RIKEN-AIP & PRAIRIE Joint Workshop

Efficient Machine Learning with Tensor Networks

Qibin Zhao

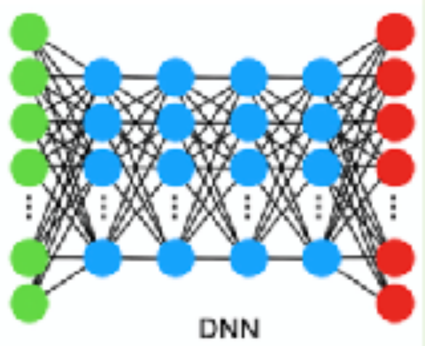
Tensor Learning Team
RIKEN AIP
https://qibinzhao.github.io

Mar 21, 2023

Trends of Machine Learning

Big Data

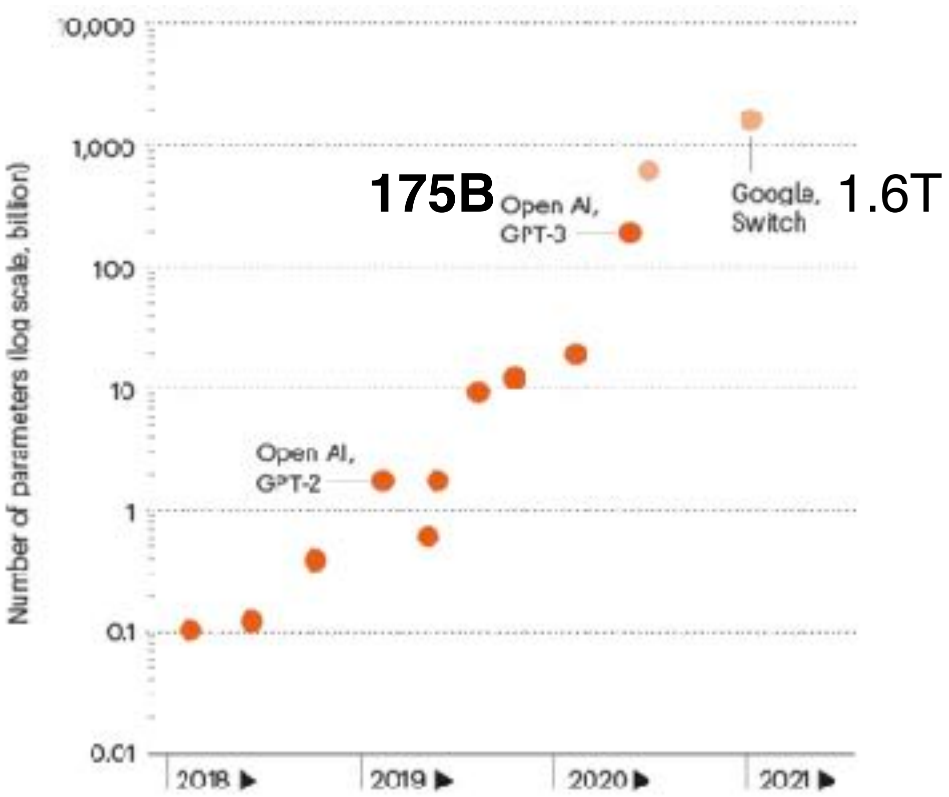
Large Model



Computation

OpenAI's GPT-3

Dataset: 45 TB text data



175B

1.6T

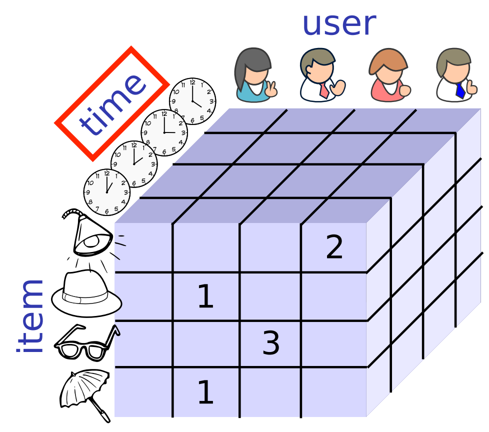
https://www.nature.com/articles/d41586-021-00530-0

OpenAI's GPT-3

- 28 TFLOPS V100
- 355 GPU years
- $4.6 M

Challenges from data perspective

- Learning knowledge from incomplete & limited data, noisy data, or adversarial corrupted data



3 order tensor

Recommender system

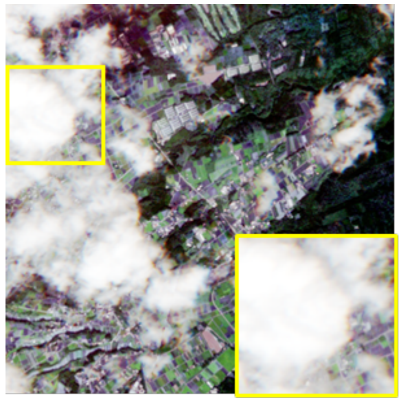
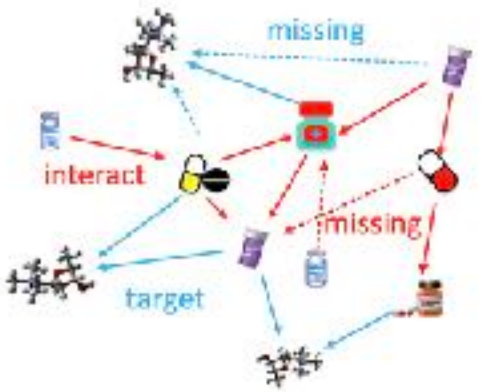
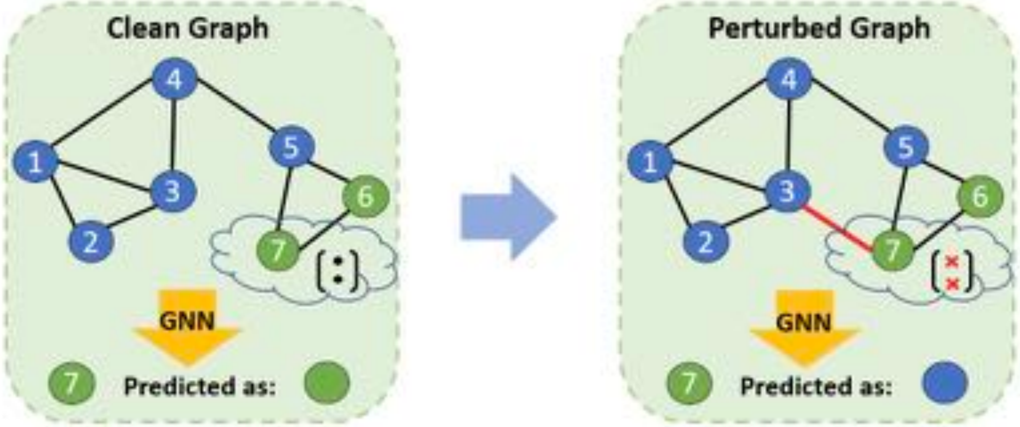


Image inpainting/denoising



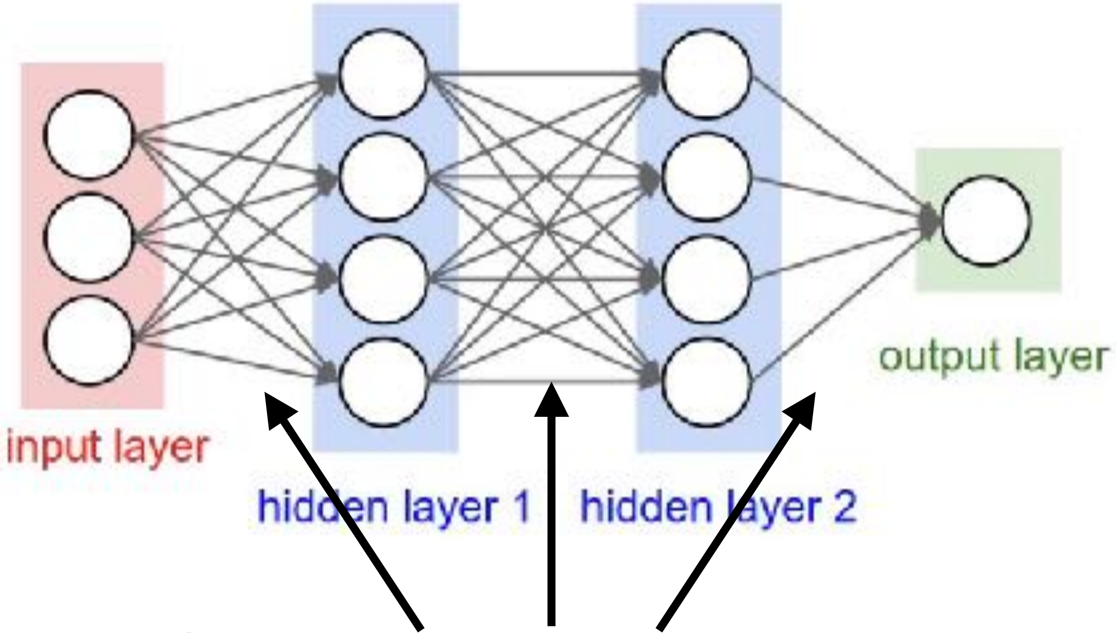
graph prediction



[Jin et al. SIGKDD 2021]

Poisoning or adversarial attack

Challenges from model perspective



Over-parameterization

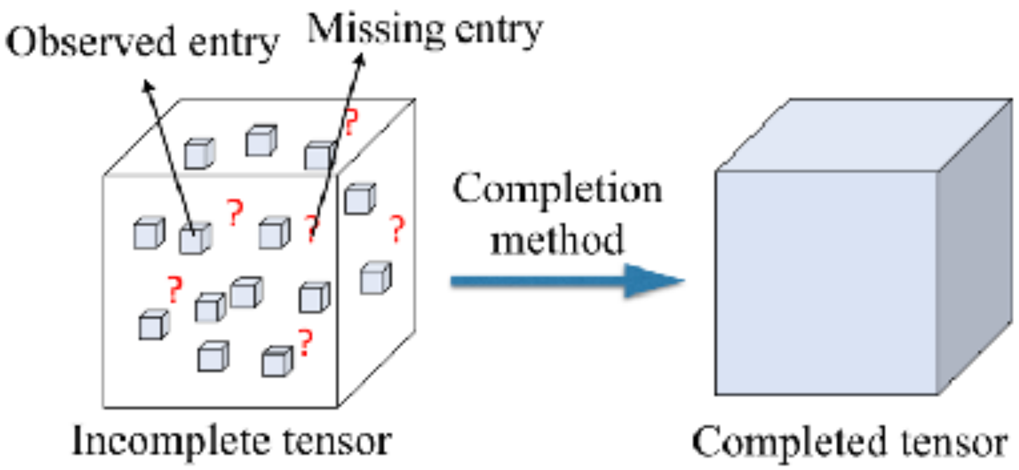
- Complex architecture, large number of parameters, heavy computation for training and inference.

- Lack of interpretability and lack of robustness to adversarial attacks.

- How to dramatically increase model capacity without significant increasing of model size?

Multi-dimensional, Incomplete and Noisy Data

- Task: learning from limited tensor entries to predict unobserved entries



$$\mathcal{Y}_{\Omega} \rightarrow \mathcal{Y}_{\bar{\Omega}}$$

- Challenges:
  - Data efficiency
  - Scalability and efficient optimization algorithms
  - Exact recovery guarantee

Tensor Completion

Objective:

$$\min_{\mathcal{X}} \|\Omega * (\mathcal{Y} - \mathcal{X})\| + R(\mathcal{X})$$

Fitting error Structure Regularizer

Popular approaches:

- Low-rankness assumption (convex, not scalable)

$$R(\mathcal{X}) = \|\mathcal{X}\|_*$$

- Decomposition based approach (optimal rank selection)

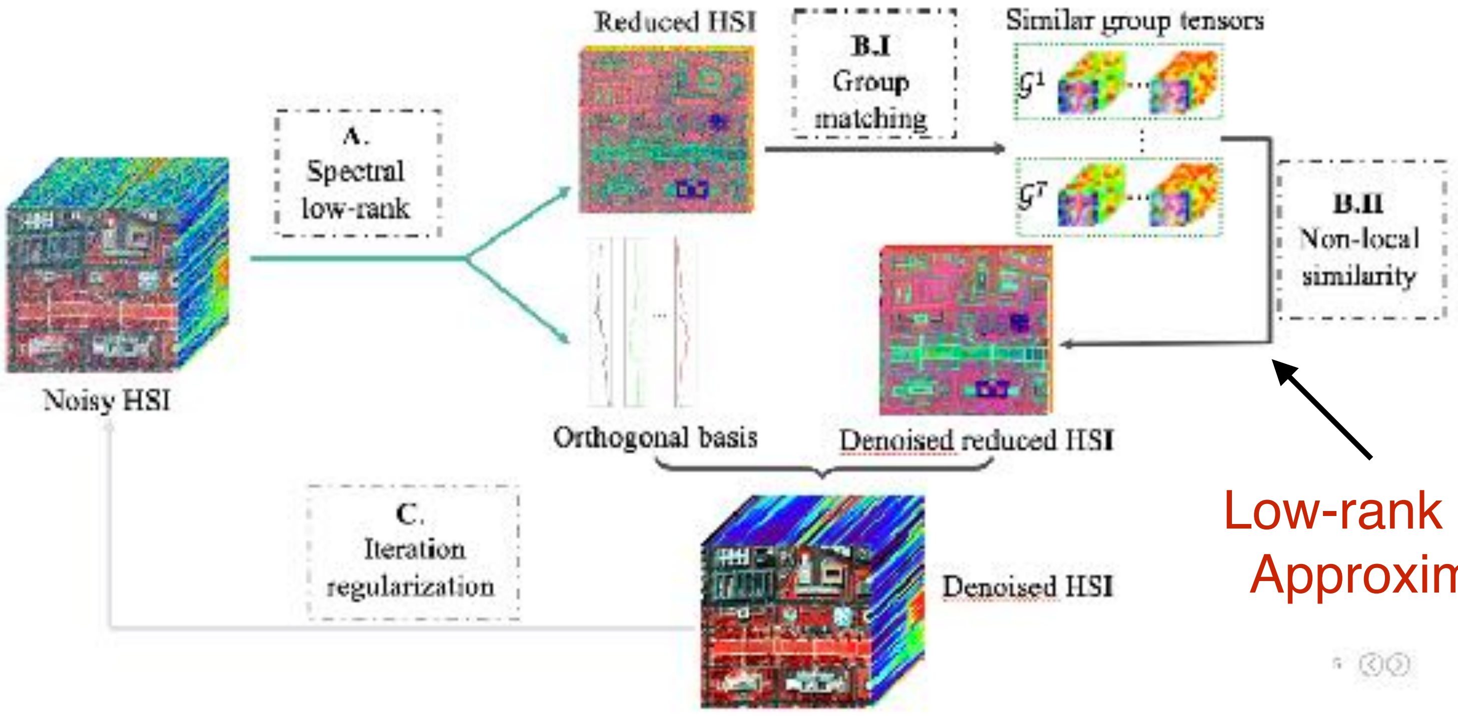
$$R(\mathcal{X}) = \|\mathcal{X} - \mathrm{TN}(\mathcal{G}_1, \ldots, \mathcal{G}_N)\|$$

- Prior knowledge (smoothness, non-negative), side information

Low-rankness under Linear Transformation

- **Image Denoising**: large scale image is not globally low-rank

(He et al., CVPR 2019)



Low-rank Tensor Approximation

(Li et al, CVPR 2019)

- **Non-uniform missing patterns** (slice, fiber missing)

$$\min_{\mathbf{X}\in\mathbb{R}^{m_1\times m_2}} \|\mathcal{Q}(\mathbf{X})\|_* \quad s.t. \|\mathcal{P}_\Omega(\mathbf{X})-\mathcal{P}_\Omega(\mathbf{Y})\|_F \leq \delta,$$

Linear transformation

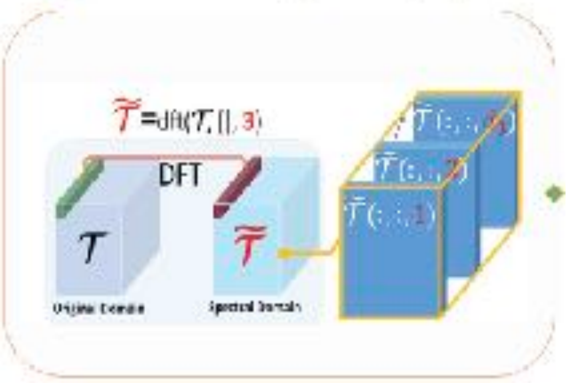
Error bound is theoretically guaranteed

Enhanced low-rank modeling for tensor SVD

(A. Wang et al., AAAI 2020)

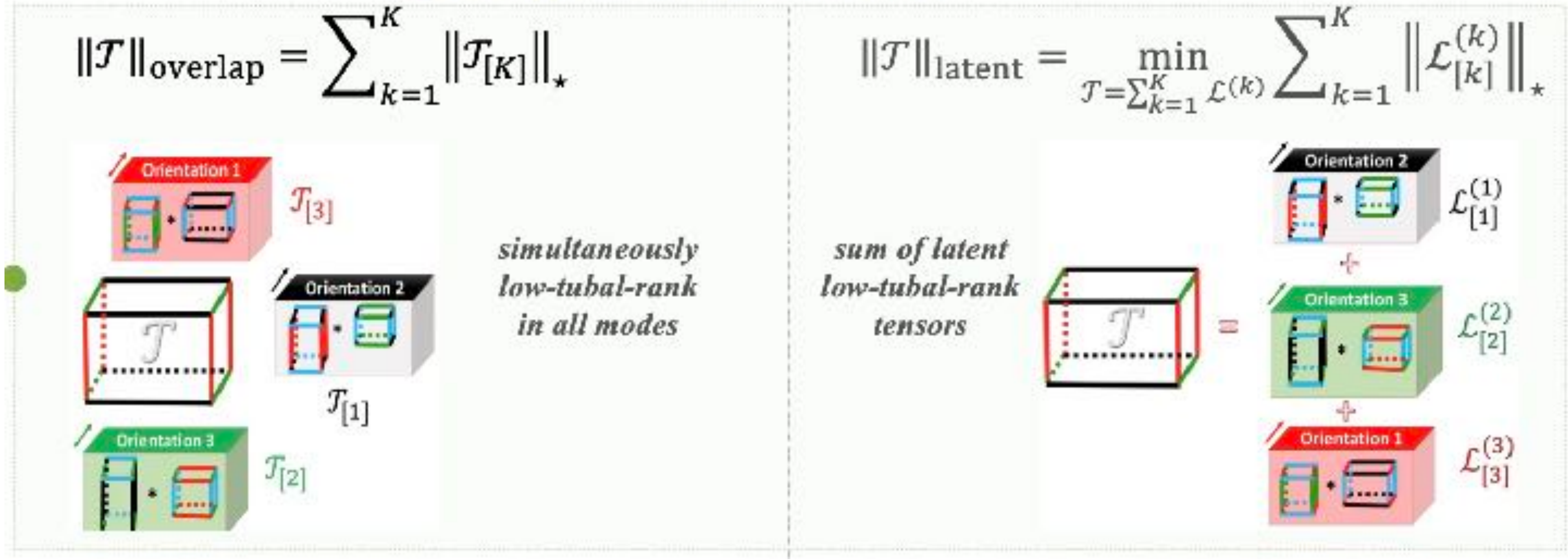
- **Problem:** t-SVD has mode sensitivity.
- Two **mode invariant tubal nuclear norms** with error bound

Mode sensitivity of TNN $\|\mathcal{T}\|_*$



$\bar{\mathcal{T}}$=dft($\mathcal{T}$,[],3)
DFT
$\mathcal{T}$
$\bar{\mathcal{T}}$
Original Domain
Spectral Domain
$\bar{\mathcal{T}}(:,:,1)$
$\bar{\mathcal{T}}(:,:,2)$
$\bar{\mathcal{T}}(:,:,3)$

Two mode invariant TNNs



$$\|\mathcal{T}\|_{\text{overlap}} = \sum_{k=1}^{K} \|\mathcal{T}_{[K]}\|_*$$

$$\|\mathcal{T}\|_{\text{latent}} = \min_{\mathcal{T}=\sum_{k=1}^{K}\mathcal{L}^{(k)}} \sum_{k=1}^{K} \left\|\mathcal{L}^{(k)}_{[k]}\right\|_*$$

$$\frac{\left\|\mathcal{L}^* - \hat{\mathcal{L}}_{\text{overlap}}\right\|_F^2}{d^K} \leq C_1\sigma^2\left(\|\mathcal{S}^*\|_0 K\log d + d^{-1}K^{-2}\sum_k r_t(\mathcal{L}^*_{[k]})\right)$$

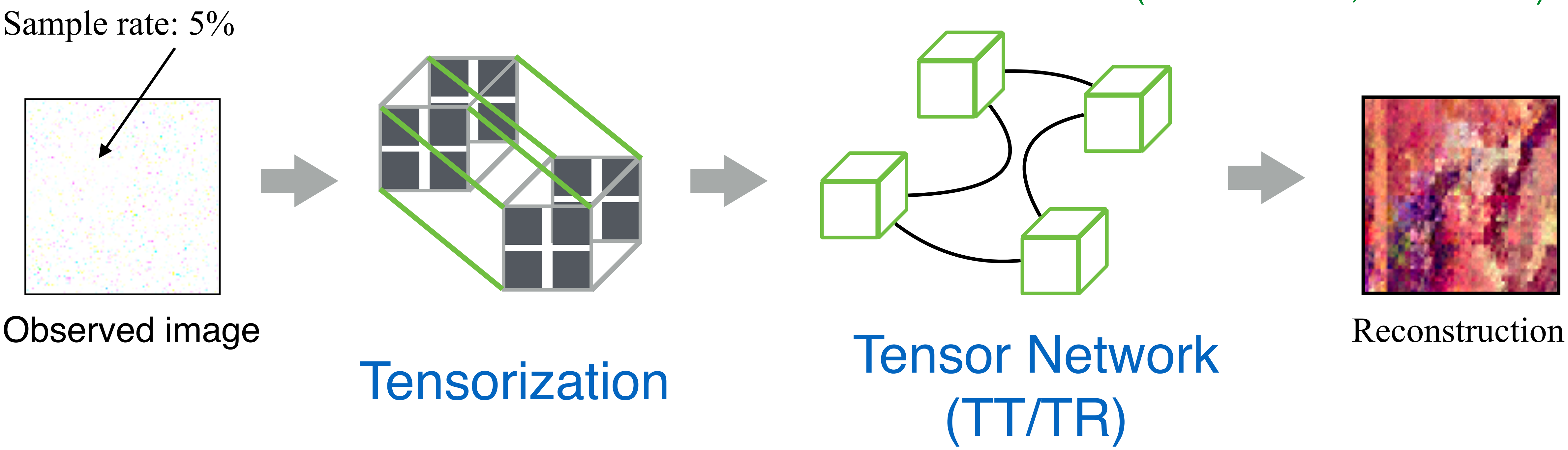
*error bounded in **sum of tubal ranks** in all modes*

$$\frac{\left\|\mathcal{L}^* - \hat{\mathcal{L}}_{\text{latent}}\right\|_F^2}{d^K} \leq C_2\sigma^2\left(\|\mathcal{S}^*\|_0 K\log d + d^{-1}\min_k r_t(\mathcal{L}^*_{[k]})\right)$$

*error bounded by mode of **minimal tubal rank***

Tensor Networks with Low-rank Cores

(L. Yuan et al., AAAI 2019)



Fitting error

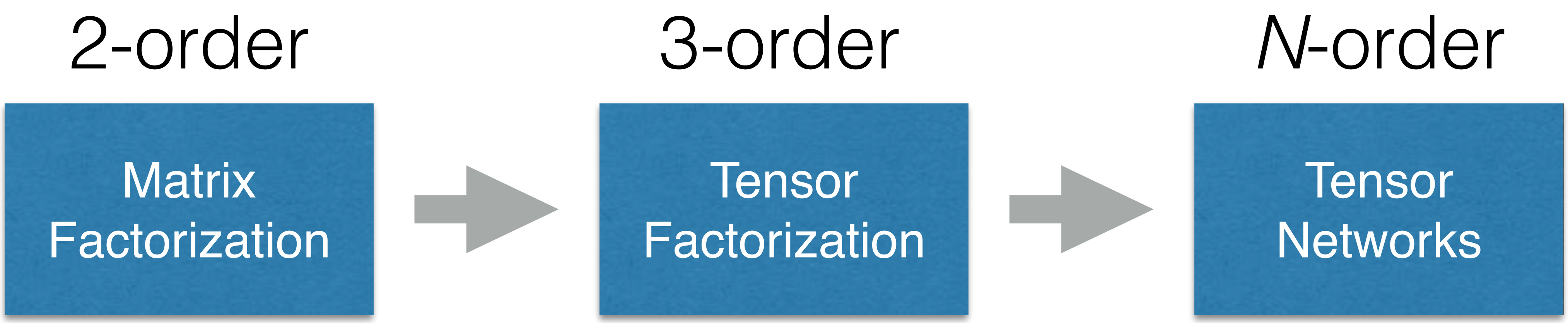
Nuclear norm on core tensor

TT/TR decomposition

$$\min_{\mathcal{G}} \left\|\Omega * (\mathcal{Y}-\hat{\mathcal{Y}})\right\|_F^2 + \lambda \sum_{n=1}^{d}\sum_{i=1}^{3}\left\|\mathbf{G}_{(i)}^{(n)}\right\|_*, \quad s.t. \quad \hat{\mathcal{Y}} = \mathrm{TR}(\mathcal{G}^{(1)},\cdots,\mathcal{G}^{(d)}).$$

- Tensorization allows for capturing complex structural dependency
- Efficient optimization by combining decomposition and nuclear norm minimization

What is Tensor Network?



2-order

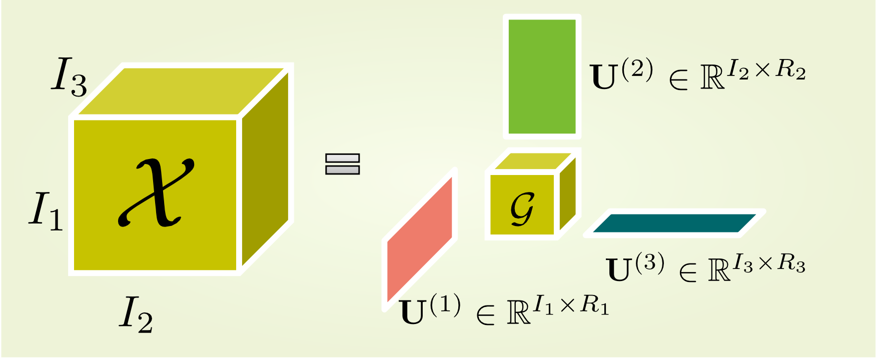
Matrix Factorization

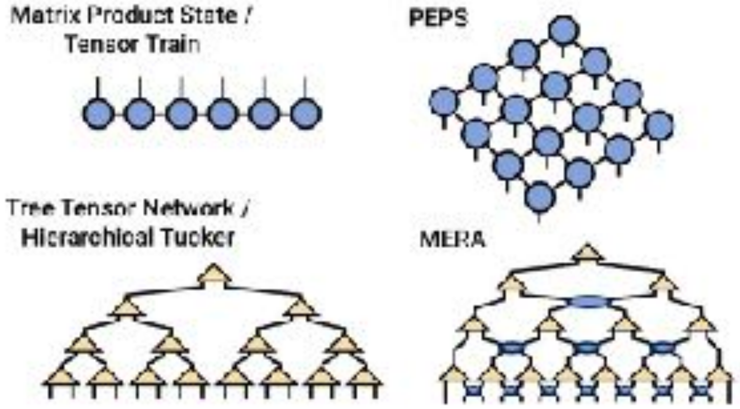
3-order

Tensor Factorization

N-order

Tensor Networks


I_3
I_1
$\mathcal{X}$
I_2
$=$
$\mathbf{U}^{(2)} \in \mathbb{R}^{I_2 \times R_2}$
$\mathcal{G}$
$\mathbf{U}^{(3)} \in \mathbb{R}^{I_3 \times R_3}$
$\mathbf{U}^{(1)} \in \mathbb{R}^{I_1 \times R_1}$


Matrix Product State / Tensor Train
PEPS
Tree Tensor Network / Hierarchical Tucker
MERA

https://tensornetwork.org

- Representation of *N*-order tensor as contractions of O(*N*) smaller tensors
- Physics: to describe entangled quantum many-body systems

Foundations and Trends® in Machine Learning [illegible]

Tensor Networks for Dimensionality Reduction and Large-scale Optimization

Part 1 Low-Rank Tensor Decompositions

Andrzej Cichocki, Namgil Lee, Ivan Oseledets, Anh-Huy Phan, Qibin Zhao and Danilo P. Mandic

now

Foundations and Trends® in Machine Learning [illegible]

Tensor Networks for Dimensionality Reduction and Large-scale Optimization

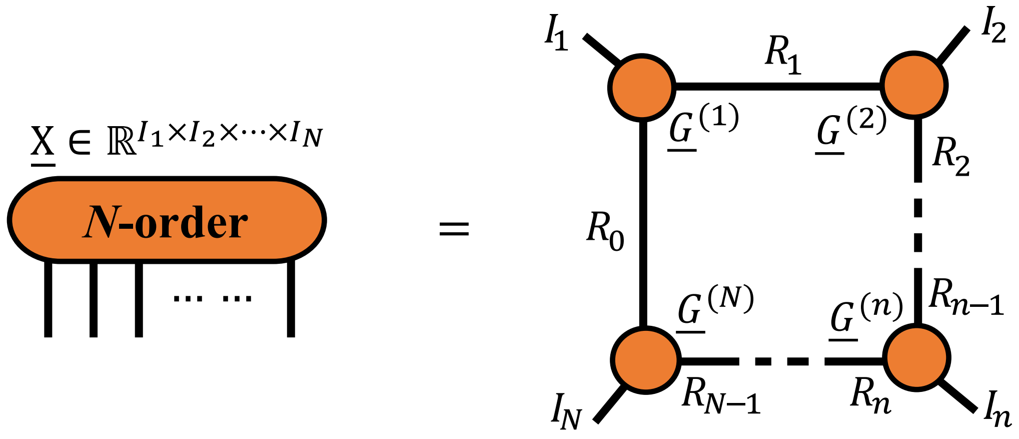
Part 2 Applications and Future Perspectives

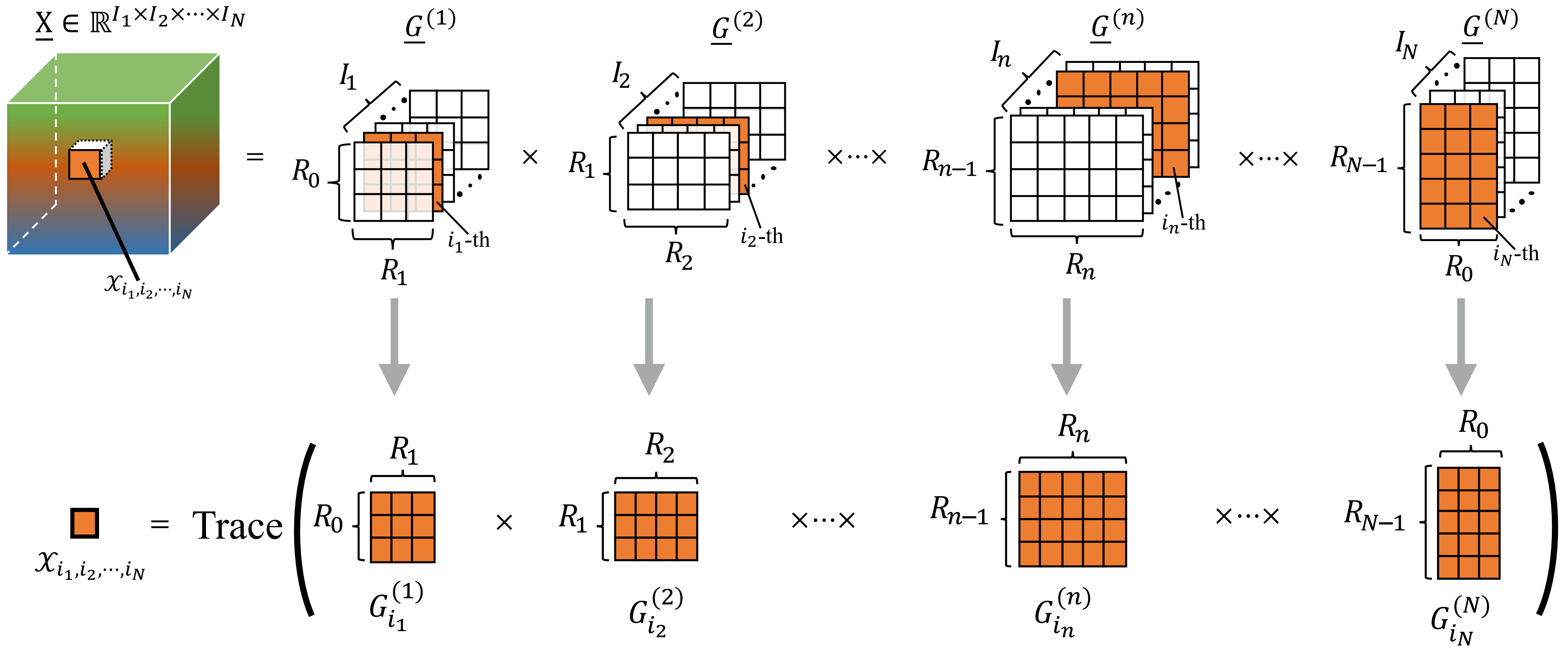
Andrzej Cichocki, Anh-Huy Phan, Qibin Zhao, Namgil Lee, Ivan Oseledets, Masashi Sugiyama and Danilo P. Mandic

now

Tensor Ring Decomposition

(Zhao et al., arXiv 2016, ICASSP 2019)



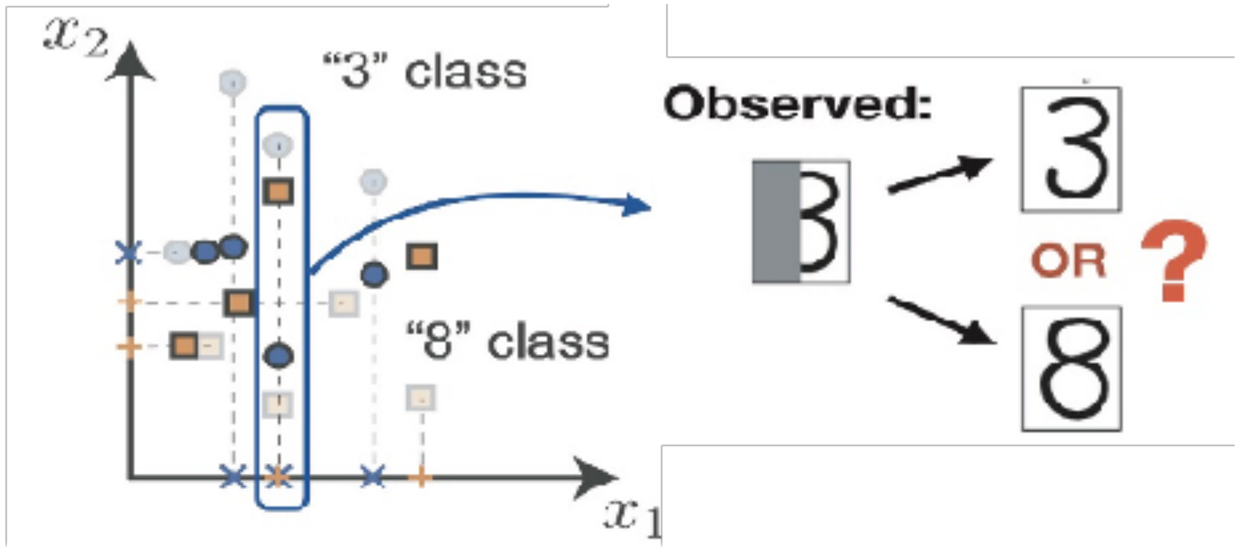


Classification of incomplete data

Problem: learning classification model from incomplete data $(x_n^{miss}, y_n), n = 1, \ldots, N$

Objective: $\hat{f}(\boxed{g(x^{miss})}, \hat{\theta}) \approx f(x, \theta)$

Reconstruction of incomplete data



Sequential approach (completion + classification)

- Cannot ensure statistical consistence of classifier
- Exact recovery is not guaranteed because label information is ignored

Simultaneous reconstruction and classification

(Caiafa et al., CVPR workshop 2021)

- Learning sparse representation and classifier collaboratively (NNs + sparse coding)

$$J(\Theta, \mathbf{D}, \mathbf{s}_i) = \frac{1}{I}\sum_{i=1}^{I}\{J_0(\Theta, \hat{\mathbf{x}}_i, y_i) + \lambda_1 J_1(\mathbf{D}, \mathbf{s}_i) + \lambda_2 J_2(\mathbf{s}_i)\}$$

Classification loss (e.g. crossentropy) for any classifier (deep network)

Representation error

$$J_1(\mathbf{D}, \mathbf{s}_i) = \frac{M}{N}\|\mathbf{m}_i * (\mathbf{x}_i - \mathbf{D}\mathbf{s}_i)\|^2,$$

Promotes sparsity

$$J_2(\mathbf{s}_i) = \frac{1}{N}\|\mathbf{s}_i\|_1$$

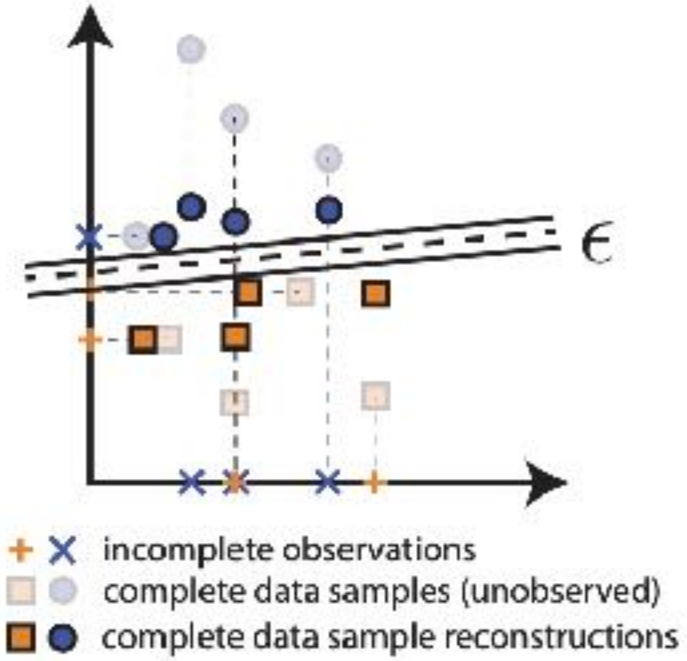
- Sufficient condition

Weights of classifier

$$\epsilon > |\langle \mathbf{w}^m, \mathbf{x}^m \rangle| + |\langle \mathbf{w}^m, \hat{\mathbf{x}}^m \rangle|$$

Original data (missing part)

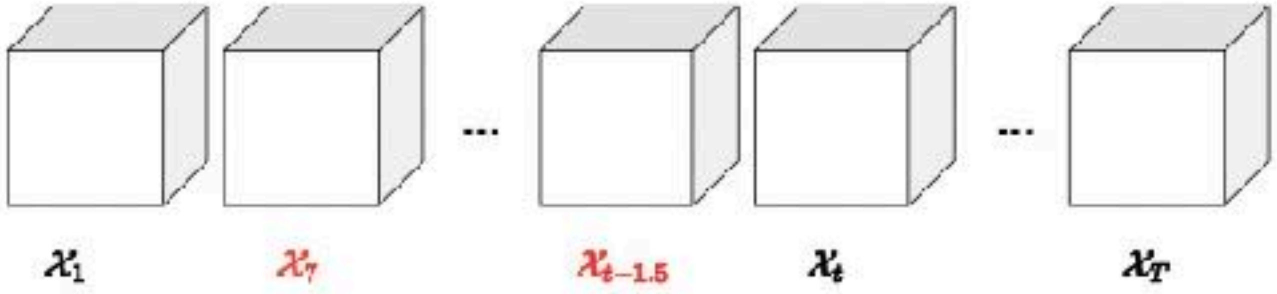
Reconstructed data (Missing part)


ϵ
+ × incomplete observations
□ ○ complete data samples (unobserved)
■ ● complete data sample reconstructions

Time series data with missing time points

Task: Given tensorial time series with **irregular/missing time steps**, to train a model for prediction on **continuous time points**.

Examples: videos with missing frames, relations between stock market prices of many companies, etc



Challenges:

- **Tensorial NN/RNN** (Bai et al. 2017): Incapable of handling irregular time steps, and prediction on decimal time points
- **Neural ODE** (Chen et al. NeurIPS 2018): Ignoring spatial structure information, large number of parameters

Tensor Neural ODE

(Bai et al., IJCNN 2021)

We directly process the tensorial time series $\{\mathcal{Y}_t\}_{t\in[0,T]}, \mathcal{Y}_t \in \mathbb{R}^{I_1\times\cdots\times I_N}$, proposing tensor neural ODE (TENODE)

$$\frac{\mathrm{d}\mathcal{Y}(t)}{\mathrm{d}t} = f_{\Theta}(\mathcal{Y}(t), \mathcal{X}(t), t)$$

with the control input $\mathcal{X}(t)$ and the initial condition $\mathcal{Y}(0) = \mathcal{Y}_0$. Parameter size: from $O(I^{2N})$ of neural ODE to $O(NI^2)$

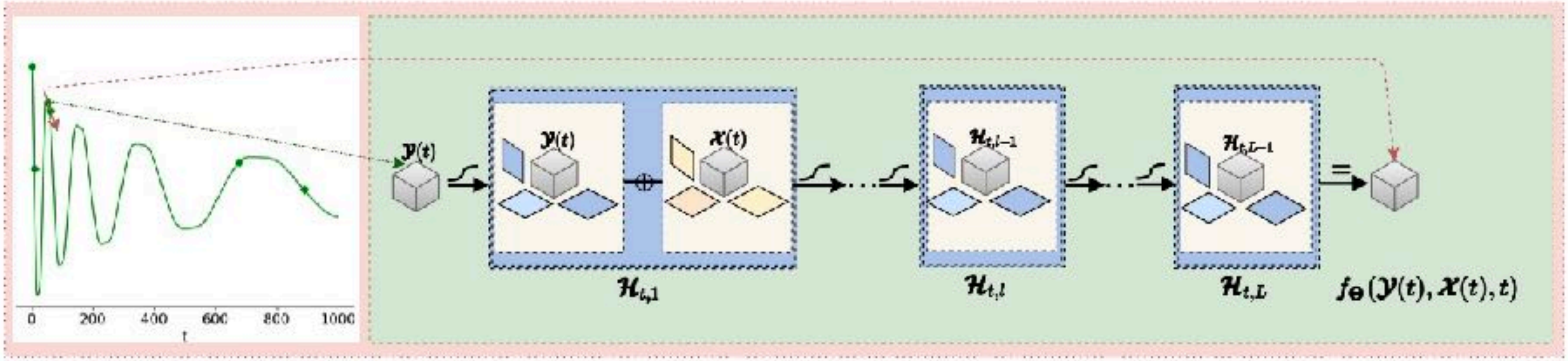
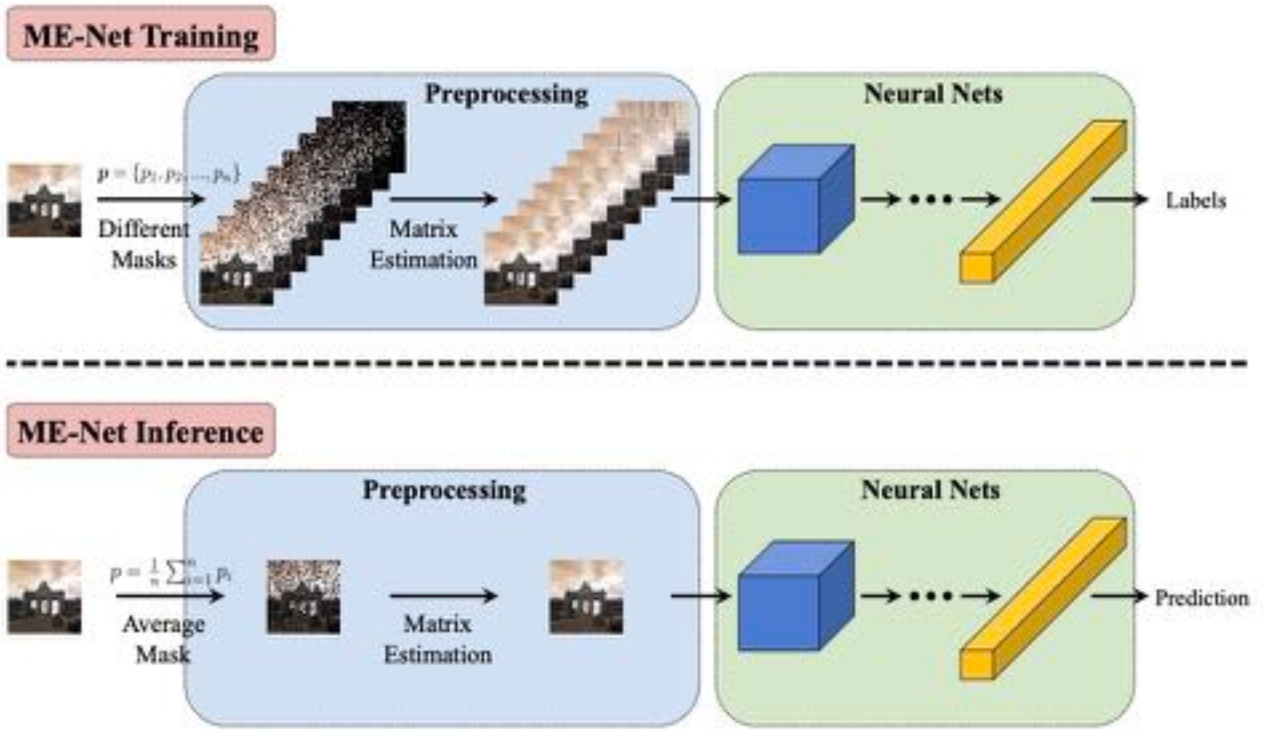


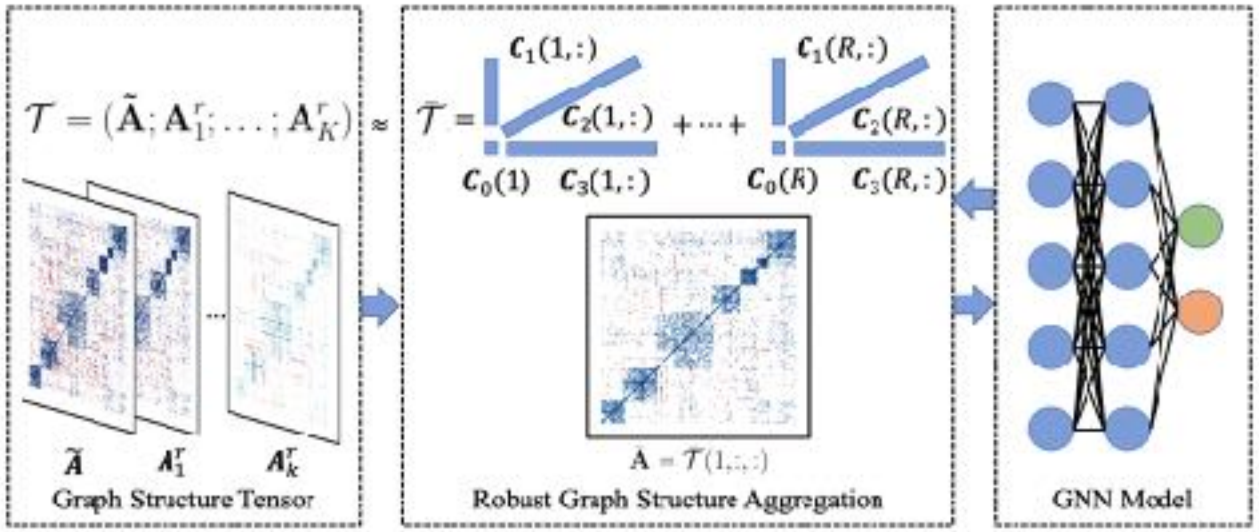
Figure 5: Architecture Overview: Tensor neural ODE (TENODE)

Removing adversarial perturbations from data

- Tensor completion can destroy adversarial perturbations [Yang et al. ICML 2019]

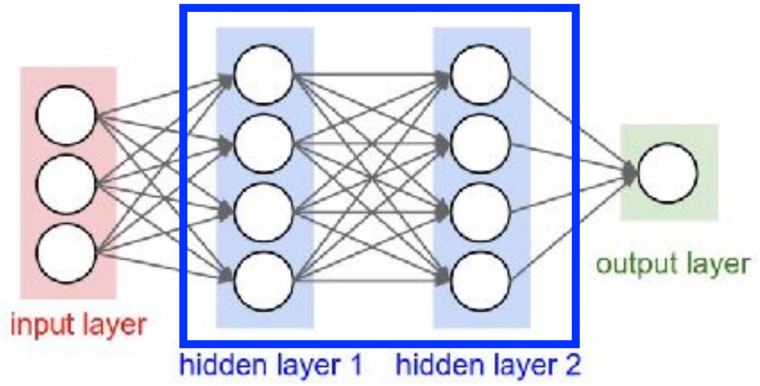


- Defending GNNs via tensor-based robust graph aggregation



Parameter efficient modeling via Tensor Networks

Model Compression



Hidden layer

$$\boldsymbol{h}_2 = \sigma(\underset{(N \times M)}{\boldsymbol{W}^T} \boldsymbol{h}_1 + \boldsymbol{b})$$

Hidden layer

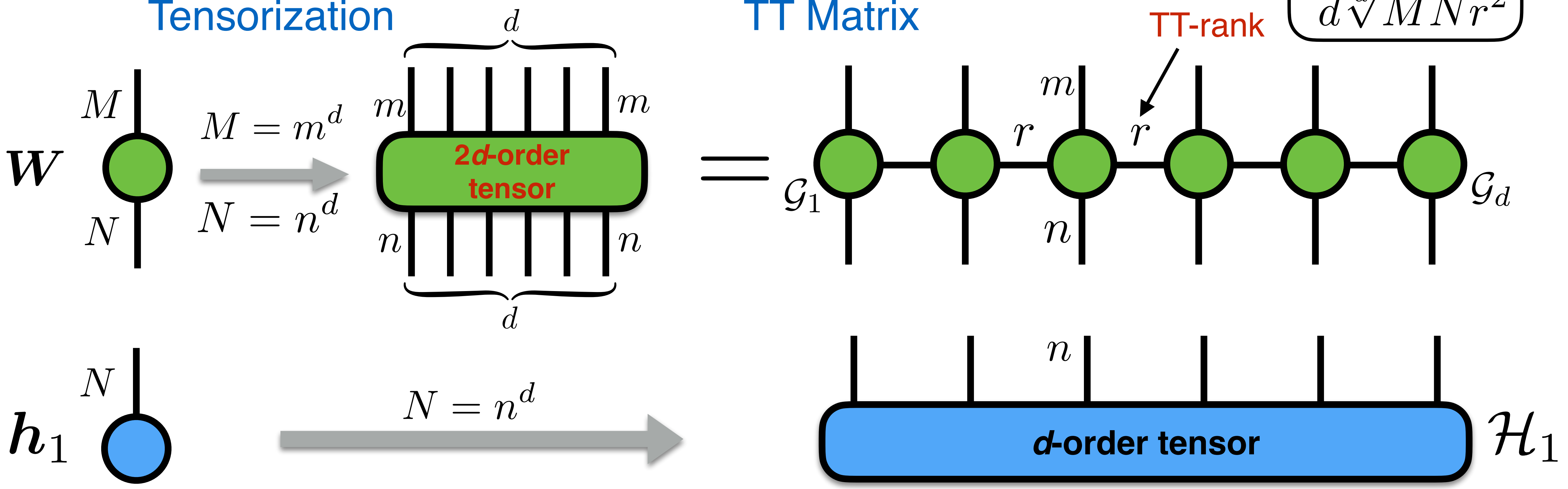
Weights

Number of parameters $d\sqrt[d]{MN}r^2$

Tensorization

TT Matrix

TT-rank



$$\boldsymbol{W}^T \boldsymbol{h}_1 \quad = \quad \mathrm{TT}(\mathcal{G}_1, \cdots, \mathcal{G}_d) \times_{1,\ldots,d} \mathcal{H}_1$$

[Novikov et al., NeurIPS 2015]

Higher-order latent factor analysis

(Tao et al., ACML 2021)

$$\boldsymbol{y} = \boldsymbol{W}\boldsymbol{\eta} + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}),$$

- Given higher-order data $\mathcal{Y} \in \mathbb{R}^{P_1 \times \cdots \times P_D}$, marginalize η gives $\mathcal{Y} \sim \mathcal{N}(0, \mathcal{V})$

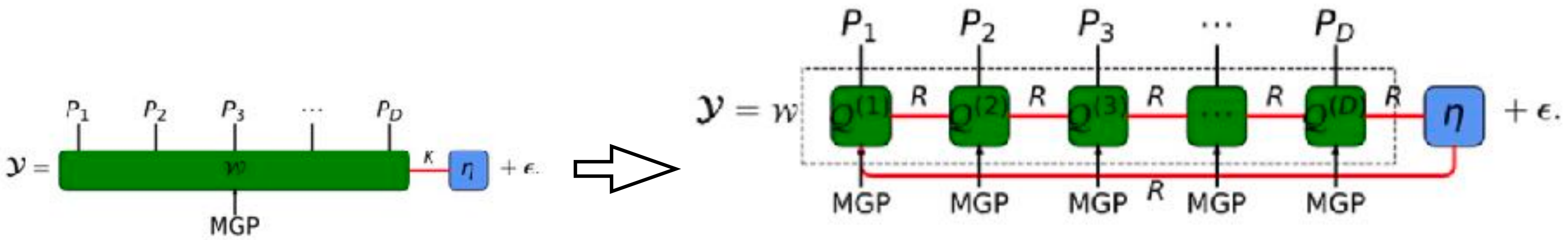
Covariance of vectors: $\boldsymbol{V}_{ij} = \mathrm{cov}(\boldsymbol{y}_i, \boldsymbol{y}_j)$.

Covariance of tensors: $\mathcal{V}_{i_1 i_2 i_3 j_1 j_2 j_3} = \mathrm{cov}(\mathcal{Y}_{i_1 i_2 i_3}, \mathcal{Y}_{j_1 j_2 j_3})$.

Core tensors

$$\mathcal{V}_{p_1 \cdots p_D p'_1 \cdots p'_D} = \underbrace{\mathrm{tr}(\boldsymbol{Q}^{(1)}[p_1] \cdots \boldsymbol{Q}^{(D)}[p_D](\boldsymbol{Q}^{(D)}[p'_D])^\intercal \cdots (\boldsymbol{Q}^{(1)}[p'_1])^\intercal)}_{\text{low-rank TR}} + \underbrace{\tau^{-1}}_{\text{noise}},$$

- TN representation of parameter W



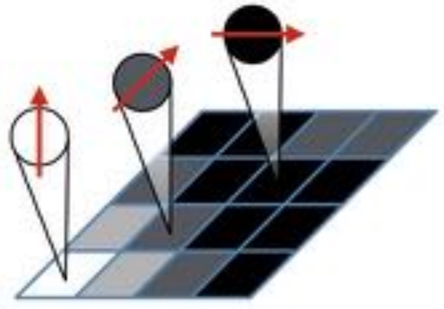
$$\mathcal{Y} = \ll \mathcal{Q}^{(1)}, \ldots, \mathcal{Q}^{(D)}, \boldsymbol{\eta} \gg + \mathcal{E},$$

TN representation of inputs

- Mapping input data into TN representation

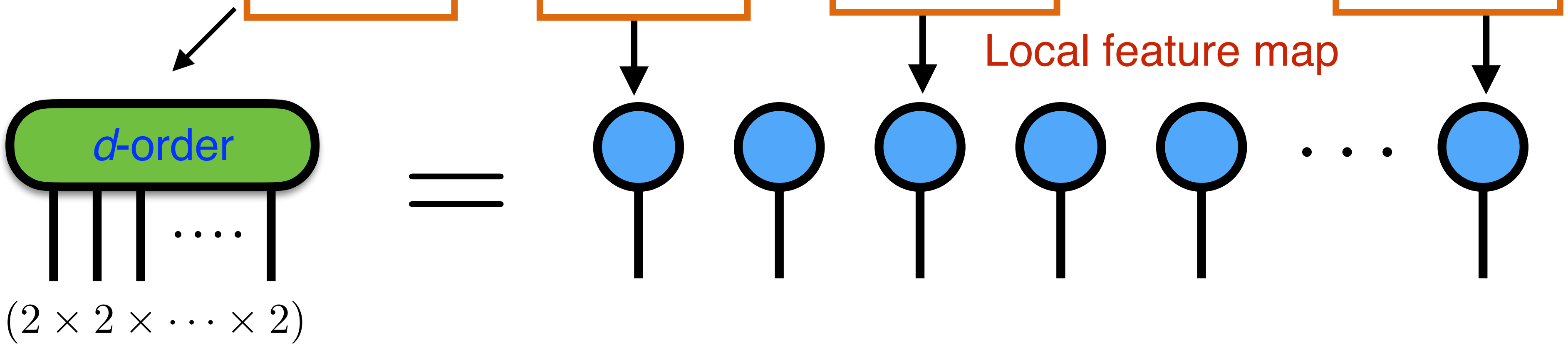
$$\mathbf{x} = [x_1, x_2, \ldots, x_d]^T$$

Inspired by “spin” vectors in quantum system



$$\phi(x_i) = \left[\cos\left(\frac{\pi}{2}x_i\right), \sin\left(\frac{\pi}{2}x_i\right)\right]^T$$

$$\boxed{\Phi(\mathbf{x})} = \boxed{\phi(x_1)} \otimes \boxed{\phi(x_2)} \otimes \cdots \otimes \boxed{\phi(x_d)}$$



Rank-1 Tensor

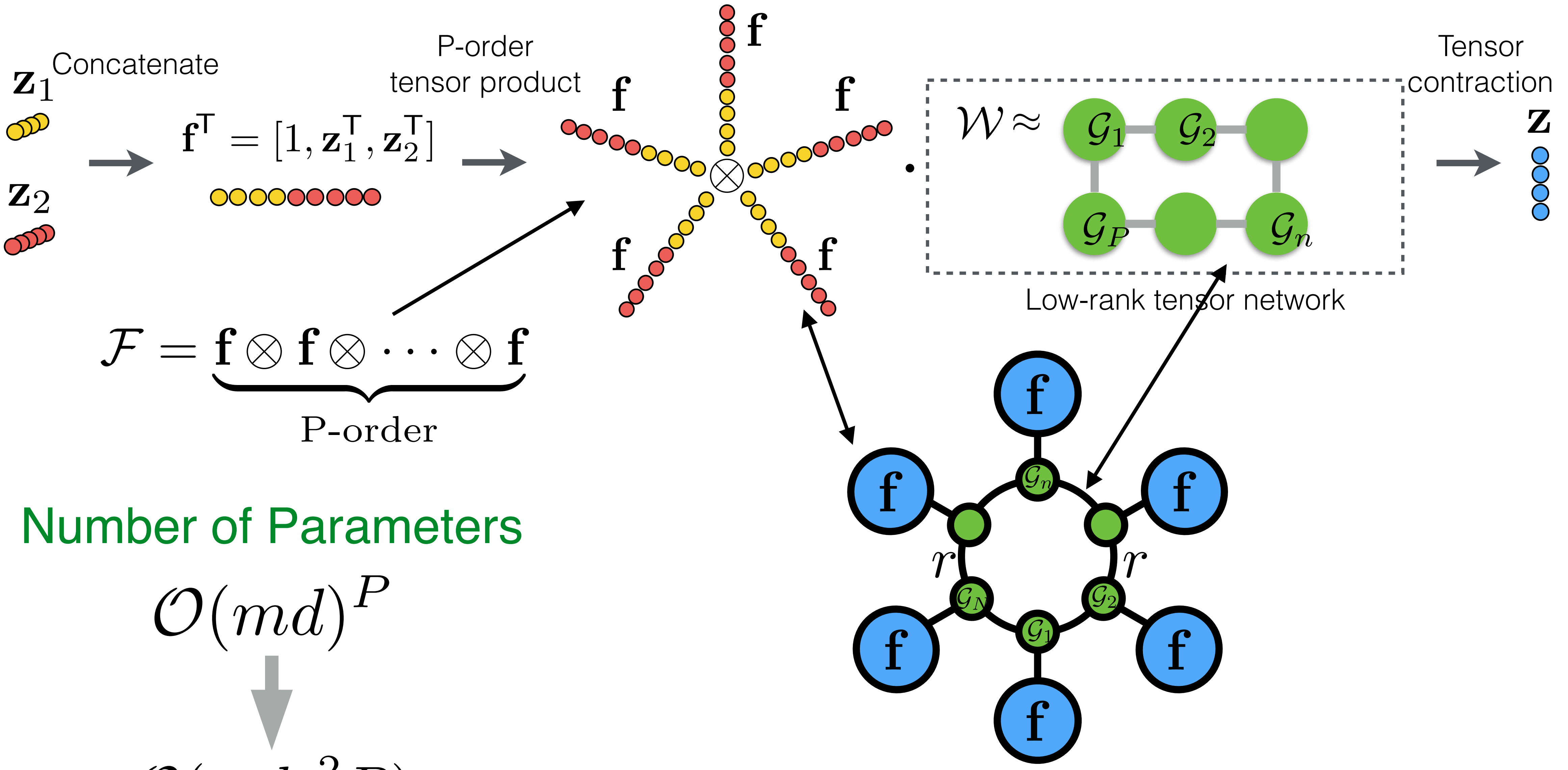
$$d \mapsto 2^d$$

- Accuracy of 99.03% on MNIST by one layer

Supervised Learning with Quantum-Inspired Tensor Networks [Stoudenmire et al., NIPS 2016]

Tensor Polynomial Pooling (PTP)

(Hou et al., NeurIPS 2019)



$$\mathcal{F} = \underbrace{\mathbf{f} \otimes \mathbf{f} \otimes \cdots \otimes \mathbf{f}}_{\text{P-order}}$$

Number of Parameters

$$\mathcal{O}(md)^P$$

$$\downarrow$$

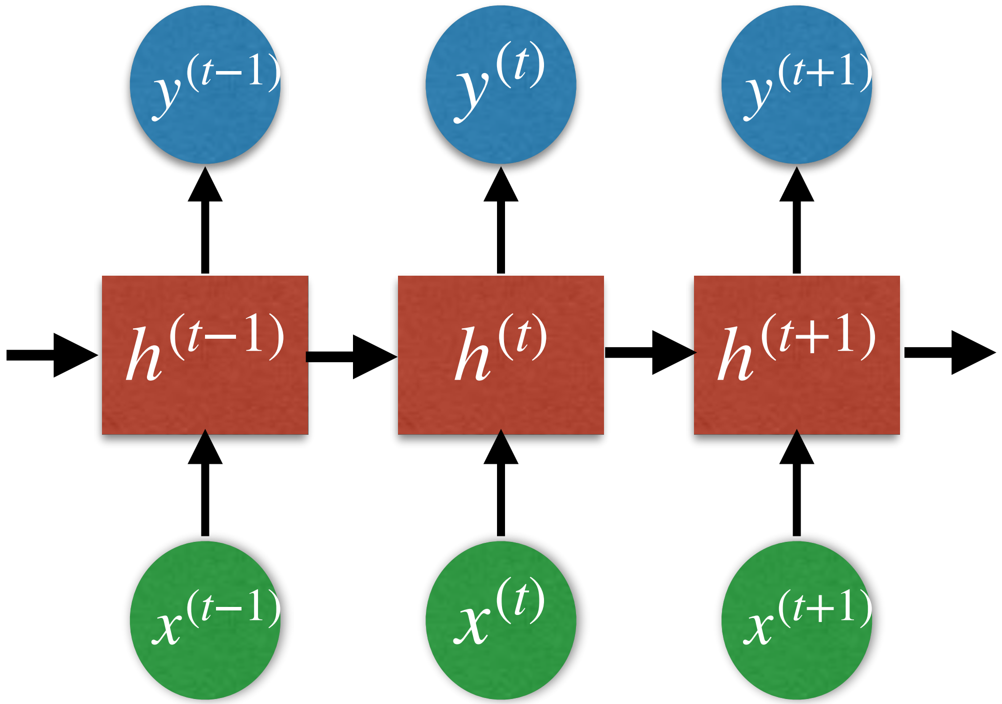
$$\mathcal{O}(mdr^2P)$$

Polynomially enhanced capacity with linearly increasing number of parameters

Tensor-Power Recurrent Models

(Li et al., AISTATS 2021)

- RNN and LSTM do not have long memory from a statistical perspective [Zhao et al., ICML 2020]



RNN

Transition function

$$h^{(t)} = \sigma(Wh^{(t-1)} + Ux^{(t)} + b)$$

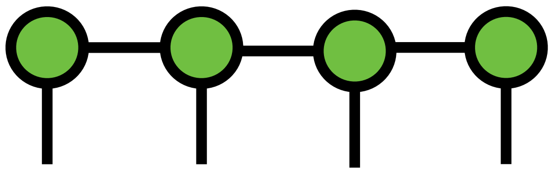
$(p+1)$-order weight tensor

$$\mathbf{h}^{(t)} = \mathcal{G} \times_1 \underbrace{\begin{pmatrix}\mathbf{x}^{(t)} \\ \mathbf{h}^{(t-1)}\end{pmatrix} \times_2 \cdots \times_p \begin{pmatrix}\mathbf{x}^{(t)} \\ \mathbf{h}^{(t-1)}\end{pmatrix}}_{p\text{-fold tensor product with itself}} = \mathcal{G} \cdot \begin{pmatrix}\mathbf{x}^{(t)} \\ \mathbf{h}^{(t-1)}\end{pmatrix}^{\otimes p}$$

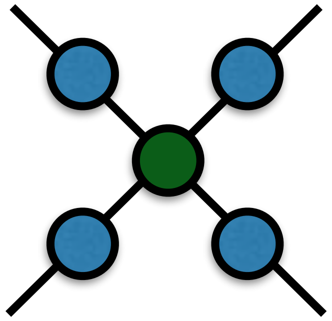
Large p leads to long memory, small p leads to short memory

Tensor Networks in Deep Learning

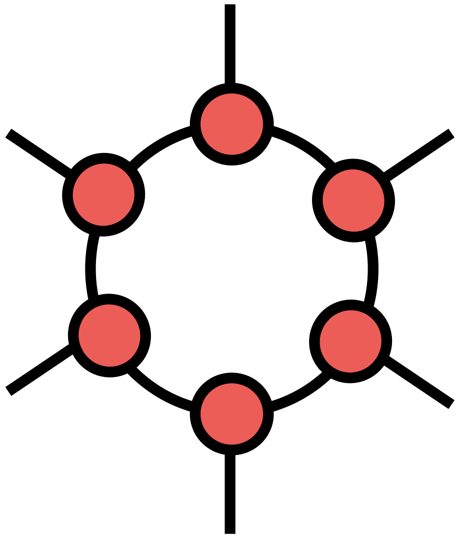
Full-connected network
(Novikov et al., 2015)

$\mathcal{Y} = \langle \mathcal{W}, \phi(\mathcal{X}) \rangle = \Big<$  $, \phi(\mathcal{X}) \Big>$

Regression network
(Kossaifi et al., 2020)

$\mathcal{Y} = \langle \mathcal{W}, \phi(\mathcal{X}) \rangle = \Big<$  $, \phi(\mathcal{X}) \Big>$

Convolutional network
(Wang et al., 2019)

$\mathcal{Y} = \langle \mathcal{W}, \phi(\mathcal{X}) \rangle = \Big<$  $, \phi(\mathcal{X}) \Big>$

Which is the optimal TN structure for machine learning tasks?

Tensor Network Structure Search (TN-SS)

Involved Algorithm

Spectral Decomp.

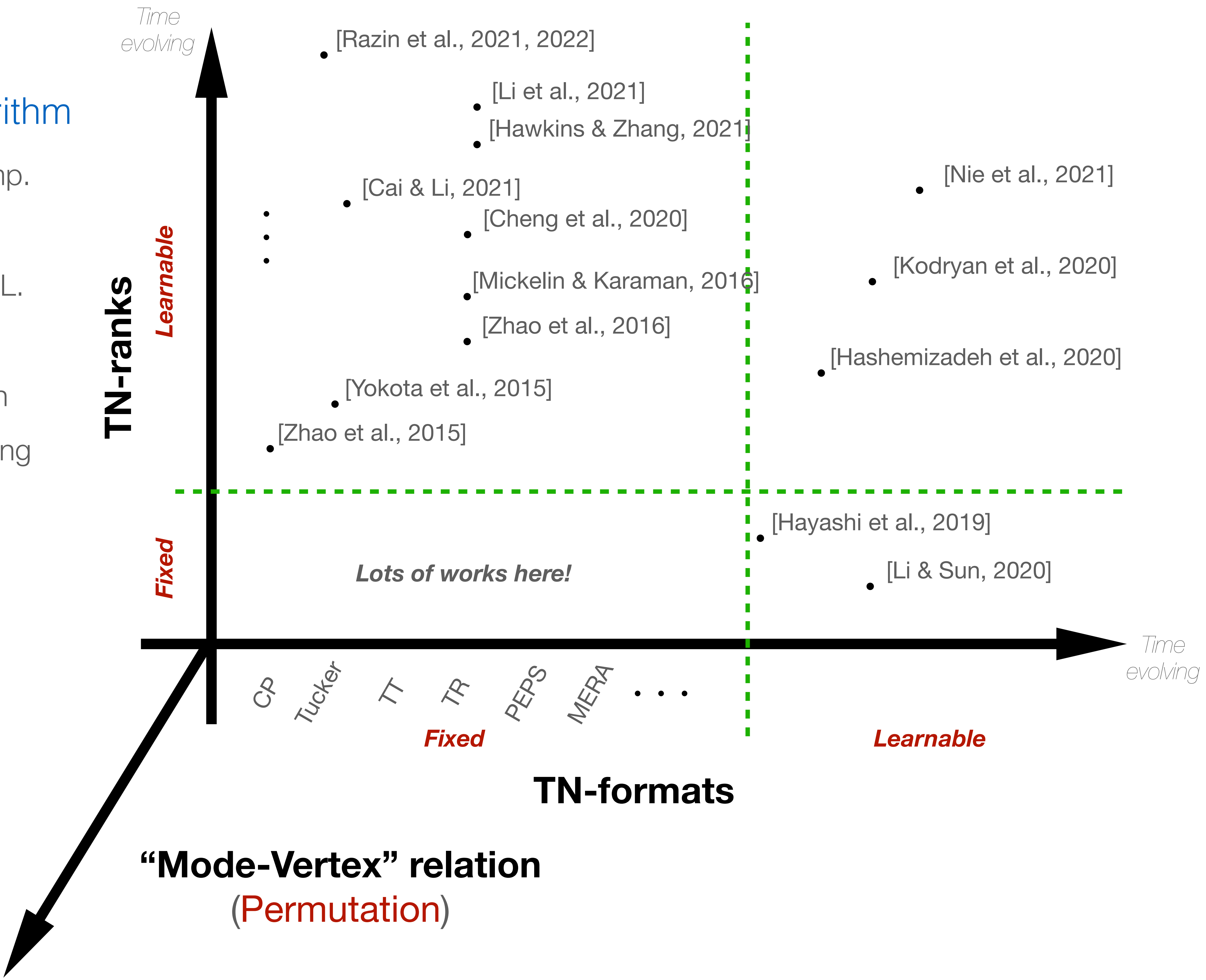
Bayesian Inf.

Reinforcement L.

Implicit regul.

Greedy search

Random sampling



Time evolving

TN-ranks

Learnable

Fixed

[Razin et al., 2021, 2022]

[Li et al., 2021]

[Hawkins & Zhang, 2021]

[Cai & Li, 2021]

[Cheng et al., 2020]

[Mickelin & Karaman, 2016]

[Zhao et al., 2016]

[Yokota et al., 2015]

[Zhao et al., 2015]

[Nie et al., 2021]

[Kodryan et al., 2020]

[Hashemizadeh et al., 2020]

[Hayashi et al., 2019]

[Li & Sun, 2020]

Lots of works here!

CP Tucker TT TR PEPS MERA . . .

Fixed

Learnable

Time evolving

TN-formats

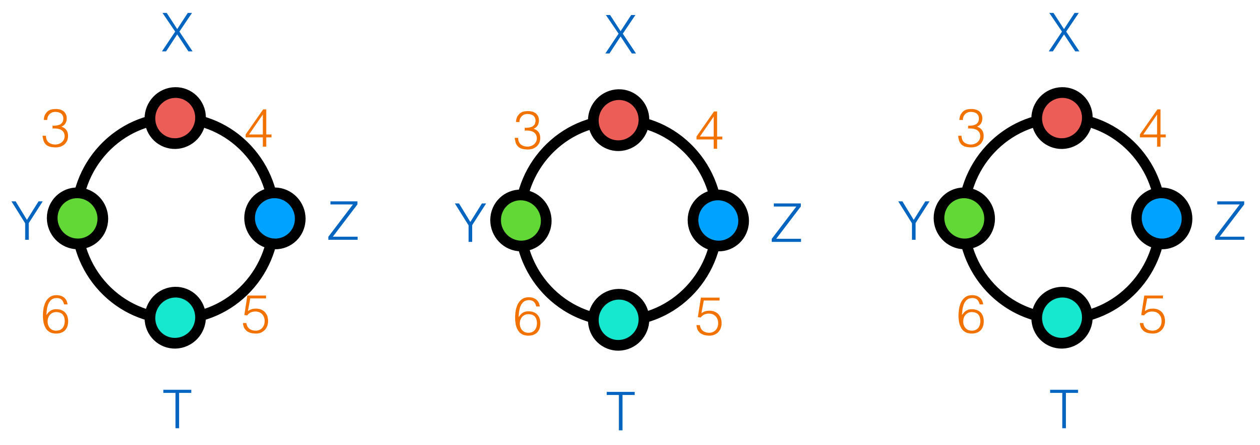
“Mode-Vertex” relation

(Permutation)

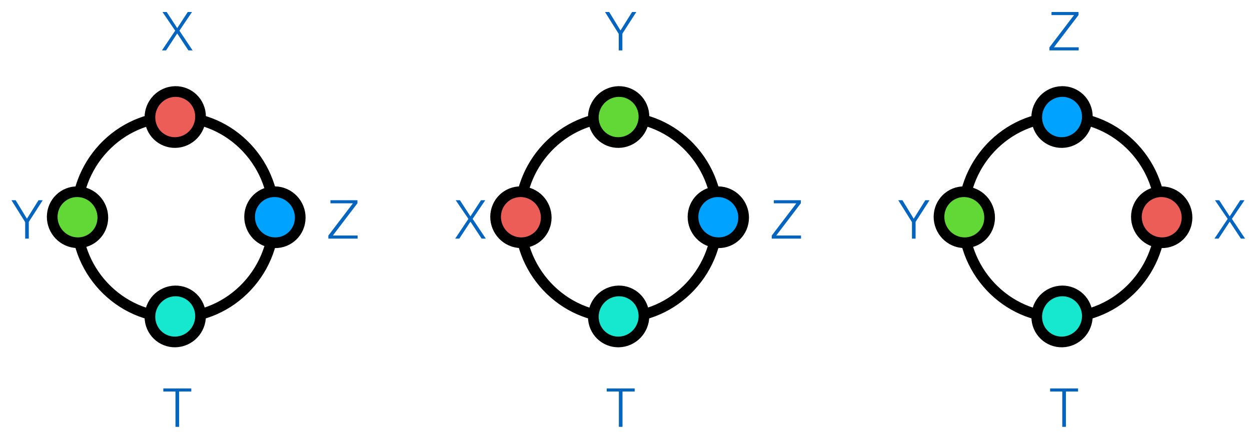
TN Structure Search

**The dangling edges are ignored.*

TN-RS
(Rank, edge labels)

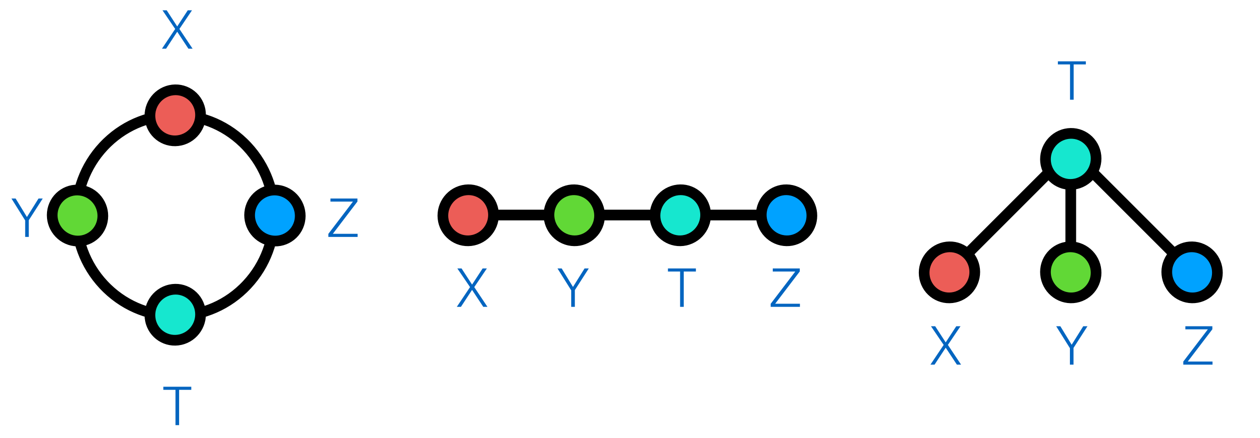


TN-PS
(Vertex Permutation)



(Li et al., ICML 2021)

TN-TS
(Network Topology)



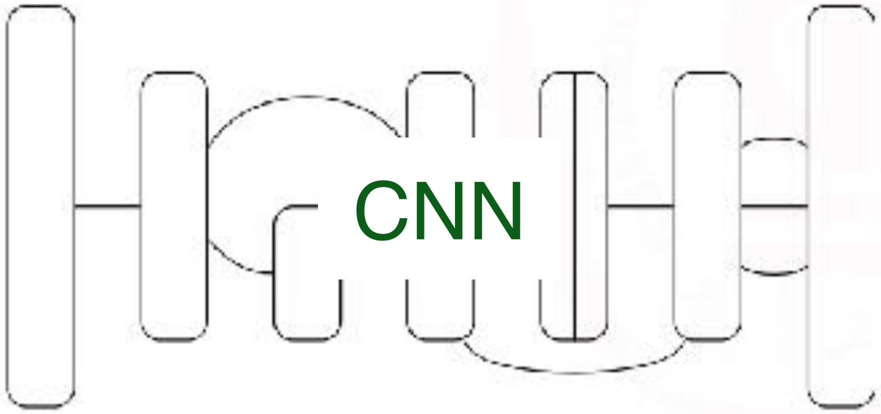
(Li et al., ICML 2020)

Image source: https://staffwww.dcs.shef.ac.uk/people/H.Lu/feeler.html

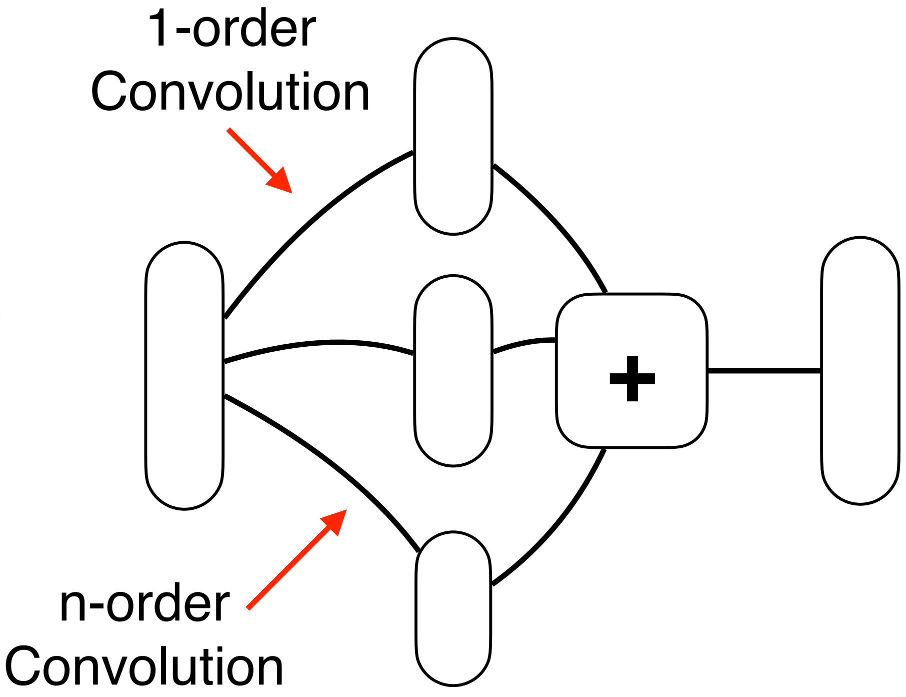
Understanding CNN from Volterra Convolution Perspective

(Li et al. JMLR 2022)

- **Theorem**: Most convolutional neural networks can be interpreted as a form of Volterra convolutions.



$$\Rightarrow \sum_{n=0} \mathbf{H}_n * \mathbf{x}^n$$



- **Volterra Convolution**

n-order kernel tensor

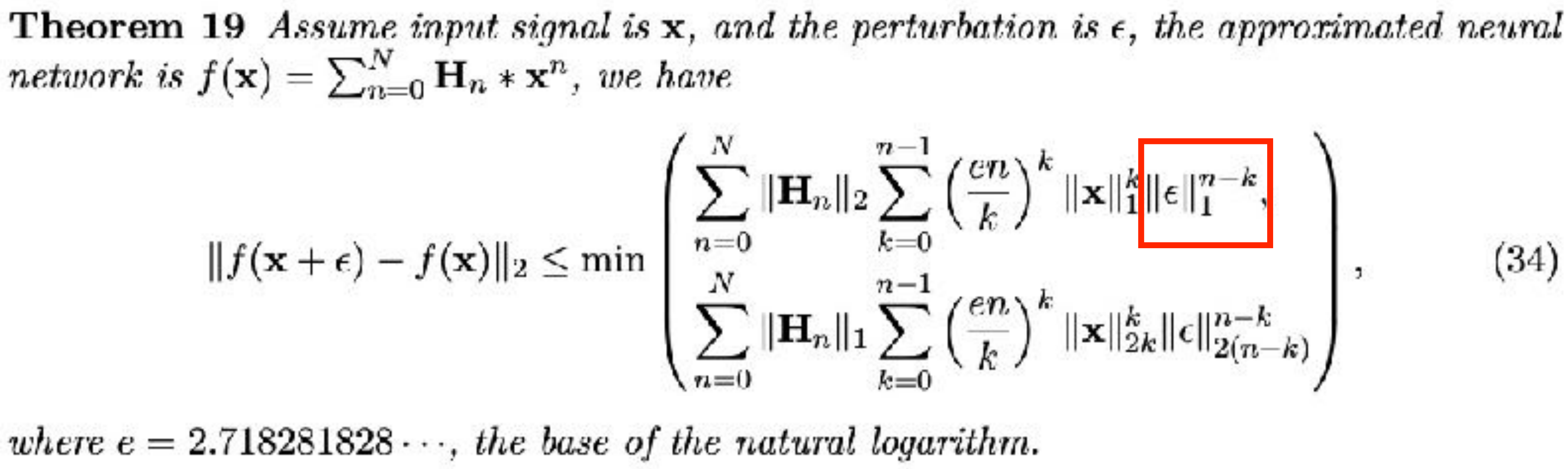
$$\left(\sum_{n=0}^{+\infty}\mathbf{H}_n * \mathbf{x}^n\right)(t) = \sum_{n=0}^{+\infty} \underbrace{\int_{-\infty}^{+\infty}\cdots\int_{-\infty}^{+\infty} H_n(\tau_1,\cdots,\tau_n)\prod_{i=1}^{n}(x(t-\tau_i)d\tau_i)}_{\text{NOT n-dimensional convolution}}$$

Black-box Attack by Volterra Convolution

(Li et al. JMLR 2022)

Well trained CNN → VC representation

- **Direct computation** (white box) or **training a VC network** by proxy kernels (black box)
- The perturbation computed by attacking VC can also **attack original CNN**.
- **Upper bound** w.r.t. perturbation



Theorem 19 *Assume input signal is* $\mathbf{x}$, *and the perturbation is* ϵ, *the approximated neural network is* $f(\mathbf{x}) = \sum_{n=0}^{N} \mathbf{H}_n * \mathbf{x}^n$, *we have*

$$\|f(\mathbf{x}+\epsilon) - f(\mathbf{x})\|_2 \leq \min \left(\begin{array}{c} \sum_{n=0}^{N} \|\mathbf{H}_n\|_2 \sum_{k=0}^{n-1} \left(\frac{en}{k}\right)^k \|\mathbf{x}\|_1^k \boxed{\|\epsilon\|_1^{n-k},} \\ \sum_{n=0}^{N} \|\mathbf{H}_n\|_1 \sum_{k=0}^{n-1} \left(\frac{en}{k}\right)^k \|\mathbf{x}\|_{2k}^k \|\epsilon\|_{2(n-k)}^{n-k} \end{array} \right), \qquad (34)$$

where $e = 2.718281828\cdots$, *the base of the natural logarithm.*

Computational Efficiency

Discovering efficient algorithms in mathematics

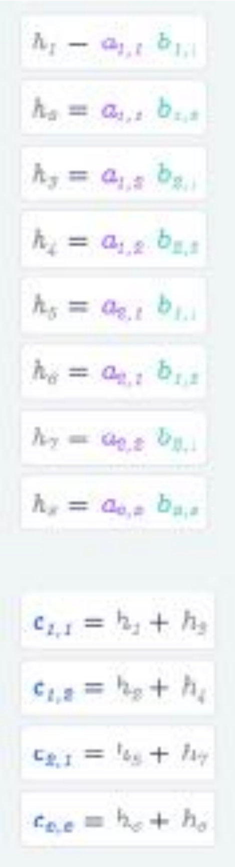
- Matrix multiplication: ubiquitous in NNs and modern computing
  - Developing computing hardware (large amounts of time and money)
  - Finding the fastest algorithm (50-year-old open question, difficult problem in mathematics)
- Example: 2 x 2 matrices

$$\begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix} \times \begin{bmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{bmatrix} = \begin{bmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{bmatrix}$$

- Unsolved problem in larger matrix cases
- Automatic algorithm discovery by AI

[Fawzi et al. Nature 2022]

Standard algorithm



$h_1 = a_{1,1}\, b_{1,1}$

$h_2 = a_{1,1}\, b_{1,2}$

$h_3 = a_{1,2}\, b_{2,1}$

$h_4 = a_{1,2}\, b_{2,2}$

$h_5 = a_{2,1}\, b_{1,1}$

$h_6 = a_{2,1}\, b_{1,2}$

$h_7 = a_{2,2}\, b_{2,1}$

$h_8 = a_{2,2}\, b_{2,2}$

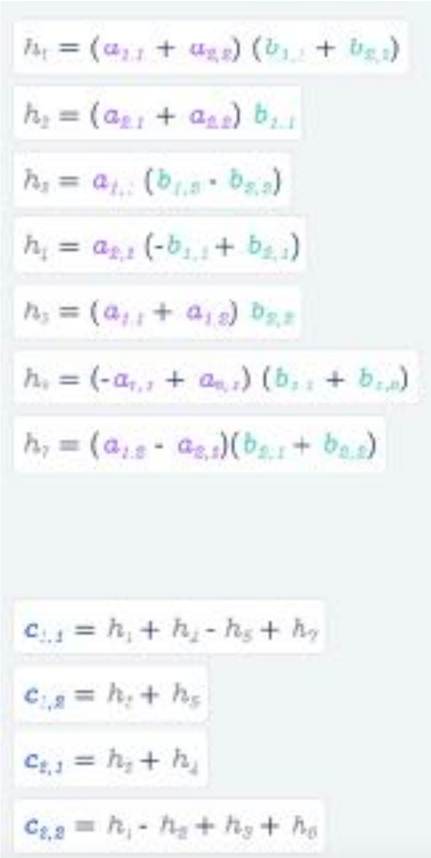
$c_{1,1} = h_1 + h_3$

$c_{1,2} = h_2 + h_4$

$c_{2,1} = h_5 + h_7$

$c_{2,2} = h_6 + h_8$

Strassen's algorithm



$h_1 = (a_{1,1} + a_{2,2})\,(b_{1,1} + b_{2,2})$

$h_2 = (a_{2,1} + a_{2,2})\, b_{1,1}$

$h_3 = a_{1,1}\,(b_{1,2} - b_{2,2})$

$h_4 = a_{2,2}\,(-b_{1,1} + b_{2,1})$

$h_5 = (a_{1,1} + a_{1,2})\, b_{2,2}$

$h_6 = (-a_{1,1} + a_{2,1})\,(b_{1,1} + b_{1,2})$

$h_7 = (a_{1,2} - a_{2,2})(b_{2,1} + b_{2,2})$

$c_{1,1} = h_1 + h_4 - h_5 + h_7$

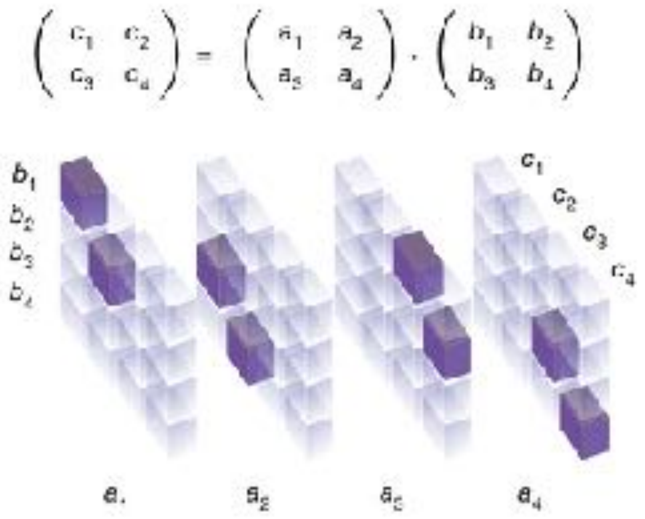
$c_{1,2} = h_3 + h_5$

$c_{2,1} = h_2 + h_4$

$c_{2,2} = h_1 - h_2 + h_3 + h_6$

AlphaTensor: Discovering novel algorithms using Tensor Decomposition

Encoding



$\begin{pmatrix} c_1 & c_2 \\ c_3 & c_4 \end{pmatrix} = \begin{pmatrix} a_1 & a_2 \\ a_3 & a_4 \end{pmatrix} \cdot \begin{pmatrix} b_1 & b_2 \\ b_3 & b_4 \end{pmatrix}$
b_1
b_2
b_3
b_4
c_1
c_2
c_3
c_4
a_1
a_2
a_3
a_4

Tensor Decomposition

$$\mathbf{U} = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & -1 \end{pmatrix}$$

$$\mathbf{V} = \begin{pmatrix} 1 & 1 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & -1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

$$\mathbf{W} = \begin{pmatrix} 1 & 0 & 0 & 1 & -1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & -1 & 1 & 0 & 0 & 1 & 0 \end{pmatrix}$$

$$\mathcal{T}_n = \sum_{r=1}^{R} \mathbf{u}^{(r)} \otimes \mathbf{v}^{(r)} \otimes \mathbf{w}^{(r)},$$

Decoding

$m_1 = (a_1 + a_4)(b_1 + b_4)$

$m_2 = (a_3 + a_4)\, b_1$

$\boxed{m_3 = a_1\,(b_2 - b_4)}$

$m_4 = a_4\,(b_3 - b_1)$

$m_5 = (a_1 + a_2)\, b_4$

$m_6 = (a_3 - a_1)(b_1 + b_2)$

$m_7 = (a_2 - a_4)(b_3 + b_4)$

$c_1 = m_1 + m_4 - m_5 + m_7$

$c_2 = m_3 + m_5$

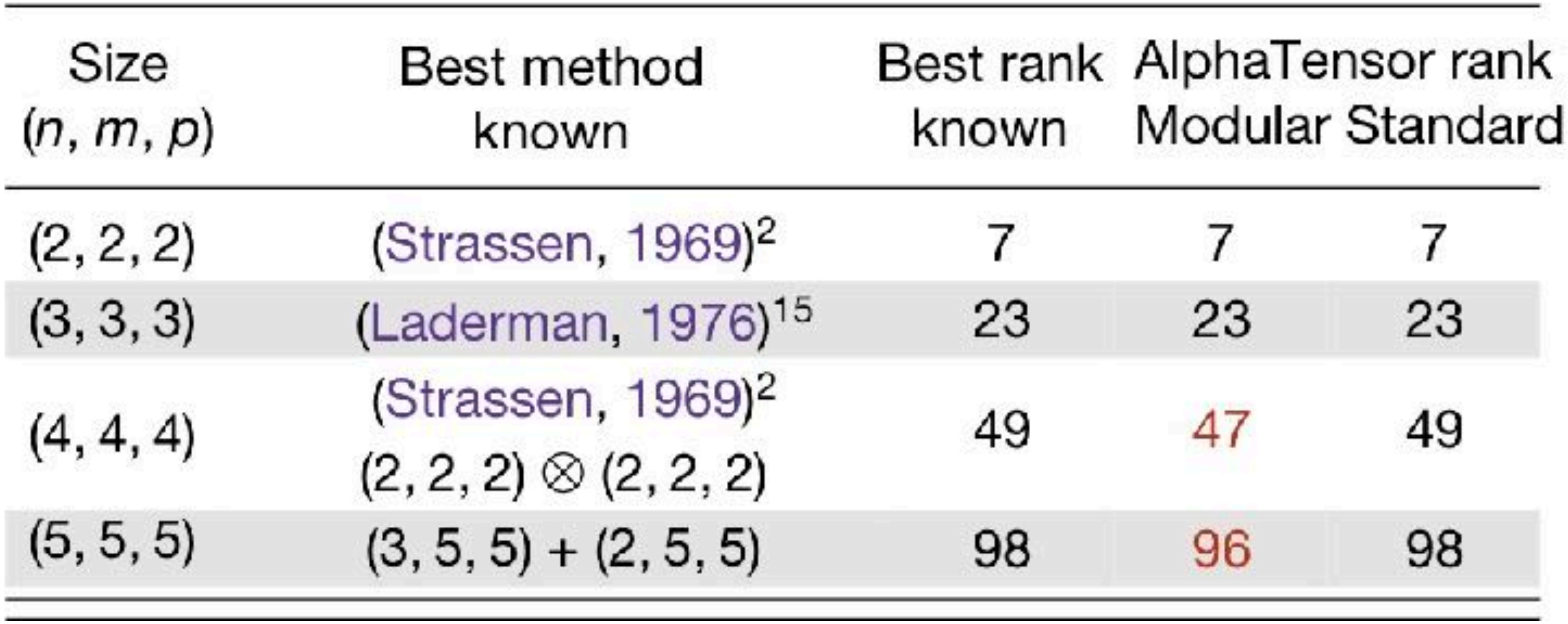
$\boxed{c_3 = m_2 + m_4}$

$c_4 = m_1 - m_2 + m_3 + m_6$

Rank of CPD determines the minimum number of multiplications

[Fawzi et al. Nature 2022]

AlphaTensor: Discovering novel algorithms in mathematics

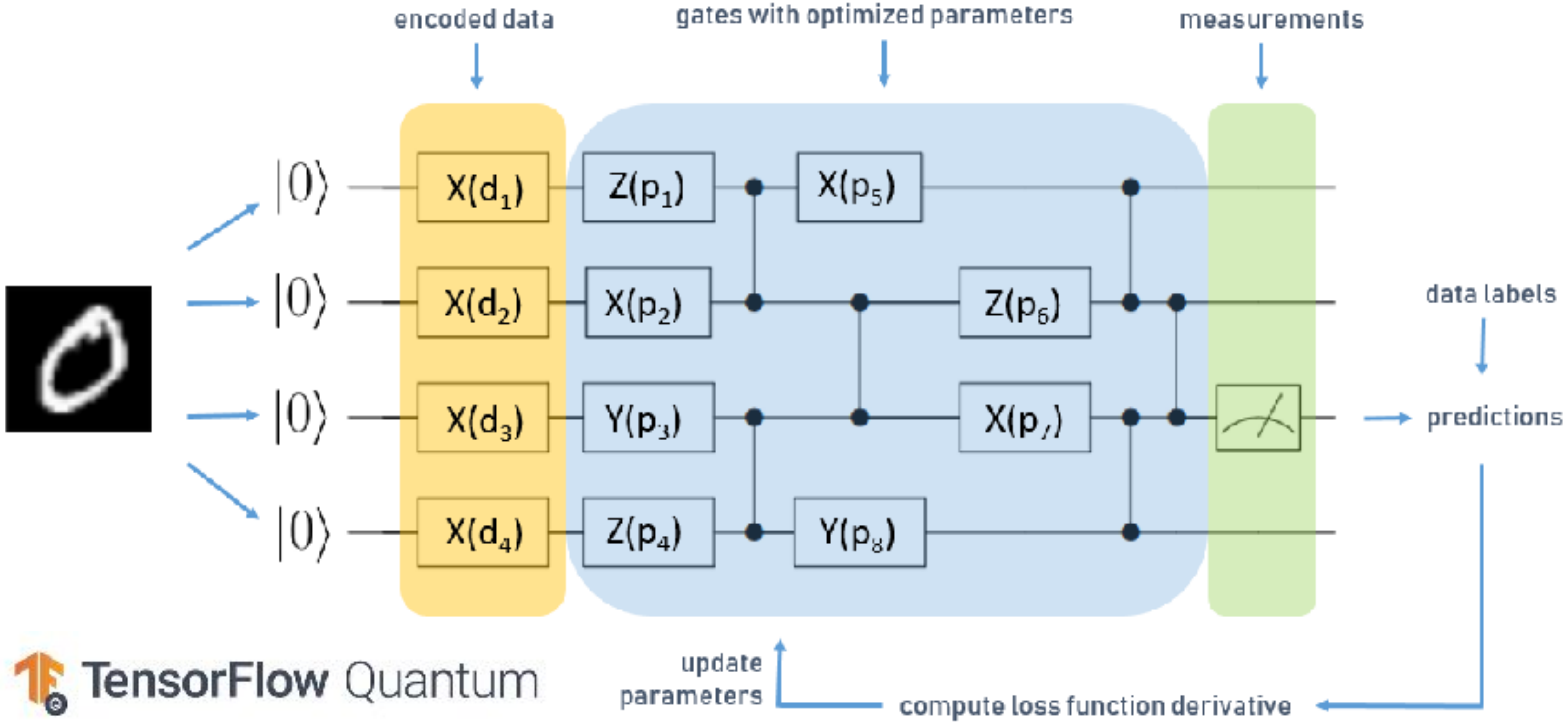


Size	Best method	Best rank	AlphaTensor rank	
(n, m, p)	known	known	Modular	Standard
(2, 2, 2)	(Strassen, 1969)[2]	7	7	7
(3, 3, 3)	(Laderman, 1976)[15]	23	23	23
(4, 4, 4)	(Strassen, 1969)[2] $(2, 2, 2) \otimes (2, 2, 2)$	49	47	49
(5, 5, 5)	(3, 5, 5) + (2, 5, 5)	98	96	98

- Discovered algorithm outperforms the two-level Strassen's algorithm (best human knowledge).
- One week later, *Manuel Kauers* and *Jakob Moosbauer* beat AlphaTensor (5 x 5 matrix , 96 -> 95). [Kauers et al. ArXiv 2022]

[Fawzi et al. Nature 2022]

Quantum Machine Learning



- Limited qubits with small scale data and model.
- Performance on ML tasks cannot compete with classical ML.

https://blog.tensorflow.org/2020/08/layerwise-learning-for-quantum-neural-networks.html

Summary

- TNs are powerful tools for representation of high-dimensional structured data.
- TNs are efficient reparameterization of deep learning models.
- However, there are some problems need to further solved prior to the real-world applications, such as TN-SS.
- Robustness to adversarial attacks, and interpretability of TN based models.
- Quantum machine learning might be potentially promising.

Acknowledgements

Chao Li

Jianfu Zhang

Andong Wang

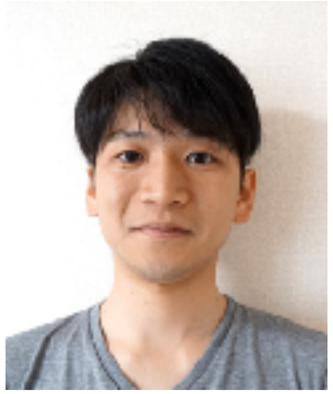
Zerui Tao

Mingyuan Bai

Andrzej Cichocki

Toshihisa Tanaka

Cesar F. Caiafa


Tatsuya Yokota

Yubang Zheng