Improved Image Classification With Token Fusion

KEONG-HUN CHOI, JIN-WOO KIM, YAO WANG, AND JONG-EUN HA

1Graduate School of Automotive Engineering, Seoul National University of Science and Technology, Seoul 01811, South Korea
2Ways1, Uiwang-si, Gyeonggi-do 16006, South Korea
3Department of Mechanical and Automotive Engineering, Seoul National University of Science and Technology, Seoul 01811, South Korea

Corresponding author: Jong-Eun Ha (jeha@seoultech.ac.kr)

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (2020R1A2C1013335).

Abstract In this paper, we propose a method to improve image classification performance using the fusion of CNN and transformer structure. In the case of CNN, information about a local area on an image can be extracted well, but global information extraction is limited. On the other hand, the transformer has an advantage in global information extraction, but it requires much memory compared to CNN. We apply CNN on an image and consider the feature vector of each pixel on the resulting feature map by CNN as a token. At the same time, the image is divided into patches, and each patch is considered a token, like a transformer. Tokens by CNN and transformer have advantages in extracting local and global information, respectively. We assume that the combination of these two types of tokens will have an improved characteristic, and we show it through experiments. We propose three methods to fuse tokens having different characteristics: (1) late token fusion with parallel structure, (2) early token fusion (3) token fusion in layer-by-layer. The proposed method shows the best classification performance in experiments using ImageNet-1K.

Index Terms Image classification, transformer, convolutional neural networks, deep learning.

I. INTRODUCTION

After the advent of deep learning, convolutional neural networks (CNNs) have become dominant approaches to image classification, object detection, and semantic segmentation [1], [2], [3], [4], [5]. Recently, transformers [6], [7] that use various attention mechanisms have become mainstream for natural language processing (NLP) tasks. Applying transformers in the vision domain also shows promising results in multiple tasks, including image classification [8], [9], object detection [10], [11], and semantic segmentation [12]. Vision transformer (ViT) [9] first showed that an image patch could be regarded as a word token in NLP. The design of ViT is used for low-level vision [13] and semantic segmentation in SETR [12].

CNNs extracts information from image regions that masks cover. Large mask sizes are required to extract information considering large areas on image. But, it requires a large amount of memory. Meanwhile, transformers can reflect global interaction through attention. Image is divided into patches, and relation among patches are reflected by attention. But, it has shortcoming that it cannot reflect the information within patch itself like CNN’s mask. We want to merge advantages of CNN and transformer in one framework.

This paper proposes a method that integrates two types of tokens for image classification. The first token type is derived from an image patch like the ViT [9]. The second token type is derived from feature vectors on feature maps by ResNet [2]. We propose a new structure that integrates CNN and transformer. Models for various types of fusing two tokens are presented. We investigate three types of fusion methods including late token fusion, early token fusion, and layer-by-layer token fusion. Experimental results show the improved image classification by the proposed algorithm.

II. RELATED WORKS

CNNs have become mainstream in various applications in computer vision. It has the form of deep neural networks where convolutional layers are stacked in a serial or parallel fashion. Though they showed promising results, they still needed help extracting global features due to the limited size of convolutional kernels. Meanwhile, transformers based
on self-attention and cross-attention offer an advantage in extracting global features compared to CNNs.

A. CONVOLUTIONAL NEURAL NETWORKS

LeCun et al. [14] proposed the first CNN for handwritten character recognition. AlexNet [15] showed a dramatic performance improvement on large-scale image classification tasks [16], and it has opened a full-blown prelude in deep learning. Soon after, various types of newly designed structures based on CNNs, such as VGG [5], GoogleNet [17], Inception [1], and ResNet [2], appeared. ResNet [2] proposed a method for the model to have deep layers leveraging residual connections.

Before the full-scale use of attention in Transformers, some approaches [18], [19], [20] have used attention mechanisms in CNNs. Wang et al. [18] used attention modules sequentially between the intermediate stages of deep residual networks by stacking them. SENet [19] and GENet [20] use adaptive channel-wise feature responses leveraging interdependencies among channels. NLNet [21] introduced self-attention into neural networks, which guarantees reflecting long-range dependencies by pairwise interactions across all spatial positions.

Also, research has been done on downsizing CNNs with the tradeoff between accuracy and efficiency [22], [23], [24]. Neural architecture search (NAS) was used to find an efficient structure for CNNs with reduced computation time in MobileNets [4], [25] and EfficientNets [3].

B. VISION TRANSFORMERS

After remarkable achievements by Transformers in natural language processing (NLP) [6], [7], many types of research [8], [9], [26], [27], [28], [29], [30], [31], [32], [33], [34], [35] have focused on introducing transformer-like structures to various vision tasks.

The pioneering work of ViT [9] uses transformers in NLP on images by considering an image patch as a word token. ViT [9] used a sizeable private dataset JFT-300M [36] to train the model. DeiT [8] proposed an algorithm to train ViT in a data-efficient manner using ImageNet-1K [16]. In T2T-ViT [28], visual tokens are embedded by the recursive aggregation of neighboring tokens into one token. TNT [30] uses the inner and outer transformer blocks to represent an image’s patch-level and pixel-level characteristics.

PVT [37] applied a pyramid structure onto ViT [9] to use multiple-scale feature maps, which is adequate for dense pixel-level prediction tasks. In CPVT [26] and CvT [31], an algorithm that integrates CNNs and transformers using a convolutional projection into a transformer block is proposed. CMT [38] further extends the integration of CNNs and transformers by investigating different components, such as shortcut and normalization functions.

Previous researches [26], [31], [38] about integrating CNNs and transformers focuses on a model structure by intermixing CNN layers and attention blocks. In this paper, we treat CNN and transformer as an independent components. Three types of fusion method including late token fusion, early token fusion, and layer-by-layer token fusion.

III. PROPOSED METHOD

The proposed method considers the feature map of CNN as a token and fuses it with a token of an image patch in ViT. CNN has the advantage of extracting local features on the image. On the other hand, it isn’t easy to reflect the global features on the entire image due to the limitation of the size of the window kernel. If the kernel size is increased, it is possible to extract global feature values, but there is a difficulty due to a memory problem. In the case of Transformers, it is possible to use image patches as tokens to reflect the global relationship between patches through the attention mechanism. However, it is challenging to extract local feature values within the patch. Also, a memory problem follows if the patch size decreases.

The proposed method uses feature maps produced by CNN. We consider a feature vector of each pixel on feature maps as a token. It can be regarded as reflecting local feature values through the kernel. Also, we use the patches as tokens after dividing the image into patches like ViT [9]. The proposed method uses these two different types of tokens. When combining two tokens, various combinations are possible depending on the difference in the token combination position and the number of tokens. In this paper, we analyze the results according to these combinations through experiments and try to find the configuration that provides the best results.

Figure 1 shows the overall structure of the proposed method. The part marked with a dotted line is a part that can be optionally selected. Token by CNN and ViT is configured to combine before, within, or after the transformer. The right side of Figure 1 shows the standard components in the Transformer.

The Transformer uses a 1D sequence of token embeddings as input. A 2D image \( x \in \mathbb{R}^{H \times W \times C} \) is reshaped into a sequence of flattened patches \( x_p \in \mathbb{R}^{N \times (P^2 \cdot C)} \). \((H, W)\) is the height and width of an image, \( C \) is the number of channels. The size of an image patch is \((P, P)\) and the number of patches is \( N = HW/P^2 \). The \( i \)-th patch \( x_{p_i} \) is flattened and mapped to \( D \) dimensions by a trainable linear projection. The output of linear projection is denoted as the patch embedding. The patch embedding is added with positional encoding \( E_{pos} \) and we denote it \( t_0 \). We consider it as a token, which is used as the input of the Transformer.

\[
    t_0 = \left[ x_{p_i}^1 E, x_{p_i}^2 E, \ldots, x_{p_i}^N E \right] + E_{pos}, E \in \mathbb{R}^{(P^2 \cdot C) \times D}, E_{pos} \in \mathbb{R}^{(N \times D)}
\]

(1)

We use feature maps by the ResNet-101 model [2]. Each pixel on a feature map is considered a token like ViT [9]. Therefore, we have two types of tokens. One is derived from an image patch like ViT [9]. The other is derived from features of the ResNet-101 model [2]. These two tokens are integrated into various forms, as shown in Fig 1. We do not use a class
token in ViT [9] because we use two types of tokens. We use all tokens for a classification task. Tokens are processed like the ViT [9].

\[
t'_l = \text{MHSA} (\text{LN} (t_{l-1})) + t_{l-1}, \quad l = 1, \cdots, L
\]

\[
t_l = \text{MLP} (\text{LN} (t'_l)) + t'_l, \quad l = 1, \cdots, L
\]

MHSA, MLP, and LN stand for multi-head self-attention, multi-layer perceptron, and layer norm, respectively. \( t_l \) represents token at layer \( l \). Token at layer \( l-1 \) is converted into token at layer \( l \) through LN, MHSA, LN, and MLP in sequence.

After integrating two types of tokens, they are processed with equal procedures for the classification, as shown in Figure 1. After pooling, they are used for classification. We divide the proposed three fusion methods according to where fusion occurs. In early token fusion, integrating tokens are processed before the Transformer encoder, while late token fusion occurs after the Transformer encoder.

### A. METHOD 1: LATE TOKEN FUSION WITH PARALLEL STRUCTURE

The proposed method is to process the token by CNN and the token by image patch in parallel, respectively, and then integrate them. Each token is processed using the encoder structure of the Transformer and has a design that fuses the processing results.

Figure 2 shows the overall structure of late token fusion. We divide an image into patches like ViT [9]. Along with this, the feature maps by CNN are used in parallel. We use a pre-trained ResNet-101 model [2].

Each token passes through the corresponding Transformer encoder. The results of the two transformer encoders have different sizes. To integrate them, it is necessary to match the size of the feature values. We design late token fusion for this purpose. Figure 2(b) shows the detailed process of the late token fusion block. The size of the feature value is expanded four times by using UpConv for the token by ViT. Finally, a token by ViT and a token by CNN are generated as one token using concatenation.

### B. METHOD 2: EARLY TOKEN FUSION

The early token fusion method integrates two tokens before transformer processing. Figure 3 shows the detailed structure of the early token fusion method. It combines the result by CNN and the original image itself.

We use a pre-trained ResNet-101 model [2]. Figure 3(b) shows the bride block structure for early token fusion. The resulting feature map by ResNet-101 at each stage follows UpConv and 1 \( \times \) 1 convolution. Finally, it is concatenated with tokens by ViT. We denote this process as a bridge block, as shown in Figure 3(b). We use five bridge blocks by reflecting the ResNet-101 structure. After five bridge blocks, we finally have a unified token of size HXWX18. It can be regarded as an image and is processed separately for each patch, like ViT.

### C. METHOD 3: TOKEN FUSION IN A LAYER-BY-LAYER

The previous two methods are structures that combine tokens before or after transformer processing. The method in this section is a structure that fuses tokens during transformer processing. Figure 4 shows the overall design. It is a structure to apply the mixing block of Figure 4(b) according to the stage of ResNet-101 [2].

The mixing block contains the Transformer encoder and processing part of the ResNet-101 feature map. Finally, the results of the two sub-modules are concatenated. We use five mixing blocks corresponding to the hierarchical structure of ResNet-101 [2]. After going through five mixing blocks, we additionally use two transformer blocks. We designed for three proposed models to have a similar size to investigate the performance of each model irrespective of the model size.
For this, all three models were configured to have 12 transformer blocks.

D. IMAGE CLASSIFICATION HEAD
Information different from the existing Transformer structure is extracted through the integration of tokens by CNN and the Transformer. For this reason, the image classification head was configured differently from the model, which uses a class token in ViT [9]. Figure 5 shows the three classification structures used in this paper. Figure 5(a) is a token-wise pooling classifier, and Figure 5(b) is a channel-wise pooling classifier. Figure 5(c) is a classifier combining token and channel-wise pooling.

IV. EXPERIMENTAL RESULTS
Experiments are done using an i9 10990 CPU and two NVIDIA RTX 3090. Only ImageNet-1K was used for training data. All input images were adjusted to a size of 224(H)X224(W)X3(C). We use ResNet-101 [2] as a pretrained model. Therefore, all input images were adjusted to a size of 256(H)X256(W)X3(C). Then, the center region having a size of 224(H)X224(W)X3(C) is cropped, and normalization is applied using the mean value of (0.485, 0.456, 0.406) and standard deviation value of (0.229, 0.224, 0.225). Horizontal and vertical flips were applied to the original image for data augmentation. Adding rotation distortion up to 15 degrees to the original image is also used.

A. EXPERIMENTAL RESULTS USING BASIC THREE MODELS
Table 1 shows the experimental results of the proposed three fusion methods. Training is done using the ImageNet-1K. The early token fusion method gives the lowest score. There is a limitation to improving the score because the rest of the model structure is the same as the existing ViT [9] except for the token fusion. The layer-by-layer channel-wise pooling method shows the best results. We conclude that the best performance by the layer-by-layer fusion method is obtained by reinforcing integration between the data inside the patch and the feature value delivered through the mixing block of Figure 4(b). In addition, a higher evaluation score could be obtained using a smaller model through this structure.

Table 2 shows the comparison results by the proposed and other methods trained only using ImageNet-1K. The best and second-best models are displayed in bold and bold italics. The same model generally performs better when using a large-size image as input. The proposed algorithm provides a state-of-the-art result. Also, the proposed algorithm uses 36% fewer parameters than the second-best algorithm of MViTv2-L [39].

Table 3 shows the experimental results using ImageNet-22K. Token fusion with the layer-by-layer model having a channel-wise pooling classification head shows the best results among the proposed models. The MViT-XL [33] shows the best result when ImageNet-22K is only used for training. The CoCa [40], which gives the best result, uses an additional dataset of JFT-3B for training.

The proposed algorithm shows results closer to the state-of-the-art using ImageNet-22K. The proposed algorithm does not give such an improvement that is noticed in other algorithms when using more training datasets, such as ImageNet-22K. This requires further investigation.

**Table 1. Results using ImageNet-1K by the proposed algorithm.**

| Method             | Classification head type | Evaluation(%) | Acc @1 | Acc @5 |
|-------------------|--------------------------|---------------|--------|--------|
| Late token fusion | Token-wise pooling       | 85.51         | 92.77  |
|                   | Channel-wise pooling     | 85.73         | 91.32  |
|                   | Mixing                   | 86.29         | 93.88  |
| Early token fusion| Token-wise pooling       | 83.01         | 91.54  |
|                   | Channel-wise pooling     | 84.94         | 93.19  |
|                   | Mixing                   | 85.27         | 94.56  |
| Layer-by-layer    | Token-wise pooling       | 86.97         | 94.25  |
|                   | Channel-wise pooling     | 87.77         | 95.93  |
|                   | Mixing                   | 87.32         | 94.97  |
FIGURE 5. Example of pooling layer (a) Average pooling by data depth (b) Average pooling by token (c) Combined pooling data from each process.

TABLE 2. Comparison of results with other algorithms using ImageNet-1K.

| Model                  | Year | Acc@1(%) | Acc@5(%) | #Params (M) | Image size |
|------------------------|------|----------|----------|-------------|------------|
| CPVT-Ti-GAP [26]       | 2021 | 74.9     | -        | 6           | 224x224    |
| DenseNet-169 [41]      | 2017 | 76.2     | 93.2     | 14          | 224x224    |
| ResNet-50 [2]          | 2016 | 76.2     | 92.9     | 25.6        | 224x224    |
| DeiT-S [8]             | 2020 | 79.8     | -        | 22          | 224x224    |
| DeiT-B [8]             | 2020 | 83.1     | -        | 85.8        | 384x384    |
| EfficientNet-B1 [3]    | 2019 | 79.1     | 94.4     | 7.8         | 240x240    |
| EfficientNet-B6 [3]    | 2019 | 84.0     | 96.8     | 43          | 528x528    |
| ResNeXt-101-64x4d [42] | 2017 | 80.9     | 95.6     | 84          | 224x224    |
| T2T-ViT-19 [28]        | 2021 | 81.2     | -        | 39          | 224x224    |
| PVT-M [37]             | 2021 | 81.2     | -        | 44.2        | 224x224    |
| Swin-T [27]            | 2021 | 81.3     | -        | 29          | 224x224    |
| Swin-B [27]            | 2021 | 83.3     | -        | 88          | 224x224    |
| CvT-S [29]             | 2021 | 82.0     | 95.9     | 24.2        | 224x224    |
| CvT-S [29]             | 2021 | 83.3     | 96.5     | 24.2        | 384x384    |
| Twins-SVT-B [43]       | 2021 | 83.1     | -        | 56          | 224x224    |
| Twins-SVT-L [43]       | 2021 | 83.3     | -        | 99.2        | 224x224    |
| ViT-B/16 [9]           | 2020 | 77.9     | -        | 55.5        | 384x384    |
| TNT-B [30]             | 2021 | 82.8     | 96.3     | 65.6        | 224x224    |
| CvT-21 [31]            | 2021 | 83.3     | -        | 31.5        | 384x384    |
| BoTNet-S1-128 [44]     | 2021 | 83.5     | 96.5     | 75.1        | 256x256    |
| CMT-S [38]             | 2022 | 83.5     | 96.6     | 25.1        | 224x224    |
| CMT-L [38]             | 2022 | 84.8     | 97.1     | 74.7        | 288x288    |
| MViTv2-B [33]          | 2022 | 84.4     | -        | 52          | 224x224    |
| MViTv2-L [20]          | 2022 | **86.0** | -        | 218         | 384x384    |
| Ours                   | -    | **87.8** | 95.9     | 140         | 224x224    |

B. EXPERIMENTAL RESULTS USING MODIFIED BASIC THREE MODELS

In this section, we present experimental results with different model configurations for three types of models. Figure 6 shows three different designs in the late token fusion method of Figure 2. The UpConv layer matched the sizes of two other tokens, and the concatenation layer was used in the combining process. Additional three types of models are configured using copy and element-wise summation.

Figure 7 shows three different configurations in the early token fusion method of Figure 3. The feature values generated through the ResNet-101 [2] are combined with the original image through a bridge block. We configure three additional models by replacing the UpConv layer inside the bridge block with copy, using only one bridge block, and connecting only the last weight data.

Figure 8 shows a different configuration in the token fusion layer-by-layer of Figure 4. The token combining process is
performed through the mixing block of the feature values generated through the ResNet-101 [2]. The method proceeds without a class token, unlike ViT [9]. In the case of additional experiments, as shown in Figure 8, a class token other than the image patch was added. It was configured to exclude it from the mixing block’s combining process.
Table 3 shows experimental results using ImageNet-22K by the modified structure of the proposed model. Results in Table 2 by the basic model show better results than the modified model structure.

In our preliminary study, we notice that the proposed network does not give an improvement in object detection as in image classification. This needs further investigation.

V. CONCLUSION

We propose an improved algorithm for image classification with token fusion. The CNN can extract local features on an image, while the transformer has the advantage of detecting global features with an attention mechanism. The proposed algorithm wants to efficiently integrate the characteristics of the CNN and transformer. Two types of tokens are integrated into the Transformer structure. The first type of token is derived from an image patch. The second type of token is derived from feature maps by a CNN. We consider a feature vector per pixel on a feature map as a patch. Three models are investigated according to the order and location of token fusion. The proposed algorithm shows the state-of-the-art result for image classification on training only using ImageNet-1K. Further research will investigate applying the proposed model into object detection and semantic segmentation.

REFERENCES

[1] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 2818–2826.
[2] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 770–778.
[3] M. Tan and Q. Le, “EfficientNet: Rethinking model scaling for convolutional neural networks,” in Proc. Int. Conf. Mach. Learn., 2019, pp. 6105–6114.
[4] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, “MobileNetV2: Inverted residuals and linear bottlenecks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 4510–4520.
[5] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” 2014, arXiv:1409.1556.
[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” 2017, arXiv:1706.03762.
[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” 2018, arXiv:1810.04805.
[8] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, “Training data-efficient image transformers & distillation through attention,” 2020, arXiv:2012.12877.
[9] D. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, “An image is worth 16x16 words: Transformers for image recognition at scale,” 2020, arXiv:2010.11929.
[10] N. Carion, F. Massa, G. N. Synnaeve, U. N. S. Torr, and S. Zagaras, “End-to-end object detection with transformers,” in Proc. Eur. Conf. Comput. Vis., 2020, pp. 213–229.
[11] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, “Deformable DETR: Deformable transformers for end-to-end object detection,” 2020, arXiv:2010.04159.
[12] S. Zheng, J. Lu, H. Zhao, X. Zhu, Z. Luo, Y. Wang, Y. Fu, J. Feng, T. Xiang, P. H. S. Torr, and L. Zhang, “Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers,” 2021, arXiv:2012.15840.
[13] H. Chen, Y. Wang, T. Guo, C. Xu, Y. Deng, Z. Liu, S. Ma, C. Xu, C. Xu, and W. Gao, “Pre-trained image processing transformer,” 2020, arXiv:2003.00364.
[14] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proc. IEEE, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in Proc. Adv. Neural Inf. Process. Syst., 2012, pp. 1097–1105.
[16] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2009, pp. 248–255.
[17] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2015, pp. 1–9.
[18] W. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, “Residual attention network for image classification,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 6450–6458.
[19] H. Lu, L. Shen, and G. Sun, “Squeeze-and-excitation networks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 7132–7141.
[20] J. Hu, L. Shen, S. Albanie, G. Sun, and A. Vedaldi, “Gather-excite: Exploiting feature context in convolutional neural networks,” 2018, arXiv:1810.12348.
[21] X. Wang, R. Girshick, A. Gupta, and K. He, “Non-local neural networks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 7794–7803.
[22] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, “GhostNet: More features from cheap operations,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2020, pp. 1577–1586.
