One Size Fits All: Can We Train One Denoiser for All Noise Levels?

Abhiram Gnansambandam [1] **Stanley H. Chan** [1 2]

Abstract

When training an estimator such as a neural network for tasks like image denoising, it is often preferred to train *one* estimator and apply it to *all* noise levels. The de facto training protocol to achieve this goal is to train the estimator with noisy samples whose noise levels are uniformly distributed across the range of interest. However, why should we allocate the samples uniformly? Can we have more training samples that are less noisy, and fewer samples that are more noisy? What is the optimal distribution? How do we obtain such a distribution? The goal of this paper is to address this training sample distribution problem from a minimax risk optimization perspective. We derive a dual ascent algorithm to determine the optimal sampling distribution of which the convergence is guaranteed as long as the set of admissible estimators is closed and convex. For estimators with non-convex admissible sets such as deep neural networks, our dual formulation converges to a solution of the convex relaxation. We discuss how the algorithm can be implemented in practice. We evaluate the algorithm on linear estimators and deep networks.

1. Introduction

1.1. "One Size Fits All" Denoisers

The following phenomenon could be familiar to those who develop learning-based image denoisers. If the denoiser is trained at a noise level σ, then its performance is maximized when the testing noise level is also σ. As soon as the testing noise level deviates from the training noise level, the performance drops (Choi et al., 2019; Kim et al., 2019). This is a typical mismatch between training and testing, which is arguably universal for all learning-based estimators. When such a problem arises, the most straight-forward solution is to create a suite of denoisers trained at different noise levels and use the one that matches best with the input noisy image (such as those used in the "Plug-and-Play" priors (Zhang et al., 2017; Chan et al., 2016; Chan, 2019)). However, this ensemble approach is not effective since the model capacity is multiple times larger than necessary.

A more widely adopted solution is to train *one* denoiser and use it for *all* noise levels. The idea is to train the denoiser using a training dataset containing images of different noise levels. The competitiveness of these "one size fits all" denoisers compared to the best individually trained denoisers has been demonstrated in (Zhang et al., 2017; 2018; Mao et al., 2016a; Remez et al., 2017). However, as we will illustrate in this paper, there is no guarantee for such arbitrarily trained one-size-fits-all denoiser to have a consistent performance over the entire noise range. At some noise levels, usually at the lower tail of the noise range, the performance could be much worse than the best individuals. The cause of this phenomenon is related to how we draw the noisy samples, which is usually *uniform* across the noise range. The question we ask here is that if we allocate more low-noise samples and fewer high-noise samples, will we be able to get a more consistent result?

1.2. Objective and Contributions

The objective of this paper is to find a sampling distribution such that for every noise level the performance is consistent. Here, by consistent we meant that the gap between the estimator and the best individuals is balanced. The idea is illustrated in Figure 1. The black curve in the figure represents the ensemble of the best individually trained denoisers. It is a virtual curve obtained by training the denoiser at each noise level. A typical "one size fits all" denoiser is trained by using noisy samples from a uniform distribution, which is denoted by the blue curve. This figure illustrates a typical in-consistence where there is a significant gap at low-noise but small gap at high noise. The objective of the paper is to find a new sampling distribution (denoted by the orange bars) such that we can achieve a consistent performance throughout the entire range. The result returned by our method is a trade-off between the overall performance and the worst cases scenarios.

[1]School of Electrical and Computer Engineering, [2]Department of Statistics, Purdue University, West Lafayette, IN 47906, United States. Correspondence to: Stanley Chan <stanchan@purdue.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020.

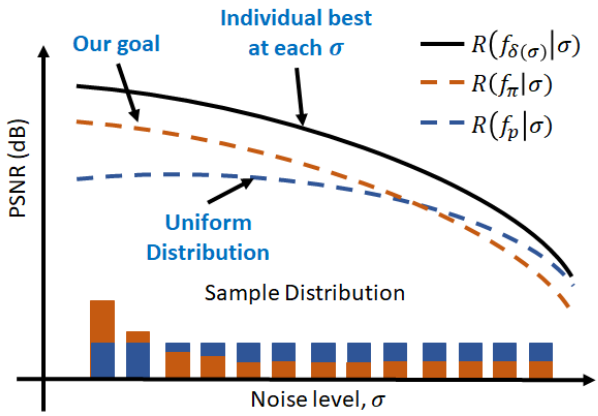


Figure 1. Illustration of the objective of this paper. The typical uniform sampling (blue bars) will yield a performance curve that is skewed towards one side of the noise range. The objective of this paper is to find an optimal sampling distribution (orange bars) such that the performance is consistent across the noise range. Notations will be defined in Section 3. We plot the risks in terms of the peak signal-to-noise ratio.

The key idea behind the proposed method is a minimax formulation. This minimax optimization minimizes the overall risk of the estimator subject to the constraint that the worst case performance is bounded. We show that under the standard convexity assumptions on the set of all admissible estimators, we can derive a provably convergent algorithm by analyzing the dual. For estimators whose admissible set is not convex, solutions returned by our dual algorithm are the convex-relaxation results. We present the algorithm, and we show that steps of the algorithm can be implemented by iteratively updating the sample distributions.

2. Related Work

While the above sampling distribution problem may sound familiar, its solution does not seem to be available in the computer vision and machine learning literature.

Image Denoising. Recent work in image denoising has been focusing on developing better neural network architectures. When encountering multiple noise levels, (Zhang et al., 2017) presented two approaches: Create a suite of denoisers at different noise levels, or train a denoiser by uniformly sampling noise levels from the range. For the former approach, (Choi et al., 2019) proposed to combine the estimators by solving a convex optimization problem. (Gharbi et al., 2016) proposed an alternative approach by introducing a noise map as an extra channel to the network. Our paper shares the same overall goal as (Kim et al., 2019). However, they address problem by modifying the network structure whereas we do not change the network. Another related work is (Gao & Grauman, 2017) which proposed an ad-hoc solution to the sample distribution. Our paper offers theoretical justification, convergence guarantee, and optimality.

Active Learning / Experimental Design. Adjusting the distribution of the training samples during the learning procedure is broadly referred to active learning in machine learning (Settles, 2009) or experimental design in statistics (Chaloner & Verdinelli, 1995). Active learning / experimental design are typically associated with limited training data (Gal et al., 2017; Sener & Savarese, 2018). The goal is to optimally select the next data point (or batch of data points) so that we can estimate the model parameters, e.g., the mean and variance. The problem we encounter here is not about limited data because we can synthesize as much data as we want since we know the image formation process. The challenge is how to allocate the synthesized data.

Constrained Optimization in Neural Network. Training neural networks under constraints have been considered in classic optimization literature (Platt & Barr, 1987) (Zak et al., 1995). More recently, there are optimization methods for solving inequality constrained problems in neural networks (Pathak et al., 2015), and equality constrained problems (Márquez-Neila et al., 2017). However, these methods are generic approaches. The convexity of our problem allows us to develop a unique and simple algorithm.

Fairness Aware Classification. The task of seeking "balanced samples" can be considered as improving the fairness of the estimator. Literature on fairness aware classification is rapidly growing. These methods include modifying the network structure, the data distribution, and loss functions (Zafar et al., 2015; Pedreshi et al., 2008; Calders & Verwer, 2010; Hardt et al., 2016; Kamishima et al., 2012). Putting the fairness as a constrained optimization has been proposed by (Zafar et al., 2017), but their problem objective and solution are different from ours.

3. Problem Formulation

3.1. Training and Testing Distributions: $\pi(\sigma)$ and $p(\sigma)$

Consider a clean signal $\boldsymbol{y} \in \mathbb{R}^n$. We assume that this clean signal is corrupted by some random process to produce a corrupted signal $\boldsymbol{x}_\sigma \in \mathbb{R}^n$. The parameter σ can be treated in a broad sense as the level of uncertainty. The support of σ is denoted by the set Ω. We assume that σ is a random variable with a probability density function $p(\sigma)$.

Examples. In a denoising problem, the image formation model is given by $\boldsymbol{x}_\sigma = \boldsymbol{y} + \sigma\boldsymbol{\eta}$ where $\boldsymbol{\eta}$ is a zero-mean unit-variance i.i.d. Gaussian noise vector. The noise level is measured by σ. For image deblurring, the model becomes $\boldsymbol{x}_\sigma = \boldsymbol{h}_\sigma * \boldsymbol{y} + \boldsymbol{\epsilon}$ where $\boldsymbol{h}_\sigma$ denotes the blur kernel with radius σ, "$*$" denotes convolution, and $\boldsymbol{\epsilon}$ is the noise. In this case, the uncertainty is associated with the blur radius. □

We focus on *learning-based* estimators. We define an estimator $f : \mathbb{R}^n \to \mathbb{R}^n$ as a mapping that takes a noisy input $\boldsymbol{x}_\sigma$ and maps it to a denoised output $f(\boldsymbol{x}_\sigma)$. We assume that f is parametrized by $\theta \in \Theta$, but for notation simplicity we omit the parameter θ when the context is clear. The set of all admissible f's is denoted as $\mathcal{F} = \{f(\cdot, \theta) \,|\, \theta \in \Theta\}$.

To train the estimator f, we draw training samples from the set $\mathcal{S} \overset{\text{def}}{=} \{\boldsymbol{x}_\sigma^{(\ell)} \mid \ell = 1, \ldots, N,\ \sigma \overset{\text{i.i.d.}}{\sim} \pi(\sigma)\}$, where ℓ refers to the ℓ-th training sample, and $\pi(\sigma)$ is the distribution of the noise levels in the training samples. Note that π is not necessarily the same as p. The distribution π is the distribution of the *training* samples, and the distribution p is the distribution of the *testing* samples. In most learning scenarios, we want π to match with p so that the generalization error is minimized. However, in this paper, we are purposely designing a π which is different from p because the goal is to seek an optimal trade-off. To emphasize the dependency of f on π, we denote f as f_π.

3.2. Risk and Conditional Risk: $R(f)$ and $R(f|\sigma)$

Training an estimator f_π requires a loss function. We denote the loss between a predicted signal $f_\pi(\boldsymbol{x}_\sigma)$ and the truth $\boldsymbol{y}$ as $\mathcal{L}(f_\pi(\boldsymbol{x}_\sigma), \boldsymbol{y})$. An example of the loss function is the Euclidean distance:

$$\mathcal{L}(f_\pi(\boldsymbol{x}), \boldsymbol{y}) = \|f_\pi(\boldsymbol{x}_\sigma) - \boldsymbol{y}\|^2. \tag{1}$$

Other types of loss functions can also be used as long as they are convex in f_π.

To quantify the performance of the estimator f_π, we define the notion of *conditional risk*:

$$R(f_\pi \mid \sigma) \overset{\text{def}}{=} \mathbb{E}_{(\boldsymbol{x}_\sigma, \boldsymbol{y})|\sigma}\Big[\mathcal{L}(f_\pi(\boldsymbol{x}_\sigma), \boldsymbol{y}) \mid \sigma\Big]. \tag{2}$$

The conditional risk can be interpreted as the risk of the estimator f_π evaluated at a particular noise level σ. The overall risk is defined through iterated expectation:

$$\begin{aligned} R(f_\pi) &\overset{\text{def}}{=} \mathbb{E}_{\sigma\sim p(\sigma)}\big\{R(f_\pi \mid \sigma)\big\} \\ &= \int \underbrace{\mathbb{E}_{(\boldsymbol{x}_\sigma, \boldsymbol{y})|\sigma}[\mathcal{L}(f_\pi(\boldsymbol{x}_\sigma), \boldsymbol{y}) \mid \sigma]}_{=R(f_\pi \mid \sigma)}\ p(\sigma) d\sigma. \end{aligned} \tag{3}$$

Note that the expectation of σ is taken with respect to the true distribution p since we are evaluating the estimator f_π.

3.3. Three Estimators: f_π, f_p and $f_{\delta(\sigma)}$

The estimator f_π is determined by minimizing the training loss. In our problem, since the training set follows a distribution $\pi(\sigma)$, it holds that f_π is determined by

$$f_\pi \overset{\text{def}}{=} \operatorname*{argmin}_f \int R(f \,|\, \sigma)\pi(\sigma)\, d\sigma. \tag{4}$$

This definition can be understood by noting that $R(f \mid \sigma)$ is the conditional risk evaluated at σ. Since $\pi(\sigma)$ specifies the probability of obtaining a noisy samples with noise level σ, the integration in (4) defines the training loss when the noisy samples are proportional to $\pi(\sigma)$. Therefore, by minimizing this training loss, we will obtain f_π.

Example. Suppose that we are training a denoiser over the range of $\sigma \in [a, b]$. If the training set contains samples whose noise levels are uniformly distributed, i.e., $\pi(\sigma) = 1/(b-a)$ for $\sigma \in [a, b]$ and is 0 otherwise, then f_π is obtained by minimizing the sum of the individual losses $f_\pi = \operatorname{argmin}_f \sum_{\ell=1}^N \mathcal{L}(f(\boldsymbol{x}_\sigma^{(\ell)}), \boldsymbol{y}^{(\ell)})$ where the N training samples have noise levels equally likely coming from the range of $[a, b]$. □

If we replace the training distribution π by the testing distribution p, then we obtain the following estimator:

$$f_p = \operatorname*{argmin}_f \int R(f \mid \sigma)p(\sigma)\, d\sigma = \operatorname*{argmin}_f \ R(f). \tag{5}$$

Since f_p minimizes the overall risk, we expect $R(f_p) \leq R(f_\pi)$ for all π. This is summarized in the lemma below.

Lemma 1. *The risk of f_p is a lower bound of the risk of all other f_π:*

$$R(f_p) \leq R(f_\pi), \qquad \forall \pi. \tag{6}$$

Proof. By construction, f_p is the minimizer of the risk according to (5), it holds that $R(f_p) = \inf_f R(f)$. Therefore, for any π we have $R(f_p) \leq R(f_\pi)$. □

The consequence of Lemma 1 is that if we minimize $R(f)$ without any constraint, we will reach a trivial solution of $\pi = p$. This explains why this paper is uninteresting if the goal is to purely minimize the generalization error without considering any constraint.

Before we proceed, let us define one more distribution δ which has a point mass at a particular σ, i.e., $p(\sigma)$ is a delta function such that $p(\sigma') = \delta(\sigma' - \sigma)$. This estimator is found by simply minimizing the training loss

$$f_{\delta(\sigma)} = \operatorname*{argmin}_f \int R(f \mid \sigma')\delta(\sigma' - \sigma)\, d\sigma', \tag{7}$$

which is equivalent to minimizing the conditional risk $f_{\delta(\sigma)} = \operatorname{argmin}_f R(f \,|\, \sigma)$. Because we are minimizing the conditional risk at a particular σ, $f_{\delta(\sigma)}$ gives the best individual estimate at σ. However, having the best estimate at σ does not mean that $f_{\delta(\sigma)}$ can generalize. It is possible that $f_{\delta(\sigma)}$ performs well for one σ but poorly for other σ's. However, the ensemble of all these point-wise estimates $\{f_{\delta(\sigma)}\}$ will form the lower bound of the conditional risks such that $R(f_{\delta(\sigma)} \mid \sigma) \leq R(f_p \mid \sigma)$ at every σ.

3.4. Main Problem (P1)

We now state the main problem. The problem we want to solve is the following constrained optimization:

$$f^* \stackrel{\text{def}}{=} \operatorname*{argmin}_{f\in\mathcal{F}} \quad R(f) \tag{P1}$$
$$\text{subject to} \quad \sup_{\sigma\in\Omega}\Big\{R(f|\sigma) - R(f_{\delta(\sigma)}|\sigma)\Big\} \le \epsilon.$$

The objective function reflects our original goal of minimizing the overall risk. However, instead of doing it without any constraint (which has a trivial solution of $f^* = f_p$), we introduce a constraint that the gap between the current estimator f and the best individual $f_{\delta(\sigma)}$ is no worse than ϵ, where ϵ is some threshold. The intuition here is that we are willing to sacrifice some of the overall risk by limiting the gap between f and $f_{\delta(\sigma)}$ so that we have a consistent performance over the entire range of noise levels.

Referring back to Figure 1, we note that the black curve is $R(f_{\delta(\sigma)}|\sigma)$. The blue curve is $R(f_p|\sigma)$ for the case where $p(\sigma)$ is a uniform distribution. The orange curve is $R(f^*|\sigma)$. We show in Section 4.2 that f^* is equivalent to f_π for some $\pi(\sigma)$. Note that all curves are the conditional risks.

4. Dual Ascent

In this section we discuss how to solve (P1). Solving (P1) is challenging because minimizing over f involves updating the estimator f which could be nonlinear w.r.t. the loss. To address this issue, we first show that as long as the admissible set $\mathcal{F}$ is convex, (P1) is convex even if the estimators f themselves are non-convex. We then derive an algorithm to solve the dual problem.

4.1. Convexity of (P1)

We start by showing that under mild conditions, (P1) is convex.

Lemma 2. *Let $\mathcal{F}$ be a closed and convex set. Then, for any convex loss function $\mathcal{L}$, the risk $R(f)$ and the conditional risk $R(f|\sigma)$ are convex in f, for any $\sigma \in \Omega$.*

Proof. Let f_1 and f_2 be two estimators in $\mathcal{F}$ and let $\lambda \in [0,1]$ be a constant. Then, by the convexity of $\mathcal{L}$, the conditional risk $R(\cdot\,|\,\sigma)$ satisfies

$$\begin{aligned} R(\lambda f_1 + (1-\lambda)f_2\,|\,\sigma) &= \mathbb{E}\big\{\mathcal{L}(\lambda f_1 + (1-\lambda)f_2)\,|\,\sigma\big\} \\ &\le \mathbb{E}\big\{\lambda\mathcal{L}(f_1) + (1-\lambda)\mathcal{L}(f_2)\,|\,\sigma\big\} \\ &= \lambda R(f_1|\sigma) + (1-\lambda)R(f_2|\sigma), \end{aligned}$$

which is convex. The overall risk $R(f)$ is found by taking the expectation of the conditional risk over σ. Since taking expectation is equivalent to integrating the conditional risk times the distribution $p(\sigma)$ (which is positive), convexity preserves and so $R(f)$ is also convex. □

We emphasize that the convexity of $R(\cdot)$ is defined w.r.t. f and not the underlying parameters (e.g., the network weights). For any convex combination of the parameters θ's, we have that $R(f(\cdot\,|\,\lambda\theta_1 + (1-\lambda)\theta_2)) \not\le \lambda R(f(\cdot\,|\,\theta_1)) + (1-\lambda)R(f(\cdot\,|\,\theta_2))$ because f is not necessarily convex.

The following corollary shows that the optimization problem (P1) is convex.

Corollary 1. *Let $\mathcal{F}$ be a closed and convex set. Then, for any convex loss function $\mathcal{L}$, (P1) is convex in f.*

Proof. Since the objective function R is convex (by Lemma 2), we only need to show that the constraint set is also convex. Note that the "sup" operation is equivalent to requiring $R(f|\sigma) - R(f_{\delta(\sigma)|\sigma}) \le \epsilon$ for *all* $\sigma \in \Omega$. Since $R(f_{\delta(\sigma)}|\sigma)$ is constant w.r.t. f, we can define $\epsilon(\sigma) \stackrel{\text{def}}{=} \epsilon + R(f_{\delta(\sigma)}|\sigma)$ so that the constraint becomes $R(f|\sigma) - \epsilon(\sigma) \le 0$. Consequently the constraint set is convex because the conditional risk $R(f|\sigma)$ is convex. □

The convexity of $\mathcal{F}$ is subtle but essential for Lemma 2 and Corollary 1. In a standard optimization over $\mathbb{R}^n$, the convexity is granted if the admissible set is an interval in $\mathbb{R}^n$. In our problem, $\mathcal{F}$ denotes the set of all admissible estimators, which by construction are parametrized by θ. Thus, the convexity of $\mathcal{F}$ requires that a convex combination of two admissible f's remains admissible. All estimators based on generalized linear models satisfy this property. However, for deep neural networks it is generally unclear how the topology looks like although some recent studies are suggesting negative results (Petersen et al., 2018). Nevertheless, even if $\mathcal{F}$ is non-convex, we can solve the dual problem which is always convex. The dual solution provides the convex-relaxation of the primal problem. The duality gap is zero when the Slater's condition holds, i.e., when $\mathcal{F}$ is convex and ϵ is chosen such that the constraint set is strictly feasible.

4.2. Dual of (P1)

Let us develop the dual formulation of (P1). The dual problem is defined through the Lagrangian:

$$\begin{aligned} &L(f,\lambda) \\ &\stackrel{\text{def}}{=} R(f) + \int \Big\{R(f|\sigma) - \underbrace{\big(R(f_{\delta(\sigma)}|\sigma) + \epsilon\big)}_{\stackrel{\text{def}}{=}\epsilon(\sigma)}\Big\}\lambda(\sigma)d\sigma \\ &= \int R(f|\sigma)\Big\{p(\sigma) + \lambda(\sigma)\Big\}d\sigma - \int \epsilon(\sigma)\lambda(\sigma)d\sigma, \end{aligned} \tag{8}$$

by which we can determine the Lagrange dual function as

$$g(\lambda) = \inf_f L(f,\lambda), \tag{9}$$

and the dual solution:

$$\lambda^* = \underset{\lambda \geq 0}{\text{argmax}} \; g(\lambda)$$
$$= \underset{\lambda \geq 0}{\text{argmax}} \left\{ \inf_f \left\{ \int R(f|\sigma)\big[p(\sigma) + \lambda(\sigma)\big] d\sigma \right\} - \int \epsilon(\sigma)\lambda(\sigma) d\sigma \right\}. \quad (10)$$

Given the dual solution λ^*, we can translate it back to the primal solution $\widehat{f}$ by minimizing the inner problem in (10), which is

$$\widehat{f} = \underset{f}{\text{argmin}} \int R(f|\sigma) \underbrace{\Big\{ p(\sigma) + \lambda^*(\sigma) \Big\}}_{\stackrel{\text{def}}{=} \pi^*(\sigma)} d\sigma. \quad (11)$$

This minimization is nothing but training the estimator f using samples who noise levels are distributed according to $p(\sigma) + \lambda^*(\sigma)$. [1] Therefore, by solving the dual problem we have simultaneously obtained the distribution $\pi^*(\sigma)$, which is $\pi^*(\sigma) = p(\sigma) + \lambda^*(\sigma)$, and the estimator $\widehat{f}$ trained using the distribution π^*.

As we have discussed, if the admissible set $\mathcal{F}$ is convex then (P1) is convex and so $\widehat{f}$ is exactly the primal solution f^*. If $\mathcal{F}$ is not convex, then $\widehat{f}$ is the solution of the convex relaxation of (P1). The duality gap is $R(f^*) - g(\lambda^*)$.

4.3. Dual Ascent Algorithm

The algorithm for solving the dual is based on the fact that the point-wise $\inf_f \; L(f, \lambda)$ is concave in λ. As such, one can use the standard dual ascent method to find the solution. The idea is to sequentially update λ's and f's via

$$f^{t+1} = \underset{f}{\text{argmin}} \int R(f|\sigma) \Big\{ p(\sigma) + \lambda^t(\sigma) \Big\} d\sigma \quad (12)$$

$$\lambda^{t+1}(\sigma) = \Big[\lambda^t(\sigma) + \alpha^t(\sigma) \Big\{ R(f^{t+1}|\sigma) - \epsilon(\sigma) \Big\} \Big]_+ \quad (13)$$

Here, α^t is the step size of the gradient ascent step, and $[\,\cdot\,]_+ = \max(\cdot, 0)$ returns the positive part of the argument. At each iteration, (12) is solved by training an estimator using noise samples drawn from the distribution $\pi(\sigma)^t = p(\sigma) + \lambda^t(\sigma)$. The λ-step in (13) computes the conditional risk $R(f^{t+1}|\sigma)$ and updates λ.

Since the dual is convex, the dual ascent algorithm is guaranteed to converge to the dual solution using an appropriate step size. We refer readers to standard texts, e.g., (CMU).

[1] For $p(\sigma) + \lambda(\sigma)$ to be a legitimate distribution, we need to normalize it by the constant $Z = \int \{p(\sigma) + \lambda(\sigma)\} d\sigma$. But as far as the minimization in (11) is concerned, the constant is unimportant.

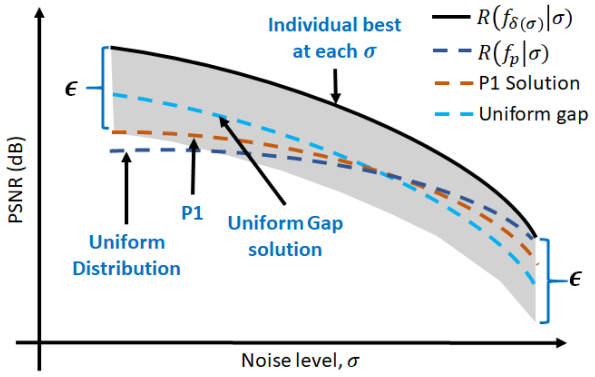


Figure 2. Difference between (P1) and (P2). In (P1), the solution only needs to make sure that the worst case gap is upper bounded by ϵ. There is no control over places where the gap is intrinsically less than ϵ. Uniform Gap problem (P2) addresses this issue by forcing the gap to be uniform. Note that neither (P1) nor (P2) is absolutely more superior. It is a trade-off between the noise levels, and how much we know about the testing distribution p.

5. Uniform Gap

The solution of (P1) depends on the tolerance ϵ. This tolerance ϵ cannot be arbitrarily small, or otherwise the constraint set will become empty. The smallest ϵ which still ensures a non-empty constraint set is defined as $\epsilon_{\min}$. The goal of this section is to determine $\epsilon_{\min}$ and discuss its implications.

5.1. The Uniform Gap Problem (P2)

The motivation of studying the so-called Uniform Gap problem is the inadequacy of (P1) when the tolerance ϵ is larger than $\epsilon_{\min}$ (i.e., we tolerate more than needed). The situation can be understood from Figure 2. For any allowable ϵ, the solution returned by (P1) can only ensure that the largest gap is no more than ϵ. It is possible that the high-ends have a significantly smaller gap than the low-ends. The gap will become uniform only when $\epsilon = \epsilon_{\min}$ which is typically not known a-priori.

If we want to maintain a constant gap throughout the entire range of σ, then the optimization goal will become minimizing the maximum risk gap and not worry about the overall risk. In other words, we solve the following problem:

$$f^* = \underset{f}{\text{argmin}} \sup_{\sigma \in \Omega} \Big\{ R(f|\sigma) - R(f_{\delta(\sigma)}|\sigma) \Big\}. \quad \text{(P2)}$$

When (P2) is solved, the corresponding risk gap is exactly $\epsilon_{\min}$, defined as

$$\epsilon_{\min} \stackrel{\text{def}}{=} \sup_{\sigma \in \Omega} \Big\{ R(f^*|\sigma) - R(f_{\delta(\sigma)}|\sigma) \Big\}. \quad (14)$$

The supremum in the above equation can be lifted because by construction, (P2) guarantees a constant gap for all σ.

The difference between (P2) and (P1) is the switched roles of the objective function and the constraint. In (P1), the tolerance ϵ defines a user-controlled upper bound on the risk gap, whereas in (P2) the ϵ is eliminated. Note that the omission of ϵ in (P2) does not imply better or worse since (P1) and (P2) are serving two different goals. (P1) utilizes the underlying testing distribution $p(\sigma)$ whereas (P2) does not. It is possible that $p(\sigma)$ is skewed towards high noise scenarios so that a constant risk gap will suffer from insufficient performance at high-noise and over-perform at low-noise which does not matter because of $p(\sigma)$.

In practice (i.e., in the absence of any knowledge about an appropriate ϵ), one can solve (P2) first to obtain the tightest gap $\epsilon_{\min}$. Once $\epsilon_{\min}$ is determined, we can choose an $\epsilon > \epsilon_{\min}$ to minimize the overall risk using (P1).

5.2. Algorithm for Solving (P2)

The algorithm to solve (P2) is slightly different from that of (P1) because of the omission of the constraint.

We first rewrite problem (P2) as

$$\begin{aligned} &\underset{f,t}{\text{minimize}} && t \\ &\text{subject to} && R(f|\sigma) - \underbrace{R(f_{\delta(\sigma)}|\sigma)}_{\stackrel{\text{def}}{=} r(\sigma)} \le t, \quad \forall \sigma. \end{aligned} \tag{15}$$

Then the Lagrangian is defined as

$$\begin{aligned} L(f,t,\lambda) &\stackrel{\text{def}}{=} t + \int \Big\{ R(f|\sigma) - r(\sigma) - t \Big\} \lambda(\sigma)\, d\sigma \qquad (16) \\ &= t \left(1 - \int \lambda(\sigma) d\sigma \right) + \int \Big\{ R(f|\sigma) - r(\sigma) \Big\} \lambda(\sigma) d\sigma. \end{aligned}$$

Minimizing over f and t yields the dual function:

$$\begin{aligned} g(\lambda) &\stackrel{\text{def}}{=} \inf_{f,t} L(f,t,\lambda) \qquad (17) \\ &= \begin{cases} \inf\limits_{f} \int \big[R(f|\sigma) - r(\sigma) \big] \lambda(\sigma) d\sigma, & \text{if } \int \lambda(\sigma) d\sigma = 1, \\ -\infty, & \text{otherwise.} \end{cases} \end{aligned}$$

Consequently, the dual problem is defined as

$$\begin{aligned} \lambda^* = \underset{\lambda \ge 0}{\text{argmax}} \ \inf_{f} &\left\{ \int \big[R(f|\sigma) - r(\sigma) \big] \lambda(\sigma) d\sigma \right\} \qquad (18) \\ &\text{subject to} \int \lambda(\sigma) d\sigma = 1. \end{aligned}$$

Again, if $\mathcal{F}$ is convex then solving the dual problem (18) is necessary and sufficient to determine the primal problem (15) which is equivalent to (P2). The dual problem is solvable using the dual ascent algorithm, where we update λ and f according to the following sequence:

$$f^{t+1} = \underset{f}{\text{argmin}} \left\{ \int \big[R(f|\sigma) - r(\sigma) \big] \lambda^t(\sigma) d\sigma \right\} \tag{19}$$

$$\lambda^{t+\frac{1}{2}} = \Big[\lambda^t + \alpha^t \big(R(f^{t+1}|\sigma) - r(\sigma) \big) \Big]_+ \tag{20}$$

$$\lambda^{t+1} = \lambda^{t+\frac{1}{2}} \Big/ \int \lambda^{t+\frac{1}{2}}(\sigma) d\sigma. \tag{21}$$

Here, (19) solves the inner optimization in (18) by fixing a λ, and (20) is a gradient ascent step for the dual variable. The normalization in (21) ensures that the constraint of (18) is satisfied. The non-negativity operation $[\,\cdot\,]_+$ in (20) can be lifted because by definition $r(\sigma) \stackrel{\text{def}}{=} R(f_{\delta(\sigma)}|\sigma) \ge R(f|\sigma)$ for all σ. The final sampling distribution is $\pi^*(\sigma) = \lambda^*(\sigma)$.

Like (P1), the dual ascent algorithm for (P2) has guaranteed convergence as long as the loss function $\mathcal{L}$ is convex.

6. Practical Considerations

The actual implementation of the dual ascent algorithms for (P1) and (P2) require additional modifications. We list a few of them here.

Finite Epochs. In principle, the f-subproblems in (12) and (19) are determined by training a network completely using the sample distributions at the t-th iteration $\pi^t(\sigma) = p(\sigma) + \lambda^t(\sigma)$ and $\pi^t(\sigma) = \lambda^t(\sigma)$, respectively. However, in practice, we can reduce the training time by training the network inexactly. Depending on the specific network architecture and problem type, the number of epochs varies between 10 - 50 epochs per dual ascent iteration.

Discretize Noise Levels. The theoretical results presented in this paper are based on continuous distributions $\lambda(\sigma)$ and $p(\sigma)$. In practice, a continuum is not necessary since nearby noise levels are usually indistinguishable visually. As such, we discretize the noise levels in a finite number of bins. And we use the average-value of each bin as the representative noise level for the bin, so that the integration can be simplified to summation.

log-Scale Constraints. Most image restoration applications measure the restoration quality in the log-scale, e.g., the peak signal-to-noise ratio (PSNR) which is defined as $\text{PSNR} = -10 \log_{10} \text{MSE}$ where MSE is the mean squared error. Learning in the log-scale can be achieved by enforcing constraint in the log-space.

We define the the log-scale risk function as:

$$R_{\log}(f|\sigma) \stackrel{\text{def}}{=} \mathbb{E}\Big[\log \mathcal{L}(f(\boldsymbol{x}_\sigma), \boldsymbol{y}) \,\Big|\, \sigma \Big]. \tag{22}$$

With this definition, it follows the the constraints in the log-scale are represented as $\sup_{\sigma \in \Omega} \{ R_{\log}(f|\sigma) -$

$R_{\log}(f_{\delta(\sigma)}|\sigma)\} \leq \epsilon$. To turn this log-scale constraint into a linear form, we use the follow lemma by exploiting the fact that the risk gap is typically small.

Lemma 3. *The log-scale constraint*

$$\sup_{\sigma\in\Omega}\left\{R_{\log}(f|\sigma) - R_{\log}(f_{\delta(\sigma)}|\sigma)\right\} \leq \epsilon \tag{23}$$

can be approximated by

$$\sup_{\sigma\in\Omega}\left\{\frac{\mathbb{E}[\mathcal{L}(f(\boldsymbol{x}_\sigma),\boldsymbol{y})]}{L_\delta(\sigma)}\right\} \leq 1+\epsilon, \tag{24}$$

where $L_\delta(\sigma)$ is a constant (w.r.t. f) such that the log of $L_\delta(\sigma)$ equals $R_{\log}(f_{\delta(\sigma)}|\sigma)$:

$$\log L_\delta(\sigma) \overset{def}{=} \mathbb{E}\left[\log \mathcal{L}(f_\delta(\boldsymbol{x}_\sigma),\boldsymbol{y})|\sigma\right]. \tag{25}$$

Proof. First, we observe that $R_{\log}(f_{\delta(\sigma)}|\sigma)$ is a deterministic quantity and is independent of f. Using the fact that $L_\delta(\sigma)$ is a deterministic constant, we can show that

$$\begin{aligned}
&R_{\log}(f|\sigma) - R_{\log}(f_{\delta(\sigma)}|\sigma)\\
&= \mathbb{E}\left[\log \mathcal{L}(f(\boldsymbol{x}_\sigma),\boldsymbol{y})|\sigma\right] - \log L_\delta(\sigma)\\
&= \mathbb{E}\left[\log\left(\frac{\mathcal{L}(f(\boldsymbol{x}_\sigma),\boldsymbol{y})}{L_\delta(\sigma)}\right)\,\middle|\,\sigma\right]\\
&= \mathbb{E}\left[\log\left(1+\frac{\mathcal{L}(f(\boldsymbol{x}_\sigma),\boldsymbol{y})-L_\delta(\sigma)}{L_\delta(\sigma)}\right)\,\middle|\,\sigma\right]\\
&\approx \mathbb{E}\left[\frac{\mathcal{L}(f(\boldsymbol{x}_\sigma),\boldsymbol{y})-L_\delta(\sigma)}{L_\delta(\sigma)}\,\middle|\,\sigma\right],
\end{aligned}$$

where we used the fact that $\mathcal{L}(f(\boldsymbol{x}_\sigma),\boldsymbol{y}) - L_\delta(\sigma) \ll L_\delta(\sigma)$ so that $\log(1+x)\approx x$. Putting these into the constraint $R_{\log}(f|\sigma) - R_{\log}(f_{\delta(\sigma)}|\sigma)\leq\epsilon$ and rearranging the terms completes the proof. □

The consequence of the above analysis leads to the following approximate problem for training in the log-scale:

$$\begin{aligned}
f^* \overset{\text{def}}{=} \operatorname*{argmin}_f \quad & R(f), \qquad\qquad \text{(P1-log)}\\
\text{s.t.} \quad & \sup_{\sigma\in\Omega}\left\{\frac{R(f|\sigma)}{L_\delta(\sigma)}\right\} \leq 1+\epsilon.
\end{aligned}$$

The implication of (P1-log) is that the optimization problem with log-scale constraints can be solved using the linear-scale approaches. Notice that the new distribution is now $\pi(\sigma) = p(\sigma) + \frac{\lambda(\sigma)}{L_\delta(\sigma)}$. The other change is that we replace $R(f_{\delta(\sigma)}|\sigma)$ with $L_\delta(\sigma)$, which are determined offline.

7. Experiments

We evaluate the proposed framework through two experiments. The first experiment is based on a linear estimator where analytic solutions are available to verify the dual ascent algorithm. The second experiment is based on training a real deep neural network.

7.1. Linear Estimator

We consider a linear (scalar) estimator so that we can access the analytic solutions. We define the clean signal as $y\sim\mathcal{N}(0,\sigma_y^2)$ and the noisy signal as $x = y+\sigma\eta$, where $\eta\sim\mathcal{N}(0,1)$. The estimator we choose here is $f_\pi(x) = a_\pi x$ for some parameter a_π depending on the underlying sampling distribution π.

Because of the linear model formulation, we can train the estimator $\widehat{a}_\pi$ using closed-form equation as

$$\widehat{a}_\pi = \operatorname*{argmin}_a \int \mathbb{E}[(ax-y)^2|\sigma]\pi(\sigma)d\sigma = \frac{\sigma_y^2}{\sigma_y^2+\overline{\sigma}_\pi^2},$$

where $\overline{\sigma}_\pi^2 \overset{\text{def}}{=} \int\sigma^2\pi(\sigma)d\sigma$. Substituting $\widehat{a}_\pi$ into the loss we can show that the conditional risk is

$$\begin{aligned}
&R(f_\pi|\sigma) = \mathbb{E}[(\widehat{a}_\pi x - y)^2|\sigma]\\
&= \frac{\sigma_y^4\left[\sigma_y^2(\sigma_y^2+\sigma^2) - 2\sigma_y^2(\sigma_y^2+\overline{\sigma}_\pi^2) + (\sigma_y^2+\overline{\sigma}_\pi^2)^2\right]}{(\sigma_y^2+\overline{\sigma}_\pi^2)^2}.
\end{aligned}$$

Based on this condition risks, we can run the dual ascent algorithm to alternatingly estimate π and $\widehat{a}_\pi$ according to (P1). Figure 3 shows the conditional risks returned by different iterations of the dual ascent algorithm. In this numerical example, we let $\sigma_y = 10$ and $\epsilon = 9$. Observe that as the dual ascent algorithm proceeds, the worst case gap is reducing [2]. When the algorithm converges, it matches exactly with the theoretical solution.

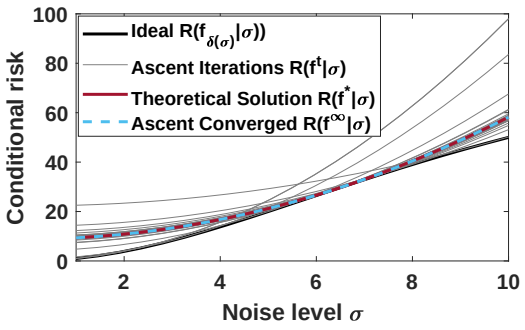


Figure 3. Conditional risks of the linear problem. As the dual ascent algorithm proceeds, the risk approaches the optimal solution.

7.2. Deep Neural Networks

The second experiment evaluates the effectiveness of the proposed framework on real deep neural networks for the task of denoising. We shall focus on the MSE loss with PSNR constraints, although our theory applies to other loss

[2] The small gap in the middle of the plot is intrinsic to this problem, since for any $\overline{\sigma}_\pi^2$ there always exists a σ such that $\overline{\sigma}_\pi^2 = \sigma$. At this σ, the conditional risk will always touch the ideal curve.

Noise level (σ)	0-10	10-20	20-30	30-40	40-50	50-60	60-70	70-80	80-90	90-100
	Ideal (Best Individually Trained Denoisers)									
PSNR	38.04	31.73	29.23	27.72	26.66	25.86	25.24	24.70	24.25	23.84
	Uniform Distribution									
Distribution	10.0%	10.0%	10.0%	10.0%	10.0%	10.0%	10.0%	10.0%	10.0%	10.0%
PSNR	37.24	31.41	29.04	27.60	26.58	25.81	25.19	24.67	24.23	23.84
	Solution to (P1) with 0.4dB gap									
Distribution	32.7%	12.0%	9.4%	7.9%	6.8%	6.3%	6.4%	6.2%	6.2%	6.1%
PSNR	37.64	31.46	29.03	27.58	26.56	25.78	25.15	24.63	24.19	23.80
	Solution to (P2)									
Distribution	81.3%	7.6%	3.4%	2.0%	1.3%	1.0%	0.9%	0.9%	0.8%	0.8%
PSNR	37.86	31.54	29.06	27.57	26.53	25.74	25.10	24.57	24.12	23.70

Table 1. Results of Section 7.2. This table shows the PSNR values returned by one-size-fits-all DnCNN denoisers whose sample distributions are defined according to (i) uniform distribution, (ii) solution of (P1), and (iii) solution of (P2).

functions such as SSIM (Zhou Wang et al., 2004) and MS-SSIM (Wang et al., 2003) also as long as they are convex. The noise model we assume is that $\boldsymbol{x}_\sigma = \boldsymbol{y} + \sigma\boldsymbol{\eta}$, where $\boldsymbol{\eta} \sim \mathcal{N}(0, \boldsymbol{I})$ with $\sigma \in [0, 100]$ (w.r.t. an 8-bit signal of 256 levels). The network we consider is a 20-layer DnCNN (Zhang et al., 2017). We choose DnCNN just for demonstration. Since our framework does not depend on a specific network architecture, the theoretical results hold regardless the choice of the networks.

The training procedure is as follows. The training set consists of 400 images from the dataset in (Martin et al., 2001). Each image has a size of 180×180. We randomly crop 50×50 patches from these images to construct the training set. The total number of patches we used is determined by the mini-batch size of the training algorithm. Specifically, for each dual ascent iteration we use 3000 mini-batches where each batch consists of 128 patches. This gives us 384k training patches per epoch. To create the noisy training samples, for each patch we add additive i.i.d. Gaussian noise where the noise level is randomly drawn from the distribution $\pi(\sigma)$. The noise generation process is done online. We run our proposed algorithm for 25 dual ascent iterations, where each iteration consists of 10 epochs. For computational efficiency, we break the noise range $[0, 100]$ into 10 equally sized bins. For example, a uniform distribution corresponds to allocating 10% of the total number of training samples per bin. The validation set consists of 12 "standard images" (e.g., Lena). The testing set is the BSD68 dataset (Roth & Black, 2005), tested individually for every noise bin. The testing distribution $p(\sigma)$ for (P1) is assumed to be uniform in Figure 4. Two other distributions are illustrated in Figure 5. Notice that (P2) does not require the testing distribution to be known.

The average PSNR values (conditional on σ) are reported in Table 1 and the performance gaps are illustrated in Figure 4. Specifically, the first two rows of the Table show the PSNR of the best individually trained denosiers and the uniform distributions. The proposed sampling distributions and the corresponding PSNR values are shown in the third row for (P1) and the fourth row for (P2). For (P1), we set the tolerance level as 0.4dB. Table 1 and Figure 4 confirm the validity of our method. A more interesting observation is the percentages of the training samples. For (P1), we need to allocate 32.7% of the data to low-noise, and this percentage goes up to 81.3% for (P2). This suggests that the optimal sampling distribution could be substantially different from the uniform distribution we use today.

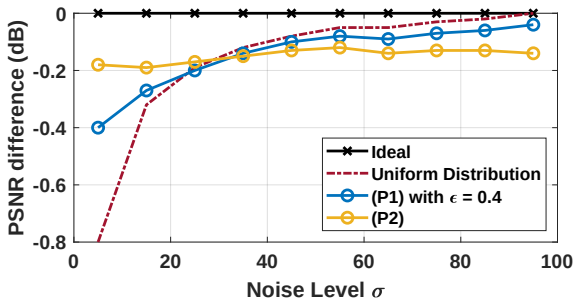
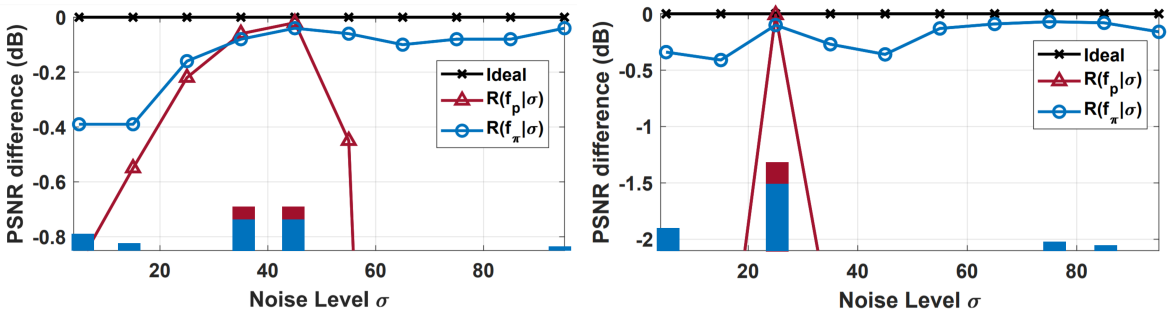


Figure 4. This figure shows the PSNR difference between the one-size-fits-all denoisers and the ideal denoiser. Observe that the uniform distribution favors high-noise cases and performs poorly on low-noise cases. By using the proposed algorithm we are able to allocate training samples such that the gap is consistent across the range. (P1) ensures that the gap will not exceed 0.4dB, whereas (P2) ensures that the gap is constant.

8. Discussions

8.1. Consistent gap = better?

It is important to note that one-size-fits-all denosiers are about the trade-off between high-noise and low-noise cases; we offer more degrees of freedom for the low-noise cases because the high-noise cases can be learned well using fewer samples. However, achieving a consistent gap does not mean that we are doing "better". The solution of (P2) is not necessarily "better" than the solution of (P1). The ultimate decision is application specific. If we care more



(a) Hypothesized distribution is Uniform[30, 50] (b) Hypothesized distribution is Uniform[20, 30].

Figure 5. Usage of (P1) when we are not certain about the true distribution. (a) The true distribution is unknown but we hypothesize that it is Uniform[30,50]. If we train the network using this distribution, we obtain the red curve. (P1) starts with this hypothesis, and returns the blue curve. (b) Same experiment by hypothesizing that the distribution is Uniform[20,30]. Observe the robust performance of our method in both cases. The experimental setup is the same as Section 7.2.

about the heavy noise cases such as imaging in the dark (e.g., (Gnanasambandam & Chan, 2020; Choi et al., 2018)) and we are willing to compromise some performance for the weak noise cases, then (P1) could be a better than the uniform gap solution returned by (P2). Vice versa, if we know nothing about the testing distribution and we want to be conservative, then (P2) is more useful.

Another consideration is how much we know about $p(\sigma)$. If we are absolutely certain that the noise is concentrated at a single value, then we should just allocate all the samples at that noise level. However, if we know something about $p(\sigma)$ but we are not absolutely sure, then (P1) can provide the worst case performance guarantee. This is illustrated in Figure 5, where we solved (P1) using two hypothesized distributions. It can be observed that if we train the network using the hypothesized distributions, the performance could be bad for extreme situations. In contrast, (P1) compromises the peak performance by offering more robust performance in other situations.

8.2. Rule-of-thumb distribution — the "80-20" rule

Suppose that we are only looking at image denoisers with $p(\sigma)$ being uniform, and our goal is to achieve a consistent gap, then we can construct some "rule-of-thumb" distributions that are applicable to a few network architectures. Figure 6 shows three deep networks: REDNet (Mao et al., 2016b), DnCNN (Zhang et al., 2017), FFDNet (Zhang et al., 2018), all trained using a so-called "80-20" rule. In this rule, we allocate the majority of the samples to the weak cases and a few to the strong cases. The exact percentage of the "80-20" rule is network dependent but the trend is usually similar. For example, in Figure 6 we allocate 70% to [0,10], 15% to [10,20], 8% to [20,30], 5% to [30,40], and 2% to [40,50] to all the three networks. While there are some fluctuations of the PSNR differences, in general the resulting curves are quite uniform.

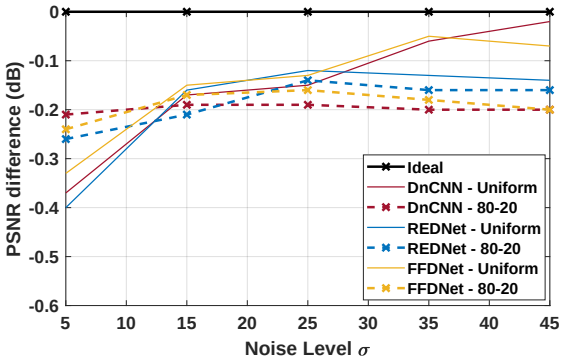


Figure 6. "80-20" Rule. We train the network with training samples drawn from different different noise levels according to the following distribution. $[0,10]-70\%$, $[10,20]-15\%$, $[20,30]-8\%$, $[30,40]-5\%$, $[40-50]-2\%$. We observe that this distribution gives reasonably consistent performance at all the noise levels over all the denoisers considered here.

9. Conclusion

Imbalanced sampling of the training set is arguably very common in image restoration and related tasks. This paper presents a framework which allows us to allocate training samples so that the overall performance of the one-size-fits-all denoiser is consistent across all noise levels. The convexity of the problem, the minimax formulation, and the dual ascent algorithm appear to be general for all learning-based estimators. The idea is likely to be applicable to adversarial training in classification tasks.

Acknowledgement

The work is supported, in part, by the US National Science Foundation under grants CCF-1763896 and CCF-1718007. The authors thank Yash Sanghvi and Guanzhe Hong for many insightful discussions, and the anonymous reviewers for the constructive feedback.

References

Machine Learning 10-725, CMU. URL https://www.stat.cmu.edu/~ryantibs/convexopt-F18/lectures/dual-ascent.pdf.

Calders, T. and Verwer, S. Three naive Bayes approaches for discrimination-free classification. *Data Mining and Knowledge Discovery*, 21(2):277–292, September 2010.

Chaloner, K. and Verdinelli, I. Bayesian experimental design: A review. *Statistical Science*, pp. 273–304, August 1995.

Chan, S. H. Performance analysis of Plug-and-Play ADMM: A graph signal processing perspective. *IEEE Transactions on Computational Imaging*, 5(2):274–286, Jun. 2019.

Chan, S. H., Wang, X., and Elgendy, O. A. Plug-and-Play ADMM for image restoration: Fixed-point convergence and applications. *IEEE Transactions on Computational Imaging*, 3(1):84–98, November 2016.

Choi, J., Elgendy, O. A., and Chan, S. H. Image reconstruction for quanta image sensors using deep neural networks. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 6543–6547, Apr. 2018.

Choi, J., Elgendy, O., and Chan, S. H. Optimal combination of Image Denoisers. *IEEE Transactions on Image Processing*, 28(8), August 2019.

Gal, Y., Islam, R., and Ghahramani, Z. Deep Bayesian active learning with image data. In *International Conference on Machine Learning*, volume 70, pp. 1183–1192, August 2017.

Gao, R. and Grauman, K. On-demand learning for deep image restoration. In *International Conference on Computer Vision*, pp. 1086–1095, 2017.

Gharbi, M., Chaurasia, G., Paris, S., and Durand, F. Deep Joint Demosaicking and Denoising. *ACM Transactions on Graphics*, 35(6):191:1–191:12, November 2016.

Gnanasambandam, A. and Chan, S. H. Image classification in the dark using quanta image sensors. *arXiv: 2006.02026*, 2020.

Hardt, M., Price, E., and Srebro, N. Equality of opportunity in supervised learning. In *Advances in Neural Information Processing Systems*, pp. 3315–3323, December 2016.

Kamishima, T., Akaho, S., Asoh, H., and Sakuma, J. Fairness-aware classifier with prejudice remover regularizer. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 35–50, September 2012.

Kim, Y., Soh, J. W., and Cho, N. I. Adaptively tuning a convolutional neural network by gate process for image denoising. *IEEE Access*, 7:63447–63456, May 2019.

Mao, X., Shen, C., and Yang, Y. Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. In *Advances in Neural Information Processing Systems*, pp. 2802–2810, December 2016a.

Mao, X., Shen, C., and Yang, Y.-B. Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. In *Advances in neural information processing systems*, pp. 2802–2810, 2016b.

Márquez-Neila, P., Salzmann, M., and Fua, P. Imposing hard constraints on deep networks: Promises and limitations. *arXiv:1706.02025*, 2017.

Martin, D., Fowlkes, C., Tal, D., and Malik, J. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *International Conference on Computer Vision*, volume 2, pp. 416–423, July 2001.

Pathak, D., Krahenbuhl, P., and Darrell, T. Constrained Convolutional Neural Networks for weakly supervised segmentation. In *International Conference on Computer Vision*, pp. 1796–1804, December 2015.

Pedreshi, D., Ruggieri, S., and Turini, F. Discrimination-aware data mining. In *International Conference on Knowledge Discovery and Data Mining*, pp. 560–568, August 2008.

Petersen, P., Raslan, M., and Voigtlaender, F. Topological properties of the set of functions generated by neural networks of fixed size. *arXiv:1806.08459*, 2018.

Platt, J. C. and Barr, A. H. Constrained differential optimization. In *International Conference on Neural Information Processing Systems*, pp. 612–621, 1987.

Remez, T., Litany, O., Giryes, R., and Bronstein, A. M. Deep class-aware image denoising. In *International Conference on Image Processing*, pp. 1895–1899, July 2017.

Roth, S. and Black, M. J. Fields of experts: A framework for learning image priors. In *Computer Vision and Pattern Recognition*, volume 2, pp. 860–867, June 2005.

Sener, O. and Savarese, S. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018.

Settles, B. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2009.

Wang, Z., Simoncelli, E. P., and Bovik, A. C. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pp. 1398–1402. IEEE, 2003.

Zafar, M. B., Valera, I., Rodriguez, M. G., and Gummadi, K. P. Fairness constraints: Mechanisms for fair classification. *arXiv:1507.05259*, 2015.

Zafar, M. B., Valera, I., Gomez Rodriguez, M., and Gummadi, K. P. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *International World Wide Web Conference*, pp. 1171–1180, April 2017.

Zak, S. H., Upatising, V., and Hui, S. Solving Linear Programming problems with Neural Networks: A comparative study. *IEEE Transactions on Neural Networks*, 6(1): 94–104, January 1995.

Zhang, K., Zuo, W., Gu, S., and Zhang, L. Learning deep CNN denoiser prior for image restoration. In *Computer Vision and Pattern Recognition*, pp. 2808–2817, 2017.

Zhang, K., Zuo, W., and Zhang, L. FFDNet: Toward a fast and flexible solution for CNN-based image denoising. *IEEE Transactions on Image Processing*, 27(9):4608–4622, May 2018.

Zhou Wang, Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.