Pattern Recognition 158 (2025) 110952

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

CosineTR: A dual-branch transformer-based network for semantic line detection

Yuqi Zhang[1], Bole Ma[1], Luyang Jin, Yuancheng Yang, Chao Tong *

School of Computer Science & Engineering, Beihang University, Beijing, China
State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, Beijing, China

ARTICLE INFO

Keywords:
Semantic line
Detection model
Visual property
Semantic features

ABSTRACT

Semantic line is a straight line based representation designed to well capture the spatial layout or structural shape of the scene in an image that is valuable as a high-level visual property. In this paper, we propose an efficient end-to-end trainable semantic line detection model named **Co**mplementary **s**emantic l**ine TR**ansformer (CosineTR), which is designed according to an old proverb "two heads are better than one". CosineTR adopts a dual-branch framework to detect semantic lines with a coarse to fine strategy. These two branches are built based on well-designed attention modules to capture multi-scale line semantic features locally and globally, and are equipped with heatmap prediction head and parameter regression head respectively to perform semantic line detection from two different perspectives. In addition, we introduce bilateral region attention and Gaussian prior cross-attention modules to reinforce semantic contexts extracted by the two branches, and couple them to form complementary feature representations by leveraging a feature interaction method. Extensive experiments demonstrate that our approach is effective and achieves competitive semantic line detection performance on multiple datasets.

1. Introduction

For most captured natural images, the spatial structure layout can often be divided by a set of meaningful end-to-end straight lines [1], so that the different regions correspond to distinct scene semantics. Such straight lines are defined as *semantic lines* [2,3]. Semantic lines are a set of intuitive and effective visual cues that can serve various applications, such as autonomous driving [4,5], assisted photography [6], image skew correction [7,8] and so on. It not only has practical importance, but also can be used as prior knowledge for downstream tasks related to scene perception and understanding, for instance, scene graph generation [9–11], image caption generation [12–14], etc.

Since deep Convolutional Neural Networks (CNNs) have become the dominant architectures for vision tasks and shown powerful capabilities for high-level semantic modeling [15,16], previous works on semantic line detection all choose CNN as the paradigm to build networks. For existing methods, they can be broadly divided into two types of schemes as illustrated in Fig. 1. One is multi-stage detection method based on randomly generated candidate lines [1–3], the first stage pools features in the region around candidate lines and feed them into a classifier to obtain line proposals, and the additional stages further select and remove line proposals to refine them. The other is based on deep Hough transform [17,18], which translates feature representation into Hough parametric domain, thus transforming the detection problem into a heatmap prediction problem.

However, both of these methods perform dense prediction that are susceptible to class imbalance. Considering the limited number of semantic regions contained in normal-size images, semantic lines are bound to be sparsely distributed, causing them to be easily swamped under the huge number of background samples during training phase. Moreover, previous works seldom focus on global interactions among a set of semantic lines, but treat them as independent lines for extracting and aggregating features from local contexts. As a result, the obtained semantic lines are weakly correlated and hard to form an global optimal semantic division for a given image.

In this paper, we go beyond the framework of previous studies and jump out of the idea of optimizing a single task model. Instead, inspired by ensemble learning and multi-task learning, we suggest that combining multiple task modes can help fully detect semantic lines that would otherwise be ignored, just as an old saying goes: "two heads are better than one". Our network, named Complementary semantic line TRansformer (CosineTR), applying a multi-scale detection method that estimates semantic lines both from heatmap prediction

* Correspondence to: 37 Xueyuan Road, Haidian District, Beijing, China.
E-mail address: tongchao@buaa.edu.cn (C. Tong).
[1] These authors contributed equally to this work.

https://doi.org/10.1016/j.patcog.2024.110952
Received 1 November 2023; Received in revised form 12 June 2024; Accepted 27 August 2024
Available online 30 August 2024

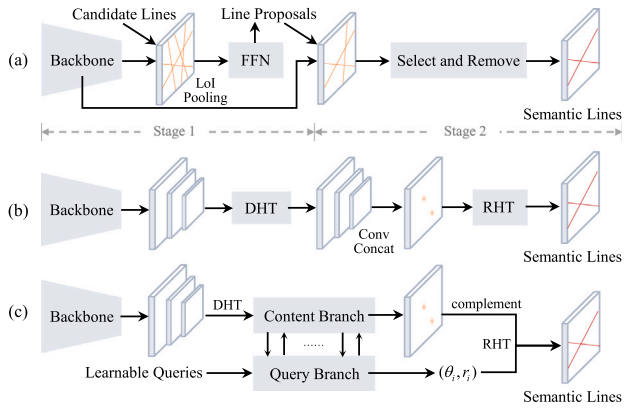


Fig. 1. Pipeline comparison between: (a) multi-stage detection model, (b) Hough transform-based model and (c) our proposed Complementary semantic line TRansformer (CosineTR). CosineTR applies two branches to capture local and global semantic information, encouraging to complementarily detect semantic lines from two perspectives.

and parameter regression. The two branches, called content branch and query branch, can effectively extract local semantic contexts and perform global interactions to achieve the overall optimal detection results. Meanwhile, considering the advantage of feature representation of Transformer [19] and its ability to capture long-term dependencies, global modeling and set prediction, we build the dual-branch decoder on top of the Transformer block [20,21], and design two attention modules to make it better focus on the region features around semantic lines.

Our contributions are summarized as follows.

- We consider the semantic line detection problem as a joint heatmap prediction and parameter regression task. A dual-branch Transformer-based network, CosineTR, is developed to undertake these processes and aim to obtain complementary detection results.
- The proposed detector decodes multi-scale features from coarse to fine, with special focus on local region context and global interaction of lines, helping to obtain the global optimal set of semantic lines.
- We introduce two attention modules, bilateral region attention and Gaussian prior cross-attention, to facilitate paying more attention to regions around semantic lines.
- We conduct experiments on multiple datasets, the results demonstrate that our method outperforms previous works and achieves state-of-the-art results.

2. Related work

2.1. Semantic line detection

Semantic lines are the general description of the spatial and scene layout of an image, with their endpoints all located on the boundaries of the image, and the two sides of the line representing different semantics. Since the main concept on semantic line detection [2] emerged, the recent surge of deep learning-based approaches has formed two types of mainstream approaches. One is the multi-stage detector similar to Faster R-CNN [22], which initiates candidate lines randomly and pools contextual features along the lines to generate line proposals, then filters and validates the proposals and regresses offsets for them to yield more precise coordinates [2,3,23]. The other kind of method relies on deep Hough transform [17,18] to parameterize lines in terms of two geometric terms: an offset and an angle, thus mapping the deep feature map from image domain to Hough domain [24]. By parallelizing operations and efficient heatmap prediction, this method shows a significant inference speed improvement. Nevertheless, the above approaches employing dense prediction are susceptible to sparse annotated examples during training phase, and they barely consider the interaction between a set of semantic lines. In this work, we present a dual-branch network to focus on local and global semantic contexts modeling, thus detect semantic lines more comprehensively.

2.2. Transformer in vision

Since Transformer [19] has made a splash in the field of natural language processing, its application to computer vision tasks is also growing rapidly [25–27]. These networks built with self-attention mechanism architecture have been shown to reach or even surpass the performance of CNNs on various dense prediction tasks [28–30]. They can produce global receptive fields, alleviate inductive bias and model long-range dependencies [31–33]. In addition to exploiting the representation capabilities of Transformer encoder to build a vision backbone, its decoder can also be developed as a detection pipeline to achieve end-to-end trainable object detection [34–36]. Instead of using hand-crafted anchor boxes, DETR employ a set of learnable query embeddings as object proxies, flexibly interacting with the region of interest features through cross-attention mechanism [37]. As detecting a harmonious set of semantic lines indispensably requires global perception, as well as interactions between each semantic line, we take advantage of Transformer to realize our dual-branch network, and introduce novel attention mechanisms to reduce the computational complexity and facilitate the training process.

3. Method

3.1. Overview

Semantic line detection aims to find an optimal set of lines that can delineate the semantic regions of an image [1]. Our proposed CosineTR (as illustrated in Fig. 2) is built to solve this problem based on Transformer block and can be adapted to any multi-scale feature extraction backbone network. More specifically, the processes of the proposed model consist of three stages:

(1) *Image Feature Extraction and Transformation*: Given an input image I, a backbone network is used to extract multi-scale feature maps $X = [X_1, X_2, X_3, X_4]$, where $X_i \in \mathbb{R}^{H_i \times W_i \times C_i}$. In general, we take features output at multiple encoding stages with $[\frac{1}{32}, \frac{1}{16}, \frac{1}{8}, \frac{1}{4}]$ original resolution. To take full advantage of line features and facilitate parallelization, we follow the previous work [18] to transform the representation from image domain to Hough domain, so that each pixel can represent a set of lines. More formally, we use two parameters (θ_l, r_l) to parameterize a straight line l, where $\theta_l \in [0, \pi)$ represents the angle between the plumb line of l and the x-axis, and $r_l \in [-\frac{\sqrt{W^2+H^2}}{2}, \frac{\sqrt{W^2+H^2}}{2}]$ denote the distance from l to the origin. For each point on the line l with coordinates (x_i, y_i), it should satisfy the parametric equation:

$$r_l = x_i \cos\theta_l + y_i \sin\theta_l, \tag{1}$$

so that the features of all pixels alone line l can be aggregated to the pixel $(\hat{\theta}_l, \hat{r}_l)$ in Hough space feature H:

$$H(\hat{\theta}_l, \hat{r}_l) = \frac{1}{|l|} \sum_{p \in l} X(p), \tag{2}$$

where p is the positional index of the pixel along l, $\hat{\theta}_l$ and $\hat{r}_l$ are calculated by:

$$\hat{\theta}_l = \lfloor \frac{\theta_l}{\Delta\theta} \rfloor, \hat{r}_l = \lfloor \frac{r_l}{\Delta r} \rfloor, \tag{3}$$

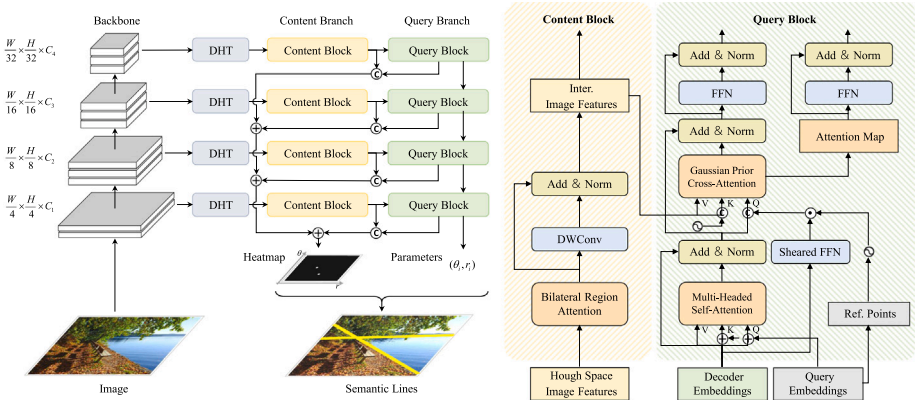


Fig. 2. Overview of the proposed Complementary semantic line TRansformer (CosineTR): the input image is fed into a backbone network to extract multi-scale feature maps, which are then converted from image space to Hough space by Deep Hough Transform (DHT) [18]. Next, the features are decoded from coarse to fine through content branch and query branch respectively, the intermediate features are fused with higher resolution deep representations to attach fine-grained information layer by layer. Finally, semantic threads are detected by the heatmap prediction header of the content branch and the parameter regression header of the query branch.

where $\Delta\theta$ and Δr are specified quantization intervals that control the size of the transformed feature map. We fix $\Delta\theta$ and adjust Δr for different encoding layers to finally obtain a series of transformed features $H = [H_1, H_2, H_3, H_4]$, where $H_i \in \mathbb{R}^{\Theta \times R_i \times C}$.

(2) *Coarse-to-Fine Decoding*: The proposed dual-branch decoder takes the transformed features H and a set of N line queries $L \in \mathbb{R}^{N \times C}$ as input to perform semantic line detection in a coarse-to-fine iterative manner. At the end of each decoding stage, we upsample the upper-level output features O_{i-1} to the same size as the current-level output features $\hat{H}_i$ and use element-wise add to fuse them:

$$O_i = \text{Upsample}(O_{i-1}) + \hat{H}_i, \tag{4}$$

and the initial output $O_1 = \hat{H}_1$. We specifically design bilateral region attention to replace the standard self-attention mechanism to focus on the essential property of semantic lines, i.e., the boundaries of semantic regions. Meanwhile, the need for global modeling is taken up by query branch, which utilizes learnable line query embeddings to focus on region of interest in intermediate features output by the content branch and enables information transfer between all semantic lines using self-attention mechanism. In addition, to better help line queries locate to their respective corresponding regions, we design Gaussian prior attention as a complement to cross-attention. Finally, at the end of each decoding stage, we concatenate feature map with attention map to complement the information generated by two branches. More details can be found in Section 3.2.

(3) *Semantic Line Prediction*: After a series of decoding stages, we obtain the final feature maps and decoder embeddings. Since the two types of features are represented differently, we assemble a heatmap prediction head to content branch and a parameter regression head to query branch to predict semantic lines separately. After both heads output semantic lines and confidence scores, the results are collected together by a simple post-processing process. Finally, we map the line parameters back into original space to represent a semantic line with a quadruple (x_1, y_1, x_2, y_2), where (x_1, y_1) and (x_2, y_2) are two intersections of the semantic line and image boundaries.

3.2. Dual-branch decoder

Different from previous works, we devise a dual-branch Transformer-based decoder to detect semantic lines. This design has the advantage of reducing the difficulty of training dense prediction tasks with sparse labels, while providing a simultaneous focus on local and global semantic modeling. As a whole, the decoder has two branches called semantic content branch and semantic query branch. These two branches consist of identical and independent content blocks and query blocks stacked multiple times, respectively. They receive feature maps of different scales as inputs, iteratively decode and make precise predictions.

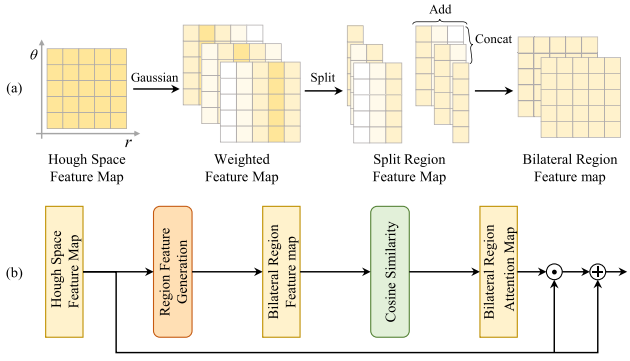


Fig. 3. Bilateral region attention. (a) Bilateral region feature generation process. It is the second step in bilateral region attention. (b) Process of bilateral region attention. Note that we omit channel dimension in the figure.

Bilateral Region Attention. The essence of a semantic line is the boundary of different semantic regions, which means the bilateral regions of the semantic line should exhibit heterogeneity. By exploiting this property, we propose the bilateral region attention as shown in Fig. 3. We implement bilateral region attention in Hough space to efficiently compute a whole attention map for all potential lines.

Given a transformed image feature map $H \in \mathbb{R}^{\Theta \times R \times C}$, our first objective is to obtain the feature representation of the adjacent regions for each line. Let us clarify again that each pixel in Hough space feature represents a set of neighboring lines, so we only need to be concerned with regions on either side of one pixel. Specifically, for each column $c \in [0, R)$, we treat it as an axis to generate weighted feature H_w^c by applying Gaussian weighting function $G(\cdot)$:

$$H_w^c(i) = H(i) \times G(d_i^c), \tag{5}$$

where i is the positional index of the pixel in the feature map and d_i^c denotes the distance of pixel i from the column c. Then, we split the feature map into left and right parts along the column c to obtain $H_l^c \in \mathbb{R}^{\Theta \times c \times D}$ and $H_r^c \in \mathbb{R}^{\Theta \times (R-c-1) \times D}$. After that, we sum the feature maps by rows and concatenate all H_l^c and H_r^c separately to obtain bilateral region feature map $H_b^l \in \mathbb{R}^{\Theta \times R \times D}$ and $H_b^r \in \mathbb{R}^{\Theta \times R \times D}$:

$$\hat{H}_l^c = \sum_{i=0}^{c-1} H_l^c(i), \tag{6}$$

$$\hat{H}_r^c = \sum_{i=c+1}^{R-1} H_r^c(i), \tag{7}$$

$$H_b^l = \text{Concat}(\hat{H}_l^0, \hat{H}_l^1, \ldots, \hat{H}_l^{R-1}), \tag{8}$$

$$H_b^r = \text{Concat}(\hat{H}_r^0, \hat{H}_r^1, \ldots, \hat{H}_r^{R-1}), \tag{9}$$

where i is the positional index of the pixel. Up to this point, for each pixel in original feature, the pixel at the same position in region feature maps contains the semantic contexts of regions on both sides of the corresponding axis line. Next, we calculate cosine similarity of H_b^l and H_b^r for each pixel to compare the semantic relevance of the left and right regions:

$$A(i) = \cos(H_b^l(i), H_b^r(i)) = \frac{H_b^l(i) \cdot H_b^r(p)}{\|H_b^l(i)\| \times \|H_b^r(i)\|}, \tag{10}$$

where $\|\cdot\|$ is the norm of vector and A is bilateral region attention map. Finally, the feature can be weighted as:

$$H_{att} = (1 + A) \odot H. \tag{11}$$

Gaussian Prior Cross-Attention. We randomly initialize a set of learnable line queries for the query branch as the basic reference of semantic lines with different locations. Once trained, these query embeddings are fixed to compute attention map with content block features and extract contexts from region of interest. Considering the original attention map is not always optimistic, we add a Gaussian prior to the cross-attention mechanism so that the region of interest of each line query will not excessively deviate from its corresponding position. Before introducing it, we first revisit the standard attention module proposed by Transformer [19]. Given a query $Q \in \mathbb{R}^{m \times d}$, a key $K \in \mathbb{R}^{n \times d}$ and a value $V \in \mathbb{R}^{n \times d}$, the attention operation is defined as:

$$\text{Attention}(Q, K, V) = \text{Softmax}(\frac{QK^T}{\sqrt{d}})V. \tag{12}$$

In the process of cross-attention, Q is mapped by decoder embeddings D_q through $W_Q \in \mathbb{R}^{d \times d}$, while K and V are mapped by the feature H_c through $W_K \in \mathbb{R}^{d \times d}$ and $W_V \in \mathbb{R}^{d \times d}$, where W_Q, W_K, and W_V are all learnable linear projection weights. More specifically, the cross-attention is calculated by $\text{Attention}(Q = D_q W_Q, K = H_c W_K, V = H_c W_V)$. As we want to introduce a spatial prior for each query, we first predict the offset $P_{offset} \in \mathbb{R}^{N \times 2}$ using a feed-forward network and query Q, and add it to the reference points. Note that the reference points $P_{ref} \in \mathbb{R}^{N \times 2}$ is predicted using the initial query L. To enhance the attention of line queries around their position, we generate the Gaussian-like weight map as:

$$G(i, j) = \exp(-\frac{(i - p_\theta)^2}{\sigma^2} - \frac{(j - p_r)^2}{\sigma^2}), \tag{13}$$

where $(i, j) \in [0, \Theta) \times [0, R)$ is the spatial indices of G, σ is the variance, (p_θ, p_r) is the coordinate of the prediction point. Thus, the gaussian prior attention can be written as:

$$\text{GPCA}(Q, K, V) = \text{Softmax}(\frac{QK^T}{\sqrt{d}} + \log G)V. \tag{14}$$

Content Block. We base the content block on standard Transformer block, with the difference being the use of bilateral region attention instead of multi-headed self-attention, so as to reduce computational complexity and focus on semantic line properties. In addition, deep convolution is added to the content block to be more adapted to processing visual modal features as well as more conducive to converging and fusing local features. Specifically, for the deep convolutional feedforward neural network, the input is the attention-weighted parameter space feature H_a. The computational procedure is shown below:

$$\begin{aligned} H_{f_1} &= \text{Conv}_{1\times1}\left(H_a\right) \\ H_{f_2} &= \text{DWConv}_{3\times3}\left(H_{f_1}\right) \\ H_{f_3} &= \text{GELU}\left(H_{f_2}\right) \\ H_{f_4} &= \text{Conv}_{1\times1}\left(H_{f_3}\right) \end{aligned} \tag{15}$$

where H_{f_i} denotes the intermediate features produced by each step, $\text{DWConv}_{3\times3}(\cdot)$ denotes the convolutional layer with a convolutional kernel size of 3×3, and $GELU(\cdot)$ denotes the Gaussian error linear cell activation function. The last part of the content block is consistent with the Transformer block and consists of jump connections and layer normalization, which is computed as shown in the following equation.

$$H_c = LN\left(H_{f_4} + H_a\right) \tag{16}$$

where $LN(\cdot)$ denotes layer normalization.

Query Block. The query block is designed based on the structure of DETR [21]. Meanwhile, in order to speed up the training to ensure its performance matching with the content block, we adopt the scheme proposed in Conditional DETR [34] to perform position correction of reference points in each decoding stage, and decoupling input features from positional encoding in cross-attention. More formally, the query block takes decoding embeddings $D_q \in \mathbb{R}^{N \times C}$ which output by the last query block and line query embeddings as input. First, the self-attention module makes all lines interact, avoiding multiple queries focusing on the same region. Then, line embeddings are computed with Hough space image features with Gaussian prior cross-attention to capture the line semantic features in different regions. At the same time, we keep the attention map $A_q \in \mathbb{R}^{\Theta \times R}$ generated in cross-attention process and send it to the next fusion process.

Attention Fusion. Since we do not want the two branches to perform the detection task completely independently, but rather to create some communication path between them so that the intermediate features can have complementary information, we design the attention fusion method. This step is very simple and only requires concatenating the decoded features H_c output from the content block with the attention map A_q output from the query block:

$$\hat{H} = \text{Concat}(H_c + A_q). \tag{17}$$

In this way, the feature maps output at each decoding stage contains the semantic information of both content and query block concerns, Integrating local and global semantic information.

3.3. Loss function

We use heatmap prediction head and parameter regression head to detect semantic lines simultaneously, and finally obtain the heatmap $P_h \in \mathbb{R}^{\Theta \times R}$ and a set of line parameters $P_l \in \mathbb{R}^{N \times 2}$. Based on this, we divide the loss into classification loss and regression loss.

Classification Loss. We consider heatmap prediction as a binary classification task for each pixel, so we compute the cross-entropy loss between ground-truth label $L_h \in \mathbb{R}^{\Theta \times R}$ and predicted map:

$$L_{ce} = -\sum_p (L_h(p) \log P_h(p) + (1 - L_h(p)) \log(1 - P_h(p))), \tag{18}$$

where p is index of the map. Besides, to overcome the problem of positive and negative sample imbalance, we also apply the dice loss:

$$L_{dice} = 1 - \frac{2|L_h \cap P_h|}{|L_h| + |P_h|}, \tag{19}$$

where $|\cdot|$ is the summation of matrix elements.

Regression Loss. Since we predict a pair of parameters and a confidence score for each line query, therefore the number of predicted results we get is greater than the number of ground-truth labels $L_l \in \mathbb{R}^{M \times 2}$. We calculate the cost function between the prediction results and the labels based on the coordinate distance and confidence, and select m lines with the smallest total cost as the positive predicted samples by bipartite matching algorithm. Then, we calculate focal loss for all predicted results:

$$\begin{aligned} L_{focal} = &- \mathbb{I}_{\{\sigma(i) \le m\}} \alpha (1 - P_l(i))^\gamma \log P_l(i) \\ &- \mathbb{I}_{\{\sigma(i) > m\}} (1 - \alpha) P_l(i)^\gamma \log(1 - P_l(i)), \end{aligned} \tag{20}$$

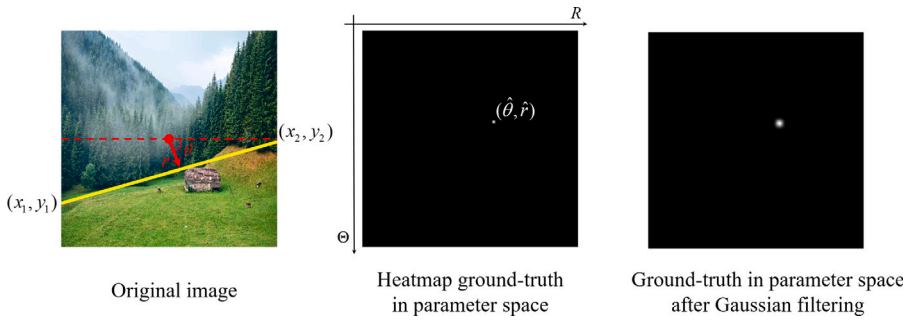


Fig. 4. Label generation process in parameter space.

where i denotes ith predicted results, $\sigma(i)$ is the index of ith line after matching, α and γ are both hyper-parameters. While for distance loss, we only consider the matched prediction lines:

$$L_{dist} = \mathbb{I}_{\{\sigma(i)\leq m\}} d(L_l(i), P_l(i)), \tag{21}$$

where $d(\cdot)$ is the sum of L1 distance between prediction and target coordinates. Finally, the total loss is represented as a weighted sum of the above losses:

$$L_{total} = \lambda_{cls} L_{cls} + \lambda_{dice} L_{dice} + \lambda_{focal} L_{focal} + \lambda_{dist} L_{dist}. \tag{22}$$

4. Experiments

4.1. Experimental settings

Implementation Details. We adopt ResNet50 [38] as our backbone, and resize input image size to 384×384. The transformed feature size has Θ equal to 96 and R equal to $[12, 24, 48, 96]$ for different decoding stages from top to down. The number of line queries N is set to 20, the feature dimension C is set to 256, and the sigma σ of prior Gaussian map is set to $\frac{1}{3}$. We use sine/cosine positional encoding in query block. The α and γ in focal loss are set to 0.25 and 2, loss weights $\lambda_{cls} = 0.3$, $\lambda_{dice} = 0.5$, $\lambda_{focal} = 1$ and $\lambda_{dist} = 10$. We train the network for 50 epochs with batch size 8, take AdamW as the model optimizer and set weight decay as 10^{-4}. The initial learning rate is set to 10^{-4} and decays according to $(1 - \frac{epoch}{total\_epoch})^{0.9}$ All experiments are conducted with a single NVIDIA RTX 3090 GPU.

Dataset. We conduct experiments on the SEL [2], SEL_Hard [3] and NKL [18] datasets. All of them contain natural images with a large number of scenes and a variable number of semantic lines are labeled for each image. Both SEL and NKL datasets provide training set and test set, while SEL-Hard is only used for testing.

Data preprocessing. In the sample label generation step, corresponding to the loss function setting case introduced in Section 3.3, we need to provide the heat map ground-truth and semantic line ground-truth in the parameter space, whose label generation process is shown in Fig. 4. Specifically, for each semantic line annotation (x_1, y_1, x_2, y_2) in the image, it is first converted into a pair of semantic line parameters (θ, r) using the line parameterized representation, which can be directly used as parameter space semantic line ground-truth. In addition, for heatmap ground-truth in parameter space, the semantic line parameters need to be further quantized according to the quantization interval $(\Delta\theta, \Delta r)$. By using the quantized line parameters $(\hat{\theta}, \hat{r})$ as an index, the pixel value of the position corresponding to this index in the heatmap label is set to 1, which means that the position corresponds to a semantic line positive sample.

Evaluation Metrics. To keep consistent with previous works, we adopt AUC [2] and HIoU [1] metric to evaluate our model. For calculating AUC, we first evaluate mIoU score between a predicted semantic line with the ground-truth label, a semantic line is regarded as correctly detected if its mIoU score with the ground-truth line is greater than a threshold τ, then we can compute the precision and the recall by:

$$\text{Precision} = \frac{N_{tp}}{N_{tp} + N_{fp}}, \text{Recall} = \frac{N_{tp}}{N_{tp} + N_{fn}}, \tag{23}$$

where N_{tp} is the number of correctly detected lines, N_{fp} is the number of wrongly detected lines and N_{fn} the number of undetected lines. Meanwhile, the F1-score can be calculated by:

$$\text{F1}-\text{score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{24}$$

By adjusting the threshold, a curve can be plotted and the AUC is the area under this curve.

In addition, HIoU measures the harmony of detected semantic lines in the context of an image. Suppose that the set of detected lines and the set of ground-truth lines divide the image into regions $S = \{s_1, s_2, \ldots, s_N\}$ and $T = \{t_1, t_2, \ldots, t_M\}$, the HIoU is defined as:

$$\text{HIoU} = \frac{\sum_{i=1}^{N} \max_k \text{IoU}(s_i, t_k) + \sum_{j=1}^{M} \max_k \text{IoU}(t_j, s_k)}{N + M}. \tag{25}$$

Table 1
Performance comparison on SEL and SEL_Hard datasets, all results are presented in percentages.

Method	SEL				SEL_Hard				FPS
	AUC_P	AUC_R	AUC_F	HIoU	AUC_P	AUC_R	AUC_F	HIoU	
SLNet [2]	84.56	82.01	83.27	78.13	79.53	67.28	72.89	59.83	22.13
DRM [3]	85.84	**86.99**	86.41	80.35	86.75	**77.76**	**82.01**	68.67	1.21
DHT [18]	87.79	80.24	83.85	79.77	84.72	72.47	78.12	64.42	**56.70**
SHNet [1]	89.60	83.91	86.66	80.92	87.34	72.35	79.14	65.89	27.13
CosineTR (ours)	**92.44**	84.35	**88.21**	**84.48**	**88.37**	73.87	80.47	**69.65**	16.20

4.2. Results and comparisons

We compare our model with previous methods on SEL and SEL_Hard datasets in Table 1, and the detection results is shown in Fig. 5. The results demonstrate that our method achieves a new state-of-the-art for HIoU metric on both dataset, which verifies that the interactive modeling of local and global semantic contexts by dual-branch allows for a more harmonious division of semantic regions for each image. Besides, our model also gains notable improvements on AUC metrics, more specific result curves are shown in Fig. 6. Notably, our model can improve recall as much as possible with high precision and is comparable to candidate line-based multi-stage detectors, which benefit from the use of dual branch architecture with two prediction heads. In contrast, the inference speed of our model is slightly lower than that of other methods, which is mainly caused by the cross-attention in the multi-stage decoding process, as well as the impact of the dual-branch architecture. Nevertheless, the speed of our method is still substantially faster than DRM, which gets a higher recall by repeatedly selecting proposal lines, while this consumes too much time.

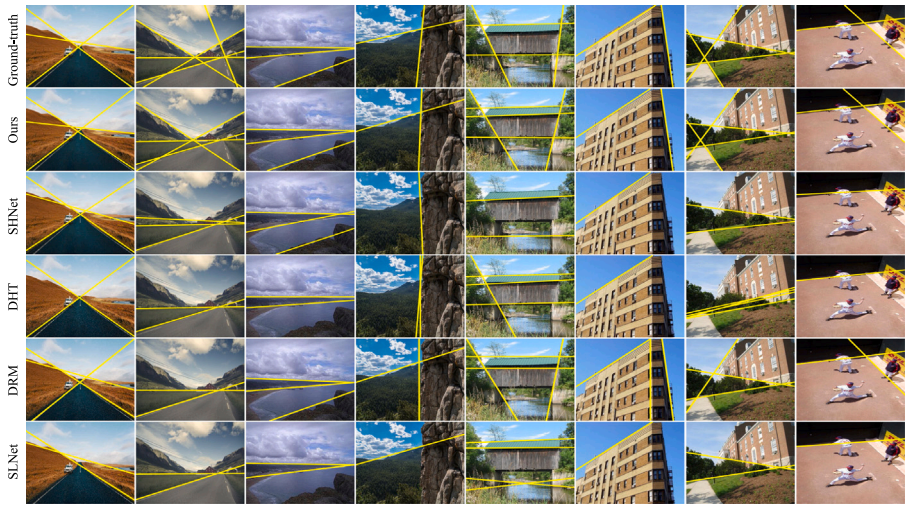


Fig. 5. Comparison of semantic line detection results on SEL and SEL_Hard datasets.

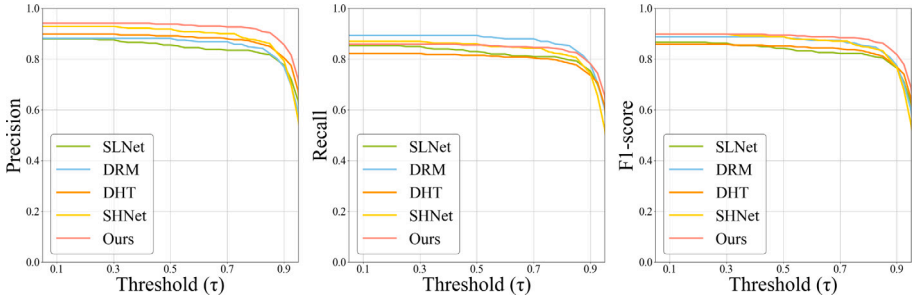


Fig. 6. Comparison of precision, recall and F1-score curves on the SEL dataset.

Table 2
Performance comparison on NKL datasets.

Method	AUC_P	AUC_R	AUC_F	HIoU
DHT [18]	75.74	76.24	75.99	70.09
SHNet [1]	81.60	81.64	81.62	74,36
CosineTR (ours)	**81.75**	**82.27**	**82.01**	**77.70**

Table 3
Ablation results of the proposed method on SEL dataset.

Method	AUC_P	AUC_R	AUC_F	HIoU
w/o BRA	91.26	84.10	87.53	83.13
w/o GPCA	91.81	83.07	87.22	83.89
w/o AF	92.02	84.29	87.99	83.90
CB only	94.48	81.60	87.51	84.45
QB only	**95.80**	80.96	87.76	84.29
ours full	92.44	**84.35**	**88.21**	**84.48**

On the NKL dataset, our model compares with the best-performing methods in each of the two types of schemes. The NKL dataset includes a wider range of complex scenarios and a larger number of semantic lines, making it more difficult to detect the optimal set of semantic lines. Table 2 shows our model also reaches state-of-the-art performance, in fact, our approach can be seen as an organic combination of these two approaches, which uses both heatmap prediction and parametric regression to perform complementary detection, thus achieving a more comprehensive detection of semantic lines.

4.3. Ablation studies

As shown in Table 3, we conduct ablation studies to explore the effects of (1) Bilateral Region Attention module, (2) Gaussian Prior Cross-Attention module, (3) Attention Fusion method, (4) the results obtained using only content branch and (5) the results obtained using only query branch. In addition, we also evaluated the impact of different decoding stages and the results of using different backbone networks.

Bilateral Region Attention. To explore the effect of applying bilateral region attention, we replace it with the spatial reduction attention (SRA) which proposed in PVT [25]. The SRA is very close to the standard attention mechanism, but in order to reduce the resource consumption in vision tasks, it reduces the length of image embedding tokens of key and value. Compared to the SRA that requires learning to focus on different regions, BRA generates heatmaps by evaluating the semantic similarity of bilateral regional features, which is more in line with the property and definition of the semantic line. Fig. 7 shows the heatmaps of bilateral region attention generated during the last decoding stage, we can clearly see that the BRA can effectively capture the boundaries of different semantic regions and reinforce the corresponding semantic contextual features, suggesting that there may be semantic lines here.

Gaussian Prior Cross-Attention. Since gaussian prior attention is a plug-and-play module attached to the cross-attention mechanism, we directly disable it in the experiments. It can be seen from Table 3 that the usage of GPCA can bring gains in detection performance, especially in the metric of AUC_R. This is due to the fact that the Gaussian prior heatmaps generated based on each line query help them to better focus on the region they are responsible for predicting. A more intuitive demonstration is shown in Fig. 8, the original cross-attention heatmaps are obtained by model learning, and although they

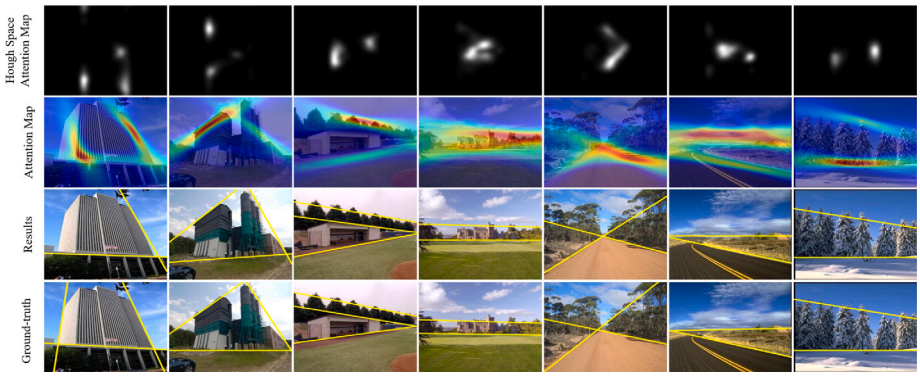


Fig. 7. Visualization of bilateral region attention heatmaps and Hough space attention maps. The heatmap is generated in Hough space and mapping to image space by inverse Hough transform.

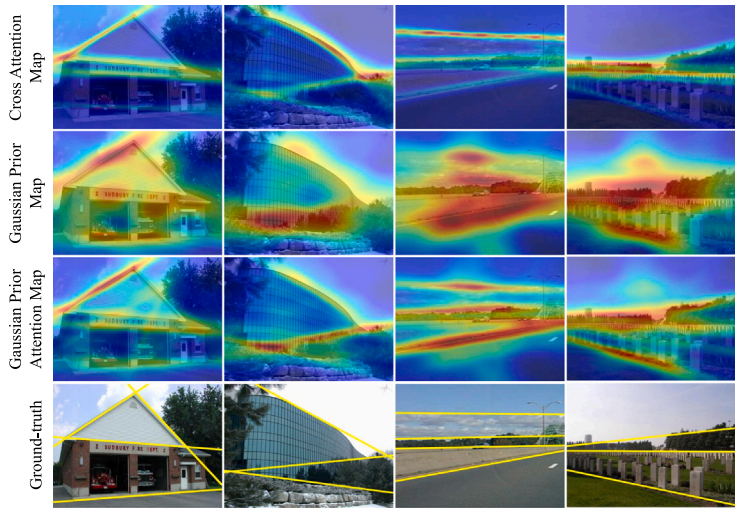


Fig. 8. Visualization of gaussian prior cross-attention heatmaps, the first row shows original cross-attention heatmaps, the second shows gaussian prior heatmaps generated according to line queries, and the third row shows gaussian prior cross-attention heatmaps.

have focused on some important semantic line locations, they still miss some key regions. By overlaying Gaussian prior heatmaps on them, it can be clearly observed that the regions of interest are closer to the location of the semantic lines in ground-truth labels, which helps the model to train faster and obtain higher recall score.

Attention Fusion. We evaluate the attention fusion method by uncoupling the feature map with the attention map after each decoding stage. We find that using AF results in a slight performance gain, which proves that establishing an communication path between two branches can effectively share semantic contexts and help both branches make better predictions.

Single Head Prediction. Since we equip the model with two independent prediction heads to detect semantic lines in the form of heat map prediction and parameter regression, respectively. We can observe the effect of complementary detection by keeping the detection results of the single prediction head, just as shown in Fig. 9. It should be noted that when combining the results of the these prediction heads, we first keep the heatmap prediction results and supplement them with the parametric prediction results, as the heatmap predictions tend to produce more accurate results. From the figure, we can find that both detection heads tend to detect the full set of semantic lines, but there may be omissions in some scenarios, but this can be complemented by the other. Combined with Table 3 we can see that although the AUC_P of the single head prediction results are higher than the full model, they have a major drawback in the AUC_R score, meaning that they have difficulty reaching comprehensive detection results alone. Conversely, when we use two prediction heads for complementary detection, although some false positive semantic lines are inevitably introduced into final results, leading to a slight decrease in AUC_P, we can capture those semantic lines that would otherwise be missed and eventually achieve the best performance in terms of the combined metric AUC_F, which confirms our conjecture that two heads are better than one.

Table 4
Detection results of adopting different backbones on SEL dataset.

Backbone	AUC_P	AUC_R	AUC_F	HIoU
VGG16 [39]	88.58	85.89	87.22	82.61
ResNet50 [38]	92.44	84.35	88.21	84.48
PVT [25]	92.30	85.48	88.76	84.89
MPViT [29]	**92.73**	**86.82**	**89.68**	**84.95**

Multi-stage Decoding. We use features gained after each decoding stage directly for prediction, and the obtained results are presented in Fig. 10. This intuitively demonstrates the coarse-to-fine decoding process of the CosineTR decoder. The first decoding stage focuses on as many semantic line proposals as possible and roughly predicts their locations. As subsequent decoding stages proceed and fine-grained semantic features are introduced, the model selects and excludes redundant lines and more finely tunes the specific locations of the predicted semantic lines, ultimately yielding accurate detection results.

Different Backbones. The proposed decoder can be adapted to any multi-scale encoding backbone network, so we tried some classical architectures, including CNN-based networks VGG16 and ResNet50,

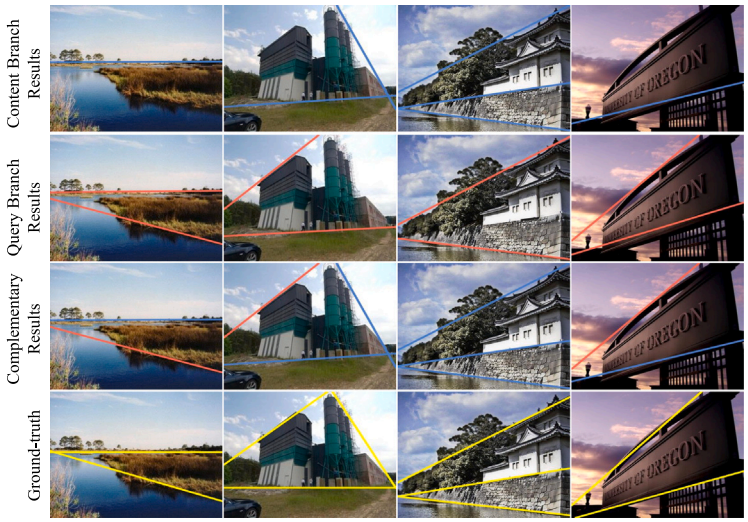


Fig. 9. Detection results of single prediction head and their complements.

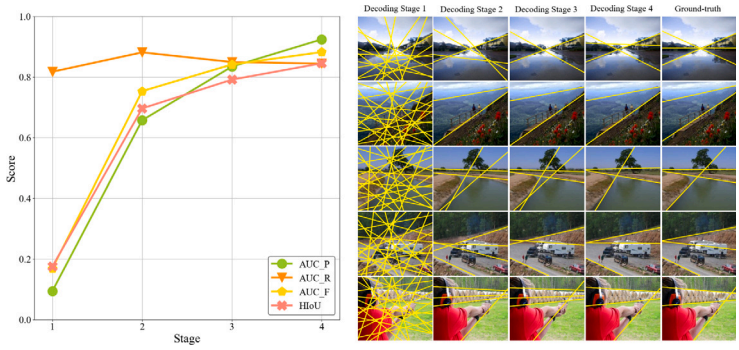


Fig. 10. Detection results of different decoding stages.

Transformer-based networks PVT, and a mixed architecture of both called MPViT, the results are presented in Table 4. We find that the Transformer-based architecture provides greater performance gains for the model due to its strong global modeling capabilities and its ability to capture long-term dependencies. The semantic line detection task relies strongly on global features and remote dependencies, so that the model needs to capture semantic contexts in a large range of regions as well as focus on the connections between different semantic regions.

4.4. Subjective semantic lines quality evaluations

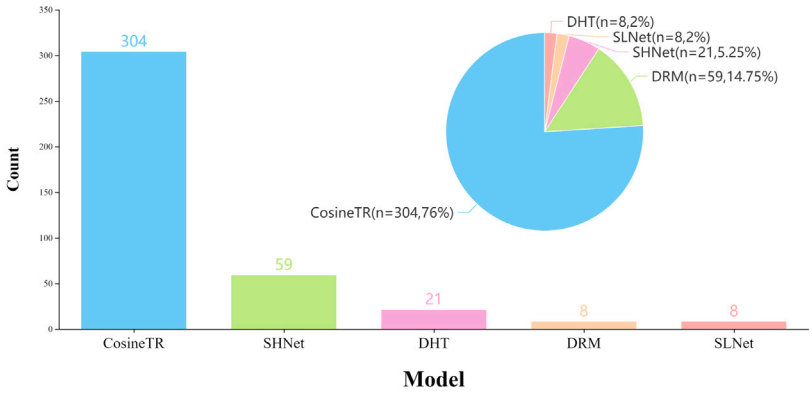
To further evaluate the effectiveness of our model, we conducted two rounds of subjective semantic lines quality evaluations. A total of 106 evaluations were conducted, with more than 10 instances involving participants with over five years of photography experience and over 30 instances involving researchers in the field of computer vision. All participants received brief training to understand the definition of semantic lines. The results of the two subjective semantic lines quality evaluations are shown in Fig. 11.

In the subjective semantic lines quality evaluation 1, 50 participants were involved. This evaluation included 48 images, encompassing ground truth images and semantic lines detection results from five models, including ours. The options for each question were in a fixed order and non-anonymous. Evaluators knew the names of the models, but we did not disclose any potential competitive relationships. As shown in Fig. 11(a), our model achieved significant results in this evaluation, demonstrating its superiority in semantic lines detection compared to other models.

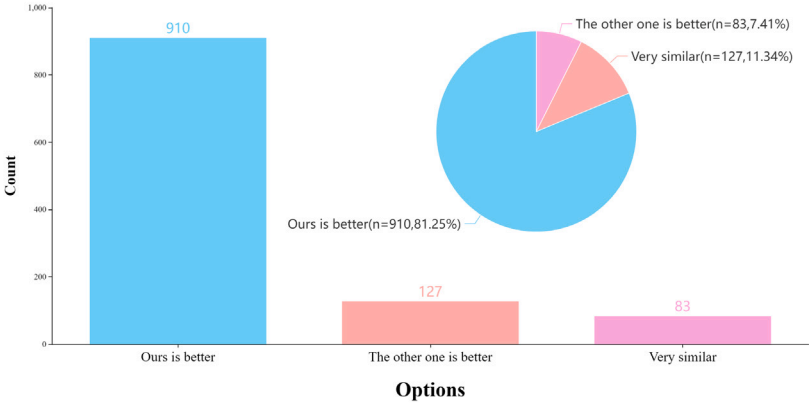
To eliminate any bias caused by the model names, we conducted the subjective semantic lines quality evaluation 2. In this evaluation, 56 participants were involved, and a total of 60 images were assessed. Each question included one ground truth image and options A and B, which contained randomly ordered semantic lines detection results from two models, and option C, "The above two are very similar". As shown in Fig. 11(b), even after removing the influence of model names, our model still outperformed the others, confirming its advantages. Examples of the two subjective semantic lines quality evaluations can be found in the Supplementary Information.

5. Conclusion

We propose the Complementary semantic line TRansformer (CosineTR), a dual-branch Transformer-based network that combines heatmap prediction and parameter regression tasks for semantic line detection. We design the attention fusion method to build communication between the two branches and present bilateral region attention and Gaussian prior cross-attention to assist the model in capturing semantic lines more effectively. Once trained, we can get complementary detection results from the two prediction heads. Experimental results demonstrate that our method achieves new state-of-the-art results. However, as shown by the visualized detection results, the two-branch structure still has the problem of duplicated attention or common omission of some semantic lines, which results in the waste of detection resources and insufficient enhancement of detection effect. Therefore, a more effective coupled detection scheme can be considered in further research by applying an artificial prior to the two branches.



(a)Subjective semantic lines quality evaluation 1.



(a)Subjective semantic lines quality evaluation 2.

Fig. 11. Results of subjective semantic line quality evaluation. In (a), the horizontal coordinates are the different models and the vertical coordinates are the count of evaluators' votes. In (b), the horizontal coordinates are the different options after collation (the pictures were randomized when the evaluators voted) and the vertical coordinates are the same as in (a).

CRediT authorship contribution statement

Yuqi Zhang: Writing – review & editing, Visualization, Methodology. **Bole Ma:** Writing – original draft, Methodology. **Luyang Jin:** Validation, Conceptualization. **Yuancheng Yang:** Investigation. **Chao Tong:** Supervision, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The authors do not have permission to share data.

Acknowledgments

This study is partially supported by National Natural Science Foundation of China (62176016, 72274127), National Key R&D Program of China (No. 2021YFB2104800), Guizhou Province Science and Technology Project: Research on Q&A Interactive Virtual Digital People for Intelligent Medical Treatment in Information Innovation Environment (supported by Qiankehe[2024] General 058), Capital Health Development Research Project (2022-2-2013), Haidian innovation and translation program from Peking University Third Hospital (HDCXZHKC2023203), and Project: Research on the Decision Support System for Urban and Park Carbon Emissions Empowered by Digital Technology - A Special Study on the Monitoring and Identification of Heavy Truck Beidou Carbon Emission Reductions.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.patcog.2024.110952.

References

[1] D. Jin, W. Park, S.-G. Jeong, C.-S. Kim, Harmonious semantic line detection via maximal weight clique selection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 16737–16745.
[2] J.-T. Lee, H.-U. Kim, C. Lee, C.-S. Kim, Semantic line detection and its applications, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 3229–3237.
[3] D. Jin, J.-T. Lee, C.-S. Kim, Semantic line detection using mirror attention and comparative ranking and matching, in: European Conference on Computer Vision, Springer, 2020, pp. 119–135.

[4] R. Doha, M. Al Hasan, S. Anwar, V. Rajendran, Deep learning based crop row detection with online domain adaptation, in: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 2773–2781.
[5] S.K. Panda, Y. Lee, M.K. Jawed, Agronav: Autonomous navigation framework for agricultural robots and vehicles using semantic segmentation and semantic line detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 6271–6280.
[6] H. Lou, H. Huang, C. Xiao, X. Jin, Aesthetic evaluation and guidance for mobile photography, in: Proceedings of the 29th ACM International Conference on Multimedia, 2021, pp. 2780–2782.
[7] F.O. Nwokoma, J.N. Odii, I.I. Ayogu, J.C. Ogbonna, Camera-based OCR scene text detection issues: A review, World J. Adv. Res. Rev. 12 (3) (2021) 484–489.
[8] C. Chan, F. Durand, P. Isola, Learning to generate line drawings that convey geometry and semantics, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 7915–7925.
[9] D. Xu, Y. Zhu, C.B. Choy, L. Fei-Fei, Scene graph generation by iterative message passing, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 5410–5419.
[10] R. Li, S. Zhang, X. He, Sgtr: End-to-end scene graph generation with transformer, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 19486–19496.
[11] J. Yang, Y.Z. Ang, Z. Guo, K. Zhou, W. Zhang, Z. Liu, Panoptic scene graph generation, in: European Conference on Computer Vision, Springer, 2022, pp. 178–196.
[12] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, Y. Bengio, Show, attend and tell: Neural image caption generation with visual attention, in: International Conference on Machine Learning, PMLR, 2015, pp. 2048–2057.
[13] V.-Q. Nguyen, M. Suganuma, T. Okatani, Grit: Faster and better image captioning transformer using dual visual features, in: European Conference on Computer Vision, Springer, 2022, pp. 167–184.
[14] Y. Ma, J. Ji, X. Sun, Y. Zhou, R. Ji, Towards local visual modeling for image captioning, Pattern Recognit. 138 (2023) 109420.
[15] C. Tong, X. Yin, J. Li, T. Zhu, R. Lv, L. Sun, J.J. Rodrigues, A shilling attack detector based on convolutional neural network for collaborative recommender system in social aware network, Comput. J. 61 (7) (2018) 949–958.
[16] C. Tong, J. Li, C. Lang, F. Kong, J. Niu, J.J. Rodrigues, An efficient deep model for day-ahead electricity load forecasting with stacked denoising auto-encoders, J. Parallel Distrib. Comput. 117 (2018) 267–273.
[17] Y. Lin, S.L. Pintea, J.C.v. Gemert, Deep hough-transform line priors, in: European Conference on Computer Vision, Springer, 2020, pp. 323–340.
[18] K. Zhao, Q. Han, C.-B. Zhang, J. Xu, M.-M. Cheng, Deep hough transform for semantic line detection, IEEE Trans. Pattern Anal. Mach. Intell. (2021).
[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: Advances in Neural Information Processing Systems, vol. 30, 2017.
[20] 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, in: International Conference on Learning Representations, 2020.
[21] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, S. Zagoruyko, End-to-end object detection with transformers, in: European Conference on Computer Vision, Springer, 2020, pp. 213–229.
[22] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: Towards real-time object detection with region proposal networks, in: Advances in Neural Information Processing Systems, vol. 28, 2015.
[23] D. Jin, C.-S. Kim, Score prediction network and graph-based selection for semantic line detection, in: 2020 International Conference on Information and Communication Technology Convergence, ICTC, IEEE, 2020, pp. 391–393.
[24] P.V. Hough, Method and means for recognizing complex patterns, 1962, US Patent 3 (6).
[25] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, 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.
[26] J. Dai, H. Li, R. Zeng, J. Bai, F. Zhou, J. Pan, KD-Former: Kinematic and dynamic coupled transformer network for 3D human motion prediction, Pattern Recognit. 143 (2023) 109806.
[27] Y. Li, J. Wang, X. Dai, L. Wang, C.-C.M. Yeh, Y. Zheng, W. Zhang, K.-L. Ma, How does attention work in vision transformers? A visual analytics attempt, IEEE Trans. Vis. Comput. Graphics (2023).
[28] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, 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.
[29] Y. Lee, J. Kim, J. Willette, S.J. Hwang, Mpvit: Multi-path vision transformer for dense prediction, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 7287–7296.
[30] T. Li, X. Dong, J. Lin, Y. Peng, A transformer-CNN parallel network for image guided depth completion, Pattern Recognit. (2024) 110305.
[31] Y. Li, C.-Y. Wu, H. Fan, K. Mangalam, B. Xiong, J. Malik, C. Feichtenhofer, Mvitv2: Improved multiscale vision transformers for classification and detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 4804–4814.
[32] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J.M. Alvarez, P. Luo, SegFormer: Simple and efficient design for semantic segmentation with transformers, Adv. Neural Inf. Process. Syst. 34 (2021).
[33] L. Yu, W. Xiang, J. Fang, Y.-P.P. Chen, L. Chi, EX-ViT: A novel explainable vision transformer for weakly supervised semantic segmentation, Pattern Recognit. 142 (2023) 109666.
[34] D. Meng, X. Chen, Z. Fan, G. Zeng, H. Li, Y. Yuan, L. Sun, J. Wang, Conditional detr for fast training convergence, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 3651–3660.
[35] Y. Li, H. Mao, R. Girshick, K. He, Exploring plain vision transformer backbones for object detection, in: European Conference on Computer Vision, Springer, 2022, pp. 280–296.
[36] X. Xie, D. Wu, M. Xie, Z. Li, GhostFormer: Efficiently amalgamated CNN-transformer architecture for object detection, Pattern Recognit. 148 (2024) 110172.
[37] J. Wang, W. Lin, C. Ma, M. Li, Z. Sun, L. Sun, Q. Huo, Robust table structure recognition with dynamic queries enhanced detection transformer, Pattern Recognit. 144 (2023) 109817.
[38] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[39] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, 2014, arXiv preprint arXiv:1409.1556.

Yuqi Zhang, Ph.D. student, School of Computer Science and Engineering, Beihang University, Beijing, China.

Bole Ma, Master of Science student, School of Computer Science and Engineering, Beihang University, Beijing, China.

Luyang Jin, Master of Science student, School of Computer Science and Engineering, Beihang University, Beijing, China.

Yuancheng Yang, Ph.D. student, School of Computer Science and Engineering, Beihang University, Beijing, China.

Dr. Chao Tong received his Ph.D. degrees in 2009 in computer science from Beihang University (BUAA). He is an associate professor in the School of Computer Science and Engineering, BUAA. He was also a visiting professor in the School of Computer Science, McGill University. He has published more than 100 referred papers and filed more than 30 patents. His current research interests include machine learning, mobile computing and social networks analysis.