

A APPENDIX

A.1 TRAINING PROCESS

The following Alg.1 provides a detailed description of the training process for the proposed C²INet method.

Algorithm 1 Training Process of C²INet.

Input: The training trajectories from K task domains $\{X_i, Y_i\}_{i=1}^K$, with a maximum prior queue capacity of γ , training L epochs for each task.

Output: The optimized model parameters θ^* .

```

1: for each  $i \in [1, K]$  do
2:   for each  $j \in [1, L]$  do
3:     if  $j = 1$  then
4:       if Online Mode then
5:         Sample  $S$  trajectories from the current task.
6:         Obtain the initial prior based on the sampled data and enqueue it.
7:       else if Offline Mode then
8:         Cluster the accessible data and enqueue the cluster centers.
9:       end if
10:    end if
11:    Maximize the loss function of the causal intervention model Eq.10 to optimize model parameters  $\theta$ .
12:    if  $j \bmod \lfloor \frac{L}{\gamma} \rfloor = 0$  then
13:      if Online Mode then
14:        Calculate the new component based on Eq.8 and Eq.9 and add it to the prior queue.
15:      else if Offline Mode then
16:        Optimize the obtained components in the prior queue based on Eq.8 and Eq.9.
17:      end if
18:    end if
19:  end for
20:  if the queue length exceeds  $\gamma$  then
21:    Pruning is performed according to Eq.10 until the quantity is reduced below  $\gamma$ .
22:  end if
23: end for
24: return The optimized model parameters  $\theta^*$ 

```

A.2 DETAILS SUPPLEMENT

A.2.1 DERIVATIONS OF THE CAUSAL INTERVENTION IN SECTION 2.3

We begin by introducing causal interventions and the related do-calculus techniques, which can be regarded as the axioms underlying our methodological derivations. Let us assume $P(Y|X)$ represents the conditional probability of Y given the variable X as an observed known. Clearly, within the system, if other confounding factors are informationally associated with x , then $P(Y|X)$ cannot accurately measure the direct relationship from X to Y . To address this issue, the formula $P(Y|do(X))$ is used to represent an intervention applied to X , specifically assigning a fixed value $X = x$, severing the informational pathways from other variables to X , thereby obtaining the direct effect between the variables. The identification of causal effects here follows the backdoor criterion (Pearl, 2016). As we will see, the do-calculus provides us with tools to identify causal effects using the causal assumptions encoded in the causal graph. It consists of three inference rules that allow us to map interventional and observational distributions whenever certain conditions are satisfied in the Structural Causal Models G , which is a Directed Acyclic Graph (DAG) that describes causal attributes and their interactions (Neal, 2020). Let X, Y, Z , and W be arbitrary disjoint sets of nodes in a causal DAG G . Let $G_{\overline{X}}$ denote the graph obtained by deleting from G all arrows pointing to nodes in X and $G_{\underline{X}}$ denote the graph obtained by deleting from G all arrows emerging from nodes

in X . To represent the deletion of both incoming and outgoing arrows, we use the notation $G_{\overline{X}\underline{Z}}$. The following three rules are valid for every interventional distribution compatible with G .

Rule 1 (Insertion/deletion of observations):

$$P(y|do(x), z, w) = P(y|do(x), w), \text{ if } (Y \perp\!\!\!\perp Z|X, W)_{G_{\overline{X}}}. \quad (13)$$

Rule 2 (Action/observation exchange):

$$P(y|do(x), do(z), w) = P(y|do(x), z, w), \text{ if } (Y \perp\!\!\!\perp Z|X, W)_{G_{\overline{X}\underline{Z}}}. \quad (14)$$

Rule 3 (Insertion/deletion of actions):

$$P(y|do(x), do(z), w) = P(y|do(x), w), \text{ if } (Y \perp\!\!\!\perp Z|X, W)_{G_{\overline{XZ(W)}}}. \quad (15)$$

Next, we present the derivation process for $P(Y|do(X))$ in the main text. Specifically, based on the intervention on X in the Structural Causal Model as depicted in Fig. 2 and the formula for conditional probability, we can deduce:

$$\begin{aligned} P(Y|do(X)) &= \mathbb{E}_{\substack{P(Z) \\ P(C)}} P(Y|do(X), Z, C) \\ &= \mathbb{E}_{\substack{P(Z) \\ P(C)}} P(Y|do(X), Z, C) P(Z|do(X), C) P(C|do(X)) \\ &= \mathbb{E}_{\substack{P(Z) \\ P(C)}} P(Y|X, Z, C) P(Z|X, C) P(C|X). \end{aligned} \quad (16)$$

In the above process, given that $Y \perp\!\!\!\perp X|Z, C$ and $Z \perp\!\!\!\perp X|C$ in $G_{\underline{X}}$, according to the *Action/observation exchange* Rule, it can be derived that $P(Y|do(X), Z, C) = P(Y|X, Z, C)$ and $P(Z|do(X), C) = P(Z|X, C)$. Similarly, with $C \perp\!\!\!\perp X$ in $G_{\overline{X}}$, by applying the *Insertion/Deletion of Actions* Rule, it can be deduced that $P(C|do(X)) = P(C|X)$.

A.2.2 DERIVATIONS OF THE OBJECTIVE FUNCTION IN SECTION 3

Eq. 4 presents our optimization objective, with the detailed derivation of the second term, the KL divergence, as follows:

$$\begin{aligned} &\sum_{i=1}^K \mathbb{E}_{\substack{P(E_i(X)) \\ Q_i(C|X)}} \text{KL}[Q_i(C|X) \|\hat{P}(C)] \\ &= \sum_{i=1}^K \mathbb{E}_{P(E_i(X))} \text{KL}[Q_i(C|X) \|\alpha_{K-1}(\dots \alpha_2(\alpha_1\hat{P}_1(C) + (1 - \alpha_1)\hat{P}_2(C)) \\ &\quad + (1 - \alpha_2)\hat{P}_3(C) + \dots) + (1 - \alpha_{K-1})\hat{P}_K(C)] \\ &= \sum_{i=1}^K \mathbb{E}_{P(E_i(X))} \text{KL}[Q_i(C|X) \|\prod_{j=1}^{K-1} \alpha_j \hat{P}_1(C) + \prod_{j=2}^{K-1} \alpha_j (1 - \alpha_1) \hat{P}_2(C) + \dots + (1 - \alpha_{K-1}) \hat{P}_K(C)]. \end{aligned} \quad (17)$$

For the convenience of calculation, We define $M_{\leq K-1}(C) = \prod_{j=1}^{K-1} \alpha_j \hat{P}_1(C) + \prod_{j=2}^{K-1} \alpha_j (1 - \alpha_1) \hat{P}_2(C) + \dots + \alpha_{K-1} (1 - \alpha_{K-2}) \hat{P}_{K-1}(C)$. Inspired by Egorov et al. (2021), Eq.17 can be

simplified as follows according to the calculation formula of limit:

$$\begin{aligned}
(17) &= \sum_{i=1}^K \mathbb{E}_{P(E_i(X))} \text{KL}[Q_i(C|X) \| (\alpha_{K-1} M_{\leq K-1}(C) + (1 - \alpha_{K-1}) \hat{P}_K(C))] \\
&= \sum_{i=1}^K \mathbb{E}_{P(E_i(X))} \mathbb{E}_{Q_i(C|X)} \left[\log \frac{Q_i(C|X)}{\alpha_{K-1} M_{\leq K-1}(C) + (1 - \alpha_{K-1}) \hat{P}_K(C)} \right] \\
&= - \sum_{i=1}^K \mathbb{E}_{P(E_i(X))} \mathbb{E}_{Q_i(C|X)} \left[\alpha_{K-1} \log \frac{M_{\leq K-1}(C)}{Q_i(C|X)} + \log \left(1 + \frac{(1 - \alpha_{K-1}) \hat{P}_K(C)}{\alpha_{K-1} M_{\leq K-1}(C)} \right) \right] \\
&= - \sum_{i=1}^K \mathbb{E}_{P(E_i(X))} \mathbb{E}_{Q_i(C|X)} \left[\log \frac{M_{\leq K-1}(C)}{Q_i(C|X)} + \log \left(\alpha_{K-1} + \frac{(1 - \alpha_{K-1}) \hat{P}_K(C)}{M_{\leq K-1}(C)} \right) \right] \\
&= - \sum_{i=1}^K \mathbb{E}_{P(E_i(X))} \mathbb{E}_{Q_i(C|X)} \left[\log \frac{M_{\leq K-1}(C)}{Q_i(C|X)} + \log((1 - \alpha_{K-1}) \left(\frac{\hat{P}_K(C)}{M_{\leq K-1}(C)} - 1 \right) + 1) \right] \\
&\approx \sum_{i=1}^K \mathbb{E}_{P(E_i(X))} [\text{KL}[Q_i(C|X) \| M_{\leq K-1}(C)] - (1 - \alpha_{K-1}) \underbrace{(\frac{\hat{P}_K(C)}{M_{\leq K-1}(C)} - 1)}_{\approx 0} + o(\alpha_{K-1})].
\end{aligned} \tag{18}$$

The derivation of the penultimate line utilizes an approximation by taking the limit value.

A.3 RELATED WORK

A.3.1 LEARNING-BASED TRAJECTORY PREDICTION

End-to-end trajectory prediction using deep learning has become the mainstream because it can account for multiple factors contributing to prediction accuracy. The most common approach involves sequential networks, which take the past path points of multiple agents across several frames, along with various attributes, to predict future movements. These networks include Recurrent Neural Networks (RNN) (Zyner et al., 2018; 2017), Long Short-Term Memory (LSTM) models (Alahi et al., 2016; Salzman et al., 2020; Xin et al., 2018), and Graph Convolutional Networks (GCN) (Shi et al., 2021; Li et al., 2019). These architectures extract attributes like speed, direction, road characteristics, and interactions, yielding high-dimensional vector representations or feature distributions in latent space. Decoders or sampling methods then use these representations to predict future trajectories.

Attention-based methods extract relational patterns in both time and space (Wu et al., 2021; Kim et al., 2020; Messaoud et al., 2020), with attention mechanisms controlling the flow of critical information. Transformer-based architectures are prominent in this category. For instance, Liu et al. (Liu et al., 2021) propose a multi-modal architecture with stacked transformers to capture features from trajectories, road data, and social interactions. Similarly, Zhao et al. (Zhao et al., 2021) utilize a transformer with residual layers and pooling operations to integrate geographical data for learning interactions. The Spatio-Temporal Transformer Networks (S2TNet) (Chen et al., 2021b) use a spatio-temporal transformer for interactions and a temporal transformer for sequences. Generative Adversarial Networks (GANs) (Gupta et al., 2018) also capture the data distribution, generating diverse and plausible trajectory predictions. Alternatively, large language models have been applied to generate trajectories based on semantics (Lan et al., 2024; Peng et al., 2024).

Recent works further integrate map or scene information for more comprehensive prediction. CNNs have been used to extract features from Bird’s Eye View (BEV) representations (Mangalam et al., 2021; Chou et al., 2020), while context rasterization techniques address Vulnerable Road User (VRU) trajectory prediction (Cui et al., 2019; Djuric et al., 2020). Additionally, some works (Gu et al., 2022; Li et al., 2023; Bae et al., 2024; Li et al., 2024; Xu & Fu, 2024) that integrate emerging research methods such as diffusion models have been proposed, which can effectively enhance the performance of trajectory prediction models in challenging situations. This paper proposes a general training method for end-to-end trajectory prediction models that enhances generalization performance with plug-and-play flexibility.

A.3.2 CAUSAL INFERENCE

The core objective of causal inference is to identify critical factors influencing outcomes. Structural causal models are commonly used for modeling, with principles like backdoor adjustment employed to intervene in causal relationships. In deep learning applications, the focus is often on analyzing confounding factors to mitigate the impact of perturbations on model performance, as demonstrated by methods in Johansson et al. (2016) and Wu et al. (2023). In recent years, causal intervention methods have been applied in trajectory prediction to enhance performance across domains. For example, Chen et al. (2021a) employs counterfactual interventions, such as using a zero vector, to reduce bias between training and deployment environments. Liu et al. (2022) highlights that target trajectory Y is often correlated with observation noise and agent densities, proposing a gradient norm penalty over empirical risk to mitigate environmental effects (Bagi et al., 2023). Additionally, Pourkeshavarz et al. (2024) leverages disentangled representation learning to isolate invariant and variant features, minimizing the latter's influence on trajectory prediction. Our work addresses catastrophic forgetting in domain-shift scenarios by learning the prior distribution of trajectory representations across contexts to identify scenario-specific confounding factors. This approach enables intuitive manipulation through controlled interventions and adapts seamlessly to continuously evolving conditions.

A.3.3 CONTINUAL LEARNING

Continual learning aims to maintain strong model performance as new domain samples are introduced, addressing catastrophic forgetting by balancing the retention of previous knowledge with the acquisition of new tasks. The most common approach uses constraint-based methods on the loss function, ensuring past learning directions are considered during gradient updates while adapting to new samples (Kirkpatrick et al., 2017; Lopez-Paz & Ranzato, 2017). However, these methods often lack interpretability and accuracy, especially when current tasks differ significantly from prior ones. An alternative approach is the rehearsal method (Wang et al., 2019; Riemer et al., 2018), which reinforces past knowledge by replaying previous samples during training. Memory-based mechanisms also preserve past information by storing samples as tasks accumulate (Chaudhry et al., 2019). Despite some work on expanding domain generalization through meta-learning (Ivanovic et al., 2023), limited research addresses continual learning in trajectory prediction, particularly in mitigating forgetting and adapting to changing scenarios. Our method integrates causal inference with a memory-based continual learning framework to develop a generalized trajectory prediction model.

A.4 MORE ABOUT EXPERIMENTS

A.4.1 ADDITIONAL RESULTS IN CONTINUAL LEARNING SETTING

Table2 and Table3 present the training results of continual learning on the Synthetic Dataset and SDD Dataset, corresponding to the results described in Section 5.3.

A.4.2 TRAINING PROCESS CURVE

Fig.4 displays the ADE's statistics for each model across various tasks during the training process, corresponding to the results described in Section 5.4.

A.4.3 METRICS FOR EACH TASK SEPARATELY

In Fig.5-7, we provide a detailed representation of performance metrics (ADE) across various tasks, all within the same continual learning framework outlined in Sec. 5.3. The severity of catastrophic forgetting differs across tasks for the ETH-UCY dataset, as depicted in Fig.5. Taking the "univ" task as an example, while most models maintain reasonable performance, the original STGAT shows the weakest results. In contrast, C²INet performs significantly better due to its effective intervention in counteracting environmental influences on trajectory representations. When training progresses to the "eth" task, the task complexity increases significantly, resulting in an ADE of over 1.3 for all models. Additionally, the performance on the previously completed "univ" task deteriorates, with its ADE rising above 0.7. C²INet exhibits the least amount of forgetting, as it effectively retains knowledge from past tasks. The fifth plot illustrates the effects of forgetting across tasks after completing the entire training cycle. It is evident that while the "hotel" task demonstrates post-training solid performance, it has a detrimental impact on other tasks, likely due to gradient

Table 2: The table presents the model’s average performance across all previously encountered tasks on Synthetic Dataset. The first column lists the newly added tasks in the continual learning setup, with the preceding colon indicating the average trajectory prediction results for all tasks encountered up to that point. All results are averaged over five runs, with the best outcomes highlighted in bold and the second-best results underlined. Color blocks indicate the ranking of different backbones for comparison.

TASK	:0.1	:0.2	:0.3	:0.4	:0.5	:0.6
STGAT	0.15/0.20	0.17/0.24	0.19/0.25	0.21/0.31	0.25/0.33	0.28/0.36
STGCNN	0.65/1.16	0.50/0.92	0.39/0.75	0.36/0.62	0.39/0.67	0.42/0.65
PECNet	0.10/0.15	0.14/0.17	0.17/0.20	0.18/0.23	0.23/0.25	0.24/0.26
COUNTERFACTUAL	0.07/0.13	0.06/0.09	0.08/0.12	0.10/0.12	0.15/0.16	0.18/0.22
INVARIANT	0.07/0.15	0.11/0.16	0.15/0.18	0.21/0.19	0.28/0.24	0.37/0.38
ADAPTIVE	0.15/0.21	0.17/0.23	0.19/0.29	0.18/0.27	0.23/0.31	0.29/0.36
EWC	0.09/0.14	0.08/0.15	0.11/0.19	0.15/0.25	0.17/0.30	0.21/0.35
MoG	0.11/0.17	0.09/0.16	0.13/0.20	0.15/0.26	0.19/0.34	0.22/0.37
Coresets	0.09/0.14	0.08/0.14	0.10/0.21	0.14/0.23	0.16/0.28	0.20/0.33
GCRL+STGAT	0.04/0.08	0.07/0.12	0.12/0.16	0.17/0.23	0.20/0.29	0.23/0.33
GCRL+STGCNN	0.24/0.43	0.30/0.57	0.28/0.52	0.31/0.49	0.35/0.59	0.40/0.66
C ² INet-online+STGAT	0.09/0.16	0.07/0.13	<u>0.10/0.19</u>	<u>0.14/0.24</u>	0.17/0.29	0.20/0.34
C ² INet-online+STGCNN	0.36/0.59	0.34/0.54	0.25/0.46	0.26/0.49	0.30/0.55	0.34/0.59
C ² INet-offline+STGAT	0.08/0.13	<u>0.07/0.12</u>	0.11/0.19	0.14/0.25	<u>0.16/0.27</u>	<u>0.18/0.27</u>
C ² INet-offline+STGCNN	0.34/0.55	0.31/0.49	0.27/0.45	0.29/0.51	0.28/0.46	0.31/0.51

Table 3: The table presents the model’s average performance across all previously encountered tasks on SDD Dataset. The first column lists the newly added tasks in the continual learning setup, with the preceding colon indicating the average trajectory prediction results for all tasks encountered up to that point. All results are averaged over five runs, with the best outcomes highlighted in bold and the second-best results underlined. Color blocks indicate the ranking of different backbones for comparison.

TASK	:bookstore	:coupa	:deathcircle	:gates	:hyang	:nexus	:little	:quad
STGAT	75.96/139.81	52.99/98.06	114.63/206.93	93.75/169.37	88.40/163.33	79.88/140.89	78.35/142.09	79.56/145.67
STGCNN	75.90/139.76	52.99/98.14	114.97/207.79	94.59/172.21	91.73/170.78	82.79/153.86	83.84/157.80	81.51/153.27
PECNet	74.41/137.86	51.79/98.23	111.03/202.42	93.12/170.10	88.69/170.23	80.54/148.39	80.19/151.79	82.49/155.83
COUNTERFACTUAL	66.93/127.97	49.84/93.21	106.03/194.52	<u>90.97/165.97</u>	86.50/158.32	75.74/135.20	75.18/137.10	78.06/142.94
INVARIANT	71.41/132.86	52.63/98.05	108.44/197.46	94.71/170.92	90.94/166.05	78.69/149.52	76.89/149.88	80.79/143.42
ADAPTIVE	86.51/162.14	65.21/116.87	121.96/216.23	99.50/182.85	98.32/179.50	90.65/173.45	94.25/176.25	93.32/174.44
EWC	70.62/131.19	54.93/102.42	117.01/212.02	97.34/176.93	96.50/176.75	87.44/160.47	88.90/163.64	88.46/162.84
MoG	71.31/134.52	55.45/104.68	118.21/215.39	98.14/178.96	99.12/181.23	90.12/171.26	90.56/169.21	91.32/169.78
Coresets	71.25/133.82	54.98/102.72	117.02/212.03	97.35/176.96	96.49/176.72	87.47/160.51	88.93/163.67	88.50/162.89
GCRL+STGAT	76.30/140.42	54.87/101.80	117.32/212.43	97.62/177.25	96.58/176.68	87.58/160.47	89.03/163.62	88.63/162.95
GCRL+STGCNN	76.32/140.44	54.87/101.80	117.29/212.35	97.62/177.25	96.57/176.65	87.58/160.47	89.03/163.62	88.63/162.95
C ² INet-online+STGAT	70.87/130.67	51.19/95.70	107.29/196.40	91.56/167.05	86.88/161.24	75.75/138.72	77.21/143.92	78.82/143.22
C ² INet-online+STGCNN	70.98/130.84	51.50/96.06	107.98/196.95	91.53/166.96	86.86/161.12	75.77/140.66	77.28/143.96	78.82/143.93
C ² INet-offline+STGAT	70.43/130.34	<u>50.87/96.04</u>	<u>106.68/197.89</u>	90.32/164.98	85.52/160.31	74.42/140.24	<u>75.90/142.60</u>	77.13/141.69
C ² INet-offline+STGCNN	<u>70.09/130.16</u>	50.99/95.98	107.01/198.07	91.27/166.58	<u>85.92/160.95</u>	74.98/140.64	76.24/142.56	<u>78.01/142.85</u>

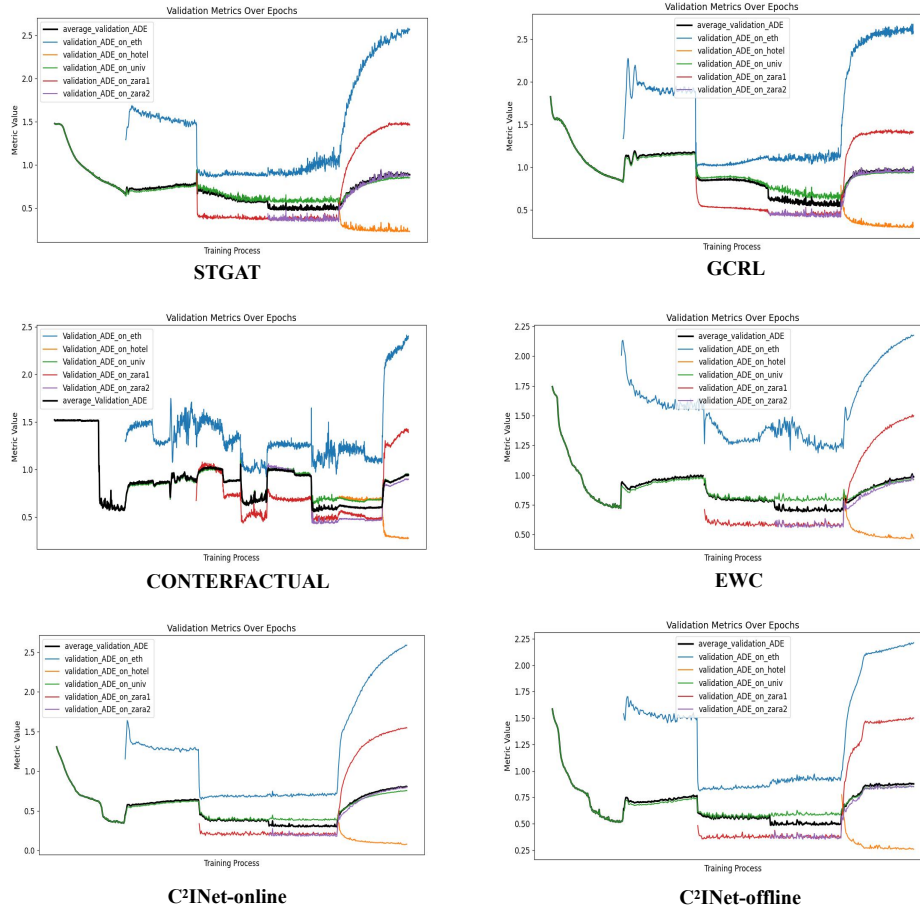


Figure 4: The ADE variations of the validation sets across five task scenarios and their average performance during the training process on the ETH-UCY dataset. The x-axis represents the number of completed training epochs, while the y-axis denotes the corresponding metric values.

optimization disrupting the model’s adaptability to previously learned tasks. Although the ADE curves for COUNTERFACTUAL and INVARIANT are relatively stable, their overall performance remains suboptimal.

Fig.6 showcases the performance of various models on the Synthesis dataset. COUNTERFACTUAL and INVARIANT exhibit relatively balanced performance across tasks "0.1" to "0.6", while baseline models such as STGAT and GCRL experience consistent performance degradation, reflecting their tendency to prioritize newly introduced tasks. In contrast, models designed explicitly for continual learning, such as Coresets, EWC, and the proposed C²INet, demonstrate a steady upward performance trend, emphasizing their ability to retain knowledge from previous tasks. In task "0.1", which contains minimal noise, these models maintain strong memory retention, underscoring their robustness in low-noise environments.

In Fig.7, the performance curves on the SDD dataset remain relatively stable across the models. Continual learning models, including C²INet, EWC, and Coresets, effectively mitigate catastrophic forgetting in tasks such as "coupa". However, their performance deteriorates on more challenging tasks like "deathcircle" due to excessive retention of information from earlier tasks. Ultimately, C²INet-offline emerges as the best-performing model, demonstrating superior training performance across the dataset.

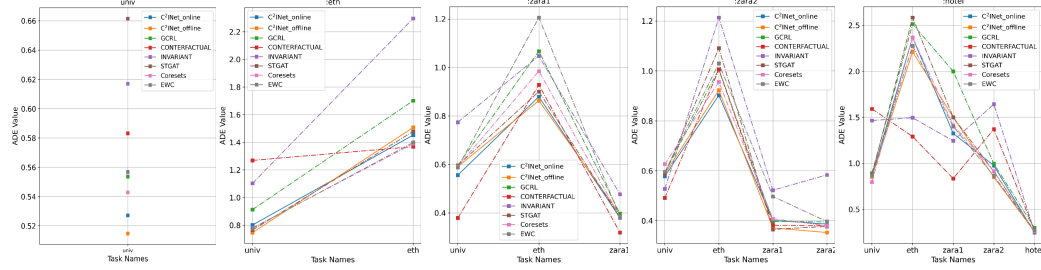


Figure 5: The Average Displacement Error (ADE) on the ETH-UCY dataset for each task, averaged over five runs under continual learning settings. The x-axis represents the sequence of completed tasks, while the y-axis indicates the corresponding metric values.

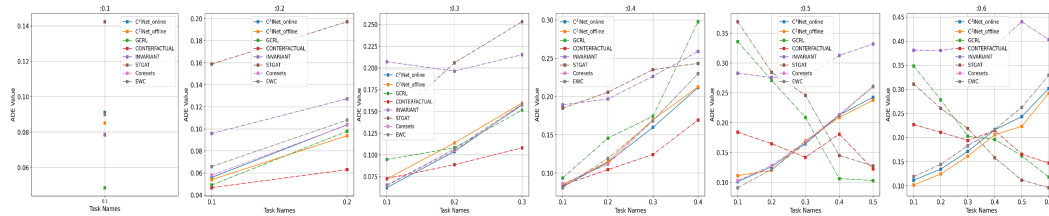


Figure 6: The Average Displacement Error (ADE) on the Synthesis dataset for each task, averaged over five runs under continual learning settings. The x-axis represents the sequence of completed tasks, while the y-axis indicates the corresponding metric values.

A.4.4 QUALITATIVE RESULTS

Fig.8 mainly displays the qualitative analysis presented in the main text Section 5.5.

A.4.5 ABLATION STUDY

In Table 4, we systematically assess the contributions of several key modules to the performance of the C²INet model in a continual learning context. The ablation process begins by isolating the impact of the symmetric KL divergence constraint from Eq.10 (Abbreviated as "Divergence"), followed

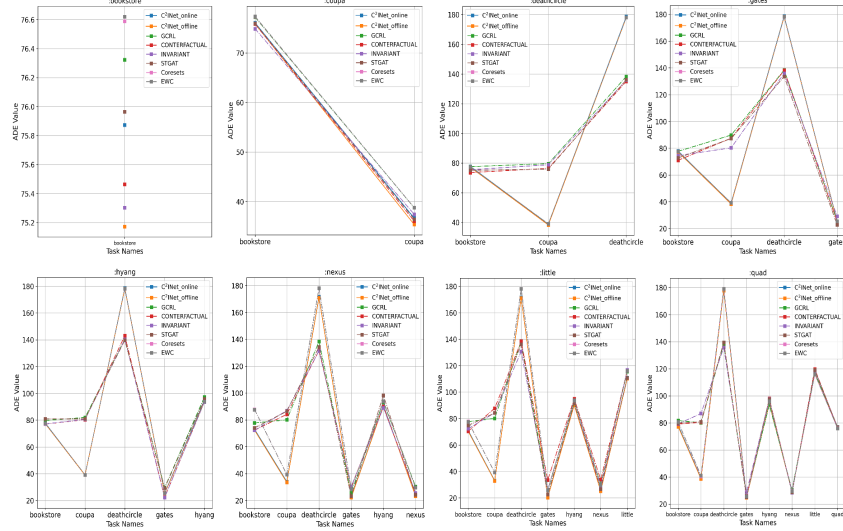


Figure 7: The Average Displacement Error (ADE) on the SDD dataset for each task, averaged over five runs under continual learning settings. The x-axis represents the sequence of completed tasks, while the y-axis indicates the corresponding metric values.

C ² I	Divergence	Weight	univ	:eth	:zara1	:zara2	:hotel
✓	✓	✓	0.820	2.228	1.420	0.827	0.247
✓	✓	✗	0.851(+0.031)	2.372(+0.144)	1.574(+0.154)	0.839(+0.012)	0.256(+0.009)
✓	✗	✓	0.883(+0.063)	2.562(+0.334)	1.663(+0.243)	0.865(+0.038)	0.285(+0.038)
✗	✗	✗	0.913(+0.093)	2.701(+0.473)	1.752(+0.332)	0.898(+0.071)	0.296(+0.049)

Table 4: Results of the ablation study, focusing on the performance of the C²INet model on the ETH-UCY dataset after the removal of certain modules. The first row of the table specifies the ablated modules and the completed training tasks. Check mark indicates the removal of the corresponding module, while cross signifies its inclusion.

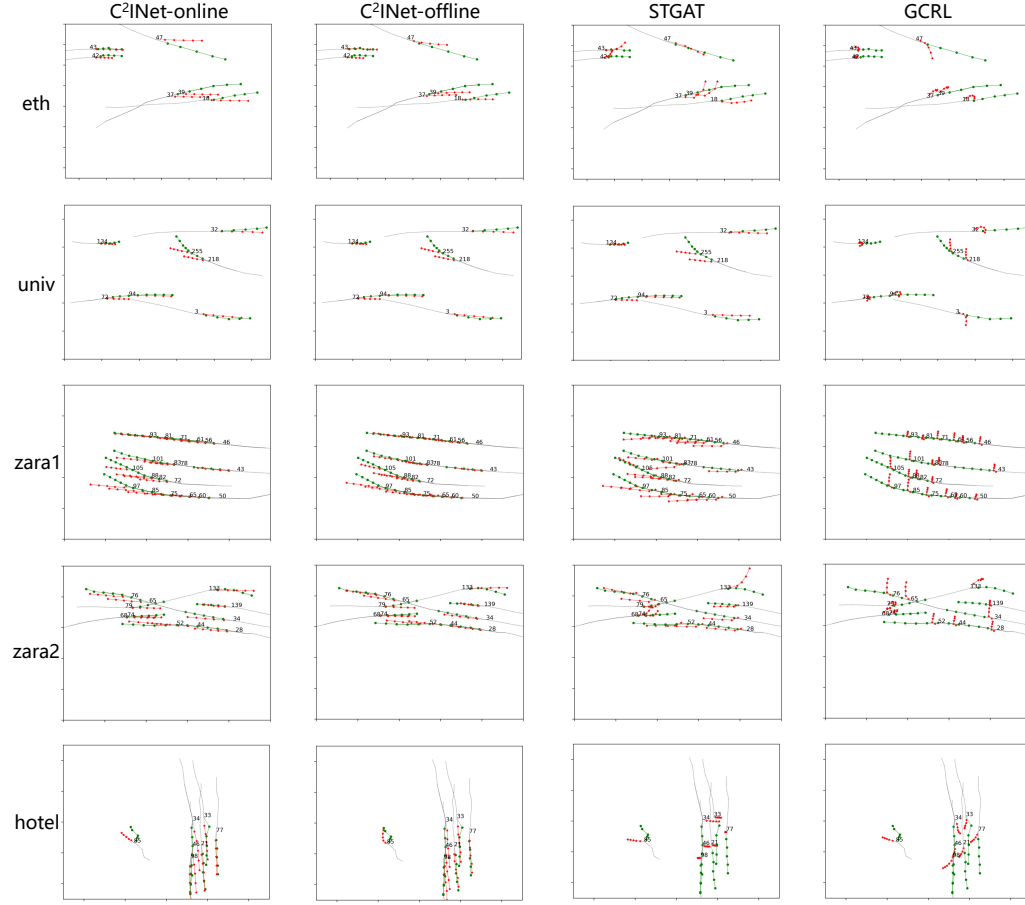


Figure 8: Qualitative results on the ETH-UCY dataset. The gray lines represent observed trajectories, the red points indicate predicted paths, and the green points denote the ground truth.

by the removal of the weight optimization procedure outlined in Eq.9, where only average weight parameters are used (Abbreviated as "Weight"). Lastly, we completely omit the Continual Causal Intervention mechanism to evaluate its significance.

A detailed comparison between the second and third rows of the table demonstrates that when weight optimization is removed and average weights are used, the model’s performance initially declines by approximately 3.7% during the early stages of training. However, the model fills the gaps as training progresses and gradually approaches near-optimal performance. This pattern indicates that the lack of weight optimization is mitigated as the prior queue accumulates more elements, diminishing its overall impact on model performance. Conversely, the evident 5% – 17% performance decline observed after removing the symmetric KL divergence constraint highlights the importance of maintaining diversity among priors. This diversity appears to be crucial in preventing overfitting to specific tasks and ensuring that the model adapts effectively across varying environments.

In the final row, the complete removal of both KL divergence constraints related to confounding factors (from Eq.10), alongside the exclusion of the weight optimization and divergence constraint mechanisms, leads to the most significant performance degradation—up to 21.6%. This decline becomes more pronounced as tasks increase, with catastrophic forgetting contributing heavily to the sharp drop in average performance. These results underscore the Continual Causal Intervention module’s essential role and the associated constraints in sustaining model robustness during continual learning.

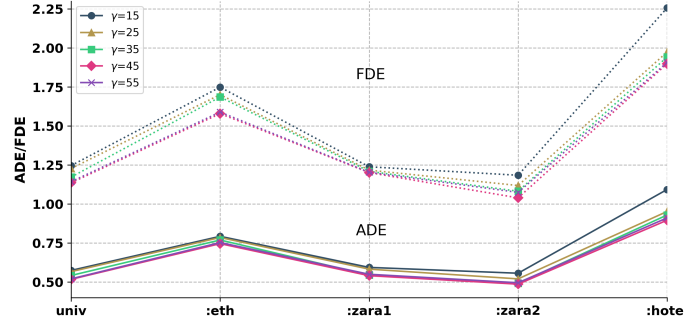


Figure 9: The prediction results of C²INet-online on the ETH-UCY dataset with varying maximum prior queue capacities γ are shown. The model regulates the number of priors through a pruning mechanism. Dashed lines represent FDE, while solid lines indicate ADE.

A.4.6 ANALYSIS OF QUEUE CAPACITY

Fig.9 illustrates the training performance of C²INet-online across varying maximum prior queue capacities γ . When the number of generated components surpasses γ after a task, the proposed pruning mechanism is activated to control the queue size. The experimental results demonstrate that the model achieves optimal performance with $\gamma = 45$, whereas $\gamma = 15$ leads to the poorest outcomes. This finding indicates that selecting an appropriately sized prior queue is crucial for maximizing model efficiency and effectiveness. Both huge and overly constrained prior queues can introduce issues—such as overfitting or reduced learning capacity—highlighting the essential function of the pruning mechanism in maintaining a balance that prevents performance degradation. These results emphasize the importance of reasonably controlled priors to enhance model robustness and adaptability in continual learning environments.

A.4.7 ANALYSIS OF TASK SEQUENCE

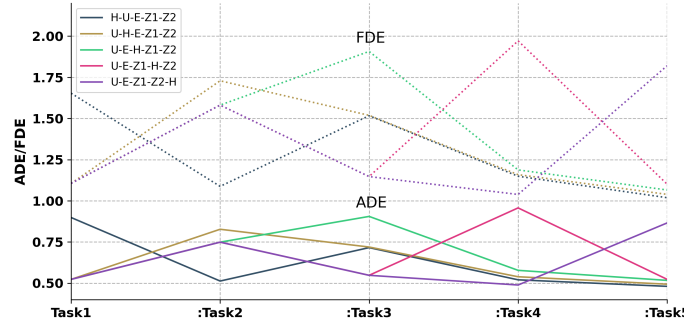


Figure 10: The prediction results of C²INet-online on the ETH-UCY dataset with different task loading sequences are presented. The task mappings are as follows: U - univ, E - eth, Z1 - zara1, Z2 - zara2, H - hotel. Dashed lines represent FDE, while solid lines indicate ADE.

Fig.10 presents the ADE performance of the model across different training task sequences. The results reveal that changes in the order of task training result in subtle but measurable variations in the model’s final average performance, with a gap of approximately 7.6% between the best and second-worst outcomes ("H-U-E-Z1-Z2" and "U-E-Z1-H-Z2"). A key observation is that the "hotel" task, which introduces substantial noise, has a noticeable impact on the overall performance across all tasks. Training the "hotel" task earlier in the sequence leads to improved final performance, suggesting that the long-term forgetting effect mitigates the adverse bias caused by the noise. This

1296 finding underscores the impact of task order in continual learning and highlights how forgetting
1297 mechanisms can mitigate confounding noise within tasks.
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349