IEEE Open Journal of the Computer Society

Received 10 February 2024; accepted 13 March 2024. Date of publication 20 March 2024;
date of current version 26 April 2024. The review of this article was arranged by Associate Editor Peng Li.

Digital Object Identifier 10.1109/OJCS.2024.3378709

A Lightweight Visual Font Style Recognition With Quantized Convolutional Autoencoder

MOSHIUR RAHMAN TONMOY [1], ABDUL FATTAH RAKIB [2], RASHIK RAHMAN [2], MD. AKHTARUZZAMAN ADNAN [2], M. F. MRIDHA [3] (Senior Member, IEEE), JIE HUANG [4], AND JUNGPIL SHIN [4] (Senior Member, IEEE)

[1]Advanced Machine Intelligence Research Lab, Dhaka 1213, Bangladesh
[2]Department of Computer Science and Engineering, University of Asia Pacific, Dhaka 1205, Bangladesh
[3]Department of Computer Science, American International University-Bangladesh, Dhaka 1229, Bangladesh
[4]School of Computer Science and Engineering, The University of Aizu, Aizuwakamatsu 965-8580, Japan

CORRESPONDING AUTHOR: JUNGPIL SHIN (e-mail: jpshin@u-aizu.ac.jp).

ABSTRACT Font style recognition plays a vital role in the field of computer vision, particularly in document and pattern analysis, and image processing. In the industry context, this recognition of font styles holds immense importance for professionals such as graphic designers, front-end developers, and UI-UX developers. In recent times, font style recognition using Computer Vision has made significant progress, especially in English. Very few works have been done for other languages as well. However, the existing models are computationally costly, time-consuming, and not diversified. In this work, we propose a state-of-the-art model to recognize Bangla fonts from images using a quantized Convolutional Autoencoder (Q-CAE) approach. The compressed model takes around 58 KB of memory only which makes it suitable for not only high-end but also low-end computational edge devices. We have also created a synthetic data set consisting of 10 distinct Bangla font styles and a total of 60,000 images for conducting this study as no dedicated dataset is available publicly. Experimental outcomes demonstrate that the proposed method can perform better than existing methods, gaining an overall accuracy of **99.95%** without quantization and **99.85%** after quantization.

INDEX TERMS Visual font recognition, computer vision, convolutional autoencoder, quantization, TinyML, edge devices, bangla font style.

I. INTRODUCTION

The rise of digitization in today's age has helped the area of graphic design to go beyond traditional boundaries and achieve unprecedented levels of richness and diversity. Fonts have turned into a vital component of this creative environment, especially in the domain of typography. In a time when visual communication is crucial, fonts are the digital canvas's creative brushstrokes. They capture a message's essence, arouse emotions, and turn plain words into compelling visual stories. Typography becomes a medium for storytelling in a visually rich setting as designers and artists use various font styles to communicate ideas and the identities of brands and products. When they come across new font styles in their daily lives, graphic artists want to be able to recognize them for future uses [1].

The development of a powerful automatic font recognition system that can recognize styles in images or photos has an opportunity to minimize the tedious manual tasks that designers often undergo. In addition to making font-related tasks simpler, such a system would improve font selection and management during the design phase. This invention will be extremely beneficial to designers because it will let them free up their vital time and allow them to quickly include variety in their creative endeavors. However, building a thorough Visual Font Recognition (VFR) system has its own set of difficulties due to the enormous variety of font types and their subtleties and complexities [2]. Though there are already many useful systems like Identifont[1] or MyFonts[2] that can identify fonts

[1][Online]. Available: www.identifont.com
[2][Online]. Available: www.myfonts.com/pages/whatthefont

and suggest similar ones, the field of font recognition is still developing, especially in languages that hold complex characters such as Bangla, Hindi, and so on. This opens the door to more inventive and productive design processes.

Bangla, also referred to as Bengali, is an international language that is native to Bangladesh and the Indian states of Tripura and West Bengal. With around 273 million speakers, Bangla is spoken by a sizeable fraction of the world's population as both a primary and second language. Remarkably, Bangla is the seventh most spoken language worldwide and the sixth most spoken native language [3]. As expected, there has been a considerable increase in demand for various Bangla font styles for design and other relevant purposes to attract Bangla-speaking audiences by bringing diversity and uniqueness in every possible form of work that relates to the use of Bangla fonts, such as newspapers, invitation, greeting cards, visiting cards, testimonies, brand logos, billboards, covering from physical platform to digital mediums as well. However, there hasn't been much advancement in the area of autonomous Bangla font recognition.

Several industries stand to gain greatly from the widespread usage of Bangla font styles. Regarding brand consistency, it ensures that a recognizable and consistent font style remains intact over a range of brand assets, strengthening the company's identification and reliability [4]. Designers may ensure visually appealing and harmonious designs by matching their creative concepts with the proper Bangla font types with the help of font recognition, which preserves design and aesthetic integrity. Understanding Bangla typefaces helps with content localization by allowing content to be customized to the linguistic and cultural tastes of various audiences who speak Bangla. This improves the information's accessibility and resonance. Furthermore, using appropriate Bangla font styles can help create more user-friendly and compelling interfaces, which will increase the user experience overall. Last but not least, font recognition is a useful tool for designers and developers as it helps with design replication, making it easier to consistently reproduce particular Bangla font styles across different projects. Simply put, the ability to identify Bangla font types gives these industries the tools they need to preserve design continuity, improve user experiences, adjust the material for a wider audience, and preserve brand consistency.

In response to the pressing demand for effective Bangla font style recognition, the goal of our research is to create a novel, lightweight convolutional autoencoder method to reduce the tedious manual work that innumerable designers, artists, and developers who deal with Bangla fonts must perform to recognize font styles accurately. This research effort seeks to provide a practical and efficient solution to enhance people's creativity and productivity in a range of professional sectors. The major contributions of this study are the following:

- We propose a lightweight Convolutional Autoencoder for effectively recognizing visual font styles from images.
- We incorporated int-8 quantization to reduce the computational complexity of the proposed model for deployment in the edge devices.
- We created a large synthetic data set employing 10 different Bangla font styles with a total of 60,000 images for this research and made it public.
- We also present a comprehensive performance analysis of our proposed method with past works as well as state-of-the-art CNNs.

The rest of the paper is organized as follows: Section II discusses the related works followed by Section III discusses a background about Convolutional Autoencoder (CAE) and model quantization. Section IV provides a brief overview of the Bangla VFR data set. Section V presents the architecture of the proposed system. Section VI contains the findings of this study and discusses model performance. Limitations and future work are discussed in Section VII and Section VIII concludes this study.

II. RELATED WORKS

This section delves into contemporary approaches in font style recognition, focusing primarily on the latest developments in Bangla font style recognition while encompassing recent advancements in other font styles as well. It provides a glimpse into their respective methodologies and performance.

A. FONT STYLE RECOGNITION IN VARIOUS LANGUAGES

Wang et al. [1] created the popular AdobeVFR dataset, and they proposed the DeepFont framework. Their approach leverages a domain adaptation technique that relies on a Stacked-CAE that uses a substantial collection of unlabeled real-world English text images, which are augmented with synthetic data that has been preprocessed in a specialized manner. The DeepFont system attains a top-5 accuracy exceeding 80%.

Li et al. [5] utilized a downsampling CNN approach for classifying font styles for Chinese characters, and attained a 99.03% accuracy with their dataset consisting of 18,000 instances of 18 distinct font styles.

Mohammadian et al. [6] employed CNN to recognize Persian font styles, and they presented top-1 accuracy of 78.0% on their own newly created dataset, at best 94.5% on the other public ones.

Tensmeyer et al. [7] proposed a font recognition model using a simple CNN and a new form of data augmentation that can enrich robustness in terms of text colors. By using the same method they found 98.8% accuracy in the Arabic dataset consisting of 40 distinct fonts as well as 86.6% in the Latin language.

Cheng et al. [2] have shown the issues of large-scale visual font recognition (VFR). They have created an ample dataset using 2,420 English font styles and developed an algorithm using the nearest class mean classifier (NCM). This classifier has given promising results in synthetic data as well as real-world test data.

Vijayakumar et al. [8] made use of the Capsule Network (CapsNet) algorithm to implement a font style classification system and attained 96.15% accuracy. However, they focused on only 3 English common font styles, namely - Times New Roman, Arial Black, and Algerian.

B. FONT STYLE RECOGNITION IN BANGLA LANGUAGE

Recently Hasan et al. [4] presented a transfer learning-based approach where 5 different types of font styles were explored with an augmented dataset of 33,800 instances and they achieved 96.23% accuracy with pre-trained VGG-16 over VGG-19 and Xception. However, their employed dataset lacks diversity. Previously, Islam et al. [9] utilized a convolutional neural network (CNN) with a space adjustment technique based on SCAE to detect Bangla fonts. They curated a dataset with 12,187 images in 7 fonts, totaling 77,728 samples, and achieved an average font recognition accuracy of 98.73%. Nonetheless, their accuracy is unstable and no separate test set evaluation has been conducted. Hasan et al. [10] also introduced a CNN-based model for recognizing Bangla font styles, achieving 96% accuracy with a dataset including partially labeled real-world data and marked synthetic data of 10 Bangla fonts. Earlier, Wang et al. [11] proposed a transfer learning-based font classifier, which can recognize 6 language fonts including Bangla, English, Arab, Japanese, Korean, and Chinese. They have blended the text with background images to mimic the real-world scenario and used 2 transfer learning models with 4 variants. The highest accuracy has been seen in VGG16 (top-5) consisting of 97.53%.

Although researchers have already explored the Bangla VFR domain, it is patent that notable advancements have yet to be made in the VFR domain as existing studies exhibit insufficient accuracy in recognizing visual font styles, especially in Bangla, and lack diversity in datasets. Furthermore, previous studies have predominantly relied on resource-intensive Transfer Learning or deep DL architectures, hindering VFR model deployment on edge devices. This study presents state-of-the-art performance in the Bangla VFR domain by utilizing a comprehensive and diverse dataset. In addition, we introduce a novel quantized lightweight architecture suitable for deployment on resource-constrained edge devices.

III. BACKGROUND

In this section, we present the necessary background to prepare the stage for the comprehension of the methodology proposed in this study. Firstly, we briefly discuss the Convolutional Autoencoder in Section III-A, followed by a brief on Quantization in Section III-B.

A. CONVOLUTIONAL AUTOENCODER

Autoencoders are deep learning architectures capable of reconstructing data instances from their feature vectors and they are learned automatically from data examples. It consists of mainly an encoder and a decoder. The Convolutional autoencoder (CAE) is a type of neural network architecture used in tasks related to image or spatial data. It combines elements of both convolutional neural networks and autoencoders to learn efficient representations of input data [12].

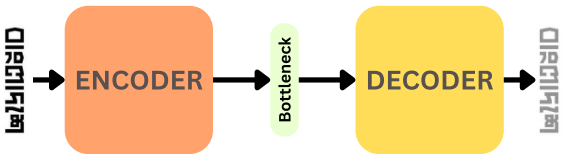


FIGURE 1. Standard architecture of a convolutional autoencoder comprising both encoder and decoder.

In the encoder part, the network employs one or more convolutional layers, often followed by pooling layers. These layers are responsible for learning to extract relevant and crucial features from the input image data. Convolutional layers are utilized to capture patterns and details of the image by applying multiple filters to the input while pooling layers are used to downsample the feature maps to reduce the spatial dimensions. The encoder work mechanism can expressed as the (1).

$$Encoder_{out} = Activation(Convolution(Input)) \quad (1)$$

After the encoder layers, we get the latent space representation that contains the compact representation of the input image where the most vital spatial features are captured. This layer is often known as the bottleneck layer which can be formalized as (2).

$$Latent\_Space = Flatten(Encoder_{out}) \quad (2)$$

Lastly, in the decoder part of the network, it mimics the encoder but in reverse. It employs multiple layers of transposed convolutions often followed by upsampling layers, and sometimes even followed by additional convolutional layers as expressed via (4).

$$Upsampled_{out} = Upsampling(Latent\_Space) \quad (3)$$

$$Decoder_{out} = Activation(Convolution(Upsampled_{out})) \quad (4)$$

The prime job of these layers in the decoder is to reconstruct the input data from the compressed representation generated by the encoder. The final layer usually matches the dimension of the original image but can vary based on the applications. Fig. 1 illustrates a standard convolutional autoencoder.

By training the network to encode and decode data, the model can learn to capture meaningful patterns and features from the input data, which can be valuable for downstream applications. Convolutional autoencoders are useful for various tasks, primarily in image denoising [13], [14], image compression [15], [16], feature extraction [17], clustering [12], [18], anomaly detection [19], [20], and so on. CAEs are also employed in image classification as employed in this study and other such tasks [21], [22] while the main motivation behind using CAEs lies in the applications mentioned above. However, CAEs can also be powerful for such tasks as shown in this study.

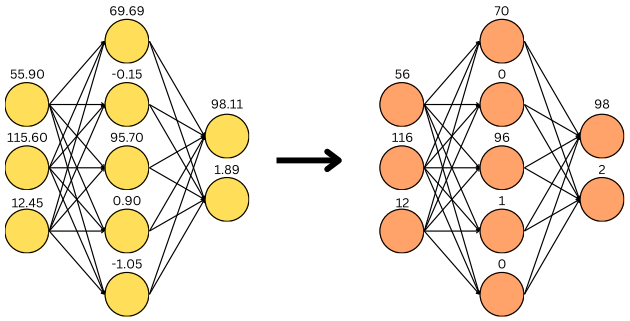


FIGURE 2. Overview of quantization of neural networks from float32 to uint8.

B. QUANTIZATION

Quantization is a way to lower the memory and computational expenses of conducting inference by using low-precision data types like the 8-bit integer (int8) instead of the common 32-bit floating point (float32) [23]. By using fewer bits, the final model should use less memory space, use less energy, and benefit from faster integer arithmetic for operations [24]. Additionally, it enables the use of models on embedded hardware, which occasionally only supports integer data types. Quantization is a popular technique for compressing complex and large DL models to deploy them in resource-constraint edge devices [25], [26]. The formula for int8 quantization is (5), where the value is firstly rounded to its nearest integer, and the clipping operation takes place to clip the value in the spectrum of [-128, 127]. On the other hand, in uint8 quantization, the clipping restrictions lie in a positive band starting from zero, as shown in (6).

$$y = Clip(round(x), -128, 127) \tag{5}$$

$$y = Clip(round(x), 0, 255) \tag{6}$$

Fig. 2 presents an overview of uint8 quantization, where values are rounded and clipped to the positive range as defined by (6).

IV. BANGLA VFR DATASET

There is no public dataset available for the Bangla visual font recognition task. For this reason, we have created a synthetic dataset of 60,000 images. Dataset generation and preparation is done by the following -

Firstly, we generated unique Bangla words from a large Bangla text corpus, and for the accomplishment we utilized the Wikipedia Bangla article corpus available publicly in Kaggle [27], a popular platform for data scientists. By preprocessing the initial generated words, we then extracted our desired list of words. Among the final list of words, we randomly picked 1000 unique words with a minimum length of 9 and a maximum length of 10 to be placed on the images.

Secondly, we collected 10 popular Bangla fonts from the two most popular Bangla font-sharing sites, namely Lipighor [28] and FontBD [29]. Finally, to generate images, firstly we picked a 400 × 400 white background image, and then for each unique word, we printed the word in the white blank image with a font size of 80 for 10 different selected fonts, thus creating 10 different classes of font style. We also added 5 different rotated copies of each image to introduce diversity in the dataset. So each word has 6 instances per class, which makes a total of 6000 images for 1000 words per class and a dataset containing 60000 images of 10 distinct Bangla font styles. We have employed Python programming language and its popular packages to accomplish the above-mentioned tasks. Table 1 presents the dataset summary and Fig. 3 illustrates sample instances of the dataset, also the sample augmentation instances are also illustrated via Fig. 4 as well.

TABLE 1. Dataset Summary

Class name	Unique Instances	Augmented	Total
FN Jagat Shonkhoneel	1000	5000	6000
FN Suborno Jayonti	1000	5000	6000
Li Alinur Phulkuri	1000	5000	6000
Li Alinur Saikat	1000	5000	6000
Li Alinur Sanghoti	1000	5000	6000
Li Fazlay Munnisha	1000	5000	6000
Li Mahfuz Himadri	1000	5000	6000
Li Niladri Russian	1000	5000	6000
Li Patabahar	1000	5000	6000
Li Upohar 56	1000	5000	6000
		Total	60000

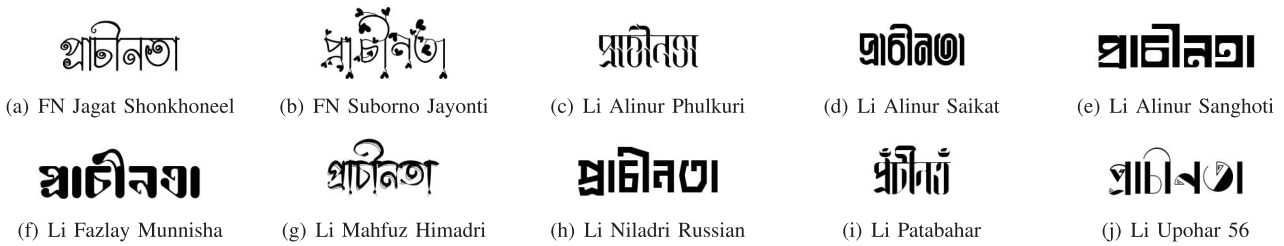
V. METHODOLOGY

In this section, we present the methodology of our proposed framework for recognizing Bangla visual fonts. We start by going over the architecture of the recognition model in Section V-A, followed by quantization of the model in Section V-B, and finally we discuss the overall workflow of our proposed framework in Section V-C.

A. RECOGNITION MODEL

As described in Section III-A, apart from classic applications such as image denoising, and image compression, Convolutional Autoencoders are powerful also for image classification tasks which can be witnessed through this study as well.

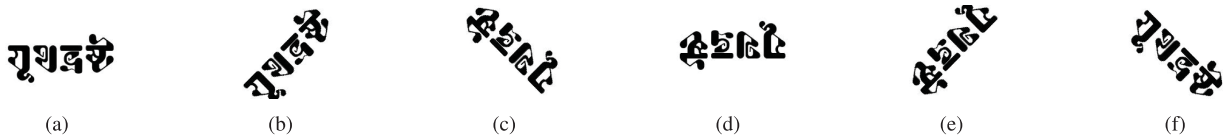
In our architecture, we employed a lightweight convolutional architecture and added a classification header on top of the decoder to accomplish our target. Fig. 5 illustrates the architecture of our employed model. As shown there, in the encoder part - the inputs first undergo two Convolution layers - layers 2 and 3, followed by a MaxPooling layer, and again through two Convolution layers - layers 5, and 6. Then a GlobalMaxPooling layer is added to extract the maximum spatial information across all channels from the feature map. This is the bottleneck layer which holds the compact representation of the spatial information of the input image. From layer 8, the decoder part comes into action, starting with reshaping



(a) FN Jagat Shonkhoneel (b) FN Suborno Jayonti (c) Li Alinur Phulkuri (d) Li Alinur Saikat (e) Li Alinur Sanghoti

(f) Li Fazlay Munnisha (g) Li Mahfuz Himadri (h) Li Niladri Russian (i) Li Patabahar (j) Li Upohar 56

FIGURE 3. Sample data instances from the Bangla VFR dataset.



(a) (b) (c) (d) (e) (f)

FIGURE 4. Sample of augmented data instances.

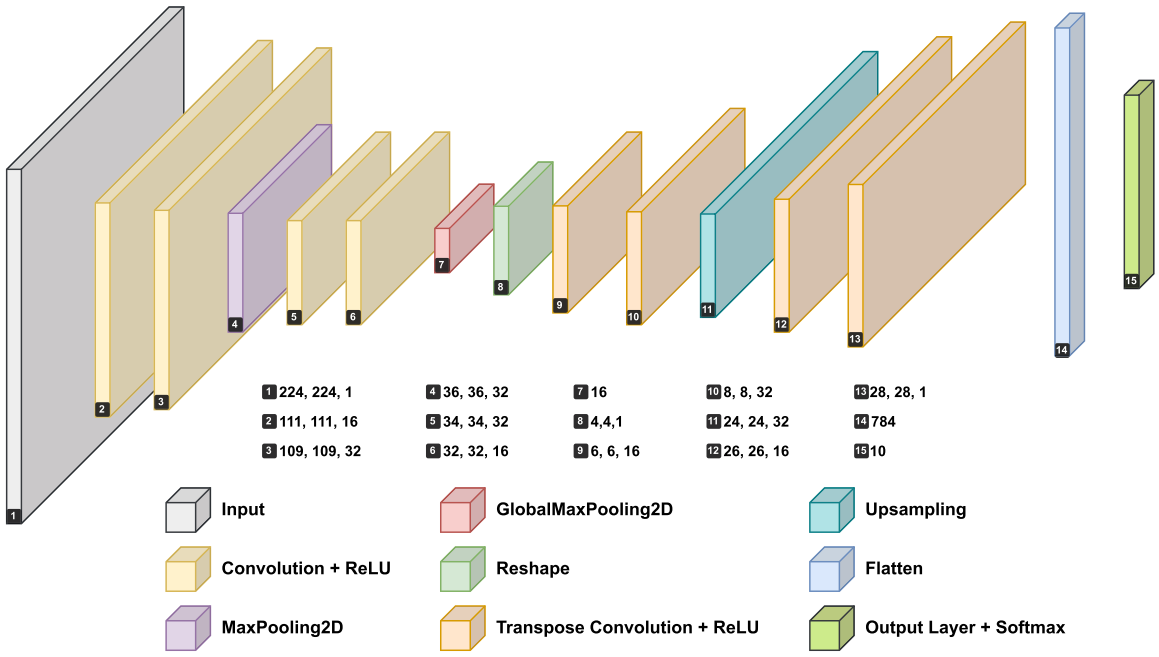


FIGURE 5. Architecture of the proposed convolutional autoencoder for recognizing Bangla visual font styles.

the tensor provided by the GlobalMaxPooling layer into a suitable shape, followed by two TransposeConvolution layers (layers 9, 10) to enlarge the encoded tensor based on the spatial information. Then we upsampled the spatial dimension by a factor of 3, followed by two TransposeConvolution again to enlarge the upsampled tensor further in the context of the spatial information.

Finally, to accomplish our classification task, at layer 14 - we flattened out the tensor containing the spatial information from the previous layer and fed it into an output layer with Softmax as the activation function to recognize the font style from the input image out of the 10 style classes utilized in the training process. Table 2 provides a comprehensive overview of the proposed architecture, detailing each layer's output shape and the associated parameters. It contains layer-by-layer information according to the Fig. 5. In total, the model comprises 36,091 parameters, all of which are trainable. This concise summary serves to elucidate the structural components and complexity of the proposed architecture.

For all the convolution layers employed here, whether regular or transpose, we used ReLU as the activation function. The formula employed in the ReLU activation function is expressed in (7). Here, it keeps positive input values the same and outputs zero for any negative input values. It introduces non-linearity and helps neural networks learn complex patterns and features in data.

$$ReLU(x) = \max(0, x) \tag{7}$$

We employed the widely used Adam optimizer [30] that incorporates adaptive learning rate and momentum feature which accelerates the optimization of deep learning models. Furthermore, the Categorical Cross-Entropy (CCE) loss

TABLE 2. Summary of the Proposed Architecture

Layer	Output Shape	Parameters
Input Layer	(224, 224, 1)	0
Conv2D	(111, 111, 16)	160
Conv2D	(109, 109, 32)	4640
MaxPooling2D	(36, 36, 32)	0
Conv2D	(34, 34, 32)	9248
Conv2D	(32, 32, 16)	4624
GlobalMaxPooling2D	(16)	0
Reshape	(4, 4, 1)	0
Conv2DTranspose	(6, 6, 16)	160
Conv2DTranspose	(8, 8, 32)	4640
UpSampling2D	(24, 24, 32)	0
Conv2DTranspose	(26, 26, 16)	4624
Conv2DTranspose	(28, 28, 1)	4624
Flatten	(784)	0
Dense	(10)	7850
Total Parameters		36,091
Trainable Parameters		36,091

function is also utilized with our proposed architecture as this study involves multiclass classification. It measures the difference between the predicted probability distribution and the true probability distribution of the classes as shown in (8). Here, y_i presents the true probability of class i (1 if the class is the true class, 0 otherwise), and $\hat{y}_i$ represents the predicted probability of class i by the model.

$$\text{CCE Loss} = -\sum_{i} y_i \log(\hat{y}_i) \tag{8}$$

Total loss is calculated according to the (9) where CCE_j represents the categorical cross-entropy loss for the jth example in the batch.

$$\text{Total Loss} = \frac{1}{N}\sum_{j=1}^{N} \text{CCE}_j \tag{9}$$

B. MODEL QUANTIZATION

We applied the quantization technique for compressing our trained CAE with the motivation of deployment into memory-limited edge devices as described in Section III-B. In our quantization process, we first trained our regular CAE over the data set, and after reaching a satisfactory performance, we applied the quantization-aware training over the trained CAE for converting it from float32 into int8 precision representation and later fine-tuned it to preserve the best possible performance closer to the original non-quantized version as quantization comes with a trade-off between model accuracy and the size of the overall architecture. Algorithm 1 depicts the step-by-step workflow of our quantization-aware training. With the help of the Tensorflow Quantization aware training API, we performed the quantization process.

Algorithm 1: Quantization-Aware Training.

Input: Images X and Labels y
Output: Quantized Bangla Font Style Recognizer
Step 1: Initialize and compile the float32 CAE model
Step 2: Train the CAE over X, y
Step 3: Quantize the trained CAE from float32 to int8
Step 4: Fine-tune the Q-CAE till convergence
Step 5: Inference on new data with the Q-CAE

Finally, we get our quantized model with int8 weights and uint8 activations.

C. PROPOSED FRAMEWORK

In Fig. 6, the overview of our proposed Bangla visual font recognition framework is portrayed. Firstly, the lightweight CAE trained on a large collection of images containing Bangla text with various font styles. Then the trained CAE is quantized to make it more lightweight for resource-constrained edge devices and further fine-tuned to maintain a plausible performance. This quantized trained CAE can employed on resource-limited edge devices such as mobile, and tablets in the form of apps, as well as browser extensions, web apps, and desktop apps, which we'll take input an image containing Bangla text and output the name of the font style name.

VI. EXPERIMENT AND RESULT ANALYSIS

In this section, we illustrate a detailed analysis of our experiment and the outcomes, starting with the description of the experimental setup utilized for conducting our experiment and evaluating the findings through various metrics.

A. EXPERIMENTAL SETUP

All the implementations for the experiment including algorithms, auxiliary functions, evaluation, etc. have been implemented using the popular deep learning framework TensorFlow and with the help of other popular packages and libraries such as Numpy, Matplotlib, and Scikit-learn. We made use of the free resources provided by Google Colab, a cloud-based platform that provides free access to GPUs, TPUs, and other auxiliary resources.

B. EVALUATION METRICS

We have evaluated our proposed architecture by employing the 4 most common and standard metrics, namely - Accuracy, Precision, Recall, and F1-score, and the respective equations are the following -

$$Accuracy = \frac{\text{True Positive + True Negative}}{\text{Total Number of Predictions}} \tag{10}$$

$$Precision = \frac{\text{True Positive}}{\text{True Positive + False Positive}} \tag{11}$$

$$Recall = \frac{\text{True Positive}}{\text{True Positive + False Negative}} \tag{12}$$

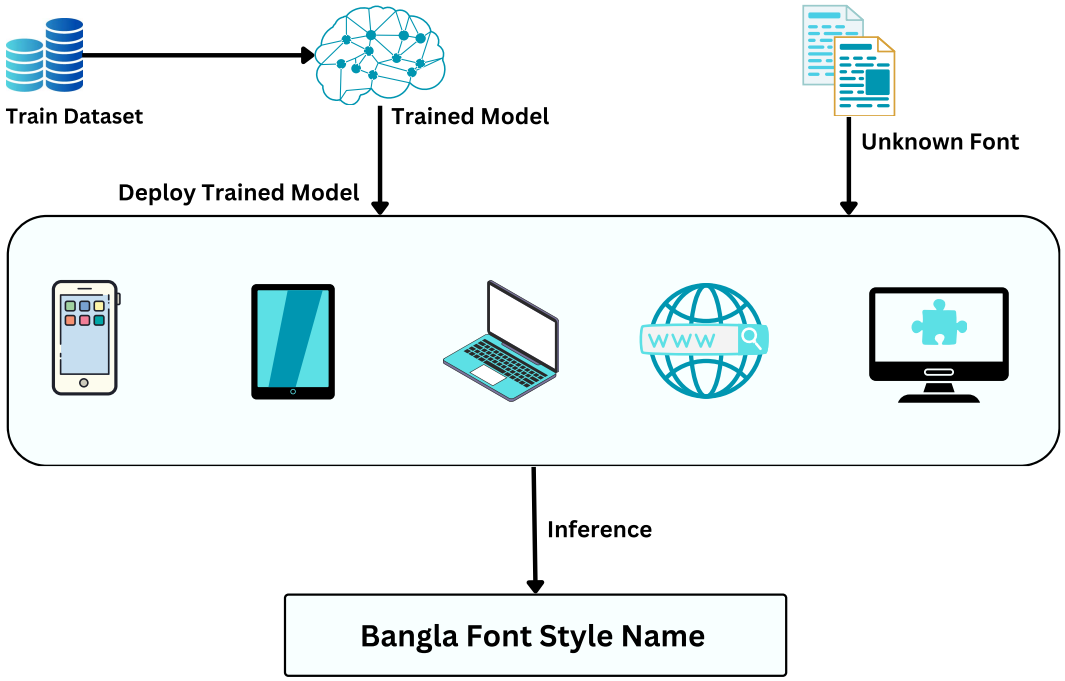


FIGURE 6. Overview of the Bangla visual font recognition system.

$$F1 - score = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{13}$$

True Positive means the model correctly predicted the positive class, and True Negative indicates that the model predicted the negative classes correctly. In addition, False Positive and False Negative both mean the model misclassified the respective classes.

C. PERFORMANCE EVALUATION

To evaluate our proposed model's performance, we start by discussing the training and validation performance in Section VI-C-1, followed by performance over the test data in Section VI-C-2, and lastly, we also discuss the performance of our model in Section VI-C-3.

1) PERFORMANCE OVER TRAIN AND VALIDATION SET

At first, we split the whole Bangla VFR dataset randomly into two subsets - train set and validation set, allocating 80% of the data for the training, and 20% for the validation, resulting in sample sizes of 48000 and 12000, respectively. Then we have trained our proposed Convolutional Auto Encoder architecture utilizing the train and validation set over 10 epochs. Fig. 7 presents the train vs. validation accuracy graph, from which we observe a smooth upward trend of the performance of our employed model over the both train and validation set. Also, as illustrated in Fig. 8, both the train and validation loss showed a smooth decline over the training epoch.

2) PERFORMANCE OVER TEST SET

To evaluate our model's performance on unseen data, we have also created a separate Test set following the same procedure

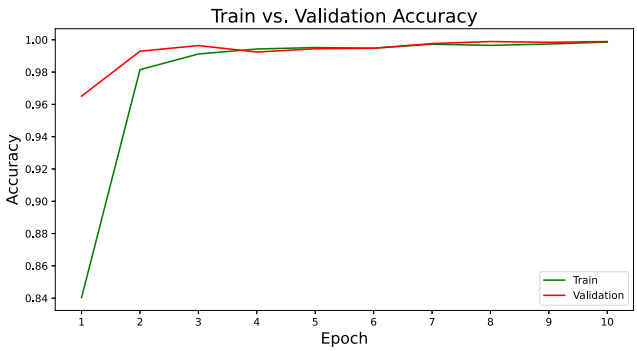


FIGURE 7. Train vs. validation accuracy of the employed convolutional autoencoder over 10 epochs.

described in Section IV. For this, we have picked 100 unique and new words. Then for each word, we have generated two instances per class with random rotation as illustrated in Fig. 4. Finally, there are 200 images per class consisting of a test set of 2,000 images in total. An in-depth classification report of our model over the test set is presented in Table 3. Since we employed a balanced dataset, both in training and testing, the accuracy metric alone is sufficient to measure the performance of our model. As our model's accuracy is satisfactory, it is obvious that performance scores on other metrics will be better as well. Here, we see that our employed CAE performs extraordinarily well on almost all the test samples and its misclassification is negligible as it only 1 misclassified only one instance from **FN Suborno Jayonti** as **FN Jagat Shonkhoneel**, illustrated in the confusion matrix in Fig. 9. Fig. 10 shows the overall accuracy score comparison over all the data subsets.

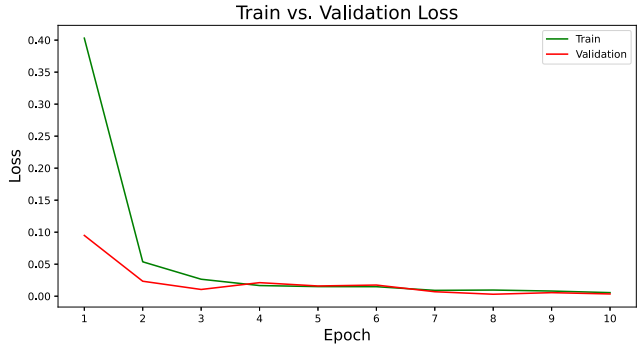


FIGURE 8. Train vs. Validation Loss of the employed Convolutional Autoencoder over 10 epochs.

TABLE 3. Classification report of the proposed convolutional autoencoder over the test set for class-wise performance evaluation

	Precision	Recall	F1-score	Support
FN Jagat Shonkhoneel	99.50	100.00	0.9975	200
FN Suborno Jayonti	100.00	99.50	0.9975	200
Li Alinur Phulkuri	100.00	100.00	1.00	200
Li Alinur Saikat	100.00	100.00	1.00	200
Li Alinur Sanghoti	100.00	100.00	1.00	200
Li Fazlay Munnisha	100.00	100.00	1.00	200
Li Mahfuz Himadri	100.00	100.00	1.00	200
Li Niladri Russian	100.00	100.00	1.00	200
Li Upohar 56	100.00	100.00	1.00	200
Accuracy			99.95	2000
Macro	99.95	99.95	0.9995	2000
Weighted	99.95	99.95	0.9995	2000

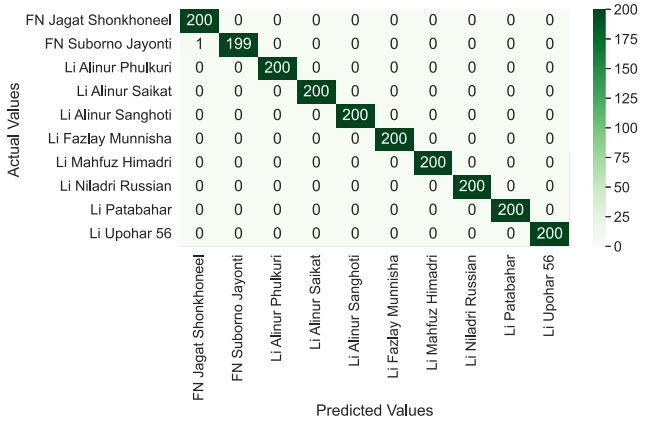


FIGURE 9. Confusion Matrix of the employed Convolutional Autoencoder over the Test set.

3) PERFORMANCE AFTER QUANTIZATION

After training and testing our CAE, we then quantized it into int8 format to make it more lightweight for resource-constraints devices. Then we fine-tuned the quantized model to maintain the performance as close as possible to the regular float64 architecture. In Fig. 11, the performance of our quantized model over the test set is depicted through the confusion matrix. In comparison to the regular model's performance, our fine-tuned model misclassified only 2 extra instances, which can be negligible considering the trade-off between model size and inference accuracy. The comparison of the test accuracy before and after quantization is shown in Fig. 12.

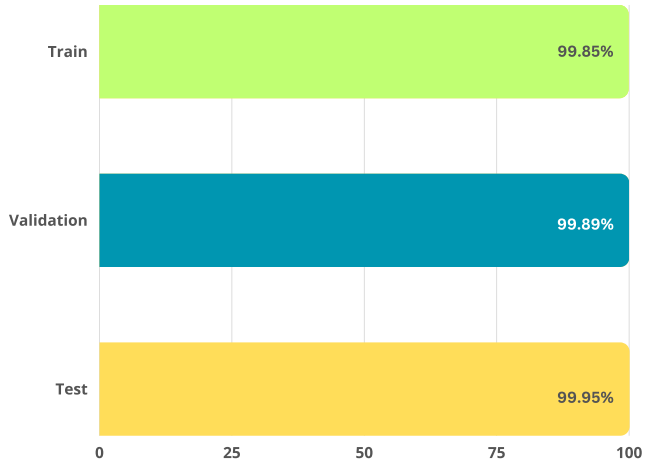


FIGURE 10. Performance of the proposed convolutional autoencoder in accuracy metric over train, validation, and test set.

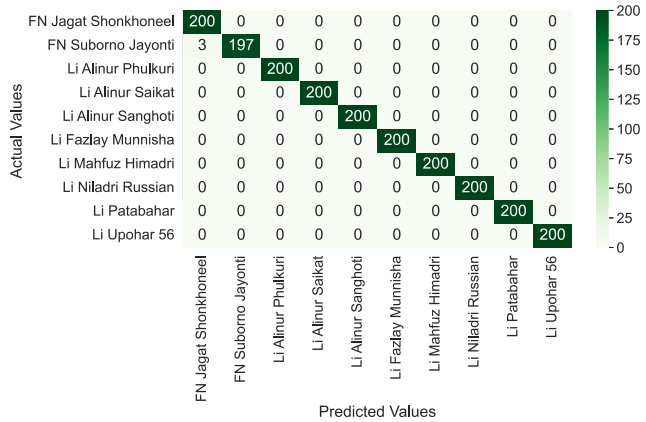


FIGURE 11. Confusion matrix of the quantized convolutional autoencoder over the test set.

TABLE 4. Comparison of Performance in Accuracy Metric Among Past Bangla Visual Font Recognition Studies

Work	Year	Method	Accuracy
Hasan et al. [4]	2022	Pre-trained VGG16	96.23%
Islam et al. [9]	2021	SCAE	98.73%
Hasan et al. [10]	2020	CNN	96.00%
Wang et al. [11]	2018	Pre-trained VGG16	97.53%
This work		CAE	99.95%
		Q-CAE	99.85%

D. PERFORMANCE COMPARISON

The performance comparison of our proposed Quantized-CAE with other related works is shown in Table 4. Our

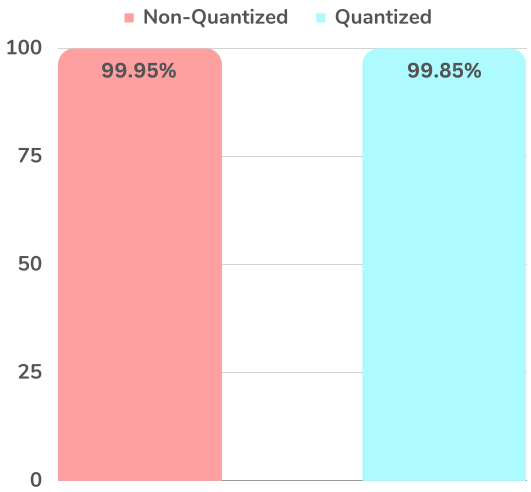


FIGURE 12. Compaision of the test accuracy of the proposed convolutional autoencoder before and after quantization.

TABLE 5. Comparison of Performance With the State-of-the-Art Baseline Models

Model	Accuracy	Weights
DenseNet121	99.15%	ImageNet
InceptionV3	95.45%	ImageNet
MobileNet	97.45%	ImageNet
MobileNetv2	96.60%	ImageNet
ResNet50	99.25%	ImageNet
VGG16	99.25%	ImageNet
VGG19	99.15%	ImageNet
Xception	90.35%	ImageNet
Proposed		
CAE	**99.95%**	
Q-CAE	**99.85%**	

proposed method secured the highest accuracy over the most recent research works, reaching 99.95% accuracy with non-quantized CAE, and 95.85% of accuracy after int8 quantization.

To demonstrate our employed model's superiority, we have also performed a baseline comparison with 8 state-of-the-art CNN models as presented in the Table 5, utilizing their pre-trained weights from the ImageNet dataset [31] and they are trained in the same environment. Our proposed Convolutional Autoencoder outperformed them by attaining higher accuracy on the test set. The comparison is illustrated using the Fig. 13.

VII. LIMITATIONS AND FUTURE WORK

The proposed Q-CAE is capable of correctly recognizing the Bangla Font styles from images containing Bangla texts. It'll be quite helpful for identifying font-style names from documents, blogs, and other sources that contain plain font-style content. However, when it comes to complex design, textual artwork, and typography, designers often tend to modify the original styles of the employed fonts such as extending certain corners of a character, squeezing sides to make it look thinner, and so on. It is also common to utilize a combination of multiple font styles in a single design i.e. attaching characters from different font styles. Recognizing font styles accurately from the above-mentioned scenarios is a tough task since in most cases, the original style isn't preserved. Furthermore, this model is not employed over 3D shapes and designs.

In future work, we aim the address the issues mentioned above for developing a robust font style recognizer that will be able to recognize fonts from designer images.

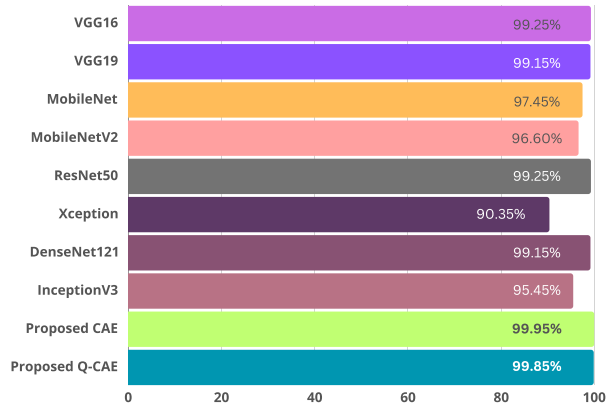


FIGURE 13. Accuracy comparison among state-of-the-art baseline models and our proposed architecture.

VIII. CONCLUSION

In this work, we proposed a lightweight Convolutional autoencoder for recognizing Bangla font styles. The quantization technique is also employed to make the proposed model further compressed and lightweight to make it suitable for deploying into resource-constrained edge devices. To conduct the study, we've created a synthetic data set consisting of a total of 60000 images of 10 distinct Bangla font styles. Our proposed model, both with and without quantization, achieve a state-of-the-art accuracy as shown in the Table 4. Our study concludes that the proposed lightweight Q-CAE can be utilized with web and mobile apps, and browser extensions as well as incorporating edge devices that will correctly recognize the font style name from the respective scenario. In future work, we aim to build a robust Bangla font style recognizer that will be able to perform well on designer images as well.

DATA AVAILABILITY

The dataset utilized in this study is publicly available in Mendeley data.

URL: https://data.mendeley.com/datasets/cnd2wh65my/1

ACKNOWLEDGMENT

The authors would like to thank the Advanced Machine Intelligence Research Lab (AMIR Lab) for resource sharing and precious opinions.

REFERENCES

[1] Z. Wang et al., "Deepfont: Identify your font from an image," in *Proc. 23rd ACM Int. Conf. Multimedia*, 2015, pp. 451–459.

[2] G. Chen et al., "Large-scale visual font recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 3598–3605.

[3] Wikipedia contributors, "Bengali language," Accessed: Oct. 10, 2023. [Online]. Available: https://en.wikipedia.org/wiki/Bengali_language

[4] S. Hasan, G. Rabbi, R. Islam, H. I. Bijoy, and A. Hakim, "Bangla font recognition using transfer learning method," in *Proc. Int. Conf. Inventive Computation Technol.*, 2022, pp. 57–62.

[5] X. Li, J. Wang, H. Zhang, Y. Huang, and H. Huang, "SwordNet: Chinese character font style recognition network," *IEEE Access*, vol. 10, pp. 8388–8398, 2022.

[6] M. Mohammadian, N. Maleki, T. Olsson, and F. Ahlgren, "Persis: A persian font recognition pipeline using convolutional neural networks," in *Proc. IEEE 12th Int. Conf. Comput. Knowl. Eng.*, 2022, pp. 196–204.

[7] C. Tensmeyer, D. Saunders, and T. Martinez, "Convolutional neural networks for font classification," in *Proc. IEEE 14th IAPR Int. Conf. document Anal. Recognit.*, 2017, pp. 985–990.

[8] D. T. Vijayakumar and M. R. Vinothkanna, "Capsule network on font style classification," *J. Artif. Intell. Capsule Netw.*, vol. 2, no. 2, pp. 64–76, 2020.

[9] M. M. Islam, A. S. A. Rabby, N. Hasan, J. Nahar, and F. Rahman, "A novel bangla font recognition approach using deep learning," in *Emerging Technologies in Data Mining and Information Security: Proceedings of IEMIS 2020*, vol. 2. Berlin, Germany: Springer, 2021, pp. 745–754.

[10] M. Zahid Hasan, K. Tanzila Rahman, R. I. Riya, K. Z. Hasan, and N. Zahan, "A CNN-based classification model for recognizing visual bengali font," in *Proc. Int. Joint Conf. Comput. Intell.*, 2020, pp. 471–482.

[11] Y. Wang, Z. Lian, Y. Tang, and J. Xiao, "Font recognition in natural images via transfer learning," in *Proc. 24th Int. Conf. MultiMedia Model.*,, 2018, pp. 229–240.

[12] X. Guo, X. Liu, E. Zhu, and J. Yin, "Deep clustering with convolutional autoencoders," in *Proc. 24th Int. Conf. Neural Inf. Process.*, 2017, pp. 373–382.

[13] L. Gondara, "Medical image denoising using convolutional denoising autoencoders," in *Proc. IEEE 16th Int. Conf. Data Mining Workshops*, 2016, pp. 241–246.

[14] M. S. R. Tusher et al., "An enhanced variational autoencoder approach for the purpose of deblurring bangla license plate images," *Int. J. Adv. Comput. Sci. Appl.*, vol. 14, no. 6, pp. 10–14569, 2023.

[15] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto, "Deep convolutional autoencoder-based lossy image compression," in *Proc. IEEE Picture Coding Symp.*, 2018, pp. 253–257.

[16] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto, "Energy compaction-based image compression using convolutional autoencoder," *IEEE Trans. Multimedia*, vol. 22, pp. 860–873, 2020.

[17] S. Ryu, H. Choi, H. Lee, and H. Kim, "Convolutional autoencoder based feature extraction and clustering for customer load analysis," *IEEE Trans. Power Syst.*, vol. 35, no. 2, pp. 1048–1060, Mar. 2020.

[18] K. Ghasedi Dizaji, A. Herandi, C. Deng, W. Cai, and H. Huang, "Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 5736–5745.

[19] J. K. Chow, Z. Su, J. Wu, P. S. Tan, X. Mao, and Y.-H. Wang, "Anomaly detection of defects on concrete structures with the convolutional autoencoder," *Adv. Eng. Informat.*, vol. 45, 2020, Art. no. 101105.

[20] L. Chen, Q. Guan, B. Feng, H. Yue, J. Wang, and F. Zhang, "A multi-convolutional autoencoder approach to multivariate geochemical anomaly recognition," *Minerals*, vol. 9, no. 5, 2019, Art. no. 270.

[21] O. E. David and N. S. Netanyahu, "Deeppainter: Painter classification using deep convolutional autoencoders," in *Proc. Artif. Neural Netw. Mach. Learn.: 25th Int. Conf. Artif. Neural Netw.*, 2016, pp. 20–28.

[22] S. Karimpouli and P. Tahmasebi, "Segmentation of digital rock images using deep convolutional autoencoder networks," *Comput. Geosci.*, vol. 126, pp. 142–150, 2019.

[23] J.-C. See, H.-F. Ng, H.-K. Tan, J.-J. Chang, W.-K. Lee, and S. O. Hwang, "Doubleqext: Hardware and memory efficient CNN through two levels of quantization," *IEEE Access*, vol. 9, pp. 169082–169091, 2021.

[24] N.-D. Ho and I.-J. Chang, "O-2A: Outlier-aware compression for 8-bit post-training quantization model," *IEEE Access*, vol. 11, pp. 95467–95480, 2023.

[25] Y. M. Kim, K. Han, W.-K. Lee, H. J. Chang, and S. O. Hwang, "Non-zero grid for accurate 2-bit additive power-of-two CNN quantization," *IEEE Access*, vol. 11, pp. 32051–32060, 2023.

[26] A. F. Rakib, R. Rahman, A. A. Razi, and A. T. Hasan, "A lightweight quantized CNN model for plant disease recognition," *Arabian J. Sci. Eng.*, vol. 49, pp. 1–12, 2023.

[27] Shazol, "Bangla wikipedia corpus," kaggle, Accessed: Aug. 18, 2023. [Online]. Available: com/datasets/shazol/bangla-wikipedia-corpus

[28] Lipighor, "Lipighor - free bangla font," Accessed: Aug. 20, 2023. [Online]. Available: https://lipighor.com

[29] FontBD, "Fontbd - free bangla font," Accessed: Aug. 22, 2023. [Online]. Available: https://fontbd.com

[30] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[31] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.

MOSHIUR RAHMAN TONMOY received the bachelor's degree in computer science and engineering from the University of Asia Pacific, Dhaka, Bangladesh, in 2023. Throughout his undergraduate journey, he showcased a profound interest in competitive programming and programming contests, nurturing his critical and logical thinking abilities. His research journey commenced with a collaboration with the Institute of Automation Research and Engineering, where he actively participated in innovative research initiatives. Since 2023, he has been a Research Assistant with Advanced Machine Intelligence Research Lab, Dhaka. His research interests include deep learning, computer vision, and federated learning, and their interdisciplinary applications aimed at enhancing societal well-being and benefiting humanity.

ABDUL FATTAH RAKIB received the B.Sc. degree in computer science and engineering from the University of Asia Pacific, Dhaka, Bangladesh, in 2022. He was a Research Intern with the Institute of Automation Research and Engineering, where he enriched his expertise in the field of artificial intelligence. He has authored or coauthored research articles in prestigious conferences and journals. His research interests include data science, computer vision, NLP, and healthcare informatics.

RASHIK RAHMAN received the bachelor's degree in computer science and engineering from the University of Asia Pacific, Dhaka, Bangladesh, in 2021. He was an AI Engineer with Quantum.AI. He joined SammTech as a Data Scientist. He is currently a full-time Faculty Member with the Department of CSE, University of Asia Pacific, Dhaka, Bangladesh. His research interests include but are not limited to computer vision, NLP, TinyML, and federated learning.

MD. AKHTARUZZAMAN ADNAN received the bachelor's degree in computer science and information technology from the Islamic University of Technology (IUT), Gazipur, Bangladesh and the master's degree in computer science and engineering from the University Teknologi Malaysia, Iskandar Puteri, Malaysia, showcasing his commitment to advancing his knowledge in the field. He is currently working toward the Ph.D. degree with Macquarie University, Sydney, NSW, Australia. He is a dedicated and accomplished Researcher. He is also an Assistant Professor with the Department of Computer Science and Engineering, University of Asia Pacific, Dhaka, Bangladesh. His early education laid the foundation for his passion for technology and its applications. Over the years, he has demonstrated a strong academic aptitude and a keen interest in the evolving domains of artificial intelligence, machine learning, deep learning, and natural language processing. He was a Reviewer for several journals and international conferences.

M. F. MRIDHA (Senior Member, IEEE) received the Ph.D. degree in AI/ML from Jahangirnagar University, Dhaka, Bangladesh, in 2017. From 2019 to 2022, he was an Associate Professor and the Chairperson of the Department of Computer Science and Engineering, Bangladesh University of Business and Technology, Dhaka. From 2012 to 2019, he was a CSE Department Faculty Member with the University of Asia Pacific, Dhaka, and the Graduate Head. He is currently an Associate Professor with the Department of Computer Science, American International University Bangladesh, Dhaka. For more than ten years, he has been with the master's and bachelor's students as a supervisor of their thesis work. His research experience, within both academia and industry results in more than 150 journals and conference publications. His research work contributed to the reputed journals of *Scientific Reports* (Nature), *Knowledge-Based Systems*, *Artificial Intelligence Review*, *Engineering Applications of Artificial Intelligence*, IEEE ACCESS, *Sensors*, *Cancers*, *Biology*, and *Applied Sciences*. His research interests include artificial intelligence, machine learning, deep learning, natural language processing, and Big Data analysis. He is also the Founder and Director of the Advanced Machine Intelligence Research Laboratory. He was a program committee Member for several international conferences/workshops and an Editorial Board Member for several journals, including *PLOS ONE journal*. He was a Reviewer for reputed journals, such as IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, *Artificial Intelligence Review*, IEEE ACCESS, *Knowledge-Based System*, *Expert System*, *Bioinformatics*, Springer, *Nature*, and MDPI, and conferences, such as ICCIT, HONET, ICIEV, IJCCI, ICAEE, ICCAIE, ICSIPA, SCORED, ISIEA, APACE, ICOS, ISCAIE, BEIAC, ISWTA, IC3e, ISWTA, CoAST, icIVPR, ICSCT, 3ICT, and DATA21.

JIE HUANG received the B.Eng. degree from the Nagoya Institute of Technology, Nagoya, Japan, in 1985, and the M. Eng. and D. Eng. degrees from Nagoya University, Nagoya, in 1987 and 1991, respectively. From 1992 to 1993, he was a System Engineer with Phenix Data Corp, Indianapolis, IN, USA. From 1993 to 1998, he was a Frontier Research Scientist with the Bio-Mimetic Control Research Center, under the Institute of Physical and Chemical Research (RIKEN). From 1998 to 2002, he was an Assistant Professor with the University of Aizu, Aizuwakamatsu, Japan. Since 2002, he has been an Associate Professor with the School of Computer Science and Engineering, University of Aizu. His research interests include sound signal processing, human audition, and robot auditory sensing systems.

JUNGPIL SHIN (Senior Member, IEEE) received the B.Sc. degree in computer science and statistics and the M.Sc. degree in computer science from Pusan National University, Busan, South Korea, in 1990 and 1994, respectively, and the Ph.D. degree in computer science and communication engineering from Kyushu University, Fukuoka, Japan, in 1999, under a scholarship from the Japanese Government (MEXT). He was an Associate Professor, a Senior Associate Professor, and Full Professor with the School of Computer Science and Engineering, University of Aizu, Aizuwakamatsu, Japan, in 1999, 2004, and 2019, respectively. He has coauthored more than 350 published papers for widely cited journals and conferences. His research interests include pattern recognition, image processing, computer vision, machine learning, human-computer interaction, non-touch interfaces, human gesture recognition, automatic control, Parkinson's disease diagnosis, ADHD diagnosis, user authentication, machine intelligence, bioinformatics, and handwriting analysis, recognition, and synthesis. He is a Member of ACM, IEICE, IPSJ, KISS, and KIPS. He was the program Chair and a program committee member for numerous international conferences. He is the Editor of IEEE journals, Springer, Sage, Taylor & Francis, MDPI Sensors and Electronics, and Tech Science. He is a reviewer for several major IEEE and SCI journals.