Supplementary Information: Membership Inference Attacks Against Temporally Correlated Data in Deep Reinforcement Learning

In Section 1, we provide additional information regarding the attack network architectures we use to design attack classifier in the *individual* and *collective* modes. In Section 2, we present the detailed results on the performance of the two attack classifiers in three standard continuous control locomotion MuJoCo tasks Hopper-v2 (Table 1), HalfCheetah-v2 (Table 2), and Ant-v2 (Table 3).

1 Network Architectures

Individual-Mode Attack Classifier Architecture- In the individual mode, we use Temporal Convolutional Network (TCN) architecture [1], which takes temporally correlated data as input and uses a hierarchy of dilated (causal) convolutional layers to capture the inherent temporal correlation in the input data. In this study, since the input data to the classifier in the individual mode is a pair of temporally correlated tensors (*i.e.* $\mathbb{R}^{2d^{\mathcal{A}}\times T}$), the size of *input channel* in TCN architecture represents twice the dimension of the action space. In the design of the TCN architecture, we use a two-layer network and set the number of hidden layers to 600. We set the kernel size to 3 and use dropout with a threshold of 0.45. For network training, we employ Adam optimizer [4] with the initial learning rate set at 0.0003 and gradient clipping at 0.35 to avoid exploding gradients. We change the initial learning rate during the training phase using the learning rate scheduler to avoid local minimum. In this regard, we use a learning rate scheduler at 100 and 200 epochs with gamma rate 0.1 to decay the learning rate with time. The batch size is set to 16 during training, and the network is trained for 300 epochs.

Collective-Mode Attack Classifier Architecture- In the collective mode, we use deep Residual Networks (ResNets) as the choice of attack classifier. In particular, we employ ResNet-18 architecture, where in addition to an input channel, there is a width dimension, which we use to input a collection of trajectories as a batch. Our input is in the form of a three-dimensional tensor (*e.g.* $\mathbb{R}^{2d^{\mathcal{A}}\times T\times m}$), where m denotes the number of trajectories in each batch, similar to the data structure used in image classification problems [2, 5, 3]. Moreover, we use Adam optimizer with learning rate 0.0008 and

weight decay 1.0, and we set the clipping size to 0.35. Finally, we use batch size 256 and train the network for 300 epochs.

2 Attack Performance

In the following tables, we show the results obtained for the attack performance on the deep RL algorithm in the three MuJoCo environments in terms of five performance metrics *Accuracy*, *Precision*, *recall*, *F1 score*, and *Matthews correlation coefficient*.

Table 1: The performance of the attack classifiers in Hopper-v2 for different maximum trajectory lengths $T_{\max}$ in terms of accuracy (ACC), precision (PR), recall (RE), F1 score (F1), and Matthews correlation coefficient (MCC). The values in parentheses show the results for the collective attack mode.

Hopper-v2

T_{max}	ACC	PR	RE	F1	MCC
10	0.67 ± 0.11 (0.72 ± 0.11)	0.73 ± 0.10 (1 ± 0.00)	0.66 ± 0.10 (0.65 ± 0.15)	0.70 ± 0.1 (0.75 ± 0.12)	0.34 ± 0.21 (0.47 ± 0.22)
50	0.76 ± 0.03 (0.89 ± 0.02)	0.79 ± 0.06 (0.94 ± 0.01)	0.73 ± 0.04 (0.83 ± 0.04)	0.75 ± 0.03 (0.88 ± 0.02)	0.52 ± 0.07 (0.78 ± 0.03)
100	0.82 ± 0.06 (0.96 ± 0.04)	0.84 ± 0.08 (0.98 ± 0.02)	0.83 ± 0.03 (0.93 ± 0.07)	0.83 ± 0.06 (0.95 ± 0.05)	0.66 ± 0.12 (0.92 ± 0.08)
200	0.83 ± 0.08 (0.95 ± 0.03)	0.83 ± 0.09 (0.96 ± 0.02)	0.87 ± 0.05 (0.94 ± 0.05)	0.85 ± 0.07 (0.95 ± 0.03)	0.67 ± 0.16 (0.90 ± 0.05)
500	0.84 ± 0.11 (0.97 ± 0.03)	0.92 ± 0.03 (0.97 ± 0.03)	0.75 ± 0.23 (0.98 ± 0.02)	0.78 ± 0.17 (0.97 ± 0.03)	0.71 ± 0.18 (0.94 ± 0.06)

Table 2: The MIA performance results for individual and collective modes in HalfCheetah-v2 for different maximum trajectory lengths $T_{\max}$ in terms of accuracy (ACC), precision (PR), recall (RE), F1 score (F1), and Matthews correlation coefficient (MCC). The values in parentheses present the results for the collective attack mode.

HalfCheetah-v2

T_{max}	ACC	PR	RE	F1	MCC
10	0.55 ± 0.03 (0.65 ± 0.03)	0.58 ± 0.05 (0.61 ± 0.01)	0.34 ± 0.06 (0.80 ± 0.10)	0.43 ± 0.06 (0.69 ± 0.04)	0.11 ± 0.07 (0.32 ± 0.07)
30	0.68 ± 0.11 (0.77 ± 0.13)	0.64 ± 0.09 (0.82 ± 0.02)	0.85 ± 0.08 (0.67 ± 0.32)	0.73 ± 0.08 (0.70 ± 0.21)	0.39 ± 0.12 (0.57 ± 0.14)
50	0.70 ± 0.02 (0.84 ± 0.07)	0.71 ± 0.03 (0.84 ± 0.08)	0.66 ± 0.09 (0.84 ± 0.10)	0.67 ± 0.05 (0.84 ± 0.08)	0.40 ± 0.04 (0.64 ± 0.10)
100	0.79 ± 0.06 (0.86 ± 0.04)	0.80 ± 0.06 (0.89 ±0.08)	0.78 ± 0.13 (0.84 ± 0.02)	0.78 ± 0.08 (0.86 ± 0.04)	0.60 ± 0.11 (0.73 ± 0.09)
200	0.81 ± 0.01 (0.96 ± 0.02)	0.79 ± 0.03 (0.98 ± 0.01)	0.86 ± 0.04 (0.93 ± 0.05)	0.82 ± 0.00 (0.95 ± 0.02)	0.62 ± 0.01 (0.91 ± 0.04)

Table 3: The performance of the attack classifiers in Ant-v2 for different maximum trajectory lengths $T_{\max}$ in terms of accuracy (ACC), precision (PR), recall (RE), F1 score (F1), and Matthews correlation coefficient (MCC). The values in parentheses show the results for the collective attack mode.

Ant-v2

T_{max}	ACC	PR	RE	F1	MCC
10	0.62 ± 0.07 (0.88 ± 0.08)	0.65 ± 0.07 (0.88 ± 0.07)	0.70 ± 0.14 (0.86 ± 0.09)	0.64 ± 0.07 (0.87 ± 0.08)	0.25 ± 0.13 (0.75 ± 0.15)
50	0.69 ± 0.03 (0.94 ± 0.007)	0.67 ± 0.06 (0.94 ± 0.005)	0.71 ± 0.07 (0.93 ± 0.02)	0.69 ± 0.06 (0.93 ± 0.04)	0.39 ± 0.07 (0.95 ± 0.01)
100	0.71 ± 0.03 (0.97 ± 0.02)	0.68 ± 0.03 (0.96 ± 0.02)	0.81 ± 0.04 (0.99 ± 0.01)	0.74 ± 0.03 (0.97 ± 0.02)	0.44 ± 0.05 (0.94 ± 0.04)
200	0.79 ± 0.03 (0.98 ± 0.02)	0.79 ± 0.08 (0.97 ± 0.01)	0.73 ± 0.06 (0.98 ± 0.01)	0.75 ± 0.004 (0.97 ± 0.02)	0.53 ± 0.05 (0.96 ± 0.04)

References

[1] S. Bai, J. Z. Kolter, and V. Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

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

[3] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[4] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[5] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.