

Group Generalized Mean Pooling for Vision Transformer

Byungsoo Ko¹, Han-Gyu Kim², Byeongho Heo³,
Sangdoon Yun³, Sanghyuk Chun³, Geonmo Gu¹, Wonjae Kim³

¹NAVER Vision, ²NAVER Clova Speech, ³NAVER AI Lab

Abstract

Vision Transformer (ViT) extracts the final representation from either class token or an average of all patch tokens, following the architecture of Transformer in Natural Language Processing (NLP) or Convolutional Neural Networks (CNNs) in computer vision. However, studies for the best way of aggregating the patch tokens are still limited to average pooling, while widely-used pooling strategies, such as max and GeM pooling, can be considered. Despite their effectiveness, the existing pooling strategies do not consider the architecture of ViT and the channel-wise difference in the activation maps, aggregating the crucial and trivial channels with the same importance. In this paper, we present Group Generalized Mean (GGeM) pooling as a simple yet powerful pooling strategy for ViT. GGeM divides the channels into groups and computes GeM pooling with a shared pooling parameter per group. As ViT groups the channels via a multi-head attention mechanism, grouping the channels by GGeM leads to lower head-wise dependence while amplifying important channels on the activation maps. Exploiting GGeM shows 0.1%p to 0.7%p performance boosts compared to the baselines and achieves state-of-the-art performance for ViT-Base and ViT-Large models in ImageNet-1K classification task. Moreover, GGeM outperforms the existing pooling strategies on image retrieval and multi-modal representation learning tasks, demonstrating the superiority of GGeM for a variety of tasks. GGeM is a simple algorithm in that only a few lines of code are necessary for implementation.

1. Introduction

Over the past several years, there has been a breakthrough of Transformer networks in the Natural Language Processing (NLP) domain [5, 19]. It brought great interests in the computer vision community to exploit the Transformer architecture for vision tasks. As a result, Vision Transformer (ViT) [21] was introduced, and its variants have shown great success in image recognition [29, 47, 66, 71], self-supervised learning [2, 3, 7, 27, 77, 78], object de-

tection [6, 24, 44, 61, 62], segmentation [9, 64], image compression [35], image retrieval [22, 23], and multi-modal representation learning [36, 49, 56]. Compared to the previous standard vision models, such as Convolutional Neural Networks (CNNs), recent works have demonstrated that ViT shows equal or superior performance in image recognition tasks when equipped with huge number of parameters or large-scale datasets [21, 57].

The ViT at early stage followed conventional architecture of Transformers used for NLP, letting the final state of an additional class token [19] be the feature representation. However, using the class token as a feature representation can ignore per-token information. Thus, there has been active studies on pooling strategies in NLP, aggregating per-token information as an alternative of the class token [25, 43, 59]. For ViT, recent works exploit the average pooling for achieving better performances than the class token [12, 52, 53], or preserving such per-token information [2, 3, 27, 77]. Alternatively, we can explore the representative pooling strategies studied in CNNs, which are reported to be effective: *i.e.*, max [58, 65] and Generalized Mean (GeM) [55] poolings. However, they have not been built with consideration of ViT architecture and do not consider channel-wise differences in the activation maps, aggregating crucial and trivial channels with the same importance.

In this paper, we present Group Generalized Mean (GGeM) pooling as a simple yet powerful pooling strategy for ViT. GGeM divides channels into multiple groups in the final activation maps and computes GeM pooling with a shared pooling parameter per group (illustrated in Fig. 1). As shown in Fig. 2, each channel in the final activation maps activates a different spatial area. GGeM considers such channel-wise differences by exploiting different trainable parameters for each group. Moreover, we have analyzed how pooling strategies work for ViT and discover the followings: 1) the pooling parameters decide the degree of concentration of gradient, 2) the grouping in GGeM leads to a lower inter-head similarity, and 3) the higher pooling parameter increases the number of heads focusing on global information. Compared to the existing

pooling strategies, GGeM shows a 0.1%p to 0.7%p performance boost and achieves a new state-of-the-art performance for ViT-Base and ViT-Large on ImageNet-1K classification tasks. Additionally, experiments on image retrieval and multi-modal representation learning demonstrate the versatility of GGeM for various vision tasks.

2. Related Work

Vision Transformer and Self-supervised Learning. Inspired by its great success in NLP tasks, ViT [21] has been introduced by following the conventional architecture of NLP Transformer, using a class token for final feature representation. Recently, a number of studies [8, 29, 47, 66, 71] have explored to improve ViT in terms of recognition performance and training efficiency. DeiT [66] exploits the distillation scheme for better training efficiency, using both class and distillation tokens for feature representation. Moreover, ViT has been actively used for Self-Supervised Learning (SSL) task [2, 3, 7, 27, 72, 77, 78]. Recent ViT based SSL approaches, such as MAE [27], SimMIM [77], Beit [3] and Data2Vec [2], feed direct loss to patch tokens for each objective (*i.e.*, pixel-wise reconstruction or masked image modeling). As the patch token contains rich spatial information, those SSL methods use the average pooling of all patch tokens to aggregate the per-token information. Our goal is to propose a generic pooling method for both scratch training and fine-tuning.

Pooling Strategies have been widely studied to design CNN-based global descriptors in image retrieval tasks [1, 33, 45, 58, 65, 73]. Popular pooling strategies in CNNs include the average pooling (a.k.a. SPoC) [1] and max pooling (a.k.a. MAC) [58, 65], which average and select maximum activations on the feature map, respectively. GeM [55] has been introduced to generalize max and average pooling by a pooling parameter. As variants of such standard pooling strategies, weighted sum pooling [34], regional MAC [65], multiscale RMAC [45], and weighed GeM [73] have been introduced. There also has been an attempt to use the attention mechanism for pooling by replacing the average pooling in CNNs with a single layer of multi-head attention block [56]. However, there are limited studies on pooling strategies w.r.t. activation maps (patch tokens) in ViT. Thus, this work covers exploring how standard pooling strategies work in ViT.

Group-wise Computation. The concept of groups has been widely studied for CNN. Group convolution has been proposed in AlexNet [41] to distribute the model over two GPUs. ResNeXt [76] presents a module that splits channel dimensions into groups as group convolution for better performance under similar computational costs. Mo-

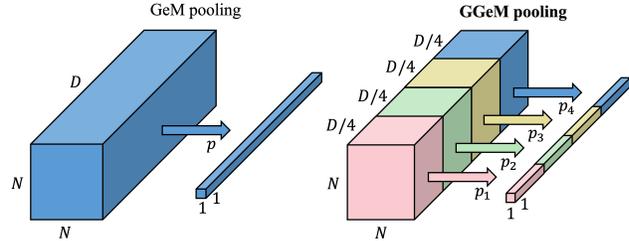


Figure 1. **Comparison between GeM and GGeM pooling.** Given $N \times N \times D$ activation maps, GeM aggregates them with a single pooling parameter p , while GGeM performs group-wise aggregation with different pooling parameter p_i for each group. In this case, the number of group G is 4.

bileNet [30] and Xception [11] adopt depth-wise separable convolutions, which are group convolutions with the same group number as the channel number. Group Normalization [74] divides the channels into groups and computes the mean and variance within each group for normalization. Likewise, the concept of the grouping can be found in Transformer. In a Transformer block, the channels of intermediate representations are divided by the number of heads (*i.e.*, channel-wise grouping) and computes attention within each head. GGeM shares the same spirit of dividing channels into groups, but it performs GeM pooling within each group for effective aggregation of per-token information.

3. Method

In this section, we propose GGeM, a pooling method for considering channel-wise differences of the activation maps by grouping channels. We first formulate ViT architecture (Sec. 3.1) and revisit representative pooling strategies studied in CNNs (Sec. 3.2). Next, we present the motivation and details of GGeM (Sec. 3.3), and analyze how pooling strategies work and affect ViT (Sec. 3.4).

3.1. Vision Transformer

This paper considers Vision Transformer models, such as ViT-Small, ViT-Base, or ViT-Large, for pooling strategies. Given an image $I \in \mathbb{R}^{H \times W \times C}$, the image tensor is reshaped into a sequence of flattened 2D patches $I' \in \mathbb{R}^{N^2 \times (R^2 C)}$, where (H, W) is the resolution of the input image, C is the number of channels, (R, R) is the resolution of each image patch, and (N, N) is the number of patches for height and width¹. Then, the sequence of patches maps to D dimensional feature by a trainable linear projection, and a learnable embedding is attached to the sequence of embedded patches as BERT’s $[class]$ token. Then, the embeddings are fed into the multiple Transformer

¹The original ViT [21] uses $I' \in \mathbb{R}^{(NM) \times (R^2 C)}$, where $(NM) = HW/R^2$ to allow non-square images. In this paper, we assume square images for simplicity.

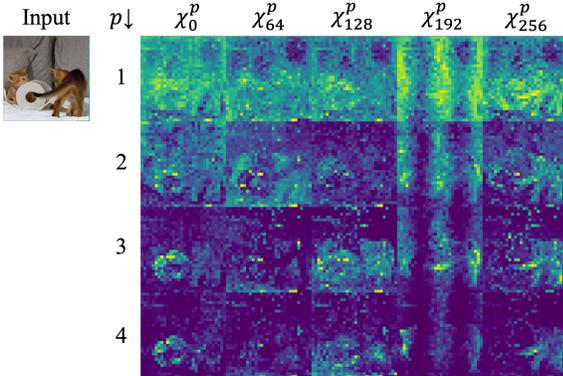


Figure 2. **Visualization of the activation map** \mathcal{X}_d^p . We use ImageNet-1K trained models using GeM pooling with fixed p . Each row indicates pooling parameter p , where $p = 1$ equals to average pooling. Each column indicates the index of random channel dimension d .

blocks, and the last Transformer block outputs $\mathbb{R}^{(N^2+1) \times D}$, where 1 is for the class token. Usually, the $1 \times D$ class token is used for the final feature representation [21, 66]. However, selecting the class token as the final representation loses the per-token information of the remaining N^2 token embeddings. Hence, we are interested in how to aggregate the per-token information of the $\mathbb{R}^{N \times N \times D}$ patch tokens.

3.2. Pooling Strategies

Average Pooling. We denote $\mathcal{X} \in \mathbb{R}^{N \times N \times D}$ as a 3D tensor of activation maps and \mathcal{X}_d as the set of $N \times N$ activations of a feature map $d \in \{1 \dots D\}$. Given the activation maps \mathcal{X} , we aggregate the spatial information into a feature representation $\mathbf{v} \in \mathbb{R}^D$, an output of a pooling process. The average pooling [1] has been a standard pooling strategy for CNNs and also has been actively exploited in ViT [2, 3, 27, 77]. Let $|\mathcal{X}_d|$ be the number of elements in the set \mathcal{X}_d , then the feature representation in the case of the conventional average pooling is as follows:

$$\mathbf{v}^{(a)} = [v_1^{(a)}, \dots, v_d^{(a)}, \dots, v_D^{(a)}]^T, \quad v_d^{(a)} = \frac{1}{|\mathcal{X}_d|} \sum_{x \in \mathcal{X}_d} x. \quad (1)$$

Max Pooling. Instead of taking the average of the feature map, the max pooling chooses the highest activation in the feature map to capture the most distinctive representation. The feature representation obtained by the conventional global max pooling [58, 65], is defined by:

$$\mathbf{v}^{(m)} = [v_1^{(m)}, \dots, v_d^{(m)}, \dots, v_D^{(m)}]^T, \quad v_d^{(m)} = \max_{x \in \mathcal{X}_d} x. \quad (2)$$

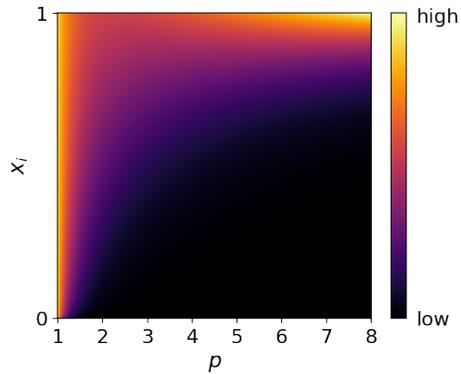


Figure 3. **The heatmap of partial derivatives** $\partial v_d^{(g)} / \partial x_i$. The feature before a pooling x_i and pooling parameter p are uniformly generated from 0 to 1 and 1 to 8, respectively.

Generalized Mean Pooling. By exploiting the generalized mean [20], the Generalized Mean Pooling (GeM) [55] has been introduced as:

$$\mathbf{v}^{(g)} = [v_1^{(g)}, \dots, v_d^{(g)}, \dots, v_D^{(g)}]^T, \quad v_d^{(g)} = \left(\frac{1}{|\mathcal{X}_d|} \sum_{x \in \mathcal{X}_d} x^{p_d} \right)^{\frac{1}{p_d}}, \quad (3)$$

where pooling parameter p_d is either trainable or fixed. In practice, a shared pooling parameter p is used for all channels as $p_d = p, \forall d \in \{1, \dots, D\}$. This is because using different pooling parameters per channel is reported to be distractive for training [55]. Thus, the term ‘‘GeM’’ in this paper without extra explanation denotes the typical GeM that uses a shared pooling parameter p for all channels. The average pooling and the max pooling are special cases of GeM pooling: GeM becomes average pooling when $p_d = 1$; GeM becomes max pooling when $p_d \rightarrow \infty$.

3.3. Group Generalized Mean Pooling

The role of pooling parameter p in GeM is to amplify the discrepancy of activations within the feature map. We visualize randomly selected feature maps \mathcal{X}_d^p of ImageNet-1K [18] trained models using GeM pooling with fixed pooling parameter p . As shown in Fig. 2, each channel dimension contains different spatial information. Moreover, the feature map of the average pooling ($p = 1$) shows activations in diverse regions, while it is shown that the larger p becomes, the more locally activation map responses. With these observations, our motivation is to stress important channels while suppressing trivial channels using different pooling parameters p_d for channels.

We propose Group Generalized Mean pooling (GGeM). GGeM divides the channels into groups and computes GeM with the pooling parameter p_i per group. Here, we assume

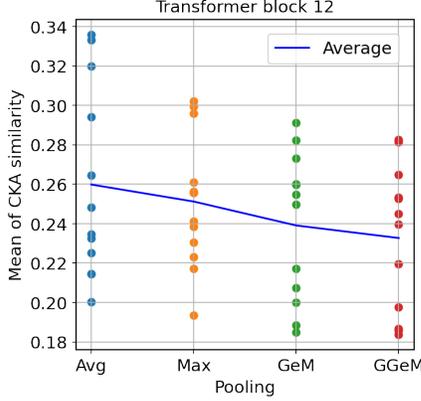


Figure 4. **Mean of CKA similarities between heads of MHA.** Each dot denotes mean of CKA similarities between the query head and other heads. The blue line denotes average of all CKA similarities between heads.

that each group of channels is ordered sequentially along the D axis. The number of groups G is a pre-defined hyper-parameter, where we suggest using the number of heads in ViT (*i.e.*, 6 for ViT-S, 12 for ViT-B, and 16 for ViT-L) as G . Let $\mathcal{P} = \{p_1, \dots, p_G\}$ be a set of trainable parameters p_i , the feature vector ($v^{(gg)}$) obtained by GGeM is defined by:

$$v^{(gg)} = [v_1^{(gg)}, \dots, v_d^{(gg)}, \dots, v_D^{(gg)}]^T, \quad (4)$$

$$v_d^{(gg)} = \left(\frac{1}{|\mathcal{X}_d|} \sum_{x \in \mathcal{X}_d} x^{p_{y(d)}} \right)^{\frac{1}{p_{y(d)}}}, \quad y(i) = \left\lceil \frac{i}{D/G} \right\rceil,$$

where $\lceil \cdot \rceil$ is the ceiling operation and D/G is the number of channels per group. GGeM with $G = 1$ is equivalent to GeM with a shared pooling parameter p for all channels. On the other hand, GGeM with $G = D$ contains D pooling parameters $\{p_1, \dots, p_D\}$, one for each channel. The comparison between GeM and GGeM is illustrated in Fig. 1.

3.4. Analysis

Gradient Flow. In order to find out the relationship between the feature before a pooling x_i ($i \in \{1, \dots, N^2\}$) and the feature after a pooling $v_d^{(g)}$ during the back-propagation, we analyze the partial derivative of the pooled feature $v_d^{(g)}$ with respect to the input of pooling layer x of Eq. 3. Such partial derivative $\partial v_d^{(g)} / \partial x_i$ is computed as follows [55]:

$$\frac{\partial v_d^{(g)}}{\partial x_i} = \frac{1}{|\mathcal{X}_d|} v_d^{(g)1-p_d} x_i^{p_d-1}. \quad (5)$$

According to Eq. 5, because all inputs share the same $|\mathcal{X}_d|$ and the same $v_d^{(g)}$, a larger input x_i achieves a larger derivative $\partial v_d^{(g)} / \partial x_i$ compared to that of a smaller input. Besides,

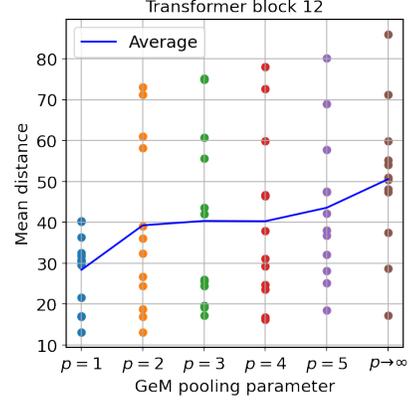


Figure 5. **Attention head mean distances with different GeM pooling parameters.** Higher mean distance indicates more globally attending heads. $p = 1$ and $p \rightarrow \infty$ are average and max pooling, respectively.

such relative difference of the derivatives caused by input difference will become much larger with a larger p_d . The growth of the relative difference is proportional to $p_d - 1$ power of the input difference because of the existence of the term $x_i^{p_d-1}$ in Eq. 5. Such phenomenon implies that: for a smaller p_d , gradients would be evenly assigned to each input; for a larger p_d , more gradient is assigned to the input with a larger value. In other words, *the pooling parameter p_d decides the degree of concentration of gradient.* The same conclusion also can be found in Fig. 3, where the heatmap of derivatives is provided with the input x_i uniformly generated from 0 to 1, and p uniformly generated from 1 to 8.

Impact of Grouping. For the well-trained Multi-Head Attention (MHA) layer of a transformer, the inter-head similarity will be small so that different heads cover different subspaces [7, 17, 48, 67]. However, MHA does not always guarantee independently acting heads focusing on different subspaces [17]. As all heads share the same p , average, max and GeM pooling can not guide each channel to have different characteristics, which is not beneficial for leading heads to make different subspaces. However, GGeM, which uses different p_i per head, is beneficial for inducing heads to learn characteristics of different subspaces because different heads can achieve gradients with different degrees of concentration. In order to show such characteristics of GGeM, we have analyzed the similarity of heads by taking the average over 2,000 input images on ImageNet-1K trained ViT-B/16. We use Centered Kernel Alignment (CKA) similarity [37] for the analysis, which is used to measure the similarity of different layers [57]. As shown in Fig. 4, GGeM, which uses different p_d for different heads, shows smaller inter-head similarity compared to other pooling methods.

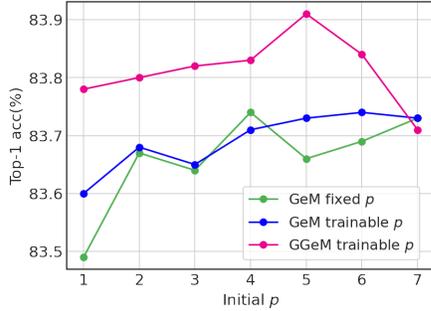


Figure 6. **Impact of initial p .** We differentiate initial p for GeM with fixed p , GeM with trainable p and GGeM with trainable p .

This implies that *GGeM is beneficial for inducing the heads to learn different representation subspaces*. Similar patterns stand out in the later Transformer blocks (*i.e.*, Transformer blocks 10 to 12). See the Appendix for the different blocks.

The weights of each head of MHA are tightly coupled as they are trained in a bundle. However, for GGeM that uses independent p_d for all dimensions (*i.e.*, $G = D$), the different degrees of gradient concentration for the different dimensions of the head can disturb the “training in a bundle” process. On the contrary, optimal GGeM with the number of groups equal to the number of heads uses one p_i per head, which will keep the weights of each head to be trained in a bundle. In conclusion, GGeM is more beneficial for performance improvement compared to other types of GeM methods in ViT.

Attention Head Mean Distance. In order to understand how pooling strategies affect ViT models, we analyze self-attention layers following [21,57]. Each self-attention layer consists of multiple self-attention heads as channel-wise grouping, and each head aggregates information from other spatial locations by attention mechanism. Here, we compute the head mean distance between a query patch position and the other positions it attends to. In detail, we weight the pixel distances with the attention weights for each attention head. Then we average it per head over 2,000 input images by using ImageNet-1K trained ViT-B/16. Interestingly, we see a clear pattern in the head mean distances as shown in Fig. 5. By increasing the pooling parameter p from average ($p = 1$) to max ($p \rightarrow \infty$) pooling, the average of all head mean distance increases, which indicates the number of heads attending on global information increases. While average pooling more attends on local information compared to other pooling methods, *the increase of p diversifies the role of the heads attending on both local and global information*. Similar patterns can be observed in the later Transformer blocks (*i.e.*, Transformer block 10 to 12). See the Appendix for the different blocks.

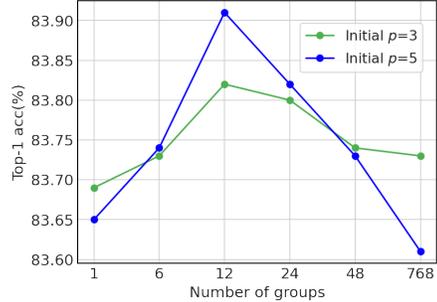


Figure 7. **Impact of number of groups.** $G = 1$ is the GeM pooling, while $G = 12$ corresponds to the number of heads in ViT-B/16.

4. Experiments

In this section, we validate the proposed GGeM on different vision tasks. We conduct ablation studies on the main properties and robustness of GGeM, and show image classification performance on scratch training, fine-tuning, linear probing and partial fine-tuning (Sec. 4.1). Next, we perform experiments on image retrieval (Sec. 4.2) and multi-modal representation learning (Sec. 4.3).

4.1. Image Classification

We perform an ablation study of GGeM and evaluate the image classification performance of different strategies on the ImageNet-1K [18] dataset. For training from scratch, we set the training epoch to 300 epochs. For fine-tuning a self-supervised model, the training epoch is set to 100 epochs and 50 epochs for ViT-B/16 and ViT-L/16, respectively. As a baseline model for the ablation study, we use MAE [27] pre-trained ViT-B/16 model unless otherwise noted in the experiment. We report top-1 validation accuracy of a 224×224 cropped image. Experimental details are in the Appendix.

4.1.1 Main Properties

Impact of Initial p . We differentiate initial p for GeM with fixed p , GeM with trainable p , and GGeM with trainable p . As shown in Fig. 6, the performances are affected by the initial p . GGeM pooling consistently outperforms GeM with fixed p and GeM with trainable p , except for the case of $p = 7$. GeM with fixed p and trainable p seems to have similar performance patterns. In all three cases, performances are increased by increasing the initial p and dropped after the optimal p . The optimal value of the initial p can vary by the models and the tasks, but the range of 3 to 5 shows stable performance for GGeM.

Impact of Number of Groups. Fig. 7 shows the influence of the number of groups G by different initial power

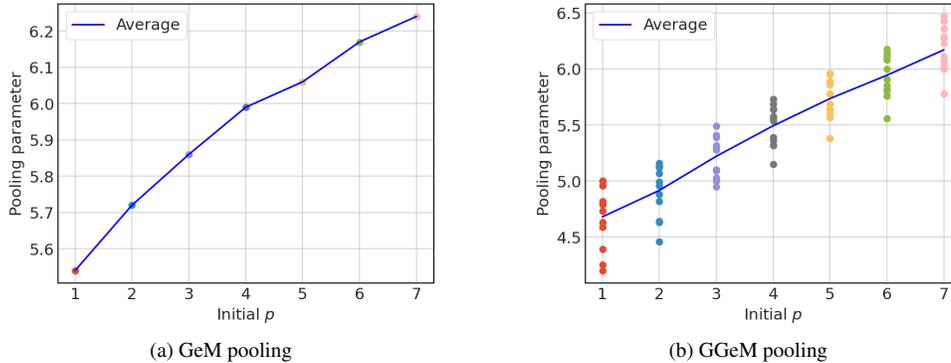


Figure 8. **Trained pooling parameters based on initial p .** We visualize trained pooling parameters by different initial p on GeM and GGeM pooling. GeM pooling has a single shared parameter, while GGeM pooling has 12 pooling parameters for ViT-B/16.

p . $G = 1$ is the GeM pooling with a single pooling parameter for all channel dimensions, while $G = 768$ indicates using different pooling parameters for each channel dimension. The performances for initial $p = 3$ and $p = 5$ increase until the G equals to the number of heads ($G = 12$ for ViT-B/16), then it drops until $G = 768$. In other words, the optimal G is the number of heads in ViT. The results are linked to the analysis in Sec. 3.4. Because of the MHA architecture, using multiple p within a head can be harmful by the different degrees of gradient concentration for the different dimensions of the head, while using a single p for all channel dimensions cannot be beneficial for achieving a small inter-head similarity.

Trained Pooling Parameters As the pooling parameter p_i is trainable, the trained value will be different from the initial value. We visualize how the pooling parameter changes with different initial values for GeM and GGeM pooling. As shown in Fig. 8, the trained pooling parameter gets higher with the increase of the initial value, and every trained pooling parameter converges between 4 and 6.5. When the initial value is 1, it increases until around 5, while the initial value of 7 decreases until around 6 to 6.5. This indicates that there is a certain convergence range (*i.e.*, 4 to 6.5). The 12 pooling parameters of GGeM are distributed within a range of 1.

4.1.2 Scratch Training and Fine-tuning

In Tab. 1, we compare the pooling strategies based on the scratch training and fine-tuning on pre-trained models by Self-Supervised Learning (SSL), including MAE [27], Beit [3], SimMIM [77] and Data2Vec [2]. Based on the trained model with label supervision, we report top-1 validation accuracy and k -NN performance ($k=12$). k -NN performance is for an auxiliary metric to see the performance of feature representation itself without a linear classification

layer. For scratch training, we follow the training recipe of [27] with small modifications for stable training and better performance than [21, 66]. As general scratch training techniques use the class token for a feature representation, our class token baseline shows a decent performance compared to DeiT. While average and GeM pooling outperform the class token, max pooling shows mixed results on ViT-S/16 and ViT-B/16. Overall, GGeM shows the best performance among all, having performance boosts between 0.3%p and 0.7%p.

For fine-tuning experiments, we fine-tune ImageNet-1K with SSL pre-trained models, which use the average pooling for baseline. We follow the same fine-tuning recipe for each work and report our reproduced score (“Average pooling”) and reported score (“Reported”). Based on the reproduced experiment, we differentiate pooling strategies. Max pooling shows similar performances to the baseline, while GeM pooling shows better performances. In comparison with Average pooling, GGeM shows the minimum of 0.2%p and the maximum of 0.4%p performance improvement on ViT-S/16 and ViT-B/16. The results show that GGeM is a superior pooling strategy in both scratch training and fine-tuning in image classification task.

4.1.3 Linear Probing and Partial Fine-tuning

We conduct linear probing and partial fine-tuning experiments based on MAE pre-trained ViT-B/16 model. For linear probing, we train a linear classifier on frozen blocks. For partial fine-tuning, we fine-tune the last several layers based on other frozen blocks by following [27]. For both GeM and GGeM, initial p is set to 3. Tab. 2 shows the results. Notably, all pooling methods are less linearly separable compared to the class token for linear probing. However, pooling methods generally outperform for a few blocks of training (*i.e.*, 1 to 4), which indicates the pooling methods perform well when non-linear layers are tuned. For

Method	Class		Average		Max		GeM		GGeM		Reported
	Acc.	k -NN	Acc.	k -NN	Acc.	k -NN	Acc.	k -NN	Acc.	k -NN	Acc.
ViT-S/16											
Scratch	80.0	79.2	80.4	80.2	77.1	76.3	80.6	80.4	80.7	80.5	79.9 [‡]
ViT-B/16											
Scratch	82.4	81.8	82.4	82.2	82.6	82.4	82.6	82.6	82.7	82.7	81.8 [‡]
MAE [27]	-	-	83.5	83.0	83.5	83.1	83.7	83.2	83.9	83.3	83.6
BeiT [†] [3]	-	-	83.6	83.4	83.5	83.1	83.7	83.5	83.8	83.6	83.7
SimMIM [77]	-	-	83.7	83.5	83.6	83.1	83.9	83.5	84.0	83.6	83.8
Data2Vec [2]	-	-	84.0	83.6	83.7	83.2	84.1	83.7	84.3	83.8	84.2
ViT-L/16											
MAE [27]	-	-	85.8	85.7	85.7	85.6	85.8	85.8	86.0	85.9	85.9
BeiT [†] [3]	-	-	85.9	85.8	86.0	85.8	86.1	85.9	86.2	86.0	86.0
Data2Vec [2]	-	-	86.5	86.4	86.6	86.4	86.6	86.4	86.7	86.5	86.6

Table 1. **Scratch training (Scratch) and fine-tuning with SSL models for ImageNet-1K classification.** Every SSL model are pre-trained with ImageNet-1K, except BeiT [3] with [†] used ImageNet-22K. We report top-1 accuracy (Acc.) and k -NN score based on trained model with label supervision. “Reported” indicates reported score in the original paper. [‡] denotes scores from DeiT [66].

Pooling	# blocks fine-tuned							
	0	1	2	4	6	8	10	12
Class	67.8	75.2	79.1	82.0	82.9	83.2	83.4	83.4
Average	66.6	77.5	80.4	82.2	82.9	83.3	83.4	83.5
Max	59.1	77.2	79.8	81.9	82.9	83.2	83.2	83.5
GeM	65.9	78.6	80.7	82.4	83.1	83.4	83.6	83.7
GGeM	65.9	79.0	80.8	82.5	83.2	83.5	83.7	83.9

Table 2. **Linear probing and partial fine-tuning.** We fine-tune partial Transformer blocks with different poolings. Tuning 0 block is linear probing and tuning 12 blocks is full fine-tuning.

the number of blocks 6 to 12, class token shows similar performance with average and max pooling. GeM and GGeM share similarly high performance, while GGeM shows the best performances for all partial fine-tuning cases. The results demonstrate that GGeM can be used for competitive representations for non-linear layer tuning.

4.1.4 Robustness

We conduct an experiment to verify how robust pooling methods are to uncommon examples. We quantitatively benchmark robustness in background change [75] with fine-tuned models on MAE. Ideally, robust models can handle background variations and locate discriminative foreground parts. With such motivation, the background change benchmark introduces ImageNet-9 (IN-9) dataset. IN-9 includes 9 coarse-grained classes and 7 variants, which are generated by mixing images from different foregrounds and backgrounds. Only-FG (O.F.), Mixed-Same (M.S.), Mixed-Rand (M.R.), and Mixed-Next (M.N.) are 4 variants with the original foreground and the modified background. No-FG (N.F.), Only-BG-B (O.BB.), and Only-BG-T (O.BT.)

Pooling	O.F.	M.S.	M.R.	M.N.	N.F.	O.BB.	O.BT.	IN-9	Mean
Class	81.3	84.3	77.5	75.0	47.4	18.8	12.9	94.0	61.7
Average	81.5	84.1	77.3	75.2	47.5	19.2	13.3	93.8	61.5
Max	81.6	84.2	77.6	76.4	48.4	18.8	12.7	94.0	61.7
GeM	81.5	84.6	78.0	76.7	47.9	19.3	13.3	93.9	61.9
GGeM	82.0	85.1	78.2	76.8	48.6	19.7	13.5	94.3	62.0

Table 3. **Robustness evaluation of pre-trained models against background change.**

are 3 variants with the masked foreground. As demonstrated in Tab. 3, GGeM shows the best performance in all variants with the highest mean performance. The results signify that the training with GGeM pooling can result in a robust model for uncommon examples.

4.2. Image Retrieval

Pooling strategies are actively studied in image retrieval tasks to find the best spatial information aggregator. We fine-tune MAE [27] pre-trained ViT-B/16 models on CUB200 [68], Cars196 [38], and Stanford Online Products (SOP) [51] with Norm-softmax loss [69, 70]. For the fair comparison, we use three different metrics: Recall@1, mean Average Precision (mAP), and R-Precision (RP) as recall-based metrics can lead to an unreliable conclusion compared to precision-based metrics [13, 14, 50]. We use initial $p = 3$ for both GeM and GGeM. As shown in Tab. 5, Max, GeM and GGeM pooling show higher performances compared to class token and average pooling, while GGeM pooling achieves the best for every benchmark. The results support the superiority of GGeM in aggregating per-token information for better feature representation.

	ImageNet	Cifar10	Cifar100	CLEVR-C	DTD	EuroSAT	FER2013	Food101	GTSRB	MNIST	RESISC45	StanfordCars	STL10	Mean
Class	11.8	42.4	16.2	8.8	6.8	14.6	15.1	8.0	6.4	6.3	19.1	0.8	69.7	17.4
Average	12.9	44.9	19.4	11.2	8.0	15.6	13.9	8.0	5.5	10.8	19.7	1.1	69.1	18.5
Max	12.6	42.5	16.4	13.0	10.0	18.9	13.1	8.2	6.6	4.4	21.7	0.8	66.4	18.1
GeM	13.6	48.4	16.5	12.2	7.9	19.8	15.7	8.4	9.9	6.8	19.2	0.8	69.5	19.1
<i>GGeM</i>	13.0	49.6	18.0	15.6	8.5	13.7	21.0	9.0	8.2	10.0	20.4	0.9	71.5	19.9

(a) Zero-shot classification

	Image-to-Text						Text-to-Image					
	Flickr30k			MSCOCO			Flickr30k			MSCOCO		
	R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10
Class	19.8	42.5	53.9	8.0	23.2	33.4	13.5	31.5	41.4	6.7	19.3	27.5
Average	20.4	42.8	54.3	9.6	25.2	35.6	14.7	31.9	41.3	7.2	19.8	28.1
Max	17.5	40.4	53.7	9.9	25.7	35.7	14.0	32.7	42.9	7.6	20.2	29.1
GeM	19.7	45.8	57.3	10.7	26.9	36.9	14.2	33.5	43.2	7.9	20.7	29.4
<i>GGeM</i>	20.4	45.3	56.7	11.5	27.7	37.9	14.4	33.7	44.0	8.4	21.3	30.3

(b) Zero-shot cross-modal retrieval

Table 4. Multi-modal representation learning with different pooling methods.

Pooling	CUB200			Cars196			SOP		
	R@1	RP	mAP	R@1	RP	mAP	R@1	RP	mAP
Class	71.9	42.4	31.7	92.1	46.8	37.9	82.7	60.5	57.6
Average	71.3	41.4	30.7	92.3	47.2	38.2	82.9	61.0	58.1
Max	74.6	43.6	33.1	93.2	49.1	40.6	83.0	61.2	58.3
GeM	75.0	44.1	34.1	93.8	50.6	42.5	83.7	62.0	59.2
<i>GGeM</i>	75.2	45.0	34.6	94.1	51.0	43.0	84.8	62.1	59.3

Table 5. Image retrieval with different pooling methods.

4.3. Multi-modal Representation Learning

Large-scale vision-language pre-training models, such as CLIP [56] and ALIGN [31], has demonstrated success over various downstream tasks. ViT-based CLIP models use the class token as image representation in the visual encoder and [EOS] (end of a sentence) token as text representation in the text encoder. Pooling strategies can be unsuitable for the text encoder because text inputs have different lengths, and the remaining tokens are filled with padding [PAD]. Thus, we differentiate pooling strategies in the visual encoder (ViT-B/32) and trained the multi-modal model with the Conceptual Captions (CC3M) dataset [60]. We use following benchmark datasets for evaluating zero-shot classification: ImageNet [18], Cifar10 [40], Cifar100 [40], CLEVR Counts (CLEVR-C) [32], Describable Textures Dataset (DTD) [15], EuroSAT [28], FER2013 [26], Food101 [4], GTSRB [63], MNIST [42], RESISC45 [10], StanfordCars [39], STL10 [16]. For zero-shot cross-modal retrieval, we use Flickr30k [54] and MSCOCO [46] benchmark datasets. The initial p is set to 3 for both GeM

and GGeM. The results are given in Tab. 4. In the zero-shot classification tasks, the best scores for each benchmark dataset vary by the choice of pooling strategies, but GGeM shows the best performance on average among all. Moreover, GGeM shows mixed performances in MSCOCO, while showing the best performance in Flickr30k. The results signify the superiority of GGeM for training large-scale vision-language pre-training tasks.

5. Conclusion

In this paper, we have introduced GGeM pooling, which considers channel-wise differences in the activation maps. GGeM aggregates per-token information of ViT by dividing channels into groups and computing GeM pooling with shared pooling parameter per group. Based on our analysis, grouping channels as the number of heads in ViT can induce heads to learn different characteristics from each other. We have shown that GGeM outperforms existing pooling strategies for scratch training and fine-tuning in image classification while showing potential to be used for other tasks such as image retrieval and multi-modal representation learning. The limitation of our work includes GGeM can work differently on ViT variants, which use a different number of heads within a model (*i.e.*, [47]). Moreover, GGeM is limited to consider channel-wise difference within the heads because of the shared pooling parameter per group. Exploring GGeM on other ViT variants and considering channel-wise difference even within the heads will be interesting future research directions.

References

- [1] Artem Babenko and Victor Lempitsky. Aggregating deep convolutional features for image retrieval. In *Proc. ICCV*, 2015. 2, 3
- [2] Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. Data2vec: A general framework for self-supervised learning in speech, vision and language. *arXiv preprint arXiv:2202.03555*, 2022. 1, 2, 3, 6, 7
- [3] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. In *Proc. ICLR*, 2022. 1, 2, 3, 6, 7
- [4] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101—mining discriminative components with random forests. In *Proc. ECCV*, 2014. 8
- [5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Proc. NeurIPS*, 2020. 1
- [6] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Proc. ECCV*, 2020. 1
- [7] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *Proc. ICCV*, 2021. 1, 2, 4
- [8] Chun-Fu Richard Chen, Quanfu Fan, and Rameswar Panda. Crossvit: Cross-attention multi-scale vision transformer for image classification. In *Proc. ICCV*, 2021. 2
- [9] Jieneng Chen, Yongyi Lu, Qihang Yu, Xiangde Luo, Ehsan Adeli, Yan Wang, Le Lu, Alan L Yuille, and Yuyin Zhou. Transunet: Transformers make strong encoders for medical image segmentation. *arXiv preprint arXiv:2102.04306*, 2021. 1
- [10] Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 2017. 8
- [11] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proc. CVPR*, 2017. 2
- [12] Xiangxiang Chu, Zhi Tian, Bo Zhang, Xinlong Wang, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Conditional positional encodings for vision transformers. *arXiv preprint arXiv:2102.10882*, 2021. 1
- [13] Sanghyuk Chun, Wonjae Kim, Song Park, Minsuk Chang, and Seong Joon Oh. Eccv caption: Correcting false negatives by collecting machine-and-human-verified image-caption associations for ms-coco. *arXiv preprint arXiv:2204.03359*. 7
- [14] Sanghyuk Chun, Seong Joon Oh, Rafael Sampaio De Rezende, Yannis Kalantidis, and Diane Larlus. Probabilistic embeddings for cross-modal retrieval. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 7
- [15] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *Proc. CVPR*, 2014. 8
- [16] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proc. AISTATS*, 2011. 8
- [17] Jean-Baptiste Cordonnier, Andreas Loukas, and Martin Jaggi. Multi-head attention: Collaborate instead of concatenate. *arXiv preprint arXiv:2006.16362*, 2020. 4
- [18] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proc. CVPR*, 2009. 3, 5, 8
- [19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *ACL*, 2019. 1
- [20] Piotr Dollár, Zhuowen Tu, Pietro Perona, and Serge Belongie. Integral channel features. 2009. 3
- [21] 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 *Proc. ICLR*, 2021. 1, 2, 3, 5, 6
- [22] Shiv Ram Dubey, Satish Kumar Singh, and Wei-Ta Chu. Vision transformer hashing for image retrieval. *arXiv preprint arXiv:2109.12564*, 2021. 1
- [23] Alaaeldin El-Nouby, Natalia Neverova, Ivan Laptev, and Hervé Jégou. Training vision transformers for image retrieval. *arXiv preprint arXiv:2102.05644*, 2021. 1
- [24] Yuxin Fang, Bencheng Liao, Xinggang Wang, Jiemin Fang, Jiyang Qi, Rui Wu, Jianwei Niu, and Wenyu Liu. You only look at one sequence: Rethinking transformer in vision through object detection. *Proc. NeurIPS*, 2021. 1
- [25] Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*, 2021. 1
- [26] Ian J Goodfellow, Dumitru Erhan, Pierre Luc Carrier, Aaron Courville, Mehdi Mirza, Ben Hamner, Will Cukierski, Yichuan Tang, David Thaler, Dong-Hyun Lee, et al. Challenges in representation learning: A report on three machine learning contests. In *Proc. NeurIPS*, 2013. 8
- [27] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2021. 1, 2, 3, 5, 6, 7
- [28] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2019. 8
- [29] Byeongho Heo, Sangdoon Yun, Dongyoon Han, Sanghyuk Chun, Junsuk Choe, and Seong Joon Oh. Rethinking spatial dimensions of vision transformers. In *Proc. ICCV*, 2021. 1, 2
- [30] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 2

- [31] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *Proc. ICML*, 2021. 8
- [32] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proc. CVPR*, 2017. 8
- [33] HeeJae Jun, Byungsoo Ko, Youngjoon Kim, Insik Kim, and Jongtaek Kim. Combination of multiple global descriptors for image retrieval. *arXiv preprint arXiv:1903.10663*, 2019. 2
- [34] Yannis Kalantidis, Clayton Mellina, and Simon Osindero. Cross-dimensional weighting for aggregated deep convolutional features. In *Proc. ECCV*, 2016. 2
- [35] Jun-Hyuk Kim, Byeongho Heo, and Jong-Seok Lee. Joint global and local hierarchical priors for learned image compression. In *Proc. CVPR*, 2022. 1
- [36] Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision. In *Proc. ICML*, 2021. 1
- [37] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International Conference on Machine Learning*, pages 3519–3529. PMLR, 2019. 4
- [38] Jonathan Krause, Jia Deng, Michael Stark, and Fei-Fei Li. Collecting a large-scale dataset of fine-grained cars. In *Proc. ICCV-W*, 2013. 7
- [39] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proc. ICCV-W*, 2013. 8
- [40] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, UT, 2009. 8
- [41] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. NeurIPS*, 2012. 2
- [42] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998. 8
- [43] Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. On the sentence embeddings from pre-trained language models. *arXiv preprint arXiv:2011.05864*, 2020. 1
- [44] Yanghao Li, Hanzi Mao, Ross Girshick, and Kaiming He. Exploring plain vision transformer backbones for object detection. *arXiv preprint arXiv:2203.16527*, 2022. 1
- [45] Yang Li, Yulong Xu, Jiabao Wang, Zhuang Miao, and Yafei Zhang. Ms-rmac: Multiscale regional maximum activation of convolutions for image retrieval. *IEEE Signal Processing Letters*, 2017. 2
- [46] 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 *Proc. ECCV*, 2014. 8
- [47] 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 *Proc. ICCV*, 2021. 1, 2, 8
- [48] Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? *Proc. NeurIPS*, 2019. 4
- [49] Norman Mu, Alexander Kirillov, David Wagner, and Saining Xie. Slip: Self-supervision meets language-image pre-training. *arXiv preprint arXiv:2112.12750*, 2021. 1
- [50] Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. A metric learning reality check. In *Proc. ECCV*, 2020. 7
- [51] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *Proc. CVPR*, 2016. 7
- [52] Zizheng Pan, Bohan Zhuang, Jing Liu, Haoyu He, and Jianfei Cai. Scalable vision transformers with hierarchical pooling. In *Proc. ICCV*, 2021. 1
- [53] Namuk Park and Songkuk Kim. How do vision transformers work? *arXiv preprint arXiv:2202.06709*, 2022. 1
- [54] Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proc. ICCV*, 2015. 8
- [55] Filip Radenović, Giorgos Tolias, and Ondřej Chum. Fine-tuning cnn image retrieval with no human annotation. *PAMI*, 41(7), 2018. 1, 2, 3, 4
- [56] 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 *Proc. ICML*, 2021. 1, 2, 8
- [57] Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? *Proc. NeurIPS*, 2021. 1, 4, 5
- [58] Ali S Razavian, Josephine Sullivan, Stefan Carlsson, and Atsuto Maki. Visual instance retrieval with deep convolutional networks. *ITE Transactions on Media Technology and Applications*, 4(3):251–258, 2016. 1, 2, 3
- [59] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019. 1
- [60] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018. 8
- [61] Hwanjun Song, Deqing Sun, Sanghyuk Chun, Varun Jampani, Dongyoon Han, Byeongho Heo, Wonjae Kim, and Ming-Hsuan Yang. An extendable, efficient and effective transformer-based object detector. *arXiv preprint arXiv:2204.07962*, 2022. 1
- [62] Hwanjun Song, Deqing Sun, Sanghyuk Chun, Varun Jampani, Dongyoon Han, Byeongho Heo, Wonjae Kim, and Ming-Hsuan Yang. Vidt: An efficient and effective fully transformer-based object detector. In *Proc. ICLR*, 2022. 1

- [63] Johannes Stallkamp, Marc Schlipf, Jan Salmen, and Christian Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural networks*, 2012. 8
- [64] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. In *Proc. ICCV*, 2021. 1
- [65] Giorgos Tolias, Ronan Sifre, and Hervé Jégou. Particular object retrieval with integral max-pooling of cnn activations. In *Proc. ICLR*, 2016. 1, 2, 3
- [66] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *Proc. ICML*, 2021. 1, 2, 3, 6, 7
- [67] Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. *arXiv preprint arXiv:1905.09418*, 2019. 4
- [68] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical report, California Institute of Technology, 2011. 7
- [69] Feng Wang, Jian Cheng, Weiyang Liu, and Haijun Liu. Additive margin softmax for face verification. *IEEE Signal Processing Letters*, 2018. 7
- [70] Feng Wang, Xiang Xiang, Jian Cheng, and Alan Loddon Yuille. Normface: L2 hypersphere embedding for face verification. 2017. 7
- [71] 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 *Proc. ICCV*, 2021. 1, 2
- [72] Chen Wei, Haoqi Fan, Saining Xie, Chao-Yuan Wu, Alan Yuille, and Christoph Feichtenhofer. Masked feature prediction for self-supervised visual pre-training. *arXiv preprint arXiv:2112.09133*, 2021. 2
- [73] Xiaomeng Wu, Go Irie, Kaoru Hiramatsu, and Kunio Kashino. Weighted generalized mean pooling for deep image retrieval. In *Proc. ICIP*, 2018. 2
- [74] Yuxin Wu and Kaiming He. Group normalization. In *Proc. ECCV*, 2018. 2
- [75] 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. 7
- [76] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proc. CVPR*, 2017. 2
- [77] Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. Simmim: A simple framework for masked image modeling. In *Proc. CVPR*, 2022. 1, 2, 3, 6, 7
- [78] Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. ibot: Image bert pre-training with online tokenizer. *arXiv preprint arXiv:2111.07832*, 2021. 1, 2

Group Generalized Mean Pooling for Vision Transformer

Supplementary Material

Contents

A Analysis Details	1
A.1. Additional Results on Impact of Grouping . . .	1
A.2. Additional Results on Attention Head Mean Distance	1
B Implementation Details	1
B.1. Pseudo Code	1
B.2. Experimental Setting	1
C Additional Experiments	3
C.1. Visualization	3

A. Analysis Details

A.1. Additional Results on Impact of Grouping

To analyze the impact of grouping in GGeM, we compute the similarity of heads in ViT. We use Centered Kernel Alignment (CKA) similarity [4] which is designed to compare representations within and across networks quantitatively. CKA similarity takes representations of two layers, $\mathbf{X} \in \mathbb{R}^{(s \times n_1)}$ and $\mathbf{Y} \in \mathbb{R}^{(s \times n_2)}$, where s and n_i are number of samples and neurons, respectively. Given $\mathbf{G} = \mathbf{X}\mathbf{X}^T$ and $\mathbf{H} = \mathbf{Y}\mathbf{Y}^T$ as the Gram matrices for the two layers, CKA similarity is defined by:

$$CKA(\mathbf{G}, \mathbf{H}) = \frac{HSIC(\mathbf{G}, \mathbf{H})}{\sqrt{HSIC(\mathbf{G}, \mathbf{G})HSIC(\mathbf{H}, \mathbf{H})}}, \quad (i)$$

where $HSIC$ is the Hilbert-Schmidt independence criterion [2]. In addition to the Fig. 4, we further visualize the mean of CKA similarities between heads in Transformer blocks 7 to 12. As shown in Fig. A, GGeM shows smaller inter-head similarity compared to other pooling methods in Transformer blocks 9 to 12. The closer the Transformer block to the pooling layer, the larger such pattern stands out. The results demonstrate that GGeM is helpful for inducing the heads to learn different representation subspaces.

A.2. Additional Results on Attention Head Mean Distance

We analyze how pooling strategies affect ViT models by comparing attention head mean distances. In addition to

Algorithm 1 GGeM: PyTorch-like Pseudo-code

```
# x: output tensor of the last Transformer block (
  batch size x tokens x dimension)
# p_params: pooling parameters, the number of
  pooling parameters is the same with groups
# groups: number of groups
# eps: minimum value for clamping
def GGeM(x):
    x = x[:, 1:, :] # Remove class token

    b, t, d = x.shape
    e = d // groups
    x = x.reshape((b, t, e, groups))

    x = x.clamp(min=eps).pow(p_params)
    x = x.mean(dim=1)
    x = x.pow(1./p_params)

    x = x.reshape((b, d))
    return x
```

Notes: `clamp` is a function to clamp all elements into the range `[min, max]`.

the Fig. 5, we further compute attention head mean distances of Transformer blocks 7 to 12. As shown in Fig. B, the mean distances increase by the pooling parameter p increases. Such a phenomenon gets more noticeable when the Transformer block is higher and closer to the pooling layer. The results show that the larger p increases the number of heads attending on the global information.

B. Implementation Details

B.1. Pseudo Code

The proposed GGeM is a simple algorithm that only a few lines of code are necessary for implementation. We include PyTorch-like pseudo-code of GGeM in Algorithm 1. When `groups=1`, the algorithm works as GeM pooling. GGeM with `groups=1` and fixed `p_params=1` will work as Average pooling. We set `groups` as the number of heads in each ViT architecture and have a shared pooling parameter per group. The `clamp` function is for ensuring every elements are larger than zero.

B.2. Experimental Setting

The default setting for ImageNet classification experiments is in Tab. A. For scratch training, we follow the training setting of [3] with modification of the learning rate for a better baseline. The exponential moving average is used

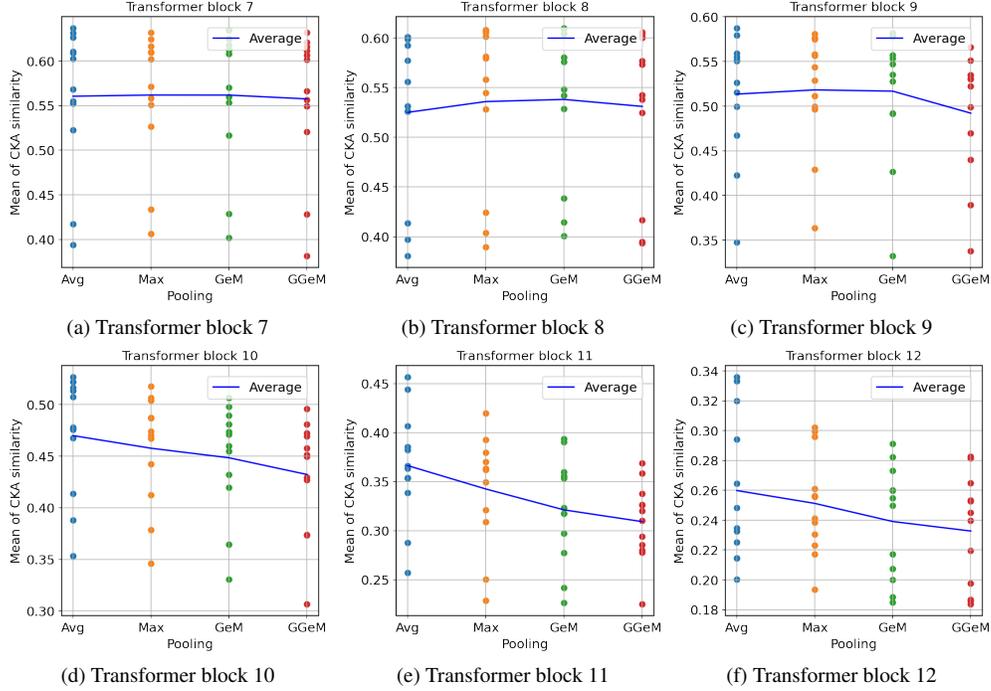


Figure A. **Mean of CKA similarities between heads of MHA.** We visualize Transformer blocks 7 to 12 of ViT-B/16. Each dot denotes mean of CKA similarities between the query head and other heads. The blue line denotes average of all CKA similarities between heads.

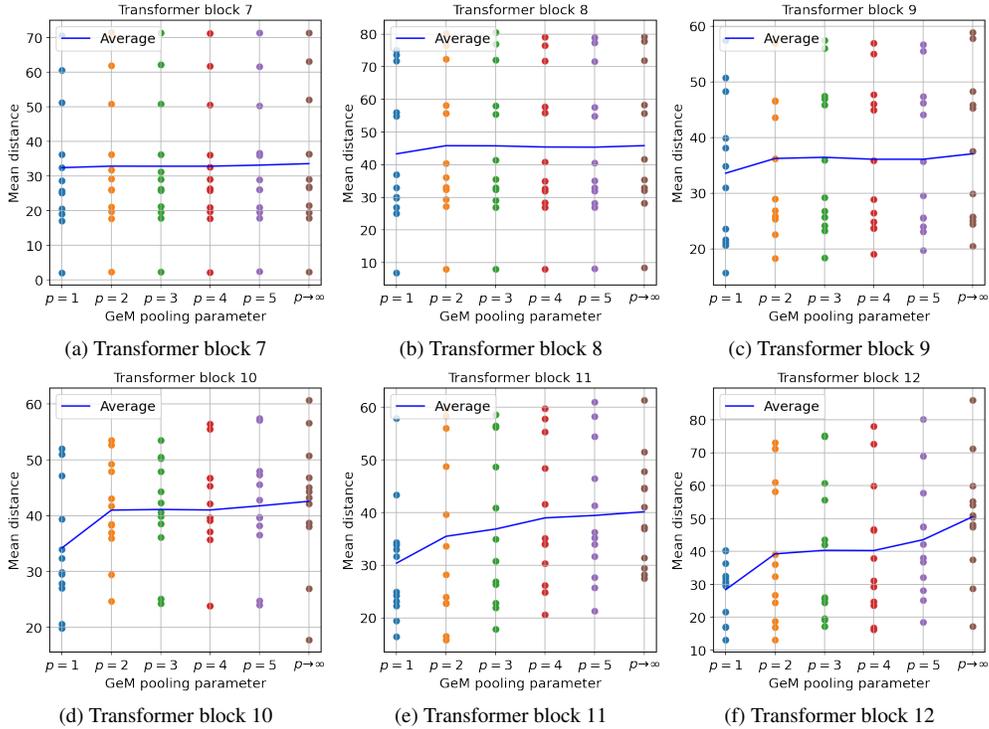


Figure B. **Attention head mean distances with different GeM pooling parameters.** We visualize Transformer blocks 7 to 12 of ViT-B/16. Higher mean distance indicates more globally attending heads. $p=1$ and $p \rightarrow \infty$ are average and max pooling, respectively.

Config	Scratch	MAE	BeiT	SimMIM	Data2Vec	Linear probing
Initial p	5	5 (B), 2 (L)	5	4	3	3
Optimizer	AdamW	AdamW	AdamW	AdamW	AdamW	LARS
Learning rate	8e-3	2e-3	4e-3	5e-3	4e-3 (B), 5e-3 (L)	0.64
Weight decay	0.3	0.05	0.05	0.05	0 (B), 0.05 (L)	0
Optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.95$	$\beta_1, \beta_2 = 0.9, 0.999$	0.9			
Layer decay	0.75	0.65 (B), 0.75 (L)	0.65 (B), 0.75 (L)	0.65	0.65	-
Batch size	4096	1024	1024	2048	1024	16384
LR schedule	cosine decay	cosine decay	cosine decay	cosine decay	cosine decay	cosine decay
Warmup epochs	20	5 (B), 10 (L)	20 (B), 5 (L)	20	10 (B), 5 (L)	10
Traning epochs	300	100 (B), 50 (L)	100 (B), 50 (L)	100	100 (B), 50 (L)	90
Augmentation	RandAug (9, 0.5)	RandAug (9, 0.5)	RandAug (9, 0.5)	RandAug (9, 0.5)	RandAug (9, 0.5)	RandomResizedCrop
Label smoothing	0.1	0.1	0.1	0.1	0.1	-
Mixup	0.8	0.8	0.8	0.8	0.8	-
Cutmix	1.0	1.0	1.0	1.0	1	-
Drop path	0.1	0.1 (B), 0.2 (L)	0.1 (B), 0.2 (L)	0.1	0.2 (B), 0.25 (L)	-
Exp. moving average	0.9999	-	-	-	-	-

Table A. Hyper-parameter setting for ImageNet-1K classification.

Config	Value	Config	Value
Initial p	3	Initial p	3
Model	ViT-B/16	Model	ViT-B/32
Optimizer	AdamW	Optimizer	AdamW
Learning rate	2e-3	Learning rate	1e-3
Weight decay	0.05	Weight decay	0.2
Optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.999$	Optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.98$
Layer-wise LR decay	0.65	Batch size	4080
Batch size	128	Learning rate schedule	cosine decay
Learning rate schedule	cosine decay	Warmup steps	3500
Warmup epochs	5	Traning epochs	25
Traning epochs	100 (Cars, CUB), 50 (SOP)	Augmentation	RandomResizedCrop
Augmentation	RandAug (9, 0.5)		
Drop path	0.1		
Base learning rate	4e-3		
Proxy LR scale	100		

(a) Image retrieval setting

(b) Multi-modal representation learning setting

Table B. Hyper-parameter settings for image retrieval and multi-modal representation learning.

only in scratch training. For fine-tuning, we follow the same training recipe of each corresponding work (MAE, BeiT, SimMIM, and Data2Vec) based on each pre-trained model and sweep over different initial p (1 to 5) to find the best performance for GeM and GGeM. We conduct the linear probing experiment by following [1], while we use the same training setting of MAE for partial fine-tuning.

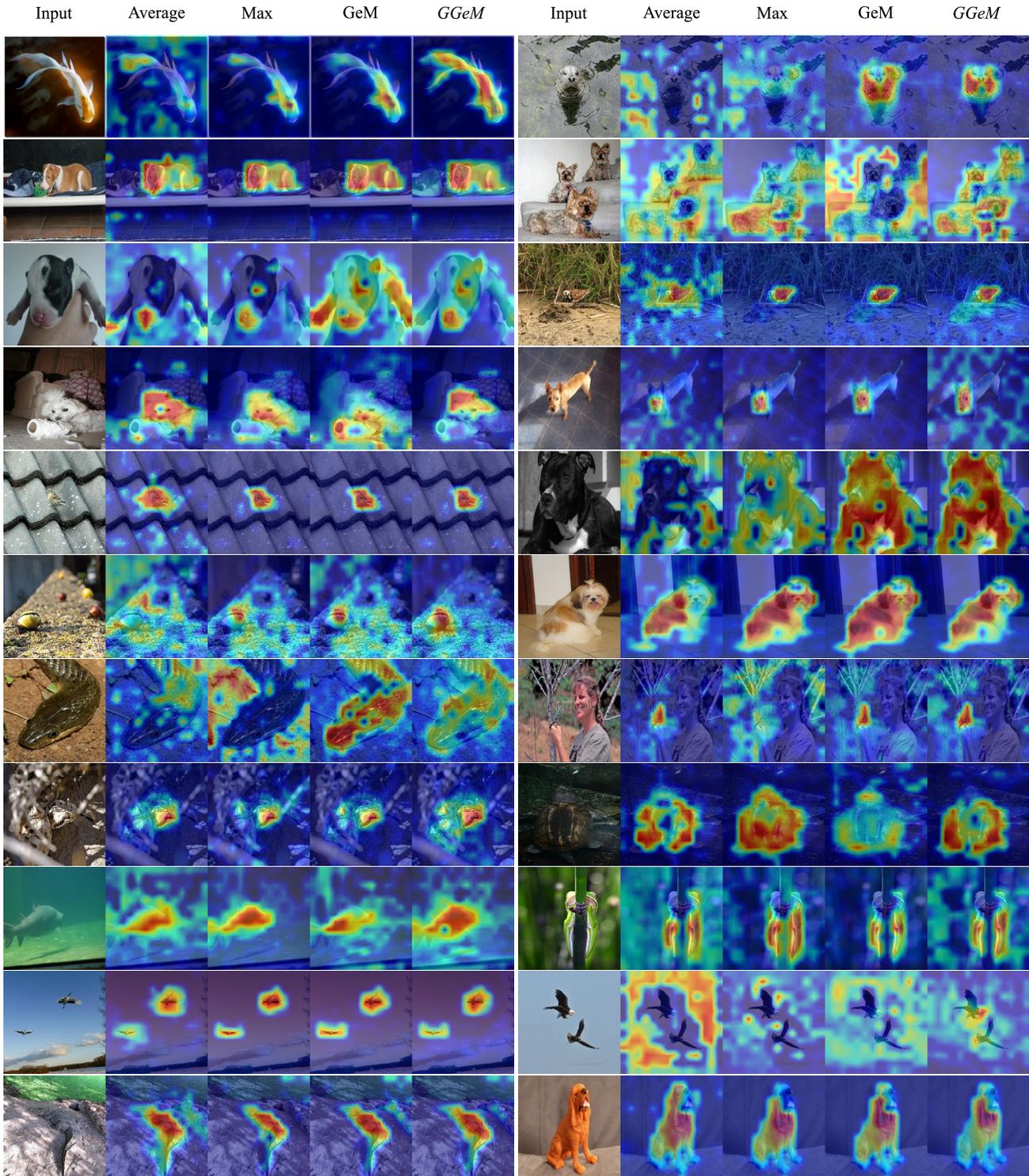
The hyper-parameter settings of image retrieval and multi-modal representation learning are in Tab. B. We conduct the image retrieval experiment based on MAE fine-tuning code. We use Norm-softmax loss [9, 10] as a baseline objective function while scaling the learning rate of proxy (proxy LR scale) with 100 for better baseline performance. For multi-modal representation learning, we use the same model architecture of CLIP [7] and the experimental setting of SLIP [5]. We use ViT-B/32 image encoder and the Transformer-based text encoder [8] with 49,152 vocab size

of byte pair encoding (BPE) tokenizer. The max sequence length was clipped by 76 by following CLIP. We implement all models using the PyTorch framework [6] and use four $8 \times$ A100-80GB servers for experiments.

C. Additional Experiments

C.1. Visualization

We visualize Class Activation Map (CAM) [12] to see how models trained with different pooling methods make specific decisions based on the input image. For the CAM method, Score-CAM [11] is used, which is a variant of CAM which removes dependence on gradients. We use ViT-B/16 fine-tuned models based on MAE, and images are randomly selected from the ImageNet validation set. The results are shown in Fig. C.



References

- [1] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *Proc. ICCV*, 2021. 3
- [2] Arthur Gretton, Kenji Fukumizu, Choon Teo, Le Song, Bernhard Schölkopf, and Alex Smola. A kernel statistical test of independence. *Proc. NeurIPS*, 2007. 1
- [3] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2021. 1
- [4] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International Conference on Machine Learning*, pages 3519–3529. PMLR, 2019. 1
- [5] Norman Mu, Alexander Kirillov, David Wagner, and Saining Xie. Slip: Self-supervision meets language-image pre-training. *arXiv preprint arXiv:2112.12750*, 2021. 3
- [6] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Proc. NeurIPS*, 2019. 3
- [7] 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 *Proc. ICML*, 2021. 3
- [8] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proc. NeurIPS*, 2017. 3
- [9] Feng Wang, Jian Cheng, Weiyang Liu, and Haijun Liu. Additive margin softmax for face verification. *IEEE Signal Processing Letters*, 2018. 3
- [10] Feng Wang, Xiang Xiang, Jian Cheng, and Alan Loddon Yuille. Normface: L2 hypersphere embedding for face verification. 2017. 3
- [11] Haofan Wang, Zifan Wang, Mengnan Du, Fan Yang, Zijian Zhang, Sirui Ding, Piotr Mardziel, and Xia Hu. Score-cam: Score-weighted visual explanations for convolutional neural networks. In *Proc. CVPR-W*, 2020. 3
- [12] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proc. CVPR*, 2016. 3