alert-to-level-2/?sh=67c264f83f93
10. Wamsley, L.: There Has Been a Shortage of Testing and Vaccines for Monkeypox. NPR (2022)
11. Hussain, M.A., Islam, T., Chowdhury, F.U.H., Islam, B.R.: Can artificial intelligence detect monkeypox from digital skin images? BioRxiv, 2022–08 (2022)
12. Biswas, D., Tešić, J.: Small object difficulty (sod) modeling for objects detection in satellite images. In: 2022 14th International Conference on Computational Intelligence and Communication Networks (CICN), pp. 125–130 (2022). IEEE
13. Rastegari, M., Ordonez, V., Redmon, J., Farhadi, A.: Xnor-net: Imagenet classification using binary convolutional neural networks. In: European Conference on Computer Vision, pp. 525–542 (2016). Springer
14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
15. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.-C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4510–4520 (2018)
16. Ali, S.N., Ahmed, M.T., Paul, J., Jahan, T., Sani, S., Noor, N., Hasan, T.: Monkeypox skin lesion detection using deep learning models: A feasibility study. arXiv preprint arXiv:2207.03342 (2022)
17. Chaplot, S., Patnaik, L.M., Jagannathan, N.R.: Classification of magnetic resonance brain images using wavelets as input to support vector machine and neural network. Biomed. Signal Process. Control **1**(1), 86–92 (2006)
18. Usman, K., Rajpoot, K.: Brain tumor classification from multimodality mri using wavelets and machine learning. Pattern Anal. Appl. **20**, 871–881 (2017)
19. Afshar, P., Mohammadi, A., Plataniotis, K.N.: Brain tumor type classification via capsule networks. In: 2018 25th IEEE International Conference on Image Processing (ICIP), pp. 3129–3133 (2018). IEEE
20. Abiwinanda, N., Hanif, M., Hesaputra, S.T., Handayani, A., Mengko, T.R.: Brain tumor classification using convolutional neural network. In: World Congress on Medical Physics and Biomedical Engineering 2018: June 3-8, 2018, Prague, Czech Republic (Vol. 1), pp. 183–189 (2019). Springer
21. Narin, A., Kaya, C., Pamuk, Z.: Automatic detection of coronavirus disease (covid-19) using x-ray images and deep convolutional neural networks. Pattern Anal. Appl. **24**, 1207–1220 (2021)
22. Wang, S., Kang, B., Ma, J., Zeng, X., Xiao, M., Guo, J., Cai, M., Yang, J., Li, Y., Meng, X., et al.: A deep learning algorithm using ct images to screen for corona virus disease (covid-19). Eur. Radiol. **31**, 6096–6104 (2021)
23. Datta Gupta, K., Sharma, D.K., Ahmed, S., Gupta, H., Gupta, D., Hsu, C.-H.: A novel lightweight deep learning-based histopathological image classification model for iomt. Neural Process. Lett. **55**(1), 205–228 (2023)
24. Deng, Y., Hou, Y., Yan, J., Zeng, D.: Elu-net: An efficient and lightweight u-net for medical image segmentation. IEEE Access **10**, 35932–35941 (2022)
25. Tuncer, T., Barua, P.D., Tuncer, I., Dogan, S., Acharya, U.R.: A lightweight deep convolutional neural network model for skin cancer image classification. Appl. Soft Comput. 111794 (2024)
26. Ahsan, M.M., Uddin, M.R., Farjana, M., Sakib, A.N., Momin, K.A., Luna, S.A.: Image data collection and implementation of deep learning-based model in detecting monkeypox disease using modified vgg16. arXiv preprint arXiv:2206.01862 (2022)
27. Irmak, M.C., Aydin, T., Yağanoğlu, M.: Monkeypox skin lesion detection with mobilenetv2 and vggnet models. In: 2022 Medical Technologies Congress (TIPTEKNO), pp. 1–4 (2022). IEEE
28. Bala, D., Hossain, M.S., Hossain, M.A., Abdullah, M.I., Rahman, M.M., Manavalan, B., Gu, N., Islam, M.S., Huang, Z.: Monkeynet: A robust deep convolutional neural network for monkeypox disease detection and classification. Neural Netw. **161**, 757–775 (2023)
29. Wang, C.-Y., Liao, H.-Y.M., Wu, Y.-H., Chen, P.-Y., Hsieh, J.-W., Yeh, I.-H.: Cspnet: A new backbone that can enhance learning capability of cnn. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pp. 390–391 (2020)
30. Kumar, A., Sharma, A., Bharti, V., Singh, A.K., Singh, S.K., Saxena, S.: Mobihisnet: a lightweight cnn in mobile edge computing for histopathological image classification. IEEE Internet Things J. **8**(24), 17778–17789 (2021)
31. Biswas, D., Tesic, J.: Domain adaptation with contrastive learning for object detection in satellite imagery. Authorea Preprints (2023)
32. Ali, S.N., Ahmed, M.T., Paul, J., Jahan, T., Sani, S.M.S., Noor, N., Hasan, T.: Monkeypox skin lesion detection using deep learning models: A preliminary feasibility study. arXiv preprint arXiv:2207.03342 (2022)
33. Ali, S.N., Ahmed, M.T., Jahan, T., Paul, J., Sani, S.M.S., Noor, N., Asma, A.N., Hasan, T.: A web-based mpox skin lesion detection system using state-of-the-art deep learning models considering racial diversity. arXiv preprint arXiv:2306.14169 (2023)
34. Biswas, D., Rahman, M.M., Zong, Z., Tešić, J.: Improving the energy efficiency of real-time dnn object detection via compression, transfer learning, and scale prediction. In: 2022 IEEE International Conference on Networking, Architecture and Storage (NAS), pp. 1–8 (2022). IEEE
35. Jaradat, A.S., Al Mamlook, R.E., Almakayeel, N., Alharbe, N., Almuflih, A.S., Nasayreh, A., Gharaibeh, H., Gharaibeh, M., Gharaibeh, A., Bzizi, H.: Automated monkeypox skin lesion detection using deep learning and transfer learning techniques. Int. J. Environ. Res. Public Health **20**(5), 4422 (2023)