entropy

MDPI

Article

Variational Characterizations of Local Entropy and Heat Regularization in Deep Learning

Nicolas García Trillos [1], Zachary Kaplan [2] and Daniel Sanz-Alonso [3,*]

[1] Department of Statistics, University of Wisconsin Madison, Madison, WI 53706, USA; garciatrillo@wisc.edu
[2] Division of Applied Mathematics, Brown University, Providence, RI 02906, USA; zachary_kaplan@brown.edu
[3] Department of Statistics, University of Chicago, Chicago, IL 60637, USA
* Correspondence: sanzalonso@uchicago.edu

Received: 7 March 2019; Accepted: 11 May 2019; Published: 20 May 2019

Abstract: The aim of this paper is to provide new theoretical and computational understanding on two loss regularizations employed in deep learning, known as local entropy and heat regularization. For both regularized losses, we introduce variational characterizations that naturally suggest a two-step scheme for their optimization, based on the iterative shift of a probability density and the calculation of a best Gaussian approximation in Kullback–Leibler divergence. Disregarding approximation error in these two steps, the variational characterizations allow us to show a simple monotonicity result for training error along optimization iterates. The two-step optimization schemes for local entropy and heat regularized loss differ only over which argument of the Kullback–Leibler divergence is used to find the best Gaussian approximation. Local entropy corresponds to minimizing over the second argument, and the solution is given by moment matching. This allows replacing traditional backpropagation calculation of gradients by sampling algorithms, opening an avenue for gradient-free, parallelizable training of neural networks. However, our presentation also acknowledges the potential increase in computational cost of naive optimization of regularized costs, thus giving a less optimistic view than existing works of the gains facilitated by loss regularization.

Keywords: deep learning; local entropy; heat regularization; variational characterizations; Kullback–Leibler approximations; monotonic training

1. Introduction

The development and assessment of optimization methods for the training of deep neural networks has brought forward novel questions that call for new theoretical insights and computational techniques [1]. The performance of a network is determined by its ability to generalize, and choosing the network parameters by finding the global minimizer of the loss may not be only unfeasible, but also undesirable. In fact, training to a prescribed accuracy with competing optimization schemes may lead, consistently, to different generalization errors [2]. A possible explanation is that parameters in flat local minima of the loss give better generalization [2–5] and that certain schemes favor convergence to wide valleys of the loss function. These observations have led to the design of algorithms that employ gradient descent on a regularized loss, actively seeking minima located in wide valleys of the original loss [5]. While it has been demonstrated that the flatness of minima cannot fully explain generalization in deep learning [6,7], there are various heuristic [8], theoretical [4], and empirical [5] arguments that support regularizing the loss. In this paper, we aim to provide new understanding on two such regularizations, referred to as local entropy and heat regularization.

Our first contribution is to introduce variational characterizations for both regularized loss functions. These characterizations, drawn from the literature on large deviations [9], naturally suggest

a two-step scheme for their optimization, based on the iterative shift of a probability density and the calculation of a best Gaussian approximation in Kullback–Leibler divergence. The schemes for both regularized losses differ only over the argument of the (asymmetric) Kullback–Leibler divergence that they minimize. Local entropy minimizes over the second argument, and the solution is given by moment matching; heat regularization minimizes over the first argument, and its solution is defined implicitly.

The second contribution of this paper is to investigate some theoretical and computational implications of the variational characterizations. On the theoretical side, we prove that if the best Kullback–Leibler approximations could be computed exactly, then the regularized losses are monotonically decreasing along the sequence of optimization iterates. This monotonic behavior suggests that the two-step iterative optimization schemes have the potential of being stable provided that the Kullback–Leibler minimizers can be computed accurately. On the computational side, we show that the two-step iterative optimization of local entropy agrees with gradient descent on the regularized loss provided that the learning rate matches the regularization parameter. Thus, the two-step iterative optimization of local entropy computes gradients implicitly in terms of expected values; this observation opens an avenue for gradient-free, parallelizable training of neural networks based on sampling. In contrast, the scheme for heat regularization finds the best Kullback–Leibler Gaussian approximation over the first argument, and its computation via stochastic optimization [10,11] involves evaluation of gradients of the original loss.

Finally, our third contribution is to perform a numerical case-study to assess the performance of various implementations of the two-step iterative optimization of local entropy and heat regularized functionals. These implementations differ in how the minimization of Kullback–Leibler is computed and the argument that is minimized. Our experiments suggest, on the one hand, that the computational overload of the regularized methods far exceeds the cost of performing stochastic gradient descent on the original loss. On the other hand, they also suggest that for moderate-sized architectures, where the best Kullback–Leibler Gaussian approximations can be computed effectively, the generalization error with regularized losses is more stable than for stochastic gradient descent over the original loss. For this reason, we investigate using stochastic gradient descent on the original loss for the first parameter updates and then switch to optimize over a regularized loss. We also investigate numerically the choice and scope of the regularization parameter. Our understanding upon conducting numerical experiments is that, while sampling-based optimization of local entropy has the potential of being practical if parallelization is exploited and backpropagation gradient calculations are expensive, existing implementations of regularized methods in standard architectures are more expensive than stochastic gradient descent and do not clearly outperform it.

Several research directions stem from this work. A broad one is to explore the use of local entropy and heat regularizations in complex optimization problems outside of deep learning, e.g., in the computation of maximum a posteriori estimates in high dimensional Bayesian inverse problems. A more concrete direction is to generalize the Gaussian approximations within our two-step iterative schemes and allow updating both the mean and covariance of the Gaussian measures. Finally, our paper highlights the unification that Gaussian Kullback–Leibler approximations gives to the loss regularizations for deep learning studied in [4,5,8]; however, a natural generalization is to consider the best Kullback–Leibler approximations over the exponential family [12] or the best approximations in other f-divergences.

The rest of the paper is organized as follows. Section 2 provides the background on optimization problems arising in deep learning and reviews various analytical and statistical interpretations of local entropy and heat regularized losses. In Section 3, we introduce the variational characterization of local entropy and derive from it a two-step iterative optimization scheme. Section 4 contains analogous developments for heat regularization. Our presentation in Section 4 is parallel to that in Section 3, as we aim to showcase the unity that comes from the variational characterizations of both loss functions.

Section 5 reviews various algorithms for Kullback–Leibler minimization, and we conclude in Section 6 with a numerical case study.

2. Background

Neural networks are revolutionizing numerous fields including image and speech recognition, language processing, and robotics [13,14]. Broadly, neural networks are parametric families of functions used to assign outputs to inputs. The parameters $x \in \mathbb{R}^d$ of a network are chosen by solving a non-convex optimization problem of the form:

$$\arg\min_x f(x) = \arg\min_x \frac{1}{N} \sum_{i=1}^{N} f_i(x), \tag{1}$$

where each f_i is a loss associated with a training example. Most popular training methods employ backpropagation (i.e., automatic differentiation) to perform some variant of gradient descent over the loss f. In practice, gradients are approximated using a random subsample of the training data known as the minibatch. Importantly, the accurate solution of the optimization problem (1) is not the end-goal of neural networks; their performance is rather determined by their generalization or testing error, that is by their ability to assign outputs to unseen examples accurately.

A substantial body of literature [1,5,7], has demonstrated that optimization procedures with similar training error may consistently lead to different testing error. For instance, large minibatch sizes have been shown to result in poor generalization [2]. Several explanations have been set forth, including overfitting, attraction to saddle points, and explorative properties [2]. A commonly-accepted theory is that flat local minima of the loss f lead to better generalization than sharp minima [2–5]. As noted in [6,7], this explanation is not fully convincing, as due to the high number of symmetries in deep networks, one can typically find many parameters that have different flatness, but define the same network. Further, reparameterization may alter the flatness of minima. While a complete understanding is missing, the observations above have prompted the development of new algorithms that actively seek minima in wide valleys of the loss f. In this paper, we provide new insights into the potential advantages of two such approaches, based on local entropy and heat regularization.

2.1. Background on Local Entropy Regularization

We will first study optimization of networks performed on a regularization of the loss f known as local entropy, given by:

$$F_\tau(x) := -\log\left(\int_{\mathbb{R}^d} \exp\big(-f(x')\big)\, \varphi_{x,\tau}(x')\, dx'\right), \tag{2}$$

where here and throughout, $\varphi_{x,\tau}$ denotes the Gaussian density in $\mathbb{R}^d$ with mean x and variance τI. For given τ, $F_\tau(x)$ averages the values of f focusing on a neighborhood of size τ. Thus, for $F_\tau(x)$ to be small, it is required that f is small throughout a τ-neighborhood of x. Note that F_τ is equivalent to f as $\tau \to 0$ and becomes constant as $\tau \to \infty$. Figure 1 shows that local entropy flattens sharp isolated minima and deepens wider minima.

A natural statistical interpretation of minimizing the loss f is in terms of maximum likelihood estimation. Given training data $\mathcal{D}$, one may define the likelihood function:

$$\rho_f(x|\mathcal{D}) \propto \exp\big(-f(x)\big). \tag{3}$$

Thus, minimizing f corresponds to maximizing the likelihood ρ_f. In what follows, we assume that ρ_f is normalized to integrate to 1. Minimization of local entropy can also be interpreted in statistical terms, now as computing a maximum marginal likelihood. Consider a Gaussian prior distribution $p(x'|x) = \varphi_{x,\tau}(x')$, indexed by a hyperparameter x, on the parameters x' of the neural

network. Moreover, assume a likelihood $p(x'|\mathcal{D}) \propto \exp\big(-f(x')\big)$ as in Equation (3). Then, minimizing local entropy corresponds to maximizing the marginal likelihood:

$$\begin{aligned} p(\mathcal{D}|x) &= \int p(\mathcal{D}|x')p(x'|x)dx' \\ &= \int \exp\big(-f(x')\big)\, \varphi_{x,\tau}(x')\, dx'. \end{aligned} \tag{4}$$

We remark that the right-hand side of Equation (4) is the convolution of the likelihood ρ_f with a Gaussian, and so, we have:

$$F_\tau(x) \propto -\log\Big(\rho_f * \varphi_{0,\tau}(x)\Big). \tag{5}$$

Thus, local entropy F_τ can be interpreted as a regularization of the likelihood ρ_f.

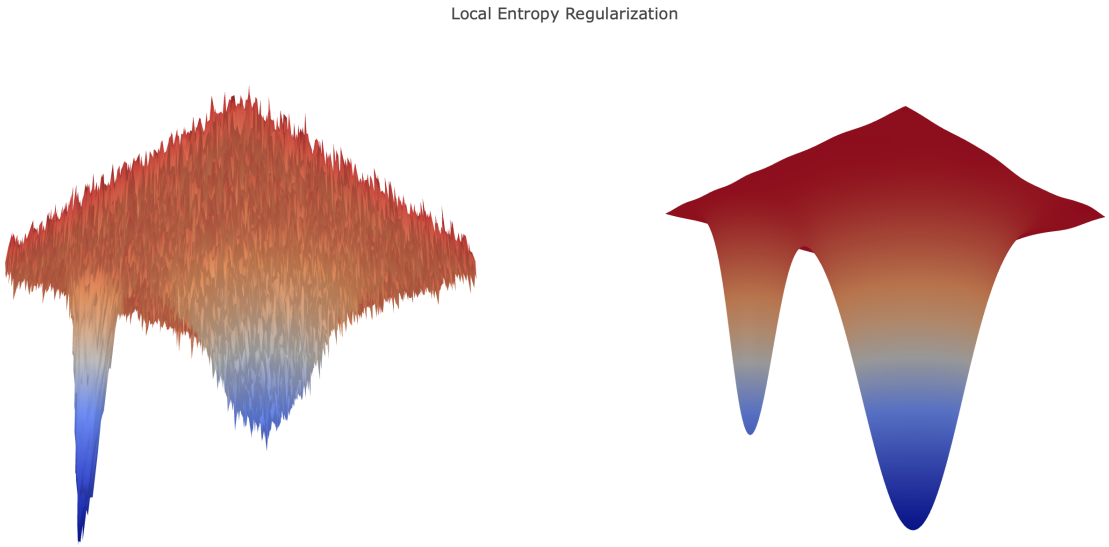

Figure 1. Toy example of local entropy regularization for a two-dimensional lost function. Note how the wider minima from the left figure deepens on the right, while the sharp minima become relatively shallower.

2.2. Background on Heat Regularization

We will also consider smoothing of the loss f through the heat regularization, defined by:

$$F^H_\tau(x) := \int_{\mathbb{R}^d} f(x')\varphi_{x,\tau}(x')\, dx'.$$

Note that F^H_τ regularizes the loss f directly, rather than the likelihood ρ_f:

$$F^H(x) = f * \varphi_{0,\tau}(x).$$

Local entropy and heat regularization are, clearly, rather different. Figure 2 shows that while heat regularization smooths the energy landscape, the relative macroscopic depth of local minima is marginally modified. Our paper highlights, however, the common underlying structure of the resulting optimization problems. Further analytical insights on both regularizations in terms of partial differential equations and optimal control can be found in [4].

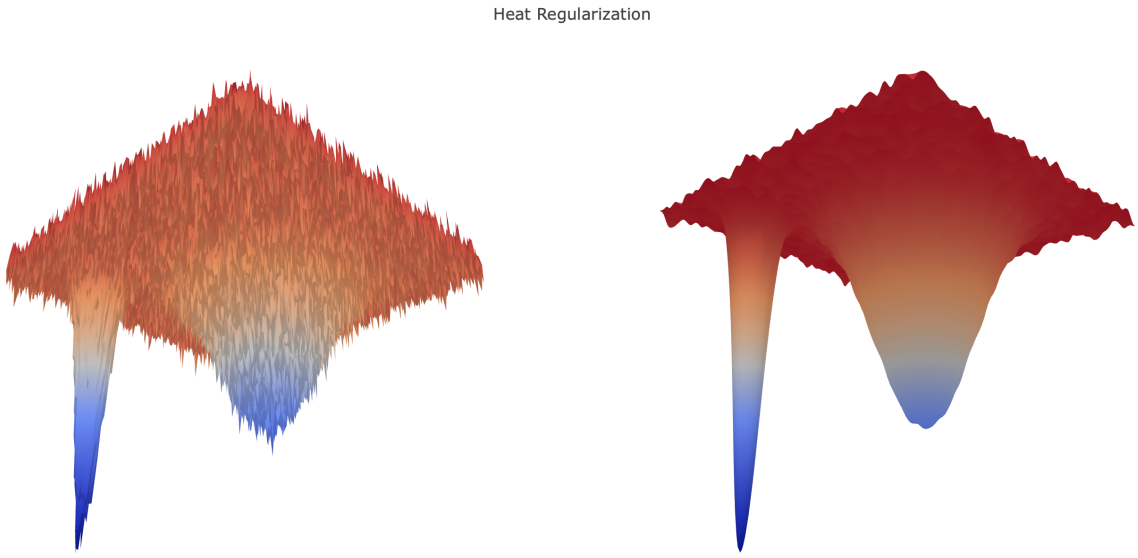

Figure 2. Toy example of heat regularization for a two-dimensional loss function. Here, the smoothing via convolution with a Gaussian amounts to a blur, altering the texture of the landscape without changing the location of deep minima.

2.3. Notation

For any $x \in \mathbb{R}^d$ and $\tau > 0$, we define the probability density:

$$q_{x,\tau}(x') := \frac{1}{Z_{x,\tau}} \exp\left(-f(x') - \frac{1}{2\tau}|x - x'|^2\right), \tag{6}$$

where $Z_{x,\tau}$ is a normalization constant. These densities will play an important role throughout.

We denote the Kullback–Leibler divergence between densities p and q in $\mathbb{R}^d$ by:

$$D_{\mathrm{KL}}(p\|q) := \int_{\mathbb{R}^d} \log\left(\frac{p(x)}{q(x)}\right) p(x)\, dx. \tag{7}$$

Kullback–Leibler is a divergence in that $D_{\mathrm{KL}}(p\|q) \geq 0$, with equality iff $p = q$. However, the Kullback–Leibler is not a distance as in particular, it is not symmetric; this fact will be relevant in the rest of this paper.

3. Local Entropy: Variational Characterization and Optimization

In this section, we introduce a variational characterization of local entropy. We will employ this characterization to derive a monotonic algorithm for its minimization. The following result is well known in large deviation theory [9]. We present its proof for completeness.

Theorem 1. *The local entropy admits the following variational characterization:*

$$\begin{aligned} F_\tau(x) &:= -\log\left(\int_{\mathbb{R}^d} \exp\left(-f(x')\right) \varphi_{x,\tau}(x') dx'\right) \\ &= \min_q \left\{\int_{\mathbb{R}^d} f(x') q(x')\, dx' + D_{\mathrm{KL}}(q\|\varphi_{x,\tau})\right\}. \end{aligned} \tag{8}$$

Moreover, the density $q_{x,\tau}$ defined in Equation (6) achieves the minimum in (8).

Proof. For any density q,

$$D_{\mathrm{KL}}(q\|q_{x,\tau}) = \int_{\mathbb{R}^d} f(x')q(x')\,dx' + D_{\mathrm{KL}}(q\|\varphi_{x,\tau}) + \log(Z_{x,\tau}). \tag{9}$$

Hence,

$$\begin{aligned} q_{x,\tau} &= \arg\min_q D_{\mathrm{KL}}(q\|q_{x,\tau}) \\ &= \arg\min \left\{ \int_{\mathbb{R}^d} f(x')q(x')\,dx' + D_{\mathrm{KL}}(q\|\varphi_{x,\tau}) + \log(Z_{x,\tau}) \right\} \\ &= \arg\min_q \left\{ \int_{\mathbb{R}^d} f(x')q(x')\,dx' + D_{\mathrm{KL}}(q\|\varphi_{x,\tau}) \right\}, \end{aligned}$$

showing that $q_{x,\tau}$ achieves the minimum. To conclude, note that $F_\tau(x) = -\log(Z_{x,\tau})$, and so, taking the minimum over q on both sides of Equation (9) and rearranging gives Equation (8). □

3.1. Two-Step Iterative Optimization

From the variational characterization (8), it follows that:

$$\arg\min_x F_\tau(x) = \arg\min_x \min_q \left\{ \int_{\mathbb{R}^d} f(x')q(x')\,dx' + D_{\mathrm{KL}}(q\|\varphi_{x,\tau}) \right\}. \tag{10}$$

Thus, a natural iterative approach to finding the minimizer of F_τ is to alternate between: (i) minimization of the term in curly brackets over densities q; and (ii) finding the associated minimizer over x. For the former, we can employ the explicit formula given by Equation (6), while for the latter, we note that the integral term does not depend on the variable x, and that the minimizer of the map:

$$x \mapsto D_{\mathrm{KL}}(q_{x_k,\tau}\|\varphi_{x,\tau})$$

is unique and given by the expected value of $q_{x_k,\tau}$. The statistical interpretation of these two steps is perhaps most natural through the variational formulation of the Bayesian update [15]: the first step finds a posterior distribution associated with likelihood $\rho_f \propto \exp(-f)$ and prior $\varphi_{x,\tau}$; the second computes the posterior expectation, which is used to define the prior mean in the next iteration. It is worth noting the parallel between this two-step optimization procedure and the empirical Bayes interpretation of local entropy mentioned in Section 2.

In short, the expression (10) suggests a simple scheme for minimizing local entropy, as illustrated in Algorithm 1. In practice, the expectation in the second step of the algorithm needs to be approximated. We will explore the potential use of gradient-free sampling schemes in Section 5.2 and in our numerical experiments.

A seemingly unrelated approach to minimizing the local entropy F_τ is to employ gradient descent and set:

$$x_{k+1} = x_k - \eta \nabla F_\tau(x_k), \tag{11}$$

where η is a learning rate. We now show that the iterates $\{x_k\}_{k=0}^K$ given by Algorithm 1 agree with those given by gradient descent with learning rate $\eta = \tau$.

Remark 1. *In this paper, we restrict our attention to the update scheme* (11) *with* $\eta = \tau$. *For this choice of learning rate, we can deduce theoretical monotonicity according to Theorem 2 below, but it may be computationally advantageous to use* $\eta \neq \tau$ *as explored in [5].*

Algorithm 1 Minimization of local entropy F_τ through optimization with respect to the second argument of the Kullback–Leibler divergence.

Choose $x_0 \in \mathbb{R}^d$ and for $k = 0, \ldots, K-1$ do:

1. Define $q_{x_k,\tau}$ as in Equation (6).
2. Define x_{k+1} as the minimizer, $\mathbb{E}_{X\sim q_{x_k,\tau}}(X)$, of the map:

$$x \mapsto D_{\mathrm{KL}}(q_{x_k,\tau} \| \varphi_{x,\tau}).$$

By direct computation:

$$\nabla F_\tau(x) = \frac{1}{\tau}\Big(x - \mathbb{E}_{X\sim q_{x,\tau}}(X)\Big).$$

Therefore,

$$\nabla F_\tau(x_k) = \frac{1}{\tau}\Big(x_k - \mathbb{E}_{X\sim q_{x_k,\tau}}(X)\Big) = \frac{1}{\tau}(x_k - x_{k+1}), \tag{12}$$

establishing that Algorithm 1 performs gradient descent with learning rate τ. This choice of learning rate leads to a monotonic decrease of local entropy, as we show in the next subsection.

3.2. Majorization-Minorization and Monotonicity

We now show that Algorithm 1 is a majorization-minimization algorithm. Let:

$$A(x,\tilde{x}) := \int_{\mathbb{R}^d} f(x') q_{\tilde{x},\tau}(x')\, dx' + D_{\mathrm{KL}}(q_{\tilde{x},\tau} \| \varphi_{x,\tau}),$$

where $q_{\tilde{x},\tau}$ is as in (6). It follows that $A(x,x) = F_\tau(x)$ for all $x \in \mathbb{R}^d$ and that $A(x,\tilde{x}) \geq F_\tau(x)$ for arbitrary $x, \tilde{x}$; in other words, A is a majorizer for F_τ. In addition, it is easy to check that the updates:

$$x_{k+1} = \arg\min_x A(x, x_k)$$

coincide with the updates in Algorithm 1. As a consequence, we have the following theorem.

Theorem 2 (Monotonicity and stationarity of Algorithm 1)**.** *The sequence* $\{x_k\}_{k=0}^K$ *generated by Algorithm 1 satisfies:*

$$F_\tau(x_k) \leq F_\tau(x_{k-1}), \quad 1 \leq k \leq K.$$

Moreover, equality holds only when x_k *is a critical point of* F_τ*.*

Proof. The monotonicity follows immediately from the fact that our algorithm can be interpreted as a majorization-minimization scheme. For the stationarity, note that Equation (12) shows that $x_k = x_{k+1}$ if and only if $\nabla F_\tau(x_k) = 0$. □

4. Heat Regularization: Variational Characterization and Optimization

In this section, we consider direct regularization of the loss function f as opposed to regularization of the density function ρ_f. The following result is analogous to Theorem 1. Its proof is similar and hence omitted.

Theorem 3. *The heat regularization F_τ^H admits the following variational characterization:*

$$\begin{aligned} F_\tau^H(x) &:= \int_{\mathbb{R}^d} f(x')\varphi_{x,\tau}(x')\,dx' \\ &= \min_q \left\{ \log\left(\int_{\mathbb{R}^d} \exp(f(x'))q(x')\,dx' \right) + D_{\mathrm{KL}}(\varphi_{x,\tau}\|q) \right\}. \end{aligned} \tag{13}$$

Moreover, the density $q_{x,\tau}$ defined in Equation (6) achieves the minimum in (13).

4.1. Two-Step Iterative Optimization

From Equation (13), it follows that:

$$\arg\min_x F_\tau^H(x) = \arg\min_x \inf_q \left\{ \log\left(\int_{\mathbb{R}^d} \exp(f(x'))q(x')\,dx' \right) + D_{\mathrm{KL}}(\varphi_{x,\tau}\|q) \right\}. \tag{14}$$

In complete analogy with Section 3, Equation (14) suggests the following optimization scheme to minimize F_τ^H.

The key difference with Algorithm 1 is that the arguments of the Kullback–Leibler divergence are reversed. While $x \mapsto D_{\mathrm{KL}}(q_{x_k,\tau}\|\varphi_{x,\tau})$ has a unique minimizer given by $\mathbb{E}_{X\sim q_{x_k,\tau}}(X)$, minimizers of $x \mapsto D_{\mathrm{KL}}(\varphi_{x,\tau}\|q_{x_k,\tau})$ need not be unique [16]. We will provide an intuitive comparison between both minimization problems in Section 5, where we interpret the minimization $x \mapsto D_{\mathrm{KL}}(q_{x_k,\tau}\|\varphi_{x,\tau})$ as mean seeking and $x \mapsto D_{\mathrm{KL}}(\varphi_{x,\tau}\|q_{x_k,\tau})$ as mode seeking. In this light, the non-uniqueness of the latter minimization may arise, intuitively, when $q_{x_k,\tau}$ is multi-modal. Despite the potential lack of uniqueness of solutions, the minimization problem is well-posed, as shown in [16], and practical computational approaches for finding minima have been studied in [11]. In this computational regard, it is important to note that the minimization $x \mapsto D_{\mathrm{KL}}(\varphi_{x,\tau}\|q_{x_k,\tau})$ is implicitly defined via an expectation, and its computation via a Robbins–Monro [10] approach requires repeated evaluation of the gradient of f. We will outline the practical solution of this optimization problem in Section 5.2.

4.2. Majorization-Minorization and Monotonicity

As in Section 3.2, it is easy to see that:

$$A^H(x,\tilde{x}) := \log\left(\int_{\mathbb{R}^d} \exp(f(x'))q_{\tilde{x},\tau}(x')\,dx' \right) + D_{\mathrm{KL}}(\varphi_{x,\tau}\|q_{\tilde{x},\tau})$$

is a majorizer for F_τ^H. This can be used to show the following theorem, whose proof is identical to that of Theorem 2 and therefore omitted.

Theorem 4 (Monotonicity of Algorithm 2)**.** *The sequence $\{x_k\}_{k=0}^K$ generated by Algorithm 2 satisfies:*

$$F_\tau^H(x_k) \le F_\tau^H(x_{k-1}), \quad 1 \le k \le K.$$

Algorithm 2 Minimization of the heat regularization F_τ^H through optimization with respect to the first argument of the Kullback–Leibler divergence.

Choose $x_0 \in \mathbb{R}^d$, and for $k = 0, \ldots, K-1$, do:

1. Define $q_{x_k,\tau}$ as in Equation (6).
2. Define x_{k+1} by minimizing the map:

$$x \mapsto D_{\mathrm{KL}}(\varphi_{x,\tau}\|q_{x_k,\tau}).$$

5. Gaussian Kullback–Leibler Minimization

In Sections 3 and 4, we considered the local entropy F_τ and heat regularized loss F_τ^H and introduced two-step iterative optimization schemes for both loss functions. We summarize these schemes here for comparison purposes:

Optimization of F_τ
Let $x_0 \in \mathbb{R}^d$, and for $k = 0, \ldots, K-1$, do:

1. Define $q_{x_k,\tau}$ as in Equation (6).
2. Let x_{k+1} be the minimizer of:

$$x \mapsto D_{\mathrm{KL}}(q_{x_k,\tau} \| \varphi_{x,\tau}).$$

Optimization of F_τ^H
Let $x_0 \in \mathbb{R}^d$, and for $k = 0, \ldots, K-1$, do:

1. Define $q_{x_k,\tau}$ as in Equation (6).
2. Let x_{k+1} be a minimizer of:

$$x \mapsto D_{\mathrm{KL}}(\varphi_{x,\tau} \| q_{x_k,\tau}).$$

Both schemes involve finding, at each iteration, the mean vector that gives the best approximation, in Kullback–Leibler, to a probability density. For local entropy, the minimization is with respect to the second argument of the Kullback–Leibler divergence, while for heat regularization, the minimization is with respect to the first argument. It is useful to compare, in intuitive terms, the two different minimization problems, both leading to a "best Gaussian". In what follows, we drop the subscripts and use the following nomenclature:

$$D_{\mathrm{KL}}(q||\varphi) = \mathbb{E}^q\left[\log\left(\frac{q}{p}\right)\right] \qquad \text{"Mean seeking"}$$
$$D_{\mathrm{KL}}(\varphi||q) = \mathbb{E}^\varphi\left[\log\left(\frac{\varphi}{q}\right)\right] \qquad \text{"Mode seeking".}$$

Note that in order to minimize $D_{\mathrm{KL}}(\varphi\|q)$, we need $\log\frac{\varphi}{q}$ to be small over the support of φ, which can happen when $\varphi \simeq q$ or $\varphi \ll q$. This illustrates the fact that minimizing $D_{\mathrm{KL}}(\varphi\|q)$ may miss out components of q. For example, in Figure 3, left panel, q is a bi-modal-like distribution, but minimizing $D_{\mathrm{KL}}(\varphi||q)$ over Gaussians φ can only give a single mode approximation, which is achieved by matching one of the modes (minimizers are not guaranteed to be unique); we may think of this as "mode seeking". In contrast, when minimizing $D_{\mathrm{KL}}(q\|\varphi)$ over Gaussians φ, we want $\log\frac{q}{\varphi}$ to be small where φ appears as the denominator. This implies that wherever q has some mass, we must let φ also have some mass there in order to keep $\frac{q}{\varphi}$ as close as possible to one. Therefore, the minimization is carried out by allocating the mass of φ in a way such that on average, the discrepancy between φ and q is minimized, as shown in Figure 3, right panel; hence the label "mean seeking."

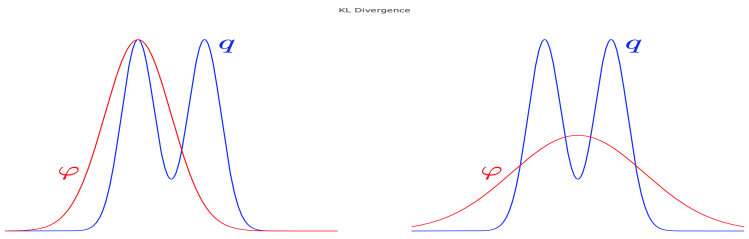

Figure 3. Cartoon representation of the mode seeking (**left**) and mean seeking (**right**) Kullback–Leibler minimization. Mean seeking minimization is employed within local-entropy optimization; mode seeking minimization is employed within optimization of the heat-regularized loss.

In the following two sections, we show that, in addition to giving rather different solutions, the argument of the Kullback–Leibler divergence that is minimized has computational consequences.

5.1. Minimization of $x \mapsto D_{\mathrm{KL}}(q_{x_k,\tau} \| \varphi_{x,\tau})$

The solution to this minimization problem is unique and given by $\mathbb{E}_{X\sim q_{x_k,\tau}}(X)$. For notational convenience, we drop the subscript k and consider the calculation of:

$$\mathbb{E}_{X\sim q_{x,\tau}}(X). \tag{15}$$

In our numerical experiments, we will approximate these expectations using stochastic gradient Langevin dynamics and importance sampling. Both methods are reviewed in the next two subsections.

5.1.1. Stochastic Gradient Langevin Dynamics

The first method that we use to approximate the expectation (15), and thus the best-Gaussian approximation for local entropy optimization, is stochastic gradient Langevin dynamics (SGLD). The algorithm, presented in Algorithm 3, was introduced in [17], and its use for local entropy minimization was investigated in [5]. The SGLD algorithm is summarized below.

Algorithm 3 Stochastic Gradient Langevin Dynamics (SGLD) algorithm for expectation approximation; the algorithm functions as a modification of a gradient-based Metropolis-Hastings Markov chain Monte Carlo algorithm.

Input: Sample size J and temperatures $\{\epsilon_j\}_{j=1}^J$.

1. Define $x^0 = x$.
2. For $j = 1, \ldots, J-1$, do:

$$x^{j+1} = x^j - \frac{\epsilon_j}{2}\left(\nabla f(x^j) - \frac{1}{\tau}(x - x^j)\right) + \eta_j, \qquad \eta_t \sim N(0, \epsilon_j).$$

Output: approximation $\mathbb{E}_{X\sim q_{x,\tau}}(X) \approx \frac{\sum_{j=1}^J \epsilon_j x^j}{\sum_{j=1}^J \epsilon_j}$.

When the function f is defined by a large sum over training data, minibatches can be used in the evaluation of the gradients $\nabla f(x^j)$. In our numerical experiments, we initialized the Langevin chain at the last iteration of the previous parameter update. Note that SGLD can be thought of as a modification of gradient-based Metropolis–Hastings Markov chain Monte Carlo algorithms, where the accept-reject mechanism is replaced by a suitable tempering of the temperatures ϵ_j.

5.1.2. Importance Sampling

We will also investigate the use of importance sampling [18], as displayed in Algorithm 4, to approximate the expectations (15); our main motivation in doing so is to avoid gradient computations, and hence to give an example of a training scheme that does not involve backpropagation.

Importance sampling is based on the observation that:

$$\mathbb{E}_{X\sim q_{x,\tau}}(X) = \int_{\mathbb{R}^d} x' q_{x,\tau}(x')\, dx' = \frac{\int_{\mathbb{R}^d} x' \exp\left(-f(x')\right) \varphi_{x,\tau}(x')\, dx'}{\int_{\mathbb{R}^d} \exp\left(-f(x')\right) \varphi_{x,\tau}(x')\, dx'},$$

and an approximation of the right-hand side may be obtained by standard Monte Carlo approximation of the numerator and the denominator. Crucially, these Monte Carlo simulations are performed sampling the Gaussian $\varphi_{x,\tau}$ rather than the original density q. The importance sampling algorithm is then given by:

Algorithm 4 Importance Sampling for estimation of the expectation in the second step of Algorithm 1. The algorithm does not require gradient evaluations, and is easily parallelizable by distributing sampling across multiple machines.

Input: sample size J.

1. Sample $\{x^j\}_{j=1}^J$ from the Gaussian density $\varphi_{x,\tau}$.
2. Compute (unnormalized) weights $w^j = \exp\left(-f(x^j)\right)$.

Output: approximation

$$\mathbb{E}_{X\sim q_{x,\tau}}(X) \approx \frac{\sum_{j=1}^J w^j x^j}{\sum_{j=1}^J w^j}. \tag{16}$$

Importance sampling is easily parallelizable. If L processors are available, then each of the processors can be used to produce an estimate using J/L Gaussian samples, and the associated estimates can be subsequently consolidated.

While the use of importance sampling opens an avenue for gradient-free, parallelizable training of neural networks, our numerical experiments will show that naive implementation without parallelization gives poor performance relative to SGLD or plain stochastic gradient descent (SGD) on the original loss. A potential explanation is the so-called curse of dimension for importance sampling [19,20]. Another explanation is that the iterative structure of SGLD allows re-utilizing the previous parameter update to approximate the following one, while importance sampling does not afford such iterative updating. Finally, SGLD with minibatches is known to asymptotically produce unbiased estimates, while the introduction of minibatches in importance sampling introduces a bias.

5.2. Minimization of $x \mapsto D_{\mathrm{KL}}(\varphi_{x,\tau} \| q_{x_k,\tau})$

A direct calculation shows that the preconditioned Euler–Lagrange equation for minimizing $x \mapsto D_{\mathrm{KL}}(\varphi_{x,\tau} \| q_{x_k,\tau})$ is given by:

$$h(x) := x - x_k + \tau \mathbb{E}_{Y\sim\varphi_{x,\tau}} \nabla f(Y) = 0.$$

Here, $h(x)$ is implicitly defined as an expected value with respect to a distribution that depends on the parameter x. The Robbins–Monro algorithm [10], displayed in Algorithm 5, allows estimating zeroes of functions defined in such a way.

Algorithm 5 The Robbins-Monro algorithm for estimating the zeros of the preconditioned Euler–Lagrange equation for minimizing the map $x \mapsto D_{\mathrm{KL}}(\varphi_{x,\tau} \| q_{x_k,\tau})$. The algorithm functions as a form of spatially-averaged gradient descent.

Input: Number of iterations J and schedule $\{a^j\}_{j=1}^J$.

1. Define $x^0 = x$.
2. For $j = 1, \ldots, J$, do:

$$x^{j+1} = x^j - a^j \left\{ x^j - x_k + \frac{\tau}{M} \sum_{m=1}^M \nabla f\left(z^{(m)}\right) \right\}, \qquad z^{(m)} \sim \varphi_{x^j,\tau}. \tag{17}$$

Output: approximation x^J to the minimizer of $x \mapsto D_{\mathrm{KL}}(\varphi_{x,\tau} \| q_{x_k,\tau})$.

The Robbins–Monro approach to computing the Gaussian approximation $(x,\tau) \mapsto D_{\mathrm{KL}}(\varphi_{x,\tau} \| q_{x_k,\tau})$ in Hilbert space was studied in [11]. A suitable choice for the step size is $a^l = cl^\alpha$, for some $c > 0$ and $\alpha \in (1/2, 1]$. Note that Algorithm 5 gives a form of spatially-averaged gradient descent, which involves

repeated evaluation of the gradient of the original loss. The use of temporal gradient averages has also been studied as a way to reduce the noise level of stochastic gradient methods [1].

To conclude, we remark that an alternative approach could be to employ Robbins–Monro directly to optimize $F^H(x)$. Gradient calculations would still be needed.

6. Numerical Experiments

In the following numerical experiments, we investigated the practical use of local entropy and heat regularization in the training of neural networks. We present experiments on dense multilayered networks applied to a basic image classification task, viz. MNIST handwritten digit classification [21]. Our choice of dataset is standard in machine learning and had been considered in previous work on loss regularizations for deep learning, e.g., [4,5]. We implemented Algorithms 3, 4, and 5 in TensorFlow, analyzing the effectiveness of each in comparison to stochastic gradient descent (SGD). We investigated whether the theoretical monotonicity of regularized losses translates into monotonicity of the held-out test data error. Additionally, we explored various choices for the hyperparameter τ to illustrate the effects of variable levels of regularization. In accordance with the algorithms specified above, we employed importance sampling (IS) and stochastic gradient Langevin dynamics (SGLD) to approximate the expectation in (15) and the Robbins–Monro algorithm for heat regularization (HR).

6.1. Network Specification

Our experiments were carried out using the following networks:

1. Small dense network: Consisting of an input layer with 784 units and a 10-unit output layer, this toy network contained 7850 total parameters and achieved a test accuracy of 91.2% when trained with SGD for five epochs over the 60,000-image MNIST dataset.
2. Single hidden layer dense network: Using the same input and output layer as the smaller network with an additional 200-unit hidden layer, this network provides an architecture with 159,010 parameters. We expect this architecture to achieve a best-case performance of 98.9% accuracy on MNIST, trained over the same data as the previous network.

We remark that our choices of network architecture were not intended to provide state-of-the-art accuracy in the classification for MNIST, but rather to illustrate the relative merits of the regularizations and optimization methods considered in this paper.

6.2. Training Neural Networks from Random Initialization

Considering the computational burden of computing a Monte Carlo estimate for each weight update, we proposed that Algorithms 3, 4, and 5 are potentially most useful when employed following SGD; although per-update progress is on par or exceeds that of SGD with step size, often called learning rate, equivalent to the value of τ, the computational load required makes the method unsuited for end-to-end training. Though in this section, we present an analysis of these algorithms used for the entirety of training, this approach is likely too expensive to be practical for contemporary deep networks.

Table 1 and the associated Figure 4 demonstrate the comparative training behavior for each algorithm, displaying the held-out test accuracy for identical instantiations of the hidden layer network trained with each algorithm for 500 parameter updates. Note that a minibatch size of 20 was used in each case to standardize the amount of training data available to the methods. Additionally, SGLD, IS, and HR each employed $\tau = 0.01$, while SGD utilized an equivalent step size, thus fixing the level of regularization in training. To establish computational equivalence between Algorithms 3, 4, and 5, we computed $\mathbb{E}_{X\sim q}(X)$ with 10^3 samples for Algorithms 3 and 4, setting $M = 30$ and performing 30 updates of the chain in Algorithm 5. Testing accuracy was computed by classifying 1000 randomly-selected images from the held-out MNIST test set. In related experiments, we observed consistent training progress across all three algorithms. In contrast, IS and HR trained more slowly,

particularly during the parameter updates following initialization. From Figure 4, we can appreciate that while SGD attempted to minimize training error, it nonetheless behaved in a stable way when plotting held-out accuracy, especially towards the end of training. SGLD on the other hand was observed to be more stable throughout the whole training, with few drops in accuracy along the sequence of optimization iterates.

Table 1. Classification accuracy on held-out test data. SGLD, stochastic gradient Langevin dynamics; IS, importance sampling; HR, heat regularization.

Weight Updates	100	200	300	400	500
SGD	0.75	0.80	0.85	0.87	0.87
IS	0.27	0.45	0.54	0.57	0.65
SGLD	0.72	0.81	0.84	0.86	0.88
HR	0.52	0.64	0.70	0.73	0.76

Figure 4. Held-out test accuracy during training for SGD (red), SGLD (blue), HR (green), and IS (black). $\tau = 0.01$ for SGLD, IS, and HR. The learning rate of SGD is also 0.01. SGLD uses temperatures $\epsilon_j = \frac{1}{1000+j}$, and HR's update schedule uses $c = 0.1$, and $\alpha = 0.7$.

While SGD, SGLD, and HR utilize gradient information in performing parameter updates, IS does not. This difference in approach contributes to IS's comparatively poor start; as the other methods advance quickly due to the large gradient of the loss landscape, IS's progress was isolated, leading to training that depended only on the choice of τ. When τ was held constant, as shown in Figure 4, the rate of improvement remained nearly constant throughout. This suggests the need for dynamically updating τ, as is commonly performed with annealed learning rates for SGD. Moreover, SGD, SGLD, and HR are all schemes that depend linearly on f, making minibatching justifiable, something, which is not true for IS.

It is worth noting that the time to train differed drastically between methods. Table 2 shows the average runtime of each algorithm in seconds. SGD performed roughly 10^3-times faster than the others, an expected result considering the most costly operation in training, filling the network weights, was performed 10^3-times per parameter update. Other factors contributing to the runtime discrepancy are the implementation specifications and the deep learning library; here, we used TensorFlow's implementation of SGD, a method for which the framework is optimized. More generally, the runtimes

in Table 2 reflected the hyperparameter choices for the number of Monte Carlo samples and will vary according to the number of samples considered.

Table 2. Runtime per weight update.

	Average Update Runtime (Seconds)
SGD	0.0032
IS	6.2504
SGLD	7.0599
HR	3.3053

6.3. Local Entropy Regularization after SGD

Considering the longer runtime of the sampling-based algorithms in comparison to SGD, it is appealing to utilize SGD to train networks initially, then shift to more computationally-intensive methods to identify local minima with favorable generalization properties. Figure 5 illustrates IS and SGLD performing better than HR when applied after SGD. HR's smooths the loss landscape, a transformation that is advantageous for generating large steps early in training, but presents challenges as smaller features are lost. In Figure 5, this effect manifests as constant test accuracy after SGD, and no additional progress is made. The contrast between each method is notable since the algorithms used equivalent step sizes; this suggests that the methods, not the hyperparameter choices, dictate the behavior observed.

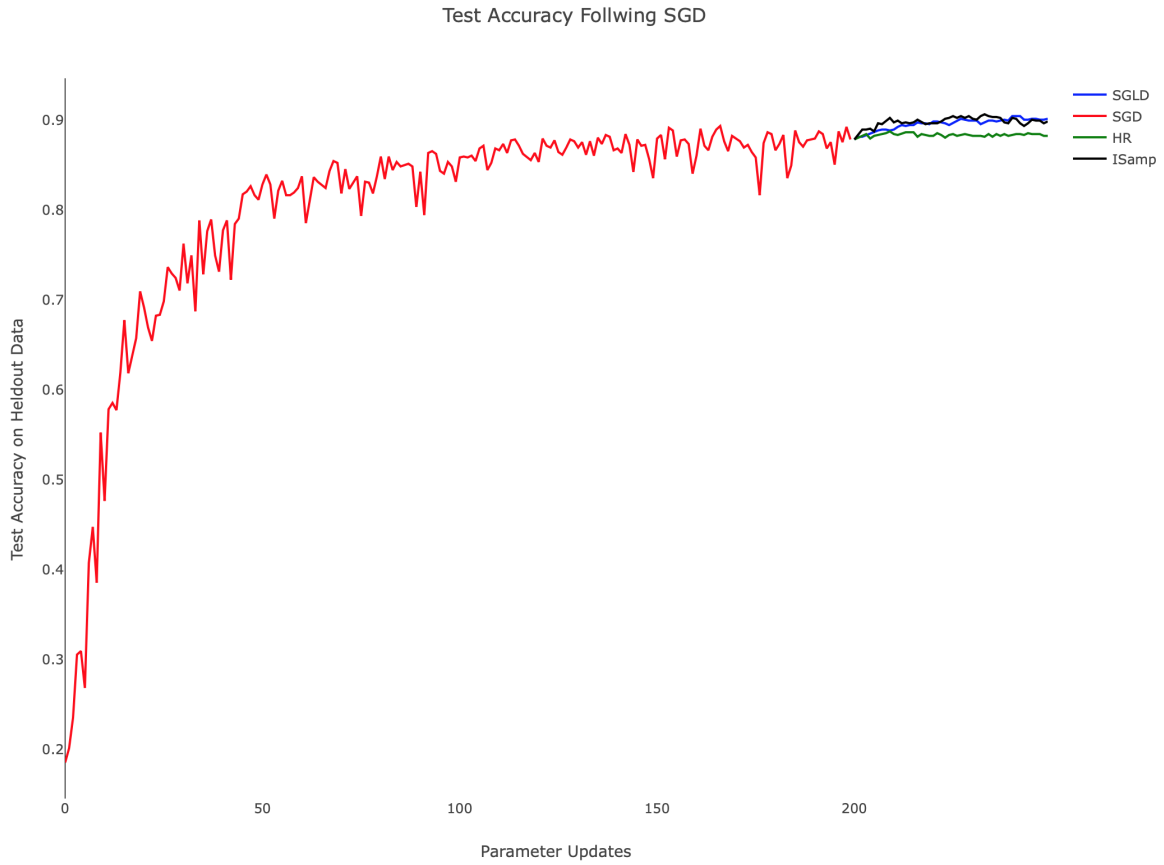

Figure 5. Training after SGD, $\tau = 0.01$ for all algorithms. The step size for the SGD is set equal to the value of τ for all three algorithms. SGLD temperatures are $\epsilon_j = \frac{1}{2000+j}$, and HR uses the same update schedule as in Figure 4.

Presumably, SGD trains the network into a sharp local minima or saddle point of the non-regularized loss landscape; transitioning to an algorithm that minimizes the local entropy regularized loss, then finds an extremum, which performs better on the test data. However, based on our experiments, in terms of held-out data accuracy, regularization in the later stages does not seem to provide significant improvement over training with SGD on the original loss.

6.4. Algorithm Stability and Monotonicity

Prompted by the guarantees of Theorems 2 and 4, which prove the effectiveness of these methods when $\mathbb{E}_{X\sim q}(X)$ is approximated accurately, we also demonstrated the stability of these algorithms in the case of an inaccurate estimate of the expectation. To do so, we explored the empirical consequences of varying the number of samples used in the Monte Carlo and Robbins–Monro calculations.

Figure 6 shows how each algorithm responds to this change. We observe that IS performed better as we refined our estimate of $\mathbb{E}_{X\sim q}(X)$, exhibiting less noise and faster training rates. This finding suggests that a highly parallel implementation of IS, which leverages the modern GPU architecture to efficiently compute the relevant expectation, may offer practicality. SGLD also benefits from a more accurate approximation, displaying faster convergence and higher final testing accuracy when comparing 10 and 100 Monte Carlo samples. HR however performs more poorly when we employ longer Robbins–Monro chains, suffering from diminished step size and exchanging quickly realized progress for less oscillatory testing accuracy. Exploration of the choices of ϵ_j and a^j for SGLD and HR remains a valuable avenue for future research, specifically in regards to the interplay between these hyperparameters and the variable accuracy of estimating $\mathbb{E}_{X\sim q}(X)$.

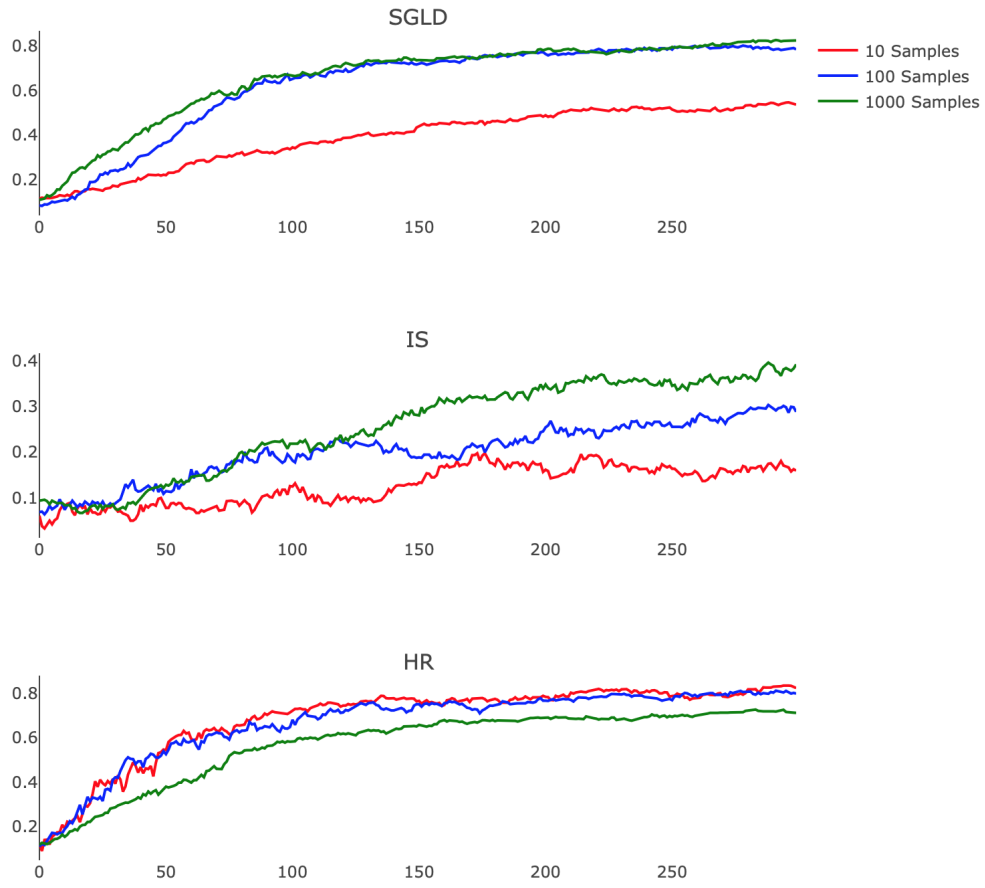

Figure 6. Training behaviors with $S_i = \{10$ (Red), 100 (Blue), 1000 (Green)$\}$ samples per parameter update. SGLD temperatures and HR schedule are the same as in Figure 4. Note that $\tau = 0.01$ throughout. To equalize computational load across algorithms, we set $M = J = \lfloor \sqrt{S_i} \rfloor$ for HR.

6.5. Choosing τ

An additional consideration of these schemes is the choice of τ, the hyperparameter that dictates the level of regularization in Algorithms 3, 4, and 5. As noted in [5], large values of τ correspond to a nearly uniform local entropy regularized loss, whereas small values of τ yield a minimally regularized loss, which is very similar to the original loss function. To explore the effects of small and large values of τ, we trained our smaller network with IS and SGLD for many choices of τ, observing how regularization alters training rates.

The results, presented in Figure 7, illustrate differences in SGLD and IS, particularly in the small τ regime. As evidenced in the leftmost plots, SGLD trained successfully, albeit slowly, with $\tau \in [0.001, 0.01]$. For small values of τ, the held-out test accuracy improved almost linearly over parameter updates, appearing characteristically similar to SGD with a small learning rate. IS failed for small τ, with highly variant test accuracy improving only slightly during training. Increasing τ, we observed SGLD reach a point of saturation, as additional increases in τ did not affect the training trajectory. We note that this behavior persisted as $\tau \to \infty$, recognizing that the regularization term in the SGLD algorithm approached a value of zero for growing τ. IS demonstrated improved training efficiency in the bottom-center panel, showing that increased τ provided favorable algorithmic improvements. This trend dissipated for larger τ, with IS performing poorly as $\tau \to \infty$. The observed behavior suggests there exists an optimal τ that is architecture and task specific, opening opportunities to further develop a heuristic to tune the hyperparameter.

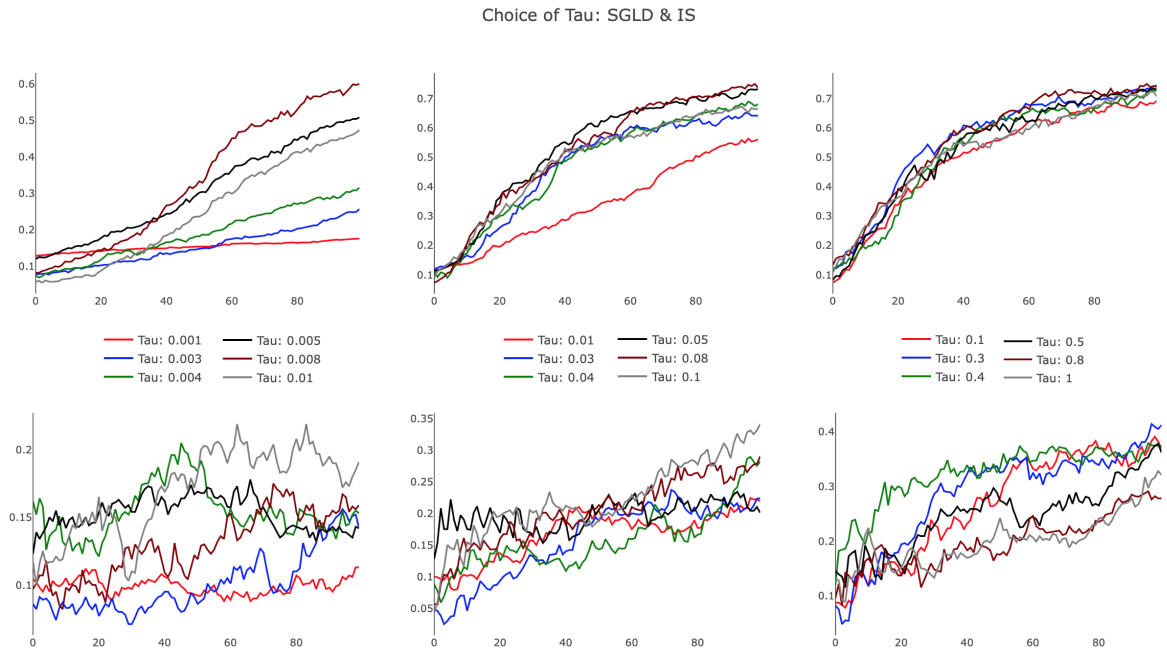

Figure 7. Training the smaller neural network with different choices for τ using SGLD and IS. Values of τ vary horizontally from very small to large: $\tau \in \{[0.001, 0.01], [0.01, 0.1], [0.1, 1]\}$. Top row shows SGLD with $\epsilon_j = \frac{1}{1000+j}$, and the bottom row shows IS. All network parameters were initialized randomly.

As suggested in [5], we investigated annealing the scope of τ from large to small values in order to examine the landscape of the loss function at different scales. Early in training, we used comparatively large values to ensure broad exploration, transitioning to smaller values for a comprehensive survey of the landscape surrounding a minima. We used the following schedule for the kth parameter update:

$$\tau(k) = \frac{\tau_0}{(1+\tau_1)^k} \tag{18}$$

where τ_0 is large and τ_1 is set so that the magnitude of the local entropy gradient is roughly equivalent to that of SGD.

As shown in Figure 8, annealing τ proved to be useful and provided a method by which training can focus on more localized features to improve test accuracy. We observed that SGLD, with a smaller value of $\tau = 0.01$, achieved a final test accuracy close to that of SGD, whereas $\tau = 1.5$ was unable to identify the optimal minima. Additionally, the plot shows that large τ SGLD trained faster than SGD in the initial 100 parameter updates, whereas small τ SGLD lagged behind. When scoping τ, we considered both annealing and reverse-annealing, illustrating that increasing τ over training produced a network that trained more slowly than SGD and was unable to achieve testing accuracy comparable to that of SGD. Scoping τ from $1.5 \to 0.01$ via the schedule (18) with $\tau_0 = 1.5$ and $\tau_1 = 0.01$ delivered advantageous results, yielding an algorithm that trains faster than SGD after initialization and achieved analogous testing accuracy. We remark that, while the preceding figures offer some insight into the various behaviors associated with difference choices of τ, a considerable amount of detail regarding proper dynamic tuning of the hyperparameter remains unknown, specifying an additional open research question. At the current stage, our recommendation for choosing τ would be to use cross-validation over several scoping schedules.

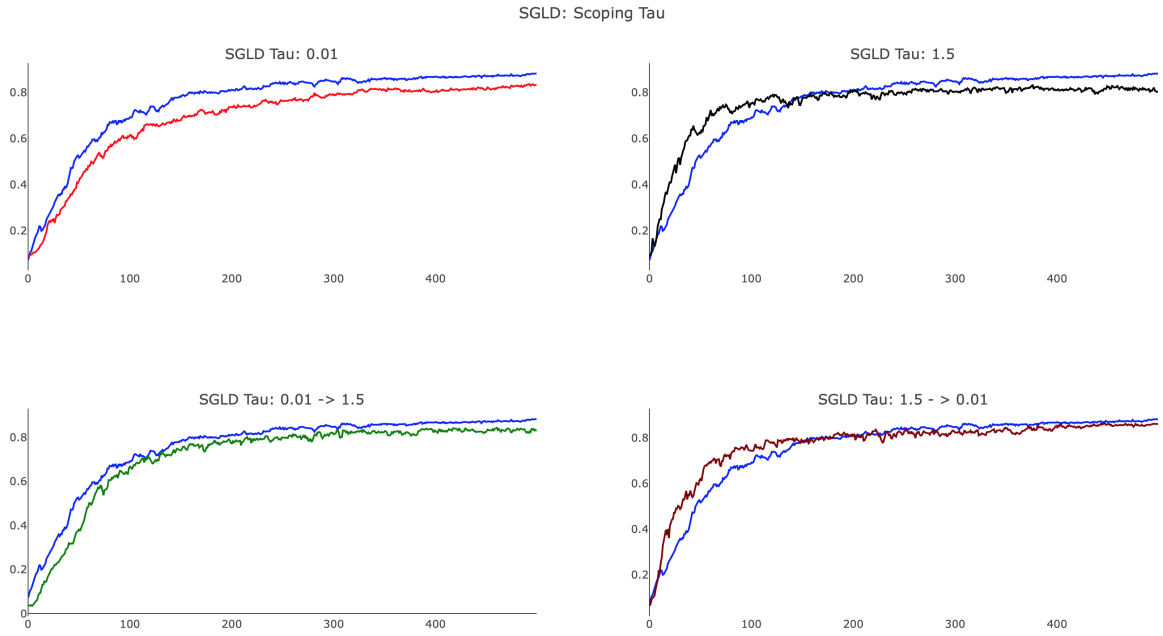

Figure 8. Examination of the effects of scoping τ during training via the update schedule (18). All four panels display SGLD with temperatures set as $\epsilon_j = \frac{1}{1000+j}$ and SGD (blue) with a learning rate of 0.01. Top: SGLD with constant τ, set as $\tau = 0.01$ and $\tau = 1.5$. Bottom: τ scoped as $\tau : 0.01 \to 1.5$ and $\tau : 1.5 \to 0.001$.

7. Conclusions

We conclude by listing some outcomes of our work:

- Information theory in deep learning: We have introduced information theoretic, variational characterizations of two loss regularizations that have received recent attention in the deep learning community. These characterizations provide a unified framework under which optimization of heat and local entropy regularized costs corresponds to finding best Gaussian approximations in Kullback–Leibler divergence with respect to its first and second argument.

- Loss regularization: We have provided a new theory that explains the gain in stability of generalization facilitated by local regularization. Our presentation provides a transparent account of the potential increase in computational cost of naive optimization of regularized costs and in that sense gives a less optimistic view than existing works of the gains facilitated by loss regularization.
- Local entropy vs. heat regularization: While our theoretical results show that the stable training of local entropy regularized networks may also be present in heat regularized costs, our numerical experiments show, in agreement with the theory provided in [4], that local entropy methods have a better empirical performance. Moreover, we have emphasized that optimization in local entropy regularized costs (as opposed to heat regularized costs) may be naturally performed by a variety of sampling methods, thus opening an avenue for further research on gradient-free training for neural networks. In this sense, and contrary to the previous bullet, our work introduces a new reason for optimism in pursuing the study of loss regularizations in deep learning.
- Extensions: It would be possible to generalize the Gaussian approximations within our two-step iterative schemes and allow updating both the mean and covariance of the Gaussian measures. More broadly, it may be interesting to generalize the methods in this paper replacing the Kullback–Leibler divergence by a more general family of divergences or considering best approximations over more general families of probability distributions.

Author Contributions: All authors contributed equally to this work.

Funding: The work of NGT and DSA was supported by the NSF Grant DMS-1912818/1912802.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Bottou, L.; Curtis, F.E.; Nocedal, J. Optimization methods for large-scale machine learning. *arXiv* **2016**, arXiv:1606.04838.
2. Keskar, N.S.; Mudigere, D.; Nocedal, J.; Smelyanskiy, M.; Tang, P.T.P. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv* **2016**, arXiv:1609.04836.
3. Hochreiter, S.; Schmidhuber, J. Flat minima. *Neural Comput.* **1997**, *9*, 1–42. [CrossRef] [PubMed]
4. Chaudhari, P.; Oberman, A.; Osher, S.; Soatto, S.; Carlier, G. Deep relaxation: partial differential equations for optimizing deep neural networks. *arXiv* **2017**, arXiv:1704.04932.
5. Chaudhari, P.; Choromanska, A.; Soatto, S.; LeCun, Y.; Baldassi, C.; Borgs, C.; Chayes, J.T.; Sagun, L.; Zecchina, R. Entropy-SGD: Biasing gradient descent into wide valleys. *arXiv* **2017**, arXiv:1611.01838,
6. Dinh, L.; Pascanu, R.; Bengio, S.; Bengio, Y. Sharp minima can generalize for deep nets. *arXiv* **2017**, arXiv:1703.04933.
7. Neyshabur, B.; Bhojanapalli, S.; McAllester, D.; Srebro, N. Exploring generalization in deep learning. In Proceedings of the NIPS 2017, Long Beach, CA, USA, 4–9 December 2017; pp. 5947–5956.
8. Baldassi, C.; Ingrosso, A.; Lucibello, C.; Saglietti, L.; Zecchina, R. Subdominant dense clusters allow for simple learning and high computational performance in neural networks with discrete synapses. *Phys. Rev. Lett.* **2015**, *115*, 128101. [CrossRef] [PubMed]
9. Dupuis, P.; Ellis, R.S. *A Weak Convergence Approach to the Theory of Large Deviations*; John Wiley & Sons: Hoboken, NJ, USA, 2011; Volume 902.
10. Robbins, H. *An Empirical Bayes Approach to Statistics*; Technical Report; Columbia University: New York, NY, USA , 1956.
11. Pinski, F.J.; Simpson, G.; Stuart, A.M.; Weber, H. Algorithms for Kullback–Leibler approximation of probability measures in infinite dimensions. *SIAM J. Sci. Comput.* **2015**, *37*, A2733–A2757. [CrossRef]
12. Wainwright, M.J.; Jordan, M.I. Graphical Models, Exponential Families, and Variational Inference. *Found. Trends Mach. Learn.* **2008**, *1*, 1–305. [CrossRef]
13. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436. [CrossRef] [PubMed]
14. Goodfellow, I.; Bengio, Y.; Courville, A.; Bengio, Y. *Deep Learning*; MIT Press Cambridge: Cambridge, MA, USA, 2016; Volume 1.

15. Garcia Trillos, N.; Sanz-Alonso, D. The Bayesian Update: Variational Formulations and Gradient Flows. *Bayesian Anal.* **2018**. [CrossRef]
16. Pinski, F.J.; Simpson, G.; Stuart, A.M.; Weber, H. Kullback–Leibler approximation for probability measures on infinite dimensional spaces. *SIAM J. Math. Anal.* **2015**, *47*, 4091–4122. [CrossRef]
17. Welling, M.; Teh, Y.W. Bayesian learning via stochastic gradient Langevin dynamics. In Proceedings of the 28th International Conference on Machine Learning (ICML-11), Bellevue, WA, USA, 28 June 2011; pp. 681–688.
18. Liu, J.S. *Monte Carlo Strategies in Scientific Computing*; Springer Science & Business Media: Berlin, Germany, 2008.
19. Sanz-Alonso, D. Importance sampling and necessary sample size: An information theory approach. *SIAM/ASA J. Uncertain.* **2018**, *6*, 867–879. [CrossRef]
20. Agapiou, S.; Papaspiliopoulos, O.; Sanz-Alonso, D.; Stuart, A.M. Importance sampling: Intrinsic dimension and computational cost. *Stat. Sci.* **2017**, *32*, 405–431. [CrossRef]
21. Lecun, Y. The MNIST Database of Handwritten Digits. Available online: http://yann.lecun.com/exdb/mnist/ (accessed on 12 May 2019).