Plug and Play: A Representation Enhanced Domain Adapter for Collaborative Perception

Tianyou Luo, Quan Yuan[†], Guiyang Luo, Yuchen Xia, Yujia Yang, Jinglin Li

Beijing University of Posts and Telecommunications
{lty349,yuanquan,luoguiyang,yuchen.xia,yangyj2022,jlli}@bupt.edu.cn
[†] Corresponding Author

Abstract. Sharing intermediate neural features enables agents to effectively see through occlusions. Due to agent diversity, some pioneering works have studied domain adaption for heterogeneous neural features. Nevertheless, these works all partially replace agents' private neural network with newly trained components, which breaks the model integrity and bidirectional compatibility of agents. In this paper, we consider an open challenge: *how to learn non-destructive domain adapters for heterogeneous legacy models to achieve collaborative perception, while compatible with continually emerging new agent models?* To overcome this challenge, we propose the first plug-and-play domain adapter (PnPDA) for heterogeneous collaborative perception. PnPDA builds a semantic calibrator based on contrastive learning to supervise domain gap bridging without destructing the original models. Semantic converter is learned to transform the semantic space of features, while semantic enhancer is utilized to enhance the representation of features. By specifying standard semantics, new models with PnPDA can easily join existing collaborations. Extensive experiments on OPV2V dataset show that PnPDA non-destructively bridges the domain gap and outperforms SOTA by 9.13%. The code is available at: https://github.com/luotianyou349/PnPDA.

Keywords: Heterogeneous collaborative perception · Domain adapter · Contrastive learning

1 Introduction

Multi-agent collaborative perception significantly enhances autonomous vehicles' comprehension of complex road environments [1]. Recent studies primarily employ feature fusion [5,14,31,32] to transmit valuable intermediate features between autonomous vehicles with low communication overhead, while improving perception precision. Nevertheless, these studies assume that the agent models as well as their intermediate features are homogeneous.

There exist various agent models in the real world, leading to significant domain gap between intermediate features generated by different model architectures and parameters. Utilizing these unintelligible features directly for cooperative perception can cause performance degradation by feature misinterpretation.

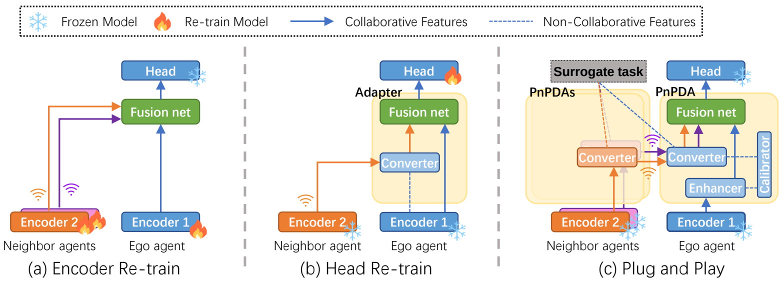


Fig. 1: The differences between heterogeneous cooperative models. (a) re-trains new encoders for all agents; (b) introduces a task-specific converter, which consequently necessitates re-training the task detection head of the ego agent; (c) our PnPDA does not require any re-training of encoders or heads, utilizes task-agnostic converters for non-destructive semantic transformation and employs two-step transformation to reduce the cost of bidirectional compatibility.

Some pioneering works (*e.g.*, HEAL [20], MPDA [30]) have emerged to address this issue. However, as shown in Fig. 1 (a) and (b), these methods either re-train the encoder of neighbor agents [20] or re-train the detection head of the ego agent [30], which break the integrity of the original agent models. If the integrity of the model is compromised, the heterogeneous cooperative perception cannot achieve backward compatibility with legacy immutable agent models. Besides, if there comes various future agent models, it requires training multiple adapters accordingly. This not only necessitates agents to maintain a large number of adapters but also increases the training cost for new models to join the collaborative network. Thus, there is an open challenge facing the heterogeneous cooperative perception: *how to learn non-destructive domain adapters for heterogeneous legacy models to achieve collaborative perception, while compatible with continually emerging new agent models?*

To address this challenge, we propose a novel plug-and-play domain adapter (PnPDA), as shown in Fig. 1 (c), empowering agents with various legacy and newly emerging models to cooperate without destructing the original models. PnPDA primarily consists of a semantic converter, a semantic enhancer, a fusion network, and a semantic calibrator used to supervise the operation of converter and enhancer. When ego agent needs to collaborate with heterogeneous features, the semantic converter transforms the received heterogeneous features into the ego semantic space, while the semantic enhancer enhances the representational capacity of the ego features. Subsequently, the fusion network processes these features with the same semantics. Because the original semantics are preserved, the fused features can be directly fed into ego's original detection head. To further reduce the overhead of bidirectional compatibility, we train a standard semantics and design a new collaborative perception process based on this standard semantics. The newly emerging agents only need to train their own converters independently, allowing them to join the collaborative network through two semantic transformations. Specifically, neighbor agents are responsible for converting their own semantics to standard semantics, while the ego agent is responsible for converting received standard semantics to ego semantics. As a side benefit of plug and play, agents using PnPDA can seamlessly switch between collaborative perception and independent perception modes.

Experiments conducted on a cooperative perception dataset OPV2V [33], demonstrate that our model achieves an average performance improvement of 9.13% and 6.55% in terms of average precision (AP) at Intersection-over-Union (IoU) thresholds of 0.5 and 0.7, respectively, compared to state-of-the-art domain adaptation methods. In summary, our contributions are as follows:

1. We propose the first plug-and-play domain adapter to address the domain gap among heterogeneous features, training converter and enhancer using knowledge distillation techniques to achieve semantic transformation.
2. We design a semantic calibrator based on contrastive learning, using task-agnostic surrogate task supervision for domain adaptation during the pre-training phase, without the need to modify the original models.
3. We design a novel collaborative perception process to ensure the bidirectional compatibility of the model. Legacy models and newly emerging models can easily join the collaborative network through semantic transformation.

2 Related Work

Cooperative perception can be categorized based on the type of collaborative data among agents, including early fusion using raw point cloud data [1], feature fusion [5, 9, 14, 17, 19, 21, 22, 31, 35, 37] using intermediate features, especially those generated by point cloud data, and last fusion [16, 24, 28] using the final detection results. Feature fusion is currently a focal point of research due to its balanced performance in terms of data transmission volume and detection accuracy. F-cooper [5] employs the maxout algorithm for feature fusion. BEVFormer [18] aggregates BEV features from multiple views and utilizes BEV features as queries for spatial and temporal interactions. CoBEVT [31] first converts image data to BEV data and then uses axial attention for feature fusion. To reduce communication costs, Where2comm [14] determines the required features for transmitting ego agent by constructing a spatial confidence map instead of global features. Yang *et al.* [35] utilize attention fusion method and historical feature information to assist perception. DiscoNet [17] employs an early fusion network as a teacher model to guide the training of the feature fusion network acting as the student model.

Domain adaptation is primarily used when there is inconsistency in the semantic space of the features that require communication. V2X-ViT [32] addresses data noise introduced by differences in sampling frequency between agents and roadside sensors by constructing a directed graph. HM-ViT [26] primarily addresses the collaborative heterogeneity between point cloud data and image data, and it can collaborate with any number of agents. TransIFF [8] treats agent-road data with different perspectives as objects requiring cross-domain adaptation and uses a cross-domain adaptation module to fuse target candidate boxes from different perspectives. MPDA [30], as a pioneer work in addressing semantic heterogeneity, uses heterogeneous features of neighbor agents as queries for attention computation to achieve cross-domain adaptation. HEAL [20] addresses the pain point of lacking forward scalability in previous works by designing a set of standard semantics. Features from different sensors are integrated into the existing collaborative network by training new encoders.

Contrastive learning forms the cornerstone of our approach. It aims to establish a fundamental representation by minimizing the semantic similarity between positive pairs and maximizing it between negative pairs, facilitating adaptation to various downstream tasks. Approaches such as [6,7,12,25] focus on learning features for the same object from diverse perspectives and distinguishing features of distinct objects to bolster the model's representation capabilities. BYOL [11] revolutionizes the paradigm by transforming it from a classification problem into a prediction problem. PointContrast [27] extends the realm of contrastive learning to point cloud data, achieving point-wise contrastive learning, albeit within the confines of indoor point cloud scenes. ProposalContrast [36] pioneers the application of contrastive learning to the autonomous driving domain, constructing positive and negative sample pairs at the object level. TARL [23] employs point cloud aggregation to establish positive and negative sample pairs by aggregating data from multiple time frames.

3 Approach

In this paper, we consider that the transfer of features during cooperative perception suffers from a huge domain gap due to model diversity. To address this issue, we propose PnPDA based on contrastive learning, which adapts the heterogeneous features received from neighbor agents for domain alignment. Fig. 2 illustrates the overview of PnPDA during the pre-training and fine-tuning stages. During pre-training stage, PnPDA comprises three main modules: semantic converters, semantic enhancer, semantic calibrator. In fine-tuning stage, the semantic calibrator is discarded, and the original detection head of ego agent is used.

3.1 Overview

Semantic converter and enhancer. Drawing inspiration from knowledge distillation [13] and DiscoNet [17], By specifying the teacher model and the student model, the converter can transform the semantics of student features into the semantic space of the teacher model, while the enhancer can enhance the representational capacity of the teacher model, as detailed in Sec. 3.2.

Semantic calibrator. Inspired by DETR [3], using queries to predict the bounding boxes of targets, we treat the enchanced feature map of the teacher model as a set of query vectors and the transformed feature map of student model as key-value pairs, fed into the calibrator. The calibrator updates student features by computing inter-domain attention to further predict the semantics of

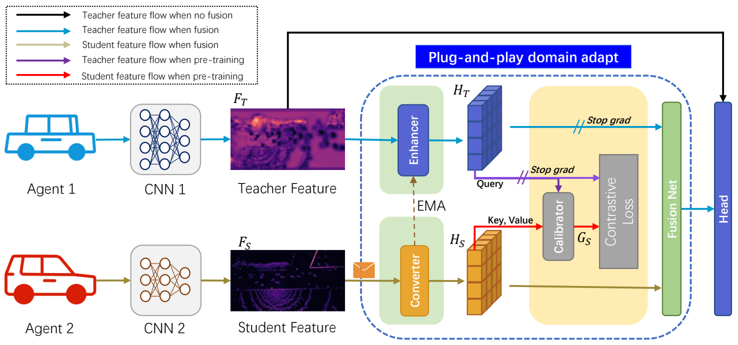

Teacher feature flow when no fusion
Teacher feature flow when fusion
Student feature flow when fusion
Teacher feature flow when pre-training
Student feature flow when pre-training
Plug-and-play domain adapt
Agent 1
CNN 1
F_T
Teacher Feature
Agent 2
CNN 2
F_S
Student Feature
Enhancer
EMA
Converter
H_T
H_S
Query
Key, Value
Stop grad
Stop grad
Calibrator
G_S
Contrastive Loss
Fusion Net
Head

Fig. 2: Overview of PnPDA. Heterogeneous features undergo domain conversion with PnPDA before being input into the fusion network. We use the original detection heads for object detection. When the ego agent does not require collaborative perception, it can bypass PnPDA and directly input ego features into its detection head.

teacher model. Then, we extract pillar-wise features corresponding to the same object in both feature maps as positive pairs and those corresponding to different objects as negative pairs to compute its contrastive loss.

Process of collaborative perception. By training a standard semantics, newly emerging models can join the collaborative network through two semantic transformations. The new models train converters to transform to standard semantics, while the ego model trains a converter from standard semantics to ego semantics. Notable, the models' structures and training methods for these two converters are consistent, differing only in the selection of input features to achieve different semantic transformation effects. In the following sections, except for special mention, we will mainly focus on the training process from the standard semantic space to the ego semantic space.

3.2 Semantic Converter and Enhancer

The input of PnPDA consists of two heterogeneous feature maps. Teacher features $F_T \in \mathbb{R}^{H\times W\times C}$ and student features $F_S \in \mathbb{R}^{H'\times W'\times C'}$, where H is height, W is width, and C is the number of channels. These feature maps are generated by point cloud encoders with different architectures. They exhibit inconsistencies not only in semantic space but also in terms of feature map dimensions and granularity due to differences in the voxel size of the feature encoders [30], *i.e.*, $H \neq H', W \neq W', C \neq C'$. Therefore, upon receiving student features, the first step is to adjust the size and channel count of the student feature map using a simple reshaping operation Γ :

$$F_S^{'} = \Gamma(F_S) \in \mathbb{R}^{H\times W\times C}. \tag{1}$$

Precisely, we train a max-pooling layer and a 1×1 convolution layer to unify the spatial dimensions and channel counts. This achieves the standardization of the granularity of the two feature maps for subsequent operations.

After reshaping the heterogeneous features, the teacher model is used to guide the transformation process of the student model. The student features undergo semantic transformation using a converter Φ_S. To better enable the student model to understand the experiences of the teacher model, we designed an enhancer Θ_T for the teacher model. Intra-domain attention is employed to semantically project features, learn their fundamental representations, and align the semantics of both features into a unified semantic space. It computes both local and global attention mechanisms. Applying global attention computation to teacher features allows for the acquisition of ample scene information in regions that may be obscured on the original feature map F_T due to occlusions and other issues. This enhances our ability to embed queries or fuse during perception in the next steps.

Specifically, Θ_T will directly utilize the intermediate features F_T generated by the teacher encoder. On the other hand, after processing the student features F_S through Γ, we feed $F_S^{'}$ into Φ_S. As shown in Fig. 3 (a), the architectures of the converter and enhancer are consistent in the semantic projection part. As a result, the projection operations for both types of features are entirely identical:

$$H_S = \Phi_S(F_S^{'}) \in \mathbb{R}^{H\times W\times C}, \quad H_T = \Theta_T(F_T) \in \mathbb{R}^{H\times W\times C}. \tag{2}$$

Then, we use gradient backpropagation to update Φ_S, and Φ_S dynamically updates its parameters based on the training loss, learning a transformation function to project into the semantic space of teacher semantics. Θ_T utilizes exponential moving average (EMA) technique to indirectly update its parameters based on feedback from Φ_S. With EMA, the teacher model's parameters undergo exponential moving average at each parameter update, preserving the original semantics of teacher features as much as possible, providing the student model with more consistent and reliable knowledge. This approach helps enhance the generalization performance of the student model, enabling it to mitigate instability during the training process, thus avoiding issues like model collapse in the pre-training phase [4].

3.3 Semantic Calibrator

Each pillar-wise feature in the BEV feature map represents spatial information for a specific region in the real world [18]. Furthermore, we regard the BEV feature map of the teacher model as a set of BEV query embeddings $Q \in \mathbb{R}^{H\times W\times C}$. Specifically, the query embedding $Q_i \in \mathbb{R}^{1\times C}$ located at $i = (x, y)$ can be seen as a candidate box querying the corresponding region in the BEV plane.

During pre-training, we employ a calibrator by computing inter-domain attention, with teacher enhanced features H_T as query embeddings and student transformed features H_S as key-values, as shown in Fig. 3 (a). Following common practices, we add 2D absolute position encoding to each query embedding, where spatial position encoding helps ensure the uniqueness of each query embedding in the searched region. Simultaneously, we use the same position encoding for

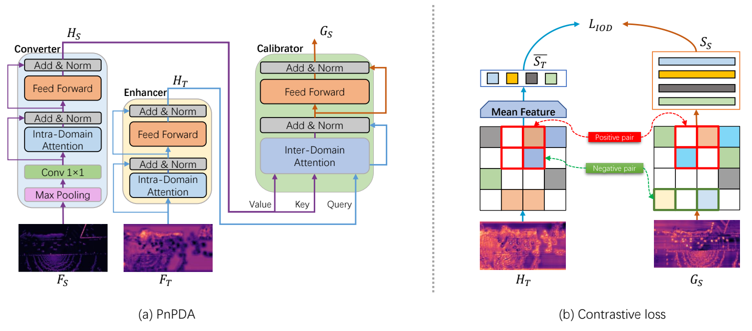


Fig. 3: PnPDA architecture. (a) The architecture of PnPDA. (b) The pipeline of contrastive loss

keys, ensuring consistency in the spatial representation of queries and keys, aiding the model in better learning semantic information for the same position. By computing inter-domain attention, we can establish direct connections between teacher features and student features. The $P_i \in \mathbb{R}^{1\times C}$ for the new feature map $H_S^{'}$ is obtained by querying the representation of H_S for corresponding Q_i at same position. Inter-domain attention is calculated as follows:

$$P_i = \text{Softmax}(\frac{Q_i H_S^{\top}}{\sqrt{d_{H_S}}})H_S, \tag{3}$$

where d_{H_S} represents the dimension size of H_S. Subsequently, we use Ψ to concatenate all query results to obtain $H_S^{'}$, according to the original positions of the Q_i. This process can be expressed as:

$$H_S' = \Psi_i^O(P_i), \tag{4}$$

where O represents the number of pillar-wise feature in the teacher feature map. Then, we use a feedforward neural network to enable $H_S^{'}$ to predict the teacher feature map, $G_S = \text{FFN}(H_S^{'})$, making the two maps as close as possible.

It is important to note that the semantic calibrator is only used during the pre-training stage. The purpose of incorporating the calibrator is to assist the converter and enhancer in learning representations better. Therefore, calibrator is related to the surrogate task, rather than the downstream task. Introducing the calibrator to specific downstream task reduces the model's generalization ability. Additionally, due to the inherent inconsistency between the two features arising from different perspectives, calibrator's goal is to minimize this difference during the pre-training stage. However, we aim to preserve this disparity during the inference process, as it can help address issues such as blind spots.

3.4 Loss

During the pre-training phase, we employ contrastive loss to learn semantic information from heterogeneous features. We treat pillar-wise features for the same object in two feature maps as positive sample pairs, while pillar-wise features for different objects are treated as negative sample pairs. The surrogate task involves maximizing the semantic similarity between positive sample pairs and minimizing it between negative sample pairs. Using a supervised approach, we perform sampling operations on the feature maps H_T and G_S, extracting pillar-wise features sets for real objects in the scene, denoted as $S_T \in \mathbb{R}^{N\times V\times C}$ and $S_S \in \mathbb{R}^{N\times V\times C}$, respectively, where N denotes the total number of objects in the scene, V denotes the number of pillar-wise features representing a single object.

Sample. For object m, based on the ground truth label (x, y, z, h, w, l, r), we obtain the coordinates of its detection box's four vertices on the BEV plane, denoted as $(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)$. Taking into account the yaw angle r, we calculate the potential maximum coverage range of the detection box as $[x_{\min}, x_{\max}], [y_{\min}, y_{\max}]$ on the x and y axis.

Considering the grid cell size s of the teacher encoder, we calculate the real-world range represented by each pillar-wise feature. For a pillar-wise feature $H_i = H[x_i, y_i]$, its detection range is defined as $x \in [x_0 + s \times x_i, x_0 + s \times (x_i + 1)], y \in [y_0 + s \times y_i, y_0 + s \times (y_i + 1)]$, where x_0 represents the lower bound of detection on the x-axis in the feature map, and y_0 represents the lower bound of detection on the y-axis. As illustrated in Fig. 3 (b), we consider all pillar-wise features within the maximum coverage range as the features set representing object m, denoted as $S^m = \{H_1, H_2, \ldots, H_k; H_i = H[x_i, y_i]\}$, where x_i satisfies $x_i \geq \frac{x_{\min}-x_0}{s}, x_i \leq \frac{x_{\max}-x_0}{s} - 1$, y_i satisfies $y_i \geq \frac{y_{\min}-y_0}{s}, y_i \leq \frac{y_{\max}-y_0}{s} - 1$.

Inter-object discrimination (IOD). To achieve instance-sensitive representation learning for objects, we conducted object-level contrastive learning. In order to enable the student model to learn more consistent representations from the teacher model, we take the average of features for each object in S_T to obtain the mean features of objects [23], denoted as $\bar{S_T} \in \mathbb{R}^{N\times C}$. Subsequently, after applying $L2$ regularization to $\bar{S_T}$ and S_S, we calculate the loss to minimize the semantic space difference between heterogeneous features.

For each pillar-wise feature v in object n, we individually calculate the temperature-controlled cosine similarity between the sampled pillar-wise features $s_S^{n,v} \in \mathbb{R}^C$ in the student model and the averaged representation $\bar{s}_T^m \in \mathbb{R}^C$ of object m in the teacher model, as:

$$\varphi_{n,v,m} = \frac{(s_S^{n,v})^\top \bar{s}_T^m}{\tau}. \tag{5}$$

Next, cross-entropy loss is used to maximize the semantic similarity between the student features and the teacher features for the same object, and to minimize the semantic similarity for different objects:

$$L_{IOD} = -\sum_{n=1}^{N}\sum_{v=1}^{V} \log\left(\frac{\exp\left(\varphi_{n,v,n}\right)}{\sum_m^N \exp\left(\varphi_{n,v,m}\right)}\right). \tag{6}$$

During the fine-tuning stage, we utilize common classification loss and regression loss commonly used in object detection [15]. This joint optimization helps the model quickly adapt to the cooperative perception task.

3.5 PnPDA in Fine-Tuning

During fine-tuning, PnPDA retain the semantic converter and the semantic enhancer and be integrated into a fusion network and original detection head. During training, the original encoder and detection head parameters of agents are frozen, ensuring that our model does not affect the perception performance of agents when cooperative perception is not in use.

Specifically, in the collaborative process, during the pre-training stage, the neighbor agents train a converter Φ_S^{out} to transform features into standard semantics, while the ego agent trains a converter Φ_S^{in} to transform features from standard semantics to ego semantics. Additionally, an enhancer Θ_T is trained simultaneously. Subsequently, during the fine-tuning stage, the new features are initially transformed through Φ_S^{out} to convert them from own semantics to standard semantics before being transmitted to the ego agent. The ego agent then utilizes Φ_S^{in} to transform the received features from standard semantics to the ego semantic space. After the two semantic transformations, the transformed neighbor features and enhanced ego features are inputted into a fusion network to generate fused features for the detection head's use.

4 Experiments

4.1 Dataset

We conducted our experiments on the publicly available large-scale cooperative perception dataset OPV2V [33]. OPV2V provides 3D point cloud data containing multiple autonomous agents at the same timestamp, eliminating the impact of misalignment in time on cooperative perception. Utilizing the OpenCDA [29], a cooperative driving simulation framework and the CARLA [10] simulator, the dataset generates a total of 10,915 frames, which are split into train/validation/test sets with quantities of 6,765/1,980/2,170, respectively.

4.2 Experimental Setup

Evaluation metrics. We use 3D detection accuracy as the metric to evaluate the performance of the model, which is measured by the AP at IoU thresholds of 0.5 and 0.7. We designed the evaluation range as $x \in [-140.8, 140.8], y \in [-40, 40]$, consistent with other works in this field [5,14,31,32].

Experimental design. In the experiments, we initially randomly select one agent as the ego agent. Subsequently, all point cloud data is reconstructed to a unified world coordinate system based on sensor extrinsics, with the ego agent as the center. During the pre-training stage, we employ early fusion method to aggregate neighbor agents point cloud data around the ego agent, which allows the teacher model to capture more comprehensive and accurate information. In the fine-tuning stage, the ego agent only utilizes its own perception data. The

Table 1: Characteristics of Encoder. The model differences among five encoders and their detection performance in homogeneous scenes.

Encoder	Voxel Resolution	Params (K)	2D/3D CNN Layers	Half Lidar Cropping Range (x&y)	AP@0.5
pp8	0.8, 0.8, 4	1035	10/0	[140.8, 40]	85.9
pp4	0.4, 0.4, 4	6578	19/0	[140.8, 40]	87.2
pp6	0.6, 0.6, 4	6578	19/0	[153.6, 38.4]	86.5
vn4	0.4, 0.4, 0.4	333	0/3	[140.8, 40]	85.5
vn6	0.6, 0.6, 0.4	333	0/3	[153.6, 38.4]	57.9

point cloud data from all neighbor agents is aggregated together to represent the complete scene, and then it is fed into the neighbor encoder.

To validate the model's capability to reduce domain gaps in diverse heterogeneous scenarios, we design two scenarios involving three different heterogeneous encoders, as detailed in Tab. 1. We also train two heterogeneous encoders as newly emerging models for the experiment on forward compatibility, detail in Sec. 4.4. To ensure that encoders produce effective representations, we train these encoders in homogeneous scenarios. The two designed scenarios are as follows:

1. Hetero Scenario 1: The ego agent uses the PointPillar [15] encoder pp8 to process point cloud data, while neighbor agent uses the PointPillar encoder pp4. Although the basic architecture of the two encoders is similar, there are differences in design details and parameter scales, representing a scenario with minor domain gap between features.
2. Hetero Scenario 2: The ego agent uses the PointPillar encoder pp8 to process point cloud data, while neighbor agent uses the VoxelNet [38] encoder vn4. These two encoders have significant differences in model design, corresponding to a scenario with substantial domain gap between features.

In the experiments, we reload and freeze agents' encoder and detection head ensuring consistent semantic features before and after the experiment. Moreover, we apply various feature fusion algorithms in each environment to verify our model's effectiveness and flexibility.

Implementation details. For the pre-trained encoders, detection heads and fusion network, we follow the same hyperparameters as their respective works. In the pre-training stage, we set τ in temperature-controlled cosine similarity to 0.1, and EMA momentum between the converter and enhancer to 0.8. Our initial learning rate is set to 0.001, using Adam as the optimizer, and the learning rate is decayed by 0.1 at the 10th and 50th epochs. In the fine-tuning stage, we use a similar training approach.

4.3 Detection Performance

We intuitively demonstrate the model's ability to reduce the domain gap between heterogeneous features by evaluating its detection performance in two

Table 2: 3D detection performance comparison on OPV2V. Experiments compare the detection performance between three cross-domain adaptation methods using different feature fusion networks in two heterogeneous scenes, using the AP with IOU of 0.5 and 0.7 as metrics.

Scenario	Adapter	Fusion Net			
		F-cooper	Where2comm	CoBEVT	V2X-ViT
Hetero 1	HETE	67.0/53.8	74.6/60.1	75.7/58.9	82.1/70.6
	MPDA	61.8/41.7	75.7/62.8	82.9/68.1	74.0/54.0
	PnPDA (ours)	**78.0/56.4**	**75.9/64.0**	**85.2/71.2**	**84.6/70.9**
Hetero 2	HETE	54.3/39.3	57.5/40.2	76.9/54.1	77.6/56.6
	MPDA	57.7/41.1	62.7/**49.8**	75.3/58.3	56.6/42.1
	PnPDA (ours)	**65.5/43.4**	**68.6**/42.8	**80.6/60.9**	**81.3/60.7**

heterogeneous collaborative scenarios. We first train adapters from pp4 to pp8 and from vn4 to pp8 during the pre-training stage, and then fine-tune these adapters for the objection detection task. To validate the effectiveness of PnPDA, experiments compare PnPDA with the existing heterogeneous feature domain adaptation method MPDA [30]. Another framework for heterogeneous feature collaborative perception, HEAL [20], primarily focuses on collaborating different sensor data, thus requiring re-training of agents' encoders. This makes it unfair to directly compare with PnPDA. Additionally, a simple domain adapter is designed, called HETE, which utilizes only a max-pooling layer and a 1×1 convolution layer for aligning spatial dimensions and feature channels, serving as the baseline. To ensure experimental fairness, in the same scenarios, the same encoder and detection head weights are loaded, and the same structured feature fusion network is used.

Tab. 2 shows the detection results of PnPDA and other domain adapters in different heterogeneous scenarios and with various feature fusion networks. In Hetero Scenario 1 where the encoder structures are the same, and the domain gap of heterogeneous features are small, a simple adaptation method can achieve good experimental results. However, in Hetero Scenario 2, where there is significant domain gap between features, PnPDA has a significant advantage compared to the baseline, with an average performance improvement of 7.43% and 4.40% in terms of AP at IoU thresholds of 0.5 and 0.7. Additionally, the adapter of MPDA is task-specific, so freezing the detection head limits its semantic adaptation capability. In both scenarios, experiments demonstrate that our PnPDA achieves an average performance improvement of 9.13% and 6.55% in terms of AP at IoU thresholds of 0.5 and 0.7, respectively, compared to SOTA domain adaptation methods. This significantly demonstrates PnPDA's ability to semantically transform heterogeneous features. From the experimental results, we also observe that fusion networks such as CoBEVT [31] and V2X-ViT [32] generally outperform fusion algorithms like F-cooper [5] and Where2comm [14]. We believe this is because CoBEVT and V2X-ViT have many learnable modules that also have a certain degree of domain adaptation capability.

4.4 Forward Compatibility

To validate the collaborative perception process for providing forward compatibility to PnPDA, experiments are conducted using pp8 as the ego model and pp4 as the standard model, while setting vn6 and pp6 as the newly emerging models. Firstly, both new models underwent training to learn the adapters for converting their own semantic space to standard semantic space. And ego agent training involves both a converter from standard semantics to ego semantics and an enhancer. Subsequently, during fine-tuning, only converter and enhancer of the new models' adapter are retained and integrated with the ego agent's converter, enhancer, fusion network, and detection head. Simultaneously, experiments included training MDPA and PnPDA as comparative experiments, utilizing pp8 as the ego model, pp6 and vn6 as neighbor models for training from scratch.

The experiments, presented in Tab. 3, reveal that the new collaborative perception process supports the integration of any newly emerging model with acceptable performance degradation, despite undergoing two semantic conversions. Importantly, when the newly emerging model is vn6, the perception performance of the standard vn6 model, surpasses that of the standard pp4 model, indicating performance degradation when the newly added model exhibits substantial domain gap from the standard model.

Table 3: Forward compatibility of PnPDA. PnPDA* directly trains a new converter from the ego model to the newly added model, while PnPDA trains two converters according to the process. (vn4)* represents using vn4 as the standard model for training.

	PnPDA*	MPDA	PnPDA
pp8-pp6	75.1	57.5	74.6
pp8-vn6	60.5	57.6	55.0/59.2(vn4)*

4.5 Impact of Neighbor Agent Quantity

In this experiment, the influence of the number of neighbor agents, varied from 1 to 6, on the performance of the collaborative perception model is investigated. All neighbor agents use pp4 as the encoder, while the ego agent uses pp8 as the encoder, and V2X-ViT is used as the fusion network. Since in this experiment, neighbor features are not fused directly through point cloud, as stated in Sec. 4.2, but are first transmitted to the ego agent for semantic transformation and feature fusion, to better simulate real collaborative scenarios, the detection performance may be weaker than the results in Sec. 4.3.

The results, as show in Fig. 4, indicate that increasing the number of neighboring agents leads to improved AP. This also demonstrates that our model supports multiple agents to collaborate in perception simultaneously.

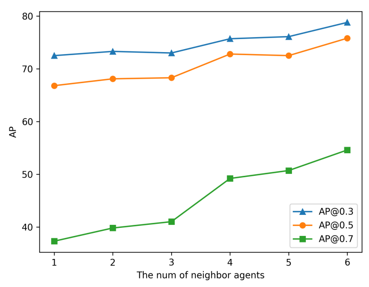


Fig. 4: The impact of the number of neighbor agents on perception performance.

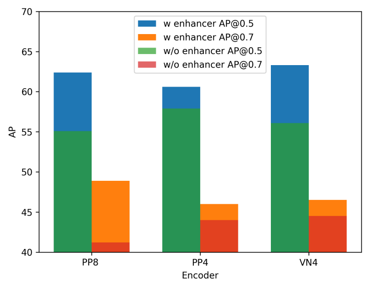


Fig. 5: The improvement in single-agent detection performance achieved by introducing enhancer.

4.6 Efficiency of Enhancer

The dedicated experiment was conducted to validate the impact of using a single enhancer on detection performance. The detection accuracy for three encoders and the detection accuracy after adding enhancer were independently trained. The results in Fig. 5 demonstrate the optimization effect of enhancer on the representational capacity of encoder output features. The detection models with added enhancer show average improvements of 5.7%/3.9% in terms of AP at IoU thresholds of 0.5 and 0.7, respectively.

4.7 Ablation Studies

To investigate the roles of each component in cooperative perception, we obtain the AP at IOU of 0.5 for different component combination. We conducted ablation experiments in Hetero Scenario 1, with the F-cooper fusion network. Our baseline model utilizes Contrastive Learning (CL) for pre-training and abandons the calibrator during the fine-tuning stage. Tab. 4 demonstrates that each component contributes to performance improvement. Particularly, the converter significantly boosts performance by 6.8%. Removing enhancer weakens the representation capability of ego features, leading to a 5.9% drop in detection performance, further confirming our conclusion in Sec. 3.2. After introducing calibrator during fine-tuning, the model's detection performance declines by 14.2%. But not using the calibrator during the pre-training stage would also lead to a decrease in performance. Because the calibrator is task-agnostic and not specific to particular downstream tasks, as discussed in Sec. 3.3. Finally, training the model from scratch without loading pre-trained weights also results in a decrease in detection accuracy.

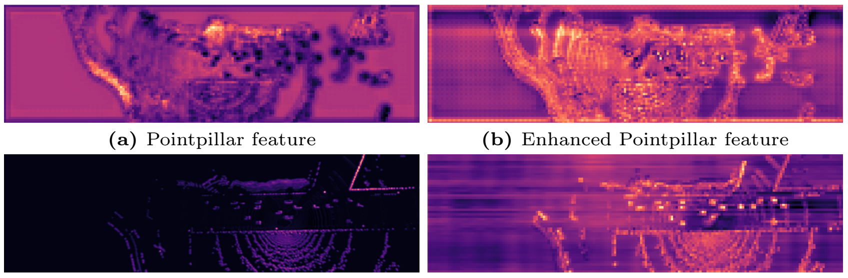
4.8 Qualitative Visualizations

We average the absolute values of all channels to visualize the feature maps and study their patterns. As shown in Fig. 6 (a) and Fig. 6 (c), there is a significant

Table 4: Ablation experiments to validate the effectiveness of components in the model. CL represents pre-training using contrastive learning. ♢ indicates that the calibrator is not used during the pre-training stage.

CL	calibrator	enhancer	converter	AP@0.5
✓		✓	✓	**78.0**
✓	✓	✓	✓	63.8
✓			✓	72.1
✓	✓		✓	61.6
✓	✓	✓		53.8
✓		✓		71.2
♢		✓	✓	74.4
		✓	✓	73.3

difference in the original semantics of the two features. For PointPillar, the target positions on the feature map have relatively lower values, while for VoxelNet, it is exactly the opposite, with lower feature values in the background regions. However, after passing through the adapter, as shown in Fig. 6 (b) and Fig. 6 (d), the patterns of the two features have clearly become closer, while still retaining the original pattern of the ego agent.



(a) Pointpillar feature **(b)** Enhanced Pointpillar feature

(c) VoxelNet feature **(d)** Transformed VoxelNet features

Fig. 6: Visualization of intermediate features before and after domain adaptation. Brighter regions correspond to higher feature values.

5 Conclusions

In this work, we attempt to address the domain adaptation among heterogeneous features using contrastive learning. Through carefully designed plug-and-play domain adapter, we successfully achieve cooperative perception for heterogeneous features without disrupting the original agent detection network. We also design a new collaborative process to support the dynamic joining of agents. This enables PnPDA to excel in collaborative performance, model protection, and bidirectional compatibility. We validate the effectiveness of the model on OPV2V dataset. However, PnPDA's limitation lies in its ability to handle only point cloud features. In future work, we will explore the use of multi-modality techniques to enable PnPDA to handle image features.

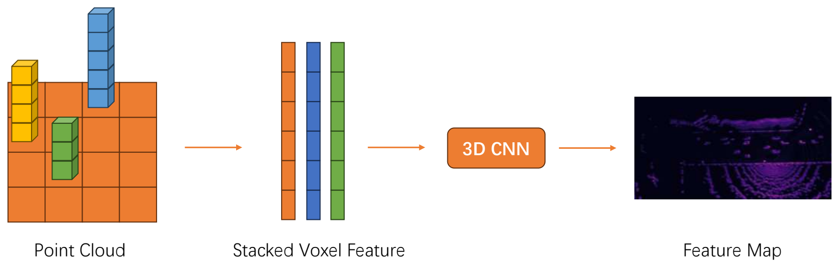
Appendix

In this supplementary material, we first provide a detailed introduction to the current mainstream point cloud feature encoders, especially two of them involved in the experiment, in Appendix A. Next, we will introduce how to use self-supervised learning (SSL) for pre-training, in Appendix B. We supplement three additional experiments to further validate the effectiveness of our model, in Appendix C.

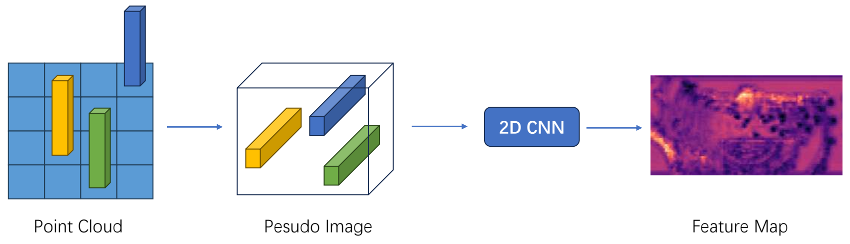
A Point Cloud Feature Encoders

The current mainstream point cloud feature encoders are PointPillar [15] , VoxelNet [38], and SECOND [34]. Due to significant differences in model architecture and processing pipelines between PointPillars and VoxelNet, there exists a substantial domain gap between two models. These encoders use voxelization techniques, mapping continuous point cloud to discrete voxel grids, then utilize convolution neural networks as backbone networks to process structured voxel data, ultimately obtaining a 2D feature map from a BEV perspective. Specifically, VoxelNet, as shown in Fig. 7 (a), divides a 3D space into equally spaced voxels and groups points based on the voxel they belong to. However, due to factors such as distance, occlusion, and uneven sampling, the LiDAR point cloud is sparse and its density varies across the entire space. Therefore, after grouping, the number of points in each voxel is different. VoxelNet introduces a hyperparameter R, which randomly samples a fixed number of points for voxels with more than R points, and fills voxels with less than R points with zeros. This helps reduce sampling bias and increase training diversity. It then employs stacked Voxel Feature Encoding layers to encode each voxel, followed by 3D convolutions to further aggregate local voxel features, transforming the point cloud into a high-dimensional voxel representation. Finally, the Region Proposal Network is used to take this representation as input and generate detection results.

PointPillar, as shown in Fig. 7 (b), differs from VoxelNet in that it only partitions the entire point cloud into numerous columns on the xOy plane, each of which has a height equal to the point cloud's height (approximately infinite in the z-direction, thus not requiring hyperparameters to control the z-dimension). Point cloud sampling works the same as VoxelNet. Due to compression of information in the spatial height, PointPillar transforms the entire point cloud data into a dense 2D tensor. Subsequently, a simplified PointNet is used, applying a linear layer, a BN layer, and a ReLU layer to each point. This process avoids 3D convolutions in intermediate layers, greatly improving computational efficiency. After obtaining the BEV feature map, the convolutional network employs a top-down network to generate features at increasingly smaller spatial resolutions and uses upsampling and concatenation on the features from top to bottom to obtain a high-dimensional representation of the features. Finally, a Single Shot Detector detection head is used to handle 3D object detection.



(a) VoxelNet



(b) PointPillar

Fig. 7: Point Cloud Encoder

B Self-Supervised Contrastive Learning

During the pre-training phase based on contrastive learning, distinguishing multiple objects from point cloud data is a critical issue. In the main text, we directly use ground-truth bounding boxes from the dataset to differentiate objects, which requires high-quality labels. However, manually annotating large-scale autonomous driving data incurs significant costs. To overcome this issue, we employ a self-supervised approach based on density-based spatial clustering (DBSCAN) [2] to differentiate objects. Specifically, we utilize DBSCAN to cluster point cloud data into clusters representing different objects and noise points. This allows us to discern different objects in the scene without explicit labels.

DBSCAN is a clustering algorithm that does not require pre-defining the number of clusters. Instead, it identifies clusters based on the density of points in the data space. Points that are closely packed together are grouped into clusters, while areas of low density serve as boundaries between clusters. Points that do not belong to any cluster are classified as outliers or noise. This ability to automatically detect clusters of arbitrary shapes and sizes makes DBSCAN particularly useful for datasets without high-quality labels. The specific workflow is as follows:

1. Select points within ϵ distance and with at least m neighbors as core points.
2. Expand clusters by linking core points with their neighbors that are reachable based on density.
3. If points in the current cluster cannot be further expanded, DBSCAN searches for unvisited points and repeats the process.

4. Points that cannot be assigned to any cluster and have fewer than m points in their neighborhood are classified as noise points.

As these clusters exhibit different shapes and sizes. To standardize their dimensions for subsequent operations, we determine the minimum and maximum values of the x and y coordinates, denoted as $x_{\min}, x_{\max}, y_{\min}, y_{\max}$, respectively, within each cluster to establish the boundaries of each object. Subsequent processing remains consistent with the supervised approach.

It is important to note that the self-supervised approach based on DBSCAN may not achieve the same precision as ground-truth bounding boxes for objects. As shown in Fig. 8, the red candidate boxes, generated by clustering, in the image exhibit certain deviations from the green ground-truth boxes. This can lead to a decrease in the performance of the model during the pre-training phase. Additionally, DBSCAN's sensitivity to noise points can indeed impact clustering effectiveness. The red circles in Fig. 8 represent false candidate boxes mistakenly identified by DBSCAN due to noise points in the point cloud data. Nevertheless, in large-scale autonomous driving datasets, high-quality manual annotations may be lacking. Using SSL method for contrastive learning can effectively address this issue.

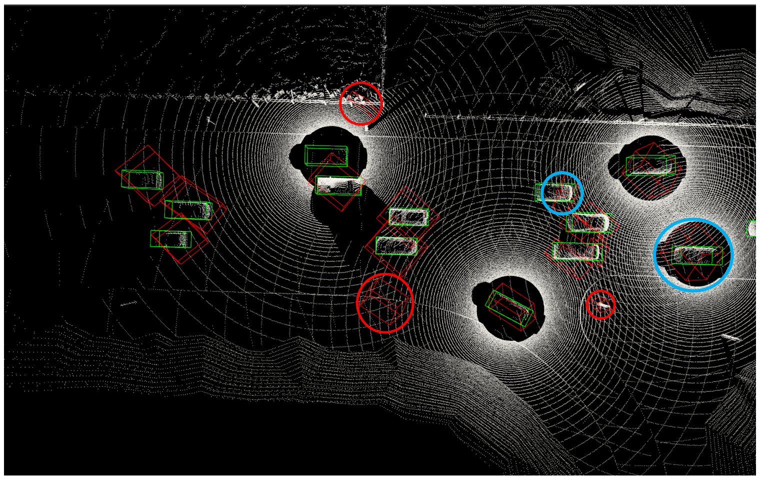


Fig. 8: Points clustered from point cloud. Red boxes indicating candidate boxes generated by clustering, green boxes indicating ground-truth target boxes. Red circles represent false candidate boxes generated due to noise, while blue circles represent the discrepancy in object accuracy between candidate boxes and ground-truth boxes.

C More Experimental Results

C.1 SSL Method

During the pre-training phase, we utilized the same dataset and parameter settings as supervised pre-training to implement self-supervised contrastive learning based on DBSCAN. To validate the model's performance, we independently trained it in Hetero Scenario 1 and Hetero Scenario 2, and compared the results with models trained using supervised pre-training.

Table 5: PnPDA employs two approaches for pre-training and evaluating detection performance, where SL represents using supervised learning during pre-training stage. The experiment employs the AP with IOU of 0.5 and 0.7 as metrics.

Scenario	Pre-training Method	Fusion Net			
		F-cooper	Where2comm	CoBEVT	V2X-ViT
Hetero 1	SSL	64.0/32.6	73.4/56.8	82.8/62.5	78.2/52.0
	SL	**78.0/56.4**	**75.9/64.0**	**85.2/71.2**	**84.6/70.9**
Hetero 2	SSL	53.7/38.4	53.6/38.3	**82.7/68.9**	**83.0/65.5**
	SL	**65.5/43.4**	**68.6/42.8**	80.6/60.9	81.3/60.7

The comparison between self-supervised and supervised pre-training methods on detection performance, as shown in Tab. 5, demonstrates that supervised approaches generally outperform self-supervised methods. In Hetero Scenario 2, within a specific fusion network, self-supervised method demonstrates superior performance, showcasing its capability to address complex challenges.

C.2 Data Utilization Efficiency

Due to the characteristics of contrastive learning, we employed training subsets of different scales for fine-tuning in Hetero Scenario 1, with the F-cooper fusion network, to investigate the efficiency of data utilization during the fine-tuning stage. We evaluated the model's performance on the entire validation set, focusing on whether the model could converge quickly when using only a portion of the training data during fine-tuning. The results in Fig. 9 show that the model can rapidly approach the level of performance achieved with the entire training sample size when using only half of the data during fine-tuning. The subsequent increase in training data has a smoothing effect on the improvement in detection performance. This is attributed to the well-established representations learned during the pre-training stage.

C.3 Model Integrity

If the original model is re-trained during the collaborative perception training phase, it may lead to a decrease in detection performance when re-performing

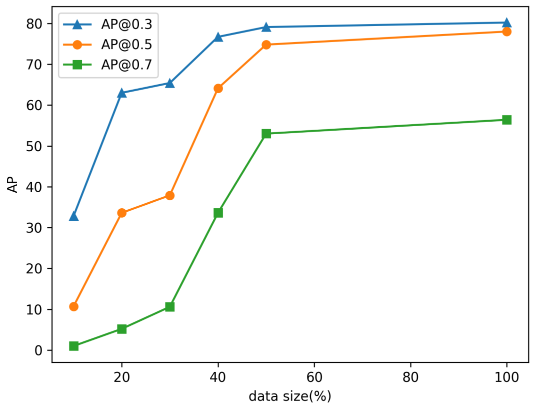


Fig. 9: illustrate the impact of training the model with data of different scales during the fine-tuning stage on the final detection performance of the model

individual detection. In the implementation scenario, when there is poor communication quality and the agent fails to receive features from neighboring devices, the agent will have to rely on independent perception. As result, we conducted experiments on MDPA and PnPDA to validate our findings.

We conduct experiments under Hetero Scenario 1 and Hetero Scenario 2, where cooperative perception employs F-cooper as the fusion network. Next, we train self-agent to neighbor-agent adapters under two different scenarios using MPDA and PnPDA, respectively. In this process, MPDA re-trains the detection head according to original specification, whereas PnPDA freezes the detection head to simulate the two models' behaviors in real-world scenarios. Finally, we evaluate the detection performance of the two models under collaborative perception and independent perception modes. The model trained with MPDA uses the re-trained detection head during independent perception. This experiment aims to investigate whether switching the agents trained with heterogeneous collaborative perception models back to independent perception mode would result in a decrease in detection performance.

Tab. 6 demonstrates that re-training the original model will result in a significant performance degradation in the original task. Therefore, if the collaborative perception of heterogeneous features is implemented by re-training model components, the agent will not be able to independently perform perception tasks and must rely on the assistance of the new components. In contrast, agents using a plug-and-play collaborative perception model do not encounter this issue. Besides, the performance of independent perception will not be affected. This enhances the flexibility of agents in performing perception tasks.

Additionally, since MPDA re-trains the detection head while PnPDA freezes the detection head, the detection results of MPDA are expected to be superior

Table 6: The detection performance of the heterogeneous collaborative model under collaborative perception and independent perception model. Collaborative refers to the model trained with adapters for collaborative perception, Independent refers to the model trained with adapters for independent perception.

Scenario	Adapter	Perception Mode	
		Collaboration	Independent
Hetero 1	MPDA	70.3/50.4	14.9/10.0
	PnPDA (ours)	78.0/56.4	**53.4/38.2**
Hetero 2	MPDA	76.1/56.4	41.1/22.0
	PnPDA (ours)	65.5/43.4	**53.4/38.2**

to those of PnPDA in Hetero scenario 2. However, even under fair conditions, our model outperforms MPDA, as demonstrated in the main text.

Acknowledgements

This work was supported in part by the National Key Research and Development Program of China under Grant 2022YFB4300400, in part by the Natural Science Foundation of China under Grant 62272053, Grant 62102041 and Grant 62001054, in part by the Beijing Nova Program under Grant 20230484364, and in part by the Young Elite Scientists Sponsorship Program by China Association for Science and Technology (CAST) under Grant 2022QNRC001.

References

1. Arnold, E., Dianati, M., de Temple, R., Fallah, S.: Cooperative perception for 3D object detection in driving scenarios using infrastructure sensors. IEEE Transactions on Intelligent Transportation System **23**(3), 1852–1864 (2020)
2. Campello, R.J., Moulavi, D., Sander, J.: Density-based clustering based on hierarchical density estimates. In: PAKDD. pp. 160–172 (2013)
3. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: ECCV. pp. 213–229 (2020)
4. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: ICCV. pp. 9650–9660 (2021)
5. Chen, Q., Ma, X., Tang, S., Guo, J., Yang, Q., Fu, S.: F-cooper: Feature based cooperative perception for autonomous vehicle edge computing system using 3D point clouds. In: SEC. pp. 88–100 (2019)
6. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: ICML. pp. 1597–1607 (2020)
7. Chen, X., He, K.: Exploring simple siamese representation learning. In: CVPR. pp. 15750–15758 (2021)
8. Chen, Z., Shi, Y., Jia, J.: TransIFF: An instance-level feature fusion framework for vehicle-infrastructure cooperative 3D detection with transformers. In: ICCV. pp. 18205–18214 (2023)

9. Cui, J., Qiu, H., Chen, D., Stone, P., Zhu, Y.: Coopernaut: End-to-end driving with cooperative perception for networked vehicles. In: CVPR. pp. 17252–17262 (2022)
10. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V.: CARLA: An open urban driving simulator. In: CoRL (2017)
11. Grill, J.B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M., et al.: Bootstrap your own latent-a new approach to self-supervised learning. In: NeurIPS. pp. 21271–21284 (2020)
12. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: CVPR. pp. 9729–9738 (2020)
13. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. In: NIPS Deep Learning Workshop. p. 9 (2015)
14. Hu, Y., Fang, S., Lei, Z., Zhong, Y., Chen, S.: Where2comm: Communication-efficient collaborative perception via spatial confidence maps. In: NeurIPS. pp. 4874–4886 (2022)
15. Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J., Beijbom, O.: Pointpillars: Fast encoders for object detection from point clouds. In: CVPR. pp. 12697–12705 (2019)
16. Li, Y., Fang, Q., Bai, J., Chen, S., Juefei-Xu, F., Feng, C.: Among us: Adversarially robust collaborative perception by consensus. In: ICCV (2023)
17. Li, Y., Ren, S., Wu, P., Chen, S., Feng, C., Zhang, W.: Learning distilled collaboration graph for multi-agent perception. In: NeurIPS. pp. 29541–29552 (2021)
18. Li, Z., Wang, W., Li, H., Xie, E., Sima, C., Lu, T., Qiao, Y., Dai, J.: BEVFormer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In: ECCV. pp. 1–18 (2022)
19. Liu, Y.C., Tian, J., Glaser, N., Kira, Z.: When2com: Multi-agent perception via communication graph grouping. In: CVPR. pp. 4106–4115 (2020)
20. Lu, Y., Hu, Y., Zhong, Y., Wang, D., Chen, S., Wang, Y.: An extensible framework for open heterogeneous collaborative perception. In: ICLR (2024)
21. Luo, G., Shao, C., Cheng, N., Zhou, H., Zhang, H., Yuan, Q., Li, J.: Edgecooper: Network-aware cooperative lidar perception for enhanced vehicular awareness. IEEE Journal on Selected Areas in Communications **42**(1), 207–222 (2024)
22. Luo, G., Zhang, H., Yuan, Q., Li, J.: Complementarity-enhanced and redundancy-minimized collaboration network for multi-agent perception. In: ACM MM. pp. 3578–3586 (2022)
23. Nunes, L., Wiesmann, L., Marcuzzi, R., Chen, X., Behley, J., Stachniss, C.: Temporal consistent 3D lidar representation learning for semantic perception in autonomous driving. In: CVPR. pp. 5217–5228 (2023)
24. Rawashdeh, Z.Y., Wang, Z.: Collaborative automated driving: A machine learning-based method to enhance the accuracy of shared information. In: ITSC. pp. 3961–3966 (2018)
25. Song, K., Xie, J., Zhang, S., Luo, Z.: Multi-mode online knowledge distillation for self-supervised visual representation learning. In: CVPR. pp. 11848–11857 (2023)
26. Xiang, H., Xu, R., Ma, J.: HM-ViT: Hetero-modal vehicle-to-vehicle cooperative perception with vision transformer. In: CVPR (2023)
27. Xie, S., Gu, J., Guo, D., Qi, C.R., Guibas, L., Litany, O.: PointContrast: Unsupervised pre-training for 3D point cloud understanding. In: ECCV. pp. 574–591 (2020)
28. Xu, R., Chen, W., Xiang, H., Xia, X., Liu, L., Ma, J.: Model-agnostic multi-agent perception framework. In: ICRA. pp. 1471–1478 (2023)

29. Xu, R., Guo, Y., Han, X., Xia, X., Xiang, H., Ma, J.: OpenCDA: An open cooperative driving automation framework integrated with co-simulation. In: ITSC. pp. 1155–1162 (2021)
30. Xu, R., Li, J., Dong, X., Yu, H., Ma, J.: Bridging the domain gap for multi-agent perception. In: ICRA. pp. 6035–6042 (2023)
31. Xu, R., Tu, Z., Xiang, H., Shao, W., Zhou, B., Ma, J.: CoBEVT: Cooperative bird's eye view semantic segmentation with sparse transformers. In: CoRL. pp. 989–1000 (2023)
32. Xu, R., Xiang, H., Tu, Z., Xia, X., Yang, M.H., Ma, J.: V2X-ViT: Vehicle-to-everything cooperative perception with vision transformer. In: ECCV. pp. 107–124 (2022)
33. Xu, R., Xiang, H., Xia, X., Han, X., Li, J., Ma, J.: OPV2V: An open benchmark dataset and fusion pipeline for perception with vehicle-to-vehicle communication. In: ICRA. pp. 2583–2589 (2022)
34. Yan, Y., Mao, Y., Li, B.: Second: Sparsely embedded convolutional detection. Sensors **18**(10), 3337 (2018)
35. Yang, K., Yang, D., Zhang, J., Li, M., Liu, Y., Liu, J., Wang, H., Sun, P., Song, L.: Spatio-temporal domain awareness for multi-agent collaborative perception. In: ICCV. pp. 23326–23335 (2023)
36. Yin, J., Zhou, D., Zhang, L., Fang, J., Xu, C.Z., Shen, J., Wang, W.: ProposalContrast: Unsupervised pre-training for lidar-based 3D object detection. In: ECCV. pp. 17–33 (2022)
37. Zhang, H., Luo, G., Li, Y., Wang, F.Y.: Parallel vision for intelligent transportation systems in metaverse: Challenges, solutions, and potential applications. IEEE Transactions on Systems, Man, and Cybernetics: Systems **53**(6), 3400–3413 (2023)
38. Zhou, Y., Tuzel, O.: VoxelNet: End-to-end learning for point cloud based 3D object detection. In: CVPR. pp. 4490–4499 (2018)