Baltic J. Modern Computing, Vol. 12 (2024), No. 4, pp. 374–386
https://doi.org/10.22364/bjmc.2024.12.4.02

DINO Pre-training for Vision-based End-to-end Autonomous Driving

Shubham JUNEJA[1], Povilas DANIUŠIS[2], Virginijus MARCINKEVIČIUS[1]

[1] Vilnius University Institute of Data Science and Digital Technologies, Akademijos str. 4, Vilnius, LT-08412, Lithuania
[2] Research Institute of Natural and Technological Sciences, Vytautas Magnus University, 53361 Kaunas, Lithuania

shubham.juneja@mif.stud.vu.lt, povilas.daniusis@vdu.lt, virginijus.marcinkevicius@mif.vu.lt

ORCID 0000-0002-7906-5688, ORCID 0000-0001-5977-827X, ORCID 0000-0002-2281-4035

Abstract. In this article, we focus on the pre-training of visual autonomous driving agents in the context of imitation learning. Current methods often rely on a classification-based pre-training, which we hypothesise to be holding back from extending capabilities of implicit image understanding. We propose pre-training the visual encoder of a driving agent using the self-distillation with no labels (DINO) method, which relies on a self-supervised learning paradigm.Our experiments in CARLA environment in accordance with the Leaderboard benchmark reveal that the proposed pre-training is more efficient than classification-based pre-training, and is on par with the recently proposed pre-training based on visual place recognition (VPRPre).

Keywords: Autonomous driving, self-supervised pre-training, DINO

1 Introduction

While autonomous driving of robots and vehicles can be achieved by breaking down the task of driving into individual sub-tasks and assigning a module for each, end-to-end learning takes the holistic approach. End-to-end learning of driving often relies upon imitation learning, i.e., given a corpus of data (for e.g., visual demonstrations), a task is learned with a machine learning method, e.g., neural network. Hence, instead of singularly learning every sub-task or programming it into a module, an entire skill is learned in the form of a policy from data. To absorb the precise behaviour from data (or demonstrations) needed to safely drive in the real world, datasets of vast sizes may be required along with stronger methods that learn from what is available. This is one of the issues that has stalled progress in research on purely vision-based end-to-end models

(Xiao et al., 2023), giving rise to hybrid methods combining end-to-end approaches with modular pipelines.

The major cause of insufficient driving performance in imitation learning-based methods stems from the problem of co-variate shift. Co-variate shift (Ross et al., 2010; Tampuu et al., 2020) is caused by occurrences of situations (or data points) at test time which have not been presented at the time of training, resulting in encountering a shift in data distributions. Specifically in the context of autonomous driving, this can be a mixture of unseen weather conditions, towns, traffic situations, and so on. While the various methods that propose aggregating of new and vital data points (Ross et al., 2010; Zhang and Cho, 2017; Prakash et al., 2020) into the existing data corpus have become an essential practice in imitation learning, there still seems to be a lot of space for the learned methods to adapt better. Besides improving data quality with data aggregation, there are other important lines of research, one of which focuses on model parameter initialisation or pre-training (Zhang et al., 2022; Wu et al., 2023; Juneja et al., 2023).

Efficient pre-training of a learning method (e.g. neural network) potentially implies parameter initialisation, that may be related to the learning task of interest. This has recently been the highlight of advances in language modelling (Minaee et al., 2024), and very commonly used in vision based learning tasks as well (i.e. models pre-trained on ImageNet (Deng et al., 2009) such as ResNet (He et al., 2016)). Majority of the works in autonomous driving rely on a classification-based pre-training (Codevilla et al., 2019; Zhang et al., 2021; Chitta et al., 2022; Xiao et al., 2023; Jia et al., 2023) and only a handful of works investigate the impact of various other pre-training methods (Zhang et al., 2022; Wu et al., 2023; Juneja et al., 2023).

A recent self-supervised method called self-distillation with no labels (DINO) (Caron et al., 2021) has shown an inherent understanding of the semantic information of an image, which implies it's potential usefulness for various computer vision tasks, including autonomous driving.

We hypothesise that pre-training a driving agent's vision encoder with heavy guidance based on labels may be indirectly or directly holding back the agent's driving performance when it comes to generalisation. Such issues could arise due to the presence of strong image-level supervision of supervised methods potentially reducing the concept of learning to a single task. This may also hold true in some very recent self-supervised pre-training methods (Wu et al., 2023; Zhang et al., 2022; Juneja et al., 2023).

We empirically investigate the above hypothesis by applying DINO pre-training (Caron et al., 2021), and comparing it with the standard supervised classification-based pre-training approach and with a recent method, visual place recognition pre-traing for driving agents (VPRPre) (Juneja et al., 2023). The contributions of this paper are as follows:

1. We propose and empirically investigate DINO pre-training for imitation learning-based autonomous driving agents.
2. Following the offline Leaderboard benchmark standard (Zhang et al., 2021; Hu et al., 2022) on the CARLA 0.9.11 simulator, we empirically demonstrate that the vision encoder pre-trained with downstream-task-agnostic DINO exhibits improved driving performance compared to vision encoders pre-trained with supervised learning (ImageNet classification). Furthermore, our experimental findings

reveal that its performance is comparable to the VPRPre encoder (Juneja et al., 2023), which was also trained in the same CARLA environment.

2 Related work

The core concept of using neural networks for autonomous driving in research was initially demonstrated by the ALVINN method (Pomerleau, 1988), and was revisited during the recent connectionist renaissance with the work of (Bojarski et al., 2016).

PilotNet, based on work from (Bojarski et al., 2016), took advantage of a deeper neural network architecture and the capability of available computing power to train it, establishing higher performance. The use of better and well-adapted architectures for driving has thereon been a strong line of research (e.g., (Codevilla et al., 2019; Daniušis et al., 2021; Chitta et al., 2022; Xiao et al., 2023; Yokoyama et al., 2024)). The most notable architecture that has been frequently used and has shown a remarkable improvement in adapting for driving is conditional imitation learning with ResNet (CILRS) (Codevilla et al., 2018, 2019), where each command is given a different multi-layer perceptron branch. Another line of research that has contributed to the progress of autonomous driving in order to reduce the effect of co-variate shift, is on how to aggregate data better into the training data corpus. While the core method of data aggregation (DAgger) (Ross et al., 2010) has brought improvement by simply aggregating corrective data where an agent misbehaves, other DAgger methods (Zhang and Cho, 2017; Prakash et al., 2020) have explored how can that be conducted more efficiently.

Various other aspects of end-to-end autonomous driving have been challenged in order to find ways to enhance its capabilities. A milestone in this research was reached by the method called Roach (Zhang et al., 2021) which questions the quality of demonstrations used for training, be it human-driven or rule-based demonstration. The researchers proposing Roach argue that the current form of demonstration may not be well informed, and hence propose a reinforcement learning agent that drives based on a birds-eye view and generates higher quality demonstrations, resulting in better data quality for a latter agent trained over these demonstrations. The latter agent drives on frontal camera view just as previously mentioned methods, resulting in improved performance. Meanwhile, CIL++ (Xiao et al., 2023) proposes enriching the vision in the imitation learning-based agent rather than demonstrations, with a higher field of view provided by two additional cameras. CIL++ also extends on the original CILRS (Codevilla et al., 2019) method by the use of transformer (Vaswani et al., 2017) architecture to fuse multiple views. Transfuser (Chitta et al., 2022) is another method that also uses a transformers-inspired architecture, and does so to explore multi-modality by extending image-based vision with LiDAR. Multi-modality for driving has also been previously explored with different architectures (Xiao et al., 2022; Juneja et al., 2021). While most methods use a similar architecture for driving, the following method named ThinkTwice (Jia et al., 2023) brings emphasis onto the decoder part of the architecture. This method modifies the decoder to be able to focus on different areas of the input image given the current context. It makes several coarse predictions and gradually refines the offset to each prediction.

All previously mentioned methods have a high dependence on vision as a modality, and in their basic forms, they utilise a vision encoder pre-trained on ImageNet (Deng et al., 2009) in a supervised way. Rather than directly learning weights from the task of driving, utilising a pre-trained vision encoder provides potentially more advanced starting point for learning. However, only a handful of works in the area of end-to-end autonomous driving have experimented with other forms of pre-training than the standard ImageNet classification-based pre-training, hence this line of work remains underexplored. As an example, a recent method pre-trains the vision encoder on the task of visual place recognition at first and then incorporates it into a full-fledged architecture for the task of driving (Juneja et al., 2023).

While VPRPre (Juneja et al., 2023) corresponds to supervised learning, some recent methods take advantage of the self-supervised learning (Zhang et al., 2022; Wu et al., 2023). Policy pre-training via geometric modelling (PP-Geo) (Wu et al., 2023) learns geometric information such as pose, depth and future ego-motion in a self-supervised manner as such information can be made available through a simulator during data collection, while labelled information can have high costs. Another self-supervised method called action conditioned contrastive pre-training (ACO) (Zhang et al., 2022) explores pre-training on contrastive representation learning over YouTube videos as a pre-training method. Being based on self-supervised learning, both PP-Geo and ACO are heavily guided by labels which may keep these methods from learning a wider set of features and focus only on the task at hand during pre-training.

Supervised approaches often require labelled data which can be expensive to scale. In contrast, self-supervised learning alleviates this by learning from alternately available meta-data rather than expensive labels, proving to be highly sample efficient.

We base our research on the recently proposed work, DINO (Caron et al., 2021), which trains over ImageNet using a self-supervised approach without the use of manual annotations. This is conducted with a multi-crop training approach applied onto a contrastive loss, in the presence of a momentum encoder (He et al., 2020). The authors explore and confirm the results on convolutional networks as well as on transformer networks. This method has been able to show that learning features in the proposed non-label guided self-supervised way can inherently enable scene layout and object boundaries understanding without any explicit labels for the same. DINO enables possibilities of exploring in similar directions as PP-Geo, ACO and VPRPre methods, and additionally investigating if supervised ImageNet pre-training may be a practice that can be considered outdated.

3 Method

End-to-end autonomous driving based on imitation learning is achieved by training over multiple sets of demonstrations. As the standard procedure followed by most imitation based techniques, training over a fixed set of demonstrations doesn't suffice, hence after the first training round another set of demonstrations are aggregated (Ross et al., 2010). This is done up-to 5 times and the results of the final iteration is reported. We aim to pre-train the vision encoder using a non-label guided self-supervised (DINO) method over a general task and then to train a visual end-to-end autonomous driving model

relying on the aforementioned DAgger approach. We reveal the details of our method in the following subsections.

3.1 DINO Pre-training

Self-supervised training uses unlabelled data and the artificial supervision signal, provided by the learning algorithm. We further utilise the DINO (Caron et al., 2021) method which performs self-supervised learning over the ImageNet dataset consisting of ≈ 1 million images, resulting in efficient image representations, that are useful for a variety of down-stream tasks.

DINO uses two networks, a student and a teacher architecture, with same number of parameters that use distillation during training. The student network g_{θ_s} with parameters θ_s is trained to match the output of a teacher network g_{θ_t} with parameters θ_t. For an input x, both student and teacher networks infer K-dimensional probability distributions, P_s and P_t respectively. Following that, the probabilities are calculated from the distributions using a softmax function with a modification where the sharpness of the distributions are controlled with a temperature parameter. For the case of the student network, the modified softmax equation can be seen in equation 1, where to calculate the probability P_s temperature parameter τ_s is used to control sharpness.

$$P_s(x)^{(i)} = \frac{\exp\left(g_\theta(x)^{(i)}/\tau_s\right)}{\sum_{k=1}^{K} \exp\left(g_\theta(x)^{(k)}/\tau_s\right)} \tag{1}$$

A similar equation is used for calculating P_t with temperature parameter τ_t. The temperature control parameters are conditioned $\tau_s > 0$, $\tau_t > 0$.

The teacher network is co-trained along with the student network but is frozen during an epoch. Instead, the exponential moving average of the weights is copied from the student network to the teacher network, using the momentum encoder technique (He et al., 2020). The update rule used is

$$\theta_t \leftarrow \lambda\theta_t + (1-\lambda)\theta_s, \tag{2}$$

where λ follows a cosine schedule from 0.996 to 1 during training. With the use of a fixed teacher network within an epoch, the learning takes place by minimising cross-entropy w.r.t. the student network parameters θ_s, as in the following equation,

$$\min_{\theta_s} H(P_t(x), P_s(x)), \tag{3}$$

where $H(a, b) = -a \log b$.

To leverage the self-supervision, DINO uses multi-crop training (Caron et al., 2020). At first, a set of multiple views or crops V of an image are formed, in two settings. First setting creates two views called global views x_1^g and x_2^g, which are crops at resolution of 224×224 that cover more than 50% of the image. The second setting creates several views called local views which are of resolution 96×96 that cover less than 50% of the image. Once the views are created, the global views are passed through the teacher network, and the local views are passed through the student network. Then modified

version of the loss in eq. 3 is used to adapt to a self-supervised setting in the following way:

$$\min_{\theta_s} \sum_{x \in \{x_1^{g_1}, x_2^{g_2}\}} \sum_{x' \in V, x' \neq x} H(P_t(x), P_s(x')) \tag{4}$$

The neural networks g_θ are composed of a backbone f and projection head h. DINO features are represented by the outputs of backbone of student network.

3.2 Driving

For the driving agent we follow the framework set in our previous work (Juneja et al., 2023). The decoder of the architecture is based upon CILRS (Codevilla et al., 2019) as commonly improved in many other works (Zhang et al., 2021; Juneja et al., 2023), where high-level command is given by the navigation system activates the corresponding branch of the decoder. This high-level command may be one of several discrete instructions, for example, follow lane, turn right, etc. To collect the initial demonstrations for the base training data, we use Roach (Zhang et al., 2021), which enables automated data collection with a reinforcement learning agent (Roach agent), that drives from bird's eye perspective. While the agent drives, it collects images from the front camera of the car along with the low-level driving commands executed. Once we have the initial dataset of demonstrations, we train our agent with a pre-trained encoder integrated into it. The trained agent is then let to drive in the simulated environment with training settings, and while this trained agent drives it is supervised by the Roach agent. Hence, at instances where the trained agent makes mistakes i.e. disagrees with the supervising Roach agent, it is corrected by the Roach agent and these instances of the demonstrations are saved for the next iteration of DAgger. This is followed by training over the aggregated set of corrected demonstrations and the initial dataset together. This process of collecting aggregated data and re-training is performed for a total of 5 times, as per the benchmark standard (Zhang et al., 2021; Hu et al., 2022).

Similar to recent works (Codevilla et al., 2019; Zhang et al., 2021; Juneja et al., 2023), our agent's architecture consists of a pre-trained vision encoder that encodes the front-view RGB image, along with a measurements encoder that encodes the current speed of the vehicle and the high-level command from the planner that is one hot encoded. Both of the encodings are then concatenated and downsized using a join module, formed by fully connected layers. The output of the join module is ran through the action branches, where each branch is a module of fully connected layers and is responsible for each high-level command. Based on the high-level command, the corresponding branch's prediction is chosen. That prediction represents the low-level driving command. For training, non-corresponding branches are zeroed out.

To mathematically represent our agent, let $X \in R^{224\times224\times3}$ be the front-view image from the vehicle. Thereon, f_E being the image encoder with parameters θ pre-trained using the DINO method, u being the vector holding measurements (current speed and high-level command), f_M denoting the measurements encoder network with parameters ξ, f_J denoting the join module with parameters ϕ that concatenates the image and measurements encodings and passes through for downsizing, f_A being the action branches module with parameters ψ which calculate a low-level command for each

high-level command, gives the representation of the network as

$$f_A(f_J(f_E(X|\theta), f_M(u|\xi)|\phi)|\psi). \tag{5}$$

As the network gives out low-level commands for all possible high-level commands, to select as per the high-level command of interest, let c_i be the one-hot encoded command which is indexed with i that zeroes out the non-command branches, we reformulate the network in statement 5 into an equation as

$$\widehat{\mathbf{a}}(X, u|\theta, \xi, \phi, \psi) := \sum_{i=0}^{n} c_i \, b_i(X, u|\theta, \xi, \phi, \psi),. \tag{6}$$

where b_i corresponds to the output of i-th branch.

For simple comparability with a baseline method, we adapt to the standard loss function used for the task of end-to-end driving (Codevilla et al., 2019; Zhang et al., 2021), which is the sum of action loss $\mathcal{L}_A$ and a speed prediction regularisation $\mathcal{L}_S$,

$$\mathcal{L}_{Agent}(\theta, \xi, \phi, \psi) = \mathcal{L}_A(\theta, \xi, \phi, \psi) + \lambda_S \cdot \mathcal{L}_S. \tag{7}$$

Action loss $\mathcal{L}_A$ is given by,

$$\mathcal{L}_A = \left\| \widehat{\mathbf{a}}(X, u|\theta, \xi, \phi, \psi) - \mathbf{a} \right\|_1, \tag{8}$$

which calculates the L1 loss between the expert action $\hat{\mathbf{a}}$ and learned method's predicted action $\mathbf{a}$. The speed prediction regularisation $\mathcal{L}_S$ is given by,

$$\mathcal{L}_S = |\hat{s} - s|. \tag{9}$$

that calculates the difference between measured speed $\hat{s}$ and predicted speed s, and is regulated by a scalar value λ_s mentioned in eq. 7.

4 Experiments

4.1 Implementation details

To quantify the impact of using DINO pre-training, by following the framework of our previous work (Juneja et al., 2023) we also implement a baseline method. The baseline method follows the standard setting as in most works that use a convolutional neural network, i.e. it uses a ResNet50 encoder. This encoder is pre-trained with supervised learning over the ImageNet dataset.

The DINO pre-trained method and the baseline contain exactly the same number of parameters, only differ in the values (or weights) they hold. While rest of the network is randomly initialised to be trained from scratch. We initialise the measurement encoder f_M with 2 fully connected layers and set the output dimension to 128 at each layer. The join module f_J is initialised with 3 fully connected layers and has the output dimensions set to 512, 512 and 256 respectively. These layers are followed by the action branches f_A which consist of 3 fully connected layers with the output dimensions 256, 256 and

2, respectively. All modules consisting of fully connected layers use a rectified linear unit activation, except the last layers in action branches.

Both methods are trained on the same initial dataset collected with the Roach agent. We then collect additional data on every trained model, following the formal DAgger procedure. We iterate with DAgger for 5 times in total and collect 5 datasets in addition to the common initial dataset, for each method. Both methods are trained for 20 epochs, with initial learning rate of $1e-4$ and later stepped down to $1/10^{th}$ of initial value 15^{th} epoch onwards. The training is carried out out on a single Nvidia RTX 3090 GPU with 24GB of memory fitting a batch size of 256. For both methods, we uniformly train on smaller resolution images compared to those used in most approaches. Specifically, we employ a resolution of 224×224, as opposed to the commonly used 256×900 resolution, following the settings of our previous work (Juneja et al., 2023) and due to limitations in computational and time resources. As the dataset size scales over DAgger iterations, the training time for each iteration scales from 10 to 25 hours. Additionally, it requires at least 30 hours to evaluate every agent that has been trained, making to total time spent over all experiments over 2 months.

4.2 Benchmark settings

We benchmark both the methods on the offline version of the Leaderboard benchmark (Zhang et al., 2021; Hu et al., 2022). The Leaderboard benchmark operates in the CARLA 0.9.11 simulator (Dosovitskiy et al., 2017), which simulates city and highway like environments for autonomous driving scenarios. The simulator lets the traffic (road traffic and pedestrian density) and weather conditions be controlled which brings a strong set of possible combinations to test agents on. The Leaderboard benchmark defines settings for training and testing, where the agent is to be trained over data from 4 different town environments with a fixed set of weather conditions, and then tested in 2 unseen environments along with unseen weather conditions.

Table 1. Weather conditions used for training, evaluation and testing.

Training weathers	Evaluation weathers	Testing weathers
Wet noon	Wet noon	Wet sunset
Clear sunset	Clear sunset	Soft rain sunset
Clear noon		
Hard rain noon		

We state the settings in Tables 1 and 2. For the evaluation task Following the benchmark standard, the agent is run from a given starting point to a given ending point, in a combination of settings of town and weather. We run the benchmark with the busy traffic setting, as done in other recent works (Zhang et al., 2021; Hu et al., 2022; Juneja et al., 2023).

Table 2. Towns used for training, evaluation and testing.

Training towns	Evaluation towns	Testing towns
Town 1	Town 1	Town 2
Town 3	Town 3	Town 5
Town 4 - train routes	Town 4 - train routes	Town - 4 test routes
Town 6	Town 6	

4.3 Metrics

To quantify the success rates of the compared methods we assess scores of two metrics, namely route completion and distance completion. Given a set of routes for every setting, i.e. train, evaluation and testing, route completion is the average percentage of routes completed in that setting. In the same way, distance completion represents the percentage of distance completed to reach the goal, averaged over all routes in a setting. While route completion measures the agent's ability to reach the goal, distance completion assesses the extent to which the agent continues to advance, even if it fails to complete the route.

5 Results

For evaluating our proposed method according to the Leaderboard benchmark standard, we progressively produce 6 trained agents from 5 iterations of data aggregation and 1 initial iteration, each for our proposed method and the baseline method, as mentioned in section 4.1. Each of these trained agents are evaluated under train town-weather conditions (i.e. in familiar settings) and test town-weather conditions (i.e. unfamiliar settings). Since the simulation sets up the environment assets such as pedestrians and traffic agents at random, we run our evaluations 3 times with different random seeds. We then also report and draw conclusions from the average performances over the metrics mentioned in the section 4.3. Furthermore, as the recent work VPRPre (Juneja et al., 2023) aligns with the pre-training line of research and was implemented and evaluated under identical settings, we also incorporate its results into our comparison.

Table 3. Route completion (%) of driving agents on training and new (testing) conditions. Highest of all DAgger iterations reported.

Pre-training Method	Train town & weather	New town & weather
Baseline	77.33 ± 4	53.20 ± 1
VPRPre	81.33 ± 4	60.25 ± 2
DINO (ours)	72.67 ± 3	62.18 ± 7

Table 4. Distance completion (%) of driving agents on training and new (testing) conditions. Highest of all DAgger iterations reported.

Pre-training Method	Train town & weather	New town & weather
Baseline	89.36 ± 2	72.23 ± 6
VPRPre	91.97 ± 3	86.01 ± 0
DINO (ours)	86.04 ± 1	82.67 ± 6

We denote the best of the scores over all DAgger iterations, in Table 3 and Table 4. Under the routes completion metric in unfamiliar settings (new town & weather) in Table 3, DINO pre-training tends to perform better than baseline pre-training and VPRPre by $\approx 9\%$ and $\approx 2\%$ on average, respectively. Whereas under familiar settings (train town & weather), while baseline pre-training overtakes the performance of DINO pre-training, it shows signs of an over-fit as the baseline method fails to show generalisation in unfamiliar conditions. This conjecture is also supported by VPRPre's scores. A similar trend can be seen in Table 4 where the distance completion metric is compared. DINO pre-trained method is able to complete higher distance than the baseline method, and comes close to VPRPre's completed distance in unfamiliar settings. Hence with empirically calculated results in Table 3 and Table 4, our hypothesis aligns with the outcomes of the performed experiments.

We also compare the scores over both metrics at every iteration of data aggregation, at every random seed and we calculate the means of the random seeds. This can be seen in Figures 1 and 2. In comparison to the baseline, DINO pre-trained method not only shows better generalisation it shows reduced over-fit and faster convergence.

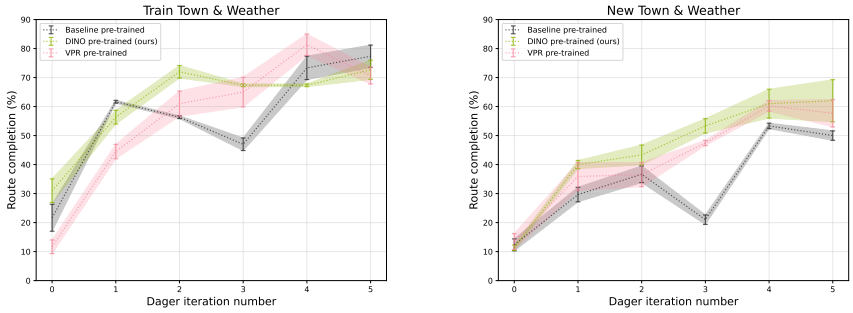

Fig. 1. Mean route completion (%) of evaluating agents over three seeds on the offline Leaderboard benchmark on training conditions (left) and testing conditions (right).

We further conjecture that the features learned in the encoder while pre-training in a non-label guided self-supervised way are much richer than the features learned while

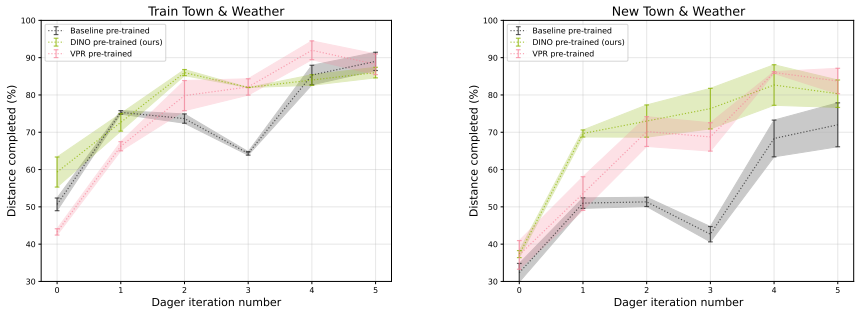

Fig. 2. Mean distance completion (%) of evaluating agents over three seeds on the offline Leaderboard benchmark on training conditions (left) and testing conditions (right).

training in a supervised way followed by the baseline method. The classical way of pre-training over ImageNet dataset with a classification loss may enable a kick-start in learning the task of interest, but due to its sole focus on a single concept of image understanding it does not converge as fast and successfully as method using DINO pre-training, as it can be seen in Figures 1 and 2. VPRPre's results show the ability of going further than our method's covered distances, yet it fails to complete as many routes. DINO pre-training is based on use of purely general set of training data relying on a much wider distribution, meanwhile VPRPre's pre-training involves exposure to training data captured in the CARLA simulator. Such exposure can be advantageous to include into DINO's pre-training and further improve results by increasing domain awareness.

Many of the current methods operate with a pre-trained ResNet vision encoder (which we choose as a baseline) and focus on exploring other parts of problem such as the decoder (Jia et al., 2023), higher field of view with better multi-view fusion (Xiao et al., 2023), attention enabled multi-modality (Chitta et al., 2022), and so on. Such an encoder is heavily guided by only classification labels to incorporate image understanding while training. Our work illustrates that dropping the reliance on such an encoder can be quite beneficial in terms of generalisation to transfer learning task of autonomous driving. Additionally this work highlights the need of better pre-training while aligning with other works (Zhang et al., 2022; Wu et al., 2023; Juneja et al., 2023) in this line of research.

6 Conclusion

We propose DINO-based pre-training of the vision encoder used for the task of learning end-to-end autonomous driving. Our experiments reveal that the suggested pre-training is more efficient in unseen environments than the popular classification-based pre-training. Moreover, DINO-based pre-training is conducted on an unrelated task and its effectiveness comes close to VPRPre (Juneja et al., 2023), which relies on additional domain awareness coming from its training data.

References

Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., et al. (2016). End to end learning for self-driving cars. *CoRR*, abs/1604.07316.

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., and Joulin, A. (2020). Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33:9912–9924.

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., and Joulin, A. (2021). Emerging properties in self-supervised vision transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*.

Chitta, K., Prakash, A., Jaeger, B., Yu, Z., Renz, K., and Geiger, A. (2022). Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Codevilla, F., Müller, M., López, A., Koltun, V., and Dosovitskiy, A. (2018). End-to-end driving via conditional imitation learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4693–4700. IEEE.

Codevilla, F., Santana, E., López, A. M., and Gaidon, A. (2019). Exploring the limitations of behavior cloning for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9329–9338.

Daniušis, P., Juneja, S., Valatka, L., and Petkevičius, L. (2021). Topological navigation graph framework. *Autonomous Robots*, 45:633–646.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. IEEE.

Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V. (2017). Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. (2020). Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Hu, A., Corrado, G., Griffiths, N., Murez, Z., Gurau, C., Yeo, H., Kendall, A., Cipolla, R., and Shotton, J. (2022). Model-based imitation learning for urban driving. *Advances in Neural Information Processing Systems*, 35:20703–20716.

Jia, X., Wu, P., Chen, L., Xie, J., He, C., Yan, J., and Li, H. (2023). Think twice before driving: Towards scalable decoders for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21983–21994.

Juneja, S., Daniušis, P., and Marcinkevičius, V. (2021). Combining multiple modalities with perceiver in imitation-based urban driving. *ALLSENSORS 2021, The Sixth International Conference on Advances in Sensors, Actuators, Metering and Sensing*.

Juneja, S., Daniušis, P., and Marcinkevičius, V. (2023). Visual place recognition pretraining for end-to-end trained autonomous driving agent. *IEEE Access*, 11:128421–128428.

Minaee, S., Mikolov, T., Nikzad, N., Chenaghlu, M., Socher, R., Amatriain, X., and Gao, J. (2024). Large language models: A survey. *arXiv preprint arXiv:2402.06196*.

Pomerleau, D. A. (1988). Alvinn: An autonomous land vehicle in a neural network. In *Proceedings of the 1st International Conference on Neural Information Processing Systems*, NIPS'88, page 305–313, Cambridge, MA, USA. MIT Press.

Prakash, A., Behl, A., Ohn-Bar, E., Chitta, K., and Geiger, A. (2020). Exploring data aggregation in policy learning for vision-based urban autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11763–11773.

Ross, S., J. Gordon, G., and Andrew Bagnell, J. (2010). A reduction of imitation learning and structured prediction to no-regret online learning. *J. Mach. Learn. Res.*, 15:627–635.

Tampuu, A., Matiisen, T., Semikin, M., Fishman, D., and Muhammad, N. (2020). A survey of end-to-end driving: Architectures and training methods. *IEEE Transactions on Neural Networks and Learning Systems*, 33(4):1364–1384.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Wu, P., Chen, L., Li, H., Jia, X., Yan, J., and Qiao, Y. (2023). Policy pre-training for autonomous driving via self-supervised geometric modeling. In *International Conference on Learning Representations*.

Xiao, Y., Codevilla, F., Gurram, A., Urfalioglu, O., and López, A. M. (2022). Multimodal end-to-end autonomous driving. *Trans. Intell. Transport. Sys.*, 23(1):537–547.

Xiao, Y., Codevilla, F., Porres, D., and Lopez, A. M. (2023). Scaling vision-based end-to-end driving with multi-view attention learning.

Yokoyama, N., Ha, S., Batra, D., Wang, J., and Bucher, B. (2024). Vlfm: Vision-language frontier maps for zero-shot semantic navigation. In *International Conference on Robotics and Automation (ICRA)*.

Zhang, J. and Cho, K. (2017). Query-efficient imitation learning for end-to-end simulated driving. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 2891–2897. AAAI Press.

Zhang, Q., Peng, Z., and Zhou, B. (2022). Learning to drive by watching youtube videos: Action-conditioned contrastive policy pretraining. *European Conference on Computer Vision (ECCV)*.

Zhang, Z., Liniger, A., Dai, D., Yu, F., and Van Gool, L. (2021). End-to-end urban driving by imitating a reinforcement learning coach. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 15222–15232.

Received July 10, 2024 , revised October 8, 2024, accepted November 3, 2024