On Approximating Total Variation Distance

Arnab Bhattacharyya
School of Computing
National University of Singapore
arnabb@nus.edu.sg

Sutanu Gayen
CSE Department
Indian Institute of Technology Kanpur
sutanu@cse.iitk.ac.in

Kuldeep S. Meel
School of Computing
National University of Singapore
meel@comp.nus.edu.sg

Dimitrios Myrisiotis
School of Computing
National University of Singapore
dimitris@nus.edu.sg

A. Pavan
Department of Computer Science
Iowa State University
pavan@iastate.edu

N. V. Vinodchandran
School of Computing
University of Nebraska-Lincoln
vinod@cse.unl.edu

August 27, 2023

Abstract

Total variation distance (TV distance) is a fundamental notion of distance between probability distributions. In this work, we introduce and study the problem of computing the TV distance of two product distributions over the domain $\{0,1\}^n$. In particular, we establish the following results.

1. The problem of exactly computing the TV distance of two product distributions is #P-complete. This is in stark contrast with other distance measures such as KL, Chi-square, and Hellinger which tensorize over the marginals leading to efficient algorithms.
2. There is a fully polynomial-time deterministic approximation scheme (FPTAS) for computing the TV distance of two product distributions P and Q where Q is the uniform distribution. This result is extended to the case where Q has a constant number of distinct marginals. In contrast, we show that when P and Q are Bayes net distributions, the relative approximation of their TV distance is NP-hard.

1 Introduction

An overarching theme in modern machine learning is the use of probability distributions to describe data. Datasets are often modeled by high-dimensional distributions with additional structures reflecting correlations among the features. In this context, a basic problem is *distance computation*: Given two distributions P and Q, compute $\rho(P,Q)$ for a distance measure ρ. For example, P and Q could be the outputs of two unsupervised learning algorithms, and one could ask how much they differ. As another example, a key component of generative adversarial networks [GPAM$^+$14, ACB17] is the discriminant which approximates the distance between the model and the true distributions.

Given two distributions P and Q over a finite domain $\mathcal{D}$, their *total variation (TV) distance*

or *statistical difference* $d_{\mathrm{TV}}(P,Q)$ is defined as

$$d_{\mathrm{TV}}(P,Q) = \max_{S\subseteq\mathcal{D}}(P(S)-Q(S)) = \frac{1}{2}\sum_{x\in\mathcal{D}}|P(x)-Q(x)|$$

which is also equal to $\sum_{x\in\mathcal{D}}\max(0,P(x)-Q(x))$. The total variation distance satisfies certain fundamental properties. First, it has a physical interpretation: The TV distance between two distributions is the maximum bias of any event with respect to the two distributions. Second, it satisfies many mathematically desirable properties: It is bounded, it is a metric, and it is invariant with respect to bijections. Because of these reasons, the total variation distance is one of the main distance measures employed in a wide range of areas including probability and statistics, machine learning, information theory, and pseudorandomness.

In this work, we study the total variation distance from a *computational* perspective. Given two distributions P and Q over a finite domain $\mathcal{D}$, how hard is it to compute $d_{\mathrm{TV}}(P,Q)$? If P and Q are explicitly specified by the probabilities of all of the points of the (discrete) domain $\mathcal{D}$, summing up the absolute values of the differences in probabilities at all points leads to a simple linear time algorithm. However, in many applications, the distributions of interest are of a high dimension with succinct representations. In these scenarios, since the size of the domain $\mathcal{D}$ is very large, an $O(|\mathcal{D}|)$ algorithm is highly impractical. Therefore, a fundamental computational question is:

> Can we design efficient algorithms (with running time polynomial in the size of the representation) for computing the TV distance between two high-dimensional distributions with succinct representations?

The simplest model for a high-dimensional distribution is the *product distribution*, which is a product of independent Bernoulli trials. More precisely, a product distribution P over $\mathcal{D}=\{0,1\}^n$ is succinctly described by n parameters $p_1,\ldots,p_n$ where each $p_i\in[0,1]$ is independently the probability that the i-th coordinate equals 1. Product distributions serve as a great testing ground for various intuitions regarding computational statistics, due to their ubiquity and simplicity. Despite their simplicity, surprisingly little is known about the complexity of computing the TV distance between product distributions. A very recent result shows the existence of a fully polynomial-time *randomized approximation* scheme (FPRAS) to relatively approximate the TV distance between two product distributions [FGJW23]. However, this result does not shed light on the complexity of the exact computation of TV distance as well as the existence of *deterministic* approximation schemes (FPTAS). Understanding the computational landscape of the total variation distance of product distributions is an important question. The present work makes significant progress towards this research goal.

1.1 Our Contributions

Our contributions are the following:

1. **We show that the exact computation of the total variation distance between two product distributions P and Q is #P-complete (Theorem 4).** This hardness result holds even when the distribution Q has at most 3 distinct one-dimensional marginals. Hence it is unlikely that there is an efficient algorithm for this computational problem, as an efficient algorithm for this problem would lead to efficient algorithms for many hard counting problems, including that of computing the number of satisfying assignments of a Boolean formula and all of the problems in the Polynomial-time Hierarchy [Sto76, Tod91].

This is a surprising result, given that for many other distance measures such as Hellinger, Chi-square, and KL, there are efficient algorithms for computing the distance between two product distributions. This is so, as these distances *tensorize* over their marginals (folklore; see also [BGKV21]), in the sense that they are easily expressible in terms of their one-dimensional marginals.

2. **We design a fully polynomial-time *deterministic* approximation scheme (FPTAS) that computes a relative approximation of the TV distance between two product distributions P and Q where Q is the uniform distribution (Theorem 10).** Building on the techniques developed, we design an FPTAS for the case when Q has a constant number of distinct one-dimensional marginals (Theorem 13; Theorem 12). This, combined with the earlier-mentioned hardness result, completely characterizes the complexity of TV distance computation for product distributions when one of the distributions has a constant number of one-dimensional marginals.

3. We investigate the complexity of the problem when the distributions P and Q are slightly more general than product distributions. In particular, **we show that it is NP-hard to relatively approximate the TV distance between two sparse *Bayesian networks* [Pea89] (see Theorem 9)**.

In summary, our study showcases the rich complexity landscape of the problem of total variation distance computation, even for simple distributions.

1.2 Organization

The rest of the paper is organized as follows: We present preliminaries in Section 2 and discuss related work in Section 3. We then present, in Section 4, the #P-hardness of the TV distance computation between product distributions. In Section 5 we present our polynomial-time deterministic approximation schemes for the estimation of TV distance between some special cases of product distributions. We conclude in Section 6. Finally, in Appendix A, we present the deferred proof of Lemma 3.

2 Preliminaries

We use $[n]$ to denote the *ordered* set $\{1, \ldots, n\}$. We will use log to denote $\log_2$ and $\mathbb{U}$ to denote the uniform distribution over the sample space. Throughout the paper, we shall assume that all probabilities are represented as rational numbers of the form a/b.

A Bernoulli distribution with parameter p is denoted by $\mathrm{Bern}(p)$. A *product distribution* is a product of independent Bernoulli distributions. A product distribution P over $\{0,1\}^n$ can be described by n parameters $p_1, \ldots, p_n$ where each $p_i \in [0,1]$ is the probability that the i-th coordinate equals 1 (such a P is usually denoted by $\bigotimes_{i=1}^{n} \mathrm{Bern}(p_i)$). For any $x \in \{0,1\}^n$, the probability of x with respect to the distribution P is given by $P(x) = \prod_{i \in S} p_i \prod_{i \in [n] \setminus S} (1 - p_i) \in [0,1]$, where $S \subseteq [n]$ is such that $i \in S$ if and only if the i-th coordinate of x is 1, independently.

DtvProduct is the following computational problem: Given two product distributions P and Q over the sample space $\{0,1\}^n$, compute $d_{\mathrm{TV}}(P, Q)$. When the distribution Q is the uniform distribution over $\{0,1\}^n$, we denote the above problem by DtvProductUnif.

A *Bayes net* is specified by a directed acyclic graph (DAG) and a sequence of conditional probability tables (CPTs), one for each of its nodes (and for each setting of the parents of each node). In this way, one may define a probability distribution over the nodes of a Bayes net. We will also consider the problem of computing $d_{\mathrm{TV}}(P, Q)$ where P and Q are Bayes net distributions, which we denote by DtvBayesNet.

A function f from $\{0,1\}^*$ to non-negative integers is in the class #P if there is a polynomial time non-deterministic Turing machine M so that for any x, $f(x)$ is equal to the number of accepting paths of $M(x)$. Our hardness result will make use of the known #P-complete problem #SubsetProd which is a counting version of the NP-complete problem SubsetProd (see [GJ79]; the proof is attributed to Yao). #SubsetProd is the following problem: Given integers $a_1, \ldots, a_n$, and a target number T, compute the number of sets $S \subseteq [n]$ such that $\prod_{i \in S} a_i = T$.

We also require a counting version of the Knapsack problem, #Knapsack which is defined as follows: Given weights $a_1, \ldots, a_n$ and capacity b, compute the number of sets $S \subseteq [n]$ such that $\sum_{i \in S} a_i \leq b$. It is known that #Knapsack is #P-complete.

The following notion of an approximation algorithm is important in this work.

Definition 1. A function $f : \{0,1\}^* \to \mathbb{R}$ admits a *fully polynomial-time approximation scheme (FPTAS)* if there is a *deterministic* algorithm $\mathcal{A}$ such that for every input x (of length n) and $\epsilon > 0$, $\mathcal{A}$ on inputs x and ε outputs a $(1+\varepsilon)$-relative approximation of $f(x)$, i.e., a value v that lies in the interval $[f(x)/(1+\varepsilon), (1+\varepsilon)f(x)]$. The running time of $\mathcal{A}$ is polynomial in n, and $1/\varepsilon$.

We require the following result from [GKM10].

Lemma 2 ([GKM10])**.** *There is an FPTAS for* #Knapsack.

In our work, we shall also use the following adaptation of the framework that was introduced by [GKM10]. We fix some terminology first. For a set $S \subseteq [n]$ its *Hamming weight (or cardinality)* $|S|$ is the number of 1's in its characteristic vector in $\{0,1\}^n$. Given a vector v in $\{0,1\}^n$ and a set $S = \{i_1, \ldots, i_k\} \subseteq [n]$, the *projection of v at S* is the string $v_{i_1} \cdots v_{i_k}$.

Lemma 3 (Following [GKM10]; proof in Appendix A)**.** *There is a deterministic algorithm that, given a* #Knapsack *instance $(a_1, \ldots, a_n, b)$ of total weight $W = \sum_i a_i + b$, $\delta > 0$, a k-size partition $S_1, \ldots, S_k$ of $[n]$ for some constant k, and $r_1, \ldots, r_k \in [n]$ such that $r_i \leq |S_i|$, outputs a $(1+\delta)$-relative approximation of the number of* Knapsack *solutions such that their projections at sets $S_1, \ldots, S_k$ have Hamming weights $r_1, \ldots, r_k$, respectively. The running time of this algorithm is polynomial in $n, \log W$, and $1/\delta$.*

3 Related Work

Most of the earlier works on computing the TV distance of succinctly represented high dimensional distributions are about the complexity and feasibility of *additive* approximations. Sahai and Vadhan [SV03] established in a seminal work that additively approximating the TV distance between two distributions that are samplable by Boolean circuits is hard for the complexity class SZK (Statistical Zero Knowledge). The complexity class SZK is fundamental to cryptography and is believed to be computationally hard. Subsequent works captured variations of this theme [GSV99, Mal15, DGPV20]: For example, [GSV99] showed that the problem of deciding whether a distribution samplable by a Boolean circuit is close or far from the uniform distribution is complete for the complexity class NISZK (Non-Interactive Statistical Zero Knowledge). Another line of work focuses on finding the complexity of computing the TV distance between two hidden Markov models culminating in the results that it is undecidable whether the TV distance is greater than a threshold or not, and that it is #P-hard to additively approximate it [CMR07, LP02, Kie18].

Complementing the above hardness results, [BGMV20] designed efficient algorithms to additively approximate the TV distance of distributions that are efficiently samplable and also efficiently computable (meaning that their probability mass function is efficiently computable). In particular, they designed efficient algorithms for additively approximating the TV distance

of structured high dimensional distributions such as Bayesian networks, Ising models, and multivariate Gaussians. In a similar vein, [PM21] studied a related *property testing* variant of TV distance, for distributions encoded by circuits.

Relative approximation of TV distance has received less attention compared to additive approximation. Very recently, [FGJW23] designed an FPRAS for relatively approximating the TV distance between two product distributions. The current work, in addition to showing that the exact computation of the TV distance between two product distributions is #P-complete, also presents deterministic approximation algorithms for a certain class of product distributions.

The work of [FGJW23] relies on *coupling techniques* from probability theory (which appear inherently randomized), whereas we design deterministic algorithms via a reduction to #Knapsack for which deterministic approximation schemes exist (see Section 5): [DFK+93] gave a subexponential-time approximation algorithm for #Knapsack. Later, [MS04] designed an FPRAS for it. Subsequently, [Dye03] presented an FPRAS for #Knapsack using simpler techniques. Later independent works of Stefankovic, Vempala, and Vigoda [SVV12] and Gopalan, Klivans, and Meka [GKM10] gave FPTAS for #Knapsack. Our work relies on the algorithms presented in [GKM10].

Finally, a work that highlights some interesting aspects of product distributions is [SRG17], whereby they show that computing r-th order statistics for product distributions is NP-hard.

4 The Hardness of Computing TV Distance

In this section, we establish hardness results. We first show that DtvProduct is #P-complete. Then we show that it is NP-hard to approximate the TV-distance between distributions that are slightly more general than product distributions. More specifically, we show that it is NP-hard to design an approximation algorithm for DtvBayesNet, even when the underlying Bayes nets are of in-degree two.

4.1 #P-Completeness of DtvProduct

We establish that following result.

Theorem 4. DtvProduct *is #P-complete. This holds even when one of the distributions has at most* 3 *distinct one-dimensional marginals.*

Proof overview: We show the hardness in two steps. In the first step, we introduce a problem called #PMFEquals and show that it is #P-hard by a reduction from #SubsetProd.

#PMFEquals is the following problem: Given a probability vector $(p_1, \ldots, p_n)$ where $p_i \in [0, 1]$ and a number v, compute the number of $x \in \{0,1\}^n$ such that $P(x) = v$, where P is the product distribution described by $(p_1, \ldots, p_n)$.

In the second step, we reduce #PMFEquals to the problem of computing the TV distance of two product distributions. For this, given a product distribution P, we construct product distributions $\hat{P}, \hat{Q}, P', Q'$ such that #PMFEquals is a polynomial-time computable function of $d_{\mathrm{TV}}(P', Q')$ and $d_{\mathrm{TV}}\left(\hat{P}, \hat{Q}\right)$. In particular, we establish that for the case where $v < 2^{-n}$ it is the case that $|\{x \mid P(x) = v\}|$ is equal to

$$\frac{d_{\mathrm{TV}}(P', Q') - d_{\mathrm{TV}}\left(\hat{P}, \hat{Q}\right)}{2\beta v}$$

for an appropriately chosen β (similarly for the case where $v \geq 2^{-n}$). Thus if there is an efficient algorithm for DtvProduct, then that algorithm can be used to efficiently solve the #P-complete problem #SubsetProd.

Detailed proof: We begin with the #P-hardness of #PMFEquals.

Lemma 5. #PMFEquals *is* #P*-hard.*

Proof. We will reduce #SubsetProd to #PMFEquals. The result will then follow from the fact that #SubsetProd is #P-hard. Let $a_1, \ldots, a_n$, and T be the numbers of an arbitrary #SubsetProd instance, namely I_S. We will create a #PMFEquals instance I_P that has the same number of solutions as I_S.

Let $p_i := \frac{a_i}{1+a_i}$ for every i and $v := T \prod_{i \in [n]} (1 - p_i)$, and observe that $a_i = \frac{p_i}{1-p_i}$. For any set $S \subseteq [n]$, we have the following equivalences:

$$\prod_{i \in S} a_i = T \Leftrightarrow \prod_{i \in S} \frac{p_i}{1-p_i} = \frac{v}{\prod_{i \in [n]}(1-p_i)} \Leftrightarrow \prod_{i \in S} p_i \prod_{i \notin S} (1-p_i) = v \Leftrightarrow P(x) = v,$$

where x is such that $x_i = 1$ if and only if $i \in S$. This completes the proof. □

We now turn to Theorem 4.

Proof of Theorem 4. We separately prove membership in #P and #P-hardness.

Membership in #P: Let P and Q be two product distributions, specified by parameters $p_1, \ldots, p_n$ and $q_1, \ldots, q_n$, respectively. Without loss of generality we shall assume that these parameters are fractions (as we only have some finite precision available). The goal is to design a nondeterministic machine $\mathcal{N}$ that takes $p_1, \ldots, p_n$ and $q_1, \ldots, q_n$ as inputs and is such that the number of its accepting paths (normalized by an appropriate quantity; see [dCSW20]) equals $d_{\mathrm{TV}}(P, Q)$.

Let M be the product of the denominators of all parameters $p_1, \ldots, p_n, q_1, \ldots, q_n$ and their complements $1-p_1, \ldots, 1-p_n, 1-q_1, \ldots, 1-q_n$. The non-deterministic machine $\mathcal{N}$ first guesses an element $i \in \{0,1\}^n$ in the sample space of P and Q, computes $|P(i) - Q(i)|$ by using the parameters $p_1, \ldots, p_n, q_1, \ldots, q_n$, then guesses an integer $0 \leq z \leq M$, and finally accepts if and only if $1 \leq z \leq M|P(i) - Q(i)|$. (Note that $M|P(i) - Q(i)| = |M \cdot P(i) - M \cdot Q(i)|$ is an integer.)

It follows that

$$d_{\mathrm{TV}}(P, Q) = \frac{1}{2} \sum_{i \in \{0,1\}^n} |P(i) - Q(i)| = \frac{\text{number of accepting paths of } \mathcal{N}}{2M}$$

since the number of accepting paths of $\mathcal{N}$ is

$$\sum_{i \in \{0,1\}^n} (M\,|P(i) - Q(i)|) = M \sum_{i \in \{0,1\}^n} |P(i) - Q(i)| = M \cdot 2 d_{\mathrm{TV}}(P, Q).$$

#P-hardness: For establishing hardness, we will reduce #PMFEquals to DtvProduct. The theorem will then follow from Lemma 5.

Let $p_1, \ldots, p_n$ and v be the numbers in an arbitrary instance of #PMFEquals where each p_i is represented as an m-bit binary fraction. With this, $P(x)$ can be represented as an nm-bit binary fraction. Thus without loss of generality, we can assume that v is also an nm-bit fraction. We distinguish between two cases depending on whether $v < 2^{-n}$ or $v \geq 2^{-n}$.

Case A: $v < 2^{-n}$**.** First, we construct two distributions $\hat{P} = \mathrm{Bern}(\hat{p}_1) \otimes \cdots \otimes \mathrm{Bern}(\hat{p}_{n+1})$ and $\hat{Q} = \mathrm{Bern}(\hat{q}_1) \otimes \cdots \otimes \mathrm{Bern}(\hat{q}_{n+1})$ over $\{0,1\}^{n+1}$ as follows: $\hat{p}_i := p_i$ for $i \in [n]$ and $\hat{p}_{n+1} := 1$; $\hat{q}_i := 1/2$ for $i \in [n]$ and $\hat{q}_{n+1} := v2^n$. We have that $d_{\mathrm{TV}}\left(\hat{P}, \hat{Q}\right)$ is equal to

$$\begin{aligned}\sum_{x \in \{0,1\}^{n+1}} \max\left(0, \hat{P}(x) - \hat{Q}(x)\right) &= \sum_{x} \max\left(0, P(x) - \frac{1}{2^n} v 2^n\right) \\ &= \sum_{x \in \{0,1\}^n} \max(0, P(x) - v) \\ &= \sum_{x : P(x) > v} (P(x) - v)\,. \end{aligned} \tag{1}$$

We now define two more distributions P' and Q' over $\{0,1\}^{n+2}$, by making use of the following claim.

Claim 6. *There exists a $\beta \in (0,1)$ such that the following hold for all x: If $P(x) < v$, then $P(x)\left(\frac{1}{2} + \beta\right) < v\left(\frac{1}{2} - \beta\right)$; if $P(x) > v$, then $P(x)\left(\frac{1}{2} - \beta\right) > v\left(\frac{1}{2} + \beta\right)$. In particular, we can take β to be equal to $\frac{1}{2^{3nm}}$.*

Proof. For Claim 6 to hold, observe that we want β to be at most $\frac{|v - P(x)|}{v + P(x)}$ for every x, so that $P(x) \neq v$. Since both v and $P(x)$ have nm-bit representations, both $|v - P(x))|$ and $v + P(x)$ have nm-bit representations. Thus $\frac{|v-P(x)|}{v+P(x)}$ can be represented as a $2nm$-bit fraction. Since this fraction is not zero, and the smallest $2nm$-bit fraction is $\frac{1}{2^{2nm}}$, choosing $\beta := \frac{1}{2^{3nm}}$ suffices. □

We now define two new distributions P' and Q' as follows: $p'_i := p_i$ for $i \in [n]$, $p'_{n+1} := 1$, and $p'_{n+2} := \frac{1}{2} + \beta$; $q'_i := \frac{1}{2}$ for $i \in [n]$, $q'_{n+1} := v2^n$, and $q'_{n+2} := \frac{1}{2} - \beta$ where β is as in Claim 6.

We establish the following claim.

Claim 7. *It is the case that $|\{x \mid P(x) = v\}|$ equals*

$$\frac{d_{\mathrm{TV}}(P', Q') - d_{\mathrm{TV}}\left(\hat{P}, \hat{Q}\right)}{2\beta v}.$$

Proof. We have that $d_{\mathrm{TV}}(P', Q')$ is equal to

$$\begin{aligned}\sum_{x \in \{0,1\}^{n+2}} \max\big(0, P'(x) - Q'(x)\big) &= \sum_{x \in \{0,1\}^n} \max\left(0, P(x)\left(\frac{1}{2} + \beta\right) - \frac{1}{2^n} v 2^n \left(\frac{1}{2} - \beta\right)\right) \\ &\quad + \sum_{x \in \{0,1\}^n} \max\left(0, P(x)\left(\frac{1}{2} - \beta\right) - \frac{1}{2^n} v 2^n \left(\frac{1}{2} + \beta\right)\right) \\ &= \sum_{x} \max\left(0, P(x)\left(\frac{1}{2} + \beta\right) - v\left(\frac{1}{2} - \beta\right)\right) \\ &\quad + \sum_{x} \max\left(0, P(x)\left(\frac{1}{2} - \beta\right) - v\left(\frac{1}{2} + \beta\right)\right) \\ &= \sum_{x : P(x) \geq v} P(x)\left(\frac{1}{2} + \beta\right) - v\left(\frac{1}{2} - \beta\right) \\ &\quad + \sum_{x : P(x) > v} P(x)\left(\frac{1}{2} - \beta\right) - v\left(\frac{1}{2} + \beta\right)\end{aligned}$$

$$
\begin{aligned}
&= \sum_{x:P(x)=v} P(x)\left(\frac{1}{2}+\beta\right) - v\left(\frac{1}{2}-\beta\right) \\
&\quad + \sum_{x:P(x)>v} P(x)\left(\frac{1}{2}+\beta\right) - v\left(\frac{1}{2}-\beta\right) \\
&\quad + \sum_{x:P(x)>v} P(x)\left(\frac{1}{2}-\beta\right) - v\left(\frac{1}{2}+\beta\right) \\
&= 2\beta v\left|\{x \mid P(x)=v\}\right| + \sum_{x:P(x)>v}\left(P(x)-v\right).
\end{aligned}
$$

The result now follows from Equation (1). The first equality follows from the definitions of P' and Q' (since $p'_{n+1}=1$). Note that when for every x with $P(x)<v$, by Claim 6, $P(x)(\frac{1}{2}+\beta)<v(\frac{1}{2}-\beta)$ and if $P(x)\geq v$, then $P(x)(\frac{1}{2}+\beta)\geq v(\frac{1}{2}-\beta)$. Also when $P(x)\leq v$, $P(x)(\frac{1}{2}+\beta)$ is at most $v(\frac{1}{2}-\beta)$ and when $P(x)>v$, by Claim 6, we have $P(x)(\frac{1}{2}-\beta)>v(\frac{1}{2}+\beta)$. These imply the third equality. The rest of the equalities holds by algebraic manipulations. □

For that matter $|\{x \mid P(x)=v\}|$ can be computed by computing $d_{\mathrm{TV}}(P',Q')$ and $d_{\mathrm{TV}}(\hat{P},\hat{Q})$. Thus the proof in this case follows by Lemma 5.

Case B: $v \geq 2^{-n}$**.** First, let us define distributions $\hat{P}=\mathrm{Bern}(\hat{p}_1)\otimes\cdots\otimes\mathrm{Bern}(\hat{p}_n)$ and $\hat{Q}=\mathrm{Bern}(\hat{q}_1)\otimes\cdots\otimes\mathrm{Bern}(\hat{q}_n)$ as follows: $\hat{p}_i := p_i$ for $i\in[n]$, $\hat{p}_{n+1} := \frac{1}{v2^n}$; $\hat{q}_i := \frac{1}{2}$ for $i\in[n]$, and $\hat{q}_{n+1}:=1$.

We now have that $d_{\mathrm{TV}}\left(\hat{P},\hat{Q}\right)$ is equal to $\frac{1}{2}\sum_x\left|\hat{P}(x)-\hat{Q}(x)\right|$ or

$$
\begin{aligned}
\sum_x \max\left(0,\hat{P}(x)-\hat{Q}(x)\right) &= \sum_x \max\left(0,P(x)\,\frac{1}{v2^n}-\frac{1}{2^n}\right)+\sum_x\max\left(0,P(x)\left(1-\frac{1}{v2^n}\right)\right) \\
&= \sum_x \max\left(0,P(x)\,\frac{1}{v2^n}-\frac{1}{2^n}\right)+1-\frac{1}{v2^n}.
\end{aligned}
$$

As earlier, we define two more distributions P' and Q', by making use of Claim 6. The new distributions P' and Q' are such that $p'_i := p_i$ for $i\in[n]$, $p'_{n+1} := \frac{1}{v2^n}$, and $p'_{n+2} := \frac{1}{2}+\beta$; $q'_i := 1/2$ for $i\in[n]$, $q'_{n+1}:=1$, and $q'_{n+2} := \frac{1}{2}-\beta$.

We establish the following claim.

Claim 8. *We have that* $|\{x \mid P(x)=v\}|$ *is equal to*

$$
\frac{2^{n-1}}{\beta}\left(d_{\mathrm{TV}}\left(P',Q'\right)-d_{\mathrm{TV}}\left(\hat{P},\hat{Q}\right)\right).
$$

Proof. By our previous discussion we know that $d_{\mathrm{TV}}(P',Q')$ is equal to

$$
\begin{aligned}
\sum_x \max\left(0,P'(x)-Q'(x)\right) &= \sum_x \max\left(0,P(x)\,\frac{1}{v2^n}\left(\frac{1}{2}+\beta\right)-\frac{1}{2^n}\left(\frac{1}{2}-\beta\right)\right) \\
&\quad + \sum_x \max\left(0,P(x)\,\frac{1}{v2^n}\left(\frac{1}{2}-\beta\right)-\frac{1}{2^n}\left(\frac{1}{2}+\beta\right)\right) \\
&\quad + \sum_x \max\left(0,P(x)\left(1-\frac{1}{v2^n}\right)\left(\frac{1}{2}+\beta\right)\right) \\
&\quad + \sum_x \max\left(0,P(x)\left(1-\frac{1}{v2^n}\right)\left(\frac{1}{2}-\beta\right)\right)
\end{aligned}
$$

$$
\begin{aligned}
&= \sum_x \max\left(0, P(x)\,\frac{1}{v2^n}\left(\frac{1}{2}+\beta\right) - \frac{1}{2^n}\left(\frac{1}{2}-\beta\right)\right) \\
&\quad + \sum_x \max\left(0, P(x)\,\frac{1}{v2^n}\left(\frac{1}{2}-\beta\right) - \frac{1}{2^n}\left(\frac{1}{2}+\beta\right)\right) \\
&\quad + \sum_x \max\left(0, P(x)\left(1-\frac{1}{v2^n}\right)\right) \\
&= \sum_{x:P(x)\geq v} \max\left(0, P(x)\,\frac{1}{v2^n}\left(\frac{1}{2}+\beta\right) - \frac{1}{2^n}\left(\frac{1}{2}-\beta\right)\right) \\
&\quad + \sum_{x:P(x)>v} \max\left(0, P(x)\,\frac{1}{v2^n}\left(\frac{1}{2}-\beta\right) - \frac{1}{2^n}\left(\frac{1}{2}+\beta\right)\right) \\
&\quad + 1 - \frac{1}{v2^n},
\end{aligned}
$$

by Claim 6, or

$$
\begin{aligned}
\sum_x \max\bigl(0, P'(x)-Q'(x)\bigr) &= \sum_{x:P(x)=v} \max\left(0, P(x)\,\frac{1}{v2^n}\left(\frac{1}{2}+\beta\right) - \frac{1}{2^n}\left(\frac{1}{2}-\beta\right)\right) \\
&\quad + \sum_{x:P(x)>v} \max\left(0, P(x)\,\frac{1}{v2^n}\left(\frac{1}{2}+\beta\right) - \frac{1}{2^n}\left(\frac{1}{2}-\beta\right)\right) \\
&\quad + \sum_{x:P(x)>v} \max\left(0, P(x)\,\frac{1}{v2^n}\left(\frac{1}{2}-\beta\right) - \frac{1}{2^n}\left(\frac{1}{2}+\beta\right)\right) \\
&\quad + 1 - \frac{1}{v2^n} \\
&= 2\beta\frac{1}{2^n}\left|\{x \mid P(x)=v\}\right| + \sum_x \max\left(0, P(x)\,\frac{1}{v2^n} - \frac{1}{2^n}\right) + 1 - \frac{1}{v2^n} \\
&= \frac{\beta}{2^{n-1}}\left|\{x \mid P(x)=v\}\right| + d_{\mathrm{TV}}\left(\hat{P}, \hat{Q}\right)
\end{aligned}
$$

by Claim 8. That is, $|\{x \mid P(x)=v\}|$ is equal to

$$
\frac{2^{n-1}\left(d_{\mathrm{TV}}(P',Q') - d_{\mathrm{TV}}\left(\hat{P},\hat{Q}\right)\right)}{\beta}.
$$

□

Thus also in this case the proof follows by Lemma 5. Finally, note that in either case the distribution $\hat{Q}$ has 2 distinct one-dimensional marginals and Q' has 3 distinct one-dimensional marginals. □

4.2 Hardness of Approximating DtvBayesNet

In this section, we prove the following.

Theorem 9. *Given two probability distributions P and Q that are defined by Bayes nets of in-degree at least two, it is* NP*-complete to decide whether $d_{\mathrm{TV}}(P,Q) \neq 0$ or not. Hence the problem of relatively approximating* DtvBayesNet *is* NP*-hard.*

Proof. The proof gives a reduction from the satisfiability problem for CNF formulas (which is NP-hard [Coo71]) to deciding whether the total variation distance between two Bayes nets

distributions is non-zero or not. Let F be a CNF formula viewed as a Boolean circuit. Assume F has n input variables $x_1, \ldots, x_n$ and m gates $\Gamma = \{y_1, \ldots, y_m\}$, where Γ is topologically sorted with y_m being the output gate. We will define two Bayes net distributions on the same directed acyclic graph G which, intuitively, is the graph of F. (By a graph of a formula we mean the directed acyclic graph that captures the circuit structure of F, whereby the nodes are either AND, OR, NOT, or variable gates, and the edges correspond to wires connecting the gates.)

The vertex set of G is split into two sets $\mathcal{X}$ and $\mathcal{Y}$, and a node Z. The set $\mathcal{X} = \{X_i\}_{i=1}^n$ contains n nodes with node X_i corresponding to variable x_i and the set $\mathcal{Y} = \{Y_i\}_{i=1}^m$ contains m nodes with each node Y_i corresponding to gate y_i. So totally there are $n+m+1$ nodes. There is directed edge from node V_i to node V_j if the gate/variable corresponding to V_i is an input to V_j.

The distributions P and Q on G are given by CPTs defined as follows. Each X_i is a uniformly random bit. For each Y_i, its conditional probability table (CPT) is deterministic: For each of the setting of the parents Y_j, Y_k the variable Y_i takes the value of the gate y_i for that setting of its inputs y_j, y_k. Finally, in the distribution P the variable Z is a random bit and in the distribution Q the variable Z is defined by the value of Y_m OR-ed with a random bit.

Note that the formula F computes a Boolean function on the input variables. Let $f : \{0,1\}^n \to \{0,1\}$ be this function. We extend f to $\{0,1\}^m$ (i.e., $f : \{0,1\}^n \to \{0,1\}^m$) to also include the values of the intermediate gates.

With this notation for any binary string XYZ of length $n+m+1$, both P and Q have a probability 0 if $Y \neq f(X)$. (In the derivation of TV distance that follows, we shall assume that $Y = f(X)$.) Let $A := \{x \mid F(x) = 1\}$ and $R := \{x \mid F(x) = 0\}$. Thus $2d_{\mathrm{TV}}(P,Q)$ can be written as

$$\sum_{X,f(X),Z} |P - Q| = \sum_{X \in A,Z} |P - Q| + \sum_{X \in R,Z} |P - Q|$$

where we have abused the notation P and Q to denote the probabilities $P(X, f(X), Z)$ and $Q(X, f(X), Z)$, respectively.

We will now compute each sum separately. First, we have that $\sum_{X \in A,Z} |P - Q|$ is equal to $\sum_{X \in A,Z=0} |P - Q| + \sum_{X \in A,Z=1} |P - Q|$ (taking cases for the value of Z) or

$$\sum_{X \in A,Z=0} \left| \frac{1}{2^{n+1}} - 0 \right| + \sum_{X \in A,Z=1} \left| \frac{1}{2^{n+1}} - \frac{1}{2^n} \right|$$

which is equal to $\frac{|A|}{2^n}$; then, we have that the quantity $\sum_{X \in R,Z} |P - Q|$ is equal to $\sum_{X \in R,Z=0} |P - Q| + \sum_{X \in R,Z=1} |P - Q|$ (taking cases for the value of Z) or

$$\sum_{X \in R,Z=0} \left| \frac{1}{2^{n+1}} - \frac{1}{2^{n+1}} \right| + \sum_{X \in R,Z=1} \left| \frac{1}{2^{n+1}} - \frac{1}{2^{n+1}} \right|$$

which is equal to 0.

Therefore $d_{\mathrm{TV}}(P,Q) = |A|/2^{n+1}$. The membership in NP follows because $d_{\mathrm{TV}}(P,Q) \neq 0$ if and only if there is an X so that $P(X) \neq Q(X)$; this can be checked in polynomial time for Bayes distributions over finite alphabets. The NP-hardness follows because the arbitarry CNF formula F is satisfiable if and only if $|A| \neq 0$ if and only if $d_{\mathrm{TV}}(P,Q) \neq 0$.

The NP-hardness of relative approximation of DtvBayesNet follows as a relative approximation $d_{\mathrm{TV}}(P,Q)$ is non-zero if and only if $d_{\mathrm{TV}}(P,Q) \neq 0$. □

5 Deterministic Approximation Schemes

It is an open problem to design an FPTAS for DtvProduct. In this section, we report progress on this by designing *deterministic* approximation algorithms for a few interesting subcases. In particular, we first provide an FPTAS for computing the total variation distance between an arbitrary product distribution P and the uniform distribution $\mathbb{U}$, and then extend to the case where Q has $O(1)$ distinct $q_i's$.

5.1 Algorithm for DtvProductUnif

We establish the following theorem.

Theorem 10. *There is an FPTAS for $d_{\mathrm{TV}}(P, \mathbb{U})$ where $P = \mathrm{Bern}(p_1) \otimes \cdots \otimes \mathrm{Bern}(p_n)$.*

Proof overview: The idea is to reduce an instance of DtvProductUnif to several instances of #Knapsack. Since the latter problem has an FPTAS (Lemma 2), the theorem follows.

For every subset $S \subseteq [n]$, we assign a non-negative weight Y_S and show that a "normalized" $d_{\mathrm{TV}}(P, \mathbb{U})$ is equal to $\sum_S Y_S$. We express the problem of computing this summation as multiple #Knapsack instances.

For this, we first show that each non-zero Y_S lies in the range $[1, V)$ (for an appropriate V that depends on the granularity of our precision). We divide the interval $[1, V)$ into subintervals of the form $\left[(1+\varepsilon)^{i-1}, (1+\varepsilon)^i\right)$ for various (yet polynomially many) values of i. Let k_i be the number of sets S for which Y_S lies in the i-th interval. Then $\sum_{i \in [\mathrm{poly}(n)]} k_i (1+\varepsilon)^i$ yields an $(1+\varepsilon)$ approximation of the normalized $d_{\mathrm{TV}}(P, \mathbb{U})$. However, computing each k_i exactly is also #P-hard. Thus we seek an approximation of each k_i.

We use a re-organization trick of summations and additional techniques to express this as several #Knapsack instances. Setting the approximation parameter for #Knapsack to ε, this leads to a $(1+\varepsilon^2)$-approximation algorithm. By setting $\varepsilon := \delta/2$, we get an $(1+\delta)$-approximation algorithm.

Detailed proof: We now give detailed technical proof. First we assume, without loss of generality, that no p_i is equal to $1/2$ since otherwise we can ignore these coordinates i. Moreover, again without loss of generality, we assume that $p_i > 1/2$ for all i, since otherwise we can flip 0 and 1 in the i-th coordinate of both P and $\mathbb{U}$.

Let M be the set of indices $i \in [n]$ such that $p_i = 1$. Let $A := \prod_{i \notin M} (1 - p_i)$ and $W := \frac{1}{2^n} \prod_{i \notin M} \frac{1}{1-p_i}$ be constants, and $W_S := \prod_{i \in S \setminus M} \frac{p_i}{1-p_i}$, $W_\emptyset := 1$.

Claim 11. *It is the case that $d_{\mathrm{TV}}(P, \mathbb{U})$ is equal to $A \cdot \sum_{S \subseteq [n] : M \subseteq S} \max(0, W_S - W)$.*

Proof. We have that $d_{\mathrm{TV}}(P, \mathbb{U})$ equals $\sum_{x \in \{0,1\}^n} \max(0, P(x) - \mathbb{U}(x))$ or

$$\begin{aligned}
&\sum_{S \subseteq [n]} \max\left(0, \prod_{i \in S} p_i \prod_{i \notin S} (1-p_i) - \frac{1}{2^n}\right) \\
&\quad = \sum_{S \subseteq [n] : M \subseteq S} \max\left(0, \prod_{i \in S} p_i \prod_{i \notin S} (1-p_i) - \frac{1}{2^n}\right) \\
&\quad = \prod_{i \notin M} (1-p_i) \cdot \sum_{S \subseteq [n] : M \subseteq S} \max\left(0, \prod_{i \in S \setminus M} \frac{p_i}{1-p_i} - \frac{1}{2^n} \prod_{i \notin M} \frac{1}{1-p_i}\right)
\end{aligned}$$

$$= A \sum_{S \subseteq [n] : M \subseteq S} \max\left(0, W_S - W\right).$$

The second equality holds by the definition of M. The third equality holds as $\prod_{i \in S} p_i = \prod_{i \in S \setminus M} p_i \prod_{i \in S \cap M} p_i = \prod_{i \in S \setminus M} p_i$. □

For notational simplicity, we assume that each p_i is represented using $\ell := \mathrm{poly}(n)$ bits. Thus each non-zero term $\max(0, P(x) - \mathbb{U}(x))$ of $d_{\mathrm{TV}}(P, \mathbb{U})$ contributes at least $m_0 := 2^{-\ell} = 2^{-\mathrm{poly}(n)}$ to $d_{\mathrm{TV}}(P, \mathbb{U})$. Hence for any S for which $\max\left(0, W_S - W\right) > 0$, its value is at least $m_{\min} := m_0 / A \geq 2^{-\mathrm{poly}(n)}$. Moreover, $\max\left(0, W_S - W\right)$ is at most $m_{\max}$, defined as

$$W_S = \prod_{i \in S \setminus M} \frac{p_i}{1 - p_i} \leq \prod_{i \in S \setminus M} \frac{1 - 2^{-\mathrm{poly}(n)}}{2^{-\mathrm{poly}(n)}} \leq 2^{\mathrm{poly}(n)}$$

by the facts that $p_i \leq 1 - 2^{-\mathrm{poly}(n)}$ for $i \notin M$ (since we use some finite precision of $\mathrm{poly}(n)$ bits), $(1 - x)/x$ is non-decreasing in x, and $(1 - x)/x \leq 1/x$ for all x. Therefore $m_{\max} \leq 2^{\mathrm{poly}(n)}$.

Consider now $Y_S := \max\left(0, W_S - W\right) / m_{\min}$ which lies in $[1, V)$ for some $V \leq m_{\max}/m_{\min} \leq 2^{\mathrm{poly}(n)}$, and let

$$[1, V) = \bigcup_{i=0}^{u-1} \left[(1+\varepsilon)^i, (1+\varepsilon)^{i+1}\right)$$

be a set of subintervals for integers $0 \leq i \leq u - 1 = \lceil \log_{1+\varepsilon} V \rceil - 1 \leq \mathrm{poly}(n) - 1 \leq \mathrm{poly}(n)$ and some $0 < \varepsilon < 1$ that we will fix later (as a function of δ).

Let the number of sets S such that Y_S is in $\left[1, (1+\varepsilon)^i\right)$ be n_i. Let the average contribution in the range $\left[(1+\varepsilon)^{i-1}, (1+\varepsilon)^i\right)$ be B_i. We have the following equation:

$$\frac{d_{\mathrm{TV}}(P, \mathbb{U})}{A \cdot m_{\min}} = n_1 B_1 + (n_2 - n_1) B_2 + (n_3 - n_2) B_3 + \cdots + (n_u - n_{u-1}) B_u. \tag{2}$$

Since $(1+\varepsilon)^{i-1} \leq B_i < (1+\varepsilon)^i$, the following estimate d is a $(1+\varepsilon)$-approximation of the RHS:

$$d := n_1(1+\varepsilon) + (n_2 - n_1)(1+\varepsilon)^2 + (n_3 - n_2)(1+\varepsilon)^3 + \cdots + (n_u - n_{u-1})(1+\varepsilon)^u. \tag{3}$$

We use a reorganization trick similar to [dCM19]; see Figure 1.

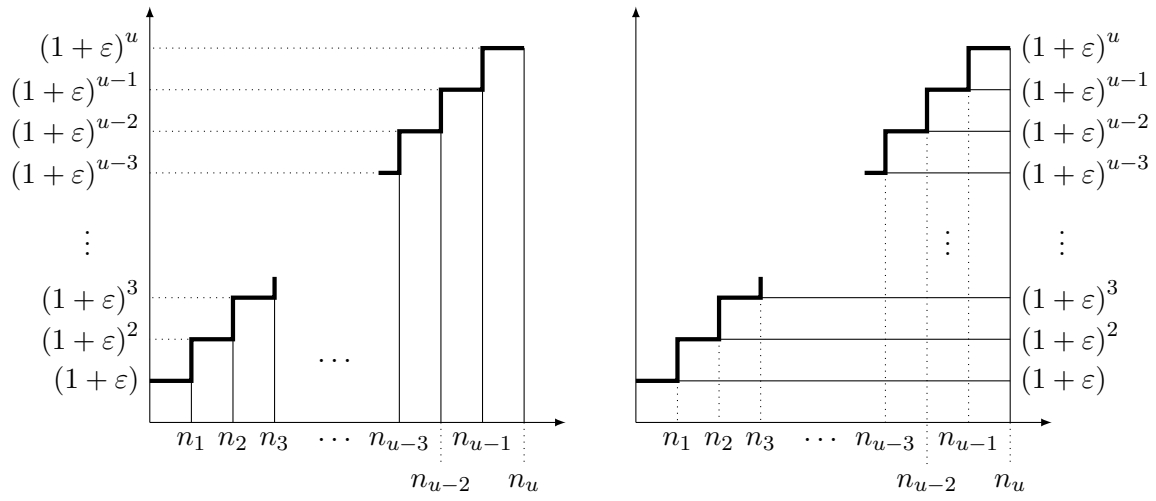

Figure 1: Reorganization trick: The area below the thick curve is calculated in two different ways.

By using the reorganization trick we have, by Equation (3),

$$d = \left((1+\varepsilon)^u - (1+\varepsilon)^{u-1}\right)(n_u - n_{u-1}) + \left((1+\varepsilon)^{u-1} - (1+\varepsilon)^{u-2}\right)(n_u - n_{u-2}) + \cdots + (1+\varepsilon)n_u. \quad (4)$$

Therefore it suffices to estimate $n_u - n_j$ for every $1 \le j \le u-1$. We know that $n_u = 2^{n-|M|}$. By definition, $t_j := n_u - n_j$ counts the sets S such that $Y_S \ge (1+\varepsilon)^j$. Let $Y := \prod_{[n]\setminus M} Y_{\{i\}}$ and observe that $Y_S \ge (1+\varepsilon)^j$ if and only if $Y_{([n]\setminus M)\setminus S} \le Y/(1+\varepsilon)^j$. Due to this bijection, t_j also counts the number of sets S such that $Y_S \le Y/(1+\varepsilon)^j$.

For every j, if $Y/(1+\varepsilon)^j < 1$ we define $t_j := 0$. Otherwise, we introduce logarithms to reduce the problem of estimating the number of sets $S \subseteq [n]$ such that $\max(0, W_S - W)/m_{\min} = Y_S \le Y/(1+\varepsilon)^j$ to a #Knapsack instance

$$\log W_S \le \log\left(m_{\min} \cdot Y/(1+\varepsilon)^j + W\right)$$

which can be more commonly written as $\sum_{i \in S\setminus M} \log w_i \le B$ for $w_i := p_i/(1-p_i)$ (by the definition of W_S) and $B := \log\left(m_{\min} \cdot Y/(1+\varepsilon)^j + W\right)$. (Note that the latter problem can be reformulated as counting the number of sets $S \subseteq [n] \setminus M$ such that $\sum_{i \in S} w_i \le B$.)

Using Lemma 2 and Equation (4) we can estimate t_j up to a $(1+\varepsilon)$-approximation in deterministic polynomial time, which in turn would give us a $(1+\varepsilon)$-approximation for d and for that matter a $(1+\varepsilon)^2$-approximation for $d_{\mathrm{TV}}(P, \mathbb{U})$ by Equation (2). Finally, we set $\varepsilon := \Omega(\delta/2)$ so that $(1+\varepsilon)^2 \le (1+\delta)$ in order to get an approximation ratio of $(1+\delta)$.

The running time is polynomial in n and $1/\delta$ because we ran a polynomial-time approximation algorithm for #Knapsack polynomially many times.

5.2 Algorithm for DtvProduct Where Q Has $O(1)$ Parameters

We will now extend to the case where Q has at most k distinct parameters. Observe that $\mathbb{U}$ can be viewed as having $k = 1$ distinct parameters (equal to $1/2$). Without loss of generality, let $Q = \bigotimes_i \mathrm{Bern}(q_i) = \mathrm{Bern}(a_1)^{z_1} \otimes \cdots \otimes \mathrm{Bern}(a_k)^{z_k}$ such that $z_1 + \cdots + z_k = n$. The main result of this section is the following.

Theorem 12. *There is an FPTAS for $d_{\mathrm{TV}}(P, Q)$ where P is an arbitrary product distribution and $Q = \bigotimes_i \mathrm{Bern}(q_i) = \mathrm{Bern}(a_1)^{z_1} \otimes \cdots \otimes \mathrm{Bern}(a_k)^{z_k}$ such that $z_1 + \cdots + z_k = n$.*

For simplicity of exposition, we will show first the result for the simpler case when $Q = \mathrm{Bern}(a)^n$.

Theorem 13. *There is an FPTAS for estimating $d_{\mathrm{TV}}(P, Q)$ where P is an arbitrary product distribution and $Q = \mathrm{Bern}(a)^n$ for an $0 \le a \le 1$.*

Our approach is to reduce this problem to #Knapsack with fixed Hamming weights. If $p_i \ge 1/2$ for all i, then the latter problem admits an FPTAS due to Lemma 3. In the scenario where there is an i such that $p_i < 1/2$ (in this case the respective Knapsack weight w_i is negative; see our discussion below), we can switch 0 and 1 in such coordinates to obtain $\mathrm{Bern}(1-p_i)$ and $\mathrm{Bern}(1-a)$, respectively. This transformation does not change the distance. We show that such instances can be reduced to #Knapsack with two fixed Hamming weights.

Proof of Theorem 13. Let M be the set of indices $i \in [n]$ such that $p_i = 1$. We have that $d_{\mathrm{TV}}(P, Q)$ is equal to $\sum_x \max(0, P(x) - Q(x))$ or

$$\sum_{S \subseteq [n] : M \subseteq S} \max\left(0, \prod_{i \in S} p_i \prod_{i \notin S} (1-p_i) - a^{|S|}(1-a)^{n-|S|}\right)$$

$$= \prod_{i \notin M} (1-p_i) \sum_{S \subseteq [n]: M \subseteq S} \max\left(0, \prod_{i \in S \setminus M} \left(\frac{p_i}{1-p_i}\right) - \frac{1}{\prod_{i \notin M}(1-p_i)} (1-a)^n \left(\frac{a}{1-a}\right)^{|S|}\right)$$

$$= A \sum_{S \subseteq [n]: M \subseteq S} \max\left(0, \prod_{i \in S \setminus M} w_i - B\left(\frac{a}{1-a}\right)^{|S|}\right)$$

for $A := \prod_{i \notin M}(1-p_i)$, $w_i := p_i/(1-p_i)$, and $B := (1-a)^n / \prod_{i \notin M}(1-p_i)$.

An argument similar to that of Theorem 10 (based again on the fact that we use finite precision) can be used to show that a normalized version of $d_{\mathrm{TV}}(P,Q)$ lies in some interval $[1,V)$ for $V \leq 2^{\mathrm{poly}(n)}$ which again we perceive as $[1,V) = \bigcup_{i=1}^{u} [(1+\varepsilon)^i, (1+\varepsilon)^{i+1})$ for $u \leq \mathrm{poly}(n)$. This enables us to use the same approach as in the proof of Theorem 10. Specifically, we approximate $d_{\mathrm{TV}}(P,Q)$ as $A \cdot m_{\min} \cdot d$ where d is defined as in Equation (3). We then approximate d as in the proof of Theorem 10, with a notable difference being that now we have to use Lemma 3 instead of Lemma 2 for the #Knapsack instances to which we reduce the estimation of $d_{\mathrm{TV}}(P,Q)$.

Therefore, following Theorem 10, it would suffice to estimate d. According to Equation (4), we shall approximate the quantities $t_j := n_u - n_j$ (n_i's as in the proof of Theorem 10), which here count the sets $S \subseteq [n]$ such that

$$\prod_{i \in S \setminus M} w_i \leq B\left(\frac{a}{1-a}\right)^{|S|} + C =: D \tag{5}$$

for $C = C(j) = m_{\min} \cdot Y/(1+\varepsilon)^j$ and the corresponding values of $m_{\min}$ and Y (see the proof of Theorem 10 for definitions). Notice how the cardinality $|S|$ of S comes up in the RHS of Equation (5). Since this quantity is not known beforehand, we shall consider cases $|S| = 1, \ldots, n$ in the #Knapsack instances that we will solve. This is the reason we use Lemma 3 instead of Lemma 2.

First assume that $w_i \geq 1$ for every i (meaning that $p_i \geq 1/2$ or $\log w_i \geq 0$ for all i); we take logarithms (as in Theorem 10) to reduce this to a #Knapsack instance (i.e., $\sum_{i \in S \setminus M} \log w_i \leq \log D$) for every fixed $|S| = 1, \ldots, n$. The latter problems can be then solved by the algorithm of Lemma 3 for $k = 1$ (in the notation of Lemma 3). Finally, we take the sum of all these counts over the possible values of $|S|$ as our estimate of t_j. Then our estimate for d will come from Equation (4) for $n_u = 2^{n-|M|}$.

Now, if for some i we have $w_i < 1$ (meaning that $p_i < 1/2$ or $\log w_i < 0$ for some i), then we switch 0 and 1 in those coordinates to get $\mathrm{Bern}(1-p_i)$ and $\mathrm{Bern}(1-a)$, respectively. Then, in Q, the first z coin biases are a and the last $n-z$ coin biases are $1-a$ without loss of generality. In that case, as before, $d_{\mathrm{TV}}(P,Q)$ is

$$A \sum_{S \subseteq [n]: M \subseteq S} \max\left(0, \prod_{i \in S \setminus M} w_i - B \prod_{i \in S} v_i\right),$$

where $w_i \geq 1$ for every i and $v_i := \frac{q_i}{1-q_i}$ whereby q_i is equal to a or $1-a$ depending on whether w_i was originally ≥ 1 or < 1, respectively.

In this case, for every S, its Hamming weight (if we identify a set $S \subseteq [n]$ with its characteristic vector in $\{0,1\}^n$) in its first z coordinates is s_1 and in its last $n-z$ coordinates is s_2. Therefore, it suffices to solve a #Knapsack instance whereby the quantity $\prod_{i \in S \setminus M} w_i - C$ is at most

$$B\left(\frac{a}{1-a}\right)^{s_1} \left(\frac{1-a}{a}\right)^{z-s_1} \left(\frac{1-a}{a}\right)^{s_2} \left(\frac{a}{1-a}\right)^{n-z-s_2}$$

for $C = C(j) = m_{\min} \cdot Y/(1+\varepsilon)^j$ as before (see the proof of Theorem 10). Note that Lemma 3 gives an algorithm for the above #KNAPSACK problem as well.

We then sum over the counts corresponding to all possible disjoint possibilities of s_1 and s_2 such that $s_1 + s_2 = |S|$, for all possible values of $|S|$, to get our estimate of t_j. Then, as earlier, our estimate for d will come from Equation (4) for $n_u = 2^{n-|M|}$. □

We now turn to Theorem 12.

Proof of Theorem 12. Let M be the set of indices $i \in [n]$ such that $p_i = 1$. First, assume that $p_i \geq 1/2$ for all i. We have that $d_{\mathrm{TV}}(P, Q)$ is

$$\sum_{S \subseteq [n]: M \subseteq S} \max\left(0, \prod_{i \in S} p_i \prod_{i \notin S} (1-p_i) - a_1^{z_{11}}(1-a_1)^{z_{10}} \cdots a_k^{z_{k1}}(1-a_k)^{z_{k0}}\right),$$

where $z_i = z_{i0} + z_{i1}$ and z_{i0} and z_{i1} denote the counts of 0's and 1's respectively in the characteristic vector of S that correspond to the a_i parameter.

Continuing our manipulation, we see that $d_{\mathrm{TV}}(P, Q)$ is equal to

$$\prod_{i \notin M}(1-p_i) \sum_{S \subseteq [n]: M \subseteq S} \max\left(0, \prod_{i \in S \setminus M}\left(\frac{p_i}{1-p_i}\right) - \frac{a_1^{z_{11}}(1-a_1)^{z_{10}} \cdots a_k^{z_{k1}}(1-a_k)^{z_{k0}}}{\prod_{i \notin M}(1-p_i)}\right)$$

$$= A \sum_{S \subseteq [n]: M \subseteq S} \max\left(0, \prod_{i \in S \setminus M} w_i - B \cdot \prod_{i=1}^{k} a_i^{z_{i1}} (1-a_i)^{z_{i0}}\right)$$

for $A := \prod_{i \notin M}(1-p_i)$, $w_i := p_i/(1-p_i)$, and $B := 1/\prod_{i \notin M}(1-p_i)$.

An argument similar to that of Theorem 10 (based again on the fact that we use finite precision) can be used to show that a normalized version of $d_{\mathrm{TV}}(P, Q)$ lies in some interval $[1, V)$ for $V \leq 2^{\mathrm{poly}(n)}$ which again we perceive as $[1, V) = \bigcup_{i=1}^{u}[(1+\varepsilon)^i, (1+\varepsilon)^{i+1})$ for $u \leq \mathrm{poly}(n)$. This enables us to use the same approach as in the proof of Theorem 10. Specifically, we approximate $d_{\mathrm{TV}}(P, Q)$ as $A \cdot m_{\min} \cdot d$ where d is defined as in Equation (3). We then approximate d as in the proof of Theorem 10, with a notable difference being that now we have to use Lemma 3 instead of Lemma 2 for the #KNAPSACK instances to which we reduce the estimation of $d_{\mathrm{TV}}(P, Q)$.

Therefore, following Theorem 10, it would suffice to estimate d. According to Equation (4), we shall approximate the quantities $t_j := n_u - n_j$ (n_i's as in the proof of Theorem 10), which here count the sets $S \subseteq [n]$ such that

$$\prod_{i \in S \setminus M} w_i \leq B \prod_{i=1}^{k} a_i^{z_{i1}} (1-a_i)^{z_{i0}} + C =: D \tag{6}$$

for $C = C(j) = m_{\min} \cdot Y/(1+\varepsilon)^j$, for the corresponding values of $m_{\min}$ and Y (see the proof of Theorem 10).

We perform this counting as follows. We partition the 2^n values of $S \subseteq [n]$ into subsets corresponding to every possibility of z_{i1}'s and z_{i0}'s between 0 and z_i, so that $|S| = \sum_{i=1}^{k} z_{i1}$. Hence there are at most $\prod_{i=1}^{k}(z_i + 1) \leq (n+1)^k$ many parts. For each such part, we solve a #KNAPSACK instance with the following constraints:

(*a*) Each z_{i1} and z_{i0} correspond to a fixed possibility determined by S.

(*b*) It is the case that $\prod_{i \in S \setminus M} w_i \leq D$ for some D determined by j and the z_{i1}'s and z_{i0}'s as in Equation (6).

Therefore for each part the number of corresponding KNAPSACK solutions can be approximately counted in polynomial time by the algorithm of Lemma 3. Our estimate for t_j then is the sum of all of these estimates, which will still be $(1+\varepsilon)$-approximate. Then (as in Theorem 10 and Theorem 13) our final estimate for d will come from Equation (4) for $n_u = 2^{n-|M|}$.

Now, if there is any $p_i < 1/2$, then we work with $(1-p_i)$ and $(1-q_i)$ at that particular coordinate and repeat the argument outlined above; this effectively doubles the number of parameters to $2k$ and the resulting algorithm would still run in polynomial time for the case where $k = O(1)$. □

6 Conclusion

We initiated a systematic study of the computational nature of the TV distance, a widely used notion of distance between probability distributions. Our findings are twofold: On the one hand, we establish hardness results for exactly computing (or approximating) the TV distance (Theorem 4; Theorem 9). On the other hand, we present efficient deterministic approximation algorithms (Theorem 10; Theorem 12; Theorem 13) for its estimation in some special cases of product distributions.

To conclude, the main open questions that arise from our work are:

1. Does there exist an FPTAS for approximating the TV distance between two product distributions?

2. For what other classes of probabilistic models do there exist TV distance approximation schemes?

3. What about other notions of distance or similarity between probabilistic models?

Acknowledgements

The work of AB was supported in part by National Research Foundation Singapore under its NRF Fellowship Programme (NRF-NRFFAI-2019-0002) and an Amazon Faculty Research Award. The work of SG was supported by an initiation grant from IIT Kanpur and a SERB award CRG/2022/007985. Pavan's work is partly supported by NSF award 2130536 and part of the work was done while visiting Simons Institute for the Theory of Computing. Vinod's work is partly supported by NSF award 2130608 and part of the work was done while visiting Simons Institute for the Theory of Computing. This work was supported in part by National Research Foundation Singapore under its NRF Fellowship Programme [NRF-NRFFAI1-2019-0004] and an Amazon Research Award. The work was done in part while AB and DM were visiting the Simons Institute for the Theory of Computing.

References

[ACB17] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *Proc. of ICML*, pages 214–223. PMLR, 2017.

[BGKV21] Arnab Bhattacharyya, Sutanu Gayen, Saravanan Kandasamy, and N. V. Vinodchandran. Testing product distributions: A closer look. In *Algorithmic Learning Theory, 16-19 March 2021, Virtual Conference, Worldwide*, volume 132 of *Proceedings of Machine Learning Research*, pages 367–396. PMLR, 2021.

[BGMV20] Arnab Bhattacharyya, Sutanu Gayen, Kuldeep S. Meel, and N. V. Vinodchandran. Efficient distance approximation for structured high-dimensional distributions via learning. In *Proc. of NeurIPS*, 2020.

[CMR07] Corinna Cortes, Mehryar Mohri, and Ashish Rastogi. L_p distance and equivalence of probabilistic automata. *Int. J. Found. Comput. Sci.*, 18(4):761–779, 2007.

[Coo71] Stephen A. Cook. The complexity of theorem-proving procedures. In Michael A. Harrison, Ranan B. Banerji, and Jeffrey D. Ullman, editors, *Proc. of STOC*, pages 151–158. ACM, 1971.

[dCM19] Alexis de Colnet and Kuldeep S. Meel. Dual hashing-based algorithms for discrete integration. In *Proc. of CP*, pages 161–176. Springer, 2019.

[dCSW20] Cassio P. de Campos, Georgios Stamoulis, and Dennis Weyland. A structured view on weighted counting with relations to counting, quantum computation and applications. *Inf. Comput.*, 275:104627, 2020.

[DFK+93] Martin E. Dyer, Alan M. Frieze, Ravi Kannan, Ajai Kapoor, Ljubomir Perkovic, and Umesh V. Vazirani. A mildly exponential time algorithm for approximating the number of solutions to a multidimensional Knapsack problem. *Comb. Probab. Comput.*, 2:271–284, 1993.

[DGPV20] Peter Dixon, Sutanu Gayen, Aduri Pavan, and N. V. Vinodchandran. Perfect zero knowledge: New upperbounds and relativized separations. In Rafael Pass and Krzysztof Pietrzak, editors, *Proc. of TCC*, volume 12550 of *Lecture Notes in Computer Science*, pages 684–704. Springer, 2020.

[Dye03] Martin E. Dyer. Approximate counting by dynamic programming. In *Proc. of STOC, June 9-11, 2003, San Diego, CA, USA*, pages 693–699. ACM, 2003.

[FGJW23] Weiming Feng, Heng Guo, Mark Jerrum, and Jiaheng Wang. A simple polynomial-time approximation algorithm for total variation distances between product distributions. In *Proc. of SOSA*, 2023.

[GJ79] M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of* NP*-Completeness*. W. H. Freeman, 1979.

[GKM10] Parikshit Gopalan, Adam R. Klivans, and Raghu Meka. Polynomial-time approximation schemes for Knapsack and related counting problems using branching programs. *Electron. Colloquium Comput. Complex.*, page 133, 2010.

[GPAM+14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[GSV99] Oded Goldreich, Amit Sahai, and Salil P. Vadhan. Can statistical zero knowledge be made non-interactive? or On the relationship of SZK and NISZK. In *Proc. of CRYPTO*, pages 467–484, 1999.

[Kie18] Stefan Kiefer. On computing the total variation distance of hidden Markov models. In *Proc. of ICALP*, pages 130:1–130:13, 2018.

[KRVZ06] Jesse Kamp, Anup Rao, Salil P. Vadhan, and David Zuckerman. Deterministic extractors for small-space sources. In *Proc. of STOC*, pages 691–700. ACM, 2006.

[LP02] Rune B. Lyngsø and Christian N. S. Pedersen. The consensus string problem and the complexity of comparing hidden Markov models. *J. Comput. Syst. Sci.*, 65(3):545–569, 2002.

[Mal15] Lior Malka. How to achieve perfect simulation and a complete problem for non-interactive perfect zero-knowledge. *J. Cryptol.*, 28(3):533–550, 2015.

[MS04] Ben Morris and Alistair Sinclair. Random walks on truncated cubes and sampling 0-1 Knapsack solutions. *SIAM J. Comput.*, 34(1):195–226, 2004.

[Pea89] Judea Pearl. *Probabilistic reasoning in intelligent systems - networks of plausible inference*. Morgan Kaufmann series in representation and reasoning. Morgan Kaufmann, 1989.

[PM21] Yash Pote and Kuldeep S. Meel. Testing probabilistic circuits. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 22336–22347, 2021.

[SRG17] David Smith, Sara Rouhani, and Vibhav Gogate. Order statistics for probabilistic graphical models. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, page 4625–4631. AAAI Press, 2017.

[Sto76] Larry J. Stockmeyer. The polynomial-time hierarchy. *Theor. Comput. Sci.*, 3(1):1–22, 1976.

[SV03] Amit Sahai and Salil P. Vadhan. A complete problem for statistical zero knowledge. *J. ACM*, 50(2):196–249, 2003.

[SVV12] Daniel Stefankovic, Santosh S. Vempala, and Eric Vigoda. A deterministic polynomial-time approximation scheme for counting Knapsack solutions. *SIAM J. Comput.*, 41(2):356–366, 2012.

[Tod91] Seinosuke Toda. PP is as hard as the polynomial-time hierarchy. *SIAM J. Comput.*, 20(5):865–877, 1991.

A Proof of Lemma 3

The proof of Lemma 3 follows by adapting the work of [GKM10]. We first fix some notation and terminology.

A (W, n)-*branching program* is a branching program of width W over n Boolean input variables. A *read-once branching program (ROBP)* is a branching program whereby each input variable is accessed only once. A *monotone* (W, n)-*ROBP* is a (W, n)-ROBP such that in each of its layers L the nodes of L are totally ordered under some relation $\prec$, and whenever $u \prec v$ for some nodes u and v it is the case that the set of partial accepting paths that start u are a subset of the the set of partial accepting paths that start at v.

Given a branching program M and a string z, the notation $M(z)$ denotes the output ("accept"/"reject") of M on input z.

An *implicit description* of a monotone ROBP is a description according to which one can efficiently check the relative order of two nodes under $\prec$ (within any layer), and given a node u one can efficiently compute its neighbors.

The following notion of small-space sources was introduced by [KRVZ06].

Definition 14 ([KRVZ06])**.** A *width-w small-space source* is described by a (w, n)-branching program D with an additional probability distribution p_v on the outgoing edges associated with vertices $v \in D$. Samples from the source are generated by taking a random walk on D according to the p_v's and outputting the labels of the edges traversed.

We require the following useful lemmas from [GKM10].

Lemma 15 ([GKM10])**.** *Given a ROBP M of width at most W and a small-space source D of width at most S, $\mathbf{Pr}_{x\sim D}[M(x) = 1]$ can be computed exactly in time $O(nSW)$.*

Lemma 16 ([GKM10])**.** *Given a (W, n)-ROBP M, the uniform distribution over M's accepting inputs, $\{x \mid M(x) = 1\}$ is a width W small-space source.*

We further require the following result from [GKM10].

Lemma 17 ([GKM10])**.** *Given a monotone (W, n)-ROBP M, $\delta > 0$, and a small-space source D over $\{0,1\}^n$ of width at most S, there exists an $(O(n^2 S/\delta), n)$-monotone ROBP M_0 such that for all z, $M(z) \leq M_0(z)$ and*

$$\mathop{\mathbf{Pr}}_{z\sim D}[M(z) = 1] \leq \mathop{\mathbf{Pr}}_{z\sim D}[M_0(z) = 1] \leq (1+\delta) \mathop{\mathbf{Pr}}_{z\sim D}[M(z) = 1]\,.$$

Moreover, given an implicit description of M and a description of D, M_0 can be constructed in deterministic time $O(n^3 S(S + \log W)\log(n/\delta)/\delta)$.

The main take-away of Lemma 17 is that the number of accepting paths of M_0 (under the distribution D) approximates the number of accepting paths of M (under the distribution D), and moreover M_0 has small width.

We now turn to the proof of Lemma 3.

Proof of Lemma 3. Take M in Lemma 17 to be a ROBP for Knapsack. In particular, M decides the validity of the inequality $\sum_{i\in S} a_i \leq b$, which may be also written as $\sum_{i\in[n]} a_i x_i \leq b$ if we let $x_i = 1$ if and only if $i \in S$. The ROBP M has $n + 1$ layers; layer 0 has a single start node. Every other layer i has a node for each partial sum $\sum_{j\leq i} a_i x_i$. For a node v in layer $i - 1$ and $x_i \in \{0, 1\}$, the x_i-th neighbor of v is $v + a_i x_i$. Naturally, the nodes in the last layer are either rejecting (if their label is more than b) or accepting (otherwise).

Note that M may have width W (at most) exponential in n; this makes it prohibitive in terms of running time to directly use Lemma 15 in order to count Knapsack solutions. Therefore, Lemma 17 comes handy here.

To apply Lemma 17, let us first note that M is monotone. Indeed, we can define a total node ordering $\prec$ within each layer of M as follows: Given two nodes u, v that both belong to some layer of M, we define $u \prec v$ if and only if $u > v$. This satisfies the requirements of a ROBP being monotone as in this case the partial solutions that start at u are a subset of the partial solutions that start at v, since the smaller partial sum v allows for more flexibility with respect to the items that we can add to its associated solution.

So by Lemma 17 we can construct in time $O(n^3 S(S + \log W)\log(n/\delta)/\delta)$ some ROBP M_0 which has width $W_0 = O(n^2 S/\delta)$ and is such that the probability that M_0 is accepting under the distribution D approximates the probability that M is accepting under the distribution D.

By Lemma 16, the Hamming weight constraints of Lemma 3 can be sampled by some small space source of width at most $S \leq \mathrm{poly}(n)$, since there is some ROBP of width $\prod_{i=1}^{k}(|S_i| + 1) \leq (n + 1)^k = \mathrm{poly}(n)$ that only accepts the set of strings that satisfy the Hamming weight constraints of Lemma 3.

This means that the width of M_0, namely W_0, is at most $O(\text{poly}(n)/\delta)$, and that M_0 can be constructed in time $O(\text{poly}(n)/\delta)$ (since $\log W = O(\text{poly}(n))$). By Lemma 15, we can compute the probability that M_0 is accepting under the distribution D in time $O(nSW_0) = O(\text{poly}(n)/\delta)$. Let p denote this probability. As a last step, we multiply p by

$$\prod_{i=1}^{k} \binom{|S_i|}{r_i},$$

which is the number of strings in the support of D (i.e., the set of strings that have non-zero probability to be sampled by D), to get the number of accepting paths of M_0.

Since $p = \mathbf{Pr}_{z\sim D}[M_0(z) = 1]$ $(1+\delta)$-approximates $\mathbf{Pr}_{z\sim D}[M_0(z) = 1]$, we get that the number of accepting paths of M_0 $(1+\delta)$-approximates the number of accepting paths of M.

The result now follows from the fact that M is a ROBP for Knapsack, so the number of accepting paths of M_0 $(1+\delta)$-approximates the number of Knapsack solutions.

Finally the running time of this procedure is polynomial in $n, \log W$, and $1/\delta$, which is polynomial in n and $1/\delta$ since the width of M is $\log W = \text{poly}(n)$. □