Journal of Statistical Mechanics: Theory and Experiment
An IOP and SISSA journal

Deep learning for physical processes: incorporating prior scientific knowledge*

Emmanuel de Bézenac[1,3], **Arthur Pajot**[1,3] and **Patrick Gallinari**[2]

[1] Sorbonne Universites, UMR 7606, LIP6, F-75005 Paris, France
[2] Sorbonne Université, UMR 7606, LIP6, Paris and Criteo AI Lab, Paris, France
E-mail: emmanuel.de-bezenac@lip6.fr, arthur.pajot@lip6.fr
and patrick.gallinari@lip6.fr

Received 3 June 2019
Accepted for publication 6 June 2019
Published 20 December 2019

CrossMark

Online at stacks.iop.org/JSTAT/2019/124009
https://doi.org/10.1088/1742-5468/ab3195

Abstract. We consider the use of deep learning methods for modeling complex phenomena like those occurring in natural physical processes. With the large amount of data gathered on these phenomena the data intensive paradigm could begin to challenge more traditional approaches elaborated over the years in fields like maths or physics. However, despite considerable successes in a variety of application domains, the machine learning field is not yet ready to handle the level of complexity required by such problems. Using an example application, namely sea surface temperature prediction, we show how general background knowledge gained from the physics could be used as a guideline for designing efficient deep learning models. In order to motivate the approach and to assess its generality we demonstrate a formal link between the solution of a class of differential equations underlying a large family of physical phenomena and the proposed model. Experiments and comparison with series of baselines including a state of the art numerical approach is then provided.

Keywords: machine learning

Presented by
Jesús Peña-Izquierdo

Journal of Statistical Mechanics: Theory and Experiment
An IOP and SISSA journal

Deep learning for physical processes: incorporating prior scientific knowledge*

Emmanuel de Bézenac[1,3], Arthur Pajot[1,3] and Patrick Gallinari[2]

[1] Sorbonne Universites, UMR 7606, LIP6, F-75005 Paris, France
[2] Sorbonne Université, UMR 7606, LIP6, Paris and Criteo AI Lab, Paris, France
E-mail: emmanuel.de-bezenac@lip6.fr, arthur.pajot@lip6.fr and patrick.gallinari@lip6.fr

Received 3 June 2019
Accepted for publication 6 June 2019
Published 20 December 2019

CrossMark

Online at stacks.iop.org/JSTAT/2019/124009
https://doi.org/10.1088/1742-5468/ab3195

Abstract. We consider the use of deep learning methods for modeling complex phenomena like those occurring in natural physical processes. With the large

handle the level of complexity required by such problems. Using an example application, namely sea surface temperature prediction, we show how general background knowledge gained from the physics could be used as a guideline for designing efficient deep learning models. In order to motivate the approach and

and the proposed model. Experiments and comparison with series of baselines including a state of the art numerical approach is then provided.

Keywords: machine learning

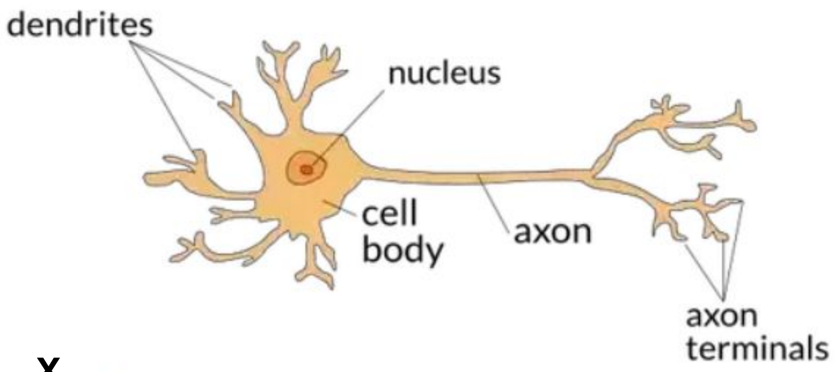
Overview

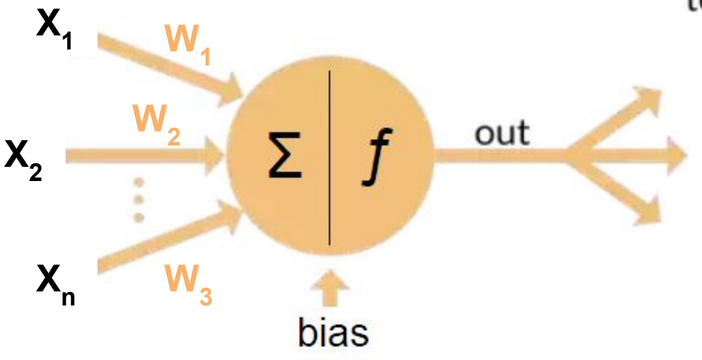
- Neural Network basic concepts
- The problem
- The model
- The results
- The conclusion

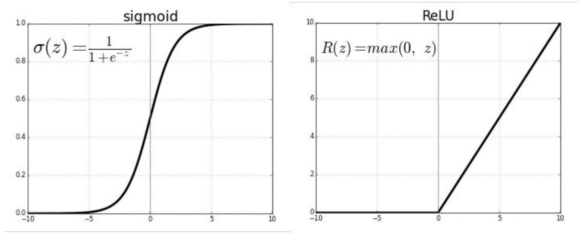
Multilayer Perceptron a.k.a **Neural Network**

Rosenblatt (**1958**)

A **brain neuron** vs an **artificial neuron** (perceptron)







Activation function

Bias

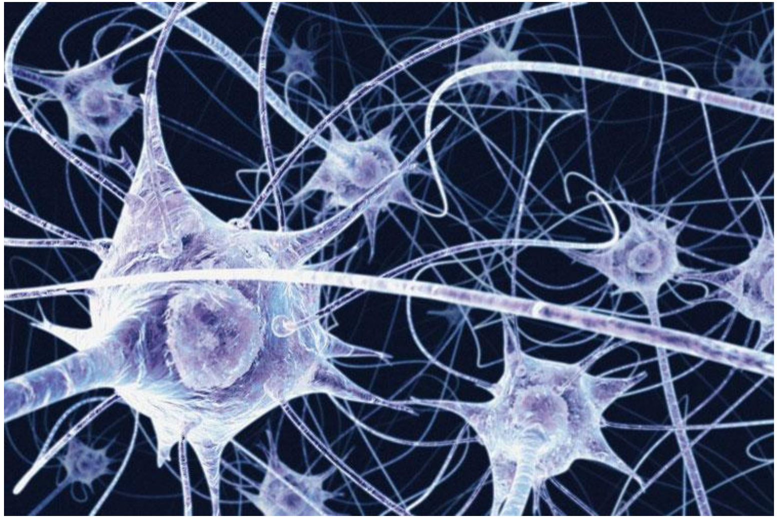
Weights

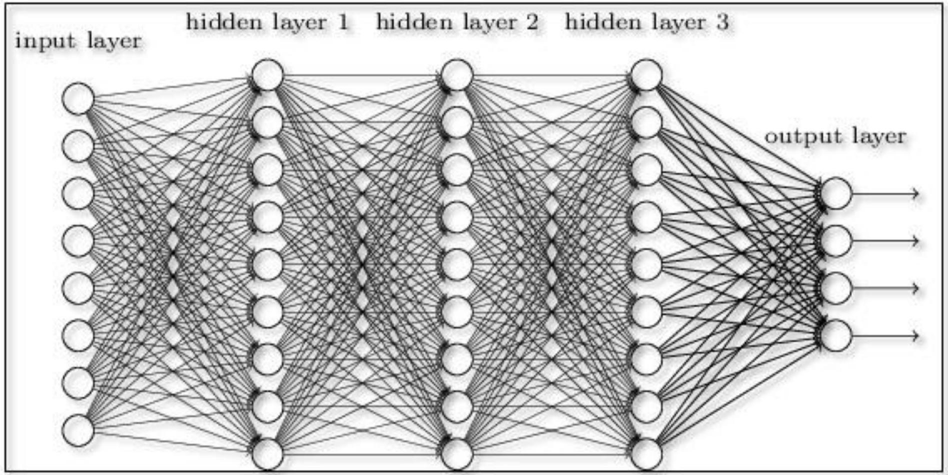
$$f\left(b + \sum_{i=1}^{n} x_i w_i\right)$$

Multilayer Perceptron a.k.a **Neural Network**

Rosenblatt (**1958**)

A **Brain** vs an **Artificial Neural Network** (Multilayer Perceptron)

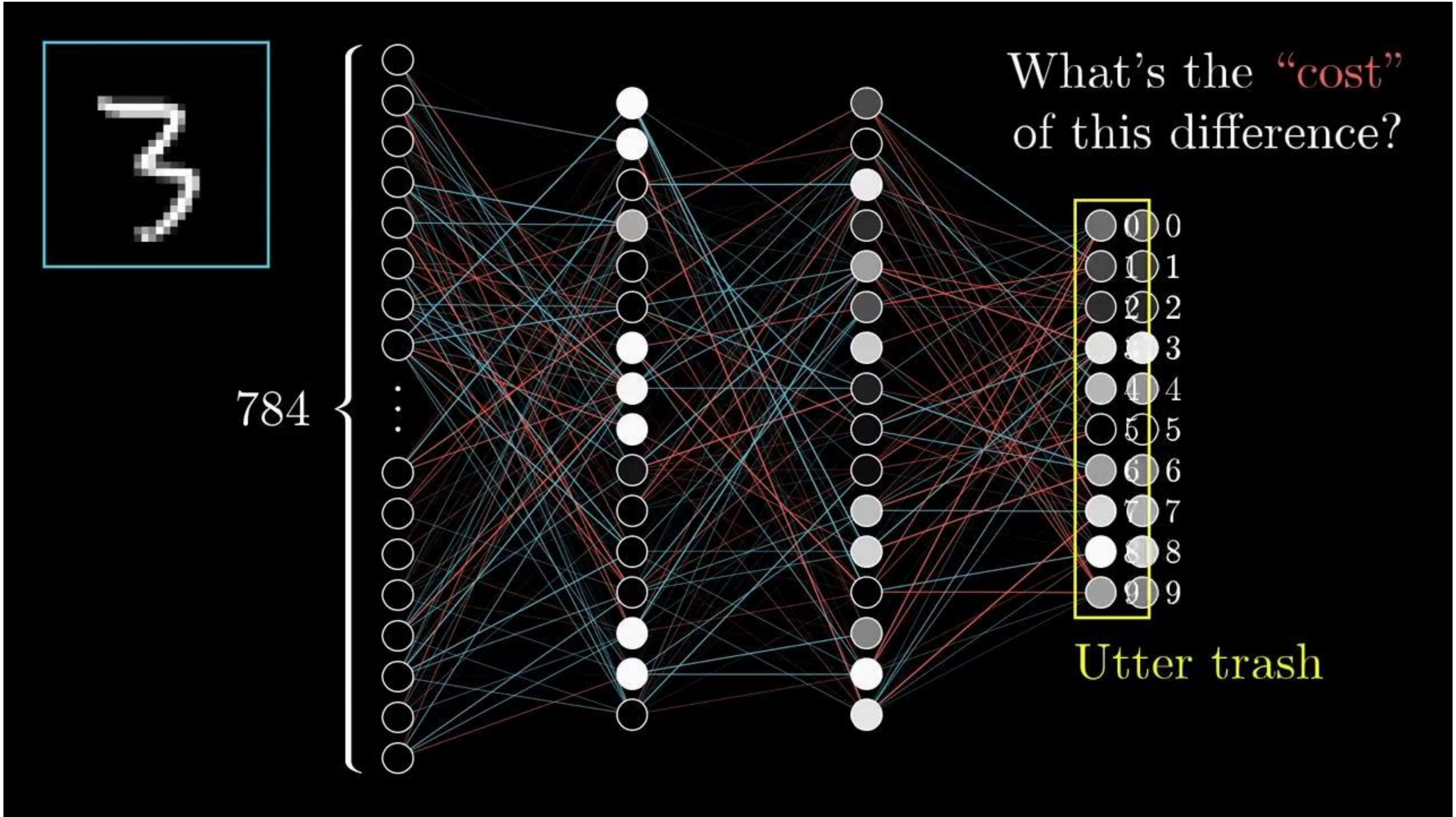




~80 Billion neurons

Learning through backpropagation

Video by
3Blue1Brown

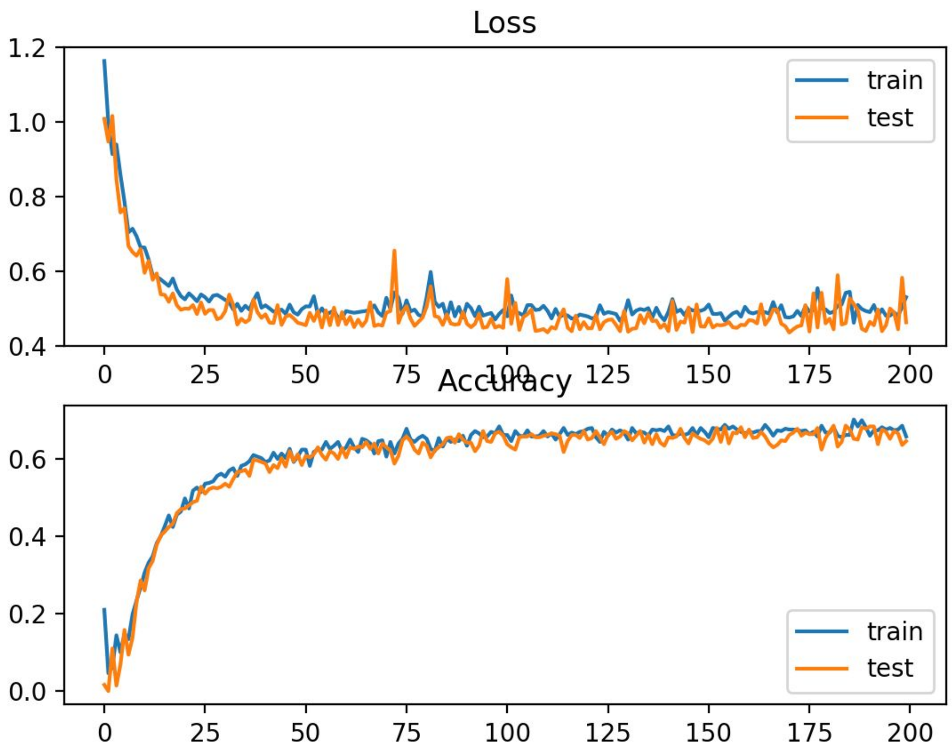


Training is just minimizing a loss function

your prediction

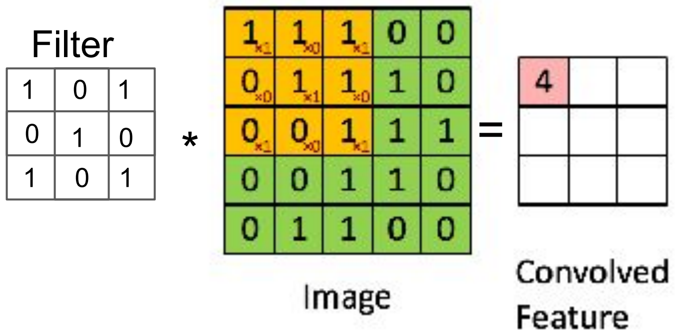
$$Loss(y, \hat{y}) = \sum_{i=1}^{n} (y - \hat{y})^2$$

ground truth

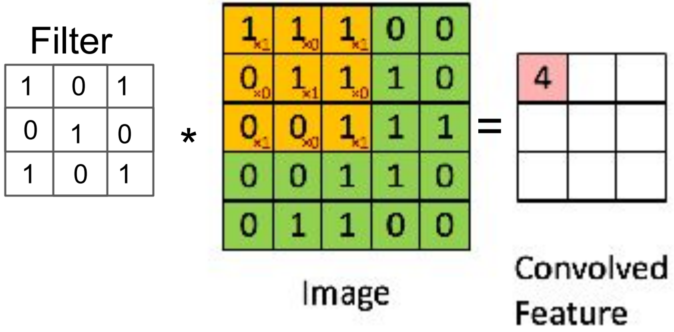


How **images** are processed by Neural Networks?

Convolutional layers

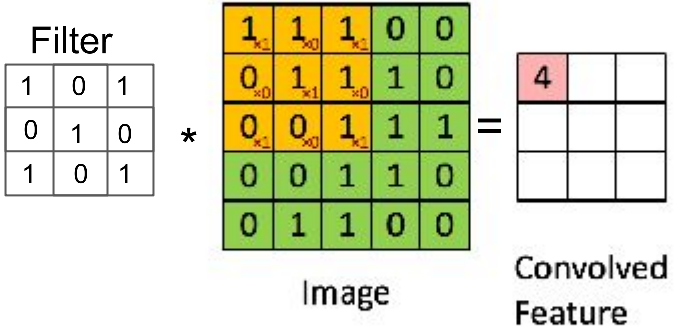


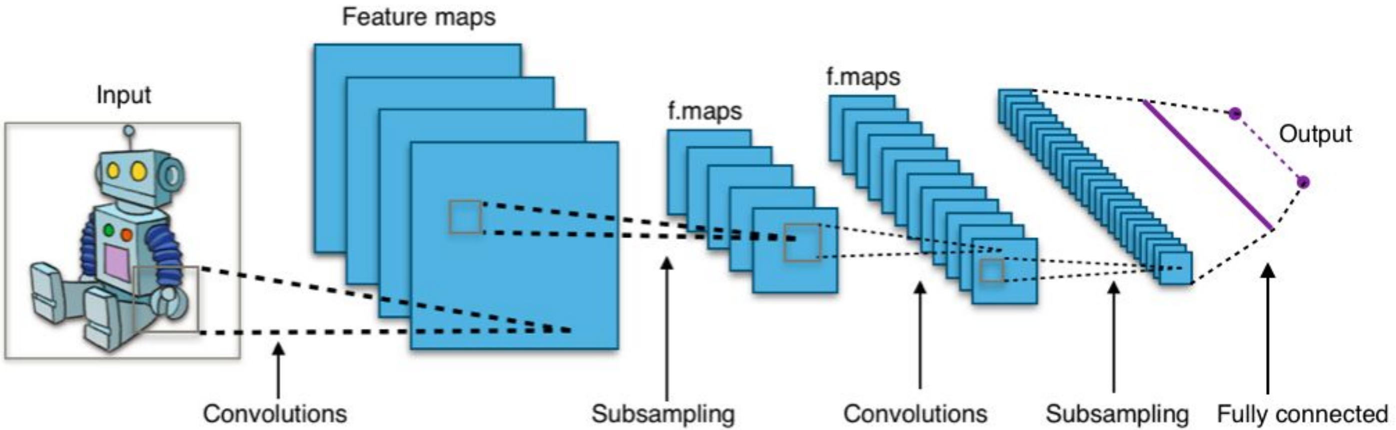
Convolutional layers



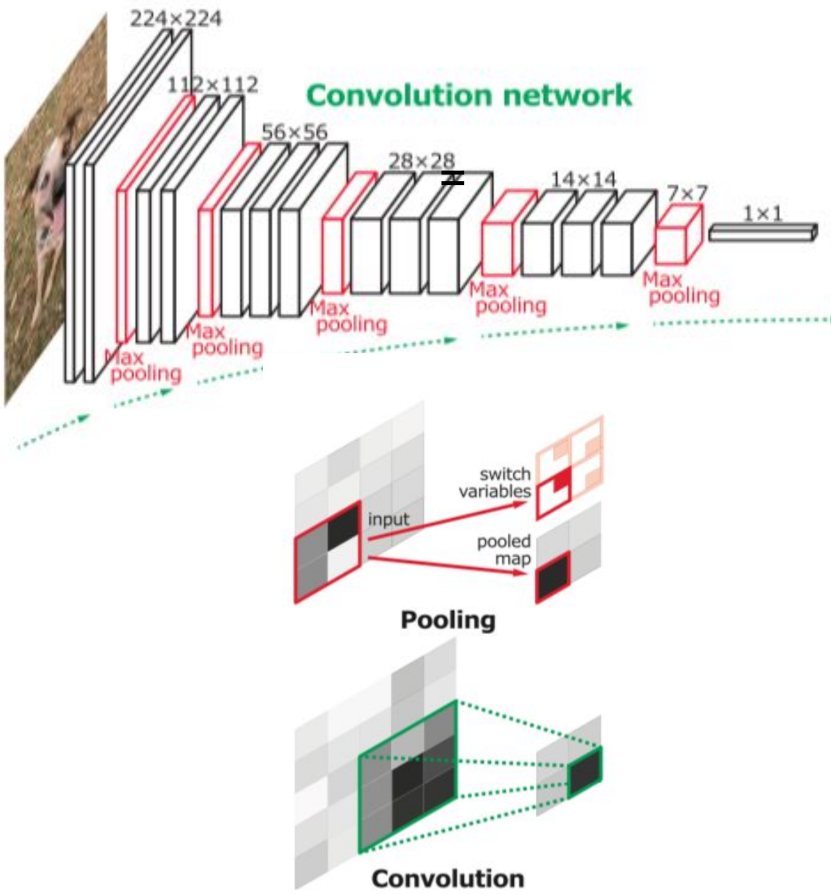
Operation	Kernel ω	Image result g(x,y)
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	

Convolutional layers



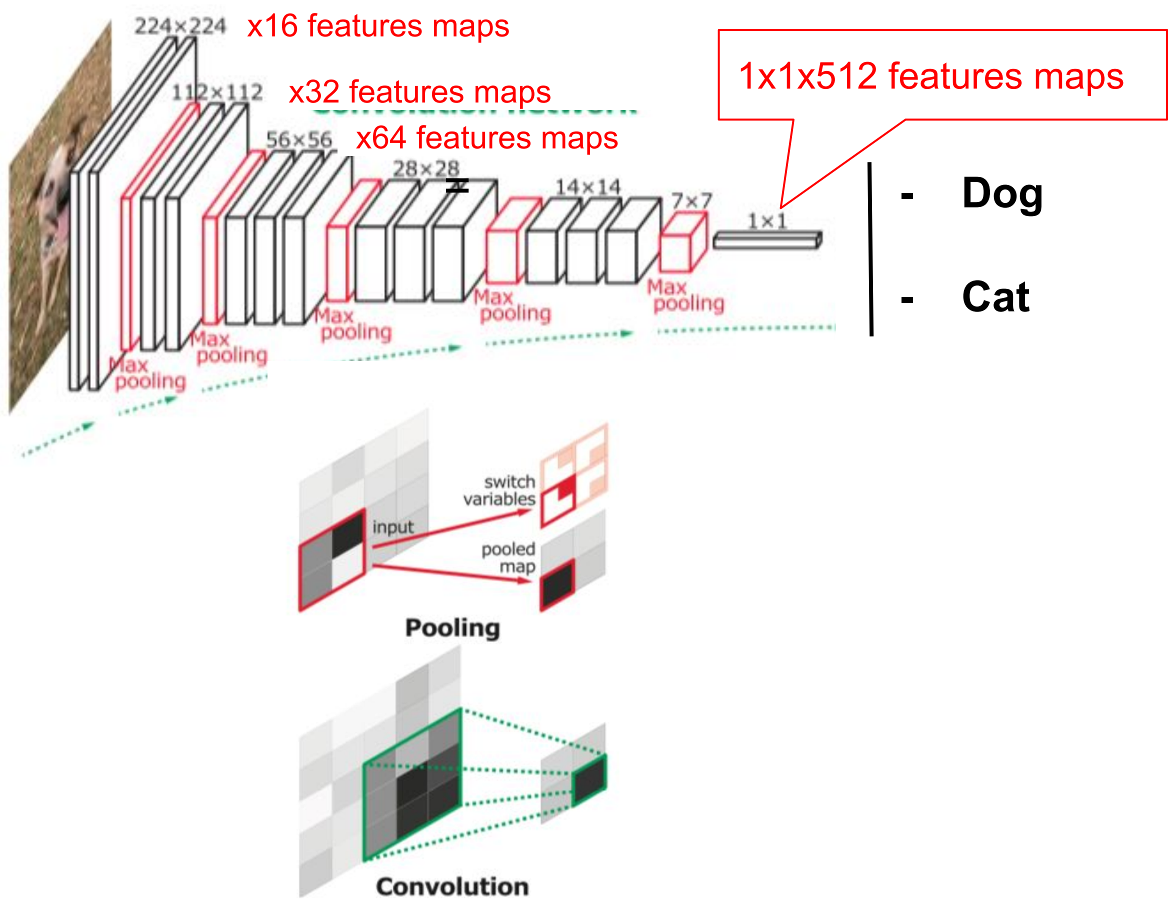


Convolutional layers (ENCODER)



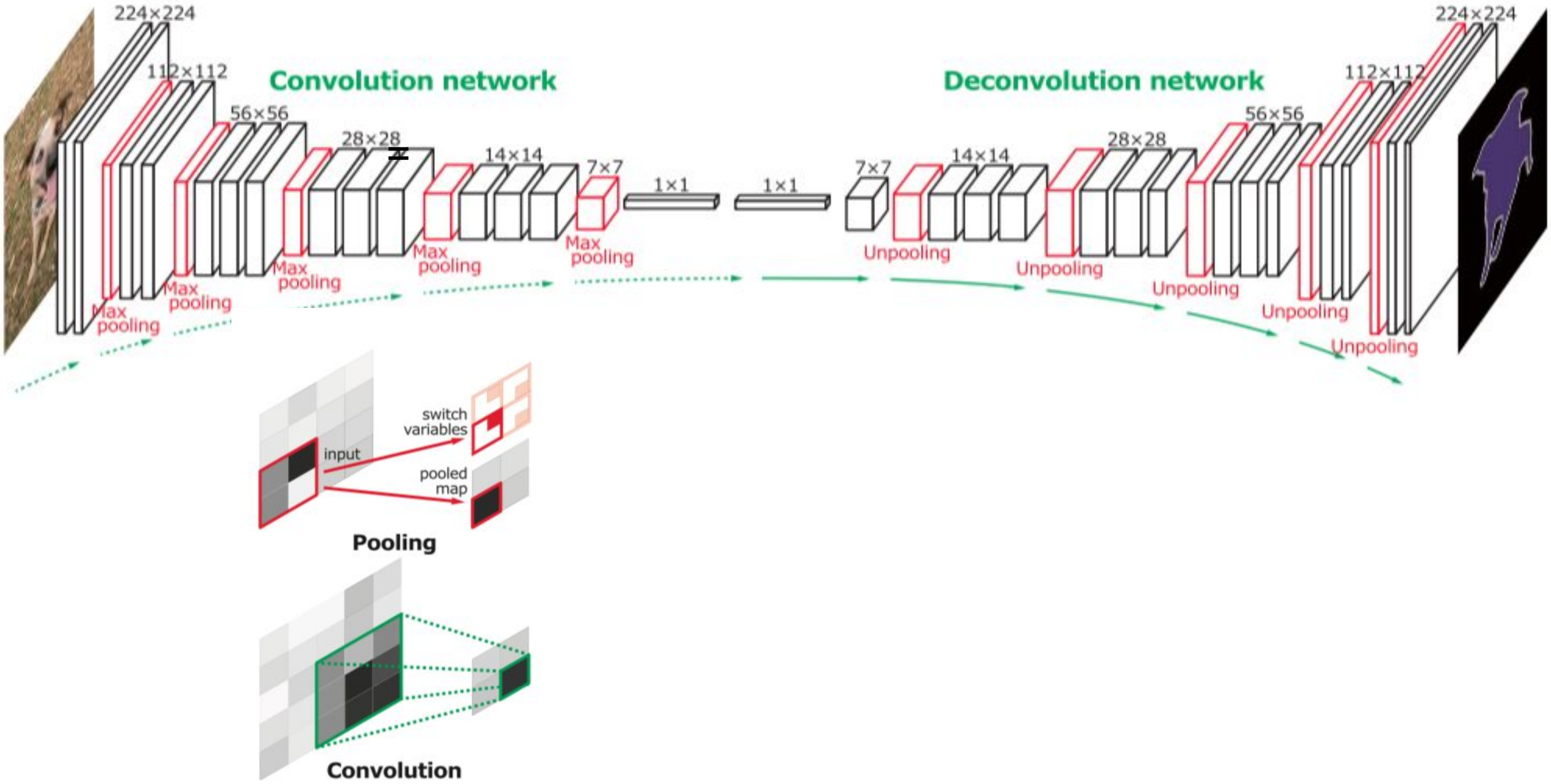
- Dog
- Cat

Convolutional layers (ENCODER)

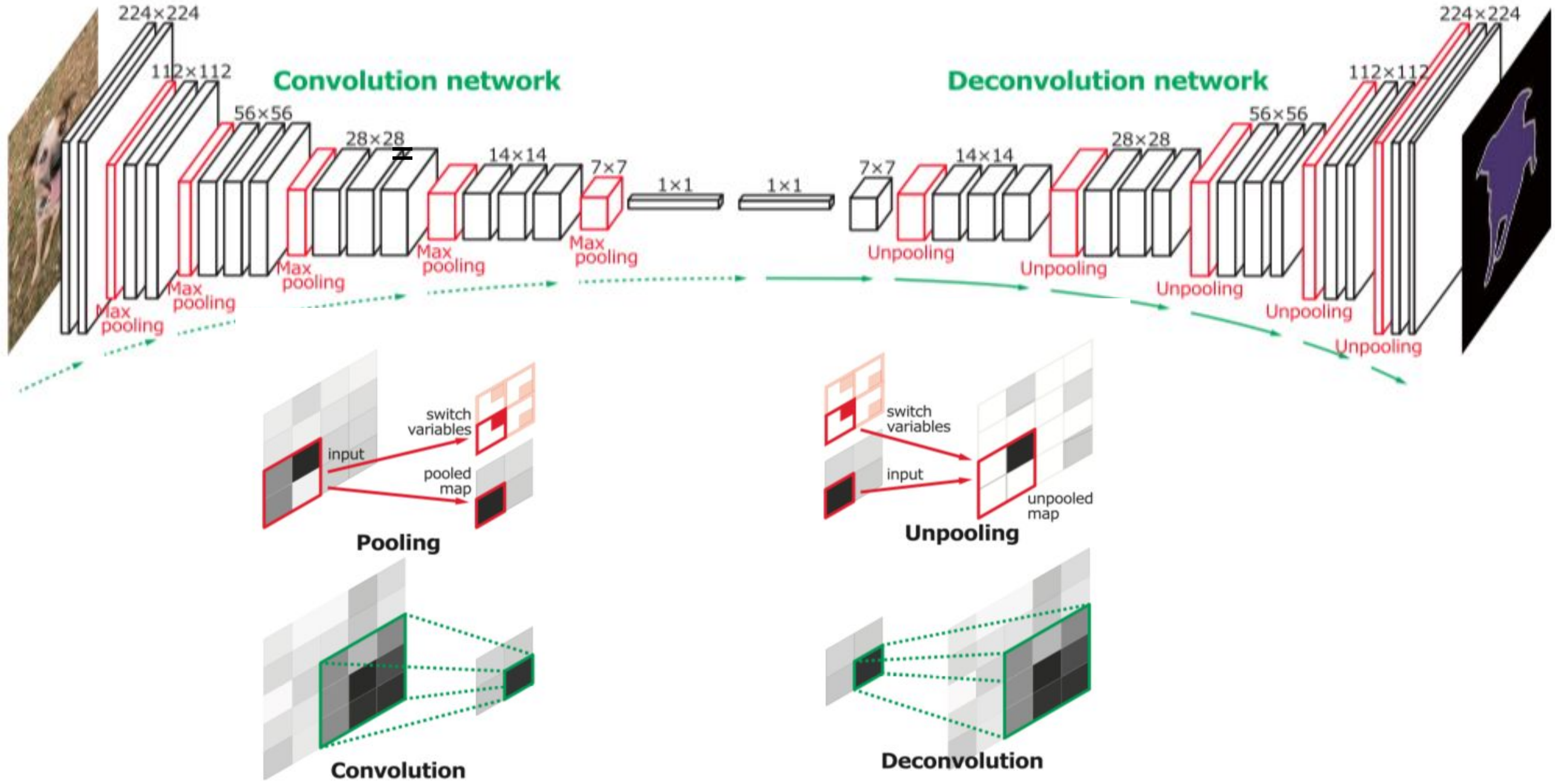


Convolutional layers (ENCODER)

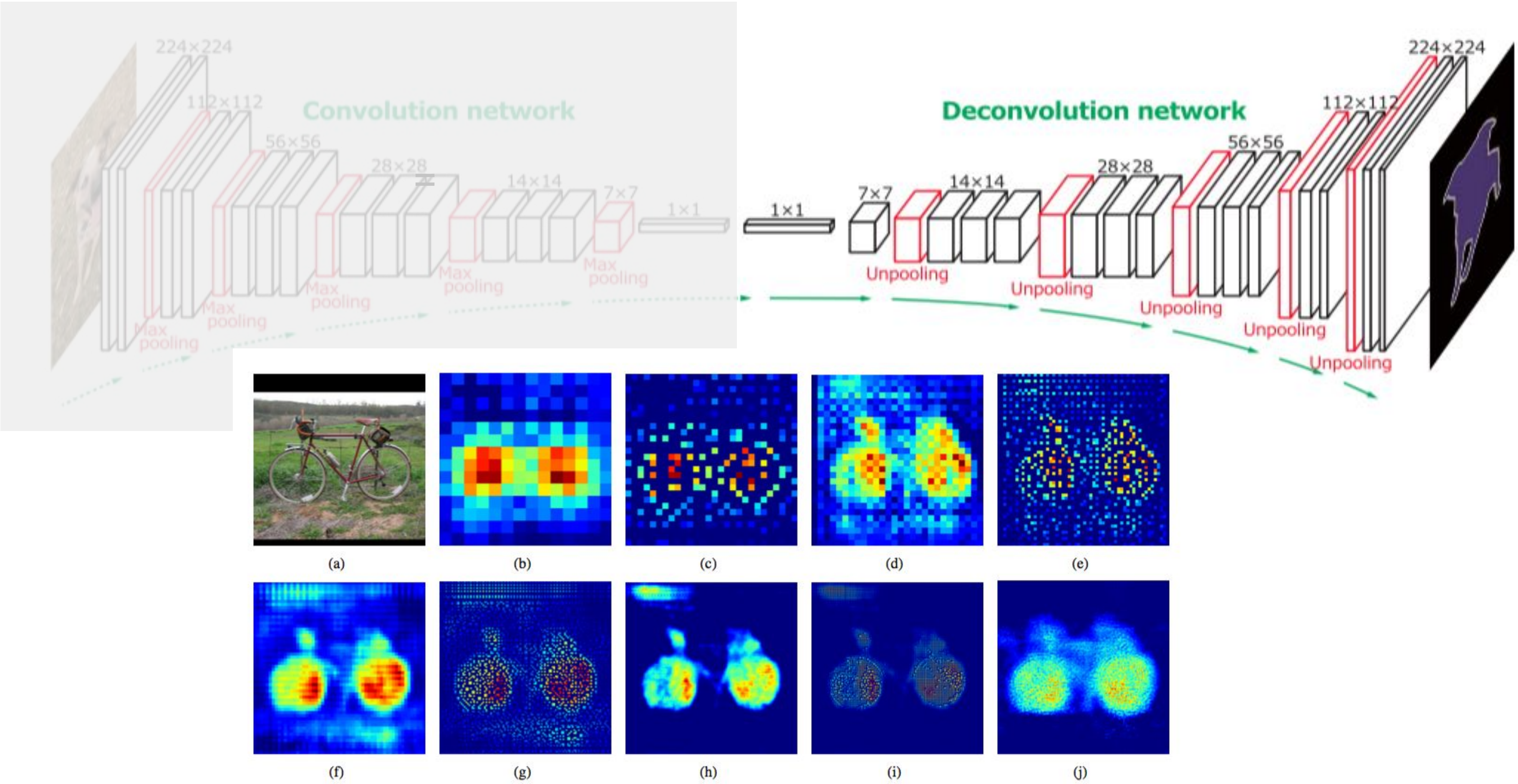
Deconvolutional layers (DECODER)



Convolutional layers (ENCODER) **Deconvolutional layers (DECODER)**

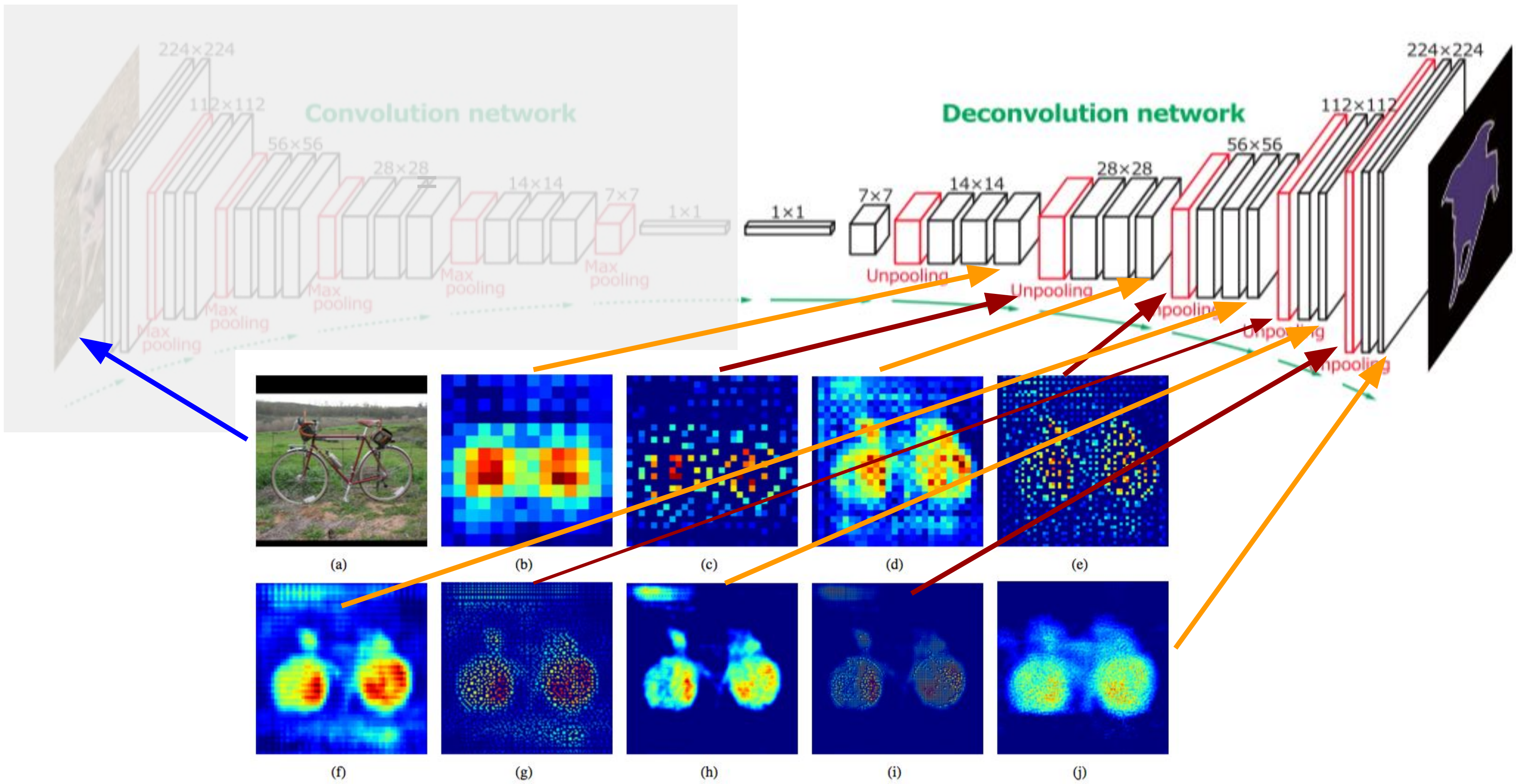


Convolutional layers (ENCODER) | Deconvolutional layers (DECODER)

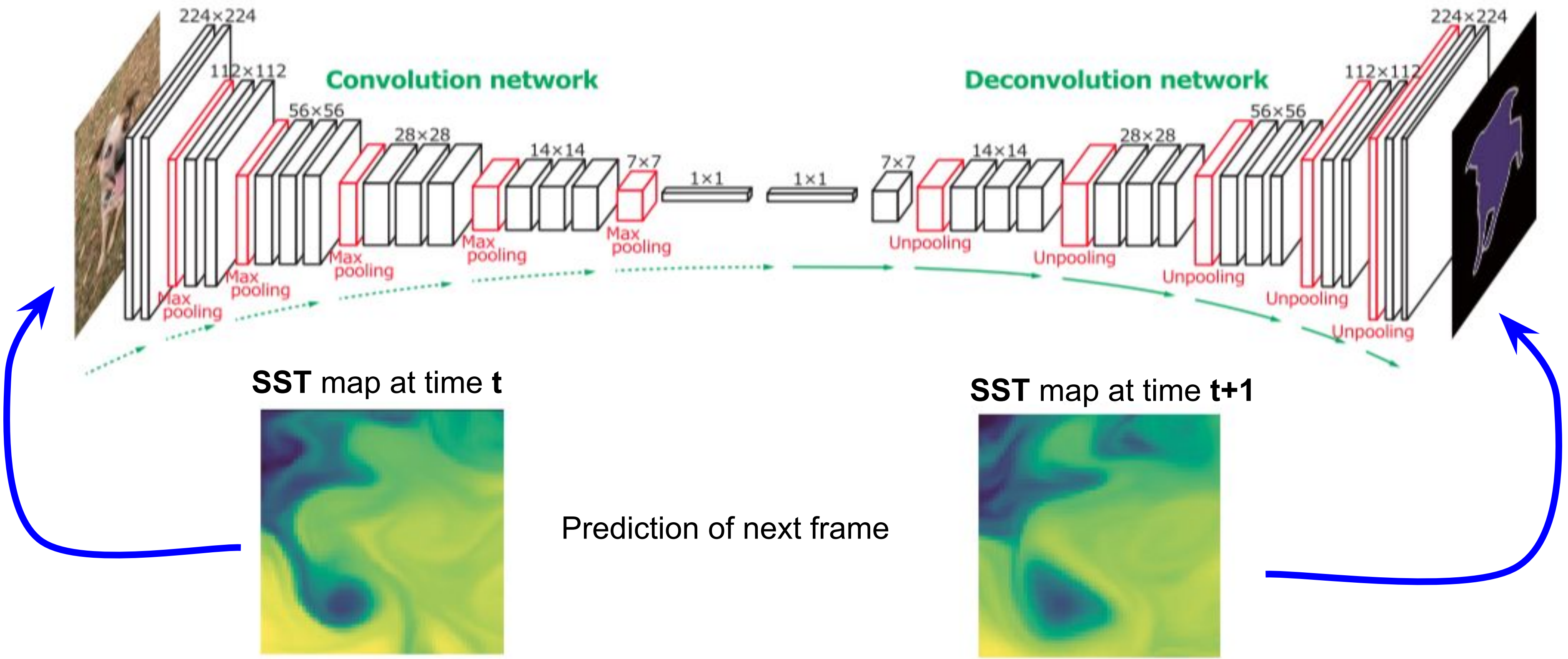


Convolutional layers (ENCODER)

Deconvolutional layers (DECODER)



Convolutional layers (ENCODER) **Deconvolutional layers (DECODER)**

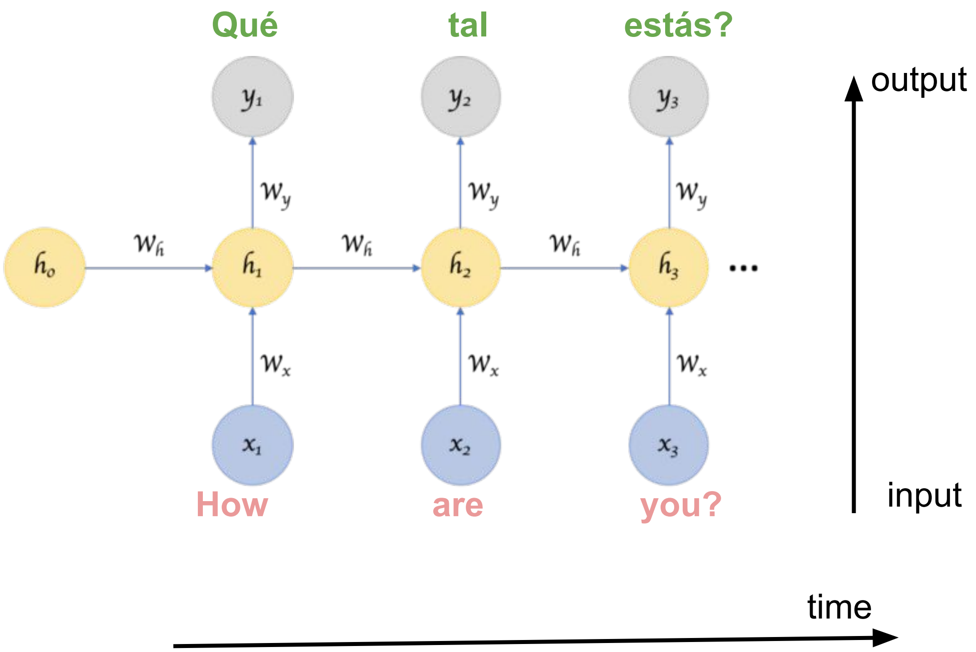


Adding the temporal dimension....

Adding the temporal dimension....

Recurrent Neural Network

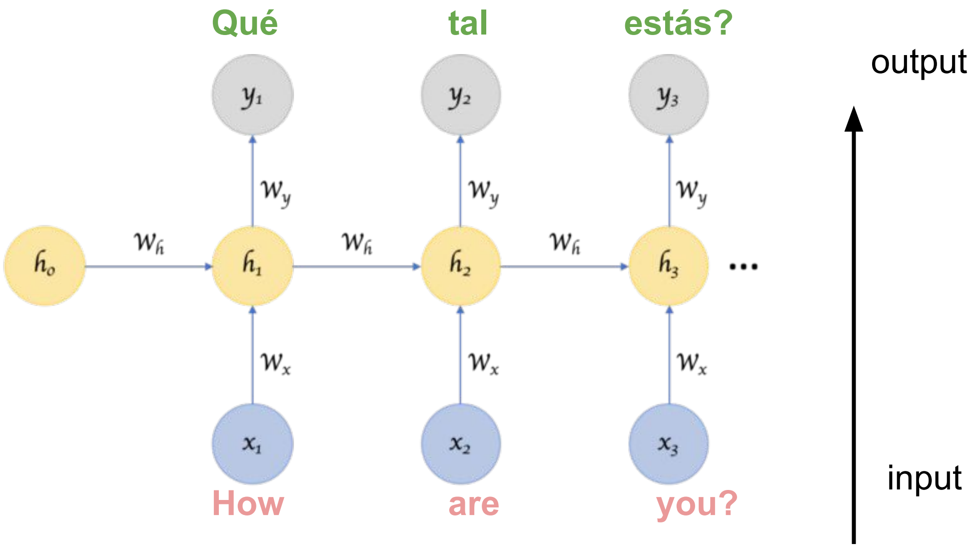
- Process **time series** inputs
- There is a **state input (hi)** related to previous inputs



Adding the temporal dimension....

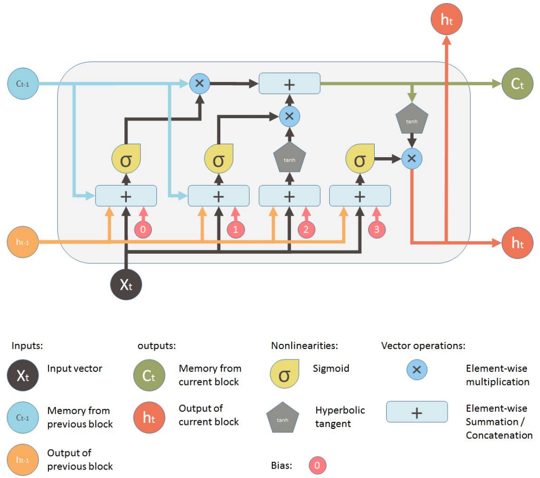
<u>Recurrent Neural Network</u>

- Process **time series** inputs
- There is a **state input (hi)** related to previous inputs



<u>Long-Short-Term-Memory Neural Network</u>

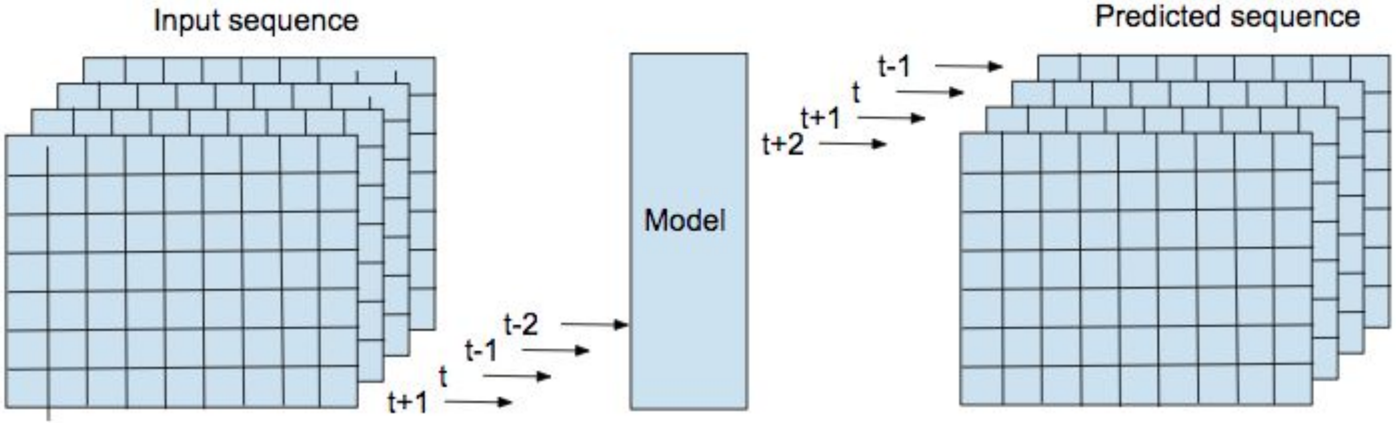
- Process time series as inputs
- Allows **storing** and **removing** context from **very far in time (memory)**



Adding the temporal dimension....

<u>Long-Short-Term-Memory **CONVOLUTIONAL** Neural Networks **(Conv-LSTM)**</u>

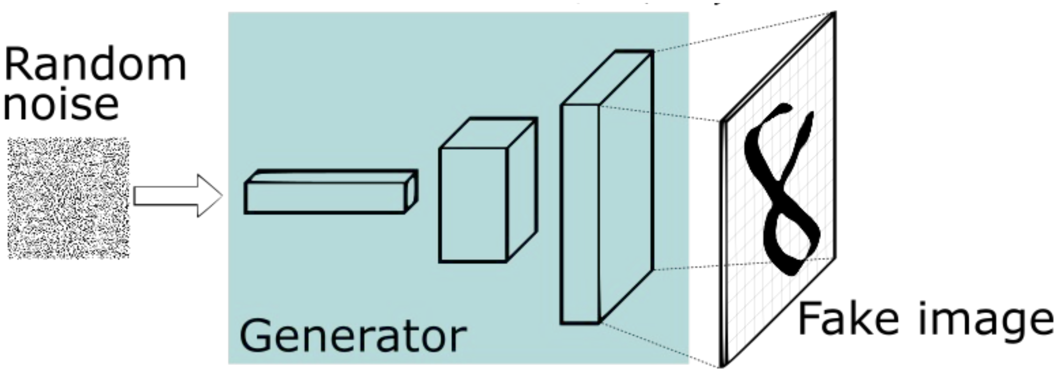
- Process **time series of images** as inputs
- Allow **storing** and **removing** context from very far in time



Generative models

Generative models.

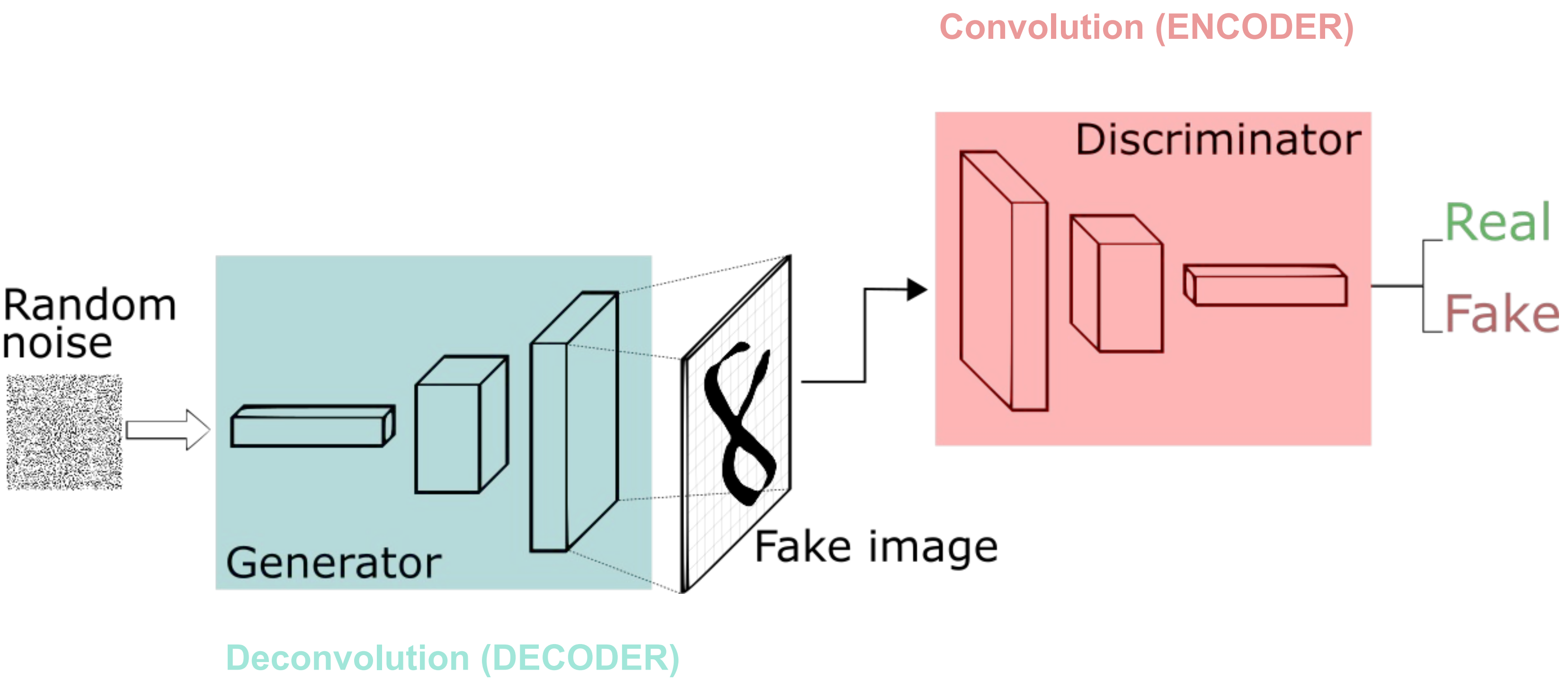
Generative Adversarial Networks (GANs)



Deconvolution (DECODER)

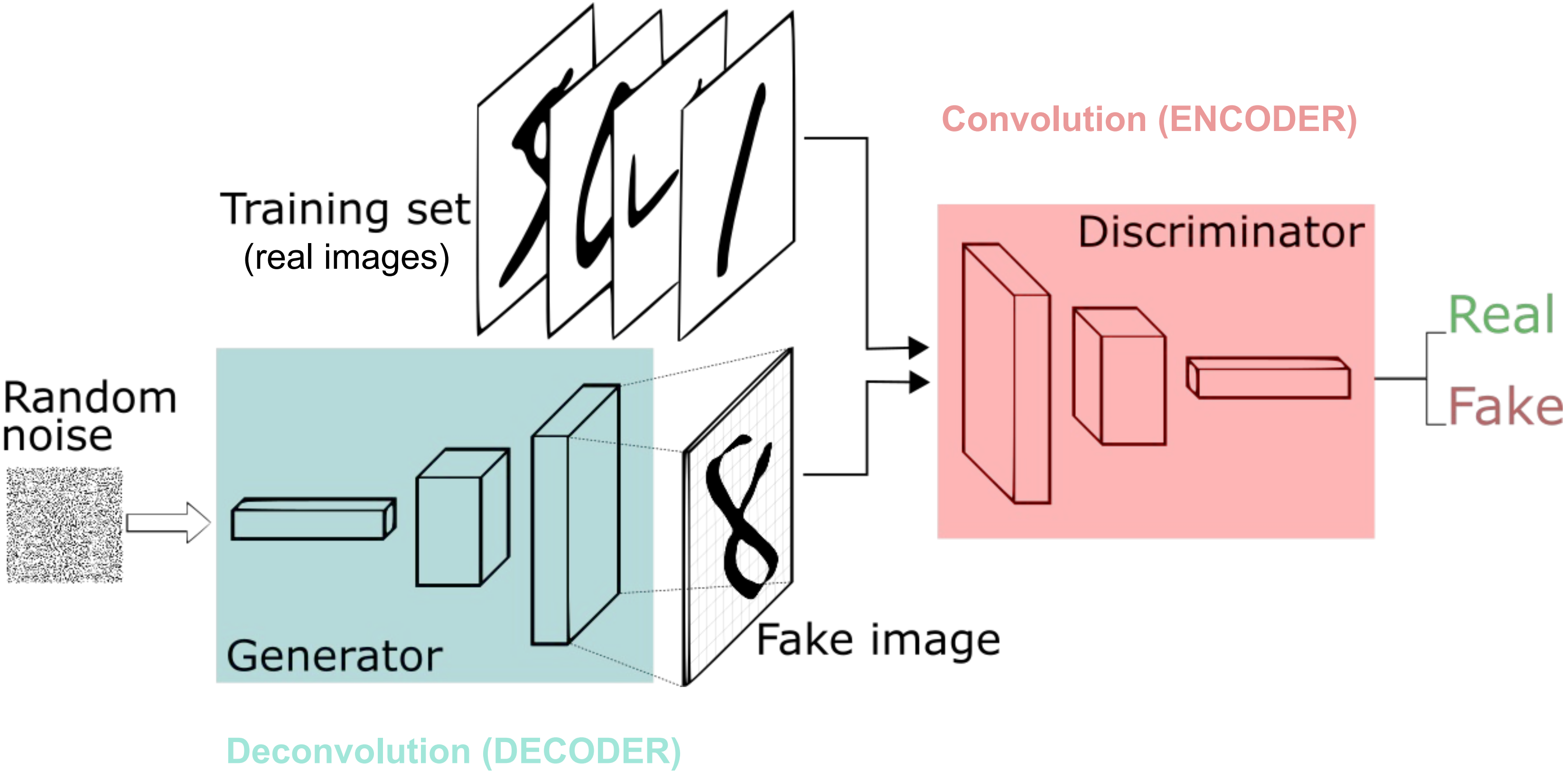
Generative models.

Generative Adversarial Networks (GANs)



Generative models.

Generative Adversarial Networks (GANs)



Summary of models

- Convolutional-Deconvolutional Neural Network (CDNN)
    - No temporal dimension explicitly included
- LSTM Convolutional-Deconvolutional Neural Network (Conv-LSTM)
    - Temporal dimension explicitly included
- Generative Adversarial Network (GAN)
    - No temporal dimension explicitly included

Summary of models

- Convolutional-Deconvolutional Neural Network (CDNN)
  - No temporal dimension explicitly included
- LSTM Convolutional-Deconvolutional Neural Network (Conv-LSTM)
  - Temporal dimension explicitly included
- Generative Adversarial Network (GAN)
  - No temporal dimension explicitly included

NON OF THEM HAS ANY IDEA OF WHAT THE LAWS OF PHYSICS ARE!!!

Journal of Statistical Mechanics: Theory and Experiment
An IOP and SISSA journal

Deep learning for physical processes: incorporating prior scientific knowledge*

Emmanuel de Bézenac[1,3], Arthur Pajot[1,3] and Patrick Gallinari[2]

[1] Sorbonne Universites, UMR 7606, LIP6, F-75005 Paris, France
[2] Sorbonne Université, UMR 7606, LIP6, Paris and Criteo AI Lab, Paris, France
E-mail: emmanuel.de-bezenac@lip6.fr, arthur.pajot@lip6.fr
and patrick.gallinari@lip6.fr

Received 3 June 2019
Accepted for publication 6 June 2019
Published 20 December 2019

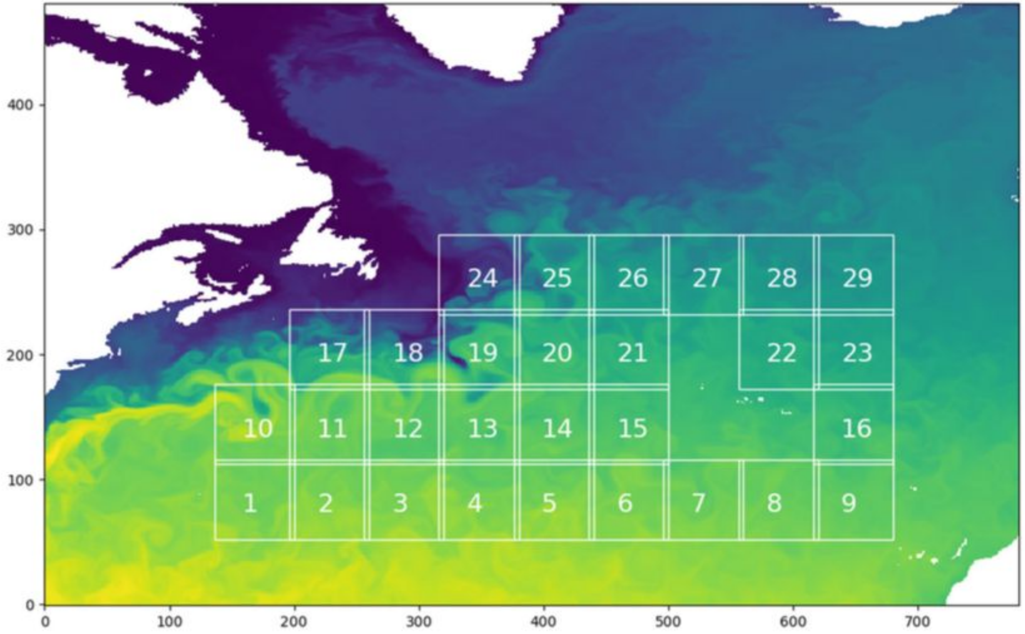
Online at stacks.iop.org/JSTAT/2019/124009
https://doi.org/10.1088/1742-5468/ab3195

Abstract. We consider the use of deep learning methods for modeling complex phenomena like those occurring in natural physical processes. With the large amount of data gathered on these phenomena the data intensive paradigm could begin to challenge more traditional approaches elaborated over the years in fields like maths or physics. However, despite considerable successes in a variety of application domains, the machine learning field is not yet ready to handle the level of complexity required by such problems. Using an example application, namely sea surface temperature prediction, we show how general background knowledge gained from the physics could be used as a guideline for designing efficient deep learning models. In order to motivate the approach and to assess its generality we demonstrate a formal link between the solution of a class of differential equations underlying a large family of physical phenomena and the proposed model. Experiments and comparison with series of baselines including a state of the art numerical approach is then provided.

Keywords: machine learning

DATA

- **Predict SST maps**
- Data comes from NEMO model with data assimilation (~5-10km resolution)
- Daily images, subregions 64x64pixels, period 2006 to 2017
- Training/validation set: 2006 to 2015 (94743 samples, 20% val.)
- Test set: 2016 to 2017
- Seasonality is removed normalizing by day-of-year mean divided by std



I: Value of image at (x,t)
w: velocity field (2-dimensional)
D: Diffusivity coefficient

MODEL - Imposing physical constraints

- The SST evolution is primary driven by the **Advection-Diffusion** equation: (no sources or sinks considered ¿?)

$$\frac{\partial I}{\partial t} + (w \cdot \nabla) I = D \nabla^2 I.$$

I: Value of image at (x,t)
***w*:** velocity field (2-dimensional)
***D*:** Diffusivity coefficient

MODEL - Imposing physical constraints

- The SST evolution is primary driven by the **Advection-Diffusion** equation:
(no sources or sinks considered ¿?)

$$\frac{\partial I}{\partial t} + (w \cdot \nabla) I = D \nabla^2 I.$$

- A discretized solution to the A-D eq. is given by:

$$\hat{I}_{t+1}(x) = \sum_{y \in \Omega} \frac{1}{4\pi D \Delta t} \mathrm{e}^{-\frac{1}{4D\Delta t}\|x - \hat{w} - y\|^2} * \; I_t(y)$$

Deep learning for physical processes: incorporating prior scientific knowledge
Emmanuel de Bézenac, Arthur Pajot and Patrick Gallinari

I: Value of image at (x,t)
w: velocity field (2-dimensional)
D: Diffusivity coefficient

MODEL - Imposing physical constraints

- The SST evolution is primary driven by the **Advection-Diffusion** equation: (no sources or sinks considered ¿?)

$$\frac{\partial I}{\partial t} + (w \cdot \nabla) I = D \nabla^2 I.$$

- A discretized solution to the A-D eq. is given by:

$$\hat{I}_{t+1}(x) = \sum_{y \in \Omega} \frac{1}{4\pi D \Delta t} \mathrm{e}^{-\frac{1}{4D\Delta t}\|x - \hat{w} - y\|^2} * I_t(y)$$

$\hat{I}_{t+1}(x)$: Value of image pixel at (x = x, t = t+1)

$I_t(y)$: Value of image pixel at (x = y, t = t)

Deep learning for physical processes: incorporating prior scientific knowledge
Emmanuel de Bézenac, Arthur Pajot and Patrick Gallinari

- I: Value of image at (x,t)
- w: velocity field (2-dimensional)
- D: Diffusivity coefficient

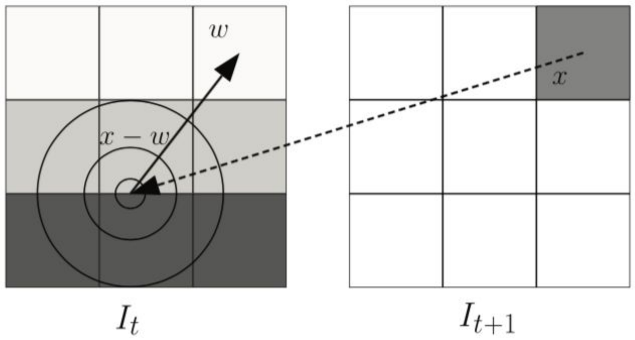
MODEL - Imposing physical constraints

- The SST evolution is primary driven by the **Advection-Diffusion** equation: (no sources or sinks considered ¿?)

$$\frac{\partial I}{\partial t} + (w \cdot \nabla) I = D \nabla^2 I.$$

- A discretized solution to the A-D eq. is given by:

$$\hat{I}_{t+1}(x) = \sum_{y \in \Omega} \frac{1}{4\pi D \Delta t} e^{-\frac{1}{4 D \Delta t} \|x - \hat{w} - y\|^2} * I_t(y)$$



Value of image pixel at (x = x, t = t+1)

Gaussian kernel with radial dependency

Value of image pixel at (x = y, t = t)

It is a weighted average of the temperatures at time t and location x, weights are larger when the pixel's positions is closer to initial position (x − w)

Deep learning for physical processes: incorporating prior scientific knowledge
Emmanuel de Bézenac, Arthur Pajot and Patrick Gallinari

I: Value of image at (x,t)
w: velocity field (2-dimensional)
D: Diffusivity coefficient

MODEL - Imposing physical constraints

- From the solution equation the **unknowns** to compute ***I(x,t+1)***:

$$\hat{I}_{t+1}(x) = \sum_{y\in\Omega} \frac{1}{4\pi D\Delta t} e^{-\frac{1}{4D\Delta t}\|x-\hat{w}-y\|^2} * I_t(y)$$

→

w: velocity field (2-dimensional)

D: Diffusivity coefficient

I: Value of image at (x,t)
w: velocity field (2-dimensional)
D: Diffusivity coefficient

MODEL - Imposing physical constraints

- From the solution equation the **unknowns** to compute ***I(x,t+1)***:

$$\hat{I}_{t+1}(x) = \sum_{y \in \Omega} \frac{1}{4\pi D\Delta t} \mathrm{e}^{-\frac{1}{4D\Delta t}\|x-\hat{w}-y\|^2} * I_t(y)$$

→

w: velocity field (2-dimensional)

D: Diffusivity coefficient

↓

Extract ***w*** and ***D*** from data

Deep learning for physical processes: incorporating prior scientific knowledge

Emmanuel de Bézenac, Arthur Pajot and Patrick Gallinari

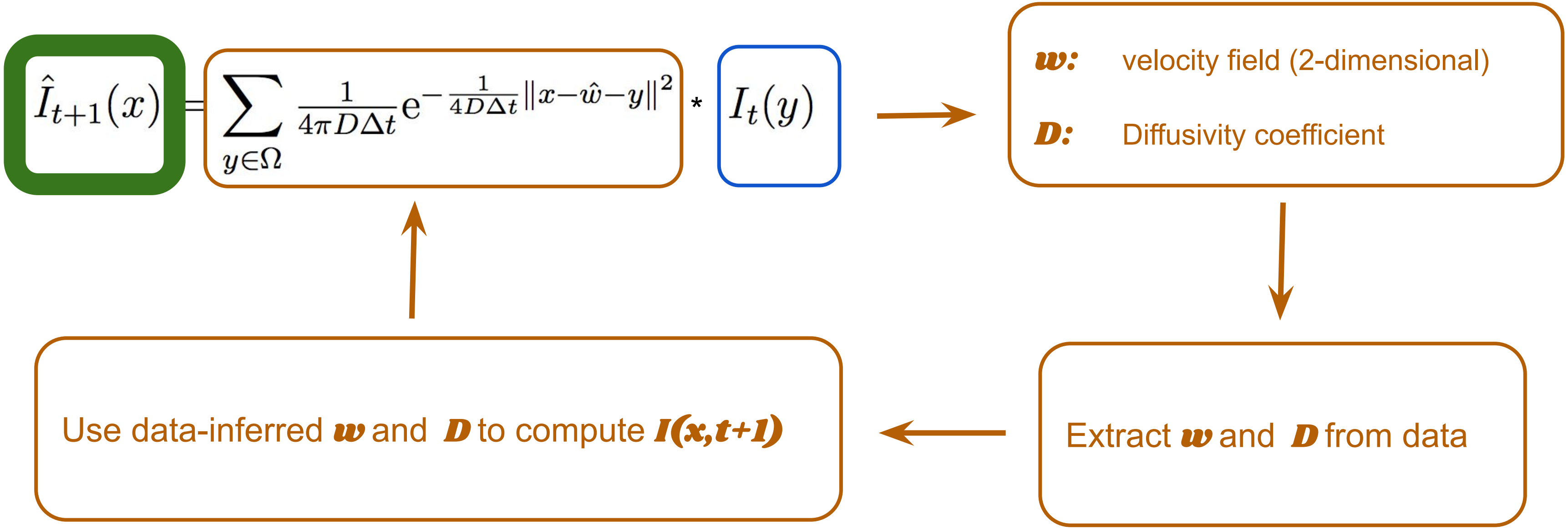
I: Value of image at (x,t)

w: velocity field (2-dimensional)

D: Diffusivity coefficient

MODEL - Imposing physical constraints

- From the solution equation the **unknowns** to compute ***I(x,t+1)***:



Deep learning for physical processes: incorporating prior scientific knowledge
Emmanuel de Bézenac, Arthur Pajot and Patrick Gallinari

I: Value of image at (x,t)
w: velocity field (2-dimensional)
D: Diffusivity coefficient

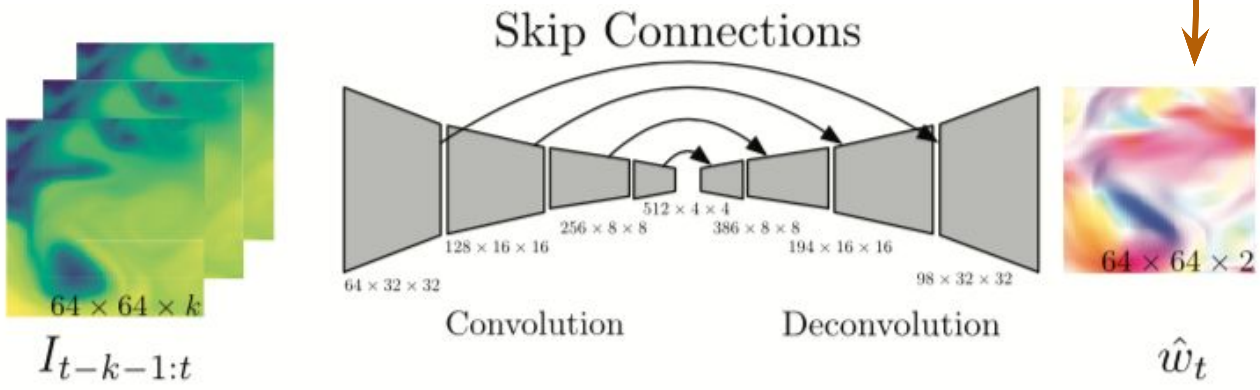
MODEL - Imposing physical constraints

- From the solution equation the **unknowns** to compute ***I(x,t+1)***:

$$\hat{I}_{t+1}(x) = \sum_{y\in\Omega} \frac{1}{4\pi D\Delta t} e^{-\frac{1}{4D\Delta t}\|x-\hat{w}-y\|^2} * I_t(y)$$

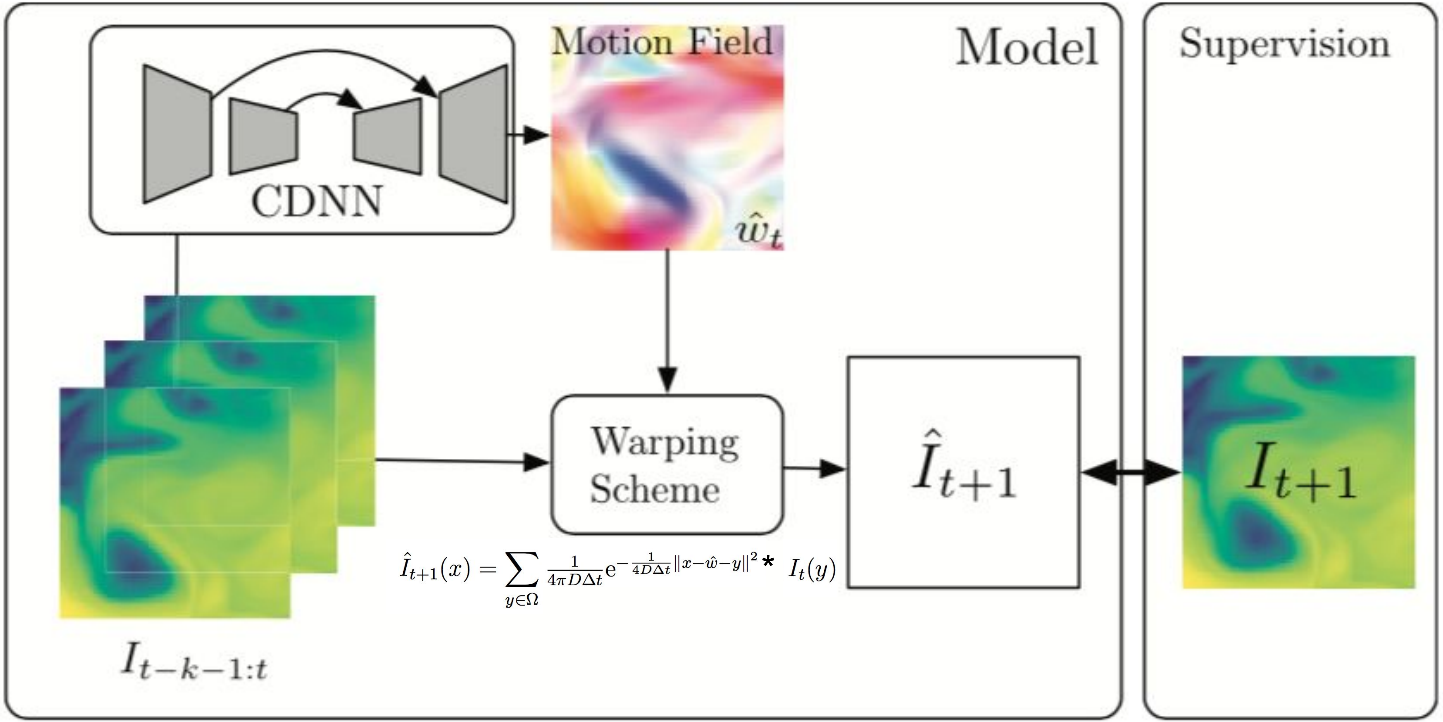
w: velocity field (2-dimensional)

D: Diffusivity coefficient

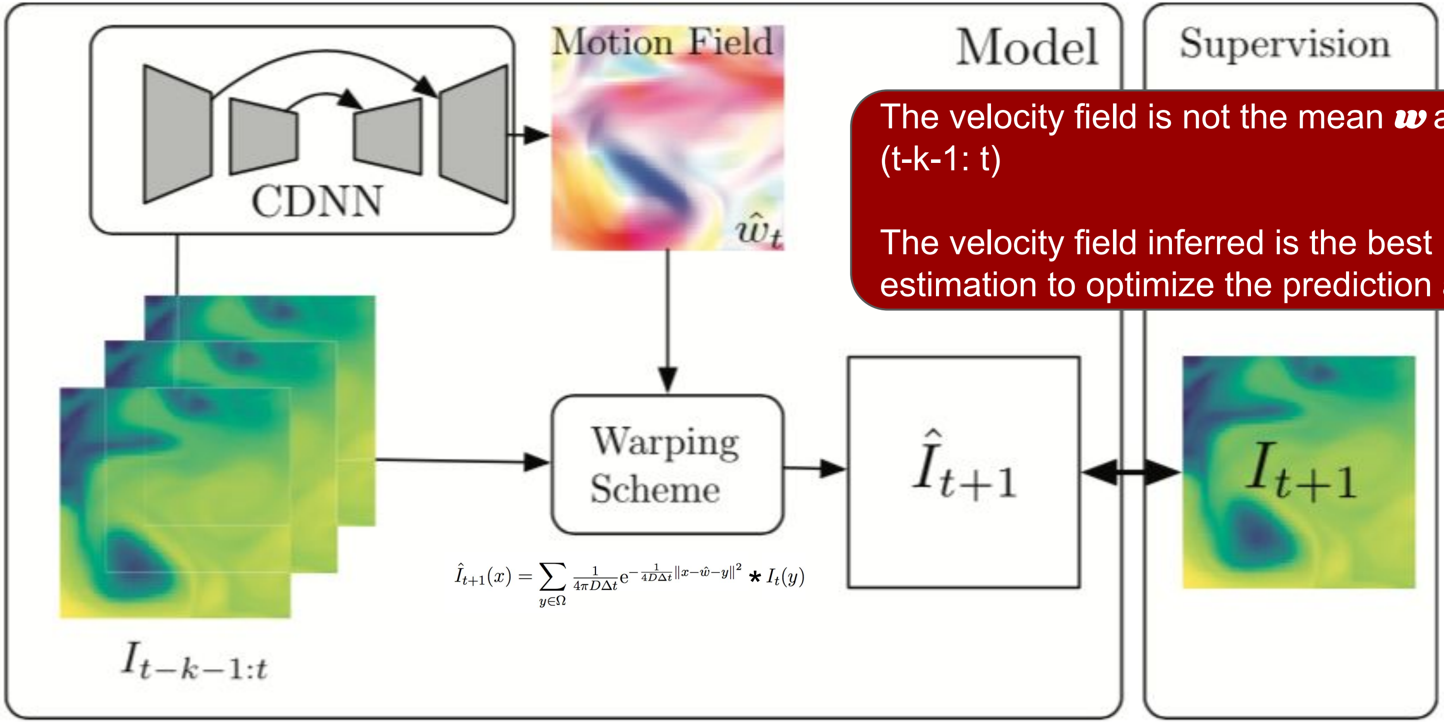


w **is 2-dimensional (u, v)**

MODEL - Learning from data constrained by physics



MODEL - Learning from data constrained by physics



The velocity field is not the mean $\boldsymbol{w}$ at time (t-k-1: t)

The velocity field inferred is the best estimation to optimize the prediction at t+1

MODEL - Learning from data constrained by physics

Loss function

$$Loss(y,\hat{y}) = \sum_{i=1}^{n} (y - \hat{y})^2$$

Additional physical constraints - Regularization:

$$L_t = \sum_{x\in\Omega} \rho(\hat{I}_{t+1}(x) - I_{t+1}(x)) + \lambda_{\text{div}}(\nabla . w_t(x))^2 + \lambda_{\text{magn}} \|w_t(x)\|^2 + \lambda_{\text{grad}} \|\nabla w_t(x)\|^2$$

Charbonnier penalty function

$$\rho(x) = (x + \epsilon)^{\frac{1}{\alpha}}$$

Minimizing large values
(smooth fields)

Minimizing divergent fields
(conservation of mass)

Minimizing sharp gradients
(smooth fields)

RESULTS - Comparing with other models

Metric - Mean Squared Error

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

RESULTS - Comparing with other models

Metric - Mean Squared Error

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Model
Numerical model (Béréziat and Herlin 2015)

State of the art numerical model

RESULTS - Comparing with other models

Metric - Mean Squared Error

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Model	
Numerical model (Béréziat and Herlin 2015)	State of the art numerical model
ConvLSTM (Shi *et al* 2015)	Convolutional NN with explicit temporal dimension
CDNN	Convolutional NN without explicit temporal dimension
GAN video generation (Mathieu *et al* 2015)	Generative Adversarial Network without explicit temp. dim.

RESULTS - Comparing with other models

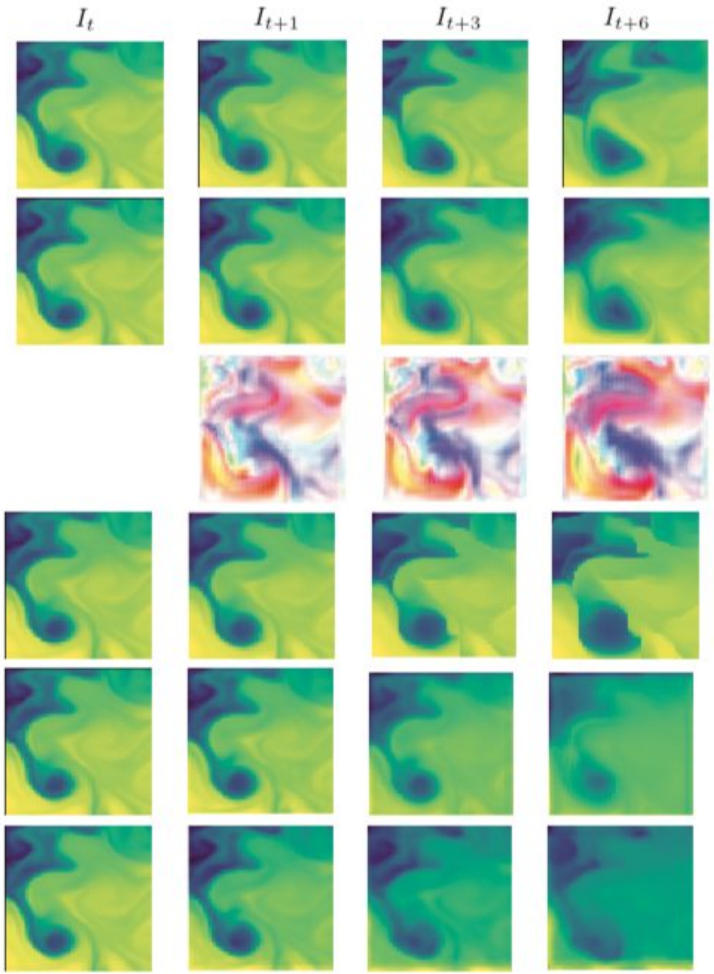
Metric - Mean Squared Error

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Model	
Numerical model (Béréziat and Herlin 2015)	State of the art numerical model
ConvLSTM (Shi *et al* 2015)	Convolutional NN with explicit temporal dimension
CDNN	Convolutional NN without explicit temporal dimension
GAN video generation (Mathieu *et al* 2015)	Generative Adversarial Network without explicit temp. dim.
Proposed model with regularization	Convolutional NN with physical constraints +
Proposed model without regularization	Convolutional NN with physical constraints

Deep learning for physical processes: incorporating prior scientific knowledge
Emmanuel de Bézenac, Arthur Pajot and Patrick Gallinari

- **Ground truth**
- **Their model**
  Convolutional NN with physical constraints +
  - Inferred velocity field
- **State of the art numerical model**
- **CDNN**
  Convolutional NN <u>without</u> explicit temporal dimension
- **ConvLSTM**
  Convolutional NN <u>with</u> explicit temporal dimension



RESULTS - Comparing with other models

Metric - Mean Squared Error

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Model	
Numerical model (Béréziat and Herlin 2015)	State of the art numerical model
ConvLSTM (Shi *et al* 2015)	Convolutional NN with temporal dimension
CDNN	Convolutional NN without explicit temporal dimension
GAN video generation (Mathieu *et al* 2015)	Generative Adversarial Network
Proposed model with regularization	Convolutional NN with physical constraints +
Proposed model without regularization	Convolutional NN with physical constraints

RESULTS - Comparing with other models

Metric - Mean Squared Error

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Model	Average score (MSE)
Numerical model (Béréziat and Herlin 2015)	1.99
ConvLSTM (Shi *et al* 2015)	5.76
CDNN	15.84
GAN video generation (Mathieu *et al* 2015)	4.73
Proposed model with regularization	**1.42**
Proposed model without regularization	2.01

RESULTS - Comparing with other models

Metric - Mean Squared Error

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Model	Average score (MSE)	Average time (s)
Numerical model (Béréziat and Herlin 2015)	1.99	4.8
ConvLSTM (Shi *et al* 2015)	5.76	0.018
CDNN	15.84	0.54
GAN video generation (Mathieu *et al* 2015)	4.73	0.096
Proposed model with regularization	**1.42**	0.040
Proposed model without regularization	2.01	0.040

Deep learning for physical processes: incorporating prior scientific knowledge
Emmanuel de Bézenac, Arthur Pajot and Patrick Gallinari

SUMMARY

- Hybrid model, data-driven solution with physical constraints beats a state of the art numerical model
- Physical constraints improve the prediction, not otherwise!
- Computing time is 2 orders of magnitude shorter with data-driven solutions

MY THOUGHTS

- Using a numerical simulation (NEMO) as ground truth, even with assimilation schemes, could have effects on the verification reliability.
    - The ground truth is not really the truth, it’s a simulation with its biases. A data-driven solution may reproduce it better than another simulation of the truth.
    - Test set should be the real SST from satellite.
- I’m surprised neglecting sinks/sources does not have a clear negative impact on predicting performance. Atmosphere interaction (mixed layer depth), upwelling/downwelling,....
- Most than the performance that can be somehow questioned, the importance of this work lies on the procedure that opens a way of imposing physical constraints to DL models.

More promising studies…

On fluid dynamics and Neural networks

Cornell University

the Simor

arXiv.org > cs > arXiv:2002.09405

Search...

Help | Advance

Computer Science > Machine Learning

Learning to Simulate Complex Physics with Graph Networks

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, Peter W. Battaglia

(Submitted on 21 Feb 2020)

Here we present a general framework for learning simulation, and provide a single model implementation that yields state-of-the-art performance across a variety of challenging physical domains, involving fluids, rigid solids, and deformable materials interacting with one another. Our framework---which we term "Graph Network-based Simulators" (GNS)---represents the state of a physical system with particles, expressed as nodes in a graph, and computes dynamics via learned message-passing. Our results show that our model can generalize from single-timestep predictions with thousands of particles during training, to different initial conditions, thousands of timesteps, and at least an order of magnitude more particles at test time. Our model was robust to hyperparameter choices across various evaluation metrics: the main determinants of long-term performance were the number of message-passing steps, and mitigating the accumulation of error by corrupting the training data with noise. Our GNS framework is the most accurate general-purpose learned physics simulator to date, and holds promise for solving a wide range of complex forward and inverse problems.

Comments: Submitted to ICML 2020

Subjects: **Machine Learning (cs.LG)**; Computational Physics (physics.comp-ph); Machine Learning (stat.ML)

Cite as: arXiv:2002.09405 [cs.LG]

(or arXiv:2002.09405v1 [cs.LG] for this version)

Bibliographic data

[Enable Bibex (What is Bibex?)]

More promising studies…

On fluid dynamics and Neural networks

Cornell University

We … the Simons …

arXiv.org > cs > arXiv:1607.03597

Search...

Help | Advanced …

Computer Science > Computer Vision and Pattern Recognition

Accelerating Eulerian Fluid Simulation With Convolutional Networks

Jonathan Tompson, Kristofer Schlachter, Pablo Sprechmann, Ken Perlin

(Submitted on 13 Jul 2016 (v1), last revised 22 Jun 2017 (this version, v6))

Efficient simulation of the Navier-Stokes equations for fluid flow is a long standing problem in applied mathematics, for which state-of-the-art methods require large compute resources. In this work, we propose a data-driven approach that leverages the approximation power of deep-learning with the precision of standard solvers to obtain fast and highly realistic simulations. Our method solves the incompressible Euler equations using the standard operator splitting method, in which a large sparse linear system with many free parameters must be solved. We use a Convolutional Network with a highly tailored architecture, trained using a novel unsupervised learning framework to solve the linear system. We present real-time 2D and 3D simulations that outperform recently proposed data-driven methods; the obtained results are realistic and show good generalization properties.

Comments: Significant revision
Subjects: **Computer Vision and Pattern Recognition (cs.CV)**
Cite as: arXiv:1607.03597 [cs.CV]
(or arXiv:1607.03597v6 [cs.CV] for this version)

Bibliographic data

[Enable Bibex (What is Bibex?)]