FabricNet: A Fiber Recognition Architecture Using Ensemble ConvNets

ABU QUWSAR OHI[1], **M. F. MRIDHA**[1], (Member, IEEE), **MD. ABDUL HAMID**[2], **MUHAMMAD MOSTAFA MONOWAR**[2], (Member, IEEE), **AND FARIS A. KATEB**[2]

[1]Department of Computer Science and Engineering, Bangladesh University of Business and Technology, Dhaka 1216, Bangladesh
[2]Department of Information Technology, Faculty of Computing and Information Technology, King AbdulAziz University, Jeddah 21589, Saudi Arabia

Corresponding author: M. F. Mridha (firoz@bubt.edu.bd)

ABSTRACT Fabric is a planar material composed of textile fibers. Textile fibers are generated from many natural sources; including plants, animals, minerals, and even, it can be synthetic. A particular fabric may contain different types of fibers that pass through a complex production process. Fiber identification is usually carried out through chemical tests and microscopic tests. However, these testing processes are complicated as well as time-consuming. We propose FabricNet, a pioneering approach for the image-based textile fiber recognition system, which may have a revolutionary impact from individual to the industrial fiber recognition process. The FabricNet can recognize a large scale of fibers by only utilizing a surface image of fabric. The recognition system is constructed using a distinct category of class-based ensemble convolutional neural network (CNN) architecture. The experiment is conducted on recognizing 50 different types of textile fibers. This experiment includes a significantly large number of unique textile fibers than previous research endeavors to the best of our knowledge. We experiment with popular CNN architectures that include Inception, ResNet, VGG, MobileNet, DenseNet, and Xception. Finally, the experimental results demonstrate that FabricNet outperforms the state-of-the-art popular CNN architectures by reaching an accuracy of 84% and F1-score of 90%.

INDEX TERMS Textile fiber recognition, image processing, convolutional neural network, pattern recognition, ensemble architecture.

I. INTRODUCTION

Textile fibers are the components that are used to construct fabrics. Commonly, the types of fibers are split into two categories: natural fibers and synthetic fibers. Natural fibers are extracted from environmental sources, whereas synthetic fibers are manufactured through machinery and chemical compounds. Such instances of natural fibers are silk, wool, cotton, etc. whereas, nylon, polyester, rayons, etc. are the example of synthetic fibers. Raw fibers are used to assemble yarns. A single yarn is assembled using one or more types of raw fibers. The yarns are further utilized to construct fabrics and particular garments.

Fiber recognition is the process of identifying raw fibers from fabrics, and it is widely used in different industrial applications. It is a well-applied method in fabric reverse engineering [1]. Garment identification is also possible using fiber recognition systems since each garment type mostly

requires fixed sets of fiber elements [2]. The fiber recognition system can also be implemented as fabric fault detection and garment inquiry systems [3].

Identifying raw textile fibers is considered difficult due to the complicated weave structure and aging of fibers [5]. Moreover, present fabrics pass through complex printing procedures that may alter the yarn structure. Classical biological methods, such as soaking, cleaning, heating, etc. are considered less effective in raw textile fiber identification. However, microscopic observations are proven to be more accurate in identifying raw textile fibers. The textile fiber recognition from manufactured fabrics is complicated since a single yarn (used to construct the fabric) can contain multiple textile fibers. In some cases, fabrics are mostly preprocessed. Moreover, microscopic observations may lead to false recognition, as numerous fibers can be used to generate a single textile yarn.

Generally, most systems identify textile fibers through microscopic cross-section images [6] and spectroscopic features [7], [8]. Fibers can be distinguished by microscopic
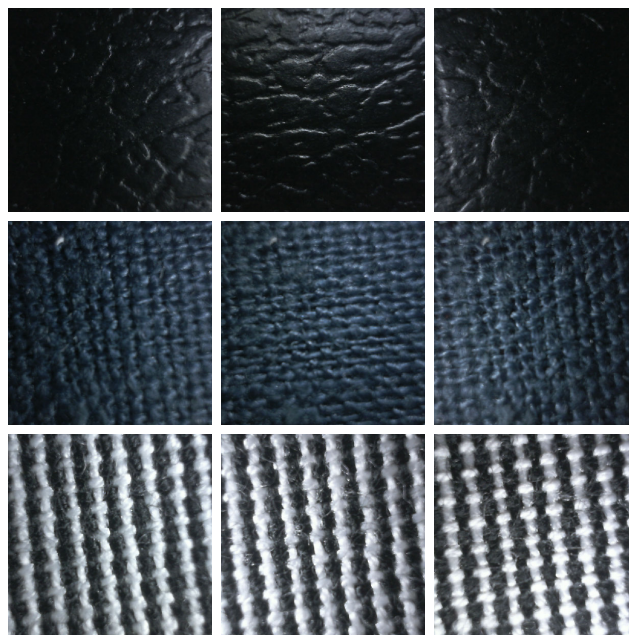
The associate editor coordinating the review of this manuscript and approving it for publication was Victor Sanchez.
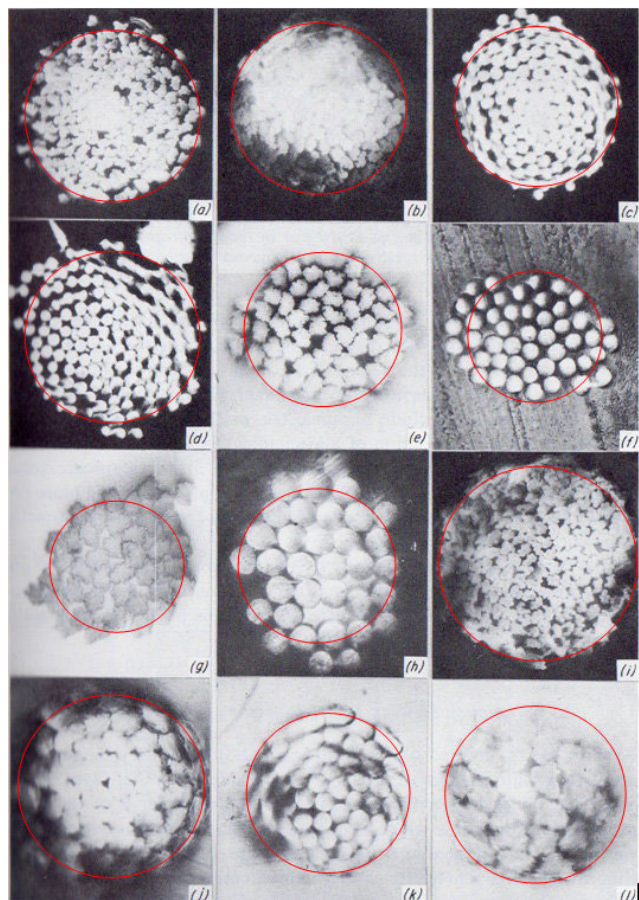
FIGURE 1. Different yarn cross sectional shapes of particular fibers: Tenasco (a,b,i), Nylon (e,g,j,l), Viscose (f,k) and Terylene. The image is adopted from the work of Hearle *et al.* [4].

FIGURE 2. The dataset contains fabric images in different light and orientations [2]. The first row illustrates fabric made of artificial leather. The second row illustrates the fabric made of silk. The third row contains images of fabric which is made of polyester and viscose (rayon).

cross-section images due to their unique geometrical properties [9]. Figure 1 illustrates different cross-section shapes of different types of yarns. Extricating cross-sectional shots is nearly impossible for industrial usage of textile fiber identification systems, as it requires a careful pre-processing and microscopes. Using cross-sectional images for recognizing textile fibers from fabrics is a critical approach in real-time industrial aspects. Hence, the cross-sectional investigation requires laboratories and is time-consuming. On the contrary, spectroscopy-based methods can be used for industrial purposes, but it is limited to recognizing only a single textile fiber from a fabric. Further, the method is not suitable for individual usage.

The research endeavor's sole purpose is to overcome the information gathering complexities of the fiber recognition procedure. Smartphones and high definition cameras have made image capturing one of the most superficial attempts for information gathering procedures. Therefore, we introduce a novel architecture that can recognize the textile fibers by a fabric surface image. Because of the availability of cameras, our proposed textile fiber recognition process can be performed by individuals, and even by automated machines in a much more convenient and flexible way. Figure 2 represents

some image samples that are used to conduct the training of our FabricNet architecture. The overall architecture performs CNN based image processing using an ensemble architecture. Since our proposed model recognizes textile fibers through fabric surface image, it can be widely applied for diverse industrial and individual applications for fault checking and authentication. Our process can further be used for textile fraud prevention and fault detection. The overall contribution of the research endeavor includes the following:

- We exploit the surface image of fabric in recognizing textile fibers, as it is one of the easiest ways for image collection.
- We outline different categories of ensemble methods, and introduce a class-based ensemble architecture that receives downsampled image data through a head CNN architecture. In class-based ensemble architecture, every single ensemble memorizes only one class. Therefore, the accuracy of FabricNet architecture increases.
- We experiment with seventeen different implementations of famous image classification architectures, including Inception, ResNet, VGG, MobileNet, DenseNet, Xception, and CU-Net. Through the result analysis, we affirm that FabricNet architecture provides better accuracy.

The remainder of this paper is outlined as follows. Section II demonstrates the procedures that are modeled to identify textile fibers. Section III presents the motivation and the architectural fundamentals of the FabricNet. Section IV contains the experimental results that are performed to evaluate FabricNet architecture. Finally, Section V concludes the paper.

II. RELATED WORKS

The textile fiber identification process can be divided into two types of tests: technical test and non-technical test [10]. Before the advancement of microscopic tests, non-technical tests for textile fiber identification were mostly conducted. Buring test, soaking test, feeling, etc. are considered as non-technical tests. The drawback of the non-technical tests is its authenticity [5].

Technical tests include the usage of microscopes and chemicals for the detection of textile fibers. Chemical tests include stain tests and solvent tests. However, the limitation of chemical tests is their inapplicability to separate multiple fibers. Therefore, chemical tests can not identify multiple fibers from fabrics [11]. Microscopic examinations are also considered as technical tests that involve identifying textile fabric using microscopes. Experts previously conducted microscope tests, and it proved to be more accurate. Nevertheless, synthetic fibers are mostly geometrically similar in microscopic view; therefore, specialists can sometimes find it challenging to distinguish fibers, even applying a microscope [12]. Figure 1 illustrates an example of a microscopic test.

The usage of image recognition systems in fiber identification has been observed since late 1980 [13]. The utilization of image processing for fiber recognition is observed since late 1990 that used Support Vector Machine (SVM) [14]. The SVM required a feature extraction method on which the effectiveness of the scheme was mostly dependent. The process also needed a microscopic cross-section view of fibers, which made the technique inefficient for real-time fiber recognition. The later research works conducted in the field of fiber recognition systems followed the same path. The purpose of the proposed architectures was to eliminate human interference from the technical microscopic fiber recognition procedure. Statistical analysis has also been introduced to identify fibers from cross-section images [15]. Nevertheless, the research work only classified two types of fibers, which is not suitable for real-world usage.

Neural networks became famous after the backpropagation learning method was introduced [16]. Neural network architecture is also introduced to identify fibers [17]. However, researches exploiting neural network architectures could only distinguish two types of fibers. Furthermore, neural networks do not fit in the image recognition process. Hence, neural network-based architectures are not suitable for recognizing a broad set of textile fibers.

CNN architectures started to rule in image identification problems through the successful breakthrough of AlexNet [18]. Till now, CNN architectures are considered the state-of-the-art image classification mechanism due to its robustness and ability to identify objects from a large set of targets. Also, CNN architectures perform auto feature extraction, eliminating the dependency from image feature extraction procedure [14]. CNN architecture has been introduced to identify fibers using cross-section microscopic images [19]. The research was carried out by identifying

seven types of fibers and achieved an acceptable accuracy level. Nevertheless, the method still depends on cross-section microscopic images, and the number of unique fibers is inadequate.

Apart from the same process of the automated identification of fibers from microscopic images, spectroscopy-based fiber identification methods have also been introduced [7], [8]. Nevertheless, the experiments of the research works were conducted with limited varieties of textile fibers, and the spectroscopy-based method can only identify a single fiber at once. The spectroscopy-based fiber identification process is also time-consuming and often requires an expert to position the spectroscope and calculate appropriate reading time correctly.

The automated fiber identification has been less exploited, and most research endeavor is conducted to identify a single type of fiber at once. This type of classification is defined as multi-class classification. Furthermore, previously implemented architectures are time-consuming and require laboratory equipment to carry out the data collection (cross-section image or spectrograph extraction) and testing process. Therefore, mass testing and validating fibers of fabric often incur high cost.

Feng *et al.* [20] have proposed a similar concept of converting the fiber recognition system into a multi-class classification problem. The authors presented a CNN-based ensemble architecture that contains an almost equivalent strategy of FabricNet architecture. The authors used a CNN architecture (defined as DFE module) to extract the fabric's necessary feature. Further, they transferred the feature vectors to a stack of ensemble network (referred to as CU module) containing three deep CNN architectures. Each of the models in the ensemble network also has inter-connectivity. Therefore, each of the models in the ensemble can be triggered by other models. Although the architecture is convincing, it suffers from overfitting issue due to enormous trainable parameters (approximately 82 million). Also, it requires extensive computations, which is about 3372 million floating point operations (FLOPs). The authors conducted evaluations in a closed source dataset and achieved 74% accuracy. In comparison, our proposed architecture achieves superior accuracy with less trainable parameters and in fewer FLOPs.

Table 1 represents a comparison of various types of fiber recognition architectures that also consider computer vision framework. The present vision-based architecture is faster, accurate, non-destructive, and do not require any expertise to identify textile fibers. By analyzing the table, it can be concluded that the computer vision based strategy is more prominent than all the previous domains introduced till now.

We present an architecture that operates over a more user-friendly information extraction process. It requires a close-shot surface image of fabric to identify the raw textile fibers. We argue that close-shot surface images contain proper fabric surface properties that are enough to identify textile fibers. Furthermore, in the current new media age,

TABLE 1. The table represents a detailed comparison of various domains of fiber recognition procedures. Among the different strategies, computer-vision-based method is fast, non-destructive, and requires no expertise. The table is a modified version, which was earlier presented by Z. Feng et al. [20].

Method	Accuracy	Requiring	Time	Destructive	Identifiable fibre number
Physical	Expert's experience	Expert	> 5 min	Yes	Expert's experience
Chemical	Expert's experience	Expert, reagent	> 10 min	Yes	Expert's experience
Microscopic	Expert's experience	Expert, microscope	> 5 min	No	Expert's experience
Spectroscopy	> 0.8	Infrared spectrometer	> 5 min	No	< 20
Cut-section	> 0.7	Optical aid	> 3 min	Yes	< 10
Computer Vision Systems (CU-Net)	0.80	Camera, computation device	< 1 s (3372 million FLOPs)	No	50
Computer Vision Systems (ours)	**0.84**	**Camera, computation device**	**< 1 s (640 million FLOPs)**	**No**	**50**

image capturing is one of the simplest functions due to the well-spread of smartphones and high-definition cameras. Therefore, the architecture changes the process of textile fiber extraction from laboratories to individuals and industries. Moreover, we achieve a satisfactory recognition accuracy that is most suitable for industrial purposes.

The proposed FabricNet architecture can identify multiple fibers at once. Therefore, the classification process of the model is multi-label. The overall architecture uses CNN containing a class-based ensemble that can individually recognize a specific fiber. We investigate with various deep learning frameworks and determine that Xception [21] performs optimally amongst the existing CNN architectures. We further adapt the structure of the Xception architecture and assign our ensemble strategy. We name the revised version of the Xception architecture as FabricNet.

III. FabricNet

A. MOTIVATION

In neural network architecture, ensemble methods [22] combine multiple sub-models to obtain better accuracy. Ensemble architectures have been proven to be less prone to overfitting and generate more accurate results than basic models [23]. Furthermore, present ensemble architectures have been able to identify complex spatial information from image patterns [24]. Therefore the ensemble methods are being implemented in different scopes, including geospatial land classification [25], face recognition [26], image segmentation [27], and so on. The process of ensemble methods can be thought of as a particular situation where a group of people will always make better decisions than a single person [28]. Dieterich [29] pointed out three reasons for which the ensemble architecture may work better than traditional architectures: 1) training phase may not contain sufficient data to train the single best classifier; 2) a single algorithm may fail to converge to the global optimum, but an ensemble starting from distinct points could lead to a better approximate global optimum; 3) the space being searched may not contain any optimum position, but an ensemble may lead this space for a better optimum position.

In deep learning architecture, three types of ensemble architectures are mostly encountered, a) stacked ensembles,

b) weight average ensembles, and c) class-based ensembles. In stacked ensemble architecture, multiple sub-models receive input data and flows the data stream to a final learning model that generates the results. Mathematically, the stacked ensemble can be presented as,

$$E(x) = f(e_1(x), e_2(x), \ldots, e_n(x))$$

Where,

$$x = \text{Input data}$$
$$E(x) = \text{Stacked ensemble model}$$
$$e_i(x) = \text{Ensemble sub-models}$$
$$n = \text{Number of ensemble sub-models}$$
$$f(x) = \text{The final learning model} \tag{1}$$

In weight average ensemble architecture, the results of multiple models are calculated separately, further combined through weight multiplication calculations to perform final prediction [30]. Mathematically, the weight average ensemble can be presented as,

$$E(x) = argmax(w_1 \times e_1(x), w_2 \times e_2(x), \ldots, w_n \times e_n(x))$$

Where,

$$x = \text{Input data}$$
$$E(x) = \text{Weight average ensemble model}$$
$$e_i(x) = \text{Ensemble sub-models}$$
$$w_i = \text{Weights for each ensemble models}$$
$$n = \text{Number of ensemble sub-models} \tag{2}$$

In class-based ensemble architecture, the number of ensemble models is equal to the number of classes [31]. Each of the ensemble models only learns to identify a specific category. The class-based ensemble architecture can be derived as,

$$E(x) = argmax(e_1(x), e_2(x), \ldots, e_n(x))$$

Where,

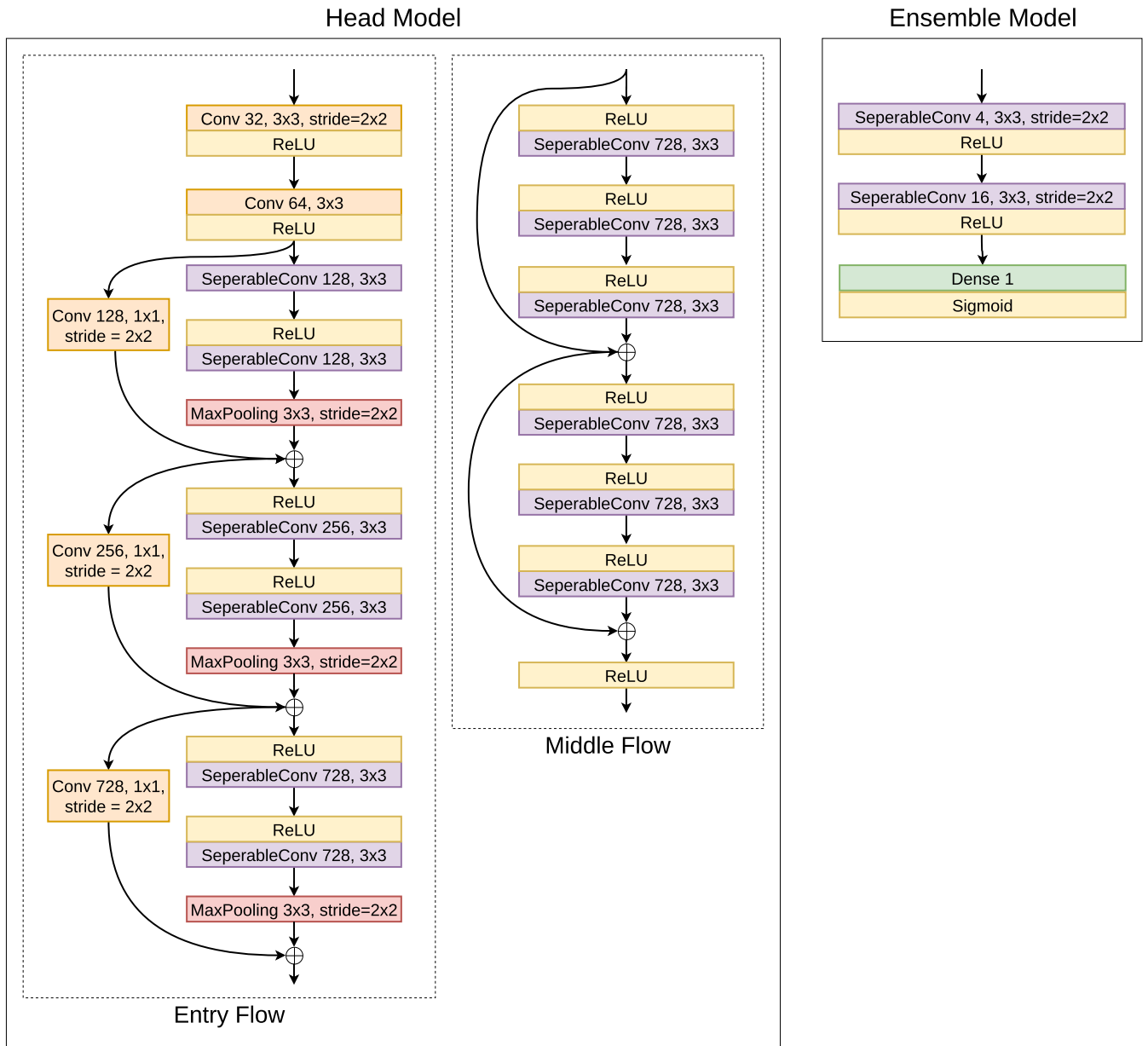$$x = \text{Input data}$$
$$E(x) = \text{Class-based ensemble model}$$
$$e_i(x) = \text{Ensemble sub-model for } i^{th} \text{ class}$$
$$n = \text{Number of output classes} \tag{3}$$

FIGURE 3. The figure illustrates the head and ensemble model of the FabricNet. The entry flow network recieves input image, and the processed data is passed to the middle flow network. It further forwards the processed data to the multiple ensemble models. Each of the convolutions is followed by a batch normalization [32] layer, not illustrated in the image.

Inspired by the performance boost of the ensemble architectures, we develop an ensemble architecture that performs mostly similar to class-based ensemble architecture. Nevertheless, our proposed architecture is slightly different than the class-based ensemble architecture. Mathematically, the proposed architecture can be derived as follows,

$$E(x) = argmax(e_1(f(x)), e_2(f(x)), \ldots, e_n(f(x)))$$

Where,

$$x = \text{Input data}$$
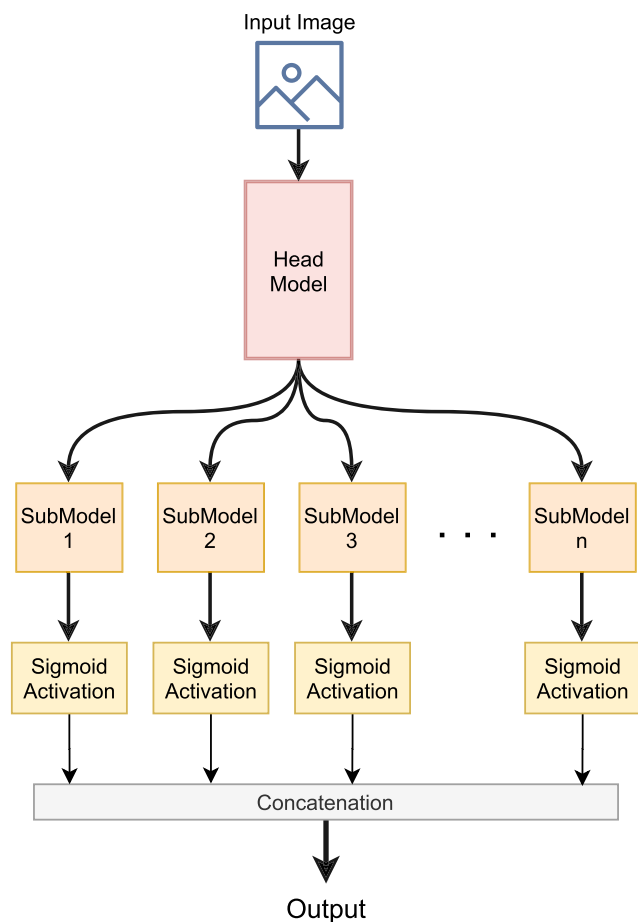$$E(x) = \text{FabricNet model}$$

$$e_i(x) = \text{Ensemble sub-model for } i^{th} \text{ class}$$
$$n = \text{Number of output classes}$$
$$f(x) = \text{Head model} \tag{4}$$

Instead of directly passing the ensembles' inputs, the proposed architecture uses an auxiliary feature extractor function defined as the head model. The head model only passes the relevant feature embeddings to the ensemble models and reduces the full dependency over the ensemble models. As a fabric can be constructed using multiple fibers, the FabricNet must output multiple classes at once, often acknowledged as multi-label classification. Therefore, the $E(x)$ function

of equation 4 may return multiple outputs at once. Also, the number of ensembles n is kept equal to the number of target classes. Furthermore, each ensemble model $e_i(x)$ specifically learns to identify a particular class. Therefore, individual ensemble models can approach an optimal state to recognize a specific class, ignoring other classes. This individuality may cause improving the accuracy of the FabricNet architecture.

FIGURE 4. The figure depicts the architectural strategy of the FabricNet model. Inputs flow through the head model, which is further passed through the class-specific ensemble submodels. Submodels contain minimal trainable parameters to avoid overfitting and reduce computational complexity.

B. ARCHITECTURE

The overall architecture of FabricNet is segmented into two parts: a head model and ensemble models. The head model directly fetches the input images and generates lower dimension embeddings. The embeddings derived by the head model are passed to the ensemble models. Each of the ensemble models is assigned to identify only a single type of fabric fibers or class, and each class's prediction is independent of the other class-based ensemble. Therefore, the number of ensemble models must be similar to the number of possible categories. Figure 4 illustrates a block diagram

of the FabricNet architecture. Using the head model poses some advantages, considering the usual class-based model. Generally, higher parameters in a CNN architecture may often cause overfitting. Passing the input through a head model causes a reduction of irrelevant features. Furthermore, it also reduces the number of trainable parameters significantly. On the whole, implementing our suggested ensemble architecture leads to the following benefits:

- Each ensemble only extracts the knowledge required to recognize a specific class. This results in acquiring an approximate optimal position for each particular class, which causes superior accuracy.
- Using the head model significantly reduces the number of parameters required for each ensemble model. This substantially reduces the FLOPs and overfitting [33].
- Apart from the general class-based ensemble, our proposed ensemble structure can be deeper with reduced computational complexity (due to head convolution non-parallelism).

Instead of implementing a new baseline architecture for the head model, we modify the existing CNN architectures. Through our investigation (illustrated in Figure 6), we affirm that Xception architecture performs superior to the currently existing popular architectures. Therefore, we fuse the ensemble methodology in Xception architecture. The Xception model counterfeits the basic properties of VGG [34] and Inception [35] network. The model utilizes a shorter kernel of the size of 3 and performs depthwise separable convolutions. The depthwise separable convolutions of the Xception architecture are performed by a depth-wise convolution followed by a pointwise convolution (in Inception architecture, the order is reversed). This strategy is based on a hypothesis that spatial feature extraction and channel-wise feature extraction procedure can be decoupled. Meanwhile, this decoupling has a huge advantage in reducing the required parameters of a convolutional layer. Xception further implements residual identity maps [36] in different layers to resolve the issue of vanishing gradient.

Xception architecture contains three types of data-flow networks. The entry flow performs depthwise separable convolutions, followed by a maxpool layer. The entry flow is followed by nine similar middle flow networks that perform depthwise separable convolution. Finally, the exit flow network performs a similar computation sequence to the entry flow, followed by a global average pooling. The model is substantially deep (126 layers), and it is often validated that deeper networks are better [34]. However, as we search for an optimal head model that will also contain ensemble models, we have to avoid over-parameterizing the head model. Over-parameterization may cause overfitting and also increase the FLOPs of the architecture.

Hence, we expel unnecessary blocks from the Xception architecture, which does not boost the prediction accuracy. While expelling the exit flow and some middle flow networks of the Xception model, we found a minimal fluctuation of F1-score and AUC score (reported in Figure 7). The metrics'

steadiness indicates that most of the lower layers are not necessary (for the experimental dataset), and they can be removed. Therefore, we only adopt the entry flow and two middle flow stacks of the Xception architecture as our head model. Also, removing the lower portion of the Xception architecture reduces trainable parameters by more than 80%. A full investigation is reported in the Result Analysis section (Section IV-D).

The architectural specifications of the head and ensemble models of the FabricNet is reported in Figure 3. Moreover, Figure 4 illustrates the overall flow of the model. We implemented separable convolutions in the ensemble model to keep the learnable parameters limited. Each ensemble model contains a single dense node with a sigmoid activation function that works as the final activation for each category. The available outputs of each ensemble are further concatenated to produce the final output of the FabricNet model. We broadly analyze and discuss our FabricNet architecture findings in the Result Analysis (Section IV-D), particularly the ensemble architecture.

As the FabricNet architecture performs a multi-label classification task, the final output contains a sigmoid activation function. Nam *et al.* has suggested that cross-entropy loss is the best choice for multi-label classification tasks [37], that is defined as follows,

$$L_{CE}(y, o) = - \left(\sum_{l} (y_l \, log \, o_l) - (1 - y_l) \, log(1 - o_l) \right)$$

$$Where,$$

$$L_{CE} = \text{Loss function}$$
$$o_l = \text{Prediction for label } l$$
$$y_l = \text{Target for label } l \tag{5}$$

Therefore, the model is trained using the aforementioned cross-entropy loss function.

IV. EXPERIMENTAL RESULTS
A. THE FABRICS DATASET
As this is a prior work investigating multiple fibers, we currently found only one dataset suitable for the experiment. The fabric dataset contains around 8000 images of different fabrics, and garments [2]. However, we found a total of 7553 images suitable for the experiment. Although the original work conducted using the dataset contained only 2000 images of fabric surfaces, the current repository contains an increased number of surface images. The dataset contains images with various lightning and orientation to make the recognition process more challenging. Figure 2 contains an example of the images. Further, Each of the fabric images contains one or more classes. The classes are the fibers that are used to construct the fabric. These fabrics contain a total of fifty types of fibers that constituted the fabrics. Figure 5 illustrates the class distribution for the images of the dataset. The dataset collectors only attempted to identify fabrics that were not blended (contained
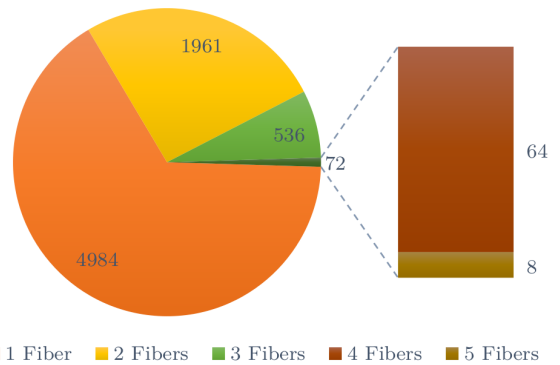

FIGURE 5. A pie chart that represents the number of images belonging to the number of fiber classes.

a single fiber) [2]. Therefore, the collectors were able to identify nine types of non-blended fabrics from the dataset.

B. EVALUATION METRICS
To evaluate and compare the results, three evaluation metrics have been used, which are presented as follows:

Accuracy: Accuracy is the simplest form of evaluation. It formally defines the ratio of correct predictions over total experiments. In multi-label classification, we consider a single prediction is accurate if all of the classes are correctly guessed. Accuracy can be defined as follows,

$$Accuracy = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \tag{6}$$

Precision: Precision is also named as the positive predictive value (or true positive rate) of a system that reports the ratio of correctly predicted positive cases over total predicted positive cases. It can be represented as,

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

F_1-Score: F_1 score represents the weighted average of precision and recall. By choosing the weight value as 2, the F_1 score can be presented as,

$$F_1 Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{8}$$

AUC Score: Area under the curve (AUC) and receiver operating characteristics (ROC) curve defines how well a model converges towards distinguishing classes accurately. In general, the metric generates a curve, where AUC represents the area under the ROC curve. In overall experiments, we use 200 thresholds to discretize the ROC curve.

FLOPs: Floating point operations (FLOPs) measures the number of arithmetic operations required to execute a single instance of a deep learning model. Models requiring higher FLOPs have higher time complexity.

Accuracy, Precision, F1-score, and AUC score generate results in the range [0, 1], whereas higher score points better performance of a system. Hence, we use the metrics mentioned above to prove the effectiveness of our model.

Moreover, we use FLOPs to measure the time complexity of each model.

C. EXPERIMENTAL SETUP

The evaluation architectures were implemented using Tensorflow [38], Keras [39], scikit-learn [40], and NumPy [41]. To lessen the architecture's bias and correctly measure each architecture's accuracy, k-fold cross-validation is performed [42]. All of the evaluations are conducted by selecting the value of $k = 4$. Therefore, the dataset is split into 50%-25%-25% train, validation, and test subsets. The reported measurements are the best performance on the validation set for a particular fold, further evaluated on the unseen test set. Each architecture is trained using $batchSize = 128$ with a maximum epoch limit of 100. With a learning rate of 0.001, Adam optimizer is used to train each of the architectures. All of the tested architectures are initialized with ImageNet trained weights, and they are further trained on the Fabrics dataset. The input image is in the shape of $120 \times 120 \times 3$. As the training dataset is small, we implemented image augmentation considering some common augmentation process that includes brightness change, contrast change, zooming, cropping, and channel shifts. We did not perform any geometrical distortions as it may change the texture pattern of the fabric weave. Every exhibited result is calculated as the mean and standard deviation for three runs for each fold (total 3×4 runs).

D. RESULT ANALYSIS

As fabric surface images are less exploited for fabric fiber recognition, we compare the FabricNet architecture with various computer vision based architecture. Also, we include the existing architecture CU-Net [20] that also operates over fabric surface image. We have implemented the DenseNet baseline for the CU-Net architecture.

Figure 6 illustrates the F1-scores obtained on the validation set while training the existing DCNN frameworks in the fabric dataset. The graphs report that Xception architecture achieves a better result on the validation set. VGG architectures do not perform adequately in the framework, mostly due to the vanishing gradient problem. On the contrary, ResNets solve the vanishing gradient problem by implementing residual identity maps. Yet they don't acquire satisfactory results mostly due to overfitting. InceptionResNet architecture achieves better results due to the proper integration of residuals and inception blocks. DenseNets require fewer parameters than Xception architecture; still, the idea of the shorter connection from input and output does not help to achieve better performance. MobileNet architectures require the least number of parameters than the other implemented models. Nevertheless, they fail to perform fiber recognition at an acceptable rate. This low F1-score of MobileNets may indicate that the issue is not with training parameters, rather than a necessity of optimal network architecture.

The actual Xception architecture consists of one entry flow, nine middle flows, and an exit flow network. Each

TABLE 2. The table illustrates the F1-score of the validation and test dataset corresponding to the different ensemble architectures. Each layer is represented as '{Sx, y, z}', where 'x' is the number of filters, 'y' is the kernel size, and 'z' is the stride. 'S' defines a depthwise separable convolution layer. Each convolution is followed by batch normalization and ReLU activation function. Trainable parameters (for each ensemble) are presented in thousand.

Architecture	Parameters (thousand)	F1-score (val)	F1-score (test)
{S64,3,2},{S64,3,2}	58	0.883±0.01	0.80±0.01
{S32,3,2},{S64,3,1}	33	0.884±0.01	0.828±0.01
{S16,3,2},{S32,3,1}	19	0.884± 0.0	0.874±0.01
{S8,3,2},{S8,3,1}	12	0.893±0.02	0.887±0.02
{S4,3,2},{S16,3,2}	**9**	**0.922±0.002**	**0.902±0.01**
{S2,3,2},{S16,3,2}	8	0.891±0.01	0.889±0.02
{S4,3,2},{S16,3,2}{S16,3,2}	9	0.90±0.01	0.898±0.02

flow network contains residual connections. To select a proper head model, we further investigate the Xception architecture with different flow settings. Figure 7 represents a validation test score of the Xception architecture tuning the number of entry flow, middle flow, and exit flow segments. Only selecting the entry flow causes the validation F1-score and AUC to decrease. Choosing different numbers of middle flow layers improves the validation score by 0.2. However, for a different number of middle flow blocks, the score remains nearly constant. We select an entry flow with two middle flow network as the head architecture of the FabricNet. Although setting six intermediate flow blocks with an entry block acquires the highest score, the improvement is negotiable. As we implement a class-wise ensemble network that will contain more parameters (in the ensemble model), we avoid over-parameterizing the head model. Avoiding over-parameterization causes the overall architecture to be less prone to overfitting. Furthermore, using one entry and two middle blocks as the head model decreases the number of the trainable parameter by 80% compared to Xception architecture. Also, reducing the number of parameters causes the reduction of FLOPs by 60%, corresponding to Xception architecture.

We further investigate for selecting the optimal ensemble model. However, it is considered to keep the ensemble network's trainable parameters as less as possible to avoid overfitting and for easy training. Hence, we search for an optimal shallow architecture as an ensemble model. Shallow architecture requires fewer parameters, and as a result, it is possible to greatly increase the number of networks in the ensemble, based on the output classes. Therefore, we investigate CNN architectures with no more than three layers. Table 2 exhibits our experiment with different ensemble architecture with the reported F1-score on validation and test dataset. In the ensemble architecture, only separable convolution is implemented as it initiates fewer parameters. A close relationship between the trainable parameters and overfitting can be observed by investigating the table data. Higher training parameters in the ensemble model results in overfitting in the validation data. By decreasing the number of training parameters, a reduction in overfitting can be
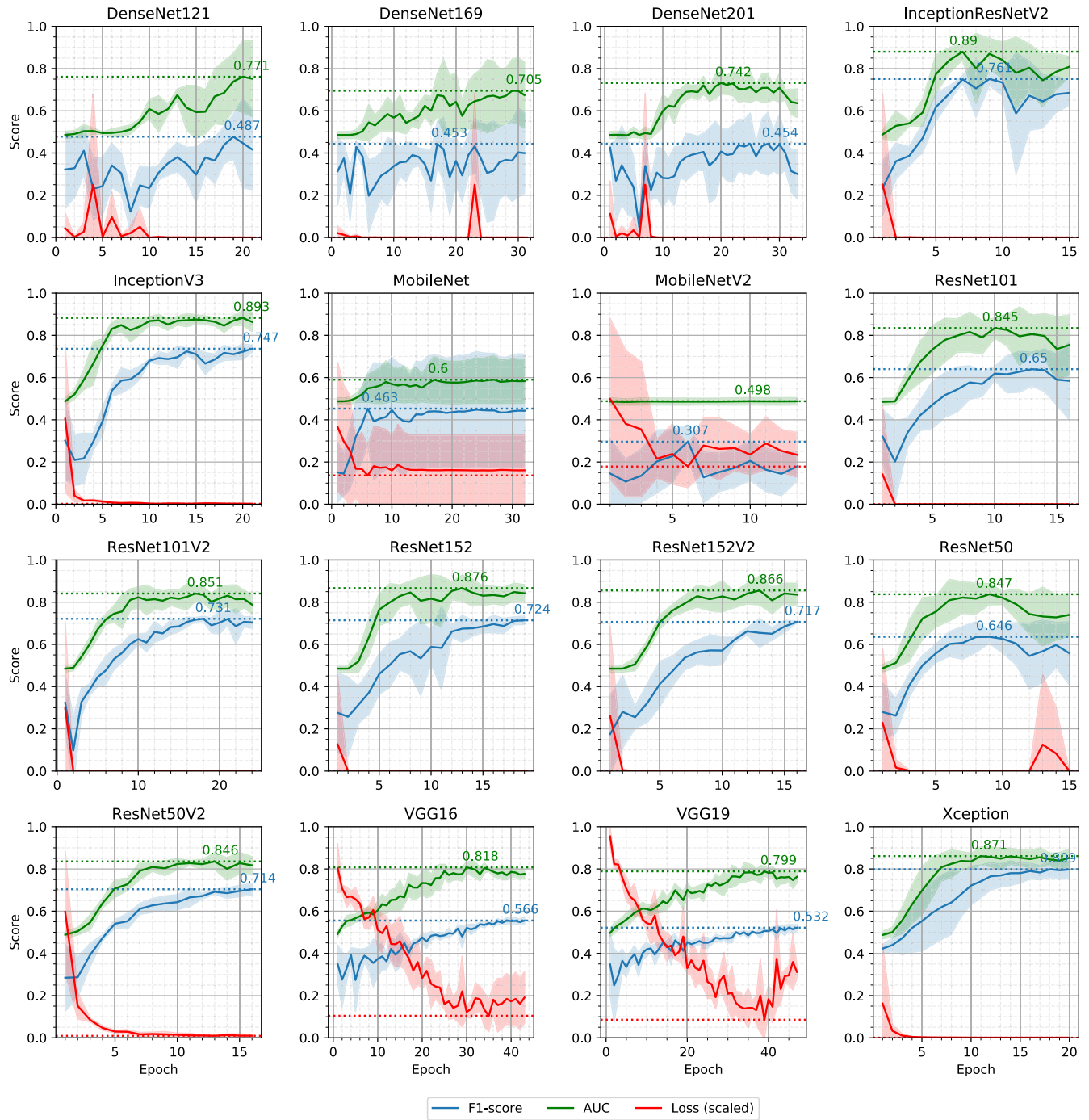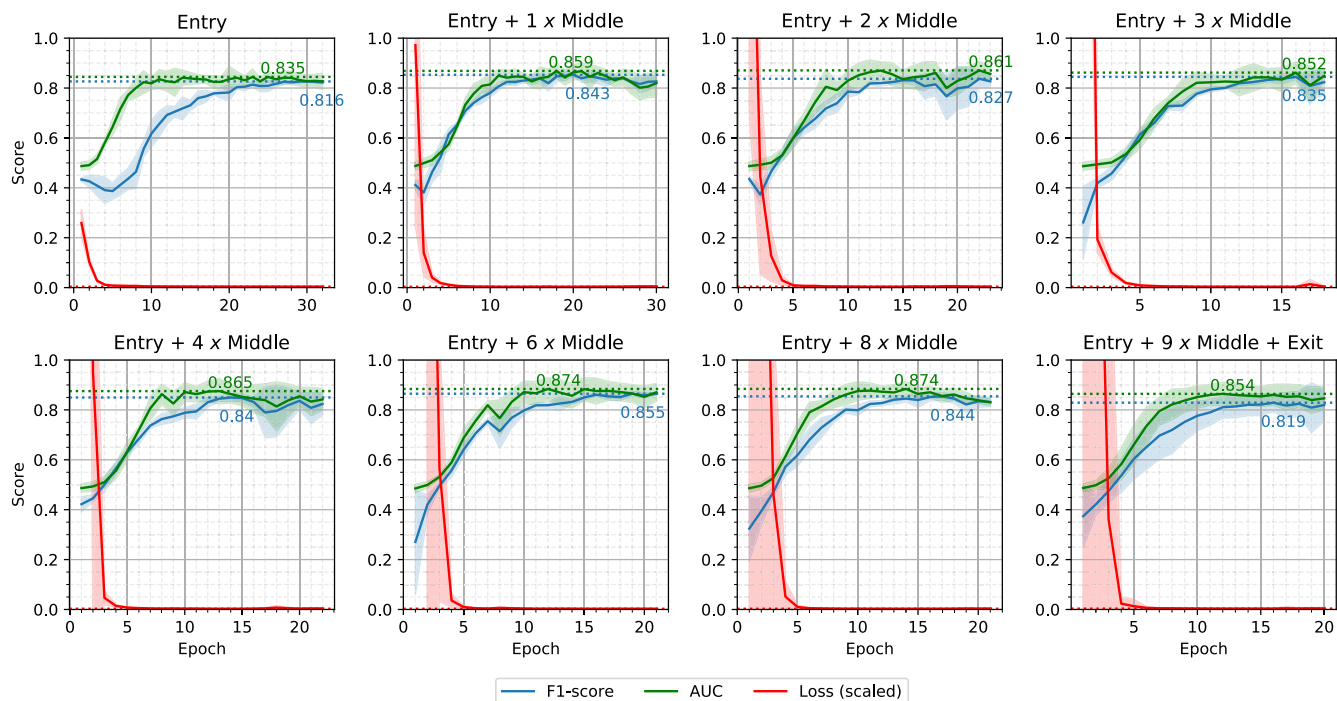
FIGURE 6. Each graph represents the loss (min-max normalized), AUC, and F1-score on the validation dataset calculated on the existing DCNN architectures' training procedure. The horizontal axis represents the training epochs, while the vertical axis represents the metric score. The train and test scores of the corresponding architectures are reported in Table 3. Zoom in for a better view.

observed. However, after a certain period, the score drops. Adding additional layers does not heavily improve the score. Thus, each of the ensemble architecture is implemented using two depthwise seperable convolutions followed by a single fully-connected node. A sigmoid activation function is used as our target output may contain multiple classes at once.

Figure 8 represents a comparison (on validation dataset) of FabricNet and the CU-net architecture. Both of the architectures follow ensemble strategy. However, the architectures' influential differences are: 1) improper class distribution for each ensemble model and 2) the number of parameters for each ensemble model. In the case of CU-Net architecture, the output of each class is generated based on the ensemble

FIGURE 7. Each graph represents the loss, AUC, and F1-score on the validation dataset calculated using different Xception architecture setups. Entry, middle, and end define the three types of network flows of Xception architecture. The number of middle flows is indicated by multiplication. The 'entry+2×middle' design is used as the head model.
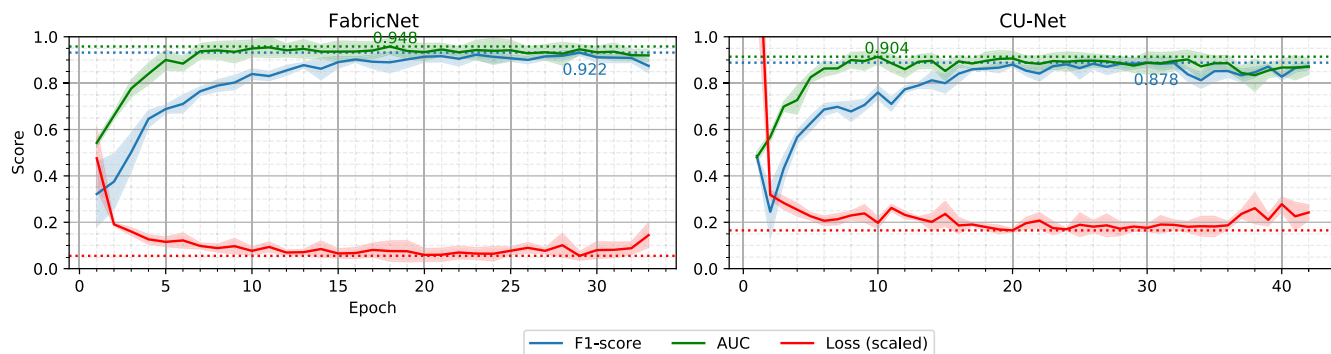
FIGURE 8. The figure illustrates a comparison of FabricNet architecture with CU-Net architecture. Both architectures are based on ensemble strategy. However, FabricNet architecture acquires higher performance as it contains class-specific models in the ensemble.
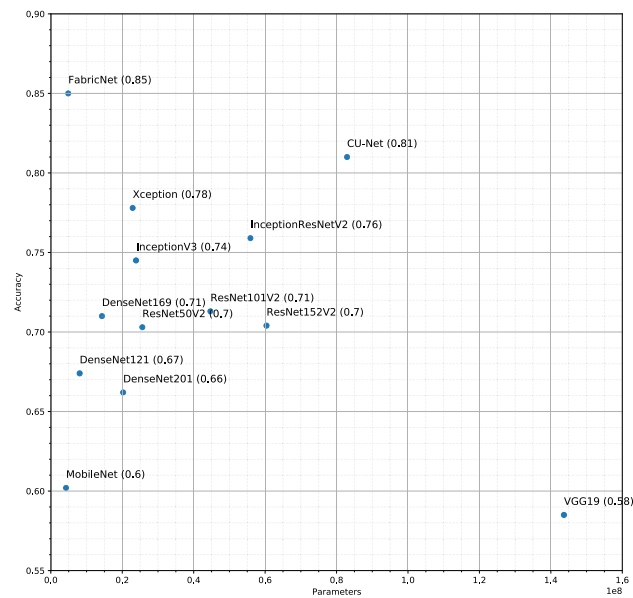
framework's decision. However, the architecture does not define specific ensemble models for each specific class. Hence, a single binary output of each class can be easily biased by multiple ensemble models. On the contrary, FabricNet architecture contains shallow CNN models specifically for each class. Hence, FabricNet avoids biased outputs for each class. Also, in comparison to CU-Net architecture, FabricNet requires a low number of trainable parameters. Lower parameters solve the overfitting issue, also models with lesser parameters are more comfortable to train.

The FabricNet architecture (network illustration in Figure 3) is compared with the DCNN architectures, presented in Table 3. The comparison reports the precision, accuracy, F1-score, and AUC score of all the architectures

calculated on the train, validation, and test dataset. The table represents the improvement of FabricNet architecture from the general implementation of Xception architecture. Although the FabricNet is a subset of the Xception architecture, joining the ensemble models boosts the F1-score of the FabricNet architecture by approximately 0.7. Figure 10 represents a scatter plot of the tested architectures and the FabricNet. The horizontal axis indicates the number of training parameters, and the vertical axis reports the F1-score. As to calculate the number of trainable parameters and the FLOPs of the FabricNet architecture, the whole ensemble of 50 classifiers is considered. The FabricNet architecture achieves the highest accuracy while keeping the training parameters at a limit of 4.8 million. On the

TABLE 3. The table reports a comparison between the FabricNet and other well-performing architectures on train validation and test dataset. The Parameters column represents the total number of trainable parameters required for each model.
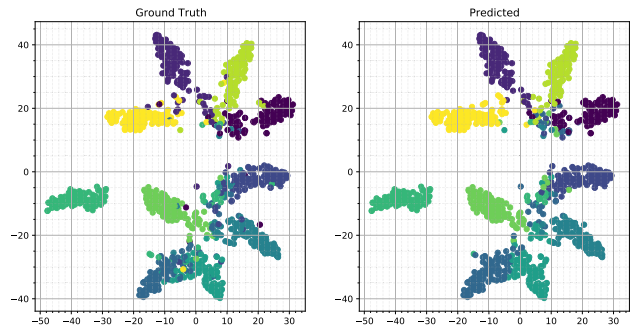
Architecture	Parameters (million)	FLOPs (million)	Train				Validation				Test			
			Precision	Accuracy	F_1-Score	AUC	Precision	Accuracy	F_1-Score	AUC	Precision	Accuracy	F_1-Score	AUC
MobileNetV2 [43]	3	96	0.964±0.0	0.851±0.01	0.957±0.003	0.998±0.0	0.718±0.235	0.394±0.077	0.307±0.069	0.498±0.007	0.553±0.01	0.494±0.01	0.418±0.01	0.503±0.01
VGG19 [34]	143	5533	0.762±0.015	0.658±0.01	0.641±0.029	0.962±0.012	0.669±0.005	0.574±0.006	0.532±0.005	0.799±0.014	0.696±0.01	0.585±0.01	0.528±0.01	0.809±0.01
VGG16 [34]	138	4356	0.79±0.001	0.746±0.01	0.689±0.014	0.973±0.006	0.686±0.017	0.585±0.0	0.566±0.005	0.818±0.016	0.687±0.01	0.583±0.01	0.561±0.01	0.844±0.01
DenseNet201 [44]	20	1134	0.724±0.086	0.765±0.01	0.594±0.136	0.839±0.085	0.603±0.084	0.528±0.007	0.454±0.138	0.742±0.043	0.809±0.01	0.662±0.01	0.676±0.01	0.85±0.01
DenseNet121 [44]	8	783	0.787±0.029	0.79±0.01	0.697±0.049	0.955±0.009	0.644±0.092	0.506±0.077	0.487±0.095	0.771±0.159	0.778±0.01	0.674±0.01	0.677±0.01	0.918±0.01
DenseNet169 [44]	14	903	0.72±0.055	0.782±0.01	0.583±0.103	0.834±0.143	0.604±0.065	0.532±0.059	0.453±0.094	0.705±0.139	0.85±0.01	0.71±0.01	0.753±0.01	0.918±0.01
ResNet152 [36]	60	3594	0.991±0.002	0.854±0.008	0.988±0.001	0.999±0.0	0.792±0.032	0.663±0.017	0.724±0.01	0.876±0.052	0.799±0.047	0.681±0.006	0.742±0.002	0.908±0.027
ResNet50 [36]	25	1197	0.98±0.002	0.856±0.003	0.974±0.006	0.998±0.0	0.754±0.021	0.632±0.054	0.646±0.02	0.847±0.04	0.828±0.002	0.698±0.005	0.747±0.003	0.896±0.017
ResNet101 [36]	44	2413	0.979±0.005	0.853±0.005	0.974±0.007	0.999±0.001	0.743±0.079	0.617±0.05	0.65±0.07	0.845±0.038	0.839±0.008	0.709±0.006	0.744±0.002	0.888±0.011
ResNet152V2 [45]	60	3481	0.982±0.006	0.856±0.01	0.978±0.006	0.999±0.0	0.779±0.033	0.642±0.023	0.717±0.006	0.866±0.059	0.856±0.01	0.704±0.01	0.782±0.01	0.871±0.01
ResNet101V2 [45]	44	2300	0.996±0.006	0.852±0.01	0.995±0.007	0.999±0.0	0.819±0.008	0.688±0.003	0.731±0.002	0.851±0.027	0.85±0.01	0.713±0.01	0.786±0.01	0.879±0.01
ResNet50V2 [45]	25	1085	0.99±0.006	0.863±0.01	0.987±0.006	0.999±0.0	0.809±0.022	0.668±0.019	0.714±0.017	0.846±0.029	0.86±0.01	0.703±0.01	0.754±0.01	0.874±0.01
InceptionV3 [46]	23	586	0.979±0.005	0.871±0.01	0.975±0.004	0.998±0.01	0.783±0.023	0.683±0.023	0.747±0.007	0.893±0.025	0.87±0.01	0.745±0.01	0.815±0.01	0.898±0.01
MobileNet [47]	4	146	0.979±0.007	0.857±0.004	0.974±0.01	0.999±0.0	0.734±0.172	0.508±0.161	0.463±0.158	0.6±0.101	0.684±0.146	0.602±0.098	0.589±0.175	0.617±0.119
InceptionResNetV2 [48]	55	1258	0.997±0.009	0.849±0.01	0.994±0.007	0.999±0.001	0.845±0.048	0.717±0.005	0.761±0.021	0.89±0.028	0.864±0.01	0.759±0.01	0.813±0.01	0.907±0.01
Xception [21]	22	1424	0.999±0.004	0.871±0.001	0.999±0.004	0.999±0.0	0.889±0.026	0.763±0.021	0.839±0.025	0.874±0.036	0.916±0.017	0.778±0.007	0.866±0.025	0.894±0.008
CU-Net [20]	82	3372	0.999±0.009	0.903±0.01	0.999±0.009	0.999±0.0	0.904±0.005	0.782±0.006	0.878±0.005	0.904±0.02	0.895±0.01	0.807±0.01	0.856±0.01	0.897±0.01
FabricNet(ours)	**4.8**	**640**	**0.999±0.009**	**0.879±0.01**	**0.999±0.009**	**0.999±0.0**	**0.929±0.001**	**0.855±0.001**	**0.922±0.002**	**0.948±0.023**	**0.912±0.01**	**0.846±0.01**	**0.902±0.01**	**0.932±0.01**

FIGURE 9. A scatter plot illustrating the test accuracy scores (vertical axis) w.r.t. the number of trainable parameters (horizontal axis). Zoom in for a better view.

FIGURE 10. The figures illustrate scatter-plots generated by the head model of FabricNet, on a small portion of the dataset (with ten classes). The head model tries to assume the classes, whereas the ensembles try to memorize the head ensemble's missed out features. Mixing the head and ensemble model enables the architecture to perform better with more class-specific feature memorization.

Figure 10 illustrates the embedding vector outputs generated by the head model of the FabricNet architecture on a small subset of the input. It can be anticipated by analyzing the figure that the head model extricates some of the necessary features. Further, the ensemble's class-specific models furnish the embeddings based on the class-specific features. In such a case, the ensemble has the primary advantage of correcting the head model's erroneous guesses and further fix the issue through memorization.

V. CONCLUSION

The paper presents an architecture FabricNet, a textile fiber recognition scheme, that can recognize multiple fibers at once by only processing the surface image of fabrics. This research work points to an immense improvement in fiber recognition tasks as the previous methods required microscopic images and spectrographs of fibers. The FabricNet is implemented based on a new idea of ensemble architecture, and to outline the difference, the paper comprises an investigation of mostly implemented ensemble architectures. The experiment is conducted using fifty types of textile fibers, and the FabricNet outperforms most of the well-known image classification architectures. We strongly believe that the overall contribution of this paper inaugurates a broader perception in the scope of image pattern recognition and industrial fiber identification research works.

contrary, the MobileNet and MobileNetV2 architectures fall behind in achieving a better score with comparable training parameters. It is a clear indication that FabricNet architecture gains superiority due to the CNN network topology.

Residual architectures (ResNet, Xception, Inception) indirectly implements the property of the ensemble strategy. Residuals not only help to solve the vanishing gradient problem but also can ignore a particular CNN block if required. However, the ignore state of a CNN block depends on the type of input, and it is adequately utilized using backpropagation. Yet, the difference between the residual ensemble and our implemented ensemble lies in the dedicated path (i.e., a layer sequence is fully dedicated to a class). Therefore each of the filters only looks for class-specific features. The dedicated feature extraction solves separating the depth filters for a class-specific identity extraction mostly occurred at the deepest layers (before fully connected layer) of a CNN architecture.

REFERENCES

[1] G. Stylios, "Engineering, re-engineering and reverse engineering of textiles or textile genetic engineering," *Int. J. Clothing Sci. Technol.*, vol. 10, no. 2, pp. 89–90, May 1998.

[2] C. Kampouris, S. Zafeiriou, A. Ghosh, and S. Malassiotis, "Fine-grained material classification using micro-geometry and reflectance," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 778–792.

[3] H. Y. Ngan, G. K. Pang, and N. H. Yung, "Automated fabric defect detection—A review," *Image Vis. Comput.*, vol. 29, no. 7, pp. 442–458, 2011.

[4] J. Hearle, P. Grosberg, and S. Backer, *Structural Mechanics of Fibers, Yarns, and Fabrics*, vol. 1. Hoboken, NJ, USA: Wiley, 1969.

[5] M. Goodway, "Fiber identification in practice," *J. Amer. Inst. Conservation*, vol. 26, no. 1, pp. 27–44, Jan. 1987.

[6] R. Drobina and M. S. Machnio, "Application of the image analysis technique for textile identification," *AUTEX Res. J.*, vol. 6, no. 1, pp. 40–48, 2006.

[7] X. Sun, M. Zhou, and Y. Sun, "Classification of textile fabrics by use of spectroscopy-based pattern recognition methods," *Spectrosc. Lett.*, vol. 49, no. 2, pp. 96–102, Feb. 2016.

[8] H. Chen, Z. Lin, and C. Tan, "Classification of different animal fibers by near infrared spectroscopy and chemometric models," *Microchem. J.*, vol. 144, pp. 489–494, Jan. 2019.

[9] L. Ying, L. Dai, J. Wu, and G. Sun, "Single fiber identification of cotton/flax fabric based on longitudinal view of microscopic fiber images," *J. Textile Res.*, no. 4, p. 2, Dec. 2012.

[10] P. D. E. A. N. Khan, N. Abir, M. A. N. Rakib, E. M. S. Bhuiyan, and M. R. Howlader, "A review paper on textile fiber identification," *IOSR J. Polym. Textile Eng.*, vol. 04, no. 02, pp. 14–20, Apr. 2017.

[11] B. P. Corbman, *Textiles: Fiber to Fabrics*. New York, NY, USA: McGraw-Hill, 1983.

[12] L. C. Sawyer, D. T. Grubb, and G. F. Meyers, "Applications of microscopy to polymers," *Polym. Microsc.*, vol. 4, pp. 248–434, Dec. 2008.

[13] M. Kinoshita, Y. Hashimoto, R. Akiyama, and S. Uchiyama, "Determination of weave type in woven fabric by digital image processing," *J. Textile Mach. Soc. Jpn.*, vol. 35, no. 2, pp. 1–4, 1989.

[14] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[15] S.-H. Chiu, J.-Y. Chen, and J.-H. Lee, "Fiber recognition and distribution analysis of PET/Rayon composite yarn cross sections using image processing techniques," *Textile Res. J.*, vol. 69, no. 6, pp. 417–422, Jun. 1999.

[16] P. Werbos, "Beyond regression: New tools for prediction and analysis in the behavioral sciences," Ph.D. dissertation, Dept. Appl. Math., Harvard Univ., Cambridge, MA, USA, 1974.

[17] B. Xu, B. Dong, and Y. Chen, "Neural network technique for fiber image recognition," *J. Ind. Textiles*, vol. 36, no. 4, pp. 329–336, Apr. 2007.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[19] X. Wang, Z. Chen, G. Liu, and Y. Wan, "Fiber image classification using convolutional neural networks," in *Proc. 4th Int. Conf. Syst. Informat. (ICSAI)*, Nov. 2017, pp. 1214–1218.

[20] Z. Feng, W. Liang, D. Tao, L. Sun, A. Zeng, and M. Song, "CU-net: Component unmixing network for textile fiber identification," *Int. J. Comput. Vis.*, vol. 127, no. 10, pp. 1443–1454, Oct. 2019.

[21] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1251–1258.

[22] L. K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 10, pp. 993–1001, Oct. 1990.

[23] S. Tao, "Deep neural network ensembles," in *Proc. Int. Conf. Mach. Learn., Optim., Data Sci.* Cham, Switzerland: Springer, 2019, pp. 1–12.

[24] J. Xu, W. Wang, H. Wang, and J. Guo, "Multi-model ensemble with rich spatial information for object detection," *Pattern Recognit.*, vol. 99, Mar. 2020, Art. no. 107098.

[25] R. Minetto, M. Pamplona Segundo, and S. Sarkar, "Hydra: An ensemble of convolutional neural networks for geospatial land classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 9, pp. 6530–6541, Sep. 2019.

[26] C. Ding and D. Tao, "Trunk-branch ensemble convolutional neural networks for video-based face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 1002–1014, Apr. 2018.

[27] T. Y. Tan, L. Zhang, C. P. Lim, B. Fielding, Y. Yu, and E. Anderson, "Evolving ensemble models for image segmentation using enhanced particle swarm optimization," *IEEE Access*, vol. 7, pp. 34004–34019, 2019.

[28] C. M. Bishop, *Neural Networks for Pattern Recognition*. London, U.K.: Oxford Univ. Press, 1995.

[29] T. G. Dietterich, "Ensemble methods in machine learning," in *Proc. Int. Workshop Multiple Classifier Syst.* Berlin, Germany: Springer, 2000, pp. 1–15.

[30] E. Acar, "Effect of error metrics on optimum weight factor selection for ensemble of metamodels," *Expert Syst. Appl.*, vol. 42, no. 5, pp. 2703–2709, Apr. 2015.

[31] T. Al-Khateeb, M. M. Masud, K. M. Al-Naami, S. E. Seker, A. M. Mustafa, L. Khan, Z. Trabelsi, C. Aggarwal, and J. Han, "Recurring and novel class detection using class-based ensemble for evolving data stream," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 10, pp. 2752–2764, Oct. 2016.

[32] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*. [Online]. Available: http://arxiv.org/abs/1502.03167

[33] A. Maida, "Cognitive computing and neural networks: Reverse engineering the brain," in *Handbook Statistics*, vol. 35. Amsterdam, The Netherlands: Elsevier, 2016, pp. 39–78.

[34] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: http://arxiv.org/abs/1409.1556

[35] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[36] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[37] J. Nam, J. Kim, E. L. Mencía, I. Gurevych, and J. Fürnkranz, "Large-scale multi-label text classification—Revisiting neural networks," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*. Berlin, Germany: Springer, 2014, pp. 437–452.

[38] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous distributed systems," 2016, *arXiv:1603.04467*. [Online]. Available: http://arxiv.org/abs/1603.04467

[39] F. Chollet. (2015). *Keras*. [Online]. Available: https://github.com/fchollet/keras

[40] F. Pedregosa, G. Varoquaux, and A. Gramfort, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Oct. 2011.

[41] S. van der Walt, S. C. Colbert, and G. Varoquaux, "The NumPy array: A structure for efficient numerical computation," *Comput. Sci. Eng.*, vol. 13, no. 2, pp. 22–30, Mar. 2011.

[42] F. Mosteller and J. W. Tukey, "Data analysis, including statistics," *Handbook social Psychol.*, vol. 2, pp. 203–2080, Dec. 1968.

[43] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.

[44] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.

[45] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2016, pp. 630–645.

[46] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826.

[47] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*. [Online]. Available: http://arxiv.org/abs/1704.04861

[48] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, p. 4.

ABU QUWSAR OHI is currently pursuing the B.Sc. degree from the Department of Computer Science and Engineering, Bangladesh University of Business and Technology. He is also working as a Research Assistant with the Bangladesh University of Business and Technology. He is a Competitive Programmer and has attained more than ten Bangladeshi national programming contests, including National Collegiate Programming Contest (NCPC) and International Collegiate Programming Contest (ICPC). He is passionate about learning fundamental algorithms, including data structures, dynamic programming, and game theory. He has working experience in famous languages, including Python, C++, and C. Moreover, he is proficient in prominent frameworks, including TensorFlow, Keras, NumPy, and Matplotlib. His research interests include deep learning algorithms, pattern recognition, computer vision, and reinforcement learning.

M. F. MRIDHA (Member, IEEE) received the Ph.D. degree in AI/ML from Jahangirnagar University, in 2017. He joined the Department of Computer Science and Engineering, Stamford University Bangladesh, in June 2007, as a Lecturer, where he was promoted a Senior Lecturer and an Assistant Professor, in October 2010 and October 2011, respectively. Then, he joined UAP, in May 2012, as an Assistant Professor. He is currently working as an Associate Professor with the Department of Computer Science and Engineering, Bangladesh University of Business and Technology. He also worked as a Faculty Member of the CSE Department, University of Asia Pacific, and as a Graduate Coordinator, from 2012 to 2019. His research experience, within both academia and industry, results in over 80 journal and conference publications. His research interests include artificial intelligence (AI), machine learning, deep learning, big data analysis, and natural language processing (NLP). For more than ten years, he has been with the master's and undergraduate students, as a Supervisor of their thesis work. He has served as a program committee member of several international conferences/workshops. He also served as an associate editor for several journals.

MD. ABDUL HAMID was born in Sonatola, Pabna, Bangladesh. His education life spans over different countries in the world. He received the B.E. degree in computer and information engineering from International Islamic University Malaysia (IIUM), from 1996 to 2001, and the combined master's-Ph.D. degree majoring in information communication from the Computer Engineering Department, Kyung Hee University, South Korea, in August 2009. He has been in the teaching profession throughout his life, which also spans over different parts of the globe. He was a Lecturer with the Computer Science and Engineering Department, Asian University of Bangladesh, Dhaka, Bangladesh, from 2002 to 2004. He was an Assistant Professor with the Department of Information and Communications Engineering, Hankuk University of Foreign Studies (HUFS), South Korea, from 2009 to 2012. He was an Assistant Professor with the Department of Computer Science and Engineering, Green University of Bangladesh, from 2012 to 2013. He was an Assistant Professor with the Department of Computer Engineering, Taibah University, Madinah, Saudi Arabia, from 2013 to 2016. He was an Associate Professor with the Department of Computer Science, Faculty of Science and Information Technology, American International University-Bangladesh, Dhaka, from 2016 to 2017. He was an Associate Professor and a Professor with the Department of Computer Science and Engineering, University of Asia Pacific, Dhaka, from 2017 to 2019. Since 2019, he has been a Professor with the Department of Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia. His research interests include network/cyber-security, natural language processing, machine learning, wireless communications, and networking protocols.

MUHAMMAD MOSTAFA MONOWAR (Member, IEEE) received the B.Sc. degree in computer science and information technology from the Islamic University of Technology (IUT), Bangladesh, in 2003, and the Ph.D. degree in computer engineering from Kyung Hee University, South Korea, in 2011. He is currently working as an Associate Professor with the Department of Information Technology, King Abdulaziz University, Saudi Arabia. He also worked as a Faculty Member of the Department of Computer Science and Engineering, University of Chittagong, Bangladesh. His research interests include wireless networks, mostly ad hoc, sensor, and mesh networks, including routing protocols, MAC mechanisms, IP and transport layer issues, cross-layer design, and QoS provisioning; security and privacy issues; and natural language processing. He has served as a program committee member of several international conferences/workshops. He served as an Editor for a couple of books published by CRC Press and Taylor & Francis Group. He also served as a guest editor for several journals.

FARIS A. KATEB received the bachelor's degree in computer science from Taibah University, the master's degree in computer science from the University of Colorado at Boulder, and the Ph.D. degree in computer science from the University of Colorado at Colorado Springs. He is currently an Assistant Professor with the Department of Information Technology, FCIT, KAU. He works on text classification (such as Twitter) and applies deep learning to specific object recognition in the computer vision field.

● ● ●