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:

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

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.

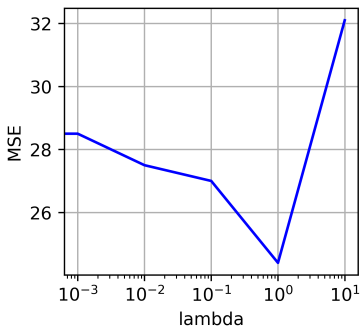


Figure 1. Influence of hyperparameter λ when training PhyDNet for Moving MNIST dataset.

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

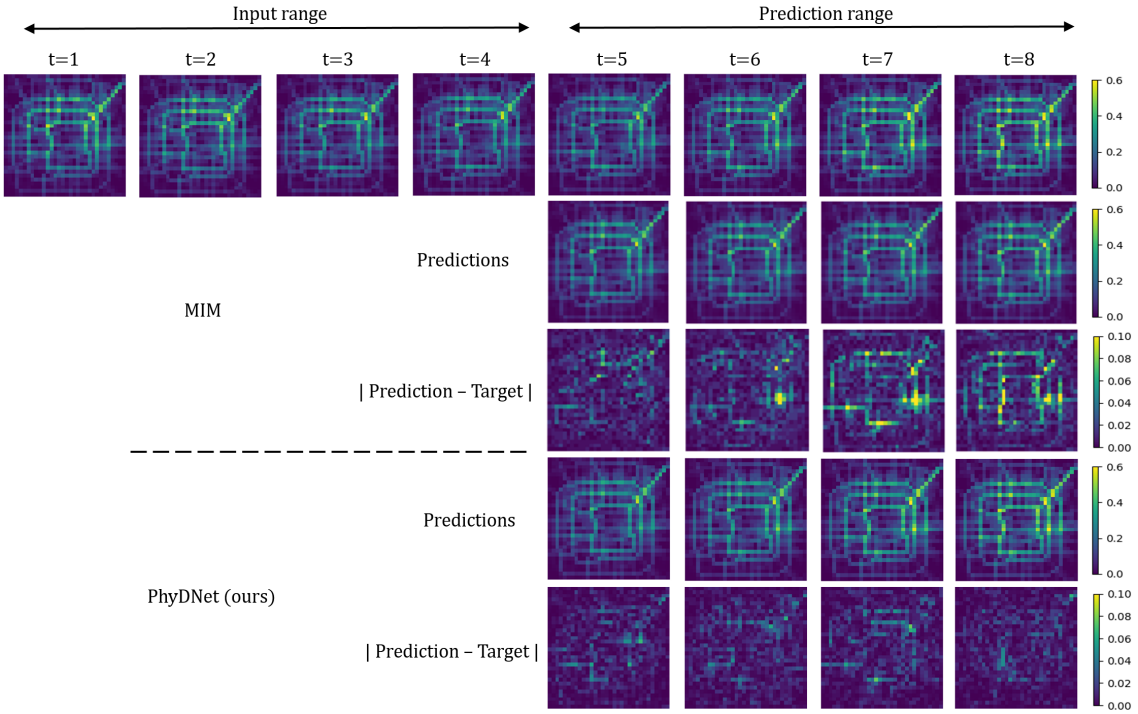


Figure 2. Additional qualitative results for Traffic BJ and comparison to Memory In Memory [11]. We see that PhyDNet absolute error are smaller than MIM errors, and independent of the spatial structure of the road network.

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-

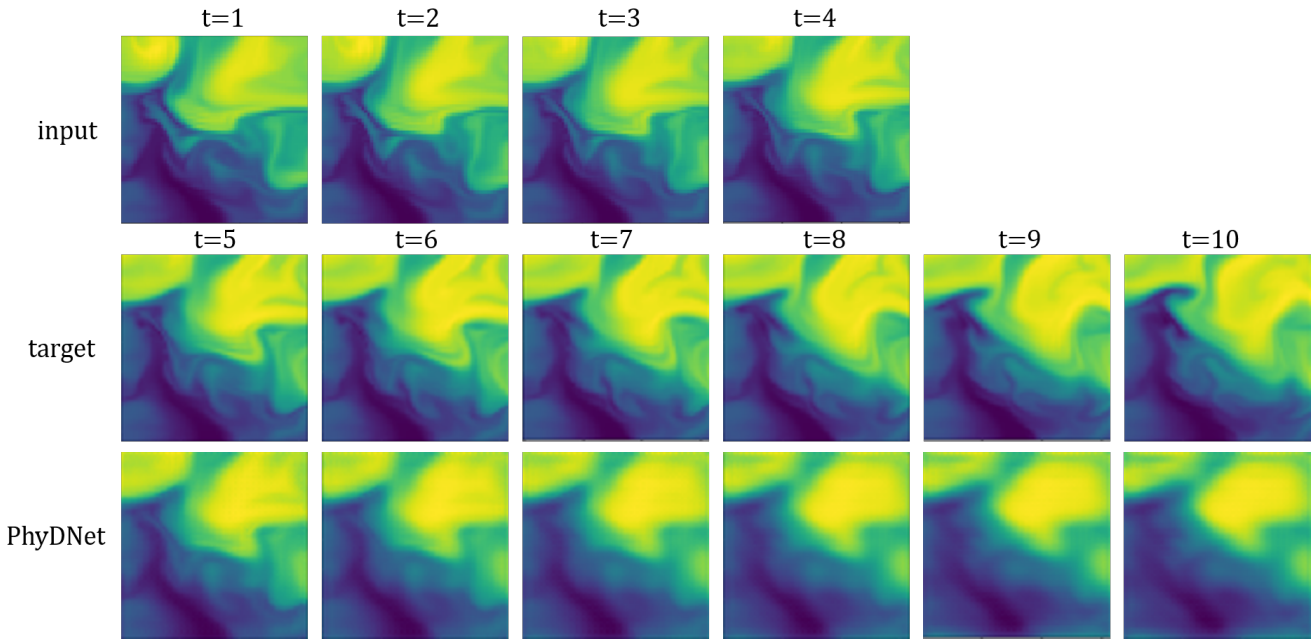


Figure 3. Additional qualitative results for Sea Surface Temperature.

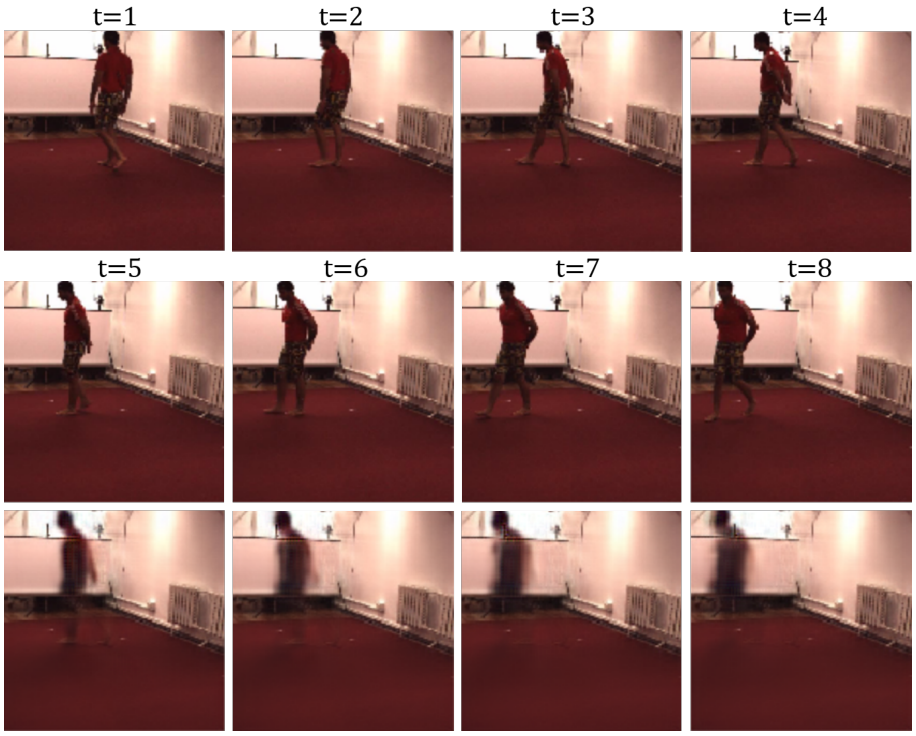


Figure 4. Additional qualitative results for Human 3.6.

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

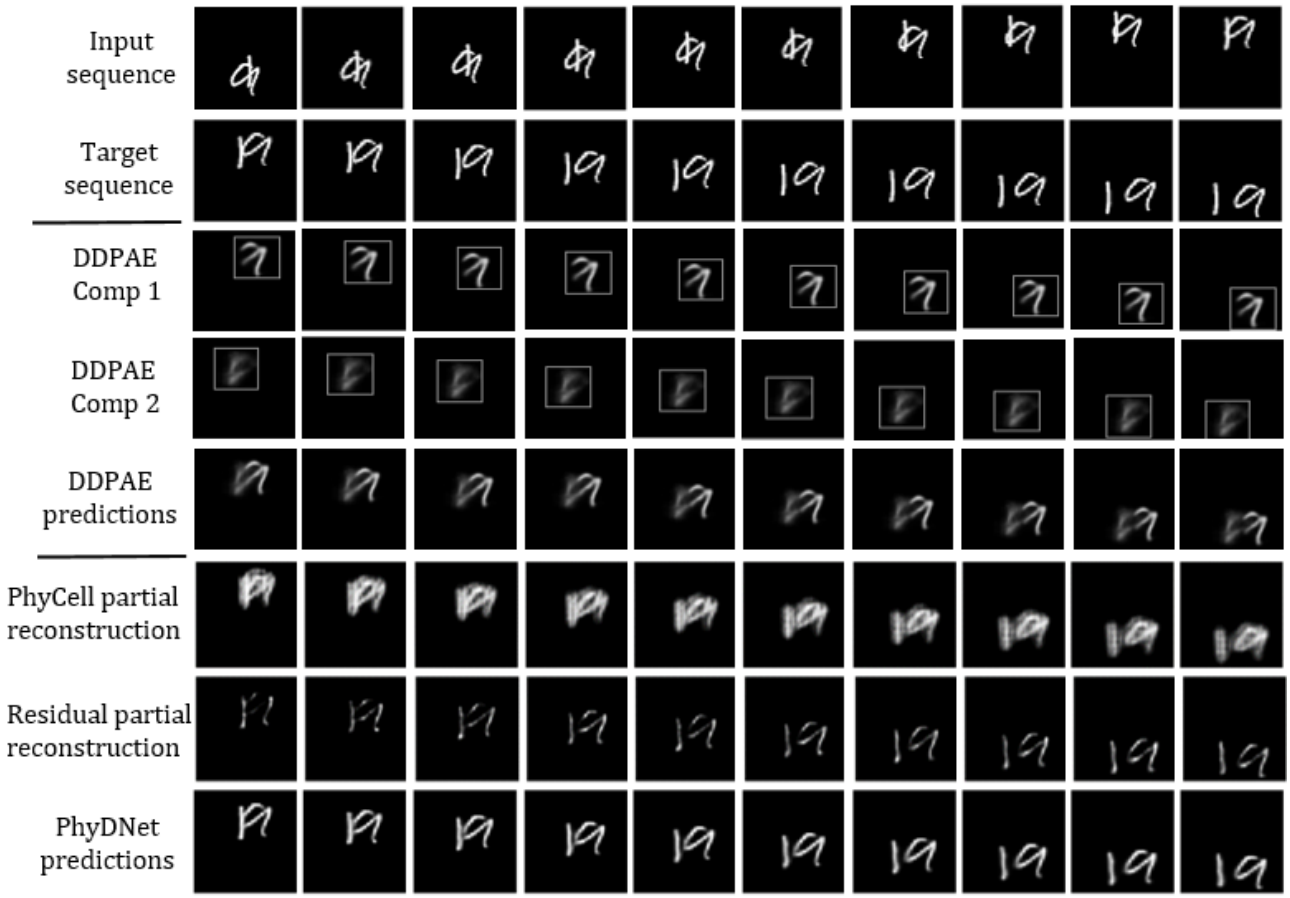


Figure 5. Detailed qualitative comparison to DDPAE [6] on Moving MNIST dataset.

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

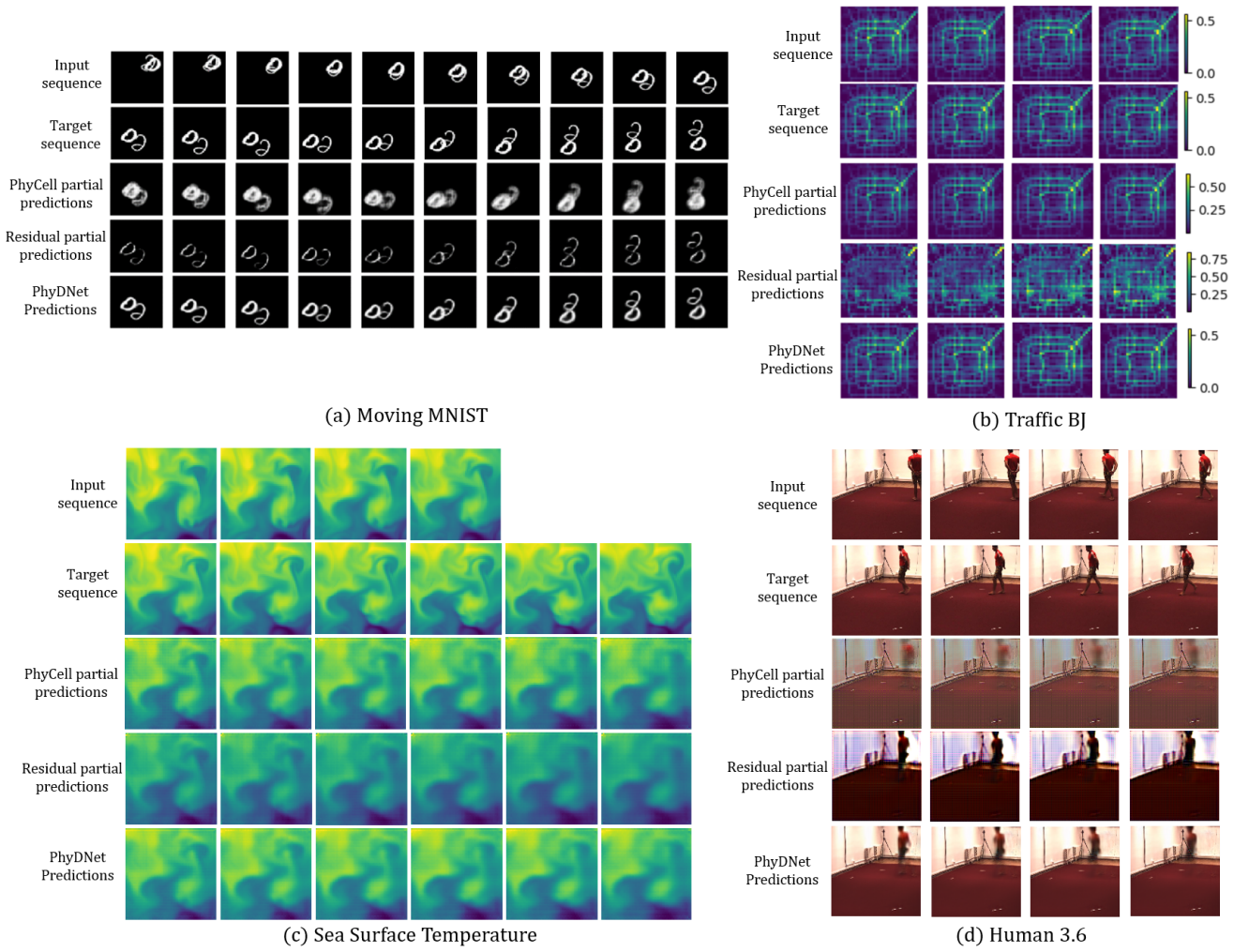
We see that a 1-layer PhyCell with 49 filters has far fewer parameters than a 3-layers ConvLSTM (with 128 filters in each layer) and obtains far better results (gain of 50 MSE points). Then PhyDNet with approximately the same number of parameters as ConvLSTM (3 million) again improves the performances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

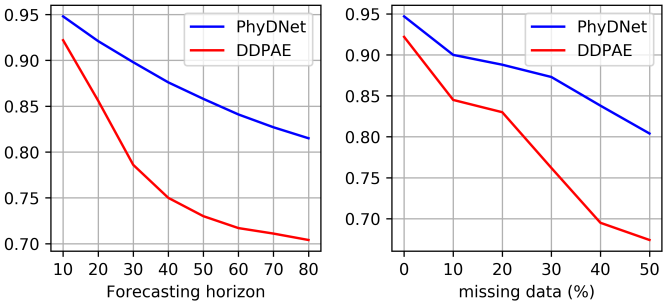


(a) Moving MNIST (b) Traffic BJ

(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