# Enriching Local Patterns with Multi-Token Attention for Broad-Sight Neural Networks

Hankyul Kang
Ajou University
hankyulkang1997@gmail.com

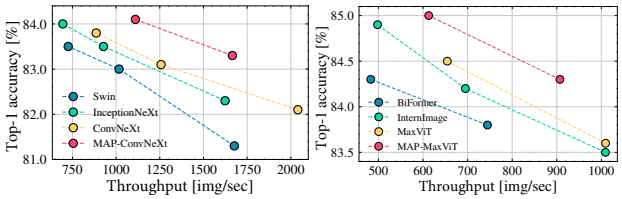
Jongbin Ryu *
Ajou University
jongbinryu@ajou.ac.kr

## Abstract

*In neural networks, recognizing visual patterns is challenging because global average pooling disregards local patterns and solely relies on over-concentrated activation. Global average pooling enforces the network to learn objects regardless of their location, so features tend to be activated only in specific regions. To support this claim, we provide a novel analysis of the problems that over-concentration brings about in networks with extensive experiments. We analyze the over-concentration through problems arising from feature variance and dead neurons that are not activated. Based on our analysis, we introduce a multi-token attention pooling layer to alleviate the over-concentration problem. Our attention-pooling layer captures broad-sight local patterns by learning multiple tokens with the proposed distillation algorithm. It resolves the high bias and high variance errors of learned multi-tokens, which is crucial when aggregating local patterns with multi-tokens. Our method applies to various vision tasks and network architectures such as CNN, ViT, and MLP-Mixer. The proposed method improves baselines with few extra resources, and a network employing our pooling method works favorably against state-of-the-art networks. We open-source the code at https://github.com/Lab-LVM/imagenet-models.*

## 1. Introduction

This paper presents a new analysis of the global average pooling (GAP) method that significantly impacts the learning process. Since gradients are backpropagated, generating discriminant features in the last pooling layer is a critical stage in deep neural networks. For this reason, the lossy GAP, the commonly used last pooling layer in CNN [31,46], ViT [53, 64], and MLP [7, 48], prevents entire networks from learning particular local information. Despite its importance in the learning process, insufficient study has been conducted to establish why this information loss occurs and how it affects the learned network [34, 37]. Therefore, in this paper, we present a new analysis that the performance degradation stems from the inability to learn local patterns properly due to the over-concentration problem.




(a) Small model (b) Large model

Figure 1. Experimental comparison of our MAP with SOTA networks in various model scales on ImageNet. Our MAP-ConvNeXt and MAP-MaxViT perform favorably against the SOTA networks.

Since GAP over-concentrates the activated neurons into the most discriminant small region (*i.e.*, center of an object), the localized patterns disappear in the pooled feature [8]. We call this drawback of GAP 'over-concentration' and point to it as the cause of network performance degradation. We empirically show that the over-concentrated feature map from GAP decreases spatial and channel diversity. The feature representation power grows when the diversity of the feature map is high; however, GAP reduces the feature diversity by over-concentrating only the specific region so that the network learns the biased information of the specific local patterns. We analyze this over-concentration problem of GAP through extensive experiments from various views in Sec. 3.

Motivated by the analysis of the over-concentration problem, we introduce multi-token attention pooling (MAP), which uses efficient attention architecture to learn the local spatial patterns with multi-tokens and does not concentrate on a specific region. To maximize the capability of learning local patterns with multi-tokens, we propose the distillation method, minimizing multi-tokens' bias and variance errors. Since we tackle using multi-tokens from an ensemble viewpoint, we address the bias and variance errors

*Corresponding author.

inherent in the ensemble approach. To this end, we reduce variance error by aggregating multiple tokens into groups and lower bias error by dispersing the centers of the token groups. Specifically, we doubly distill multi-tokens to handle both bias and variance error; positively distill tokens to reduce variance error, while negatively distill tokens to decrease the bias error. This learning approach eventually enriches local patterns in learned neural networks by facilitating multi-token deployment in an efficient attention-pooling layer.

Various experiments are conducted to prove the usefulness of our method, and we present several benefits of our MAP through the experimental results. First, our MAP can be generalized to apply most network architectures, including CNN, ViT, and even MLP-Mixer. Second, we only modify a last pooling layer of neural networks so that ours is fast and performs favorably against state-of-the-art networks, as shown in Fig. 1. Third, unlike previous pooling methods, the proposed MAP is highly applicable to the downstream task. We introduce a new tweak specific to dense prediction to make ours useful for downstream tasks.

In summary, we analyze the over-concentration problem owing to GAP, which justifies creating the proposed MAP. We show that the proposed MAP performs well compared to baseline backbones and state-of-the-art methods in image classification, object detection, and semantic segmentation tasks. We condense the contribution of this paper as follows:

- We present a new analysis that global average pooling degrades the performance of the network due to the over-concentration problem. We observe that global average pooling over-concentrates on the subtle region, thereby hindering networks from learning diverse local patterns.
- We, in accordance with the analysis, propose the MAP method that efficiently learns the local spatial patterns, which uses multiple class tokens with a doubly distilled learning approach.
- We show the proposed MAP exhibits significantly improved throughput and classification performance compared to state-of-the-art networks. We also demonstrate that a network with our MAP achieves competitive performance on fine-tuning and downstream tasks.

## 2. Preliminary: Class Attention Pooling

We compare GAP with class attention pooling [52] (CAP) to address the over-concentration problem. Since the CAP applies the class tokens to local pixels separately, locality remains in the pooled features, thus avoiding the over-concentration problem. In this part, we introduce the CAP method before comparing it to GAP for the over-concentration problem in Sec. 3. CAP aggregates image features $x_{img} \in \mathbb{R}^{n\times d}$ into a class token $x_v \in \mathbb{R}^{1\times d}$ by using multi-head class attention (MHCA) as:

$$\begin{aligned} Q &= W_q x_v + b_q \in \mathbb{R}^{1\times d}, \\ K &= W_k z + b_k \in \mathbb{R}^{(1+n)\times d}, \\ V &= W_v z + b_v \in \mathbb{R}^{(1+n)\times d}, \end{aligned} \tag{1}$$

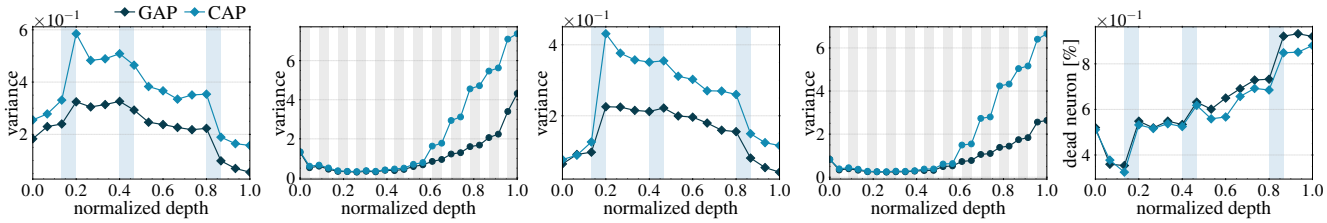
where $z = [x_v, x_{img}] \in \mathbb{R}^{(1+n)\times d}$, $[W_q, W_k, W_v, W_o]$ denote embedding matrix, and $[b_q, b_k, b_v, b_o]$ represent the corresponding bias. The complexity of this attention operation for CAP is much lower compared to the vanilla self-attention method. Since only class tokens are used as a query, the complexity of CAP is $O(N)$, which grows linearly as the number of image feature dimension $N$ grows. In contrast, the complexity of vanilla self-attention, which uses image features for querying and keying, is $O(N^2)$. Therefore, we could exploit CAP that maximizes local information using manageable computational complexity.

## 3. Analysis

In this section, we analyze GAP in terms of the over-concentration problem by comparing it with CAP. Previously, Raghu et al. demonstrated that ViT trained with GAP shows less precise localization than CAP on the CKA heatmap analysis [37]. Delving deeper, we examine the origins of the over-concentration problem and explore its practical drawbacks. For this, we analyze layer-wise feature diversity (Sec. 3.1), dead neurons (Sec. 3.2), and non-salient object recognition tasks (Sec. 3.3) by using GAP and CAP. Our extensive analyses confirm that networks utilizing GAP concentrate fewer local patterns while using CAP learns broad-sighted local patterns for visual recognition tasks.

### 3.1. Feature Diversity

Feature over-concentration entails that semantically similar features span an entire spatial area, resulting in the extinction of minor but discriminant local features in a network. The over-concentration is primarily caused by GAP, which focuses only on small regional information due to its spatially average down operation. This issue can also be found in network visualization studies where the class activation map (CAM) resides in small regions [69]. To overcome this problem, they diversify the input [58] and network architecture [57] to expand the activated regions in a feature map. These studies focus mostly on segmentation tasks in which CAM results are employed as weak labels by expanding the activation map to cover the entire object. We assume that the reason for this problem that activated neurons are concentrated in a narrow region is due to the GAP's average down operation, which is also analyzed in [37]. GAP recognizes objects regardless of location, so only the specific location is activated, and the rest of the patterns are discarded [35].




(a) Channel diversity (b) Spatial diversity (c) Dead neuron

Figure 2. Experimental study on channel and spatial diversity [34] and dead neurons. We measure the diversity as the feature variance of each layer. We use ResNet (left) and DeiT-S/16 (right) as backbone networks. In all cases, CAP achieves higher diversity compared to GAP. For dead neurons, we measure the ratio of non-activated neurons in each layer. As a result, CAP activates more neurons than GAP. Refer to Sec.3 of the supplementary material for details.

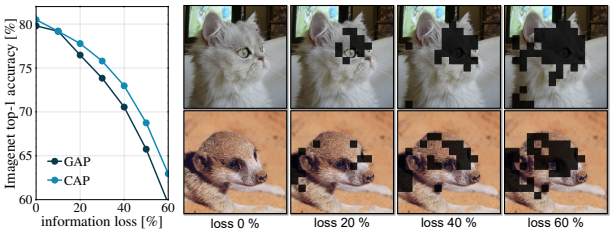



Figure 3. Experimental study of non-salient object recognition on ImageNet using ResNet50 with different pooling layers. We measure classification accuracy after artificially removing the object's salient area using a self-supervised ViT model DINO [6].

We verify our analysis of over-concentration by feature diversity. Each channel of the feature map represents the pattern of an object. If the channel diversity is high, neurons are activated at various spatial positions. Both channel and spatial diversity indicate the diversity of information possessed by the feature map, and the more monotonic features, the lower the diversity. Therefore, over-concentrated features are monotonic; thus, the diversity is inevitably low. We compare the feature diversities of networks with GAP and CAP as shown in Figs. 2a and 2b. Unsurprisingly, the CAP pools each local pixel independently and achieves much higher diversity on both CNN and ViT architecture. This finding supports that the GAP leads to the over-concentration problem of entire network layers.

### 3.2. Dead Neuron

The over-concentration exacerbates the dead neuron [33] problem in neural networks. Due to over-concentration, local spatial patterns outside the central position are not activated even though they have information that can identify objects. A spatial position ignored in GAP at the end of a network loses the opportunity to be learned in all layers during backpropagation. Therefore, as shown in Fig. 2c, a network trained with GAP generates much more dead neurons than CAP. These differences significantly affect the network performance, and a network with CAP learns richer local patterns, thus alleviating the dead neuron problem.

### 3.3. Non-salient recognition

We point out that GAP is vulnerable to the disappearance of salient regions due to over-concentration. Since GAP recognizes objects by concentrating on a narrow local region, occlusion of that region can lead to failure in object recognition. As shown in Fig. 3, as the disappeared salient region becomes larger, the performance of GAP decreases rapidly. This is because the over-concentrated region of GAP is too narrow, so GAP does not activate salient features outside the over-concentration region. However, the CAP that independently pools all local pixels has a broad concentration region, so it recognizes the occluded object better than GAP.

## 4. Method

### 4.1. Multi-Token Attention Pooling

Based on our analysis, we propose multi-token attention pooling to enrich local patterns with multiple class tokens in neural networks. Since the class token is a key for learning locality, we deploy multiple of them to maximize the learning of the local patterns in the attention pooling. To use this approach, we explore previous studies [39, 40, 61] that have addressed using multiple class tokens in deep neural networks. Among them, we employ the multi-branch head architectural design with Gramian matrix attention [40] to produce multiple class tokens. For every branch, we create a token group consisting of multiple class tokens, so we end up with multiple token groups. We optimize these multiple token groups by the branched group distillation methods as shown in Fig. 5c and Algorithm 1.

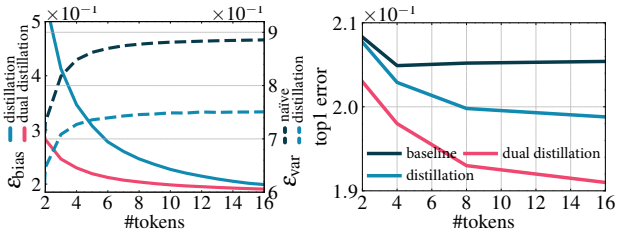
### 4.2. Distillation for Multi-Token Attention

#### 4.2.1 High Bias and Variance Error of Multi-Tokens

Previous works [36, 52] employ the CAP method to learn richer information in the Vision Transformer. However, we extend this attention-pooling method into multiple class to-

**Algorithm 1.** Pseudocode of MAP with our distillation method in a PyTorch style.

```
# T,G,D,C: #tokens,#groups,#dims,#classes
# MHCA: Multi-Head Cross-Attention
# Step 1. Initialization

backbone = Network(dim=D)
t, fc = Param(G, T, D), Param(G, T+1, C, D)
mhca = [MHCA(dim=D) for _ in range(G)]

for x,y in dataloader:
  # Step 2. Multi-token Attention Pooling
  x = backbone(x)
  logit = zeros(G, T+1, C)
  for g in range(G):
    t_avg = t[g].sum(dim=0)#(1,D)
    token = cat([t[g],t_avg],dim=0)#(T+1,D)
    token = mhca[g](q=token,k=x,v=x)#(T+1,D)
    logit[g] = fc[g].matmul(token)#(T+1,C)

  # Step 3. Distillation
  loss_intra = loss_inter = 0
  logit_total = logit.sum(dim=(0,1))
  loss_ce = CE(logit_total, y)

  for g in range(G):
    logit_agg = logit[g,:T].sum(dim=0)
    logit_avg = logit[g,T]
    loss_intra += KL(logit_avg, logit_agg)
    loss_inter += KL(logit_agg, logit_total)

  loss = loss_ce + loss_intra - loss_inter
  loss.backward()
```

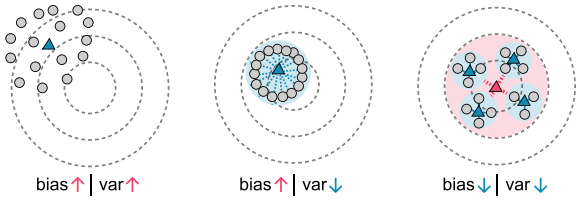



(a) Bias and var error (b) Top1 error

Figure 4. Experimental study of different multi-tokens learning methods on ImageNet. We scale up the number of tokens from 2 to 16. It is shown that our learning method (red line) effectively optimizes both bias and variance errors for more class tokens.

kens by addressing high bias and variance errors. As shown in Fig. 4 and Tab. 9, we observe that simply increasing the number of tokens does not improve accuracy.

To optimize the use of multiple tokens, we introduce a learning algorithm that uses two distillation [21] losses with branched token groups. For these losses, we partition the class tokens into multiple groups in our architecture to optimize both the bias and variance errors, as shown in Fig. 5. Aggregating tokens in the same group reduces variance error, while diversifying between different token groups decreases bias error. As a result, the proposed learning strategy effectively reduces bias and variance errors by jointly managing intra- and inter-group distillation losses. To validate the proposed method, we perform an experiment that




(a) Baseline (b) Distillation (c) Our distillation

Figure 5. Visualization for logit distribution of (a) baseline multiple class tokens, (b) inter-group distillation for the token group, and (c) our inter- and intra-group distillation algorithm for branched token groups. While the inter-group distillation learning method of (b) gathers multiple tokens to the center (denoted as a blue triangle) of its group, the proposed learning method of (c) conducts the multiple token gathering in scattered multiple token branches. Thus, the final aggregated logit (denoted as a red triangle) from multiple token groups in (c) shows the most accurate result with the least error.

measures image classification error per number of tokens and decomposes it into bias and variance error, as shown in Fig. 4. In this experiment, we observe that using baseline multiple class tokens (black line) faces performance saturation at a smaller number of class tokens, as shown in Fig. 4b. However, our learning strategy (red line) resolves the performance saturation issue by alleviating high bias and high variance, as shown in Fig. 4a. Specifically, we define bias and variance errors as [12, 14]:

$$\varepsilon_{\text{bias}} = \mathcal{L}(\hat{y}_*, y),\ \varepsilon_{\text{var}} = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}(\hat{y}_i, \hat{y}_*), \tag{2}$$

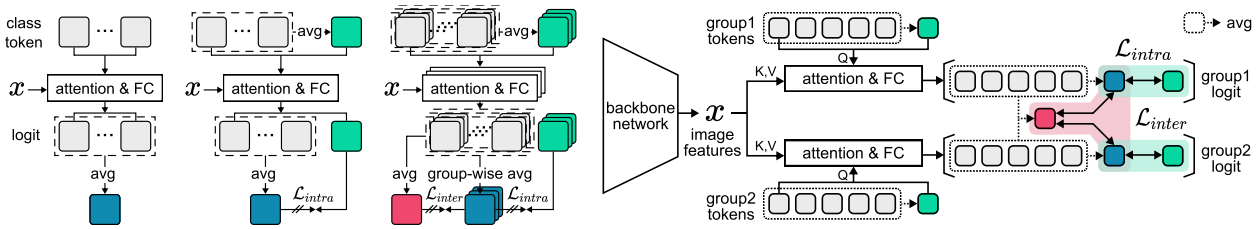
where $\mathcal{L}$ denotes loss function, $\hat{y}_i$ is predicted label from $i-th$ class token, $\hat{y}_*$ represents the predicted label from aggregated multiple tokens, and $N$ stands for the number of tokens. Following [12], we use multiclass *zero-one* loss $\mathcal{L}_{01}(\hat{y}, y) = \mathbb{1}_{\hat{y}\neq y}$ as our objective function in Eq. (2).

#### 4.2.2 Intra-Group Distillation

We introduce an intra-group distillation loss to reduce the variance error in the group of multiple tokens. Precisely, we pass the averaged and original class tokens through the attention and FC classifier layer to compute logits of aggregated and each individual token denoted as the green and gray square in Fig. 6b. The logits from the averaged class tokens distill the group class tokens to be gathered toward the averaged one. In this way, original multiple class tokens learn common knowledge from the averaged one, and thus, the variance error within the same group is reduced. We formulate the intra-group distillation loss with the logits as:

$$\mathcal{L}^{intra} = p^m(log(p^m) - log(p^c)), \tag{3}$$

where $p^m$ and $p^c$ represent the logits of multiple class tokens and their aggregated ones in a group.




(a) Multi-token (b) Distillation (c) Our distillation (d) Architectural design for our multi-branch approach with MAP

Figure 6. The architectural design of (a) the baseline multiple class tokens, (b) intra-group distillation for the token group, (c) our inter- and intra-group distillation algorithm for branched token groups, and (d) our multi-branch architecture with MAP. To optimize the use of multiple tokens, we train networks with two different distillation methods. Our MAP employs multi-branch attention classifiers, in which each classifier performs multi-token attention pooling.

#### 4.2.3 Inter-Group Distillation

We present another distillation method with the branched token groups to decrease the bias error of a network. The proposed strategy for increasing the KL divergence with a negative weight parameter ($\lambda_2$ in Eq. (5)) with the branched multiple heads has been studied in [40] to reduce the correlation of multiple branches for the network generalization. This strategy encourages the network to learn diverse information, thereby preventing it from getting stuck in local minima by minimizing bias errors. Therefore, following this finding, we lower the correlation of logits from multiple token groups with KL divergence as a form of regularization, encouraging each group to acquire distinct features. Consequently, this inter-group distillation algorithm with the branched multiple token groups reduces bias error, bringing the final prediction $p^*$ closer to the ground truth label. We define the inter-group distillation loss as:

$$\mathcal{L}_g^{inter} = p^*(log(p^*) - log(p_g^m)), \quad p^* = \frac{1}{N_G}\sum_{g=1}^{N_G} p_g^m, \quad (4)$$

where $N_G$ denotes the number of token groups, and the subscript $g$ represents the group index. By putting them together, we establish our final loss as:

$$\mathcal{L} = \mathcal{L}^{CE} + \lambda_1 \sum_{g=1}^{N_G} \mathcal{L}_g^{intra} + \lambda_2 \sum_{g=1}^{N_G} \mathcal{L}_g^{inter}, \quad (5)$$

where $\lambda_1$ and $\lambda_2$ denote the weight hyper-parameters of intra- and inter-group distillation loss. We configure the weights as $\lambda_1 = 1.0$ and $\lambda_2 = -0.8$ where the $\lambda_1$ ensures the gathering logits of multiple class tokens in a token group while the $\lambda_2$ reduce the correlation of logits from the multiple token groups. As shown in Fig. 4, we can achieve a better bias-variance error trade-off of a network by using the proposed distillation algorithm. We show the architecture design for this algorithm in Fig. 6c and its application to the multi-branch approach of a network in Fig. 6d.

### 4.3. Dense prediction tweak

To apply our approach of multiple class tokens for dense prediction tasks, we divide the multiple output tokens of MAP by the number of network stages and broadcast them to each stage, as shown in the supplementary. In this way, after the multi-scale feature passes the MAP, the divided tokens are fed back to each scale, diversifying the network's connectivity.

Thus, each network stage will enjoy the benefits of MAP for the dense prediction tasks. Several studies [30, 31, 40] have explored the construction of dense prediction networks through the interconnection of different stage's outputs in a backbone. However, this paper presents an efficient model that allocates multiple class tokens for each network stage to improve dense prediction performance using our MAP.

## 5. Experiment

### 5.1. Image Classification

We evaluate the proposed MAP on ImageNet [10]. We replace the network's last pooling layer with our MAP. To show the generalization ability of our MAP, we apply ours to CNN (ResNet [18], ConvNeXt [31]), Transformer (DeiT [50], PiT [20]), Hybrid (MaxViT [53]), and MLP (ResMLP [49]). We train networks for 300 epochs except for Tab. 1 with 100 epochs. Following recent training trend [31, 50], we use commonly adopted augmentation methods including CutMiX [66], MixUp [67], Random Erasing [68], and Rand-Augmentation [9], which are provided in TIMM [59] library.

We demonstrate that the proposed MAP achieves favorable performance against strong pooling method baselines and could be widely used in various networks. We directly apply various pooling methods to our architecture design. First, Tab. 1 shows that the proposed MAP performs better than the previous pooling methods by more than +3.6% margin with affordable additional resources. Second, Tab. 2 shows that the proposed MAP improves various baseline networks about +2.0% ResNet, +2.1% PiT-

Table 1. Experimental study on the comparison between our MAP and other pooling methods. We use ResNet50 as a backbone and train it with 100 epochs.

| Pooling | Param (M) | FLOPs (G) | Top-1 Acc. (%) |
|---|---|---|---|
| GAP [18] | 25.6 | 4.1 | 77.1 |
| VGG [43] | 235.3 | 4.3 | 76.3 |
| DFT [42] | 92.7 | 4.1 | 77.6 |
| CAP [52] | 59.1 | 4.5 | 78.5 |
| MAP | 38.0 | 4.5 | **80.7** (3.6) |

Table 2. Experimental study of GAP, CAP, and MAP methods on baseline networks. Our MAP performs well compared to others with each baseline. MAP-light utilizes averaged tokens only during the inference phase, offering the advantage of reducing parameters. MAP-light adjusts the channel dimensions of CAP to compare ours with others so that we can use similar resources.

| Pooling | Param (M) | FLOPs (G) | Top-1 Acc. (%) |
|---|---|---|---|
| Convolution: ResNet50 [18] | | | |
| GAP [18] | 25.6 | 4.1 | 79.8 |
| CAP [52] | 59.1 | 4.5 | 80.6 |
| MAP-light | 35.0 | 4.5 | 81.3 |
| MAP | 38.0 | 4.5 | **81.8** (2.0) |
| Transformer: PiT-S [20] | | | |
| GAP [18] | 23.3 | 2.4 | 79.8 |
| CAP [52] | 44.3 | 2.5 | 81.2 |
| MAP | 36.2 | 2.6 | **81.9** (2.1) |
| MLP: ResMLP-S24 [49] | | | |
| GAP [18] | 30.0 | 6.0 | 79.4 |
| CAP [52] | 52.9 | 6.3 | 80.0 |
| MAP | 43.3 | 6.2 | **81.0** (1.6) |

S, +1.6% MLP-Mixer with manageable additional computational costs, which indicates the generalization ability of our approach to various architectures. Finally, Tab. 3 shows that, with our MAP, the ResNet [18], ConvNeXt [31], and MaxViT [61] achieve competitive results against SOTA networks. Therefore, we demonstrate that utilizing our MAP in a network yields substantial performance improvement while maintaining a **high throughput**. More performance comparisons and ablations can be found in Sec.4 of the supplementary material.

## 5.2. Downstream task

We validate the applicability of the proposed MAP to the downstream task by evaluating its performance in three dense prediction experiments. Unlike the existing pooling method, a network employing our MAP may directly use features of the pooling layer for the dense prediction. Therefore, we confirm that MAP improves baseline networks in object detection, semantic segmentation, and instance segmentation in Tabs. 4 and 5. Our MAP uses a modest amount

Table 3. Experimental results on the ImageNet dataset using original Rendition (R) [19], V2 [38], Real [1], and Val [10] labels. The throughput of each network is measured by a single RTX 3090 GPU. We report the accuracy of networks on R, V2, and Real labels by utilizing publicly available checkpoints. It shows that our networks are considerably **faster** than other SOTA networks while achieving higher accuracy.

| Network | Throughput (img/s) | Param. (M) | FLOPs (G) | ImageNet Top1 (%) R | V2 | Real | Val |
|---|---|---|---|---|---|---|---|
| Swin-T [30] | 1676 | 28.3 | 4.4 | 41.5 | 69.5 | 86.7 | 81.3 |
| BiFormer-T [71] | 1569 | 13.1 | 2.2 | 45.3 | 70.7 | 87.1 | 81.4 |
| PoolFormer-S36 [63] | 1156 | 31.0 | 5.0 | 42.1 | 69.9 | 86.6 | 81.4 |
| ConvNeXt-T [31] | 2040 | 29.0 | 4.5 | 47.2 | 71.0 | 87.3 | 82.1 |
| PoolFormer-M36 [63] | 802 | 56.0 | 8.8 | 43.2 | 70.8 | 86.9 | 82.1 |
| InceptionNeXt-T [65] | 1624 | 28.1 | 4.2 | 47.0 | 72.0 | 87.5 | 82.3 |
| DaViT-T [11] | 1634 | 28.3 | 4.5 | 47.3 | 71.9 | 87.5 | 82.8 |
| Swin-S [30] | 1017 | 49.6 | 8.5 | 45.2 | 71.8 | 87.7 | 83.0 |
| ConvNeXt-S [31] | 1257 | 50.0 | 8.7 | 49.6 | 72.4 | 88.1 | 83.1 |
| NAT-T [16] | 1289 | 28.0 | 4.3 | 44.9 | 72.1 | 87.9 | 83.2 |
| MogaNet-T [26] | 805 | 25.0 | 5.0 | 40.4 | 72.7 | 88.0 | 83.4 |
| InceptionNeXt-S [65] | 928 | 49.4 | 8.4 | 50.3 | 73.3 | 88.2 | 83.5 |
| Swin-B [30] | 726 | 87.7 | 15.1 | 46.5 | 72.3 | 87.9 | 83.5 |
| BiFormer-S [71] | 745 | 26.0 | 4.5 | 49.7 | 73.6 | 88.3 | 83.8 |
| (Ours) MAP-ResNet50 | **2819** | 38.0 | 4.5 | 45.9 | 70.2 | 86.7 | 81.8 |
| (Ours) MAP-ConvNeXt-T | **1665** | 47.8 | 4.9 | 48.7 | 72.5 | 88.0 | 83.3 |
| (Ours) MAP-ConvNeXt-S | **1111** | 82.8 | 9.2 | 52.0 | 73.8 | 88.6 | 84.1 |
| InternImage-T [54] | 1009 | 29.9 | 4.8 | 48.5 | 72.9 | 88.0 | 83.5 |
| MaxViT-T [53] | 1009 | 30.9 | 5.4 | 48.8 | 72.9 | 88.0 | 83.6 |
| ConvNeXt-B [31] | 886 | 89.0 | 15.4 | 51.3 | 73.7 | 88.3 | 83.8 |
| InternImage-S [54] | 695 | 50.1 | 8.2 | 50.4 | 74.0 | 88.6 | 84.2 |
| NAT-B [16] | 614 | 90.0 | 13.7 | 49.7 | 74.1 | 88.6 | 84.3 |
| BiFormer-B [71] | 483 | 57.0 | 9.8 | 49.6 | 74.0 | 88.5 | 84.3 |
| MogaNet-S [26] | 400 | 44.0 | 9.9 | 49.4 | 74.0 | 88.5 | 84.3 |
| MaxViT-S [53] | 654 | 69.0 | 11.7 | 50.9 | 73.9 | 88.5 | 84.5 |
| MogaNet-B [26] | 284 | 83.0 | 15.9 | 50.0 | 74.1 | 88.4 | 84.7 |
| NFNet-F1 [4] | 239 | 132.6 | 35.5 | 47.5 | 71.9 | 88.1 | 84.7 |
| InternImage-B [54] | 498 | 97.5 | 16.0 | 52.4 | 74.7 | 88.9 | 84.9 |
| MaxViT-B [53] | 361 | 120.0 | 23.4 | 52.2 | 74.3 | 88.6 | 85.0 |
| (Ours) MAP-MaxViT-T | **907** | 50.0 | 5.8 | 51.2 | 74.3 | 88.8 | 84.3 |
| (Ours) MAP-MaxViT-S | **613** | 100.9 | 11.8 | 54.1 | 74.8 | 88.9 | 85.0 |

Table 4. Experimental study of downstream task using Mask-RCNN [17] and Cascade Mask-RCNN [5] on COCO2017 [28] for object detection and instance segmentation. Following [31], we measure FLOPs and FPS with image sizes of $1280 \times 800$.

| Network | Param. (M) | FLOPs (G) | FPS (img/s) | $AP^b$ (%) | $AP^b_{50}$ (%) | $AP^b_{75}$ (%) | $AP^m$ (%) | $AP^m_{50}$ (%) | $AP^m_{75}$ (%) |
|---|---|---|---|---|---|---|---|---|---|
| Mask R-CNN [5] (1 × schedule) | | | | | | | | | |
| ResNet50 [18] | 44 | 260 | 21.8 | 40.2 | 61.4 | 43.7 | 36.9 | 58.4 | 39.4 |
| MAP-ResNet50 | 61 | 285 | 18.7 | **41.4** | 63.2 | 45.6 | **38.1** | 60.2 | 41.0 |
| PVT-S [55] | 44 | 245 | 19.9 | 40.2 | 62.9 | 43.4 | 37.2 | 59.7 | 39.5 |
| MAP-PVT-S | 47 | 249 | 19.0 | **40.7** | 62.7 | 43.9 | **37.5** | 59.7 | 39.8 |
| Cascade Mask R-CNN [5] (3 × schedule) | | | | | | | | | |
| Swin-T [30] | 86 | 742 | 11.1 | 50.4 | 69.2 | 54.7 | 43.7 | 66.6 | 47.3 |
| ConvNeXt-T [31] | 86 | 741 | 11.8 | 50.4 | 69.1 | 54.8 | 43.7 | 66.5 | 47.3 |
| MAP-ConvNeXt-T | 88 | 750 | 11.5 | **51.1** | 69.3 | 55.8 | **44.2** | 66.9 | 48.2 |
| Swin-S [30] | 107 | 832 | 8.5 | 51.9 | 70.7 | 56.3 | 45.0 | 68.2 | 48.8 |
| ConvNeXt-S [31] | 108 | 827 | 9.0 | 51.9 | 70.8 | 56.5 | 45.0 | 68.4 | 49.1 |
| MAP-ConvNeXt-S | 110 | 836 | 8.9 | **52.5** | 70.8 | 57.0 | **45.2** | 68.3 | 49.3 |

Table 5. Experimental study of semantic segmentation downstream tasks using Semantic FPN [23] and UperNet [60] on the ADE20K [70] dataset. Following ConvNext [31], we report multi-crop mIOU to measure FLOPs and FPS with the same image sizes of 2048 × 512 for all networks.

| Network | Param. (M) | FLOPs (G) | FPS (img/s) | mIOU (%) |
|---|---|---|---|---|
| Semantic FPN [23] (80K schedule) | | | | |
| ResNet50 [18] | 29 | 178 | 38.6 | 37.3 |
| MAP-ResNet50 | 45 | 203 | 34.6 | **39.2** |
| PVT-S [55] | 28 | 163 | 31.8 | 39.3 |
| MAP-PVT-S | 31 | 167 | 28.6 | **40.1** |
| UperNet [60] (160K schedule) | | | | |
| Swin-T [30] | 60 | 941 | 25.8 | 45.8 |
| ConvNeXt-T [31] | 60 | 939 | 31.7 | 46.7 |
| MAP-ConvNeXt-T | 63 | 948 | 31.5 | **48.1** |
| ConvNeXt-S [31] | 82 | 1027 | 25.1 | 49.3 |
| MAP-ConvNeXt-S | 84 | 1037 | 24.1 | **49.5** |

Table 6. Ablation study of #Groups and #Tokens per group. We report top-1 accuracy of the 100 epoch training setting on ImageNet-1K with different numbers of groups (#G), tokens per group (#T), and class tokens (#CT).

| #G | #T | #CT (#G×#T) | Throughput (img/sec) | FLOPs (G) | Δ (↓) | Top-1 Acc. (%) | Δ (↑) |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 3094 | 4.6 | - | 78.5 | - |
| | 2 | 2 | 3073 | 4.6 | +0.0 | 79.2 | **+0.7** |
| | 4 | 4 | 3021 | 4.7 | +0.1 | 79.5 | **+1.0** |
| | 8 | 8 | 2979 | 4.8 | +0.2 | 79.8 | **+1.3** |
| 2 | 1 | 2 | 2920 | 4.5 | - | 79.7 | - |
| | 4 | 8 | 2819 | 4.5 | +0.0 | 80.7 | **+1.0** |
| 4 | 1 | 4 | 2528 | 4.7 | - | 80.5 | - |
| | 4 | 16 | 2417 | 4.8 | +0.1 | 81.0 | **+0.5** |
| 16 | 1 | 16 | 1652 | 5.0 | - | 80.6 | - |

Table 7. Ablation study of the distillation loss. We report top-1 accuracy on ImageNet-1K under the 100 epoch training setup with different numbers of class tokens (#VT).

| DiD | #VT=1 | #VT=2 | #VT=4 | #VT=8 | #VT=16 | Avg |
|---|---|---|---|---|---|---|
| ✗ | 78.5 | 78.7 | 79.1 | 79.2 | 79.0 | 79.0 |
| ✓ | - | 79.2 | 79.5 | 79.8 | 80.2 | 79.7 |
| Δ (↑) | - | **+0.5** | **+0.4** | **+0.6** | **+1.2** | **+0.7** |

of resources in all experiments while working favorably.

## 6. Ablation study

We perform various ablation studies on the ImageNet [10] dataset using the ResNet50 backbone. First, Tab. 6 shows an ablation study of the number of groups and tokens per group. This table demonstrates that our MAP works well compared to GA [40] (*i.e.*, #T=1) using manageable additional costs. Next, Tab. 7 analyzes the effectiveness of our distillation method. Our distillation loss brings consistent performance improvement in all token numbers. Furthermore, we compare MAP with the original ViT in Tab. 8. We also evaluate ViT-style MAP, which uses multiple class tokens (*i.e.*, 16) from the initial stage to perform full attention in a network. On the other hand, our MAP utilizes multiple class tokens at the last stage to perform class attention (CA) operations. Tab. 8 showcases that our MAP works favorably compared to ViT-S/B and ViT-style MAP. Moreover, Tab. 9 shows a step-wise evaluation to validate performance improvement of each element in various networks (*i.e.*, ResNet, ConvNeXt, PiT, and DeiT)

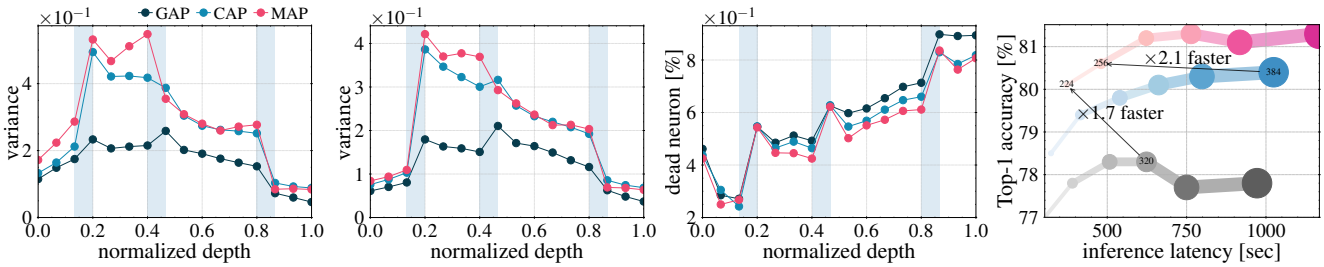
We analyze network behaviors according to the number

Table 8. Ablation study of ViT-style MAP. We report top-1 accuracy on the ImageNet dataset to compare our MAP, ViT-S/B, and ViT-style MAP methods.

| Network | Thrughput (img/sec) | Param. (M) | FLOPs (G) | ImageNet Top1 (%) R | V2 | Real | Val |
|---|---|---|---|---|---|---|---|
| ViT-S | 2611 | 22.1 | 4.3 | 42.3 | 68.5 | 85.7 | 79.8 |
| ViT-style MAP | 2417 | 27.8 | 4.6 | 40.5 | 67.5 | 85.3 | 79.1 |
| Our MAP | 2287 | 36.9 | 4.5 | 46.6 | 71.0 | 87.3 | 81.8 |
| ViT-B | 1011 | 86.6 | 16.9 | 44.9 | 71.2 | 86.8 | 81.8 |

Table 9. Ablation study of the proposed method. We train each network on ImageNet with 100 epochs. We confirm that our distillation methods with a branched token group improve the network's performance in all cases.

| Net | Class token | Intra distill | Inter distill | FLOPs (G) | Throughput (img/sec) | Top-1 (Acc. %) |
|---|---|---|---|---|---|---|
| *CNNs* | | | | | | |
| ResNet50 | - | - | - | 4.1 | 3334 | 77.1 |
| | ✓ | - | - | 4.4 | 2951 | **78.5** (+1.4) |
| | ✓ | ✓ | - | 4.5 | 2835 | **80.2** (+1.7) |
| | ✓ | ✓ | ✓ | 4.5 | 2819 | **80.7** (+0.5) |
| ConvNeXt-T | - | - | - | 4.5 | 2039 | 80.4 |
| | ✓ | - | - | 4.8 | 1864 | **81.3** (+0.9) |
| | ✓ | ✓ | - | 4.9 | 1792 | **82.2** (+0.9) |
| | ✓ | ✓ | ✓ | 5.3 | 1600 | **82.6** (+0.4) |
| *Transformers* | | | | | | |
| DeiT-S | - | - | - | 4.3 | 2611 | 75.8 |
| | ✓ | - | - | 4.4 | 2427 | **77.4** (+1.6) |
| | ✓ | ✓ | - | 4.4 | 2340 | **78.0** (+0.6) |
| | ✓ | ✓ | ✓ | 4.6 | 2078 | **79.7** (+1.7) |
| PiT-S | - | - | - | 2.4 | 2710 | 77.0 |
| | ✓ | - | - | 2.5 | 2534 | **78.1** (+1.1) |
| | ✓ | ✓ | - | 2.6 | 2440 | **79.2** (+1.1) |
| | ✓ | ✓ | ✓ | 2.8 | 2151 | **79.7** (+0.5) |




(a) Channel diversity (b) Spatial diversity (c) Dead neuron (d) Accuracy

Figure 7. Experimental study on the feature diversity, dead neurons, and latency versus accuracy for GAP, CAP, and our MAP on ImageNet. All networks are trained for 100 epochs. (a,b): MAP increases channel and spatial diversity compared to GAP and CAP. (c): MAP reduces dead neurons compared to other methods. (d): MAP achieves better top-1 accuracy at 1.7 and 2.1 times faster than GAP and CAP. The radius of each circle represents the input image size.

of tokens and observe diversified feature effects in learned neural networks, which support our initial claims on feature diversity and dead neurons. As shown in Figs. 7a to 7c, the multi-tokens (shown as magenta line) increase the channel and spatial diversity while decreasing the dead neurons. Also, in Fig. 7d, the performance of the image classification task is improved as the number of tokens increases without lowering throughput. Therefore, it is evident that using multiple class tokens facilitates learning of richer local patterns, thereby enhancing the network's performance.

## 7. Related Work

**Conventional pooling methods.** The fully connected (FC) layer is employed as the final aggregating method in early CNN architecture [25], AlexNet [24] and VGGNet [43]. The FC layer is straightforward and effective in encoding fine-grained local features, but it requires a large number of parameters and is weak in translation invariance property. To improve the translation invariance property, the GAP layer is introduced in modern CNN architectures [18, 45]. Recent works [2] incorporate the GAP layer in ViT, yielding favorable results compared to the class token. Although GAP has been the standard pooling method, there have been numerous attempts to replace it with cutting-edge methods such as orderless [15], bilinear [29], and DFT pooling [42, 62]. However, none of these methods can exceed GAP regarding throughput and model performance because GAP uses a superfast average down operation layer while ensuring the translational invariance property. Therefore, GAP is still the last pooling layer in current architectures [4, 47, 53].

**Class Token and Multi-feature Aggregation.** The class token was initially designed in vision transformer (ViT) architecture [13]. Subsequently, it was found that the CAP can be improved when inserting it into the last multi-head self-attention (MHSA) layer in a network [52]. As a result of this finding, CAP has now been exploited in numerous vision transformer architectures [52, 61]. In addition, there have been a few attempts to apply CAP in CNN architectures [39, 51]. However, these methods are generally difficult to use for reasons such as tailoring the backbone architecture to a specific condition. Recent work [40] trains the robust class tokens using multiple Gramian head classifiers with the analysis of the generalization approach of the random forest [3, 41]. Fusing multi-level features has been extensively studied in downstream tasks [27, 32]. Also, in the image classification task, there have been studies aggregating various levels of features [22, 40, 44] or spatially long-range dependencies [56].

## 8. Conclusion

In this paper, we present a new analysis of GAP that leads to the problem of over-concentration. We analyze the consequences of the over-concentration problem on network learning via feature diversity, dead neurons, and salient object recognition. This analysis is the basis for our MAP method, which focuses on acquiring abundant local patterns. With our MAP, we also offer a distillation algorithm to use the multi-tokens efficiently. Moreover, our MAP is readily applicable to downstream tasks, such as object detection and segmentation. Although we only replace the last pooling layer with MAP, it exhibits notably better performance than SOTA networks. We hope that our results will serve as the groundwork for future research.

**Acknowledgment.** This paper was supported in part by the ETRI Grant funded by Korean Government (Fundamental Technology Research for Human-Centric Autonomous Intelligent Systems) under Grant 24ZB1200, Artificial Intelligence Innovation Hub (RS-2021-II212068), Artificial Intelligence Convergence Innovation Human Resources Development (IITP-2024-RS-2023-00255968), and the NRF Grant (RS-2024-00356486).

## References


[1] Lucas Beyer, Olivier J Hénaff, Alexander Kolesnikov, Xiaohua Zhai, and Aäron van den Oord. Are we done with imagenet? *arXiv:2006.07159*, 2020.

[2] Lucas Beyer, Xiaohua Zhai, and Alexander Kolesnikov. Better plain vit baselines for imagenet-1k. *arXiv:2205.01580*, 2022.

[3] Leo Breiman. Random forests. *Machine learning*, 2001.

[4] Andy Brock, Soham De, Samuel L Smith, and Karen Simonyan. High-performance large-scale image recognition without normalization. In *ICML*, 2021.

[5] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *CVPR*, 2018.

[6] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021.

[7] Shoufa Chen, Enze Xie, Chongjian Ge, Runjian Chen, Ding Liang, and Ping Luo. Cyclemlp: a mlp-like architecture for dense visual predictions. *PAMI*, 2023.

[8] Vincent Christlein, Lukas Spranger, Mathias Seuret, Anguelos Nicolaou, Pavel Král, and Andreas Maier. Deep generalized max pooling. In *ICDAR*, 2019.

[9] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPRW*, 2020.

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *ICCV*, 2009.

[11] Mingyu Ding, Bin Xiao, Noel Codella, Ping Luo, Jingdong Wang, and Lu Yuan. Davit: Dual attention vision transformers. In *ECCV*, 2022.

[12] Pedro Domingos. A unified bias-variance decomposition. In *ICML*, 2000.

[13] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2020.

[14] Stuart Geman, Elie Bienenstock, and René Doursat. Neural networks and the bias/variance dilemma. *Neural computation*, 1992.

[15] Yunchao Gong, Liwei Wang, Ruiqi Guo, and Svetlana Lazebnik. Multi-scale orderless pooling of deep convolutional activation features. In *ECCV*, 2014.

[16] Ali Hassani, Steven Walton, Jiachen Li, Shen Li, and Humphrey Shi. Neighborhood attention transformer. In *CVPR*, 2023.

[17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[19] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *ICCV*, 2021.

[20] Byeongho Heo, Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Junsuk Choe, and Seong Joon Oh. Rethinking spatial dimensions of vision transformers. In *ICCV*, 2021.

[21] Geoffrey Hinton. Distilling the knowledge in a neural network. *arXiv:1503.02531*, 2015.

[22] Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens Van Der Maaten, and Kilian Q Weinberger. Multi-scale dense convolutional networks for efficient prediction. *arXiv:1703.09844*, 2017.

[23] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *CVPR*, 2019.

[24] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012.

[25] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

[26] Siyuan Li, Zedong Wang, Zicheng Liu, Cheng Tan, Haitao Lin, Di Wu, Zhiyuan Chen, Jiangbin Zheng, and Stan Z Li. Moganet: Multi-order gated aggregation network. In *ICLR*, 2023.

[27] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017.

[28] 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 *ECCV*, 2014.

[29] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. Bilinear cnn models for fine-grained visual recognition. In *ICCV*, 2015.

[30] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.

[31] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *CVPR*, 2022.

[32] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

[33] Lu Lu, Yeonjong Shin, Yanhui Su, and George Em Karniadakis. Dying relu and initialization: Theory and numerical examples. *arXiv:1903.06733*, 2019.

[34] Namuk Park and Songkuk Kim. How do vision transformers work? In *ICLR*, 2022.

[35] Suo Qiu. Global weighted average pooling bridges pixel-level localization and image-level classification. *arXiv:1809.08264*, 2018.

[36] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[37] Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? In *NeurIPS*, 2021.

[38] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *ICML*, 2019.

[39] Tal Ridnik, Gilad Sharir, Avi Ben-Cohen, Emanuel Ben-Baruch, and Asaf Noy. Ml-decoder: Scalable and versatile classification head. In *WACV*, 2023.

[40] Jongbin Ryu, Dongyoon Han, and Jongwoo Lim. Gramian attention heads are strong yet efficient vision learners. In *ICCV*, 2023.

[41] Jongbin Ryu, Gitaek Kwon, Ming-Hsuan Yang, and Jongwoo Lim. Generalized convolutional forest networks for domain generalization and visual recognition. In *ICLR*, 2019.

[42] Jongbin Ryu, Ming-Hsuan Yang, and Jongwoo Lim. Dft-based transformation invariant pooling layer for visual classification. In *ECCV*, 2018.

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

[44] Shuyang Sun, Jiangmiao Pang, Jianping Shi, Shuai Yi, and Wanli Ouyang. Fishnet: A versatile backbone for image, region, and pixel level prediction. In *NeurIPS*, 2018.

[45] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.

[46] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019.

[47] Mingxing Tan and Quoc Le. Efficientnetv2: Smaller models and faster training. In *ICML*, 2021.

[48] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. In *NeurIPS*, 2021.

[49] Hugo Touvron, Piotr Bojanowski, Mathilde Caron, Matthieu Cord, Alaaeldin El-Nouby, Edouard Grave, Gautier Izacard, Armand Joulin, Gabriel Synnaeve, Jakob Verbeek, et al. Resmlp: Feedforward networks for image classification with data-efficient training. *PAMI*, 2022.

[50] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021.

[51] Hugo Touvron, Matthieu Cord, Alaaeldin El-Nouby, Piotr Bojanowski, Armand Joulin, Gabriel Synnaeve, and Hervé Jégou. Augmenting convolutional networks with attention-based aggregation. *arXiv:2112.13692*, 2021.

[52] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *ICCV*, 2021.

[53] Zhengzhong Tu, Hossein Talebi, Han Zhang, Feng Yang, Peyman Milanfar, Alan Bovik, and Yinxiao Li. Maxvit: Multi-axis vision transformer. In *ECCV*, 2022.

[54] Wenhai Wang, Jifeng Dai, Zhe Chen, Zhenhang Huang, Zhiqi Li, Xizhou Zhu, Xiaowei Hu, Tong Lu, Lewei Lu, Hongsheng Li, et al. Internimage: Exploring large-scale vision foundation models with deformable convolutions. In *CVPR*, 2023.

[55] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *ICCV*, 2021.

[56] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018.

[57] Yude Wang, Jie Zhang, Meina Kan, Shiguang Shan, and Xilin Chen. Self-supervised equivariant attention mechanism for weakly supervised semantic segmentation. In *CVPR*, 2020.

[58] Yunchao Wei, Jiashi Feng, Xiaodan Liang, Ming-Ming Cheng, Yao Zhao, and Shuicheng Yan. Object region mining with adversarial erasing: A simple classification to semantic segmentation approach. In *CVPR*, 2017.

[59] Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019.

[60] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *ECCV*, 2018.

[61] Lian Xu, Wanli Ouyang, Mohammed Bennamoun, Farid Boussaid, and Dan Xu. Multi-class token transformer for weakly supervised semantic segmentation. In *CVPR*, 2022.

[62] Yuhao Xu and Hideki Nakayama. Dct based information-preserving pooling for deep neural networks. In *ICIP*, 2019.

[63] Weihao Yu, Mi Luo, Pan Zhou, Chenyang Si, Yichen Zhou, Xinchao Wang, Jiashi Feng, and Shuicheng Yan. Metaformer is actually what you need for vision. In *CVPR*, 2022.

[64] Weihao Yu, Chenyang Si, Pan Zhou, Mi Luo, Yichen Zhou, Jiashi Feng, Shuicheng Yan, and Xinchao Wang. Metaformer baselines for vision. *PAMI*, 2023.

[65] Weihao Yu, Pan Zhou, Shuicheng Yan, and Xinchao Wang. Inceptionnext: When inception meets convnext. *arXiv*, 2023.

[66] 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 *ICCV*, 2019.

[67] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *ICLR*, 2018.

[68] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *AAAI*, 2020.

[69] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *CVPR*, 2016.

[70] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *CVPR*, 2017.

[71] Lei Zhu, Xinjiang Wang, Zhanghan Ke, Wayne Zhang, and Rynson WH Lau. Biformer: Vision transformer with bi-level routing attention. In *CVPR*, 2023.