BEVFormer^{++}: Improving BEVFormer for 3D Camera-only Object Detection: 1st Place Solution for Waymo Open Dataset Challenge 2022

Zhiqi Li[1,2*], **Hanming Deng**[1*], **Tianyu Li**[1*], **Yangyi Huang**[1*], **Chonghao Sima**[1*], **Xiangwei Geng**[1*], **Yulu Gao**[1*], **Wenhai Wang**[1*], **Yang Li**[1], **Lewei Lu**[1]

[1]Shanghai AI Laboratory [2]Nanjing University

Abstract

This report introduces the solution for Waymo Open Dataset Challenge 2022 from Shanghai AI Lab. Built upon our strong baseline, BEVFormer, the performance of our method is improved with several simple and yet effective techniques. These techniques include the adoption of several detector heads, LET-IoU based assignment/post processing, ensemble of 26 model results, etc. With the use of our methods, we achieve the 1st place on 3D Camera-only object detection track in Waymo Open Dataset Challenge 2022.

1. Introduction

The Waymo Open Dataset (WOD) Challenges are the largest and most challenging self-driving perception competition [18]. At CVPR 2022, WOD released a new competition for 3D detection using only camera-only input. In the camera-only 3D detection track, the challenge requires the algorithm to detect the 3D bounding boxes of objects with only camera data.

Our solution focuses on extracting image features into high quality 3D representation. While many camera based 3D object detection pipelines have been proposed in recent years [7, 10, 16, 21], we seek to build upon one baseline model that has two desired properties. First, accurate transformation from 2D image feature to 3D features. Second, general and application-friendly feature representation that can be a good test bed for different detection methods. Thus we choose BEVFormer [10], the state-of-the-art multi-camera recognition pipeline as a start point, and improve it for 3D camera-only detection task step by step.

2. Our Solution

In this section, we present the details of our model. We begin by introducing our baseline model BEVFormer and several main architecture improvements. Then we introduce the bells and whistles used to improve the baseline performance. Finally, we introduce the ensemble pipeline that produces our test results.

2.1. Baseline Detectors

We deploy the BEVFormer, which adopts a spatio-temporal transformer to generate BEV features from multi-view inputs, as a baseline 3D camera-only detector. As shown in Fig. 1, BEVFormer consists of three modules: backbone network, BEV encoder and detection head. Since BEVFormer is used to generate BEV features, the design of BEV-based detection heads is perpendicular to the BEVFormer.

BEVFormer provides high quality BEV features, which allows us to explore the design of different detection heads in both image and lidar-based 3D detection. In this challenge, We use three different detection heads to combine with BEVFormer into three different detection models. These three heads covers three main categories of detector design, including anchor-free, anchor-based and center-based. We choose different types of detector heads that differ as much as possible in design, so as to fully leverage different detection frameworks for their potential in different scenarios, because we think these different heads facilitate the final ensemble results.

Deformable DETR head Original BEVFormer uses a modified Deformable DETR decoder as its 3D detector [1, 10, 24], which can detect 3D bounding boxes end-to-end without NMS. For this head, we follow the original design but use Smooth L1 loss to replace the origin L1 loss.

Freeachor head We also adopt the FreeAnchor [23] as our 3D detector, which is an anchor-based detector that can automatically learns the matching of anchors. We compute the LET-IoU between prediction results and ground truth.

Centerpoint head The Centerpoint [22] head is the last detector head we utilize, which is a powerful center-based anchor-free 3D detector.

*These authors contributed equally to this work.

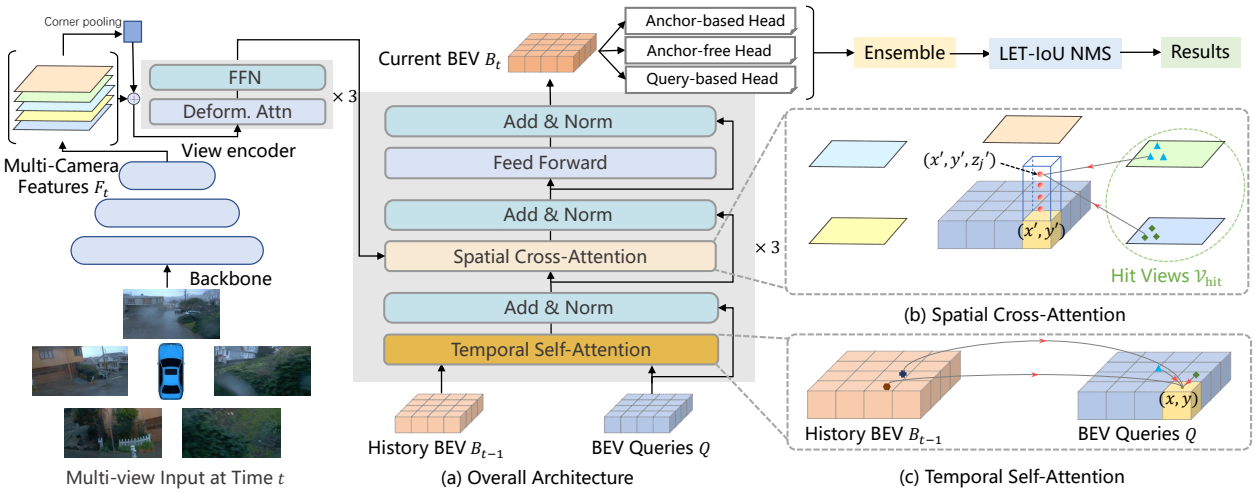

Figure 1: **Overall architecture of BEVFormer.** (a) The encoder layer of BEVFormer contains grid-shaped BEV queries, temporal self-attention, and spatial cross-attention. (b) In spatial cross-attention, each BEV query only interacts with image features in the regions of interest. (c) In temporal self-attention, each BEV query interacts with two features: the BEV queries at the current timestamp and the BEV features at the previous timestamp. We also add corner pooling and view encoder to refine view feautres better.

2.2. Improving BEVFormer

Deformable DETR view encoder. In the original BEVFormer design, there is no separate encoder module for view features, which is a potential shortcoming of the model. So we use the encoder of Deformable DETR to refine the multi-scale view features that output from FPN and then feed these refined view features into the BEV encoder [24]. To save GPU memory, we only use 3 encoder layers.

Conv offset in Temporal Self Attention (TSA). Original BEVFormer design uses TSA to construct association between the BEV features of the same objects at different time, especially moving objects. Since the resolution of BEV features is much smaller than the object size, we use one convolutions layer to predict TSA offsets instead of linear layers to obtain a larger receptive field on BEV features. The kernel size, stride and padding of this convolution layer are 3, 1, 1.

Corner Pooling. To enlarge the receptive field of feature sampling in the Camera-BEV feature transformation in BEVFormer's Camera-BEV cross attention, we introduce corner pooling on the output of FPN feature [9]. Corner pooling aggregates features along horizontal and vertical direction in the camera feature map. WE add pooling features with shape $1 \times W$ and $H \times 1$ to original features with broadcasting.

LET-IoU based Assignment. While LET-IoU [8] is used in NMS, we also find that LET-IoU based assignment is more friendly to camera-based 3D object detection under LET-IoU metrics. We apply LET-IoU in the assignment of Free anchor head we used. This changes the assigned anchors to be distributed along radial direction from the camera center, resulting better alignment of visual features when projected from BEV to camera coordinates.

LET-IoU based NMS. Since this challenge proposes a new metric LET-AP [8] which uses LET-IoU to match the ground truth and prediction results, we also design a NMS based on LET-IoU to remove redundant results. Actually, this design is more suitable for Camera-only 3D-detectors. Since the 3D IoU of two mutually redundant results may be small, this leads to failing to remove many false positive results. With LET-IoU, redundant results tend to have higher IoU, thus can be more thoroughly removed.

2D detection auxiliary loss. We apply an additional 2d detection head on image features after FPN and train it with projected 2D bounding boxes. The signal goes directly to 2D image features and provides better image feature supervision. The 2D detector we use is FCOS [19]. The relative weight between 2D and 3D detector loss is learnable.

Global position regression. We apply additional loss on regression of global position for Defornbale DETR head for better localization performance since its object queries are not location-specific. The global position is a tuple of $(X, Y, \sqrt{X^2 + Y^2})$, where X, Y are the coordinates.

EMA. Following YOLOX [5], we use Exponential Moving Average (EMA) weights updating. EMA copies a backup of model weights and the EMA weights are updated with a sliding average. After the training phase, the model weights are replaced by EMA weights for prediction.

Multi-scale and Flip Training. Mutli-scale and flip training are the most simple and effective way to improve the performance of the model. In this work, the input image is scaled by a factor between 0.5 and 1.2 and flipped by a ratio

of 0.5. Since BEVFormer uses sequence input, we ensure that the transformations are consistent for each frame of the sequence.

2.3. Expert Model

Since the Waymo Open Dataset has a highly imbalanced data distribution [18], we follow previous works on the 2D Detection track [2, 4] to train multiple expert models on re-sampled subsets. WOD contains only 81k cyclist annotations while having a much larger number of vehicles(9.0M) and pedestrians(2.7M). We sampled a class rebalanced subset from the training set with the most frequent annotations of each category by the ratio of 1:1:1 for cyclists, pedestrians, and vehicles. Besides, according to context information about the time of the day provided in the scenes, we also sampled a context time rebalanced subset by the ratio of 1:1:2 for nighttime, dawn/dusk, and daytime scenes. We fine-tuned the aforementioned detection models on each subset to get the final expert models.

2.4. Ensemble

In the ensemble phase, we use the improved version of the weighted boxes fusion (WBF) [17]. Inspired by Adj-NMS [12], we add matrix NMS [20] after the original fusion to filter out redundant boxes. In order to introduce the multi-scale and flip results, we use a two-step ensemble procedure. In step 1, we use WBF to integrate the prediction results from multi-scale to generate flip and noflip results for each model. In step 2, we put all model results together and adopt WBF to get the final results. Considering the diversity of the performance of each model, the parameter adjustment is much complex, so the evolution algorithm is used to search the WBF parameters and model weights. We use Evolution in NNI [15] to automatic search parameters, where the population size is 100. The search process is based on the performance of the 3000 validation images, and different classes are searched separately.

3. Experiments

3.1. Dataset and Evaluation

Dataset. The Waymo Open Dataset v1.3 [18] contains 798, 202 and 80 video sequences in the training, validation, and testing sets, respectively. Each sequence has 5 views of side left, front left, front, front right, and side right, and the image resolution is 1920×1280 pixels or 1920×886 pixels. Due to limited computational resources, we sample 1 frame of every 5 frames from the training set to form a *mini-train* set to quickly verify the effect of different implementations [18].

3.2. Implementation Details

BEVFormer. By default, we utilize the output multi-scale features from FPN [11] with sizes of 1/16, 1/32, 1/64 and the dimension of $C = 256$. We use 3 layers instead of 6 to save GPU memory. For WOD, the default spatial shape of BEV queries is 300×220 for mini-train and 450×330 for the final models, the perception ranges are [-35.0m, 75.0m] for the X−axis and [-75.0m, 75.0m] for the Y−axis. During training, we use a temporal sequence that consists of 4 frames, the time interval between frames is about 0.5s. During inference phase, we convert the 10Hz video into five 2Hz sub-videos for temporal inference, and the i-th sub-video only contains frames with the frame index are multiple of i. For the samples that without enough history, we pad the sequence with the the first valid sample. Other settings are the same to original BEVFormer.

Deformable DETR Detector. Following BEVFormer and Deformable DETR, this head uses a single-scale BEV features as the input of the decoder, predicting 3D bounding boxes and velocity rather than 2D bounding boxes, and only using smooth L1 loss to supervise 3D bounding box regression. With the detection head, our model can end-to-end predict 3D bounding boxes and velocity without the NMS post-processing.

Centerpoint Detector. Centerpoint head takes single-scale BEV features as input, and uses a separate DCN branch to handle each category. We use gaussian Focal Loss and smooth L1 loss to supervise classification and bounding box regression, and the corresponding loss weights are 1 and 0.5.

FreeAnchor Detector. FreeAnchor head also uses a single-scale BEV features as input. We set anchor size $2.08 \times 4.73 \times 1.77$, $0.84 \times 1.81 \times 1.77$, $0.84, 0.91, 1.74$ for car, cyclist and pedestrian respectively. The unit of anchor size is meter. Box IoU threshold of 0.5 is used in assignment for enlarged 2D boxes under BEV. The direction classification loss is also adopted.

Training Strategy. Following BEVFormer, by default, models are trained with AdamW [14] optimizer and 12 epochs, a learning rate of 2×10^{-4}, a weight decay of 0.01, a total batch of 8 on 8 NVIDIA A100 GPUs. We use cosine annealing to schedule the learning rate. We use R101-DCN [3, 6] as backbone during the exploration stage and finally use Swin-Large [13] to build stronger models. No external data are used in the final results.

3.3. Ablations

Improvements over BEVFormer baseline. The results of applying techniques introduced in Section 2.2 on all three detection heads we've used are shown in Tab. 1.

Expert models. The performance of expert models are shown in Tab. 2. The expert models fine-tuned on the class rebalanced subsets improved the performance of Deformable DETR on cars and pedestrians by 0.9% and 0.5%, and 0.4%, 1.1% for Centerpoint on cars and cyclists, respectively. And the time-rebalanced expert models brought an

ID	DeD	FrA	CeP	CovO	DE	CoP	DA	GR	MS	EMA	LE	Backbone	DS	LET-mAPL	LET-mAP	LET-mAPH
0	✓											R101	mini	34.6	50.2	46.1
1	✓			✓								R101	mini	35.9	51.8	48.1
2	✓				✓							R101	mini	36.1	52.2	48.1
3	✓					✓						R101	mini	35.6	51.1	46.9
4	✓						✓					R101	mini	36.2	52.2	48.1
5	✓							✓				R101	mini	35.4	50.9	47.2
6	✓								✓			R101	mini	35.4	51.1	47.1
7	✓									✓		R101	mini	34.9	50.3	46.3
8*	✓											SwinL	mini	40.0	55.6	51.9
9*	✓			✓	✓	✓	✓	✓	✓	✓		SwinL	mini	44.7	60.8	55.5
10		✓										R101	mini	35.9	49.9	45.9
11		✓									✓	R101	mini	36.3	51.1	46.6
12			✓									R101	mini	34.0	47.9	43.5
13	✓			✓	✓	✓	✓	✓	✓	✓		SwinL	full	48.4	64.8	60.4
14		✓		✓	✓	✓	✓	✓	✓	✓		SwinL	full	47.2	61.2	56.8
15		✓		✓	✓	✓	✓	✓	✓	✓	✓	SwinL	full	47.6	61.4	57.0
16			✓	✓	✓	✓	✓	✓	✓	✓		SwinL	full	41.9	54.6	48.2

Table 1: Ablation studies on `val` set with our improvements on BEVFormer. DeD (Deformable Detr head). FrA (FreeAnchor head). CeP (Centerpoint head). ConvO (Conv offsets in TSA). DE (Deformable view Encoder). CoP (Corner Pooling). LE (LET-IoU based Assignment). DA (2D Auxiliary loss). GR (Global location regression). DS (Dataset). The mini dataset contains 1/5 training data. * notes we train the model with 24 epochs.

ID	Head	LET-mAPL						
		Overall	Car	Cyc	Ped	Day	Night	D/D
13	DeD	48.4	60.4	37.0	47.8	49.2	37.0	49.5
13.1	expert 1	48.9	61.3	37.1	48.3			
13.2	expert 2	48.7				49.5	36.2	49.9
14	FrA	47.2	62.7	36.7	42.1	48.4	33.4	46.0
14.1	expert 1	47.2	62.8	36.6	42.3			
14.2	expert 2	47.0				48.5	33.3	46.3
16	CeP	41.9	56.8	29.7	39.3	42.8	31.9	42.2
16.1	expert 1	42.4	57.2	30.8	39.2			
16.2	expert 2	42.3				43.2	31.8	42.1

Table 2: Performance of expert models. The overall LET-mAPL metrics and that on each category or each time-of-day subset of the models are listed in the table. Expert 1 is class rebalanced and Expert 2 is time rebalanced. D/D is short for dawn/dusk.

increase of 0.3% and 0.4% for Deformable DETR and Centerpoint in the overall performance. The strengths of expert models were further utilized in our final solution by model ensembling.

Ensemble. Tab. 3 presents the ablation results of ensemble. As shown in the 2nd and 3rd row, we can get better results by combining different models in the ensemble process. The 4th row is the result of using the search algorithm to ensemble 20 model results, which obtains a very large gain, indicating that our ensemble process is very effective. These models results consist of 18 nolfip and flip testing results of 9 models in Tab. 2 and 2 results of ID 15 model in the Tab. 1. Based on the ensemble results, we continue to introduce two-step ensemble and let-iou based NMS. From 5th row and 6th row, we can see that those two tricks have brought about an increase of 0.7% and 1.2% respectively.

Head	LET-mAPL	LET-mAP	LET-mAPH
FrA	47.6	61.4	57.0
+DeD	49.1	65.2	60.2
+CeP	50.6	66.5	61.2
20 Search	53.2	69.3	64.9
+Two-Step	53.9	69.2	64.5
+L-NMS	55.1	71.1	66.2

Table 3: Performance of ensemble model on the validation set. Two-Step (Two-step ensemble). L-NMS (LET-IoU based NMS)

Head	Backbone	Params.	FLOPs	FPS
D-Detr	R101	102M	2469G	2.8
D-Detr	Swin-L	333M	8864G	1.2

Table 4: FPS is measured on A100 GPU. The shape of image is 1920×1280. The shape of BEV queries is 450×330.

Method	LET-mAPL	LET-mAP	LET-mAPH
WaymoBaseline	14.77	22.61	18.17
Our Solution	56.16	70.69	65.93

Table 5: Our final results.

4. Final Results

We show the FLOPs and FPS of our final model in Tab. 4. For our final results, apart from the pipeline we introduce in Sec. 3.3, we add another three models corresponding to ID 13, 14, 16 in Tab. 1, and train them on both training and validation splits. Thus we use ensemble of $20 + 3 \times 2$ inference results and the final results are shown in Tab. 5.

References

[1] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: European conference on computer vision. pp. 213–229. Springer (2020)

[2] Chen, S., Wang, Y., Huang, L., Ge, R., Hu, Y., Ding, Z., Liao, J.: 2nd place solution for waymo open dataset challenge–2d object detection. arXiv preprint arXiv:2006.15507 (2020)

[3] Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., Wei, Y.: Deformable convolutional networks. In: Proceedings of the IEEE international conference on computer vision. pp. 764–773 (2017)

[4] Ding, Z., Hu, Y., Ge, R., Huang, L., Chen, S., Wang, Y., Liao, J.: 1st place solution for waymo open dataset challenge–3d detection and domain adaptation. arXiv preprint arXiv:2006.15505 (2020)

[5] Ge, Z., Liu, S., Wang, F., Li, Z., Sun, J.: Yolox: Exceeding yolo series in 2021. arXiv preprint arXiv:2107.08430 (2021)

[6] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)

[7] Huang, J., Huang, G., Zhu, Z., Du, D.: Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. arXiv preprint arXiv:2112.11790 (2021)

[8] Hung, W.C., Kretzschmar, H., Casser, V., Hwang, J.J., Anguelov, D.: Let-3d-ap: Longitudinal error tolerant 3d average precision for camera-only 3d detection. arXiv preprint arXiv:2206.07705 (2022)

[9] Law, H., Deng, J.: Cornernet: Detecting objects as paired keypoints. In: Proceedings of the European conference on computer vision (ECCV). pp. 734–750 (2018)

[10] Li, Z., Wang, W., Li, H., Xie, E., Sima, C., Lu, T., Yu, Q., Dai, J.: Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. arXiv preprint arXiv:2203.17270 (2022)

[11] Lin, T.Y., Dollár, P., Girshick, R.B., He, K., Hariharan, B., Belongie, S.J.: Feature pyramid networks for object detection. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 936–944 (2017)

[12] Liu, Y., Song, G., Zang, Y., Gao, Y., Xie, E., Yan, J., Loy, C.C., Wang, X.: 1st place solutions for openimage2019–object detection and instance segmentation. arXiv preprint arXiv:2003.07557 (2020)

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

[14] Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: ICLR (2019)

[15] NNI: https://github.com/microsoft/nni

[16] Philion, J., Fidler, S.: Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In: European Conference on Computer Vision. pp. 194–210. Springer (2020)

[17] Solovyev, R., Wang, W., Gabruseva, T.: Weighted boxes fusion: Ensembling boxes from different object detection models. Image and Vision Computing **107**, 104117 (2021)

[18] Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., et al.: Scalability in perception for autonomous driving: Waymo open dataset. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 2446–2454 (2020)

[19] Tian, Z., Shen, C., Chen, H., He, T.: Fcos: Fully convolutional one-stage object detection. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 9627–9636 (2019)

[20] Wang, X., Zhang, R., Kong, T., Li, L., Shen, C.: Solov2: Dynamic and fast instance segmentation. Advances in Neural information processing systems **33**, 17721–17732 (2020)

[21] Wang, Y., Guizilini, V.C., Zhang, T., Wang, Y., Zhao, H., Solomon, J.: Detr3d: 3d object detection from multi-view images via 3d-to-2d queries. In: Conference on Robot Learning. pp. 180–191. PMLR (2022)

[22] Yin, T., Zhou, X., Krahenbuhl, P.: Center-based 3d object detection and tracking. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11784–11793 (2021)

[23] Zhang, X., Wan, F., Liu, C., Ji, R., Ye, Q.: Freeanchor: Learning to match anchors for visual object detection. Advances in neural information processing systems **32** (2019)

[24] Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J.: Deformable detr: Deformable transformers for end-to-end object detection. In: International Conference on Learning Representations (2020)