Tractable Computation of Expected Kernels

Wenze Li*[1] **Zhe Zeng*[2]** **Antonio Vergari[2]** **Guy Van den Broeck[2]**
[1]Tsinghua University [2]University of California, Los Angeles

Motivation

Computation of Expected Kernels is omnipresent in kernel-based frameworks.

$$\mathbb{E}_{\mathbf{x}\sim p,\mathbf{x}'\sim q}[k(\mathbf{x},\mathbf{x}')] = \int_{\mathbf{x},\mathbf{x}'} p(\mathbf{x})q(\mathbf{x}')k(\mathbf{x},\mathbf{x}')d\mathbf{x}d\mathbf{x}'$$

Examples

- Squared MMD

$$\mathbb{E}_{\mathbf{x}\sim p,\mathbf{x}'\sim p}[k(\mathbf{x},\mathbf{x}')] + \mathbb{E}_{\mathbf{x}\sim q,\mathbf{x}'\sim q}[k(\mathbf{x},\mathbf{x}')] - 2\mathbb{E}_{\mathbf{x}\sim p,\mathbf{x}'\sim q}[k(\mathbf{x},\mathbf{x}')]$$

- Kernelized Discrete Stein Discrepancy

$$\mathbb{E}_{\mathbf{x}\sim q,\mathbf{x}'\sim q}[k_p(\mathbf{x},\mathbf{x}')]$$

- Support Vector Regression for Missing Data

$$\mathbb{E}_{\mathbf{x}\sim p}\left[\Sigma_i w_i k(\mathbf{x}^{(i)},\mathbf{x}) + \boldsymbol{b}\right]$$

Probabilistic Circuits and Kernel Circuits

We consider the circuit representation:

$$f_n(\mathbf{X}) = \begin{cases} l_n(\phi(n)) & \textit{if } n \textit{ is an input unit} \\ \Pi_{c\in \mathrm{in}(n)} f_c(\mathbf{X}) & \textit{if } n \textit{ is a product unit} \\ \Sigma_{c\in \mathrm{in}(n)} \theta_c f_c(\mathbf{X}) & \textit{if } n \textit{ is a sum unit} \end{cases}$$

For ***probabilistic circuits (PC)***:

- *A PC on domain $\mathfrak{X}$ is a circuit encoding a non-negative function p: $\mathfrak{X} \to \mathbb{R}^{\geq 0}$.*

For ***kernel circuits (KC)***:

- *A KC on domain $\mathfrak{X} \times \mathfrak{X}$ is a circuit encoding a symmetric kernel function k: $\mathfrak{X} \times \mathfrak{X} \to \mathbb{R}^{+}$.*

Structured properties for tractable computation:

- ***Decomposable***
  all inputs of product units depend on disjoint sets of variables
- ***Smooth***
  all inputs of sum units depend on the same variable sets
- ***Compatible***
  KCs and PCs are smooth and decompose in the same way

Complexity Results

In general, computing expected kernels is not tractable.

We consider ***expressive models*** represented as ***probabilistic circuits*** [1]:

- *p and q are **decomposable** and **smooth** probabilistic circuits* ⇒ *proved #P-Hard!* ☹
- *p and q are **compatible** probabilistic circuits* ⇒ *proved #P-Hard!* ☹
- *p and q are **compatible** probabilistic circuits, k is a kernel circuit **compatible** with p and q* ⇒ *polytime algorithm!* ☺

References

[1] Choi, YooJung, Antonio Vergari, and Guy Van den Broeck. "Probabilistic Circuits: A Unifying Framework for Tractable Probabilistic Models."

[2] Liu, Qiang, and Jason Lee. "Black-box importance sampling." Artificial Intelligence and Statistics. PMLR, 2017.

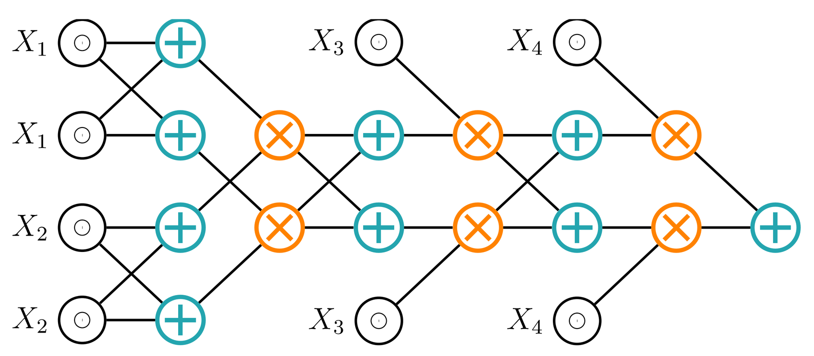
Recursive Computation of Expected Kernels

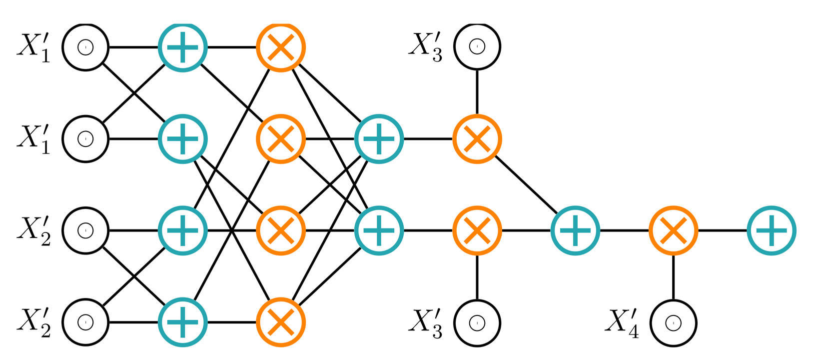
[Sum nodes]

$$p(\mathbf{X}) = \Sigma_i w_i p_i(\mathbf{X}), q(\mathbf{X}') = \Sigma_j w_j' q_j(\mathbf{X}')$$
$$k(\mathbf{X},\mathbf{X}') = \Sigma_l w_l'' k_l(\mathbf{X},\mathbf{X}')$$
$$M_k(p,q) = \Sigma_{i,j,l} w_i w_j' w_l'' M_{k_l}(p_i,q_j)$$

[Product nodes]

$$p(\mathbf{X}) = \Pi_i p_i(\mathbf{X}_i), q(\mathbf{X}') = \Pi_i q_i(\mathbf{X}_i')$$
$$k(\mathbf{X},\mathbf{X}') = \Pi_i k_i(\mathbf{X}_i,\mathbf{X}_i')$$
$$M_k(p,q) = \Pi_i M_{k_i}(p_i,q_i)$$

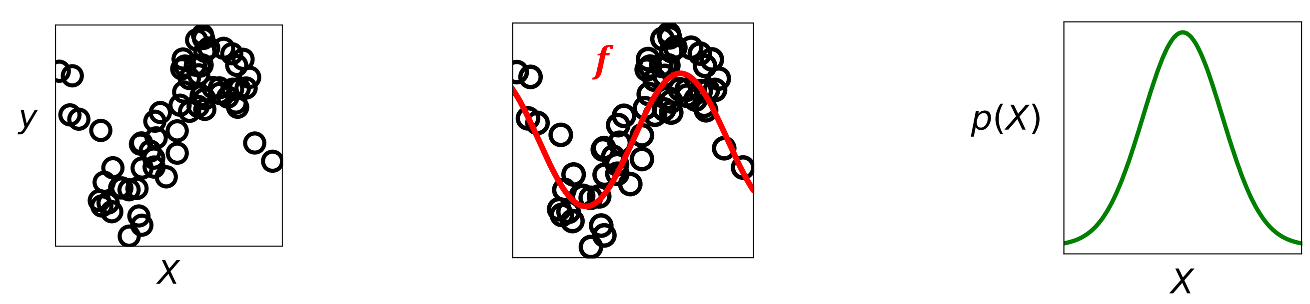




The expected kernels are computed ***exactly*** in $\mathcal{O}(|p||q||k|)$.

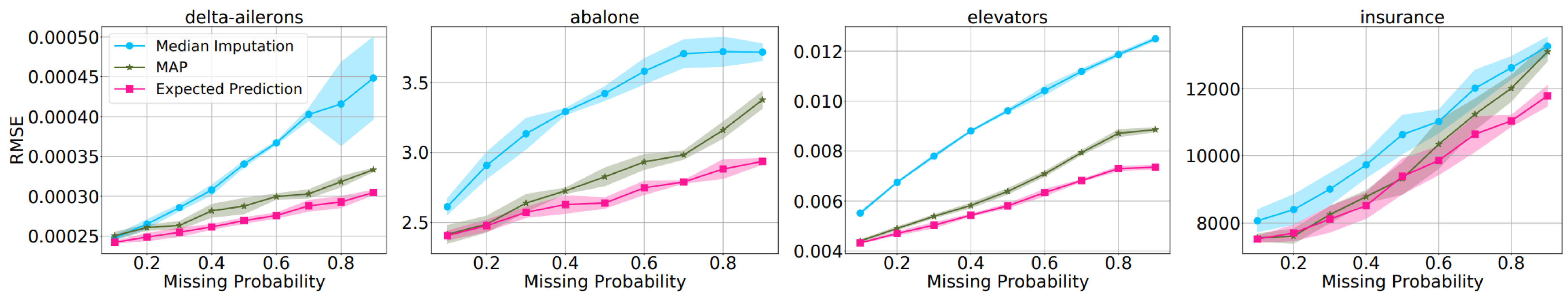
SVR with Missing Data

Given an SVR predictive model f with partial evidence $\mathbf{x}_s$,



we want to compute:

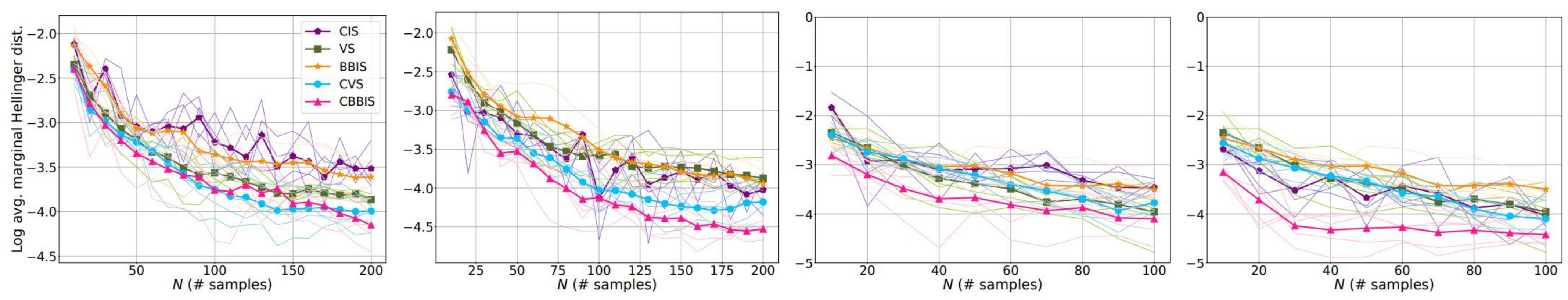
$$\mathbb{E}_{\mathbf{x}_c\sim p(\mathbf{X}_c|\mathbf{x}_s)}[f(\mathbf{x}_s,\mathbf{x}_c)] = \Sigma_i w_i \mathbb{E}_{\mathbf{x}_c\sim p(\mathbf{X}_c|\mathbf{x}_s)}\left[k(\mathbf{x},\mathbf{x}^{(i)})\right] + \boldsymbol{b}$$



Collapsed Black-box Importance Sampling

Collapsed Black-box importance sampling (CBBIS) is a collapsed variant of BBIS [2], which minimizes the conditional Kernelized Discrete Stein Discrepancy:

$$\mathbb{S}_s(q_s\|p) = \mathbb{E}_{\mathbf{x}_s,\mathbf{x}_s'\sim q_s(\mathbf{X}_s)}\left[\mathbb{E}_{\mathbf{x}_c\sim p(\mathbf{X}_c|\mathbf{x}_s),\mathbf{x}_c'\sim p(\mathbf{X}_c|\mathbf{x}_s')}[k_p(\mathbf{x},\mathbf{x}')]\right]$$



Code available: https://github.com/UCLA-StarAI/ExpectedKernels **UAI 2021 – July 2021, Online**