Quantized Generative Models for Solving Inverse Problems

Nareddy Kartheek Kumar Reddy, Vinayak Killedar, Chandra Sekhar Seelamantula
Department of Electrical Engineering, Indian Institute of Science, Bengaluru, 560012.
nareddyreddy@iisc.ac.in, vinayak.killedar@gmail.com, css@iisc.ac.in

Abstract

Generative priors have been shown to be highly successful in solving inverse problems. In this paper, we consider quantized generative models i.e., the generator network weights come from a learnt finite alphabet. Quantized neural networks are efficient in terms of memory and computation. They are ideally suited for deployment in a practical setting involving low-precision hardware. In this paper, we solve non-linear inverse problems using quantized generative models. We introduce a new meta-learning framework that makes use of proximal operators and jointly optimizes the quantized weights of the generative model, parameters of the sensing network, and the latent-space representation. Experimental validation is carried out using standard datasets – MNIST, CIFAR10, SVHN, and STL10. The results show that the performance of 32-bit networks can be achieved using 4-bit networks. The performance of 1-bit networks is about 0.7 to 2 dB inferior, while saving significantly (32×) on the model size.

1. Introduction

Consider \( x \in \mathbb{R}^n \) compressed to measurement \( y \in \mathbb{R}^m \), \( m \ll n \), through a linear or nonlinear measurement operator \( \mathcal{A} \). The problem is ill-posed since \( m \ll n \), and infinite solutions exist. Stated formally, \( y = \mathcal{A}(x) \), (1)

where \( \mathcal{A} \) is the sensing operator. In compressed sensing, the sensing operator is linear and involves a matrix multiplication: \( \mathcal{A}(x) = Ax \), whereas in compressed phase retrieval: \( \mathcal{A}(x) = |Ax|^2 \). In neural network based sensing, \( \mathcal{A} = \mathcal{A}_\phi \) is a network with parameters \( \phi \).

The reconstruction of \( x \) from measurement \( y \) involves two steps — enforcing measurement consistency and structural constraints. Let \( \hat{x} \) be the reconstructed signal, then first step is to ensure the measurement consistency, i.e., \( \mathcal{A}_\phi(\hat{x}) \) has to be close to \( y \) in minimum \( \ell_2 \)-norm sense. The second step imposes the constraint that \( \hat{x} \) has the desired structure, which in our case, \( \hat{x} \) must be an image. We rely on quantized generative models as an image prior to satisfy the conditions from the second step.

Generative models for solving inverse problems: Generative models have been used to solve inverse problems [2, 4, 6, 12–16, 33] and are shown to outperform the traditional optimization methods using a small number of measurements. Bora et al. enforced the constraint that the signal \( x \) lies in the range-space of the pre-trained generator network \( \mathcal{G}_\theta \) [4]. Yan et al. introduced meta-learning in the context of compressed sensing to jointly optimize the generator weights and latent space [33]. Killedar et al. enforced sparsity in the latent space to obtain superior performance over the state-of-the-art methods [14–16]. In a survey paper, Zhao et al. discusses the use of generative models for solving inverse problems [36] ranging from X-ray computed tomography to synthetic aperture radar. Song et al. used pre-trained diffusion models to solve the nonlinear inverse problems [28].

Quantization in neural networks: The downside with neural networks is the large model size, and significant computational requirement. Quantization of weights is a promising strategy to reduce the overall model footprint and to simplify arithmetic operations. Neural networks with binary parameters and/or activations (BNNs) have been shown to be promising for solving classification problems [10, 11, 26, 35, 37], continual learning [21], language modeling [9, 24], semantic segmentation [32], video processing [23], compressed image recovery [27], etc. In the context of generative modeling, Dong and Yang [7] use binary neurons at the output layer of the generator network for modeling discrete distributions. Using linear combinations of \( 2k \) binary values, Wán et al. were able to compress the generative models effectively [29]. Wang et al. developed a variant of the expectation-maximization (EM) algorithm for training the quantized GANs [31]. By utilizing multiple compression techniques such as model distillation, channel pruning, and quantization, GAN slimming method achieved \( 47 \times \) reduction in model size [30]. Andreev [1] experimented with various quantization methods specific to StyleGAN, Self-Attention GAN, and CycleGAN for efficient inference.
2. Quantized Generative models as an Image Prior for Solving Inverse Problems

In this paper, we use quantized generative models \((G_{\theta_Q})\) as an image prior. We investigate the performance-quantization trade-off when the generator weights are quantized to a low precision: 1, 2, 3, and 4 bits. We refer to the quantized generative model as Q-GEN. The sensing operator \(A_\phi\) is a neural network with parameters \(\phi\). We pursue the following optimization objective:

\[
\min_{z, \theta_Q, \phi} \|z\|_0 \quad \text{s.t.} \quad y = A_\phi(x), \quad x = G_{\theta_Q}(z). \tag{2}
\]

Solving for the optimal network parameters \(\phi, \theta_Q\) requires minimizing an appropriate loss function. We define a joint loss function that captures measurement inconsistency, isometric properties in sensing network, and image gradient loss. Consider the following optimization problem

\[
\min_{z, \theta_Q, \phi} \mathcal{L} = \mathcal{L}_G + \mathcal{L}_{A_\phi} + \mathcal{L}_{GDL}, \quad \text{s.t.} \quad \|z\|_0 \leq s, \tag{3}
\]

where

\[
\mathcal{L}_G = \mathbb{E}_{z}(\|y - A_\phi(G_{\theta_Q}(z))\|^2_2),
\]

\[
\mathcal{L}_{A_\phi} = \mathbb{E}_{x_1, x_2}(\|A_\phi(x_1) - A_\phi(x_2)\|_2 + \delta\|x_1 - x_2\|_2),
\]

\[
\mathcal{L}_{GDL} = \sum_{i, j} \|\nabla_i x - \nabla_i G_{\theta_Q}(z)\|^2_2 + \|\nabla_j x - \nabla_j G_{\theta_Q}(z)\|^2_2,
\]

with \(\delta\) and \(\gamma\) being the parameters in set-restricted eigenvalue condition (S-REC) loss \(\mathcal{L}_{A_\phi}\) [33]. The objective function is a sum of three terms: \(\mathcal{L}_G\) ensures measurement consistency; \(\mathcal{L}_{A_\phi}\), the S-REC loss, restricts the isometry in the sensing operator \(A_\phi\); and the gradient difference loss \(\mathcal{L}_{GDL}\) promotes reconstructions that are sharp [20]. We explain the importance of S-REC loss next. The signals \(x_1\) and \(x_2\) are sampled from the true data distribution and generator network \(G_{\theta_Q}\), respectively. When \(x_1\) is different from \(x_2\), we prefer that the corresponding measurements are also different. \(\mathcal{L}_{A_\phi}\) ensures that this property holds by penalizing the difference between \(\|x_1 - x_2\|_2\) and \(\|A_\phi(x_1) - A_\phi(x_2)\|_2\). This loss is inspired by the restricted isometry condition from compressed sensing.

The sparsity constraint in the latent representation \(z\) results in the range-space of the generator network \(G_{\theta_Q}\) to be composed of a union-of-manifolds [15, 16]. The optimization of \(z\) involves minimizing \(\mathcal{L}_G\) while satisfying the feasibility condition \(\|z\|_0 \leq s\). This approach is referred to as sparsity-driven latent-space sampling (SDLSS). An analysis of the distribution of the learned quantized weights from different quantized models is provided in Fig. 1. These weights are obtained after training the quantized generated models on MNIST and CIFAR-10 datasets.

Algorithm 1: Training a quantized generative model (Q-GEN) and nonlinear sensing network.

\textbf{Input:} Training data \(\{x_i, y_i\}_{i=1}^N\), sensing operator \(A_\phi\), learning rate \(\alpha\), number of iterations \(T\), and sparsity \(s\)

\textbf{Initialization:} Full-precision generator network parameters \(\theta\)

\begin{algorithmic}
  \REPEAT
  \STATE \(\theta_Q \leftarrow Q(\theta)\) as per LinearQuantize method [11]
  \FOR {\(i = 1\) \TO \(N\)}
    \STATE Initialize \(z\)
    \FOR {\(t = 1\) \TO \(T\)}
      \STATE Compute \(z \leftarrow \mathcal{P}_s(z - \beta \nabla_z f(y, z))\)
    \ENDFOR
  \ENDFOR
  \STATE Compute \(\mathcal{L}\) per Eq. (3)
  \STATE Update \(\theta\) per Eq. (6)
  \STATE Update \(\phi\) \leftarrow ADAM(\phi, \nabla_\phi \mathcal{L}_{A_\phi})$
  \UNTIL {iteration limit}
\end{algorithmic}
3. Meta-Learning for Quantized Generative Image Prior

Inspired by [33], we use the meta-learning framework to jointly optimize the parameters \( \phi \) of the full-precision sensing network, \( \theta^Q \) of the coarsely quantized generative model, and the latent space variable \( z \). Meta-learning, or learning to learn [34], is a training strategy used to improve the performance of models that involve multiple tasks [8].

The sparse code \( z \) is learned via proximal gradient-descent, whereas the sensing network parameters \( \phi \) are updated using the ADAM optimizer [17]. The quantized weights of the generator \( \theta^Q \) are updated using the straight-through-estimation method (STE) [3].

The optimization problem for obtaining the sparse code is given by
\[
\min_{z} f(y, z) = \|y - A\phi(G_{\theta^Q}(z))\|_2^2 \quad \text{s.t.} \quad \|z\|_0 \leq s.
\]  
(4)

The solution to the above problem can be obtained using the proximal gradient method, and the update equation is given by
\[
z \leftarrow \mathcal{P}_s\left(z - \beta \nabla_z f(y, z)\right),
\]  
(5)

where \( \mathcal{P}_s(\cdot) \) is the hard-thresholding operator that sets all entries in the vector to zero apart from the \( s \)-largest entries in magnitude, and \( \beta > 0 \) is the step-size parameter.

The optimization of the quantized parameters of the generator network follows a standard STE procedure [3]. The full-precision weights are also stored and used for computing the quantized weights at the beginning of every forward pass. We use the LinearQuantize method [11] to obtain the low-precision weights from the full-precision weights. In the forward pass, a generator with coarsely quantized weights is used for reconstructing the image. The optimization cost contains the measurement loss \( \mathcal{L}_G \), S-REC loss \( \mathcal{L}_{\lambda} \) and image gradient difference loss \( \mathcal{L}_{GDL} \). The gradients \( \nabla_{\theta} \mathcal{L}_{\theta = \theta^Q} \) are computed at the quantized weights and are used for updating the full-precision weights \( \theta \). The weight updates are performed using ADAM [17] optimizer:
\[
\theta \leftarrow \text{ADAM}(\theta, \nabla_{\theta} \mathcal{L}_{\theta = \theta^Q}, \alpha, \beta_1, \beta_2),
\]  
(6)

where \( \alpha \) is the learning rate and \( \beta_1, \beta_2 \) are the ADAM optimization variables (default values of 0.9, 0.999, respectively). The procedure for updating the sparse latent representation, training the quantized generative model and sensing network is given in Algorithm 1.
Table 1. Performance comparison of quantized versus full-precision models. The 4-bit models perform close to 32-bit SDLSS models across the datasets. The 32-bit DCS model outperforms the 32-bit SDLSS and its quantized counterparts in STL10 and SVHN datasets. The 32-bit SDLSS model has a superior performance over 32-bit DCS model on CIFAR10 dataset. The 4-bit SDLSS model performs on-par with the 32-bit DCS model on the CIFAR10 dataset.

<table>
<thead>
<tr>
<th>DATASET</th>
<th>PROPOSED METHODS</th>
<th>LITERATURE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-BIT SDLSS</td>
<td>2-BIT SDLSS</td>
</tr>
<tr>
<td>CIFAR10</td>
<td>(PSNR, SSIM)</td>
<td>(PSNR, SSIM)</td>
</tr>
<tr>
<td></td>
<td>22.07, 0.6787</td>
<td>22.29, 0.6961</td>
</tr>
<tr>
<td>SVHN</td>
<td>31.71, 0.9213</td>
<td>32.06, 0.9201</td>
</tr>
<tr>
<td>STL10</td>
<td>20.65, 0.5543</td>
<td>21.24, 0.5996</td>
</tr>
</tbody>
</table>

Illustrative reconstructions from MNIST and SVHN are shown in Fig 4 and Fig 5, respectively. The performance of the full precision models (32-bit DCS, 32-bit SDLSS) is superior to that of the quantized models. However, visually, the results generated by the coarsely quantized models are comparable to the images generated from full-precision models. In some cases the quantized models perform slightly superior to the full-precision models, for instance, the generated digit ‘0’ highlighted using the white boundary has a better loop closure than that output by the full-precision model.

We tested the Q-GEN model on images coming from CIFAR10, SVHN, and STL10 datasets. We used DCGAN generator and quantized its convolutional filters to 1, 2, 3, and 4 bits. The reconstruction performance is given in Table 1. The difference between full-precision and the quantized models varies from as low as 0.27 dB in the case of 4-bit models to about 2.3 dB in the 1-bit case. The performance of the quantized models in reconstructing color images inspires greater confidence for deploying them in energy-constrained settings. The quantized networks offer lower model sizes, which is attractive for resource-constrained devices, with a marginal performance drop.

5. Conclusion

We addressed the problem of compressed image recovery with a neural network used as the nonlinear sensing operator. We used a generative model with quantized weights as an image prior and incorporated sparsity into the latent space. Experimental results show that 4-bit quantized generative models perform close to full-precision (32-bit) models while reducing the model footprint by 8×. The 1-bit model has a 32× smaller footprint with a mild degradation in performance (PSNR of 0.74 dB in CIFAR10, 1.39 dB in STL10, and 2.3 dB in SVHN datasets). The analysis of the quantized weights shows that the presence of a few large weights makes the 4-bit model superior to the 1-bit counterpart. With a rapidly growing concern for saving energy in the context of AI, low-precision models offer a viable energy saving alternative. Future work would include using quantized generative image priors for other imaging applications such as ultrasound imaging and lensless imaging.
References


