EAN: An Efficient Attention Module Guided by Normalization for Deep Neural Networks

Jiafeng Li[1], **Zelin Li**[2], **Ying Wen**[1*]

[1]School of Communication and Electronic Engineering, East China Normal University
[2]New York University Shanghai
51205904113@stu.ecnu.edu.cn, zelin.li.nyu@gmail.com, ywen@cs.ecnu.edu.cn

Abstract

Deep neural networks (DNNs) have achieved remarkable success in various fields, and two powerful techniques, feature normalization and attention mechanisms, have been widely used to enhance model performance. However, they are usually considered as two separate approaches or combined in a simplistic manner. In this paper, we investigate the intrinsic relationship between feature normalization and attention mechanisms and propose an Efficient Attention module guided by Normalization, dubbed EAN. Instead of using costly fully-connected layers for attention learning, EAN leverages the strengths of feature normalization and incorporates an Attention Generation (AG) unit to re-calibrate features. The proposed AG unit exploits the normalization component as a measure of the importance of distinct features and generates an attention mask using *GroupNorm*, L_2 *Norm*, and *Adaptation* operations. By employing a grouping, AG unit and aggregation strategy, EAN is established, offering a unified module that harnesses the advantages of both normalization and attention, while maintaining minimal computational overhead. Furthermore, EAN serves as a plug-and-play module that can be seamlessly integrated with classic backbone architectures. Extensive quantitative evaluations on various visual tasks demonstrate that EAN achieves highly competitive performance compared to the current state-of-the-art attention methods while sustaining lower model complexity.

Introduction

Due to the powerful ability of feature extraction, Convolutional Neural Networks have proven to be critical and robust in many computer vision tasks, such as image classification (Wang et al. 2023), object detection (Woo et al. 2023) and semantic segmentation (Li et al. 2023). Within this context, two techniques, namely feature normalization and attention mechanisms, play a pivotal role in boosting model performance due to their simplicity and effectiveness. Both techniques have gained considerable interest and are extensively employed in vision models, as they share a common goal of improving the feature extraction process.

Normalization techniques (Huang et al. 2023) have been widely studied for their crucial role in training deep neural networks, ensuring stability during training and faster convergence. The process of normalization consists of two components: feature standardization and affine transformation. For feature standardization, different methods are proposed for computing the mean and standard deviation to model the general data distribution. Batch Normalization (BN) (Ioffe and Szegedy 2015) is one of the early proposed and widely used normalization methods. It normalizes the feature map with the mean and standard deviation calculated along with the mini-batch, height, and width dimensions of a feature map. Based on BN, many normalization variants, such as Layer Normalization (LN) (Ba, Kiros, and Hinton 2016), Instance Normalization (IN) (Ulyanov, Vedaldi, and Lempitsky 2016), Group Normalization (GN) (Wu and He 2018), and Representative Batch Normalization (RBN) (Gao et al. 2021) have successively been proposed. These methods differ in their statistical computations. For instance, LN, IN, and GN normalize the features using statistics from the channel, sample, and channel group dimensions. For affine transformation, the standardized data is scaled and shifted independently for each channel to modulate feature distribution. This step helps to ensure that the standardized data aligns with the desired range and distribution required for a given task. Different strategies for learning the affine transformation parameters have been introduced, such as Switchable Normalization (SN) (Luo et al. 2018) and Exemplar Normalization (EN) (Zhang et al. 2020). For example, SN aims to learn a set of scaling parameters to adaptively choose the type of normalization that is most effective for the given input, while EN addresses a dynamic learning-to-normalize problem by adapting and learning data-dependent normalizations for different inputs. After standardization and affine transformation, these normalization techniques significantly enhance the training stability, optimization efficiency, and generalization ability of DNNs.

At the same time, the attention mechanism has emerged as a widely adopted approach to augment the feature representation ability of networks by focusing on regions with essential semantic features while suppressing irrelevant ones. Various types of attention methods tailored to different feature dimensions have been explored, including spatial attention, channel attention, and self-attention. Among these, channel attention has gained widespread popularity, as it learns importance weights for different channels using fully-connected layers. To ensure computational efficiency, Global Average Pooling

[*]Corresponding author: Ying Wen.

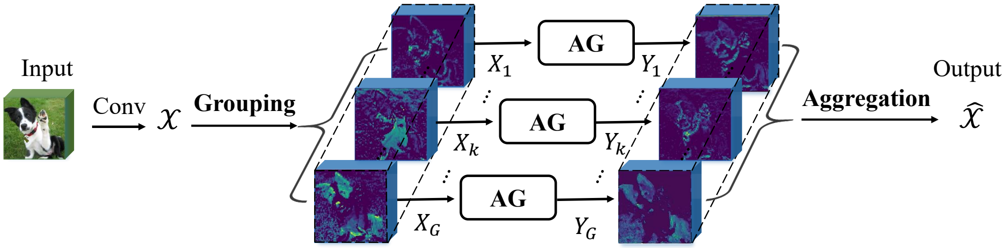


Figure 1: An overview of the proposed EAN module. It comprises three steps: Grouping, AG unit, and Aggregation. The Attention Generation (AG) unit works in parallel to process the sub-features of each feature group.

(GAP) has become a common choice due to its simplicity. However, the simplicity of GAP limits its ability to capture complex information in diverse inputs. To address this limitation, some researches like CBAM (Woo et al. 2018) and SRM (Lee, Kim, and Nam 2019) further use global max pooling and global standard deviation pooling to enhance the performance of GAP. FcaNet (Qin et al. 2021) introduces an idea that GAP is a special case of 2D DCT and proposes a multi-spectral channel attention framework that integrates frequency domain analysis and attention mechanisms. SA-Net (Zhang and Yang 2021) adopts Shuffle Units to capture feature dependencies in both spatial and channel dimensions. Moreover, SE-Net (Hu, Shen, and Sun 2018) employs two fully connected (FC) layers to capture intricate channel-wise dependencies, while ECA-Net (Wang et al. 2020) simplifies channel weight computations using a 1D convolution to reduce the redundancy of FC layers. Coordinate Attention (CA) (Hou, Zhou, and Feng 2021) incorporates positional information into channel attention by employing two 1D feature encoding processes along two spatial directions, allowing it to capture long-range dependencies. However, it is essential to note that these methods generally suffer from either converging difficulty or heavy model complexity and computation burdens, which hinder their practical usability.

Rather than treating feature normalization and attention as separate entities, certain studies have made an attempt to combine them. For example, Attentive Normalization (AN) (Li, Sun, and Wu 2020) learns a combination of affine transformations and generates a final transformation guided by attention. Meanwhile, NA (Ma et al. 2020) computes the similarity between query position and global context and applies the normalization method to design self-attention modules. However, these approaches merely employ a simplistic combination of normalization and attention techniques, without fully exploring their intrinsic connections. Without a doubt, this is not the most effective way to integrate them. A question has been raised, can we integrate feature normalization and attention in a more efficient manner?

To address this query, our investigation delves into the inner components of feature normalization and attention and finds a commonality in their internal parameters. Specifically, we observe that the affine transformation parameters in feature normalization (Eqn. 2) serve a similar role as the channel-wise scaling parameters in attention (Eqn. 3) during the re-calibration of input features, establishing a fundamental connection for their effective integration (Eqn. 4), thereby avoiding the demand for costly external fully-connected layers. Building upon the abovementioned observations, this paper proposes an Efficient Attention module guided by Normalization, dubbed EAN, which elegantly integrates normalization and attention with minimum computational overhead. EAN consists of three steps: Grouping, AG unit, and Aggregation. Firstly, a grouping strategy is introduced to partition the input features into multiple sub-features. Subsequently, an AG unit is proposed to generate an attention mask and perform parallel re-calibration of these sub-features (as depicted in Figure 1). Finally, the re-calibrated features are aggregated to obtain the enhanced output features. In contrast to existing attention modules that heavily rely on fully-connected layers for attention mask generation, our proposed AG unit leverages the affine transformation parameters derived from the internal statistics of features during the normalization process to determine the attention parameters, making the integration seamless and efficient.

Our main contributions can be summarized as follows:

- We propose an AG unit, an efficient attention generation module that ingeniously leverages the affine parameters during normalization to derive channel-wise attention parameters through *GroupNorm*, *L2 Norm* and *Adaptation* operations. This enables the network to re-calibrate features based on their internal statistical characteristics.
- We propose EAN, a lightweight plug-and-play module, that can be seamlessly inserted into existing architectures without any minor adjustments, enhancing feature representations and accelerating training and convergence.
- Extensive experimental results on ImageNet, CIFAR, and MS COCO datasets validate the superiority of the EAN module over state-of-the-art attention methods in terms of accuracy and convergence speed, with almost no extra parameters and computational overhead.

Methodology

In this section, we elaborate on the details of the proposed Efficient Attention module guided by Normalization (EAN), which takes intermediate feature tensor χ as input and obtains enhanced feature maps $\widehat{\chi}$ of the same size to χ.

Overview of the EAN

The EAN module consists of three steps, as depicted in Figure 1: Grouping, AG unit, and Aggregation.

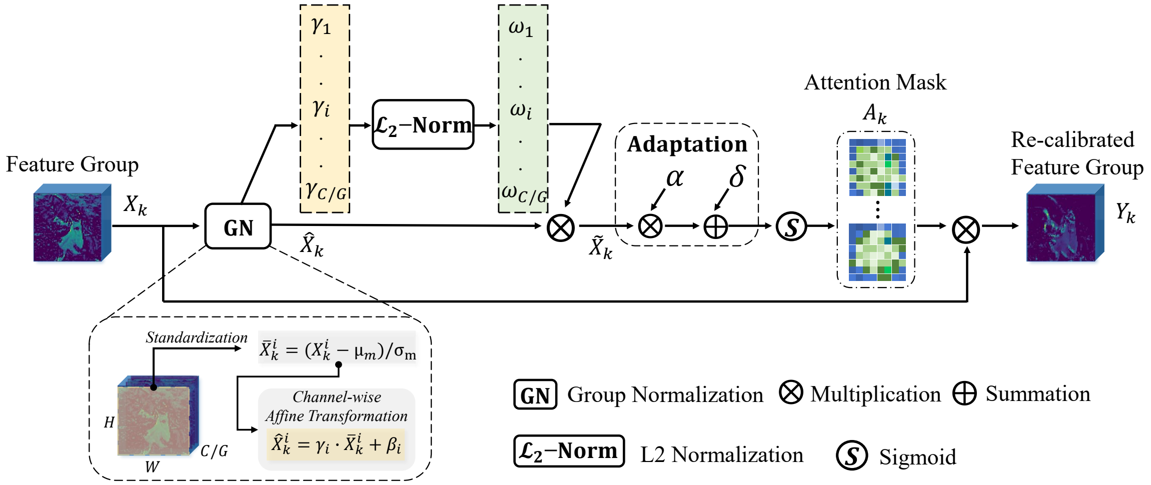


Figure 2: The detailed structure of the Attention Generation (AG) unit in the EAN module. Given the grouped features, AG generates attention weights by performing *GroupNorm*, L_2 *Norm*, and *Adaptation* operations.

Grouping. To obtain high-quality semantic feature information, EAN employs a grouping strategy to divide input features into channel-wise sub-features. For a given intermediate feature map $\chi \in \mathbb{R}^{C \times H \times W}$, where C is the channel axis, H and W are the spatial height and width axes, EAN initially splits the intermediate features χ into G groups along the channel dimension, which can be expressed as $\chi = [X_1, \ldots, X_k, \ldots, X_G]$, where each feature group $X_k \in \mathbb{R}^{C/G \times H \times W}$ contains C/G number of feature channels. Recent research (Li, Li, and Yang 2022) reported that the grouped features can effectively capture various discriminative semantic information throughout the network learning process. With this insight, we adopt a grouping scheme, as shown in Figure 1, to create distinct feature groups X_k that effectively capture specific classes of semantic responses during training. The EAN's grouping step ensures that each feature group X_k contributes to the network's specific informative semantic representations while avoiding interference between them. Be aware that the grouping number G has impacts on the performance of our module, and this will be thoroughly discussed in the experimental section.

However, due to the presence of inherent noise and redundant patterns, achieving well-distributed feature responses can be a challenging task. To address this, the EAN module incorporates an attention generation (AG) unit, which computes an attention mask for each feature group X_k. The AG unit is designed to re-calibrate the semantic information within each feature group, thereby enhancing its representation capability.

Attention Generation (AG) Unit. After the grouping step, each feature group X_k is obtained. In contrast to the conventional approach, which involves using fully-connected layers on X_k, introducing excessive parameters and computations, we delve into the potential advantages of leveraging the normalization technique widely applied in DNNs. In the following, we initially revisit the intrinsic connection between normalization and attention and then provide a detailed explanation of how the AG unit efficiently integrates them.

Revisit Normalization and Attention To delve deeper into the relationship between normalization and attention, we first review their standard operation. Given an input tensor $X_k \in \mathbb{R}^{C/G \times H \times W}$.

A typical feature normalization (Norm) includes two parts: standardization and affine transformation, which can be written as follows:

$$\begin{aligned} Standardization : \bar{X}_k &= \frac{X_k - \mu}{\sqrt{\sigma^2 + \epsilon}} \\ Affine : \hat{X}_k &= \gamma * \bar{X}_k + \beta \end{aligned} \tag{1}$$

where μ and σ denote the mean and standard deviation calculated within X_k, ϵ is a small positive constant added for the sake of division stability, and γ and β are learned scaling and bias factors for affine transformation. Then, Eqn. 1 can be rewritten as:

$$\hat{X}_k = \boldsymbol{Norm}(X_k) = f(X_k; \gamma, \beta) \tag{2}$$

where $\boldsymbol{Norm}(\cdot)$ is the normalization operation, consists of a function $f(\cdot)$ that normalizes the input features X_k with the learnable parameters γ and β.

For widely adopted attention mechanisms (Attn) that learn attention masks to re-calibrate features. The general process can be formulated as:

$$\hat{X}_k = \boldsymbol{Attn}(X_k) = \mathcal{A}(X_k; \theta) \tag{3}$$

where $\boldsymbol{Attn}(\cdot)$ is the attention operation, comprises a function $\mathcal{A}(\cdot)$ that reweights the X_k using learnable parameters θ.

We observe that the learned scaling parameters γ in the normalization function (Eqn. 2) and the learnable re-scaling parameters θ in the attention function (Eqn. 3) play an equal role in learning to re-calibrate features. Thus, we can directly leverage the learned scaling factors γ from normalization to acquire the re-scaling parameters θ_γ for attention. Further, we propose a novel Attention Generation (AG) unit that intrinsically combines normalization and attention into a unified module instead of using a two-module setup like Norm+Attn. The AG unit can be summarized as:

$$\hat{X}_k = \boldsymbol{AG}(X_k) = \mathcal{A}(X_k; \theta_\gamma) \tag{4}$$

where θ_γ represents the re-scaling parameters derived from the learned scaling factors γ in normalization.

Details of AG Unit As illustrated in Figure 2, the AG unit is composed of three essential components: *GroupNorm*, *L_2 Norm*, and *Adaptation*. For each feature group $X_k \in R^{C/G \times H \times W}$, the global contextual information contained in each feature channel space $< H, W >$ contributes differently to the entire feature group semantic response. Therefore, *GroupNorm (GN)* is adopted as our method of feature normalization, allowing us to learn instance-specific channel-wise affine transformations. The scaling factor γ in affine transformation plays a crucial role in adjusting the features to enhance their representational capability.

Taking this into account, we first standardize feature $X_k^i \in R^{C/G \times H \times W}$ in its i-th channel space $< H, W >$ as shown in the GN part of Figure 2, i is a feature channel index. This involves subtracting the mean value μ_m and dividing it by the standard deviation σ_m as follows :

$$\hat{X}_k^i = \boldsymbol{GN}(X_k) = \gamma_i * \frac{X_k^i - \mu_m}{\sqrt{\sigma_m^2 + \epsilon}} + \beta_i \tag{5}$$

where μ_m and σ_m are the mean and standard deviation calculated along the spatial dimension $m = H \times W$, ϵ is used to avoid zero variance, γ_i and β_i are learned scaling and bias factors in i-th feature channel for affine transformation.

It's worth noting that we leverage the learned scaling factors $\gamma \in R^{C/G}$, obtained from the affine transformation in GN layers to assess the variance of the global contextual information of each channel, as shown in the branch of Figure 2. This assessment enables us to evaluate the amount of global contextual information spatially across various feature channels. Specifically, a larger value of γ indicates richer spatial contextual information, reflecting greater variation in spatial pixels. Conversely, a smaller value of γ reveals the feature map dominated by noise or containing redundant contextual information. In a word, the learned scaling factors γ contribute differently to the feature semantic response and serve as an indicator of the importance of distinct features.

Next, to further strengthen the magnitude of the learned scaling factors across channels, we employ L_2 Normalization (*L_2 Norm*) for cross-channel operations. Let $W_\gamma = [\omega_1, \ldots, \omega_i, \ldots, \omega_{C/G}]$, the normalized correlation weights $W_\gamma \in R^{C/G}$ are obtained by Eqn. 6, indicating the relative importance of different features.

$$\omega_i = \frac{\gamma_i}{\|\gamma_i\|_2} = \frac{\gamma_i}{[\left(\sum_{i=1}^{C/G} \gamma_i^2\right) + \varepsilon]^{\frac{1}{2}}}, i = 1, 2, \cdots, C/G \tag{6}$$

where $\|\bullet\|_2$ is the L_2 normalization, ε is a small constant, γ_i and w_i are the learned scaling factor and weight coefficient at the position of the ith feature channel.

The obtained attention weights W_γ are the re-scaling parameters in Eqn. 4, which is derived from the scaling factors γ in GN (Eqn. 5). Then W_γ are individually assigned to each feature map on a channel-wise basis as follows:

$$\tilde{X}_k = W_\gamma \left(GN\left(X_k\right)\right) \tag{7}$$

Subsequently, we utilize an *adaptation* scheme where a pair of learnable parameters, $\alpha, \delta \in R^{C/G}$, are introduced to adapt $\tilde{X}_k$, enhancing its representation capability, as shown in Figure 2. These parameters have a negligible impact on computational costs compared to overall model parameters. Then an attention mask A_k is generated as Eqn. 8, providing guidance to re-calibrate the semantic response of the feature group. This whole process is achieved by a simple adaptation scheme with the $Sigmoid$ activation. Finally, the output feature group Y_k is re-calibrated as Eqn. 9:

$$A_k = Sigmoid\left(\alpha \tilde{X}_k + \delta\right) \tag{8}$$

$$Y_k = A_k \otimes X_k \tag{9}$$

where $\otimes$ denotes element-wise multiplication.

In brief, the AG unit utilizes the strengths of *GroupNorm*, *L_2 Norm* and *Adaptation*, efficiently guiding the feature group X_k to extract valuable semantic details while simultaneously reducing irrelevant contextual information.

Aggregation. Following the AG unit, the re-calibrated feature groups Y_k are aggregated to form the output feature $\hat{\mathcal{X}} = [Y_1, \ldots, Y_k, \ldots, Y_G]$. Throughout the continuous training iterations of the network, these output features gradually capture significant semantic responses while effectively suppressing redundant contextual information, such as noise.

In summary, the EAN module is established through the following three steps: Grouping, AG unit, and Aggregation. The grouping strategy is employed to obtain distinct feature groups. The AG unit is proposed to re-calibrate the semantic response within these feature groups. The aggregation scheme is used to combine the re-calibrated feature groups.

Integration Strategy

Our approach is implemented as a plug-and-play module, allowing it to be seamlessly inserted into any existing framework. However, like other attention methods, the specific placement of our module within the model is critical. Therefore, we incorporate the EAN module into four representative architectures with diverse building blocks.

For CNNs without skip connections, such as VGG, we directly embed the EAN into the convolutional layer. For CNNs with skip connections, such as ResNet and ResNeXt (with basic block or bottleneck design), we integrate the EAN module into the last layer of the residual block. In architectures utilizing the inverted residual block, such as the MobileNet series, the EAN module is incorporated during the expansion stage to ensure attention focuses on the largest representation. In MobileNeXt's sandglass block, the EAN module is plugged into the last convolutional layer of each block. The performance of these integrations will be evaluated in subsequent experiments, and it is worth noting that the associated parameter and computational costs are negligible.

Experiments

To evaluate the effectiveness of the proposed EAN module, we have conducted comprehensive experiments on a number of tasks and datasets. Specifically, we conduct experiments on image classification benchmarks, including ImageNet-1K (Russakovsky et al. 2015), CIFAR-100 and CIFAR-10 (Krizhevsky, Hinton et al. 2009), to validate the improvement brought by the EAN module in base CNN models. Additionally, we compare it with other state-of-the-art attention modules. Furthermore, we conduct experiments on object detection benchmarks, including MS COCO (Lin et al. 2014) and PASCAL VOC (Everingham et al. 2010) to examine the effectiveness and generality of the EAN module.

Datasets and Experimental Settings

Datasets. ImageNet-1K dataset is a large-scale image classification dataset, containing 1.28 million training images and 50k validation images from 1k classes. CIFAR dataset, including CIFAR-10 and CIFAR-100, consists of 50k training images and 10k validation images, which are divided into 10 and 100 classes, respectively. PASCAL VOC dataset, which has 20 classes, contains more than 22k images for training and 5k images for validation. MS COCO dataset, which is divided into 80 classes, has more than 118k images for training and 5k images for validation.

Implementation Details. For ImageNet-1K and CIFAR datasets, we follow a similar training scheme in (He et al. 2016), performing standard practices and data augmentation. Networks are trained for 200 epochs on CIFAR with SGD optimizer with a weight decay of $5 \times e^{-4}$ and a momentum of 0.9. The initial learning rate is initialized to 0.05 and is decayed by 0.1 at 100 and 150 of the epochs. For lightweight networks, we adopt the cosine learning schedule and set the weight decay to $4 \times e^{-5}$. For other networks, the batch size is 256 with an initial learning rate of 0.1. The weight decay is set to $1 \times e^{-4}$. For MS COCO and PASCAL VOC, we use SGD optimizer and set batch size to 16 and 32 respectively. The learning rate is set to 0.01 and 0.001 with a 500 iterations warmup. We train 200k iterations and reduce the learning rate by a factor of 10 at 120k and 180k iterations. The adaption parameters α and δ in the EAN module are initialized to 0. For fair comparisons, all models in each experiment are trained from scratch on NVIDIA Tesla V100 GPU with the default training strategy and no other tricks are used.

Ablation Studies

In this section, we report the ablation experiments on the CIFAR dataset, to thoroughly investigate the components of the EAN for a better understanding of its characteristics.

The Grouping Number G. The EAN module uses the number of groups G to control the diversity of semantic sub-features. Setting G to be too high or too low can result in inadequate feature representation. Thus, it is crucial to explore a moderate value of G. Figure 3 showcases the results of the optimal G with ResNet-50 and MobileNeXt on the CIFAR-100 dataset. As is observed, both models' performance exhibits an increasing trend initially and then declines

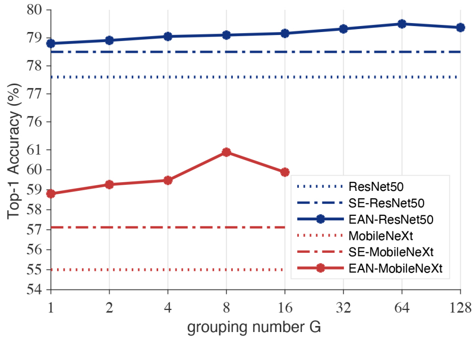


Figure 3: Results of our EAN with various group numbers of G using ResNet-50 and MobileNeXt as backbone models.

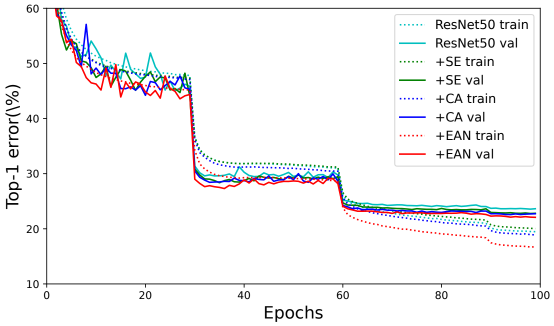


Figure 4: Training curve comparisons for ResNet50 with different methods on ImageNet-1K.

Architecture	Params/FLOPs	C10(%)	C100(%)
MobileNetV2	2.35M/6.79M	83.56	53.31
+SE	2.92M/7.42M	84.93	54.52
+CBAM	2.92M/8.03M	84.89	54.93
+ECA	2.35M/6.88M	85.10	55.12
+SA	2.35M/6.81M	85.45	55.61
+SGE	2.35M/6.86M	85.56	55.59
+CA	3.21M/8.69M	85.89	55.85
+EAN(Ours)	**2.35M/6.79M**	**86.74**	**56.20**
MobileNeXt	2.38M/6.92M	85.71	54.77
+SE	3.10M/7.87M	86.04	57.12
+CBAM	3.03M/8.47M	86.52	57.53
+ECA	2.38M/7.18M	86.61	57.74
+SA	2.38M/7.14M	86.78	58.03
+SGE	2.38M/7.16M	86.94	58.36
+CA	3.46M/8.95M	87.12	58.58
+EAN(Ours)	**2.38M/6.92M**	**88.00**	**60.88**
ShuffleNet	1.36M/45.75M	91.06	69.20
+SE	1.53M/46.39M	91.63	69.64
+CBAM	1.62M/48.29M	91.89	70.23
+EAN(Ours)	**1.36M/45.75M**	**92.25**	**70.86**
DenseNet-121	7.05M/898.23M	95.47	79.20
+SE	7.33M/898.88M	95.63	79.73
+CBAM	7.19M/899.07M	95.77	79.97
+EAN(Ours)	**7.05M/898.23M**	**96.12**	**80.61**
EfficientNetB0	4.66M/8.79M	90.36	66.80
+SE	6.18M/10.36M	91.34	67.78
+CBAM	6.18M/11.89M	91.66	68.55
+EAN(Ours)	**4.66M/8.79M**	**92.14**	**69.35**

Table 1: Image classification results for common lightweight networks on CIFAR-10 and CIFAR-100 dataset.

Architecture	Params(M)	FLOPs(G)	Top-1 Acc(%)
ResNet50	23.71	1.30	77.60
+SE	28.75	1.31	78.54
+CBAM	28.74	1.32	78.65
+SA	23.71	1.31	78.88
+CA	27.51	1.34	78.95
+EAN(Ours)	**23.71**	**1.30**	**79.45**
ResNet101	42.70	2.52	77.78
+SE	52.22	2.53	78.96
+CBAM	52.18	2.54	78.55
+SA	42.70	2.52	79.44
+CA	49.90	2.59	79.05
+EAN(Ours)	**42.70**	**2.52**	**80.05**
ResNet56	0.86	0.127	71.50
+SE	0.88	0.127	72.24
+CBAM	0.88	0.128	72.65
+SA	0.86	0.127	73.10
+CA	0.89	0.128	73.25
+EAN(Ours)	**0.86**	**0.127**	**73.79**
WRN-28(w=10)	36.55	5.96	78.65
+SE	37.09	5.96	79.42
+CBAM	37.09	5.96	79.05
+CA	37.36	5.97	79.35
+EAN(Ours)	**36.55**	**5.96**	**79.86**
ResNeXt-29	25.14	4.05	81.10
+SE	26.18	4.05	81.99
+CBAM	26.17	4.05	81.82
+CA	26.70	4.07	82.10
+EAN(Ours)	**25.14**	**4.05**	**82.68**

Table 2: Comparison of SOTA attention methods across a range of ResNets and its variants on CIFAR-100 dataset.

as G increases. The MobileNeXt is a lightweight model and the maximum G is 16. Thus, we recommend setting the G to 8 for lightweight models and 64 for larger models, as this configuration strikes the balance between semantic diversity and effective representation for optimal network performance.

Image Classification

In order to evaluate the effect of the proposed EAN, in this section, we conduct several experiments on two widely-used image classification datasets: CIFAR and ImageNet-1k.

Results on CIFAR We first conduct experiments using classical lightweight networks, including MobileNetV2 (Sandler et al. 2018), MobileNeXt (Zhou et al. 2020), ShuffleNetV2 (Ma et al. 2018), DenseNet-121 (Huang et al. 2017), and EfficientNetB0 (Tan and Le 2021), on CIFAR-10 and CIFAR-100 datasets. We compare the performance of EAN against SOTA attention methods such as ECA (Wang et al. 2020), SA (Zhang and Yang 2021), and SGE (Li, Li, and Yang 2022). The results in Table 1 demonstrate that, on lightweight backbones, EAN outperforms all competing attention modules in terms of accuracy with no additional computational overhead.

Furthermore, we expand our experiments to a range of ResNet architecture and its variants, such as ResNet-50, ResNet-101, ResNet-56, WideResNet-28 (WRN-28) (Zagoruyko and Komodakis 2016), and ResNeXt-29 (Xie et al. 2017). The results are summarized in Table 2, which show EAN surpasses SOTA methods with almost no extra parameters and computations. Specifically, when using ResNeXt-29 as a backbone, the EAN shares almost the same model complexity with the original network, but achieves 1.95% gains in Top-1 accuracy, which demonstrates the EAN is lighter and more efficient.

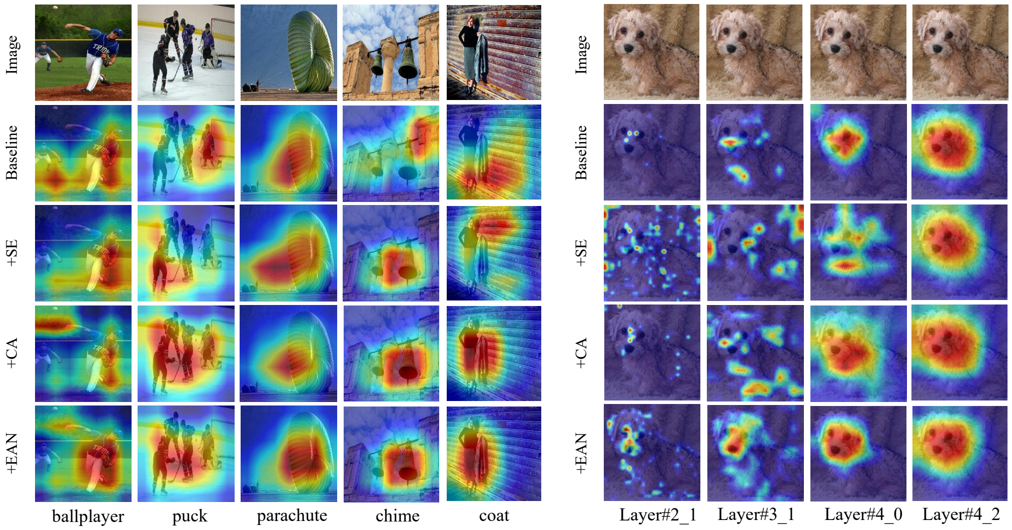
Architecture	Params	FLOPs	Inference	Top-1(%)
MobileNeXt(MX)	3.54M	343.66M	115FPS	72.45
SE-MX	4.19M	345.95M	**80FPS**	73.51
CBAM-MX	4.19M	347.49M	62FPS	73.67
SGE-MX	3.54M	345.30M	67FPS	73.58
PdfAM-MX	3.54M	343.66M	64FPS	73.60
CA-MX	4.39M	356.50M	55FPS	73.89
EAN-MX(Ours)	**3.54M**	**343.66M**	78FPS	**74.21**
MobileNetV2(MV2)	3.50M	327.49M	138FPS	71.53
SE-MV2	4.08M	331.40M	**92FPS**	72.32
CBAM-MV2	4.07M	332.37M	70FPS	72.35
SGE-MV2	3.50M	330.83M	84FPS	72.45
PdfAM-MV2	3.50M	327.49M	78FPS	72.38
CA-MV2	4.37M	346.59M	64FPS	72.55
EAN-MV2(Ours)	**3.50M**	**327.49M**	79FPS	**72.80**
ResNet50(R50)	25.56M	4.11G	114FPS	76.45
SE-R50	28.09M	4.12G	78FPS	77.31
CBAM-R50	28.07M	4.13G	58FPS	77.42
SGE-R50	25.56M	4.12G	73FPS	77.23
PdfAM-R50	25.56M	4.11G	62FPS	77.33
CA-R50	28.08M	4.16G	62FPS	77.40
EAN-R50(Ours)	**25.56M**	**4.11G**	**78FPS**	**77.87**

Table 3: Comparison of different attention methods on ImageNet-1K dataset.

Results on ImageNet-1K We further explore whether the superior performance of the EAN module could be generalized to other datasets. Comparative experiments are carried out using a wide range of attention mechanisms on ImageNet-1K, including SE (Hu, Shen, and Sun 2018), CBAM (Woo et al. 2018), SGE (Li, Li, and Yang 2022), PdfAM (Xie and Zhang 2023), and CA (Hou, Zhou, and Feng 2021). The baseline backbones are MobileNeXt, MobileNetV2, and ResNet-50. All results, as shown in Table 3, indicate that the EAN module consistently outperforms other attention methods while introducing almost no additional parameters. Specifically, our EAN-embedded models exhibit superior performance, surpassing the original MobileNeXt, MobileNetV2, and ResNet-50 by 2.42%, 1.78%, and 1.86% respectively. Moreover, EAN-R50 shares comparable model complexity with the PdfAM module while achieving a 0.67% advantage over Top-1 accuracy. Table 3 also compares the inference speed. Our EAN obtains a similar inference FPS compared to SE and proves to be more efficient than other attentions (i.e., CA), as in MobileNeXt comparisons (78FPS vs. 55FPS). Additionally, Figure 4 shows the training and validation curves of ResNet-50 with EAN, which constantly demonstrates improved and accelerated training and convergence, underscoring the effectiveness and stability of our method.

Object Detection

To verify the generalization performance of EAN, we further conduct experiments on object detection, including MS COCO (Lin et al. 2014) and PASCAL VOC (Everingham et al. 2010) datasets. We adopt SSDLite (Liu et al. 2016) and Faster-RCNN (Ren et al. 2015) framework as our detection method, using ImageNet pre-trained MobileNeXt and ResNet-50 as our baseline networks.



(a) GradCAM of different objects

(b) GradCAM at different layers

Figure 5: Visualization of class activation mapping using MobileNeXt and ResNet50 as backbone networks.

Backbone	Detector	Params(M)	mAP(%)
MobileNeXt	SSDLite320	4.4	72.3
SE-MobileNeXt	SSDLite320	4.8	72.4
CA-MobileNeXt	SSDLite320	4.9	72.9
EAN-MobileNeXt	SSDLite320	4.4	**73.4**

Table 4: Object detection results of different methods on PASCAL VOC 2012 val set.

Backbone	Params(M)	AP@.5	AP@.75	mAP
MobileNeXt	4.3	37.4	22.7	22.3
SE-MobileNeXt	4.7	39.3	23.4	22.6
CA-MobileNeXt	4.8	39.8	24.1	23.0
EAN-MobileNeXt	**4.3**	**40.2**	**24.6**	**23.5**
ResNet50	41.7	58.4	39.1	36.4
SE-ResNet50	44.3	60.1	40.9	37.7
CA-ResNet50	44.7	**60.8**	41.7	38.0
EAN-ResNet50	**41.7**	60.8	**42.2**	**38.4**

Table 5: Object detection results on MS COCO 2017 val set.

Results on PASCAL VOC In Table 4, we can observe that when the same SSDLite320 detector is adopted, MobileNeXt with our EAN embedded achieves superior gain (1.52%) over the original network while performing better than SE and CA block with fewer model complexity.

Results on MS COCO We further explore the EAN module using Faster RCNN and SSDLite320 on the COCO 2017 validation set, reporting the results in terms of mAP, AP@.5, and AP@.75. Table 5 shows that our method achieves the best performance in both detection frameworks compared to its corresponding attention variants. Specifically, our EAN outperforms the original ResNet50 and MobileNeXt by 5.5% and 5.4%, respectively, and surpasses CA by 1.1% and 2.3% with almost no extra parameters. These results highlight the efficiency and lightweight nature of the EAN, showing its superior transferable capabilities across various vision tasks.

Visualization

To provide a more intuitive demonstration of the effectiveness of the EAN module, we sample 5 images from ImageNet-1K validation dataset and utilize GradCAM (Selvaraju et al. 2017) to visualize the class activation mapping. For comparative analysis, we also generate heat maps for SE and CA embedded models. The visualizations are presented in Figure 5, where Figure 5(a) and Figure 5(b) correspond to MobileNeXt and ResNet50 respectively, both trained on ImageNet-1K. In Figure 5(a), a comparison between EAN, SE, and CA reveals that the EAN activation mapping encompasses a larger region of the relevant objects, such as the ”ballplayer” and the ”parachute”. Figure 5(b) displays attention heat maps from four different layers, illustrating EAN's impact in reducing background noise and refining attention. As a result, the activation map becomes more precise, capturing critical and accurate locations for semantic representations.

Conclusion

In this paper, we explore the intrinsic relationship between two widely used techniques for enhancing models: feature normalization and attention. Further, we propose an Efficient Attention module guided by Normalization, named EAN. EAN incorporates an AG unit to derive attention weights using parameter-efficient normalization and guide the network to capture relevant semantic responses while suppressing irrelevant ones. The AG unit harnesses the strengths of normalization and attention and combines them into a unified module to enhance feature representation. EAN is also a plug-and-play module. Extensive experiments on multiple benchmark datasets validate the superior accuracy and convergence of our EAN compared to state-of-the-art methods.

Acknowledgments

This work was supported in part by the National Nature Science Foundation of China (62273150, 62171323), Shanghai Natural Science Foundation (22ZR1421000), Shanghai Outstanding Academic Leaders Plan (21XD1430600), Science and Technology Commission of Shanghai Municipality (22DZ2229004).

References

Ba, J. L.; Kiros, J. R.; and Hinton, G. E. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Everingham, M.; Van Gool, L.; Williams, C. K.; Winn, J.; and Zisserman, A. 2010. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2): 303–338.

Gao, S.-H.; Han, Q.; Li, D.; Cheng, M.-M.; and Peng, P. 2021. Representative batch normalization with feature calibration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 8669–8679.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hou, Q.; Zhou, D.; and Feng, J. 2021. Coordinate attention for efficient mobile network design. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 13713–13722.

Hu, J.; Shen, L.; and Sun, G. 2018. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 7132–7141.

Huang, G.; Liu, Z.; Van Der Maaten, L.; and Weinberger, K. Q. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4700–4708.

Huang, L.; Qin, J.; Zhou, Y.; Zhu, F.; Liu, L.; and Shao, L. 2023. Normalization techniques in training dnns: Methodology, analysis and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Ioffe, S.; and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, 448–456. pmlr.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

Lee, H.; Kim, H.-E.; and Nam, H. 2019. Srm: A style-based recalibration module for convolutional neural networks. In *Proceedings of the IEEE/CVF International conference on computer vision*, 1854–1862.

Li, K.; Wang, Y.; Zhang, J.; Gao, P.; Song, G.; Liu, Y.; Li, H.; and Qiao, Y. 2023. Uniformer: Unifying convolution and self-attention for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Li, X.; Sun, W.; and Wu, T. 2020. Attentive normalization. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVII 16*, 70–87. Springer.

Li, Y.; Li, X.; and Yang, J. 2022. Spatial group-wise enhance: Enhancing semantic feature learning in cnn. In *Proceedings of the Asian Conference on Computer Vision*, 687–702.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, 740–755. Springer.

Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; and Berg, A. C. 2016. Ssd: Single shot multibox detector. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, 21–37. Springer.

Luo, P.; Ren, J.; Peng, Z.; Zhang, R.; and Li, J. 2018. Differentiable learning-to-normalize via switchable normalization. *arXiv preprint arXiv:1806.10779*.

Ma, N.; Zhang, X.; Zheng, H.-T.; and Sun, J. 2018. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European conference on computer vision (ECCV)*, 116–131.

Ma, X.; Guo, J.; Chen, Q.; Tang, S.; Yang, Q.; and Fu, S. 2020. Attention meets normalization and beyond. In *2020 IEEE International Conference on Multimedia and Expo (ICME)*, 1–6. IEEE.

Qin, Z.; Zhang, P.; Wu, F.; and Li, X. 2021. Fcanet: Frequency channel attention networks. In *Proceedings of the IEEE/CVF international conference on computer vision*, 783–792.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3): 211–252.

Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; and Chen, L.-C. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4510–4520.

Selvaraju, R. R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; and Batra, D. 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, 618–626.

Tan, M.; and Le, Q. 2021. Efficientnetv2: Smaller models and faster training. In *International conference on machine learning*, 10096–10106. PMLR.

Ulyanov, D.; Vedaldi, A.; and Lempitsky, V. 2016. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*.

Wang, Q.; Wu, B.; Zhu, P.; Li, P.; Zuo, W.; and Hu, Q. 2020. ECA-Net: Efficient channel attention for deep convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 11534–11542.

Wang, W.; Dai, J.; Chen, Z.; Huang, Z.; Li, Z.; Zhu, X.; Hu, X.; Lu, T.; Lu, L.; Li, H.; et al. 2023. Internimage: Exploring large-scale vision foundation models with deformable convolutions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14408–14419.

Woo, S.; Debnath, S.; Hu, R.; Chen, X.; Liu, Z.; Kweon, I. S.; and Xie, S. 2023. Convnext v2: Co-designing and scaling convnets with masked autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 16133–16142.

Woo, S.; Park, J.; Lee, J.-Y.; and Kweon, I. S. 2018. Cbam: Convolutional block attention module. In *Proceedings of the European conference on computer vision (ECCV)*, 3–19.

Wu, Y.; and He, K. 2018. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, 3–19.

Xie, J.; and Zhang, J. 2023. Less is more important: an attention module guided by probability density function for convolutional neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 2947–2955.

Xie, S.; Girshick, R.; Dollár, P.; Tu, Z.; and He, K. 2017. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1492–1500.

Zagoruyko, S.; and Komodakis, N. 2016. Wide residual networks. *arXiv preprint arXiv:1605.07146*.

Zhang, Q.-L.; and Yang, Y.-B. 2021. Sa-net: Shuffle attention for deep convolutional neural networks. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2235–2239. IEEE.

Zhang, R.; Peng, Z.; Wu, L.; Li, Z.; and Luo, P. 2020. Exemplar normalization for learning deep representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 12726–12735.

Zhou, D.; Hou, Q.; Chen, Y.; Feng, J.; and Yan, S. 2020. Rethinking bottleneck structure for efficient mobile network design. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, 680–697. Springer.