A Comparison on Instance Segmentation Models

Nipun Anoob[1], Sanju Jacob Ebey[2], Praveen P[3], Prasidh Prabudhan [4], Paul Augustine[5]
[1,2,3,4,5]Rajagiri School of Engineering & Technology, Kochi, India
Email: nipunanoob@gmail.com[1], sanju.ebey@gmail.com[2],
praveenpradeepkumar44@gmail.com[3], prasidhprabudhan123@gmail.com[4], paula@rajagiritech.edu.in[5]

2021 10th International Conference on Advances in Computing and Communications (ICACC) | 978-1-6654-3919-0/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICACC-202152719.2021.9708272

***Abstract*—Object detection is an important process in computer vision projects/tasks. It is a technique which tries to predict the location of an object by drawing a bounding box around it. This however does not give us any idea about the actual shape of the object. For this we must employ the next stage of computer vision task which is known as Instance Segmentation. This task can be used to find the shape of an object along with its bounding box. In this survey paper, we discuss some of the models that can achieve the task of instance segmentation and a dataset has been discussed. The goal of the paper is to give the reader an idea about the field of instance segmentation.**

***Index Terms*—Instance segmentation, MASK R-CNN, YOLACT++, Swin-L, QueryInst, DetectorRS, GCNet**

I. Introduction

Deep Learning is a field of computer vision derived from Machine learning. It is concerned with Artificial Neural Networks, which are algorithms that are inspired by the structure and function of a human brain. Artificial Intelligence is the technique that the computers use to mimic human behavior. Deep learning algorithms attempt to perform similar to how a human brain would, given a problem with a logical structure the algorithm tries to arrive at a conclusion similar to how a human brain would arrive on conclusions through analysis of given data. To make this possible, deep learning uses a multi-layered structure of algorithms. These structures are called neural networks.
Instance Segmentation is a deep learning based method of identifying each object instance for every known object within an image. Instance segmentation requires the correct detection of all objects in an image while also precisely segmenting each instance. Instance segmentation is done in two steps :

- **Object Detection** Object Detection - Detection of all objects in an image. In this step, classification of individual objects is done and then each object instance is localized using a bounding box.
- **Semantic Segmentation** - Each pixel in an image is fixed into a set of categories without differentiating object instances.

In this survey paper we are analysing the different instance segmentation models,and we are comparing the results of these models. The models being discussed here are Mask RCNN, YOLACT++, Swin-L, QueryInst, GCNet, DetectorRS.

II. Instance Segmentation Models

A. Swin-L

The Swin-L Transformer [12] first divides an input RGB picture into non overlapping boxes. Every box is considered as a "token". At the stage 1 of model, a box size of 4×4 was used. Then, we convert the raw-valued features extracted from the images and display it as an arbitrary dimension taken as C and we use transformer blocks with different self-attention values which are applied on these box tokens. We reduce the amount of tokens using box merging layers to increase the depth of network. The box merging layer in Stage 2 concatenates the features of every group of 2×2 neighboring boxes, and we apply a linear layer on the new 4C-dimensional concatenated features. Therefore at this stage, we decrease the amount of tokens by a factor of 4 (2 x downsampling of resolution), and therefore the output dimension is equal to 2C. We later use transformer blocks for transforming the features with resolution held at (H/ 8 x W /8). This first block of box merging and transformation is indicated as "Stage 2". The step of box merging and transformation is repeated as "Stage 3" and "Stage 4", with predetermined output resolutions of ($H/16 \times W/16$) for Stage 3 and ($H/32 \times W/32$) for Stage 4, respectively. These stages combined together will result in a hierarchical feature representation, having feature map resolutions similar to other convolution networks like ResNet and Inception. Hence, the proposed architecture claims to replace backbone networks in existing methods for numerous vision tasks. The Swin-L architecture is shown in figure 1.

B. YOLACT++

The paper introduced a branch used to re-score the predicted masks with respect to their mask IoU with ground-truth of the image. The Fast Mask Re-Scoring Network consists of a 6-layer FCN and a final global pooling layer. YOLACT takes cropped mask prediction as input and outputs the mask IoU for each object class of the model.The model then does recording of each mask.This is done by finding the product of predicted mask IoU for the predicted class and the confidence value of that class.

Addition of deformable convolution layers into the backbone network of YOLACT, leads to slight higher mAP with a few milliseconds overhead. DCN layers are used to reinforce ability of the network to monitor images with different scales, rotations, and aspect ratios. We use a flexible sampling strategy in the case of YOLACT model unlike the

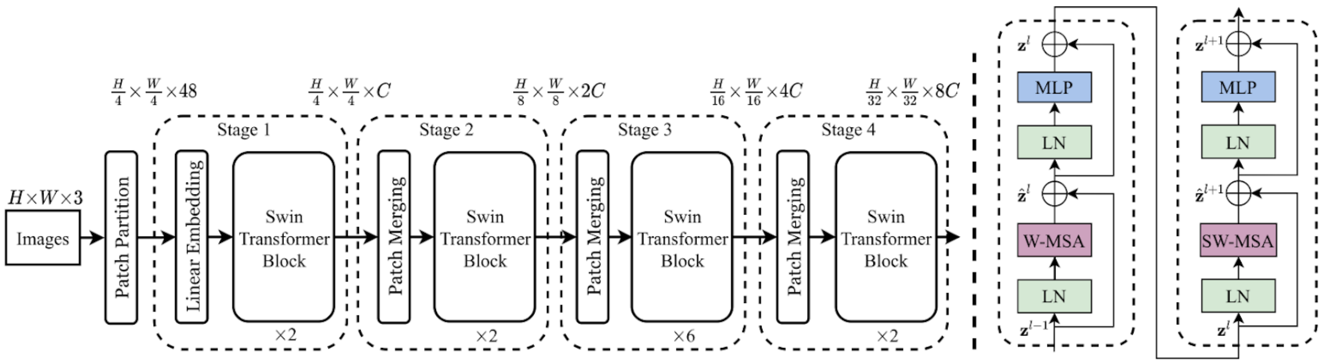

Fig. 1: Swin-L Architecture

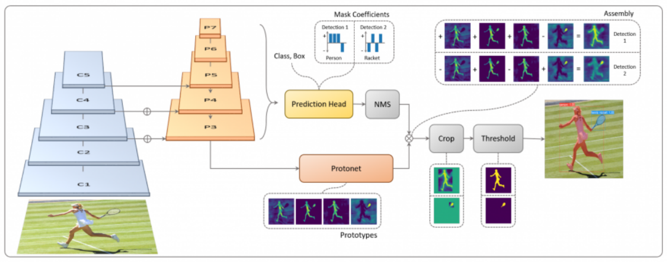

Fig. 2: YOLACT++ Architecture

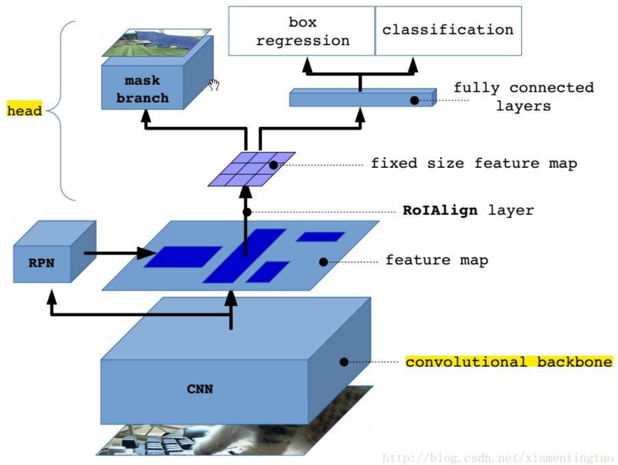

Fig. 3: Mask R-CNN Architecture

sampling strategy used for two-stage methods like Mask RCNN.

The YOLACT model uses an anchor-based backbone detector which is used to select the their scales and aspect ratios of a predetermined anchor. We use these anchors and contrast them with the anchor design of RetinaNet and RetinaMask. Multiscale anchors are used at every FPN level to ensure the YOLACT model gives us the best speed vs. performance trade off. The YOLACT++ architecture is shown in figure 2.

C. Mask R-CNN

Mask R-CNN [13] is a surprisingly simple, flexible, and fast system that can produce a state-of-the-art instance segmentation result. Mask R-CNN was the first to implement a branch for masking objects on each Region of Interest (RoI).

Mask R-CNN adopts the same two-stage procedure as Faster R-CNN for Instance segmentation, the first stage, called a region Proposal Network (RPN), proposes candidate objects for bounding boxes. The second stage extracts features using RoIPool/RoIAlign from each candidate box and performs classification and bounding-box regression and along with that Mask R-CNN also outputs a binary mask for each ROI.

An image is provided as the input to a convolutional neural network which provides a convolutional feature map. Instead of using a selective search algorithm on the feature map to identify the region of interest, a separate network is used to predict the region proposals. This network is called Region Proposal Network. The Regional proposal network divides the region of interest into multiple blocks of varying size and ratio and assigns an objectness score, i.e. blocks with very relevant data get high objectness score, the blocks with high objectness score and used first. Now these different sized blocks undergo either RoIAlign or RoIPool . The predicted region proposals are reshaped using a RoIAlign layer. In RoIAlign , image is first divided into different RoI bins of equal size and use bilinear interpolation to compute the exact values of the input features at four points in each RoI bin which are regularly accessed, and aggregate the result.
The Mask R-CNN architecture is shown in figure 3.

For masking Branch, each mask head contains a classifier to identify the class, a regressor to identify the x, y, w, h of the ROI, Fully Convolutional Network (FCN) to predict mask per class. Training has to be done in order for the algorithm to detect Mask and for that a dataset is required.

D. QueryInst

QueryInst [14] is Instances as Queries,it is a query based end-to-end instance segmentation method. QueryInst operates using with a six dynamic mask heads and query based object detector. These mask heads are driven by parallel supervision. It is built on Sparse R-CNN , which has six query stages. The object detection pipeline is depicted can be formulated as follows:

$$\mathrm{x}_t\mathrm{box} \longleftarrow \mathrm{P}^{\mathrm{box}}(\mathrm{x}^{\mathrm{FPN}}, \mathrm{b}_{t-1}) \tag{1}$$
$$\mathrm{q}_{t-1}{}^{*} \longleftarrow \mathrm{MSA}_t(\mathrm{q}_{t-1}), \tag{2}$$
$$\mathrm{x}_t{}^{\mathrm{box}*}, \mathrm{q}_t \longleftarrow \mathrm{DynConv}_t{}^{\mathrm{box}}(\mathrm{x}_t{}^{\mathrm{box}}, \mathrm{q}_{t-1}{}^{*}) \tag{3}$$
$$\mathrm{b}_t \longleftarrow \mathrm{B}_t(\mathrm{x}_t{}^{\mathrm{box}*}) \tag{4}$$

where $q\epsilon R^{NXd}$ denotes the object query. N and d denote the length (number) and dimension of query q. At stage t, a pooling operator P box extracts the current stage bounding box features x_t^{box} from FPN features x^{FPN} under the guidance of previous stage bounding box predictions b_{t-1}. At the same time a multi-head self-attention module MSA_t is applied to the input query q_{t-1} to get the transformed query q^*_{t-1} Then, a box dynamic convolution module $DynConv_t^{box}$ takes x_t^{box} and q^*_{t-1} as inputs and enhances the x_t^{box} by reading q^*_{t-1} while generating q_t for the next stage. Finally, the enhanced bounding box features x_t^{box*} are fed into the box prediction branch B_t for current bounding box prediction b_t.

For mask prediction head for query based instance segmentation dynamic mask heads driven by parallel supervision is used. The dynamic mask head at stage t consists of a dynamic mask convolution module $DynConv_t^{box}$ mask head M_t The mask generation pipeline is reformulated as follows:

$$\mathrm{x}_t^{\mathrm{mask}} \longleftarrow \mathrm{P}^{\mathrm{mask}}(\mathrm{x}^{\mathrm{FPN}}, \mathrm{b}^{t}) \tag{5}$$
$$\mathrm{x}_t{}^{\mathrm{mask}*} \longleftarrow \mathrm{DynConvt}^{\mathrm{mask}}(\mathrm{x}_t{}^{\mathrm{mask}}, \mathrm{q}_{t-1}{}^{*}) \tag{6}$$
$$\mathrm{m}_t \longleftarrow \mathrm{M}_t(\mathrm{x}_t{}^{\mathrm{mask}*}) \tag{7}$$

Dynamic mask heads in parallel with each other are set up, which transform each mask ROI feature x_t^{mask} in $DynConv_t^{mask}$ according to the corresponding query q^*_{t-1}, and are simultaneously trained in all stages. Inside $DynConv_t^{mask}$, the query acts as memory and is read by mask RoI features x_t^{mask} in the forward pass and written by x_t^{mask} in the backward pass. During training, the per-mask information not only flows back to mask RoI features x_t^{mask} , but also to the object query q^*_{t-1}, which is intrinsically one-to-one interlinked in different stages. Therefore the per-mask information flow is naturally established by leveraging the inherent properties of query based frameworks, with no additional connection needed.

After the training is completed, the information for mask prediction is stored in queries. During inference, all dynamic mask heads in 5 intermediate stages are discarded and only use the final stage predictions for inference. The queries implicitly carry the multi-stage information for mask prediction, which is read by mask RoI features x_t^{mask} in dynamic mask convolution $DynConv_t^{mask}$ at the last stage for final mask generation. Without $DynConv_t^{mask}$, the link between mask RoI features and the query is lost, and mask heads in different stages are isolated. Even though parallel supervision is applied to all mask heads, the information related to mask generation cannot flow into queries. In this condition, QueryInst degenerates to Cascade Mask R-CNN with a fixed number (i.e., N) of proposals across all stages.

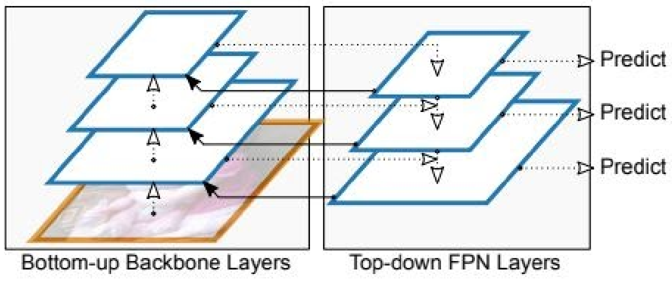

Fig. 4: Recursive Feature Pyramid

E. DetectorRS

Human visual perception detects objects by either enhancing or suppressing neuron activation by passing high-level semantic meaningful data through feedback connections. The technique of looking and thinking twice is derived from human system and is used in computer vision and has shown great results. Object detecting models like faster R-CNN are used to detect objects. Cascade R-CNN [5] is used to develop a multi-stage detector where the following detector heads are trained with more accurate examples.

The proposed Recursive Feature Pyramid (RFP) is built on top of the Feature Pyramid Networks (FPN) by installing additional feedback connections from the FPN layers into the bottom-up backbone layers. RFP is shown in figure 4. RFP repeatedly increases performance of FPN to produce powerful representations similar to the cascade R-CNN. To increase performance and speed up training, the feedback connections bring the features that directly receive gradients from the detector heads back to the low levels of the bottom-up backbone.

Switchable Atrous Convolution (SAC) convolves the same input feature with a variety of atrous rates and collects the outcome with the help of switch functions. SAC is shown in figure 5. These switch functions are spatially dependent. All the standard 3x3 convolutional layers in the bottom-up backbone are converted to SAC which increase the performance of the detector by a large margin. Some old techniques use conditional convolution which also binds results of a range of convolutions as a unique result. SAC does not require its architecture to be trained from scratch as it has a feature which can be used to convert pretrained standard CN easily.SAC uses a weight locking method which same weights for atrous convolutions trainable difference.

F. GCnet

The dependencies among pixels in an image is captured for better understanding the visual scene. There is a variety

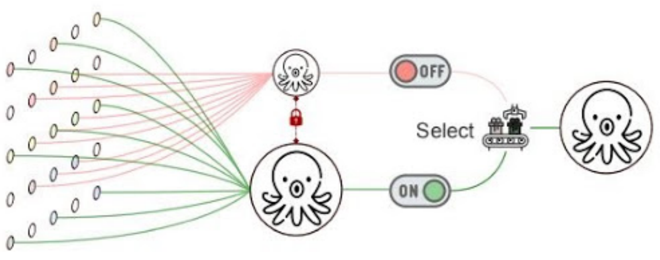

Fig. 5: Switchable Atrous Convolution

of applications like object detection, image classification and segmentation. In CNN, convolution layers are deeply stacked to model dependencies. Relationships between pixels are modeled by each layer within a local neighborhood. However, it is computationally inefficient for the direct repetition of convolutional layers and difficult to optimize, because of the difficulties in delivering messages between distant positions. To solve this problem, the non local network uses a layer to model long range dependencies, via a self attention mechanism. Initially, an attention map is created by computing a pairwise relations with the query position and the remaining positions with a non-local network. Then the characteristics of all locations are merged by a weighted sum with the weights described by the attention map. Attributes of every query position are added with the merged features to obtain the final result.

Query positions have the same attention maps, hence the learnt dependency is primarily query-independent. From this observation, the non-local block is ordered in which a query-independent attention map is distinctly used for all query's. The result is then obtained by the combination of attributes using the weights given in the attention map. It requires less computation for various object detection tasks as the block is highly accurate. The block design contains three steps :

- A module with models contexts by combining the features of all points jointly to develop a global context attribute.
- A module to show the channel-wise inter dependencies.
- A fusion module to combine the global context feature into features of all positions.

Furthermore, the number of parameters is reduced drastically by changing the single layer transformation function of the non-local block with a restriction of two layers, which results in the formation of a global context (GC) block. GC block requires less computation and due to this reason, it can be applied to all residual blocks in the Resnet architecture.

III. Dataset used

COCO is a large-scale dataset which is primarily used for object detection, segmentation, etc. COCO has several features:

- Super pixel stuff segmentation
- 330K images (more than 200K labelled)
- 1.5 million object instances
- 80 object categories
- 91 stuff categories
- 5 captions per image
- 250,000 people with key points

The COCO dataset is a labelled dataset and the dataset gives data to train segmentation models. These models are trained by detecting the common objects in the dataset and then use this knowledge for recognizing objects in real scenarios. The COCO dataset is used for evaluating the enhancement of these models as they are used.

IV. Summary

Table 1 shows a comparison between the models. They were compared based on the performance done after training on the COCO dataset.

Here AP^{bbox} denotes the average precision values of the bounding box of an object in an image predicted by the models, AP50 and AP75 is the average precision of the models only for candidates with 50% and 75% region on the predicted bounding box when compared to the ground truth, AP^{mask} denotes average precision of the mask head of the models.

Here Swin-L outperforms all other models in bounding box precision and masking precision but to achieve this precision the model has to be trained for a prolonged duration.Mask R-CNN has a lower training time but lack the precision similar to Swin-L. At the same time the models like DetectorRS have very satisfiable precision with an affordable training time.

	AP^{bbox}	AP50	AP75	AP^{mask}
Mask R-CNN	39.8	60	30	37.1
YOLACT++	32.3	53.8	36.9	34.6
QueryInst	56.1	74.2	53.8	49.1
GCNet	48.4	68.9	49.6	45.4
Swin-L	58.0	70.4	56.3	51.1
DetectorRS	51.3	72.0	53.3	48.5

TABLE I: Comparison Survey on Instance Segmentation Models

V. Future scope

Instance segmentation is a complex task As for the MS COCO dataset, the total average precision is nearly 50%, which means that there is still changes to be made which could improve the average precision. As an example, the model is trained on an 8-GPU machine where each GPU has 8 clips in a minibatch where the total minibatch size is 64 clips. The models are trained for a total of 400k iterations, with a learning rate of 0.01 and this is reduced 10 times for every consecutive 150k iterations. Tasks like instance segmentation are computationally expensive due to the hardware limitations. Speed optimization is a major concern for real-time instance segmentation. Detection of small objects is still a tedious task. The end-to-end based systems design and their training is still an issue. Human pose estimation and human parsing datasets have been made available.

VI. Conclusion

This survey paper compares various instance segmentation models. We have identified that Swin-L is the most accurate model in terms of the mAP value but it has a higher training time. Instance segmentation is a recent advancement in the field of computer vision and is still progressing. The pros and cons have been discussed. The popular COCO dataset has been used for our survey. The outcome of this survey is a comparison between the latest instance segmentation models. It has broadened our knowledge in instance segmentation. The models were able to detect the object and mask the object. All the models used in this survey were recently introduced. The models were analysed using mAP and bounding box precision values.

References

[1] Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Jianfeng Gao, Songhao Piao, Ming Zhou, et al. Unilmv2: Pseudo-masked language models for unified language model pre-training. In International Conference on Machine Learning, pages 642–652. PMLR, 2020.

[2] Josh Beal, Eric Kim, Eric Tzeng, Dong Huk Park, Andrew Zhai, and Dmitry Kislyuk. Toward transformer-based object detection. arXiv preprint arXiv:2012.09958, 2020.

[3] E. Xie, P. Sun, X. Song, W. Wang, X. Liu, D. Liang, C. Shen, and P. Luo, “PolarMask: Single Shot Instance Segmentation With Polar Representation,” 2020, pp. 12 193–12 202. [Online]. Available: https://openaccess.thecvf.com/content CVPR 2020/html/ Xie PolarMask Single Shot Instance Segmentation With Polar Representation CVPR 2020 paper.html

[4] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, “Yolact: Real-time instance segmentation,” in The IEEE International Conference on Computer Vision (ICCV), October 2019

[5] Zhaowei Cai and Nuno Vasconcelos. Cascade R-CNN: delving into high quality object detection. In CVPR, 2018.

[6] Ali Athar, S. Mahadevan, Aljosa Osep, L. Leal-Taixe, and B. Leibe. Stem-seg: Spatio-temporal embeddings for instance segmentation in videos. In ECCV, 2020.

[7] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In ECCV, 2018.

[8] Md Amirul Islam, Mrigank Rochan, Neil DB Bruce, and Yang Wang. Gated feedback refinement network for dense image labeling. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 3751–3759, 2017. 2

[9] Yukang Chen, Tong Yang, Xiangyu Zhang, Gaofeng Meng, Xinyu Xiao, and Jian Sun. Detnas: Backbone search for object detection. In Advances in Neural Information Processing Systems, 2019.

[10] H. Zhang, K. Dana, J. Shi, Z. Zhang, X. Wang, A. Tyagi, and A. Agrawal, “Context encoding for semantic segmentation,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 7151–7160. [5] X. Wang, R. Girsh

[11] C. Feichtenhofer, H. Fan, J. Malik, and K. He, “Slowfast networks for video recognition” arXiv preprint arXiv:1812.03982, 2018.

[12] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, Baining Guo ”Swin Transformer: Hierarchical Vision Transformer using Shifted Windows” arXiv:2103.14030 , 2021.

[13] Kaiming He, Georgia Gkioxari, Piotr Dollár, Ross Girshick ”Mask R-CNN” arXiv:1703.06870 , 2018.

[14] Yuxin Fang, Shusheng Yang, Xinggang Wang, Yu Li, Chen Fang, Ying Shan, Bin Feng, Wenyu Liu ”Instances as Queries” arXiv:2105.01928, 2021