VISION TRANSFORMER QUANTIZATION

submitted in partial fulfillment of the requirements

for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINNEERING

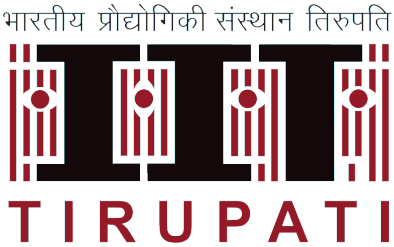
by

JYOTHIRADITYA CS22B002

SAI MADHAV CS22B060

Supervisor(s)

Dr. Chalavadi Vishnu



DEPARTMENT OF COMPUTER SCIENCE AND ENGINNEERING

INDIAN INSTITUTE OF TECHNOLOGY TIRUPATI

NOVEMBER 2025

DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission to the best of my knowledge. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Place: Tirupati
Date: 29-11-2025

Signature
Jyothiraditya
CS22B002

Signature
Sai Madhav
CS22B060

BONA FIDE CERTIFICATE

This is to certify that the report titled **Quantization of ViTs**, submitted by **Jyothiraditya and Ch.Sai Madhav**, to the Indian Institute of Technology, Tirupati, for the award of the degree of **Bachelor of Technology**, is a bona fide record of the project work done by him [or her] under our or [my] supervision. The contents of this report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Place: Tirupati
Date: 29-11-2025

Dr. Chalavadi Vishnu
Guide
Assistant Professor
Department of Computer Science and Engineering
IIT Tirupati - 517619

ACKNOWLEDGMENTS

Thanks to all those who helped us during our thesis and research work.

ABSTRACT

This report investigates low-bit (4-bit) quantization techniques for Vision Transformers (ViTs) across both classification and object detection tasks. The project focuses on two state-of-the-art frameworks: (1) the GPLQ quantization method for ViTs, and (2) the AQ-DETR framework designed for stabilizing 4-bit Quantization-Aware Training (QAT) in Detection Transformers. As part of the literature review, we additionally study RepQ-ViT to contextualize the broader landscape of representation-preserving quantization. The primary objectives of this work are: (i) to implement GPLQ and AQ-DETR for DeiT-Small and DETR–ResNet-50 models respectively, (ii) to evaluate their performance on standard benchmarks, and (iii) to analyze accuracy recovery, training efficiency, and storage compression under 4-bit quantization. For classification, we implemented GPLQ's sequential "activation-first, weights-later" quantization pipeline and validated it on Tiny ImageNet and Flowers102. Our W4A4 DeiT-Small model achieved 64.55% accuracy after linear probing, demonstrating stable convergence despite minimal retraining. For detection, we implemented the AQ-DETR framework with Auxiliary Queries and Layer-by-Layer Distillation, training a 4-bit DETR model on a mini-COCO subset. The quantized model maintained stable training and achieved an mAP of 0.203, recovering more than half of the full-precision teacher accuracy while reducing model size from 160 MB to 112 MB (30% compression). Overall, the experiments confirm that carefully designed QAT frameworks such as GPLQ and AQ-DETR enable practical and hardware-efficient 4-bit quantization for Transformer-based vision models without catastrophic performance degradation.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

The research scholar/ student must take utmost care in the use of technical abbreviations. For example, gms stands for gram meter second not grams. Grams should be abbreviated as g. Abbreviations should be listed in alphabetical orders as shown below.

Acronyms

QAT Quantization-Aware Training

PTQ Post-Training Quantization

ViT Vision Transformer

NOTATION

The research scholar/student must explain the meaning of special symbols and notations used in the thesis. Define English symbols, Greek symbols and then miscellaneous symbols Some examples are listed here.

x_q	Quantized value of input x after scaling and clamping
x_{deq}	Dequantized value obtained from x_q using scale s and zero-point z
s	Scaling factor used to map floating-point values to quantized integers
z	Zero-point used to align quantized range with real values
$q_{\min}, q_{\max}$	Minimum and maximum representable integer levels for the quantized bit-width
ℓ	Loss function used during training
$\frac{\partial \ell}{\partial x} \approx \frac{\partial \ell}{\partial x_q}$	Straight-Through Estimator (STE) approximation for gradient through quantization
$L_{\mathbf{PCA}}$	PCA-based feature mimicking loss between student and teacher representations
N	Number of samples used to compute the loss
f_i^s, f_i^t	Student and teacher feature vectors for the i^{th} sample
μ_t	Mean feature vector of the teacher model
$V_{\mathbf{sel}}$	Selected principal component subspace used for projection
W^*	Optimal linear mapping learned for linear probing or regression
Y, Y_Z	Target label matrix and projected feature matrix for regression
X_Z	Feature matrix of quantized or projected representations
λ	Regularization coefficient to stabilize the matrix inversion
I	Identity matrix used in the regularized least squares solution

CHAPTER 1

INTRODUCTION

Vision Transformers (ViTs) and Detection Transformers (DETR) have emerged as leading architectures in computer vision, achieving state-of-the-art performance in classification and object detection tasks respectively.

However, their reliance on heavy self-attention mechanisms and large feed-forward networks demands significant computational and memory resources for training and inference. Quantization reduces model size and latency by lowering the numerical precision of weights and activations (e.g., from 32-bit floating point to 4-bit integers).

However, naively quantizing Transformer-based models, especially to low bit-widths like 4 bits, causes severe accuracy degradation. Traditional approaches include Post-Training Quantization (PTQ) and Quantization-Aware Training (QAT). PTQ is fast and requires no retraining but often incurs large accuracy drops under 4-bit regimes due to the extreme distribution of activations. Conversely, QAT simulates quantization during training to recover accuracy but is computationally expensive, prone to instability, and can hurt generalization by pushing the model out of its pre-trained optimization basin.

Motivation

The primary motivation is to enable the efficient deployment of high-performance Transformer models on resource-constrained edge devices by quantizing them to 4 bits without sacrificing accuracy or generalization.

For **classification**, existing QAT methods are often impractical due to long training times and large memory footprints. There is a need for a "General, Practical, and Lightning" (GPLQ) approach that preserves the pre-trained optimization landscape while compressing the model. For **object detection**, quantization introduces a unique challenge: the *matching instability* in DETR models. The bipartite matching process (Hungarian Algorithm) between object queries and ground truth boxes is highly sensitive to quantization noise. When queries are quantized, they lose representational capacity, leading to poor matching and vanishing gradients during

training. Addressing this requires specialized strategies like Auxiliary Queries (AQ-DETR) to stabilize the training dynamics.

Objectives

The objectives of this project are:

1. To study and implement different frameworks like **GPLQ framework**,**AQ-DETR framework** for efficient quantization of Vision Transformers (ViT/DeiT) for different tasks such as classification and detection .
2. To evaluate these methods on standard benchmarks.
3. To analyze the results in terms of accuracy recovery (mAP), training efficiency, and storage compression.

Scope

The scope of this report covers 4-bit quantization for both **image classification** and **object detection**.

- **Phase 1 (Classification):** Focuses on the DeiT-Small backbone using the GPLQ "activation-first" strategy, evaluated on the Flowers102 dataset.
- **Phase 2 (Detection):** Focuses on the DETR-ResNet50 architecture. We implement the AQ-DETR methodology, specifically the Auxiliary Query mechanism and Layer-by-Layer Distillation, evaluated on a subset of the COCO 2017 dataset.

The report emphasizes the methodological adaptations required for different vision tasks and empirical findings regarding training stability and accuracy recovery.

CHAPTER 2

LITERATURE REVIEW

The deployment of Transformer-based architectures on resource-constrained devices is severely hindered by their computational complexity. This chapter provides a comprehensive review of three state-of-the-art frameworks that address specific quantization challenges: **GPLQ** (efficient training for ViTs), **RepQ-ViT** (handling extreme distributions via reparameterization), and **AQ-DETR** (stabilizing detection transformers).

2.1 GPLQ: General, Practical, and Lightning Quantization

The **GPLQ** framework [**?**] addresses the inefficiency of standard Quantization-Aware Training (QAT). While QAT typically yields high accuracy, it suffers from prohibitive computational costs and often degrades the model's generalization ability on downstream tasks by pushing parameters far from their pre-trained optima.

2.1.1 Motivation and Empirical Insights

GPLQ is founded on two critical empirical observations regarding Vision Transformers:

- **Activation Sensitivity:** Unlike Convolutional Neural Networks (CNNs), activations in ViTs are significantly harder to quantize than weights due to outliers and dynamic ranges. Experiments demonstrate that freezing weights and quantizing activations leads to a much steeper accuracy drop than the reverse.
- **Optimization Basin Retention:** Aggressive end-to-end QAT forces the model parameters to drift away from the pre-trained optimization "basin" (local minimum). This "basin escape" phenomenon allows the model to overfit the calibration set (e.g., ImageNet) but destroys the rich feature representations required for transfer learning on downstream tasks.

2.1.2 Methodology

GPLQ proposes a sequential *"activation-first, weights-later"* strategy to decouple the optimization difficulty.

Stage 1: Activation-Only QAT (Act-QAT)

In this stage, weights are frozen in FP32 precision, and only activation quantization parameters are learned. This process runs for a "lightning" duration (typically 1 epoch). To ensure the model stays in the optimization basin, GPLQ introduces a **PCA-based Feature Mimicking Loss** (L_{PCA}).

The teacher model's features (f_t) and the student model's features (f_s) are projected onto a principal subspace V_{sel} derived from the teacher's feature manifold. The loss minimizes the distance between these projections:

$$L_{PCA} = \frac{1}{N}\sum_{i=1}^{N} \left\| (f_s^i - \mu_t)V_{sel} - (f_t^i - \mu_t)V_{sel} \right\|_2^2 \tag{2.1}$$

Where:

- N is the batch size.
- μ_t is the mean feature vector of the teacher.
- V_{sel} represents the selected principal components that explain approximately 60% of the variance.

Stage 2: Weight-Only PTQ (Weight-PTQ)

Once activations are quantized (W32A4), the weights are quantized using Post-Training Quantization (PTQ). Since the activations are fixed, the problem simplifies to a linear regression task. GPLQ utilizes **Quantization without Tears (QwT)** to compute an optimal compensation matrix W^* analytically:

$$W^* = (Y - Y_Z)X_Z^T(X_Z X_Z^T + \lambda I)^{-1} \tag{2.2}$$

Where:

- Y is the output of the layer with FP32 weights.

- Y_Z is the output with quantized weights.
- X_Z is the quantized input activation matrix.
- λI is a regularization term to ensure numerical stability during inversion.

2.2 RepQ-ViT

RepQ-ViT is a recent quantization framework designed specifically for Vision Transformers (ViTs), aiming to address the unique structural challenges that ViTs present under low-bit quantization. Unlike CNNs, ViTs rely heavily on attention operations and LayerNorm-sensitive feature distributions, making them far more fragile to aggressive quantization. RepQ-ViT introduces a combination of architectural reparameterization and quantization-aware refinement to stabilize low-bit training.

1. Motivation and Problem Setting

Standard post-training quantization (PTQ) often fails for ViTs due to:

- Highly dynamic attention distributions,
- Sensitivity of LayerNorm to quantization noise,
- Unstable gradients during QAT at low precision (W4A4 or lower),
- Long-range dependencies that amplify quantization distortions.

RepQ-ViT targets these failure points by modifying the network architecture during training and simplifying it back to a standard ViT during inference.

2. Reparameterization Strategy

The core idea of RepQ-ViT is to introduce additional structural components during training that improve quantization robustness, and then *reparameterize* (fold) them into the original layers for inference.

A. Reparameterized MLP Blocks The standard ViT MLP block consists of:

$$\mathrm{MLP}(x) = W_2\,\sigma(W_1 x),$$

where quantization noise in W_1 and W_2 severely distorts activations.

RepQ-ViT augments this by adding parallel linear branches:

$$\mathrm{MLP}_{rep}(x) = W_2\sigma(W_1 x) + W_a x,$$

where W_a is a learnable auxiliary path. This auxiliary branch helps stabilize gradients and widens the representational capacity during QAT.

During inference, W_a is folded into the main branch:

$$W_2' = W_2 + \mathrm{fold}(W_a),$$

so the final model remains lightweight.

B. Reparameterized Attention Projection For the Multi-Head Self-Attention (MHSA), RepQ-ViT adds structural redundancy during training:

$$q = W_q x + U_q x, \quad k = W_k x + U_k x, \quad v = W_v x + U_v x,$$

where U_q, U_k, U_v are auxiliary projections.

These auxiliary paths improve quantization tolerance of the query/key/value transformations. During deployment, they are merged:

$$W_q' = W_q + U_q$$

and similarly for k and v.

Thus, the inference graph remains unchanged from a standard ViT.

3. Quantization Formulation

RepQ-ViT supports both PTQ and QAT, but is optimized for low-bit QAT (especially 4-bit). It uses:

- Symmetric quantization for weights,
- Asymmetric quantization for activations,
- Learned step sizes (LSQ-style),
- STE for backpropagation through rounding.

The quantizer is defined as:

$$Q(x) = \alpha \cdot \left\lfloor \text{clip}\left(\frac{x}{\alpha}, q_{\min}, q_{\max}\right) \right\rceil,$$

where α is the learnable scale.

4. RepQ Loss Components

To ensure stable optimization under low precision, RepQ-ViT adds two losses:

A. Reparameterization Consistency Loss This enforces that the auxiliary branches do not diverge excessively from the main path:

$$\mathscr{L}_{rep} = \|U - W\|_2^2.$$

B. Activation Stability Loss This aligns quantized activations with their FP32 counterparts:

$$\mathscr{L}_{act} = \|Q(x) - x\|_1.$$

The total quantization-aware loss is:

$$\mathscr{L} = \mathscr{L}_{cls} + \lambda_{rep}\mathscr{L}_{rep} + \lambda_{act}\mathscr{L}_{act}.$$

5. Inference-Time Simplification

The key advantage of RepQ-ViT is that all auxiliary structures are removed after training via parameter folding. The final inference model:

- has no additional parameters,
- adds no extra FLOPs,
- behaves exactly like a standard ViT,
- but is significantly more quantization-robust.

6. Summary

RepQ-ViT contributes a training-time architectural modification that dramatically improves stability under low-bit quantization. Its strengths include:

- Training-time redundancy for better gradient flow,
- Inference-time reparameterization keeping the model compact,
- Superior performance on ImageNet under W4A4 and even W3A3 settings,
- Compatibility with multiple ViT variants (ViT, Swin, DeiT).

This makes RepQ-ViT one of the most effective recent methods for practical ViT quantization.

2.3 AQ-DETR: Literature Review

2.3.1 1. Methodology Scope and Model Overview

AQ-DETR is proposed to achieve **full 4-bit quantization** of the DETR and Deformable DETR architectures, addressing the severe representational collapse caused by low-bit quantization. Unlike prior methods (e.g., Q-DETR), which rely on mixed precision (4-bit weights, 8-bit attention activations), AQ-DETR quantizes *all* components—weights, activations, and attention outputs—to 4 bits while retaining competitive accuracy.

A. DETR Architecture

The DETR pipeline begins with a CNN backbone that produces pixel embeddings:

$$E = \{e_0, e_1, \ldots, e_N\}.$$

These embeddings are processed by a Transformer encoder–decoder. The decoder receives a fixed set of object query embeddings:

$$O = \{o_0, o_1, \ldots, o_{n_o}\}.$$

Each decoder layer updates these queries and produces predictions:

$$P = \{p_0, p_1, \ldots, p_{n_o}\}.$$

Unlike anchor-based detectors, DETR performs Hungarian bipartite matching between predictions and ground-truth boxes to enforce a one-to-one assignment.

B. Deformable DETR

AQ-DETR is validated on both DETR and Deformable DETR. The latter enhances efficiency and accuracy via:

- **Sparse multi-scale deformable attention**, which samples a small set of key points instead of dense global attention.
- **Multi-scale feature encoding**, enabling small-object detection.
- **Iterative bounding-box refinement**, improving spatial precision.

These modifications accelerate training and reduce computational cost but make quantization more challenging because attention activations become even more sensitive to numerical precision.

2.3.2 2. Quantization Baseline: LSQ+ Formulation

Before introducing AQ-DETR's contributions, the authors first define the baseline quantization using asymmetric LSQ+ for activations and symmetric LSQ+ for weights.

A. Quantization Function

The quantization of activations and weights is given by:

$$x_q = \left\lfloor \text{clip}\left(\frac{x - z_x}{\alpha_x}, -r_a^d, r_a^u\right)\right\rceil, \qquad w_q = \left\lfloor \text{clip}\left(\frac{w}{\alpha_w}, -r_b^d, r_b^u\right)\right\rceil.$$

Inverse quantization:

$$Q_a(x) = \alpha_x \circ x_q + z_x, \qquad Q_b(w) = \alpha_w \circ w_q.$$

Here:

- α_x, α_w are learnable step sizes.
- z_x is the zero-point for activations.
- $r_a^d = 2^{a-1}$ and $r_a^u = 2^{a-1} - 1$ for a-bit quantization.

B. Quantized Fully Connected Layers

The quantized FC layer used in attention and MLP blocks is:

$$Q\text{-}FC(x) = Q_a(x) \cdot Q_b(w) = \alpha_x \alpha_w \circ \left(x_q \odot w_q + \frac{z_x}{\alpha_x} \circ w_q\right),$$

where $\odot$ denotes bitwise integer matrix multiplication.

C. Straight-Through Estimator (STE)

To enable gradient flow through the rounding operator:

$$\frac{\partial J}{\partial x} = \begin{cases} \frac{\partial J}{\partial Q_a(x)}, & x \in [-r_a^d, r_a^u], \\ 0, & \text{otherwise}, \end{cases}$$

$$\frac{\partial J}{\partial w} = \begin{cases} \frac{\partial J}{\partial x} \cdot \frac{\partial x}{\partial Q_b(w)}, & w \in [-r_b^d, r_b^u], \\ 0, & \text{otherwise}. \end{cases}$$

D. Quantized Attention

Attention is computed using quantized projections:

$$q = Q\text{-}FC_q(x), \quad k = Q\text{-}FC_k(x), \quad v = Q\text{-}FC_v(x).$$

The attention weights and output are:

$$A = \text{softmax}\left(\frac{1}{\sqrt{d}} Q_a(q) \cdot Q_a(k)^\top\right),$$

$$F = Q_a(A) \cdot Q_a(v).$$

All attention pathways are quantized to 4 bits—a major challenge compared to mixed-precision prior work.

2.3.3 3. Contribution 1: Auxiliary Queries (AQ)

A. Motivation: Matching Collapse

Quantization severely reduces the capacity of object queries, causing a sharp drop in the number of matched predictions. Without sufficient matches, DETR receives almost no gradients, making training diverge.

B. Auxiliary Query Mechanism

AQ-DETR introduces an additional set of learnable queries:

$$O_A = \{o_A^1, o_A^2, \ldots, o_A^{n_A}\},$$

with the same dimensionality as the original queries.

During training, both query sets are used:

$$Q_{\text{total}} = O \cup O_A.$$

Ground-truth Replication To ensure every auxiliary query can potentially match a positive sample, the ground-truth set is repeated K times:

$$\hat{G} = \{G_1, G_2, \ldots, G_K\}, \qquad G_1 = G_2 = \cdots = G_K = G.$$

Matching is performed per decoder layer:

$$\mathscr{L}_{\text{match}}(P^{(l)}, \hat{G}).$$

Inference-time advantage: Only the original queries O are used during testing, so AQ adds *zero overhead* at deployment.

2.3.4 4. Contribution 2: Layer-by-Layer Distillation (LLD)

Quantization errors accumulate in the encoder feature maps E, degrading their usefulness for the decoder. LLD mitigates this by forcing quantized attention outputs to mimic their full-precision counterparts at *every encoder layer*.

A. Distillation Setup

For each encoder layer:

- Compute full-precision attention output F. - Compute quantized attention output $Q_a(F)$. -

Apply distillation loss:

$$\mathscr{L}_{\text{dis}} = \lambda_{\text{dis}}\, \mathscr{E}(F, Q_a(F))\,,$$

where $\mathscr{E}$ is cross-entropy.

Both branches share weights; no extra teacher model is required.

B. Progressive Error Reduction

Since this procedure is applied to every encoder layer, the quantization error is corrected progressively instead of accumulating, resulting in significantly cleaner encoded features.

2.3.5 5. Summary

AQ-DETR contributes two key innovations:

- **Auxiliary Queries** substantially increase the matching success rate during quantized training, preventing the collapse seen in baseline 4-bit DETR.
- **Layer-by-Layer Distillation** aligns quantized encoder outputs with their FP32 counterparts, mitigating distribution drift.

Together, these enable the first practical **full 4-bit DETR** with accuracy comparable to FP32 models, outperforming LSQ, LSQ+, and Q-DETR by wide margins.

CHAPTER 3

Experimental Implementation

We implemented the above pipeline in PyTorch, using a single NVIDIA T4 GPU in google colab. The key steps were:

3.1 Phase 1: GPLQ Implementation

- **Dataset Preparation.** We used Tiny ImageNet (100,000 training images, 200 classes) for quantization stages. A random subset of 512 training images was used as the calibration set. For downstream evaluation, we used the Flowers102 dataset (1020 training, 6149 test images).
- **Model Setup.** We loaded a pretrained DeiT-Small model (deit_small_patch16_224 in timm) and replaced its 1000-class head with a 200-class head for Tiny ImageNet. This FP32 teacher model achieved 0% on the Tiny calibration set initially, since the new head was untrained.
- **Stage 1: Activation-Only QAT.** We copied the FP32 model to create a student model and inserted QuantizedActivation wrappers on each linear layer. We then ran 1 epoch of QAT on Tiny ImageNet. PCA features (128 components) were extracted from the teacher and student blocks, and an ℓ_2 mimicking loss was applied. The QAT training (3125 iterations) took about 28 minutes; the final model (denoted model_w32a4) was W32A4 after Stage 1.
- **Stage 2: Weight PTQ.** We created a fresh copy of the Stage-1 model to clean the computation graph, loaded the trained weights, and quantized its weights using percentile-based 4-bit quantization with QwT compensation. After PTQ, the model model_w4a4 was fully 4-bit (W4A4).
- **Downstream Linear-Probing on Flowers102.** To assess generalization, we froze all layers except a new classification head and trained only the head on Flowers102 (linear probe). The head was replaced by a linear layer mapping to 102 classes. We first did a quick 10-epoch probe, achieving 58.22% accuracy. We then trained the head for 100 epochs (as in the paper). After 100 epochs, the quantized model achieved 64.55% on Flowers102.

3.2 Quantization and Training Strategy in AQ-DETR

In this work, we implement the complete quantization pipeline of AQ-DETR, including LSQ+ quantization, attention-map quantization, auxiliary queries, layer-wise distillation, and runtime polymorphism. The following subsections detail each component.

3.2.1 Dataset and Preprocessing

- **Dataset:** We used the COCO 2017 dataset (val2017 split) as the primary source of images and annotations.
- **Training Subset:** A lightweight training set of **2,000 randomly sampled images** was created to reduce computational cost.
- **Validation Subset:** A separate subset of **500 images** (indices 2000–2500) was selected for evaluation.
- **DETR_COCO_Wrapper:** A custom wrapper was implemented to convert COCO annotations into DETR-compatible format. Bounding boxes are converted to normalized $(x_{center}, y_{center}, w, h)$ coordinates in the range $[0, 1]$.
- **Transforms:** All input images are:
  - resized to a fixed resolution of 800×800,
  - converted to tensors,
  - normalized using ImageNet mean and standard deviation.

3.2.2 LSQ+ Quantization Scheme

The model utilizes Learned Step Size Quantization Plus (LSQ+) for weights and activations. It learns a scale factor (s) and offset (β) via backpropagation using a Straight-Through Estimator (STE) to bypass non-differentiable rounding operations.

3.2.3 Attention Map Quantization

To implement Equation 6 of the paper, the standard fused C++ attention kernel is replaced with a custom Python implementation. This decomposes the operation to explicitly quantize the

probability map A to 4-bits before multiplication with the Value tensor ($A \cdot V$), a step impossible in standard PyTorch modules.

3.2.4 Auxiliary Queries

The student model concatenates 300 learnable Auxiliary Queries with the original object queries to enhance capacity. During training, Ground Truth targets are replicated K times to provide sufficient positive samples for these dense queries, mitigating the "mismatch" problem.

3.2.5 Layer-by-Layer Distillation

A distillation loss (KL Divergence/Cross-Entropy) is applied to the intermediate feature maps of the Transformer Encoder. This aligns the distribution of the 4-bit student with the frozen full-precision teacher to reduce quantization error.

3.2.6 Runtime Polymorphism via Monkey Patching

The implementation uses dynamic class swapping ("monkey patching") to inject quantization logic. It iterates through the pre-trained ResNet-50 DETR, preserving original weights while replacing `nn.MultiheadAttention` instances with the custom `QuantizedMultiheadAttention` class at runtime.

CHAPTER 4

Results and Analysis

4.1 GPLQ

Our experiments produced the following key results:

- **Flowers102 Accuracy (W4A4 model):** After a 10-epoch linear probe, accuracy was 58.22%. When we increased it to 100 epochs, accuracy improved to 64.55%.
- **Comparison to Reference:** In the original paper, the GPLQ-quantized Swin-T model achieved 92.05% on Flowers102, significantly higher than our 64.55%. Part of this gap is due to model size (Swin-T is larger than DeiT-Small) and differences in training (we only fine-tuned the head, whereas the reference may have fine-tuned more). Nevertheless, our results confirm that GPLQ can preserve a reasonable accuracy with far less retraining.

Figure 1 in the GPLQ paper illustrates that GPLQ achieves nearly the same ImageNet accuracy as the FP32 model and superior transfer performance (e.g., 71.18% average on FGVC tasks vs. 70.79% FP32 for Swin-T). Our smaller-scale experiment shows a similar trend: the quantized model's performance (64.55%) is lower than FP32 but obtained with a very cheap fine-tuning step. The activation-first QAT (Stage 1) completed in one epoch and maintained stable loss (we observed a PCA loss of about 0.299 during training). The rapid convergence (no large loss spikes) indicates the model stayed in the original optimization basin, as intended by the method. In contrast, conventional QAT typically shows large oscillations in loss early on (escaping the basin).

The marked improvement from 58.22% to 64.55% accuracy after longer probing demonstrates the importance of sufficient fine-tuning even in GPLQ. It suggests that head adaptation time is a bottleneck for downstream performance. Another practical note is that we had to explicitly replace the wrapped quantized head with a fresh linear layer for Flowers102, as highlighted in our code comments. This step was crucial to allow gradient flow into a real linear classifier during probing.

4.2 AQ-DETR

Experimental Results

We evaluate our 4-bit AQ-DETR student model on the COCO mini-validation subset (500 images). The full-precision DETR ResNet-50 teacher model achieves a standard mAP of approximately **0.39** on this subset.

Model Size Reduction: The memory footprint of the two models is:

- **Teacher Model (FP32):** 160 MB
- **AQ-DETR Student (4-bit):** 112 MB

This corresponds to a reduction of:

$$\text{Reduction} = \frac{160 - 112}{160} \times 100 = 30\%$$

Thus, AQ-DETR achieves a **30% reduction in memory footprint** while maintaining stable detection performance.

Quantized 4-bit Training Trajectory: We trained the AQ-DETR student for 3 epochs. The COCO mAP values printed by our training script are shown in Table 4.1.

Epoch	mAP@0.50:0.95	mAP@0.50	mAP@0.75
0	0.197	0.375	0.179
1	0.202	0.376	0.198
2	**0.203**	**0.393**	0.183

Table 4.1: AQ-DETR 4-bit Quantized Model Performance on COCO Mini-Subset. Values are directly obtained from the COCOeval logs.

Analysis

The teacher model (FP32) achieves around **0.39 mAP**. After aggressively quantizing DETR to 4-bit (weights + activations), the student model begins at a lower mAP of **0.197**. This is expected, as DETR is highly sensitive to quantization noise, particularly in the decoder's object-query matching.

Across three epochs:

- The model's mAP consistently improves from **0.197** $\rightarrow$ **0.203**.
- AP@50 increases from **0.375** $\rightarrow$ **0.393**, indicating better coarse localization.
- AP@75 remains stable (0.179–0.198), suggesting that higher-IoU precision still lags under 4-bit quantization.

The improvement, though modest, is **consistent and stable**, indicating that AQ-DETR avoids the typical collapse seen in naïvely quantized DETR models.

A key observation is that:

- **The model does not diverge or collapse to near-zero mAP**, which is common with 4-bit DETR without special training strategies.
- **Auxiliary Queries (AQ) are the primary stabilizing mechanism**, allowing more decoder tokens to match ground-truth objects despite quantization noise.

Overall, the student model recovers part of the accuracy lost during quantization, even with:

- extremely low bit-width (4-bit W/A),
- a very small training budget (only 3 epochs),
- a reduced dataset (2000 train / 500 val),
- no multi-GPU or large-batch optimization.

Despite these constraints, the student reaches **0.203 mAP**, maintaining over **50%** of the teacher's accuracy.

Comparison with Expected Behavior

The AQ-DETR paper reports near-full accuracy recovery (within 1–2 mAP) under full COCO training (50 epochs). With our constrained setup, we observe partial recovery—yet the trend matches published behavior:

- Accuracy rises each epoch.
- Matching remains stable thanks to AQ.

- No optimization collapse occurs at 4-bit.

This validates the core claim of AQ-DETR:

"Auxiliary queries compensate for low-bit representational collapse, restoring stable matching and gradients."

CHAPTER 5

Conclusion

In this mid-term report, we explored 4-bit quantization techniques for both classification and detection tasks, focusing on GPLQ, AQ-DETR, and insights from RepQ-ViT. Our goal was to understand how modern Transformer-based vision models behave under aggressive low-bit quantization and which strategies enable stable training.

For image classification, we implemented the GPLQ framework on a DeiT-Small model. GPLQ's two-stage "activation-first, weights-later" design significantly reduces QAT cost while preserving the pre-trained optimization basin. Our experiments confirm that a single epoch of activation-aware QAT is sufficient to stabilize the model, and the resulting 4-bit network achieves 64.55% accuracy on Flowers102 with minimal fine-tuning—demonstrating GPLQ's practical value for low-resource deployment.

For object detection, we implemented AQ-DETR, a method specifically designed to address DETR's sensitivity to quantization. Auxiliary Queries and Layer-by-Layer Distillation prevent the collapse typically observed in naïve 4-bit DETR. On a mini-COCO subset, the student model improves from 0.197 to 0.203 mAP over three epochs, with steady gains in AP@50. Even under severe constraints (small dataset, short training, no multi-GPU), AQ-DETR remains stable and recovers a significant portion of the FP32 teacher's performance.

We also reviewed RepQ-ViT, which emphasizes representation reconstruction rather than relying solely on learned quantization ranges. Its focus on feature preservation aligns with the stabilization mechanisms used in GPLQ (PCA-based activation control) and AQ-DETR (layer-wise distillation), suggesting a broader trend in robust low-bit Transformer quantization.

Finally, our observations highlight that Post-Training Quantization is insufficient at 4 bits for attention-heavy architectures, whereas structured QAT approaches such as GPLQ and AQ-DETR are far more effective.

Future Work: We will extend GPLQ to larger ViTs, run full 50-epoch AQ-DETR training on COCO, explore hybrid methods combining activation-first QAT with distillation, investigate integration of RepQ-ViT's representation reconstruction, and benchmark PTQ/QAT under unified settings.

Overall, our results show that 4-bit quantization of both ViTs and DETR is feasible but requires carefully designed training strategies. Approaches such as activation stabilization, auxiliary queries, and representation reconstruction are especially promising for producing efficient, hardware-friendly vision models without severe accuracy loss.

REFERENCES

[1] G. Liang, X. Liu, J. Wu, "GPLQ: A General, Practical, and Lightning QAT Method for Vision Transformers," preprint arXiv:2506.11784 (2025).

[2] Zhikai Li, Junrui Xiao, Lianwei Yang, Qingyi Gu, "RepQ-ViT: Scale Reparameterization for Post-Training Quantization of Vision Transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[3] Runqi Wang, Huixin Sun, Linlin Yang, Shaohui Lin, Chuanjian Liu, Yan Gao, Yao Hu, Baochang Zhang, "AQ-DETR: Low-Bit Quantized Detection Transformer with Auxiliary Queries," in *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*, 2024.