# Amortized-Precision Quantization for Early-Exit Vision Transformers

**Rui Fang*** **Hsi-Wen Chen*** **Ming-Syan Chen**
National Taiwan University
{rfang, hwchen}@arbor.ee.ntu.edu.tw
mschen@ntu.edu.tw

## Abstract

Vision Transformers (ViTs) achieve strong performance across vision tasks, yet their deployment with low-precision early exiting remains fragile. Existing quantization methods assume static full-depth execution, making them unstable when exit decisions are perturbed by quantization noise, which can amplify errors along dynamic inference paths. In this paper, we introduce **Amortized-Precision Quantization (APQ)**, a utilization-aware formulation that accounts for layer utilization when modeling quantization error and reveals *depth–precision* trade-offs. Building on APQ, we propose **Mutual Adaptive Quantization with Early Exiting (MAQEE)**, a bi-level framework that jointly optimizes exit thresholds and bit-widths under explicit risk control to improve inference stability. MAQEE establishes a superior *Pareto frontier* in the accuracy–efficiency trade-off across classification, detection, and segmentation, with comparisons at predefined task-specific performance targets. On ImageNet, it achieves 79.41% Top-1 accuracy while reducing BOPs by 10.9% relative to utilization-only allocation. Measurements of the deployed models on an RTX 5090 show GPU throughput improvements of 2.26× on ImageNet and 3.06× on CIFAR-100 over full-depth FP32 inference. Source code is available at https://github.com/Rui-Fang/MAQEE.

## 1 Introduction

Vision Transformers (ViTs) [7] achieve strong performance across classification [3], detection [2], and segmentation [10], but their high computational cost limits deployment on edge devices [33, 48]. Quantization [4] improves efficiency by reducing weight and activation precision. Early approaches adopt *Fixed-Precision Quantization (FPQ)* [14, 17, 47], but uniform bit-width assignment ignores the heterogeneous sensitivity of ViT layers to quantization noise [23, 36]. *Mixed-Precision Quantization (MPQ)* [16, 43] partially addresses this issue by assigning layer-wise bit-widths based on sensitivity, yet remains designed for static, full-depth inference.

However, modern ViT architectures increasingly rely on *dynamic inference*, such as Early Exiting (EE) [19, 37, 46], where lightweight exit heads enable input-dependent termination to reduce latency. This creates a mismatch with conventional quantization: MPQ calibrates precision under a fixed execution path, whereas EE induces stochastic, input-dependent depths [28].[2] As shown in the left panel of Fig. 1, quantization noise can perturb confidence estimates for early exiting, leading to either *premature* or *delayed* exits. Our preliminary results show this effect: quantizing EE-trained models causes up to a 50% accuracy drop, while applying EE to already quantized models makes it difficult to properly train exit heads, resulting in limited efficiency gains. Low-precision weights limit the flexibility needed to learn reliable exit decisions.

*Equal contribution.

[2]Mixture of Experts (MoE) [13, 30] is another dynamic inference paradigm, but unlike EE, it selects a fixed sub-network per input rather than changing the executed depth. Thus, EE is more sensitive to quantization error.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

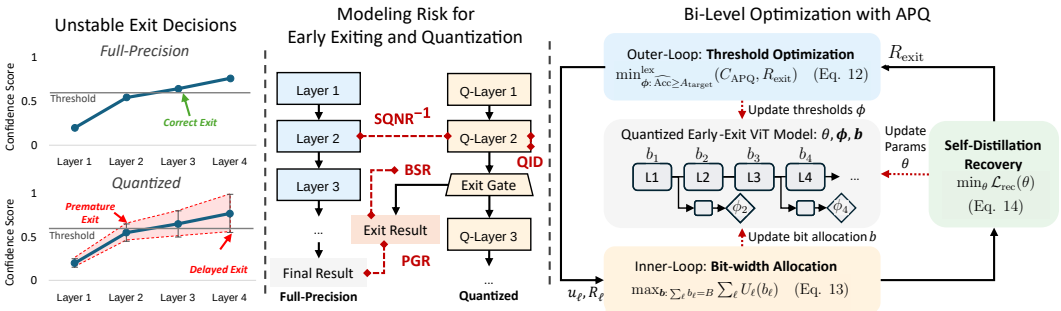



Figure 1: **Overview of MAQEE.** *Left*: Quantization error perturbs early-exit decisions, causing either premature or delayed exits. *Middle*: Risk modeling for early exiting and quantization, including performance gap risk (PGR), boundary sensitivity risk (BSR), inverse SQNR, and quantization-induced drift (QID). *Right*: MAQEE solves Amortized-Precision Quantization (APQ) via bi-level optimization, where exit thresholds and layer-wise bit-widths are jointly optimized, with self-distillation for accuracy recovery.

Motivated by these observations, we propose **Amortized-Precision Quantization (APQ)**, a quantization paradigm tailored to early-exit inference. Unlike conventional approaches that assume a fixed execution path, APQ treats precision as a resource whose impact depends on dynamic layer utilization. This induces a utilization-aware depth–precision trade-off: precision allocation should account for how often each layer is reached, how its quantization error affects subsequent inference paths, and how it may mislead exit decisions. Our theoretical analysis shows that static precision allocation fails to capture these effects, explaining the robustness degradation observed when quantization is naively combined with early exiting.

To effectively optimize APQ, we introduce **Mutual Adaptive Quantization with Early Exiting (MAQEE)**, a bi-level optimization framework that jointly optimizes exit thresholds and layer-wise bit-widths, as illustrated in the middle and right panels of Figure 1. MAQEE models instability from both exit-decision uncertainty and representation distortion, using four risk terms to capture performance degradation, boundary sensitivity, quantization noise, and changes in residual-update magnitude. These risks are aggregated into a unified risk score that guides the alternating optimization of thresholds and bit-widths within a bi-level iterative procedure. With lightweight self-distillation and efficient warm-starting, MAQEE improves low-precision recovery and accelerates convergence.

**Our contributions are summarized as follows.** First, we formalize the incompatibility between quantization and early exiting, and propose Amortized-Precision Quantization (APQ) to characterize inference complexity under dynamic execution. Second, we propose Mutual Adaptive Quantization with Early Exiting (MAQEE), a bi-level framework that jointly optimizes exit thresholds and bit-width allocation through risk-aware control. Third, we theoretically explain why static FPQ and MPQ are mismatched with early exiting and show that MAQEE enables more stable utilization-aware optimization. Finally, experiments across three vision tasks show that MAQEE forms a superior *Pareto frontier* in the accuracy–efficiency trade-off. Comparisons with joint quantization–early-exit methods and simple allocation strategies assess the benefit of risk-aware optimization, while GPU measurements establish the efficiency of the deployed models.

## 2 Related Work

Vision Transformers (ViTs) [7] capture long-range dependencies through self-attention [27, 40], but their high computation and memory costs limit deployment in real-time or resource-constrained settings [33, 48]. Quantization reduces this burden with low-bit weights and activations [4, 17]. Fixed-Precision Quantization (FPQ) [14, 47] applies uniform bit-widths but ignores heterogeneous layer sensitivity in ViTs [23, 36], leading to weak robustness and suboptimal accuracy–efficiency trade-offs [11]. Mixed-Precision Quantization (MPQ) [16, 43], along with PTQ [6, 21, 25, 35] and QAT [20, 26, 41], improves layer-wise precision allocation, but still assumes a static full-depth inference path. In parallel, dynamic inference adapts computation to input complexity by activating only part of the network [12, 13, 30]. Early Exiting (EE) [32, 45] attaches auxiliary classifiers to intermediate layers so confident samples terminate early, with ViT-based extensions showing promising efficiency gains [1, 28, 46]. However, quantization noise can disrupt confidence estimation and

cause premature or delayed exits. McQueen [31] learns layer-wise precision under a global budget in early-exit networks. QuEE [29] selects among classifiers with different exit depths and quantization levels. MAQEE jointly optimizes precision and exit thresholds using layer utilization and quantization risk, then deploys the resulting mixed-precision allocation. We introduce *Amortized-Precision Quantization* to analyze how quantization and early exiting affect each other.

## 3 Problem Formulation

This section introduces **Amortized-Precision Quantization (APQ)** and formulates its corresponding optimization problem. A detailed notation table is provided in Appendix D.

**Quantization.** Quantization replaces floating-point operations with low-bit integer arithmetic to reduce memory footprint and inference cost [14]. Given bit-width $b$, a scale factor $s$, and a zero-point $t$, values are mapped to a $b$-bit integer range, e.g., $[-2^{b-1}, 2^{b-1}-1]$ for signed quantization, and dequantization approximates the original value by $s(x^q - t)$. Existing strategies include *Fixed-Precision Quantization (FPQ)* [14, 47], which uses a uniform bit-width, and *Mixed-Precision Quantization (MPQ)* [16, 43], which assigns sensitivity-aware layer-wise bit-widths.

Consider a network with $L$ layers and a set of available bit-widths $\mathcal{B}$, e.g., $\{2, 4, 8\}$. Each layer $\ell$ is assigned a bit-width $b_\ell \in \mathcal{B}$, giving an allocation $\boldsymbol{b} = \{b_\ell\}_{\ell=1}^{L}$. Let $c_\ell(b_\ell)$ denote the computational cost of layer $\ell$ under bit-width $b_\ell$. We measure cost by *bit operations (BOPs)*, commonly defined as FLOPs $\times B_w \times B_a$, where $B_w$ and $B_a$ are the weight and activation bit-widths [12, 33]. Since conventional FPQ and MPQ assume that all inputs traverse all $L$ layers, their expected complexity depends only on the static allocation $\boldsymbol{b}$, following a deterministic full-depth path.

**Early Exiting.** Early exiting shortens inference by attaching auxiliary classifiers, or exit heads, to intermediate layers, allowing samples to terminate once predictions become sufficiently confident [32]. This induces *stochastic, input-dependent inference depths*: easy samples exit early, while harder ones propagate deeper. Let $\mathcal{E} \subseteq \{1, \dots, L-1\}$ denote the intermediate exit layers. Formally, let $z_\ell(x)$ denote the full-precision logits at layer $\ell$, and define the exit confidence as $\rho_\ell(x) := \max_j \mathrm{softmax}(z_\ell(x))_j$. An input exits at the first exit layer $\ell \in \mathcal{E}$ such that $\rho_\ell(x) \geq \phi_\ell$, where $\phi_\ell \in [0, 1]$ is the exit threshold; otherwise, it continues until layer $L$, which serves as the fallback exit. The exit policy is parameterized by $\boldsymbol{\phi} = \{\phi_e\}_{e \in \mathcal{E}}$.

To quantify stochastic execution under early exiting, we define the utilization indicator $u_\ell(x, \boldsymbol{\phi}) = \mathbf{1}\{x \text{ reaches layer } \ell \mid \boldsymbol{\phi}\}$, and its expectation over the data distribution $\mathcal{D}$, $u_\ell(\boldsymbol{\phi}) = \mathbb{E}_{x \sim \mathcal{D}}[u_\ell(x, \boldsymbol{\phi})]$, which measures the probability that layer $\ell$ is activated. The objective of early exiting is to reduce computation cost while preserving accuracy; however, its exit behavior fundamentally alters how quantization noise accumulates and propagates across layers.

**Amortized-Precision Quantization (APQ).** Conventional quantization assumes a fixed full-depth inference path, where all layers are executed for every input. Under early exiting, however, execution depth becomes stochastic and input-dependent, so quantization noise is exposed and propagated only along realized inference paths. This creates a mismatch for static bit allocation: layers with different execution frequencies may have different effective precision requirements. To address this, we propose *Amortized-Precision Quantization (APQ)*, which accounts for layer-wise execution frequency when allocating precision under early exiting.

**Definition 1** (Amortized-Precision Quantization (APQ))**.** *Consider a network with $L$ layers, where each layer $\ell$ is assigned a bit-width $b_\ell \in \mathcal{B}$ from the available bit-width set $\mathcal{B}$. The amortized computational complexity of a bit allocation $\boldsymbol{b}$ under an early-exit policy $\boldsymbol{\phi}$ is defined as*

$$C_{\mathrm{APQ}}(\boldsymbol{b}, \boldsymbol{\phi}) = \sum_{\ell=1}^{L} \mathbb{E}_{x \sim \mathcal{D}}[u_\ell(x, \boldsymbol{\phi}) \cdot c_\ell(b_\ell)] = \sum_{\ell=1}^{L} u_\ell(\boldsymbol{\phi}) \cdot c_\ell(b_\ell), \tag{1}$$

*where $c_\ell(b_\ell)$ is the cost of layer $\ell$ under bit-width $b_\ell$, $u_\ell(x, \boldsymbol{\phi}) \in \{0, 1\}$ indicates whether layer $\ell$ is executed for input $x$, and $u_\ell(\boldsymbol{\phi}) = \mathbb{E}_{x \sim \mathcal{D}}[u_\ell(x, \boldsymbol{\phi})]$ is its utilization under the exit policy.*

Since $u_\ell(\boldsymbol{\phi})$ depends on exit thresholds, APQ naturally leads to a joint optimization of bit-widths $\boldsymbol{b}$ and exit policy $\boldsymbol{\phi}$ to balance accuracy and amortized low-bit cost:

$$(\boldsymbol{b}^*, \boldsymbol{\phi}^*) = \arg\min_{\boldsymbol{b}, \boldsymbol{\phi}} \ \mathbb{E}_{(x,y) \sim \mathcal{D}}\big[\mathcal{L}\big(f_{\theta, \boldsymbol{b}, \boldsymbol{\phi}}(x), y\big)\big] + \lambda C_{\mathrm{APQ}}(\boldsymbol{b}, \boldsymbol{\phi}), \tag{2}$$

where $\mathcal{L}$ denotes the task-specific loss, $f_{\theta,\boldsymbol{b},\boldsymbol{\phi}}$ denotes the quantized early-exit network, and $\lambda$ controls the accuracy–efficiency trade-off. More broadly, APQ provides a utilization-aware view for dynamic inference paradigms such as recursive Transformers [34], MoE routing [51], and layer skipping [8], where input-dependent computation induces non-uniform exposure of layers or modules to quantization noise.

## 4 Why Static Precision Allocation Fails under Dynamic Inference

This section analyzes the *mutual interference* between quantization and early exiting under stochastic low-bit execution. Because early exiting induces input-dependent stopping depths, quantization affects not only predictions but also executed-depth distributions, while early exiting in turn changes each layer's exposure to quantization noise. This random-stopping view explains why static precision allocation is mismatched with dynamic inference.

**Quantization Perturbs the Exit Policy.** For each exit layer $\ell \in \mathcal{E}$, let $\Delta_\ell(x) = \rho_\ell(x) - \phi_\ell$ denote the full-precision confidence margin relative to the exit threshold $\phi_\ell$. Let $\tilde{\rho}_{\ell,\boldsymbol{b}}(x)$ denote the confidence after quantization under bit allocation $\boldsymbol{b}$, and define the induced confidence perturbation as

$$\xi_{\ell,\boldsymbol{b}}(x) = \tilde{\rho}_{\ell,\boldsymbol{b}}(x) - \rho_\ell(x). \tag{3}$$

We define the full-precision and quantized exit decisions as $e_\ell(x) = \mathbf{1}\{\rho_\ell(x) \geq \phi_\ell\}$ and $\tilde{e}_\ell(x) = \mathbf{1}\{\tilde{\rho}_{\ell,\boldsymbol{b}}(x) \geq \phi_\ell\}$, respectively. The two decisions can differ only when the perturbation crosses the confidence boundary: if $\tilde{e}_\ell(x) \neq e_\ell(x)$, then $|\Delta_\ell(x)| \leq |\xi_{\ell,\boldsymbol{b}}(x)|$. Thus, exit instability depends jointly on the margin distribution around the threshold and the magnitude of the quantization-induced confidence perturbation.

**Theorem 1.** *For an exit layer $\ell \in \mathcal{E}$, assume that the confidence perturbation $\xi_{\ell,\boldsymbol{b}}(x)$ satisfies the tail bound*

$$\Pr(|\xi_{\ell,\boldsymbol{b}}(x)| > \tau_\ell) \leq q_\ell(\tau_\ell, b_\ell),$$

*where $q_\ell(\tau_\ell, b_\ell)$ is non-increasing in $\tau_\ell$ and non-decreasing as $b_\ell$ decreases. Then, for any tolerance $\tau_\ell > 0$, the mis-exit probability satisfies*

$$\Pr(\tilde{e}_\ell(x) \neq e_\ell(x)) \leq \Pr(|\Delta_\ell(x)| \leq \tau_\ell) + q_\ell(\tau_\ell, b_\ell).$$

Theorem 1 decomposes exit instability into a boundary-mass term and a quantization-perturbation tail term. The term $\Pr(|\Delta_\ell(x)| \leq \tau_\ell)$ captures samples near the confidence boundary, where small perturbations may flip the exit decision. The term $q_\ell(\tau_\ell, b_\ell)$ captures the probability that the quantization-induced perturbation exceeds the tolerated margin. As $b_\ell$ decreases, this tail term increases, yielding a larger upper bound on the exit-decision mismatch probability.

Since early exiting stops at the first confident layer, such layer-wise mismatches can shift the realized stopping-depth distribution, leading to premature exits that harm accuracy or delayed exits that reduce efficiency (see Corollary 1 in Appendix A.2).

**Random-Depth Amplifies Quantization Error.** We next show that early exiting turns quantization from a fixed-depth approximation problem into a random-depth distortion problem. Let $h_\ell(x)$ and $\tilde{h}_{\ell,\boldsymbol{b}}(x)$ denote the full-precision and quantized hidden states at layer $\ell$. Suppose that, for $\ell = 1, \ldots, L$, the layer-wise distortion satisfies

$$\|\tilde{h}_{\ell,\boldsymbol{b}}(x) - h_\ell(x)\| \leq \gamma_{\ell-1}\|\tilde{h}_{\ell-1,\boldsymbol{b}}(x) - h_{\ell-1}(x)\| + \varepsilon_\ell(b_\ell),$$

where $\varepsilon_\ell(b_\ell)$ denotes the local distortion introduced at layer $\ell$, $\gamma_{\ell-1}$ captures sensitivity from layer $\ell-1$ to layer $\ell$, and the input distortion is initialized as $\|\tilde{h}_{0,\boldsymbol{b}}(x) - h_0(x)\| = 0$.

**Theorem 2.** *For any realized stopping depth $k$, the accumulated representation error satisfies*

$$\|\tilde{h}_{k,\boldsymbol{b}}(x) - h_k(x)\| \leq \sum_{\ell=1}^{k} \varepsilon_\ell(b_\ell) \prod_{r=\ell+1}^{k} \gamma_{r-1},$$

*where the empty product is defined as 1. Under early exiting with random stopping depth $K_{\boldsymbol{\phi}}(x)$, the expected distortion satisfies*

$$\mathbb{E}_{x\sim\mathcal{D}}\left[\|\tilde{h}_{K_{\boldsymbol{\phi}}(x),\boldsymbol{b}}(x) - h_{K_{\boldsymbol{\phi}}(x)}(x)\|\right] \leq \mathbb{E}_{x\sim\mathcal{D}}\left[\sum_{\ell=1}^{K_{\boldsymbol{\phi}}(x)} \varepsilon_\ell(b_\ell) \prod_{r=\ell+1}^{K_{\boldsymbol{\phi}}(x)} \gamma_{r-1}\right].$$

This theorem shows that quantization error under early exiting is path-dependent: its impact depends on local representation distortion, downstream propagation, and the probability that each layer is reached. Errors in frequently executed or shallow layers can have a larger amortized impact, while rarely executed layers may contribute less.

**Static Precision Allocation Is Mismatched.** Conventional FPQ and MPQ allocate bit-widths according to static layer sensitivity, implicitly assuming that every layer is executed for every input. Under early exiting, however, the contribution of layer $\ell$ is weighted by its utilization probability

$$u_\ell(\boldsymbol{\phi}) = \Pr_{x\sim\mathcal{D}}(K_{\boldsymbol{\phi}}(x) \geq \ell)\,.$$

Let $S_\ell$ denote the static quantization sensitivity of layer $\ell$. Under APQ, the relevant marginal importance becomes the utilization-weighted quantity $u_\ell(\boldsymbol{\phi})S_\ell$, rather than $S_\ell$ alone.

**Theorem 3.** *Assume that static MPQ ranks layers by their static sensitivities $S_\ell$, while APQ ranks layers by their utilization-weighted sensitivities $u_\ell(\boldsymbol{\phi})S_\ell$. If there exist two layers $i$ and $j$ such that*

$$S_i > S_j \quad \textit{but} \quad u_i(\boldsymbol{\phi})S_i < u_j(\boldsymbol{\phi})S_j,$$

*then the static MPQ ranking is inconsistent with the APQ ranking. Moreover, under equal marginal bit-cost increments and marginal APQ benefit proportional to $u_\ell(\boldsymbol{\phi})S_\ell$, the static MPQ allocation can be suboptimal for the APQ objective.*

Theorem 3 shows that static quantization can be suboptimal even with accurate full-depth sensitivity estimates, because it ignores layer utilization under early exiting. This motivates incorporating layer utilization into risk estimation and cost evaluation when jointly optimizing precision and exit thresholds.

**Takeaway.** The above results yield two main observations. First, Theorem 1 shows that quantization can flip exit decisions near the boundary, motivating the decision-level boundary sensitivity risk (BSR). Second, Theorem 2 shows that quantization error is path-dependent under early exiting, motivating the representation-level quantization-induced drift (QID) and utilization-aware bit-width allocation. Together, these results explain the objective mismatch in Theorem 3: static MPQ ignores utilization, whereas APQ calls for a framework that jointly optimizes bit-width allocation and exit-threshold adaptation.

## 5 MAQEE

To optimize APQ effectively, we need to balance dynamic execution depth and layer-wise precision under quantization. Exit thresholds determine which layers are executed, while bit-width allocation controls how quantization error affects exit decisions and downstream representations. This coupling naturally gives rise to **Mutual Adaptive Quantization with Early Exiting (MAQEE)**, a bi-level optimization framework that jointly optimizes exit thresholds and layer-wise bit-width allocation. Detailed pseudocode is given in Appendix C.

### 5.1 Bi-Level Optimization with APQ

Since APQ aims to reduce expected amortized complexity while preserving prediction accuracy, the problem is naturally formulated as a **bi-level optimization** over $(\boldsymbol{b}, \boldsymbol{\phi})$. Recall that $u_\ell(\boldsymbol{\phi})$ denotes the probability that inference reaches layer $\ell$. We denote the expected task loss under APQ as follows.

$$\mathcal{L}(\boldsymbol{b}, \boldsymbol{\phi}) := \mathbb{E}_{(x,y)\sim\mathcal{D}}\Big[\mathcal{L}\big(f_{\theta,\boldsymbol{b},\boldsymbol{\phi}}(x), y\big)\Big]. \tag{4}$$

**Outer-loop (Exit-threshold optimization).** Let $\boldsymbol{\phi} = \{\phi_e\}_{e\in\mathcal{E}}$ denote the confidence thresholds of the exit heads. The outer problem optimizes these thresholds to balance prediction accuracy and amortized computational cost:

$$\boldsymbol{\phi}^* = \arg\min_{\boldsymbol{\phi}} \mathcal{L}\big(\boldsymbol{b}^*(\boldsymbol{\phi}), \boldsymbol{\phi}\big) + \lambda_{\text{outer}}\, C_{\text{APQ}}(\boldsymbol{b}^*(\boldsymbol{\phi}), \boldsymbol{\phi}), \tag{5}$$

where $\lambda_{\text{outer}}$ controls the trade-off between accuracy and amortized computational cost, and $\boldsymbol{b}^*(\boldsymbol{\phi})$ denotes the optimal bit-width allocation induced by the exit policy $\boldsymbol{\phi}$ through the inner loop.

**Inner-loop (Bit-width allocation).** For fixed exit thresholds $\boldsymbol{\phi}$, the bit-width allocation $\boldsymbol{b} = \{b_\ell\}_{\ell=1}^{L}$ is optimized to minimize amortized complexity while preserving accuracy:

$$\boldsymbol{b}^*(\boldsymbol{\phi}) = \arg\min_{\boldsymbol{b}} \mathcal{L}(\boldsymbol{b}, \boldsymbol{\phi}) + \lambda_{\text{inner}}\, C_{\text{APQ}}(\boldsymbol{b}, \boldsymbol{\phi}), \tag{6}$$

where $\lambda_{\text{inner}}$ controls the strength of compression.

This bi-level formulation explicitly couples quantization with early exiting. The exit policy $\boldsymbol{\phi}$ determines layer utilization $u_\ell(\boldsymbol{\phi})$, while the bit allocation $\boldsymbol{b}$ affects confidence estimates and may alter realized exits. This coupling reflects the objective mismatch in Theorem 3: static FPQ and MPQ ignore stochastic utilization and cannot fully capture the APQ objective. By jointly optimizing $\boldsymbol{\phi}$ and $\boldsymbol{b}$, APQ balances accuracy and efficiency under dynamic inference.

Since directly solving this bi-level problem is difficult due to discrete exit decisions and quantization-induced representation shifts, we introduce risk measures to guide exit-threshold adaptation and bit-width reallocation.[3] We then present the iterative optimization procedure with lightweight self-distillation for accuracy recovery. The empirical solver uses an accuracy constraint for threshold selection and a fixed bit budget for allocation; Appendix B.9 gives the implementation details.

### 5.2 Modeling Risk for Early Exiting and Quantization

As shown in Section 4, APQ instability arises from both decision uncertainty and representation distortion. We therefore introduce four complementary risk scores to capture these effects. We write $\tilde{\rho}_\ell(x) = \tilde{\rho}_{\ell,\boldsymbol{b}}(x)$ for the confidence under the current bit allocation.

**Boundary Sensitivity Risk (BSR).** To characterize the instability of exit decisions under quantization noise, we introduce Boundary Sensitivity Risk at each exit layer $\ell \in \mathcal{E}$:

$$\text{BSR}_\ell = \mathbb{E}_{x\sim\mathcal{D}}\Big[u_\ell(x, \boldsymbol{\phi})\, \mathbf{1}\big\{|\tilde{\rho}_\ell(x) - \phi_\ell| < \tau_\ell\big\}\Big], \tag{7}$$

where $\tau_\ell$ is a tolerance margin. BSR measures the fraction of samples that reach the exit and lie near the exit boundary. This measure is motivated by the boundary-mass term in Theorem 1. A high BSR indicates that small quantization perturbations can easily flip exit outcomes.

**Quantization-Induced Drift (QID).** Let $h_\ell(x)$ and $\tilde{h}_\ell(x)$ denote the full-precision and quantized activations at layer $\ell$, respectively. To capture changes in residual-update magnitude under quantization, we introduce Quantization-Induced Drift:

$$\text{QID}_\ell = \left[\log \frac{\tilde{\sigma}_\ell + \epsilon}{\sigma_\ell + \epsilon}\right]^2, \tag{8}$$

where $\sigma_\ell$ and $\tilde{\sigma}_\ell$ denote the magnitudes of the full-precision and quantized residual updates, respectively. Larger QID indicates a greater relative change in residual-update magnitude.

Beyond the two APQ-specific risks above, we further incorporate two standard scores from early exiting and quantization to capture task-level degradation and hidden-state distortion.

**Performance Gap Risk (PGR).** We quantify early-exit degradation via Performance Gap Risk [15]. For each exit head $\ell$, it is defined as

$$\text{PGR}_\ell = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathcal{L}(\hat{y}_\ell(x), y) - \mathcal{L}(\hat{y}_L(x), y)\right], \tag{9}$$

where $\mathcal{L}$ is the task-specific objective, $\hat{y}_\ell(x)$ is the prediction from the exit head at layer $\ell$, and $\hat{y}_L(x)$ is the full-depth prediction. PGR measures the excess loss incurred when exiting at layer $\ell$ instead of using the full-depth output.

**Inverse Signal-to-Quantization-Noise Ratio (SQNR$^{-1}$).** As a supporting representation-level risk, we also use the inverse signal-to-quantization-noise ratio [36], defined as

$$\text{SQNR}_\ell^{-1} = \mathbb{E}_{x\sim\mathcal{D}}\left[\frac{\|h_\ell(x) - \tilde{h}_\ell(x)\|^2}{\|h_\ell(x)\|^2}\right]. \tag{10}$$

It quantifies the relative power of quantization noise and complements QID by measuring pointwise representation distortion.

[3] The NP-hardness of APQ is proved in Theorem 4, with details in Appendix A.5.

**Overall Risk Metric.** We combine the above risks into a per-layer instability score:

$$R_\ell = \sum_{j \in \{\mathrm{PGR},\mathrm{BSR},\mathrm{QID},\mathrm{SQNR}\}} w_j \mathcal{N}(r_j)_\ell, \tag{11}$$

Here $r_j = (r_{j,1}, \ldots, r_{j,L})$ collects risk component $j$ across layers, $w_j$ is its weight, and $\mathcal{N}$ normalizes it across layers. PGR and BSR capture decision-level risk, while QID and $\mathrm{SQNR}^{-1}$ capture representation-level risk. Appendix B.9 details the risk estimators, weights, and mapping from exits to blocks.

### 5.3 Iterative Optimization for APQ

Building on the proposed risk formulation, we develop a unified iterative optimization framework for APQ. Each iteration alternates among three stages: (i) applying a recovery step to the model parameters $\theta$; (ii) optimizing the exit thresholds $\boldsymbol{\phi}$ under a fixed bit allocation $\boldsymbol{b}$; and (iii) reallocating layer-wise bit-widths according to the normalized risks.

**Exit-Threshold Optimization (Outer-loop).** Under a fixed bit allocation $\boldsymbol{b}$, we rank exit policies that meet the accuracy target by computational cost and exit risk:

$$\boldsymbol{\phi}^\star \in \underset{\boldsymbol{\phi}:\, \widehat{\mathrm{Acc}}(\boldsymbol{b},\boldsymbol{\phi}) \geq A_{\mathrm{target}}}{\overset{\mathrm{lex}}{\arg\min}} \Big( C_{\mathrm{APQ}}(\boldsymbol{b}, \boldsymbol{\phi}),\ R_{\mathrm{exit}}(\boldsymbol{b}, \boldsymbol{\phi}) \Big). \tag{12}$$

Here $R_{\mathrm{exit}}$ is the ordered pair of boundary sensitivity and exit prediction loss. We compare policies first by cost, then by boundary sensitivity, and finally by prediction loss. We search for thresholds via grid search with local refinement. Confidence scores $\tilde{\rho}_\ell(x)$ are obtained using one forward pass over the held-out calibration set, after which candidate thresholds can be efficiently evaluated without retraining [15].

**Bit-width Allocation (Inner-loop).** Given optimized early-exit thresholds $\boldsymbol{\phi}$, we allocate per-layer bit-widths $\boldsymbol{b}$ under a total-bit budget $B$ by evaluating the estimated risk reduction $\Delta R_\ell(b_\ell)$ from assigning an additional bit to layer $\ell$. Under the current exit distribution, the corresponding increase in amortized computational cost is

$$\Delta\mathrm{BOPs}_\ell(\boldsymbol{b}, \boldsymbol{\phi}) = C_{\mathrm{APQ}}(\boldsymbol{b}^{+\ell}, \boldsymbol{\phi}) - C_{\mathrm{APQ}}(\boldsymbol{b}, \boldsymbol{\phi}),$$

where $\boldsymbol{b}^{+\ell}$ increases only $b_\ell$ by one bit. Let $U_\ell(b_\ell)$ denote the accumulated estimated risk reduction from increasing layer $\ell$ from the minimum bit-width to $b_\ell$. We select

$$\boldsymbol{b}^\star \in \arg\max_{\boldsymbol{b} \in \mathcal{B}^L} \sum_{\ell=1}^{L} U_\ell(b_\ell) \quad \text{s.t.} \quad \sum_{\ell=1}^{L} b_\ell = B, \tag{13}$$

where $B = L\bar{b}$ for average weight precision $\bar{b}$. Dynamic programming accumulates estimated risk reduction and BOPs across layers, preferring lower BOPs when the estimated reductions are equal. Appendix B.9 gives the risk-reduction estimates and solver details.

**Self-distillation Recovery.** We use self-distillation to recover performance under low precision and dynamic execution. The teacher parameters $\theta_{\mathrm{T}}$ remain fixed. The recovery objective jointly enforces task supervision and distributional alignment:

$$\mathcal{L}_{\mathrm{rec}}(\theta) = \sum_{e \in \mathcal{E} \cup \{L\}} \omega_e \big[ (1 - \eta_e) \mathcal{L}_{\mathrm{CE},e}(\theta) + \eta_e \mathcal{L}_{\mathrm{KL},e}(\theta; \theta_{\mathrm{T}}) \big], \tag{14}$$

where $\mathcal{E}$ is the set of intermediate exits, $\omega_e$ weights each classifier, and $\eta_e$ balances task supervision and teacher-to-student distillation. By aligning the quantized early-exit student with its full-precision teacher, this step compensates for quantization-induced representation drift and stabilizes exit decisions.

**Warm-Started Iteration.** To accelerate iterative optimization, we initialize exit-threshold search from the previous solution.

Finally, MAQEE alternates between threshold updates and bit-allocation updates to balance accuracy and amortized computation. Threshold updates determine which layers are reached, while the resulting risks guide bit allocation. In contrast, static FPQ and MPQ ignore stochastic layer utilization and can therefore be suboptimal under early exiting. The search stops when an allocation repeats or the round budget is exhausted. We select the evaluated model with the lowest BOPs among those meeting the target accuracy.

# 6 Experiments

In the following, we evaluate MAQEE on three vision tasks and three backbones, comparing it with early-exit methods, quantization baselines, joint methods, and simple precision-allocation strategies.

## 6.1 Experimental Setup

**Datasets.** We evaluate MAQEE on image classification (CIFAR-100 [18], ImageNet [5]), semantic segmentation (SceneParse150 [50]), and object detection (MS COCO [22]), reporting classification accuracy, mean intersection over union (mIoU), and mAP@0.5, respectively. Experiments are conducted on three backbones: DeiT [38], ViT [7], and Swin [24]. For classification and segmentation, we use standard cross-entropy loss; for object detection, we use the bipartite matching loss [2].

**Baselines.** We compare MAQEE against representative early-exiting baselines, including ViT-EE [1] and LGViT [46], as well as state-of-the-art quantization methods for ViTs. Specifically, we include fixed-precision quantization (FPQ) methods: RepQ [21] and ERQ [49], and a mixed-precision quantization (MPQ) method: MPTQ [36]. FP4/4 uses 4-bit weights in every layer, whereas MP4/4 varies weight precision across layers with an average of 4 bits. Both use 4-bit activations. We also adapt McQueen and QuEE to CIFAR-100/ViT-B. McQueen uses an equivalent 4-bit budget; QuEE uses W3–W5 candidate paths. Appendix B.3 provides the additional protocols.

**Evaluation Protocol.** For a fair comparison [33], we evaluate the accuracy–efficiency trade-off under two complementary settings. *(i) Controlled Performance.* We measure the exit layer $\hat{L}$ and bit operations $\hat{\mathrm{BOPs}}$ required to reach a predefined target performance, set to the quantization-only baseline without early exiting: 87% for CIFAR-100, 79% for ImageNet, 30% for SceneParse150, and 55% for MS COCO. Models that fail to meet the target are reported with $\hat{L}$ and $\hat{\mathrm{BOPs}}$ as N/A. *(ii) Standard Configuration.* Baseline models are evaluated using their default thresholds and quantization settings. Under Standard Configuration, we report the average exit layer $\bar{L}$, the average bit operations $\bar{\mathrm{BOPs}}$, and the task performance. Bold entries highlight MAQEE. Dashes indicate quantities that are not applicable or not reported; N/A indicates that the target performance was not reached. All results are averaged over 10 runs, with standard deviations less than 3%. Additional details and hyperparameter settings are provided in Appendix B.1.

## 6.2 Quantitative Results

**Accuracy and Efficiency.** Table 1 reports classification results for DeiT-B and ViT-L under Controlled Performance and Standard Configuration. Results for Swin and ViT-B are given in Table 6 in the appendix. Across all backbones, MAQEE reduces BOPs by over 95%. In contrast, fixed-precision baselines such as RepQ and ERQ are less stable. RepQ can suffer severe accuracy degradation because uniform quantization amplifies representation-level errors caused by layer sensitivity and quantization noise (Theorem 2). Although MPTQ uses mixed precision, it assumes a static inference path and ignores input-dependent layer utility under early exiting, which can lead to unnecessarily deep exits or suppressed early-exit behavior (Theorem 3).

**Segmentation and Detection.**

To assess cross-task generalization, Table 2 evaluates MAQEE on semantic segmentation and object detection using ViT-B and DeiT-B backbones, respectively, with ViT-EE as the exiting head. Under Standard Configuration, MAQEE reduces segmentation BOPs by 9.49% and improves mIoU by 4.68 percentage points over ViT-EE+MPTQ. On detection, it improves mAP@0.5 by 9.79 percentage points at comparable BOPs. In object detection, however, early exiting with quantization still causes a noticeable accuracy drop, as detection requires multi-scale features that can be truncated by early exits and distorted by quantization.

**Joint Methods.** Against joint quantization–early-exit methods (Table 3), MAQEE reaches 88.51% on CIFAR-100/ViT-B at $\bar{\mathrm{BOPs}} = 2.57\mathrm{E}11$, compared with 72.44% at $\bar{\mathrm{BOPs}} = 3.24\mathrm{E}11$ for QuEE and 54.63% for McQueen under an equivalent 4-bit budget. Relative to QuEE, MAQEE gains 16.07 percentage points while reducing BOPs by 20.7%. It also exceeds the 87% performance target, which neither adaptation reaches. Adaptation details are given in Appendix B.3.

Table 1: Classification results on DeiT-B and ViT-L under Controlled Performance and Standard Configuration.

| | Bits (W/A) | Quant | EE | CIFAR-100 | | | | | ImageNet | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Controlled Perf. | | Std. Config | | | Controlled Perf. | | Std. Config | | |
| | | | | $\hat{L}\downarrow$ | BÔPs $\downarrow$ | $\bar{L}\downarrow$ | BŌPs $\downarrow$ | Acc. (%)$\uparrow$ | $\hat{L}\downarrow$ | BÔPs $\downarrow$ | $\bar{L}\downarrow$ | BŌPs $\downarrow$ | Acc. (%)$\uparrow$ |
| DeiT-B | FP32 | – | – | – | – | *12.00* | *1.80E13* | *91.86* | – | – | *12.00* | *1.80E13* | *83.10* |
| | | | ViT-EE | 7.20 | 1.84E13 | 8.10 | 1.97E13 | 89.24 | 7.89 | 1.76E13 | 8.15 | 1.97E13 | 80.51 |
| | | | LGViT | 5.47 | 1.61E13 | 6.33 | 1.70E13 | 89.03 | 6.19 | 1.72E13 | 7.18 | 1.87E13 | 81.70 |
| | FP4/4 | RepQ | ViT-EE | N/A | N/A | 9.91 | 3.59E11 | 84.79 | N/A | N/A | 11.42 | 3.85E11 | 70.58 |
| | | | LGViT | N/A | N/A | 9.37 | 3.42E11 | 75.43 | N/A | N/A | 10.03 | 3.56E11 | 69.31 |
| | | ERQ | ViT-EE | 8.77 | 3.23E11 | 9.28 | 3.35E11 | 87.86 | 11.79 | 3.91E11 | 10.86 | 3.72E11 | 77.59 |
| | | | LGViT | N/A | N/A | 9.00 | 3.34E11 | 78.51 | N/A | N/A | 9.09 | 3.37E11 | 71.32 |
| | MP4/4 | MPTQ | ViT-EE | 8.81 | 3.25E11 | 8.51 | 3.17E11 | 84.57 | 11.80 | 3.94E11 | 10.56 | 3.65E11 | 76.15 |
| | | | LGViT | 11.35 | 3.93E11 | 7.76 | 3.07E11 | 83.15 | 12.00 | 4.09E11 | 8.86 | 3.34E11 | 73.58 |
| | **MAQEE** | | | **5.86** | **2.61E11** | **5.98** | **2.64E11** | **87.86** | **7.92** | **3.10E11** | **7.59** | **3.01E11** | **78.57** |
| ViT-L | FP32 | – | – | – | – | *24.00* | *6.31E13* | *93.25* | – | – | *24.00* | *6.31E13* | *85.30* |
| | | | ViT-EE | 11.75 | 3.11E13 | 12.94 | 3.41E13 | 92.40 | 12.43 | 3.36E13 | 16.01 | 4.37E13 | 82.86 |
| | | | LGViT | 8.34 | 2.26E13 | 10.85 | 2.93E13 | 92.53 | 9.95 | 2.69E13 | 12.52 | 3.38E13 | 82.02 |
| | FP4/4 | RepQ | ViT-EE | N/A | N/A | 23.11 | 1.12E12 | 81.31 | N/A | N/A | 19.95 | 9.81E11 | 76.62 |
| | | | LGViT | 23.94 | 1.14E12 | 20.69 | 1.01E12 | 85.97 | N/A | N/A | 15.76 | 8.18E11 | 74.02 |
| | | ERQ | ViT-EE | N/A | N/A | 22.58 | 1.09E12 | 82.34 | N/A | N/A | 21.53 | 1.05E12 | 58.89 |
| | | | LGViT | 21.06 | 1.02E12 | 20.74 | 1.01E12 | 85.59 | 22.20 | 1.08E12 | 17.53 | 8.95E11 | 76.52 |
| | MP4/4 | MPTQ | ViT-EE | N/A | N/A | 21.96 | 1.06E12 | 83.31 | N/A | N/A | 21.33 | 1.04E12 | 75.44 |
| | | | LGViT | 20.28 | 9.92E11 | 21.64 | 1.05E12 | 84.33 | 21.32 | 1.01E12 | 17.10 | 8.53E11 | 75.12 |
| | **MAQEE** | | | **9.66** | **5.64E11** | **9.71** | **5.79E11** | **88.17** | **12.39** | **6.32E11** | **18.64** | **9.15E11** | **80.29** |

Table 2: Performance on semantic segmentation (ViT-B) and object detection (DeiT-B).

| Method | Segmentation (SceneParse150) | | | | | Detection (MS COCO) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Controlled Perf. | | Std. Config | | | Controlled Perf. | | Std. Config | | |
| | $\hat{L}\downarrow$ | BÔPs$\downarrow$ | $\bar{L}\downarrow$ | BŌPs$\downarrow$ | mIoU (%)$\uparrow$ | $\hat{L}\downarrow$ | BÔPs$\downarrow$ | $\bar{L}\downarrow$ | BŌPs$\downarrow$ | mAP@0.5 (%)$\uparrow$ |
| *FP32* | – | – | 12.00 | 1.89E13 | 34.14 | – | – | 12.00 | 1.95E13 | 68.23 |
| ViT-EE+ERQ | N/A | N/A | 8.10 | 3.17E11 | 27.34 | N/A | N/A | 8.72 | 3.34E11 | 40.42 |
| ViT-EE+MPTQ | 11.46 | 3.94E11 | 8.08 | 3.16E11 | 26.52 | N/A | N/A | 8.59 | 3.30E11 | 41.04 |
| **MAQEE** | **5.74** | **2.60E11** | **6.79** | **2.86E11** | **31.20** | **10.40** | **3.77E11** | **8.68** | **3.32E11** | **50.83** |

**Pareto Frontier.** Figure 2 shows that MAQEE forms the Pareto frontier, achieving higher accuracy at comparable cost or lower cost at comparable accuracy under both throughput and BOPs comparisons. This advantage comes from jointly optimizing bit-width allocation and exit thresholds under the APQ objective, rather than treating quantization and early exiting separately. By coupling decision-level risks (PGR and BSR) with representation-level risks (SQNR$^{-1}$ and QID), MAQEE stabilizes exit behavior and reduces exit depth and BOPs by up to 50% relative to strong baselines, consistent with the utilization-aware principle in Theorem 3.

### 6.3 Hardware Efficiency

Table 4 reports ViT-B deployment measurements on an RTX 5090 at batch size 1. Compared with full-depth FP32 inference, MAQEE reduces latency from 1.608 to 0.711 ms on ImageNet and from 1.643 to 0.537 ms on CIFAR-100, yielding throughput gains of 2.26$\times$ and 3.06$\times$, respectively. Peak allocated memory falls by 64.9% and 63.9%. CIFAR-100 has a shallower average exit depth in these deployments (5.98 versus 6.65 on ImageNet), consistent with its larger runtime gain. The deployment combines packed low-bit kernels with early exiting, so the measured gains reflect both precision reduction and skipped computation. BOPs quantifies precision-weighted arithmetic, whereas GPU runtime also includes memory access, kernel execution, and exit-control overhead. Appendix B.5 provides the timing and deployment protocol.

### 6.4 Ablation Studies

Table 5 shows that the full model achieves the best controlled-performance trade-off, with stable early exits and reduced cost. Without PGR, the allocation score omits exit prediction loss, and the model fails to reach the target accuracy. Removing BSR impairs exit stability, as quantization per-

Table 3: Joint methods on CIFAR-100/ViT-B.

| Method | Top-1 (%) | BŌPs↓ | Bits |
|---|---|---|---|
| *FP32 (full)* | *90.47* | – | *32.00* |
| McQueen | 54.63 | – | 4.00 |
| QuEE | 72.44 | 3.24E11 | 4.80 |
| **MAQEE** | **88.51** | **2.57E11** | **4.00** |

Table 4: GPU efficiency on ViT-B.

| Dataset | Model | ms | img/s | Speedup | Peak MiB |
|---|---|---|---|---|---|
| ImageNet | FP32 | 1.608 | 621.9 | 1.00× | 350.10 |
| | **MAQEE** | **0.711** | **1406.3** | 2.26× | **122.81** |
| CIFAR-100 | FP32 | 1.643 | 608.5 | 1.00× | 347.56 |
| | **MAQEE** | **0.537** | **1861.6** | 3.06× | **125.47** |

Table 5: Ablation study on ImageNet/ViT-B.

| Method | Controlled Perf. | | Std. Config | | |
|---|---|---|---|---|---|
| | $\hat{L}\downarrow$ | BŌPs↓ | $\bar{L}\downarrow$ | BŌPs↓ | Acc. (%)↑ |
| **MAQEE** | **7.04** | **2.85E11** | **7.26** | **2.93E11** | **79.41** |
| w/o PGR | N/A | N/A | 4.23 | 2.12E11 | 61.18 |
| w/o BSR | 7.59 | 3.12E11 | 7.01 | 2.74E11 | 76.53 |
| w/o SQNR$^{-1}$ | 7.35 | 2.98E11 | 7.28 | 2.95E11 | 78.35 |
| w/o QID | 7.17 | 2.94E11 | 7.18 | 2.95E11 | 79.06 |

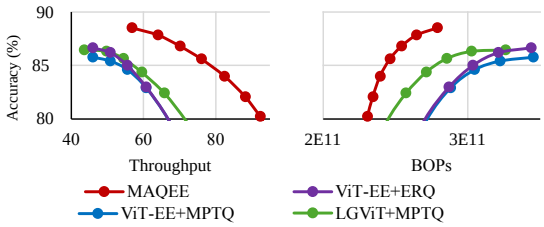



Figure 2: Accuracy versus throughput and BOPs.

turbations more easily flip exit decisions; satisfying the same accuracy constraint therefore requires more conservative execution, increasing effective depth and BOPs. Removing SQNR$^{-1}$ eliminates relative noise-energy guidance, making mixed-precision allocation less reliable and degrading the efficiency–accuracy trade-off. Without QID, allocation omits changes in residual-update magnitude. BOPs increase even when average exit depth changes only marginally.

**Representation drift.** To examine how block quantization affects representations, we lower one block at a time from W4 to W3 and measure changes in its residual update and in the CLS token. Residual-update changes correlate with local and final-layer CLS drift (Spearman $\rho = 0.779$ and $0.646$), linking local quantization sensitivity to downstream drift. Appendix B.4 gives the protocol and exit-decision analysis.

## 7 Limitations

MAQEE estimates layer utilization and quantization risk on a calibration distribution. Changes in input difficulty can alter both the exit distribution and the precision needed along the executed paths. Thresholds selected on the source distribution provide different operating points, but choosing thresholds for an unknown distribution shift remains an open problem. Performance on segmentation and detection depends on the quality of intermediate features, which can be affected by both early exits and quantization. Hardware gains also depend on low-bit kernel support and the overhead of dynamic execution. Extending APQ to autoregressive models requires estimating layer utilization for each token and accounting for states reused across generated tokens.

## 8 Conclusion

In this work, we propose **MAQEE**, a bi-level optimization framework for efficient ViT inference under the **APQ** paradigm. APQ accounts for the expected utilization of each layer, while MAQEE combines this utilization with representation and exit-decision risks to guide precision allocation. Experiments across classification, segmentation, and detection show improved accuracy–computation trade-offs by jointly optimizing precision and exit thresholds. Measurements on an RTX 5090 show GPU throughput gains of 2.26× on ImageNet and 3.06× on CIFAR-100. Future work will address adaptation under distribution shifts and extend APQ to broader dynamic inference schemes, such as recursive Transformers, MoE routing, and layer skipping.

## Acknowledgments

This work was supported by the National Science and Technology Council, Taiwan, under Grant NSTC 115-2223-E-002-001; and by the Ministry of Education, Taiwan, through the Higher Education Sprout Project—the Featured Areas Research Center Program, under Grant 115L9009.

## References

[1] A. Bakhtiarnia, Q. Zhang, and A. Iosifidis. Multi-exit vision transformer for dynamic inference. *arXiv preprint arXiv:2106.15183*, 2021.

[2] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020.

[3] C.-F. R. Chen, Q. Fan, and R. Panda. CrossViT: Cross-attention multi-scale vision transformer for image classification. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 357–366, 2021.

[4] M. Courbariaux, Y. Bengio, and J.-P. David. BinaryConnect: Training deep neural networks with binary weights during propagations. *Advances in neural information processing systems*, 28, 2015.

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. IEEE, 2009.

[6] Y. Ding, H. Qin, Q. Yan, Z. Chai, J. Liu, X. Wei, and X. Liu. Towards accurate post-training quantization for vision transformer. In *Proceedings of the 30th ACM international conference on multimedia*, pages 5380–5388, 2022.

[7] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[8] M. Elhoushi, A. Shrivastava, D. Liskovich, B. Hosmer, B. Wasti, L. Lai, A. Mahmoud, B. Acun, S. Agarwal, A. Roman, et al. LayerSkip: Enabling early exit inference and self-speculative decoding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12622–12642, 2024.

[9] Y. Fang, B. Liao, X. Wang, J. Fang, J. Qi, R. Wu, J. Niu, and W. Liu. You only look at one sequence: Rethinking transformer in vision through object detection. *Advances in Neural Information Processing Systems*, 34:26183–26197, 2021.

[10] Y. Gao, M. Zhou, and D. N. Metaxas. UTNet: a hybrid transformer architecture for medical image segmentation. In *International conference on medical image computing and computer-assisted intervention*, pages 61–71. Springer, 2021.

[11] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, and K. Keutzer. A survey of quantization methods for efficient neural network inference. In *Low-power computer vision*, pages 291–326. Chapman and Hall/CRC, 2022.

[12] X. Hu, Z. Chen, D. Yang, Z. Xu, C. Xu, Z. Yuan, S. Zhou, and J. Yu. MoEQuant: Enhancing quantization for mixture-of-experts large language models via expert-balanced sampling and affinity guidance. *arXiv preprint arXiv:2505.03804*, 2025.

[13] C. Hwang, W. Cui, Y. Xiong, Z. Yang, Z. Liu, H. Hu, Z. Wang, R. Salas, J. Jose, P. Ram, et al. Tutel: Adaptive mixture-of-experts at scale. *Proceedings of Machine Learning and Systems*, 5:269–287, 2023.

[14] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. 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.

[15] M. Jazbec, A. Timans, T. H. Veljković, K. Sakmann, D. Zhang, C. A. Naesseth, and E. Nalisnick. Fast yet safe: Early-exiting with risk control. *Advances in Neural Information Processing Systems*, 37:129825–129854, 2024.

[16] J.-Y. Jeon, X. T. Nguyen, S. Ryu, and H.-J. Lee. USDN: A unified sample-wise dynamic network with mixed-precision and early-exit. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 646–654, 2024.

[17] R. Krishnamoorthi. Quantizing deep convolutional networks for efficient inference: A whitepaper. *arXiv preprint arXiv:1806.08342*, 2018.

[18] A. Krizhevsky et al. Learning multiple layers of features from tiny images. 2009.

[19] S. Laskaridis, A. Kouris, and N. D. Lane. Adaptive inference through early-exit networks: Design, challenges and directions. In *Proceedings of the 5th International Workshop on Embedded and Mobile Deep Learning*, pages 1–6, 2021.

[20] Y. Li, S. Xu, B. Zhang, X. Cao, P. Gao, and G. Guo. Q-ViT: Accurate and fully quantized low-bit vision transformer. *Advances in neural information processing systems*, 35:34451–34463, 2022.

[21] Z. Li, J. Xiao, L. Yang, and Q. Gu. RepQ-ViT: Scale reparameterization for post-training quantization of vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17227–17236, 2023.

[22] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[23] H. Liu, S. Elkerdawy, N. Ray, and M. Elhoushi. Layer importance estimation with imprinting for neural network quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2408–2417, 2021.

[24] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[25] Z. Liu, Y. Wang, K. Han, W. Zhang, S. Ma, and W. Gao. Post-training quantization for vision transformer. *Advances in Neural Information Processing Systems*, 34:28092–28103, 2021.

[26] M. Nagel, M. Fournarakis, Y. Bondarenko, and T. Blankevoort. Overcoming oscillations in quantization-aware training. In *International Conference on Machine Learning*, pages 16318–16330. PMLR, 2022.

[27] M. Raghu, T. Unterthiner, S. Kornblith, C. Zhang, and A. Dosovitskiy. Do vision transformers see like convolutional neural networks? *Advances in neural information processing systems*, 34:12116–12128, 2021.

[28] H. Rahmath P, V. Srivastava, K. Chaurasia, R. G. Pacheco, and R. S. Couto. Early-exit deep neural network-a comprehensive survey. *ACM Computing Surveys*, 57(3):1–37, 2024.

[29] F. Regol, J. Chataoui, B. Charpentier, M. Coates, P. Piantanida, and S. Gunnemann. Predicting probabilities of error to combine quantization and early exiting: QuEE. *arXiv preprint arXiv:2406.14404*, 2024.

[30] C. Riquelme, J. Puigcerver, B. Mustafa, M. Neumann, R. Jenatton, A. Susano Pinto, D. Keysers, and N. Houlsby. Scaling vision with sparse mixture of experts. *Advances in Neural Information Processing Systems*, 34:8583–8595, 2021.

[31] U. Saxena and K. Roy. McQueen: Mixed precision quantization of early exit networks. In *BMVC*, pages 511–513, 2023.

[32] T. Schuster, A. Fisch, J. Gupta, M. Dehghani, D. Bahri, V. Tran, Y. Tay, and D. Metzler. Confident adaptive language modeling. *Advances in Neural Information Processing Systems*, 35:17456–17472, 2022.

[33] Y. Shang, G. Liu, R. Kompella, and Y. Yan. Quantized-ViT efficient training via Fisher matrix regularization. In *International Conference on Multimedia Modeling*, pages 270–284. Springer, 2024.

[34] Z. Shen, Z. Liu, and E. Xing. Sliced recursive transformer. In *European Conference on Computer Vision*, pages 727–744. Springer, 2022.

[35] H. Shi, X. Cheng, W. Mao, and Z. Wang. P2-ViT: Power-of-two post-training quantization and acceleration for fully quantized vision transformer. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2024.

[36] Y.-S. Tai and A.-Y. Wu. MPTQ-ViT: Mixed-precision post-training quantization for vision transformer. *arXiv preprint arXiv:2401.14895*, 2024.

[37] S. Teerapittayanon, B. McDanel, and H.-T. Kung. BranchyNet: Fast inference via early exiting from deep neural networks. In *2016 23rd international conference on pattern recognition (ICPR)*, pages 2464–2469. IEEE, 2016.

[38] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021.

[39] F. Valade, M. Hebiri, and P. Gay. EERO: Early exit with reject option for efficient classification with limited budget. *arXiv preprint arXiv:2402.03779*, 2024.

[40] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[41] L. Wang, C. Zeng, M. Zhang, J. Wu, and L. Nie. Domain aware post training quantization for vision transformers in deployment. *Pattern Recognition*, page 112182, 2025.

[42] X. Wei, R. Gong, Y. Li, X. Liu, and F. Yu. QDrop: Randomly dropping quantization for extremely low-bit post-training quantization. In *International Conference on Learning Representations*, 2022.

[43] J. Xiao, Z. Li, L. Yang, and Q. Gu. Patch-wise mixed-precision quantization of vision transformer. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE, 2023.

[44] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo. SegFormer: Simple and efficient design for semantic segmentation with transformers. *Advances in neural information processing systems*, 34:12077–12090, 2021.

[45] J. Xin, R. Tang, J. Lee, Y. Yu, and J. Lin. DeeBERT: Dynamic early exiting for accelerating BERT inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2246–2251, 2020.

[46] G. Xu, J. Hao, L. Shen, H. Hu, Y. Luo, H. Lin, and J. Shen. LGViT: Dynamic early exiting for accelerating vision transformer. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 9103–9114, 2023.

[47] J. Yang, X. Shen, J. Xing, X. Tian, H. Li, B. Deng, J. Huang, and X.-s. Hua. Quantization networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7308–7316, 2019.

[48] J. Zheng, L. Yang, Y. Li, K. Yang, Z. Wang, and J. Zhou. Lightweight vision transformer with spatial and channel enhanced self-attention. In *2023 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 1484–1488. IEEE, 2023.

[49] Y. Zhong, Y. Huang, J. Hu, Y. Zhang, and R. Ji. Towards accurate post-training quantization of vision transformers via error reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

[50] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba. Scene parsing through ADE20K dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 633–641, 2017.

[51] Y. Zhou, T. Lei, H. Liu, N. Du, Y. Huang, V. Zhao, A. M. Dai, Q. V. Le, J. Laudon, et al. Mixture-of-experts with expert choice routing. *Advances in Neural Information Processing Systems*, 35:7103–7114, 2022.

# A Detailed Proof

## A.1 Proof of Theorem 1

**Theorem 1.** *For an exit layer $\ell \in \mathcal{E}$, assume that the confidence perturbation $\xi_{\ell,\boldsymbol{b}}(x)$ satisfies the tail bound*

$$\Pr(|\xi_{\ell,\boldsymbol{b}}(x)| > \tau_\ell) \le q_\ell(\tau_\ell, b_\ell),$$

*where $q_\ell(\tau_\ell, b_\ell)$ is non-increasing in $\tau_\ell$ and non-decreasing as $b_\ell$ decreases. Then, for any tolerance $\tau_\ell > 0$, the mis-exit probability satisfies*

$$\Pr(\tilde{e}_\ell(x) \neq e_\ell(x)) \le \Pr(|\Delta_\ell(x)| \le \tau_\ell) + q_\ell(\tau_\ell, b_\ell).$$

*Proof.* A mis-exit can occur only when the quantization-induced perturbation crosses the full-precision confidence boundary. Therefore, if $\tilde{e}_\ell(x) \neq e_\ell(x)$, then $|\xi_{\ell,\boldsymbol{b}}(x)| \ge |\Delta_\ell(x)|$. For any tolerance $\tau_\ell > 0$, this implies

$$\{\tilde{e}_\ell(x) \neq e_\ell(x)\} \subseteq \{|\Delta_\ell(x)| \le \tau_\ell\} \cup \{|\xi_{\ell,\boldsymbol{b}}(x)| > \tau_\ell\}.$$

By the union bound,

$$\Pr(\tilde{e}_\ell(x) \neq e_\ell(x)) \le \Pr(|\Delta_\ell(x)| \le \tau_\ell) + \Pr(|\xi_{\ell,\boldsymbol{b}}(x)| > \tau_\ell)\,.$$

Applying the assumed tail bound to the second term gives

$$\Pr(\tilde{e}_\ell(x) \neq e_\ell(x)) \le \Pr(|\Delta_\ell(x)| \le \tau_\ell) + q_\ell(\tau_\ell, b_\ell).$$

The theorem follows. □

## A.2 Proof of Corollary 1

**Corollary 1.** *Let $K_{\boldsymbol{\phi}}(x)$ and $\tilde{K}_{\boldsymbol{\phi},\boldsymbol{b}}(x)$ denote the first exit layers in $\mathcal{E}$ where the full-precision and quantized confidences reach or exceed $\phi_\ell$, respectively, with fallback value $L$ if no earlier exit is triggered. Then*

$$\Pr\Big(\tilde{K}_{\boldsymbol{\phi},\boldsymbol{b}}(x) \neq K_{\boldsymbol{\phi}}(x)\Big) \le \sum_{\ell \in \mathcal{E}} \Pr(\tilde{e}_\ell(x) \neq e_\ell(x))\,.$$

*Consequently,*

$$\Big|\mathbb{E}_{x\sim\mathcal{D}}\Big[\tilde{K}_{\boldsymbol{\phi},\boldsymbol{b}}(x)\Big] - \mathbb{E}_{x\sim\mathcal{D}}[K_{\boldsymbol{\phi}}(x)]\Big| \le (L-1)\sum_{\ell \in \mathcal{E}} \Pr(\tilde{e}_\ell(x) \neq e_\ell(x))\,.$$

*Proof.* If the full-precision and quantized stopping depths differ, then the two exit policies must disagree at least at one exit layer. Otherwise, if $\tilde{e}_\ell(x) = e_\ell(x)$ for all $\ell \in \mathcal{E}$, the first layer satisfying the exit condition would be the same under full precision and quantization, implying $\tilde{K}_{\boldsymbol{\phi},\boldsymbol{b}}(x) = K_{\boldsymbol{\phi}}(x)$. Hence,

$$\{\tilde{K}_{\boldsymbol{\phi},\boldsymbol{b}}(x) \neq K_{\boldsymbol{\phi}}(x)\} \subseteq \bigcup_{\ell \in \mathcal{E}} \{\tilde{e}_\ell(x) \neq e_\ell(x)\}.$$

Applying the union bound gives

$$\Pr\Big(\tilde{K}_{\boldsymbol{\phi},\boldsymbol{b}}(x) \neq K_{\boldsymbol{\phi}}(x)\Big) \le \sum_{\ell \in \mathcal{E}} \Pr(\tilde{e}_\ell(x) \neq e_\ell(x))\,.$$

For the expectation bound, since both stopping depths take values in $\{1, \ldots, L\}$, we have

$$\Big|\tilde{K}_{\boldsymbol{\phi},\boldsymbol{b}}(x) - K_{\boldsymbol{\phi}}(x)\Big| \le (L-1)\cdot \mathbf{1}\{\tilde{K}_{\boldsymbol{\phi},\boldsymbol{b}}(x) \neq K_{\boldsymbol{\phi}}(x)\}.$$

Taking expectation over $x \sim \mathcal{D}$ yields

$$\Big|\mathbb{E}_{x\sim\mathcal{D}}[\tilde{K}_{\boldsymbol{\phi},\boldsymbol{b}}(x)] - \mathbb{E}_{x\sim\mathcal{D}}[K_{\boldsymbol{\phi}}(x)]\Big| \le (L-1)\Pr\Big(\tilde{K}_{\boldsymbol{\phi},\boldsymbol{b}}(x) \neq K_{\boldsymbol{\phi}}(x)\Big)\,.$$

Combining this inequality with the union bound proves the claim. The corollary follows. □

### A.3 Proof of Theorem 2

**Theorem 2.** *For any realized stopping depth $k$, the accumulated representation error satisfies*

$$\|\tilde{h}_{k,\boldsymbol{b}}(x) - h_k(x)\| \le \sum_{\ell=1}^{k} \varepsilon_\ell(b_\ell) \prod_{r=\ell+1}^{k} \gamma_{r-1},$$

*where the empty product is defined as* 1*. Under early exiting with random stopping depth $K_{\boldsymbol{\phi}}(x)$, the expected distortion satisfies*

$$\mathbb{E}_{x\sim\mathcal{D}}\left[\|\tilde{h}_{K_{\boldsymbol{\phi}}(x),\boldsymbol{b}}(x) - h_{K_{\boldsymbol{\phi}}(x)}(x)\|\right] \le \mathbb{E}_{x\sim\mathcal{D}}\left[\sum_{\ell=1}^{K_{\boldsymbol{\phi}}(x)} \varepsilon_\ell(b_\ell) \prod_{r=\ell+1}^{K_{\boldsymbol{\phi}}(x)} \gamma_{r-1}\right].$$

*Proof.* Let

$$E_k(x) = \|\tilde{h}_{k,\boldsymbol{b}}(x) - h_k(x)\|.$$

By the assumed layer-wise distortion bound, for each layer $\ell = 1, \dots, L$,

$$E_\ell(x) \le \gamma_{\ell-1} E_{\ell-1}(x) + \varepsilon_\ell(b_\ell),$$

with initial distortion $E_0(x) = 0$. Unrolling this recurrence up to depth $k$ gives

$$E_k(x) \le E_0(x) \prod_{r=1}^{k} \gamma_{r-1} + \sum_{\ell=1}^{k} \varepsilon_\ell(b_\ell) \prod_{r=\ell+1}^{k} \gamma_{r-1}.$$

Since $E_0(x) = 0$, this reduces to

$$\|\tilde{h}_{k,\boldsymbol{b}}(x) - h_k(x)\| \le \sum_{\ell=1}^{k} \varepsilon_\ell(b_\ell) \prod_{r=\ell+1}^{k} \gamma_{r-1},$$

where the empty product is defined as 1.

Since the above bound holds for any realized stopping depth $k$, we can substitute the random stopping depth $K_{\boldsymbol{\phi}}(x)$ for $k$. Taking expectation over $x \sim \mathcal{D}$ yields

$$\mathbb{E}_{x\sim\mathcal{D}}\left[\|\tilde{h}_{K_{\boldsymbol{\phi}}(x),\boldsymbol{b}}(x) - h_{K_{\boldsymbol{\phi}}(x)}(x)\|\right] \le \mathbb{E}_{x\sim\mathcal{D}}\left[\sum_{\ell=1}^{K_{\boldsymbol{\phi}}(x)} \varepsilon_\ell(b_\ell) \prod_{r=\ell+1}^{K_{\boldsymbol{\phi}}(x)} \gamma_{r-1}\right].$$

The theorem follows. □

### A.4 Proof of Theorem 3

**Theorem 3.** *Assume that static MPQ ranks layers by their static sensitivities $S_\ell$, while APQ ranks layers by their utilization-weighted sensitivities $u_\ell(\boldsymbol{\phi})S_\ell$. If there exist two layers $i$ and $j$ such that*

$$S_i > S_j \quad \textit{but} \quad u_i(\boldsymbol{\phi})S_i < u_j(\boldsymbol{\phi})S_j,$$

*then the static MPQ ranking is inconsistent with the APQ ranking. Moreover, under equal marginal bit-cost increments and marginal APQ benefit proportional to $u_\ell(\boldsymbol{\phi})S_\ell$, the static MPQ allocation can be suboptimal for the APQ objective.*

*Proof.* The first claim follows directly from the assumed sensitivity and utilization-weighted orderings. Static MPQ ranks layer $i$ above layer $j$ because $S_i > S_j$, whereas APQ ranks layer $j$ above layer $i$ because $u_i(\boldsymbol{\phi})S_i < u_j(\boldsymbol{\phi})S_j$. Hence, the two rankings are inconsistent.

For the second claim, consider a fixed bit budget that allows assigning one additional bit-width increment to exactly one of the two layers. Under equal marginal bit-cost increments, the two choices have the same budget cost. If the marginal APQ benefit is proportional to $u_\ell(\boldsymbol{\phi})S_\ell$, then assigning the additional precision to layer $j$ yields a larger APQ benefit than assigning it to layer $i$, since

$$u_j(\boldsymbol{\phi})S_j > u_i(\boldsymbol{\phi})S_i.$$

However, static MPQ selects layer $i$ because $S_i > S_j$. Therefore, the static MPQ allocation is suboptimal for the APQ objective under this budget. The theorem follows. □

### A.5 Proof of Theorem 4

**Theorem 4.** *The APQ bi-level optimization is NP-hard. Moreover, each level is NP-hard on its own: (i) with fixed exit thresholds $\boldsymbol{\phi}$, the inner bit-width allocation is NP-hard; and (ii) with fixed bit-widths $\boldsymbol{b}$, the outer exit-threshold optimization is also NP-hard.*

*Proof.* We first prove that the inner-level bit-width allocation is NP-hard when the thresholds $\boldsymbol{\phi}$ are fixed (Lemma 1), and then show that the outer-level threshold optimization is NP-hard when the bit-widths $\boldsymbol{b}$ are fixed (Lemma 2).

**Lemma 1.** *Fix any thresholds $\boldsymbol{\phi}$. Consider the decision problem: "Does there exist a bit allocation $\boldsymbol{b} \in \mathcal{B}^L$ (e.g., discrete per-layer bit choices) such that the expected complexity $\mathbb{E}_x[\sum_\ell u_\ell(x, \boldsymbol{\phi}) c_\ell(b_\ell)]$ is at most $C_{\max}$ while the expected loss $\mathbb{E}_{(x,y)}[\mathcal{L}_{\mathrm{CE}}(f_{\theta,\boldsymbol{b},\boldsymbol{\phi}}(x), y)]$ is at most $\mathcal{L}_{\max}$?" This problem is NP-hard.*

*Proof.* Reduce from the classical *0–1 Knapsack*. Given items $\{1, \ldots, L\}$ with values $v_\ell > 0$, weights $w_\ell > 0$, and capacity $W$, ask whether there exists a subset $S$ with $\sum_{\ell \in S} w_\ell \le W$ and $\sum_{\ell \in S} v_\ell \ge V$. Construct an APQ instance with $L$ layers and a *binary* bit-choice set $\mathcal{B}_\ell = \{b_\ell^{(0)}, b_\ell^{(1)}\}$ per layer: i) Complexity mapping: set $c_\ell(\cdot)$ and (fixed) utilization $u_\ell(\cdot, \boldsymbol{\phi})$ so that $\mathbb{E}_x[u_\ell c_\ell(b_\ell^{(1)})] - \mathbb{E}_x[u_\ell c_\ell(b_\ell^{(0)})] = w_\ell$. ii) Accuracy mapping: choose a dataset and a head such that turning on the high-bit option at layer $\ell$ reduces the loss by exactly $v_\ell$ in expectation (e.g., via a separable surrogate $D_\ell(b_\ell)$ scaled appropriately). Set $C_{\max} = \sum_\ell \mathbb{E}_x[u_\ell c_\ell(b_\ell^{(0)})] + W$ and $\mathcal{L}_{\max} = L_0 - V$, where $L_0$ is the loss under all baselines $b_\ell^{(0)}$. Then selecting $S = \{\ell : b_\ell = b_\ell^{(1)}\}$ is feasible iff the knapsack instance is feasible. Hence NP-hardness follows. □

**Lemma 2.** *Fix any bit-widths $\boldsymbol{b}$. Consider the decision problem: "Does there exist thresholds $\boldsymbol{\phi}$ such that the expected complexity $C_{\mathrm{APQ}}(\boldsymbol{b}, \boldsymbol{\phi})$ is at most $C_{\max}$ while the expected loss $\mathbb{E}_{(x,y)}[\mathcal{L}_{\mathrm{CE}}(f_{\theta,\boldsymbol{b},\boldsymbol{\phi}}(x), y)]$ is at most $\mathcal{L}_{\max}$?" This problem is NP-hard.*

*Proof.* Reduce from *0–1 Knapsack*. Take a two-exit cascade (layers 1 and 2) with fixed logits/softmax scores precomputed on a finite dataset $\{x_i\}_{i=1}^n$. Let $\tilde{\rho}_{1,\boldsymbol{b}}(x_i) \in (0, 1)$ be the confidence at the first exit and suppose the final exit (layer 2) always predicts correctly with unit cost, while exiting at layer 1 costs zero additional depth but may misclassify some points. Associate each item $i$ with a sample $x_i$ and set: i) If $x_i$ exits at layer 1 (i.e., $\tilde{\rho}_{1,\boldsymbol{b}}(x_i) \ge \phi_1$), we *save* expected depth $w_i > 0$ (benefit), but we incur a loss penalty $p_i \ge 0$ if that early decision is wrong. ii) If it proceeds to layer 2 (i.e., $\tilde{\rho}_{1,\boldsymbol{b}}(x_i) < \phi_1$), we pay depth $w_i$ but incur no penalty.

Choose the surrogate loss so that total early-exit penalty equals $\sum_{i \in S} p_i$, where $S = \{i : \tilde{\rho}_{1,\boldsymbol{b}}(x_i) \ge \phi_1\}$. Then the constraints $C_{\mathrm{APQ}}(\boldsymbol{b}, \boldsymbol{\phi}) \le C_{\max}$ and $\mathbb{E}[\mathcal{L}_{\mathrm{CE}}] \le \mathcal{L}_{\max}$ become $\sum_{i \in S} w_i \ge V$ and $\sum_{i \in S} p_i \le W$, exactly the knapsack feasibility test. Thus finding $\phi_1$ is NP-hard. The argument extends to multiple exits by letting only the first threshold be active. □

Concluding Lemmas 1 and 2, each level already involves an NP-hard subproblem, even when the other level is fixed. Consequently, the joint bi-level problem is at least as hard as solving either level in isolation, thereby implying NP-hardness. Hence, the theorem follows. □

## B Experiments

### B.1 Implementation Details

All training runs are conducted on NVIDIA H100 GPUs. We adopt the SegFormer [44] head on a ViT-B for semantic segmentation, and conduct object detection experiments with a DeiT-B backbone [9]. For each model, we configure four exit heads: the first two are convolutional, and the latter two are attention-based. The exit heads are inserted at layers 4, 6, 8, and 10 in DeiT-B and ViT-B, and at layers 8, 12, 16, and 20 in ViT-L. For the Swin model, they are placed at the 2$^{\text{nd}}$, 4$^{\text{th}}$, 14$^{\text{th}}$, and 20$^{\text{th}}$ layers, approximately uniformly distributed with respect to computational cost. All models are trained with AdamW and a cosine-decay schedule with a $1 \times 10^{-5}$ peak LR.

Table 6: Classification results on Swin and ViT-B.

| | Bits (W/A) | Quant | EE | CIFAR-100 | | | | | ImageNet | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Controlled Perf. | | Std. Config | | | Controlled Perf. | | Std. Config | | |
| | | | | $\hat{L}\downarrow$ | BÔPs $\downarrow$ | $\bar{L}\downarrow$ | BŌPs $\downarrow$ | Acc. (%)$\uparrow$ | $\hat{L}\downarrow$ | BÔPs $\downarrow$ | $\bar{L}\downarrow$ | BŌPs $\downarrow$ | Acc. (%)$\uparrow$ |
| Swin | FP32 | – | – | – | – | *24.00* | *1.39E13* | *94.16* | – | – | *24.00* | *1.39E13* | *83.50* |
| | | | ViT-EE | 13.59 | 9.30E12 | 16.47 | 1.09E13 | 90.80 | 16.62 | 1.23E13 | 18.23 | 1.35E13 | 83.68 |
| | | | LGViT | 9.88 | 1.02E13 | 13.80 | 9.41E12 | 89.38 | 11.13 | 8.83E12 | 14.20 | 1.10E13 | 82.70 |
| | FP4/4 | RepQ | ViT-EE | N/A | N/A | 18.72 | 2.12E11 | 82.23 | N/A | N/A | 20.16 | 2.27E11 | 74.75 |
| | | | LGViT | N/A | N/A | 13.69 | 1.65E11 | 74.38 | N/A | N/A | 18.97 | 2.19E11 | 69.85 |
| | | ERQ | ViT-EE | 23.72 | 2.69E11 | 21.26 | 2.29E11 | 83.58 | 23.46 | 2.65E11 | 20.96 | 2.31E11 | 75.45 |
| | | | LGViT | N/A | N/A | 19.76 | 1.96E11 | 76.74 | N/A | N/A | 20.80 | 2.38E11 | 73.97 |
| | MP4/4 | MPTQ | ViT-EE | N/A | N/A | 21.01 | 2.36E11 | 81.62 | N/A | N/A | 20.99 | 2.39E11 | 71.55 |
| | | | LGViT | N/A | N/A | 16.56 | 1.90E11 | 75.07 | N/A | N/A | 21.23 | 2.42E11 | 74.35 |
| | **MAQEE** | | | **12.58** | **1.53E11** | **13.79** | **1.65E11** | **89.13** | **17.60** | **2.09E11** | **18.75** | **2.12E11** | **80.78** |
| ViT-B | FP32 | – | – | – | – | *12.00* | *1.80E13* | *90.47* | – | – | *12.00* | *1.80E13* | *83.10* |
| | | | ViT-EE | 7.09 | 1.81E13 | 7.27 | 1.89E13 | 87.51 | 7.65 | 1.90E13 | 8.68 | 2.05E13 | 79.60 |
| | | | LGViT | 5.90 | 1.68E13 | 6.33 | 1.76E13 | 88.51 | 6.69 | 1.80E13 | 7.08 | 1.86E13 | 80.30 |
| | FP4/4 | RepQ | ViT-EE | 11.78 | 3.94E11 | 9.90 | 3.51E11 | 85.71 | N/A | N/A | 10.83 | 3.71E11 | 74.67 |
| | | | LGViT | N/A | N/A | 8.08 | 3.13E11 | 73.52 | N/A | N/A | 8.74 | 3.31E11 | 72.30 |
| | | ERQ | ViT-EE | 8.77 | 3.23E11 | 9.28 | 3.35E11 | 87.82 | N/A | N/A | 9.88 | 3.49E11 | 77.35 |
| | | | LGViT | N/A | N/A | 9.00 | 3.34E11 | 77.64 | N/A | N/A | 9.12 | 3.40E11 | 73.06 |
| | MP4/4 | MPTQ | ViT-EE | 8.93 | 3.27E11 | 8.71 | 3.22E11 | 85.41 | N/A | N/A | 10.26 | 3.58E11 | 76.56 |
| | | | LGViT | 10.80 | 3.74E11 | 8.05 | 3.14E11 | 84.44 | 10.25 | 3.68E11 | 9.33 | 3.45E11 | 77.53 |
| | **MAQEE** | | | **5.04** | **2.37E11** | **5.83** | **2.57E11** | **88.51** | **7.04** | **2.85E11** | **7.26** | **2.93E11** | **79.41** |

The admissible weight precisions range from $b_{\min} = 3$ to $b_{\max} = 6$. The empirical solver settings and recovery losses are specified in Appendices B.9 and B.10.

### B.2 Results on DeiT-B and ViT-L

On DeiT-B, MAQEE exits approximately 3–4 layers earlier than the strongest quantized baseline at the same target accuracy (Table 1). Although MPQ alleviates part of the accuracy degradation of FPQ under Early Exiting, its assumption of a fixed inference path constrains exit efficiency, leaving it less effective than MAQEE. A similar pattern is observed for ViT-L. While the increased parameter count provides higher capacity, the greater depth also amplifies quantization-induced perturbations, leading to limited gains in quantized model performance. On ViT-L, MAQEE exits approximately 9–11 layers earlier than the strongest quantized baseline at the same target accuracy.

### B.3 Additional Experimental Settings

McQueen and QuEE are implemented from their papers on CIFAR-100/ViT-B. McQueen optimizes continuous layer precision under an adaptive budget penalty and then discretizes the allocation. QuEE retains W3, W4, and W5 block parameterizations to construct candidate paths. Its original higher-bit ViT setting and the W3–W5 adaptation therefore differ in the quality of the candidate classifiers. MAQEE uses a single mixed-precision allocation. Long-Tail gives shallow layers higher precision, while utilization-only allocates precision according to execution frequency; both share the average W4 and A4 constraints in Table 7.

### B.4 Risk Diagnostics

The perturbation study starts from uniform W4A4 and lowers one block to W3 at a time. The 10,000 evaluation images are disjoint from the risk-score set, and the reference model is restored before each intervention. Perturbing blocks in the highest quartile of the residual-update diagnostic score yields 2.47 times the local CLS drift and 1.63 times the final-layer drift observed in the lowest quartile. Controlling for layer depth gives a partial rank correlation of 0.671 ($p = 0.0266$) with local drift.

Table 7: Allocation strategies on ImageNet/ViT-B with an average W4 budget and A4 activations.

| Allocation | Top-1 (%) | $\bar{L}\downarrow$ | $\bar{\text{BOPs}}\downarrow$ |
|---|---|---|---|
| Uniform | 73.56 | 9.12 | 3.40E11 |
| Long-Tail | 78.63 | 8.75 | 3.36E11 |
| Utilization-only | 78.72 | 9.26 | 3.29E11 |
| **MAQEE** | **79.41** | **7.26** | **2.93E11** |

Table 8: ImageNet/ViT-B performance under different exit policies.

| Policy | ImageNet Top-1 (%) | $\bar{L}\downarrow$ | $\bar{\text{BOPs}}\downarrow$ |
|---|---|---|---|
| Aggressive | 75.57 | 5.68 | 2.61E11 |
| Default | 79.41 | 7.26 | 2.93E11 |
| Conservative | 80.10 | 8.13 | 3.10E11 |

For BSR, we examine downstream exit gates reached by both the reference and perturbed models. Decisions within 0.02 of the confidence threshold flip at a rate of 48.62%, compared with 26.23% for other decisions. The risk ratio is 1.85 (95% bootstrap CI: [1.76, 1.94]). Near-boundary decisions have premature- and delayed-exit risk ratios of 2.70 and 1.65, respectively. Flip rates decrease from 49.32% at boundary distances below 0.005 to 16.60% at distances of at least 0.20. These results connect proximity to the exit threshold with observed decision instability.

### B.5 Deployment Protocol

The deployment uses PyTorch 2.9.0+cu128 on an NVIDIA GeForce RTX 5090. CIFAR-100 uses the ordered 10,000-image test set; ImageNet uses 10,000 shuffled validation images with seed 777. Inputs are resized to $224 \times 224$. The runtime divides the backbone into blocks 1–4, 5–6, 7–8, 9–10, and 11–12. Each segment includes confidence evaluation. The host reads the resulting exit decision before executing the next segment. Logical W2–W4 weights use packed INT4 containers, while W5–W8 use INT8 containers; activations use A4. Peak allocation measures live allocated memory, separately from reserved CUDA Graph pools. The deployment configurations are evaluated with their own fixed thresholds.

### B.6 Allocation Strategies

Table 7 separates the benefit of utilization from that of risk-aware allocation. Long-Tail assigns more precision to shallow layers, while utilization-only uses their execution frequency. MAQEE improves Top-1 accuracy over utilization-only by 0.69 percentage points and reduces computation by 3.60E10 BOPs. The full optimization evaluates the effect of precision changes on representations and exit decisions, adding calibration work to the simpler allocation rules. At inference, it uses the resulting fixed allocation and confidence thresholds.

### B.7 Threshold Policies and ImageNet-R

We hold the ImageNet/ViT-B allocation fixed and vary the source-domain exit policy. Table 8 shows that the aggressive, default, and conservative policies span 75.57–80.10% ImageNet accuracy and 2.61E11–3.10E11 BOPs. All three policies were constructed on ImageNet before the ImageNet-R experiments. We evaluate the default and conservative policies on ImageNet-R with their thresholds, allocation, and model parameters fixed. Table 9 reports the results on ImageNet-R as a distribution-shift stress test. The uniform and MAQEE comparisons share the backbone, exit heads, ImageNet calibration data, and source-domain threshold-selection procedure. Both allocations experience a substantial accuracy decrease with the default policy. MAQEE improves accuracy over uniform W4A4 + EE by 1.19 and 4.08 percentage points under the default and conservative policies. Runtime measurements cover all 30,000 ImageNet-R images. Relative to full-depth W4A4, the conservative MAQEE policy reduces latency by 9.26% while trading 3.19 percentage points of Top-1 accuracy.

Table 9: ImageNet-R performance and GPU efficiency on ViT-B.

| Model / policy | Top-1 (%) | $\bar{L}\downarrow$ | ms | img/s |
|---|---|---|---|---|
| FP32, full depth | 51.55 | 12.00 | 1.608 | 622.0 |
| Uniform W4A4, full depth | 38.34 | 12.00 | 1.279 | 781.6 |
| Uniform W4A4 + EE, default | 15.16 | 8.57 | – | – |
| Uniform W4A4 + EE, conservative | 31.07 | 10.85 | – | – |
| **MAQEE, default** | **16.35** | **8.54** | **0.940** | **1063.8** |
| **MAQEE, conservative** | **35.15** | **10.41** | **1.161** | **861.4** |

Table 10: ImageNet classification results on ConvNeXt-Base.

| Model | Top-1 (%) | $\bar{L}\downarrow$ | $\bar{\text{BOPs}}\downarrow$ |
|---|---|---|---|
| FP32, full depth | 83.99 | 36.00 | – |
| EERO, uniform W4A4 | 72.92 | 30.51 | 2.21E11 |
| **MAQEE, mixed precision** | **77.55** | **25.34** | **1.90E11** |

### B.8 Transfer to a CNN Backbone

We evaluate ConvNeXt-Base on ImageNet with EERO [39] early-exit heads and QDrop [42] quantization. MAQEE assigns W3–W5 to 36 pointwise blocks with an exact average weight precision of W4 and fixed A4 activations. The quantized models share exit heads, quantization technique, and evaluation protocol. In Table 10, MAQEE improves accuracy by 4.63 percentage points and reduces computation by 3.10E10 BOPs. The allocation varies within stages, assigning additional precision to selected blocks rather than following a monotonic depth schedule.

### B.9 Risk and Allocation Details

The threshold solver ranks policies by calibration cost and exit risk subject to the accuracy target in Eq. (12). The allocator maximizes the utility in Eq. (13) under the bit budget $B = L\bar{b}$. For classification, the implementation uses mean exit cross-entropy to form the allocation scores. Subtracting the shared mean final-classifier loss gives the PGR in Eq. (9); this common offset cancels under the percentile normalization below. Each exit's loss and BSR are assigned to the blocks since the preceding exit; the final segment uses the final-classifier loss and zero BSR. For each exit, $u_e(x, \boldsymbol{\phi})$ records whether the current quantized model reaches it. BSR counts samples that reach an exit and satisfy $|\tilde{\rho}_e - \phi_e| < 0.02$, divided by the calibration-set size.

For QID, $\sigma_\ell$ and $\tilde{\sigma}_\ell$ are the teacher and student residual-update RMS magnitudes, each averaged over batches. We then take their student-to-teacher ratio with $\epsilon = 10^{-8}$ in Eq. (8). Near a ratio of one, squared log deviation approximates squared relative deviation. For SQNR, let $d_\ell = h_\ell - h_{\ell-1}$ and $\tilde{d}_\ell = \tilde{h}_\ell - \tilde{h}_{\ell-1}$ denote their residual updates. We use

$$r_{\mathrm{SQNR},\ell} = 10 \log_{10} \frac{\sum_{x \in \mathcal{D}_{\mathrm{cal}}} \|d_\ell(x) - \tilde{d}_\ell(x)\|^2}{\sum_{x \in \mathcal{D}_{\mathrm{cal}}} \|d_\ell(x)\|^2}. \tag{15}$$

This residual-update estimator measures layer-local quantization noise for allocation. Components are scaled and clipped using their 10th and 90th percentiles across layers; constant components map to 0.5. Weights for SQNR, PGR, BSR, and QID are 1.0, 0.5, 0.5, and 0.5. The allocation sensitivity is $s_\ell = s_{\min} + (1 - s_{\min})\mathcal{N}(R)_\ell$, with $s_{\min} = 0.05$.

For allocation, we model the remaining quantization risk at bit-width $b$ as $s_\ell 2^{-2b}$, holding $s_\ell$ fixed during each allocation step. The estimated reduction from one additional bit and its accumulated benefit are

$$\begin{aligned} \Delta R_\ell(b) &= s_\ell\big(2^{-2b} - 2^{-2(b+1)}\big), \\ U_\ell(b) &= \sum_{q=b_{\min}}^{b-1} \Delta R_\ell(q) = s_\ell\big(2^{-2b_{\min}} - 2^{-2b}\big). \end{aligned} \tag{16}$$

The factor $2^{-2b}$ models squared quantization error as proportional to the squared quantization step. Candidate BOPs are evaluated using the current exit distribution. For each layer and candidate bit-

width, the solver computes the cost increase relative to the all-minimum-bit allocation; this equals the sum of the one-bit increments defined in Section 5.3.

Weight bit-widths range from 3 to 6, with activations fixed at 4 bits. For 12 blocks at average W4, $B = 48$. Dynamic programming tracks accumulated utility and BOPs at each bit sum. At each layer, it retains the path with the highest utility, choosing lower BOPs when utilities are equal.

Threshold search uses confidence quantiles and local refinement. Among policies that meet the accuracy target, we compare BOPs first, followed by the two components of $R_{\text{exit}}$ in Eq. (12): BSR and exit prediction loss. For each sample and intermediate exit, we record whether the sample reaches that exit and its confidence lies within 0.02 of the threshold. BSR is the mean of these indicators over all calibration samples and intermediate exits. Exit prediction loss is the mean loss at the selected exit for each sample, including those that reach the final classifier. Any remaining ties favor the policy whose accuracy exceeds the target by the smaller amount. If no policy meets the target, it selects the highest-accuracy policy, using BOPs to break ties. Across rounds, selection minimizes BOPs among target-feasible models; otherwise, it chooses the model whose accuracy is closest to the target, using BOPs to break ties.

### B.10 Recovery Implementation

For classification, let $z_{\mathrm{T}}(x)$ be the frozen teacher's final logits and $\tilde{z}_e(x)$ the student logits at classifier $e$. We use

$$\begin{aligned} p_{\mathrm{T}}(x) = \mathrm{softmax}(z_{\mathrm{T}}(x)), \quad p_e(x) = \mathrm{softmax}(\tilde{z}_e(x)), \\ \ell_{\mathrm{CE},e}(x,y) = \mathrm{CE}(\tilde{z}_e(x), y), \quad \ell_{\mathrm{KL},e}(x) = D_{\mathrm{KL}}(p_{\mathrm{T}}(x) \| p_e(x)). \end{aligned} \tag{17}$$

Both distributions use temperature 1. The losses $\mathcal{L}_{\mathrm{CE},e}$ and $\mathcal{L}_{\mathrm{KL},e}$ in Eq. (14) average these sample losses, using the mask below at intermediate exits. If $\beta$ is the intermediate-exit loss coefficient and $v_e$ the relative exit weight, then $\omega_e = \beta v_e / \sum_{j \in \mathcal{E}} v_j$ for intermediate exits and $\omega_L = 1 - \beta$.

**Confidence-based sample masking.** Let $e_1 < \cdots < e_H$ denote the intermediate exits. During recovery, the active-sample mask at exit $e_j$ is

$$m_j(x) = \prod_{k=1}^{j} \mathbf{1}\{\tilde{\rho}_{e_k}(x) < 1.1\phi_{e_k}\}. \tag{18}$$

CE and KL are averaged over active samples, with a minimum denominator of one. Crossing $1.1\phi_e$ excludes a sample at that and all later intermediate exits; the final loss is unmasked. Inference exits at the first confidence reaching $\phi_e$.

## C Bi-Level Optimization Algorithm

**Algorithm 1** Bi-level optimization with recovery

**Require:** FP anchor $\theta_{\mathrm{FP}}$, frozen teacher $\theta_{\mathrm{T}}$; datasets $\mathcal{D}_{\mathrm{train}}, \mathcal{D}_{\mathrm{cal}}, \mathcal{D}_{\mathrm{eval}}$; initial allocation $\boldsymbol{b}^{(0)}$, thresholds $\boldsymbol{\phi}^{(0)}$; target $A_{\mathrm{target}}$, average precision $\bar{b}$, admissible bits $\mathcal{B}$, round budget $K_{\max}$.
**Ensure:** Recovered model $\theta^\star$, allocation $\boldsymbol{b}^\star$, thresholds $\boldsymbol{\phi}^\star$.
**Initialization:** $B \leftarrow L\bar{b}$, $\mathcal{V} \leftarrow \varnothing$, $\mathcal{C} \leftarrow \varnothing$.
**for** $k = 0, \ldots, K_{\max} - 1$ **do**
  $\mathcal{V} \leftarrow \mathcal{V} \cup \{\boldsymbol{b}^{(k)}\}$.
  **Self-distillation recovery:**
  Construct the quantized model with $\boldsymbol{b}^{(k)}$ from $\theta_{\mathrm{FP}}$.
  $\theta^{(k)} \leftarrow \mathrm{Recover}(\mathcal{D}_{\mathrm{train}}, \theta_{\mathrm{T}}, \boldsymbol{\phi}^{(k)}; \mathcal{L}_{\mathrm{rec}})$ (Eq. (14)).
  Collect fresh confidences $\tilde{\rho}_\ell(x)$ and representation statistics on $\mathcal{D}_{\mathrm{cal}}$.
  **Outer loop (threshold search):**
  Initialize threshold search with $\boldsymbol{\phi}^{(k)}$; build and refine candidate policies $\mathcal{T}_\phi$ from confidence quantiles.
  $\mathcal{F}_k \leftarrow \{\boldsymbol{\phi} \in \mathcal{T}_\phi : \widehat{\mathrm{Acc}}(\boldsymbol{b}^{(k)}, \boldsymbol{\phi}) \geq A_{\mathrm{target}}\}$.
  $\widehat{\boldsymbol{\phi}}^{(k)} \leftarrow \arg\min^{\mathrm{lex}}_{\boldsymbol{\phi} \in \mathcal{F}_k} \big(C_{\mathrm{APQ}}(\boldsymbol{b}^{(k)}, \boldsymbol{\phi}), R_{\mathrm{exit}}(\boldsymbol{b}^{(k)}, \boldsymbol{\phi})\big)$ (Eq. (12)).
  Evaluate accuracy $a_k$ and BOPs $c_k$ on $\mathcal{D}_{\mathrm{eval}}$.
  $\mathcal{C} \leftarrow \mathcal{C} \cup \{(\theta^{(k)}, \boldsymbol{b}^{(k)}, \widehat{\boldsymbol{\phi}}^{(k)}, a_k, c_k)\}$.
  **Inner loop (bit allocation):**
  Compute block risks $r_{j,\ell}$ under $\widehat{\boldsymbol{\phi}}^{(k)}$, mapping exit risks to block segments.
  $R_\ell \leftarrow \sum_j w_j \mathcal{N}(r_j)_\ell$ for all $\ell = 1, \ldots, L$.
  **for** $\ell = 1, \ldots, L$ **do**
    $s_\ell \leftarrow s_{\min} + (1 - s_{\min})\mathcal{N}(R)_\ell$.
    $U_\ell(b) \leftarrow \sum_{q=b_{\min}}^{b-1} \Delta R_\ell(q)$, $b \in \mathcal{B}$ (Eq. (16)).
    Compute candidate BOPs increments under $\widehat{\boldsymbol{\phi}}^{(k)}$.
  **end for**
  $\boldsymbol{b}^{(k+1)} \leftarrow \arg\max_{\boldsymbol{b} \in \mathcal{B}^L, \sum_\ell b_\ell = B} \sum_{\ell=1}^{L} U_\ell(b_\ell)$ by dynamic programming (Eq. (13)).
  At each bit-sum state, prefer lower accumulated BOPs when utilities are equal.
  **if** $\boldsymbol{b}^{(k+1)} \in \mathcal{V}$ **then**
    **break**
  **end if**
  $\boldsymbol{\phi}^{(k+1)} \leftarrow \widehat{\boldsymbol{\phi}}^{(k)}$.
**end for**
$\mathcal{C}_{\mathrm{feas}} \leftarrow \{c \in \mathcal{C} : a(c) \geq A_{\mathrm{target}}\}$; $c^\star \leftarrow \arg\min_{c \in \mathcal{C}_{\mathrm{feas}}} \mathrm{BOPs}(c)$.
**return** $(\theta^\star, \boldsymbol{b}^\star, \boldsymbol{\phi}^\star)$ from $c^\star$.

# D Notation Table

Table 11: Notation used throughout the paper.

| Symbol | Description |
|---|---|
| $L$ | Number of layers. |
| $\ell$ | Layer index, $\ell = 1, \ldots, L$. |
| $x, y$ | Input sample and label. |
| $\mathcal{D}$ | Data distribution. |
| $\theta$ | Model parameters. |
| $\mathcal{B}$ | Set of admissible bit-widths, e.g., $\{2, 4, 8\}$. |
| $b_\ell$ | Bit-width assigned to layer $\ell$, with $b_\ell \in \mathcal{B}$. |
| $\boldsymbol{b}$ | Layer-wise bit allocation, $\boldsymbol{b} = \{b_\ell\}_{\ell=1}^{L}$. |
| $B_w, B_a$ | Weight and activation bit-widths. |
| $B$ | Total bit budget, $B = L\bar{b}$. |
| $c_\ell(b_\ell)$ | Computational cost of layer $\ell$ under bit-width $b_\ell$. |
| $C_{\mathrm{APQ}}(\boldsymbol{b}, \boldsymbol{\phi})$ | Expected bit operations per input under allocation $\boldsymbol{b}$ and exit policy $\boldsymbol{\phi}$. |
| $z_\ell(x)$ | Full-precision logits at layer $\ell$. |
| $\rho_\ell(x)$ | Full-precision confidence, $\rho_\ell(x) = \max_j \mathrm{softmax}(z_\ell(x))_j$. |
| $\tilde{\rho}_\ell(x)$ | Quantized confidence; shorthand for $\tilde{\rho}_{\ell,\boldsymbol{b}}(x)$. |
| $\phi_\ell$ | Confidence threshold at exit layer $\ell \in \mathcal{E}$. |
| $\boldsymbol{\phi}$ | Exit policy, $\boldsymbol{\phi} = \{\phi_e\}_{e \in \mathcal{E}}$. |
| $K_{\boldsymbol{\phi}}(x)$ | Stopping depth induced by the full-precision exit policy. |
| $\widetilde{K}_{\boldsymbol{\phi},\boldsymbol{b}}(x)$ | Stopping depth after quantization. |
| $u_\ell(x, \boldsymbol{\phi})$ | Indicator that input $x$ reaches layer $\ell$. |
| $u_\ell(\boldsymbol{\phi})$ | Expected utilization of layer $\ell$, $\mathbb{E}_{x\sim\mathcal{D}}[u_\ell(x, \boldsymbol{\phi})]$. |
| $\Delta_\ell(x)$ | Confidence margin, $\Delta_\ell(x) = \rho_\ell(x) - \phi_\ell$. |
| $\xi_{\ell,\boldsymbol{b}}(x)$ | Quantization-induced confidence perturbation. |
| $h_\ell(x)$ | Full-precision hidden state at layer $\ell$. |
| $\tilde{h}_{\ell,\boldsymbol{b}}(x)$ | Quantized hidden state at layer $\ell$. |
| $\varepsilon_\ell(b_\ell)$ | Local quantization error bound at layer $\ell$. |
| $\gamma_\ell$ | Local error amplification factor. |
| $S_\ell$ | Static quantization sensitivity of layer $\ell$. |
| $\mathrm{PGR}_\ell$ | Excess loss relative to the full-depth output at layer $\ell$. |
| $\mathrm{BSR}_\ell$ | Boundary sensitivity risk at layer $\ell$. |
| $\mathrm{QID}_\ell$ | Quantization-induced drift at layer $\ell$. |
| $\sigma_\ell, \tilde{\sigma}_\ell$ | Full-precision and quantized residual-update magnitudes. |
| $\mathrm{SQNR}_\ell^{-1}$ | Inverse signal-to-quantization-noise ratio at layer $\ell$. |
| $R_\ell$ | Overall risk score of layer $\ell$. |
| $R_{\mathrm{exit}}$ | Ordered pair of policy boundary sensitivity and exit prediction loss. |
| $w_j$ | Weight of risk component $j$. |
| $U_\ell(b)$ | Accumulated estimated risk reduction at bit-width $b$. |
| $\Delta R_\ell(b)$ | Estimated risk reduction from increasing $b$ by one bit. |
| $\Delta\mathrm{BOPs}_\ell$ | Amortized cost increase from one additional bit at layer $\ell$. |
| $\boldsymbol{b}^{+\ell}$ | Allocation with only layer $\ell$ increased by one bit. |
| $\lambda$ | Trade-off coefficient in the APQ objective. |
| $\lambda_{\mathrm{outer}}$ | Trade-off coefficient for threshold optimization. |
| $\lambda_{\mathrm{inner}}$ | Trade-off coefficient for bit-width allocation. |
| $\mathcal{E}$ | Set of intermediate exit layers. |
| $\mathcal{T}_\phi$ | Set of candidate exit policies. |
| $s_\ell$ | Normalized layer sensitivity for precision allocation. |
| $\bar{b}$ | Prescribed average weight precision. |
| $\widehat{\mathrm{Acc}}(\boldsymbol{b}, \boldsymbol{\phi})$ | Estimated accuracy on the calibration set. |
| $A_{\mathrm{target}}$ | Target accuracy for threshold and model selection. |
| $\hat{L}$ | Exit layer required to reach the target performance. |
| $\bar{L}$ | Average exit layer. |
| $\widehat{\mathrm{BOPs}}$ | BOPs required to reach the target performance. |
| $\overline{\mathrm{BOPs}}$ | Average BOPs under the standard configuration. |

## NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims are stated in the abstract and Section 1. They concern the APQ formulation, the MAQEE framework, the risk-aware optimization of exit thresholds and bit-widths, and the empirical accuracy–efficiency trade-offs. These claims are supported by the theoretical analysis in Section 4 and the experiments in Section 6.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 7 discusses calibration, deployment, and generalization limits. Section 6.2 reports segmentation and detection results. Section 8 also discusses future extensions beyond early-exit ViTs, such as recursive Transformers, MoE routing, and layer skipping.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete proof?

   Answer: [Yes]

   Justification: The assumptions for the main theoretical results are stated in Section 4 and Section 5. Complete proofs are provided in Appendix A.

4. **Experimental result reproducibility**

   Question: Does the paper disclose the information needed to reproduce the main experimental results?

   Answer: [Yes]

   Justification: Section 6 describes the datasets, tasks, backbones, baselines, evaluation protocol, metrics, and quantization settings. Additional implementation details are provided in Appendix B.1.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code?

   Answer: [Yes]

   Justification: The paper uses public benchmark datasets. The source-code link is provided in the abstract.

6. **Experimental setting/details**

   Question: Does the paper specify all training and test details necessary to understand the results?

   Answer: [Yes]

   Justification: The experimental setup is described in Section 6, including datasets, backbones, baselines, W4A4 quantization, controlled-performance targets, and evaluation metrics. Additional hyperparameters and implementation details are reported in Appendix B.1.

7. **Experiment statistical significance**

   Question: Does the paper report error bars or other appropriate information about statistical significance?

   Answer: [Yes]

   Justification: The paper reports accuracy, average exit layer, and BOPs under standardized evaluation protocols, along with measures of statistical significance.

8. **Experiments compute resources**

   Question: Does the paper provide sufficient information on compute resources?

   Answer: [Yes]

   Justification: Compute resources and implementation details are reported in Appendix B.1.

9. **Code of ethics**

   Question: Does the research conform to the NeurIPS Code of Ethics?

   Answer: [Yes]

   Justification: The paper studies efficient inference for vision transformers using quantization and early exiting. It uses public benchmark datasets and does not involve human-subject experiments, private data collection, deception, or high-risk deployment.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive and negative societal impacts?

    Answer: [N/A]

    Justification: This paper focuses on technical improvements in efficient ViT inference and does not directly involve topics with clear positive or negative societal impacts.

11. **Safeguards**

    Question: Does the paper describe safeguards for responsible release of high-risk assets?

    Answer: [N/A]

    Justification: The paper does not release high-risk pretrained language models, image generators, scraped datasets, or sensitive data. It releases code for reproducing experiments on public vision benchmarks.

12. **Licenses for existing assets**

    Question: Are existing assets properly credited and are licenses respected?

    Answer: [Yes]

    Justification: The paper cites the datasets, backbones, and baseline methods used in the experiments, as described in Section 6 and the references.

13. **New assets**

    Question: Are new assets introduced in the paper well documented?

    Answer: [Yes]

    Justification: The new asset is the MAQEE codebase, linked in the abstract. Implementation details are provided in Appendix B.1.

14. **Crowdsourcing and research with human subjects**

    Question: Does the paper include details for crowdsourcing or human-subject experiments?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or human-subject research.

15. **Institutional review board approvals**

    Question: Does the paper describe IRB approvals or equivalent review for human-subject research?

    Answer: [N/A]

    Justification: The paper does not involve human-subject research, so IRB approval is not applicable.

16. **Declaration of LLM usage**

    Question: Does the paper describe LLM usage if it is important to the core method?

    Answer: [N/A]

    Justification: LLMs are not used as an important, original, or non-standard component of the proposed method. The paper focuses on quantization and early-exit optimization for vision transformers.