HyperVQ: MLR-based Vector Quantization in Hyperbolic Space

Anonymous authors
Paper under double-blind review

Abstract

The success of models operating on tokenized data has heightened the need for effective tokenization methods, particularly in vision and auditory tasks where inputs are naturally continuous. A common solution is to employ Vector Quantization (VQ) within VQ Variational Autoencoders (VQVAEs), transforming inputs into discrete tokens by clustering embeddings in Euclidean space. However, Euclidean embeddings not only suffer from inefficient packing and limited separation—due to their polynomial volume growth—but are also prone to codebook collapse, where only a small subset of codebook vectors are effectively utilized. To address these limitations, we introduce HyperVQ, a novel approach that formulates VQ as a hyperbolic Multinomial Logistic Regression (MLR) problem, leveraging the exponential volume growth in hyperbolic space to mitigate collapse and improve cluster separability. Additionally, HyperVQ represents codebook vectors as geometric representatives of hyperbolic decision hyperplanes, encouraging disentangled and robust latent representations. Our experiments demonstrate that HyperVQ matches traditional VQ in generative and reconstruction tasks, while surpassing it in discriminative performance and yielding a more efficient and disentangled codebook.

1 Introduction

Tokenization has become an essential step in numerous data processing pipelines with the surge of transformer-based models applied to diverse tasks and domains. While tokenizing discrete data like text is straightforward, continuous data types such as images or audio demand a distinct approach.

Typically, to discretize data, a VQVAE (Van Den Oord et al., 2017) is employed, and the resulting discrete tokens serve as input for training other token-based models. The choice of tokenization method significantly influences the model's performance, depending on the task. For instance, tokens trained with reconstruction objectives excel in generative tasks such as VQVAE (Van Den Oord et al., 2017) and soundstream (Zeghidour et al., 2021), where the goal is to reconstruct input or generate new samples. In contrast, discriminative tasks like quality-independent representation learning for robust image classification (Yang et al., 2023) or learning semantic speech representations (Baevski et al., 2020b) utilize a discriminative objective, guiding tokens to be less redundant and more discriminative. Additionally, disentangled features simplify the learning process for other models built on top of tokenized representations. In this work, we explore a disentangled tokenization method suitable for both generative and discriminative tasks.

Despite the popularity of VQVAEs, a well-known issue in the standard (Euclidean) setting is *codebook collapse*, wherein only a small subset of codebook vectors is effectively utilized during training. This phenomenon arises partly due to the polynomial volume growth in Euclidean space, which leads to inefficient cluster separation and a tendency for the encoder outputs to concentrate around a few dominant codebook vectors. Such collapse reduces the representational capacity of the model, negatively impacting both generative and discriminative performance.

Recent trends highlight the advantages of learning representations in non-Euclidean spaces, especially the capability of hyperbolic spaces to naturally induce compact embeddings due to exponential volume growth with radius. Hyperbolic representations are known for better generalization, less overfitting, reduced computational complexity, lower data requirements, and preservation of local and geometric information (Peng et al., 2021). Several works have explored hyperbolic spaces for learning text representations (Nickel & Kiela, 2017), images (Khrulkov et al., 2020; Atigh et al., 2022), audio (Petermann et al., 2023), etc., demonstrating advantages over Euclidean counterparts. Crucially, the exponential volume growth in hyperbolic space can improve cluster separability and mitigate codebook collapse by better distributing latent representations within a bounded geometry.

In this work, we harness the properties of hyperbolic spaces to develop a robust and efficient tokenization method that addresses codebook collapse and enhances disentanglement. We propose formulating the nearest neighbor search problem in VQ as a multinomial logistic regression in hyperbolic space, with quantized vectors serving as representative points of hyperbolic decision hyperplanes. Our approach, **HyperVQ**, differs from Gumbel Vector Quantization (GumbelVQ) (Baevski et al., 2020a), which predicts codebook vector indices using the straight-through estimator (Bengio et al., 2013). The key distinctions lie in our choice of the latent space and the selection of geometrically constrained codebook vectors, rather than optimizing a separate codebook matrix. By leveraging the exponential volume growth of hyperbolic space, HyperVQ not only alleviates codebook collapse but also fosters implicit disentanglement of the latent space. Moreover, selecting a representative point on the decision hyperplane as the codebook vector enhances robustness to noise and outliers, making training less sensitive to codebook vector initialization. Our experimental results validate that the proposed method improves the discriminative performance of tokenization while maintaining generative quality.

The contributions of this work are summarized as follows:

- **First realization of hyperbolic vector quantization**: We introduce a vector quantization method, a key building block of modern deep learning architectures, for hyperbolic spaces.
- **Geometrically constrained codebook vectors**: Our formulation leads to enhanced disentanglement and compact latent representations while mitigating codebook collapse.
- **Improved discriminative performance**: We demonstrate that HyperVQ not only preserves generative performance on par with standard VQ approaches but also achieves superior discriminative capabilities.

2 Related Work

2.1 Discrete Representation Learning

Discrete representation of data is essential for various purposes, including compression and tokenization. While vector quantization Gray (1984) has been a critical component of classical signal processing methods, it has gained significant traction in the deep learning community. First proposed in Van Den Oord et al. (2017), the VQVAE applies vector quantization to the latent space of a VAE, transitioning from a continuous to a discrete latent space. This approach offers advantages, such as learning powerful priors on the discrete space, as demonstrated by pixelCNN Van den Oord et al. (2016).

More recently, tokenized image synthesis methods like VQGAN Esser et al. (2021), VQGAN+CLIP Crowson et al. (2022), and VQ-Diffusion Gu et al. (2022) have shown impressive results in image generation. While discrete representation has excelled in generative tasks, vector quantization has also been integrated into discriminative tasks. For instance, Yang et al. Yang et al. (2023) proposed combining quantized representation with unquantized ones for quality-independent representation learning, and Lu et al. Lu et al. (2023) introduced a hierarchical vector quantized transformer for unsupervised anomaly detection. Vector quantization's applicability extends beyond computer vision to domains such as audio Baevski et al. (2020b); Zeghidour et al. (2021); Baevski et al. (2020a).

While several works have explored enhancing vector quantization operations to address issues like index collapse Łańcucki et al. (2020); Roy et al. (2018); Takida et al. (2022), to the best of our knowledge, no work has explored simultaneously improving the generative and discriminative capabilities of quantized representations.

2.2 Hyperbolic Deep Learning

Non-Euclidean geometry offers a promising approach to uncover inherent geometrical structures in high-dimensional data. Hyperbolic spaces, known for their exponential volume growth with respect to radius, induce low-distortion embeddings and provide better model interpretation Peng et al. (2021). Hyperbolic embeddings exhibit robustness to noise and outliers and demonstrate better generalization capabilities with reduced overfitting, computational complexity, and data requirements Nickel & Kiela (2017); Khrulkov et al. (2020).

Various works, including Ganea et al. (2018); Shimizu et al. (2021), lay the mathematical foundation for creating neural networks in hyperbolic space. Building upon these foundations, methods have been proposed to address tasks such as image segmentation Atigh et al. (2022), audio source separation Petermann et al. (2023), image-text representation learning Desai et al. (2023), and variational autoencoders Mathieu et al. (2019); Skopek et al. (2020).

While a solid foundation of hyperbolic neural network building blocks have been laid, there still lacks a principled and effective vector quantization method utilizing the hyperbolic space. Since VQ is such an important mechanism of several modern neural network architectures across domains. In this work, we explore the definition of a VQ method for the hyperbolic space which can improve feature disentanglement while maining the geometric properties of the quantized latents.

3 Preliminaries

In this section, we provide a concise overview of non-Euclidean geometry and relevant theories crucial for understanding the foundations of our work. While we touch upon the basics, for an in-depth exploration, we recommend referring to Ganea et al. (2018). Additionally, we introduce the concepts of vector quantization and vector quantized variational autoencoders, laying the groundwork for the subsequent discussions.

3.1 Hyperbolic Geometry

3.1.1 Riemannian Manifold

A manifold is a n-dimensional topological space $\mathcal{M}$, such that $\forall x \in \mathcal{M}$, the tangent space, $\mathcal{T}_x\mathcal{M}$ of $\mathcal{M}$ at x, can be approximated by $\mathbb{R}^n$. A manifold paired with a group of Riemannian metric tensors, $g_x : \mathcal{T}_x\mathcal{M} \times \mathcal{T}_x\mathcal{M} \to \mathbb{R}^n$, is called a Riemannian manifold $(\mathcal{M}, g)$.

3.1.2 Hyperbolic Space and Poincaré Ball Model

A Riemannian manifold is called a hyperbolic space if its sectional curvature is negative everywhere. There are several models to represent the n-dimensional hyperbolic space with constant negative curvature, such as the Hyperboloid model, the Poincaré ball model, the Beltrami-Klein model, etc.

In this work, we use the Poincaré ball model, which is defined by the pairing of the manifold $\mathbb{P}^n_c = \left\{x \in \mathbb{R}^n \mid \|x\| < \frac{1}{\sqrt{c}}\right\}$ and the metric $g^{\mathbb{P}}_x = \lambda_x^2 g^{\mathbb{R}^n}_x$ where c is the radius of the ball, $g^{\mathbb{R}^n}_x$ is the Euclidean metric on $\mathbb{R}^n$ and $\lambda_x = 2(1 - \|x\|^2)^{-1}$ is called the conformal factor between the two metrics. Two metrics are conformal if they define the same angles.

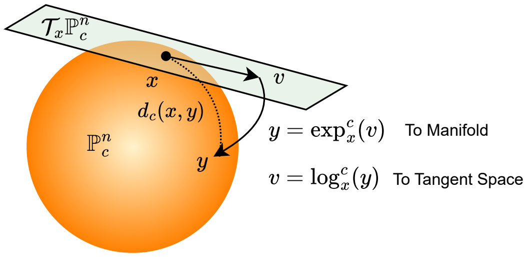

$\mathcal{T}_x\mathbb{P}^n_c$
v
x
$d_c(x,y)$
$\mathbb{P}^n_c$
y
$y = \exp^c_x(v)$ To Manifold
$v = \log^c_x(y)$ To Tangent Space

Figure 1: Illustration of the Poincaré ball model and its associated exponential and logarithmic maps

3.1.3 Gyrovector Spaces and Poincaré Operations

Gyrovector spaces provide an algebraic structure to the hyperbolic space, which is analogous to the vector space structure of the Euclidean space. Several operations on the Poincaré ball model $\mathbb{P}_c^n$ are defined under this framework such as Möbius addition $\oplus_c$, distance between points on the manifold $d_c(x, y)$, exponential and logarithmic maps. We refer the reader to Ganea et al. (2018) for a detailed explanation of these operations. Here we present the definitions of the exponential and logarithmic maps which are pertinent to our work. The following operations are defined under the framework of gyrovector spaces and considering the Poincaré ball model $\mathbb{P}_c^n$:

$$\exp_x^c(v) = x \oplus_c \left(\tanh\left(\sqrt{c}\frac{\lambda_x \|v\|}{2} \right) \frac{v}{\sqrt{c}\|v\|} \right) \tag{1}$$

$$\log_x^c(y) = \frac{2}{\sqrt{c}\lambda_x} \tanh^{-1}\left(\sqrt{c}\, \|-x \oplus_c y\| \right) \frac{-x \oplus_c y}{\|-x \oplus_c y\|} \tag{2}$$

The exponential and logarithmic maps allow us to move between the tangent space (Euclidean) and the manifold as shown in Figure 1.

3.1.4 Poincaré Hyperplanes, MLR and Unidirectional MLR

In Ganea et al. (2018), the authors generalized the concept of a hyperplane in Euclidean space to hyperbolic space, by defining it as the set of all geodesics containing an arbitrary point $p \in \mathbb{P}_c^n$ and orthogonal to a tangent vector $a \in \mathcal{T}_p\mathbb{P}_c^n \setminus \{\mathbf{0}\}$.

$$H_{a,p}^c = \{x \in \mathbb{P}_c^n : \langle -p \oplus_c x, a \rangle = 0\} \tag{3}$$

However, as shown in Shimizu et al. (2021), this definition causes over-parameterization of the hyperplane. They instead proposed an unidirectional hyperplane formulation by fixing the choice of the point on the manifold to be parallel to the normal vector. This is done by introducing a new scalar parameter $r_k \in \mathbb{R}$ which allows the definition of the bias vector in terms of r_k and the normal vector a_k, as,

$$q_k = \exp_0^c(r_k[a_k]) \tag{4}$$

and correspondingly, the hyperplane can be defined in terms of q_k as,

$$\bar{H}_{a_k,r_k}^c := \{x \in \mathbb{P}_c^n \mid \langle -q_k \oplus_c x, a_k \rangle = 0\} \tag{5}$$

Following this new formulation, the hyperbolic multinomial logistic regression (MLR), which is a generalization of the Euclidean MLR formulation (from the perspective of distances to margin hyperplanes), to perform classification tasks in hyperbolic space was defined. The unidirectional hyperbolic MLR formulation given K classes, $k \in \{1, 2, \ldots, K\}$ and for all $x \in \mathbb{P}_c^n$ is defined as follows:

$$\begin{aligned} v_k &= p(y = k \mid x) \\ &\propto \operatorname{sign}\left(\langle -q_k \oplus_c x, a_k \rangle \right) \|a_k\|\, d_c\left(z_h, \bar{H}_{a_k,r_k}^c \right) \end{aligned} \tag{6}$$

3.2 Vector Quantization and Limitations in Euclidean VQVAEs

Vector Quantization (VQ) is a widely used data compression and signal processing technique wherein a continuous input signal $z_e \in \mathbb{R}^N$ is approximated by mapping it to the nearest vector from a finite codebook $C = \{C_1, C_2, \ldots, C_K\}$. Formally,

$$k = q(z_e) \;=\; \arg\min_k \|z_e - C_k\|, \quad \hat{z}_e = z_q = C_k, \tag{7}$$

where k is the index of the nearest codebook vector. VQ is central to applications such as image and speech compression, clustering, and quantization tasks, by reducing data dimensionality while preserving essential information.

Built atop this idea, the Vector Quantized Variational Autoencoder (VQVAE) introduces a discrete latent space between the encoder and decoder of a standard Variational Autoencoder. The encoder outputs a continuous latent vector z_e, which is quantized into z_q via VQ, and subsequently passed to the decoder for input reconstruction. Although this discrete mapping is non-differentiable, end-to-end training is facilitated by a straight-through estimator that bypasses the discontinuity. The training objective typically includes a commitment loss:

$$\mathcal{L} = \log p\big(x \mid z_q\big) \; + \; \big\|\mathrm{sg}[z_e] - z_q\big\|_2^2 \; + \; \beta \, \big\|z_e - \mathrm{sg}[z_q]\big\|_2^2, \tag{8}$$

where $\mathrm{sg}[\cdot]$ is the stop-gradient operator and β weights the commitment term. Variants of VQVAE have explored alternative codebook selection methods, such as replacing the nearest-neighbor lookup with a Gumbel Softmax sampling mechanism (Bengio et al., 2013; Baevski et al., 2020a).

Despite their widespread applicability, Euclidean VQVAEs can suffer from codebook collapse, wherein only a small subset of codebook vectors is effectively utilized. Concretely, let $\mathcal{B} = \{z_e^{(i)}\}_{i=1}^N$ be a batch of encoded samples, and define the codebook usage distribution

$$p\big(C_k\big) \; = \; \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\big(q(z_e^{(i)}) = k\big), \tag{9}$$

where $\mathbb{I}(\cdot)$ is the indicator function and $q(\cdot)$ is the nearest-neighbor assignment. Codebook collapse arises when $p(C_k) \approx 0$ for most k, meaning the majority of samples map to a small subset of codebook vectors. This tendency is exacerbated by the polynomial volume growth in Euclidean space, limiting how effectively clusters can be separated as the model's capacity or data complexity grows. Consequently, the encoder outputs can concentrate around a few dominant codebook vectors, reducing the representational capacity of the model and negatively impacting both its generative and discriminative performance.

Our work addresses these shortcomings by exploring a hyperbolic formulation of vector quantization. By exploiting the exponential volume growth of hyperbolic space, we seek to better distribute latent representations across codebook vectors and mitigate collapse, thereby enhancing the overall robustness and efficiency of the learned discrete representations.

4 Method

4.1 Overview

In an n-dimensional Euclidean space $\mathbb{R}^n$, the volume of an n-ball of radius r grows as $V_{\mathrm{E}}(r) \; \propto \; r^n$, making it necessary to expand r substantially to maintain clear separation among multiple clusters. By contrast, in n-dimensional hyperbolic space ($\mathbb{P}_c^n$), the volume of a ball of radius r satisfies

$$V_{\mathrm{H}}(r) \; \propto \; \sinh^{n-1}(r) \; \approx \; e^{(n-1)r} \quad \text{for large } r. \tag{10}$$

Hence, if we consider k clusters each modeled as an n-ball, their total volumes scale as

$$V_{\text{total,E}} \propto \; k \, r^n \quad \big(\text{Euclidean space}\big), \tag{11}$$

$$V_{\text{total,H}} \propto \; k \, e^{(n-1)r} \quad \big(\text{Hyperbolic space}\big). \tag{12}$$

Because the entire hyperbolic space $\mathbb{P}_c^n$ lies within a unit ball of Euclidean radius $1/\sqrt{c}$, distances near the boundary grow exponentially, allowing more efficient packing of clusters within a bounded region while preserving separation. In Euclidean space, by contrast, one must increase the radius r significantly to maintain cluster separation, resulting in less efficient packing.

Motivated by this property, we formulate the Vector Quantization (VQ) operation as a multinomial logistic regression (MLR) directly in hyperbolic space. Specifically, rather than assigning latent vectors to the nearest codebook entry in a Euclidean sense, we perform classification in $\mathbb{P}_c^n$ using hyperbolic MLR. To further promote disentanglement, we propose using the representative points of hyperbolic decision hyperplanes as

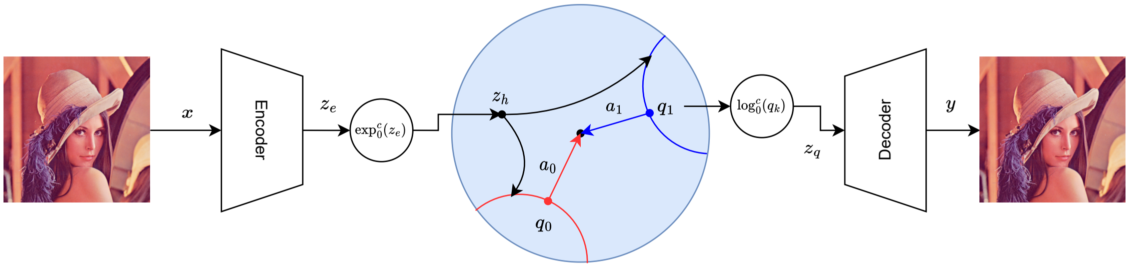


Figure 2: Illustration of HyperVQ in a VQVAE setting. The key steps are the projection of the euclidean embeddings to the hyperbolic space, classification of the projected embeddings to one of K classes, and utilization of the representative point on the hyperplane, q_k as the codebook vector which is projected back to the Euclidean space.

codebook vectors. Unlike an arbitrarily learned embedding matrix, these representative points capture the geometric structure more faithfully and encourage distinct separations in the latent space.

Figure 2 illustrates the overall pipeline of an encoder-decoder framework augmented with our HyperVQ. The encoder and decoder remain in Euclidean space, while the codebook lookup (i.e., quantization) occurs in $\mathbb{P}_c^n$. Algorithm 1 provides a task-agnostic training procedure, and Section 4.2 details the hyperbolic MLR formulation and the geometric constraints that define our codebook vectors. Different tasks may employ diverse encoder-decoder architectures, but the quantization mechanism in hyperbolic space stays the same, exploiting the exponential volume growth to achieve compact and efficient clustering of embeddings.

In the following subsections, we provide a detailed description of HyperVQ.

4.2 HyperVQ

4.2.1 Hyperbolic MLR for Vector Quantization

Given an encoder output $z_e \in \mathbb{R}^d$, we first obtain its hyperbolic embedding $z_h = \exp_0^c(z_e)$, where $\exp_0^c(\cdot)$ is the exponential map at the origin (Equation (1)). We then treat the codebook selection problem as a classification (Baevski et al. (2020a)) in $\mathbb{P}_c^n$ via hyperbolic multinomial logistic regression (MLR). Let $\ell_k(z_h)$ be the logit for codebook index $k \in \{1, 2, \ldots, K\}$. To assign z_h to one of the K codebook vectors, we apply a Gumbel Softmax over these logits:

$$p_k(z_h) = \frac{\exp\big((\ell_k(z_h) + g_k)/\tau\big)}{\sum_{j=1}^{K} \exp\big((\ell_j(z_h) + g_j)/\tau\big)}, \tag{13}$$

where $\{g_1, \ldots, g_K\}$ are i.i.d. samples from the Gumbel distribution Jang et al. (2017) and $\tau > 0$ is a temperature parameter. The codebook index is chosen by

$$k^* = \arg\max_k \; p_k(z_h). \tag{14}$$

To enable end-to-end training, we use the straight-through estimator Bengio et al. (2013) when backpropagating through the discrete choice. This ensures gradients pass from the decoder's reconstruction loss (or downstream objectives) back to the hyperbolic logits $\ell_k(\cdot)$. We rely on the unidirectional MLR formulation (discussed in Equation (6)) for calculating these logits in a parameter-efficient manner.

4.2.2 Geometrically Constrained Codebook Vectors

The selection of codebook vectors is a critical factor in vector quantization. Traditional methods include choosing cluster centroids via online K-Means (as in standard VQVAE) or learning an embedding matrix (as in Gumbel Vector Quantization). While K-Means centroids reflect intrinsic geometry in Euclidean space, direct extensions to hyperbolic space require iteratively computing central measures (Lou et al., 2020), which is computationally expensive and numerically sensitive near the Poincaré ball boundary. On the other hand, embedding matrices optimized purely by a reconstruction objective may overfit to input patterns, reducing diversity among codebook entries and weakening disentanglement.

To overcome these limitations, we assign each codebook entry to be the representative point of a hyperbolic decision hyperplane, denoted by q_k. In particular, q_k encodes both the position and orientation of the k-th hyperplane in $\mathbb{P}_c^n$ (see Section 3.1.4 for its unidirectional definition). Unlike a centroid-based approach that averages embeddings, q_k remains stable under outliers and preserves local geometric structure. Moreover, the decision hyperplane formulation naturally enforces distinct regions in the latent space, improving disentanglement across different codebook entries.

Since the decoder typically operates in Euclidean space, we apply the logarithmic map $\log_0^c$ to project q_k from $\mathbb{P}_c^n$ back to $\mathbb{R}^d$. Referring to equation 4 for $q_k = \exp_0^c(r_k[a_k])$, we have

$$z_q \;=\; \log_0^c\big(q_k\big) \;=\; \log_0^c\Big(\exp_0^c\big(r_k[a_k]\big)\Big) \;=\; r_k\,[a_k], \tag{15}$$

where r_k is a scalar and $a_k \in \mathbb{R}^n$ represents the normal vector in the tangent space. In essence, each hyperbolic codebook vector in Euclidean space becomes the product $r_k[a_k]$. This mapping incorporates the hyperbolic geometry (via r_k and a_k) without requiring computationally heavy centroid calculations.

An important consequence of using q_k is that the decoder is forced to reconstruct from geometrically meaningful and relatively disentangled features. As a result, each codebook entry captures a distinct direction and radius in the tangent space, and the exponential growth property of $\mathbb{P}_c^n$ helps maintain clear separations between codebook vectors. This structure not only mitigates codebook collapse but also enhances discriminative performance, since the latent space is partitioned by well-defined hyperplanes rather than overlapping or redundant embedding vectors.

Algorithm 1 HyperVQ Training

```
Initialization: Initial network parameters of Encoder ($\theta_E$), Decoder ($\theta_D$), and Quantizer ($\theta_Q = \{a_k, r_k\}$ for $k = 0, 1, \ldots, K$ ) and temperature, $\tau \in [\tau_{\max}, \tau_{\min}]$, and decay factor $\delta$
Training:
for each iteration, $j$ do
    sample data, $x$
    $z_e = \text{Encoder}(x)$
    $z_h = \exp_0^c(z_e)$
    $logits = \text{unidirectional\_mlr}(z_h)$ (Equation (6))
    $k = \text{argmax}(\text{gumbel\_softmax}(logits, \tau))$
    $z_q = r_k[a_k]$ (Equation (15))
    $\tau = \max\left(\tau_{\max} \cdot \delta^j, \tau_{\min}\right)$
    $y = \text{Decoder}(z_q)$
    Update $\theta_E, \theta_D, \theta_Q$ by gradient descent
end for
```

5 Experiments

To assess the effectiveness of our proposed method, we conducted experiments across diverse tasks, encompassing image reconstruction, generative modeling, image classification, and feature disentanglement. The implementation details and results of each experiment are elaborated upon in the following subsections.

5.1 Implementation Details

Our method was implemented using the PyTorch library Paszke et al. (2019). Model training was conducted on 4 A100 GPUs utilizing the Adam optimizer Kingma & Ba (2014), with a learning rate set at

3e-4 and a batch size of 128 per GPU, unless otherwise specified. For hyperbolic functions, we employed the geoopt library Kochurov et al. (2020) and referenced the official implementation of hyperbolic neural networks++ Shimizu et al. (2021).

5.2 Reconstruction and Generative Modeling

To assess the generative modeling capabilities of the proposed HyperVQ method, we conducted experiments on the Cifar100 Krizhevsky (2009) and ImageNet Russakovsky et al. (2015) datasets. We trained VQVAEs with the original formulation, referred to as KmeansVQ, and our proposed hyperbolic formulation, termed as HyperVQ, for varying numbers of codebook vectors K. The encoder comprises an initial convolutional encoder block, followed by a stack of residual blocks, before being passed through the quantization layer. The decoder is the inverse of the encoder, with the residual blocks and the convolutional encoder block replaced by a transposed convolutional decoder block for upsampling. The reconstruction results are presented in Table 1. By comparing the reconstruction mean squared error (MSE) of the two methods, we observed that our proposed method performs on par with the original formulation. Following this, we trained a generative 15-layer GatedPixelCNN Van den Oord et al. (2016) model using the quantized representations obtained from VQVAE models trained on the ImageNet dataset with $K = 512$ codebook vectors for 50 epochs.

Table 1: Comparison of reconstruction mean squared error (MSE) for KmeansVQ and HyperVQ on Cifar100 and ImageNet datasets for different codebook sizes, K.

	Cifar100 (32×32) ↓			ImageNet (128×128) ↓		
K	512	256	128	512	256	128
KmeansVQ	0.264	0.264	0.250	0.175	0.199	0.225
HyperVQ	0.216	0.241	0.256	0.175	0.202	0.217

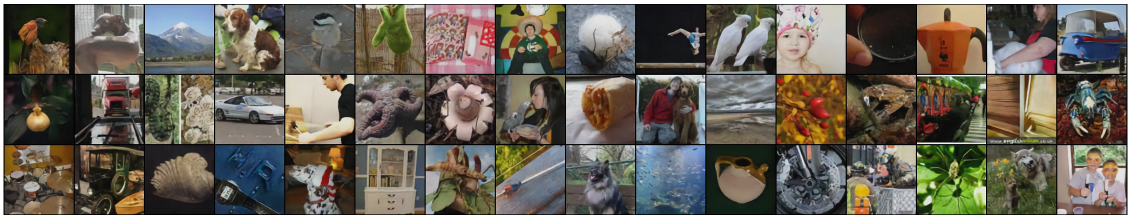
Table 2: Generative modeling performance of GatedPixelCNN with VQVAE quantized representations on ImageNet, comparing HyperVQ and the original formulation in terms of Fréchet Inception Distance (FID) and Inception Score (IS)

	FID ↓	IS ↑
KmeansVQ	143.78	5.50 ± 0.07
HyperVQ	**130.31**	**5.77 ± 0.05**

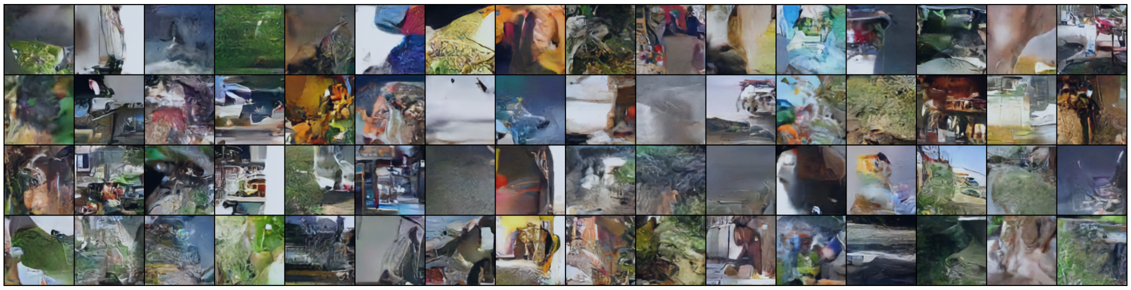
As shown in Table 2, HyperVQ demonstrates comparative generative modeling performance compared to the original formulation, measured by the Fréchet Inception Distance (FID)Heusel et al. (2017) and Inception Score (IS)Salimans et al. (2016). Additionally, Figure 3a displays some reconstructions achieved with HyperVQ on the ImageNet validation set (128×128), while Figure 3b presents samples generated by the GatedPixelCNN model. We reiterate here, that he goal of this experiment is to show the effectiveness of HyperVQ and not to achieve state-of-the-art.

5.3 Image Classification

To substantiate our claim that HyperVQ learns a more disentangled representation, thereby enhancing discriminative performance, we conducted experiments on the Cifar100 dataset. In these experiments, we initially trained VQVAEs with various quantization methods and a varying number of codebook vectors, K. Subsequently, we employed the pre-trained VQVAE as a feature extractor and trained a lightweight classifier atop it, aconsisting of a convolutional layer with ReLU activation, followed by a global average pooling layer and two fully connected layers to obtain the logits. Only the classifier block is trained using the cross-entropy loss, while keeping the parameters of the convolutional encoder, residual blocks, and the quantization layer



(a) HyperVQ reconstruction



(b) Samples from GatedPixelCNN trained on top of HyperVQ encodings

Figure 3: Image reconstruction and generative modeling results on ImageNet.

fixed. The goal of this experiment is to show the effectiveness of HyperVQ embeddings and not to achieve state-of-the-art accuracy.

For this experiment, in addition to KmeansVQ and HyperVQ, we also included GumbelVQ for comparison. All models underwent training for 500 epochs, utilizing the same settings as described in Section 5.1.

Table 3: Evaluation of discriminative performance using pre-trained encoder and quantizer of a VQ-VAE as feature extractors on the Cifar100 test set.

K	512	256	128	64
KmeansVQ	30.04	29.54	30.23	29.58
GumbelVQ	25.99	26.93	26.94	27.45
HyperVQ	**31.59**	**31.06**	**30.63**	**30.14**

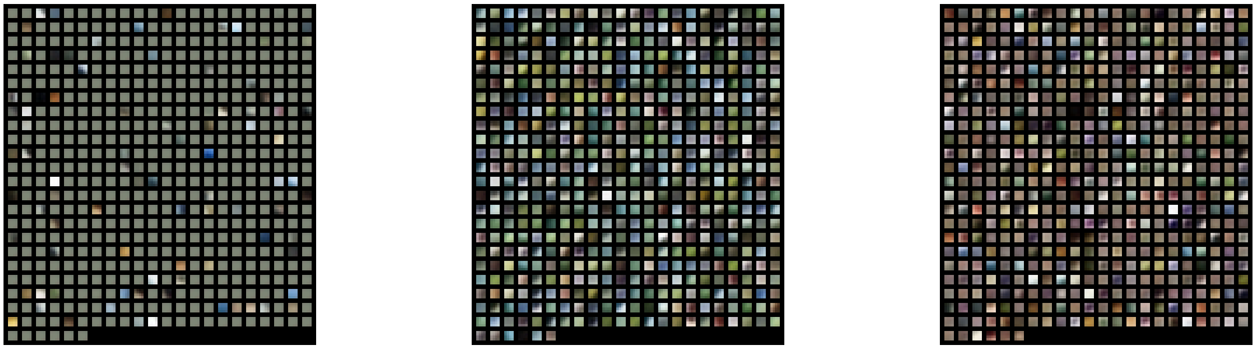
Table 4: Evaluation of codebook usage of the different quantization methods, as measured by average perplexity over the Cifar100 test set.

	Perplexity ($K = 512$)
KmeansVQ	58.09
GumbelVQ	169.59
HyperVQ	**220.22**

The results presented in Table 3 indicate that HyperVQ consistently outperforms other methods in terms of classification accuracy, irrespective of the number of codebook vectors.

To gain insights into the reasons behind this improved performance, we visualize the codebook vectors learned by different methods in Figure 4 and compare the codebook usage for reconstruction based on the perplexity on the test set, as shown in Table 4. We treat the codebook vector as a latent representation with 1×1 spatial dimensions and decode it using the corresponding pretrained VQVAE decoder for visualization purposes.

Insights drawn from the codebook visualization and perplexities yield several interesting observations. Firstly, it is evident that KmeansVQ exhibits low codebook usage, as illustrated in Figure 4a, where a substantial portion of the codebook vectors appears to be invalid and unused. While GumbelVQ demonstrates higher codebook usage compared to KmeansVQ and learns valid codebook vectors, the vectors themselves lack strong discriminative qualities. This limitation stems from their optimization for reconstruction, leading to some redundancy in the codebook visualizations (Figure 4b), ultimately resulting in lower classification accuracy compared to KmeansVQ.



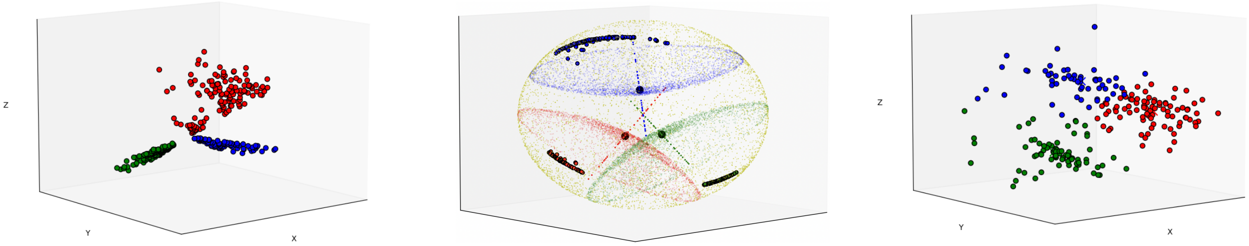
(a) KmeansVQ (b) GumbelVQ (c) HyperVQ

Figure 4: Visualization of the codebook vectors for VQVAE trained on Cifar100 with different quantization methods for $K = 512$.

Contrastingly, the codebook vectors learned by HyperVQ exhibit both the highest perplexity (Table 4) and less redundancy (Figure 4c). This characteristic enhances the discriminative ability of the quantized embeddings, translating into the highest classification accuracy among the methods considered.

5.3.1 Disentanglement and Cluster Quality

To assess whether HyperVQ is capable of learning a more disentangled representation, we trained a HyperVQVAE model on the simple MNIST dataset. We maintained the latent dimension as 3, with 16 codebook vectors, for easy visualization without any post-processing. For comparison, we also trained a KmeansVQ-VAE model with the same configuration. To facilitate visualization, we sampled 1000 embeddings and plotted them against 3 codebook vectors each for HyperVQ and KmeansVQ.



(a) HyperVQ (z_e) (b) HyperVQ (z_h,$\bar{H}^c_{a_k,r_k}$, and z_q) (c) KmeansVQ (z_e)

Figure 5: Vizualization and comparison of the embeddings learnt by the KmeansVQVAE and HyperVQVAE models on MNIST dataset. Figure 5a shows the pre-pojection Euclidean embeddings learnt by the HyperVQ, and Figure 5b shows the projected embeddings along with the decision hyperplanes and quantized vectors. Figure 5c shows the embeddings learnt by the KmeansVQ model,

Table 5: Cluster quality and robustness assessment under standard and noisy conditions, comparing Silhouette score and Davies-Bouldin Index for KmeansVQ and HyperVQ.

	Standard Conditions		Noisy Conditions	
	Silhouette Score↑	DB Index↓	Silhouette Score↑	DB Index↓
KmeansVQ	0.487	0.732	0.310	0.915
HyperVQ	**0.565**	**0.553**	**0.564**	**0.560**

As depicted in Figure 5, the clusters of the HyperVQVAE are more compact and disentangled than those of the VQVAE. This phenomenon arises because the hyperbolic MLR formulation encourages embeddings to be highly localized within the regions enclosed by the decision hyperplanes, thus inducing implicit disentanglement of the latent space.

Apart from the visualization, we also performed cluster quality assessment by computing the Silhoutte score Rousseeuw (1987) and Davies-Bouldin Index Davies & Bouldin (1979). To test the robustness, we applied random noise in the form of random rotations, flip, and gaussian noise and computed the above scores. From Table 5, we can see that HyperVQ is more robust towards noisy inputs and maintains its cluster compactness even under noisy conditions and in general performs better than KmeansVQ.

5.4 Quality Independent Representation Learning

We also applied HyperVQ to the VQ-SA model Yang et al. (2023). This model introduces a VQ layer and a self-attention layer, combining quantized and unquantized features before the final classifier of a pretrained backbone network. This augmentation aims to enhance robustness to corruptions by learning quality-independent features. The VQ-SA model undergoes training on the ImageNet dataset, incorporating corruptions from the ImageNet-C dataset Hendrycks & Dietterich (2019) for data augmentation. In the original work, they employed $K = 10000$ and trained for 300 epochs. In our case, to conserve computational resources, we used $K = 1024$ and trained for 50 epochs. We also trained the original VQ-SA model under the same settings for fair comparison. Classification accuracy is reported on the clean validation set of ImageNet, along with accuracy on known and unknown corruptions in the ImageNet-C dataset. Additionally, we report the mean corruption error (mCE) Hendrycks & Dietterich (2019). From the Table 6, we can see that the hyperVQ-SA method significantly outperforms the original VQ-SA

Table 6: Classification accuracy on ImageNet and ImageNet-C for the VQ-SA method with different quantization methods.

	Clean ↑	Known ↑	Unknown ↑	mCE ↓
ResNet50	**75.704**	37.39	48.94	76.43
VQ-SA	70.178	58.64	53.55	52.65
HyperVQ-SA	74.344	**62.30**	**56.09**	**47.61**

5.5 Application to Speech Self-Supervised Learning

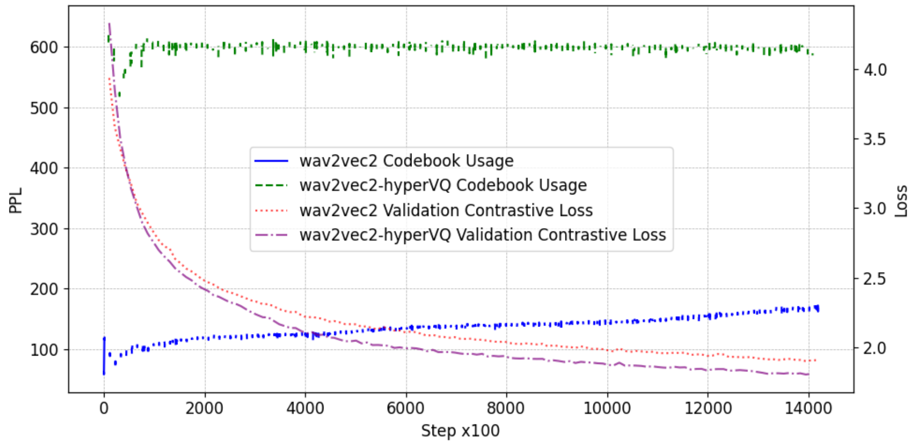


Figure 6: Application of hyperVQ to wav2vec 2.0 pre-training

To explore whether HyperVQ provides performance gains beyond the domain of computer vision, we applied it to the wav2vec-2.0 (Baevski et al., 2020b) pre-training framework. In wav2vec-2.0 pre-training, intermediate convolutional features are quantized, and the transformer backbone predicts the quantization IDs from masked continuous convolutional features. The model is optimized using a contrastive loss. In our experiments, we replaced the standard vector quantization module in wav2vec-2.0 with HyperVQ and conducted pre-training on the LibriSpeech 960h dataset. As shown in Figure 6, HyperVQ achieves consistently higher codebook usage, which is reflected in the higher perplexity and resulting in a lower validation contrastive loss, thereby demonstrating the generalizability and efficacy of our proposed method. These results indicate that HyperVQ's structured representation contributes to more efficient learning dynamics, reinforcing its applicability in diverse domains beyond vision.

5.6 Ablation Study

We conducted an ablation study to understand the individual and combined impacts of the hyperbolic Multinomial Logistic Regression (MLR) formulation and the use of the representative point on the decision hyperplane as the codebook vector in our proposed method.

Our evaluation compared the performance of our HyperVQ method against traditional vector quantization approaches such as KmeansVQ and GumbelVQ, as well as two variants designed for this study: HyperKmeansVQ, which uses codebook vectors in hyperbolic space learned with the KmeansVQ mechanism and distances induced by the Poincaré ball model; and HyperEmbMatVQ, which applies the hyperbolic MLR formulation but uses the embedding matrix as the codebook vectors, similar to GumbelVQ but with logits derived from the hyperbolic MLR.

Table 7: Comparison of Classification Accuracy on CIFAR-100 for various vector quantization methods with a codebook size of 512.

	Accuracy (%)
KmeansVQ	30.04
HyperKmeansVQ	22.42
GumbelVQ	25.99
HyperEmbMatVQ	28.28
HyperVQ	**31.59**

The findings from our study, as depicted in Table Table 7, highlight the benefits of incorporating the hyperbolic MLR formulation with the embedding matrix, which notably improves classification accuracy over the GumbelVQ model. Additionally, leveraging the representative point on the decision hyperplane as the codebook vector further enhances classification performance, establishing our HyperVQ method as the superior approach among those tested.

6 Limitations and Conclusion

We proposed an alternative formulation for vector quantization utilizing hyperbolic space, demonstrating its effectiveness in enhancing the discriminative power of quantized embeddings while preserving the generative capabilities of the VQVAE. The hyperbolic MLR formulation was shown to encourage a more compact representation in the pre-quantized Euclidean latent space. Through various experiments, we illustrated that HyperVQ can serve as a drop-in replacement for multiple methods across domains, improving their performance.

However, it is important to note that the projections to and from the hyperbolic space exhibit sensitivity to numerical instability, particularly near the boundary of the Poincaré ball. This sensitivity raises concerns, especially when employing low-precision methods, such as 16-bit floating-point numbers, during model training, as it may lead to instability. Consequently, this limitation could restrict the application of HyperVQ in very large models, which are typically trained using low or mixed precision. Further analysis of the stability of HyperVQ under low-precision training is left for future work.

References

Mina Ghadimi Atigh, Julian Schoep, Erman Acar, Nanne Van Noord, and Pascal Mettes. Hyperbolic image segmentation. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 4453–4462, 2022.

Alexei Baevski, Steffen Schneider, and Michael Auli. vq-wav2vec: Self-supervised learning of discrete speech representations. In International Conference on Learning Representations, 2020a. URL `https://openreview.net/forum?id=rylwJxrYDS`.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. Advances in neural information processing systems, 33: 12449–12460, 2020b.

Yoshua Bengio, Nicholas Léonard, and Aaron C. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. CoRR, abs/1308.3432, 2013.

Katherine Crowson, Stella Biderman, Daniel Kornis, Dashiell Stander, Eric Hallahan, Louis Castricato, and Edward Raff. Vqgan-clip: Open domain image generation and editing with natural language guidance. In European Conference on Computer Vision, pp. 88–105. Springer, 2022.

David L. Davies and Donald W. Bouldin. A cluster separation measure. IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-1(2):224–227, 1979. doi: 10.1109/TPAMI.1979.4766909.

Karan Desai, Maximilian Nickel, Tanmay Rajpurohit, Justin Johnson, and Shanmukha Ramakrishna Vedantam. Hyperbolic image-text representations. In International Conference on Machine Learning, pp. 7694–7731. PMLR, 2023.

Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 12873–12883, 2021.

Octavian Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic neural networks. Advances in neural information processing systems, 31, 2018.

R. Gray. Vector quantization. IEEE ASSP Magazine, 1(2):4–29, 1984. doi: 10.1109/MASSP.1984.1162229.

Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image synthesis. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10696–10706, 2022.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. Proceedings of the International Conference on Learning Representations, 2019.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems, 30, 2017.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In International Conference on Learning Representations, 2017. URL `https://openreview.net/forum?id=rkE3y85ee`.

Valentin Khrulkov, Leyla Mirvakhabova, Evgeniya Ustinova, Ivan Oseledets, and Victor Lempitsky. Hyperbolic image embeddings. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6418–6428, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

Max Kochurov, Rasul Karimov, and Serge Kozlukov. Geoopt: Riemannian optimization in pytorch, 2020.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Master's thesis, Department of Computer Science, University of Toronto, 2009.

Adrian Łańcucki, Jan Chorowski, Guillaume Sanchez, Ricard Marxer, Nanxin Chen, Hans JGA Dolfing, Sameer Khurana, Tanel Alumäe, and Antoine Laurent. Robust training of vector quantized bottleneck models. In 2020 International Joint Conference on Neural Networks (IJCNN), pp. 1–7. IEEE, 2020.

Aaron Lou, Isay Katsman, Qingxuan Jiang, Serge Belongie, Ser-Nam Lim, and Christopher De Sa. Differentiating through the fréchet mean. In International Conference on Machine Learning, pp. 6393–6403. PMLR, 2020.

Ruiying Lu, YuJie Wu, Long Tian, Dongsheng Wang, Bo Chen, Xiyang Liu, and Ruimin Hu. Hierarchical vector quantized transformer for multi-class unsupervised anomaly detection. In Thirty-seventh Conference on Neural Information Processing Systems, 2023. URL `https://openreview.net/forum?id=clJTNssgn6`.

Emile Mathieu, Charline Le Lan, Chris J Maddison, Ryota Tomioka, and Yee Whye Teh. Continuous hierarchical representations with poincaré variational auto-encoders. Advances in neural information processing systems, 32, 2019.

Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. Advances in neural information processing systems, 30, 2017.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc., 2019. URL `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`.

Wei Peng, Tuomas Varanka, Abdelrahman Mostafa, Henglin Shi, and Guoying Zhao. Hyperbolic deep neural networks: A survey. IEEE Transactions on pattern analysis and machine intelligence, 44(12):10023–10044, 2021.

Darius Petermann, Gordon Wichern, Aswin Subramanian, and Jonathan Le Roux. Hyperbolic audio source separation. In ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1–5. IEEE, 2023.

Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics, 20:53–65, 1987. ISSN 0377-0427. doi: https://doi.org/10.1016/0377-0427(87)90125-7. URL `https://www.sciencedirect.com/science/article/pii/0377042787901257`.

Aurko Roy, Ashish Vaswani, Arvind Neelakantan, and Niki Parmar. Theory and experiments on vector quantized autoencoders. CoRR, abs/1805.11063, 2018.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. International Journal of Computer Vision (IJCV), 115(3):211–252, 2015.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. Advances in neural information processing systems, 29, 2016.

Ryohei Shimizu, YUSUKE Mukuta, and Tatsuya Harada. Hyperbolic neural networks++. In International Conference on Learning Representations, 2021. URL `https://openreview.net/forum?id=Ec85b0tUwbA`.

Ondrej Skopek, Octavian-Eugen Ganea, and Gary Bécigneul. Mixed-curvature variational autoencoders. In 8th International Conference on Learning Representations (ICLR 2020)(virtual). International Conference on Learning Representations, 2020.

Yuhta Takida, Takashi Shibuya, WeiHsiang Liao, Chieh-Hsin Lai, Junki Ohmura, Toshimitsu Uesaka, Naoki Murata, Shusuke Takahashi, Toshiyuki Kumakura, and Yuki Mitsufuji. Sq-vae: Variational bayes on discrete representation with self-annealed stochastic quantization. arXiv preprint arXiv:2205.07547, 2022.

Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. Advances in neural information processing systems, 29, 2016.

Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. Advances in neural information processing systems, 30, 2017.

Zhou Yang, Weisheng Dong, Xin Li, Mengluan Huang, Yulin Sun, and Guangming Shi. Vector quantization with self-attention for quality-independent representation learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 24438–24448, 2023.

Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. Soundstream: An end-to-end neural audio codec. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 30: 495–507, 2021.