ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 979-8-3503-4485-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICASSP48485.2024.10446730

# DATA-DRIVEN LATTICES FOR VECTOR QUANTIZATION

*Natalie Lang, Itamar Assaf, Omer Bokobza, and Nir Shlezinger*

## ABSTRACT

Lattice quantization implements vector quantization with a simple structured formulation, that is fully determined by the lattice generator matrix and a distance metric. The conventional approach constructs lattices for quantization by minimizing a bound on the rate-distortion tradeoff, which holds for non-overloaded quantizers, while in practice, overloading prevention typically affects performance. In this work we propose a novel technique for constructing lattice that considers possibly overloaded quantizers, for which we *learn* the lattice generator matrix by directly evaluating the distortion at its output. For training purposes, we convert the continuous-to-discrete quantizer mapping into a differentiable machine learning model, optimized in an unsupervised manner to best fit the data. Subsequently, the data-driven lattice is fixed and ordinarily combined into the quantization process. We provide numerical studies showing that our method attains improved performance compared with alternative lattice designs for various dimensions, and generalizes well to unseen data.

***Index Terms***— Lattice quantization, generator matrix.

## 1. INTRODUCTION

The quantization of continuous-valued signals into finite-bit representations plays a critical role in digital signal processing and compression [1]. Quantizers are typically required to achieve a desired tradeoff between the quantization rate and the distortion induced by discretization. This tradeoff can be improved by jointly quantizing multiple samples via *vector quantization* compared to each scalar sample separately [2].

Various approaches were proposed for designing quantizers. Generally speaking, quantization can be represented as a codebook of digital representations [3]. Such codebooks can be tuned from data using classic clustering-type methods [4], with more recent techniques employing deep learning [5]. In practice, one is often interested in utilizing structured mappings, and particularly uniform quantizers and their vector general form of lattice quantizers, as opposed to arbitrary codebooks [6]. Such structured quantizers are simple, support nested implementations [3], and, when combined with dithering [7, 8], eliminate dependence between the distortion and the signal in a universal manner [9], i.e., regardless of the input distribution. This property was exploited for compression in distributed machine learning systems [10–12].

The authors are with the School of ECE, Ben-Gurion University of the Negev, Be'er-Sheva, Israel (e-mails: {langn, itamaras, omerbok}@post.bgu.ac.il; nirshl@bgu.ac.il).

The continuous-to-discrete mapping of lattice quantizers is dictated by the *lattice generator matrix* and a distance metric [13]. A common approach to evaluate and design dithered lattice quantizers uses the ability to analytically bound its rate-distortion tradeoff when the quantizer is not overloaded. The resulting bound can be used to design the lattice, either analytically in some settings [14, 15] or via iterative optimization [16]. However, this approach assumes that the quantizer is not overloaded; what actually often requires truncating or scaling the input, which in turn induces excessive distortion. Consequently, the conventional lattice design objective may not faithfully reflect the performance of the resulting quantizer, motivating a design that accounts for overloading.

In this work we propose a data-driven method for designing lattice quantizers. Instead of considering the overloading-free rate-distortion bound, our approach optimizes the finite-but-possibly-overloaded lattice by directly evaluating the distortion at its output. Following model-based deep learning methodology [17, 18], we convert the lattice quantizer into a trainable discriminative model [19], tuning its generator matrix to best fit the data. Inspired by [20], our training method utilizes a dedicated fixed-input deep neural network (DNN) during training to obtain the generator matrix as a function of the DNN weights solely, while providing a differentiable approximation of the lattice continuous-to-discrete mapping to enable gradient-based optimization. When training is concluded, we fix the learned matrix, thus fully preserving the operation of conventional lattice quantizers. Our numerical results show that our approach achieves an enhanced distortion compared with alternative techniques constructing lattices for quantization, for various lattice dimensions.

The rest of this paper is organized as follows: Section 2 briefly reviews lattice quantizers and formulates the design problem. Section 3 details the proposed data-driven scheme, which we numerically evaluate in Section 4. Concluding remarks are provided in Section 5.

## 2. SYSTEM MODEL

### 2.1. Lattice Quantization

Vector quantization is the encoding of a set of continuous-amplitude quantities into a finite-bit representation [1]. Lattice quantizers are form of vector quantizers where the digital representations are taken from points on a lattice, formally defined as follows [9]:

**Definition 2.1** (Lattice Quantizer)**.** *A lattice quantizer of dimension $L \in \mathbb{Z}^+$ with generator matrix $\boldsymbol{G} \in \mathbb{R}^{L \times L}$ maps*

*$\boldsymbol{x} \in \mathbb{R}^L$ into a discrete representation $Q_{\mathcal{L}}(\boldsymbol{x})$ by selecting the nearest point in the lattice $\mathcal{L} \triangleq \{\boldsymbol{G}\boldsymbol{l} : \boldsymbol{l} \in \mathbb{Z}^L\}$, i.e.,*

$$Q_{\mathcal{L}}(\boldsymbol{x}) = \arg\min_{\boldsymbol{z}\in\mathcal{L}} \|\boldsymbol{x} - \boldsymbol{z}\|. \tag{1}$$

$Q_{\mathcal{L}}$ partitions $\mathbb{R}^L$ into cells centered around the lattice points, where the basic cell is $\mathcal{P}_0 = \{\boldsymbol{x} : Q_{\mathcal{L}}(\boldsymbol{x}) = \boldsymbol{0}\}$. When $L = 1$, $Q_{\mathcal{L}}(\cdot)$ specializes scalar uniform quantization.

In Def. 2.1, the number of lattice points in $\mathcal{L}$ is countable but infinite. Thus, to obtain a *finite-bit* representation, it is common to restrict $\mathcal{L}$ to include only points in a given sphere of radius $\gamma$, i.e., replace $\mathcal{L}$ in (1) with

$$\mathcal{L}_\gamma \triangleq \{\boldsymbol{z} \in \mathcal{L} : \|\boldsymbol{z}\| \le \gamma\}. \tag{2}$$

An event in which the input does not reside in this sphere, i.e, when $Q_{\mathcal{L}}(\boldsymbol{x}) \neq Q_{\mathcal{L}_\gamma}(\boldsymbol{x})$, is referred to as *overloading* [7]. The number of points in $\mathcal{L}_\gamma$ dictates the *quantization rate*, defined as the number of bits per sample, i.e., $R \triangleq \frac{1}{L}\log_2|\mathcal{L}_\gamma|$.

Lattice quantization yields a distortion term $\boldsymbol{e} \triangleq Q_{\mathcal{L}}(\boldsymbol{x}) - \boldsymbol{x}$ that is deterministically determined by $\boldsymbol{x}$. It is often combined with *probabilistic quantization* techniques, and particularly with subtractive dithered quantization (SDQ) [7, 8]:

**Definition 2.2** (SDQ)**.** *Let $\boldsymbol{d}$ be drawn uniformly from $\mathcal{P}_0$. The SDQ of $\boldsymbol{x} \in \mathbb{R}^L$ with lattice $\mathcal{L}$ is given by*

$$Q_{\mathcal{L}}^{\mathrm{SDQ}}(\boldsymbol{x}) = Q_{\mathcal{L}}(\boldsymbol{x} + \boldsymbol{d}) - \boldsymbol{d}. \tag{3}$$

Lattice SDQ is a preferable vector quantization technique due to its structured form and statistical properties. Specifically, it realizes universal quantization in the absence of overloading, as then, the distortion induced by $Q_{\mathcal{L}}^{\mathrm{SDQ}}(\boldsymbol{x})$ is an additive noise that is independent of $\boldsymbol{x}$ and its distribution [13].

### 2.2. Problem Formulation

The above statistical properties of non-overloaded lattice SDQ allows to bound the rate-distortion tradeoff of $Q_{\mathcal{L}}^{\mathrm{SDQ}}(\cdot)$ by an expression depending on the lattice via its normalized second moment [9]. However, using this measure to design lattice quantizers may neither lead to a unique quantizer [13], nor reflect the distortion achieved with overloading.

Accordingly, we seek to design a lattice that is directly based on the distortion of a finite-bit lattice quantizer, i.e.,

$$\boldsymbol{G}^\star = \arg\min_{\boldsymbol{G}\in\mathbb{R}^{L\times L}} \mathbb{E}\left\{\left\|\boldsymbol{x} - Q_{\mathcal{L}_\gamma(\boldsymbol{G})}^{\mathrm{SDQ}}(\boldsymbol{x})\right\|\right\}, \tag{4}$$

where $\mathcal{L}_\gamma(\boldsymbol{G})$ is the finite support lattice (2) with generator matrix $\boldsymbol{G}$. We do not assume knowledge of the distribution of $\boldsymbol{x}$. However, for design purposes, we assume access to a dataset comprised of $N$ realizations, denoted $\mathcal{D} = \{\boldsymbol{x}^{(n)}\}_{n=1}^N$.

## 3. DATA-DRIVEN LATTICE QUANTIZERS

In this section we present the proposed scheme for tuning lattice quantizers from data based on (4). We first present how we convert lattice SDQ into a machine learning model and its training procedure in Subsections 3.1-3.2, respectively, and provide a discussion in Subsection 3.3.

### 3.1. Architecture

We leverage deep learning tools for tunning the lattice generator matrix $\boldsymbol{G}$. To that aim, for training purposes, we replace the operation given in Def. 2.2 with a *surrogate trainable machine learning model*, parameterized by $\boldsymbol{w}$, denoted $\tilde{Q}(\boldsymbol{x};\boldsymbol{w})$. This conversion is comprised of DNN augmentation and differentiable approximation; illustrated in Fig. 1, summarized as Algorithm 1, and formalized below.

**DNN Augmentation:** Instead of treating $\boldsymbol{G}$ as a trainable parameter, we follow the deep prior approach of [20], and set it to be the output of a DNN with weights $\boldsymbol{w}$, *fixed* input $\boldsymbol{u}$, and $L^2$ output neurons. The reshaped DNN output, denoted $\boldsymbol{G}_{\boldsymbol{w}}(\boldsymbol{u})$, is used as the generator matrix. This augmentation exploits the abstractness of DNNs, while stabilizing and facilitating the learning procedure

**Differentiable Approximation:** The formulation of (1) limits the ability to apply gradient-based optimization due to its continuous-to-discrete nature. Therefore, we approximate it as a differentiable mapping in two stages: first, we replace the $\arg\min$ operation in (1), which involves searching over the lattice, with element-wise uniform quantization in the inverse lattice domain. Then, following [21], we approximate uniform quantization with a differentiable scalar function applied element-wise, denoted $\phi(\cdot)$, such that

$$\tilde{Q}(\boldsymbol{x};\boldsymbol{w}) = \boldsymbol{G}_{\boldsymbol{w}}(\boldsymbol{u})\phi\left(\boldsymbol{G}_{\boldsymbol{w}}^{-1}(\boldsymbol{u})\Pi_\gamma(\boldsymbol{x}+\boldsymbol{d})\right) - \boldsymbol{d}. \tag{5}$$

In (5), $\boldsymbol{d} \in \mathbb{R}^L$ is the dither signal, uniformly distributed over the basic lattice cell $\mathcal{P}_0$, and $\Pi_\gamma(\cdot)$ truncates per-element values larger than $\gamma$ to constrain a finite-support lattice (finite number of bits), i.e.,

$$\Pi_\gamma(x) = \begin{cases} x, & \text{if } x < |\gamma|; \\ \mathrm{sign}(x)\cdot\gamma, & \text{else}, \end{cases} \qquad x \in \mathbb{R}. \tag{6}$$

Both $\boldsymbol{d}$ and $\Pi_\gamma(\cdot)$ account for the definition of $Q_{\mathcal{L}_\gamma}^{\mathrm{SDQ}}(\cdot)$. Notice that $\Pi_\gamma(x)$ in (6) can be viewed as a form of activation function serving to represent the finite support of the lattice. Finally, the differentiable element-wise approximation $\phi(\cdot)$ in (5) is given by

$$\phi(x) = x + 0.2\cos(2\pi(x+0.25)), \quad x \in \mathbb{R}, \tag{7}$$

where its formulation was inspired by the Fourier series expansion of the rounding function.

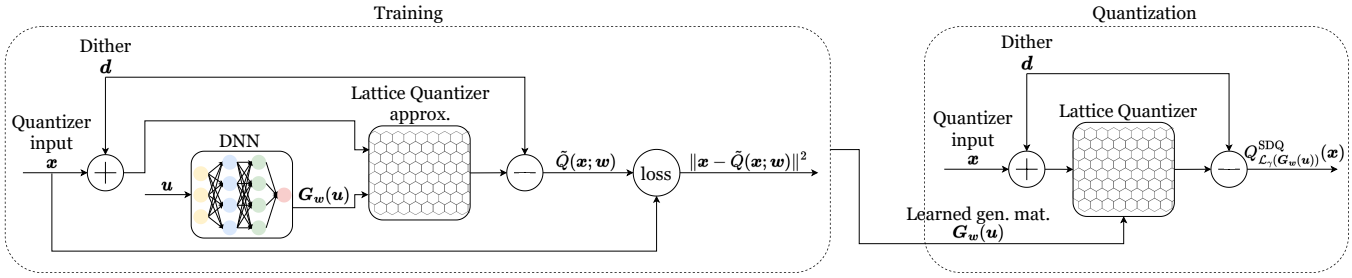


**Fig. 1**. Overview of data-driven lattice quantizers. The left dashed box represents the training stage using the approximation in (5) to learn the generator matrix $\boldsymbol{G}$, that is then fixed and utilized for the quantization routine, described in the right dashed box.

### 3.2. Training

The approximation of $Q^{\text{SDQ}}_{\mathcal{L}_\gamma}(\boldsymbol{x})$ as a machine learning model is used to adjust $\boldsymbol{w}$. As we seek to minimize the distortion, we use the empirical mean squared error (MSE) as our loss measure, i.e., the loss evaluated over a dataset $\mathcal{D}$ is given by

$$\mathcal{E}_{\mathcal{D}}(\boldsymbol{w}) = \frac{1}{|\mathcal{D}|}\sum_{\boldsymbol{x}\in\mathcal{D}}\left\|\boldsymbol{x}-\tilde{Q}(\boldsymbol{x};\boldsymbol{w})\right\|^2. \tag{8}$$

The loss in (8) enables unsupervised learning of the lattice, as, e.g., no ground-truth digital representation is required. The differentiability of (5) enables training using conventional deep learning optimizers as mini-batch stochastic gradient descent (SGD). When training is concluded, i.e., $\boldsymbol{w}$ is learned, the resulting $\boldsymbol{G_w}(\boldsymbol{u})$ is used for standard lattice SDQ as in Def. 2.2.

**Algorithm 1:** Data-Driven Lattice Learning

**Init:** Randomize $\boldsymbol{w}$ and fix parameters $\eta, i_{\max}, \boldsymbol{u}$.
**Input:** Training set $\mathcal{D}$
**for** $i = 0, 1, \ldots, i_{\max}-1$ **do**
  Randomly divide $\mathcal{D}$ into $Q$ batches $\{\mathcal{D}_q\}_{q=1}^{Q}$;
  **for** $q = 1, \ldots, Q$ **do**
    Apply surrogate model $\tilde{Q}(\cdot;\boldsymbol{w})$ to $\mathcal{D}_q$ via (5);
    Compute batch loss $\mathcal{E}_{\mathcal{D}_q}(\boldsymbol{w})$ via (8);
    Update $\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta\nabla_{\boldsymbol{w}}\mathcal{E}_{\mathcal{D}_q}(\boldsymbol{w})$;
**Output:** Generator matrix $\boldsymbol{G_w}(\boldsymbol{u})$

### 3.3. Discussion

The proposed data-driven Algorithm 1 enables learning lattice quantizers from data. We evaluate the generator matrix based on its distortion, without relying on analytical bounds derived by assuming zero overloading. Our design is particularly geared towards employing deep learning tools, while using deep prior techniques for facilitating the tuning of $\boldsymbol{G}$. This allows obtaining lattice quantizers with improved distortion compared to previous designs, by enabling some level of overloading, as empirically demonstrated in Section 4.

The conversion of lattice quantizers into a trainable machine learning models implies that one can consider alternative measures to the MSE loss in (8). For instance, it can facilitate designing lattice quantizers whose digital representation is most useful for some down-stream task, i.e., task-based quantization [6], and can possibly be integrated into DNN-aided task-based qutntizers, e.g., [22,23]. Additionally, while our design considers a quantizer with a fixed support $\gamma$, it can be extended to also learn a support that is most suitable in potentially learning to balance the amount of overloading. We leave these extensions for future study.

## 4. NUMERICAL STUDY

In this section we numerically evaluate our approach[1], demonstrating that data-driven learned lattices yields improved SNRs compared with conventional rate-distortion bound design. We consider the quantization of synthetic data in Subsection 4.1, and of real-data gradients in the application of federated learning (FL) in Subsection 4.2.

### 4.1. Synthetic data

To implement Algorithm 1, we first synthetically initialize the training dateset $\mathcal{D}$ by realizing $L \times 1$ vectors with correlated entries of the form $\{\boldsymbol{F}\boldsymbol{x}^{(n)}\}_{n=1}^{N}$. E.g., for $L = 2$, we randomize each of $\boldsymbol{F}$ and $\boldsymbol{x}^{(n)}$ entries from the normal $\mathcal{N}(1,2)$ and uniform $\mathcal{U}[0,1)$ distributions, respectively. For evaluation purposes, we also construct a test dataset $\mathcal{D}_{\text{test}}$, similarly distributed as the training set $\mathcal{D}$. The surrogate model of the lattice SDQ quantizer $\tilde{Q}(\cdot;\boldsymbol{w})$ is set to be a multi-layer perceptron (MLP) with a fixed all ones input $\boldsymbol{u}$, two hidden layers, and intermediate LeakyReLU activations. Using learning rate $\eta = 1e^{-5}$ we complete $i_{\max} = 100$ training iterations of mini-batch SGD before fixing $\boldsymbol{G_w}(\boldsymbol{u})$.

First, to comprehend the learning outcomes, we construct a visualization of the learned generator matrix $\boldsymbol{G_w}(\boldsymbol{u})$ referenced to the classic hexagonal matrix, and illustrate in Fig. 2 the lattice points and quantization decision cells obtained from both matrices for $L = 2$ and $R = 2$. There, it is revealed that the learned matrix is an approximated rotated version of the hexagonal matrix, alongside the existence of overloading.

[1]The source code used in our experimental study, including all the hyperparameters, is available online at https://github.com/BokoAssaf/DeepLatticeUVEQ.

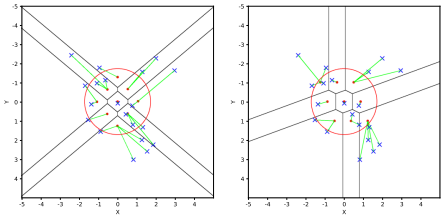

**Fig. 2**. Scatter illustrations of the lattices obtained from the learned (left) and hexagonal (right) generator matrices. Red •'s mark lattice points; blue ×'s are representative data samples, with green lines connecting to the matched lattice points.

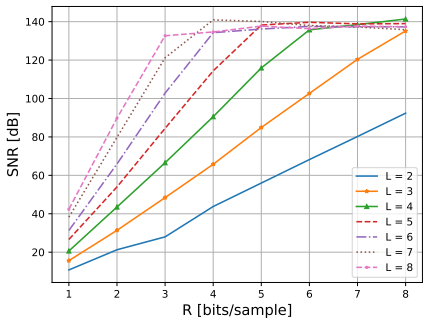


**Fig. 3**. SNR vs. bit-rate for different lattice dimensions.

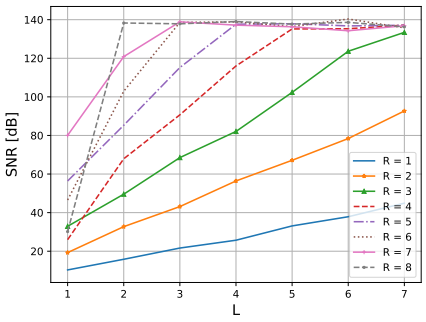


**Fig. 4**. SNR vs. lattice dimension for different bit-rates.

Next, we evaluate the quantization performance using the learned $\boldsymbol{G}_{\boldsymbol{w}}(\boldsymbol{u})$. To that aim, we depict in Fig. 3-4 the SNR obtained for different bit rates $R$ and lattice dimensions $L$ combinations, all using the same dynamic-rage $\gamma$; where the SNR is computed as the estimated variance of the input divided by the estimated variance of the distortion, i.e.,

$$\text{SNR} \triangleq \frac{1}{|\mathcal{D}_{\text{test}}|} \sum_{\boldsymbol{x}\in\mathcal{D}_{\text{test}}} \text{Var}(\boldsymbol{x}) / \text{Var}\left(\boldsymbol{x} - Q^{\text{SDQ}}_{\mathcal{L}_\gamma(\boldsymbol{G}_{\boldsymbol{w}}(\boldsymbol{u}))}(\boldsymbol{x})\right).$$

As expected, Fig. 3 shows that the higher the bit-rate the finer and accurate the discretization is (as a result of using more codewords), and similarly, Fig. 4 demonstrates that a higher lattice dimension results with a better compression (by effectively leveraging the correlated entries dependencies [3]); both outcome with an enhanced SNR. We observe in Figs. 3-4 a saturation of the curves, due to using a fixed dynamic range $\gamma$. That is, starting from a specif $L$ and $R$, further increment of either one or both of them is negligible in SNR, as the latter is dominated by the overloading distortion.

### 4.2. Application: Federated Learning

To examine the generalization power of the matrix learned in the previous subsection using synthetic data, we evaluate its performance on real-world quantities in FL applications [24]. In FL, a group of remote edge users are collaboratively training a model using their local data and devices with periodic aggregations orchestrated by a global shared sever [25]. This process entails communication overhead, and typically involves compression schemes [10–12] employed on the model updates (gradients) of the local users.

We consider the local SGD-based federated training of a handwritten digit classification model using the MNIST dataset [26]. The data, comprised of $28 \times 28$ gray-scale images divided into $60,000$ training examples and $10,000$ test examples, is uniformly distributed among $K = 30$ users. We used FL to train three different architectures: a linear regression model; an MLP with two hidden layers and intermediate ReLU activations; and a convolutional neural network (CNN) composed of two convolutional layers followed by two fully-connected ones, with intermediate ReLU activations and max-pooling layers. All three models use a softmax output layer. To serve as a baseline, we compare the quantization of the resulting model updates using our lattice design learned from the synthetic data of Subsection 4.1 (coined *Lrn*) to the scheme of [16], which adjusts a generator matrix assuming zero-overloading, denoted *Opt*.

**Table 1**. Baselines' SNR resulting from different lattice dimensions for a fixed bit-rate $R = 1$ (top), as well as different bit-rates for a fixed lattice dimension $L = 4$ (buttom).

| | $L=2$ | | | $L=5$ | | | $L=8$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | Linear | MLP | CNN | Linear | MLP | CNN | Linear | MLP | CNN |
| Opt | 1.80 | 1.28 | 2.54 | 25.38 | 23.57 | 27.74 | 40.47 | 38.5 | 41.2 |
| Lrn | 10.74 | 10.29 | 12 | 28.8 | 27.44 | 32.12 | 48.78 | 46.54 | 50.36 |
| | $R=2$ | | | $R=4$ | | | $R=5$ | | |
| Opt | 37.71 | 36.71 | 39.8 | 85.737 | 84.51 | 89.21 | 109.8 | 108.57 | 112.22 |
| Lrn | 44.88 | 42.54 | 48.95 | 91.32 | 90.7 | 100.17 | 116.56 | 115.19 | 119.91 |

Table 1 reports the FL training averaged SNR obtained for both methods with different $R, L$ choices. Our learned option generalized well to unseen data and surpasses its optimized counterpart regardless of the chosen $R, L$ or architecture, stressing that allowing a certain amount of overloading, as illustrated in Fig. 2, can be beneficial in forming lattice quantizers. Nevertheless, both schemes realize an improvement in the SNR once lattices of higher rates or dimensions are considered, while more significantly for the former.

## 5. CONCLUSIONS

We present learnable lattices for vector quantization, realizing SDQ with a data-driven lattice generator matrix. In contrast with existing lattice quantization techniques, we detach the highly common yet less realistic assumption of zero-overloading probability, by directly minimizing the quantization error rather than the rate-distortion tradeoff bound. Our algorithm converts lattice SDQ into a trainable machine learning model, from which the generator matrix is fixed and conventionally incorporated in the quantization outline. The provided numeral evaluations demonstrate that the proposed approach operates more reliably than alternative counterparts, revealing that overloading can aid constructing better lattices designs.

## 6. REFERENCES


[1] R. M. Gray and D. L. Neuhoff, "Quantization," *IEEE Trans. Inf. Theory*, vol. 44, no. 6, pp. 2325–2383, 1998.

[2] R. Gray, "Vector quantization," *IEEE ASSP Mag.*, vol. 1, no. 2, pp. 4–29, 1984.

[3] Y. Polyanskiy and Y. Wu, "Lecture notes on information theory," *Lecture Notes for 6.441 (MIT), ECE563 (University of Illinois Urbana-Champaign), and STAT 664 (Yale)*, 2012-2017.

[4] Y. Linde, A. Buzo, and R. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. 28, no. 1, pp. 84–95, 1980.

[5] A. Van Den Oord and O. Vinyals, "Neural discrete representation learning," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[6] N. Shlezinger, Y. C. Eldar, and M. R. Rodrigues, "Hardware-limited task-based quantization," *IEEE Trans. Signal Process.*, vol. 67, no. 20, pp. 5223–5238, 2019.

[7] R. M. Gray and T. G. Stockham, "Dithered quantizers," *IEEE Trans. Inf. Theory*, vol. 39, no. 3, pp. 805–812, 1993.

[8] S. P. Lipshitz, R. A. Wannamaker, and J. Vanderkooy, "Quantization and dither: A theoretical survey," *Journal of the audio engineering society*, vol. 40, no. 5, pp. 355–375, 1992.

[9] R. Zamir and M. Feder, "On universal quantization by randomized uniform/lattice quantizers," *IEEE Trans. Inf. Theory*, vol. 38, no. 2, pp. 428–436, 1992.

[10] N. Shlezinger, M. Chen, Y. C. Eldar, H. V. Poor, and S. Cui, "UVeQFed: Universal vector quantization for federated learning," *IEEE Trans. Signal Process.*, vol. 69, pp. 500–514, 2020.

[11] N. Lang, E. Sofer, T. Shaked, and N. Shlezinger, "Joint privacy enhancement and quantization in federated learning," *IEEE Trans. Signal Process.*, vol. 71, pp. 295–310, 2023.

[12] N. Lang, N. Shlezinger, R. G. D'Oliveira, and S. E. Rouayheb, "Compressed private aggregation for scalable and robust federated learning over massive networks," *arXiv preprint arXiv:2308.00540*, 2023.

[13] R. Zamir and M. Feder, "On lattice quantization noise," *IEEE Trans. Inf. Theory*, vol. 42, no. 4, pp. 1152–1159, 1996.

[14] S. Lyu, Z. Wang, C. Ling, and H. Chen, "Better lattice quantizers constructed from complex integers," *IEEE Trans. Commun.*, vol. 70, no. 12, pp. 7932–7940, 2022.

[15] E. Agrell and B. Allen, "On the best lattice quantizers," *IEEE Trans. Inf. Theory*, 2023, early access.

[16] E. Agrell and T. Eriksson, "Optimization of lattices for quantization," *IEEE Trans. Inf. Theory*, vol. 44, no. 5, pp. 1814–1828, 1998.

[17] N. Shlezinger and Y. C. Eldar, "Model-based deep learning," *Foundations and Trends® in Signal Processing*, vol. 17, no. 4, pp. 291–416, 2023.

[18] N. Shlezinger, Y. C. Eldar, and S. P. Boyd, "Model-based deep learning: On the intersection of deep learning and optimization," *IEEE Access*, vol. 10, pp. 115 384–115 398, 2022.

[19] N. Shlezinger and T. Routtenberg, "Discriminative and generative learning for linear estimation of random signals [lecture notes]," *IEEE Signal Process. Mag.*, 2023, early access.

[20] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *IEEE conference on computer vision and pattern recognition*, 2018, pp. 9446–9454.

[21] N. Shlezinger, A. Amar, B. Luijten, R. J. van Sloun, and Y. C. Eldar, "Deep task-based analog-to-digital conversion," *IEEE Trans. Signal Process.*, vol. 70, pp. 6021–6034, 2022.

[22] N. Shlezinger and Y. C. Eldar, "Deep task-based quantization," *Entropy*, vol. 23, no. 1, p. 104, 2021.

[23] M. Malka, S. Ginzach, and N. Shlezinger, "Learning multi-rate vector quantization for remote deep inference," in *IEEE International Conference on Acoustics, Speech, and Signal Processing Workshops (ICASSPW)*, 2023.

[24] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[25] T. Gafni, N. Shlezinger, K. Cohen, Y. C. Eldar, and H. V. Poor, "Federated learning: A signal processing perspective," *IEEE Signal Process. Mag.*, vol. 39, no. 3, pp. 14–41, 2022.

[26] L. Deng, "The MNIST database of handwritten digit images for machine learning research," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 141–142, 2012.