Fast Inference and Update of Probabilistic Density Estimation on Trajectory Prediction
—–Supplementary Material—–

Takahiro Maeda   Norimichi Ukita
Toyota Technological Institute, Japan
{sd21601, ukita}@toyota-ti.ac.jp

A. Comparison with Flomo [7]

Although Flomo [7] is the closest work to our FlowChain, we didn't include the results of Flomo in Sec.4, because the reproducible training code or pretrained model is unavailable. Furthermore, Flomo reports the scores based on the train-validation-test splits different from ours. Thus, we made a fair comparison along with the evaluation in the Social-GAN [3] paper on *ETH/UCY* dataset and Trajnet [5] paper on *SDD* dataset. We used the official implementation of Flomo for comparison. Note that KDE is applied to the generated trajectories of Flomo for estimating temporal densities, as mentioned in Sec. 2 of the main paper.

The results of trajectory prediction accuracy are shown in Tab. 5 and Tab. 6. Our FlowChain achieves better results than Flomo on most splits on *ETH/UCY* and *SDD*.

The accuracy and computational time of density estimation of Flomo are compared with FlowChain as shown in Tab. 7. Flomo takes a much longer time for density estimation (i.e., 12397.4 milliseconds) because Flomo needs KDE for temporal density estimation. Furthermore, the accuracy of density estimation of Flomo is lower than that of FlowChain over EMD on *Simfork* dataset (i.e, 1,834 vs. 1.408) and log-probability on *ETH/UCY* dataset (i.e, -24.05 vs. -0.26). We also visualize the estimated densities of Flomo and FlowChain in Fig. 9 and Fig. 10. On *Simfork* dataset, Flomo estimated too large and elliptical-shape densities, unlike the tight round-shape ground truth distribution, as shown in the first row of Fig. 9. On *ETH/UCY* dataset also, Flomo could not suggest tight densities. These poor performances on density estimation are the limitations of KDE on limited samples.

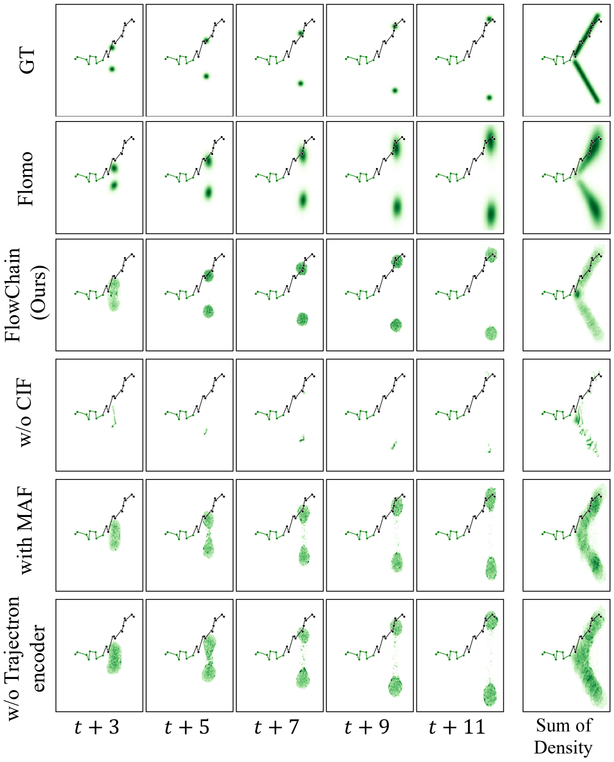

Figure 9. **Temporal estimated densities of Flomo and ablated models on *Simfork* dataset.**

B. Effectiveness of the FlowChain Architecture

We validated the effectiveness of the FlowChain architecture by changing each component. We replace our three choices: CIF [1], RealNVP [2], and Trajectron [6] encoder. As mentioned in Sec. 3 of the main paper, we use conditional CIFs as the base normalizing flow model for better expressive power. For replacing CIF, vanilla conditional normalizing flow models are employed, denoted by "w/o CIF". As mentioned in Sec. 4 of the main paper, we use a three-layer RealNVP inside the CIF and Trajectron++ encoder for a temporal-social encoder. For replacing RealNVP choice, we choose the MAF [4] instead of RealNVP, denoted by "with MAF". For replacing the Trajectron++ encoder, one transformer layer is applied for encoding the observed past trajectory, denoted by "w/o Trajectron encoder".

Table 5. **Quantitative comparison on *ETH/UCY* dataset with Best-of-20 metrics.** Scores are in meters. Lower is better.

Method	**ETH**		**HOTEL**		**UNIV**		**ZARA1**		**ZARA2**		**Mean**	
	ADE	FDE	ADE	FDE	ADE	FDE	ADE	FDE	ADE	FDE	ADE	FDE
Flomo [7]	0.58	1.02	0.34	0.63	**0.29**	**0.52**	0.22	**0.39**	0.28	0.53	0.34	0.62
FlowChain (Ours)	**0.55**	**0.99**	**0.20**	**0.35**	**0.29**	0.54	**0.22**	0.40	**0.20**	**0.34**	**0.29**	**0.52**
FlowChain w/o CIF	0.70	1.35	0.27	0.53	0.33	0.67	0.23	0.45	**0.20**	0.39	0.35	0.68
FlowChain with MAF	0.57	1.07	0.22	0.37	**0.29**	0.55	0.23	0.43	0.21	0.40	0.30	0.56
FlowChain w/o Trajectron encoder	0.62	1.23	0.34	0.72	0.33	0.62	0.23	0.42	0.21	0.39	0.35	0.68

Table 6. **Quantitative comparison on *SDD* dataset with Best-of-20 metrics.** Scores are in pixels. Lower is better.

Method	ADE	FDE
Flomo [7]	10.78	17.36
FlowChain (Ours)	9.93	17.17
FlowChain w/o CIF	**9.70**	**17.13**
FlowChain with MAF	14.58	24.70
FlowChain w/o Trajectron encoder	18.92	30.94

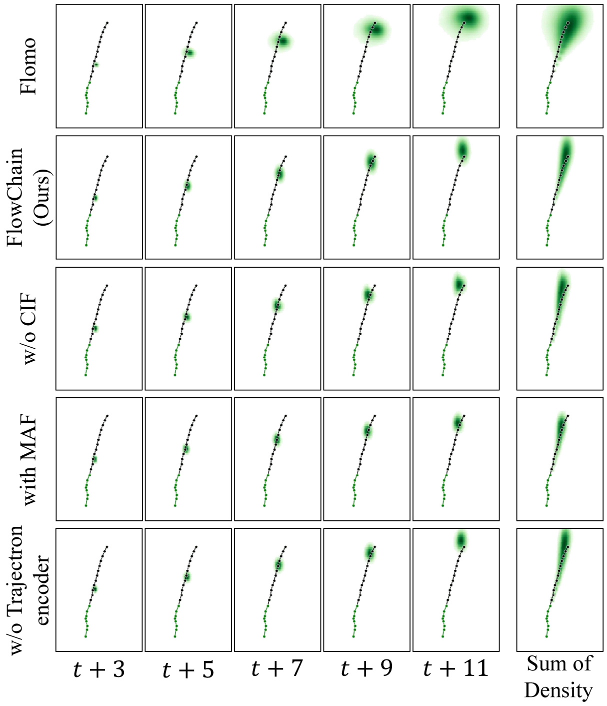

Figure 10. **Temporal estimated densities of Flomo and ablated models on *ETH/UCY* dataset.**

Note that this transformer layer encodes each trajectory independently and thus cannot account for the social interactions.

The results of trajectory prediction accuracy are shown in Tab. 5 and Tab. 6. By replacing CIF, the accuracy degrades on *ETH/UCY* but gets slightly better on *SDD*. This is because the multi-modal densities are less observed in *SDD* and thus models without CIF still fit the *SDD* dataset. However, as shown in Fig. 9, models without CIF do not fit the multi-modal density and estimate thin densities. We further confirmed the poor accuracy of density estimation without CIF by comparing the accuracy of density estimation, as shown in Tab. 7. We choose conditional CIFs as the base normalizing flow models because our main focus is on better and faster density estimation. As mentioned in Sec. 3.2 of the main paper, we can see a little fluctuation in the estimated densities by models with CIFs. However, this fluctuation can be ignored because estimated densities are smooth enough especially for the real data like *ETH/UCY* dataset, as shown in Fig. 10.

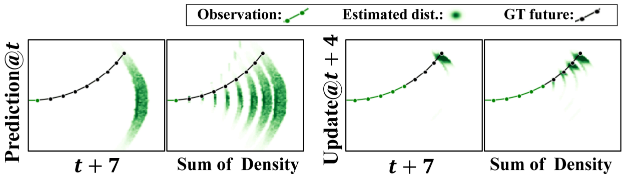

Figure 11. **Temporal estimated densities on the synthesized unicycle motion dataset.**

By replacing RealNVP, the accuracy degrades on both *ETH/UCY* and *SDD*. We find RealNVP a good fit for our chaining architecture.

By replacing the Trajectron encoder, the accuracy degrades especially on *SDD* because the simple transformer encoder cannot consider the social interactions as mentioned above. Furthermore, the Trajctron encoder needs less computational cost than a simple transformer encoder despite accounting for the social interactions, as shown in Tab. 7.

C. Experiment on Unicycle Motion Dataset.

Our Simfork dataset, which only has simple bifurcations, cannot fully show the advantage of our method over Gaussian Mixture Models. Therefore, we created a synthesized dataset based on the unicycle motion model. As shown in Fig 11, FlowChain successfully captures the challenging banana-like shape distributions and produces reliable updates, which is hard for a mixture Kalman filter assuming Gaussian distributions.

Table 7. **Accuracy and computational time comparison of density estimation.** Scores are in milliseconds for computational time. While lower is better for EMD on *Simfork* dataset, higher is better for log-probablity on *ETH/UCY* dataset. We report the averaged EMD and log-probability over the 12 prediction steps.

Method	**Comp. Time**		**Accuracy of Density Estimation**	
	All	**w/o KDE**	**EMD ↓**	**log-probability ↑**
Flomo [7]	12397.4	24.2	1.834	-24.05
FlowChain (Ours)	37.0	-	**1.408**	**-0.26**
FlowChain w/o CIF	26.7	-	3.313	-9.48
FlowChain with MAF	44.2	-	2.475	-0.55
FlowChain w/o Trajectron encoder	38.7	-	2.369	-0.33

References

[1] Rob Cornish, Anthony Caterini, George Deligiannidis, and Arnaud Doucet. Relaxing bijectivity constraints with continuously indexed normalising flows. In *ICML*, pages 2133–2143, 2020.

[2] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *ICLR*, 2017.

[3] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social gan: Socially acceptable trajectories with generative adversarial networks. In *CVPR*, pages 2255–2264, 2018.

[4] George Papamakarios, Theo Pavlakou, and Iain Murray. Masked autoregressive flow for density estimation. *NeurIPS*, 30, 2017.

[5] Amir Sadeghian, Vineet Kosaraju, Agrim Gupta, Silvio Savarese, and Alexandre Alahi. Trajnet: Towards a benchmark for human trajectory prediction. *arXiv preprint*, 2018.

[6] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *ECCV*, pages 683–700, 2020.

[7] Christoph Schöller and Alois Knoll. Flomo: Tractable motion prediction with normalizing flows. In *IROS*, pages 7977–7984, 2021.