Deep Learning Networks for Handwritten Bangla Character Recognition

Halima Begum, Muhammed Mazharul Islam, Humaira Sabira Eva, Naim Hossain Emon, Farhan Ahmed Siddique

***Abstract*—In recent years, deep convolutional networks (DCNN) have gained popularity for different classification (or recognition) tasks. In this paper, three well known DCNN structures were used, i.e., AlexNet, SqueezeNet and GoogLeNet, and their classification performances in recognizing handwritten Bangla isolated characters were compared. These networks have simpler structures compared to the recent DCNNs. Experiments on a standard Bangla database revealed that the overall performance of GoogLeNet is slightly better than the other two networks. Further analysis using saliency maps of the test samples revealed the important features that are learned by the networks for classifying characters. This information led us to understand why classification of some samples fail and how to rectify these.**

***Index Terms*—Bangla handwriting recognition, transfer learning, AlexNet, GoogLeNet, SqueezeNet, Grad-CAM, Guided Grad-CAM, F_1 score, precision, recall.**

I. Introduction

DEVELOPING effective handwritten character recognition (or classification) systems have been an active area of research for the last few decades [1]. The digitalization of Bangla handwritten scripts will pave the way to preserve a large number of old handwritten legal documents and books in digital formats, as well as make it possible to retrieve information from hand written documents (e.g., filled-out forms, postal addresses etc.). Regrettably, although Bangla is the native language of about 230 million people in the world [2], the research on Bangla handwritten character recognition is comparatively new.

The Bangla script consists of 11 vowels, 39 consonants and 10 numerals in addition to many compound characters (created by the combination of consonants), and a number of vowel-modifiers and consonant-modifiers. The classification of Bangla handwritten characters has proven to be a challenging task due to the presence of several characters that have strikingly similar shapes. The only distinguishing feature among these shapes are simply the presence or absence of either a dot, or a short curved (or straight) line. The similarity in shapes, coupled with the significant variation in handwriting styles among individuals, adds an extra layer of complexity to the classification of these characters.

In recent years, deep convolutional neural networks (DCNNs) have shown promising results in recognizing handwritten characters of different languages from scanned images [3], [4]. In contrast to the 'hard-coded-feature' based systems, the DCNNs can learn the features by themselves from the given data [2]. For the solution of a particular problem using DCNN, a decisive factor for the researchers is whether to select a well-known pre-trained deep neural network (and use the transfer learning approach), or to create a new network structure. In general, the transfer learning approach has demonstrated its effectiveness in reducing training time significantly. [5].

Despite the typically superior performance of DCNNs in classification tasks overall, they may fail to accurately predict the correct classes for some specimens. There could be many reasons why a DCNN classification system is unable to detect a sample of a handwritten character properly. Identifying the causes of the failure (e.g., whether it is due to an ambiguous handwriting, or an incorrectly labeled test sample, or the network's inherent weaknesses in identifying features from samples etc.) is important because, this information provides insight into understanding how to make the overall character recognition (or classification) system robust.

In this paper, the performances of three well-known but comparatively simple DCNNs, i.e., AlexNet [6], SqueezNet [7], and GoogLeNet [8], were explored in recognizing handwritten isolated Bangla characters from images. A standard Bangla database was used for this purpose. To reduce the training time, the transfer learning approach was adopted, and the performances of the trained networks were compared using a standard metric. Moreover, saliency maps [9] [10] of the test samples were investigated to get an insight into what features of an input image are deemed important by a network while making the decision about that sample's class.

II. Literature Review

Researchers have proposed various approaches to solve the problems related to Bangla handwritten character

Manuscript received December 31, 2022; revised September 1, 2023.

H. Begum is an Assistant Professor at the Department of Electrical and Electronic Engineering, East West University, A/2, Aftabnagar, Dhaka, Bangladesh, (e-mail: hab@ewubd.edu, phone number: +88-09666775577, ext:186).

M. M. Islam is an Assistant Professor at the Department of Electrical and Electronic Engineering, East West University, A/2, Aftabnagar, Dhaka, Bangladesh, (e-mail: mmai@ewubd.edu).

H. S. Eva is a PhD Student at the Department of Electrical Engineering and Computer Science, South Dakota State University, United States, (e-mail: HumairaSabira.Eva@jacks.sdstate.edu).

N. H. Emon is an Assistant Engineer, Engineering and Design Department, Reverie Power and Automation Engineering Limited, Bangladesh, (e-mail: naim.emon@reverie-bd.com).

F. A. Siddique is a Graduate Student, Master's Program in Smart System Integrated Solutions, Aalto University, Finland, (email:farhanahmed.siddique@aalto.fi)

recognition. From the 'hard-coded-feature' extraction and recognition methods to modern day artificial neural network approaches - every research was driven by the need for a robust recognition system.

Liu et al. (2009) [11] proposed a gray-scale normalization and gradient direction histogram framework for the numeral character recognition with recognition accuracy of 99.40% on the 'ISI Bangla numeral dataset' [12].

Bag et al. (2014) [13] extracted topological features, i.e., structural convexity from skeletal representation of the Bangla compound characters and achieved a detection rate of 96.17% on the handwritten samples.

Alom et al. (2018) [14] used a machine learning based approach to recognize Bangla optical characters. They compared the performances of several established architectures, i.e., All-Conv, ResNet, Fractal Net, NiN, VGG-16 and DenseNet. They trained and tested these networks separately for Bangla numerals, alphabets, and special characters using the 'CMATERdb' database [15]. Among all the frameworks they used, DenseNet produced the highest accuracy of 99.13% on digit recognition, 98.31% on handwritten alphabet recognition, and 98.18% on special character recognition.

Maitra et al. (2015) [16] proposed to use a convolutional neural network (CNN) only as a feature extractor and used separate SVMs to train on the extracted features for the classification. Their trained network could identify 98.375% of Bangla numerals and 95.6% of Bangla basic characters.

Chowdhury, R. R., Hossain et al. (2019) [17] trained a CNN with the 'BanglaLekha-Isolated' database. They worked only on the Bangla alphabets (i.e., only 50 character classes), and obtained an accuracy of 91.81%. Later, they applied data augmentation techniques to expand the number of samples to $200,000$. After training the model with these augmented samples, the recognition performance improved to 95.25%.

Hazra et al. (2021) [18] proposed a novel CNN architecture having four layers, i.e., a convolutional layer, a nonlinear activation layer, a pooling layer, and a fully connected layer. They used 'CMATERdb' and 'ISI Bangla' database to train their network. In this study, they experimented with different batch sizes and optimization techniques to find out the best possible combination.

It is observed that some researchers introduced novel networks while other used well-known, pre-trained standard networks. Although they presented the overall recognition accuracies of their systems, any analysis on the recognition failures (of the samples of handwritten characters) is missing.

III. Description of the Methodology

Three DCNN (i.e., AlexNet, SqueezeNet, and GoogLeNet) were used in this paper for the classification task. All three networks were already pre-trained on the ImageNet database, i.e., these networks already had assigned weights (for classifying images) from the ImageNet challenge [19].

However, the last two layers of the networks were changed to match with the number of Bangla character classes in the database used here. This approach was adopted because, the initial layers of a CNN are already trained as filters to extract low level features (e.g., the presence of horizontal, slanted or vertical lines from images etc.). These features are common for handwritten characters as well. This approach reduces the training time of the networks for the Bangla character dataset significantly.

This section (section III) is divided into four subsections. Section III-A contains the description of the standard database (used in this study) and the preprocessing of the samples (to get these ready for the networks). In section III-B, the structures of the three networks are briefly discussed. Section III-C describes the training procedure of these networks.

After training the networks, the classification performances are evaluated using the test samples. To gain a deeper insight into the learning behavior or patterns of neural networks, saliency maps are often utilised. These maps are valuable visualizations tools that reveal the specific regions or features of the input data that influence the networks' decision-making process. Analysis of the saliency maps can therefore assist us in understanding how a network learns and which areas of the input data are the most important for its class predictions. The saliency maps used in this paper are explained in section III-D.

A. Description of the Standard Database and Preprocessing Steps

It is imperative to train any DCNN with a standard database that was prepared considering the high degree of variability in the handwriting styles. To the best of our knowledge, 'ISI' [12], 'CMATERdb' [15], and 'BanglaLekha-Isolated'[20] are the only three freely available standard databases on Bangla isolated handwritten characters.

In the case of 'ISI' database, only a small subset of *Bangla numerals* is publicly accessible, not the full dataset. On the other hand, the 'CMATERdb' has fewer samples than 'BanglaLetha-Isolated' database [21]. The expected variability in the handwritten characters (in terms of writing styles) is higher in the 'BanglaLekha-Isolated' database due to the comparatively larger number of samples. Considering these facts, the 'BanglaLekha-Isolated' database was used in this paper.

This database contains scanned images of isolated Bangla (handwritten) characters. The total number of character classes in the database is 84, which consists of 50 basic Bangla characters (vowels and consonants), 10 Bangla numerals, and 24 compound characters. A list of all the characters in the database is shown in Table I against its assigned class numbers. Classes C01 to C11 represent the vowels অ to ঔ, classes C12 to C50 represent the consonants ক to ঁ, and classes C51 to C60 represent the Bangla digits ০ to ৯. The rest of the classes (i.e., classes C61 to C84) represent the compound characters. It should be mentioned that, the list of compound characters found in this database is non-exhaustive, and there are other compound characters

TABLE I
LIST OF ALL CHARACTERS IN THE DATABASE

Class	Character	Class	Character	Class	Character
C01	অ	C29	দ	C57	৬
C02	আ	C30	ধ	C58	৭
C03	ই	C31	ন	C59	৮
C04	ঈ	C32	প	C60	৯
C05	উ	C33	ফ	C61	ক্ষ
C06	ঊ	C34	ব	C62	ব্দ
C07	ঋ	C35	ভ	C63	ঙ্গ
C08	এ	C36	ম	C64	স্ক
C09	ঐ	C37	য	C65	স্ফ
C10	ও	C38	র	C66	স্থ
C11	ঔ	C39	ল	C67	চ্ছ
C12	ক	C40	শ	C68	ক্ত
C13	খ	C41	ষ	C69	স্ন
C14	গ	C42	স	C70	ঞ্চ
C15	ঘ	C43	হ	C71	স্প
C16	ঙ	C44	ড়	C72	হ্ম
C17	চ	C45	ঢ়	C73	প্ত
C18	ছ	C46	য়	C74	ম্ব
C19	জ	C47	ৎ	C75	ন্ড
C20	ঝ	C48	ং	C76	ঙ্ক
C21	ঞ	C49	ঃ	C77	থ্ব
C22	ট	C50	ঁ	C78	ষ্ঠ
C23	ঠ	C51	০	C79	হ্ল
C24	ড	C52	১	C80	ষ্প
C25	ঢ	C53	২	C81	ন্দ
C26	ণ	C54	৩	C82	ঙ্খ
C27	ত	C55	৪	C83	হ্ম
C28	থ	C56	৫	C84	ষ্ঠ

Fig. 1. Samples from the dataset indicating varying image sizes (W × H shown in pixels).

in the Bangla language. However, these 24 compound characters are the ones commonly used [20].

There are about 2000 samples for each of the 84 character classes in this database. These samples are gray scale images, and they are not size-normalized. The sample with the smallest dimension (belonging to class C51, the Bangla numeral '০') had a size of 29×29 pixels, while the sample with the largest dimensions (belonging to class C78, the Bangla compound character 'ষ্ঠ') had a size of 359×359 pixels. Fig.1 demonstrates a few samples of different sizes from the database.

Through visual inspection, some samples in this database were found to be corrupt (i.e., distorted). The distortion originated during the scanning process while building the database. These distorted images were discarded from the database because the represented characters were barely recognizable by human eyes.

To avoid bias in the classification, an equal number of samples was used for each character class (i.e., 1800 samples per class). This resulted in a total $1800 \times 84 = 151,200$ number of samples in the database. To prepare the test and training datasets, 360 samples per class were randomly selected as our test set, and the rest of the samples were used to train the networks (with validation). After training, these test samples were used to evaluate the recognition (or classification) performance of

TABLE II
SUMMARY OF DATABASE

	Per class	Classes	Total
Training samples	1,440	84	120,960
Test samples	360	84	30,240
Total samples	1,800	84	511,200

TABLE III
IMAGE SIZES FOR THE NETWORKS

Input image size (pixels)	AlexNet	SqueezeNet	GoogLeNet
	227 × 227	227 × 227	224 × 224

the three networks. To make sure that the classification performances of the three networks are comparable, the same test set, as well as the same training set, was used for all networks. Table II contains the summary of the dataset.

Different convolutional networks have different requirements for the input image sizes. For example, for both the AlexNet and SqueezeNet networks, the input image size must be 227 × 227 pixels, while the size requirement is 224×224 pixels for GoogLeNet. Therefore, the character images (of different sizes) in the original database were resized according to the requirements of each network. For resizing, the bicubic interpolation method was used, where the weighted average of the nearest 16 square pixels in the neighborhood was used [22]. Table III shows the required input image sizes for the three networks.

B. Brief Description of the Networks

AlexNet:

This was the first famous deep convolutional neural network (DCNN) and was designed by the SuperVision group [6]. It has a total of 5 convolutional layers and 3 fully connected layers. For the first layer, they used 96 filters, and the size of each filter was 11 × 11 × 3. Max pooling layers follow the initial two convolution layers. The third, fourth and fifth convolution layers are directly connected, and another max pooling layer follows the fifth convolution layer, whose output passes into a sequence of two fully connected layers. The second fully connected layer feeds into an 84 class (label) softmax classifier. A ReLu activation function is applied after each convolution layer as well as each fully connected layer.

GoogLeNet:

The GoogLeNet [8] is a 22-layer DCNN, which is structurally different from AlexNet. The use of 1 by 1 convolution layers and global average pooling layers have enabled it to create deeper architecture by minimizing computational costs. To better handle images at different scales, inception layers with convolution filters of different sizes were introduced in this network. To mitigate the vanishing gradient problem as well as to provide regularization, the inception architecture uses some intermediate classifier branches during training.

SqueezeNet:

The SqueezeNet is a DCNN developed by Iandola et. al [7], which aims to provide AlexNet level accuracy, while using 50 times fewer number of parameters. The heart of the SqueezeNet architecture is the 'Fire' module. A fire module consists of a 'squeeze convolution layer' followed by an 'expand convolutional layer'. A 'squeeze convolution layer' has filters of size 1 by 1, whereas an 'expand convolutional layer' has filters of two different sizes: 1 by 1, and 3 by 3. The SqueezeNet architecture has adopted three strategies to reduce the number of parameters or computational costs, i.e., use of 1 by 1 filters instead of 3 by 3 filters, a decrease in the number of input channels in the first convolutional layer (which further reduces the number of parameters in the network), and late 'down sampling' to retain more spatial information in the activation maps or feature maps.

C. Training of the Networks

The training and testing were performed using a personal computer with an Intel® Xeon® (E3-1231 v3) 3.40 GHz quad-core processor, 16 Gigabytes of RAM (of which 15.8 Gigabytes were usable), and 64 bit Windows operating system as the host. No graphical processing units (GPU) were used for computing.

For transfer learning, the last two layers of the pre-trained networks were replaced. The size of the output layer (i.e., the last layer) was changed to 84 to match the number of classes. The training was performed for 5 epochs, and the mini batch size was set to 64. 'Stochastic Gradient Descent with Momentum (SGDM)' was used to update the weights [23].

D. Saliency Map Generation

A visual inspection of a network's feature maps (that are generated from the input images) are often used to understand the reasons behind the network's decisions in discriminating classes. However, it was found that feature maps are not very intuitive as a visual representation since the maps are not focused on the important regions of the input image. Therefore, researchers have proposed other techniques such as Grad-CAM, Guided Grad-CAM [9] [10] to produce better visualization of the important parts used for classification decisions.

Grad-CAM: Gradient-weighted class activation map (or Grad-CAM for short) is a popular technique used for neural networks to view the local regions responsible for class discriminations [9]. This is one kind of post-hoc attention technique, and is applicable to an already trained model. The Grad-CAM technique generates localization heat-maps for an image that indicate which part of the image is emphasized by the network for its decision making.

The localization technique of Grad-CAM is "class discriminative", i.e., it focuses on what distinguishes classes, and therefore, helps to better understand the networks. To get the localization maps, the gradients that are input to the last layer of a convolutional neural network (CNN) are used.

For a particular class c, the gradients of the class score y^c is computed with respect to the feature maps of the convolutional layer (i.e., A^k). The weight for each neuron is then obtained by applying a global average pooling (i.e., GAP) of this gradient (i.e., $\frac{\partial y^c}{\partial A^k}$) over the size of the layer, i.e.,

$$\alpha_k^c = GAP\left(\frac{\partial y^c}{\partial A^k}\right) \tag{1}$$

The localization maps are then obtained by a linear combination of the feature maps scaled by the neuron weights, i.e.,

$$L^c = ReLU\left(\sum_k \alpha_k^c A^k\right) \tag{2}$$

where, L^c is the localization map (same size as the last convolutional layer), α_k^c is the weight of the neurons (for the class c and the k-th convolutional layer), and A^k is the activation map of the k-th convolutional layer. The $ReLU$ function is used to remove the non-positive values (as these values possibly belong to other classes). The final heatmap L_U^c is obtained by mapping this localization map L^c on to the original image (so that the important portions that are relevant for the class c can be understood more clearly).

Guided Grad-CAM: This is the combination of two techniques, i.e., the Guided Backpropagation technique, and the Grad-CAM technique [9] [10]. To be more specific, first the Guided Backpropagation heatmap (or saliency map) of a sample is generated, and then, the Guided Grad-CAM map of that sample is obtained by first doing a bicubic upsampling of the Grad-CAM localization map to the size of the image, and then performing a Hadamard product [24] with the Guided Backpropagation heatmap, i.e.,

$$M^c = L_U^c \odot H^c \tag{3}$$

Here, M^c is the Guided Grad-CAM map, L_U^c is the Grad-CAM heatmap (upscaled to the image size), $\odot$ denotes the Hadamard product, and H^c is the saliency map from Guided Backpropagation.

To get the Guided Backpropagation saliency map, the gradient of the class specific neurons with respect to the input image are calculated through backpropagation. However, during the backpropagation, whenever passing a non-linearity (i.e., $ReLU$ function), the negative values of the gradient are masked out, thereby removing the effect of the negative gradients with an additional "guidance". The idea is to remove the neurons which do not increase the activation of the higher layers. This results in a gradient with respect to the image (i.e., a gradient attribution map) highlighting the areas of interest for a network. Here, the calculated gradients are not the actual ones, but are the ones obtained by data substitution (i.e., imputed gradients). The operation of the modified $ReLU$ function during the Guided Backpropagation can be expressed as, $R^l = \max(0, sgn(f^l)) \cdot \max(0, sgn(R^l)) \cdot R^{l+1}$. Here, f^l is the activation map of the l-th layer in the forward pass, and R^{l+1} is the gradient of the $(l+1)$-th layer during backpropagation. Also, $sgn(\cdot)$ is the signum (or sign) function, and can be expressed as,

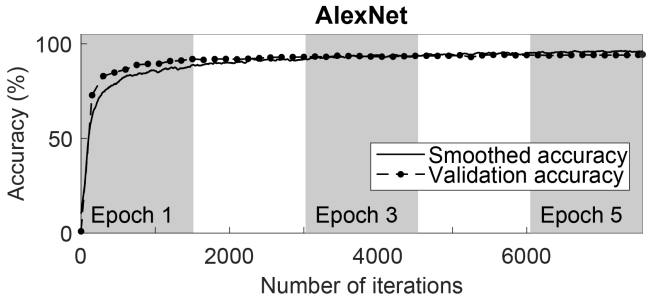


Fig. 2. Accuracy vs. no. of iterations during training for AlexNet

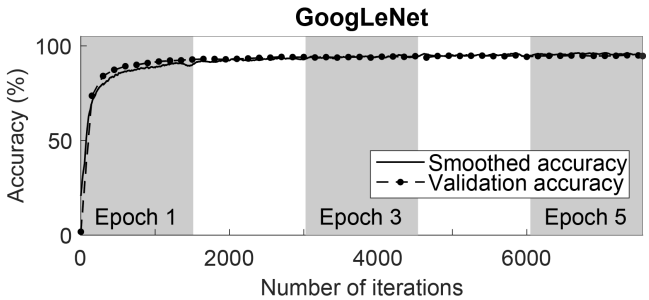


Fig. 3. Accuracy vs. no. of iterations during training for GoogLeNet

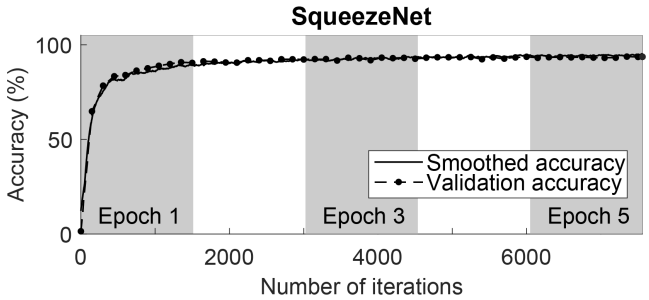


Fig. 4. Accuracy vs. no. of iterations during training for SqueezeNet.

$sgn(f^l) = 2H(f^l) - 1$. The Heaviside function $H(f^l)$ can be expressed using the Iverson brackets, i.e., $H(f^l) = [f^l > 0]$.

Eventually, the max$(\cdot)$ function mentioned above replaces each element in the activation map and the gradients with either 0 (if the i-th element value is $f_i^l < 0$ or $R_i^{l+1} < 0$), or with 1 (if the i-th element value is $f_i^l \geq 0$ or $R_i^{l+1} \geq 0$). The Guided Grad-CAM produces a saliency map at a more finer pixel level, and thus is used to help produce a sharper localization by modifying the Grad-CAM heatmap.

IV. Results and Analysis

The three networks were trained (with validation) over the training samples as described in section III-C. The accuracy plots for the three networks are shown in Fig.2, 3, and 4. The smoothed training accuracy as well as the validation accuracy follows an increasing trend until a peak accuracy is achieved.

A. Classification Performance of the Networks

After training each network, the samples of the test set (i.e., a total of $360 \times 84 = 30,240$ number of samples) (section III-C) were passed to each of the networks to predict their classes. During the classification process, a network calculates the class-specific probabilities of a test sample. The predicted class for a test sample is the one which has the highest assigned probability.

For each network, the classified test samples can be categorized into four groups: true positives (TPc), false positives (FPc), true negatives (TNc), and false negatives (FNc). For samples of class c, the true positives (TPc) represent the number of samples correctly classified as belonging to the class c, the false positives (FPc) indicate the number of samples from the other classes that are incorrectly classified as belonging to the class c, the false negatives (FNc) represent the number of samples (from the class c) that are misclassified as belonging to the other classes. Lastly, the true negatives (TNc) represent the number of samples from the other classes that are correctly classified as not belonging to the class c.

The "precision" and "recall" scores of a particular class provide us with important insight about the network's classification performance. The metrics are in fact calculated from TP_c, TN_c, FP_c, and FN_c. For the class c, the precision P_c and R_c are calculated as,

$$P_c = \frac{TP_c}{TP_c + FP_c}, R_c = \frac{TP_c}{TP_c + FN_c} \tag{4}$$

The precision and recall scores of a class lie within the range $[0, 1]$. A class-specific high precision value indicates that the network classifies most of the samples of that class correctly, and also does not produce too many misclassifications *into that class* (i.e., a higher TP_c, and a lower FP_c). A high recall value, on the other hand, indicates a higher rate of correct classification *for that particular class* (note: in our case, $TP_c + FN_c = 360$).

However, instead of using P_c and R_c, a single metric called the F_1 score, which harmonizes the precision and recall scores, is usually used [25] [26]. The F_1 score combines these parameters while providing equal weightage to the importance of both. For a class c, the F_1 score is expressed as,

$$F_{1c} = \frac{2(P_c \times R_c)}{P_c + R_c} \tag{5}$$

The F_{1c} score ranges from 0 to 1, and a class score closer to 1 represents a better *correct* classification of samples into that class (along with a lower number of misclassifications). A poor score would occur if either precision, or recall is (or both are) poor.

Table IV shows the F_1 scores (calculated from the classification results of the test data set) of the three networks for all the 84 classes. As mentioned earlier, for each class, a network with a higher score implies that it performed a better classification of the test samples for that class.

Fig.5 represents the histogram of the F_1 scores shown in Table IV. In the histogram, the horizontal axis represents the F_1 scores with bin sizes of 0.05, while the vertical axis represents the number of classes with a score falling within a bin (i.e., the freuqncy of occurrances). The figures clearly illustrate that both AlexNet (AN) and SqueezeNet (SN) have 45 occurrences (classes) where the

TABLE IV
PER-CLASS F_1 SCORES FOR ALEXNET, GOOGLENET, AND SQUEEZENET

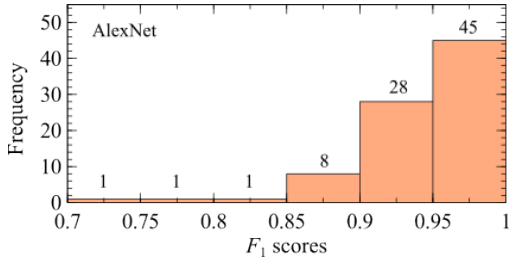
Cls	AN	GN	SN	Cls	AN	GN	SN
C01	0.96	0.97	0.97	C43	0.95	0.92	0.89
C02	0.98	0.98	0.99	C44	0.98	0.99	0.98
C03	0.96	0.97	0.95	C45	0.98	0.98	0.98
C04	0.94	0.96	0.95	C46	0.97	0.97	0.96
C05	0.93	0.94	0.92	C47	0.98	0.98	0.98
C06	0.95	0.96	0.93	C48	0.98	0.97	0.99
C07	0.96	0.97	0.95	C49	0.99	0.99	0.99
C08	0.96	0.98	0.97	C50	0.98	0.98	0.98
C09	0.96	0.98	0.97	C51	0.98	0.95	0.98
C10	0.92	0.90	0.93	C52	0.97	0.95	0.96
C11	0.96	0.96	0.97	C53	0.97	0.93	0.95
C12	0.96	0.95	0.94	C54	0.93	0.94	0.93
C13	0.92	0.93	0.89	C55	0.98	0.98	0.98
C14	0.95	0.97	0.94	C56	0.94	0.95	0.92
C15	0.90	0.90	0.87	C57	0.94	0.95	0.94
C16	0.95	0.95	0.94	C58	0.98	0.98	0.98
C17	0.95	0.93	0.94	C59	0.98	0.99	0.98
C18	0.96	0.96	0.95	C60	0.95	0.92	0.94
C19	0.96	0.96	0.96	C61	0.79	0.79	0.79
C20	0.97	0.97	0.96	C62	0.98	0.98	0.98
C21	0.98	0.98	0.96	C63	0.93	0.93	0.93
C22	0.97	0.97	0.97	C64	0.87	0.90	0.88
C23	0.92	0.94	0.91	C65	0.86	0.91	0.88
C24	0.87	0.87	0.86	C66	0.94	0.96	0.96
C25	0.93	0.94	0.93	C67	0.96	0.96	0.97
C26	0.88	0.91	0.92	C68	0.98	0.98	0.97
C27	0.92	0.92	0.92	C69	0.97	0.97	0.96
C28	0.90	0.91	0.88	C70	0.98	0.98	0.97
C29	0.97	0.98	0.98	C71	0.81	0.85	0.80
C30	0.93	0.96	0.94	C72	0.72	0.74	0.73
C31	0.90	0.90	0.92	C73	0.97	0.96	0.97
C32	0.93	0.94	0.93	C74	0.94	0.97	0.96
C33	0.96	0.96	0.95	C75	0.92	0.90	0.89
C34	0.93	0.95	0.96	C76	0.98	0.98	0.96
C35	0.87	0.91	0.88	C77	0.94	0.95	0.92
C36	0.90	0.91	0.91	C78	0.95	0.96	0.95
C37	0.86	0.86	0.79	C79	0.94	0.95	0.95
C38	0.97	0.98	0.97	C80	0.87	0.88	0.85
C39	0.97	0.98	0.97	C81	0.96	0.96	0.97
C40	0.92	0.94	0.93	C82	0.94	0.96	0.96
C41	0.94	0.95	0.95	C83	0.89	0.91	0.88
C42	0.91	0.92	0.88	C84	0.94	0.96	0.94

Cls: Class, AN: AlexNet, GN: GoogLeNet, SN: SqueezeNet

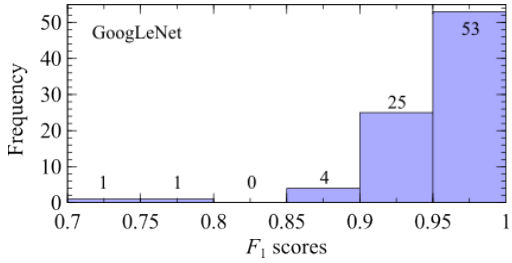
scores exceed 0.95 (i.e., scores greater than or equal to 0.95 and less than 1). Additionally, AN has 28 classes and SN has 23 classes with scores ranging between 0.9 and 0.95.

For GoogLeNet (GN), the frequency of scores above 0.95 is obtained for 53 classes, which is considerably higher than the frequencies for the other two networks in the same range. Overall, GN achieves an F_1 score of over 0.9 for 78 out of 84 classes. In comparison, AN has the same score range for 73 classes, while SN achieves this score range for only 68 classes. Hence, the histograms indicate that GN outperforms the other two networks in classifying the test data (since more classes haver higher scores). Furthermore, a similar pattern can be observed between the frequency score distributions of AN and SN.

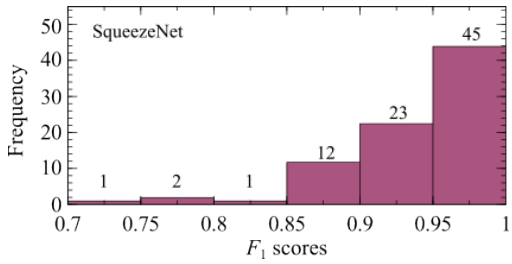
The histograms display an overview of the overall classification performance of the networks. However, for better comparison of the classification performance, it is worthwhile to compare the 'per class' classification performances as well. Fig.6 depicts the F_1 scores for each class of the three networks (in three separate plots). It can be observed that the F_1 scores for all the three networks follow the same pattern, i.e., if one network



(a) Distribution of F_1 scores (for AlexNet)



(b) Distribution of F_1 (for GoogLeNet)



(c) Distribution of F_1 (for SqueezeNet)

Fig. 5. Histogram of the F_1 scores for the three networks.

fares poorly (i.e., has a low F_1 score), or does well (i.e., has a high F_1 score) for a particular class, then the other two networks follow suit. This pattern is not without exception though, and for some of the classes, the performance (i.e. the score) differences are quite large (e.g., the classes C37, C43, C65, and C71) (Table IV).

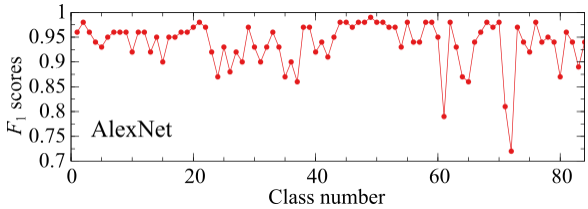
The difference in F_1 scores (for each class c) between any two networks (e.g., AN and SN) can be calculated as,

$$d_c^{(AS)} = F_{1_c}^{(AN)} - F_{1_c}^{(SN)} \tag{6}$$

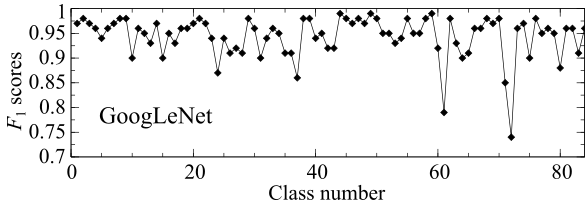
where, $F_{1_c}^{(AN)}$ and $F_{1_c}^{(SN)}$ are the F_1 scores of class c for AN and SN respectively, and $d_c^{(AS)}$ is the calculated difference in scores between these two networks. The score differences can be used to compare the performances between the networks (for each class).

Fig.7 displays the score differences calculated between each pair of networks. The score differences between GN (GoogLeNet) & SN (SqueezeNet), GN (GoogLeNet) & AN (AlexNet), and AN (AlexNet) & SN (SqueezeNet) are shown in Fig.7a, Fig.7b, and Fig.7c respectively for each class. In the figures, *a positive score difference* indicates a better F_1 score for GN compared to SN (in Fig.7a), GN compared to AN (in Fig.7b), and SN compared to AN (in Fig.7c).

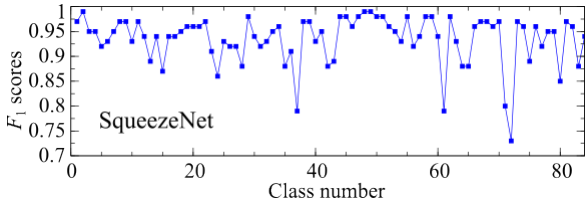
Since the F_1 scores were computed up to two decimal places (as shown in Table IV), the minimum possible difference between scores is 0.01. In both Fig.7a and Fig.7c,



(a) F_1 scores of AlexNet per class

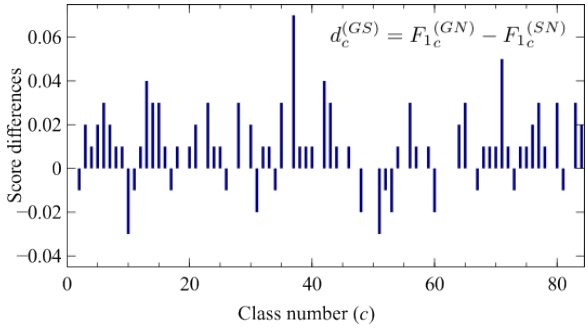


(b) F_1 scores of GoogLeNet per class

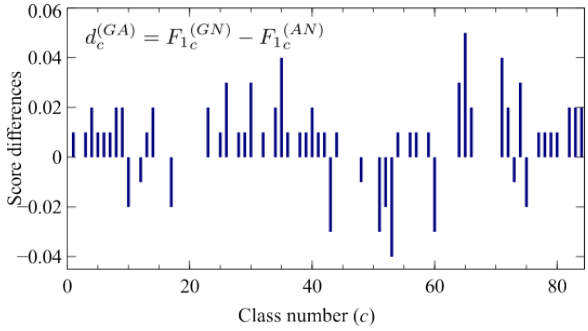


(c) F_1 scores of SqueezeNet per class

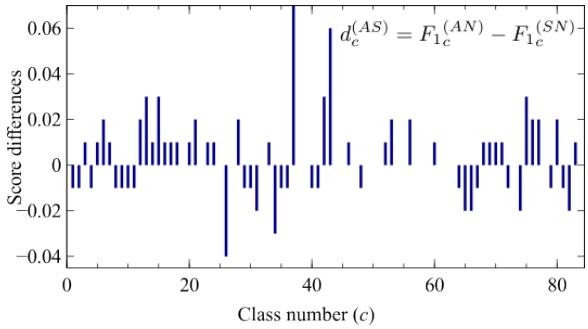
Fig. 6. F_1 scores of the three networks (plotted for each class) seem to indicate that all the network performances follow a similar pattern.



(a) Differences between F_1 scores of GN and SN for each class



(b) Differences between F_1 scores of GN and AN for each class



(c) Differences between F_1 scores of AN and SN for each class

Fig. 7. F_1 score differences between AN (AlexNet) & SN (SqueezeNet), GN (GoogLeNet) & SN (SqueezeNet), and AN (AlexNet) & SN (SqueezeNet) for each class. Overall, scores for GN are comparatively better than the other two.

the maximum score difference of 0.07 was observed for the class C37. For this particular class, SqueezeNet (SN) received a relatively lower score of 0.79, while both AlexNet (AN) and GoogLeNet (GN) achieved identical scores of 0.86. The second-highest score difference, 0.06, was observed for the class C43 in Fig.7b. For this class, SN obtained the lowest score of 0.89, AN had the highest score of 0.95, and GN attained a score of 0.92, which is lower than AN's score for that particular class.

The plots provide a graphical view on the comparative performance (for each class) among the networks. It can be seen that GN has overall better scores than both AN and SN (i.e., as indicated by the fewer numbers of negative values in the plots in Fig.7a and 7b). Out of the total 84 classes, GN scores better than SN in 50 classes, while both have the same scores for 19 classes. In the remaining 15 classes, SN scores better than GN. From the cases where SqueezeNet (SN) surpasses GoogLeNet (GN), it was observed that the maximum score difference amounts to 0.03 for classes C10 and C51, and there is a difference of 0.02 for classes C31, C48, C53, and C60.

Again, from Fig.7b, out of the 84 classes, GN scores better than AN in 43 classes, and GN has the same scores as AN in 30 classes. AN has a better score than GN in the rest of the 11 classes. Also, it can be observed that AN outperforms GN more strongly than SN does, i.e., AN scores 0.04 more than GN for the class C53, 0.03 more in the classes C43, C51, and C60, and 0.02 more in classes C10, C17, C52, and C75.

Between AN and SN Fig.7c, the latter scores 0.04 more than AN for class C26. Moreover, SN scores 0.03 more than AN for the class C34, while the difference is 0.02 for the classes C31, C65, C66, C74, and C82. AN and SN has the same score for 24 classes, while AN scores better than SN in 34 classes. For the remaining 26 classes, SN performs better than AN. Although none of these two networks clearly outperforms the other one, but AN seems to have a slight edge over SN considering the performance.

Poor Performance of all Networks for a Few Classes:

For a few of the classes (e.g., C61, C71, and C72), *all* the three networks exhibit poor classification performances. The F_1 scores for the networks were significanly low (i.e., 0.8 or less) for these three classes. In fact, the lowest scores for all three networks were for class C72 (0.72 for AN, 0.73 for SN, and 0.74 for GN). On the other hand, class C61 has a score of 0.79 (the same for all the three networks). This score happens to be the second-lowest score for all the three networks (by the

TABLE V
CLASSIFICATION RESULT FOR THE CLASSES C61 AND C72

		C61	C72
AlexNet	True Positive	305	250
	False Negative	55	110
	False Positive	112	84
SqueezeNet	True Positive	315	224
	False Negative	45	136
	False Positive	130	30
GoogLeNet	True Positive	314	252
	False Negative	46	108
	False Positive	114	69

way, SqueezeNet has the same score for another class, C37).

The characters represented by the classes C61 (ক্ষ) and C72 (ক্ষ) have extremely similar looking visual patterns. The only difference between these two is that, one has a loop-like extension at the rightmost end, while the other has a vertical line. When these characters are not used within the context of a meaningful word, differentiating between these two symbols from handwritings can be quite challenging even for native users of Bangla.

For the classified test samples of these two classes, Table V shows the number of true positives, false negatives, and false positives. We see that the class C61 indeed has a high number of false positives for all the networks, while the class C72 has a high number of false negatives. This would explain the poor F_1 scores (equations 4,5) obtained for both these classes.

In fact, most of the compound characters have plenty of visual similarities that could confuse the networks. For these classes (i.e., C61 to C84), GoogLeNet performs comparatively better than the other two (Fig.7).

Quantifying the Overall Performance of a Network:

So far, the discussion was on class specific network performances. However, in cases where a classification task involves numerous classes, and the performance scores exhibit variance, a single metric is often employed to effectively quantify the overall performance. One commonly used metric in such scenarios is the micro-averaged F_1 score [27]. For a particular network, the micro averaged F_1 score is obtained by,

$$F_{1micro-avg} = \frac{2(P_{total} \times R_{total})}{(P_{total} + R_{total})} \tag{7}$$

where,

$$P_{total} = \frac{\sum_{c=1}^{84} TP_c}{\sum_{c=1}^{84} TP_c + \sum_{c=1}^{84} FP_c} \tag{8}$$

and

$$R_{total} = \frac{\sum_{c=1}^{84} TP_c}{\sum_{c=1}^{84} TP_c + \sum_{c=1}^{84} FN_c} \tag{9}$$

Here, TP_c is the number of true positive samples, FP_c is the number of false positive samples, and FN_c is the number of false negative samples of class c respectively. The summation is performed over all classes.

TABLE VI
MICRO AVERAGED F_1 SCORES OF THE NETWORKS ON THE TEST DATA OF ALL CHARACTER CLASSES

Summary of network performance		
AlexNet	SqueezeNet	GoogLeNet
0.9396	0.9361	0.9453

TABLE VII
A FEW OF THE 568 CHARACTER IMAGES MISCLASSIFIED AS THE SAME CLASS BY ALL THREE NETWORKS

Image	Label	Recognized by all networks as class	Comments
	C01 (অ)	C39 (ল)	Ambiguous writing
	C02 (আ)	C01 (অ)	Wrong label
	C03 (ই)	C04 (ঈ)	Ambiguous writing
	C12 (ক)	C20 (ঝ)	Ambiguous writing
	C22 (ট)	C25 (ঢ)	Wrong label
	C53 (২)	C43 (হ)	Wrong label
	C71 (স্প)	C80 (শ্প)	Wrong label

Table VI shows the micro averaged F_1 scores for the three networks. From the table, we see that GoogLeNet slightly outperforms the other two networks on the overall classification task. Between the other two, SqueezeNet performs a little worse than AlexNet.

Finally, we found that there were a total of 568 test samples which were misclassified by all the three networks. Coincidentally, all three networks predicted the *same* incorrect class for *each* of these samples. A few examples of such samples are shown in Table VII with the original character images, their labeled class numbers, and the predicted classes. By sifting through all these characters, it was found that these samples were either incorrectly labeled in the original database or were unclear due to ambiguous writing styles. These comments are placed alongside the images as well in the Table VII. The samples with unclear or ambiguous writing styles will be mis-interpreted even by natives and can easily be misclassified into the same classes as did the DCNNs. If both types of samples are discarded from the test data set, the networks' micro-averaged F_1 scores show a slight improvement as indicated in Table VIII.

TABLE VIII
ADJUSTED MICRO AVERAGED F_1 SCORES OF THE THREE NETWORKS ON THE TEST DATA OF ALL CHARACTERS CLASSES AFTER DISCARDING THE 568 TEST SAMPLES

Total 29,672 images			
Micro averaged F_1 score	AlexNet	SqueezeNet	GoogLeNet
	0.9576	0.9541	0.9634

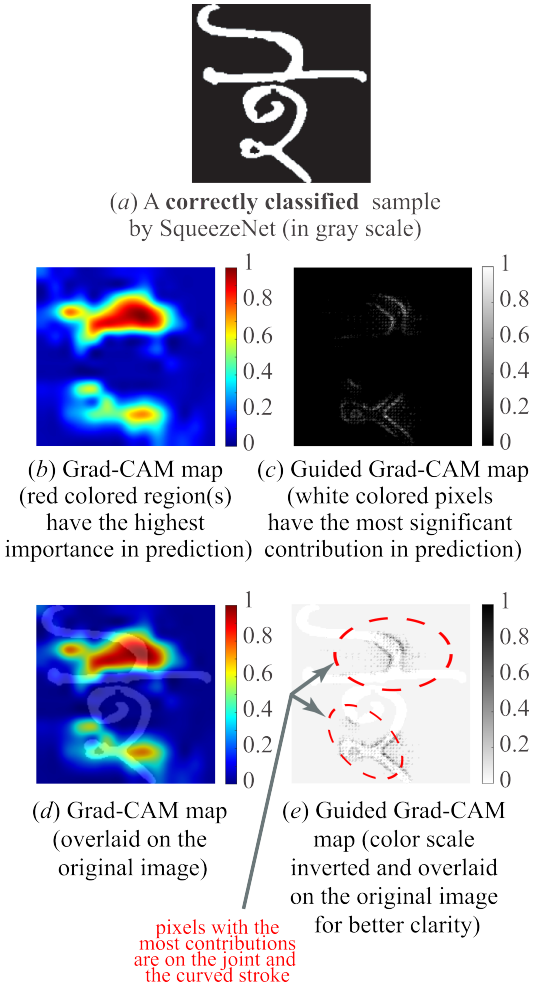


Fig. 8. Example of a correctly classified sample of class C03 by SqueezeNet. Here, (a) is the original image and (b, c) are the Grad-CAM and Guided Grad-CAM maps respectively. From these maps, the *correlation* between the highlighted regions (and pixels) in the maps and the strokes of the sample character are *not clear*. The maps are therefore re-drawn in (d, e) with the original image overlaid. The color scale of (c) was also inverted while plotting (e) for clarity of the map. As displayed now by the red colored portions in (d), the most important regions are the top curved stroke around the joint with the horizontal line and the curved stroke at the bottom. As displayed in (e), the pixels around the curved stroke near the joint as well as the curved stroke at the bottom contribute the most for a correct prediction of the sample.

B. Saliency Maps of Test Samples to Understand Classifications Failures

DCNNs perform classification by learning distinctive features about the different classes during training. Understanding the specific features that the networks focus on during classification can provide insights into why some samples might be misclassified by one network while correctly classified by the other two. Saliency maps serve as valuable tools that allow us to peer inside the networks and explore their behavior.To investigate the networks' behaviour, the saliency maps of the samples were plotted using two techniques, i.e., Grad-CAM, and Guided Grad-CAM (as described in section III-D).

SqueezeNet: Fig.8(a) shows the (original) image of a correctly classified sample (of class C03) by SqueezeNet. Fig.8(b) and (c) shows the heatmaps generated by the Grad-CAM and the Guided Grad-CAM techniques re-

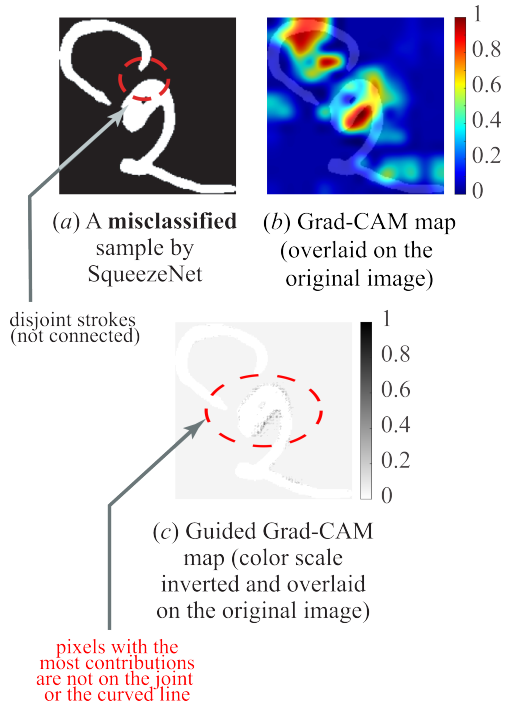


Fig. 9. Example of an *incorrectly* classified sample of class C03 by SqueezeNet. Here, (a) is the original image, and (b, c) are the Grad-CAM and the Guided Grad-CAM maps overlaid on the original image ((c) is also color scale inverted). For this sample, the same regions as in Fig.8 and the pixels in that region are *not* of interest for prediction (as displayed in (b), (c)), and therefore the sample is misclassified.

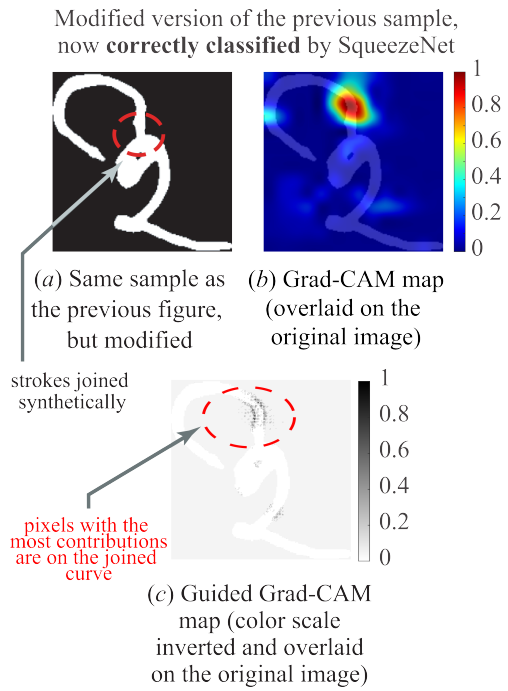


Fig. 10. The sample of Fig.9 after modification is **correctly** classified by SqueezeNet. Here, (a) is image of the modified sample, and (b), (c) show the saliency maps. The Grad-CAM and Guided Grad-CAM maps indicate that the network now focuses on *the curved stroke*, thus resulting in a correct classification.

spectively for this sample and Fig.8(d) and (e) present the same heatmaps, but this time they are overlaid on the original sample to provide a clearer understanding of the contributions of the regions/ pixels in the context of the sample. In the case of Guided Gard-CAM maps, since the pixels from only a small area are in focus, therefore, inverting the color scale provide enhanced clarity. Therefore, Fig.8(e) was plotted with an inverted color scale for improved visualization. Note that the maps are scaled from 0 to 1 for ease of comparison.

The Grad-CAM map (i.e., Fig.8(d)) reveals the *local regions* of the image that received the most emphasis from the network during the classification (i.e., recognition) process. The region(s) with values closer to 1 indicate having a higher level of emphasis during the classification (colored bright red). On the other hand, the Guided Grad-CAM plot highlights the *pixels* in the image that contribute the most to the activation map for this particular class (i.e., C03). These pixels are responsible for the final classification result (i.e., recognition). In the maps, these pixels are darker in color. Looking at the Grad-CAM map (Fig.8(d)), it can be observed that the network focuses on two regions for this class - one is where the upper curved stroke joins with the horizontal stroke, and the other is the bottom curved stroke. The Guided Grad-CAM map (Fig.8(c)) highlights the corresponding pixels for this class, thereby further confirming that the pixels in these regions significantly contribute to the activation for this class.

Another sample of the same class (i.e., class C03) is shown in Fig.9(a), which was **misclassified** only by SqueezeNet. In this sample of class C03, unlike the previous sample, the upper curved stroke is *not* connected to any horizontal lines. The Grad-CAM map as well as the Guided Grad-CAM map of this sample are shown in Fig.9(b) and (c) respectively with the original sample as the background. The Grad-CAM map indicate that now, there are multiple regions which are emphasized by the network, and from the Guided Grad-CAM map (Fig.9(c)), it is clear that the pixels of the upper curved stroke of the character did not contribute to the activation (i.e., the pixels that have most contribution are located in a different region). Therefore, we hypothesized that, the activation did not have any contribution from the pixels of the upper curved stroke due to the *disjoint* of the upper stroke from the main body of the character.

To validate this hypothesis, we generated a synthetic sample (using an image editing tool) by joining the upper curved stroke with the main body of the misclassified sample. This modified sample is shown in Fig.10(a). To investigate the impact of the synthetic joint on the classification outcome, the modified sample was tested with the SqueezeNet network. Remarkably, the network correctly classified the modified sample. Furthermore, saliency maps in Fig.10(b) and (c) indicate that the network now focuses on the region of the upper curved stroke instead, and the pixels in this area contribute the most to the classification (just like the sample that was classified correctly). This verifies our conjecture that the joint of the curve is indeed considered as an important feature by SqueezeNet.

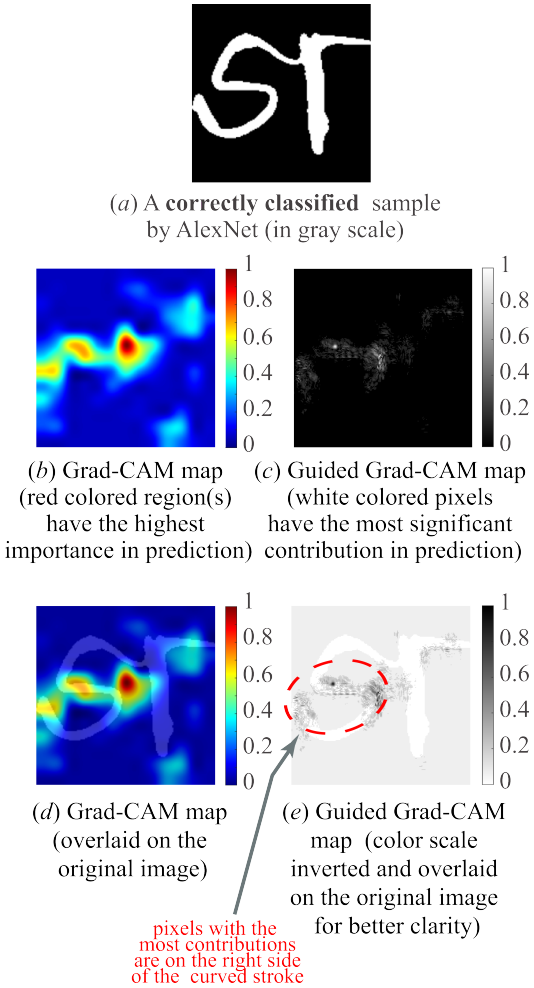


Fig. 11. Example of a correctly classified sample of class C14 by AlexNet. Here, (a) is the original image, and (b, c) are the Grad-CAM and Guided Grad-CAM maps respectively. From these maps, the *correlation* between the highlighted regions (and pixels) in the maps and the strokes of the sample character are *not clear*. The maps are therefore re-drawn in (d, e) with the original image overlaid (color scale in (e) is inverted from the cale in (c)). The most important region is the right side of the curved stroke (as displayed in (d)) (i.e., the red colored portion). The pixels in this region contribute the most for a correct prediction of the sample (as displayed in (e)).

AlexNet: For AlexNet, two samples of class C14 were taken to perform a similar type of experiment. For the correctly classified sample (shown in Fig.11(a)), the Grad-CAM and the Guided Grad-CAM maps indicate that for this class, the emphasis is on right side of the curved stroke (as displayed in Fig.11 (d) and (e)).

However, for a misclassified sample of class C14 (shown in Fig.12(a)), the emphasis shifts to the bottom portion of the curved stroke (Fig.12(b), and (c)). It was noticed that unlike the correctly classified sample of this class, this sample had a comparatively smaller gap (i.e., opening) between the two left-hand side strokes. Again, we hypothesized this to be the cause of the misclassification.

Fig.13(a) shows the same misclassified sample, but with a synthetically enlarged gap (by erasing a small part of the curved stroke using an image editing tool). The corresponding saliency maps (as shown in Fig.13(b) and (c)) indicate that the pixels to the right side of the curved line are now contributing to the classification. The

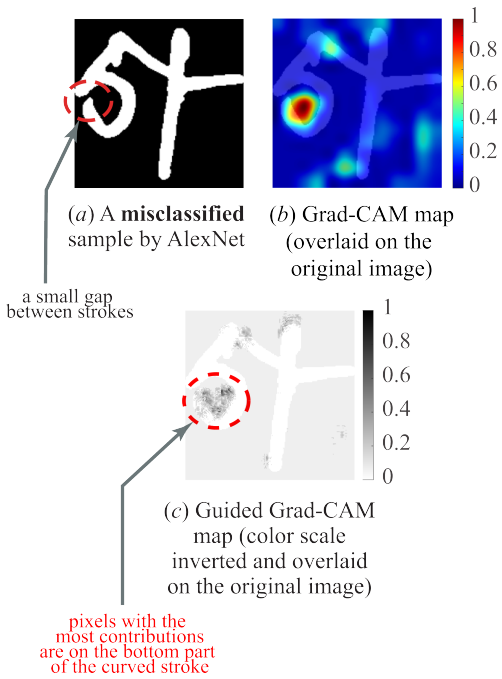


Fig. 12. Example of an *incorrectly* classified sample of class C14 by AlexNet. Here, (a) is the original image, and (b, c) are the Grad-CAM and the Guided Grad-CAM maps overlaid on the original image ((c) is also color scale inverted). For this sample, the region of interest for prediction is the bottom portion of the curved stroke (as displayed in (b), (c)). The smaller gap between the strokes changes the decision-making region and therefore the sample is misclassified.

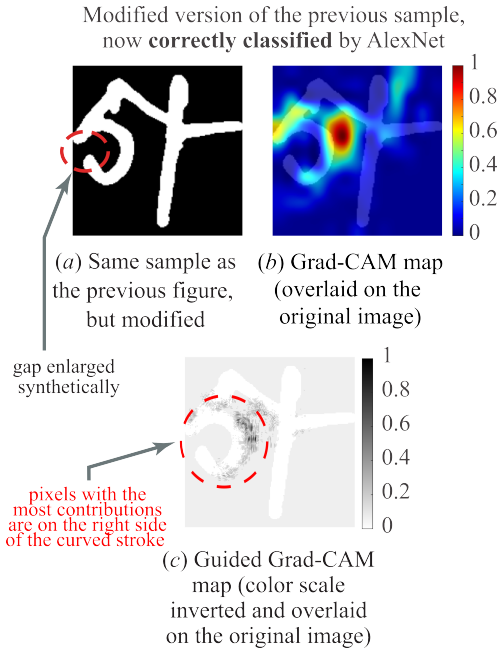


Fig. 13. The sample of Fig.12 after modification is **correctly** classified by AlexNet. Here, (a) is image of the modified sample, and (b), (c) show the saliency maps. The Grad-CAM and Guided Grad-CAM maps indicate that the network now focuses on **right side** of the curved stroke.

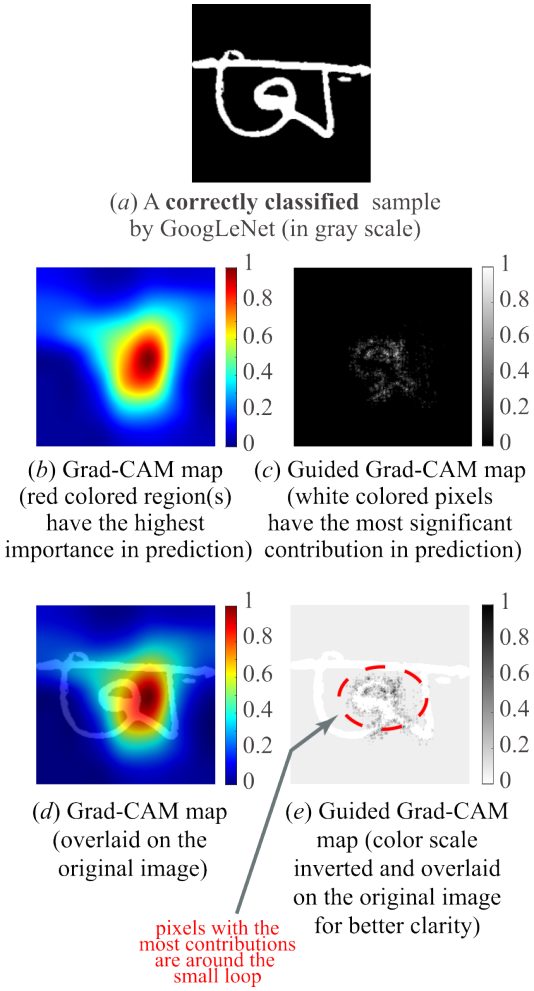


Fig. 14. Example of a correctly classified sample of class C01 by GoogLeNet. Here, (a) is the original image, and (b, c) are the Grad-CAM and Guided Grad-CAM maps respectively. From these maps, the *correlation* between the highlighted regions (and pixels) in the maps and the strokes of the sample character are *not clear*. The maps are therefore re-drawn in (d, e) with the original image overlaid (with color scale inverted in (e)). The most important region is around the small loop (as displayed in (d)) (i.e., the red colored portion). The pixels around the small loop contribute the most for a correct prediction of the sample (as displayed in (e)).

fact that this modified sample has now been properly identified proves again that our hypothesis is correct.

GoogLeNet: Lastly, for a correctly identified sample of C01 (Fig.14(a)) by GoogleNet, the Grad-CAM and Guided Grad-CAM maps (Fig.14(d) and (e)) clearly indicate that the main focus is on the small loop below the horizontal line.

Examining another sample from the same class C01 (Fig.15(a)) which was misclassified solely by GoogLeNet, it was observed that it does not contain the loop like feature at the beginning of the curve unlike Fig.15(a). Therefore, the saliency maps of this misclassified sample focuses on the region and pixels along the neck of the curved stroke (Fig.15(b) and (c)). From this observation, we hypothesized that the absence of the loop in this sample is contributing in misleading the network's decision.

After the inclusion of a small loop in the misclassified sample using an image editing tool (as shown in Fig.16(a)), the modified sample was found to be correctly classified by GoogleNet. The saliency maps indicate that

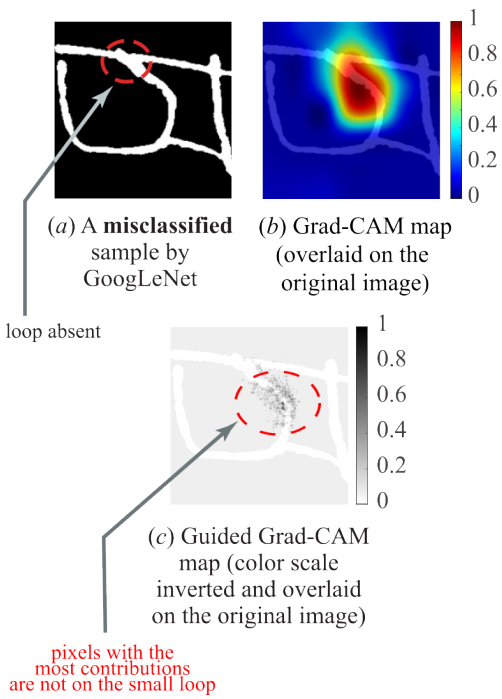


Fig. 15. Example of an *incorrectly* classified sample of class C01 by GoogLeNet. Here, (a) is the original image, and (b, c) are the Grad-CAM and the Guided Grad-CAM maps overlaid on the original image ((c) is also color scale inverted). For this sample, the region of interest is different than in Fig.14 for prediction (as displayed in (b), (c)), and therefore the sample is misclassified. The absence of a distinct loop changes the decision-making region.

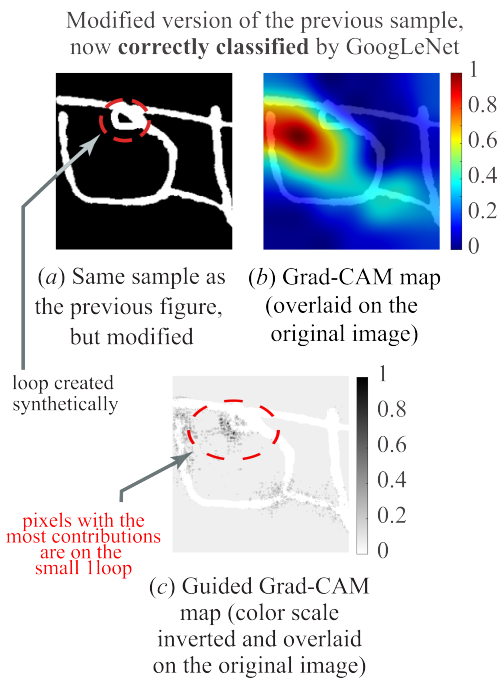


Fig. 16. The maps after modifying the sample of 14. It is now **correctly** classified by GoogLeNet. (a) is the sample after modification, (b, c) are the Grad-CAM and Guided Grad-CAM map. The maps indicate that the network now focuses on the region around the small loop.

pixels around the loop are indeed contributing to the classification (Fig.16(b) and (c)).

From the study on the correctly classified and misclassified samples using saliency maps, weaknesses in networks' feature learning ability can be identified. Training the networks with more samples of these outlier characteristics will pave the way of overcoming these weaknesses.

In deep learning research, augmented data is commonly used to train networks where adequate training data is not available. This augmentation process often includes geometric transformations, color space transformations, image blurring, and other techniques. For Bangla handwriting recognition, instead of using these generalized augmentation methods, a more focused approach can be adopted by augmenting the data based on observations from recognition failures that display outlier characteristics, as identified through saliency maps here. This targeted augmentation strategy has the potential to enhance the network's learning capabilities and improve its robustness by specifically addressing areas where the network struggles with accurate recognition.

V. Conclusion

In this paper, the classification performances of three deep convolutional neural networks (i..e, AlexNet, SqueezeNet, and GoogLeNet) were investigated on a standard database of Bangla handwritten characters. Rather than training these networks from scratch, in this paper, already pre-trained versions of the networks (on ImageNet database) were used. Through transfer learning, further training was carried out (using a standard Bangla database) to tune these networks to classify Bangla handwritten character samples. The same set of training, test, and validation data samples were used to compare the networks' classification performances with each other. Both class-wise and overall performances were discussed. The micro averaged F_1 scores of the classified samples show that GoogLeNet (F_1 : 0.9634) has a better classification performance than both AlexNet (F_1 : 0.9576) and SqueezeNet (F_1 : 0.9541).

This paper includes further analysis with the use of saliency maps (i.e., Grad-CAM and Guided Grad-CAM) to identify the class discriminative features of the trained networks. This helped to figure out why some samples were misclassified by a particular network only. As networks are often trained with augmented data, this information about features can be used to prepare a set of (targeted) augmented training samples to improve a network's learning ability for Bangla handwriting recognition.

Although this paper showed the analysis using three networks, this research could easily be extended to include other networks with smaller number of parameters. Higher classification performances could be achieved by creating a network specific augmented database of samples for all the character classes.

References

[1] H. F. Schantz, *History of OCR, Optical Character Recognition.* Recognition Technologies Users Association, 1982.

[2] S. Roy, N. Das, M. Kundu, and M. Nasipuri, "Handwritten Isolated Bangla Compound Character Recognition: A New Benchmark using a Novel Deep Learning Approach," *Pattern Recognition Letters*, vol. 90, pp. 15–21, 2017.

[3] T. S. Gunawan, A. Noor, M. Kartiwi, "Development of English Handwritten Recognition using Deep Neural Network," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 10, no. 2, pp. 562–568, 2018.

[4] D. Cireşan and U. Meier, "Multi-column Deep Neural Networks for Offline Handwritten Chinese Character Classification," in *2015 International Joint Conference on Neural Networks (IJCNN),*12–17 July, 2015, Killarney, Ireland, pp. 1–6.

[5] A. A. Abbasi, L. Hussain, I. A. Awan, I. Abbasi, A. Majid, M. S. A. Nadeem, and Q.-A. Chaudhary, "Detecting Prostate Cancer using Deep Learning Convolution neural network with transfer learning approach," *Cognitive Neurodynamics*, vol. 14, no. 4, pp. 523–533, 2020.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet Classification with Deep Convolutional Neural Networks," *Advances in Neural Information Processing Systems*, vol. 25, no. 6, pp. 84–90, 2012.

[7] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level Accuracy with 50x Fewer Parameters and< 0.5 MB Model Size," *arXiv preprint arXiv:1602.07360*, 2016.

[8] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going Deeper with Convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7–12 June, 2015, Boston, MA, USA, pp. 1–9.

[9] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual Explanations from Deep Networks via Gradient-based Localization," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 22–29 October, 2017, Venice, Italy, pp. 618–626.

[10] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for Simplicity: The All Convolutional Net," *arXiv preprint arXiv:1412.6806*, 2014.

[11] C.-L. Liu and C. Y. Suen, "A New Benchmark on the Recognition of Handwritten Bangla and Farsi Numeral Characters," *Pattern Recognition*, vol. 42, no. 12, pp. 3287–3295, 2009.

[12] U. Bhattacharya and B. B. Chaudhuri, "Handwritten Numeral Databases of Indian Scripts and Multistage Recognition of Mixed Numerals," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 3, pp. 444–457, 2008.

[13] S. Bag, G. Harit, and P. Bhowmick, "Recognition of Bangla Compound Characters using Structural Decomposition," *Pattern Recognition*, vol. 47, no. 3, pp. 1187–1201, 2014.

[14] M. Z. Alom, P. Sidike, M. Hasan, T. M. Taha, and V. K. Asari, "Handwritten Bangla Character Recognition using the State-of-the-Art Deep Convolutional Neural Networks," *Computational Intelligence and Neuroscience*, vol. 2018, 2018.

[15] N. Das, K. Acharya, R. Sarkar, S. Basu, M. Kundu, and M. Nasipuri, "A Benchmark Image Database of Isolated Bangla Handwritten Compound Characters," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 17, no. 4, pp. 413–431, 2014.

[16] D. S. Maitra, U. Bhattacharya, and S. K. Parui, "CNN based Common Approach to Handwritten Character Recognition of Multiple Scripts," in *2015 13th International Conference on Document Analysis and Recognition (ICDAR),*23–26 August, 2015, Tunis, Tunisia, pp. 1021–1025.

[17] R. R. Chowdhury, M. S. Hossain, R. ul Islam, K. Andersson, and S. Hossain, "Bangla Handwritten Character Recognition using Convolutional Neural Network with Data Augmentation," in *2019 Joint 8th International Conference on Informatics, Electronics & Vision (ICIEV) and 2019 3rd International Conference on Imaging, Vision & Pattern Recognition (icIVPR),* 30 May–2 June, 2019, Spokane, WA, USA, pp. 318–323.

[18] A. Hazra, P. Choudhary, S. Inunganbi, and M. Adhikari, "Bangla-Meitei Mayek Scripts Handwritten Character Recognition using Convolutional Neural Network," *Applied Intelligence*, vol. 51, pp. 2291–2311, 2021.

[19] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision*, vol. 115, pp. 211–252, 2015.

[20] M. Biswas, R. Islam, G. K. Shom, M. Shopon, N. Mohammed, S. Momen, and A. Abedin, "Banglalekha-Isolated: A Multi-purpose Comprehensive Dataset of Handwritten Bangla Isolated Characters," *Data in Brief*, vol. 12, pp. 103–107, 2017.

[21] A. S. A. Rabby, S. Haque, S. Islam, S. Abujar, and S. A. Hossain, "Bornonet: Bangla Handwritten Characters Recognition using Convolutional Neural Network," *Procedia Computer Science*, vol. 143, pp. 528–535, 2018.

[22] R. Keys, "Cubic Convolution Interpolation for Digital Image Processing," *IEEE Transactions on Acoustics, Speech, and Signal processing*, vol. 29, no. 6, pp. 1153–1160, 1981.

[23] N. Qian, "On the Momentum Term in Gradient Descent Learning Algorithms," *Neural Networks*, vol. 12, no. 1, pp. 145–151, 1999.

[24] J. R. Schott, *Matrix Analysis for Statistics.* John Wiley & Sons, 2016.

[25] L. Derczynski, "Complementarity, F-score, and NLP evaluation," in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, 23–28 May, 2016, Portorož, Slovenia, pp. 261–266.

[26] Dorcas. Esan, Pius. A Owolawi, and Chunling Tu, "Anomalous Detection in Noisy Image Frames using Cooperative Median Filtering and KNN," *IAENG International Journal of Computer Science*, vol. 49, no. 1, pp. 1–10, 2022.

[27] Zheng Chai, Yanying Li, Aili Wang, Chen Li, Baoshuang Zhang, and Huanhuan Gong, "An Efficient Pseudo Nearest Neighbor Classifier," *IAENG International Journal of Computer Science*, vol. 48, no. 4, pp. 1075–1086, 2021.