Model Based Open Set Recognition

Omer Hazan and Tal Kozakov

August 4, 2025

1 Motivation and Background

Radio Frequency Fingerprint Identification (RFFI) enables device-level authentication by exploiting unique hardware-induced imperfections in RF signals, caused by manufacturing variations in components like oscillators and power amplifiers. This makes RFFI a powerful tool for identifying devices in wireless environments such as mobile phones, IoT nodes, and UAVs. However, real-world settings are inherently Open Set (OS), where new, previously unseen devices may appear. This necessitates solving the problem of Open Set Recognition (OSR)—distinguishing whether a received signal comes from a known device or an unknown one—making open-set detection a critical component of any robust RFFI system.

Recent advances in deep learning, such as CNNs, Vision Transformers, and autoencoder-based models, have demonstrated state-of-the-art performance in RFFI by extracting highly discriminative, high-dimensional features from RF signals. However, these methods are typically optimized for closed-set classification and tend to be overconfident when presented with signals from unknown devices. This limitation motivates the exploration of techniques for open-set detection.

To address these challenges, we propose using Mahalanobis Distance (MD) instead of Euclidean distance in the VQ-VAE entropy-based open-set detection framework. By accounting for covariance structure, MD offers a more statistically grounded similarity measure, improving the separation between known and unknown devices.

2 Problem Formulation

We assume that the latent feature representations extracted from RF signals form class-specific clusters in a d-dimensional space. For each known device class $c \in \{1, \ldots, C\}$, these features $x \in \mathbb{R}^d$ are modeled as samples drawn from a multivariate Gaussian distribution:

$$x \,|\, Y = c \sim \mathcal{N}(\mu_c, \Sigma),$$

where μ_c is the class mean vector and Σ is a shared covariance matrix. Under this model, the class-conditional likelihood of a feature vector x is given by:

$$p(x \,|\, Y = c) = (2\pi)^{-\frac{d}{2}} |\Sigma|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(x - \mu_c)^\top \Sigma^{-1} (x - \mu_c) \right).$$

Maximizing this likelihood is equivalent to minimizing the Mahalanobis distance:

$$D_M(x, c) = \sqrt{(x - \mu_c)^\top \Sigma^{-1} (x - \mu_c)}.$$

Thus, for closed-set classification, the predicted class of x is:

$$c^* = \arg \min_{c \in \{1, \ldots, C\}} D_M(x, c).$$

For open-set recognition, the challenge is to detect when x does not belong to any of the known device classes. This can be formulated as:

$$x \in \text{open set} \quad \text{if} \quad D_M(x) = \min_c D_M(x, c) > \theta,$$

where θ is a threshold separating in-distribution signals from those generated by previously unseen devices. This formulation connects the open-set detection task to the statistical likelihood of a sample under the learned Gaussian class models.

3 Method

3.1 Data Driven Vector-Quantization

Our methodology utilizes data-driven vector quantizers. Quantizers with input size d are defined using a finite codebook $\mathcal{Q} \subset \mathbb{R}^d$, such that each $\mathbf{x} \in \mathbb{R}^d$

is mapped into a B-bit representation $Q(\mathbf{x}) \in \mathcal{Q}$, where $B = \log_2 |\mathcal{Q}|$, typically by selecting the closest codeword in $\mathcal{Q}$ [1].

When $d > 1$, this process is referred to as vector quantization, as opposed to scalar quantization. Theoretical results show that vector quantization is optimal under certain assumptions, achieving lower distortion for a given bit budget [1].

Vector quantizers can be tuned from data, i.e., multiple realizations of $\mathbf{x}$. A classic approach to learn $\mathcal{Q}$ is the Linde–Buzo–Gray (LBG) algorithm [2], where k codewords are chosen as centroids of clustered data points, with $k = |\mathcal{Q}|$.

Recent works in deep learning demonstrated the advantages of using learned codebooks that were trained on data. A representative example is the vector quantized variational autoencoder (VQ-VAE) [3]. In the VQ-VAE, the input $\mathbf{x}$ is processed by a deep neural network (DNN) encoder into $\mathbf{x}^{\mathrm{e}} = f_{\mathrm{e}}(\mathbf{x})$, e.g., a low-dimensional representation of the input. Subsequently, the vector $\mathbf{x}^{\mathrm{e}}$ is decomposed into M vectors of size d, denoted $\{\mathbf{x}^{\mathrm{e}}_m\}_{m=1}^{M}$, and each is represented by the closest codeword in $\mathcal{Q}$:

$$\mathbf{q}^m = \arg\min_{\mathbf{q}\in\mathcal{Q}} \|\mathbf{x}^{\mathrm{e}}_m - \mathbf{q}\|_2. \tag{1}$$

The quantized $\mathbf{q} = Q(\mathbf{x}^{\mathrm{e}}) = [\mathbf{q}^1, \dots, \mathbf{q}^M] \in \mathcal{Q}^M$ is represented using $M \cdot B$ bits. A DNN decoder maps the quantized representation into the overall output $\hat{\theta} = f_d(\mathbf{q})$. To jointly train the encoder-decoder while learning the codebook, the VQ-VAE uses a three-term loss function:

$$\mathcal{L}^{\mathrm{tot}}(\theta; \mathbf{x}) = \mathcal{L}(\theta; \hat{\theta}) + \|\mathrm{sg}(\mathbf{x}^{\mathrm{e}}) - \mathbf{q}\|_2^2 + \beta\|\mathbf{x}^{\mathrm{e}} - \mathrm{sg}(\mathbf{q})\|_2^2, \tag{2}$$

where $\mathrm{sg}(\cdot)$ is the stop-gradient operator, $\mathcal{L}\big(\theta; \hat{\theta}\big)$ is the task-dependent loss, and $\beta > 0$ is a hyperparameter. While alternative loss measures have been recently proposed to boost codebook utilization [4], (2) remains the standard objective for training DNN-aided vector quantizers [5].

3.2 Five-Stage Training Procedure of the Baseline Work

The baseline VQ-VAE based framework for RFFI employs a five-stage training pipeline designed to progressively improve feature representations and quantization quality. The stages are as follows:

1. **Stage 1 – Encoder-Classifier Training (CE Loss):** A convolutional neural network (CNN) encoder $f_e(\cdot)$ is trained jointly with a multi-layer perceptron (MLP) classifier $g(\cdot)$ using the standard cross-entropy (CE) loss:

   $$\mathcal{L}_{\text{CE}} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c}\log \hat{y}_{i,c},$$

   where $y_{i,c}$ is the one-hot ground truth label of sample i, $\hat{y}_{i,c} = g(f_e(x_i))$ is the predicted probability for class c, N is the number of samples, and C is the number of known classes.

2. **Stage 2 – Encoder Fine-Tuning with Triplet Loss:** The encoder weights obtained from Stage 1 are further fine-tuned using a triplet loss to improve intra-class compactness and inter-class separability. For each triplet (x^a, x^p, x^n), consisting of an anchor, a positive sample (same class), and a negative sample (different class), the triplet loss is:

   $$\mathcal{L}_{\text{Triplet}} = \frac{1}{N_t}\sum_{i=1}^{N_t}\max\Big(0,\ \|z_i^a - z_i^p\|_2^2 - \|z_i^a - z_i^n\|_2^2 + \alpha\Big),$$

   where $z_i = f_e(x_i)$ are latent features, α is a margin parameter, and N_t is the number of triplets.

3. **Stage 3 – Classifier Retraining on Refined Features:** A new MLP classifier is trained on top of the encoder from Stage 2 using the CE loss $\mathcal{L}_{\text{CE}}$ as defined above.

4. **Stage 4 – Codebook Construction with LBG:** The Linde–Buzo–Gray (LBG) algorithm [2] is used to construct a codebook $\mathcal{Q} = \{\mathbf{q}_1, \ldots, \mathbf{q}_K\}$ by minimizing the vector quantization distortion:

   $$\mathcal{L}_{\text{LBG}} = \frac{1}{N}\sum_{i=1}^{N}\min_{\mathbf{q}_j\in\mathcal{Q}}\|z_i - \mathbf{q}_j\|_2^2,$$

   where $z_i = f_e(x_i)$ are the encoder features.

5. **Stage 5 – Joint VQ-VAE Training (Quantization Loss):** The final VQ-VAE model is initialized with the encoder and classifier from Stage 3 and the codebook from Stage 4. It is trained using the VQ-VAE loss:
$$\mathcal{L}_{\text{VQ}} = \mathcal{L}_{\text{task}} + \|\text{sg}[z] - q\|_2^2 + \beta\|z - \text{sg}[q]\|_2^2,$$
where $z = f_e(x)$, q is the closest codeword from $\mathcal{Q}$, $\text{sg}[\cdot]$ is the stop-gradient operator, $\beta > 0$ is a weighting parameter, and $\mathcal{L}_{\text{task}}$ is the CE loss of the classifier.

This staged procedure ensures that the learned codebook and quantized feature space preserve discriminative information while enabling open-set detection based on codeword distances.

3.3 Proposed Modifications

Our work introduces two main changes to the baseline five-stage pipeline: (i) modifying the triplet loss to use the class mean as the positive sample with Mahalanobis distance, and (ii) adapting the VQ-VAE quantizer to Mahalanobis distance, where the covariance matrix is estimated from the codewords. In both cases, we found that the estimated covariance matrices were not always invertible due to the high dimensionality and limited batch size. Therefore, we regularize the covariance estimates by adding a small diagonal term:
$$\tilde{\Sigma} = \Sigma + \lambda I, \quad \lambda = 10^{-3},$$
where I is the identity matrix.

3.3.1 Triplet Loss with Class Mean and Mahalanobis Distance

In the baseline method, the triplet loss uses Euclidean distance and individual positive samples. We instead define the positive sample for a class c as the mean of the latent features of class c within the current batch:

$$\bar{z}_c = \frac{1}{|\mathcal{B}_c|} \sum_{i:y_i=c} z_i,$$

where $\mathcal{B}_c$ is the set of indices of samples from class c in the batch, and $z_i = f_e(x_i)$ are the encoder features.

We replace the Euclidean distance with the regularized Mahalanobis distance:

$$D_M(z, \bar{z}_c) = \sqrt{(z - \bar{z}_c)^\top \tilde{\Sigma}_b^{-1} (z - \bar{z}_c)},$$

where $\tilde{\Sigma}_b = \Sigma_b + 10^{-3} I$, with

$$\Sigma_b = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} (z_i - \bar{z})(z_i - \bar{z})^\top, \quad \bar{z} = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} z_i.$$

The modified triplet loss is then:

$$\mathcal{L}_{\text{Triplet}}^{M} = \frac{1}{N_t} \sum_{i=1}^{N_t} \max\Big(0,\, D_M^2(z_i^a, \bar{z}_{c^a}) - D_M^2(z_i^a, z_i^n) + \alpha\Big),$$

where z_i^a is the anchor, $\bar{z}_{c^a}$ is the class mean of the anchor's class, and z_i^n is the negative sample.

3.3.2 Mahalanobis Distance Quantizer

In the VQ-VAE stage, the original quantizer selects the nearest codeword using the Euclidean distance:

$$q = \arg \min_{\mathbf{q}_j \in \mathcal{Q}} \|z - \mathbf{q}_j\|_2^2.$$

We replace this with a Mahalanobis distance-based quantizer:

$$q = \arg \min_{\mathbf{q}_j \in \mathcal{Q}} D_M^2(z, \mathbf{q}_j),$$

where

$$D_M^2(z, \mathbf{q}_j) = (z - \mathbf{q}_j)^\top \tilde{\Sigma}_Q^{-1} (z - \mathbf{q}_j),$$

and the covariance $\tilde{\Sigma}_Q = \Sigma_Q + 10^{-3} I$ is estimated from the codewords:

$$\Sigma_Q = \frac{1}{K} \sum_{j=1}^{K} (\mathbf{q}_j - \bar{\mathbf{q}})(\mathbf{q}_j - \bar{\mathbf{q}})^\top, \quad \bar{\mathbf{q}} = \frac{1}{K} \sum_{j=1}^{K} \mathbf{q}_j.$$

The quantization and commitment losses in the VQ-VAE remain as in the baseline, but the codeword assignment step now uses D_M instead of Euclidean distance.

4 Experiments and Results

4.1 Close-Set Results

We report two types of close-set classification results: (i) performance on the original 30-device training set with augmentation, and (ii) transferability of the trained feature extractor to 10 unseen devices.

4.1.1 Training on 30 Devices.

The first evaluation is based on a dataset consisting of 30 wireless devices [6], each represented by 1000 packets. The dataset was augmented using various channel effects as described in [6], and additive white Gaussian noise (AWGN) was applied for improved robustness of the feature extractor. Specifically, each packet was assigned a random SNR uniformly sampled from the range $(20, 80)\,\mathrm{dB}$. The dataset was split into 90% training and 10% validation sets to promote generalization and avoid overfitting. We report the close-set classification accuracy on the validation set, providing a measure of the feature extractor's ability to distinguish between the 30 known devices under diverse channel and noise conditions.

4.1.2 Generalization to 10 Unseen Devices.

To evaluate the generalization capability of the learned feature extractor, we consider a separate dataset of 10 devices that were not part of the initial training phase. We reuse the feature extractor trained on the 30-device dataset and freeze its parameters. A new codebook is generated using the Linde–Buzo–Gray (LBG) algorithm [2], and a new classifier is trained on top of the frozen encoder. For this evaluation, we use 100 packets per device for training and another 100 packets per device for testing, without any data augmentation or code-induced noise. This setup isolates the ability of the learned encoder to generalize device-specific features to previously unseen devices.

The results of both experiments are summarized in Table 1, which reports the accuracy on the validation and test sets for the two scenarios described above.

Table 1: Close-Set Classification Accuracy Comparison

Test Case	Ours (Mahalanobis)	Baseline (Euclidean)
30 Devices (Validation Set)	84.77%	**85.17%**
10 Unseen Devices (Test Set)	**98.6%**	97.90%

4.2 Open-Set Recognition Results

To evaluate the open-set detection performance of our model, we conduct experiments using an additional set of 5 unseen devices, each contributing 100 packets. These devices were not used during the training of the feature extractor or the construction of the codebook. As the known-class (in-distribution) reference, we use the 100 test packets from the 10 unseen devices previously described in the close-set generalization experiment (i.e., the test set used after training the classifier and codebook on the other 100 packets per device).

For each packet, we compute the inverse Mahalanobis distance between its latent representation and all codewords in the learned codebook:

$$s_j(x) = \frac{1}{D_M(x, \mathbf{q}_j) + \epsilon} \quad \text{for } j = 1, \ldots, K,$$

where $\mathbf{q}_j$ is the jth codeword and $D_M(x, \mathbf{q}_j)$ is the Mahalanobis distance as defined in Section 2. $\epsilon = 10^{-6}$ for stability

These inverse distances form a histogram over the codebook for each input, from which we extract three scalar summary statistics to serve as open-set recognition (OSR) scoring functions:

- **Entropy:** Measures the uncertainty in the histogram:

$$H(x) = -\sum_{j=1}^{K} p_j(x) \log p_j(x),$$

where $p_j(x) = \frac{s_j(x)}{\sum_{k=1}^{K} s_k(x)}$ is the normalized histogram.

- **Highest Peak:** The maximum histogram value:

$$\max_j p_j(x)$$

- **Mean:** The arithmetic mean of the histogram values:

$$\mu(x) = \frac{1}{K} \sum_{j=1}^{K} p_j(x)$$

Each of these metrics is used as a scalar anomaly score for OSR. Specifically, we compute the Receiver operating characteristic (ROC) curve and the Area uder curve (AUC) by treating the 10-device test packets as in-distribution and the 5 new device packets as out-of-distribution. The resulting AUC scores indicate the effectiveness of each metric in separating known and unknown device classes.

The AUC results for each metric are reported in Table 2.

Table 2: AUC Scores for Open-Set Recognition Metrics

Metric	**Ours (Mahalanobis)**	**Baseline (Euclidean)**
Entropy	**1.000**	0.9945
Highest Peak	**1.000**	0.9977
Mean	**0.9998**	0.9901

5 Discussion

The experimental results demonstrate that incorporating Mahalanobis distance into the VQ-VAE framework improves both close-set classification on unseen devices and open-set recognition (OSR), as shown by higher accuracy and AUROC metrics (Table 2). This improvement stems from Mahalanobis distance's ability to account for the covariance structure in the latent space, resulting in more compact and better-separated class clusters than Euclidean distance. However, Mahalanobis-based modeling introduces training challenges. Estimating covariance matrices is sensitive to batch size and codebook content, often requiring regularization ($1 \times 10^{-3} I$) to ensure numerical stability.

To analyze codebook behavior, we visualize the inverse Mahalanobis distances between a sample's latent vector and all codewords. Each histogram corresponds to one sample, with bar height reflecting $1/D_M(x, q)$. Green bars indicate codewords belonging to the same class as the input. This allows us to assess activation sharpness and codeword selectivity.

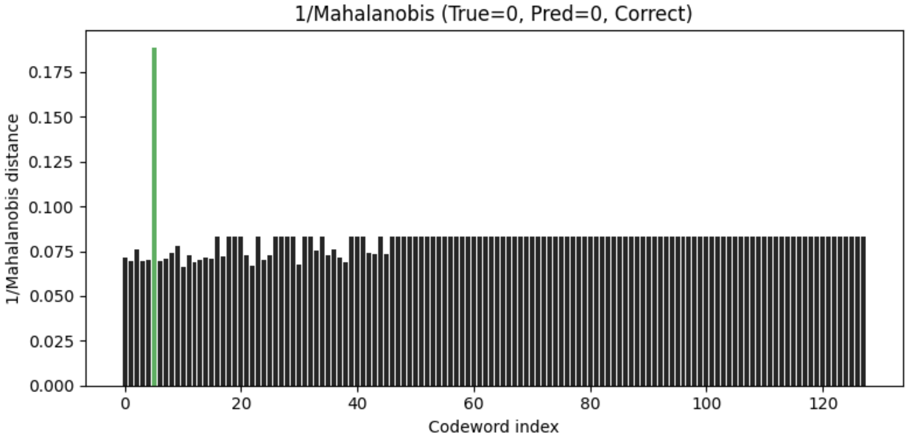

Figure 1: Single-codeword collapse

Figure 1 shows a collapse mode where a class is assigned just one codeword far from the expected 13 codewords/class (given 128 codewords and 10 classes). This limits expressiveness and harms discrimination.

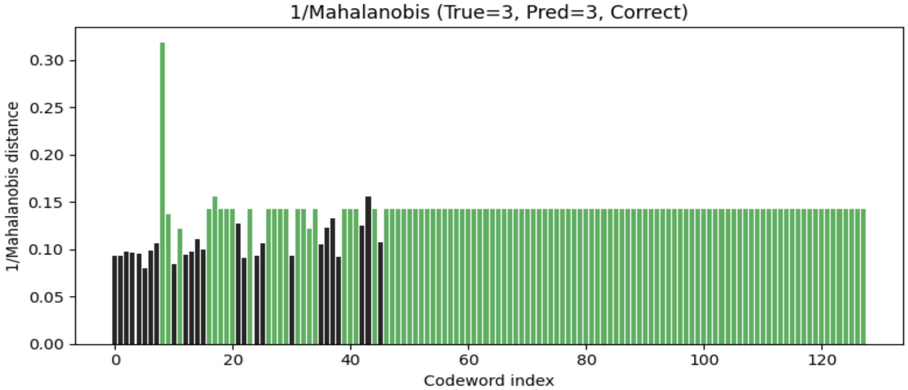

Figure 2: Collapse onto class 3

Figure 2 shows another failure case codebook collapse onto class 3, leaving others underrepresented.

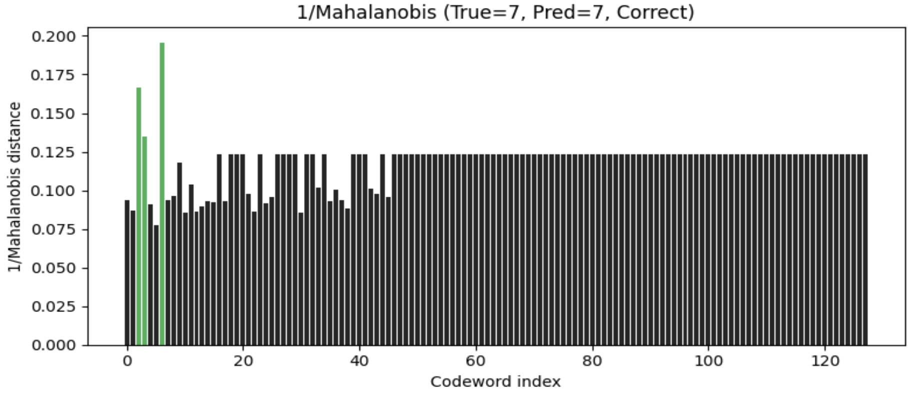

Figure 3: Multiple-codeword assignment

Figure 3 shows a rare improved scenario with more balanced assignments.

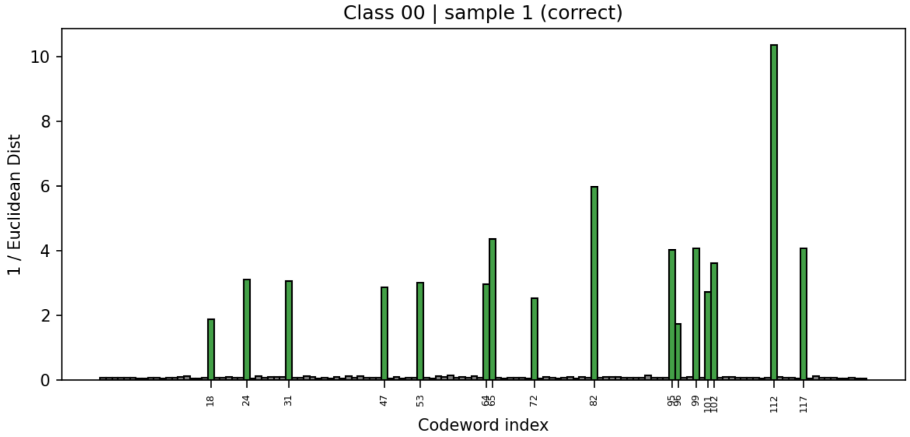

Figure 4: Euclidean quantization

By contrast, Figure 4 illustrates the Euclidean case, where codewords are better distributed—closer to the ideal 13 per class.

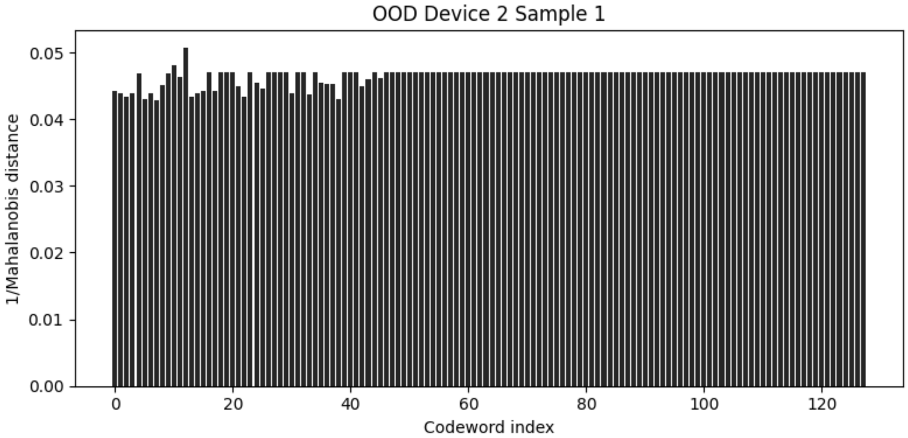

Figure 5: Rogue sample histogram

Figure 5 shows a rogue sample's histogram. Despite codeword collapse issues, Mahalanobis-based models yield more consistent rogue sample histograms across devices, contributing to improved OSR robustness.

Overall, while Mahalanobis distance enhances the statistical fidelity of the latent space and improves generalization, it requires careful tuning and suffers from codebook instability. Future work may focus on stabilizing quantization under Mahalanobis geometry using entropy regularization or adaptive pruning strategies.

References

[1] 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.

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

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

[4] C. Fifty, R. G. Junkins, D. Duan, A. Iger, J. W. Liu, E. Amid, S. Thrun, and C. Ré, “Restructuring vector quantization with the rotation trick,” *arXiv preprint arXiv:2410.06424*, 2024.

[5] M. Huang *et al.*, “Towards accurate image coding: Improved autoregressive image generation with dynamic vector quantization,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 22 596–22 605.

[6] G. Shen, J. Zhang, A. Marshall, and J. R. Cavallaro, “Towards scalable and channel-robust radio frequency fingerprint identification for lora,” *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 774–787, 2022.