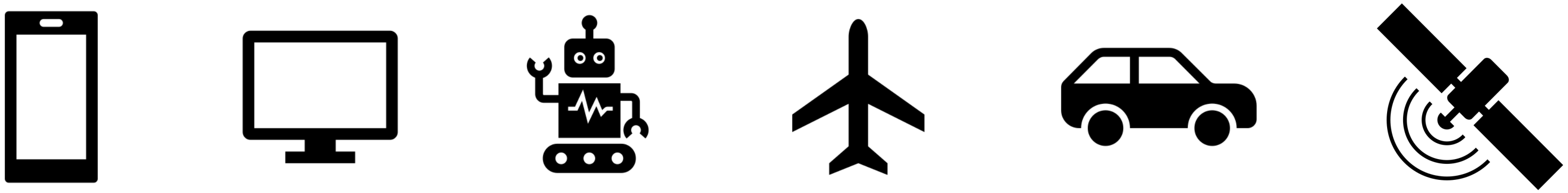
端侧大模型和智能体：挑战和尝试
LLM-powered Mobile Devices

Mengwei Xu (徐梦炜)
Beijing University of Posts and Telecommunications
https://xumengwei.github.io



Aren't they smart.. Already?

- Yes, to a certain extent.

DNN-embedded mobile apps

- Increased by almost 10x (2018 to 2021)[1,2]
- Downloaded billions of times in one year
- Include almost every high-popularity app
- Up to 200+ DNNs in a single app[3]

[1] Mengwei Xu, et al. "A First Look at Deep Learning Apps on Smartphones". In WWW 2019
[2] Mario Almeida, et al. "Smart at what cost? Characterising Mobile Deep Neural Networks in the wild". In IMC 2021.
[3] Through offline communication with application developers.

Aren’t they smart.. Already?

- Yet, not even close to our expectation.

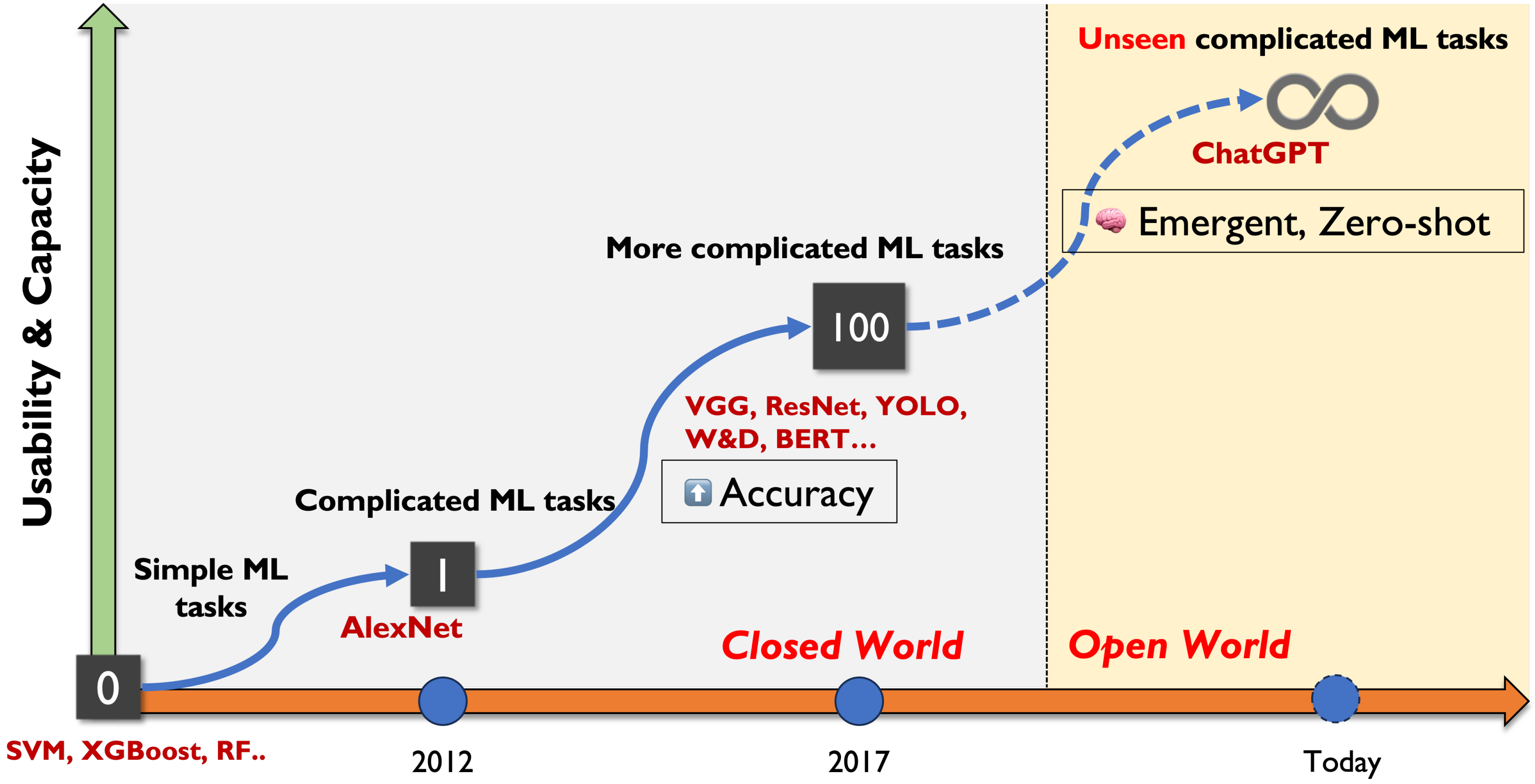
“AI is a mirror, reflecting not only our intellect, but our values and fears."

A cool example of "smart device"

- *Comprehend physically*
- *Proactively sense, plan, and action*
- *Retrieval from Internet or Remote DB*
- *Predict the future (multimodal)*
- *Instruction following*
- *Fast response*

The opportunity: LLM

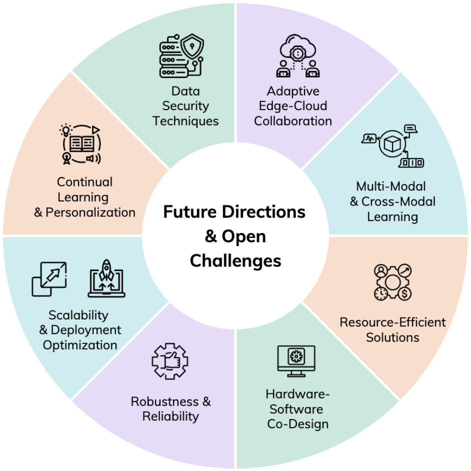
- To bring mobile devices the "next-level" intelligence



- Comprehend human language
- Zero-shot & in-context learning
- Multimodal alignment and input/output
- Reasoning & Planning
- Long context

On-device LLM is crucial

- On-device LLMs handle language tasks in a way that is ..
  - ✓ cost-efficient (important, obviously)
  - ✓ more available (even w/o network)
  - ✓ faster (not always)
  - ✓ privacy-preserving (very important, LLMs can leverage almost every bits of local data)



- LLMs on devices does not obviate mega-scale LLMs on clouds!
  - Creating music/poetry, solving math problems, etc.

[1] Jiajun Xu, et al. "On-Device Language Models: A Comprehensive Review". In preprint'24.

On-device LLM is crucial

- We already have a mobile device that can function with high intelligence!

A *mobile device* that can comprehend, reason, and plan without a cloud!

So, what's unique to mobile LLM?

(compared to traditional DNN-powered apps)

Workload:	fragmented tasks	→	a unified agent
OS:	model-agnostic	→	LLM-native
Hardware:	heterogeneous H/W	→	DSA-dominated

So, what's unique to mobile LLM?

(compared to traditional DNN-powered apps)

Workload:	fragmented tasks	→	a unified agent
OS:	model-agnostic	→	LLM-native
Hardware:	heterogeneous H/W	→	DSA-dominated

The source of research/industrial opportunities

Call for full-stack design

- Our response: **agent-model-runtime-OS** co-design

Agent

Device control and GUI agents **testbed** [LlamaTouch, UIST'24], **datasets** [DroidCall, preprint'24][SHORTCUTSBENCH, ICLR'25], and **privacy enhancements**[SILENCE, NeurIPS'24]

Model

A training-from-scratch, fully-reproducible **SLM family** [PhoneLM, preprint'24], Any-to-any modality **mobile foundation model** [M4, MobiCom'24] , and **Federated LLM** techniques [FwdLLM, ATC'24][AdaFL, MobiCom'23] [FeS, MobiCom'23]

Runtime

Acceleration through **NPU** [llm.npu, ASPLOS'25], **SpecDecoding** [LLMCad, TMC'24] , **Sparsity** [EdgeMoE, TMC'25], **Early Exiting** [Recall, preprint'24], etc

OS

LLMaaS Context Management [LLMS, preprint'24] and **QoS** [ELMS, preprint'24]

An e2e demo

DroidCall: A Dataset for LLM-powered Android Intent Invocation

Weikai Xie[1] Li Zhang[1] Shihe Wang[1] Rongjie Yi[1] Mengwei Xu[1]

Agent

PhoneLM: an Efficient and Capable Small Language Model Family through Principled Pre-training

Rongjie Yi[1] Xiang Li[1] Weikai Xie[1] Zhenyan Lu[1] Chenghua Wang[1] Ao Zhou[1] Shangguang Wang[1]
Xiwen Zhang[2] Mengwei Xu[1]

Model

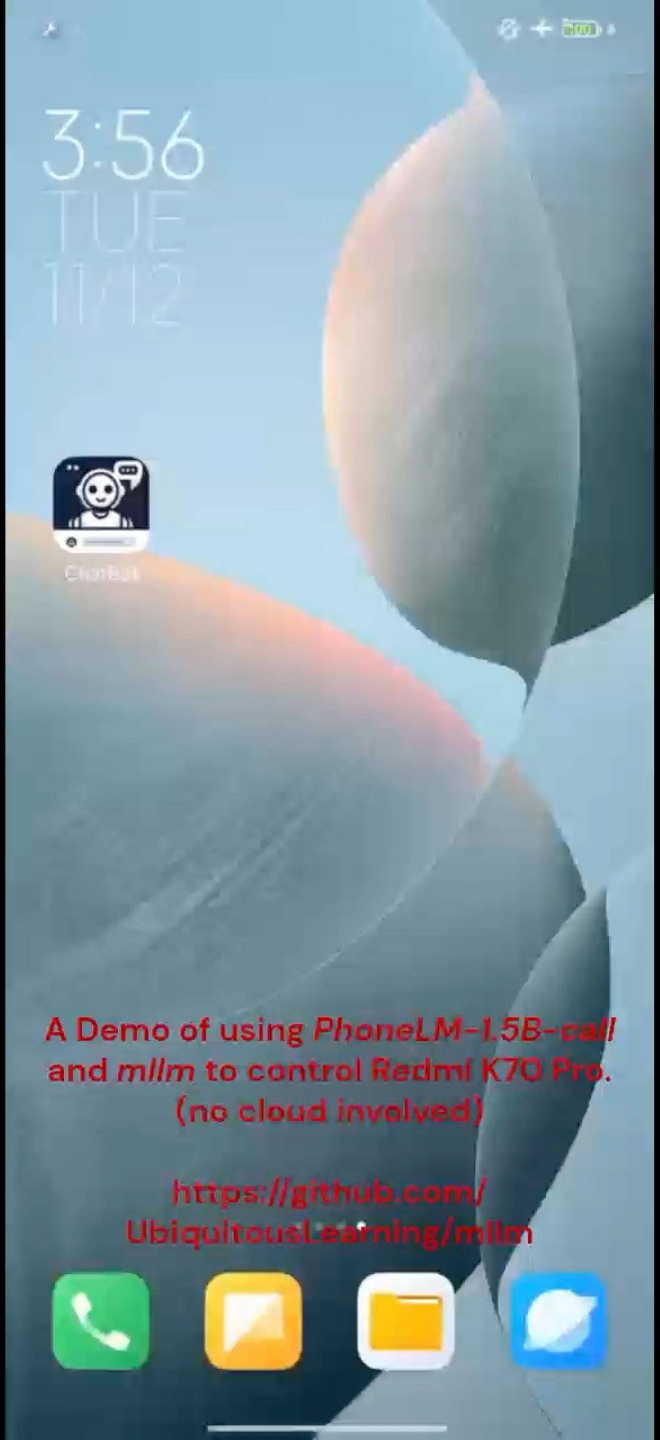
Code | Issues 9 | Pull requests 2 | Actions | Security 2 | Insights

mllm Public — Watch 17 | Fork 58 | Starred 516

Empowering 1000 tokens/second on-device LLM prefilling with mllm-NPU

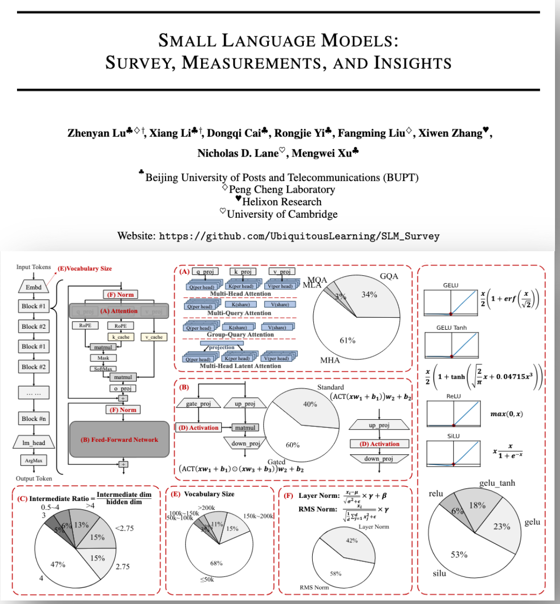
Daliang Xu[♦], Hao Zhang[◇], Liming Yang[♦], Ruiqi Liu[♦], Gang Huang[♦], Mengwei Xu[◇*], Xuanzhe Liu[♦]
[♦]Peking University, [◇]Beijing University of Posts and Telecommunications

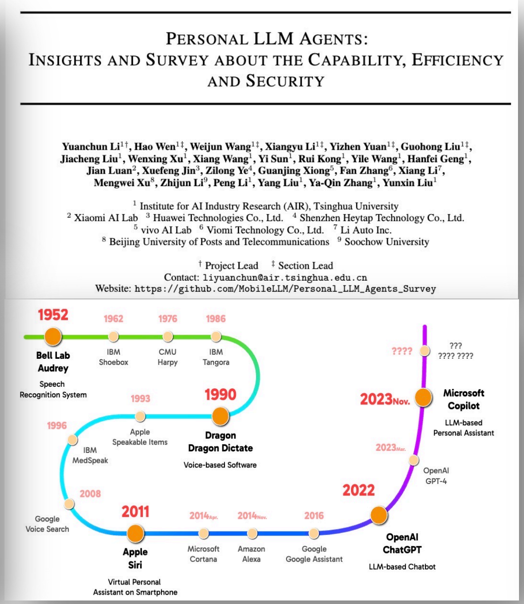
Runtime

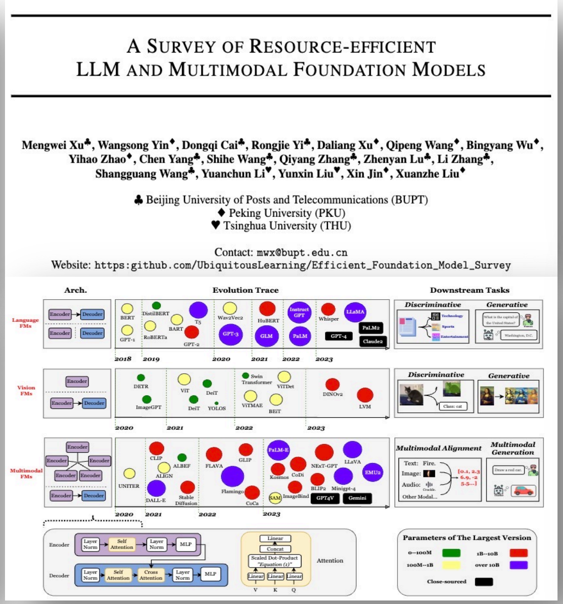


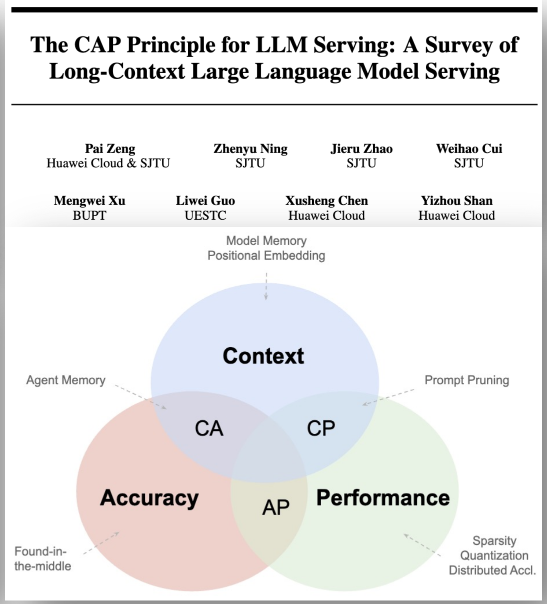
Call for full-stack design

- Our response: **agent-model-runtime-OS** co-design









[1] “Small Language Models: Survey, Measurements, and Insights”, Zhenyan Lu, et al.
[2] “Personal LLM Agents: Insights and Survey about the Capability, Efficiency and Security”, Yuanchun Li, et al.
[3] “A Survey of Resource-efficient LLM and Multimodal Foundation Models”, Mengwei Xu, et al.
[4] “The CAP Principle for LLM Serving: A Survey of Long-Context Large Language Model Serving”, Pai Zeng, et al.

So, what's unique to mobile LLM?
(compared to traditional DNN-powered apps)

Workload:	fragmented tasks	→	a unified agent
OS:	model-agnostic	→	LLM-native
Hardware:	heterogeneous H/W	→	DSA-dominated

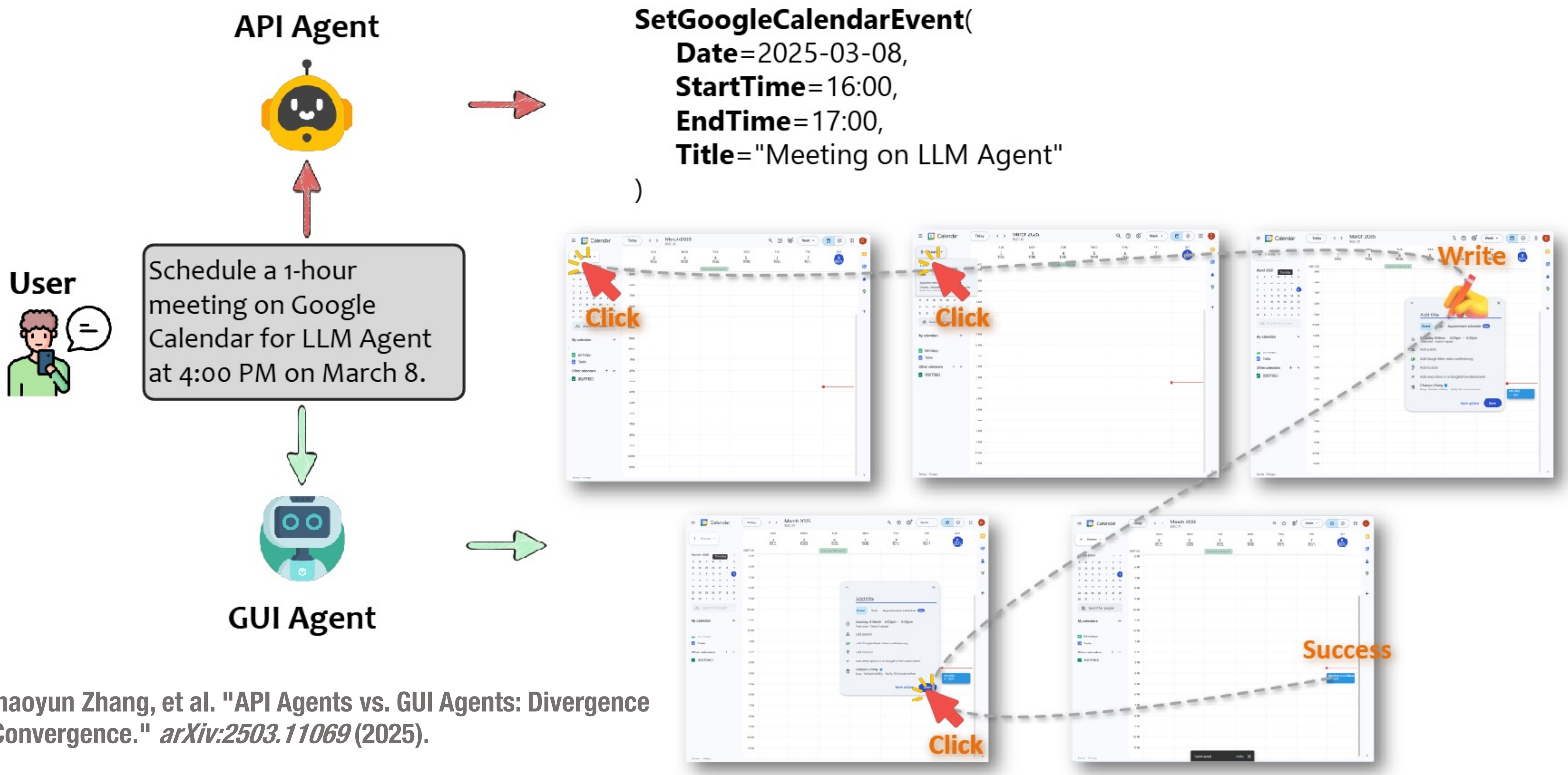
How to build a capable, generalized, and personalized mobile agent?

Our vision of an agent

Making electronic devices (smartphones, robots, cars, IoTs, satellites) more accessible to anyone (those with cognitive difficulties) at anytime (when driving)

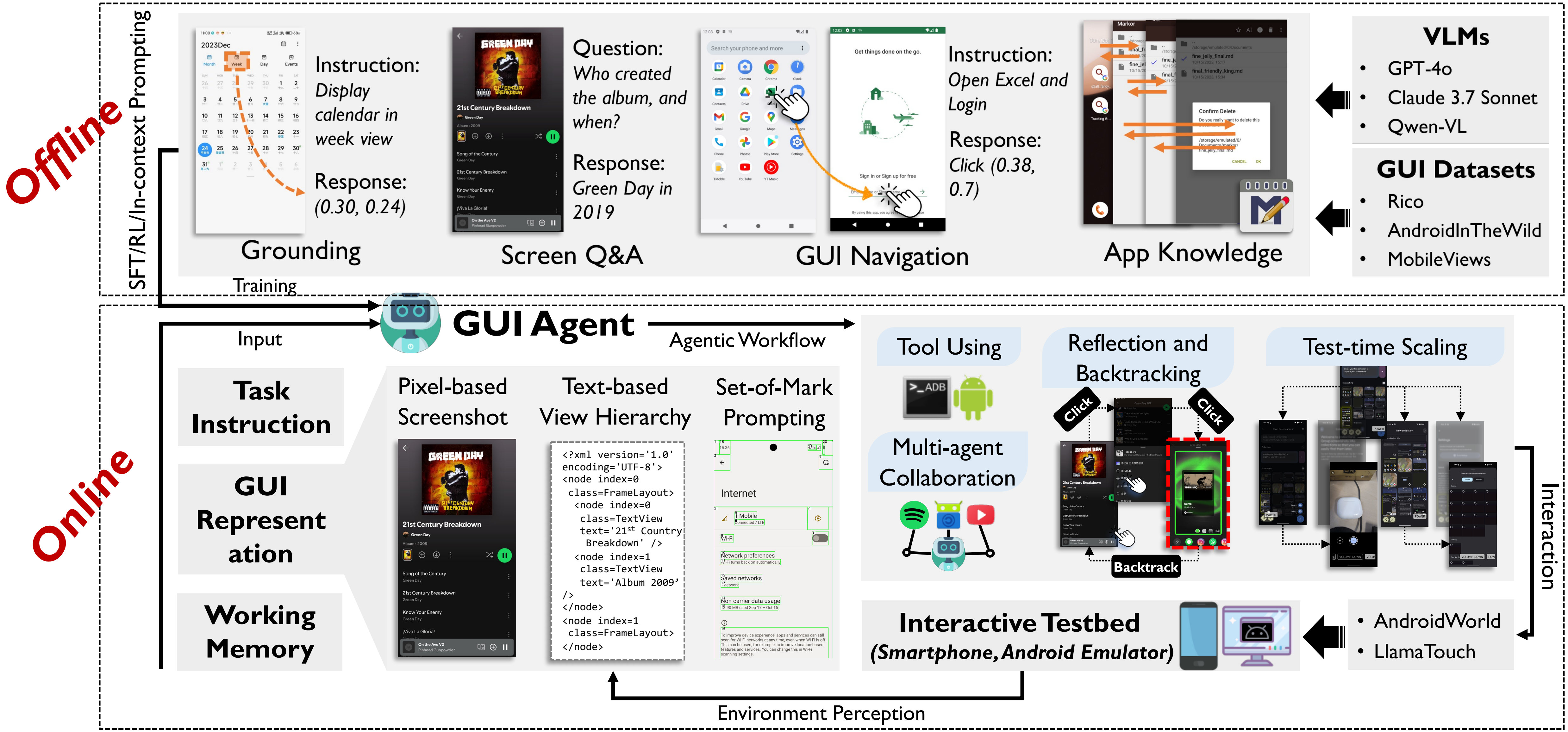
- *Comprehend physically*
- *Proactively sense, plan, and action*
- *Retrieval from Internet or Remote DB*
- *Predict the future (multimodal)*
- *Instruction following*
- *Fast response*

General approaches: API (MCP) vs. GUI

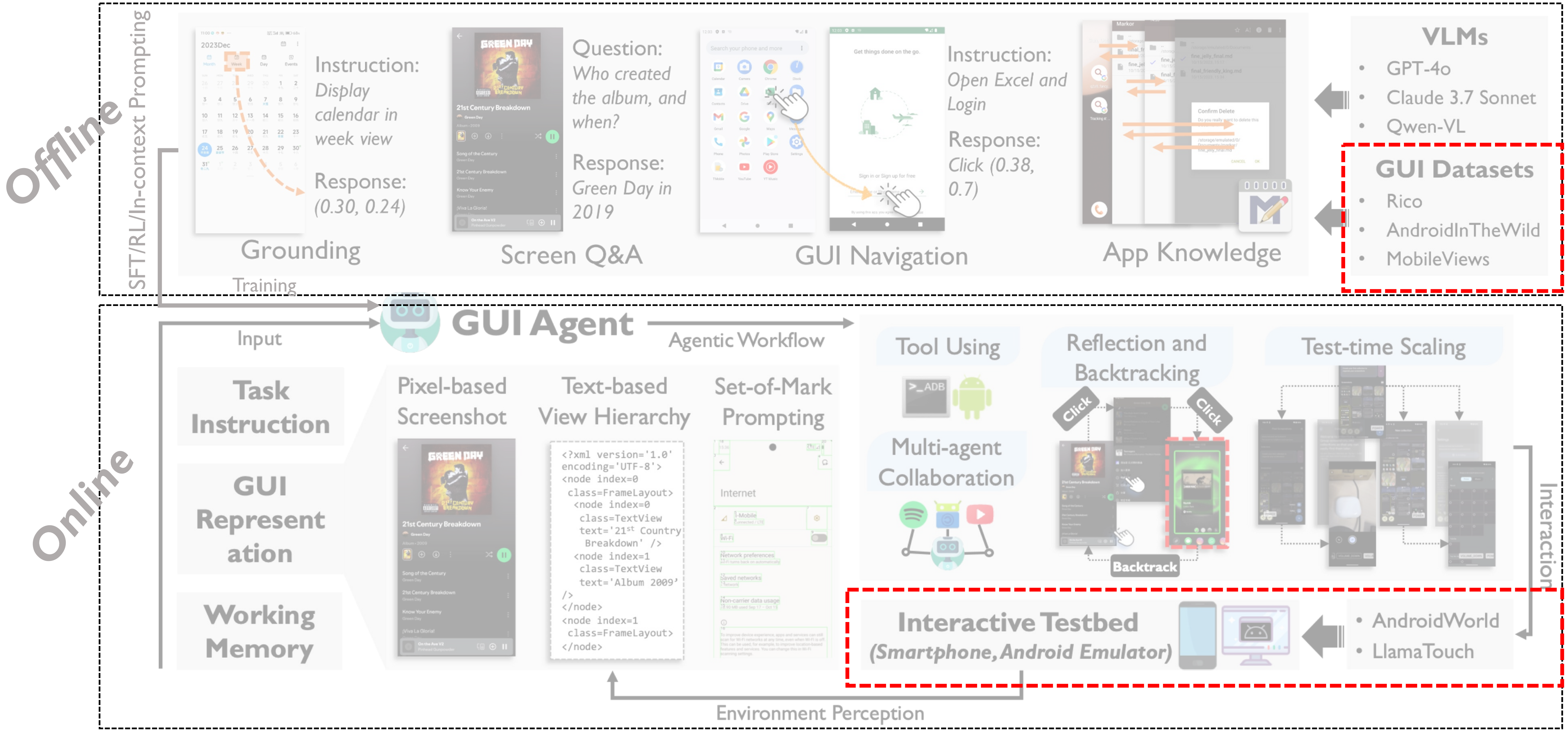


[1] Chaoyun Zhang, et al. "API Agents vs. GUI Agents: Divergence and Convergence." *arXiv:2503.11069* (2025).

GUI Agent: Status Quo



GUI Agent: Status Quo



Collecting mobile GUI datasets, with good quality and quantity

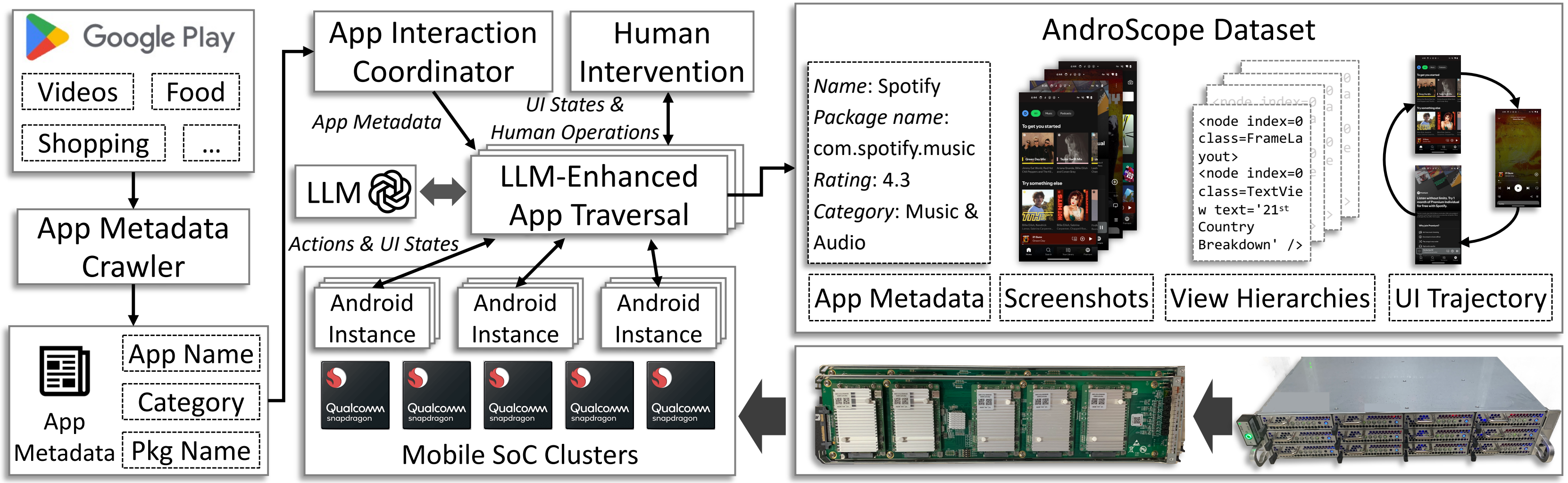
[arxiv'24] MobileViews: A Large-Scale Mobile GUI Dataset

Data at

https://huggingface.co/datasets/mllmTeam/MobileViews

MobileView: largest (>1M) open mobile GUI dataset

- LLM-enhanced app traversal + SoC Clusters



- Handling log-in actions through LLM
- 2x 2U SoC clusters, 120 Snapdragon 865 in total, further virtualized

- **Largest coverage**
  - **>1M UIs**
  - **>20K apps**
- **Bilingual apps**
- **Both UI and VH**
- **Action traces**

Mobile Screen Dataset	Scale		Content Diversity					Data Collection	
	Apps	Unique Screens	App Metadata	Screenshot-VH Pair	UI Trajectory	Chinese UI	Different Resolution	Automation	Hardware
Rico (Deka et al., 2017)	9,772	63,370	✓	✓	✓	✗	✗	✗	Physical Devices
PixelHelp (Li et al., 2020a)	4	187	✗	✓	✓	✗	✗	✗	Emulators
Screen2words (Wang et al., 2021)	6,269	22,417	✓	✓	✗	✗	✗	✗	N/A
ScreenQA (Baechler et al., 2024)	N/A	35,352	✓	✓	✗	✗	✗	✗	N/A
META-GUI (Sun et al., 2022)	11	24,825	✗	✓	✓	✗	✗	✗	Physical Devices
DroidTask (Wen et al., 2024)	13	362	✗	✓	✓	✗	✗	✗	Emulators
AITW (Rawles et al., 2023)	357	2,282,533	✗	✗	✓	✗	✓	✗	Emulators
LlamaTouch (Zhang et al., 2024b)	57	3,281	✗	✓	✓	✗	✗	✗	Emulators
GUIScope	**30,037**	**1,213,866**	✓	✓	✓	✓	✓	✓	**SoC clusters**

[1] Longxi Gao, et al. "MobileViews: A Large-Scale Mobile GUI Dataset". In preprint'24.

Benchmarking mobile GUI agents, properly and efficiently

[UIST'24] LlamaTouch: A Faithful and Scalable Testbed for Mobile UI Task Automation

Code at https://github.com/LlamaTouch/LlamaTouch

LlamaTouch: mobile GUI testbed

- Existing approaches: human/LLM eval.; step-wise action match
- Our approach: critical states matching

Match Type	State Type	Primitive	Keyword	Use Case
Fuzzy match	UI state	Screen info	fuzzy<-1>	Check if the contents on two screens are approximately identical.
		Textbox	fuzzy<n>	Check if the content of the target textbox is semantically similar to the content of the original textbox<n> in the ground-truth UI.
Exact match		Activity	activity	A coarse-grained approach to determining if two UIs represent the same functional screen in an application.
		UI component	exact<n>, exclude<n>	Check if the UI component is exactly identical to the UI component<n>, or does not occur, in the ground-truth UI.
	System state	(Un)installation	installed<app>, uninstalled<app>	Check if the target application named "app" has been successfully installed/uninstalled.
	Action	Action	click<n>, type<input_text>	Check if two actions and their parameters are identical.

Two types of matching primitives: Exact match and fuzzy match

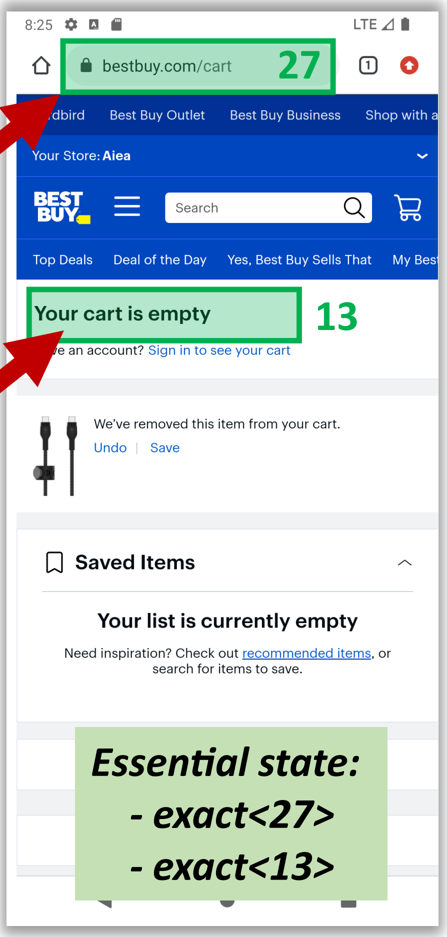
Table 7: Accuracy (Acc. %) of different evaluation approaches among all successful tasks in human validation.

Mobile Agent	Step-wise action match	LCS action match	LlamaTouch	Human
	Acc.	Acc.	Acc.	# succ
AutoUI	0.00	0.00	77.78	9
AutoDroid	0.00	0.00	73.91	69
AppAgent	0.00	3.03	93.94	33
CoCo-Agent	0.00	0.00	70.00	10
Average	0.00	0.76	78.91	30

High eval. acc

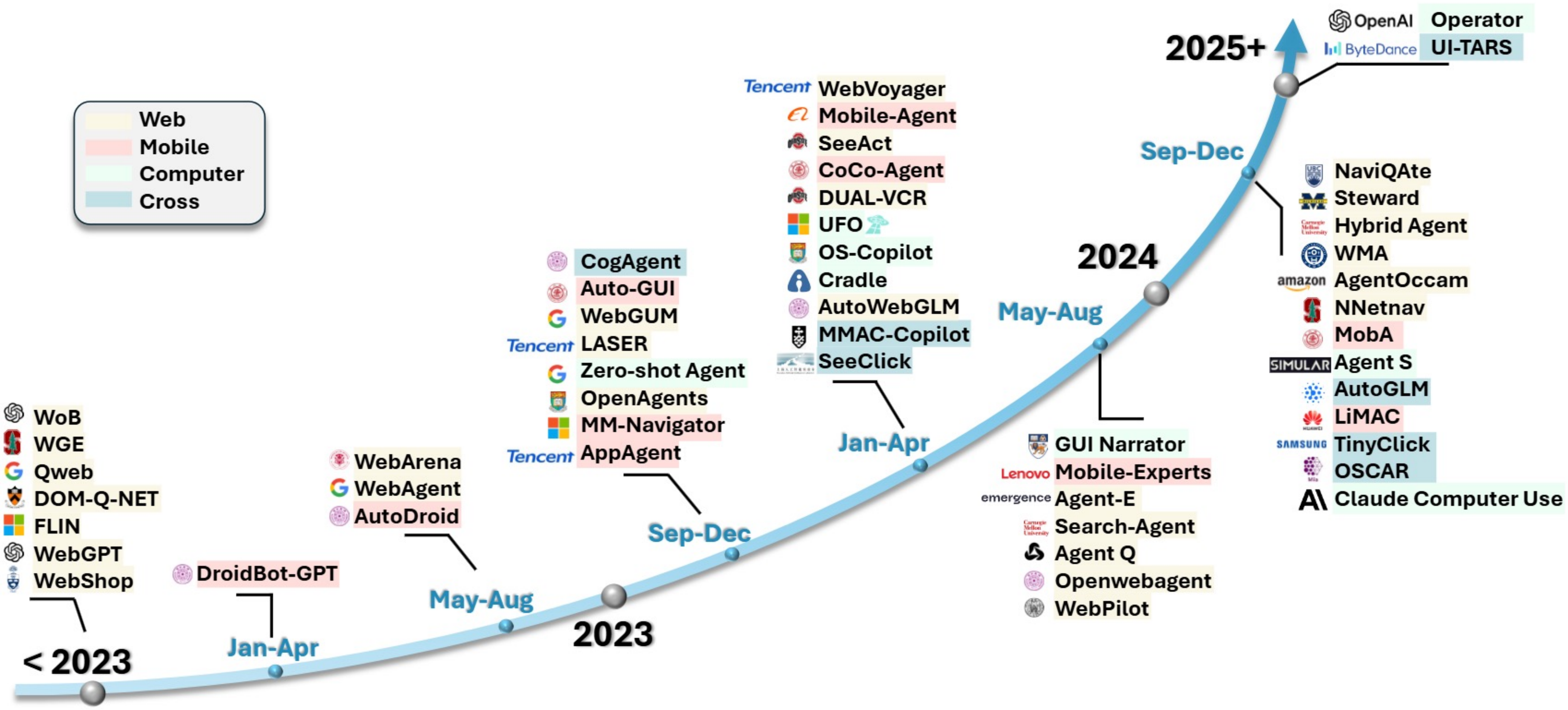
Task: Empty my shopping cart on BestBuy

- Exact match on the URL: “bestbuy.com/cart”
- Exact match on the UI text “Your cart is empty”
- All others (actions, UI components) are omitted.



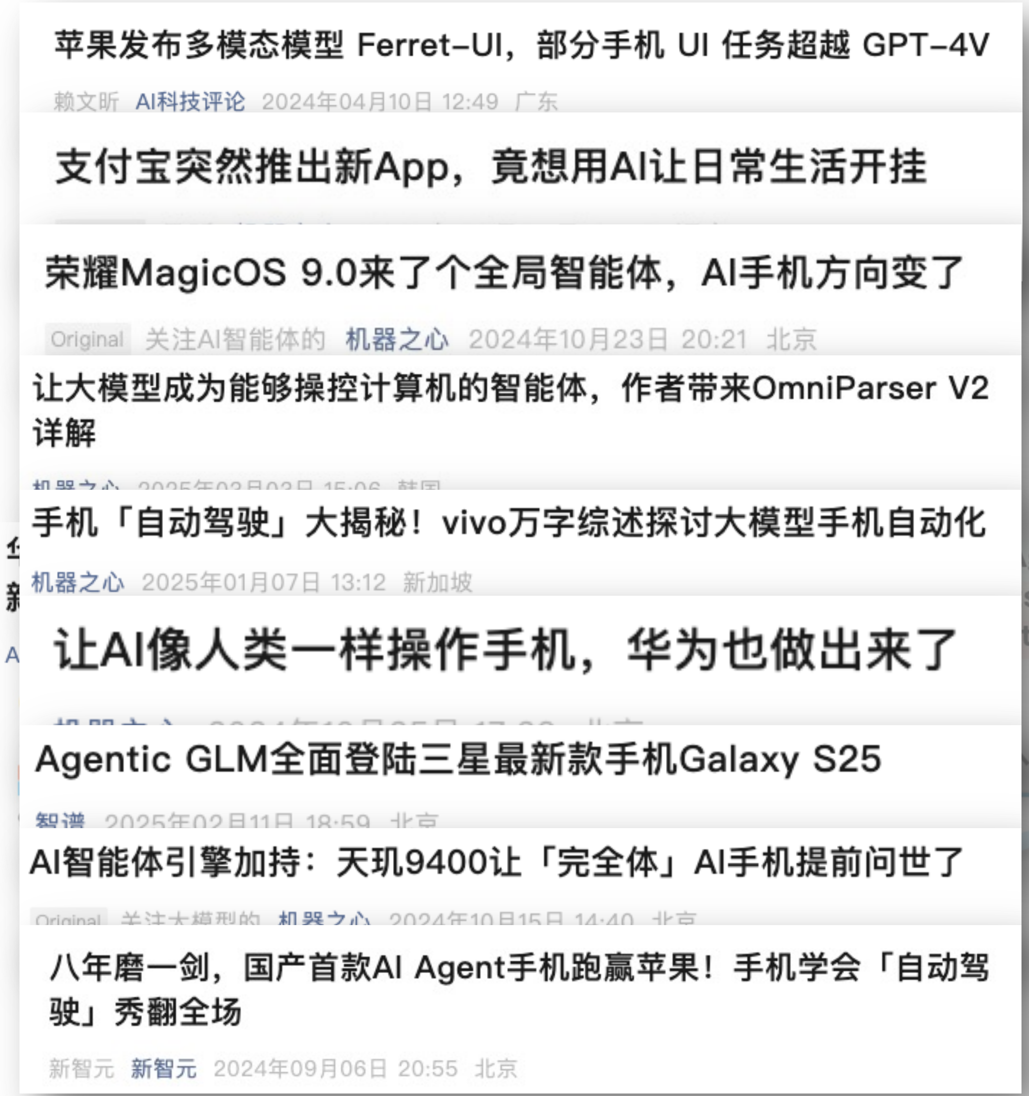
[1] Li Zhang, et al. “LlamaTouch: A Faithful and Scalable Testbed for Mobile UI Task Automation”. In UIST’24.

GUI Agent: Status Quo

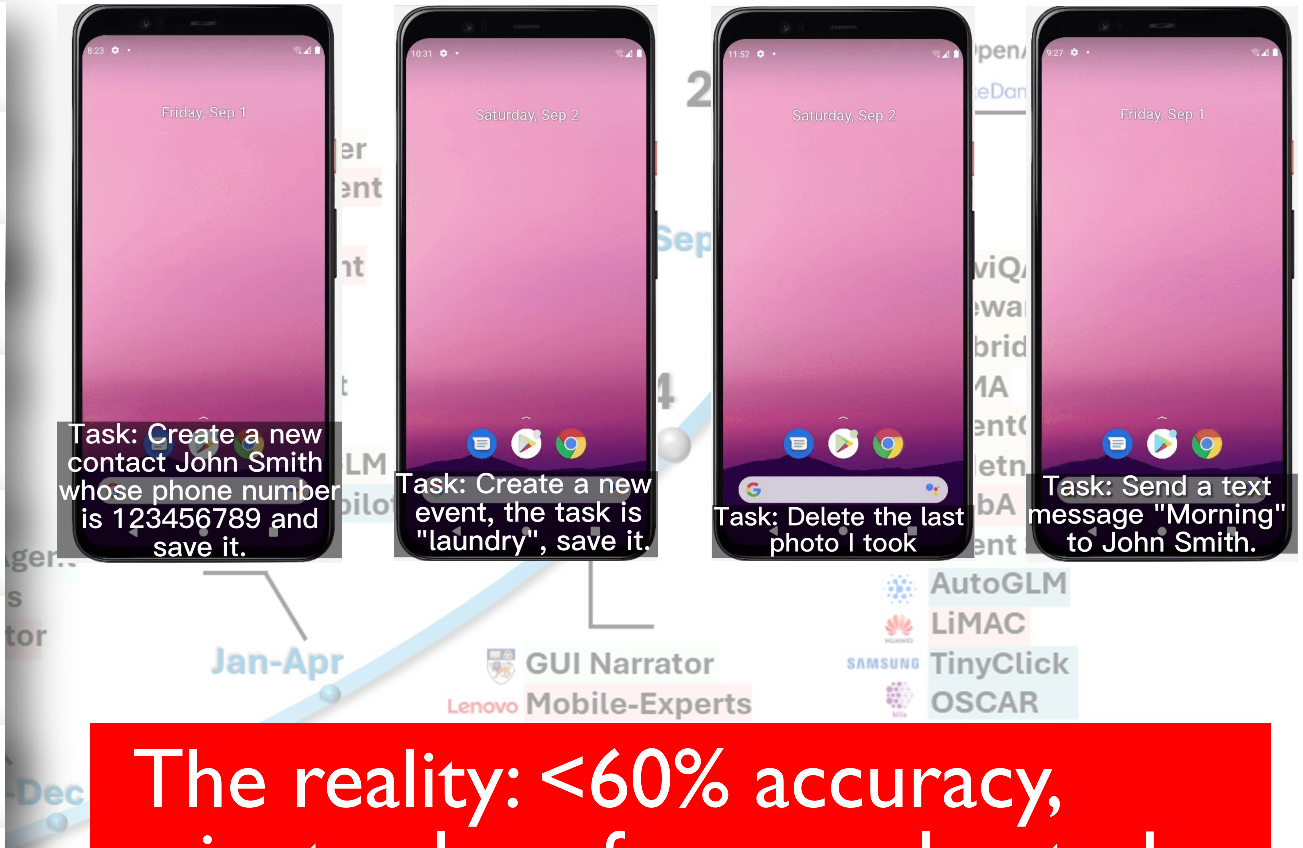


[1] Chaoyun Zhang, et al. "Large Language Model-Brained GUI Agents:A Survey." *arXiv:2411.18279* (2024).

GUI Agent: Status Quo


苹果发布多模态模型 Ferret–UI，部分手机 UI 任务超越 GPT–4V
赖文昕 AI科技评论 2024年04月10日 12:49 广东
支付宝突然推出新App，竟想用AI让日常生活开挂
荣耀MagicOS 9.0来了个全局智能体，AI手机方向变了
Original 关注AI智能体的 机器之心 2024年10月23日 20:21 北京
让大模型成为能够操控计算机的智能体，作者带来OmniParser V2详解
手机「自动驾驶」大揭秘！vivo万字综述探讨大模型手机自动化
机器之心 2025年01月07日 13:12 新加坡
让AI像人类一样操作手机，华为也做出来了
Agentic GLM全面登陆三星最新款手机Galaxy S25
AI智能体引擎加持：天玑9400让「完全体」AI手机提前问世了
八年磨一剑，国产首款AI Agent手机跑赢苹果！手机学会「自动驾驶」秀翻全场
新智元 新智元 2024年09月06日 20:55 北京

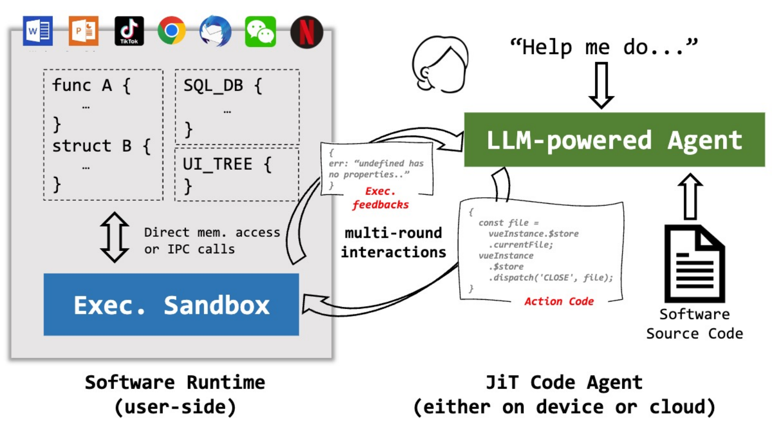
[1] sources: https://autodroid-sys.github.io/

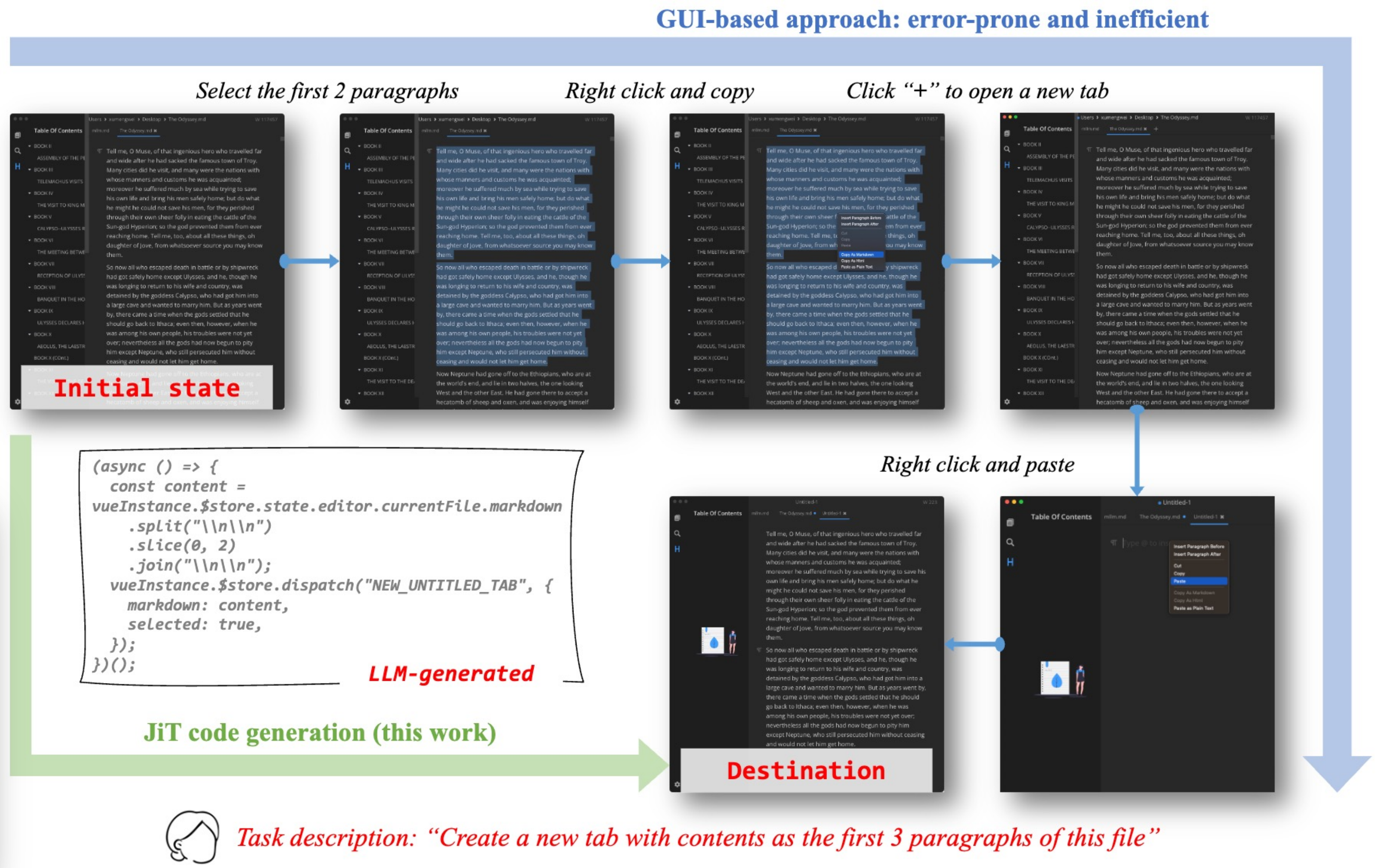


The reality: <60% accuracy, minutes-long for complex tasks.

A Third Path? Codegen is all you need

Just-in-time code generation and in-app execution





[1] Mengwei Xu. "Every Software as an Agent: Blueprint and Case Study" *arXiv:2502.04747* (2025).

So, what's unique to mobile LLM?

(compared to traditional DNN-powered apps)

Workload:	fragmented tasks	→	a unified agent
OS:	model-agnostic	→	LLM-native
Hardware:	heterogeneous H/W	→	DSA-dominated

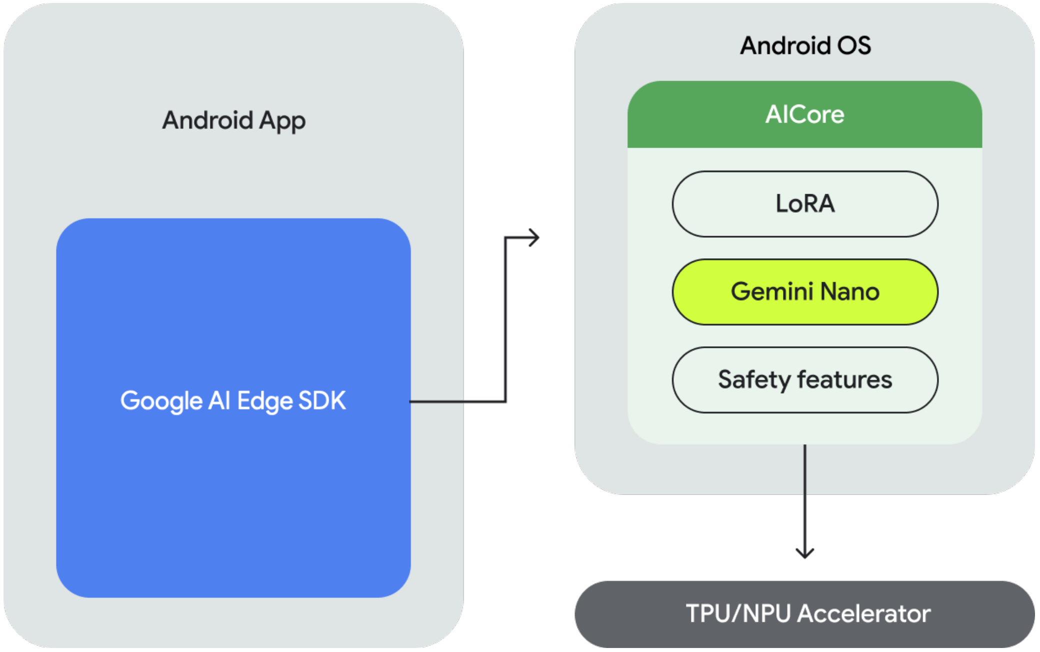
How should OS better serve/manage device-wise LLM requests?

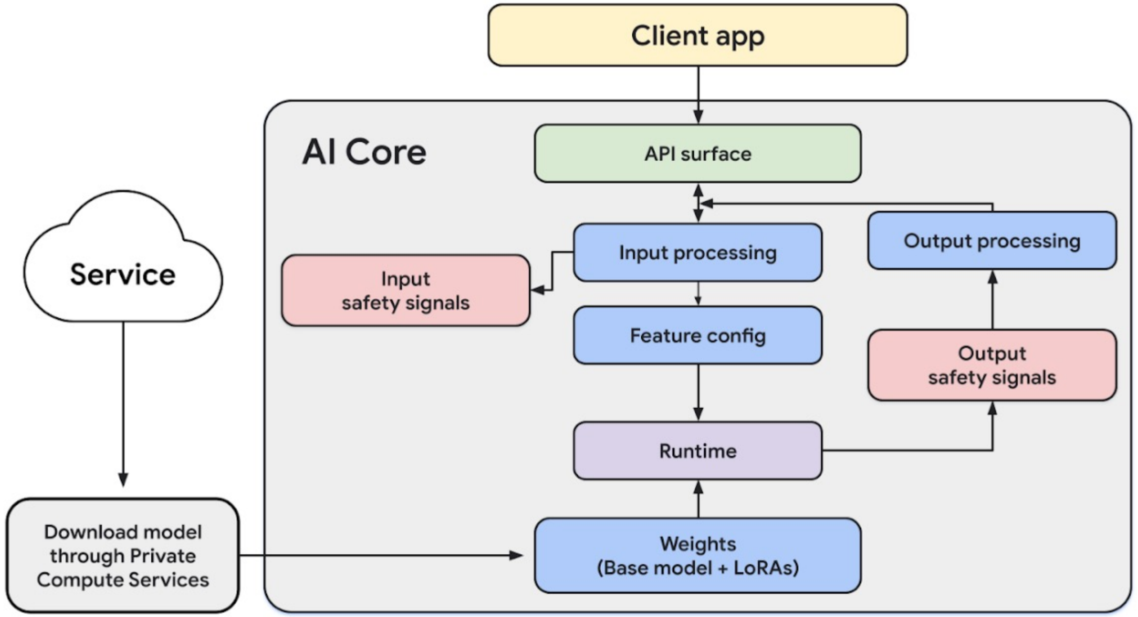
LLM as a system service

- LLM integrated into OS as a **system service** (LLMaaS)
  - Scales to infinite number of tasks
  - Hardware-design-friendly
  - OS gains full visibility into LLM requests
- Opening new research opportunities and challenges
  - *Efficiency*: how to schedule, batch, and cache-reuse system-wise LLM requests? How to manage the LLM context states across apps?
  - *Security*: how to protect app-owned LoRa? How to isolate cross-app requests?
  - *Usability*: how to upgrade LLM? How to design LLMaaS interface?
  - Etc..

A pioneering case: Android

- Android introduced an LLM system service (a.k.a. AICore)
  - Since end of 2024, but still in preview





[1] Source: https://developer.android.com/ai/gemini-nano

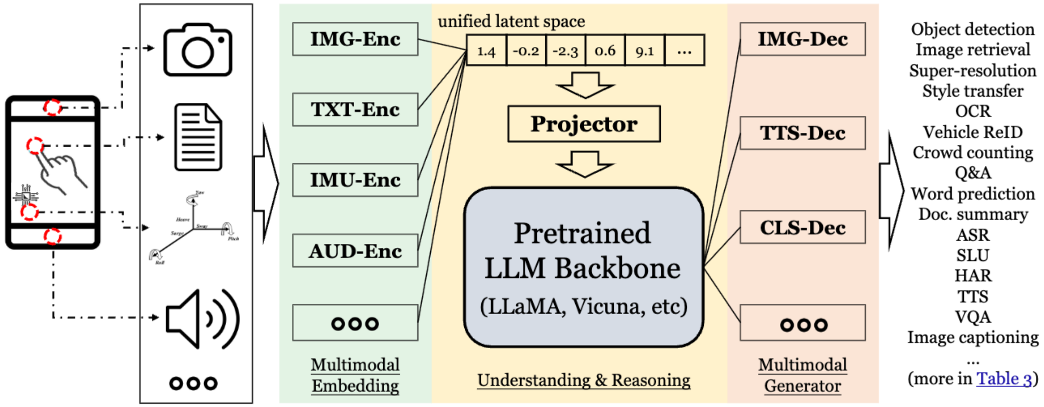
Towards a capable and generalizable LLMaaS

[MobiCom'24] Mobile Foundation Model as Firmware

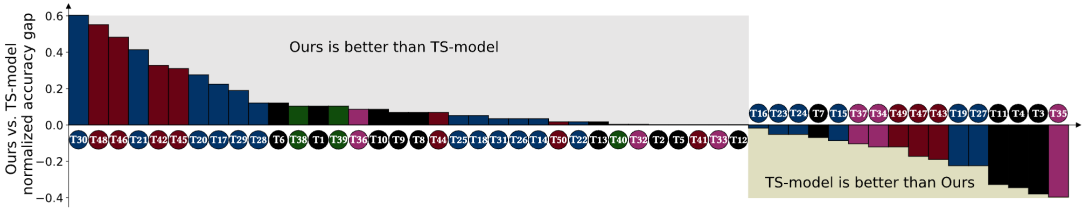
Code at https://github.com/UbiquitousLearning/MobileFM

M4: a one-size-fits-all mobile MLLM

- Can one model (as an OS service) solve all mobile AI tasks?



M4: Any-to-any modality MLLM



M4 outperforms prior arts on most tasks

Category	Tasks	Mobile Application	Dataset	Specific-Models	Results	Metrics
NLP	Input word prediction T1	Input method (GBoard)	PTB	RNN [23]	0.17*	Accuracy
	Question answering T2 T3	Private assistant (Siri)	SQuAD v2.0	RoBERTa [35]	0.79*	F1
			TyDi QA	AraELECTRA [36]	0.87	F1
	Machine translation T4	Translator (Google Translate)	wmt22 en-de	Transformer [32]	0.34*	BLEU
	Emoji prediction T5	Input method (GBoard)	tweet_eval	RoBERTa [22]	0.33*	Accuracy
	Emotion prediction T6	Conversational analytics (Clarabridge)	go_emotion	RoBERTa [29]	0.57*	Accuracy
	Sentiment analysis T7	Conversational analytics (Clarabridge)	tweet_eval	RoBERTa [27]	0.77*	Accuracy
	Text classification T8 T9	Spam SMS filtering (Truecaller)	ag_news	BERT [37]	0.93*	Accuracy
			SST2	DistilBERT [38]	0.91*	Accuracy
	Grammatical error correction T10	Writing assistant (Grammarly)	JFLEG	FLAN-t5 [30]	0.68*	BLEU
	Text summary T11	Reading assistant (ChatPDF)	CNN Daily Mail	BART [5]	0.43*	ROUGE1
	Code document generation T12	Code editor (Javadoc)	CodeSearchNet	CodeT5-base [20]	0.33*	ROUGE1
	Code generation T13	Code editor (Copilot)	Shellcode_IA32	CodeBERT [21]	0.92	BLEU
CV	Object detection T14 T15	Augmented Reality (Google Lens)	COCO	Libra-rcnn [24]	0.43*	mAP
			LVIS	X-Paste [33]	0.51	AP
	Image retrieval T16	Image searcher (Google Photos)	Clothes Retrieval	Resnet50-arcface [31]	0.90*	Recall
	Super-resolution T17	Video/Image super-resolution (VSCO)	set5	Real-ESRGAN [19]	0.82*	SSIM
	Styler transfer T18	Painting & Beatifying (Meitu)	COCO, Wikiart	StyleGAN-nada [4]	0.23	CLIP score
	Semantic segmentation T19 T20	Smart camera (Segmentix)	ADE20K-150	Deeplabv3plus [25]	0.43*	mIoU
			PASCAL VOC	Deeplabv3plus [26]	0.79*	mIoU
	Optical character recognition T21	Intelligent document automation [illegible]	Rendered SST2	CLIP [34]	0.71	Accuracy
	Im[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
		[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	Tr[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	Ve[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	Gender recognition T26	Smart camera (Face++)	Adience	MiVOLO-D1 [2]	0.96	Accuracy
	Location recognition T27	Navigation search (Google Maps)	Country211	CLIP [34]	0.46	Accuracy
	Pose estimation T28	AI fitness coach (Keep)	AP-10K	ViTPose [134]	0.69	AP
	Video classification T29	Video player (YouTube)	kinetics400	SlowFast [28]	0.79	Accuracy
	Crowd Counting T30	Smart camera (Fitness Tracking)	UCF-QNRF	CSS-CCNN [12]	437	MAE
	Image matting T31	Virtual backgrounds (Zoom)	RefMatte-RW100	MDETR [79]	0.06	MSE
Audio	Automatic speech recognition T32	Private assistant (Siri)	LibriSpeech	CTC+attention [14]	3.16%*	WER
	Spoken language understanding T33 T34	Private assistant (Siri)	FSC	Transformer [18]	0.37%	WER
			SLURP	CRDNN [3]	0.82*	Accuracy
	Emotion recognition T35	Emoji recommendation (WeChat)	IEMOCAP	ECAPA-TDNN [15]	0.64*	Accuracy
	Audio classification T36	Music discovery (Shazam)	ESC-50	ACDNet [1]	0.87	Accuracy
	Keyword spotting T37	Private assistant (Siri)	Speech command	Cnn-trad-fpool3 [17]	0.88*	Accuracy
Sensing	Human activity recognition T38 T39 T40	AI fitness coach (Keep)	Using Smartphones	TS-TCC [51]	0.90	Accuracy
			HHAR	LIMU-BERT [16]	0.84	Accuracy
			MotionSense	LIMU-BERT [16]	0.91	Accuracy
Multimodal	Text-to-speech T41	Voice broadcast (WeChat reading)	LJSpeech	Transformer [13]	3.26	MCD
	Audio captioning T42 T43	Hearing-impaired accessibility (Ava)	Clotho	Transformer [10]	0.52*	BLEU
			AudioSet	Transformer [10]	0.64	BLEU
	Image captioning T44 T45	Visual-impaired accessibility (Supersence)	MSCOCO'14	LSTM [7]	0.73*	BLEU
			Flickr8k	LSTM [110]	0.58	BLEU
	Text-to-image retrieval T46 T47	Image search (Google Photos)	Flickr8k	NAPReg [69]	0.39	Recall
			Flickr30k	CLIP [34]	0.69	Recall
	Audio/Text-to-image generation T48	Art creation (Verb Art)	VGGSound	Wav2clip [11]	99.89	FID
	Visual question answering T49 T50	Visual-impaired accessibility (Answerables)	VQA v2.0	MUTAN [41]	0.63	Accuracy
			VizWiz	MUTAN [41]	0.52	Accuracy

Tested on 50 mobile AI tasks

[1] Jinliang Yuan, et al. "Mobile Foundation Model as Firmware". In MobiCom'24.

Towards elastic LLMaaS

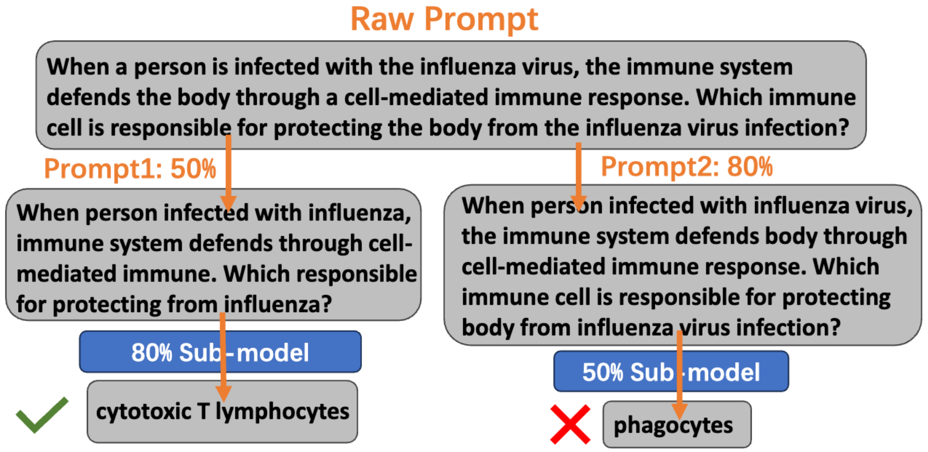
[arixv'24] ELMS: Elasticized Large Language Models On Mobile Devices

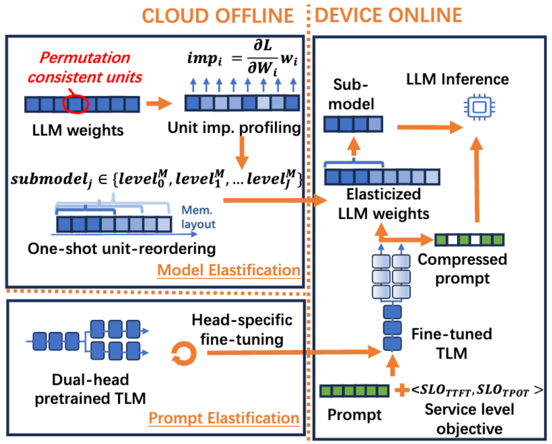
Serving LLM requests with different QoS

- Key idea: a joint planning of token/model pruning

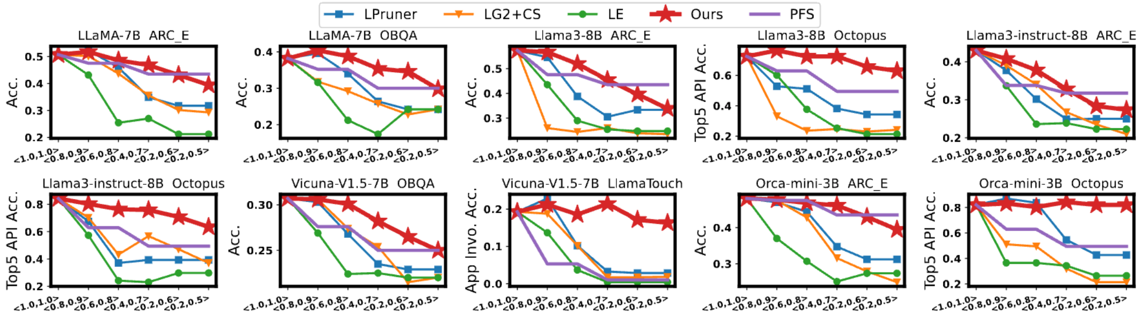
Mobile LLM App.	Service-Level Objective
Chatbot [9]	Readable TTFT/TPOT
Always-on Voice Assistant [6, 16]	Very-Low TTFT, medium TPOT
Background Screen-Event Recorder [15]	Tolerable TTFT/TPOT
Smart Message Reply [5]	Low TTFT, low TPOT
API-Calling Agent [23]	Low TTFT, acceptable TPOT
UI-Automation Agent [71, 84]	Low TTFT, acceptable TPOT

Different apps demand diversified QoS





A offline-guided, joint planning of token pruning and weights pruning



Significant improvement over static approaches

[1] Wangsong Yin, et al. "ELMS: Elasticized Large Language Models On Mobile Devices". In preprint'24.

So, what's unique to mobile LLM?

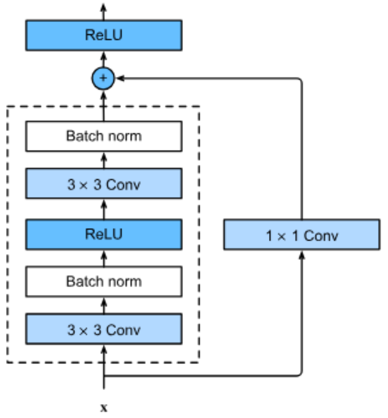
(compared to traditional DNN-powered apps)

Workload:	fragmented tasks	→	a unified agent
OS:	model-agnostic	→	LLM-native
Hardware:	heterogeneous H/W	→	DSA-dominated

How to serve LLM requests with low latency and energy efficiency?

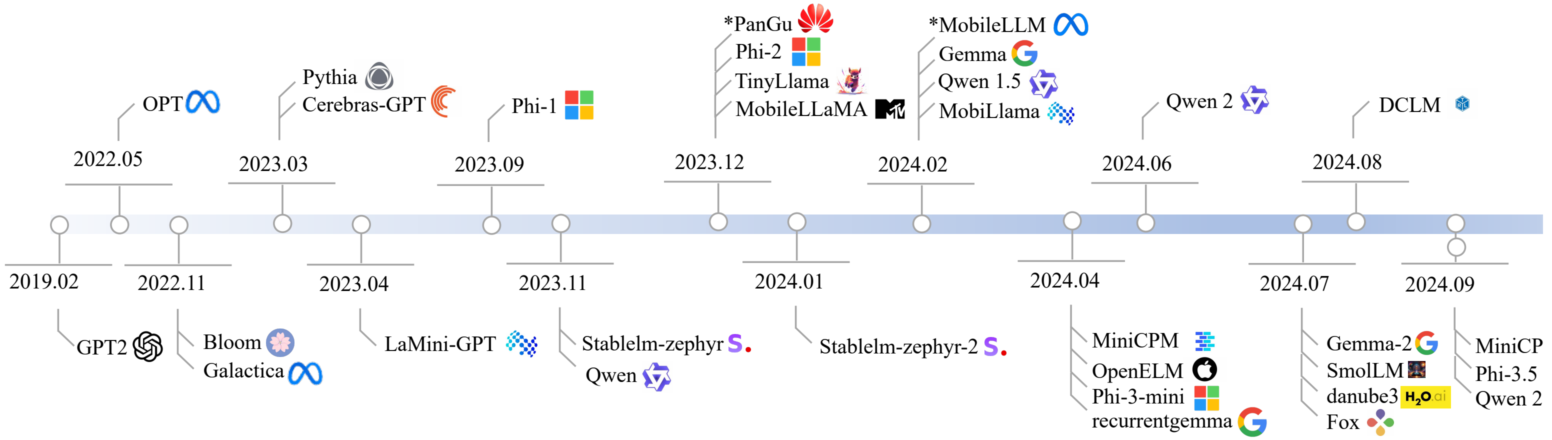
On-device LLM needs LLM-processor

- On-device **resource scarcity** further exacerbated.



ResNet, YoLo, LSTM, etc
(<200M)
<100ms to process one image
<100MB memory footprint
Easy to quantize (integer-only)
Static shape and cost

vs.



Small Language Models (1B~5B)

>10sec to process one prompt on CPU
>1GB memory footprint
Difficult to quantize (FP required)
Dynamic shape and increased cost with longer prompt

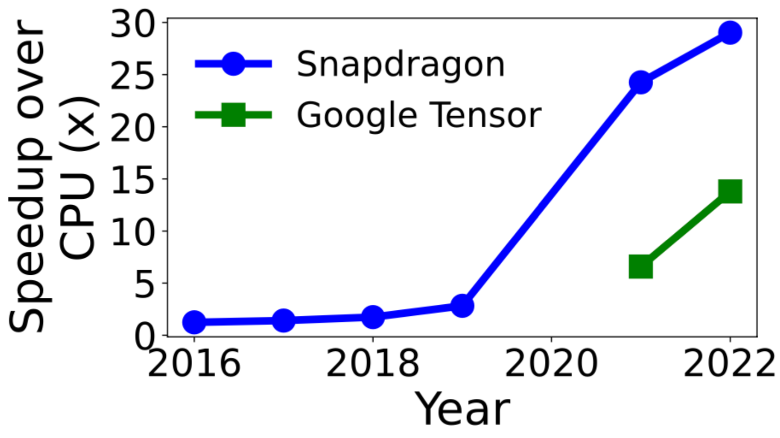
[1] Zhenyan Lu, et al. "Small Language Models: Survey, Measurements, and Insights". In preprint'24.

On-device LLM needs NPU

- DSA (LLM-processor) is the answer to on-device LLM.

Vendor	Latest NPU	SDK	Open	Group	INT8 Perf.
Qualcomm	Hexagon NPU [15]	QNN [23]	×	×	73 TOPS
Google	Edge TPU [17]	Edge TPU API [7]	×	×	4 TOPS
MediaTek	MediaTek APU 790 [11]	NeuroPilot [13]	×	N/A	60 TOPS
Huawei	Ascend NPU [6]	HiAI [9]	×	×	16 TOPS

"Open": Open-source?; "Group": Support per-group quantization MatMul? "N/A": No available documents for public; "INT8 Perf.": Int8 performance.



- *The gap between CPU/GPU and NPU increases over time*
  - *Moore's law still stands for NPU*
- *The gap of energy efficiency is even larger*

[1] Jinliang Yuan. "Mobile Foundation Model as Firmware". In MobiCom'24.

Filling the design gap between legacy NPUs and modern LLM inference

[ASPLOS'25] Fast On-device LLM Inference with NPUs

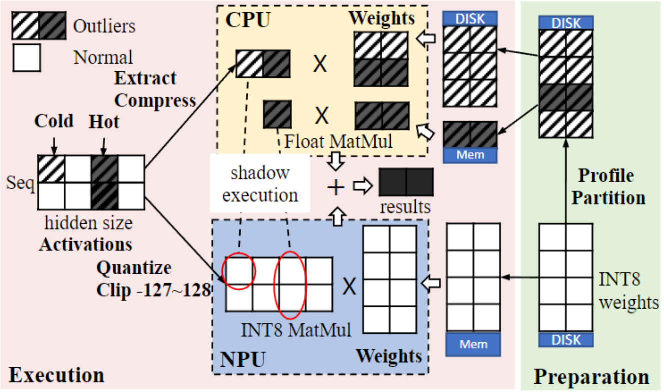
Code at https://github.com/UbiquitousLearning/mllm

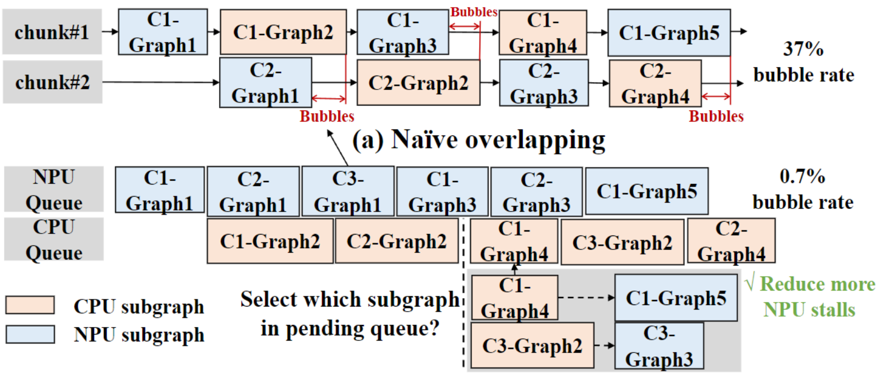
llm.npu: accelerating LLM prefilling with NPU

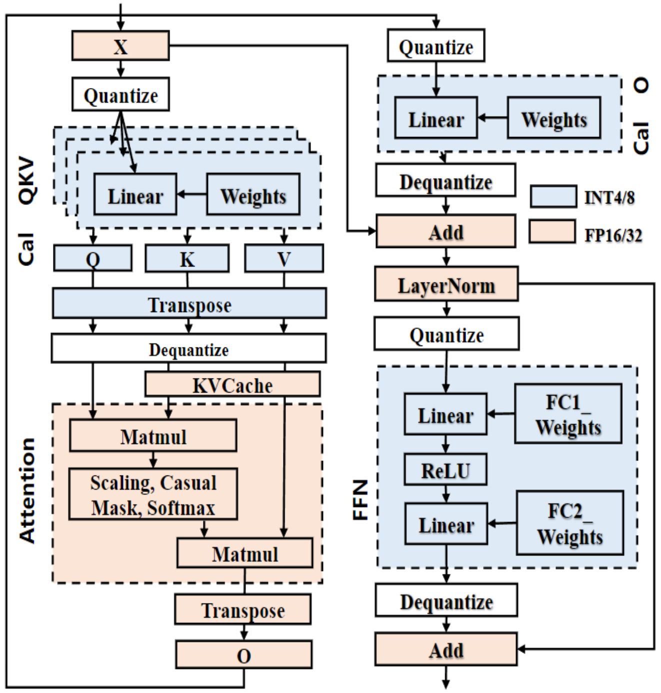
- Legacy mobile NPU has poor support for

  (1) Dynamic shape; (2) FP operations; (3) group-level quantization

- *llm.npu* proposes
  - Chunked prefill with partial sharing
  - Shadow outlier execution across CPU/NPU
  - Our-of-order scheduling among CPU/NPU



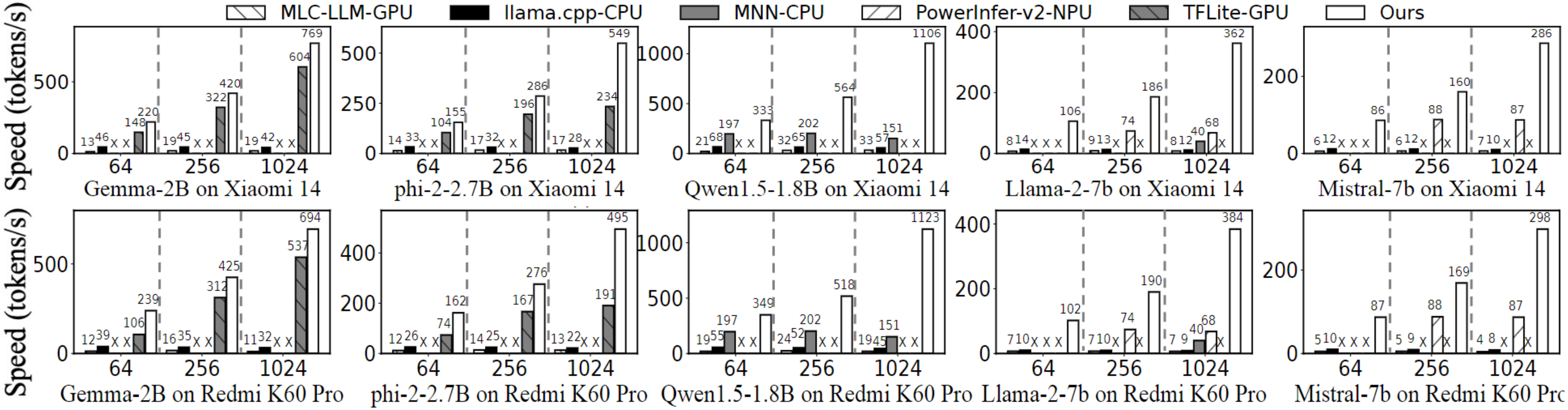




[1] Daliang Xu, et al. "Fast On-device LLM Inference with NPUs". In ASPLOS'25.

Highlighted results

Prefill speed under different prompt lengths on different devices (datasets: Longbench-2wiki-Multi-doc QA)
Baselines: MLC-LLM (GPU), llama.cpp (CPU), MNN (CPU), PowerInfer-v2 (NPU), TFLite (GPU)



7.3×–18.4×faster than baselines on CPU, and 1.3×–43.6× on GPU with prompt length of 1024
Achieves >1000 tokens/second on Qwen1.5-1.8B (for the first time)

[1] Daliang Xu, et al. "Fast On-device LLM Inference with NPUs". In ASPLOS'25.

Filling the design gap between legacy NPUs and modern LLM training

[USENIX ATC'24] FwdLLM: Efficient Federated Finetuning of Large Language Models with Perturbed Inferences

Code at https://github.com/UbiquitousLearning/FwdLLM

FwdLLM: BP-free LLM finetuning

- Key idea: leveraging forward gradient for LLM finetuning

$$\boldsymbol{g}(\boldsymbol{\theta}) = (\nabla f(\boldsymbol{\theta}) \cdot \boldsymbol{v})\,\boldsymbol{v}$$

A random, independent perturbation with same size as trainable weights θ

Forward gradient, an <u>unbiased estimator</u> of $f(\theta)'s$ gradient

The directional derivative of f at point θ in direction v. Computing it takes only forward, no need for backpropagation

Compared to BP approach:

- Legacy NPU-compatible
- More memory efficient

- Further optimizations:
  - var.-controlled perturbation pacing
  - discriminative perturbation sampling

Methods	Mem. (GB)	Centralized Training (A100) Acc.	Round	Time	Federated Learning Acc.	Round	Time
BP, FP16	39.2	89.7	500	0.1 hrs	N/A due to memory inefficiency on Pixel 7 Pro (8GB)		
BP, INT8	32.4	88.6	500	0.06 hrs			
BP, INT4	28.5	87.8	500	0.04 hrs			
Ours, FP16	15.6	87.0	240	1.5 hrs			
Ours, INT8	7.9	86.9	260	0.8 hrs			
Ours (CPU), INT4	4.0	85.8	130	0.25 hrs	85.8	130	0.19 hrs
Ours (NPU*), INT4							0.07 hrs

- Highlighted results: federated Llama-7B finetuning on devices, with significant speedup and memory saving

[1] Mengwei Xu, et al. "FwdLLM: Efficient Federated Finetuning of Large Language Models with Perturbed Inferences". In ATC'24.

Filling the design gap between legacy NPUs and modern LLM design

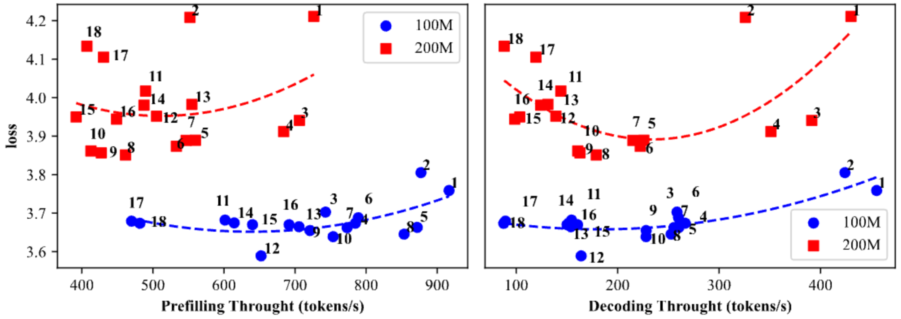
[arxiv'24] PhoneLM: an Efficient and Capable Small Language Model Family through Principled Pre-training

Code at https://github.com/UbiquitousLearning/PhoneLM

Models at https://huggingface.co/mllmTeam/PhoneLM-1.5B

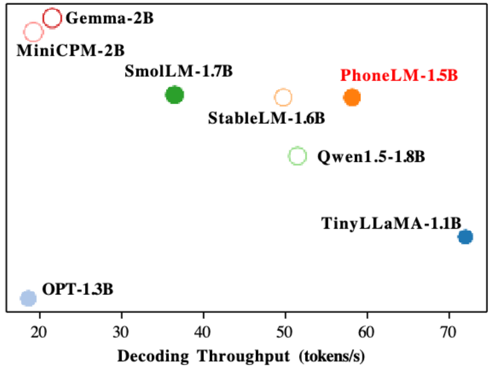
PhoneLM: efficient SLMs for devices

- An argument: SLM shall adapt to the target device hardware
  - Hardware-specific, ahead-of-pretraining hyperparameter search for runtime resource efficiency



The SLM design (hyper-parameters) has more impacts on the runtime performance than the capability

Name	Size	Date	Training tokens	HellaSwag	WinoGrande	PIQA	SciQ	BoolQ	ARC Easy	ARC Challenge	Average
Pythia (EleutherAI, 2023.03a)	1.4B	23.03	207B	52.0	57.2	71.1	79.2	63.2	53.9	28.3	57.84
OPT (Facebook, 2022.05a)	1.3B	22.05	180B	53.7	59.0	71.0	78.1	57.2	51.3	28.0	56.90
BLOOM (BigScience, 2022.11b)	1.1B	22.11	350B	43.0	54.9	67.2	74.6	59.1	45.4	25.6	52.83
TinyLlama (Unknown, 2023.12)	1.1B	23.12	3B	59.1	58.9	73.0	82.3	58.6	55.7	31.0	59.80
MobileLLaMA (Meituan, 2023.12)	1.4B	23.12	1.3T	56.1	59.4	73.0	81.9	56.7	55.8	30.3	59.03
MobiLlama (MBZUAI, 2024.02)	1B	24.02	1.25T	62.2	59.3	74.8	82.8	60.3	56.4	31.7	61.07
OpenELM (Apple, 2024.04)	1.1B	24.04	1.5T	64.8	61.7	75.6	83.6	63.6	55.4	32.3	62.43
DCLM (Toyota, 2024.08)	1.4B	24.08	4.3T	53.6	66.3	77.0	94.0	71.4	74.8	41.2	68.33
SmolLM (HuggingFace, 2024.07)	1.7B	24.07	1T	49.6	60.9	75.8	93.2	66.0	76.4	43.5	66.49
Qwen 1.5 (Alibaba, 2024.02)	1.8B	24.02	2.4T	60.9	60.5	74.2	89.4	66.5	59.1	34.7	63.61
Galactica (Facebook, 2022.11)	1.3B	22.11	106B	41.0	54.4	63.8	87.7	62.0	58.6	30.5	56.86
StableLM 2 (StabilityAI, 2024.01)	1.6B	24.01	2T	68.8	64.1	75.1	76.9	80.0	60.3	39.2	66.34
Cerebras-GPT (Cerebras, 2023.03a)	1.3B	23.03	371B	38.4	51.9	66.8	73.0	59.3	45.8	25.3	51.50
MiniCPM (OpenBMB, 2024.04)	1B	24.04	1.2T	67.5	63.7	75.1	91.0	70.5	62.9	38.1	66.97
MiniCPM (OpenBMB, 2024.04)	2B	24.04	1.2T	67.2	63.9	76.1	92.5	74.6	69.0	42.7	69.43
Gemma (Google, 2024.02)	2B	24.02	3T	71.4	65.2	78.4	91.4	69.9	72.3	42.0	70.09
Gemma 2 (Google, 2024.07)	2B	24.07	2T	55.0	68.7	78.7	96.0	73.6	80.3	46.9	71.31
PhoneLM	**1.5B**	24.11	1.5T	**66.9**	**63.0**	**77.3**	**88.8**	**65.5**	**69.7**	**39.9**	**67.31**



SOTA tradeoff between capability and efficiency

[1] Rongjie Yi, et al. "PhoneLM: an Efficient and Capable Small Language Model Family through Principled Pre-training". In preprint'24.

Looking into the future..

Mortal Computation

- If we abandon immortality and accept that the knowledge is inextricable from the precise physical details of a specific piece of hardware, we get two big benefits:

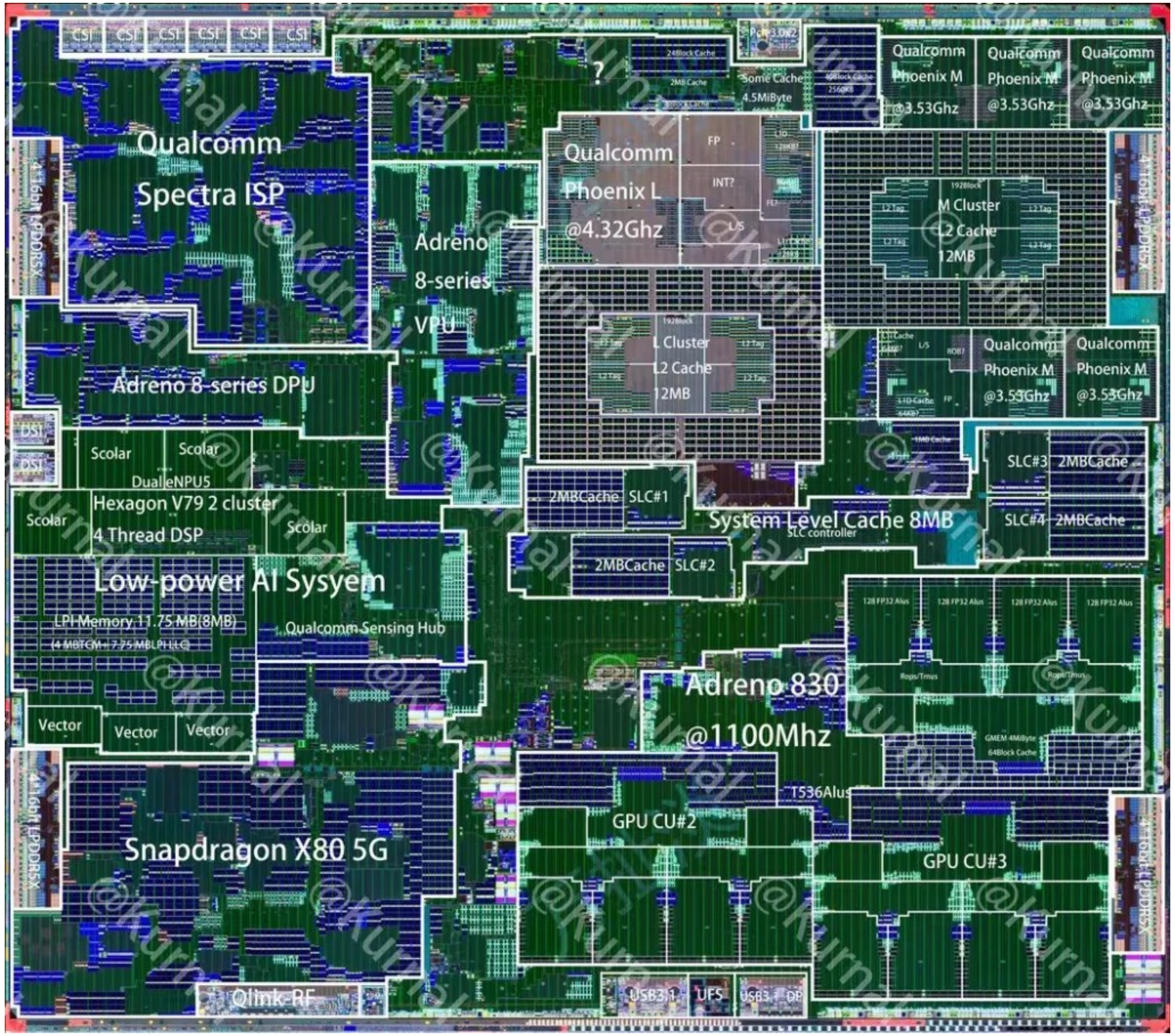
- Huge energy savings
  - We can use very low power analog computation.

- Much cheaper hardware
  - The hardware could be grown cheaply in 3-D instead of being manufactured very precisely in 2-D.
  - This would require lots of new nano-technology or perhaps genetic re-engineering of biological neurons.

We shall probably look for hardware-software co-evolution, e.g., mortal computation by Geoffrey Hinton

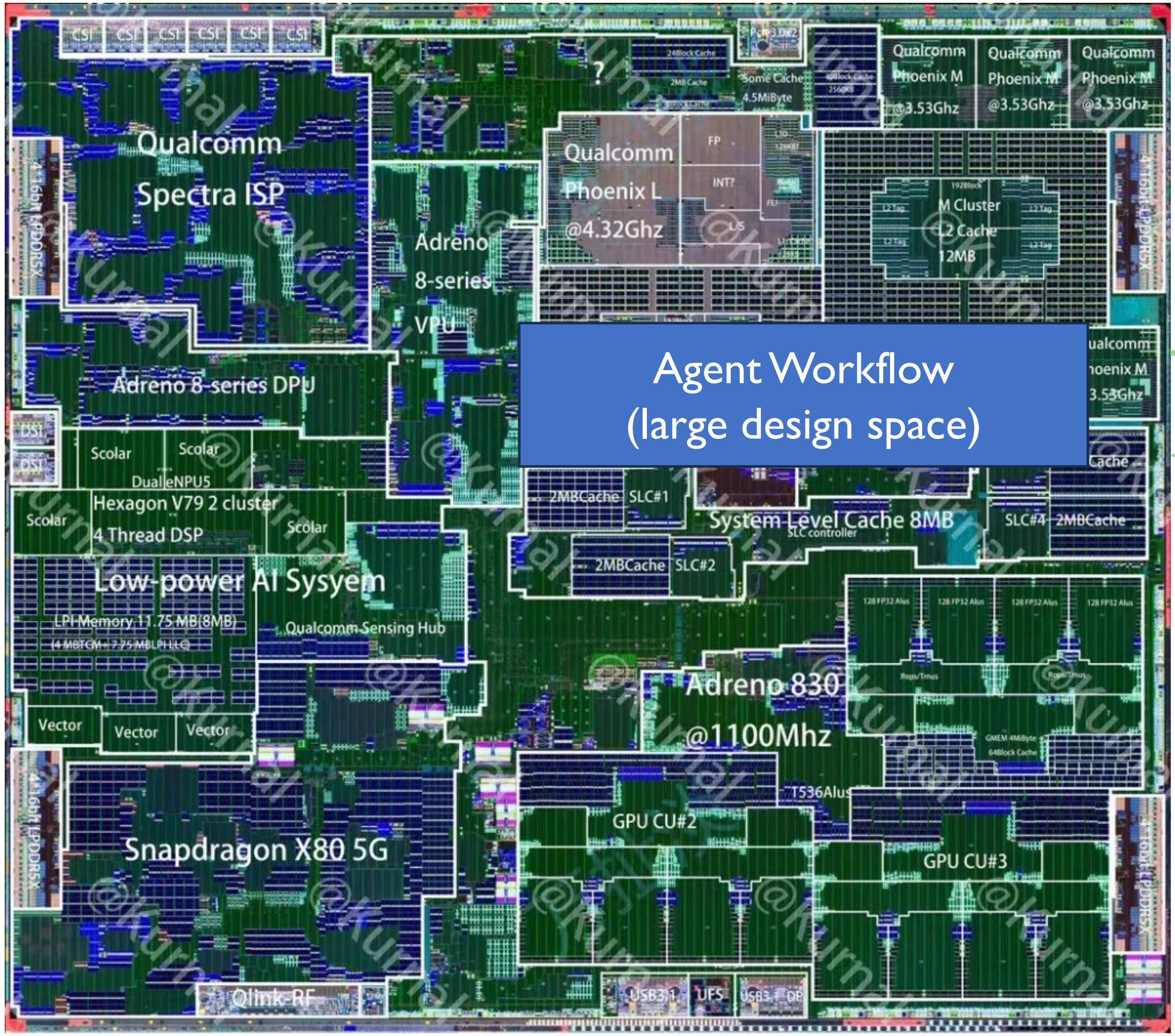
"Two paths to Intelligence" by Geoffrey Hinton

The Future: full-stack design!



https://innogyan.in/2024/10/28/die-shot-of-snapdragon-8-elite-reveals-component-space-allocation/

The Future: full-stack design!



Prompt/LoRa

Response

A One-Size-Fits-All LLM

A Tiny Kernel

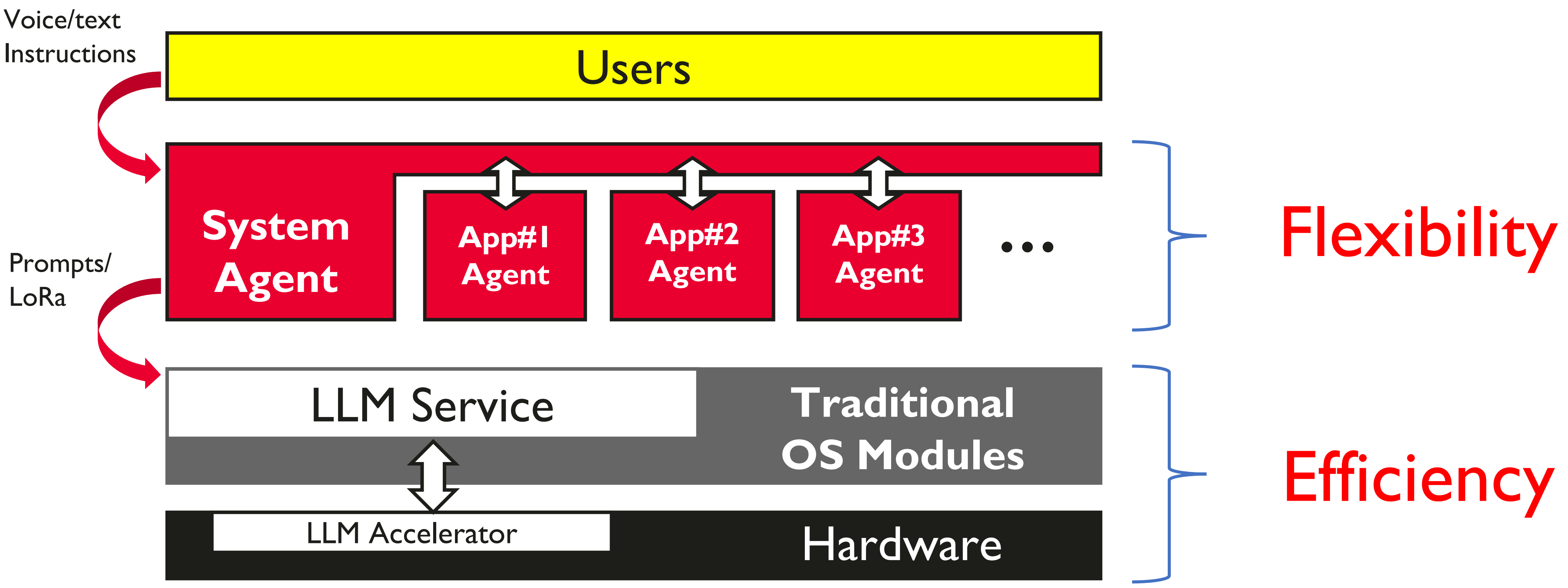
LLM Processor | A Dedicated HBM Unit

Time to sacrifice flexibility for efficiency!
(we still have flexibility at agent workflow)

https://innogyan.in/2024/10/28/die-shot-of-snapdragon-8-elite-reveals-component-space-allocation/

The Future: full-stack design!

- **One LLM, Many Agents**



Takeaways

- On-device LLM is reinventing the mobile devices
  - A total paradigm shift of mobile AI ecosystem
- It calls for full-stack LLM research
  - OS, runtime, model, and application (agent)