Probabilistic Neural Kernel Tensor Decomposition

Conor Tillinghast, Shikai Fang, Kai Zhang, Shandian Zhe

University of Utah, Temple University

ICDM 2020

Outline

THE UNIVERSITY OF UTAH

- Background on Tensor decomposition
- Our method, POND (Probabilistic Neural Kernel Tensor Decompostion)
- Comparison of POND to other methods for tensor completion
- Application to Click-Through-Rate (CTR) prediction

Tensor Decomposition

THE UNIVERSITY OF UTAH

- Tensors are an important tool in studying multiway data
- Tensor decomposition estimates a set of latent factors that represent the nodes in each mode of the tensor
- Numerous applications such as in recommendation systems and CTR prediction
- Difficulties include sparsity of data, which makes it easy for models to overfit

Preliminaries on Decomposition

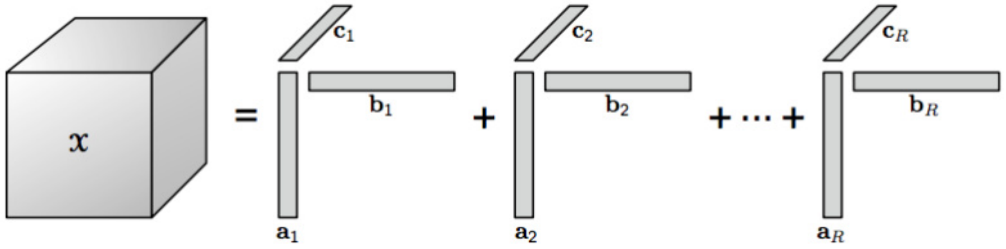
- Consider a tensor (user, item, shopping site, time)
  - Four modes
- For each of the nodes in the modes associate a latent factor vector (i.e. a vector for each item, vector for a person)
  - For each of the modes, vectors can be made into a matrix
  - Rank is dimension of the vectors

- Goal is to learn the set of factor matrices that can be used to accurately reconstruct the tensor

$$\mathcal{U} = \{\mathbf{U}^1, \dots, \mathbf{U}^K\}$$

Drawbacks of Current Methods

- CP and Tucker are classical models
  - Rely on multilinear assumptions
  - Ignores all non-linear interactions



- Multilayer perceptron models severely overfit due to data sparsity

- Convolution neural networks (CostCo) achieve better performance

- Many models do not include uncertainty formation
  - Likely more confidence in nodes that are observed often

POND: Probabilistic Neural Kernel Tensor Decomposition

THE UNIVERSITY OF UTAH

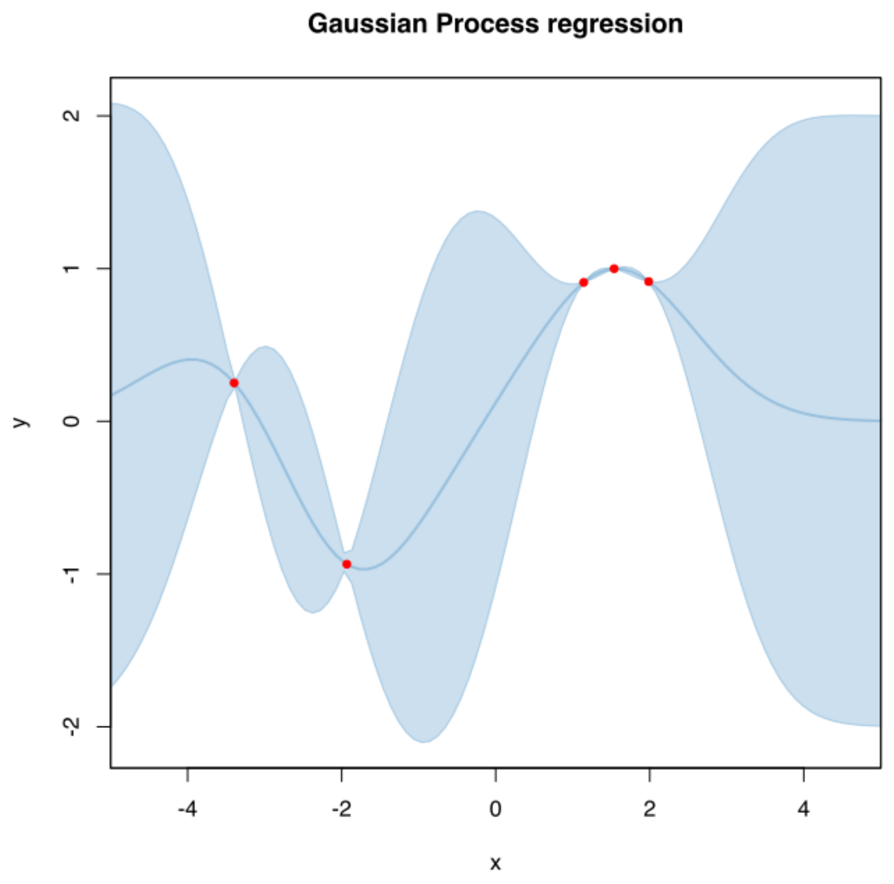
- POND uses GP regression with a neural network kernel
  - GP: Avoids overfitting but captures non linear relationships
  - Neural network kernel: improves expressive power compared to shallow kernels and can adapt to the complexity of the observed data
- Bayesian Approach: POND uses variational inference to approximate the posterior distribution of the latent factors
  - Useful in quantifying uncertainty of factors and confidence in predictions
- POND is learned using an efficient stochastic algorithm
  - Scalable to large datasets

GP Regression

- Non parametric form regression
  - The kernel specifies the degree of correlation between points
  - Simple kernels can make oversimplified assumptions

$$k_{\text{RBF}}(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\frac{1}{\eta}\|\mathbf{x}_i - \mathbf{x}_j\|^2)$$

- Allows for quantification of uncertainty in predictions



GP Tensor Decomposition

THE UNIVERSITY OF UTAH

- Input set of observed indices and values

$$\mathbf{i} = (\mathbf{i}_1, \ldots, \mathbf{i}_K) \qquad\qquad y_{\mathbf{i}}$$

  - Each index corresponds to a set of latent factors

$$\mathbf{x_i} = [(\mathbf{u}^1_{i_1})^\top, \ldots, (u^K_{i_K})^\top]^\top$$

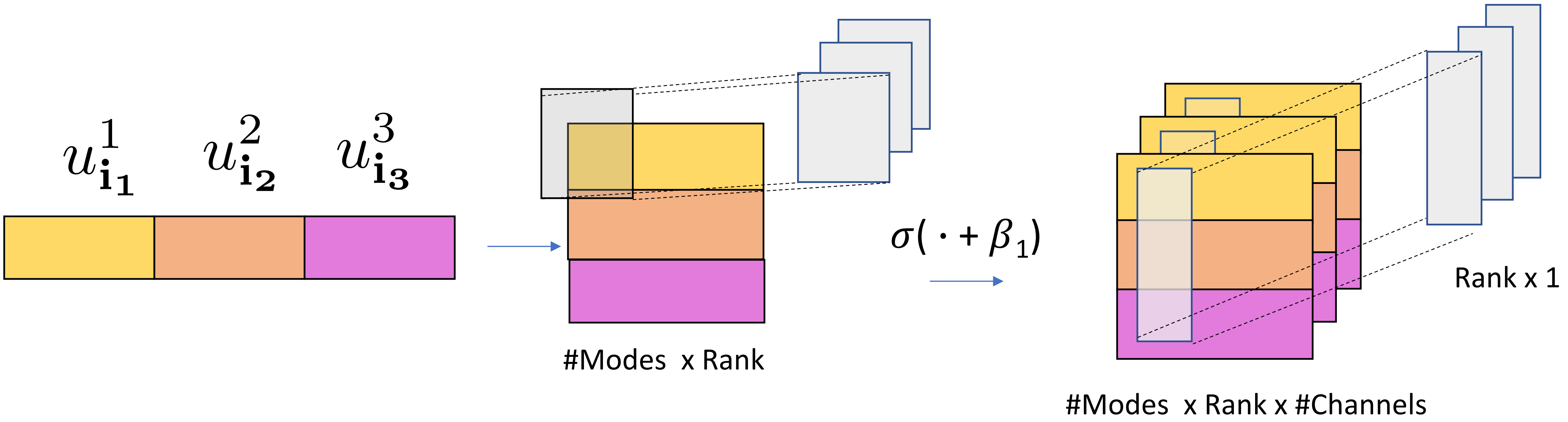
- Assume relationship between set of latent factors and values is given by GP plus some noise

$$\mathbf{y_i} = f(\mathbf{x_i}) + \epsilon_{\mathbf{i}}$$

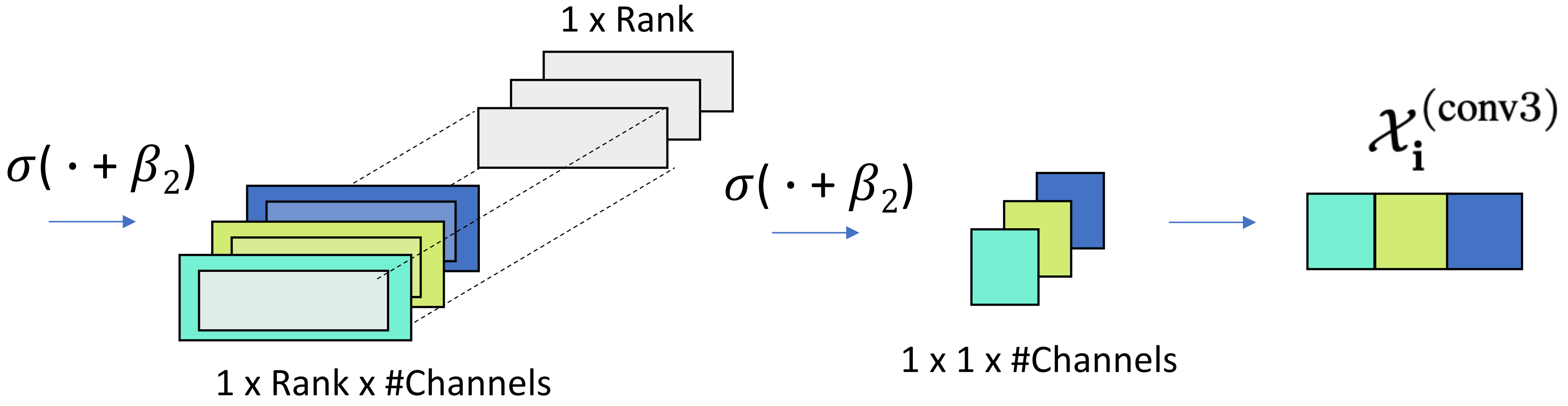
- Add normal prior over the latent factors
  - Captures uncertainty in latent factors
  - Latent Factor GP model
- Assume covariance function is given by a neural kernel

POND: Neural Kernel

Input index: $\mathbf{i} = (\mathbf{i_1}, \mathbf{i_2}, \mathbf{i_3})$



POND: Neural Kernel



$$k(\mathbf{x_i}, \mathbf{x_j}) = \exp\left(-\frac{\|\text{vec}(\mathcal{X}_{\mathbf{i}}^{\text{conv3}}) - \text{vec}(\mathcal{X}_{\mathbf{j}}^{\text{conv3}})\|^2}{\eta}\right)$$

Model Estimation

- Analytically intractable with factor matrices coupled in neural kernel
- Use sparse variational inference
  - Introduce approximate posteriors

$$\mathcal{N}\left(\mathbf{u}_t^k | \boldsymbol{\alpha}_t^k, \mathrm{diag}(\mathbf{v}_t^k)\right)$$

- Minimize the KL divergence of the variational approximation and true posterior $\mathrm{KL}\left(q(\mathcal{U}, \mathbf{b}, \mathbf{f}_S) \| p(\mathcal{U}, \mathbf{b}, \mathbf{f}_S | \mathbf{y}_S)\right)$
- ELBO decomposes over entries
  - Can use variations of stochastic gradient descent to optimize with the reparametrization trick

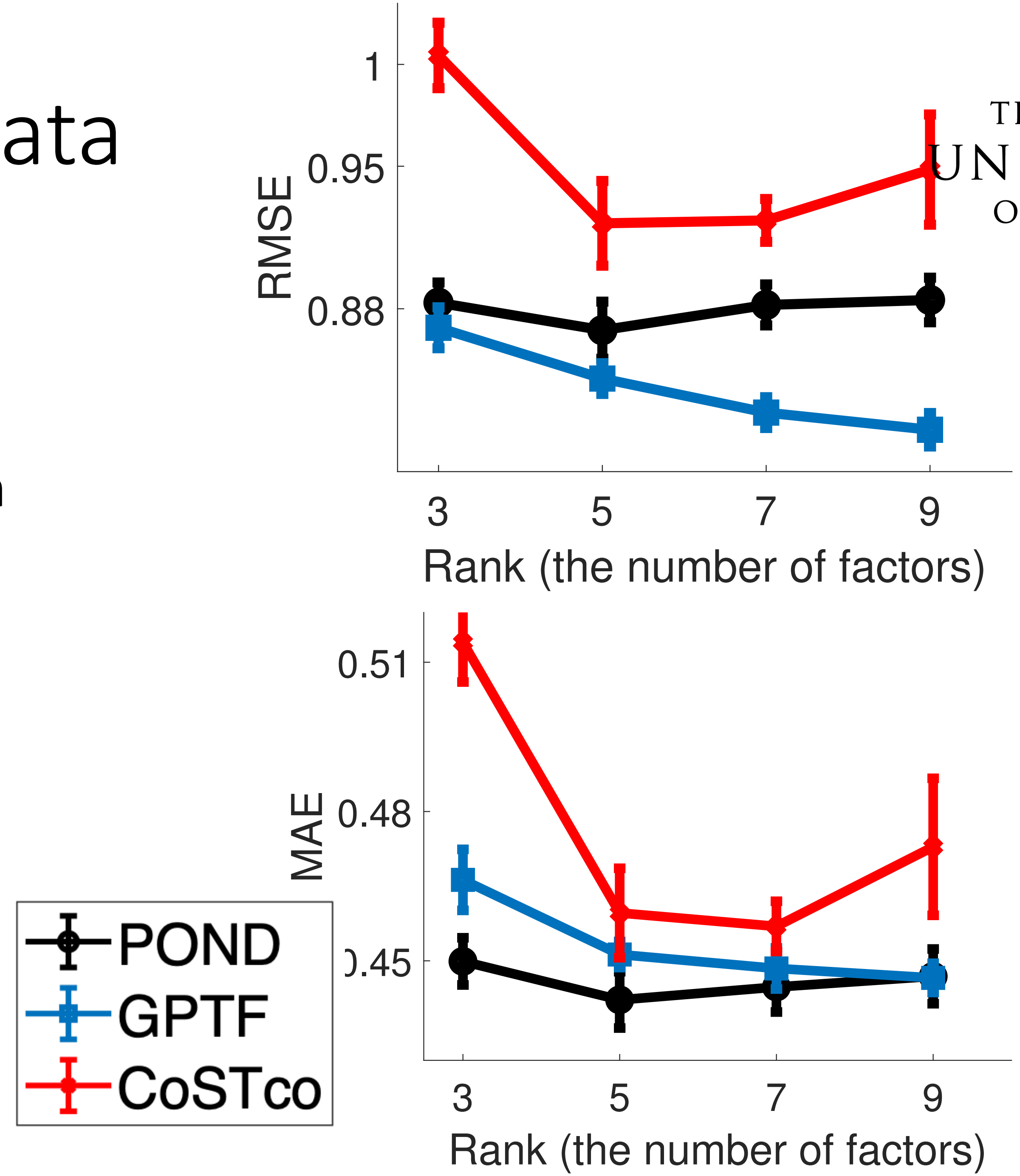
Tensor Completion Experiments

- Competing Methods
  - DFNT
    - Uses shallow RBF kernel with a different ELBO
    - Only returns point estimates
  - CostCo
    - Convolutional neural network
      - 2 convolutional layers followed by dense layers
  - GPTF
    - Our method with a shallow RBF kernel
  - P-Tucker
    - A probabilistic Tucker decomposition
  - Two CP decompositions
    - CP-ALS
    - CP-WOPT

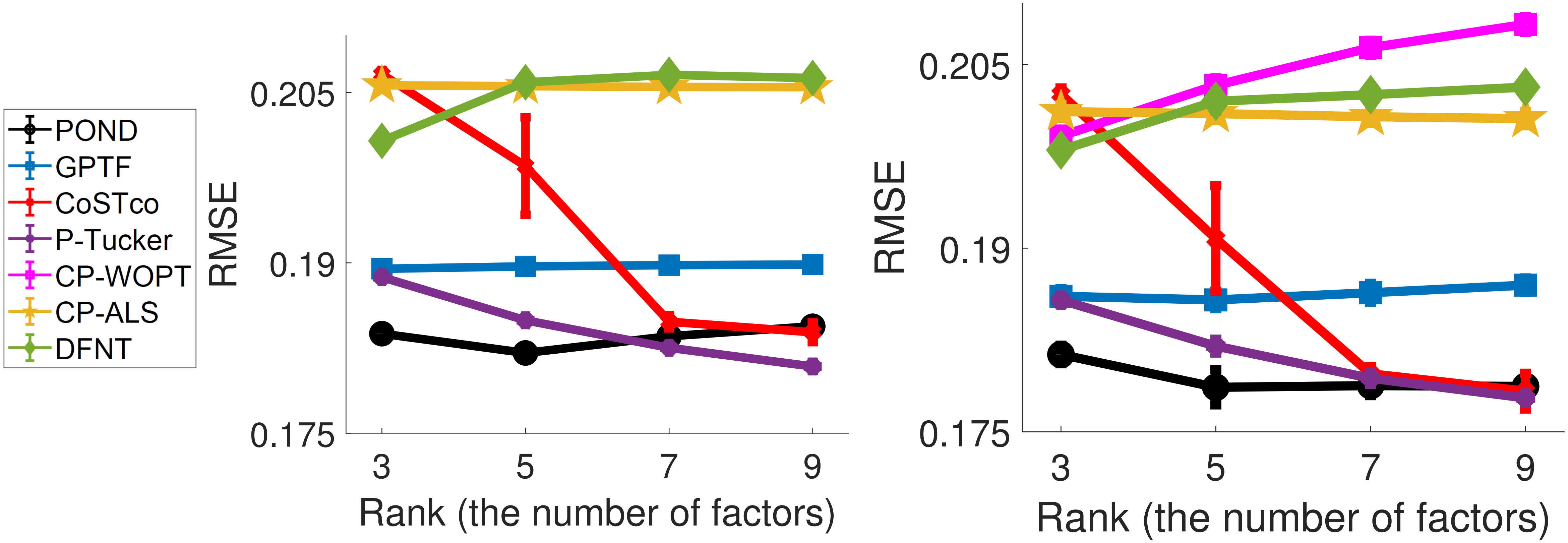
- Datasets
  - ALOG
    - 200 x 100 x 200
    - (user, action, resource)
    - 0.66% observed
  - MovingMNIST
    - 20 x 100 x 64 x 64
    - (video, frame, row, column)
    - 3% and 10 % observed
  - ExtremeClimate
    - 360 × 768 × 1152 x 16
    - (time, lattitute, longitude, variable)
    - 0.0008% Observed

POND on Small Data

- ALOG dataset 200 x 100 x 200
- Bayesian methods outperform CoSTco
- POND is able to learn the compexity, does not overfit as much as CostCo
- On simple data the shallow kernel is sufficient



MovingMNIST



3% Observed

10% Observed

Extreme Climate

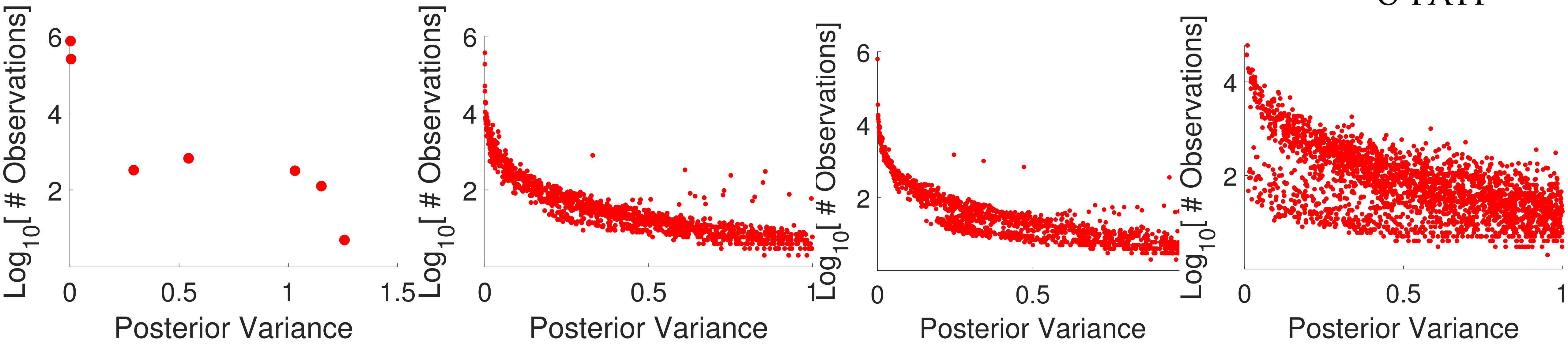
metric	method/rank	3	5	10	20
RMSE	CP-ALS	0.7904 ± 0.0022	0.7904 ± 0.0022	0.7904 ± 0.0022	0.7904 ± 0.0022
	CP-WOPT	2.3604 ± 0.1462	3.3917 ± 0.1670	6.0489 ± 0.2027	1.8680 ± 0.0179
	P-Tucker	0.1038 ± 0.0046	0.1496 ± 0.0147	0.1731 ± 0.0029	0.2632 ± 0.0049
	DFNT	0.1412 ± 0.0014	0.4534 ± 0.0042	0.7900 ± 0.0021	0.7900 ± 0.0021
	CoSTco	0.0842 ± 0.0009	0.0849 ± 0.0009	0.0839 ± 0.0009	$\mathbf{0.0833 \pm 0.0011}$
	GPTF	0.0916 ± 0.0016	0.0969 ± 0.0015	0.969 ± 0.0014	0.0938 ± 0.0016
	POND	$\mathbf{0.0829 \pm 0.0012}$	$\mathbf{0.0827 \pm 0.0012}$	$\mathbf{0.0837 \pm 0.0013}$	0.0847 ± 0.0012
MAE	CP-ALS	0.7369 ± 0.0026	0.7369 ± 0.0026	0.7369 ± 0.0025	0.7369 ± 0.0025
	CP-WOPT	1.0552 ± 0.0136	1.3527 ± 0.0117	2.4118 ± 0.0225	1.3271 ± 0.0091
	P-Tucker	0.0601 ± 0.0014	0.0831 ± 0.0066	0.1116 ± 0.0023	0.1961 ± 0.0035
	DFNT	0.0974 ± 0.0019	0.3865 ± 0.0048	0.7369 ± 0.0023	0.7369 ± 0.0023
	CoSTco	0.0508 ± 0.0006	0.0514 ± 0.0006	0.0505 ± 0.0006	0.0498 ± 0.0006
	GPTF	0.0581 ± 0.0012	0.0621 ± 0.0010	0.0621 ± 0.0014	0.0597 ± 0.0011
	POND	$\mathbf{0.0491 \pm 0.0007}$	$\mathbf{0.0492 \pm 0.0006}$	$\mathbf{0.0495 \pm 0.0007}$	$\mathbf{0.0497 \pm 0.0007}$

POND for CTR prediction

- Avazu data available on Kaggle
  - Records 10 days of ad impressions and clicks
  - Extract 4-mode tensor (banner position, site, app, device)
    - 7 x 2854 x 4114 x 6061
  - Binary tensor, 17.4% clicks

- Use POND with Probit likelihood

CTR Latent Factors

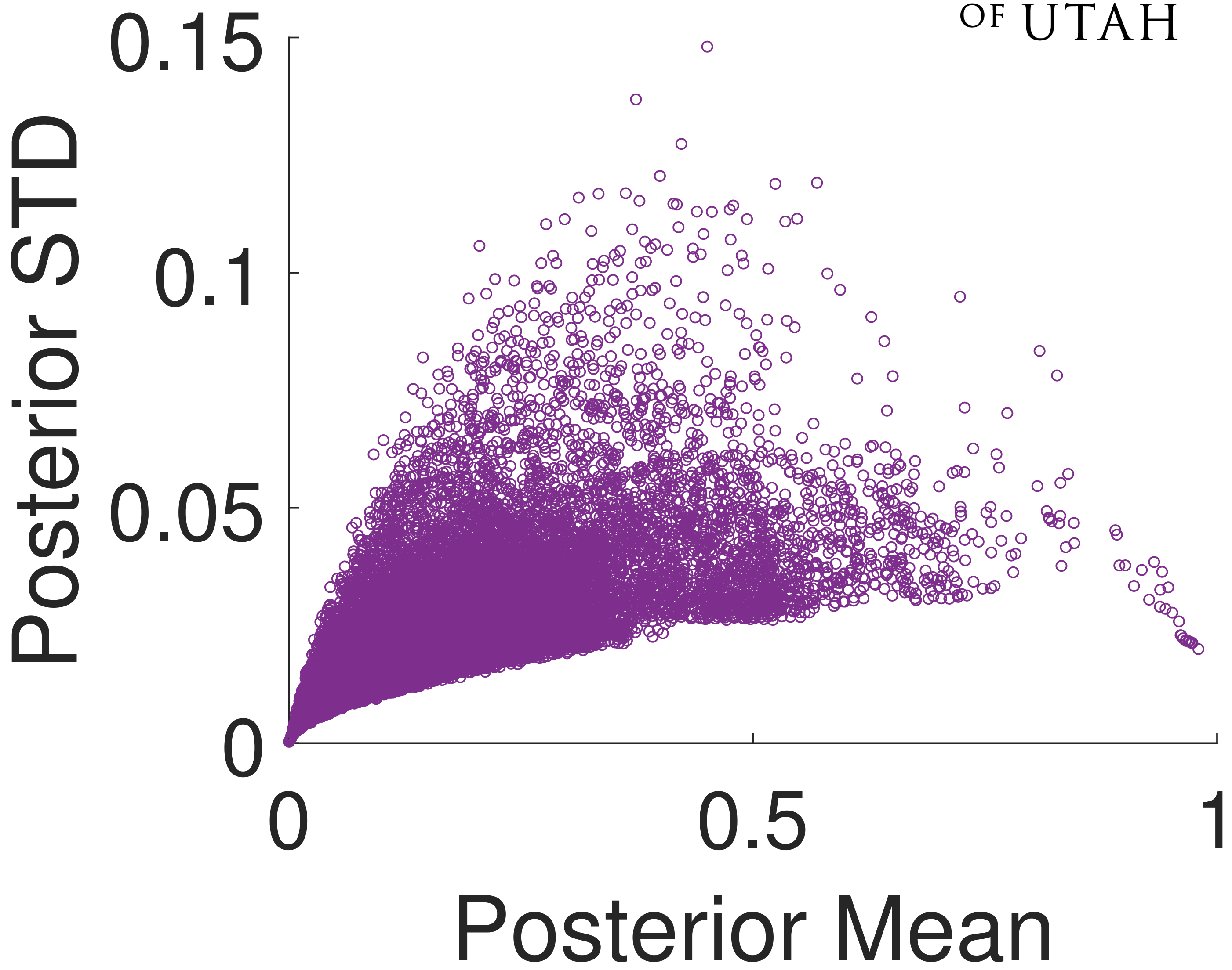


Banner Position **Site** **App** **Device**

- As number of observations increases, variance decreases
  - Less uncertain about active nodes

CTR Click Probabilities

- Plot of mean probability of a click
- Most ads are not clicked so more confident about prediction of 0 (no clicks)



Summary

- POND is a scalable Bayesian approach to tensor decomposition
  - Gaussian process regression with neural kernel captures nonlinearities without overfitting
  - Captures uncertainty information
- Experiments demonstrates POND's excellent performance
  - Non-linear methods greatly outperform multilinear CP and Tucker decompositions
  - POND performs as well as or better than CostCo but also incorporates uncertainty information

Thank You!

THE UNIVERSITY OF UTAH