EasyChair Preprint

№ 11138

TransCenter: Transformer in Heatmap and a New Form of Bounding Box

Deqi Liu, Aimin Li, Mengfan Cheng and Dexu Yao

October 23, 2023

TransCenter: Transformer in Heatmap and A New Form of Bounding Box

Deqi Liu[1], Aimin Li[2], Mengfan Cheng[3], Dexu Yao[4]

[1] Key Laboratory of Computing Power Network and Information Security, Ministry of Education, Shandong Computer Science Center, Qilu University of Technology (Shandong Academy of Sciences), Jinan, China
[2] Shandong Engineering Research Center of Big Data Applied Technology, Faculty of Computer Science and Technology, Qilu University of Technology (Shandong Academy of Sciences), Jinan, China
[3, 4] Shandong Provincial Key Laboratory of Computer Networks, Shandong Fundamental Research Center for Computer Science, Jinan, China
15910804870@163.com[1], lam@qlu.edu.cn[2], 10431210366@stu.qlu.edu.cn[3], 10431210601@stu.qlu.edu.cn[4]

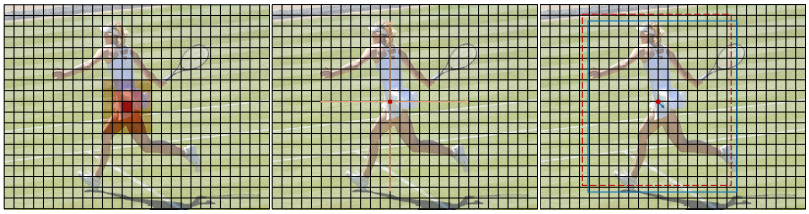
Abstract. In current heatmap-based object detection, the task of heatmap is to predict the position of keypoints and its category. However, since objects of the same category share the same channel in the heatmap, it is possible for their keypoints to overlap. When this phenomenon occurs, existing heatmap-based detectors are unable to differentiate between the overlapping keypoints. To address the above issue, we have designed a new heatmap-based object detection model, called TransCenter. Our model decouples the tasks of predicting the object category and keypoint position, and treats object detection as a set prediction task. We use a label assignment strategy to divide the predicted sets into positive and negative samples for training. The purpose of this is to allow different objects to have their own heatmap channel without sharing with other, thereby completely eliminating the occurrence of overlapping. To make the model easier to learn, we leverage the characteristic that heatmaps can reduce the solution space, proposed a novel approach for predicting bounding boxes. We use the encoder-decoder structure in transformers, treat the prediction of bounding boxes as an encoding task, use the form of a heatmap to represent the position and size. Then, we treat category prediction and offset prediction of the bounding box as decoding tasks, where the offset prediction is outputted through regression.

Keywords: Heatmap, Keypoint, Object Detection, Keypoint Overlap.

1 Introduction

In current object detection algorithms, there are two types of output formats for prediction. One is to directly output specific coordinates, which is based on regression [1-6]. The other is to output a Gaussian heatmap of object keypoints, which is based on heatmap [7-9], the output is shown in Figure 1. The second method is actually more like a classification task. Initially, the heatmap-based method was often used in the field

of human pose estimation [10-11]. Later, as object detection algorithms developed, people applied the heatmap-based method to object detection. One of the most intuitive advantages of this method is that the model does not require the construction of complex anchors, because each pixel in the heatmap can be approximated as an anchor [12-14] in a sense.



(a) (b) (c)

Fig. 1. Representation of bounding boxes in heatmap-based models. (a) Illustrates how a heatmap represents the position of an keypoint. (b) Represents the regression of width and height at that location. (c) Represents the offset of keypoint.

Unlike regression-based approaches, directly output the 2D coordinates of an object is an extremely nonlinear process [15]. It is also a more challenging form of supervised learning, as the network has to independently convert spatial positions into coordinates. The heatmap-based method utilizes the explicit rendering of the Gaussian heatmap, allowing the model to learn the output target distribution by learning a simple filtering method that filters the input image into the final desired Gaussian heatmap [16]. This greatly simplifies the learning difficulty of the model and is very consistent with the characteristics of convolutional. Furthermore, the regression-based method has a faster training and inference speed and can achieve end-to-end full-differentiation training, but they are prone to overfitting and have poor generalization ability. Compared with the regression, the heatmap-based method specifies the learned distribution, which is more robust for various situations (occlusion, motion blur, truncation, etc.) than it. Additionally, the heatmap-based method can explicitly suppress the response at non-keypoints.

After analyzing the strengths and weaknesses of both output forms, we decided to integrate them in our model. Our model employs CNN as the backbone to extract low-level features, and then utilizes Transformer to capture global dependencies at a higher level. We assign the task of predicting bounding boxes to the encoding layer of the Transformer. Unlike any other forms of bounding box prediction, we approximate both the position and size prediction of bounding boxes as a classification task, using the form of Gaussian heatmap for output. This approach significantly reduces the solution space and makes it easier for the network to learn. However, relying solely on the prediction of bounding boxes through the encoding layer is not accurate enough, as the limitations of the heatmap result in the position coordinates and sizes being quantized, leading to a significant error when mapping the predicted bounding boxes to the original image. To reduce the prediction error, we not only classify the object categories in

the decoding layer, but also use regression to more accurately predict the position and size offsets of the bounding boxes, thus achieving more precise localization.

In current heatmap-based models, the task of the heatmap is not only to predict the position of keypoints, but also to predict its category. The heatmap has K channels, which is equal to the number of categories in the dataset, each channel is responsible for predicting different category. When multiple objects of the same category share one channel, the overlap of their keypoints is inevitable, Figure 2 shows the overlap of keypoints. This is why larger heatmap sizes lead to better detection in these models. Larger heatmap can preserve more feature information and ensure sufficient distances between the keypoints of different objects, enabling the model to distinguish them effectively. In our model, we implemented a simple solution to this problem. By decoupling the task of predicting keypoint position from its category prediction, we were able to assign one heatmap exclusively to each object, ensuring that each keypoint corresponded to a unique location on the heatmap, and the constraint of heatmap size on the model is alleviated.

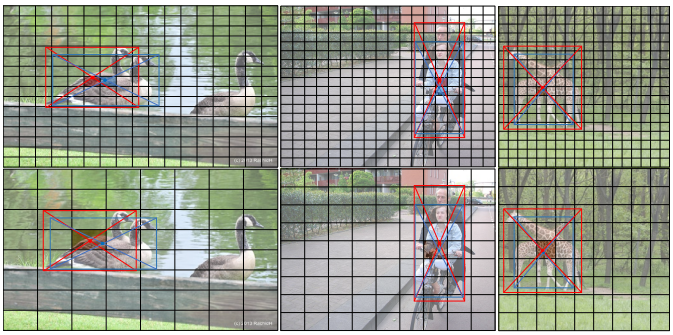


Fig. 2. The figure above shows the situation when the geometric center of an object is used as a keypoint. As the model gets deeper, the downsampling rate of the image increases, which may cause different object keypoints to appear in the same cell.

As we have decoupled the task of object classification from that of position prediction, in order to establish accurate association between the objects and their bounding boxes, we have drawn inspiration from DETR [17]. We consider an image as a set, with the objects in the image being the items in this set. Each item is composed of a category label and a bounding box. Our model predicts a fixed-size set, with the set size N being much larger than the number of objects present in the image. During the training process, we match each predicted object with every object in the ground-truth set, generating a matching cost for each pair of predicted and ground-truth objects. We then use these matching costs to assign positive and negative samples, and train the model accordingly.

We summarize our contributions as follows:

- We proposed a novel form of predicting bounding boxes, which is different from any previous output forms. We approximate the prediction of the location and size of the bounding box as a classification problem, using the response values at each position in the heatmap to determine the specific location and

size of the bounding box. This approach can narrow down the solution space and make the network easier to learn.

- Unlike other heatmap-based object detectors, we decoupled the category prediction from the keypoint position prediction. This allows each object to have its own heatmap, thus eliminating the overlapping keypoints problem that arises from multiple objects sharing the same heatmap channel.

2 Related Work

2.1 Heatmap in Human Pose Estimation

The heatmap-based approach has become the mainstream method in this field. This approach trains the model to learn a Gaussian probability distribution map by rendering each point in the ground-truth as a Gaussian heatmap. The network output consists of K heatmaps, corresponding to K keypoints, and the final estimation is obtained by using argmax or soft-argmax to locate the point with the highest value.

2.2 Heatmap in Object Detection

The most mainstream approach in the object detection field is still obtaining the position and size of the bounding box through direct regression. However, many heatmap-based object detection models have emerged so far. These models can be broadly categorized into two types. The first type outputs one keypoint of the object through a heatmap, such as CenterNet [9], which considers the geometric center of the object as the keypoint, and then obtains the precise size of the object through regression. The second type predicts multiple different keypoints of the object. Then, through some matching method, the keypoints belonging to the same object are associated together to determine the specific position and size of the bounding box, thus avoiding direct regression of coordinates and size. Representatives of this type of method include CornerNet [7], which determines the bounding box by predicting the two diagonal points of an object, and ExtremeNet [18], which uses five heatmaps to predict the four extreme points and central region of an object, etc. However, since heatmaps can only predict a rough position, such methods still require regression to obtain position offsets of keypoints for more precise adjustments.

2.3 Transformer with Heatmap

The Transformer was originally proposed by Vaswani et al.[19] and was initially applied to the field of natural language processing. In recent years, the Transformer has also gained significant attention in the field of computer vision [20, 21]. Sen Yang et al.[22] applied the Transformer to the task of heatmap prediction, and only used the encoder. They believed that pure heatmap prediction is simply an encoding task, and that the Transformer-based keypoint localization method is consistent with the interpretability of activation maximization [23]. Up to now, there are very few methods that

use the Transformer for heatmap prediction tasks, and most of them combine the Transformer with regression techniques. Therefore, this novel combination of the Transformer with heatmap prediction is a bold attempt for us.

3 Method

3.1 Model Structrue

Usually, heatmap-based models choose to use HourglassNet [24] to produce high-resolution feature maps, as this network structure is capable of capturing and integrating information at all scales of the image. However, our network uses a lighter backbone, ResNet-50 [25], instead. We pass the extracted low-level features to a Transformer to obtain a more advanced feature representation. Our Transformer consists of an encoder and a decoder. The main task of the encoder is to perform a rough prediction of the bounding box, including its position and size. The main task of the decoder is to predict the object category and adjust the bounding box, which includes position offsets and size offsets. The overall structure of the model is shown in Figure 3.

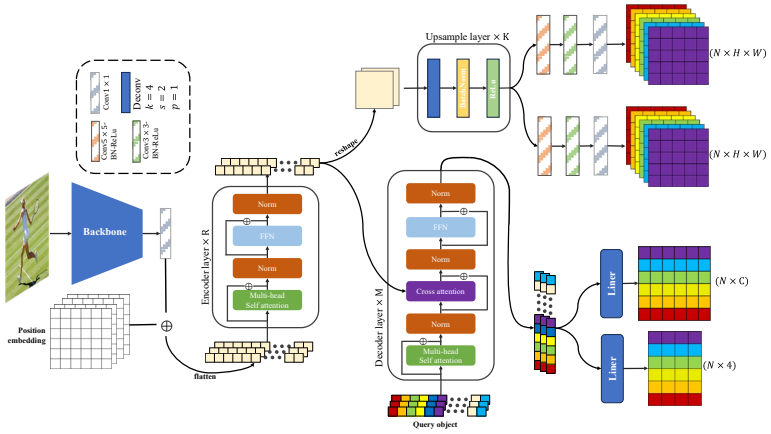


Fig. 3. Structure of the model. Our model predicts a set of size N, where each item in the set consists of a position heatmap, a size heatmap, a category, and offsets to ensure that each object has its own individual heatmap.

The low-level features extracted by the CNN are first compressed in channel dimension via a convolutional layer with a kernel size of 1, and then sent to the encoder of Transformer. In the encoder, since the feature maps output by the backbone is flattened into a 1D sequence of pixels, the Transformer can calculate the correlation between each pixel and all other pixels of the feature maps. The encoder consists of several encoding layers, each composed of multi-head self-attention and a FFN (feedforward neural network). A normalization module follows each module. The output of the encoding layer is fed into a continuous upsampling operation [26] before being fed into the bounding box prediction head, which is composed of convolutional layers. In the

first two layers of the prediction head, larger-sized convolutional kernels are used to aggregate information from the feature map. Finally, a convolutional kernel with a size of 1 is used to obtain the position and size of bounding boxes, which are output as Gaussian heatmaps.

The output of the encoder is also fed into the decoder. The decoder consists of multiple decoding layers, each composed of multi-head self-attention, cross-attention, and FFN. Like the encoding layers, there is also a normalization module following each module in the decoder. The output of the decoder is then separately sent to the classification head and offset prediction head, both of which are composed of linear layers.

3.2 Bounding Box in Heatmap Form

In current mainstream object detection models, the predicted form of bounding boxes are usually in the form of numerical values, which is a common approach in regression-based models. In heatmap-based models, however, only the coordinate of the bounding box is output as heatmap, while the size of the bounding box is obtained as specific values through regression. In our experiments, we have demonstrated that the size of the bounding box can also be obtained in the form of heatmap, as shown in Figure 4. We can view the heatmap as a 2D coordinate system with limited width and height, and for the position of an object's keypoint (we consider the geometric center of the object as the keypoint), we can determine them based on the response value at each coordinate. For the size of the object, it is also a 2D data consisting of width and height, which can be output in heatmap form as well. The x-value of this coordinate can represent the width of the object, while the y-value represents its height. This output format greatly reduces the prediction difficulty of the network, and allows for faster convergence of the network.

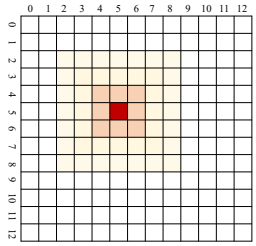


Fig. 4. Represent the bounding box in the form of a heatmap. The darker the color on the heatmap, the higher the response value. If the image represents a position heatmap, the horizontal axis denotes the x-coordinate of the keypoint, and the vertical axis symbolizes the y-coordinate. Consequently, the keypoint is at the (5, 5) coordinates. Alternatively, if the image represents a size heatmap, the horizontal axis signifies the object's width, while the vertical axis represents its height, thus, the size can be expressed as (5, 5).

Predicting bounding boxes using heatmaps can be approximated as a classification task. In this case, the coordinates in the heatmap can be considered as “categories”. As the decreases of size, during the initial model learning phase, the probability of

"guessing" the correct "category" increases. Thus, the size of heatmap determine the lower limit of the model. Similarly, due to the heatmap's constraints, coordinates can only appear as integers. Thus, when a bounding box is mapped back to its original-sized image after downsampling, it inevitably results in quantization errors. A smaller solution space also imposes limitations on the model's upper limit. To effectively compensate for the generated errors, we predict the offsets of the position and size through a regression method following the decoding layer.

3.3 Offset

In heatmap-based models, the ground-truth representation of offsets is illustrated as in Equation 1.

$$\begin{cases} x - \lfloor x \rfloor \\ y - \lfloor y \rfloor \\ w - \lfloor w \rfloor \\ h - \lfloor h \rfloor \end{cases} \tag{1}$$

In this formula, x and y represent center coordinates, while w and h denote width and height. However, such a representation may not be suitable for our model, as during the training process, the offset loss struggles to decrease significantly. After comparing the format of labels for offsets in CenterNet [9], we found that, offset regression is performed for specific locations. However, using Equation 1 to calculate offsets is not effective in representing spatial positions. Therefore, we have chosen to use Equation 2 to create the labels for offsets.

$$\begin{cases} (x - \lfloor x \rfloor) * e^{(\lfloor x \rfloor / S_w)} \\ (y - \lfloor y \rfloor) * e^{(\lfloor y \rfloor / S_h)} \\ (w - \lfloor w \rfloor) * e^{(\lfloor w \rfloor / S_w)} \\ (h - \lfloor h \rfloor) * e^{(\lfloor h \rfloor / S_h)} \end{cases} \tag{2}$$

In the formula, S_w and S_h represent the width and height of the heatmap. We multiply the offset with the coordinates and size, thus incorporating spatial information into offset. The reason for using exp is that after quantization, the coordinates and sizes may become 0. The division by the size of heatmap is for normalization.

3.4 Label Assignment

Due to the fact that the predicted set is far greater than the ground-truth set, we need to divide the predicted set into positive and negative samples, with the number of positive equaling the number of ground-truth items. We use the Hungarian algorithm to assign labels, and the cost matrix will be constructed using classification cost, L1 cost, and GIOU cost. When computing the L1 and GIOU costs between predicted results and ground-truth, we first map these three components (position heatmap, size heatmap,

and offset) to the image space to obtain specific bounding boxes. Subsequently, by utilizing the Hungarian algorithm, we perform a one-to-one pairing match between the predicted boxes and ground-truth boxes. The reason for selecting this holistic approach for calculation is because if we separately calculate the costs for these three components, the weightings of each component are not easy to balance, which could lead to a certain part dominates the allocation of samples.

3.5 Loss Function

For the calculation of the losses, we did not convert the output into bounding boxes as in calculating costs, but instead calculated the losses for each component of the model separately.

For the prediction of categories, we defined the output format of the network as (B, N, C+1), where B is the batch size, N is the fixed size of the set, and C is the number of categories. We set C+1 categories in total, with the additional one defined as the background. In order to avoid the interference of a large number of background classes, we set the weight of the background to 0.1. For the calculation of classification loss, we chose to use binary cross-entropy function, as shown in Equation 3.

$$L=\frac{-1}{pos+neg*0.1}\sum_{i}^{N}\begin{cases}[y_i\log(\hat{y}_i)+(1-y_i)\log(1-\hat{y}_i)] & if\ \ y_i\ \ not\ \ background\\ [y_i\log(\hat{y}_i)+(1-y_i)\log(1-\hat{y}_i)*0.1] & otherwise\end{cases} \tag{3}$$

Where *pos* signifies the quantity of positive samples, and *neg* signifies the quantity of negative samples, $pos+neg=N$.

When calculating the loss of the heatmap, instead of equally penalizing negative locations, we reduce the penalty given to negative locations within a radius of the positive location. This is because even if a negative bounding box is close enough to its ground-truth, it can still result in a bounding box that overlaps sufficiently with the ground-truth box. In our model, we use Gaussian heatmaps for both the position heatmaps and the size heatmaps when labeling ground-truth values. The outputs of these two heatmaps are both formatted as (B, N, H, W), where N is a fixed set size and H, W are the size of heatmaps. We use Gaussian Focal Loss to calculate the losses, all channels are involved in the calculation, as shown in Equation 4.

$$L=\frac{-1}{pos+neg*0.1}\sum_{n=1,y=1,x=1}^{N,H,W}\begin{cases}(1-p_{nyx})^{\alpha}\log(p_{nyx}) & if\ \ g_{nyx}=1\\ (1-g_{nyx})^{\beta}(p_{nyx})^{\alpha}\log(1-p_{nyx}) & otherwise\end{cases} \tag{4}$$

Where H and W are the size of heatmap, α and β are two hyperparameters, we use $\alpha=2$ and $\beta=4$. p_{nyx} is the prediction value in (x,y), and the weight of penalty g_{nyx} at location (x,y) is calculated based on the Gaussian radius r, as shown in Equation 5.

$$g_{nyx}=\exp(\frac{(x-\hat{x})^2+(y-\hat{y})^2}{-2\varphi^2}) \tag{5}$$

Where $(\hat{x}, \hat{y})$ denote the positive coordinates, and $x \in [\hat{x} - r, \hat{x} + r], y \in [\hat{y} - r, \hat{y} + r]$, φ is an object size-adaptive standard deviation, default $\varphi = \frac{2r+1}{6}$.

When calculating the offset loss, we use the SmoothL1 loss. Unlike the calculation for heatmaps, we only select positive samples for calculation, because in the offset values, 0 represents a distance, while in heatmaps, 0 represents "none". These have completely different properties, and it is meaningless to calculate the offset for a non-existent bounding box.

4 Experiment

Our experiments were carried out on the PASCAL VOC 2007+2012 [28] dataset. On the PASCAL VOC dataset, we used 17K labeled images from the entire dataset for training and 2K labeled images for validation. Training was conducted using a single A100 GPU. Experimental results demonstrate that the model is effective in distinguish objects with overlapping centers because each object has its own independent heatmap, as shown in Figure 5.

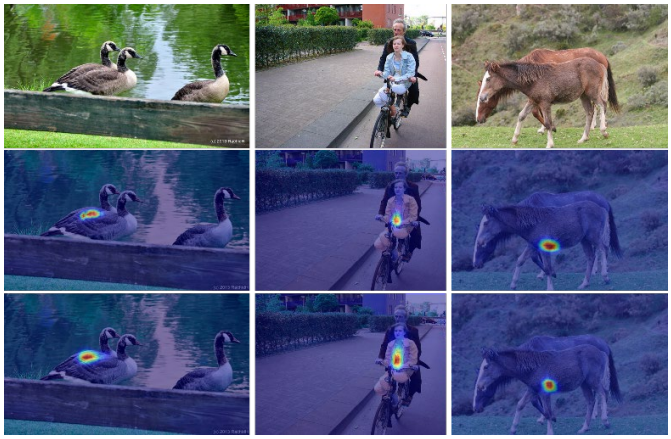


Fig. 5. When the centers of the objects overlap, the heatmap channels of the two objects are independent of each other, so it does not hinder the model discrimination.

We set the input image dimensions to 512×512 and chose ResNet-50 as the backbone. After downsampling, the size of feature map fed into the transformer is 16×16. The output from the encoding layer undergoes upsampling via transposed convolution, resulting in our final heatmap size of 64×64. We counted the number of parameters and Flops of other heatmap-based models, as shown in Table 1. In contrast, the number of parameters and Flops of our model are far less than them. We compared our model's detection performance with other models, demonstrating that our model achieves good detection results while having significantly fewer parameters and computation requirements than other models, as shown in Table 2.

Table 1. Our model has far fewer parameters and Flops than other heatmap-based models.

Model	Flops	Params
CornerNet [7]	452.96G	201.04M
CentripetalNet [27]	491.70G	205.76M
CenterNet [9]	292.70G	191.25M
TransCenter 16×16 (Our)	25.07G	45.79M
TransCenter 32×32 (Our)	28.81G	46.84M
TransCenter 64×64 (Our)	43.74G	47.89M
TransCenter 128×128 (Our)	103.46G	48.94M

Table 2. AP comparison of our model with other heatmap-based models.

Model	$AP_{0.5:0.95}$	$AP_{0.5}$	$AP_{0.75}$	AP_S	AP_M	AP_L
CenterNet	54.4	78.2	59.5	16.7	34.1	63.9
CornerNet	55.9	72.3	59.3	12.7	35.6	64.7
CentripetalNet	57.2	77.0	61.0	26.1	37.5	65.6
TransCenter (Our)	**53.7**	**77.6**	**58.7**	**17.4**	**32.3**	**64.9**

We find that these models use HourglassNet to extract the underlying features. The advantage of HourglassNet is that it can output a high-resolution feature map, capture and integrate the information of all scales of the image, but the cost is that it needs to pay a huge amount of calculations and parameters, as shown in Table 3. In addition, these models also use keypoint pooling to improve the detection effect, which also requires a lot of computing resources. In contrast, our model is much more lightweight.

Table 3. Parameters and Flops of each module.

Model	Flops	Params
HourglassNet [24]	234.522G	187.7M
ResNet-50 [25]	20.366G	23.508M
CornerPooling [7]	25.3G	1.542M
CenterPooling [27]	39.812G	2.427M

4.1 Lower Limit of Model

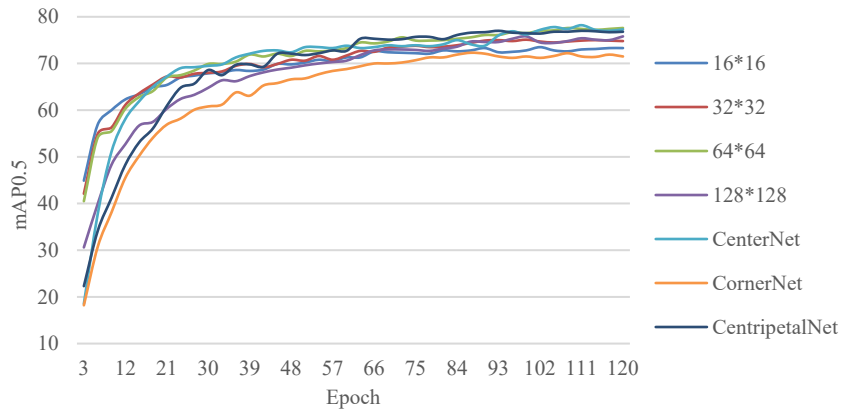


Fig. 6. It can be seen from the curve that mAP0.5 of our model is much higher than other models in the early training period.

In the experiment, we adjusted the size of the heatmap to 16×16 , 32×32 , 64×64 and 128×128 . After many comparative experiments, we find that the lower limit of our model is much higher than other models. Our model has reached 44.9 (16×16), 42.3 (32×32), 40.5 (64×64) and 38.2 (128×128) mAP0.5 in the initial rounds of training, as shown in Figure 6. Two points can be seen from this set of data, first, the lower limit of the model is inversely proportional to the size of the heatmap. Secondly, our model only needs less time cost to achieve a relatively satisfactory detection effect. At present, the evaluation indicators of the model are all aimed at the upper detection limit of the model, but we believe that a higher lower limit of the model can make more trade-offs between time cost, equipment cost and detection effect.

4.2 Upper Limit of Model

Table 4 shows the detection effect of the model with different scales of heatmaps. We find that the size of the heatmap is not necessarily proportional to the upper limit of the model. Because the size affects the model in many ways. As the heatmap size decrease, the position and size prediction become easier. However, this does not necessarily translate to better model performance. Smaller size makes position and size predictions easier, but also rough. Consequently, the role of offset becomes much more apparent. Suppose the input image size is 512×512 , and the output heatmaps are 16×16 . In this case, a 0.5 offset maps to $0.5/16\times512=16$ pixels in the original image. As the size increase, the position and size prediction become more difficult. Conversely, the impact of offset on the prediction results will diminish.

Table 4. We conducted several comparative experiments and proved that the heatmap size of 64×64 has the best detection results. Moreover, the Flops and the number of parameters at this size are only 15% and 25% of CenterNet's respectively. Compared to CentripetalNet, Flops has only 8.9% of it and 23.3% of its parameters.

Model	$AP_{0.5:0.95}$	$AP_{0.5}$	$AP_{0.75}$
TransCenter(16×16)	50.5	73.5	56.3
TransCenter(32×32)	52.4	75.1	57.7
TransCenter(64×64)	53.7	77.6	58.7
TransCenter(128×128)	52.8	75.8	59.1

In addition to this, the heatmap's learning capabilities for position and size differ. Regarding position learning, an object's feature information generally gathers at its location, making convolution operations well-suited since their role is to aggregate local information. In contrast, learning size is relatively more challenging since there is no inherent relationship between the size and position. It is difficult to aggregate complete feature information through convolutional local operations and convert it into size, as shown in Figure 7. Furthermore, the feature map size has varying effects on the detection performance of objects of different sizes. Larger feature maps help capture smaller objects, while smaller feature maps are more accommodating for larger objects. Therefore, for our model, a larger heatmap size is not always better, nor is a smaller size.

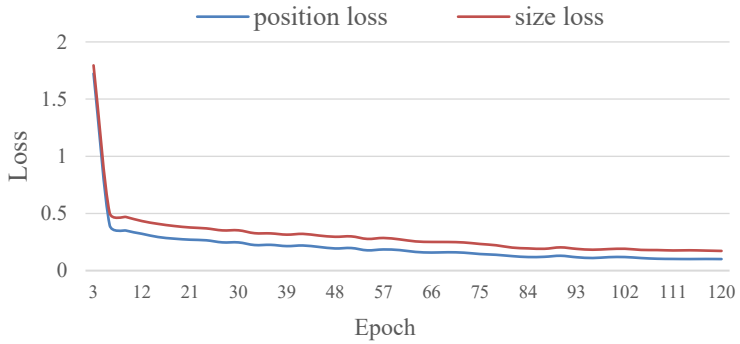


Fig. 7. The position loss decreases faster than the size loss, which fully demonstrates that the model's ability to learn size and position is not the same.

Conclusion

We proposed a novel method to predict the position and size of the bounding box in the form of heatmap, so as to greatly reduce the solution space, which is more conducive to the learning of the model, and also improves the prediction lower limit of the model. We decouple the position prediction task from the category prediction task, thus thoroughly solving the problem of keypoint overlap in heatmap-based models. Although

the current experimental results are not enough to reach the level of SOTA, but compared with other heatmap-based models, we have fewer parameters and less computation, and this cost is completely acceptable. We will continue this research direction and continue to optimize our model to achieve better detection results.

Acknowledgment

This work was supported by the Key R&D Plan of Shandong Province, China (No.2021CXGC010102).

Reference

1. J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 779-788, doi: 10.1109/CVPR.2016.91.
2. J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 6517-6525, doi: 10.1109/CVPR.2017.690.
3. Redmon, Joseph and Ali Farhadi. “YOLOv3: An Incremental Improvement.” ArXiv abs/1804.02767 (2018): n. pag.
4. Bochkovskiy, Alexey et al. “YOLOv4: Optimal Speed and Accuracy of Object Detection.” ArXiv abs/2004.10934 (2020): n. pag.
5. R. Girshick, J. Donahue, T. Darrell and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 2014, pp. 580-587, doi: 10.1109/CVPR.2014.81.
6. K. He, X. Zhang, S. Ren and J. Sun, "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 37, no. 9, pp. 1904-1916, 1 Sept. 2015, doi: 10.1109/TPAMI.2015.2389824.
7. Law, H., Deng, J. CornerNet: Detecting Objects as Paired Keypoints. Int J Comput Vis 128, 642–656 (2020).
8. K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang and Q. Tian, "CenterNet: Keypoint Triplets for Object Detection," 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South), 2019, pp. 6568-6577, doi: 10.1109/ICCV.2019.00667.
9. Zhou, X., Koltun, V., Krähenbühl, P. (2020). Tracking Objects as Points. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, JM. (eds) Computer Vision – ECCV 2020. ECCV 2020. Lecture Notes in Computer Science(), vol 12349. Springer, Cham.
10. Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. In Int. Conf. Comput. Vis., pages 2961–2969, 2017.
11. Sven Kreiss, Lorenzo Bertoni, and Alexandre Alahi. Pifpaf: Composite fields for human pose estimation. In IEEE Conf. Comput. Vis. Pattern Recog., pages 11977–11986, 2019.
12. S. Ren, K. He, R. Girshick and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137-1149, 1 June 2017, doi: 10.1109/TPAMI.2016.2577031.
13. Girshick, Ross B.. “Fast R-CNN.” 2015 IEEE International Conference on Computer Vision (ICCV) (2015): 1440-1448.

14. Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. 2016. R-FCN: object detection via region-based fully convolutional networks. In Proceedings of the 30th International Conference on Neural Information Processing Systems (NIPS'16). Curran Associates Inc., Red Hook, NY, USA, 379–387.
15. Nibali, Aiden et al. “Numerical Coordinate Regression with Convolutional Neural Networks.” ArXiv abs/1801.07372 (2018): n. pag.
16. Jin, Haibo et al. “Pixel-in-Pixel Net: Towards Efficient Facial Landmark Detection in the Wild.” International Journal of Computer Vision 129 (2020): 3174 - 3194.
17. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S. (2020). End-to-End Object Detection with Transformers. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, JM. (eds) Computer Vision – ECCV 2020. ECCV 2020. Lecture Notes in Computer Science(), vol 12346. Springer, Cham.
18. Zhou, Xingyi et al. “Bottom-Up Object Detection by Grouping Extreme and Center Points.” 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2019): 850-859.
19. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17). Curran Associates Inc., Red Hook, NY, USA, 6000–6010.
20. Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. Stand-alone self-attention in vision models. In NIPS, pages 68–80, 2019.
21. Irwan Bello, Barret Zoph, Quoc Le, Ashish Vaswani, and Jonathon Shlens. Attention augmented convolutional networks. In ICCV, pages 3286–3295, 2019.
22. S. Yang, Z. Quan, M. Nie and W. Yang, "TransPose: Keypoint Localization via Transformer," 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 2021, pp. 11782-11792, doi: 10.1109/ICCV48922.2021.01159.
23. Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. Technical Report, University of Montreal, 1341(3):1, 2009.
24. Newell, A., Yang, K., Deng, J. (2016). Stacked Hourglass Networks for Human Pose Estimation. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds) Computer Vision – ECCV 2016. ECCV 2016. Lecture Notes in Computer Science(), vol 9912. Springer, Cham. https://doi.org/10.1007/978-3-319-46484-8_29
25. K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90.
26. M. D. Zeiler, D. Krishnan, G. W. Taylor and R. Fergus, "Deconvolutional networks," 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2010, pp. 2528-2535, doi: 10.1109/CVPR.2010.5539957.
27. Z. Dong, G. Li, Y. Liao, F. Wang, P. Ren and C. Qian, "CentripetalNet: Pursuing High-Quality Keypoint Pairs for Object Detection," 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 10516-10525, doi: 10.1109/CVPR42600.2020.01053.
28. Everingham M , Gool L V , Williams C K I ,et al.The Pascal Visual Object Classes (VOC) Challenge[J].International Journal of Computer Vision, 2010, 88(2):303-338.DOI:10.1007/s11263-009-0275-4.