ACL 2026
SAN DIEGO
JULY 2–7

Unlocking the Edge deployment and ondevice acceleration of multi-LoRA enabled one-for-all foundational LLM

Sravanth Kodavanti, Sowmya Vajrala, Srinivas Miriyala, Utsav Tiwari, Uttam Kumar, Utkarsh Kumar Mahawar, Achal Pratap Singh, Arya D, Narendra Mutyala, Vikram Nelvoy Rajendiran, Sharan Allur, Euntaik Lee, Dohyoung Kim, HyeonSu Lee, Gyusung Cho, JungBae Kim

June 4th 2026

Edge deployment requires a fundamental shift from dynamic recompilation to immutable inference graphs.

Server-Side Architecture	On-Device Architecture (The Bottleneck)
Constraint: Virtually unlimited memory and compute.	**Constraint:** Stringent memory limits and strict runtime performance guarantees.
Execution: Models dynamically recompiled on the fly.	**Execution:** Operations locked into a **Frozen Inference Graph.**
Adaptation: Traditional LoRA merging is statically compiled during or post-training.	**Adaptation Disconnect:** Hardware resists traditional LoRA merging, creating massive latency and memory overhead when switching tasks.

Engineering a 'One-for-All' system demands optimization across memory, compute, and decoding

Memory & State Optimization	Compute & Architectural Optimization	Decoding Acceleration
LoRA-as-Input Bypassing the frozen graph barrier by treating adapters as runtime inputs.	**Linear-to-Convolutional Transforms** Unlocking embedded NPU hardware accelerators designed for vision.	**Concurrent Token Generation (CTG)** Shared KV-caches outputting parallel stylistic streams in one pass.
INT4 Quantization Mixed-precision compression to INT4 (weights) and INT8 (activations).	**Attention Parallelism** Decomposing multi-head sequential dependencies for parallel execution.	**Dynamic Self-Speculative Decoding (DS2D)** Tree-based branching that eliminates the need for secondary draft models.

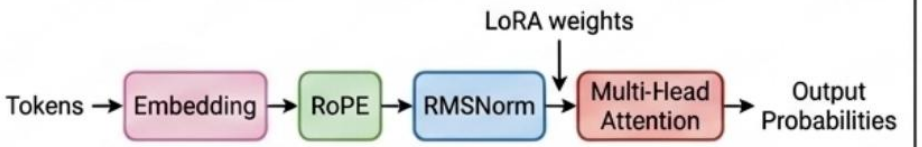
Treating LoRA weights as runtime inputs eliminates recompilation and memory bloat

1 **Traditional Approach:**	[Frozen Graph]	+	[Static LoRA Merge]	→ System Crash / OOM
2 **LoRA-as-Weight-Adjustment:**	[Frozen Graph]	+	[Dynamic Weight Updates]	→ Increased Latency & Recompilation
3 **LoRA-as-Prefix:**	[Frozen Graph]	+	[LoRA as Prefix Tokens]	→ Context Window Bloat & Reduced Accuracy
4 **Proposed Architecture:**	[Frozen Graph (with Placeholders)]	+	[Tokens + LoRA Weights]	→ Dynamic Task Switching

By assuming **uniform LoRA dimensions** across tasks, specific LoRA weights are injected directly as runtime inputs alongside text tokens, bypassing the need to alter the immutable graph.

Performance Comparison (1B Model, GS24)		
Masking Approach	894 MB Size	75ms Latency
LoRA-as-Input	686 MB Size	52ms Latency

Graph 1: LoRA weights as inputs



Structural reparameterization aligns LLM architecture with embedded hardware strengths

Linear-to-Convolutional Transformation	Attention Parallelism & Graph Simplification
The Concept: Dense linear layers are mathematically reparameterized as 1×1 convolutions. **The Hardware Advantage:** Modern mobile NPUs and GPUs excel at computer vision tasks. This transformation allows the LLM to exploit highly optimized convolution kernels, Winograd algorithms, and pipelined execution units, drastically reducing energy overhead.	**The Concept:** Multi-head attention (MHA) is decomposed into parallel single-head attention (SHA) paths. **The Hardware Advantage:** Enables independent, head-wise scheduling across heterogeneous compute units. **Graph Compilation:** Constant folding (precomputing static tensors) and operator fusion (merging linear/activation layers) minimize NPU kernel launch overhead.

Concurrent Token Generation (CTG) yields multiple stylistic responses in a single inference pass

6x Latency Reduction

174ms sequential drop to 63ms concurrent generation.

The Problem: Personalization features (e.g., 'Smart Reply') require up to 8 diverse tone prompts. Running a LoRA-enabled LLM 8 times sequentially increases latency 8x.

↓

Step 1: Perform a single prefill pass to establish a shared KV-Cache.

↓

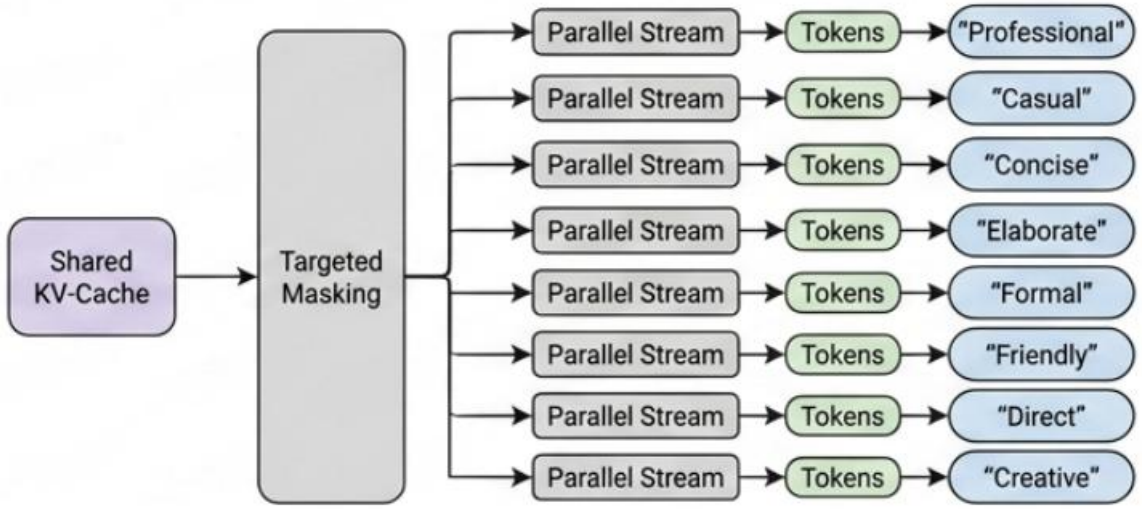
Step 2: Divide the KV-Cache into 8 isolated segments using targeted masking.

↓

Step 3: Alter first-token sampling across 8 parallel streams.

⬇

Result: Generate 8 distinct response styles simultaneously on the exact same frozen inference graph.



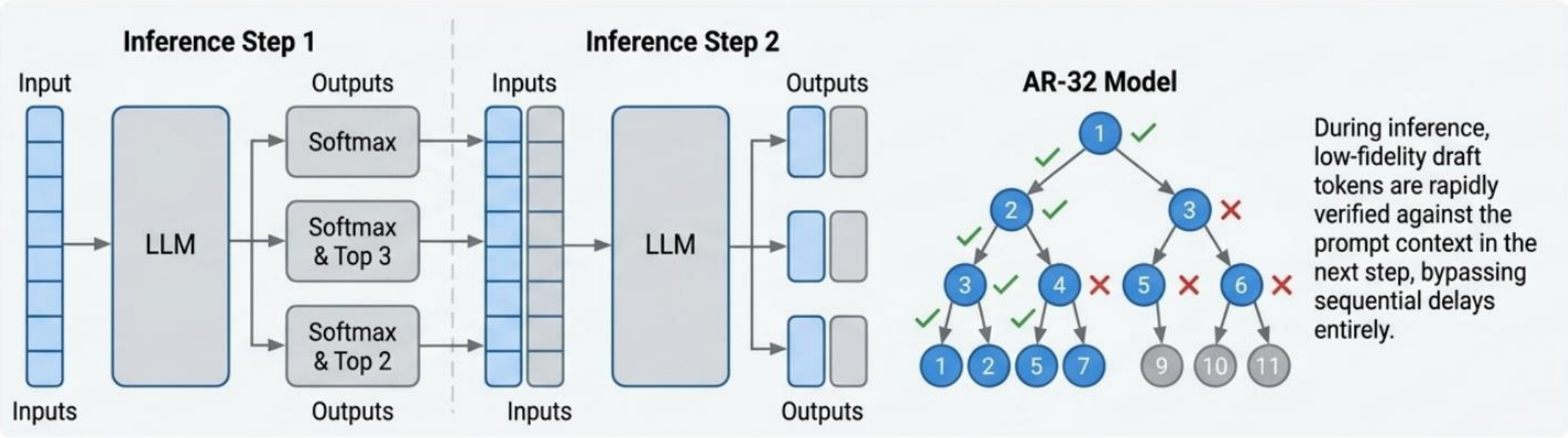
Dynamic Self-Speculative Decoding (DS2D) achieves semi-autoregressive generation without draft models

The Constraint: Standard speculative decoding requires secondary draft models, completely violating strict edge memory limits.

Step 1: Prefix Tuning
Task-specific forecast embeddings are appended directly to the prompt embedding.

Step 2: Semi-Autoregressive Output
The frozen LLM is tricked into generating multiple tokens simultaneously. The first token remains high-fidelity; the subsequent tokens serve as self-generated "drafts".

Step 3: Tree-Based Verification
During inference, low-fidelity draft tokens are rapidly verified against the prompt context in the next step, bypassing sequential delays entirely.



DS2D unlocks unprecedented tokens-per-second throughput on the Galaxy S25 Ultra (3B Model)

Use Case	Prefill (w/o DS2D → w/ DS2D)	Decode Time (w/o DS2D → w/ DS2D)	Tokens/sec
Correction	208ms → **50ms**	22.3ms → **19.9ms**	**44.84 t/s**
Summarization	797ms → **52ms**	24.1ms → **18.9ms**	**41.46 t/s**
AI Brief	404ms → **52ms**	37.7ms → **18.9ms**	**26.49 t/s**

Key Takeaway: DS2D drives up to a **2.3x** speedup in decode time, establishing real-time semi-autoregressive capabilities entirely on the mobile NPU.

Rigorous INT4 quantization achieves 3x model compression while preserving high-fidelity multilingual accuracy

The Compression Benchmark

3x ROM Footprint Reduction

Original FP16 LLM + LoRA = 1800MB + 120MB

Optimized INT4 LLM + LoRA = 600MB + 120MB

Mechanism: Quantization-Aware Training (QAT) with Int8 activations and Int4 weights.

Multilingual Fidelity Matrix

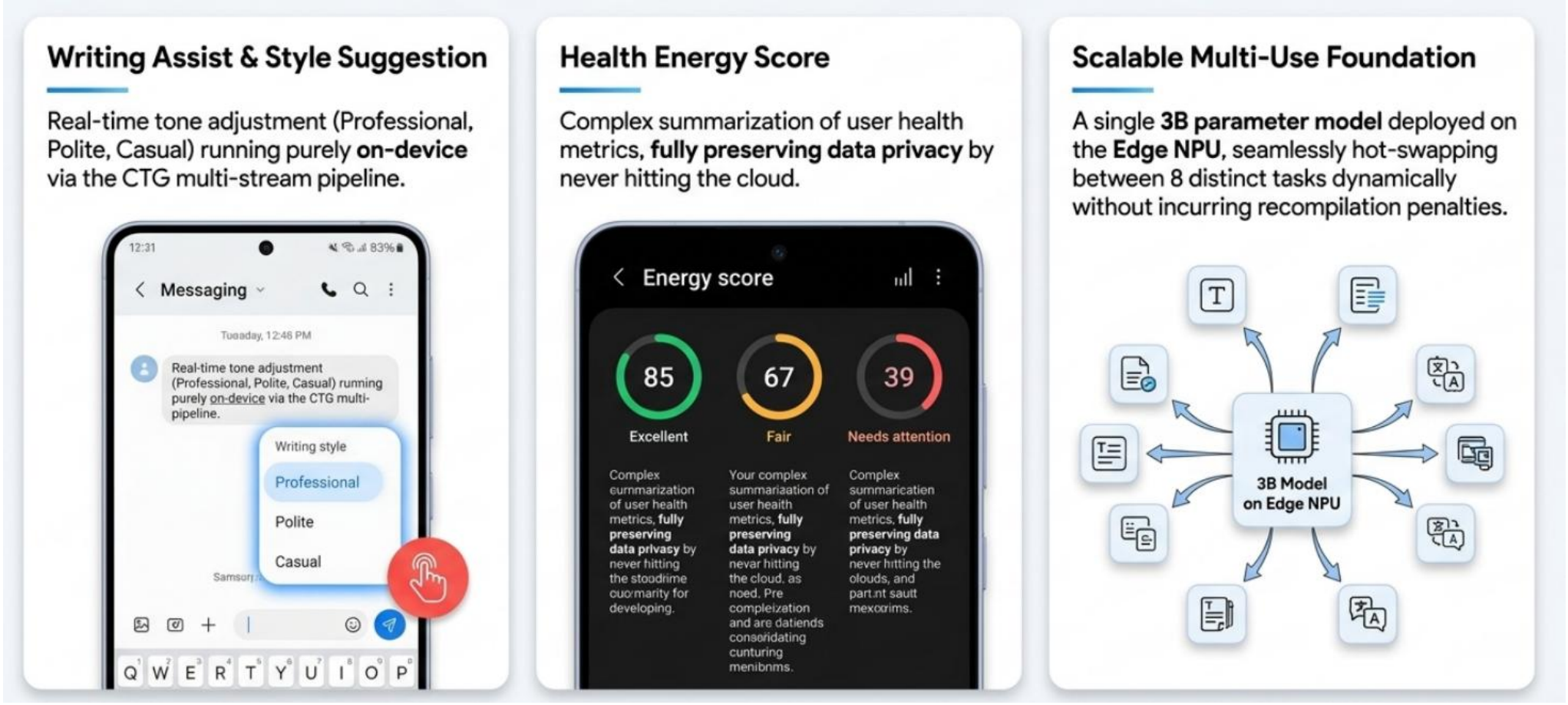
Task Accuracy (1B Model, GS24)

- Correction (English): 97.6%
- Style (Spanish): 98.6%
- Smart Reply (Italian): 100.7%

Relative Accuracy (3B Model, GS25, vs FP32 Baseline)

Evaluated via G-Eval. Across 9 languages (including Korean, German, and Chinese), the QAT on-device models perform at 96% to 109% relative accuracy compared to uncompressed baselines.

Feasibility Proven: Commercialization within the Samsung Galaxy S24,S25 'Galaxy AI' ecosystem



A definitive blueprint for the future of sustainable, on-device Generative AI

The Hardware Reality By reimagining LLMs through the strict lens of NPU constraints—embracing Immutable Graphs and 1x1 Convolutions—mobile inference is no longer bottlenecked by architecture.
The Software Reality LoRA-as-input and DS2D mathematically prove that advanced, multi-task AI capabilities do not strictly require dynamic recompilation or reliance on cloud servers.

This framework elevates multi-task foundational LLMs from a theoretical exercise to a fully commercialized, cloud-independent reality on edge devices.

THANK YOU!

Read full paper