Unsupervised Neural Universal Denoiser for Finite-Input General-Output Noisy Channel

Tae-Eon Park and Taesup Moon
Department of Electrical and Computer Engineering
Sungkyunkwan University (SKKU), Suwon, Korea 16419
{pte1236, tsmoon}@skku.edu

Abstract

We devise a novel neural network-based universal denoiser for the finite-input, general-output (FIGO) channel. Based on the assumption of known noisy channel densities, which is realistic in many practical scenarios, we train the network such that it can denoise as well as the *best* sliding window denoiser for *any* given underlying clean source data. Our algorithm, dubbed as Generalized CUDE (Gen-CUDE), enjoys several desirable properties; it can be trained in an unsupervised manner (solely based on the noisy observation data), has much smaller computational complexity compared to the previously developed universal denoiser for the same setting, and has much tighter upper bound on the denoising performance, which is obtained by a theoretical analysis. In our experiments, we show such tighter upper bound is also realized in practice by showing that Gen-CUDE achieves much better denoising results compared to other strong baselines for both synthetic and real underlying clean sequences.

1 Introduction

Denoising is a ubiquitous problem that lies at the heart of a wide range of fields such as statistics, engineering, bioinformatics, and machine learning. While numerous approaches have been undertaken, many of them focused on the case for which both the input and output of a noisy channel are continuous-valued (Donoho and Johnstone, 1995; Elad and Aharon, 2006; Dabov et al., 2007). In addition, discrete denoising, in which the input and output of the channel take their values in some finite set, have also been considered more recently (Weissman et al., 2005; Moon et al., 2016; Moon and Weissman, 2009).

In this paper, we focus on the hybrid case, namely, the setting in which the underlying clean input source is finite-valued, while the noisy channel output can be continuous-valued. Such scenario naturally occurs in several applications; for example, in DNA sequencing, the finite-valued nucleotides (A,C,G,T) are typically sequenced through observing the continuous-valued light intensities, also known as *flowgrams*. Other examples can be found in digital communication, in which the finite-valued codewords are modulated, e.g., QAM, and sent via a Gaussian channel, as well as in speech recognition, in which the finite-valued phonemes are observed as continuous-valued speech waveforms. In all of above examples, the goal of denoising is to recover the underlying finite-valued clean input source from the continuous-valued noisy observations.

There are two standard approaches for tackling above problem: supervised learning and Bayesian learning approaches. The supervised learning collects many clean-noisy paired data and learn a parametric model, e.g., neural networks, that maps noisy to clean data. While simple and straightforward, applying supervised learning often becomes challenging for the applications in which collecting underlying clean data is unrealistic. For such *unsupervised* setting, a common practice is to apply the Bayesian learning framework. That is, assume the existence of stochastic models on the source and noisy channel, then pursue the optimum estimation with respect to the learned joint distribution. Such approach makes sense for the case in which precisely modeling or designing the clean source is possible, *e.g.*, in digital communication, but limitations can also arise when the assumed stochastic model fails to accurately reflect the real data distribution.

Proceedings of the 23rdInternational Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108.

As a third alternative, the so-called *universal* approach has been proposed in (Weissman et al., 2005; Dembo and Weissman, 2005). Namely, while remaining in the unsupervised setting as in the Bayesian learning, the approach makes *no* assumption on the source stochastic model and instead applies the competitive analysis framework; namely, it focuses on the class of *sliding window denoisers* and aims to asymptotically achieve the performance of the best sliding window denoiser for *all* possible sources, solely based on the knowledge on the noisy channel model. The pioneering work, (Weissman et al., 2005), devised Discrete Universal DEnoiser (DUDE) algorithm, which handled the finite-input, finite-output (FIFO) setting, and (Dembo and Weissman, 2005) extended it to the case of finite-input, general-output (FIGO) channels, the setting on which this paper focuses.

While above both universal schemes enjoyed strong theoretical performance guarantees, they both had critical algorithmic limitations. Namely, the original DUDE becomes very sensitive to the selection of a hyperparameter, i.e., the window size k, and the generalized scheme for FIGO channel additionally suffered from the prohibitive computational complexity. Recently, (Moon et al., 2016; Ryu and Kim, 2018) employed neural networks in place of a counting vector used in DUDE and showed their schemes can significantly improve the denoising performance and robustness of DUDE. In this paper, we aim to extend the generalized scheme of (Dembo and Weissman, 2005) for the FIGO channel toward the direction of (Moon et al., 2016; Ryu and Kim, 2018), i.e., utilize neural networks to achieve much faster and better performance. Such extension is not straightforward, as we argue in the later sections, due to the critical difference that the channel has the continuous-valued outputs.

Our contribution is threefold:

- Algorithmic: We develop a new neural network-based denoising algorithm, dubbed as Generalized CUDE (Gen-CUDE), which can run orders of magnitude faster than the previous state-of-the-art in (Dembo and Weissman, 2005).
- Theoretical: We give a rigorous theoretical analyses on the performance of our method and obtain a much tighter upper bound on the average loss compared to that of (Dembo and Weissman, 2005).
- Experimental: We compare our algorithm on denoising both the simulated and real source data and show the superb performance compared to other strong baselines.

2 Notations and Problem Setting

We follow (Dembo and Weissman, 2005) but give more succinct notations. Throughout this paper, we will generally denote a sequence (n-tuple) as, e.g., $a^n = (a_1, \ldots, a_n)$, and a_i^j referes to the subsequence $(a_i, \ldots, a_j)$. We denote the clean, underlying source data as x^n and assume each component x_i takes a value in some finite set $\mathcal{A} = \{0, \ldots, M-1\}$. The lowercase letters are used to denote the *individual* sequences or the realization of a random sequence. We assume the noisy channel, denoted as $\mathcal{C}$, is *memoryless* and is given by the set $\{f_a\}_{a\in\mathcal{A}}$, in which f_a denoting the density with respect to the Lebesgue measure[1] associated with the channel output distribution for an input symbol a. Following states the mild assumption that we make on $\{f_a\}_{a\in\mathcal{A}}$ throughout the paper.

Assumption 1 *The set of densities $\{f_a\}_{a\in\mathcal{A}}$ is a set of linearly independent functions in $L_1(\mu)$.*

Given above channel $\mathcal{C}$, the noise-corrupted version of the source sequence x^n is denoted as $Y^n = (Y_1, \ldots, Y_n)$. Note we used the uppercase letter to emphasize the randomness in the noisy observation. Now, consider a measurable quantizer $Q : \mathbb{R} \to \mathcal{A}$, which quantizes the channel output to symbols in $\mathcal{A}$, and the induced channel $\mathbf{\Pi}$, a $M \times M$ channel transition matrix induced by Q and $\{f_a\}$. We denote the quantized output of Y^n by Z^n, and the (x, z)-th element of $\mathbf{\Pi}$ can be computed as

$$\mathbf{\Pi}(x, z) = \int_{y:Q(y)=z} f_x(y)dy. \tag{1}$$

Note Assumption 1 ensures that $\mathbf{\Pi}$ is an invertible matrix. Moreover, we denote $Z_i \triangleq Q(Y_i)$ as the quantized version of Y_i.

Given the entire *continuous-valued* noisy observation Y^n, the denoiser reconstructs the original *discrete* input x^n with $\hat{X}^n = (\hat{X}_1(Y^n), \ldots, \hat{X}_n(Y^n))$, where each reconstructed symbol $\hat{X}_i(Y^n)$ takes its value in $\mathcal{A}$. The fidelity of the denoising is measured by the average loss

$$L(x^n, \hat{X}^n(Y^n)) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{\Lambda}\Big(x_i, \hat{X}_i(Y^n)\Big), \tag{2}$$

in which $\mathbf{\Lambda} \in \mathbb{R}^{M\times M}$ is a per-symbol bounded loss matrix. Moreover, we denote $\mathbf{\Lambda}_{\hat{x}}$ as the $\hat{x}$-th column of $\mathbf{\Lambda}$ and $\Lambda_{\max} = \max_{x,\hat{x}} \mathbf{\Lambda}(x, \hat{x})$.

Then, for a probability vector $\mathbf{P} \in \Delta^M$, the *Bayes envelope*, $U(\mathbf{P})$, is defined to be

$$U(\mathbf{P}) \triangleq \min_{\hat{x}\in\mathcal{A}} \sum_{x\in\mathcal{A}} \mathbf{\Lambda}(x, \hat{x})\mathbf{P}(x), \tag{3}$$

[1] We assume such density always exists for concreteness.

which, in words, stands for the minimum achievable expected loss in estimating the source symbol that is distributed according to $\mathbf{P}$. The argument that achieves (3) is denoted by $\mathcal{B}(\mathbf{P})$, the *Bayes response* with respect to $\mathbf{P}$. Furthermore, in the later sections, we extend the notion of Bayes response by using $\mathbf{P}$ that is not necessarily a probability vector.

The k-th order sliding-window denoisers are the denoisers that are defined by the time-invariant mappings $g_k : \mathbb{R}^{2k+1} \to \mathcal{A}$. That is, $\hat{X}_i(Y^n) = g_k(Y_{i-k}^{i+k})$. We also denote the tuple $\mathbf{Y}_{-i}^{(k)} \triangleq (Y_{i-k}^{i-1}, Y_{i+1}^{i+k})$ as the k-th order context around the noisy symbol Y_i.

3 Related Work

3.1 DUDE, Neural DUDE and CUDE

A straightforward baseline for the FIGO channel setting is to simply quantize the continuous-valued output and apply the discrete denoising algorithm to estimate the underlying clean source. While such scheme is clearly suboptimal since it significantly discards the information observed in Y^n, we briefly review the previous work on discrete denoising so that we can build intuitions for devising our algorithm for the FIGO channel.

DUDE was devised by (Weissman et al., 2005) and is a two-pass, sliding-window denoiser for the FIFO setting. In discrete denoising, we denote Z^n as the finite-valued noisy sequence, $\mathbf{Z}_{-i}^{(k)} \triangleq (Z_{i-k}^{i-1}, Z_{i+1}^{i+k})$ as the k-th order context around Z_i, and $\mathbf{\Gamma}$ as the Discrete Memoryless Channel (DMC) transition matrix that induces the noisy sequence Z^n from the clean x^n. Then, the reconstruction of DUDE at location i is defined to be

$$\hat{X}_i(\mathbf{Z}_{-i}^{(k)}, Z_i) = \arg\min_{\hat{x}\in\hat{\mathcal{X}}} \hat{\mathbf{p}}_{\text{emp}}(\cdot|\mathbf{Z}_{-i}^{(k)})^\top \mathbf{\Gamma}^\dagger [\mathbf{\Lambda}_{\hat{x}} \odot \boldsymbol{\gamma}_{Z_i}], \quad (4)$$

in which $\mathbf{\Gamma}^\dagger$ is a Moore-Penrose pseudo-inverse of $\mathbf{\Gamma}$ (assuming $\mathbf{\Gamma}$ is full row-rank), $\boldsymbol{\gamma}_z$ is the z-th column of $\mathbf{\Gamma}$, and $\hat{\mathbf{p}}_{\text{emp}}(\cdot|\mathbf{Z}_{-i}^{(k)}) \in \mathbb{R}^{|\mathcal{Z}|}$ is an empirical probability vector on Z_i given the context $\mathbf{Z}_{-i}^{(k)}$, obtained from the entire noisy sequence Z^n. That is, for a k-th order double-sided context $\mathbf{Z}^{(k)}$, the z-th element of $\hat{\mathbf{p}}_{\text{emp}}(\cdot|\mathbf{Z}_{-i}^{(k)})$ becomes

$$\hat{\mathbf{p}}_{\text{emp}}(z|\mathbf{Z}^{(k)}) = \frac{|\{j : \mathbf{Z}_{-j}^{(k)} = \mathbf{Z}^{(k)}, Z_j = z\}|}{|\{j : \mathbf{Z}_{-j}^{(k)} = \mathbf{Z}^{(k)}\}|}. \quad (5)$$

The main intuition for obtaining (4) is to show that the true posterior distribution can be approximated by using (5) and inverting the DMC channel, $\mathbf{\Gamma}$. That is, the following approximation

$$p(x_i|Z_{i-k}^{i+k}) \approx \left(\boldsymbol{\gamma}_{Z_i} \odot [\mathbf{\Gamma}^{\dagger\top} \hat{\mathbf{p}}_{\text{emp}}(\cdot|\mathbf{Z}_{-i}^{(k)})]\right)_{x_i} \quad (6)$$

holds with high probability with large n (Weissman et al., 2005, Section IV.B). Then, for each location i, (4) is the $\mathcal{B}(\boldsymbol{\gamma}_{Z_i} \odot [\mathbf{\Gamma}^{\dagger\top} \hat{\mathbf{p}}_{\text{emp}}(\cdot|\mathbf{Z}_{-i}^{(k)})])$, the Bayes response with respect to the right-hand side of (6). (Weissman et al., 2005) showed the DUDE rule, (4), can universally attain the denoising performance of the best k-th order sliding window denoiser for *any* x^n.

Neural DUDE (N-DUDE) was recently proposed by (Moon et al., 2016), and it identified that the limitation of DUDE follows from the empirical count step in (5). Namely, the count happens totally separately for each context $\mathbf{C}$, even if the contexts can be very similar to each other. To that end, N-DUDE implements a *single* neural network-based sliding-window denoiser such that the information among similar contexts can be shared through the network parameters. That is, N-DUDE defines $\mathbf{p}_{\texttt{N-DUDE}}^k(\mathbf{w}, \cdot) : \mathcal{Z}^{2k} \to \Delta^{|\mathcal{S}|}$, in which $\mathbf{w}$ stands for the parameters in the network, and $\mathcal{S}$ is a set of single-symbol denoisers, $s : \mathcal{Z} \to \mathcal{A}$, which map $\mathcal{Z}$ to $\mathcal{A}$. Thus, $\mathbf{p}_{\texttt{N-DUDE}}^k(\mathbf{w}, \cdot)$ takes the context $\mathbf{Z}_{-i}^{(k)}$ and outputs a probability distribution on the single-symbol denoisers to apply to Z_i, for each i. Note for discrete denoising, $|\mathcal{S}|$ has to be finite, hence the network has the structure of a multi-class classification network.

To train the network parameters $\mathbf{w}$, N-DUDE defines the objective function

$$\mathcal{L}(\mathbf{w}, Z^n) \triangleq \frac{1}{n} \sum_{i=1}^{n} \mathbb{C}.\mathbb{E}\Big(\mathbf{L}_{\text{new}}^\top \mathbb{1}_{Z_i}, \mathbf{p}_{\texttt{N-DUDE}}^k(\mathbf{w}, \mathbf{C}_i)\Big),$$

in which $\mathbb{C}.\mathbb{E}(\mathbf{g}, \mathbf{p})$ stands for the (unnormalized) cross-entropy, and $\mathbf{L}_{\text{new}}^\top \mathbb{1}_{Z_i}$ is the *pseudo-label* vector for the i-th location, calculated from the unbiased estimate of the true expected loss which can be computed with $\mathbf{\Lambda}$, $\mathbf{\Gamma}$, and Z^n (more details are in (Moon et al., 2016)). Note the dependency of the objective function on Z^n is highlighted, hence, the training of $\mathbf{w}$ is done in an unsupervised manner together with the knowledge of the channel.

Once the objective function is minimized via stochastic gradient descent, the converged parameter is denoted as $\mathbf{w}^\star$. Then, the single-letter mapping defined by N-DUDE for the context $\mathbf{Z}_{-i}^{(k)}$ is expressed as $s_{k,\texttt{N-DUDE}}(\mathbf{Z}_{-i}^{(k)}, \cdot) = \arg\max_{s\in\mathcal{S}} \mathbf{p}_{\texttt{N-DUDE}}^k(\mathbf{w}^\star, \mathbf{Z}_{-i}^{(k)})_s$, and the reconstruction at location i becomes

$$\hat{X}_{i,\texttt{N-DUDE}}(\mathbf{Z}_{-i}^{(k)}, Z_i) = s_{k,\texttt{N-DUDE}}(\mathbf{Z}_{-i}^{(k)}, Z_i). \quad (7)$$

(Moon et al., 2016) shows N-DUDE significantly outperforms DUDE and is more robust with respect to k.

CUDE was proposed by (Ryu and Kim, 2018) following-up on N-DUDE, which took an alternative and simpler approach of using neural network

to extend DUDE. Namely, instead of using the empirical distribution in (5), CUDE learns a network $\mathbf{p}^k_{\texttt{CUDE}}(\mathbf{w},\cdot):\mathcal{Z}^{2k}\to\Delta^{|\mathcal{Z}|}$, which takes the context $\mathbf{Z}^{(k)}_{-i}$ as input and outputs a prediction for Z_i, by minimizing $\frac{1}{n}\sum_{i=1}^n \mathbb{C}.\mathbb{E}(\mathbb{1}_{Z_i},\mathbf{p}^k_{\texttt{CUDE}}(\mathbf{w},\mathbf{Z}^{(k)}_{-i}))$. Thus, the network aims to directly learn the conditional distribution of Z_i given its context $\mathbf{Z}^{(k)}_{-i}$. Once the minimizer $\mathbf{w}^*$ is obtained, CUDE then simply plugs in $\mathbf{p}^k_{\texttt{CUDE}}(\mathbf{w}^*,\mathbf{Z}^{(k)}_{-i})$ in place of $\hat{\mathbf{p}}_{\text{emp}}(\cdot|\mathbf{Z}^{(k)}_{-i})$ in (4). (Ryu and Kim, 2018) shows that CUDE outperforms N-DUDE primarily due to the reduced output size of the neural network, *i.e.*, $|\mathcal{Z}|$ vs. $|\mathcal{S}|=|\mathcal{A}|^{|\mathcal{Z}|}$.

3.2 Generalized DUDE for FIGO channel

(Dembo and Weissman, 2005) extended DUDE algorithm specifically for the FIGO channel case, and we refer to their scheme as Generalized DUDE (Gen-DUDE) from now on. The key challenge arises in the FIGO channel for applying the DUDE framework is that it becomes impossible to obtain an empirical distribution like (5) based on counting for each context, because there are infinitely many possible contexts.

Therefore, by denoting $\mathbf{P}(X_i|y^{i+k}_{i-k})\in\Delta^M$ as the conditional probability vector on X_i given the $(2k+1)$-tuple y^{i+k}_{i-k}, Gen-DUDE first identifies that the denoising rule at location i should be the Bayes response

$$\hat{X}_i(y^n)=\mathcal{B}(\mathbf{P}(X_i\mid y^{i+k}_{i-k}))=\ \mathcal{B}(\mathbf{P}(X_i,y^{i+k}_{i-k})),\quad(8)$$

in which the second equality in (8) follows from ignoring the normalization factor of $\mathbf{P}(X_i|y^{i+k}_{i-k})\in\Delta^M$. Note, $\mathbf{P}(X_i,y^{i+k}_{i-k})$ is not a probability vector, but the notion of Bayes response still holds. Now, the joint distribution can be expanded as

$$\begin{aligned}p(X_i=a,y^{i+k}_{i-k})&=\sum_{u^k_{-k}:u_0=a}p(X^{i+k}_{i-k}=u^k_{-k},y^{i+k}_{i-k})\\&=\sum_{u^k_{-k}:u_0=a}\underbrace{\Big[\prod_{j=-k}^{k}f_{u_j}(y_{i+j})\Big]}_{(a)}\underbrace{p(X^{i+k}_{i-k}=u^k_{-k})}_{(b)},\end{aligned}\quad(9)$$

in which term (a) of (9) follows from the memoryless assumption on the channel $\mathcal{C}$. Then, Gen-DUDE approximates term (b) of (9), which is now the distribution on the finite-valued source $(2k+1)$-tuples, by computing the empirical distribution of the *quantized* noisy sequence Z^n and inverting the induced DMC matrix $\mathbf{\Pi}$, both of which are defined in Section 2. Once the approximation for (9) is done, the Gen-DUDE simply computes the Bayes response as in (8) with the approximate joint probability vector. For more details, we refer to the paper (Dembo and Weissman, 2005).

The main critical drawback of Gen-DUDE is the computation required for approximating (9). Namely, the summation in (9) is over all possible $(2k+1)$-tuples of the source symbols, of which complexity grows exponentially with k. Therefore, the running time of the algorithm becomes totally impractical even for the modest alphabet sizes, e.g., 4 or 10, as shown in our experimental results in the later section. Moreover, such exponential dependency on k also appears in the theoretical analyses of Gen-DUDE. That is, it is shown that the upper bound on the probability that the average loss of Gen-DUDE deviates from that of the best sliding-window denoiser is proportional to the doubly exponential term $C^{M^{2k+1}}$, which quickly becomes meaningless for, again, modest size of M and k. Motivated by such limiations, we introduce neural networks to efficiently approximate $\mathbf{P}_{X_i,y^{i+k}_{i-k}}$ and compute the Bayes response to significantly improve the Gen-DUDE method.

4 Main Results

4.1 Intuition for Gen-CUDE

As mentioned above, the Gen-DUDE suffers from high computational complexity due to the expansion given in (9) that requires the summation over the exponentially many (in k) terms. The main reason for such expansion in (Dembo and Weissman, 2005) was to utilize the tools of DUDE for approximating term (b) in (9), which inevitably requires to enumerate all the $2k$-tuple terms. Hence, we instead try to directly approximate $\mathbf{P}(X_i|y^{i+k}_{i-k})$ using a neural network.

Our algorithm is inspired by N-DUDE and CUDE, mentioned in Section 3.1, which show much better traits compared to the original DUDE. However, we easily notice that the approach of N-DUDE cannot be applied to the FIGO channel case, because there will be infinitely many single-symbol denoisers $s:\mathcal{Y}\to\mathcal{A}$. Hence, the output layer of the network should perform some sort of regression, instead of the classification as in N-DUDE, but obtaining the pseudo-label for training in that case is far from being straightforward. Therefore, we take an inspiration from CUDE and develop our Generalized CUDE (Gen-CUDE).

In order to build the core intuition for our method, first consider the quantized noisy sequence Z^n and the induced DMC matrix $\mathbf{\Pi}$ (defined in (1)). That is, $Z_i=Q(Y_i)$ where $Q(\cdot)$ is the quantizer introduced in Section 2. Furthermore, denote $\mathbf{P}(X_0|y^k_{-k})\in\Delta^M$ and $\mathbf{P}(Z_0|y^k_{-k})\in\Delta^M$ as the conditional probability vectors of X_0 and Z_0 given a $(2k+1)$ tuple y^k_{-k} that appear in the noisy observation Y^n. Also, let $\boldsymbol{f}_{X_0}(y_0)\in\mathbb{R}^M$ be the vector of density values of which a-th element

is $f_a(y_0)$. We treat all the vectors as *row* vectors. The following lemma builds the key motivation.

Lemma 1 *Given y_{-k}^k, the following holds.*

$$\mathbf{P}(X_0|y_{-k}^k) \propto [\mathbf{P}(Z_0|\mathbf{y}_{-0}^{(k)}) \cdot \mathbf{\Pi}^{-1}] \odot \boldsymbol{f}_{X_0}(y_0) \tag{10}$$

Namely, we can compute $\mathbf{P}(X_0|y_{-k}^k)$ up to a normalization constant using the conditional distribution on Z_0, and the information on the channel $\mathcal{C}$.

Proof: We have the following chain of equalities for the conditional distribution $p(x_0|y_{-k}^k)$:

$$p(x_0|y_{-k}^k) = \frac{p(x_0, y_{-k}^k)}{p(y_{-k}^k)} = \frac{p(x_0, \mathbf{y}_{-0}^{(k)}) f_{x_0}(y_0)}{p(y_{-k}^k)} \tag{11}$$

$$= p(x_0|\mathbf{y}_{-0}^{(k)}) f_{x_0}(y_0) \cdot \frac{p(\mathbf{y}_{-0}^{(k)})}{p(y_{-k}^k)}, \tag{12}$$

in which the second equality of (11) follows from the memoryless property of the densities $\{f_a\}_{a\in\mathcal{A}}$ of $\mathcal{C}$, and $\mathbf{y}_{-0}^{(k)}$ stands for the double-sided context (y_{-k}^{-1}, y_1^k). Now, by denoting $z_0 = Q(y_0)$ as the quantized version of y_0, we have the following relation.

$$p(z_0|\mathbf{y}_{-0}^{(k)}) = \sum_{x_0} p(z_0|x_0, \mathbf{y}_{-0}^{(k)}) p(x_0|\mathbf{y}_{-0}^{(k)})$$

$$= \sum_{x_0} \mathbf{\Pi}(x_0, z_0) p(x_0|\mathbf{y}_{-0}^{(k)}), \tag{13}$$

in which (13) follows from the channel $\mathcal{C}$ being memoryless and utilizing the notation of the induced DMC matrix, $\mathbf{\Pi}$, defined in (13). Thus, following the row vector notations as mentioned above, we have

$$\mathbf{P}(Z_0|\mathbf{y}_{-0}^{(k)}) = \mathbf{P}(X_0|\mathbf{y}_{-0}^{(k)}) \cdot \mathbf{\Pi}. \tag{14}$$

By inverting $\mathbf{\Pi}$ in (14), and combining with (12) and dropping the term $\frac{p(\mathbf{y}_{-0}^{(k)})}{p(y_{-k}^k)}$, we have the lemma. ■

From the lemma, we can see that once we have accurate approximation of the conditional distribution $\mathbf{P}(Z_0|\mathbf{y}_{-0}^{(k)})$, then we can apply (10) and obtain the Bayes response with respect to $[\mathbf{P}(Z_0|\mathbf{y}_{-0}^{(k)}) \cdot \mathbf{\Pi}^{-1}] \odot \boldsymbol{f}_{X_0}(y_0)$. Now, following the spirit of CUDE, we utilize neural network to approximate $\mathbf{P}(Z_0|\mathbf{y}_{-0}^{(k)})$ from the observed data. We concretely describe our Gen-CUDE algorithm in the next subsection.

4.2 Algorithm Description

Inspired by (10) and CUDE, we try to use a single neural network to learn the k-th order sliding window denoiser. First of all, define $\mathbf{p}^k(\mathbf{w}, \cdot) : \mathbb{R}^{2k} \to \Delta^M$ as a feed-forward neural network we utilize. With weight parameter $\mathbf{w}$, the network takes context $\mathbf{y}_{-i}^{(k)}$ as input and send out $\mathbf{P}(Z_0|\mathbf{y}_{-i}^{(k)})$ as output. To learn the parameter $\mathbf{w}$, we define the objective function as

$$\mathcal{L}_{\texttt{Gen-CUDE}}(\mathbf{w}, Y^n) \triangleq \frac{1}{n-2k} \sum_{i=k}^{n-k} \mathbb{C}.\mathbb{E}\Big(\mathbb{1}_{Z_i}, \mathbf{p}^k(\mathbf{w}, \mathbf{Y}_{-i}^{(k)})\Big).$$

Namely, minimizing $\mathcal{L}_{\texttt{Gen-CUDE}}$ leads to training the network to predict the *quantized* middle symbol Z_i based on the continuous-valued context $\mathbf{Y}_{-i}^{(k)}$, hence, the network can maintain the multi-class classification structure with the ordinary softmax output layer. The minimization is done by the stochastic gradient descent-based optimization methods such as Adam (Kingma and Ba, 2014). Once the minimization is done, we denote the converged weight vector as $\mathbf{w}^*$. Then, by motivated by Lemma 1, we define our `Gen-CUDE` denoiser as the Bayes response with respect to $[\mathbf{p}^k(\mathbf{w}^*, \mathbf{Y}_{-i}^{(k)}) \cdot \mathbf{\Pi}^{-1}] \odot \mathbf{f}_{X_0}(Y_i)$ for each i. Following summarizes our algorithm.

Algorithm 1 Gen-CUDE algorithm

Input: Noisy sequence Y^n, Context size k, $\mathcal{C} = \{f_a\}_{a\in\mathcal{A}}$, $\mathbf{\Lambda}$, Quantizer $Q(\cdot)$
Output: Denoised sequence $\hat{X}_{\mathrm{NN}}^n = \{\hat{X}_{i,\mathrm{NN}}(Y^n)\}_{i=1}^n$

Obtain the quantized sequence Z^n using $Q(\cdot)$
Compute $\mathbf{\Pi}$ as (1) and initialize $\mathbf{p}^k(\mathbf{w}, \cdot)$
Obtain $\mathbf{w}^*$ minimizing $\mathcal{L}_{\texttt{Gen-CUDE}}(\mathbf{w}, Y^n)$
if $i = k+1, \ldots, n-k$ **then**
  Compute $[\mathbf{p}^k(\mathbf{w}^*, \mathbf{Y}_{-i}^{(k)}) \cdot \mathbf{\Pi}^{-1}] \odot \boldsymbol{f}_{X_i}(Y_i)$
  $\hat{X}_{i,\mathrm{NN}}(y^n) = \mathcal{B}\big([\mathbf{p}^k(\mathbf{w}^*, \mathbf{Y}_{-i}^{(k)}) \cdot \mathbf{\Pi}^{-1}] \odot \boldsymbol{f}_{X_i}(Y_i)\big)$
else
  $\hat{X}_{i,\mathrm{NN}}(Y^n) = Z_i$
end if
Obtain $\hat{X}_{\mathrm{NN}}^n(Y^n) = \{\hat{X}_{i,\mathrm{NN}}(Y^n)\}_{i=1}^n$

4.3 Theoretical Analysis

In this subsection, we give a theoretical analysis on Gen-CUDE, which follows similar steps as in (Dembo and Weissman, 2005) but derives a much tighter upper bound on the average loss of Gen-CUDE. As a performance target for the competitive analysis, we define the minimum expected loss of x^n for the kth-order sliding-window denoiser by

$$D_{x^n}^k = \min_{g_k} \mathbb{E}\Big[\frac{1}{n-2k} \sum_{i=k+1}^{n-k} \mathbf{\Lambda}\Big(x_i, g_k\Big(Y_{i-k}^{i+k}\Big)\Big)\Big]. \tag{15}$$

Now, we introduce a regularity assumption to carry out analysis for the performance bound.

Assumption 2 *Consider the network parameter $\mathbf{w}^*$ learned by minimizing $\mathcal{L}_{\texttt{Gen-CUDE}}$. Then, we assume*

there exists a sufficiently small $\epsilon' > 0$ *such that*

$$\left\| \mathbf{P}(Z_0|\mathbf{y}_{-0}^{(k)}) - \mathbf{p}^k(\mathbf{w}^*, \mathbf{y}_{-0}^{(k)}) \right\|_1 \leq \epsilon'$$

holds for all contexts $\mathbf{y}_{-0}^{(k)} \in \mathbb{R}^{2k}$.

Assumption 2 is based on the universal approximation theorem (Cybenko, 1989; Hornik et al., 1989), which ensures that there always exists a neural network that can approximate any function with arbitrary accuracy. Thus, we assume that the neural network learned by minimizing $\mathcal{L}_{\texttt{Gen-CUDE}}$ results in an accurate enough approximation of the true probability vector $\mathbf{P}(Z_0|\mathbf{y}_{-0}^{(k)})$.

Now, by letting

$$\hat{\mathbf{P}}(X_0|y_{-k}^k) \triangleq \frac{p(\mathbf{y}_{-0}^{(k)})}{p(y_{-k}^k)} [\mathbf{p}^k(\mathbf{w}^*, \mathbf{y}_{-0}^{(k)}) \mathbf{\Pi}^{-1}] \odot \mathbf{f}_{X_0}(y_0),$$

we can then show from Assumption 2 that

$$\mathbb{E} \| \mathbf{P}(X_0|Y_{-k}^k) - \hat{\mathbf{P}}(X_0|Y_{-k}^k) \|_1 \leq \epsilon^*, \tag{16}$$

for $\epsilon^* = \epsilon' \sum_{a=0}^{M-1} \|\pi_a^{-1}\|_2$, in which $\mathbb{E}(\cdot)$ is the expectation with respect to Y_{-k}^k, and π_a^{-1} stands for the a-th column of $\mathbf{\Pi}^{-1}$. The proof of (16) is given Lemma 2 in the Supplementary Material, and it plays an important role in proving the main theorem.

Before stating the main theorem, we first introduce $\mathcal{R}_\delta$, which is a quantizer that rounds each component of a probability vector to the nearest integer multiple of $\delta > 0$ in $[0, 1]$. Then, consider a denoiser $\hat{X}_{NN}^{n,\delta}(Y^n)$ of which i-th component ($k \leq i \leq n-k$) is defined as $\hat{X}_{i,\text{NN}}^{\delta}(Y^n) = \mathcal{B}(\hat{\mathbf{P}}^\delta(X_i|Y_{i-k}^{i+k}))$, where $\hat{\mathbf{P}}^\delta(X_i|Y_{i-k}^{i+k}) = \mathcal{R}_\delta(\hat{\mathbf{P}}(X_i|Y_{i-k}^{i+k}))$. Note when δ is small enough, the performance of $\hat{X}_{\text{NN}}^{n,\delta}(Y^n)$ would be close to that of Gen-CUDE. Now, we have the following theorem.

Theorem 1 *Consider* ϵ^* *in (16). Then, for all* $k, n \geq 1$, $\delta > 0$, *and* $\epsilon > \Lambda_{\max} \cdot (3\epsilon^* + \frac{M \cdot \delta}{2})$, *and for all* x^n,

$$\Pr\left(|L_{\hat{X}_{NN}^{n,\delta}}(x^n, Y^n) - D_{x^n}^k| > \epsilon \right) \leq C_1(k, \delta, M) \exp\left(- \frac{2(n-2k)}{(2k+1)} C_2(\epsilon, \epsilon^*, \Lambda_{\max}, M, \delta) \right),$$

in which $C_1(k, \delta, M) \triangleq 2(2k+1)[\frac{1}{\delta} + 1]^M$ *and* $C_2(\epsilon, \epsilon^*, \Lambda_{\max}, M, \delta) \triangleq (\epsilon - \Lambda_{\max} \cdot (3\epsilon^* + \frac{M \cdot \delta}{2}))^2 \cdot \frac{1}{\Lambda_{\max}^2}$.

Proof: The full proof of the theorem as well as necessary lemmas are given in the Supplementary Material.

Theorem 1 states that for any x^n, with high probability, Gen-CUDE can essentially achieve the performance of the best sliding-window denoiser with the same order k. Note that our bound has the constant term $[\frac{1}{\delta}+1]^M$, whereas the paralleling result in (Dembo and Weissman, 2005) has $[\frac{1}{\delta}+1]^{M^{2k+1}}$. Removing such doubly exponential dependency on k in our result is mainly due to our directly modeling the marginal posterior distribution via neural network, as opposed to the modeling of the joint posterior of the $(2k+1)$-tuple in the previous work. This improvement carries over to the better empirical performance of the algorithm given in the next section.

5 Experimental Results

5.1 Setting and baselines

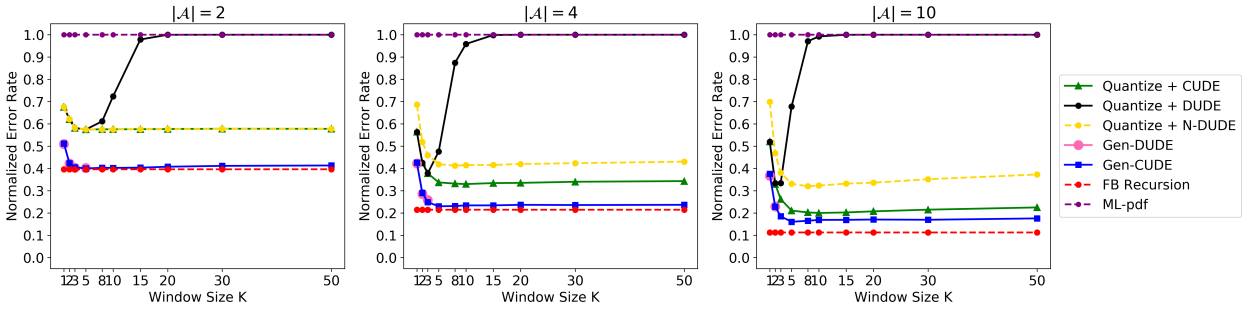
We have experimented with both synthetic and real DNA source data and verified the effectiveness of our proposed Gen-CUDE algorithm. The noisy channel $\mathcal{C} = \{f_a\}_{a \in \mathcal{A}}$ was assumed to be known, and the noisy observation Y^n was generated by corrupting the source sequence x^n. We used the Hamming loss as our $\mathbf{\Lambda}$ to measure the denoising performance.

We have compared the performance of `Gen-CUDE` with several baselines. The simplest baseline is `ML-pdf`, which carries out the symbol-by-symbol maximum likelihood estimate, i.e., $\hat{X}_i(Y^n) = \arg\max_a f_a(Y_i)$. The other baselines are schemes that apply discrete denoising algorithms on the quantized Z^n using the induced DMC $\mathbf{\Pi}$. That is, these schemes simply throw away the continuous-valued observation Y^n and the density values. We denoted such schemes as `Quantized+DUDE`, `Quantized+N-DUDE`, and `Quantized+CUDE`. We also employed `Gen-DUDE` as a baseline for FIGO channel. For neural network training, we used a fully-connected network with ReLU (Nair and Hinton, 2010) activations . For more details on the implementation, the code is available online[2].

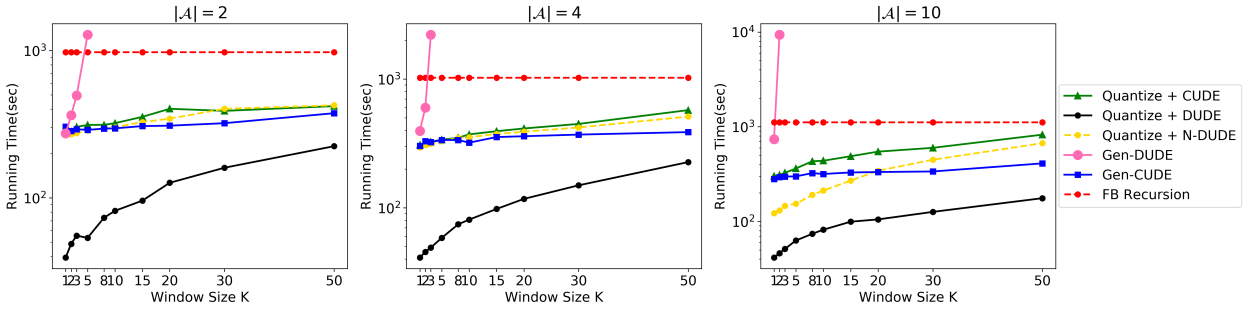
5.2 Synthetic source with Gaussian noise

For the synthetic source data case, we generated the clean sequence x^n from a symmetric Markov chain. We varied the alphabet size $|\mathcal{A}| = 2, 4, 10$, and the source symbol was encoded to have odd integer values $\mathcal{O} = \{\pm(2\ell - 1) : 1 \leq \ell \leq |\mathcal{A}|/2\}$. The transition probability of the Markov source was set to 0.9 for staying on the same state and $0.1/|\mathcal{A}|$ for transitioning to the other state. The sequence length was $n = 3 \times 10^6$, and the noisy channel was set to be the standard additive white Gaussian, $\mathcal{N}(0, 1)$. The neural network had 6 fully-connected layers and 200 nodes in each layer. For the quantizer $Q(\cdot)$ in all of our experiments, we simply rounded to the nearest integer among $\mathcal{O}$. Note

[2] `https://github.com/pte1236/Gen-CUDE`



(a) Denoising Performace



(b) Running Time

Figure 1: Synthetic source data case. (a) Denoising results and (b) running time (including training time). The left, center, and right plots correspond to the case of $|\mathcal{A}| = 2$, 4, and 10, respectively.

that $Q(\cdot)$ can be freely selected for `Gen-CUDE` as long as the induced DMC, $\mathbf{\Pi}$, is invertible, and we show the little effect of the choice of $Q(\cdot)$ on the denoising performance in the Supplementary Material.

The denoising performance as well as the running time of each scheme is given in Figure 1, and the performance in Figure 1(a) was normalized with the performance of the simple quantizer, $\hat{X}_i = Z_i = Q(Y_i)$. Note for the symmetric Gaussian noise, `ML-pdf` becomes equivalent to applying $Q(\cdot)$, but they can become different for general noise densities. Moreover, we compared the performance with `FB-Recursion` (Ephraim and Merhav, 2002, Section V.), which is the optimal scheme for the given setting since the noisy sequence becomes a hidden Markov process (HMP).

From the figures, we can make several observations. Firstly, we note that the neural network-based schemes, i.e., `Quantize+N-DUDE`, `Quantize+CUDE`, and our `Gen-CUDE`, are very robust with respect to the window size k. The effect of the window size k for `Gen-CUDE` not being huge compared to `Gen-DUDE` can be predicted from the bound in Theorem 1. In contrast, `Quantize+DUDE` becomes quite sensitive to k as has been identified in (Moon et al., 2016). Secondly, we observe our `Gen-CUDE` always achieves the best denoising performance among the baselines and gets close to the optimal `FB-Recursion`. Note `Gen-CUDE` knows nothing about the source sequence x^n, whereas `FB-Recursion` exactly knows the source Markov model. Moreover, while `Gen-DUDE` performs almost as well as `Gen-CUDE` for $|\mathcal{A}| = 2$ with appropriate k, its performance significantly deteriorates when the alphabet size grows. We see that the gap between `Gen-CUDE` and `FB recursion` widens (although not much) as the alphabet size M increases, which can also be predicted from the bound in Theorem 1. Thirdly, `Gen-DUDE` suffers from the prohibitive computational complexity as k grows, as shown in Figure 1(b), while the running time of our `Gen-CUDE` is orders of magnitude faster than that of `Gen-DUDE` and more or less constant with respect to k. From this reason, `Gen-DUDE` can be run only for small k values. Fourthly, we note `Quantize+CUDE` also performs reasonably well, and it outperforms all discrete denoising baselines as also shown in (Ryu and Kim, 2018). However, since it discards the additional soft information in the continuous-valued observation and density values, `Gen-CUDE`, which is tailored for the FIGO channel, outperforms `Quantize+CUDE` with a significant gap.

5.3 DNA source with homopolymer errors

Now, we verify the performance of `Gen-CUDE` on real DNA sequencing data. We focus on the homopolymer errors which is the dominant error type in sequencing by synthesis methods, e.g., Roche 454 pyrosequencing

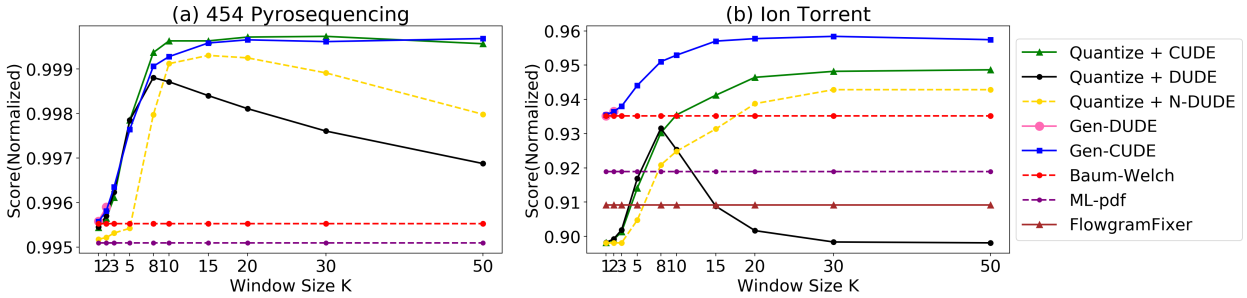


Figure 2: Normalized similarity scores against the clean reference DNA sequence for both sequencing platforms.

(Quince et al., 2011) or Ion Torrent Personal Genome Machine (PGM) (Bragg et al., 2013). In those methods, each nucleotide in turn is iteratively washed over with a pre-determined short sequence of bases known as the wash cycle, and the continuous-valued flowgrams are observed. Recently, (Lee et al., 2017) describes how we can interpret the base-calling procedure of such sequencers exactly as our FIGO channel setting by mapping the DNA sequence into sequence of integers (of homopolymer length), which becomes the input to the noisy channel $\{f_a\}_{a\in\mathcal{A}}$, the flowgram densities for each homopolymer length. This interpretation is possible since the order of the nucleotides in the wash cycle is fixed. Denoising in such setting can correct insertion and deletion errors, the dominant and notoriously hard types of errors in such sequencers.

We used `Artificial.dat`, a public dataset used in (Quince et al., 2011) for 454 pyrosequencing, and `IonCode0202.CE2R.raw.dat`, a data obtained from an internal source, for Ion Torrent after preprocessing both datasets. We *simulated* the channel of each platform to obtain the noisy sequence that is corrupted with the homopolymer errors. The used noisy channel density was provided for 454, but not for Ion Torrent, hence, we estimated the density for Ion Torrent with a small holdout set with clean source using Gaussian kernel density estimation with bandwidth 0.6. The used densities for 454 and Ion Torrent are shown in the Appendix D of the Supplementary Material. The total sequence length n for 454 and Ion Torrent data was 6,845,624 and 4,101,244, respectively. Moreover, the wash cycle of 454 and Ion Torrent was `TACG` and `TACGTACGTCTGAGCATCGATCGATGTACAGC`, respectively. We set the maximum homopolymer length to 9, hence, the source symbol can take values among $\{0,\dots,9\}$. The neural network for `Gen-CUDE` had 7 layers with 500 nodes in each layer. After denoising, the error correction performance was compared with the similarity score between the clean reference sequence and the denoised sequence (after converting back from the sequence of homopolymer lengths to the DNA sequence). The score was computed with the Pairwise2 module of Biopython, a common alignment tool to compute the similarity between DNA sequences (Chang et al., 2010).

Figure 2 shows the error correction performance for both 454 and Ion Torrent platform data. The score is normalized so that 1 corresponds to the perfect recovery. We also included `Baum-Welch` (Baum et al., 1970) that treats the source as a Markov source and estimates the transition probability before applying the `FB-recursion`. We can make the following observations. Firstly, we note that `Baum-Welch` is no longer optimum since the source DNA sequence is far from being a Markov. Secondly, `Quantize+DUDE`, which was used to correct the homopolymer errors in (Lee et al., 2017), turns out to be suboptimal, as also observed in Figure 1. Thirdly, `Gen-CUDE` again achieves the best error correction performance for both 454 and Ion Torrent data. Note for 454, in which the original error rate is quite small, the performance of `Quantize+CUDE` and `Gen-CUDE` becomes almost indistinguishable, but in Ion Torrent, of which noise density has a higher variance and non-zero means, the performance gap between the two widens. Fourthly, in line with Figure 1, we were not able to run `Gen-DUDE` for more than $k = 2$. Finally, for Ion Torrent, we also run `FlowgramFixer` (Golan and Medvedev, 2013), the state-of-the-art homopolymer error correction tool for Ion Torrent, but it showed the worst performance.

6 Conclusion

We devised a novel unsupervised neural network-based `Gen-CUDE` algorithm, which carries out universal denoising for FIGO channel. Our algorithm was shown to significantly outperform previously developed algorithm for the same setting, `Gen-DUDE`, both in denoising performance and computation complexity. We also give a rigorous theoretical analyses on the scheme and obtain a tighter upper bound on the average error compared to `Gen-DUDE`. Our experimental results show promising results, and as a future work, we plan to apply our method to *real* noisy data denoising and make more algorithmic improvements, e.g., using adaptive quantizers instead of simple rounding.

Acknowledgement

This work is supported in part by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) [No.2016-0-00563, Research on adaptive machine learning technology development for intelligent autonomous digital companion], [No.2019-0-00421, AI Graduate School Support Program (Sungkyunkwan University)], [No.2019-0-01396, Development of framework for analyzing, detecting, mitigating of bias in AI model and training data], and [IITP-2019-2018-0-01798, ITRC Support Program]. The authors also thank Seonwoo Min, Byunghan Lee and Sungroh Yoon for their helpful discussions on DNA sequence denoising, and thank Jae-Ho Shin for providing the raw Ion Torrent dataset.

References

Baum, L., Petrie, T., Soules, G., and Weiss, N. (1970). A maximization technique occuring in the statistical analysis of probabilistic functions of Markov chains. *Annals of Mathematical Statistics*, 41(164-171).

Bragg, L. M., Stone, G., Butler, M. K., Hugenholtz, P., and Tyson, G. W. (2013). Shining a light on dark sequencing: characterising errors in ion torrent pgm data. *PLoS computational biology*, 9(4):e1003031.

Chang, J., Chapman, B., Friedberg, I., Hamelryck, T., De Hoon, M., Cock, P., Antao, T., and Talevich, E. (2010). Biopython tutorial and cookbook. *Update*, pages 15–19.

Cybenko, G. (1989). Approximation by superposition of a sigmoidal function. *Math. Control Systems Signals*, 2(4):303–314.

Dabov, K., Foi, A., Katkovnik, V., and Egiazarian, K. (2007). Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Trans. Image Processing*, 16(8):2080–2095.

Dembo, A. and Weissman, T. (2005). Universal denoising for the finite-input general-output channel. *IEEE transactions on information theory*, 51(4):1507–1517.

Donoho, D. and Johnstone, I. (1995). Adapting to unknown smoothness via wavelet shrinkage. *Journal of American Statistical Association*, 90(432):1200–1224.

Elad, M. and Aharon, M. (2006). Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Trans. Image Processing*, 54(12):3736–3745.

Ephraim, Y. and Merhav, N. (2002). Hidden markov processes. *IEEE Trans. Inform. Theory*, 48(6):1518–1569.

Golan, D. and Medvedev, P. (2013). Using state machines to model the ion torrent sequencing process and to improve read error rates. *Bioinformatics*, 29(13):i344–i351.

Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2:359–366.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lee, B., Moon, T., Yoon, S., and Weissman, T. (2017). DUDE-Seq: Fast, flexible, and robust denoising for targeted amplicon sequencing. *PLoS ONE, 12(7):e0181463*.

Moon, T., Min, S., Lee, B., and Yoon, S. (2016). Neural universal discrete denoiser. In *Advances in Neural Information Processing Systems*, pages 4772–4780.

Moon, T. and Weissman, T. (2009). Discrete denoising with shifts. *IEEE Trans. Inform. Theory*, 55(11):5284–5301.

Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814.

Quince, C., Lanzen, A., Davenport, R. J., and Turnbaugh, P. J. (2011). Removing noise from pyrosequenced amplicons. *BMC bioinformatics*, 12(1):38.

Ryu, J. and Kim, Y.-H. (2018). Conditional distribution learning with neural networks and its application to universal image denoising. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 3214–3218. IEEE.

Weissman, T., Ordentlich, E., Seroussi, G., Verdu, S., and Weinberger, M. (2005). Universal discrete denoising: Known channel. *IEEE Trans. Inform. Theory*, 51(1):5–28.