Neural Image Compression Using Masked Sparse Visual Representation

Wei Jiang
Futurewei Technologies Inc.
Santa Clara, CA
wjiang@futurewei.com

Wei Wang
Futurewei Technologies Inc.
Santa Clara, CA
rickweiwang@futurewei.com

Yue Chen
Futurewei Technologies Inc.
Santa Clara, CA
ychen@futurewei.com

Abstract

We study neural image compression based on the Sparse Visual Representation (SVR), where images are embedded into a discrete latent space spanned by learned visual codebooks. By sharing codebooks with the decoder, the encoder transfers integer codeword indices that are efficient and cross-platform robust, and the decoder retrieves the embedded latent feature using the indices for reconstruction. Previous SVR-based compression lacks effective mechanism for rate-distortion tradeoffs, where one can only pursue either high reconstruction quality or low transmission bitrate. We propose a Masked Adaptive Codebook learning (M-AdaCode) method that applies masks to the latent feature subspace to balance bitrate and reconstruction quality. A set of semantic-class-dependent basis codebooks are learned, which are weighted combined to generate a rich latent feature for high-quality reconstruction. The combining weights are adaptively derived from each input image, providing fidelity information with additional transmission costs. By masking out unimportant weights in the encoder and recovering them in the decoder, we can trade off reconstruction quality for transmission bits, and the masking rate controls the balance between bitrate and distortion. Experiments over the standard JPEG-AI dataset demonstrate the effectiveness of our M-AdaCode approach.

1. Introduction

Neural image compression (NIC) has been actively studied in recent years. Using neural networks (NN), the encoder transforms the input image into a compact latent representation, based on which the decoder reconstructs the output image. NIC has two general research topics: (1) how to learn an effective and expressive latent representation, and (2) how to quantize and encode the latent representation for efficient transmission. So far, the most popular framework is based on hyperpriors [3] (shown in Figure 1a). An entropy model is used to encode/decode the quantized latent, which marries classical entropy coding with NN-based representation learning in a Variational AutoEncoder (VAE) structure. Many improvements have been made to the entropy model [13, 24, 29] to speedup computation and improve reconstruction quality.

In this work, we investigate a different framework for NIC based on the Sparse Visual Representation (SVR) (shown in Figure 1d). We learn discrete generative priors as visual codebooks, and embed images into a discrete latent space spanned by the codebooks. By sharing the learned codebooks between the encoder and decoder, images can be mapped to integer codeword indices in the encoder, and the decoder can use these indices to retrieve the corresponding codeword latent feature for reconstruction.

One major benefit of the SVR-based compression is the robustness to heterogeneous platforms by transferring integer indices. One caveat of the hyperprior framework is the extreme sensitivity to small differences between the encoder and decoder in calculating the hyperpriors P [4]. Even perturbations caused by floating round-off error can lead to catastrophic error propagation in the decoded latent feature $\hat{Y}$. Most works simply assume homogeneous platforms and deterministic CPU calculation in the entropy model, which is unfortunately impractical. In real applications, senders and receivers usually use different hardware or software platforms where the numerical round-off difference well exists, and not using GPU to avoid the non-deterministic GPU calculation largely limits the computation speed. Only a few works have addressed this problem, *e.g.*, by using integer NN to prevent non-deterministic GPU computation [4] or by designing special NN modules that are friendly to CPU computation to speed up inference [34]. However, such solutions cannot be flexibly generalized to arbitrary network architectures. In comparison, SVR-based compression not only avoids the computational sensitive entropy model, but also brings additional benefits from SVR-based restoration, such as the improved robustness against input image degradations, and the freedom of expanding latent feature dimensions without increasing bitrates.

In particular, we address the challenging dilemma of previous SVR-based compression in trading off bitrate and distortion: it is difficulty to achieve high-quality (HQ) reconstruction using one low-bitrate semantic-class-agnostic codebook, and it is difficulty to achieve low bitrate using multiple HQ semantic-class-dependent codebooks. Due to the complexity of visual content in natural images, the expressiveness and richness of one semantic-class-agnostic codebook (*e.g*., MAsked Generative Encoder as MAGE [21]) limits the reconstruction quality, while the additional image-adaptive information for recovering a rich feature for HQ reconstruction (*e.g*., image-Adaptive Codebook learning as AdaCode [23]) consumes too many bits to transfer.

We propose a Masked Adaptive Codebook learning (M-AdaCode) method for practical SVR-based compression, which applies masks to the latent feature subspaces to balance bitrates and reconstruction quality. Specifically, we build our method on top of AdaCode [23] by adding an effective weight masking and refilling mechanism. A set of semantic-class-dependent basis codebooks are learned, and a weight map to combine these basis codebooks are adaptively determined for each input image. Adaptively combing the rich codebooks provides additional fidelity information for HQ reconstruction, but with high bit costs due to the transmission overhead of the dense weight map. By masking out unimportant weights in the encoder and recovering the weight map later in the decoder, we can reduce the transmission bits by compromising reconstruction performance. The masking rate controls the tradeoff between bitrate and reconstruction distortion. As shown in Figure 2, our method practically operates over a variety of bitrates, in contrast to previous SVR-based compression that only works in ultra-low or high bitrate ranges.

Our M-AdaCode can also be seen as a method of Masked Image Modeling (MIM) [14, 21]. Instead of applying masks in the spatial domain, we apply masks over latent feature subspaces. Using the redundant information in the latent space the HQ feature can be recovered from the degraded masked version, so that the masked SVR has improved representation efficiency to reduce transmission costs. We evaluate our approach over the standard JPEG-AI dataset [2]. Our method is compared with the State-Of-The-Art (SOTA) class-agnostic SVR method MAGE [21] that uses spatial-masking MIM, and with the SOTA class-dependent SVR method AdaCode [23] that uses a dense weight map. Experiments demonstrate the effectiveness of our M-AdaCode method.

2. Related Works

2.1. Sparse Visual Representation Learning

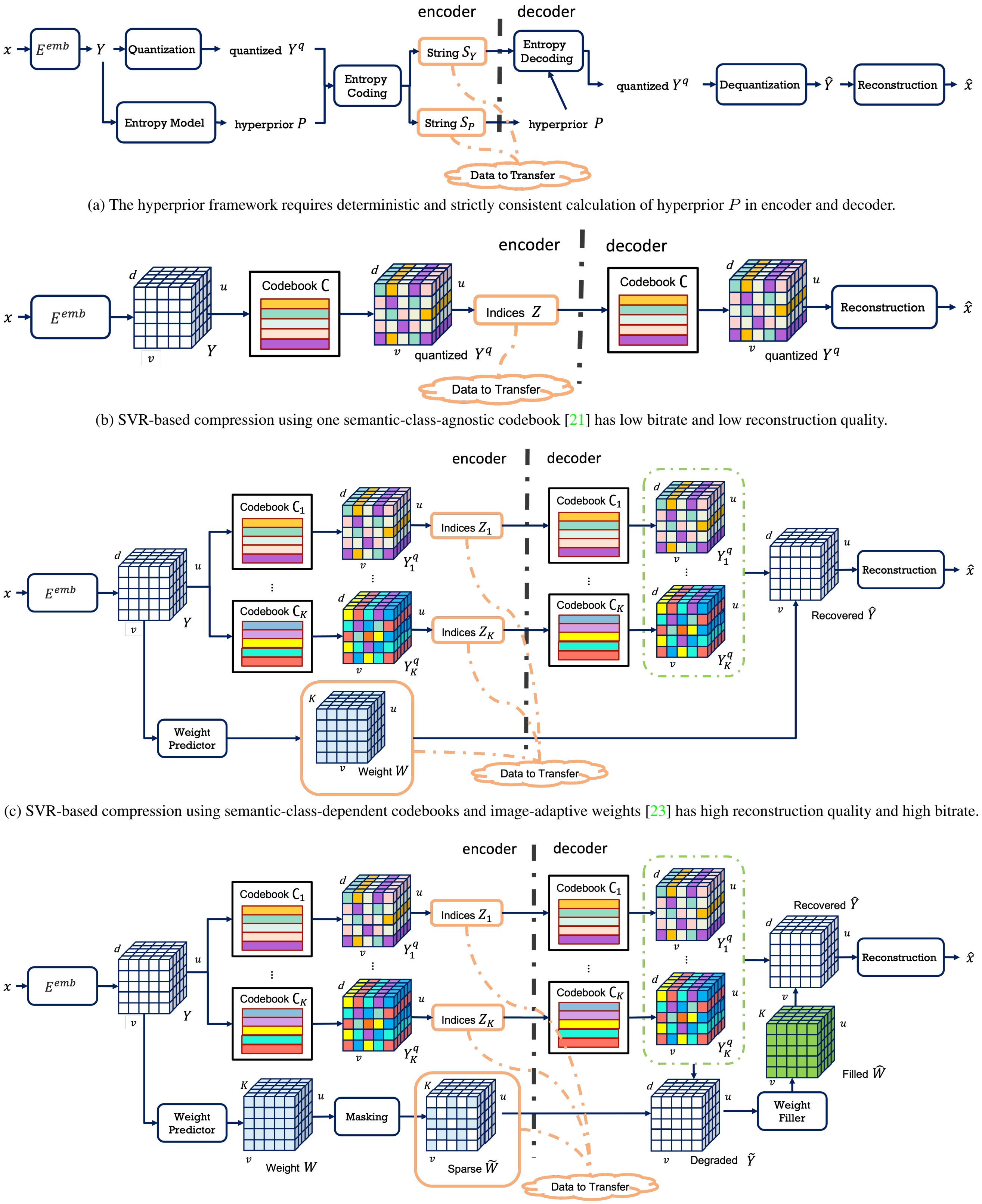
Discrete generative priors have shown impressive performance in image restoration tasks like super-resolution [7], denoising [11], compression [17] *etc*. By embedding images into a discrete latent space spanned by learned visual codebooks, the SVR has improved robustness to various image degradations. For instance, VQ-VAE [27] learns a highly compressed codebook by a vector-quantized autoencoder. VQGAN [11] further improves restoration quality by using Generative Adversarial Networks (GAN) with adversarial and perceptual loss. In general, it is difficult to learn a single general codebook for all image categories. Natural images have very complicated visual content, and a class-agnostic codebook usually has limited representation power for HQ reconstruction. Therefore, most methods focus on specific image categories (*e.g*., faces, architectures). For instance, SVR has achieved great success in face generation due to the highly structured characteristics of human faces, where an HQ codebook can be learned with generic and rich details for HQ face restoration [31, 35].

For general natural images, to improve the restoration power of SVR, the recent AdaCode method [23] uses an image-adaptive codebook learning approach. Instead of learning a single codebook for all categories of images, a set of basis codebooks are learned, each corresponding to a semantic partition of the latent space. A weight map to combine such basis codebooks are adaptively determined for each input image. By learning the semantic-class-guided codebooks, the semantic-class-agnostic restoration performance can be largely improved.

2.2. Neural Image Compression

There are two main research topics for NIC: how to learned an image latent representation, and how to quantize and encode the latent representation. One most popular framework is based on hyperpriors [3], where the image is transformed into a dense latent representation, and an entropy model encodes/decodes the quantized latent representation for efficient transmission. Many improvements have been made to improve the transformation for computing the latent [9, 24, 36] and/or the entropy model [13, 24, 29]. GAN has also been used for learning a good transformation [1, 6, 25]. However, studies show that there are complex competing relations among bitrate, distortion, and perceptual quality [5, 6]. As a result, previous GAN-based NIC methods focus on very low-bitrate scenarios where low fidelity is less important than the good perceptual quality from generated textures and details.

One vital issue of the hyperprior framework is the extreme sensitivity to small differences between the encoder and decoder in calculating the hyperpriors [4]. Even floating round-off error can lead to catastrophic error propagation in the decoded latent feature. The problem is largely overlooked, where most works simply assume homogeneous platforms and deterministic CPU calculation. Some work uses integer NN to prevent non-deterministic GPU computation [4]. Some work designs special NN module that is computational friendly to CPU to speed up infer-



(a) The hyperprior framework requires deterministic and strictly consistent calculation of hyperprior P in encoder and decoder.

(b) SVR-based compression using one semantic-class-agnostic codebook [21] has low bitrate and low reconstruction quality.

(c) SVR-based compression using semantic-class-dependent codebooks and image-adaptive weights [23] has high reconstruction quality and high bitrate.

(d) SVR-based compression using M-AdaCode with flexible weight masking and refilling for tradeoffs between bitrate and reconstruction quality.

Figure 1. Different neural image compression frameworks.

ence [34]. However, such solutions cannot be easily generalized to arbitrary network architectures.

2.3. SVR-based Compression

SVR is intuitively suitable for compression among GAN-based generative methods. SVR represents images by codeword indices, based on which the decoder can retrieve the corresponding codeword feature for reconstruction. The integer indices are easy to transfer, and are robust to small computation differences in heterogeneous hardware and software platforms.

However, due to the difficulty of learning SVR for HQ restoration over general images, previous methods use SVR for very low-bitrate cases, where reconstruction with low fidelity yet good perceptual-quality is tolerated. For example, MIM is combined with product quantization of VQ-VAE in [10] to achieve extreme compression rates. Other methods focus on special content categories that can be better modeled by SVR, such as human faces. For example, face reenactment is used to compress face videos based on codebooks of facial keypoints [30]. CodeFormer face restoration [35] is used to combine a VQGAN with highly compressed low-quality features to trade off perceptual quality and fidelity [17].

As for general images, to the best of our knowledge, no existing work studies SVR-based compression with normal bitrates. Although the AdaCode method [23] can achieve high restoration quality, it is not compression-friendly due to the high transmission overhead for the predicted image-adaptive weight map.

2.4. Masked Image Modeling

MIM has been shown effective in learning HQ visual representations via self-supervised learning. Early methods like MAE [14] and CMAE [15] favor the performance of the representations on downstream tasks instead of the quality of the reconstructed images. The recent MAGE [21] learns a generic VQGAN representation by a single token-based MIM framework with variable masking ratios, which improves unconditioned image generation performance.

3. Approach

The general architecture of the baseline SVR-based image compression framework can be summarized in Figure 1b. An input image $X \in \mathbb{R}^{w \times h \times c}$ is first embedded into a latent feature $Y \in \mathbb{R}^{u \times v \times d}$ by an embedding network E^{emb}. Using a learned codebook $\mathcal{C} = \{c_l \in \mathbb{R}^d\}$, the latent Y is further mapped into a discrete quantized latent feature $Y^q \in \mathbb{R}^{u \times v \times d}$. Specifically, each super-pixel $y^q(l)$ ($l = 1, \ldots, u \times v$) in Y^q corresponds to a codeword $c_l \in \mathcal{C}$ that is closest to the corresponding latent feature $y(l)$ in Y:

$$c_l = argmin_{c_i \in \mathcal{C}} D(c_i, y(l))).$$

Since $y^q(l)$ can be represented by the index z_l of the codeword c_l, the entire Y^q can be mapped to an n-dim vector Z of integers, $n = u \times v$. Z can be efficiently transmitted to the decoder with very little bit consumption, *e.g.*, 10 bits/super-pixel for a codebook with 1024 codewords, and the compression rate can be quite high. On the decoder side, using the codebook $\mathcal{C}$, the quantized feature Y^q is first retrieved based on the received codeword indices Z, and then a reconstruction network reconstructs the output image $\hat{x}$ based on Y^q. One example of this baseline SVR-based compression method is MAGE [21], which uses MIM to learn a general SOTA visual codebook for general image reconstruction with very low bitrates.

Aiming at improving the quality of the learned SVR for general image restoration, the AdaCode method [23] (as described in Figure 1c) learns a set of basis codebooks $\mathcal{C}_1, \ldots, \mathcal{C}_K$, each corresponding to a semantic partition of the latent space. For each individual input, a weight map $W \in \mathbb{R}^{u \times v \times K}$ is computed to combine the basis codebooks for adaptive image restoration. Specifically, the embedded latent feature Y is mapped to a set of quantized latent features $Y_1^q, \ldots, Y_K^q$ using each of the basis codebooks, respectively. Then a recovered latent $\hat{Y}$ is computed as a reconstructed version of latent Y, where for each super-pixel $\hat{y}(l)$ in the recovered $\hat{Y}$ ($l = 1, \ldots, u \times v$):

$$y(l) = \sum\nolimits_{j=1}^{K} w_j(l) y_j^q(l), \tag{1}$$

where $w_j(l)$ is the weight of the j-th codebook for the l-th super-pixel in W.

This framework generates a more expressive recovered latent $\hat{Y}$ that preserves the fidelity cue of each input image than using a single semantic-class-agnostic codebook, and achieves SOTA reconstruction performance. However, it is not suitable for compression. The weight map W needs to be transmitted for each input image, which consumes too many bits. As a result, AdaCode operates in the very high-bitrate range when used for compression.

We propose a practical SVR-based compression framework that can operate in normal bitrate range. The main target is to recover a rich latent $\hat{Y}$ on the decoder side with as little transmitted data as possible. This is in comparison to the extreme case of MAGE that does not use any information to recover a rich latent, or Adacode that uses a dense weight map but ignores transmission costs. Figure 1d gives the detailed architecture of our M-AdaCode method. We use a weight masking and refilling mechanism. The encoder masks out unimportant weights in the weight map to reduce the amount of bits to transfer, which results in a degraded latent $\tilde{Y}$ on the decoder side. Then the decoder re-predicts a full weight map $\hat{W}$ based on the degraded $\tilde{Y}$ for combining codebooks, and computes the recovered latent $\hat{Y}$ for final image reconstruction. The masking rate controls the bitrate, ranging from using full weight map as AdaCode to only one codebook similar to MAGE.

From another perspective, our M-AdaCode can be seen as an MIM method. Instead of applying masks in the spatial domain, we apply masks over latent feature subspaces, and use the redundant information in the feature subspace to recover the HQ latent feature from the degraded masked version. By controlling the masking rate, we tune the representation efficiency of SVR by trading off reconstruction quality for transmission bits.

3.1. Weight Masking and Refilling

Let m denote the number of codebooks to keep for each super-pixel, $1 \leq m \leq K$. Given the predicted weight map $W \in \mathbb{R}^{u \times v \times K}$, the encoder masks out $K - m$ items in each vector $\mathbf{w}_l \in \mathbb{R}^K$ corresponding to the l-th super-pixel ($l = 1, \ldots, u \times v$). The masked out items have smallest absolute values to minimize the impact on the degraded latent $\tilde{Y}$. Then for each super-pixel, only the non-zero remaining weights (16 bits per weight item) and the corresponding codebook indices (floor($\log_2 K$) bits per weight item) need to be transmitted, totalling $(16 + \text{floor}(\log_2 K)) \times m$ instead of the original $16 \times K$. Parameter m provides the tradeoff between bitrate and reconstruction quality. In general, the larger the number of codebooks to use, the better the reconstruction quality and the larger the bitrate.

On the decoder side, using the received masked weight map $\tilde{W}$, the degraded latent $\tilde{Y}$ can be computed in the same way as Equation (1), where only the corresponding codebooks with non-zero weights contribute to the feature computation for each super-pixel. Based on this degraded latent $\tilde{Y}$, the weight filler network predicts another full weight map $\hat{W}$ as a refilled version of the original weight map W. This refilled $\hat{W}$ is used to weighted combine quantized latent features $Y_1^q, \ldots, Y_K^q$ to recover the latent $\hat{Y}$, which is used to reconstruct the output image.

Specifically, the weight filler has the same network structure with the weight predictor in [23], consisting of four residual swin transformer blocks (RSTBs) [22] and a convolution layer to match the channels of weight map and codebook number K.

3.2. Single Codebook Setup

The above weight masking and refilling mechanism can be further optimized when only one codebook is used for each super-pixel. That is, we can further reduce the transmission bits by slightly modifying the weight predictor network, so that we do not need to transfer any weight parameters to the decoder. Specifically, a gumbel softmax layer [16] is added onto the weight predictor so that one-hot weight entry is obtained for each super-pixel indicating the codebook to be used with importance weight as 1. In other words, only the $u \times v \times$ floor($\log_2 K$) bits for codebook indices need to be transmitted to the decoder to retrieve the degraded latent $\tilde{Y}$.

It is worth mentioning that an intuitive alternative of the above single codebook setting is to treat all basis codebooks as one big codebook and skip weight prediction, where one codeword index is assigned to each super-pixel in the combined codebook. However, this alternative does not work in practice since the basis codebooks are learned separately, making it hard to directly compare their codeword features to obtain a cohesive index due to the scale difference.

3.3. Training Process

We adopt the embedding network E^{emb} and the pre-trained semantic-class-dependent basis codebooks $\mathcal{C}_1, \ldots, \mathcal{C}_K$ from AdaCode [23], which partition the latent feature space into non-overlapping cells in K different ways. They are kept fixed during our training process. Then we train the weight predictor, weight filler, the reconstruction network, and the GAN discriminator. On image level, the L1 loss $\mathcal{L}_1(\hat{x}, x)$, the pereptual loss $\mathcal{L}_{per}(\hat{x}, x)$ [18] and the adversarial loss $\mathcal{L}_{adv}(\hat{x}, x)$ [12] are minimized to reduce the distortion between reconstructed $\hat{x}$ and input x. On feature level, the contrastive loss $\mathcal{L}_{con}(\hat{Y}, Y)$ [8] is minimized to regularize the recovered latent $\hat{Y}$. Same as [23], the straight-through gradient estimator [26] is used for back-propagating the non-differentiable vector quantization process during training.

4. Experiments

Experimental Setup Our experiments are based on the JPEG-AI dataset [2,28], which has 5664 images with a large variety of visual content and resolutions up to 8K. The training, validation, and test set have 5264, 350, and 50 images, respectively. The dataset is developed by the JPEG standardization organization to provide standard tools to evaluate NIC methods in the field.

Following similar procedures as AdaCode [23], the training patches have 512×512 resolution, which are firstly randomly cropped from the training images, and then degraded by using the degradation model of BSRGAN [32]. For test evaluation, the maximum resolution of inference tiles is 1080×1080.

The training stage has 200K iterations with Adam optimizer and a batch size of 64, using 8 NVIDIA Tesla V100 GPUs. The learning rate for the generator and discriminator are fixed as 1e-4 and 4e-4, respectively.

Evaluation Metrics For reconstruction distortion, we measure PSNR and SSIM, as well as the perceptual LPIPS [33]. The bitrate is measured by bpp (bit-per-pixel): $bpp = B/h \times w$. The overall bits $B = b_c + b_w$ consist of b_c for transmitting codebook indices $Z_1 \ldots, Z_K$ and b_w for transmitting the sparse weight map $\tilde{W}$. The naive calculation

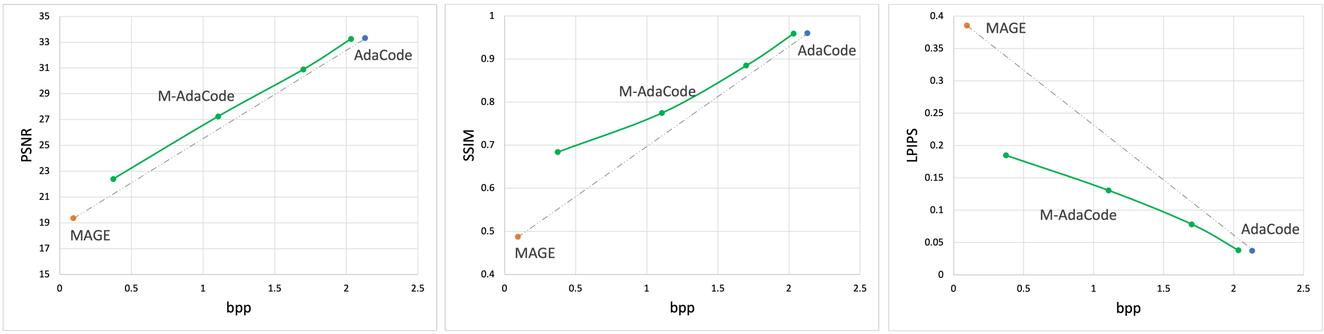


Figure 2. Quantitative comparison with SOTA SVR-based compression methods. **PSNR/SSIM**: the higher, the better. **LPIPS**: the lower, the better. Previous MAGE [21] and AdaCode [23] operate with very low or high bitrates. M-Adacode provides better rate-distortion tradeoffs over a range of bitrates.

is $b_c = u \times v \times \sum_{k=1}^{K} \text{floor}(\log_2 n_k)$ (n_k is the codebook size for $\mathcal{C}_k$). There are many methods to efficiently reduce b_c by losslessly compressing the integer codebook indices, such as [19, 20] with at lease 2× to 3× bit reduction. Reducing b_c is a universal topic for SVR-based compression, which is out of the scope of this paper. We focus on reducing b_w to trade off reconstruction quality for bitrate. For b_w, the required bits for each super-pixel falls into the range of $[\text{floor}(\log_2 K), K \times 16]$, where the minimum $\text{floor}(\log_2 K)$ corresponds to the single-codebook setting discussed in Section. 3.2, and the maximum $K \times 16$ corresponds to AdaCode [23]. For other cases using m codebooks for each super-pixel ($1 < m < K$), we have $b_w = u \times v \times (16 + \text{floor}(\log_2 K)) \times m$.

4.1. Reconstruction performance

Figure 2 gives the rate-distortion comparison of different methods. For M-AdaCode, the performance under 4 settings are tested, where each super-pixel uses $m = 1, \ldots, 4$ codebooks, respectively. The bit counts b_c shown in the figure are computed by simply using the zip software to compress the integer codebook indices, which gives roughly 2× bit reduction comparing to the naive calculation. From the figure, MAGE and AdaCode operate as SVR-based compression methods for extreme scenarios. MAGE targets at a very low bitrate (< 0.1 bpp) with perceptually reasonable generation. AdaCode targets at high reconstruction quality but has a very high bitrate (> 2 bpp). The dotted line connecting these two methods are the conceptual rate-distortion tradeoffs that an SVR-based compression method should be able to provide based on previous methods. As shown in the figure, our M-AdaCode can operate over a wide range of bitrates in between, and can give much better rate-distortion tradeoffs. Table 1 summarizes the performance gains M-AdaCode achieves comparing to the conceptual baseline. Basically, M-Adacode performs much better in terms of SSIM and perceptual LPIPS. The improvements over PSNR are not as significant. This is as expected since the strength of generative methods is to generate rich details to improve perceptual quality, and such rich details do not necessarily match original inputs at the pixel level.

bpp	PSNR	SSIM	LPIPS
0.373	5.3%	23.8%	45.3%
1.016	3.6%	7.2%	60.4%
1.701	1.7%	2.8%	29.5%
2.033	1.8%	2.3%	29.8%

Table 1. Improvements of M-AdaCode over conceptual baseline.

Figure 3 gives some examples of the reconstruction results comparing different methods, for images with different visual content and with different resolutions. The corresponding quantitative performance of these examples are also listed. As clearly shown in the figure, by transferring the full weight map, "AdaCode" can recover rich and accurate details. Using a single codebook per super-pixel, "M-AdaCode 1-codebook" can generate visually pleasing results with reasonable details while preserving good fidelity to the ground-truth. In comparison, using one generic codebook without image-adaptive information, the reconstructed image using "MAGE" presents lots of artifacts or inconsistent details. In many cases, using only two codebooks per super-pixel, "M-AdaCode 2-codebook" can reconstruct images with quite good visual quality.

4.2. Ablation Study

In this section, we investigate the importance of weight filler and the effectiveness of the single codebook setting of Section 3.2. Without the weight filler, the decoder directly uses degraded latent $\tilde{Y}$ to reconstruct output $\hat{x}$. In this case, only the weight predictor and reconstruction network are trained in the training process of Section 3.3. Without the single codebook setting, the same network structure (without gumbel softmax) for the weight predictor is used when only one codebook is kept for each super-pixel, and the bit

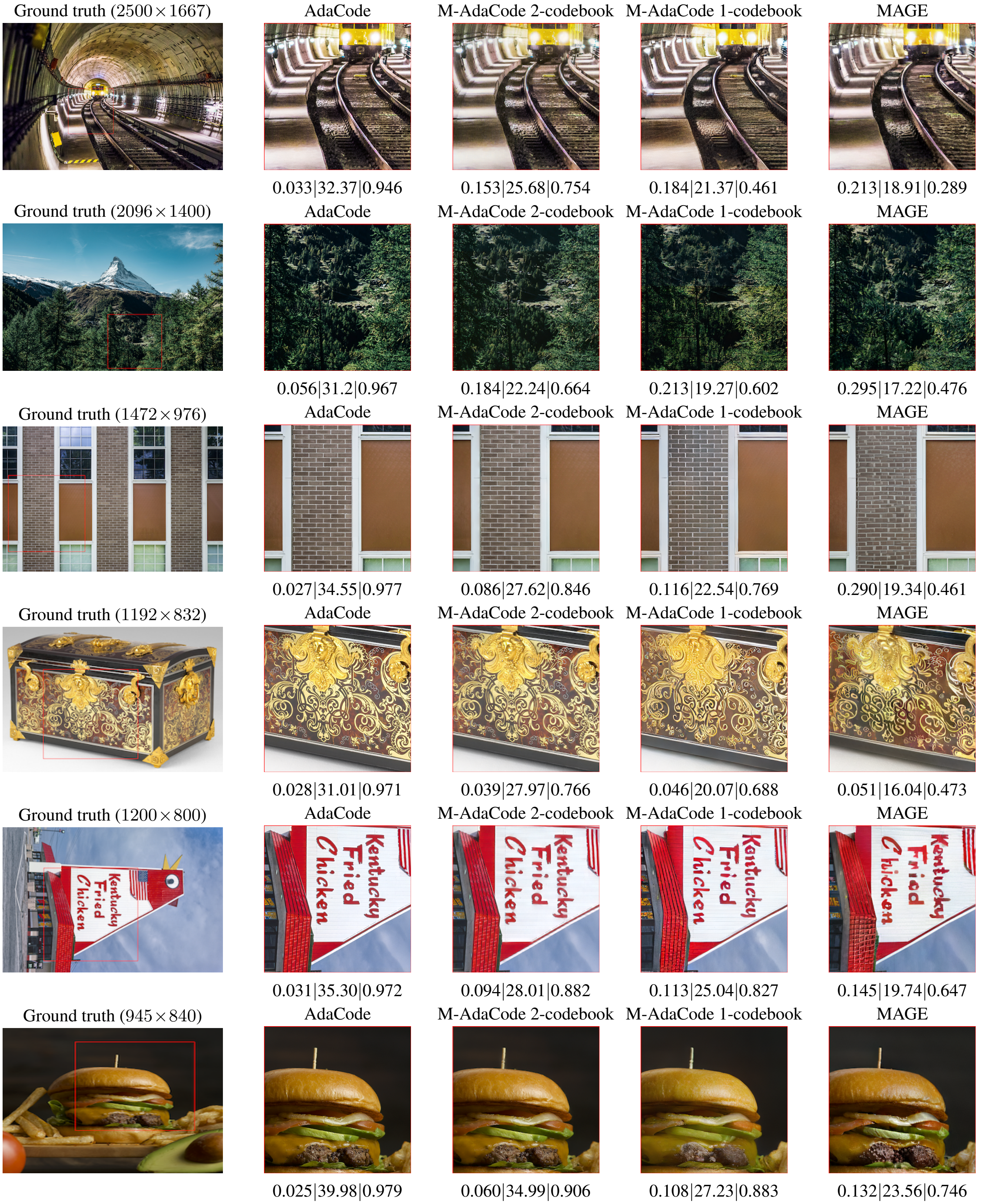


Figure 3. Reconstruction examples. Numbers under each result are "LPIPS|PSNR|SSIM". "M-AdaCode 1-codebook" and "M-AdaCode 2-codebook" are M-AdaCode using 1 codebook or 2 codebooks per super-pixel, respectively.

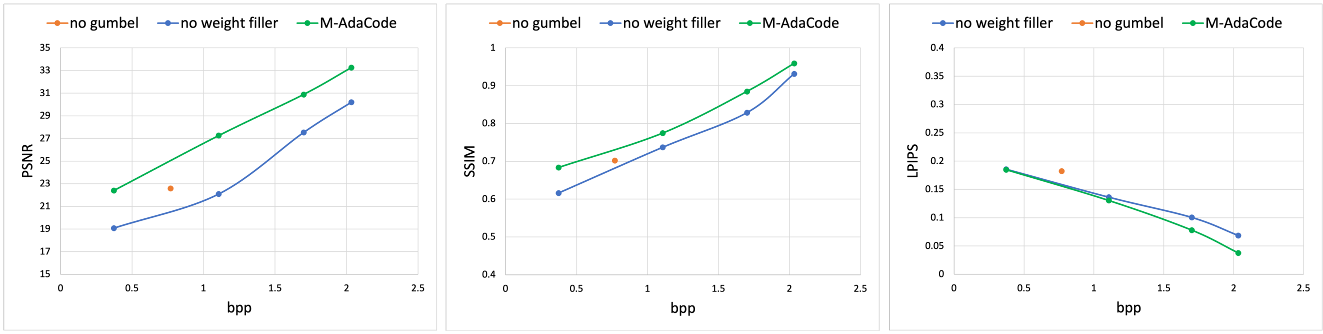


Figure 4. Ablation study: performance without weight filler and performance without single codebook setting. Weight filler can largely improve pixel-level distortion, and the single codebook setting can reduce bitrate without hurting distortion.

count for the weight map is $b_w = u \times v \times (16 + \text{floor}(\log_2 K))$.

Figure 4 gives the performance comparison with M-AdaCode. When one codebook is used for each superpixel, the single codebook setting can achieve equivalent distortion performance with a 52% reduction on bitrate. Using the weight filler, the performance of pixel-level PSNR and SSIM are significantly better than direct reconstruction from the degraded latent feature, especially for lower bitrates. The influence of weight filler reduces as the bitrate increases. As for LPIPS, even without weight filler, by training good reconstruction network the generated image still has reasonable perceptual quality.

4.3. More Discussions

Advantages As mentioned before, SVR-based compression has the advantage of being robust against small transmission and calculation errors across heterogeneous hardware and software platforms. Moreover, the proposed M-AdaCode framework has some additional appealing features.

First, the granularity of the learened basis codebooks to model the separated latent space impacts the reconstruction quality. In general, more basis codebooks with finer granularity give better reconstruction quality, but with a price of larger bitrates. M-AdaCode gives a method to trade off distortion and bitrate. Potentially, we can pretrain many basis codebooks to model the vast visual content space, and customize a limited number of codebooks for each particular data domain based on practical needs.

Second, the dimensionality of the latent feature space, *i.e.*, the codeword feature dimension, also impacts the reconstruction quality. Usually, more dimensions give more representation capacity, leading to better reconstruction but with the price of more storage and computation costs. When the codeword feature dimension increases, M-AdaCode does not increase the bitrate by transferring codeword indices. So potentially, we can use rich representation with large feature dimensions, as long as being permitted by the computation and storage requirements.

Limitations As a generative image modeling method, the SVR-based compression has a competing goal of generative visual quality and pixel-level fidelity to the input. This is an advantage when the input has low or mediocre quality, especially when the input has degradations. In such cases, the target can be interpreted as to restore the conceptual high-quality clean input from the degraded version, and using high-quality codewords is robust to recover good visual details. However, when the input has ultra-high quality, the generated details may be inconsistent to the input and may hurt the performance, since in such cases the target is to recover the exact input itself. Therefore, in practical usage, it may be hard for a particular method to work universally better than others, and we may need to selectively choose which method to use when compressing images with different quality and different content.

5. Conclusion

We propose an SVR-based image compression method, M-AdaCode, by using masks over the latent feature subspace to balance bitrate and reconstruction quality. The encoder embeds images into discrete latent subspaces spanned by multiple basis codebooks that are learned in a semantic-class-dependent fashion, and transfers integer codeword indices that are efficient and cross-platform robust. By deriving image-adaptive weights to combine the basis codebooks, a rich latent feature can be recovered for high quality reconstruction. Using the redundant information in the latent subspaces, unimportant weights can be masked out in the encoder and recovered later in the decoder, to trade off reconstruction quality for transmission bits. The masking rate controls the balance between bitrate and distortion. Experiments over the standard JPEG-AI dataset show that comparing to previous SVR-based compression methods that operate over very low or very high bitrates, our M-AdaCode achieves better rate-distoration tradeoffs over a large range of bitrates.

References

[1] E. Agustsson, M. Tschannen, F. Mentzer, R. Timofte, and L. Van Gool. Generative adversarial networks for extreme learned image compression. In *ICCV*, 2019. 2

[2] J. Ascenso, P. Akyazi, F. Pereira, and T. Ebrahimi. Learning-based image coding: early solutions reviewing and subjective quality evaluation. *SPIE Photonics Europe - Optics, Photonics and Digital Technologies for Imaging Applications VI*, 2020. 2, 5

[3] J. Balle, D. Minnen, S. Singh, S. Hwang, and N. Johnston. Variational image compression with a scale hyperprior. In *ICLR*, 2018. 1, 2

[4] J. Ballé, N. Johnston, and D. Minne. Integer networks fro data compression with latent-variable models. In *ICLR*, 2019. 1, 2

[5] Y. Blau and T. Michaeli. The perception-distortion tradeoff. In *CVPR*, 2018. 2

[6] Y. Blau and T. Michaeli. Rethinking lossy compression: The rate-distortion-perception tradeoff. *arXiv preprint: arXiv:1901.07821*, 2019. 2

[7] C. Chen, X. Shi, Y. Qin, X. Li, X. Han, T. Yang, and S. Guo. Real-rorld blind super-resolution via feature matching with implicit high-resolution priors. In *ACM MM*, 2022. 2

[8] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020. 5

[9] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto. Learned image compression with discretized gaussian mixture likelihoods and attention modules. In *CVPR*, 2020. 2

[10] A. El-Nouby, M. Muckle, K. Ullrich, I. Laptev, J. Verbeek, and H. Jegou. Image compression with product quantized masked image modeling. *arXiv preprint: arXiv:2212.07372*, 2022. 4

[11] P. Esser, R. Rombach, and B. Ommer. Taming transformers for high-resolution image synthesis. In *CVPR*, 2021. 2

[12] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NeurIPS*, 2014. 5

[13] D. He, Y. Zheng, B. Sun, Y. Wang, and H. Qin. Checkerboard context model for efficient learned image compression. In *CVPR*, 2021. 1, 2

[14] K. He, X. Chen, S. Xie, Y. Li amd mP. Dollar, and R. Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022. 2, 4

[15] Z. Huang, X. Jin, C. Lu, Q. Hou, M. Cheng, D. Fu, X. Shen, and J. Feng. Contrastive masked autoencoders are stronger vision learners. In *ICLR*, 2023. 4

[16] E. Jiang, S. Gu, and B. Poole. Categorical reparameterization with gumbel-softmax. In *ICLR*, 2017. 5

[17] W. Jiang, H. Choi, and F. Racape. Adaptive human-centric video compression for humans and machines. In *CVPRW*, 2023. 2, 4

[18] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016. 5

[19] D. Lemire and L. Boytsov. Decoding billions of integers per second through vectorization. *arXiv preprint, arXiv:1209.2137*, 2021. 6

[20] D. Lemire, L. Boytsov, and N. Kurz. Simd compression and the intersection of sorted integers. *arXiv preprint,arXiv:1401.6399*, 2020. 6

[21] T. Li, H. Chang, S. Mishra, H. Zhang, D. Katabi, and D. Krishna. Mage: Masked generative encoder to unify representation learning and image synthesis. In *CVPR*, 2023. 2, 3, 4, 6

[22] J. Liang, J. Cao, G. Sun, K. Zhang, L. Van Gool, and R. Timofte. Swinir: Image restoration using swin transformer. In *ICCV*, 2021. 5

[23] K. Liu, Y. Jiang, I. Choi, and J. Gu. Learning image-adaptive codebooks for class-agnostic image restoration. In *ICCV*, 2023. 2, 3, 4, 5, 6

[24] F. Mentzer, G. Toderici, D. Minnen, S. Hwang, S. Caelles, M. Lucic, and E. Agustsson. Vct: A video compression transformer. *arXiv preprint: arXiv:2206.07307*, 2022. 1, 2

[25] F. Mentzer, G. Toderici, M. Tschannen, and E. Agustsson. High-fidelity generative image compression. In *NeurIPS*, 2020. 2

[26] A. Oord, O. Vinyals, and K. Kavukcuoglu. Neural discrete representation learning. In *NeurIPS*, 2017. 5

[27] A. Van Den Oord and O. Vinyals amd K. Kavukcuoglu. Neural discrete representation learning. In *NeurIPS*, 2017. 2

[28] White paper. Iso/iec jtc 1/sc29/wg1 n90049 white paper on jpeg ai scope and framework v1.0. 2021. 5

[29] Y. Qian, M. Lin, X. Sun, and Z. Tanand R. Jin. Entroformer: A transformer-based entropy model for learned image compression. *arXiv preprint: arXiv:2202.05492*, 2022. 1, 2

[30] T. Wang, A. Mallya, and M. Liu. One-shot free-view neural talking-head synthesis for video conferencingr. In *CVPR*, 2021. 4

[31] Z. Wang, J. Zhang, R. Chen, W. Wang, and P. Luo. Restoreformer: High-quality blind face restoration from undegraded key-value pairs. In *CVPR*, 2022. 2

[32] K. Zhang, J. Liang, L. Van Gool, and R. Timofte. Designing a practical degradation model for deep blind image super-resolution. In *ICCV*, 2021. 5

[33] R. Zhang, P. Isola, A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 5

[34] Z. Zheng, X. Wang, X. Lin, and S. Lv. Get the best of the three worlds: Real-time neural image compression in a non-gpu environment. In *ACM MM*, 2021. 1, 4

[35] S. Zhou, K. Chan, C. Li, and C.C. Loy. Towards robust blind face restoration with codebook lookup transformer. In *NeurIPS*, 2022. 2, 4

[36] R. Zou, C. Song, and Z. Zhang. The devil is in the details: Window-based attention for image compression. In *CVPR*, 2022. 2