Approximate Inference via Weighted Rademacher Complexity

Jonathan Kuck
Computer Science Department
Stanford University
jkuck@cs.stanford.edu

Ashish Sabharwal
Allen Institute for Artificial Intelligence
Seattle, WA
ashishs@allenai.org

Stefano Ermon
Computer Science Department
Stanford University
ermon@cs.stanford.edu

Abstract

Rademacher complexity is often used to characterize the learnability of a hypothesis class and is known to be related to the class size. We leverage this observation and introduce a new technique for estimating the size of an arbitrary weighted set, defined as the sum of weights of all elements in the set. Our technique provides upper and lower bounds on a novel generalization of Rademacher complexity to the weighted setting in terms of the weighted set size. This generalizes Massart's Lemma, a known upper bound on the Rademacher complexity in terms of the unweighted set size. We show that the weighted Rademacher complexity can be estimated by solving a randomly perturbed optimization problem, allowing us to derive high-probability bounds on the size of any weighted set. We apply our method to the problems of calculating the partition function of an Ising model and computing propositional model counts (#SAT). Our experiments demonstrate that we can produce tighter bounds than competing methods in both the weighted and unweighted settings.

Introduction

A wide variety of problems can be reduced to computing the sum of (many) non-negative numbers. These include calculating the partition function of a graphical model, propositional model counting (#SAT), and calculating the permanent of a non-negative matrix. Equivalently, each can be viewed as computing the discrete integral of a non-negative weight function. Exact summation, however, is generally intractable due to the curse of dimensionality (Bellman 1961).

As alternatives to exact computation, variational methods (Jordan et al. 1998; Wainwright, Jordan, and others 2008) and sampling (Jerrum and Sinclair 1996; Madras 2002) are popular approaches for approximate summation. However, they generally do not guarantee the estimate's quality.

An emerging line of work estimates and formally bounds propositional model counts or, more generally, discrete integrals (Ermon et al. 2013a; Chakraborty, Meel, and Vardi 2013; Ermon et al. 2014; Zhao et al. 2016). These approaches reduce the problem of integration to solving a small number of optimization problems involving the same weight function but subject to additional random constraints introduced by a random hash function. This results in approximating the #P-hard problem of exact summation (Valiant 1979) using the solutions of NP-hard optimization problems.

Optimization can be performed efficiently for certain classes of weight functions, such as those involved in the computation of the permanent of a non-negative matrix. If instead of summing (permanent computation) we maximize the same weight function, we obtain a maximum weight matching problem, which is in fact solvable in polynomial time (Kuhn 1955). However, adding hash-based constraints makes the maximum matching optimization problem intractable, which limits the application of randomized hashing approaches (Ermon et al. 2013c). On the other hand, there do exist fully polynomial-time randomized approximation schemes (FPRAS) for non-negative permanent computation (Jerrum, Sinclair, and Vigoda 2004; Bezáková et al. 2006). This gives hope that approximation schemes may exist for other counting problems even when optimization with hash-based constraints is intractable.

We present a new method for approximating and bounding the size of a general weighted set (i.e., the sum of the weights of its elements) using geometric arguments based on the set's shape. Our approach, rather than relying on hash-based techniques, establishes a novel connection with Rademacher complexity (Shalev-Shwartz and Ben-David 2014). This generalizes geometric approaches developed for the unweighted case to the weighted setting, such as the work of Barvinok (1997) who uses similar reasoning but without connecting it with Rademacher complexity. In particular, we first generalize Rademacher complexity to weighted sets. While Rademacher complexity is defined as the maximum of the sum of Rademacher variables over a set, *weighted* Rademacher complexity also accounts for the weight of each element in the set. Just like Rademacher complexity is related to the size of the set, we show that *weighted* Rademacher complexity is related to the total weight of the set. Further, it can be estimated by solving multiple instances of a maximum weight optimization problem, subject to random Rademacher perturbations. Notably, the resulting optimization problem turns out to be computationally much simpler than that required by the aforementioned randomized hashing schemes. In particular, if the weight function is log-supermodular, the corresponding weighted Rademacher complexity can be estimated efficiently, as our perturbation does not change the original optimization problem's complexity (Orlin 2009;

Bach and others 2013).

Our approach most closely resembles a recent line of work involving the Gumbel distribution (Hazan and Jaakkola 2012; Hazan, Maji, and Jaakkola 2013; Hazan et al. 2016; Balog et al. 2017; Mussmann and Ermon 2016; Mussmann, Levy, and Ermon 2017). There, the Gumbel-max idea is used to bound the partition function by performing MAP inference on a model where the unnormalized probability of each state is perturbed by random noise variables sampled from a Gumbel distribution. While very powerful, exact application of the Gumbel method is impractical, as it requires exponentially many independent random perturbations. One instead uses local approximations of the technique.

Empirically, on spin glass models we show that our technique yields tighter upper bounds and similar lower bounds compared with the Gumbel method, given similar computational resources. On a suite of #SAT model counting instances our approach generally results in substantially tighter upper bounds given limited computation and, on certain classes of instances, significantly tighter lower bounds as well.

Background

Rademacher complexity is an important tool used in learning theory to bound the generalization error of a hypothesis class (Shalev-Shwartz and Ben-David 2014).

Definition 1. *The Rademacher complexity of a set $A \subseteq \mathbb{R}^n$ is defined as:*

$$R(A) \coloneqq \frac{1}{n}\mathbb{E}_{\mathbf{c}}\left[\sup_{\mathbf{a}\in A}\sum_{i=1}^{n} c_i a_i\right], \tag{1}$$

where $\mathbb{E}_{\mathbf{c}}$ denotes expectation over $\mathbf{c}$, and $\mathbf{c}$ is sampled uniformly from $\{-1,1\}^n$.

As the name suggests, it is a measure of the complexity of set A (which, in learning theory, is usually a hypothesis class). It measures how "expressive" A is by evaluating how well we can "fit" to a random noise vector $\mathbf{c}$ by choosing the closest vector (or hypothesis) from A. Intuitively, Rademacher complexity is related to $|A|$, the number of vectors in A, another crude notion of complexity of A. However, it also depends on how vectors in A are arranged in the ambient space $\mathbb{R}^n$. A central focus of this paper will be establishing quantitative relationships between $R(A)$ and $|A|$.

A key property of Rademacher complexity that makes it extremely useful in learning theory is that it can be estimated using a small number of random noise samples $\mathbf{c}$ under mild conditions (Shalev-Shwartz and Ben-David 2014). The result follows from McDiarmid's inequality:

Proposition 1 (McDiarmid, 1989)**.** *Let $X_1, \ldots, X_m \in \mathcal{X}$ be independent random variables. Let $f : \mathcal{X}^m \mapsto \mathbb{R}$ be a function that satisfies the bounded differences condition that $\forall i \in \{1, \ldots, m\}$ and $\forall x_1, \ldots, x_m, x_i' \in \mathcal{X}$:*

$$|f(x_1, \ldots, x_i, \ldots, x_m) - f(x_1, \ldots, x_i', \ldots, x_m)| \leq d_i.$$

Then for all $\epsilon > 0$

$$\Pr\left[\left|f\left(X_1, \ldots, X_m\right) - \mathbb{E}\left[f\left(X_1, \ldots, X_m\right)\right]\right| \geq \epsilon\right] \leq \exp\left(\frac{-2\epsilon^2}{\sum_j d_j^2}\right).$$

McDiarmid's inequality says we can bound, with high probability, how far a function f of random variables may deviate from its expected value, given that the function does not change much when the value of a single random variable is changed. Because the function in Eq. (1) satisfies this property (Shalev-Shwartz and Ben-David 2014), we can use Eq. (1) to bound $R(A)$ with high probability by computing the supremum for only a small number of noise samples $\mathbf{c}$.

Problem Setup

In this section we formally define our problem and introduce the optimization oracle central to our solution. Let $w : \{-1,1\}^n \to [0,\infty)$ be a non-negative weight function. We consider the problem of computing the sum

$$Z(w) = \sum_{\mathbf{x}\in\{-1,1\}^n} w(\mathbf{x}).$$

Many problems, including computing the partition function of an undirected graphical model, where $w(\mathbf{x})$ is the unnormalized probability of state $\mathbf{x}$ (see Koller and Friedman (2009)), propositional model counting (#SAT), and computing the permanent of a non-negative matrix can be reduced to calculating this sum. The problem is challenging because explicit calculation requires summing over 2^n states, which is computationally intractable in cases of interest.

Due to the general intractability of exactly calculating $Z(w)$, we focus on an efficient approach for estimating $Z(w)$ which additionally provides upper and lower bounds that hold with high probability. Our method depends on the following assumption:

Assumption 1. *We assume existence of an optimization oracle that can output the value*

$$\delta(\mathbf{c}, w) = \max_{\mathbf{x}\in\{-1,1\}^n}\{\langle \mathbf{c}, \mathbf{x}\rangle + \log w(\mathbf{x})\} \tag{2}$$

for any vector $\mathbf{c} \in \{-1,1\}^n$ and weight function $w : \{-1,1\}^n \to [0,\infty)$.

Note that throughout the paper we simply denote $\log_2$ as $\log$ and $\log_e$ as $\ln$, and assume $\log 0 = -\infty$. Assumption 1 is reasonable, as there are many classes of models where such an oracle exists. For instance, polynomial time algorithms exist for finding the maximum weight matching in a weighted bipartite graph (Hopcroft and Karp 1971; Jonker and Volgenant 1987). Graph cut algorithms can be applied to efficiently maximize a class of energy functions (Kolmogorov and Zabin 2004). More generally, MAP inference can be performed efficiently for any log-supermodular weight function (Orlin 2009; Chakrabarty, Jain, and Kothari 2014; Fujishige 1980). Our perturbation preserves the submodularity of $-\log w(\mathbf{x})$, as $\langle \mathbf{c}, \mathbf{x}\rangle$ can be viewed as n independent single variable perturbations, so we have an efficient optimization oracle whenever the original weight function is log-supermodular. Further, notice that this is a much weaker assumption compared with the optimization oracle required by randomized hashing methods (Chakraborty, Meel, and Vardi 2013; Ermon et al. 2014; Zhao et al. 2016).

If an approximate optimization oracle exists that can find a value within some known bound of the maximum, we can

modify our bounds to use the approximate oracle. This may improve the efficiency of our algorithm or extend its use to additional problem classes. For the class of log-supermodular distributions, approximate MAP inference is equivalent to performing approximate submodular minimization (Jegelka, Lin, and Bilmes 2011).

We note that even when an efficient optimization oracle exists, the problem of exactly calculating $Z(w)$ is generally still hard. For example, polynomial time algorithms exist for finding the maximum weight perfect matching in a weighted bipartite graph. However, computing the permanent of a bipartite graph's adjacency matrix, which equals the sum of weights for all perfect matchings or $Z(w)$, is still #P-complete(Jerrum, Sinclair, and Vigoda 2004). A fully polynomial randomized approximation scheme (FPRAS) exists (Jerrum, Sinclair, and Vigoda 2004; Bezáková et al. 2006), based on Markov chain Monte Carlo to sample over all perfect matchings. However, the polynomial time complexity of this algorithm suffers from a large degree, limiting its practical use.

Weighted Rademacher Bounds on $Z(w)$

Our approach for estimating the sum $Z(w) = \sum_{\mathbf{x}} w(\mathbf{x})$ is based on the idea that the Rademacher complexity of a set is related to the set's size. In particular, Rademacher complexity is monotonic in the sense that $R(A) \leq R(B)$ whenever $A \subseteq B$. Note that monotonicity does not hold for $|A| \leq |B|$, that is, $R(A)$ is monotonic in the contents of A but not necessarily in its size. We estimate the sum of arbitrary non-negative elements by generalizing the Rademacher complexity in definition 2.

Definition 2. *We define the weighted Rademacher complexity of a weight function $w : \{-1, 1\}^n \to [0, \infty)$ as*

$$\mathcal{R}(w) = \mathbb{E}_{\mathbf{c}}\left[\max_{\mathbf{x}\in\{-1,1\}^n} \{\langle \mathbf{c}, \mathbf{x}\rangle + \log w(\mathbf{x})\}\right], \quad (3)$$

for $\mathbf{c}$ sampled uniformly from $\{-1, 1\}^n$.

In the notation of Eq. (2), the weighted Rademacher complexity is simply $\mathcal{R}(w) = \mathbb{E}_{\mathbf{c}}[\delta(\mathbf{c}, w)]$. For a set $A \subseteq \{-1, 1\}^n$, let $I_A : \{-1, 1\}^n \to \{0, 1\}$ denote the indicator weight function for A, defined as $I_A(\mathbf{x}) = 1 \iff \mathbf{x} \in A$. Then $\mathcal{R}(I_A) = R(A)$, that is, the weighted Rademacher complexity is identical to the standard Rademacher complexity for indicator weight functions. For a general weight function, the weighted Rademacher complexity extends the standard Rademacher complexity by giving each element (hypothesis) its own weight.

Algorithmic Strategy

The main idea of this paper is to use the weighted Rademacher complexity $\mathcal{R}(w)$ to provide probabilistic estimates of $Z(w)$, the total weight of w.

This is a reasonable strategy because as we have seen before, for an indicator weight function $I_A : \{-1, 1\}^n \to \{0, 1\}$, $\mathcal{R}(I_A)$ reduces to the standard Rademacher complexity $R(A)$, and $Z(I_A) = |A|$ is simply the cardinality of the set. Therefore we can use known quantitative relationships between $R(A)$ and $|A|$ from learning theory to estimate $|A| = Z(I_A)$ in terms of $R(A) = \mathcal{R}(I_A)$. Although not formulated in the framework of Rademacher complexity, this is the strategy used by Barvinok (1997).

Here, we generalize these results to general weight functions w and show that it is, in fact, possible to use $\mathcal{R}(w)$ to obtain estimates of $Z(w)$. This observation can be turned into an algorithm by observing that $\mathcal{R}(w)$ is the expectation of a random variable concentrated around its mean. Therefore, as we will show in Proposition 2, a small number of samples suffices to reliably estimate $\mathcal{R}(w)$ (and hence, $Z(w)$) with high probability. Whenever w is 'sufficiently nice' and we have access to an optimization oracle, the estimation algorithm is efficient.

Algorithm 1 Rademacher Estimate of $\log Z(w)$

Inputs: A positive integer k and weight function $w : \{-1, 1\}^n \to [0, \infty)$.
Output: A number $\bar{\delta}_k(w)$ which approximates $\log Z(w) = \log\left(\sum_{\mathbf{x}\in\{-1,1\}^n} w(\mathbf{x})\right)$.

1. Sample k vectors $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_k$ independently and uniformly from $\{-1, 1\}^n$.
2. Apply the optimization oracle of assumption 1 to each vector $\mathbf{c}$ and compute the mean
$$\bar{\delta}_k(w) = \frac{1}{k}\sum_{i=1}^{k} \max_{\mathbf{x}\in\{-1,1\}^n} \{\langle \mathbf{c}_i, \mathbf{x}\rangle + \log w(\mathbf{x})\}.$$
3. Output $\bar{\delta}_k(w)$ as an estimator of $\mathcal{R}(w)$ and thus $\log Z(w)$.

Bounding Weighted Rademacher Complexity

The weighted Rademacher complexity is an expectation over optimization problems. The optimization problem is defined by sampling a vector, or direction since all have length $\sqrt{n}$, uniformly from $\{-1, 1\}^n$ and finding the vector $\mathbf{x}$ that is most aligned (largest dot product) after adding $\log w(\mathbf{x})$. Our first objective is to derive bounds on the weighted Rademacher complexity in terms of the sum $Z(w)$.

We begin with the observation that it is impossible to derive bounds on the Rademacher complexity in terms of set size that are *tight for sets of all shapes*. To gain intuition, note that in high dimensional spaces the dot product of any particular vector and another chosen uniformly at random from $\{-1, 1\}^n$ is close to 0 with high probability. The distribution of weight vectors throughout the space may take any geometric form. One extreme configuration is that all vectors with large weights are packed tightly together, forming a Hamming ball. At the other extreme, all vectors with large weights could be distributed uniformly through the space. As Figure 1 illustrates, a *large* set of tightly packed vectors and a *small* set of well-distributed vectors will both have similar Rademacher complexity. Thus, bounds on Rademacher complexity that are based on the underlying set's size fundamentally cannot always be tight for all distributions. Nevertheless, the lower and upper bounds we derive next are tight enough to be useful in practice.

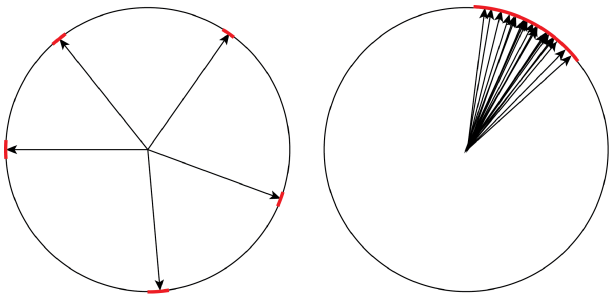

Figure 1: Illustration mapping a set of vectors in high dimensional space $\{-1,1\}^n$ to the unit circle. Red regions correspond to regions of space that have a large dot product with some vector in the set. Left: when the size of a set is small, very few regions have a large dot product with any vector in the set, so the Rademacher complexity will be small. Right: when a *large set* of vectors is tightly packed in a *small region* of space, the Rademacher complexity will remain relatively small. In both left and right figures we have similar (small) Rademacher complexities, yet different set sizes. This illustrates why tight bounds on the set size based on Rademacher complexity are difficult to achieve.

Lower bound. To lower bound the weighted Rademacher complexity we adapt the technique of (Barvinok 1997) for lower bounding the standard Rademacher complexity. The high level idea is that the space $\{-1,1\}^n$ can be mapped to the leaves of a binary search tree. By following a path from the root to a leaf, we are dividing the space in half n times, until we arrive at a leaf which corresponds to a single element (with some fixed weight). By judiciously choosing which half of the space (branch of the tree) to recurse into at each step we derive the bound in Lemma 1, whose proof is given in the appendix.

Lemma 1. *For any $\beta \in (0, 1/2)$, the weighted Rademacher complexity of a weight function $w : \{-1,1\}^n \to [0,\infty)$ is lower bounded by*

$$\mathcal{R}(w) \geq \log w^*(\beta) + \frac{n\log(1-\beta) + \log Z(w) - \log w^*(\beta)}{\log\left(\frac{1-\beta}{\beta}\right)},$$

with

$$w^*(\beta) = \begin{cases} w_{max} = \max_{\mathbf{x}} w(\mathbf{x}), & \text{if } \beta \geq 1/3 \\ w_{min} = \min_{\mathbf{x}}\{w(\mathbf{x}) : w(\mathbf{x}) > 0\}, & \text{if } \beta \leq 1/3 \end{cases}.$$

Upper bound. In the unweighted setting, a standard upper bound on the Rademacher complexity is used in learning theory to show that the Rademacher complexity of a small hypothesis class is also small, often to prove PAC-learnability. Massart's Lemma (see (Shalev-Shwartz and Ben-David 2014), lemma 26.8) formally upper bounds the Rademacher complexity in terms of the size of the set. This result is intuitive since, as we have noted, the dot product between any one vector $\mathbf{x} \in \{-1,1\}^n$ is small with most other vectors $\mathbf{c} \in \{-1,1\}^n$. Therefore, if the set is small the Rademacher complexity must also be small.

Adapting the proof technique of Massart's Lemma to the weighted setting we arrive at the following bound, whose proof may be found in the appendix:

Lemma 2. *For any $\lambda > 0$, $\gamma > 0$, and weight functions $w, w^\gamma : \{-1,1\}^n \to [0,\infty)$ with $w^\gamma(\mathbf{x}) = w(\mathbf{x})^\gamma$, the weighted Rademacher complexity of w^γ is upper bounded by*

$$\mathcal{R}(w^\gamma) \leq \frac{1}{\lambda}\log Z(w) + \frac{\lambda\gamma - 1}{\lambda}\log w^*(\lambda) + \lambda\frac{n}{2}, \quad (4)$$

with

$$w^*(\lambda,\gamma) = \begin{cases} w_{max} = \max_{\mathbf{x}} w(\mathbf{x}), & \text{if } \lambda\gamma \geq 1 \\ w_{min} = \min_{\mathbf{x}}\{w(\mathbf{x}) : w(\mathbf{x}) > 0\}, & \text{if } \lambda\gamma \leq 1 \end{cases}.$$

Note that for an indicator weight function we recover the bound from Massart's Lemma by setting $\lambda = \sqrt{\frac{2\log Z}{n}}$.

Corollary 2.1. *In the limit as $\gamma \to \infty$ by setting*

$$\lambda = \sqrt{\frac{2\log\frac{Z(w)}{w_{max}}}{n}}.$$

we recover the bound $w_{max} \leq Z(w)$ in Lemma 2.

Lemma 2 holds for any $\lambda > 0$. In general we set $\gamma = 1$ and optimize over λ to make the bound as tight as possible, comparing the result with the trivial bound given by Corollary 2.1. More sophisticated optimization strategies over λ and γ could result in a tighter bound.

Bounding the Weighted Sum $Z(w)$

With our bounds on the weighted Rademacher complexity from the previous section, we now present our method for efficiently bounding the sum $Z(w)$. Proposition 2 states that we can estimate the weighted Rademacher complexity using the optimization oracle of assumption 1.

Proposition 2. *For $\mathbf{c} \in \{-1,1\}^n$ sampled uniformly at random, the bound*

$$\mathcal{R}(w) - \sqrt{6n} \leq \delta(\mathbf{c}, w) \leq \mathcal{R}(w) + \sqrt{6n} \quad (5)$$

holds with probability greater than .95.

Proof. By applying Proposition 1 to the function $f_w(\mathbf{c}) = \delta(\mathbf{c}, w)$, and noting the constant $d_i = 2$, we have

$$P[|\delta(\mathbf{c}, w) - \mathcal{R}(w)| > \sqrt{6n}] \leq e^{-3} \leq .05.$$

This finishes the proof. □

To bound $Z(w)$ we use our optimization oracle to solve a perturbed optimization problem, giving an estimate of the weighted Rademacher complexity, $\mathcal{R}(w)$. Next we invert the bounds on $\mathcal{R}(w)$ (Lemmas 1 and 2) to obtain bounds on $Z(w)$. We optimize the parameters λ and β (from equations 1 and 2) to make the bounds as tight as possible. By applying our optimization oracle repeatedly, we can reduce the slack introduced in our final bound when estimating $\mathcal{R}(w)$ (by Lemma 2) and arrive at our bounds on the sum $Z(w)$, stated in the following theorem.

Theorem 1. *With probability at least 0.95, the sum $Z(w) = \sum_{\mathbf{x}\in\{-1,1\}^n} w(\mathbf{x})$ of any weight function $w : \{-1,1\}^n \to [0,\infty)$ is bounded by the outputs of algorithms 2 and 3 as*

$$\psi_{LB} < \log Z(w) < \psi_{UB}.$$

Algorithm 2 Rademacher Lower Bound for $\log Z(w)$

Inputs: The estimator $\bar{\delta}_k(w)$ output by algorithm 1, k used to compute $\bar{\delta}_k(w)$, and optionally w_{min} and w_{max}.
Output: A number ψ_{LB} which lower bounds $\log Z(w)$.

1. If $\log w_{min}$ was provided as input, calculate

$$\lambda = \frac{\bar{\delta}_k(w) - \sqrt{\frac{6n}{k}} - \log w_{min}}{n}$$

2. If $\log w_{min}$ was provided as input and $\lambda \leq 1$,

$$\psi_{LB} = \frac{(\bar{\delta}_k(w) - \sqrt{\frac{6n}{k}} - \log w_{min})^2}{2n} + \log w_{min}.$$

3. Otherwise,

$$\psi_{LB} = \bar{\delta}_k(w) - \sqrt{\frac{6n}{k}} - \frac{n}{2}.$$

4. Output the lower bound $\max\{\psi_{LB}, \log w_{max}\}$.

Algorithm 3 Rademacher Upper Bound for $\log Z(w)$

Inputs: The estimator $\bar{\delta}_k(w)$, k used to compute $\bar{\delta}_k(w)$, and optionally w_{min} and w_{max}.
Output: A number ψ_{UB} which upper bounds $\log Z(w)$.

1. If w_{min} was provided as input, calculate

$$\beta_{min} = \frac{\bar{\delta}_k(w) + \sqrt{\frac{6n}{k}} - \log w_{min}}{n}.$$

2. If w_{max} was provided as input, calculate

$$\beta_{max} = \frac{\bar{\delta}_k(w) + \sqrt{\frac{6n}{k}} - \log w_{max}}{n}.$$

3. Set the value

$$\beta_{opt} = \begin{cases} \beta_{min}, & \text{if } 0 < \beta_{min} < \frac{1}{3} \\ \beta_{max}, & \text{if } \frac{1}{3} < \beta_{max} < \frac{1}{2} \\ \frac{1}{2}, & \text{if } \frac{1}{2} < \beta_{max} \\ \frac{1}{3}, & \text{otherwise} \end{cases}.$$

4. Output the upper bound ψ_{UB}:
   (a) If $\beta_{opt} = \frac{1}{3}$, $\psi_{UB} = \bar{\delta}_k(w) + \sqrt{\frac{6n}{k}} + n \log\left(\frac{3}{2}\right)$.
   (b) If $\beta_{opt} = \frac{1}{2}$, $\psi_{UB} = n + \log w_{max}$.
   (c) Otherwise,

$$\psi_{UB} = n\beta_{opt} \log\left(\frac{1-\beta_{opt}}{\beta_{opt}}\right) - n \log(1-\beta_{opt}) + \log w^*,$$

$$\text{where } w^* = \begin{cases} w_{min}, & \text{if } \beta_{opt} < \frac{1}{3} \\ w_{max}, & \text{if } \beta_{opt} > \frac{1}{3} \end{cases}.$$

Experiments

The closest line of work to this paper showed that the partition function can be bounded by solving an optimization problem perturbed by Gumbel random variables (Hazan and Jaakkola 2012; Hazan, Maji, and Jaakkola 2013; Hazan et al. 2016; Kim, Sabharwal, and Ermon 2016; Balog et al. 2017). This approach is based on the fact that

$$\ln Z(w) = \mathbb{E}_\gamma \left[\max_{\mathbf{x} \in \{-1,1\}^n} \{\ln w(\mathbf{x}) + \gamma(\mathbf{x})\} \right],$$

where *all* 2^n random variables $\gamma(\mathbf{x})$ are sampled from the Gumbel distribution with scale 1 and shifted by the Euler-Mascheroni constant to have mean 0. Perturbing all 2^n states with IID Gumbel random variables is intractable, leading the authors to bound $\ln Z(w)$ by perturbing states with a combination of low dimensional Gumbel perturbations. Specifically the upper bound

$$\ln Z(w) \leq \Theta_{UB} = \mathbb{E}_\gamma \left[\max_{\mathbf{x} \in \{-1,1\}^n} \left\{ \ln w(\mathbf{x}) + \sum_{i=1}^n \gamma_i(x_i) \right\} \right]$$

(Hazan et al. 2016) and lower bound

$$\ln Z(w) \geq \Theta_{LB} = \mathbb{E}_\gamma \left[\max_{\mathbf{x} \in \{-1,1\}^n} \left\{ \ln w(\mathbf{x}) + \sum_{i=1}^n \frac{1}{n} \gamma_i(x_i) \right\} \right]$$

(Balog et al. 2017, p. 6) hold in expectation, where $\gamma_i(x)$ for $i = 1, \ldots, n$ are sampled from the Gumbel distribution with scale 1 and shifted by the Euler-Mascheroni constant to have mean 0.

To obtain bounds that hold with high probability using Gumbel perturbations we calculate the slack term (Hazan et al. 2016, p. 32)

$$\epsilon_g = \min \left\{ 2\sqrt{n} \left(1 + \sqrt{\frac{1}{2k} \ln \frac{2}{\alpha}} \right)^2, \sqrt{n} \max \left\{ \frac{4}{k} \ln \frac{2}{\alpha}, \sqrt{\frac{32}{k} \ln \frac{2}{\alpha}} \right\} \right\},$$

giving upper and lower bounds $\theta_{UB} = \Theta_{UB} + \epsilon_g$ and $\theta_{LB} = \Theta_{LB} - \frac{\epsilon_g}{n}$ that hold with probability $1 - \alpha$ where k samples are used to estimate the expectation bounds.

We note the Gumbel expectation upper bound takes nearly the same form as the weighted Rademacher complexity, with two differences. The perturbation is sampled from a Gumbel distribution instead of a dot product with a vector of Rademacher random variables and, without scaling, the two bounds are naturally written in different log bases.

We experimentally compare our bounds with those obtained by Gumbel perturbations on two models. First we bound the partition function of the spin glass model from (Hazan et al. 2016). For this problem the weight function is given by the unnormalized probability distribution of the spin glass model. Second we bound the propositional model counts (#SAT) for a variety of SAT problems. This problem falls into the unweighted category where every weight is either 0 or 1, specifically every satisfying assignment has weight 1 and we bound the total number of satisfying assignments.

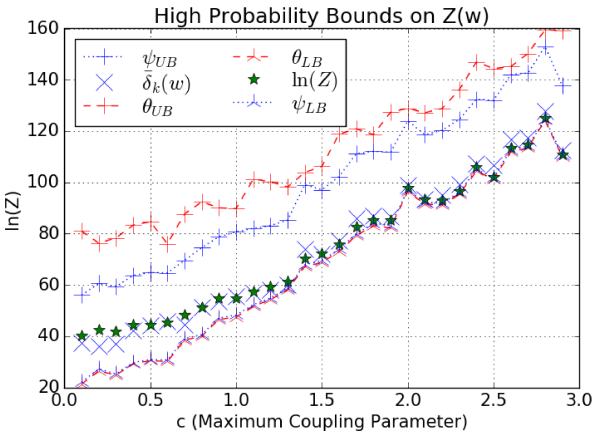

Figure 2: Bounds for a 7x7 spin glass model with $k = 5$ (for both methods), that hold with probability .95. Our bounds and estimator are scaled to match Gumbel log base e bounds.

Spin Glass Model

Following (Hazan et al. 2016), we bound the partition function of a spin glass model with variables $x_i \in \{-1, 1\}$ for $i = 1, 2, \ldots, n$, where each variable represents a spin. Each spin has a local field parameter θ_i which corresponds to its local potential function $\theta_i(x_i) = \theta_i x_i$. We performed experiments on grid shaped models where each spin variable has 4 neighbors, unless it occupies a grid edge. Neighboring spins interact with coupling parameters $\theta_{i,j}(x_i, x_j) = \theta_{i,j} x_i x_j$. The potential function of the spin glass model is

$$\theta(x_1, x_2, \ldots, x_n) = \sum_{i \in V} \theta_i x_i + \sum_{(i,j) \in E} \theta_{i,j} x_i x_j,$$

with corresponding weight function

$$w(\mathbf{x}) = \exp\left(\sum_{i \in V} \theta_i x_i + \sum_{(i,j) \in E} \theta_{i,j} x_i x_j \right).$$

We compare our bounds on a 7x7 spin glass model. We sampled the local field parameters θ_i uniformly at random from $[-1, 1]$ and the coupling parameters uniformly at random from $[0, c)$ with c varying. Non-negative coupling parameters make it possible to perform MAP inference efficiently using the graph-cuts algorithm (Kolmogorov and Zabin 2004; Greig, Porteous, and Seheult 1989). We used the python maxflow module wrapping the implementation from Boykov and Kolmogorov (2004).

Figure 2 shows bounds that hold with probability .95, where all bounds are computed with $k = 5$. For this value of k, our approach produces tighter upper bounds than using Gumbel perturbations. The crossover to a tighter Gumbel perturbation upper bound occurs around $k \approx 15$. Lower bounds are equivalent, although we note it is trivial to recover this bound by simply calculating the largest weight over all states.

Propositional Model Counting

Next we evaluate our method on the problem of propositional model counting. Given a boolean formula F, this poses the question of how many assignments $\mathbf{x}$ to the underlying boolean variables result in F evaluating to true. Our weight function is given by $w(\mathbf{x}) = 1$ if $F(\mathbf{x})$ evaluates to true, and 0 otherwise.

We performed MAP inference on the perturbed problem using the weighted partial MaxSAT solver MaxHS (Davies 2013). Ground truth was obtained for a variety of models[1] using three exact propositional model counters (Thurley 2006; Sang et al. 2004; Oztok and Darwiche 2015)[2]. Table 1 shows bounds that hold with probability .95 and $k = 1$. While the Gumbel lower bounds are always trivial, we produce non-trivial lower bounds for several model instances. Our upper bounds are generally tighter than Gumbel upper bounds.

Analysis

Our bounds are much looser than those computed by randomized hashing schemes (Chakraborty, Meel, and Vardi 2013; Ermon et al. 2013d; 2013b; Zhao et al. 2016), but also require much less computation (Ermon et al. 2013c; Achim, Sabharwal, and Ermon 2016). While our approach provides polynomial runtime guarantees for MAP inference in the spin glass model after random perturbations have been applied, randomized hashing approaches do not. For propositional model counting, we found that our method is computationally cheaper by over 2 orders of magnitude than results reported in Zhao et al. (2016). Additionally, we tried reducing the runtime and accuracy of randomized hashing schemes by running code from Zhao et al. (2016) with f values of 0, .01, .02, .03, .04, and .05. We set the maximum time limit to 1 hour (while our method required .01 to 6 seconds of computation for reported results). Throughout experiments on models reported in Table 1 our approach still generally required orders of magnitude less computation and also found tighter bounds in some instances.

Empirically, our lower bounds were comparable to or tighter than those obtained by Gumbel perturbations on both models. The weighted Rademacher complexity is generally at least as good an estimator of $\log Z$ as the Gumbel upper bound, however it is only an estimator and not an upper bound. Our upper bound using the weighted Rademacher complexity, which holds in expectation, is empirically weaker than the corresponding Gumbel expectation upper bound. However, the slack term needed to transform our expectation bound into a high probability bound is tighter than the corresponding Gumbel slack term. Since both slack terms approach 0 in the limit of infinite computation ($k = \infty$, the number of samples used to estimate the expectation bound), this can result in a trade-off where we produce a tighter upper bound up to some value of k, after which the Gumbel bound becomes tighter.

[1]The models can be downloaded from http://reasoning.cs.ucla.edu/c2d/results.html

[2]Precomputed counts were downloaded from https://sites.google.com/site/marcthurley/sharpsat/benchmarks/collected-model-counts

Model Name	#Variables	#Clauses	ln(Z)	$\bar{\delta}_k(w)$	Rademacher ψ_{UB}	Gumbel θ_{UB}	Rademacher ψ_{LB}	Gumbel θ_{LB}
log-1.cnf	939	3785	47.8	46.4	**361.7**	400.9	**1.0**	-0.3
log-2.cnf	1337	24777	24.2	58.9	**466.2**	508.0	**0.9**	-0.2
log-3.cnf	1413	29487	26.4	31.2	**418.4**	524.0	**1.8**	-0.2
log-4.cnf	2303	20963	65.3	110.2	743.1	**615.6**	**0.4**	-0.2
tire-1.cnf	352	1038	20.4	36.0	**191.1**	262.9	**0.6**	-0.4
tire-2.cnf	550	2001	27.3	61.0	286.5	**261.5**	**0.2**	-0.5
tire-3.cnf	577	2004	26.1	53.4	**284.3**	299.1	**0.4**	-0.4
tire-4.cnf	812	3222	32.3	47.1	**334.7**	411.6	**0.8**	-0.3
ra.cnf	1236	11416	659.2	625.2	**856.7**	1074.5	**48.8**	0.2
rb.cnf	1854	11324	855.9	852.6	**1285.1**	1403.9	**63.6**	0.2
sat-grid-pbl-0010.cnf	110	191	54.7	51.3	**76.2**	176.6	**0.4**	-0.5
sat-grid-pbl-0015.cnf	240	436	125.4	113.7	**166.4**	308.6	**3.5**	-0.1
sat-grid-pbl-0020.cnf	420	781	220.4	210.7	**291.1**	497.4	**10.9**	0.1
sat-grid-pbl-0025.cnf	650	1226	348.3	346.6	**450.5**	668.2	**24.3**	0.2
sat-grid-pbl-0030.cnf	930	1771	502.4	483.8	**644.6**	896.7	**36.5**	0.2
c432.isc	196	514	25.0	45.7	**132.5**	212.4	**0.0**	-0.5
c499.isc	243	714	28.4	60.3	**164.0**	221.9	**0.0**	-0.5
c880.isc	417	1060	41.6	92.2	**267.6**	337.5	**0.2**	-0.3
c1355.isc	555	1546	28.4	65.8	**294.3**	345.2	**0.1**	-0.3
c1908.isc	751	2053	22.9	93.6	**389.9**	407.0	**0.0**	-0.3
c2670.isc	1230	2876	161.5	248.1	719.3	**705.4**	**3.0**	-0.1

Table 1: Empirical comparison of our estimate of and bounds on propositional model counts against bounds based on Gumbel perturbations. Meta column descriptions, left to right: model name and information, natural logarithm of ground truth model counts and our estimator, upper bounds, and lower bounds. Bounds hold with probability .95 and $k = 1$ for both methods. Tighter bounds are in **bold**.

Conclusion

We introduced the weighted Rademacher complexity, a novel generalization of Rademacher complexity. We showed that this quantity can be used as an estimator of the size of a weighted set, and gave bounds on the weighted Rademacher complexity in terms of the weighted set size. This allowed us to bound the sum of any non-negative weight function, such as the partition function, in terms of the weighted Rademacher complexity. We showed how the weighted Rademacher complexity can be efficiently approximated whenever an efficient optimization oracle exists, as is the case for a variety of practical problems including calculating the partition function of certain graphical models and the permanent of non-negative matrices. Experimental evaluation demonstrated that our approach provides tighter bounds than competing methods under certain conditions.

In future work our estimator $\mathcal{R}(w)$ and bounds on $Z(w)$ may be generalized to other forms of randomness. Rather than sampling $\mathbf{c}$ uniformly from $\{-1, 1\}^n$, we could conceivably sample each element c_i from some other distribution, such as the uniform distribution over $[-1, 1]$, a Gaussian, or Gumbel. Our bounds should readily adapt to continuous uniform or gaussian distributions, although derivations may be more complex in general. As another line of future work, the weighted Rademacher complexity may be useful beyond approximate inference to learning theory.

Acknowledgments

We gratefully acknowledge funding from Ford, FLI and NSF grants #1651565, #1522054, #1733686. We also thank Tri Dao, Aditya Grover, Rachel Luo, and anonymous reviewers.

References

Achim, T.; Sabharwal, A.; and Ermon, S. 2016. Beyond parity constraints: Fourier analysis of hash functions for inference. In *International Conference on Machine Learning*, 2254–2262.

Bach, F., et al. 2013. Learning with submodular functions: A convex optimization perspective. *Foundations and Trends® in Machine Learning* 6(2-3):145–373.

Balog, M.; Tripuraneni, N.; Ghahramani, Z.; and Weller, A. 2017. Lost relatives of the Gumbel trick. In *34th International Conference on Machine Learning*, 371–379.

Barvinok, A. I. 1997. Approximate counting via random optimization. *Random Structures and Algorithms* 11(2):187–198.

Bellman, R. E. 1961. *Adaptive control processes: a guided tour*. Princeton university press.

Bezáková, I.; Štefankovič, D.; Vazirani, V. V.; and Vigoda, E. 2006. Accelerating simulated annealing for the permanent and combinatorial counting problems. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, 900–907.

Boykov, Y., and Kolmogorov, V. 2004. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE transactions on pattern analysis and machine intelligence* 26(9):1124–1137.

Chakrabarty, D.; Jain, P.; and Kothari, P. 2014. Provable submodular minimization using wolfe's algorithm. In *Advances in Neural Information Processing Systems*, 802–809.

Chakraborty, S.; Meel, K. S.; and Vardi, M. Y. 2013. A

scalable approximate model counter. In *International Conference on Principles and Practice of Constraint Programming*, 200–216. Springer.

Davies, J. 2013. *Solving MAXSAT by Decoupling Optimization and Satisfaction*. Ph.D. Dissertation, University of Toronto.

Ermon, S.; Gomes, C.; Sabharwal, A.; and Selman, B. 2013a. Taming the curse of dimensionality: Discrete integration by hashing and optimization. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, 334–342.

Ermon, S.; Gomes, C. P.; Sabharwal, A.; and Selman, B. 2013b. Embed and project: Discrete sampling with universal hashing. In *Advances in Neural Information Processing Systems (NIPS)*, 2085–2093.

Ermon, S.; Gomes, C. P.; Sabharwal, A.; and Selman, B. 2013c. Optimization with parity constraints: From binary codes to discrete integration. In *Proc. of the 29th Conference on Uncertainty in Artificial Intelligence (UAI)*.

Ermon, S.; Gomes, C. P.; Sabharwal, A.; and Selman, B. 2013d. Taming the curse of dimensionality: Discrete integration by hashing and optimization. In *Proc. of the 30th International Conference on Machine Learning (ICML)*.

Ermon, S.; Gomes, C.; Sabharwal, A.; and Selman, B. 2014. Low-density parity constraints for hashing-based discrete integration. In *International Conference on Machine Learning*, 271–279.

Fujishige, S. 1980. Lexicographically optimal base of a polymatroid with respect to a weight vector. *Mathematics of Operations Research* 5(2):186–196.

Greig, D. M.; Porteous, B. T.; and Seheult, A. H. 1989. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society. Series B (Methodological)* 271–279.

Hazan, T., and Jaakkola, T. S. 2012. On the partition function and random maximum a-posteriori perturbations. In Langford, J., and Pineau, J., eds., *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, 991–998. New York, NY, USA: ACM.

Hazan, T.; Orabona, F.; Sarwate, A. D.; Maji, S.; and Jaakkola, T. 2016. High dimensional inference with random maximum a-posteriori perturbations. *arXiv preprint arXiv:1602.03571*.

Hazan, T.; Maji, S.; and Jaakkola, T. 2013. On sampling from the Gibbs distribution with random maximum a-posteriori perturbations. In *Advances in Neural Information Processing Systems*, 1268–1276.

Hopcroft, J. E., and Karp, R. M. 1971. A $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. In *Switching and Automata Theory, 1971., 12th Annual Symposium on*, 122–125. IEEE.

Jegelka, S.; Lin, H.; and Bilmes, J. A. 2011. On fast approximate submodular minimization. In *Advances in Neural Information Processing Systems*, 460–468.

Jerrum, M., and Sinclair, A. 1996. The markov chain monte carlo method: an approach to approximate counting and integration. *Approximation algorithms for NP-hard problems* 482–520.

Jerrum, M.; Sinclair, A.; and Vigoda, E. 2004. A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. *Journal of the ACM (JACM)* 51(4):671–697.

Jonker, R., and Volgenant, A. 1987. A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing* 38(4):325–340.

Jordan, M. I.; Ghahramani, Z.; Jaakkola, T. S.; and Saul, L. K. 1998. An introduction to variational methods for graphical models. *NATO ASI SERIES D BEHAVIOURAL AND SOCIAL SCIENCES* 89:105–162.

Kim, C.; Sabharwal, A.; and Ermon, S. 2016. Exact sampling with integer linear programs and random perturbations. In *Proc. 30th AAAI Conference on Artificial Intelligence*.

Koller, D., and Friedman, N. 2009. *Probabilistic graphical models: principles and techniques*. MIT press.

Kolmogorov, V., and Zabin, R. 2004. What energy functions can be minimized via graph cuts? *IEEE transactions on pattern analysis and machine intelligence* 26(2):147–159.

Kuhn, H. W. 1955. The hungarian method for the assignment problem. *Naval Research Logistics (NRL)* 2(1-2):83–97.

Madras, N. N. 2002. *Lectures on monte carlo methods*, volume 16. American Mathematical Soc.

McDiarmid, C. 1989. On the method of bounded differences. *Surveys in combinatorics* 141(1):148–188.

Mussmann, S., and Ermon, S. 2016. Learning and inference via maximum inner product search. In *International Conference on Machine Learning*, 2587–2596.

Mussmann, S.; Levy, D.; and Ermon, S. 2017. Fast amortized inference and learning in log-linear models with randomly perturbed nearest neighbor search. *UAI*.

Orlin, J. B. 2009. A faster strongly polynomial time algorithm for submodular function minimization. *Mathematical Programming* 118(2):237–251.

Oztok, U., and Darwiche, A. 2015. A top-down compiler for sentential decision diagrams. In *IJCAI*, 3141–3148.

Sang, T.; Bacchus, F.; Beame, P.; Kautz, H. A.; and Pitassi, T. 2004. Combining component caching and clause learning for effective model counting. In *SAT*.

Shalev-Shwartz, S., and Ben-David, S. 2014. *Understanding machine learning: From theory to algorithms*. Cambridge university press.

Thurley, M. 2006. sharpsat-counting models with advanced component caching and implicit bcp. In *SAT*, 424–429.

Valiant, L. G. 1979. The complexity of enumeration and reliability problems. *SIAM Journal on Computing* 8(3):410–421.

Wainwright, M. J.; Jordan, M. I.; et al. 2008. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning* 1(1–2):1–305.

Zhao, S.; Chaturapruek, S.; Sabharwal, A.; and Ermon, S. 2016. Closing the gap between short and long xors for model counting. In *AAAI*, 3322–3329.