Motion Style Transfer: Modular Low-Rank Adaptation for Deep Motion Forecasting

Anonymous Author(s)
Affiliation
Address
email

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, which projects and adapts the style features at a low-dimensional bottleneck; (ii) a modular adapter strategy, which disentangles the features of scene context and motion history to facilitate a fine-grained choice of adaptation layers. Experiment results show that our method improves the performance of modern forecasting models by 25% using only 30 samples from the target domain on Stanford Drone and Intersection Drone dataset. Moreover, while updating less than 5% extra parameters, our method achieves 20% relative improvement compared to fully fine-tuned models on Lyft Level 5 dataset.

Keywords: Motion Forecasting, Distribution Shifts, Transfer Learning

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. 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 [1, 2, 3, 4]. However, existing deep motion forecasting models suffer from inferior performance when they encounter novel scenarios [5].

A forecasting model trained on a specific domain with large-scale data [6, 3] may not generalize to other domains. One solution is to develop models with strong priors robust to distribution shifts [7, 5]. Another approach involves adapting the model to the target domain as efficiently as possible. In this work, we tackle the latter challenge of efficiently adapting a forecasting model pretrained on one domain with sufficient data to new domains such as unseen agent types and scene contexts.

To adapt a forecasting model to new domains, we leverage the fact that the knowledge learned by deep neural networks is transferrable. A simple yet competitive approach for transfer learning is fine-tuning a pretrained model on the data collected from target domains. However, directly updating the model encoder without exploiting the inherent structure of the distributional shifts, is often sample-inefficient. Given the observation that the physical laws behind motion dynamics are generally invariant across geographical locations and agent types, it can be advantageous for an

Submitted to the 6th Conference on Robot Learning (CoRL 2022). Do not distribute.

adaptation algorithm to only account for the changes in domain-specific 'style' – the way an agent interacts with its surroundings (static or dynamic).

In this work, we aim to effectively adapt a deep forecasting model from one style to another with limited data. We refer to this task as *motion style transfer*. We introduce very few extra parameters, in parallel to encoder layers, that learn the style shift corresponding to the target scenario. The learned style shift is subsequently injected into the frozen pretrained model. Our main motivation is that the style shifts across forecasting domains often reside in a low-dimensional space. We formulate this intuition in the form of Motion Style Adapter (MoSA) inspired by recent works in language [8, 9]. MoSA comprises two trainable matrices with low rank where the first matrix is responsible for extracting the style factors to be updated, while the second enforces the updates. Thanks to the low-rank decomposition, MoSA learns the style updates by adding and fine-tuning less than 2% of the parameters in *each layer*.

To further boost the sample efficiency of motion style transfer, we introduce a modular adaptation strategy that aims to update only a subset of layers that accounts for the underlying style shifts. Specifically, we consider forecasting architectures that disentangle the fine-grained context and past trajectory by means of two independent low-level encoders. This modular design allows us to flexibly inject our MoSA only to one encoder while leaving the other unchanged. By choosing the module to be adapted based upon the type of style transfer setup *known a-priori*, our method yields substantial performance gains for adaptation in the low-resource setting.

We empirically demonstrate the efficiency of MoSA on the state-of-the-art model Y-Net [2] on the heterogenous SDD [6] and inD [10] datasets in various style transfer setups. Next, we highlight the potential of a modularized style transfer strategy on two setups: Agent Motion Style Transfer and Scene Style Transfer. Finally, 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 [11]. 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 and inD. Moreover, our design outperforms standard fine-tuning techniques by 20% on the Level 5 dataset.

2 Background

Motion forecasting. Classical models described the interactions between various agents based on domain knowledge but often failed to model complex social interactions in crowds [12, 13, 14]. Following the success of Social LSTM [1], various data-driven forecasting models have been proposed to capture social interactions directly from observed data [3, 15, 16, 4]. 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. Recent works proposed to address this challenge by incorporating strong priors into learning objectives [17, 18] or architecture designs [7]. In this work, we take another approach and adapt a pretrained model to target domains as efficiently as possible.

Transfer learning. We are interested in the inductive transfer learning setting for deep motion forecasting [19]. Features in deep neural networks have been observed to transition from general to task-specific from the initial to the final layers [20]. The two most common transfer learning techniques are feature-based transfer [21, 22] and fine-tuning [23, 24], with fine-tuning outperforming its counterpart [23]. Recently, there has been a rising interest in developing parameter-efficient fine-tuning (PET) methods in both language and vision [25, 8, 9]. Due to the parameter-efficient approach, PET methods have been shown to outperform fine-tuning in low-resource settings [9, 26]. In this work, we empirically demonstrate that a pretrained model can be easily adapted to the idiosyncrasies of a target agent, location, or both using our motion adapters.

Style transfer. In vision, style transfer refers to the task of transforming an input image into a particular artistic style while preserving some notion of content [27]. In motion context, the popular work of Robicquet et al. [6] defined *navigation style* as the way different agents interact with their surroundings. It introduced social sensitivity as two handcrafted descriptions of agent style and

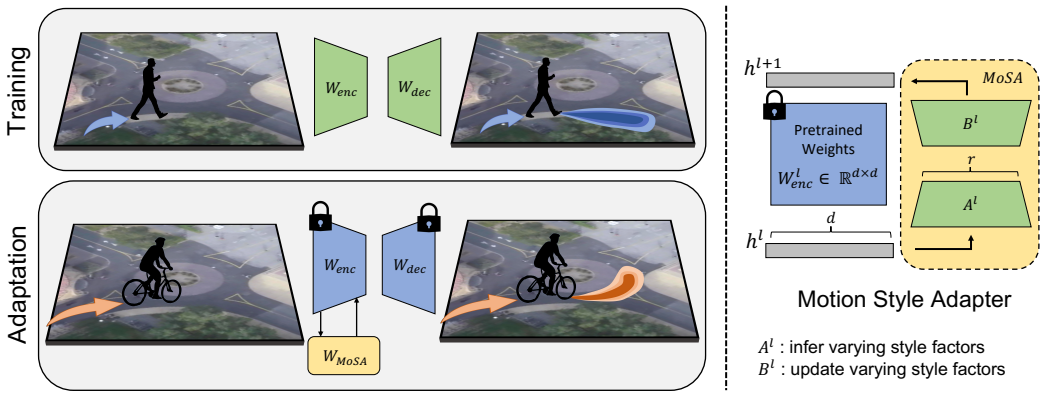

Figure 1: We present an efficient transfer learning technique that adapts a forecasting model trained with sufficient labeled data (e.g., pedestrians), to novel domains exhibiting different motion styles (e.g., cyclists). We freeze the pretrained model and only tune a few additional parameters that aim 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).

provided them as input to the social force model [12]. In this work, we model style as a latent variable that is learned in a data-driven manner. Moreover, given limited samples from the target domain, our adapter design infers the style factors that vary in this new domain and inject the corresponding shifts into the motion encoder.

3 Method

Deep neural networks have shown remarkable performance in motion forecasting thanks to the availability of large-scale datasets. However, these models often struggle to generalize when unseen scenarios are encountered in the real world due to underlying differences in motion style. In this section, we first formally introduce the problem setting of motion style transfer. Subsequently, we introduce the design of our style adapter modules that help to effectively tackle motion style transfer. Finally, we describe the application of style adapters to a modularized architecture in order to perform style transfer in a more efficient manner.

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 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. Consider the inductive transfer learning setting for deep motion forecasting across different motion styles. Specifically, we are provided a forecasting model trained on large quantities of data comprising a particular set of style(s) and 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. The context can take various forms like RGB images or rasterized maps. 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 $\cup_{s \in S} D_s = (x_i, y_i)_{i \in \{1,\ldots,N\}}$, where S is a collection of motion styles observed within the dataset. The model is trained to minimize:

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

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 corresponding dataset $D_{s'} = (x'_i, y'_i)_{i \in \{1,\ldots,N_{target}\}}$ of size N_{target}. The common approach to tackling such shifts is to fine-tune the entire or part of the pretrained model. Fine-tuning optimizes an objective similar to training, but on the new dataset:

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

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. Instead, most of the behavioral dynamics are governed by universal physical laws (e.g., influence of inertia, collision-avoidance). We therefore propose to freeze the weights of the pretrained forecasting model and introduce motion style adapters, termed *MoSA*, to capture the target motion style. As shown Fig. 1, we adapt a pre-trained forecasting model by fine-tuning W_{MoSA} with the following objective:

$$\mathcal{L}_{adapt}(D_{s'}; W_{MoSA}) = \frac{1}{N_{target}} \sum_{i=1}^{N_{target}} \mathcal{L}(x'_i, y'_i; W_{MoSA}). \tag{3}$$

3.2 Motion Style Adapters

Our main intuition is that the style shifts across forecasting domains are usually localized – they are often due to the changes in only a few variables of the underlying motion generation process. 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 (e.g., avoid collisions, move towards goal) remain the same across domains for all agents. 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 style factors that are required to be updated to match the target style, while the second matrix B performs the desired update. The low rank r realizes our intuition that style updates reside in a low-dimensional space, by restricting the number of style factors that get updated ($r << d$ 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 $W_{MoSA} = BA$. The pretrained model is frozen and only A and B are trained.

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 as shown below:

$$\text{(Train)} \;\; h^{l+1} = W^l_{enc} h^l, \tag{4}$$

$$\text{(Adapt)} \;\; h^{l+1} = W^l_{enc} h^l + W^l_{MoSA} h^l = W^l_{enc} h^l + B^l A^l h^l. \tag{5}$$

It has been shown that initialization plays a crucial role in parameter-efficient transfer learning [25, 8]. Therefore, following common practices, matrices A and B are initialized with a near-identity function

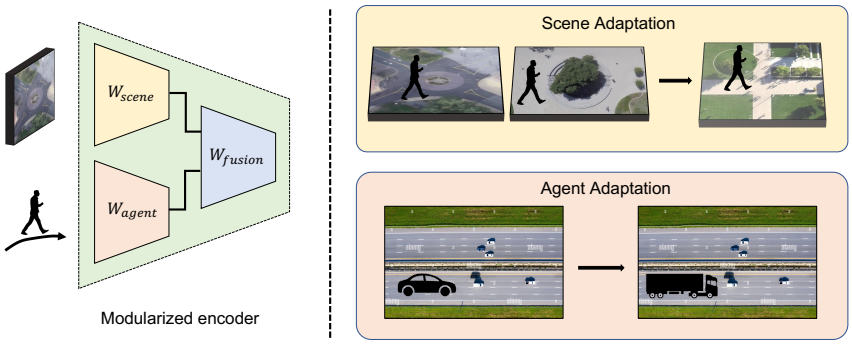

Figure 2: Our modular style transfer strategy updates only a subset of the encoder to account for the underlying style shifts. For instance, we adapt the scene encoder only to model scene style shifts (top right). On the other hand, for the underlying agent motion shift, we only update the agent motion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

[8], so that the original network is unaffected when training starts. Furthermore, the near-identity initialization provides flexibility to these modules to ignore certain layers during motion 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 which we describe next.

3.3 Decoupling Motion Style Adapters

Motion style can be decoupled into scene-specific style and agent-specific style. Scene-specific style dictates the changes in motion due to physical scene structures. For instance, cyclists prefer to stay on the lanes while pedestrians move along sidewalks. The agent-specific style captures the underlying navigation preferences of different agents like distance to others and preferred speed. Modularizing the encoder into two parts that account for the physical scene and agent's past motion independently can help to decouple the functionality of our motion style adapters and further improve performance.

Consider the modularized motion encoder designs shown in Fig. 2. The modularized 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 help guide which modules within the model are required to be updated to the target style. As demonstrated in Section. 4.4, given different categories of style transfer setups, decoupling style adapters can improve the adaptation performance while significantly reducing the number of updated parameters.

4 Experiments

Datasets. We use a total of three datasets to study the performance of motion style adapters: Stanford Drone Dataset (SDD) [6], the Intersection Drone Dataset (InD) [10], and Level 5 Dataset (L5) [11]. We consider both short-term and long-term motion forecasting setups. SDD and inD with different heterogeneous agents exhibiting different motion styles provide the ideal setup to validate motion style transfer techniques. L5 dataset provides 25-second long sequences of ego vehicle to study the effects of style transfer on long-term self-driving settings.

Baselines. We compare MoSA with full model fine-tuning [FT] [23] and partial model fine-tuning [ET] [7] paradigms. We also compare with another parallel adapter design [PA] proposed by Rebuffi et al. [28], which does not incorporate the low-rank constraint. For architectures with normalization layers, we compare against the adaptive normalization strategy [29] [BN]. For a fair comparison, we use the state-of-the-art Y-Net architecture [30] for all baselines on SDD and inD. On L5, we use

Table 1: Evaluation of adaptation methods for motion style transfer (pedestrians to bikers) on SDD and 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 SDD is 58 pixels and on inD is 33 pixels. Our proposed motion style adapters (MoSA) outperform competitive baselines and improve upon the generalization error by $> 25\%$ in both setups. Mean and standard deviation were calculated over 5 runs.

	Stanford Drone Dataset			Intersection Drone Dataset		
N_{target}	10	20	30	10	20	30
FT	57.28 ± 1.21	52.61 ± 0.87	46.31 ± 1.79	27.92 ± 1.99	25.15 ± 1.08	23.18 ± 0.64
ET [7]	51.88 ± 1.32	46.78 ± 1.78	43.13 ± 1.03	28.06 ± 0.68	23.19 ± 1.39	21.13 ± 1.00
PA [28]	52.77 ± 0.85	47.75 ± 1.83	44.70 ± 1.28	28.71 ± 1.50	26.10 ± 0.74	25.00 ± 1.08
MoSA (ours)	$\mathbf{49.98 \pm 1.05}$	$\mathbf{45.55 \pm 0.77}$	$\mathbf{41.69 \pm 0.88}$	$\mathbf{25.18 \pm 0.72}$	$\mathbf{21.70 \pm 0.84}$	$\mathbf{20.35 \pm 1.18}$

the Vision Transformer (ViT) architecture [31] across all methods. The implementation details and hyperparameters for each method and model training are provided in the supplementary materials.

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 [1]. For multiple predictions, the final error is reported as the `min` error over all predictions [16]. 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.

4.1 Motion Style Transfer across Agents on Stanford Drone Dataset

We perform short-term prediction, in which the future trajectory is predicted for the next 4.8 seconds, given 2.5 seconds of observation. The main objective of efficient motion style transfer is to adapt the model with a limited number of samples. We choose to adapt our model on cyclists in *deathCircle_0* as there exists a clear distinction between the motion style of pedestrians and cyclists (see Fig. 3). We use the publicly available Y-Net model trained on pedestrian data across *all* scenes and adapt it to the cyclists' data in *deathCircle_0*. We adapt the model using $N_{target} = \{10, 20, 30\}$ samples.

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 error of 58 pixels. Our MoSA design reduces this error by $\sim 30\%$ using only 30 samples. Moreover, MoSA outperforms the baselines while keeping the pretrained model frozen and updating only 0.5% additional parameters.

We qualitatively analyze the Y-Net goal decoder outputs after model adaptation using MoSA. Fig. 4 illustrates the updates in the goal decoder output (red means increase in focus and blue means decrease in focus). Adapted Y-Net successfully learns the style differences between the behavior of pedestrians and bikers: 1) it correctly infers valid areas of traversal, 2) effectively captures the multimodality of bikers, and 3) updates the motion style parameters as the new biker goal positions (red) are farther from the end of the observation position, compared to the un-adapted goal positions (blue).

4.2 Motion Style Transfer across Scenes on Intersection Drone Dataset

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 [2] in which the model is trained on pedestrians in $\{scene2, scene3, scene4\}$ and tested on unseen scene $scene1$. We adapt the model using $N_{target} = \{10, 20, 30\}$ samples.

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. 1 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.

Figure 3: Distribution of trajectories of pedestrians (blue) and bikers (red) on *deathCircle*.

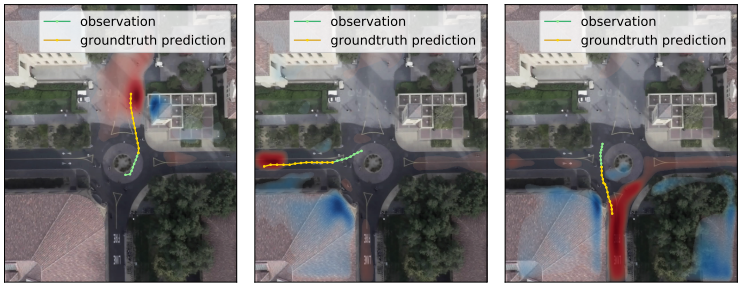

Figure 4: Illustration of the difference in goal decoder output of Y-Net on the adaptation of pedestrian-trained model using our proposed style-injection modules (rank=1) (Red is positive, blue is negative). During adaptation, Y-Net learns to focus on the road lanes for biker forecasting.

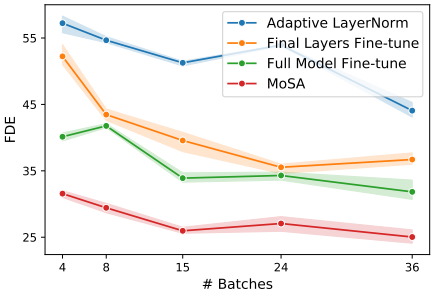

Figure 5: Evaluation of adaptation techniques for long-term motion prediction (25 secs) on Level 5. Error in meters for 5 seeds.

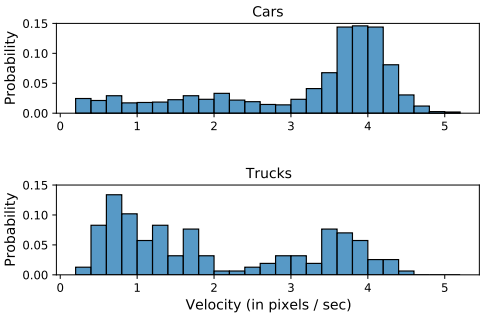

Figure 6: Velocity distribution of *car* and *truck* in inD scene1. Static trajectories have been filtered.

4.3 Motion Style Transfer across Scenes on Lyft Level 5 Dataset

To demonstrate the generalizability of our approach, we apply MoSA to L5 dataset. We divide the dataset into two splits based on the data collection locations and thereby, construct a scene style shift scenario. We train a Vision Transformer Tiny (ViT-Tiny) model on the majority route and adapt it to the smaller route that was not seen during training. To simulate low-resource settings, we provide the frames, sampled at different rates, that cover the unseen route only once.

Fig. 5 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. It is apparent that our low-rank design is a smarter way of adapting models in the motion context as compared to fine-tuning the whole or the last few layers of the model.

4.4 Modular Motion Style Adapters

Now, we empirically demonstrate the effectiveness of applying our motion style adapters on top of a modularized architecture on two setups: Scene generalization and Agent Motion generalization.

Setup: We modularize the Y-Net architecture similar to Fig. 2. We treat scene and agent motion independently for the first two layers of the encoder before fusing the learned representations. We refer to this design as Y-Net-Mod. The detailed design can be found in the supplementary. Given Y-Net-Mod, we consider five cases given the module list on which MoSA is applied: (1) scene motion 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].

Agent motion generalization: In $scene1$ of inD, cars and trucks share the same scene context differing only in their velocity distribution (see Fig. 6). Fig. 7 represents the performance of style

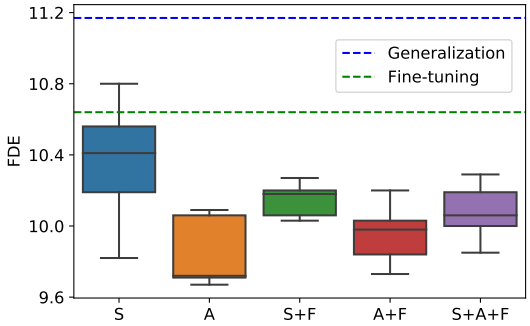

Figure 7: Agent Motion Transfer. Style Transfer from *cars* to *trucks* on InD scene1 with $N_{target} = 20$ samples using different MoSA configurations. [A] performs best while [S] worsens performance. Error reported is Top-20 FDE in pixels across 5 seeds.

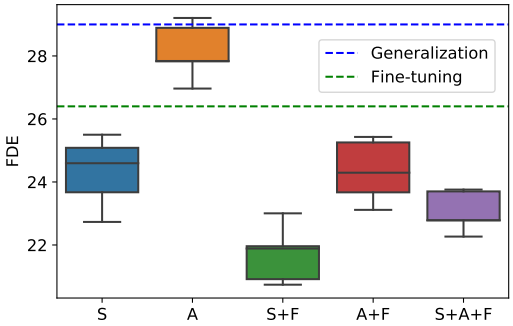

Figure 8: Scene Transfer. Style Transfer across scenes on InD *pedestrians* with $N_{target} = 20$ samples using different MoSA configurations.[S+F] performs best while [A] worsens performance. Error reported is Top-20 FDE in pixels across 5 seeds.

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]).

Scene generalization: We train the Y-Net-Mod model on pedestrian data on scene ids = $\{2, 3, 4\}$ and adapt it on $scene1$ of inD. Fig. 8 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 to boost the performance of MoSA.

Additional experiments and implementation details. Please refer to supplementary.

5 Limitations

In this work, we demonstrated the effectiveness of decoupling our proposed motion style adapters using Y-Net-Mod. However, during training, we do not enforce any constraints on the learning objective to favor effective modularization that can result in more efficient adaptation. Developing training strategies that allow quick adaptation using MoSA is a potential line of future work.

Another limitation of current work is the requirement of human intervention in determining the modules that need to be adapted given the nature of the style shift. A future direction of research is automating the selection of layers to be adapted for target domains.

6 Conclusion

In this work, we tackle 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 present 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 demonstrates the effectiveness of our approach.

References

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

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

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

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

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

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

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

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

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

[10] 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. *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 1929–1934, 2020.

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

[12] 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.

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

[14] G. Antonini and M. Bierlaire. Discrete choice models for pedestrian walking behavior. 2006.

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

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

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

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

[19] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22:1345–1359, 2010.

[20] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? *ArXiv*, abs/1411.1792, 2014.

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

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

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

[24] A. Radford and K. Narasimhan. Improving language understanding by generative pre-training. 2018.

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

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

[27] V. Dumoulin, J. Shlens, and M. Kudlur. A learned representation for artistic style. *ArXiv*, abs/1610.07629, 2017.

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

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

[30] K. Mangalam, Y. An, H. Girase, and J. Malik. From goals, waypoints & paths to long term human trajectory forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15233–15242, 2021.

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