Modular Low-Rank Style Transfer for Deep Motion Forecasting

Parth Kothari [*1] **Danya Li** [*1] **Yuejiang Liu** [1] **Alexandre Alahi** [1]

Abstract

Deep motion forecasting models have achieved great success when trained on a massive amount of data. Yet, they often perform poorly when training data is limited. To address this challenge, we propose a transfer learning approach for efficiently adapting pre-trained forecasting models to new domains, such as unseen agent types and scene contexts. Unlike the conventional fine-tuning approach that updates the whole encoder, our main idea is to reduce the amount of tunable parameters that can precisely account for the target domain-specific motion style. To this end, we introduce two components that exploit our prior knowledge of motion style shifts: (i) a low-rank motion style adapter that projects and adjusts the style features at a low-dimensional bottleneck; (ii) a modular adapter strategy that disentangles the features of scene context and motion history to facilitate a fine-grained choice of adaptation layers. Our method outperforms existing fine-tuning methods on three real-world datasets, namely, Stanford Drone, Lyft Level 5 and Intersection Drone in low-shot transfer.

1. Introduction

Motion forecasting is an essential pillar for the successful deployment of autonomous systems in environments comprising various heterogeneous agents. It presents the challenges of modeling (i) physical laws (e.g., goal-directed behaviors, avoiding collisions) that govern general motion dynamics of all agents; and (ii) social norms (e.g., the minimum separation distance, preferred speed) that influence the navigation styles of different agents across different locations. Owing to the success of deep neural networks on large-scale datasets, learning prediction models in a data-driven manner has become a de-facto approach for motion forecasting and has shown impressive results (Alahi et al., 2016; Mangalam et al., 2021; Salzmann et al., 2020).

However, existing deep motion forecasting models suffer from inferior performance when they encounter novel scenarios (Wang et al., 2022b; Xu et al., 2022). For instance, a network trained with large-scale data for pedestrian forecasting struggles to directly generalize to cyclists. Some recent methods propose to incorporate strong priors robust to the underlying distribution shifts (Liang et al., 2020; Liu et al., 2021; Bhattacharyya et al., 2022). Yet, these priors often make strong assumptions on the distribution shifts, which may not hold in practice. This shortcoming motivates the following transfer learning paradigm: Adapting a forecasting model pretrained on one domain with sufficient data to new domains such as unseen agent types and scene contexts *as efficiently as possible*.

One common transfer learning approach is fine-tuning a pre-trained model on data collected from target domain. However, directly updating the model is often sample inefficient, as it fails to exploit the inherent structure of the distributional shifts in motion context. In the forecasting setup, the physical laws behind motion dynamics are generally invariant across geographical locations and agent types: all agents move towards their goal and avoid collisions. As a result, the distribution shift can be largely attributed to the changes in the motion style, defined as the way an agent interacts with its surroundings. Given this decoupling of motion dynamics, it can be efficient for an adaptation algorithm to only account for updates in the target motion style.

In this work, we efficiently adapt a deep forecasting model from one motion style to another. We refer to this task as *motion style transfer*. We retain the domain-invariant dynamics by freezing the pre-trained network weights. To learn the underlying shifts in style during adaptation, we introduce motion style adapters (MoSA), which are new modules inserted in parallel to the encoder layers. The style shift learned by MoSA is injected into the frozen pre-trained model. We hypothesize that the style shifts across forecasting domains often reside in a low-dimensional space. To formulate this intuition, we design MoSA as a low-dimensional bottleneck, inspired by recent works in language (Hu et al., 2021; Mahabadi et al., 2021). Specifically, MoSA comprises two trainable matrices with a low rank. The first

[*]Equal contribution [1]École Polytechnique Fédérale de Lausanne (EPFL), Switzerland. Correspondence to: Alexandre Alahi <alexandre.alahi@epfl.ch>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

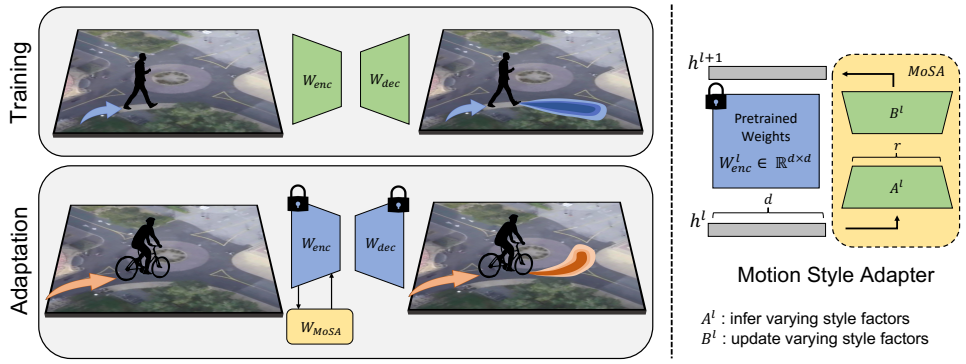


Figure 1. We present an efficient transfer learning technique that adapts a forecasting model trained with sufficient labeled data (e.g., pedestrians), to novel domains exhibiting different motion styles (e.g., cyclists). We freeze the pretrained model and only tune a few additional parameters that aim to learn the underlying style shifts (left). We hypothesize that the style updates across domains lie in a low-dimensional space. Therefore, we propose motion style adapters with a low-rank decomposition ($r \ll d$), designed to infer and update the few style factors that vary in the target domain (right).

matrix is responsible for extracting the style factors to be updated, while the second enforces the updates. Our method introduces and updates less than 2% of total parameters.

In low-resource settings, it can be difficult for MoSA to distinguish the relevant encoder layers updates from the irrelevant ones, resulting in sub-optimal performance. To facilitate an informed choice of adaptation layers, we propose a modularized adaptation strategy. Specifically, we consider forecasting architectures that disentangle the fine-grained scene context and past agent motion using two independent low-level encoders. This design allows flexible injection of MoSA to one encoder while leaving the other unchanged. Given the style transfer setup, our modular adaptation strategy yields substantial performance gains in low-data regime.

We empirically demonstrate the efficiency of MoSA on the state-of-the-art model Y-Net (Mangalam et al., 2021) on the heterogenous SDD (Robicquet et al., 2016). To showcase the generalizability of MoSA in self-driving applications, we adapt a large-scale model trained on one part of the city to an unseen part, on the Level 5 Dataset (Houston et al., 2020). Through extensive experimentation, we quantitatively and qualitatively show that given just 10-30 samples in the new domain, MoSA improves the generalization error by 25% on SDD. Moreover, our design outperforms standard fine-tuning techniques by 20% on the Level 5 dataset.

2. Related Work

Distribution shifts. The primary challenge in adapting to new domains lies in tackling the underlying distributional shifts. One ambitious approach is developing domain generalization techniques that aim to learn models that directly function well in unseen test domains (Gulrajani & Lopez-Paz, 2021; Blanchard et al., 2011). Negative data augmentation techniques have been applied in a limited scope to reduce collisions (Liu et al., 2021) and off-road predictions (Zhu et al., 2021) on new domains. Domain adaptation is another line of work that allows a learning algorithm to observe a set of unlabelled test samples. While this approach has been shown effective in a variety of supervised tasks in vision (Csurka, 2020; Wang & Deng, 2018; Zhao et al., 2020), it is not the ideal setup for motion forecasting setup because labels in the form of future trajectories are fairly easy to acquire. Therefore, in this work, we take an alternate approach of transfer learning using limited data.

Transfer learning. The standard approach of fine-tuning the entire or part of the network (Howard & Ruder, 2018; Radford & Narasimhan, 2018) has been shown to outperform feature-based transfer strategy (Cer et al., 2018; Mikolov et al., 2013). Recently, there has been a growing interest in developing parameter-efficient fine-tuning (PET) methods in both language and vision, as they not only yield a compact model (Houlsby et al., 2019; Hu et al., 2021; Mahabadi et al., 2021), but also show promising results in outperforming fine-tuning in low-resource settings (Mahabadi et al., 2021; Liu et al., 2022a). Similar in spirit to PET methods, we introduce additional parameters in our network but with an objective of style-conditioned motion generation.

3. Method

3.1. Motion Style Transfer

Motion style. Modelling agent motion behavior involves learning the social norms (e.g., minimum separation distance to others, preferred speed, valid areas of traversal) that dictate the motion of the agent in its surroundings. These norms differ across agents as well as locations. For instance, the preferred speed of pedestrians differs from that

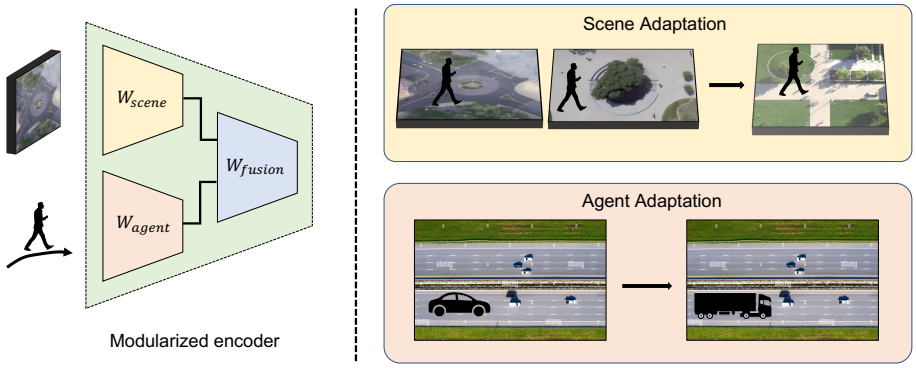


Figure 2. Our modular style transfer strategy updates only a subset of the encoder to account for the underlying style shifts. For instance, we adapt the scene encoder only to model scene style shifts (top right). On the other hand, for the underlying agent motion shift, we only update the agent motion encoder (bottom right). This strategy boosts performance in low-resource settings.

of cyclists; the separation distance between pedestrians in parks differs from that in train stations. To describe these agent-specific (or scene-specific) elements that govern underlying motion behavior, we define the notion of "motion style". Motion style is the collective umbrella that models the social norms of an agent given its surroundings.

Problem statement. We are provided a forecasting model trained on large quantities of data comprising a particular set of style(s). Our goal is to adapt the model to the idiosyncrasies of a target style as efficiently as possible. We denote the model input and ground-truth future trajectory of an agent i using x_i and y_i respectively. The input x_i comprises the past trajectory of the agent, surrounding neighbors, and the surrounding context map. We assume that the data corresponding to an agent type is generated by an underlying distribution $\mathcal{P}_{X,Y}(\cdot; s)$ parameterized by s, the style of the agent. As mentioned earlier, the style is dictated by both the agent type and its surroundings.

Training. The forecasting model has an encoder-decoder architecture (see Fig. 1) with weights W_{enc} and W_{dec} respectively. The training dataset, D_S of size N is given by $\cup_{s\in S} D_s = (x_i, y_i)_{i\in\{1,\dots,N\}}$, where S is a collection of motion styles observed within the dataset. The model is trained to minimize:

$$\mathcal{L}_{train}(D_S; W_{enc}, W_{dec}) = \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}(x_i, y_i; W_{enc}, W_{dec}). \quad (1)$$

Adaptation. When a novel scenario with style s' ($s' \notin S$) is encountered, it leads to a distribution shift and the learned model often struggles to directly generalize to the new dataset $D_{s'} = (x'_i, y'_i)_{i\in\{1,\dots,N_{target}\}}$ of size N_{target}.

In this work, we aim to develop an adaptation strategy for efficient motion style transfer, i.e., cases where N_{target} is small ($N_{target} \ll N$). Often, motion behaviors do not change drastically across domains. We therefore propose to freeze weights of the pretrained forecasting model and introduce motion style adapters, termed *MoSA*, to capture the target motion style. As shown in Fig. 1, we adapt a pre-trained forecasting model by fine-tuning W_{MoSA} with the following objective:

$$\mathcal{L}_{adapt}(D_{s'}; W_{MoSA}) = \frac{1}{N_{target}}\sum_{i=1}^{N_{target}} \mathcal{L}(x'_i, y'_i; W_{MoSA}). \quad (2)$$

3.2. Motion Style Adapters

Our main intuition is that the style shifts across forecasting domains are usually localized – only a few variables of the underlying motion generation process change. Therefore, during style transfer, we only need to adapt the distribution of this small portion of latent factors, while keeping the rest of the factors constant. These updates would correspond to the changes in motion style ($s \rightarrow s'$) in the target domain, as the general principles of motion dynamics remain the same across domains. We design motion style adapters, referred to as MoSA, to carry out these updates.

Our proposed MoSA design comprises a small number of extra parameters added to the model during adaptation (see Fig. 1). Each module comprises two trainable weight matrices of low rank, denoted by A and B. The first matrix A is responsible for inferring the target style factors, while the second matrix B performs the desired update. The low rank r realizes our intuition by restricting the number of style factors that gets updated ($r \ll d$, where d is the dimension size of an encoder layer). Therefore, during adaptation, the weight updates of the encoder are constrained with our low-rank decomposition $W_{MoSA} = BA$.

For brevity, let us consider the adaptation of encoder layer l with input h^l and output h^{l+1}. As shown in Fig. 1, W^l_{enc} and W^l_{MoSA} are multiplied with the same input h^l, and their respective output vectors are summed coordinate-wise:

$$\text{(Train)}\quad h^{l+1} = W^l_{enc} h^l, \quad (3)$$

$$\text{(Adapt)}\quad h^{l+1} = W^l_{enc} h^l + W^l_{MoSA} h^l = W^l_{enc} h^l + B^l A^l h^l. \quad (4)$$

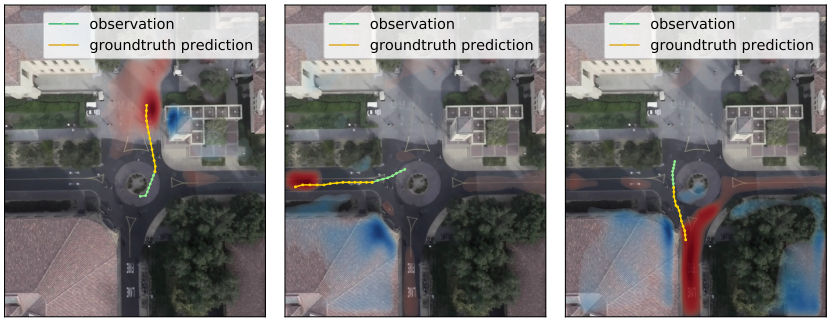


Figure 3. Difference in goal decoder output of Y-Net on the adaptation of pedestrian-trained model using our proposed style-injection modules (Red is positive, blue is negative). During adaptation, Y-Net learns to focus on the road lanes for cyclist forecasting.

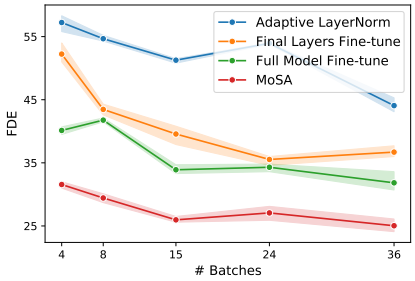


Figure 4. Evaluation of adaptation techniques for long-term motion prediction (25 secs) on Level 5. Error in meters for 5 seeds.

Following common practices (Houlsby et al., 2019; Hu et al., 2021), matrices A and B are initialized with a near-identity function (Hu et al., 2021), so that the original network is unaffected when training starts. Such an initialization also provides flexibility to ignore certain layers during style updates. Despite this flexibility, the total number of extra parameters is significant and can be inconducive to efficient style transfer. Therefore, to further boost sample efficiency, we present a modular adaptation strategy in the next.

3.3. Modular Adaptation Strategy

Motion style can be decoupled into scene-specific style and agent-specific style. Scene-specific style dictates changes in motion due to physical scene structures. The agent-specific style captures the underlying navigation preferences of different agents, e.g., distance to others and preferred speed.

Consider the modularized encoder design shown in Fig. 2. The encoder models the input scene and agent's past history independently. The fusion encoder then fuses the two representations together. This design has the advantage to decouple the task of the style adapters into scene-specific updates and agent-specific updates. Given the modularized setup, the nature of the underlying distribution shifts can guide which modules within the model need to be updated to the target style. Given the style transfer setup, decoupling style adapters can improve the adaptation performance while significantly reducing the number of updated parameters.

4. Experiments

4.1. Motion Style Transfer across Agents on SDD

We perform short-term prediction, in which the future trajectory is predicted for the next 4.8 seconds, given 3.2 seconds of observation. We use the publicly available Y-Net model trained on pedestrian data across *all* scenes and adapt it to cyclists in *deathCircle_0* as there exists a clear distinction between the motion style of pedestrians and cyclists (see Fig. 7). Adaptation uses $N_{target} = \{10, 20, 30\}$ samples.

Table 1. Evaluation of adaptation methods for motion style transfer (pedestrians to bikers) on SDD using few samples $N_{target} = \{10, 20, 30\}$. Error reported is Top-20 FDE in pixels. The generalization error is 58 pixels. Our proposed MoSA outperforms competitive baselines and improve upon generalization error by $> 25\%$. Mean and standard deviation were calculated over 5 runs.

N_{target}	10	20	30
FT	57.28 ± 1.21	52.61 ± 0.87	46.31 ± 1.79
ET (Liu et al., 2022b)	51.88 ± 1.32	46.78 ± 1.78	43.13 ± 1.03
PA (Rebuffi et al., 2018)	52.77 ± 0.85	47.75 ± 1.83	44.70 ± 1.28
MoSA (ours)	$\mathbf{49.98 \pm 1.05}$	$\mathbf{45.55 \pm 0.77}$	$\mathbf{41.69 \pm 0.88}$

Tab. 1 quantifies the performance of various style transfer techniques. The model trained on pedestrians does not generalize to cyclists as evidenced by the high generalization FDE. Our MoSA design reduces this error by $\sim 30\%$ using only 30 samples. Moreover, MoSA outperforms the baselines while updating only 0.5% additional parameters. We also qualitatively analyze the Y-Net goal decoder outputs after model adaptation using MoSA in Fig. 3. One can see that adapted Y-Net successfully learns the style differences between behaviors of pedestrians and bikers.

4.2. Motion Style Transfer across Scenes on L5

We apply MoSA to L5 dataset, where we divide the dataset into two splits based on data collection locations and thereby, construct a scene-style shift scenario (see Fig. 8). We train a Vision Transformer Tiny (ViT-Tiny) model on the majority route and adapt it to a smaller unseen route. To simulate low-resource settings, we provide the frames, sampled at different rates, that cover the unseen route only once. Fig. 4 quantitatively evaluates the performance of various adaptation strategies. MoSA performs superior in comparison to different baselines while adding and updating only 5% of the full model parameters.

For modularization strategy experiments and implementation details: Please refer to appendices.

References

Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., and Savarese, S. Social lstm: Human trajectory prediction in crowded spaces. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 961–971, 2016.

Antonini, G. and Bierlaire, M. Discrete choice models for pedestrian walking behavior. 2006.

Bhattacharyya, P., Huang, C., and Czarnecki, K. Ssl-lanes: Self-supervised learning for motion forecasting in autonomous driving. *arXiv preprint arXiv:2206.14116*, 2022.

Blanchard, G., Lee, G., and Scott, C. D. Generalizing from several related classification tasks to a new unlabeled sample. In *NIPS*, 2011.

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

Caesar, H., Uijlings, J., and Ferrari, V. Coco-stuff: Thing and stuff classes in context. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1209–1218, 2018.

Cer, D. M., Yang, Y., yi Kong, S., Hua, N., Limtiaco, N., John, R. S., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., Strope, B., and Kurzweil, R. Universal sentence encoder for english. In *EMNLP*, 2018.

Coscia, P., Castaldo, F., Palmieri, F. A. N., Alahi, A., Savarese, S., and Ballan, L. Long-term path prediction in urban scenarios using circular distributions. *Image Vis. Comput.*, 69:81–91, 2018.

Csurka, G. Deep visual domain adaptation. *2020 22nd International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, pp. 1–8, 2020.

de Vries, H., Strub, F., Mary, J., Larochelle, H., Pietquin, O., and Courville, A. C. Modulating early visual processing by language. *ArXiv*, abs/1707.00683, 2017.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv*, abs/2010.11929, 2021.

Gulrajani, I. and Lopez-Paz, D. In search of lost domain generalization. *ArXiv*, abs/2007.01434, 2021.

Gupta, A., Johnson, J., Fei-Fei, L., Savarese, S., and Alahi, A. Social gan: Socially acceptable trajectories with generative adversarial networks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2255–2264, 2018.

Helbing and Molnár. Social force model for pedestrian dynamics. *Physical review. E, Statistical physics, plasmas, fluids, and related interdisciplinary topics*, 51 5:4282–4286, 1995.

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., de Laroussilhe, Q., Gesmundo, A., Attariyan, M., and Gelly, S. Parameter-efficient transfer learning for nlp. In *ICML*, 2019.

Houston, J. L., Zuidhof, G. C. A., Bergamini, L., Ye, Y., Jain, A., Omari, S., Iglovikov, V. I., and Ondruska, P. One thousand and one hours: Self-driving motion prediction dataset. In *CoRL*, 2020.

Howard, J. and Ruder, S. Universal language model fine-tuning for text classification. In *ACL*, 2018.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Kothari, P., Kreiss, S., and Alahi, A. Human trajectory forecasting in crowds: A deep learning perspective. *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–15, 2021. doi: 10.1109/TITS.2021.3069362.

Li, Y., Wang, N., Shi, J., Liu, J., and Hou, X. Revisiting batch normalization for practical domain adaptation. *ArXiv*, abs/1603.04779, 2017.

Liang, J., Jiang, L., and Hauptmann, A. Simaug: Learning robust representations from simulation for trajectory prediction. In *European Conference on Computer Vision*, pp. 275–292. Springer, 2020.

Liu, H., Tam, D., Muqeeth, M., Mohta, J., Huang, T., Bansal, M., and Raffel, C. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *ArXiv*, abs/2205.05638, 2022a.

Liu, Y., Yan, Q., and Alahi, A. Social nce: Contrastive learning of socially-aware motion representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15118–15129, 2021.

Liu, Y., Cadei, R., Schweizer, J., Bahmani, S., and Alahi, A. Towards robust and adaptive motion forecasting: A causal representation perspective. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17081–17092, 2022b.

Mahabadi, R. K., Henderson, J., and Ruder, S. Compacter: Efficient low-rank hypercomplex adapter layers. In *NeurIPS*, 2021.

Mangalam, K., An, Y., Girase, H., and Malik, J. From goals, waypoints & paths to long term human trajectory forecasting. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15213–15222, 2021.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. Distributed representations of words and phrases and their compositionality. In *NIPS*, 2013.

Radford, A. and Narasimhan, K. Improving language understanding by generative pre-training. 2018.

Rebuffi, S.-A., Bilen, H., and Vedaldi, A. Efficient parametrization of multi-domain deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8119–8127, 2018.

Robicquet, A., Sadeghian, A., Alahi, A., and Savarese, S. Learning social etiquette: Human trajectory understanding in crowded scenes. In *ECCV*, 2016.

Salzmann, T., Ivanovic, B., Chakravarty, P., and Pavone, M. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *ECCV*, 2020.

Vemula, A., Muelling, K., and Oh, J. Social attention: Modeling attention in human crowds. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1–7, 2018.

Wang, L., Hu, Y., Sun, L., Zhan, W., Tomizuka, M., and Liu, C. Transferable and adaptable driving behavior prediction. *ArXiv*, abs/2202.05140, 2022a.

Wang, L., Hu, Y., Sun, L., Zhan, W., Tomizuka, M., and Liu, C. Transferable and adaptable driving behavior prediction. *arXiv preprint arXiv:2202.05140*, 2022b.

Wang, M. and Deng, W. Deep visual domain adaptation: A survey. *Neurocomputing*, 312:135–153, 2018.

Xu, Y., Wang, L., Wang, Y., and Fu, Y. Adaptive trajectory prediction via transferable gnn. *arXiv preprint arXiv:2203.05046*, 2022.

Zhao, S., Li, B., Reed, C., Xu, P., and Keutzer, K. Multi-source domain adaptation in the deep learning era: A systematic survey. *ArXiv*, abs/2002.12169, 2020.

Zhu, D., Zahran, M., Li, L. E., and Elhoseiny, M. Motion forecasting with unlikelihood training in continuous space. In *Conference on Robot Learning*, pp. 1003–1012. PMLR, 2021.

Appendices

The appendices are organized as follows: First, we describe additional related works. Next, we describe the datasets, pretrained models, the adaptation baselines, and metrics used in our work. Next, we describe the additional experiments to demonstrate the effectiveness of our proposed methods. Finally, we describe the implementation details of all experiments involved. Our codes are available at `https://github.com/vita-epfl/motion-style-transfer`.

A. Additional Related Work

Motion forecasting. Classical models described the interactions between various agents based on domain knowledge but often failed to model complex social interactions in crowds (Helbing & Molnár, 1995; Coscia et al., 2018; Antonini & Bierlaire, 2006). Following the success of Social LSTM (Alahi et al., 2016), various data-driven forecasting models have been proposed to capture social interactions directly from observed data (Kothari et al., 2021; Vemula et al., 2018; Gupta et al., 2018; Salzmann et al., 2020). One common shortcoming of current methods is their reliance on a large and diverse set of training data, which may not be readily available for novel agents and locations. In this work, we adapt a pretrained forecasting model to unseen target domains (novel agent types and scenes) as efficiently as possible.

Style transfer. The popular work of Robicquet et al. (2016) defined *navigation style* as the way different agents interact with their surroundings. It introduced social sensitivity as two handcrafted descriptions of agent style and provided them as input to the social force model (Helbing & Molnár, 1995). In this work, we model style as a latent variable that is learned in a data-driven manner. (Wang et al., 2022a) performed online adaptation across different scenarios for vehicle prediction domains. Closely related to ours, Liu et al. (2022b) decoupled domain-invariant laws and domain-specific style inside their causal forecasting framework. However, their method imposes the strong constraint of requiring access to multiple environments of varying style during training. Furthermore, we decouple motion style into scene-style components and agent-style components to favour efficient adaptation.

B. Datasets

We use a total of three datasets to study the performance of motion style adapters: Stanford Drone Dataset (SDD) (Robicquet et al., 2016), the Intersection Drone Dataset (InD) (Bock et al., 2020), and Level 5 Dataset (L5) (Houston et al., 2020). We consider both short-term and long-term motion forecasting setups.

B.1. Stanford Drone Dataset (SDD)

SDD comprises 20 top-down scenes on the Stanford campus with various agent types (i.e., pedestrians, bicyclists, car, skateboarders, buses, golf carts). We perform short-term prediction where we give 3.2 seconds trajectories and output the future 4.8 seconds. Following the same pre-processing procedure in Mangalam et al. (2021), we filter out short trajectories below 8 seconds in duration, split temporally discontinued trajectories, and then use a sliding window approach without overlap to split the cleaned trajectories. After those steps, the dataset contains 14860 pedestrian trajectories and 5152 bicyclist trajectories. The semantic segmentation has 6 classes, namely pavement, terrain, structure, tree, road, and others (Caesar et al., 2018).

B.2. Intersection Drone Dataset (inD)

The inD Dataset comprises four distinct road intersections, namely *scene1*, *scene2*, *scene3*, and *scene4*, with various agent types, i.e., cars, pedestrians, bicyclists, trucks and buses. We perform both short-term and long-term predictions. The short-term prediction outputs 4.8 seconds trajectories given 3.2 seconds observation, while the long-term prediction generates 30 seconds future trajectories given 5 seconds of observed ones. We use similar pre-processing steps as for SDD. Additionally, we convert the data from the real-world coordinates to pixel coordinates using the provided scaling factors (Bock et al., 2020). After the pre-processing steps, inD contains 1396 long-term pedestrian trajectories, 1508 short-term car trajectories, and 157 short-term trucks trajectories.

B.3. Lyft Level 5 Dataset (L5)

Lyft Level 5 Prediction is a self-driving dataset, containing over 1,000 hours of real-world vehicle data, where each scene lasts 25 seconds. We perform long-term motion forecasting where the ego vehicle moves in a closed loop for the entirety of the scene for 25 seconds, while the surrounding agents follow log-replay (Houston et al., 2020). In the real-world, ego vehicles can come across novel scene contexts, for example, road constructions.

In summary, SDD and inD with different heterogeneous agents provide the ideal setup to validate motion style transfer techniques. L5 dataset provides long sequences of ego vehicle to study the effects of style transfer on long-term self-driving settings.

C. Pre-trained Model Details

We utilize the state-of-the-art model Y-Net (Mangalam et al., 2021) for experiments on SDD and inD. We further propose an alternate design of Y-Net termed Y-Net-Mod to

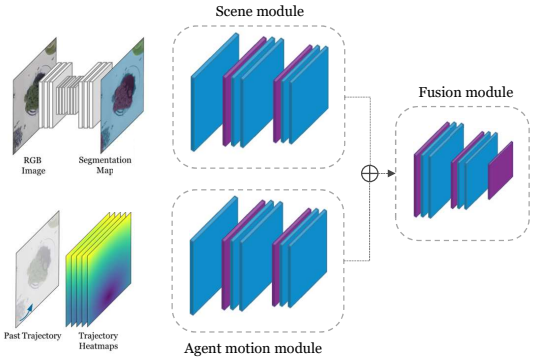
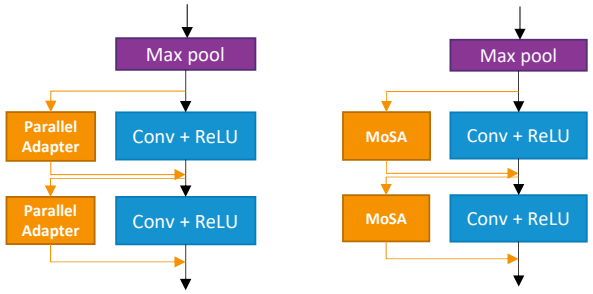


Figure 5. Y-Net-Mod encoder architecture.



(a) Parallel residual adapters (Rebuffi et al., 2018).

(b) Modular Style Adapters (ours).

Figure 6. Illustration of different adapter designs applied to the Y-Net encoder layers.

Table 2. Generalization performance of Y-Net and Y-Net-Mod on two modularization setups. Errors reported are Top-20 ADE/FDE in pixels. Y-Net-Mod performs at par with the original Y-Net.

Experiment	Y-Net	Y-Net-Mod
Scene transfer on inD	6.44 / 10.72	6.60 / 11.17
Agent motion transfer on inD	24.48 / 33.54	24.56 / 29.07

demonstrate the efficacy of our modular adaptation strategy. Finally, we utilize the ViT-Tiny (Dosovitskiy et al., 2021) architecture for experiments on L5.

C.1. Y-Net

Y-net (Mangalam et al., 2021) comprises three sub-networks: the scene heatmap encoder, the waypoint heatmap decoder, and the trajectory heatmap decoder. Specifically, the encoder is designed as a U-net encoder which consists of one center convolutional layer, four intermediate blocks where each uses max pooling and two convolutional layers, and one final max pooling layer. It takes as input the concatenation of the scene semantic map and past trajectory heatmap.

C.2. Y-Net-Mod

We construct Y-Net-Mod on top of the original Y-Net architecture. The modification treats the scene context and agent motion *independently* before fusing their representations together. As shown in Fig. 5, the first three layers of the original encoder are decoupled into scene context and past agent motion modules in order to learn their representations independently. Subsequently, the representations are fused together using the fusion encoder, that is similar in design to the last two layers of Y-Net. The original number of channels in each encoder layer of Y-Net are evenly divided between each module in Y-Net-Mod encoder so that the latter is compatible with the Y-Net decoders.

Benchmarking Y-Net and Y-Net-Mod: To illustrate that our modification to Y-Net does not result in severe drop in performance, we benchmark the performance of Y-Net and Y-Net-Mod on inD for the modularization setups presented in main draft: 1) scene transfer of pedestrians, 2) agent motion transfer from cars to trucks. Table. 2 illustrates that modularization of the Y-Net encoder does not lead to a significant drop in the performance.

C.3. Vision Transformer

We utilize the official ViT-Tiny architecture (Dosovitskiy et al., 2021) for the Level 5 dataset. We only modify the last layer to output the forecasting predictions in the form of x, y coordinates for T_{pred} time-steps.

D. Adaptation Techniques

We compare the following methods during adaptation for the experiments reported in the paper:

Full Model Finetuning (FT) (Howard & Ruder, 2018): We update the weights of the entire model . The learning rate (LR) is 5e-5, unless mentioned otherwise.

Partial Model Finetuning (ET) (Liu et al., 2022b): We update the weights of the Y-Net encoder for SDD and inD , and the last two layers of ViT for Level 5. The LR is 5e-4, unless mentioned otherwise.

Parallel Adapters (PA) (Rebuffi et al., 2018): We insert a convolutional layer with filter size of 3 in parallel to each encoder layer and update the weights of these layers. The LR is 5e-5, unless mentioned otherwise. See Fig. 6a.

Adaptive Layer Normalization (BN) (Li et al., 2017; de Vries et al., 2017): We update the weights and biases of the layer normalization, wherever present. The LR is 1e-4, unless mentioned otherwise.

Motion Style Adapters (MoSA) [Ours]: We insert our motion style adapters in parallel to each encoder layer in SDD and inD, and in parallel to query and value matrices of multi-headed attention in L5. During modularized adaptation, we add our modules across the specified encoders only rather than the entire encoder. The LR is 3e-3 and the rank r is 1, unless mentioned otherwise. See Fig. 6b.

D.1. Initialization of Motion Style Adapters

Matrices A and B are initialized such that the original network is not affected when training starts. Specifically, we use a random Gaussian initialization for A and zero for B. This initialization scheme allows these modules to be ignored at certain layers if there is no need for a change in activation distribution.

D.2. Metrics

We use the established Average Displacement Error (ADE) and Final Displacement Error (FDE) metrics for measuring the performance of model predictions. ADE is calculated as the l_2 error between the predicted future and the ground truth averaged over the entire trajectory while FDE is the l_2 error between the predicted future and ground truth for the final predicted point (Alahi et al., 2016). For multiple predictions, the final error is reported as the `min` error over all predictions (Gupta et al., 2018). Additionally, we define the generalization error as the error of the pretrained model on the target domain. The more the dissimilarity between the source domain and target domain, the higher the generalization error.

E. Additional Experiments for Motion Style Adapters

Besides the two motion style transfer experiments presented in the paper, we have an additional experiment on inD to demonstrate the efficacy of our motion style adapters.

E.1. Motion Style Transfer across Scenes on inD

In this setup, we perform long-term prediction, in which future trajectory in the next 30 seconds is predicted, given 5 seconds of observation. We use the publicly available Y-Net model and follow the experimental protocol of (Mangalam et al., 2021) in which the model is trained on pedestrians in $\{scene2, scene3, scene4\}$ and tested on pedestrians from the unseen scene $scene1$. $N_{target} = \{10, 20, 30\}$ samples are used during adaptation.

Despite the long-term prediction setup, the generalization error, in this case, is 33 pixels that are lower compared to the previous SDD setup, as the target domain is more similar to the source domain. Tab. 3 quantifies the performance of scene style transfer across all methods. Once again, using just 30 samples, MoSA improves the generalization error by $\sim 40\%$ and outperforms its counterparts.

Table 3. Evaluation of adaptation methods for scene style transfer on InD using few samples $N_{target} = \{10, 20, 30\}$. Error reported is Top-20 FDE in pixels. The generalization error on inD is 33 pixels. Our proposed MoSA outperforms competitive baselines and improve upon the generalization error by $> 25\%$. Mean and standard deviation were calculated over 5 runs.

N_{target}	10	20	30
FT	27.92 ± 1.99	25.15 ± 1.08	23.18 ± 0.64
ET (Liu et al., 2022b)	28.06 ± 0.68	23.19 ± 1.39	21.13 ± 1.00
PA (Rebuffi et al., 2018)	28.71 ± 1.50	26.10 ± 0.74	25.00 ± 1.08
MoSA (ours)	$\mathbf{25.18 \pm 0.72}$	$\mathbf{21.70 \pm 0.84}$	$\mathbf{20.35 \pm 1.18}$

Figure 7. Distribution of trajectories of pedestrians (blue) and bikers (red) on *deathCircle*.

E.2. Additional visualization for SDD and L5

Fig. 7 presents the trajectory distribution of pedestrians and cyclists on SDD, where we can see clear distinction. Fig. 8 shows the way we split L5 dataset to construct a scene-transfer scenario.

F. Additional Experiments for Modular Structure

Now, we empirically demonstrate the effectiveness of applying our motion style adapters on top of a modularized architecture on three setups: motion style transfer across agents and across scenes on inD and motion style transfer across agent motion on SDD.

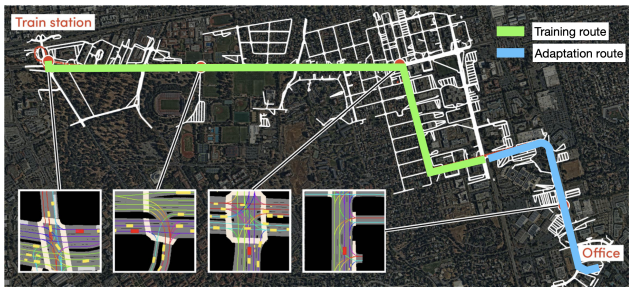


Figure 8. Training-Adaptation split for Level 5 prediction dataset. Model trained on the green route and adapted on the blue route. The distribution shift arises from differing styles in different scene locations. The inherent motion style parameters of the agent remain same (ego vehicle in both cases). Background taken from (Houston et al., 2020).

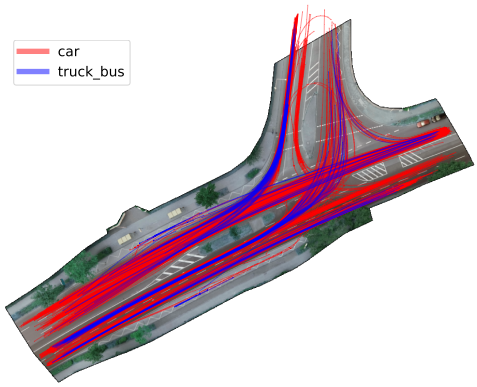


Figure 9. Car and truck trajectory distribution in inD dataset *scene1*. The two types of agents share the same scene context.

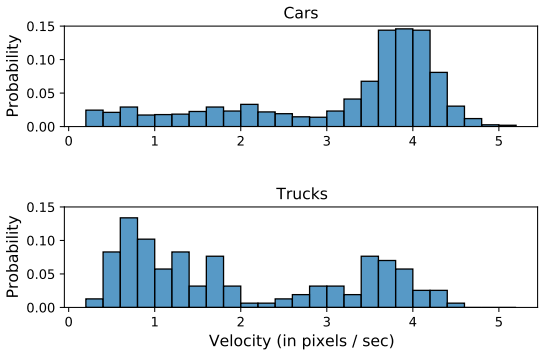


Figure 10. Velocity distribution of *car* and *truck* in inD scene1. Static trajectories have been filtered.

We modularize the Y-Net architecture as shown in Fig. 5, where we treat scene and agent motion independently for the first two layers of the encoder before fusing the learned representations. Given Y-Net-Mod, we consider five cases given the module list on which MoSA is applied: (1) scene only [S], (2) agent motion encoder only [A], (3) scene and fusion encoder [S+F], (4) agent motion and fusion encoder [A+F], and (5) scene, agent motion and fusion encoders together [S+A+F].

F.1. Motion Style Transfer across Agents on inD

In $scene1$ of inD, cars and trucks share the same scene context (see Fig. 9) differing only in their velocity distribution (see Fig. 10). Fig. 11 represents the performance of style transfer from cars to trucks on 20 samples under five different adaptation cases. It is interesting to note that adapting the agent motion encoder alone [A] performs the best while including the scene encoder for adaptation deteriorates performance ([S] worse than [A], [S+A+F] worse than [A+F]).

Fig. 13 also qualitatively shows the output difference before and after using various modularized adaptation strategies. We can see that adapting the agent motion encoder alone indeed captures the speed changes from cars to trucks, and updating this single module is already sufficient to infer such change.

F.2. Motion Style Transfer across Scenes on inD

We train the Y-Net-Mod model on pedestrian data on scene ids = $\{2, 3, 4\}$ and adapt it on $scene1$ of inD. Fig. 12 represents the performance of scene style transfer on 20 samples in the five cases. Contrary to the previous setup, adapting the scene encoder [S] is clearly more important than the agent motion encoder [A]. Further, adapting the agent encoder deteriorates performance ([S+A+F] worse than [S+F]). It is clear that modularization helps boost MoSA performance.

We qualitatively show the goal decoder output difference of three examples in Fig. 14. We can observe that the adapted model learns to cross at a particular position of the road, which is not presented in the training data.

F.3. Motion Style Transfer across Agent Motion on SDD

We also demonstrate the efficacy of modularized architecture on the SDD dataset. We utilize the cyclists data on *deathCircle_0* scene. The training and adaptation data are constructed based on the average speed of the cyclist trajectories. The training set contains low-speed trajectories with the speed in the range of 0.5 to 3.5 pixel per second. The adaptation set has cyclist trajectories with an average speed in the range of 4 to 8 pixels per second. We use $N_{target} = \{20\}$ trajectories for adaptation. Given our setup, the dominant underlying style factor that changes across domains is the agent speed distribution (the scene context and the type of agent are fixed). We benchmark the performance of Y-Net-Mod given the five modularization strategies de-

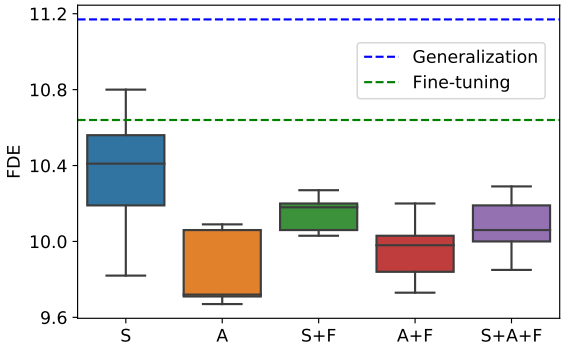


Figure 11. Agent Motion Transfer. Style Transfer from *cars* to *trucks* on InD scene1 with $N_{target} = 20$ samples using different MoSA configurations. [A] performs best while [S] worsens performance. Error reported is Top-20 FDE in pixels across 5 seeds.

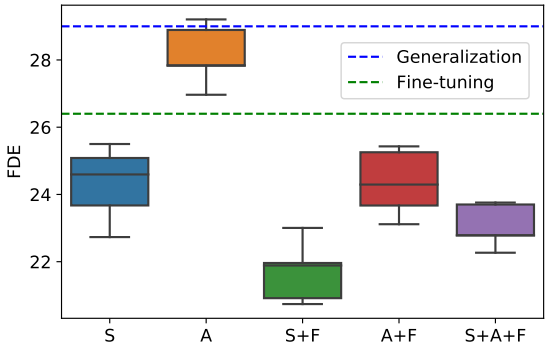


Figure 12. Scene Transfer. Style Transfer across scenes on InD *pedestrians* with $N_{target} = 20$ samples using different MoSA configurations.[S+F] performs best while [A] worsens performance. Error reported is Top-20 FDE in pixels across 5 seeds.

scribed in the paper. Rank of MoSA is set to 2.

Fig. 15 illustrates the performance of various modular adaptation strategies. The generalization error is very high as the error accumulates quickly for fast-moving agents. Using just 20 samples, updating only the agent motion module [A] leads to the best performance, while including the scene context module [S] during adaptation worsens performance.

G. Implementations Details

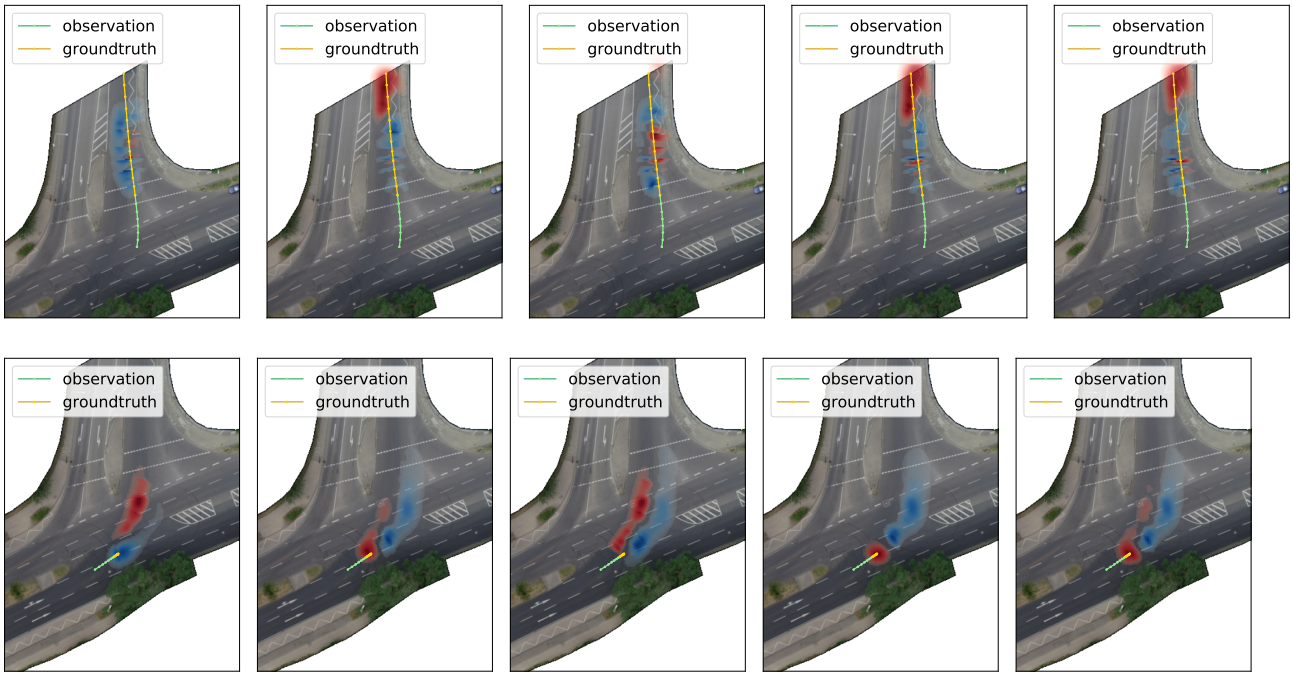
We present implementation details and hyperparameters for each method and model training. For each experiment, the best model is chosen based on the performance on the validation set. All model pretraining follows the training, validation and test split of 7:1:2. During adaptation, all experiments utilize Adam optimizer and a batch size of 10, unless mentioned otherwise. Learning rate for FT is 5e-5, 5e-4 for ET, 5e-5 for PA, 1e-4 for BN, and 5e-3, unless mentioned otherwise. The details for our designed experiments are listed as below.

Motion Style Transfer across Agents on SDD. We pretrain Y-Net network for 100 epochs and learning rate of 5e-5. Rest of the hyper-parameters are kept the same as (Mangalam et al., 2021). We adapt the pretrained model using $N_{target} = \{10, 20, 30\}$ trajectories and utilize 80 trajectories for validation. We adapt the pretrained model for 100 epochs with an early stop value of 30 epochs. For MoSA, the rank value is set to 3.

Motion Style Transfer across Scenes on inD. In this experiment, we utilize the pretrained model provided by Y-Net paper (Mangalam et al., 2021). We adapt this model using $N_{target} = \{20\}$ trajectories and use 40 trajectories for validation. The pretrained model is 100 epochs. Fig. 14 illustrates the adaptation performance of MoSA using 30 samples. MoSA learns the unseen behavior of pedestrians crossing at a particular segment of the road, that was unobserved in the training scenes of inD.

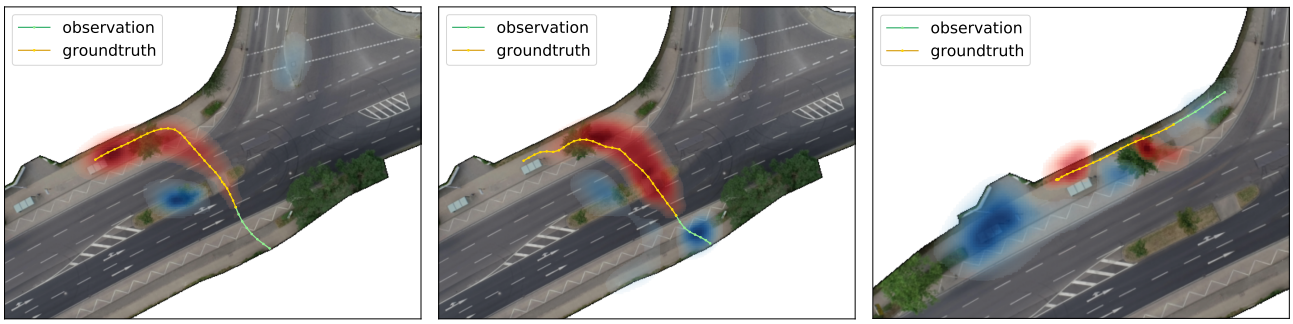
Motion Style Transfer across Scenes on L5. To simulate a scene context transfer scenario, we split the dataset as shown in Fig. 8. The ViT-Tiny model is trained on the majority route shown in green and adapted to the blue route that was not seen during training. We follow the training strategy provided in Houston et al. (2020): specifically, the network is trained to accept BEV rasters of size 224×224 pixels (centered around the SDV) to predict future (x, y) positions over a 1.2 second horizon. The hyper-parameters of the ViT-Tiny architecture are kept the same as in (Dosovitskiy et al., 2021). We train the model on the training data corresponding to the majority route for 15 epochs using batch size of 64. To simulate low-resource settings during adaptation, the network is shown the unseen route only once, sampled at different rates. As a result, we adapt for $N_{batches} = \{4, 8, 15, 24, 36\}$ with a batch size of 64. We benchmark the following four cases: 1) Full model fine-tuning with learning rate of 1e-4, 2) Final layer fine-tuning with learning rate of 3e-4, 3) Adaptive layer normalization with a learning rate of 1e-4, 4) MoSA (ours) with rank value of 8 and learning rate of 3e-3. We apply MoSA across the query and value matrices of each attention layer. We observe that applying MoSA across the feed-forward layers deteriorated performance. We adapt all the methods for 250 epochs using a one-cycle learning rate scheduler.

Motion Style Transfer across Agents on inD with Modular Structure. We perform style transfer from cars to trucks in *scene1* of inD. Cars and trucks data have different speed distribution (see Fig. 9) but share the same context as shown in Fig. 9. Trajectories with an average speed less



(a) S (b) A (c) S+F (d) A+F (e) S+A+F

Figure 13. Illustration of the difference in goal decoder output of Y-Net-Mod with various adaptation strategies for agent motion style transfer on InD (Red is positive, blue is negative). In both cases (two rows), adapting the agent motion encoder alone [A] is sufficient to effectively transfer the model from cars to trucks, as they share the same scene context.



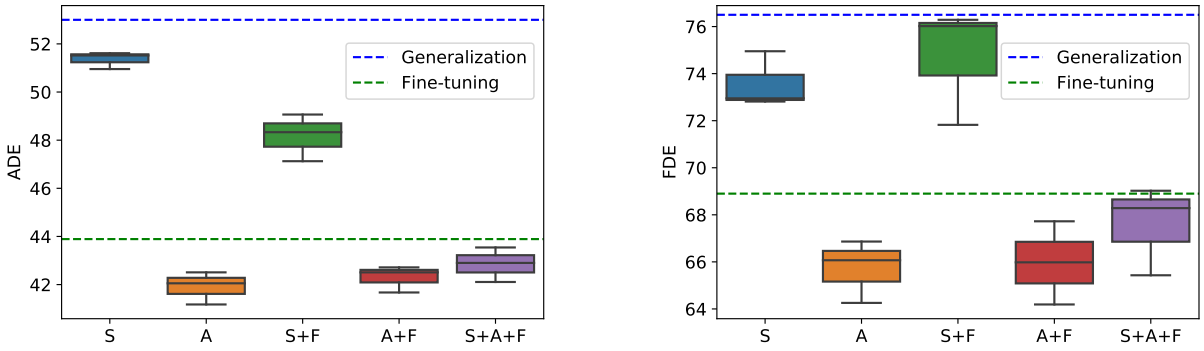
(a) Sample 1 (b) Sample 2 (c) Sample 3

Figure 14. Illustration of the difference in goal decoder output of Y-Net on the scene style transfer on InD using our motion style adapters (Red is positive, blue is negative). Observe that in the adapted scene, pedestrians tend to cross at a particular segment of the road. This behavior did not occur during the pre-training. MoSA learns this novel behavior using just 30 samples of adaptation.

than 0.2 pixels per second are filtered out. We train a Y-Net-Mod model on cars and adapt the model to trucks in which $N_{target} = \{20\}$ trajectories are used for adaptation and 40 trajectories for validation.

Motion Style Transfer across Scenes on inD with Modular Structure. We pretrained Y-Net-Mod model using pedestrian data from scene ids = $\{2, 3, 4\}$ and transferred to pedestrian data from *scene1* following the setup in (Mangalam et al., 2021). The adaptation uses $N_{target} = \{20\}$ for fine-tuning and 20 trajectories for validation.

Motion Style Transfer across Agent Motion on SDD with Modular Structure. We pretrain a Y-Net-Mod model using slow cyclists from *deathCircle_0* scene and adapt to fast cyclists from the same scene. The pretraining set has 1213 trajectories. The adaptation set has 381 trajectories where 50 trajectories are used for validation.



(a) Top-20 Average Displacement Error

(b) Top-20 Final Displacement Error

Figure 15. Agent Motion Transfer on SDD. Style Transfer from *slow* cyclists to *fast* cyclists on *death_circle* scene with $N_{target} = 20$ samples using different MoSA configurations. [A] performs best while [S] worsens performance. Error reported across 5 seeds.

H. Conclusion

In this work, we tackled the task of efficient motion style transfer wherein we adapt a trained forecasting model on a target domain comprising limited samples of unseen target styles. We argue that we only need to model the underlying style shift across domains, which often reside in a low-dimensional space. We formulated this intuition into our motion style adapter (MoSA) design, which is trained to infer and update the style factors of variation in the target domain while keeping the pretrained parameters frozen. Additionally, we presented a modularized style transfer strategy that updates only a subset of the model given the nature of the style transfer problem. Extensive experimentation on three real-world datasets demonstrated the effectiveness of our approach.