Uncertainty-aware pseudo labels for domain adaptation in pedestrian trajectory prediction

Atanas Poibrenski, Farzad Nozarian, Farzaneh Rezaeianaran, Christian Müller

Abstract— **Learning-based trajectory prediction models are increasingly being used in a wide range of AI applications, such as autonomous driving. However, existing methods usually ignore the potential distribution shift between the train and test environments. This inevitably results in an increased prediction error in the new domain. Towards this end, we present a novel model-agnostic student-teacher model that leverages the recent advances in self-training and utilizes predicted pseudo trajectories from the target domain in order to improve its generalization capabilities. More specifically, we propose to train the model using both trajectories from the source domain and predicted pseudo trajectories from the target domain. Since the predicted trajectories can be noisy, we weigh them by the epistemic uncertainty of the model using MC-dropout, giving more weight to the more certain ones. Additionally, we show that the domain gap can be reduced further by augmenting the source data. Experiments on the ETH and UCY datasets show the effectiveness of our framework on domain adaptation for pedestrian trajectory prediction.**

Index Terms **- trajectory prediction, domain adaptation, pseudo-labels, self-training**

I. Introduction

Learning-based trajectory prediction methods are gaining more attention in recent years due to their potential use in areas such as autonomous driving [30], video surveillance [31], collision avoidance and planning in robotics [32]. However, existing methods usually require huge amounts of annotated data and tend to learn a generic motion pattern. Moreover, when a trained model on the original scenario (*i.e.,* source domain) is applied to a new scenario (*i.e.,* target domain), it may not generalize well and the prediction error increases [33]. As it can be observed in Fig. 1, pedestrian trajectory distributions may vary between different domains and adaptation is crucial for the successful transfer of models between different environments.

To this end, in this work, we present a model-agnostic learning framework that aims to bridge the domain gap between the source and target domain for the task of pedestrian trajectory prediction. Our method can be applied to any trajectory prediction model. More specifically, we explore the scenario where we have access to observed and annotated future trajectories in the source domain, but only the observed trajectories are available in the target domain. We start by training a model on the source domain in a supervised fashion. Then we switch to a self-training regime, utilizing both labeled and unlabeled trajectory data. For the labeled data we use both the original together with an augmented version of the source data. For the unlabeled data, the model predicts pseudo trajectories (labels) which are used for training and are weighted by the epistemic uncertainty of the model in the loss function. We show that predicted trajectories with less uncertainty have lower prediction error on average. The self-training is an iterative process where we utilize a student-teacher framework [34]. The student is initialized with the weights of the pre-trained model on the source data. At each training step with both labeled and unlabeled data, the teacher generates pseudo-labels to train the student and maintains an exponential moving average (EMA) of the student's weights. The teacher model is used for the final evaluation on the target domain.

In this work, we tackle the domain shift problem in trajectory prediction. To the best of our knowledge, this is the first work to use self-training with pseudo labels in such a setting. Our key contributions are fourfold.

- We propose to use a model-agnostic student-teacher framework for domain adaptation in pedestrian trajectory prediction.
- We propose to weight the pseudo trajectories (labels) by their variance, given by forwarding an input to the model multiple times and utilizing MC-Dropout.
- We show that augmenting the source data with rotations can significantly improve the generalization of the model.
- Experiments on five different datasets with different trajectory distributions verify the effectiveness of our method.

The remainder of this paper is structured as follows. Related work is summarized in section II, and our proposed solution is described and discussed in section III. This is followed by the comparative experimental performance evaluation in section IV before we conclude in section V.

II. Related Work

A. Trajectory Prediction

Trajectory prediction aims to predict the future locations of pedestrians based on their past locations and surroundings. Earlier works utilize Kalman Filters [1], Hidden Markov Models [2] and other mathematical models such as Gaussian Process [3] and Markov Decision Process [4] in order to make the prediction. Additionally, some works model the dynamics of pedestrians explicitly [10].

Recently, a plethora of deep learning solutions have been proposed. LSTM [5] is often used to encode the history of

Authors are with the German Research Center for Artificial Intelligence (DFKI). Saarland Informatics Campus, Stuhlsatzenhausweg 3, Saarbruecken, Germany. (email: `firstname.lastname@dfki.de`)

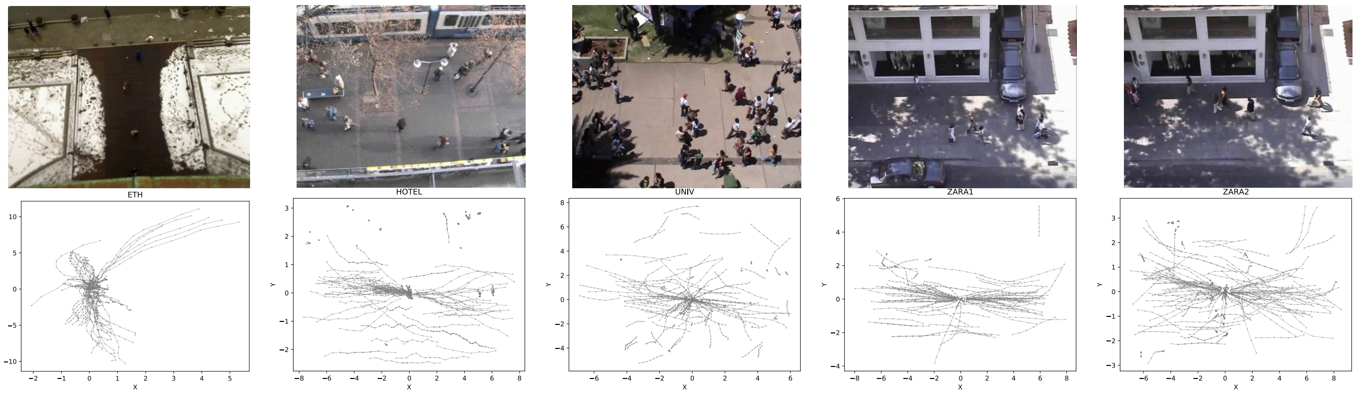


Fig. 1: Example frame for each dataset with their corresponding 2D trajectory distribution. Each scene has a unique distribution of pedestrian trajectories, due to the different static and dynamic objects present. ETH has the most distinct trajectory distribution due to its vertical orientation of walkable space. Datasets from left to right - ETH, HOTEL, UNIV, ZARA1, ZARA2.

the agent state and decode the future trajectory, respectively [6]. Due to the highly uncertain future, stochastic decoders are typically used to sample multiple trajectories, e.g., using GANs [7] or conditional variational autoencoders (CVAEs) [8], [14]. Alternatively, [9] proposed a network that predicts a multimodal distribution over the future locations of the agent. Motivated by the Language Processing field, some works utilize the Transformer architecture [11] for the trajectory prediction task [12]. More recently, some works predict the future trajectories of pedestrians by estimating and using their final goals [13]. Different from recent methods, our work tackles the domain adaptation problem in pedestrian trajectory prediction.

B. Domain Adaptation

Domain Adaptation (DA) is a technique used to address the domain shift challenge that arises when attempting to transfer knowledge from one domain to another that is similar but distinct [15]. Recently, it has attracted a lot of attention, motivating a considerable amount of works, especially in the classification [16], object detection [18] and semantic segmentation [17] tasks.

The most commonly used methods in DA include adaptation in the input space, feature space, output space, and model-based adaptation. The objective of DA in the input space is to produce samples from the source domain that closely resemble the target domain [19]. Other domain adaptation methods unify the source and target domains by creating a domain-invariant feature representation. The source and target inputs are projected into a shared feature space and their distributions are aligned by minimizing a distance measure such as CORAL [20], maximum mean discrepancy (MMD) [21], or adversarial loss [22]. Pseudo-label-based [23] domain adaptation methods are the most common domain adaptation methods in the output space. A model is trained in a supervised fashion with labeled and unlabeled data simultaneously. In scenarios where the data is unlabeled, a technique called Pseudo-Labeling is employed. This technique involves selecting the class with the highest predicted probability during each weight update and treating it as if it were a true label. These pseudo-labels are then used as a substitute for the actual labels in training the model. Lastly, model-based domain adaptation approaches reduce the domain shift of a model by imposing constraints on its parameters [24]. In this work, we focus on pseudo-labels as well as augmenting the input space of the source domain for domain adaptation.

Furthermore, DA can be divided into unsupervised and semi-supervised DA. In unsupervised DA [25], there is no access to labeled samples from the target domain, in contrast to semi-supervised DA [26], where a small portion of labeled samples from the target domain is accessible during training. In this work, we focus on the unsupervised DA setting.

The area of domain adaptation in trajectory prediction has not been extensively researched within the scientific community. Recently, [27] proposes a cross-domain trajectory prediction network which encodes observed trajectories from the source and target domains, then their features are aligned by a cross-domain feature discriminator. In [28], a domain-invariant graph neural network (GNN) is used to explore the structural motion knowledge in trajectory prediction, where the domain-specific knowledge is reduced. Alternatively, [29] uses meta-learning, where Bayesian regression is employed to incorporate an adaptive layer into existing trajectory prediction models, enabling efficient domain transfer through offline fine-tuning, online adaptation, or a combination of both. However, to the best of our knowledge, there is no other work that utilizes self-training and pseudo-labels for domain adaptation in the task of trajectory prediction.

III. Proposed Method

Our proposed framework is illustrated in Fig. 2. For both the student and the teacher models, we use the recent state-of-the-art trajectory prediction model SGNet [13].

A. Problem Description

At time step t, given the observed trajectory $K_t^i = \{O_{t-e+1}^i, O_{t-e+2}^i, ..., O_t^i\}$ of a pedestrian i in the last time

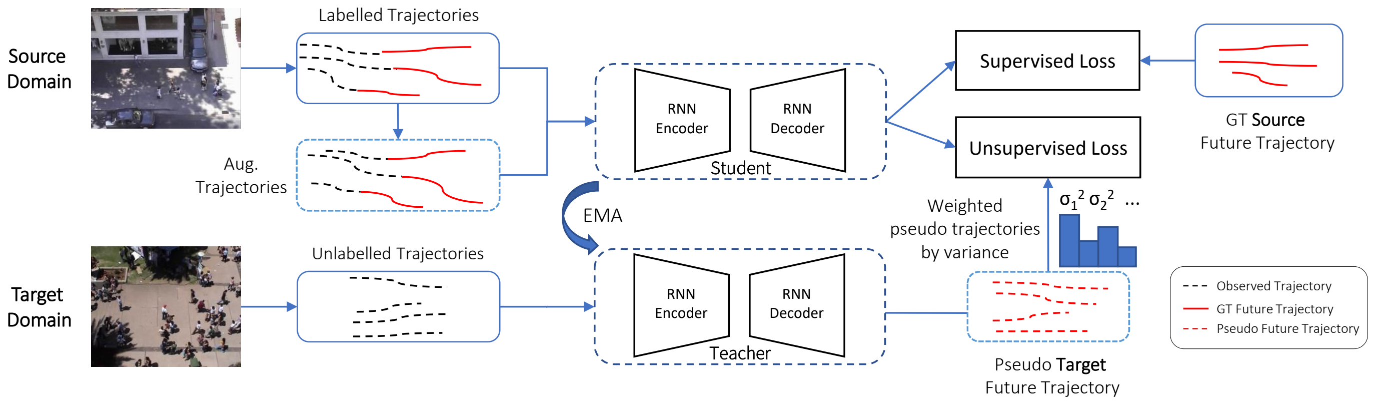


Fig. 2: Architecture of our proposed training framework. Given the validation dataset from the target domain and the training dataset from the source domain, unlabeled (only observed part) and labeled (observed + future) trajectories are extracted, respectively. The set of labeled trajectories is further enlarged through augmentation. All the trajectories are then fed into the student-teacher EMA model self-training loop. The student-teacher model tries to learn an observed trajectory's continuation (future). The supervised loss is calculated using both the ground truth future trajectories from the source domain together with self-predicted pseudo future trajectories from the target domain. Since the pseudo trajectories are noisy, they are weighted by their uncertainty (inverse variance). The final teacher is used for inference on the target domain's test set.

steps e, the goal is to predict the future trajectory $\bar{K}_t^i = \{O_{t+1}^i, O_{t+2}^i, ..., O_{t+d}^i\}$ in the next time steps d, where $O_t^i = (x_t^i, y_t^i) \in \mathbb{R}^2$ denotes the 2D coordinates of a pedestrian. Formally, the aim is to find the parameters w^* of a model $f(\cdot)$, that, given the observed trajectory of a pedestrian, will predict the future trajectory as:

$$\tilde{K}_t^i = f(K_t^i, w^*) \tag{1}$$

During training, we assume access to a dataset $D_\mathcal{S} = \{K^j, \bar{K}^j\}_{j=1}^{|D_\mathcal{S}|}$, which contains observed-future trajectory pairs for all pedestrians in the source domain $\mathcal{S}$. Additionally, we have access to the observed trajectories from the target validation dataset $D_{\mathcal{T}val} = \{K^j\}_{j=1}^{|D_{\mathcal{T}val}|}$. The aim is to obtain a model that can be effectively deployed to the target test dataset $D_{\mathcal{T}test}$.

B. Model Pretraining and Data Augmentation

We start by pretraining our model SGNet [13] on the given source domain dataset. The model consists of a recurrent neural network and estimates and uses the final locations of each pedestrian at multiple temporal scales. In particular, it utilizes an encoder that captures historical information, a stepwise goal estimator that predicts successive goals into the future, and a decoder that predicts the future trajectory. We use stochastic gradient descent to find the parameters of the model w, by minimizing the Root Mean Square Error (RMSE):

$$RMSE(D_\mathcal{S}) = \sqrt{\frac{1}{|D_\mathcal{S}|}\sum_{i=1}^{|D_\mathcal{S}|}(\tilde{K}_i - \bar{K}_i)^2} \tag{2}$$

given the observed and future trajectory pairs from the source domain $D_\mathcal{S}$, where $\tilde{K}$ and $\bar{K}$ are the predicted and ground truth future trajectories. We employ early-stopping [42] to prevent the model from overfitting on the training data by utilizing a small validation set from the source domain.

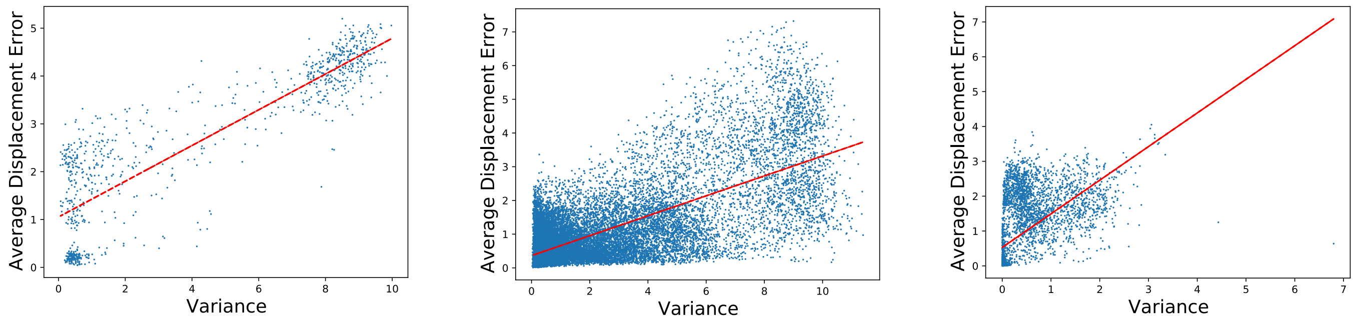
To improve the generalization capabilities of the model further, we enlarge the domain trajectory source dataset $D_\mathcal{S}$ by augmenting it via rotations. A rotation matrix R is defined as:

$$R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}, \tag{3}$$

where θ is uniformly random generated from the interval $[-\frac{\pi}{3}, \frac{\pi}{3}]$. We multiply the rotation matrix R with every 2D coordinate at each time step for each trajectory in the domain dataset $D_\mathcal{S}$, effectively rotating each trajectory by a random angle between $-60°$ and $60°$. This way, we acquire a new rotated dataset $D_{\mathcal{S}rot}$, which has the same cardinality as $D_\mathcal{S}$. The pretrained model weights w, together with the datasets $D_\mathcal{S}$, $D_{\mathcal{S}rot}$ and $D_{\mathcal{T}val}$, will be used for self-training in Section III-D.

C. Uncertainty Estimation

The future trajectories that will be used for self-training in the dataset $D_{\mathcal{T}val}$ need to be predicted by our model. Therefore, we propose measuring those predictions' reliability by quantifying the model uncertainty during forecasting using stochastic approximation. In general, the model uncertainty, or epistemic uncertainty can be estimated from the mean and variance of the marginal distribution $p(y^*|x^*, X, Y)$ of a Bayesian neural network, where X and Y represent the training input and output samples while y^* represents the predicted output for some new test input x^*. The marginal distribution depends on the posterior $p(\omega|X, Y)$, where ω represents the network's weights. Since this posterior is intractable, we can use Monte Carlo dropout as a variational approximation to such Bayesian Neural Networks [43] without modifying our existing network architecture. We apply MC Dropout by performing multiple stochastic



(a) ADE vs Var(ETH→HOTEL) (b) ADE vs Var (ETH→UNIV) (c) ADE vs Var (HOTEL→ZARA2)

Fig. 3: Visualization of the Average Displacement Error (ADE) of the pseudo trajectories against their variance (Var.) on 3 chosen tasks. The line of best fit (red line) shows the linear correlation between variance and error. As the variance of the pseudo trajectories increases, their error also increases on average.

forward passes with the means of activated dropout in our model during the inference phase. The probability of dropout is set as $p = 0.6$ and the inference is run $N = 20$ times to obtain a set of predicted trajectories $\{\tilde{K}_1^*, \tilde{K}_2^*, ..., \tilde{K}_N^*\}$ for each observed pedestrian trajectory K^* in $D_{\mathcal{T}val}$. We can then estimate the mean μ and variance σ^2 of the predictions, where the variance indicates model uncertainty:

$$\mu = \frac{1}{N}\sum_{i=1}^{N}\tilde{K}_i^* \tag{4}$$

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(\tilde{K}_i^* - \mu)^2 \tag{5}$$

For the final prediction of each observed trajectory, we pick one at random from the 20 predictions. We found this to perform better than taking the average. Additionally, we store the variance σ^2 of each trajectory. Our target validation set now has pseudo labels (future trajectories) together with the observed trajectories $D_{\mathcal{T}val} = \{K^j, \tilde{K}^j\}_{j=1}^{|D_{\mathcal{T}val}|}$, which will be used in the next self-training step.

D. Student-teacher self-training

We begin by initializing identical student and teacher networks $f_{student}$ and $f_{teacher}$ with the pre-trained weights w as described in Section III-B. We continue with a self-training loop for 100 epochs. At each epoch, we first augment the source trajectory dataset $D_{\mathcal{S}}$ with rotations to generate a new dataset $D_{\mathcal{S}rot}$, as illustrated in Section III-B. Using the observed trajectories from the validation set of the target domain $D_{\mathcal{T}val}$, we predict pseudo future trajectories with the current teacher model by employing MC-dropout and storing the variance σ^2 of each pseudo trajectory (see Section III-C). The student $f_{student}$ is trained using both supervised and unsupervised loss. The supervised loss is calculated with samples from $D_{\mathcal{S}}$ and $D_{\mathcal{S}rot}$, while the unsupervised loss is calculated with samples from $D_{\mathcal{T}val}$. To prioritize the learning on target data, we assign a weight λ to the unsupervised loss. The total loss function L_{total} is defined as:

$$\mathrm{L}_{total} = RMSE(D_{\mathcal{S}}) + \lambda RMSE_{\sigma^2}(D_{\mathcal{T}val}) + RMSE(D_{\mathcal{S}rot}) \tag{6}$$

where $RMSE$ is defined in Eq. 2. $RMSE_{\sigma^2}$ is the weighted Root Mean Square Error:

$$RMSE_{\sigma^2}(D_{\mathcal{T}val}) = \sqrt{\frac{1}{|D_{\mathcal{T}val}|}\sum_{i=1}^{|D_{\mathcal{T}val}|}\frac{1}{\sigma_i^2}(\tilde{K}_i - \bar{K}_i)^2} \tag{7}$$

where $\tilde{K}$ is the current predicted trajectory, $\bar{K}$ is the predicted future trajectory from the previous epoch, and σ^2 is the variance of the predicted trajectory from the previous epoch. The predicted (pseudo) trajectories $\bar{K}$ cannot be used directly as ground truth labels, since they are noisy. Therefore, we weight their contribution by their uncertainty *i.e.,* by their inverse variance. This weighting is motivated by Fig. 3, where the linear relationship between the variance of the pseudo trajectories and their prediction error is shown. The teacher $f_{teacher}$ weights are then updated as an Exponential Moving Average (EMA) of the student weights:

$$f_{teacher}^i = f_{teacher}^{i-1} + (1-\alpha)f_{student}^i \tag{8}$$

where i is the current epoch and α is a smoothing coefficient hyperparameter. Both student and teacher model outputs can be used for prediction, but at the end of the training, the teacher prediction is more likely to be correct. The complete self-training algorithm is outlined below in Alg. 1.

IV. Evaluation

A. Experimental Setting

*1) **Datasets**:* We evaluate our proposed framework on two publicly available datasets: ETH [35] and UCY [36]. As illustrated in Fig. 1, we conduct our experiments on the ETH-eth, ETH-hotel scenes as well as on the UCY-uni, UCY-zara1 and UCY-zara2 scenes. These datasets are used as a standard trajectory prediction benchmark and comprise of trajectories captured at 2.5 Hz ($\Delta t = 0.4s$). In total, there are 1536 pedestrians and 4 unique environments. The

Algorithm 1: Student-teacher self-training

```
1  Pretrain a network $f_\theta$, using the trajectory samples from $D_\mathcal{S}$
2  $f^0_{student} \leftarrow f_\theta$
3  $f^0_{teacher} \leftarrow f_\theta$
4  for $i \leftarrow 1$ to $MaxIterations$ do
5      $D_{\mathcal{S}rot} \leftarrow$Augment $D_\mathcal{S}$          // Section III-B
6      $\tilde{K} \leftarrow f^i_{teacher}(K)$          // Section III-C
7      $D_{\mathcal{T}val} \leftarrow \{K, \tilde{K}\}$
8      $\tilde{D} \leftarrow D_\mathcal{S} \cup D_{\mathcal{S}rot} \cup D_{\mathcal{T}val}$
9      Train $f^i_{student}$ using samples from $\tilde{D}$          // Eq. 6
10     $f^i_{teacher} \leftarrow f^{i-1}_{teacher} + (1-\alpha) f^i_{student}$          // Eq. 8
11 return $f_{teacher}$
```

trajectory distributions from Fig.1 together with the statistics from Table I show the different biases present in the datasets used.

TABLE I: Statistics from the five different datasets used. *seqNum* denotes the number of sequences to be predicted, *avgNum* denotes the average number of pedestrians per sequence, *avgVel* denotes the average pedestrian velocity (m/s), *avgAcc* denotes the average pedestrian acceleration (m/s^2) and std denotes the standard deviation.

Metric	ETH	HOTEL	UNIV	ZARA1	ZARA2	std
seqNum	70	301	947	602	921	383.63
avgNum	2.59	3.50	25.70	3.74	6.33	9.79
avgVel	0.44	0.18	0.20	0.37	0.20	0.12
avgAcc	0.13	0.06	0.04	0.04	0.03	0.04

*2) **Evaluation Protocol:*** For evaluation, we follow the protocol introduced in [28]. We use each of the 5 scenes as a training source domain $\mathcal{S} \in \{eth, hotel, univ, zara1, zara2\}$ and evaluate our model on the other 4 scenes as the target domain $\mathcal{T}$, where $\mathcal{S} \neq \mathcal{T}$. This results in 20 different cross-domain pairings: A $\rightarrow$ B/C/D/E, B $\rightarrow$ A/C/D/E, C $\rightarrow$ A/B/D/E, D $\rightarrow$ A/B/C/E and E $\rightarrow$ A/B/C/D, where A, B, C, D, and E denote ETH, HOTEL, UNIV, ZARA1, and ZARA2, respectively. During training, our model has access to the source trajectory domain together with only the observed trajectory from the validation set of the target domain, as described in Section III-A. The test set of the target domain is used for final evaluation. It should be noted that the validation and test sets of the target domain are independent of each other, and there are no overlapping trajectory samples.

*3) **Baselines:*** Our first five baselines, namely **Social-STGCNN** [37], **PECNet** [38], **RSBG** [39], **Tra2Tra** [40] and **SGCN** [41] are trained on the source domain and evaluated on the target domain. **T-GNN** [28] has access to both the source domain and the observed trajectory from the validation set of the target domain for adaptation. **Trajectron++** [29] proposes offline adaptation, where a small dataset from the target environment is assumed to be provided, as well as online adaptation, where the model is gradually adapted to the new domain as new target data comes in. We use their best results for comparison. We train our base model SGNet [13] on the source domain and evaluate it on the target domain. Additionally, we create an SGNet oracle that has access to the source domain and the full validation set of the target domain. Each of the baselines samples 20 predictions, and the best is used for the final evaluation on the test set of the target domain.

*4) **Evaluation Metrics:*** Following previous works [13], [28], [29], we adopt two metrics for the performance evaluation. *Average Displace Error (ADE)* is the mean-squared error between the ground truth and the predicted trajectory at each time step:

$$ADE = \frac{\sum_{i=1}^{N} \sum_{t=obs+1}^{pred} \|\bar{K}^i_t - \tilde{K}^i_t\|_2}{N * pred} \tag{9}$$

Final Displacement Error (FDE) is the Euclidean distance between the ground truth and the prediction at the final time step:

$$FDE = \frac{\sum_{i=1}^{N} \|\bar{K}^i_{pred} - \tilde{K}^i_{pred}\|_2}{N} \tag{10}$$

where N is the total number of pedestrians in the target domain, obs is the number of observed frames, $pred$ is the number of predicted frames, and $\tilde{K}^i_t$ and $\bar{K}^i_t$ are predicted and ground-truth trajectory coordinates, respectively.

*5) **Implementation Details:*** Similar to all baselines, 8 frames are observed (3.2 seconds) and 12 frames (4.8 seconds) in the future are predicted. In our experiments, we set $\alpha = 0.99$ and $\lambda = 2$. All the training is done with a learning rate of 1×10^{-4}, batch size of 128, and the Adam [44] optimizer. We use the default network parameters as in the original SGNet [13].

B. Results

The experimental results are shown in Table II and Table III. Our proposed method achieves the lowest error on average and is able to outperform all baselines on almost all domain adaptation tasks. Results indicate that training with pseudo-labeled trajectories from the target domain together with a mixture of ground truth and augmented source trajectories can greatly improve the domain adaptation. However, **Trajectron++** [29] outperforms our method the most in tasks A2B and B2A due to the fact that their method has access to a small set of target trajectories. Additionally, scenes A and B have the biggest domain shift between each other, since A has mostly vertical trajectories and scene B has mostly horizontal ones (see Fig. 1). Moreover, there is still a gap between our method and the proposed oracle, which has access to the validation set of the target domain. We believe this is due to the noisy pseudo trajectories used for training. Despite the weighting based on their variance or uncertainty, there are many pseudo trajectories that have low variance and high error and vice versa, as illustrated in Fig. 3. Additional source augmentation can further help in reducing this gap. Moreover, in some tasks such as E2D and D2E, all baselines have relatively small error and the gap between the oracle and our method is small as well. This is due to the similar physical environments of scene D (ZARA1) and scene E (ZARA2) which result in similar trajectory distributions, as

TABLE II: ADE (Average Displacement Error) Our proposed framework compared to other baselines on 20 different domain adaptation settings. “2” represents from source domain to target domain. A, B, C, D, and E denote ETH, HOTEL, UNIV, ZARA1, and ZARA2 datasets

Method	A2B	A2C	A2D	A2E	B2A	B2C	B2D	B2E	C2A	C2B	C2D	C2E	D2A	D2B	D2C	D2E	E2A	E2B	E2C	E2D	AVG
Social-STGCNN [37]	1.83	1.58	1.30	1.31	3.02	1.38	2.63	1.58	1.16	0.70	0.82	0.54	1.04	1.05	0.73	0.47	0.98	1.09	0.74	0.50	1.22
PECNet [38]	1.97	1.68	1.24	1.35	3.11	1.35	2.69	1.62	1.39	0.82	0.93	0.57	1.10	1.17	0.92	0.52	1.01	1.25	0.83	0.61	1.31
RSBG [39]	2.21	1.59	1.48	1.42	3.18	1.49	2.72	1.73	1.23	0.87	1.04	0.60	1.19	1.21	0.80	0.49	1.09	1.37	1.03	0.78	1.38
Tra2Tra [40]	1.72	1.58	1.27	1.37	3.32	1.36	2.67	1.58	1.16	0.70	0.85	0.60	1.09	1.07	0.81	0.52	1.03	1.10	0.75	0.52	1.25
SGCN [41]	1.68	1.54	1.26	1.28	3.22	1.38	2.62	1.58	1.14	0.70	0.82	0.52	1.05	0.97	0.80	0.48	0.97	1.08	0.75	0.51	1.22
T-GNN [28]	1.13	1.25	0.94	1.03	2.54	1.08	2.25	1.41	0.97	0.54	0.61	0.23	0.88	0.78	0.59	0.32	0.87	0.72	0.65	0.34	0.96
Trajectron++ [29]	**0.33**	**0.56**	0.50	**0.38**	**0.80**	0.60	**0.43**	0.31	1.03	0.41	0.41	0.38	0.93	0.32	0.48	0.35	0.91	0.31	0.49	0.44	0.52
SGNet [13]	2.75	1.14	0.68	0.67	1.21	0.81	1.50	0.92	0.41	0.22	**0.17**	**0.14**	0.47	0.57	0.42	**0.18**	0.47	0.57	0.39	0.21	0.69
SGNet (Ours)	1.04	0.59	**0.32**	0.40	0.99	**0.38**	0.64	**0.29**	**0.39**	**0.15**	**0.17**	**0.14**	**0.42**	**0.25**	**0.33**	**0.18**	**0.43**	**0.20**	**0.29**	**0.20**	**0.39**
SGNet (Oracle)	*0.17*	*0.26*	*0.32*	*0.19*	*0.56*	*0.27*	*0.33*	*0.19*	*0.35*	*0.11*	*0.16*	*0.12*	*0.41*	*0.14*	*0.24*	*0.15*	*0.38*	*0.14*	*0.24*	*0.20*	*0.25*

TABLE III: FDE (Final Displacement Error) Our proposed framework compared to other baselines on 20 different domain adaptation settings. “2” represents from source domain to target domain. A, B, C, D, and E denote ETH, HOTEL, UNIV, ZARA1, and ZARA2 datasets.

Method	A2B	A2C	A2D	A2E	B2A	B2C	B2D	B2E	C2A	C2B	C2D	C2E	D2A	D2B	D2C	D2E	E2A	E2B	E2C	E2D	AVG
Social-STGCNN [37]	3.24	2.86	2.53	2.43	5.16	2.51	4.86	2.88	2.30	1.34	1.74	1.10	2.21	1.99	1.41	0.88	2.10	2.05	1.47	1.01	2.30
PECNet [38]	3.33	2.83	2.53	2.45	5.23	2.48	4.90	2.86	2.22	1.32	1.68	1.12	2.20	2.05	1.52	0.88	2.10	1.84	1.45	0.98	2.29
RSBG [39]	3.42	2.96	2.75	2.50	5.28	2.59	5.19	3.10	2.36	1.55	1.99	1.37	2.28	2.22	1.77	0.97	2.19	2.29	1.81	1.34	2.50
Tra2Tra [40]	3.29	2.88	2.66	2.45	5.22	2.50	4.89	2.90	2.29	1.33	1.78	1.09	2.26	2.12	1.63	0.92	2.18	2.06	1.52	1.17	2.34
SGCN [41]	3.22	2.81	2.52	2.40	5.18	2.47	4.83	2.85	2.24	1.32	1.71	1.03	2.23	1.90	1.48	0.97	2.10	1.95	1.52	0.99	2.29
T-GNN [28]	2.18	2.25	1.78	1.84	4.15	1.82	4.04	2.53	1.91	1.12	1.30	0.87	1.92	1.46	1.25	0.65	1.86	1.45	1.28	0.72	1.82
Trajectron++ [29]	**0.65**	1.18	1.06	0.81	**1.59**	1.19	**0.89**	0.64	1.98	0.81	0.93	0.84	1.81	0.57	1.01	0.70	1.71	0.54	1.02	0.96	1.04
SGNet [13]	5.10	1.99	1.10	1.03	2.22	1.67	3.27	1.92	0.78	0.50	**0.35**	**0.28**	0.88	1.15	0.84	0.33	0.88	1.20	0.77	0.38	1.33
SGNet (Ours)	1.77	**1.02**	**0.51**	**0.59**	1.82	**0.71**	1.38	**0.55**	**0.71**	**0.29**	**0.35**	**0.28**	**0.71**	**0.47**	**0.60**	**0.31**	**0.73**	**0.37**	**0.56**	**0.35**	**0.71**
SGNet (Oracle)	*0.24*	*0.46*	*0.55*	*0.32*	*0.99*	*0.51*	*0.62*	*0.33*	*0.63*	*0.19*	*0.30*	*0.23*	*0.69*	*0.23*	*0.45*	*0.27*	*0.63*	*0.22*	*0.45*	*0.35*	*0.43*

shown in Fig. 1. Alternatively, scene transfers from and to A (ETH) have a larger error on average since the distribution of trajectories is more vertical compared to the other scenes, as depicted in Fig. 1. This distribution gap further illustrates the need for domain adaptation techniques.

C. Ablation Study

In this section, we study the contribution of each component in our proposed method. This is achieved by measuring the average error of the model after the addition of each component, as shown in Table IV. For all variants, we use the same evaluation as in Section IV-A, where 20 future trajectories are predicted, and the best one is chosen for the error computation. We start by measuring the average ADE and FDE of our baseline SGNet [13] on the 20 different domain adaptation tasks. Next, we add MC-Dropout (M) to the pre-trained baseline on the source domain, in order to obtain 20 different predictions on the target domain, during inference (Section III-C). This results in a 5.79% improvement in ADE and 6.76% improvement in FDE compared to the baseline. Incorporating Self-training (S) into the model reduces the error further by 13.04% in ADE and 15.03% in FDE compared to the baseline. Finally, adding Augmentation (A) to the source domain results in an overall improvement of 43.47% in ADE and 46.61% in FDE. The results indicate the effectiveness of each component. We believe that self-training can reduce the domain gap even further by focusing on methods for reducing bad pseudo trajectories with high error. This is left for future work as well as for the trajectory prediction community for further research.

V. Conclusion

In this work, we have presented a model-agnostic student-teacher self-training framework, that leverages pseudo trajectories from the target domain in order to bridge the domain gap for the task of trajectory prediction. We have also shown an approach to weighting the pseudo trajectories during self-training based on their uncertainty. Additionally, we show how simple data augmentation can improve the domain adaptation further. Our experiments and ablation study confirm the effectiveness of the proposed method. We believe that pedestrian trajectory prediction is an important problem in the context of autonomous vehicles. Successful generalization of pedestrian trajectory prediction across different environments will directly translate to better motion planning of an autonomous vehicle. A limitation of the proposed approach is the assumption of the existence of observed trajectory data from the target domain. Ongoing work is concerned with further improving the quality of pseudo trajectories by various methods such as filtering as well as extending our experiments to different road agents (e.g. vehicles, cyclists), to different trajectory prediction models and datasets.

TABLE IV: Ablation study of the different components in our proposed method. ”M” denotes MC-Dropout, ”S” denotes Self-training, and ”A” denotes Augmentation.

Variant	Error (ADE/FDE)
SGNet [13]	0.69/1.33
SGNet + M	0.65/1.24
SGNet + M + S	0.60/1.13
SGNet + M + S + A	**0.39/0.71**

Acknowledgment

This research was supported by the German Federal Ministry for Education and Research (BMB+F) in the project MOMENTUM.

References

[1] Rudolf E. Kálmán. 1960. A New Approach to Linear Filtering and Prediction

[2] Dimitrios Makris and Tim J. Ellis. 2002. Spatial and Probabilistic Modelling of Pedestrian Behaviour. In BMVC.

[3] Carl Edward Rasmussen and Christopher K. I. Williams. 2005. Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning). The MIT Press.

[4] Dimitrios Makris and Tim Ellis. Spatial and probabilistic modelling of pedestrian behaviour. In Proceedings of the British Machine Vision Conference, pages 54.1–54.10, 2002. 3

[5] Sepp Hochreiter and Jurgen Schmidhuber. Long short-term memory. ¨ Neural computation, 9(8): 1735–1780, 1997

[6] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 961–971, 2016.

[7] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social GAN: Socially acceptable trajectories with generative adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2255–2264, 2018.

[8] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In Proceedings of the European Conference on Computer Vision (ECCV), pp. 683–700, 2020. Vision (2019)

[9] Osama Makansi, Eddy Ilg, Ozg ¨ un C¸ ic¸ek, and Thomas Brox. Overcoming limitations of mixture den- ¨ sity networks: A sampling and fitting framework for multimodal future prediction. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7137–7146, 2019

[10] Helbing and Molnár. "Social force model for pedestrian dynamics." Physical review. E, Statistical physics, plasmas, fluids, and related interdisciplinary topics 51 5 (1995): 4282-4286 .

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems, pages 5998–6008, 2017.

[12] Giuliari, Francesco et al. "Transformer Networks for Trajectory Forecasting." 2020 25th International Conference on Pattern Recognition (ICPR) (2020): 10335-10342.

[13] Wang, Chuhua et al. "Stepwise Goal-Driven Networks for Trajectory Prediction." IEEE Robotics and Automation Letters PP (2021): 1-1.

[14] Poibrenski, Atanas et al. "M2P3: multimodal multi-pedestrian path prediction by self-driving cars with egocentric vision." Proceedings of the 35th Annual ACM Symposium on Applied Computing (2020): n. pag.

[15] Zhang, Youshan. "A Survey of Unsupervised Domain Adaptation for Visual Recognition." ArXiv abs/2112.06745 (2021): n. pag.

[16] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada, "Maximum classifier discrepancy for unsupervised domain adaptation," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 3723–3732

[17] S. Sankaranarayanan, Y. Balaji, A. Jain, S. Nam Lim, and R. Chellappa, "Learning from synthetic data: Addressing domain shift for semantic segmentation," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 3752–3761

[18] Xu, Mengde et al. "End-to-End Semi-Supervised Object Detection with Soft Teacher." 2021 IEEE/CVF International Conference on Computer Vision (ICCV) (2021): 3040-3049.

[19] Hazan, Alon et al. "AdapterNet - learning input transformation for domain adaptation." ArXiv abs/1805.11601 (2018): n. pag.

[20] Sun, Baochen and Kate Saenko. "Deep CORAL: Correlation Alignment for Deep Domain Adaptation." ECCV Workshops (2016).

[21] Long, Mingsheng et al. "Learning Transferable Features with Deep Adaptation Networks." ArXiv abs/1502.02791 (2015): n. pag.

[22] Tzeng, Eric et al. "Adversarial Discriminative Domain Adaptation." 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017): 2962-2971.

[23] Lee, Dong-Hyun. "Pseudo-Label : The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks." (2013).

[24] Khan, M.A.A.H.; Roy, N.; Misra, A. Scaling human activity recognition via deep learning-based domain adaptation. In Proceedings of the International Conference on Pervasive Computing and Communications, Athens, Greece, 19–23 March 2018; pp. 1–9

[25] Liu, Xiaofeng et al. "Deep Unsupervised Domain Adaptation: A Review of Recent Advances and Perspectives." ArXiv abs/2208.07422 (2022): n. pag.

[26] Saito, Kuniaki et al. "Semi-Supervised Domain Adaptation via Minimax Entropy." 2019 IEEE/CVF International Conference on Computer Vision (ICCV) (2019): 8049-8057.

[27] Huang, Pingxuan et al. "Cross-domain Trajectory Prediction with CTP-Net." CAAI International Conference on Artificial Intelligence (2021).

[28] Xu, Yi et al. "Adaptive Trajectory Prediction via Transferable GNN." 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2022): 6510-6521.

[29] Ivanovic, B. et al. "Expanding the Deployment Envelope of Behavior Prediction via Adaptive Meta-Learning." ArXiv abs/2209.11820 (2022): n. pag.

[30] Liu, Jianbang et al. "A Survey on Deep-Learning Approaches for Vehicle Trajectory Prediction in Autonomous Driving." 2021 IEEE International Conference on Robotics and Biomimetics (ROBIO) (2021): 978-985.

[31] Kanu-Asiegbu, Asiegbu Miracle et al. "Leveraging Trajectory Prediction for Pedestrian Video Anomaly Detection." 2021 IEEE Symposium Series on Computational Intelligence (SSCI) (2021): 01-08.

[32] Zhu, Hai et al. "Learning Interaction-Aware Trajectory Predictions for Decentralized Multi-Robot Motion Planning in Dynamic Environments." IEEE Robotics and Automation Letters 6 (2021): 2256-2263.

[33] Zhuang, Fuzhen et al. "A Comprehensive Survey on Transfer Learning." Proceedings of the IEEE 109 (2019): 43-76.

[34] Tarvainen, Antti and Harri Valpola. "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results." NIPS (2017).

[35] Pellegrini, S., Ess, A., Schindler, K., Van Gool, L.: You'll never walk alone: Modeling social behavior for multi-target tracking. In: ICCV. pp. 261–268. IEEE (2009)

[36] Lerner, A., Chrysanthou, Y., Lischinski, D.: Crowds by example. In: Computer Graphics Forum. vol. 26, pp. 655–664. Wiley Online Library (2007)

[37] Abduallah Mohamed, Kun Qian, Mohamed Elhoseiny, and Christian Claudel. Social-STGCNN: A social spatiotemporal graph convolutional neural network for human trajectory prediction. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 14424– 14432, 2020.

[38] Karttikeya Mangalam, Harshayu Girase, Shreyas Agarwal, Kuan Hui Lee, Ehsan Adeli, Jitendra Malik, and Adrien Gaidon. It is not the journey but the destination: Endpoint conditioned trajectory prediction. In Proceedings of the European Conference on Computer Vision, pages 759– 776, 2020

[39] Jianhua Sun, Qinhong Jiang, and Cewu Lu. Recursive social behavior graph for trajectory prediction. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 660–669, 2020.

[40] Yi Xu, Dongchun Ren, Mingxia Li, Yuehai Chen, Mingyu Fan, and Huaxia Xia. Tra2tra: Trajectory-to-trajectory prediction with a global social spatial-temporal attentive neural network. IEEE Robotics and Automation Letters, 6(2):1574– 1581, 2021

[41] Liushuai Shi, Le Wang, Chengjiang Long, Sanping Zhou, Mo Zhou, Zhenxing Niu, and Gang Hua. Sgcn: Sparse graph convolution network for pedestrian trajectory prediction. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 8994–9003, 2021

[42] Caruana, Rich et al. "Overfitting in Neural Nets: Backpropagation, Conjugate Gradient, and Early Stopping." NIPS (2000).

[43] Gal, Yarin and Zoubin Ghahramani. "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning." ArXiv abs/1506.02142 (2015): n. pag.

[44] P. Kingma Diederik and Ba Jimmy. Adam: A method for stochastic optimization. In Proceedings of the International Conference on Learning Representations, 2015