Received XX Month, XXXX; revised XX Month, XXXX; accepted XX Month, XXXX; Date of publication XX Month, XXXX; date of current version XX Month, XXXX.

Digital Object Identifier 10.1109/OJIM.2022.1234567

LArcNet: LIGHTWEIGHT NEURAL NETWORK FOR REAL-TIME SERIES AC ARC FAULT DETECTION

KAMAL CHANDRA PAUL*, GRADUATE STUDENT MEMBER, IEEE, CHEN CHEN†, MEMBER, IEEE, YAO WANG‡, MEMBER, IEEE, AND TIEFU ZHAO*, SENIOR MEMBER, IEEE

[1]Department of Electrical and Computer Engineering, University of North Carolina at Charlotte, Charlotte, NC 28223 USA
[2]Center for Research in Computer Vision (CRCV), University of Central Florida, Orlando, FL 32816 USA
[3]School of Electrical Engineering, Hebei University of Technology, Tianjin 300400, China

CORRESPONDING AUTHOR: Kamal Chandra Paul (e-mail: kpaul9@charlotte.edu).

ABSTRACT Detecting series AC arc faults in diverse residential loads is challenging due to variations in load characteristics and noise. While traditional AI-based algorithms can be effective, they often involve high computational complexity, limiting their real-time implementation on resource-constrained edge devices. This paper introduces LArcNet (Lightweight Arc Fault Detection Network), a novel, lightweight, and rapid-response algorithm for series AC arc fault detection. LArcNet combines a teacher-student knowledge distillation approach with an efficient convolutional neural network architecture to achieve high accuracy with minimal computational demand. This streamlined yet robust design makes LArcNet ideally suited for resource-constrained embedded systems, achieving an arc fault detection accuracy of 99.31%. The model is optimized and converted into TensorFlow Lite format to reduce size and latency, enabling deployment on low-power embedded devices such as the Raspberry Pi and the STM32 microcontrollers. Test results demonstrate LArcNet's inference times of just 0.20 ms on the Raspberry Pi 4B and 3 ms on the STM32H743ZI2, surpassing other leading models in operational speed while maintaining competitive accuracy in arc fault detection.

INDEX TERMS Arc discharge, arc fault detection, artificial neural network, convolutional neural network (CNN), deep learning, knowledge distillation, machine learning, series AC arc.

I. INTRODUCTION

ARC fault, characterized by the luminous discharge of electricity, poses significant safety hazards. These faults often arise from factors such as loose cable connections, wire aging, insulation breakdown due to external intrusion, or physical damage to cables. Capable of raising temperatures beyond 5000^oC, even at low currents ranging from 3 A to 12 A, arc faults can lead to electrical fires, resulting in property damage, injury, or even fatalities. In the United States, arc fault is one of the leading causes of residential fire hazards. A survey by Zebra found that 36.3% of fire hazards are attributed to electrical issues [1], while the Industrial Safety and Hygiene News reports over 30,000 arc flash incidents annually, leading to significant casualties and hospitalizations. Standards set by entities like the International Electrotechnical Commission (IEC), National Electrical Code (NEC), and Underwriters' Laboratories (UL) mandate arc fault detection mechanisms in household appliances for safety [2]–[4]. In the U.S., devices such as arc fault circuit interrupters (AFCI) or arc fault detection devices (AFDD) have been compulsory in households since 2002.

Arc faults are typically categorized into series and parallel types as shown in Fig. 1. Parallel arc faults are relatively easier to detect due to their high current characteristics, unlike series AC arc faults, which are more challenging to identify due to series impedance limitations and ambiguous arc features. This complexity frequently leads to false alarms or nuisance tripping, a problem exacerbated when certain residential loads inherently draw current in a manner similar to arc faults under normal operating conditions [5].

Conventional arc fault detection techniques primarily utilize feature extraction from time or frequency domains, and often integrate both approaches. One of these techniques is the Fast Fourier Transform (FFT), which is employed

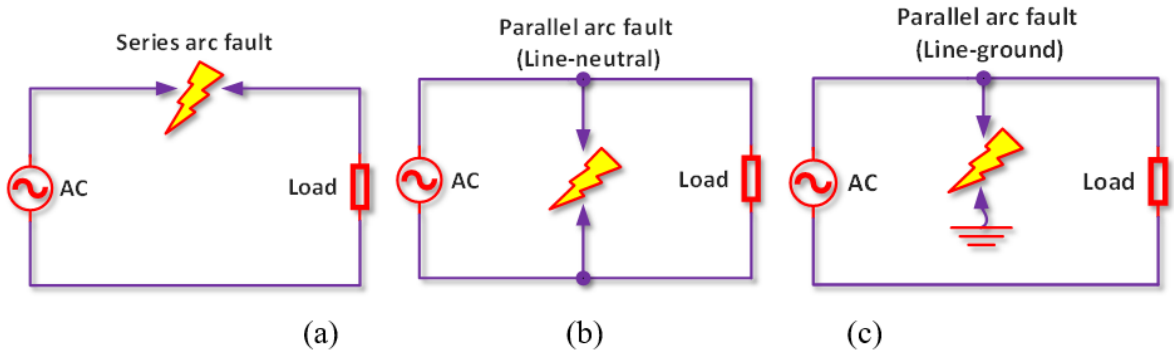


FIGURE 1. **Types of arc faults- (a) series arc fault; (b) parallel arc fault (line-neutral); (c) parallel arc fault (line-ground).**

to transform a signal from its original time domain into the frequency domain. Another noteworthy method is the Wavelet Transform (WT), which decomposes a signal into its constituent wavelets, providing a detailed time-frequency representation. A specialized variant of WT, the Discrete Wavelet Transform (DWT), is tailored for analyzing discrete or quantized data sets [6]–[9]. In addition, the Chirp Zeta Transform (CZT) is utilized specifically for detecting series AC arc faults. However, a common limitation of these methods is their reliance on manual threshold adjustments, which can lead to false positives under conditions of varying load and environmental noise [10].

Due to their high classification accuracy, artificial intelligence (AI) and machine learning (ML) algorithms are increasingly researched for arc fault detection. A variety of classification algorithms, including Support Vector Machine (SVM), Particle Swarm Optimization combined with Self-Organizing Map Neural Network, Recurrent Neural Network (RNN), Learning Vector Quantization Neural Network (LVQ-NN), Decision Tree-based Algorithm, Random Forest, Backpropagation Neural Network, Artificial Neural Network, and Convolutional Neural Network, have been employed for arc fault classification, often coupled with data preprocessing techniques [11]–[23]. Some of these algorithms also possess the capability to classify load types or groups where arc faults occur [12]–[14]. However, these methods typically involve time domain, frequency domain, or a combination of both for feature extraction prior to data input into neural network models. Notably, many studies do not report on the real-time applicability of these models, or they exhibit runtimes too extensive for practical implementation in lower-end commercial microcontroller units (MCUs) [15]–[18], [20], [24]–[26]. For instance, Wang *et al.* [27] employed a lightweight CNN architecture for series arc fault detection in specific load types, but the low sampling frequency of 2.5 kHz could potentially miss crucial arc signatures in the input current signal. Additionally, the use of 2D input matrix formation, through point-by-point isometric mapping, introduces unnecessary computational load, which could be streamlined to 1D time series data for model simplification.

According to the International Electrotechnical Commission (IEC) standard [2], recommended maximum break times for a 230V power supply system are 1 s for 2.5 A current and 120 ms for 63 A load current, with corresponding times for a 120 V system being 1 s for 5 A and 140 ms for 63 A load currents. In real-time operation, considering data preprocessing, acquisition, testing times, possible signaling delays, and circuit breaking times, an effective arc fault detection system should detect faults in less than 16.67 ms for a 60 Hz power supply system, or less than 20 ms for a 50 Hz system. Models that are too cumbersome in inference time compared to data acquisition may miss critical samples, potentially leading to undetected arc faults. Additionally, while large-scale deep neural networks may excel in classification due to over-parameterization and generalization, their computational complexity necessitates substantial storage space and memory. Pre-extraction of primary features adds to this burden makes their deployment on resource-limited edge devices challenging.

Therefore, the development of an efficient, lightweight deep learning model for arc fault detection is essential. Such models can be designed using efficient building blocks like depthwise separable convolution blocks (as seen in MobileNet, ShuffleNet, and EffNet) or through model compression and acceleration techniques like parameter pruning and sharing, low-rank factorization, and transferred compact convolutional filters [28]–[30]. Knowledge distillation (KD) has recently garnered attention for its ability to transfer knowledge from larger, more complex networks to smaller, simpler ones, resulting in highly efficient and lightweight models [31], [32].

This paper proposes a CNN-based series AC arc fault detection algorithm that combines a lightweight CNN architecture with a teacher-student knowledge distillation technique. This model achieves high accuracy in arc fault detection. It is optimized using the TensorFlow Lite (Lite-RT) tool, resulting in a reduced binary size suitable for implementation in resource-limited edge devices. The performance of the optimized model is evaluated on Raspberry PI 4B and STM32H743ZI2 devices,

TABLE 1. Brief database of different load groups

Load Groups	Loads	Total Samples
Resistive type loads (RE)	Electrical heater, electric iron, incandescent lamps, electric kettle	12675
Motor loads (MO)	Capacitor start motor, vacuum cleaner, electric hand tool (drill)	6373
Power Electronics-enabled and SMPS loads (PESMPS)	Switch-mode power supply loads, dimmer (thyristor type)	7411
Gas discharge lamp (GDL) loads	Halogen lamps, fluorescent lamps	3692
Total number of samples		30151

demonstrating its practicality and real-time operational capability.

The key contributions of this work are as follows:

1) This article proposed LArcNet, an ultra-fast & lightweight AI algorithm, to detect series AC arc faults using raw current data collected from a test bench as per the IEC62606 standard. This model combines a teacher-student knowledge distillation method with an efficient CNN architecture, achieving a high fault detection accuracy of 99.31%.
2) The model avoids the pitfalls of traditional bottleneck designs to ensure low computational cost. To optimize and enhance LArcNet for real-time use on resource-constrained devices, model compression strategies including knowledge distillation and mixed precision training were applied. These adaptations, alongside TensorFlow Lite optimization significantly minimized LArcNet's memory usage and computational load. The optimized LArcNet model demonstrated an inference time of only 0.20 ms in Raspberry Pi 4B and 3 ms in STM32H743ZI2 MCU making it highly suitable for real-time fault detection in embedded systems.
3) The proposed model is designed to detect arc faults and identify the load types associated with these faults. It achieves an 8-class load classification accuracy of 98.85%.

The rest of this paper is arranged as follows. Section II provides a brief overview of the data collection method and arc characteristics. The methodology and architecture of the proposed LArcNet model are illustrated in section III. Experimental settings and implementation details have been depicted in section IV. Experimental results, discussions, and hardware implementation have been provided in section V. Comparison of the proposed model with state-of-the-art models has been discussed in section VI. Section VII concludes this paper and indicates the future research prospect.

II. ARC FAULT DATABASE

A. DATA COLLECTION AND DESCRIPTION OF THE DATABASE

The data for this study was collected from a test bench designed following the IEC62606 standard [2], featuring a microcontroller-based Arc Data Acquisition (DAQ) board. Arc faults were initiated using an arc generator (Fig. 2) with graphite and copper electrodes, as well as a cable specimen. Current sensing was done through a current transformer (CT) within a 220 V, 50 Hz power system. The dataset comprises 30,151 samples divided among four major load types, each sampled at 83.33 kHz and subsequently downsampled to 10 kHz, resulting in 200-point 1D time series data per sample normalized via Min-Max scaling. Additionally, 2,266 samples, including arc fault, normal, and transient conditions across various load types, were added for training and testing the model.

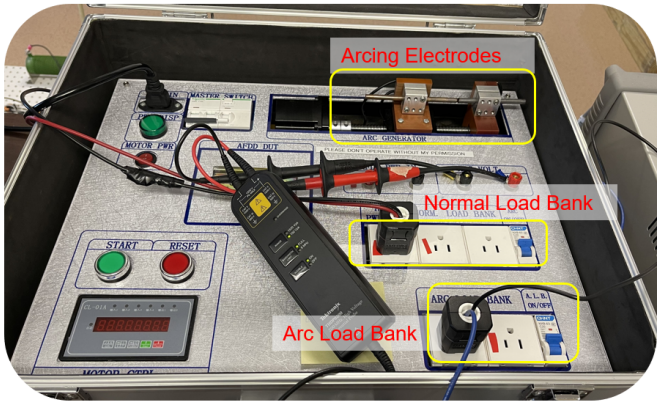


FIGURE 2. Arc fault generation unit consisting of a stationary electrode and a movable electrode.

Arc faults were realistically generated to reflect common issues like loose connections and insulation failures. Resistors were added to comply with IEC62606 power specifications. The loads were categorized into four types: Resistive (RE), Motor (MO), Power Electronics-Enabled and Switch-mode Power Supplies (PESMPS), and Gas Discharge Lamps (GDL). Resistive loads exhibit near-sinusoidal currents, motors display high inrush currents, PESMPS include broad harmonic content from devices like dimmers and computers, and GDL pertains to various lamps. The dataset is detailed in Table 1. Arcing currents are labeled with even numbers and normal currents with odd numbers for precise identification of load type as well as an arc.

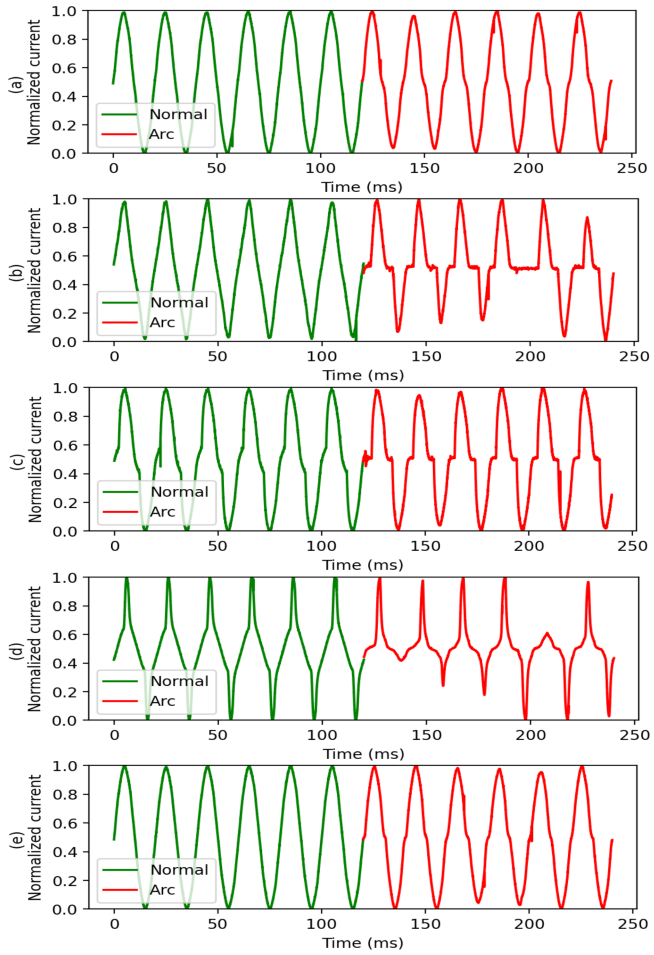


FIGURE 3. **Visualization of normal & arcing currents of a) resistive, b) electric drill, c) thyristor type dimmer, d) switch-mode power supply, and e) halogen lamp loads.**

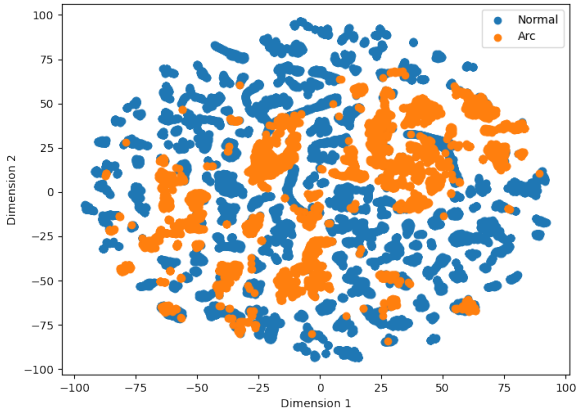


FIGURE 4. **T-SNE projection of arc fault data (all load categories) in the frequency domain.**

B. CHARACTERISTICS OF ARC FAULTS AND VISUALIZATION OF DATA

Arcing currents have distinct features compared to normal load currents, including waveform distortions, reduced amplitude, increased high-frequency harmonics, and shorter conduction angles. These characteristics can vary significantly between different load types, and some normal currents (e.g., from dimmers) may resemble arcing currents (e.g., from heaters), complicating identification. The unique attributes of arcing include prolonged stagnation periods and intensified waveform distortions, which are illustrated in Fig. 3 comparing normal and arcing currents.

To analyze arc fault data more thoroughly, Fast Fourier Transformation (FFT) was applied at a 40 kHz sampling rate to capture high-frequency details. Using t-Distributed Stochastic Neighbor Embedding (T-SNE), the data was visualized in a lower-dimensional space to maintain data point relationships. This visualization, depicted in Fig. 4, reveals a complex distribution of arcing and normal currents, emphasizing the challenge of establishing a definitive threshold for their distinction in the frequency domain.

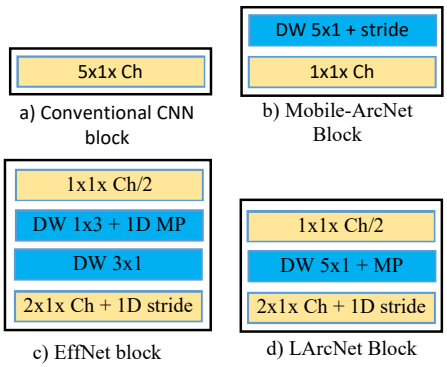


FIGURE 5. **Neural network model building blocks a) conventional CNN, b) Mobile-ArcNet [29], c) EffNet [30] and d) LArcNet. 'DW', 'MP' & "Ch" denote depthwise convolution, Max Pooling, and number of filters, respectively.**

III. PROPOSED METHODOLOGY

This section elaborates on the fundamental building block of LArcNet, architecture and methodology underlying the proposed LArcNet model. LArcNet integrates advanced design strategies from both efficient neural network architectures, such as EffNet [30] and network compression techniques including knowledge distillation. The LArcNet model is constructed using the knowledge distillation method and a series of LArcNet blocks (as shown in Fig. 5d), each with specific number of filters. This hybrid approach combines the strengths of these methods to optimize performance and reduce computational load. The model is compared against a baseline CNN model constructed using conventional CNN blocks (Fig. 5a) and the Mobile-ArcNet model which was constructed following the MobileNet architectural block (Fig. 5b).

A. BUILDING BLOCK OF THE PROPOSED MODEL

The LArcNet model employs an efficient network architecture inspired by the EffNet block [30] as illustrated in Fig. 5c. The building block of LArcNet is depicted in

This article has been accepted for publication in IEEE Open Journal of Industry Applications. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJIA.2024.3522364

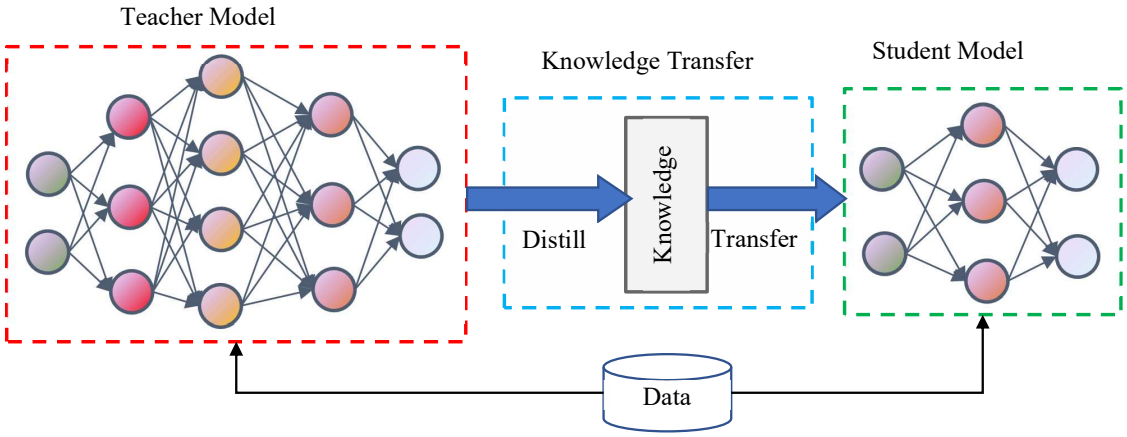


FIGURE 6. **The framework of teacher-student model for knowledge distillation.**

Fig. 5d. Designed to minimize computational complexity, the LArcNet block begins with a pointwise convolution layer, followed by depthwise convolution and Max Pooling in the subsequent sub-layer. This configuration significantly reduces computational demands and the number of trainable parameters compared to traditional CNN structures. Notably, the pointwise convolution layer uses half the number of filters typically found in standard CNN models. The depthwise separable layer, sized 5x1, maintains the initial filter count and is followed by a 1D Max Pooling layer with a pool size of 2x1. The subsequent 2x1 convolutional layer employs the full filter count with a 1D stride of 2. Diverging from MobileNet's design, LArcNet avoids stringent bottleneck structures, leading to a lighter model. Focused on 1D time-series data, LArcNet omits certain sub-layers from the EffNet block. Experiments demonstrated that a 5x1 kernel size in the depthwise sub-layer achieves a better balance between computational demand and performance efficiency than the 3x1 size.

B. KNOWLEDGE DISTILLATION ALGORITHM

The proposed LArcNet model was constructed from sequential LArcNet blocks and incorporates the teacher-student knowledge distillation (KD) technique. This method serves as a model compression strategy for deep neural networks, enhancing the model's efficiency significantly. Knowledge distillation is predicated on a paradigm where a larger, more complex teacher network imparts knowledge to a smaller student network. The student network strives to emulate the teacher network's functionality, often achieving comparable, if not superior, performance. The KD technique is composed of three fundamental elements: the knowledge itself, the teacher-student architectural framework, and the distillation process. A generalized schematic of the teacher-student configuration for knowledge distillation has been illustrated in Fig. 6. The proposed LArcNet model employs response-based knowledge, featuring a robust teacher model paired with a less complex student model. The strength of the teacher model ensures a comprehensive extraction of information from the dataset. This knowledge is subsequently refined and imparted to the student model, which, despite its simpler architecture, retains the core structural framework of the teacher model, including the convolutional neural network (CNN) and fully connected (FC) layers. However, the student model differentiates itself by reducing the number of filters and neurons in these layers. The distilled student model, representing LArcNet, was then deployed for arc fault detection.

Within the LArcNet architecture, the output layer employs a 'softmax' activation function. This function transforms the logits, denoted as z_i, corresponding to each class into normalized probabilities, represented as p_i. It accomplishes this by exponentiating each logit, followed by normalization, which involves dividing the exponentiated logit of a given class by the sum of exponentiated logits for all classes. This comparative process ensures that the probabilities of all classes sum up to one, thereby providing a probabilistic interpretation of the model's outputs. p_i is expressed as follows,

$$p_i = \frac{e^{\frac{z_i}{T}}}{\sum_j e^{\frac{z_j}{T}}} \tag{1}$$

where, the term T denotes the temperature parameter within the softmax function. Typically, T is assigned a default value of 1. However, an increase in T results in a "softer" probability distribution across the classes, effectively smoothing the output probabilities. The teacher model undergoes training with a transfer set, utilizing an elevated value of T within its softmax function to generate softened class distributions. This approach facilitates the transfer of knowledge to the student model, which is trained using this softened probability distribution as a target for each instance in the transfer set.

For instances where the correct labels are known within the transfer set, the efficacy of this technique can be augmented by training the student model with a weighted average of two distinct objective functions. The first

objective function is the cross-entropy with the softened targets produced by the teacher model, calculated at the elevated temperature used to soften the probabilities. The second objective function is the cross-entropy with the actual labels, computed using the logits from the softmax function of the student model at the standard temperature of $T = 1$.

Suppose the comprehensive teacher model yields a softened probability denoted by q_i and possesses logits labeled as u_i. The cross-entropy loss in this framework can be expressed as follows:

$$C(x) = -\sum_i q_i(x) \log p_i(x) \tag{2}$$

where, x is the input feature. The cross-entropy gradient $\frac{\partial C}{\partial z_i}$, of the distilled model [31], [33] can be expressed as

$$\begin{aligned}
\frac{\partial C}{\partial z_i} &= \frac{\partial}{\partial z_i}\left(-\sum_i q_i \log p_i\right) \\
&= -q_i \frac{1}{p_i}\left(\frac{\partial p_i}{\partial z_i}\right) \\
&= -q_i \frac{1}{p_i}\frac{\partial}{\partial z_i}\left(\frac{e^{\frac{z_i}{T}}}{\sum_j e^{\frac{z_j}{T}}}\right) \\
&= -q_i \frac{1}{p_i}\left[\frac{\frac{1}{T}e^{\frac{z_i}{T}}\sum_j e^{\frac{z_j}{T}} - \frac{1}{T}\left(e^{\frac{z_i}{T}}\right)^2}{\left(\sum_j e^{\frac{z_j}{T}}\right)^2}\right] \\
&= -q_i \frac{1}{p_i}\frac{1}{T}\left[\frac{e^{\frac{z_i}{T}}}{\sum_j e^{\frac{z_j}{T}}} - \left(\frac{e^{\frac{z_i}{T}}}{\sum_j e^{\frac{z_j}{T}}}\right)^2\right] \\
&= \frac{1}{T}(p_i - q_i) \\
&= \frac{1}{T}\left(\frac{e^{\frac{z_i}{T}}}{\sum_j e^{\frac{z_j}{T}}} - \frac{e^{\frac{u_i}{T}}}{\sum_j e^{\frac{u_j}{T}}}\right)
\end{aligned} \tag{3}$$

where, u_i represents the logits derived from the teacher model, while q_i signifies the softened target probabilities calculated at a transfer training temperature T. When employing a high temperature setting, the gradient of the cross-entropy loss with respect to the logits, z_i, can be approximated as follows:

$$\frac{\partial C}{\partial z_i} \approx \frac{1}{T}\left(\frac{1 + \frac{z_i}{T}}{N + \sum_j \frac{z_j}{T}} - \frac{1 + \frac{u_i}{T}}{N + \sum_j \frac{u_j}{T}}\right) \tag{4}$$

Under the assumption of zero-mean logits for each transfer case as per Hinton *et al.* [31], where $\sum_j z_j = \sum_j u_j = 0$, equation (4) simplifies to:

$$\frac{\partial C}{\partial z_i} \approx \frac{1}{NT^2}(z_i - u_i) \tag{5}$$

With the logits assumed to have zero mean for each transfer case, the process of distillation becomes analogous to minimizing the squared error $\frac{1}{2}(z_i - u_i)^2$ in the high-temperature limit of T. At lower temperatures, the model's focus on matching logits diminishes. Consequently, the teacher model's loss function is not explicitly constrained during training, which may introduce noise. Moreover, logits with negative values provide insightful information about the knowledge encoded by the teacher model.

C. NETWORK ARCHITECTURE OF THE PROPOSED MODEL

The LArcNet model architecture includes both a teacher and a student network, each featuring three LArcNet blocks and three fully connected (FC) layers. The teacher network uses 256, 512, and 512 filters across its blocks, with FC layers of 128, 64, and 8 neurons, respectively. However, the final model—referred to as LArcNet—is the student network, optimized for deployment. This student network, designed for efficiency, uses fewer filters—16, 32, and 32—and has FC layers with 64, 32, and 8 neurons to balance performance and simplicity without sacrificing accuracy Fig. 7 and Table 2 depict their architectures data flow respectively.

An aggressive bottleneck for data flow through a network can lead to a significant reduction of important features, which may negatively impact smaller, deep models. The LArcNet model addresses this by minimizing computational complexity and kFLOPs, achieved through an efficient block design and knowledge distillation to reduce the number of trainable parameters. Unlike the Efficient-ArcNet model, which has a bottleneck factor of 4 (highlighted in red in Table 2), the LArcNet student model limits the bottleneck factor to a maximum of 2 (highlighted in green), promoting smoother data flow and enhanced efficiency. This approach enables the use of narrower models without compromising the retention of critical feature information.

IV. EXPERIMENTAL SETTINGS AND IMPLEMENTATION DETAILS

This section details the experimental setup and implementation of the LArcNet model. The dataset was split into training, testing, and validation sets in a 75:15:10 ratio, with labels converted to OneHot encoding. Model inputs consisted of raw current data sampled at 10 kHz, normalized using Min-Max normalization.

The model training used mixed precision, with 16-bit half-precision for data processing and 32-bit single-precision for weight updates and loss scaling. Both the teacher and student models were trained over 300 epochs with a batch size of 100, utilizing an Adam Optimizer with learning rate of 0.001. Parameters for the distillation process included an alpha value of 0.5 and a temperature of 20, which provided nuanced guidance and helped prevent overfitting. The training and validation accuracies and losses for both models were closely monitored, as depicted in Fig. 8. The training and validation trendlines closely align, indicating no overfitting.

Baseline and Mobile-ArcNet models underwent 250 epochs of training with an adaptive learning rate strategy starting from 0.001, adjusted based on validation loss. Both

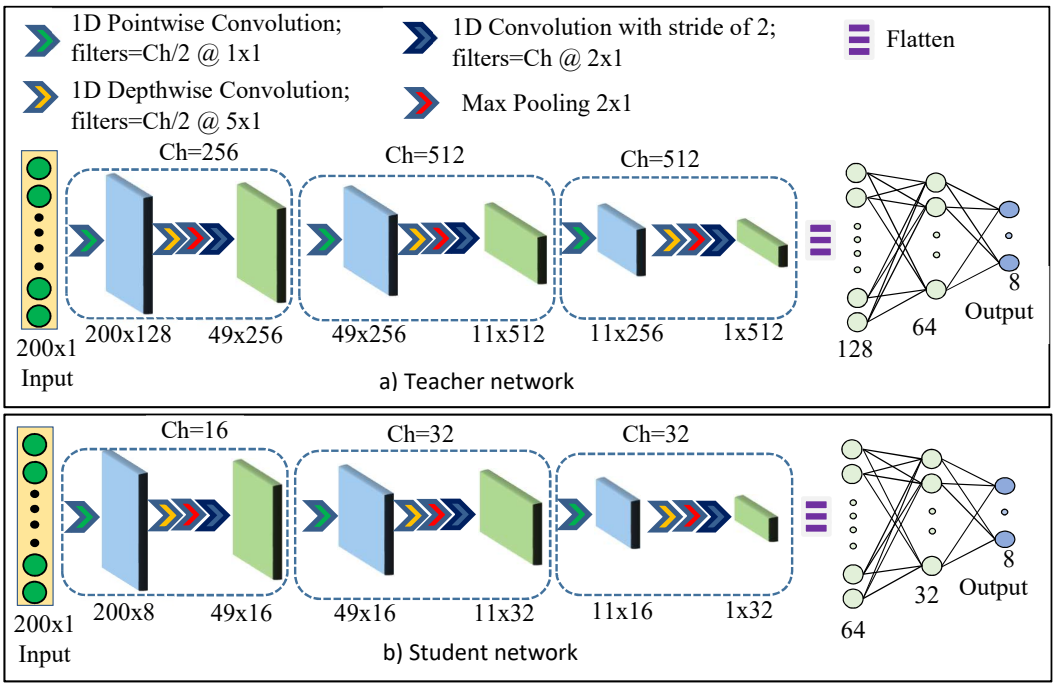


FIGURE 7. **Architectures of LArcNet (a) teacher and (b) student networks featuring three LArcNet blocks each. Blocks include a pointwise convolution with Ch/2 filters, a 5x1 depthwise convolution, a 2x1 Max Pooling layer, and a 2x1 convolution with Ch filters, ending with three fully connected layers, the last containing 8 neurons.**

TABLE 2. **Network architecture of LArcNet (student model) compared with Baseline, Mobile-ArcNet, and Efficient-ArcNet models. FC (64,32,8) indicates three fully connected layers with 64, 32, and 8 neurons, targeting 8 output classes for four load groups. 'dw' and 'mp' represent depthwise convolution and Max Pooling layers. Efficient-ArcNet features a bottleneck factor of 4 (red), while LArcNet's bottleneck factor is up to 2 (green).**

Baseline		Mobile-ArcNet		Efficient-ArcNet [13]		LArcNet (Proposed)	
Layers	Params	Layers	Params	Layers	Params	Layers	Params
5x1x96 + mp of 2	576	5x1x96 + mp of 2	576	1x1x48 dw 5x1 + mp of 2 2x1x96 + stride of 2	96 2592 9,312	1x1x8 dw 5x1 + mp of 2 2x1x16 + stride of 2	16 112 272
5x1x128 + mp of 2	61,568	dw 5x1 + stride of 2 1x1x128	12,896 16,512	1x1x64 dw 5x1 + mp of 2 2x1x128 + stride of 2	9,208 4,480 16,512	1x1x16 dw 5x1 + mp of 2 2x1x32 + stride of 2	272 352 1,056
5x1x128 + mp of 2	82,048	dw 5x1 + stride of 2 1x1x128 + mp of 4	17,152 16,512	1x1x64 dw 5x1 + mp of 2 2x1x128 + stride of 2	4,480 16,512 8,256	1x1x16 dw 5x1 + mp of 2 2x1x32 + stride of 2	528 352 1,056
FC (64,32,8)	17,4440	FC (64,32,8)	41,568	FC (64,32,8)	10,600	FC (64,32,8)	4,456
Total Params	318,016		107,016		79,048		8,472

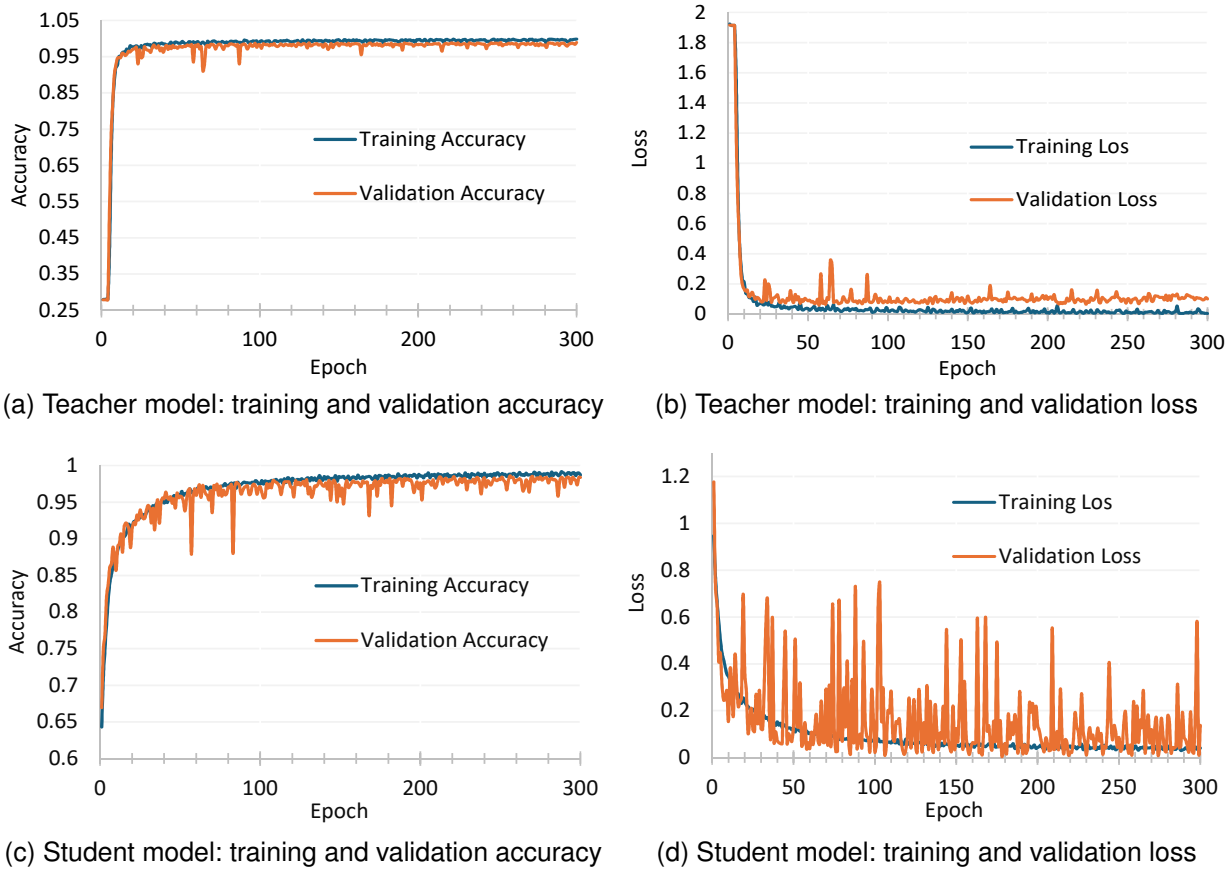
models used a batch size of 100, a patience parameter of 10, and a reduction factor of 0.1 for learning rate adjustments.

All models were trained using TensorFlow with Keras, employing "categorical_crossentropy" as the loss function and ReLU in the convolution layers. The output layers used the "softmax" activation function. Mixed-precision training involved converting data samples to 16-bit floating points for efficiency. Extensive hyperparameter tuning optimized the model's performance.

The experimental circuit diagram for offline analysis of arc fault detection and implementation using Raspberry Pi 4B along with a few test loads are shown in Fig. 9. The model is also tested on the STM32H743ZI2 platform.

V. EXPERIMENTAL RESULTS AND DISCUSSION

This section presents the experimental results of the LArcNet model along with the Baseline and Mobile-ArcNet models.



(a) Teacher model: training and validation accuracy

(b) Teacher model: training and validation loss

(c) Student model: training and validation accuracy

(d) Student model: training and validation loss

FIGURE 8. Training and Validation Accuracy and Loss for teacher and student models. The left figures show the accuracy metrics, while the right figures display the loss metrics for both models over the training epochs.

A. EXPERIMENTAL RESULTS

In addition to testing the LArcNet model, other comparative lightweight and baseline models were also verified using the same dataset. LArcNet_FFT model has the same architecture as the student model except it has frequency domain data as input. All models were trained on the same dataset. The Baseline model, comprising only conventional convolutional blocks, has the highest number of trainable parameters and limited computational efficiency, despite the inclusion of Max Pooling layers. Conversely, the Mobile-ArcNet model has a higher parameter count (107.2k) compared to LArcNet and was not trained using knowledge distillation. The student model of LArcNet, however, utilized the knowledge distillation technique, benefiting from the comprehensive teacher model. The LArcNet student model features the smallest number of trainable parameters among the models, totaling only 8.4k. Both the Raspberry Pi 4B and the STM32H743ZI2 microcontrollers were used model implementation.

The kFLOPS (kilo floating-point operations per second) were estimated for each model to assess computational complexity. FLOPs is a measure of the performance of a computer system. It is used to estimate the amount of computational resources it requires to perform a certain task. Higher FLOPs values correspond to more complex computations. In comparison to the Baseline model, the Mobile-ArcNet model requires only 24% of the computational resources (0.24 factor), while the LArcNet model requires just 1% (0.01 factor), with a total computational load of only 182.27 kFLOPs. These results indicate that the proposed LArcNet model has the lowest computational complexity, making it highly lightweight. Detailed results are presented in Table 3.

Accuracy-wise, LArcNet achieved the highest arc fault detection accuracy of 99.31%. The baseline model scored the lowest in arc fault detection at 99.22% (Table 3). The Load classification and arc fault classification accuracy using the LArcNet model is depicted in the confusion matrix (Table 4), highlighting the accuracy for 8-class classifications of different load groups and misclassification percentages. Even-numbered labels indicate arc faults, and odd-numbered labels normal currents. The GDL loads demonstrated the lowest accuracy due to their characteristics, where normal currents can resemble arc faults, leading to significant misclassification. Approximately 10.23% GLD arcs are mistakenly classified as resistive arcs. Overall binary classification accuracy for arc fault detection was calculated

This article has been accepted for publication in IEEE Open Journal of Industry Applications. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJIA.2024.3522364

TABLE 3. Experimental results of LArcNet and other models. Runtime is evaluated in Raspberry PI 4B.

Model[a]	8 Class Accuracy (%)	Arc Classification Accuracy (%)	Trainable Params	kFLOPs	Factor	Runtime (ms)
KD Teacher	99.14	99.25	1013.83 k	35,977.4	1.87	*
Baseline	98.96	99.22	318.63 k	19205.4	1	3.27
Mobile-ArcNet	99.00	99.29	107.02 k	4542.93	0.24	1.43
LArcNet_FFT[c]	-	94.63	14.62 k	379.52	0.02	1.00[b]
SVM Model	-	99.20	-	-	-	15
LArcNet (Proposed)[d]	98.85	99.31	**8.4 k**	**182.27**	**0.01**	**0.20**

[a] All models are optimized using Lite-RT optimization tool. Data normalization per sample takes only 0.8μs.

[b] Time taken to perform FFT per sample is 0.67 ms and inference time is 0.33 ms

[c] Input to the model is 1D data in the frequency domain (40 kHz)

[d] Input to the model is raw data with Min-Max normalization.

* Didn't implement in MCU due to heavy computational cost (kFLOPS).

TABLE 4. Confusion matrix for 8 class classification using LArcNet model. Correctly predicted percentage values are diagonal bold face numbers. Even numbered labels are arc classes and odd numbered labels represent normal classes.

		Predicted Class							
	Label	0	1	2	3	4	5	6	7
True Class: Resistive loads	0	**97.63%**	1.74%	0.47%	0%	0%	0%	0%	0.16%
	1	0.47%	**99.29%**	0%	0%	0%	0%	0%	0.24%
True Class: Motor Loads	2	0.35%	0%	**98.25%**	0.70%	0.35%	0%	0.35%	0%
	3	0%	0%	0%	**100%**	0%	0%	0%	0%
True Class: Power electronics & SMPS loads	4	0.23%	0%	0.46%	0%	**97.23%**	2.08%	0%	0%
	5	0%	0%	0%	0%	0%	**100%**	0%	0%
True Class: Gas Discharge lamps	6	10.23%	0%	0%	1.14%	0%	0%	**87.50%**	1.14%
	7	0%	0%	0%	0%	0%	0%	0%	**100%**

by summing correctly classified arc faults and dividing by the total number of test samples.

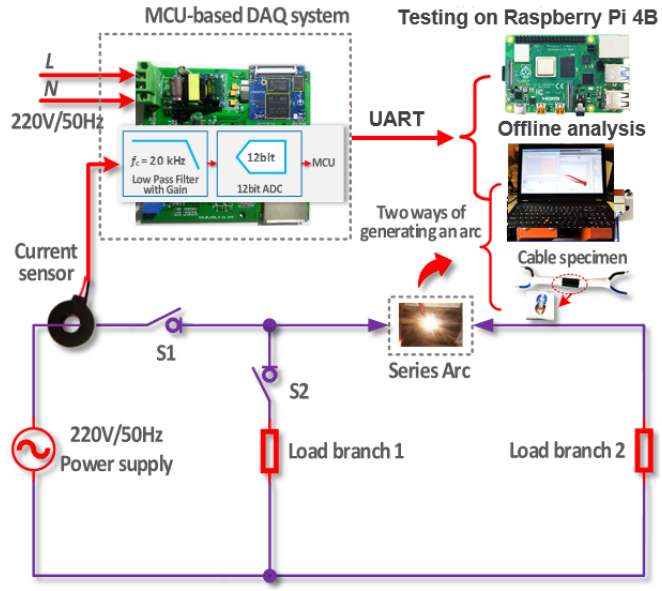
Precision, recall, and binary classification accuracy for arc and normal current samples were determined without considering load classification. The precision and recall matrix of the LArcNet model, presented in Table 5, was derived from Table 4. High precision and recall values indicate the model's practicality and low rate of false positives.

The teacher model, with over 1,013 k trainable parameters and an 11 MB file size, serves as a robust, high-accuracy benchmark, achieving approximately 99.14% accuracy for load classification and 99.25% accuracy for arc fault classification. The smaller, distilled student model, with only 8.4 k parameters and a file size of about 90 kB, demonstrates the effectiveness of knowledge distillation, achieving an impressive 99.31% accuracy for arc fault classification. The primary benefit of knowledge distillation is creating a high-performance, lightweight model that operates efficiently on resource-limited devices, such as edge processors.

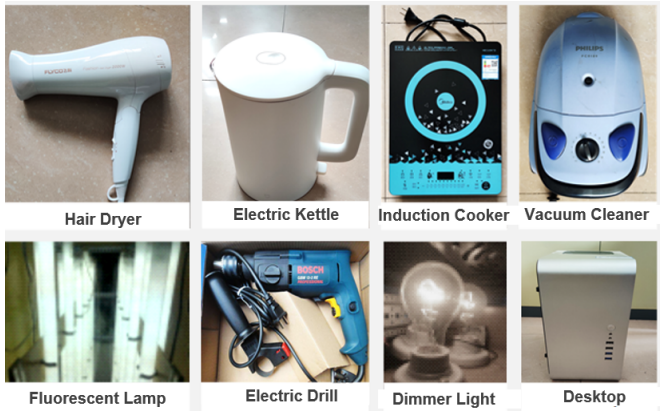
While the teacher model achieves high accuracy, it is computationally intensive and unsuitable for real-time or embedded applications due to its large size and high complexity. The distillation process transfers critical, high-level feature representations from the teacher to the student model, enabling the student to approximate the teacher's performance with a fraction of the computational resources. This transfer allows the student model to maintain high accuracy in arc fault detection while being deployable in environments where storage, memory, and computational capacity are constrained, which would not be feasible with the teacher model alone.

Furthermore, experiments exploring additional simplifications of the student model reveal that while some runtime gains are achievable, they come at the cost of a significant drop in accuracy, falling to around 95%. Thus, the teacher-student approach with knowledge distillation represents an optimal solution, providing a compact, deployable model that retains high accuracy, a critical requirement for real-world applications.

This article has been accepted for publication in IEEE Open Journal of Industry Applications. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJIA.2024.3522364



(a) Experimental diagram



(b) Pictures of a few test loads

FIGURE 9. **Experimental platform of arc fault detection and a few test loads.**

The LArcNet model adopts a streamlined yet highly efficient structure, designed to leverage knowledge distillation for enhanced arc fault detection. It includes a large and comprehensive teacher model that excels at capturing arc features with high accuracy, allowing the student model to achieve a similar level of performance while remaining lightweight. The Mobile-ArcNet model retains its original first layer without a depthwise layer, whereas LArcNet replaces this first layer with more efficient design blocks, significantly reducing computational load. Additionally, the model incorporates a bottleneck factor of 2, which optimizes data flow and reduces computational complexity. LArcNet uses pointwise and depthwise separable convolutions with a moderate bottleneck to reduce kFLOPs, creating a compact, accurate model for series AC arc fault detection. This design lowers computational costs and runtime, making it ideal for real-time, resource-limited applications.

TABLE 5. **Precision, recall, and accuracy of LArcNet**

		Predicted Class		
		Arc	Normal	Total
Actual Class	Arc	**1413**	25	1438
	Normal	6	**3069**	3075
	Total	1419	3094	4513

Precision	99.58%
Recall	98.26%
F1 Score	98.92%
Overall accuracy	99.31%

B. MODEL OPTIMIZATION AND HARDWARE IMPLEMENTATION

TensorFlow Lite (Lite-RT) [34], an open-source deep learning framework, offers tools for running TensorFlow models on low-power edge devices, such as microcontrollers and mobile devices with limited resources. Lite-RT enables model optimization to achieve reduced binary size and latency. The student model of LArcNet has been optimized using Lite-RT, resulting in a substantial decrease in binary size (from 600 kB to 49 kB) and enhanced efficiency for edge device operations. The optimization and conversion to a Lite-RT model were conducted using TensorFlow's Python API.

The optimized LArcNet model was implemented on a Raspberry Pi 4B to assess its performance. This process involved loading the Lite-RT model, initializing the interpreter, and allocating tensors to set up the model's input and output specifications. The model was then tested using all test samples, with performance measured across five runs. Results are presented in Table in Table 6. The average inference time per sample for the optimized LArcNet model was notably swift, at just 0.20 ms. This optimization process did not compromise accuracy. The recorded test time encompasses sample retrieval, Min-Max normalization, label fetching, and sample testing. Fig. 11 provides a graphical comparison of runtime and accuracy across different models, illustrating that LArcNet surpasses other models in runtime efficiency. Additionally, the variance in runtime across five runs was low (average variance of 0.00296), indicating consistent performance.

Parallel to LArcNet, the Baseline, Mobile-ArcNet, and LArcNet_FFT models were also optimized and their inference times were assessed. Table 3 summarizes the inference time per sample for these models. LArcNet demonstrates superior performance in both accuracy and runtime, confirming its suitability for real-time practical implementation in commercial MCUs. The runtime assessments were based on testing samples covering one cycle of power frequency, underscoring the model's efficiency for real-time applications.

TABLE 6. Runtime per sample for LArcNet model using Raspberry Pi 4B MCU.

Sl. No.	Average runtime (ms)	Largest runtime (ms)	Smallest runtime (ms)	Variance of Runtime
1	0.201	1.24	0.199	0.0004
2	0.20	1.176	0.2	0.0002
3	0.202	1.201	0.199	0.0003
4	0.199	1.300	0.2	0.0004
5	0.203	1.258	0.198	0.00035
Average	**0.20**	**1.217**	**0.198**	**0.000365**

Upon deployment of the proposed model on the STM32H743ZI2 MCU, the system utilizes 57.17 kB of flash memory and 13.04 kB of RAM, demonstrating a compact footprint suitable for embedded applications. The model's complexity is quantified at 95,544 Multiply-Accumulate Cycles (MACC), reflecting its computational demand. On average, one sample inference takes about 3 ms (see Table 7), underscoring the model's capability to perform in real-time scenarios. These metrics highlight the model's efficiency and compatibility with resource-constrained environments, emphasizing its potential for integration into real-time systems where memory and processing power are limited. No compression techniques were applied in this instance, yet the model maintains a balance between performance and resource usage, confirming its practical applicability in such settings. Analysis of these deployment characteristics has been completed, affirming the model's readiness for operational use.

TABLE 7. Runtime comparison of LArcNet model in different platforms.

Model	Platform	Average runtime (ms)
LArcNet	Raspberry PI 4B	0.2
LArcNet	STM32H743ZI2	3

C. TESTING WITH EXPANDED DATA CLASSES

The proposed LArcNet model was evaluated using an expanded set of load classes, including diverse electrical devices such as power drills, blow heaters, vacuum cleaners, electric kettles, jigsaws, food processors, computers, Dolce Gusto espresso machines, multifunctional printers, and pumps. This data was derived from the study cited in [35] and supplemented with additional data collected for this research. A total of 2,146 new samples (1,042 arc and 1,104 normal) were added to the database for training, testing, and evaluation. Additionally, 120 samples of transient condition data were tested using the proposed model. The evaluation results are summarized in Table 8. When tested with this comprehensive dataset, the model achieved an accuracy of 98.17%. Furthermore, under transient conditions with a limited sample set, the model successfully detected arc faults with an accuracy of 97.88%. These results underscore the robustness of LArcNet in handling a variety of operational scenarios and load conditions.

TABLE 8. Results of LArcNet model with expanded classes of data.

Data condition	Samples	Accuracy (%)
All classes	4942	98.17
Transient condition	120	97.88

D. RESULTS USING SUPPORT VECTOR MACHINE ALGORITHM

As a part of this study, a Support Vector Machine (SVM) with a radial basis function (RBF) kernel was employed to classify the dataset. Hyperparameter optimization was conducted using GridSearchCV, which systematically evaluates a predefined grid of hyperparameters to identify the best-performing configuration. The SVM model was trained and evaluated using 2-fold cross-validation, focusing on optimizing the C and gamma parameters. The results indicated that the optimal hyperparameters were C = 100 and gamma = 0.1, resulting in a cross-validation accuracy of 99.16%. When applied to an independent test set, the model achieved an accuracy of 99.20%, indicating that the model's performance on unseen data is consistent with the cross-validation results. These findings suggest that the SVM with an RBF kernel can also be well-suited for the binary classification task. The learning curve as shown in Fig. 10 demonstrates that the SVM model is not overfitted with these hyperparameter settings. On Raspberry PI 4B model the SVM model takes around 15 ms (see Table 3) to infer a sample. In terms of inference time it is higher than the proposed LArcNet model.

VI. COMPARISON OF ARC FAULT DETECTION METHODS

This section provides a comprehensive comparison of LArcNet with other recent methodologies in AC arc fault detection, as shown in Table 9. Reference [12] initially reported the longest runtime among the compared techniques at 31 ms per sample due to a lack of optimization. This method was optimized in this study, reducing its runtime significantly to 2.64 ms, demonstrating potential efficiency gains through optimization. Other studies, such as those in References [15] and [38], reported good arc fault detection accuracies (98.70% & 94.30%, respectively). However, they do not provide comprehensive implementation details on low-end commercial MCUs which is crucial for practical applications. SAFNet [27] employs a 2D input that increases

TABLE 9. Comparison of attributes with prior methods. The best results are boldfaced.

Model	Samples Tested	Accuracy (%)	Trainable Params	kFLOPs	Sampling Rate (kHz)	Runtime (ms)	Implementation Device
SAFNet [27]	4950	99.44	593.32 k	12202.24	2.5	26.48	Jetson Nano
ArcNet [12]	4513	**99.47**	189.64 k	18209.5	10	31	Raspberry PI 3B
RF-DNN [36]	1759	97.5	-	-	10	18.95	TMS320F28335
HTFNN [14]	1000	99.00	-	-	-	3	STM32F407ZG
FCNN [37]	-	98.05	-	-	5	6.38	Jetson Nano
LightGBM [11]	2400	97.06	-	-	-	300	Jetson Nano
LArcNet (Ours)	4513	99.31	**8.4 k**	**182.27**	10	**0.20** **3.00**	Raspberry PI 4B STM32H743ZI2

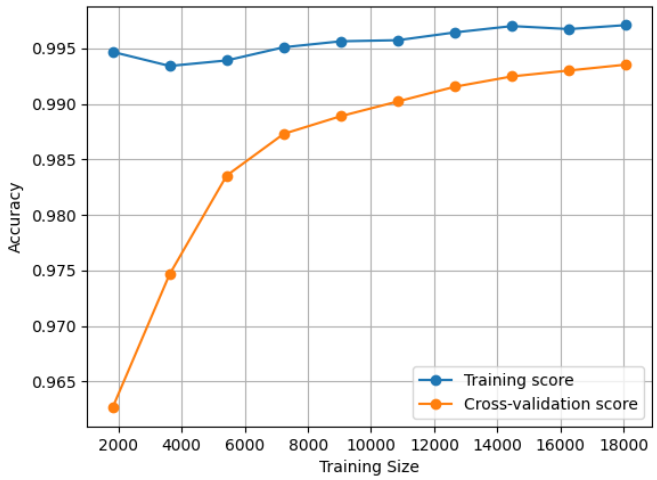


FIGURE 10. Learning curve of SVM model to check overfitting.

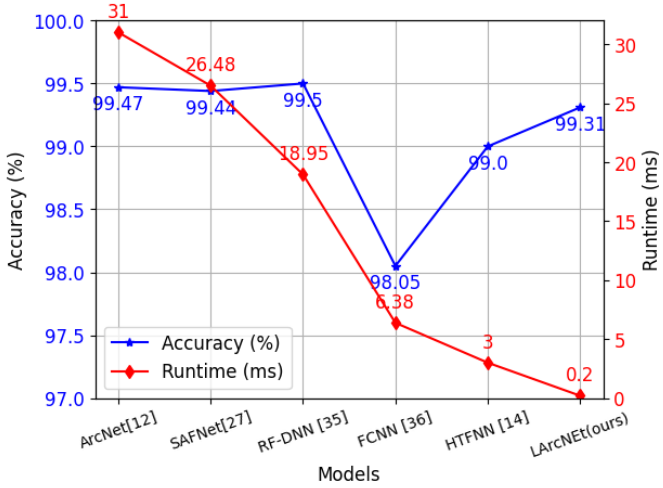


FIGURE 11. Comparison of accuracy & runtime of different models.

the computational load, resulting in a slower inference time of 26.48 ms per sample compared to LArcNet. It also takes more than 2.26 MB memory space which is unsuitable for low-end MCUs. Additionally, SAFNet's testing on the Jetson Nano, which has superior GPU capabilities compared to the Raspberry Pi 4, focuses on a single load type, limiting its applicability compared to LArcNet's multi-load capacity. Reference [11] proposed a method with lower accuracy and a higher computational burden when implemented on a similar platform as LArcNet.

In contrast, LArcNet not only identifies various load groups but also detects arc faults across four major load groups with an accuracy of 99.31%. It achieves a remarkably low runtime of only 0.2 ms per sample. This efficiency results from LArcNet's triple simplification approach: architectural optimization to reduce computational load, model compression via knowledge distillation to enhance performance, and Lite-RT optimization to decrease binary size and latency. This comprehensive strategy makes LArcNet highly suitable for real-time applications in resource-constrained environments. Contemporary arc fault detection methods are also compared in Fig. 11. Overall, LArcNet stands out as a leading solution in arc fault detection, offering a good balance between high accuracy and low computational demand, making it ideal for practical implementation in edge devices.

VII. CONCLUSIONS

In this study, an innovative and ultra-fast algorithm for high-performance arc fault detection, LArcNet, was proposed. LArcNet combines an efficient network architecture with advanced model compression techniques. Using a teacher-student knowledge distillation approach, it processes raw current inputs to detect arc faults with minimal computational complexity and high efficiency. Its lightweight yet robust structure achieves arc fault detection accuracies comparable to traditional models like Baseline CNN and Mobile-ArcNet. The model attains a load classification accuracy of 98.85% and an arc fault detection accuracy of 99.31%. While traditional CNN-based models may offer similar accuracy, they do so at a significantly higher computational cost compared to LArcNet.

Converting LArcNet into a TensorFlow Lite model significantly reduced its binary size and latency, making it ideal for real-time applications. When tested on a Raspberry Pi 4B and an STM32 MCU, the optimized LArcNet model

demonstrated inference times of just 0.20 ms and 3 ms per sample, respectively. This efficiency underscores LArcNet's suitability for commercial deployment and its effectiveness in detecting series AC arc faults. Additionally, the model can identify specific load types associated with arc faults. With a low memory footprint and minimal computational demands, LArcNet is well-suited for implementation on cost-effective commercial microcontrollers, providing an economical solution for arc fault detection.

Overall, LArcNet achieves high accuracy with significantly reduced inference times, making it highly suitable for resource-constrained embedded systems. Future work will focus on enhancing LArcNet's adaptability to handle unknown load scenarios, further increasing its robustness and applicability across a wider range of real-world conditions.

REFERENCES

[1] T. Covington, "House FIre Statistics and Facts in 2020," Apr 2021. [Online]. Available: https://tinyurl.com/bdzne7rt

[2] I. E. Commission *et al.*, "General requirements for arc fault detection devices; IEC 62606," *International Electrotechnical Commission: Geneva, Switzerland*, 2017.

[3] UL1699, "Standard for Safety Arc Fault Circuit-Interrupter," Underwriters Laboratories Inc., 2017.

[4] NEC, "National Electrical Code Handbook, 2014 ed." *National Fire Protection Association*, 2014.

[5] G. Artale, A. Cataliotti, V. Cosentino, and G. Privitera, "Experimental characterization of series arc faults in ac and dc electrical circuits," in *2014 IEEE International Instrumentation and Measurement Technology Conference (I2MTC) Proceedings*. IEEE, 2014, pp. 1015–1020.

[6] Y. Wang, F. Zhang, S. Zhang, and G. Yang, "A novel diagnostic algorithm for ac series arcing based on correlation analysis of high-frequency component of wavelet," *COMPEL-The international journal for computation and mathematics in electrical and electronic engineering*, 2017.

[7] D. O. Anggriawan, A. E. Rheinanda, M. K. Khafidli, E. Prasetyono, and N. A. Windarko, "Series arc fault breaker in low voltage using microcontroller based on fast fourier transform," *EMITTER International Journal of Engineering Technology*, vol. 9, no. 2, pp. 239–251, 2021.

[8] P. Qi, S. Jovanovic, J. Lezama, and P. Schweitzer, "Discrete wavelet transform optimal parameters estimation for arc fault detection in low-voltage residential power networks," *Electric power systems research*, vol. 143, pp. 130–139, 2017.

[9] M. K. Khafidli, E. Prasetyono, D. O. Anggriawan, A. Tjahjono, and M. H. R. A. Syafii, "Implementation AC series arc fault recognition using mikrokontroller based on fast Fourier transform," in *2018 International Electronics Symposium on Engineering Technology and Applications (IES-ETA)*. IEEE, 2018, pp. 31–36.

[10] W. Chi-Jui, C. Yung-Sung, C. Hui-Hsiang, and S. Wen-Yuan, "Investigation and Test of Arc Fault Circuit Interrupter Applied to Electric Power Circuits and Devices in Taiwan," *Journal of Occupational Safety and Health*, vol. 20, no. 1, pp. 116–124, 2012.

[11] Y. Meng, Q. Yang, S. Chen, Q. Wang, and X. Li, "Multi-branch ac arc fault detection based on iceemdan and lightgbm algorithm," *Electric Power Systems Research*, vol. 220, p. 109286, 2023.

[12] Y. Wang, L. Hou, K. C. Paul, Y. Ban, C. Chen, and T. Zhao, "ArcNet: Series AC Arc Fault Detection Based on Raw Current and Convolutional Neural Network," *IEEE Transactions on Industrial Informatics*, 2021.

[13] K. C. Paul, T. Zhao, C. Chen, Y. Ban, and Y. Wang, "Efficient-ArcNet: Series AC Arc Fault Detection using Lightweight Convolutional Neural Network," in *2021 IEEE Energy Conversion Congress and Exposition (ECCE)*. IEEE, 2021, pp. 1327–1333.

[14] Y. Wang, F. Zhang, X. Zhang, and S. Zhang, "Series AC Arc Fault Detection Method Based on Hybrid Time and Frequency Analysis and Fully Connected Neural Network," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 12, pp. 6210–6219, 2018.

[15] W. Li, Y. Liu, Y. Li, and F. Guo, "Series Arc Fault Diagnosis and Line Selection Method Based on Recurrent Neural Network," *IEEE Access*, vol. 8, pp. 177 815–177 822, 2020.

[16] J. Jiang, W. Li, Z. Wen, Y. Bie, H. Schwarz, and C. Zhang, "Series Arc Fault Detection Based on Random Forest and Deep Neural Network," *IEEE Sensors Journal*, 2021.

[17] M. A. Abdulrachman, E. Prasetyono, D. O. Anggriawan, and A. Tjahjono, "Smart detection of AC series arc fault on home voltage line based on fast Fourier transform and artificial neural network," in *2019 International Electronics Symposium (IES)*. IEEE, 2019, pp. 439–445.

[18] J. E. Siegel, S. Pratt, Y. Sun, and S. E. Sarma, "Real-time Deep Neural Networks for internet-enabled arc-fault detection," *Engineering Applications of Artificial Intelligence*, vol. 74, pp. 35–42, 2018.

[19] X. Han, D. Li, L. Huang, H. Huang, J. Yang, Y. Zhang, X. Wu, and Q. Lu, "Series Arc Fault Detection Method Based on Category Recognition and Artificial Neural Network," *Electronics*, vol. 9, no. 9, p. 1367, 2020.

[20] N. Qu, J. Chen, J. Zuo, and J. Liu, "PSO-SOM neural network algorithm for series arc fault detection," *Advances in Mathematical Physics*, vol. 2020, 2020.

[21] N. Qu, J. Zuo, J. Chen, and Z. Li, "Series arc fault detection of indoor power distribution system based on LVQ-NN and PSO-SVM," *IEEE Access*, vol. 7, pp. 184 020–184 028, 2019.

[22] K. C. Paul, L. Schweizer, T. Zhao, C. Chen, and Y. Wang, "Series AC Arc Fault Detection Using Decision Tree-Based Machine Learning Algorithm and Raw Current," in *2022 IEEE Energy Conversion Congress and Exposition (ECCE)*, 2022, pp. 1–8.

[23] X. Ning, D. Sheng, J. Zhou, Y. Liu, Y. Wang, H. Zhang, and X. Lei, "Arc_effnet: A novel series arc fault detection method based on lightweight neural network," *Electronics*, vol. 12, no. 22, p. 4617, 2023.

[24] Y. Liu, G. Yang, and H. Wang, "Series arc fault detection under vibration condition based on nmmb," *Sensors*, vol. 24, no. 3, p. 959, 2024.

[25] J. Jiang, Z. Wen, M. Zhao, Y. Bie, C. Li, M. Tan, and C. Zhang, "Series Arc Detection and Complex Load Recognition Based on Principal Component Analysis and Support Vector Machine," *IEEE Access*, vol. 7, pp. 47 221–47 229, 2019.

[26] R. Jiang and Y. Zheng, "Series arc fault detection using regular signals and time-series reconstruction," *IEEE Transactions on Industrial Electronics*, vol. 70, no. 2, pp. 2026–2036, 2022.

[27] Z. Wang, S. Tian, H. Gao, C. Han, and F. Guo, "An on-line detection method and device of series arc fault based on lightweight cnn," *IEEE Transactions on Industrial Informatics*, 2023.

[28] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6848–6856.

[29] 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," *arXiv preprint arXiv:1704.04861*, 2017.

[30] I. Freeman, L. Roese-Koerner, and A. Kummert, "Effnet: An efficient structure for convolutional neural networks," in *2018 25th IEEE International Conference on Image Processing (ICIP)*. IEEE, 2018, pp. 6–10.

[31] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[32] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: A survey," *International Journal of Computer Vision*, vol. 129, no. 6, pp. 1789–1819, 2021.

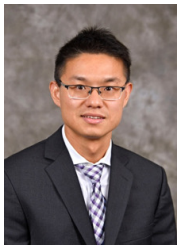
[33] Wikipedia, "Knowledge distillation," Jan 2023. [Online]. Available: https://en.wikipedia.org/wiki/Knowledge_distillation

[34] Google, "Introducing LiteRT: Google's high-performance runtime for on-device AI, formerly known as TensorFlow Lite." [Online]. Available: https://ai.google.dev/edge/litert/models/convert_tf

[35] F. Ferracuti, P. Schweitzer, and A. Monteriu, "Arc fault detection and appliances classification in AC home electrical networks using recurrence quantification plots and image analysis," *Electric Power Systems Research*, vol. 201, p. 107503, 2021.

[36] J. Jiang, W. Li, Z. Wen, Y. Bie, H. Schwarz, and C. Zhang, "Series arc fault detection based on random forest and deep neural network," *IEEE Sensors Journal*, vol. 21, no. 15, pp. 17 171–17 179, 2021.
[37] A. Tang, Z. Wang, S. Tian, H. Gao, Y. Gao, and F. Guo, "Series Arc Fault Identification Method Based on Lightweight Convolutional Neural Network," *IEEE Access*, 2024.
[38] Y. Wang, F. Zhang, and S. Zhang, "A new methodology for identifying arc fault by sparse representation and neural network," *IEEE Transactions on Instrumentation and Measurement*, vol. 67, no. 11, pp. 2526–2537, 2018.

KAMAL CHANDRA PAUL(GS'21) is currently a postdoctoral fellow at the Department of Electrical and Computer Engineering, University of North Carolina at Charlotte (UNCC), USA, where he completed his PhD in 2024. He received his bachelor's degree in Electrical and Electronic Engineering (EEE) from Khulna University of Engineering and Technology (KUET) in 2009 and obtained his MS degree in Electrical Engineering from the Ingram School of Engineering, Texas State University, USA in 2018. He previously served as a faculty member at the Department of EEE, International University of Business Agriculture and Technology (IUBAT), and as a lecturer in the Department of EEE at World University of Bangladesh. His research interests encompass AC and DC arc fault protection, battery fault diagnostics, data science, and the application of machine learning in power electronics and renewable energy systems.



CHEN CHEN(M'18) is an assistant professor at the Center for Research in Computer Vision, University of Central Florida. He received the Ph.D. degree from the Department of Electrical Engineering, University of Texas at Dallas in 2016 where he received the David Daniel Fellowship (Best Doctoral Dissertation Award). His research interests include computer vision, efficient deep learning, and federated learning. He is an Associate Editor of IEEE Transactions on Circuits and Systems for Video Technology (T-CSVT), Journal of Real-Time Image Processing, and IEEE Journal on Miniaturization for Air and Space Systems.

YAO WANG(M'16) was born in Hebei, China, in 1981. He received the B.S., M.S., and Ph.D. degrees in electrical engineering from the Hebei University of Technology, Tianjin, China, in 2006, 2009, and 2012, respectively. He is currently an Associate Professor with the State Key Laboratory of Reliability and Intelligence of Electrical Equipment, School of Electrical Engineering, Hebei University of Technology. He has authored or coauthored more than ten technical articles. His current research interests include intellectualization of electrical apparatus and fault diagnosis of electrical apparatus.

TIEFU ZHAO(S'06–M'10–SM'12) received the B.S. and M.S. degrees from Tsinghua University, Beijing, China, in 2003 and 2005, respectively, and the Ph.D. degree from North Carolina State University, Raleigh, in 2010, all in electrical engineering.

From 2010 to 2016, he was with Eaton Corporation Research & Technology, Milwaukee, WI. Since 2016, he has been an Assistant Professor of Electrical and Computer Engineering and an Associate of the Energy Production and Infrastructure Center (EPIC) with the University of North Carolina at Charlotte, Charlotte, NC, USA. He has published over 40 papers in refereed journals and international conference proceedings. He has 12 patents awarded. His current research interests include solid state transformer, solid state circuit protection, wireless power transfer, wide bandgap device based power conversion, power electronics reliability and fault detection. Dr. Zhao has served as an Associate Editor for the IEEE Journal of Emerging and Selected Topics in Power Electronics (JESTPE).