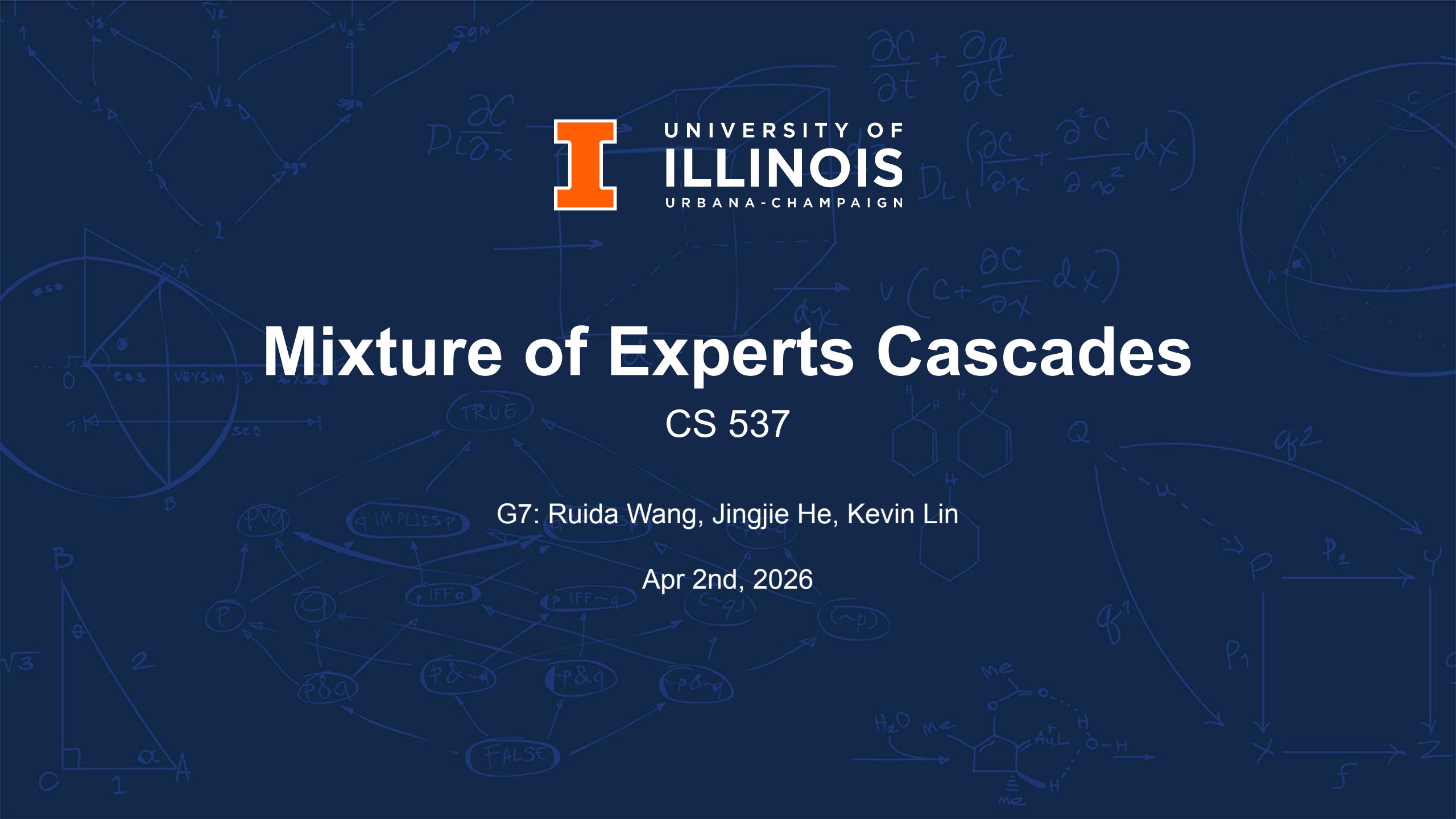


UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Mixture of Experts Cascades

CS 537

G7: Ruida Wang, Jingjie He, Kevin Lin

Apr 2nd, 2026

Overview

Part 1: MoE in Mainstream AI

- Introduction to MoE
  - MoE in LLMs (DeepSeek)
- Time and Efficiency
  - Time-MoE: boosting performance while reducing runtime requirements

Part 2: MoE in IoT

- Real-time Schedulability
  - Scheduling Classifiers: latency-performance trade-offs
- IDK Cascades
  - Cascades with classifier dependencies
  - Cascades optimized for time-series input data
  - Class hierarchies in cascades

MoE In Mainstream AI

Introduction to MoE

Introduction to MoE

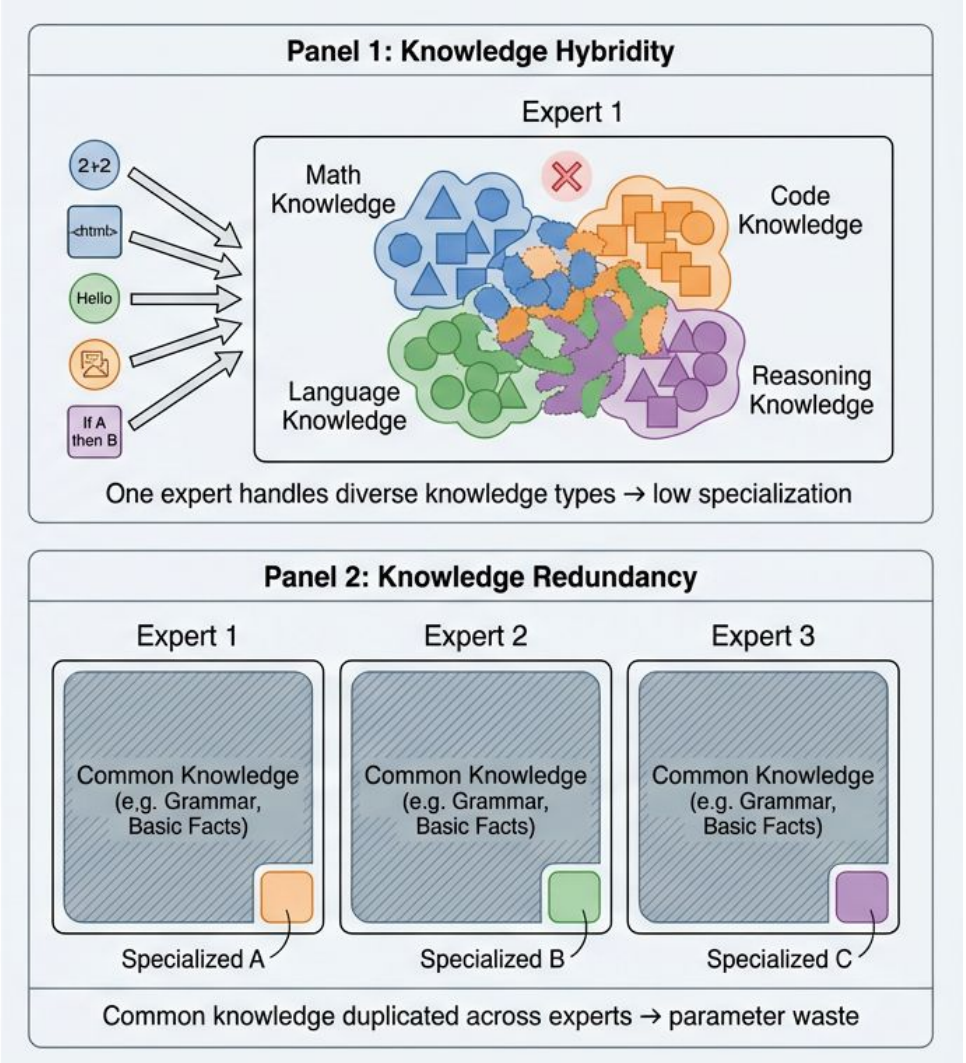
- To understand the Mixture-of-Experts (MoE) architecture, one must first grasp the concept of an "Expert."
- An expert can be formally defined as a specialized machine learning model trained to accomplish a narrowly defined task.
- In the context of Large Language Models (LLMs), an expert constitutes a small module possessing knowledge within a specific domain (e.g., experts in Mathematics, Physics, or Literature).
- Within the Internet of Things (IoT) context, experts can be broadly categorized into two types:
  - Uni-modality expert: A set of models that is trained with different constraints (such as different in size/training data) that operates on a uniform modality of data
  - Multi-modality experts: A set of models that handles different types of data. For example experts in sound, vision, time-series etc. modality of data.
- With the definition of “expert”, we define the “mixture-of-expert” as the methods that develop frameworks to orchestra the experts to have better performance in model capability or to save memory.
- One of the methods to organize the MoE system is the “MoE cascade”

DeepSeekMoE: Towards Ultimate Expert Specialization in Mixture-of-Experts Language Models

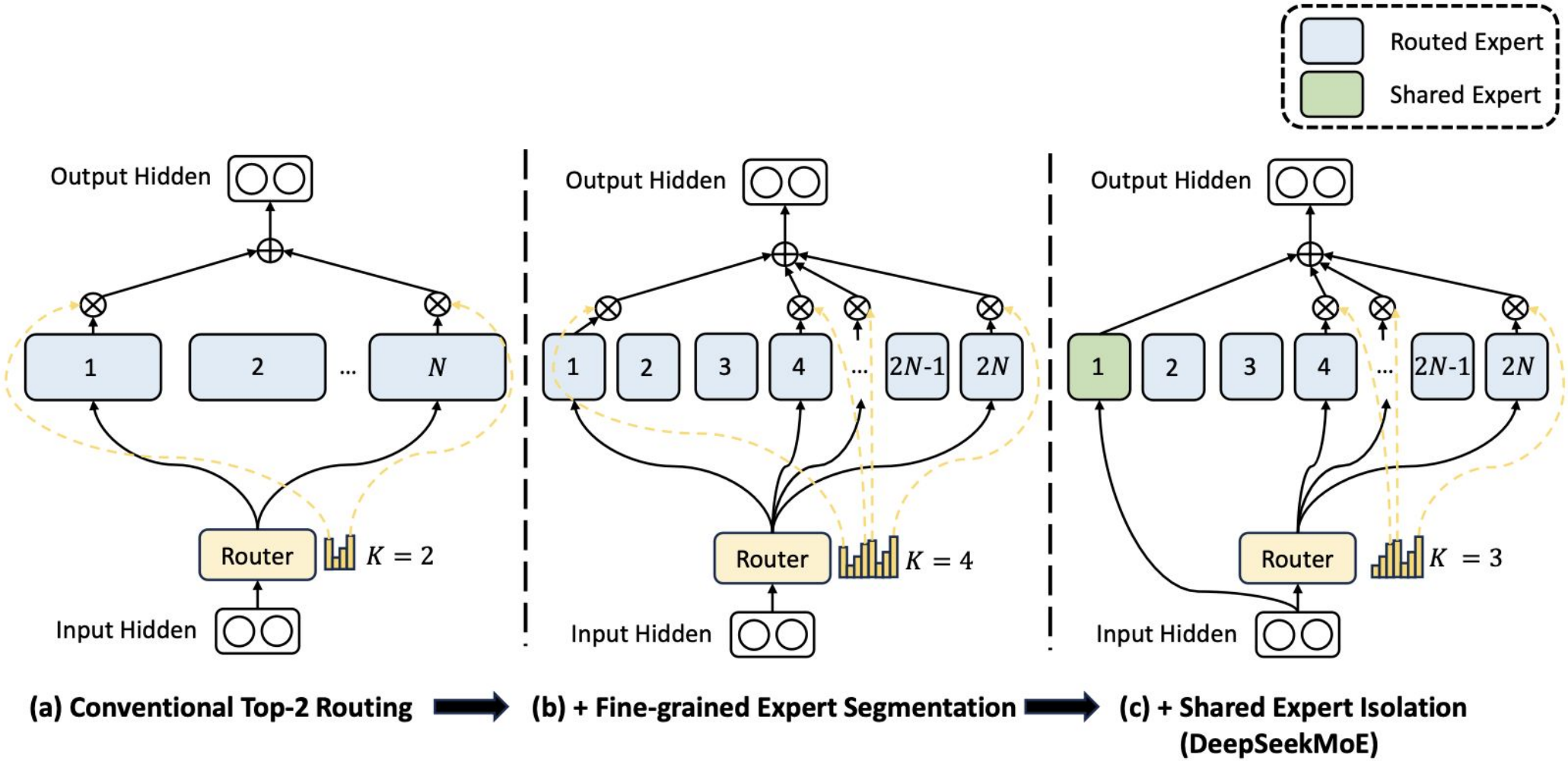
Background and Motivation

The MoE system in the context of LLMs faces the following major challenges:

- **Knowledge Hybridity:** Traditional MoE methods would select 2-4 experts out of 8-16 total experts per token of generation. The small amount of experts may lead to the expert have mixed set of knowledge, making it less specialized
- **Knowledge Redundancy:** The token assigned to different experts may require some common knowledge to solve. However, such common knowledge may span across all experts



Methodology



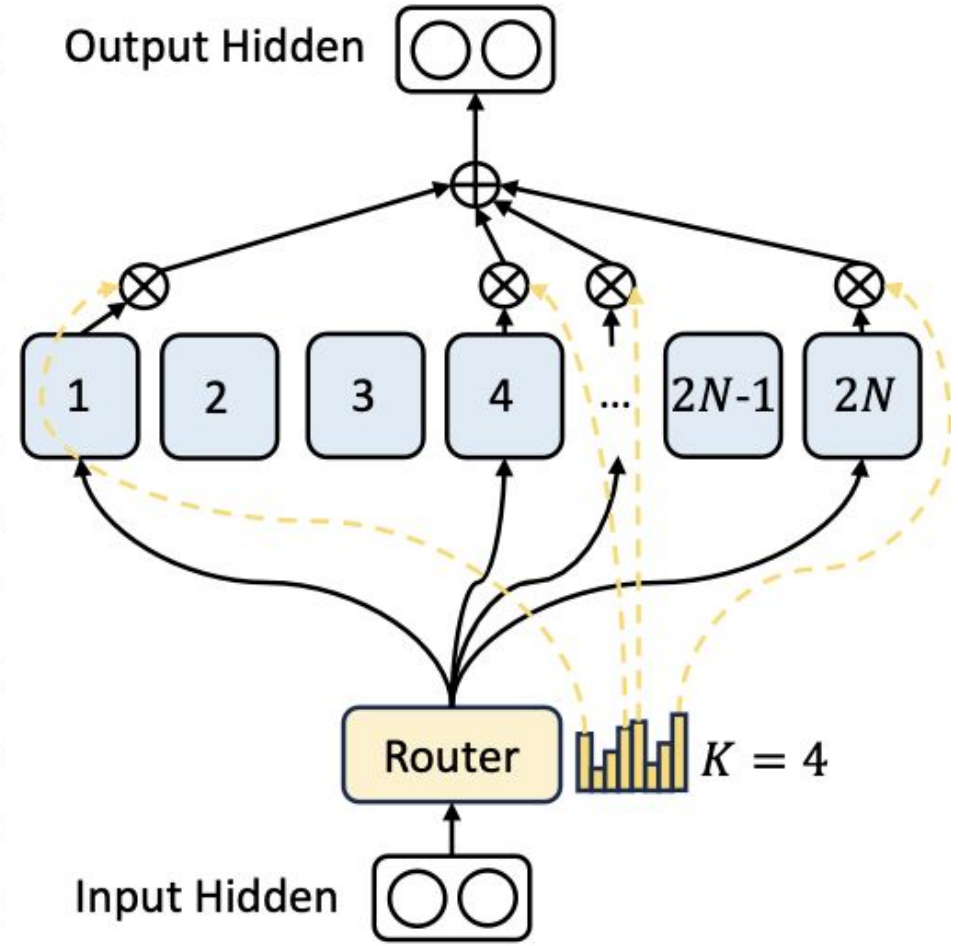
Methodology: Fine-grained Expert Segmentation

- This expert segmentation method divides the limited amount of large experts into many small experts.
- It enables each experts to be more specialized in its certain problem, instead of obtaining many mixing knowledges.
- Split N big experts that may have many knowledge in it into mN small experts that can be specialized in specific fields. When choosing the results. change from choosing K experts into choosing mK experts.
- Formally, the expert system is:

$$\mathbf{h}_t^l = \sum_{i=1}^{mN} \left(g_{i,t} \, \mathrm{FFN}_i \left(\mathbf{u}_t^l \right) \right) + \mathbf{u}_t^l,$$

$$g_{i,t} = \begin{cases} s_{i,t}, & s_{i,t} \in \mathrm{Topk}(\{s_{j,t} | 1 \leqslant j \leqslant mN\}, mK), \\ 0, & \text{otherwise}, \end{cases}$$

$$s_{i,t} = \mathrm{Softmax}_i \left(\mathbf{u}_t^{l^T} \mathbf{e}_i^l \right),$$



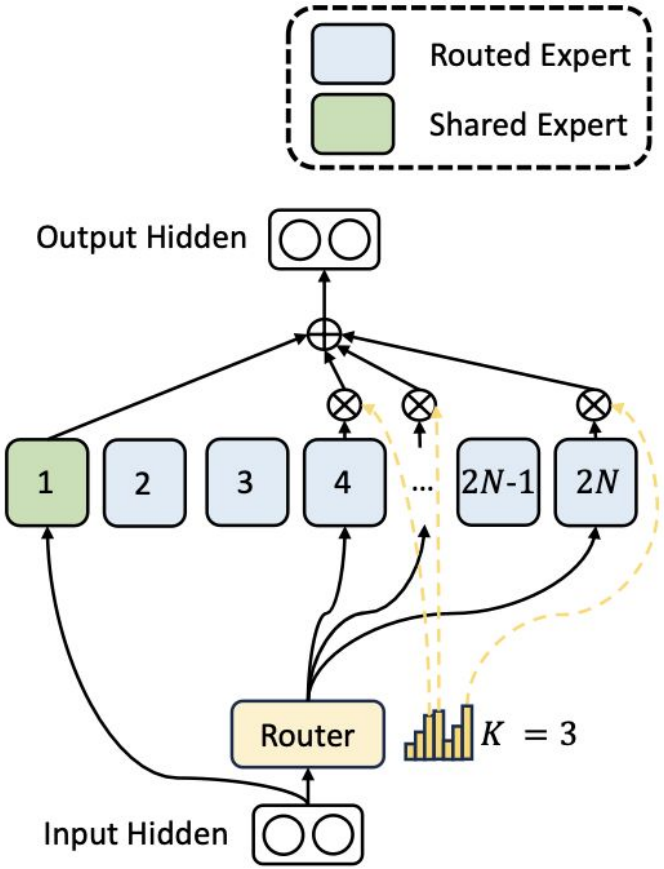
Methodology: Shared Expert Isolation

- This method aims to solve the problem of general-world knowledge is repeatedly stored in many experts.
- It solves it by introducing a shared expert that is always called. This shared expert will uniformly store the general-world knowledge.
- Formally the MoE routing becomes:

$$\mathbf{h}_t^l = \sum_{i=1}^{K_s} \mathrm{FFN}_i\left(\mathbf{u}_t^l\right) + \sum_{i=K_s+1}^{mN} \left(g_{i,t}\,\mathrm{FFN}_i\left(\mathbf{u}_t^l\right)\right) + \mathbf{u}_t^l,$$

$$g_{i,t} = \begin{cases} s_{i,t}, & s_{i,t} \in \mathrm{Topk}(\{s_{j,t} | K_s + 1 \leqslant j \leqslant mN\}, mK - K_s), \\ 0, & \text{otherwise}, \end{cases}$$

$$s_{i,t} = \mathrm{Softmax}_i\left(\mathbf{u}_t^{l^T} \mathbf{e}_i^l\right).$$



Methodology: Balancing the load

- From the system performance point-of-view. The traditional MoE methods, when scaled, facing the following problems:
  - **Routing Collapse:** The router tends to always choose some experts for all tokens and ignoring the other experts.
  - **Load Imbalance:** When experts are distributed among many devices, the routing need to consider the balance between devices.
- The work solves it by introducing the following loss when training
  - Expert-level balance loss
  - Device-level balance loss

$$\mathcal{L}_{\text{ExpBal}} = \alpha_1 \sum_{i=1}^{N'} f_i P_i,$$

$$f_i = \frac{N'}{K'T} \sum_{t=1}^{T} \mathbb{1}(\text{Token } t \text{ selects Expert } i),$$

$$P_i = \frac{1}{T} \sum_{t=1}^{T} s_{i,t},$$

$$\mathcal{L}_{\text{DevBal}} = \alpha_2 \sum_{i=1}^{D} f'_i P'_i,$$

$$f'_i = \frac{1}{|\mathcal{E}_i|} \sum_{j \in \mathcal{E}_i} f_j,$$

$$P'_i = \sum_{j \in \mathcal{E}_i} P_j,$$

Experiment Setup

- The proposed DeepSeek-MoE model is trained under 100B and 2T token size.
- It is evaluated under the following tasks:
  - Language Modeling
  - Language understanding and reasoning
  - Reading Comprehension
  - Code generation
  - Closed-book question answering
- The target of the experiment is to demonstrate:
  - Under the same training condition, the DeepSeek-MoE can achieve better performance compared to other MoE methods
  - The DeepSeek-MoE is comparable to the dense counterparties with less activated parameters
  - The proposed modules is working as designed.

Experiment: Same training condition comparison

Metric	# Shot	Dense	Hash Layer	Switch	GShard	DeepSeekMoE
# Total Params	N/A	0.2B	2.0B	2.0B	2.0B	2.0B
# Activated Params	N/A	0.2B	0.2B	0.2B	0.3B	0.3B
FLOPs per 2K Tokens	N/A	2.9T	2.9T	2.9T	4.3T	4.3T
# Training Tokens	N/A	100B	100B	100B	100B	100B
Pile (Loss)	N/A	2.060	1.932	1.881	1.867	**1.808**
HellaSwag (Acc.)	0-shot	38.8	46.2	49.1	50.5	**54.8**
PIQA (Acc.)	0-shot	66.8	68.4	70.5	70.6	**72.3**
ARC-easy (Acc.)	0-shot	41.0	45.3	45.9	43.9	**49.4**
ARC-challenge (Acc.)	0-shot	26.0	28.2	30.2	31.6	**34.3**
RACE-middle (Acc.)	5-shot	38.8	38.8	43.6	42.1	**44.0**
RACE-high (Acc.)	5-shot	29.0	30.0	30.9	30.4	**31.7**
HumanEval (Pass@1)	0-shot	0.0	1.2	2.4	3.7	**4.9**
MBPP (Pass@1)	3-shot	0.2	0.6	0.4	0.2	**2.2**
TriviaQA (EM)	5-shot	4.9	6.5	8.9	10.2	**16.6**
NaturalQuestions (EM)	5-shot	1.4	1.4	2.5	3.2	**5.7**

Table 1 | Evaluation results for validation experiments. **Bold** font indicates the best. Compared with other MoE architectures, DeepSeekMoE exhibits a substantial performance advantage.

Experiment: Compared with dense model

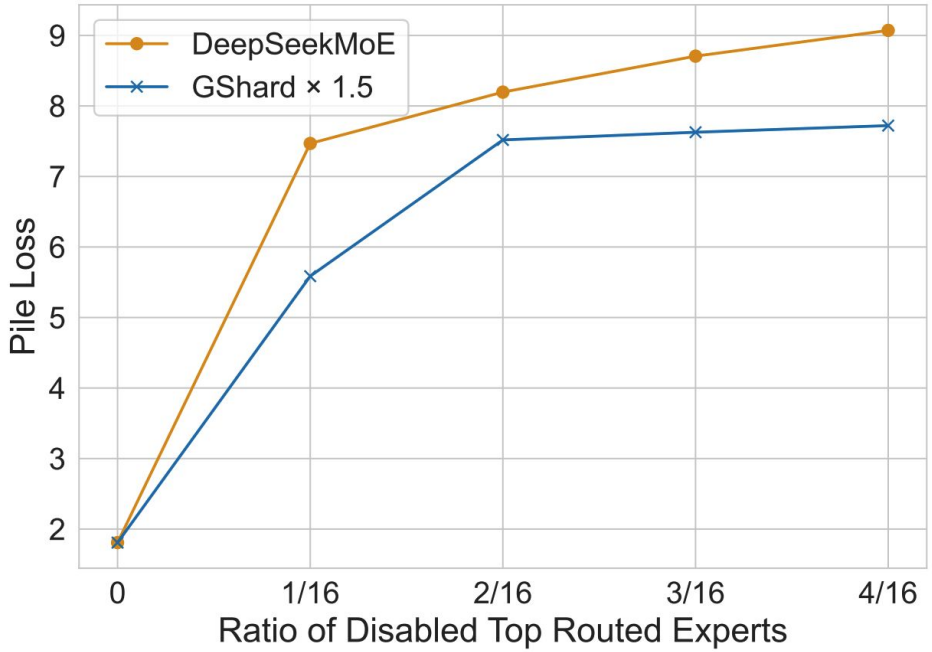
Metric	# Shot	DeepSeek 7B (Dense)	DeepSeekMoE 16B
# Total Params	N/A	6.9B	16.4B
# Activated Params	N/A	6.9B	2.8B
FLOPs per 4K Tokens	N/A	183.5T	74.4T
# Training Tokens	N/A	2T	2T
Pile (BPB)	N/A	0.75	**0.74**
HellaSwag (Acc.)	0-shot	75.4	**77.1**
PIQA (Acc.)	0-shot	79.2	**80.2**
ARC-easy (Acc.)	0-shot	**67.9**	**68.1**
ARC-challenge (Acc.)	0-shot	48.1	**49.8**
RACE-middle (Acc.)	5-shot	**63.2**	61.9
RACE-high (Acc.)	5-shot	**46.5**	**46.4**
DROP (EM)	1-shot	**34.9**	32.9
GSM8K (EM)	8-shot	17.4	**18.8**
MATH (EM)	4-shot	3.3	**4.3**
HumanEval (Pass@1)	0-shot	26.2	**26.8**
MBPP (Pass@1)	3-shot	**39.0**	**39.2**
TriviaQA (EM)	5-shot	59.7	**64.8**
NaturalQuestions (EM)	5-shot	22.2	**25.5**
MMLU (Acc.)	5-shot	**48.2**	45.0
WinoGrande (Acc.)	0-shot	**70.5**	**70.2**
CLUEWSC (EM)	5-shot	**73.1**	72.1
CEval (Acc.)	5-shot	**45.0**	40.6
CMMLU (Acc.)	5-shot	**47.2**	42.5
CHID (Acc.)	0-shot	**89.3**	**89.4**

Table 3 | Comparison between DeepSeek 7B and DeepSeekMoE 16B. **Bold** font indicates the best or near the best. With only 40.5% of computations, DeepSeekMoE 16B achieves comparable performance with DeepSeek 7B.

Metric	# Shot	LLaMA2 7B	DeepSeekMoE 16B
# Total Params	N/A	6.7B	16.4B
# Activated Params	N/A	6.7B	2.8B
FLOPs per 4K Tokens	N/A	187.9T	74.4T
# Training Tokens	N/A	2T	2T
Pile (BPB)	N/A	0.76	**0.74**
HellaSwag (Acc.)	0-shot	75.6	**77.1**
PIQA (Acc.)	0-shot	78.0	**80.2**
ARC-easy (Acc.)	0-shot	**69.1**	68.1
ARC-challenge (Acc.)	0-shot	49.0	**49.8**
RACE-middle (Acc.)	5-shot	60.7	**61.9**
RACE-high (Acc.)	5-shot	45.8	**46.4**
DROP (EM)	1-shot	**34.0**	32.9
GSM8K (EM)	8-shot	15.5	**18.8**
MATH (EM)	4-shot	2.6	**4.3**
HumanEval (Pass@1)	0-shot	14.6	**26.8**
MBPP (Pass@1)	3-shot	21.8	**39.2**
TriviaQA (EM)	5-shot	63.8	**64.8**
NaturalQuestions (EM)	5-shot	**25.5**	**25.5**
MMLU (Acc.)	5-shot	**45.8**	45.0
WinoGrande (Acc.)	0-shot	69.6	**70.2**
CLUEWSC (EM)	5-shot	64.0	**72.1**
CEval (Acc.)	5-shot	33.9	**40.6**
CMMLU (Acc.)	5-shot	32.6	**42.5**
CHID (Acc.)	0-shot	37.9	**89.4**

Table 4 | Comparison between LLaMA2 7B and DeepSeekMoE 16B. With only 39.6% of computations, DeepSeekMoE 16B outperforms LLaMA2 7B on the majority of benchmarks.

Experiment: Ablation Studies



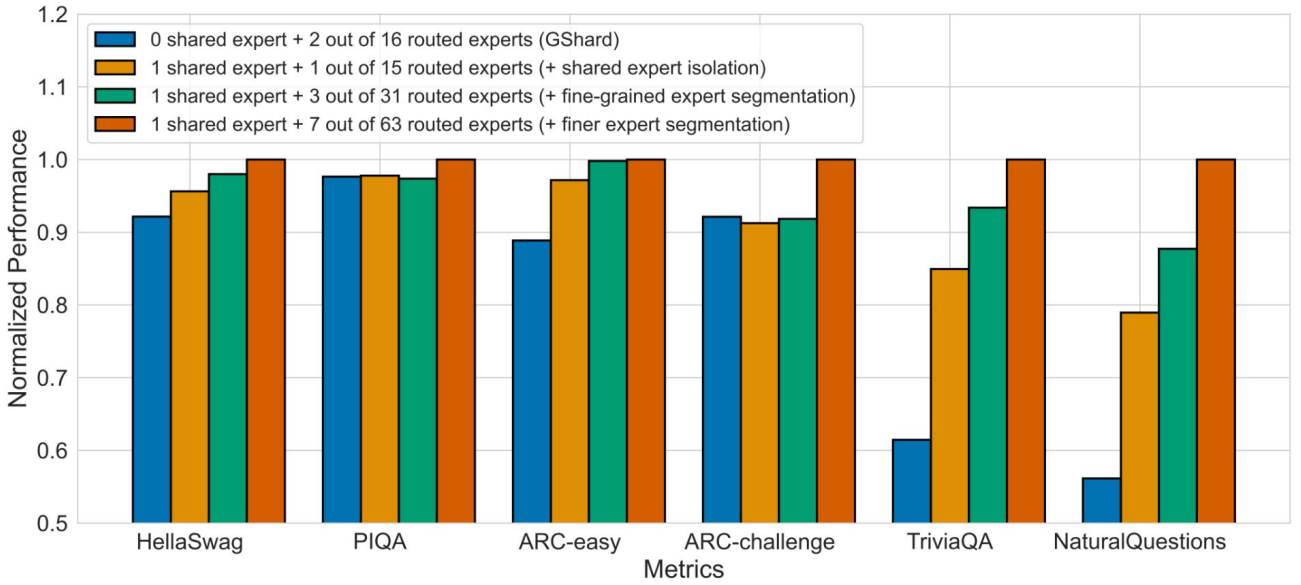


Figure 3 | Ablation studies for DeepSeekMoE. The performance is normalized by the best performance for clarity in presentation. All compared models have the same number of parameters and activated parameters. We can find that fine-grained expert segmentation and shared expert isolation both contribute to stronger overall performance.

MoE In Mainstream AI

Time & Efficiency

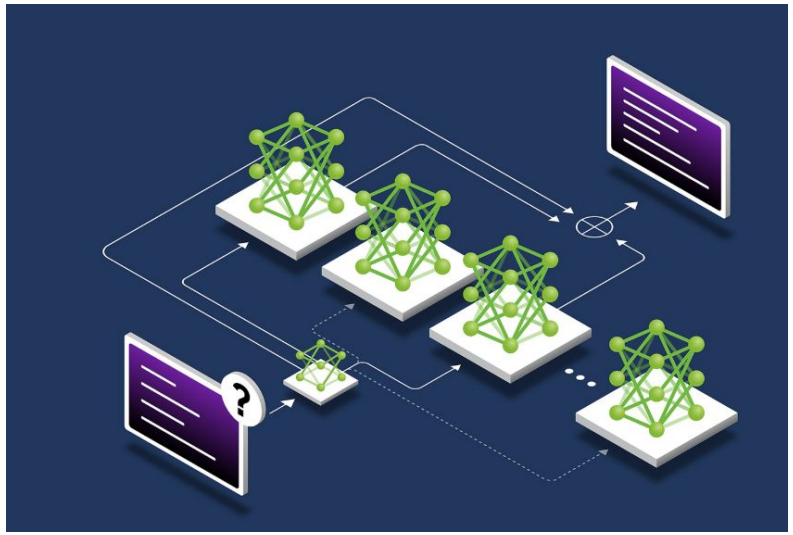
Why do we care about MoE?

In deployment consideration:

- Reducing computation load
- Reducing training cost
- Maintain higher-equivalent model capacity

In security and robustness consideration:

- Higher robustness
- Reducing chance of false classification



Case study: TIME-MOE: BILLION-SCALE TIME SERIES FOUNDATION MODELS WITH MIXTURE OF EXPERTS

Background & Motivation

For training large time-series foundation model, cost and model size become real limitation, how can we solve the bottleneck?

Solution: As MoE success on LLM, can we use MoE?

Challenge in time series data:

- Time-Series data are heterogeneous, make expert selection difficult
- Time-Series data face generalization limitation

Evaluation Task: Future Prediction, where H denote forecast horizon

$$\hat{\mathbf{X}}_{T+1:T+H} = f_\theta\left(\mathbf{X}_{1:T}\right) \in \mathbb{R}^H$$

Input Toke Embedding:

$$\mathbf{h}_t^0 = \mathrm{SwiGLU}(x_t) = \mathrm{Swish}\left(Wx_t\right) \otimes \left(Vx_t\right),$$

MoE Transformer Block:

$$\begin{aligned}
\mathbf{u}_t^l &= \mathrm{SA}\left(\mathrm{RMSNorm}\left(\mathbf{h}_t^{l-1}\right)\right) + \mathbf{h}_t^{l-1}, \\
\bar{\mathbf{u}}_t^l &= \mathrm{RMSNorm}\left(\mathbf{u}_t^l\right), \\
\mathbf{h}_t^l &= \mathrm{Mixture}\left(\bar{\mathbf{u}}_t^l\right) + \mathbf{u}_t^l.
\end{aligned}$$

$$\mathrm{Mixture}\left(\bar{\mathbf{u}}_t^l\right) = g_{N+1,t}\,\mathrm{FFN}_{N+1}\left(\bar{\mathbf{u}}_t^l\right) + \sum_{i=1}^{N}\left(g_{i,t}\,\mathrm{FFN}_i\left(\bar{\mathbf{u}}_t^l\right)\right),$$

$$g_{i,t} = \begin{cases} s_{i,t}, & s_{i,t} \in \mathrm{Topk}(\{s_{j,t} | 1 \le j \le N\}, K), \\ 0, & \text{otherwise}, \end{cases}$$

$$g_{N+1,t} = \mathrm{Sigmoid}\left(\mathbf{W}_{N+1}^l \bar{\mathbf{u}}_t^l\right),$$

$$s_{i,t} = \mathrm{Softmax}_i\left(\mathbf{W}_i^l \bar{\mathbf{u}}_t^l\right),$$

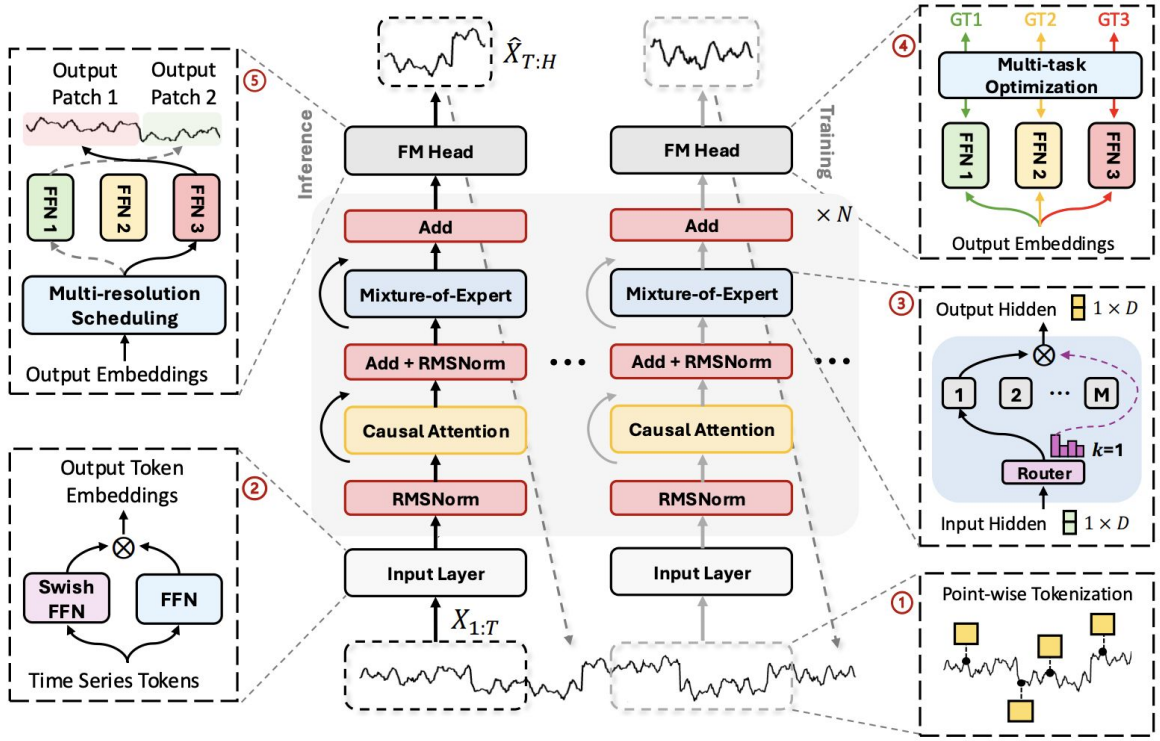


Figure 2: The architecture of TIME-MOE, which is a decoder-only model. Given an input time series of arbitrary length, ① we first tokenize it into a sequence of data points, ② which are then encoded. These tokens are processed through N-stacked backbone layers, primarily consisting of causal multi-head self-attention and ③ sparse temporal mixture-of-expert layers. During training, ④ we optimize forecasting heads at multiple resolutions. For model inference, TIME-MOE provides forecasts of flexible length by ⑤ dynamically scheduling these heads. Details about the causal multi-head self-attention are in Appendix B and illustrated in Figure 5.

Dataset: Time-300B

Table 1: Key statistics of the pre-training dataset `Time-300B` from various domains.

	Energy	Finance	Healthcare	Nature	Sales	Synthetic	Transport	Web	Other	Total
# Seqs.	2,875,335	1,715	1,752	31,621,183	110,210	11,968,625	622,414	972,158	40,265	48,220,929
# Obs.	15.981 B	413.696 K	471.040 K	279.724 B	26.382 M	9.222 B	2.130 B	1.804 B	20.32 M	309.09 B
Percent%	5.17 %	0.0001%	0.0001%	90.50 %	0.008 %	2.98%	0.69 %	0.58 %	0.006 %	100%

Training Loss:

$$\mathcal{L} = \frac{1}{P}\sum_{j=1}^{P}\mathcal{L}_{\text{ar}}\left(\mathbf{X}_{t+1:t+p_j}, \hat{\mathbf{X}}_{t+1:t+p_j}\right) + \alpha\mathcal{L}_{\text{aux}},$$

$$\mathcal{L}_{\text{ar}}\left(x_t, \hat{x}_t\right) = \begin{cases} \frac{1}{2}\left(x_t - \hat{x}_t\right)^2, & \text{if } |x_t - \hat{x}_t| \le \delta, \\ \delta \times \left(|x_t - \hat{x}_t| - \frac{1}{2} \times \delta\right), & \text{otherwise,} \end{cases} \qquad \mathcal{L}_{\text{aux}} = N\sum_{i=1}^{N} f_i r_i, \quad f_i = \frac{1}{KT}\sum_{t=1}^{T}\mathbb{I}\left(\text{Time point } t \text{ selects Expert } i\right), \quad r_i = \frac{1}{T}\sum_{t=1}^{T} s_{i,t},$$

Model Configuration:

	Layers	Heads	Experts	K	d_{model}	d_{ff}	d_{expert}	Activated Params	Total Params
TIME-MOE$_{\text{base}}$	12	12	8	2	384	1536	192	50 M	113 M
TIME-MOE$_{\text{large}}$	12	12	8	2	768	3072	384	200 M	453 M
TIME-MOE$_{\text{ultra}}$	36	16	8	2	1024	4096	512	1.1 B	2.4 B

Evaluation: Training Cost Change

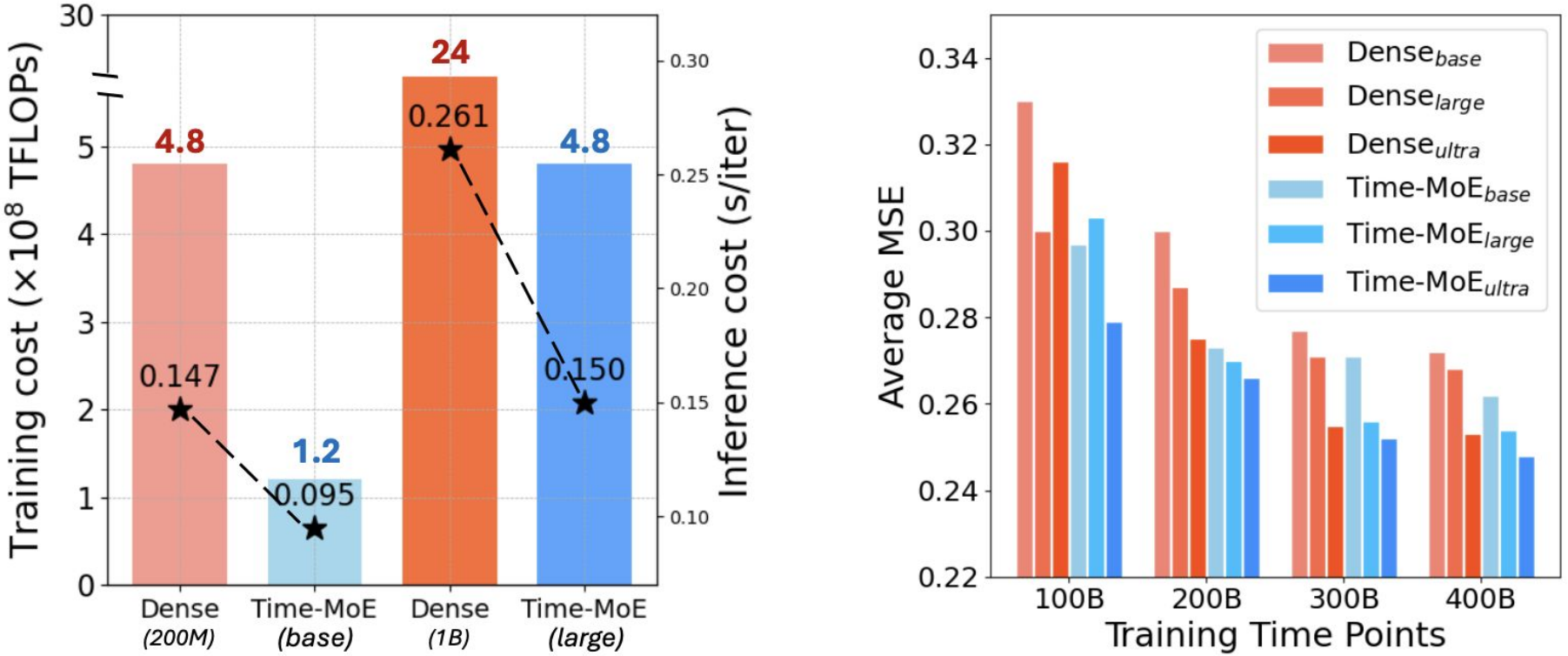


Figure 3: Scalability analysis. **(Left)** Comparison of dense and sparse models in terms of training and inference costs. **(Right)** Average MSE for 96-horizon forecasting across six benchmarks, comparing TIME-MOE and dense models, both trained from scratch with varying data sizes.

Evaluation: Performance Change

Performance Evaluation:

- **Lower** MAE/MSE flag better model performance
- Quantitative comparison highlight the **gain** of time-MoE performance across other SOTA model

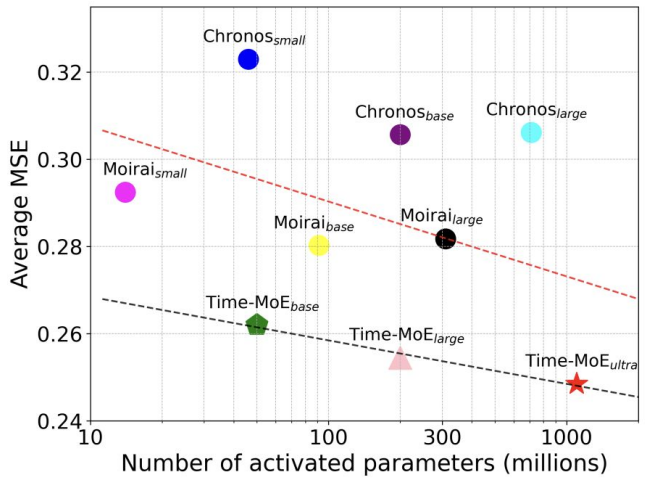


Table 4: Full results of in-domain forecasting experiments. A lower MSE or MAE indicates a better prediction. Full-shot results besides Global Temp are obtained from (Liu et al., 2024b). **Red**: the best, Blue: the 2nd best.

Models		Time-MoE (Ours)						Full-shot Time Series Models															
		Time-MoE$_{base}$		Time-MoE$_{large}$		Time-MoE$_{ultra}$		iTransformer		TimeMixer		TimesNet		PatchTST		Crossformer		TiDE		DLinear		FEDformer	
Metrics		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.345	0.373	0.335	0.371	**0.323**	**0.365**	0.386	0.405	0.375	0.400	0.384	0.402	0.414	0.419	0.423	0.448	0.479	0.464	0.386	0.400	0.376	0.419
	192	0.372	0.396	0.374	0.400	**0.359**	**0.391**	0.441	0.436	0.436	0.429	0.421	0.429	0.460	0.445	0.471	0.474	0.525	0.492	0.437	0.432	0.420	0.448
	336	0.389	**0.412**	0.390	**0.412**	**0.388**	0.418	0.487	0.458	0.484	0.458	0.491	0.469	0.501	0.466	0.570	0.546	0.565	0.515	0.481	0.459	0.459	0.465
	720	0.410	0.443	**0.402**	**0.433**	0.425	0.450	0.503	0.491	0.498	0.482	0.521	0.500	0.500	0.488	0.653	0.621	0.594	0.558	0.519	0.516	0.506	0.507
	Avg.	0.379	0.406	0.375	**0.404**	**0.373**	0.406	0.454	0.447	0.448	0.442	0.454	0.450	0.468	0.454	0.529	0.522	0.540	0.507	0.455	0.451	0.440	0.459
ETTh2	96	0.276	0.340	0.278	**0.335**	**0.274**	0.338	0.297	0.349	0.289	0.341	0.340	0.374	0.302	0.348	0.745	0.584	0.400	0.440	0.333	0.387	0.358	0.397
	192	0.331	0.371	0.345	0.373	**0.330**	**0.370**	0.380	0.400	0.372	0.392	0.402	0.414	0.388	0.400	0.877	0.656	0.528	0.509	0.477	0.476	0.429	0.439
	336	0.373	0.402	0.384	0.402	**0.362**	**0.396**	0.428	0.432	0.386	0.414	0.452	0.541	0.426	0.433	1.043	0.731	0.643	0.571	0.594	0.541	0.496	0.487
	720	0.404	0.431	0.437	0.437	**0.370**	**0.417**	0.427	0.445	0.412	0.434	0.462	0.657	0.431	0.446	1.104	0.763	0.874	0.679	0.831	0.657	0.463	0.474
	Avg.	0.346	0.386	0.361	0.386	**0.334**	**0.380**	0.383	0.406	0.364	0.395	0.414	0.496	0.386	0.406	0.942	0.683	0.611	0.549	0.558	0.515	0.436	0.449
ETTm1	96	0.286	0.334	0.264	0.325	**0.256**	**0.323**	0.334	0.368	0.320	0.357	0.338	0.375	0.329	0.367	0.404	0.426	0.364	0.387	0.345	0.372	0.379	0.419
	192	0.307	0.358	0.295	0.350	**0.281**	**0.343**	0.377	0.391	0.361	0.381	0.374	0.387	0.367	0.385	0.450	0.451	0.398	0.404	0.380	0.389	0.426	0.441
	336	0.354	0.390	**0.323**	0.376	0.326	**0.374**	0.426	0.420	0.390	0.404	0.410	0.411	0.399	0.410	0.532	0.515	0.428	0.425	0.413	0.413	0.445	0.459
	720	0.433	0.445	**0.409**	**0.435**	0.454	0.452	0.491	0.459	0.454	0.441	0.478	0.450	0.454	0.439	0.666	0.589	0.487	0.461	0.474	0.453	0.543	0.490
	Avg.	0.345	0.381	**0.322**	**0.371**	0.329	0.373	0.407	0.409	0.381	0.395	0.400	0.405	0.387	0.400	0.513	0.495	0.419	0.419	0.403	0.406	0.448	0.452
ETTm2	96	0.172	0.265	**0.169**	0.259	0.183	0.273	0.180	0.264	0.175	**0.258**	0.187	0.267	0.175	0.259	0.287	0.366	0.207	0.305	0.193	0.292	0.203	0.287
	192	0.228	0.306	**0.223**	**0.295**	**0.223**	0.301	0.250	0.309	0.237	0.299	0.249	0.309	0.241	0.302	0.414	0.492	0.290	0.364	0.284	0.362	0.269	0.328
	336	0.281	0.345	0.293	0.341	**0.278**	**0.339**	0.311	0.348	0.298	0.340	0.321	0.351	0.305	0.343	0.597	0.542	0.377	0.422	0.369	0.427	0.325	0.366
	720	0.403	0.424	0.451	0.433	0.425	0.424	0.412	0.407	**0.391**	**0.396**	0.408	0.403	0.402	0.400	1.730	1.042	0.558	0.524	0.554	0.522	0.421	0.415
	Avg.	**0.271**	0.335	0.284	0.332	0.277	0.334	0.288	0.332	0.275	**0.323**	0.291	0.332	0.280	0.326	0.757	0.610	0.358	0.403	0.350	0.400	0.304	0.349
Weather	96	0.151	0.203	**0.149**	**0.201**	0.154	0.208	0.174	0.214	0.163	0.209	0.172	0.220	0.177	0.218	0.158	0.230	0.202	0.261	0.196	0.255	0.217	0.296
	192	0.195	0.246	**0.192**	**0.244**	0.202	0.251	0.221	0.254	0.208	0.250	0.219	0.261	0.225	0.259	0.206	0.277	0.242	0.298	0.237	0.296	0.276	0.336
	336	0.247	0.288	**0.245**	**0.285**	0.252	0.287	0.278	0.296	0.251	0.287	0.280	0.306	0.278	0.297	0.272	0.335	0.287	0.335	0.283	0.335	0.339	0.380
	720	0.352	0.366	0.352	0.365	0.392	0.376	0.358	0.349	**0.339**	**0.341**	0.365	0.359	0.354	0.348	0.398	0.418	0.351	0.386	0.345	0.381	0.403	0.428
	Avg.	0.236	0.275	**0.234**	0.273	0.250	0.280	0.257	0.278	0.240	**0.271**	0.259	0.286	0.258	0.280	0.258	0.315	0.270	0.320	0.265	0.316	0.308	0.360
Global Temp	96	0.192	0.328	0.192	0.329	**0.189**	**0.322**	0.223	0.351	0.215	0.346	0.250	0.381	0.219	0.349	0.272	0.406	0.223	0.352	0.221	0.354	0.261	0.392
	192	0.238	**0.375**	0.236	**0.375**	**0.234**	0.376	0.282	0.404	0.266	0.393	0.298	0.418	0.269	0.395	0.305	0.435	0.278	0.401	0.257	0.388	0.299	0.423
	336	0.259	**0.397**	0.256	**0.397**	**0.253**	0.399	0.313	0.431	0.313	0.430	0.315	0.434	0.319	0.435	0.352	0.468	0.330	0.440	0.294	0.418	0.341	0.454
	720	0.345	0.465	0.322	0.451	**0.292**	**0.426**	0.393	0.488	0.468	0.536	0.407	0.497	0.452	0.526	0.508	0.562	0.485	0.544	0.380	0.479	0.359	0.469
	Avg.	0.258	0.391	0.251	0.388	**0.242**	**0.380**	0.303	0.419	0.316	0.426	0.318	0.433	0.315	0.426	0.359	0.468	0.329	0.434	0.288	0.410	0.315	0.435
Average		0.306	0.362	0.304	0.359	**0.301**	**0.358**	0.349	0.382	0.337	0.375	0.356	0.400	0.349	0.382	0.560	0.516	0.421	0.439	0.387	0.416	0.375	0.417
1st Count		4		21		33		0		7		0		0		0		0		0		0	

MoE in IoT

Real-time Schedulability

Case study: Scheduling Classifiers for Real-Time Hazard Perception Considering Functional Uncertainty

Background & Motivation

To run real time hazard detection, suppose we choose MoE model structure, how to balance the tradeoff between performance and latency/false-alarm? In edge device, equivalent expert run might be prohibited. What should be the order of exper to run?

Problem: How should we arrange the sequence expert to run under latency and minimum FN rate constraints?

- **When all expert return 0, means no hazard**
- **When one expert return 1. means hazard**

- *A*: deepsense_both: Uses both seismic and acoustic data, and processes it using the DeepSense neural network architecture [32].
- *B*: deepsense_both_contras: Similar to *A*, but was trained using contrastive learning [23].
- *C*: deepsense_acoustic: Uses only acoustic data, and processes it using the DeepSense neural network architecture [32].
- *D*: deepsense_seismic: Similar to *C*, but uses only seismic data.
- *E*: cnn_both: Uses both seismic and acoustic data, and processes it using a standard convolutional neural network.
- *F*: cnn_acoustic: Uses only acoustic data, and processes it using a standard convolutional neural network.
- *G*: cnn_seismic: Similar to *F*, but uses only seismic data.

Background: Modality Choice Trade Off

Denotation:

- FP: False Positive
- FN: False Negative
- WCET: Worst-case execution times
- TCET: Typical-case execution times

Generic Observation:

As more modality being applied, FP rate **Increase**, and FN rate **Decrease**. On the contrary, as a runtime cost of model application, WCET and TCET also **Increase**.

Table 4: Extended profile table for the Multi-Modal case study

Binary	Classifiers S	$FP(S)$	$FN(S)$	$WCET(S)$	$TCET(S)$	$ESCAP(S)$
00000	∅	0.0000	1.0000	0	0	CD
00001	A	0.0075	0.1183	0.025121	0.018166	DE
00010	B	0.0042	0.1383	0.023854	0.017788	DE
00011	AB	0.0100	0.0933	0.048975	0.035954	E
00100	C	0.0200	0.3600	0.017554	0.012263	D
00101	AC	0.0250	0.0933	0.042675	0.030429	E
00110	BC	0.0242	0.1067	0.041408	0.030051	D
00111	ABC	0.0275	0.0850	0.066529	0.048217	∅
01000	D	0.0358	0.2083	0.01618	0.011878	C
01001	AD	0.0417	0.0883	0.0413	0.030044	E
01010	BD	0.0383	0.1067	0.040033	0.029666	E
01011	ABD	0.0433	0.0750	0.065154	0.047832	∅
01100	CD	0.0542	0.0850	0.033734	0.024141	∅
01101	ACD	0.0575	0.0700	0.058854	0.042307	∅
01110	BCD	0.0567	0.0767	0.057587	0.041929	∅
01111	ABCD	0.0592	0.0667	0.082708	0.060095	∅
10000	E	0.0250	0.1850	0.0053	0.004112	CD
10001	AE	0.0292	0.0900	0.030421	0.022277	D
10010	BE	0.0275	0.1083	0.029154	0.0219	D
10011	ABE	0.0308	0.0800	0.054274	0.040066	∅
10100	CE	0.0425	0.1267	0.022854	0.016374	D
10101	ACE	0.0458	0.0783	0.047975	0.03454	∅
10110	BCE	0.0450	0.0867	0.046708	0.034162	D
10111	ABCE	0.0475	0.0750	0.071828	0.052328	∅
11000	DE	0.0592	0.0983	0.021479	0.01599	C
11001	ADE	0.0617	0.0700	0.0466	0.034156	∅
11010	BDE	0.0600	0.0850	0.045333	0.033778	∅
11011	ABDE	0.0625	0.0650	0.070454	0.051944	∅
11100	CDE	0.0750	0.0667	0.039033	0.028252	∅
11101	ACDE	0.0767	0.0617	0.064154	0.046418	∅
11110	BCDE	0.0758	0.0650	0.062887	0.046041	∅
11111	ABCDE	0.0775	0.0600	0.088008	0.064206	∅

Background: Expert Correlation

Pearson's correlation :

$$r_{xy} = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{N}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{N}(y_i - \bar{y})^2}}$$

- From behavior, we observed moderate correlation, flag consistent expert prediction and moderate knowledge complement
- From execution time, we observed low correlation, flags independent runtime relationship

Table 2: Behavior: Pearson Correlation Coefficients

	A	B	C	D	E	F	G
A	1	0.931	0.725	0.815	0.860	0.717	0.687
B	0.931	1	0.721	0.832	0.872	0.723	0.6944
C	0.725	0.721	1	0.570	0.687	0.819	0.440
D	0.815	0.832	0.570	1	0.747	0.579	0.699
E	0.860	0.872	0.687	0.747	1	0.711	0.717
F	0.717	0.723	0.819	0.579	0.711	1	0.433
G	0.687	0.694	0.440	0.699	0.717	0.433	1

Table 3: Execution Times: Pearson Correlation Coefficients

	A	B	C	D	E	F	G
A	1	0.031	0.036	-0.011	-0.027	-0.009	0.022
B	0.031	1	0.009	0.024	-0.040	-0.013	-0.024
C	0.036	0.009	1	0.000	0.029	-0.004	0.031
D	-0.011	0.024	0.000	1	-0.020	0.062	-0.058
E	-0.027	-0.040	0.029	-0.020	1	-0.007	0.076
F	-0.009	-0.013	-0.004	0.062	-0.007	1	0.024
G	0.022	-0.024	0.031	-0.058	0.076	0.024	1

Methodology

To better characterize the optimization problem, we define:

Static optimality:
FN rate < H
WCET < L

Typical-Case optimality:
FN rate < H
TCET < L
If TCET< T < WCET
Excape set exist.

Clairvoyant optimality:
FN rate < H
ACET < L

Clairvoyant/Static Optimal Finding:
Given all combinations of expert table, find the entries of combination with minimum FP when FN < H and ACET/WCET < L

Methodology: Typical Optimal Derive

Node Formation

Each node represent a batch of expert to run. Node with already FN < H will have no child, as there is no way to decrease FP by adding more expert to run. They have **SOLID** boundary.

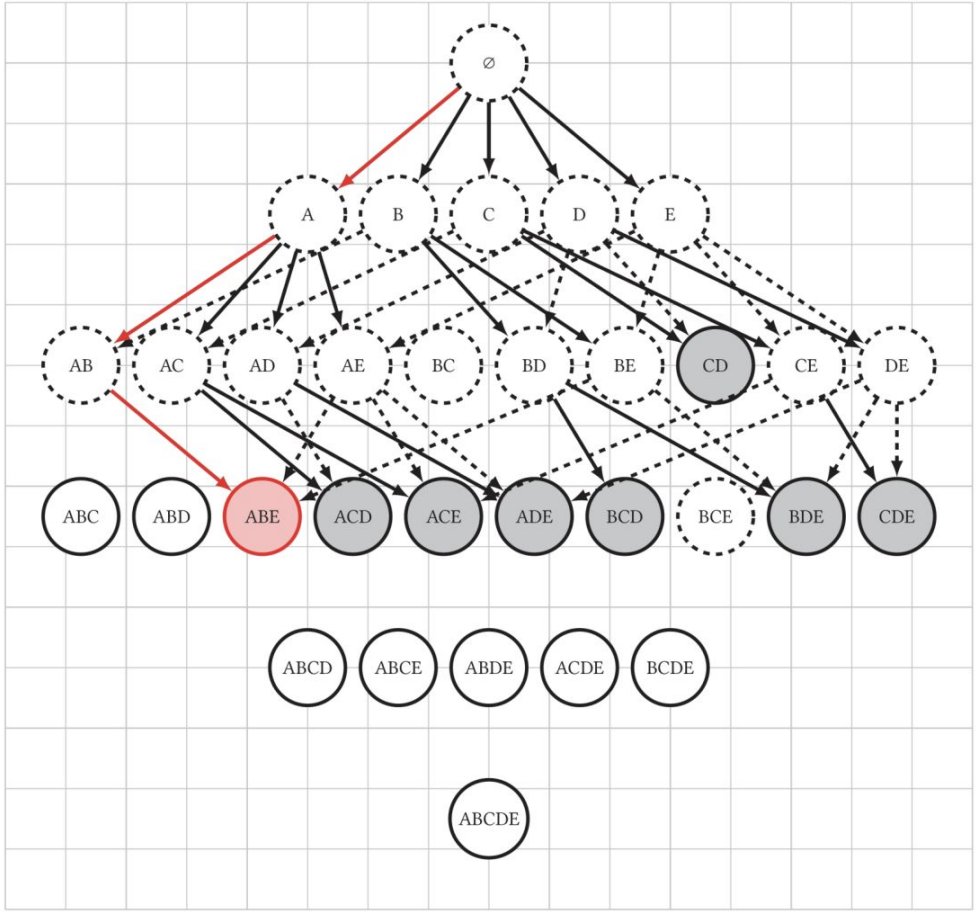
Edge Formation

Each edge iff valid:

$$TCET(P) + WCET(Q - P) + WCET(ESCAP(Q)) \leq L$$

When multiple edge exist, **Solid** edge maximize the slack of time constraint, where the rest and **Dashed**, slack is:

$$L - (TCET(P) + WCET(Q - P) + WCET(ESCAP(Q))$$



Find start reachable leaf with min FP!

Figure 3: DAG representation.

Evaluation

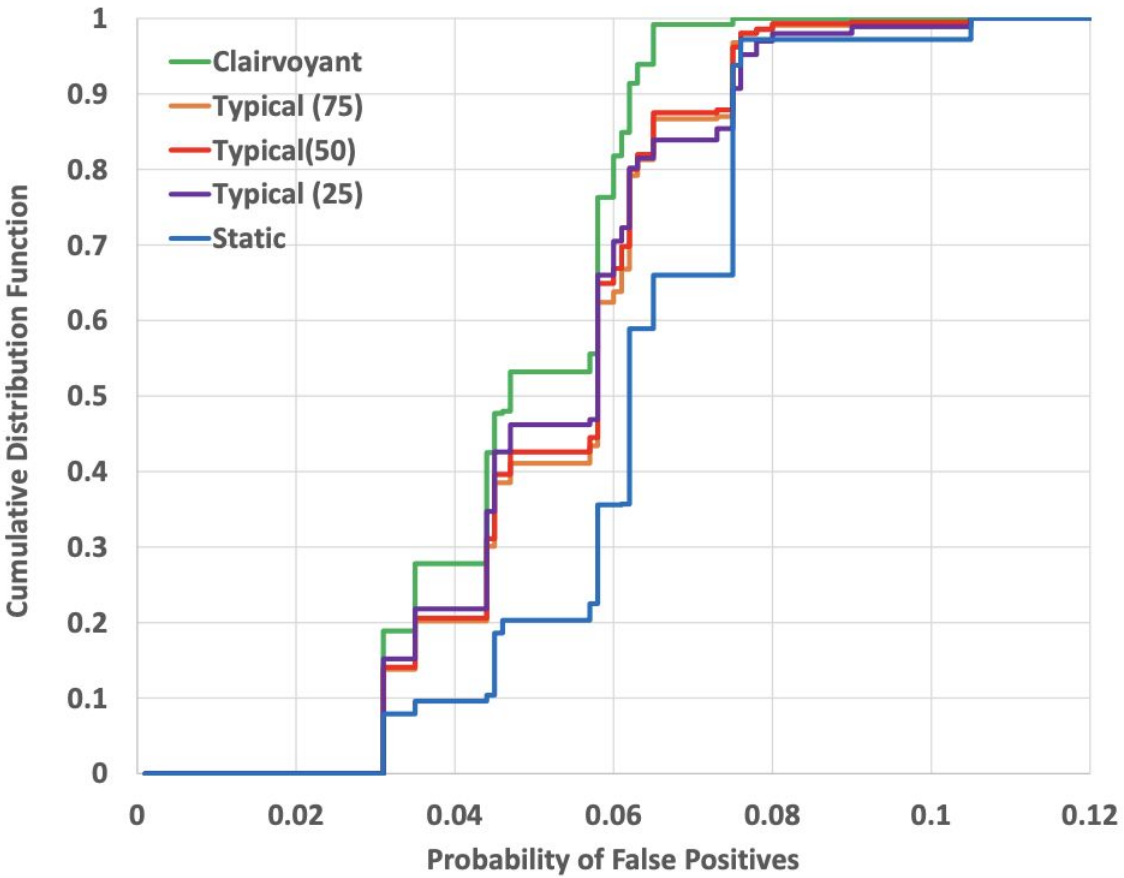


Figure 4: CDF: Multi-Modal case study, 7 classifiers.

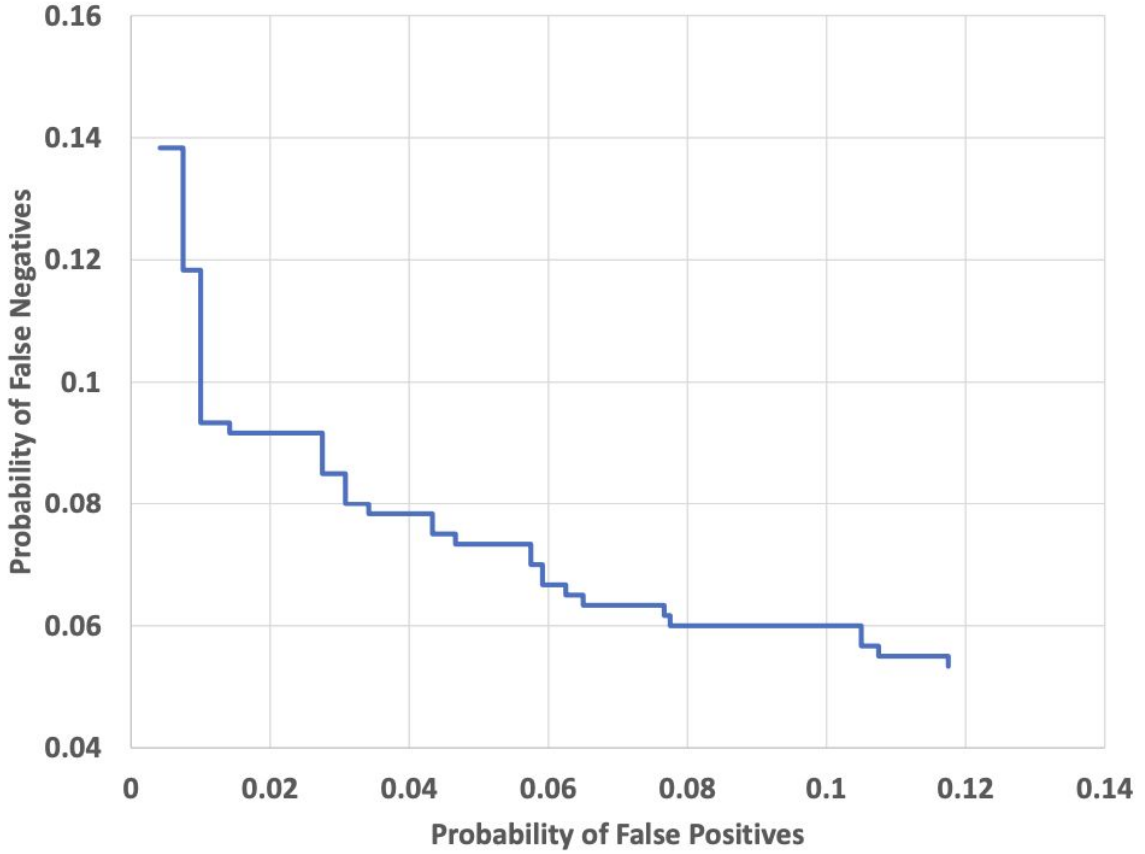
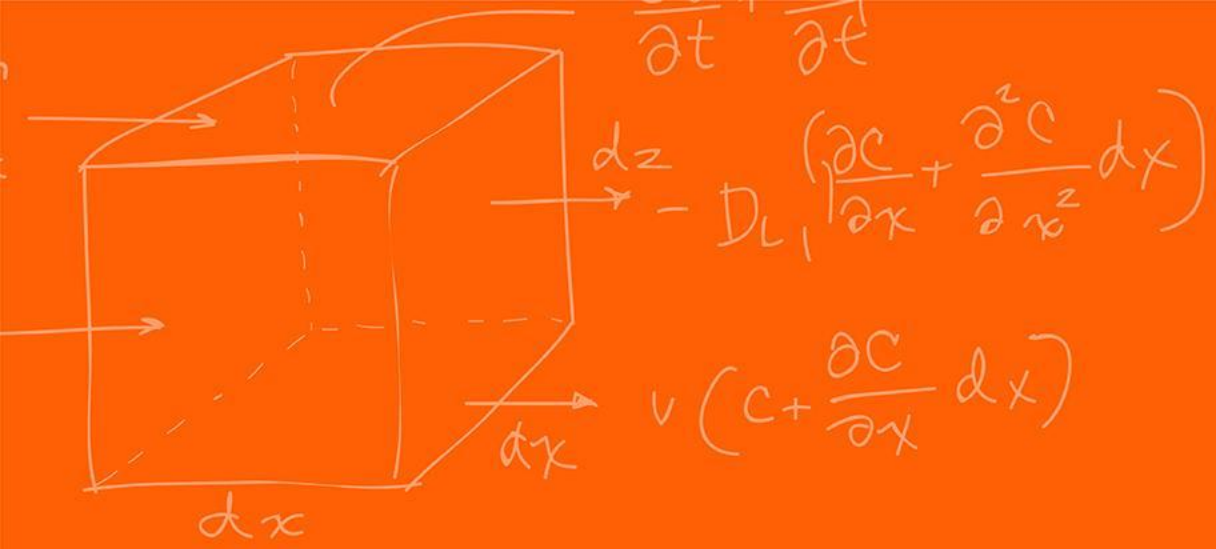
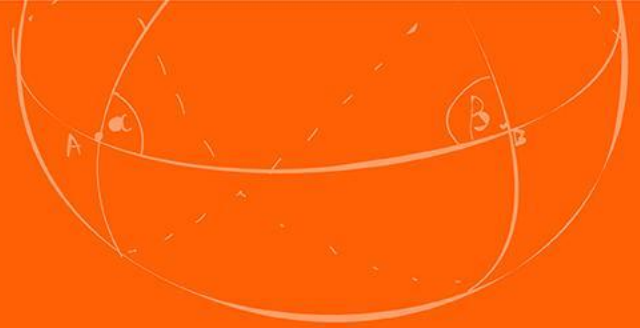


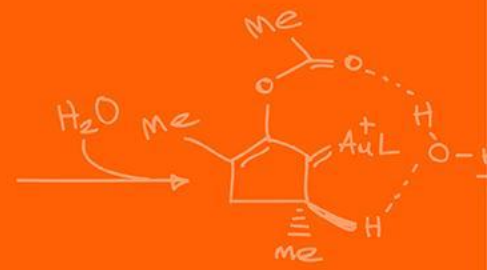
Figure 5: Pareto Front: Multi-Modal case study, 7 classifiers.

MoE in IoT

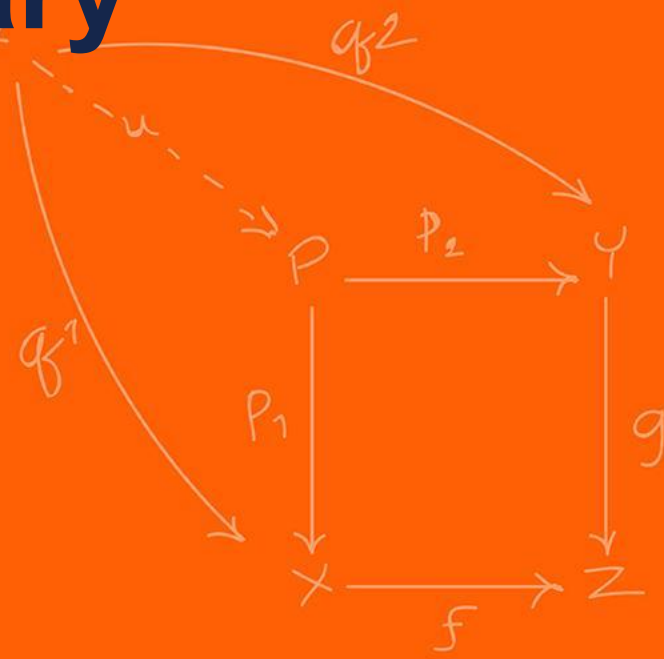
IDK Cascades





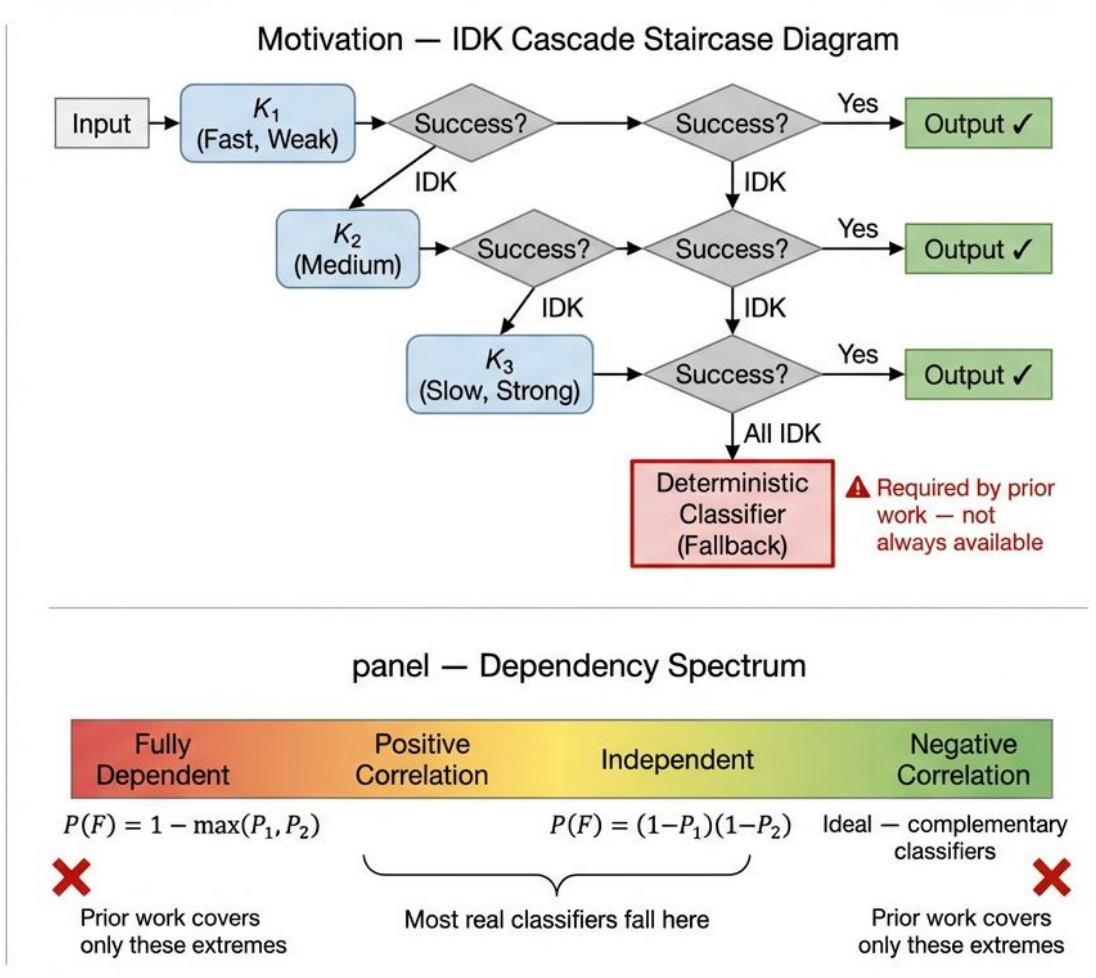


Cascades with Arbitrary Dependencies



Background and Motivation

- IDK classifier: Instead of directly obtain the result, the classifier has the choice to say “I don’t know” (IDK) and pass it on.
- IDK cascade: The scheduling method for IDK classifier. When the weaker classifier says IDK, the strong one will take the position.
- However, it requires a deterministic classifier to be the fail-safe choice.
- When we try to generalize IDK classification problem, we identified the following types of dependencies: Fully dependent, independent, partial (positive/negative) dependent.
- Prior works processes only the fully dependent and a mixture of two types. However, most classifiers are in the middle



System Setting

- The classifier constraints: (C_i, P_i), where C_i means the execution time, and P_i means the success rate.
- The dependency of classifiers

Case	Condition	Meaning
Fully Dependent	$P(F) = 1 - \max(P_1, P_2)$	Inputs that fail the weaker classifier also fail the stronger one
Independent	$P(F) = (1 - P_1)(1 - P_2)$	Failures of the two classifiers do not influence each other
Positive Correlation (Partial Dependency)	$(1 - P_1)(1 - P_2) < P(F) < 1 - \max(P_1, P_2)$	Common among classifiers of the same type
Negative Correlation	$P(F) < (1 - P_1)(1 - P_2)$	Complementary classifiers — **the ideal case**

- In the paper's setting, the algorithm is applied to general models instead of specific classifiers. Additionally, the model execution time is bounded and the successful classification probability is lower-bounded.
- Furthermore, we assume the number of classifier should not exceed 12.

Finding the parameters for the IDK cascades

- The paper further finds the C_i and P_i in the IDK classifiers and their dependency.
- They introduce two cases to study:
  - **Single-modality case:** ResNet classification of images, where the difference in experts is the model size
  - **Multi-modality case:** Acoustic, Seismic, and Vision modalities with four classifiers. Namely deepsense_both_contras (Seismic + Acoustic), cnn_acoustic (Acoustic), deepsense_seismic (Seismic), and yolov5s-compressed (Vision).
- It further calculates the Prob-S and Prob-A based on the data and modalities:
  - Prob-S: Given a specific classifier set S, whether it successfully classifies the thing

    Prob-S therefore denotes the probability that exactly the specific pattern of IDK classifiers indicated by 1's will be able to classify an input, and those indicated by 0's will not and so will return IDK.

  - Prob-A: Whether there exists any classifier in the set S that can successfully classify.

    Prob-A denotes the probability that at least one of IDK classifiers indicated by 1's will be able to classify an input, and is calculated from the Prob-S values.

Using DAG searching to find the cascade algorithm,

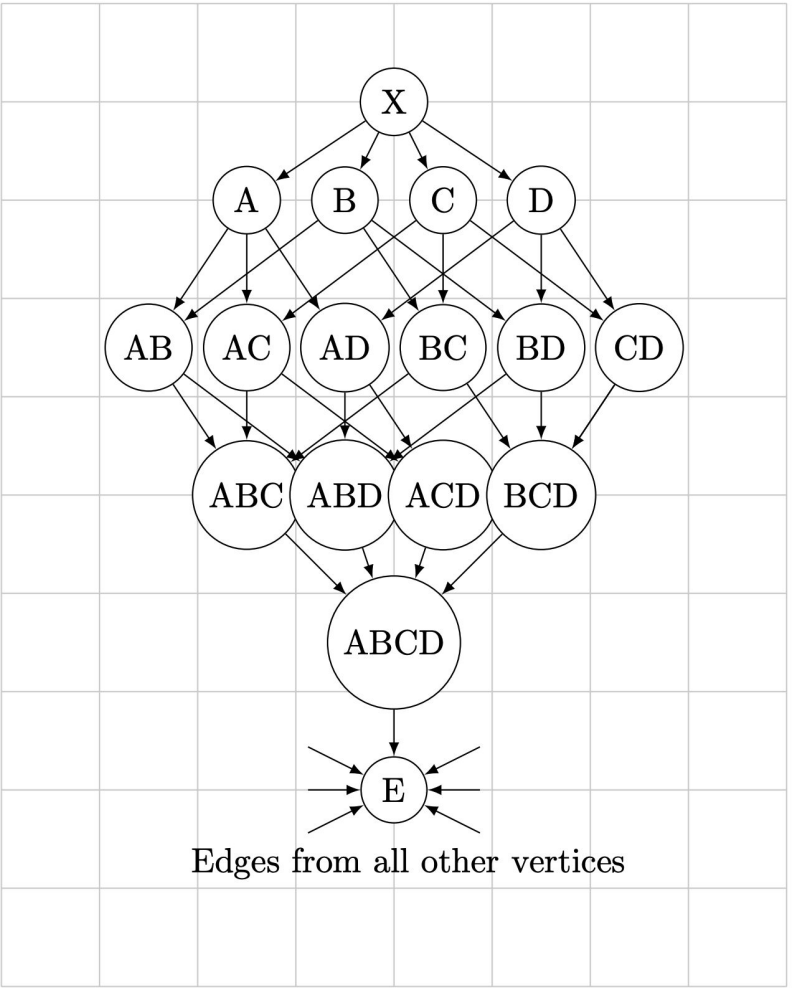
- For K classifiers, the total execution traces we need to search are:

$$\sum_{r=0}^{k} \frac{k!}{(k-r)!}$$

- The DAG setting: when classifier K_i' is in the classification set S, the expected latency addition is:

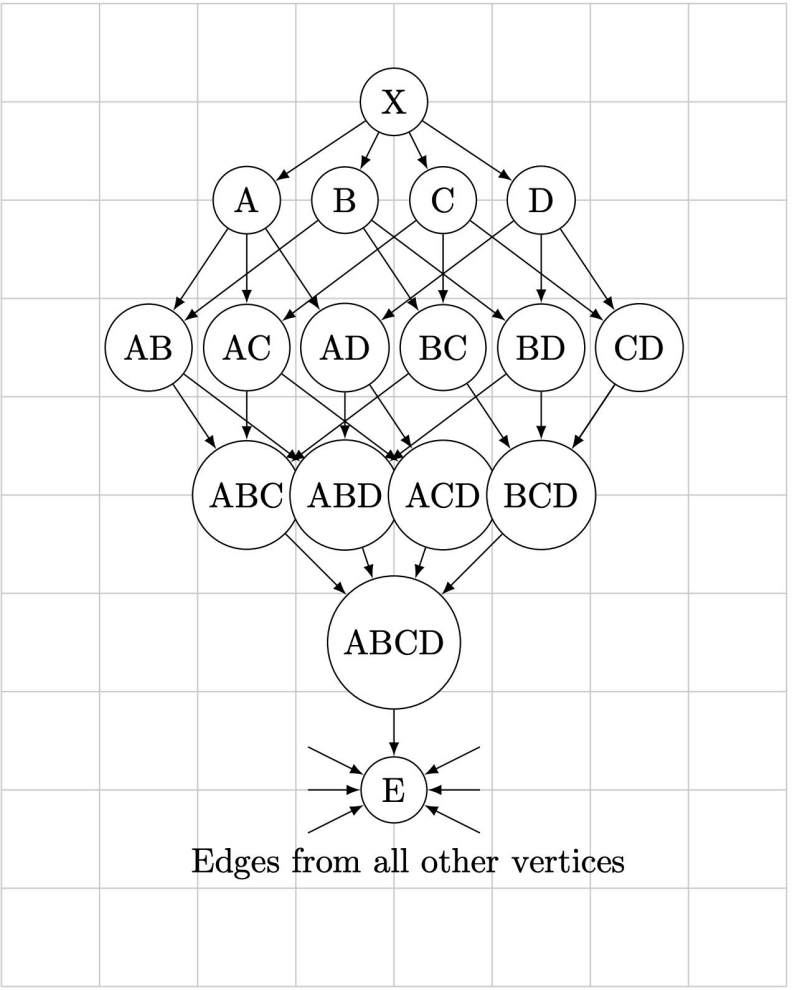
$$\bar{C}_i' \times (1 - \hat{P}[S])$$

  We can observe that is is only related to the elements in S with no relation to the ordering.
- Furthermore, we can have an example DAG as the plot shows



Using DAG searching to find the cascade algorithm

- The edge of DAG should assign the weight of the latency for adding the classifier K
- Because search space for direct search is too large for the DAG, we perform pruning to the graph for more efficient searching.
- Pruning strategies:
  - Prune out every case that exceeds the sum of C_i for all classifier in a set exceeds the wall-time
  - Prune out all set that have $\hat{P} (S) < L$, which is lower than the accuracy constraint
  - If no route in DAG left, then the problem is considered unsolvable.
- The pruning strategy largely reduce the time constraint from O(n!) to O(4^n)
- The algorithm is to find the smallest weight route from the beginning *X* to the end of graph *E*.



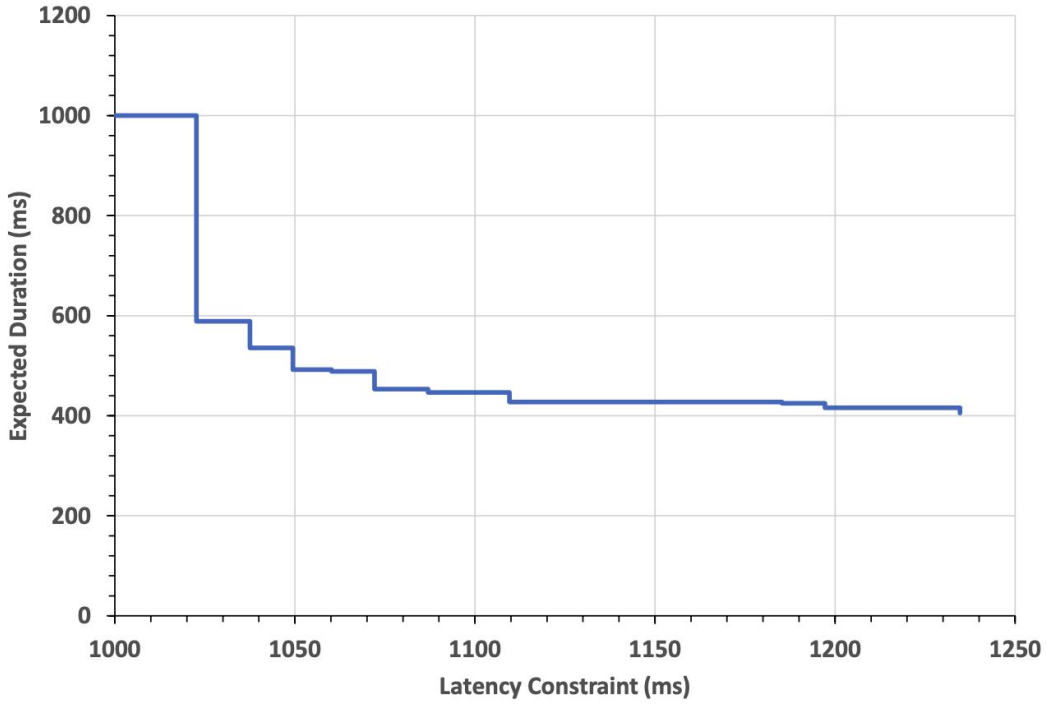
Case study result

- The ResNet case study provides the following results

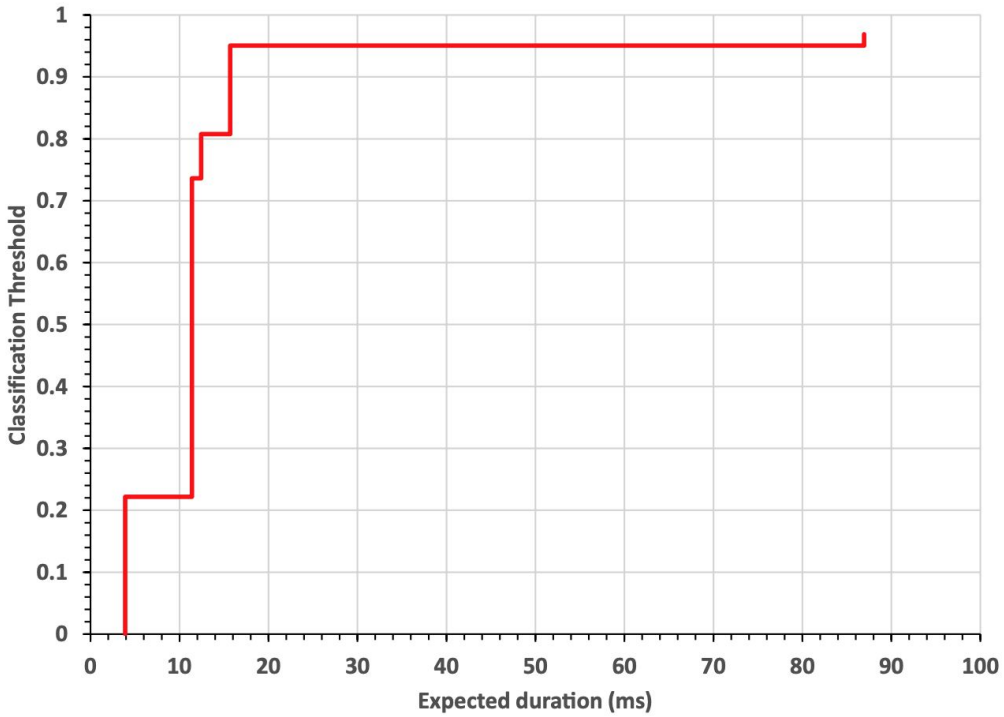
Assumption	Selected Cascade	Estimated Expected Duration	Actual Expected Duration
Independent (Baruah 2021)	$\langle A, B, C, D, E \rangle$ (suboptimal)	111ms (severe underestimate)	405.44ms
Fully Dependent (Baruah 2022)	$\langle A, D, E \rangle$ (suboptimal)	484.49ms (overestimate)	449.93ms
Arbitrary Dependences (This Paper)	$\langle A, C, B, D, E \rangle$ **(optimal)**	405.39ms (accurate)	—

- For multi-modal case study, we find it reaches conclusion that independent or fully dependent behavior do not result in optimal or correct results for IDK classifiers with arbitrary dependencies.
- Thus, it indicates that when the dependencies are not pre-defined, the DAG searching algorithm method for MoE cascade is necessary.

Study on latency constraints and classification threshold



When we tried to add a tighter latency constraint, the expected duration will decrease, this is because the latency constraint will rule-out available choices of schedules



Since there is no “always correct” classifier, we replace it with a threshold L and requires the cascade successful rate to be ≥ L. We find that when the threshold is higher, the duration is larger. However, for ResNet case, the 0.951 would be a good balance point.

Validation

Table 13 ResNet: validation

IDK cascade	$\langle A, B, D \rangle$	$\langle A, C, D \rangle$	$\langle B, C, D \rangle$	$\langle A, B, C, D \rangle$
Expected Duration (ms)	788.50	800.36	870.14	878.45
Average Duration (ms)	766.32	782.44	850.41	853.70
Percentage Difference	2.81%	2.24%	2.27%	2.82%
Probability of Classification	0.65394	0.66412	0.66942	0.6824
Frequency of Classification	0.6266	0.6322	0.6379	0.6459
Difference	2.73%	3.19%	3.15%	3.65%

The paper also applies 10,000 images in ImageNetV2 to verify the accuracy of the ResNet case study schedules' predicated and real time and classification accuracy.

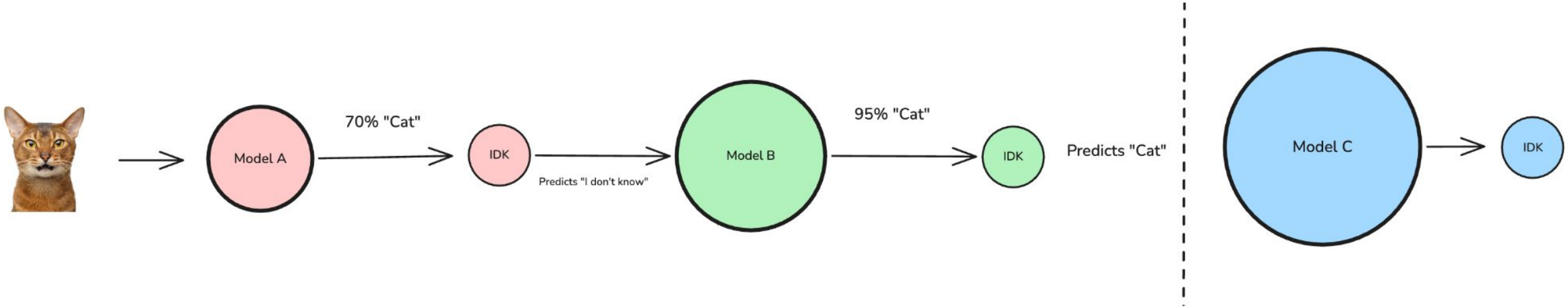
Conclusion for experiments

- The Any dependency assumption for this paper is necessary. Because independent and fully dependent setting may result in submotial cascade and has large difference in estimate the duration
- The optimal cascade is mostly non-trivial and hard to obtained through simple heuristic
- Latency constraint would change the optimal cascade, we have to balance the worst-case and average-case performance
- The classification threshold we proposed would be a effective replacement of the deterministic classifier
- The same set of expert may results in different schedule under the different constraint, there is no globally optimal cascades.
- The optimal cascades is decided by the latency budget and the choice of classification threshold.

Cascades with Time-series Input

Static IDK Classifier

- Assumes no dependencies between different inputs
- Will start from the first model in the cascade every time



IDK Classifiers with Time-series input

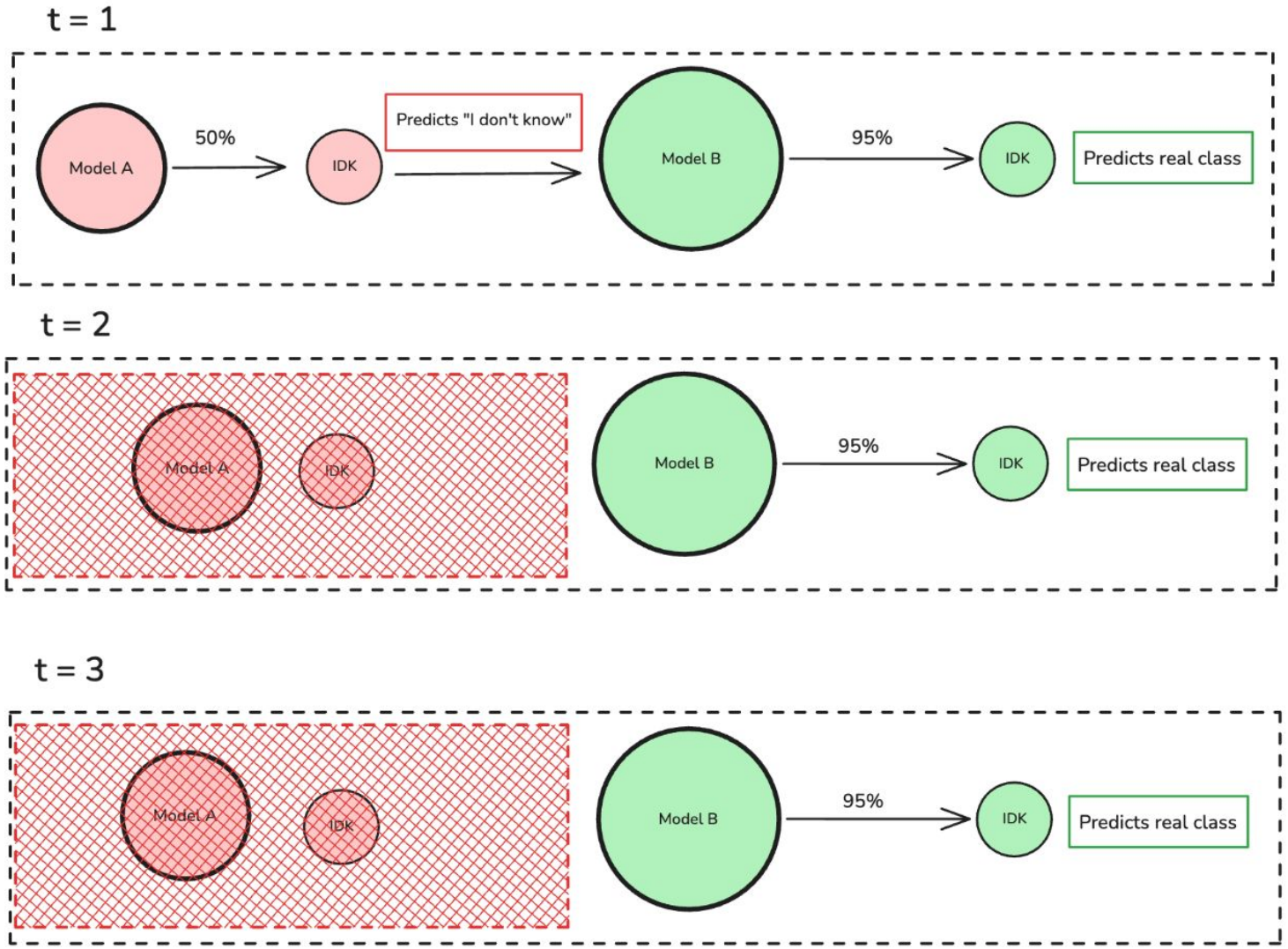
t=1

t=2

t=3

IDK Classifiers with Time-series input

High λ



Kunal Agrawal, Sanjoy Baruah, Alan Burns, and Jinhao Zhao. "IDK cascades for time-series input streams." In 2024 IEEE Real-Time Systems Symposium (RTSS), pp. 83-95. IEEE, 2024.

IDK Classifiers with Time-series input

Algorithm 1: Dynamic $(\mathcal{K} = \langle K_1, K_2, \ldots, K_N \rangle)$

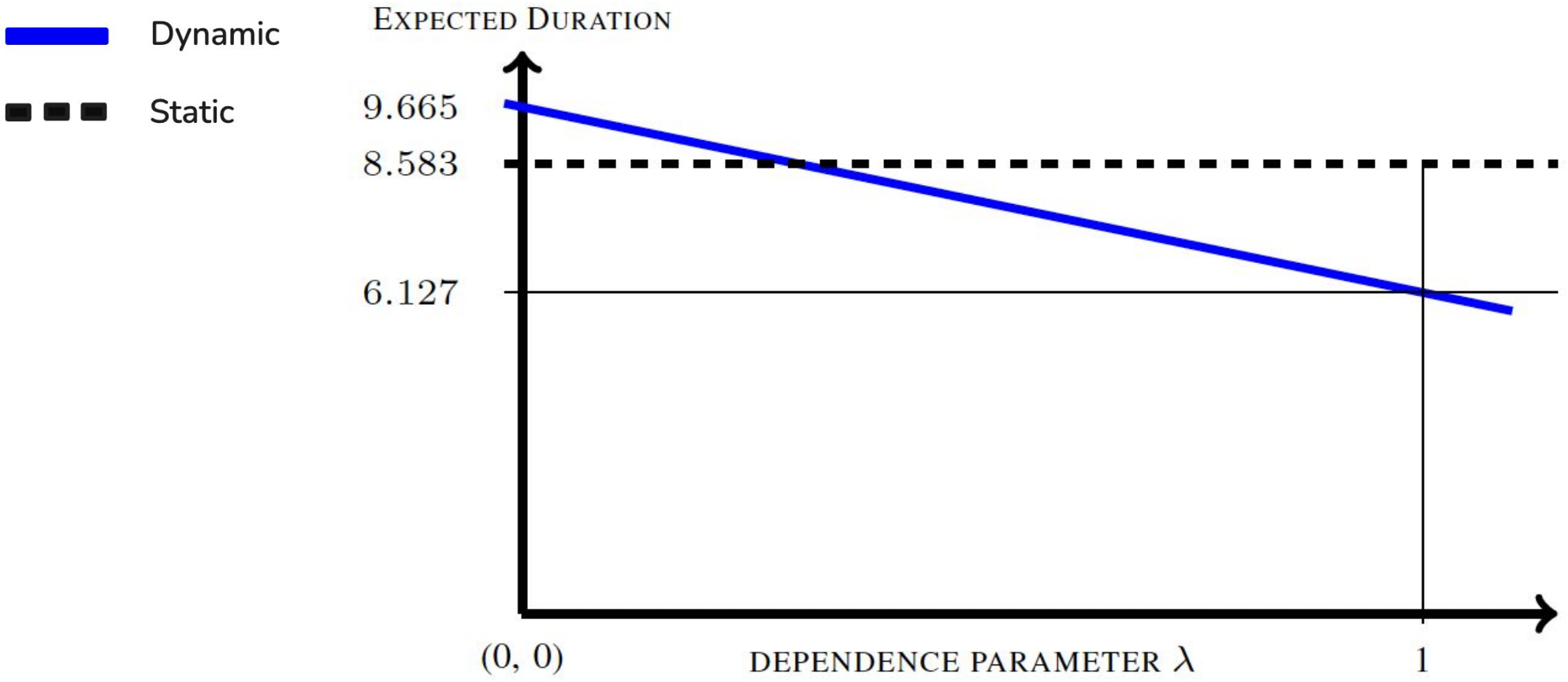
```
1 b = 1 // Start executing the cascade here
2 for each input do
3     Execute classifiers K_b, K_{b+1}, ... until K_i (i ≥ b)
        classifies the input with a non-IDK class
4     output this non-IDK class
5     if (i == b) then // Try and improve b
6         Execute classifiers K_{b-1}, K_{b-2}, ... until K_i
            classifies the input as IDK
7         b = (i + 1) // updating b
8     else
9         b = i
```

Start cascade from b and continue until real output

See if previous classifiers could've done the job

Kunal Agrawal, Sanjoy Baruah, Alan Burns, and Jinhao Zhao. "IDK cascades for time-series input streams." In 2024 IEEE Real-Time Systems Symposium (RTSS), pp. 83-95. IEEE, 2024.

IDK Classifiers with Time-series input



Kunal Agrawal, Sanjoy Baruah, Alan Burns, and Jinhao Zhao. "IDK cascades for time-series input streams." In 2024 IEEE Real-Time Systems Symposium (RTSS), pp. 83-95. IEEE, 2024.

IDK Classifiers with Time-series input

t=1

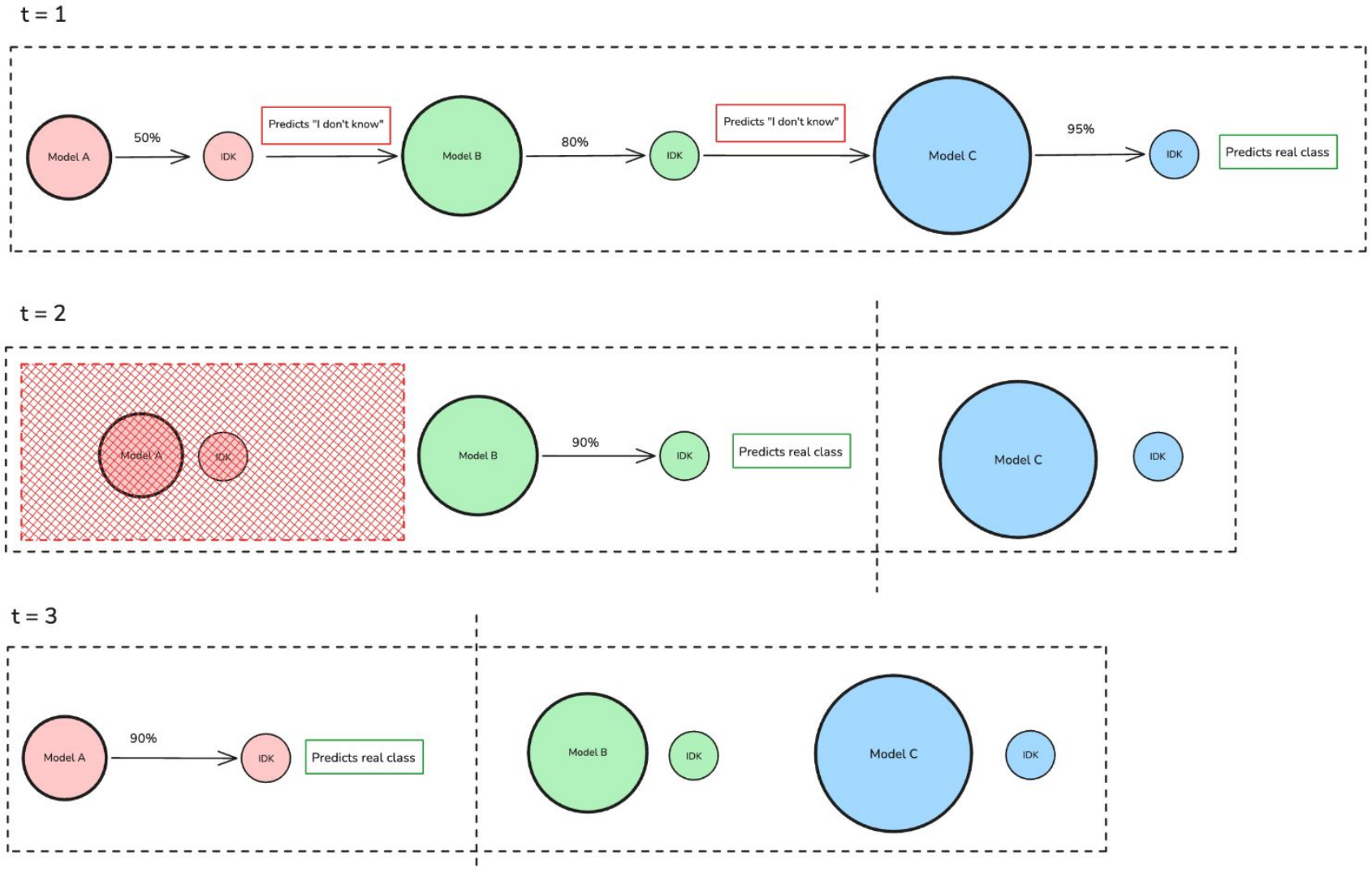
Low λ

t=2

t=3

IDK Classifiers with Time-series input

Low λ



Kunal Agrawal, Sanjoy Baruah, Alan Burns, and Jinhao Zhao. "IDK cascades for time-series input streams." In 2024 IEEE Real-Time Systems Symposium (RTSS), pp. 83-95. IEEE, 2024.

IDK Classifiers with Time-series input

Algorithm 2: Dynamic-k $(\mathcal{K} = \langle K_1, K_2, \ldots, K_N \rangle, k)$

```
   // k is the skip-back factor
1  b = 1 for each input do
2      Execute the classifiers in K in order starting at
         K_max(b−k,1), until some K_i classifies the input
3      output this non-IDK class
4      if (i == b) then // Try and improve b
5          Execute classifiers K_b−1, K_b−2, ... until K_i
             classifies the input as IDK
6          b = (i + 1)
7      else
8          b = i
```

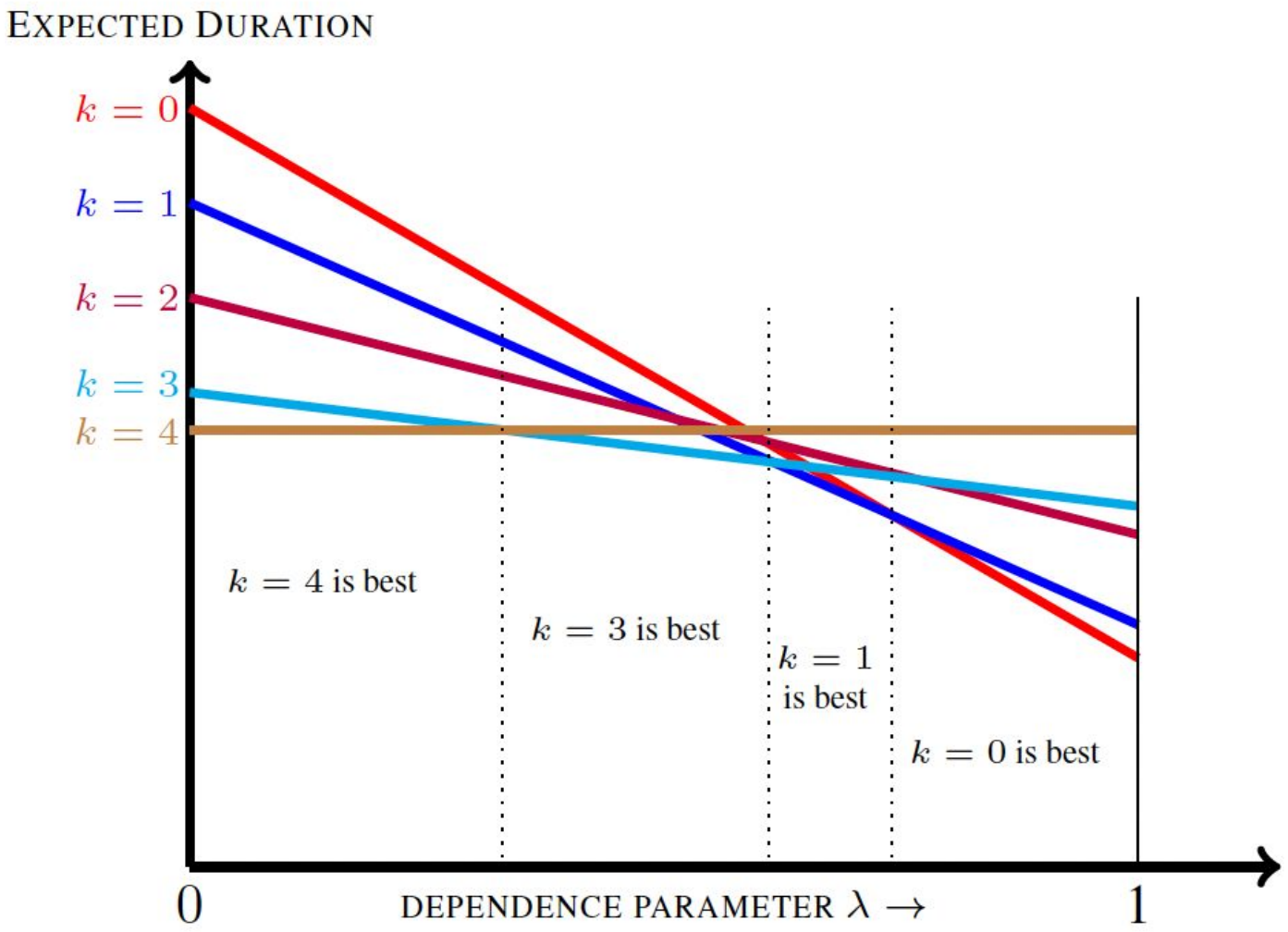
Start cascade from b minus skip-back factor

See if previous classifiers could've done the job

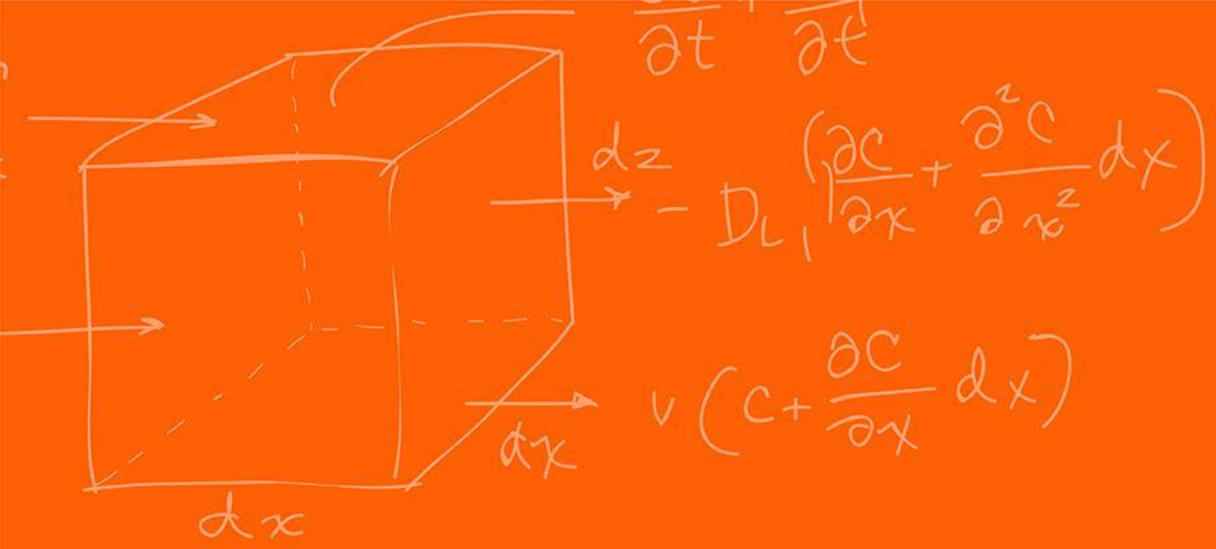
Kunal Agrawal, Sanjoy Baruah, Alan Burns, and Jinhao Zhao. "IDK cascades for time-series input streams." In 2024 IEEE Real-Time Systems Symposium (RTSS), pp. 83-95. IEEE, 2024.

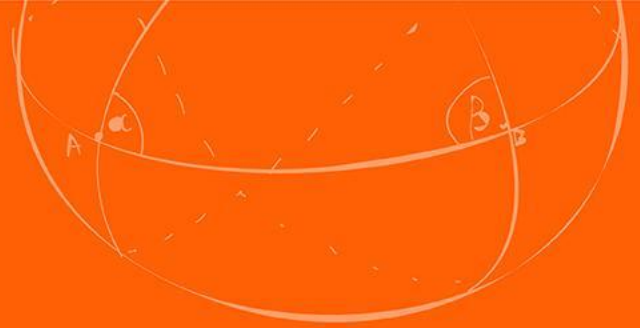
IDK Classifiers with Time-series input

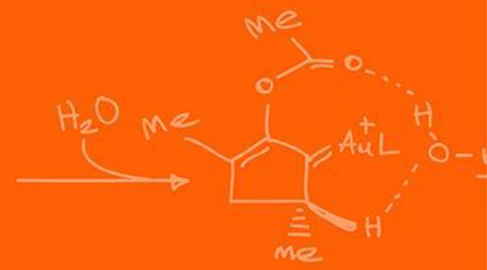
- For maintaining optimal average performance, k (step-back factor) should have an inverse relationship with dependence parameter lambda



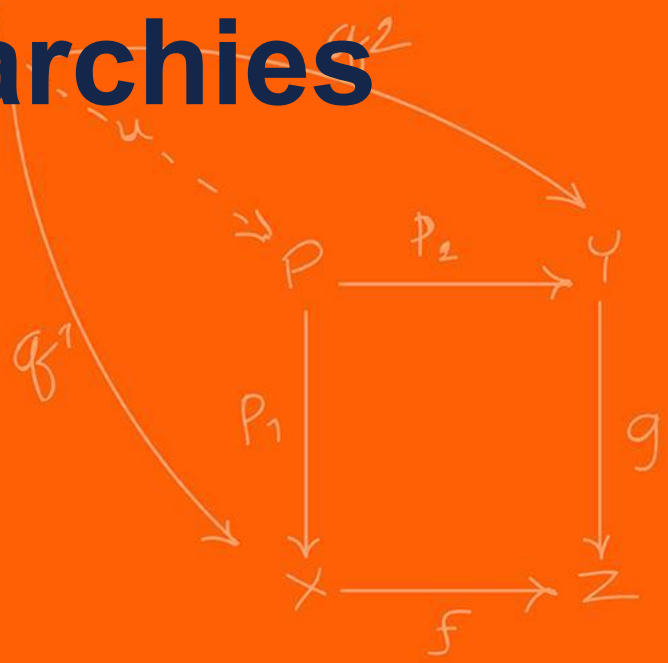
Kunal Agrawal, Sanjoy Baruah, Alan Burns, and Jinhao Zhao. "IDK cascades for time-series input streams." In 2024 IEEE Real-Time Systems Symposium (RTSS), pp. 83-95. IEEE, 2024.







Cascades with Class Hierarchies

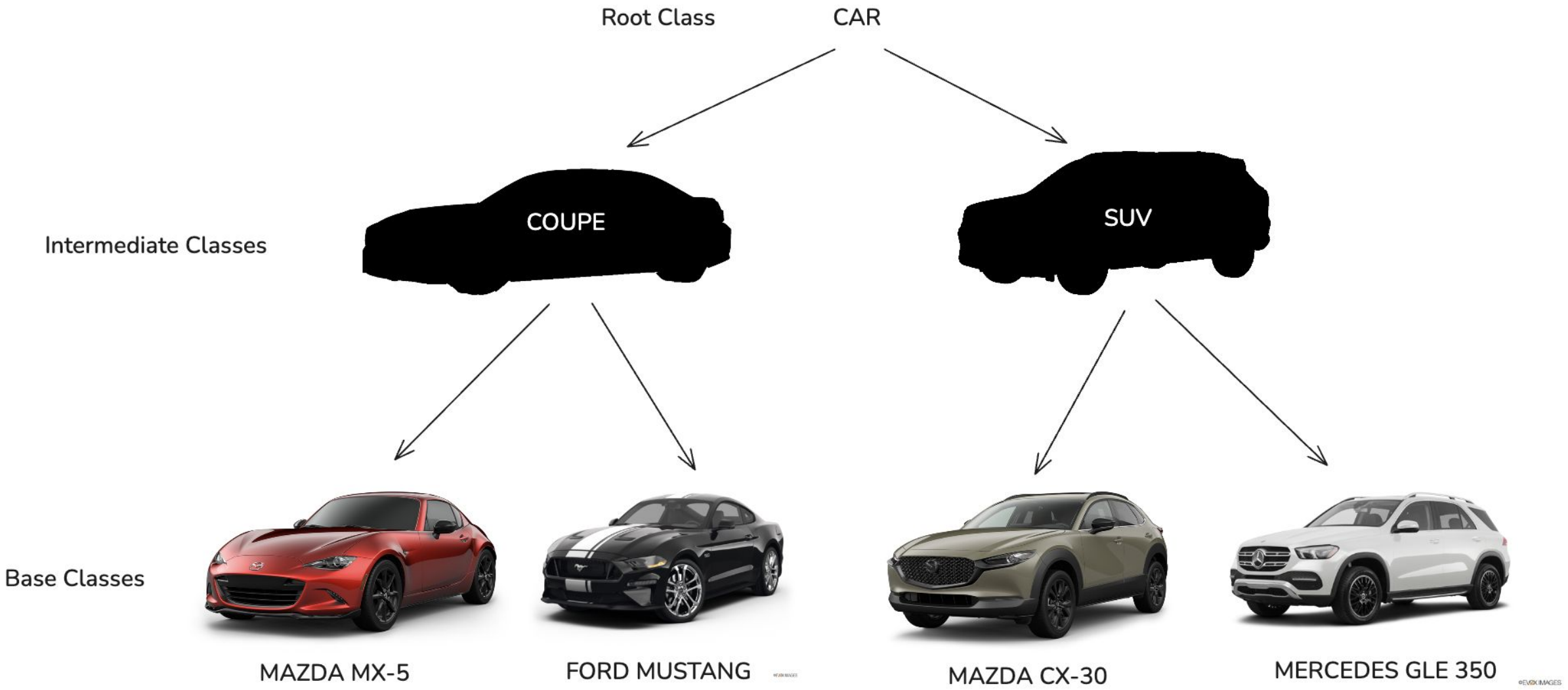


Breaking down the classifiers

- To make things easier for the mixture of “experts”, or our classifiers in these cascades, we can use class hierarchies
- It’s likely easier to distinguish between higher level classes, such as “dog” versus “cat”, than it is to distinguish between specific breeds of dogs.

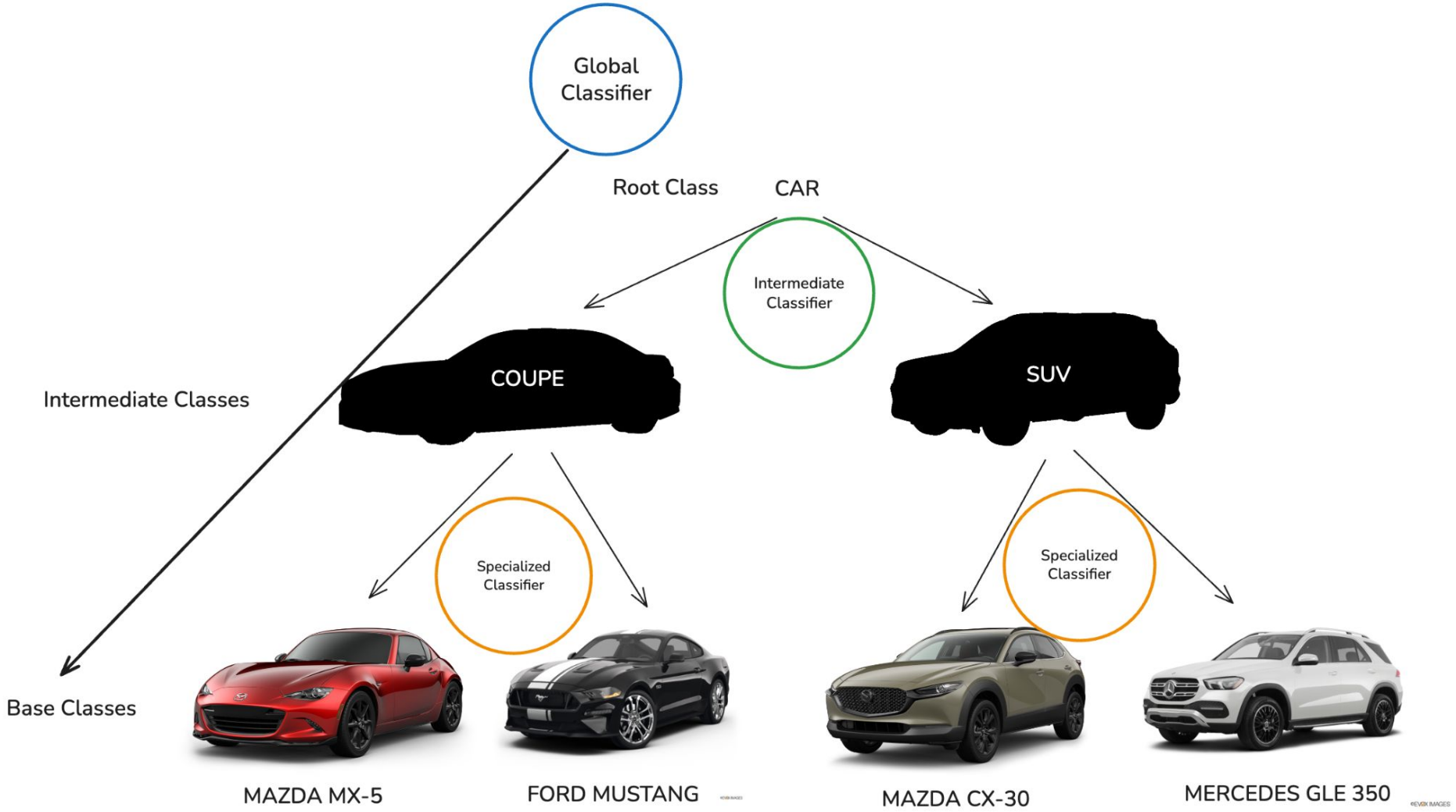
Tarek Abdelzaher, Sanjoy Baruah, Alan Burns, and Yigong Hu. "Timely Classification of Hierarchical Classes." In 2025 IEEE Real-Time Systems Symposium (RTSS), pp. 204-215. IEEE, 2025.

IDK Classifiers with Hierarchies



Tarek Abdelzaher, Sanjoy Baruah, Alan Burns, and Yigong Hu. "Timely Classification of Hierarchical Classes." In 2025 IEEE Real-Time Systems Symposium (RTSS), pp. 204-215. IEEE, 2025.

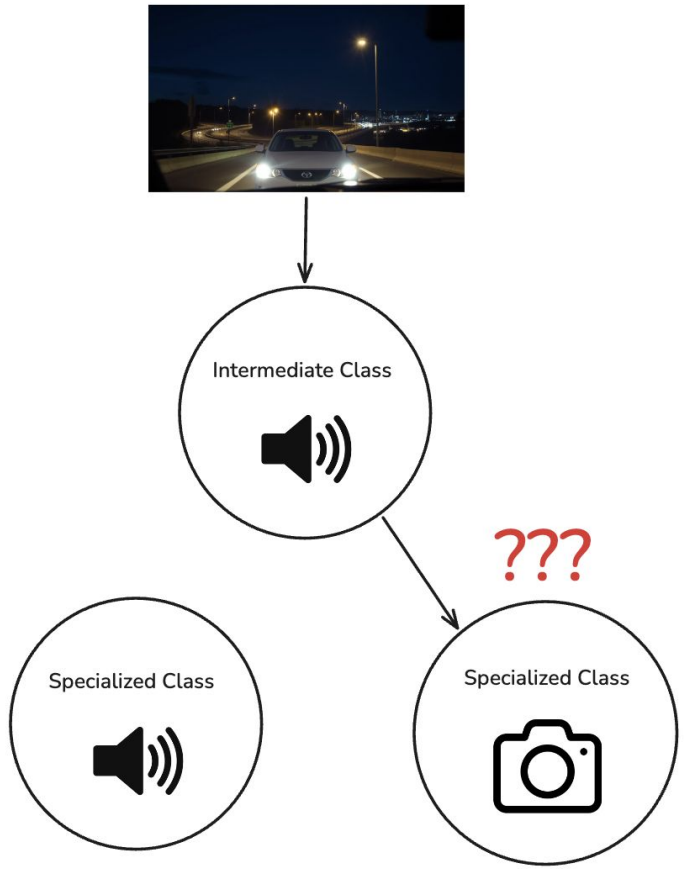
IDK Classifiers with Hierarchies



Tarek Abdelzaher, Sanjoy Baruah, Alan Burns, and Yigong Hu. "Timely Classification of Hierarchical Classes." In 2025 IEEE Real-Time Systems Symposium (RTSS), pp. 204-215. IEEE, 2025.

Optimal cascade with class hierarchies

- While class hierarchies by themselves may yield an improvement over certain non-hierarchical cascades, there a a few other considerations for optimality
- Not as trivial as simply breaking down each “level” of classes into their own individual cascade
- Requires profiling for correlations, an optimal structure may include global classifiers alongside intermediate and specialized classifiers.

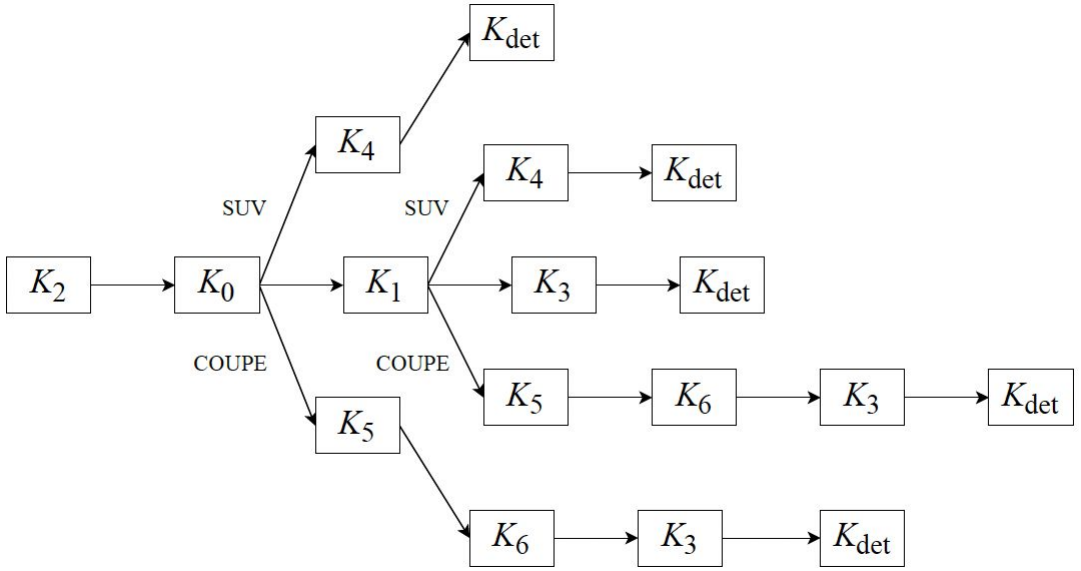


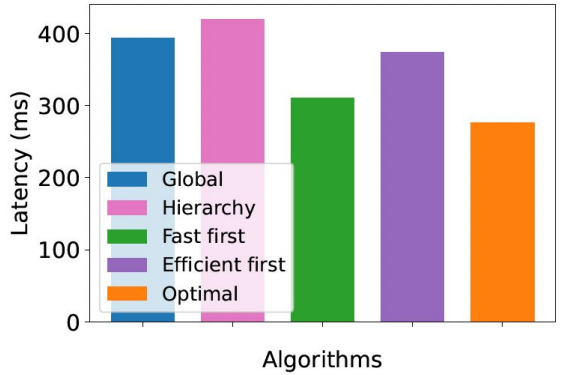
Tarek Abdelzaher, Sanjoy Baruah, Alan Burns, and Yigong Hu. "Timely Classification of Hierarchical Classes." In 2025 IEEE Real-Time Systems Symposium (RTSS), pp. 204-215. IEEE, 2025.

IDK Classifiers with Hierarchies

Optimal cascade with class hierarchies

- While class hierarchies by themselves may yield an improvement over certain non-hierarchical cascades, there a a few other considerations for optimality
- Not as trivial as simply breaking down each “level” of classes into their own individual cascade
- Requires profiling for correlations, an optimal structure may include global classifiers alongside intermediate and specialized classifiers.

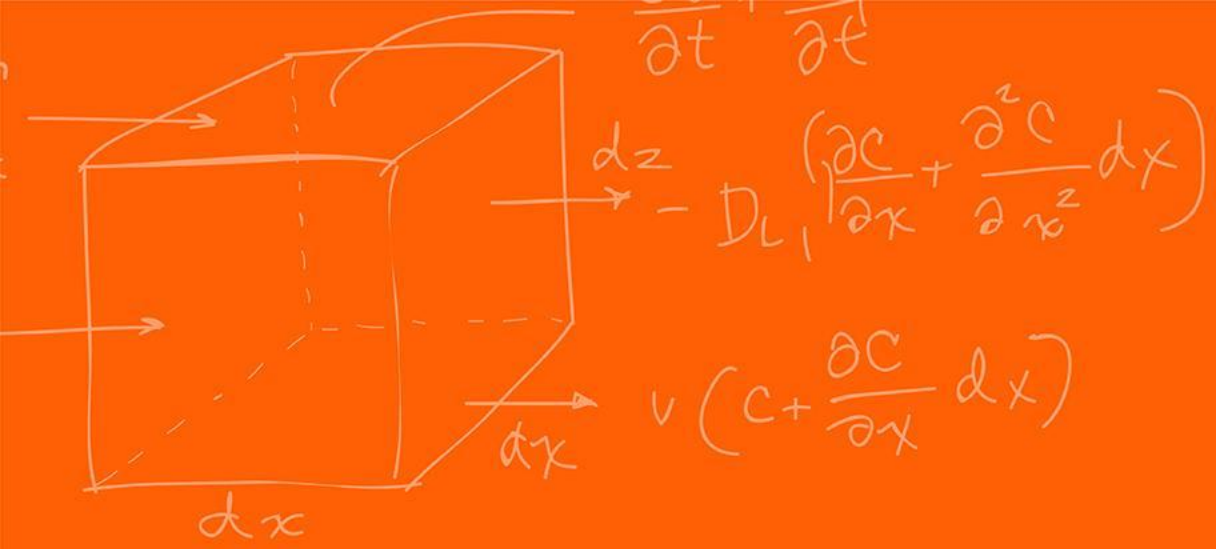


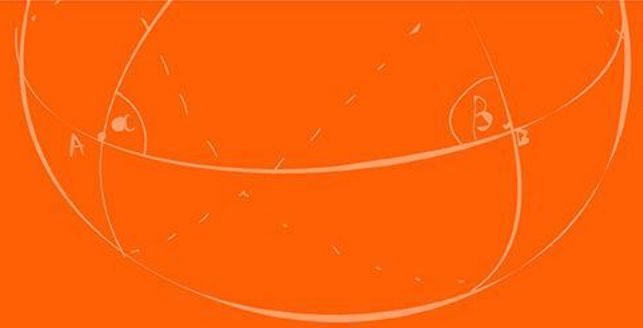


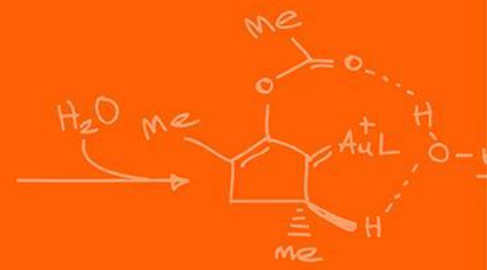
Tarek Abdelzaher, Sanjoy Baruah, Alan Burns, and Yigong Hu. "Timely Classification of Hierarchical Classes." In 2025 IEEE Real-Time Systems Symposium (RTSS), pp. 204-215. IEEE, 2025.

IDK Cascades Takeaways

- IDK cascades can be thought of as a special case of MoE
- We can exploit certain dependencies between classifiers and successive inputs to improve the ordering and performance of cascades







The Grainger College of Engineering

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN