Journal of Information Hiding and Multimedia Signal Processing ©2025 ISSN 2073-4212
Ubiquitous International Volume **16**, Number **4**, December **2025**

BIMask Learning: A Saliency-Guided Masked Training

Ahmed Elshazly
Department of Computer Engineering and Artificial Intelligence
Military Technical College
Cairo, Egypt
ahmedelshazly@mtc.edu.eg

Ahmed Elliethy
Department of Computer Engineering and Artificial Intelligence
Military Technical College
Cairo, Egypt
a.s.elliethy@mtc.edu.eg

Mohamed A. Elshafey
Department of Computer Engineering and Artificial Intelligence
Military Technical College
Cairo, Egypt
m.shafey@mtc.edu.eg

Received September 26, 2025, revised October 20, 2025, accepted October 22, 2025.

Abstract. *Deep learning models often require increased size and complexity to achieve high performance, which presents significant challenges for deployment on resource-constrained embedded systems. Traditional model optimization techniques, such as pruning and quantization, reduce model size or parameter count but frequently result in reduced accuracy. In this paper, we propose BIMask Learning, a novel training optimization technique. Our approach begins by generating saliency maps from a pre-trained model to identify high-importance regions within the input images. Then, based on a predefined masking ratio, we construct binary masks that preserve these salient regions. These masks are applied to the original dataset to create a masked version, which is then combined with the original data to form a mixed training set. Finally, we fine-tune the model on this enriched dataset, which encourages the model to focus on discriminative features. We evaluate our method on CIFAR-10 and CIFAR-100 using VGG and ResNet architectures. The results demonstrate consistent accuracy improvements, with average gains of 0.90% on CIFAR-10 and 0.76% on CIFAR-100, all without increasing the model size or parameter count.*
Keywords: BIMask, Curriculum, Learning, Optimization, Deep Learning

1. Introduction.

Deep learning models are becoming increasingly powerful, but this also raises their computational demands. Deploying such models on embedded platforms—such as satellites, mobile phones, and smart cars—has gained importance, yet these platforms are constrained by limited processing power, memory, and energy, making deployment challenging [1, 2].

Optimization techniques address this challenge, including pruning [3], quantization [4, 5], Knowledge Distillation (KD) [6, 7], and Curriculum Learning (CL) [8, 9]. Quantization and pruning reduce model size and computation at the cost of accuracy. KD improves accuracy without increasing complexity but requires a large pretrained teacher model. CL enhances learning by gradually increasing task difficulty. Each method has trade-offs, which will be discussed further in the related work section.

This paper introduces BIMask Learning, a training optimization technique that boosts accuracy without changing model architecture or parameters. Instead, it improves training data by reducing background

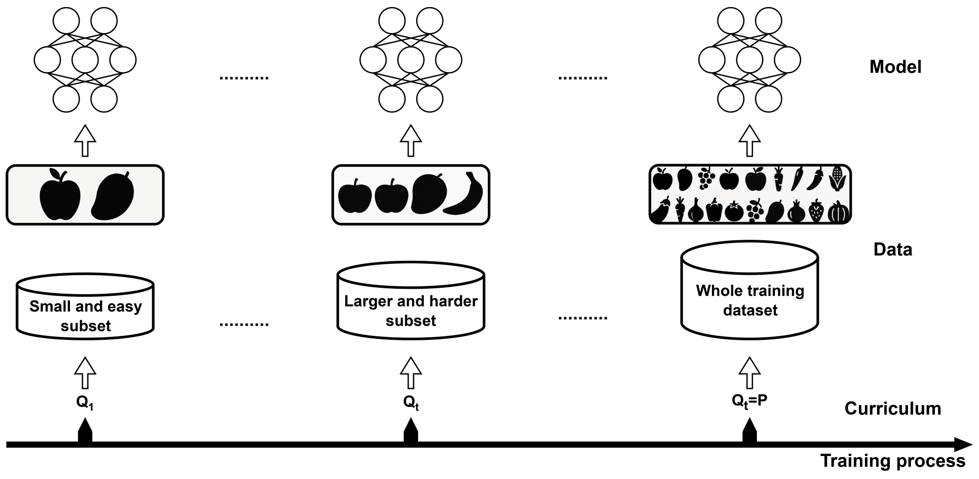

Figure 1. Curriculum Learning (CL) outlines progressive data introduction: from easy subsets (Q_1) to the full training dataset (P), where Q_t is the subset at time t.

distractors. The process begins with standard training, followed by generating saliency maps [10] to highlight key regions. Using a masking ratio, binary masks preserve high-saliency areas, which are applied to create a masked dataset. The model is then fine-tuned on both original and masked datasets, encouraging it to focus on essential object features while remaining invariant to background noise. BIMask Learning thus enhances accuracy without increasing model size or complexity.

The rest of this paper is structured as follows: Related Work reviews prior studies, Methodology introduces the BIMask Learning paradigm, Experiments describe the setup and results, Discussion provides analysis, and Conclusion presents final remarks.

2. Related Work.

In this section, we review major software optimization techniques: quantization [4, 5], pruning [3], Knowledge Distillation (KD) [6, 7], and Curriculum Learning (CL) [8].

Table 1. Comparison of model learning optimization techniques' advantages and disadvantages.

Technique	Advantages	Disadvantages
Quantization	Reduced memory, faster computation, lower power consumption	Accuracy loss, compatibility issues, fine-tuning needed
Pruning	Smaller model, faster computation, lower power consumption	Accuracy loss, over-pruning risk, training complexity
Knowledge Distillation	Improved accuracy, no model size increase, no extra parameters	Requires teacher model, longer training, higher cost
Curriculum Learning	Improved accuracy, no model size increase, no extra parameters	Design challenges, training complexity, performance variation

Quantization reduces weight and activation precision from 32-bit floating-point to lower bit formats (e.g., 8-bit), leading to smaller memory usage, faster inference, and lower power consumption—critical for embedded systems [4, 5]. However, it often reduces accuracy and may face hardware compatibility issues. The two main types are Post-Training Quantization (PTQ) [11], where a pretrained model is quantized and fine-tuned, and Quantization-Aware Training (QAT) [12], where quantization is incorporated during training [13].

Pruning [3] removes redundant weights to reduce computations and power at the cost of some accuracy. Structured pruning removes whole patterns such as neurons or channels [14], while unstructured pruning removes individual weights, offering more flexibility [15].

Knowledge Distillation (KD) [6, 7] improves accuracy without increasing model size by training a smaller "student" under supervision of a larger "teacher." Methods include Response-Based KD (RBKD), aligning student outputs with the teacher's, and Feature-Based KD (FBKD), matching internal representations [16]. While effective, KD depends on the teacher model and adds training complexity. Prominent approaches include FitNets [17], Attention Transfer (AT) [18], Factor Transfer (FT) [19], and Supervised Contrastive KD (SCKD) [20].

The training optimization techniques aim to improve the model's accuracy without increasing its size or its number of parameters. Unlike KD techniques, these techniques do not require a larger pretrained teacher to assist the student in enhancing its accuracy. One of these techniques is CL [8] technique, where the training procedure is organized in a way that the model begins by learning from simpler instances and gradually progresses to more complex ones, as shown in Fig. 1. However, one drawback of CL technique is that designing an effective curriculum and selecting suitable training samples can be challenging, especially with large datasets. Table 1 shows a comparison between all the discussed optimization techniques in terms of their advantages and disadvantages.

We compare our proposed technique, BIMask Learning, with several recent curriculum learning (CL) approaches. These include Curriculum by Smoothing (CBS) [21], which involves three stages: smoothing the training data with Gaussian filters, gradually reducing the smoothing to increase complexity, and applying task-specific loss functions to maintain training stability. EfficientTrain [22] begins with simpler tasks and progressively increases difficulty using more complex patterns while optimizing resources through adaptive sample selection and dynamic training schedules for faster convergence. Self-taught Learning [23] starts with pretraining on a large unlabeled dataset, followed by fine-tuning on a smaller labeled one, effectively leveraging both data types to enhance generalization. LCDnet-CL [24] integrates a transfer-scoring function to rank sample difficulty and utilizes a ResNet-50 backbone for efficient feature extraction. Finally, Learning Rate Curriculum (LeRaC) [23] adjusts learning rates across network layers—starting with higher rates for shallow layers and lower ones for deeper layers—gradually balancing them over time to improve learning stability without altering the data itself.

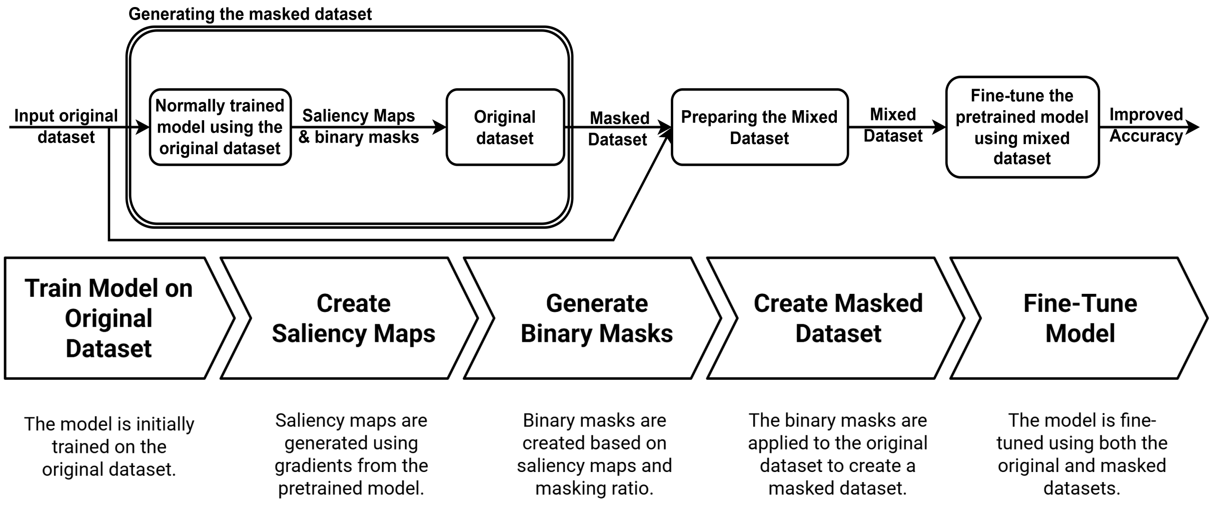
Generating the masked dataset
Input original dataset
Normally trained model using the original dataset
Saliency Maps & binary masks
Original dataset
Masked Dataset
Preparing the Mixed Dataset
Mixed Dataset
Fine-tune the pretrained model using mixed dataset
Improved Accuracy
Train Model on Original Dataset
Create Saliency Maps
Generate Binary Masks
Create Masked Dataset
Fine-Tune Model
The model is initially trained on the original dataset.
Saliency maps are generated using gradients from the pretrained model.
Binary masks are created based on saliency maps and masking ratio.
The binary masks are applied to the original dataset to create a masked dataset.
The model is fine-tuned using both the original and masked datasets.

FIGURE 2. Framework of the proposed BIMask Learning technique.

3. Methodology.

Most of the optimization techniques focus on reducing model size or its number of parameters, such as pruning and quantization [2]. Unfortunately, these techniques can cause a significant decrease in the model's accuracy. There is also KD techniques [6, 7], which can increase the accuracy of the model named "student" without any increased model complexity. Unfortunately, KD requires a larger pretrained "teacher" model to guide the training of the smaller "student" model. It means that we can't perform KD if we don't have a larger pretrained model.

The proposed BIMask Learning methodology, illustrated in Figure 2, consists of three main stages: saliency maps generation, mask construction, and fine-tuning. First, saliency maps are generated from a pretrained model to highlight the most informative regions of each image. These maps are then converted into binary masks using a predefined threshold ratio, isolating the key object areas while suppressing less relevant background information. The resulting masks are then applied to the original dataset to create a masked version of the training data. In the final stage, the model is fine-tuned using a balanced combination of original and masked samples. This dual-training strategy enables the network to concentrate on essential object features while still leveraging contextual background details, ultimately enhancing accuracy and generalization without increasing model complexity.

Algorithm 1: BIMask Learning

```
   Input: Pretrained model $\mathcal{M}$, Original dataset $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, Target layer $\mathcal{T}_f$,
          Mask ratio $\mu$, Number of epochs $E$, Learning rate $\alpha$, Batch size $B$
   Output: Fine-tuned model $\mathcal{M}_{ft}$
1  Step 1: Compute saliency maps
2  for each image $(\mathbf{x}_i, y_i) \in D$ do
3      Compute saliency map $S_i = \left|\frac{\partial \mathcal{M}(\mathbf{x}_i)}{\partial \mathbf{x}_i}\right|$;
4  Step 2: Create saliency-masked dataset
5  for each image $(\mathbf{x}_i, y_i) \in D$ do
6      Apply binary mask to $\mathbf{x}_i$ based on $S_i$ and mask ratio $\mu$;
7      Create masked image $\mathbf{x}_i^{mask} = \mathbf{x}_i \cdot \mathbf{M}_i$;
8  Step 3: Create mixed dataset
9  Combine original dataset $D$ and saliency-masked dataset $D^{mask}$ into a mixed dataset $D^{mixed}$;
10 Step 4: Fine-tune the model on mixed dataset and validate
11 for each epoch $e = 1$ to $E$ do
12     for each batch $(\mathbf{x}_j, y_j) \in D^{mixed}$ do
13         Perform forward pass through $\mathcal{M}$ and compute the loss:
                   $\mathcal{L}_{total} = \mathcal{L}_{CE}(\mathcal{M}(\mathbf{x}_j), y_j)$
           Perform backward pass and update model weights;
14     Evaluate the model on the validation set;
15     If validation accuracy improves, save the model;
```

As we can see in Fig. 2 and Algorithm 1, the process consists of three main blocks. The first block involves creating the masked dataset, starting with the generation of saliency maps [10]. These maps highlight the regions that are most important for the model's predictions. We begin by training the model on the original dataset, then use it to calculate the saliency scores for the same dataset. The saliency score, computed using Eq. 1, for each image $\mathbf{x}_i$ is computed based on the gradients of the trained model, specifically at the target layer. In particular, we compute the gradient of the predicted class score with respect to the feature map at the target layer $\mathcal{T}_f$:

$$S_i = \left|\frac{\partial \mathcal{M}(\mathbf{x}_i)}{\partial \mathbf{x}_i}\right|, \tag{1}$$

where:

- S_i represents the saliency score for the image $\mathbf{x}_i$.
- $\mathcal{M}(\mathbf{x}_i)$ is the model's output for the image $\mathbf{x}_i$.
- $\frac{\partial \mathcal{M}(\mathbf{x}_i)}{\partial \mathbf{x}_i}$ is the gradient of the output with respect to the input image $\mathbf{x}_i$, calculated at the target layer $\mathcal{T}_f$.

To choose the target layer for creating saliency maps, we need to ensure that it captures high-level features and exhibits high sensitivity to the input. This is typically found in the mid- to deeper layers of the network, particularly the convolutional layers. In our work, we always choose the target layer to be the last convolutional layer in the model, just before the pooling layer.

Then, we create a binary mask for each image $\mathbf{x}_i$ using its corresponding saliency map S_i and the given masked ratio μ. Finally, we create the masked dataset by applying the binary masks on the original dataset. The masked image $\mathbf{x}_i^{mask}$ is computed using Eq. 2:

$$\mathbf{x}_i^{mask} = \mathbf{x}_i \cdot \mathbf{M}_i, \tag{2}$$

where $\mathbf{M}_i = \text{binary\_mask}(S_i, \mu)$ is the binary mask keeping the salient parts of the image.

In the second block, we simply prepare the mixed dataset by combining the original dataset with the masked dataset, which is generated by applying each image with its corresponding binary mask. The mixed dataset provides more generalization because it consists of the masked dataset, which provides focused feature learning, and the original dataset, which provides contextual learning. In the final step we fine-tune the pretrained model using the mixed dataset. During fine-tuning, the model is trained using the cross-entropy loss on the mixed dataset. The total loss is computed using Eq. 3:

$$\mathcal{L}_{total} = \mathcal{L}_{CE}(\mathcal{M}(\mathbf{x}_j), y_j), \tag{3}$$

where $\mathcal{L}_{CE}$ is the cross-entropy loss between the model's prediction and the true label y_j. In the following section we will discuss the results of our experiments.

Table 2. This table compares the performance of the enhanced models using BIMask Learning against their baseline accuracies, tested on CIFAR-10 and CIFAR-100 datasets. The table shows the average accuracy rates (in %) along with the accuracy improvements (in parentheses). The final row shows the average accuracy gain.

Model	**CIFAR-10**		**CIFAR-100**	
	Baseline Acc.	Enhanced Acc.	Baseline Acc.	Enhanced Acc.
VGG8	91.6	92.52 (**0.92**)	70.36	70.78 (**0.42**)
VGG11	92.42	93.36 (**0.94**)	72.25	73.16 (**0.91**)
VGG13	93.90	94.81 (**0.91**)	74.64	76.25 (**1.61**)
ResNet14	91.51	92.42 (**0.91**)	68.25	68.66 (**0.36**)
ResNet20	92.39	93.32 (**0.93**)	69.06	69.52 (**0.46**)
ResNet56	93.71	94.63 (**0.92**)	73.03	73.45 (**0.42**)
ResNet18	89.20	90.12 (**0.92**)	71.70	73.08 (**1.38**)
Wide-ResNet-50	91.22	91.93 (**0.71**)	68.14	68.65 (**0.51**)
Average Gain	**0.90**		**0.76**	

Table 3. This table compares the performance of the fine-tuned models using masked datasets only against their baseline accuracies, tested on CIFAR-10 and CIFAR-100 datasets. The table shows the average accuracy rates (in %) along with the accuracy changes (in parentheses). The final row shows the average accuracy loss.

Model	**CIFAR-10**		**CIFAR-100**	
	Baseline Acc.	Enhanced Acc.	Baseline Acc.	Enhanced Acc.
VGG8	91.60	90.53 (**-1.07**)	70.36	69.23 (**-1.13**)
VGG11	92.42	91.25 (**-1.17**)	72.25	71.11 (**-1.14**)
VGG13	93.90	92.56 (**-1.34**)	74.64	73.23 (**-1.41**)
ResNet14	91.51	90.43 (**-1.08**)	68.25	67.02 (**-1.23**)
ResNet20	92.39	91.42 (**-0.97**)	69.06	68.01 (**-1.05**)
ResNet56	93.71	92.61 (**-1.10**)	73.03	71.95 (**-1.08**)
ResNet18	93.71	92.67 (**-1.04**)	73.03	71.82 (**-1.21**)
Wide-ResNet-50	91.22	90.11 (**-1.11**)	68.14	66.95 (**-1.19**)
Average Loss	**-1.11**		**-1.18**	

4. **Experiments.**

In this section, we present a series of experiments to show how effective BIMask Learning is, in different scenarios. We performed these experiments on various deep learning models from the VGG and ResNet families. From the VGG [25] family, we used VGG8, VGG11, and VGG13, and from the ResNet [26] family, we used ResNet14, ResNet20, ResNet56 and ResNet18.

We performed these experiments using CIFAR-10 and CIFAR-100 datasets [27]. Both are widely used benchmarks for image classification, each containing 60,000 color images of size 32×32 pixels. CIFAR-10 consists of 10 classes with 6,000 images per class, while CIFAR-100 includes 100 classes with 600 images each, offering a more challenging task due to its higher class diversity and finer-grained categories. In

Table 4. This table compares the performance of BIMask Learning against different CL techniques, tested on CIFAR-10 and CIFAR-100 datasets. The experiments are conducted using ResNet-18 and Wide-ResNet-50 architectures. The table shows the average accuracy rates (in %) along with the accuracy improvements (in parentheses) for each technique, with the highest value bolded and the second highest underlined.

Model	(CL) technique	**CIFAR-10**	**CIFAR-100**
ResNet18	Baseline accuracy	89.20	71.70
	CBS [21]	89.53 (0.33)	72.80 (1.10)
	EfficientTrain [22]	89.51 (0.31)	72.83 (1.13)
	Self-taught [23]	89.48 (0.28)	72.10 (0.40)
	LCDnet-CL [24]	89.36 (0.16)	71.06 (-0.64)
	LeRaC [23]	89.56 (0.36)	72.72 (1.02)
	BIMask (ours)	**90.12 (0.92)**	**73.08 (1.38)**
Wide-ResNet-50	Baseline accuracy	91.22	68.14
	CBS [21]	89.05 (-2.17)	65.73 (-2.41)
	EfficientTrain [22]	91.03 (-0.19)	69.14 (1.00)
	Self-taught [23]	91.00 (-0.22)	68.48 (0.34)
	LCDnet-CL [24]	91.38 (0.16)	68.85 (0.71)
	LeRaC [23]	91.58 (0.36)	**69.38 (1.24)**
	BIMask (ours)	**91.93 (0.71)**	68.65 (0.51)

Table 5. This table compares the performance of different KD techniques on the CIFAR-10 and CIFAR-100 datasets. The student model (ResNet20) is trained using either a baseline teacher (ResNet56) or an enhanced teacher with BIMask Learning. The last row show the student's baseline accuracy. Results include average accuracy (%) and accuracy gains (in parentheses).

Method	**CIFAR-10**		**CIFAR-100**	
	Teacher Acc. (93.71)	Enhanced teacher (94.63)	Teacher Acc. (73.03)	Enhanced teacher (73.45)
KD [6]	92.91 (0.52)	93.05 (0.66)	70.66 (1.60)	70.78 (1.72)
FitNets [17]	92.65 (0.26)	92.76 (0.37)	69.21 (0.15)	69.34 (0.28)
AT [18]	93.01 (0.62)	93.08 (0.69)	70.55 (1.49)	70.67 (1.61)
FT [19]	92.85 (0.46)	92.96 (0.57)	69.84 (0.78)	69.95 (0.89)
SCKD [20]	93.32 (0.93)	93.41 (1.02)	71.45 (2.39)	71.58 (2.52)
Baseline accuracy	92.39		69.06	

Table 6. This table compares the enhanced accuracies of VGG13 on CIFAR-10 and CIFAR-100 datasets targeing individual layer of the last four convolutional layers. The table shows the average accuracy rates (in %) along with the accuracy changes (in parentheses), with the highest value bolded. The last row shows the baseline accuracy of the VGG13.

Target Layer	**CIFAR-10**	**CIFAR-100**
Fourth-to-Last	94.51 (0.61)	75.12 (0.48)
Third-to-Last	94.63 (0.73)	75.43 (0.79)
Second-to-Last	94.72 (0.82)	75.89 (1.25)
Last	**94.81 (0.91)**	**76.25 (1.61)**
Baseline accuracy	93.90	74.64

Table 7. This table compares the enhanced accuracies of VGG8 on CIFAR-10 and CIFAR-100 datasets across different masking ratios. The table shows the average accuracy rates (in %) along with the accuracy changes (in parentheses), with the highest value bolded. The last row shows the baseline accuracy of the VGG8.

Masking Ratio	**CIFAR-10**	**CIFAR-100**
5%	92.21 (0.61)	70.52 (0.16)
7.5%	92.35 (0.75)	70.68 (0.32)
10%	**92.52 (0.92)**	**70.78 (0.42)**
15%	92.26 (0.66)	70.51 (0.15)
20%	92.08 (0.48)	70.24 (-0.12)
30%	91.92 (0.32)	70.01 (-0.35)
Baseline accuracy	91.6	70.36

BIMask Learning, these datasets are used to generate masked versions of the images, which are then combined with the original data to form a mixed dataset for model fine-tuning.

All the results in the tables represent the average over 5 runs. In all the experiments, we used the last convolutional layer as the target layer. We also used a masking ratio of 7.5% and 10% for ResNet and VGG models, respectively.

In the first experiment shown in Table 2, we compare the accuracies of the enhanced models, using BIMask Learning, against their baseline accuracies. The proposed BIMask Learning method boosts accuracy by an average of 0.90% on CIFAR-10 and, importantly, sustains this improvement on the larger CIFAR-100 dataset with a 0.76% average gain, demonstrating its scalability and consistent effectiveness. While the improvement is more modest than on CIFAR-10, its consistent performance on large-scale dataset underscores the method's robust generalization to more complex, real-world visual recognition tasks.

In the second experiment, we use the masked dataset only to fine-tune the pretrained models. As we can see in Table 3, in the case of both CIFAR-10 and CIFAR-100 datasets, the accuracies of the models decrease due to the lack of contextual information in the masked dataset. This shows the importance of keeping the contextual information provided by the original dataset.

In the third experiment, we compared the results of BIMask Learning against several CL techniques, including CBS [21], EfficientTrain [22], Self-taught learning [23], LCDnet-CL [24], and LeRaC [23].As shown in Table 4, our proposed technique outperforms most of these state-of-the-art methods in nearly all cases, except for Wide-ResNet-50 on CIFAR-100. The results of the other techniques are taken from the paper on the Learning Rate Curriculum (LeRaC) technique [23].

In the fourth experiment, we applied BIMask Learning to enhance the accuracy of ResNet56 and used it as the teacher for a ResNet20 student in various knowledge distillation methods. As shown in Table 5, using the enhanced ResNet56 teacher resulted in higher student accuracy compared to using the baseline teacher.

To examine how target layer depth affects accuracy improvement, we used the VGG13 model—the deepest among the tested VGG variants—with a fixed masking ratio of 10%. As shown in Table 6, BIMask Learning was applied separately to the last four convolutional blocks. The results indicate that deeper target layers lead to greater accuracy gains. This improvement occurs because deeper layers capture higher-level semantic features, producing more discriminative saliency maps that better identify and preserve important image regions during masking.

To investigate the impact of masking ratios on accuracy enhancement, we selected VGG8, as it is the smallest model among all the VGG models, enabling faster execution of multiple experiments with different masking ratio values. We performed this experiment using the last convolutional layer as the target layer for BIMask Learning. In Table 7, we compare the enhanced accuracies of VGG8 across different masking ratio values. As shown, the best accuracy is achieved with a masking ratio of 10%.

For the implementation details and access to the code, please check our GitHub repository: https://github.com/AhmedElshazly19/BIMask-Learning.

5. Discussion.

This section describes how the proposed BIMask Learning enhances accuracy without increasing model complexity. Saliency maps are generated from a pretrained model to create binary masks, which are applied to form a masked dataset. The model is then fine-tuned using a mix of original and masked data, allowing it to focus on key object features while suppressing background noise. Results show consistent accuracy gains across VGG and ResNet models, with deeper networks (e.g., ResNet18) benefiting most from richer saliency maps. However, using only masked data for training reduces accuracy because the model lacks the necessary background context to generalize effectively.

However, the proposed BIMask Learning method also has some limitations. The primary limitation is the increased computational cost, as the process requires calculating saliency maps and creating a masked dataset for the entire training set. Additionally, the effectiveness of the proposed BIMask Learning method is highly dependent on the model architecture. It yields minimal improvements for shallow models because the technique relies on the rich, high-level feature representations that are only adequately developed in deeper networks to generate meaningful and accurate saliency guidance.

6. **Conclusion.** In this paper, we proposed BIMask Learning, a training optimization technique. In this technique, we use the saliency maps calculated at a selected layer to generate binary masks. Then we used these binary masks to create the masked dataset. Finally, we fine-tuned the model using a mix of both the original and the masked datasets.

In experimental work, the proposed BIMask Learning technique is approved to increase the accuracy of the model without increasing its size or its number of parameters. The increase is due to the mixed dataset, which incorporates both contextual information and relevant object features. We tested and verified our proposed technique using VGG and ResNet models on CIFAR-10 and CIFAR-100 datasets.

REFERENCES

[1] A. Elshazly, A. Elliethy, M. A. Elshafey. Tactics overview for implementing high-performance computing on embedded platforms. In *IOP Conference Series: Materials Science and Engineering*, volume 1172, number 1, page 012034, 2021.

[2] Chellammal Surianarayanan, John Jeyasekaran Lawrence, Pethuru Raj Chelliah, Edmond Prakash, Chaminda Hewage. A survey on optimization techniques for edge artificial intelligence (AI). *Sensors*, 23(3):1279, 2023.

[3] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, Trevor Darrell. Rethinking the value of network pruning. *arXiv preprint arXiv:1810.05270*, 2018.

[4] Dwith Chenna. Edge AI: quantization as the key to on-device smartness. *Jour. ID*, 4867:9994, 2023.

[5] Chun Yang, Ruiyao Zhang, Long Huang, Shutong Ti, Jinhui Lin, Zhiwei Dong, Songlu Chen, Yan Liu, Xucheng Yin. A survey of quantization methods for deep neural networks. *Chinese Journal of Engineering*, 45(10):1613–1629, 2023.

[6] Geoffrey Hinton, Oriol Vinyals, Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[7] Jianping Gou, Baosheng Yu, Stephen J. Maybank, Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129:1789–1819, 2021.

[8] Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E. Taylor, Peter Stone. Curriculum learning for reinforcement learning domains: A framework and survey. *Journal of Machine Learning Research*, 21(181):1–50, 2020.

[9] Xin Wang, Yudong Chen, Wenwu Zhu. A survey on curriculum learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):4555–4576, 2021.

[10] Ahmed Alqaraawi, Martin Schuessler, Philipp Weiss, Enrico Costanza, Nadia Berthouze. Evaluating saliency map explanations for convolutional neural networks: a user study. In *Proceedings of the 25th International Conference on Intelligent User Interfaces*, 2020, pages 275–285.

[11] Zhenhua Liu, Yunhe Wang, Kai Han, Wei Zhang, Siwei Ma, Wen Gao. Post-training quantization for vision transformer. *Advances in Neural Information Processing Systems*, 34:28092–28103, 2021.

[12] Mengzhao Chen, Wenqi Shao, Peng Xu, Jiahao Wang, Peng Gao, Kaipeng Zhang, Ping Luo. Efficientqat: Efficient quantization-aware training for large language models. *arXiv preprint arXiv:2407.11062*, 2024.

[13] Xiaotian Zhao, Ruge Xu, Xinfei Guo. Post-training quantization or quantization-aware training? That is the question. In *2023 China Semiconductor Technology International Conference (CSTIC)*, pages 1–3, 2023.

[14] Dongxian Wu, Yisen Wang. Adversarial neuron pruning purifies backdoored deep models. *Advances in Neural Information Processing Systems*, 34:16913–16925, 2021.

[15] Xiaolong Ma, Sheng Lin, Shaokai Ye, Zhezhi He, Linfeng Zhang, Geng Yuan, Sia Huat Tan, Zhengang Li, Deliang Fan, Xuehai Qian, et al. Non-structured DNN weight pruning—Is it beneficial in any platform? *IEEE Transactions on Neural Networks and Learning Systems*, 33(9):4930–4944, 2021.

[16] Chuanguang Yang, Xinqiang Yu, Zhulin An, Yongjun Xu. Categories of Response-Based, Feature-Based, and Relation-Based Knowledge Distillation. In *Advancements in Knowledge Distillation: Towards New Horizons of Intelligent Systems*, pages 1–32, 2023.

[17] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, Yoshua Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014.

[18] Nikos Komodakis, Sergey Zagoruyko. Paying more attention to attention: improving the performance of convolutional neural networks via attention transfer. In *ICLR*, 2017.

[19] Jangho Kim, SeongUk Park, Nojun Kwak. Paraphrasing complex network: Network compression via factor transfer. *Advances in Neural Information Processing Systems*, 31, 2018.

[20] Ahmed Elshazly, Ahmed Elliethy, Mohamed A. Elshafey. Supervised Contrastive Knowledge Distillation. *International Journal of Intelligent Engineering & Systems*, 17(6), 2024.

[21] Samarth Sinha, Animesh Garg, Hugo Larochelle. Curriculum by smoothing. *Advances in Neural Information Processing Systems*, 33:21653–21664, 2020.

[22] Yulin Wang, Yang Yue, Rui Lu, Tianjiao Liu, Zhao Zhong, Shiji Song, Gao Huang. Efficient-train: Exploring generalized curriculum learning for training visual backbones. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pages 5852–5864.

[23] Florinel-Alin Croitoru, Nicolae-Cătălin Ristea, Radu Tudor Ionescu, Nicu Sebe. Learning rate curriculum. *International Journal of Computer Vision*, 133(1):291–314, 2025.

[24] Muhammad Asif Khan, Hamid Menouar, Ridha Hamila. LCDnet: a lightweight crowd density estimation model for real-time video surveillance. *Journal of Real-Time Image Proc.*, 20(2):29, 2023.

[25] Karen Simonyan, Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conf. on Comp. Vision and Pattern Recognition*, pages 770–778, 2016.

[27] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Toronto, ON, Canada*, 2009.