PennState

A Comprehensive Survey of Small Language Models in the Era of Large Language Models

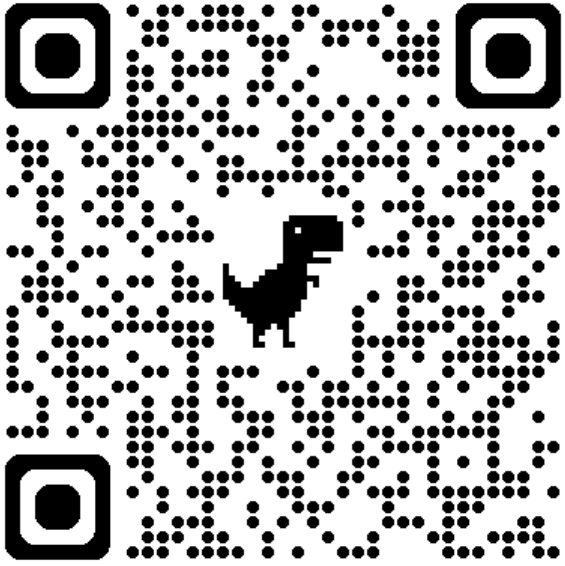
Techniques, Enhancements, Applications, Collaboration with LLMs, and Trustworthiness

Fali Wang (fairyfali.github.io)

January 16, 2025 *(Work in submission)*

Related Materials

- Paper: arXiv
- Github: Github
- English Blog: in Linkin
- Chinese Blog: in Wechat



GitHub Repo

Outline[1]

[1]Here * means the section is key

Why SLMs?

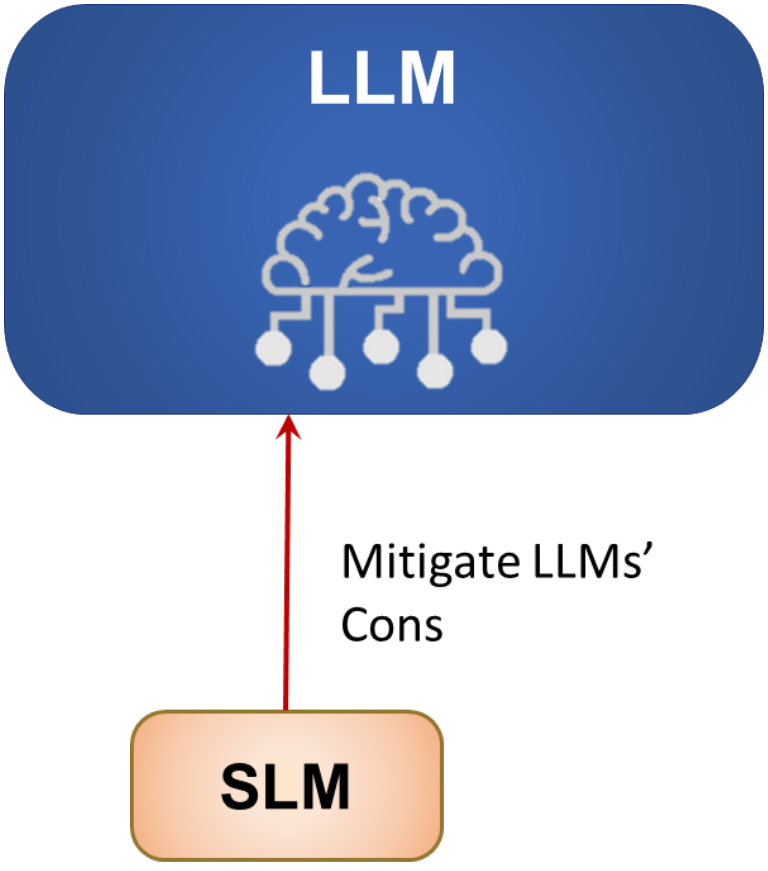
Pros:

- Emergent ability
- Generalizability

Cons:

- Privacy leakage
- On-device deployment
- Inference latency
- Expensive fine-tuning
- Inferior to specialized models

Why SLMs?



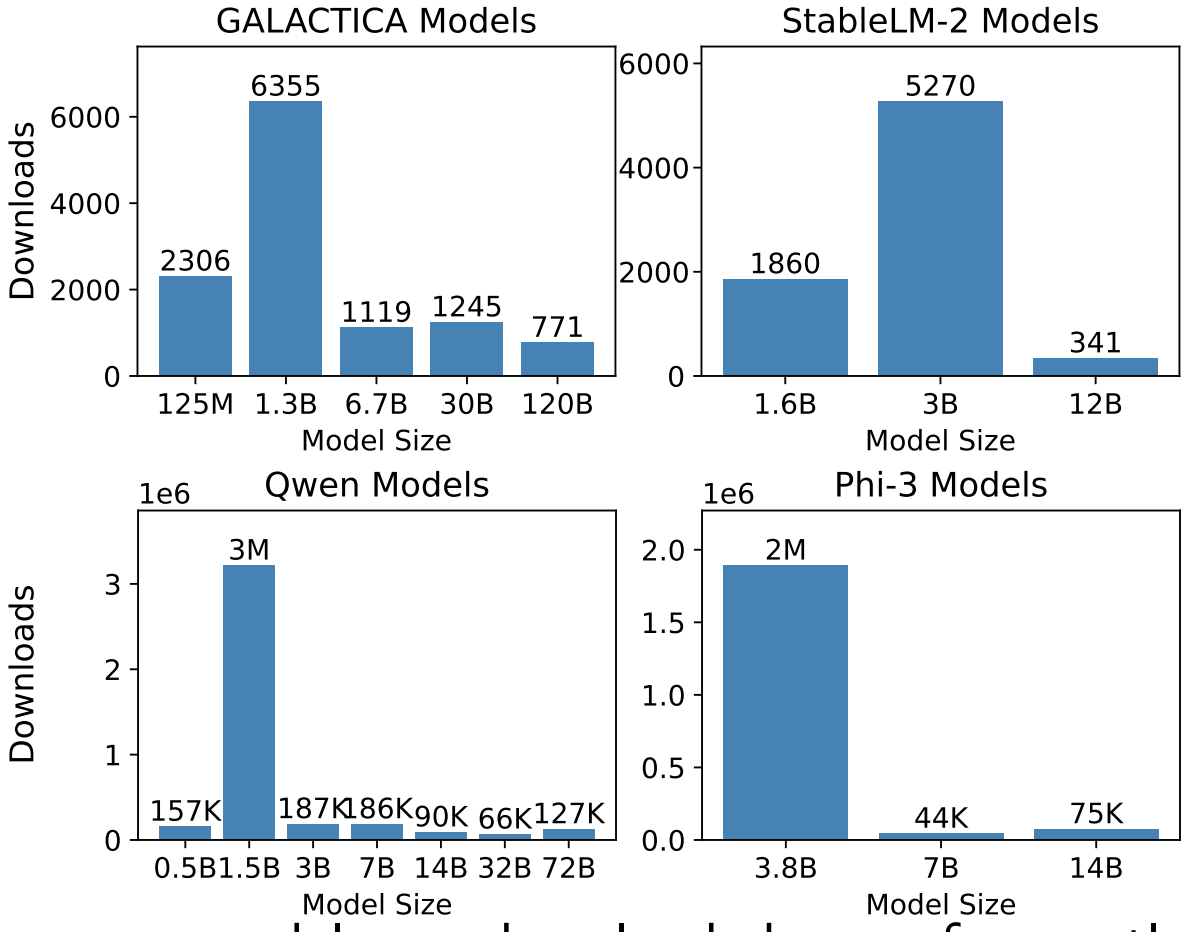
Pros:

- Emergent ability
- Generalizability

Cons:

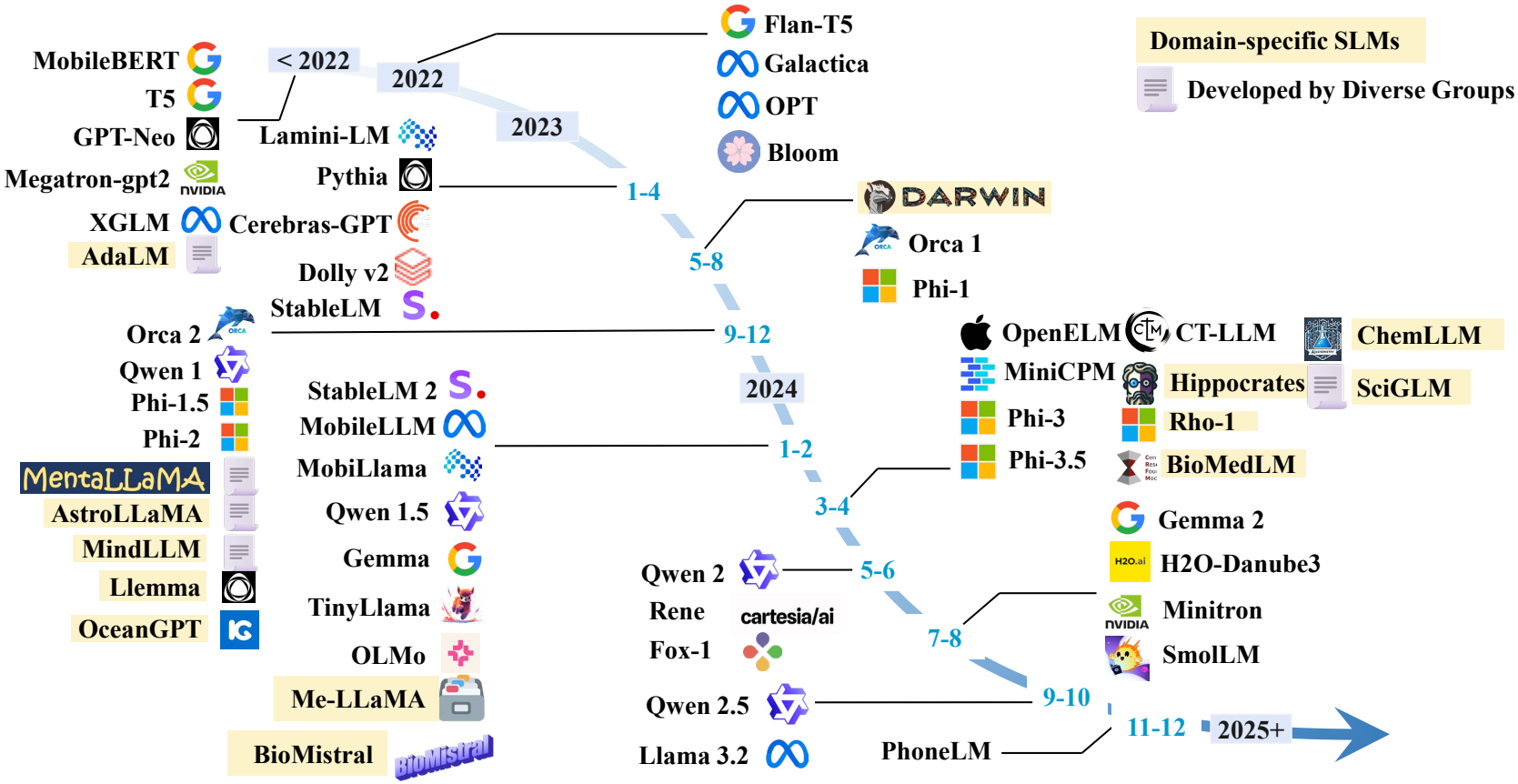
- Privacy leakage
- On-device deployment
- Inference latency
- Expensive fine-tuning
- Inferior to specialized models

Smaller Language Models are Popular



Smaller language models are downloaded more frequently than larger models in the Hugging Face community.

Timeline of Existing SLMs



Issues of Existing SLM Definition

- **Relative Definition**: Lu et al. [2024], Van Nguyen et al. [2024], Chen and Varoquaux [2024] view "small" as relative to "large."
- **Perspective of Mobile Devices**: MobileLLM Liu et al. [2024] categorizes SLMs as models with fewer than one billion parameters, suitable for mobile devices with up to 6GB memory.
- **Perspective of Emergent Ability**: SLMs typically range from a few million to a few billion (under 7B or 10B) [2], often lacking emergent abilities Fu et al. [2023].
- However, they lack consensus and no clear boundaries between SLMs and LLMs. 7B LMs belong to an LLM or SLM?

[2] The Rise of Small Language Models: Efficiency and Customization for AI

Our SLM Definition

- Considering both capability and resource constraints, our definition is:

Def 1: Our SLM Definition

Given specific tasks and resource constraints, we define SLMs as falling within a range where the lower bound is the minimum size at which the model exhibits emergent abilities for a specialized task, and the upper bound is the largest size manageable within limited resource conditions.

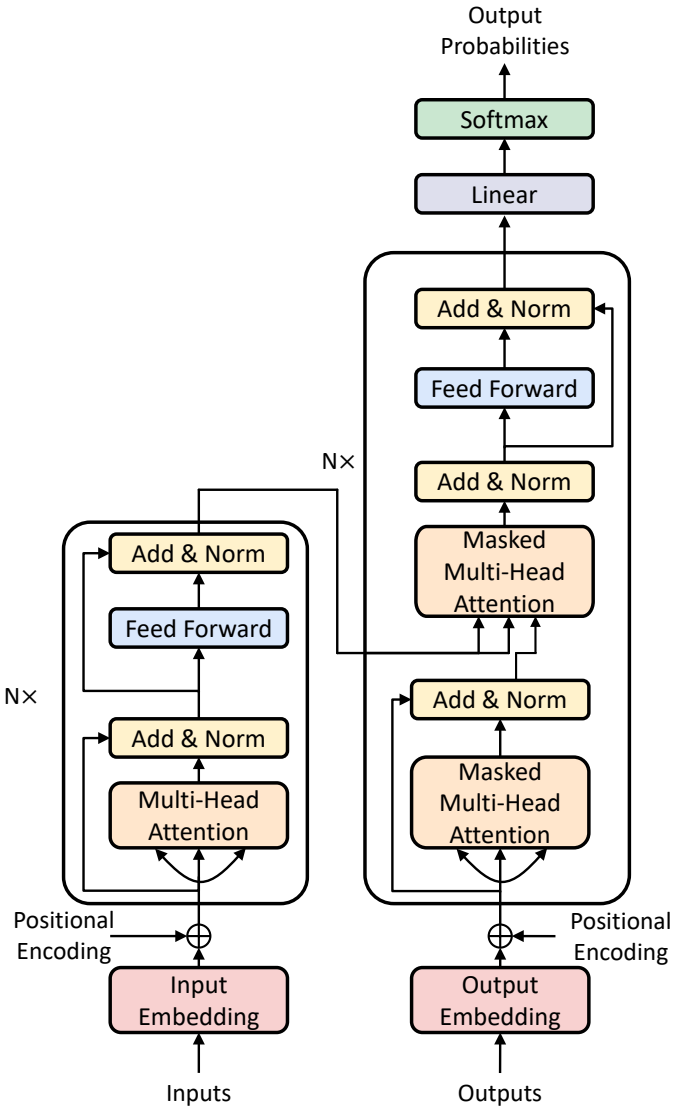
Outline

Architecture

Transformer with Self-attention mechanism

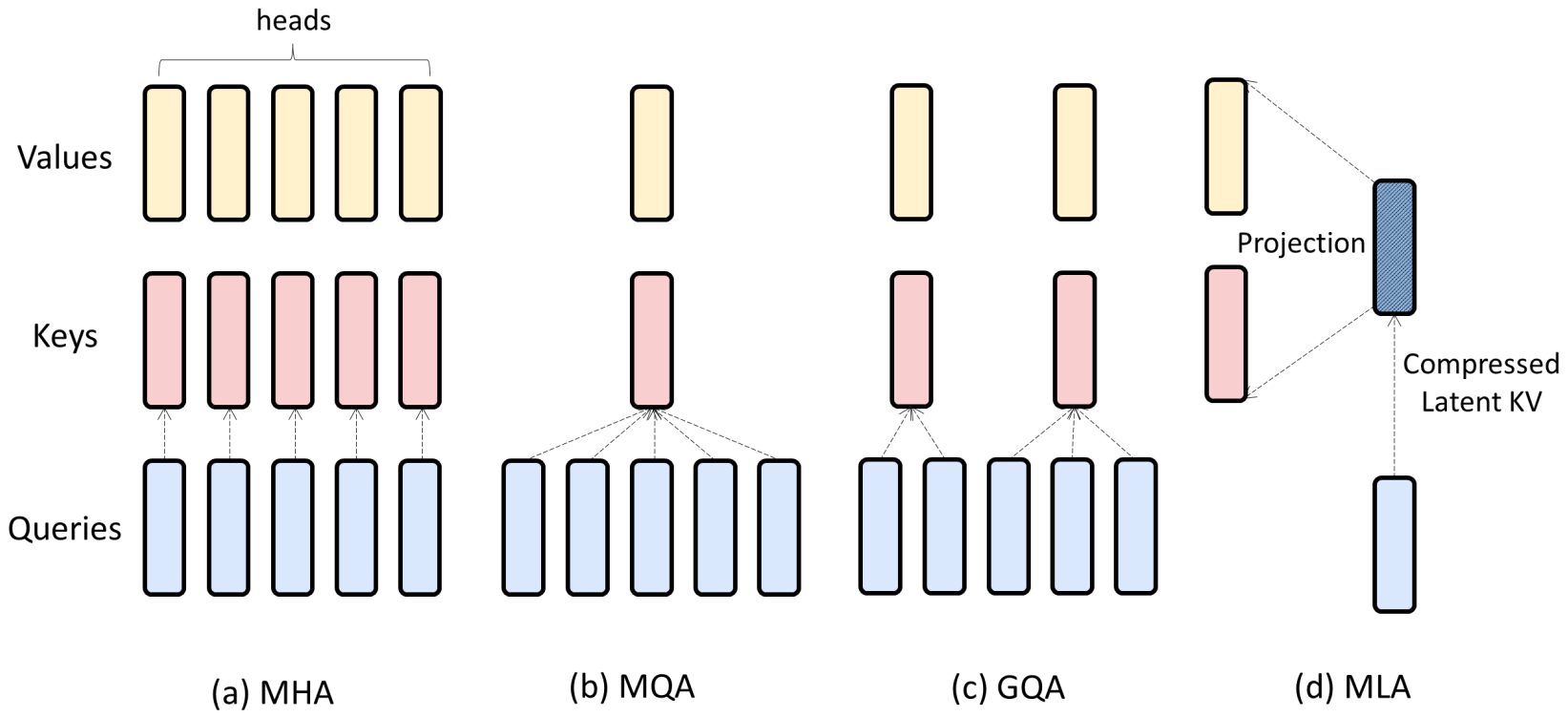
State Space Models with Recurrent states

Transformer - Overview



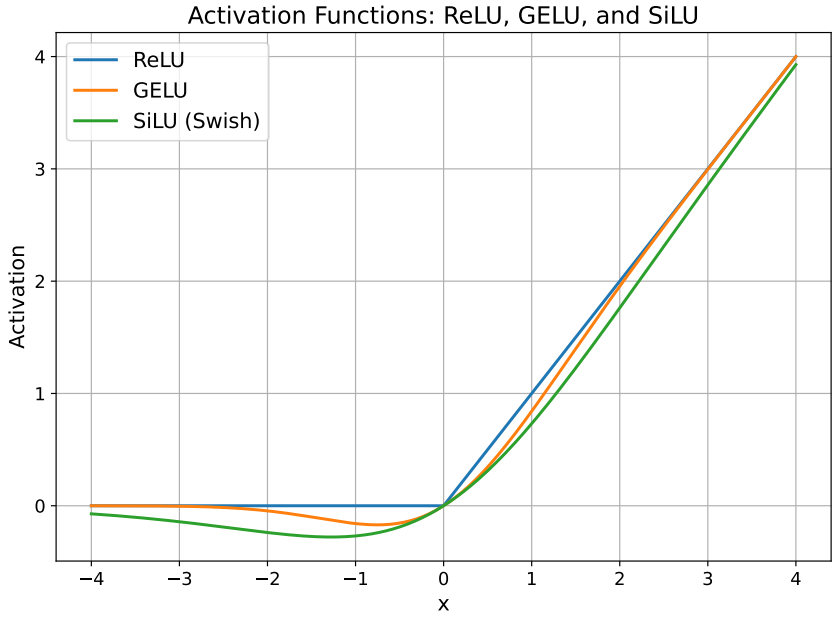
- Positional Embedding
  - Sinusoidal Positional Embedding
  - Rotary Positional Embedding
- Self-attention mechanism
  - Multi-Head Attention
  - Multi-Query Attention
  - Grouped Query Attention
  - Multi-Head Latent Attention
- Feedforward Network, with activation func:
  - ReLU, GELU, SiLU, SwiGLU
- Layer Normalization
  - Layer Norm
  - RMS Norm

Transformer - Attention Types



SLMs favor GQA as it could balance functionality with cache space (less cache also contributes to computing efficiency and speed).

Transformer - activation function in FFNs



- ReLU: $\max(0, x)$
- GELU: $x \cdot \frac{1}{2}\left[1 + \text{erf}\left(\frac{x}{\sqrt{2}}\right)\right]$
- SiLU (i.e. Swish): $x \cdot \frac{1}{1+e^{-x}}$
- SwiGLU: $\text{Swish}(x \cdot W + b) \odot (x \cdot V + c)$

ReLU promotes greater sparsity, enabling faster computations. SwiGLU is a parametric function that adapts to various tasks, enhancing capabilities. SLMs prefer SiLU for its balance of efficiency and capability.

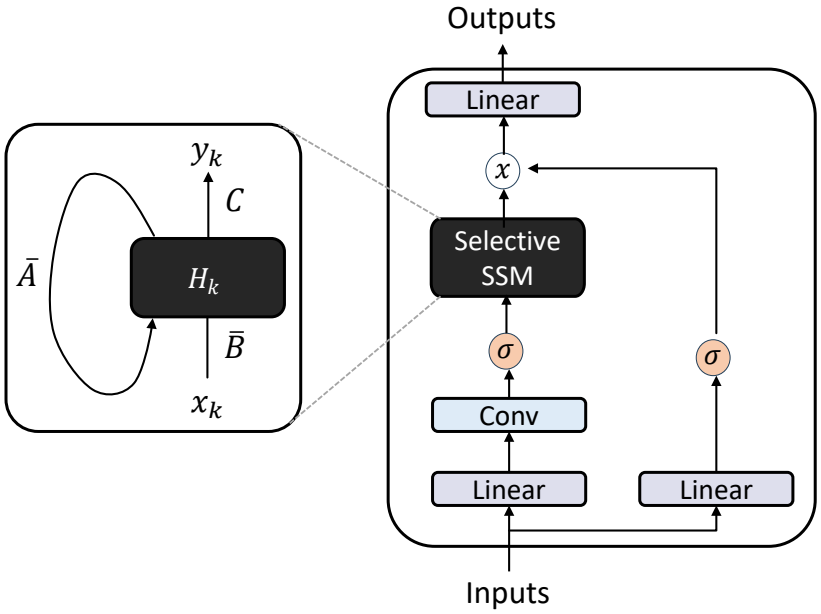
Transformer - Layer Normalization

- **Non-Parametric Layer Norm**: $\text{LN}(x) = \frac{x-\mu}{\sigma}$
- **Parametric Layer Norm**: $\text{PLN}(x) = \gamma\left(\frac{x-\mu}{\sigma}\right) + \beta$
- **RMS Norm**: $\text{RMSNorm}(x) = \gamma\frac{x}{\sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2 + \epsilon}} + \beta$

  where N is the number of inputs, x_i is the i-th input, γ and β are learnable parameters for adaptive scaling and bias, and ϵ is a small constant to prevent division by zero.

RMS Norm is preferred over Layer Norm due to its computational demands and expressiveness balance.

SSMs - Mamba



- Transformer has fast training but slow inference.
- Mamba, based on SSMs (similar to RNNs), focuses on the immediate previous hidden state and offers fast inference by
  - Dynamic selection mechanism.
  - Hardware-aware Algorithm.

Mamba achieves *higher parameter utilization* and *faster inference* than Transformer, making it more suitable for SLMs.

Outline

Acquisition methods

- Pre-train on large corpus then fine-tune
  - Collecting data $\rightarrow$ pre-processing $\rightarrow$ tokenization for vocab $\rightarrow$ training
  - Loss function: MLM, NTP
  - Optimizer: Adam
  - Parameter-efficient fine-tuning: LoRA, Prefix-tuning, prompt tuning, and adapters.
- Acquisition from LLMs via Compression
  - Pruning
  - Knowledge distillation

Outline

- Introduction
- Architecture
- Pre-trained and compressed SLMs
- *Enhancement Strategy for SLMs
- *Existing SLMs
- *On-device Applications
- *Deployment of SLMs
- *SLMs for LLMs
- *Synergy between SLMs and LLMs
- Trustworthiness
- Future Directions

Enhancement Strategy

Pre-training SLMs from scratch

Supervised Fine-tuning

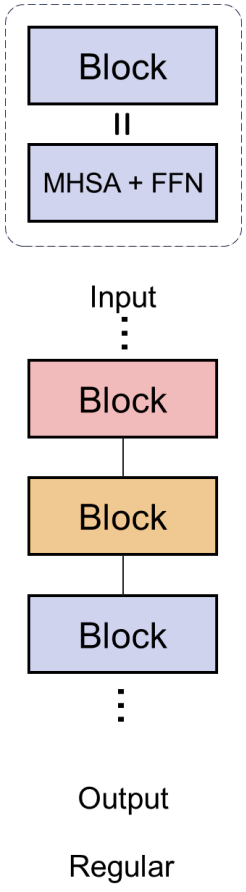
Knowledge Distillation-Data Quality

Knowledge Distillation-Distribution Mismatch

Enhancement Strategy

- **Pre-training SLMs from scratch**: Architecture choice; Parameter Sharing; Data Filtering; Multiple-round training.
- **Supervised Fine-tuning**: Pretrain-then-finetune; Instruction tuning; Preference optimization.
- **Knowledge Distillation**: Data quality; Distribution mismatch; Domain gap.

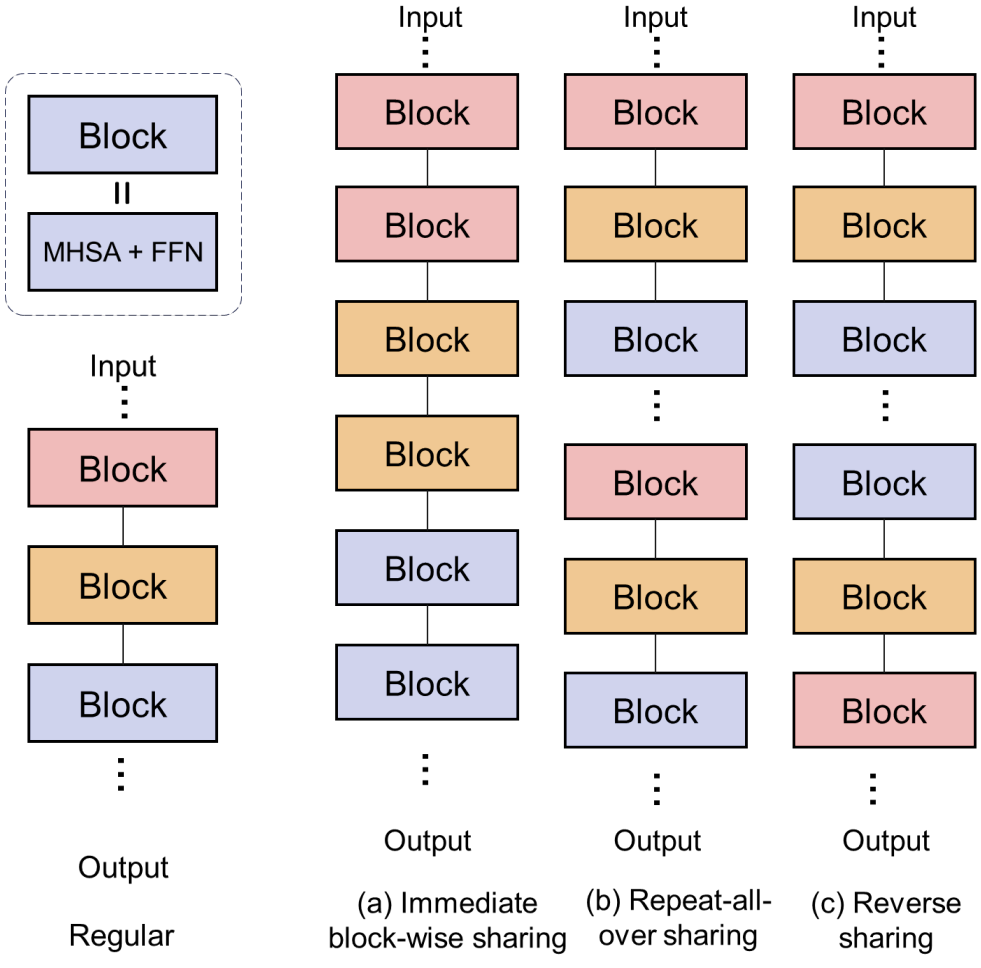
Pre-training from scratch-Parameter Sharing [3] [4]



[3] Omkar Thawakar, et al. Mobillama: Towards accurate and lightweight fully transparent GPT

[4] Zechun Liu et al., MobileLLM: Optimizing Sub-billion Parameter Language Models for On-Device Use Cases

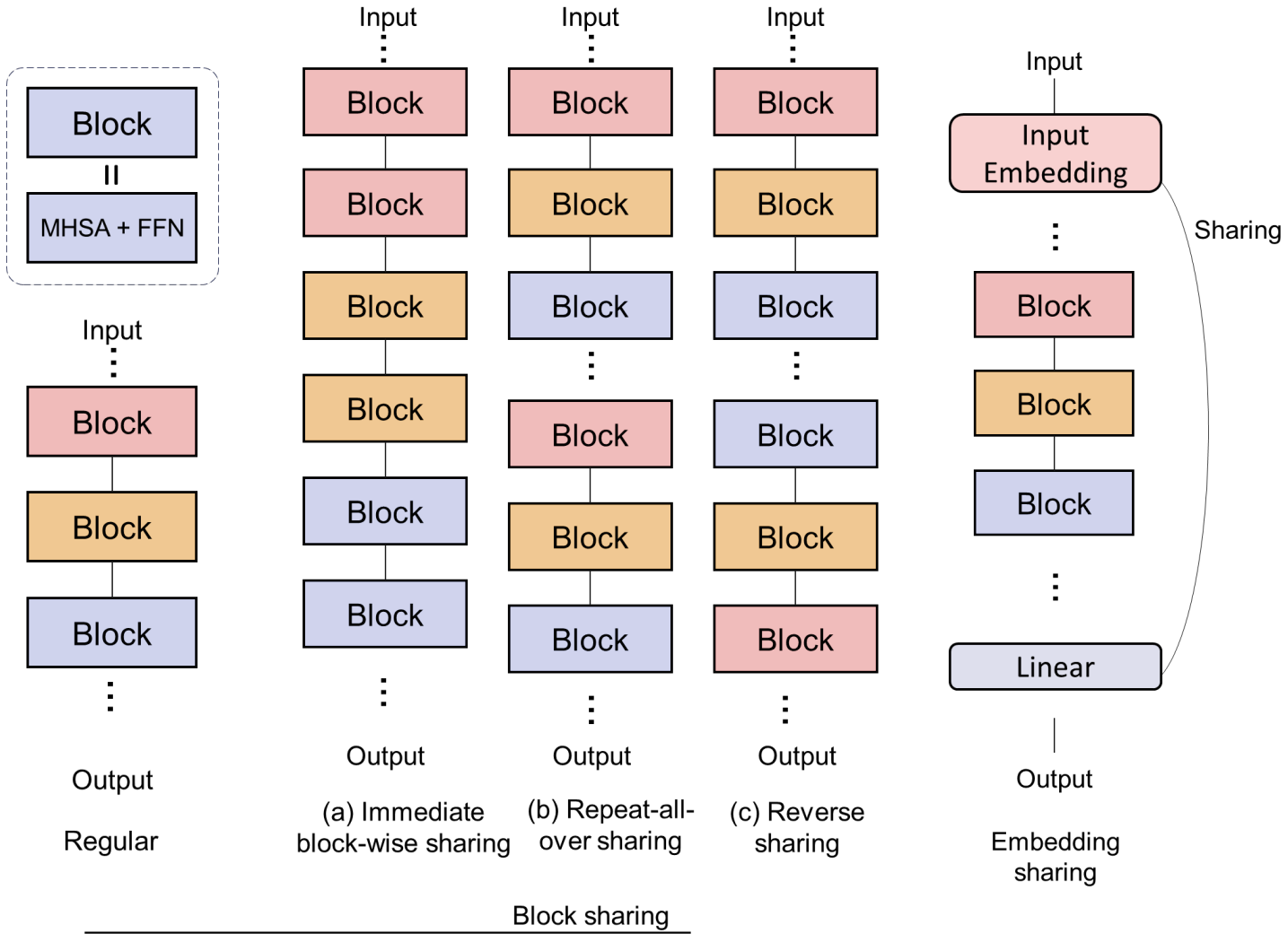
Pre-training from scratch-Parameter Sharing [3] [4]



[3] Omkar Thawakar, et al. Mobillama: Towards accurate and lightweight fully transparent GPT

[4] Zechun Liu et al., MobileLLM: Optimizing Sub-billion Parameter Language Models for On-Device Use Cases

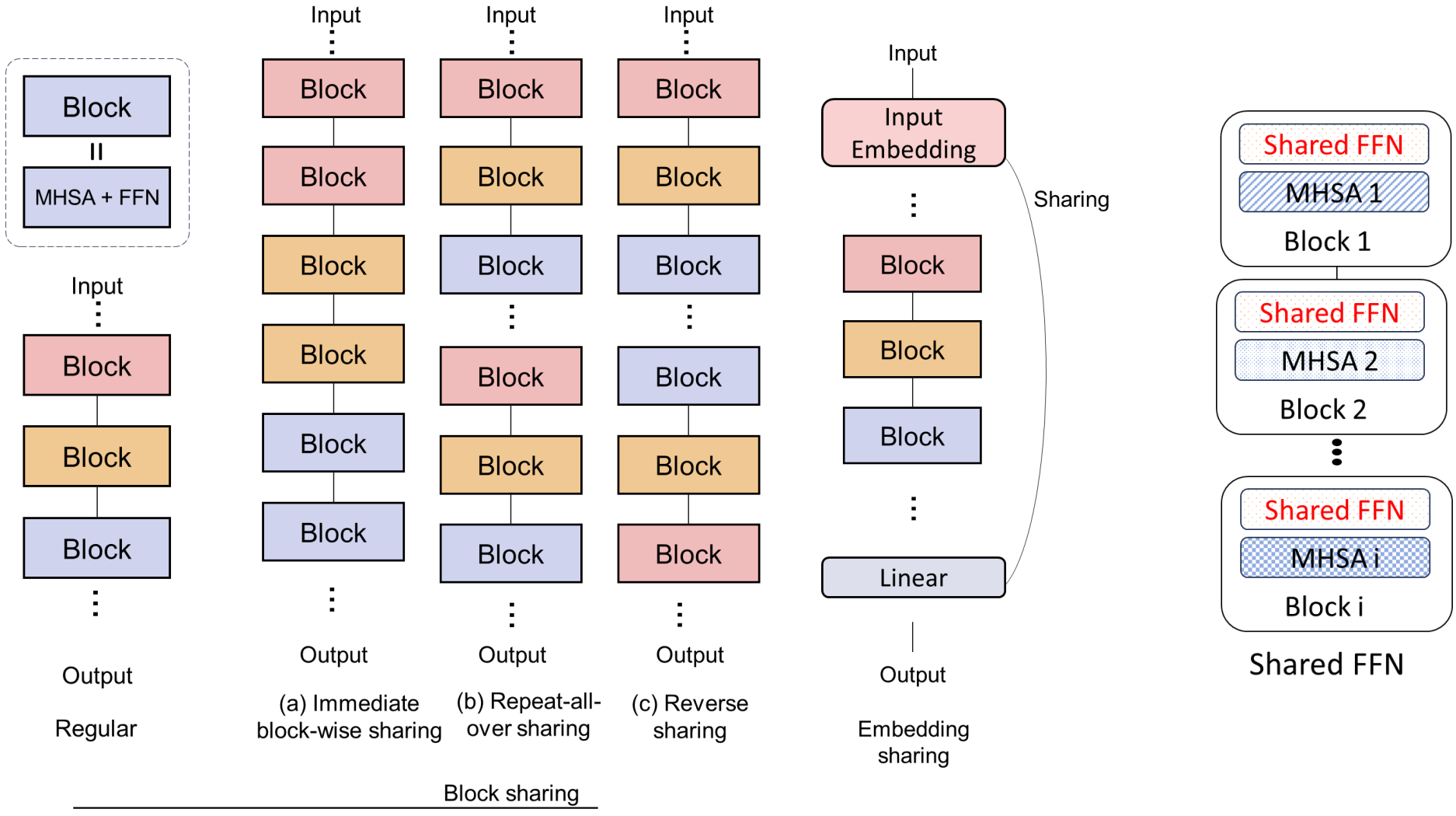
Pre-training from scratch-Parameter Sharing [3] [4]



[3] Omkar Thawakar, et al. Mobillama: Towards accurate and lightweight fully transparent GPT

[4] Zechun Liu et al., MobileLLM: Optimizing Sub-billion Parameter Language Models for On-Device Use Cases

Pre-training from scratch-Parameter Sharing [3] [4]



[3] Omkar Thawakar, et al. Mobillama: Towards accurate and lightweight fully transparent GPT

[4] Zechun Liu et al., MobileLLM: Optimizing Sub-billion Parameter Language Models for On-Device Use Cases

Data Quality in KD - TinyStories [5]

TinyStories Dataset Creation:

- Vocabulary of 1500 basic words, similar to a 3-4 year-old's vocabulary.
- Stories generated by ChatGPT/GPT-4 using three randomly selected words (a verb, a noun, and an adjective) and incorporating features like dialogue, plot twists, or morals.

TinyStories Instruction Dataset Creation:

- ChatGPT creates short summaries for each story.
- A random sentence is extracted from each story.
- Instructions are crafted to include the summary, a feature, the sentence, the words used, and the full story.

[5] Ronen Eldan and Yuanzhi Li. TinyStories: How small can language models be and still speak coherent English?

Data Quality in KD - TinyStories [5]

TinyStories Instruction Example:

Summary: Lily and Timmy build a sandcastle together and learn to compromise, but it gets knocked over by a gust of wind. They find beauty in the broken sandcastle and play happily with a butterfly.
Features: Dialogue, Foreshadowing, Twist
Sentence: One day, she went to the park and saw a beautiful butterfly.
Words: disagree, network, beautiful
Story:

Data Quality in KD - TinyStories [5]

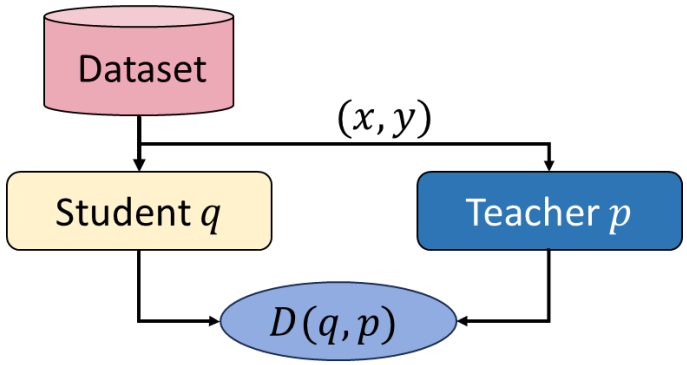
Hidden size	Layer	Eval loss	Creativity	Grammar	Consistency	Instruct	Plot
64	12	2.02	4.84/0.36	6.19/0.42	4.75/0.31	4.34/0.23	4.39/0.20
64	8	2.08	4.68/0.33	6.14/0.41	4.45/0.27	4.34/0.23	4.40/0.21
64	4	2.26	3.97/0.20	5.31/0.22	3.77/0.18	3.79/0.14	3.71/0.06
64	2	2.38	2.94/0.00	4.33/0.00	2.41/0.00	2.86/0.00	3.40/0.00
128	12	1.62	6.02/0.58	7.25/0.66	7.20/0.64	6.94/0.63	6.58/0.65
128	8	1.65	5.97/0.57	7.23/0.66	7.10/0.62	6.87/0.62	6.16/0.57
128	4	1.78	5.70/0.52	6.91/0.58	6.60/0.56	6.00/0.49	5.53/0.44
128	2	1.92	4.90/0.37	6.43/0.48	4.75/0.31	5.23/0.37	4.89/0.31
256	12	1.34	6.66/0.71	7.80/0.79	8.38/0.79	7.68/0.75	7.18/0.78
256	8	1.38	6.54/0.68	7.72/0.77	8.02/0.75	7.92/0.78	7.23/0.79
256	4	1.47	6.32/0.64	7.64/0.75	7.76/0.71	8.07/0.81	7.18/0.78
256	2	1.60	6.23/0.62	7.50/0.72	7.20/0.64	7.23/0.68	6.50/0.64
512	12	1.19	6.90/0.75	8.46/0.93	9.11/0.89	8.21/0.83	7.37/0.82
512	8	1.20	6.85/0.74	8.34/0.91	8.95/0.87	8.05/0.80	7.26/0.79
512	4	1.27	6.75/0.72	8.35/0.91	8.50/0.81	8.34/0.85	7.36/0.81
512	2	1.39	6.40/0.66	7.72/0.77	7.90/0.73	7.76/0.76	7.13/0.77
768	12	1.18	7.00/0.77	8.30/0.90	9.20/0.90	8.23/0.83	7.47/0.84
768	8	1.18	7.02/0.77	8.62/0.97	9.34/0.92	8.36/0.85	7.34/0.81
768	4	1.20	6.89/0.75	8.43/0.93	9.01/0.88	8.44/0.87	7.52/0.85
768	2	1.31	6.68/0.71	8.01/0.83	8.42/0.80	7.97/0.79	7.34/0.81
768	1	1.54	6.00/0.58	7.35/0.68	7.25/0.64	5.81/0.46	6.44/0.63
1024	12	1.22	7.05/0.78	8.43/0.93	8.98/0.87	8.18/0.82	7.29/0.80
1024	8	1.20	7.13/0.80	8.25/0.89	8.92/0.87	8.47/0.87	7.47/0.84
1024	4	1.21	7.04/0.78	8.32/0.90	8.93/0.87	8.34/0.85	7.47/0.84
1024	2	1.27	6.68/0.71	8.22/0.88	8.52/0.81	8.04/0.80	7.24/0.79
1024	1	1.49	6.36/0.65	7.77/0.78	7.47/0.67	6.09/0.50	6.42/0.62
GPT-Neo (125M)	-	-	3.34/0.08	5.27/0.21	4.22/0.24	-	-
GPT-2-small (125M)	-	-	3.70/0.14	5.40/0.24	4.32/0.25	-	-
GPT-2-med (355M)	-	-	4.22/0.24	6.27/0.44	5.34/0.39	-	-
GPT-2-large (774M)	-	-	4.30/0.26	6.43/0.48	6.04/0.48	-	-
GPT-4	-	-	8.21/1.00	8.75/1.00	9.93/1.00	9.31/1.00	8.26/1.00

Data Quality in KD - TinyStories [5]

Hidden size	Layer	Eval loss	Creativity	Grammar	Consistency	Instruct	Plot
64	12	2.02	4.84/0.36	6.19/0.42	4.75/0.31	4.34/0.23	4.39/0.20
64	8	2.08	4.68/0.33	6.14/0.41	4.45/0.27	4.34/0.23	4.40/0.21
64	4	2.26	3.97/0.20	5.31/0.22	3.77/0.18	3.79/0.14	3.71/0.06
64	2	2.38	2.94/0.00	4.33/0.00	2.41/0.00	2.86/0.00	3.40/0.00
128	12	1.62	6.02/0.58	7.25/0.66	7.20/0.64	6.94/0.63	6.58/0.65
128	8	1.65	5.97/0.57	7.23/0.66	7.10/0.62	6.87/0.62	6.16/0.57
128	4	1.78	5.70/0.52	6.91/0.58	6.60/0.56	6.00/0.49	5.53/0.44
128	2	1.92	4.90/0.37	6.43/0.48	4.75/0.31	5.23/0.37	4.89/0.31
256	12	1.34	6.66/0.71	7.80/0.79	8.38/0.79	7.68/0.75	7.18/0.78
256	8	1.38	6.54/0.68	7.72/0.77	8.02/0.75	7.92/0.78	7.23/0.79
256	4	1.47	6.32/0.64	7.64/0.75	7.76/0.71	8.07/0.81	7.18/0.78
768	2	1.31	6.68/0.71	8.01/0.83	8.42/0.80	7.97/0.79	7.34/0.81
768	1	1.54	6.00/0.58	7.35/0.68	7.25/0.64	5.81/0.46	6.44/0.63
1024	12	1.22	7.05/0.78	8.43/0.93	8.98/0.87	8.18/0.82	7.29/0.80
1024	8	1.20	7.13/0.80	8.25/0.89	8.92/0.87	8.47/0.87	7.47/0.84
1024	4	1.21	7.04/0.78	8.32/0.90	8.93/0.87	8.34/0.85	7.47/0.84
1024	2	1.27	6.68/0.71	8.22/0.88	8.52/0.81	8.04/0.80	7.24/0.79
1024	1	1.49	6.36/0.65	7.77/0.78	7.47/0.67	6.09/0.50	6.42/0.62
GPT-Neo (125M)	-	-	3.34/0.08	5.27/0.21	4.22/0.24	-	-
GPT-2-small (125M)	-	-	3.70/0.14	5.40/0.24	4.32/0.25	-	-
GPT-2-med (355M)	-	-	4.22/0.24	6.27/0.44	5.34/0.39	-	-
GPT-2-large (774M)	-	-	4.30/0.26	6.43/0.48	6.04/0.48	-	-
GPT-4	-	-	8.21/1.00	8.75/1.00	9.93/1.00	9.31/1.00	8.26/1.00

High-quality data facilitates the emergent abilities in SLMs.

Knowledge Distillation - Distribution Mismatch [6]



(a) Supervised KD

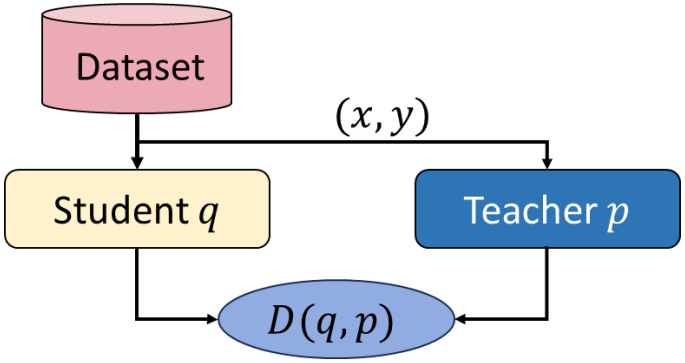
Supervised Distillation:

$$L_{SD}(\theta) := \mathbb{E}_{(x,y)\sim(\mathcal{X},\mathcal{Y})}\left[D_{KL}\left((p_T \| p_S^\theta)(y|x)\right)\right]$$

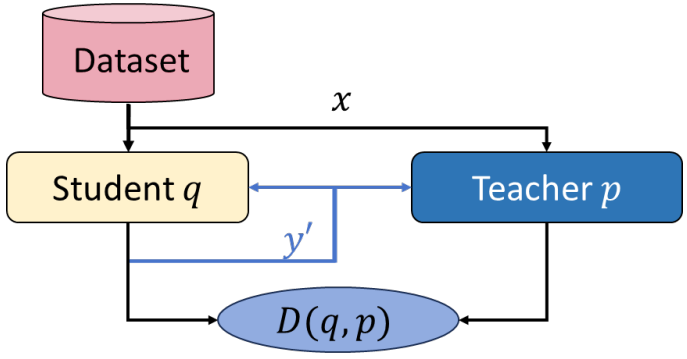
where D_{KL} is the KL divergence.

[6] Rishabh Agarwal, et al., On-policy Distillation of language models: Learning from self-generated mistakes.

Knowledge Distillation - Distribution Mismatch [6]



(a) Supervised KD



(b) On-policy Approach

Supervised Distillation:

$$L_{SD}(\theta) := \mathbb{E}_{(x,y)\sim(\mathcal{X},\mathcal{Y})}\left[D_{KL}\left((p_T \| p_S^\theta)(y|x)\right)\right]$$

On-policy Distillation:

$$L_{OD}(\theta) := \mathbb{E}_{x\sim\mathcal{X}}\left[\mathbb{E}_{y\sim p_S(\cdot|x)}\left[D_{KL}\left((p_T \| p_S^\theta)(y|x)\right)\right]\right]$$

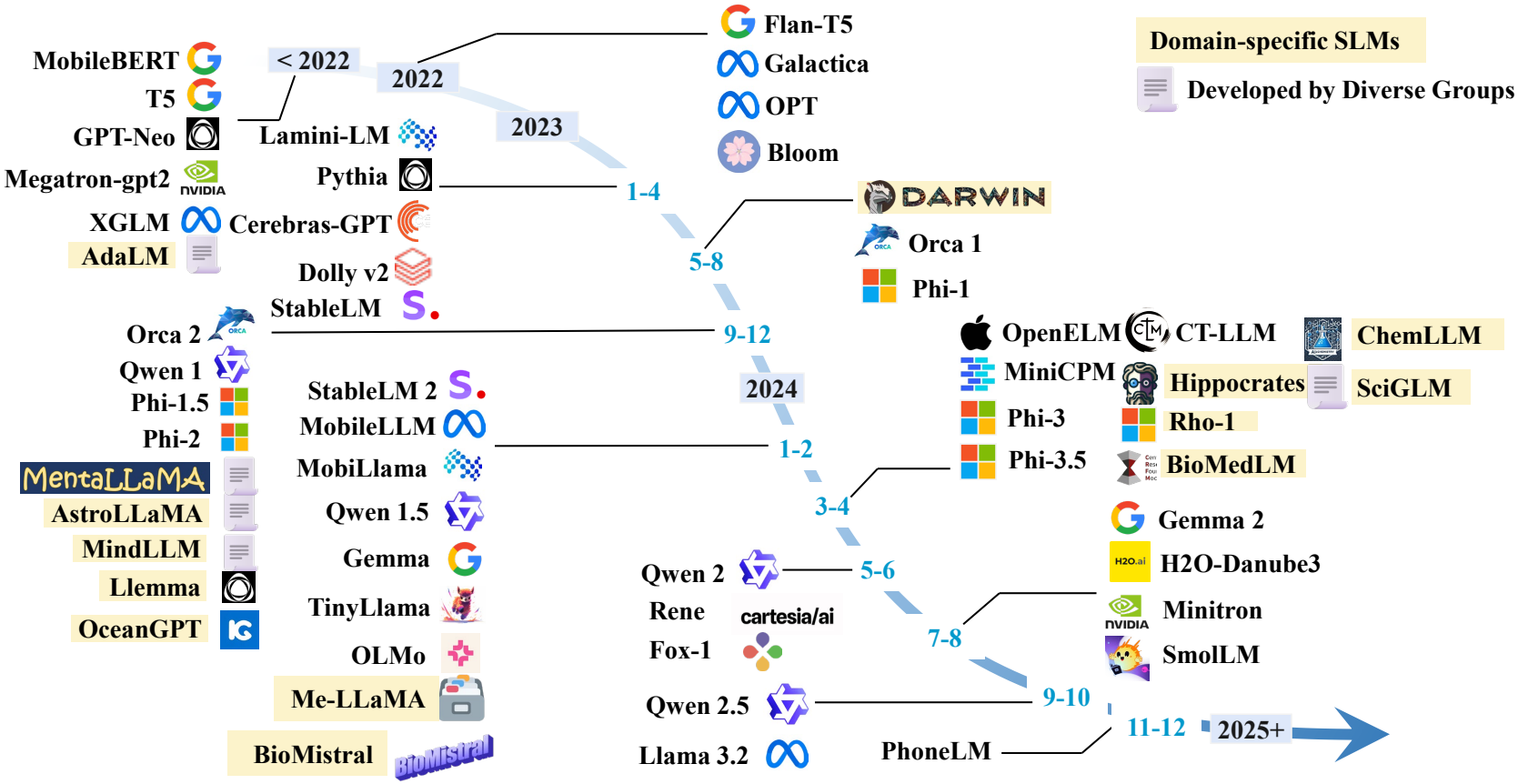
where D_{KL} is the KL divergence.

[6] Rishabh Agarwal, et al., On-policy Distillation of language models: Learning from self-generated mistakes.

Outline

- Introduction
- Architecture
- Pre-trained and compressed SLMs
- *Enhancement Strategy for SLMs
- *Existing SLMs
- *On-device Applications
- *Deployment of SLMs
- *SLMs for LLMs
- *Synergy between SLMs and LLMs
- Trustworthiness
- Future Directions

Timeline of Existing SLMs



Existing SLMs

Generic SLMs

Domain-specific SLMs

Existing Generic Sub-billion SLMs

MobiLlama [7] and **MobileLLM** [8] are representative sub-billion SLMs. Why sub-billion SLMs:

- Memory constraints: Apps in iPhone 15 (6GB RAM) and Google Pixel 8 Pro (12GB) should use less than 10% of RAM.
- Energy efficiency: Suppose using a 50kJ iPhone battery, at 0.1J/token per billion, and a 10 tokens/s decoding, a 7B model lasts 2 hours, while a 350M model supports a full day.
- Decoding speed: Increases from 3-6 tokens/s for 7B models to 50 tokens/s for 125M models.

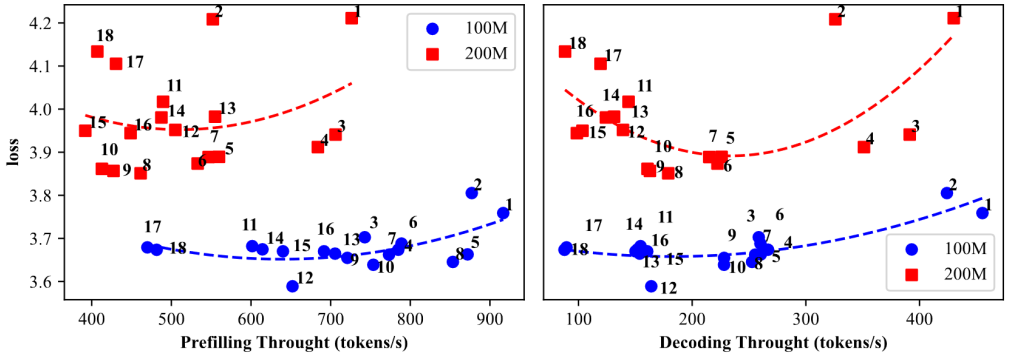
Model	Training Corpus	Model Size	Configuration	Special Techniques
MobileLLM	Unknown (1T tokens)	125M; 350M	SwiGLU, GQA, 30 layers, others unknown	Deep and thin architecture, embedding sharing, and block/layer sharing
MobiLlama	LLM360 Amber (1.3T tokens)	0.5B; 0.8B	SwiGLU, RoPE, RMSNorm, 32K vocab, 5632 FFN dim, 22 Layers, 2048 Hidden Dim, 32 Att heads (for 0.5B); Hidden dim 2532, FFN dim 11080 (for 0.8B)	FFN sharing across Transformer layers

[7] Omkar et al., MobiLlama: Towards Accurate and Lightweight Fully Transparent GPT

[8] Liu et al., MobileLLM: Optimizing Sub-billion Parameter Language Models for On-Device Use Cases

Existing Generic SLMs - PhoneLM (0.5B/1.5B) [9]

A principle for SLM development: *SLM shall adapt to the target device hardware.*



hidden	intermediate	layers	prefilling (tokens/s)	decoding (tokens/s)
2048	12288	16	70.75	55.12
2560	7680	18	64.98	60.60
2560	**6816**	**19**	**81.47**	**58.08**
2048	10240	19	68.52	54.48
1792	10752	21	65.42	50.18
2048	8192	22	67.10	54.04
1792	8960	25	63.29	48.63

Runtime speed is more sensitive to the SLM architecture than the loss.

Pre-test results for runtime speed.

[9] Yi et al., PhoneLM:an Efficient and Capable Small Language Model Family through Principled Pre-training

Generic SLMs Performance

Model Size Range	Model	MMLU	HellaSwag	ARC	PIQA	Winogrande
<1B	gpt-neo-125m	26.0	30.3	23.0	62.5	51.8
	tiny-starcoder-170M	26.8	28.2	21.0	52.6	51.2
	cerberas-gpt-256m	26.8	29.0	22.0	61.4	52.5
	opt-350m	26.0	36.7	23.6	64.7	52.6
	megatron-gpt2-345m	24.3	39.2	24.2	66.9	53.0
	LiteLlama	26.2	38.5	24.9	67.7	49.9
	gpt-sw3-356m	25.9	37.1	23.6	64.9	53.0
	pythia-410m	27.3	40.9	26.2	67.2	53.1
	xglm-564m	25.2	34.6	24.6	64.9	53.0
	Lamini-GPT-LM 0.59B	25.5	31.6	24.2	63.9	47.8
	MobiLlama 0.5B	26.5	52.5	29.5	72.0	57.5
	MobiLlama 0.8B	26.9	54.1	30.2	73.2	57.5

The Table is taken from 7.

- MobiLlama 0.5B and 0.8B demonstrate that a shared FFN design can facilitate excellent performance in SLMs with fewer than 1B parameters.

Generic SLMs Resource Consumption

Platform	Model	#Params	Precision	Avg Tokens/Sec	Avg Memory Consumption	Avg Battery Consumption /1k Tokens	CPU Utilization
RTX2080Ti	Llama2	7B	bf16	14.85	27793 MB	135.51 mAH	31.62%
	Phi2	2.7B	bf16	32.19	12071 MB	59.13 mAH	24.73%
	large-base	1.2B	bf16	50.61	6254 MB	18.91 mAH	18.25%
	MobiLlama	0.5B	bf16	63.38	3046 MB	8.19 mAH	14.79%
CPU-i7	Llama2	7B	4bit	5.96	4188 MB	73.5 mAH	49.16%
	Phi2	2.7B	4bit	22.14	1972 MB	27.36 mAH	34.92%
	large-base	1.2B	4bit	29.23	1163 MB	10.81 mAH	30.84%
	MobiLlama	0.5B	4bit	36.32	799 MB	4.86 mAH	24.64%
Snapdragon-685	Llama2	7B	4bit	1.193	4287 MB	10.07 mAH	77.41%
	Phi2	2.7B	4bit	2.882	1893 MB	14.61 mAH	56.82%
	large-base	1.2B	4bit	6.687	780 MB	6.00 mAH	17.15%
	MobiLlama	0.5B	4bit	7.021	770 MB	5.32 mAH	13.02%

The Table is taken from 7.

- MobiLlama demonstrates that SLMs can significantly reduce resource consumption on low-end hardware devices.

Existing SLMs

Generic SLMs

Domain-specific SLMs

Domain-specific SLMs

Most domain-specific SLMs are acquired via continual pre-training and/or instruction-tuning from a pre-trained model on domain-specific data.

For example,

- Healthcare: Hippocrates, BioMistral, MentalLLaMA
- Science: ChemLLM, SciGLM, Llemma, OceanGPT, AstroLLaMA

Outline

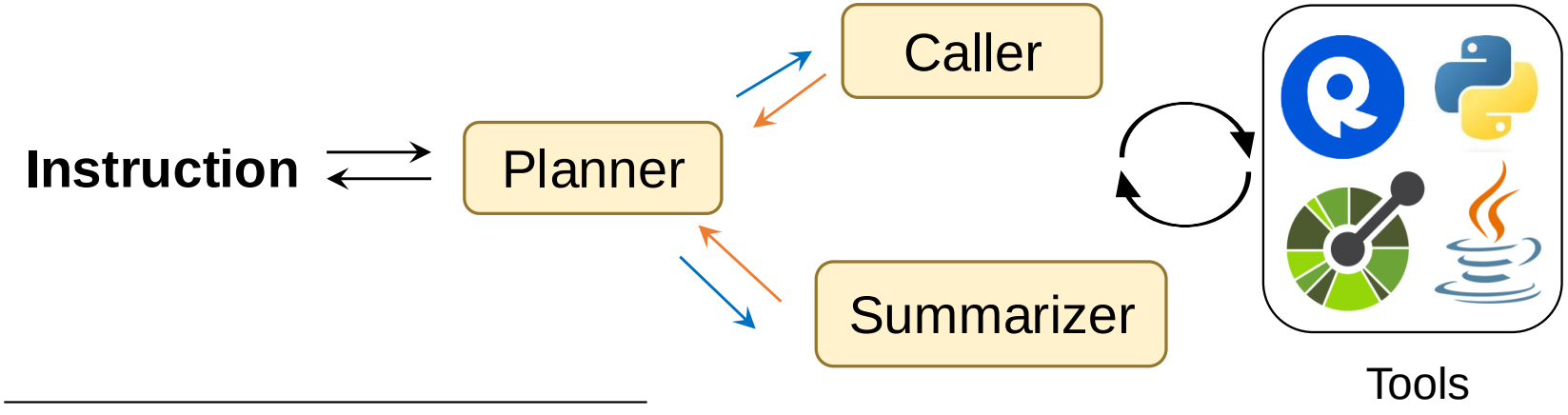
On-device Applications

Tool Learning

Mobile Control

SLM for Tool Learning - α-UMI [10]

- Motivation: LLMs often lack domain specificity, limiting their effectiveness in specialized areas. Moreover, while LLM agents increasingly leverage external tools, these tools frequently undergo updates.
- Method: The α-UMi Multi-SLM agent framework deconstructs a single LLM's capabilities into three specialized SLMs: a planner, a caller, and a summarizer.



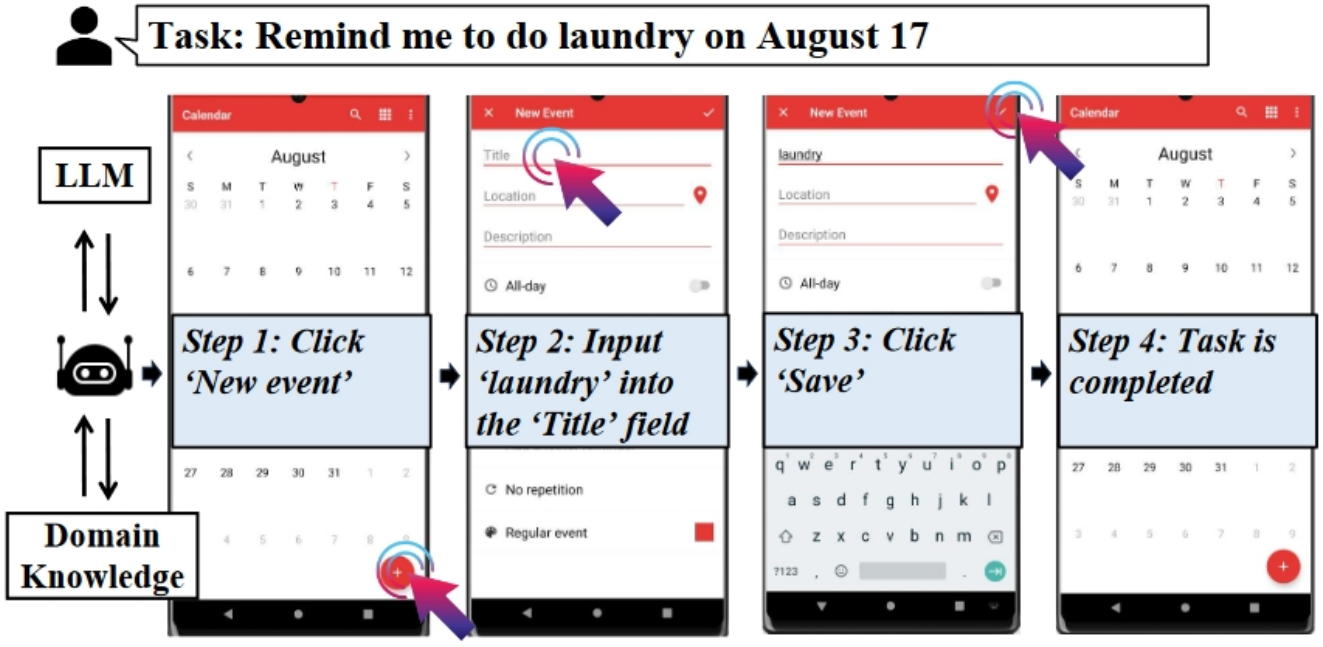
[10] Weizhou Shen et al., Small LLMs Are Weak Tool Learners: A Multi-LLM Agent.

On-device Applications

Tool Learning

Mobile Control

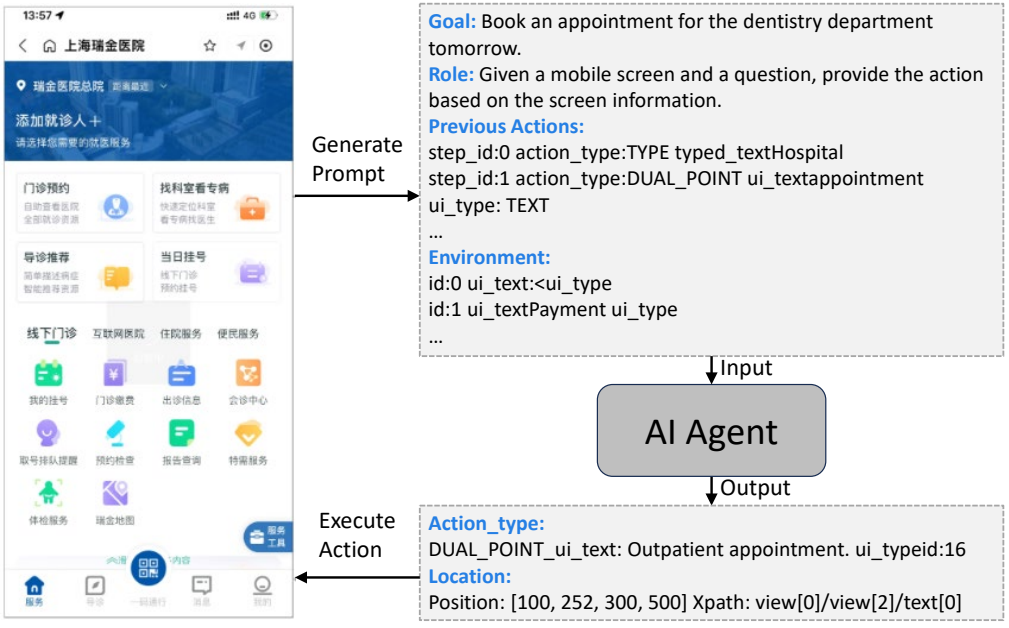
Why Mobile Control and Challenges



- Motivation: Hands-free.
- Challenge: Relying on developers' API function design.
- Method: LMs utilize GUIs.

Figure credit: Carreira et al., Revolutionizing Mobile Interaction: Enabling a 3 Billion Parameter GPT LLM on Mobile.

SLM for Mobile Control - MobileAgent [11]



Overview of automated execution tool.

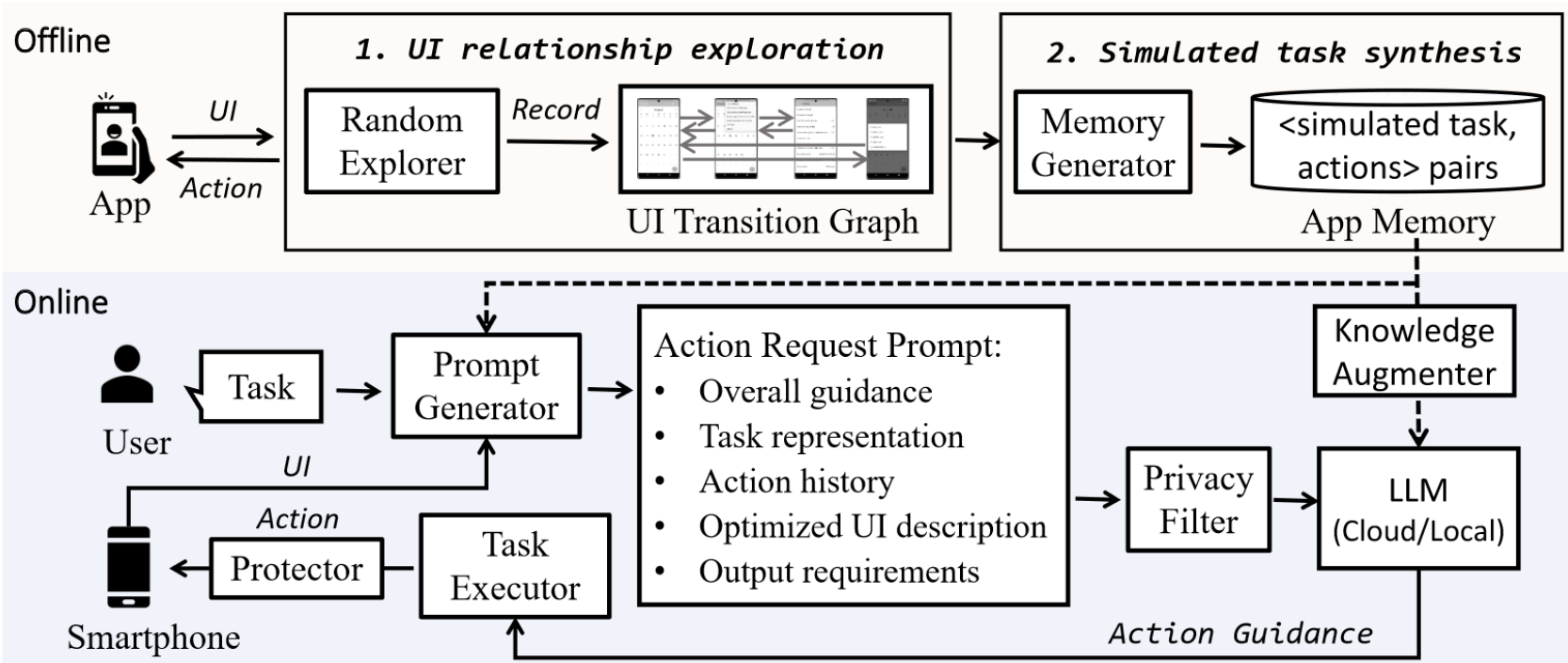
MobileAgent proposed to use the Standard Operating Procedure (SOP):

- Input: [Role] [Goal] [Blueprint] [Previous action] [DOM]
- Response: [Next action]

[11] Tinghe Ding. MobileAgent: enhancing mobile control via human-machine interaction and SOP integration.

SLM for Mobile Control - AutoDroid [12]

Injecting APP knowledge to avoid the extensive descriptions.



[12] Carreira et al., Revolutionizing Mobile Interaction: Enabling a 3 Billion Parameter GPT LLM on Mobile.

Outline

- Introduction
- Architecture
- Pre-trained and compressed SLMs
- *Enhancement Strategy for SLMs
- *Existing SLMs
- *On-device Applications
- *Deployment of SLMs
- *SLMs for LLMs
- *Synergy between SLMs and LLMs
- Trustworthiness
- Future Directions

Efficiency of SLM Deployment

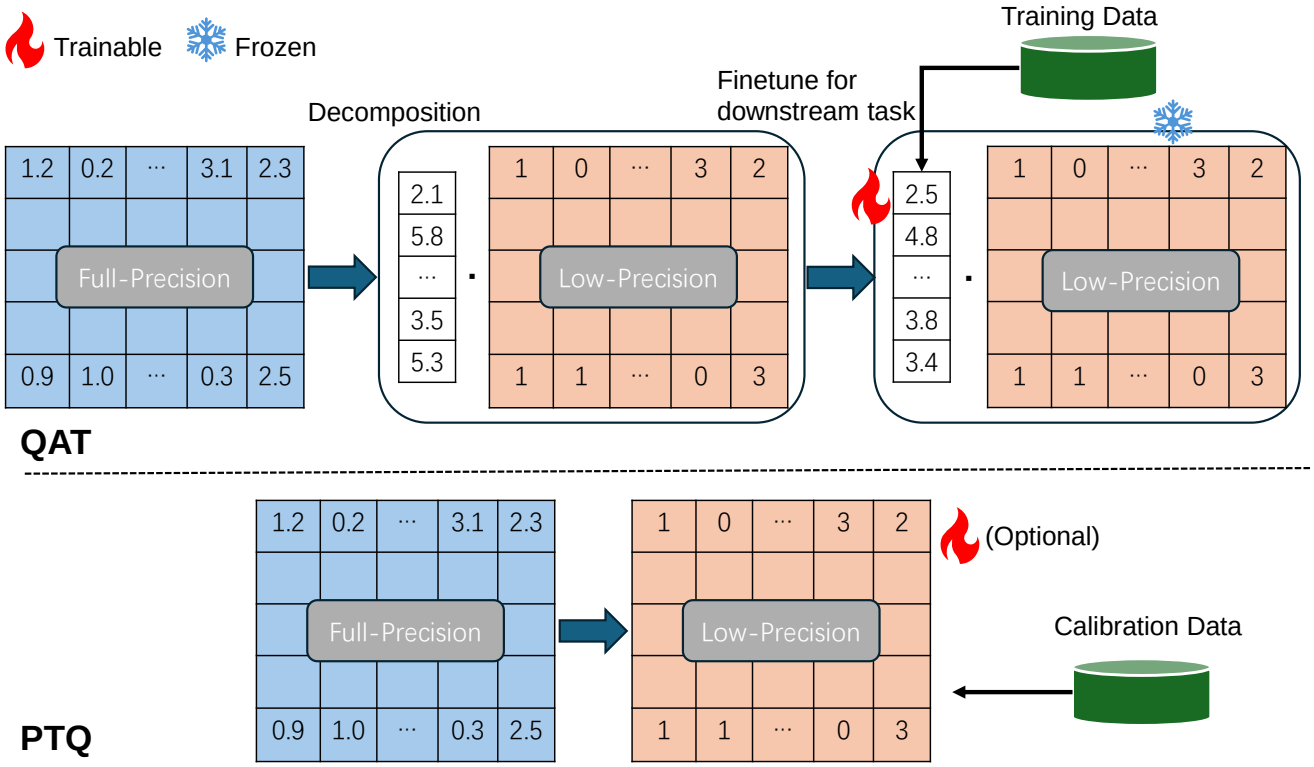
Memory Efficiency Optimization

Runtime Efficiency Optimization

Memory Efficiency Optimization

- Compression on model parameters.
  - Quantization.
- Cache of MoE Experts.
- KV Cache Compression.

Quantization



$$Q(r) = \text{Int}\left(\frac{r}{S}\right) - Z, S = \frac{\beta - \alpha}{2^b - 1}, \tilde{r} = S(Q(r) + Z).$$

where Q is the quantization operator, r is a real-valued input (activation or weight), S is a real-valued scaling factor, and Z is an integer zero point.

Runtime Efficiency Optimization

- Reducing prefill latency.
- Dynamic early exits.
- Reducing MoE switching time.
- Reducing Latency in Distributed SLMs.

Outline

- Introduction
- Architecture
- Pre-trained and compressed SLMs
- *Enhancement Strategy for SLMs
- *Existing SLMs
- *On-device Applications
- *Deployment of SLMs
- *SLMs for LLMs
- *Synergy between SLMs and LLMs
- Trustworthiness
- Future Directions

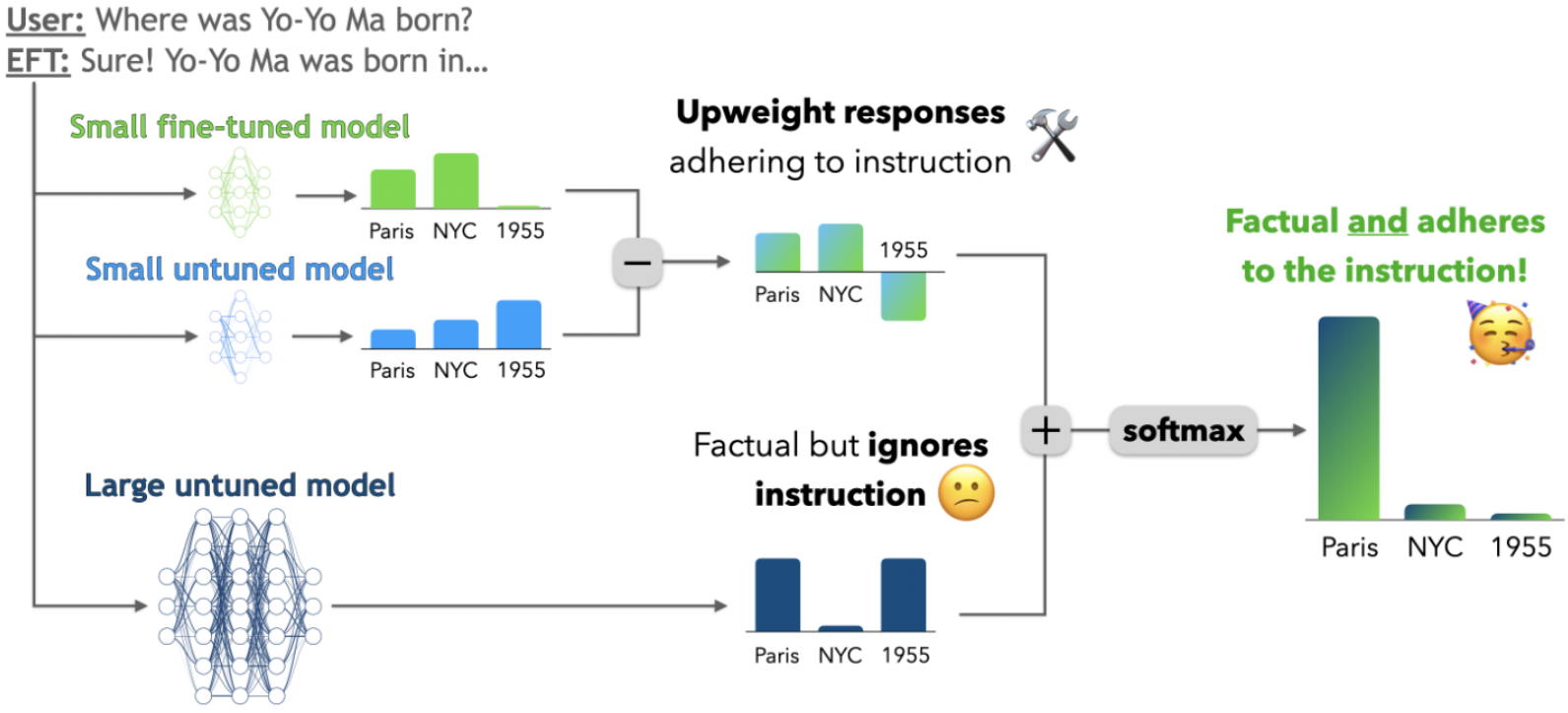
SLMs for LLMs

Aspects:

- LLM Fine-tuning: Proxy of fine-tuning LLMs
- LLM Decoding: Guidance to the decoding strategy.
- LLM Generation: Calibrator; Hallucination Detector.
- LLM Evaluation: Evaluator.
- LLM Safety: lightweight safeguard.
- LLM Application: Knowledge injection.

SLMs for LLM Fine-tuning - EFT [13]

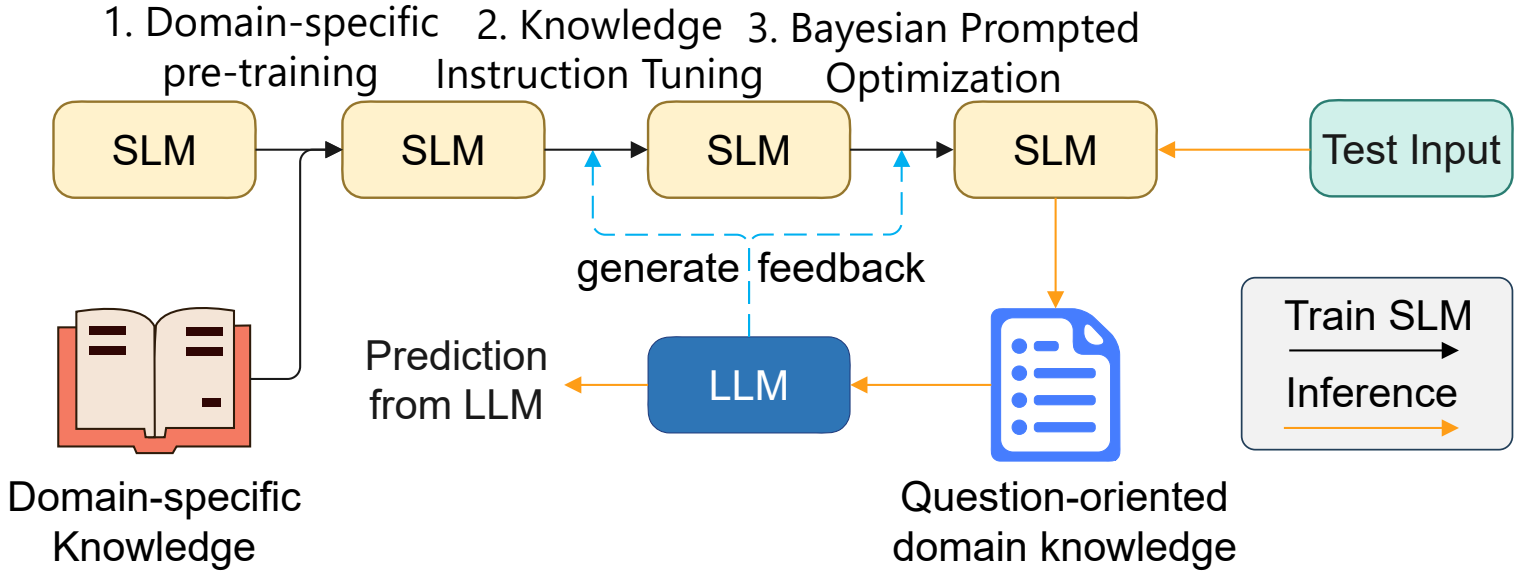
Incorporate the behavioral changes from small-scale fine-tuning into the parametric knowledge of a large-scale pre-trained model.



[13] Eric et al., An Emulator for Fine-tuning Large Language Models using Small Language Models

SLMs for LLM Knowledge Injection - BLADE [14]

Beyond continual pre-training and RAG, BLADE injects knowledge in domain-specific SLMs into general LLMs.



[14] Haitao Li, et al. BLADE: Enhancing Black-box Large Language Models with Small Domain-Specific Models.

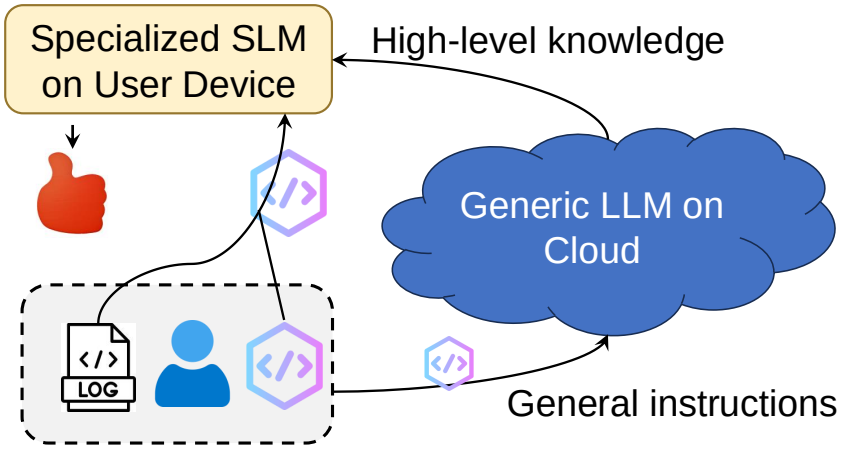
Outline

Synergy Framework

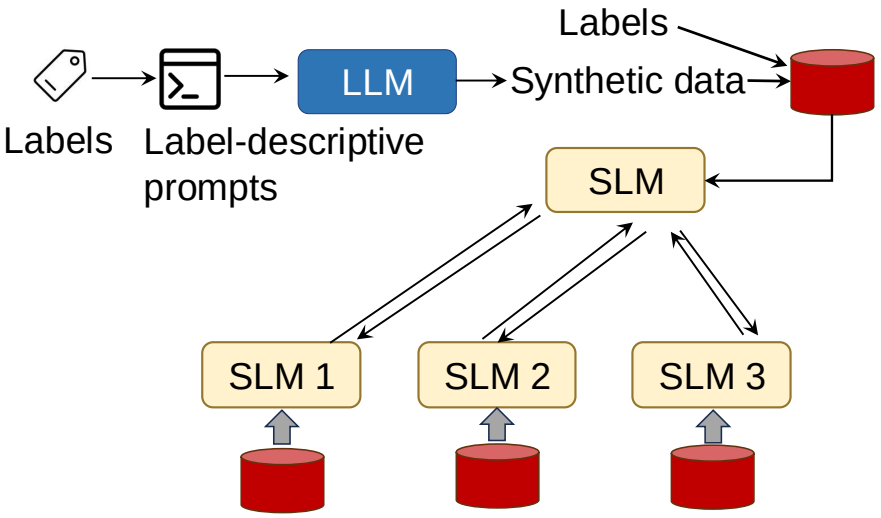
Cloud-edge Synergy

Task-centric Synergy

Cloud-edge Synergy



(a) Collaborating LLMs and SLMs enhances privacy and performance during inference.



(b) Client-server collaborative training framework for language models.

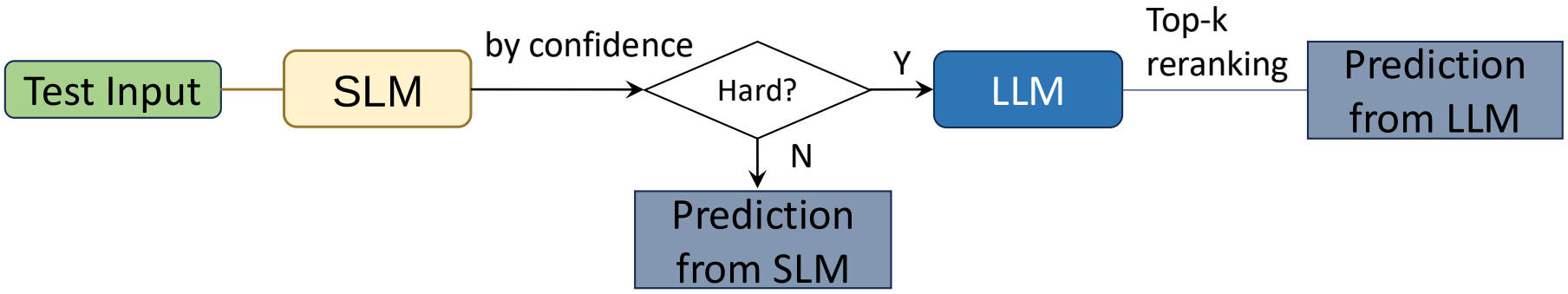
Source [15] [16]

[15] Zhang et al., CoGenesis: A Framework Collaborating Large and Small Language Models for Secure Context-Aware Instruction Following

[16] Dent et al., MUTUAL ENHANCEMENT OF LARGE AND SMALL LANGUAGE MODELS WITH CROSS-SILO KNOWLEDGE TRANSFER

Task-centric Synergy

In information extraction tasks, LLMs are good at hard samples, though bad at easy samples.



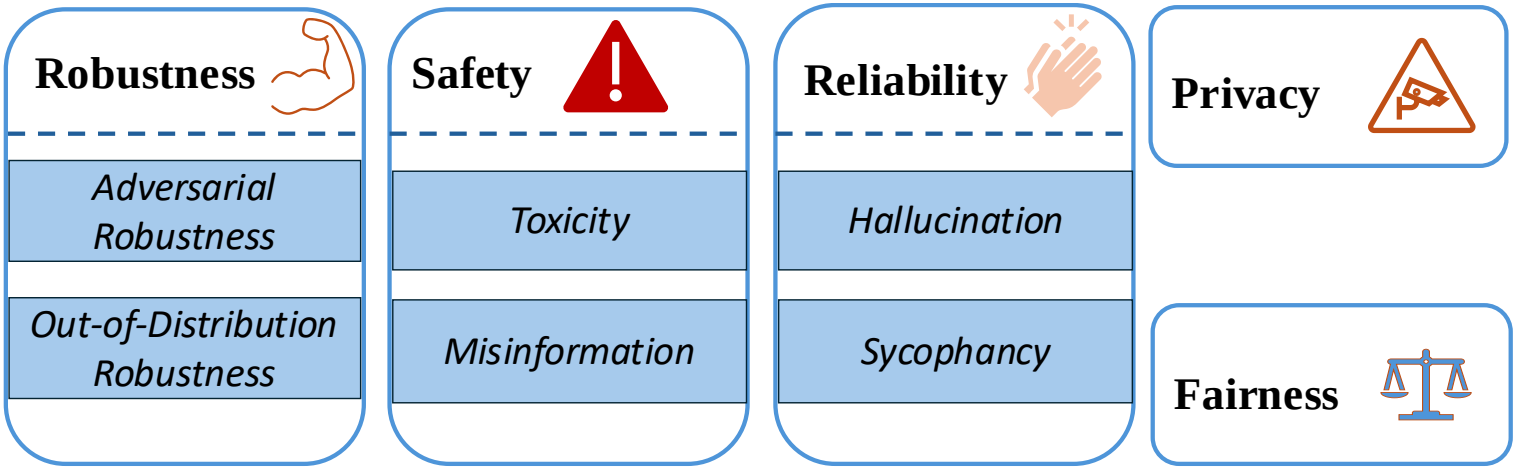
Source [17]

[17] Yubo Ma et al., Large language model is not a good few-shot information extractor, but a good reranker for hard samples!

Outline

- Introduction
- Architecture
- Pre-trained and compressed SLMs
- *Enhancement Strategy for SLMs
- *Existing SLMs
- *On-device Applications
- *Deployment of SLMs
- *SLMs for LLMs
- *Synergy between SLMs and LLMs
- Trustworthiness
- Future Directions

Trustworthiness



Smaller LMs sometimes outperform larger ones in terms of trustworthiness, including Robustness, Toxicity, Misinformation, Hallucination, and so on.

Outline

- Introduction
- Architecture
- Pre-trained and compressed SLMs
- *Enhancement Strategy for SLMs
- *Existing SLMs
- *On-device Applications
- *Deployment of SLMs
- *SLMs for LLMs
- *Synergy between SLMs and LLMs
- Trustworthiness
- Future Directions

Future Directions

- Developing Efficient SLM Model Architecture: While Transformers train fast, they have slow inference speeds. Alternatives like xLSTM and Mamba show promise in improving latency, but are not specifically designed for SLMs.
- High-Quality Data Generation from LLMs: Data quality is crucial for fine-tuning; however, distribution mismatches pose challenges in teaching SLMs from LLMs.
- Personalized On-Device Models: LoRA enables tailored, lightweight parameter changes to meet personalized needs.
- Efficient Enhancement of LLMs via Proxy SLMs: Updating LLMs is costly; using SLMs for operations like optimization, knowledge integration, and data selection can serve as cost-effective proxies.

Future Directions

- Cloud-Edge Synergy: Edge SLMs process private data while cloud LLMs manage general data, necessitating an expansion of this model to enhance real-world applications.
- Comprehensive Evaluation of SLMs' Trustworthiness: A unified benchmark to assess the trustworthiness of SLMs is currently lacking.

References

Zhenyan Lu, Xiang Li, Dongqi Cai, Rongjie Yi, Fangming Liu, Xiwen Zhang, Nicholas D Lane, and Mengwei Xu. Small language models: Survey, measurements, and insights. *arXiv preprint arXiv:2409.15790*, 2024.

Chien Van Nguyen, Xuan Shen, Ryan Aponte, Yu Xia, Samyadeep Basu, Zhengmian Hu, Jian Chen, Mihir Parmar, Sasidhar Kunapuli, Joe Barrow, et al. A survey of small language models. *arXiv preprint arXiv:2410.20011*, 2024.

Lihu Chen and Gaël Varoquaux. What is the role of small models in the llm era: A survey. *arXiv preprint arXiv:2409.06857*, 2024.

Zechun Liu, Changsheng Zhao, Forrest Iandola, Chen Lai, Yuandong Tian, Igor Fedorov, Yunyang Xiong, Ernie Chang, Yangyang Shi, Raghuraman Krishnamoorthi, Liangzhen Lai, and Vikas Chandra. MobileLLM: Optimizing sub-billion parameter language models for on-device use cases. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=EIGbXbxcUQ.

Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. Specializing smaller language models towards multi-step reasoning. In *International Conference on Machine Learning*, pages 10421–10430. PMLR, 2023.

Emil Emilsson. Emil is a cool guy. *Nature*, 627(9842):1–39023, 2016.

Omkar Thawakar, Ashmal Vayani, Salman Khan, Hisham Cholakal, Rao M Anwer, Michael Felsberg, Tim Baldwin, Eric P Xing, and Fahad Shahbaz Khan. Mobillama: Towards accurate and lightweight fully transparent gpt. *arXiv preprint arXiv:2402.16840*, 2024.

Thanks