Pyramid Transformer for Traffic Sign Detection

Omid Nejati Manzari  
*School of Electrical Engineering, Iran University of Science and Technology, Tehran, Iran*  
onmid_nejaty@elec.iust.ac.ir

Amin Boudesh  
*The Raderon Artificial Intelligence Laboratory, Burnaby, British Columbia, Canada*  
a.boudesh@modares.ac.ir

Shahriar B. Shokouhi  
*School of Electrical Engineering, Iran University of Science and Technology, Tehran, Iran*  
bshokouhi@iust.ac.ir

Abstract—Automatic detection and classification of traffic signs have become an essential asset in the visual system of autonomous vehicles and self-driving cars. Recently, vision transformers have achieved remarkable performance on various benchmarks of visual tasks. We observe that prior ViTs could not provide satisfactory gain in traffic sign detection because the dataset size of this task is very small, and the class distribution of traffic signs is highly unbalanced. To solve the problems, we propose a novel Pyramid Transformer with hybrid architecture in this paper. Specifically, Pyramid Transformer follows a hierarchical architecture to build a feature pyramid with local and global information by using atrous convolutions. Furthermore, it inherits an intrinsic inductive bias and aims to learn multi-scale feature representation for objects of varying sizes, thereby enhancing the network robustness against the size discrepancy of traffic signs. We conduct experiments on the German Traffic Sign Detection Benchmark (GTSDB), which results demonstrate the superiority of the proposed model in the traffic sign detection task. More specifically, Pyramid Transformer achieves 77.8% mAP on GTSDB when applied to the Cascade RCNN as the backbone, which surpasses all the state-of-the-art methods.

Keywords—Object Detection; Vision Transformer; Traffic Sign Detection; Self-Driving Cars

I. INTRODUCTION

Traffic sign recognition and detection (TSRD) are crucial to driver assistance systems and self-driving cars. One of the main prerequisites for the widespread implementation of TSRD is an algorithm that is robust, reliable, and high accuracy in various real-world scenarios. Besides the large variation in traffic signs to detect, the traffic images taken on the road are not ideal. They are often distorted by camera motion, challenging weather conditions, and poor lighting conditions that significantly make it difficult to deploy in real scenarios.

TSRD is still challenging study, and several works have been conducted on it. The most important of these studies can be found in [1]. TSRD consists of two independent branches: Traffic Sign Recognition (TSR) and Traffic Sign Detection (TSD). Traditionally, the methods of TSD usually depend on manual feature extraction. Features are extracted manually from different attributes, including edge detection, color information, and geometrical shapes. The color-based method generally consists of a thresholding segmentation scheme of traffic sign sections in a specific color space, such as Chroma-Hue-Luminance (CHL), Hue-Saturation-Intensity (HSI) [2], and others [3–5]. However, a major weakness of color-based methods is that they are very sensitive to the change of lighting, which can frequently happen in realistic scenarios. To address this problem, shape-based methods have been widely used in the previous literature, which includes Fast Fourier Transform (FFT) [6], Haar-Wavelet features [7], Histogram-Oriented Gradients (HOG), and Canny-Edge detection [8]. However, because of camera movements during live video stream, disorientation, scale variation, and occlusion of traffic sign regions hinder the practical application of these approaches. In contrast, TSR methodologies use Convolutional Neural Networks (CNN) to extract deep features, including MCDNN and Robust Embedded network [9]. Unfortunately, until today there is no comprehensive model to describe the best classifier and the perfect feature extraction backbone for TSR.

In this study, we particular focus on designing new backbone for traffic sign detection. As far as the authors know, no previous studies have applied Vision Transformers to this specific task. The Transformer was first introduced in the field of natural language processing (NLP), which is a deep neural network based on a self-attention mechanism. In contrast to the traditional recurrent neural networks and convolutional neural networks (CNNs), Transformer is an attention-only structure without recursion or convolution operations, which improve parallel efficiency without sacrificing the performance [10]. Vision Transformers are established as novel backbone for vision-related issues. Transformers (ViTs) have emerged with encouraging performance in computer vision tasks, and their structures are under development.
Therefore, we introduce a novel Pyramid Transformer enhanced by locality inductive bias and pyramid module, which combines two kinds of basic blocks, i.e., pyramid block (PB) and normal block (NB). PBs help to downsample and embed the input images into patches with strong multi-scale features, while NBs attempt to jointly capture local and hierarchical global attributes in the patch sequence. Furthermore, each block consists of two paralleled branches, including a paralleled local self-attention and convolution layers accompanied by a feed-forward network (FFN). Significantly, PB has an additional pyramid module, which comprises atrous convolutions with different dilation rates to capture local and global features into tokens. Experiments show that our scheme can effectively improve the detection speed and accuracy on GTSDB.

II. METHOD

This section provides a brief overview of ViT and then describes the proposed Pyramid Transformer architecture in details.

A. Revisit vision transformer

Our method is based on Vision Transformer (ViT), so we first provide a brief overview of ViT [10]. In contrast to CNN-based approaches for image classification, ViT utilizes a purely attention-based mechanism. In each transformer model, two basic components are included: a multi-head self-attention (MSA) and a feed-forward network (FFN) with layer normalization and residual shortcuts. In order to adapt transformers to vision tasks, ViT divides images into fixed-sized patches, for example $16 \times 16$, and then linearly projects them into tokens. To form the input sequence, patch tokens are additionally appended with class tokens. To learn the positional information of each token, we add a learnable absolute positional embedding before feeding it to transformer encoders. At the end of the network, the class token is used as the final feature representation. ViT can be expressed as follows:

\begin{align}
    x_0 &= [x_{\text{patch}}][x_{\text{cls}}] + x_{\text{pos}}, \\
    y_k &= x_{k-1} + \text{MSA}(\text{LN}(x_{k-1})), \\
    x_k &= y_k + \text{FFN}(\text{LN}(y_k)),
\end{align}

where $x_{\text{cls}} \in R^{1 \times C}$ and $x_{\text{patch}} \in R^{N \times C}$ represent class tokens and patch tokens, respectively, while $x_{\text{pos}} \in R^{(1+N) \times C}$ represents position embeddings, in the following equation, $K$ is the layer index, $C$ is the number of patch tokens, and $N$ is the embedding dimension. However, vanilla ViT is ideally capable of learning global interactions between all patch tokens; the memory complexity of self-attention increases when there are many tokens since the computational cost grows quadratically. As a result, the vanilla ViT model cannot extend to vision applications requiring high-resolution details, such as object detection. We propose the pyramid module for the vision transformer to address these issues.

B. Overview architecture of Pyramid transformer

Pyramid transformer aims to introduce the pyramid structure into the Vision Transformer, so that it can generate multi-scale feature maps for dense prediction tasks. As shown in Figure 1, The proposed architecture is composed of two types of blocks, PBs and NBs. The function of PBs is to embed multi-scale context into tokens, and NBs are used further to inject convolutional bias into Transformers. Fig 1 shows that Pyramid Transformer offers four stages to downsample an input image of size $x \in R^{H \times W \times C}$, where four PBs are used to reduce the features gradually by $4 \times, 2 \times, 2 \times$, and $2 \times$, respectively. At
TABLE I: Performance for object detection on the GTSDB

| Decoder       | backbone     | params (M) | AP  | AP<sup>75</sup> | AP<sup>50</sup> |
|---------------|--------------|------------|-----|---------------|----------------|
| Faster RCNN   | ResNet-50 [11] | 44         | 63.4 | 67.3          | 83.1           |
|               | PVT-S [12]   | 44         | 65.4 | 67.3          | 87.7           |
|               | Swin-T [13]  | 48         | 68.4 | 72.6          | 90.6           |
|               | DAT-T [14]   | 46         | 68.3 | 72.7          | 91.1           |
|               | Ours         | 36         | **70.2** | **74.6**   | **91.2**       |
| Cascade RCNN  | ResNet-50 [11] | 82         | 70.7 | 74.6          | 88.4           |
|               | PVT-T [12]   | 83         | 75.2 | 79.4          | 93.9           |
|               | Swin-T [13]  | 86         | 74.5 | 78.8          | 93.3           |
|               | DAT-T [14]   | 86         | 75.4 | 79.9          | 94.2           |
|               | Ours         | 74         | **77.8** | **81.8**   | **96.5**       |

each stage, several normal blocks are sequentially groped following the PB. Note that all of NBs in each group have the same isotropic design. The number of normal cells determines model parameters and size. Using this method, Pyramid transformer can extract a feature pyramid from different stages $F_1, F_2, F_3, F_4$, which is used by decoder heads designed for different downstream tasks.

C. Pyramid Block

In each stage, pyramid blocks are employed to embed local information and multiscale feature into tokens rather than immediately splitting and flattening images. It uses a linear patch embedding layer, which introduces intrinsic inductive bias scale-invariance from convolutions. In Fig 1, the PB is divided into two parallel branches, which together model long-range and local dependency, attended by an FFN that transforms the features. The image $x$ is the input for the first PB, and the input feature of $i$th PB is denoted as $h_i \in R^{H_i \times W_i \times D_i}$. In the branch of the global dependency, firstly, $h_i$ enter a Pyramid Reduction Module (PRM) for multiscale context extraction as follows:

$$PRM(f_i) = Cat([Conv_{ij}(h_i; s_{ij}; r_i)])$$  \hspace{1cm} (4)

where $Conv_{ij}$ denotes the $j$th convolutional layer in the PRM. The predefined dilation rate set $S_i$ is used to calculate $s_{ij}$ corresponding to the $i$th PB. The feature dimension is reduced by a ratio $r_i$ from the predefined reduction ratio group using stride convolution. The concatenation of the features after convolution occurs through the channel, namely, $h_{i,s}^{ms} \in R^{(W/p) \times (H/p) \times (D/s)}$, where $s$ is the number of dilation rates. Next, MHSA module then processes $h_{i,s}^{ms}$ to model long-range dependencies. Moreover, we embed local context through Parallel Convolutional Modules (PCM), which are fused as follows:

$$h_i^g = MHSA_i(Img2Seq(h_{i,s}^{ms}))$$  \hspace{1cm} (5)

$$h_i^{g'} = h_i^g + PCM_i(h_i).$$  \hspace{1cm} (6)

$PCM$ consists of sequentially stacked convolution layers, batch normalizations (BN), and an $Img2Seq$ operation. Furthermore, parallel convolutions use stride convolutions to match the PRM’s spatial downsampling ratio. As a result, the token features learned both low-level and multi-scale information, implying that PB acquires local and intrinsic inductive bias by design. FFN processes the fused tokens, and after the resulting sequence is converted to feature maps. This feature fed into the following PB or NB.

D. Normal Block

Normal blocks have a similar structure to pyramid blocks, except that there is no PRM in NBs, as shown on the right side of Fig 1. Because feature maps after PBs have relatively small spatial dimensions, PRM is not needed in NBs. The class token from the third PB is concatenated with $F$ from the third PB and then combined with positional embeddings to result the input patches for the next NBs. Class tokens are randomly generated at the beginning of the training phase and stay the same during the procedure. The tokens are pushed into the MSA module in the same way as the PB. The resulting sequence is transformed into feature maps during this process and then fed into the PRM. Notably, the class token is discarded because PRM has no spatial connection to other visual tokens. Stacked convolutions are used in PRM to further reduce the parameters in NBs. An element-wise sum is then applied to the features from MSA and PRM. Finally, the FFN processes the result tokens to obtain the output features of NB.

III. EXPERIMENTS

In this section, as a first step, we describe the datasets used for training our Pyramid Transformer, followed by an explanation of the experimental settings. In the final step, we focus on the extensive experimental results of our Transformer model compared to SOTAs studies regarding traffic sign detection.
A. Dataset and evaluation measures

We test our proposed model on the German Traffic Sign Detection Benchmark (GTSDB) [15] dataset. Several reasons explain why this dataset is chosen over others, including its popularity and frequent use in studies to compare traffic sign detection methods. The GTSDB dataset contains natural traffic scenes recorded in weather conditions (rain, fog, sunny) and various types of roads (urban, highway, rural) captured during daylight and twilight. This dataset includes 900 full images comprising 1206 traffic signs, split into a train set of 600 images (846 traffic signs) and a test set with 300 images (360 traffic signs). The images contain zero, one, or multiple traffic signs, which naturally suffer from differences in orientation, occlusions, or low-light conditions. The GTSDB has 43 classes in total, also divided into four superclasses: mandatory, prohibitory, danger, and other. For the GTSDB classes, we evaluate on the subset of the 43 traffic signs because traffic signs in the same superclass still contain different information, such as “speedlimit30” and “speedlimit70”.

The mean average precision (mAP), which is commonly used as an evaluation metric in the object detection dataset [16], is used as the standard criterion in this study. Average Precision (AP) describes the trade-off between precision and recall by describing the area under the precision-recall curve. The AP for one class object is calculated, and the mAP is the average value for all classes in the dataset. Further, the Intersection-over-Union (IoU) used in this paper has values of 0.5 and 0.75 for computing mAP.

B. Implementation details

We aim to evaluate our Pyramid Transformer by utilizing two representative meta-architectures: Faster RCNN [17] and Cascade RCNN [18]. Particularly, our transformer architecture is used to generate the backbones of these frameworks. As fair comparisons, we similarly train all the models under the scheme as in [19]. Also, we try to make compatible settings for following works. Further, we show detection results on the GTSDB dataset [48] in Table 1 with standard schedule (36epochs). The Faster RCNN train was performed in 16 batch sizes for 36 epochs by using AdamW. Initially, the learning rate is 0.0001, starting with a 600 – iteration warmup, and declining by 0.1 at epochs 7 and 11. We use weight decay 0.0002 and consistency loss regularization of 0.2. The implementation is adopted from MMDetection [20] toolboxes. According to [19], for 3x training of Cascade RCNN, we use weight decay of 0.0005 and an initial learning rate of 0.0002 for the training network. All other hyperparameters follow the default settings used by Swin [13]. We commonly follow the multi-train scheme to randomly resize the input image so that its longer dimension is less than 1333 while shorter dimension is between 800 and 480.

C. Comparison with the state-of-the-art

Table 1 summarizes the results, showing that Pyramid Transformer achieves the best performance while requiring the least number of parameters. Thanks to the introduced pyramid module and locality inductive bias, With Faster RCNN as the decoder for the 36 epochs setting, the proposed model achieved 8.1% AP\textsuperscript{50}, 7.3% AP\textsuperscript{75}, and 6.8% AP performance gains over Swin [13]. It also considerably outperforms other backbones like ResNet and PVT, owning to our model’s efficient structure design. From table 1 our method has the highest mAP when using Cascade RCNN as the decoder, obtaining 77.8% AP, 81.8% AP\textsuperscript{75}, and 96.5% AP\textsuperscript{50} when training only 36 epochs, which our method improved 2.1%mAP on Faster RCNN and 2.4% mAP on Cascade RCNN. The superiority of the Pyramid Transformer is due to the fact that introducing inductive bias into the pyramid feature module helps our model better utilize the data to further improve performance for object detection.

IV. CONCLUSION

In this paper, we introduced a novel vision transformer based on two-staged frameworks for traffic sign detection, which integrates spatial structure and local information into vision Transformers using two basic blocks (Pyramid blocks and Normal blocks). Meanwhile, the multi-scale Pyramid features from our model can be applied to various dense prediction visual tasks. Experimental evaluations on the GTSDB dataset approve that our Pyramid Transformer attains better performance compared to state-of-the-art studies. Our next steps will be to investigate more traffic signs from classes that are seldom seen in this benchmark. We also plan to enhance the speed of the process in order to run it on embedded devices in real-time.

REFERENCES

[1] A. Ellahyani, I. El Jaafari, and S. Charfi, “Traffic sign detection for intelligent transportation systems: a survey,” in E3S web of conferences, vol. 229. EDP Sciences, 2021, p. 01006.
[2] Z. Huang, Y. Yu, J. Gu, and H. Liu, “An efficient method for traffic sign recognition based on extreme learning machine,” IEEE transactions on cybernetics, vol. 47, no. 4, pp. 920–933, 2016.
[3] T. Chen and S. Lu, “Accurate and efficient traffic sign detection using discriminative adaboost and support vector regression,” IEEE Transactions on Vehicular Technology, vol. 65, no. 6, pp. 4006–4015, 2015.
[4] H. Kashiyani and S. B. Shokouhi, “Patchwise object tracking via structural local sparse appearance model,” in 2017 7th International Conference on Computer and Knowledge Engineering (ICCKE). IEEE, 2017, pp. 125–131.
[5] H. Asgarian, A. Amirkhani, and S. B. Shokouhi, “Fast drivable area detection for autonomous driving with
deep learning,” in 2021 5th International Conference on Pattern Recognition and Image Analysis (IPRIA). IEEE, 2021, pp. 1–6.

[6] F. Larsson and M. Felsberg, “Using fourier descriptors and spatial models for traffic sign recognition,” in Scandinavian conference on image analysis. Springer, 2011, pp. 238–249.

[7] F. Larsson, M. Felsberg, and P.-E. Forssen, “Correlating fourier descriptors of local patches for road sign recognition,” IET Computer Vision, vol. 5, no. 4, pp. 244–254, 2011.

[8] Y. Lu, J. Lu, S. Zhang, and P. Hall, “Traffic signal detection and classification in street views using an attention model,” Computational Visual Media, vol. 4, no. 3, pp. 253–266, 2018.

[9] O. N. Manzari and S. B. Shokouhi, “A robust network for embedded traffic sign recognition,” in 2021 11th International Conference on Computer Engineering and Knowledge (ICCKE). IEEE, 2021, pp. 447–451.

[10] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly et al., “An image is worth 16x16 words: Transformers for image recognition at scale,” arXiv preprint arXiv:2010.11929, 2020.

[11] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[12] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, “Pyramid vision transformer: A versatile backbone for dense prediction without convolutions,” in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 568–578.

[13] Z. Liu, H. Hu, Y. Lin, Z. Yao, Z. Xie, Y. Wei, J. Ning, Y. Cao, Z. Zhang, L. Dong et al., “Swin transformer v2: Scaling up capacity and resolution,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 12009–12019.

[14] Z. Xia, X. Pan, S. Song, L. E. Li, and G. Huang, “Vision transformer with deformable attention,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 4794–4803.

[15] J. Stallkamp, M. Schlipsing, J. Salmén, and C. Igel, “Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition,” Neural networks, vol. 32, pp. 323–332, 2012.

[16] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes (voc) challenge,” International journal of computer vision, vol. 88, no. 2, pp. 303–338, 2010.

[17] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” Advances in neural information processing systems, vol. 28, 2015.

[18] Z. Cai and N. Vasconcelos, “Cascade r-cnn: Delving into high quality object detection,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 6154–6162.

[19] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, “Swin transformer: Hierarchical vision transformer using shifted windows,” in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 10012–10022.

[20] K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu et al., “Mmdetection: Open mmlab detection toolbox and benchmark,” arXiv preprint arXiv:1906.07155, 2019.