2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Task-Driven Domain-Agnostic Learning with Information Bottleneck for Autonomous Steering

Yu Shen, Laura Zheng, Tianyi Zhou and Ming C. Lin

***Abstract*— Environments for autonomous driving can vary from place to place, leading to challenges in designing a learning model for a new scene. Transfer learning can leverage knowledge from a learned domain to a new domain with limited data. In this work, we focus on end-to-end autonomous driving as the target task, consisting of both *perception* and *control*. We first utilize *information bottleneck analysis* to build a *causal graph* that defines our framework and the loss function; then we propose a novel *domain-agnostic learning* method for autonomous steering based on our analysis of training data, network architecture, and training paradigm. Experiments show that our method outperforms other SOTA methods.**

I. Introduction

Autonomous driving (AD) has the potential to create safer and more efficient transportation systems by reducing congestion and accidents due to human errors. Central to AD, autonomous steering is a complex task and requires the choreography of many components to operate. One essential component is the perception-control module that maps sensor data to control commands (e.g., steering angles). With recent advances in machine learning, especially deep learning [27], the perception-control module is increasingly enabled by learning-based algorithms, which leverage multimodal input from sensors including cameras, Lidar, and radar to navigate autonomous vehicles (AVs). While each type of sensor offers its unique strength in detecting the environment, the RGB camera is one of the most universal and accessible sensors due to its rich visual information and affordable cost.

Many real-world images are collected for training AVs. Example datasets include KITTI [14], NVIDIA [7], Waymo Open Dataset [38], CityScapes [10], and BDD100K [44]. In addition to real-world images, simulators or synthesis images are also heavily used in training AVs [6]. Example simulation platforms include CARLA [11], the Udacity Self-Driving Car Simulator [1], and NVIDIA Drive Constellation [2]. Different datasets have different data distributions, resulting in domain gaps – between real-world domains(e.g., different roads, traffic conditions, and/or sensors, etc.) or between real and virtual domains (e.g., different image styles, driving scenarios, etc.). In real-world applications, usually, only limited data is available in a new scenario. Ideally, it is desirable to use existing data from the source domain to help the learning in the target scenario. The challenge is how to utilize both rich source domain data and limited target domain data.

To address this issue, we first analyze the causal relationship between variables in tasks with a structural causal graph according to information theory [3]. We decouple the randomness in the input variable into the task label and nuisance, both depending on the domain. We then extract domain-invariant latent variables and domain-specific latent variables from the input. The final latent variable combines both domain-invariant and domain-specific latent variables. We perform such decomposition because both domain-agnostic information in the source domain (e.g., general driving policy) and domain-specific information in the target domain (e.g., color distribution in scenes) can contribute to the target domain performance, if the target domain is known. We then design an optimization goal that considers sufficiency, invariance, and implicit minimality, which provides high-level guidance for the framework design.

Based on the analysis, we propose a novel framework for domain-agnostic learning in the steering task, i.e., improve the target domain performance with additional source domain data. To extract the domain-specific features and domain-agnostic features, we use one adapter for each domain and a shared domain-agnostic feature extractor for all domains. The sufficiency and invariance loss objective ensures that the shared feature extractor can extract as many domain-agnostic features as possible and that the remaining information can be captured as domain-specific features by adapters. To further improve the transferability, our method also has two additional add-ons: (1) a training curriculum that gradually increases the ratio of target domain data in each epoch for better knowledge transfer from the source domain to the target domain, and (2) tuned input data, which involves a style-transferred branch in the architecture to help extract domain-specific information. See details in Sec. IV.

Overall, the main contributions of this work include:

- **Information theoretic analysis for transfer learning tasks**, i.e., improve the target domain performance with additional source domain data, providing theory support to our framework design.
- **A novel framework for domain-agnostic learning in end-to-end steering**, by utilizing both *domain-agnostic information across domains* and *domain-specific information in the target domain*, with additional add-ons like improved training paradigm and the style-transferred branch to further enhance the performance.

II. Related Work

A. Transfer Learning

Transfer learning aims to transfer the knowledge learned in a source domain to a target domain. It has been studied

Authors are with Department of Computer Science, University of Maryland at College Park, USA. *{yushen, lyzheng, tianyi, lin}@umd.edu*

in different settings. Task adaptation is one of them when target domain data labels are available. LWF [31] is able to learn in the target domain while keeping the memory of the source domain without storing source domain data. DELTA [30] proposes a novel regularized transfer learning framework, preserving the outer layer outputs of the target network. BSS [8] presents a novel regularization approach to penalizing smaller singular values so that untransferable spectral components are suppressed. StochNorm [26] proposes a two-branch design with one branch normalized by mini-batch statistics and the other branch normalized by moving statistics. Co-Tuning [43] is a two-step framework that can learn the relationship between source categories and target categories, and use source and target labels to collaboratively supervise the fine-tuning process. Bi-Tuning [46] presents a general learning framework to fine-tune both supervised and unsupervised pre-trained representations to downstream tasks. The disentanglement of domain-specific information was studied by applying independent Batch Normalization (BN) layers to different domains in AdvProp BN [41] and StochNorm [26]. However, the domain-specific information might not be fully captured by BN layers with a few parameters. There are also domain adaptation methods like DANN [13], ADDA [40], BSP [9], FADA [32], that can utilize target domain data even without ground-truth labels.

Compared to previous works, our method is among the first that considers all three perspectives, i.e., training data, training paradigm, and network architecture, while most others consider one or two perspectives.

B. Virtual and Real Data

While collecting real-world data can be expensive and challenging, the virtual world enables the economical production of a large amount of data. In order to study how virtual images influence learning-based tasks, researchers have adopted various style-transfer techniques. For example, Movshovitz-Attias et al. [33] explore the effect of state-of-the-art rendering techniques on the viewpoint estimation task of objects. Another style-transfer technique, Generative Adversarial Networks (GANs), has been used for domain transfer between different types of images [24]. For example, CycleGAN [47] and CyCADA [18] have been successfully applied to style-transfer unpaired images in training data. Nowadays the diffusion modal also shows potential to achieve successful style transfer [45], [16]. Finally, the blending of a virtual world and the real world has shown potential for learning-based driving tasks. Li et al. [29] proposed an augmented autonomous driving simulation (AADS), which introduces simulated traffic flows into real-world environments. The training environment is obtained by scanning the real world with lidar and cameras, while simulated traffic flows, including vehicles and pedestrians, are mapped onto the scanned environment. This method captures the benefits of a fully controlled virtual environment while retaining realism.

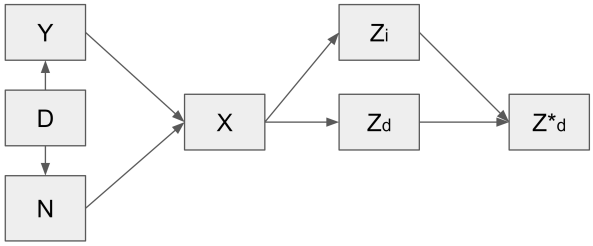


Fig. 1: A structural causal graph that shows causal relationships between variables in our autonomous-steering model. We decouple the randomness in the input variable X into the task label Y and nuisance N, both depending on the domain D. We then extract domain-invariant latent variable Z_i and domain-specific latent variable Z_d from input X. The final latent variable Z_d^* combines Z_i and Z_d.

III. Information Theoretic Analysis

Problem Description. A major challenge for autonomous driving is the variety of driving scenarios: it is infeasible to train a model for all possible scenarios. In a new scenario as the target domain, an emerging challenge is to train a reliable model using data collected in known scenarios (source domain) and a few (insufficient) target-domain data, which can outperform the model trained using the target-domain data only.

Information-Theoretic Analysis. Learning features transferable from a source domain to improve the target-domain performance is a critical challenge in practice. The transfer learning performance, from an information-theoretic perspective, can be improved by leveraging the causal relations between variables. While prior work [28] focuses on finding invariant features for better generalization ability, our method utilizes both domain-agnostic (invariant) features and domain-specific features to better serve our transfer learning goal.

In Fig. 1, we provide a structural causal graph that describes the causal relationships between variables in our problem. In particular, follow [42], an input image is assumed to be generated from a label Y (e.g., for the steering angle) and a nuisance variable N independent of the label, both of which depend on the domain D of the image. Inspired by [28], we extract a domain-agnostic latent variable Z_i and a domain-specific latent variable Z_d from input X. Combining Z_i and Z_d produces a compressed latent variable Z_d^*.

According to the information bottleneck [39] and the causal graph in Fig. 1, an information-theoretic objective for representation learning of Z_d^* can be written as

$$\max_{\theta_{g_i},\theta_{g_d},\theta_{f_{d^*}}} I(Z_d^*, Y) - \lambda I(Y, D|Z_d^*) \tag{1}$$

where g_i is the feature extractor (for domain-agnostic features), g_{d^j} are adapters for each domain, and f_{d^*} is the determinator. $I(X, Y)$, the mutual information between X and Y, is the KL divergence between the joint distribution and the product of the marginals. The first term in Eq. (1) aims to *maximizes the mutual information between the label and combined features*. This helps learn the combined features to

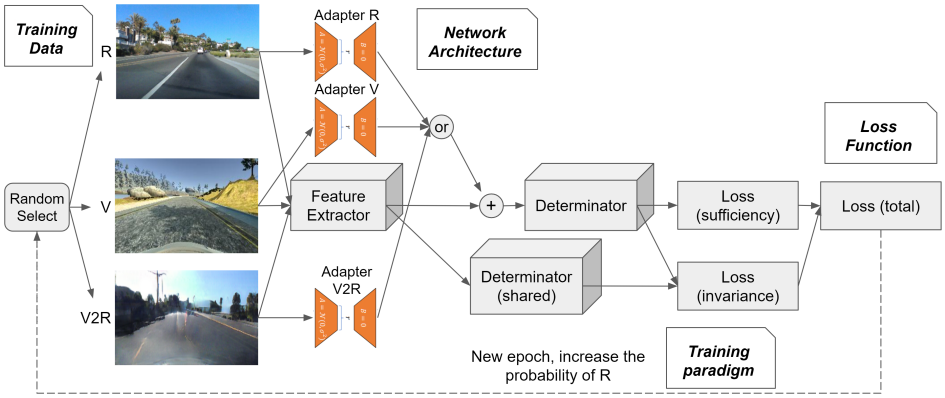


Fig. 2: **Our framework**: In each epoch, the input data is randomly selected from multiple domain data, then fed into a shared feature extractor and a domain-specific adapter for each domain. The combined output feature will then be used to determine the final steering angle (sufficiency loss). An additional invariance loss is used to force the feature extractor to extract as much domain-invariant information as possible. The final loss contains both a sufficiency term and an invariance term. The initial probability of the target domain is small, but will gradually increase as training progresses, enabling better transfer of domain-agnostic knowledge from source to target domain.

capture and preserve key information of the task label. On the other hand, the second term *minimizes the mutual information between the label and the domain given the combined feature*. Theoretically, it is necessary to have one additional minimality term in order to remove the redundancy in the latent feature as much as possible. However, in practice, supervised training of the neural network naturally learns the minimal sufficient representation for the task. So, we removed this item for simplicity.

IV. Our Method

According to the analysis in Sec. III, we design the learning framework as follows.

A. Framework

Model Architecture. Our framework is shown in Fig. 2. According to the information theory analysis in Sec. III, we developed a neural network architecture that can be trained on multi-domain data and capture both domain-agnostic and domain-specific features critical to the steering angle prediction task, in which the former features can capture the front-end perception, back-end steering control, etc., while the latter captures the image style, background data distribution, etc. In the neural network, we use a feature extractor shared across different domains to produce the domain-agnostic features and employ an adapter per domain to produce the domain-specific features.

Training Paradigm. In each epoch, we randomly select training data from different domains, e.g., real, virtual, and style-transferred datasets, according to their pre-defined probabilities[1]. The input image will be fed into the shared feature extractor and a domain-dependent adapter, whose output features will be combined and then used to determine the final steering angle. A sufficiency loss $\mathcal{L}_{d^*}$ is computed by comparing the predicted angle with the ground truth and used as an estimate to $I(Z_d^*, Y)$ in Eq. (1). To estimate $I(Y, D|Z_d^*)$, we train another determinator to predict the steering angle only using the domain-agnostic features, inducing a loss $\mathcal{L}_i$. An invariance loss $\mathcal{L}_{d^*} - \mathcal{L}_i$ is then used as an estimate of $I(Y, D|Z_d^*)$.

[1]Detailed discussion in the next paragraph

Training Curriculum. We start training by learning the source domain first and then gradually increasing the target domain data. This is achieved by initializing a small probability of sampling the target domain R, then gradually increasing it so the main focus of the training will be transferred from the source to the target domain. In Sec. IV-B.2, we show that such *progressive finetuning is better than uniformly sampling training data from both domains*. An explanation for this phenomenon is, *learning two skills together is harder than learning one skill first and then another*.

Loss Functions. Our training objective can be formulated as an instantiation of Eq. (1), whose two mutual information terms are estimated by the two aforementioned losses, i.e., the sufficiency loss $\mathcal{L}_{d^*}$ and the invariance loss $\mathcal{L}_{d^*} - \mathcal{L}_i$. The sufficiency loss aims to minimize the error of using the combined representation Z_d^* to predict Y. Hence, it tries to maximize $I(Z_d^*, Y)$. The invariance loss, on the other hand, aims to minimize the gap between the error $\mathcal{L}_{d^*}$ of predicting Y using Z_d^* and the error $\mathcal{L}_i$ of predicting Y using the domain-agnostic representation Z_i only. Hence, it enforces Z_i to capture all the domain-agnostic information regarding Y and remains to Z_d the domain-specific information of Y only. Thereby, Z_d^* contains all the information D carries regarding Y and $I(Y, D|Z_d^*)$ can be minimized. This equivalence was rigorously proved in IIB [28]. Therefore, the information-theoretical objective Eq. (1) reduces to

$$\min_{g_i, g_d, f_{d^*}} \max_{f_a} \mathcal{L}_{d^*}(g_i, g_d, f_{d^*}) + \lambda(\mathcal{L}_{d^*}(g_i, g_d, f_{d^*}) - \mathcal{L}_i(g_i, f_a)), \tag{2}$$

where g_i and g_d are domain-agnostic and domain-specific features, respectively. And f_{d^*} and f_a respectively denote the determinator using the combined features and the denominator

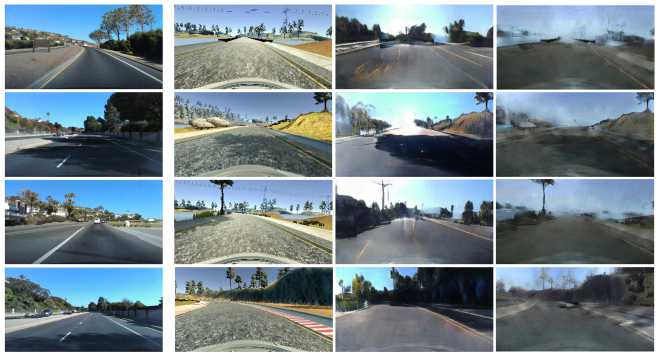
using the domain-agnostic features g_i only. Specifically,

$$\mathcal{L}_{d^*} = \mathbf{E}_{x,y}[L(y, f_{d^*}(g_i(x), g_d(x)))] \quad (3)$$

$$\mathcal{L}_i = \mathbf{E}_{x,y}[L(y, f_i(g_i(x)))] \quad (4)$$

In our experiments, we use L2 (MSE) loss for all losses.

Training Data Curation. We add another style-transferred domain (i.e., the V2R branch in Fig. 2) to the framework to better separate the (domain-specific) style and (domain-agnostic) content of the images. Compared to the original real content + real style images and virtual content + virtual style images, style-transferred images refer to real content + virtual style or virtual content + real style. Inspired by recent works [35], [36], style-transferred images provide more "hints" as prior information disentangling the content and style to help the model learn better. The style transferred data can be generated by CycleGAN [47], a generative model that can exchange the image style of two sets of unpaired images by using forward and backward supervision. We provide examples in Fig. 3.



(a) Real (R) (b) Virtual (V) (c) CycleGAN (T_C) (d) MUNIT (T_M)

Fig. 3: Sample images of various datasets. (a) the Nvidia dataset [7] (real dataset, denoted by R). (b) the Udacity dataset [1] (virtual dataset, denoted by V). (c) style-transferred images from virtual to real using CycleGAN [47] (denoted by T_C). (d) style-transferred images from virtual to real using MUNIT [21] (denoted by T_M). All datasets to be released.

B. Experiments

*1) **Setups**:* All experiments are conducted using one Intel(R) Xeon(TM) W-2123 CPU, two Nvidia GTX 1080 GPUs, and 32G RAM. We use the Adam optimizer [25] with learning rate 0.0001 and batch size 128 for training. The maximum number of epochs is 1,000.

Datasets. We use the SullyChen dataset [7] as our real dataset, which contains approximately 63,000 images at the resolution 455 $\times$ 256. We use the data from the Udacity Self-Driving Car Simulation [1] as our virtual dataset, which includes about 10,615 images at the resolution 320 $\times$ 160 and is collected on a simulated suburban driving track. We also use Audi [15], and Honda [34] as real datasets, and CARLA simulator [12] as a virtual environment for experiments.

Evaluation Metric. Following [37], we use mean accuracy (MA) to evaluate our regression task, since it can represent the overall performance under different thresholds. We first define the accuracy with respect to a particular threshold τ as $acc_\tau = count(|v_{predicted} - v_{actual}| < \tau)/n$, where n denotes the number of test cases; $v_{predicted}$ and v_{actual} indicate the predicted and ground-truth value, respectively. Then, MA is computed as $\sum_\tau acc_{\tau \in \mathcal{T}}/|\mathcal{T}|$, where $\mathcal{T} = \{1.5, 3.0, 7.5, 15, 30\}$ contains empirically selected thresholds of steering angles. We also use other metrics like Mean Square Error (MSE).

Backbone. We choose the model by Bojarski et al. [4] as the default backbone. We also use other backbones in our experiments, e.g., ResNet, LSTM, etc.

Notation and Training Strategies. In this work, we also explore different ways to combine datasets for learning, in order to determine their potential to improve learning performance. We define the following notation for any two datasets A and B.

- $\mathrm{train}(A + B)$: simply combine datasets A and B and use the combined dataset for training;
- $\mathrm{train}(A) \rightarrow \mathrm{train}(B)$: use A to pretrain a model, then use B to retrain the model; and
- $\mathrm{train}(A) \rightarrow \mathrm{ptrain}(B)$: use A to pretrain a model, then use B to retrain the model by only updating partial weights of the model [5].

The output of the above-mentioned three training methods is a learned model. We use $\mathrm{MA}_A(M)$ to denote the MA score of testing a learned model M using the test set extracted from dataset A. For all datasets, we split them into the training set and test set using the ratio 10:1.

*2) **Analysis Experiments**:* We analyze how training data (image style, data amount), network architecture (Batch Norm layers, Adapters), and training paradigm (finetuning, partially finetuning, reinitialization) influence the transfer learning.

Does Image Style Transfer Reduce Domain Gap? To reduce the gap between two domains, the first intuition is to improve the visual similarity of the training data. In this regard, we study how image style can influence the end-to-end steering task.

Some existing works can change the style of an image set to the style of another image set. We use two style-transfer algorithms in this work, a straight-forward color remapping (non-learning-based method) and CycleGAN [47] (learning-based method). We perform cross comparison on six domains, R (real dataset), V (virtual dataset), RV_{CGAN} (real to virtual with CycleGAN), VR_{CGAN} (virtual to real with CycleGAN), RV_{CR} (real to virtual with color remapping), and VR_{CR} (virtual to real with color remapping), which is a combination of real/virtual content + real/virtual style (+ learning/non-learning-based method). We train models on each of them separately, then test on them separately. Table I shows the cross comparison results. We observe:

1) **Image content is more important than image style**. When testing on real-content datasets (R, RV_{CGAN}, RV_{CR}), the models trained on real-content datasets perform better than the models trained on virtual-content datasets, no matter they are real or virtual style (the bolden numbers are greater than the unbolden numbers).

TABLE I: Mean Accuracy cross comparison. RV stands for transferring real dataset to virtual style, VR for transferring virtual dataset to real style. $CGAN$ for the Cycle-GAN method, and CR for the color remapping method.

	Train					
Test	R	V	RV_{CGAN}	VR_{CGAN}	RV_{CR}	VR_{CR}
R	**88.36%**	*31.16%*	**48.83%**	26.87%	**70.17%**	30.08%
RV_{CGAN}	**51.42%**	34.22%	**80.08%**	29.34%	**53.18%**	*38.86%*
RV_{CR}	**60.89%**	35.86%	**48.18%**	27.79%	**85.50%**	*37.41%*

TABLE II: Mean Accuracy comparison with different training paradigms. From (a) we can verify the existence of a domain gap between the virtual, style-transferred and real datasets. From (b,c,d,e,f), we find that (e) **"finetuning with reinitialization" outperforms other training paradigms**.

	Model (M)	$\mathrm{MA}_R(M)$
(a) Single dataset	train(R)	88.36%
	train($R1$)	32.02%
	train(V)	31.16%
	train(T_C)	26.87%
	train(T_M)	25.56%
(b) Simply combine	train($R1$ + R)	82.32%
	train(V + R)	75.74%
	train(T_C + R)	75.44%
	train(T_M + R)	76.85%
(c) Finetuning	train($R1$) $\rightarrow$ train(R)	81.93%
	train(V) $\rightarrow$ train(R)	83.54%
	train(T_C) $\rightarrow$ train(R)	82.70%
	train(T_M) $\rightarrow$ train(R)	79.04%
(d) Partially finetuning	train($R1$) $\rightarrow$ ptrain(R)	70.86%
	train(V) $\rightarrow$ ptrain(R)	73.66%
	train(T_C) $\rightarrow$ ptrain(R)	77.17%
	train(T_M) $\rightarrow$ ptrain(R)	72.97%
(e) Finetuning with reinitialization	train($R1$) $\rightarrow$ train(R)	**88.71%**
	train(V) $\rightarrow$ train(R)	**87.50%**
	train(T_C) $\rightarrow$ train(R)	**83.12%**
	train(T_M) $\rightarrow$ train(R)	**80.26%**
(f) Partially finetuning with reinitialization	train($R1$) $\rightarrow$ ptrain(R)	76.94%
	train(V) $\rightarrow$ ptrain(R)	75.08%
	train(T_C) $\rightarrow$ ptrain(R)	77.78%
	train(T_M) $\rightarrow$ ptrain(R)	74.28%

2) *With the same content during training and test (the boldened numbers), using same style is better than using different styles* (the diagonal of the boldened numbers are greater than other numbers).
3) With different content during training and test (the unboldened numbers), same style does not necessarily perform better (when testing on R, the model trained on V performs best but they are not in the same style, similar for testing on RV_{CGAN} and RV_{CR}).

We also evaluate the domain gap using Fréchet Inception Distance (FID) [17] to minimize the need for training models in new domains. However, we found that **FID is not necessarily an effective metric for evaluating the domain gap in steering task**.

Training Paradigm. The most common technique in transfer learning to cross the domain gap is modifying the training paradigm, i.e., changing the way of training without modifying the network architecture. Some popular methods [22] include,

- *Finetuning*. Retrain a model on the target domain which is pretrained on the source domain.
- *Partially finetuning*. Finetuning a model with fixed weights of specific layers, e.g., CNN layers.
- *Finetuning with reinitialization*. Reinitialize specific layers before retraining. We use header reinitialization.

TABLE III: Mean Accuracy (MA) comparison with different network architectures. **adapter achieves best MA**.

	M	$\mathrm{MA}_R(M)$
(a) Finetuning + reinit header + BN	train(V) $\rightarrow$ train(R)	80.53%
	train($R1$) $\rightarrow$ train(R)	80.77%
(b) AdvProp BN	train(R, V)	71.22%
	train($R, R1$)	75.83%
(c) Finetuning + reinit header + adapter	train(V) $\rightarrow$ train(R)	**81.32%**
	train($R1$) $\rightarrow$ train(R)	**82.71%**

TABLE IV: Accuracy comparison with domain-adaptation & task-adaptation methods. Ours outperforms others with highest accuracy (mAcc) & lowest mean square error (MSE).

		$\mathrm{MA}_R(M)$ (%) on different angle threshold τ (degree)						
	Method	$\tau = 1.5$	$\tau = 3.0$	$\tau = 7.5$	$\tau = 15$	$\tau = 30$	**mAcc**	**MSE**
	Baseline	59.5%	82.1%	93.9%	96.3%	98.6%	86.04%	0.96
(a) Domain Adaptation	DANN [13]	28.9%	52.5%	79.3%	92.2%	97.3%	70.04%	0.58
	ADDA [40]	33.6%	54.3%	84.4%	93.2%	97.5%	72.6%	0.43
	BSP [9]	38.9%	60.4%	87.5%	95.1%	98.4%	76.06%	0.32
(b) Task Adaptation	DELTA [30]	61.9%	80.9%	93.9%	97.7%	99.2%	86.72%	0.16
	BSS [8]	67.0%	83.4%	93.8%	97.5%	98.8%	88.1%	0.21
	StochNorm [26]	53.7%	78.5%	92.8%	97.3%	99.2%	84.3%	0.18
	Ours	**70.5%**	**84.3%**	**93.8%**	**97.9%**	**99.4%**	**89.2%**	**0.15**

Table II shows the Mean Accuracy (MA) comparison with different training paradigms and source domains. From (a) we can verify the existence of a domain gap between the virtual, style-transferred and real datasets. (b) is a basic paradigm that trains a model with source and target domain data simply combined. From (b,c,d,e,f), we find that **(e) "finetuning with reinitialization" outperforms other training paradigms in the list**, no matter using real ($R1$, Audi dataset [15]), virtual (V), or style transferred (T_C, T_M) datasets as source domain.

Network Architecture. Except for the training paradigm related methods, there are methods that achieve transfer learning by modifying the network architecture, e.g., batch norm layers, adapters, etc. Usually they also need to have specific training paradigms, e.g., adding batch norm layers, retraining the model with fixed CNN layers but trainable batch norm weights. Here we mainly investigate two additive network components,

- **Batch Norm (BN)** layer [23]. Batch normalization is a method used to make training of artificial neural networks faster and more stable through normalization of the layers' inputs by re-centering and re-scaling. Different domains have different feature distributions, which can be aligned by adding batch norm layers in the network.
- **Adapter** [19]. Adapters are new modules added between layers of a pre-trained network. They add only a few trainable parameters per task, and new tasks can be added without revisiting previous ones. The parameters of the original network remain fixed, yielding a high degree of parameter sharing.

In Table III, we compare normal BN [23], AdvProp BN [41], and adapter [20] under the best training paradigms in previous experiments. **adapter achieves the best performance in the list.**

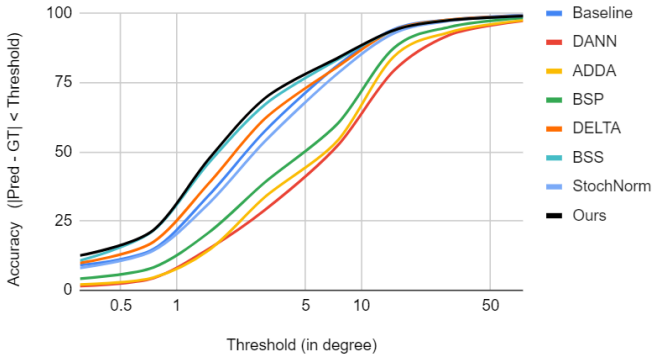


Fig. 4: **Threshold-Accuracy Curve**. Our method (in black) achieves the best (highest) performance – above all other methods.

C. *Final Results*

Comparison with other task adaptation methods. We compare our method with other SOTA task adaptation methods, i.e., DELTA [30], BSS [8], StochNorm [26]. All of them use both source and target data and labels. Our method outperforms others by up to **2.29%**. See Table IV and Fig. 4.

Comparison with other domain adaptation methods. We compare our method with other classic domain adaptation methods, i.e., DANN [13], ADDA [40], BSP [9]. Since those domain adaptation methods do not use target domain labels, our method can achieve up to **15.45%** improvement. See Table IV.

Comparison on other metrics. We use dynamically interpolated thresholds to test the accuracy, $\tau = 3(y_{gt}/30)^{1/\alpha}$ (degree). When $\alpha = 1$, the threshold is 10% of the ground truth steering angle, while $\alpha = \infty$, the threshold is a constant value 3 (degree). Our method outperforms other techniques under all metrics. Please see Table V.

Comparison on other datasets. We compare our method with BSS [8] on SullyChen, Audi, and Honda dataset. Our method achieves better performance across all datasets. See Table VI.

Comparison on other backbones. We compare our method with BSS [8] on PilotNet, ResNet, and LSTM. Our method achieves better performance across all backbones. See Table VII.

Ablation study. To analyze how each component (ADP is the adapter, STB stands for style transferred branch, DP stands for dynamic probability for each domain) takes effect, we conduct an ablation study on each. Results show that removing any component will lead to a performance degradation. This result suggests that each one does indeed contribute to the overall performance. See Table VIII.

V. Conclusion

In autonomous driving, applying knowledge learned in a known domain to an unknown domain is still one of the key challenges due to the variety of driving scenarios in the real world (and virtual world). Domain-agnostic learning, or transfer learning, makes it possible to achieve knowledge transfer between different domains. In this work, we investigate how training data (in terms of image style and data amount), network architecture (Batch-Norm layers, Adapters), and training paradigm (finetuning, partially finetuning, reinitialization) affect the domain-agnostic learning in the end-to-end steering task. Based on the analysis, we propose a novel domain-agnostic learning framework with (1) domain-specific adapters and shared modules to learn domain-specific and task-agnostic information that are both useful to the steering task; (2) style-transferred data domain to help disentangle domain-specific information; (3) gradually increased ratio of target domain data in each epoch for better knowledge transfer from the source to the target domain.

Limitations and Future Works: (a) The style-transferred data is generated by a style-transfer network, which can be merged into our framework in the future. (b) The distribution of the steering values is highly non-uniform and likely domain-dependent. In our future studies, we will explore whether there exists a better training paradigm that can take advantage of this specific prior.

TABLE V: Mean Accuracy comparison with domain adaptation and task adaptation methods on different metrics. Our method outperforms others under nearly all angle thresholds.

		$\mathrm{MA}_R(M)$ (%) on different angle threshold $\tau = 3(y_{gt}/30)^{1/\alpha}$ (degree)					
	Method	$\alpha = 1(\tau = 0.1y_{gt})$	$\alpha = 2$	$\alpha = 5$	$\alpha = 10$	$\alpha = \infty(\tau = 3)$	**mAcc**
	Baseline	31.4%	50.0%	69.7%	74.8%	79.7%	61.13%
(a) Domain Adaptation	DANN [13]	11.3%	22.9%	37.3%	44.9%	51.4%	33.55%
	ADDA [40]	11.9%	20.7%	38.5%	46.1%	52.3%	33.91%
	BSP [9]	17.0%	31.8%	44.9%	52.0%	58.6%	40.86%
(b) Task Adaptation	DELTA [30]	33.6%	56.2%	72.3%	76.6%	79.3%	63.59%
	BSS [8]	36.3%	**63.1%**	74.6%	78.3%	81.6%	66.80%
	StochNorm [26]	29.5%	43.8%	63.7%	71.7%	76.8%	57.07%
	Ours	**36.7%**	60.4%	**77.5%**	**80.5%**	**82.6%**	**67.54%**

TABLE VI: Comparison on different datasets.

		$\mathrm{MA}_R(M)$ (%) on different angle threshold τ (degree)						
	Method	$\tau = 1.5$	$\tau = 3.0$	$\tau = 7.5$	$\tau = 15$	$\tau = 30$	**mAcc**	**MSE**
(a) SullyChen	BSS [8]	67.0%	83.4%	93.8%	97.5%	98.8%	88.1%	0.21
	Ours	70.5%	84.3%	93.8%	97.9%	99.4%	**89.2%**	**0.15**
(b) Audi	BSS [8]	59.5%	72.3%	81.6%	86.9%	89.7%	78%	0.88
	ours	62.5%	75.3%	84.8%	89.1%	92.4%	**80.8%**	**0.65**
(c) Honda	BSS [8]	55.4%	70.9%	77.8%	83.7%	86.5%	74.86%	1.16
	ours	57.6%	73.9%	80.2%	85.7%	89.1%	**77.3%**	**0.91**

TABLE VII: Comparison on different backbones.

		$\mathrm{MA}_R(M)$ (%) on different angle threshold τ (degree)						
	Method	$\tau = 1.5$	$\tau = 3.0$	$\tau = 7.5$	$\tau = 15$	$\tau = 30$	**mAcc**	**MSE**
(a) PilotNet	BSS [8]	67.0%	83.4%	93.8%	97.5%	98.8%	88.1%	0.21
	Ours	70.5%	84.3%	93.8%	97.9%	99.4%	**89.2%**	**0.15**
(b) ResNet	BSS [8]	71.8%	84.9%	93.8%	97.4%	98.3%	89.24%	0.15
	ours	72.3%	85.6%	94.5%	98.2%	99.5%	**90.02%**	**0.13**
(c) LSTM	BSS [8]	73.1%	85.4%	94%	97.5%	98.9%	89.78%	0.14
	ours	74.5%	86.9%	95.1%	98.6%	99.7%	**90.96%**	**0.12**

TABLE VIII: Ablation study. ADP for adapter, STB for style transferred branch, DP for dynamic probability in each domain, and IBL for information bottleneck loss.

	$\mathrm{MA}_R(M)$ (%) on different angle threshold τ (degree)						
Method	$\tau = 1.5$	$\tau = 3.0$	$\tau = 7.5$	$\tau = 15$	$\tau = 30$	**mAcc**	**MSE**
Baseline	59.5%	82.1%	93.9%	96.3%	98.6%	86.04%	1.96
Ours w/o ADP	58.4%	80.3%	93.4%	97.7%	98.6%	85.68%	0.19
Ours w/o STB	68.0%	81.6%	94.1%	97.7%	99.0%	88.08%	0.16
Ours w/o DP	65.6%	82.2%	93.4%	97.3%	98.8%	87.46%	0.18
Ours w/o IBL	69.3%	84.0%	**93.9%**	97.5%	99.0%	88.74%	0.18
Ours	**70.5%**	**84.3%**	93.8%	**97.9%**	**99.4%**	**89.2%**	**0.15**

REFERENCES

[1] Udacity's Self-Driving Car Simulator, https://github.com/udacity/self-driving-car-sim, 2017.

[2] NVIDIA DRIVE™ Constellation AV Simulator, https://www.nvidia.com/en-us/self-driving-cars/drive-constellation/, 2019.

[3] Robert B Ash. *Information theory*. Courier Corporation, 2012.

[4] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

[5] Jason Brownlee. *Better Deep Learning: Train Faster, Reduce Overfitting, and Make Better Predictions*. Machine Learning Mastery, 2018.

[6] Qianwen Chao, Huikun Bi, Weizi Li, Tianlu Mao, Zhaoqi Wang, Ming C. Lin, and Zhigang Deng. A survey on visual traffic simulation: models, evaluations, and applications in autonomous driving. *Computer Graphics Forum*, 39(1):287–308, 2019.

[7] Sully Chen. A collection of labeled car driving datasets, https://github.com/sullychen/driving-datasets, 2018.

[8] Xinyang Chen, Sinan Wang, Bo Fu, Mingsheng Long, and Jianmin Wang. Catastrophic forgetting meets negative transfer: Batch spectral shrinkage for safe transfer learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[9] Xinyang Chen, Sinan Wang, Mingsheng Long, and Jianmin Wang. Transferability vs. discriminability: Batch spectral penalization for adversarial domain adaptation. In *International conference on machine learning*, pages 1081–1090. PMLR, 2019.

[10] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[11] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

[12] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

[13] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR, 2015.

[14] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[15] Jakob Geyer, Yohannes Kassahun, Mentar Mahmudi, Xavier Ricou, Rupesh Durgesh, Andrew S. Chung, Lorenz Hauswald, Viet Hoang Pham, Maximilian Mühlegg, Sebastian Dorn, Tiffany Fernandez, Martin Jänicke, Sudesh Mirashi, Chiragkumar Savani, Martin Sturm, Oleksandr Vorobiov, Martin Oelker, Sebastian Garreis, and Peter Schuberth. A2D2: Audi Autonomous Driving Dataset. 2020.

[16] Mark Hamazaspyan and Shant Navasardyan. Diffusion-enhanced patchmatch: A framework for arbitrary style transfer with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 797–805, 2023.

[17] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[18] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. *arXiv preprint arXiv:1711.03213*, 2017.

[19] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019.

[20] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[21] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 172–189, 2018.

[22] Mohammadreza Iman, Hamid Reza Arabnia, and Khaled Rasheed. A review of deep transfer learning and recent advancements. *Technologies*, 11(2):40, 2023.

[23] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

[24] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[25] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[26] Zhi Kou, Kaichao You, Mingsheng Long, and Jianmin Wang. Stochastic normalization. *Advances in Neural Information Processing Systems*, 33:16304–16314, 2020.

[27] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[28] Bo Li, Yifei Shen, Yezhen Wang, Wenzhen Zhu, Dongsheng Li, Kurt Keutzer, and Han Zhao. Invariant information bottleneck for domain generalization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 7399–7407, 2022.

[29] W. Li, C. W. Pan, R. Zhang, J. P. Ren, Y. X. Ma, J. Fang, F. L. Yan, Q. C. Geng, X. Y. Huang, H. J. Gong, W. W. Xu, G. P. Wang, D. Manocha, and R. G. Yang. AADS: Augmented autonomous driving simulation using data-driven algorithms. *Science Robotics*, 4(28), 2019.

[30] Xingjian Li, Haoyi Xiong, Hanchao Wang, Yuxuan Rao, Liping Liu, Zeyu Chen, and Jun Huan. Delta: Deep learning transfer using feature map with attention for convolutional networks. *arXiv preprint arXiv:1901.09229*, 2019.

[31] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

[32] Saeid Motiian, Quinn Jones, Seyed Iranmanesh, and Gianfranco Doretto. Few-shot adversarial domain adaptation. *Advances in neural information processing systems*, 30, 2017.

[33] Yair Movshovitz-Attias, Takeo Kanade, and Yaser Sheikh. How useful is photo-realistic rendering for visual learning? In *European Conference on Computer Vision*, pages 202–217. Springer, 2016.

[34] Vasili Ramanishka, Yi-Ting Chen, Teruhisa Misu, and Kate Saenko. Toward driving scene understanding: A dataset for learning driver behavior and causal reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7699–7707, 2018.

[35] Yu Shen, Xijun Wang, Peng Gao, and Ming C Lin. Auxiliary modality learning with generalized curriculum distillation. 2023.

[36] Yu Shen, Luyu Yang, Xijun Wang, and Ming C Lin. Small-shot multi-modal distillation for vision-based autonomous steering. 2023.

[37] Yu Shen, Laura Zheng, Manli Shu, Weizi Li, Tom Goldstein, and Ming Lin. Gradient-free adversarial training against image corruption for learning-based steering. *Advances in Neural Information Processing Systems*, 34:26250–26263, 2021.

[38] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. *arXiv*, pages arXiv–1912, 2019.

[39] Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

[40] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7167–7176, 2017.

[41] Cihang Xie, Mingxing Tan, Boqing Gong, Jiang Wang, Alan L Yuille, and Quoc V Le. Adversarial examples improve image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 819–828, 2020.

[42] Kaiwen Yang, Yanchao Sun, Jiahao Su, Fengxiang He, Xinmei Tian, Furong Huang, Tianyi Zhou, and Dacheng Tao. Adversarial auto-augment with label preservation: A representation learning principle guided approach. *Advances in Neural Information Processing Systems*, 35:22035–22048, 2022.

[43] Kaichao You, Zhi Kou, Mingsheng Long, and Jianmin Wang. Co-tuning for transfer learning. *Advances in Neural Information Processing Systems*, 33:17236–17246, 2020.

[44] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2018.

[45] Yuxin Zhang, Nisha Huang, Fan Tang, Haibin Huang, Chongyang Ma, Weiming Dong, and Changsheng Xu. Inversion-based style transfer with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10146–10156, 2023.

[46] Jincheng Zhong, Ximei Wang, Zhi Kou, Jianmin Wang, and Mingsheng Long. Bi-tuning of pre-trained representations. *arXiv preprint arXiv:2011.06182*, 2020.

[47] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.