Resolving Copycat Problems in Visual Imitation Learning via Residual Action Prediction

Chia-Chi Chuang[*1], Donglin Yang[*1], Chuan Wen[*1], and Yang Gao[†1,2]

[1] Institute for Interdisciplinary Information Sciences, Tsinghua University
[2] Shanghai Qi Zhi Institute
{zhuangjq19,ydl18,cwen20}@mails.tsinghua.edu.cn,
gaoyangiiis@tsinghua.edu.cn

Abstract. Imitation learning is a widely used policy learning method that enables intelligent agents to acquire complex skills from expert demonstrations. The input to the imitation learning algorithm is usually composed of both the current observation and historical observations since the most recent observation might not contain enough information. This is especially the case with image observations, where a single image only includes one view of the scene, and it suffers from a lack of motion information and object occlusions. In theory, providing multiple observations to the imitation learning agent will lead to better performance. However, surprisingly people find that sometimes imitation from observation histories performs worse than imitation from the most recent observation. In this paper, we explain this phenomenon from the information flow within the neural network perspective. We also propose a novel imitation learning neural network architecture that does not suffer from this issue by design. Furthermore, our method scales to high-dimensional image observations. Finally, we benchmark our approach on two widely used simulators, CARLA and MuJoCo, and it successfully alleviates the copycat problem and surpasses the existing solutions.

Keywords: Imitation learning; Autonomous driving; Copycat problem

1 Introduction

Learning to control in complex environments is a challenging task. Imitation learning is a powerful technique that learns useful skills from a pre-collected expert demonstration [44, 1, 32, 30, 17]. Compared with reinforcement learning [25, 33, 34], imitation learning is generally more data-efficient and does not require destructive exploration. Behavioral cloning (BC) [31, 20, 26, 13, 4] is one of the most widely used imitation algorithms. It directly mimics the expert behavior by learning the mapping from the observations to the expert actions with supervised

---

[*] Equal contribution.
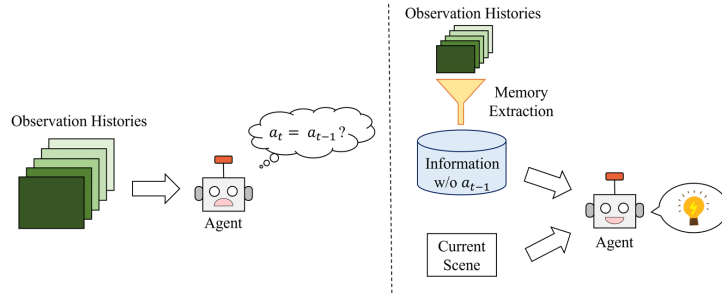[†] Corresponding author.

Fig. 1. The copycat problem & our solution: Left: Behavioral cloning from observation histories might learn a shortcut that directly outputs its previous action as the current action. Unfortunately, this leads to inferior performance during the test time. Right: We propose to solve the problem by a copycat-free memory extraction module. The history information is carefully extracted such that the shortcut no longer exists.

learning. Imitation learning has achieved many successes in the past in domains like autonomous driving [31, 4] and drone flying [22]. Imitation learning has also become an essential component in many other policy learning algorithms [32, 21].

In an Markov Decision Process (MDP), the input to the behavioral cloning algorithm can be the current state (Behavioral Cloning from Single Observation, i.e., BCSO). However, in practice, the agent's observation might be far from Markovian. This is the case, especially with visual observations. The most recent visual frame usually misses essential information, such as the objects' motion and appearance that are occluded in the current frame but visible in previous frames. Thus in many applications, it is more reasonable to provide the behavioral cloning algorithm with the observation histories (Behavioral Cloning from Observation Histories, i.e., BCOH) instead of a single observation. This allows the agent to access the state of the problem better.

However, recent works [14, 40] find that behavioral cloning with observation histories can sometimes perform even worse than behavioral cloning from the most recent observation. This phenomenon is counter-intuitive: the more complete input features (observation history) should lead to better downstream performance, but in practice, it won't! [14, 41–43] explain this phenomenon as BCOH is prone to learn an action prediction shortcut instead of the correct concept. BCOH predicts the current action a_t from the observation histories $o_t, o_{t-1}, \cdots$. However, the observation histories are actually obtained by executing previous expert actions $a_{t-1}, a_{t-2}, \cdots$ in the environments. Usually, the actions of an agent transit smoothly. Thus the BCOH agent tends to recover a_{t-1} from the observation histories and predict a_t based on that. This phenomenon is referred to as `copycat problem` or `inertia problem` in the literature [41, 42, 9]. It is a kind of causal confusion that severely limits the application of behavioral cloning algorithms to a broader range of the problem.

Early works [14, 41, 42] have proposed several approaches to resolve the causal confusion problem. However, they are either limited to only dealing with low-

dimensional state-based environments [41] or suffer from performance limitations when dealing with high-dimensional image observations [14, 42]. In this paper, we approach the problem from a neural network information flow view and propose a neural network architecture that does not have the copycat problem by design. Since the BCOH method learns the incorrect solution because the previous action a_{t-1} is a shortcut solution, we propose to design the neural network architecture such that it cannot learn to act based on the previous actions. Fig. 1 illustrates the concept of why copycat problem occurs and our solution. Unlike previous approaches, which achieve similar effects by using adversarial training [41] or using per sample re-weighting [42], our policy avoids the incorrect solution more thoroughly by cutting the information flow of the wrong clue. Thus, our method trains more easily and achieves better task performances.

We benchmark our method in a challenging autonomous driving environment, CARLA [10]. CARLA is a visually realistic driving simulator. It is widely used as a benchmark for autonomous driving and imitation learning [8, 16, 9]. Besides CARLA, we evaluate the performance in a commonly used robotics simulator, MuJoCo [39], with image-based observation. Our method outperforms all previous approaches on these challenging benchmarks.

2   Related Work

Imitation Learning and Copycat Problems: Imitation learning [44, 1, 30] is a powerful technology to learn complex policies from expert demonstrations. Among the different imitation learning methods, behavioral cloning is a simple but effective paradigm that directly regresses the expert actions from the observations. However, like other imitation methods, behavioral cloning suffers from the distributional shift that small errors will accumulate over time during testing and finally make the imperfect imitators encounter out-of-distribution states [32]. We focus on a specific phenomenon arising under the distributional shift in the partially observed settings – the copycat problem [41–43]. Although environmental interactions [17, 14, 5] or a queryable expert [32, 38, 19, 37, 36] can resolve it, purely offline methods [41, 2, 42, 43] are understudied. The specific definition and existing solutions for the copycat problem will be introduced in detail in Sec. 3.3. In this paper, we propose a new neural network architecture for imitation policies to solve the copycat problem more thoroughly.

Shortcut Learning: While the numerous success stories of deep neural networks (DNN) have rapidly spread over science, industry, and society, its limitations are coming into focus. For example, in computer vision, DNN image classifier tends to rely on texture rather than shape [12] and the background rather than objects [3, 43]. And in NLP, some language models turn out to depend on spurious features like word length without understanding the content of a sentence [28, 24]. [11] summarize this phenomenon and name it "shortcut learning," i.e., the DNNs prefer to learn the more straightforward solution (shortcut) rather than taking more effort to understand the intended solution. We regard the copycat problem as an instance of shortcut learning: when the scenes and

labels change smoothly, the neural networks tend to cheat by extrapolating from the historical information, a shortcut solution. Beyond imitation learning, the copycat problem also exists in other sequential tasks, such as robotics manipulation [23], tracking [46], forecasting [18], etc. Our method may generalize to these tasks, and we leave this to our future work.

3   Preliminaries

3.1   Partially Observed Markov Decision Process

The decision process of an agent equipped with sensors can be best described by a partially observed Markov decision process (POMDP) since the equipped sensors such as cameras or LIDARs can only perceive part of the environment at each time step. A POMDP can be formalized as a tuple $(\mathbf{S}, \mathbf{A}, \mathbf{T}, r, \mathbf{O})$, where $\mathbf{S}$ is the state space of the environment, $\mathbf{A}$ is the action space of the agent, $\mathbf{T}$ is the transition probabilities between states with a given action, $r : \mathbf{S} \times \mathbf{A} \to \mathbb{R}$ is the reward function and $\mathbf{O}$ is the observation space of the agent. In practice, the dimension of $\mathbf{S}$ is much greater than the dimension of $\mathbf{O}$. At each time step t, the agent has no access to the underlying true state s_t and has to take the action a_t according to the observation o_t. To deal with partial observation, it is common to take the historical information into account [27, 25, 2, 41, 42], constructing observation history $\tilde{o}_t = [o_t, o_{t-1}, \cdots, o_{t-H}]$ where H is the history length.

In the imitation learning setup, we are given a demonstration dataset composed of observation-action tuples: $\{(o_t, a_t)\}$. They represent good behaviors, i.e., the trajectories that achieve high accumulated rewards. However, the reward is not provided in the imitation learning setup. Instead, the goal of imitation learning is to learn a function that maps observation histories $\tilde{o}_t$ to the expert action a_t that leads to high accumulated rewards.

3.2   Behavioral Cloning:

Among all the variants of imitation learning, we focus on behavioral cloning (BC), a straightforward but powerful approach to mimic the expert behaviors from a pre-collected demonstration $\mathcal{D} = \{(o_i, a_i)\}_{i=1}^N$, where the N is the number of samples. BC reduces the complex policy learning into supervised learning by maximizing the log-likelihood of the action a_t conditioned on the observation history $\tilde{o}_t$ so that the agent can behave as similarly to the expert as possible, i.e., $\theta^* = \arg\max_\theta \mathbb{E}_\mathcal{D}[\log P(a_t|\tilde{o}_t; \theta)]$. $H = 0$ is BC from single observation (BCSO), and $H > 0$ is BC from observational histories (BCOH).

3.3   Copycat Problem

The copycat problem is a phenomenon that the BC agent tends to infer a "copycat" action similar to the previous action. When the copycat problem happens,
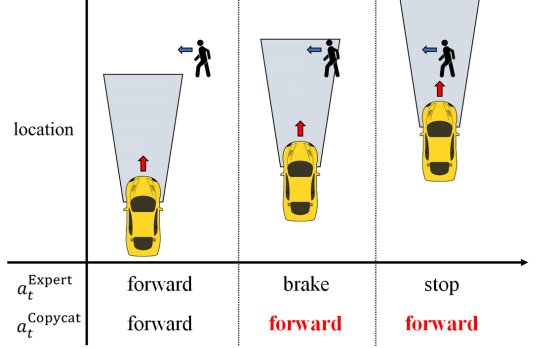
Fig. 2. An illustrative example of the copycat problem: The autonomous vehicle encounters a pedestrian. The trapezoidal area indicates the region can be observed by the driving agent. The expert agent would brake as soon as it sees the pedestrian, but the copycat agent tends to repeat previous actions and move forward.

both the training and the validation loss are low, but the in-environment evaluation has poor performance. It is not an instance of the over-fitting problem. Prior works [2, 9] showed using only a single observation o_t can achieve better performance than using multiple observations on their tasks. In addition, [14, 41, 42] show that the agent can completely fail when the input contains information about the previous actions. The copycat problem is a widely existing phenomenon in many downstream tasks when using BC, such as autonomous driving [42, 40] and robot control [14, 41].

Fig. 2 displays a concrete example of the copycat problem. In a driving scenario, the expert brakes immediately when it observes a person crossing the road so it can stop timely to prevent a traffic accident from happening. On the other hand, a copycat agent tends to keep its own previous action and usually brakes late or doesn't brake at all. Similar failure cases have been confirmed by multiple previous works [40, 42].

The reason why copycat problems happen in BC has been investigated in previous works [14, 41, 42]. It happens because there exists a shortcut from the input observation history $\tilde{o}_t$ to the target current action a_t. The input observation history is obtained by executing the expert actions $a_{t-H}, a_{t-H+1}, \cdots, a_{t-1}$ in the environment. Since most sequential decision problems exhibit a smooth transition property, expert actions also usually change smoothly. This means the previous action a_{t-1} is very close to the prediction target a_t. In many cases, the agent predicts the action a_t either completely or partially based on the information of the previous action a_{t-1}. However, decisions based on an agent's own previous action could be wrong in many cases, such as the vehicle stopping for the pedestrian case discussed above.

4   Methodology

Since the copycat problem is caused by the model learning the spurious prediction pathway from the previous action a_{t-1} to the current action a_t, in this section, we propose a neural network architecture that doesn't have this pathway by design. I.e., we would like our neural network to be unable to access the previous action a_{t-1} information according to the neural network architecture. We note that similar ideas have been explored by [41]. However, they propose to achieve this with adversarial networks, and we show that our solution has a much better scaling property than the previous one.
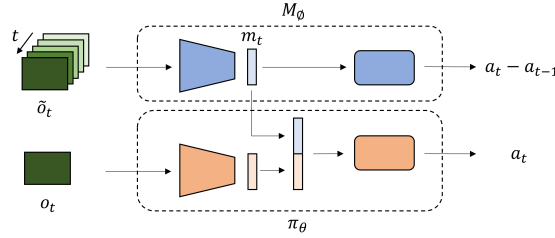

Fig. 3. Our framework: We propose to solve the problem with one memory extraction stream (upper stream M_ϕ) and a policy stream (lower stream π_θ). The memory extraction stream extracts copycat-free historical features, and the policy stream fuse the current observation and the history to output final decisions. Furthermore, blocks with the same color indicate they are updated in the same optimization step.

4.1   Model Architecture

Since the source of copycat problem is the nuisance information about a_{t-1} implied in the temporal information in the observation history $\tilde{o}_t$, the principle of designing the imitation model architecture should be to extract as much information as possible from the observations to predict the action a_t while ensuring that most information about a_{t-1} is removed. Only the observation history $\tilde{o}_t$ contains information about a_{t-1}, and the current observation o_t does not, so we split the neural network into two pathways: the $\tilde{o}_t$ pathway and the o_t pathway. We propose a **memory extraction module** to extract history-related features m_t from the $\tilde{o}_t$ input, such that it contains little information about a_{t-1}. Then we concatenate the feature m_t and the current observation o_t to predict the final action. We name the second network **policy network** since it fuses the history information and the current information to make the decision. Fig. 3 shows our framework, and we describe each of the components as follows.

Memory Extraction Module The memory extraction module M_ϕ aims to get the additional copycat-free feature from $\tilde{o}_t$ and compensate for the missing information due to the partial observation when predicting the action a_t. We

extract the intermediate embedding of M_ϕ, named memory feature m_t, to represent the historical information. To make m_t free from the copycat problem, we design a specific prediction objective for the memory extraction module M_ϕ. As discussed in Sec. 3.3, the copycat problem is mainly caused by the excessive information about a_{t-1} in $\tilde{o}_t$. Motivated by this, m_t is expected to contain the essential information for predicting the action a_t while avoiding the copycat problem by retaining as little information about a_{t-1} as possible. I.e. M_ϕ should be trained to maximize the mutual information between m_t and a_t while minimizing that between m_t and a_{t-1}. This goal can be formalized as a maximizing conditional mutual information objective:

$$\phi^* = \arg\max_\phi I_\phi(m_t; a_t|a_{t-1}). \tag{1}$$

This conditional mutual information maximization objective allows the learned representation m_t to contain as much unique information about a_t as possible. Furthermore, since a_{t-1} is in the condition, there is no incentive for m_t to contain any information about a_{t-1}. Due to the limited model capacity and training data, more unique information about a_t in m_t results in the suppression of information about a_{t-1}. We remark that though this objective cannot guarantee to remove all information about a_{t-1} from m_t, the results in Section 5.5 indicate that it is sufficient to resolve most of the copycat problems in the visual tasks.

In practice, Eq. (1) is challenging to implement and unstable to optimize. Therefore, we propose to optimize over a lower bound on the conditional mutual information to learn the representation m_t better.

Theorem 1 (lower bound of Eq. (1)). *Let $r_t := a_t - a_{t-1}$ be the action residual, then we have $I_\phi(m_t, a_t|a_{t-1}) \geq \mathbf{H}(r_t|a_{t-1}) - \mathbf{H}_\phi(r_t|m_t)$, where $\mathbf{H}$ represents the Shannon entropy.*

We can maximize this lower bound to approximate Eq. (1). Because the first term is not correlated with the parameter ϕ, the objective to optimize M_ϕ can be rewrited as $\phi^* = \arg\min_\phi H_\phi(r_t|m_t)$. Furthermore, minimizing the conditional mutual information is equivalent to the maximum likelihood estimation, so we propose an **action residual prediction** objective for the memory extraction module M_ϕ:

$$\phi = \arg\max_\phi \mathbb{E}_\mathcal{D}[\log P(a_t - a_{t-1}|\tilde{o}_t; \phi)]. \tag{2}$$

Intuitively, this approximated objective function aims to predict the change of actions. Since the expert actions in a decision process usually transit smoothly, predicting the action change approximately predicts the critical differences in the environments that need expert action change.

Trained with Eq. (2), the memory extraction module M_ϕ optimizes over the objective in Eq. (1) as well, i.e. extracting the sufficient information about a_t and removing the shortcut information about a_{t-1}. Therefore, the memory feature m_t provides additional information from history and cuts off the shortcut path $a_{t-1} \rightarrow a_t$ by removing most information about a_{t-1}. With the proposed architecture, the policy model π_θ will take the advantage of both the visual clues in o_t and the complementary information provided by m_t to learn the correct way for predicting a_t.

Policy Module The policy module π_θ takes the current observation o_t and the memory feature m_t as input and predicts a_t, where m_t is extracted from $M_\phi(\tilde{o}_t)$ and contains additional and copycat-free historical information about predicting a_t. In practice, we process the observation with a convolutional neural network and fuse the mid-layer representation with the memory extraction module by concatenation. The objective of the policy model is:

$$\theta^* = \arg\max_\theta \mathbb{E}_\mathcal{D}[\log P(a_t|o_t, m_t; \theta)]. \qquad (3)$$

4.2   Implementation details

In our implementation, we feed o_t and $\tilde{o}_t$ into π_θ and M_ϕ respectively. We take the intermediate feature of M_ϕ as m_t and then fuse it into π_θ. Specifically, we directly concatenate m_t with the intermediate feature of π_θ through a stop-gradient layer and feed them into the following layers of π_θ. The respective targets of π_θ and M_ϕ are a_t and $a_t - a_{t-1}$. π_θ and M_ϕ are trained jointly with the objectives Eq. (3) and Eq. (2). We will introduce experiment setup and other implementation details in Sec. 5.1 and Appendix respectively.

5   Experiments

In this section, we aim to answer the following questions. 1) How well does our method performs? Does it outperform the previously proposed methods? 2) We would like to understand the role of the memory extraction module and provide empirical evidence that it indeed excludes information about a_{t-1}, thus resolving the copycat problem. 3) We would like to understand in what circumstances the proposed model is better qualitatively.

To answer these questions and evaluate our method, we conduct experiments in the CARLA [10] autonomous driving simulator and three standard OpenAI Gym MuJoCo [39] continuous control environments: Hopper, HalfCheetah and Walker2D.

5.1   Experiment Setup

CARLA[10]: CARLA is a photo-realistic autonomous driving simulator that features realistic driving scenarios. CARLA is also the most challenging and widely used environment to benchmark how well the copycat problem is resolved in the literature [42]. We mostly follow the experimental setting from [9], but withhold the ego-agent velocity from the observation for all methods following [42] to ensure the environment is partially observed. The expert demonstration dataset is collected by a scripted expert. Under the hood, the expert knows the scene layout, including information such as the location and speed of other vehicles and pedestrians.

Each expert driving trajectory can be described by a list of (o_t, c_t, a_t) tuples, where t indexes the timestep. Here o_t is the observation in the form of an

RGB image. In our setup, the image is resized to 200×88. c_t is the driving command (one of the following commands: follow the lane, turn left, turn right, and go straight). The driving command is necessary because one has to command the autonomous driving agent about where to go. Finally, $a_t \in [-1, 1]^2$ is a two-dimensional vector for controlling the steer and acceleration. Here the positive acceleration means throttle, and negative means brake. We use the CARLA100 [9] dataset that consists of 100 hours of driving data. Table 1 shows actions are smooth most of the time, so we can effectively verify whether the method can deal with the copycat problem. The BC agent aims to learn a function $f(\tilde{o}_t, c_t) \to a_t$. We follow previous setups [42] and use $H = 6$ for BCOH.

Table 1. The distribution of $||a_t - a_{t-1}||_2^2$ in CARLA100

$[0, 10^{-3})$	$[10^{-3}, 10^{-2})$	$[10^{-2}, 10^{-1})$	$\geq 10^{-1}$
68.8%	7.8%	14.3%	9.0%

CARLA *NoCrash* benchmark [9]:   *NoCrash* benchmark [9] is a series of navigation tasks consisting of 25 routes over 4 kinds of weather. Finishing a route means following the given route within the time constraint without collisions. There are three traffic conditions: *Empty*, *Regular*, and *Dense* from easy to difficult. We focus on the *Dense* and *Regular* to ensure the agent learns the information about pedestrians or vehicles on the road. In addition to the same weathers and town as the training data, we evaluate the performance in the new town and the new weather. For each experiment, we count the number of successful episode denoted as $\#SUCCESS$ to be the main performance metric. We will describe model detail and other important metrics in Appendix.

MuJoCo-Image(Hopper,Halfcheetah,Walker2D) [39]: Following the same setting in [42], we set the observation o_t at time step t to be a 128×128 RGB image of the rendering environment and $H = 1$ for BCOH. These control tasks have various state and action spaces, transition dynamics, and reward functions. Expert datasets are collected by a TRPO agent [33] with access to fully observed states. Each dataset is a list of (o_t, a_t) tuples, where t denotes the timestep. We collected 1k samples for HalfCheetah, and 20k for Hopper and Walker2D. The BC agent is required to learn a function $f(\tilde{o}_t) \to a_t$. The performance metric is the average reward from the environment. We will report the details about the model architectures in Appendix.

5.2 Previous Methods

We compare our method to previous methods that are proposed to solve the copycat problem. We introduce those methods as follows.

BCSO & BCOH: For CARLA, We use the model proposed in [9] as the baseline model, and it was proposed along with CARLA *Nocrash* benchmark. For MuJoCo, we follow the model in [42].

DAGGER [32]: DAGGER is a widely used online method in BC to deal with the causal confusion by querying experts to label new trajectories generated by

the agent and then updating it. We note that this is considered an oracle method since DAGGER requires extra online interactions and expert labels. Therefore, we set the number of online queries to 150k for CARLA and 1k for MuJoCo.

Historical-Dropout (HD) [2]:   **HD** randomly dropouts historical frames to cope with the copycat problem. In practice, we add a dropout layer with a probability of 0.5 on past observations $o_{t-1}, o_{t-2}, \cdots, o_{t-H}$.

Fighting Copycat Agent (FCA) [41]:   **FCA** is an adversarial training approach to remove the information about the previous action in the observation histories. It encodes the observation histories to a latent vector, asks it to predict the current action with supervised loss, and asks it not to predict the previous action using adversarial loss.

Keyframe-Focused (Keyframe) [42] :   **Keyframe** re-weights the samples according to a weighting defined by how fast action changes. The places where action changes unexpectedly are considered as important timesteps.

Note that there are some other methods [7, 29, 45, 6] that achieve better performance on the CARLA benchmark. However, our goal is to resolve the copycat problem, instead of achieving the best results on CARLA. Following prior works [42], we take [9] as the base algorithm. We expect our techniques will also benefits more complicated imitation methods.

5.3   Results

We report the mean and standard deviation of the performance metric for all entries and analyze the experiment results in this section.

Table 2. The #*SUCCESS* on CARLA *Nocrash* benchmark. For each method, we train 3 policies from different initial seeds.

Environment	Train Town & Weather		New Weather		New Town	
Traffic	*Regular*	*Dense*	*Regular*	*Dense*	*Regular*	*Dense*
BCSO	37.6 ± 6.1	13.1 ± 1.8	18.3 ± 1.7	5.7 ± 5.7	10.3 ± 2.9	2.0 ± 0.8
BCOH	67.1 ± 10.8	34.1 ± 7.5	26.0 ± 6.5	6.0 ± 2.9	27.0 ± 5.0	15.3 ± 2.9
OURS	$\mathbf{78.1 \pm 0.9}$	$\mathbf{52.0 \pm 2.3}$	$\mathbf{39.3 \pm 5.8}$	$\mathbf{19.0 \pm 2.7}$	$\mathbf{40.7 \pm 7.0}$	$\mathbf{25.7 \pm 4.2}$
DAGGER	69.7 ± 8.4	42.7 ± 5.7	24.7 ± 4.8	14.7 ± 4.2	34.0 ± 4.3	12.0 ± 4.5
HD	70.1 ± 4.0	35.6 ± 3.5	28.0 ± 6.5	16.7 ± 6.2	32.0 ± 4.3	11.3 ± 2.5
FCA	58.0 ± 8.0	31.2 ± 5.2	20.0 ± 11.5	8.7 ± 4.0	21.3 ± 3.1	8.3 ± 2.9
Keyframe	74.4 ± 7.3	41.9 ± 6.2	34.0 ± 5.4	13.7 ± 5.4	33.3 ± 7.3	16.3 ± 3.9

CARLA *Nocrash* benchmark results are shown in the Table 2. We conclude that our method achieves the best performance compared to all previous approaches, including the oracle approach DAGGER. Besides our superior performance, we would also like to describe the results qualitatively. BCSO in CARLA severely fails because the agent can not observe the motion of the surrounding agent, as well as that of itself. Thus, BCOH is necessary in this case. We observe severe copycat problems with BCOH. A lot of failure cases for BCOH are when it is stopped for some obstacle, and it cannot restart again due to the copycat

issue. This leads to a high timeout failure rate. Fig. 4 shows the copycat problem of BCOH and these are the main reasons for the poor performance. The oracle method DAGGER is better than baseline in the *Dense* traffic case but similar to baseline in *Regular* traffic. We hypothesize that the *Dense* traffic evaluation has a significant distributional shift compared to the training data. HD performs similarly to BCOH, indicating it can not solve the copycat problem in this challenging case. FCA fails to outperform the BCOH baseline because it can not handle the large complex input space with the adversarial training mechanism which was also mentioned in [42]. Keyframe significantly outperforms the BCOH baseline but is still worse than our method.

Fig. 4. Visualizations of BCOH and our method in CARLA: Each row is a video sequence. The first two rows show a case where BCOH is stopped for a pedestrian and never restarts, while our method handles that successfully. The last two rows show a case where BCOH passes an intersection and then fails to stop behind a vehicle while our method executes the proper behavior. The "throttle/keep/brake" indicates the acceleration of the agent.

MuJoCo-Image results are shown in Table 3. BCOH yields higher rewards than BCSO due to access to motion information, and the performance can be further improved by addressing the copycat problem. DAGGER performs well on MuJoCo and gets the best results in Hopper. HD fails in these settings and has a similar performance to BCSO, indicating it cannot resolve the copycat problem. FCA yields a lower reward than BCOH because it is difficult to handle high-dimensional input with adversarial learning. For the other offline method Keyframe, though it significantly improves the performance, ours outperforms it in Walker2D and HalfCheetah and achieves a comparable performance in Hopper. Results on more MuJoCo environments are shown in Appendix.

Table 3. The average reward in MuJoCo-Image. For each method and task, we train 5 policies from different initial seeds.

Method	BCSO	BCOH	OURS	DAGGER	HD	FCA	Keyframe
Hopper	601 ± 168	740 ± 35	894 ± 38	$\mathbf{1034 \pm 45}$	617 ± 111	735 ± 106	951 ± 117
Walker2D	481 ± 40	614 ± 107	$\mathbf{800 \pm 58}$	699 ± 111	594 ± 61	534 ± 99	769 ± 91
HalfCheetah	4 ± 5	615 ± 41	$\mathbf{914 \pm 115}$	822 ± 186	96 ± 40	270 ± 168	819 ± 96

5.4   Ablation Studies

To evaluate the effect of each module and technique we propose in Sec. 4.1, we conduct ablation study experiments in CARLA *Nocrash Dense*. The results are shown in Table 4, and we provide detailed analysis for each ablation, respectively.

Table 4. Ablation results in CARLA *Nocrash Dense*

Architecture	$\#Success(\uparrow)$
Ours	$\mathbf{52.0 \pm 2.3}$
Memory only: residual controller	0.0 ± 0.0
Memory only: learned controller	0.0 ± 0.0
Memory module objective: a_t	41.3 ± 1.9
Memory module objective: a_{t-1}	47.0 ± 5.1
Without stop-gradient layer	45.0 ± 5.3

The memory extracted module only: We want to show both parts of our framework are necessary. From the BCSO result, we know the policy module alone achieves inferior performance. To evaluate the memory extraction module alone, we set up two experiments that only uses the well-trained memory extracted module to derive the action: one uses the residual directly (**Memory only: residual controller**), i.e., it uses the output of the memory extracted module plus the previous action to get the current action $a_t = a_{t-1} + r_t$; the other uses the extracted feature (**Memory only: learned controller**) to control the vehicle with the same controller as the baseline model, i.e., the controller is trained by using m_t from memory extracted module to fit a_t. However, the imitators in both setups fail in every episode. It indicates the memory extracted feature m_t alone contains insufficient information for the task. The residual prediction not only removes the information about a_{t-1} but also erases some necessary information about the current scene. Therefore, the policy module completes information related to control by adding the current observation o_t. This illustrates that the policy stream is an indispensable part of our framework.

Alternative memory module objectives: Moreover, we would like to investigate the performance of alternative memory module designs. Our design is based on a lower bound of the conditional mutual information objective. Here we train the model from scratch by replacing the memory module prediction objective to a_t and a_{t-1} respectively. As shown in **memory module objective:** a_t and **memory module objective:** a_{t-1}, both of the objective changes lead

to deterioration of the performances. This indicates that other features are less informative or more likely to suffer from the copycat problem.

Without stop-gradient layer: The stop-gradient is applied to the memory feature m_t to ensure the policy module doesn't directly learn shortcut information from the observation histories. After removing it, the result shows the performance degradation, indicating that the imitation learner might learn some copycat information from the observation histories. Though the action residual prediction objective removes some shortcut information about a_{t-1} and improves the performance compared to BCOH, the stop-gradient is indispensable for the policy module to cut off the nuisance information flow from observation histories.

5.5 Analysis

The performance of our framework is clearly above that of alternative methods on the CARLA *Nocrash* benchmark. In this subsection, we prove that the training objective proposed in Eq. (2) can successfully resolve the copycat problem. We design two analytical experiments to evaluate our method qualitatively. The first analysis directly proves our method has fewer copycat behaviors by an intervention experiment. The second one validates our training objective indeed reduces the mutual information between a_{t-1} and m_t. We choose BCOH and Keyframe as the baselines for these two analytical experiments.

Table 5. The analysis results on the validation set

Method	BCOH	Keyframe	OURS
Change(%)($\downarrow$)	40.88	22.96	**16.18**
MSE loss $\times 10^{-2}$($\uparrow$)	4.83	5.04	**7.50**

Intervention on Observation Histories To verify whether our model learns the causal effect of the observation histories $\tilde{o}_t$, we propose an intervention analysis. We replace the original $\tilde{o}_t$ with a counterfactual history $do(\tilde{o}_t) = [o_t, o_t, \cdots, o_t]$, i.e., all histories are the current observation o_t. For a copycat agent, then with the counterfactual observation histories $do(\tilde{o}_t)$ as the input, it will have an illusion that the vehicle is in the stationary state and will merely repeat the previous motion of staying still. A causally correct agent should perform according to the content of the scene. We test our model and the baseline models under the cases where the vehicles have positive speed and the latest action is non-stop (denoted by $\pi(\cdot) > 0$ with a slight abuse of notation). A causally correct agent should drive forward in this case, i.e. $\pi(do(\tilde{o}_t)) > 0$. We count the percentage where the model stopped on the intervened observation, i.e., $\frac{N(v>0, \pi(\tilde{o}_t)>0, \pi(do(\tilde{o}_t))=0)}{N(v>0, \pi(\tilde{o}_t)>0)}$ where $N(.)$ denotes the counting function. The behavior change percentage can be regarded as the error rate of this experiment.

The 2nd row of Tab. 5 shows the intervention results. As shown in the table, 40.88% of samples change in BCOH, even there is no need for the agent to stop

in these scenes (e.g., no vehicles or pedestrians ahead; the traffic signal is not red). It illustrates that BCOH makes decisions by merely repeating the previous action but ignoring the perception of the current scene in nearly half of the cases. The metric of Keyframe is relatively lower than that of BCOH, which indicates Keyframe does not rely just on the previous action and partially mitigates the copycat problem. Meanwhile, OURS is only 16.18%, showing that our approach significantly alleviates the copycat problem.

Mutual Information In Sec. 4, we set up a conditional mutual information objective, i.e. maximize $I(m_t; a_t | a_{t-1})$. And we use an approximation to this objective during the optimization process. In this analysis, we empirically evaluate how much information about a_{t-1} is left in m_t. This would help to justify both the objective function as well as the approximation. More concretely, we regress a_{t-1} from the representation with a 2-layer MLP. We calculate the validation loss of a_{t-1} and use it as a proxy to the negative of the mutual information between the representation and a_{t-1}. The less the information, the less likely the method would suffer from the copycat issue. We evaluate our method's representation m_t and corresponding layers of the BCOH and the Keyframe methods.

The results in the 3rd row of Tab. 5 show that the feature in BCOH is more predictive of the previous action. This conclusion matches the preliminary results that BCOH is more prone to copycat problems and learns the spurious temporal correlation. The MSE of Keyframe is slightly larger than that of BCOH, which shows that this method removes part of the information about the previous action and mitigates the copycat problem. Meanwhile, the MSE of OURS is the largest, which indicates that the extracted feature m_t in our model contains the smallest amount of mutual information with a_{t-1} and cuts off the shortcut of simply imitating the previous action from observation histories.

6   Conclusion

Behavioral cloning is a fundamental algorithm that helps the agent to imitate complex behaviors. However, it suffers from causal issues due to the temporal structure of the problem. We view the causal issue from an information flow perspective and propose a simple yet effective method to improve performance drastically. Our method outperforms previous solutions to the copycat problem. In the future, we would like to investigate casual issues in a broader range of algorithms, e.g., reinforcement learning and other sequential decision problems.

7   Acknowledgement

References

1. Argall, B.D., Chernova, S., Veloso, M.M., Browning, B.: A survey of robot learning from demonstration. Robotics Auton. Syst. **57**(5), 469–483 (2009)
2. Bansal, M., Krizhevsky, A., Ogale, A.S.: Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. In: Bicchi, A., Kress-Gazit, H., Hutchinson, S. (eds.) Robotics: Science and Systems XV, University of Freiburg, Freiburg im Breisgau, Germany, June 22-26, 2019 (2019)
3. Beery, S., Horn, G.V., Perona, P.: Recognition in terra incognita. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XVI. Lecture Notes in Computer Science, vol. 11220, pp. 472–489. Springer (2018)
4. Bojarski, M., Testa, D.D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L.D., Monfort, M., Muller, U., Zhang, J., Zhang, X., Zhao, J., Zieba, K.: End to end learning for self-driving cars. CoRR **abs/1604.07316** (2016)
5. Brantley, K., Sun, W., Henaff, M.: Disagreement-regularized imitation learning. In: 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020. OpenReview.net (2020)
6. Chen, D., Koltun, V., Krähenbühl, P.: Learning to drive from a world on rails. CoRR **abs/2105.00636** (2021)
7. Chen, D., Zhou, B., Koltun, V., Krähenbühl, P.: Learning by cheating. In: Kaelbling, L.P., Kragic, D., Sugiura, K. (eds.) 3rd Annual Conference on Robot Learning, CoRL 2019, Osaka, Japan, October 30 - November 1, 2019, Proceedings. Proceedings of Machine Learning Research, vol. 100, pp. 66–75. PMLR (2019)
8. Codevilla, F., Müller, M., López, A.M., Koltun, V., Dosovitskiy, A.: End-to-end driving via conditional imitation learning. In: 2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018. pp. 1–9. IEEE (2018)
9. Codevilla, F., Santana, E., López, A.M., Gaidon, A.: Exploring the limitations of behavior cloning for autonomous driving. In: 2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019. pp. 9328–9337. IEEE (2019)
10. Dosovitskiy, A., Ros, G., Codevilla, F., López, A.M., Koltun, V.: CARLA: an open urban driving simulator. In: 1st Annual Conference on Robot Learning, CoRL 2017, Mountain View, California, USA, November 13-15, 2017, Proceedings. Proceedings of Machine Learning Research, vol. 78, pp. 1–16. PMLR (2017)
11. Geirhos, R., Jacobsen, J., Michaelis, C., Zemel, R.S., Brendel, W., Bethge, M., Wichmann, F.A.: Shortcut learning in deep neural networks. Nat. Mach. Intell. **2**(11), 665–673 (2020)
12. Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F.A., Brendel, W.: Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019. OpenReview.net (2019)
13. Giusti, A., Guzzi, J., Ciresan, D.C., He, F., Rodriguez, J.P., Fontana, F., Faessler, M., Forster, C., Schmidhuber, J., Caro, G.D., Scaramuzza, D., Gambardella, L.M.: A machine learning approach to visual perception of forest trails for mobile robots. IEEE Robotics Autom. Lett. **1**(2), 661–667 (2016)
14. de Haan, P., Jayaraman, D., Levine, S.: Causal confusion in imitation learning. In: Wallach, H.M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E.B.,

Garnett, R. (eds.) Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada. pp. 11693–11704 (2019)

15. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016. pp. 770–778. IEEE Computer Society (2016)

16. Heinze-Deml, C., Meinshausen, N.: Conditional variance penalties and domain shift robustness. Mach. Learn. **110**(2), 303–348 (2021)

17. Ho, J., Ermon, S.: Generative adversarial imitation learning. In: Lee, D.D., Sugiyama, M., von Luxburg, U., Guyon, I., Garnett, R. (eds.) Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain. pp. 4565–4573 (2016)

18. Hu, P., Huang, A., Dolan, J.M., Held, D., Ramanan, D.: Safe local motion planning with self-supervised freespace forecasting. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021. pp. 12732–12741. Computer Vision Foundation / IEEE (2021)

19. Laskey, M., Lee, J., Fox, R., Dragan, A.D., Goldberg, K.: DART: noise injection for robust imitation learning. In: 1st Annual Conference on Robot Learning, CoRL 2017, Mountain View, California, USA, November 13-15, 2017, Proceedings. Proceedings of Machine Learning Research, vol. 78, pp. 143–156. PMLR (2017)

20. LeCun, Y., Muller, U., Ben, J., Cosatto, E., Flepp, B.: Off-road obstacle avoidance through end-to-end learning. In: Advances in Neural Information Processing Systems 18 [Neural Information Processing Systems, NIPS 2005, December 5-8, 2005, Vancouver, British Columbia, Canada]. pp. 739–746 (2005)

21. Levine, S., Koltun, V.: Guided policy search. In: Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013. JMLR Workshop and Conference Proceedings, vol. 28, pp. 1–9. JMLR.org (2013)

22. Loquercio, A., Kaufmann, E., Ranftl, R., Dosovitskiy, A., Koltun, V., Scaramuzza, D.: Deep drone racing: From simulation to reality with domain randomization. IEEE Trans. Robotics **36**(1), 1–14 (2020)

23. Mandlekar, A., Xu, D., Wong, J., Nasiriany, S., Wang, C., Kulkarni, R., Fei-Fei, L., Savarese, S., Zhu, Y., Martín-Martín, R.: What matters in learning from offline human demonstrations for robot manipulation. In: Faust, A., Hsu, D., Neumann, G. (eds.) Conference on Robot Learning, 8-11 November 2021, London, UK. Proceedings of Machine Learning Research, vol. 164, pp. 1678–1690. PMLR (2021)

24. McCoy, T., Pavlick, E., Linzen, T.: Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In: Korhonen, A., Traum, D.R., Màrquez, L. (eds.) Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers. pp. 3428–3448. Association for Computational Linguistics (2019)

25. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M.A., Fidjeland, A., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., Hassabis, D.: Human-level control through deep reinforcement learning. Nat. **518**(7540), 529–533 (2015)

26. Mülling, K., Kober, J., Kroemer, O., Peters, J.: Learning to select and generalize striking movements in robot table tennis. Int. J. Robotics Res. **32**(3), 263–279 (2013)

27. Murphy, K.: A survey of pomdp solution techniques. Environment **2** (10 2000)
28. Niven, T., Kao, H.: Probing neural network comprehension of natural language arguments. In: Korhonen, A., Traum, D.R., Màrquez, L. (eds.) Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers. pp. 4658–4664. Association for Computational Linguistics (2019)
29. Ohn-Bar, E., Prakash, A., Behl, A., Chitta, K., Geiger, A.: Learning situational driving. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020. pp. 11293–11302. Computer Vision Foundation / IEEE (2020)
30. Osa, T., Pajarinen, J., Neumann, G., Bagnell, J.A., Abbeel, P., Peters, J.: An algorithmic perspective on imitation learning. CoRR **abs/1811.06711** (2018)
31. Pomerleau, D.: ALVINN: an autonomous land vehicle in a neural network. In: Touretzky, D.S. (ed.) Advances in Neural Information Processing Systems 1, [NIPS Conference, Denver, Colorado, USA, 1988]. pp. 305–313. Morgan Kaufmann (1988)
32. Ross, S., Gordon, G.J., Bagnell, D.: A reduction of imitation learning and structured prediction to no-regret online learning. In: Gordon, G.J., Dunson, D.B., Dudík, M. (eds.) Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011. JMLR Proceedings, vol. 15, pp. 627–635. JMLR.org (2011)
33. Schulman, J., Levine, S., Moritz, P., Jordan, M.I., Abbeel, P.: Trust region policy optimization. CoRR **abs/1502.05477** (2015)
34. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. CoRR **abs/1707.06347** (2017)
35. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. Int. J. Comput. Vis. **128**(2), 336–359 (2020)
36. Spencer, J.C., Choudhury, S., Venkatraman, A., Ziebart, B.D., Bagnell, J.A.: Feedback in imitation learning: The three regimes of covariate shift. CoRR **abs/2102.02872** (2021)
37. Sun, W., Bagnell, J.A., Boots, B.: Truncated horizon policy search: Combining reinforcement learning & imitation learning. In: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings. OpenReview.net (2018)
38. Sun, W., Venkatraman, A., Gordon, G.J., Boots, B., Bagnell, J.A.: Deeply aggrevated: Differentiable imitation learning for sequential prediction. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017. Proceedings of Machine Learning Research, vol. 70, pp. 3309–3318. PMLR (2017)
39. Todorov, E., Erez, T., Tassa, Y.: Mujoco: A physics engine for model-based control. In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2012, Vilamoura, Algarve, Portugal, October 7-12, 2012. pp. 5026–5033. IEEE (2012)
40. Wang, D., Devin, C., Cai, Q., Krähenbühl, P., Darrell, T.: Monocular plan view networks for autonomous driving. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2019, Macau, SAR, China, November 3-8, 2019. pp. 2876–2883. IEEE (2019)
41. Wen, C., Lin, J., Darrell, T., Jayaraman, D., Gao, Y.: Fighting copycat agents in behavioral cloning from observation histories. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (eds.) Advances in Neural Information Processing

Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual (2020)
42. Wen, C., Lin, J., Qian, J., Gao, Y., Jayaraman, D.: Keyframe-focused visual imitation learning. In: Meila, M., Zhang, T. (eds.) Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event. Proceedings of Machine Learning Research, vol. 139, pp. 11123–11133. PMLR (2021)
43. Wen, C., Qian, J., Lin, J., Teng, J., Jayaraman, D., Gao, Y.: Fighting fire with fire: Avoiding dnn shortcuts through priming. In: International Conference on Machine Learning. pp. 23723–23750. PMLR (2022)
44. Widrow, B., Smith, F.W.: Pattern-recognizing control systems (1964)
45. Zhang, Z., Liniger, A., Dai, D., Yu, F., Gool, L.V.: End-to-end urban driving by imitating a reinforcement learning coach. CoRR **abs/2108.08265** (2021)
46. Zhou, X., Koltun, V., Krähenbühl, P.: Tracking objects as points. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J. (eds.) Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part IV. Lecture Notes in Computer Science, vol. 12349, pp. 474–490. Springer (2020)