Tractable Computation of Expected Kernels

Wenzhe Li[*1] **Zhe Zeng**[*2] **Antonio Vergari**[2] **Guy Van den Broeck**[2]

[1]Tsinghua University
[2]University of California, Los Angeles

scott.wenzhe.li@gmail.com, {zhezeng, aver, guyvdb}@cs.ucla.edu

Abstract

Computing the expectation of kernel functions is a ubiquitous task in machine learning, with applications from classical support vector machines to exploiting kernel embeddings of distributions in probabilistic modeling, statistical inference, causal discovery, and deep learning. In all these scenarios, we tend to resort to Monte Carlo estimates as expectations of kernels are intractable in general. In this work, we characterize the conditions under which we can compute expected kernels exactly and efficiently, by leveraging recent advances in probabilistic circuit representations. We first construct a circuit representation for kernels and propose an approach to such tractable computation. We then demonstrate possible advancements for kernel embedding frameworks by exploiting tractable expected kernels to derive new algorithms for two challenging scenarios: 1) reasoning under missing data with kernel support vector regressors; 2) devising a collapsed black-box importance sampling scheme. Finally, we empirically evaluate both algorithms and show that they outperform standard baselines on a variety of datasets.

1 INTRODUCTION

Kernel functions have been prominent in the machine learning community for decades. Kernels provided a convenient notion of inner product for high-dimensional feature maps [Cortes and Vapnik, 1995, Schölkopf et al., 1998] and have been extended to represent distributions as elements in a reproducing kernel Hilbert space (RKHS). They have contributed to various fundamental tasks including sample testing [Gretton et al., 2012, Jitkrittum et al., 2017], group anomaly detection [Muandet and Schölkopf, 2013] and causal discovery [Chen et al., 2014].

One fundamental computation that naturally arises in these kernel-embedding based frameworks is to compute the expectations of a kernel function w.r.t. distributions over its inputs. For instance, it arises in integral probability metrics (IPMs) [Müller, 1997] when the functional space is chosen as an RKHS and distributions are characterized by their kernel embeddings. However, such expectations are computationally hard in general and most existing methods resort to Monte Carlo estimators for approximation.

In this paper, we investigate how to derive a tractable algorithm to compute these kernel expectations, thus enabling the aforementioned frameworks to perform exact inference without relying on unreliable approximations. We do so by leveraging recent advances in tractable probabilistic modeling. Specifically, our algorithmic contribution will take advantage of representing both the kernels and the input distributions participating in the expectation as *circuits*.

Circuit representations [Vergari et al., 2019, Choi et al., 2020] reconcile and abstract from the different graphical and syntactic representations of both classical tractable probabilistic models such as mixture models (e.g., mixtures of Gaussian distributions), bounded-treewidth graphical models [Koller and Friedman, 2009, Meila and Jordan, 2000] and more recent ones such as probabilistic circuits [Choi et al., 2020, Vergari et al., 2021] like arithmetic circuits [Darwiche, 2003], probabilistic sentential decision diagrams (PSDDs) [Kisa et al., 2014], sum-product networks (SPNs) [Poon and Domingos, 2011], and cutset networks [Rahman et al., 2014]. As such, our analysis within the framework of circuit representations will help trace the boundaries of tractable computations of kernel expectations, delivering a general and efficient scheme that can be flexibly applied to many kernel-embedding scenarios and different tractable probabilistic model formalisms.

For this representation language, we characterize under which structural constraints on kernel functions and proba-

[*]Authors contributed equally. This research was performed while W.L. was visiting UCLA remotely.

Accepted for the 37th Conference on Uncertainty in Artificial Intelligence (UAI 2021).

bility distributions the expectations of kernels can be computed exactly and efficiently. We show how kernel functions can be represented as circuits with the requisite structural properties, and construct a recursive algorithm that delivers the tractable computation of their expectation in time polynomial in the size of the circuit representations.

Moreover, we demonstrate how the tractable computation of expected kernels can serve as a powerful tool to derive novel kernel-based algorithms on two challenging tasks when using kernel embeddings to represent features as well as distributions. The first is to enable kernel support vector regressors to deal with missing data by computing their expected predictions [Anderson and Gupta, 2011, Khosravi et al., 2019a]. In the second, we derive a novel collapsed black-box importance sampling scheme using the kernelized Stein discrepancy [Liu and Lee, 2017] for efficient approximate inference over factor graph models that do not have a tractable representation. We compare each algorithm with existing baselines on different real-world datasets and problems, showing that our exact expected kernels yield better inference performance.

2 EXPECTED KERNELS

We use uppercase letters X for random variables and lowercase letters x for their assignments. Analogously, we denote a set of random variables in bold uppercase $\mathbf{X}$ and their assignments in bold lowercase $\mathbf{x}$. The domain of variables $\mathbf{X}$ is denoted by $\mathcal{X}$. The cardinality of $\mathcal{X}$ is denoted by $|\mathcal{X}|$.

We are interested in the modular operation of computing expected kernels. This task naturally arises in various kernel-embedding based frameworks.

Definition 2.1 (Expected Kernel)**.** *Given two distributions p and q over variables $\mathbf{X}$ on domain $\mathcal{X}$, and a positive definite kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, the expected kernel, that is, the expectation of the kernel function k with respect to the distributions p and q is defined as follows.*

$$M_k(p, q) := \mathbb{E}_{\mathbf{x}\sim p, \mathbf{x}'\sim q}[k(\mathbf{x}, \mathbf{x}')] \tag{1}$$

Expected kernels are omnipresent in machine learning. For instance, one of the most well-known IPMs, the squared maximum mean discrepancy (MMD) [Gretton et al., 2012] is defined as $MMD^2[\mathcal{H}, p, q] = M_k(p, p) + M_k(q, q) - 2M_k(p, q)$ and measures the distance between two distributions p and q whose embeddings via a kernel k live in a RKHS $\mathcal{H}$. However, the computation cost of expected kernels is prohibitive in general, even for distributions that are tractable for other inference scenarios, as the next theorem illustrates.

Theorem 2.2. *There exist representations of distributions p and q that are tractable for computing marginal, conditional, and maximum a-posteriori (MAP) probabilities, yet computing the expected kernel of a simple kernel k that is the Kronecker delta is already #P-hard.*

Concretely, we show that this is true for probabilistic circuit representations, which unify several tractable probabilistic model representations. We defer the proof of the above statement to Section 4 after circuits are introduced.

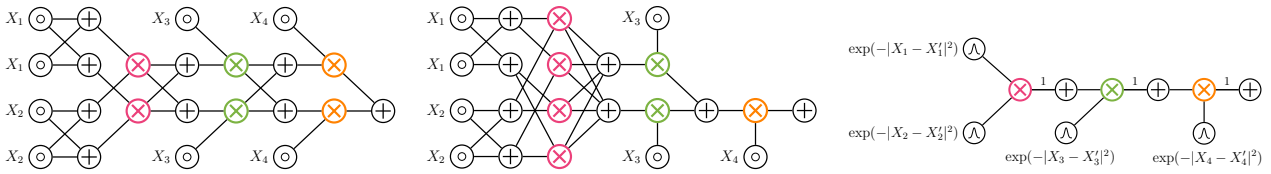
The most commonly adopted solution to estimating Equation 1 and circumventing its computational challenge is to approximate it by sampling. Instead, we are interested in defining a large model class guaranteeing its tractable computation and thus providing an efficient algorithm to compute it exactly. We will show that this is possible by leveraging circuit representations of functions. In summary, we first adopt the probabilistic circuit representations for distributions, and further build a circuit representation for kernel functions to allow an exact computation of the expected kernels to be described in circuit operations. Then, we exploit the structural constraints on circuits such that the computational complexity can be bounded to be polytime in the size of circuits. The necessary background on circuits is presented in Section 3 and the tractable computation of expected kernels is demonstrated in Section 4.

Expected Kernels in Action Our proposed tractable computation of expected kernel can be applied to expressive distribution families and it can potentially lead to new advances in kernel-based frameworks. To demonstrate this, we show how tractable expected kernels give rise to novel algorithms for two challenging tasks, where the kernels serve as *embeddings for features* in one algorithm, and as *embeddings for distributions* in the other, covering the two most popular usages of kernel functions. The first one is to reason about kernel-based support regression models in the presence of missing features. The second one is to perform black-box importance sampling with collapsed samples, where expected kernels are leveraged to obtain the kernelized discrepancy between collapsed samples, which further gives the optimal importance weights. We will show the detailed descriptions of the proposed algorithms in Section 5 and their empirical evaluation in Section 7.

3 CIRCUIT REPRESENTATION

Circuits are parameterized representations of functions as computational graphs. They provide a language to characterize the tractability of function operations in terms of structural constraints over these computational graphs. Next we first introduce circuits and their properties.

Definition 3.1 (Circuit)**.** *A circuit f over variables $\mathbf{X}$ is a parameterized computational graph encoding a function $f(\mathbf{X})$ and comprising three kinds of computational units: input, product, and sum. Each inner unit n (i.e., product or sum unit) receives inputs from some other units, denoted*



(a) a structure-decomposable PC (b) a PC compatible with the PC in (a) (c) a KC kernel-compatible with (a) and (b)

Figure 1: *Examples of circuit representations.* Units in the computational graph include sum units, product units, univariate input distribution units represented with a circle and labeled by their scopes, and non-linear input function units represented with a curve and labeled by the input functions. Sum parameters are omitted for visual clarity. The feed-forward evaluation (input before outputs) is intended from left to right. The rightmost unit is the output of the circuit. All product nodes are colored according to their scopes: $\{X_1, X_2\}$ in pink, $\{X_1, X_2, X_3\}$ in green, and $\mathbf{X}$ in orange.

$\mathsf{in}(n)$. *Each unit n encodes a function f_n as follows:*

$$f_n(\phi(n)) = \begin{cases} l_n(\phi(n)) & \text{if } n \text{ is an input unit} \\ \prod_{c \in \mathsf{in}(n)} f_c(\phi(c)) & \text{if } n \text{ is a product unit} \\ \sum_{c \in \mathsf{in}(n)} \theta_c f_c(\phi(c)) & \text{if } n \text{ is a sum unit} \end{cases}$$

where $\theta_c \in \mathbb{R}$ are the parameters associated with each sum node, and input units encode parameterized functions l_n over variables $\phi(n) \subseteq \mathbf{X}$, also called their scope. The scope of an inner unit is the union of the scopes of its inputs: $\phi(n) = \bigcup_{c \in \mathsf{in}(n)} \phi(c)$. The final output unit (the root of the circuit) encodes $f(\mathbf{X})$.

Circuits can be understood as compact representations of polynomials, whose indeterminates are the functions encoded by the input units. They are assumed to be simple enough to allow locally tractable computations which further forms global operations with tractability guarantees.

Most well-known circuit classes are various forms of probabilistic circuits (PCs) [Vergari et al., 2019, Choi et al., 2020]. PCs provide a unified framework where probabilistic inference operations are cleanly mapped to the circuit representations. As such, they abstract from the many graphical formalism for tractable probabilistic models, from classical shallow mixtures [Koller and Friedman, 2009, Meila and Jordan, 2000] to more recent deep variants [Poon and Domingos, 2011, Peharz et al., 2020]. Specifically, a PC encodes a (possibly unnormalized) probability distribution over a collection of variables in a recursive manner.

Definition 3.2 (Probabilistic Circuits)**.** *A PC on domain $\mathcal{X}$ is a circuit encoding a non-negative function $p : \mathcal{X} \to \mathbb{R}^{\geq 0}$.*

A circuit p can be evaluated in time linear in its size denoted by $|p|$, i.e., the number of edges in its computational graph. For example, computing $p(\mathbf{X} = \mathbf{x})$ in a PC can be done in a *feedforward* way, evaluating input units before outputs, and hence in time linear in the size of the PC.

W.l.o.g., we will assume that units in circuits alternate layer-wise between sum and product units and that every product unit receives only two inputs. Both requirements can be easily enforced in any circuit structure with a polynomial increase in its size [Peharz et al., 2020, Vergari et al., 2015]. Furthermore, in this work we focus on discrete variables. For conciseness, we denote the circuit by the same notation as the function that it represents, for instance, a PC p refers to the circuit representation of the distribution p.

Properties of Circuits The tractability of computing quantities of interest involving the function encoded in a circuit, also called queries, can be characterized by *structural constraints* on the computational graph of its circuit [Darwiche and Marquis, 2002]. Next we introduce the structural properties that will be sufficient for the tractable computation of the expected kernels. We refer the interested reader to Choi et al. [2020] for additional properties enabling other tractable inference scenarios.

Definition 3.3 (Smoothness)**.** *A circuit is smooth, if for every sum node n, its inputs $\mathsf{in}(n)$ share the same scope, i.e., $\forall c, c' \in \mathsf{in}(n), \phi(c) = \phi(c')$.*

Some examples of smooth circuits are mixture models: they comprise a single sum node over tractable input distributions that have to share the same scope. For example, a Gaussian mixture model (GMM) can be represented as a smooth circuit with a single sum unit and several input units, each of which encodes a (multivariate) Gaussian density defined over the same set of variables.

Definition 3.4 (Determinism)**.** *A circuit is deterministic if the inputs of every sum unit have disjoint supports.*

Determinism in PCs enables the tractable computation of MAP inference. In this work, determinism will play a role in exactly computing the KSD between discrete distributions (see Corollary 4.7).

Definition 3.5 (Decomposability)**.** *A circuit is decomposable, if for every product node n, its inputs $\mathsf{in}(n)$ have disjoint scopes, i.e., $\forall c, c' \in \mathsf{in}(n), c \neq c' : \phi(c) \cap \phi(c') = \emptyset$.*

Decomposable product nodes encode local factorizations. For example, a decomposable product node n over variables $\mathbf{X}$ with inputs from two units can be written as

$f_n(\mathbf{X}) = f_{\mathsf{L}}(\mathbf{X}_{\mathsf{L}})f_{\mathsf{R}}(\mathbf{X}_{\mathsf{R}})$, where $\mathbf{X}_{\mathsf{L}}$ and $\mathbf{X}_{\mathsf{R}}$ form a partition of $\mathbf{X}$. Taken together, smoothness and decomposability are sufficient and necessary for performing tractable integration over arbitrary sets of variables in a single feedforward pass, which allows to compute marginals and conditionals in time linear in the circuit size [Choi et al., 2020]. To characterize tractable kernel expectations, we will need the multiple circuits participating in it to have product units that decompose their scopes in a "synchronized" way. This property, called compatibility, is formalized recursively as follows.

Definition 3.6 (Compatibility)**.** *Two circuits f and g are compatible if (i) they are smooth and decomposable, and (ii) for any pair of product units $n \in f$ and $m \in g$ that share the same scope, they decompose in the same way, i.e., for every unit $c \in \mathsf{in}(n)$, there must exist a unique unit $c' \in ch(m)$ such that $\phi(c) = \phi(c')$.*

Definition 3.7 (Structured-decomposability)**.** *A circuit is structured-decomposable if it is compatible with itself.*

Notice that structured-decomposable circuits are a strict subclass of decomposable circuits. An example of a structured-decomposable PC is shown in Figure 1a. The way that a structured-decomposable circuit hierarchically partitions its scope can be compactly represented by a graph called *vtree* [Pipatsrisawat and Darwiche, 2008], *pseudo-forest* [Jaeger, 2004] or *pseudo-tree* [Dechter and Mateescu, 2007]. In a nutshell, compatible structured-decomposable circuits conform to the same hierarchical partitioning over their variables. Figure 1a and Figure 1b show two compatible PCs. This additional requirement enables also the tractable computation of moments of predictive models [Khosravi et al., 2019a] and the probability of logical constraints [Bekker et al., 2015, Choi et al., 2015].

Construction of PCs As mentioned before, several classes of tractable probabilistic graphical models (PGMs) including Chow-Liu trees [Chow and Liu, 1968] and hidden Markov models (HMMs) [Rabiner and Juang, 1986] can be represented as compact PCs with certain structural properties. The process of translating one graphical representation into a circuit is called *compilation* and has received much attention in the literature [Chavira and Darwiche, 2005, Darwiche, 2011]. In particular, Shen et al. [2016] propose a very efficient compilation scheme that compiles a factor graph into a structured-decomposable PC by first representing each factor as a PC and then multiplying them together.

Besides compiling PCs from other tractable models, we can also directly learn PCs from data [Lowd and Domingos, 2012, Rooshenas and Lowd, 2014, Peharz et al., 2020]. Recently learning algorithms tailored towards structured-decomposable PCs have been proposed [Liang and Van den Broeck, 2017, Dang et al., 2020]. For our experiments we will employ STRUDEL [Dang et al., 2020] for its simplicity and speed.

4 TRACTABLE COMPUTATION OF EXPECTED KERNELS

Computing expected kernels is a #P-hard problem in general. It involves summation over exponentially many states in the distribution space. We first provide a formal proof for the hardness statement provided in Theorem 2.2.

Proof. [**Theorem 2.2**] Consider the case when p and q are both structured-decomposable and deterministic probabilistic circuits, and the positive definite kernel k is a Kronecker delta function defined as $k(\mathbf{x}, \mathbf{x}') = 1$ if and only if $\mathbf{x} = \mathbf{x}'$. Then computing the expected kernel $M_k(p, q)$ is equivalent to computing the quantity $\sum_{\mathbf{x}\in\mathcal{X}} p(\mathbf{x})q(\mathbf{x})$, which has been shown to be #P-hard by Vergari et al. [2021]. Therefore, computing the expected kernel is #P-hard. □

From the proof we can tell that mild structural constraints on circuits are not enough to reduce the computational complexity. We provide another proof in Appendix where a pair of probabilistic circuits with different constraints is considered. Together they show that it is highly challenging to derive sufficient structural constraints to guarantee tractability.

The aim of this section is to investigate under what structural constraints on circuits an exact and efficient computation of expected kernels is possible. But before we characterize tractability in the circuit language, we need to consider *whether also kernels can be represented as circuits*. To answer this question we define kernel circuits (KCs) to be the circuit representations of kernel functions that measure similarities between input pairs defined on the kernel domain.

Definition 4.1. *A KC on domain $\mathcal{X}\times\mathcal{X}$ is a circuit encoding a symmetric kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$.*

Remark. To verify that a given KC is positive definite, it is sufficient to verify that the input units are positive definite kernels and that the sum parameters are positive since the positive definite kernel family is closed under summation and product. Moreover, it can be done tractably in time linear in the number of input units in the KC.

Figure 1c shows an example kernel circuit. We further define the left (resp. right) projection of a KC given $\mathbf{x} \in \mathcal{X}$ to be $k(\cdot, \mathbf{x}) : \mathcal{X} \to \mathbb{R}^+$ (resp. $k(\mathbf{x}, \cdot) : \mathcal{X} \to \mathbb{R}^+$). Intuitively, for the tractability of expected kernels, the KC should have its structure conform to the distributions that it measures, which allows the measurement to be broken down into basic ones along the circuit. Next, we characterize the structural constraints on KCs suitable for such a computation.

Definition 4.2 (Kernel Compatibility)**.** *Let p and q be a pair of compatible circuits. A kernel circuit $k(\mathbf{X}, \mathbf{X}')$ is kernel-compatible with the circuit pair $p(\mathbf{X})$ and $q(\mathbf{X}')$ if*

Algorithm 1 $M_{k_l}(p_n, q_m)$ — Computing the expected kernel

Require: Two compatible PCs p_n and q_m, and a KC k_l that is kernel-compatible with the PC pair p_n and q_m.

if n, m, l are input units **then**
  return $M_{k_l}(p_n, q_m)$
else if n, m, l are sum units **then** ▷ cf. Prop. 4.4
  return $\sum_{i\in\mathsf{in}(n), j\in\mathsf{in}(m), c\in\mathsf{in}(l)} \theta_i\delta_j\gamma_c\, M_{k_c}(p_i, q_j)$
else if n, m, l are product units **then** ▷ cf. Prop. 4.5
  return $M_{k_\mathsf{L}}(p_{n_\mathsf{L}}, q_{m_\mathsf{L}}) \cdot M_{k_\mathsf{R}}(p_{n_\mathsf{R}}, q_{m_\mathsf{R}})$

i) the kernel circuit k is smooth and decomposable, and
ii) the left and right projections of k are compatible with circuit p and q respectively for any $\mathbf{x} \in \mathcal{X}$.

For example, the KC shown in Figure 1c is kernel-compatible with the circuit pair shown in Figure 1a and Figure 1b. Intuitively, a KC with kernel compatibility measures the similarity between the two probability distributions in a hierarchical way.

Note that many commonly used kernels have a circuit representations that exhibits kernel compatibility. These include several exponentiated forms such as the radial basis function kernel (RBF) and the exponentiated Hamming kernel. To see how, consider an RBF kernel $k(\mathbf{X}, \mathbf{X}') = \exp(-\sum_{i=1}^{4} |X_i - X_i'|^2)$. It can be represented by a KC with one product unit connected to four input units each of which represents the basic function $\exp(-|X_i - X_i'|^2)$. Given a pair of compatible PCs p and q as in Figure 1a and Figure 1b, we can always transform the KC of an RBF kernel into a circuit compatible with p and q by "splitting" its product unit into intermediate products that are compatible with the product units in p and q and by introducing dummy sum units receiving single inputs and with parameter $\theta = 1$. The resulting KC is shown in Figure 1c.

Next we show our main result: kernel compatibility is sufficient to guarantee the tractability of expected kernels.

Theorem 4.3. *Let p and q be a pair of compatible PCs, and k be a kernel circuit. If k is kernel-compatible with p and q, the expected kernel $M_k(p, q)$ can be computed exactly in $\mathcal{O}(|p||q||k|)$ time.*[1]

The proof is by construction. Intuitively, the computation of expected kernels can be recursively "broken down" along the circuit structures, until we reach collections of input units for which we can assume the integrals in the expectations to be tractably computed. The next proposition shows this recursion over circuits whose outputs are sums.

[1]As the algorithm will show, this is not a tight bound and in practice the effective number of recursive calls will be much smaller than $|p||q||k|$.

Proposition 4.4. *Let p_n and q_m be two smooth probabilistic circuits over variables $\mathbf{X}$ whose output units n and m are sum units, denoted by $p_n(\mathbf{X}) = \sum_{i\in\mathsf{in}(n)} \theta_i p_i(\mathbf{X})$ and $q_m(\mathbf{X}) = \sum_{j\in\mathsf{in}(m)} \delta_j q_j(\mathbf{X})$ respectively. Let k_l be a kernel circuit with its output unit being a sum unit l, denoted by $k_l(\mathbf{X}) = \sum_{c\in\mathsf{in}(l)} \gamma_c k_c(\mathbf{X})$. Then it holds that*

$$M_{k_l}(p_n, q_m) = \sum_{i\in\mathsf{in}(n)} \theta_i \sum_{j\in\mathsf{in}(m)} \delta_j \sum_{c\in\mathsf{in}(l)} \gamma_c\, M_{k_c}(p_i, q_j). \tag{2}$$

This way, the expected kernel can be computed by the weighted sum of a number of simpler expected kernel computations over the input units. Analogously, the expected kernel computation can be broken down at the product units as follows thanks to compatibility.

Proposition 4.5. *Let p_n and q_m be two compatible probabilistic circuits over variables $\mathbf{X}$ whose output units n and m are product units, denoted by $p_n(\mathbf{X}) = p_{n_\mathsf{L}}(\mathbf{X}_\mathsf{L})p_{n_\mathsf{R}}(\mathbf{X}_\mathsf{R})$ and $q_m(\mathbf{X}) = q_{m_\mathsf{L}}(\mathbf{X}_\mathsf{L})q_{m_\mathsf{R}}(\mathbf{X}_\mathsf{R})$. Let k_l be a kernel circuit that is kernel-compatible with the circuit pair p_n and q_m with its output unit being a product unit denoted by $k_l(\mathbf{X}, \mathbf{X}') = k_\mathsf{L}(\mathbf{X}_\mathsf{L}, \mathbf{X}'_\mathsf{L})k_\mathsf{R}(\mathbf{X}_\mathsf{R}, \mathbf{X}'_\mathsf{R})$. Then it holds that*

$$M_{k_l}(p_n, q_m) = M_{k_\mathsf{L}}(p_{n_\mathsf{L}}, q_{m_\mathsf{L}}) \cdot M_{k_\mathsf{R}}(p_{n_\mathsf{R}}, q_{m_\mathsf{R}}).$$

Lastly, for the base cases of the recursion we can have that either both p and q comprise a single input distribution (sharing the same scope), or one of them is an input distribution and the other a sum unit.[2] The first case is easily computable in polytime by the assumption in Theorem 4.3. Note that this assumption is generally easy to meet as the double summation in $M_k(p_n, q_m)$ for input distributions can be computed in polytime by enumeration, since input distributions have limited scopes (generally univariate) and $p(\mathbf{x})q(\mathbf{x}')k(\mathbf{x}, \mathbf{x}')$ can be computed in closed form for decomposable kernels k and commonly used distributions such as discrete distributions as in our case. The second corner case reduces to the first when noting that computing $M_k(p_n, q_m)$ for an input distribution and a mixture of input distributions reduces to computing a weighted sum of expectations followed by applying Proposition 4.4. Algorithm 1 summarizes the whole computation of the expected kernel M_k, which requires only polynomial complexity when caching repeated calls.

As direct results of Theorem 4.3, we show that two common kernelized discrepancies in reproducing kernel Hilbert space (RKHS) can be tractably computed if the same structural constraints apply to the distributions and kernels.

Corollary 4.6. *Following the assumptions in Theorem 4.3, the squared maximum mean discrepancy $MMD[\mathcal{H}, p, q]$ in*

[2]The other unit cannot be a product unit otherwise compatibility would be violated.

RKHS $\mathcal{H}$ associated with kernel k as defined in Gretton et al. [2012] can be tractably computed in time $\mathcal{O}(|p||q||k|)$.

Corollary 4.7. *Following the assumptions in Theorem 4.3, if the probabilistic circuit p further satisfies determinism, the kernelized discrete Stein discrepancy (KDSD) $\mathbb{D}^2(q \parallel p) = \mathbb{E}_{\mathbf{x},\mathbf{x}'\sim q}[k_p(\mathbf{x},\mathbf{x}')]$ in the RKHS associated with kernel k as defined in Yang et al. [2018] can be tractably computed.*

The computation of expected kernels by circuit operations allows us to compute the kernel-embedding based statistics exactly and efficiently. This further gives rise to interesting applications part of which will be shown in the next section. We leave the further explorations on what other statistics will benefit from the proposed computation of expected kernels and what more applications will be inspired as future work.

5 EXPECTED KERNELS IN ACTION

In this section we will show how the tractable computation of expected kernels can be leveraged in 1) *kernel embedding for features* to derive an inference algorithm for support vector regression (SVR) under missing data; 2) *kernel embedding for distributions* to derive a collapsed estimator in black-box importance sampling (IS). We further demonstrate the effectiveness of both proposed expected-kernel based algorithms empirically in Section 7.

5.1 SVR FOR MISSING DATA

Support vector machines (SVMs) for classification and regression are widely used in machine learning [Noble, 2006]. SVMs' foundations have great theoretical appeal, and they are still widely used in practice. How to deal with missing features in SVMs has been an active area of research [Aydilek and Arslan, 2013, Saar-Tsechansky and Provost, 2007, Marlin, 2008].

In this section, we aim to tackle missing features in SVR at deployment time from a principled probabilistic perspective, like in Anderson and Gupta [2011], but for a larger model class represented as circuits. We propose to leverage PCs to learn the joint feature distribution, and then exploit tractable expected kernels to efficiently compute the expected predictions of SVR models. More formally, given a set of input variables $\mathbf{X}$ (features) with domain $\mathcal{X}$ and a variable Y (target) with domain $\mathcal{Y}$, and a kernel function k, a kernelized SVR learns from a dataset $\{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^n$ to predict for new inputs with a function f taking the form

$$f(\mathbf{X}) = \sum_{i=1}^{n} w_i k(\mathbf{x}^{(i)}, \mathbf{X}) + b. \tag{3}$$

Existing works to handle missing features at deployment time include imputation strategies that substitutes missing values with reasonable alternatives such as the mean or median, estimated from training data. The imputation methods are typically heuristic and model-agnostic, and sometimes make strong distributional assumptions such as total independence of the feature variables. As demonstrated in Khosravi et al. [2019b], computing expected predictions is not only theoretically principled but practically effective.

Definition 5.1 (Expected prediction)**.** *Given a predictive model $f : \mathcal{X} \to \mathcal{Y}$, a distribution $p(\mathbf{X})$ over features $\mathbf{X}$ and a partial assignment $\mathbf{x_s}$ for variables $\mathbf{X_s} \subset \mathbf{X}$, the expected prediction of f w.r.t. p is*

$$\mathbb{E}_{\mathbf{x_c}\sim p(\mathbf{X_c}|\mathbf{x_s})}[f(\mathbf{x})], \tag{4}$$

where $\mathbf{X_c} = \mathbf{X}\backslash\mathbf{X_s}$ and where $\mathbf{x}$ is the completed feature vector consisting of both $\mathbf{x_c}$ and $\mathbf{x_s}$.

Intuitively, the expected prediction of a SVR given a partial feature vector can be thought of as reweighting all possible completions by their probability. Expected prediction enjoys the theoretical guarantee that it is consistent under both missing completely at random (MCAR) and missing at random (MAR) mechanisms, if f has been trained on complete data and is Bayes optimal [Josse et al., 2019].

Proposition 5.2. *Given a SVR model f with a KC k, and a structured-decomposable PC p for the feature distribution, the expected prediction of f can be tractably computed in time $\mathcal{O}(|k||p|)$.*

Proof. The expected prediction of f w.r.t. p can be rewritten as a linear combination of expected kernels.

$$\mathbb{E}_{\mathbf{x_c}\sim p(\mathbf{X_c}|\mathbf{x_s})}[f(\mathbf{x})] = \sum_{i=1}^{n} w_i \mathbb{E}_{\mathbf{x_c}\sim p(\mathbf{X_c}|\mathbf{x_s})}[k(\mathbf{x}, \mathbf{x}^{(i)})] + b.$$

Note that the task of computing the doubly expected kernel in Definition 2.1 subsumes the task of computing a singly expected kernel where one of the inputs to the kernel function is a constant vector $\mathbf{x}_i$ instead of a variable and both Theorem 4.3 and Algorithm 1 apply here. □

5.2 COLLAPSED BLACK-BOX IMPORTANCE SAMPLING

Black-box importance sampling (BBIS) [Liu and Lee, 2017] is a recently introduced algorithm to flexibly perform approximate probabilistic inference on intractable distributions. By weighting samples from an arbitrary proposal as to minimize a kernelized Stein discrepancy (KSD), BBIS can accurately estimate continuous target distributions.

In this section, we first show that the BBIS algorithm can be extended to discrete distributions by adopting a recently proposed kernelized discrete Stein discrepancy (KDSD) [Yang et al., 2018] that serves as the discrete counterpart for KSD.

We further show that the BBIS algorithm can be improved by using collapsed samples, which is made possible by the tractable computation of expected kernels.

We start with a brief overview of how to construct the KDSD. For a finite domain $\mathcal{X}$, a *cyclic permutation* denoted by $\neg$ is a bijection associated with some ordering of elements in $\mathcal{X}$ that maps an element in $\mathcal{X}$ to the next one according to the ordering. A *partial difference operator* Δ^* for any function f on domain $\mathcal{X}$ is defined as $\Delta^* f(\mathbf{x}) := (\Delta_1^* f(\mathbf{x}), \cdots, \Delta_D^* f(\mathbf{x}))$, with $\Delta_i^* f(\mathbf{x}) := f(\mathbf{x}) - f(\neg_i \mathbf{x})$ for $i = 1, 2, \cdots, D$ with $D = |\mathbf{X}|$. Now we are ready to define the (difference) score function, an important tool for determining a probability distribution. The score function is defined as $\boldsymbol{s}_p(\mathbf{x}) := \Delta^* p(\mathbf{x})/p(\mathbf{x})$, a vector-valued function with its i-th dimension being $\boldsymbol{s}_{p,i}(\mathbf{x}) := \Delta_i^* p(\mathbf{x})/p(\mathbf{x})$. Then the KDSD between two distributions p and q is defined as

$$\mathbb{D}(q \parallel p) := \sup_{\boldsymbol{f} \in \mathcal{F}} \mathbb{E}_{\mathbf{x} \sim q(\mathbf{X})}[\mathcal{T}_p \boldsymbol{f}(\mathbf{x})], \tag{5}$$

with the functional space $\mathcal{F}$ being RKHS associated with a strictly positive definite kernel k, and the operator $\mathcal{T}_p$ being the *Stein difference operator* defined as $\mathcal{T}_p \boldsymbol{f} := \boldsymbol{s}_p(\mathbf{x}) \boldsymbol{f}^\top - \Delta \boldsymbol{f}(\mathbf{x})$. The KDSD is a proper divergence measure in the sense that for any strictly positive distribution p and q, the KDSD $\mathbb{D}(q \parallel p) = 0$ if and only if $p = q$ [Yang et al., 2018]. Moreover, a nice property of the KDSD is that even though it involves a variational optimization problem in its definition, it admits a closed-form representation as

$$\mathbb{S}(q \parallel p) := \mathbb{D}^2(q \parallel p) = \mathbb{E}_{\mathbf{x}, \mathbf{x}' \sim q}[k_p(\mathbf{x}, \mathbf{x}')], \tag{6}$$

with the kernel function k_p defined as

$$\begin{aligned} k_p(\mathbf{x}, \mathbf{x}') = \boldsymbol{s}_p(\mathbf{x})^\top k(\mathbf{x}, \mathbf{x}') \boldsymbol{s}_p(\mathbf{x}') - \boldsymbol{s}_p(\mathbf{x})^\top \Delta^{\mathbf{x}'} k(\mathbf{x}, \mathbf{x}') \\ - \Delta^{\mathbf{x}} k(\mathbf{x}, \mathbf{x}')^\top \boldsymbol{s}_p(\mathbf{x}') + tr(\Delta^{\mathbf{x}, \mathbf{x}'} k(\mathbf{x}, \mathbf{x}')), \end{aligned}$$

where the superscript $\mathbf{x}$ and $\mathbf{x}'$ of the difference operator specifies the variables that it operates on.

We can now proceed to propose a BBIS algorithm for categorical distributions. Given a set of samples $\{\mathbf{x}^{(i)}\}_{i=1}^n$ generated from some unknown proposal q possibly from some black-box mechanism, Categorical BBIS computes the importance weights for the samples by minimizing the KDSD between q and target distribution p formulated as

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \left\{ \boldsymbol{w}^\top \boldsymbol{K}_{\boldsymbol{p}} \boldsymbol{w} \,\middle|\, \sum_{i=1}^n w_i = 1,\ w_i \geq 0 \right\}, \tag{7}$$

where $\boldsymbol{K}_p$ is a Gram matrix with entries $[\boldsymbol{K}_p]_{ij} = k_p(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})$ and $\boldsymbol{w} = (w_1, \cdots, w_n)$ is the weight vector. We prove that the BBIS for categorical distributions enjoys the same convergence guarantees as its continuous counterpart. Due to space constraints, we defer both the algorithm details and convergence proofs to the Appendix.

However, a computational bottleneck in BBIS limits its scalability, the construction of the Gram matrix. We therefore propose a collapsed variant of BBIS to accelerate it by delivering equally good approximations with fewer samples. Collapsed samplers, also known as *cutset* or *Rao-Blackwellised* samplers [Casella and Robert, 1996], improve over classical particle-based methods by limiting sampling to a subset of the variables while pairing it with some closed-form representation of a conditional distribution over the rest.

Specifically, let $(\mathbf{X_s}, \mathbf{X_c})$ be a partition for variables $\mathbf{X}$. A *weighted collapsed sample* for variables $\mathbf{X}$ takes the form of a triplet $(\mathbf{x_s}, p(\mathbf{X_c} \mid \mathbf{x_s}), w)$ where $\mathbf{x_s}$ is an assignment for the sampled variables $\mathbf{X_s}$, $p(\mathbf{X_c} \mid \mathbf{x_s})$ is a conditional distribution over the *collapsed set* $\mathbf{X_c}$, and w the importance weight. We now show how to distill a *conditional* KDSD, in order to extend BBIS to the collapsed sample scenario.

Definition 5.3 (Conditional KDSD)**.** *Assume given a strictly positive distribution p and a strictly positive proposal distribution of the sampled set $q_{\mathbf{s}}$, where the variable subset $\mathbf{X_s}$ defines the samples. The full distribution defined by the collapsed samples is $q(\mathbf{x}) = q_{\mathbf{s}}(\mathbf{x_s}) p(\mathbf{x_c} \mid \mathbf{x_s})$. The conditional KDSD (CKDSD) is defined as the KDSD between distributions p and q, i.e., $\mathbb{S}_{\mathbf{s}}(q_{\mathbf{s}} \parallel p) := \mathbb{S}(q \parallel p)$.*

Proposition 5.4. *The CKDSD between the two positive distributions p and q admits a closed form as*

$$\mathbb{S}_{\mathbf{s}}(q_{\mathbf{s}} \parallel p) = \mathbb{E}_{\mathbf{x_s}, \mathbf{x_s}' \sim q_{\mathbf{s}}(\mathbf{X_s})}[k_{p,\mathbf{s}}(\mathbf{x_s}, \mathbf{x_s}')], \tag{8}$$

where $k_{p,\mathbf{s}}$ denotes a conditional kernel function defined as

$$k_{p,\mathbf{s}}(\mathbf{x_s}, \mathbf{x_s}') = \mathop{\mathbb{E}}_{\mathbf{x_c} \sim p(\mathbf{X_c} \mid \mathbf{x_s}), \mathbf{x_c}' \sim p(\mathbf{X_c} \mid \mathbf{x_s}')} [k_p(\mathbf{x}, \mathbf{x}')]\,. \tag{9}$$

Similar to the optimization in Equation 7 for BBIS, given a set of collapsed samples $\{(\mathbf{x_s}^{(i)}, p(\mathbf{X_c} \mid \mathbf{x_s}^{(i)}))\}_{i=1}^n$, the problem of computing importance weights can be cast as minimizing the empirical CKDSD between the collapsed samples and the target distribution p as follows.

$$\mathbb{S}_{\mathbf{s}}(\{\mathbf{x_s}^{(i)}, w_i\} \parallel p) = \boldsymbol{w}^\top \boldsymbol{K}_{p,\mathbf{s}} \boldsymbol{w} \tag{10}$$

where $\boldsymbol{w}$ is the vector of sample weights and $\boldsymbol{K}_{p,\mathbf{s}}$ is the Gram matrix with entries $[\boldsymbol{K}_{p,\mathbf{s}}]_{ij} = k_{p,\mathbf{s}}(\mathbf{x_s}^{(i)}, \mathbf{x_s}^{(j)})$. Now the key question is whether the conditional kernel function $k_{p,\mathbf{s}}$ can be computed tractably. We show that this is possible with the tractable computation of expected kernels.

Proposition 5.5. *Let $p(\mathbf{X_c} \mid \mathbf{x_s})$ be a PC that encodes a conditional distribution over variables $\mathbf{X_c}$ conditioned on $\mathbf{X_s} = \mathbf{x_s}$, and k be a KC. If the PC $p(\mathbf{X_c} \mid \mathbf{x_s})$ and $p(\mathbf{X_c} \mid \mathbf{x_s}')$ are compatible and k is kernel-compatible with the PC pair for any $\mathbf{x_s}$, $\mathbf{x_s}'$, then the conditional kernel function $k_{p,\mathbf{s}}$ can be tractably computed.*

This finishes the construction of a BBIS scheme using the collapsed samples, which we name CBBIS. The complete algorithmic recipe for CBBIS is presented in Algorithm 2

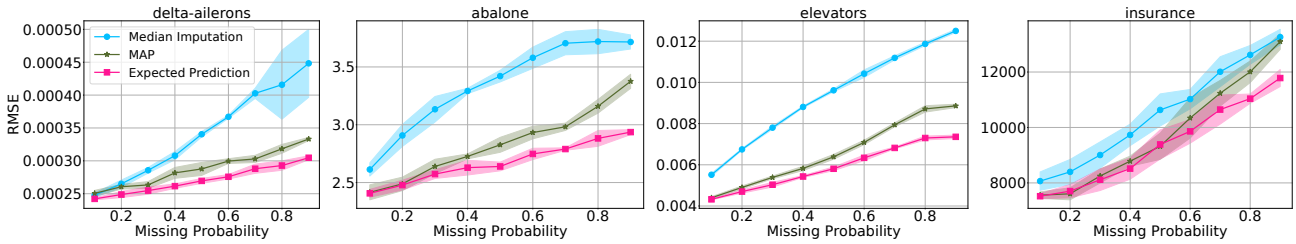


Figure 2: Evaluating RMSE (y-axis) of the predictions of SVR under different percentages of missing features (x-axis) over four real-world regression datasets. Overall, our expected predictions outperform median imputation and MAP.

Algorithm 2 CBBIS$(p, q_{\mathbf{S}}, k, n)$

Input: target distribution p over variables $\mathbf{X}$, black-box mechanism $q_{\mathbf{S}}$, kernel function k, number of samples n
Output: a set of weighted collapsed samples

Sample $\{\mathbf{x_s}^{(i)}\}_{i=1}^{n}$ from $q_{\mathbf{S}}$
for $i = 1, \ldots, n$ **do**
    Compile $p(\mathbf{X_c} \mid \mathbf{x_s}^{(i)})$ into a PC ▷ cf. Sec. 7.2
for $i = 1, \ldots, n$ **do**
    for $j = 1, \ldots, n$ **do**
        $[\boldsymbol{K}_p]_{ij} = k_{p,\mathbf{S}}(\mathbf{x_s}^{(i)}, \mathbf{x_s}^{(j)})$ ▷ cf. Prop. 5.5
$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \left\{ \boldsymbol{w}^\top \boldsymbol{K}_p \boldsymbol{w} \,\middle|\, \sum_{i=1}^{n} w_i = 1,\ w_i \geq 0 \right\}$
return $\{(\mathbf{x_s}^{(n)}, p(\mathbf{X_c} \mid \mathbf{x_s}^{(n)}), w_i^*)\}_{i=1}^{n}$

6 RELATED WORK

The idea of composing kernels with sums and products first emerged in the literature of the automatic statistician, and is applied to structure discovery for Gaussian processes and nonparametric regression tasks [Duvenaud et al., 2013]. Compositional kernel machines [Gens and Domingos, 2017] further leverage sum-product functions [Friesen and Domingos, 2016] for a tractable instance-based method for object recognition. Instead, we provide the general theoretical foundations for the tractable computation of expected kernels.

Our proposed BBIS scheme extends the original black-box importance sampling to discrete domains, which have not been explored yet, contrary to the continuous case [Cockayne et al., 2019, Oates et al., 2014]. Alternatives to black-box optimization include directly approximating the proposal distribution to compute the importance weights [Delyon et al., 2016]. The KSD [Liu and Wang, 2016, Liu et al., 2016] and its variants [Yang et al., 2018, Wang et al., 2019, 2018, Singhal et al., 2019], when applied to particle-based inference, consider the particles to be fully instantiated while our proposed conditional KDSD generalizes it to collapsed particles.

Closely related, works in probabilistic graphical models represent collapsed particles by circuits. The approximate compilation proposed by Friedman and Van den Broeck [2018] employs online collapsed importance sampling (CIS) partially compiling the target distribution into a sentential decision diagram (SDD) [Darwiche, 2011]. Rahman et al. [2019] propose to use a cutset network, a smooth, decomposable and deterministic PC to distill a collapsed Gibbs sampling (CGS) scheme for Bayesian networks. Arithmetic circuits [Darwiche, 2003], other kinds of PCs that can be compiled from Bayesian networks have been used in the context of variational approximations [Lowd and Domingos, 2010, Vlasselaer et al., 2015, Shih and Ermon, 2020].

7 EMPIRICAL EVALUATION

In this section, we empirically evaluate our two novel algorithms, and show how tractable expected kernels can benefit scenarios where the kernels serve as *embedding for features* and *embedding for distributions*.[3] We provide preliminary experiments to answer the following questions: **(Q1)** Do expected predictions at deployment time improve predictions over common imputation techniques to deal with missingness for SVR? **(Q2)** How is the performance of CBBIS when compared to other IS methods? **(Q3)** How much does collapsing more variables improve estimation quality?

7.1 REGRESSION UNDER MISSING DATA

We compare our expected prediction with median imputation techniques and another natural and strong baseline: imputing missing values by MAP inference over the learned data distribution. We evaluate all competitors on four common regression benchmarks from several domains following Khosravi et al. [2019a]. For each benchmark, we adopt the STRUDEL algorithm [Dang et al., 2020] to learn structured-decomposable and deterministic PCs from data to represent the data distributions. STRUDEL initializes from a Chow-Liu tree [Chow and Liu, 1968]. Then the structure learning is performed by doing heuristic-based greedy search over possible structures. Intuitively, it iteratively models the data with variable heuristic and edge heuristic. Recall from Section 3 that deterministic PCs can perform exact MAP inference in polytime and thus the MAP imputation can be done tractably.

[3] Code for reproducing our empirical evaluation can be found at `github.com/UCLA-StarAI/ExpectedKernels`

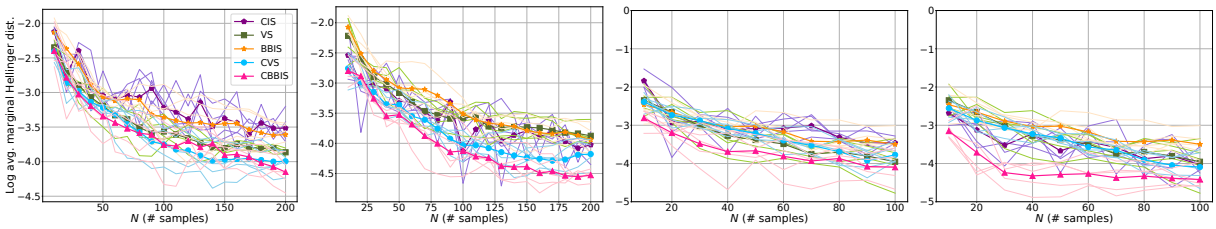


Figure 3: Log average marginal Hellinger distance (y-axis) vs. different sample sizes (N, x-axis), evaluated on an Ising or ASIA model as a target distribution (p) with Gibbs chain as a proposal distribution (q). The target distribution and the percentage of collapsed variables are (from left to right): (Ising, 25%); (Ising, 50%); (ASIA, 25%); (ASIA, 50%).

For the missingness setting, we assume data to be MCAR with missing probability $\pi \in \{0.1, 0.2, \ldots, 0.9\}$, each of which reports the average result over five independent trials. We employ RBF kernels, which are naturally compatible with any structured-decomposable PCs (see Section 4).

Figure 2 summarizes our results: we can answer **Q1** in a positive way since expected prediction performs equally well or better than other imputation methods. This is because expected prediction computes the exact expectation over expressive distributions while other imputation techniques consider a single possible completion and make additional restrictive distributional assumptions.

7.2 APPROXIMATE INFERENCE VIA CBBIS

We empirically evaluate our CBBIS scheme against different baselines on some synthetic benchmarks where we can exactly measure approximation quality. For each baseline, we measure the quality of the estimated marginals for each variable against a ground truth target distribution represented as an Ising model on a 4×4 grid whose potentials have been randomly generated. To show our methods are suitable for different graph structures, we also test on the Bayesian network ASIA [Lauritzen and Spiegelhalter, 1988]. We report in log scale the average Hellinger distance between estimated marginals and ground-truth marginals across all variables over five runs.

We compare our proposed CBBIS in Algorithm 2 against the following baselines: a vanilla Gibbs sampler (VS), a collapsed Gibbs sampling scheme (CVS), Categorical black-box importance sampling (BBIS), and online collapsed importance sampling (CIS) proposed by Friedman and Van den Broeck [2018],[4] cf. Section 6. For both BBIS and CBBIS we use Gibbs chains as proposal mechanisms. Note that CIS employs a different and adaptive proposal scheme where new samples and variables to be collapsed are heuristically selected by computing marginals via the SDD that compiles the collapsed distribution.

For the kernel function in KDSD, we follow the kernel choice in Yang et al. [2018], that is, the exponential Hamming kernel. The quadratic programming problem to retrieve the optimal weights in BBIS and CBBIS is solved by CVXOPT [Vandenberghe, 2010]. To obtain the PC representation of collapsed samples, we use the compilation algorithm by Shen et al. [2016] for collapsed samples in both CBBIS and CVS. The compilation step is fast. For each collapsed sample, the compilation algorithm translates the conditional Ising model and the conditional Bayesian networks in our case to structured decomposable PCs within seconds. For CIS, we adopt the default compilation algorithm in its implementation. We collapse 25% and 50% of the variables for methods exploiting collapsed samples: CVS, CIS and CBBIS. Figure 3 summarizes our results: we can answer **Q2** in a positive way since CBBIS performs equally well or better than other baselines. Moreover, for **Q3**, we can see that methods with collapsed samples, CBBIS and CVS, outperform their non-collapsed counterparts, BBIS and VS respectively, i.e., collapsing helps boosting estimation. It is more evident when collapsing half of the variables.

[4] github.com/UCLA-StarAI/Collapsed-Compilation

8 CONCLUSION

We introduced kernel circuits, which enable us to derive the sufficient structural constraints for a tractable computation of expected kernels. We further demonstrate how this tractable computation gives rise to two novel kernel-embedding based algorithms.

Acknowledgements

This work is supported in part by NSF grants #CCF-1837129, #IIS-1956441, #IIS-1943641, DARPA grant #N66001-17-2-4032, a Sloan Fellowship, and gifts from Intel and Facebook Research. ZZ is supported by a NEC Student Research Fellowship.

References

Hyrum S Anderson and Maya R Gupta. Expected kernel for missing features in support vector machines. In

2011 IEEE Statistical Signal Processing Workshop (SSP), pages 285–288. IEEE, 2011.

Ibrahim Berkan Aydilek and Ahmet Arslan. A hybrid method for imputation of missing values using optimized fuzzy c-means with support vector regression and a genetic algorithm. *Information Sciences*, 233:25–35, 2013.

Jessa Bekker, Jesse Davis, Arthur Choi, Adnan Darwiche, and Guy Van den Broeck. Tractable learning for complex probability queries. In *NeurIPS*, pages 2242–2250, 2015.

G. Casella and C. P Robert. Rao-blackwellisation of sampling schemes. *Biometrika*, 83(1):81–94, 1996.

M. Chavira and A. Darwiche. Compiling bayesian networks with local structure. In *IJCAI*, volume 5, pages 1306–1312, 2005.

Z. Chen, K. Zhang, L. Chan, and B. Schölkopf. Causal discovery via reproducing kernel hilbert space embeddings. volume 26, pages 1484–1517. MIT Press, 2014.

Arthur Choi, Guy Van Den Broeck, and Adnan Darwiche. Tractable learning for structured probability spaces: A case study in learning preference distributions. In *IJCAI*, page 2861–2868, 2015.

YooJung Choi, Antonio Vergari, and Guy Van den Broeck. Probabilistic circuits: A unifying framework for tractable probabilistic modeling. 2020.

CKCN Chow and Cong Liu. Approximating discrete probability distributions with dependence trees. *IEEE transactions on Information Theory*, 14(3):462–467, 1968.

Jon Cockayne, Chris J Oates, Timothy John Sullivan, and Mark Girolami. Bayesian probabilistic numerical methods. *SIAM Review*, 61(4):756–789, 2019.

Corinna Cortes and Vladimir Vapnik. Support vector machine. volume 20, pages 273–297, 1995.

M. Dang, A. Vergari, and G. Van den Broeck. Strudel: Learning structured-decomposable probabilistic circuits. In *PGM*, sep 2020.

Adnan Darwiche. A differential approach to inference in bayesian networks. *Journal of the ACM (JACM)*, 50(3): 280–305, 2003.

Adnan Darwiche. Sdd: A new canonical representation of propositional knowledge bases. In *IJCAI*, 2011.

Adnan Darwiche and Pierre Marquis. A knowledge compilation map. *Journal of Artificial Intelligence Research*, 17:229–264, 2002.

Rina Dechter and Robert Mateescu. And/or search spaces for graphical models. *Artificial intelligence*, 171(2-3): 73–106, 2007.

B. Delyon, F. Portier, et al. Integral approximation by kernel smoothing. *Bernoulli*, 22(4):2177–2208, 2016.

D. Duvenaud, J. Lloyd, R. Grosse, J. Tenenbaum, and G. Zoubin. Structure discovery in nonparametric regression through compositional kernel search. In *ICML*, pages 1166–1174, 2013.

Tal Friedman and Guy Van den Broeck. Approximate knowledge compilation by online collapsed importance sampling. In *NeurIPS*, pages 8024–8034, 2018.

Abram Friesen and Pedro Domingos. The sum-product theorem: A foundation for learning tractable models. In *ICML*, pages 1909–1918. PMLR, 2016.

Robert Gens and Pedro Domingos. Compositional kernel machines. 2017.

A. Gretton, K. M Borgwardt, M. J Rasch, B. Schölkopf, and A. Smola. A kernel two-sample test. volume 13, pages 723–773. JMLR. org, 2012.

Manfred Jaeger. Probabilistic decision graphs—combining verification and ai techniques for probabilistic inference. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 12(supp01):19–42, 2004.

Wittawat Jitkrittum, Wenkai Xu, Zoltán Szabó, Kenji Fukumizu, and Arthur Gretton. A linear-time kernel goodness-of-fit test. In *NIPS*, pages 262–271, 2017.

Julie Josse, Nicolas Prost, Erwan Scornet, and Gaël Varoquaux. On the consistency of supervised learning with missing values. *arXiv preprint arXiv:1902.06931*, 2019.

P. Khosravi, YooJung Choi, Y. Liang, A. Vergari, and G. Van den Broeck. On tractable computation of expected predictions. In *NeurIPS*, pages 11169–11180, 2019a.

P. Khosravi, Y. Liang, Y. Choi, and G. Van den Broeck. What to expect of classifiers? reasoning about logistic regression with missing features. In *IJCAI*, 2019b.

Doga Kisa, Guy Van den Broeck, Arthur Choi, and Adnan Darwiche. Probabilistic sentential decision diagrams. In *KR*, pages 1–10, 2014.

Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

S. L Lauritzen and D. J Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society: Series B (Methodological)*, 50(2):157–194, 1988.

Y. Liang and G. Van den Broeck. Towards compact interpretable models: Shrinking of learned probabilistic sentential decision diagrams. In *IJCAI 2017 Workshop on Explainable Artificial Intelligence (XAI)*, August 2017.

Qiang Liu and Jason Lee. Black-box importance sampling. In *AISTATS*, pages 952–961. PMLR, 2017.

Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. In *NeurIPS*, pages 2378–2386, 2016.

Qiang Liu, Jason Lee, and Michael Jordan. A kernelized stein discrepancy for goodness-of-fit tests. In *ICML*, pages 276–284, 2016.

Daniel Lowd and Pedro Domingos. Approximate inference by compilation to arithmetic circuits. In *NeurIPS*, pages 1477–1485, 2010.

Daniel Lowd and Pedro Domingos. Learning arithmetic circuits. *arXiv preprint arXiv:1206.3271*, 2012.

Benjamin Marlin. *Missing data problems in machine learning*. PhD thesis, 2008.

M. Meila and M. I Jordan. Learning with mixtures of trees. *JMLR*, 1(Oct):1–48, 2000.

Krikamol Muandet and Bernhard Schölkopf. One-class support measure machines for group anomaly detection. In *UAI*, pages 449–458, 2013.

Alfred Müller. Integral probability metrics and their generating classes of functions. pages 429–443. JSTOR, 1997.

William S Noble. What is a support vector machine? *Nature biotechnology*, 24(12):1565–1567, 2006.

Chris J Oates, Mark Girolami, and Nicolas Chopin. Control functionals for monte carlo integration. *arXiv preprint arXiv:1410.2392*, 2014.

R. Peharz, S. Lang, A. Vergari, K. Stelzner, A. Molina, M. Trapp, G. Van den Broeck, K. Kersting, and Z. Ghahramani. Einsum networks: Fast and scalable learning of tractable probabilistic circuits. In *ICML*, 2020.

K. Pipatsrisawat and A. Darwiche. New compilation languages based on structured decomposability. In *AAAI*, volume 8, pages 517–522, 2008.

Hoifung Poon and Pedro Domingos. Sum-product networks: A new deep architecture. In *ICCV Workshops*, pages 689–690. IEEE, 2011.

L. Rabiner and B. Juang. An introduction to hidden markov models. *ieee assp magazine*, 3(1):4–16, 1986.

T. Rahman, P. Kothalkar, and V. Gogate. Cutset networks: A simple, tractable, and scalable approach for improving the accuracy of chow-liu trees. In *ECML-PKDD*, pages 630–645. Springer, 2014.

Tahrima Rahman, Shasha Jin, and Vibhav Gogate. Cutset bayesian networks: A new representation for learning rao-blackwellised graphical models. In *IJCAI*, pages 5751–5757, 2019.

Amirmohammad Rooshenas and Daniel Lowd. Learning sum-product networks with direct and indirect variable interactions. In *ICML*, pages 710–718. PMLR, 2014.

Maytal Saar-Tsechansky and Foster Provost. Handling missing values when applying classification models. 2007.

B. Schölkopf, A. Smola, and K. Müller. Nonlinear component analysis as a kernel eigenvalue problem. volume 10, pages 1299–1319. MIT Press, 1998.

Yujia Shen, Arthur Choi, and Adnan Darwiche. Tractable operations for arithmetic circuits of probabilistic models. In *NeurIPS*, pages 3936–3944, 2016.

A. Shih and S. Ermon. Probabilistic circuits for variational inference in discrete graphical models. *NeurIPS*, 33, 2020.

Raghav Singhal, Xintian Han, Saad Lahlou, and Rajesh Ranganath. Kernelized complete conditional stein discrepancy. *arXiv preprint arXiv:1904.04478*, 2019.

Lieven Vandenberghe. The cvxopt linear and quadratic cone program solvers. 2010.

Antonio Vergari, Nicola Di Mauro, and Floriana Esposito. Simplifying, regularizing and strengthening sum-product network structure learning. In *ECML-PKDD*, pages 343–358. Springer, 2015.

Antonio Vergari, Nicola Di Mauro, and Guy Van den Broeck. Tractable probabilistic models: Representations, inference, learning and applications. UAI Tutorial, 2019.

Antonio Vergari, YooJung Choi, Anji Liu, Stefano Teso, and Guy Van den Broeck. A compositional atlas of tractable circuit operations: From simple transformations to complex information-theoretic queries, 2021.

Jonas Vlasselaer, Guy Van den Broeck, Angelika Kimmig, Wannes Meert, and Luc De Raedt. Anytime inference in probabilistic logic programs with tp-compilation. In *IJCAI*, volume 2015, pages 1852–1858, 2015.

Dilin Wang, Zhe Zeng, and Qiang Liu. Stein variational message passing for continuous graphical models. In *ICML*, pages 5219–5227. PMLR, 2018.

Dilin Wang, Ziyang Tang, Chandrajit Bajaj, and Qiang Liu. Stein variational gradient descent with matrix-valued kernels. *NeurIPS*, 32:7834, 2019.

Jiasen Yang, Qiang Liu, Vinayak Rao, and Jennifer Neville. Goodness-of-fit testing for discrete distributions via stein discrepancy. In *ICML*, pages 5561–5570, 2018.

Tractable Computation of Expected Kernels (Supplementary material)

Wenzhe Li[*1] **Zhe Zeng**[*2] **Antonio Vergari**[2] **Guy Van den Broeck**[2]

[1]Tsinghua University
[2]University of California, Los Angeles

scott.wenzhe.li@gmail.com, {zhezeng, aver, guyvdb}@cs.ucla.edu

1 PROOFS

We first present another hardness result about the computation of expected kernels besides Theorem 2.2.

Theorem 1.1. *There exist representations of distributions p and q that are smooth and compatible, yet computing the expected kernel of a simple kernel k that is the Kronecker delta is already #P-hard.*

Proof. (an alternative proof to the one in Section 4) Consider the case when the positive definite kernel k is a Kronecker delta function defined as $k(\mathbf{x}, \mathbf{x}') = 1$ if and only if $\mathbf{x} = \mathbf{x}'$. Moreover, assume that the probabilistic circuit p is smooth and decomposable, and that $q = p$. Then computing the expected kernel is equivalent to computing the power of a probabilistic circuit p, that is, $M_k(p, q) = \sum_{\mathbf{x}\in\mathcal{X}} p^2(\mathbf{x})$ with $\mathcal{X}$ being the domain of variables $\mathbf{X}$. Vergari et al. [2021] proves that the task of computing $\sum_{\mathbf{x}\in\mathcal{X}} p^2(\mathbf{x})$ is #P-hard even when the PC p is smooth and decomposable, which concludes our proof. □

Proposition 4.4 Let p_n and q_m be two compatible probabilistic circuits over variables $\mathbf{X}$ whose output units n and m are sum units, denoted by $p_n(\mathbf{X}) = \sum_{i\in\mathsf{in}(n)} \theta_i p_i(\mathbf{X})$ and $q_m(\mathbf{X}) = \sum_{j\in\mathsf{in}(m)} \delta_j q_j(\mathbf{X})$ respectively. Let k_l be a kernel circuit with its output unit being a sum unit l, denoted by $k_l(\mathbf{X}) = \sum_{c\in\mathsf{in}(l)} \gamma_c k_c(\mathbf{X})$. Then it holds that

$$M_{k_l}(p_n, q_m) = \sum_{i\in\mathsf{in}(n)} \theta_i \sum_{j\in\mathsf{in}(m)} \delta_j \sum_{c\in\mathsf{in}(l)} \gamma_c \, M_{k_c}(p_i, q_j). \tag{1}$$

Proof. $M_{k_l}(p_n, q_m)$ can be expanded as

$$\begin{aligned}
&M_{k_l}(p_n, q_m)\\
&= \sum_{\mathbf{x}} \sum_{\mathbf{x}'} p_n(\mathbf{x}) q_m(\mathbf{x}') k_l(\mathbf{x}, \mathbf{x}')\\
&= \sum_{\mathbf{x}} \sum_{\mathbf{x}'} \sum_{i\in\mathsf{in}(n)} \theta_i p_i(\mathbf{x}) \sum_{j\in\mathsf{in}(m)} \delta_j q_j(\mathbf{x}') \sum_{c\in\mathsf{in}(l)} \gamma_c k_c(\mathbf{x}, \mathbf{x}')\\
&= \sum_{i\in\mathsf{in}(n)} \theta_i \sum_{j\in\mathsf{in}(m)} \delta_j \sum_{c\in\mathsf{in}(l)} \gamma_c \, M_{k_c}(p_i, q_j).
\end{aligned}$$

□

Proposition 4.5 Let p_n and q_m be two compatible probabilistic circuits over variables $\mathbf{X}$ whose output units n and m are product units, denoted by $p_n(\mathbf{X}) = p_{n_\mathsf{L}}(\mathbf{X}_\mathsf{L}) p_{n_\mathsf{R}}(\mathbf{X}_\mathsf{R})$ and $q_m(\mathbf{X}) = q_{m_\mathsf{L}}(\mathbf{X}_\mathsf{L}) q_{m_\mathsf{R}}(\mathbf{X}_\mathsf{R})$. Let k be a kernel circuit that is kernel-compatible with the circuit pair p_n and q_m with its output unit being a product unit denoted by $k(\mathbf{X}, \mathbf{X}') = k_\mathsf{L}(\mathbf{X}_\mathsf{L}, \mathbf{X}'_\mathsf{L}) k_\mathsf{R}(\mathbf{X}_\mathsf{R}, \mathbf{X}'_\mathsf{R})$. Then it holds that

$$M_k(p_n, q_m) = M_{k_\mathsf{L}}(p_{n_\mathsf{L}}, q_{m_\mathsf{L}}) \cdot M_{k_\mathsf{R}}(p_{n_\mathsf{R}}, q_{m_\mathsf{R}}).$$

Proof. $M_k(p_n, q_m)$ can be expanded as

$$\begin{aligned}
&M_k(p_n, q_m)\\
&= \sum_{\mathbf{x}} \sum_{\mathbf{x}'} p_n(\mathbf{x}) q_m(\mathbf{x}') k(\mathbf{x}, \mathbf{x}')\\
&= \sum_{\mathbf{x}} \sum_{\mathbf{x}'} p_{m_\mathsf{L}}(\mathbf{x}_\mathsf{L}) p_{m_\mathsf{R}}(\mathbf{x}_\mathsf{R}) q_{n_\mathsf{L}}(\mathbf{x}_\mathsf{L}) q_{n_\mathsf{R}}(\mathbf{x}_\mathsf{R}) k_\mathsf{L}(\mathbf{x}_\mathsf{L}, \mathbf{x}'_\mathsf{L}) k_\mathsf{R}(\mathbf{x}_\mathsf{R}, \mathbf{x}'_\mathsf{R})\\
&= M_{k_\mathsf{L}}(p_{n_\mathsf{L}}, q_{m_\mathsf{L}}) \cdot M_{k_\mathsf{R}}(p_{n_\mathsf{R}}, q_{m_\mathsf{R}}).
\end{aligned}$$

□

Corollary 4.6. Following the assumptions in Theorem 4.3, the squared maximum mean discrepancy $MMD[\mathcal{H}, p, q]$ in RKHS $\mathcal{H}$ associated with kernel k as defined in Gretton et al. [2012] can be tractably computed.

[*]Authors contributed equally. This research was performed while W.L. was visiting UCLA remotely.

Accepted for the 37th Conference on Uncertainty in Artificial Intelligence (UAI 2021).

Proof. This is an immediate result following Theorem 4.3 by rewriting MMD as defined in Gretton et al. [2012] in the form of a linear combination of expected kernels, that is, $MMD^2[\mathcal{H}, p, q] = M_k(p,p) + M_k(q,q) - 2M_k(p,q)$. □

Corollary 4.7. Following the assumptions in Theorem 4.3, if the probabilistic circuit p further satisfies determinism, the kernelized discrete Stein discrepancy (KDSD) $\mathbb{D}^2(q \parallel p) = \mathbb{E}_{\mathbf{x},\mathbf{x}'\sim q}[k_p(\mathbf{x},\mathbf{x}')]$ in the RKHS associated with kernel k as defined in Yang et al. [2018] can be tractably computed.

Before showing the proof for Corollary 4.7, we first give definitions that are necessary for defining KDSD as follows to be self-contained.

Definition 1.2 (Cyclic permutation)**.** *For a finite set $\mathcal{X}$ and $D = |\mathcal{X}|$, a cyclic permutation $\neg : \mathcal{X} \to \mathcal{X}$ is a bijective function such that for some ordering $a_1, a_2, \cdots, a_D$ of the elements in $\mathcal{X}$, $\neg a_i = a_{(i+1) \mod D}$, $\forall i = 1, 2, \cdots, D$.*

Definition 1.3 (Partial difference operator)**.** *For any function $f : \mathcal{X} \to \mathbb{R}$ with $D = |\mathcal{X}|$, the partial difference operator is defined as*

$$\Delta_i^* f(\mathbf{X}) := f(\mathbf{X}) - f(\neg_i \mathbf{X}), \forall i = 1, \cdots, D, \quad (2)$$

with $\neg_i \mathbf{X} := (X_1, \cdots, \neg X_i, \cdots, X_D)$. Moreover, the difference operator is defined as $\Delta^* f(\mathbf{X}) := (\Delta_1^* f(\mathbf{X}), \cdots, \Delta_D^* f(\mathbf{X}))$. Similarly, let $\urcorner$ be the inverse permutation of $\neg$, and Δ denote the difference operator defined with respect to $\urcorner$, i.e.,

$$\Delta_i f(\mathbf{X}) := f(\mathbf{X}) - f(\urcorner_i \mathbf{X}), i = 1, \cdots, D.$$

Definition 1.4 (Difference score function)**.** *The (difference) score function is defined as $\boldsymbol{s}_p(\mathbf{X}) := \frac{\Delta^* p(\mathbf{X})}{p(\mathbf{X})}$ on domain $\mathcal{X}$ with $D = \mid \mathcal{X} \mid$, a vector-valued function with its i-th dimension being*

$$\boldsymbol{s}_{p,i}(\mathbf{X}) := \frac{\Delta_i^* p(\mathbf{X})}{p(\mathbf{X})} = 1 - \frac{p(\neg_i \mathbf{X})}{p(\mathbf{X})}, i = 1, 2, \cdots, D. \quad (3)$$

Given the above definitions, the discrete *Stein discrepancy* between two distributions p and q is defined as

$$\mathbb{D}(q \parallel p) := \sup_{\boldsymbol{f} \in \mathcal{H}} \mathbb{E}_{\mathbf{x}\sim q(\mathbf{X})}[\mathcal{T}_p \boldsymbol{f}(\mathbf{x})], \quad (4)$$

where $\boldsymbol{f} : \mathcal{X} \to \mathbb{R}^D$ is a *test* function, belonging to some function space $\mathcal{H}$ and $\mathcal{T}_p$ is the so-called *Stein difference operator*, which is defined as

$$\mathcal{T}_p \boldsymbol{f} = \boldsymbol{s}_p(\mathbf{x}) \boldsymbol{f}^\top - \Delta \boldsymbol{f}(\mathbf{x}). \quad (5)$$

If the function space $\mathcal{H}$ is an reproducing kernel Hilbert space (RKHS) on $\mathcal{X}$ equipped with a kernel function $k(\cdot,\cdot)$, then a *kernelized discrete Stein discrepancy* (KDSD) is defined and admits a closed-form representation as

$$\mathbb{S}(q \parallel p) := \mathbb{D}^2(q \parallel p) = \mathbb{E}_{\mathbf{x},\mathbf{x}'\sim q}[k_p(\mathbf{x},\mathbf{x}')]. \quad (6)$$

Here, the kernel function k_p is defined as

$$\begin{aligned} k_p(\mathbf{x},\mathbf{x}') = \boldsymbol{s}_p(\mathbf{x})^\top k(\mathbf{x},\mathbf{x}') \boldsymbol{s}_p(\mathbf{x}') - \boldsymbol{s}_p(\mathbf{x})^\top \Delta^{\mathbf{x}'} k(\mathbf{x},\mathbf{x}') \\ - \Delta^{\mathbf{x}} k(\mathbf{x},\mathbf{x}')^\top \boldsymbol{s}_p(\mathbf{x}') + tr(\Delta^{\mathbf{x},\mathbf{x}'} k(\mathbf{x},\mathbf{x}')), \end{aligned}$$

where the difference operator $\Delta^{\mathbf{x}}$ is as in Definition 1.3. The superscript $\mathbf{x}$ specifies the variables that it operates on.

Proof. [Corollary 4.7] By the definition of difference score functions, the close form of KDSD can be further rewritten as follows.

$$\begin{aligned} &\mathbb{E}_{\mathbf{x},\mathbf{x}'\sim q}[k_p(\mathbf{x},\mathbf{x}')] \\ =&\sum_{i=1}^{D} \mathbb{E}_{\mathbf{x},\mathbf{x}'\sim q}[\frac{p(\neg_i \mathbf{x}) p(\neg_i \mathbf{x}')}{p(\mathbf{x}) p(\mathbf{x}')} k(\mathbf{x},\mathbf{x}') - \frac{p(\neg_i \mathbf{x})}{p(\mathbf{x})} k(\mathbf{x}, \neg_i \mathbf{x}') \\ &- \frac{p(\neg_i \mathbf{x}')}{p(\mathbf{x}')} k(\neg_i \mathbf{x}, \mathbf{x}') + k(\neg_i \mathbf{x}, \neg_i \mathbf{x}')] \\ =&\sum_{i=1}^{D} [M_k(q\frac{\tilde{p}_i}{p}, q\frac{\tilde{p}_i}{p}) - M_k(q\frac{\tilde{p}_i}{p}, \tilde{q}_i) \\ &- M_k(\tilde{q}_i, q\frac{\tilde{p}_i}{p}) + M_k(\tilde{q}_i, \tilde{q}_i)] \end{aligned} \quad (7)$$

where D denotes the cardinality of the domain of variables $\mathbf{X}$, the probablity $\tilde{p}_i(\mathbf{X}) := p(\neg_i \mathbf{X})$ and the probablity $\tilde{q}_i(\mathbf{X}) := q(\neg_i \mathbf{X})$. Notice that the cyclic permutation $\neg_i$ operates on individual variable and the resulting PC $\tilde{p}_i$ and $\tilde{q}_i$ retains the same structure properties as PCs p and q respectively. To prove that KDSD can be tratably computed, it suffices to prove that the expected kernel terms in Equation 7 can be tractably computed.

For a deterministic and structured-decomposable PC p, since PC $\tilde{p}_i$ retains the same structure, then resulting ratio $\tilde{p}_i/p$ is again a smooth circuit compatible with p by Vergari et al. [2021]. Moreover, since PC p and q are compatible, the circuit $\tilde{p}_i/p$ is compatible with PC q. Thus, the resulting product $q\frac{\tilde{p}_i}{p}$ is a circuit that is smooth and compatible with both p and q by Theorem B.2 and thus compatible with $\tilde{q}_i$. By similar arguments, we can verify that all the circuit pair in the expected kernel terms in Equation 7 satisfy the assumptions in Theorem 4.3 and thus they are amenable to the tractable computation we propose in Algorithm 1, which finishes our proof.

□

Proposition (convergence of Categorical BBIS). Let $f(\mathbf{x})$ be a test function. Assume that $f - \mathbb{E}_p[f] \in \mathcal{H}_p$, with

$\mathcal{H}_p$ being the RKHS associated with the kernel function k_p, and $\sum_i w_i = 1$, then it holds that

$$\left|\sum_{n=1}^{N} w_n f(x_n) - \mathbb{E}_p f\right| \leq C_f \sqrt{\mathbb{S}(\{\mathbf{x}^{(n)}, w_n\} \parallel p)},$$

where $C_f := \parallel f - \mathbb{E}_p f \parallel_{\mathcal{H}_p}$. Moreover, the convergence rate is $\mathcal{O}(N^{-1/2})$.

Proof. Let $\hat{f}(\mathbf{x}) := f(\mathbf{x}) - \mathbb{E}_p f$, then it holds that

$$\begin{aligned}
\left|\sum_{n=1}^{N} w_n f(\mathbf{x}^{(n)}) - \mathbb{E}_p f\right| &= \left|\sum_{n=1}^{N} w_n \hat{f}(\mathbf{x}^{(n)})\right| \\
&= \left|\sum_{n=1}^{N} w_n \langle \hat{f}, k_p(\cdot, \mathbf{x}^{(n)})\rangle\right| \\
&= \left|\langle \hat{f}, \sum_{n=1}^{N} w_n k_p(\cdot, \mathbf{x}^{(n)})\rangle_{\mathcal{H}_p}\right| \\
&\leq \parallel \hat{f} \parallel_{\mathcal{H}_p} \cdot \parallel \sum_{n=1}^{N} w_n k_p(\cdot, \mathbf{x}^{(n)}) \parallel_{\mathcal{H}_p} \\
&= \parallel \hat{f} \parallel_{\mathcal{H}_p} \cdot \sqrt{\mathbb{S}(\{\mathbf{x}^{(n)}, w_n\} \parallel p)}.
\end{aligned}$$

We further prove the convergence rate of the estimation error by using the importance weights as reference weights. Let $v_n^* = \frac{1}{n} p(\mathbf{x}^{(n)})/q(\mathbf{x}^{(n)})$. Then $\mathbb{S}(\{\mathbf{x}^{(n)}, v_n^*\} \parallel p)$ is a degenerate V-statistics [Liu and Lee, 2017] and it holds that $\mathbb{S}(\{\mathbf{x}^{(n)}, v_n^*\} \parallel p) = \mathcal{O}(N^{-1})$. Moreover, we have that $\sum_{n=1}^{N} v_n^* = 1 + \mathcal{O}(N^{-1/2})$, which we denote by Z, i.e., $Z = \sum_{n=1}^{N} v_n^*$. Let $w_n^* = v_n^*/Z$, then it holds that

$$\mathbb{S}(\{\mathbf{x}^{(n)}, w_n^*\} \parallel p) = \frac{\mathbb{S}(\{\mathbf{x}^{(n)}, v_n^*\} \parallel p)}{Z^2} = \mathcal{O}(N^{-1}).$$

Therefore,

$$\begin{aligned}
\left|\sum_{n=1}^{N} w_n f(\mathbf{x}^{(n)}) - \mathbb{E}_p f\right| &\leq \parallel \hat{f} \parallel_{\mathcal{H}_p} \cdot \sqrt{\mathbb{S}(\{\mathbf{x}^{(n)}, w_n\} \parallel p)} \\
&\leq \parallel \hat{f} \parallel_{\mathcal{H}_p} \cdot \sqrt{\mathbb{S}(\{\mathbf{x}^{(n)}, w_n^*\} \parallel p)} \\
&= \mathcal{O}(N^{-1/2}).
\end{aligned}$$

□

Proposition 5.5. Let $p(\mathbf{X_c} \mid \mathbf{x_s})$ be a PC that encodes a conditional distribution over variables $\mathbf{X_c}$ conditioned on $\mathbf{X_s} = \mathbf{x_s}$, and k be a KC. If the PC $p(\mathbf{X_c} \mid \mathbf{x_s})$ and $p(\mathbf{X_c} \mid \mathbf{x_s}')$ are compatible and k is kernel-compatible with the PC pair for any $\mathbf{x_s}$, $\mathbf{x_s}'$, then the conditional kernel function $k_{p,\mathbf{s}}$ as defined in Proposition 5.4 can be tractably computed.

Proof. From Proposition 5.4, $k_{p,\mathbf{s}}$ can be written as

$$k_{p,\mathbf{s}} = \sum_{i=1}^{D} \mathbb{E}_{\mathbf{x_c} \sim p(\mathbf{X_c}|\mathbf{x_s}), \mathbf{x_c}' \sim p(\mathbf{X_c}|\mathbf{x_s}')}[k_{p,i}(\mathbf{x}, \mathbf{x}')],$$

where $k_{p,i}$ can be expanded as follows.

$$\begin{aligned}
k_{p,i}(\mathbf{x}, \mathbf{x}') =& \frac{p(\neg_i \mathbf{x}) p(\neg_i \mathbf{x}')}{p(\mathbf{x}) p(\mathbf{x}')} k(\mathbf{x}, \mathbf{x}') - \frac{p(\neg_i \mathbf{x})}{p(\mathbf{x})} k(\mathbf{x}, \neg_i \mathbf{x}') \\
&- \frac{p(\neg_i \mathbf{x}')}{p(\mathbf{x}')} k(\neg_i \mathbf{x}, \mathbf{x}') + k(\neg_i \mathbf{x}, \neg_i \mathbf{x}').
\end{aligned}$$

for any $i \in \mathbf{c}$, given that none of the variables in $\mathbf{X_s}$ is flipped in the above formulation, kernel $k_{p,i}$ can be further written as

$$\begin{aligned}
k_{p,i}(\mathbf{x}, \mathbf{x}') =& \frac{p(\neg_i \mathbf{x_c} \mid \mathbf{x_s}) p(\neg_i \mathbf{x_c}' \mid \mathbf{x_s}')}{p(\mathbf{x_c} \mid \mathbf{x_s}) p(\mathbf{x_c}' \mid \mathbf{x_s}')} k(\mathbf{x}, \mathbf{x}') \\
&- \frac{p(\neg_i \mathbf{x_c} \mid \mathbf{x_s})}{p(\mathbf{x_c} \mid \mathbf{x_s})} k(\mathbf{x}, \neg_i \mathbf{x}') \\
&- \frac{p(\neg_i \mathbf{x_c}' \mid \mathbf{x_s}')}{p(\mathbf{x_c}' \mid \mathbf{x_s}')} k(\neg_i \mathbf{x}, \mathbf{x}') \\
&+ k(\neg_i \mathbf{x}, \neg_i \mathbf{x}').
\end{aligned}$$

By substituting $k_{p,i}$ into the expected kernel in the expectation of $k_{p,i}$ with respect to the conditional distributions can be simplified to be a constant zero, that is,

$$\mathbb{E}_{\mathbf{x_c} \sim p(\mathbf{X_c}|\mathbf{x_s}), \mathbf{x_c}' \sim p(\mathbf{X_c}'|\mathbf{x_s}')}[k_{p,i}(\mathbf{x}, \mathbf{x}')] = 0.$$

Thus, $k_{p,\mathbf{s}}$ can be expanded as

$$\begin{aligned}
k_{p,\mathbf{s}}(\mathbf{x}, \mathbf{x}') &= \mathbb{E}_{\mathbf{x_c} \sim p(\mathbf{X_c}|\mathbf{x_s}), \mathbf{x_c}' \sim p(\mathbf{X_c}|\mathbf{x_s}')}[\sum_{i \in \mathbf{s}} k_{p,i}(\mathbf{x}, \mathbf{x}')] \\
&= \sum_{i \in \mathbf{s}} [\frac{p(\neg_i \mathbf{x_s}) p(\neg_i \mathbf{x_s}')}{p(\mathbf{x_s}) p(\mathbf{x_s}')} \cdot M_{k(\cdot,\cdot)}(p(\cdot \mid \neg_i \mathbf{x_s}), p(\cdot \mid \neg_i \mathbf{x_s}')) \\
&\quad - \frac{p(\neg_i \mathbf{x_s})}{p(\mathbf{x_s})} \cdot M_{k(\cdot,\neg_i \cdot)}(p(\cdot \mid \neg_i \mathbf{x_s}), p(\cdot \mid \mathbf{x_s}')) \\
&\quad - \frac{p(\neg_i \mathbf{x_s}')}{p(\mathbf{x_s}')} \cdot M_{k(\neg_i \cdot,\cdot)}(p(\cdot \mid \mathbf{x_s}), p(\cdot \mid \neg_i \mathbf{x_s}')) \\
&\quad + M_{k(\neg_i \cdot,\neg_i \cdot)}(p(\cdot \mid \mathbf{x_s}), p(\cdot \mid \mathbf{x_s}'))].
\end{aligned}$$

As Theorem 4.3 has shown that $M_k(p, q)$ can be computed exactly in time linear in the size of each PC, $k_{p,\mathbf{s}}(\mathbf{x}, \mathbf{x}')$ can also be computed exactly in time $\mathcal{O}(|p_1||p_2||k|)$, where p_1 and p_2 denote circuits that represent the conditional probability distribution given the index set, i.e., $p(\cdot \mid \mathbf{x_s})$ or $p(\cdot \mid \neg_i \mathbf{x_s})$. □

2 ALGORITHMS

Algorithm 1 summarizes how to perform the BBIS scheme we propose for Categorical distributions, and generate a set of weighted samples.

Algorithm 1 CATEGORICALBBIS(p, q, k, n)

Input: target distributions p over variables $\mathbf{X}$, a black-box mechanism q, a kernel function k and number of samples n
Output: weighted samples $\{(\mathbf{x}^{(i)}, w_i^*)\}_{i=1}^n$

Sample $\{\mathbf{x}^{(i)}\}_{i=1}^n$ from q
for $i = 1, \ldots, n$ **do**
    for $j = 1, \ldots, n$ **do**
        $[\boldsymbol{K}_p]_{ij} = k_p(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})$ ▷ cf. Section 5.2
$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \left\{ \boldsymbol{w}^\top \boldsymbol{K_p} \boldsymbol{w} \,\middle|\, \sum_{i=1}^n w_i = 1,\ w_i \geq 0 \right\}$
return $\{(\mathbf{x}^{(i)}, w_i^*)\}_{i=1}^n$