Research on Panoptic Segmentation Algorithm Based on Transformer

Abstract

Panoptic segmentation is a kind of image segmentation task that integrates image semantic segmentation and instance segmentation, which not only focuses on the stuff of the background such as sky, road, etc., but also focuses on the things of different individuals such as people, car, etc. In order to learn more comprehensive image features, the FPN-style mask head in Mask R-CNN is added on top of the End-to-end Detection Transformer. more comprehensive image features, a new FPN-style mask head in Mask R-CNN is added on top of End-to-end Detection Transformer, which converts the output features of Transformer Decoder into instance segmentation map. Finally, the PQ in COCO 2017 Validation is 42.8 for PQ of stuff and 35.8 for PQ of things, which is in the high level compared with the same period.

I. Introduction

With the development of computer hardware and Internet technology, not only are computers able to handle more complex tasks, but also major manufacturers and companies are able to collect more types and amounts of data, sometimes text, sometimes speech, and sometimes images. When more of this data is available, the need for more complex computer tasks gradually arises, and this is the case with image recognition. Early image recognition was to read the image as a matrix, where each pixel point was treated as a feature, so the number of pixel points was the number of features. Limited by the hardware of the device, early image recognition can only accomplish the task of classifying image features on pictures with particularly small pixels, and most algorithms are based on traditional statistics and probability theory reasoning, without substantial breakthroughs.

With the emergence of CNNs such as LetNet-5 [1], AlexNet [2], VGG [3], and ResNet [4] at the beginning of the 21st century, the field of image recognition is no longer limited to categorizing images, and more complex tasks such as object detection and image segmentation have been proposed. Earlier in image segmentation task is to classify images at pixel level, J. Long, et al. [5] proposed Full Convolutional Neural Network FCN in 2015, which is the first deep learning based semantic segmentation model. The model can be divided into the structure of encoder-decoder, the encoder, i.e., the structure of CNN is used to extract the feature map; the decoder consists of multiple up-sampling layers, which are up-sampled by the feature map to the original image size of the segmentation map, so the subsequently proposed U-Net [6], V-Net [7] and other deep learning models for semantic segmentation are designed based on the encoder-decoder structure. With the development of image semantic segmentation, scholars believe that the sky, roads and pedestrians, vehicles can’t be confused, B. Hariharan, et al. [8] that there should exist a method to be able to segment the target object Things for pedestrians, vehicles, that is, instance segmentation in the middle of the development of image segmentation. Instance segmentation is different from semantic segmentation, which is to form a segmentation map of target objects on the detected target, which can be understood as a two-stage task of semantic segmentation of objects of object frames after generating object prediction frames from object detection, so scholars add FCN decoder similar structure on the object detection model in order to generate target segmentation map. Until 2017 K. He, et al. [9] proposed Mask R-CNN more flexible and simple two-stage instance segmentation model based on R-CNN framework. The model achieves better experimental results on MS COCO [10] dataset, adopts ResNet-FPN structure, i.e., multi-layer feature map fusion in FPN, whose multi-scale feature maps are conducive to multi-scale objects as well as small objects detection, and finally obtains the instance segmentation maps by bilinear interpolation up sampling, which has good segmentation quality for object targets. With the development of single-stage algorithms for object detection, single-stage instance segmentation is also emerging, as represented by YOLCAT [11], which adds the mask generation method to the single-stage object detection model. On the basis of instance segmentation and semantic segmentation, the panoptic segmentation tasks of Unifing Things and Stuff were derived, which added pixel-level segmentation of the backscene on the basis of instance segmentation. However, the distinction between foreground and rear view will lead to poor definition of evaluation metric, so A. Kirillov, et al. [12] proposed the evaluation metric for panoptic segmentation: Panoptic Quality (PQ), which is a unification of Things and Stuff's evaluation metric for panoptic segmentation, and it can evaluate the model's performance in panoptic segmentation in a more effective and comprehensive way.

The development in the field of computer vision with the victory of convolutional neural network in the ImageNet competition in 2012 has accelerated the development of various fields based on convolutional neural network, at the same time the field of computer vision has brought about deep learning, which has led to the rapid development of natural language processing. In 2017 Vaswani A. et al [13] published in the field of Natural Language Processing Transformer model, which introduces an encoder-decoder structure with an attention mechanism for sequence-to-sequence modeling of machine translation tasks. Compared with previous sequence-to-sequence models, Transformer accomplishes sentence prediction with varying lengths, reflecting the model's strong generalization ability, which is derived to the greatest extent from the ability of the attention mechanism to perceive global features. In view of Transformer's excellent global perception ability, A. Dosovitskiy, et al. [14] first proposed to use the Transformer encoder in an image classification task by flattening each piece of a chunked image and treating it as a single Token of the input vector, and finally adding a multilayer perceptron to realize the classification task, and the subsequent experiments confirmed that such a The subsequent experiments confirmed that this kind of architecture abandons convolution, which makes the model accuracy much higher than the traditional convolutional neural network, so Transformer formally enters the vision of scholars in the field of computer vision. 2020 N. Carion, et al. [15] proposed the end-to-end object detection model DETR based on Transformer, which eliminates the tedious Archer Box pre-setup in the architectures such as R-CNN, YOLCAT, and so on. The cumbersome Archer Box pre-setting and Non-Maximum suppression post-processing are eliminated, and the end-to-end object detection depth model is truly realized by objects queries. Although its performance in object detection is not as good as other architectures in the same period, this DETR architecture without cumbersome pre/post-processing became the pioneering work of this model, which has been widely used by subsequent scholars, such as the segmentation architectures SETR proposed in recent years [16], MaskFormer [17], are end-to-end architectures based on Transformer, which is based on convolutional neural network local receptive field and Transformer global features, which can understand the image features more comprehensively and thus accomplish more complex visual tasks.

In this paper, it will introduce a Mask R-CNN style Mask head based on the framework of DETR to process the output of Transformer Decoder for panoptic segmentation task, and perform model replication and algorithm profiling at COCO 2017 Panoptic Validation. Due to the large number of parameters in the Transformer architecture itself that makes it difficult to train, the pre-trained model that has been completed by ResNet in the ImageNet competition is taken as a base on Backbone to reduce the training time required for the convergence of model parameters.

II. Methods

This chapter will introduce the underlying DETR architecture and how to introduce the Mask Head of the Mask R-CNN into the structure, and dissect the loss function and evaluation metrics required for training.

A. *DETR architecture*

DETR is mainly composed of three parts, Backbone, Transformer, and the output FFN of shared parameters, and the original structure is shown in Fig. 1.

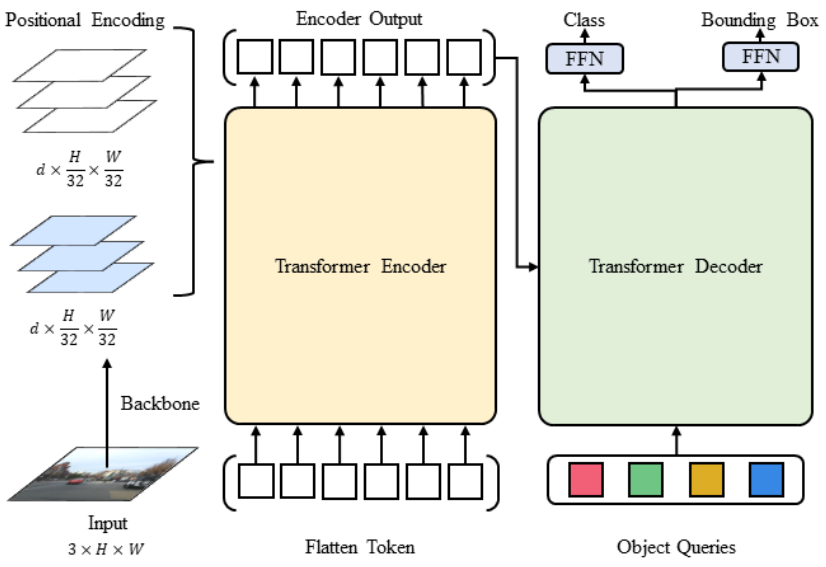

Figure 1. DETR original structure.

(1) *Backbone*

This part is to use CNN to extract features from the original image, remove the global average pooling, fully connected layer of the CNN final output classifier, and perform deep convolution operation on the input three-channel color image, i.e., the three-channel image features are mapped to the features of more number of channels to get the multi-channel feature map. Assuming that the width H and height W pixels of the input image are 224, a more representative ResNet-50 forward is shown in Fig. 2.

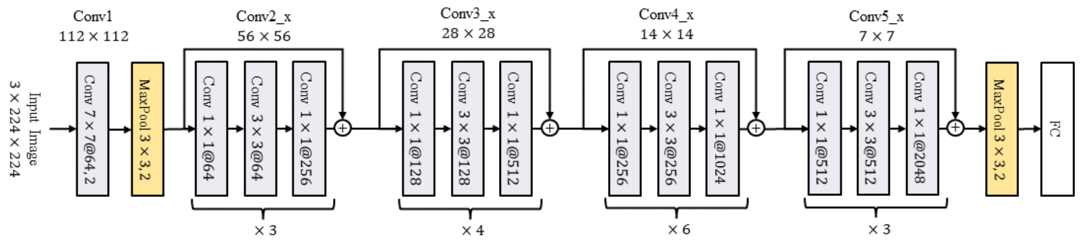

Figure 2. ResNet-50 forward calculation.

Backbone is the structure that removes the last yellow square, leaving only 2D convolutional, pooling layers. When initializing the kernel weight parameters of each convolutional layer, in order to reduce the convergence time required for model training, the parameters are initialized in the form of a normal distribution. In the selection of Backbone, considering the number of parameters of the overall model, it should choose the structure with as few convolutional layers as possible, with the best performance as possible, and with the subsequent Transformer structure, so chosing the ResNet series of networks as Backbone, which takes more original features into consideration. In addition to the convolutional structure, each layer also incorporates the residual connection, which accumulates the outputs of the previous module with the same size to the next module, and then adds the outputs of the next module with the same size to the next module. output of one module is accumulated to the next output feature map, so that the model can learn some of the features before the convolution operation.

(2) Positional Encoding

CNN has excellent image feature extraction, but the extracted multi-channel feature maps do not have good relative position information, i.e., the features do not have informative features between them, while the input to the Transformer model is word vectors with contextual relationships, so it is necessary to add position information i.e., positional coding to the multi-channel features. Based on the idea of bottleneck layer in residual network, the position information is accumulated to the feature vector and the feature vector has position information. There are two methods of position encoding in Transformer: the first is based on the idea of word index vector Embedding, which gives a learnable weight matrix to the input vectors, and the position encoding is obtained by learning while training the model parameters; the second method is based on the position encoding vectors generated by empirical formulas.

Mask is required before generating the positional coding, as each image in the dataset has a different size, the training process is a batch training approach, the Transformer's features can be input with different lengths of the features, but it is necessary to iterate all the images in the current batch filled with blank pixel points with a value of 0 to the maximum width and height in the current batch of images in order to form a uniform size. Because of the filled blank pixels, the training process needs to add a set of feature Masks in the input process in order to prevent the blank pixels from interfering with the learning of the Transformer parameters, and to perform Masking operation for the subsequent Multi-head Attention (MHA). When the batch size is 2, the feature Masks are generated as in Fig. 3.

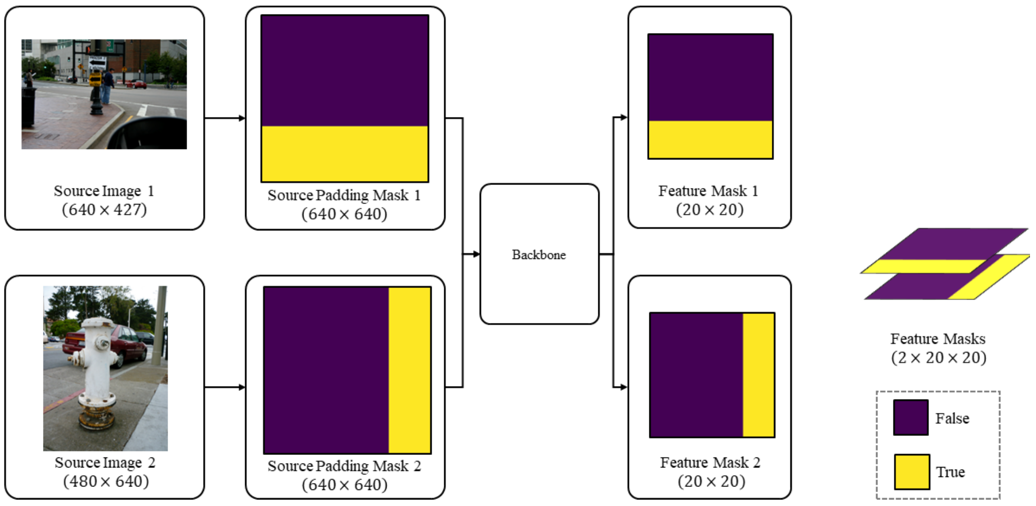

Figure 3. Generate batch feature mask.

Where the position where the Mask element is False indicates that the position feature does not need to be masked, on the contrary when the element is True an infinitesimal number is accrued to the corresponding position of the feature vector in the MHA's Masking so that the feature map in the subsequent linear mapping of the multiple attention the corresponding value is also infinitesimal so that the attention weight is close to 0 at the corresponding position.

To generate the Positional Encoding, a sine-cosine function will be taken. The binarized vector Not-Mask is generated according to the Mask with its opposite, i.e., the elements of the original position are changed from False to True, and True to False, and the Not-Mask is copied d times in order to initialize the Positional Encoding Vector $PE \in R^{d\times\frac{H}{32}\times\frac{W}{32}}$ at the same size as the feature map. Subsequent Cumulative sum for $\frac{d}{2}\times\frac{H}{32}\times\frac{W}{32}$ cumulative and horizontal coordinate directions, and the remaining cumulative and vertical coordinates, will result in two sets of vectors of relative positions as in Fig. 4.

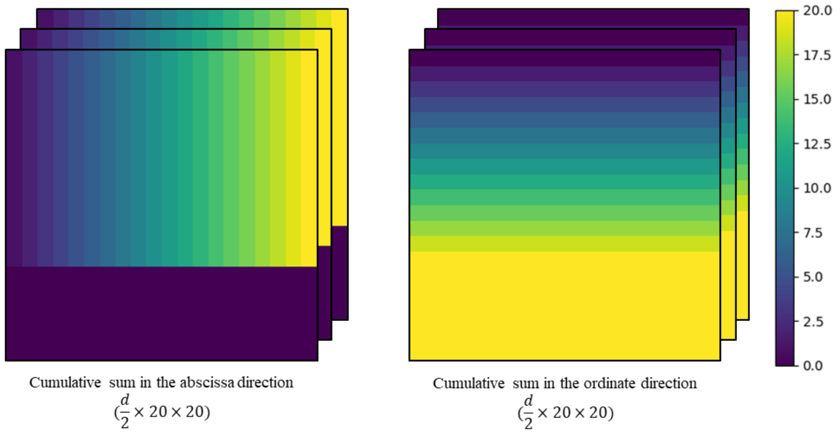

Figure 4. Two parts relative positions vector.

The embedding dimension d is equally divided into two parts for cumulative sums in different directions to achieve simultaneous consideration of the relationship between the horizontal and vertical coordinates in the image plane. Subsequently, the elements of PE are nonlinearly transformed using the sine-cosine function to generate a positional code with relative positional information. For the position pos, the value $PE(pos,i)$ of the dimension index i is as in equation (1).

$$PE(pos,i)=\begin{cases}\sin\left(\frac{pos}{10000^{\frac{2k}{d}}}\right), & i=2k,\\ \cos\left(\frac{pos}{10000^{\frac{2k}{d}}}\right), & i=2k+1\end{cases} \tag{1}$$

(3) Transformer

The Transformer structure in DETR is similar to the original structure in that the input vectors are adjusted to be 3D features in the shape of $d\times\frac{H}{32}\times\frac{W}{32}$, whereas the input vectors of the original structure are in the shape of 2D, so $\frac{H}{32}\times\frac{W}{32}$ is flattened, i.e., the 3D features are transformed into Flatten Token in the shape of $d\times\left(\frac{H}{32}\times\frac{W}{32}\right)$.Before entering the Transformer, an additional convolution layer of 1×1 kernel size is required to achieve a certain degree of dimensionality reduction of the multichannel features. Transformer needs to go through an additional convolutional layer of 1×1 kernel size to achieve a certain degree of dimensionality reduction of the multi-channel features. The structure of Transformer Encoder and Decoder in DETR is shown in Fig. 5.

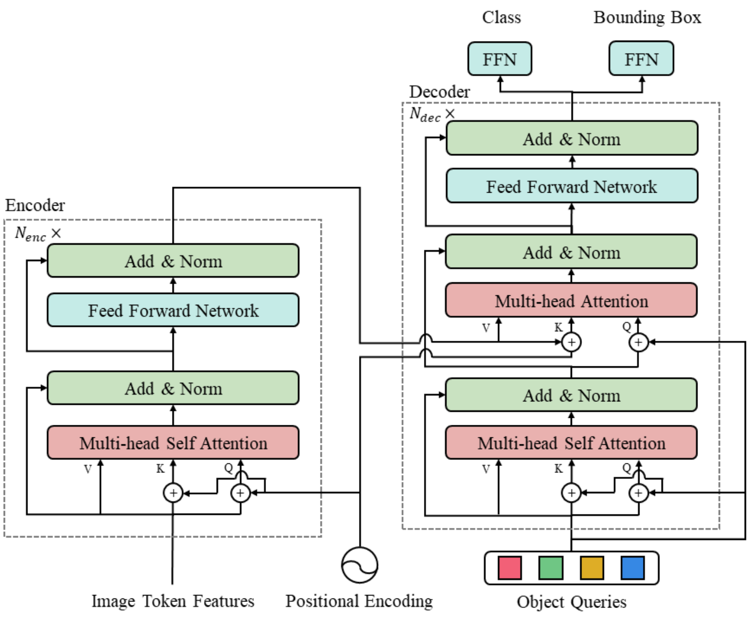

Figure 5. Detailed structure of Transformer in DETR.

The Encoder and Decoder of Transformer are composed of N_{enc} Encoder blocks and N_{dec} decoder blocks respectively, and each encoder block and decoder block have similar structure with MHA, residual add, Layer Normalization (LN), Feed Forward Network (FFN), the decoder block has one more MHA than the encoder block to accept the output of the encoder, and Multi-head Self Attention (MHSA) is essentially the same as the MHA. Multi-head Self Attention (MHSA) is essentially the same as MHA, except that MHSA focuses more on "self-attention", and the detailed structure of MHA is shown in Fig. 6.

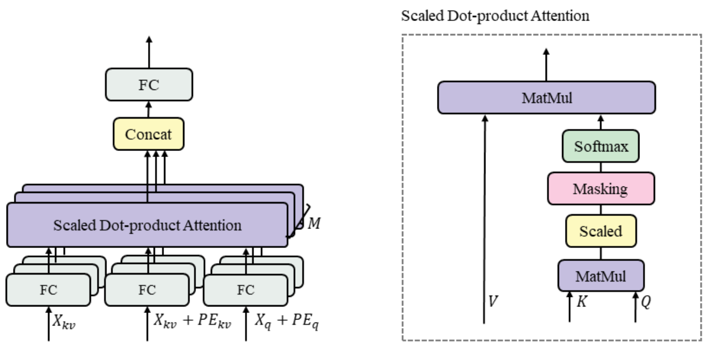

Figure 6. MHA and Scaled Dot-product Attention.

MHA is essentially a soft query, which is a linear mapping of input vectors through Fully Connected (FC) to get three different matrices Q, K, and V. In the model, K and V are from the same input vector, and Q is mapped from the same input vector as MHSA. In MHA, the input vectors with larger shapes are split into M heads for concatenation computation, and then finally spliced together after a linear mapping to get the same shape as the input vectors. The input vector with the same shape as the input vector is obtained by a linear mapping. In the attention of a single head, the Scaled Dot-product Attention operation is adopted, which is calculated according to $Q \cdot K^T$ to obtain a square matrix, and after scaled reduction, Masking, Softmax, it becomes a weight matrix with element values within $[0,1]$, i.e., the attention weights. The new vector obtained after multiplying the attention weights with V is the feature vector processed according to the attention weights generated by Q. Define the input vector as X. The whole MHA formula is shown in Eqs. (2-3).

$$\mathrm{MHA}\left(X_q, X_{kv}, T, L\right) = L \cdot \left[\mathrm{Attention}\left(X_q, X_{kv}, T_1\right); \cdots; \mathrm{Attention}\left(X_q, X_{kv}, T_M\right)\right] \tag{2}$$

$$\mathrm{Attention}\left(\mathrm{X_q}, X_{kv}, T_m\right) = T_{m1} X_{kv} \cdot \mathrm{Softmax}\left(\frac{\left(T_{m2}\left(X_{kv} + PE\right)\right)^T \left(T_{m1}\left(X_q + PE\right)\right)}{\sqrt{d}}\right) \tag{3}$$

Where, $T_m = \{T_{m1}, T_{m2}, T_{m3}\}, m = 1,2,\dots,M$ denotes the three FC layer weight matrices before the Scaled Dot-product Attention in the m-th header. L denotes the weight of the FC layer after the connection. This leads to $\mathrm{MHSA}(X, T, L) = \mathrm{MHA}(X, X, T, L)$. After MHA computation, the features are subjected to residual accumulation and layer normalization to enhance the original features to prevent too much loss in the computation process. In a single encoder or decoder block, an FFN consisting of FC-ReLU-Dropout-FC is included to propagate the MHA computed features forward after two successive linear mappings, in which the two FC layers have the weight dimension of d_{ff}, i.e., the feature vector with the input dimension of d is mapped into the d_{ff} dimension space through the first FC layer. ff dimension space, and mapped back to the d dimension space with the same shape as the input vector through the second FC layer.

In the original Transformer model used for Machine Translation, the input to the decoder starts from a 1×1 vector of start symbol token, predicts the next word and adds it to the current vector to get a 2×1 vector of tokens, which is then fed into the decoder, and so on until the next word predicted is the end symbol token position. But for DETR, only class and bounding box need to be predicted on object detection, so it is sufficient to pass the FFN with shared parameters after decoder. In the input part of the decoder, the original author is defining the object queries, which are essentially the weights of the Embedding layer, to complete the cumbersome anchor box preprocessing in the previous object detection in a learnable way, thus reducing the dependence on empirical people and realizing the end-to-end simplicity, the only drawback is the increase of the The only drawback is that it increases the convergence time of the model parameters during the training process.

B. *Mask head*

DETR is an architecture for object detection, using this framework as a basis to accomplish panoptic segmentation, as Mask R-CNN adds Mask head to the shared parameter FFN part to generate the segmentation map, and the mask head structure is shown in Fig. 7.

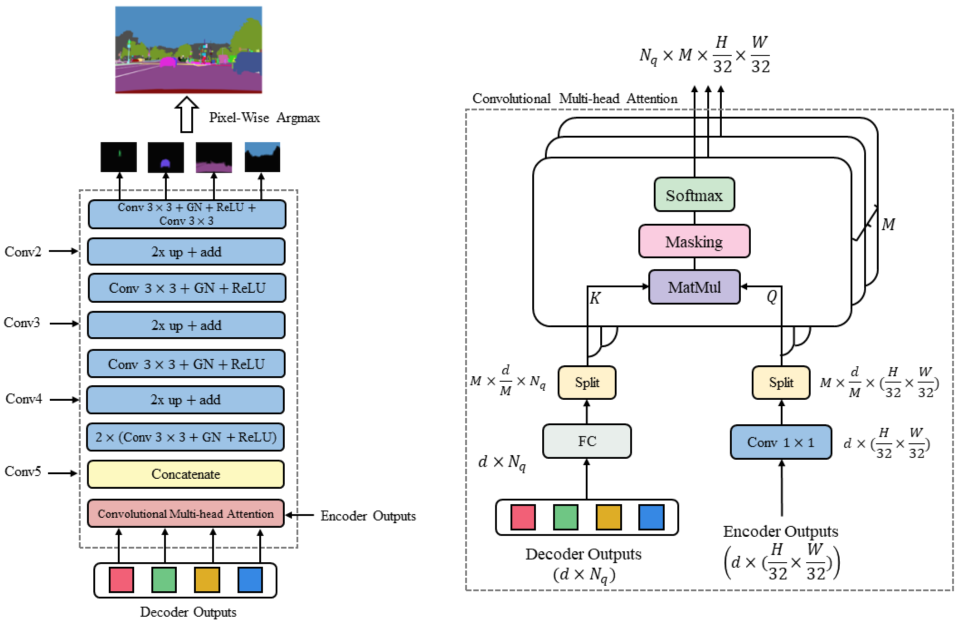
Pixel-Wise Argmax
Conv 3 × 3 + GN + ReLU + Conv 3 × 3
Conv2
2x up + add
Conv 3 × 3 + GN + ReLU
Conv3
2x up + add
Conv 3 × 3 + GN + ReLU
Conv4
2x up + add
2 × (Conv 3 × 3 + GN + ReLU)
Conv5
Concatenate
Convolutional Multi-head Attention
Encoder Outputs
Decoder Outputs
$N_q \times M \times \frac{H}{32} \times \frac{W}{32}$
Convolutional Multi-head Attention
Softmax
Masking
MatMul
K
Q
M
$M \times \frac{d}{M} \times N_q$
Split
Split
$M \times \frac{d}{M} \times (\frac{H}{32} \times \frac{W}{32})$
FC
Conv 1 × 1
$d \times N_q$
$d \times (\frac{H}{32} \times \frac{W}{32})$
Decoder Outputs
$(d \times N_q)$
Encoder Outputs
$\left(d \times (\frac{H}{32} \times \frac{W}{32})\right)$

Figure 7. Mask head structure.

Convolutional Mult-head Attention is a method of processing encoder and decoder outputs defined by MHA, which uses its soft query property to convert the features of object queries and encoder outputs into "attention" features of objects. "Attention" features, in general, is to treat stuff and things as the same type of objects, and generate feature maps for $N = N_q$ objects. Masking is similar to MHA's Masking, which also accumulates an infinitesimal number of blank pixels to achieve the purpose of ignoring these blank pixels. When the model parameters converge, these "attention" features can well represent the position and volume of the object in the image, so the subsequent FPN-Style CNN is adopted, according to the number of channels of each layer of Backbone's feature map, the "attention" feature map is convolved with the "attention" feature map, and the "attention" feature map is convolved with the number of channels of each layer of Backbone's feature map. According to the number of channels in each layer of Backbone's feature map, the convolution operation is performed on the "attention" feature map and the corresponding layer feature map is summed up to learn the details in the original image. After the convolution operation followed by a 2-fold Nearest Interpolation up-sampling method, gradually convert the $\frac{H}{32} \times \frac{W}{32}$ instance feature maps to $\frac{H}{4} \times \frac{W}{4}$ instance feature maps, and finally use the instance feature maps with a convolution layer of 3×3 kernel size, and the masks of each object will be obtained after the convolution operation, according to which the masks can be obtained from the objects in the $\frac{H}{4} \times \frac{W}{4}$ size map of the object's relative position and region information.

Finally, the generated object masks are screened by Pixel-wise Argmax to fuse a mask with different instance ids, and finally Bilinear interpolation is taken to restore the instance id masks to the original image size. As mentioned above, Convolutional Multi-head Attention, which is similar to the MHA soft query mechanism, is used in the mask head to accomplish the pixel segmentation of the object, so the decoder outputs can be based on the FPN of the shared parameters at the same time in order to accomplish the object category and the bound box prediction, i.e., the N_q feature maps are obtained. The obtained N_q masks, classes and bounding boxes are one-to-one correspondence without any other post-processing, which saves a lot of workload to some extent.

C. *Loss function*

Due to the shared parameters FFN and mask head of DETR, three types of losses, segmentation, box, and class, are used for model training as in equation (4).

$$L = \lambda_{\text{seg}} L_{\text{seg}} + \lambda_{\text{box}} L_{\text{box}} + \lambda_{\text{cls}} L_{\text{cls}} \tag{4}$$

(a) Hungarian matcher

In the inference phase of the model, a collection of N predictions is obtained which are matched with Ground Truth (GT) in order to compute the loss, so a Hungarian matcher is needed to match the predictions with GT. The algorithm produces an optimal bisection match between the predicted and real objects, defining the set of matches as in equation (5).

$$\hat{\sigma} = \arg\min_{\sigma \in \mathfrak{S}_N} \sum_{i}^{N} L_{\text{match}}\left(y_i, \hat{y}_{\sigma(i)}\right) \tag{5}$$

$$L_{\text{match}}\left(y_i, \hat{y}_{\sigma(i)}\right) = -\mathrm{I}_{\{c_i \neq \emptyset\}} \hat{p}_{\sigma(i)}(c_i) + \mathrm{I}_{\{c_i \neq \emptyset\}} \mathcal{L}_{box}\left(b_i, \hat{b}_{\sigma(i)}\right) \tag{6}$$

where $\hat{\sigma}$ denotes the optimal pairing of the set of N predicted objects $\sigma \in \mathfrak{S}_N$ searched in the set of GTs with the lowest loss; $\hat{y}$ denotes the set of N predicted objects; y denotes the set of GTs populated by ϕ (non-object classes) by the number up to N; and $L_{\text{match}}(y_i, \hat{y}_{\sigma(i)})$ denotes the computation of between the true value y_i and the predicted value $\hat{y}_{\sigma(i)}$ of its matching Pair-Wise Matching Cost, which is a matching cost that takes into account the similarity between categories, bounding boxes and GTs at the same time. For each element of y, define $y_i = \{c_i, b_i\}$; where c_i denotes the category label of the object (including ϕ); and $b_i \in [0,1]^4$ denotes the coordinates of the center of the true frame, its vector of widths and heights with respect to the image size. The Pair-Wise Matching Cost for a predicted category c_i with probability $\hat{p}_{\sigma(i)}(c_i)$ for $\sigma(i)$ and a prediction frame $\hat{b}_{\sigma(i)}$ is shown in Equation (6).

The combined loss of each $\hat{y}_{\sigma(i)}$ and each object of y in the category, bounding box is calculated according to$L_{\text{match}}(y_i, \hat{y}_{\sigma(i)})$ and the predicted box $\hat{b}_{\sigma(i)}$ with the smallest loss is the best matching box for the real box b_i. Then the Hungarian loss of all the above pairings is calculated as in equation (7).

$$\mathcal{L}_{\text{Hungarian}}(y, \hat{y}) = \sum_{i=1}^{N} [-\log \hat{p}_{\hat{\sigma}(i)}(c_i) + I_{\{c_i \neq \emptyset\}} L_{box}(b_i, \hat{b}_{\hat{\sigma}(i)})] \tag{7}$$

(b) Bounding box loss

The second part of the above Hungarian matching algorithm is to score the bounding box, since DETR is directly predicting the bounding box by calculating the relative error L_1 paradigm loss and intersection over union ratio for scoring the bounding box, it will be due to the scale of the different sizes of the bounding box may appear to have a similar relative error, so the Generalized Intersection over Union (GIoU) loss and the L_1 loss is taken as the bounding box scoring L_{box} as in equations (8-9):

$$L_{\text{box}}(b_i, \hat{b}_{\sigma(i)}) = \lambda_{\text{iou}} L_{\text{iou}}(b_i, \hat{b}_{\sigma(i)}) + \lambda_{L1} \| b_i - \hat{b}_{\sigma(i)} \|_1 \tag{8}$$

$$L_{\text{iou}}(b_{\sigma(i)}, \hat{b}_i) = 1 - \left(\frac{|b_{\sigma(i)} \cap \hat{b}_i|}{|b_{\sigma(i)} \cup \hat{b}_i|} - \frac{|B(b_{\sigma(i)}, \hat{b}_i) \setminus b_{\sigma(i)} \cup \hat{b}_i|}{|B(b_{\sigma(i)}, \hat{b}_i)|} \right) \tag{9}$$

where λ_{iou},λ_{L_1} is a parameter that adjusts the proportion of GIoU and L_1 paradigm loss,. $|b_{\sigma(i)} \cap \hat{b}_i|$ denotes the area of the intersection between the true bounding box $b_{\sigma(i)}$ and the predicted bounding box $\hat{b}_i$, $|b_{\sigma(i)} \cup \hat{b}_i|$ denotes the area of the concatenated set; $|B(b_{\sigma(i)}, \hat{b}_i)|$ denotes the area of the minimum closure region, $|B(b_{\sigma(i)}, \hat{b}_i) \setminus b_{\sigma(i)} \cup \hat{b}_i|$ denotes the area of the minimum closure region and the area of the difference set of the concatenated set of $b_\sigma(i)$ and $\hat{b}_i$.

(c) Segmentation loss

The Segmentation mask can be viewed as a pixel-level classification problem for binary classification, which consists of a Dice loss L_{dice} and a Focal loss L_{focal}, i.e., $L_{\text{seg}} = \lambda_{\text{dice}} L_{dice} + \lambda_{\text{focal}} L_{\text{focal}}$, as in Eq. (10-13):

$$L_{\text{dice}}(\boldsymbol{m}, \hat{\boldsymbol{m}}) = 1 - \frac{2|\boldsymbol{m} \odot \text{Sigmoid}(\hat{\boldsymbol{m}})| + 1}{|\boldsymbol{m}| + |\text{Sigmoid}(\hat{\boldsymbol{m}})| + 1} \tag{10}$$

$$L_{\text{focal}}(\boldsymbol{m}, \hat{\boldsymbol{m}}) = \frac{1}{N} \cdot \text{Mean}(\alpha_t (1 - p_t)^\gamma \odot \text{BCE}(\boldsymbol{m}, \hat{\boldsymbol{m}})) \tag{11}$$

$$p_t = \text{Sigmoid}(\hat{\boldsymbol{m}}) \odot \boldsymbol{m} + (1 - \text{Sigmoid}(\hat{\boldsymbol{m}})) \cdot (1 - \boldsymbol{m}) \tag{12}$$

$$\alpha_t = \alpha \odot \boldsymbol{m} + (1 - \alpha)(1 - \boldsymbol{m}) \tag{13}$$

where λ_{dice},λ_{focal} ditto are the training hyperparameters; $\boldsymbol{m}, \hat{\boldsymbol{m}}$ denote the individual true mask and predicted mask probabilities of the match. BCE refers to Binary Cross Entropy, the loss function used for binary classification.

(d) Classification loss

Classification loss consists of counting loss and category loss: since DETR directly generates object prediction frames and categorization, there will be ϕ in the prediction, and the error of counting the number of non-ϕ and the number of GTs will be used as the counting loss for categorization scoring; secondly, the category loss is based on the calculation of cross entropy of matched prediction frame categories and GT categories, i.e., back to the calculation of the loss of the multiclassification problem, and the whole categorization loss L_{cls} consists of the counting loss and the categorization loss as shown in Eq. (14).

$$L_{\text{cls}} = \lambda_{L_1} \left|\left| N_{\hat{\sigma}} - N_{\sigma} \right|\right|_1 + \left(-\sum_{i}^{N} c_{\sigma(i)} \log(\hat{c}_i) \right) \tag{14}$$

D. Evaluation metric

In traditional image segmentation tasks, Average Pixel Accuracy is used as an evaluation metric, which only works on the pixel level; while in instance segmentation tasks, Average Precision (AP) and IoU are used as evaluation metrics, which only work on the instance level; therefore, metrics metrics for panorama segmentation have three major requirements:(a) Completeness: metrics should be handled in a unified way to (a) Completeness: metrics should be handled in a unified way for Stuff and Things classes, capturing all aspects; (b) Interpretability: evaluation metrics should be clearer and more interpretable compared to the model architecture; and (c) Simplicity: metrics are easy to define and implement, and can be computed efficiently to achieve fast evaluation. Therefore, Panoptic Quality (PG) is derived as an evaluation metric for panoptic segmentation by unifying all classes of Things and Stuff, as in equation (15).

$$\text{PQ} = \frac{\sum_{(p,\, g)\in \text{TP}} IoU(p,\, g)}{|\text{TP}| + \frac{1}{2}|\text{FP}| + \frac{1}{2}|\text{FN}|} = \underbrace{\frac{\sum_{(p,\, g)\in TP} \text{IoU}(p,\, g)}{|\text{TP}|}}_{\text{Segmentation Quality (SQ)}} + \underbrace{\frac{2|\text{TP}|}{2|\text{TP}| + |\text{FP}| + |\text{FN}|}}_{\text{Recognition Quality (RQ)}} \tag{15}$$

Where p denotes the predicted single object; g denotes the true labeling that matches p; TP (True Positives), FP (False Positives), and FN (False Negatives) denote the predicted and GT-matched segmentation segments, the mismatched predicted segments, and the set of mismatched GTs. SQ is the introduction of TP to the IoU, the The segmentation quality of Things and Stuff are combined to make the score; RQ is similar to the F1 score of the binary classification task, and its nature is the same as the F1 score, which combines the evaluation indexes of precision and recall.

III. Experiments

A. Dataset

MS COCO (Microsoft Common Object in Context) dataset is a large dataset first released by Microsoft in 2014, the team collects about 330,000 images of daily life scenes and manually annotates 1.5 million object instances on 200,000 of them, which contains 80 classes of target detection and segmentation, 91 classes of pixel level image segmentation, human body key-points, and other tasks of the labeling content. The panoptic segmentation annotation was added to the second version of Training and Validation proposed in 2017.

B. Results

Model training and validation were done on a Linux device with Intel(R) Xeon(R) Gold 6258R CPU @ 2.70GHz processor and 2 NVIDIA GeForce RTX 3080 Ti 12GB graphics cards. The training process setup hyperparameters are shown in Table 1.

Table 1. Hyperparameters for training.

Hyperparameter	Value	Hyperparameter	Value
N_{enc}	6	Dropout	0.1
N_{dec}	6	λ_{cls}	1
N	100	λ_{seg}	1
M	8	λ_{box}	5
d	256	λ_{dice}	1
d_{ff}	2048	λ_{focal}	1
λ_{iou}	2	λ_{L_1}	1

Here the trained completed weights of the DETR base architecture are introduced and only the mask head is trained with model weights. AdamW is taken as the optimizer and trained for 100 epochs, the experimental results obtained are compared with the base of other models as shown in Table 2.

Table 2. Experimental results PQ comparison.

Model	PQ	SQ	RQ	PQ^{th}	SQ^{th}	RQ^{th}	PQ^{st}	SQ^{st}	RQ^{st}
PanopticFPN++	42.4	79.3	51.6	49.2	82.4	58.8	32.3	74.8	40.6
UPSnet	42.5	78.0	52.5	48.6	79.4	59.6	33.4	75.9	41.7
DETR	**42.8**	78.6	**53.8**	48.1	78.9	59.4	**35.8**	**78.2**	**45.1**

From the above, DETR with the addition of mask head gets good results in COCO 2017 Validation experiments, and its comprehensive strength exceeds that of other architectures dedicated to panoptic segmentation in the same period. In panoptic segmentation, the pixel segmentation accuracy for STUFF far exceeds that of other models, and is slightly lacking in THINGS. It is noted that the accuracy in RQ, i.e., for instance classification, is higher than the previous ones in both synthesis, things and stuff, which proves that the features obtained from object queries of Transformer decoder can still maintain the independence and unity of the feature information when they are used in both the shared parameter FFN and the mask head, and the generated instance segmentation maps have a strong correlation with the instance segmentation maps. The generated instance segmentation graph and the category probability have a strong correlation, and the DETR-based architecture obtains a higher quality performance while completing the task than other multi-branch task architectures that are also used for panoptic segmentation. The visualization part of the results is shown in Figure 8.

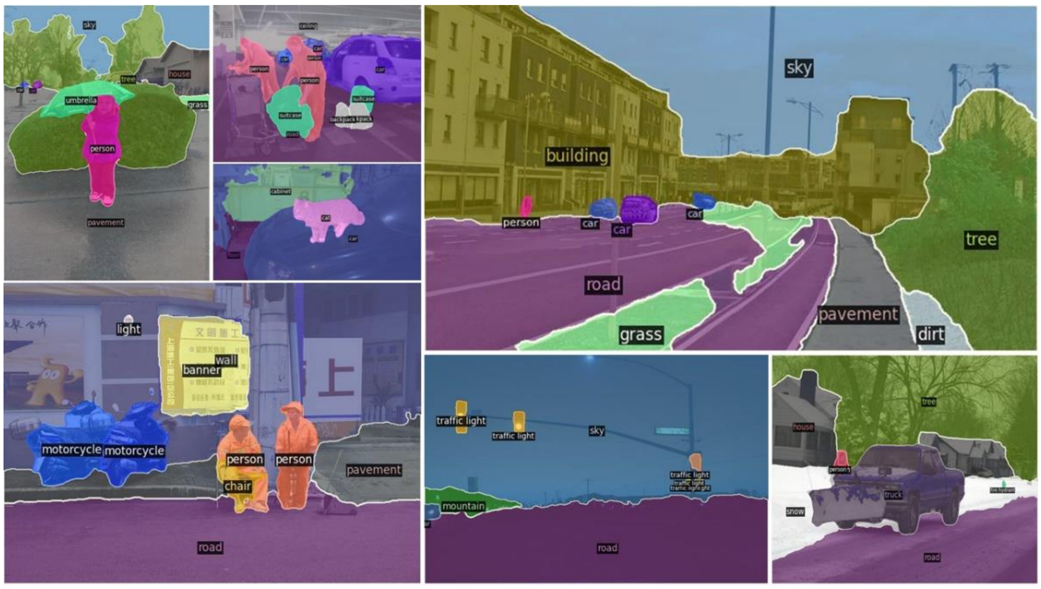

Figure 8. Visualization of small part segmentation figure results.

IV. Conclusions

In this paper, on the basis of DETR with end-to-end object detection architecture, a new mask head with similar style to Mask R-CNN is added to accomplish the segmentation map task of converting features into panoptic segmentation format. the FPN-style multiscale fusion features taken in the mask head , the loss function is increased by dice and focal loss, both of which are optimized to some extent for the extreme imbalance in the number of stuff and things samples (pixel points) in the panoptic segmentation task. For the processing of the final output part, the shared-parameter FFN and the Convolutional Multi-head Attention mask head based on the MHA soft query not only handle the output of the transformer encoder and decoder properly, but also solve the problem of instance segmentation pixel loss in the previous panoptic segmentation in which instance segmentation pixel region and category classification information are not related. The above is only the parameters of the mask head trained on the basis of the DETR infrastructure, if it start training the whole architecture from 0, we may be able to achieve higher accuracy.

Reference

[1] Y. Lecun, L. Bottou, Y. Bengio, et al., “Gradient-Based Learning Applied to Document Recognition”, *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-3324, 1998.

[2] A. Krizhevsky, I. Sutskever, G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks”, *Advances in Neural Information Processing Systems*, vol. 25, pp. 1106-1114, 2012.

[3] K. Simonyan, A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition”, *3rd International Conference on Learning Representations (ICLR 2015)*, 2015.

[4] K. He, X. Zhang, S. Ren, et al., “Deep Residual Learning for Image Recognition”, *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016*), pp. 770-778, 2016.

[5] J. Long, E. Shelhamer, T. Darrell, “Fully convolutional networks for semantic segmentation”, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2015)*, pp. 3431-3440, 2015.

[6] O. Ronneberger, P. Fischer, T. Brox, “U-Net: Convolutional Networks for Biomedical Image Segmentation”, *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015 - 18th International Conference Munich*, vol. 9351, pp. 234-241, 2015.

[7] F. Milletari, N. Navab, S. A. Ahmadi, “V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation”, *Fourth International Conference on 3D Vision (3DV 2016)*, pp. 565-571, 2016.

[8] B. Hariharan, P. A. Arbeláez, R. B. Girshick, et al., “Simultaneous Detection and Segmentation”, *Computer Vision - ECCV 2014 - 13th European Conference*, vol. 8695, pp. 297-312, 2014.

[9] K. He, G. Gkioxari, P. Dollár, et al., “Mask R-CNN”, *IEEE International Conference on Computer Vision (ICCV 2017)*, pp. 2980-2988, 2017.

[10] T. Y. Lin, M. Maire, S. J. Belongie, et al., “Microsoft COCO: Common Objects in Context”, *Computer Vision - ECCV 2014 - 13th European Conference*, pp. 740-755, 2014.

[11] D. Bolya, C. Zhou, F. Xiao, et al., “YOLACT: Real-Time Instance Segmentation”, *2019 IEEE/CVF International Conference on Computer Vision (ICCV 2019*), pp. 9156-9165, 2019.

[12] A. Kirillov, K. He, R. B. Girshick, et al., “Panoptic Segmentation”, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2019)*, pp. 9404-9413, 2019.

[13] A. Vaswani, N. Shazeer, N. Parmar, et al., “Attention is All You Need”, *Advances in Neural Information Processing Systems: Vol. 30*, pp. 5998-6008, 2017.

[14] A. Dosovitskiy, L. Beyer, A. Kolesnikov, et al., “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, *9th International Conference on Learning Representations (ICLR 2021)*, 2021.

[15] N. Carion, F. Massa, G. Synnaeve, et al., “End-to-End Object Detection with Transformers”, *Computer Vision - ECCV 2020 - 16th European Conference*, vol. 12346, pp. 213-229, 2020.

[16] S. Zheng, J. Lu, H. Zhao, et al., “Rethinking Semantic Segmentation From a Sequence-to-Sequence Perspective With Transformers”, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2021*), pp. 6881-6890, 2021.

[17] B. Cheng, A. G. Schwing, A. Kirillov, “Per-Pixel Classification is Not All You Need for Semantic Segmentation”, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021 (NeurIPS 2021)*, pp. 17864-17875, 2021.