Rethinking Spatial Dimensions of Vision Transformers

Byeongho Heo[1] Sangdoo Yun[1] Dongyoon Han[1] Sanghyuk Chun[1] Junsuk Choe[2*] Seong Joon Oh[1]

[1] NAVER AI Lab [2] Department of Computer Science and Engineering, Sogang University

Abstract

Vision Transformer (ViT) extends the application range of transformers from language processing to computer vision tasks as being an alternative architecture against the existing convolutional neural networks (CNN). Since the transformer-based architecture has been innovative for computer vision modeling, the design convention towards an effective architecture has been less studied yet. From the successful design principles of CNN, we investigate the role of spatial dimension conversion and its effectiveness on transformer-based architecture. We particularly attend to the dimension reduction principle of CNNs; as the depth increases, a conventional CNN increases channel dimension and decreases spatial dimensions. We empirically show that such a spatial dimension reduction is beneficial to a transformer architecture as well, and propose a novel Pooling-based Vision Transformer (PiT) upon the original ViT model. We show that PiT achieves the improved model capability and generalization performance against ViT. Throughout the extensive experiments, we further show PiT outperforms the baseline on several tasks such as image classification, object detection, and robustness evaluation. Source codes and ImageNet models are available at `https://github.com/naver-ai/pit`.

1. Introduction

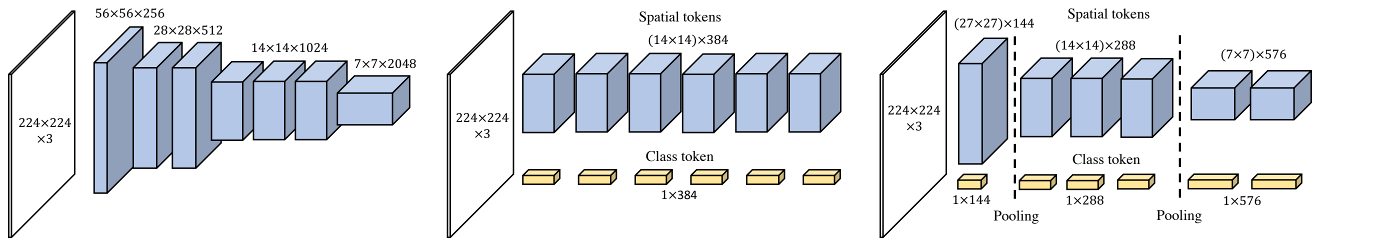
The architectures based on the self-attention mechanism have achieved great success in the field of Natural Language Processing (NLP) [34]. There have been attempts to utilize the self-attention mechanism in computer vision. Non-local networks [37] and DETR [4] are representative works, showing that the self-attention mechanism is also effective in video classification and object detection tasks, respectively. Recently, Vision Transformer (ViT) [9], a transformer architecture consisting of self-attention layers, has been proposed to compete with ResNet [13], and shows that it can achieve the best performance without convolution operation on ImageNet [8]. As a result, a new direction of network architectures based on self-attention mechanism, not convolution operation, has emerged in computer vision.

ViT is quite different from convolutional neural networks (CNN). Input images are divided into 16×16 patches and fed to the transformer network; except for the first embedding layer, there is no convolution operation in ViT, and the position interactions occur only through the self-attention layers. While CNNs have restricted spatial interactions, ViT allows all the positions in an image to interact through transformer layers. Although ViT is an innovative architecture and has proven its powerful image recognition ability, it follows the transformer architecture in NLP [34] without any changes. Some essential design principles of CNNs, which have proved to be effective in the computer vision domain over the past decade, are not sufficiently reflected. We thus revisit the design principles of CNN architectures and investigate their efficacy when applied to ViT architectures.

CNNs start with a feature of large spatial sizes and a small channel size and gradually increase the channel size while decreasing the spatial size. This dimension conversion is indispensable due to the layer called spatial pooling. Modern CNN architectures, including AlexNet [21], ResNet [13], and EfficientNet [32], follow this design principle. The pooling layer is deeply related to the receptive field size of each layer. Some studies [6, 26, 5] show that the pooling layer contributes to the expressiveness and generalization performance of the network. However, unlike the CNNs, ViT does not use a pooling layer and uses the same spatial dimension for all layers.

First, we verify the advantages of dimensions configurations on CNNs. Our experiments show that ResNet-style dimensions improve the model capability and generalization performance of ResNet. To extend the advantages to ViT, we propose a Pooling-based Vision Transformer (PiT). PiT is a transformer architecture combined with a newly designed pooling layer. It enables the spatial size reduction in the ViT structure as in ResNet. We also investigate the benefits of PiT compared to ViT and confirm that ResNet-style dimension setting also improves the performance of ViT. Finally, to analyze the effect of PiT compared to ViT, we

[*] Work done as a research scientist at NAVER AI Lab.



(a) ResNet-50

(b) ViT-S/16

(c) PiT-S

Figure 1. **Schematic illustration of dimension configurations of networks.** We visualize ResNet50 [13], Vision Transformer (ViT) [9], and our Pooling-based Vision Transformer (PiT); (a) ResNet50 gradually downsamples the features from the input to the output; (b) ViT does not change the spatial dimensions; (c) PiT involves ResNet style spatial dimension into ViT.

analyze the attention matrix of transformer block with entropy and average distance measure. The analysis shows the attention patterns inside layers of ViT and PiT, and helps to understand the inner mechanism of ViT and PiT.

We verify that PiT improves performances over ViT on various tasks. On ImageNet classification, PiT and outperforms ViT at various scales and training environments. Additionally, we have compared the performance of PiT with various convolutional architectures and have specified the scale at which the transformer architecture outperforms the CNN. We further measure the performance of PiT as a backbone for object detection. ViT- and PiT-based deformable DETR [44] are trained on the COCO 2017 dataset [24] and the result shows that PiT is even better than ViT as a backbone architecture for a task other than image classification. Finally, we verify the performance of PiT in various environments through the robustness benchmark.

2. Related works

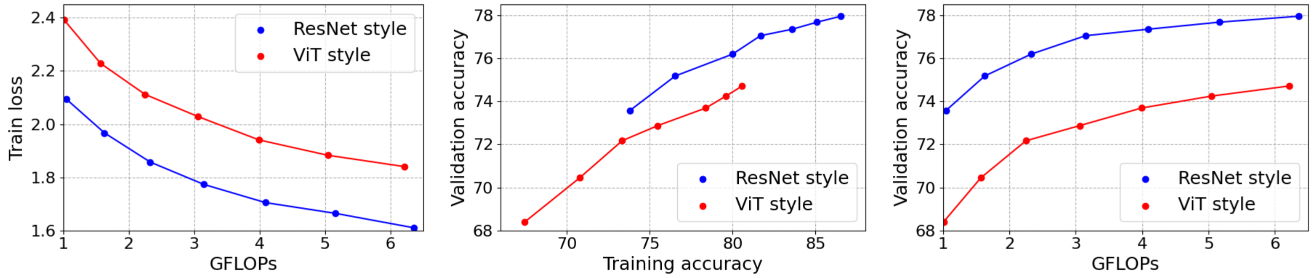
2.1. Dimension configuration of CNN

Dimension conversion can be found in AlexNet [21], which is one of the earliest convolutional networks in computer vision. AlexNet uses three max-pooling layers. In the max-pooling layer, the spatial size of the feature is reduced by half, and the channel size is increased by the convolution after the max-pooling. VGGnet [30] uses 5 spatial resolutions using 5 max-pooling. In the pooling layer, the spatial size is reduced by half and the channel size is doubled. GoogLeNet [31] also used the pooling layer. ResNet [13] performed spatial size reduction using the convolution layer of stride 2 instead of max pooling. It is an improvement in the spatial reduction method. The convolution layer of stride 2 is also used as a pooling method in recent architectures (EfficietNet [32], MobileNet [29, 19]). PyramidNet [11] pointed out that the channel increase occurs only in the pooling layer and proposed a method to gradually increase the channel size in layers other than the pooling layer. ReXNet [12] reported that the channel configuration of the network has a significant influence on the network performance. In summary, most convolution networks use a dimension configuration with spatial reduction.

2.2. Self-attention mechanism

Transformer architecture [34] significantly increased the performance of the NLP task with the self-attention mechanism. Funnel Transformer [7] improves the transformer architecture by reducing tokens by a pooling layer and skip-connection. However, because of the basic difference between the architecture of NLP and computer vision, the method of applying to pool is different from our method. Some studies are conducted to utilize the transformer architecture to the backbone network for computer vision tasks. Non-local network [37] adds a few self-attention layers to CNN backbone, and it shows that the self-attention mechanism can be used in CNN. [28] replaced 3×3 convolution of ResNet to local self-attention layer. [36] used an attention layer for each spatial axis. [2] enables self-attention of the entire spatial map by reducing the computation of the attention mechanism. Most of these methods replace 3x3 convolution with self-attention or adds a few self-attention layers. Therefore, the basic structure of ResNet is inherited, that is, it has the convolution of stride 2 as ResNet, resulting in a network having a dimension configuration of ResNet.

Only the vision transformer uses a structure that uses the same spatial size in all layers. Although ViT did not follow the conventions of ResNet, it contains many valuable new components in the network architecture. In ViT, layer normalization is applied for each spatial token. Therefore, layer normalization of ViT is closer to positional normalization [22] than a layer norm of convolutional neural network [1, 39]. Although it overlaps with the lambda network [2], it is not common to use global attention through all blocks of the network. The use of class tokens instead of global average pooling is also new, and it has been reported that separating tokens increases the efficiency of distillation [33]. In addition, the layer configuration, the skip-connection position, and the normalization position of the Transformer are also different from ResNet. Therefore, our study gives a direction to the new architecture.



(a) Model capability

(b) Generalization performance

(c) Model performance

Figure 2. **Effects of the spatial dimensions in ResNet50 [13].** We verify the effect of the spatial dimension with ResNet50. As shown in the figures, ResNet-style is better than ViT-style in the model capability, generalization performance, and model performance.

3. Revisiting spatial dimensions

In order to introduce dimension conversion to ViT, we investigate spatial dimensions in network architectures. First, we verify the benefits of dimension configuration in ResNet architecture. Although dimension conversion has been widely used for most convolutional architectures, its effectiveness is rarely verified. Based on the findings, we propose a Pooling-based Vision Transformer (PiT) that applies the ResNet-style dimension to ViT. We propose a new pooling layer for transformer architecture and design ViT with the new pooling layer (PiT). With PiT models, we verify whether the ResNet-style dimension brings advantages to ViT. In addition, we analyze the attention matrix of the self-attention block of ViT to investigate the effect of PiT in the transformer mechanism. Finally, we introduce PiT architectures corresponding to various scales of ViT.

3.1. Dimension setting of CNN

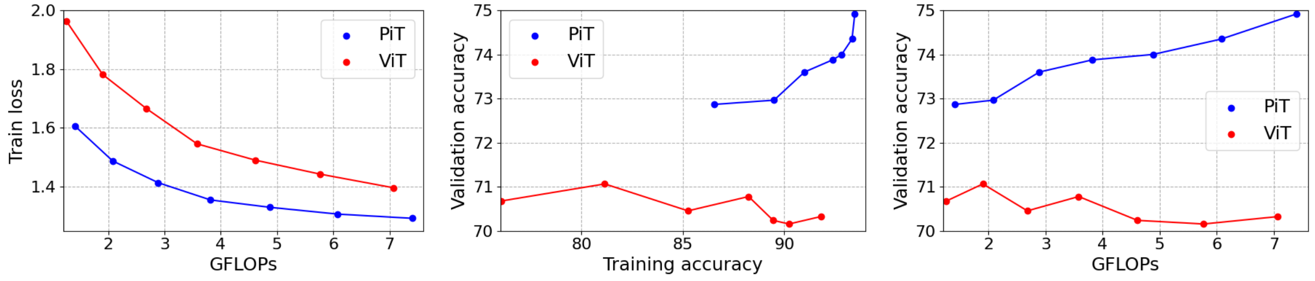
As shown in Figure 1 (a), most convolutional architectures reduce the spatial dimension while increases the channel dimension. In ResNet50, a stem layer reduces the spatial size of an image to 56×56. After several layer blocks, Convolution layers with stride 2 reduce the spatial dimension by half and double the channel dimension. The spatial reduction using a convolution layer with stride 2 is a frequently used method in recent architectures [32, 29, 19, 12]. We conduct an experiment to analyze the performance difference according to the presence or absence of the spatial reduction layer in a convolutional architecture. ResNet50, one of the most widely used networks in ImageNet, is used for architecture and is trained over 100 epochs without complex training techniques. For ResNet with ViT style dimension, we use the stem layer of ViT to reduce the feature to 14×14 spatial dimensions while reducing the spatial information loss in the stem layer. We also remove the spatial reduction layers of ResNet to maintain the initial feature dimensions for all layers like ViT. We measured the performance for several sizes by changing the channel size of ResNet.

First, we measured the relation between FLOPs and training loss of ResNet with ResNet-style or ViT-style dimension configuration. As shown in Figure 2 (a), ResNet (ResNet-style) shows lower training loss over the same computation costs (FLOPs). It implies that ResNet-style dimensions increase the capability of architecture. Next, we analyzed the relation between training and validation accuracy, which represents the generalization performance of architecture. As shown in Figure 2 (b), ResNet (ResNet-style) achieves higher validation accuracy than ResNet (ViT-style). Therefore, ResNet-style dimension configuration is also helpful for generalization performance. In summary, ResNet-style dimension improves the model capability and generalization performance of the architecture and consequently brings a significant improvement in validation accuracy as shown in Figure 2 (c).

3.2. Pooling-based Vision Transformer (PiT)

Vision Transformer (ViT) performs network operations based on self-attention, not convolution operations. In the self-attention mechanism, the similarity between all locations is used for spatial interaction. Figure 1 (b) shows the dimension structure of this ViT. Similar to the stem layer of CNN, ViT divides the image by patch at the first embedding layer and embedding it to tokens. Basically, the structure does not include a spatial reduction layer and keeps the same number of spatial tokens overall layer of the network. Although the self-attention operation is not limited by spatial distance, the size of the spatial area participating in attention is affected by the spatial size of the feature. Therefore, in order to adjust the dimension configuration like ResNet, a spatial reduction layer is also required in ViT.

To utilize the advantages of the dimension configuration to ViT, we propose a new architecture called Pooling-based Vision Transformer (PiT). First, we designed a pooling layer for ViT. Our pooling layer is shown in Figure 4. Since ViT handles neuron responses in the form of 2D-matrix rather than 3D-tensor, the pooling layer should sep-



(a) Model capability

(b) Generalization performance

(c) Model performance

Figure 3. **Effects of the spatial dimensions in vision transformer (ViT) [9].** We compare our Pooling-based Vision Transformer (PiT) with original ViT at various aspects. PiT outperforms ViT in capability, generalization performance, and model performance.

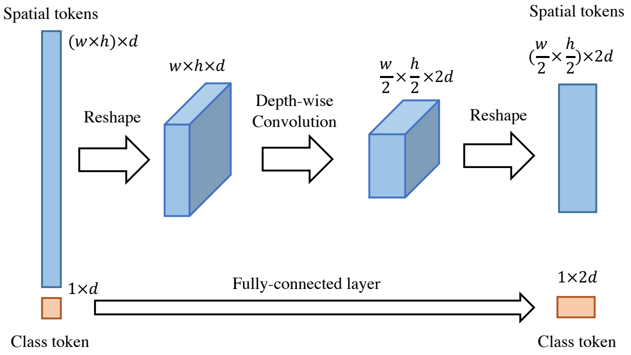


Figure 4. **Pooling layer of PiT architecture.** PiT uses the pooling layer based on depth-wise convolution to achieve channel multiplication and spatial reduction with small parameters.

arate spatial tokens and reshape them into 3D-tensor with spatial structure. After reshaping, spatial size reduction and channel increase are performed by depth-wise convolution. And, the responses are reshaped into a 2D matrix for the computation of transformer blocks. In ViT, there are parts that do not correspond to the spatial structure, such as a class token or distillation token [33]. For these parts, the pooling layer uses an additional fully-connected layer to adjust the channel size to match the spatial tokens. Our pooling layer enables spatial reduction on ViT and is used for our PiT architecture as shown in Figure 1 (c). PiT includes two pooling layers which make three spatial scales.

Using PiT architecture, we performed an experiment to verify the effect of PiT compared to ViT. The experiment setting is the same as the ResNet experiment. Figure 3 (a) represents the model capability of ViT and PiT. At the same computation cost, PiT has a lower train loss than ViT. Using the spatial reduction layers in ViT also improves the capability of architecture. The comparison between training accuracy and validation accuracy shows a significant difference. As shown in Figure 3 (b), ViT does not improve validation accuracy even if training accuracy increases. On the other hand, in the case of PiT, validation accuracy increases as training accuracy increases. The big difference in generalization performance causes the performance difference between PiT and ViT as shown in Figure 3 (c). The phenomenon that ViT does not increase performance even when FLOPs increase in ImageNet is reported in ViT paper [9]. In the training data of ImageNet scale, ViT shows poor generalization performance, and PiT alleviates this. So, we believe that the spatial reduction layer is also necessary for the generalization of ViT. Using the training trick is a way to improve the generalization performance of ViT in ImageNet. The combination of training tricks and PiT is covered in the experiment section.

3.3. Attention analysis

We analyze the transformer networks with measures on attention matrix [35]. We denotes $\alpha_{i,j}$ as (i,j) component of attention matrix $A \in \mathbb{R}^{M\times N}$. Note that attention values after soft-max layer is used, i.e. $\sum_i \alpha_{i,j} = 1$. The attention entropy is defined as

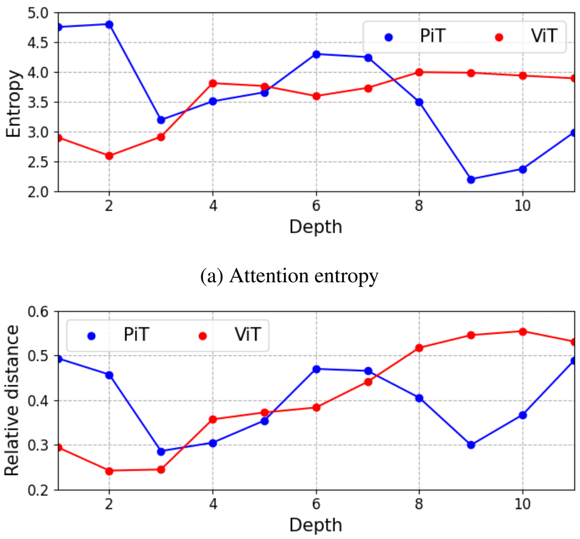
$$\text{Entropy} = -\frac{1}{N}\sum_j^N \sum_i \alpha_{i,j} \log \alpha_{i,j}. \tag{1}$$

The entropy shows the spread and concentration degree of an attention interaction. A small entropy indicates a concentrated interaction, and a large entropy indicates a spread interaction. We also measure an attention distance,

$$\text{Distance} = \frac{1}{N}\sum_j^N \sum_i \alpha_{i,j} \|\boldsymbol{p}_i - \boldsymbol{p}_j\|_1. \tag{2}$$

$\boldsymbol{p}_i$ represents relative spatial location of i-th token $(x_i/W, y_i/H)$ for feature map $F \in \mathbb{R}^{H\times W\times C}$. So, the attention distance shows a relative ratio compared to the overall feature size, which enables comparison between the different sizes of features. We analyze transformer-based models (ViT-S [33] and PiT-S) and values are measured overall validation images and are averaged over all heads of each layer. Our analysis is only conducted for the spatial tokens rather than the class token following the previous study [35]. We also skip the attention of the last transformer block since the spatial tokens of the last attention are independent of the network outputs.

The results are shown in Figure 5. In ViT, the entropy and the distance increase as the layer become deeper. It implies that the interaction of ViT is concentrated to close tokens at the shallow layers and the interaction is spread in a wide range of tokens at the deep layers. The entropy and distance pattern of ViT is similar to the pattern of transformer in the language domain [35]. PiT changes the patterns with the spatial dimension setting. At shallow layers (1-2 layers), large spatial size increases the entropy and distance. On the other hand, the entropy and distance are decreased at deep layers (9-11 layers) due to the small spatial size. In short, the pooling layer of PiT spreads the interaction in the shallow layers and concentrates the interaction in the deep layers. In contrast to discrete word inputs of the language domain, the vision domain uses image-patch inputs which require pre-processing operations such as filtering, contrast, and brightness calibration. In shallow layers, the spread interaction of PiT is close to the pre-processing than the concentrated interaction of ViT. Also, compared to language models, image recognition has relatively low output complexity. So, in deep layers, concentrated interaction might be enough. There are significant differences between the vision and the language domain, and we believe that the attention of PiT is suitable for image recognition backbone.



(a) Attention entropy

(b) Spatial distance of interaction

Figure 5. **Attention analysis.** We investigate the attention matrix of the self-attention layer. Figure (a) shows the entropy and figure (b) shows the interaction distance. PiT increases the entropy and the distance in shallow layers and decreases in deep layers.

3.4. Architecture design

The architectures proposed in ViT paper [9] aimed at datasets larger than ImageNet. These architectures (ViT-Large, ViT-Huge) have an extremely large scale than general ImageNet networks, so it is not easy to compare them with other networks. So, following the previous study [33] of Vision Transformer on ImageNet, we design the PiT at a scale similar to the small-scale ViT architectures (ViT-Base, ViT-Small, ViT-Tiny). In the DeiT paper [33], ViT-Small and ViT-Tiny are named DeiT-S and DeiT-Ti, but to avoid confusion due to the model name change, we use ViT for all models. Corresponding to the three scales of ViT (tiny, small, and base), we design four scales of PiT (tiny, extra small, small, and base). Detail architectures are described in Table 1. For convenience, we abbreviate the model names: Tiny - Ti, eXtra Small - XS, Small - S, Base - B FLOPs and spatial size were measured based on 224×224 image. Since PiT uses a larger spatial size than ViT, we reduce the stride size of the embedding layer to 8, while patch-size is 16 as ViT. Two pooling layers are used for PiT, and the channel increase is implemented as increasing the number of heads of multi-head attention. We design PiT to have a similar depth to ViT, and adjust the channels and the heads to have smaller FLOPs, parameter size, and GPU latency than those of ViT. We clarify that PiT is not designed with large-scale parameter search such as NAS [25, 3], so PiT can be further improved through a network architecture search.

Network	Spatial size	# of blocks	# of heads	Channel size	FLOPs
ViT-Ti [33]	14 x 14	12	3	192	1.3B
PiT-Ti	27 x 27 14 x 14 7 x 7	2 6 4	2 4 8	64 128 256	0.7B
PiT-XS	27 x 27 14 x 14 7 x 7	2 6 4	2 4 8	96 192 384	1.4B
ViT-S [33]	14 x 14	12	6	384	4.6B
PiT-S	27 x 27 14 x 14 7 x 7	2 6 4	3 6 12	144 288 576	2.9B
ViT-B [9]	14 x 14	12	12	768	17.6B
PiT-B	31 x 31 16 x 16 8 x 8	3 6 4	4 8 16	256 512 1024	12.5B

Table 1. **Architecture configuration.** The table shows spatial sizes, number of blocks, number of heads, channel size, and FLOPs of ViT and PiT. The structure of PiT is designed to be as similar as possible to ViT and to have less GPU latency.

Architecture	FLOPs	# of params	Throughput (imgs/sec)	Vanilla	+CutMix [41]	+DeiT [33]	+Distill⚗ [33]
ViT-Ti [33]	1.3 B	5.7 M	2564	68.7%	68.5%	72.2%	74.5%
PiT-Ti	0.7 B	4.9 M	3030	71.3%	72.6%	73.0%	74.6%
PiT-XS	1.4 B	10.6 M	2128	72.4%	76.8%	78.1%	79.1%
ViT-S [33]	4.6 B	22.1 M	980	68.7%	76.5%	79.8%	81.2%
PiT-S	2.9 B	23.5 M	1266	73.3%	79.0%	80.9%	81.9%
ViT-B [9]	17.6 B	86.6 M	303	69.3%	75.3%	81.8%	83.4%
PiT-B	12.5 B	73.8 M	348	76.1%	79.9%	82.0%	84.0%

Table 2. **ImageNet performance comparison with ViT.** We compare the performances of ViT and PiT with some training techniques on ImageNet dataset. PiT shows better performance with low computation compared to ViT.

4. Experiments

We verified the performance of PiT through various experiments. First, we compared PiT at various scales with ViT in various training environments of ImageNet training. And, we extended the ImageNet comparison to architectures other than Transformer. In particular, we focus on the comparison of the performance of ResNet and PiT, and investigate whether PiT can beat ResNet. We also applied PiT to an object detector based on deformable DETR [44], and compared the performance as a backbone architecture for object detection. To analyze PiT in various views, we evaluated the performance of PiT on robustness benchmarks.

4.1. ImageNet classification

We compared the performance of PiT models of Table 1 with corresponding ViT models. To clarify the computation time and size of the network, we measured FLOPs, the number of parameters, and GPU throughput (images/sec) of each network. The GPU throughput was measured on NVIDIA V100 single GPU with 128 batch-size. We trained the network using four representative training environments. The first is a vanilla setting that trains the network without complicated training techniques. The vanilla setting has the lowest performance due to the lack of techniques to help generalization performance and also used for the previous experiments in Figure 2, 3. The second is training with CutMix [41] data augmentation. Although only data augmentation has changed, it shows significantly better performance than the vanilla setting. The third is the DeiT [33] setting, which is a compilation of training techniques to train ViT on ImageNet-1k [8]. DeiT setting includes various training techniques and parameter tuning, and we used the same training setting through the official open-source code. However, in the case of Repeated Augment [18], we confirmed that it had a negative effect in a small model, and it was used only for Base models. The last is a DeiT setting with knowledge distillation. The distillation setting is reported as the best performance setting in DeiT [33] paper. The network uses an additional distillation token and is trained with distillation loss [17] using RegNetY-16GF [27] as a teacher network. We used AdamP [16] optimizer for all settings, and the learning rate, weight decay, and warmup were set equal to DeiT [33] paper. We train models over 100 epochs for Vanilla and CutMix settings, and 300 epochs for DeiT and Distill⚗ settings.

The results are shown in Table 2. Comparing the PiT and ViT of the same name, the PiT has fewer FLOPs and faster speed than ViT. Nevertheless, PiT shows higher performance than ViT. In the case of vanilla and CutMix settings, where a few training techniques are applied, the performance of PiT is superior to the performance of ViT. Even in the case of a DeiT and distill settings, PiT shows comparable or better performance to ViT. Therefore, PiT can be seen as a better architecture than ViT in terms of performance and computation. The generalization performance issue of ViT in Figure 3 can also be observed in this experiment. Like ViT-S in the Vanilla setting and ViT-B in the CutMix setting, ViT often shows no increase in performance even when the model size increases. On the other hand, the performance of PiT increases according to the model size in all training settings. it seems that the generalization performance problem of ViT is alleviated by the pooling layers.

We compared the performance of PiT with the convolutional networks. In the previous experiment, we performed the comparison in the same training setting using the similarity of architecture. However, when comparing various architectures, it is infeasible to unify with a setting that works well for all architectures. Therefore, we performed the comparison based on the best performance reported for each architecture. But, it was limited to the model trained using only ImageNet images. When the paper that proposed the architecture and the paper that reported the best performance was different, we cite both papers. When the architecture is different, the comparison of FLOPs often fails to reflect the actual throughput. Therefore, we re-measured the GPU throughput and number of params on a single V100 GPU and compared the top-1 accuracy for the performance index. Table 3 shows the comparison result. In the case of the PiT-B scale, the transformer-based archi-

Network	# of params	Throughput (imgs/sec)	Accuracy
ResNet18 [13, 42]	11.7M	4545	72.5%
MobileNetV2 [29]	3.5M	3846	72.0%
MobileNetV3 [19]	5.5M	3846	75.2%
EfficientNet-B0 [32]	5.3M	2857	77.1%
ViT-Ti [33]	5.7M	2564	72.2%
PiT-Ti	4.9M	3030	73.0%
ViT-Ti⚗ [33]	5.7M	2564	74.5%
PiT-Ti⚗	4.9M	3030	74.6%
ResNet34 [13, 38]	21.8M	2631	75.1%
ResNet34D [14, 38]	21.8M	2325	77.1%
EfficientNet-B1 [32]	7.8M	1754	79.1%
PiT-XS	10.6M	2128	78.1%
PiT-XS⚗	10.6M	2128	79.1%
ResNet50 [13, 42]	25.6M	1266	80.2%
ResNet101 [13, 42]	44.6M	757	81.6%
ResNet50D [14, 38]	25.6M	1176	80.5%
EfficientNet-B2 [32]	9.2M	1333	80.1%
EfficientNet-B3 [32]	12.2M	806	81.6%
RegNetY-4GF [27]	20.6M	1136	79.4%
ResNeSt50 [43]	27.5M	877	81.1%
ViT-S [33]	22.1M	980	79.8%
PiT-S	23.5M	1266	80.9%
ViT-S⚗ [33]	22.1M	980	81.2%
PiT-S⚗	23.5M	1266	81.9%
ResNet152 [13, 42]	60.2M	420	81.9%
ResNet101D [14, 38]	44.6M	354	83.0%
ResNet152D [14, 38]	60.2M	251	83.7%
EfficientNet-B4 [32]	19.3M	368	82.9%
RegNetY-16GF [27]	83.6M	352	80.4%
ResNeSt101 [43]	48.3M	398	83.0%
ViT-B [9, 33]	86.6M	303	81.8%
PiT-B	73.8M	348	82.0%
ViT-B⚗ [9, 33]	86.6M	303	83.4%
PiT-B⚗	73.8M	348	84.0%

Table 3. **ImageNet performance.** We compare our PiT-(Ti, XS, S, and B) models with the counterparts which have a similar number of parameters. ⚗ means a model trained with distillation [33].

tecture (ViT-B, PiT-B) outperforms the convolutional architecture. Even in the PiT-S scale, PiT-S shows superior performance than convolutional architecture (ResNet50) or outperforms in throughput (EfficientNet-b3). However, in the case of PiT-Ti, the performance of convolutional architectures such as ResNet34 [13], MobileNetV3 [19], and EfficientNet-b0 [32] outperforms ViT-Ti and PiT-Ti. Overall, the transformer architecture shows better performance than the convolutional architecture at the scale of ResNet50 or higher, but it is weak at a small scale. Creating a lightweight transformer architecture such as MobileNet is one of the future works of ViT research.

Setting	Architecture	Throughput (imgs/sec)	Accuracy
Long training (1000 epochs)	ViT-Ti⚗ [33]	2564	76.6%
	PiT-Ti⚗	3030	76.4%
	PiT-XS⚗	2128	80.6%
	ViT-S⚗ [33]	980	82.6%
	PiT-S⚗	1266	82.7%
	ViT-B⚗ [33]	303	84.2%
	PiT-B⚗	348	84.5%
Large resolution (384×384)	ViT-B [33]	91	83.1%
	PiT-B	82	83.0%
	ViT-B⚗ [33]	91	84.5%
	PiT-B⚗	82	84.6%

Table 4. **Extended training settings.** We compare the performance of PiT with ViT for long training (1000 epochs) and fine-tune on large resolution (384×384)

Backbone	Avg. Precision at IOU			Params.	Latency (ms / img)
	AP	AP_{50}	AP_{75}		
ResNet50 [13]	41.5	60.5	44.3	41.0 M	49.7
ViT-S [33]	36.9	57.0	38.0	34.9 M	55.2
PiT-S	39.4	58.8	41.5	36.6 M	46.9

Table 5. **COCO detection performance based on Deformable DETR [44].** We evaluate the performance of PiT as a pretrained backbone for object detection.

Additionally, we conduct experiments on two extended training schemes: long training and fine-tune on large resolution. Table 4 shows the results. As shown in the previous study [33], the performance of ViT is significantly improved on the long training scheme (1000 epochs). So, we validate PiT on the long training scheme. As shown in Table 4, PiT models show comparable performance with ViT models on the long training scheme. Although the performance improvement is reduced than the Distill⚗ setting, PiTs still outperform ViT counterparts in throughput. Fine-tuning on large resolution (384 × 384) is a famous method to train a large ViT model with small computation. In the large resolution setting, PiT has comparable performance with ViT, but, worse than ViT on throughput. It implies that PiT is designed for 224 × 224 and the design is not compatible for the large resolution. However, we believe that PiT can outperform ViT with a new layer design for 384 × 384.

4.2. Object detection

We validate PiT through object detection on COCO dataset [24] in Deformable-DETR [44]. We train the detectors with different backbones including ResNet50, ViT-S, and our PiT-S. We follow the training setup of the original paper [44] except for the image resolution. Since the original image resolution is too large for transformer-based backbones, we halve the image resolution for training and

	Standard	Occ	IN-A [15]	BGC [40]	FGSM [10]
PiT-S	80.8	74.6	21.7	21.0	29.5
ViT-S [33]	79.8	73.0	19.1	17.6	27.2
ResNet50 [13]	76.0	52.2	0.0	22.3	7.1
ResNet50† [38]	79.0	67.1	5.4	32.7	24.7

Table 6. **ImageNet robustness benchmarks.** We compare three comparable architectures, PiT-B, ViT-S, and ResNet50 on various ImageNet robustness benchmarks, including center occlusion (Occ), ImageNet-A (IN-A), background challenge (BGC), and fast sign gradient method (FGSM) attack. We evaluate two ResNet50 models from the official PyTorch repository, and the well-optimized implementation [38], denoted as †.

test of all backbones. We use bounding box refinement and a two-stage scheme for the best performance [44]. For multi-scale features for ViT-S, we use features at the 2nd, 8th, and 12th layers following the position of pooling layers on PiT. All detectors are trained for 50 epochs and the learning rate is dropped by factor 1/10 at 40 epochs.

Table 5 shows the measured AP score on val2017. The detector based on PiT-S outperforms the detector with ViT-S. It shows that the pooling layer of PiT is effective not only for ImageNet classification but also for pretrained backbone for object detection. We measured single image latency with a random noise image at resolution 600×400 PiT based detector has lower latency than detector based on ResNet50 or ViT-S. Although PiT detector cannot beat the performance of the ResNet50 detector, PiT detector has better latency, and improvement over ViT-S is significant. Additional investigation on the training settings for PiT based detectors would improve the performance of the PiT detector.

4.3. Robustness benchmarks

In this subsection, we investigate the effectiveness of the proposed architecture in terms of robustness against input changes. We presume that the existing ViT design concept, which keeps the spatial dimension from the input layer to the last layer, has two conceptual limitations: *Lack of background robustness* and *sensitivity to the local discriminative visual features*. We, therefore, presume that PiT, our new design choice with the pooling mechanism, performs better than ViT for the background robustness benchmarks and the local discriminative sensitivity benchmarks.

We employ four different robustness benchmarks. **Occlusion benchmark** measures the ImageNet validation accuracy where the center 112×112 patch of the images is zero-ed out. This benchmark measures whether a model only focuses on a small discriminative visual feature or not. **ImageNet-A** (IN-A) is a dataset constructed by collecting the failure cases of ResNet50 from the web [15] where the collected images contain unusual backgrounds or objects with very small size [23]. From this benchmark, we can infer how a model is less sensitive to unusual backgrounds or object size changes. However, since IN-A is constructed by collecting images (queried by 200 ImageNet subclasses) where ResNet50 predicts a wrong label, this dataset can be biased towards ResNet50 features. We, therefore, employ **background challenge** (BGC) benchmark [40] to explore the explicit background robustness. The BGC dataset consists of two parts, foregrounds, and backgrounds. This benchmark measures the model validation accuracy while keeping the foreground but adversarially changing the background from the other image. Since BGC dataset is built upon nine subclasses of ImageNet, the baseline random chance is 11.1%. Lastly, we tested adversarial attack robustness using the fast gradient sign method (FGSM) [10].

Table 6 shows the results. First, we observe that PiT shows better performances than ViT in all robustness benchmarks, despite they show comparable performances in the standard ImageNet benchmark (80.8 vs. 79.8). It supports that our dimension design makes the model less sensitive to the backgrounds and the local discriminative features. Also, we found that the performance drops for occluded samples by ResNet50 are much dramatic than PiT; 80.8 → 74.6, 5% drops for PiT, 79.0 → 67.1, 15% drops for ResNet50. This implies that ResNet50 focuses more on the local discriminative areas, by the nature of convolutional operations. Interestingly, in Table 6, ResNet50 outperforms ViT variants in the background challenge dataset (32.7 vs. 21.0). This implies that the self-attention mechanism unintentionally attends more backgrounds comparing to ResNet design choice. Overcoming this potential drawback of vision transformers will be an interesting research direction.

5. Conclusion

In this paper, we have shown that the design principle widely used in CNNs - the spatial dimensional transformation performed by pooling or convolution with strides, is not considered in transformer-based architectures such as ViT; ultimately affects the model performance. We have first studied with ResNet and found that the transformation in respect of the spatial dimension increases the computational efficiency and the generalization ability. To leverage the benefits in ViT, we propose a PiT that incorporates a pooling layer into Vit, and PiT shows that these advantages can be well harmonized to ViT through extensive experiments. Consequently, while significantly improving the performance of the ViT architecture, we have shown that the pooling layer by considering spatial interaction ratio is essential to a self-attention-based architecture.

Acknowledgement We thank NAVER AI Lab members for valuable discussion and advice. NSML [20] has been used for experiments. We thank the reviewers for the productive feedback.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 2

[2] Irwan Bello. Lambdanetworks: Modeling long-range interactions without attention. In *International Conference on Learning Representations*, 2021. 2

[3] Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. *arXiv preprint arXiv:1812.00332*, 2018. 5

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020. 1

[5] Nadav Cohen, Or Sharir, and Amnon Shashua. On the expressive power of deep learning: A tensor analysis. In *Conference on learning theory*, pages 698–728. PMLR, 2016. 1

[6] Nadav Cohen and Amnon Shashua. Inductive bias of deep convolutional networks through pooling geometry. In *International Conference on Learning Representations*, 2016. 1

[7] Zihang Dai, Guokun Lai, Yiming Yang, and Quoc V Le. Funnel-transformer: Filtering out sequential redundancy for efficient language processing. *arXiv preprint arXiv:2006.03236*, 2020. 2

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 1, 6

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 1, 2, 4, 5, 6, 7

[10] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 8

[11] Dongyoon Han, Jiwhan Kim, and Junmo Kim. Deep pyramidal residual networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5927–5935, 2017. 2

[12] Dongyoon Han, Sangdoo Yun, Byeongho Heo, and YoungJoon Yoo. Rethinking channel dimensions for efficient model design. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 732–741, 2021. 2, 3

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1, 2, 3, 7, 8

[14] Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. Bag of tricks for image classification with convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 7

[15] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. *arXiv preprint arXiv:1907.07174*, 2019. 8

[16] Byeongho Heo, Sanghyuk Chun, Seong Joon Oh, Dongyoon Han, Sangdoo Yun, Gyuwan Kim, Youngjung Uh, and Jung-Woo Ha. Adamp: Slowing down the slowdown for momentum optimizers on scale-invariant weights. In *International Conference on Learning Representations*, 2021. 6

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

[18] Elad Hoffer, Tal Ben-Nun, Itay Hubara, Niv Giladi, Torsten Hoefler, and Daniel Soudry. Augment your batch: Improving generalization through instance repetition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8129–8138, 2020. 6

[19] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1314–1324, 2019. 2, 3, 7

[20] Hanjoo Kim, Minkyu Kim, Dongjoo Seo, Jinwoong Kim, Heungseok Park, Soeun Park, Hyunwoo Jo, KyungHyun Kim, Youngil Yang, Youngkwan Kim, Nako Sung, and Jung-Woo Ha. NSML: meet the mlaas platform with a real-world case study. *CoRR*, abs/1810.09957, 2018. 8

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012. 1, 2

[22] Boyi Li, Felix Wu, Kilian Q Weinberger, and Serge Belongie. Positional normalization. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 2

[23] Xiao Li, Jianmin Li, Ting Dai, Jie Shi, Jun Zhu, and Xiaolin Hu. Rethinking natural adversarial examples for classification models. *arXiv preprint arXiv:2102.11731*, 2021. 8

[24] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 2, 7

[25] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018. 5

[26] Filip Radenović, Giorgos Tolias, and Ondřej Chum. Fine-tuning cnn image retrieval with no human annotation. *IEEE transactions on pattern analysis and machine intelligence*, 41(7):1655–1668, 2018. 1

[27] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10428–10436, 2020. 6, 7

[28] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jon Shlens. Stand-alone self-attention in vision models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 2

[29] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 2, 3, 7

[30] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2

[31] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 2

[32] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019. 1, 2, 3, 7

[33] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021. 2, 4, 5, 6, 7, 8

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017. 1, 2

[35] Jesse Vig and Yonatan Belinkov. Analyzing the structure of attention in a transformer language model. *arXiv preprint arXiv:1906.04284*, 2019. 4, 5

[36] Huiyu Wang, Yukun Zhu, Bradley Green, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Axial-deeplab: Stand-alone axial-attention for panoptic segmentation. In *European Conference on Computer Vision*, pages 108–126. Springer, 2020. 2

[37] 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. 1, 2

[38] Ross Wightman. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019. 7, 8

[39] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018. 2

[40] Kai Xiao, Logan Engstrom, Andrew Ilyas, and Aleksander Madry. Noise or signal: The role of image backgrounds in object recognition. *arXiv preprint arXiv:2006.09994*, 2020. 8

[41] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6023–6032, 2019. 6

[42] Sangdoo Yun, Seong Joon Oh, Byeongho Heo, Dongyoon Han, Junsuk Choe, and Sanghyuk Chun. Re-labeling imagenet: from single to multi-labels, from global to localized labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2340–2350, 2021. 7

[43] Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Zhi Zhang, Haibin Lin, Yue Sun, Tong He, Jonas Mueller, R Manmatha, et al. Resnest: Split-attention networks. *arXiv preprint arXiv:2004.08955*, 2020. 7

[44] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 2, 6, 7, 8