Vector Quantized Variational Image Compression

Timur Akhtyamov [1]

Abstract

During recent years, deep learning-based data compression, especially image and video compression, gained a popularity due to the content-adaptive properties of the neural networks and their overall generative abilities. Still, outperforming classical image codecs remains quite challenging task. In this work we explore the potential of one of the specific class of models - vector quantized variational autoencoders (VQ-VAE) - to the task of variational image compression. We provide several experiments and analyses of their results, making our conclusions on potential applications of such models to the practical image compression.

1. Introduction

In this section, we provide a brief introduction to the area of image compression, paying additional focus on the entropy coding part. We then provide a brief overview of the modern deep image compression methods by introducing main mathematical concepts from founding papers and a short glance at recent papers in this area.

1.1. Image compression

Data compression, and particularly image compression, plays crucial role in telecommunication and multimedia, since one can easily show that representing images in the raw format will require enormous amount of storage even for simple media content. That's why the problem of data compression dates back to the early days of computer science (Shannon, 1948).

A general pipeline of the image compression (which is also actual for the deep image compression) is shown at Fig. 2. Roles of the analysis and synthesis transforms are dual: on one hand, they produce data format that will be later passed to the encoder in a way that achieves more efficient compression, and on the other hand, those transforms may be viewed in some cases as dimensionality reduction. Encoding and decoding is performed using the wide apparatus of the Information and Coding theory. What is important here is that most of the coding methods assume discrete data, usually called symbols.

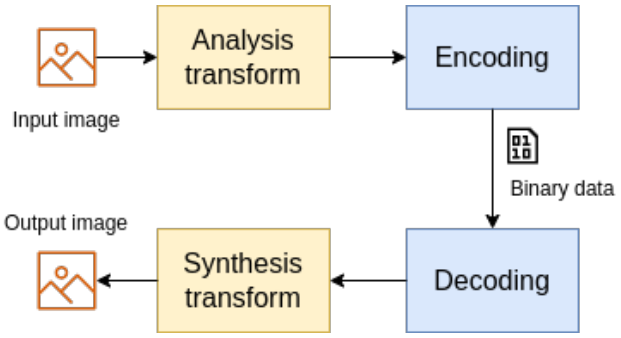

Figure 1. General scheme of the image compression.

For today, a wide range of classical image compression methods are used, which can perform lossless (e.g. without distortion of the original data) and lossy (with the original data distortion in sake of larger compression ratio). Among them, the most used are JPEG(lossy), PNG (lossless), BPG (lossy).

1.2. Entropy coding

It is important to pay a little more attention to the coding part, since on practice it is the part where the most of the compression is achieved. Entropy coding is a large family of coding algorithms that rely on Shannon's source coding theorem that establishes theoretical lower bound on amount of data required to compress a stream of i.i.d. random variables can be expressed via Shannon's information entropy of the source:

$$H(X) = \mathbb{E}\left[-\log p(X)\right]. \tag{1}$$

One of the common properties of the entropy coding methods is that more common symbols are represented with fewer bits.

One of the most famous approaches in this area is the Huffman coding (Huffman, 1952), whose variants like Deflate is

[1]Skolkovo Institute of Science and Technology, Moscow, Russia.. Correspondence to: Timur Akhtyamov <timur.akhtyamov@skoltech.ru>.

widely used in codecs like JPEG and PNG. On practice, it often performs suboptimally, and that's why a set of arithmetic coding approaches were developed, which can use arbitrary data probability distribution and achieve entropy of less then 1 bit per symbol. Its successors, asymmetric numeral systems (ANS), combine the ideas of both arithmetic coding and Huffman coding (Duda, 2013).

Usually, most of the works assume that entropy coding is performed by an external module, which requires as an input distribution of the symbols. One can use independent distribution of the symbols, or assume an autoregressive approach. For example, one knows that in unconditional distribution of the English alphabet symbols letter "a" has high probability, but if in the message previous letter was "a", it means that probability of observing letter "a" again is very low now. In this concept, previous data that is used to adjust distribution is called context, and allows to reduce required bits, which follows from the Shannon's theorem.

At this point, one may notice that deep learning models, especially generative models, may fit into this pipeline: they potentially can perform both dimensionality reduction and estimation of the data distribution that can be used in the entropy coding.

1.3. Variational image compression

In founding works in the area of image compression (Ballé et al., 2016; Theis et al., 2017) deep image compression problem was formulated as a rate-distortion (RD) optimization problem:

$$\mathcal{L} = R + \lambda D, \tag{2}$$

where $\mathcal{L}$ is the target loss function, R and D are the rate and distortion measures and λ is the rate-distortion trade-off coefficient which is usually set manually to vary how much distortion one may allow in sake of rate savings. Distortion is usually expressed in terms of mean squared error (MSE), and rate measure is based on Shannon's entropy:

$$R = \mathbb{E}_{\hat{\mathbf{y}}\sim m(\hat{\mathbf{y}})}\left[-\log_2 p_{\hat{\mathbf{y}}}(\hat{\mathbf{y}})\right], \tag{3}$$

where $\hat{\mathbf{y}}$ is the discrete representation of some vector $\mathbf{y}$, $m(\hat{\mathbf{y}})$ is the underlying distribution of data, and $p_{\hat{\mathbf{y}}}(\hat{\mathbf{y}})$ is called the entropy model that is later used for the entropy coding. A typical pipeline is to use autoencoder models with encoder $g_a(\mathbf{x})$ (analysis transform) and decoder $g_s(\hat{\mathbf{y}})$ (synthesis transform) to obtain representation $\mathbf{y}$ of the image $\mathbf{x}$, and discrete representation $\hat{\mathbf{y}}$ is obtained via round quantization. To keep this discretization differentiable, a common practice proposed in (Ballé et al., 2015) is to substitute round operation with adding uniform noise $\mathcal{U}(-0.5, 0.5)$ which just makes model robust to rounding perturbations.

Succeeding work (Ballé et al., 2018) de facto laid a foundation for the recent deep image compression models. Authors represented the rate-distortion optimization problem as a variational inference problem. The synthesis transform can be viewed as a generative model which generates an image from latent representation, and analysis transform can be viewed as an inference model which infers the latent representation from the image. Thus, if we write some true posterior $p_{\tilde{\mathbf{y}}|\mathbf{x}}(\tilde{\mathbf{y}}|\mathbf{x})$ and parametric variational density $q(\tilde{\mathbf{y}}|\mathbf{x})$, expectation of the Kullback-Leibler divergence that is minimized over the data distribution $p(\mathbf{x})$ can be written as:

$$\begin{aligned}\mathbb{E}_{\mathbf{x}\sim p_{\mathbf{x}}} D_{\mathrm{KL}}\left[q||p_{\tilde{\mathbf{y}}|\mathbf{x}}\right] = \mathbb{E}_{\mathbf{x}\sim p_{\mathbf{x}}}\mathbb{E}_{\tilde{\mathbf{y}}\sim q}\left[\log q(\tilde{\mathbf{y}}|\mathbf{x})\right. \\ \left. - \log p_{\mathbf{x}|\tilde{\mathbf{y}}}(\mathbf{x}|\tilde{\mathbf{y}}) - \log p_{\tilde{\mathbf{y}}}(\tilde{\mathbf{y}}) + \text{const.}\right. \end{aligned} \tag{4}$$

We now can break down the terms. The first term, due to the uniform noise pseudo-quantization described above, can be written as:

$$q(\tilde{\mathbf{y}}|\mathbf{x}, \phi_g) = \prod_i \mathcal{U}(\tilde{y}_i|y_i - 0.5, y_i + 0.5), \tag{5}$$

where g_a is the analysis transform parametrized by parameters ϕ_g, $\mathbf{y} = g_a(\mathbf{x})$ is the unquantized latent representation of the image, and $\tilde{\mathbf{y}}$ is the noised latent representation, which is during the actual compression process will be substituted by rounded representation $\hat{\mathbf{y}}$. Due to the usage of the uniform distribution, it can be technically ignored in the KL expression. Next, if we assume that the second term has a Gaussian nature in the form:

$$p_{\mathbf{x}|\tilde{\mathbf{y}}}(\mathbf{x}|\tilde{\mathbf{y}}) = \mathcal{N}(\mathbf{x}|\tilde{\mathbf{x}}, (2\lambda)^{-1}\mathbf{1}), \tag{6}$$

where $\tilde{\mathbf{x}} = g_a(\tilde{\mathbf{y}})$ is the output of the analysis transform g_a parametrized by θ_g, then minimization of this KL term will be equal to the minimization of the weighed MSE, as in the original expression of the RD loss. Finally, the third term represents the differential entropy, which is in practice can be a close approximation the Shannon's discrete entropy, and thus this term corresponds to the rate term in the RD objective.

Going further, authors report that $\hat{\mathbf{y}}$ still contains spatial correlations. To model and capture those dependencies, a set of random variables $\tilde{\mathbf{z}}$ is introduced, conditioned on which initial latent variables are assumed to be independent. In the proposed model, each element $\tilde{y}_i$ is modeled as a zero-mean Gaussian variable with its own standard deviation σ_i that is defined by parametric transform $\tilde{\sigma} = h_s(\tilde{\mathbf{z}})$ parametrized by θ_h:

$$p_{\tilde{\mathbf{y}}|\tilde{\mathbf{z}}}(\tilde{\mathbf{y}}|\tilde{\mathbf{z}}) = \prod_i \left(\mathcal{N}(0, \tilde{\sigma}_i^2) * \mathcal{U}(-0.5, 0.5)\right). \tag{7}$$

This concept is known as hyper prior. In this case, the model for the compression of the $\mathbf{y}$ representation is updated in proposed way, and since $\mathbf{z}$ is also needs to be transmitted, the

same model that was used for $\mathbf{y}$ is re-used. The hyper prior model is built on top of the analysis and synthesis transform, introducing hyper encoder h_a and hyper decoder h_s as a layers stacked on top of the main encoder and decoder g_a and g_s. Regarding the RD objective, in this case we just add a second rate term that is responsbile for the $\mathbf{z}$ compression rate.

Later, several extensions of this model were introduced, including mean-scale and autoregressive entropy models (Minnen et al., 2018), models based on architectures beyond standard convolutional networks like transformers (Lu et al., 2021); also there there are works that consider learning the entropy models completely separately from the main encoders (Qian et al., 2022).

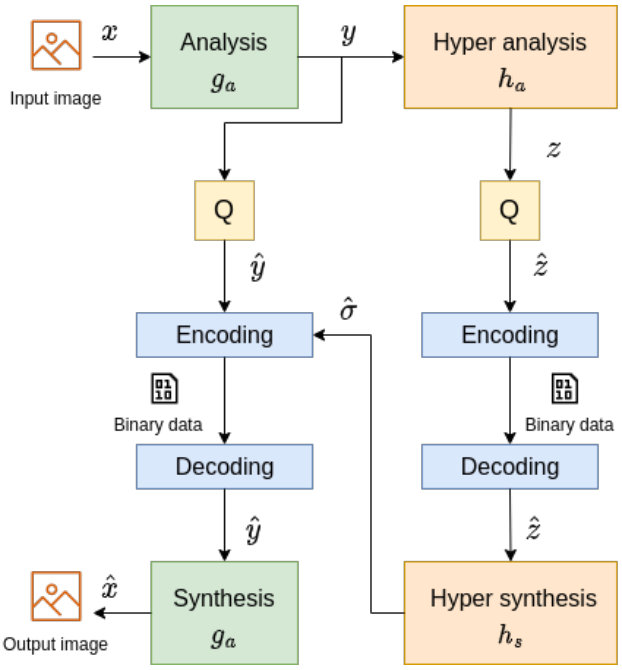

Figure 2. Variational image compression with hyperprior scheme. "Q" stands for quantization.

2. Vector Quantized Variational Autoencoders

Vector Quantized (VQ) Autoencoders is a family of generative models that operate in the discrete latent vectors space. This family is mostly repesented by VQ-VAE (Van Den Oord et al., 2017) and VQ-VAE-2 models (Razavi et al., 2019), also some extensions exist, for example as parts of larger models, like DALL-E (Ramesh et al., 2021).

Main idea of the VQ-VAE is to learn discrete embedding space by approximating initial latent vector with its nearest neighbour from the finite set of learned vectors called the code book. Learning of this finite set is performed using exponential moving average technique. The original version of VQ-VAE suffered from the quite low quality of generated and reconstructed images. The next version of the model, VQ-VAE-2, tackles this issue using two latents (from different code books): the top latent, which models the global information from the image, and the bottom latent, conditioned on top latent, which handles local details of the image. With this approach, better quality of reconstructions and samples was achieved. In both of the models, prior is usually trained separately from the autoencoder. In VQ-VAE-2, the PixelSNAIL (Chen et al., 2018) autoregressive model is used to model priors, and trained separately for the bottom and top latents.

Our interest in the VQ models is dual. Firstly, application of the VQ models may dramatically reduce the dimensionality of the transmitted data. Despite in practice dimensionality reduction in deep image compression models plays auxiliary role (in fact, many models consider latent vectors dimensionality which is close to the dimensionality of the original image) and most of the compression is been contributed by the entropy model, in case of VQ role of the dimensionality reduction may become more significant, since now we do not need to transmit whole latent components, but only indices of its spatial components (assuming that transmitter and receiver already know the code book). For example, if one consider VQ-VAE model with input image (in PyTorch notation) of size $(3, 256, 256)$, and latent vector of size $(192, 64, 64)$ where 192 is one of the common channels dimension for the compression models, and code book of size 512, this latent can be transmitted using the $(64, 64)$ table of 16-bit integers, which out of the box gives the number of $(16 * 64 * 64)/(256 * 256) = 1$ bits per pixel. Further entropy coding may reduce this number even more. To handle the more or less acceptable image quality, we proceed with VQ-VAE-2 model.

Secondly, an interesting observation can be seen regarding the VQ-VAE-2 model. The idea of the top and bottom latents resembles the idea of hyper prior from the previous section. As in the case of hyper prior, the top encoder is built on top of the bottom encoder, and the top latent is used as a condition for the distribution model of the bottom latent. Thus, for us it is interesting to see how this architecture will show itself in the image compression task.

3. Experiments and results

3.1. Models

In our experiments, we consider following models and their modifications:

- VQ-VAE-2 + PixelSNAIL as an entropy model (combines convolutions and attention layers)

- VQ-VAE-2 + Transformer-based block-autoregressive model inspired by (Lu et al., 2021)
- Conv-HyperPrior: Generic convolution scale hyper prior compression model

More details on the VQ models modifications are presented in Table 1. In the indices input mode, indices first converted to one-hot encoded vectors and then processed.

All or these three models share the same main autoencoders architecture which is inspired by the work (Mentzer et al., 2022) and uses combination of convolutions and attention layers. For the models based on VQ-VAE-2, main encoder, top and bottom entropy models are trained separately as proposed in the original work on VQ-VAE-2. Convolutional scale hyper prior model is trained end-to-end with additional post-training fine-tuning of the hyper entropy model.

Entropy coding for VQ-VAE-2 based models is performed with the torchac library, for the convolutional scale hyper prior model CompressAI framework (Bégaint et al., 2020) is used. For both of the solutions, resulting bit rate diverges from the Shannon's entropy based estimation only staring from the 3d decimal. Thus, since the entropy coding is a slow process, especially for the autoregressive models, in the bit rate evaluation we proceed with Shannon's entropy based estimation.

3.2. Datasets

Training of the models is performed using the Vimeo90k (Xue et al., 2019) dataset. We take the first frames from each clip and perform 256×256 center crop, resuling in a diverse images dataset. Evaluation of the results is performed using the Kodak dataset, which is a collection of 24 losslessly compressed images widely used for both quantitative and qualitative evaluation of the deep image compression models.

Training time differs for our models. For example, using the A100 Nvidia GPU, convolutional model can be trained on Vimeo90k relatively fast, but for the VQ-VAE-2 based models training takes from 12-24 hours and even more for some settings.

Also it is important to notice that while for the convolutional model, as well as for the most of the deep image compression models, continuous distributions are used, while VQ-VAE-2 based models use discrete distributions.

For all models, target image size is $(256, 256)$.

3.3. Metrics

In the image compression community, two main aspects are evaluated: distortion and compression rate. Distortion is usually expressed in terms of peak signal to noise ratio (PSNR), which directly depends on MSE, and compression rate is expressed via bits per pixel (bpp) metric, which can be intuitively understood as a "density" of the compression:

$$\text{bpp} = \frac{\text{Total number of compressed bits}}{\text{Number of pixels in image}}.$$

For example, for any RGB image stored in raw format with 8-bit integers the bpp metric is equal to 24.

3.4. Results

Evaluation is performed on the test subset of the Vimeo90k dataset and on Kodak dataset. For Vimeo90k $(256, 256)$ central crops are used, for Kodak whole images are used (they first split into patches and passed independently through the network, and then reconstructed patches assembled back). Results are presented in Tables 2 and 3. Fig. 3 also shows the reference performance of the pre-trained on Vimeo90k deep image compression models, available in CompressAI model zoo. As authors state, those models were trained and fine-tuned for up to 12 days.

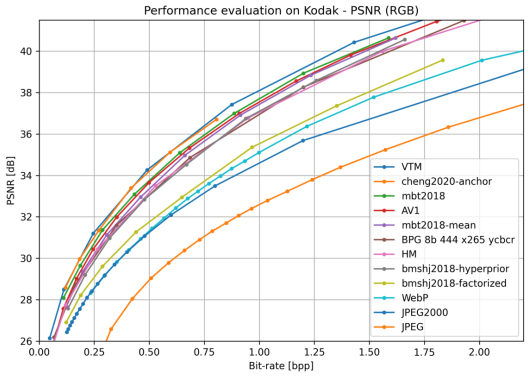

Figure 3. Performance of the different deep image codecs from CompressAI model zoo on Kodak dataset. Image from (Bégaint et al., 2020).

Comparing models to each other on Vimeo dataset, we see that we VQ models outperform convolutional model. Despite such comparison may be unfair since extensive fine-tuning which requires a lot of time and resources was not performed, such initial results may still be informative. Comparing models to each other, we see that PixelSNAIL outperforms transformer, potentially due to the complexity of the last one which requires more sophisticated training procedure. For PixelSNAIL, increasing the latent vectors dimensionality resulted in both PSNR and BPP increments, as expected.

Results on Kodak datasets first of all tell about quite poor generalization of the VQ models. In their case, both dis-

Table 1. Modifications of the VQ models.

NAME	CODE BOOK SIZE	LATENT DIM	ENTROPY MODEL INPUT
VQ-PIXELSNAIL-1	512	64	INDICES
VQ-PIXELSNAIL-2	2048	256	VECTORS
VQ-PIXELSNAIL-3	512	64	VECTORS

Table 2. Results on the Vimeo90k test subset.

MODEL	PSNR (DB)	BPP
VQ-PIXELSNAIL-1	35.78	0.38
VQ-PIXELSNAIL-2	36.45	0.43
VQ-PIXELSNAIL-3	34.30	0.69
CONV-HYPERPRIOR	33.58	0.66

Table 3. Results on the Kodak dataset.

MODEL	PSNR (DB)	BPP	BPP (ZLIB)
VQ-PIXELSNAIL-1	29.99	0.47	0.77
VQ-PIXELSNAIL-2	30.62	0.55	0.90
VQ-PIXELSNAIL-3	28.91	0.70	0.74
CONV-HYPERPRIOR	32.17	1.11	-

tortion and rate degradation is observed, and one of the potential reason for that is the nature of the vector quantized code books. Regarding the Conv-HyperPrior model, in its case distortion drop was not so significant, but rate has failed dramatically. This tells about easier generalization of such convolutional main autoencoder and requirement for the more sophisticated fine-tuning of the entropy models.

For the Kodak dataset, we also perform comparison with the naive compression based on Python's built-in zlib compression methods (which are based on Deflate algorithm). In this case, we extract top and bottom arrays with indices (in 16-bit integer format) and pass it to zlib methods with maximum compression rate. The effect of the VQ models is that such simple compression method shows results compared with sophisticated entropy coding methods. As a result, if one manages to obtain VQ autoencoder that produces very high reconstruction quality (e.g. PSNR ≥ 38.00 dB), it may automatically become SOTA even with simple zlib compression.

Again, on Kodak dataset our models perform much worse then professionally trained and fine-tuned CompressAI model zoo and classical image codecs like VTM. But still, in terms of Vimeo90k test subset performance is good, and even a little bit better generalization may produce results closer to the SOTA.

4. Conclusions

In this project, we provided a proof-of-concept of the application of vector quantized autoencoders to the task of variational image compression. Main outcome here is that VQ models have a potential in this area, and for now main challenge here is the overfitting and poor generalization. Potentially this can be solved with adaptive vector quantization, e.g. using quantized vectors as the most part of the data, but keeping some space to to store some continuous data that defines adaptive transformations of the quantized vectors specific to the image content. Tackling those issues is a promising topic for the furure research.

References

Ballé, J., Laparra, V., and Simoncelli, E. P. Density modeling of images using a generalized normalization transformation. *arXiv preprint arXiv:1511.06281*, 2015.

Ballé, J., Laparra, V., and Simoncelli, E. P. End-to-end optimized image compression. *arXiv preprint arXiv:1611.01704*, 2016.

Ballé, J., Minnen, D., Singh, S., Hwang, S. J., and Johnston, N. Variational image compression with a scale hyperprior. *arXiv preprint arXiv:1802.01436*, 2018.

Bégaint, J., Racapé, F., Feltman, S., and Pushparaja, A. Compressai: a pytorch library and evaluation platform for end-to-end compression research. *arXiv preprint arXiv:2011.03029*, 2020.

Chen, X., Mishra, N., Rohaninejad, M., and Abbeel, P. Pixelsnail: An improved autoregressive generative model. In *International Conference on Machine Learning*, pp. 864–872. PMLR, 2018.

Duda, J. Asymmetric numeral systems: entropy coding combining speed of huffman coding with compression rate of arithmetic coding. *arXiv preprint arXiv:1311.2540*, 2013.

Huffman, D. A. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101, 1952.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference*

Figure 4. Example of the Kodak image reconstruction by VQ-PixelSNAIL-1 model. Top: original image, bottom: reconstructed image. For this image, BPP is 0.59 and PSNR is 27.74.

on Machine Learning (ICML 2000), pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Lu, M., Guo, P., Shi, H., Cao, C., and Ma, Z. Transformer-based image compression. *arXiv preprint arXiv:2111.06707*, 2021.

Mentzer, F., Toderici, G., Minnen, D., Hwang, S.-J., Caelles, S., Lucic, M., and Agustsson, E. Vct: A video compression transformer. *arXiv preprint arXiv:2206.07307*, 2022.

Minnen, D., Ballé, J., and Toderici, G. D. Joint autoregressive and hierarchical priors for learned image compression. *Advances in neural information processing systems*, 31, 2018.

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

Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pp. 8821–8831. PMLR, 2021.

Razavi, A., Van den Oord, A., and Vinyals, O. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, 2019.

Shannon, C. E. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

Theis, L., Shi, W., Cunningham, A., and Huszár, F. Lossy image compression with compressive autoencoders. *arXiv preprint arXiv:1703.00395*, 2017.

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

Xue, T., Chen, B., Wu, J., Wei, D., and Freeman, W. T. Video enhancement with task-oriented flow. *International Journal of Computer Vision*, 127(8):1106–1125, 2019.