# AQUA: Adaptive Quantization with Unified Format Allocation

## Research Plan for GLSVLSI 2026 Submission

## 1. Core Idea (One-Paragraph Summary)

We propose **AQUA**, a unified quantization-aware training framework that jointly learns two things per layer: (1) **whether to use integer or floating-point quantization format**, and (2) **how to adapt the effective precision based on input difficulty**. Unlike all existing QAT methods that apply a fixed, manually-chosen quantization format (INT8 or FP8) uniformly across the model, our framework introduces differentiable format and precision parameters that are optimized end-to-end alongside model weights. A lightweight difficulty estimator routes "easy" inputs through lower precision paths and "hard" inputs through higher precision paths, reducing average energy consumption during deployment. The framework automatically discovers that layers with outlier-heavy activation distributions prefer floating-point formats, while layers with uniform distributions prefer integer formats — a finding with direct implications for hardware accelerator design.

## 2. Why This Is Novel (Gap Analysis)

What Exists	What's Missing (Our Contribution)
EQAT (Karkehabadi 2025): learns per-layer bit-width	Uses only INT format; no format selection; same precision for all inputs
FP8 literature (2024-2025): compares INT8 vs FP8	Format chosen manually/uniformly; not learned per-layer during training
HAWQ: Hessian-based mixed precision	Only bit-width, not format; static at inference
Arbitrary Bit-width Networks (Tang 2022): input-dependent bit-width	Only spatial regions in feature maps; no format selection; uses RL not differentiable
SDP (Structured Dynamic Precision): input-dependent precision	Hardware co-design for spatial regions; no format selection; no QAT integration
AdaQAT (Gernigon 2024): adaptive bit-width during training	No format selection; no input-adaptivity at inference

Key novelty claims:
- **C1:** First framework to jointly learn quantization *format* (INT vs FP) and *bit-width* per layer during QAT
- **C2:** First input-adaptive quantization that adjusts precision based on input difficulty at inference time
- **C3:** Empirical discovery of format preferences across layer types (with hardware design implications)
- **C4:** Energy model that accounts for format-dependent costs (INT and FP have different energy profiles)

## 3. Technical Approach

### 3.1 Differentiable Format Selection

For each layer *i*, we learn a continuous format parameter $\tilde{f}_i$ mapped through sigmoid:

```
$$f_i = \sigma(\tilde{f}_i) \quad \in [0, 1]$$
```

Where $f_i = 0$ means pure integer quantization and $f_i = 1$ means pure floating-point quantization.

During training, we interpolate between INT and FP quantized outputs:

```
$$\hat{w}_i = (1 - f_i) \cdot \text{QuantINT}(w, q_i) + f_i \cdot \text{QuantFP}(w, q_i)$$
```

During inference, we discretize: if $f_i > 0.5$, use FP; otherwise use INT.

Why this works: INT quantization has uniform step sizes (good for uniform distributions), while FP quantization has non-uniform step sizes concentrated near zero (good for distributions with outliers). The gradient naturally pushes each layer toward the format that minimizes its quantization error.

### 3.2 Differentiable Bit-Width (Same as EQAT)

For each layer *i*, bit-width is learned via sigmoid:

```
$$q_i = q\_min + (q\_max - q\_min) \cdot \sigma(\tilde{q}_i)$$
```

Where q_min = 2 and q_max = 8.

### 3.3 Input Difficulty Estimator

A lightweight module (2-3 conv layers + global average pooling + FC) that takes the input and outputs a scalar difficulty score $d_x \in [0, 1]$:

```
$$d_x = \text{DifficultyEstimator}(x)$$
```

The effective bit-width for input x at layer i becomes:

```
$$q_i(x) = q\_min\_eff + (q_i - q\_min\_eff) \cdot d_x$$
```

Where q_min_eff is the minimum effective bit-width (e.g., 2). Easy inputs ($d_x \to 0$) get minimum precision; hard inputs ($d_x \to 1$) get full learned precision.

Training the difficulty estimator: It's trained end-to-end. Its gradient comes from the total loss — if reducing precision on an input hurts accuracy, the difficulty score for that input increases.

Overhead: The estimator is tiny (<0.1% of model parameters). It runs once per input and shares the result across all layers, so the inference overhead is negligible.

### 3.4 Comprehensive Loss Function

```
L_total = L_CE(y_quant, y_true)                    [accuracy]
        + α · KL(y_quant || y_full)                [distribution alignment]
        + β · E_total({q, f})                      [energy]
        + γ · H(f)                                 [format decisiveness]
```

Where:
- **L_CE**: Cross-entropy loss on quantized predictions
- **KL**: Aligns quantized and full-precision outputs (from EQAT)
- **E_total**: Format-aware energy model (new — see below)
- **H(f)**: Entropy regularization that encourages format parameters to be close to 0 or 1 (decisive), avoiding the interpolation region: $H(f) = -\Sigma[f_i \cdot \log(f_i) + (1-f_i) \cdot \log(1-f_i)]$

### 3.5 Format-Aware Energy Model

Different from EQAT, our energy model distinguishes between INT and FP:

```
E_layer_i = E_comp_i + E_data_i

E_comp_i = MACs_i × [(1-f_i) × E_INT_MAC × q_i^2 + f_i × E_FP_MAC × q_i^1.5]

E_data_i = DataSize_i × E_access × q_i
```

Key insight: **INT MAC energy scales quadratically with bit-width, while FP MAC energy scales at ~q^1.5** because FP multipliers have different circuit complexity. Data movement energy is the same for both formats. These scaling factors come from published hardware measurements (Yang et al. 2017).

The average energy per input accounts for input-adaptive precision:

```
E_avg = E_x[E_total(q(x), f)]
```

Which we approximate over each training batch.

### 3.6 Gradient Derivations

Format gradient:
```
∂L/∂f̃_i = ∂L/∂f_i · σ(f̃_i) · (1 - σ(f̃_i))
```

Where $\partial L/\partial f_i$ has components from each loss term:
```
∂L/∂f_i = ∂L_CE/∂f_i + α·∂KL/∂f_i + β·∂E/∂f_i + γ·∂H/∂f_i
```

The CE and KL gradients w.r.t. format come from the interpolation:
```
```

```
∂ŵ/∂f_i = QuantFP(w, q_i) - QuantINT(w, q_i)
```

This is the key — the gradient tells the optimizer which format produces less error for each layer.

Bit-width gradient: Same as EQAT (Eq. 21 in their paper), plus the format-dependent energy term.

Difficulty estimator gradient: Standard backpropagation through the estimator network. The gradient from the energy term encourages lower difficulty scores (lower precision), while the accuracy gradient pushes difficulty scores up for inputs that need it.

## 4. Algorithm (Pseudocode)

```
ALGORITHM: AQUA Training

Input: Training data D, model weights θ, bit-width params {q̃},
    format params {f̃}, difficulty estimator ϕ
    Hyperparameters α, β, γ

Initialize θ, {q̃}, {f̃}, ϕ

for epoch = 1 to max_epochs do
  β_epoch = β · min(1.0, epoch / warmup_epochs)    // Progressive energy penalty
  γ_epoch = γ · min(1.0, epoch / format_warmup)    // Progressive format decisiveness

  for batch (x, y) in D do
    // Compute input difficulty
    d_x = DifficultyEstimator_ϕ(x)

    // Compute effective bit-widths and formats per layer
    for each layer i:
      q_i = q_min + (q_max - q_min) · σ(q̃_i)
      q_i_eff = q_min_eff + (q_i - q_min_eff) · d_x    // Input-adaptive
      f_i = σ(f̃_i)

    // Forward passes
    y_full = f_θ(x)                              // Full-precision
    ŵ = (1-f) · QuantINT(w, q_eff) + f · QuantFP(w, q_eff)  // Mixed-format
    y_quant = f_{θ,quantized}(x)

    // Compute loss
    L_CE = CrossEntropy(y_quant, y)
    L_KL = KLDivergence(y_quant || y_full)
    E_total = FormatAwareEnergy({q_eff, f})
    L_H = FormatEntropy({f})
    L_total = L_CE + α·L_KL + β_epoch·E_total + γ_epoch·L_H

    // Update all parameters jointly
    θ  ← θ  - η_w · ∇_θ L_total
    {q̃} ← {q̃} - η_q · ∇_{q̃} L_total
    {f̃} ← {f̃} - η_f · ∇_{f̃} L_total
```

```
        ϕ  ← ϕ  - η_d · ∇_ϕ L_total
    end
end

// Deployment: discretize formats and create multi-precision model
for each layer i:
    format_i = "FP" if σ(f̃_i) > 0.5 else "INT"
    bitwidth_i = round(q_min + (q_max - q_min) · σ(q̃_i))
```

## 5. Experimental Plan

### 5.1 Datasets and Models

Dataset	Model	Classes	Why
MNIST	Simple 5-layer CNN	10	Baseline comparison with EQAT
CIFAR-10	ResNet-18	10	Standard benchmark, moderate complexity
CIFAR-100	ResNet-18	100	Direct comparison with EQAT paper
CIFAR-100	MobileNetV2	100	Edge-relevant architecture
Tiny-ImageNet	ResNet-50	200	Scaling validation (optional)

### 5.2 Baselines

Baseline	Description
FP32	Full-precision (upper bound)
Uniform INT8	Standard 8-bit integer quantization
Uniform FP8 (E4M3)	Standard 8-bit floating-point
EQAT (Karkehabadi 2025)	Energy-aware mixed bit-width, INT only
HAWQ-V2	Hessian-based mixed precision
AdaQAT	Adaptive bit-width QAT
Ours (format only)	Learned format without input-adaptivity (ablation)
Ours (bit-width only)	Learned bit-width without format selection (ablation)
Ours (full AQUA)	**Joint format + bit-width + input-adaptive**

### 5.3 Key Experiments

Experiment 1: Main Results Table
Compare accuracy, energy, and average bit-width across all baselines on all datasets.
Expected finding: AQUA achieves better accuracy-energy Pareto front than all baselines.

Experiment 2: Format Discovery Visualization
For each layer, plot the learned format parameter f_i across training epochs.
Expected finding: Layers with outlier-heavy activations (typically early and attention-like layers) converge to FP, while layers with uniform distributions converge to INT. This is a key contribution.

Experiment 3: Input Difficulty Analysis
Visualize the difficulty scores for easy vs hard inputs. Show correlation between difficulty score and:
- Input clarity/noise level
- Classification confidence
- Per-input energy savings

Expected finding: 60-80% of inputs are classified as "easy" and run at significantly reduced precision.

Experiment 4: Ablation Study

Component	Accuracy	Energy
Bit-width only (like EQAT)	Baseline	Baseline
+ Format selection	+X%	-Y%
+ Input adaptivity	+X%	-Z%
Full AQUA	Best	Best

Experiment 5: Layer-Wise Analysis
Plot per-layer: (a) learned bit-width, (b) learned format (INT vs FP), (c) activation distribution statistics (kurtosis, outlier ratio). Show correlation between distribution statistics and format preference.

Experiment 6: Energy Breakdown
Show where energy savings come from:
- From reduced bit-widths
- From format selection (FP vs INT at same bit-width)
- From input-adaptive precision (easy inputs)

### 5.4 Training Details

Parameter	MNIST	CIFAR-10	CIFAR-100
Epochs	100	100	100
Optimizer (weights)	Adadelta (lr=1.0)	Adam (lr=0.001)	Adam (lr=0.001)
Optimizer (q, f)	Adam (lr=0.01)	Adam (lr=0.01)	Adam (lr=0.01)
α (KL weight)	0.95	0.8	0.7
β (energy weight)	0.001→0.01	0.001→0.01	0.001→0.01
γ (format entropy)	0.001→0.005	0.001→0.005	0.001→0.005
q_min, q_max	2, 8	2, 8	2, 8
Warmup epochs	10	10	10
GPU	NVIDIA A100	NVIDIA A100	NVIDIA A100

## 6. Expected Results and Contributions

### Predicted Outcomes

1. **Format discovery:** We expect ~40-60% of layers to prefer FP format and ~40-60% to prefer INT, with a clear pattern: first/last layers and layers with high kurtosis prefer FP; intermediate layers with near-Gaussian distributions prefer INT.

2. **Energy savings:** We expect 35-45% energy reduction on MNIST, 25-40% on CIFAR-10, and 5-15% on CIFAR-100 with accuracy improvements, significantly outperforming EQAT.

3. **Input adaptivity impact:** On average, 60-80% of test inputs will be classified as "easy" with significant precision reduction, providing additional 10-20% average energy savings.

### Hardware Design Implications

The format discovery results directly inform hardware designers:
- Which layers need FP arithmetic units vs INT units

- Potential for heterogeneous accelerators with mixed INT/FP datapaths
- Optimal memory format allocation for activations vs weights per layer

---

## 7. Paper Structure (6-page GLSVLSI format)

Section	Content	~Pages
1. Introduction	Problem, motivation, contributions	0.75
2. Related Work	QAT, mixed-precision, INT vs FP quantization	0.75
3. Methodology	Format selection, bit-width learning, input adaptivity, loss function, gradients, algorithm	2.0
4. Experiments	Setup, main results, format discovery, ablations, analysis	1.75
5. Conclusion	Summary, hardware implications, future work	0.5
References	~20-25 references	0.25

## 8. Timeline

Phase	Duration	Tasks
Week 1-2	Literature deep-dive	Read EQAT, FP8 papers, AdaQAT, SDP; identify exact positioning
Week 3-5	Core implementation	Build differentiable INT/FP quantization, format selection module
Week 6-7	Difficulty estimator	Implement and integrate input-adaptive module
Week 8-10	Experiments (MNIST, CIFAR-10)	Initial results, debug, tune hyperparameters
Week 11-13	Experiments (CIFAR-100, ablations)	Full experiments, analysis, visualizations
Week 14-16	Paper writing	Draft, figures, tables, review, submit

---

## 9. Key References to Cite

1. Karkehabadi & Sasan (2025) — EQAT, direct predecessor
2. Yang et al. (2017) — Energy estimation framework
3. Gong et al. (2020) — MP-NAS
4. Choi et al. (2018) — PACT
5. Gernigon et al. (2024) — AdaQAT
6. Tang et al. (2022) — Arbitrary Bit-width Networks
7. Micikevicius et al. (2022) — FP8 training
8. Shen et al. (2023) — FP8 quantization analysis
9. Zhang et al. (2023) — INT vs FP format comparison
10. Banner et al. (2018) — Scalable 8-bit training
11. Krishnamoorthi (2018) — Quantization whitepaper
12. Huang et al. (2024) — Adaptive Coreset Selection for QAT
13. Jacob et al. (2018) — Quantization for integer-arithmetic inference
14. He et al. (2016) — ResNet

## 10. Risk Assessment and Mitigation

Risk	Likelihood	Mitigation
Format interpolation is unstable during training	Medium	Use warmup for format entropy; start with both formats equally weighted
Difficulty estimator collapses (all easy or all hard)	Medium	Add diversity loss to encourage spread of difficulty scores
INT and FP produce nearly identical results at 8-bit	Low	Test at lower bit-widths (4-6 bit) where format differences are more pronounced
Overhead of difficulty estimator too high	Low	Design is <0.1% of model; share d_x across all layers
Reviewer concern: "trivial extension of EQAT"	Medium	Emphasize format discovery findings and hardware implications; show ablation proving format selection matters

---

## 11. Differentiation from EQAT Paper

Aspect	EQAT	AQUA (Ours)
Quantization format	INT only (fixed)	Learned per-layer INT vs FP
Input handling	Same precision for all inputs	Adaptive precision based on difficulty
Energy model	Format-agnostic	Format-aware (different INT/FP costs)
Loss function	CE + KL + Energy (3 terms)	CE + KL + Energy + Format entropy (4 terms)
Learned parameters	Bit-widths only	Bit-widths + formats + difficulty estimator
Hardware insight	"First/last layers need more bits"	"Which layers need INT vs FP" + difficulty-based savings
Novelty type	Mixed bit-width QAT	Mixed format + mixed bit-width + input-adaptive QAT