Future-Aware Interaction Network For Motion Forecasting

Supplementary Material

Table of Contents

A. Implementation Details

In this section, we provide additional details to facilitate the implementation and reproducibility of the proposed FINet.

A.1. Experiment Details

Our models are trained for 60 epochs using the AdamW optimizer [1] with a batch size of 16 per GPU. The training process is end-to-end, utilizing a learning rate of 0.002 and a weight decay of 0.01. An agent-centric coordinate system is employed, with scene elements sampled within a 150-meter radius of the target agents. All training is performed on 10 NVIDIA RTX A5000 GPUs while inference is conducted on a single NVIDIA RTX A5000 GPU with batch size equals to one.

A.2. Architecture Details

We present the architecture details of the proposed method, which consists of Lightweight Scene Encoder (LSEnc), Future-Aware Interaction Mamba (FIM) and Temporal Enhanced Decoder (TEDec).

Lightweight Scene Encoder (LSEnc) processes both input trajectory data and lane data simultaneously. For trajectory data processing, we employ four Mamba blocks.

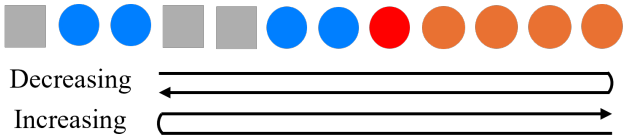


Figure A. Scan Order

For lane data processing, we utilize a PointNet architecture consisting of two Conv-BatchNorm-ReLU-Conv layers, followed by max pooling at the end.

Future-Aware Interaction Mamba (FIM) utilizes a two-stage approach for spatial interaction modeling. In the first stage, four Bi-Mamba blocks are applied, while in the second stage, two Bi-Mamba blocks are used to refine the results.

Temporal Enhanced Decoder (TEDec) consists of six CAM blocks, each comprising a cross-attention block followed by a Bi-Mamba block. Auxiliary supervision is applied at the output of the second CAM block.

The effectiveness of the number of layers is shown in Tab. A.

A.3. Mamba Flops Computation

We use the widely adopted library thop[1] to compute the model's floating point operations per second (FLOPs). Since Mamba is not a built-in module in PyTorch, we implement a customized version of the thop function to compute the FLOPs for Mamba, following the suggestion in Mamba official repository[2], which points out that Mamba's FLOPs can be approximated as basic operation FLOPs plus selective scan mechanism (SSM) FLOPs:

$$FLOPs(SSM) = B \times L \times 9 \times d_{model} \times d_{state} \quad (1)$$

where B is the batch size (set to 1 during testing), L is the sequence length, and d_{model} and d_{state} represent the input and internal state feature sizes, respectively.

Although Mamba significantly reduces FLOPs, its efficiency improvement is not proportional, likely due to differences in hardware utilization between Mamba (SSM) and Transformers. While Transformers have quadratic complexity in sequence length and high FLOPs, they are highly optimized for GPU parallelism with efficient matrix multiplications and attention kernels. In contrast, Mamba, despite its lower FLOPs, relies more on sequential computation and memory-bound operations, which GPUs struggle to parallelize efficiently. Additionally, Transformers benefit

[1] https://github.com/Lyken17/pytorch-OpCounter
[2] https://github.com/state-spaces/mamba/issues/110

FIM		TEDec		b-minFDE$_6$ (↓)	minADE$_6$ (↓)	minFDE$_6$ (↓)	MR$_6$ (↓)	minADE$_1$ (↓)	minFDE$_1$ (↓)
2	2	2	4	1.95	0.66	1.29	0.16	1.61	4.07
2	4	2	4	1.97	0.66	1,31	9.16	1.60	4.05
4	2	2	4	1.93	0.65	1.27	0.15	1.57	3.94
4	4	2	4	1.97	0.66	1.31	0.16	1.60	4.04
4	2	2	2	1.94	0.65	1.28	0.15	1.58	3.97
4	2	2	4	1.93	0.65	1.27	0.15	1.57	3.94
4	2	4	2	1.96	0.66	1.30	0.16	1.59	4.00
4	2	4	4	1.95	0.65	1.29	0.16	1.58	3.98

Table A. Effect of the number of layers in both FIM and TEDec.

Method	b-minFDE$_6$ (↓)	minADE$_6$ (↓)	minFDE$_6$ (↓)	MR$_6$ (↓)	minADE$_1$ (↓)	minFDE$_1$ (↓)	MR$_1$ (↓)
QCNet* [2]	1.91	0.65	1.29	0.16	1.69	4.30	<u>0.59</u>
FINet (Ours)*	1.93	0.66	1.27	0.15	1.58	3.97	0.57
Gnet[3]	1.90	0.69	1.34	0.18	1.90	4.40	0.18
TENET[4]	1.90	0.70	1.38	0.19	1.90	4.69	0.19
MacFormer [5]	1.90	0.70	1.38	0.19	1.90	4.69	0.19
QML [6]	1.95	0.69	1.39	0.19	1.84	4.98	0.62
BANet [7]	1.92	0.71	1.36	0.19	1.79	4.61	0.60
Forecast-MAE [8]	1.91	0.69	1.34	0.17	1.66	4.15	0.59
QCNet [2]	**1.78**	**0.62**	**1.19**	**0.14**	<u>1.56</u>	<u>3.96</u>	**0.55**
FINet (Ours)	<u>1.81</u>	**0.62**	**1.19**	**0.14**	**1.55**	**3.87**	<u>0.56</u>

Table B. Model ensemble results on Argoverse 2 dataset. We present the methods without model ensemble for reference, which are marked with the symbol "*". For each metric, the best result is in **bold**, and the second best result is <u>underlined</u>.

from extensive kernel optimizations (e.g., FlashAttention), whereas Mamba's operations may not yet be as optimized. As a result, the real-world speedup remains limited despite the substantial FLOP reduction. However, specialized hardware designed for state-space models may mitigate this issue by optimizing memory access patterns and accelerating sequential operations, making Mamba more computationally efficient in practice.

B. Additional Quantitative Results

In this section, to pursue a more comprehensive comparison, we provide additional quantitative results of the proposed FINet.

B.1. Model ensembling

We demonstrate the results of the model ensemble, which is a common technique to improve the accuracy of final predictions, on the Argoverse 2 dataset. The experimental results are shown in Table B. Specifically, seven models are trained using different random seeds, learning rates, and training epochs. A total of 42 future trajectories are predicted for each agent, which are then clustered into six centers using the k-means clustering algorithm. The final predicted trajectories are determined by averaging the trajectories within each cluster. From the experiment results, We observe that the proposed method outperforms previous approaches in overall performance. Given its superior efficiency and low latency, this model ensembling technique, combined with the proposed method, holds great promise for deployment in real-world applications, where previous methods often suffer from inefficiency.

B.2. More Ablation Study

B.2.1. Effectiveness of MambaBlock

We evaluate the effectiveness of different modules in each component of the proposed FINet, including the attention module, single-direction Mamba (Si-Mamba), and bidirectional Mamba (Bi-Mamba). The experimental results are presented in Tab. C. For historical trajectory encoding in the Lightweight Scene Encoder, we observe that both Si-Mamba and Bi-Mamba outperform the attention module by a significant margin. We attribute this to the Mamba block's suitability for sequential data processing, as its scanning mechanism explicitly injects inductive bias. Furthermore, Bi-Mamba achieves performance comparable to Si-Mamba. For efficiency consideration, Si-Mamba is selected in the proposed method. For spatial interaction modeling in the Future-Aware Interaction Mamba, we observe that Si-Mamba achieves performance similar to the attention module, while Bi-Mamba outperforms both. Consequently, Bi-Mamba is chosen in the proposed method, demonstrating the effectiveness of our design in adapting

	Method	b-minFDE$_6$ (↓)	minADE$_6$ (↓)	minFDE$_6$ (↓)	MR$_6$ (↓)	minADE$_1$ (↓)	minFDE$_1$ (↓)
LSEnc	Attn	1.98	0.69	1.32	0.17	1.74	4.25
	Si-Mamba	1.93	0.65	1.27	0.15	1.57	3.94
	Bi-Mamba	1.94	0.65	1.28	0.15	1.57	3.95
FIM	Attn	1.94	0.68	1.28	0.16	1.64	4.02
	Si-Mamba	1.98	0.66	1.32	0.16	1.60	4.04
	Bi-Mamba	1.93	0.65	1.27	0.15	1.57	3.94
TEDec	Attn	2.20	0.73	1.51	0.19	1.57	3.92
	Si-Mamba	1.94	0.65	1.29	0.16	1.58	3.99
	Bi-Mamba	1.93	0.65	1.27	0.15	1.57	3.94

Table C. We evaluate the effectiveness of different module in various components of the proposed method. Specifically, "LSEnc" refers to the Lightweight Scene Encoder, "FIM" represents the Future-Aware Interaction Mamba, and "TEDec" denotes the Temporal Enhanced Decoder. Additionally, "Attn" stands for the Attention Module, "Si-Mamba" refers to the single-direction Mamba module, and "Bi-Mamba" represents the bidirectional Mamba module. The module achieving the best performance is highlighted in **bold**.

Position	b-minFDE$_6$ (↓)	minADE$_6$ (↓)	minFDE$_6$ (↓)	MR$_6$ (↓)	minADE$_1$ (↓)	minFDE$_1$ (↓)
Start	1.93	0.65	1.27	0.15	1.57	3.94
End	1.97	0.66	1.30	0.16	1.59	4.00

Table D. Inductive bias position.

the Mamba architecture for spatial interaction modeling. Finally, we evaluate future trajectory refinement in the Temporal Enhanced Decoder. A similar conclusion is drawn: the Mamba block excels over the attention module in sequential data modeling, as indicated by a clear performance gap. However, Bi-Mamba demonstrates a slight performance advantage over Si-Mamba and is therefore chosen as the final design.

We observe that Si-Mamba excels in encoding trajectory data, while Bi-Mamba is better suited for spatial interaction modeling and trajectory refinement. We hypothesize that the latter task demands a more comprehensive understanding of the entire input data, making Bi-Mamba a more appropriate choice.

B.2.2. Inductive bias position

The effectiveness of the inductive bias, as demonstrated in the main paper, plays a crucial role in accurately predicting the future trajectory. We found that removing the inductive bias or applying it to all future trajectory tokens results in suboptimal performance. Here, we verify the optimal position for incorporating it. In our experiments, we apply the inductive bias either to the first future trajectory token or to the last future trajectory token, with the results presented in Tab. D. Our observations indicate that the former yields significantly better performance. We hypothesize that placing the inductive bias at the beginning has the greatest impact on subsequent tokens due to the scanning mechanism, whereas placing it at the end substantially diminishes this effect.

B.2.3. Supervision on predicted offset

We investigate the effectiveness of aligning the predicted endpoint with the future ground truth trajectory endpoint, with the experimental results presented in the tab. E. Our observations indicate that learning both the first and second endpoints without any supervision already yields strong performance. However, applying the alignment strategy to the first predicted endpoint has minimal impact on performance. In contrast, applying alignment strategy to the second predicted endpoint leads to noticeable improvements on the $K = 6$ metrics and significant gains on the $K = 1$ metrics. This suggests that at later stages, tokens near the future trajectory endpoint should have a greater impact, which is achieved by placing these tokens at the end of the sequence in ARS, whereas earlier stages should explore more freely and focus on capturing general information. Finally, applying the alignment strategy to both predicted endpoints does not lead to further performance improvement, which further verify the above hypothesis.

B.2.4. FIM Scan Order

We explore the scan order of the Bi-Mamba block in Future-Aware Interaction Mamba (FIM), where the tokens are already reordered using the adaptive reorder strategy (ARS). The experimental results are presented in Tab. F. For clarity, we visualize the scan order in Fig. A, where "Decreasing" refers to scanning the sequence in order of decreasing distance (adopted in our proposed method), while "Increasing" scans the sequence in order of increasing distance. Our observations show that the "Decreasing" order consistently achieves better overall performance compared to "Increasing." This suggests that information should first propagate effectively from distant to closer points along the future trajectory endpoints and then propagate back, rather than the other way around.

RP1	RP2	b-minFDE$_6$ (↓)	minADE$_6$ (↓)	minFDE$_6$ (↓)	MR$_6$ (↓)	minADE$_1$ (↓)	minFDE$_1$ (↓)
		1.94	0.65	1.28	0.16	1.62	4.10
✓		1.94	0.66	1.29	0.16	1.62	4.09
	✓	1.93	0.65	1.27	0.15	1.57	3.94
✓	✓	1.94	0.66	1.28	0.15	1.58	3.96

Table E. Supervision on predicted offset.

Method	b-minFDE$_6$ (↓)	minADE$_6$ (↓)	minFDE$_6$ (↓)	MR$_6$ (↓)	minADE$_1$ (↓)	minFDE$_1$ (↓)
Increasing	1.94	0.65	1.28	0.16	1.59	3.98
Decreasing	1.93	0.65	1.27	0.15	1.57	3.94

Table F. FIM Scan Order.

B.2.5. Future Trajectory Interpolation

Finally, we investigate interpolating future trajectories with and without the current state $\mathcal{ST}_0^{scene}$. The results in Tab. G show that interpolated future trajectories starting from the current state (with $\mathcal{ST}_0^{scene}$) not only align with intuition but also achieve overall better performance, particularly on minFDE$_1$. This demonstrates the effectiveness of our design.

C. Broader Impact and Limitations

C.1. Broader Impact

Our work leverages the highly efficient State Space Model, specifically Mamba, for motion forecasting in autonomous driving. By carefully adapting the Mamba architecture to effectively model both spatial and temporal information, our approach enables precise and reliable future trajectory prediction while maintaining computational efficiency. This efficiency is particularly crucial for real-world autonomous driving applications, where onboard computational resources are often constrained, making fast and accurate motion forecasting essential for safe and responsive navigation.

C.2. Potential Limitations

Despite the advancements of the proposed method, certain limitations remain. There is still significant room for performance improvement, particularly in handling complex scenarios where the method may struggle to generate sufficiently accurate or diverse trajectories. Additionally, the predicted trajectories at consecutive timestamps may lack temporal consistency. In future work, we aim to refine both historical and current predictions to enhance overall performance and ensure smoother, more coherent trajectory predictions over time.

D. Public Resource Used

In this section, we acknowledge the use of the following public resources, during this work:

- Pytorch [3] ..... Pytorch License
- QCNet [4] ..... Apache License 2.0
- argoverse-api [5] ..... Apache License 2.0
- av2-api[6] ..... MIT License
- forecast-mae [7] ..... MIT License
- pytorch-OpCounter[8] ..... MIT License

E. More Qualitative Visualization.

More qualitative visualizations are shown in Fig. B, which further demonstrate the proposed method can produce more accurate and diverse future trajectories.

References

[1] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 1

[2] Zikang Zhou, Jianping Wang, Yung-Hui Li, and Yu-Kai Huang. Query-centric trajectory prediction. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 17863–17873, 2023. 2, 5

[3] Xing Gao, Xiaogang Jia, Yikang Li, and Hongkai Xiong. Dynamic scenario representation learning for motion forecasting with heterogeneous graph convolutional recurrent networks. *IEEE Robot. Autom. Lett.*, 8(5):2946–2953, 2023. 2

[4] Yuting Wang, Hangning Zhou, Zhigang Zhang, Chen Feng, Huadong Lin, Chaofei Gao, Yizhi Tang, Zhenting Zhao, Shiyu Zhang, Jie Guo, et al. Tenet: Transformer encoding network for effective temporal flow on motion prediction. *arXiv preprint arXiv:2207.00170*, 2022. 2

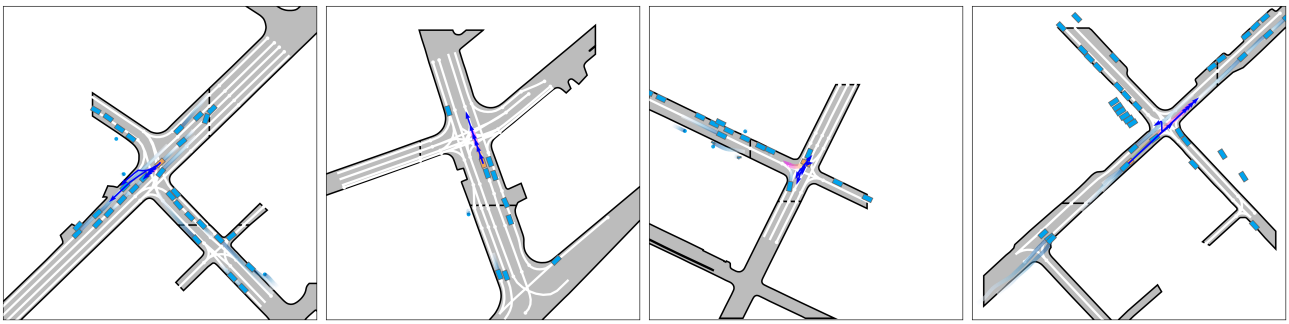
[5] Chen Feng, Hangning Zhou, Huadong Lin, Zhigang Zhang, Ziyao Xu, Chi Zhang, Boyu Zhou, and Shaojie Shen. Macformer: Map-agent coupled transformer for real-time and robust trajectory prediction. *IEEE Robot. Autom. Lett.*, 2023. 2

[6] Tong Su, Xishun Wang, and Xiaodong Yang. Qml for argoverse 2 motion forecasting challenge. *arXiv preprint arXiv:2207.06553*, 2022. 2

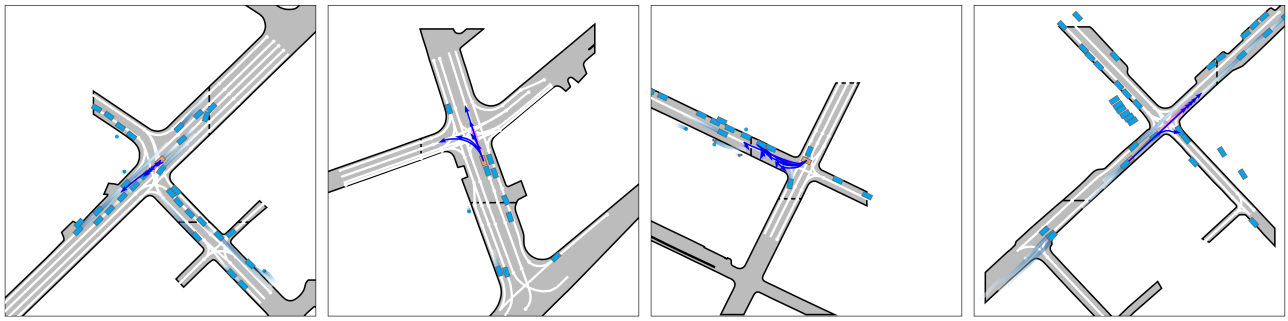
[3] https://github.com/pytorch/pytorch
[4] https://github.com/ZikangZhou/QCNet/tree/main
[5] https://github.com/argoverse/argoverse-api
[6] https://github.com/argoverse/av2-api
[7] https://github.com/jchengai/forecast-mae
[8] https://github.com/Lyken17/pytorch-OpCounter

$\mathcal{ST}_0^{scene}$	**b-minFDE**$_6$ (↓)	**minADE**$_6$ (↓)	**minFDE**$_6$ (↓)	**MR**$_6$ (↓)	**minADE**$_1$ (↓)	**minFDE**$_1$ (↓)
w/o	1.95	0.65	1.29	0.16	1.60	4.03
with	1.93	0.65	1.27	0.15	1.57	3.94

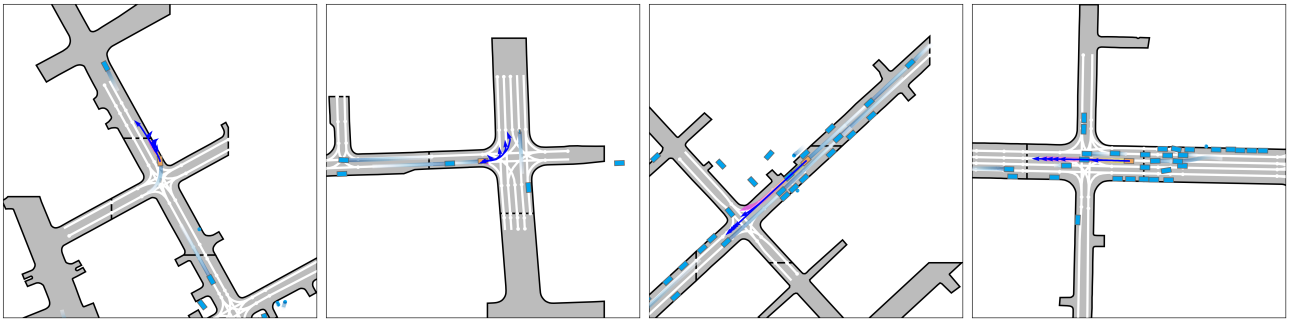
Table G. Future trajectory interpolation.



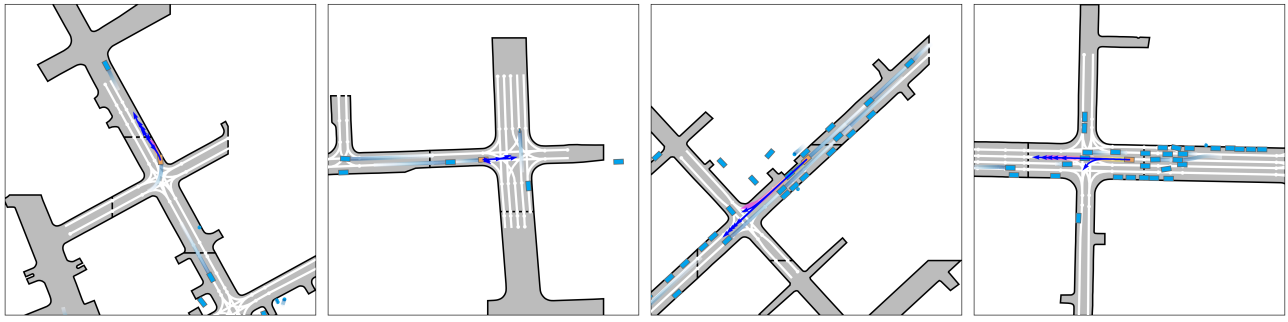
(a) QCNet[2]



(b) FINet (Ours)



(c) QCNet[2]



(d) FINet (Ours)

Figure B. More qualitative visualization. We compare FINet to the state-of-the-art method, QCNet [2]. Blue arrows represent the predicted future trajectories (K=6), while the pink arrow denotes the ground truth future trajectory. The orange bounding box indicates the focal agent, while the blue bounding boxes denote surrounding agents.

[7] Chen Zhang, Honglin Sun, Chen Chen, and Yandong Guo. Banet: Motion forecasting with boundary aware network. *arXiv preprint arXiv:2206.07934*, 2022. 2

[8] Jie Cheng, Xiaodong Mei, and Ming Liu. Forecast-mae: Self-supervised pre-training for motion forecasting with masked autoencoders. In *Int. Conf. Comput. Vis.*, pages 8679–8689, 2023. 2