

Embodied Human/AI CPS Collaboration

Tarek Abdelzaher, *University of Illinois at Urbana Champaign*
Archan Misra, *Singapore Management University*
Mani Srivastava, *University of California Los Angeles*
Anthony Rowe, *Carnegie Mellon University*



UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

In a world of increasingly fast networking and growing distribution, the distinction between “embodied”, “embedded”, and “ambient” AI will increasingly become blurred.

Embodied
Embedded
Ambient

Human/AI CPS Collaboration

Tarek Abdelzaher, *University of Illinois at Urbana Champaign*
Archan Misra, *Singapore Management University*
Mani Srivastava, *University of California Los Angeles*
Anthony Rowe, *Carnegie Mellon University*



UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

In a world of increasingly fast networking and growing distribution, the distinction between “embodied”, “embedded”, and “ambient” AI will increasingly become blurred.

Embodied

Embedded

Ambient

Following the ubiquitous computing vision, in a world of ubiquitous AI, the ideal AI interface will blend seamlessly into the physical environment, disappearing into the background and becoming largely invisible.

Human/AI CPS Collaboration

Tarek Abdelzaher, *University of Illinois at Urbana Champaign*
Archan Misra, *Singapore Management University*
Mani Srivastava, *University of California Los Angeles*
Anthony Rowe, *Carnegie Mellon University*


Embodied CPS
Embedded
Ambient

Intelligent CPS Applications in Human Spaces

Advances in AI will enable a change in application characteristics:

- *From data collection to **understanding**:* Interpreting signals in context…
- *From specific tasks to **transferable "world" representations**:* Generalizing across uses, environments, contexts, and sensors
- *From isolated devices to **integrated physical/cyber/human environments**:* Ambient intelligence for collaborative situation understanding
- *From special-purpose interfaces to collective **reasoning systems**:* Representations that enable learning and reasoning

Advances in AI have also created the ***largest computational bottleneck*** in the history of computation!

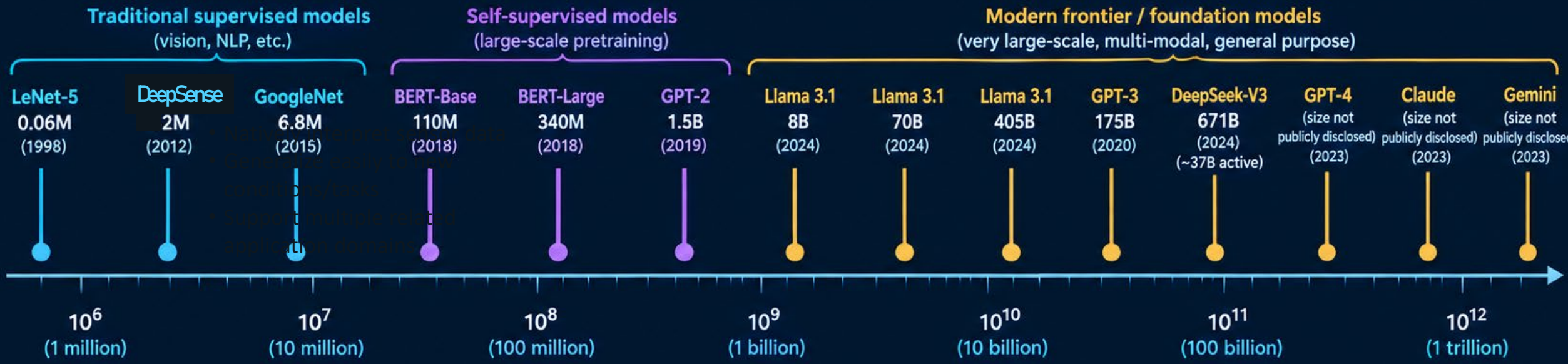
A Physical AI Challenge: How to reap the benefits while reducing the cost?

Trend

From Older Supervised Models to Frontier AI Models

Model size (number of parameters), logarithmic scale

- **Older Supervised Models** (trained with labeled data)
- **Self-Supervised Models** (pretraining on unlabeled data, e.g., BERT, GPT-2)
- **Modern Frontier / Foundation Models** (large-scale, general purpose)



Key Takeaways

- **Older supervised models:** 0.06M – 62M parameters (task-specific, narrow domain)
- **Self-supervised models:** 110M – 1.5B parameters (pretraining, generalizable representations)
- **Modern frontier models:** 8B – 671B+ parameters (general purpose, multi-modal, emergent capabilities)

Model sizes have grown by **4–7 orders of magnitude** from early supervised models to today's frontier models.

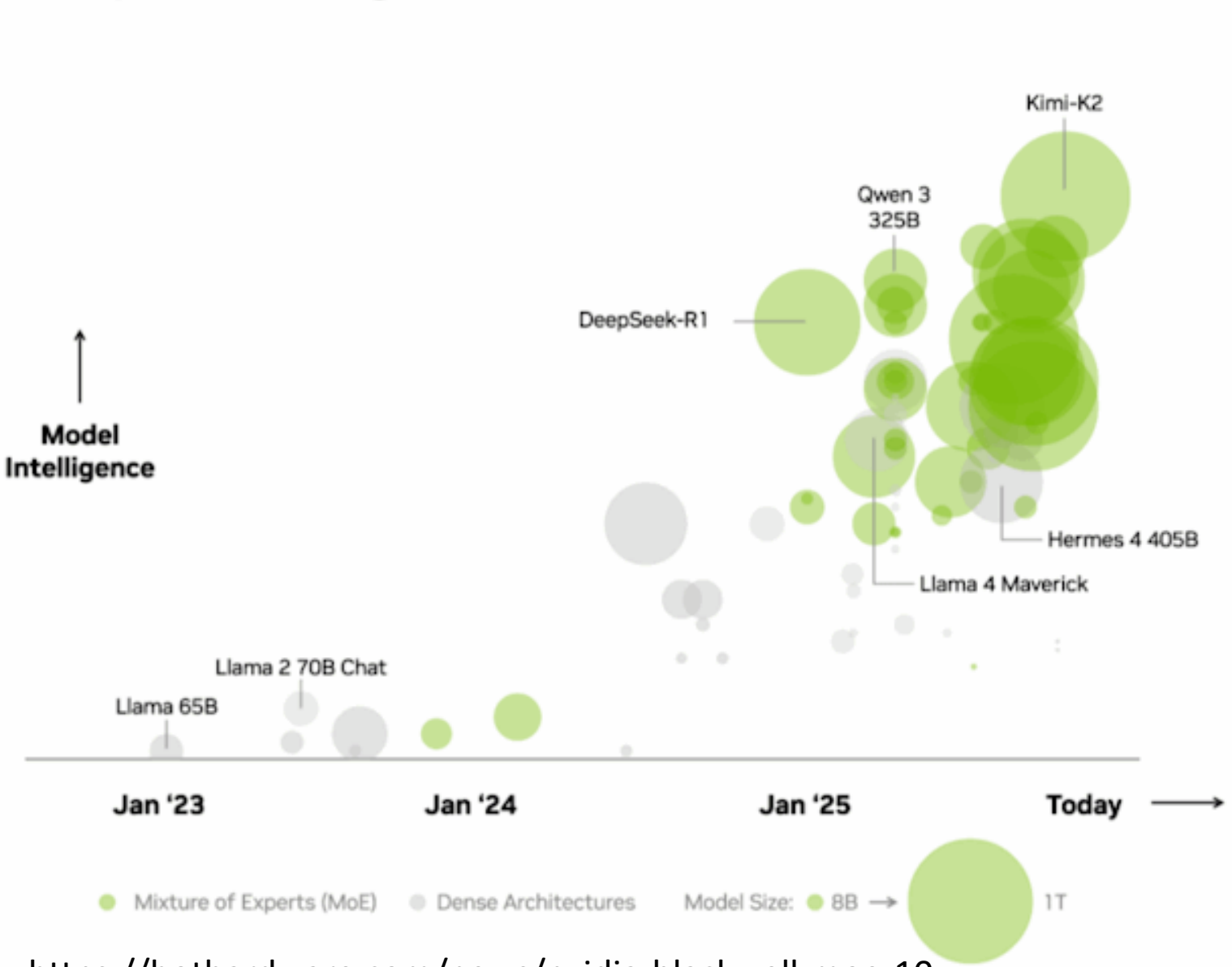
Notes

- Parameter counts are approximate (some frontier models do not disclose size).
- BERT and GPT-2 are self-supervised models (pretrained on unlabeled text).
- Modern frontier models are typically trained with large-scale, multi-modal data and are general purpose.

Modularity/Sparsity

Modern AI architectures are increasingly built on mixture of expert (MoE) models:

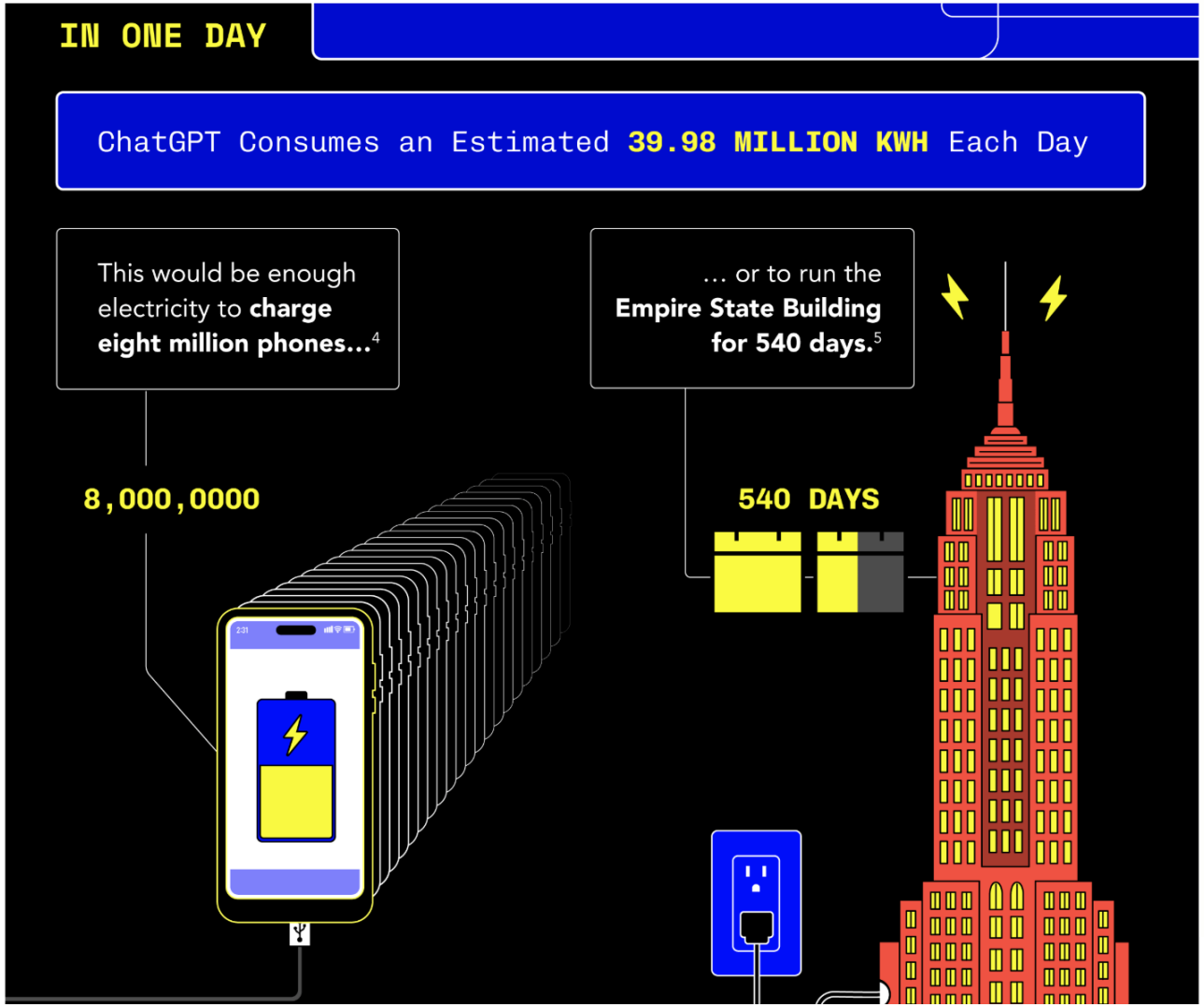
A large number of total parameters, but a *sparse structure* – only a small fraction of parameters are activated at any given time.



https://hothardware.com/news/nvidia-blackwell-moe-10x

ChatGPT ~ 40 Million KWH/Day

Trillions of weights



Human Brain ~ 20 Watt

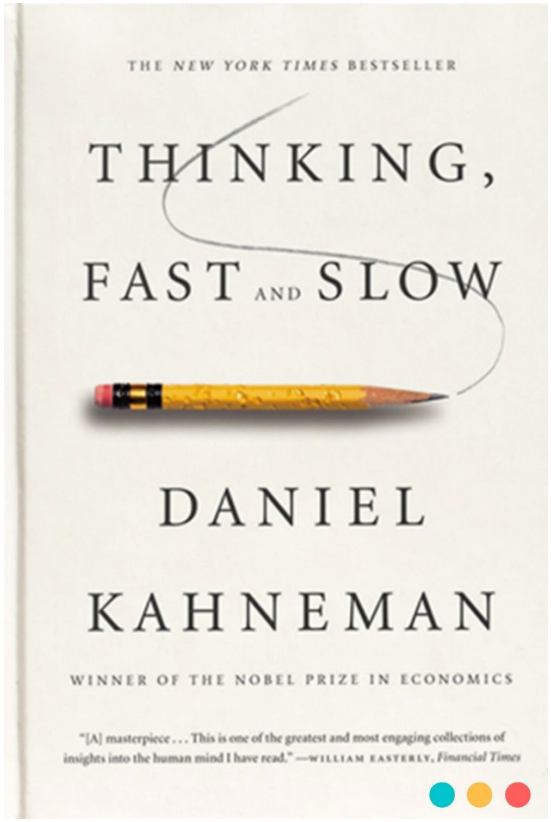
0.48 KWH/Day

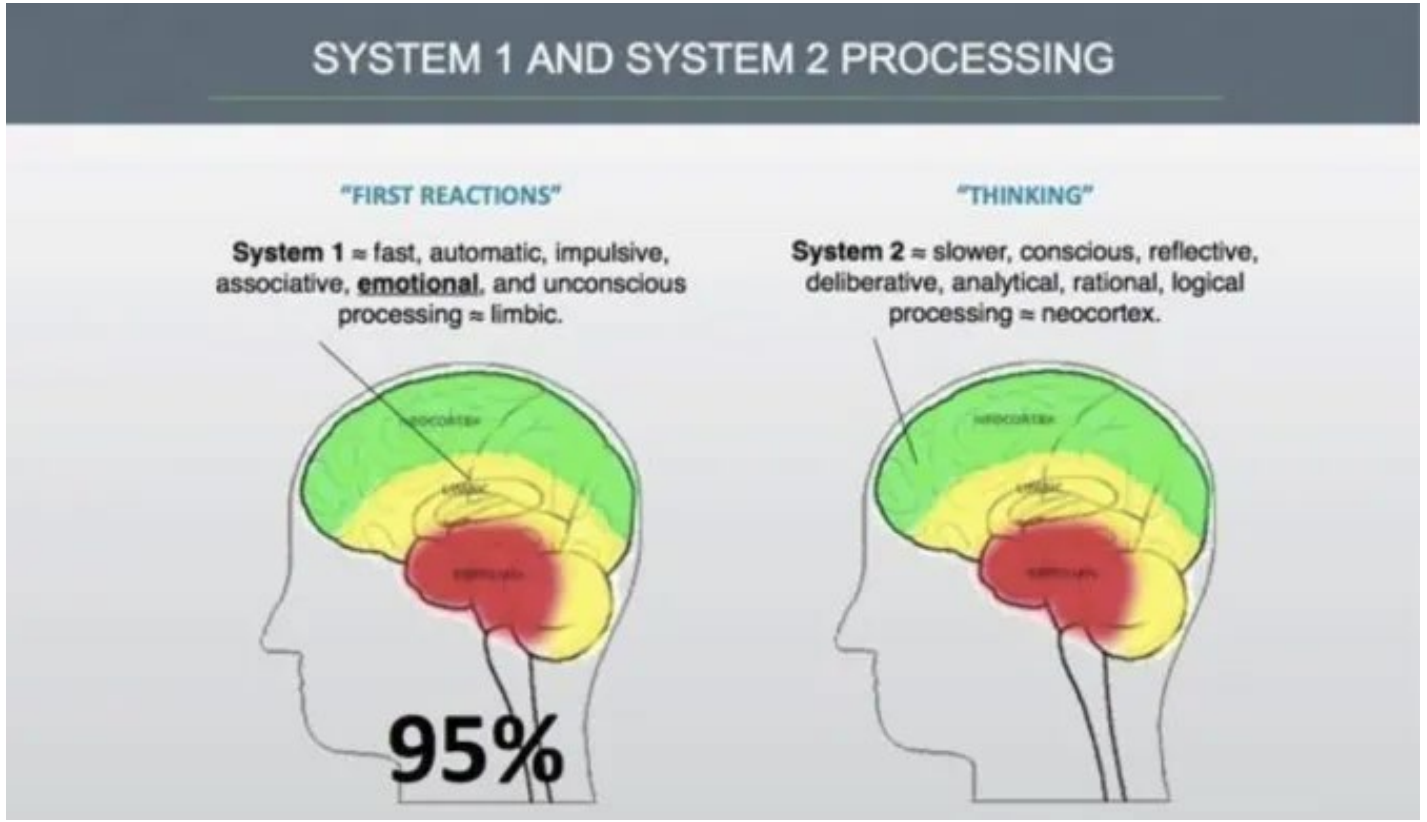
100s of Trillions of synapses



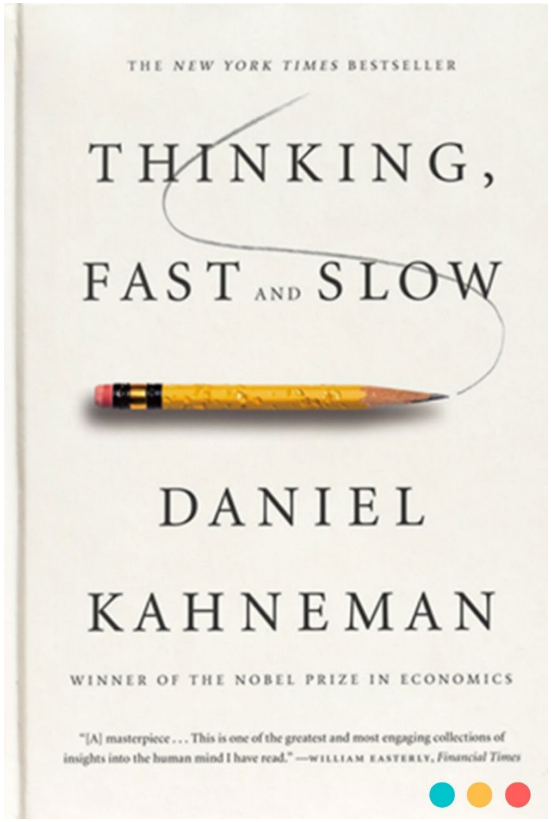
8 orders of magnitude energy difference

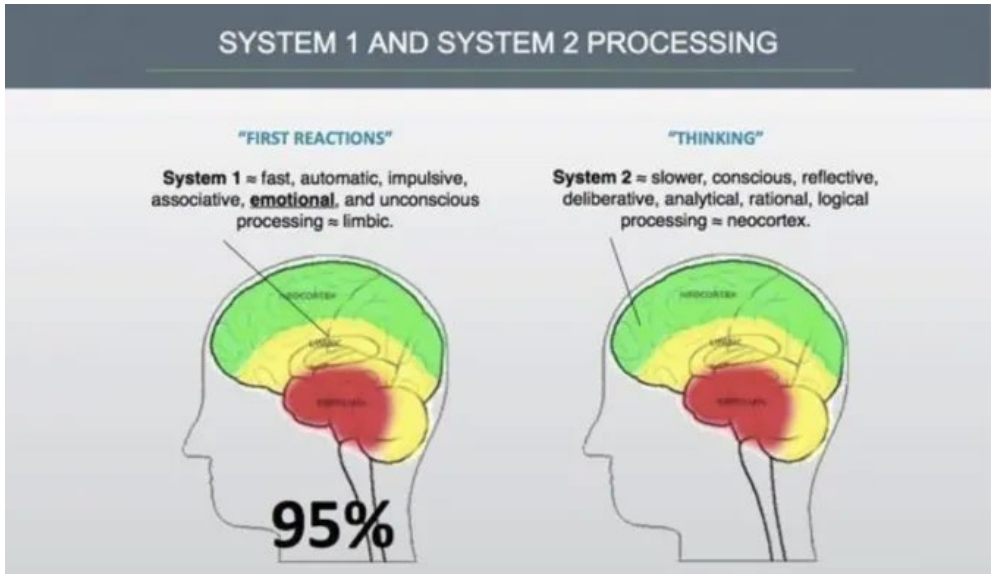
Another Dimension of Modularity (Inspired by Cognition)





Another Dimension of Modularity (Inspired by Cognition)





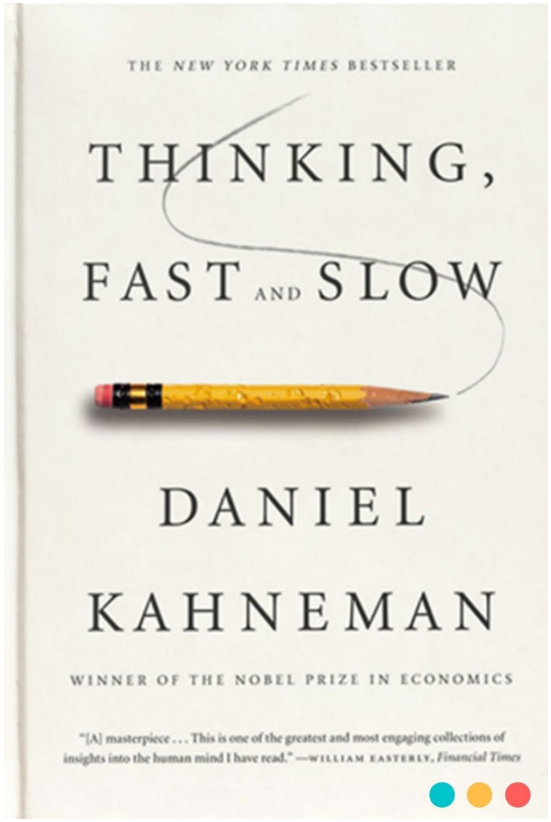
External Stimuli → Perception, Interpretation, Reaction Cache → Other Processing

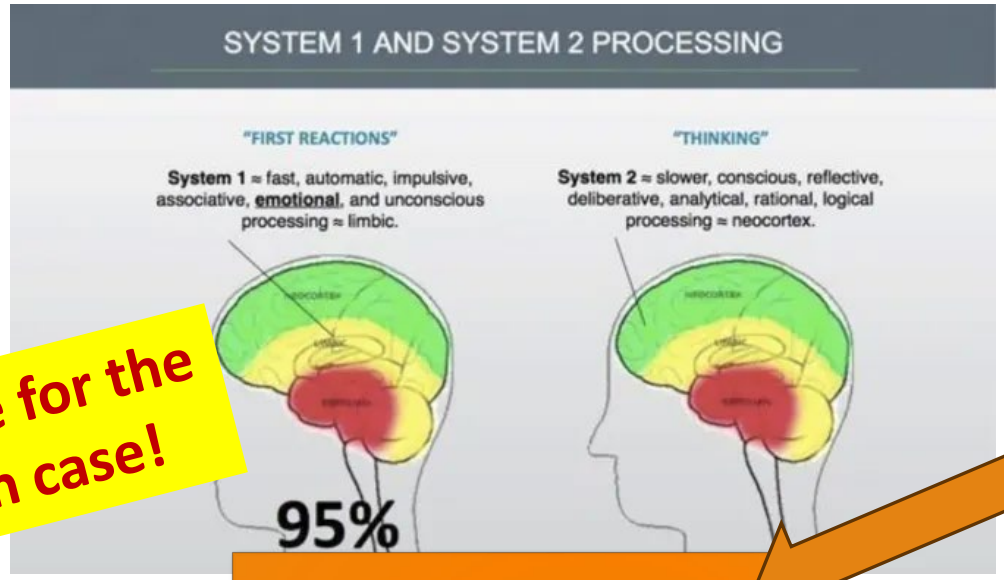
95%

Abstract, conceptual reasoning

Example: OpenAI GPT-5 — explicit fast/reasoning routing

Another Dimension of Modularity (Inspired by Cognition)





Optimize for the common case!

Significant Opportunities for Advancing *Efficient* Pervasive, Ubiquitous, and CPS Applications! ***Micro Foundation Models*** **(μFMs)**

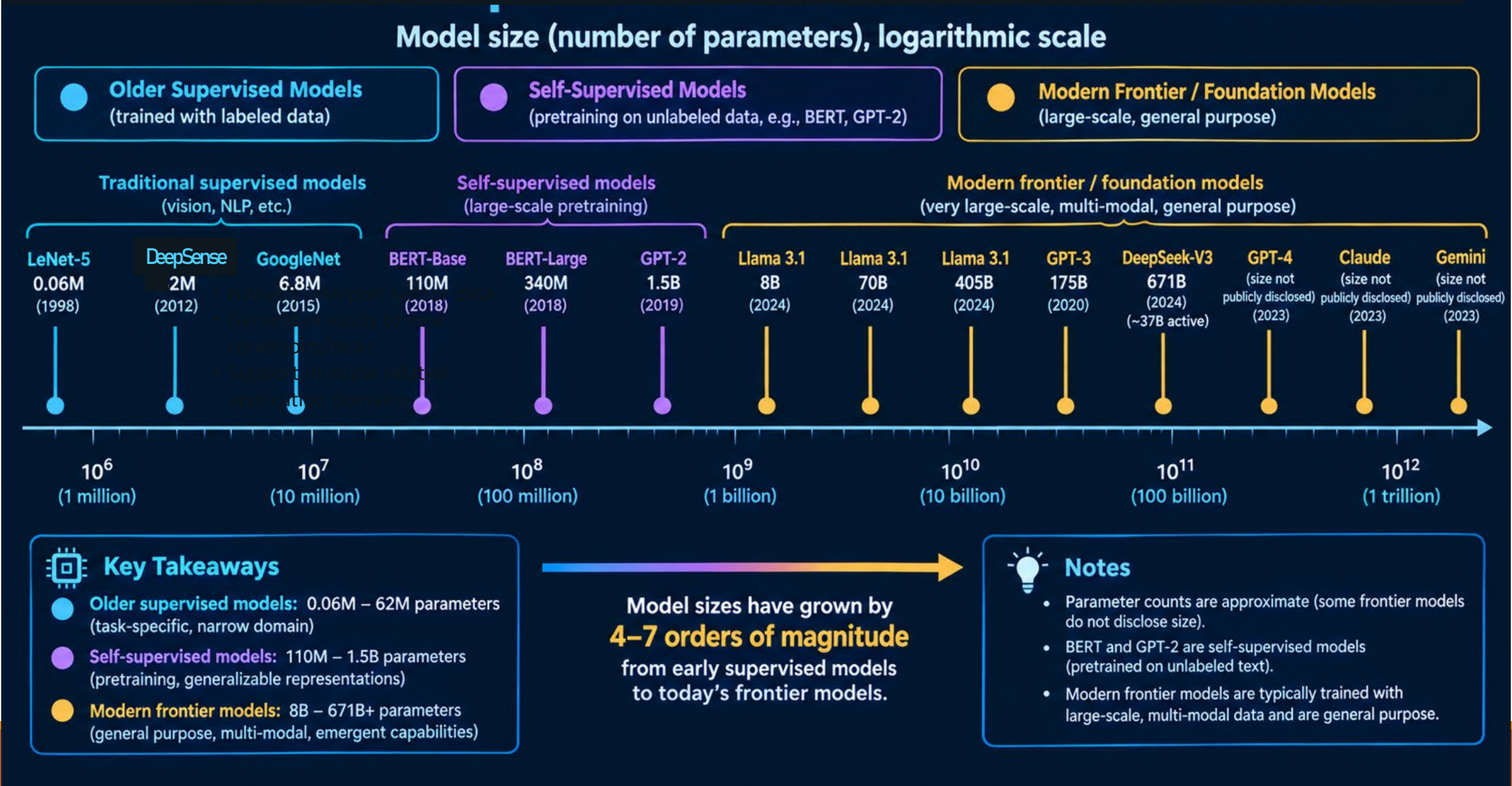
External Stimuli → Perception, Interpretation, Reaction Cache → Other Processing

95%

Abstract, conceptual reasoning

Example: OpenAI GPT-5 — explicit fast/reasoning routing

System-1 Micro-Foundation Models : Optimizing for the Common Case in Tactical AI



Key Takeaways

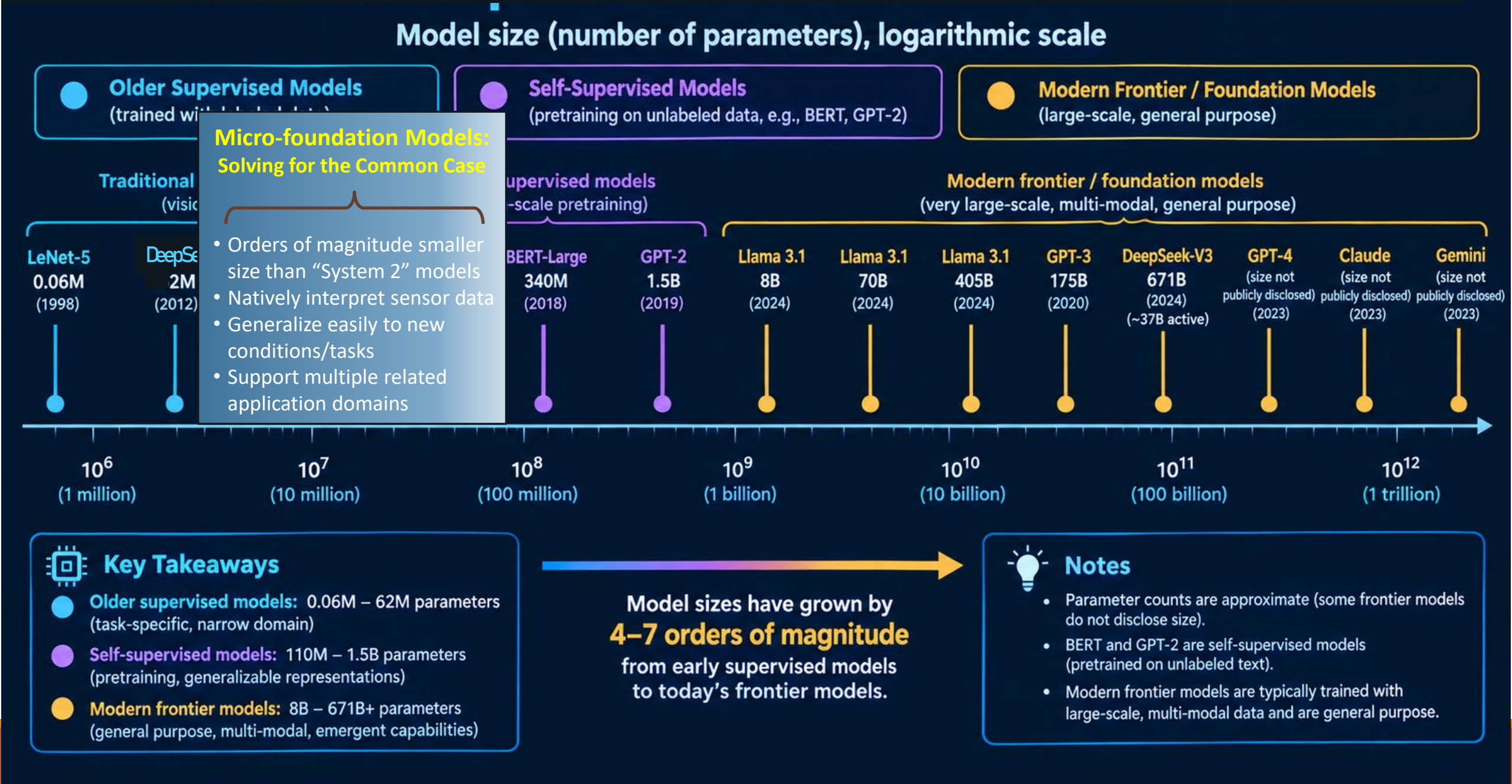
- **Older supervised models:** 0.06M – 62M parameters (task-specific, narrow domain)
- **Self-supervised models:** 110M – 1.5B parameters (pretraining, generalizable representations)
- **Modern frontier models:** 8B – 671B+ parameters (general purpose, multi-modal, emergent capabilities)

Model sizes have grown by **4–7 orders of magnitude** from early supervised models to today's frontier models.

Notes

- Parameter counts are approximate (some frontier models do not disclose size).
- BERT and GPT-2 are self-supervised models (pretrained on unlabeled text).
- Modern frontier models are typically trained with large-scale, multi-modal data and are general purpose.

System-1 Micro-Foundation Models : Optimizing for the Common Case in Tactical AI



Key Takeaways

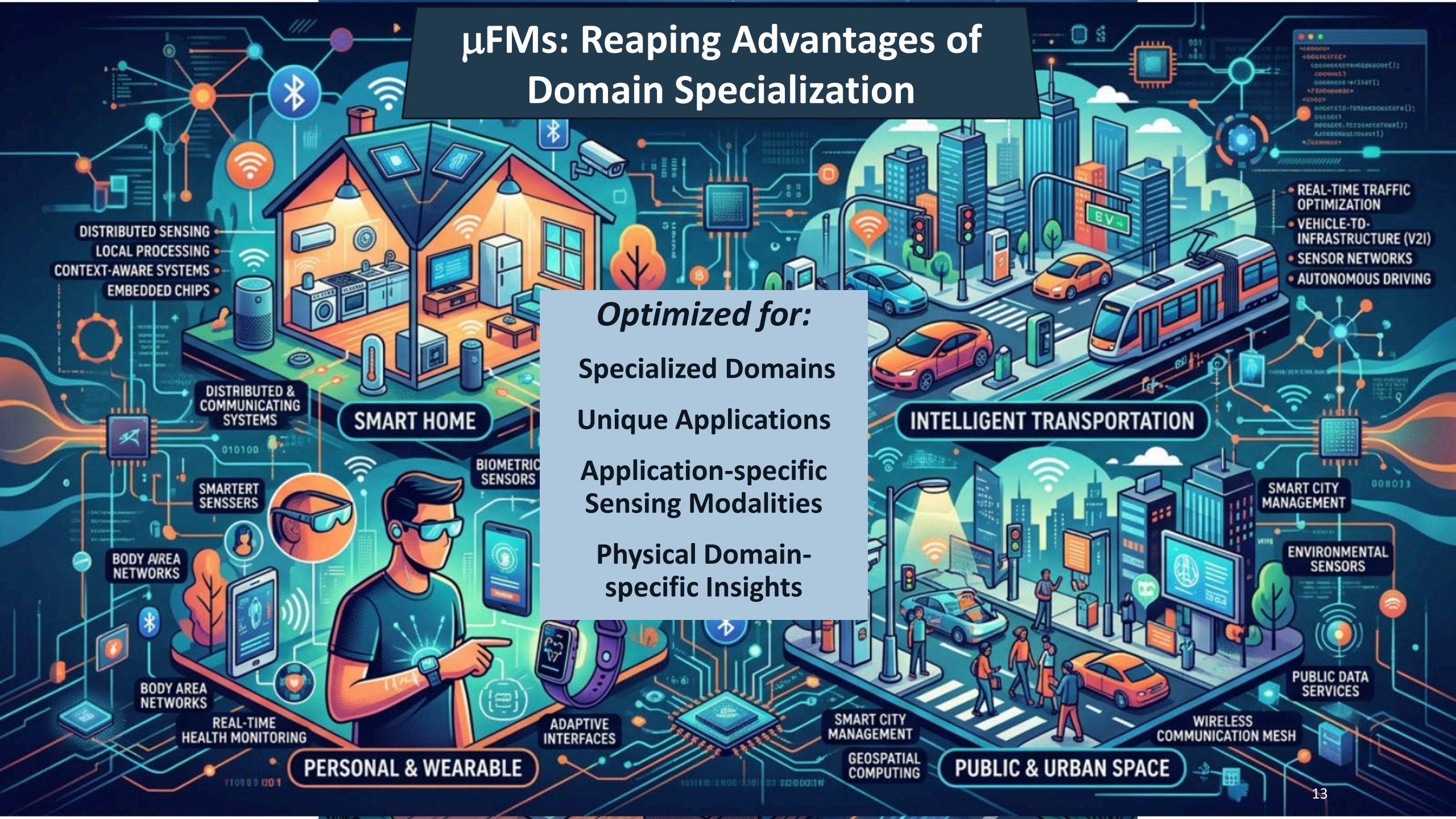
- **Older supervised models:** 0.06M – 62M parameters (task-specific, narrow domain)
- **Self-supervised models:** 110M – 1.5B parameters (pretraining, generalizable representations)
- **Modern frontier models:** 8B – 671B+ parameters (general purpose, multi-modal, emergent capabilities)

Model sizes have grown by **4–7 orders of magnitude** from early supervised models to today's frontier models.

Notes

- Parameter counts are approximate (some frontier models do not disclose size).
- BERT and GPT-2 are self-supervised models (pretrained on unlabeled text).
- Modern frontier models are typically trained with large-scale, multi-modal data and are general purpose.

µFMs: Reaping Advantages of Domain Specialization



Optimized for:

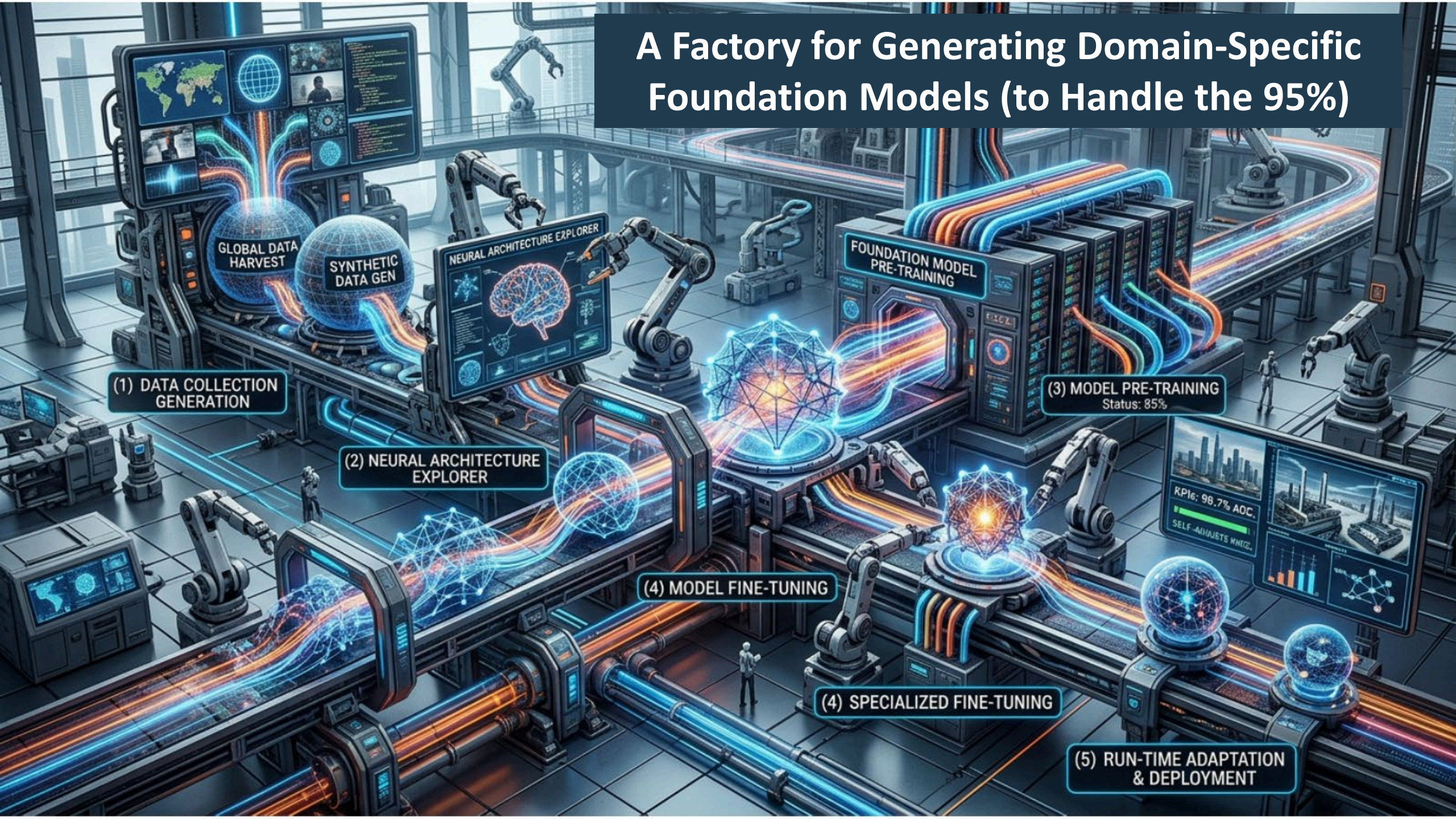
Specialized Domains

Unique Applications

Application-specific Sensing Modalities

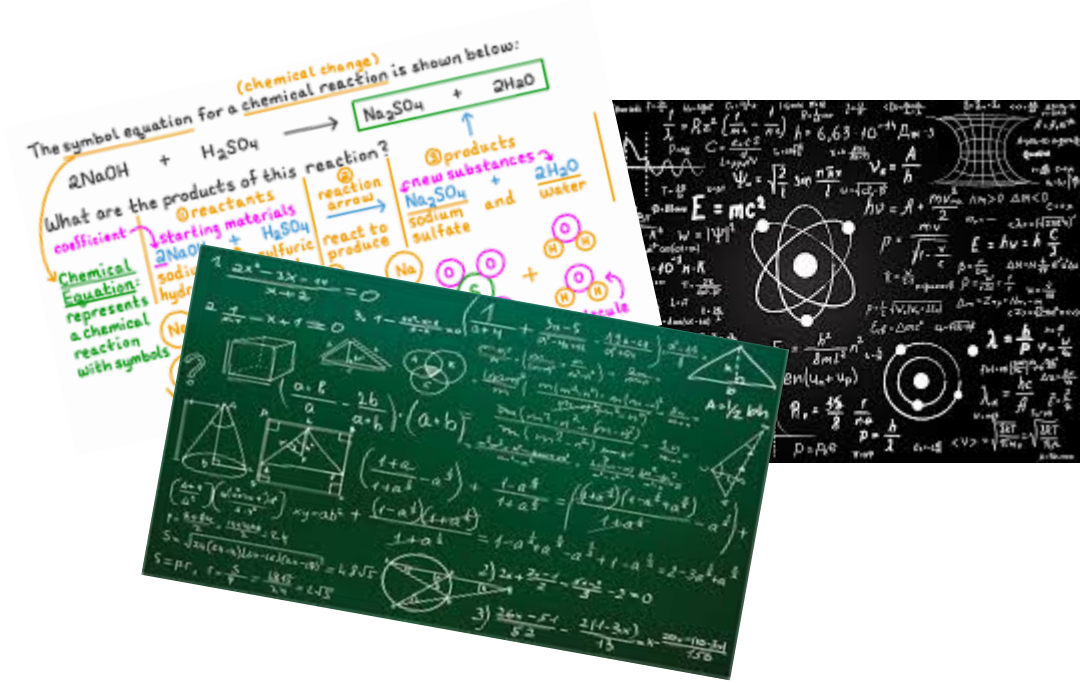
Physical Domain-specific Insights

A Factory for Generating Domain-Specific Foundation Models (to Handle the 95%)



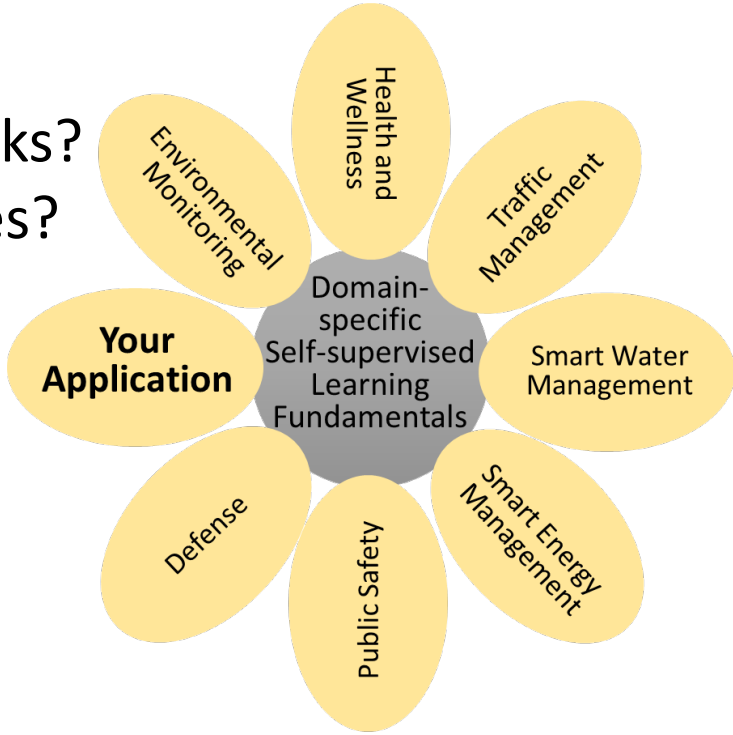
What Guides the Principles of Production of *Domain-Specific* (Micro) Foundation Models?

(… for pervasive, ubiquitous and CPS/IoT applications)



Conventional (scientific) foundational knowledge

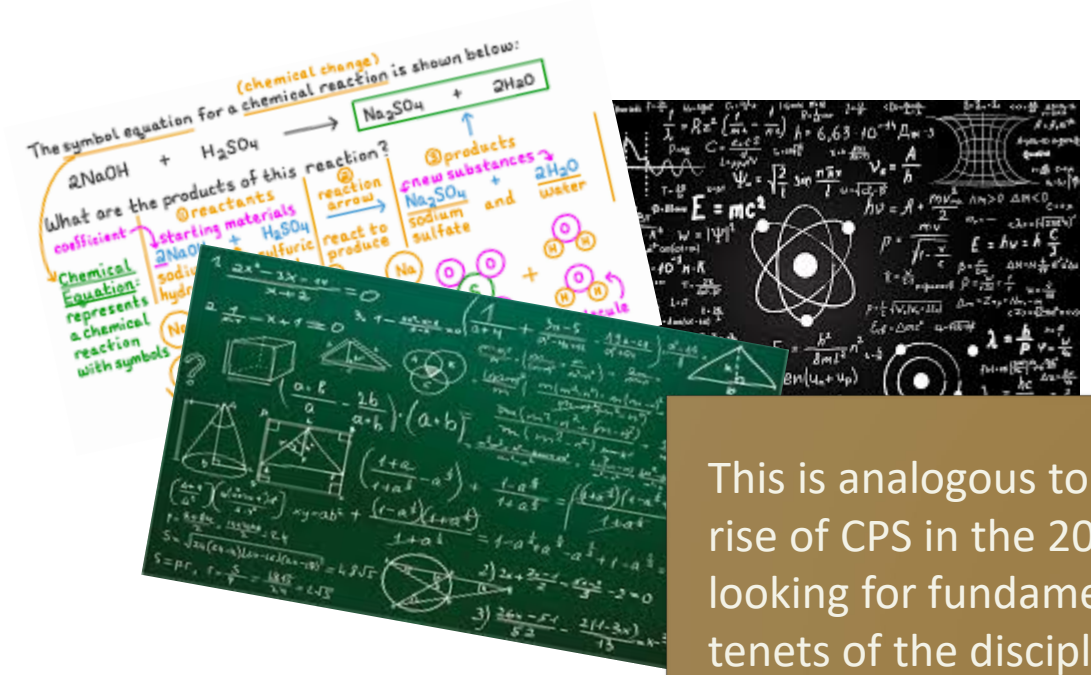
Bottlenecks?
Challenges?



What general principles underlie the production of domain-specific foundation models (for CPS/IoT applications)?

What Guides the Principles of Production of *Domain-Specific* (Micro) Foundation Models?

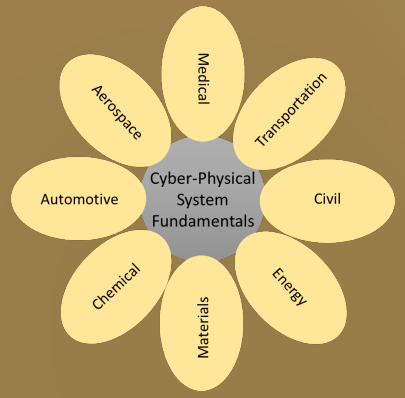
(… for pervasive, ubiquitous and CPS/IoT applications)



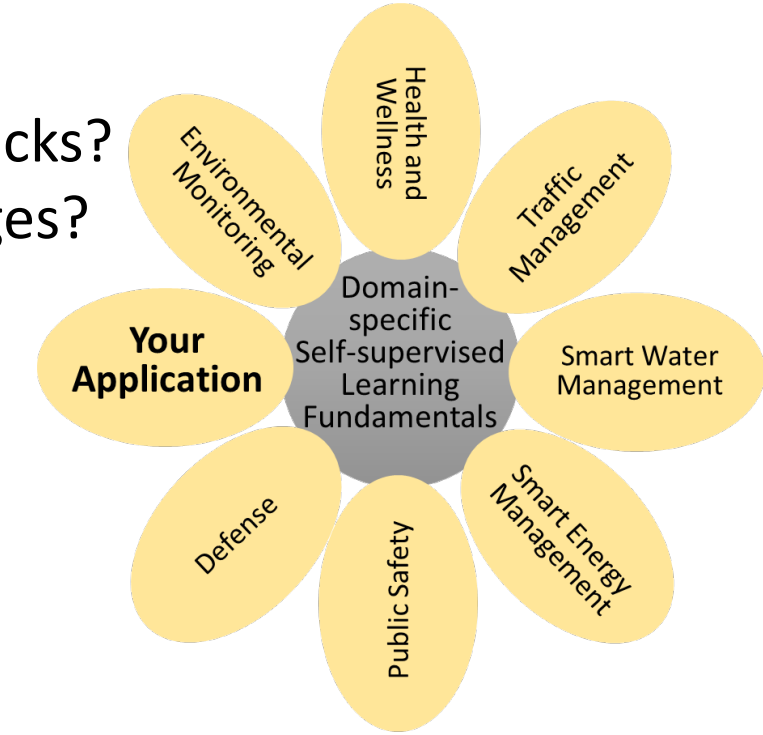
Conventional (scientific) foundational knowledge

This is analogous to the rise of CPS in the 2000s, looking for fundamental tenets of the discipline.

(flower by Jeannette Wing)



Bottlenecks?
Challenges?



What general principles underlie the production of domain-specific foundation models (for CPS/IoT applications)?

Guiding Challenge: The Resource Bottleneck

AI for Embedded Processors



Raspberry Pi Model 4

Power: ~ 10 Watt

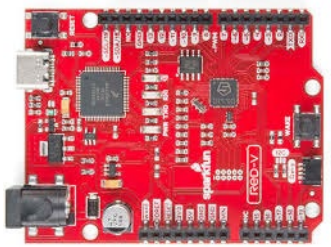
AI for Microcontrollers



Arduino

~ 1 Watt

Hyper-tiny AI



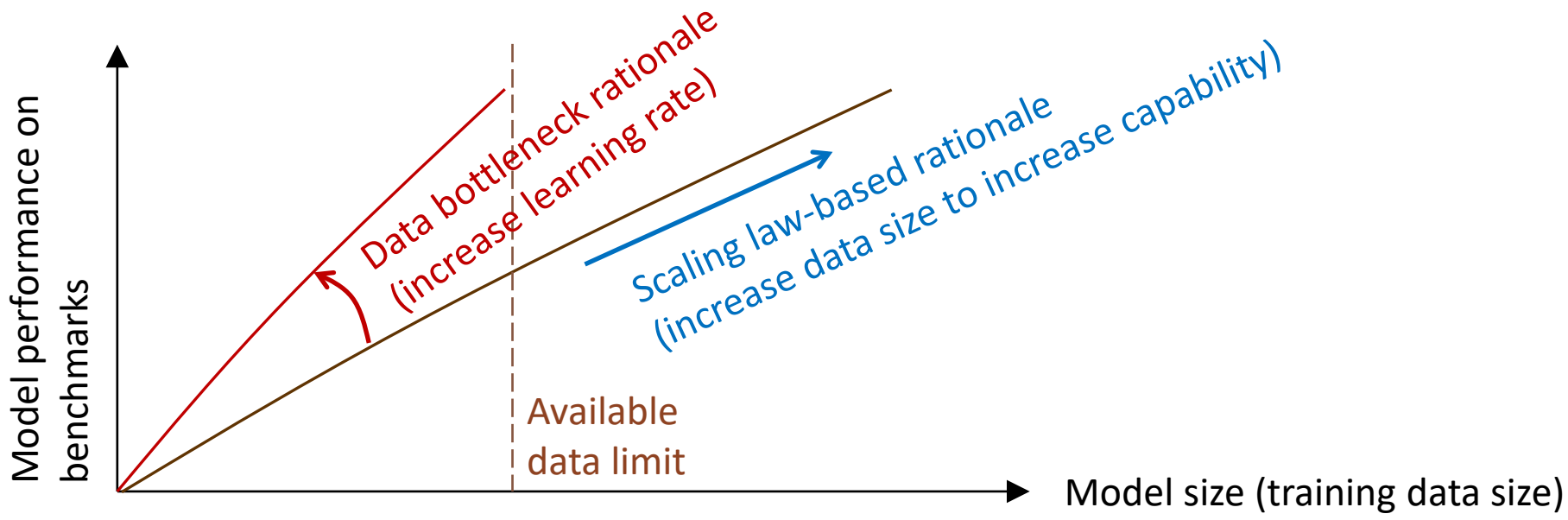
SparkFun RedBoard

~ 0.1 Watt

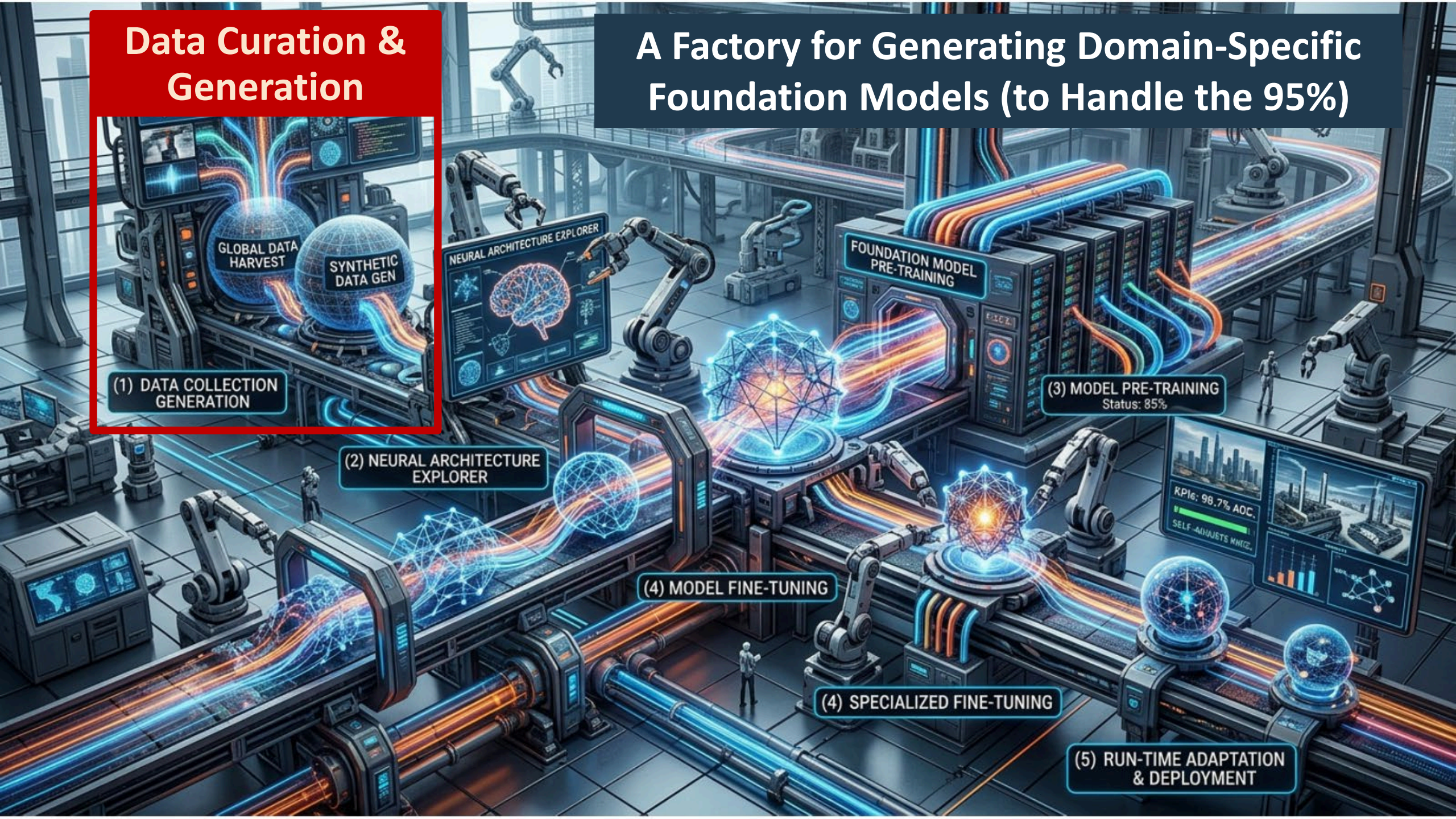
Guiding Challenge: The Data Bottleneck

In domain-specific models, one can't rely on model scaling laws to improve model capabilities (because model size is limited by training data size)

Must re-think the training pipeline to inject the "right amount" of application bias such that the learning rate from limited data is accelerated.

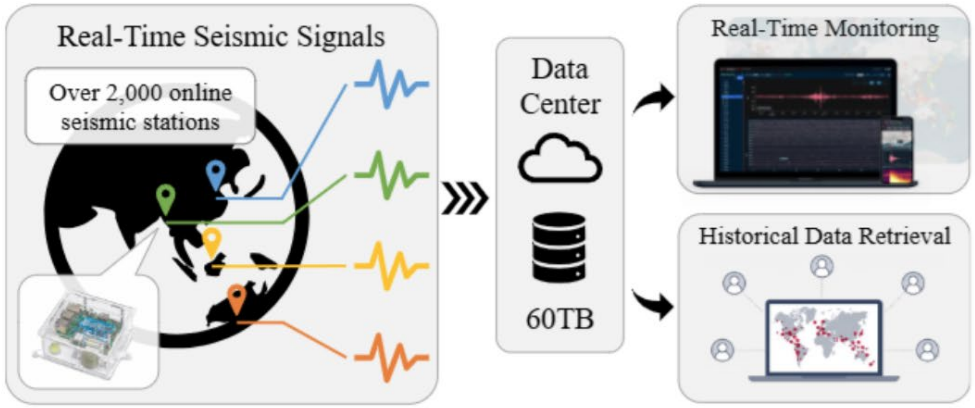


A Factory for Generating Domain-Specific Foundation Models (to Handle the 95%)

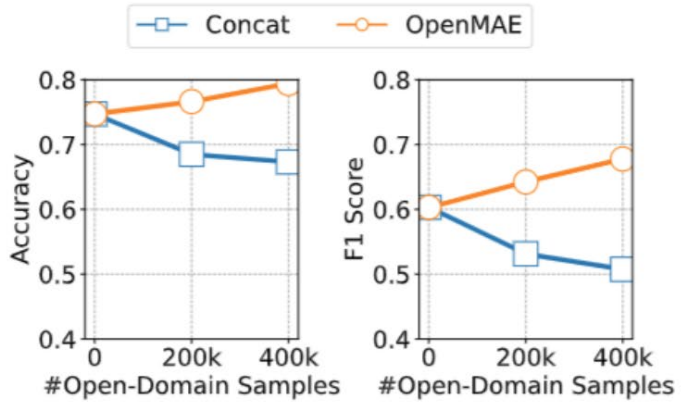


Exploitation of Unvetted Open Domain Data

Can unvetted measurements of a relevant modality (e.g., vibration sensing) be used to enhance foundation model training?



Raspberry Shake data center (SDC). Image courtesy of Raspberry Shake website. It contains over 60TB of seismic data collected from thousands of stations across all seven continents, being the largest community-powered seismic network in the world. Users can monitor real-time signals and download historical data.

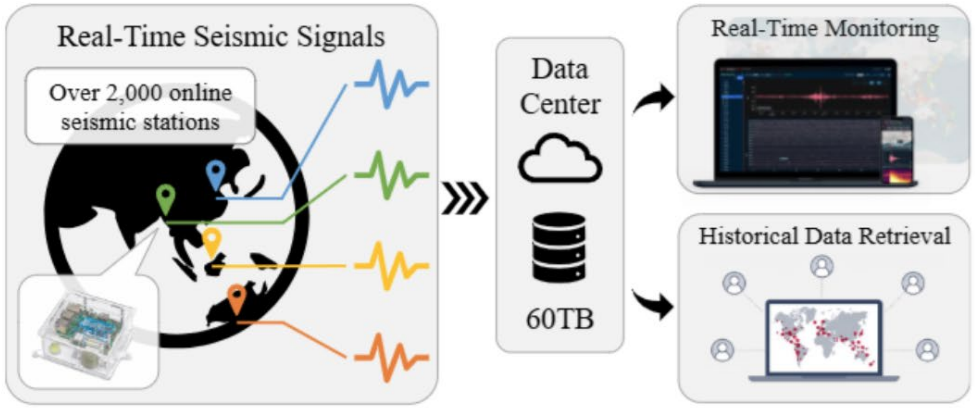


Finetuning performance of models pre-trained on open-domain data. "Concat" means directly combining open-domain and in-domain samples for pretraining. Directly concatenating two types of data leads to performance degradation.

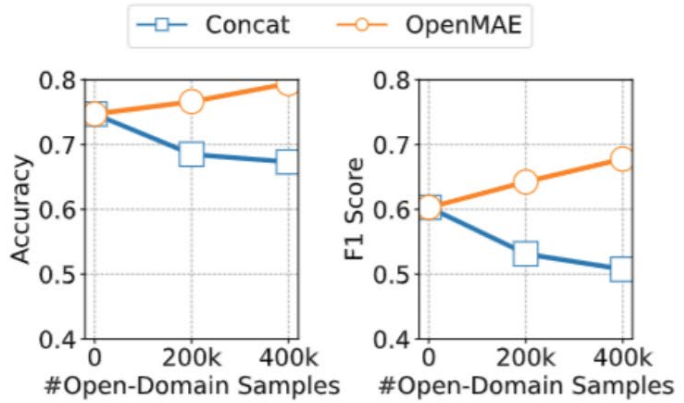
Pre-training a model with unvetted data can degrade downstream performance. Need automated means to vet/augment data!

Exploitation of Unvetted Open Domain Data

Can unvetted measurements of a relevant modality (e.g., vibration sensing) be used to enhance foundation model training?



Raspberry Shake data center (SDC). Image courtesy of Raspberry Shake website. It contains over 60TB of seismic data collected from thousands of stations across all seven continents, being the largest community-powered seismic network in the world. Users can monitor real-time signals and download historical data.



Finetuning performance of models pre-trained on open-domain data. "Concat" means directly combining open-domain and in-domain samples for pretraining. Directly concatenating two types of data leads to performance degradation.

Pre-training a model with unvetted data can degrade downstream performance. Need automated means to vet/augment data!

99% of the sensor data out there might be noise! Garbage in, garbage out!

A Factory for Generating Domain-Specific Foundation Models (to Handle the 95%)



Towards a Data-Efficient Self-Supervised Learning/Inference Pipeline

Part I: The tokenizer

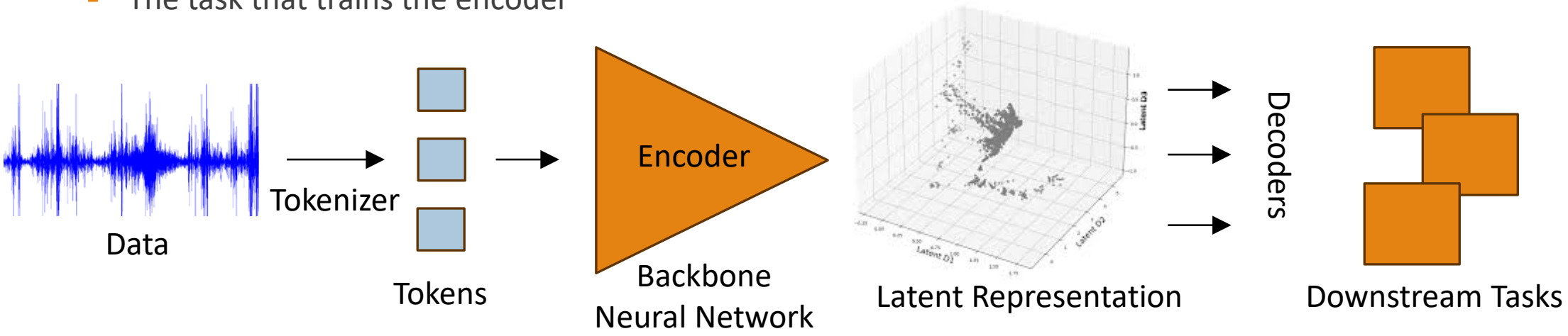
- Breaks the input stream into pieces to be individually encoded.

Part II: The backbone neural network model (or encoder)

- The neural network that converts the stream of input tokens into a latent representation suitable for downstream diagnostics and prediction tasks

Part III: The pretext task(s)

- The task that trains the encoder



Towards a Data-Efficient Self-Supervised Learning/Inference Pipeline

Part I: The tokenizer

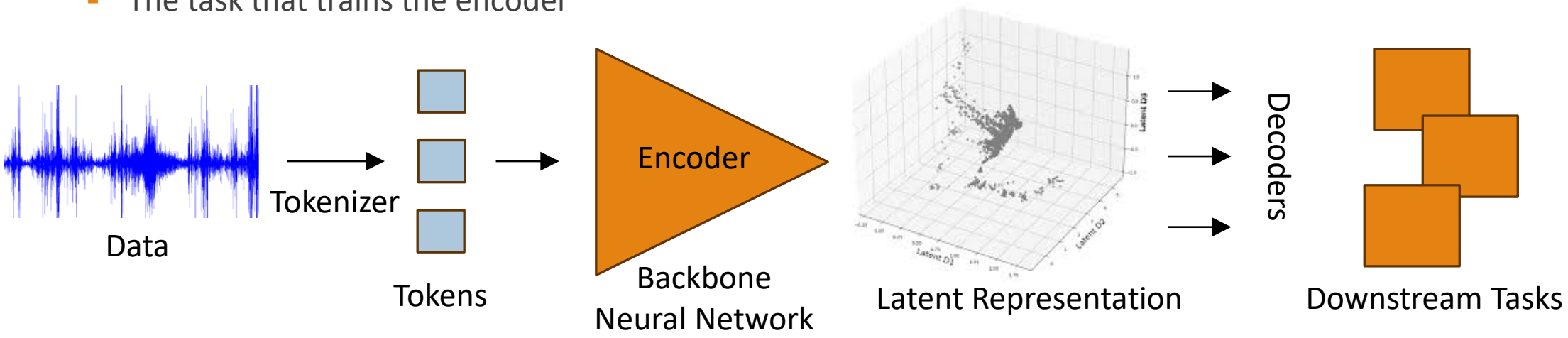
- Breaks the input stream into pieces to be individually encoded.

Part II: The backbone neural network model (or encoder)

- The neural network that converts the stream of input tokens into a latent representation suitable for downstream diagnostics and prediction tasks

Part III: The pretext task(s)

- The task that trains the encoder

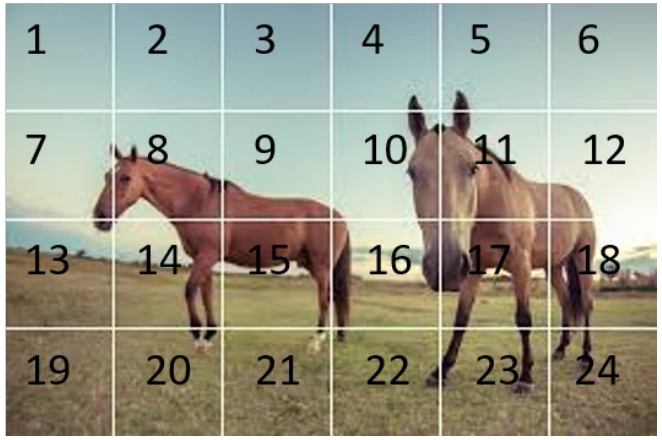


Tokenization

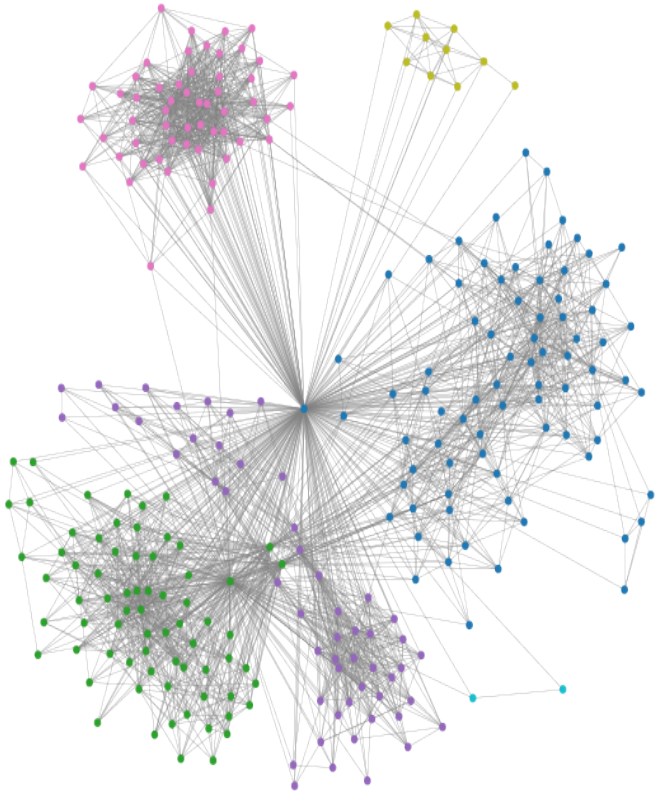
Text: Break sentence into words.

Large Language Models (LLMs), such as GPT-3 and GPT-4, util ize a process called tokenization. Tokenization involves br eaking down text into smaller units, known as tokens, which the model can process and understand. These tokens can rang e from individual characters to entire words or even larger chunks, depending on the model. For GPT-3 and GPT-4, a Byte Pair Encoding (BPE) tokenizer is used. BPE is a subword tok enization technique that allows the model to dynamically bu ild a vocabulary during training, efficiently representing common words and word fragments. Although the core tokeniza tion process remains similar across different versions of t hese models, the specific implementation can vary based on the model's architecture and training objectives.

Images: Break into patches.



Graphs: Break into nodes.

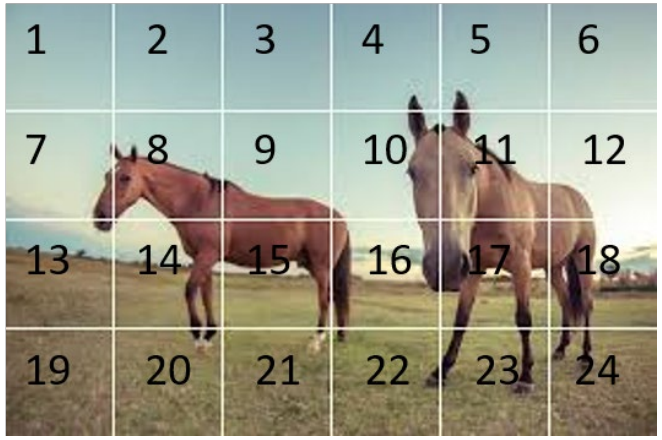


Tokenization

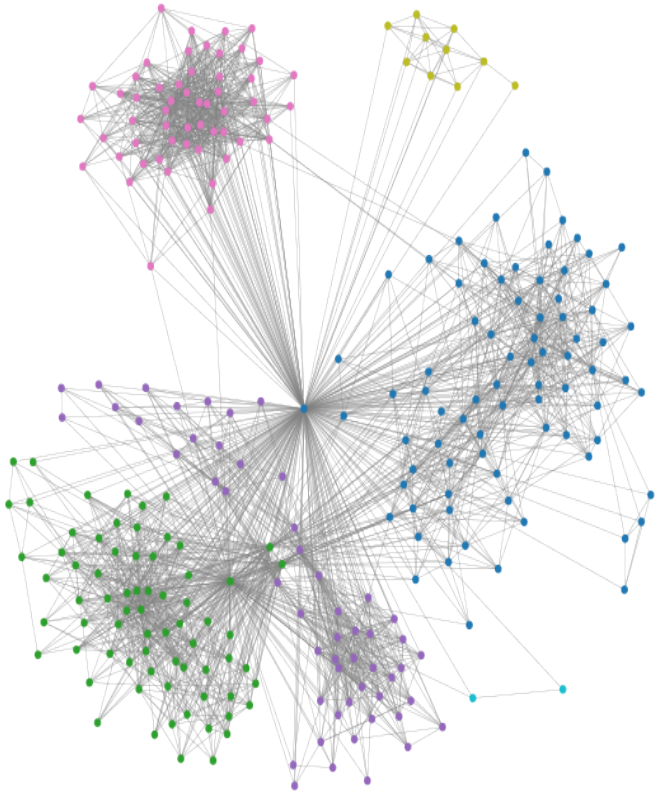
Text: Break sentence into words.

Large Language Models (LLMs), such as GPT-3 and GPT-4, utilize a process called tokenization. Tokenization involves breaking down text into smaller units, known as tokens, which the model can process and understand. These tokens can range from individual characters to entire words or even larger chunks, depending on the model. For GPT-3 and GPT-4, a Byte Pair Encoding (BPE) tokenizer is used. BPE is a subword tokenization technique that allows the model to dynamically build a vocabulary during training, efficiently representing common words and word fragments. Although the core tokenization process remains similar across different versions of these models, the specific implementation can vary based on the model's architecture and training objectives.

Images: Break into patches.

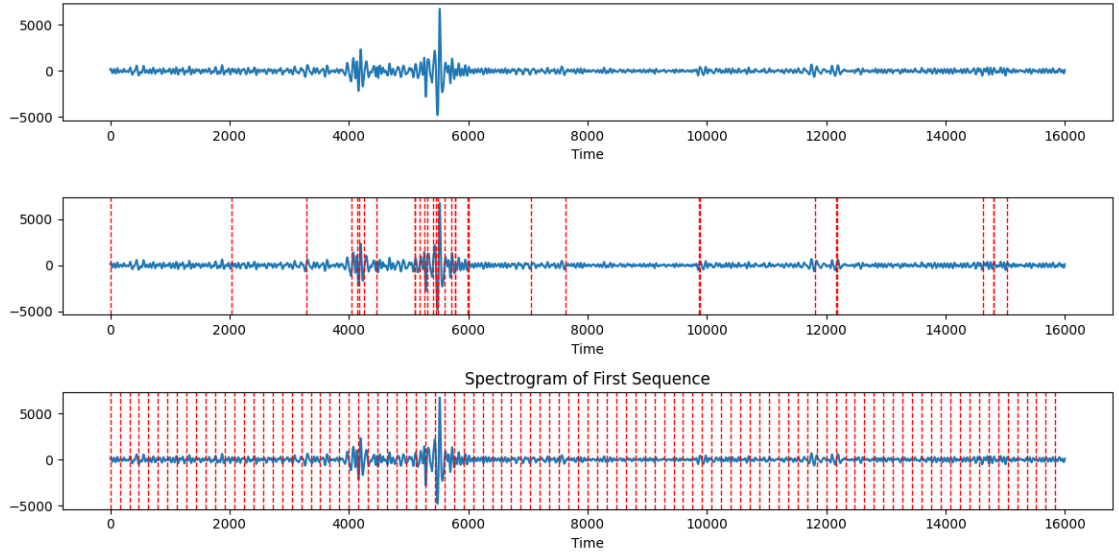


Graphs: Break into nodes.



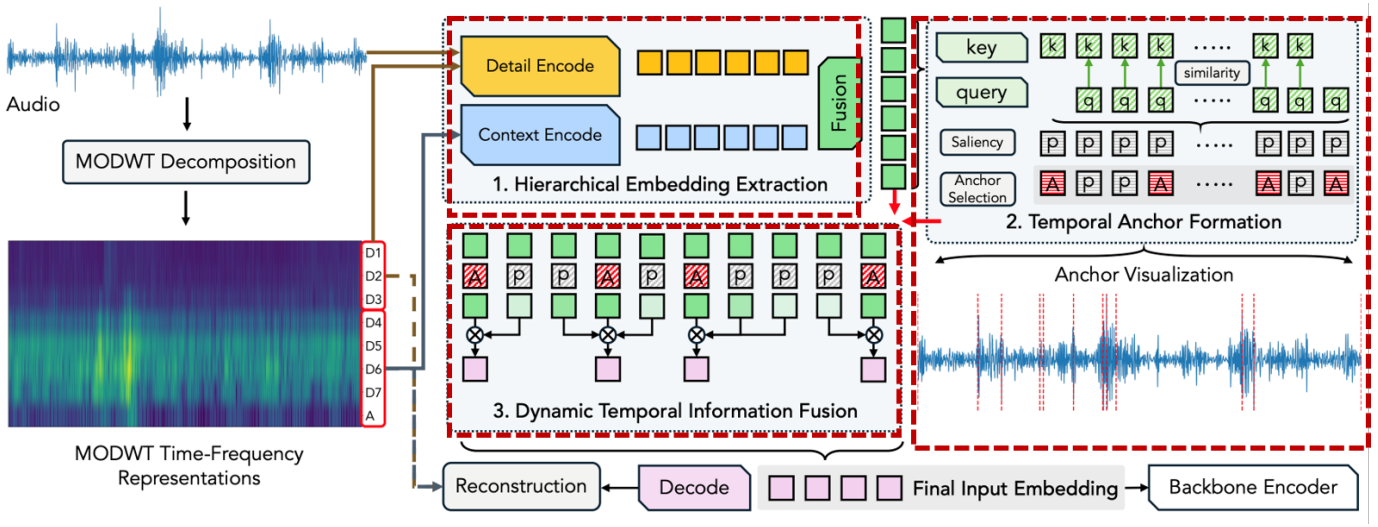
Tokenization for domain-specific time-series data

- **Fixed-length Tokenization:** Each batch of n samples is converted into a token
- **Variable-length Tokenization:** Samples are clustered into groups of different sizes depending on latent information content. Each cluster is converted into a token.



Tokenization

1. Hierarchical Embedding: Capture multi-scale embeddings of physical signals
2. Temporal Anchor Placement: Dynamically identify transition (anchors) between events
3. Dynamic Temporal Information Fusion: Adaptively condense redundant intervals



Evaluation

Achieving better accuracy with fewer tokens

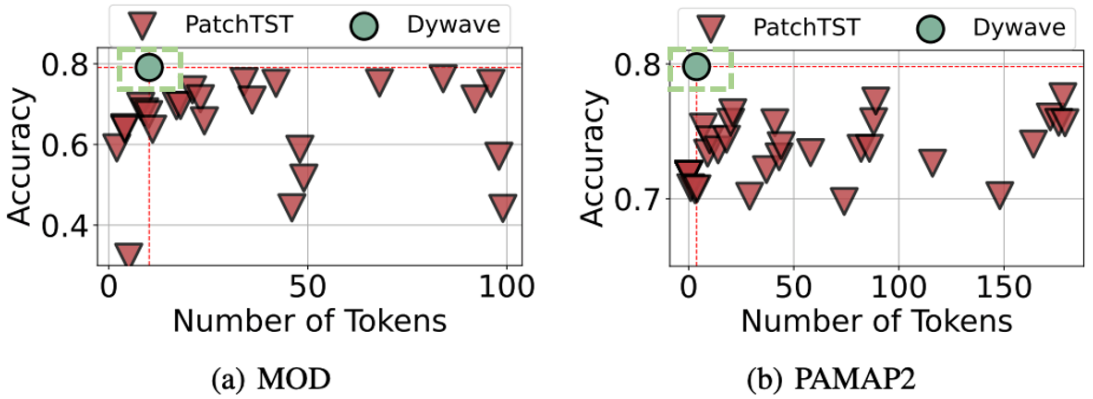


Table 1. Short-context classification performance. Best performance is **bolded** and second best is *underlined*.

Model	Dataset	MOD seismic		PAMAP2 acc		RWHAR acc		PAMAP2 gyro		RWHAR gyro	
		Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
Transformer	PatchTST (Nie et al., 2023)	*0.7743*	*0.7653*	0.7629	0.7549	*0.8840*	*0.8871*	*0.6498*	*0.6343*	*0.6740*	*0.5804*
	MedFormer (Wang et al., 2024)	0.4059	0.3581	*0.7709*	*0.7792*	0.8240	0.8160	0.6317	0.6111	0.5239	0.4308
	DropPatch (Qiu et al., 2025)	0.7582	0.7455	0.6565	0.6432	0.6809	0.6414	0.3876	0.3256	0.4980	0.3764
	WaveToken (Masserano et al., 2025)	0.5358	0.5136	0.4381	0.3658	0.2908	0.2128	0.1475	0.0624	0.2591	0.1595
	Multi-Patch (Naghashi et al., 2025)	0.6739	0.6598	0.7512	0.7388	0.8627	0.8357	0.6037	0.5803	0.5609	0.4701
	Dywave (ours)	**0.7917**	**0.7922**	**0.7977**	**0.7905**	**0.9094**	**0.8932**	**0.7118**	**0.6948**	**0.7294**	**0.7407**
Mamba2	PatchTST (Nie et al., 2023)	*0.7696*	*0.7582*	*0.7488*	*0.7445*	*0.7340*	*0.7429*	0.7058	0.6881	*0.7744*	*0.7758*
	MedFormer (Wang et al., 2024)	0.7006	0.6873	0.7019	0.6770	0.7144	0.6267	0.7019	0.6770	0.7236	0.7140
	DropPatch (Qiu et al., 2025)	0.7525	0.7450	0.5110	0.4824	0.4126	0.2915	0.3644	0.3302	0.4126	0.2915
	WaveToken (Masserano et al., 2025)	0.4233	0.3867	0.5919	0.5569	0.3589	0.2889	0.3454	0.2638	0.4951	0.4027
	Multi-Patch (Naghashi et al., 2025)	0.7033	0.6993	0.7307	0.7176	0.7224	0.5944	*0.7307*	*0.7176*	0.7380	0.7259
	Dywave (ours)	**0.8078**	**0.8010**	**0.8072**	**0.7980**	**0.8517**	**0.8225**	**0.7338**	**0.7272**	**0.7761**	**0.7959**

The Self-Supervised Learning/Inference Pipeline

Part I: The tokenizer

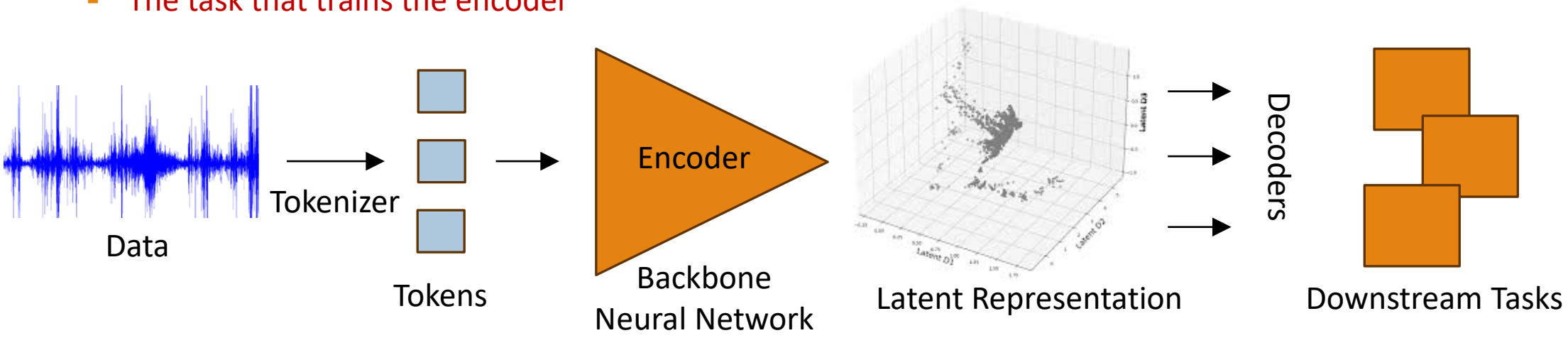
- Breaks the input stream into pieces to be individually encoded.

Part II: The backbone neural network model (or encoder)

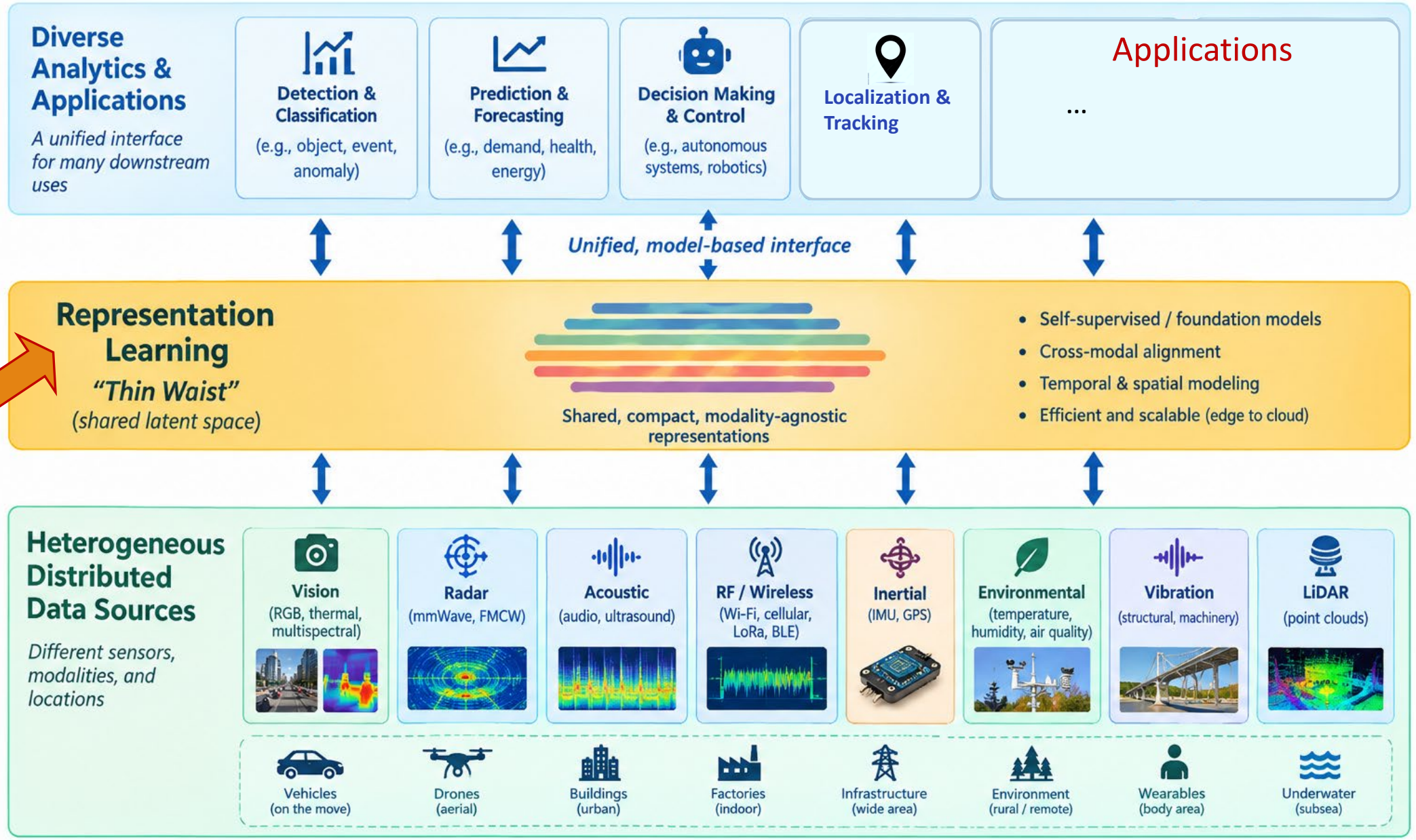
- The neural network that converts the stream of input tokens into a latent representation suitable for downstream diagnostics and prediction tasks

Part III: The pretext task(s)

- The task that trains the encoder



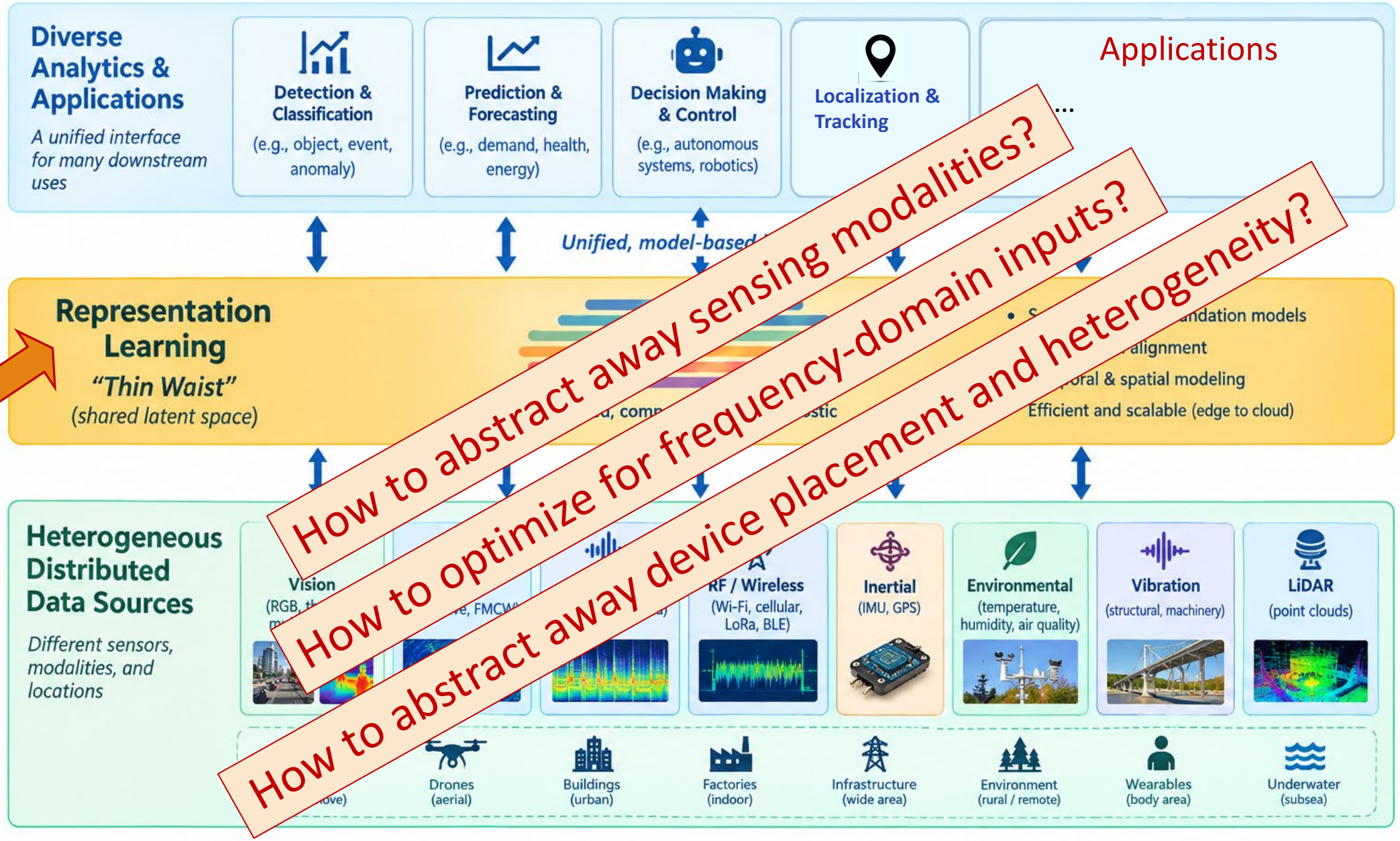
Representation Learning as a “Thin Waist” of a “System 1”-Based Architecture



Thin Waist (Created by new Encoder + Pretext Tasks)

Encoder

Representation Learning as a “Thin Waist” of a “System 1”-Based Architecture



Thin Waist (Created by new Encoder + Pretext Tasks)

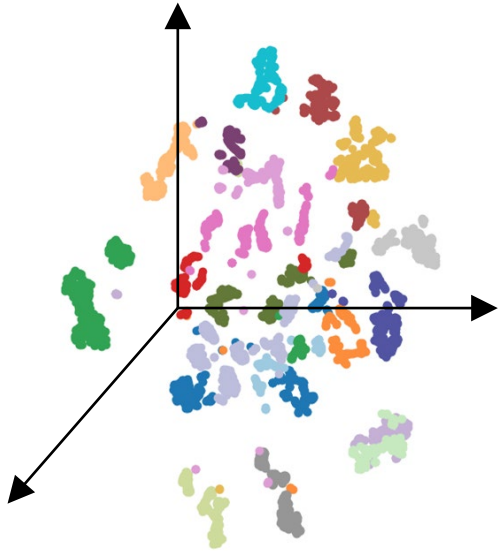
Encoder

VibroFM: A Miniature "Vibrometry" Foundation Model (Using *Multimodal* Contrastive Learning)

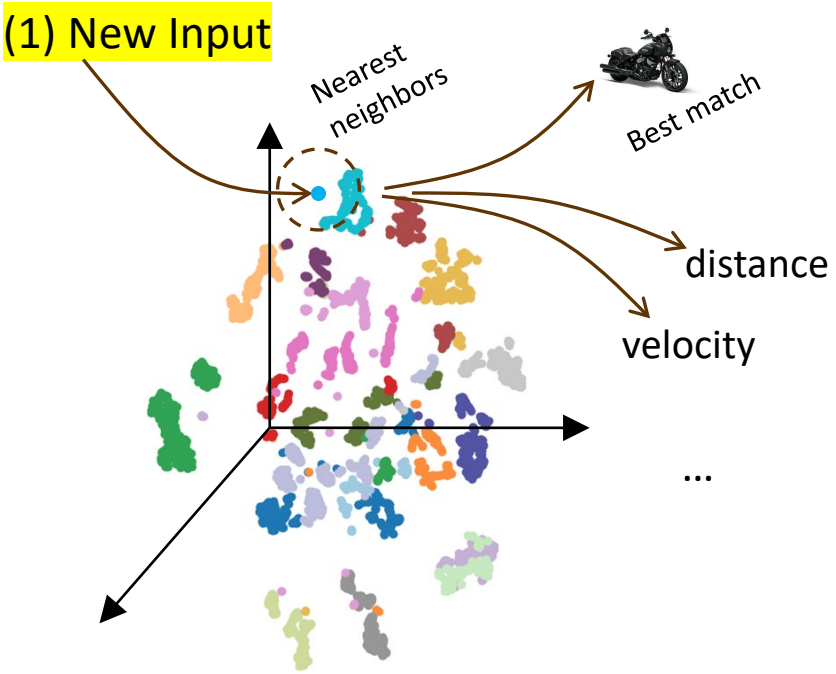


Self-supervised Pretraining

?



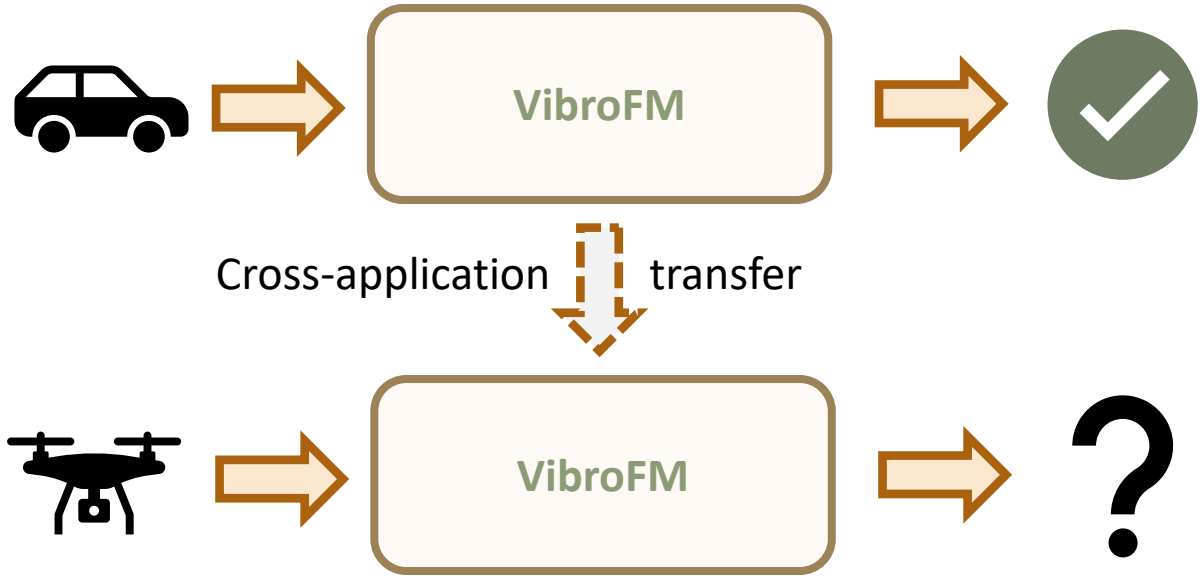
Semantic encoding in a sufficiently high-dimensional space



Fine-tuning: insert a few labeled points to map space regions to labels

Representation Generalizability

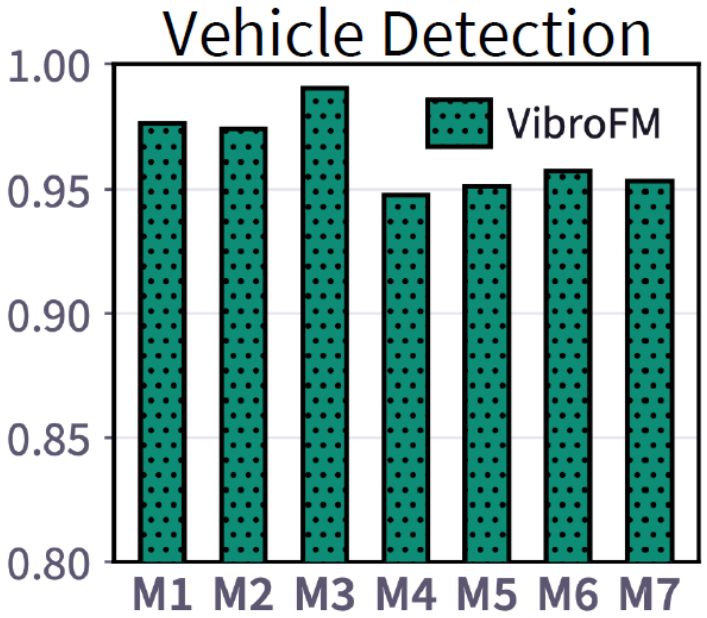
Will representation trained for one application (e.g., vehicular transportation) be useful for another (e.g., drone detection)?

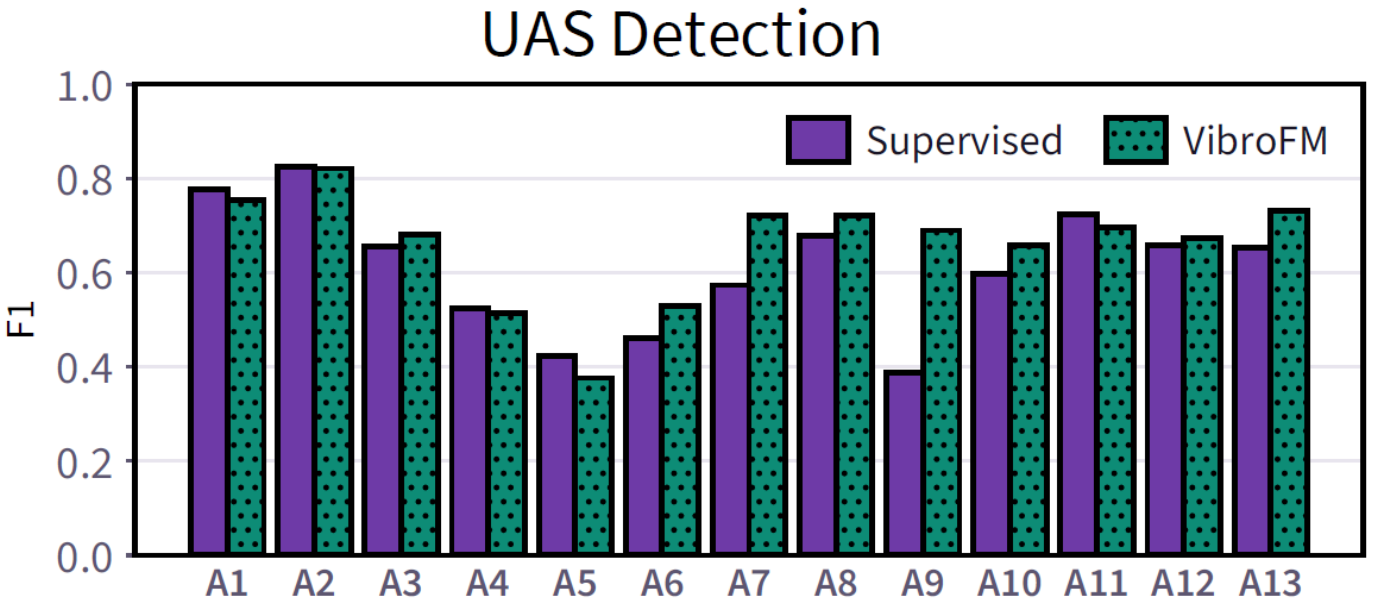


Representation Generalizability

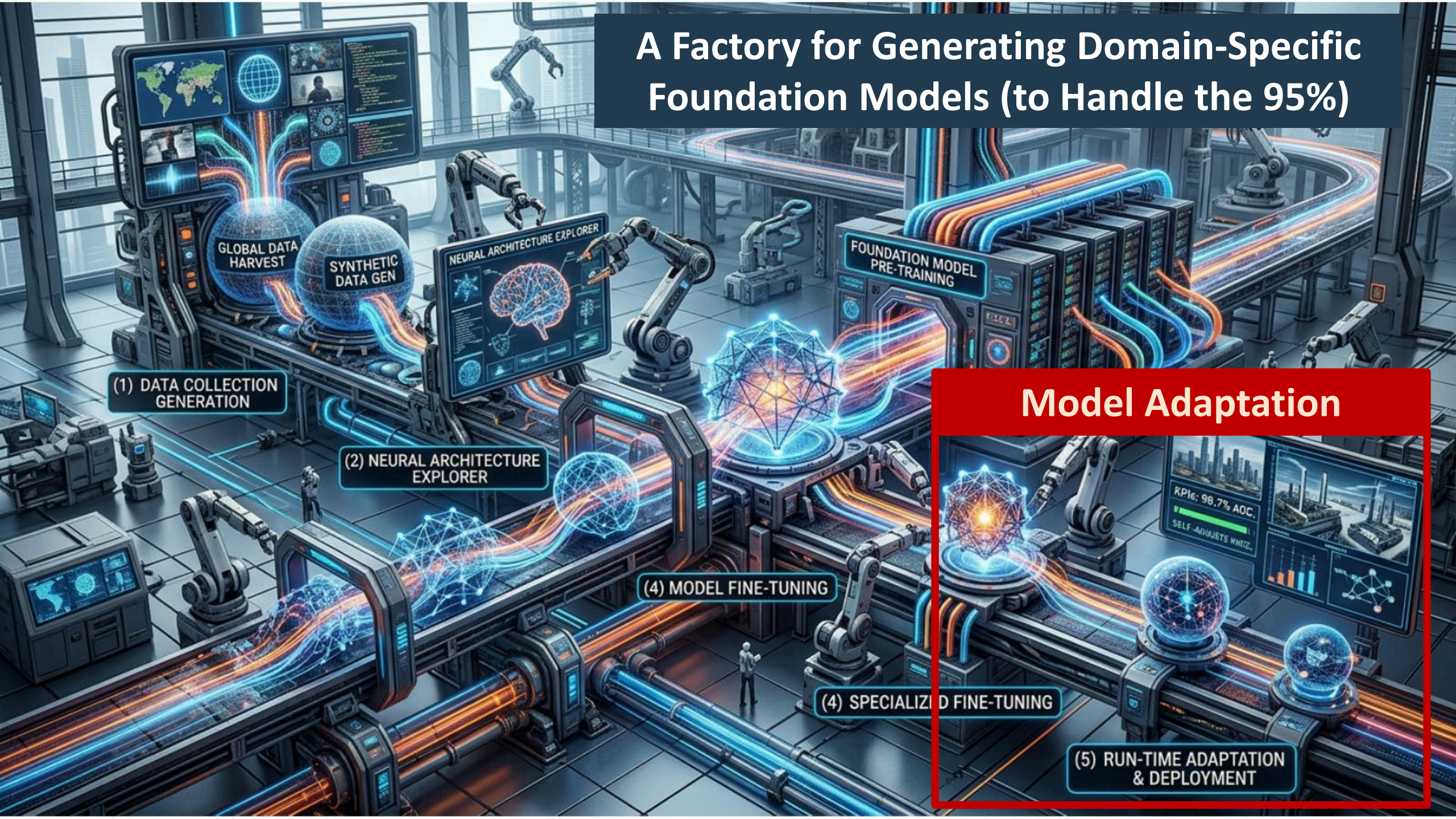
Observation:

- The same VibroFM encoder (pre-training on vehicle data) was used to encode UAV data.
- A classification head was added to classify drones (UAS)
- Drone classification based on the VibroFM representation outperformed (in many cases) a drone classifier designed from scratch in a supervised fashion from labeled drone data!





A Factory for Generating Domain-Specific Foundation Models (to Handle the 95%)



Adapting the Model to Target Hardware, Input Quality, and Data Complexity

Differences in Target Hardware

AI for Embedded Processors



Raspberry Pi Model 4

Power: ~ 10 Watt

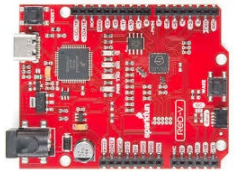
AI for Microcontrollers



Arduino

~ 1 Watt

Hyper-tiny AI



SparkFun RedBoard

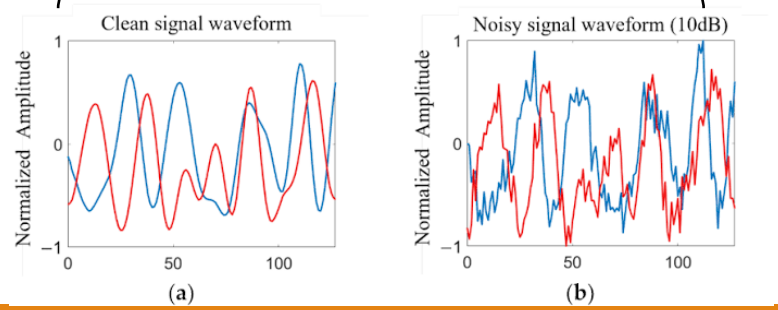
~ 0.1 Watt





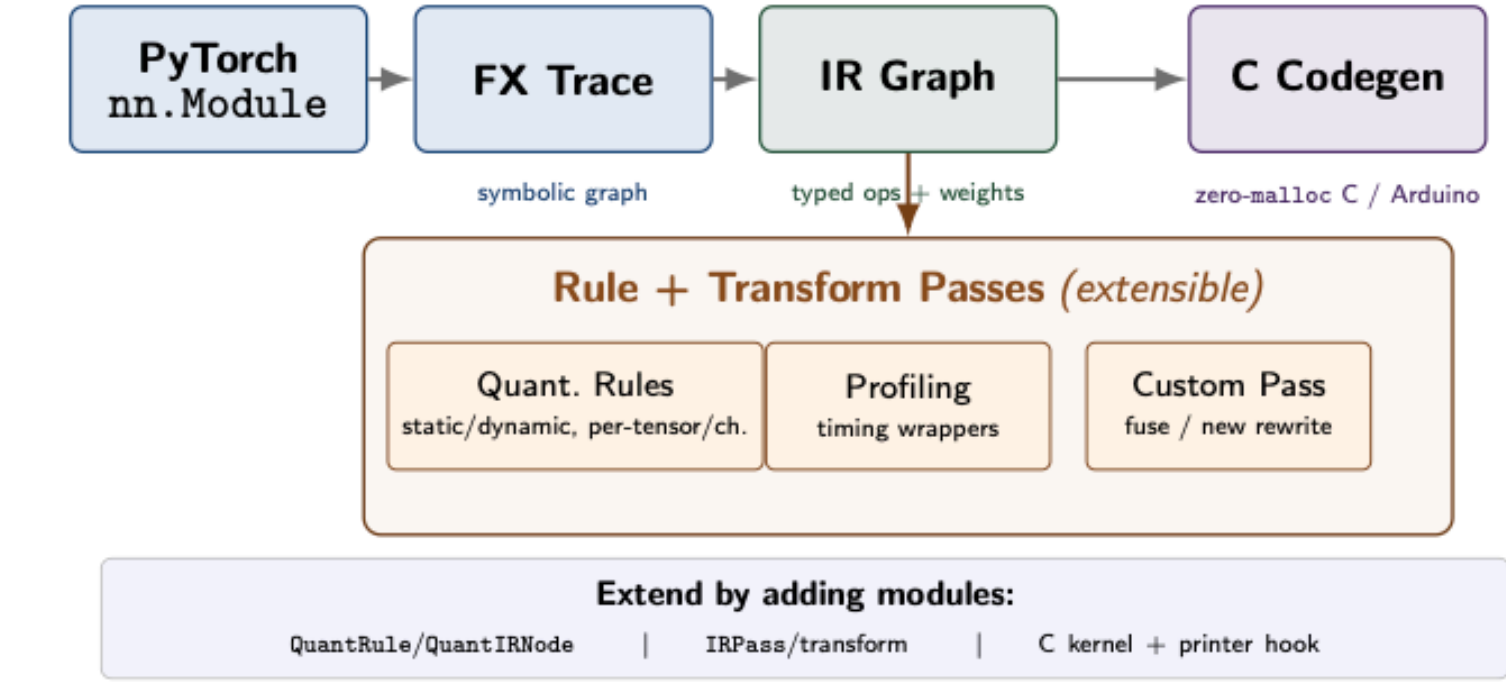
Differences in Data Complexity

Differences in Input Quality



A Toolchain For Adaptive Model Quantization

General Compiler for AI Models to produce Adaptively Quantized Versions



A new quantization scheme = a rule that matches layers and a node that owns its C emission.

Summary

Key takeaways:

- The CPS community has a great role to play in adapting the state of the art in training and inference to the needs of physical/embodied AI applications
- One may need to rethink the balance between “system 2” and “system 1” solutions for the physical/embodied AI application space, optimizing for the 95%!
- The idea of a factory for “System 1” solutions (for physical AI) deserves more exploration, complementing the popular “System 2” models (LLMs/VLMs/VLAs) to attain significant savings!