Efficient learning of nonlinear prediction models with time-series privileged information

Bastian Jung
Chalmers University of Technology
mail@bastianjung.com

Fredrik D. Johansson
Chalmers University of Technology
fredrik.johansson@chalmers.se

Abstract

In domains where sample sizes are limited, efficient learning algorithms are critical. Learning using privileged information (LuPI) offers increased sample efficiency by allowing prediction models access to types of information at training time which is unavailable when the models are used. In recent work, it was shown that for prediction in linear-Gaussian dynamical systems, a LuPI learner with access to intermediate time series data is never worse and often better in expectation than any unbiased classical learner. We provide new insights into this analysis and generalize it to nonlinear prediction tasks in latent dynamical systems, extending theoretical guarantees to the case where the map connecting latent variables and observations is known up to a linear transform. In addition, we propose algorithms based on random features and representation learning for the case when this map is unknown. A suite of empirical results confirm theoretical findings and show the potential of using privileged time-series information in nonlinear prediction.

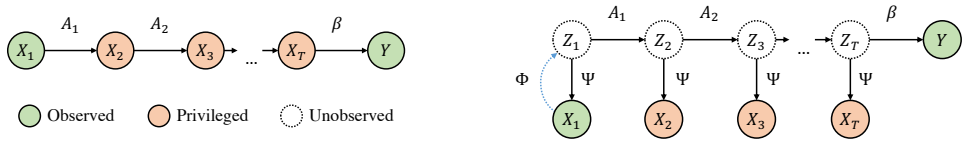
1 Introduction

In data-poor domains, making the best use of all available information is central to efficient and effective machine learning. Despite this, supervised learning is often applied in such a way that informative data is ignored. A good example of this is learning to predict the condition of a patient at a set follow-up time based on information of the first medical examination. Classical supervised learning makes use only of the initial data to predict the disease status at the follow-up, although in many cases data about medications, laboratory tests or vital signs are routinely collected about patients also at intermediate time points. This information is *privileged* (Vapnik and Vashist, 2009), as it is unavailable at the time of prediction, but can be used for training a model.

Learning using privileged information (LuPI) (Vapnik and Vashist, 2009), *generalized distillation* (Lopez-Paz et al., 2016) and *multi-view learning* (Rosenberg and Bartlett, 2007) have been proposed to increase the sample efficiency by leveraging privileged information in learning. Theoretical results guarantee improved learning rates (Pechyony and Vapnik, 2010; Vapnik et al., 2015) or tighter generalization bounds (Wang, 2019) for large sample sizes under appropriate assumptions. However, privileged information is not always beneficial; it must be related to the task at hand (Jonschkowski et al., 2015). Previous works do not identify such settings. Moreover, existing analyses do not state when learning with privileged information is preferable to classical learning for problems with small sample sizes—which is where efficiency is needed the most.

Karlsson et al. (2022) studied LuPI in the context of predicting an outcome observed at the end of a time series based on variables collected at the first time step. They showed that making use of data from intermediate time steps in particular settings always leads to lower or equal prediction risk—for any sample size—compared to the best unbiased model which does not make use of this privileged information. However, their method called *learning using privileged time series* (LuPTS) was limited to settings where the outcome function, and estimators of it, are linear functions of baseline features.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).



(a) Linear dynamical system, fully observed

(b) Linear latent dynamics, nonlinear observations

Figure 1: Comparison between the linear data generating process and the latent nonlinear generalization in this work. Ψ indicates the observation function of the latent system, with Φ its left inverse.

Moreover, their analysis did not study how variance reduction behaves as a function of increased input dimension. Hayashi et al. (2019) also learned from privileged intermediate time points but their study was limited to empirical results for classification using generalized distillation.

There is an abundance of real-world prediction tasks with fixed follow-up times which can be framed as having access to privileged time-series information. Examples include predicting 90-day patient mortality (Karhade et al., 2019) or patient readmission in healthcare (Mortazavi et al., 2016), the churn of users of an online service over a fixed period (Huang et al., 2012) or yearly crop yields from satellite imagery of farms (You et al., 2017). In these cases, privileged time-series information comprises daily patient vitals, intermediate user interactions and daily satellite imagery respectively. We are motivated by finding sample-efficient learning algorithms that utilize privileged time-series information to improve upon classical learning alternatives in such settings.

Contributions. We extend the LuPTS framework to nonlinear models and prediction tasks in latent dynamical systems (Section 3). In this setting, we prove that learning with privileged information leads to lower risk when the nonlinear map connecting latent variables and observations is known up to a linear transform (Section 3.1). In doing so, we also find that when the representation dimension grows larger than the number of samples, the benefit of privileged information vanishes. We show that a privileged learner using random feature maps can learn optimal models consistently, even when the relationship between latent and observed variables is unknown, and give a practical algorithm based on this idea (Section 3.2). However, random feature methods may suffer from bias in small samples. As a remedy, we propose several representation learning algorithms aimed at trading off bias and variance (Section 3.3). In experiments, we find that privileged time-series learners with either random features or representation learning reduce variance and improve latent state recovery in small-sample settings (Section 4) on both synthetic and real-world regression tasks.

2 Prediction and privileged information in nonlinear time series

We aim to predict outcomes Y in $\mathcal{Y} \subseteq \mathbb{R}^q$ based on covariates X_1 in $\mathcal{X} \subseteq \mathbb{R}^k$. X_1 are observed at a baseline time point $t = 1$, starting a time series on the form $X_1, X_2, \ldots X_T, Y$. Outcomes are assumed to be compositions of a representation Φ, a linear map θ and Gaussian noise ϵ,

$$Y = h(X_1) + \epsilon, \quad \text{where} \quad h(x_1) \coloneqq \theta^\top \Phi(x_1), \quad \text{and} \quad \epsilon \sim \mathcal{N}(0, \tilde{\sigma}_Y^2)\,. \tag{1}$$

In addition to observations of X_1 and Y, *privileged information* (PI) is available in the form of samples of random variables, $X_2, ..., X_T$, from intermediate time points between X_1 and Y, all taking values in $\mathcal{X}$. Two data generating processes (DGPs) with this structure are illustrated in Figure 1. Unlike baseline variables X_1, privileged information is observed *only at training time, not at test time*. Therefore, it can only benefit learning, and not inference. Data sets of $m \in \mathbb{N}_+$ training samples, $D \coloneqq \{(x_{i,1}, x_{i,2}, ..., x_{i,T}, y_i)\}_{i=1}^m$, are drawn independently and identically distributed from a fixed, unknown distribution p over all random variables in our system. We let $\mathbf{X}_t = [x_{1,t}, ..., x_{m,t}]^\top$ denote the data matrix of features observed at time $t = 1, ..., T$ and $\mathbf{Y} = [y_1, ..., y_m]^\top$ the vector of all outcomes observed in D. A learning algorithm $\mathscr{A} : \mathcal{D} \to \mathcal{H}$ maps data sets $D \in \mathcal{D}$ to hypotheses $\hat{h} \in \mathcal{H}$. An *efficient* algorithm $\mathscr{A}$ has small expected risk $\overline{R}_p(\mathscr{A})$ with respect to a loss $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ over training sets of fixed size m drawn from p,

$$\overline{R}_p(\mathscr{A}) \coloneqq \mathbb{E}_D[R_p(\hat{h})] \quad \text{where} \quad \hat{h} = \mathscr{A}(D) \quad \text{and} \quad R_p(\hat{h}) \coloneqq \mathbb{E}_p[L(\hat{h}(X_1), Y)]\,. \tag{2}$$

We study the regression setting with L the squared error, $L(y, y') = \|y - y'\|_2^2$. In analysis, we focus on univariate outcomes, $q = 1$, but all results extend to multivariate outcomes, $q > 1$.

Algorithm 1: Generalized LuPTS

Input: Data $D = (\{\mathbf{X}_t\}, \mathbf{Y})$; Repr. $\hat{\Phi}$ or kernel κ

if using a fixed representation $\hat{\Phi}$ **then**

$\hat{\mathbf{Z}}_t = [\hat{\Phi}(x_{1,1}), ..., \hat{\Phi}(x_{m,1})]^\top$ for $t = 1, ..., T$

$\hat{\theta}_{\text{P}} := \Big[\prod_{t=1}^{T-1} \underbrace{(\hat{\mathbf{Z}}_t^\top \hat{\mathbf{Z}}_t)^\dagger \hat{\mathbf{Z}}_t^\top \hat{\mathbf{Z}}_{t+1}}_{\hat{A}_t} \Big] \underbrace{(\hat{\mathbf{Z}}_T^\top \hat{\mathbf{Z}}_T)^\dagger \hat{\mathbf{Z}}_T^\top \mathbf{Y}}_{\hat{\beta}}$

$\hat{h}_{\text{P}}(\cdot) := \hat{\theta}_{\text{P}}^\top \hat{\Phi}(\cdot)$

else if using kernel κ **then**

$\hat{\mathbf{K}}_t = \kappa(\mathbf{X}_t, \mathbf{X}_t)$ for $t = 1, ..., T$

$\boldsymbol{\alpha} := \hat{\mathbf{K}}_1^\dagger \left[\prod_{t=2}^{T} \hat{\mathbf{K}}_t \hat{\mathbf{K}}_t^\dagger \right] \mathbf{Y}$

$\hat{h}_{\text{P}}(\cdot) := \sum_{i=1}^{n} \alpha_i \kappa(x_{i,1}, \cdot)$

return $\hat{h}_{\text{P}}$

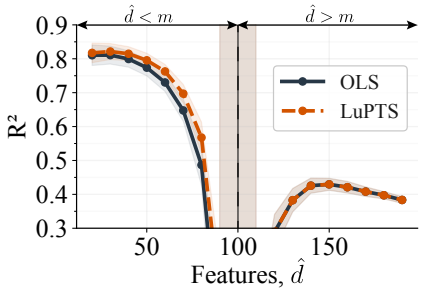


Figure 2: Two regimes of generalized LuPTS with varying feature dimension $\hat{d}$, and sample size $m = 100$. When $m > \hat{d}$, generalized LuPTS is provably never worse than OLS. When $m \leq \hat{d}$, they are equivalent. See Appendix F for details.

Our main goal is to compare *privileged learners* $\mathscr{A}_{\text{P}}$, which make use of privileged information, to *classical learners* $\mathscr{A}_{\text{C}}$ which learn without PI. We seek to identify conditions and algorithms for which privileged learning is more efficient, i.e., it leads to lower risk in expectation, $\overline{R}(\mathscr{A}_{\text{P}}) \leq \overline{R}(\mathscr{A}_{\text{C}})$ for the same number of training samples. We describe such a setting next.

2.1 Privileged information in latent dynamical systems

We study tasks where data are produced by a latent dynamical process. Observations X_t are generated from unobserved latent states Z_t through a nonlinear transformation Ψ, see Figure 1b. This means that the target of learning, $h(X_1) = \mathbb{E}[Y \mid X_1]$, is nonlinear in X_1. Latent dynamical systems like these have proven successful at modelling a variety of phenomena as for example fluid dynamics in physics (Lee and Carlberg, 2019) and human brain activity in neuroscience (Kao et al., 2015).

Assumption 1 (Latent linear-Gaussian system)**.** *Let $Z_1, ..., Z_T$ be latent states related as a linear-Gaussian dynamical system in a space $\mathcal{Z} \subseteq \mathbb{R}^d$ with Z_1 of arbitrary distribution. Further, let the observation function be an injective map $\Psi : \mathcal{Z} \to \mathcal{X}$ with left-inverse (representation) $\Phi : \mathcal{X} \to \mathcal{Z}$. With $A_1, ..., A_{T-1} \in \mathbb{R}^{d \times d}$, $\beta \in \mathbb{R}^d$ assume that $Z_1, ..., Z_T, X_1, ..., X_T, Y$ are generated as*

$$Z_t = A_{t-1}^\top Z_{t-1} + \epsilon_t \quad \textit{for} \quad t \geq 2, \quad \textit{and} \quad \forall t : X_t = \Psi(Z_t) \quad \textit{and} \quad Y = \beta^\top Z_T + \epsilon_Y \ .$$

where $\epsilon_t \sim \mathcal{N}(0, \sigma_t^2 I)$ and $\epsilon_Y \sim \mathcal{N}(0, \sigma_Y^2)$.

It is easy to verify that Assumption 1 is consistent with (1), but stronger: there are more systems that map X_1 to Y as $\theta^\top \Phi(X_1)$ than systems where additionally $X_t = \Psi(Z_t)$. Nevertheless, it is much more general than the results of Karlsson et al. (2022), limited to the linear setting, i.e. $\Phi(x_1) = x_1$.

Next, we present a generalized version of the LuPTS algorithm of Karlsson et al. (2022) that is provably preferable to classical learning when data is generated according to Assumption 1 and Φ is known up to a linear transform. Further, we discuss how a privileged learner can be made universally consistent for unknown representations Φ when combined with random features.

3 Efficient learning with time-series privileged information

We analyze and compare learners from the privileged and classical paradigms which produce estimates of the form $\hat{h}(x_1) = \hat{\theta}^\top \hat{\Phi}(x_1)$, as motivated by (1). We let each use representations $\hat{\Phi} : \mathcal{X} \to \hat{\mathcal{Z}} \subseteq \mathbb{R}^{\hat{d}}$ from a family $\mathcal{F}$ *shared by both paradigms*, so that the hypothesis class $\mathcal{H} \ni \hat{h}$ is shared as well.

3.1 Learning with true representations known up to a linear transform

In this section, we assume that privileged and classical learners have access to a common, fixed representation function $\hat{\Phi}$ which is related to the true representation function through a linear transform, meaning there exists a matrix B such that $\hat{\Phi}(\cdot) = B\Phi(\cdot)$.

Classical learning. As a classical learner to serve as a strong comparison point for our privileged learners, we use the ordinary least-squares (OLS) linear estimator applied to the latent variables at the first time point inferred by $\hat{\Phi}$. With $\hat{\mathbf{Z}}_1 = [\hat{\Phi}(x_{1,1}), ..., \hat{\Phi}(x_{m,1})]^\top \in \mathbb{R}^{m \times \hat{d}}$,

$$\mathscr{A}_{\mathrm{C}}(D) = \hat{h}_{\mathrm{C}}(\cdot) := \hat{\theta}_{\mathrm{C}}^\top \hat{\Phi}(\cdot) \quad \text{where} \quad \hat{\theta}_{\mathrm{C}} := (\hat{\mathbf{Z}}_1^\top \hat{\mathbf{Z}}_1)^\dagger \hat{\mathbf{Z}}_1^\top \mathbf{Y} \ . \tag{3}$$

When $m > \hat{d}$ and Φ is known up to linear transform, $\hat{h}_{\mathrm{C}}$ is the minimum-risk unbiased estimator of h which *does not* use privileged information. To accommodate the underdetermined case where $m < \hat{d}$, the matrix inverse is replaced by the Moore-Penrose pseudo inverse $(\cdot)^\dagger$ (Penrose, 1956).

Generalized LuPTS. For privileged learning, we first compute $\hat{\mathbf{Z}}_t = [\hat{\Phi}(x_{1,t}), ..., \hat{\Phi}(x_{m,t})]^\top$ for $t = 1, ..., T$. Then, we independently fit parameter estimates $\hat{A}_1, ..., \hat{A}_T, \hat{\beta}$ of the dynamical system shown in Figure 1b by minimizing the squared error in single-step predictions. This equates to a series of OLS estimates in $\hat{\mathcal{Z}}$. At test time, baseline variables x_1 are embedded with $\hat{\Phi}$ and the latent dynamical system is simulated for T time steps to form a prediction $\hat{h}_{\mathrm{P}}(x_1) = (\hat{A}_1 \cdots \hat{A}_T \beta)^\top \hat{\Phi}(x_1)$. Putting this together, we arrive at Algorithm 1 which we call *generalized LuPTS*. We may apply a simple matrix identity and replace terms $\hat{\mathbf{Z}}_t \hat{\mathbf{Z}}_t^\top$ in Algorithm 1 by the Gram matrices $\hat{\mathbf{K}}_t = \kappa(\mathbf{X}_t, \mathbf{X}_t)$ of a reproducing kernel κ with corresponding (implicit) feature map $\hat{\Phi}$, $\kappa(x, x') = \langle \hat{\Phi}(x), \hat{\Phi}(x') \rangle$ (Smola and Schölkopf, 1998). This variant allows for learning with unknown Φ.

We now state a result which says that under Assumption 1, for an appropriate fixed representation $\hat{\Phi}$ or kernel κ, generalized LuPTS is never worse in expectation than the classical learner in (3).

Theorem 1. *Let D be a data set of size m drawn from p, consistent with Assumption 1. Assume that the left inverse $\Phi : \mathcal{X} \to \mathcal{Z}$ of the observation function Ψ is known up to a linear transform, explicitly or through a kernel $\kappa(x, x') = \langle \hat{\Phi}(x), \hat{\Phi}(x') \rangle$, i.e., there exists a matrix B with linearly independent columns such that $\hat{\Phi}(x) = B\Phi(x)$ with $\hat{\Phi} : \mathcal{X} \to \hat{\mathcal{Z}}$ for all $x \in \mathcal{X}$. Then, it holds for the privileged learner $\mathscr{A}_{\mathrm{P}}(D) = \hat{h}_{\mathrm{P}}$ (Algorithm 1) and the classical learner $\mathscr{A}_{\mathrm{C}}(D) = \hat{h}_{\mathrm{C}}$* (3)*,*

$$\overline{R}(\mathscr{A}_{\mathrm{P}}) = \overline{R}(\mathscr{A}_{\mathrm{C}}) - \mathbb{E}_{\hat{h}_{\mathrm{P}}, X_1}[\mathrm{Var}_D(\hat{h}_{\mathrm{C}}(X_1) \mid \hat{h}_{\mathrm{P}})] \ . \tag{4}$$

Proof sketch. First, we show that the predictions made by generalized LuPTS are invariant to a linear transform B applied to $\hat{\Phi}(\cdot)$ during training, see Appendix A. Then we consider two cases: (i) When $\hat{d} \leq m$ we may re-purpose the proof of Theorem 1 in Karlsson et al. (2022), (ii) When $m < \hat{d}$ Proposition 1 below directly implies $\mathbb{E}_{\hat{h}_{\mathrm{P}}, X_1}[\mathrm{Var}_D(\hat{h}_{\mathrm{C}}(X_1) \mid \hat{h}_{\mathrm{P}})] = 0$ and $\overline{R}(\mathscr{A}_{\mathrm{P}}) = \overline{R}(\mathscr{A}_{\mathrm{C}})$. □

Theorem 1 implies that the privileged learner is at least as sample efficient as the classical one since $\mathrm{Var}(\cdot) \geq 0$ and thus $\overline{R}(\mathscr{A}_{\mathrm{P}}) \leq \overline{R}(\mathscr{A}_{\mathrm{C}})$ for the same number of training samples m. The result is a direct generalization of the main result in Karlsson et al. (2022). We also observe that generalized LuPTS and the classical learner coincide under certain conditions when $m \leq \hat{d}$.

Proposition 1. *Let $\hat{\Phi} : \mathcal{X} \to \hat{\mathcal{Z}} \subseteq \mathbb{R}^{\hat{d}}$ be any map with corresponding kernel κ. Let $\hat{\mathbf{K}}_t$ be the Gram matrix of κ applied to $\mathbf{X}_t$ and let $\hat{h}_{\mathrm{C}}, \hat{h}_{\mathrm{P}}$ be classical* (3) *and privileged (Algorithm 1) estimates. Then,*

$$\hat{\mathbf{K}}_t \textit{ is invertible for all } t \implies \hat{h}_{\mathrm{P}} = \hat{h}_{\mathrm{C}}.$$

$\hat{\mathbf{K}}_t$ is noninvertible whenever $m > \hat{d}$, assuming linearly independent features.

Proof sketch. When the Gram matrices $\mathbf{K}_t$ are invertible, the pseudo-inverse coincides with the inverse, and factors $\mathbf{K}_t \mathbf{K}_t^\dagger$ in the LuPTS estimators cancel, making LuPTS and CL equal. □

Remarks. Theorem 1 extends the applicability of LuPTS to a) nonlinear prediction through a fixed feature map or b) kernel estimation and c) to the underdetermined case of $m < \hat{d}$. While previous work is restricted to observed linear systems our result considers the case of latent linear dynamics which only need to be identified up to a linear transform (see Figure 1). Proposition 1 does not claim that no preferable privileged learner exists; it is a statement only about generalized LuPTS. We may relate the result to the double descent characteristic previously observed for other linear estimators for fixed m and varying $\hat{d}$ (Loog et al., 2020). After a phase transition around $\hat{d} = m$ LuPTS's variance reduces for a second time when it becomes equivalent to the classical learner (see Figure 2).

3.2 Random feature maps for unknown representations

When the true Φ is entirely unknown, as is often the case in practice, using a poor representation $\hat{\Phi}$ may yield biased results for both classical and privileged learners. A common solution in non-linear prediction is to use a universal kernel, such as the Gaussian-RBF kernel. These have dense reproducing-kernel Hilbert spaces which allow approximation of any continuous function. However, universal kernels also have positive-definite and thus invertible Gram matrices (Hofmann et al., 2008), which according to Proposition 1 eliminates any gain in sample efficiency of generalized LuPTS.

Instead, we combine our algorithm with an approximation of universal kernels—random feature maps. These methods project inputs x onto $\hat{d}$ features by a random linear map $W \in \mathbb{R}^{k \times \hat{d}}$, and a nonlinear element-wise activation function. By choosing $\hat{d} < m$, we can benefit from the function approximation properties of universal kernels (see discussion below) and the variance reduction of generalized LuPTS. Popular random features include random Fourier features (RFF) (Rahimi and Recht, 2007) and random ReLU features (RRF) (Sun et al., 2018),[1]

$$\hat{\Phi}^{\gamma}_{\text{RFF}}(x) = \sqrt{2/\hat{d}}\left[\cos(\sqrt{2\gamma}W_{\mathcal{N}}^{\top}x + b)\right] \quad \text{and} \quad \hat{\Phi}^{\gamma}_{\text{RRF}}(x) = f_{+}(\gamma W_{\mathcal{U}}^{\top}[x;1])\ .$$

where $f_{+}(z) = \max(0, z)$ is the rectifier (ReLU) function and $\gamma > 0$ is a bandwidth hyperparameter.

For large enough numbers of random features and training samples, any continuous function h can be approximated up to arbitrary precision by a linear map ω applied to the random features, e.g., $\hat{h}(x) = \omega^{\top}\hat{\Phi}^{\gamma}_{\text{RRF}}(x)$, see Sun et al. (2019); Rudi and Rosasco (2017). Applying the same argument to the step-wise estimators of LuPTS we can justify using random features in Algorithm 1 by the following observation: *Under appropriate assumptions, we can construct a privileged learner using random feature maps which is a universally consistent estimator of* $h(x_1) = \mathbb{E}[Y|x_1]$. The precise construction deviates somewhat from Algorithm 1, but follows the same structure. We refer to Appendix B for a precise statement. As universal consistency describes the asymptotic behaviour in the limit of infinite samples and random features, this offers only limited insight into the benefits of privileged information in small sample settings, where performance will be a bias-variance trade-off.

Variance reduction & bias amplification. Generalized LuPTS is only guaranteed lower variance compared to the classical estimator under Assumption 1, although our empirical results (Section 4) suggest this applies more widely. When $\hat{\Phi}$ is a bad approximation of Φ, generalized LuPTS may amplify bias, increasing with the number of privileged time points, compared to classical learning. We show this theoretically in Appendix C and also empirically in Appendix F. Whether generalized LuPTS is still preferable to classical learning in terms of prediction risk appears to depend on the amount of bias that gets amplified. Our experiments imply the variance reduction mostly dominates when using random features, whereas this is not always the case for linear LuPTS. The phenomenon of bias amplification is familiar from e.g., model-based and model-free reinforcement learning (Kober et al., 2013). As the bias with random features may still be high for small sample settings, we next present privileged representation learning algorithms to trade off bias and variance more efficiently.

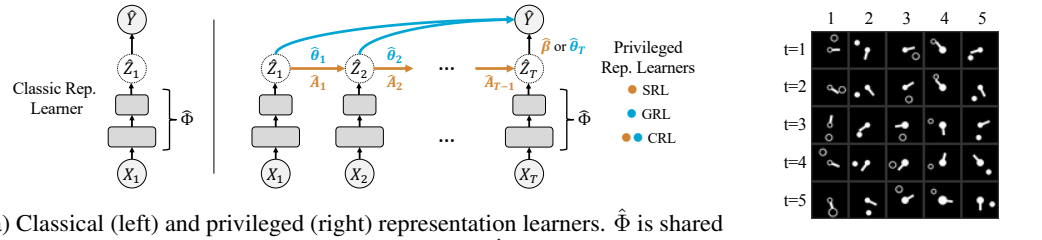
3.3 Privileged time-series representation learning

Up until now the representation $\hat{\Phi}$ was considered fixed, either because Φ was known up to a linear transform (explicitly or implicitly) or because of the use of random feature methods. Generalized LuPTS (Algorithm 1) produces minimizers $\{\hat{A}_t\}, \hat{\beta}$ of the following objective for fixed $\hat{\Phi}$,

$$\mathcal{L}_{\text{SRL}}(\hat{\Phi}, \{\hat{A}_t\}, \hat{\beta}) := \frac{1}{NT}\sum_{i=1}^{N}\left[\sum_{t=1}^{T-1}\frac{1}{\hat{d}}\left\|\hat{A}_t^{\top}\hat{\Phi}(x_{i,t}) - \hat{\Phi}(x_{i,t+1})\right\|_2^2 + \frac{1}{q}\left\|\hat{\beta}^{\top}\hat{\Phi}(x_{i,T}) - y_i\right\|_2^2\right]\ . \quad (5)$$

Objective (5) and the systems described by Assumption 1 lend themselves to methods which also learn the representation Φ in addition to the latent dynamics $\{A_t, \beta\}$. Next, we present three algorithms which combine the ideas of generalized LuPTS with the expressiveness of deep representation learning. All learners use equivalent encoders to represent $\hat{\Phi}(\cdot)$ and linear layers to model the relations between the latent variables $\{Z_t\}$ and the outcome Y. The classical learner predicts the outcome linearly from $\hat{Z}_1$. All architectures under consideration are visualized jointly in Figure 3a.

[1] For RFF, $[W_{\mathcal{N}}]_{ij} \sim \mathcal{N}(0,1)$, and for RRF, $[W_{\mathcal{U}}]_{ij} \sim \mathcal{U}(-1,1)$ and $b_i \sim \mathcal{U}(0, 2\pi)$.



(a) Classical (left) and privileged (right) representation learners. $\hat{\Phi}$ is shared across all time steps. GRL models the direct maps $\hat{\theta}_t$ to the outcome; SRL models the single steps $\hat{A}_t$ and $\hat{\beta}$. CRL combines the two.

(b) Five example sequences from **Clocks-LGS** image data.

Figure 3: Representation learning architectures (left) and samples from **Clocks-LGS** (right).

SRL. The first privileged representation learner directly optimizes objective (5), just like generalized LuPTS, but now also fitting the representation $\hat{\Phi}$, parameterized by a neural network. We refer to this model as *stepwise representation learner* (SRL). As we will see in experiments, a drawback of this approach is that representations may favor predicting transitions $\hat{z}_{i,t} \to \hat{z}_{i,t+1}$ with small error, while losing information relevant for the target outcome in the process. At test time, for a new input x_1, SRL composes the stepwise dynamics to output $\hat{h}_{\text{P}}(x_1) = \hat{\beta}^\top \hat{A}_T^\top \dots \hat{A}_1^\top \hat{\Phi}(x_1)$.

CRL and GRL. To make sure that the learned representation $\hat{\Phi}$ retains information about Y, we add linear outcome supervision to the representation $\hat{\Phi}(X_t)$ at each time step t. Recall that, by Assumption 1, the expected outcome is linear in the latent state at *any* time step. We introduce a hyperparameter $\lambda \in [0, 1]$ to trade off the two types of losses and arrive at the *combined representation learner* (CRL). With $\alpha = (\hat{\Phi}, \{\hat{A}_t\}, \{\hat{\theta}_t\})$ the entire parameter vector, CRL minimizes the objective

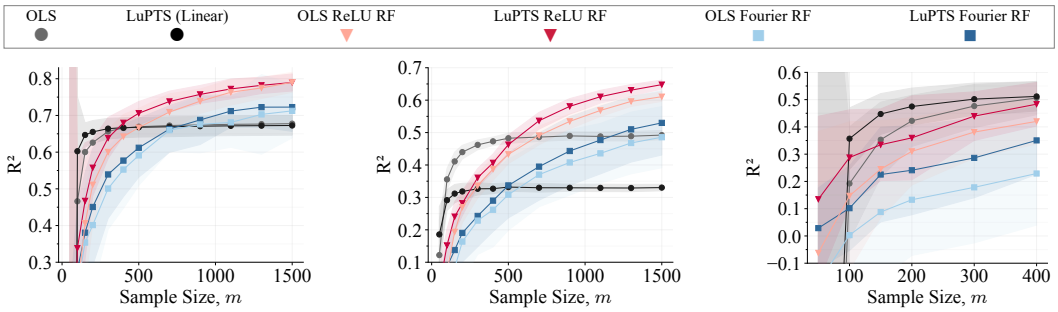
$$\mathcal{L}_{\text{CRL}}(\alpha) := \frac{\lambda}{NTq} \sum_{i,t} \left\| \hat{\theta}_t^\top \hat{\Phi}(x_{i,t}) - y_i \right\|_2^2 + \frac{1-\lambda}{N(T-1)\hat{d}} \sum_{i,t} \left\| \hat{A}_t^\top \hat{\Phi}(x_{i,t}) - \hat{\Phi}(x_{i,t+1}) \right\|_2^2. \quad (6)$$

We make test-time predictions using $\hat{h}_{\text{P}}(x_1) = \hat{\theta}_1^\top \hat{\Phi}(x_1)$. In experiments, we highlight the case $\lambda = 1$ where only outcome supervision is used as *greedy representation learner* (GRL). For a precise definition of the GRL objective, see Appendix D. GRL is related to multi-view learning, in which prediction of the same quantity is made from multiple "views" (cf. time points) (Zhao et al., 2017).

4 Experiments

We compare classical learning to variants of generalized LuPTS (Algorithm 1) and the privileged representation learners of Section 3.3 on two synthetic and two real-world time-series data sets. We (i) verify our theoretical findings by analyzing the sample efficiency and bias-variance characteristics of the given algorithms; (ii) demonstrate that generalized LuPTS with random features succeeds in settings where linear LuPTS suffers from large bias; (iii) point out that privileged representation learners offer even greater sample efficiency in practice and (iv) study how well these algorithms recover the true latent variables $\{Z_t\}$ and how this relates to predictive accuracy.

Experimental setup. We report the mean coefficient of determination (R^2), proportional to the squared-error risk $\overline{R}$, for varying sample sizes, sequence lengths and prediction horizons. Experiments are repeated and averaged over different random seeds. In each repetition, a given model performs hyperparameter tuning on the training data using random search and five-fold cross-validation before being re-trained on all training data. The test set size is 1000 samples for synthetic data and 20% of all data available for real-world data sets. We consider six privileged learners of two groups. The first group comprises generalized LuPTS with the linear kernel (LuPTS) and the two random feature maps shown in Section 3.2: Random Fourier features (Fourier RF) and random ReLU features (ReLU RF). The classical learners for this group are OLS estimators used with the same kernel or feature map. The second group consists of the representation learners SRL, CRL and GRL. For tabular data, their encoder is a multi-layer perceptron with three hidden layers of 25 neurons each. For the image data they use LeNet-5 (LeCun et al., 1989). The classical learner (Classic Rep.) uses the same encoder with a linear output layer. The results presented were found to be robust to small changes in training



(a) **Traffic**, $T = 3$. (b) **Square-Sign**, $T = 5$, $d = 10$. (c) **ADNI**, $T = 4$.

Figure 4: Predictive accuracy of generalized LuPTS on three data sets, over 60 repetitions. The shaded area represents one standard deviation above and below the mean over repetitions.

parameters such as learning rate. For details on the training process we refer to Appendix E. All experiments required less than 3000 GPU-h to complete using NVIDIA Tesla T4 GPUs.[2]

Data sets. We briefly describe evaluation data sets and refer to Appendix E for further details. First, we create two synthetic data sets in which latent states and outcomes are generated from linear-Gaussian systems as in Assumption 1. To produce the observations $\{X_t\}$ we use a deterministic nonlinear function $\Psi : \mathcal{Z} \to \mathcal{X}$. In the first synthetic data set, which we call **Square-Sign**, the nonlinear transformation $\Psi : \mathbb{R}^d \to \mathbb{R}^{2d}$ maps each latent feature $Z_{(t,k)}$ to a two dimensional vector such that

$$X_t := \Psi(Z_{(t)}) = [Z^2_{(t,1)}, \mathrm{sgn}(Z_{(t,1)}), \ldots, Z^2_{(t,d)}, \mathrm{sgn}(Z_{(t,d)})]^\top.$$

The second synthetic data set uses the same latent linear system with $Z_t \in \mathbb{R}^2$ and produces square images (28×28 pixels) as observations. As the images are reminiscent of clocks we refer to this data set as **Clocks-LGS**. Example sequences of these observations are presented in Figure 3b. The angle, size and fill of the two clock hands encode the value of the corresponding latent variable. The outcome Y is a linear function of Z_T with $q = 1$. For **Square-Sign**, $q = 3$.

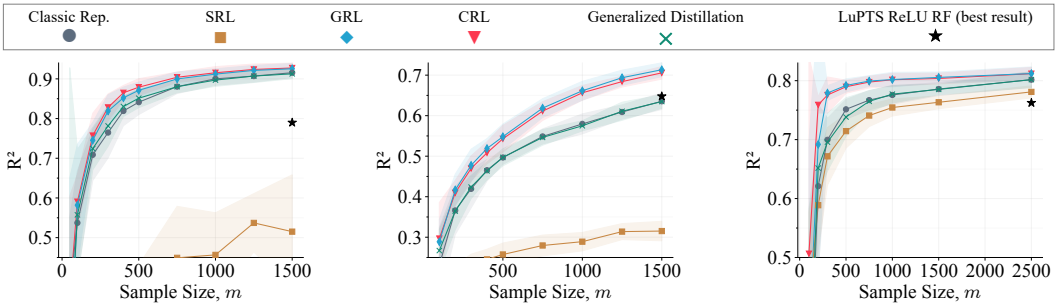
The Metro Interstate traffic volume data set (**Traffic**) (Hogue, 2012) contains hourly records of the traffic volume on the interstate 94 between Minneapolis and St. Paul, MN. In addition, the data contains weather features and a holiday indication. We predict the traffic volume for a fixed time horizon given the present observations. Privileged information is observed every four hours.

We also predict the progression of Alzheimer's disease (AD) as measured by the outcome of the Mini Mental State Examination (MMSE) (Galea and Woodward, 2005). The anonymized data were obtained through the Alzheimer's Disease Neuroimaging Initiative (**ADNI**) (ADNI, 2004). The outcome of interest is the MMSE score 48 months after the first examination. Privileged information are the measurements at 12, 24 and 36 months. In addition we tested our algorithms on the $\mathbf{PM}_{2.5}$ data set (Liang et al., 2016), where we predict the air quality in five Chinese cities (see Appendix F).

4.1 Sample efficiency, bias and variance

The main goal of our work is to improve learning efficiency by incorporating privileged time-series information. Across all prediction tasks and sample sizes, nonlinear variants of generalized LuPTS outperform their classical counterpart in terms of sample efficiency as can be seen in Figure 4. On the **Traffic** prediction task, LuPTS ReLU RF outperforms linear LuPTS as the former appears to exhibit less bias. On the synthetic data of **Square-Sign**, linear models reach their best accuracy quickly, while they are limited by their lack of expressiveness. Generalized LuPTS amplifies this bias, making linear LuPTS worse than OLS. Random feature methods attain higher accuracy here but are generally less sample efficient. Generalized LuPTS combined with random features manages to decrease this gap significantly. On **ADNI**, we don't see a benefit of using nonlinear models in general. We make similar observations on the $\mathbf{PM}_{2.5}$ air quality data set, see Appendix F.

[2]Code to reproduce all results is available at https://github.com/Healthy-AI/glupts.



(a) **Traffic**, $T = 3$. (b) **Square-Sign**, $T = 5$, $d = 10$. (c) **Clocks-LGS**, $T = 6$, $q = 1$.

Figure 5: Predictive accuracy of the representation learners over 25 repetitions. Generalized distillation is used as proposed by Hayashi et al. (2019). For details, see Appendix E.

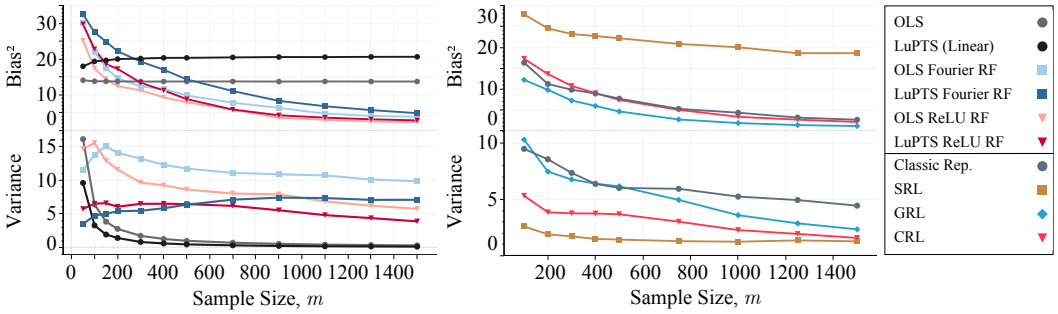
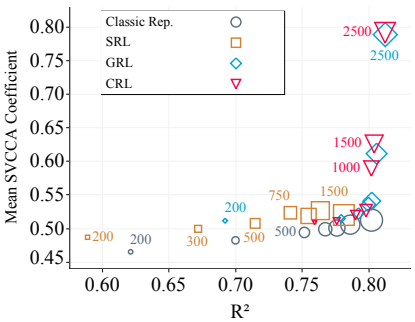


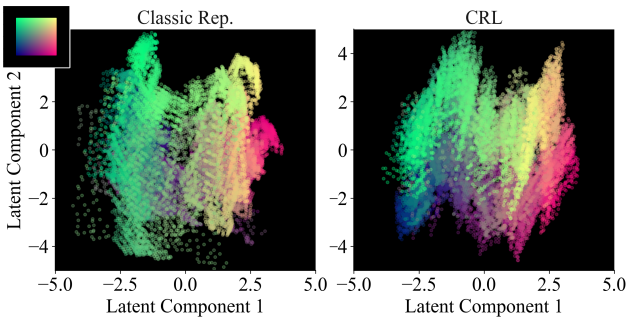
Figure 6: Estimated squared bias and variance of models trained on **Square-Sign** ($T = 5$, $d = 10$, $q = 3$) over random training sets (60 for left group, 25 for the right), evaluate on 1000 test points.

The representation learners proposed in Section 3.3 are evaluated on the same data sets and on the image prediction task **Clocks-LGS**. Example results are displayed in Figure 5. These demonstrate that directly transferring the LuPTS objective to neural networks in the form of SRL results in subpar performance. SRL does not seem to have a strong enough incentive to learn representations which accurately predict the outcome. Generalized distillation for privileged time series as suggested by Hayashi et al. (2019), does not improve upon classical learning in our tasks. On all experiments displayed in Figure 5, CRL and GRL outperform the classical learner. The predictive accuracy of these models is similar on most tasks, as may be explained by the fact that CRL may reduce to GRL when choosing $\lambda = 1$ in objective 6. Noticeably, the general observation of GRL and CRL being more sample efficient than the classic model, neither appears to depend on the neural architecture used for the encoder, nor does the modality of the data play an important role, as the image prediction task **Clocks-LGS** (Figure 5c) demonstrates. For additional empirical results, we refer to Appendix F.

To analyze the bias and variance characteristics of our algorithms, we can estimate the expected squared prediction *bias*, $\mathbb{E}_{X_1}[(\mathbb{E}_D[\mathscr{A}(D)](X_1) - \mathbb{E}[Y|X_1])^2]$, by computing $\mathbb{E}_Y[Y|X_1]$ in synthetic DGPs in addition to the variance of the different estimators. Figure 6 depicts bias and variance for all models on the **Square-Sign** data. On the left panel, all variants of generalized LuPTS exhibit lower variance than classical learning, despite being biased. This holds generally: *Across all experiments, we never encountered an example where the use of privileged information has not resulted in lower variance compared to classical learning.* On the contrary, the privileged learners suffer higher bias than the comparable classical learners because of the bias compounding over the individual prediction steps as shown in Appendix C. For the random feature variants however, the bias decreases with the number of samples. The representation learners display similar characteristics on the right panel of Figure 6. Learning the transitions between latent variables Z_t appears to be associated with low variance and high bias as demonstrated by the results of SRL. GRL however, which does not model these transitions, exhibits the lowest bias and the largest variance of all privileged learners. As CRL is able to trade off between these two objectives, the estimates for its variance and bias lie in between the corresponding values of the other two privileged learners.



(a) Mean R^2 and SVVCA coefficient over 25 training runs. Markers represent average results over repeated experiments with size indicating the sample size, m.



(b) Visualizing the representations learned by Classic Rep. and CRL after applying SVCCA. The recovery target is shown in the top left corner. Both estimators are best in class and trained on 2500 samples.

Figure 7: Analyzing the representations learned by the models of Section 3.3 using SVVCA on the **Clocks-LGS** image regression data set (same setup as in Figure 5c).

4.2 Latent variable recovery

Under Assumption 1, it is sufficient to identify the representation function Φ up to a linear transform B to have provable gains from privileged information over a classical learner. In synthetic data, we can assess to what extent a representation $\hat{\Phi}$ with this property has been found. As a proxy for the existence of such a transform, we can compute a measure of correlation between $\hat{\Phi}(X)$ and $\Phi(X)$, such as the Canonical Correlation Analysis (CCA) (Hotelling, 1936). Raghu et al. (2017) introduced a modified version called Singular Vector Canonical Correlation Analysis (SVCCA) to compute correlations when dealing with noisy dimensions in neural network representations.

The mean SVCCA coefficients $\bar{\rho}$ and predictive accuracy of the representation learners are visualized in Figure 7a. One notices that GRL and CRL produce higher correlation coefficients than the classical learner while also predicting the outcome on the **Clocks-LGS** task more accurately. Further comparing the representations learned by privileged and classical models, Figure 7b shows a visual example of the improved latent recovery of CRL on the same task. Concluding, the use of privileged time-series information does not only increase the sample efficiency of existing algorithms but can also aid the recovery of latent variables in latent dynamical systems.

5 Related work

Existing analyses for learning with privileged information guarantee improvements in sample efficiency under strong assumptions but are insufficient to establish a clear preference for LuPI learners in small-sample settings, as argued in the Introduction. We are not aware of results tight enough to establish a lower bound on the risk of a classical learner larger than an upper bound for a PI learner. Additionally, our problem, see especially (5)–(6), is related to multi-task (representation) learning (Maurer et al., 2016). However, our goal is different in that only a single task is of interest after learning. In estimation of causal effects, learning from *surrogate outcomes* (Prentice, 1989) has been proposed as a way to increase sample efficiency. Surrogates are variables related to the outcome which may be available even when the outcome is not (Athey et al., 2019). We can view these as privileged information. While the problem shares structure with ours, the goal is to compensate for *missing* outcomes and analytical results give no guarantees for improved efficiency when both surrogates outcomes are always observed (Chen et al., 2008; Kallus and Mao, 2020).

Theorem 1 relies on the representation function Φ being identified up to a linear transformation. Nonlinear independent component analysis (ICA) aims to solve precisely this problem (Hyvärinen and Pajunen, 1999). It has been shown that the nonstationarity of time series provides the conditions for identifiability through time-contrastive learning (Hyvarinen and Morioka, 2016). These results make for a potentially interesting connection between learning using privileged time series and the identifiability of nonlinear ICA. We hope to explore this connection in future work.

6 Discussion

We have presented learning algorithms for predicting nonlinear outcomes by utilizing time-series privileged information during training. We prove that our estimator is preferable to classical learning when data is generated from a latent-variable dynamical system with partially known components. The proof holds for the case when the latent dynamical system is recovered by the representation function up to a linear transform, assuming that a left inverse of the true observation generating function exists. However, this assumption does not appear to be necessary, as our empirical results demonstrate that privileged learning is preferable to classical learning even when these conditions cannot be guaranteed. Consequently, a more general theoretical result where less is known about the latent system seems attainable. For example, one might consider a case in which only a few independent components of the latent variables are recovered by the representation used by the learning algorithm, while other components are treated as noise.

When the latent dynamical system is entirely unknown, we create practical estimators using random feature embeddings which outperform the corresponding classical learner across experiments. We show that a universally consistent learner can be constructed based on this idea, with slightly different form. As a further alternative, we propose representation learning methods of related form using neural networks and demonstrate the empirical benefits also of this estimator over classical learning. In experiments, we analyze how the gap in risk between privileged and classical learning changes for different prediction horizons, as displayed in Figure 16 in Appendix F. The results suggest that the risk advantage of privileged learners grows with the sequence length of the prediction task, despite the fact that the task becomes more difficult at the same time.

Our work focuses on the setting where data is observed as a time series. This setting is chosen for its simple causal structure given by (latent) linear-Gaussian systems and because it can be motivated from many different applications. However, the ideas presented are not specifically tied to time and also apply in the case when all variables are observed simultaneously as long as the causal structure remains sequential. Moreover, we believe that the theory presented here is not limited to sequential settings and generalizes to other causal structures, in particular directed acyclic graphs that connect the baseline covariates to the outcome. In either case, one might only have access to the baseline variable at test time. For example, the timely collection of data for all covariates at test time might be very expensive or even impossible.

As pointed out before, Theorem 1 requires the recovery of the latent variables up to a linear transformation. Nonlinear independent component analysis (ICA) (Hyvarinen and Morioka, 2016) aims to solve precisely this problem and has been applied to time series via time-contrastive learning. This makes for an interesting connection between learning using privileged information and nonlinear ICA, as the experiments of Section 4.2 suggest that privileged time-series information aids the recovery of latent variables. Other remaining challenges include providing risk guarantees for learning with biased representations (including deep neural networks), with regularized estimators, and for more general data generating processes with weaker structural assumptions. We are hopeful that the utility demonstrated in this work will inspire future research to overcome these limitations.

Acknowledgments and Disclosure of Funding

We would like to thank Anton Matsson and Rickard Karlsson for insightful feedback and the Alzheimer's Neuroimaging Initiative (ADNI) for collecting and providing the data on Alzheimer's disease used in this project. The present work was funded in part by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

References

ADNI. the Alzheimer's Disease Neuroimaging Initiative (ADNI), 2004. URL `http://adni.loni.usc.edu`.

Susan Athey, Raj Chetty, Guido W Imbens, and Hyunseung Kang. The surrogate index: Combining short-term proxies to estimate long-term treatment effects more rapidly and precisely. Technical report, National Bureau of Economic Research, 2019.

Song Xi Chen, Denis HY Leung, and Jing Qin. Improving semiparametric estimation by using surrogate data. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(4): 803–823, 2008.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

Mary Galea and Michael Woodward. Mini-mental state examination (mmse). *Australian Journal of Physiotherapy*, 51(3):198, 2005. ISSN 0004-9514. doi: https://doi.org/10.1016/S0004-9514(05)70034-9.

Shogo Hayashi, Akira Tanimoto, and Hisashi Kashima. Long-term prediction of small time-series data using generalized distillation. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.

Thomas Hofmann, Bernhard Schölkopf, and Alexander J. Smola. Kernel methods in machine learning. *The Annals of Statistics*, 36(3), jun 2008.

John Hogue. Metro interstate traffic volume data set, 2012. URL `https://archive.ics.uci.edu/ml/datasets/Metro+Interstate+Traffic+Volume`.

Harold Hotelling. Relations between two Sets of Variates*. *Biometrika*, 28(3-4):321–377, 12 1936. ISSN 0006-3444. doi: 10.1093/biomet/28.3-4.321.

Bingquan Huang, Mohand Tahar Kechadi, and Brian Buckley. Customer churn prediction in telecommunications. *Expert Systems with Applications*, 39(1):1414–1425, 2012.

Aapo Hyvarinen and Hiroshi Morioka. Unsupervised feature extraction by time-contrastive learning and nonlinear ica. *Advances in Neural Information Processing Systems*, 29, 2016.

Aapo Hyvärinen and Petteri Pajunen. Nonlinear independent component analysis: Existence and uniqueness results. *Neural networks*, 12(3):429–439, 1999.

Rico Jonschkowski, Sebastian Höfer, and Oliver Brock. Patterns for learning with side information. *arXiv preprint arXiv:1511.06429*, 2015.

Nathan Kallus and Xiaojie Mao. On the role of surrogates in the efficient estimation of treatment effects with limited outcome data. *arXiv preprint arXiv:2003.12408*, 2020.

Jonathan C. Kao, Paul Nuyujukian, Stephen I. Ryu, Mark M. Churchland, John P. Cunningham, and Krishna V. Shenoy. Single-trial dynamics of motor cortex and their applications to brain-machine interfaces. *Nature Communications*, 6(1), July 2015. doi: 10.1038/ncomms8759.

Aditya V Karhade, Quirina CBS Thio, Paul T Ogink, Christopher M Bono, Marco L Ferrone, Kevin S Oh, Philip J Saylor, Andrew J Schoenfeld, John H Shin, Mitchel B Harris, et al. Predicting 90-day and 1-year mortality in spinal metastatic disease: development and internal validation. *Neurosurgery*, 85(4):E671–E681, 2019.

Rickard Karlsson, Martin Willbo, Zeshan Hussain, Rahul G Krishnan, David Sontag, and Fredrik D Johansson. Using time-series privileged information for provably efficient learning of prediction models. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics.*, 2022.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

Jens Kober, J. Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *Int. J. Rob. Res.*, 32(11):1238–1274, sep 2013. ISSN 0278-3649. doi: 10.1177/0278364913495721.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, 1(4): 541–551, 12 1989. ISSN 0899-7667. doi: 10.1162/neco.1989.1.4.541.

Kookjin Lee and Kevin Carlberg. Deep conservation: A latent-dynamics model for exact satisfaction of physical conservation laws, 2019.

Xuan Liang, Shuo Li, Shuyi Zhang, Hui Huang, and Song Xi Chen. Pm2.5 data reliability, consistency, and air quality assessment in five chinese cities. *Journal of Geophysical Research: Atmospheres*, 121(17):10220–10236, September 2016. ISSN 2169-897X. doi: 10.1002/2016JD024877.

Marco Loog, Tom Viering, Alexander Mey, Jesse H. Krijthe, and David M. J. Tax. A brief prehistory of double descent. *Proceedings of the National Academy of Sciences*, 117(20):10625–10626, 2020. doi: 10.1073/pnas.2001875117.

D. Lopez-Paz, B. Schölkopf, L. Bottou, and V. Vapnik. Unifying distillation and privileged information. In *International Conference on Learning Representations (ICLR)*, November 2016.

Andreas Maurer, Massimiliano Pontil, and Bernardino Romera-Paredes. The benefit of multitask representation learning. *Journal of Machine Learning Research*, 17(81):1–32, 2016.

Bobak J Mortazavi, Nicholas S Downing, Emily M Bucholz, Kumar Dharmarajan, Ajay Manhapra, Shu-Xia Li, Sahand N Negahban, and Harlan M Krumholz. Analysis of machine learning techniques for heart failure readmissions. *Circulation: Cardiovascular Quality and Outcomes*, 9 (6):629–640, 2016.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

Dmitry Pechyony and Vladimir Vapnik. On the theory of learnining with privileged information. *Advances in neural information processing systems*, 23, 2010.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Roger Penrose. On best approximate solutions of linear matrix equations. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 52, pages 17–19. Cambridge University Press, 1956.

Ross L Prentice. Surrogate endpoints in clinical trials: definition and operational criteria. *Statistics in medicine*, 8(4):431–440, 1989.

Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability, 2017.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2007.

David S Rosenberg and Peter L Bartlett. The rademacher complexity of co-regularized kernel classes. In *Artificial Intelligence and Statistics*, pages 396–403. PMLR, 2007.

Alessandro Rudi and Lorenzo Rosasco. Generalization properties of learning with random features. *Advances in neural information processing systems*, 30, 2017.

Alex J Smola and Bernhard Schölkopf. *Learning with kernels*, volume 4. Citeseer, 1998.

Y Sun, A Gilbert, and A Tewari. On the approximation capabilities of relu neural networks and random relu features. *arXiv preprint arXiv:1810.04374*, 2019.

Yitong Sun, Anna Gilbert, and Ambuj Tewari. On the approximation properties of random relu features, 2018.

Vladimir Vapnik and Akshay Vashist. A new learning paradigm: Learning using privileged information. *Neural networks*, 22(5-6):544–557, 2009.

Vladimir Vapnik, Rauf Izmailov, et al. Learning using privileged information: similarity control and knowledge transfer. *J. Mach. Learn. Res.*, 16(1):2023–2049, 2015.

Weiran Wang. Everything old is new again: A multi-view learning approach to learning using privileged information and distillation. *arXiv preprint arXiv:1903.03694*, 2019.

Jiaxuan You, Xiaocheng Li, Melvin Low, David Lobell, and Stefano Ermon. Deep gaussian process for crop yield prediction based on remote sensing data. In *Thirty-First AAAI conference on artificial intelligence*, 2017.

Jing Zhao, Xijiong Xie, Xin Xu, and Shiliang Sun. Multi-view learning overview: Recent progress and new challenges. *Information Fusion*, 38:43–54, 2017. ISSN 1566-2535. doi: https://doi.org/10.1016/j.inffus.2017.02.007.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] The introduction refers to the section where each contribution is made.
   (b) Did you describe the limitations of your work? [Yes] See e.g., the discussion in Section 6 and on the applicability of Theorem 1 in Section 3.1 and 3.2.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] We have not identified any potential negative societal impact directly related to our work.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Assumption 1 and Theorem 1.
   (b) Did you include complete proofs of all theoretical results? [Yes] Yes, see the Appendix.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] A URL to a code repository is included in the paper.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 4 and Appendix E
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Each result figure includes error regions.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Section 4
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Appendix E.2
   (b) Did you mention the license of the assets? [Yes] See Appendix E.2
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] Only original code.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] ADNI data is anonymized.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Appendix

A Proof of Theorem 1

Our proof requires an additional technical assumption: that the matrix of true latent states $\mathbf{Z}_t = [\Phi(x_{1,t}), ..., \Phi(x_{m,t})]^\top$, for all t, for a random data set D has independent columns with probability 1. This implies that $\text{rank}(\mathbf{Z}_t) = d$ and $m \geq d$. We consider this a minor restriction since it would only be violated if either a) two or more components of Z_t were perfectly correlated—in this case, a smaller system with same distributions over observations could always be constructed—or b) if we observe fewer samples than necesary to determine the system ($m < d$). Note that this *does not* require that the dimension $\hat{d}$ of the estimated representation $\hat{\Phi}$ is smaller than m. We begin by proving that both classical and LuPTS estimators are invariant to a particular form of linear transformation of the representation $\hat{\Phi}$.

Lemma 1. *Assume we have a latent linear Gaussian system as defined in Assumption 1 such that for a data set D of m samples, the matrix of true latent states $\mathbf{Z}_t = [\Phi(X_{1,t}); ...; \Phi(X_{m,t})]$ has linearly independent columns with probability 1. Let $\mathscr{A}_{\text{P}}^{\Phi}$ be the LuPTS algorithm using the system's true map $\Phi(\cdot)$ with $\Phi(\Psi(z)) = z \; \forall z \in \mathcal{Z}$. Let $\mathscr{A}_{\text{P}}^{\hat{\Phi}}$ be the same algorithm using a different map $\hat{\Phi}(\cdot)$. We assume that $\exists B : \hat{\Phi}(x) = B\Phi(x) \forall x \in \mathcal{X}$. Analogously, we denote the classical learners $\mathscr{A}_{\text{C}}^{\hat{\Phi}}$ and $\mathscr{A}_{\text{C}}^{\Phi}$. If $B \in \mathbb{R}^{\hat{d} \times d}$ has linearly independent columns we have*

$$\hat{h}_{\text{P}}^{\hat{\Phi}}(x) = \hat{h}_{\text{P}}^{\Phi}(x)$$
$$\textit{and} \;\; \hat{h}_{\text{C}}^{\hat{\Phi}}(x) = \hat{h}_{\text{C}}^{\Phi}(x).$$

Proof. Let $\mathbf{Z}_t \in \mathbb{R}^{m \times d}$ be made up of the rows $\mathbf{Z}_{t(i,:)} = \Phi(\mathbf{X}_{t(i,:)})$ when $\mathbf{X}_t \in \mathbb{R}^{m \times k}$ is the design matrix belonging to data set D. In the same fashion we define $\hat{\mathbf{Z}}_t \in \mathbb{R}^{m \times \hat{d}}$ using the map $\hat{\Phi}$ instead. By assumption, $\mathbf{Z}_t$ and $B \in \mathbb{R}^{\hat{d} \times d}$ have independent columns such that $B^\dagger B = I$. These assumptions are used for matrix identities involving the Moore-Penrose inverse below. We compute the prediction on a new test point x for the classical learner:

$$\begin{aligned}
\hat{h}_{\text{C}}^{\hat{\Phi}}(x) &= \left((\hat{\mathbf{Z}}_1^\top \hat{\mathbf{Z}}_1)^\dagger \hat{\mathbf{Z}}_1^\top Y\right)^\top \hat{\Phi}(x) \\
&= \left(\left((\mathbf{Z}_1 B^\top)^\top (\mathbf{Z}_1 B^\top)\right)^\dagger (\mathbf{Z}_1 B^\top)^\top \mathbf{Y}\right)^\top B\Phi(x) \\
&= \left((\mathbf{Z}_1 B^\top)^\dagger (B\mathbf{Z}_1^\top)^\dagger (\mathbf{Z}_1 B^\top)^\top \mathbf{Y}\right)^\top B\Phi(x) \\
&= \left((B^\top)^\dagger \mathbf{Z}_1^\dagger (\mathbf{Z}_1^\top)^\dagger B^\dagger (\mathbf{Z}_1 B^\top)^\top \mathbf{Y}\right)^\top B\Phi(x) \\
&= \left((B^\top)^\dagger \left(\mathbf{Z}_1^\top \mathbf{Z}_1\right)^\dagger \underbrace{B^\dagger B}_{=I} \mathbf{Z}_1^\top \mathbf{Y}\right)^\top B\Phi(x) \\
&= \left(\left(\mathbf{Z}_1^\top \mathbf{Z}_1\right)^\dagger \mathbf{Z}_1^\top \mathbf{Y}\right)^\top \underbrace{B^\dagger B}_{=I} \Phi(x) \\
&= \hat{h}_{\text{C}}^{\Phi}(x)
\end{aligned}$$

The arguments for the privileged learners are analogous:

$$\begin{aligned}
\hat{h}_{\text{P}}^{\hat{\Phi}}(x) &= \left[(\hat{\mathbf{Z}}_1^\top \hat{\mathbf{Z}}_1)^\dagger \hat{\mathbf{Z}}_1^\top \hat{\mathbf{Z}}_2 \; (\hat{\mathbf{Z}}_2^\top \hat{\mathbf{Z}}_2)^\dagger \hat{\mathbf{Z}}_2^\top \hat{\mathbf{Z}}_3 \; \ldots \; (\hat{\mathbf{Z}}_T^\top \hat{\mathbf{Z}}_T)^\dagger \hat{\mathbf{Z}}_T^\top \mathbf{Y}\right]^\top \hat{\Phi}(x) \\
&= \Big[(B^\top)^\dagger (\mathbf{Z}_1^\top \mathbf{Z}_1)^\dagger B^\dagger B \mathbf{Z}_1^\top \mathbf{Z}_2 B^\top (B^\top)^\dagger (\mathbf{Z}_2^\top \mathbf{Z}_2)^\dagger B^\dagger B \mathbf{Z}_2^\top \mathbf{Z}_3 B^\top
\end{aligned}$$

$$
\begin{aligned}
&\dots\ (B^\top)^\dagger(\mathbf{Z}_T^\top\mathbf{Z}_T)^\dagger B^\dagger B\mathbf{Z}_T^\top\mathbf{Y}\Big]^\top B\Phi(x)\\
&= \Big[(\mathbf{Z}_1^\top\mathbf{Z}_1)^\dagger\mathbf{Z}_1^\top\mathbf{Z}_2(\mathbf{Z}_2^\top\mathbf{Z}_2)^\dagger\mathbf{Z}_2^\top\mathbf{Z}_3(\mathbf{Z}_T^\top\mathbf{Z}_T)^\dagger\mathbf{Z}_T^\top\mathbf{Y}\Big]^\top B^\dagger B\Phi(x)\\
&= \hat{h}_{\mathrm{P}}^{\Phi}(x)
\end{aligned}
$$

□

Proof of Theorem 1. We consider the generalized LuPTS estimator $\hat{h}_{\mathrm{P}}^{\hat{\Phi}}(\cdot)$ treating different cases for the number of samples m and latent state dimension d in turn. By the added technical assumption, that the true latent state $\mathbf{Z}_t = [\Phi(x_{1,t}), ..., \Phi(x_{m,t})]^\top$ has rank d for all t with probability 1, and the assumption that $\hat{\Phi}$ is linearly close to Φ, by a matrix $B \in \mathbb{R}^{\hat{d}\times d}$ of rank d such that $\hat{\mathbf{Z}}_t = \mathbf{Z}_t B^\top$, we get that rank$(\hat{\mathbf{Z}}_t) = d$ for all t. This also implies $d \leq \hat{d}$.

(i) $m = d$: In this case, the Gram matrix $\mathbf{K}_t = \hat{\mathbf{Z}}_t\hat{\mathbf{Z}}_t^\top$ has full rank and thus is invertible for all t. By Proposition 1, LuPTS coincides with the classical learner. Hence, $\mathbb{E}_{\hat{h}_{\mathrm{P}},X_1}[\mathrm{Var}_D(\hat{h}_{\mathrm{C}}(X_1) \mid \hat{h}_{\mathrm{P}})] = 0$ and $\overline{R}(\mathscr{A}_{\mathrm{P}}) = \overline{R}(\mathscr{A}_{\mathrm{C}})$.

(ii) $m < d$: In this case, there does not exists a linearly close, as defined above, representation $\hat{\Phi}$ to Φ since the rank of $\hat{\mathbf{Z}}_t$ must be smaller than d. This contradicts that rank$(\hat{\mathbf{Z}}_t) = d$. Independently, if the conditions of Proposition 1 hold, the same equivalence holds as in the case $m = d$.

(iii) $m > d$: In this case, the kernel Gram matrix $\mathbf{K}_t = \hat{\mathbf{Z}}_t\hat{\mathbf{Z}}_t^\top$ has rank $d < m$ and is never invertible.

Three sub-cases remain: a) When $\hat{d} = d$, the matrix B is invertible and square, the covariance matrix $\hat{\Sigma}_t = \hat{\mathbf{Z}}_t^\top\hat{\mathbf{Z}}_t$ is invertible for all t and our estimator coincides with linear LuPTS (Karlsson et al., 2022) in the space implied by $\hat{\Phi}$. To see this, note that Lemma 1 implies that $\hat{h}_{\mathrm{P}}^{\hat{\Phi}}(\cdot)$ makes the same predictions as a different generalized LuPTS estimator $\hat{h}_{\mathrm{P}}^{\Phi}(\cdot)$ using the true map Φ when the two representation functions are related through B, as defined in the Theorem statement. Consequently, we may analyze the latter estimator instead of the first. It uses the parameter

$$
\hat{\theta}_{\mathrm{P}} := \left[\prod_{t=1}^{T-1}\underbrace{(\mathbf{Z}_t^\top\mathbf{Z}_t)^\dagger\mathbf{Z}_t^\top\mathbf{Z}_{t+1}}_{\hat{A}_t}\right]\underbrace{(\mathbf{Z}_T^\top\mathbf{Z}_T)^\dagger\mathbf{Z}_T^\top\mathbf{Y}}_{\hat{\beta}}\ .
$$

We know by assumption that the covariance matrices $\Sigma_t = (\mathbf{Z}_t^\top\mathbf{Z}_t) \in \mathbb{R}^{d\times d}$ have full rank for all t. This implies that the Moore-Penrose pseudoinverse $(\cdot)^\dagger$ may be replaced by the regular matrix inverse $(\cdot)^{-1}$ in the expression above, yielding

$$
\hat{\theta}_{\mathrm{P}} = \left[\prod_{t=1}^{T-1}\underbrace{(\mathbf{Z}_t^\top\mathbf{Z}_t)^{-1}\mathbf{Z}_t^\top\mathbf{Z}_{t+1}}_{\hat{A}_t}\right]\underbrace{(\mathbf{Z}_T^\top\mathbf{Z}_T)^{-1}\mathbf{Z}_T^\top\mathbf{Y}}_{\hat{\beta}}
$$

which is equivalent to the LuPTS estimator of Karlsson et al. (2022) used on a linear-Gaussian system in the space $\mathcal{Z}$ rather than in $\mathcal{X}$. In this case, Theorem 1 from Karlsson et al. (2022) yields the desired result. b) If $\hat{d} > d$, $\hat{\Sigma}$ is not invertible but, due to Lemma 1, we can instead study a representation which is an appropriate linear transform $B \in \mathbb{R}^{\hat{d}\times d}$ away from $\hat{\mathbf{Z}}$, and apply the Karlsson et al. (2022) result as described for the case $\hat{d} = d$. Note that in this case B is non-square but has linearly independent columns as required. c) If $\hat{d} < d$, the assumed matrix B cannot exist with the stated conditions (the assumptions of Theorem 1 are not satisfied).

□

B Universality of random features

A learning algorithm $\mathscr{A}$ is said to be universally consistent if, for any continuous function h, the output of $\mathscr{A}$ converges in probability to h. That is, for a random dataset D_m of m i.i.d. samples

drawn from a distribution p, and any $\epsilon > 0$,

$$\lim_{m\to\infty} \Pr[\|\mathscr{A}(D_m) - h\|_{L^2(p)} > \epsilon] = 0 \,.$$

Sun et al. (2019) prove that (norm-bounded) linear regression applied to random ReLU features (RRF) is universally consistent. Specifically, for any $\epsilon, \delta > 0$, there is a finite number of random features $\hat{d}$ and samples m, such that the estimator

$$\hat{h}_{\mathrm{RRF}}(x) = \hat{\theta}^\top \hat{\Phi}^{\gamma,\hat{d}}_{\mathrm{RRF}}(x) \quad \text{with} \quad \hat{\theta} = \operatorname*{arg\,min}_{\theta:\|\theta\|_2^2<R^2} \frac{1}{m}\sum_{i=1}^{m}(\theta^\top \hat{\Phi}^{\gamma,\hat{d}}_{\mathrm{RRF}}(x_i) - y_i)^2$$

achieves an error of at most ϵ with probability $\geq 1-\delta$ for univariate continuous functions of x. Universal consistency (as $\hat{d} \to \infty, m > \hat{d}$)) follows as a result. We can apply the same idea to a version of generalized LuPTS by considering each parameter estimate of the latent dynamical system given $\hat{\Phi}$, with norms restricted by R. We drop the subscript RRF from $\hat{\Phi}$ moving forward, and continue to use random ReLU features. For the final prediction step of Y, we let

$$\hat{\beta} = \operatorname*{arg\,min}_{b:\|b\|_2^2<R^2} \frac{1}{m}\sum_{i=1}^{m}(b^\top \hat{\Phi}^{\gamma,\hat{d}_T}(x_{i,T}) - y_i)^2 \tag{7}$$

Progressing recursively backward from $t = T$, we let

$$[\hat{A}_t]_{j,:} = \operatorname*{arg\,min}_{a:\|a\|_2^2<R^2} \frac{1}{m}\sum_{i=1}^{m}(a^\top \hat{\Phi}^{\gamma,\hat{d}_t}(x_{i,t}) - \hat{\Phi}^{\gamma,\hat{d}_{t+1}}(x_{i,t+1})_j)^2 \;\text{ for }\; j \in [\hat{d}_{t+1}], t \in [T-1] \tag{8}$$

where $\hat{\Phi}^{\gamma,\hat{d}_t}$ are random features specific to t. Since the target of each prediction at $t+1$ is fixed with respect to the features used as input at t, the result from Sun et al. (2019) can be used to give a learning guarantee for each step. The construction differs from the standard generalized LuPTS formulation as $\hat{\Phi}$ is not shared between time steps, and so $\hat{A}_t$ will be non-square in general. We believe that this is merely a limitation of the proof technique and that the results hold for shared random features and square transitions.

Let $\mathcal{H}_{\mathrm{ReLU}}$ denote the set of linear functions applied to $\hat{d}$ random ReLU features, with uniform random projection coefficients $\omega_j \sim \mathcal{U}(\mathbb{S}^d)$, $j = 1, ..., \hat{d}$.

$$\mathcal{H}_{\mathrm{ReLU},\hat{d}} = \left\{ h : \mathcal{X} \to \mathbb{R}, h(x) = \sum_{j=1}^{\hat{d}} a_j \sigma(\omega_j^\top [x; 1]) \right\} \,.$$

Corollary 1 (Follows from Proposition 5 in Sun et al. (2019))**.** *Let Assumption 1 hold with noiseless transitions and outcomes, $\epsilon_t = 0$ for $t = 2, ..., T$, and $\epsilon_Y = 0$. Define $g_t^*(x_t) := \Psi(A_t^\top \Phi(x_t)) = x_{t+1}$ and assume that for any fixed RRF representation $\hat{\Phi}^{\gamma,\hat{d}_{t+1}}$, each component $j = 1, ..., \hat{d}_{t+1}$ of the transition target satisfies $\hat{\Phi}^{\gamma,\hat{d}_{t+1}}(g_t^*(\cdot))(j) \in \mathcal{H}_{\mathrm{ReLU},\hat{d}_t}$. Let $\hat{A}_t$ be the minimizer of the single-step transitions as defined in* (8)*. Then, for any $\delta > 0, \epsilon > 0$ there is a number of random features $\hat{d} = \hat{d}(\epsilon, \delta)$ and samples $m = m(\epsilon, \delta)$, such that with probability $\geq 1-\delta$,*

$$\|\hat{A}_t^\top \hat{\Phi}^{\gamma,\hat{d}_t}(x_t) - \hat{\Phi}^{\gamma,\hat{d}_{t+1}}(x_{t+1})\| \leq \epsilon \,.$$

The result follows from Proposition 5 in Sun et al. (2019) applied to the transition functions in our problem. Putting this together for all time-steps, we get the following result.

Proposition 2 (Universal consistency of RRF privileged learner)**.** *Let Assumption 1 hold with $\epsilon_t = 0, \epsilon_Y = 0$. By Corollary1 and Sun et al. (2019), we have for any $\delta > 0$, $\epsilon > 0, \gamma > 0$ and a sequence $(\hat{d}_1, ..., \hat{d}_T)$ of sufficiently large numbers of random features and samples m, that with probability at least $1 - \delta/T$, for the privileged estimator defined in* (8)*,* (7)*,*

$$\|\hat{\beta}^\top \hat{\Phi}^{\gamma,\hat{d}_T}(X_T) - Y\|_{L^2(p)} \leq \epsilon$$

and

$$\forall t = 1, ..., T-1 : \|\hat{A}_t^\top \hat{\Phi}^{\gamma,\hat{d}_t}(X_t) - \hat{\Phi}^{\gamma,\hat{d}_{t+1}}(X_{t+1})\|_{L^2(p)} \leq \epsilon,$$

Then, further assume that the largest eigenvalue $\lambda_{\max}(\hat{A}_t) \leq C$ for any t and $\|\hat{\beta}\| \leq C$. Then, with probability at least $1-\delta$,

$$\|(\hat{A}_1 \cdots \hat{A}_{T-1}\hat{\beta})^\top \hat{\Phi}^{\gamma,\hat{d}_1}(X_1) - Y\|_{L^2(p)} \leq TC^T\epsilon$$

Proof. Let $\|\cdot\| = \|\cdot\|_{L^2(p)}$. Then, letting $\hat{\Phi}^t = \hat{\Phi}^{\gamma,\hat{d}_t}$, and applying a union bound to each of the T (ϵ,δ)-assumptions, and a series of Cauchy-Schwarz inequalities,

$$
\begin{aligned}
&\|(\hat{A}_1 \cdots \hat{A}_{T-1}\hat{\beta})^\top \hat{\Phi}^1(X_1) - Y\| \\
&= \|(\hat{A}_1 \cdots \hat{A}_{T-1}\hat{\beta})^\top \hat{\Phi}^1(X_1) - \hat{\beta}^\top \hat{\Phi}^T(X_T) + \hat{\beta}^\top \hat{\Phi}^T(X_T) - Y\| \\
&\leq \|(\hat{A}_1 \cdots \hat{A}_{T-1}\hat{\beta})^\top \hat{\Phi}^1(X_1) - \hat{\beta}^\top \hat{\Phi}^T(X_T)\| + \underbrace{\|\hat{\beta}^\top \hat{\Phi}^T(X_T) - Y\|}_{\leq \epsilon} \\
&\leq \|(\hat{A}_1 \cdots \hat{A}_{T-1}\hat{\beta})^\top \hat{\Phi}^1(X_1) - \hat{\beta}^\top \hat{A}_{T-1}^\top \hat{\Phi}^{T-1}(X_{T-1}) \\
&+ \hat{\beta}^\top (\hat{A}_{T-1}^\top \hat{\Phi}^{T-1}(X_{T-1}) - \hat{\Phi}^T(X_T))\| + \epsilon \\
&\leq \|(\hat{A}_1 \cdots \hat{A}_{T-1}\hat{\beta})^\top \hat{\Phi}^1(X_1) - (\hat{A}_{T-1}\hat{\beta})^\top \hat{\Phi}^{T-1}(X_{T-1})\| \\
&+ \underbrace{\|\hat{\beta}^\top (\hat{A}_{T-1}^\top \hat{\Phi}^{T-1}(X_{T-1}) - \hat{\Phi}^T(X_T))\|}_{\leq C\epsilon} + \epsilon \\
&\ldots \\
&\leq TC^T\epsilon\,.
\end{aligned}
$$

□

Remark 1. Proposition 2 shows that a privileged learner with random ReLU features can be turned into a universally consistent estimator of any (noiseless) continuous function of X_1 by letting each time step have its own random feature representation of appropriate size and adding a norm constraint to each linear transformation. The construction in (8), (7) deviates from Algorithm 1 primarily in that the random feature representations used at each time step are different, but the overall structure is maintained: At training, predictions of Y are made from an embedding $\hat{Z}_T = \hat{\Phi}^T(X_T)$ of X_T, predictions of $\hat{Z}_T$ are made from an embedding $\hat{Z}_{T-1} = \hat{\Phi}^{T-1}(X_{T-1})$, and so on. At test time, the transition matrices $\hat{A}_1, ..., \hat{A}_T, \hat{\beta}$ are multiplied and applied to $\hat{Z}_1 = \hat{\Phi}^1(X_1)$. We conjecture that a similar argument can be applied to the construction in Algorithm 1 by letting the dimension of each time step approach ∞.

C Compounding bias

We can describe the compounding bias of the LuPTS estimator due to a biased representation $\hat{\Phi}$, in comparison with the standard OLS estimator, by propagating the error in $\hat{\Phi}$ through the estimates. Assume that

$$Y = \theta^\top \Phi(X_1) + \epsilon = (A_1 \cdots A_{T-1}\beta)^\top \Phi(X_1) + \epsilon'$$

Then, let $Z_t = \Phi(X_t)$ and for an estimate $\hat{\Phi}$, assumed for simplicity to have the same dimension, $\hat{d} = d$,

$$\hat{Z}_t = \hat{\Phi}(X_t) = \Phi(X_t) + R_t$$

where R_t is the residual w.r.t. Φ. Let bold-face variables indicate multi-sample equivalents of all variables. Further, define $\Sigma_t = \mathbf{Z}_t^\top \mathbf{Z}_t$ and $\hat{\Sigma}_t = \hat{\mathbf{Z}}_t^\top \hat{\mathbf{Z}}_t$.

Fitting $\hat{\theta}$ to $\hat{\Phi}$ using the classical learner (OLS) yields an estimate

$$\hat{\theta}_c = \hat{\Sigma}_1^{-1} \hat{\mathbf{Z}}_1^\top \mathbf{Y}$$

Now, define $\Omega_t = \mathbf{R}_t^\top \hat{\mathbf{Z}}_t + \mathbf{Z}_t^\top \mathbf{R}_t$ and we have

$$
\begin{aligned}
\hat{\theta}_c &= (\Sigma_1 + \Omega_1)^{-1} (\mathbf{Z}_1 + \mathbf{R}_1)^\top \mathbf{Y} \\
&= (\Sigma_1^{-1} + \Delta_1)(\mathbf{Z}_1 + \mathbf{R}_1)^\top \mathbf{Y} \\
&= \hat{\theta}_c^* + (\Delta_1 \hat{\mathbf{Z}}_1^\top + \Sigma_1^{-1} \mathbf{R}_1^\top)\mathbf{Y}
\end{aligned}
$$

where $\Delta_1 = -\Sigma_1^{-1}\Omega_1(\Sigma_1 + \Omega_1)^{-1}$, $\hat{\theta}_c^*$ is the OLS estimate of θ for the true Φ and the second line follows from the Woodbury matrix identity. The norm of Δ_1 is related to the condition number of Σ_1. The expectation of the first term is θ, and the expectation of the remaining terms is the bias.

Now, we can do the same thing for the privileged estimator. Let's start with $T = 2$.

$$\begin{aligned}\hat{\theta}_p &= \hat{\Sigma}_1^{-1}\hat{\mathbf{Z}}_1^\top\hat{\mathbf{Z}}_2\hat{\Sigma}_2^{-1}\hat{\mathbf{Z}}_2^\top Y \\ &= (\Sigma_1^{-1}+\Delta_1)(\mathbf{Z}_1+\mathbf{R}_1)^\top(\mathbf{Z}_2+\mathbf{R}_2)(\Sigma_2^{-1}+\Delta_2)(\mathbf{Z}_2+\mathbf{R}_2)^\top\mathbf{Y} \\ &= \hat{\theta}_p^* + \hat{A}_1(\Sigma_2^{-1}\mathbf{R}_2^\top + \Delta_2\hat{\mathbf{Z}}_2^\top)\mathbf{Y} + (\Sigma_1^{-1}\hat{\mathbf{Z}}_1^\top\mathbf{R}_2 + \Sigma_1^{-1}\mathbf{R}_1^\top\hat{\mathbf{Z}}_2 + \Delta_1\hat{\mathbf{Z}}_1^\top\hat{\mathbf{Z}}_2)\hat{\beta}\ .\end{aligned}$$

Thus, the difference in bias between the two estimators is

$$\begin{aligned}\mathbb{E}[\hat{\theta}_c - \hat{\theta}_p] = &\underbrace{\mathbb{E}[\hat{\theta}_c^* - \hat{\theta}_p^*]}_{=0} \\ &+ \mathbb{E}[(\Delta_1\hat{\mathbf{Z}}_1^\top + \hat{\Sigma}_1^{-1}\mathbf{R}_1^\top)\mathbf{Y} - \hat{A}_1(\Delta_2\hat{\mathbf{Z}}_2^\top + \hat{\Sigma}_2^{-1}\mathbf{R}_2^\top)\mathbf{Y} \\ &- (\Sigma_1^{-1}\hat{\mathbf{Z}}_1^\top\mathbf{R}_2 + \Sigma_1^{-1}\mathbf{R}_1^\top\hat{\mathbf{Z}}_2 + \Delta_1\hat{\mathbf{Z}}_1^\top\hat{\mathbf{Z}}_2)\hat{\beta}]\end{aligned}$$

More generally, we can express this difference recursively as below.

Proposition 3. *Let $\hat{\theta}_p \coloneqq \hat{A}_1 \cdots \hat{A}_T\hat{\beta}$ be a privileged estimator using a linearly biased representation $\hat{\Phi}$, and let $\hat{A}_t^*$ be the same estimator using an unbiased representation Φ^*. Then, the bias of $\hat{\theta}_p$ is*

$$\mathbb{E}[\hat{\theta}_p - \theta] = \mathbb{E}[\hat{\theta}_p - \hat{\theta}_p^*] = \mathbb{E}[E_T\hat{\beta}^* + (\hat{A}_1 \cdots \hat{A}_T)(\hat{\beta} - \hat{\beta}^*)]\ ,$$

where E_t is the compounded error in transition dynamics, computed recursively as follows

$$E_t \coloneqq (\hat{A}_1 \cdots \hat{A}_t) - (\hat{A}_1^* \cdots \hat{A}_t^*) = E_{t-1}\hat{A}_t^* + (\hat{A}_1 \cdots \hat{A}_{t-1})(\hat{A}_t - \hat{A}_t^*)\ ,$$

with $E_0 = 0$. In the worst case, the bias of $\hat{\theta}_p$ grows exponentially with T.

D Privileged time series representation learners

We expand on the description of the greedy representation learner (GRL) described as a special case of CRL in Section 3.3. To avoid the information loss of SRL, we consider its conceptual opposite, using privileged time series information *only* to predict the outcome. To do this, a linear output layer $\hat{\theta}_t^\top\hat{Z}_t$ is used to predict Y at every time step t. Recall that, by Assumption 1, the expected outcome is linear in the latent state at *any* time step. The method is related to multi-view learning, in which prediction of the same quantity is made from multiple "views" (Zhao et al., 2017). We dub the model *greedy privileged representation learner* (GRL), which minimizes the objective

$$\mathcal{L}_{\text{GRL}}(\hat{\Phi}, \{\hat{\theta}_t\}) \coloneqq \frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T} w_t\left\|\hat{\theta}_t^\top\hat{\Phi}(x_{i,t}) - y_i\right\|_2^2. \tag{9}$$

During inference this algorithm returns $\hat{h}_{\text{P}}(\cdot) = \hat{\theta}_1^\top\hat{\Phi}(\cdot)$. Compared to the objective of CRL in 6, we introduce an additional hyperparameter $\lambda \in (0, 1)$ to place more weight on the loss term that is relevant at inference time. As a consequence, we choose $w_t = \lambda$ for $t = 1$ and $w_t = 1 - \lambda$ otherwise. We expect GRL to have less bias than SRL, but higher variance since less structure is imposed on the representation $\hat{\Phi}$.

E Experiment setup & data processing

E.1 Detailed experiment setup

In the following we give a detailed description of the experimental setup used to obtain the results presented in Section 4 as well as the additional results that are part of this section. For a given data set, we select a combination of training set sizes and sequence length. For each unique combination of these parameters the models of interest are trained repeatedly with different random sampling. For each repetition the data is split into a train and a test set randomly before hyperparameter tuning and model training are performed. At last each model's predictions on the test set are scored by computing the coefficient of determination R^2. On synthetic data the test set contains 1000 samples. In the case of real-world data, where samples are limited, we test on 20% of all available data.

Hyperparameter	Description	Used in Algorithm	Value Range
n_{RF}	number of random features	all random feature methods	$[0.05m, 0.8m]$
γ_{RRF}	bandwith parameter	Random ReLU methods	$[0.01, 10]$
γ_{RFF}	bandwith parameter	Random Fourier methods	$[0.001, 0.1]$
λ	loss function parameter	GRL & CRL	$[0, 1]$

Table 1: Overview of all hyperparameter determined by hyperparameter tuning. m denotes the number of samples, meaning that n_{RF} is chosen from different ranges depending on the size of the training set.

The preprocessing used for real-world data and the generation procedure of synthetic data is unique to each data set. We refer to the data set specific subsections for detailed descriptions of how each data set is processed. During the experiments each model uses standardized data for training and inference. To perform the data rescaling we use the StandardScaler implementation that is part of scikit-learn (Pedregosa et al., 2011).

Hyperparameter tuning. The tuning of hyperparameters is carried out for each repetition and is implemented using random search and five-fold cross-validation. Each hyperparameter is sampled from a fixed interval of possible values. An overview of the ranges of different hyperparameters determined through random-search is provided in Table 1. For the experiments with variants of generalized LuPTS we sample ten sets of hyperparameters before retraining on all training data using the best set of parameters. For the representation learners we merely sample five values for λ.

Neural network training. The training of neural networks involves many choices and hyperparameters. We choose PyTorch's implementation (Paszke et al., 2019) of the Adam optimizer (Kingma and Ba, 2017) to train the representation learning models. If not specified otherwise the results shown in this project are obtained using a learning rate of 0.0001, a batch size of 30, leaky ReLU activations and a maximum of 1500 training epochs. In the case of neural network models, the sample sizes reported as part of the experiments denote the combined size of the training and validation set, where the validation set contains 20% of those samples. We use early stopping during the training process by keeping track of the validation loss. If a model does not improve the validation loss over a waiting period of 200 epochs we stop training early and set the network parameters to the values that obtained the lowest validation loss up until that point. In order to make sure that results are not dependent on the specific choice of the parameters just described, we performed additional experiments with different parameter choices. The results were found to be robust to small changes in these parameters.

Generalized distillation. In order to compare our algorithms to the alternative of using generalized distillation for privileged time series as presented by Hayashi et al. (2019), we implemented a model that (i) produces hypotheses of the same class as our other algorithms and (ii) that adopts the learning paradigm of a student model incorporating soft targets produced by a teacher model into its loss function. For tabular data our teacher model is a multi-layer-perceptron (MLP) with $T * k$ input neurons such that all $\{X_t\}$ are concatenated and then used as input. The teacher MLP makes use of five hidden layers, each consisting of 100 neurons. In the case of the image data generated by **Clocks-LGS**, the teacher uses an implementation of LeNet-5 on all variables X_t (while sharing the encoder parameters) before concatenating the 25-dimensional output of this encoder from different time steps. This combined representation is then processed by an MLP with a single hidden layer with 25 neurons. The loss function used for the student model producing the estimate $\hat{h}_{\text{C}}$ is architecturally identical to the classic representation learner used in each of the experiments. When training the student model the mean squared error on the data and the error corresponding to the soft targets of the teacher model are combined via a hyperparameter λ:

$$\mathcal{L}_{GD} \coloneqq \lambda \frac{1}{m} \sum_{i=1}^{m} \|\hat{h}_{\text{C}}(x_i) - y_i\|_2^2 + (1-\lambda) \frac{1}{m} \sum_{i=1}^{m} \|\hat{h}_{\text{C}}(x_i) - \hat{h}_{\text{teacher}}(x_i)\|_2^2$$

The hyperparameter is determined via hyperparameter tuning in all repetitions as described for other hyperparameters in this section. All other training procedures follow the same logic as described above.

Resources. For the training of the representation learning algorithms we use a cluster of graphics processing units (GPUs) in order to reach the number of experiment repetitions required for our work. A single experiment like shown in Figure 5c takes several hours on 100 NVIDIA Tesla T4 GPUs. While the random feature methods do not require GPU training, they still require hyperparameter tuning which is why we compute results such as presented in Figure 4 on many CPU cores in parallel. While the experiments on neural networks cannot reasonably be reproduced on a single desktop machine, this is still possible within a few days for the random feature methods.

Latent variable recovery and SVCCA. In order to assess to what extent a representation has been found that is linearly related to the true latent variables we use SVCCA as described by Raghu et al. (2017), meaning we first use PCA retaining at least 99% of variation before then applying CCA. For the visualization shown in Figure 7b we construct a grid of points (150 $times$ 150) around the origin and assign each point a unique color. Then we compute an image using the observation generating function $\Psi : \mathcal{Z} \to \mathcal{X}$ of the **Clocks-LGS** data set for each point. The observations are passed to the encoder $\hat{\Phi}$ of the two representation learners producing estimates of the latent variables. We then map the estimates to the ground truth linearly using SVCCA before plotting the result.

E.2 Alzheimer progression

To test our algorithms on the task of predicting the progression of Alzheimer's disease (AD) we use an anonymized data set obtained through the Alzheimer's Disease Neuroimaging Initiative (**ADNI**) (ADNI, 2004) under the LONI Research License. The initiative is large multi-site research study on the brains of over 2000 AD patients which collects many features such as genetic, imaging and biospecimen biomarkers. The data consists of measurements taken every 3 months with some observations missing. The outcome of interest in our experiments is the Mini Mental State Experiment (MMSE) score 48 months after the first measurement(Galea and Woodward, 2005). Privileged information are the measurements taken between those time points, at 12, 24 and 36 months into the program.

Data processing. The processing procedure used in this project is borrowed directly from the work of Karlsson et al. (2022). There is a large amount of missing information in the ADNI data set. The missingness varies with the time of when measurements were taken. Further some subjects were not present at some of the follow-up examinations. To deal with the missingness patients without an observation for the final follow up (the outcome Y) are excluded from our experiments. Further, we also require that patients are present at all intermediate time steps (12, 24 and 36 months after the first measurement) which we use as privileged information. We one-hot encode categorical features and exclude features for which more than 70% of the observations are missing. To deal with the remaining missing values, mean imputation is used. Due to the filtering that we apply as a result of the missing data we only obtain 502 suitable sequences that we can use for our experiments.

E.3 Traffic data

The **Traffic** data set (Hogue, 2012) obtained through the UCI machine learning repository (Dua and Graff, 2017) contains hourly measurements of the traffic volume as well as weather features and holiday information. The raw data contains 48.204 records. An overview of all available features is given in Table 2.

Data processing. We noticed extreme outliers in the data set as well as implausible numerical values for the temperature and rain features. Further, records for some of the hours of the timeframe (2012 - 2018) covered by this data set are missing. To deal with the extreme outliers we calculate the mean and standard deviation of each feature and remove records which contain values that are further than six standard deviations from the mean of a particular feature. We also remove a feature entirely if there is no variation left after this filtering. This is the case for the snowfall feature as snow is very rare in Minneapolis. From the date and time of each record we calculate the weekday which we add

Feature	Type	Description
Date Time	Timestamp	date and time (CST)
Holiday	String	name of holiday if applicable
Weather Description	String	brief free text description of the weather
Weather Main	Categorial	contains categories like clear, clouds, or rain
Rain_1h	Numerical	rain in $\frac{L}{h\ m^2}$
Snow_1h	Numerical	snow in $\frac{L}{h\ m^2}$
Temp	Numerical	temperature in Kelvin
Traffic Volume	Numerical	hourly reported westbound traffic volume

Table 2: Features available in the **Traffic** data set.

as a one-hot encoded feature and also represent the hour of the day $h \in \{0, 1, \dots 23\}$ as two separate periodic features given by

$$t_{\text{periodic}} = \left[\sin\left(\frac{2\pi \cdot h}{24}\right),\ \cos\left(\frac{2\pi \cdot h}{24}\right) \right] . \tag{10}$$

This ensures that a timestamp just before midnight produces similar features compared to just after midnight. We one-hot encode the holiday information, making no difference between different types of holidays, and make this feature persist over a full calendar day. In the original data set the holiday information is only specified for the first hour of the day. The column Weather_Main contains some weather conditions that are very rare, such as smoke and squall. As a consequence we group the different conditions before encoding them as binary variables. In particular we make drizzle, rain and squall one single feature while also grouping together fog, haze, mist and smoke as they all affect visibility.

time series selection. After this preprocessing that leaves only numerical values and one-hot encoded categorical values we group the data together as time series used for the experiments. In order to do so we specify a desired sequence length $T + 1$ and a sequence step size in hours. With this information we iterate through the data set assembling time series with i) no values missing ii) the correct length and step size and iii) at least a seven hour gap between each pair of sequences. The third condition is introduced to make sure one does not end up with very similar cases (for short sequences in particular) in training and test set.

E.4 Square-Sign

The **Square-Sign** data set serves as a test environment for learning from privileged time series information where one can assure the conditions necessary for Assumption 1 to hold. In particular this means creating a linear-Gaussian system which remains unobserved and combining it with an observation generating function $\psi : \mathcal{Z} \to \mathcal{X}$.

Latent linear-Gaussian system. The first component that makes up the generation process for **Square-Sign** (and **Clocks-LGS**) is the linear-Gaussian system which is latent, just as depicted on the right side of Figure 1b with $Z_t \in \mathbb{R}^d$. The first step in the data creation process is sampling each of the d components of Z_0 from $\mathcal{N}(0, 5)$. Then the subsequent latent variables Z_{t+1} are computed as

$$Z_{t+1} \coloneqq A_t^\top Z_t + \epsilon,\ \epsilon \in \mathbb{R}^d,\ \epsilon_j\ \mathcal{N}(0, 1)\ .$$

For the outcome we use the same form but with different dimensionality:

$$Y \coloneqq \beta^\top Z_T + \epsilon_y,\ \epsilon_y \in \mathbb{R}^q$$

Off-diagonal elements of the transition matrices $A_t \in \mathbb{R}^{d \times d}$ are sampled from a Normal distribution $\mathcal{N}(0, 0.2)$ while the diagonal elements are set to one. In a second step we compute the spectral radius of the randomly created matrices A_t via eigenvalue decomposition, obtaining the components $U \Lambda U^\top$. We then set the spectral radius to a predefined value $\lambda_{max} = 1.3$ and reassemble the matrix as

$$A_t \leftarrow U \frac{\lambda_{max}}{\lambda_s} \Lambda U^\top\ .$$

The coefficients of β are drawn from the same normal distribution as the ones of A_t but undergo no further changes.

[-8.75, 1.71] [8.12, -3.06] [-12.30, -17.37]

Figure 8: Pairs of corresponding latent vectors Z_t and images as generated in **Clocks-LGS**.

Observation generating function. As the dimensionality d of the latent space $\mathcal{Z}$ is not fixed we use an observation generating function that is not restricted to a specific value of d. For each element in $Z_t \in \mathcal{Z} = \mathbb{R}^d$ we create two elements in $X_t \in \mathcal{X} = \mathbb{R}^{2d}$ by denoting its sign separately from its square. This gives the following nonlinear observation generating function:

$$X_t \coloneqq \psi(Z_{(t)}) = [Z_{(t,1)}^2, \mathrm{sgn}(Z_{(t,1)}), \ldots, Z_{(t,d)}^2, \mathrm{sgn}(Z_{(t,d)})]^\top$$

E.5 Clocks-LGS

This data set serves the purpose of testing our algorithms on a different modality with high dimensional data. In particular the idea was to use image data as this is a domain where neural networks have been very successful. For this reason we combine a latent dynamical system with an image generation process which we explain in detail in this section.

Latent linear-Gaussian system. We use exactly the same setup as we do for the **Square-Sign** latent dynamical system as described in Section E.4. The only difference here is the dimensionality of the latent variables, transition matrices and the outcome. For **Clocks-LGS** we generally have $d = 2$ and $q \in \{1, 2\}$.

Image generation. The second part of **Clocks-LGS** is creating images from two dimensional latent vectors $Z_t = [Z_t^{(1)}, Z_t^{(2)}]^\top$. The goal was to keep it the process simple while using small black and white images of 28×28 pixels. In addition we wanted each image to have no ambiguity with respect to the latent state it represents. We represent the first component by a clock hand mounted at the image center. One can think of $Z_t^{(1)}$ as the angle in radian, meaning the hand points straight up for $Z_t^{(1)} = 0$ or $Z_t^{(1)} = 2\pi$ and straight down for $Z_t^{(1)} = \pi$. To visualize a full rotation we increase the size of the cirle around the image center in discrete steps for each mutliple of 2π. For negative values the circle is empty (black) while it is filled (white) for positive values. For the second component we make use of the same logic but instead of a clock hand, we only use a circle that orbits the image center. The two hands cannot obscure each other as the orbiting cirle uses a larger radius. Figure 8 shows three examples of pairs of corresponding latent vectors and generated images.

E.6 $\mathbf{PM}_{2.5}$ air quality

Due to health concerns the air quality in Chinese cities has become an important topic. The $\mathbf{PM}_{2.5}$ data set contains hourly meteorologic information and the concentration of small particles ($\mathrm{PM}_{2.5}$) for the cities Beijing, Shanghai, Guangzhou, Chengdu and Shenyang (Liang et al., 2016). The individual features available for all cities are listed in Table 3. In addition to the features listed, the data includes the date and time of each record. Just like in the preprocessing of **Traffic** we compute a periodic time feature using expression 10 to represent the time of day of each record. For each numerical feature we calculate the mean and standard deviation and remove rows with values that are more extreme than six standard deviations from the mean. We also remove rows with missing categorical features, which are then represented as one-hot vectors. Apart from differences in the preprocessing we consider the same prediction task as Karlsson et al. (2022) which is predicting the future particle concentration for a fixed time horizon given current observations.

Feature	Type	Description
season	Numerical	season (1 to 4) of the data in this row
PM	Numerical	$PM_{2.5}$ particle concentration in $\mu g/m^3$
DEWP	Numerical	dew point in °C
TEMP	Numerical	air temperature in °C
HUMI	Numerical	humidity in %
PRES	Numerical	atmospheric pressure in hPa
cbwd	Categorical	combined wind direction in {N,W,S,E, NW, SW, NE, SE}
Iws	Numerical	cumulated wind speed in m/s
precipitation	Numerical	hourly precipitation in mm
Iprec	Numerical	cumulated precipitation in mm

Table 3: Features available as part of the $\mathbf{PM}_{2.5}$ data set on the air quality of five large Chinese cities.

F Additional experiment results

In the course of this section we present a larger scope of our experimental results. We demonstrate the predictive accuracy of the algorithms introduced in Sections 3.2 and 3.3 in terms of the mean coefficient of determination R^2 over different settings on five data sets. The variation of the results over repetitions is represented by the shaded areas in the visualizations, which denotes one standard deviation above and below the mean value. We also show empirically that the bias of generalized LuPTS increases with the number of privileged time steps when a poor representation function is used. This can be seen in Figure 16a where we test privileged and classical learners over 500 different systems of the **Square-Sign** type over different sequence lengths. In addition we further illustrate how privileged information can improve the recovery of latent variables of the data generating process by providing more visualizations in the style of Figure 7a on the **Clocks-LGS** and **Square-Sign** data set. These can be found in Figures 13 and 16.

In the following we first provide the experiment details for visualizing the phase transition of generalized LuPTS as seen in Figure 2. In the subsections thereafter the material is organized by data set.

F.1 Two regimes of generalized LuPTS

As seen in Figure 2 and demonstrated by Proposition 1, generalized LuPTS becomes equivalent to the corresponding classical learner when the number of features $\hat{d}$ is larger than the number of samples m. In the following we provide the experiment details that led to Figure 2.

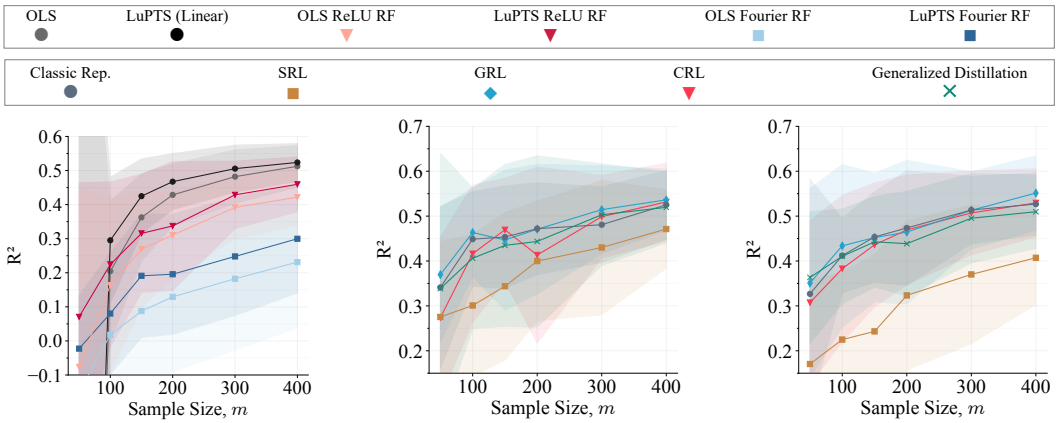
In order to evaluate the dependency on the number of features we used linear LuPTS and OLS on a synthetically generated linear-Gaussian system as displayed in Figure 1a. We use the same setup as for the latent dynamics in **Square-Sign** but without a nonlinear observation generating function. For each number of features $k = \hat{d} = d$ we sample 50 such systems with different dynamics, each producing a training set with $m = 100$ and test set of 1000 samples. The systems are all configured with $T = 3$ and $q = 10$. We train and score both estimators on all of the data generating systems before computing the mean coefficient of determination over all systems with the same number of features.

F.2 Alzheimer progression

F.3 Traffic data

F.4 Clock_LGS

F.5 Square-Sign

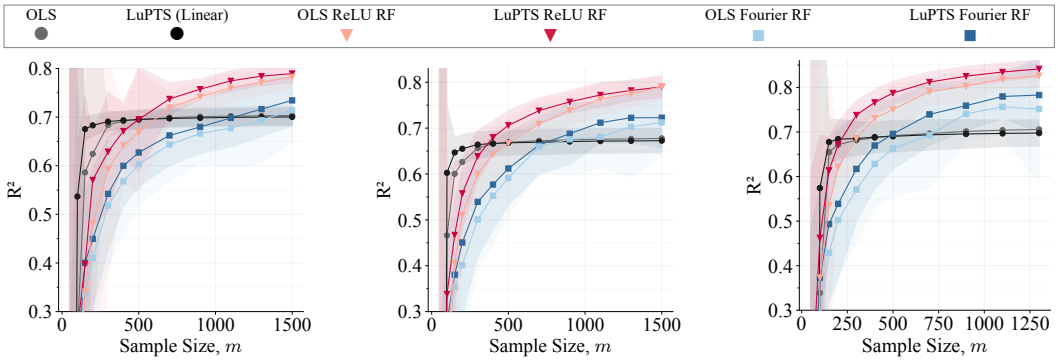


(a) Generalized LuPTS on the reduced setting with a single privileged time point, $T = 2$.

(b) Representation learners and generalized distillation on the reduced setting with one privileged time step, $T = 2$.

(c) Representation learners and generalized distillation on the full setting with three privileged time steps, $T = 4$.

Figure 9: Generalized LuPTS and the representation learning algorithms tested in terms of their predictive accuracy on different settings of the **ADNI** prediction task. Each experiments are based on 25 repetitions while the random feature experiment consists of 60 repetitions.



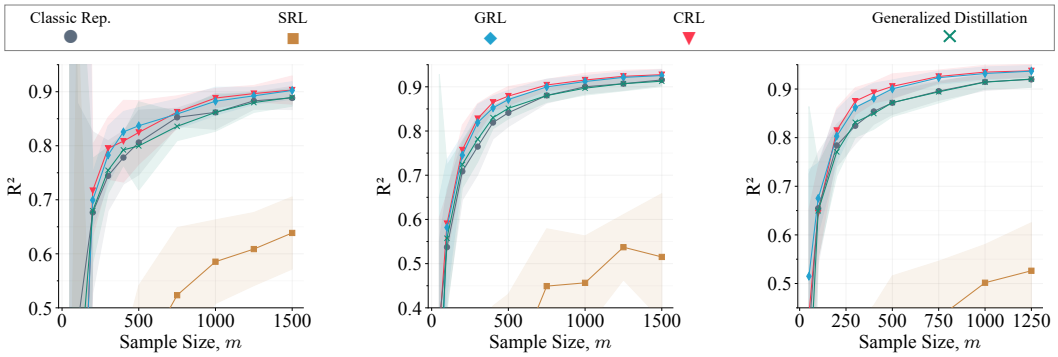
(a) $T = 2$.

(b) $T = 3$.

(c) $T = 4$.

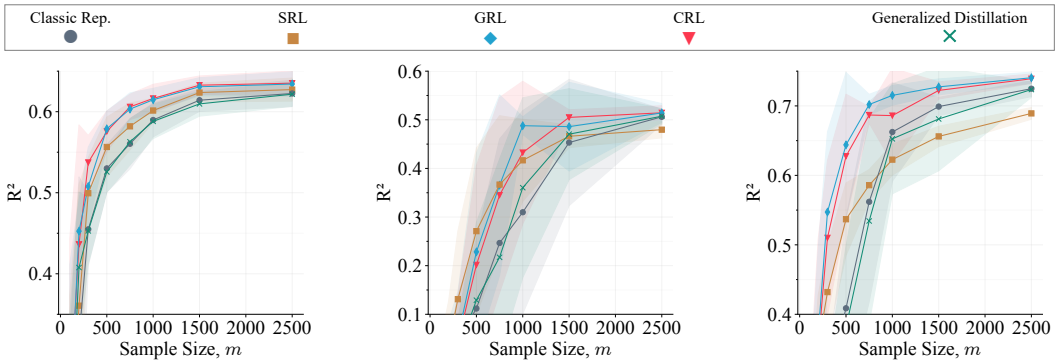
Figure 10: Prediction accuracy of the different variants of generalized LuPTS on **Traffic** with varying sequence lengths. Based on 60 repetitions and four hour steps in each time series.

F.6 $\mathbf{PM}_{2.5}$ air quality



(a) $T = 2$. (b) $T = 3$. (c) $T = 4$.

Figure 11: Prediction accuracy of the different representation learning algorithms and generalized distillation on **Traffic** with varying sequence lengths. The results are based on 25 repetitions and four hour steps in each time series.



(a) $T = 2$. (b) $T = 3$. (c) $T = 5$.

Figure 12: Prediction accuracy of the representation learning algorithms and generalized distillation on **Clocks-LGS** with a two dimensional output $q = 2$ and varying sequence lengths. Each experiment is based on 25 repetitions.

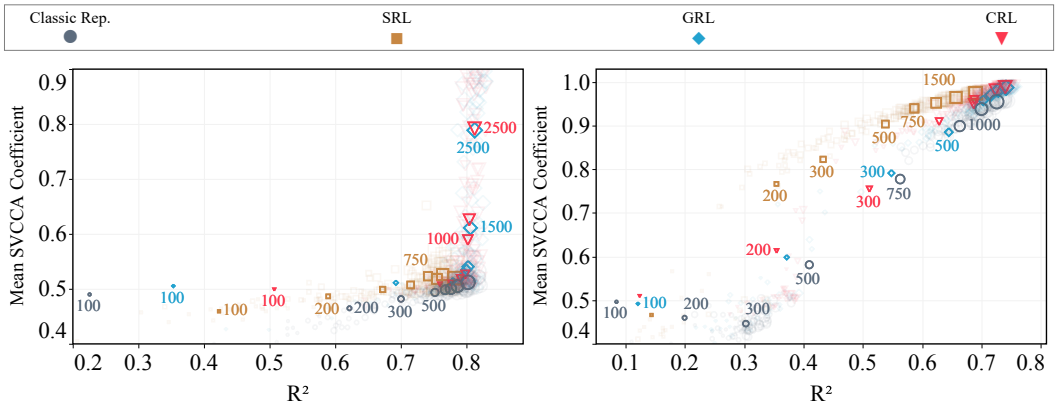
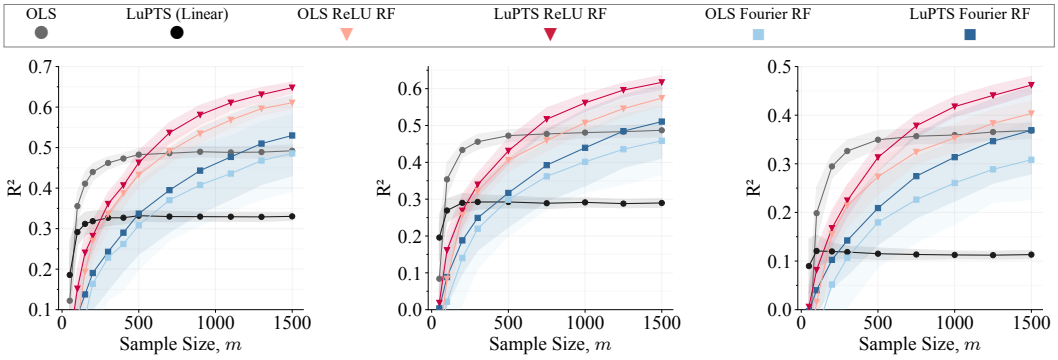


Figure 13: Mean SVCCA coefficients and R^2 for the representation learners on the **Clocks-LGS** prediction task. The experiment was set up for sequences of length six ($T = 5$), with different outcomes of different dimensionality: $q = 1$ on the left and $q = 2$ on the right. Just as in Figure 7a solid marks represent the mean over 25 repetitions while the faded marks denote the individual training runs. The annotations refer to the size of the training sets.

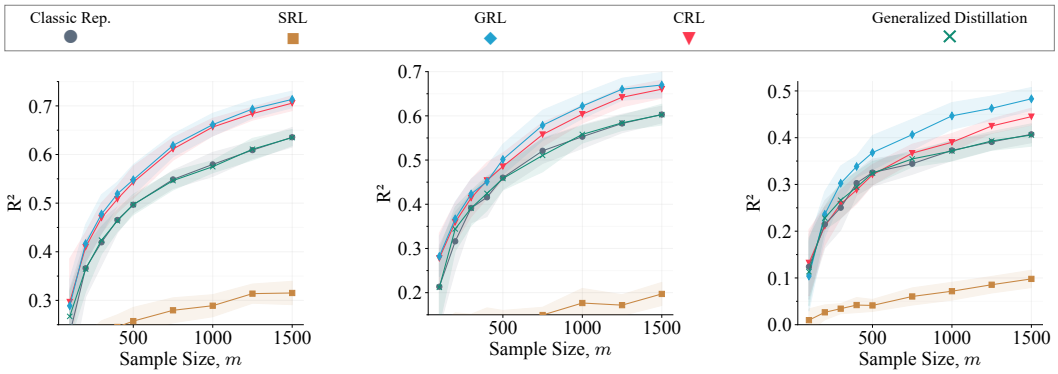


(a) Sequences of length six, four privileged time points, $T = 5$.

(b) Sequences of length seven, five privileged time points, $T = 6$.

(c) Sequences of length eight, six privileged time points, $T = 7$.

Figure 14: Predictive accuracy (R^2) of the different variants of generalized LuPTS applied to the prediction task offered by **Square-Sign**. The DGP was configured with different sequence lengths and the experiments represent 60 repetitions.

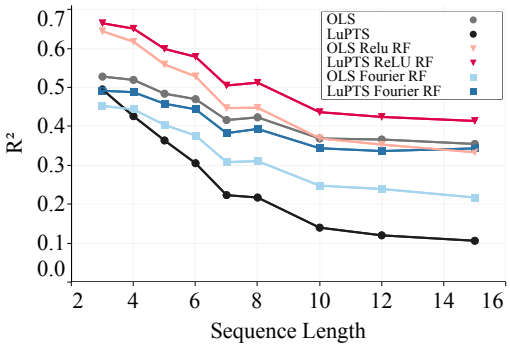


(a) Sequences of length six, four privileged time points, $T = 5$.

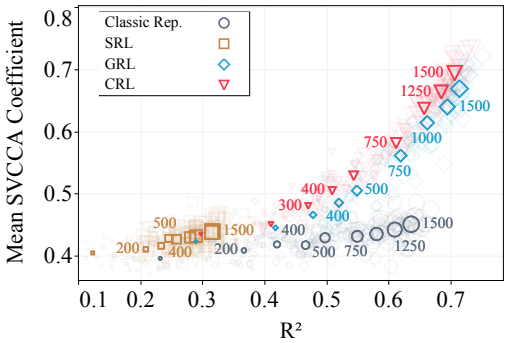
(b) Sequences of length seven, five privileged time points, $T = 6$.

(c) Sequences of length eight, six privileged time points, $T = 7$.

Figure 15: Predictive accuracy of the different representation learners introduced in Section 3.3 when applied to the **Square-Sign** data set for different sequence lengths. The experiments are based on 25 repetitions.

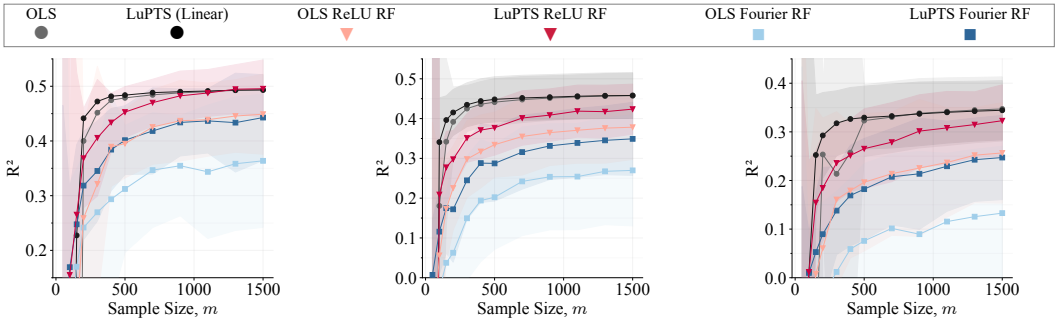


(a) Mean R^2 over 500 repetitions for different sequence lengths on **Square-Sign** ($d = 10$, $q = 3$) with a sample size of $m = 1000$. Each run is performed on a different set of randomly sampled transition dynamics.

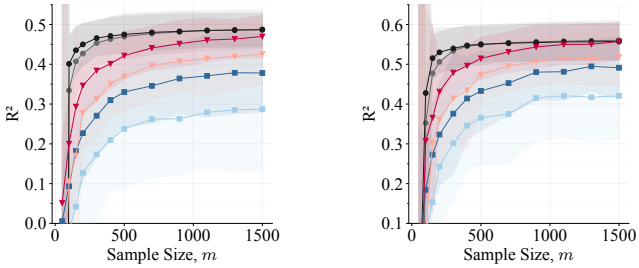


(b) Mean SVCCA coefficients and R^2 for the representation learners on the **Square-Sign** data set configured with ($T = 5, d = 10, q = 3$). Solid marks represent the mean over 25 repetitions for a fixed training set size while the faded marks denote individual runs. The annotations refer to the number of samples in the training data.

Figure 16: Generalized LuPTS applied on the **Square-Sign** task for different sequence lengths (left panel), reporting the mean coefficient of determination R^2 over different systems. The right panel shows analysis of the predictive accuracy and latent recovery of the representation learners in the style of Figure 7a.

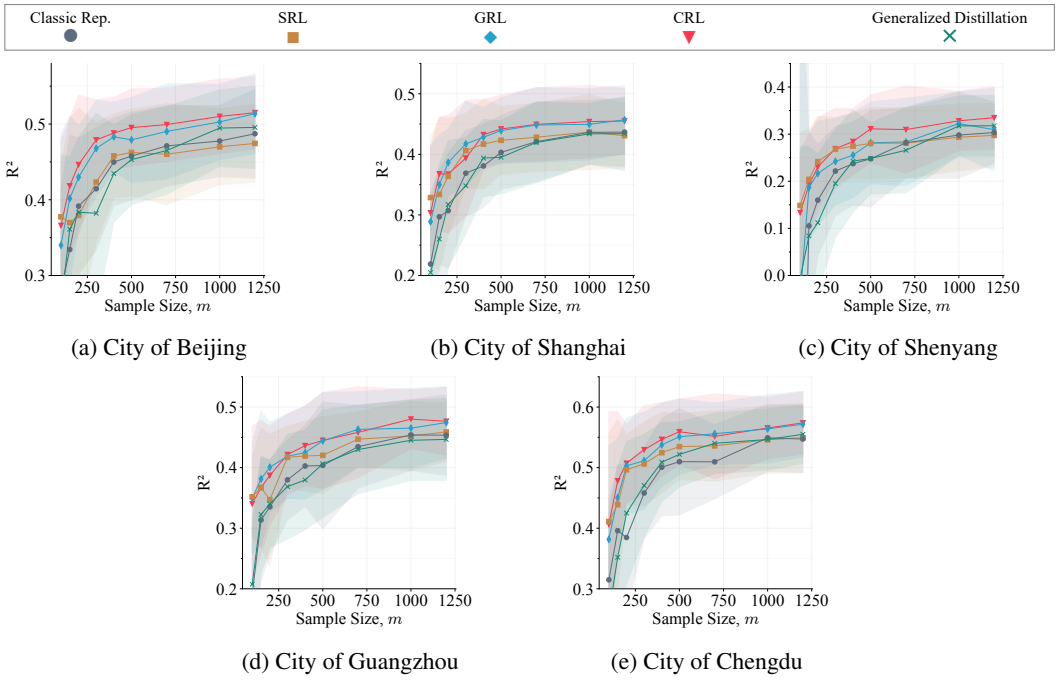


(a) City of Beijing (b) City of Shanghai (c) City of Shenyang



(d) City of Guangzhou (e) City of Chengdu

Figure 17: Predictive accuracy of the different generalized LuPTS variants on the prediction task posed by the $\mathbf{PM}_{2.5}$ data set. All experiments use time series of length five ($T = 4$), where each time step is two hours long. The results were computed based on 60 repetitions.



(a) City of Beijing (b) City of Shanghai (c) City of Shenyang

(d) City of Guangzhou (e) City of Chengdu

Figure 18: Evaluation of the sample efficiency of the represenation learning algorithms of Section 3.3 on the $\mathbf{PM}_{2.5}$ air quality prediction task. All experiments use time series of length five ($T = 4$), where each time step represents two hours. The results were computed based on 25 repetitions.