Codebook Transfer with Part-of-Speech for Vector-Quantized Image Modeling

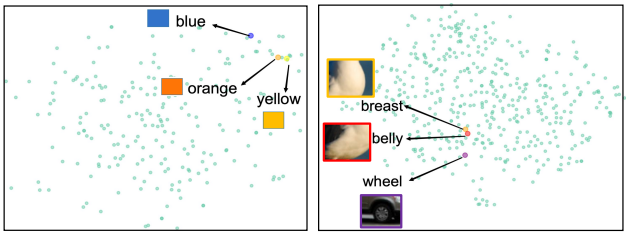
Baoquan Zhang[1], Huaibin Wang[1], Chuyao Luo[1], Xutao Li[1,3], Guotao Liang[1,3], Yunming Ye*[1,3], Xiaochen Qi[2], Yao He[2]
[1] Harbin Institute of Technology, Shenzhen; [2] ShenZhen SiFar Co., Ltd.; [3] Peng Cheng Laboratory
baoquanzhang@hit.edu.cn, 22S051022@stu.hit.edu.cn, luochuyao.dalian@gmail.com,
lianggt@pcl.ac.cn, {lixutao, yeyunming}@hit.edu.cn, {joeqxc1974, heyao18818}@gmail.com

Abstract

Vector-Quantized Image Modeling (VQIM) is a fundamental research problem in image synthesis, which aims to represent an image with a discrete token sequence. Existing studies effectively address this problem by learning a discrete codebook from scratch and in a code-independent manner to quantize continuous representations into discrete tokens. However, learning a codebook from scratch and in a code-independent manner is highly challenging, which may be a key reason causing codebook collapse, i.e., some code vectors can rarely be optimized without regard to the relationship between codes and good codebook priors such that die off finally. In this paper, inspired by pretrained language models, we find that these language models have actually pretrained a superior codebook via a large number of text corpus, but such information is rarely exploited in VQIM. To this end, we propose a novel codebook transfer framework with part-of-speech, called VQCT, which aims to transfer a well-trained codebook from pretrained language models to VQIM for robust codebook learning. Specifically, we first introduce a pretrained codebook from language models and part-of-speech knowledge as priors. Then, we construct a vision-related codebook with these priors for achieving codebook transfer. Finally, a novel codebook transfer network is designed to exploit abundant semantic relationships between codes contained in pretrained codebooks for robust VQIM codebook learning. Experimental results on four datasets show that our VQCT method achieves superior VQIM performance over previous state-of-the-art methods.

1. Introduction

With the development of multi-modal learning on representation and generation, unifying all modality with transformer has attracted increasing interest on computer vision and multi-modal domains [9, 37, 41]. It is well-known that transformer [28] is proposed for modeling discrete token sequence data like text, which is very difficult to directly apply on continuous image data. To address this problem, Vector-Quantized Image Modeling (VQIM) is proposed and has received wide attention recently. VQIM, as a fundamental problem in machine learning, aims to encode an image with a discrete token sequence similar to texts [20, 27].



(a) Adjective Codebook Space. (b) Noun Codebook Space.

Figure 1. Existing language models actually have provided a superior codebook, which contains abundant semantic relationships between codes and where some vision-related tokens (adjective and noun) can be well transferred to VQIM. For example, the vision-similar "orange" and "yellow" are indeed closer than dissimilar "yellow" and "blue" in adjective space (a); In noun space (b), some vision-similar parts like "breast" and "belly" are also closer than dissimilar "breast" and "wheel". Resorting to such relationships, VQIM codebook collapse can be alleviated, *e.g.*, although the "orange" code is not selected to optimize, but its code vector can also be well learned with its close relationship to the "yellow" code.

Existing studies [5, 7, 27, 36, 38] effectively address this VQIM problem by learning a discrete codebook from scratch and in a code-independent manner to quantize continuous feature representation into a discrete token sequence. For example, in [27], Oord et al. propose an encoder-decoder network (called VQ-VAE) to learn and quantize a latent feature space by selecting its nearest neighbor in the codebook as the discrete vector (*i.e.*, token) and training the VQ-VAE model by a simple reconstruction loss. Esser et al. [7] further enhance the VQ-VAE by additionally introducing an adversarial loss. Although these methods have shown superior performance on VQIM, they suffer from a codebook collapse issue [25], *i.e.*, only a few code vectors perform optimization during training, whereas a majority of them are never updated (*i.e.*, "die off"). This

*Corresponding author.

limits the VQIM performance in these existing methods. To address this codebook collapse issue, recently, some new VQIM techniques are developed from the perspective of codebook update [4, 24, 31, 38], quantization [1, 26, 29], or regularization [36] for robust codebook learning.

In this paper, we also focus on robust codebook learning but present a new perspective (*i.e.*, codebook transfer) for alleviating codebook collapse. Our insight is that neglecting the relationship between code vectors and codebook priors to learn a discrete codebook from scratch is actually very difficult, which may be a key reason causing codebook collapse, *i.e.*, some code vectors can rarely be optimized without regard to the relationship between codes and good codebook priors such that die off finally. Inspired by recent pretrained language models [14, 22, 23], we find that some superior codebooks have been well pretrained in some models (*e.g.*, CLIP [23] and GloVe [21]) and these code vectors in the codebook are not fully independent, which contains abundant transferable relationships between codes from language to vision [32, 35]. For example, as shown in Figure 1, 1) the vision-similar "orange" and "yellow" are indeed closer than vision-dissimilar "yellow" and "blue" in adjective codebook space; and 2) two parts ("breast" and "belly") with similar vision are also closer than "belly" and "wheel" with dissimilar vision in noun codebook space. Resorting to such transferable relationships, VQIM codebook can be well learned by code cooperative optimization for alleviating the codebook collapse, *e.g.*, although the "orange" code is not selected to optimize, but its code vector can also be well learned by resorting to its relationship with the "yellow" code (see Figure 3 for more details).

Based on this idea, we propose a novel codebook transfer framework with part-of-speech, which transfers the abundant semantic knowledge of codebook from pretrained language models in order to enhance VQIM codebook learning, called VQCT. Specifically, we first introduce a pretrained language model and part-of-speech knowledge (e.g., WordNet) as priors. Then, we construct a set of vision-related codebook (*i.e.*, adjective and noun codebooks) from the pretrained codebook of language models by filtering out vision-unrelated tokens according to their part-of-speech. Finally, a novel graph convolution-based codebook transfer networks is designed to model the VQIM codebook in a transfer mapping manner from language to vision, which aims to resort to the abundant knowledge from pretrained codebooks to enhance the VQIM codebook learning. The advantage of such design is that 1) the abundant knowledge from well-pretrained codebooks can be fully exploited for providing good codebook priors; and 2) our codebook is generative but not directly learnable such that the semantic relationships between codes can be fully exploited for achieving cooperative optimization between codes.

Our main contributions can be summarized as follows:

- We propose a new perspective, *i.e.*, codebook transfer from language models to VQIM, to alleviate the codebook collapse issue. Its advantage is that the abundant transferable relationships from language codebooks can be fully exploited for enhancing codebook learning.
- Resorting to part-of-speech knowledge, we construct a set of vision-related codebooks (i.e., adjective and noun codebooks) and design a novel graph convolution-based codebook transfer network. In particular, our codebook is generative rather than directly optimized. Its advantage is that cooperative optimization between codes can be achieved for alleviating codebook collapse issue.
- We conduct comprehensive experiments on four datasets, which verify the effectiveness of our VQCT method.

2. Related Works

2.1. Vector-Quantized Image Modeling

Vector-Quantized Image Modeling (VQIM) is a challenging machine learning task, which aims to encode an image with a discrete token sequence like a text token sequence [3, 6, 10, 13, 15, 19, 39]. To address this problem, a large number of VQIM methods have been proposed, which aim to learn a discrete codebook from scratch and in a code-independent manner to quantize continuous representation into a discrete token sequence [7, 27, 36, 38]. Specifiaclly, Oord et al. [27] first achieve a VQIM model, called VQ-VAE, by following the framework of Variational Auto-Encode (VAE) and replacing its prior dirtribution with a discrete deterministic distribution (i.e., a codebook). Based on the superiority of VQ-VAE, a large number of studies [7, 12, 33, 36, 38] further improve its VQIM performance and produce a series of VQ-VAE variants. For example, in [7], Esser et al. further propose a VQ-GAN by introducing an adversarial training loss to enhance the generation quality of VQ-VAE. Yu et al. [33] design a vision transformer (ViT) to encode image representations for improving the modeling quality of convolution networks in VQIM. Huang et al. [12] propose a dynamic quantization VAE (DQ-VAE) for learning a compact code representation in a variable-length manner. Although these methods have shown superior performance, recent studies show that these methods suffer from a codebook collapse issue [36], *i.e.*, only a few codes perform optimization during training, whereas a majority of them are never updated (*i.e.*, "die off"). This limits the VQIM performance of these existing methods.

To address this issue, some new VQIM techniques [26, 36, 38] are proposed from the perspective of codebook update, quantization, or regularization. For example, in [26], a stochastically quantized variational autoencoder (SQ-VAE) is proposed from the perspective of quantization, which alleviates the codebook collapse issue in a self-annealed stochastic quantization manner. Zheng et al. [38] develop an online codebook learning strategy from the per-

spective of codebook update, which aims to restart these unused code vectors into activation status. Zhang et al. [36] design a regularized vector quantization strategy to mitigate codebook collapse issues, *i.e.*, introducing a prior distribution to regularize the codebook utilization for VQIM. In this paper, we also focus on learning a robust codebook for VQIM. However, different from these existing methods, we present a new codebook transfer perspective to address the codebook collapse issue, which effectively exploits abundant transferable relationships between codes to achieve cooperative optimization between codes.

2.2. Pretrained Language Models

Pretrained Language Models (PLM) [14] have experienced tremendous success with a large number of text corpus, which aims to learn a language understanding and generation model in a self-supervised manner [2]. Early models, such as Word2Vector [18] and GloVe [21], laid the foundation for this progress, which focuses on learning a good representation in a self-supervised manner for each word (*i.e.*, token) and then leveraging it to understand/generate text data. With the superiority of transformer, a large number of end-to-end pretained language or vision-language models, e.g., BERT [14], GPT [22], CLIP [23], and ImageBind [9], are proposed with a large number of text or text-image pair corpora, which demonstrates significant breakthroughs across numerous language or multi-modal tasks. For example, in [23], Radford et al. propose a Contrastive Language-Image Pre-Training (CLIP) to learn a robust representation for each text and image. In [14], Devlin et al. design a new language representation model (i.e., Bidirectional Encoder Representations from Transformers, called BERT), and then learn it in a masked language model manner.

In this paper, we find that the pretrained word embeddings (*i.e.*, tokens) from PLM have actually provided a good codebook prior and propose a codebook transfer framework to enhance VQIM codebook learning. We note that a concurrent work with our VQCT is SPAE [34]. However, different from SPAE that directly regarding the pretrained codebook from PLM as VQIM codebook, we target at transfering the pretrained codebook from PLM to VQIM, and carefully design a novel codebook transfer network with part-of-speech. Its advantage is abundant semantic relationships from pretrained codebooks can be fully exploited for cooperative optimization between codes.

3. Methodology

3.1. Preliminaries: VQ-VAE

VQ-VAE [27], as a pioneering work on VQIM, aims to learn an encoder-decoder network and a discrete codebook to quantize an image into a discrete token sequence. Formally, let $f_{\theta_e}(\cdot)$ with parameter θ_e, $f_{\theta_d}(\cdot)$ with parameter θ_d, and $\mathcal{C} = \{(k, e_k \in \mathrm{R}^{n_c})\}_{k=1}^{K}$ with parameter $\{e_k \in \mathrm{R}^{n_c}\}_{k=1}^{K}$ denote the encoder, decoder, and discrete codebook, respectively. Among them, K is the size of the codebook, e_k denotes the $k-$th code vector in the codebook, and n_c is the vector dimension. Given an image $x \in \mathrm{R}^{W\times H\times C}$, we first leverage the encoder $f_{\theta_e}(\cdot)$ to encode it into a set of continuous feature vectors $Z \in \mathrm{R}^{w\times h\times n_c}$, *i.e.*, $Z = f_{\theta_e}(x)$. Then, a quantization operation $q(\cdot)$ is employed to quantize each continuous feature vector $z \in Z$ into a discrete code sequence, *i.e.*, selecting its nearest neighbor in the codebook $\mathcal{C}$ as its discrete code sequence D^q. That is,

$$D^q = q(Z, \mathcal{C}) = \arg \min_{k\in[0,K-1]} \|z - e_k\|_2^2. \tag{1}$$

As a result, a quantized feature Z^q can be obtained, *i.e.*, $Z^q = e_{D^q}$. After that, the quantized feature vector Z^q is fed into the decoder $f_{\theta_d}(\cdot)$ to reconstruct the origin image, *i.e.*, $\hat{x} = f_{\theta_d}(Z^q)$. Finally, the encoder $f_{\theta_e}(\cdot)$, decoder $f_{\theta_d}(\cdot)$, and codebook $\mathcal{C} = \{(k, e_k \in \mathrm{R}^{n_c})\}_{k=1}^{K}$ are jointly learned by minimizing the following loss objective:

$$\begin{aligned} L =& \|x - \hat{x}\|_2^2 + \|sg[f_{\theta_e}(x)] - Z^q\|_2^2 \\ &+ \|f_{\theta_e}(x) - \beta sg[Z^q]\|_2^2, \end{aligned} \tag{2}$$

where $sg[\cdot]$ is a stop-gradient operator and β is a hyper-parameter. The first term is reconstruction loss L_{rec}, and others are codebook loss L_{cod} which trains the codebook to represent continuous features. With the superiority of VQ-VAE, recent studies propose some varieties, e.g., VQ-GAN that introduce an adversarial loss L_{adv} to improve image generation quality of VQ-VAE.

Although these methods have shown superior performance, they perform optimization only for the active codes (*i.e.*, these codes D^q selected by the quantization operation $q(\cdot)$) and others remain unchanged. This results in that some code vectors may be never optimized such that die off finally, which is called a *codebook collapse issue*.

3.2. Overall Framework of VQCT

Recently, some studies also attempt to address the codebook collapse issue, but most existing methods focus on learning a codebook from scratch and in a code-independent manner. However, in this paper, our insight is neglecting the relationship between code vectors and codebook priors to learn a discrete codebook from scratch is actually very difficult, which may be a key reason causing codebook collapse. Based on this, we propose a novel graph convolution-based code transfer framework with part-of-speech for VQIM, called VQCT. The main idea is that instead of directly learning a codebook from scratch, we introduce a well-pretrained codebook from language models, and part-of-speech knowledge as priors, and then resort to the abundant semantics and relationships contained in these priors to enhance the VQIM codebook learning.

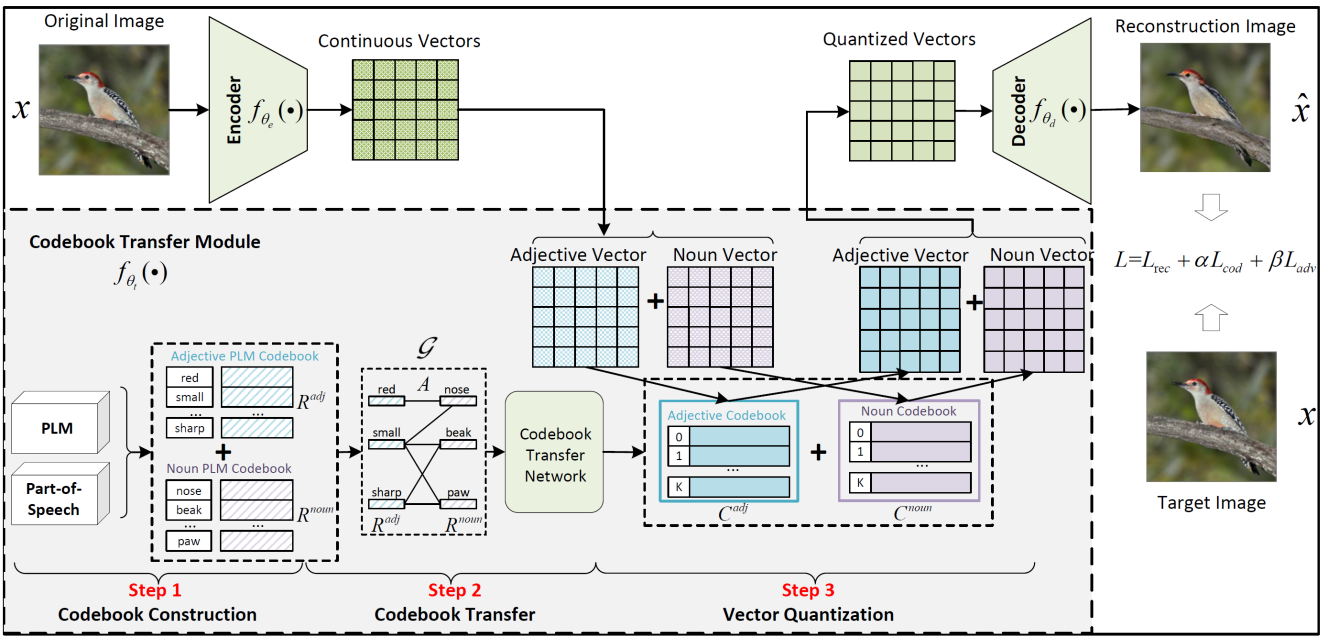


Figure 2. Illustration of our codebook transfer framework with part-of-speech, *i.e.*, VQCT, which consists of an encoder, a codebook transfer module, and a decoder. Here, the encoder aims to represent an image as a set of spatial continuous vectors. Then, the codebook transfer module is employed to generate a codebook in a transfer manner from pretrained language models (PLM) to VQIM and quantize the continuous vector into a set of quantized vectors. Finally, the decoder is used to reconstruct original images with the quantized vectors.

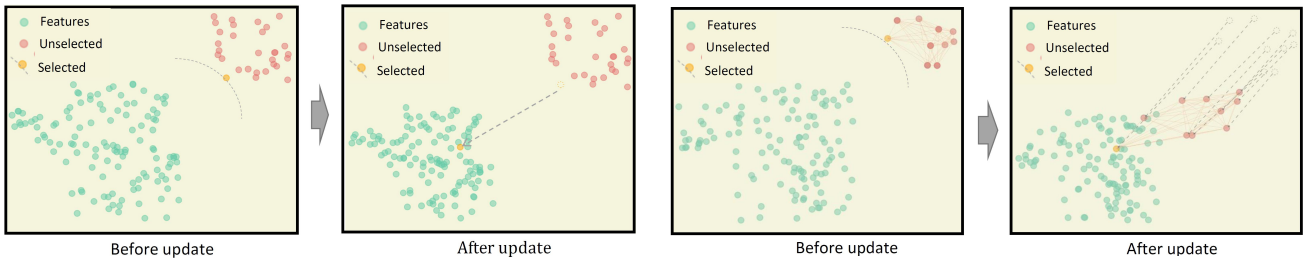
As shown in Figure 2, our VQCT framework consists of three modules, *i.e.*, an encoder $f_{\theta_e}(\cdot)$ with parameter θ_e, a codebook transfer module $f_{\theta_t}(\cdot)$ with parameter θ_t, and a decoder $f_{\theta_d}(\cdot)$ with parameter θ_d. Given an image x, the encoder $f_{\theta_e}(\cdot)$ is employed to encode continuous spatial vectors. Then, instead of directly learning a codebook from scratch, based on the pretrained codebook and part-of-speech priors, the codebook transfer module $f_{\theta_t}(\cdot)$ accounts for predicting a codebook in a transfer manner from language models to VQIM, which is then leveraged to quantize the continuous spatial representations into a discrete vectors. Finally, the decoder $f_{\theta_d}(\cdot)$ is used to reconstruct the original image $\hat{x}$. The workhorse of our VQCT is the codebook transfer network $f_{\theta_t}(\cdot)$. Next, we elaborate on them.

3.3. Codebook Transfer Module

Instead of directly learning a codebook from scratch, the codebook transfer module $f_{\theta_t}(\cdot)$ aims to generate a VQIM codebook in a transfer manner from pretrained language models. Its advantage is that abundant semantic relationships between codes can be fully exploited for cooperative learning between codes. Next, we introduce how to generate the VQIM codebook and quantize continuous vectors into quantized vectors, including the following three steps:

Step 1: ***Codebook Construction.*** In fact, some pretrained language models (e.g., CLIP [23], GloVe [21], or BERT [14]) have provided superior codebooks (*i.e.*, the embeddings of word tokens), but such information rarely be exploited. Our main idea is transferring these pretrained codebook to VQIM for enhancing VQIM codebook learning. However, transferring all word tokens to VQIM is impractical since the number of word tokens is very large, which imposes a considerable computational burden. Intuitively, a large number of words are all vision-unrelated (e.g., pron., adv., art., prep., or conj.), only few words are vision-related (e.g., adjective and noun). To this end, we first introduce a well-trained codebook from pretrained language models (PLM) and part-of-speech knowledge (e.g., WordNet) as priors. Then, based on the part-of-speech knowledge, we filter out all vision-unrelated words and only the adjective and noun are retained to construct the vision-related codebook, called "Adj. PLM Codebook" and "Noun. PLM Codebook", respectively. Let $R^{adj} = \{r_i^{adj}\}_{i=0}^{K_{adj}-1}$ and $R^{noun} = \{r_i^{noun}\}_{i=0}^{K_{noun}-1}$ denote the set of 'Adj PLM Codebook" and 'Noun PLM Codebook", respectively, where K_{adj} and K_{noun} is the code number.

Step 2: ***Codebook Transfer.*** The above "Adj. PLM codebook" R^{adj} and "Noun. PLM codebook" R^{noun} contains abundant semantics and relationships between word tokens. A simple approach is concatenating these two codebooks (R^{adj} and R^{noun}) as a whole codebook, and then directly regarding it as the VQIM codebook or employing a simple multilayer perceptron (MLP) to generate the VQIM codebook. However, the "Adj. PLM codebbok" R^{adj} and "Noun. PLM codebbok" R^{adj} are not independent, which are jointly used to describe a visual feature, *i.e.*, an adjective



(a) Distribution Change of VQ-VAE (b) Distribution Change of Our VQCT

Figure 3. Illustration of codebook optimization. Here, we take a two-dimensional toy setting as an example to show distribution change of codebook when performing optimization. Different from VQ-VAE (a) that only the active "lucky" seeds (in Peach) are optimized but the other "dead" vectors (in Red) are not optimized and remain fixed, our VQCT update all code vectors in the codebook, although only an active code vector is selected in the codebook to perform optimization, with the abundant semantic relationships from pretrained codebook.

is generally used to modify a noun such as "sharp beak".

To achieve this idea, as shown in Figure 2, we first construct a modifying graph from a large number text corpus to model the modifying relationships between the adjective and noun. Let $\mathcal{G} = (R, A)$ denote the constructed modifying graph where $R = \{R^{adj}, R^{noun}\}$ is node representation from pretrained PLM codebook and $A = \{A_{ij}\}$ denote the modifying relationship matrix between the adjective and noun. In the modifying relationship matrix A, $A_{ij} = 1$ if the noun j is modified with the adjective i; otherwise $A_{ij} = 0$. Then, we regard the modifying graph $\mathcal{G} = (R, A)$ as inputs and design a graph convolution-based codebook transfer network (GCCTN) $f_{\theta_t}(\cdot)$ to transfer the adjective and noun PLM codebooks to VQIM. That is,

$$(\mathcal{C}_{adj}, \mathcal{C}_{noun}) = f_{\theta_t}(\mathcal{G}), \quad \mathcal{G} = (R, A), \tag{3}$$

where the GCCTN network consists of three graph convolution layers, which is followed by ReLU activation in each layer, respectively. As a result, a set of VQIM codebooks (*i.e.*, an adjective codebook and a noun codebook) is obtained, which is denoted by $\mathcal{C}_{adj} = \{(k, e_k \in \mathrm{R}^{n_c})\}_{k=1}^{K_{adj}}$ and $\mathcal{C}_{noun} = \{(k, e_k \in \mathrm{R}^{n_c})\}_{k=1}^{K_{noun}}$, respectively.

Step 3: ***Vector Quantization.*** Given an image x, we first leverage the encoder $f_{\theta_e}(\cdot)$ to obtain its continuous vectors, which is divided into two parts along its channels, *i.e.*, an adjective vector and a noun vector. Then, based on the adjective codebook $\mathcal{C}_{adj}$ and noun codebook $\mathcal{C}_{noun}$, we emply a quantization operation $q(\cdot)$ to quantize each continuous adjective/noun vector into a discrete code sequence, respectively, *i.e.*, selecting its nearest neighbor in the adjective codebook $\mathcal{C}_{adj}$ and noun codebook $\mathcal{C}_{noun}$ as its discrete code sequence (see Eq. 1), respectively. After that, a set of quantized adjective vectors and noun vectors can be obtained, which is futher concatenated as quantized vectors. Finally, we feed the quantized vectors into the decoder $f_{\theta_d}(\cdot)$ for reconstructing the origin image $\hat{x}$.

Note that our VQCT only focus on codebook learning, which can be easily integrated into some existing VQIM methods. For example, our VQCT will become 1) VQCT-VQ-VAE when we adopt Eq. 3 to train our model; and 2) VQCT-VQ-GAN by adding an adversarial loss on 1). In next sections, we take VQCT-VQ-GAN as the main method to conduct experiments due to its superiority on image synthesis, which is simply called as VQCT.

3.4. Analysis of Codebook Optimization

To analyze how our VQCT alleviate codebook collapse issue, we take a two-dimensional codebook setting as a toy example and then visualize the process of codebook optimization in Figure 3. For clarity, we only show the optimization process of a code vector. From Figure 3, we can see that only an active (*i.e.*, selected) code vector can be optimized whereas others are never updated (*i.e.*, "die off") for VQ-VAE, however our VQCT performs cooperative optimization between codes for VQIM, *i.e.*, all codes in our codebook can be performed optimization by resorting to semantic relationship between codes, although only a code vector is selected to learn. This is because 1) our codebook is generative but not directly learnable and 2) our optimization variable is the parameter θ_t of codebook transfer network instead of the codebook, such that all codes can achieve optimization with their semantic relationships.

4. Experiments

4.1. Experimental Settings

Dataset. We evaluate our method over four public datasets, including ADE20K [40], CelebA-HQ [17], CUB-200 [30], and MSCOCO [16]. Following [36], we resize each image as 256 × 256 resolution for experiment evaluation.

Evaluation Metrics. We select recent VQ-VAE [27], VQ-GAN [7], Gumbel-VQ [1], and CVQ [38] as our baselines and evaluate these models with image reconstruction performance, which includes four evaluation metrics, *i.e.* Frechet Inception Score (FID) [11] for showing the perceptual similarity of reconstructed images, and $l1$, $l2$, and Peak Signal-to-noise Ratio (PSNR) [8] is employed to measure the pixel-level similarity of image reconstruction.

Table 1. Results of image reconstruction on ADE20K, CelebA-HQ, CUB-200, and MS-COCO. The best results are highlighted in bold.

Models	ADE20K [40]				CelebA-HQ [17]				CUB-200 [30]				MS-COCO [16]			
	FID↓	PSNR↑	$l1$↓	$l2$↓	FID↓	PSNR↑	$l1$↓	$l2$↓	FID↓	PSNR↑	$l1$↓	$l2$↓	FID↓	PSNR↑	$l1$↓	$l2$↓
VQ-VAE [27]	116.85	21.08	0.1282	**0.0368**	36.08	**25.29**	0.0719	0.0139	54.92	24.38	0.0849	0.0183	86.21	**23.55**	**0.0933**	**0.0226**
VQ-GAN [7]	22.04	20.42	0.1290	0.0451	5.66	24.10	0.0798	0.0175	3.63	22.19	0.1051	0.0319	14.45	20.21	0.1311	0.0475
Gumbel-VQ [1]	24.12	20.04	0.1359	0.0482	6.22	23.65	0.0837	0.0194	3.45	22.11	0.1048	0.0318	15.30	20.00	0.1354	0.0488
CVQ [38]	33.63	19.91	0.1379	0.0486	5.19	23.15	0.0917	0.0214	3.61	22.29	0.1034	0.0302	9.94	20.48	0.1253	0.0443
VQCT(ours)	**20.25**	**21.30**	**0.1144**	0.0374	**5.02**	25.18	**0.0699**	**0.0134**	**2.13**	**25.35**	**0.0704**	**0.0160**	**9.82**	21.46	0.1108	0.0366

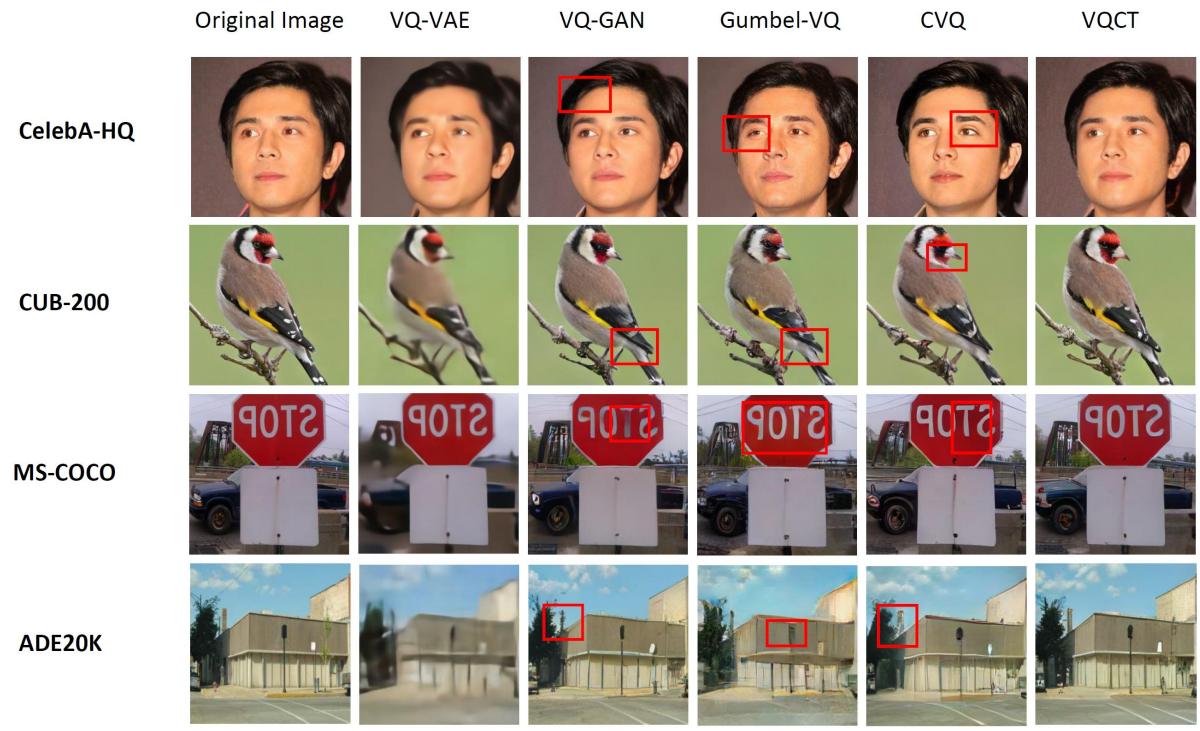


Figure 4. Reconstructed images from different VQIM methods on four datasets. Here, the red-color boxes highlight reconstruction details.

Implementation Details. In our experiments, all parameter settings of our VQCT are same with these existing VQIM methods (e.g., VQVAE, VQGAN, and CVQ) for fair comparison. The key difference lies in that the output of encoder and the input of decoder. We divide the output of encoder into two parts, *i.e.*, an adjective vector and a noun vector, and then concatenate the quantized adjective and noun vectors as the inputs of decoder for reconstruction.

4.2. Discussion of Results

Quantitative Evaluation. In Table 1, we select some state-of-the-art VQIM methods (*i.e.*, VQ-VAE, VQ-GAN, Gumbel-VQ, and CVQ) as our baselines and then report the performance of their image reconstruction. It can be found that our VQCT method outperforms these state-of-the-art methods on most evaluation, which suggests that our VQCT method is effective for VQIM. Specifically, compared with VQ-VAE and VQ-GAN, our method better models the codebook by introducing a well-pretrained codebook from language models and word class knowledge as priors. The results show our method is more effective, especially on FID. It's worth noting that our method also beats Gumbel-VQ, and CVQ, which also focus on alleviate the codebook collapse issue. This shows the effectiveness of our VQCT.

Qualitative Evaluation. We qualitatively compare reconstruction performance of our VQCT and baseline methods (*i.e.*, VQ-VAE, VQ-GAN, Gumbel-VQ, and CVQ). The results are shown in Figure 4. From these results, we can see that our VQ-CT achieves the best reconstruction quality, which can be well aligned with original images in terms of detailed textures. This further verifies the effectiveness.

4.3. Ablation Study

Is introducing the pretrained codebook effective for VQIM? In Table 2, we conduct an ablation study to show the effectiveness of introducing the pretrained codebook. Specially, (i) we implement VQ-GAN as our baseline (i.e. without introducing codebook prior into VQIM); (ii) we

Table 2. Ablation study of pretrained codebook on CUB-200. 'RI' denote random initialization of adjective and noun codebook.

	Setting	CUB-200 [30]			
		FID↓	PSNR↑	$l1$↓	$l2$↓
(i)	**Baseline**(VQ-GAN)	3.63	22.19	0.1051	0.0249
(ii)	**Our VQCT** (RI)	3.45	23.34	0.0908	0.0298
(ii)	**Our VQCT**	2.13	25.35	0.0704	0.0160

Table 3. Ablation study of our codebook transfer network on CUB-200. Here, 'SCB' denote concatenating adj and noun codebooks as a codebook. 'MCB' denote regarding adj and noun codebooks as two codebooks, respectively. "MLP" denotes implementing our codebook transfer network with MLP. "GCN" denotes implementing the our codebook transfer network with GCN.

	Setting	CUB-200 [30]			
		FID↓	PSNR↑	$l1$↓	$l2$↓
(i)	**Baseline**	20.06	16.94	0.2001	0.0902
(ii)	**+Finetune**	3.98	22.42	0.1036	0.0294
(iii)	**+SCB+MLP**	5.64	21.18	0.1147	0.0386
(iv)	**+SCB+GCN**	3.71	22.72	0.0973	0.0283
(v)	**+MCB+MLP**	2.33	24.87	0.0743	0.0178
(vi)	**+MCB+GCN (VQCCT)**	2.13	25.35	0.0704	0.0160

implement our VQCT but randomly initialize our adj and noun codebook and Multilayer Perceptron (MLP) is employed as our code transfer network; and (iii) we implement our VQCT, *i.e.*, introducing the pretrained codebook to enhance codebook learning. From the results, we can see that the performance of (iii) outperforms (i) and (ii) by a large margin, which is reasonable because the semantic relationships between codes from the pretrained codebook can be fully exploited for providing good codebook priors such that more robust codebook can be learned for VQIM. This suggests that our VQCT (*i.e.*, introducing the pretrained codebook from language models) is very effective for VQIM.

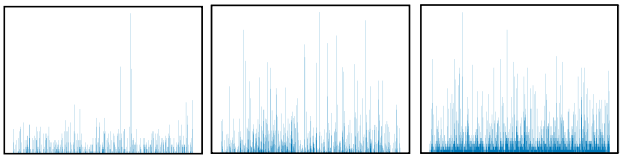
Is our codebook transfer network effective? In Table 3, we analyzed the effectiveness of our codebook transfer network. Specifically, (i) we first directly concatenate the pretrained adj and noun embedding as the codebook of VQIM which is freezed, i.e., our baseline (can be regarded as SPAE [34]); (ii) we further finetune the codebook instead of freezing codebook on (i); (iii) we replace finetuning codebook with our codebook transfer on (ii) and implement the codebook transfer network with MLP; (iv) we implement the codebook transfer network with Graph Convolutional Network (GCN) on (iii); (v) we split the adj and noun codebook as two codebooks on (iii); (vi) we split the adj and noun codebook as two codebooks on (iv), which is exactly our VQCT. From the results of (i) $\sim$ (vi), we observe that: 1) the performance of (iv) $\sim$ (vi) exceeds (i) $\sim$ (ii) by a large margin, which means that it is helpful to employ a codebook transfer network to transfer pretrained codebook to VQIM for robust codebook learning; 2) the performance of (iv)/(vi) outperforms (iii)/(v), which shows the superiority of using GCN to model the relationship between adjective and noun codebooks. This is because the relationships from adjective and noun part-of-speech can be fully exploited for codebook learning. Finally, comparing the results of (v) $\sim$ (vi) with (iii) $\sim$ (iv), we find that the performance of modeling adj and noun codebook as two codebooks is more superior than a single codebook. This is reasonable because the adj and noun are often used together to describe vision features.

Table 4. Ablation study of VQCT generalization on CUB-200.

	Setting	CUB-200 [30]			
		FID↓	PSNR↑	$l1$↓	$l2$↓
(i)	**VQ-VAE**	54.92	24.38	0.0849	0.0183
	VQ-VAE+VQCT	24.39	25.77	0.0720	0.0128
(ii)	**VQGAN**	3.63	22.19	0.1051	0.0319
	VQGAN+VQCT	2.13	25.35	0.0704	0.0160

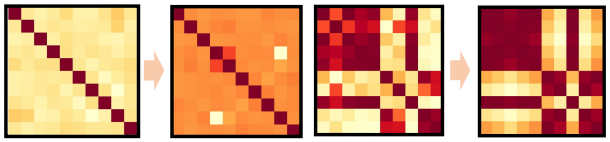
Table 5. Ablation study of language models on CUB-200.

	Setting	CUB-200 [30]			
		FID↓	PSNR↑	$l1$↓	$l2$↓
(i)	**Baseline**	3.63	22.19	0.1051	0.0319
(ii)	**+GloVe**	3.43	22.99	0.0931	0.0271
(iii)	**+CLIP**	2.13	25.35	0.0704	0.0160



(a) VQGAN (b) VQCT (Adj) (c) VQCT (Noun)

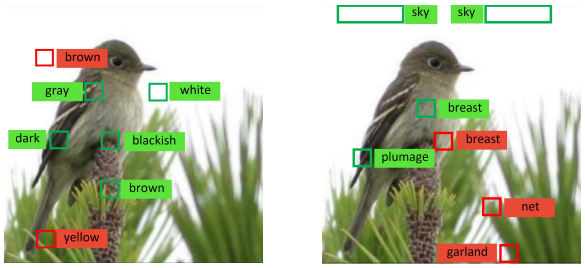
Figure 5. Visualization of codebook utilization on CUB-200.



(a) Similarity Change of VQ-GAN (b) Similarity Change of VQCT

Figure 6. Visualization of codebook similarity on CUB-200.

Is our VQCT general on other VQIM methods? In fact, our VQCT can be regarded as introducing the codebook transfer strategy into VQ-GAN, which is very general on other VQIM methods. To illustrate this point, we conduct an extension experiment on VQ-VAE and VQ-GAN. Specifically, we first report the performance of image reconstruction with VQ-VAE and VQ-GAN, respectively, and then we add our codebook transfer strategy to enhance the robust codebook learning for VQIM. The experimental results are shown in Table 4. From these results, the VQIM performance of VQ-VAE and VQ-GAN achieve significant performance improvement after applying our code-



(a) Adjective Space (b) Noun Space

Figure 7. Visualization of vision-language alignment on CUB-200. Wrong/right matching is marked in red/green, respectively.

book transfer strategy to the codebook of VQ-VAE and VQ-GAN, which verifies the universality of our VQCT.

How does pretrained codebook from different language models impact the performance of our VQCT? In Table 5, we analyze the impact of different language models including Glove [21] and CLIP [23] on VCQT. Specifically, (i) we first select VQ-GAN as our baseline; (ii) we introduce a pretrained codebook from the GloVe-based language model on (i) ; (iii) we introduce a pretrained codebook from the CLIP-based language model on (i). From these results, we can see that 1) the image reconstruction performance of (ii) $\sim$ (iii) exceed (i) by a large margin. This is reasonable because we introduce a prerained codebook from language model to enhance codebook learning on (ii) $\sim$ (iii); and 2) the setting of (iii) (introducing pretrained codebook from CLIP) performs more superior than other settings. Such advantage may be from the pretraining of multimodal alignment. Hence, CLIP is a default setting in our approach.

Can our VQCT alleviate the codebook collapse issue? To answer this question, we select VQ-GAN as baseline and then visualize the codebook utilization of the baseline and our VQCT. The visualization result is shown in Figure 5. From Figure 5, we can see that 1) in VQ-GAN, the codebook utilization is very low (around 27.90% codes are used and 72.10% codes dies off); while 2) the codebook utilization achieves significant improvement, around 42.20% adj and noun codes is used after applying our VQCT.

Can our VQCT achieve code cooperative optimization? In Figure 6, we select VQ-GAN as baseline and then visualize the cosine similarity between codes. For clarity, we randomly select 10 codes to conduct experiments. We can see that compared with VQ-GAN, the similarity between codes of our VQCT can indeed be well maintained during training, which means that our VQCT can achieve the cooperative optimization between codes.

Can our VQCT align vision and language? In Figure 7, we visualize the codes of adjective and noun and mark some right or wrong matching cases. From results, we find that our VQCT can align vision and language in some cases, although by only using unsupervised learning manner, which verifies the effectiveness of our codebook transfer.

Table 6. Results (FID↓) of image synthesis on CelebA-HQ [17] datasets. The best results are highlighted in bold.

Models	VQ-VAE [27]	VQ-GAN [7]	Gumbel-VQ [1]	Reg-VQ [1]	VQCT(ours)
FID	39.57	17.42	16.78	15.34	**14.47**

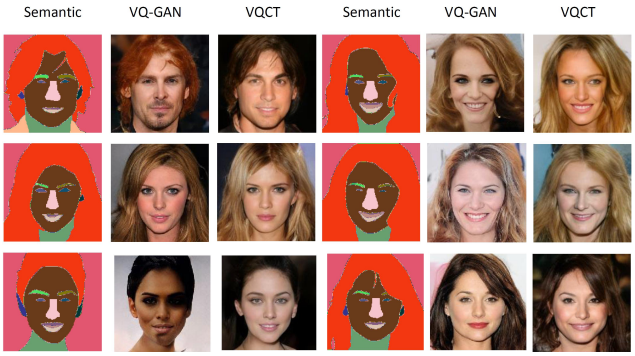


Figure 8. Semantic image synthesis on CelebA-HQ.

4.4. Application: Image Synthesis

Following [7], we conduct a simple experiment to verify the effectiveness of our VQCT on downstream tasks, *i.e.*, semantic image synthesis, where LDM [7] is employed to achieve the prediction of the adjective and noun codes.

Quantitative Evaluation. In Table 6, we select latest VQ-VQE, VQ-GAN, Gumbel-VQ, and Reg-VQ as baselines and report the FID performance. From results, we can see that our VQCT achieve superior performance over these baseline methods. This further verifies the effectiveness of our VQCT on downstream semantic image synthesis.

Qualitative Evaluation. In Figure 8, we show some generation examples of VQ-GAN and our VQCT on image synthesis and completion. From results, we can see that our VQCT indeed can achieve high quality image synthesis.

5. Conclusions

This paper proposes a novel codebook transfer framework for vector-quantized image modeling. In particular, we introduce a pretrained codebook and part-of-speech knowledge as priors and then design a graph convolution codebook transfer network to generate codebook. Its advantage is rich semantic from pretrained codebook can be fully exploited for codebook learning. Results show that our VQCT achieves superior performance over previous methods.

Acknowledgments

This work was supported by the Shenzhen Peacock Program under Grant No. ZX20230597, NSFC under Grant No. 62272130 and Grant No. 62376072, and the Shenzhen Science and Technology Program under Grant No. KCXFZ20211020163403005.

References

[1] Alexei Baevski, Steffen Schneider, and Michael Auli. vq-wav2vec: Self-supervised learning of discrete speech representations. *arXiv preprint arXiv:1910.05453*, 2019. 2, 5, 6, 8

[2] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021. 3

[3] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11315–11325, 2022. 2

[4] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A generative model for music. *arXiv preprint arXiv:2005.00341*, 2020. 2

[5] Ming Ding, Zhuoyi Yang, Wenyi Hong, Wendi Zheng, Chang Zhou, Da Yin, Junyang Lin, Xu Zou, Zhou Shao, Hongxia Yang, et al. Cogview: Mastering text-to-image generation via transformers. *Advances in Neural Information Processing Systems*, 34:19822–19835, 2021. 1

[6] Xiaoyi Dong, Jianmin Bao, Ting Zhang, Dongdong Chen, Weiming Zhang, Lu Yuan, Dong Chen, Fang Wen, Nenghai Yu, and Baining Guo. Peco: Perceptual codebook for bert pre-training of vision transformers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 552–560, 2023. 2

[7] 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*, pages 12873–12883, 2021. 1, 2, 5, 6, 8

[8] Fernando A Fardo, Victor H Conforto, Francisco C de Oliveira, and Paulo S Rodrigues. A formal evaluation of psnr as quality measurement parameter for image segmentation algorithms. *arXiv preprint arXiv:1605.07116*, 2016. 5

[9] Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. Imagebind: One embedding space to bind them all. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15180–15190, 2023. 1, 3

[10] 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*, pages 10696–10706, 2022. 2

[11] 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. 5

[12] Mengqi Huang, Zhendong Mao, Zhuowei Chen, and Yongdong Zhang. 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*, pages 22596–22605, 2023. 2

[13] Mengqi Huang, Zhendong Mao, Quan Wang, and Yongdong Zhang. Not all image regions matter: Masked vector quantization for autoregressive image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2002–2011, 2023. 2

[14] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, page 2, 2019. 2, 3, 4

[15] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11523–11532, 2022. 2

[16] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014. 5, 6

[17] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pages 3730–3738, 2015. 5, 6, 8

[18] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. 3

[19] Jia Ning, Chen Li, Zheng Zhang, Chunyu Wang, Zigang Geng, Qi Dai, Kun He, and Han Hu. All in tokens: Unifying output space of visual tasks via soft token. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19900–19910, 2023. 2

[20] Jialun Peng, Dong Liu, Songcen Xu, and Houqiang Li. Generating diverse structure for image inpainting with hierarchical vq-vae. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10775–10784, 2021. 1

[21] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014. 2, 3, 4, 8

[22] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019. 2, 3

[23] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 2, 3, 4, 8

[24] Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, 2019. 2

[25] Aurko Roy, Ashish Vaswani, Arvind Neelakantan, and Niki Parmar. Theory and experiments on vector quantized autoencoders. *arXiv preprint arXiv:1805.11063*, 2018. 1

[26] 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. 2

[27] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. 1, 2, 3, 5, 6, 8

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 1

[29] Tung-Long Vuong, Trung Le, He Zhao, Chuanxia Zheng, Mehrtash Harandi, Jianfei Cai, and Dinh Phung. Vector quantized wasserstein auto-encoder. *arXiv preprint arXiv:2302.05917*, 2023. 2

[30] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 5, 6, 7

[31] Will Williams, Sam Ringer, Tom Ash, David MacLeod, Jamie Dougherty, and John Hughes. Hierarchical quantized autoencoders. *Advances in Neural Information Processing Systems*, 33:4524–4535, 2020. 2

[32] Chen Xing, Negar Rostamzadeh, Boris Oreshkin, and Pedro O O Pinheiro. Adaptive cross-modal few-shot learning. *Advances in Neural Information Processing Systems*, 32, 2019. 2

[33] Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. Vector-quantized image modeling with improved vqgan. *arXiv preprint arXiv:2110.04627*, 2021. 2

[34] Lijun Yu, Yong Cheng, Zhiruo Wang, Vivek Kumar, Wolfgang Macherey, Yanping Huang, David A Ross, Irfan Essa, Yonatan Bisk, Ming-Hsuan Yang, et al. Spae: Semantic pyramid autoencoder for multimodal generation with frozen llms. *arXiv preprint arXiv:2306.17842*, 2023. 3, 7

[35] Baoquan Zhang, Xutao Li, Yunming Ye, Zhichao Huang, and Lisai Zhang. Prototype completion with primitive knowledge for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3754–3762, 2021. 2

[36] Jiahui Zhang, Fangneng Zhan, Christian Theobalt, and Shijian Lu. Regularized vector quantization for tokenized image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18467–18476, 2023. 1, 2, 3, 5

[37] Renrui Zhang, Jiaming Han, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, Peng Gao, and Yu Qiao. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *arXiv preprint arXiv:2303.16199*, 2023. 1

[38] Chuanxia Zheng and Andrea Vedaldi. Online clustered codebook. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22798–22807, 2023. 1, 2, 5, 6

[39] Chuanxia Zheng, Tung-Long Vuong, Jianfei Cai, and Dinh Phung. Movq: Modulating quantized vectors for high-fidelity image generation. *Advances in Neural Information Processing Systems*, 35:23412–23425, 2022. 2

[40] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 633–641, 2017. 5, 6

[41] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*, 2023. 1