Tractable Computation of Expected Kernels

Wenzhe Li*
Tsinghua University

Zhe Zeng*
University of California, Los Angeles

Antonio Vergari
University of California, Los Angeles

Guy Van den Broeck
University of California, Los Angeles
**Expected Kernels**

Motivation

Given two distributions $p$ and $q$, and a kernel $k$, the task is to compute the expected kernel

$$\mathbb{E}_{x \sim p, x' \sim q}[k(x, x')]$$
**Expected Kernels**

**Motivation**

Given two distributions $p$ and $q$, and a kernel $k$, the task is to compute the expected kernel

$$E_{x \sim p, x' \sim q}[k(x, x')]$$

⇒ In kernel-based frameworks, expected kernels are omnipresent!
**Expected Kernels**

**Motivation**

Given two distributions \( p \) and \( q \), and a kernel \( k \), the task is to compute the expected kernel

\[
\mathbb{E}_{x \sim p, x' \sim q}[k(x, x')]
\]

\( \Rightarrow \) In kernel-based frameworks, expected kernels are omnipresent!

**Squared Maximum Mean Discrepancy (MMD)**

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

**Motivation**

Given two distributions \( p \) and \( q \), and a kernel \( k \), the task is to compute the expected kernel

\[
\mathbb{E}_{x \sim p, x' \sim q}[k(x, x')]
\]

⇒ *In kernel-based frameworks, expected kernels are omnipresent!*

**Kernelized Discrete Stein Discrepancy (KDSD)**

\[
\mathbb{E}_{x, x' \sim q}[k_p(x, x')]
\]
**Expected Kernels**

**Motivation**

Given two distributions \( p \) and \( q \), and a kernel \( k \), the task is to compute the expected kernel

\[
\mathbb{E}_{x \sim p, x' \sim q}[k(x, x')]
\]

\[\Rightarrow \text{In kernel-based frameworks, expected kernels are omnipresent!}\]

**Kernelized Support Vector Regressor (SVR) with missing features**

\[
\mathbb{E}_{x \sim p}[\sum_i w_i k(x^{(i)}, x) + b]
\]
**Challenge**

Reliability vs. Flexibility

\[ E_{x \sim p, x' \sim q}[k(x, x')] = \int_{x, x'} p(x)q(x')k(x, x') \, dx \, dx' \]

Hard to compute in general.
\[ \Rightarrow \text{approximate with Monte Carlo or variational inference} \]

**PRO.** Efficient computation

**CON.** no guarantees on error bounds
Challenge
Reliability vs. Flexibility

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

Hard to compute in general.
\[ \Rightarrow \text{approximate with Monte Carlo or variational inference} \]

**PRO.** Efficient computation

**CON.** no guarantees on error bounds

\( p, q, k \) fully factorized

**PRO.** Tractable exact computation

**CON.** Model being too restrictive
**Challenge**

Reliability vs. Flexibility

$$
E_{x \sim p, x' \sim q}[k(x, x')] = \int_{x, x'} p(x)q(x')k(x, x') \, dx \, dx'
$$

Hard to compute in general.

⇒ approximate with Monte Carlo or variational inference

**trade-off?**

<table>
<thead>
<tr>
<th><strong>PRO.</strong> Efficient computation</th>
<th><strong>p, q, k</strong> fully factorized</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CON. no guarantees</strong> on error bounds</td>
<td><strong>PRO.</strong> Tractable exact computation</td>
</tr>
<tr>
<td><strong>CON.</strong> Model being too restrictive</td>
<td></td>
</tr>
</tbody>
</table>

3/22
Circuits

Probabilistic Circuits
deep generative models + deep guarantees

Kernel Circuits
express kernels as circuits

⇒ $\mathbb{E}_{x \sim p, x' \sim q}[k(x, x')]$
A simple tractable distribution is a PC

\[ \text{e.g., a multivariate Gaussian} \]
I. A simple tractable distribution is a PC
II. A convex combination of PCs is a PC

⇒ e.g., a mixture model
I. A simple tractable distribution is a PC
II. A convex combination of PCs is a PC
III. A product of PCs is a PC
Probabilistic Circuits (PCs)

Tractable computational graphs
Probabilistic Circuits (PCs)

Tractable computational graphs
Chow-Liu trees
[Chow and Liu 1968]

Junction trees
[Bach and Jordan 2001]

HMMs
[Rabiner and Juang 1986]

C Nets
[Rahman et al. 2014]

SPNs
[Poon et al. 2011]

PSDDs
[Kisa et al. 2014]

PDGs
[Jaeger 2004]
Which structural constraints ensure tractability?
A PC is **decomposable** if all inputs of product units depend on disjoint sets of variables. A PC is **smooth** if all inputs of sum units depend on the same variable sets.

---

**Darwiche and Marquis, “A knowledge compilation map”, 2002**
decomposable + smooth PCs = ...

\[ MAR \quad \int p(z, y) \, dZ \]

\[ CON \quad \frac{\int p(z, y, h) \, dH}{\int \int p(z, y, h) \, dH \, dY} \]

decomposable + smooth PCs = ...

\[ \text{MAR} \quad \int p(z, y) \, dZ \]

\[ \text{CON} \quad \frac{\int p(z, y, h) \, dH}{\int \int p(z, y, h) \, dH \, dY} \]

What about the expected kernel \( \mathbb{E}_{x \sim p, x' \sim q}[k(x, x')] \)?
Can we represent \textbf{kernels as circuits} to characterize tractability of its queries?
Kernel Circuits (KCs)

**Exa.** Radial basis function (RBF) kernel $k(X, X') = \exp \left( -\sum_{i=1}^{4} |X_i - X'_i|^2 \right)$

\[
\begin{align*}
\exp(-|X_1 - X'_1|^2) \land \\
\exp(-|X_2 - X'_2|^2) \land \\
\exp(-|X_3 - X'_3|^2) \land \\
\exp(-|X_4 - X'_4|^2)
\end{align*}
\]
**Kernel Circuits (KCs)**

*Exa.* Radial basis function (RBF) kernel $k(X, X') = \exp \left( -\sum_{i=1}^{4} |X_i - X'_i|^2 \right)$

$\exp(-|X_1 - X'_1|^2) \land \exp(-|X_2 - X'_2|^2) \land \exp(-|X_3 - X'_3|^2) \land \exp(-|X_4 - X'_4|^2)$

*decomposable* if all inputs of product units depend on disjoint sets of variables
**Kernel Circuits (KCs)**

**Exa.** Radial basis function (RBF) kernel $k(X, X') = \exp \left( -\sum_{i=1}^{4} |X_i - X'_i|^2 \right)$

- Decomposable if all inputs of product units depend on disjoint sets of variables
- Smooth if all inputs of sum units depend on the same variable sets
Kernel Circuits (KCs)

Common kernels can be compactly represented as **decomposable** + **smooth** KCs:

RBF, (exponentiated) Hamming, polynomial ...
Expected Kernel
tractable computation via circuit operations

i) PCs $p$ and $q$, and KC $k$ are decomposable + smooth
Expected Kernel

tractable computation via circuit operations

i) PCs $p$ and $q$, and KC $k$ are **decomposable** + **smooth**

ii) PCs $p$ and $q$, and KC $k$ are **compatible**

$\Rightarrow$ decompose in the same way
Expected Kernel

tractable computation via circuit operations

i) PCs $p$ and $q$, and KC $k$ are **decomposable + smooth**

ii) PCs $p$ and $q$, and KC $k$ are **compatible**
**Expected Kernel**

tractable computation via circuit operations

i) PCs $p$ and $q$, and KC $k$ are **decomposable + smooth**

ii) PCs $p$ and $q$, and KC $k$ are **compatible**
**Expected Kernel**

*tractable computation via circuit operations*

i) PCs $p$ and $q$, and KC $k$ are **decomposable + smooth**

ii) PCs $p$ and $q$, and KC $k$ are **compatible**
**Expected Kernel**

tractable computation via circuit operations

i) PCs $p$ and $q$, and KC $k$ are **decomposable + smooth**

ii) PCs $p$ and $q$, and KC $k$ are **compatible**

Then computing expected kernels can be done **tractably** by a forward pass

$$\Rightarrow \mathcal{O}(|p||q||k|)$$
**smooth** + **decomposable** + **compatible** = **tractable** $E[k]$ 

[Sum Nodes] $p(X) = \sum_i w_i p_i(X)$, $q(X') = \sum_j w'_j q_j(X')$, and kernel $k(X, X') = \sum_l w''_l k_l(X, X')$: 

\[
p(X) = \sum_i w_i p_i(X) = \sum_i w_i \prod_{j=1}^d \exp\left(-\left|X_j - X'_j\right|^2\right) \prod_{j=1}^d p_{i,j}(X_j) \\
q(X') = \sum_j w'_j q_j(X') = \sum_j w'_j \prod_{j=1}^d \exp\left(-\left|X_j - X'_j\right|^2\right) \prod_{j=1}^d q_{j,i}(X'_j) \\
k(X, X') = \sum_l w''_l k_l(X, X') = \sum_l w''_l \prod_{j=1}^d \exp\left(-\left|X_j - X'_j\right|^2\right) \\
\]

\[
X_1, X_2, X_3, X_4, X_1', X_2', X_3', X_4'
\]

\[
p(X) = \sum_i w_i p_i(X) = \sum_i w_i \prod_{j=1}^d \exp\left(-\left|X_j - X'_j\right|^2\right) \prod_{j=1}^d p_{i,j}(X_j) \\
q(X') = \sum_j w'_j q_j(X') = \sum_j w'_j \prod_{j=1}^d \exp\left(-\left|X_j - X'_j\right|^2\right) \prod_{j=1}^d q_{j,i}(X'_j) \\
k(X, X') = \sum_l w''_l k_l(X, X') = \sum_l w''_l \prod_{j=1}^d \exp\left(-\left|X_j - X'_j\right|^2\right) \\
\]

\[
X_1, X_2, X_3, X_4, X_1', X_2', X_3', X_4'
\]
smooth + decomposable + compatible = tractable $E[k]$

[Sum Nodes] $p(X) = \sum_i w_i p_i(X)$, $q(X') = \sum_j w'_j q_j(X')$, and kernel $k(X, X') = \sum_l w''_l k_l(X, X')$:

\[
E[p, q][k(X, X')] = \sum_{i,j,l} w_i w'_j w''_l E[p_i, q_j][k_l(X, X')]
\]

$\Rightarrow$ expectation is “pushed down” to inputs
**smooth** + **decomposable** + **compatible** = **tractable** $E[k]$

[Product Nodes] $p_x(X) = \prod_i p_i(X_i)$, $q_x(X') = \prod_i q_i(X'_i)$, and kernel $k_x(X, X') = \prod_i k_i(X_i, X'_i)$:
**smooth** + **decomposable** + **compatible** = **tractable** $E[k]$

[Product Nodes] $p_x(x) = \prod_i p_i(x_i), \quad q_x(x') = \prod_i q_i(x'_i)$, and kernel $k_x(x, x') = \prod_i k_i(x_i, x'_i)$:

$$E_{p_x, q_x} [k_x(x, x')] = \prod_i E_{p_i, q_i} [k_i(x_i, x'_i)] \quad \Rightarrow \quad \text{expectation decomposes into easier ones}$$
smooth + decomposable + compatible = tractable $E[k]$

Algorithm 1 $E_{p_n,q_m}[k_l]$ — Computing the expected kernel

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

1: if $m, n, l$ are input nodes then 
2: return $E_{p_n,q_m}[k_l]$
3: else if $m, n, l$ are sum nodes then 
4: return $\sum_{i \in \text{in}(n), j \in \text{in}(m), c \in \text{in}(l)} w_i w'_j w''_c E_{p_i,q_j}[k_c]$
5: else if $m, n, l$ are product nodes then 
6: return $E_{p_{nL},q_{mL}}[k_L] \cdot E_{p_{nR},q_{mR}}[k_R]$

Computation can be done in one forward pass!

$\Rightarrow$ squared maximum mean discrepancy $MMD[p, q]$ [Gretton et al. 2012]

$\Rightarrow$ + determinism, kernelized discrete Stein discrepancy (KDSD) [Yang et al. 2018]

$\Rightarrow$ support vector regression (SVR) with missing features
Support vector regression with missing features

Given a regressor $f : \mathcal{X} \rightarrow \mathcal{Y}$, in the case when only features $X_o = x_o$ are observed and features $X_m$ are missing, with $X = (X_o, X_m)$, the expected prediction is

$$\mathbb{E}_{X_m \sim p(X_m|x_o)}[f(x_o, X_m)]$$
Support vector regression with missing features

For a kernel support vector regressor \( f(x) = \sum_{i=1}^{m} w_i k(x_i, x) + b \), in the case when only features \( X_o = x_o \) are observed and features \( X_m \) are missing, with \( X = (X_o, X_m) \), the expected prediction is

\[
\mathbb{E}_{x_m \sim p(x_m|x_o)}[f(x_o, x_m)] = \sum_{i=1}^{m} w_i \mathbb{E}_{x_m \sim p(x_m|x_o)}[k(x_i, (x_o, x_m))] + b
\]
Support vector regression with missing features

Expected prediction improves over the baselines
Applications

- Support vector regression with missing features
- Collapsed black-box importance sampling

⇒ What about intractable models?
Takeaways

#1: you can be both tractable and expressive

#2: circuits are a foundation for tractable inference over kernels
More on circuits ...

*Probabilistic Circuits: A Unifying Framework for Tractable Probabilistic Models*
starai.cs.ucla.edu/papers/ProbCirc20.pdf

*Probabilistic Circuits: Representations, Inference, Learning and Theory*
youtube.com/watch?v=2RAG5-L9R70

*Probabilistic Circuits*
arranger1044.github.io/probabilistic-circuits/

*Foundations of Sum-Product Networks for probabilistic modeling*
tinyurl.com/w65po5d
References


