TrajFlow: Learning the Distribution over Trajectories

Anna Mészáros*, Javier Alonso-Mora, and Jens Kober

***Abstract*— Predicting the future behaviour of people remains an open challenge for the development of risk-aware autonomous vehicles. An important aspect of this challenge is effectively capturing the uncertainty inherent to human behaviour. This paper studies an approach for multi-modal probabilistic motion forecasting of an agent with improved accuracy in the predicted sample likelihoods. Our approach achieves state-of-the-art results on the inD dataset when evaluated with the standard metrics employed for motion forecasting. Furthermore, our approach also achieves state-of-the-art results when evaluated with respect to the likelihoods it assigns to its generated trajectories. Evaluations on artificial datasets indicate that the distributions learned by our model closely correspond to the true distributions observed in data and are not as prone to being over-confident in a single outcome in the face of uncertainty.**

I. Introduction

As we strive towards developing autonomous agents that are active in human-centered environments, we should provide our agents with the means to understand their environment and anticipate how it may change in the future. Of particular interest is predicting the future trajectories of people – from pedestrians and cyclists to road agents – to improve motion planning algorithms. However, most times the trajectories of people are not deterministic, instead they tend to take on a probabilistic form, sometimes involving complex and multi-modal distributions. An example of such a multi-modal distribution can be seen when vehicles approach an intersection like in Fig. 1. In such a case, one can consider the three possible options of going straight, left, or right as the modes of the distribution. These are the most obvious high-level modes one might expect to see. However, there can also be other distinct modes present in data which may not be as obvious and may get overlooked by methods which rely on a predefined number of modes.

Ideally, when learning a distribution over the expected motion of people in a scene, the learned model should have the following capabilities:

1) it should be able to capture all the expected modes.
2) It should be real-time capable, and
3) it should be able to assign a meaningful likelihood to the samples it generates.

By having all these aspects, these models have the potential to provide more informative predictions to motion planners for uncertain, dynamic environments.

This research is supported by NWO-NWA project "Acting under uncertainty" (ACT), NWA.1292.19.298.

The authors are with Cognitive Robotics, Delft University of Technology, The Netherlands (e-mail: a.meszaros, j.alonsomora, j.kober@tudelft.nl).

*Corresponding Author

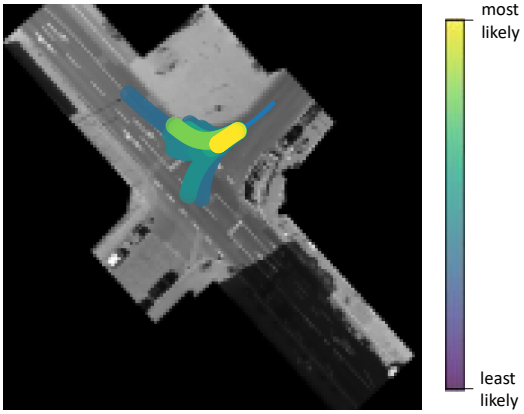

Fig. 1: Predictions of TrajFlow. The colour of each trajectory corresponds to its likelihood. The thin blue line corresponds to the past trajectory.

Out of the existing approaches, a great deal of focus has gone into providing predictions of distributions at individual time steps. These kinds of predictions can, however, lead to more conservative strategies within the subsequent motion planning [1]. For this reason, it would be more beneficial to capture the distribution over trajectories as a whole.

At the same time, current datasets (e.g. ETH [2]/UCY [3], inD [4], Argoverse [5], etc.) do not provide the means for systematically evaluating the performance of learned models in terms of the distributions that they predict. The issue lies in the fact that within existing datasets, there is only a single sample of the ground truth future per situation. This makes it increasingly difficult to determine the extent to which probabilistic predictions reflect the underlying distributions present within these datasets.

The contributions of this paper are two-fold. As a first step, we introduce an architecture for learning a prediction model that has the three previously mentioned capabilities. To enable an effective learning of a given distribution over complex and multivariate trajectories we introduce an intermediate representation of the trajectories using a Recurrent Neural Network Autoencoder. This intermediate representation captures the most relevant features of the trajectories and in turn also simplifies the learning of the underlying distribution. Secondly, we evaluate our architecture against a baseline method with focus on their likelihood predictions. Through this, we demonstrate why there is a need for more focus on the learned distributions and structured datasets for the evaluation of probabilistic models.

II. Related Work

Several methodologies for predicting distributions over traffic agents' future motions have been proposed. Gen-

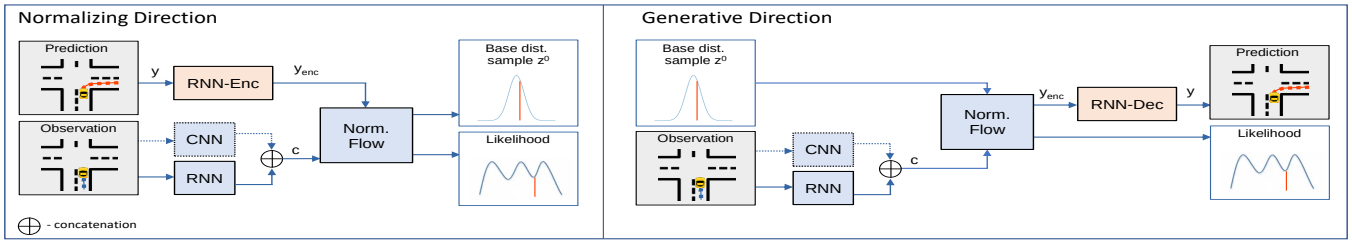

Fig. 2: TrajFlow architecture overview for both directions of the normalizing flow. In the normalizing direction, the NF transforms the encoded future trajectory into a sample within its base distribution and returns the likelihood of the encoded trajectory. In the generative direction, the NF generates encoded future trajectories, which are then decoded, and also provides the likelihood of the encoded trajectory.

erative networks such as Generative Adversarial Networks (GANs) [6], [7] and networks based on Variational Autoencoders (VAE) such as Conditional VAEs (CVAEs) [8], [9] and Variational Recurrent Neural Networks (VRNNs) [10], [11] are particularly interesting as they have the potential to learn complex distributions without specifying the number of expected modes. Nevertheless, neither of the two enable an exact calculation of the likelihoods of generated samples.

Recently, new promising approaches based on Normalizing Flows (NFs) [12]–[16] have been introduced for learning the complex distributions over agents' future motions. While the above NF methods already demonstrate good qualitative results in predicting multiple future trajectories, they have yet to make full use of the exact likelihoods inferred by the learned models. In this paper we propose an architecture which makes use of encoded future motions to facilitate the learning of the underlying distributions present in data. This results in a model which is able to provide likelihoods that are not only relevant for ranking generated samples, but likelihoods that are also in line with the likelihoods of the true underlying probability distribution function.

III. Method

A. Normalizing Flows

Normalizing Flows [17], [18] constitute a family of generative methods which enable exact likelihood computation. They are based on the concept of transforming complex distributions through a series of differentiable bijective functions into a simple distribution for which the probability distribution function is known – most commonly a standard normal distribution. For an extensive overview of flow models, we refer the reader to [19].

Our method employs the same form of NF structure as the one used in FloMo [13]. In FloMo this architecture is employed for learning distributions directly on two-dimensional trajectories. Within TrajFlow we instead generate a lower dimensional representation of the trajectories. FloMo can then be seen as a specific version of TrajFlow, where the encoder and decoder are identity functions. Below we present two important aspects of the theory behind NFs which are relevant for the reasoning behind our architecture.

1) Sampling and Likelihood Evaluation: To obtain a sample from the desired complex distribution over outputs $\mathbf{Y}$, one can sample from the base distribution and propagate it in the generative direction. The Probability Density Function (PDF) of the complex distribution can then also be obtained in terms of the PDF of the base distribution as follows:

$$\begin{aligned} p_y(\mathbf{y}) &= p_z(F(\mathbf{y}))|\det J_F(\mathbf{y})| \\ &= p_z(G^{-1}(\mathbf{y}))|\det J_{G^{-1}}(\mathbf{y})| \\ &= p_z(\mathbf{z})|\det J_G(\mathbf{z})|^{-1}, \end{aligned} \tag{1}$$

where $\mathbf{z}$ is a sample from the base distribution and J_G is the Jacobian of the generative transformation G. The absolute determinant of the Jacobian $|\det J_G(\mathbf{z})|$ quantifies the relative change of volume within a small neighbourhood of $\mathbf{z}$ when transforming it to a sample $\mathbf{y}$ using G.

2) Training Normalizing Flows: In order to learn the parameters of the NF, the KL-divergence can be employed for minimizing the distance between the target distribution $p_y^*(\mathbf{y})$ and the distribution learned by the model $p_y(\mathbf{y})$:

$$\begin{aligned} \mathcal{L} &= D_{\mathrm{KL}}[p_y^*(\mathbf{y})||p_y(\mathbf{y})] \\ &= -\mathbb{E}_{p_y^*(\mathbf{y})}[\log p_z(F(\mathbf{y})) + \log|\det J_F(\mathbf{y})|] \\ &\approx -\frac{1}{N}\sum_{n=1}^{N} \log p_z(F(\mathbf{y}_n)) + \log|\det J_F(\mathbf{y}_n)|, \end{aligned} \tag{2}$$

where N refers to the number of samples in a training batch.

B. Network Architecture

An overview of the TrajFlow architecture can be seen in Fig. 2. What makes our architecture unique and better equipped at learning meaningful likelihoods of a distribution is the introduction of a Recurrent Neural Network Autoencoder (RNN-AE), shown in orange, for learning an encoded representation $\mathbf{y}_{\mathrm{enc}}$ of the future trajectory $\mathbf{y}$.

1) The RNN-Autoencoder: is based on a Gated Recurrent Unit (GRU). Before passing the trajectory $\mathbf{y}$ to the GRU, it is embedded by a linear layer into a matrix of the size $T_{\mathrm{pred}} \times \mathrm{em\_size}$. The output of the GRU is passed through a linear layer which transforms the last dimension of the layer's input into $\mathrm{enc\_size}$. To obtain a compact representation of the form $1 \times \mathrm{enc\_size}$ we take the final row of the output. This compact representation is then provided to the NF for learning the distribution in the encoded space.

In order to decode the predicted encodings, we employ an auto-regressive decoder structure. The encoding is passed through a linear layer, which retains the dimensions of the input, before being passed through a GRU. Finally the output

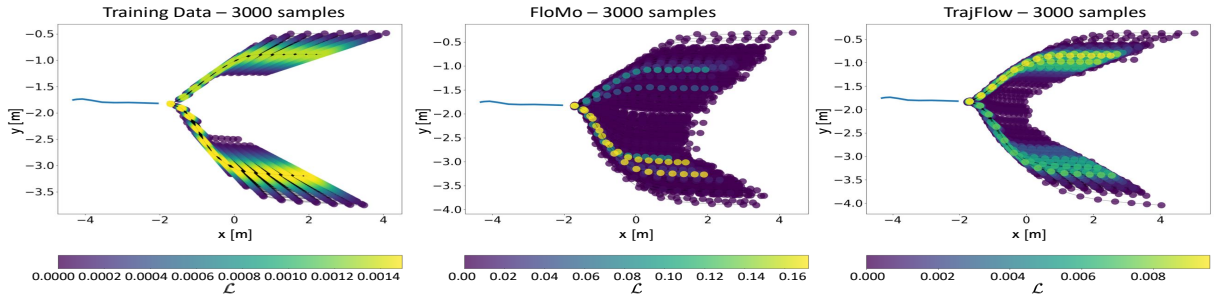

Fig. 3: Trajectory distributions for the case of $s \sim \mathcal{N}(\mu = 1, \sigma = 0.15)$. On the left is the training distribution consisting of 3000 samples. The remaining two images are trajectories with their likelihoods as generated by FloMo and TrajFlow respectively.

is passed through a linear layer which transforms the output into a vector of the form 1×2. This corresponds to the predicted position of a single time step. The final hidden output of the GRU is then taken as the new input to the initial linear layer, and the decoding procedure is repeated as many times as the number of desired time steps T_{pred}.

IV. Experiments

We test our proposed method on two toy datasets in which we know the expected distributions of the future trajectories in order to test the capabilities of our model for predicting sample likelihoods. Additionally, we test our method on real data with the inD dataset [4], which is one of the standard datasets used for motion forecasting. This evaluation is primarily useful for verifying that the introduction of the RNN-AE does not lead to a performance degradation in the trajectory predictions themselves. FloMo is used as a baseline and tests are performed using code available on GitHub [1]. Below are some highlights of our results. For more detailed experimental results and implementation details we would like to refer our readers to [20].

A. Evaluation on Generated Bi-Modal Datasets (Case 1)

In order to evaluate the capability of our model to capture the underlying distribution of observed data, we construct a collection of bi-modal datasets. As a basis for generating the distribution within each mode, we recorded two trajectories with distinct directions and a future segment of 14 time steps. These segments were then scaled with a factor s sampled from a desired distribution. This way, we obtain a distribution of trajectories. To obtain the true distribution function, we fit a D-dimensional Gaussian to each mode, where D is $2T_{\text{pred}}$. For the past, we use a single observation of 10 time steps.

We perform this test both using our architecture and the FloMo architecture. For both models we evaluate the KL-Divergence using $N = 100$ samples (see Fig. 4). We can observe that FloMo exhibits a noticeably higher divergence from the true distribution, with the divergence value on the sampled trajectories generally being above 6, compared to our approach which shows divergence values of around 1 to 1.5. Similarly, the KL-Divergence between the training

[1] https://github.com/cschoeller/flomo_motion_prediction

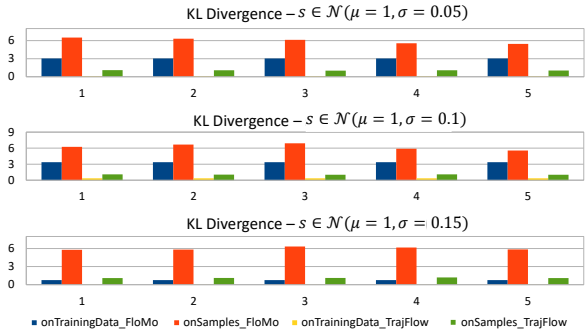

Fig. 4: KL-Divergence of FloMo and TrajFlow on the bi-modal trajectory distributions. The 5 sets of bars correspond to 5 sets of 100 samples; relevant for the results on the sampled data. 'On training data' and 'on samples' refer to evaluations of the likelihoods that the models assign to the original training data and their own generated trajectories respectively.

distribution and the distribution fitted over the training samples by FloMo is noticeably higher than that of our model. A qualitative look at the predictions is provided in Fig. 3. Overall, our approach is able to better capture the underlying distribution over the trajectories.

B. Evaluation on Artificially Generated Datasets (Case 2)

The second collection of datasets consists of trajectories with three distinct possible directions, i.e. modes – straight, left, and right. All trajectories start by moving straight and may branch off left or right. At 30 specific time points along the trajectory which the agent chooses to follow, the prediction model leverages the previous 8 time steps of observations to forecast the subsequent 22 time steps. The percentage of trajectories going in each direction from a given current point is taken as the ground truth likelihood.

Over the 6 scenes we observe a strong tendency towards a single mode within the predicted likelihoods of the FloMo model (see Fig. 5 for example). This is the case even when there are no features within the past trajectory by which to discriminate the possible future outcomes, such as at the very beginning of the sequence. In the case of our model, the summed likelihoods within a mode more closely capture the expected distributions of the trajectories. Fig. 5 shows an example of the predicted likelihoods for each of the three modes as the agent follows a straight trajectory.

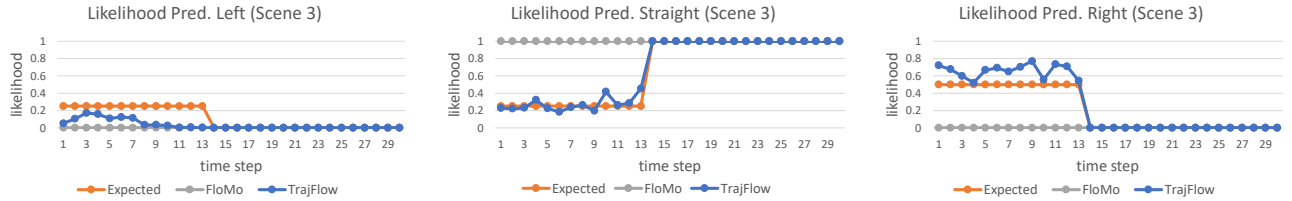

Fig. 5: Likelihood Predictions for all three modes for one of the scenes in the second artificial dataset.

Model	1s Err	2s Err	3s Err	4s Err	5s Err
HBA-Flow [12]	0.19	0.44	0.82	1.21	1.74
FloMo [13] (all classes)	0.07	0.18	0.30	0.45	0.73
TrajFlow (Ours) (all classes)	**0.06**	**0.13**	**0.21**	**0.35**	**0.57**
FloMo [13] (only vehicles)	0.10	0.26	0.44	0.71	1.15
TrajFlow (Ours) (only vehicles)	**0.09**	**0.17**	**0.30**	**0.50**	**0.82**

TABLE I: Five-fold cross-validation results on inD dataset. Expresses the Euclidean error (Err) per second of the top 10% samples out of 50 samples.

C. Evaluations on inD

For comparability, we perform the evaluations with $1/5$ resolution in accordance to [12]. For encoding the context we make use of both the RNN encoder for the past trajectory, and the CNN encoder for the static scene information (see Fig. 2). The results can be seen in Tab. I. We include the results of our model and FloMo both when trained and tested on all the classes, which include pedestrians, bicycles, cars, trucks, and busses as well as when only vehicles (cars, trucks, and busses) are considered. The reason is that, since motorized vehicles tend to move faster, the largest errors that will tend to be observed in the evaluations will be within those classes. We compare our results to those of HBA-Flow, presented in [12]. It can be seen that our method outperforms state-of-the-art methods in both cases.

Qualitatively, we observe that our method can capture the different possible future trajectories. To further examine the quality of our model's predictions we compare them to an estimated distribution of vehicle trajectories within the dataset. The distribution of the trajectories is obtained by segmenting the scenes based on the road a vehicle is coming from and observing whether the vehicle goes left, right, straight or is idle in the future which gives us 4 relevant modes. For each road we determine the percentage of future trajectories belonging to each of the modes. To compare to the trajectories in the dataset we employ the same approach on the predictions of our model. Fig. 6 shows an example case. We observe that the predictions of our model tend to gravitate around the percentages seen in the dataset, while FloMo tends to have a bias towards going straight. Whether high confidence in a single outcome is desired in specific cases is difficult to discern from the given dataset. It is safe to say that if this is a general bias across most of the cases from a road in which other possible motions have a non-negligible likelihood of occurring, which Fig. 6 points to, the model runs the risk of disregarding the remaining motions.

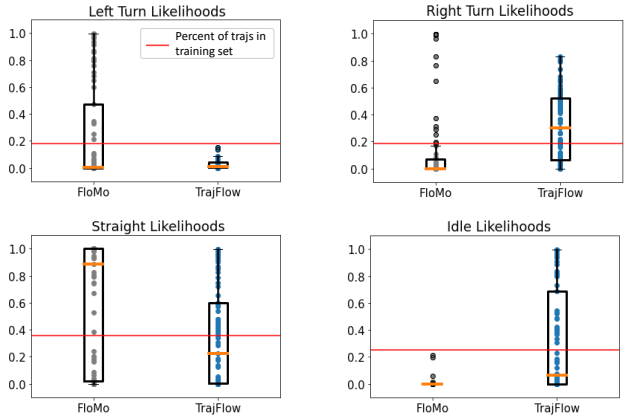

Fig. 6: Boxplots of the predicted likelihoods per mode for the scene in Fig. 1 with vehicles approaching the intersection from the northeast road.

V. Conclusions

In this work we investigated a motion forecasting approach based on Normalizing Flows. We were able to corroborate that models based on NFs can capture multivariate and multi-modal distributions solely from data without prior assumptions over the number of expected modes; something which has been shown in related works on NF-based models. Secondly, we proposed an approach which better captures the underlying distributions of the data, resulting in better likelihood values of generated samples.

To showcase the true potential of our approach, however, we had to devise simulated test cases. Current datasets are focused towards capturing naturalistic motion data from human traffic participants. As such, these datasets inherently have only a single deterministic ground truth. Nuances in the past trajectories may provide a model enough information to provide a single confident prediction for a given situation. This may not always be the case, however, and overconfidence in a single prediction can end up being detrimental to downstream motion planning tasks. If there is only a single ground truth trajectory on which to evaluate the predictions of a model it becomes virtually impossible to determine whether high confidence in a prediction is indeed desired or if a model is inherently prone to generate biased predictions. An important matter which remains to be addressed is the establishment of diverse and realistic datasets for evaluating probabilistic motion forecasting methods.

REFERENCES

[1] L. Janson, E. Schmerling, and M. Pavone, "Monte carlo motion planning for robot trajectory optimization under uncertainty," in *Robotics Research*. Springer, 2018, pp. 343–361.

[2] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," in *2009 IEEE 12th international conference on computer vision*. IEEE, 2009, pp. 261–268.

[3] A. Lerner, Y. Chrysanthou, and D. Lischinski, "Crowds by example," in *Computer graphics forum*, vol. 26, no. 3. Wiley Online Library, 2007, pp. 655–664.

[4] J. Bock, R. Krajewski, T. Moers, S. Runde, L. Vater, and L. Eckstein, "The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections," in *2020 IEEE Intelligent Vehicles Symposium (IV)*, 2020, pp. 1929–1934.

[5] M.-F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan *et al.*, "Argoverse: 3d tracking and forecasting with rich maps," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 8748–8757.

[6] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social gan: Socially acceptable trajectories with generative adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2255–2264.

[7] J. Amirian, J.-B. Hayet, and J. Pettré, "Social ways: Learning multi-modal distributions of pedestrian trajectories with gans," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.

[8] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data," in *European Conference on Computer Vision*. Springer, 2020, pp. 683–700.

[9] A. Bhattacharyya, D. O. Reino, M. Fritz, and B. Schiele, "Euro-pvi: Pedestrian vehicle interactions in dense urban centers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 6408–6417.

[10] B. Brito, H. Zhu, W. Pan, and J. Alonso-Mora, "Social-vrnn: one-shot multi-modal trajectory prediction for interacting pedestrians," *arXiv preprint arXiv:2010.09056*, 2020.

[11] A. Bertugli, S. Calderara, P. Coscia, L. Ballan, and R. Cucchiara, "Ac-vrnn: Attentive conditional-vrnn for multi-future trajectory prediction," *Computer Vision and Image Understanding*, vol. 210, p. 103245, 2021.

[12] A. Bhattacharyya, C.-N. Straehle, M. Fritz, and B. Schiele, "Haar wavelet based block autoregressive flows for trajectories," in *DAGM German Conference on Pattern Recognition*. Springer, 2020, pp. 275–288.

[13] C. Schöller and A. Knoll, "Flomo: Tractable motion prediction with normalizing flows," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst.* IEEE, 2021, pp. 7977–7984.

[14] N. Rhinehart, K. M. Kitani, and P. Vernaza, "R2p2: A reparameterized pushforward policy for diverse, precise generative path forecasting," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 772–788.

[15] N. Rhinehart, R. McAllister, K. Kitani, and S. Levine, "Precog: Prediction conditioned on goals in visual multi-agent settings," in *Proceedings of the IEEE/CVF Int. Conf. on Comput. Vis.*, 2019, pp. 2821–2830.

[16] J. Sun, Z. Wang, J. Li, and C. Lu, "Unified and fast human trajectory prediction via conditionally parameterized normalizing flow," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 842–849, 2021.

[17] E. G. Tabak and E. Vanden-Eijnden, "Density estimation by dual ascent of the log-likelihood," *Communications in Mathematical Sciences*, vol. 8, no. 1, pp. 217–233, 2010.

[18] E. G. Tabak and C. V. Turner, "A family of nonparametric density estimation algorithms," *Communications on Pure and Applied Mathematics*, vol. 66, no. 2, pp. 145–164, 2013.

[19] G. Papamakarios, E. T. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan, "Normalizing flows for probabilistic modeling and inference." *J. Mach. Learn. Res.*, vol. 22, no. 57, pp. 1–64, 2021.

[20] A. Mészáros, J. Alonso-Mora, and J. Kober, "Trajflow: Learning the distribution over trajectories," *arXiv preprint arXiv:2304.05166*, 2023.