GAN for image compression

ECE 285: Video and Image Compression

Presenter: You-Yi Jau, M.S. in UCSD

GENERATIVE ADVERSARIAL NETWORKS FOR EXTREME LEARNED IMAGE COMPRESSION

Eirikur Agustsson*, Michael Tschannen*, Fabian Mentzer,* Radu Timofte & Luc Van Gool
{aeirikur, mentzerf, radu.timofte, vangool}@vision.ee.ethz.ch, michaelt@nari.ee.ethz.ch
ETH Zurich

CVPR 2018

Generative adversarial networks for extreme learned image compression
E Agustsson, M Tschannen, F Mentzer, R Timofte… - arXiv preprint arXiv …, 2018

Outlines

- Motivation
- Related works
- Method overview
- Method details
- Experiments
- Discussion

Compressed images

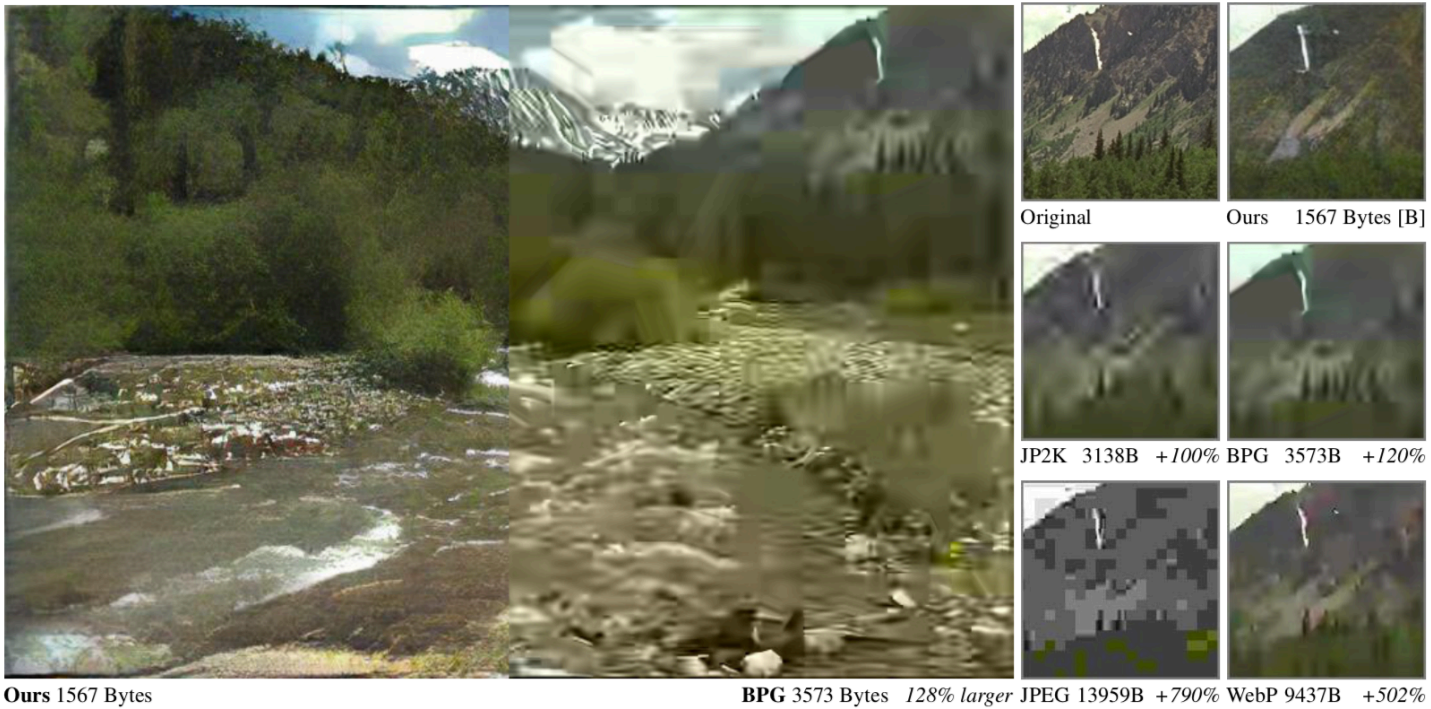


Figure 1: Visual comparison of our result to that obtained by other codecs. Note that even when using more than twice the number of bytes, all other codecs are outperformed by our method visually.

Motivation and problem description

- How to compress images?
  - Deep compression systems works competitive with engineered codecs
    - WebP, JPEG2000, BPG (State-of-the-art)
  - Previous deep compression systems are optimized for traditional distortion metrics
    - peak signal-to-noise ratio (PSNR), multi-scale structural similarity (MS-SSIM)
  - How to achieve very low bitrates (below 0.1 bits per pixel (bpp))?
- Problem description
  - Input: image
  - Compressed Codes: hidden layer
  - Output: Compressed image

AutoEncoder: Rate-Distortion optimization

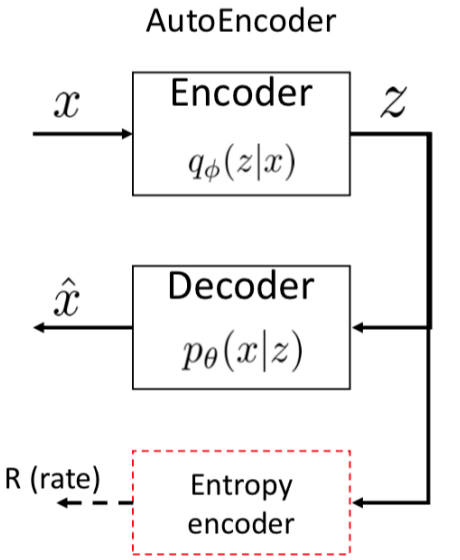
- Let's consider Maximum likelihood reconstruction with the Gaussian model
- Then, the distortion D is

$$D = \mathbb{E}_{q(z|x)}\Big[-\log p_\theta(x|z)\Big]$$

- To encode Z, use a known prior p(z)
- Rate of Z, $\mathbb{E}_{q(z|x)}\Big[-\log p(z)\Big]$
- However, If we apply neighbor x information to estimate z, we can reduce the entropy of z

$$R(z) = \mathbb{E}_{q(z|x)}\Big[-\log p(z) + \log q_\phi(z|x)\Big] = \mathbb{E}_{q(z|x)}\Big[\frac{q_\phi(z|x)}{p(z)}\Big] = \mathbb{D}_{\mathrm{KL}}(q_\phi(z|x)\ ||\ p(z))$$

$$\min_{\theta,\phi} \underbrace{\mathbb{E}_{q(z|x)}\Big[-\log p_\theta(x|z)\Big]}_{D(\theta)} + \lambda \underbrace{\mathbb{D}_{\mathrm{KL}}(q_\phi(z|x)\ ||\ p(z))}_{R(\phi)}$$



$$\min_{\theta,\phi} D(\theta) + \lambda \cdot R(\phi)$$

22

Related works

- Traditional methods
  - WebP
  - JPEG2000
  - **BPG** (Better Portable Graphics)
    - the state-of-the-art engineered codec
    - Created by Fabrice Bellard in 2014
- Deep methods
  - Auto-encoder
  - Variational Auto-encoder (VAE)
  - Recurrent neural network (RNN)
  - Generative Adversarial Network (GAN)

Deep learning methods in recent years

- Conditional probability models for deep image compression
  - F Mentzer, E Agustsson, M Tschannen, R Timofte… - Proceedings of the IEEE …, 2018
- Full resolution image compression with **recurrent neural networks**
  - G Toderici, D Vincent, N Johnston, S Jin Hwang… - Proceedings of the IEEE …, 2017
- Lossy image compression with compressive **autoencoders**
  - L Theis, W Shi, A Cunningham, F Huszár - arXiv preprint arXiv:1703.00395, 2017
- End-to-end optimized image compression
  - J Ballé, V Laparra, EP Simoncelli - arXiv preprint arXiv:1611.01704, 2016
- Real-time adaptive image compression
  - O Rippel, L Bourdev - Proceedings of the 34th International Conference on …, 2017
- Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding
  - S Han, H Mao, WJ Dally - arXiv preprint arXiv:1510.00149, 2015
- **Generative adversarial networks for extreme learned image compression**
  - E Agustsson, M Tschannen, F Mentzer, R Timofte… - arXiv preprint arXiv …, 2018

Related works

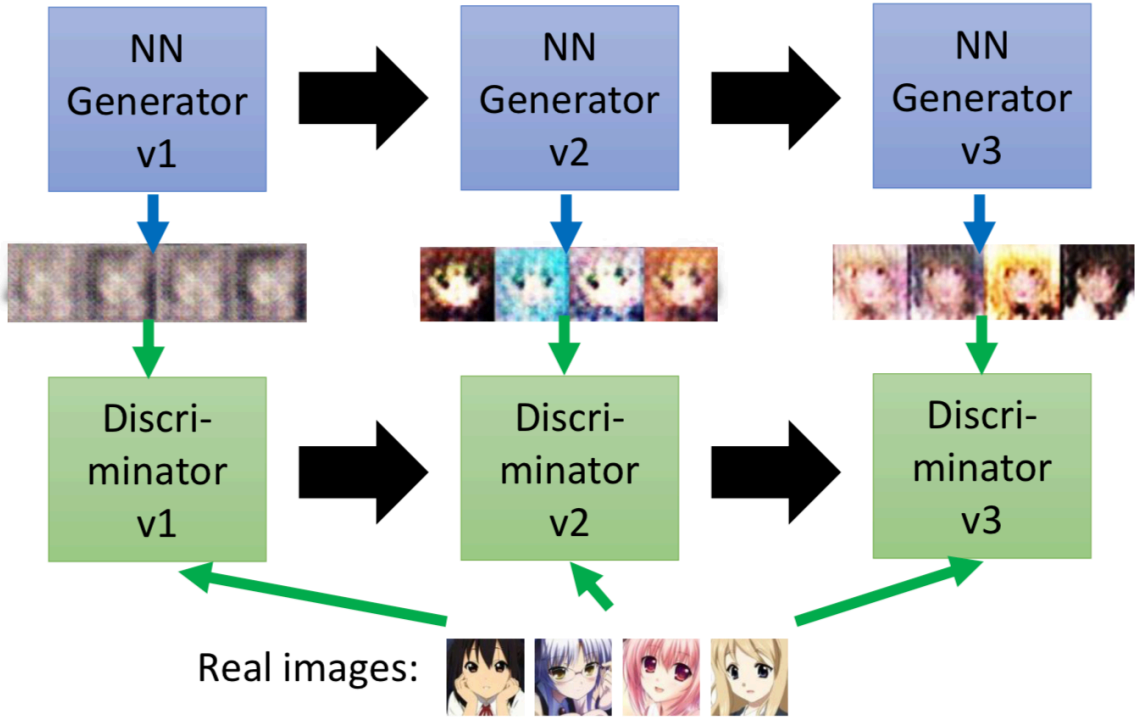
- Generative Adversarial Network (GAN)

Hung-yi Lee - MLDS18 (Chinese)

Basic Idea of GAN

This is where the term "***adversarial***" comes from.

You can explain the process in different ways.....

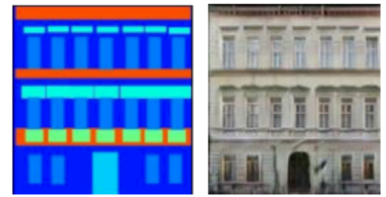


Hung-yi Lee - MLDS18 (Chinese)

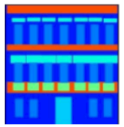
Related works

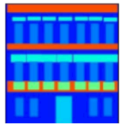
- Conditional Generative Adversarial Network (cGAN)

Image-to-image



- Experimental results

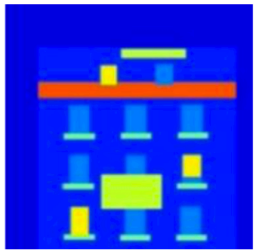




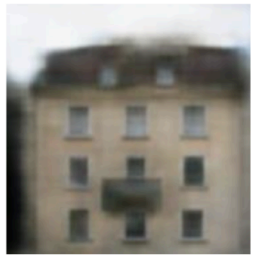
scalar

z

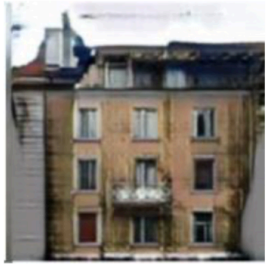
Testing:



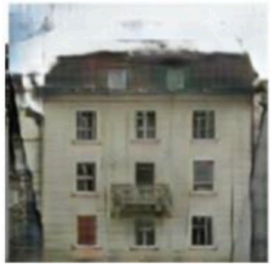
input



close



GAN



GAN + close

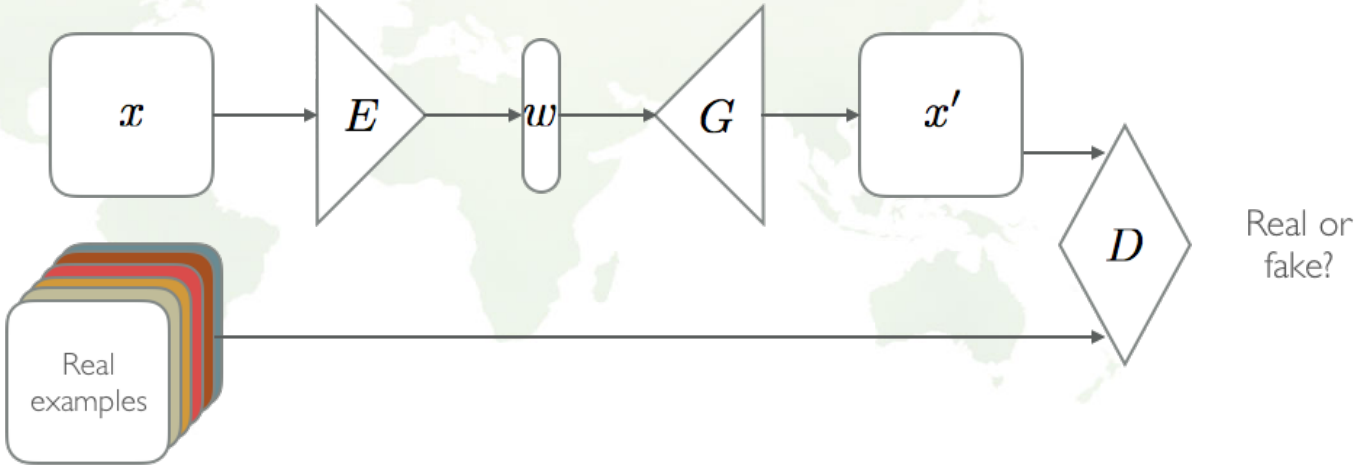
Hung-yi Lee - MLDS18 (Chinese)

Related works

- Deep Image Compression
  - Reduce the entropy

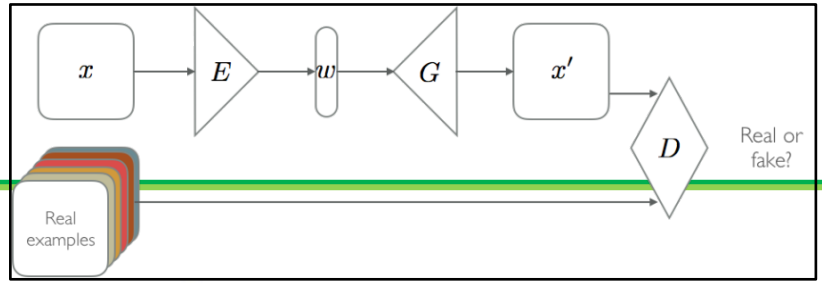
Method overview

- Auto-encoder GANs
  - E: Encoder
  - G: Generator (Decoder)
  - D: Discriminator
  - x: image
  - w: code



SqueezedIn: Extreme image compression with Deep Learning by Michael Lomnitz

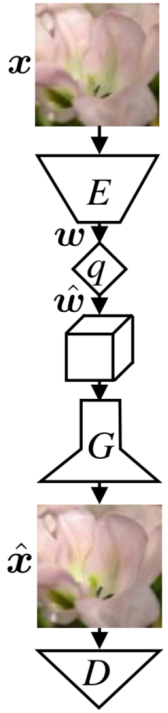
Two types of compression



- Generative compression (GC)
  - preserving the overall image content
- Selective generative compression (SC)
  - completely generating parts of the image from a semantic label map
  - preserving user-defined regions with a high degree of detail

Generative compression (GC)

- Encoder
- Quantization
- Generator
- Discriminator



Soft-to-hard vector quantization for end-to-end learning compressible representations

E Agustsson, F Mentzer, M Tschannen, L Cavigelli… - Advances in Neural …, 2017

Model detail and analysis

- Objective function
  - GAN

$$\min_G \max_D \quad \mathbb{E}[f(D(\boldsymbol{x}))] + \mathbb{E}[g(D(G(\boldsymbol{z})))], \tag{1}$$

  - cGAN

$$\mathcal{L}_{\text{cGAN}} := \max_D \quad \mathbb{E}[f(D(\boldsymbol{x}, \boldsymbol{s}))] + \mathbb{E}[g(D(G(\boldsymbol{z}, \boldsymbol{s}), \boldsymbol{s}))]. \tag{3}$$

Model detail and analysis

- Objective function
  - Image compression: entropy

$$\mathbb{E}[d(\boldsymbol{x}, \hat{\boldsymbol{x}})] + \beta H(\hat{\boldsymbol{w}}). \tag{4}$$

where d is a loss that measures how perceptually similar $\hat{\boldsymbol{x}}$ is to $\boldsymbol{x}$. Given a differentiable estimator of the entropy $H(\hat{\boldsymbol{w}})$, the weight β controls the bitrate of the model (large β pushes the bitrate down). However, since the number of dimensions $\dim(\hat{\boldsymbol{w}})$ and the number of levels L are finite, the entropy is bounded by (see, e.g., (Cover & Thomas, 2012))

$$H(\hat{\boldsymbol{w}}) \leq \dim(\hat{\boldsymbol{w}}) \log_2(L). \tag{5}$$

Model detail and analysis

- cGAN

$$\mathcal{L}_{\text{cGAN}} := \max_{D} \quad \mathbb{E}[f(D(\boldsymbol{x}, \boldsymbol{s}))] + \mathbb{E}[g(D(G(\boldsymbol{z}, \boldsymbol{s}), \boldsymbol{s}))]. \tag{3}$$

- Image compression: entropy

$$\mathbb{E}[d(\boldsymbol{x}, \hat{\boldsymbol{x}})] + \beta H(\hat{\boldsymbol{w}}). \tag{4}$$

- Objective function

$$\min_{E,G} \max_{D} \quad \mathbb{E}[f(D(\boldsymbol{x}))] + \mathbb{E}[g(D(G(\boldsymbol{z}))] + \lambda \mathbb{E}[d(\boldsymbol{x}, G(\boldsymbol{z}))] + \beta H(\hat{\boldsymbol{w}}), \tag{6}$$

Selection of beta

$$\min_{E,G} \max_{D} \quad \mathbb{E}[f(D(\boldsymbol{x}))] + \mathbb{E}[g(D(G(\boldsymbol{z}))] + \lambda \mathbb{E}[d(\boldsymbol{x}, G(\boldsymbol{z}))] + \beta H(\hat{\boldsymbol{w}}), \qquad (6)$$

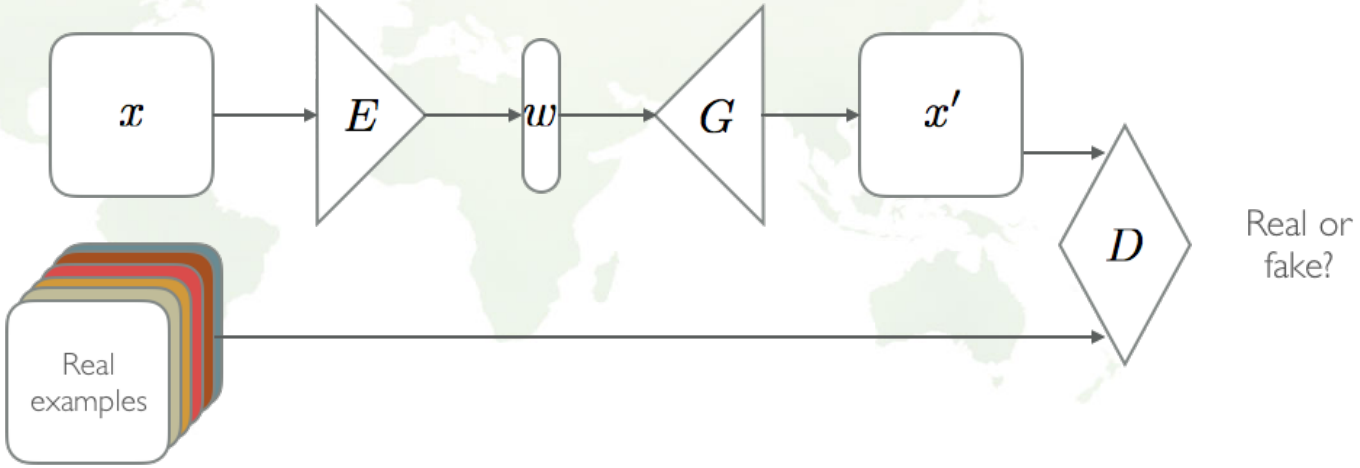
- Beta → infinity
  - pushes the bitrate down

$$, H(\hat{\boldsymbol{w}}) \rightarrow 0,$$

- Beta → 0
  - Becomes original GAN
  - suboptimal bitrates

Method overview

- Auto-encoder GANs
  - E: Encoder
  - G: Generator (Decoder)
  - D: Discriminator
  - x: image
  - w: code



SqueezedIn: Extreme image compression with Deep Learning by Michael Lomnitz

Selective generative compression (SC)

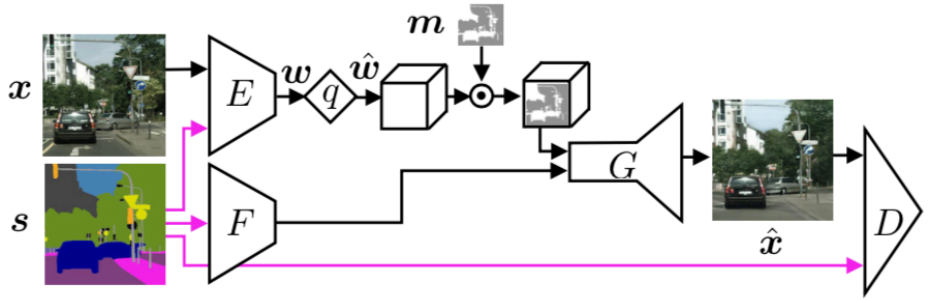
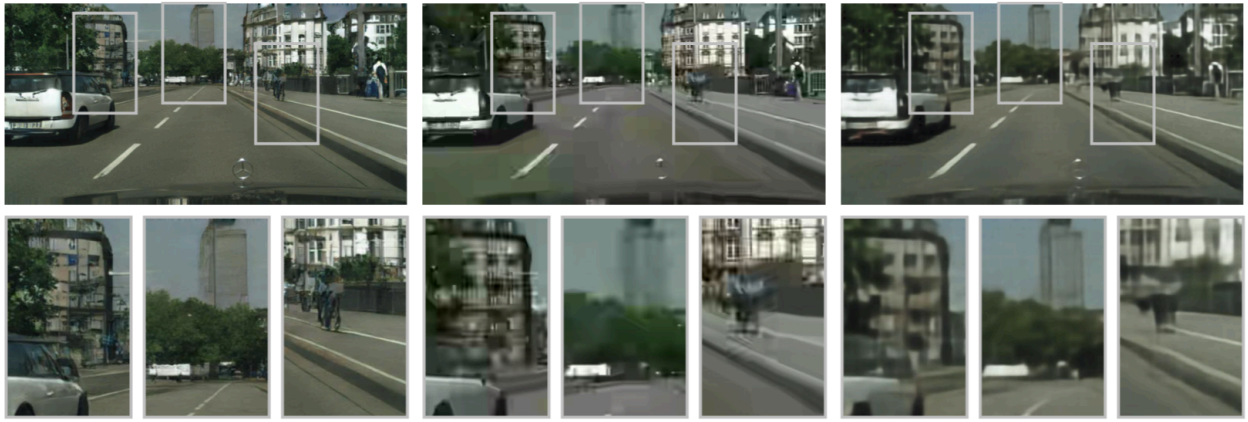


Figure 8: Structure of the proposed SC network. E is the encoder for the image $\boldsymbol{x}$ and the semantic label map $\boldsymbol{s}$. q quantizes the latent code $\boldsymbol{w}$ to $\hat{\boldsymbol{w}}$. The subsampled heatmap multiplies $\hat{\boldsymbol{w}}$ (pointwise) for spatial bit allocation. G is the generator/decoder, producing the decompressed image $\hat{\boldsymbol{x}}$, and D is the discriminator used for adversarial training. F extracts features from $\boldsymbol{s}$.

Experiments



Ours, 0.035bpp, 21.8dB BPG, 0.039bpp, 26.0dB MSE bl., 0.035bpp, 24.0dB

Figure 2: Visual example of images produced by our GC network with $C = 4$ along with the corresponding results for BPG, and a baseline model with the same architecture ($C = 4$) but trained for MSE only (MSE bl.), on Cityscapes. The reconstruction of our GC network is sharper and has more realistic texture than those of BPG and the MSE baseline, even though the latter two have higher PSNR (indicated in dB for each image) than our GC network. In particular, the MSE baseline produces blurry reconstructions even though it was trained on the Cityscapes data set, demonstrating that domain-specific training alone is not enough to obtain sharp reconstructions at low bitrates.

Compare with prior work

Original | Ours, 0.0668bpp | Rippel et al., 0.0928bpp (+39%)

Figure 14: Notice that our model produces much better sky and grass textures than Rippel & Bourdev (2017), and also preserves the texture of the light tower more faithfully.

Oren Rippel and Lubomir Bourdev. Real-time adaptive image compression. In Proceedings of the 34th International Conference on Machine Learning, volume 70 of Proceedings of Machine Learning Research, pp. 2922–2930, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

Compare with prior work

Original | Ours, 0.0651bpp | Rippel et al., 0.0840bpp (+29%)

Figure 13: Notice that compared to Rippel & Bourdev (2017), our model produces smoother lines at the jaw and a smoother hat, but proides a worse reconstruction of the eye.

Oren Rippel and Lubomir Bourdev. Real-time adaptive image compression. In Proceedings of the 34th International Conference on Machine Learning, volume 70 of Proceedings of Machine Learning Research, pp. 2922–2930, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

User study

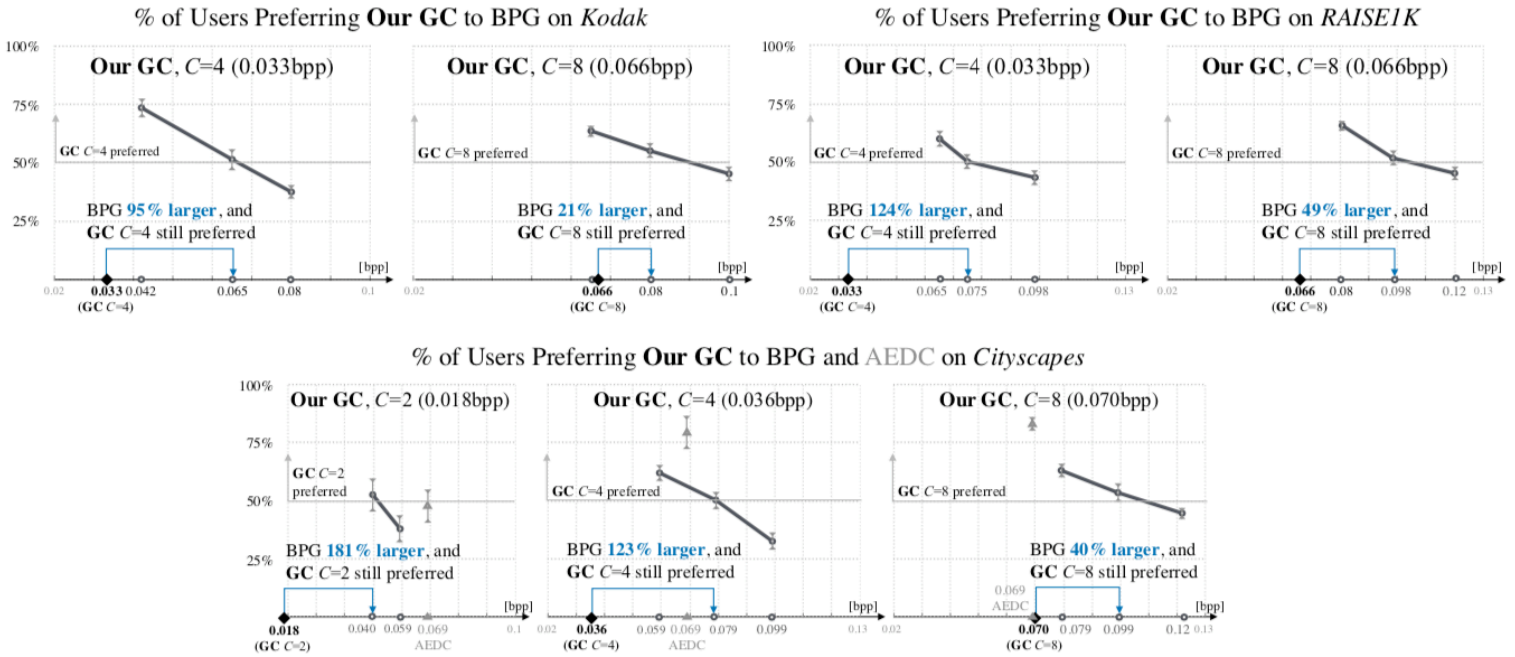


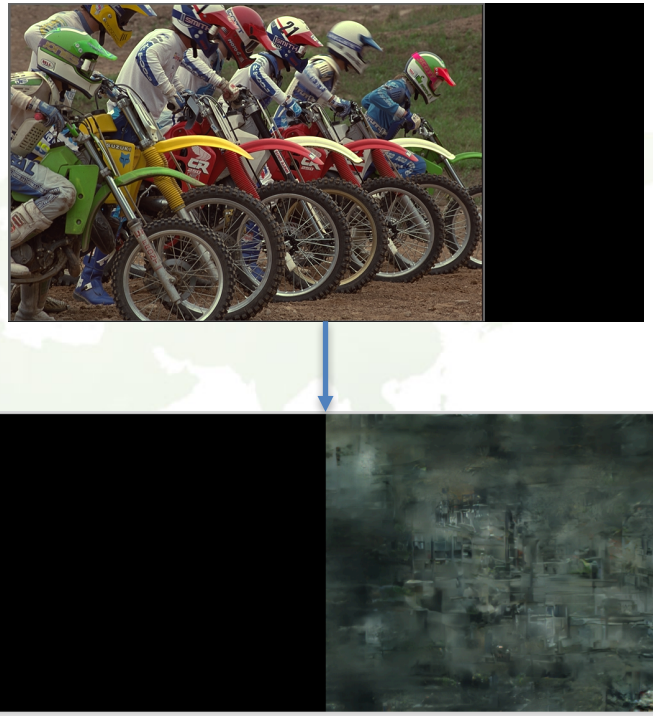
Figure 4: User study results evaluating our GC models on Kodak, RAISE1K (top) and Cityscapes (bottom). For Kodak and RAISE1K, we use GC models trained on Open Images, without any semantic label maps. For Cityscapes, we used GC (D^+), using semantic label maps only for D and only during training. The standard error is computed over per-user mean preference percentages. The blue arrows visualize how many more bits BPG uses when $> 50\%$ users still prefer our result.

Discussion

- What are the tricks?
  - The model saves information
    - Number of variables: 16M
    - Total bytes: 651 MB
    - checkpoint size: >2.5 GB

Discussion

- How to run the code?
  - Download image
  - Resize to [512, 1024]
  - Feed in to the model



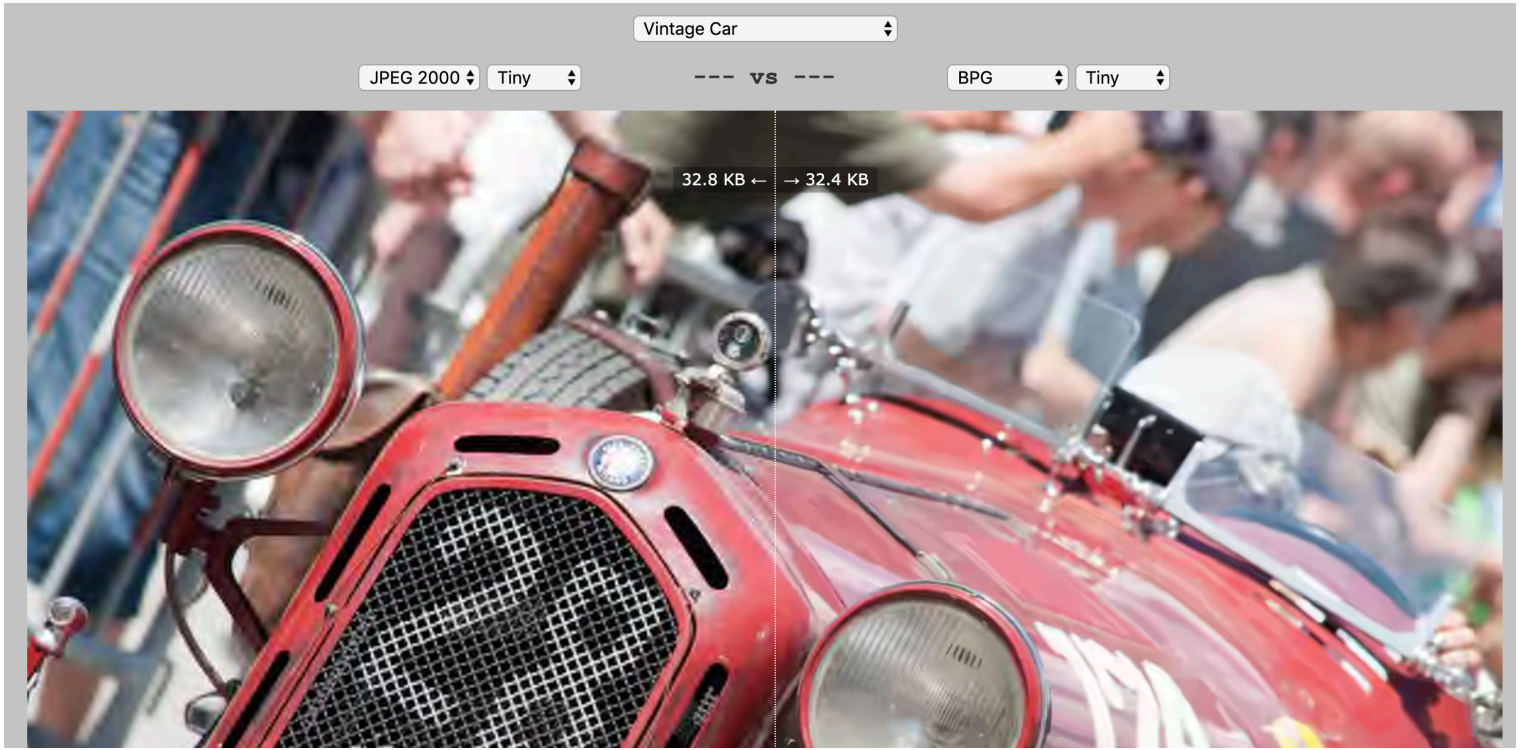
eric-yyjau/generative-compression: TensorFlow Implementation of Generative Adversarial Networks for Extreme Learned Image Compression

Discussion

- Is the comparison fair?

	Traditional method	Deep method
Method	BPG	GAN
Time complexity	Fast	5s/iteration
bpp		<0.1bbp
Overhead	Small (similar to JPEG)	Model weights: Large

Traditional method



BPG Image Comparison

Traditional method

Discussion

- Is the work practical?

Future work

- Deep learning method requires to store model weights
- Recover an image takes up to seconds
- Light-weighted models?

Questions?

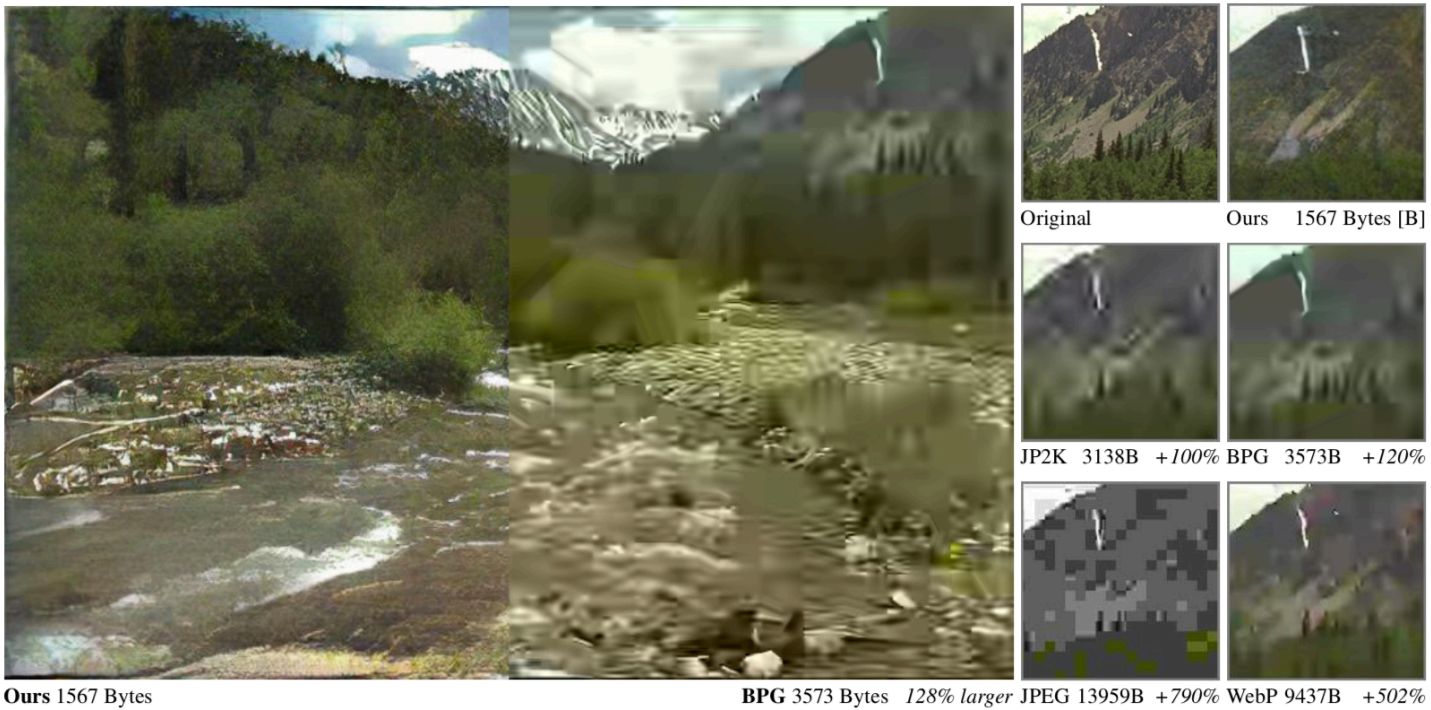


Figure 1: Visual comparison of our result to that obtained by other codecs. Note that even when using more than twice the number of bytes, all other codecs are outperformed by our method visually.

Thank you

Backup

- **Encoders SC:**
  - Semantic label map encoder: `c7s1-60,d120,d240,d480,d960`
  - Image encoder: `c7s1-60,d120,d240,d480,c3s1-`C`,q,c3s1-480,d960`

  The outputs of the semantic label map encoder and the image encoder are concatenated and fed to the generator/decoder.

- **Generator/decoder:** `c3s1-960,R960,R960,R960,R960,R960,R960,R960,R960,R960,u480,u240,u120,u60,c7s1-3`

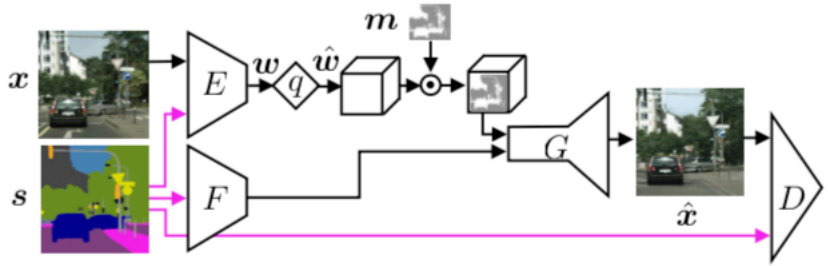


Figure 8: Structure of the proposed SC network. E is the encoder for the image $\boldsymbol{x}$ and the semantic label map $\boldsymbol{s}$. q quantizes the latent code $\boldsymbol{w}$ to $\hat{\boldsymbol{w}}$. The subsampled heatmap multiplies $\hat{\boldsymbol{w}}$ (pointwise) for spatial bit allocation. G is the generator/decoder, producing the decompressed image $\hat{\boldsymbol{x}}$, and D is the discriminator used for adversarial training. F extracts features from $\boldsymbol{s}$.

A COMPARISON WITH STATE-OF-THE-ART

	BPG	Rippel et al. (2017)	Minnen et al. (2018)	Ours (GC)
Learned	No	Yes	Yes	Yes
Arithmetic encoding	Adaptive	Adaptive	Adaptive	Static
Context model	CABAC	Autoregressive	Autoregressive	None
Visualized bitrates [bpp][5]	all[6]	0.08–	0.12–	0.033–0.066
GAN	No	Non-standard	No	f-div. based
S.o.t.a. in MS-SSIM	No	No	Yes	No
S.o.t.a. in PSNR	No	No	Yes	No
Savings to BPG in PSNR			8.41%	
Savings to BPG in User Study				17.2–48.7%

Table 1: Overview of differences between (Minnen et al., 2018) (s.o.t.a. in MS-SSIM and PSNR), to BPG (previous s.o.t.a. in PSNR) and (Rippel & Bourdev, 2017) (s.o.t.a. in MS-SSIM in 2017, also used GANs).

Generative Adversarial Networks: Given a data set $\mathcal{X}$, Generative Adversarial Networks (GANs) can learn to approximate its (unknown) distribution $p_{\boldsymbol{x}}$ through a generator $G(\boldsymbol{z})$ that tries to map samples $\boldsymbol{z}$ from a fixed prior distribution $p_{\boldsymbol{z}}$ to the distribution $p_{\boldsymbol{x}}$. The generator G is trained in parallel with a discriminator D by searching (using stochastic gradient descent (SGD)) for a saddle point of a mini-max objective

$$\min_G \max_D \quad \mathbb{E}[f(D(\boldsymbol{x}))] + \mathbb{E}[g(D(G(\boldsymbol{z})))], \tag{1}$$

where G and D are DNNs and f and g are scalar functions. The original paper (Goodfellow et al., 2014) uses the "Vanilla GAN" objective with $f(y) = \log(y)$ and $g(y) = \log(1-y)$. This corresponds to G minimizing the Jensen-Shannon (JS) Divergence between the (empirical) distribution of $\boldsymbol{x}$ and $G(\boldsymbol{z})$. The JS Divergence is a member of a more generic family of f-divergences, and Nowozin et al. (2016) show that for suitable choices of f and g, all such divergences can be minimized with (1). In particular, if one uses $f(y) = (y-1)^2$ and $g(y) = y^2$, one obtains the Least-Squares GAN (Mao et al., 2017) (which corresponds to the Pearson χ^2 divergence), which we adopt in this paper. We refer to the divergence minimized over G as

$$\mathcal{L}_{\text{GAN}} := \max_D \quad \mathbb{E}[f(D(\boldsymbol{x}))] + \mathbb{E}[g(D(G(\boldsymbol{z})))]. \tag{2}$$

Two types of compression

- Generative compression (GC)
  - preserving the overall image content while generating structure of different scales such as leaves of trees or windows in the facade of buildings, and
- selective generative compression (SC)
  - completely generating parts of the image from a semantic label map while preserving user-defined regions with a high degree of detail.