Disentangling Physical Dynamics from Unknown Factors for Unsupervised Video Prediction

Vincent Le Guen, Nicolas Thome

Disentangling Physical Dynamics from Unknown Factors for Unsupervised Video Prediction

Vincent Le Guen [1,2], Nicolas Thome [2]
[1] EDF R&D, Chatou, France
[2] CEDRIC, Conservatoire National des Arts et Métiers, Paris, France

Abstract

Leveraging physical knowledge described by partial differential equations (PDEs) is an appealing way to improve unsupervised video prediction methods. Since physics is too restrictive for describing the full visual content of generic videos, we introduce PhyDNet, a two-branch deep architecture, which explicitly disentangles PDE dynamics from unknown complementary information. A second contribution is to propose a new recurrent physical cell (PhyCell), inspired from data assimilation techniques, for performing PDE-constrained prediction in latent space. Extensive experiments conducted on four various datasets show the ability of PhyDNet to outperform state-of-the-art methods. Ablation studies also highlight the important gain brought out by both disentanglement and PDE-constrained prediction. Finally, we show that PhyDNet presents interesting features for dealing with missing data and long-term forecasting.

1. Introduction

Video forecasting consists in predicting the future content of a video conditioned on previous frames. This is of crucial importance in various contexts, such as weather forecasting [73], autonomous driving [29], reinforcement learning [43], robotics [16], or action recognition [33]. In this work, we focus on unsupervised video prediction, where the absence of semantic labels to drive predictions exacerbates the challenges of the task. In this context, a key problem is to design video prediction methods able to represent the complex dynamics underlying raw data.

State-of-the-art methods for training such complex dynamical models currently rely on deep learning, with specific architectural choices based on 2D/3D convolutional [40, 62] or recurrent neural networks [66, 64, 67].

To improve predictions, recent methods use adversarial training [40, 62, 29], stochastic models [7, 41], constraint predictions by using geometric knowledge [16, 24, 75] or by disentangling factors of variation [60, 58, 12, 21].

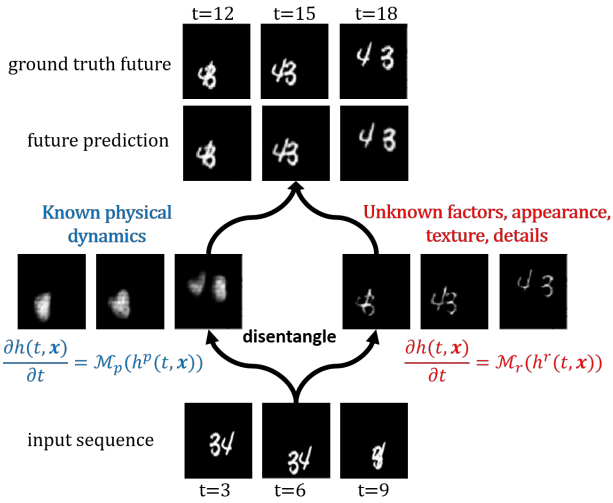


Figure 1. PhyDNet is a deep model mapping an input video into a latent space $\mathcal{H}$, from which future frame prediction can be accurately performed. PhyDNet learns $\mathcal{H}$ in an unsupervised manner, such that physical dynamics and unknown factors necessary for prediction, *e.g.* appearance, details, texture, are disentangled.

Another appealing way to model the video dynamics is to exploit prior physical knowledge, *e.g.* formalized by partial differential equations (PDEs) [11, 55]. Recently, interesting connections between residual networks and PDEs have been drawn [71, 37, 8], enabling to design physically-constrained machine learning frameworks [48, 11, 55, 52]. These approaches are very successful for modeling complex natural phenomena, *e.g.* climate, when the underlying dynamics is well described by the physical equations in the input space [48, 52, 35]. However, such assumption is rarely fulfilled in the pixel space for predicting generalist videos.

In this work, we introduce PhyDNet, a deep model dedicated to perform accurate future frame predictions from generalist videos. In such a context, physical laws do not apply in the input pixel space ; the goal of PhyDNet is to learn a semantic latent space $\mathcal{H}$ in which they do, and are disentangled from other factors of variation required to per-

form future prediction. Prediction results of PhyDNet when trained on Moving MNIST [56] are shown in Figure 1. The left branch represents the physical dynamics in $\mathcal{H}$; when decoded in the image space, we can see that the corresponding features encode approximate segmentation masks predicting digit positions on subsequent frames. On the other hand, the right branch extracts residual information required for prediction, here the precise appearance of the two digits. Combining both representations eventually makes accurate prediction successful.

Our contributions to the unsupervised video prediction problem with PhyDNet can be summarized as follows:

- We introduce a global sequence to sequence two-branch deep model (section 3.1) dedicated to jointly learn the latent space $\mathcal{H}$ and to disentangle physical dynamics from residual information, the latter being modeled by a data-driven (ConvLSTM [73]) method.
- Physical dynamics is modeled by a new recurrent physical cell, PhyCell (section 3.2), discretizing a broad class of PDEs in $\mathcal{H}$. PhyCell is based on a prediction-correction paradigm inspired from the data assimilation community [1], enabling robust training with missing data and for long-term forecasting.
- Experiments (section 4) reveal that PhyDNet outperforms state-of-the-art methods on four generalist datasets: this is, as far as we know, the first physically-constrained model able to show such capabilities. We highlight the importance of both disentanglement and physical prediction for optimal performances.

2. Related work

We review here related multi-step video prediction approaches dedicated to long-term forecasting. We also focus on unsupervised training, *i.e.* only using input video data and without manual supervision based on semantic labels.

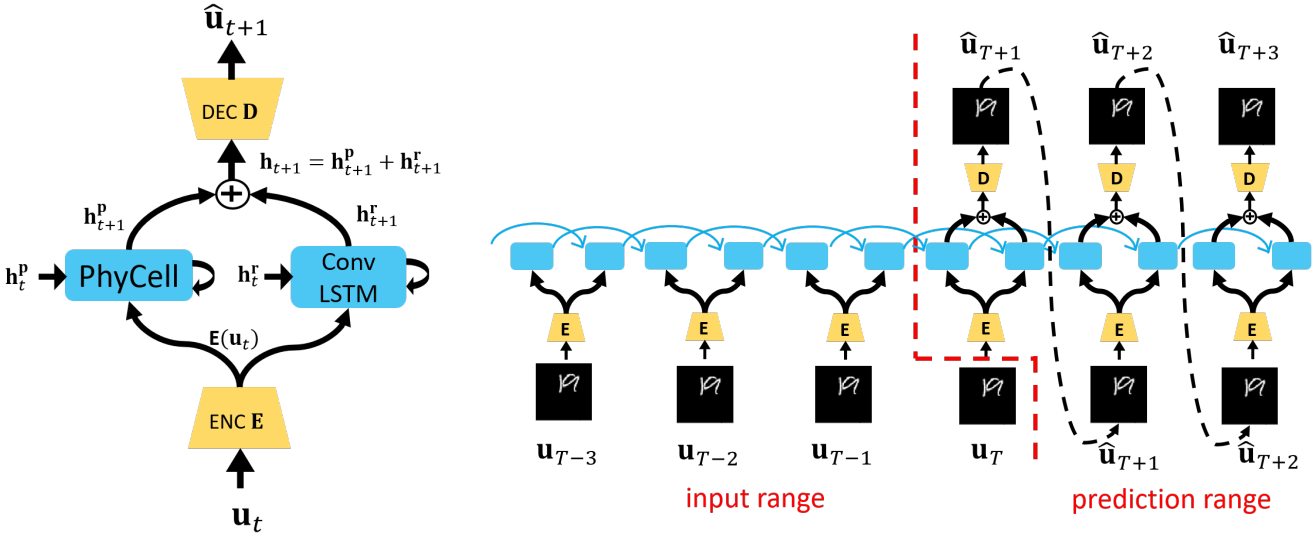
Deep video prediction Deep neural networks have recently achieved state-of-the-art performances for data-driven video prediction. Seminal works include the application of sequence to sequence LSTM or Convolutional variants [56, 73], adopted in many studies [16, 36, 74]. Further works explore different architectural designs based on Recurrent Neural Networks (RNNs) [66, 64, 44, 67, 65] and 2D/3D ConvNets [40, 62, 50, 6]. Dedicated loss functions [10, 30] and Generative Adversarial Networks (GANs) have been investigated for sharper predictions [40, 62, 29]. However, the problem of conditioning GANs with prior information, such as physical models, remains an open question.

To constrain the challenging generation of high dimensional images, several methods rather predict geometric transformations between frames [16, 24, 75] or use optical flow [46, 38, 33, 32, 31]. This is very effective for short-term prediction, but degrades quickly when the video content evolves, where more complex models and memory about dynamics are required.

A promising line of work consists in disentangling independent factors of variations in order to apply the prediction model on lower-dimensional representations. A few approaches explicitly model interactions between objects inferred from an observed scene [14, 27, 76]. Relational reasoning, often implemented with graphs [2, 26, 53, 45, 59], can account for basic physical laws, *e.g.* drift, gravity, spring [70, 72, 42]. However, these methods are object-centric, only evaluate on controlled settings and are not suited for general real-world video forecasting. Other disentangling approaches factorize the video into independent components [60, 58, 12, 21, 19]. Several disentanglement criteria are used, such as content/motion [60] or deterministic/stochastic [12]. In specific contexts, the prediction space can be structured using additional information, *e.g.* with human pose [61, 63] or key points [41], which imposes a severe overhead on the annotation budget.

Physics and PDEs Exploiting prior physical knowledge is another appealing way to improve prediction models. Earlier attempts for data-driven PDE discovery include sparse regression of potential differential terms [5, 52, 54] or neural networks approximating the solution and response function of PDEs [49, 48, 55]. Several approaches are dedicated to a specific PDE, *e.g.* advection-diffusion in [11]. Based on the connection between numerical schemes for solving PDEs (*e.g.* Euler, Runge-Kutta) and residual neural networks [71, 37, 8, 78], several specific architectures were designed for predicting and identifying dynamical systems [15, 35, 47]. PDE-Net [35, 34] discretizes a broad class of PDEs by approximating partial derivatives with convolutions. Although these works leverage physical knowledge, they either suppose physics behind data to be explicitly known or are limited to a fully visible state, which is rarely the case for general video forecasting.

Deep Kalman filters To handle unobserved phenomena, state space models, in particular the Kalman filter [25], have been recently integrated with deep learning, by modeling dynamics in learned latent space [28, 69, 20, 17, 3]. The Kalman variational autoencoder [17] separates state estimation in videos from dynamics with a linear gaussian state space model. The Recurrent Kalman Network [3] uses a factorized high dimensional latent space in which the linear Kalman updates are simplified and don't require computationally-heavy covariance matrix inversions. These methods inspired by the data assimilation community [1, 4] have advantages in missing data or long-term forecasting contexts due to their mechanisms decoupling latent dynamics and input assimilation. However, they assume simple latent dynamics (linear) and don't include any physical prior.



(a) PhyDNet disentangling recurrent bloc **(b) Global seq2seq architecture**

Figure 2. **Proposed PhyDNet deep model for video forecasting.** a) The core of PhyDNet is a recurrent block projecting input images $\mathbf{u_t}$ into a latent space $\mathcal{H}$, where two recurrent neural networks disentangle physical dynamics (PhyCell, section 3.2) from residual information (ConvLSTM). Learned physical $\mathbf{h}^{\mathbf{p}}_{t+1}$ and residual $\mathbf{h}^{\mathbf{r}}_{t+1}$ representations are summed before decoding to predict the future image $\hat{\mathbf{u}}_{t+1}$. b) Unfolded in time, PhyDNet forms a sequence to sequence (seq2seq) architecture suited for multi-step video prediction. Dotted arrows mean that predictions are reinjected as next input only for the ConvLSTM branch, and not for PhyCell, as explained in section 3.3.

3. PhyDNet model for video forecasting

We introduce PhyDNet, a model dedicated to video prediction, which leverages physical knowledge on dynamics, and disentangles it from other unknown factors of variations necessary for accurate forecasting. To achieve this goal, we introduce a disentangling architecture (section 3.1), and a new physically-constrained recurrent cell (section 3.2).

Problem statement As discussed in introduction, physical laws do not apply at the pixel level for general video prediction tasks. However, we assume that there exists a conceptual latent space $\mathcal{H}$ in which physical dynamics and residual factors are linearly disentangled. Formally, let us denote as $\mathbf{u} = \mathbf{u}(t, \mathbf{x})$ the frame of a video sequence at time t, for spatial coordinates $\mathbf{x} = (x, y)$. $\mathbf{h}(t, \mathbf{x}) \in \mathcal{H}$ is the latent representation of the video up to time t, which decomposes as $\mathbf{h} = \mathbf{h^p} + \mathbf{h^r}$, where $\mathbf{h^p}$ (resp. $\mathbf{h^r}$) represents the physical (resp. residual) component of the disentanglement. The video evolution in the latent space $\mathcal{H}$ is thus governed by the following partial differential equation (PDE):

$$\frac{\partial \mathbf{h}(t,\mathbf{x})}{\partial t} = \frac{\partial \mathbf{h^p}}{\partial t} + \frac{\partial \mathbf{h^r}}{\partial t} := \mathcal{M}_p(\mathbf{h^p}, \mathbf{u}) + \mathcal{M}_r(\mathbf{h^r}, \mathbf{u}) \quad (1)$$

$\mathcal{M}_p(\mathbf{h^p}, \mathbf{u})$ and $\mathcal{M}_r(\mathbf{h^r}, \mathbf{u})$ represent physical and residual dynamics in the latent space $\mathcal{H}$.

3.1. PhyDNet disentangling architecture

The main goal of PhyDNet is to learn the mapping from input sequences to a latent space which approximates the disentangling properties formalized in Eq (1).

To reach this objective, we introduce a recurrent bloc which is shown in Figure 2(a). A video frame $\mathbf{u}_t$ at time t is mapped by a deep convolutional encoder $\mathbf{E}$ into a latent space representing the targeted space $\mathcal{H}$. $\mathbf{E}(\mathbf{u}_t)$ is then used as input for two parallel recurrent neural networks, incorporating this spatial representation into a dynamical model.

The left branch in Figure 2(a) models the latent representation $\mathbf{h^p}$ fulfilling the physical part of the PDE in Eq (1), *i.e.* $\frac{\partial \mathbf{h^p}(t,\mathbf{x})}{\partial t} = \mathcal{M}_p(\mathbf{h^p}, \mathbf{u})$. This PDE is modeled by our recurrent physical cell described in section 3.2, PhyCell, which leads to the computation of $\mathbf{h}^{\mathbf{p}}_{t+1}$ from $\mathbf{E}(\mathbf{u}_t)$ and $\mathbf{h}^{\mathbf{p}}_t$. From the machine learning perspective, PhyCell leverages physical constraints to limit the number of model parameters, regularizes training and improves generalization.

The right branch in Figure 2(a) models the latent representation $\mathbf{h^r}$ fulfilling the residual part of the PDE in Eq (1), *i.e.* $\frac{\partial \mathbf{h^r}(t,\mathbf{x})}{\partial t} = \mathcal{M}_r(\mathbf{h^r}, \mathbf{u})$. Inspired by wavelet decomposition [39] and recent semi-supervised works [51], this part of the PDE corresponds to unknown phenomena, which do not correspond to any prior model, and is therefore entirely learned from data. We use a generic recurrent neural network for this task, *e.g.* ConvLSTM [73] for videos, which computes $\mathbf{h}^{\mathbf{r}}_{t+1}$ from $\mathbf{E}(\mathbf{u}_t)$ and $\mathbf{h}^{\mathbf{r}}_t$.

$\mathbf{h}_{t+1} = \mathbf{h}^{\mathbf{p}}_{t+1} + \mathbf{h}^{\mathbf{r}}_{t+1}$ is the combined representation processed by a deep decoder $\mathbf{D}$ to forecast the image $\hat{\mathbf{u}}_{t+1}$.

Figure 2(b) shows the "unfolded" PhyDNet. An input video $\mathbf{u}_{1:T} = (\mathbf{u}_1, ..., \mathbf{u}_T) \in \mathbb{R}^{T \times n \times m \times c}$ with spatial size

$n \times m$ and c channels is projected into $\mathcal{H}$ by the encoder $\mathbf{E}$ and processed by the recurrent block unfolded in time. This forms a Sequence To Sequence architecture [57] suited for multi-step prediction, outputting Δ future frame predictions $\hat{\mathbf{u}}_{T+1:T+\Delta}$. Encoder, decoder and recurrent block parameters are all trained end-to-end, meaning that PhyDNet learns itself without supervision the latent space $\mathcal{H}$ in which physics and residual factors are disentangled.

3.2. PhyCell: a deep recurrent physical model

PhyCell is a new physical cell, whose dynamics is governed by the PDE response function $\mathcal{M}_p(\mathbf{h}^{\mathbf{p}}, \mathbf{u})$[1]:

$$\mathcal{M}_p(\mathbf{h}, \mathbf{u}) := \Phi(\mathbf{h}) + \mathcal{C}(\mathbf{h}, \mathbf{u}) \tag{2}$$

where $\Phi(\mathbf{h})$ is a physical predictor modeling only the latent dynamics and $C(\mathbf{h}, \mathbf{u})$ is a correction term modeling the interactions between latent state and input data.

Physical predictor: $\Phi(\mathbf{h})$ in Eq (2) is modeled as follows:

$$\Phi(\mathbf{h}(t, \mathbf{x})) = \sum_{i,j:i+j\leq q} c_{i,j} \frac{\partial^{i+j}\mathbf{h}}{\partial x^i \partial y^j}(t, \mathbf{x}) \tag{3}$$

$\Phi(\mathbf{h}(t, \mathbf{x}))$ in Eq (3) combines the spatial derivatives with coefficients $c_{i,j}$ up to a certain differential order q. This generic class of linear PDEs subsumes a wide range of classical physical models, *e.g.* the heat equation, the wave equations, the advection-diffusion equations.

Correction: $\mathcal{C}(\mathbf{h}, \mathbf{u})$ in Eq (2) takes the following form:

$$\mathcal{C}(\mathbf{h}, \mathbf{u}) := \mathbf{K}(t, \mathbf{x}) \odot [\mathbf{E}(\mathbf{u}(t, \mathbf{x})) - (\mathbf{h}(t, \mathbf{x}) + \Phi(\mathbf{h}(t, \mathbf{x}))] \tag{4}$$

Eq (4) computes is the difference between the latent state after physical motion $\mathbf{h}(t, \mathbf{x}) + \Phi(\mathbf{h}(t, \mathbf{x}))$ and the embedded new observed input $\mathbf{E}(\mathbf{u}(t, \mathbf{x}))$. $\mathbf{K}(t, \mathbf{x})$ is a gating factor, where $\odot$ is the Hadamard product.

3.2.1 Discrete PhyCell

We discretize the continuous time PDE in Eq (2) with the standard forward Euler numerical scheme [37], leading to the discrete time PhyCell (derivation in supplementary 1.1):

$$\mathbf{h}_{t+1} = (1 - \mathbf{K}_t) \odot (\mathbf{h}_t + \Phi(\mathbf{h}_t)) + \mathbf{K}_t \odot \mathbf{E}(\mathbf{u}_t) \tag{5}$$

Depicted in Figure 3, PhyCell is an atomic recurrent cell for building physically-constrained RNNs. In our experiments, we use one layer of PhyCell but one can also easily stack several PhyCell layers to build more complex models, as done for stacked RNNs [66, 64, 67]. To gain insight into PhyCell in Eq (5), we write the equivalent two-steps form:

$$\begin{cases} \tilde{\mathbf{h}}_{t+1} = \mathbf{h}_t + \Phi(\mathbf{h}_t) & \textbf{Prediction} \quad (6) \\ \mathbf{h}_{t+1} = \tilde{\mathbf{h}}_{t+1} + \mathbf{K}_t \odot \left(\mathbf{E}(\mathbf{u}_t) - \tilde{\mathbf{h}}_{t+1}\right) & \textbf{Correction} \quad (7) \end{cases}$$

[1] In the sequel, we drop the index $\mathbf{p}$ in $\mathbf{h}^{\mathbf{p}}$ for the sake of simplicity

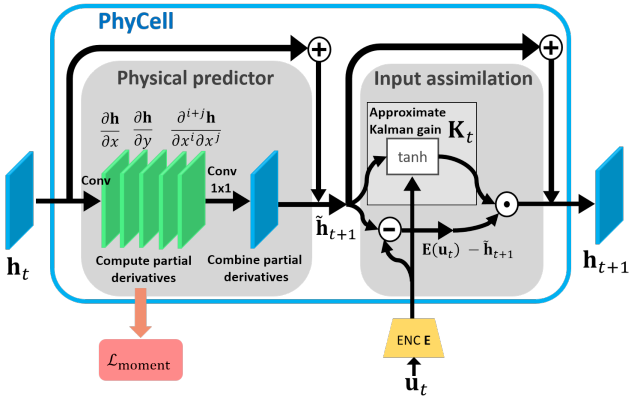


Figure 3. PhyCell recurrent cell implements a two-steps scheme: physical prediction with convolutions for approximating and combining spatial derivatives (Eq (6) and Eq (3)), and input assimilation as a correction of latent physical dynamics driven by observed data (Eq (7)). During training, the filter moment loss in red (Eq (10)) enforces the convolutional filters to approximate the desired differential operators.

The prediction step in Eq (6) is a physically-constrained motion in the latent space, computing the intermediate representation $\tilde{\mathbf{h}}_{t+1}$. Eq (7) is a correction step incorporating input data. This prediction-correction formulation is reminiscent of the way to combine numerical models with observed data in the data assimilation community [1, 4], *e.g.* with the Kalman filter [25]. We show in section 3.3 that this decoupling between prediction and correction can be leveraged to robustly train our model in long-term forecasting and missing data contexts. $\mathbf{K}_t$ can be interpreted as the Kalman gain controlling the trade-off between both steps.

3.2.2 PhyCell implementation

We now specify how the physical predictor Φ in Eq (6) and the correction Kalman gain $\mathbf{K}_t$ in Eq (7) are implemented.

Physical predictor: we implement Φ using a convolutional neural network (left gray box in Figure 3), based on the connection between convolutions and differentiations [13, 35]. This offers the possibility to learn a class of filters approximating each partial derivative in Eq (3), which are constrained by a kernel moment loss, as detailed in section 3.3. As noted by [35], the flexibility added by this constrained learning strategy gives better results for solving PDEs than handcrafted derivative filters. Finally, we use 1×1 convolutions to linearly combine these derivatives with $c_{i,j}$ coefficients in Eq (3).

Kalman gain: We approximate $\mathbf{K}_t$ in Eq (7) by a gate with learned convolution kernels $\mathbf{W}_h$, $\mathbf{W}_u$ and bias $\mathbf{b}_k$:

$$\mathbf{K}_t = \tanh\left(\mathbf{W}_h * \tilde{\mathbf{h}}_{t+1} + \mathbf{W}_u * \mathbf{E}(\mathbf{u}_t) + \mathbf{b}_k\right) \tag{8}$$

Note that if $\mathbf{K}_t = \mathbf{0}$, the input is not accounted for and the dynamics follows the physical predictor ; if $\mathbf{K}_t = 1$, the latent dynamics is resetted and only driven by the input. This is similar to gating mechanisms in LSTMs or GRUs.

Discussion: With specific Φ predictor, $\mathbf{K}_t$ gain and encoder $\mathbf{E}$, PhyCell recovers recent models from the literature:

model	Φ	$\mathbf{K}_t$	$\mathbf{E}$
PDE-Net [34]	Eq (6)	**0**	**Id**
Advection-diffusion flow [11]	advection-diffusion predictor	**0**	**Id**
RKF [3]	locally linear, no phys. constraint	approx. Kalman gain	deep encoder
PhyDNet (ours)	Eq (6)	Eq (8)	deep encoder

PDE-Net [35] directly works on raw pixel data (identity encoder $\mathbf{E}$) and assumes Markovian dynamics (no correction, $\mathbf{K}_t{=}\mathbf{0}$): the model solves the autonomous PDE $\frac{\partial \mathbf{u}}{\partial t} = \Phi(\mathbf{u})$ given in Eq (6) but in pixel space. This prevents from modeling time-varying PDEs such as those tackled in this work, *e.g.* varying advection terms. The flow model in [11] uses the closed-form solution of the advection-diffusion equation as predictor ; it is however limited only to this PDE, whereas PhyDNet models a much broader class of PDEs. The Recurent Kalman Filter (RKF) [3] also proposes a prediction-correction scheme in a deep latent space, but their approach does not include any prior physical information, and the prediction step is locally linear, whereas we use deep models. An approximated form of the covariance matrix is used for estimating $\mathbf{K}_t$ in [3], which we find experimentally inferior to our gating mechanism in Eq (8).

3.3. Training

Given a training set of N videos $\boldsymbol{\mathcal{D}} = \left\{\mathbf{u}^{(i)}\right\}_{i=\{1:N\}}$ and PhyDNet parameters $\mathbf{w} = (\mathbf{w_p}, \mathbf{w_r}, \mathbf{w_s})$, where $\mathbf{w_p}$ (resp. $\mathbf{w_r}$) are parameters of the PhyCell (resp. residual) branch, and $\mathbf{w_s}$ are encoder and decoder shared parameters, we minimize the following objective function:

$$\mathcal{L}(\boldsymbol{\mathcal{D}}, \mathbf{w}) = \mathcal{L}_{\text{image}}(\boldsymbol{\mathcal{D}}, \mathbf{w}) + \lambda \mathcal{L}_{\text{moment}}(\mathbf{w_p}) \qquad (9)$$

We use the L^2 loss for the image reconstruction loss $\mathcal{L}_{\text{image}}$, as commonly done in the literature [66, 64, 44, 65, 67].

$\mathcal{L}_{\text{moment}}(\mathbf{w_p})$ imposes physical constraints on the k^2 learned filters $\left\{\mathbf{w}^k_{p,i,j}\right\}_{i,j\leq k}$, such that each $\mathbf{w}^k_{p,i,j}$ of size $k \times k$ approximates $\frac{\partial^{i+j}}{\partial x^i y^j}$. This is achieved by using a loss based on the moment matrix $\mathbf{M}(\mathbf{w}^k_{p,i,j})$ [34], representing the order of the filter differentiation [13]. $\mathbf{M}(\mathbf{w}^k_{p,i,j})$ is compared to a target moment matrix $\mathbf{\Delta}^k_{i,j}$ (see $\mathbf{M}$ and $\mathbf{\Delta}$ computations in supplementary 1.2), leading to:

$$\mathcal{L}_{\text{moment}} = \sum_{i\leq k} \sum_{j\leq k} ||\mathbf{M}(\mathbf{w}^k_{p,i,j}) - \mathbf{\Delta}^k_{i,j}||_F \qquad (10)$$

Prediction mode An appealing feature of PhyCell is that we can use and train the model in a "prediction-only" mode by setting $\mathbf{K}_t = \mathbf{0}$ in Eq (7), *i.e.* by only relying on the physical predictor Φ in Eq (6). It is worth mentioning that the "prediction-only" mode is not applicable to standard Seq2Seq RNNs: although the decomposition in Eq (2) still holds, *i.e.* $\boldsymbol{\mathcal{M}}_r(\mathbf{h}, \mathbf{u}) = \Phi(\mathbf{h}) + \mathcal{C}(\mathbf{h}, \mathbf{u})$, the resulting predictor is naive and useless for multi-step prediction $\tilde{\mathbf{h}}_{t+1} = 0$, see supplementary 1.3).

Therefore, standard RNNs are not equipped to deal with unreliable input data $\mathbf{u}_t$. We show in section 4.4 that the gain of PhyDNet over those models increases in two important contexts with unreliable inputs: multi-step prediction and dealing with missing data.

4. Experiments

4.1. Experimental setup

Datasets We evaluate PhyDNet on four datasets from various origins. **Moving MNIST** [56] is a standard synthetic benchmark in video prediction with two random digits bouncing on the walls. **Traffic BJ** [77] represents complex real-world traffic flows, which requires modeling transport phenomena and traffic diffusion for prediction. **SST** (Sea Surface Temperature) [11] consists in meteorological data, whose evolution is governed by the physical laws of fluid dynamics. Finally, **Human 3.6** [22] represents general human actions with complex 3D articulated motions. We give details about all datasets in supplementary 2.1.

	Moving MNIST			**Traffic BJ**			**Sea Surface Temperature**			**Human 3.6**		
Method	MSE	MAE	SSIM	MSE $\times 100$	MAE	SSIM	MSE $\times 10$	MAE	SSIM	MSE / 10	MAE /100	SSIM
ConvLSTM [73]	103.3	182.9	0.707	48.5*	17.7*	0.978*	45.6*	63.1*	0.949*	50.4*	18.9*	0.776*
PredRNN [66]	56.8	126.1	0.867	46.4	17.1*	0.971*	41.9	62.1	0.955	48.4	18.9	0.781
Causal LSTM [64]	46.5	106.8	0.898	44.8	16.9*	0.977*	39.1*	62.3*	0.929*	45.8	17.2	0.851
MIM [67]	44.2	101.1	0.910	42.9	16.6*	0.971*	42.1*	60.8*	0.955*	42.9	17.8	0.790
E3D-LSTM [65]	41.3	86.4	0.920	43.2*	16.9*	0.979*	34.7*	59.1*	0.969*	46.4	16.6	0.869
Advection-diffusion [11]	-	-	-	-	-	-	34.1*	54.1*	0.966*	-	-	-
DDPAE [21]	38.9	90.7*	0.922*	-	-	-	-	-	-	-	-	-
PhyDNet	**24.4**	**70.3**	**0.947**	**41.9**	**16.2**	**0.982**	**31.9**	**53.3**	**0.972**	**36.9**	**16.2**	**0.901**

Table 1. Quantitative forecasting results of PhyDNet compared to baselines using various datasets. Numbers are copied from original or citing papers. * corresponds to results obtained by running online code from the authors. The first five baseline are general deep models applicable to all datasets, whereas DDPAE [21] (resp. advection-diffusion flow [11]) are specific state-of-the-art models for Moving MNIST (resp. SST). Metrics are scaled to be in a similar range across datasets to ease comparison.

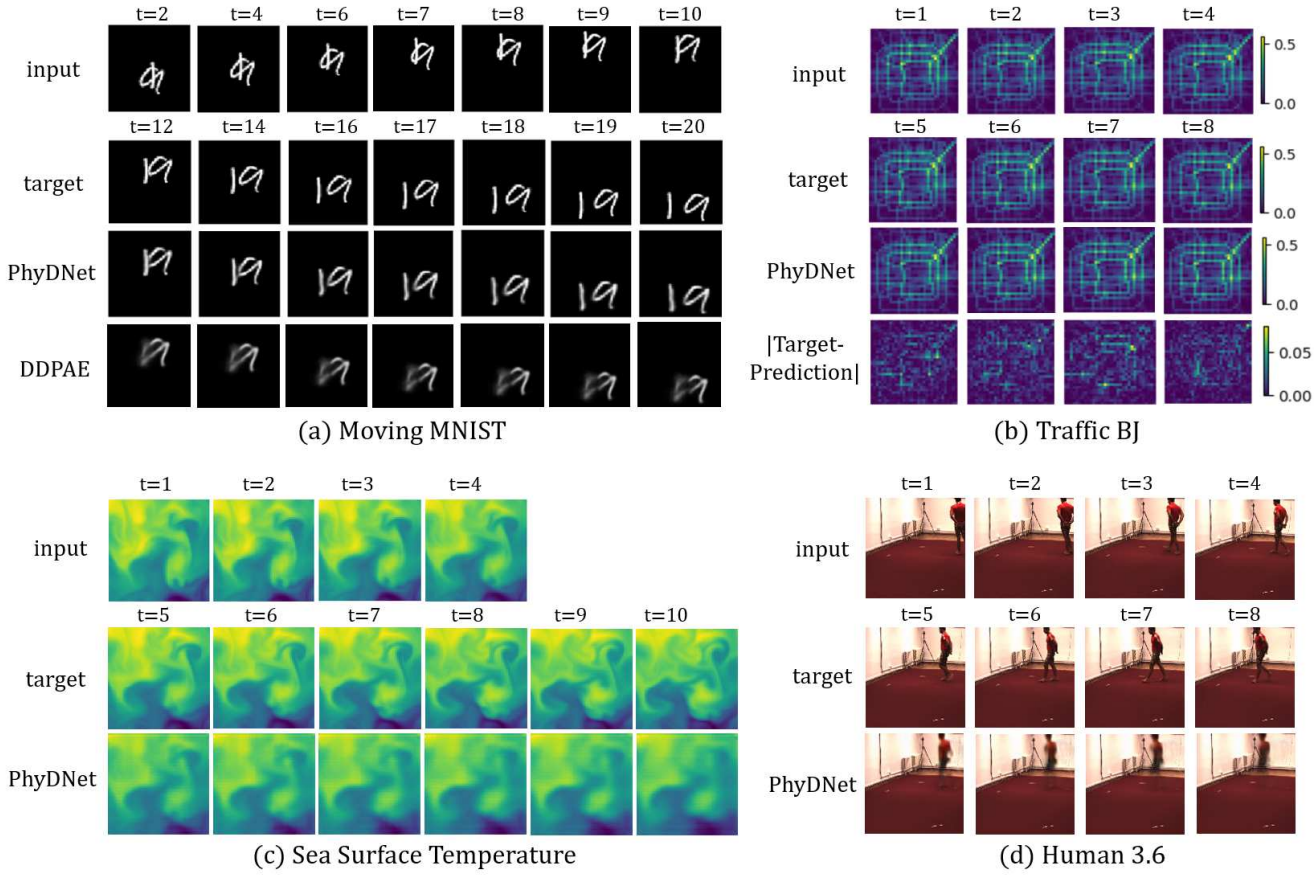


Figure 4. Qualitative results of the predicted frames by PhyDNet for all datasets. First line is the input sequence, second line the target and third line PhyDNet prediction. For Moving MNIST, we add a fourth line with the comparison to DDPAE [21] and for Traffic BJ the difference |Prediction-Target| for better visualization.

Network architectures and training PhyDNet shares a common backbone architecture for all datasets where the physical branch contains 49 PhyCells (7×7 kernel filters) and the residual branch is composed of a 3-layers ConvLSTM with 128 filters in each layer. We set up the trade-off parameter between $\mathcal{L}_{\text{image}}$ and $\mathcal{L}_{\text{moment}}$ to $\lambda = 1$. Detailed architectures and λ impact are given in supplementary 2.2. Our code is available at https://github.com/vincent-leguen/PhyDNet.

Evaluation metrics We follow evaluation metrics commonly used in state-of-the-art video prediction methods: the Mean Squared Error (MSE), Mean Absolute Error (MAE) and the Structural Similarity (SSIM) [68] that computes the perceived image quality with respect to a reference. Metrics are averaged for each frame of the output sequence. Lower MSE, MAE and higher SSIM indicate better performances.

4.2. State of the art comparison

We evaluate PhyDNet against strong recent baselines, including very competitive data-driven RNN architectures: ConvLSTM [73], PredRNN [66], Causal LSTM [64], Memory in Memory (MIM) [67]. We also compare to methods dedicated to specific datasets: DDPAE [21], a disentangling method specialized and state-of-the-art on Moving MNIST ; and the physically-constrained advection-diffusion flow model [11] that is state-of-the-art for the SST dataset.

Overall results presented in Table 1 reveal that PhyDNet outperforms significantly all baselines on all four datasets. The performance gain is large with respect to state-of-the-art general RNN models, with a gain of 17 MSE points for Moving MNIST, 6 MSE points for Human 3.6, 3 MSE points for SST and 1 MSE point for Traffic BJ. In addition, PhyDNet also outperforms specialized models: it gains 14 MSE points compared to the disentangling DDPAE model [21] specialized for Moving MNIST, and 2 MSE points compared to the advection-diffusion model [11] dedicated to SST data. PhyDNet also presents large and consistent gains in SSIM, indicating that image quality is greatly improved by the physical regularization. Note that for Human 3.6, a few approaches use specific strategies dedicated to human motion with additional supervision, *e.g.* hu-

	Moving MNIST			Traffic BJ			Sea Surface Temperature			Human 3.6		
Method	MSE	MAE	SSIM	MSE × 100	MAE	SSIM	MSE × 10	MAE	SSIM	MSE / 10	MAE / 100	SSIM
ConvLSTM	103.3	182.9	0.707	48.5*	17.7*	0.978*	45.6*	63.1*	0.949*	50.4*	18.9*	0.776*
PhyCell	50.8	129.3	0.870	48.9	17.9	0.978	38.2	60.2	0.969	42.5	18.3	0.891
PhyDNet	**24.4**	**70.3**	**0.947**	**41.9**	**16.2**	**0.982**	**31.9**	**53.3**	**0.972**	**36.9**	**16.2**	**0.901**

Table 2. An ablation study shows the consistent performance gain on all datasets of our physically-constrained PhyCell vs the general purpose ConvLSTM, and the additional gain brought up by the disentangling architecture PhyDNet. * corresponds to results obtained by running online code from the authors.

man pose in [61]. We perform similarly to [61] using only unsupervised training, as shown in supplementary 2.3. This is, to the best of our knowledge, the first time that physically-constrained deep models reach state-of-the-art performances on generalist video prediction datasets.

In Figure 4, we provide qualitative prediction results for all datasets, showing that PhyDNet properly forecasts future images for the considered horizons: digits are sharply and accurately predicted for Moving MNIST in (a), the absolute traffic flow error is low and approximately spatially independent in (b), the evolving physical SST phenomena are well anticipated in (c) and the future positions of the person is accurately predicted in (d). We add in Figure 4(a) a qualitative comparison to DDPAE [21], which fails to predict the future frames properly. Since the two digits overlap in the input sequence, DPPAE is unable to disentangle them. In contrast, PhyDNet successfully learns the physical dynamics of the two digits in a disentangled latent space, leading a correct prediction. In supplementary 2.4, we detail this comparison to DPPAE, and provide additional visualizations for all datasets.

4.3. Ablation Study

We perform here an ablation study to analyse the respective contributions of physical modeling and disentanglement. Results are presented in Table 2 for all datasets. We see that a 1-layer PhyCell model (only the left branch of PhyDNet in Figure 2(b)) outperforms a 3-layers ConvLSTM (50 MSE points gained for Moving MNIST, 8 MSE points for Human 3.6, 7 MSE points for SST and equivalent results for Traffic BJ), while PhyCell has much fewer parameters (270,000 *vs.* 3 million parameters). This confirms that PhyCell is a very effective recurrent cell that successfully incorporates physical prior in deep models. When we further add our disentangling strategy with the two-branch architecture (PhyDNet), we have another performance gap on all datasets (25 MSE points for Moving MNIST, 7 points for Traffic and SST, and 5 points for Human 3.6), which proves that physical modeling is not sufficient by itself to perform general video prediction and that learning unknown factors is necessary.

We qualitatively analyze in Figure 5 partial predictions of PhyDNet for the physical branch $\hat{\mathbf{u}}^{\mathbf{p}}_{t+1} = \mathbf{D}(\mathbf{h}^{\mathbf{p}}_{t+1})$ and residual branch $\hat{\mathbf{u}}^{\mathbf{r}}_{t+1} = \mathbf{D}(\mathbf{h}^{\mathbf{r}}_{t+1})$. As noted in Figure 1

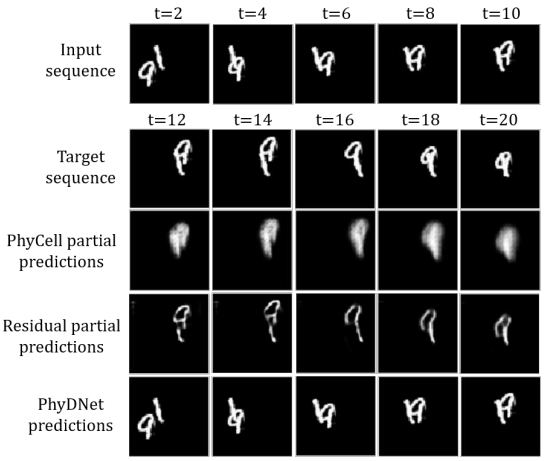


Figure 5. Qualitative ablation results on Moving MNIST: partial predictions show that PhyCell captures coarse localisation of digits, whereas the ConvLSTM branch models the fine shape details of digits. Every two frames are displayed.

for Moving MNIST, $\mathbf{h}^{\mathbf{p}}$ captures coarse localisations of objects, while $\mathbf{h}^{\mathbf{r}}$ captures fine-grained details that are not useful for the physical model. Additional visualizations for the other datasets and a discussion on the number of parameters are provided in supplementary 2.5.

Influence of physical regularization We conduct in Table 3 a finer ablation on Moving MNIST to study the impact of the physical regularization $\mathcal{L}_{\text{moment}}$ on the performance of PhyCell and PhyDNet. When we disable $\mathcal{L}_{\text{moment}}$ for training PhyCell, performances improve by 7 points in MSE. This underlines that physical laws alone are too restrictive for learning dynamics in a general context, and that complementary factors should be accounted for. On the other side, when we disable $\mathcal{L}_{\text{moment}}$ for training our disentangled architecture PhyDNet, performances decrease by 5 MSE points (29 *vs* 24.4) compared to the physically-constrained version. This proves that physical constraints are relevant, but should be incorporated carefully in order to make both branches cooperate. This enables to leverage physical prior, while keeping remaining information necessary for pixel-level prediction. Same conclusions can be drawn for the other datasets, see supplementary 2.6.

Method	MSE	MAE	SSIM
PhyCell	50.8	129.3	0.870
PhyCell without $\mathcal{L}_{\text{moment}}$	43.4	112.8	0.895
PhyDNet	**24.4**	**70.3**	**0.947**
PhyDNet without $\mathcal{L}_{\text{moment}}$	29.0	81.2	0.934

Table 3. Influence of physical regularization for Moving MNIST.

4.4. PhyCell analysis

4.4.1 Physical filter analysis

With the same general backbone architecture, PhyDNet can express different PDE dynamics associated to the underlying phenomena by learning specific $c_{i,j}$ coefficients combining the partial derivatives in Eq (3). In Figure 6, we display the mean amplitude of the learned coefficients $c_{i,j}$ with respect to the order of differentiation. For Moving MNIST, the 0^{th} and 1^{st} orders are largely dominant, meaning a purely advective behaviour coherent with the piecewise-constant translation dynamics of the dataset. For Traffic BJ and SST, there is also a global decrease in amplitude with respect to order, we nonetheless notice a few higher order terms appearing to be useful for prediction. For Human 3.6, where the nature of the prior motion is less obvious, these coefficients are more spread across order derivatives.

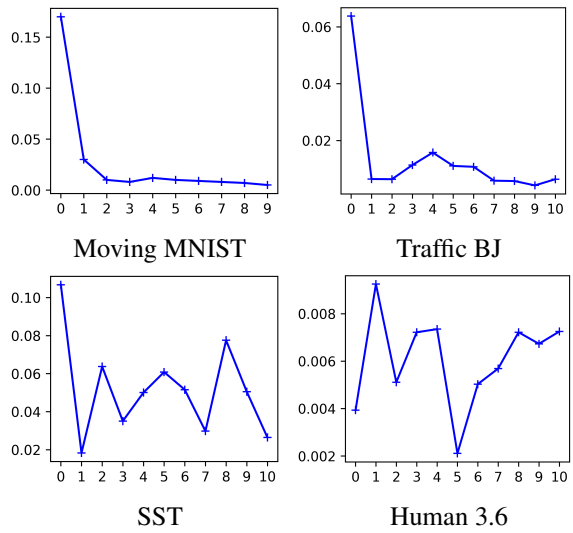


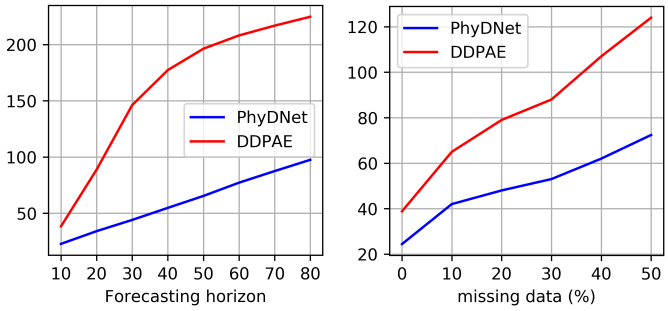
Figure 6. Mean amplitude of the combining coefficients $c_{i,j}$ with respect to the order of the differential operators approximated.

4.4.2 Dealing with unreliable inputs

We explore here the robustness of PhyDNet when dealing with unreliable inputs, that can arise in two contexts: long-term forecasting and missing data. As explained in section 3.3, PhyDNet can be used in a prediction mode in this context, limiting the use of unreliable inputs, whereas general RNNs cannot. To validate the relevance of the prediction mode, we compare PhyDNet to DDPAE [21], based on a standard RNN (LSTM) as predictor module. Figure 7 presents the results in MSE obtained by PhyDNet and DDPAE on Moving MNIST (see supplementary 2.7 for similar results in SSIM).

For long-term forecasting, we evaluate the performances of both methods far beyond the prediction range seen during training (up to 80 frames), as shown in Figure 7(a). We can see that the performance drop (MSE increase rate) is approximately linear for PhyNet, whereas it is much more pronounced for DDPAE. For example, PhyDNet for 80-steps prediction reaches similar performances in MSE than DDPAE for 20-steps prediction. This confirms that PhyDNet can limit error accumulation during forecasting by using a powerful dynamical model.

Finally, we evaluate the robustness of PhyDNet on DDPAE on missing data, by varying the ratio of missing data (from 10 to 50%) in input sequences during training and testing. A missing input image is replaced with a default value (0) image. In this case, PhyCell relies only on its latent dynamics by setting $\mathbf{K}_t = 0$, whereas DDPAE takes the null image as input. Figure 7(b) shows that the performance gap between PhyDNet and DDPAE increases with the percentage of missing data.



(a) Long-term forecasting (b) Missing data

Figure 7. MSE comparison between PhyDNet and DDPAE [21] when dealing with unreliable inputs.

5. Conclusion

We propose PhyDNet, a new model for disentangling prior dynamical knowledge from other factors of variation required for video prediction. PhyDNet enables to apply PDE-constrained prediction beyond fully observed physical phenomena in pixel space, and to outperform state-of-the-art performances on four generalist datasets. Our introduced recurrent physical cell for modeling PDE dynamics generalizes recent models and offers the appealing property to decouple prediction from correction. Future work include using more complex numerical schemes, *e.g.* Runge-Kutta [15], and extension to probabilistic forecasts with uncertainty estimation [18, 9], *e.g.* with stochastic differential equations [23].

References

[1] M. Asch, M. Bocquet, and M. Nodet. *Data assimilation: methods, algorithms, and applications*, volume 11. SIAM, 2016. 2, 4

[2] P. Battaglia, R. Pascanu, M. Lai, D. J. Rezende, et al. Interaction networks for learning about objects, relations and physics. In *Advances in neural information processing systems (NeurIPS)*, pages 4502–4510, 2016. 2

[3] P. Becker, H. Pandya, G. Gebhardt, C. Zhao, C. J. Taylor, and G. Neumann. Recurrent Kalman networks: Factorized inference in high-dimensional deep feature spaces. In *International Conference on Machine Learning (ICML)*, pages 544–552, 2019. 2, 5

[4] M. Bocquet, J. Brajard, A. Carrassi, and L. Bertino. Data assimilation as a learning tool to infer ordinary differential equation representations of dynamical models. *Nonlinear Processes in Geophysics*, 26(3):143–162, 2019. 2, 4

[5] S. L. Brunton, J. L. Proctor, and J. N. Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016. 2

[6] W. Byeon, Q. Wang, R. Kumar Srivastava, and P. Koumoutsakos. ContextVP: Fully context-aware video prediction. In *European Conference on Computer Vision (ECCV)*, pages 753–769, 2018. 2

[7] L. Castrejon, N. Ballas, and A. Courville. Improved conditional VRNNs for video prediction. In *International Conference on Computer Vision (ICCV)*, 2019. 1

[8] T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud. Neural ordinary differential equations. In *Advances in neural information processing systems (NeurIPS)*, 2018. 1, 2

[9] C. Corbière, N. Thome, A. Bar-Hen, M. Cord, and P. Pérez. Addressing failure prediction by learning model confidence. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2902–2913, 2019. 8

[10] M. Cuturi and M. Blondel. Soft-dtw: a differentiable loss function for time-series. In *International Conference on Machine Learning (ICML)*, pages 894–903, 2017. 2

[11] E. de Bezenac, A. Pajot, and P. Gallinari. Deep learning for physical processes: Incorporating prior scientific knowledge. *International Conference on Learning Representations (ICLR)*, 2018. 1, 2, 5, 6

[12] E. L. Denton et al. Unsupervised learning of disentangled representations from video. In *Advances in neural information processing systems (NeurIPS)*, pages 4414–4423, 2017. 1, 2

[13] B. Dong, Q. Jiang, and Z. Shen. Image restoration: Wavelet frame shrinkage, nonlinear evolution PDEs, and beyond. *Multiscale Modeling & Simulation*, 15(1):606–660, 2017. 4, 5

[14] S. A. Eslami, N. Heess, T. Weber, Y. Tassa, D. Szepesvari, G. E. Hinton, et al. Attend, infer, repeat: Fast scene understanding with generative models. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3225–3233, 2016. 2

[15] R. Fablet, S. Ouala, and C. Herzet. Bilinear residual neural network for the identification and forecasting of geophysical dynamics. In *2018 26th European Signal Processing Conference (EUSIPCO)*, pages 1477–1481. IEEE, 2018. 2, 8

[16] C. Finn, I. Goodfellow, and S. Levine. Unsupervised learning for physical interaction through video prediction. In *Advances in neural information processing systems (NeurIPS)*, pages 64–72, 2016. 1, 2

[17] M. Fraccaro, S. Kamronn, U. Paquet, and O. Winther. A disentangled recognition and nonlinear dynamics model for unsupervised learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3601–3610, 2017. 2

[18] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning (ICML)*, pages 1050–1059, 2016. 8

[19] H. Gao, H. Xu, Q.-Z. Cai, R. Wang, F. Yu, and T. Darrell. Disentangling propagation and generation for video prediction. In *International Conference on Computer Vision (ICCV)*, pages 9006–9015, 2019. 2

[20] T. Haarnoja, A. Ajay, S. Levine, and P. Abbeel. Backprop KF: Learning discriminative deterministic state estimators. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4376–4384, 2016. 2

[21] J.-T. Hsieh, B. Liu, D.-A. Huang, L. F. Fei-Fei, and J. C. Niebles. Learning to decompose and disentangle representations for video prediction. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 517–526, 2018. 1, 2, 5, 6, 7, 8

[22] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, 2013. 5

[23] J. Jia and A. R. Benson. Neural jump stochastic differential equations. In *Advances in Neural Information Processing Systems*, pages 9843–9854, 2019. 8

[24] X. Jia, B. De Brabandere, T. Tuytelaars, and L. V. Gool. Dynamic filter networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 667–675, 2016. 1, 2

[25] R. Kalman. A new approach to linear filtering and prediction problems. *Trans. ASME, D*, 82:35–44, 1960. 2, 4

[26] T. Kipf, E. Fetaya, K.-C. Wang, M. Welling, and R. Zemel. Neural relational inference for interacting systems. In *International Conference on Machine Learning (ICML)*, pages 2693–2702, 2018. 2

[27] A. Kosiorek, H. Kim, Y. W. Teh, and I. Posner. Sequential attend, infer, repeat: Generative modelling of moving objects. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 8606–8616, 2018. 2

[28] R. G. Krishnan, U. Shalit, and D. Sontag. Deep Kalman filters. *ArXiv*, abs/1511.05121, 2015. 2

[29] Y.-H. Kwon and M.-G. Park. Predicting future frames using retrospective cycle GAN. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1811–1820, 2019. 1, 2

[30] V. Le Guen and N. Thome. Shape and time distortion loss for training deep time series forecasting models. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4191–4203, 2019. 2

[31] Y. Li, C. Fang, J. Yang, Z. Wang, X. Lu, and M.-H. Yang. Flow-grounded spatial-temporal video prediction from still images. In *European Conference on Computer Vision (ECCV)*, pages 600–615, 2018. 2

[32] X. Liang, L. Lee, W. Dai, and E. P. Xing. Dual motion GAN for future-flow embedded video prediction. In *International Conference on Computer Vision (ICCV)*, pages 1744–1752, 2017. 2

[33] Z. Liu, R. A. Yeh, X. Tang, Y. Liu, and A. Agarwala. Video frame synthesis using deep voxel flow. In *International Conference on Computer Vision (ICCV)*, pages 4463–4471, 2017. 1, 2

[34] Z. Long, Y. Lu, and B. Dong. PDE-Net 2.0: Learning PDEs from data with a numeric-symbolic hybrid deep network. *Journal of Computational Physics*, page 108925, 2019. 2, 5

[35] Z. Long, Y. Lu, X. Ma, and B. Dong. PDE-Net: Learning PDEs from data. In *International Conference on Machine Learning*, pages 3214–3222, 2018. 1, 2, 4, 5

[36] C. Lu, M. Hirsch, and B. Scholkopf. Flexible spatio-temporal networks for video prediction. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6523–6531, 2017. 2

[37] Y. Lu, A. Zhong, Q. Li, and B. Dong. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. In *International Conference on Machine Learning (ICML)*, pages 3282–3291, 2018. 1, 2, 4

[38] Z. Luo, B. Peng, D.-A. Huang, A. Alahi, and L. Fei-Fei. Unsupervised learning of long-term motion dynamics for videos. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2203–2212, 2017. 2

[39] S. Mallat. *A wavelet tour of signal processing*. Elsevier, 1999. 3

[40] M. Mathieu, C. Couprie, and Y. LeCun. Deep multi-scale video prediction beyond mean square error. In *International Conference on Learning Representations (ICLR)*, 2015. 1, 2

[41] M. Minderer, C. Sun, R. Villegas, F. Cole, K. Murphy, and H. Lee. Unsupervised learning of object structure and dynamics from videos. In *Advances in neural information processing systems (NeurIPS)*, 2019. 1, 2

[42] D. Mrowca, C. Zhuang, E. Wang, N. Haber, L. F. Fei-Fei, J. Tenenbaum, and D. L. Yamins. Flexible neural representation for physics prediction. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 8799–8810, 2018. 2

[43] J. Oh, X. Guo, H. Lee, R. L. Lewis, and S. Singh. Action-conditional video prediction using deep networks in Atari games. In *Advances in neural information processing systems (NeurIPS)*, pages 2863–2871, 2015. 1

[44] M. Oliu, J. Selva, and S. Escalera. Folded recurrent neural networks for future video prediction. In *European Conference on Computer Vision (ECCV)*, pages 716–731, 2018. 2, 5

[45] R. Palm, U. Paquet, and O. Winther. Recurrent relational networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3368–3378, 2018. 2

[46] V. Patraucean, A. Handa, and R. Cipolla. Spatio-temporal video autoencoder with differentiable memory. In *ICLR 2016 Workshop Track*, 2015. 2

[47] T. Qin, K. Wu, and D. Xiu. Data driven governing equations approximation using deep neural networks. *Journal of Computational Physics*, 2019. 2

[48] M. Raissi. Deep hidden physics models: Deep learning of nonlinear partial differential equations. *The Journal of Machine Learning Research*, 19(1):932–955, 2018. 1, 2

[49] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics informed deep learning (part ii): Data-driven discovery of nonlinear partial differential equations. *arXiv preprint arXiv:1711.10566*, 2017. 2

[50] F. A. Reda, G. Liu, K. J. Shih, R. Kirby, J. Barker, D. Tarjan, A. Tao, and B. Catanzaro. SDC-Net: Video prediction using spatially-displaced convolution. In *European Conference on Computer Vision (ECCV)*, pages 718–733, 2018. 2

[51] T. Robert, N. Thome, and M. Cord. Hybridnet: Classification and reconstruction cooperation for semi-supervised learning. In *European Conference on Computer Vision (ECCV)*, pages 153–169, 2018. 3

[52] S. H. Rudy, S. L. Brunton, J. L. Proctor, and J. N. Kutz. Data-driven discovery of partial differential equations. *Science Advances*, 3(4):e1602614, 2017. 1, 2

[53] A. Sanchez-Gonzalez, N. Heess, J. T. Springenberg, J. Merel, M. Riedmiller, R. Hadsell, and P. Battaglia. Graph networks as learnable physics engines for inference and control. In *International Conference on Machine Learning (ICML)*, pages 4467–4476, 2018. 2

[54] H. Schaeffer. Learning partial differential equations via data discovery and sparse optimization. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473(2197):20160446, 2017. 2

[55] S. Seo and Y. Liu. Differentiable physics-informed graph networks. *arXiv preprint arXiv:1902.02950*, 2019. 1, 2

[56] N. Srivastava, E. Mansimov, and R. Salakhudinov. Unsupervised learning of video representations using LSTMs. In *International Conference on Machine Learning (ICML)*, pages 843–852, 2015. 2, 5

[57] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems (NeurIPS)*, pages 3104–3112, 2014. 4

[58] S. Tulyakov, M.-Y. Liu, X. Yang, and J. Kautz. Mocogan: Decomposing motion and content for video generation. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1526–1535, 2018. 1, 2

[59] S. van Steenkiste, M. Chang, K. Greff, and J. Schmidhuber. Relational neural expectation maximization: Unsupervised discovery of objects and their interactions. In *International Conference on Learning Representations (ICLR)*, 2018. 2

[60] R. Villegas, J. Yang, S. Hong, X. Lin, and H. Lee. Decomposing motion and content for natural video sequence prediction. *International Conference on Learning Representations (ICLR)*, 2017. 1, 2

[61] R. Villegas, J. Yang, Y. Zou, S. Sohn, X. Lin, and H. Lee. Learning to generate long-term future via hierarchical prediction. In *International Conference on Machine Learning (ICML)*, pages 3560–3569, 2017. 2, 7

[62] C. Vondrick, H. Pirsiavash, and A. Torralba. Generating videos with scene dynamics. In *Advances In Neural Information Processing Systems (NeurIPS)*, pages 613–621, 2016. 1, 2

[63] J. Walker, K. Marino, A. Gupta, and M. Hebert. The pose knows: Video forecasting by generating pose futures. In *International Conference on Computer Vision (ICCV)*, pages 3332–3341, 2017. 2

[64] Y. Wang, Z. Gao, M. Long, J. Wang, and P. S. Yu. PredRNN++: Towards a resolution of the deep-in-time dilemma in spatiotemporal predictive learning. *arXiv preprint arXiv:1804.06300*, 2018. 1, 2, 4, 5, 6

[65] Y. Wang, L. Jiang, M.-H. Yang, L.-J. Li, M. Long, and L. Fei-Fei. Eidetic 3D LSTM: A model for video prediction and beyond. In *International Conference on Learning Representations (ICLR)*, 2019. 2, 5

[66] Y. Wang, M. Long, J. Wang, Z. Gao, and S. Y. Philip. PredRNN: Recurrent neural networks for predictive learning using spatiotemporal lstms. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 879–888, 2017. 1, 2, 4, 5, 6

[67] Y. Wang, J. Zhang, H. Zhu, M. Long, J. Wang, and P. S. Yu. Memory in memory: A predictive neural network for learning higher-order non-stationarity from spatiotemporal dynamics. In *Computer Vision and Pattern Recognition (CVPR)*, pages 9154–9162, 2019. 1, 2, 4, 5, 6

[68] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli, et al. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. 6

[69] M. Watter, J. Springenberg, J. Boedecker, and M. Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in neural information processing systems (NeurIPS)*, pages 2746–2754, 2015. 2

[70] N. Watters, D. Zoran, T. Weber, P. Battaglia, R. Pascanu, and A. Tacchetti. Visual interaction networks: Learning a physics simulator from video. In *Advances in neural information processing systems (NeurIPS)*, pages 4539–4547, 2017. 2

[71] E. Weinan. A proposal on machine learning via dynamical systems. *Communications in Mathematics and Statistics*, 5(1):1–11, 2017. 1, 2

[72] J. Wu, E. Lu, P. Kohli, B. Freeman, and J. Tenenbaum. Learning to see physics via visual de-animation. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 153–164, 2017. 2

[73] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems (NeurIPS)*, pages 802–810, 2015. 1, 2, 3, 5, 6

[74] J. Xu, B. Ni, Z. Li, S. Cheng, and X. Yang. Structure preserving video prediction. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1460–1469, 2018. 2

[75] T. Xue, J. Wu, K. Bouman, and B. Freeman. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. In *Advances in neural information processing systems (NeurIPS)*, pages 91–99, 2016. 1, 2

[76] Y. Ye, M. Singh, A. Gupta, and S. Tulsiani. Compositional video prediction. In *Computer Vision and Pattern Recognition (CVPR)*, pages 10353–10362, 2019. 2

[77] J. Zhang, Y. Zheng, and D. Qi. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 5

[78] M. Zhu, B. Chang, and C. Fu. Convolutional neural networks combined with Runge-Kutta methods. In *International Conference on Learning Representations (ICLR)*, 2019. 2

Disentangling Physical Dynamics from Unknown Factors for Unsupervised Video Prediction
Supplementary Material

Vincent Le Guen [1,2], Nicolas Thome [2]
[1] EDF R&D, Chatou, France
[2] CEDRIC, Conservatoire National des Arts et Métiers, Paris, France

1. PhyDNet model

1.1. Discrete PhyCell derivation

PhyCell dynamics is governed by the PDE:

$$\begin{aligned}\frac{\partial \mathbf{h}}{\partial t}(t,\mathbf{x}) &= \Phi(\mathbf{h}) + \mathcal{C}(\mathbf{h},\mathbf{u}) \\ &= \Phi(\mathbf{h}(t,\mathbf{x})) + \mathbf{K}(t,\mathbf{x}) \odot \\ &\quad (\mathbf{E}(\mathbf{u}(t,\mathbf{x})) - (\mathbf{h}(t,\mathbf{x}) + \Phi(\mathbf{h}(t,\mathbf{x}))))\end{aligned}$$

By Euler discretization $\frac{\partial \mathbf{h}}{\partial t} = \delta \mathbf{h}_t = \mathbf{h}_t - \mathbf{h}_{t-1}$, we get:

$$\begin{aligned}\mathbf{h}_{t+1} - \mathbf{h}_t &= \Phi(\mathbf{h}_t) + \mathbf{K}_t \odot (\mathbf{E}(\mathbf{u}_t) - (\mathbf{h}_t + \Phi(\mathbf{h}_t))) \\ \mathbf{h}_{t+1} &= \mathbf{h}_t + \Phi(\mathbf{h}_t) + \mathbf{K}_t \odot (\mathbf{E}(\mathbf{u}_t) - (\mathbf{h}_t + \Phi(\mathbf{h}_t))) \\ \mathbf{h}_{t+1} &= (1 - \mathbf{K}_t) \odot (\mathbf{h}_t + \Phi(\mathbf{h}_t)) + \mathbf{K}_t \odot \mathbf{E}(\mathbf{u}_t)\end{aligned}$$

1.2. Moment matrix

For a filter $\mathbf{w}$ of size $k \times k$, the moment matrix $\mathbf{M}(\mathbf{w})$ is a matrix of size $k \times k$ defined as:

$$\mathbf{M}(\mathbf{w})_{i,j} = \frac{1}{i!j!} \sum_{u=-\frac{k-1}{2}}^{\frac{k-1}{2}} \sum_{v=-\frac{k-1}{2}}^{\frac{k-1}{2}} u^i v^j \mathbf{w}[u,v]$$

for $i, j = 0, ..., k-1$.

For any function $h : \mathbb{R}^2 \longrightarrow \mathbb{R}$, we consider the convolution of h with the filter $\mathbf{w}$. Taylor's expansion gives:

$$\begin{aligned}&\sum_{u=-\frac{k-1}{2}}^{\frac{k-1}{2}} \sum_{v=-\frac{k-1}{2}}^{\frac{k-1}{2}} \mathbf{w}[u,v] h(x + \delta x.u, y + \delta y.v) \\ &= \sum_{u=-\frac{k-1}{2}}^{\frac{k-1}{2}} \sum_{v=-\frac{k-1}{2}}^{\frac{k-1}{2}} \mathbf{w}[u,v] \sum_{i,j=1}^{k-1} \frac{\partial^{i+j} h}{\partial x^i \partial y^j}(x,y) \frac{u^i v^j}{i!j!} \delta x^i \delta y^j \\ &\quad + o(|\delta x|^{k-1} + |\delta y|^{k-1}) \\ &= \sum_{i,j=1}^{k-1} \mathbf{M}(\mathbf{w})_{i,j} \delta x^i \delta y^j \frac{\partial^{i+j} h}{\partial x^i \partial y^j}(x,y) + o(|\delta x|^{k+1} + |\delta y|^{k-1})\end{aligned}$$

This equation shows that we can control the differential order approximated by the filter $\mathbf{w}$ by imposing constraints on its moment matrix $\mathbf{M}(\mathbf{w})$. For example, in order to approximate the differential operator $\frac{\partial^{a+b}}{\partial x^a \partial y^b}(.)$, it suffices to impose $\mathbf{M}(\mathbf{w})_{i,j} = 0$ for $i \neq a$ and $j \neq b$. By denoting $\mathbf{\Delta}^k_{i,j}$ the Kronecker matrix of size $k \times k$, which equals 1 at position (i,j) and 0 elsewhere, we thus enforce the moment matrix $\mathbf{M}(\mathbf{w})$ to match the target $\mathbf{\Delta}^k_{a,b}$ with the Frobenius norm. This justifies the choice of our moment loss for enforcing each filter $\mathbf{w}^k_{p,i,j}$ to approximate the corresponding derivative $\frac{\partial^{i+j}}{\partial x^i \partial y^j}(.)$:

$$\mathcal{L}_{\text{moment}} = \sum_{i \leq k} \sum_{j \leq k} ||\mathbf{M}(\mathbf{w}^k_{p,i,j}) - \mathbf{\Delta}^k_{i,j}||_F$$

1.3. Prediction mode training

We show in section 1.3.1 that the decomposition $\boldsymbol{\mathcal{M}}_r(\mathbf{h},\mathbf{u}) = \Phi(\mathbf{h}) + \mathcal{C}(\mathbf{h},\mathbf{u})$ still holds for standard Seq2Seq models (RNN, GRU, LSTM). As mentioned in the submission, the resulting predictor Φ is, however, naive and useless for multi-step prediction, *i.e.* $\Phi(\mathbf{h}) = -\mathbf{h}$ and $\tilde{\mathbf{h}}_{t+1} = 0$.

In multi-step prediction, the option followed by standard Seq2seq models is to recursively reinject back predictions as ground truth input for the next time steps. Scheduled Sampling [1] is a solution to mitigate error accumulation and train/test discrepancy, that we use in our ConvLSTM branch. This is, however, inferior to the results obtained with our PhyCell trained in the "prediction-only" mode, as shown in the section 4.4 of the submission.

1.3.1 PDE formulation for standard RNNs

Vanilla RNN The equations for the vanilla RNN are:

$$\mathbf{h}_t = \tanh(\mathbf{W}_h \mathbf{h}_{t-1} + \mathbf{W}_u \mathbf{u}_t + \mathbf{b})$$

with weight matrices $\mathbf{W}_h$, $\mathbf{W}_u$ and bias $\mathbf{b}$.
By approximating $\frac{\partial \mathbf{h}}{\partial t} = \delta \mathbf{h}_t = \mathbf{h}_t - \mathbf{h}_{t-1}$, we get the PDE:

$$\begin{aligned}\frac{\partial \mathbf{h}}{\partial t}(t, \mathbf{x}) &= \boldsymbol{\mathcal{M}}(\mathbf{h}, \mathbf{u}) \\ &= \tanh(\mathbf{W}_h \mathbf{h}(t) + \mathbf{W}_u \mathbf{u}(t) + \mathbf{b}) - \mathbf{h}(t)\end{aligned}$$

A linear decoupling of this PDE is

$$\frac{\partial \mathbf{h}}{\partial t}(t, \mathbf{x}) = \Phi(\mathbf{h}) + \mathcal{C}(\mathbf{h}, \mathbf{u})$$

with $\Phi(\mathbf{h}) = -\mathbf{h}(t)$ and $\mathcal{C}(\mathbf{h}, \mathbf{u}) = \tanh(\mathbf{W}_h \mathbf{h}(t) + \mathbf{W}_u \mathbf{u}(t) + \mathbf{b})$ which gives in discrete time the prediction-correction scheme:

$$\begin{cases} \tilde{\mathbf{h}}_{t+1} = 0 & (1) \\ \mathbf{h}_{t+1} = \tilde{\mathbf{h}}_{t+1} + \tanh\left(\mathbf{W}_h \mathbf{h}_{t-1} + \mathbf{W}_u \mathbf{u}_t + \mathbf{b}\right) & (2) \end{cases}$$

We see that the prior predictor Φ brings no information and that the correction step drives the whole dynamics.

Gated Recurrent Unit (GRU) The equations of the Gated Recurrent Unit [2] are:

$$\begin{aligned}\mathbf{r}_t &= \sigma(\mathbf{W}_{rh} \mathbf{h}_{t-1} + \mathbf{W}_{ru} \mathbf{u}_t + \mathbf{b}_r) \\ \mathbf{z}_t &= \sigma(\mathbf{W}_{zh} \mathbf{h}_{t-1} + \mathbf{W}_{zu} \mathbf{u}_t + \mathbf{b}_z) \\ \mathbf{g}_t &= \tanh(\mathbf{W}_{gh}(\mathbf{r}_t \odot \mathbf{h}_{t-1}) + \mathbf{W}_{gu} \mathbf{u}_t + \mathbf{b}_g) \\ \mathbf{h}_t &= \mathbf{z}_t \odot \mathbf{h}_{t-1} + (1 - \mathbf{z}_t) \odot \mathbf{g}_t\end{aligned}$$

where $\mathbf{r}_t$ is the reset gate, $\mathbf{z}_t$ is the update gate and $\mathbf{g}_t$ is the update vector.
By approximating $\frac{\partial \mathbf{h}}{\partial t} = \delta \mathbf{h}_t = \mathbf{h}_t - \mathbf{h}_{t-1}$, we get the PDE:

$$\begin{aligned}\frac{\partial \mathbf{h}}{\partial t}(t, \mathbf{x}) &= \boldsymbol{\mathcal{M}}(\mathbf{h}, \mathbf{u}) \\ &= \mathbf{z}(t) \odot \mathbf{h}(t) + (1 - \mathbf{z}(t)) \odot \mathbf{g}(t) - \mathbf{h}(t)\end{aligned}$$

A linear decoupling of this PDE is

$$\frac{\partial \mathbf{h}}{\partial t}(t, \mathbf{x}) = \Phi(\mathbf{h}) + \mathcal{C}(\mathbf{h}, \mathbf{u})$$

with $\Phi(\mathbf{h}) = -\mathbf{h}(t)$ and $\mathcal{C}(\mathbf{h}, \mathbf{u}) = \mathbf{z}(t) \odot \mathbf{h}(t) + (1 - \mathbf{z}(t)) \odot \mathbf{g}(t)$ which gives in discrete time the prediction-correction scheme:

$$\begin{cases} \tilde{\mathbf{h}}_{t+1} = 0 & (3) \\ \mathbf{h}_{t+1} = \tilde{\mathbf{h}}_{t+1} + \mathbf{z}_t \odot \mathbf{h}_{t-1} + (1 - \mathbf{z}_t) \odot \mathbf{g}_t & (4) \end{cases}$$

We again see that the prior predictor Φ brings no information and that the correction step drives the whole dynamics.

Long Short-Term Memory (LSTM) We give the formulation for the standard LSTM [5] (the ConvLSTM [13] can be immediately deduced by replacing matrix products by convolutions). The LSTM equations are:

$$\begin{aligned}\mathbf{i}_t &= \sigma(\mathbf{W}_{ih} \mathbf{h}_{t-1} + \mathbf{W}_{iu} \mathbf{u}_t + \mathbf{b}_i) \\ \mathbf{f}_t &= \sigma(\mathbf{W}_{fh} \mathbf{h}_{t-1} + \mathbf{W}_{fu} \mathbf{u}_t + \mathbf{b}_f) \\ \mathbf{g}_t &= \tanh(\mathbf{W}_{gh} \mathbf{h}_{t-1} + \mathbf{W}_{gu} \mathbf{u}_t + \mathbf{b}_g) \\ \mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t \\ \mathbf{o}_t &= \sigma(\mathbf{W}_{oh} \mathbf{h}_{t-1} + \mathbf{W}_{ou} \mathbf{u}_t + \mathbf{b}_o) \\ \mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t)\end{aligned}$$

where $\mathbf{i}_t$ is the input gate, $\mathbf{f}_t$ the forget gate, $\mathbf{g}_t$ the input-modulation gate, $\mathbf{o}_t$ the output gate, $\mathbf{c}_t$ the cell state and $\mathbf{h}_t$ the latent state. We define the LSTM augmented latent state as:

$$\bar{\mathbf{h}} = \begin{pmatrix} \mathbf{g} \\ \mathbf{c} \end{pmatrix}$$

The augmented state $\bar{\mathbf{h}}$ thus verifies the PDE:

$$\frac{\partial \bar{\mathbf{h}}}{\partial t} = \begin{pmatrix} \frac{\partial \mathbf{h}}{\partial t} \\ \frac{\partial \mathbf{c}}{\partial t} \end{pmatrix} = \begin{pmatrix} \mathbf{o}(t) \odot \tanh(\mathbf{c}(t)) - \mathbf{h}(t)) \\ \mathbf{f}(t) \odot \mathbf{c}(t) + \mathbf{i}(t) \odot \mathbf{g}(t) - \mathbf{c}(t) \end{pmatrix}$$

A linear decoupling of this PDE is

$$\frac{\partial \bar{\mathbf{h}}}{\partial t}(t, \mathbf{x}) = \Phi(\bar{\mathbf{h}}) + \mathcal{C}(\bar{\mathbf{h}}, \mathbf{u})$$

with $\Phi(\bar{\mathbf{h}}) = -\bar{\mathbf{h}}(t)$ and

$$\mathcal{C}(\bar{\mathbf{h}}, \mathbf{u}) = \begin{pmatrix} \mathbf{o}(t) \odot \tanh(\mathbf{c}(t)) \\ \mathbf{f}(t) \odot \mathbf{c}(t) + \mathbf{i}(t) \odot \mathbf{g}(t) \end{pmatrix}$$

which gives in discrete time the prediction-correction scheme:

$$\begin{cases} \tilde{\bar{h}}_{t+1} = 0 & (5) \\ \bar{\mathbf{h}}_{t+1} = \tilde{\bar{h}}_{t+1} + \begin{pmatrix} \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \\ \mathbf{f}_t \odot \mathbf{c}_t + \mathbf{i}_t \odot \mathbf{g}_t \end{pmatrix} & (6) \end{cases}$$

We again see that the prior predictor Φ brings no information and that the correction step drives the whole dynamics.

2. Experiments

2.1. Datasets

Moving MNIST is a standard benchmark in video prediction [8] consisting in two random MNIST digits bouncing on the walls of a 64×64 grid. We predict 10 future frames given 10 input frames. Training sequences are generated on the fly and the test set of 10000 sequences is provided by [8].

Traffic BJ consists in traffic flow data collected by taxicabs in Beijing [14]. Each 32×32 image is a 2-channels heat map with leaving/entering traffic. Video prediction on such real-world complex data require modeling transport phenomena and traffic diffusion. Following the setting of [14, 11, 10], we predict 4 future frames given 4 input frames.

SST consists in daily Sea Surface Temperature (SST) data from the sophisticated simulation engine NEMO (Nucleus for European Modeling of the Ocean), as in [3]. SST evolution is governed by the physical laws of fluid dynamics. We predict 4 frames of size 64×64 given 4 input frames.

Human 3.6 contains 3.6 million images of human actions [7]. Following the setting of [11], we use only the "walking" scenario with subjects S1, S5, S6, S7, S8 for training, and S9, S11 for testing. We predict 4 future images of size $128 \times 128 \times 3$ given 4 input images.

2.2. Model architectures and training

Model architectures

We give here the architecture of the encoder and decoder for all datasets. They share common building blocs, composed of convolutions, GroupNorm activation functions [12] and LeakyRelu non-linearities. We define:

- conv-block(input, output, stride) = {Conv2D + GroupNorm + LeakyRelu(0.2)}
- upconv-block(input,output,stride)={TransposedConv2D + GroupNorm + LeakyRelu(0.2) }
- upconv(input,output,stride)=TransposedConv2D(input, output, stride)

For each of the following architectures, we use skip connections from the encoder to the decoder, as classically done, *e.g.* in [4].

Moving MNIST:

Encoder	Decoder
conv-block(1,8,1)	upconv-block(128,64,1)
conv-block(8,16,1)	upconv-block(128,32,2)
conv-block(16,32,2)	upconv-block(64,32,1)
conv-block(32,32,1)	upconv-block(64,16,2)
conv-block(32,64,2)	upconv-block(32,8,1)
conv-block(64,64,1)	upconv(16,1,1)

Traffic:

Encoder	Decoder
conv-block(2,32,1)	upconv-block(256,64,1)
conv-block(32.64,2)	upconv-block(128,32,2)
conv-block(64,128,1)	upconv(64,2,1)

SST:

Encoder	Decoder
conv-block(1,32,1)	upconv-block(256,64,1)
conv-block(32.64,2)	upconv-block(128,32,2)
conv-block(64,128,1)	upconv(64,1,1)

Human 3.6:

Encoder	Decoder
conv-block(3,16,1)	upconv-block(256,128,1)
conv-block(16,32,1)	upconv-block(256,64,2)
conv-block(32,64,2)	upconv-block(128,64,1)
conv-block(64,64,1)	upconv-block(128,32,2)
conv-block(64,128,2)	upconv-block(64,16,1)
conv-block(128,128,1)	upconv(32,3,1)

Influence of λ

We show in Figure 1 the influence of parameter λ balancing $\mathcal{L}_{\text{image}}$ and $\mathcal{L}_{\text{moment}}$ when training PhyDNet for Moving MNIST dataset. When λ decreases towards 0, MSE tends towards the unconstrained case at 29. MSE reaches a minimum around $\lambda = 1$. When λ further increases, physical regularization is too high and MSE increases above 30. In the paper, we fix $\lambda = 1$ for all datasets.

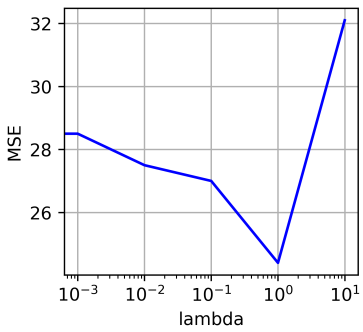


Figure 1. Influence of hyperparameter λ when training PhyDNet for Moving MNIST dataset.

2.3. State-of-the art comparison

We show here that PhyDNet results are equivalent on Human 3.6 to a recent baseline that explicitly uses additional human pose annotations [9]. In the supplementary of

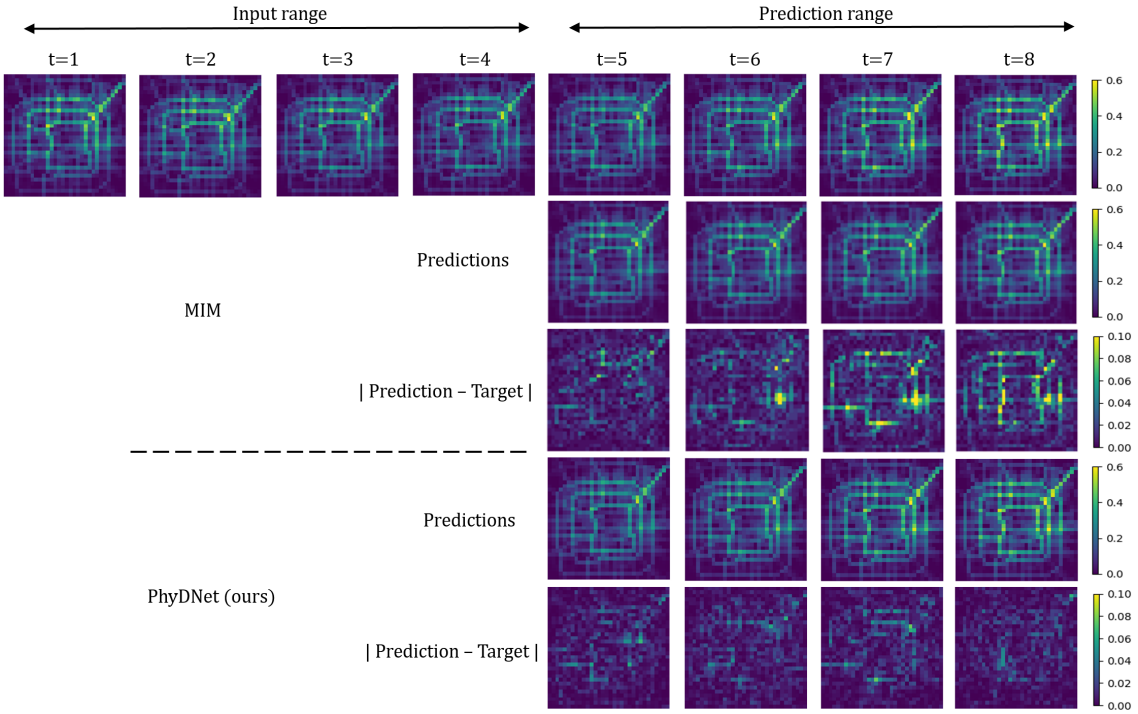


Figure 2. Additional qualitative results for Traffic BJ and comparison to Memory In Memory [11]. We see that PhyDNet absolute error are smaller than MIM errors, and independent of the spatial structure of the road network.

their paper [9], the authors evaluate their model with Peak Signal over Noise Ratios (PSNR) curves with respect to the forecasting horizon for all deciles of motion in Human 3.6 videos. Regarding prediction horizon up to $\Delta = 4$, their method obtains a PSNR always below 21 and around 22 for the 1^{st} decile (with the least human motion). In comparison, PhyDNet attains a per-frame MSE of 369, corresponding to a PSNR of 21.2. This shows that PhyDNet performs similarly than [9] for the prediction horizon considered, without requiring additional human pose annotations.

2.4. Additional visualisations

To complement Figure 4 in submission, we give further qualitative prediction of PhyDNet on Traffic BJ (Figure 2) with a comparison with Memory in Memory [11] that is state-of-the-art for this dataset. We see that PhyDNet leads to sharper results and a lower absolute error. Interestingly, PhyDNet absolute errors are approximately spatially independent, whereas MIM errors tend to be higher at a few keys locations of Beijing road network.

We also provide additional prediction visualisations for Sea Surface Temperature (Figure 3) and Human 3.6 (Figure 4) which confirm the good behaviour of PhyDNet.

We add a detailed qualitative comparison to DDPAE in Figure 5. DDPAE is a specific disentangling method for Moving MNIST that extracts the positions of the two digits and tracks them with a predictive recurrent neural network. In this example, DDPAE fails to disentangle the two digits (components 1 and 2) in Figure 5 when they overlap in the input sequence, resulting in blurry predictions. In contrast, PhyDNet successfully learns a latent space in which the two digits are disentangled, resulting in far better predictions in terms of sharpness and position of the digits.

2.5. Ablation study

We give in Figure 6 additional visualisations completing Figure 5 in submission. We qualitatively analyze partial predictions of PhyDNet for the physical branch $\hat{\mathbf{u}}^{\mathbf{p}}_{t+1} = \mathbf{D}(\mathbf{h}^{\mathbf{p}}_{t+1})$ and residual branch $\hat{\mathbf{u}}^{\mathbf{r}}_{t+1} = \mathbf{D}(\mathbf{h}^{\mathbf{r}}_{t+1})$. For Moving MNIST (a) and Human 3.6 (d), $\mathbf{h}^{\mathbf{p}}$ captures coarse localisations of objects, while $\mathbf{h}^{\mathbf{r}}$ captures fine-grained details that are not useful for the physical model. For Traffic BJ, $\mathbf{h}^{\mathbf{p}}$ captures the main patterns of the road network, while $\mathbf{h}^{\mathbf{r}}$ models remaining details. Finally for SST, the visual difference between $\mathbf{h}^{\mathbf{p}}$ and $\mathbf{h}^{\mathbf{r}}$ is slighter, but the cooperation between both branches is crucial, as shown by quantitatives results.

2.6. Influence of physical regularization

We provide the detailed ablation study for all datasets in Table 1 that complements Table 3 in submission. When we disable $\mathcal{L}_{\text{moment}}$ for training PhyCell, performances improve for all datasets (improvement of 7 MSE points for Moving MNIST, 5 points for Traffic BJ, 3 points for SST and Hu-

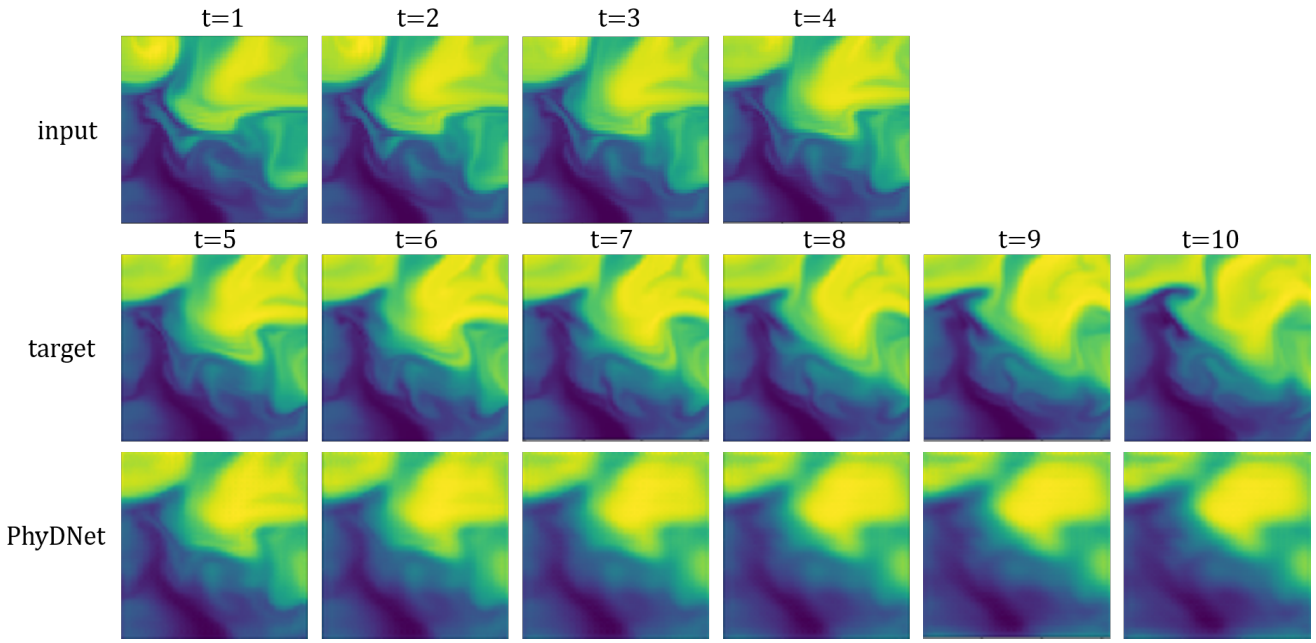


Figure 3. Additional qualitative results for Sea Surface Temperature.

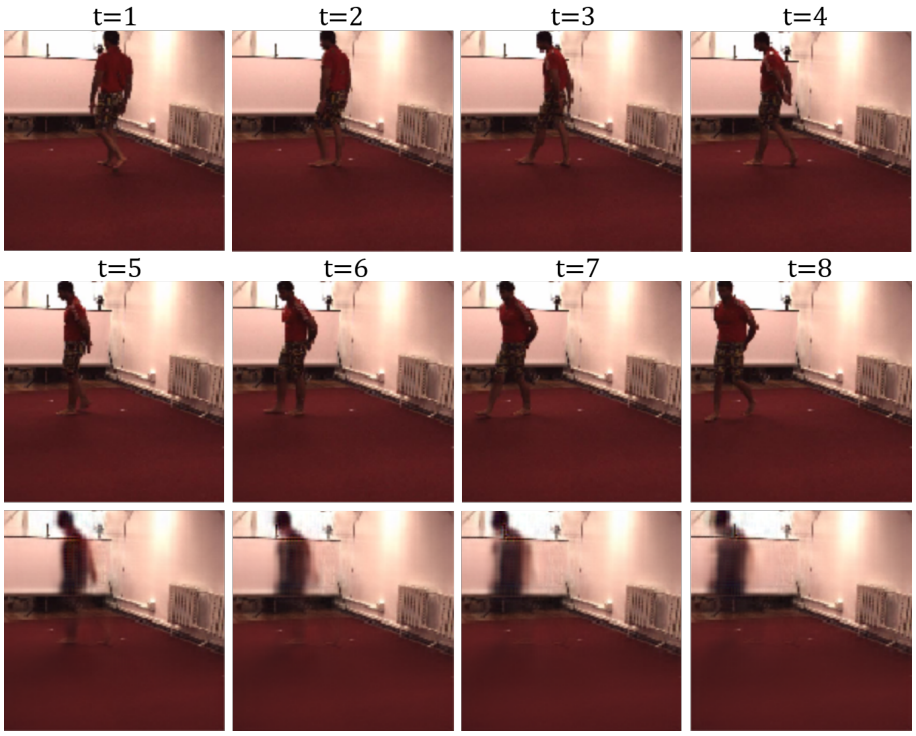


Figure 4. Additional qualitative results for Human 3.6.

man 3.6). This again shows that physical constraints alone are too restrictive for learning dynamics in a general context, where other factors are required for prediction. When we further include PhyCell in our two-branches disentangling architecture PhyDNet, there is another huge performance gain compared to PhyCell (improvement of 25 MSE points on Moving MNIST, 7 points for Traffic and SST, 5 points for Human 3.6). We also remark that when we disable $\mathcal{L}_{\text{moment}}$ for training PhyDNet, we get worse performances (drop of 5 MSE points for Moving MNIST and 2

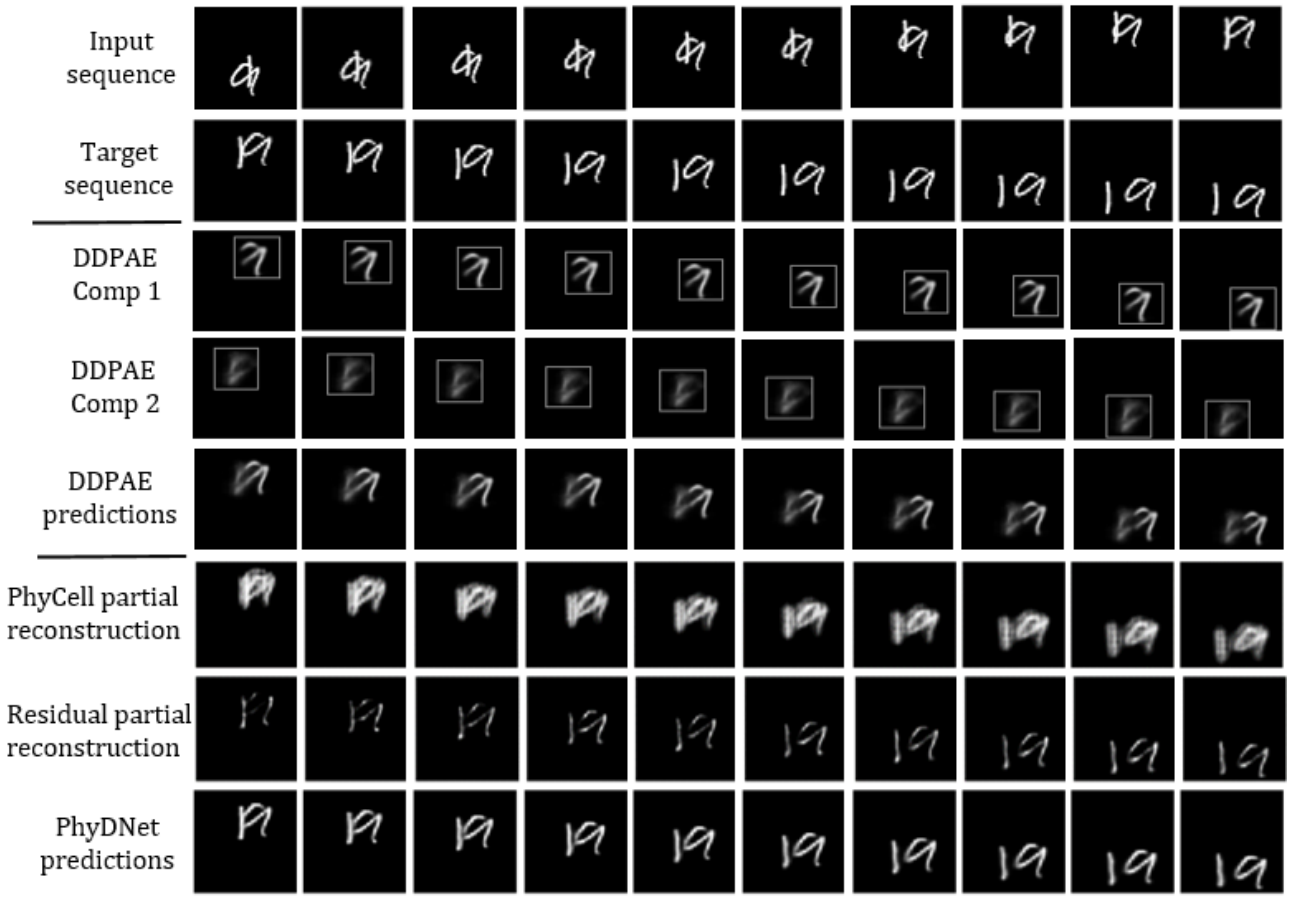


Figure 5. Detailed qualitative comparison to DDPAE [6] on Moving MNIST dataset.

points for Traffic) or equivalent performances (difference below 0.5 MSE point for SST and Human 3.6). This again confirms the relevance of physical constraints. To complement the discussion of Table 3 in submission, we give here in Table 2 the approximate number of models parameters of trained models:

method	number of parameters
ConvLSTM	3.10^6
PhyCell	370.10^3
PhyDNet	3.10^6

Table 2. Number of parameters of models trained on Moving MNIST

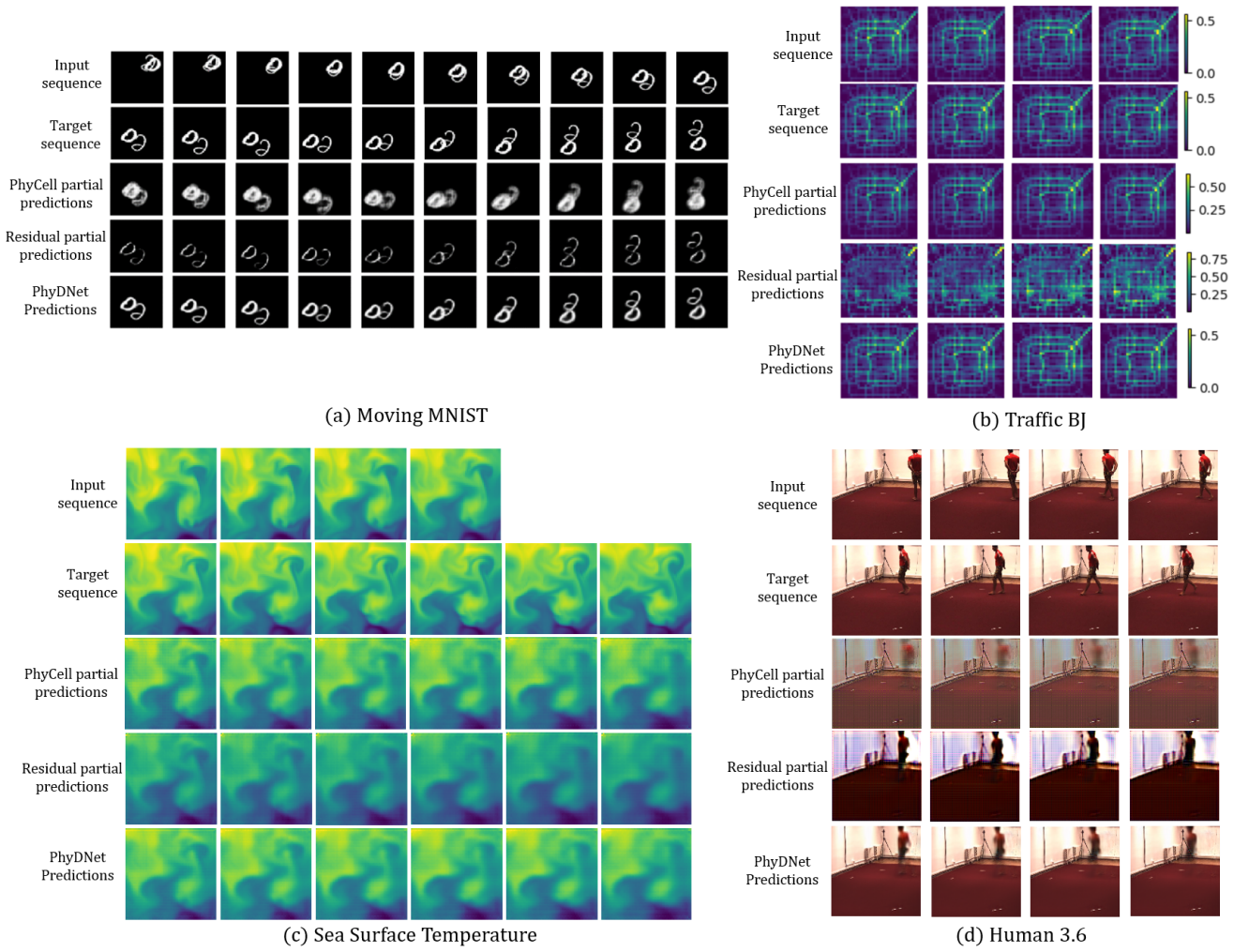
We see that a 1-layer PhyCell with 49 filters has far fewer parameters than a 3-layers ConvLSTM (with 128 filters in each layer) and obtains far better results (gain of 50 MSE points). Then PhyDNet with approximately the same number of parameters as ConvLSTM (3 million) again improves the performances by 25 MSE points, reaching a state-of-the-art MSE score of 24.4.

2.7. Dealing with unreliable inputs

In section 4.4.2 of the submission, we discuss the advantages of the "prediction only" of PhyDNet when dealing with unreliable inputs. We compared PhyDNet with DDPAE [6] and show a MSE comparison in the context of

Method	**Moving MNist**			**Traffic BJ**			**Sea Surface Temperature**			**Human 3.6**		
	MSE	MAE	SSIM	MSE $\times 100$	MAE	SSIM	MSE $\times 10$	MAE	SSIM	MSE /10	MAE /100	SSIM
ConvLSTM	103.3	182.9	0.707	48.5*	17.7*	0.978*	45.6*	63.1*	0.949*	50.4*	18.9*	0.776*
PhyCell	50.8	129.3	0.870	48.9	17.9	0.978	38.2	60.2	0.969	42.5	18.3	0.891
PhyCell without $\mathcal{L}_{\text{moment}}$	43.4	112.8	0.895	43.6	16.89	0.980	35.4	56.0	0.970	39.6	17.4	0.894
PhyDNet	**24.4**	**70.3**	**0.947**	**41.9**	**16.2**	**0.982**	**31.9**	53.3	**0.972**	36.9	16.2	0.901
PhyDNet without $\mathcal{L}_{\text{moment}}$	29.0	81.2	0.934	43.9	16.6	0.981	32.3	**53.1**	0.971	**36.7**	**15.9**	**0.904**

Table 1. A detailed ablation study shows the impact of the physical regularization $\mathcal{L}_{\text{moment}}$ on the performances of PhyCell and PhyDNet for all datasets.

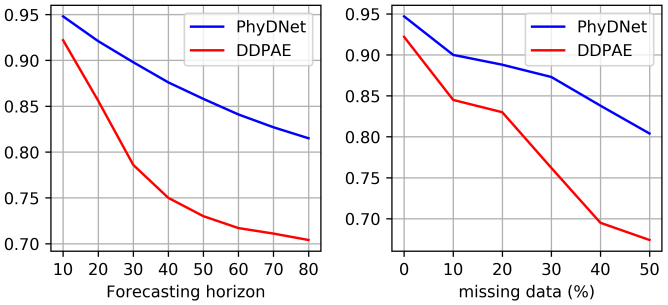


(a) Moving MNIST (b) Traffic BJ

(c) Sea Surface Temperature (d) Human 3.6

Figure 6. Additional ablation visualisations for all datasets.

long-term forecasting and missing data. Here we show in Figure 7 the SSIM results of this experiment. We can see that the performance drop of DDPAE is much more pronounced when the forecasting horizon or the missing data rate increases, which confirms the good behaviour of PhyDNet.



Long-term forecasting Missing data

Figure 7. SSIM comparison between PhyDNet and DDPAE [6] when dealing with unreliable inputs.

References

[1] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Informa-*

tion Processing Systems, pages 1171–1179, 2015. 1

[2] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014. 2

[3] Emmanuel de Bezenac, Arthur Pajot, and Patrick Gallinari. Deep learning for physical processes: Incorporating prior scientific knowledge. *arXiv preprint arXiv:1711.07970*, 2017. 3

[4] Emily L Denton et al. Unsupervised learning of disentangled representations from video. In *Advances in neural information processing systems*, pages 4414–4423, 2017. 3

[5] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 2

[6] Jun-Ting Hsieh, Bingbin Liu, De-An Huang, Li F Fei-Fei, and Juan Carlos Niebles. Learning to decompose and disentangle representations for video prediction. In *Advances in Neural Information Processing Systems*, pages 517–526, 2018. 6, 7

[7] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE transactions on pattern analysis and machine intelligence*, 36(7):1325–1339, 2013. 3

[8] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In *International Conference on Machine Learning*, pages 843–852, 2015. 2

[9] Ruben Villegas, Jimei Yang, Yuliang Zou, Sungryull Sohn, Xunyu Lin, and Honglak Lee. Learning to generate long-term future via hierarchical prediction. In *ICML*, pages 3560–3569, 2017. 3, 4

[10] Yunbo Wang, Lu Jiang, Ming-Hsuan Yang, Li-Jia Li, Mingsheng Long, and Li Fei-Fei. Eidetic 3d lstm: A model for video prediction and beyond. 2018. 3

[11] Yunbo Wang, Jianjin Zhang, Hongyu Zhu, Mingsheng Long, Jianmin Wang, and Philip S Yu. Memory in memory: A predictive neural network for learning higher-order non-stationarity from spatiotemporal dynamics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9154–9162, 2019. 3, 4

[12] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 3–19, 2018. 3

[13] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810, 2015. 2

[14] Junbo Zhang, Yu Zheng, and Dekang Qi. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 3