Jumping through Local Minima: Quantization in the Loss Landscape of Vision Transformers

Natalia Frumkin
The University of Texas at Austin
nfrumkin@utexas.edu

Dibakar Gope
Arm Inc.
dibakar.gope@arm.com

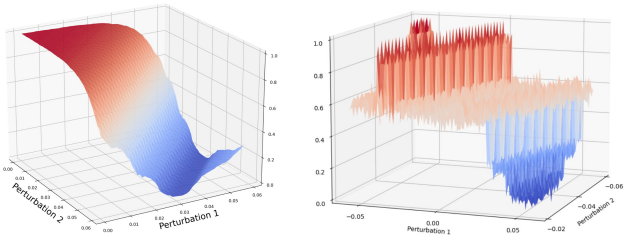
Diana Marculescu
The University of Texas at Austin
dianam@utexas.edu

Abstract

Quantization scale and bit-width are the most important parameters when considering how to quantize a neural network. Prior work focuses on optimizing quantization scales in a global manner through gradient methods (gradient descent & Hessian analysis). Yet, when applying perturbations to quantization scales, we observe a very jagged, highly non-smooth test loss landscape. In fact, small perturbations in quantization scale can greatly affect accuracy, yielding a $0.5-0.8\%$ *accuracy boost in 4-bit quantized vision transformers (ViTs). In this regime, gradient methods break down, since they cannot reliably reach local minima. In our work, dubbed Evol-Q, we use evolutionary search to effectively traverse the non-smooth landscape. Additionally, we propose using an infoNCE loss, which not only helps combat overfitting on the small calibration dataset (*$1,000$ *images) but also makes traversing such a highly non-smooth surface easier. Evol-Q improves the top-1 accuracy of a fully quantized ViT-Base by* 10.30%, 0.78%, *and* 0.15% *for 3-bit, 4-bit, and 8-bit weight quantization levels. Extensive experiments on a variety of CNN and ViT architectures further demonstrate its robustness in extreme quantization scenarios. Our code is available at* https://github.com/enyac-group/evol-q .

1. Introduction

Quantization is a widespread technique for efficient neural network inference: reducing data precision from 32 bits to ≤ 8 bits is an effective approach to aggressively reduce model footprint and speed up computation. Network quantization is an extremely important tool for deploying models in cloud [28] and edge settings [27], where we aim to maximize accuracy while reducing the computational burden. We consider the post-training quantization (PTQ) setting, where there is access to a small ($\sim 1,000$ image) calibration dataset but no access to the original training dataset. PTQ is an integral component of model deployment, when a carefully curated full-precision model is too expensive to retrain. Our work, Evol-Q, is a PTQ method for vision transformers which leverages four key observations:



(a) ResNet-18 Test Loss (b) DeiT-Tiny Test Loss

Figure 1: We perturb along two basis vectors of one layer/block's quantization scales. The test loss landscape during perturbation is smooth in the CNN case (a), and highly non-smooth in the ViT case (b).

1. **Small perturbations in quantization scale can lead to significant improvement in quantization accuracy.** For example, small adjustments in a self-attention block's scale can induce a $\pm$ 1% change in accuracy.

2. As shown in Fig. 1, **quantized vision transformers (ViTs) have an extremely non-smooth loss landscape, particularly with respect to the perturbation in quantization scales**, making stochastic gradient descent a poor choice for optimization. We use evolutionary search to favor nearby local minima to significantly improve accuracy ($\sim 0.5-1\%$).

3. In comparison to non-contrastive loss functions such as mean squared error, cosine similarity, and the KL divergence, **contrastive losses tend to**

smooth the loss landscape, as observed in our experiments and supported by recent work [10]. This finding inspires the use of contrastive loss to further facilitate the quantization scale search process. Contrastive losses, specifically the infoNCE loss in this work, also helps in combating overfitting on the small calibration dataset by incorporating negative examples into the loss.

4. **The Evol-Q framework generalizes to CNN quantization as well**, since the infoNCE loss provides a smoother landscape than other losses.

Combining these observations, we devise a new optimization scheme to adjust the quantization scales of low bit-width ViTs. Instead of using gradient descent to optimize all network parameters or estimating a noisy Hessian, we propose a series of cheap evolutionary search procedures to successively minimize the quantization error. Evol-Q injects small perturbations into the quantization scale at one layer and uses a global infoNCE loss to evaluate them. In the next section, we will show how prior work does not properly address the non-smooth ViT loss landscape.

2. Related Work

CNN Quantization: When quantizing ViTs, it is natural to borrow techniques from CNN quantization and apply them to vision transformers. In the case of general quantization, one can consider either naive methods (such as MinMax [14] quantization) or complex methods (such as gradient descent) to find the quantization scale. Naive techniques include MinMax [14], Log2 [3], or Percentile quantization, where we apply some statistical analysis on the values of a model tensor. While simple to implement, these methods can result in unacceptable accuracy degradation, particularly in the 3-bit and 4-bit case. This leads us to employ more advanced strategies to recoup some accuracy lost by quantization.

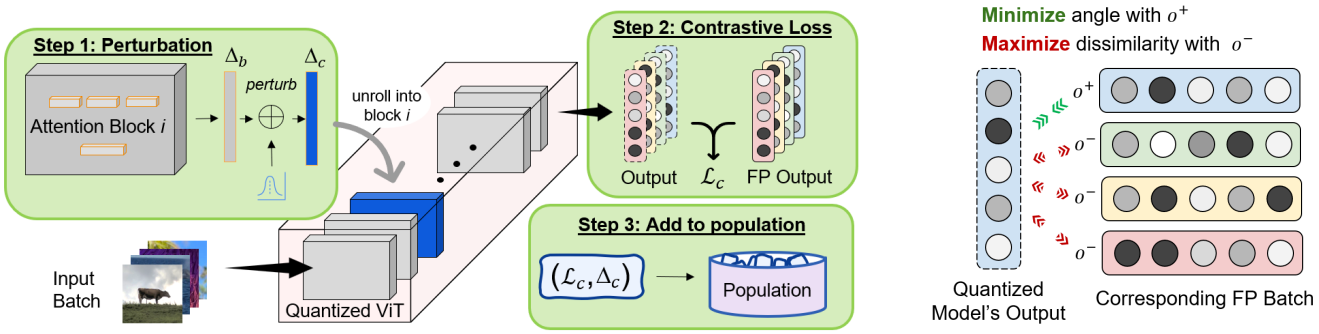
Complex methods involve techniques such as gradient descent[24], Hessian analysis [15], or knowledge distillation [6] to maximize the quantized model's accuracy. In particular, many methods employ a layer-wise loss [29, 13, 2]. A layer-wise loss can serve as a good proxy for a smooth global loss [23], yet a local loss is unlikely to resemble the ViT's highly non-smooth global landscape [1]. While Hessian-based methods [15] can utilize second order information, the ViT loss landscape resembles an "egg carton" with a high density of extremal points. This space cannot be traversed well with only first and second-order gradient information. Moreover, BRECQ [15] assumes the Hessian is positive semi-definite (PSD) which is a poor assumption for any non-smooth landscape.

ViT Quantization: Some work has already achieved very good accuracy on vision transformers. PTQ-for-ViT [22] learns a quantization scale for each layer by using two loss functions: (1) a ranking loss for each multi-head attention module; and (2) cosine similarity for the MLP layer. The layer-wise loss may achieve good accuracy, but with the non-smooth ViT landscape, this approach may end up at a local maximum and may be very sensitive to initialization. PTQ4ViT [31] also employs a Hessian-guided metric for guided global optimization similar to BRECQ [15]. As we mentioned before, the PSD assumption on the Hessian breaks down for our "egg carton"-shaped loss landscape. In contrast, PSAQ-ViT-V2 [16] uses a student-teacher MinMax game to minimize the KL divergence between the full precision and quantized models. The KL divergence is applied to kernel density estimations which are much more robust than Hessian-based or gradient-based techniques. We believe this technique can traverse the non-smooth ViT landscape, and despite being data-free, it has the best accuracy of all other methods we compare against.

We apply Evol-Q, our quantization scale learning method, on top of the FQ-ViT [20] which incorporates Log2 quantization and an integer Softmax function for end-to-end 8-bit quantization. FQ-ViT [20] is surprisingly effective on ViTs, likely because Log2 quantization can compensate for the asymmetric distributions following the Softmax and GeLU layers.

Quantization-Aware Training (QAT) has shown impressive results on vision transformers [30, 19, 18, 17], yet these methods consider training with the entire dataset rather than an unlabeled calibration dataset.

Contrastive Loss: In this work, we use a global infoNCE loss as our preferred loss function for both CNN and ViT quantization. Since we have such a small calibration dataset, the infoNCE loss helps generalize to the unseen test set. We are the first work to use a infoNCE loss in this manner. Prior work has combined quantization and self-supervised learning in a joint training framework [4, 11] allowing for regularization from both the contrastive loss and a quantization loss in training. In particular, SSQL [4] uses a joint SimSiam-L_2 loss during training to improve quantization for all bit-widths, whereas our method considers how the infoNCE loss, in conjunction with an evolutionary search, is used to traverse the highly non-smooth loss landscape and optimize the quantization scale for *a specific bit-width at test time*. Moreover, SSQL [4] uses the loss as regularization and not as a



(a) A single cycle of block-wise evolutionary search.

(b) Visualization of the infoNCE Loss

Figure 2: An overview of Evol-Q. On the left, we show one cycle completed on a single block. Each block has C cycles of evolutionary search, and we perform P passes over all blocks. On the right, we provide intuition for the infoNCE loss (Step 2), where we encourage similarity between the quantized and corresponding predictions while simultaneously maximizing dissimilarity between unlike predictions.

proxy for the test loss . Another work [26] applies contrastive learning to binarized ResNet and VGG models. They apply a layer-wise infoNCE loss, showing that it achieves good results for the 2-bit loss landscape of small CNNs. While a layer-wise loss is sub-optimal, the ability for the infoNCE loss to perform well on a binary loss landscape is great motivation for our work.

In summary, prior work has migrated CNN quantization techniques to ViT quantization without addressing the non-smooth ViT loss landscape. In the coming section, we present Evol-Q as an effective solution to this problem.

3. The Evol-Q Framework

In the non-smooth ViT quantization landscape, first and second-order gradient methods are not effective and cannot handle the large number of local minima. In such a regime, small perturbations in quantization scale can lead to a significant boost in accuracy. We apply evolutionary search using an infoNCE loss to evaluate these perturbations, enabling Evol-Q to traverse a non-smooth surface and minimize overfitting on the calibration dataset.

We begin with an overview of uniform quantization and then dive into the core components of our method: traversing perturbations using evolutionary search and evaluating them using an infoNCE loss.

3.1. Uniform, End-to-End Quantization

We consider uniform quantization, where full precision values are mapped to a uniform range based on the quantization bit-width (b). Uniform quantization is formally defined as:

$$Q(\mathbf{x}, \delta, b) = \texttt{clip}(\left\lfloor \frac{\mathbf{x}}{\delta} \right\rceil, -2^{b-1}+1, 2^{b-1}-1) \quad (1)$$

where $\mathbf{x}$ is a full-precision vector and δ is the quantization scale. We can generalize this idea to a tensor $\mathbf{X}$ and a corresponding quantization vector Δ (*e.g,* each element in Δ corresponds to a channel in channel-wise quantization). In our framework, we learn these initial quantization parameters using a fast, layer-wise framework such as FQ-ViT [20], and then use evolutionary search to adjust the scales of each attention block's projection layers.

3.2. Where to Perturb?

A ViT model [8] consists of 12 multi-head self-attention (MHSA) blocks stacked on top of each other. Each block applies the following transformation on a given query (Q), key (K), and value (V):

$$MHSA(Q, K, V) = \texttt{concat}(H_0, H_1, \dots H_N)W^O \quad (2)$$

where each attention head H_i is:

$$H_i(Q, K, V) = \texttt{softmax}(\frac{(W^Q Q)(W^K K)^T}{\sqrt{d_k}}) \cdot W^V V \quad (3)$$

Our method applies end-to-end quantization meaning that for each attention block we quantize all $3N+1$ weight tensors and $6N+1$ intermediary activations where N is number of heads. The quantization scales of all weights and activations can be concatenated and viewed as the vector Δ. This stacked vector is very important for understanding our algorithm – we can perturb the scales for all weights and activations simultaneously by perturbing Δ. We perturb by sampling

from a uniform ball centered around Δ:

$$\mathbf{\Delta}_{\texttt{new}} \sim \mathcal{U}(\mathbf{\Delta}, -\epsilon, \epsilon) \tag{4}$$

where ϵ controls the size of the uniform ball. Perturbations within a small ϵ-ball ($\epsilon \approx 10^{-4}$) yields a change in accuracy of $\pm 1\%$ top-1 accuracy for 4-bit quantization. This is the same order of magnitude used to compare a variety of quantization methods, further illustrating that small perturbations matter considerably for quantization performance.

We will show in sequel how evolutionary search can quickly evaluate multiple perturbations and choose which local minima to hop into.

3.3. Global Search for Quantization Scales

As previously mentioned, we perturb the quantization scales (Δ) for one attention block at a time. If we perturb too many scales at once, we end up traversing a very high dimensional search space and the number of iterations for evolutionary search convergence increases exponentially. Of course, gradient descent is often effective for such large search spaces, but first-order methods will not work well in our non-smooth loss regime. After partitioning our search space in a block-wise manner, we find evolutionary search to perform well and achieve acceptable convergence times (on par with the runtime of other PTQ methods).

We apply a small evolutionary search algorithm for each attention block (shown in Fig. 2), and repeat this for all blocks in the model. A block's evolutionary search algorithm has C cycles, where each cycle spawns a perturbation. The search population is initially set to $|K|$ copies of the original scale, and each cycle progressively updates the population of scales. During one cycle, we choose a parent scale from the population and a perturbation (child scale) is then spawned from the uniform distribution parameterized by the parent scale (see Eq. (4)). The child scales are then placed into the population for the next cycle. We evaluate each child scale using the fitness function defined in Algorithm 2 which applies a global infoNCE loss. The block-wise search algorithm is shown in Algorithm 1. We refer the reader to evolutionary computing literature [9] for more details on parent, child, and fitness function.

In summary, our block-wise evolutionary search consists of P passes over all attention blocks. For each attention block b, we apply a small evolutionary algorithm for C cycles, meaning that each block's quantization scale is adjusted $P \times C$ times. A full enumeration of the search settings are in Tab. 1.

Algorithm 1 Block-wise Evolutionary Search

Input: Calibration Dataset D_C
Quantized Model M_Q, Full Precision Model M_F
Number of passes P, cycles C
Population size K, Sample size S

for passes in $0:P$ **do**
  ▷ traverse all attention blocks
  for each attention block b **do**
    ▷ sub-problem illustrated in Fig. 2
    pop $\leftarrow$ [ ]
    ▷ init population with K perturbations
    while | pop | $< K$ **do**
      ▷ `Fitness` defined in Algorithm 2
      fitness $\leftarrow$ `Fitness`(b, M_Q, M_F, D_C)
      pop.`insert`(($\mathbf{\Delta}_{\texttt{b}}$, fitness))
    ▷ Iterate to find best child Δ
    for cycles in $0:C$ **do**
      ▷ From population, sample a parent Δ
      samples $\leftarrow$ [ ]
      while | samples | $< S$ **do**
        samples $\leftarrow$ `RandomElement`(pop)
      parent $\leftarrow$ `BestFitness`(samples)

      ▷ Generate child and add to population
      $\mathbf{\Delta}_{\texttt{child}} \leftarrow$ `Perturb`($\mathbf{\Delta}_{\texttt{parent}}$)
      fitness $\leftarrow$ `Fitness`(D_C, M_Q, M_F)
      pop.`insert`(($\mathbf{\Delta}_{\texttt{child}}$, fitness))
      ▷ remove candidate with lowest fitness
      pop.`DeleteDead`()
    ▷ Choose best perturbation as b's final Δ
    $\mathbf{\Delta}_{\texttt{b}} \leftarrow$ `BestFitness`(pop)
return M_Q ▷ model with updated scales

3.4. The infoNCE Loss for Scale Search

If we apply evolutionary search on the test loss landscape, we can quickly traverse the space of local minima and find the best quantization scale. Unfortunately, the test loss is not known to us, and the small calibration dataset may produce a loss landscape that is different from the true loss landscape. We find that the infoNCE loss can incorporate negative samples to reduce the model's tendency to overfit on positive bias.

The infoNCE loss is a common contrastive loss function used in self-supervised learning to smooth the loss landscape [10], prevent representation collapse [12] and encourage discrimination between the target representation and a set of negative examples. We find the infoNCE loss to be very effective to prevent overfitting of a *quantized network's* representation in the same way that it is used to develop richer representations in a

self-supervised setting [10] (see supplementary materials for details). Inspired by Chen et al. [5], we use the infoNCE [25] loss:

$$\mathcal{L}_c = -\log \frac{\exp(p \cdot o^+/\tau)}{\exp(p \cdot o^+/\tau) + \sum_{o^-} \exp(p \cdot o^-/\tau)} \quad (5)$$

where p is the prediction of the **quantized model**. The infoNCE loss is **sampled within the full-precision model's batch**, where o^+ is the corresponding prediction to p, and o^- is a prediction of other images in the same batch. In Algorithm 2, we show how the infoNCE is evaluated in the fitness function for Evol-Q.

Algorithm 2 Fitness Function

Input: calibration dataset D_C
quantized model M_Q, full precision model M_F

for batch in D_C **do**
  p = M_Q(batch)
  o = M_F(batch)
  score += `infoNCE`(p, o) ▷ Eq. (5)
return score / `size`(D_C)

4. Results

In the following section, we present results on a variety of vision transformers and show the consistency of our method under standard 8-bit quantization and in extreme quantization schemes (3-bit and 4-bit weights). We present results for end-to-end quantization, where all weights and activations are quantized.

4.1. Setup

In our post-training (PTQ) setup, the calibration dataset is $1,000$ randomly sampled images from the ImageNet training set. Experiments are conducted on ImageNet (ILSVRC2012), and we evaluate top-1 accuracy for a variety of ViT model families. For Evol-Q, the initial quantized model (M_Q in Algorithm 1) is generated using FQ-ViT, and our method perturbs its quantization scales to yield better performance. FQ-ViT is an end-to-end quantization framework that uses MinMax [14] for weight quantization and Log2 [3] for activation quantization. We refer to FQ-ViT and our code for other quantization settings. The Evol-Q search parameters (from Algorithm 1) are in Tab. 1.

4.2. 8-bit Quantization

We compare our standard 8-bit quantization with state-of-the-art methods in Tab. 2. Evol-Q improves over existing *end-to-end* quantization techniques by 0.1%, 1.2%, and 0.15% for DeiT-Small, DeiT-Base, and ViT-Base, respectively.

passes	P	10
population size	K	15
cycles	C	3
samples	S	10
mutation range	ϵ	$10^{-3}/10^{-4}$

Table 1: Block-wise evolutionary search settings. The mutation range is 10^{-3} for 8W8A, and 10^{-4} for 4W8A and 3W8A.

8-bit weights, 8-bit activations (8W8A)				
Method	DeiT-T	DeiT-S	DeiT-B	ViT-B
PSAQ-ViT	71.56	76.92	79.10	37.36
PTQ4ViT	-	79.47	81.48	84.25
FQ-ViT	71.61	79.17	81.20	83.31
PSAQ-ViT-V2†	**72.17**	79.56	81.52	-
Evol-Q (ours)	71.63	**79.57**	**82.67**	**84.40**

† Does not quantize Softmax/GeLU layers

Table 2: Top-1 Accuracy on ImageNet using 8W8A quantization. -T, -S, & -B refer to Tiny, Small, and Base models respectively.

We also compare with PSAQ-ViT-V2 [16] and find that it outperforms our method by 0.5% for DeiT-Tiny. However, PSAQ-ViT-V2 is not a full end-to-end quantization method since it *does not quantize the activations* following the Softmax and GeLU layers. These activations are typically very sensitive to quantization and are often maintained at full precision. We applied Evol-Q on an end-to-end quantized model, so we are forced to quantize the post-Softmax/GeLU activations. We leave it to future work to apply our technique on top of PSAQ-ViT-V2, but expect similar improvements to what we achieved with FQ-ViT.

4.3. 4-bit Quantization

Moving from 8-bit to 4-bit weight quantization, we see an accuracy degradation of about $2-5\%$ across all models. In Tab. 3, Evol-Q performs similarly to what is shown for 8-bit quantization. In particular, we still see improvement for DeiT-Small, DeiT-Base, and ViT-Base, but now the top-1 accuracy improvement is 0.13%, 0.16%, and 0.77%, respectively.

4.4. 3-bit Quantization

We report 3-bit quantization results in Tab. 4 to show that Evol-Q extends to more extreme quantization scenarios. In particular, Evol-Q improves accuracy over FQ-ViT by 10.3% for ViT-Base and 3.7% for DeiT-Tiny.

4-bit weights, 8-bit activations (4W8A)				
Method	DeiT-T	DeiT-S	DeiT-B	ViT-B
PSAQ-ViT	65.57	73.23	77.05	25.34
PTQ4ViT	-	-	64.39	-
FQ-ViT	66.91	76.93	79.99	78.73
PSAQ-ViT-V2†	**68.61**	76.36	79.49	-
Evol-Q (ours)	67.29	**77.06**	**80.15**	**79.50**

† Does not quantize Softmax/GELU layers

Table 3: Top-1 Accuracy on ImageNet using 4W8A quantization.

3-bit weights, 8-bit activations (3W8A)				
Method	DeiT-T	DeiT-S	DeiT-B	ViT-B
FQ-ViT	35.79	60.58	72.11	55.33
Evol-Q (ours)	**39.45**	**61.16**	**72.41**	**65.63**

Table 4: Top-1 Accuracy on ImageNet with 3W8A quantization.

By reducing the precision from 32 to 3 bits, we achieve a >10X reduction in memory footprint while still maintaining a reasonable accuracy for DeiT-Base. We refer to supplementary materials for ablations on using OMSE [7] and bias correction [2] for 3W8A which dramatically improves our method's performance.

4.5. Extending to Swin & LeViT Models

We run our experiments on additional model families to ensure that our method is applicable to different types of attention blocks. Swin transformers [21] have the same macro-architecture as DeiT and ViT models, with the exception that the Swin transformer block is a windowed attention. We see results for 4-bit Swin transformers in Tab. 5.

Method	Swin-T	Swin-S	Swin-B
PSAQ-ViT	71.79	75.14	-
PSAQ-ViT-V2†	76.28	78.86	-
FQ-ViT	**80.73**	82.13	82.73
Evol-Q (ours)	80.43	**82.63**	**83.07**

† Does not quantize Softmax/GELU layers

Table 5: Top-1 Accuracy for 4W8A Swin Models.

Prior quantization techniques do not consider LeViT models, a family of ViTs that improve the inference speed of existing transformers. Using a longer convolutional backbone, they can achieve similar results to classic ViTs while also reducing the complexity of the transformer blocks in favor of ResNet-like stages. We include the LeViT family in our experiments to illustrate how our method can be extended beyond the standard block size. We can see 4W8A results for the LeViT model family in Tab. 6. Across the board, we see a significant improvement in LeViT quantization as compared to FQ-ViT, our baseline.

Method	LeViT -128S	LeViT -192	LeVit -256	LeViT -384
FQ-ViT	14.90	17.00	61.33	64.60
Evol-Q (ours)	**29.20**	**30.37**	**64.57**	**69.50**

Table 6: Top-1 Accuracy for 4W8A LeViT models.

In fairness, we have not applied any other techniques to boost LeViT's accuracy (doing so may inflate our method's improvement), so we leave it to future work to incorporate other quantization techniques on top of our framework.

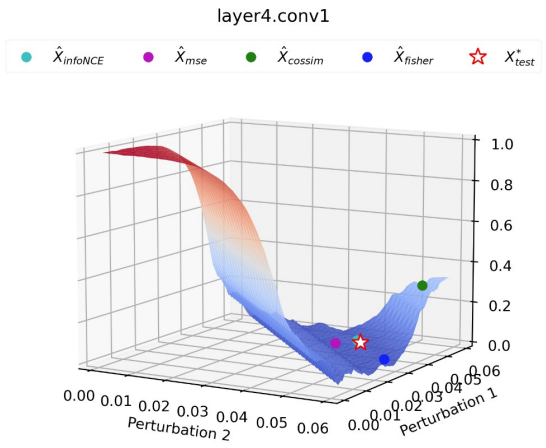
5. Analysis

In the following section, we support the three observations set forth in the introduction. First, we show how **the non-smooth ViT loss landscape compares with the CNN one**, and discuss how a variety of loss functions perform in the Evol-Q framework. Next, we visualize the layer-wise weight distributions to illustrate how small perturbations can yield a significant jump in accuracy. Finally, we report runtime and various ablations to contextualize our method in the broader space of quantization techniques. For more ablations and analysis, please refer to the supplementary materials.

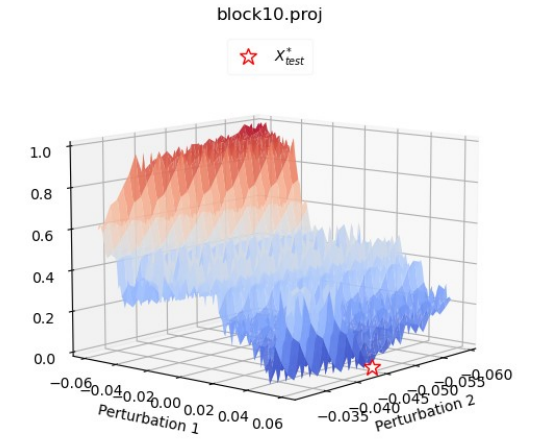
5.1. The Test Loss Landscape of ViTs

In Fig. 3, we show that perturbing quantization scale yields a smooth test loss landscape for ResNet-18 and a jagged, non-smooth landscape for DeiT-Tiny. This loss landscape illustrates how the test loss is related to the Δ_{10} scale at block #10. **We perturb along two basis vectors for the Δ_{10} and observe the effect on the test loss.** The DeiT-Tiny loss landscape is very complex and highly non-linear, whereas the ResNet-18 landscape is comparatively smooth. Intuitively, we hypothesize that the presence of many GeLUs and Softmax functions induces in the DeiT loss landscape many more extreme points than in the ResNet loss landscape.

For the smooth landscape in Fig. 3a, the infoNCE loss is clearly optimal since it is closest to the global minimum. If we plot the MSE, Cosine (Cossim), and Fisher loss landscapes, we find that they are not as smooth as the infoNCE loss in the CNN case (see supplementary materials, Sec A). The infoNCE loss helps to provide a smoother loss with respect to the calibration dataset by incorporating negative samples to reduce bias [10].



(a) Resnet-18 loss landscape



(b) DeiT-Tiny loss landscape

Figure 3: A comparison of the test loss landscapes for 4-bit quantized CNNs and ViTs. In Fig. 3a, we show how small perturbations in the 4^{th} convolutional layer yields a smooth test loss landscape. In Fig. 3b, we apply perturbations to attention block #10 and the resultant loss landscape is highly non-smooth.

In the non-smooth landscape, Fig. 3b, the global minimum is very hard to find. In fact, proximity to the global loss in this landscape is not a good indicator of loss function quality, since there are many local maxima in close proximity (with an ϵ- ball) of the global minimum. In Sec. 5.3, we provide an empirical justification that the infoNCE loss prevents overfitting and is superior to other loss functions.

5.2. Gradient Descent vs. Evolutionary Search

In Fig. 4, we show how evolutionary search finds candidates close the local minima, whereas gradient descent breaks down the ViT loss landscape. We show three initial points, where gradient descent either oscillates (init 3) or does not converge to a local minima (init 1 and 2). In contrast, evolutionary search generates a candidate perturbation and steps in the direction of the best candidate. We find that evolutionary search is very good at finding the closest local minima, which is sufficient to get an accuracy boost of $\sim 0.4\%$ in this loss landscape.

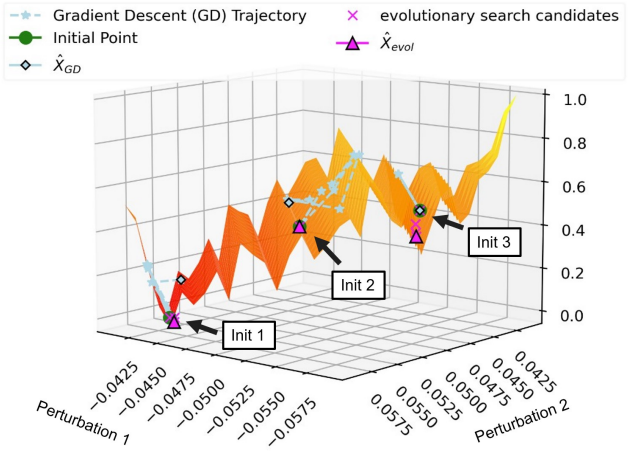


Figure 4: A zoomed in section of the landscape in Fig. 3b, where we perform gradient descent and evolutionary search for three initial points. We show the solutions of evolutionary search ($\hat{X}_{evol}$) and gradient descent ($\hat{X}_{GD}$) after 10 iterations.

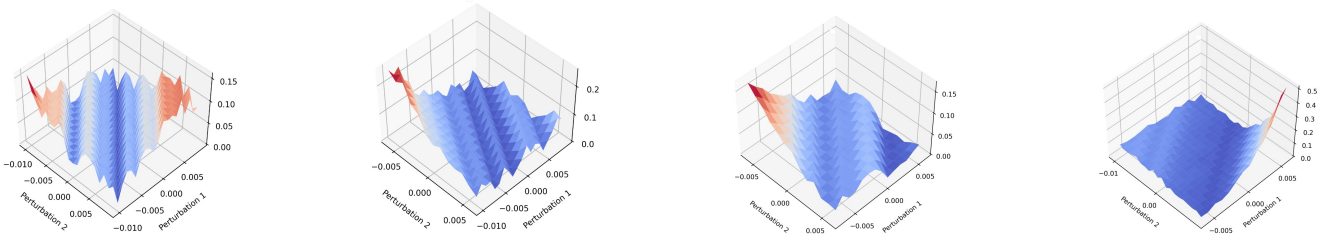
In Table 7, we show quantitatively that gradient-based optimizers underperform in comparison to evolutionary search for the same block-wise setting as in Evol-Q. We believe that non-smoothness at the block level is what makes these gradient-based techniques ineffective.

Method	DeiT-T	DeiT-S	DeiT-B	ViT-B
SGD	71.57	79.25	81.24	83.40
Adam	71.29	79.25	81.24	83.25
AdamW	71.37	79.00	81.30	83.36
Evol-Q (ours)	**71.63**	**79.57**	**82.67**	**84.40**

Table 7: Comparison with gradient-based optimizers using 8-bit weights, 8-bit activations (8W8A).

5.3. Loss Function Choice

We compare the infoNCE (contrastive) loss with other common loss functions in Fig. 6. We find mean-squared error (MSE) to be equally (if not more) effective in the initial iterations of Evol-Q. However, as the number of passes grows, MSE does not perform as well as the infoNCE loss. Both cosine similarity and the Kullback–Leibler divergence (KL) fail to improve performance as the number of iterations increases. We postulate that the poor performance of these traditional loss functions is due to overfitting to the cali-



(a) Query, Key, Value (b) Projection Layer (c) FC Layer #1 (d) FC Layer #2

Figure 5: Loss Landscapes for the 4-bit quantized QKV, Projection, and Fully Connected (FC) layers in self-attention block #5. We perturb the the quantization scale along two basis vectors (Perturbation 1 & 2) to visualize the loss landscape. These landscapes capture a zoomed in region around the global minimum of the full landscape. The FC layers exhibit relative smoothness around the global minimum whereas the QKV & Projection layers are not easily traversible. The Projection layer is particularly difficult for gradient methods because it has 4 deep minima in close proximity to the global minimum.

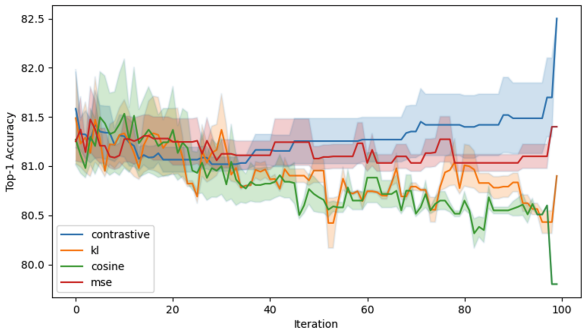


Figure 6: Comparing four loss functions in the Evol-Q framework on ViT-Base. The infoNCE (contrastive) loss prevents overfitting to the calibration dataset, whereas all other loss functions cannot improve accuracy beyond the initialized quantization scheme.

bration dataset. On the other hand, the infoNCE loss is naturally regularized by the negative samples in the batch, allowing for it to preserve the quantization parameters that help discriminate between classes.

5.4. Which layers contribute to non-smoothness?

In Fig. 5, we visualize which layers of the self-attention mechanism yield the non-smooth loss curve. We show that learning the quantization scale for query, key, value (QKV) and projection layers is more difficult because their loss landscape is filled with local minima. We observe this property across different self-attention blocks and advocate for using ES to jump through the field of local minima.

5.5. Layer-wise Weight Distributions

In Fig. 7, we compare the weight distributions of the full precision, FQ-ViT, and Evol-Q quantization schemes. Evol-Q's quantized values are only a small adjustment of FQ-ViT's, yet Evol-Q has a 0.8% improvement. In summary, a small adjustment in scale yields a significant boost in accuracy.

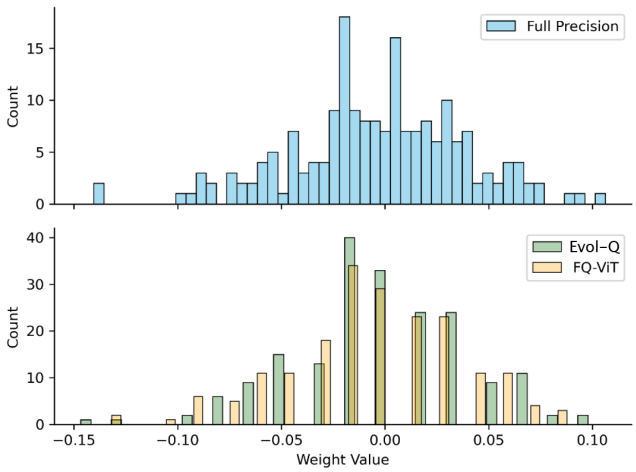


Figure 7: The weight distribution of the projection layer for attention block #1 of ViT-Base. The top plot shows the full precision weight distribution and the bottom plot shows the 8-bit weights using both FQ-ViT and Evol-Q.

This observation is consistent with results in AdaRound [23], where authors show how choosing the correct rounding scheme can significantly impact performance. Unlike AdaRound [23], our method traverses the global loss landscape (rather than a layer-wise proxy) whereas AdaRound assumes a diagonal Hessian which does not hold in the ViT landscape.

We refer to Sec. F of the supplementary materials for more discussion on layer-wise distributions.

	ResNet-18	ResNet-50	RegNet-3.2GF
BRECQ [16]	69.60	75.05	74.21
Evol-Q†	**70.10**	**77.30**	**76.87**

† Using BRECQ[16] as pre-quantized model M_Q

Table 8: Top-1 Accuracy on ImageNet using 4W4A quantization.

5.6. Generalization to CNNs

Our method begins with a pre-quantized model, M_Q, and adjusts the quantization scales to improve accuracy. We only require that the model can be abstracted into blocks, which makes our method readily applicable to other types of models such as CNNs, LSTMs, and Graph Neural Networks. In Tab. 8, we show how Evol-Q is run on top of BRECQ to achieve state-of-the-art CNN quantization. In this case, our block is one convolutional layer and Δ is the stacked vector of quantization scales for the one layer's weight matrix. We find Evol-Q's method to be suitable for CNN quantization, achieving 1-2.5% accuracy boost over BRECQ for 4-bit quantization. We refer to supplementary material for an explanation of using the infoNCE loss to smooth out the CNN loss landscape.

5.7. Pareto Front for 8-bit Quantized ViTs

Evol-Q improves over existing PTQ techniques for vision transformers. In Fig. 8, Evol-Q is on the Pareto front in terms of both top-1 accuracy and runtime for 8-bit ViT-Base. Current ViT-specific QAT methods [19, 30] do not report 8-bit accuracy, so we do not include them here. These QAT methods are likely to reach the Pareto front, but would take much longer than existing PTQ methods.

All open-source methods are run on a single Nvidia A100 GPU, but some code is not open-sourced at the time of submission. PSAQ-ViT-V2 does not report runtime, so we estimate it to be 60 minutes based on PSAQ-ViT and the relative cost of additional steps.

5.8. Runtime

We run our method on an Nvidia A100-PCIE-40GB Tensor Core GPU and find that all experiments take less than one hour to run. The average runtime is shown in Tab. 9. We use PyTorch 1.9.1, built with the CUDA 11.1 toolkit.

	DeiT-T	DeiT-S	DeiT-B	ViT-B
Runtime (m)	41.5	46.3	41.6	43.2

Table 9: Evol-Q Runtime (in minutes) on one Nvidia A100 GPU

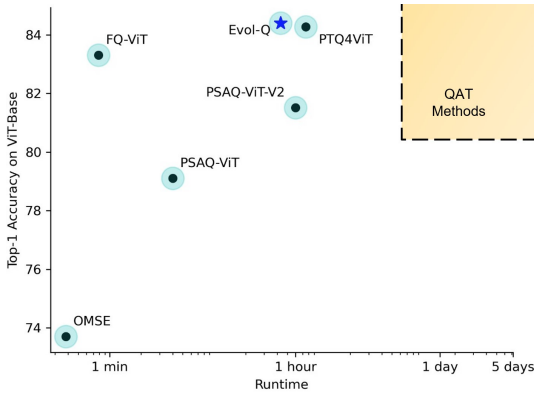


Figure 8: The runtime of 8-bit quantization methods for ViTs. We consider PTQ methods, and show the potential for QAT methods (orange square).

In Fig. 8, we compare runtime with other ViT quantization methods and demonstrate that our method achieves superior accuracy on ViT-Base. Fig. 8 only captures the Top-1 Accuracy of ViT-Base (or, alternatively, DeiT-base if ViT-Base is unavailable). We refer to the supplementary material for a wider discussion on how this plot changes with different models.

6. Conclusion

Evol-Q achieves state-of-the-art results on ViT's highly non-smooth loss landscape with a high density of extremal points. Prior work on ViT quantization does not address the non-smooth loss landscape, nor how small perturbations in quantization scale can affect performance. Using evolutionary search and an infoNCE loss, Evol-Q evaluates small perturbations in quantization scale, improving accuracy by $\sim 0.5\%$ for 4-bit quantized vision transformers.

7. Acknowledgements

Partial work completed during a summer internship at Arm Ltd. A special thank you to Jesse Beu for overseeing this internship project, and to Feng Liang for helpful discussion.

References

[1] Haoli Bai, Wei Zhang, Lu Hou, Lifeng Shang, Jing Jin, Xin Jiang, Qun Liu, Michael Lyu, and Irwin King. Binarybert: Pushing the limit of bert quantization. *arXiv preprint arXiv:2012.15701*, 2020. 2

[2] Ron Banner, Yury Nahshan, and Daniel Soudry. Post training 4-bit quantization of convolutional networks for rapid-deployment. *Advances in Neural Information Processing Systems*, 32, 2019. 2, 6

[3] Jingyong Cai, Masashi Takemoto, and Hironori Nakajo. A deep look into logarithmic quantization of model parameters in neural networks. In *Proceedings of the 10th International Conference on Advances in Information Technology*, pages 1–8, 2018. 2, 5

[4] Yun-Hao Cao, Peiqin Sun, Yechang Huang, Jianxin Wu, and Shuchang Zhou. Synergistic self-supervised and quantization learning. In *European Conference on Computer Vision*, pages 587–604. Springer, 2022. 2

[5] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9640–9649, 2021. 5

[6] Yoojin Choi, Jihwan Choi, Mostafa El-Khamy, and Jungwon Lee. Data-free network quantization with adversarial knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 710–711, 2020. 2

[7] Yoni Choukroun, Eli Kravchik, Fan Yang, and Pavel Kisilev. Low-bit quantization of neural networks for efficient inference. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 3009–3018. IEEE, 2019. 6

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 3

[9] Agoston E Eiben and James E Smith. *Introduction to evolutionary computing*. Springer, 2015. 4

[10] Philip Fradkin, Lazar Atanackovic, and Michael R Zhang. Robustness to adversarial gradients: A glimpse into the loss landscape of contrastive pre-training. In *First Workshop on Pre-training: Perspectives, Pitfalls, and Paths Forward at ICML 2022*. 2, 4, 5, 6

[11] Yonggan Fu, Qixuan Yu, Meng Li, Xu Ouyang, Vikas Chandra, and Yingyan Lin. Contrastive quant: quantization makes stronger contrastive learning. In *Proceedings of the 59th ACM/IEEE Design Automation Conference*, pages 205–210, 2022. 2

[12] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020. 4

[13] Itay Hubara, Yury Nahshan, Yair Hanani, Ron Banner, and Daniel Soudry. Accurate post training quantization with small calibration sets. In *International Conference on Machine Learning*, pages 4466–4475. PMLR, 2021. 2

[14] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2704–2713, 2018. 2, 5

[15] Yuhang Li, Ruihao Gong, Xu Tan, Yang Yang, Peng Hu, Qi Zhang, Fengwei Yu, Wei Wang, and Shi Gu. Brecq: Pushing the limit of post-training quantization by block reconstruction. *arXiv preprint arXiv:2102.05426*, 2021. 2

[16] Zhikai Li, Mengjuan Chen, Junrui Xiao, and Qingyi Gu. Psaq-vit v2: Towards accurate and general data-free quantization for vision transformers. *arXiv preprint arXiv:2209.05687*, 2022. 2, 5

[17] Zhikai Li and Qingyi Gu. I-vit: Integer-only quantization for efficient vision transformer inference. *arXiv preprint arXiv:2207.01405*, 2022. 2

[18] Zhengang Li, Mengshu Sun, Alec Lu, Haoyu Ma, Geng Yuan, Yanyue Xie, Hao Tang, Yanyu Li, Miriam Leeser, Zhangyang Wang, et al. Auto-vit-acc: An fpga-aware automatic acceleration framework for vision transformer with mixed-scheme quantization. *arXiv preprint arXiv:2208.05163*, 2022. 2

[19] Zhexin Li, Tong Yang, Peisong Wang, and Jian Cheng. Q-vit: Fully differentiable quantization for vision transformer. *arXiv preprint arXiv:2201.07703*, 2022. 2, 9

[20] Yang Lin, Tianyu Zhang, Peiqin Sun, Zheng Li, and Shuchang Zhou. Fq-vit: Post-training quantization for fully quantized vision transformer. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 1173–1179, 2022. 2, 3

[21] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021. 6

[22] Zhenhua Liu, Yunhe Wang, Kai Han, Wei Zhang, Siwei Ma, and Wen Gao. Post-training quantization for vision transformer. *Advances in Neural Information Processing Systems*, 34:28092–28103, 2021. 2

[23] Markus Nagel, Rana Ali Amjad, Mart Van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or down? adaptive rounding for post-training quantization. In *International Conference on Machine Learning*, pages 7197–7206. PMLR, 2020. 2, 8

[24] Yury Nahshan, Brian Chmiel, Chaim Baskin, Evgenii Zheltonozhskii, Ron Banner, Alex M Bronstein, and Avi Mendelson. Loss aware post-training quantization. *Machine Learning*, 110(11):3245–3262, 2021. 2

[25] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 5

[26] Yuzhang Shang, Dan Xu, Ziliang Zong, Liqiang Nie, and Yan Yan. Network binarization via contrastive learning. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XI*, pages 586–602. Springer, 2022. 3

[27] Sangeetha Siddegowda, Marios Fournarakis, Markus Nagel, Tijmen Blankevoort, Chirag Patel, and Abhijit Khobare. Neural network quantization with ai model efficiency toolkit (aimet). *arXiv preprint arXiv:2201.08442*, 2022. 1

[28] Han Vanholder. Efficient inference with tensorrt. In *GPU Technology Conference*, volume 1, page 2, 2016. 1

[29] Di Wu, Qi Tang, Yongle Zhao, Ming Zhang, Ying Fu, and Debing Zhang. Easyquant: Post-training quantization via scale optimization. *arXiv preprint arXiv:2006.16669*, 2020. 2

[30] Sheng Xu, Yanjing Li, Teli Ma, Bohan Zeng, Baochang Zhang, Peng Gao, and Jinhu Lu. Tervit: An efficient ternary vision transformer. *arXiv preprint arXiv:2201.08050*, 2022. 2, 9

[31] Zhihang Yuan, Chenhao Xue, Yiqi Chen, Qiang Wu, and Guangyu Sun. Ptq4vit: Post-training quantization for vision transformers with twin uniform quantization. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XII*, pages 191–207. Springer Nature Switzerland Cham, 2022. 2