IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 25 January 2024, accepted 10 February 2024, date of publication 21 February 2024, date of current version 27 February 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3367429

RESEARCH ARTICLE

Optimizing Target Recognition in Synthetic Aperture Radar Imagery: A Hyperparameter-Tuned Approach With Iterative Transfer Learning and Branched-Convolutional Neural Network

BILEESH PLAKKAL BABU AND SWATHI JAMJALA NARAYANAN, (Member, IEEE)
School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, Tamil Nadu 632014, India

Corresponding author: Swathi Jamjala Narayanan (jnswathi@vit.ac.in)

This work was supported by the Vellore Institute of Technology.

ABSTRACT Real-world deployment of Automatic Target Recognition (ATR) in Synthetic Aperture Radar (SAR) often faces challenges due to the computational demands of Convolutional Neural Networks (CNNs). This paper proposes an innovative solution, combining Iterative Transfer Learning (ITL) with a lightweight branched-CNN architecture, to address these limitations. The proposed approach cleverly decomposes the multi-class classification problem into smaller, binary subtasks. Each branch in the network, consisting of specialized Fully Connected (FC) layers, acts as an expert in identifying a specific target class. These branches are trained sequentially, focusing on one class at a time using a One-vs-All (OVA) strategy. This simplification reduces the model's complexity, enabling efficient performance even with a smaller CNN. Furthermore, the branched architecture significantly alleviates the need for a large labeled dataset. By dividing the problem into binary tasks, the model learns effectively even with limited data, making it suitable for resource-constrained scenarios. During inference, the branch with the highest output probability determines the final target class. The model's performance was adjusted by meticulous hyperparameter tuning of batch size, learning rate, and number of epochs, resulting in exceptional accuracy on the MSTAR dataset. Featuring a mere 0.2 million parameters and 0.2 million Multiply-Accumulate Operations (MACCs), it achieves an impressive accuracy of 98.48% under standard conditions. This model performs better than DenseNet-161, a substantially bigger model with 130 times more parameters and nearly 1000 times more MACCs. Furthermore, the model consistently achieved accuracies of 97.83%, 98.15%, and 98.53% across diverse operating conditions, solidifying its potential for SAR applications.

INDEX TERMS Automatic target recognition, computer vision, convolutional neural network, hyperparameter tuning, synthetic aperture radar.

I. INTRODUCTION

Synthetic Aperture Radar (SAR) is a very effective imaging technology that finds use in several domains, including both military and civilian sectors, owing to its ability to operate in all weather conditions and provide images with exceptional quality [1], [2]. The process of evaluating SAR images can present difficulties, mostly attributed to the presence of speckle noise [3]. This challenge necessitates a certain level of skill to effectively address and mitigate its impact. The circumstance has resulted in the emergence of Automatic Target Recognition (ATR) systems that are specially tailored for SAR data. SAR ATR systems, as envisioned by Lincoln Laboratory, typically follow a sequential architecture

The associate editor coordinating the review of this manuscript and approving it for publication was Turgay Celik.

with three distinct stages: detection, discrimination, and classification [4].

The detection stage utilizes a Constant False Alarm Rate (CFAR) detector to eliminate the possible noise from SAR images. The discrimination phase involves the utilization of a binary classifier to effectively eliminate false alarms that may arise due to the presence of objects such as trees, buildings, and automobiles. The classification phase assigns precise classifications to the identified targets. The classification step is particularly intricate, given that the precision of target categorization plays a vital role in determining the overall effectiveness of the SAR ATR system.

The implementation of SAR ATR classification falls into three distinct categories, each defined by its unique methodology: template matching, model-based, and machine learning-based approaches [5]. All methods consist of a training phase conducted offline and a classification step performed online. During the offline phase, template-based technologies are employed to generate and retain templates for diverse targets under varying situations. During the online phase, the template of a specific target is compared to all existing templates to identify the most suitable match. It should be noted that template-based approaches have significant drawbacks in terms of memory storage requirements and computational efficiency [6], [7].

Model-based techniques employ physical or conceptual models, such as Computer-Aided Design (CAD) models or three-dimensional scattering centre models, to depict objectives [8], [9]. During the offline phase, a physical model that applies to all targets is created. Online classification entails feature extraction from the SAR image and subsequent comparison within a defined feature space [10], [11]. It should be noted that model-based techniques are complex and need a significant understanding of SAR target signatures [12].

Machine learning-based SAR ATR systems achieved superior performance compared to traditional template matching and model-based approaches. These systems generate classifiers in the offline phase by exploiting extracted features from training samples. Online testing involves applying trained classifiers to assign class labels to targets based on their specific characteristics. However, traditional machine learning methods often rely on hand-crafted features, limiting their overall performance due to inherent biases and potential for overlooking relevant information.

The emergence of deep learning, namely Convolutional Neural Networks (CNNs), demonstrated notable advancements through the automation of feature extraction [13], [14]. CNNs are successfully utilized in the domain of SAR target detection [15]. This application has yielded significant enhancements in accuracy, with performance metrics increasing from 84.7% to a remarkable level above 99%. These developments are more advanced than other conventional techniques like SVM and AdaBoost [16].

Deeper CNN models tend to exhibit improved classification accuracy [17]. These models are characterized by increased complexity, with a large number of parameters necessitating substantial amounts of annotated data for training. The process of producing labelled SAR data is both time-intensive and costly, hence requiring the implementation of efficient training methods when working with a restricted quantity of annotated data.

The issues associated with computational complexity and weight parameters of CNN classifiers in SAR ATR have been tackled by researchers using structural changes and model compression approaches [18]. To decrease the computational load, CNN's architecture can be modified by structural changes, which may involve the substitution of convolutional layers with FC layers [19]. In certain cases, FC layers are completely removed and substituted by SVM modules [20].

To address concerns with computational complexity and parameter reduction, some researchers have concentrated on CNN model compression strategies. The common model compression approaches used on CNN include knowledge distillation, quantization, pruning, and low-rank optimization. Pruning involves removing the weight connections that are least important in generating a satisfactory outcome, which lowers the total number of weight parameters. As opposed to quantization, which uses fewer bits to express the weight value. Zhong et al. [21] were the first to use pruning and quantization to SAR ATR. While pruning and quantization lower the model's complexity and parameter count, the precision of the model suffers significantly.

To compute the multiplication and addition of weight parameters more effectively, low-rank factorization modifies the matrix representation. CNN kernels are transformed into sparse ones via the low-rank factorization method used by Yu et al. [22]. The deployment of such models on an embedded system is constrained by the demand for a certain hardware configuration.

Knowledge distillation, which leverages a large teacher model to train a smaller student, emerges as a popular technique for model compression in SAR ATR. Notably, Min et al. [23] utilized this approach to construct a lightweight two-layered CNN. Similarly, Zhang et al. [24] achieved a remarkable 65-fold compression of the A-convnet model through knowledge distillation. Yu et al. [25] further refined the process by proposing a multi-layer adaptive network that employs a maximum gradient criterion to effectively extract discriminative target features. However, a persistent limitation of knowledge distillation lies in its dependence on a pre-trained primary network and often necessitates training from scratch.

Researchers have used techniques like data augmentation and transfer learning to lessen the problem caused by a lack of labeled data. Data augmentation refers to the process of transforming a real image using image processing techniques to produce more samples [26]. The technique of transfer learning involves the utilization of pre-trained CNN models, which are then fine-tuned using target data. The utilization of augmented data and transfer learning has demonstrated

TABLE 1. Comparison of strategies for enhancing CNN performance in SAR ATR.

Method	Issues Addressed	Limitations	Reference
Structural modification	Computational Complexity and huge number of weight parameters	• Difficulty in learning long-range dependencies • Reduced representational power	[18]–[20]
Model compression		• Accuracy degradation • Loss of model flexibility • Hardware constraints	[21]–[25]
Data augmentation	Overfitting due to limited labelled data	• Difficulty in choosing optimal augmentation strategy. • Limited effectiveness on certain tasks.	[26]–[31]
Transfer learning		• Dependence on domain expertise. • Increased model complexity.	[36]–[44]
Ensemble of CNN		• Increases total number of Multiply-Accumulate Operations (MACCs).	[45], [46]

efficacy in mitigating overfitting and enhancing the resilience of models [27], [28], [29], [30].

A multiview deep CNN was suggested by Pei et al. and trained on multiview SAR data produced using augmentation methods [29]. The suggested CNN model produced higher performance and only needed a few raw SAR images to train the network. To improve the training data for complex-valued CNN training, Wang et al. adopted sub-aperture decomposition [31]. Ravikumar et al. finds the effect of the structure of the neural network on its performance [32], [33], [34], [35].

Researchers have looked at how transfer learning might help with CNN's overfitting problems [36], [37], [38]. CNN models that have already been trained on a sizable dataset are used for transfer learning, and the target data is used to refine the final layers. To predict saliency, Chaabouni et al. suggested a domain-dependent transfer learning approach using CNN that significantly decreased the overfitting problems [39]. To effectively transfer information from many imperfect sources, Ding et al. presented a bi-directional low-rank transfer learning framework [40].

Furthermore, Dong et al. provide evidence for the successful transfer of semantic information between domains [41]. With enough simulated data and little actual SAR data, Wang et al. [42] trained a CNN model. Like this, Malmgren et al. used transfer learning to improve a CNN model that had been pre-trained using simulated data [36]. In addition, Wang et al. used a decomposition method and optimized a CNN model that had been previously trained on ImageNet data [43]. Transfer learning and data augmentation may both effectively reduce CNN overfitting, however deep CNN models, which have numerous weight parameters and a large computational cost, are the ones that typically use both techniques [38], [44]. Because of this, pre-trained models are inappropriate for embedded applications like SAR ATR.

Despite the progress made in the field, it is important to note that deep CNN models with many parameters are not well-suited for embedded applications such as SAR ATR due to their significant processing requirements [45]. Hence, the exploration of integrating CNNs to enhance recognition accuracy has been undertaken [46]. Despite its effectiveness, this method suffered from significantly increased computational complexity.

Table 1 summarizes existing methods for optimizing CNN performance. Techniques like structural modification and model compression effectively reduce parameter count and model complexity. However, these approaches often come with trade-offs: structural modifications can limit representational power and learning long-range dependencies, while compression techniques often introduce accuracy degradation. Similarly, data augmentation and transfer learning, while effective in alleviating overfitting due to limited data, exhibit domain dependence. While ensemble methods effectively mitigate overfitting, their deployment necessitates increased computational resources due to a rise in MACCs. Existing research has made significant strides in reducing CNN complexity and overfitting, but often through approaches that lack a holistic perspective. This limitation leaves room for further advancements, particularly in leveraging the iterative potential of transfer learning.

This study presents a novel iterative transfer learning (ITL) applied to a branched-CNN architecture, culminating in a lightweight CNN with compelling advantages for SAR ATR. The proposed approach uses sophisticated deep learning techniques, rigorous hyperparameter tuning, and a branched-CNN architecture to automate target detection in SAR data. By leveraging ITL, we demonstrate a significant reduction in the need for complex deep CNN structures, a groundbreaking advancement with ramifications that transcend prior SAR ATR research. The main contributions are:

- Effective SAR target identification via ITL: This study presents a novel method for SAR target identification that combines a branched-CNN architecture with ITL. This strategy shows promising results in SAR target detection applications by drastically lowering computational complexity and memory requirements.
- Hyperparameter optimization: By carefully adjusting hyperparameters to provide the best outcomes, systematically improve the performance of neural network models.
- CNN branch specialization: Every CNN branch has been fine-tuned to identify distinct target classes, which enhances recognition precision.

II. METHODOLOGY

In this work, a branched-CNN is suggested for the recognition of military vehicles in SAR images, which can overcome the computational cost of a conventional CNN. This section begins with a brief description of the conventional CNN followed by the architecture of branched-CNN, which is followed by the training algorithm named as ITL.

A. CONVOLUTIONAL NEURAL NETWORK

CNNs are a form of deep learning neural network that excels at image categorization and object recognition [47]. Conventionally, a CNN consists of convolutional layers (CONV), pooling layers (POOL), and fully-connected layers (FC). In convolutional layer, a kernel K of size $k \times k$ convolves over an input image $I(x, y)$ of size $n \times n$, and generates feature-map F of size $((n-k)/s) \times ((n-k)/s)$, where s is the stride of the convolutional operation. The overall operation is formulated as:

$$M_i^l = f(\sum M_i^{l-1} \odot K_i^l + b_l) \tag{1}$$

where $\odot$ represents the convolution operation, M_i^l is the i^{th} feature map of layer l, b_l is the bias vector of the l^{th} layer, and $f(\cdot)$ is the activation function which helps in the curve fitting of the model for complex data. ReLU [48] is a popular non-linear activation function that is defined as

$$f(M_i^l(x, y)) = max\{0, M_i^l(x, y)\} \tag{2}$$

Pooling is another layer used in CNN that brings translational invariance. This layer, also known as the subsampling layer, maps only a subregion of the previous feature map. Maxpooling is a commonly used pooling technique that computes the highest value in a selected subregion of the feature map. Maxpooling operation can be computed as

$$M_i^{l+1}(x, y) = \max(M_i^l(x + u, y + u)) \tag{3}$$

After successive convolution and pooling layers, the feature map is converted into a one-dimensional vector to feed as input to the FC layer, where the neurons are arranged linearly. Each FC layer adds non-linearity to the existing input data. The final FC layer is a softmax layer responsible for the multiclass classification. The softmax function computes the posterior probability for each class of the target. The softmax function is expressed as

$$P(y_i|M^L) = \frac{exp(M_i^l)}{\sum_{j=1}^{R} exp(M_j^L)} \tag{4}$$

where M^L is the input to the softmax layer, $P(yi)$ denotes the probability for a target to be in i^{th} class and R denotes the total classes.

The training of a CNN requires the optimization of a loss function. Let $T = \{(x^i, y^i), i = 1, 2, \ldots, R\}$ be the SAR ATR training image set, where x^i denotes a SAR training image sample, and y^i is its matching label. The formula for the cross-entropy-loss function is

$$L(\phi) = \frac{-1}{R} \sum_{i=1}^{R} log(y^i|x^i, \phi) \tag{5}$$

The loss function is used to indicate how different the actual and anticipated probability are from each other. CNN training uses the Adam optimization technique to reduce the loss function which is based on the gradient descent principle and optimizes the weight and bias values during CNN training [49].

During the deployment of CNN, two crucial factors that come under consideration are the computational burden and the memory requirement of CNN. The number of MACC gives a direct measure of the computational requirement of CNN. The MACC of the convolution operation in CNN layers are computed as:

$$MACC_s = k \times k \times C_{in} \times F \times F \times C_{out} \tag{6}$$

where k denotes the kernel size, F denotes the feature map size, and the number of input channels is indicated by C_{in}. The output channels are represented by C_{out}. In two FC layers of P and Q neurons, the number of MACC operations is calculated as:

$$MACC_f = P \times Q \tag{7}$$

MACC in the CONV layers is more in number than MACC in the FC layers. That means CONV layers are the primary cause of the computational burden of CNNs.Likewise, the CONV layer's parameter count is determined by

$$W_s = \sum_{l=1}^{N} (F_{l-1}^2 N_l K^2) + N_l \tag{8}$$

The parameters' count in FC layers is calculated as

$$W_f = \sum_{l=1}^{L} N_{l-1} N_l \tag{9}$$

The parameters in CONV layers is fewer than in the FC layers due to the weight-sharing mechanism of CONV

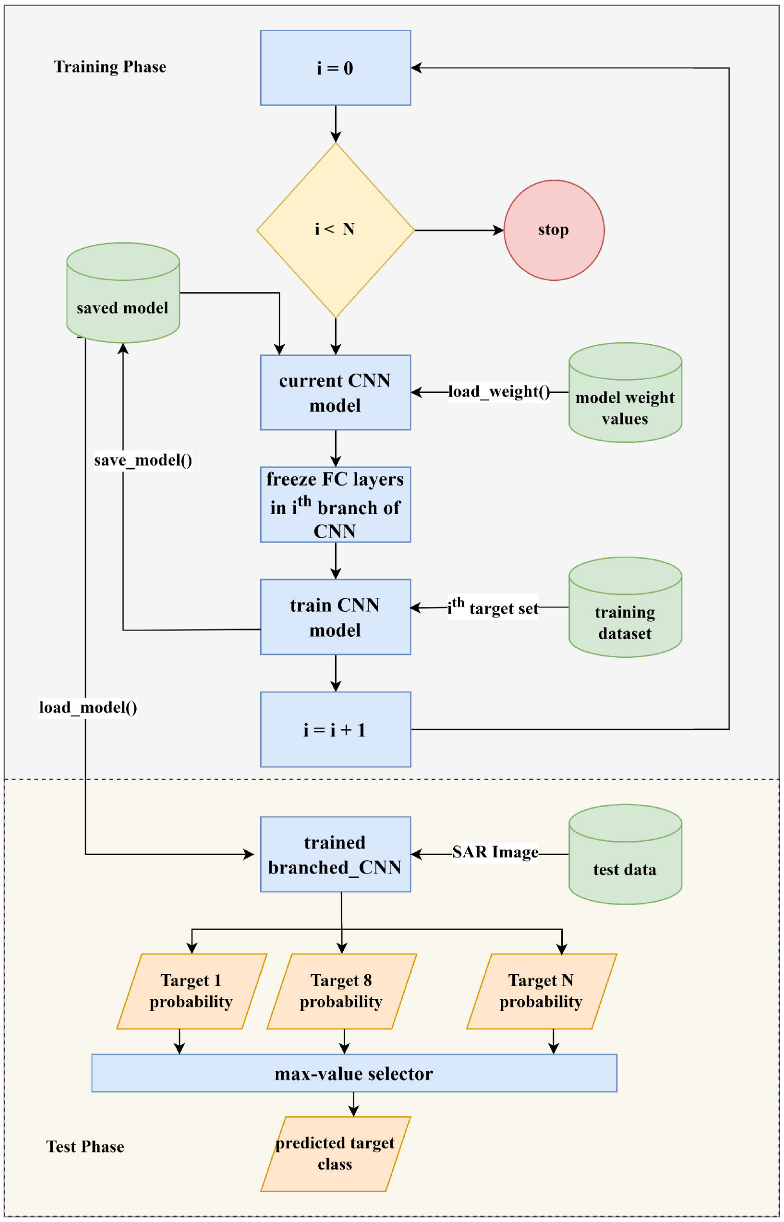

FIGURE 1. **Proposed model.**

layers. The conventional CNN is computationally expensive and has excessive weight parameters that cause overfitting. In the SAR ATR domain, computationally less expensive and low-memory models are required for the application, and ordinary CNN is not appropriate. Furthermore, large CNN model causes overfitting due to the scarcity of labeled training data in the SAR domain.

By incorporating pre-existing weights, transfer learning can alleviate the risk of overfitting in newly trained models. Here, the weight values are transferred to the layers of a new model and it is finetuned on new data before its deployment. Transfer learning is defined in terms of domain and task such that, for an initial domain D_i and a task T_i, a final domain D_f and task T_f, where $D_i \neq D_f$ or $T_i \neq T_f$, the transfer learning improves the learning of the final domain's predictive function $F_f(\cdot)$ in D_f using the knowledge in D_i and T_i. Although transfer learning is computationally efficient and helps achieve better result using a small dataset, it demands the initial domain and final domain to be similar. i.e., $D_i \approx D_f$. However, in the SAR ATR domain, an initial domain similar to the final one is not available for transfer learning. Most current work uses the model trained on the ImageNet dataset for transfer learning. Hence, such models are very deep and consist of huge parameters and are computationally expensive.

B. ARCHITECTURE OF BRANCHED-CNN

In the context of constructing an effective classification model, it is essential to meticulously evaluate three connected

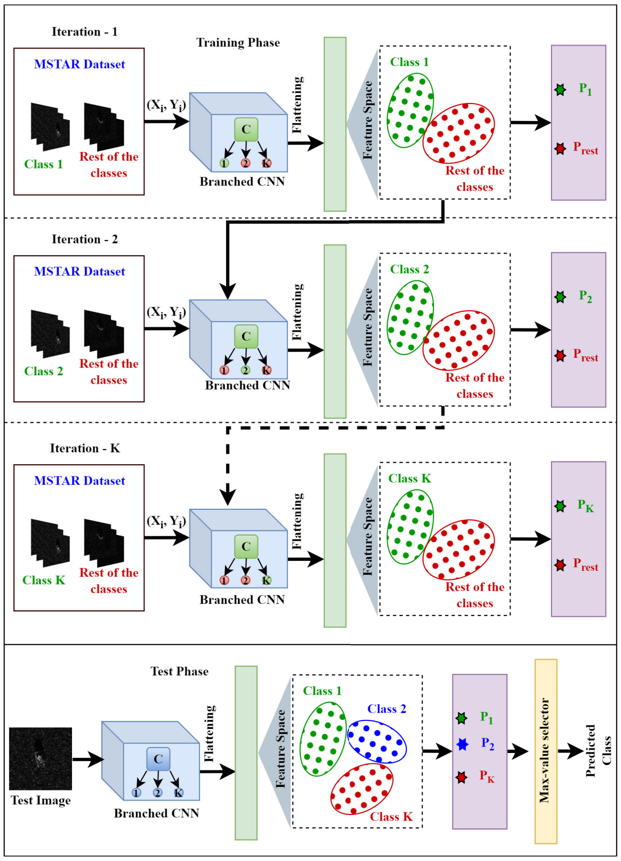

FIGURE 2. **Training of branched-CNN.**

factors: Vapnik-Chervonenkis (VC) dimension [50], model complexity, and the number of classes. The VC dimension measures a model's ability to detect complex correlations in data. Larger values indicate greater flexibility, but also a more significant possibility of overfitting, especially when there is little information available.

A sophisticated CNN can efficiently capture intricate data characteristics with many dimensions. Excessive complexity can result in overfitting, compromising the ability to generalize. On the other hand, a basic CNN could fail to adequately represent the minor differences across classes, resulting in an inadequate fit to the data.

The interaction is further complicated by the number of classes. As classes increase, the complexity also increases, requiring more advanced ability to distinguish between different features. As a result, it is generally necessary to have a greater VC dimension and a more intricate CNN architecture to preserve performance and prevent overfitting. It is crucial to carefully manage this intricacy since an excessive level of complexity might have negative consequences, such as overfitting and increased processing demands. Thus, the most effective classification models are found in the ideal balance between capturing complex connections and retaining the generalization capacity.

It is essential to carefully evaluate the VC dimension, CNN complexity, and the number of classes, which typically requires data-driven experiments and analysis to attain the necessary equilibrium. The proposed method utilizes a One-vs-All (OVA) approach to train the FC layers. The suggested approach is depicted in Fig. 1.

During the first iteration of transfer learning, the model assigns one target as the positive class and categorizes all others as negative. Through this, each branch of the fully connected layers is converted into a specialized binary classifier, optimized for accurately identifying its designated class. This technique cleverly breaks down the intricate multi-class classification issue into smaller, more feasible binary jobs. This modularity promotes the precise performance of each binary classification task, even inside a CNN with fewer parameters. As a result, the model can efficiently acquire knowledge and accurately identify targets, even when training data is scarce. The effectiveness of this unique ITL approach is in the ability to achieve performance with a reduced number of training samples. This approach overcomes the conventional trade-off between precision and limitations in resources, enabling effective and precise identification of targets in situations when data are scarce.

A model possessing a greater VC dimension can discern intricate correlations within the data, although necessitating a larger amount of training data. The VC dimension of a classifier is directly related to the model's complexity Z. The relationship between the number of classes R and the model complexity Z is defined as

$$R = 2^{Z/d} \tag{10}$$

where d is the input data dimension.

The classifier's complexity Z is directly related to the number of classes to predict for a fixed dimension of training samples. So, splitting the multi-class classification task into multiple binary classification task, where each task is solved by one branch of the branched-CNN improves the classification result even with fewer weight parameters.

However, the act of developing distinct classifiers for each target variable results in an augmented computational load. As an illustration, the CNN described in [46] is composed of distinct classifiers that are utilized to forecast the class of individual targets. Consequently, the overall count of MACC operations is elevated as a result of the several convolutional layers present in the ensemble model. The formulation of the total number of MACC operations in the CNN as presented in [46] is expressed as

$$MACC_{total} = \sum_{CNN_{i=i}}^{R} ((k^i)^2 \times C_{in}^i \times (F^i)^2 \times C_{out}^i)(N_{l-1}^i \times N_l^i) \tag{11}$$

where R is the number of target classes. Branched-CNN is an enhanced version of CNN in [46], where the redundant convolutional layers have been replaced with two convolutional layer common for all the targets. By restructuring the architecture, the total MACC operations in branched-CNN are calculated as

$$MACC_{total} = (k^2 \times C_{in} \times F^2 \times C_{out}) \sum_{CNN_{i=1}}^{R} (N_{l-1}^{i} \times N_{l}^{i}) \tag{12}$$

In branched-CNN, the purpose of each CNN branch is to identify the distinctive qualities of a certain target. Figure 2's depiction of the branched-CNN training process demonstrates how this model maximizes efficiency by employing shared initial convolutional layers on all targets. Similar to a traditional CNN, the branched-CNN has typical layers such as CONV, batch normalization layers (BN), ReLU, POOL, and FC.

A key deviation from conventional CNNs lies in the segmented architecture of branched-CNNs. They bifurcate into separate branches, each housing a compact group of FC layers. These branches specialize in individual target identification, functioning as independent experts. The branch tasked with “T72” classification, for instance, only focuses on extracting the essential features of this specific target. During the testing phase, each branch of the branched-CNN provides a probability estimate for each target. The branch that generates the highest predicted value determines the class of the target. Fig. 3 shows the architecture of the branched-CNN.

The CNN described in [46] is similar to the early layers of the branched-CNN. The branched-CNN receives a 24x24 grayscale image and does a 5x5 convolution operation on it. All layers make use of the ReLU activation function, except for the final FC layer. The outcome of the first convolution phase is a 16x20x20-pixel feature map. Batch normalization, a 2x2 pooling procedure, and feature mapping reduce feature map's size to 16x10x10. The second CONV layer, which has 32 convolution operations, further decreases feature map's size to 32x6x6. In the wake of further 2x2 max-pooling procedures, the feature map shrinks to 32x3x3. Flattening of feature map into a 1x288 vector makes it a common input for many branches. Each branch consists of three FC layers, each with 64, 32, and 2 neurons.

C. TRAINING ALGORITHM FOR BRANCHED-CNN

The branched-CNN training algorithm depicted in Algorithm 1 is an ITL mechanism that takes the dataset $\{D\}$ and the set of target class $\{V\}$ as inputs. It returns the trained CNN model $\{U\}$ and its weight parameters $\{W\}$ as output. The CNN model is initialized with the branched-CNN architecture using the function $branched_{CNN}()$, and the values of the weight matrix $\{W\}$ are randomly initialized from a Gaussian distribution using the function $rand(G)$. The training of branched-CNN is an iterative transfer learned mechanism. That is, instead of using the conventional transfer learning method where the initial domain D_i and final domain D_f are not similar, in branched-CNN, $D_i \approx D_f$.

Algorithm 1 Pseudocode for Training of Branched-CNN

Input: Training dataset D, Target class {V}
Output: Trained model U, trained weights W

```
1:  procedure branched-CNN($D$, $V$)
2:      $U \leftarrow branched_{CNN}$
3:      $W \leftarrow rand(G)$
4:      for each class $i$ in $\{V\}$ do
5:          $B \leftarrow load(U, W)$
6:          for each branch $b$ in $B$ do
7:              if $b \neq i$ then
8:                  $\hat{B} \leftarrow freeze(B, b)$
9:          $U, W \leftarrow train(\hat{B}, D)$
10:     return $U, W$
```

During each iteration of the algorithm, the task of classification changes. Hence, similar to conventional transfer learning, $T_i \neq T_f$, where T_i and T_j represent the initial task and final task, respectively. In branched-CNN, each branch is trained for a specific target class. That is, for every target class in $\{V\}$, the CNN model B is loaded with the previously trained model U and its trained weights W. Then, every branch except the branch corresponding to the current target class is set into freeze mode using the function *freeze*(). Hence, the weights in the layers that in freeze mode are not updated during that particular training iteration. Once the model $\hat{B}$ trained on the subset of the dataset D, the updated model, B and the weights, W are saved. After training the model on each target class, the final model U and the corresponding weight values are returned. During prediction, a given target's class is predicted by each branch of the branched-CNN. The branch that predicts the highest probability determines the target's actual class.

III. RESULT ANALYSIS

This section introduces the MSTAR dataset and then presents the outcomes of tests carried out in various operating environments. On a 2.6 GHz CPU, 16 GB of RAM, and a 4 GB GPU card, the suggested branched-CNN is trained. The Keras API and Microsoft Windows 10 Pro-64-bit were used to build each approach.

A. DATASET

Using the MSTAR public dataset, the efficacy of SAR ATR classification algorithms is assessed and contrasted [51]. This dataset has SAR images of 10 military vehicles with a resolution of 0.3 meters. Fig. 4 shows images of these objects taken using SAR and optical technology.

Due to speckle noise and shadows created by the imaging angle, two objects in Fig. 4 that exhibit considerable changes in the optical view are difficult to identify from one another. The MSTAR collection's SAR images were captured under two distinct operating circumstances. Training images have a 17-degree depression under Standard Operating Conditions

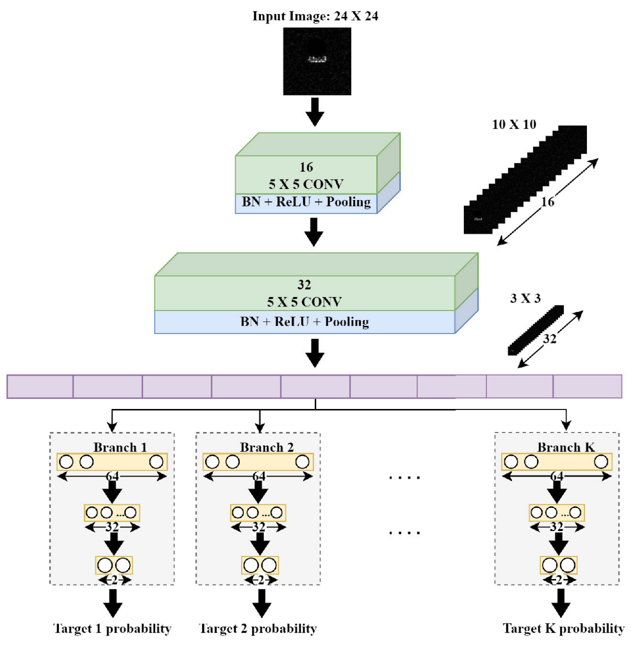

FIGURE 3. Architecture of branched-CNN.

(SOC), whereas test images have a 15-degree depression. The experimental setup employed 2747 images for training and 2425 images for validation. The target's background, depression angle, and intra-class variability alter during Extended Operating Conditions (EOC).

B. RESULTS IN HYPERPARAMETER TUNING

Hyperparameters are predetermined settings that regulate many elements of the training process and model architecture, in contrast to model parameters, which are learned from the training data. The selection of hyperparameters may have a substantial influence on a model's performance, including how quickly it converges, how well it generalizes, and how resistant it is to overfitting or underfitting. To find the configuration that gives the optimum model performance for a certain job, hyperparameter tuning entails methodically examining various hyperparameter values or combinations. A machine learning or deep learning model must be successfully hyperparameter-tuned to produce cutting-edge findings and make sure that it is appropriate for the task at hand. The hyperparameters used in this model are batch size, learning rate and epoch. Figure 5-12 guided the selection of hyperparameters based on its training (blue) and validation (red) loss curves.

1) BATCH SIZE

A model's training and its performance are significantly impacted by the choice of batch size. To investigate the influence of batch size on model performance, branched-CNN is trained using various batch sizes (16, 32, 64, and 128) and monitored the loss on both the training and validation sets as shown in Fig. 5, Fig. 6, Fig. 7, and Fig. 8. Smaller batch sizes (16 and 32) exhibited more frequent updates and noise, leading to less smooth but potentially faster convergence. The model achieved the lowest validation loss with a batch size of 32, indicating superior generalization to unseen data. The training loss for batch sizes 64 and 128 continued to decrease, while validation loss plateaued or slightly increased, suggesting potential overfitting. This underscores the importance of monitoring validation loss to prevent overfitting. Although the batch size of 32 caused divergence of the loss curve for one target (target 6), but it made it possible for the model to train well for most of the other targets. The model was able to update its weights more often with a batch size of 32 than with higher batch sizes, perhaps promoting convergence. From the loss curve analysis of batch sizes 16, 32, 64, and 128, a batch size of 32 was optimal for branched-CNN.

2) EPOCH

Examining the evolution of the loss function across training epochs provides valuable insights into the optimization process and model performance. Figure 9 depicts the loss curve for our model trained for 200 epochs. Examining key epochs – 50, 100, and 200, reveals intriguing dynamics in the learning process. The curve exhibits a steep initial descent in the first 50 epochs, signifying rapid progress in minimizing the loss. This suggests the model rapidly grasped the underlying patterns in the data, leading to significant error reduction. However, the curve does not flatten completely, indicating potential for further refinement.Between epochs 50 and 100, the curve exhibits a less dramatic, yet sustained, decrease in loss. This phase likely reflects the model fine-tuning its internal parameters and adjusting its understanding of the data's complexities. Interestingly, the curve reaches its minimal point at epoch 100, highlighting this epoch as the one with the optimal model performance. Beyond epoch 100, the loss curve plateaus with minimal fluctuations. This indicates that the model has converged to a local minimum of the loss function. While further training epochs might result in minute loss reductions, the risk of overfitting increases significantly, potentially negating the gains in generalizability.

3) LEARNING RATE

The comparison of loss curves across different learning rates underscores the crucial role of this hyperparameter in shaping the optimization process. To investigate the role of learning rate in the optimization process, branched-CNN is trained with three different learning rates: 0.1, 0.01, and 0.001. Analyzing the respective loss curves illustrated in Fig. 10, Fig. 11, and Fig. 12 reveals fascinating insights into the interplay between learning rate and model convergence. The validation loss curves for a learning rate of 0.1 and 0.01 exhibits a rapid initial descent, suggesting aggressive exploration of the loss landscape. Furthermore, these curves show a sudden convergence even before 10 epochs, raising concerns about potential overfitting. The curve for the learning rate of 0.001 strikes a remarkable balance between speed and stability. It starts with a moderate descent rate,

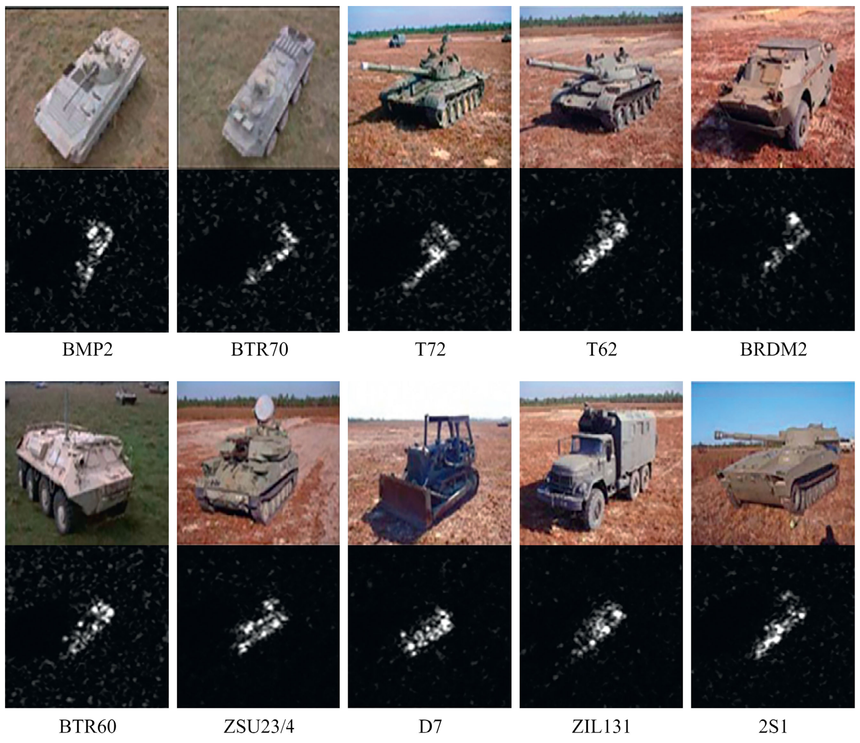

FIGURE 4. Optical and SAR images of military vehicles.

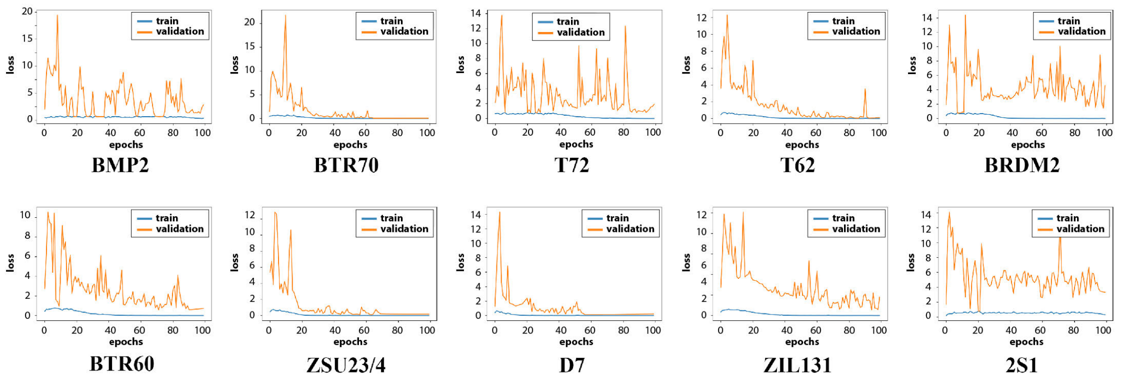

FIGURE 5. Loss plot for batch size 16.

ensuring stable exploration, and gradually converges to a significantly lower loss plateau compared to the other two rates. This minimal loss value at epoch 100 highlights the effectiveness of this learning rate in driving the model towards the optimal solution.

From the experiments, the following hyperparameters are selected: learning rate = 0.001, batch size = 32, and epoch = 100. The model is trained to optimize the categorial cross-entropy loss function with an Adam optimizer.

C. RESULTS UNDER SOC

The effectiveness of ITL on branched-CNN is assessed by SOC on a ten-class classification test. Figure 13 illustrates the training and test images that differs in their azimuth and depression angles. Training images have a 17-degree depression angle, while test images captured at a 15-degree.

Figure 14 depicts the confusion matrix for categorizing the 10 military vehicles under SOC. The target's real class is represented by the rows in the confusion matrix and the classifier's prediction appears in the columns. As seen in Fig. 14, the overall classification accuracy of branched-CNN under SOC is 98.48%, indicating the effectiveness of the recommended strategy in these circumstances.

The performance of ITL on the branched-CNN is then assessed using its error rate, number of parameters, MACC, and training data. The aggregate percentage of targets that were improperly identified is known as the error rate, which highlights the classifier's flaws. MACC is a statistic to gauge the model's level of computational complexity. The

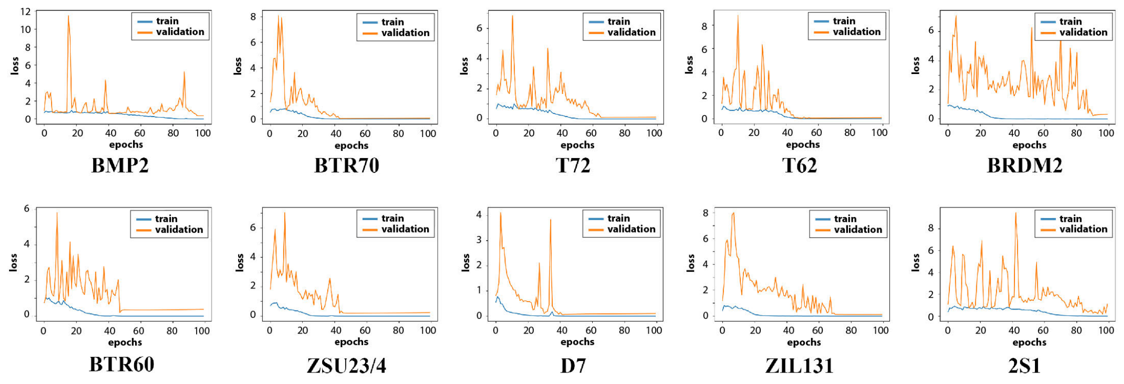

FIGURE 6. Loss plot for batch size 32.

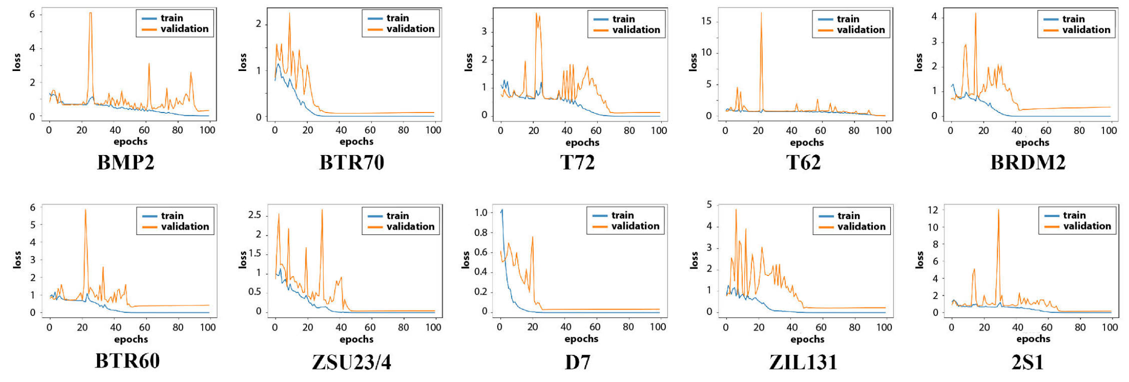

FIGURE 7. Loss plot for batch size 64.

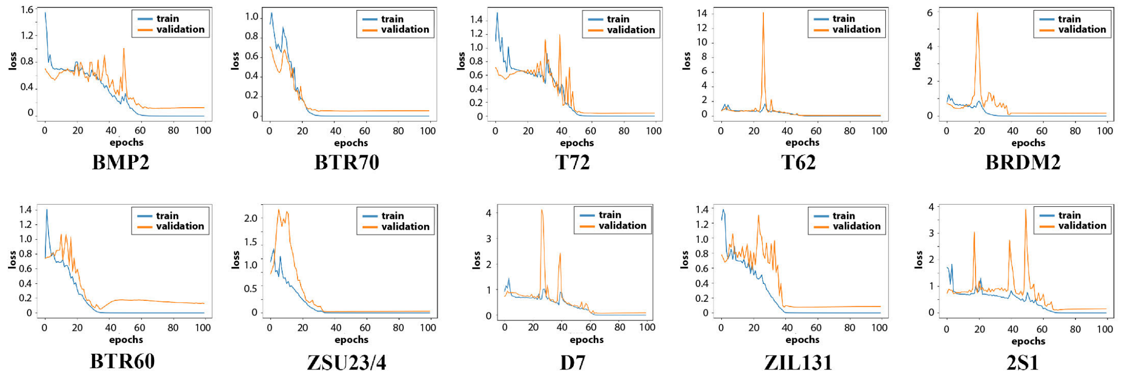

FIGURE 8. Loss plot for batch size 128.

amount of weight parameters reveals the CNN model's size. A lightweight-CNN model is more effective for deployment since it can be trained more rapidly and stored directly on the chip. It is imperative to train a CNN model with fewer images due to the limited number of annotated SAR images available. Table 2 contrasts the outcomes of the branched-CNN with the standard SAR ATR models. Among the compared CNNs, CNN in [30] received its result by our tests, however, the results of the other CNNs are taken straight from their publications.

Table 2 shows that compared to the major baseline models, the suggested branched-CNN has a reduced error rate. All

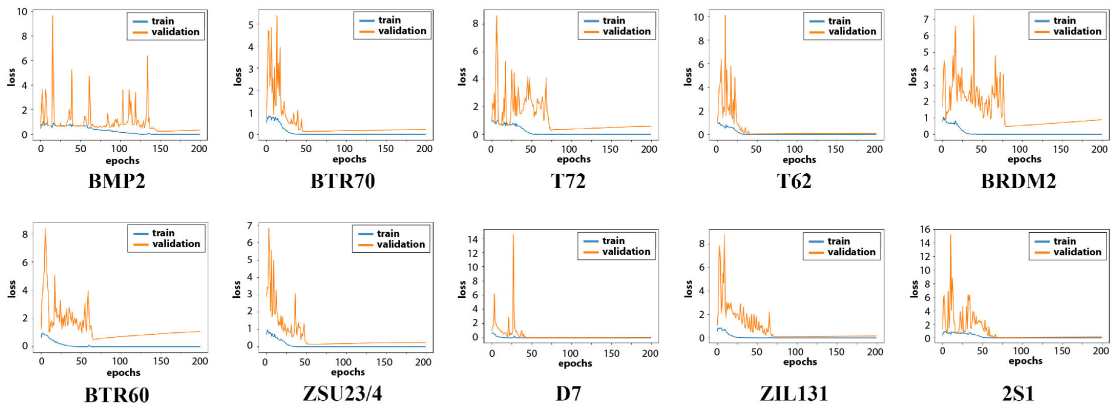

FIGURE 9. Loss plot for 200 epochs.

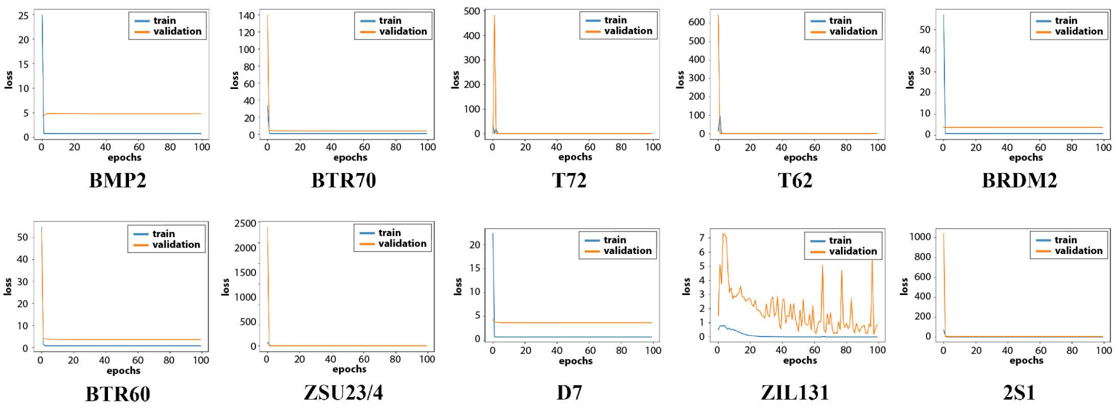

FIGURE 10. Loss plot for learning rate=0.1.

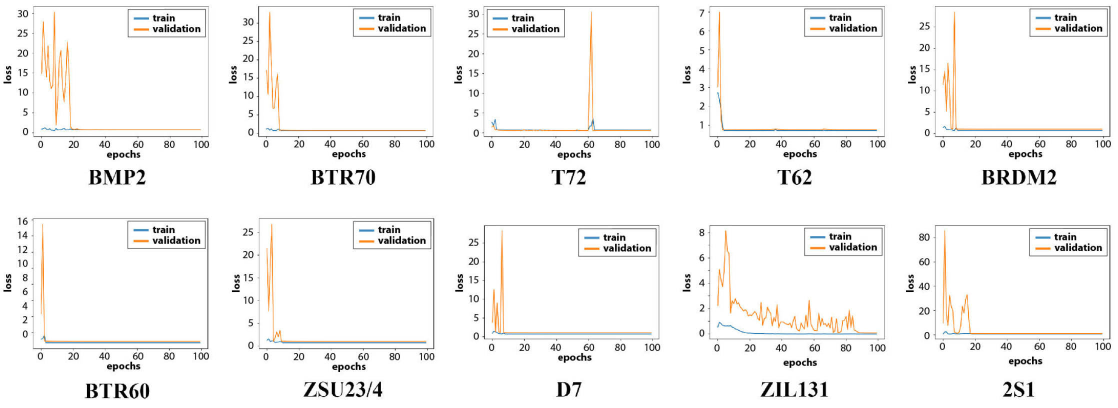

FIGURE 11. Loss plot for learning rate=0.01.

the networks created for SAR target identification achieve an error rate of around 2%. The suggested model is also the second lightweight CNN model in terms of parameter count. Branched-CNN is the most affordable model when comparing the MACC operations of other models. The proposed model employs ten times fewer MACC operations than CNN in [30] and [46], although it uses the same training data as branched-CNN. According to empirical evidence, the ITL on the branched-CNN simplify the model's computations without significantly affecting classification accuracy.

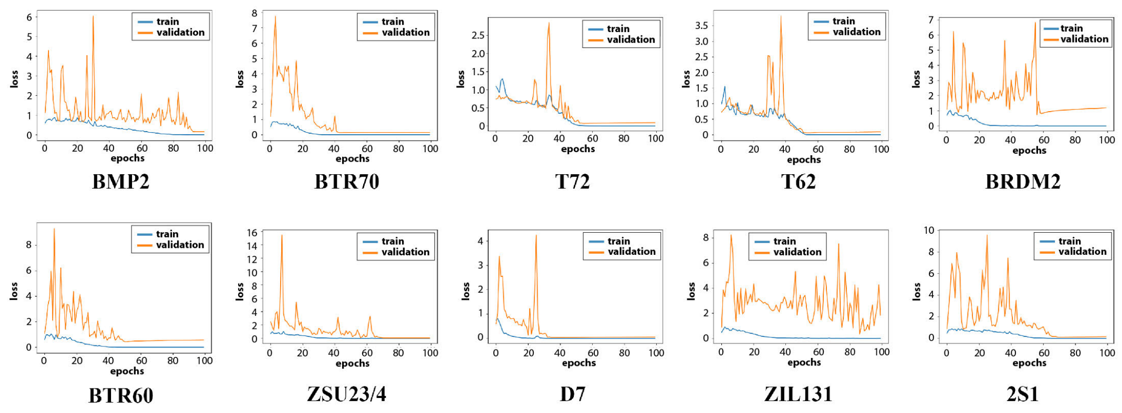

FIGURE 12. Loss plot for learning rate=0.001.

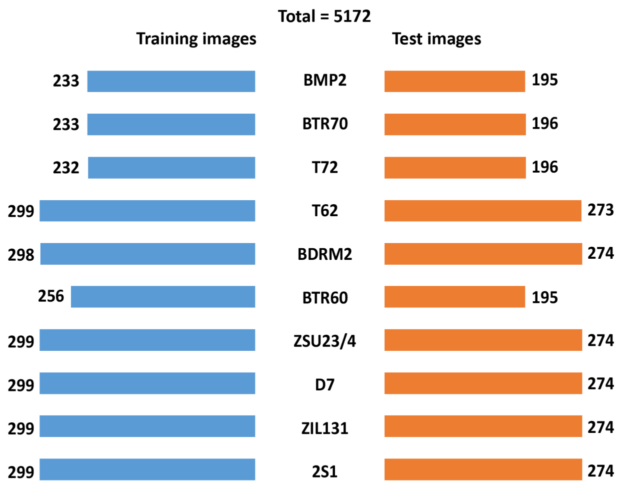

FIGURE 13. Training set and test set under SOC.

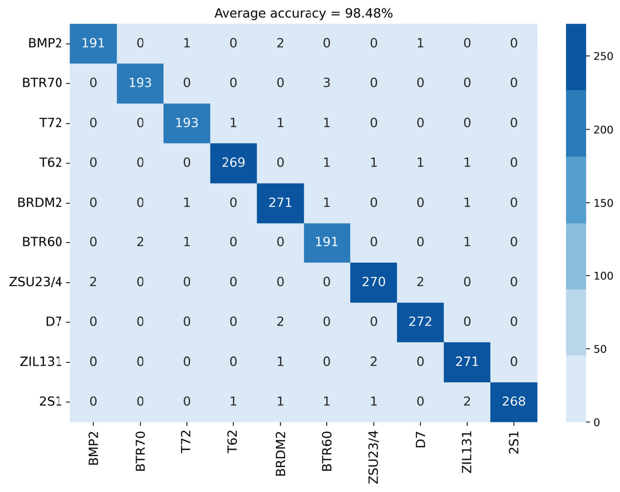

FIGURE 14. SOC-Confusion matrix of ten-class classification.

TABLE 2. Performance comparison of branched-CNN with baseline models.

Error rate	Parameter count	MACC	No. of training samples	Reference
1.61	2.2×10^5	4.34×10^6	4.62×10^6	[21]
1.8	6.04×10^4	4.05×10^6	4.66×10^4	[23]
1.94	1.06×10^6	1.69×10^6	2.74×10^3	[30]
1.19	0.34×10^6	9.61×10^6	2.74×10^3	[46]
1.52	2.19×10^5	2.67×10^5	2.74×10^3	proposed method

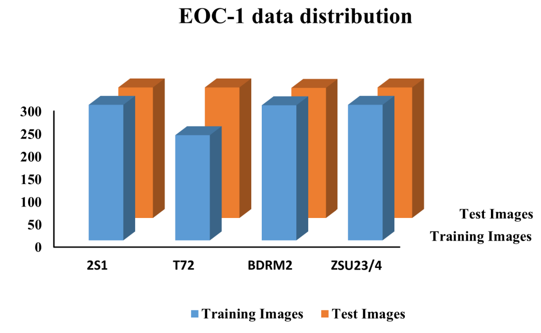

FIGURE 15. EOC-1 data distribution.

The proposed model is compared with the existing major transfer learning models and is shown in Table 3. Table 3 compares the performance of various state-of-the-art models in SAR ATR. While the proposed model achieves an accuracy of 98.48%, which is slightly lower than some other models like DenseNet-161, it offers significant advantages in terms of model complexity. With only 2.19×10^5 parameters, the proposed model is over 130 times smaller than DenseNet-161 and over 200 times smaller than ResNet-101, making it a more lightweight and efficient option for resource-constrained environments. Additionally, the proposed model requires only 2.74×10^3 training samples, demonstrating its ability to learn effectively from limited data. These comparisons highlight the trade-off between accuracy and model complexity, suggesting that the proposed model may be a suitable choice for applications where computational efficiency and resource limitations are major concerns. The proposed model offers a good balance

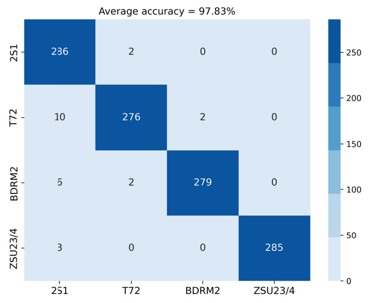

FIGURE 16. Confusion matrix for EOC-1.

TABLE 3. Comparison of proposed model with state-of-the-art transfer learning methods for SAR target recognition models.

Model	Accuracy	No. of parameters	MACC	No. of training samples
Alexnet [52]	99.15%	6.1×10^7	5.53×10^{10}	1.37×10^4
VGGNet16 [53]	99.03%	1.38×10^8	1.53×10^{10}	1.37×10^4
Inception-V3 [54]	99.37%	2.38×10^7	6.04×10^9	1.37×10^4
ResNet-101 [55]	99.79%	4.45×10^7	1.78×10^{13}	1.37×10^4
DenseNet-161 [56]	99.92%	2.86×10^7	3.01×10^{11}	1.37×10^4
Proposed Model	98.48%	2.19×10^5	2.67×10^5	2.74×10^3

between accuracy and model complexity. While it does not achieve the highest accuracy on this dataset, it is significantly smaller and more efficient than the other models under comparison. This makes it a good choice for SAR ATR system.

D. RESULTS UNDER EOC

Changes in depression angles can affect SAR images. For certain applications, it is crucial to assess a SAR ATR system's resilience to variations in depression angle. For studies based on depression angle variation (EOC-1), Fig. 15 shows the data distribution of the training and test images.

The EOC-1 experiment focuses on analyzing the robustness of a target recognition system under differing visual perspectives. This involves four distinct targets, presented in Fig. 15. Notably, the training images captured these targets at a fixed depression angle of 17 degrees, while the test samples depict them at a different angle of 30 degrees. This shift in perspective adds a layer of complexity, simulating real-world scenarios where viewing angles can vary significantly.

The performance of the system under this simulated variation is further analyzed through the EOC-1 confusion matrix, showcased in Fig. 16. This matrix provides a detailed breakdown of the system's classification accuracy for each target, revealing potential misidentifications at the 30-degree angle.

The confusion matrix illustrated in Fig. 16 evaluates the classification model's performance in EOC-1 scenario. The model demonstrates exceptional performance overall, with high levels of accuracy across all classes. 2S1 and ZSU23/4 classes exhibit near-perfect classification, with

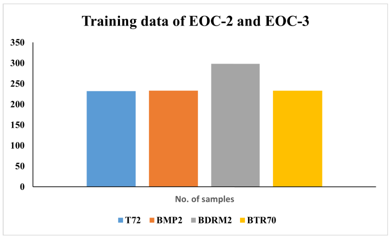

FIGURE 17. Training data distribution of EOC-2 and EOC-3.

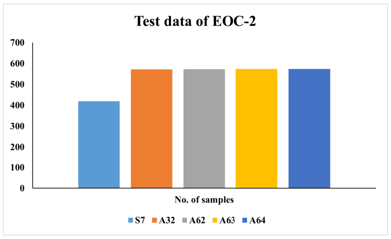

FIGURE 18. Test data distribution of EOC-2.

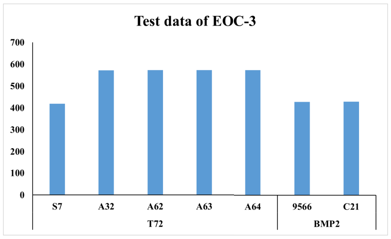

FIGURE 19. Test data distribution of EOC-3.

only a few misclassifications. T72 and BRDM2 classes also achieve strong results, with only minor confusion between them. Misclassifications primarily occur between T72 and BRDM2, suggesting potential similarities in their features that the model might need further refinement to distinguish. It is evident that the model is resistant to changes in depression angle. The robustness of the suggested model is examined for various target versions and setups. In this experimental context, the training set comprises four targets at a 17-degree depression angle, as shown in Fig. 17. The target T72 has five configuration options in the EOC-2 test set as shown in Fig. 18. Like this, the version-variants of EOC test set shown in Fig. 19 has five T72 variants and two BMP2 variants.

The results for version-variant and configuration-variant are depicted in Fig. 20 and Fig. 21, respectively. From Fig. 20, it can be observed that the model achieved a commendable overall accuracy of 98.15% on the EOC-2 test data, correctly classifying the majority of instances across all target classes. The model demonstrated particularly strong performance in classifying BMP2 variants and also performed remarkably well in classifying T72 variants, with minimal confusion between T72 and other target classes. Similarly, the model effectively distinguished between BMP2 and T72 variants, with very few instances of one being misclassified as the other. Furthermore, misclassifications involving BDRM2 and BTR70 were relatively infrequent, suggesting the model's ability to differentiate these targets.

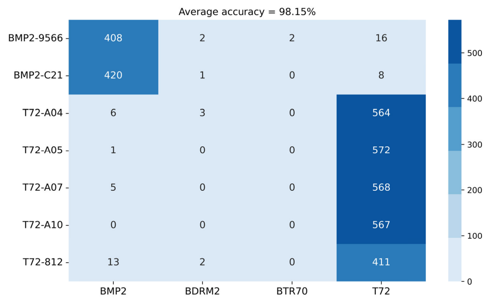

FIGURE 20. **Confusion matrix for EOC-2.**

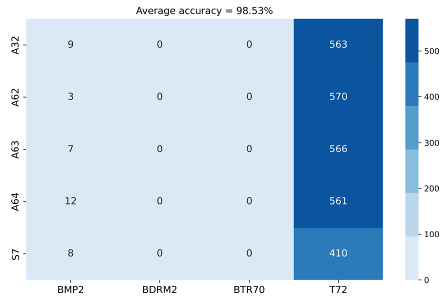

FIGURE 21. **Confusion matrix for EOC-3.**

The confusion matrix illustrated in Fig. 21 meticulously evaluate the model's performance in classifying five different configuration variants of the T72 military vehicle, aiming to distinguish them from BMP2, BDRM2, and BTR70 targets. The model exhibited outstanding performance in classifying T72 variants, with exceptionally high accuracy of 98.53% on the EOC-3 test data. The matrix showcases a remarkably low number of misclassifications involving other target classes (BMP2, BDRM2, BTR70). This signifies the model's robust ability to differentiate T72 variants from non-T72 targets and suggests that the model effectively captures the distinct features of each variant. The results show that the suggested network recognize the targets with varying versions and configurations.

While the proposed lightweight architecture's efficiency is commendable, its representational power may be constrained compared to more complex CNNs. Future research should delve into effective methods for balancing model complexity and performance. Exploring techniques like model compression techniques or architectural search methods could potentially yield even more efficient architectures without substantial accuracy degradation. Moreover, the current study primarily focuses on controlled SAR image target recognition. Rigorous evaluation under real-world conditions featuring diverse noise levels, clutter, and environmental variations is crucial for practical applications. This necessitates further investigation into domain adaptation techniques or adversarial training to enhance the model's robustness and generalizability.

IV. CONCLUSION

The need for extensive data and computational resources poses a persistent challenge for traditional SAR target recognition techniques, even as CNNs have emerged as the preferred approach. This study proposes a novel method, ITL with branched-CNN, that alleviates these limitations. Instead of tackling multi-class classification directly, ITL decomposes it into smaller, binary subtasks. Each branch of the CNN specializes in identifying a specific target class, akin to a team of experts. This simplifies the learning process and reduces the model's complexity. Furthermore, training these branches sequentially using a OVA approach allows for efficient optimization even with limited labeled data. During inference, the branch with the highest confidence score determines the final target class. Hyperparameter tuning of batch size, epochs, and learning rate further fine-tuned the model's performance. By synergizing hyperparameter optimization and the power of ITL, branched-CNN achieves an exceptional balance between performance and efficiency. Its impressive SOC accuracy of 98.48% and strong performance under diverse conditions (EOC-1: 97.83%, EOC-2: 98.15%, EOC-3: 98.53%) are attained while maintaining a remarkably lightweight architecture of 0.2 million parameters and 0.2 million MACCs. This accomplishment paves the way for efficient and robust SAR target recognition applications.

While this study demonstrates the effectiveness of ITL within the chosen CNN architecture, its complete potential remains unexplored. Future work will involve applying ITL to a wider range of CNN architectures to assess its generalizability and potential for further accuracy enhancement. This exploration may reveal architecture-specific adaptations or limitations deserving further investigation.

REFERENCES

[1] Y. Li, L. Du, and D. Wei, "Multiscale CNN based on component analysis for SAR ATR," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5211212, doi: 10.1109/TGRS.2021.3100137.

[2] G. Dong and H. Liu, "Global receptive-based neural network for target recognition in SAR images," *IEEE Trans. Cybern.*, vol. 51, no. 4, pp. 1954–1967, Apr. 2021, doi: 10.1109/TCYB.2019.2952400.

[3] P. Singh, A. Shankar, and M. Diwakar, "Review on nontraditional perspectives of synthetic aperture radar image despeckling," *J. Electron. Imag.*, vol. 32, no. 2, Nov. 2022, Art. no. 021609, doi: 10.1117/1.jei.32.2.021609.

[4] K. El-Darymli, P. McGuire, D. Power, and C. Moloney, "Target detection in synthetic aperture radar imagery: A state-of-the-art survey," *J. Appl. Remote Sens.*, vol. 7, no. 1, Mar. 2013, Art. no. 071598, doi: 10.1117/1.jrs.7.071598.

[5] K. El-Darymli, E. W. Gill, P. Mcguire, D. Power, and C. Moloney, "Automatic target recognition in synthetic aperture radar imagery: A state-of-the-art review," *IEEE Access*, vol. 4, pp. 6014–6058, 2016, doi: 10.1109/ACCESS.2016.2611492.

[6] L. M. Novak, S. D. Halversen, G. Owirka, and M. Hiett, "Effects of polarization and resolution on SAR ATR," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 33, no. 1, pp. 102–116, Jan. 1997, doi: 10.1109/7.570713.

[7] L. M. Novak, G. J. Owirka, and W. S. Brower, "Performance of 10- and 20-target MSE classifiers," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 36, no. 4, pp. 1279–1289, Oct. 2000, doi: 10.1109/7.892675.

[8] T. D. Ross, J. J. Bradley, L. J. Hudson, and M. P. O'Connor, "SAR ATR: So what's the problem? An MSTAR perspective," *Proc. SPIE*, vol. 3721, pp. 662–672, Apr. 1999, doi: 10.1117/12.357681.

[9] K. Ikeuchi, M. D. Wheeler, T. Yamazaki, and T. Shakunaga, "Model-based SAR ATR system," *Proc. SPIE*, vol. 2757, pp. 376–387, Jun. 1996, doi: 10.1117/12.242040.

[10] R. Hummel, "Model-based ATR using synthetic aperture radar," in *Proc. Rec. IEEE Int. Radar Conf*, May 2000, pp. 856–861, doi: 10.1109/RADAR.2000.851947.

[11] J. R. Diemunsch and J. Wissinger, "Moving and stationary target acquisition and recogntion (MSTAR) model-based automatic target recognition: Search technology for a robust ATR," *Proc. SPIE*, vol. 3370, pp. 481–492, Sep. 1998, doi: 10.1117/12.321851.

[12] U. Srinivas, V. Monga, and R. G. Raj, "SAR automatic target recognition using discriminative graphical models," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 50, no. 1, pp. 591–606, Jan. 2014, doi: 10.1109/TAES.2013.120340.

[13] Z. Han, D. Hong, L. Gao, J. Yao, B. Zhang, and J. Chanussot, "Multimodal hyperspectral unmixing: Insights from attention networks," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5524913, doi: 10.1109/TGRS.2022.3155794.

[14] N. Hatami, Y. Gavet, and J. Debayle, "Classification of time-series images using deep convolutional neural networks," 2017, *arXiv:1710.00886*.

[15] F. Gao, T. Huang, J. Sun, J. Wang, A. Hussain, and E. Yang, "A new algorithm for SAR image target recognition based on an improved deep convolutional neural network," *Cogn. Comput.*, vol. 11, no. 6, pp. 809–824, Dec. 2019, doi: 10.1007/s12559-018-9563-z.

[16] S. Chen, H. Wang, F. Xu, and Y.-Q. Jin, "Target classification using the deep convolutional networks for SAR images," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 8, pp. 4806–4817, Aug. 2016, doi: 10.1109/TGRS.2016.2551720.

[17] Y. Guo, Z. Pan, M. Wang, J. Wang, and W. Yang, "Learning capsules for SAR target recognition," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, pp. 4663–4673, 2020, doi: 10.1109/JSTARS.2020.3015909.

[18] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "A survey of model compression and acceleration for deep neural networks," 2017, *arXiv:1710.09282*.

[19] X. Ding, C. Xia, X. Zhang, X. Chu, J. Han, and G. Ding, "RepMLP: Re-parameterizing convolutions into fully-connected layers for image recognition," 2021, *arXiv:2105.01883*.

[20] S. A. Wagner, "SAR ATR by a combination of convolutional neural network and support vector machines," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 52, no. 6, pp. 2861–2872, Dec. 2016, doi: 10.1109/TAES.2016.160061.

[21] C. Zhong, X. Mu, X. He, J. Wang, and M. Zhu, "SAR target image classification based on transfer learning and model compression," *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 3, pp. 412–416, Mar. 2019, doi: 10.1109/LGRS.2018.2876378.

[22] M. Yu, G. Dong, H. Fan, and G. Kuang, "SAR target recognition via local sparse representation of multi-manifold regularized low-rank approximation," *Remote Sens.*, vol. 10, no. 2, p. 211, Feb. 2018, doi: 10.3390/rs10020211.

[23] R. Min, H. Lan, Z. Cao, and Z. Cui, "A gradually distilled CNN for SAR target recognition," *IEEE Access*, vol. 7, pp. 42190–42200, 2019, doi: 10.1109/ACCESS.2019.2906564.

[24] F. Zhang, Y. Liu, Y. Zhou, Q. Yin, and H.-C. Li, "A lossless lightweight CNN design for SAR target recognition," *Remote Sens. Lett.*, vol. 11, no. 5, pp. 485–494, May 2020, doi: 10.1080/2150704x.2020.1730472.

[25] X. Yu, F. Dong, H. Ren, C. Zhang, L. Zou, and Y. Zhou, "Multilevel adaptive knowledge distillation network for incremental SAR target recognition," *IEEE Geosci. Remote Sens. Lett.*, vol. 20, pp. 1–5, 2023, doi: 10.1109/LGRS.2023.3263659.

[26] E. D. Cubuk, B. Zoph, D. Mané, V. Vasudevan, and Q. V. Le, "AutoAugment: Learning augmentation strategies from data," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 113–123, doi: 10.1109/CVPR.2019.00020.

[27] M. Liu, L. Peng, X. Liu, L. Dong, M. Hui, and Y. Zhao, "SAR image classification based on CNN in real and simulation datasets," in *Proc. 9th Int. Conf. Graphic Image Process. (ICGIP)*, Apr. 2018, pp. 820–827, doi: 10.1117/12.2303468.

[28] J. Ding, B. Chen, H. Liu, and M. Huang, "Convolutional neural network with data augmentation for SAR target recognition," *IEEE Geosci. Remote Sens. Lett.*, vol. 13, no. 3, pp. 364–368, Mar. 2016, doi: 10.1109/LGRS.2015.2513754.

[29] J. Pei, Y. Huang, W. Huo, Y. Zhang, J. Yang, and T.-S. Yeo, "SAR automatic target recognition based on multiview deep learning framework," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 4, pp. 2196–2210, Apr. 2018, doi: 10.1109/TGRS.2017.2776357.

[30] Y. Yan, "Convolutional neural networks based on augmented training samples for synthetic aperture radar target recognition," *J. Electron. Imag.*, vol. 27, no. 2, p. 1, Apr. 2018, Art. no. 023024, doi: 10.1117/1.jei.27.2.023024.

[31] R. Wang, Z. Wang, K. Xia, H. Zou, and J. Li, "Target recognition in single-channel SAR images based on the complex-valued convolutional neural network with data augmentation," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 59, no. 2, pp. 796–804, Apr. 2023, doi: 10.1109/TAES.2022.3190804.

[32] A. Ravikumar and H. Sriraman, "Acceleration of image processing and computer vision algorithms," in *Handbook of Research on Computer Vision and Image Processing in the Deep Learning Era*. Pennsylvania, PA, USA: IGI-Global, Nov. 2022. [Online]. Available: https://www.igi-global.com/chapter/acceleration-of-image-processing-and-computer-vision-algorithms/www.igi-global.com/chapter/acceleration-of-image-processing-and-computer-vision-algorithms/313986.

[33] A. Ravikumar and H. Sriraman, "Computationally efficient neural rendering for generator adversarial networks using a multi-GPU cluster in a cloud environment," *IEEE Access*, vol. 11, pp. 45559–45571, 2023, doi: 10.1109/ACCESS.2023.3274201.

[34] A. Ravikumar, H. Sriraman, P. M. Sai Saketh, S. Lokesh, and A. Karanam, "Effect of neural network structure in accelerating performance and accuracy of a convolutional neural network with GPU/TPU for image analytics," *PeerJ Comput. Sci.*, vol. 8, p. e909, Mar. 2022, doi: 10.7717/peerj-cs.909.

[35] A. Ravikumar, "Non-relational multi-level caching for mitigation of staleness & stragglers in distributed deep learning," in *Proc. 22nd Int. Middleware Conf. Doctoral Symp.*, Dec. 2021, pp. 15–16.

[36] D. Malmgren-Hansen, A. Kusk, J. Dall, A. A. Nielsen, R. Engholm, and H. Skriver, "Improving SAR automatic target recognition models with transfer learning from simulated data," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 9, pp. 1484–1488, Sep. 2017, doi: 10.1109/LGRS.2017.2717486.

[37] M. Rostami, S. Kolouri, E. Eaton, and K. Kim, "Deep transfer learning for few-shot SAR image classification," *Remote Sens.*, vol. 11, no. 11, p. 1374, Jun. 2019, doi: 10.3390/rs11111374.

[38] Z. Huang, Z. Pan, and B. Lei, "What, where, and how to transfer in SAR target recognition based on deep CNNs," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 4, pp. 2324–2336, Apr. 2020, doi: 10.1109/TGRS.2019.2947634.

[39] S. Chaabouni, J. Benois-Pineau, and C. Ben Amar, "ChaboNet : Design of a deep CNN for prediction of visual saliency in natural video," *J. Vis. Commun. Image Represent.*, vol. 60, pp. 79–93, Apr. 2019, doi: 10.1016/j.jvcir.2019.02.004.

[40] Z. Ding, M. Shao, and Y. Fu, "Transfer learning for image classification with incomplete multiple sources," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2016, pp. 2188–2195, doi: 10.1109/IJCNN.2016.7727470.

[41] J. Dong, Y. Cong, G. Sun, Z. Fang, and Z. Ding, "Where and how to transfer: Knowledge aggregation-induced transferability perception for unsupervised domain adaptation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 46, no. 3, pp. 1664–1681, Mar. 2021, doi: 10.1109/TPAMI.2021.3128560.

[42] K. Wang, G. Zhang, and H. Leung, "SAR target recognition based on cross-domain and cross-task transfer learning," *IEEE Access*, vol. 7, pp. 153391–153399, 2019, doi: 10.1109/ACCESS.2019.2948618.

[43] Z. Wang, L. Du, J. Mao, B. Liu, and D. Yang, "SAR target detection based on SSD with data augmentation and transfer learning," *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 1, pp. 150–154, Jan. 2019, doi: 10.1109/LGRS.2018.2867242.

[44] Z. Huang, C. O. Dumitru, Z. Pan, B. Lei, and M. Datcu, "Classification of large-scale high-resolution SAR images with deep transfer learning," *IEEE Geosci. Remote Sens. Lett.*, vol. 18, no. 1, pp. 107–111, Jan. 2021, doi: 10.1109/LGRS.2020.2965558.

[45] C. Wu, J. H. Lee, T. S. Wan, Y. M. Chan, and C. S. Chen, "Merging well-trained deep CNN models for efficient inference," in *Proc. Asia–Pacific Signal Inf. Process. Assoc. Annu. Summit Conf. (APSIPA ASC)*, Dec. 2020, pp. 1594–1600.

[46] B. P. Babu and S. J. Narayanan, "One-vs-all convolutional neural networks for synthetic aperture radar target recognition," *Cybern. Inf. Technol.*, vol. 22, no. 3, pp. 179–197, Sep. 2022, doi: 10.2478/cait-2022-0035.

[47] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998, doi: 10.1109/5.726791.

[48] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn. (ICML)*, 2010, pp. 807–814.

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

[50] V. N. Vapnik and A. Ya. Chervonenkis, "On the uniform convergence of relative frequencies of events to their probabilities," in *Measures of Complexity: Festschrift for Alexey Chervonenkis*. Cham, Switzerland: Springer, 2015, pp. 11–30, doi: 10.1007/978-3-319-21852-6_3.

[51] T. D. Ross, S. W. Worrell, V. J. Velten, J. C. Mossing, and M. L. Bryant, "Standard SAR ATR evaluation experiments using the MSTAR public release data set," *Proc. SPIE*, vol. 3370, pp. 566–573, Sep. 1998, doi: 10.1117/12.321859.

[52] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 25, 2012, pp. 84–90, doi: 10.1145/3065386.

[53] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[54] 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.

[55] H. Kaiming, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[56] 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. 2261–2269.

BILEESH PLAKKAL BABU received the B.Tech. degree in computer science and engineering from Kerala University, in 2012, and the M.Tech. degree in computer science and engineering from Visvesvaraya Technological University, in 2014. He is currently pursuing the Ph.D. degree in computer science and engineering with Vellore Institute of Technology, India. His research interests include pattern recognition, radar image analysis, machine learning, and deep learning.

SWATHI JAMJALA NARAYANAN (Member, IEEE) received the Ph.D. degree from Vellore Institute of Technology (VIT), Vellore, India, in 2015. She is currently a Professor Grade 1 with the School of Computer Science and Engineering, VIT. She is having 17 years of teaching experience in computer science. Her research interests include soft computing, pattern recognition, machine learning, and data mining. She is a member of the International Association of Engineers. She is also a Lifetime Member of the Computer Society of India and the Soft Computing Research Society. She has been awarded with the Best Ph.D. Thesis by the Computer Society of India. She is a Faculty Coordinator of the IEEE Computer Society Student Chapter in VIT.