Not All Image Regions Matter: Masked Vector Quantization for Autoregressive Image Generation (Supplementary Material)

Mengqi Huang[1], Zhendong Mao[1, 3]*, Quan Wang[2], Yongdong Zhang[1, 3]
[1]University of Science and Technology of China, Hefei, China; [2]MOE Key Laboratory of Trustworthy, Distributed Computing and Service, Beijing University of Posts and Telecommunications, Beijing, China;
[3]Institute of Artificial intelligence, Hefei Comprehensive National Science Center, Hefei, China
huangmq@mail.ustc.edu.cn, {zdmao, zhyd73}@ustc.edu.cn, wangquan@bupt.edu.cn

1. Detailed Implementations

Training Objective For MQ-VAE. Our proposed MQ-VAE requires no additional loss to guide the learning of adaptive mask and de-mask mechanisms. The adaptive mask and de-mask module are trained together with all other modules using VQGAN's loss and nothing special. The scoring net of the adaptive mask module is initialized by PyTorch's default Kaiming Initialization. The encoder E, decoder G, codebook $\mathcal{C}$ as well as our proposed adaptive mask module and adaptive de-mask module are trained jointly with respect to the loss $\mathcal{L} = \mathcal{L}_{\text{recon}} + \beta\mathcal{L}_{\text{commit}}$ with a multiplicative factor $\beta > 0$. The reconstruction loss $\mathcal{L}_{\text{recon}}$ and the commitment loss $\mathcal{L}_{\text{commit}}$ are defined as:

$$\mathcal{L}_{\text{recon}} = \|\mathbf{X} - \tilde{\mathbf{X}}\|_2^2,$$
$$\mathcal{L}_{\text{commit}} = \left\|\hat{\mathbf{Z}} - \text{sg}\left[\hat{\mathbf{Z}}^{\boldsymbol{q}}\right]\right\|_2^2,$$

where $\text{sg}[\cdot]$ is the stop-gradient operator, and the straight-through estimator [11] is used for the back-propagation through the vector quantization. We also trained with adversarial learning to improve the perceptual quality of reconstructed images. The patch-based adversarial loss [4] and the perceptual loss [5] are used together as described in the previous study [3, 7].

Architecture of MQ-VAE. For the architecture of MQ-VAE, we follow the architecture of VQ-GAN [3] and RQ-VAE [7] to give a fair comparison. To extract the feature map of resolution 8×8, we add two residual blocks with 512 channels each followed by a down-/up-sampling block. To extract the feature map of resolution of 32×32, we remove the last two residual blocks and the corresponding down-/up-sampling block in the encoder and decoder. The size of the codebook is all set to be 1024, if not specified.

*Zhendong Mao is the corresponding author.

Code layers	Position layers	FID↓
12	12	7.67
6	12	8.32
6	18	8.10
12	6	7.74
18	6	7.53

Table 1. Ablations on different layers of small-version Stackformer on FFHQ.

Architecture of Stackformer. The Stackformer, which consists of the Code-Transformer and Position-Transformer, adopts a stack of causal self-attention blocks [12] for each compartment. For the text-to-image generation on MS-COCO [8] benchmark, the length of the text condition is set to be 32, and the last token in text conditions predicts the token at the first position of images. Please refer to Table 2 for the detailed hyper-parameters for Stackformer.

Training Details. To report the main results, we conduct experiments on eight RTX-3090 GPUs. To report the ablation and analysis results, we conduct experiments on four RTX-3090 GPUs. We trained our model following previous works [3, 7, 14]. To be specific, for FFHQ, MQ-VAE is trained for 150 epochs with batch size 8 on each GPU. We use the Adam optimizer [6] with $\beta_1 = 0.5$ and $\beta_2 = 0.9$. The base learning rate is set to be 0.0000045 following [3]. The learning rate is linearly warmed up during the first 5 epochs. As for the adversarial and perceptual loss, we follow the experimental setting of [3, 7]. To be specific, the weight for adversarial loss is set to be 0.75 and the weight for perceptual loss is set to be 1.0. The results in the main paper do not use any other tricks such as the random restart of unused codes proposed in JukeBox [2] to increase the codebook usage, to give a fair comparison

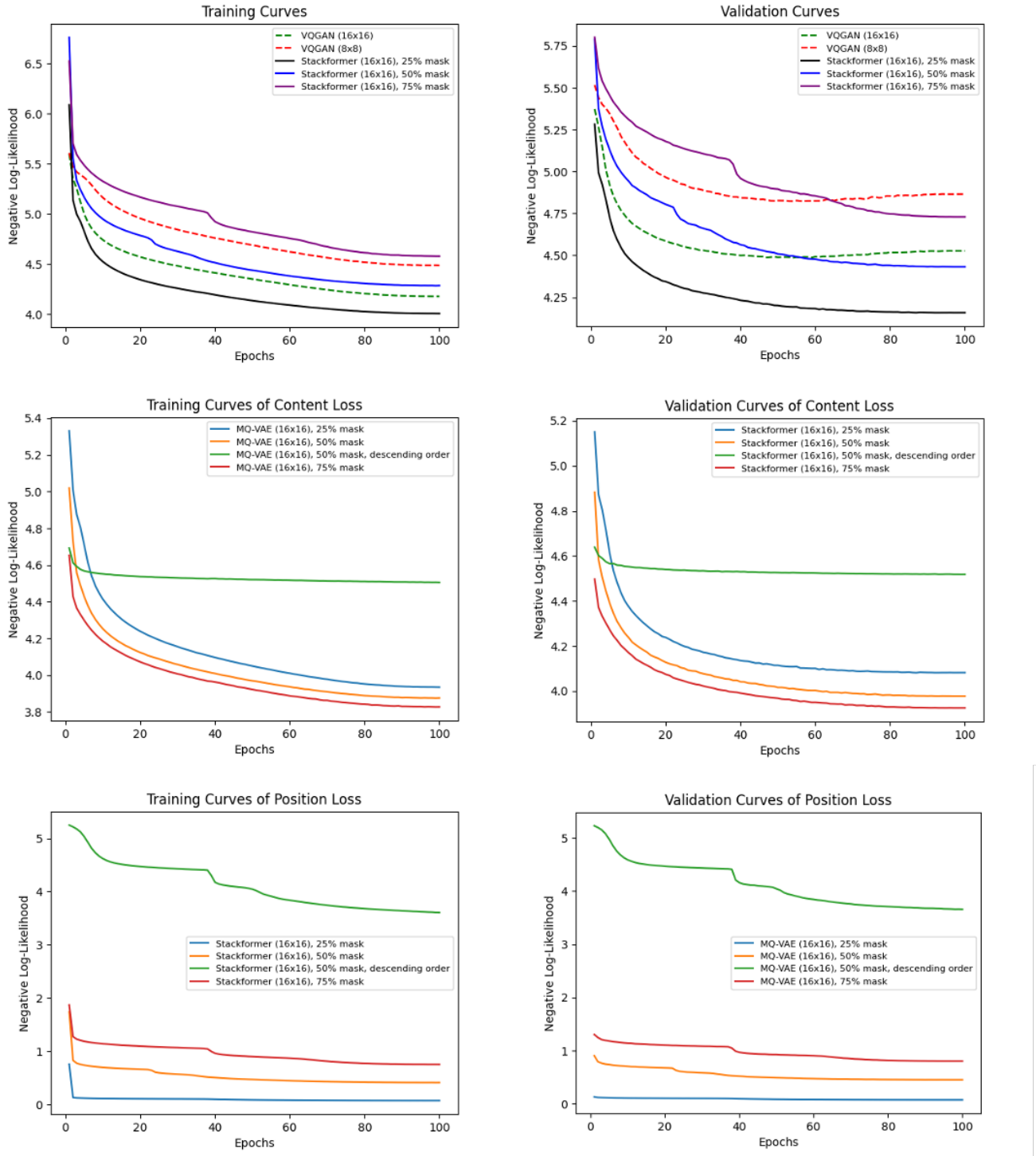

Figure 1. Detailed Training and Validation curves of VQGAN baseline and our proposed Stackformer with different experimental settings on FFHQ benchmark.

with VQGAN [3]. For ImageNet, MQ-VAE is trained for 50 epochs. The learning rate is linearly warmed up for 0.5 epochs. For MSCOCO, we directly use the pretrained MQ-VAE on ImageNet without finetuning.

Figure 2. Unconditional generated 256×256 images by our models of different mask radio on FFHQ.

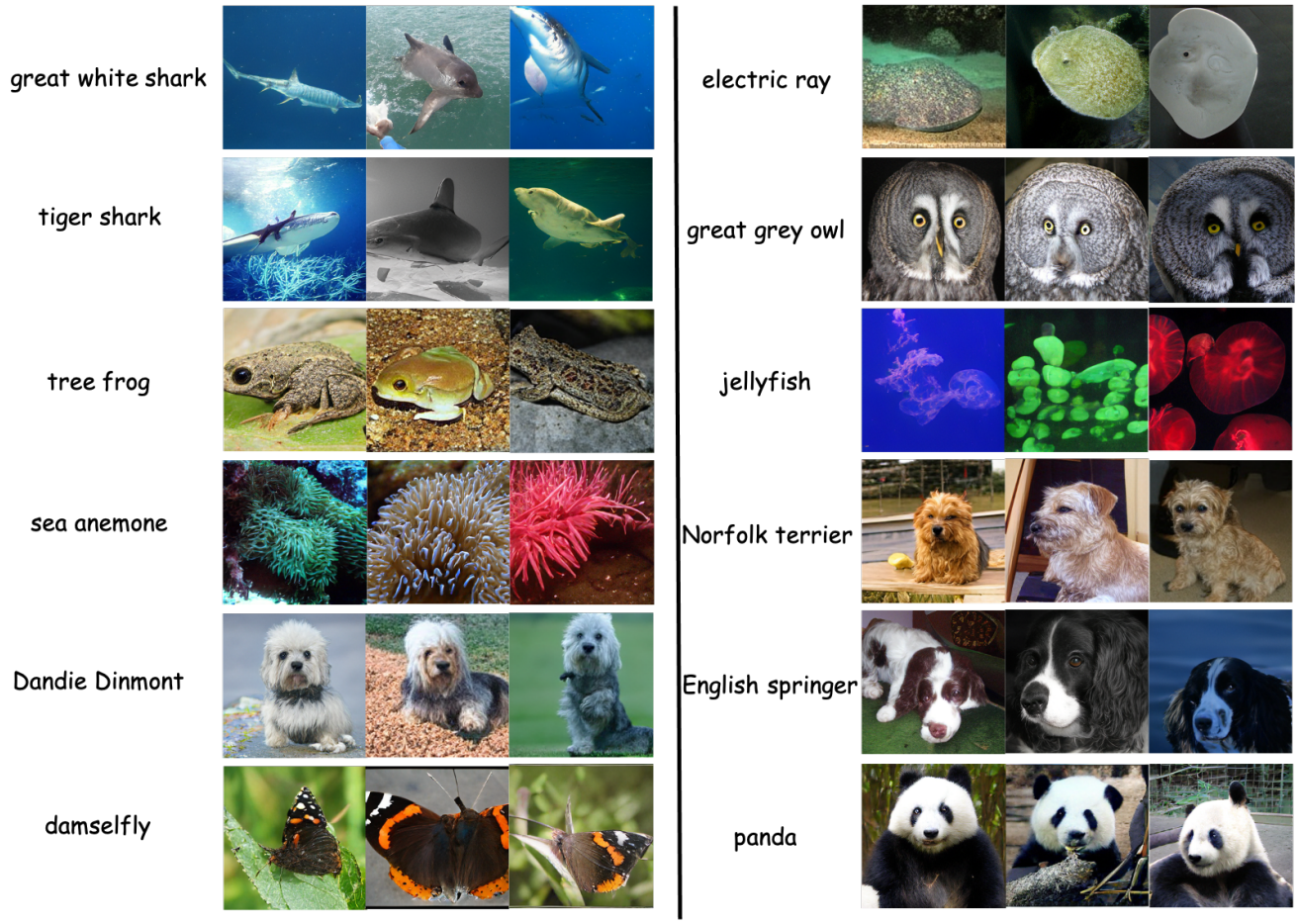

Figure 3. Class-conditional generated 256×256 images by our models on ImageNet.

All the Stackformer models are trained using AdamW optimizer [9] with $\beta_1 = 0.9$ and $\beta_2 = 0.95$. The weight decay is set to be 0.01. We use a cosine learning rate decay schedule with 0.0005 of the initial learning rate. The Stackformer is trained for 100 epochs for FFHQ and ImageNet. As for MS-COCO, we train Stackformer for 120 epochs following previous text-to-image works [1, 10, 13] to give a fair comparison. The batch size is all set to be 10 at each GPU. In all experiments, the dropout rate of each self-attention block is set to 0.1. Different from previous works [3, 7], we do not use early stopping since the overfitting problem does not exist in our Stackformer thanks to the masked vector quantization.

	N_{Code}	N_{Position}	# Params	n_e	n_{head}	dropout
Stackformer	18	6	307M	1024	16	0.1
Stackformer	36	12	607M	1024	16	0.1

Table 2. The hyperparameters for implementing Stackformer. N_{Code} denotes the number of transformer encoders for Code-Transformer and $N_{Position}$ denotes the number of transformer encoders for Position-Transformer. n_e denotes the dimensionality of Stackformer. n_{head} denotes the number of heads used in the multi-head self-attention.

Model Setting	mask radio (%)	f	L	FID
Stackformer	50	16	128	8.36
Stackformer, w/o absolute sequence	50	16	128	8.53
Stackformer, descending order	50	16	128	26.87
Stackformer$_r$	50	16	128	8.69

Table 3. Additional ablation studies on FFHQ benchmark. Here f is the downsampling factor. L is the coding length.

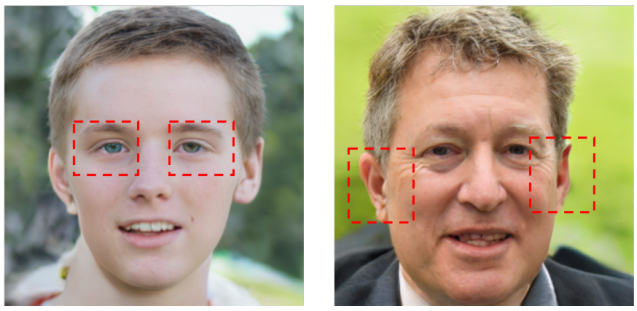

Figure 4. Some typical artifacts exist in the generated images by our method, *e.g.*, the different color our eyes (left) and the different shape of ears (right). The results are generated by Stackformer with MQ-VAE (f=16, 50% mask radio)

2. Additional Experiments

Impact of different Code & Position layers. We study the impact of position and code prediction accuracy in Table 1. We found that the accuracy of code is more important than position, *e.g.*, FID drops 0.65 (from 7.67 to 8.32) when 6 code layers are removed, while FID only drops 0.07 (from 7.67 to 7.74) when 6 position layers are removed. The reason behind this is two-fold: (1) the accuracy of position is based on the code since we first predict the code and then the position. (2) the position is very easy to learn since we find the position loss drops very quickly during training, see Figure 1 for the detailed training curves.

Impact of sequence order. As shown in Table 3, Stackformer trained with importance descending order resulted in very poor results (26.87 FID score) compared with the one trained with raster scan rearranged (8.36 FID score). We visualize the detailed training and validation curves in Figure 1. We find that both the content loss (code loss described in the main paper) and the position loss of Stackformer trained with importance descending order are hard to converge, which we hypothesize the reason lies in the dramatic position changes of adjacent code. And we find that the raster-scan reordered paradigm could converge quickly.

Impact of different Stackformer structure. Our proposed Stackformer also has a variant, i.e., switch the order of Code-Transformer and Position-Transformer, which we denote as Stackformer$_r$. As shown in Table 3, we find that Stackformer result in slightly better generation quality compared with its variant, *i.e.*, Stackformer$_r$ (8.36 compared with 8.69), which indicates that first modeling codes and then modeling their position is a better choice.

Impact of sequence position. As shown in Table 3, we find that when we remove the extra learned absolute sequence position embedding in the input of Code-Transformer, the performance slightly decreases (from 8.36 to 8.53), which denotes that adding the position in the sequence could make the model make use of the order of the sequence.

Additional Results of Generated Images by Stackformer. We provide more generated examples of our method on the challenge ImageNet, as shown in Figure 3. Also, We visualize the generation results of different mask radios, as shown in Figure 2.

Analysis between predicted scores and encoded image features. We find that the encoded image feature values do correlate with the predicted scores of the scoring net in the adaptive mask module. We do the Pearsonr analysis and find the p-value is 0.058 (lower means more correlated) on ImageNet's val-set. The reason is the encoded features' values also somewhat represent images' structures (see Figure 5).

3. Potential Societal Impacts

The proposed generative model can serve as a data engine to alleviate the challenge of data collection. More im-

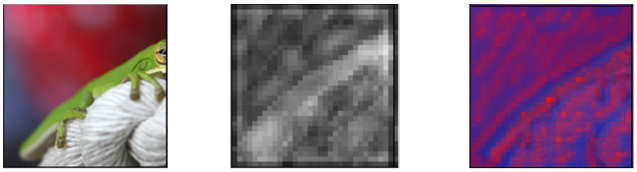

Figure 5. Analysis between predicted scores and encoded image features. We show that the image feature values do correlate with the predicted scores.

portantly, using synthesized image examples helps avoid privacy concerns. However, the abuse of advanced generative models may create fake media materials, which demands caution in the future.

4. Typical artifacts

In this section, we show some typical artifacts that exist in the generated images by our proposed method to give a fair and comprehensive understanding of our works. We observe that when the mask radio increase (like 50%), there exist some detail inconsistent in the final generated images, *e.g.*, the different color generated eyes or the different shape of generated ears, as shown in Figure 4. We hypothesize the reason lies in that an improper high mask radio will inevitably mask some important regions which mask the demask module fail to recover a totally consistent feature map and finally result in detail inconsistent in the generated images.

5. Additional visualization

We provide more representative random samples in Figure 6 and Figure 7 to give a more comprehensive observation. The examples of MQ-VAE are 75% mask ratio on the 32×32 feature map and result in the total code sequence length of 256. Both MQ-VAE and VQGAN are trained with 10 epochs on ImageNet. The examples of VQGAN are on the 16×16 feature map. The visualization validates that our proposed mask mechanism successfully learns to preserve the important structural regions. We also visualize side-by-side reconstruction results between the proposed MQ-VAE and VQGAN, which shows that MQ-VAE results in better image reconstruction quality with the same code sequence length (256). For example, the second row in Figure 4 in the supplementary shows that our MQ-VAE reconstruction result has a clearer human face (e.g., eyes, nose, and mouse) while the VQGAN reconstruction result is much more distorted.

References

[1] Zhangling Chen, Ce Wang, Huaming Wu, Kun Shang, and Jun Wang. Dmgan: discriminative metric-based generative adversarial networks. *Knowledge-Based Systems*, 192:105370, 2020. 3

[2] 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. 1

[3] 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, 3

[4] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017. 1

[5] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016. 1

[6] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 1

[7] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. *arXiv preprint arXiv:2203.01941*, 2022. 1, 3

[8] 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 *European conference on computer vision*, pages 740–755. Springer, 2014. 1

[9] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 3

[10] Ming Tao, Hao Tang, Songsong Wu, Nicu Sebe, Xiao-Yuan Jing, Fei Wu, and Bingkun Bao. Df-gan: Deep fusion generative adversarial networks for text-to-image synthesis. *arXiv preprint arXiv:2008.05865*, 2020. 3

[11] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. 1

[12] 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

[13] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1316–1324, 2018. 3

[14] Chuanxia Zheng, Long Tung Vuong, Jianfei Cai, and Dinh Phung. Movq: Modulating quantized vectors for high-fidelity image generation. *arXiv preprint arXiv:2209.09002*, 2022. 1

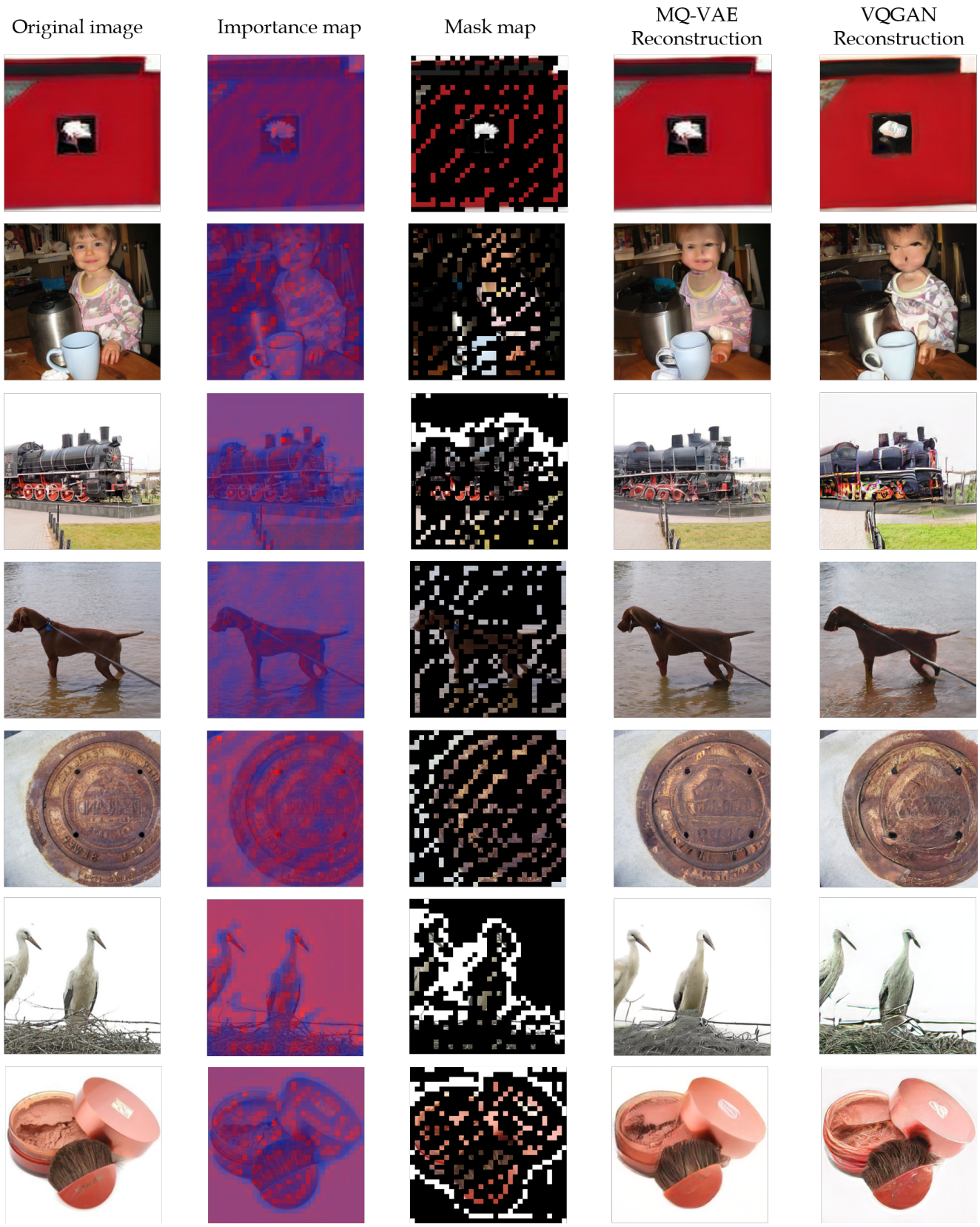

Figure 6. The visualization of our adaptive mask module which learns to mask unimportant regions on ImageNet and side-by-side reconstruction results between our proposed MQ-VAE and VQGAN. In the importance map, red denotes high scores while blue denotes low scores.

Figure 7. The visualization of our adaptive mask module which learns to mask unimportant regions on ImageNet and side-by-side reconstruction results between our proposed MQ-VAE and VQGAN. In the importance map, red denotes high scores while blue denotes low scores.