Attention-based deep neural networks (DNNs) that emphasize the informative information in a local receptive field of an input image have successfully boosted the performance of deep learning in various challenging problems. In this paper, we propose a Dense-and-Implicit-Attention (DIA) unit that can be applied universally to different network architectures and enhance their generalization capacity by repeatedly fusing the information throughout different network layers. The communication of information between different layers is carried out via a modified Long Short Term Memory (LSTM) module within the DIA unit that is in parallel with the DNN. The sharing DIA unit links multi-scale features from different depth levels of the network implicitly and densely. Experiments on benchmark datasets show that the DIA unit is capable of emphasizing channel-wise feature interrelation and leads to significant improvement of image classification accuracy. We further empirically show that the DIA unit is a nonlocal normalization tool that enhances the Batch Normalization. The code is released at https://github.com/gbup-group/DIANet.

1 Introduction

Attention, a cognitive process that selectively focuses on a small part of information while neglects other perceivable information [1], has been used to effectively ease neural networks from learning large information contexts from images [2, 3], sentences [4, 5, 6] and videos [7]. Especially in computer vision, DNNs incorporated with special operators that mimic the attention mechanism can process informative regions in an image efficiently. Empirical results in [8, 9, 2, 10, 4, 11] have demonstrated the visual representations improvement of DNNs by these special attention operators [12]. Recently, attention operators are modularized and stacked into popular networks as attention modules [13, 14] for further performance improvement. The design of these modules may be task-dependent [15, 16, 17, 18, 12, 19, 20].

In previous works, attention-modules are used individually in each layer throughout DNNs. Even a tiny add-in module with a small number of parameters per layer will primarily increase the total number of parameters as the network depth increases. Besides, the potential network redundancy can also hinder the learning capability of DNNs [14, 21]. Therefore, it is crucial to design efficient
attention modules that can maintain reasonable parameter cost and avoid feature redundancy to improve the performance of DNNs.

1.1 Our contribution

To tackle the challenge mentioned above, we propose a Dense-and-Implicit Attention (DIA) unit to enhance the generalization capacity of DNNs. The DIA unit can recurrently fuse multi-scale information from previous network layers to later network layers to model attention. The structure and computation flow of a DIA unit is visualized in Figure 1. There are three parts in the DIA unit. The part denoted by $\circ$ extracts spatial-wise [22], channel-wise [15] or multi-scale features [16, 17] from the feature map in the current layer. The second part denoted by $\bullet$ is the main module in the DIA unit to model network attention and is the key innovation of the proposed method. Particularly, we apply Long Short-Term Memory (LSTM) [23] module that not only connects two adjacent layers, but also fuses the information from previous layers, creating nonlocal information communication throughout the DNN. Other network structures can also be explored to implement the second part and this is left as future work. The third part denoted by $\oplus$ adjusts (e.g., re-scaling or re-distributing) the feature map in the current layer according to the feedback of the second part.

Characteristics and Advantages. (1) As shown in Figure 1, the DIA unit is placed parallel to the network backbone, and it is shared with all the layers in the same stage (the collection of successive layers with same spatial size, as defined in [13]). It also links different layers implicitly and densely, which improves the interaction of layers at different depth. (2) As the DIA unit is shared, the number of parameter increment from the DIA unit remains roughly constant as the depth of the network increases. (3) DIA unit adaptively learns the non-local scaling that has the normalization effect.

Figure 1: DIA units. $F_{ext}$ means the operation for extracting different scales of features. $F_{cmp}$ means the operation for emphasizing features.

In this work, we focus on studying the DIA unit using a modified LSTM that can learn the channel-wise relationship and long distance dependency between layers. As shown in Figure 2, a global average pooling (GAP) layer (as the $\circ$ in Figure 1) is used to extract global information from current layer; A LSTM module (as the $\bullet$ in Figure 1) is used to integrate multi-scale information and there are three inputs passed to the LSTM: the extracted global information from current raw feature map, the hidden state vector $h_{t-1}$, and cell state vector $c_{t-1}$ from previous layers. Then the LSTM outputs the new hidden state vector $h_t$ and the new cell state vector $c_t$. The new hidden state vector $h_t$ stores the information from the $t^{th}$ layer and its preceding layers. The new hidden state vector $h_t$ (dubbed as attention vector in our work) is then applied back to the raw feature map by channel-wise multiplication (as the $\oplus$ in Figure 1) to emphasize the feature importance in each feature map.

The LSTM in the DIA unit plays a role to bridge the current layer and preceding layers such that the DIA unit can adaptively learn the non-linearity relationship between features in two different scales. The first scale of features is the internal information of the current layer, and the second scale represents the outer information from the preceding layers. The non-linearity relationship between these two scales will benefit attention modeling for the current layer. Additionally, the operation of channel-wise multiplication in the final step of the DIA unit is similar to the scaling operation of Batch Normalization. Indeed, the empirical results show that DIA has non-local normalization effect. With our modified LSTM, the additional parameter cost is economical in DIA unit.

1.2 Organization

In Section 2, some of the attention-based networks are reviewed, and the difference between them and ours are discussed. Then in Section 3, the formal definition of Dense-and-Implicit Attention Network is introduced. In Section 4, we conduct experiments on Benchmarks datasets to empirically demonstrate the effectiveness of DIA unit. In Section 5, the influence of hyper-parameter on our
model has been studied experimentally. Finally, in Section 6, some evidence of long-distance dependence in our model is shown, and the normalization effects of DIA unit are also investigated.

Figure 2: Illustration of DIANet with LSTM. In the LSTM module, $c_t$ is the cell state vector and $h_t$ is the hidden state vector. GAP means global average pool and $\otimes$ means channel-wise multiplication.

2 Related Works

Attention Mechanism in Computer Vision. [8, 24] use attention mechanism in image classification via utilizing a recurrent neural network to select and process local regions at high resolution sequentially. Concurrent attention-based methods tend to construct operation modules to capture non-local information in an image [18, 20], and model the interrelationship between channel-wise features [15, 12]. The combination of multi-level attentions are also widely studied [17, 16, 25, 26]. Prior works [18, 20, 15, 12, 17, 16, 25] usually insert an attention module in each layer independently. In this work, the DIA unit is innovatively shared for all the layers in the same stage of the network, and the existing attention modules can be composited into the DIA unit readily. Besides, we adopt a global average pooling in part 1 to extract global information and a channel-wise multiplication in part 2 to enhance the importance of features, which is similar to SENet [15].

Dense Network Topology. DenseNet proposed in [14] connects all pairs of layers directly with an identity map. Through reusing features, DenseNet enjoys the advantage of higher parameter efficiency, the better capacity of generalization, and more accessible training than alternative architectures [27, 13, 28]. Instead of explicitly dense connections, the DIA unit implicitly links layers at different depth via a shared module and leads to dense connection. Empirical results in Section 6 shows that the DIA unit has a similar effect as DenseNet.

Multi-level Feature Integration. [29] experimentally analyzes that even the simple aggregation of low-level visual features sampled from wide inception field can be efficient and robust for context representation, which inspires [15, 12] to incorporate multi-level features to improve the network representation. [30] also demonstrates that by biasing the feature response in each convolutional layers using different activation functions, the deeper layer could achieve the better capacity of capturing the abstract pattern in DNN. In DIA unit, the high non-linearity relationship between multi-scale features are learned and integrated via the LSTM module, which is conducive to better model the attention and improve learning performance reported in Section 4, 5 and 6.

3 Dense-and-Implicit Attention Network (DIANet)

In this section, we will formally introduce the DIA unit and elaborate on how it implicitly connects all layers densely. Afterwards, a DIANet is referred to a network built with DIA units.

3.1 Formulation of DIANets

As shown in Figure 2 when a DIANet is built with a residual network [13], the input of the $t^{th}$ layer is $x_t \in \mathbb{R}^{W \times H \times N}$, where $W$, $H$ and $N$ mean width, height and the number of channels, respectively. $f(\cdot; \theta_1^{(t)})$ is the residual mapping at the $t^{th}$ layer with parameters $\theta_1^{(t)}$ as introduced in [13]. Let $a_t = f(x_t; \theta_1^{(t)}) \in \mathbb{R}^{W \times H \times N}$. Next, a global average pooling denoted as GAP(·) is applied to $a_t$ to extract global information from features in the current layer. Then GAP($a_t$) $\in \mathbb{R}^N$ is passed to LSTM along with a hidden state vector $h_{t-1}$ and a cell state vector $c_{t-1}$ ($h_0$ and $c_0$ are initialized as zero vectors). The LSTM finally generates a current hidden state vector $h_t \in \mathbb{R}^N$ and a cell state vector $c_t \in \mathbb{R}^N$ as

$$ (h_t, c_t) = \text{LSTM}(\text{GAP}(a_t), h_{t-1}, c_{t-1}; \theta_2). \quad (1) $$
In our model, the hidden state vector $h_t$ is regarded as attention vector to adaptively recalibrate feature maps. We apply channel-wise multiplication $\otimes$ to enhance the importance of features, i.e., $x_{t+1} = x_t + a_t \otimes h_t$. Table 1 shows the formulation of ResNet, SENet, and DIANet, and Part (b) is the main difference between them. The LSTM module is used repeatedly and shared with different layers in parallel to the network backbone. Therefore the number of parameters $\theta_2$ in a LSTM does not depend on the number of layers in the backbone, e.g., $t$. SENet utilizes a attention-module consisted of fully connected layers to model the channel-wise dependency for each layer independently [15]. The total number of parameters brought by the add-in modules depends on the number of layers in the backbone and increases with the number of layers.

|       | ResNet         | SENet         | DIANet (ours) |
|-------|----------------|---------------|---------------|
| (a)   | $a_t = f(x_t; \theta_1^{(1)})$ | $a_t = f(x_t; \theta_1^{(1)})$ | $a_t = f(x_t; \theta_1^{(2)})$ |
| (b)   | $x_{t+1} = x_t + a_t$ | $h_t = \text{FC}(\text{GAP}(a_t); \theta_2^{(1)})$ | $(h_t, c_t) = \text{LSTM}(\text{GAP}(a_t), h_{t-1}, c_{t-1}; \theta_2)$ |
| (c)   | $x_{t+1} = x_t + a_t \otimes h_t$ | $x_{t+1} = x_t + a_t \otimes h_t$ | $x_{t+1} = x_t + a_t \otimes h_t$ |

Table 1: Formulation for the structure of ResNet, SENet, and DIANet. $f$ is the convolution layer. FC means fully connected layer and GAP indicates global average pooling.

### 3.2 Modified LSTM Module (DIA-LSTM Module)

Now we introduce the modified LSTM module. As shown in Figure 3, compared to the standard LSTM [23] module, there are two modifications in our purposed LSTM: 1) a shared linear transformation to reduce input dimension; 2) a careful selected activation function for better performance.

A standard LSTM consists of four linear transformation layers as shown in Figure 3 (Left). Since $y_t$, $h_{t-1}$ and $h_t$ are of the same dimension $N$, the standard LSTM may cause $8N^2$ parameter increment as shown in the Appendix. When the number of channels is large, e.g., $N = 2^{10}$, the parameter increment of add-in LSTM module in the DIA unit will be over 8 million, which can hardly be tolerated.

![Figure 3: Left: standard LSTM module. Right: DIA-LSTM module. We highlight the modified component in DIA-LSTM. “σ” means the sigmoid activation. “Linear” means the linear transformation.](image)

Hence, to avoid such a scenario, we propose a modified LSTM (denoted as DIA-LSTM) as follows:

1. **Activation Function** We change the output layer’s activation function from $\tanh$ to $\text{sigmoid}$ and further discussion will be presented in ablation study;

2. **Parameter Reduction** As shown in Figure 3 (Left), $h_{t-1}$ and $y_t$ are passed to four linear transformation layers with the same input and output dimension $N$. In the DIA-LSTM, a linear transformation layer (denoted as “Linear1” in Figure 3 (Right)) with a smaller output dimension are applied to $h_{t-1}$ and $y_t$. We use reduction ratio $r$ in the Linear1. Specifically, we reduce the dimension of the input from $1 \times 1 \times N$ to $1 \times 1 \times \frac{N}{r}$ and then apply the $\text{ReLU}$ activation function to increase non-linearity in this module. The dimension of the output from $\text{ReLU}$ function are changed back to $1 \times 1 \times N$ when the output is passed to those four linear transformation functions. This modification can enhance the relationship between the inputs for different parts in DIA-LSTM and also effectively reduce the number of parameters by sharing a linear transformation for dimension reduction. The number of parameter increment reduces from $8N^2$ to $10N^2/r$ as shown in the Appendix, and we
find that when we choose an appropriate reduction ratio $r$, we can make a better trade-off between parameter reduction and the performance of DIANet. Further experimental results will be discussed in the ablation study later.

**Implicit and Dense Connection.** Now we illustrate how the DIA unit connects all the layers in the same stage implicitly and densely. Consider a stage consisting many layers, as shown in Figure 4 (Left), it is an explicit structure of DIANet and the one layer seems not to be connected to the other layers except the network backbone. However, as shown in Figure 4 (Right), there are indirect connections between the current layer and the preceding layers with the help of the shard DIA unit. Unlike DenseNet [14] that concatenates the raw features of all previous layers directly, the layer receives the information of the preceding layers through data-driven learning within the DIA unit. We call them implicit connections. Since the DIA unit is shared, there is communication between every pair of layers, which leads to dense connections over all layers.

### 4 Experiments

In this section, we evaluate the performance of the DIA unit in image classification task and empirically demonstrate its effectiveness. We conduct experiments on popular networks for benchmark datasets. Since the SENet [15] is also a channel-specific attention model, we compare the DIANet with the SENet. For a fair comparison, we adjust the reduction ratio such that the number of parameters of the DIANet is similar to that of the SENet.

**Dataset and Model.** We conduct experiments on CIFAR10, CIFAR100 [31], and ImageNet 2012 [32] using ResNet [13], PreResNet [33], WRN [34] and ResNeXt [35], CIFAR10 or CIFAR100 has 50k train images and 10k test images of size 32 by 32, but has 10 and 100 classes respectively. ImageNet 2012 dataset [32] comprises 1.28 million training and 50k validation images from 1000 classes, and the random cropping of size 224 by 224 is used in our experiments. The implementation details can be found in the Appendix.

**Image Classification.** As shown in Table 2, the DIANet improves the testing accuracy significantly over the original networks and consistently comparing with SENet for different datasets. In particular, the performance improvement of the ResNet with the DIA unit is most remarkable. Due to the popularity of ResNet, the DIA unit may be applied in other computer vision tasks.

|                  | Dataset | original | SENet | DIANet |           | #P(M) | DIAunit | r |
|------------------|---------|----------|-------|--------|-----------|-------|---------|---|
|                  |         | #P(M)    | top1-acc. | #P(M) | top1-acc. | #P(M) | top1-acc. |   |
| ResNet164        | CIFAR100| 1.73     | 73.43  | 1.93   | 75.03     | 1.95  | 76.67   | 4 |
| PreResNet164     | CIFAR100| 1.73     | 76.53  | 1.92   | 77.41     | 1.96  | 78.20   | 4 |
| WRN52-4          | CIFAR100| 12.07    | 79.75  | 12.42  | 80.35     | 12.30 | 80.99   | 4 |
| ResNext101,8x32  | CIFAR100| 32.14    | 81.18  | 34.03  | 82.45     | 33.01 | 82.46   | 4 |
| ResNet164        | CIFAR10 | 1.70     | 93.54  | 1.91   | 94.27     | 1.92  | 94.58   | 4 |
| PreResNet164     | CIFAR10 | 1.70     | 95.01  | 1.90   | 95.18     | 1.94  | 95.23   | 4 |
| WRN52-4          | CIFAR10 | 12.05    | 95.96  | 12.40  | 95.95     | 12.28 | 96.17   | 4 |
| ResNext101,8x32  | CIFAR10 | 32.09    | 95.73  | 33.98  | 96.09     | 32.96 | 96.24   | 4 |
| ResNet34         | ImageNet| 21.81    | 73.93  | 21.97  | 74.39     | 21.98 | 74.60   | 20|
| ResNet50         | ImageNet| 25.58    | 76.01  | 28.09  | 76.61     | 28.38 | 77.24   | 20|
| ResNet152        | ImageNet| 60.27    | 77.58  | 66.82  | 78.36     | 65.85 | 78.87   | 20|
| ResNext50,32x4   | ImageNet| 25.03    | 77.19  | 27.56  | 78.04     | 27.83 | 78.32   | 10|

Table 2: Testing accuracy (%) on CIFAR10, CIFAR100 and ImageNet 2012. “#P(M)” means the number of parameters (million). The rightmost “r” indicates the reduction ratio of DIANet.
5 Ablation Study

In this section, we conduct ablation experiment to explore how to embed the DIA unit in different neural network structures better and gain a deeper understanding of the role of each component in the DIA unit. All ablation experiments are performed on CIFAR-100 dataset with ResNet. First, we discuss the effect of the reduction ratio \( r \) introduced in Section 3.2. Then the performance of the DIA unit in ResNet of different lengths will be explored. We test the performance of DIA-LSTM with different activation function at the output layer. Also, we study the capacity of the stacked DIA-LSTM with a different number of DIA-LSTM modules.

Reduction ratio. The reduction ratio rate is the only hyperparameter in our DIANet, as mentioned in Section 3.2. Improving the performance with light parameter increment is one of the main characteristics in our model. This part investigates the trade-off between model complexity and its performance. As shown in Table 3, we find out that the testing accuracy of the DIANet declines slightly with the increasing reduction rate. In particular, when \( r = 16 \), the parameter increment is 0.05M comparing with ResNet164. The testing accuracy of the DIANet is 76.50% while that of the original network is 73.43%, which makes the DIA unit has potential in a variety of practical applications, especially those in which small model size is of importance.

| Ratio | #P(M) | top1-acc. |
|-------|-------|-----------|
| 1     | 2.59(±0.86) | 76.88     |
| 4     | 1.95(±0.22)  | 76.67     |
| 8     | 1.84(±0.11)  | 76.42     |
| 16    | 1.78(±0.05)  | 76.50     |

Table 3: Test accuracy (%) with varying reduction ratio on CIFAR100.

CIFAR-100 | SENet | DIANet\((r = 4)\) |
|---------|-------|----------------|
| Depth   | #P(M) | top1-acc. | #P(M) | top1-acc. |
| ResNet83 | 0.99 | 74.67 | 1.11(±0.12) | 75.02     |
| ResNet164 | 1.93 | 75.03 | 1.95(±0.02) | 76.67     |
| ResNet245 | 2.87 | 75.03 | 2.78(±0.09) | 76.79     |
| ResNet407 | 4.74 | 75.54 | 4.45(±0.29) | 76.98     |

Table 4: Test accuracy (%) with models of different depth on CIFAR100.

The depth of the neural network. Generally, in practice, deep DNNs with a large number of parameters do not guarantee sufficient performance improvement since, on the one hand, deeper networks probably contain extreme feature and parameter redundancy [14]. On the other hand, the gradient vanishing problem will be worse and make the DNN hard to be trained [36, 37, 38]. Therefore, designing a new structure of deep neural networks [13, 14, 38, 15, 12, 18] is of great necessity. Since the DIA unit also changes the network topology of the DNN backbones, therefore evaluating the effectiveness of DIANet structure is of great importance. Here we investigate how the depth of the DNN influence the DIANet in two parts: (1) the performance of DIANet compared to SENet of varying depth; (2) the parameter increment of DIANet in DNN. The results in Table 4 show that as the depth of the ResNet increases from 83 layers to 407 layers, the DIANet can achieve higher classification accuracy improvement than the SENet with less number of parameters. Moreover, DIANet83 (for simplicity, 83 denotes the depth of ResNet backbone) can achieve the competitive result as SENet245, and DIANet164 can outperform all the SENet results with at least 1.13% and at most 58.8% parameter reduction. They imply that the DIANet is of higher parameter efficiency than SENet. The results also suggest that: for DIANet, as shown in Figure 2, the deeper network means that the DIA-LSTM module will pass more layers recurrently. The DIA-LSTM can handle the interrelationship between the information of different layers in much deeper DNN and better find out the long distance dependency between layers. Therefore the DIANet can effectively avoid feature redundancy.

| Activation | #DIA-LSTM | top1-acc. |
|------------|------------|-----------|
| sigmoid    | 1          | 76.67     |
| tanh       | 1          | 75.24     |
| sigmoid    | 3          | 75.20     |
| tanh       | 3          | 76.47     |

Table 5: Test accuracy (%) on CIFAR100. The effect of activation function at the output layer in DIA-LSTM and the number of staking DIA-LSTM modules.

Activation function and the number of stacking DIA-LSTM modules. We choose two different activation functions (sigmoid and tanh) in the output layer of DIA-LSTM and two different numbers (one and three) of stacking DIA-LSTM cells to explore the effects of these two factors on classification...
performance. In Table 5, we find that the performance has been significantly improved after replacing \textit{tanh} in the standard LSTM with \textit{sigmoid}. As shown in Figure 3 (Right), this activation function is located in the output layer, which directly changes the effect of memory unit $c_t$ on the output of the output gate.

When we use \textit{sigmoid} in the output layer of DIA-LSTM, an increasing number of stacking DIA-LSTM modules does not necessarily lead to performance improvement but may lead to considerable performance degradation. However, when we choose \textit{tanh}, the situation is different. It suggests that for the DIA unit to be effective, fine structural adjustments are necessary.

6 Analysis

In this section, we study some properties of DIANet, including feature integration and normalization effect. In DIANet, the deeper layers connect the shallower layers via DIA-LSTM module. Firstly, the random forest model [39] is used to visualize how the current layer depends on the preceding layers. Secondly, we study the normalization effect of DIANet by removing the Batch Normalization [40] in the networks. Also, the tests on removal of skip connection and data augment show some positive side-effects of DIANet.

Feature Integration. Here we try to understand the dense connection from the numerical perspective. As shown in Figure 2 and 4, the DIA-LSTM bridges the connections between layers by propagating the information forward through $h_t$ and $c_t$. Moreover, $h_t$ at different layers are also integrating with $h_{t'}$, $1 \leq t' < t$ in DIA-LSTM. Notably, $h_t$ is applied directly to the features in the network at each layer $t$. Therefore the relationship between $h_t$ at different layers somehow reflects connection degree of different layers. We explore the nonlinear relationship between the hidden state $h_t$ of DIA-LSTM and the preceding hidden state $h_{t-1}, h_{t-2}, ..., h_1$, and visualize how the information coming from $h_{t-1}, h_{t-2}, ..., h_1$ contributes to $h_t$. To reveal this relationship, we consider using the random forest to visualize variable importance. The random forest can return the contributions of input variables to the output separately in the form of importance measure, e.g., Gini importance [39]. The computation details of Gini importance can be referred to the Appendix. Take $h_n, 1 \leq n < t$ as input variables and $h_t$ as output variable, we find out the Gini importance of each variable $h_n, 1 \leq n < t$. ResNet164 contains three stages, and each stage consists of 18 layers. We conduct three Gini importance computation to each stage separately. As shown in Figure 5, each row presents the importance of source layers $h_n, 1 \leq n < t$ contributing to the target layer $h_t$. In each sub-graph of Figure 5, the diversity of variable importance distribution indicates that the current layer utilizes the information of the preceding layers. The interaction between shallow and deep layers in the same stage reveals the effect of implicitly dense connection. In particular, taking $h_{17}$ in stage 1 (the last row) as an example, $h_{16}$ or $h_{15}$ does not intuitively provide the most information for $h_{17}$, but $h_5$ does. We conclude that the DIA unit can adaptively integrate information between multiple layers.

Figure 5: Visualization of feature integration for each stage by random forest.

Moreover, in Figure 5 (stage 3), the information interaction with previous layers in stage 3 is more intense and frequent than that of the first two stages. Correspondingly, as shown in Table 6 (Left), in the experiments when we remove the DIA unit in stage 3, the classification accuracy decreases from 76.67 to 75.40. However, when it in stage 1 or 2 is removed, the performance degradation is very similar, falling to 76.27 and 76.25 respectively. Also note that for DIANet, the number of parameters increment in stage 2 is much larger than that of stage 1. It implies that the significant performance degradation after the removal of stage 3 may be not only due to the reduction of the number of parameters but also due to the lack of dense feature integration.
Table 6: (Left) The effect of removal of DIA units in different stage. We test on CIFAR100 with ResNet164. (Right) Test accuracy (%) of the models without data augment.

|       | stage1 | stage2 | stage3 |
|-------|--------|--------|--------|
| #P(M) | 1.94   | 1.90   | 1.78   |
| #P(M) | 0.01   | 0.05   | 0.17   |
| top1-acc. | 76.27 | 76.25 | 75.40 |
| top1-acc. | 0.40  | 0.42  | 1.27   |

| Models | CIFAR-10 | CIFAR-100 |
|--------|----------|-----------|
| ResNet164 | 87.32 | 60.92 |
| SENet    | 88.30   | 62.91    |
| DIANet   | 89.25   | 66.73    |

Table 7: Testing accuracy (%). We train models of different depth without BN on CIFAR-100. “nan” indicates the numerical explosion.

|       | original | SENet | DIANet(k = 16) |
|-------|----------|-------|----------------|
|       | #P(M) | top1-acc. | #P(M) | top1-acc. | #P(M) | top1-acc. |
| ResNet83 | 0.88 | nan | 0.98 | nan | 0.94 | 70.58 |
| ResNet164 | 1.70 | nan | 1.91 | nan | 1.76 | 72.36 |
| ResNet245 | 2.53 | nan | 2.83 | nan | 2.58 | 72.35 |
| ResNet326 | 3.35 | nan | 3.75 | nan | 3.41 | nan |

Normalizations Effect of DIANet. Small changes in shallower hidden layers may be amplified as the information propagates within the deep architecture and sometimes result in a numerical explosion. Batch Normalization (BN) [40] is widely used in the modern deep networks since it stabilizes the training by standardizing the input of each layer. DIA unit re-adjusts the feature maps by channel-wise multiplication, which plays a role of scaling similar to BN. In this part, we empirically claim that DIA unit has a normalization effect. As shown in Table 7, different models trained with varying depth in CIFAR-100, and BNs are removed in these networks to eliminate their normalization effect.

Removal of skip connection. The skip connection has become a necessary structure for training the DNN [33]. Without skip connection, the DNN is hard to train due to the reasons like the gradient vanishing. Figure 6(a-b) shows the training curves if we remove the last one-third of the skip connections at each stage. We can find that the decline of train loss in DIANet is more stable and finally DIANet achieves a smaller training loss. At the same time, DIANet has higher test accuracy than SENet and original ResNet. To some extent, it shows that DIA unit can alleviate the gradient vanishing in deep network training.

Without data augment. Explicit dense connections may help bring more efficient usage of parameters, which makes the neural network less prone to overfit [14]. Although the dense connections in DIA are implicit, the DIANet still shows the ability to reduce overfitting. To verify it, We train the models without data augment to reduce the influence of regularization from data augment. As shown in Table 6 (Right), DIANet achieves lower testing error than ResNet164 and SENet. To some extent, the implicit and dense structure of DIANet may have regularization effect.
7 Conclusion

In this paper, we proposed a Dense-and-Implicit Attention (DIA) unit to enhance the generalization capacity of deep neural networks by recurrently fusing feature attention throughout different layers. Experiments showed that the DIA unit could be universally applied to different network architectures and improve their performance. The DIA unit is also supported by empirical analysis and extensive ablation study. Notably, the DIA unit can be constructed as a global networks module and can implicitly change the topology of existing network backbones.

Acknowledgments

S. Liang and H. Yang gratefully acknowledge the support of National Supercomputing Center (NSCC) SINGAPORE [41] and High Performance Computing (HPC) of National University of Singapore for providing computational resources, and the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research. Sincerely thank Xin Wang from Tsinghua University for providing personal computing resource. H. Yang thanks the support of the start-up grant by the Department of Mathematics at the National University of Singapore, the Ministry of Education in Singapore for the grant MOE2018-T2-2-147.

References

[1] John R Anderson. Cognitive psychology and its implications. Macmillan, 2005.
[2] Kelvin Xu, Jimmy Lei Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37, ICML’15, pages 2048–2057. JMLR.org, 2015.
[3] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. arXiv preprint arXiv:1508.04025, 2015.
[4] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Advances in neural information processing systems, pages 5998–6008, 2017.
[5] Denny Britz, Anna Goldie, Minh-Thang Luong, and Quoc Le. Massive exploration of neural machine translation architectures. arXiv preprint arXiv:1703.03906, 2017.
[6] Jianpeng Cheng, Li Dong, and Mirella Lapata. Long short-term memory-networks for machine reading. EMNLP 2016, 2016.
[7] Antoine Miech, Ivan Laptev, and Josef Sivic. Learnable pooling with context gating for video classification. arXiv preprint arXiv:1706.06905, 2017.
[8] Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. Recurrent models of visual attention. In Advances in neural information processing systems, pages 2204–2212, 2014.
[9] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473, 2014.
[10] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In European Conference on Computer Vision, pages 483–499. Springer, 2016.
[11] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 6077–6086, 2018.
[12] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Andrea Vedaldi. Gather-excite: Exploiting feature context in convolutional neural networks. In Advances in Neural Information Processing Systems, pages 9401–9411, 2018.
[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Computer Vision and Pattern Recognition, 2016.
[14] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. Densely connected convolutional networks. In Computer Vision and Pattern Recognition, 2017.
[15] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 7132–7141, 2018.
[16] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 3–19, 2018.

[17] Jongchan Park, Sanghyun Woo, Joon-Young Lee, and In So Kweon. Bam: Bottleneck attention module. *arXiv preprint arXiv:1807.06514*, 2018.

[18] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7794–7803, 2018.

[19] Xiang Li, Wenhai Wang, Xiaolin Hu, and Jian Yang. Selective kernel networks. 2019.

[20] Yue Cao, Jiarui Xu, Stephen Lin, Fangyun Wei, and Han Hu. Gcnet: Non-local networks meet squeeze-excitation networks and beyond. *arXiv preprint arXiv:1904.11492*, 2019.

[21] Wenhai Wang, Xiang Li, Jian Yang, and Tong Lu. Mixed link networks. *arXiv preprint arXiv:1802.01808*, 2018.

[22] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pages 2017–2025, 2015.

[23] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[24] Bo Zhao, Xiao Wu, Jiashi Feng, Qiang Peng, and Shuicheng Yan. Diversified visual attention networks for fine-grained object classification. *IEEE Transactions on Multimedia*, 19(6):1245–1256, 2017.

[25] Fei Wang, Mengqing Jiang, Chen Qian, Shuo Yang, Cheng Li, Honggang Zhang, Xiaogang Wang, and Xiaou Tang. Residual attention network for image classification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[26] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Training very deep networks. In *Advances in neural information processing systems*, pages 2377–2385, 2015.

[27] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML’15, pages 448–456. JMLR.org, 2015.

[28] The computational work for this article was partially performed on resources of the national supercomputing centre, singapore (https://www.nscc.sg).
A Introduction of Implementation detail

| ResNet164 | PreResNet164 | WRN52-4 | ResNext101-8x32 |
|-----------|--------------|---------|----------------|
| Batch size | 128          | 128     | 128            | 128            |
| Epoch      | 180          | 164     | 200            | 300            |
| Optimizer  | SGD(0.9)     | SGD(0.9)| SGD(0.9)       | SGD(0.9)       |
| depth      | 164          | 164     | 52             | 101            |
| schedule   | 60/120       | 81/122  | 80/120/160     | 150/225        |
| wd         | 1.00E-04     | 1.00E-04| 5.00E-04       | 5.00E-04       |
| gamma      | 0.1          | 0.1     | 0.2            | 0.1            |
| widen-factor | -           | -       | 4              | 4              |
| cardinality | -           | -       | 8              | -              |
| lr         | 0.1          | 0.1     | 0.1            | 0.1            |
| $F_{ext}()$ | GAP         | BN+GAP  | BN+GAP         | GAP            |
| drop       | -            | -       | 0.3            | -              |

Table 8: Implementation detail for CIFAR10/100 image classification. Normalization and standard data augmentation (random cropping and horizontal flipping) are applied to the training data. GAP and BN denote Global Average Pooling and Batch Normalization separately.

| ResNet34 | ResNet50 | ResNet152 | ResNext50-32x4 |
|----------|----------|-----------|----------------|
| Batch size | 256      | 256       | 256            | 256            |
| Epoch     | 120      | 120       | 120            | 120            |
| Optimizer | SGD(0.9) | SGD(0.9)  | SGD(0.9)       | SGD(0.9)       |
| depth     | 34       | 50        | 152            | 50             |
| schedule  | 30/60/90 | 30/60/90  | 30/60/90       | 30/60/90       |
| wd        | 1.00E-04 | 1.00E-04  | 5.00E-04       | 5.00E-04       |
| gamma     | 0.1      | 0.1       | 0.2            | 0.1            |
| lr        | 0.1      | 0.1       | 0.1            | 0.1            |
| $F_{ext}()$ | GAP     | GAP       | GAP            | GAP            |

Table 9: Implementation detail for ImageNet 2012 image classification. Normalization and standard data augmentation (random cropping and horizontal flipping) are applied to the training data. The random cropping of size 224 by 224 is used in these experiments. GAP denote Global Average Pooling.

| Batch size | train batchsize |
|----------------------|------------------|
| Epoch                 | number of total epochs to run |
| Optimizer             | Optimizer |
| depth                 | the depth of the network |
| schedule              | Decrease learning rate at these epochs |
| wd                    | weight decay |
| gamma                 | learning rate is multiplied by gamma on schedule |
| widen-factor          | Widen factor |
| cardinality           | Model cardinality (group) |
| lr                    | initial learning rate |
| $F_{ext}()$           | extract features(Figure 1) |
| drop                  | Dropout ratio |

Table 10: The Additional explanation
B Gini importance

Algorithm 1 Calculate features integration by Gini importance from Random Forest

**Input:** $H$: composed of $h_1, h_2, \ldots, h_t$ from stage $i$;
#The size of $H$ is $(b_z \times c_z \times f_z)$
#$b_z$ denotes the batch size of $h_t$
#$c_z$ denotes the number of the feature maps’ channel in current stage
# $f_z$ denotes the number of layers in current stage

**Output:** The hotmap $G$ about the features integration for stage $i$;

1: initial $G = \emptyset$
2: for $j = 1$ to $f_z - 1$ do
3:   $x \leftarrow [h_1, h_2, \ldots, h_{j-1}]$;
4:   $y \leftarrow [h_j]$;
5:   $x \leftarrow x$.reshape($b_z \times (f_z - j) \times c_z$);
6:   $RF \leftarrow $ RandomForestRegressor();
7:   $RF.fit(x, y)$;
8:   $\text{Gini importances} \leftarrow RF.feature_importances_;$
#The length of Gini_importance is $(f_z - j) \times c_z$
9:   $res \leftarrow \emptyset$;
10:  $s \leftarrow 0$;
11:  $cnt \leftarrow 0$;
12:  for $k = 0$ to $(f_z - j)$ do
13:     $s \leftarrow s + \text{Gini_importance}(k)$;
14:     $cnt \leftarrow cnt + 1$;
15:     if $cnt == c_z - 1$ then
16:        $res.add(s);$
17:        $s \leftarrow 0$;
18:        $cnt \leftarrow 0$;
19:     end if
20:     $G.add(res/max(res));$
21: end for
22: end for

C Number of parameter of LSTM

Suppose the input $y_t$ is of size $N$ and the hidden state vector $h_{t-1}$ is also of size $N$.

**Standard LSTM** As shown in Figure (3) (Left), in the standard LSTM, there requires 4 linear transformation to control the information flow with input $y_t$ and $h_{t-1}$ respectively. The output size is set to be $N$. To simplify the calculation, the bias is omitted. Therefore, for the $y_t$, the number of parameters of 4 linear transformation is equal to $4 \times n \times n$. Similarly, the number parameters of 4 linear transformation with input $h_{t-1}$ is equal to $4 \times n \times n$. The total of parameters equals to $8n^2$.

**DIA-LSTM** As shown in Figure (3) (Right), there is a linear transformation to reduce the dimension at the beginning. The dimension of input $y_t$ will reduce from $N$ to $N/r$ after the first linear transformation. The number of parameters for the linear transformation is equal to $n \times n/r$. Then the output will be passed into 4 linear transformation same as the standard LSTM. the number parameters of 4 linear transformation is equal to $4 \times n/r \times n$. Therefore, for input $y_t$ and reduction ratio $r$, the number of parameters is equal to $5n^2/r$. Similarly, the number of parameters with input $h_{t-1}$ is the same as that concerning $y_t$. The total of parameters equals to $10n^2/r$. 

12