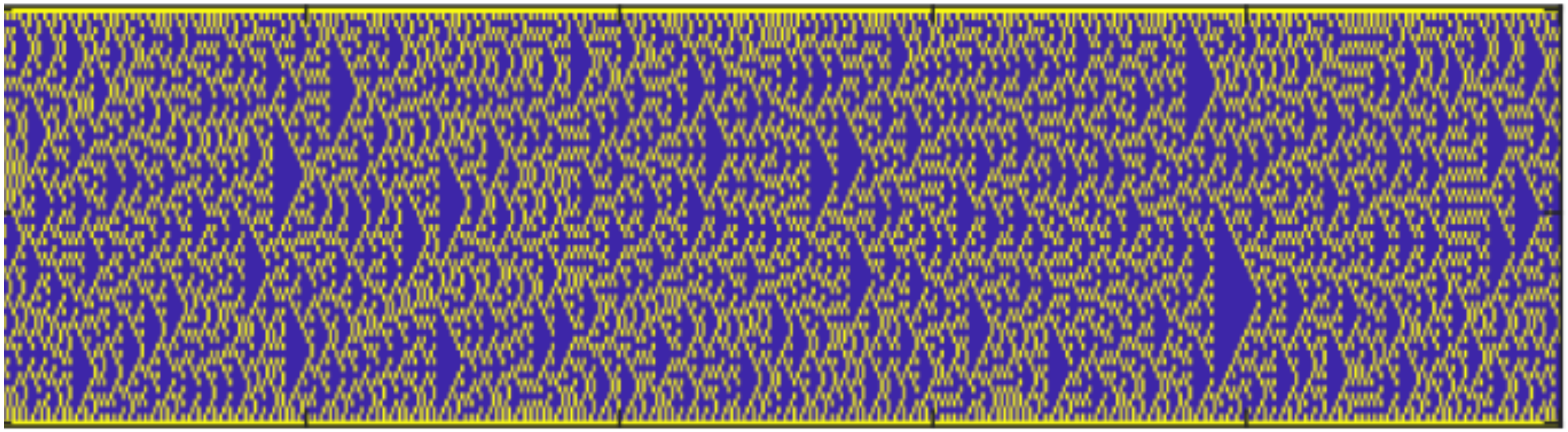
MPOs for sequence to sequence learning



CQT
13/07/2018

Dario Poletti

Singapore University of Technology and Design

SUTD-MIT INTERNATIONAL DESIGN CENTRE (IDC)

Ministry of Education SINGAPORE

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

MPOs for sequence to sequence learning

Machine learning
- supervised, unsupervised
- some tasks
- classical and quantum

Many body quantum physics
- using machine learning
- interpreting machine learning
- Novel algorithms

MPS for classification

MPO for sequence to sequence

Outlook

MPOs for sequence to sequence learning

Machine learning
- **supervised, unsupervised**
- some tasks
- classical and quantum

Supervised: you feed possible inputs and outputs to the model which tries to figure out which conditional probability they were extracted from

MNIST data set

MPOs for sequence to sequence learning

Machine learning
- **supervised, unsupervised**
- some tasks
- classical and quantum

Supervised: you feed possible inputs and outputs to the model which tries to figure out which conditional probability they were extracted from

MNIST data set

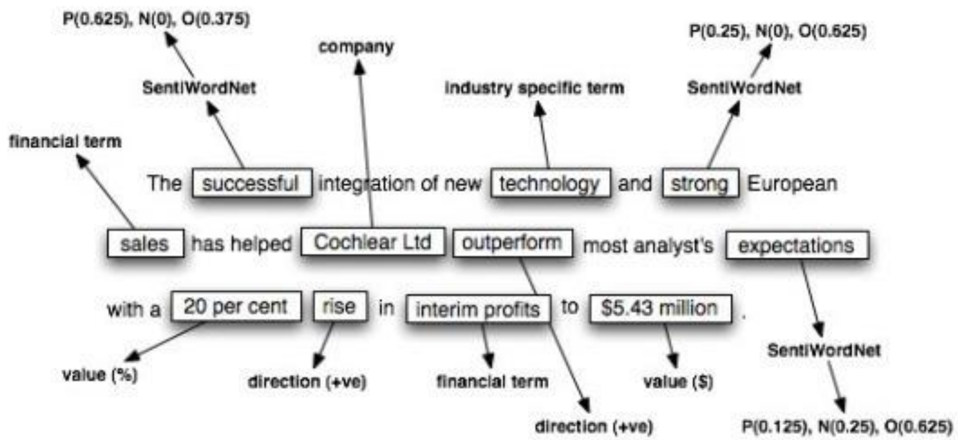
Unsupervised: you feed unlabeled data and the model infers a function that describes an inner structure

Han et al. arxiv:1709.01662

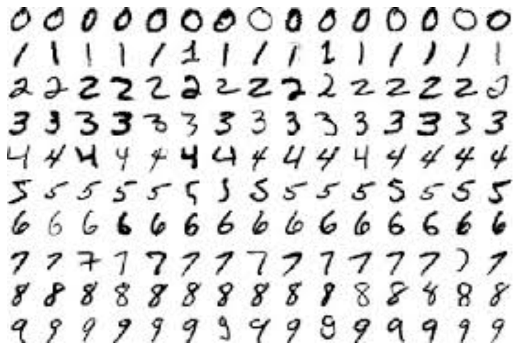
MPOs for sequence to sequence learning

Machine learning
- supervised, unsupervised
- **some tasks**
- classical and quantum

Natural language processing

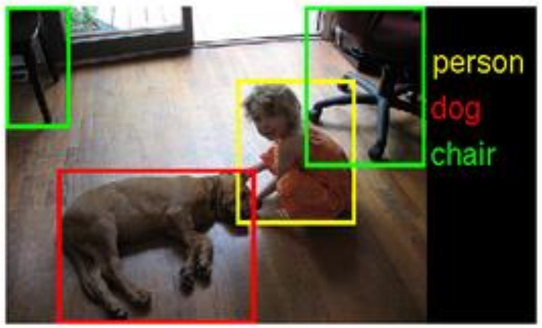


classification

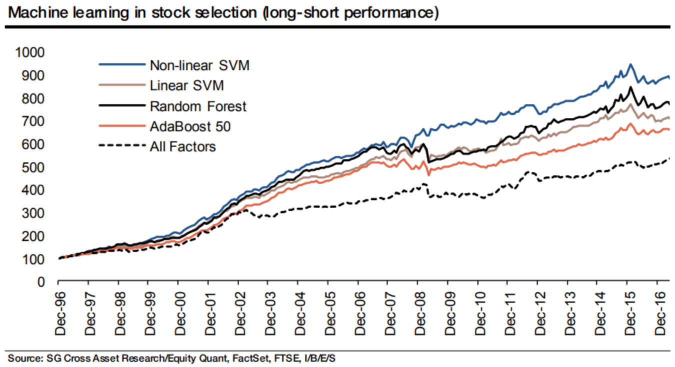


translation

image recognition



stock prediction



MPOs for sequence to sequence learning

Machine learning
- supervised, unsupervised
- some tasks
- **classical and quantum**

You can use a classical computer or a quantum computer

-> use wave-functions, use quantum algorithms

MPOs for sequence to sequence learning

Many body quantum physics
- using machine learning
- interpreting machine learning
- Novel algorithms

https://physicsml.github.io/pages/papers.html

MPOs for sequence to sequence learning

Many body quantum physics
- **using machine learning**
- interpreting machine learning
- Novel algorithms

Machine learning for phase recognitions

J. Carrasquilla, and R.G. Melko, Nature Physics 13, 431 (2017).
Y. Zhang, and E.A. Kim, Phys. Rev. Lett. 118, 216401 (2017).
K. Ch'ng, N. Vazquez, and E. Khatami, Phys. Rev. E 97, 013306 (2018).
P. Zhang, H. Shen, and H. Zhai, Phys. Rev. Lett. 120, 066401 (2018).
E. P. van Nieuwenburg, Y.-H. Liu, and S. D. Huber, Nat. Phys. 13, 435 (2017).
P. Broecker, J. Carrasquilla, R. G. Melko, and S. Trebst, Scientific Reports 7, 8823 (2017).
K. Ch'ng, J. Carrasquilla, R. G. Melko, and E. Khatami, Phys. Rev. X 7, 031038 (2017).
…

MPOs for sequence to sequence learning

Many body quantum physics
- **using machine learning**
- interpreting machine learning
- Novel algorithms

Boost efficiency of codes or provide new codes

L.F. Arsenault, A. Lopez-Bezanilla, O.A. von Lilienfeld, and A.J. Millis, Phys. Rev. B 90, 155136 (2014).
L.-F. Arsenault, O. A. von Lilienfeld, and A. J. Millis, arXiv:1506.08858 (2015).
G. Torlai and R. G. Melko, Phys. Rev. B 94, 165134 (2016).
M. H. Amin, E. Andriyash, J. Rolfe, B. Kulchytskyy, and R. Melko, arXiv:1601.02036.
J. Liu, Y. Qi, Z. Y. Meng, and L. Fu, Phys. Rev. B 95, 041101 (2017).
L. Huang and L. Wang, Phys. Rev. B 95, 035105 (2017).
K.-I. Aoki and T. Kobayashi, Mod. Phys. Lett. B, 1650401 (2016).
G. Carleo, and M. Troyer, Science 355, 602 (2017).
Y. Nomura, A.S. Darmawan, Y. Yamaji, and M. Imada, Phys. Rev. B 96, 205152 (2017).
S. Czischek, M. Garttner, and T. Gasenzer, arxiv:1803.08321.
…

MPOs for sequence to sequence learning

Many body quantum physics
- using machine learning
- **interpreting machine learning**
- Novel algorithms

Interpreting the networks and/or why they work

C. Beny, arXiv:1301.3124.
P. Mehta and D. J. Schwab, arXiv:1410.3831.
H.W. Lin, M. Tegmark, and D. Rolnick, Journal of Statistical Physics 168, 1223 (2017).

Why does deep and cheap learning work so well?*

Henry W. Lin, Max Tegmark, and David Rolnick
Dept. of Physics, Harvard University, Cambridge, MA 02138
Dept. of Physics, Massachusetts Institute of Technology, Cambridge, MA 02139 and
Dept. of Mathematics, Massachusetts Institute of Technology, Cambridge, MA 02139

(Dated: July 21 2017)

We show how the success of deep learning could depend not only on mathematics but also on physics: although well-known mathematical theorems guarantee that neural networks can approximate arbitrary functions well, the class of functions of practical interest can frequently be approximated through "cheap learning" with exponentially fewer parameters than generic ones. We explore how properties frequently encountered in physics such as symmetry, locality, compositionality, and polynomial log-probability translate into exceptionally simple neural networks. We further argue that when the statistical process generating the data is of a certain hierarchical form prevalent in physics and machine-learning, a deep neural network can be more efficient than a shallow one. We formalize these claims using information theory and discuss the relation to the renormalization group. We prove various "no-flattening theorems" showing when efficient linear deep networks cannot be accurately approximated by shallow ones without efficiency loss; for example, we show that n variables cannot be multiplied using fewer than 2^n neurons in a single hidden layer.

MPOs for sequence to sequence learning

Many body quantum physics
- using machine learning
- interpreting machine learning
- **Novel algorithms**

Use quantum many body inspired algorithms to prepare new machine learning models

With MPS

E.M. Stoudenmire and D.J. Schwab, Advances In Neural Information Processing Systems 29, 4799 (2016).
Z.-Y. Han, J. Wang, H. Fan, L. Wang, and P. Zhang, arxiv:1709.01662.
E.M. Stoudenmire, Quantum Science and Technology (2018).
A. Novikov, M. Trofimov, and I. Oseledets, arxiv:1605.03795, ICLR (2017).
Huggins et al. arxiv:1803.11537
our work
…

MPOs for sequence to sequence learning: **MPS basics**

BIG PROBLEM!!!

- Size of $|\psi >$ scales as d^L where d is the size of the local Hilbert space while L is the size of the system

$$|\psi\rangle = \sum_{\sigma_1,\sigma_2,\cdots,\sigma_L} C^{\sigma_1,\sigma_2,\cdots,\sigma_L} |\sigma_1, \sigma_2, \cdots, \sigma_L\rangle$$

MPOs for sequence to sequence learning: **MPS basics**

BIG PROBLEM!!!

- Size of $|\psi>$ scales as d^L where d is the size of the local Hilbert space while L is the size of the system

$$|\psi\rangle = \sum_{\sigma_1,\sigma_2,\cdots,\sigma_L} C^{\sigma_1,\sigma_2,\cdots,\sigma_L}|\sigma_1,\sigma_2,\cdots,\sigma_L\rangle$$

- Rewrite $|\psi>$ as the product of a series of 3-dimensional tensors

$$c_{\sigma_1,\ldots,\sigma_L} = \sum_{a_1,a_2,\ldots,a_{L+1}} M^{\sigma_1}_{a_1,a_2} M^{\sigma_2}_{a_2,a_3} \ldots M^{\sigma_L}_{a_L,a_{L+1}}$$

- Now it scales polynomial with system size dD^2L

Standard tool for 1-dimensional quantum many body systems!

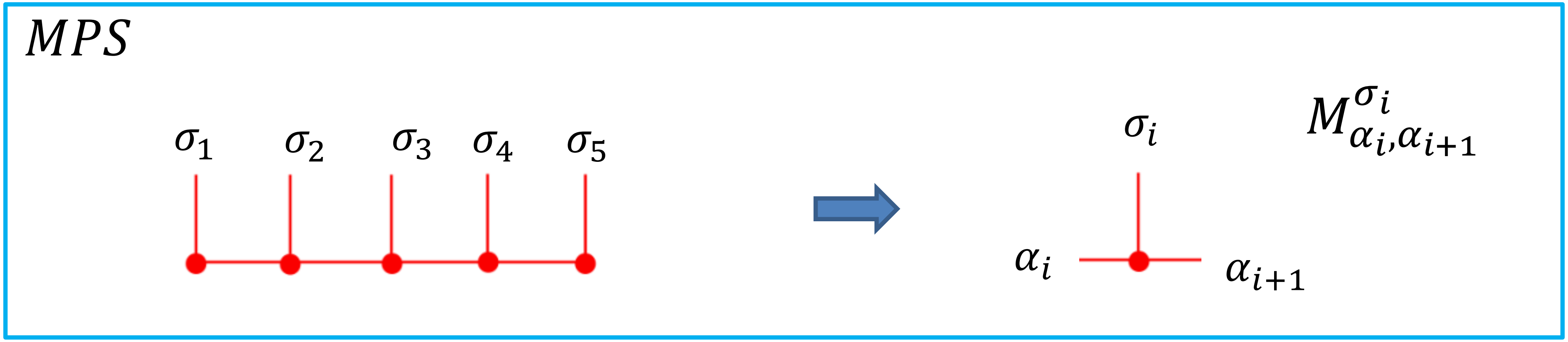
U. Schollwock, Ann. Phys. 326, 96 (2011)

MPOs for sequence to sequence learning: **MPS basics**

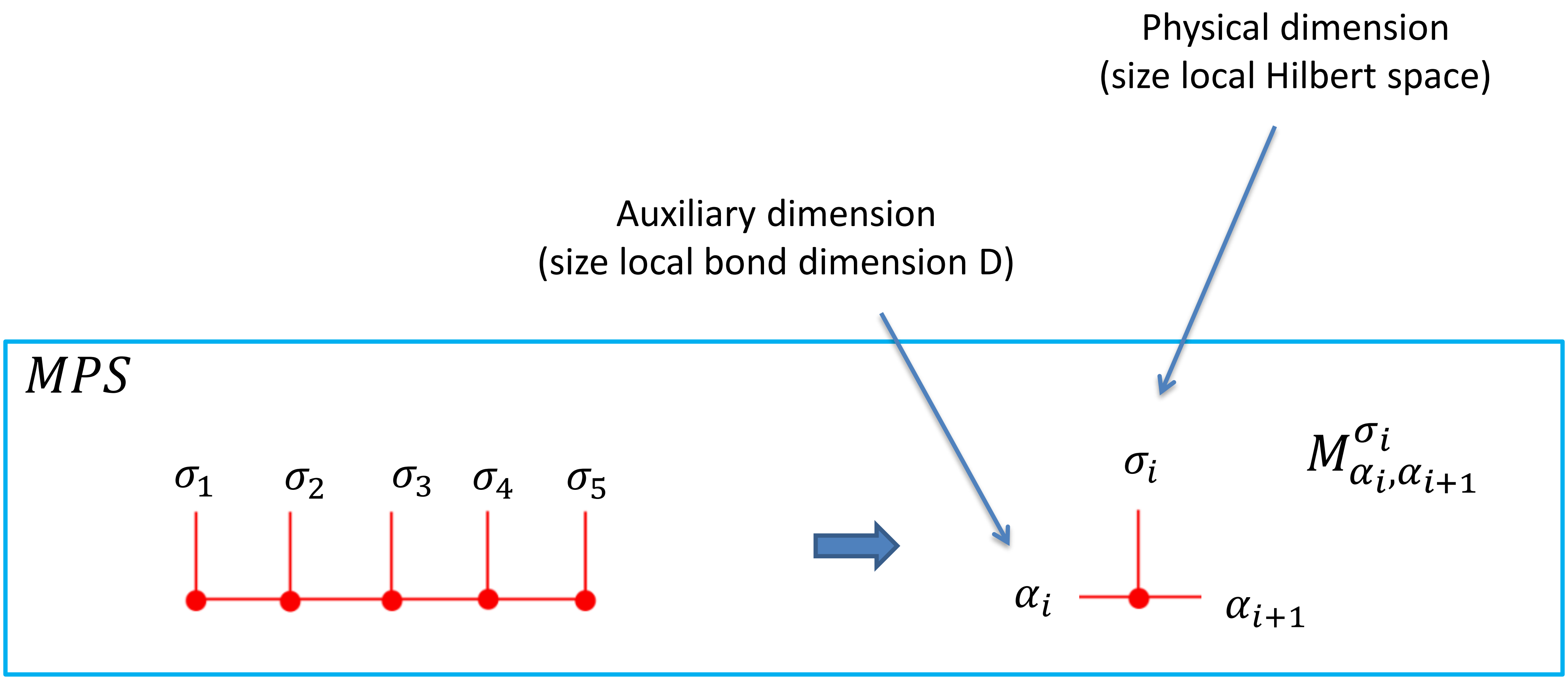
$$c_{\sigma_1,\ldots,\sigma_L} = \sum_{a_1,a_2,\ldots,a_{L+1}} M^{\sigma_1}_{a_1,a_2} M^{\sigma_2}_{a_2,a_3} \ldots M^{\sigma_L}_{a_L,a_{L+1}}$$

MPOs for sequence to sequence learning: **MPS basics**

$$c_{\sigma_1,\ldots,\sigma_L} = \sum_{a_1,a_2,\ldots,a_{L+1}} M^{\sigma_1}_{a_1,a_2} M^{\sigma_2}_{a_2,a_3} \ldots M^{\sigma_L}_{a_L,a_{L+1}}$$

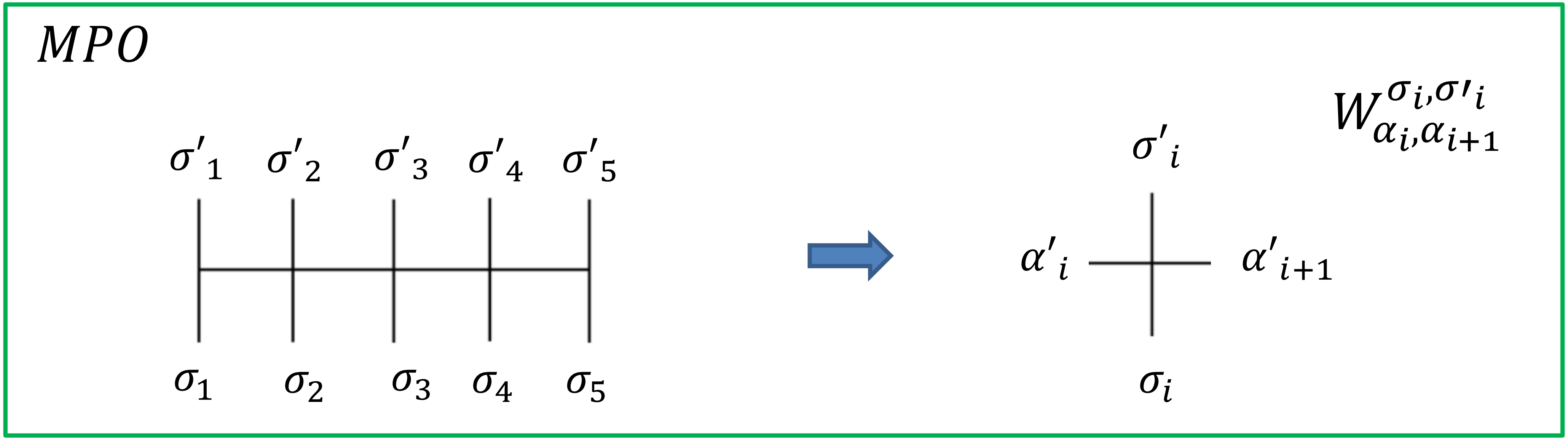


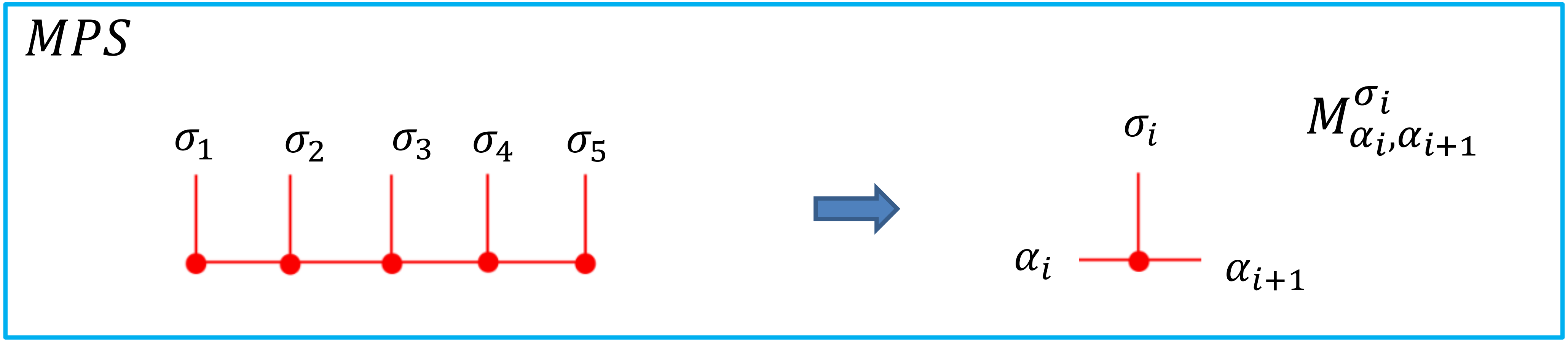
MPOs for sequence to sequence learning: **MPS basics**



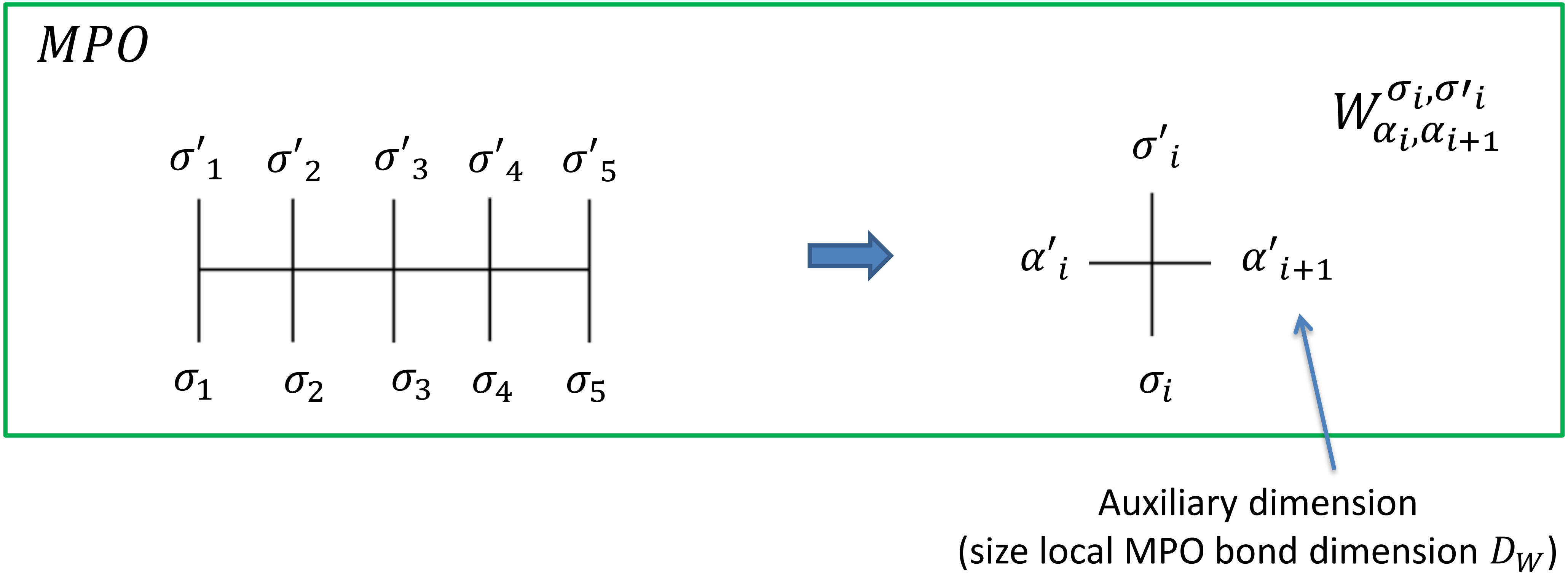
$M^{\sigma_i}_{\alpha_i,\alpha_{i+1}}$

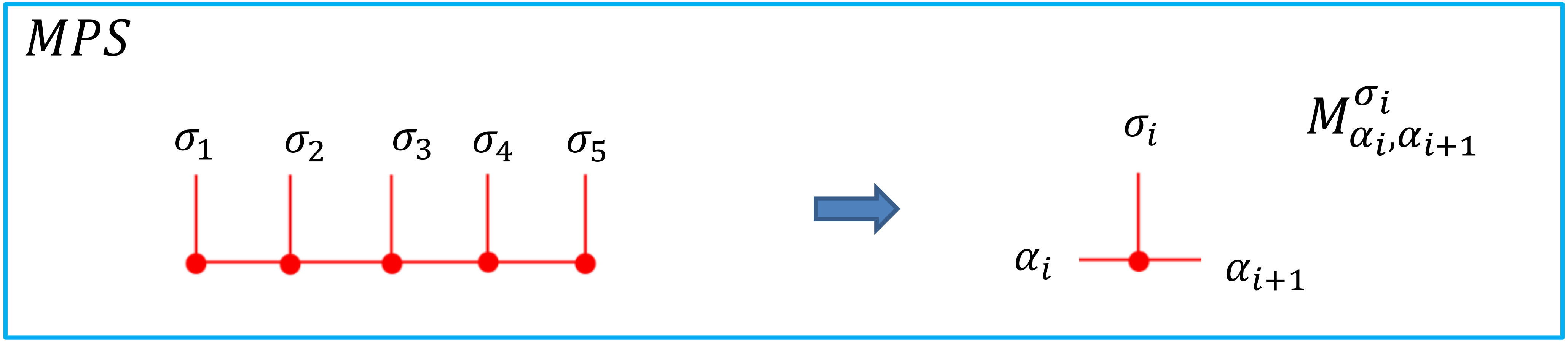
MPOs for sequence to sequence learning: **MPS basics**





MPOs for sequence to sequence learning: **MPS basics**





MPOs for sequence to sequence learning

MPO for sequence to sequence

- Method
- Cellular automata
    - Analytical solutions
    - Performance
    - Comparison to Conditional Random Fields
- Discrete nonlinear maps
    - Performance
    - Comparison to LSTM
- Classification

MPOs for sequence to sequence learning: **method**

Words, sentences, data, images can be written as a sequence

$$(x_{k,1}, x_{k,2}, \dots, x_{k,L})$$

and there is some function which from the input sequence returns an output sequence

$$\vec{y}_i = f(\vec{x}_i)$$

Can we train an MPO that would return an output

$$(\bar{y}_{k,1}, \bar{y}_{k,2}, \dots, \bar{y}_{k,L}).$$

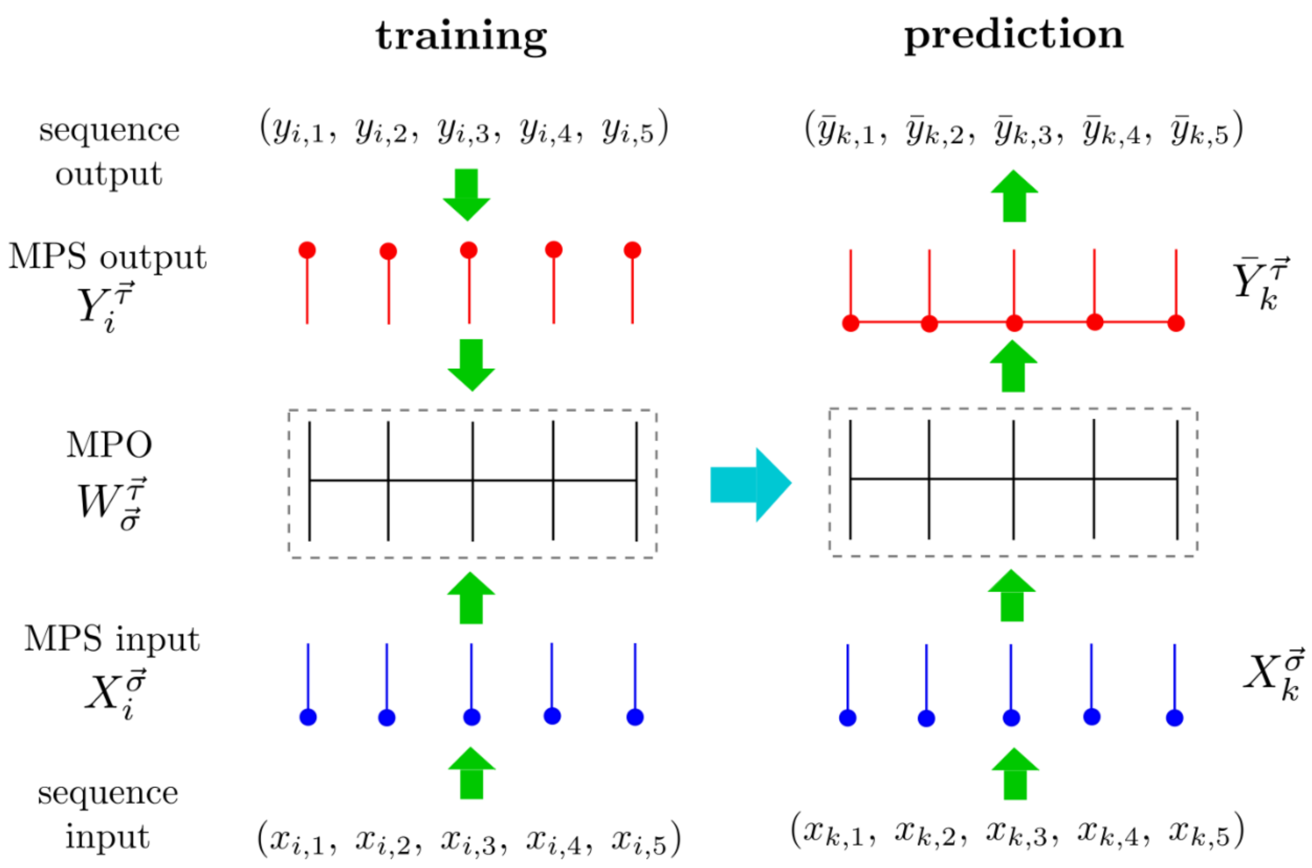
which is close to $\vec{y}_i$?

Here we will focus on equal input and output sizes L.

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **method**

The main points of our MPO model are described in this figure

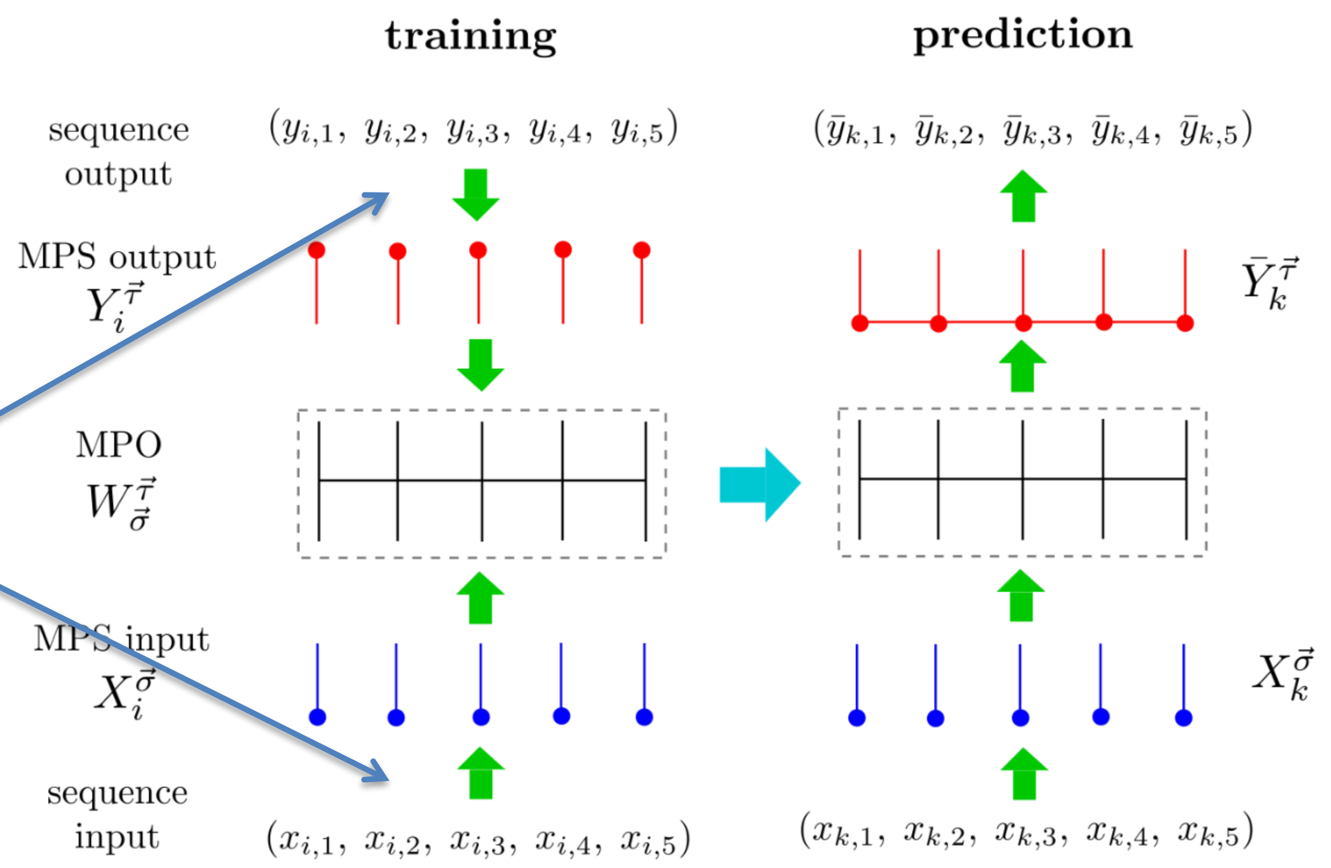


C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **method**

From **input/output sequences** to **input/output MPSs**

Conversion of sequence to MPS



C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **method**

For real number, for example from 0 to 1, we map each site to a vector

$$\left(\sqrt{1-x_{i,l}^2}\,, x_{i,l}\right)$$

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **method**

For real number, for example from 0 to 1, we map each site to a vector

$$\left(\sqrt{1 - x_{i,l}^2}\,,\, x_{i,l}\right)$$

For integer numbers you choose a vector of the size given by the possible outcomes
The you put a 1 in the location corresponding to the value

0 -> (1,0,0,0,0,0)
…
3 -> (0,0,0,1,0,0)
…

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **method**

For real number, for example from 0 to 1, we map each site to a vector

$$\left(\sqrt{1-x_{i,l}^2}\,,\, x_{i,l}\right)$$

For integer numbers you choose a vector of the size given by the possible outcomes
The you put a 1 in the location corresponding to the value

0 -> (1,0,0,0,0,0)
…
3 -> (0,0,0,1,0,0)
…

Then

$$\vec{x}_i \to X_i^{\vec{\sigma}} = \sum_{a_0,\ldots,a_L} X_{i,a_0,a_1}^{\sigma_1} X_{i,a_1,a_2}^{\sigma_2} \ldots X_{i,a_{L-1},a_L}^{\sigma_L} \qquad \text{inputs}$$

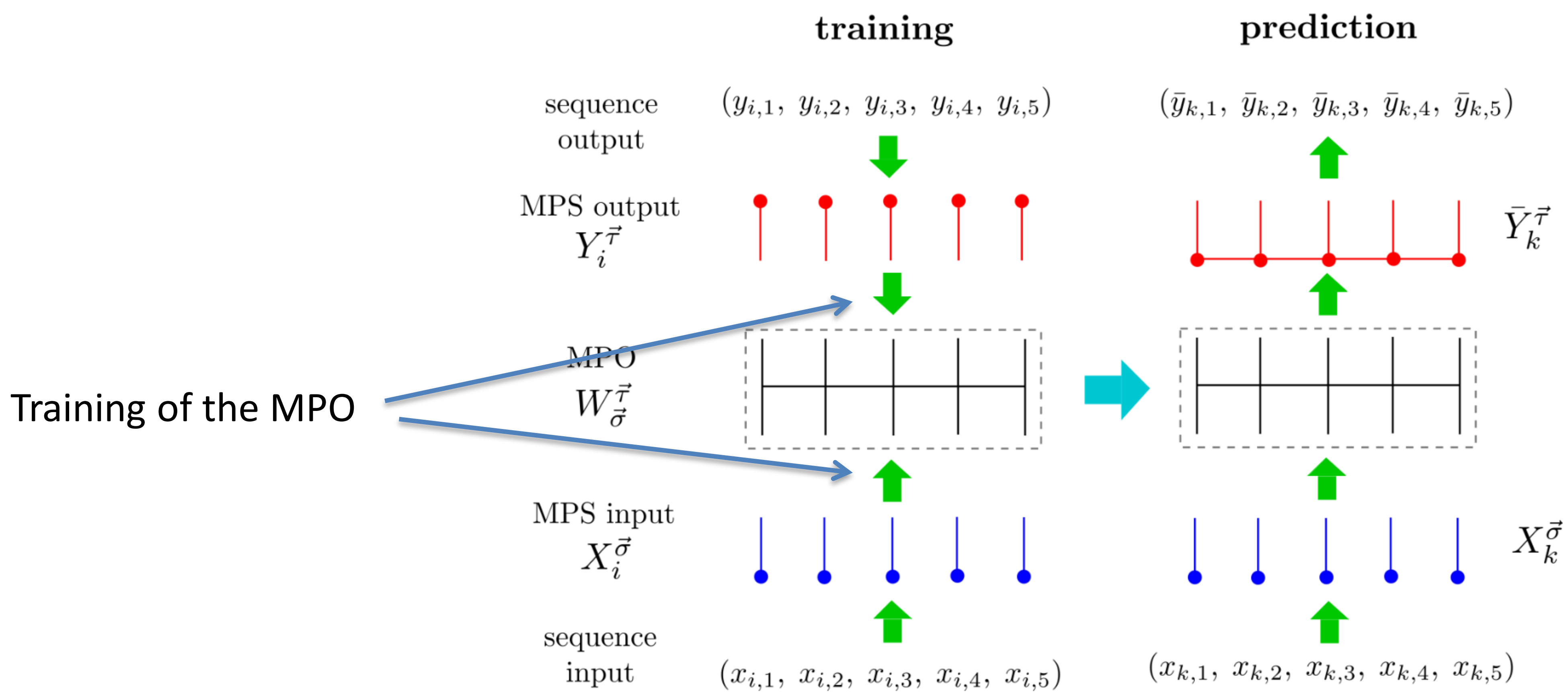
$$\vec{y}_i \to Y_i^{\vec{\tau}} = \sum_{c_0,\ldots,c_L} Y_{i,c_0,c_1}^{\tau_1} Y_{i,c_1,c_2}^{\tau_2} \ldots Y_{i,c_{L-1},c_L}^{\tau_L} \qquad \text{outputs}$$

where all the $a_l = c_l = 1$ only.

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **method**

From **input/output MPS** to **trained MPO**



C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **method**

We define the **cost function C**

$$C(W_{\vec{\sigma}}^{\vec{\tau}}) = \sum_{i=1}^{N} \left(\bar{Y}_i^{\vec{\tau}\dagger} - Y_i^{\vec{\tau}\dagger}\right) \left(\bar{Y}_i^{\vec{\tau}} - Y_i^{\vec{\tau}}\right) + \alpha \, \mathrm{tr} \left(W_{\vec{\sigma}}^{\vec{\tau}\dagger} W_{\vec{\sigma}}^{\vec{\tau}}\right)$$

$$\bar{Y}_i^{\vec{\tau}} = W_{\vec{\sigma}}^{\vec{\tau}} X_i^{\vec{\sigma}}$$

Predicted output

Distance between predicted and output MPS

Regularization term

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **method**

We define the **cost function C**

$$C(W_{\vec{\sigma}}^{\vec{\tau}}) = \sum_{i=1}^{N} \left(\bar{Y}_i^{\vec{\tau}\dagger} - Y_i^{\vec{\tau}\dagger}\right) \left(\bar{Y}_i^{\vec{\tau}} - Y_i^{\vec{\tau}}\right) + \alpha \, \mathrm{tr} \left(W_{\vec{\sigma}}^{\vec{\tau}\dagger} W_{\vec{\sigma}}^{\vec{\tau}}\right)$$

$$\bar{Y}_i^{\vec{\tau}} = W_{\vec{\sigma}}^{\vec{\tau}} X_i^{\vec{\sigma}}$$

Predicted output

Distance between predicted and output MPS

Regularization term

We minimize the cost function iteratively over the local MPOs W

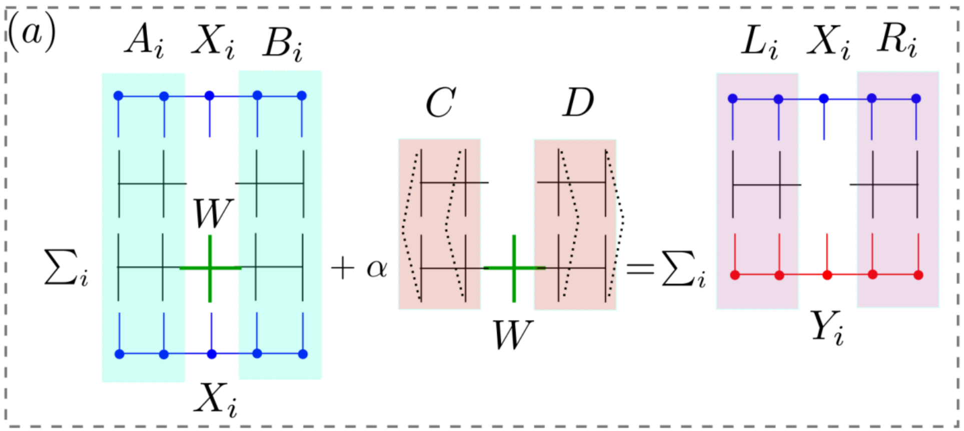
$$\frac{\partial C(\hat{W})}{\partial W_{b_{l-1},b_l}^{\sigma_l,\tau_l}} = 0.$$

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **method**

Minimization is graphically depicted here

$$\frac{\partial C(\hat{W})}{\partial W^{\sigma_l,\tau_l}_{b_{l-1},b_l}} = 0.$$



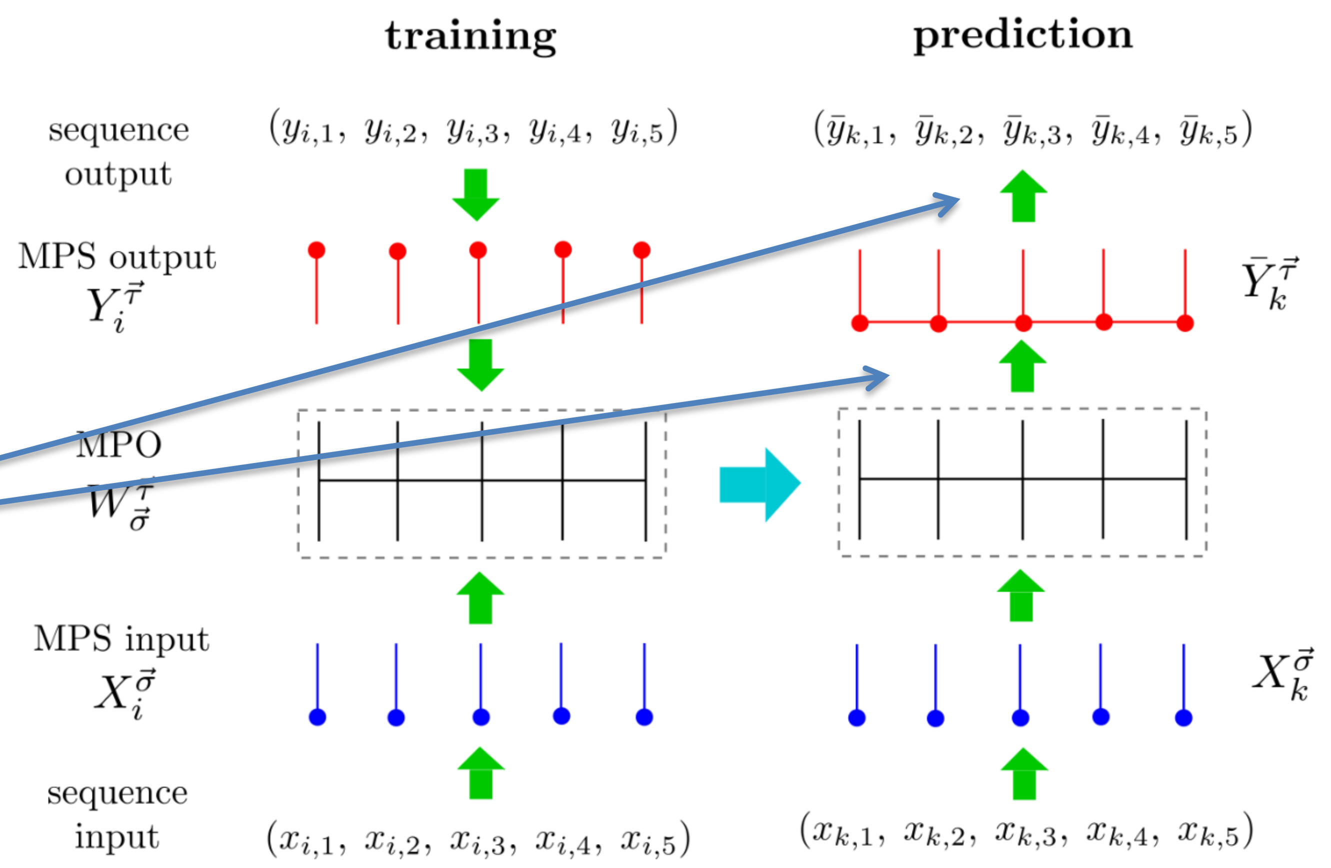
Fairly straightforward linear problem to solve $\quad M\,W = V$

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **method**

Generation of **output MPS** and its conversion to **sequence**

MPS prediction
+
Sequence prediction



C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **method**

The **output MPS** is simply given by the product of **input MPS** with the **trained MPO**

$$\bar{Y}_i^{\vec{\tau}} = W_{\vec{\sigma}}^{\vec{\tau}} X_i^{\vec{\sigma}} = \sum_{\bar{c}_0,\ldots,\bar{c}_L} \bar{Y}_{i,\bar{c}_0,\bar{c}_1}^{\tau_1} \ldots \bar{Y}_{i,\bar{c}_{L-1},\bar{c}_L}^{\tau_L},$$

where $$\bar{Y}_{i,\bar{c}_{l-1},\bar{c}_l}^{\tau_l} = \sum_{\sigma_l} W_{b_{l-1},b_l}^{\sigma_l,\tau_l} X_{i,a_{l-1},a_l}^{\sigma_l}.$$

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **method**

The **output MPS** is simply given by the product of **input MPS** with the **trained MPO**

$$\bar{Y}_i^{\vec{\tau}} = W_{\vec{\sigma}}^{\vec{\tau}} X_i^{\vec{\sigma}} = \sum_{\bar{c}_0,\ldots,\bar{c}_L} \bar{Y}^{\tau_1}_{i,\bar{c}_0,\bar{c}_1} \ldots \bar{Y}^{\tau_L}_{i,\bar{c}_{L-1},\bar{c}_L},$$

where $$\bar{Y}^{\tau_l}_{i,\bar{c}_{l-1},\bar{c}_l} = \sum_{\sigma_l} W^{\sigma_l,\tau_l}_{b_{l-1},b_l} X^{\sigma_l}_{i,a_{l-1},a_l}.$$

Conversion of output MPS to **sequence**

1) We convert the MPS to a bond dimension D=1 MPS
2) We check with physical index σ_l has the largest occupation
3) We chose this physical index as the value for the sequence

$(x_{k,1},\ x_{k,2},\ x_{k,3},\ x_{k,4},\ x_{k,5})$

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

We can now study the performance

We will study two sequence to sequence problems
- Evolution of cellular automata
- Discrete nonlinear maps

And one classification problem
- MNIST digits classification

C. Guo, et al. arxiv:1803.10908 (2018)

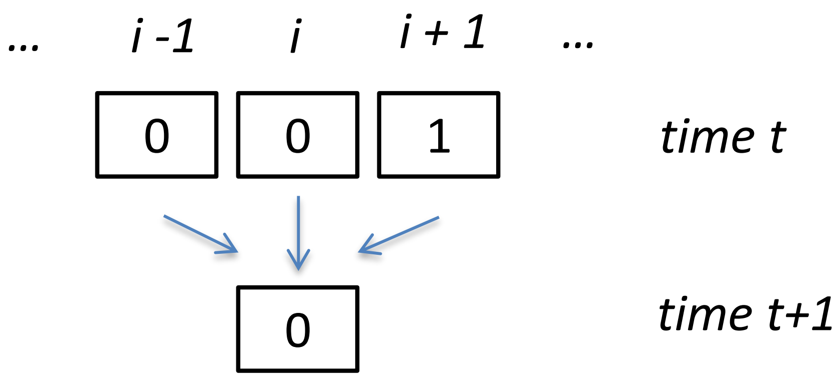
MPOs for sequence to sequence learning: **performance**

Cellular automata

MPOs for sequence to sequence learning: **performance**

Cellular automata

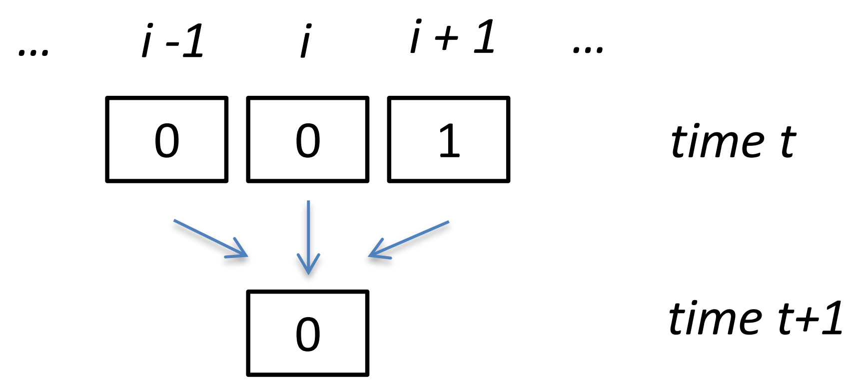
256 rules have been classified (Wolfram1980)
Given the value at one position and its nearest neighbors, the position at the next time step is decided



MPOs for sequence to sequence learning: **performance**

Cellular automata

256 rules have been classified (Wolfram1980)
Given the value at one position and its nearest neighbors, the position at the next time step is decided



We also consider **long-range** rules, where the evolution at site i depends on site $i+d$ with $d>1$.

We focus on rules 153, 153-long-range, 18 and 30 … easy to figure what each rule does …

MPOs for sequence to sequence learning: **performance**

The **evolution** of a string of 0s and 1s due to some cellular automata rules can be written **exactly** using **MPOs**.
For the 256 rules, the MPO bond dimension D_W is at most 4.

MPOs for sequence to sequence learning: **performance**

The **evolution** of a string of 0s and 1s due to some cellular automata rules can be written **exactly** using **MPOs**.
For the 256 rules, the MPO bond dimension D_W is at most 4.

Example: Rule 153 (for which D_W =2)

The product of the MPOs chosen in this manner is only different from 0 for the correct output sequence.

$$W^{0,0}_{b_{l-1},b_l} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad W^{0,1}_{b_{l-1},b_l} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$
$$W^{1,1}_{b_{l-1},b_l} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \quad W^{1,0}_{b_{l-1},b_l} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}$$

MPOs for sequence to sequence learning: **performance**

The **evolution** of a string of 0s and 1s due to some cellular automata rules can be written **exactly** using **MPOs**.
For the 256 rules, the MPO bond dimension D_W is at most 4.

Example: Rule 153 (for which D_W =2)

The product of the MPOs chosen in this manner is only different from 0 for the correct output sequence.

$$W^{0,0}_{b_{l-1},b_l} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad W^{0,1}_{b_{l-1},b_l} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$
$$W^{1,1}_{b_{l-1},b_l} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \quad W^{1,0}_{b_{l-1},b_l} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}$$

We use fix boundary conditions which translate to

$$W^{0,0}_{b_0,b_1} = [0,\ 1], \quad W^{0,1}_{b_0,b_1} = [1,\ 0]$$
$$W^{1,1}_{b_0,b_1} = [0,\ 1], \quad W^{1,0}_{b_0,b_1} = [1,\ 0]$$

First site

$$W^{0,0}_{b_{L-1},b_L} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad W^{0,1}_{b_{L-1},b_L} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$
$$W^{1,1}_{b_{L-1},b_L} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad W^{1,0}_{b_{L-1},b_L} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Last site

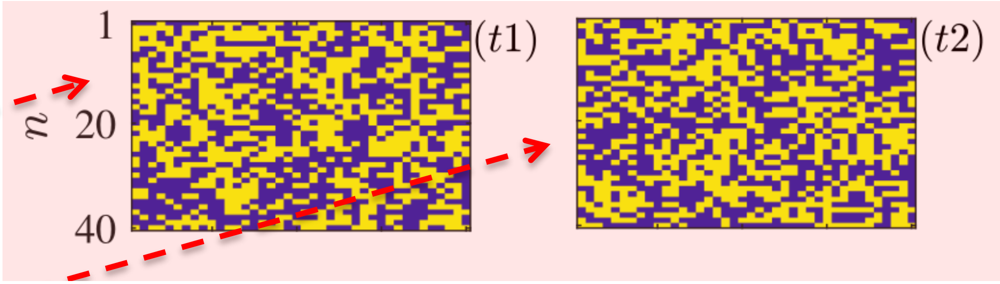
MPOs for sequence to sequence learning: **performance**

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

Example of 40
training **input** data

Example of
corresponding 40
training **output** data



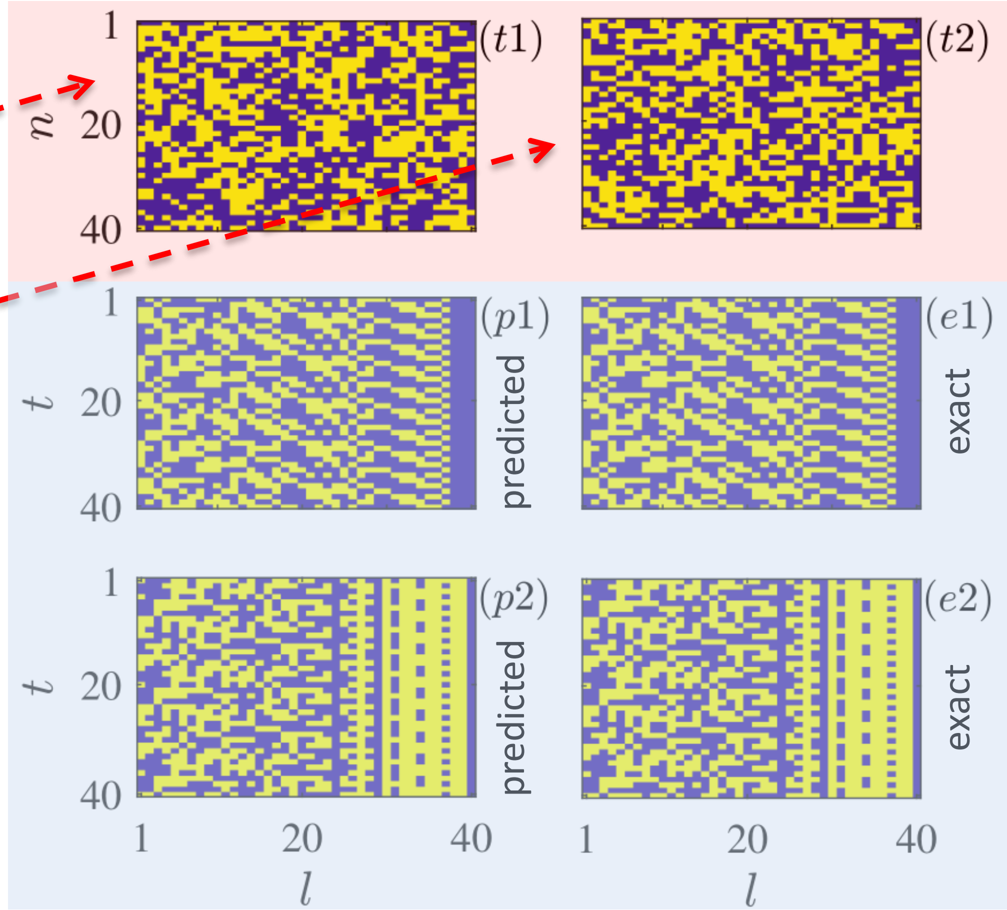
C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

Example of 40 training **input** data

Example of corresponding 40 training **output** data

2 examples of **perfectly predicted** evolution for long range rule 153



C. Guo, et al. arxiv:1803.10908 (2018)

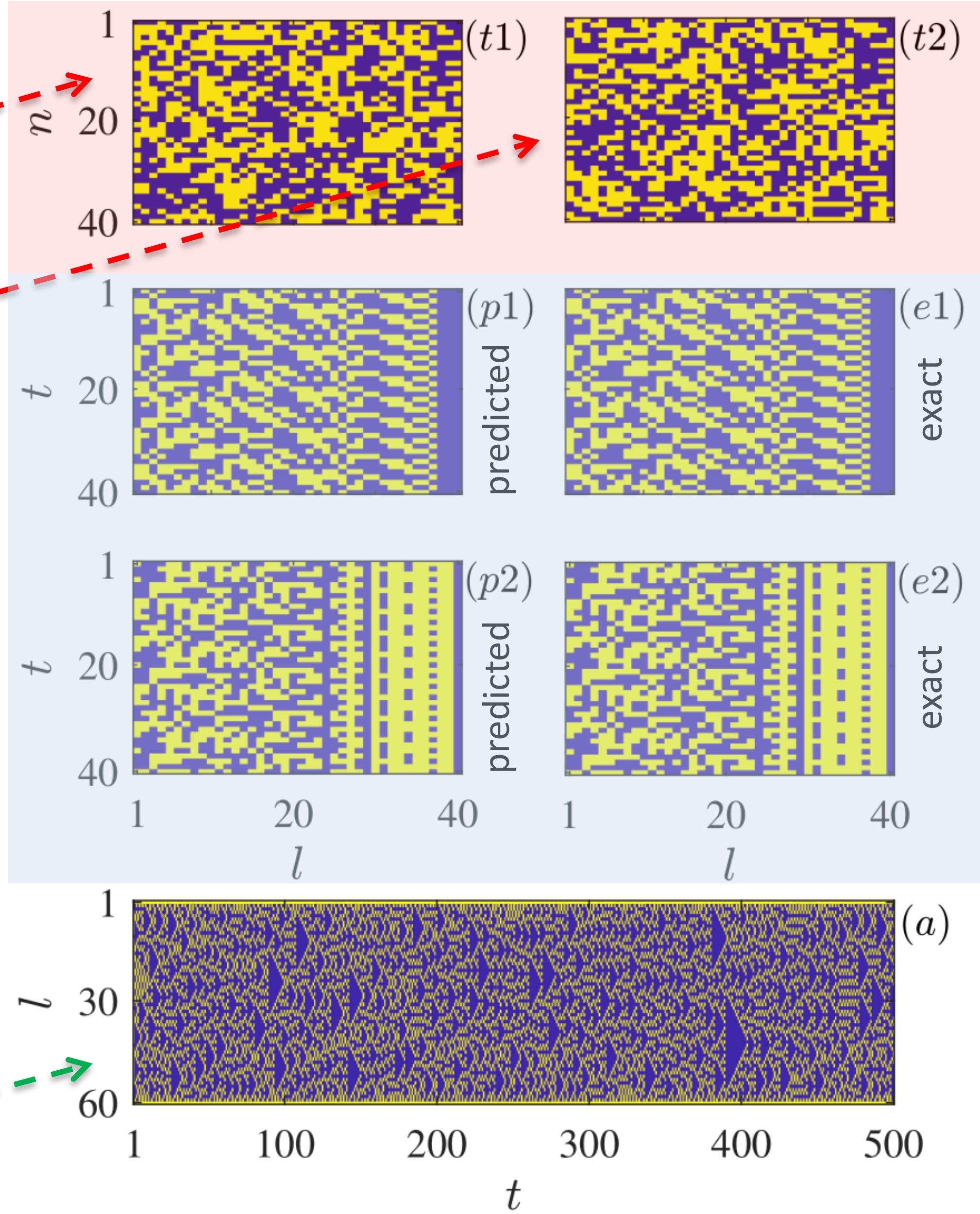
MPOs for sequence to sequence learning: **performance**

Example of 40 training **input** data

Example of corresponding 40 training **output** data

2 examples of **perfectly predicted** evolution for long range rule 153

example of rule 18 over 60 sites with evolution with **long period**



C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

We quantify the error ε as

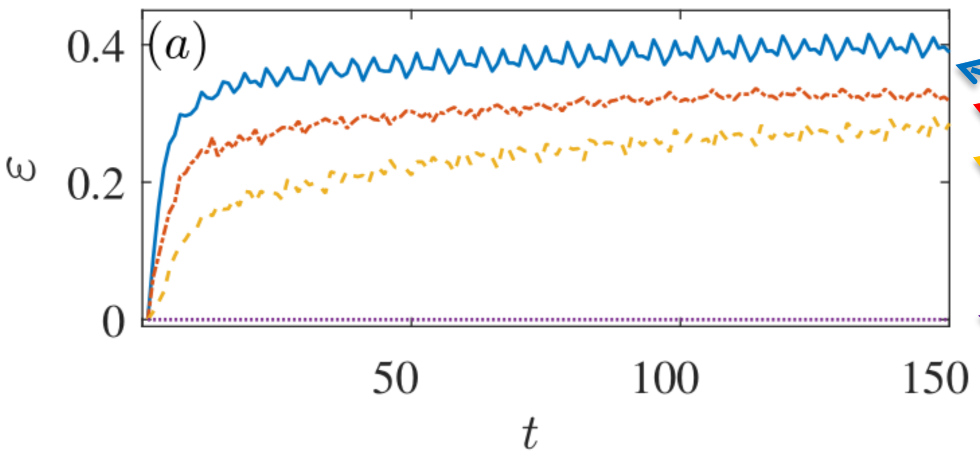
$$\varepsilon = \sum_{k,l} |y_{k,l}(t) - \bar{y}_{k,l}(t)|/(L\ N)$$

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

We quantify the error ε as

$$\varepsilon = \sum_{k,l} |y_{k,l}(t) - \bar{y}_{k,l}(t)|/(L\ N)$$



D_W=5

D_W=6

D_W=7

D_W=8 Perfect evolution

We consider N=100 initial conditions for a system of size L=40 and we evolve for 150 steps. We used M=7000 samples to train.

If the bond dimension is large enough, we get perfect predictions.

C. Guo, et al. arxiv:1803.10908 (2018)

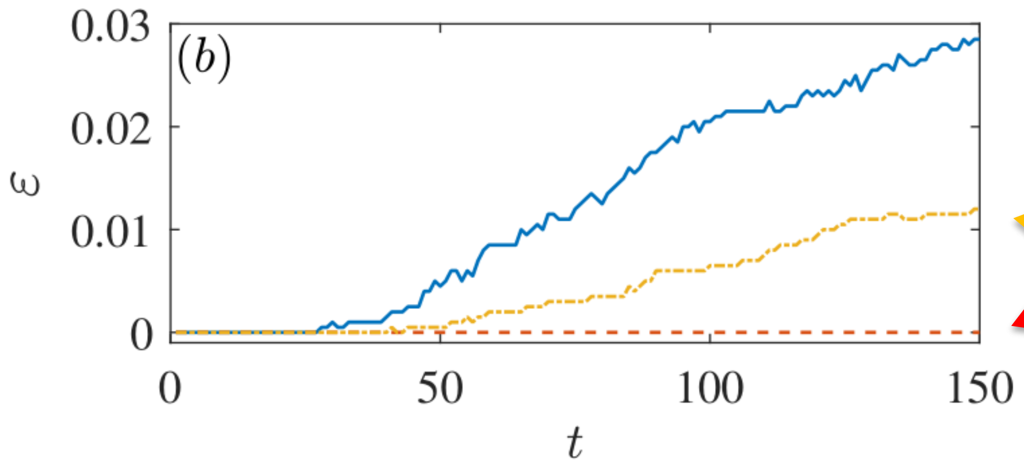
MPOs for sequence to sequence learning: **performance**

We also consider the presence of wrong training data. The quantify the error ε_r is the percentage of output data which are chosen from uniform random distribution.

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

We also consider the presence of wrong training data. The quantify the error ε_r is the percentage of output data which are chosen from uniform random distribution.



M=3000 $\varepsilon_r = 0.2$

M=7000 $\varepsilon_r = 0.2$
Perfect evolution

M=7000 $\varepsilon_r = 0.4$

We consider N=100 initial conditions for a system of size L=40 and we evolve for 150 steps. We used D_W=8 bond dimension.

Larger sample size can compensate for the errors in the training data.

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

Comparison to conditional random fields (CRF) model

Some **weights** vector
which will be optimized

Some vector function of
the input and output
vectors -> **features**

$$p(\vec{y}_i|\vec{x}_i) = \frac{\exp(\vec{w}^{\mathbf{T}} \vec{f}(\vec{x}_i, \vec{y}_i))}{Z(\vec{x})}$$

$$Z(\vec{x}_i) = \sum_{\vec{y}_k} \exp(\vec{w}^{\mathbf{T}} \vec{f}(\vec{x}_i, \vec{y}_k))$$

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

Comparison to conditional random fields (CRF) model

Some **weights** vector which will be optimized

Some vector function of the input and output vectors -> **features**

$$p(\vec{y}_i|\vec{x}_i) = \frac{\exp(\vec{w}^{\mathbf{T}}\vec{f}(\vec{x}_i,\vec{y}_i))}{Z(\vec{x})}$$

$$Z(\vec{x}_i) = \sum_{\vec{y}_k} \exp(\vec{w}^{\mathbf{T}}\vec{f}(\vec{x}_i,\vec{y}_k))$$

Minimize the log-likelihood to compute the parameters

$$\frac{\partial \mathcal{L}}{\partial w_k} = 0$$

$$\mathcal{L}(\vec{w}) = -\sum_i \log p(\vec{y}_i|\vec{x}_i) + \lambda \vec{w}^{\mathbf{T}}\vec{w}$$

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

Comparison to conditional random fields (CRF) model

Some **weights** vector which will be optimized

Some vector function of the input and output vectors -> **features**

$$p(\vec{y}_i|\vec{x}_i) = \frac{\exp(\vec{w}^{\mathbf{T}} \vec{f}(\vec{x}_i, \vec{y}_i))}{Z(\vec{x})}$$

$$Z(\vec{x}_i) = \sum_{\vec{y}_k} \exp(\vec{w}^{\mathbf{T}} \vec{f}(\vec{x}_i, \vec{y}_k))$$

Minimize the log-likelihood to compute the parameters

$$\frac{\partial \mathcal{L}}{\partial w_k} = 0$$

$$\mathcal{L}(\vec{w}) = -\sum_i \log p(\vec{y}_i|\vec{x}_i) + \lambda \vec{w}^{\mathbf{T}} \vec{w}$$

CRF can also give **perfect predictions** for the evolution of cellular automata.
However, this only occurs if the **features are chosen correctly**, otherwise the results could be completely random despite a large size of features vector.

The **MPO** model **"finds"** the relevant **features**.

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

Discrete nonlinear maps

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

Discrete nonlinear maps

We consider a nonlinear and beyond nearest-neighbor model for evolution of a probability distribution

$$
\begin{aligned}
P_{l,t+1} = P_{l,t} \; & + g_1/2\left[(P_{l-1,t})^{m_1} + (P_{l+1,t})^{m_1} - 2\,(P_{l,t})^{m_1}\right] \\
& + g_2/2\left[(P_{l-2,t})^{m_2} + (P_{l+2,t})^{m_2} - 2\,(P_{l,t})^{m_2}\right]
\end{aligned}
$$

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

Discrete nonlinear maps

We consider a nonlinear and beyond nearest-neighbor model for evolution of a probability distribution

$$\begin{aligned} P_{l,t+1} = P_{l,t} \; &+ g_1/2\left[(P_{l-1,t})^{m_1} + (P_{l+1,t})^{m_1} - 2\,(P_{l,t})^{m_1}\right] \\ &+ g_2/2\left[(P_{l-2,t})^{m_2} + (P_{l+2,t})^{m_2} - 2\,(P_{l,t})^{m_2}\right] \end{aligned}$$

For training we use random inputs (each site is taken from uniform distribution [0,1] and then the total probability is normalized) and their corresponding outputs.

We then compare to 100 initial conditions chosen as

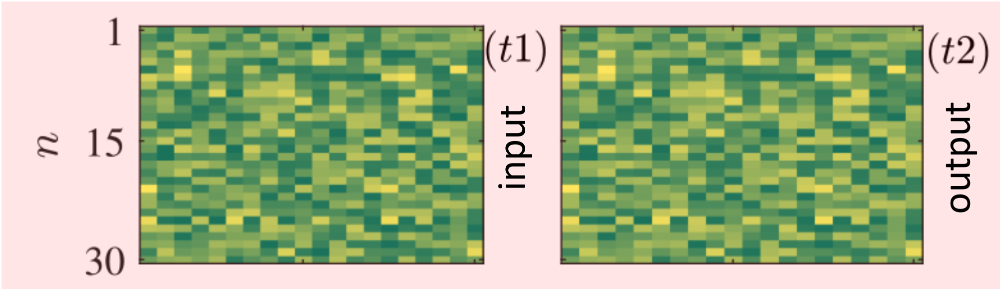
$$P_{l,t=1} = (1 + \cos(2\pi l \lambda/L)) \exp(-((l - l_0)^2)/(2v))/\Gamma$$

where λ, l_0 and v are chosen randomly and Γ is the normalization.

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

$m_1 = 3, g_1 = -0.1, m_2 = 2\,, g_2 = 0.5$

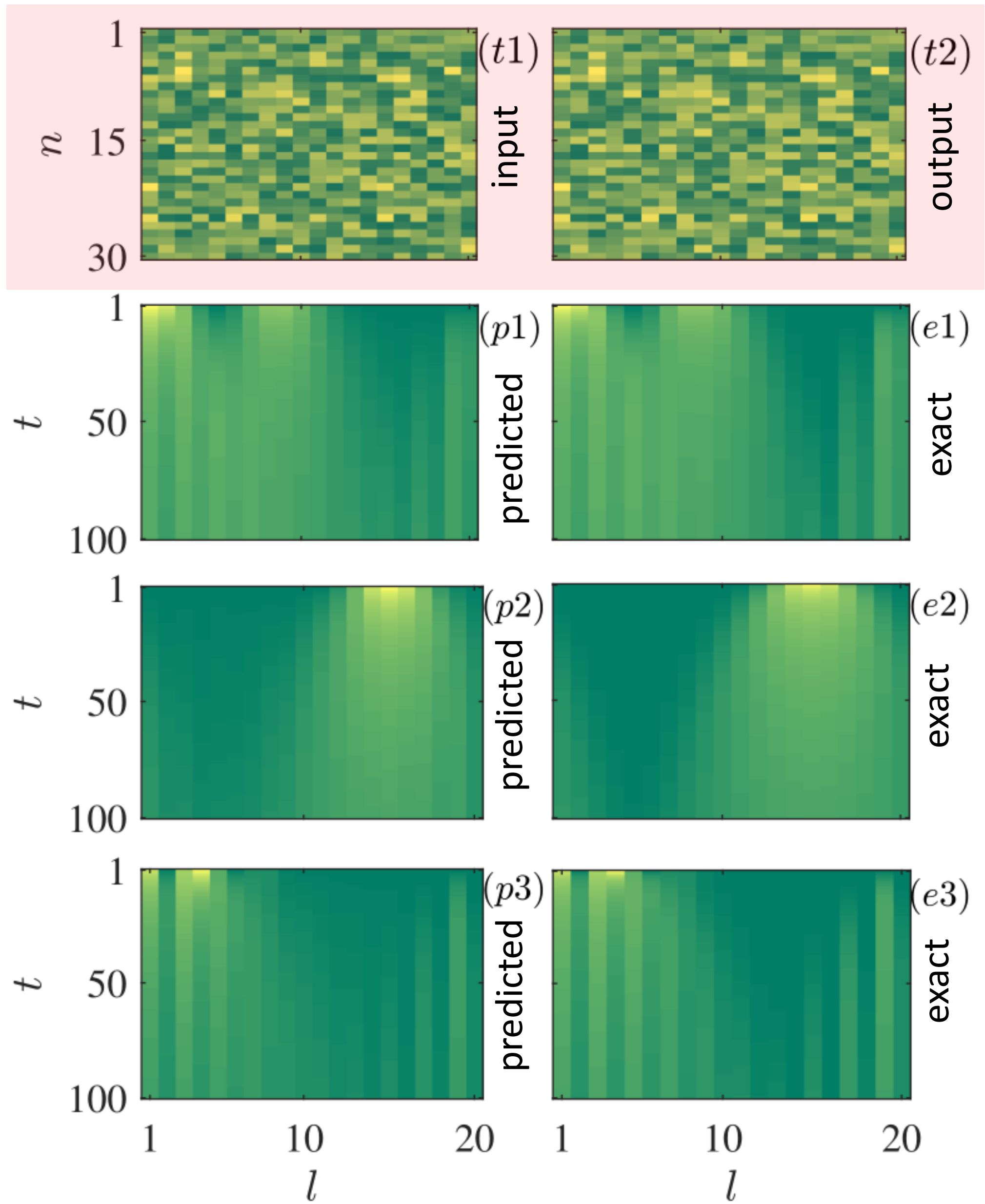


Training data, input and output examples

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

$m_1 = 3, g_1 = -0.1, m_2 = 2\,, g_2 = 0.5$



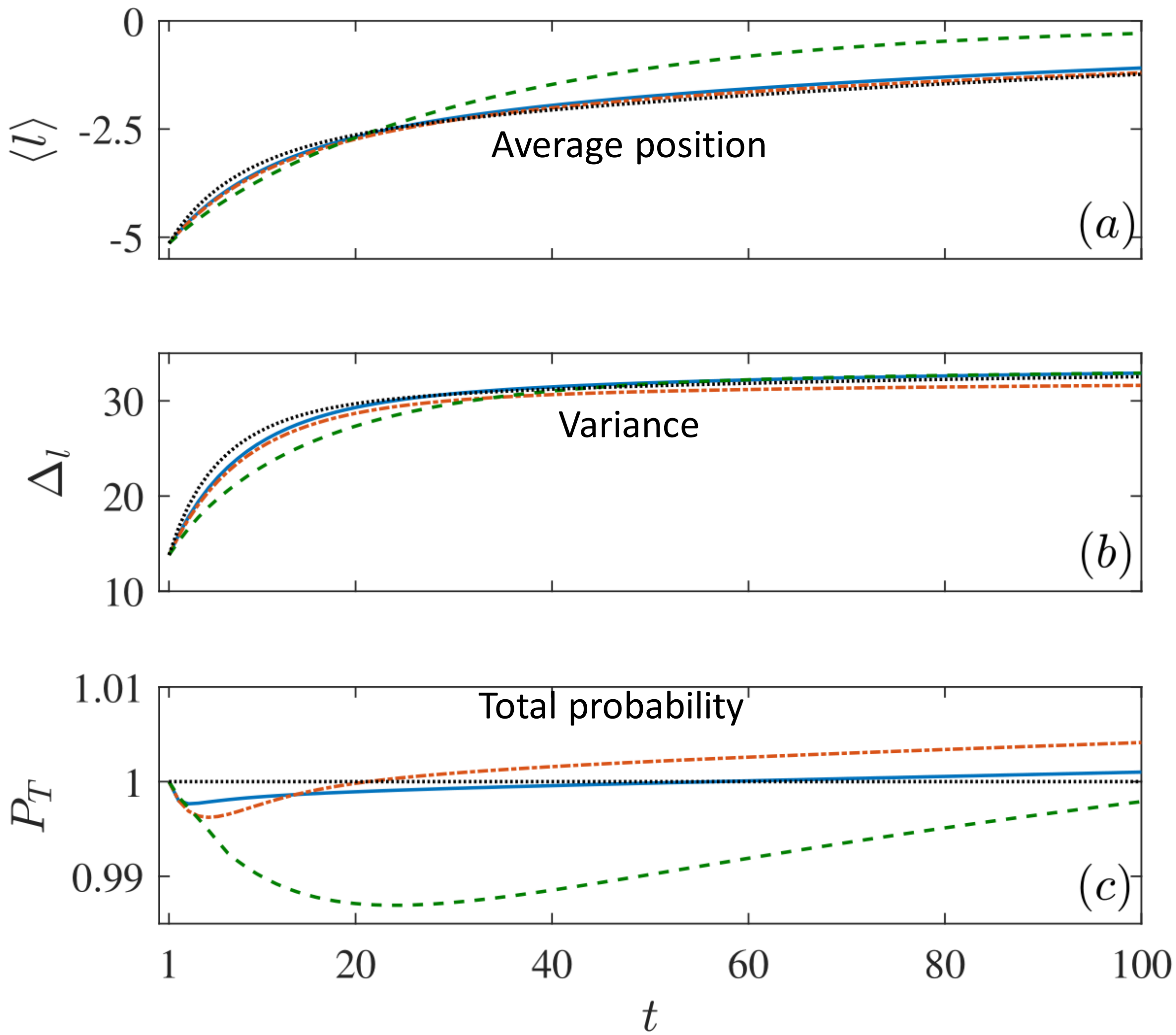
Training data, input and output examples

Example of 3 different initial conditions for L=20, M=20000 and D_W=20.

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

L=20, M=20000.



D_W=5 and $\varepsilon_t = 10^{-5}$

D_W=20 and $\varepsilon_t = 10^{-5}$

D_W=20 and $\varepsilon_t = 10^{-7}$

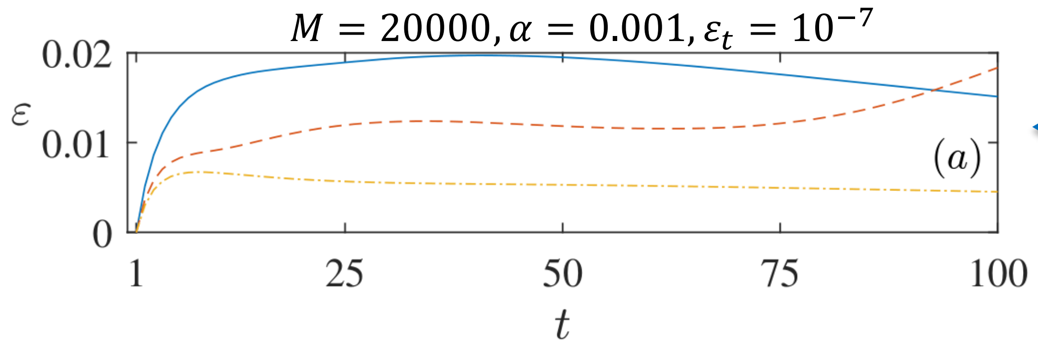
Exact

$m_1 = 3, g_1 = -0.1, m_2 = 2\ , g_2 = 0.5$

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

Comparison of performance for different hyperparameters.
Average of N=100 initial conditions and L=20, M=20000.



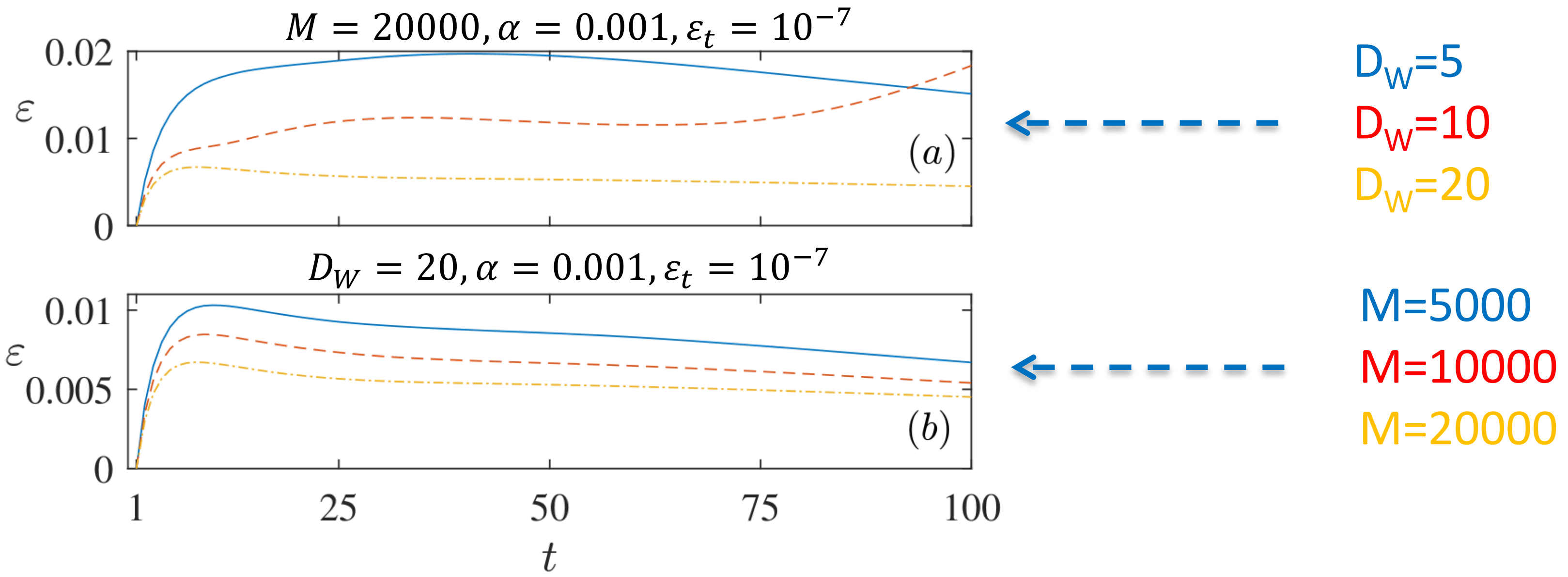
D_W=5
D_W=10
D_W=20

$$m_1 = 3, g_1 = -0.1, m_2 = 2\,, g_2 = 0.5$$

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

Comparison of performance for different hyperparameters.
Average of N=100 initial conditions and L=20, M=20000.

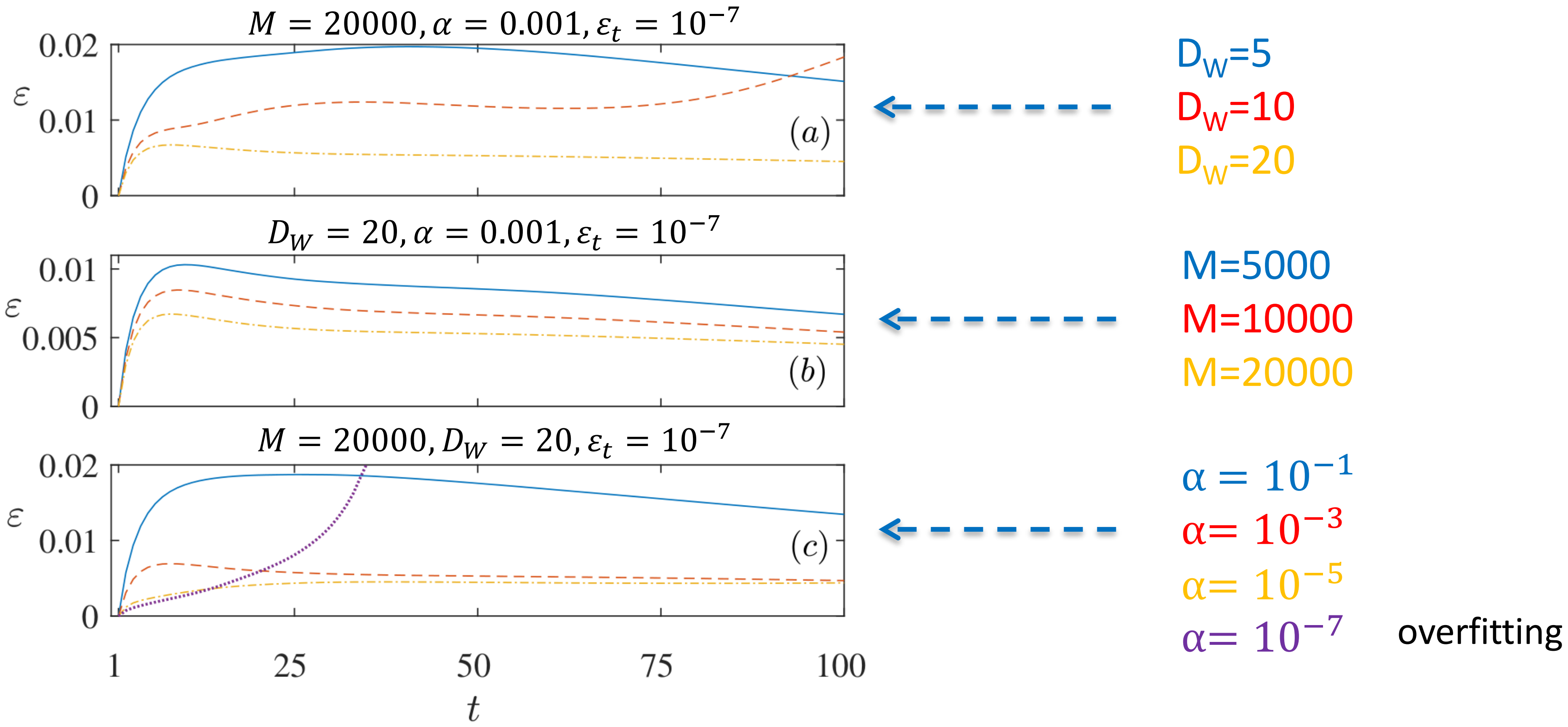


$$m_1 = 3, g_1 = -0.1, m_2 = 2, g_2 = 0.5$$

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

Comparison of performance for different hyperparameters.
Average of N=100 initial conditions and L=20, M=20000.

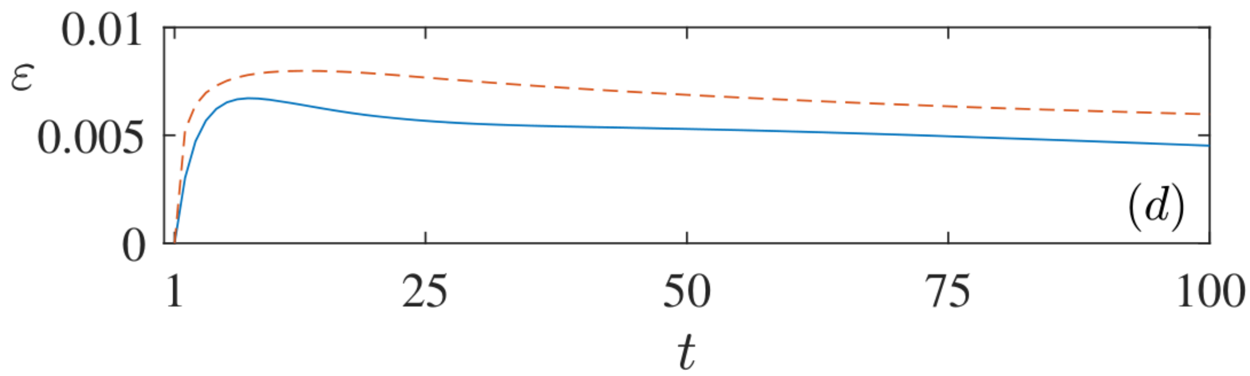


$$m_1 = 3, g_1 = -0.1, m_2 = 2\,, g_2 = 0.5$$

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

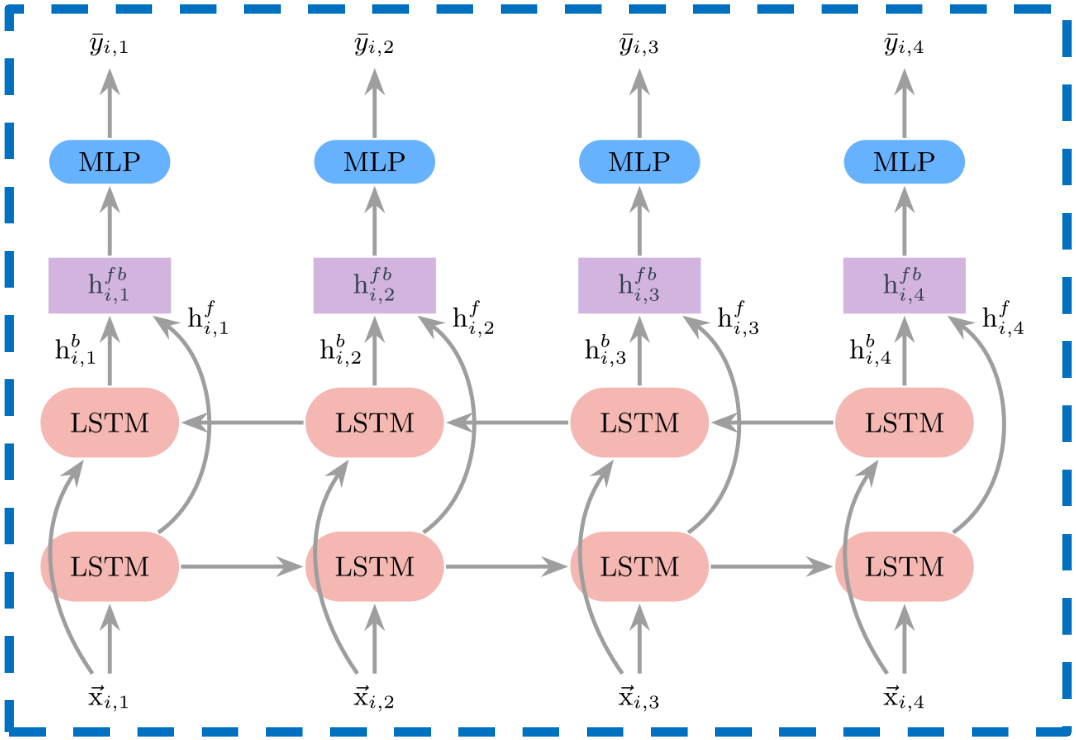
Comparison to LSTM bi-directional neural network



LSTM

MPO D_W=20 and $\varepsilon_t = 10^{-7}$

LSTM bi-directional neural network



$$\vec{i}_{i,l} = \sigma\left(\mathbf{W}_{\mathrm{xi}}\,\vec{x}_{i,l} + \vec{b}_{\mathrm{xi}} + \mathbf{W}_{\mathrm{hi}}\,\vec{h}_{i,l-1} + \vec{b}_{\mathrm{hi}}\right)$$

$$\vec{f}_{i,l} = \sigma(\mathbf{W}_{\mathrm{xf}}\,\vec{x}_{i,l} + \vec{b}_{\mathrm{xf}} + \mathbf{W}_{\mathrm{hf}}\,\vec{h}_{i,l-1} + \vec{b}_{\mathrm{hf}})$$

$$\vec{g}_{i,l} = \tanh\left(\mathbf{W}_{\mathrm{xg}}\,\vec{x}_{i,l} + \vec{b}_{\mathrm{xg}} + \mathbf{W}_{\mathrm{hg}}\,\vec{h}_{i,l-1} + \vec{b}_{\mathrm{hg}}\right)$$

$$\vec{o}_{i,l} = \sigma(\mathbf{W}_{\mathrm{xo}}\,\vec{x}_{i,l} + \vec{b}_{\mathrm{xo}} + \mathbf{W}_{\mathrm{ho}}\,\vec{h}_{i,l-1} + \vec{b}_{\mathrm{ho}})$$

$$\vec{c}_{i,l} = \vec{f}_{i,l} \odot \vec{c}_{i,l-1} + \vec{i}_{i,l} \odot \vec{g}_{i,l}$$

$$\vec{h}_{i,l} = \vec{o}_{i,l} \odot \tanh(\vec{c}_{i,l})$$

C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

Classification tasks: the MPO algorithm can be used for this too.

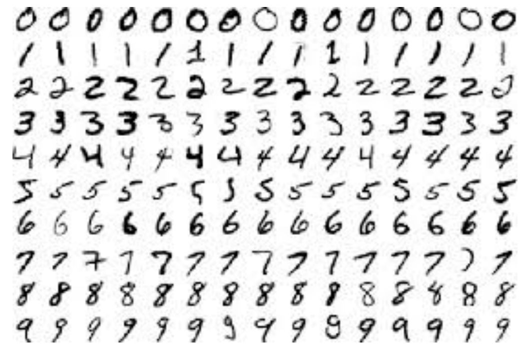
C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

Classification tasks: the MPO algorithm can be used for this too.

Key is to encode the few possible outputs in a way that can perform effectively.

MNIST -> 28 x 28 pixels to 10 digits (training set 60000 images, development 10000)



0 -> **1**000000000000000…
1-> 000000**1**000000000…
2-> 000000000000**1**000…
…

one-hot vectors in which the 1s are evenly spaced

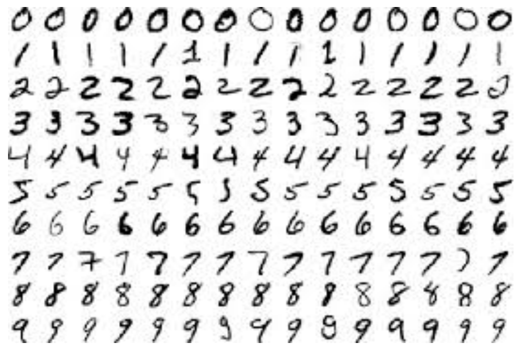
C. Guo, et al. arxiv:1803.10908 (2018)

MPOs for sequence to sequence learning: **performance**

Classification tasks: the MPO algorithm can be used for this too.

Key is to encode the few possible outputs in a way that can perform effectively.

MNIST -> 28 x 28 pixels to 10 digits (training set 60000 images, development 10000)



0 -> 1000000000000000…
1-> 0000001000000000…
2-> 0000000000001000…
…

one-hot vectors in which the 1s are evenly spaced

We obtain:
D_W=10 -> 93.9% accuracy on training set and 94.0% on development set
D_W=20 -> 97.6% accuracy on training set and 97.2% on development set

C. Guo, et al. arxiv:1803.10908 (2018)

Conclusions and outlook

MPS and MPO approaches seem to carry some potential.

The MPO approach we introduced:

- Allows to perform sequence to sequence prediction
  - Cellular automata
  - Discrete nonlinear maps
- Seems to be comparable with some state-of-the-art ML models for different tasks
  - It requires less previous knowledge than CRF
  - It can perform faster than LSTM bi-directional NN
- Can be used for classification

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

Thank you!

Financial support

Ministry of Education SINGAPORE

SUTD-MIT INTERNATIONAL DESIGN CENTRE (IDC)

ASIAN OFFICE OF AEROSPACE R&D

Conclusions and outlook

MPS and MPO approaches seem to carry some potential.

The MPO approach we introduced:

- Allows to perform sequence to sequence prediction
  - Cellular automata
  - Discrete nonlinear maps
- Seems to be comparable with some state-of-the-art ML models for different tasks
  - It requires less previous knowledge than CRF
  - It can perform faster than LSTM bi-directional NN
- Can be used for classification

Outlook:

- Study of texts -> part-of-speech tagging, entity recognition
  inherently 1D problems
- Extend to sequences of variable and/or infinite length
- Probabilistic inputs and outputs
- Various optimizations
- Interpretation of machine learning

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

Thank you!

Financial support

Ministry of Education SINGAPORE

SUTD-MIT INTERNATIONAL DESIGN CENTRE (IDC)

ASIAN OFFICE OF AEROSPACE R&D