# *TurboTrain*: Towards Efficient and Balanced Multi-Task Learning for Multi-Agent Perception and Prediction

Supplementary Material

## A. Details of Problem Formulation

### A.1. Explanation of Different Training Strategies

In Sec. 3 of the main paper, we show the comparison of one-time training, manual training, and *TurboTrain* in Fig.2. The numerical experiment results of three training strategies are provided in Tab. S1.

**One-time training strategy**. This strategy directly trains all the modules in the end-to-end framework. Row 1 (One-time (s-f)) indicates we only train *multi-agent single-frame perception*. Row 2 (One-time (s-a)) indicates we only train *single-agent perception and prediction*. Row 3 (One-time (all)) indicates we only train *multi-agent perception and prediction*.

**Manual training**. Rows four to seven indicate the performance breakdown of the manual training strategy. For the manual training strategy, we divide the training process into four stages. *Stage 1: Single-Agent Detection* (Row 4). We train the detection backbone on single frames to build a robust foundation for object recognition. *Stage 2: Single-agent Temporal Prediction* (Row 5). We freeze the detection backbone and train a dedicated temporal network and prediction head to capture complex temporal dynamics for the prediction task. *Stage 3: Single-agent Perception and Prediction Joint Fine-Tuning* (Row 6). We unfreeze the entire network and fine-tune all components end-to-end, harmonizing the perception and prediction tasks. *Stage 4: Multi-Agent Fusion* (Row 7). We incorporate a multi-agent fusion module with dynamic loss weighting to jointly optimize spatiotemporal representations across agents, ensuring balanced performance.

**TurboTrain**. Row 8 indicates that we only use the Pretrain stage of TurboTrain and finetune the model without using the Balance stage. Row 9 indicates we use the whole TurboTrain pipeline. From the results, we observe that 1) one-time training faces challenges with multi-agent spatiotemporal feature learning; 2) manual training strategy avoids such learning failure but requires more monitoring stages; 3) TurboTrain achieves superior performance while avoiding such learning inefficiency.

### A.2. Motivating Research Questions

**Q: Why is it challenging for efficient and balanced multi-task learning for multi-agent perception and prediction and why is such a problem important?**

Table S1. Comparison of One-time training, Manual training, and *TurboTrain* on the V2XPnP-Seq-VC dataset with the V2XPnP model. -x represents the performance decline compared to the manual training strategy, and +x represents the improvement. In one-time training strategy, "s-f" indicates the single frame model, "s-a" means the single agent model, and "all" represents the multi-frame multi-agent model. Moreover, "No Bal" represents a pretrain-only model without the balance stage.

| Strategy | **AP@0.5**(%) ↑ | ADE(m) ↓ | FDE(m) ↓ | MR(%) ↓ | **EPA**(%) ↑ |
|---|---|---|---|---|---|
| One-time (s-f) | 68.3-2.0 | - | - | - | - |
| One-time (s-a) | 55.1-15.2 | 1.60 | 2.99 | 37.5 | 29.9-12.9 |
| One-time (all) | 39.0-31.3 | 1.42 | 2.44 | 30.0 | 15.0-27.8 |
| Single-agent Det | 45.2 | - | - | - | - |
| Temporal Pred | 48.2 | 2.03 | 3.55 | 38.3 | 20.2 |
| Joint Tuning | 57.4 | 1.58 | 2.89 | 38.3 | 32.5 |
| Multi-agent Fus | 70.3 | 1.53 | 2.80 | 37.8 | 42.8 |
| TurboTrain (No Bal) | 70.3 | 1.54 | 2.80 | 37.2 | 43.4 |
| **TurboTrain** | **72.2**+1.9 | **1.49** | **2.75** | **35.0** | **45.5**+2.7 |

**A:** Training a system that processes data from multiple agents across several frames presents significant challenges. Conventional one-time training, where all tasks are learned jointly from scratch, fails to capture the intricate features that arise from merging temporal and multi-agent data. This is due to the inherent complexity of integrating spatial and temporal information, which often leads to unstable training and suboptimal performance. For instance, tasks like detection may dominate, thereby overshadowing others such as trajectory prediction. As demonstrated in Tab. S1, configurations involving either single-agent multi-task or multi-agent single-task scenarios do not encounter these issues, whereas the multi-agent multi-task setup places much higher demands on the feature learning process. Moreover, manual multi-stage training strategies heavily depend on the careful selection of checkpoints at each stage, and system errors tend to accumulate as the number of stages increases. This approach also relies on annotated labels, making it less effective in scenarios where data is scarce. These challenges highlight the need to thoroughly investigate and address this learning problem. The following questions further clarify our motivation for developing the solutions presented in our work.

**Q: Why use mask reconstruction pretraining?**

**A:** We adopt mask reconstruction pretraining to enable our model to learn robust features without relying on annotated

labels. This approach generates a comprehensive 4D representation, which serves as a strong initialization for critical components in our end-to-end framework—namely, the 3D Detection Encoder, Temporal Fusion Module, and Multi-agent Fusion Module. As demonstrated in our main paper, each of these modules plays an essential role in system performance. By masking portions of the temporal input data and requiring the network to reconstruct them for each agent, we compel the model to learn the scene's underlying structure as well as sensor distributions of heterogeneous agents. Our dual reconstruction strategy, which operates at both the point level and voxel level, allows the pretrained module to capture detailed 3D structures for the perception task and to understand scene occupancy layouts for the prediction task. This methodology proves particularly valuable when data is scarce or incomplete, as it enables the model to extract fine details at multiple scales, improving its ability to handle occlusions and missing data—common challenges in real-world driving scenarios. Furthermore, our experiments confirm the effectiveness of this approach in data-scarce environments.

**Q: Why use balancing in multi-task learning?**

**A:** Balancing is critical in our multi-task setup. Different tasks (like detection and prediction) can produce conflicting gradient signals during training, meaning one task improves while another is worse. Our approach uses a conflict-suppressing gradient alignment mechanism to remove the conflict components between multiple tasks. This helps stabilize training and improves overall performance by mitigating the negative cross-task interference. Moreover, our hybrid training strategy alternates between free and balanced gradient steps, making the training process more efficient without a heavy computational cost.

# B. Implementation Details

In this section, we provide detailed configurations for our *TurboTrain* paradigm. Moreover, we go further to the baseline models used in our experiments for cooperative perception and prediction tasks.

## B.1. Baseline Model Details

**LiDAR Perception Backbone.** We adopt the anchor-based SECOND model [3] as the LiDAR feature extraction backbone for all the baseline models. The voxel resolution in the experiment is set to 0.1 m in both the $x$ and $y$ directions and 0.2 m in the $z$ direction, with a maximum of 5 points per voxel and up to 32,000 voxels. Additionally, we configure 2 anchors per BEV grid cell.

**Multi-Agent End-to-End Methods.** As research into end-to-end multi-agent perception and prediction tasks is still in the early stage, most existing studies primarily focus on single-frame spatial fusion or incorporating short-term temporal information. Therefore, we adopt the V2XPnP end-to-end framework and benchmark [8] to reimplement and evaluate state-of-the-art (SOAT) V2X fusion methods. Specifically, we follow the V2XPnP framework and leverage the baseline temporal fusion model of V2XPnP to build the end-to-end spatiotemporal fusion framework for multiple tasks and incorporate SOAT multi-agent fusion methods as baselines, including *FFNet* [6], *DiscoNet* [4], *CoBEVFlow*, *F-Cooper* [1], and V2XPnP [8]. From the main experiment results in Sec. 5, we demonstrate that our *TurboTrain* method is generalizable across various fusion methods.

**Map Feature Extraction.** HD maps for prediction are modeled as sets of polylines, where each map polyline comprises 10 sequential points. For BEV projection, each grid cell retains the five nearest polylines to reduce redundancy. Each waypoint is defined by seven attributes: $(x, y, d_x, d_y, type, x_{pre}, y_{pre})$, which captures its spatial coordinates, orientation, lane type, and preceding position. These attributes are embedded via MLP layers and the map interaction is captured by one Transformer layer.

## B.2. Additional Pretraining Details

**Formulation of occupancy reconstruction loss.** Given the predicted occupancy value and the ground truth occupancy value, we define the occupancy prediction loss as:

$$\mathcal{L}_{occ} = -\alpha \left(1 - Pr^i\right)^{\gamma} \log(Pr^i), \tag{S1}$$

where $Pr^i$ represents the predicted probability of voxel $i$. The weighting factor $\alpha$ is set to 2, and the weighting factor $\gamma$ is set to 0.25.

## B.3. Detailed Experimental Setting

**Training details.** The pretraining network contains a 3D Backbone, Temporal Fusion module, Multi-agent Fusion Module, and two separate lightweight decoder heads for reconstruction tasks. During the finetuning stage, only the pretrained 3D Backbone, Temporal Fusion module and Multi-agent Fusion Module will be taken. During pretraining, we employ AdamW [2] optimizer with a weight decay of $1 \times 10^{-2}$ to optimize our models. We train the model with a batch size of 4 for 15 epochs using a learning rate of 0.002, and we decay the learning rate with a cosine annealing [5]. We use a masking ratio of 0.7 in our main experiments and a fixed predicted point cloud number of 20 for point cloud reconstruction. During the fine-tuning stage, the optimization process is identical to the train-from-scratch baselines.

**Testing details.** During testing, a fixed agent is designated as the ego agent in each cooperative scenario, while during training, the ego agent is randomly assigned. Following the real-world evaluation setting [7], the communication range is set to 50 meters with surrounding agents evaluated within the range of $x \in [-102.4, 102.4]$ m and $y \in [-40, 40]$ m,

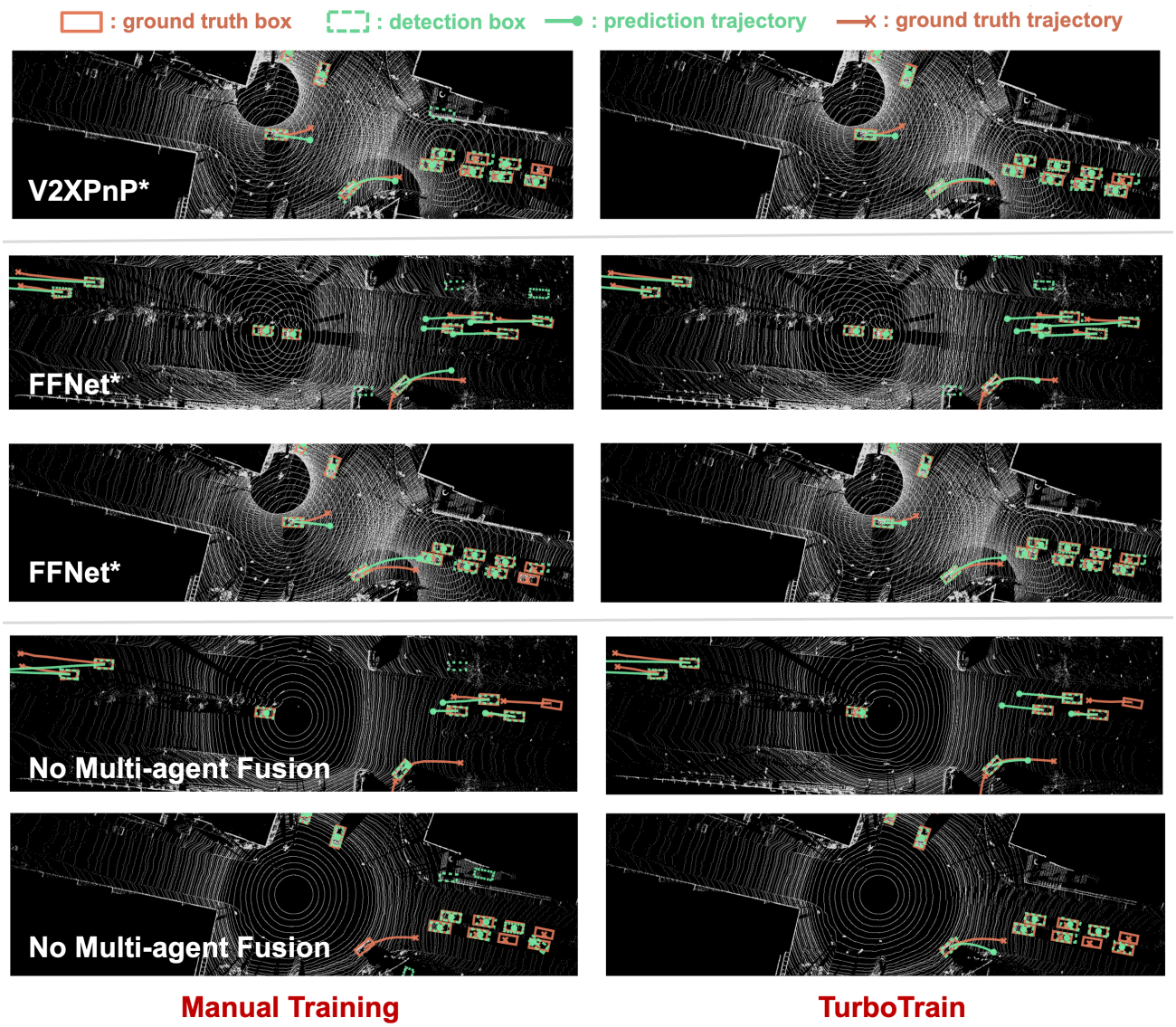


Figure S1. Qualitative comparison of different training strategies applied to the different fusion methods for multi-agent perception and prediction tasks. The proposed *TurboTrain* framework significantly enhances both detection and prediction quality over different fusion methods. Models marked with * indicate our reimplementations to ensure consistency with prior works.

and messages exceeding 50 meters are discarded. The historical observation length is set to 2 seconds (2 Hz), while the prediction horizon extends to 3 seconds (2 Hz). All models are trained using the Adam optimizer [2] with early stopping.

## C. Additional Results

### C.1. Ablation Studies

**Ablation on Reconstruction Objective Design.** We evaluate the impact of different reconstruction objectives by varying the number of historical frames used, as shown in Tab. S2. Specifically, we compare reconstructing from all available historical frames versus using only half. Given that the historical observation window is set to 2 seconds, our results indicate that leveraging a larger number of past frames substantially enhances the model's performance, highlighting the importance of temporal context in representation learning.

**Ablation on Masking ratio.** We investigate the effect of varying the masking ratio and observe that a ratio of 0.7 yields the best performance, as shown in Tab. S3. This suggests that an optimal level of information removal is crucial for effective learning, balancing the trade-off between preserving sufficient context and encouraging robust feature

extraction.

Table S2. Ablation on Input Temporal Frames on V2XPnP-Seq-VC dataset with V2XPnP model. $T$ denotes the total number of historical frames.

| Input Frames | **AP@0.5** ↑ | **EPA** ↑ |
|---|---|---|
| $T/2$ | 64.2 | 38.8 |
| $T$ | **70.3** | **43.4** |

Table S3. Ablation on Reconstruction Mask Ratio on V2XPnP-Seq-VC dataset with V2XPnP model.

| Mask Ratio | **AP@0.5** ↑ | **EPA** ↑ |
|---|---|---|
| 0.7 | **70.3** | **43.4** |
| 0.8 | 67.1 | 42.9 |
| 0.9 | 66.4 | 42.1 |

## C.2. Qualitative Results

Fig. S1 shows additional qualitative comparisons across vehicle-centric and vehicle-to-vehicle scenario with various models: (1) No Multi-Agent Fusion, (2) FFNet [6], and (3) V2XPnP [8]. Our TurboTrain framework consistently improves both detection and prediction performance across these configurations.

## References


[1] Qi Chen, Xu Ma, Sihai Tang, Jingda Guo, Qing Yang, and Song Fu. F-cooper: Feature based cooperative perception for autonomous vehicle edge computing system using 3d point clouds. In *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing*, pages 88–100, 2019. 2

[2] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 2, 3

[3] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12697–12705, 2019. 2

[4] Yiming Li, Shunli Ren, Pengxiang Wu, Siheng Chen, Chen Feng, and Wenjun Zhang. Learning distilled collaboration graph for multi-agent perception. In *Thirty-fifth Conference on Neural Information Processing Systems (NeurIPS 2021)*, 2021. 2

[5] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. 2

[6] Haibao Yu, Yingjuan Tang, Enze Xie, Jilei Mao, Ping Luo, and Zaiqing Nie. Flow-based feature fusion for vehicle-infrastructure cooperative 3d object detection. *Advances in Neural Information Processing Systems*, 36, 2024. 2, 4

[7] Zhaoliang Zheng, Xin Xia, Letian Gao, Hao Xiang, and Jiaqi Ma. Cooperfuse: A real-time cooperative perception fusion framework. In *2024 IEEE Intelligent Vehicles Symposium (IV)*, pages 533–538, 2024. 2

[8] Zewei Zhou, Hao Xiang, Zhaoliang Zheng, Seth Z Zhao, Mingyue Lei, Yun Zhang, Tianhui Cai, Xinyi Liu, Johnson Liu, Maheswari Bajji, et al. V2XPnP: Vehicle-to-everything spatio-temporal fusion for multi-agent perception and prediction. *arXiv preprint arXiv:2412.01812*, 2024. 2, 4