Cross-Sight: Recommending Pre-trained Models using Transformers

Masters Thesis Defense

Parth Patil

December 12, 2025

Advisor: Dr. James C. Davis

Committee : Dr. Qi Guo, Dr. George K. Thiruvathukal

PURDUE UNIVERSITY

DUALITY LAB

Outline

- Outcomes
- Problem Overview
- Background and Related Works
  - Understanding Model Spider (SOTA)
  - Cross Attention vs Self Attention
- Part 1: Vision for PTM Recommendation
  - Gaps in the current landscape
  - Hardware aware PTM Recommendation
- Part 2: Cross-Sight
  - Existing Challenges and Key novel contributions
  - Walkthrough of the pipeline and experimental setup
  - Results and comparisons
- Limitations and Future Work
- Conclusion and Discussion

Outcomes

Thesis Related

- **Recommending Pre-Trained Models for IoT Devices**
  - **Parth V. Patil**, Wenxin Jiang, Huiyun Peng, et al. 2025 IEEE/ACM 47th International Conference on Software Engineering: 7th International Workshop on Software Engineering Research & Practices for the Internet of Things **(SERP4IoT)**, Ottawa, Canada, 2025

- **Cross-Sight [Working Title]**
  - [In preparation] **Parth V. Patil …**

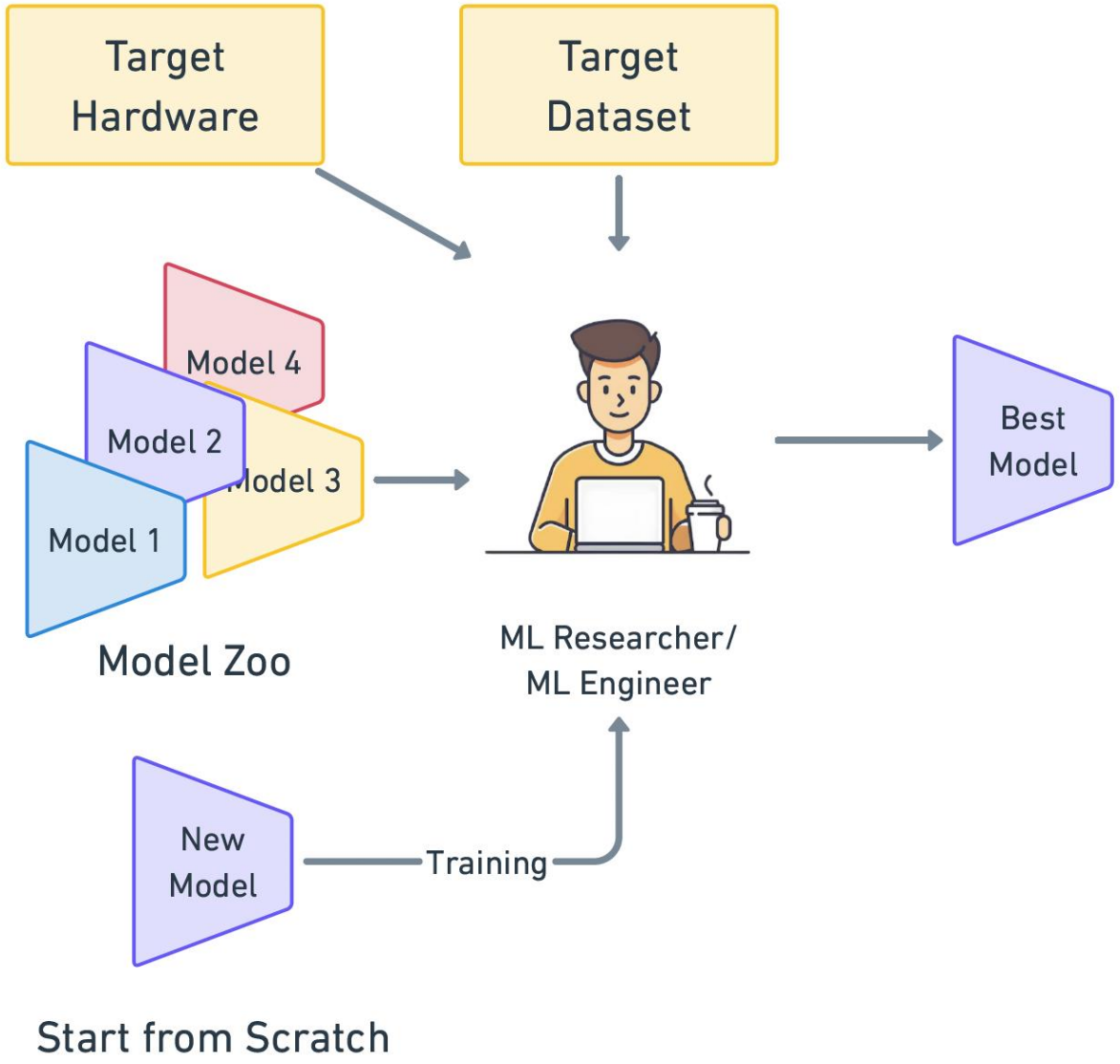
Other Projects

- **A Unit Proofing Framework for Code-level Verification: A Research Agenda**
  - Paschal C. Amusuo, **Parth V. Patil**, Owen Cochell, et al.
  - 2025 IEEE/ACM 47th International Conference on Software Engineering: New Ideas and Emerging Results **(ICSE-NIER)**, Ottawa, ON, Canada, 2025
- **Do Unit Proofs Work? An Empirical Study of Compositional Bounded Model Checking for Memory Safety Verification.**
  - Paschal C Amusuo, Owen Cochell, Taylor Le Lievre, **Parth V Patil**, et al.
  - 48th IEEE/ACM International Conference on Software Engineering (ICSE) 2026.

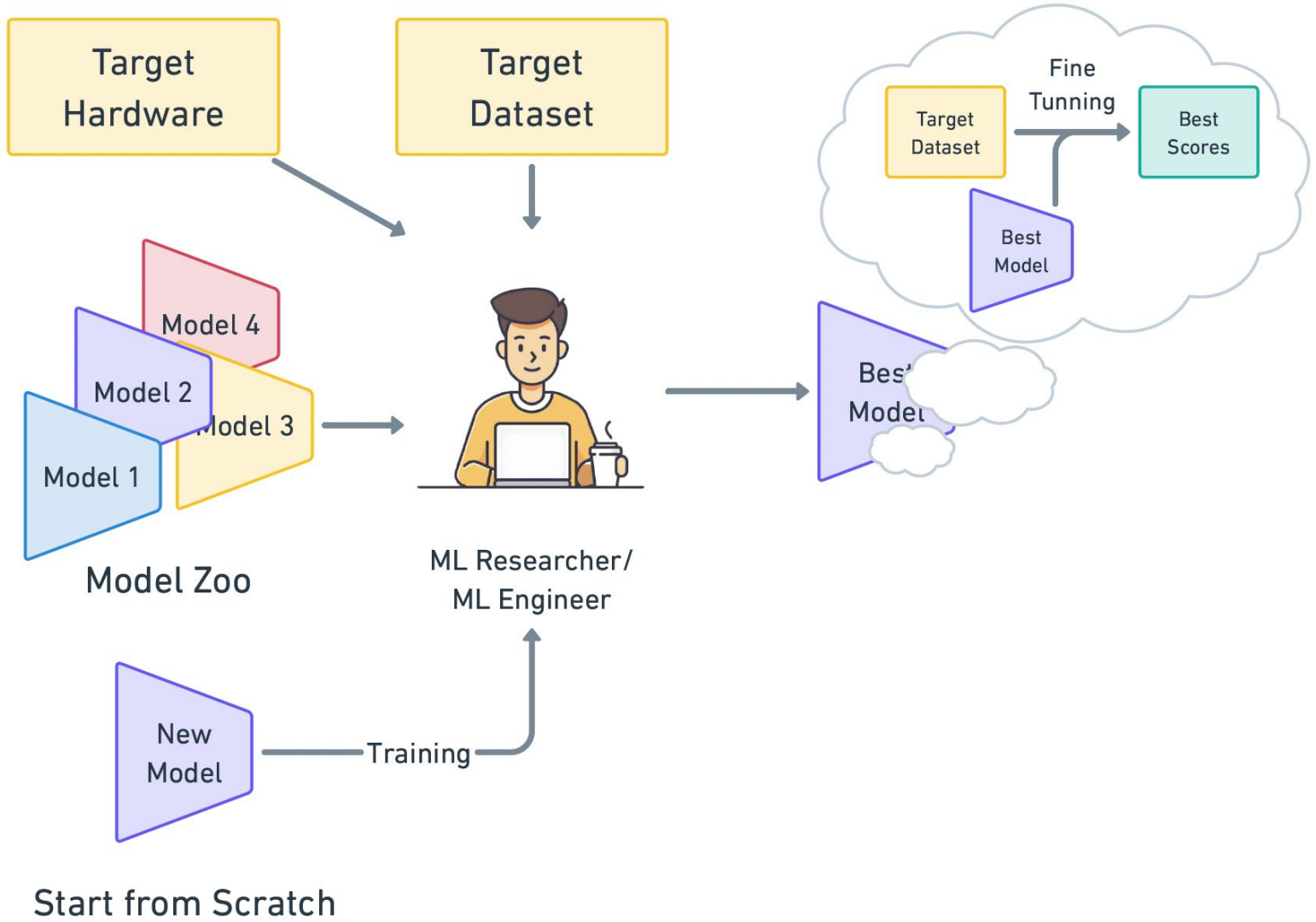
Thesis Summary

- **Given Zoo + New Dataset + Constraints, which PTM to use?**
- “Too many” Pre-Trained Models (PTMs) available on HuggingFace, Kaggle, or other Model Hubs (~2M).
- Identified key-short comings in current leading approach.
- Proposed a vision for integrating additional constraints into selection pipeline.
- Conducted a pilot study demonstrating measurable improvements over SOTA methods, along with detailed evaluation results.

Overview


Target Hardware
Target Dataset
Model 4
Model 2
Model 3
Model 1
Model Zoo
ML Researcher/
ML Engineer
Best Model
New Model
Training
Start from Scratch

Overview


Target Hardware
Target Dataset
Fine Tunning
Target Dataset
Best Scores
Best Model
Model 4
Model 2
Model 3
Model 1
Model Zoo
ML Researcher/ ML Engineer
New Model
Training
Start from Scratch

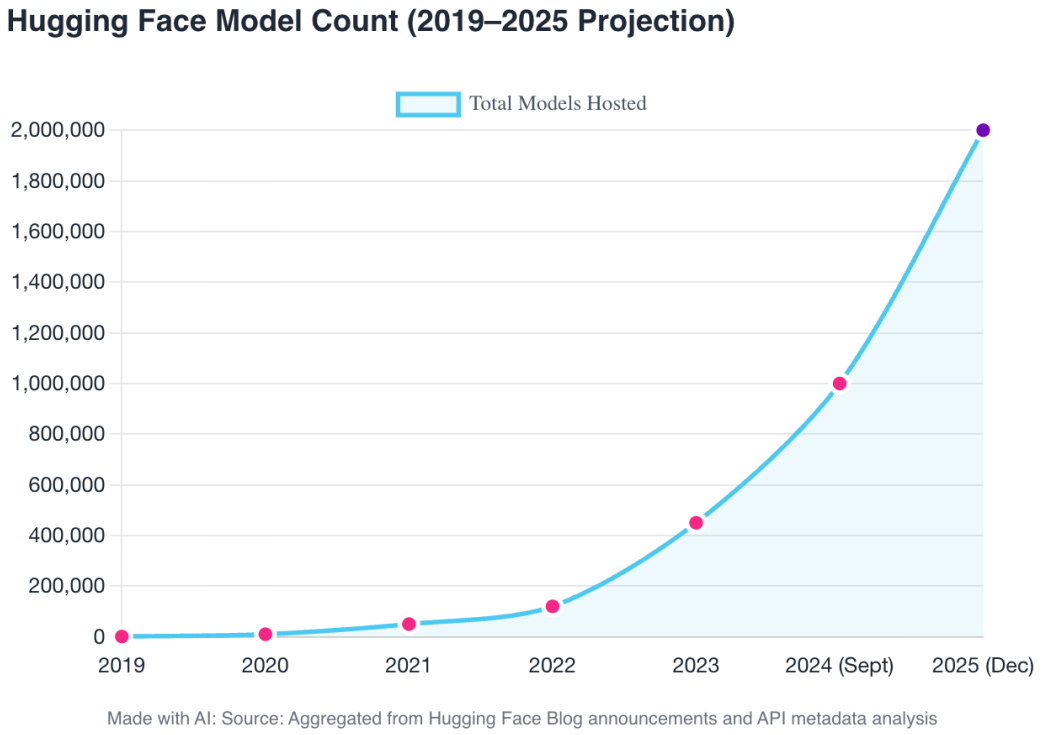
Problem Overview

- Problem: Too many Models
- Problem: Fine-tuning is costly
- Problem: Hardware & Architecture Selection

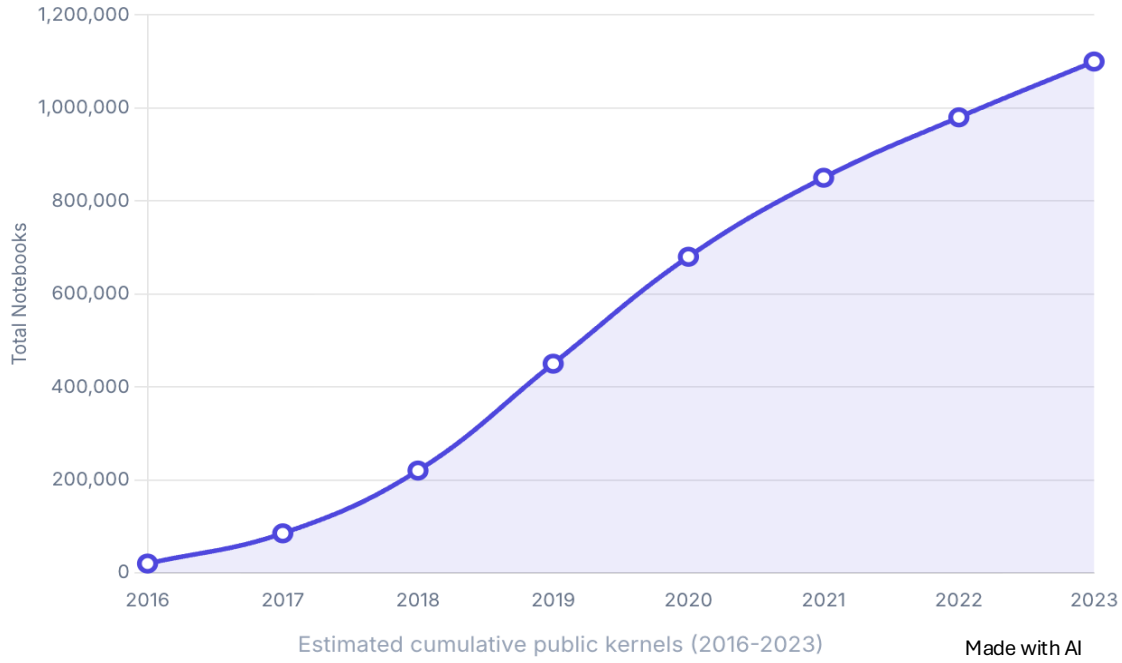
[1] Celeb on X (Twitter): https://x.com/calebfahlgren/status/1839311565524185347

Problem: Too many Models

- Hugging Face has 2,261,609 PTMs (2025)
- Model sizes usually ~ GB(s)



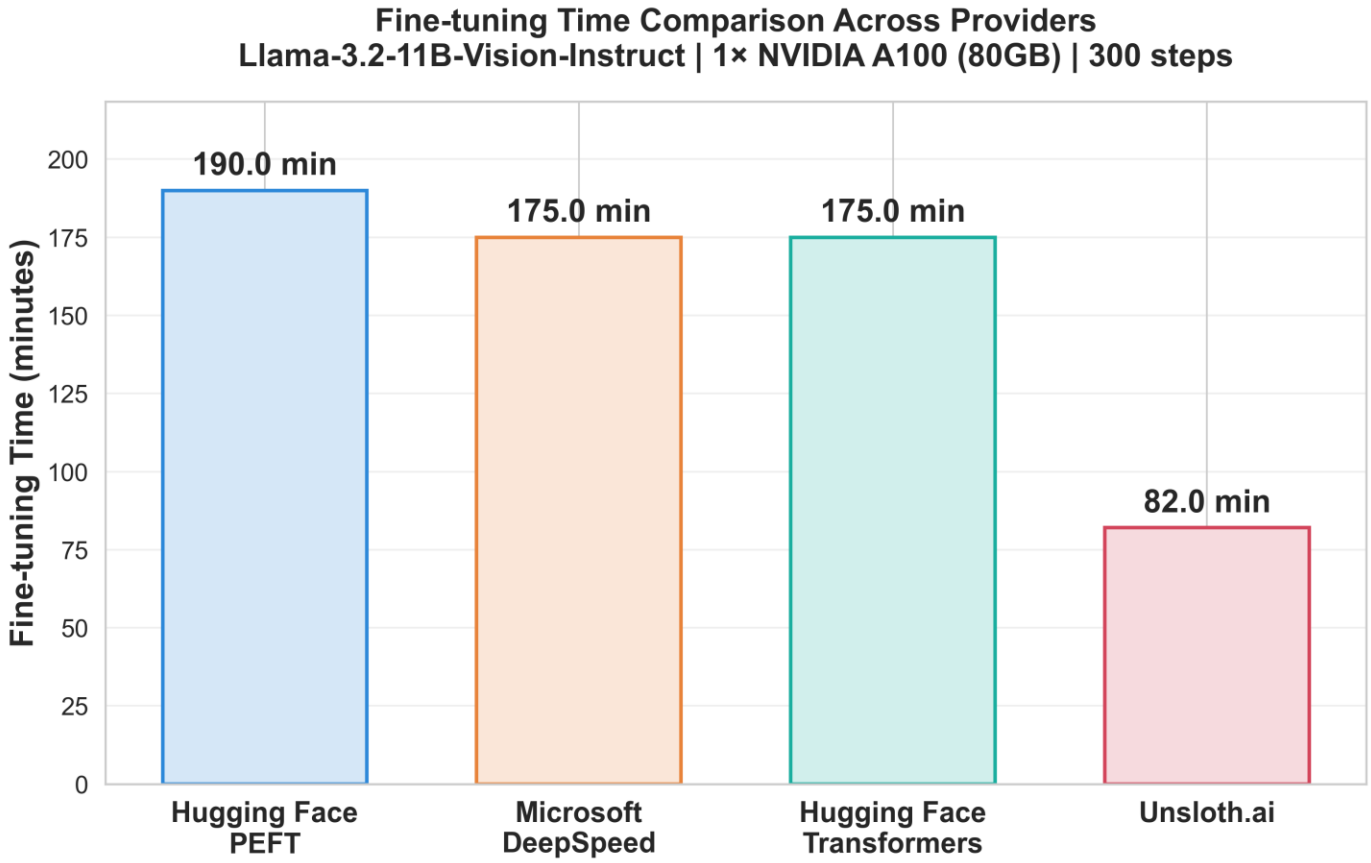
Hugging Face 🤗
[1]



Kaggle

Problem: Fine-tuning is costly

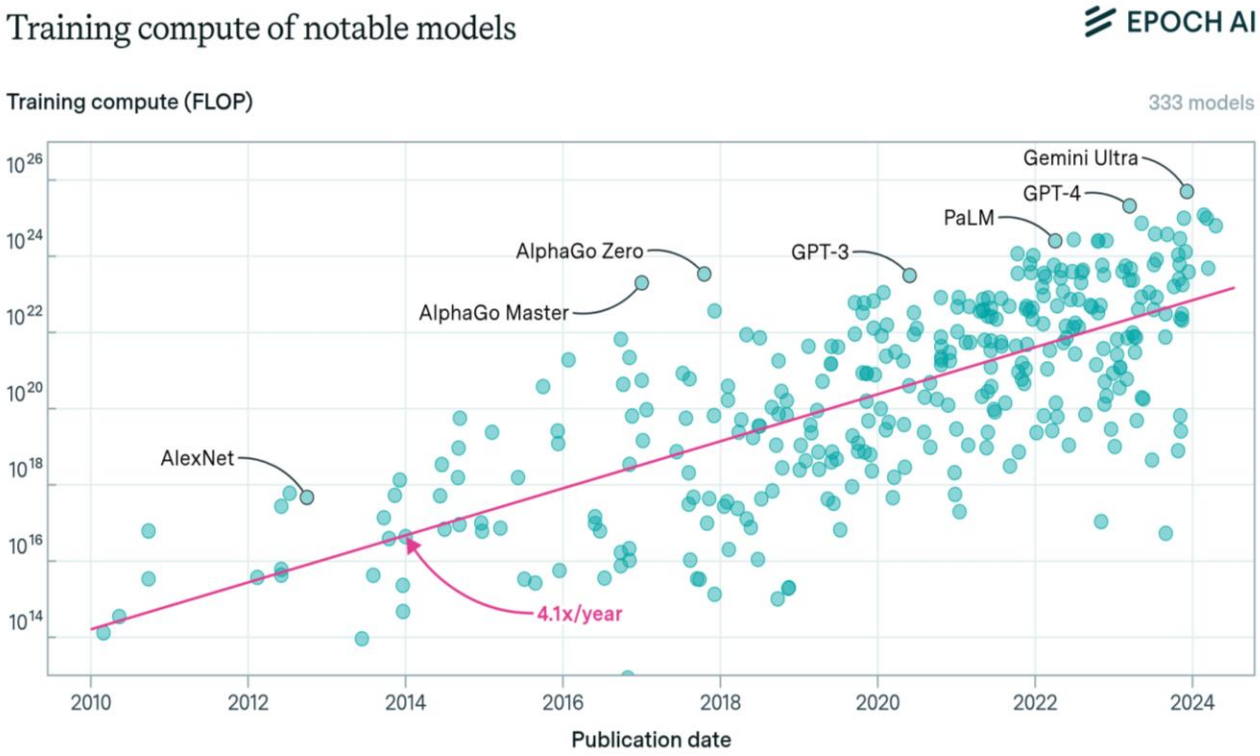
- Takes hours to fine-tune on a new dataset
  - Average ~ 100 minutes per model
  - For 100 models = 10000 minutes ~ 200 hours ~ 7 days
  - Additional setup time of pipeline for each unique model.

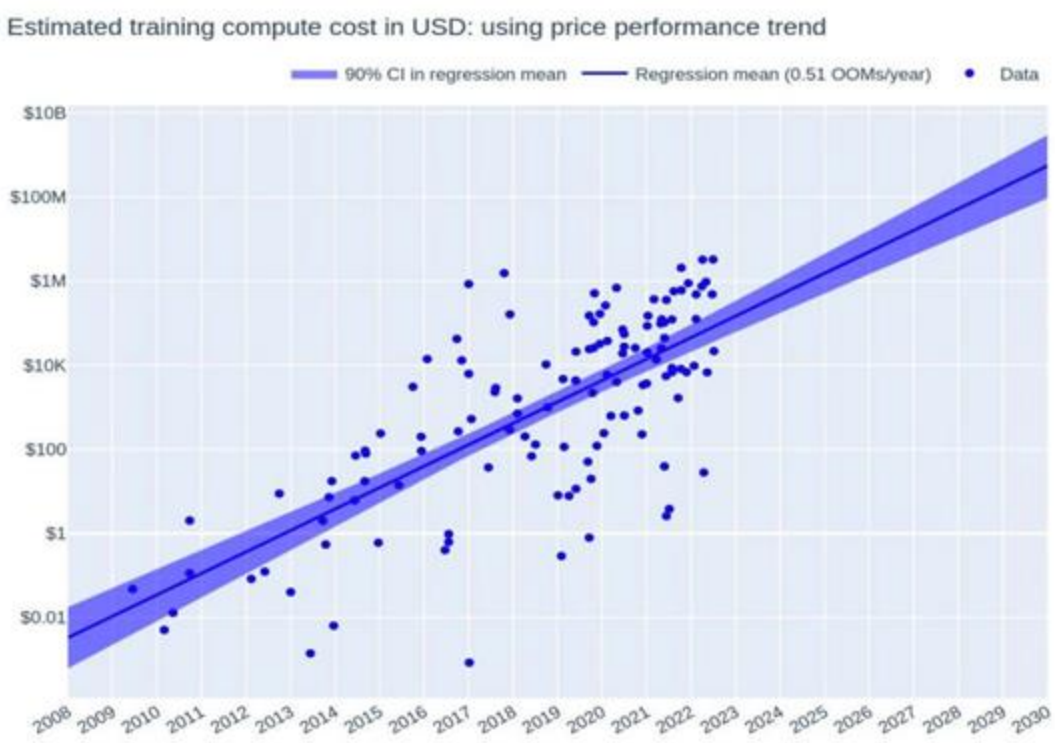


Epoch AI 2023: Training Compute of Frontier AI Models Grows by 4-5x per Year
Yalalov 2023: AI Model Training Costs Are Expected to Rise from $100 Million to $500 Million by 2030

Problem: Fine-tuning is costly

- Higher price ($) to fine-tune 100's of large models
- Also, applies to starting from scratch.



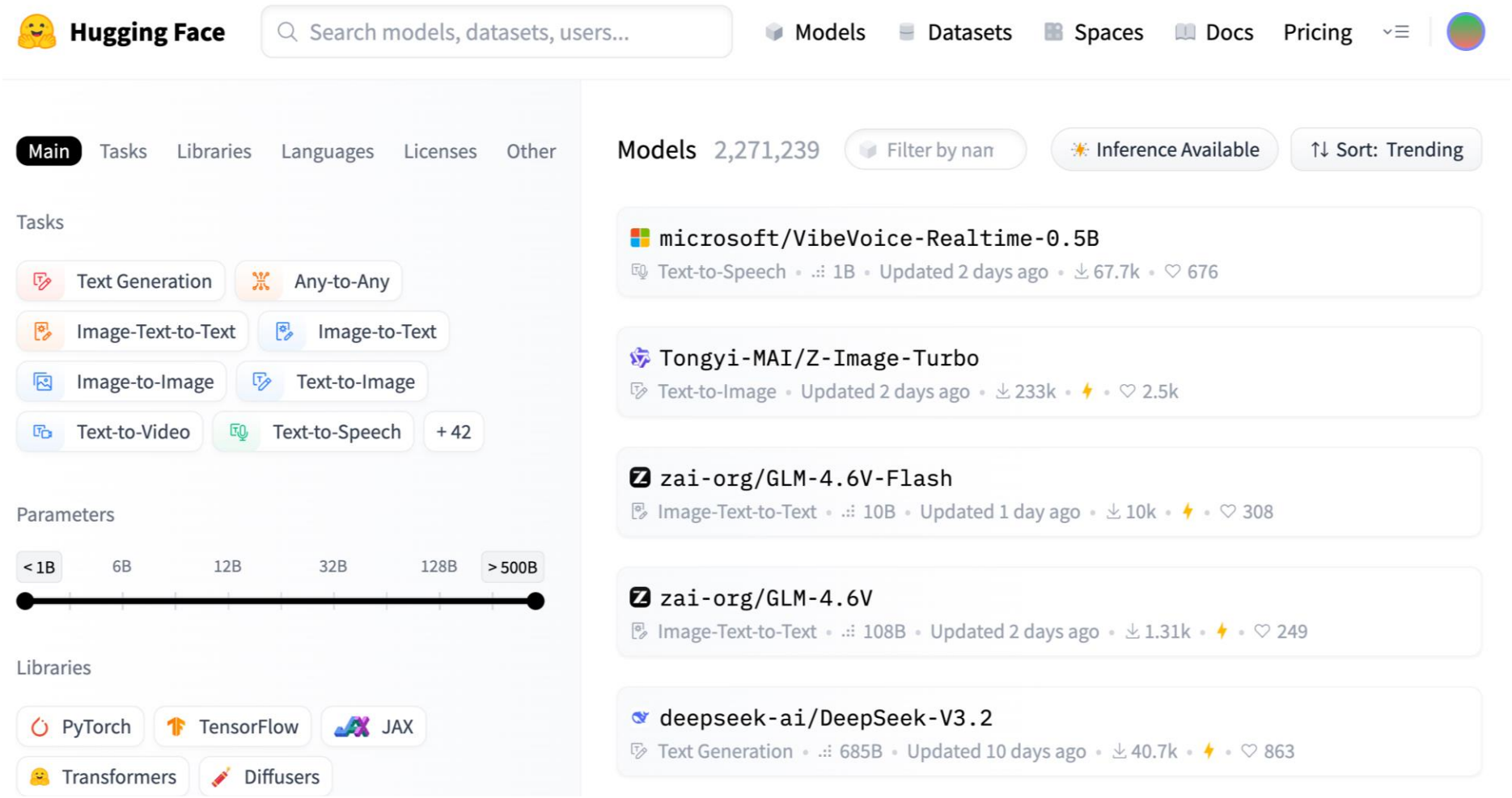


[1] Jiang, et. Al. 2023 : An Empirical Study of Pre-Trained Model Reuse in the Hugging Face Deep Learning Model Registry

Problem: Hardware & Architecture Selection

- Hardware Selection
  - Most approaches only consider accuracy of fine-tuned model will recommending.
  - No consideration for Hardware performance, Energy Footprint, etc.
- Architecture Selection
  - They search for candidate architectures “built for the problem that [they are] trying to solve”, browsing model registries or relevant papers [1].
  - This process inherently depends on comparing self-reported performances and is susceptible to non-reproducible results.

Problem: Hardware & Architecture Selection



3. Background and Related Works

PURDUE UNIVERSITY®

Background

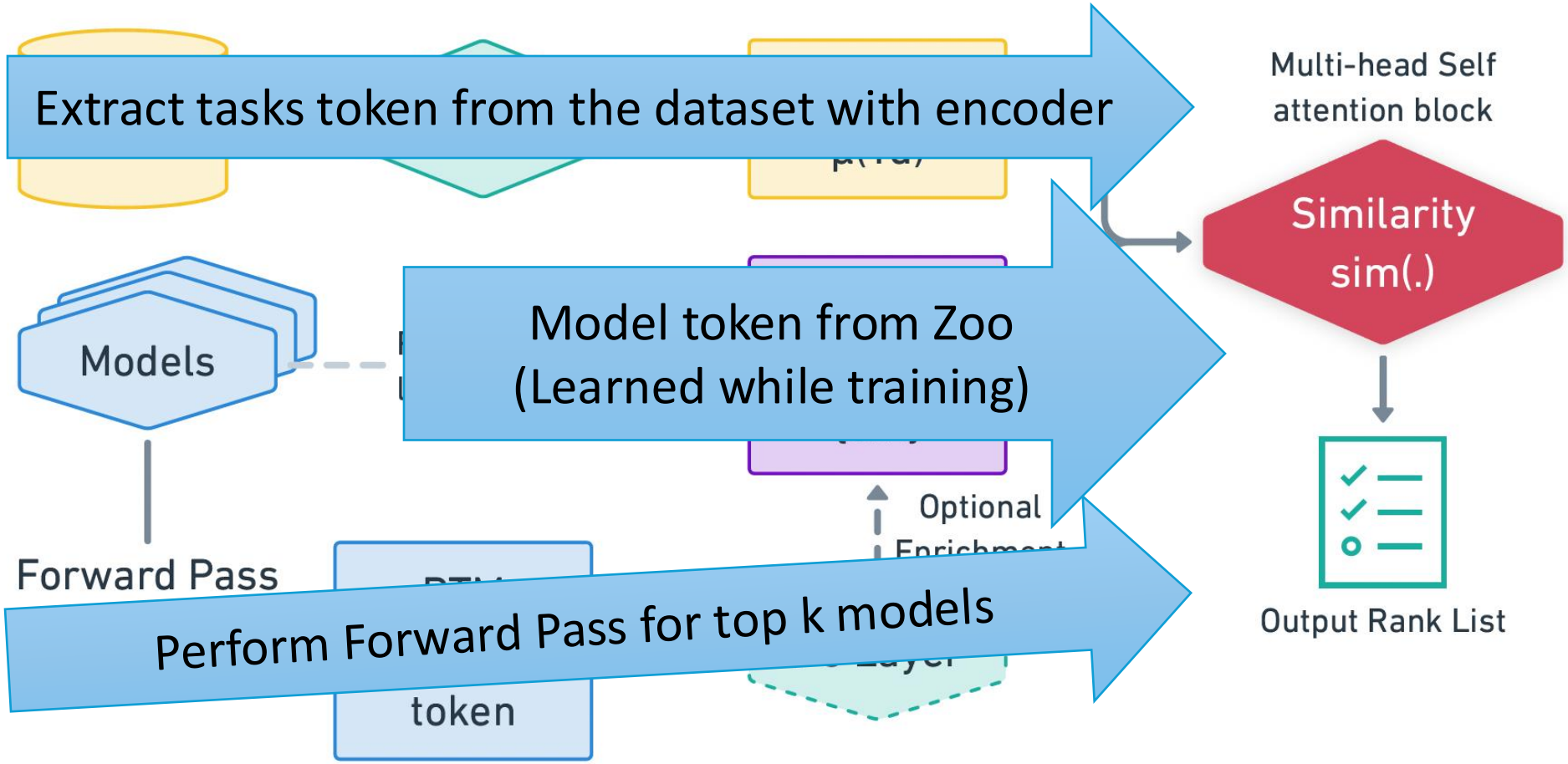
- Heuristics Based Approaches:
  - These methods introduce a novel heuristic or score to rank PTMs
  - They use some combination of forward pass, source dataset, and source labels to construct a score.
- Learning Based Approaches:
  - Employ ML techniques to eliminate the need for forward passes, on source PTMs.
  - Encodes both the PTM and target dataset into a latent space, and makes comparisons in the Latent space.

Background

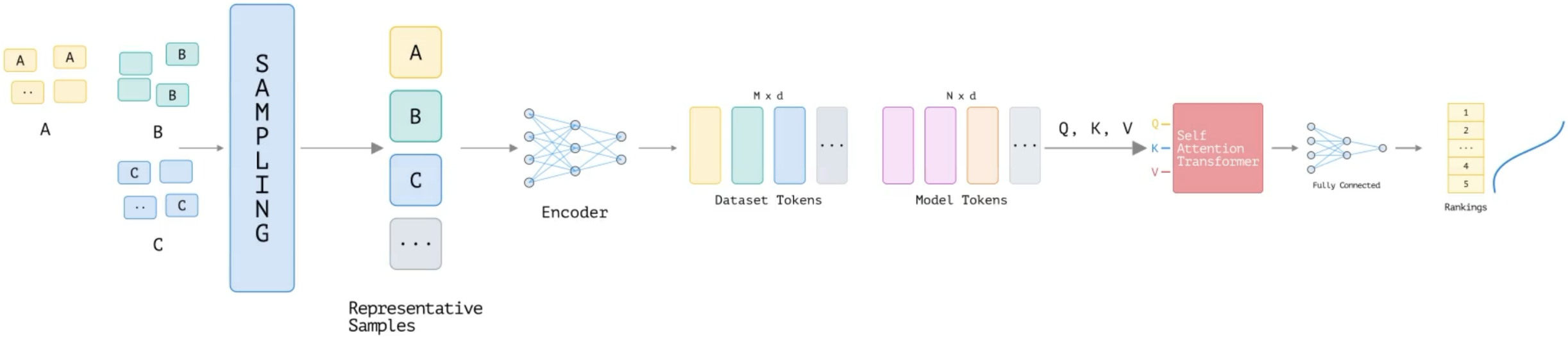
- Heuristics Based Approaches:
  - LEEP, N-LEEP, NCE, H-Score, PAC-Tran, LogME
- Learning Based Approaches:
  - Model Spider, EMMS, Fenne

Zhang, et al 2023: MODELSPIDER: Learning to Rank Pre-Trained Models Efficiently

SOTA - Model Spider



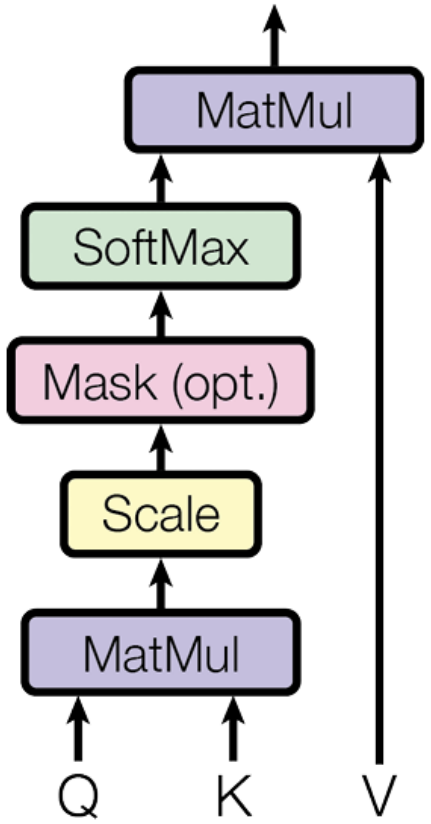
Model Spider Architecture



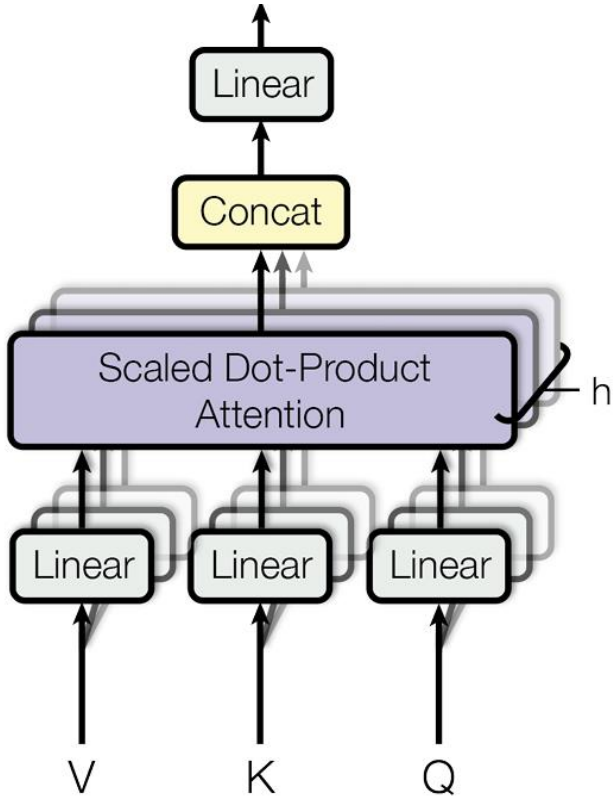
[1] Vaswani, et. Al. 2017 : Attention is All you Need

Cross Attention vs Self Attention

Scaled Dot Product Attention [1]

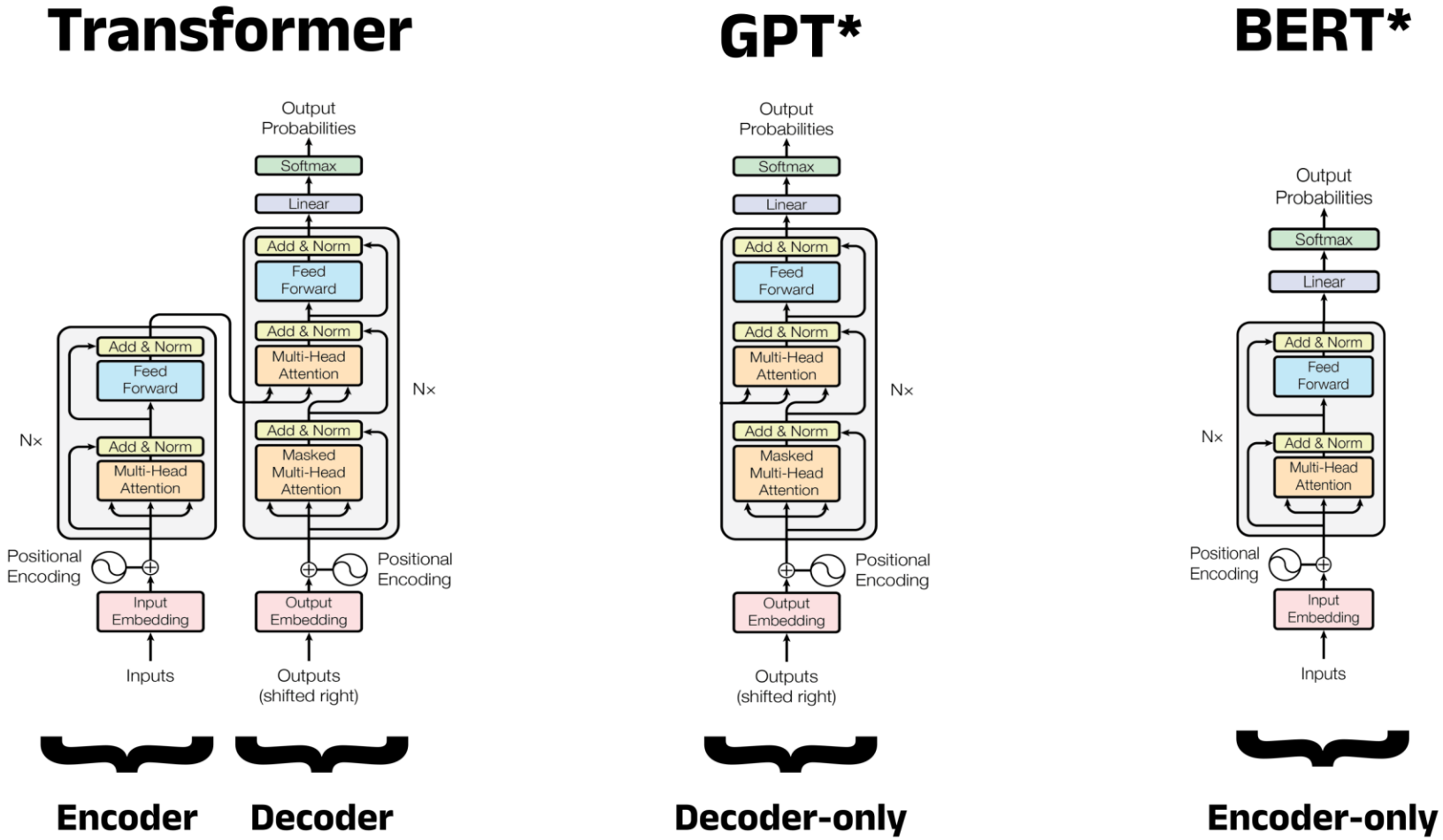


Multi-Head Attention Block [1]



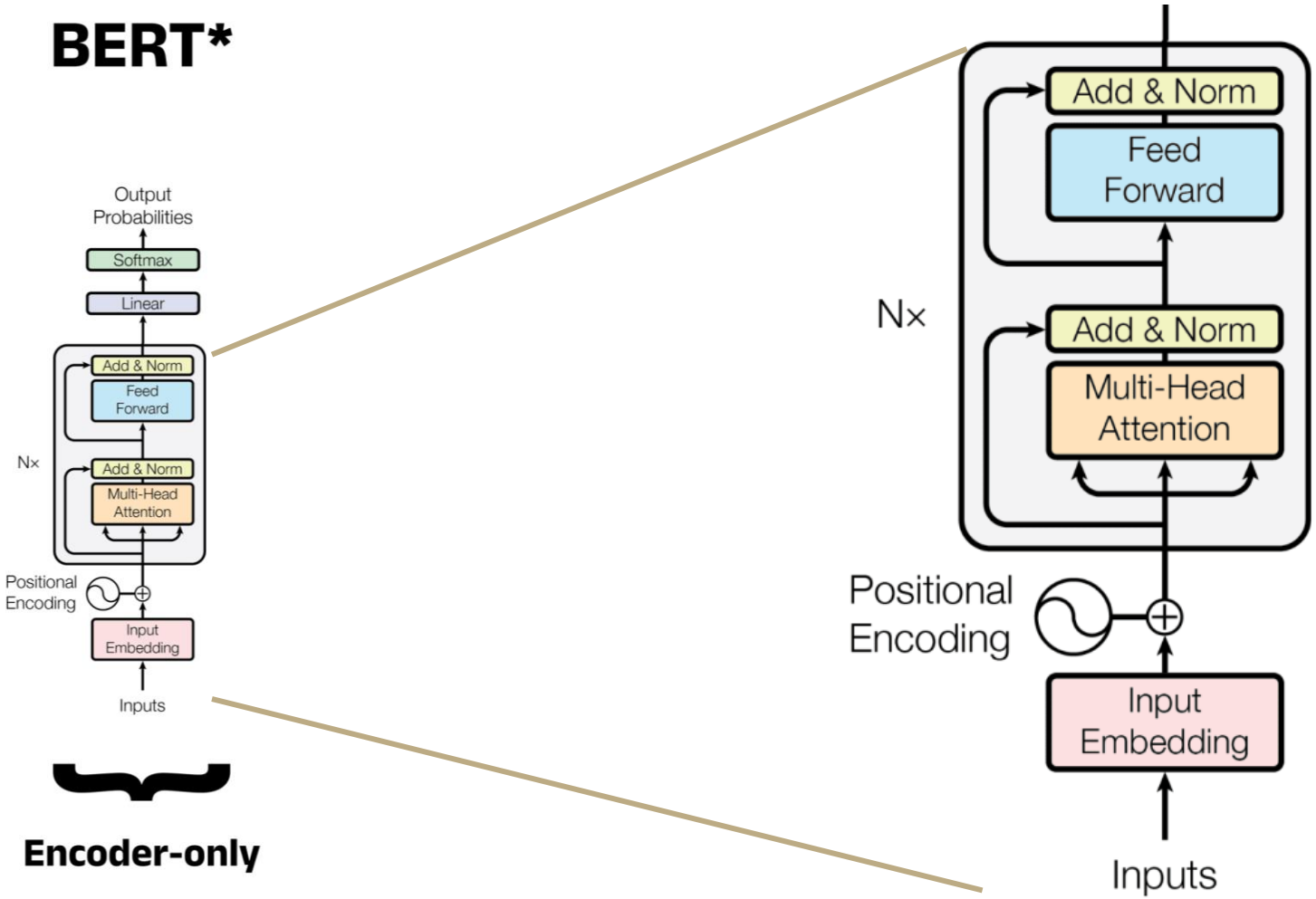
[1] Bradney Smith 2024: A Complete Guide to BERT with Code

Cross Attention vs Self Attention



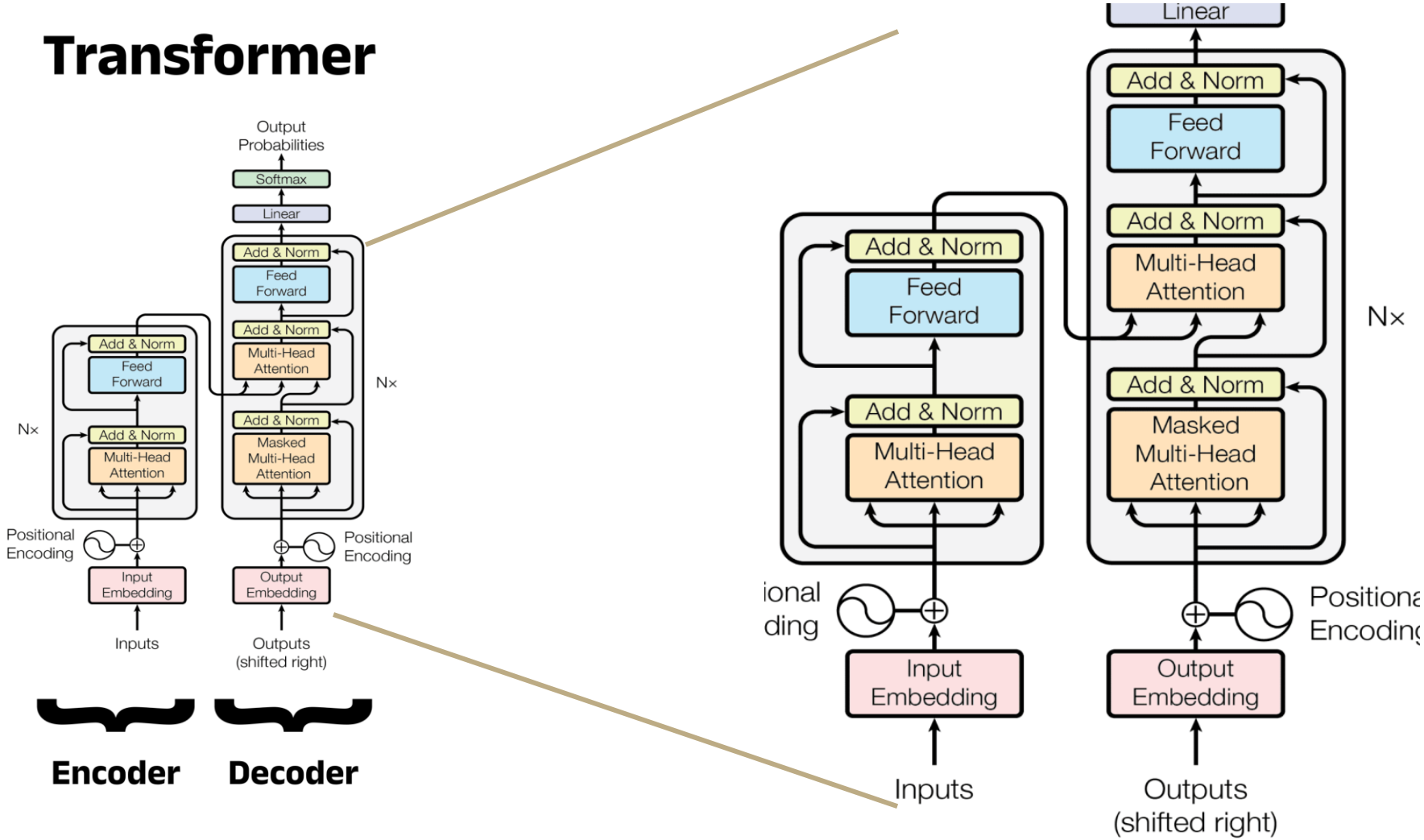
[1] Bradney Smith 2024: A Complete Guide to BERT with Code

Self Attention (Encoder only)



[1] Bradney Smith 2024: A Complete Guide to BERT with Code

Cross Attention (Encoder – Decoder)



4. Vision for PTM Recommendation

From the SERP4IoT Paper
“Recommending Pre-Trained Models for IoT Devices”
Parth Patil, et al.

PURDUE UNIVERSITY®

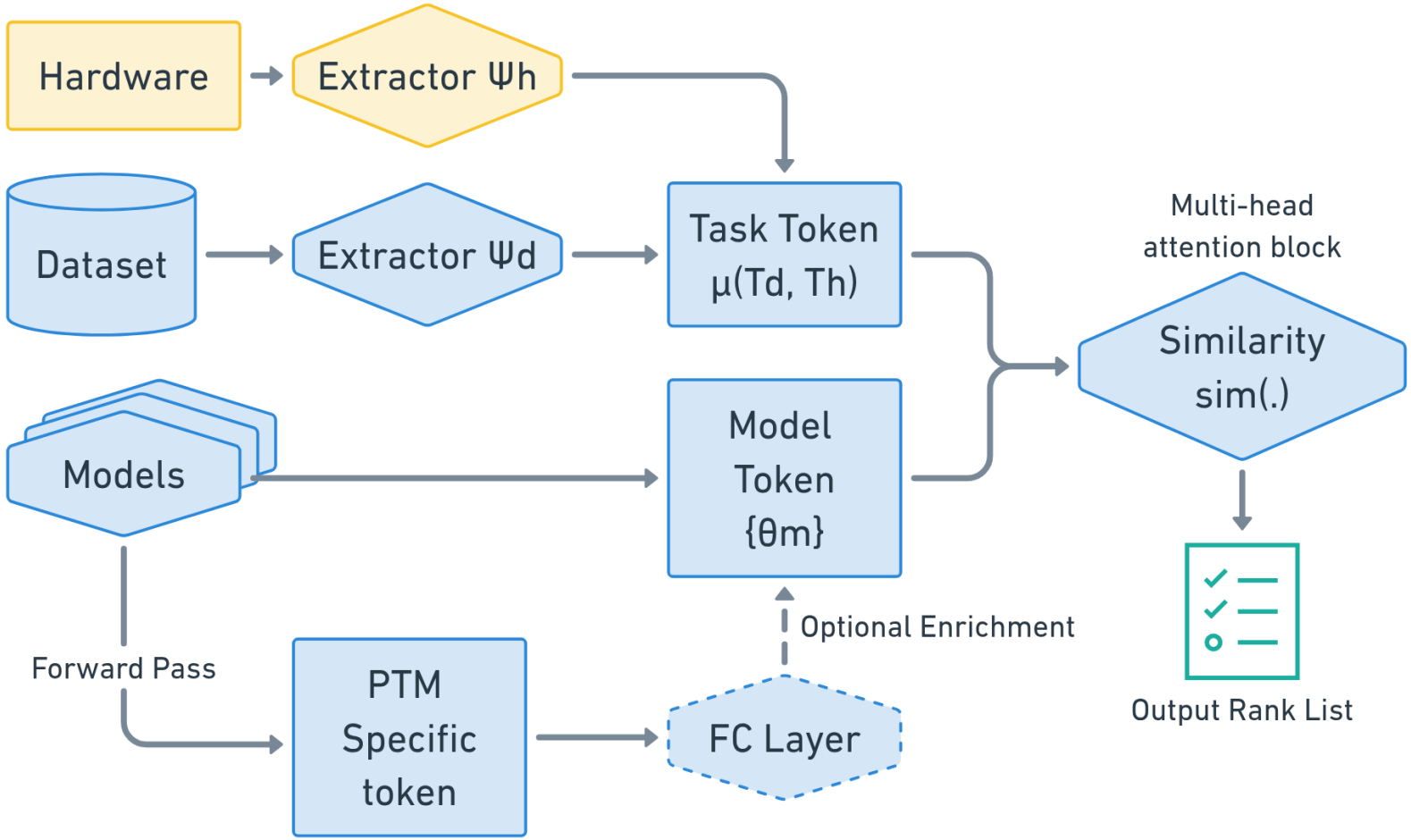
Gaps in the current landscape

- Gap 1: Lack of constraints beyond accuracy
  - Fail to consider the hardware performance implications when selecting models
  - No room to consider multiple constraints at the same time.
- Gap 2: Lack of Ground truth PTM rankings for devices
  - No empirical data is widely available to be used as ground truth.
  - Prevents finding patterns or correlations, or training ML.

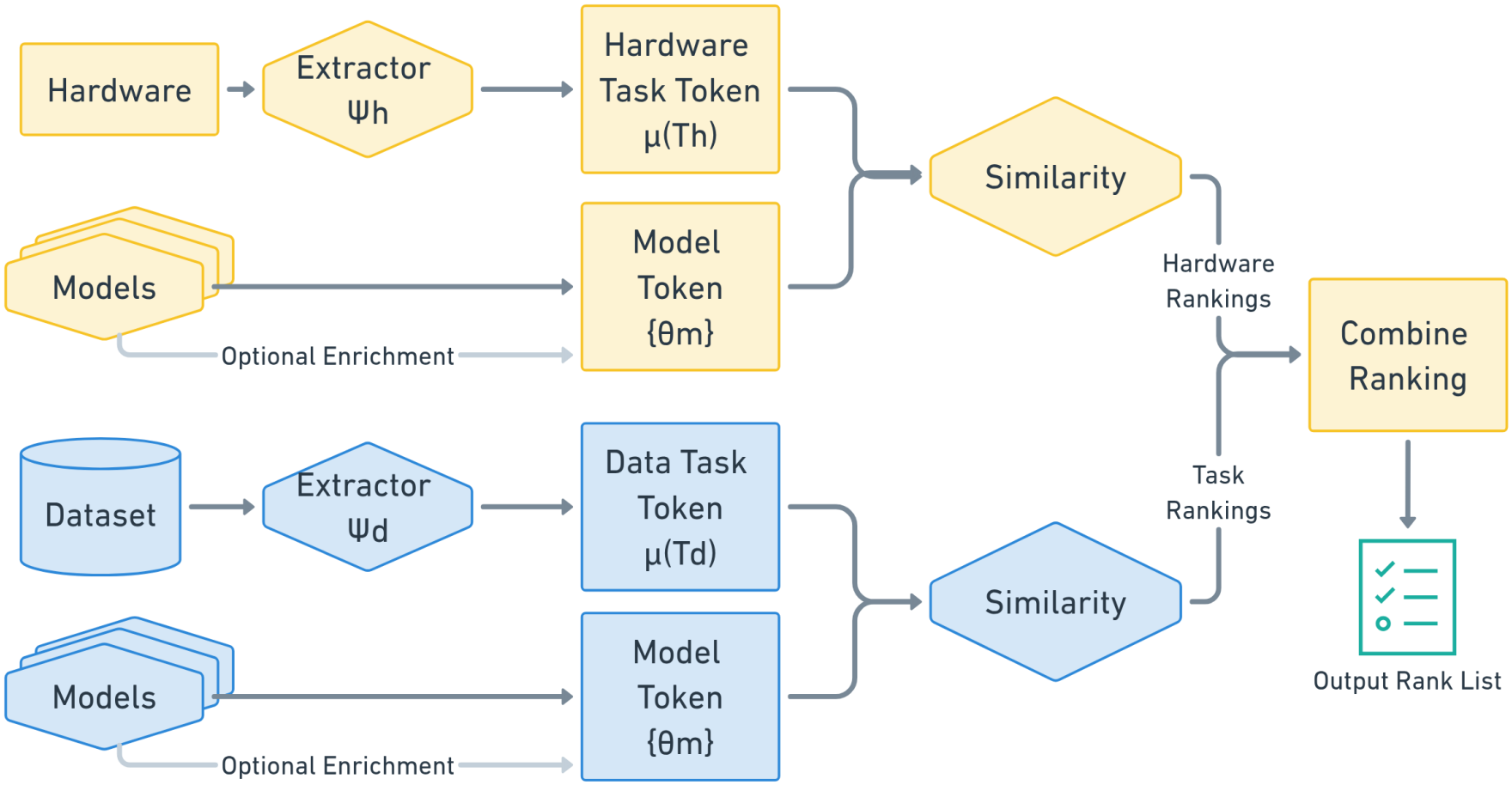
Hardware aware PTM Recommendation

- Proposed two unique methods to make the Model recommendation hardware aware.
- Approach 1: Model Spider ***Fusion***
- Approach 2: Model Spider ***Shadow***

Model Spider Fusion



Model Spider Shadow



Hardware aware PTM Recommendation

- Currently, there is a lack of datasets that track model performance along with hardware specifications.

- To address this shortfall, we put forth a roadmap to make a ground truth dataset by:
  - Define important metrics
  - Propose methodology to collect data.
  - Convert data into model rankings.

5. Cross-sight

A) Challenges and Key contributions

PURDUE UNIVERSITY®

Hypotheses

- H1: Using Cross attention would yield substantial improvement.
  - Since the model and dataset token belong to two different domains, this problem is better suited for Cross attention based architectures, compared to self attention based transformers.
- H2: Model Token can be generated using model weights and biases
  - Instead of relying on learnable model token (which limits the model zoo only to the models used while training), token can be generated with an encoder.

Key novel contributions

Key Improvements over Model Spider:

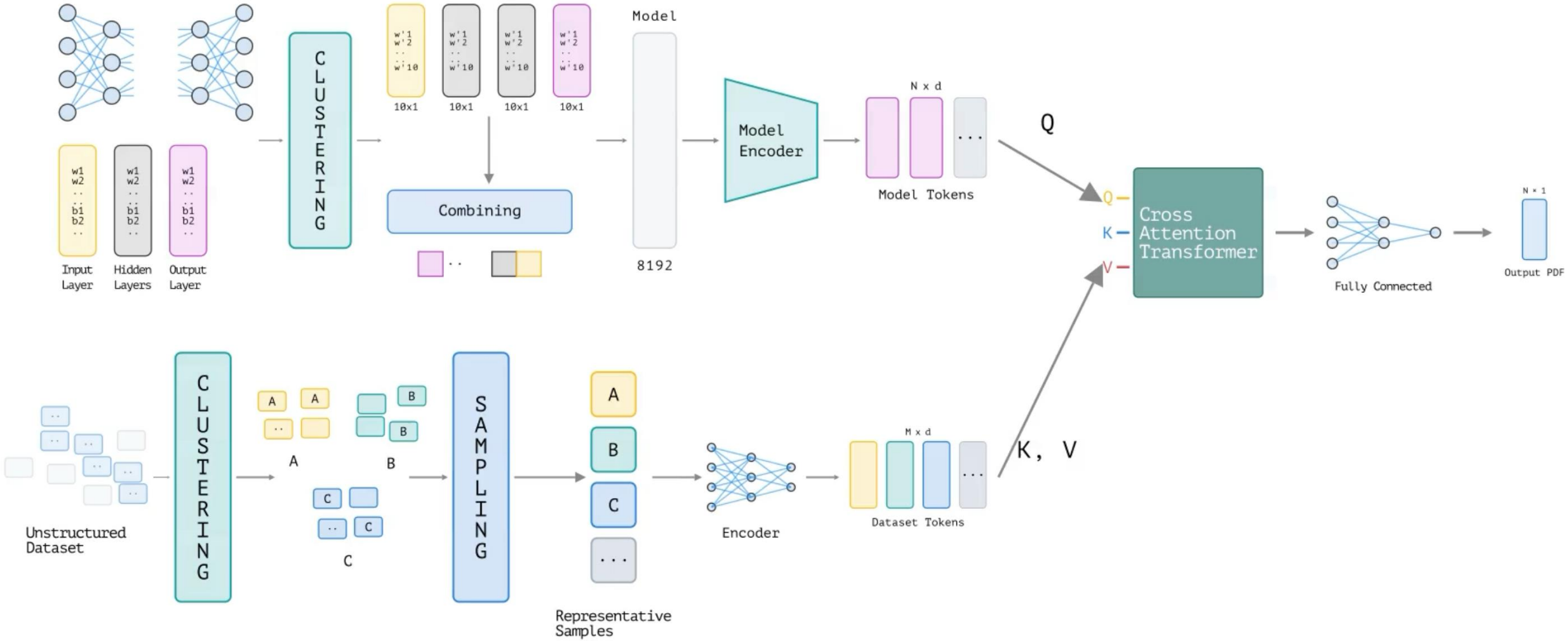
- Uses cross attention-based transformer (encoder-decoder type)
- Use a six-layered transformer block, instead of the single layer.
- Introduces a pipeline for PTM-to-Vector encoder, which enables an open model zoo rather than a closed one

5. Cross-sight

B) Walkthrough and experimental setup

PURDUE UNIVERSITY®

Cross-Sight Architecture



Preliminary Experimental Setup

- Goal: Evaluate cross-attention and model pipeline

For training:

- 11 Labeled dataset from three domains
  - Digits (MNIST, ..) , Fashion (Fashion MNIST, ..) , Generic (ImageNet-1k, ..)
- 31 Models in the Zoo.
  - Each model is trained on one of the dataset.
- Ground truth rankings are obtained by combining reported accuracy and similarity between datasets.

Preliminary Experimental Setup

$$\text{where } s_k = \begin{cases} a_{d_t,m_k} + 1 & \text{if } m_k \text{ trained on } d_t \\ a_{d_b,m_k} \cdot \sigma(d_t, d_b) & \text{if } m_k \text{ trained on } d_b \neq d_t \end{cases}$$

$$\boldsymbol{\pi} = \text{argsort}\left(\mathbf{s} + \boldsymbol{\epsilon}\right)_{desc}$$

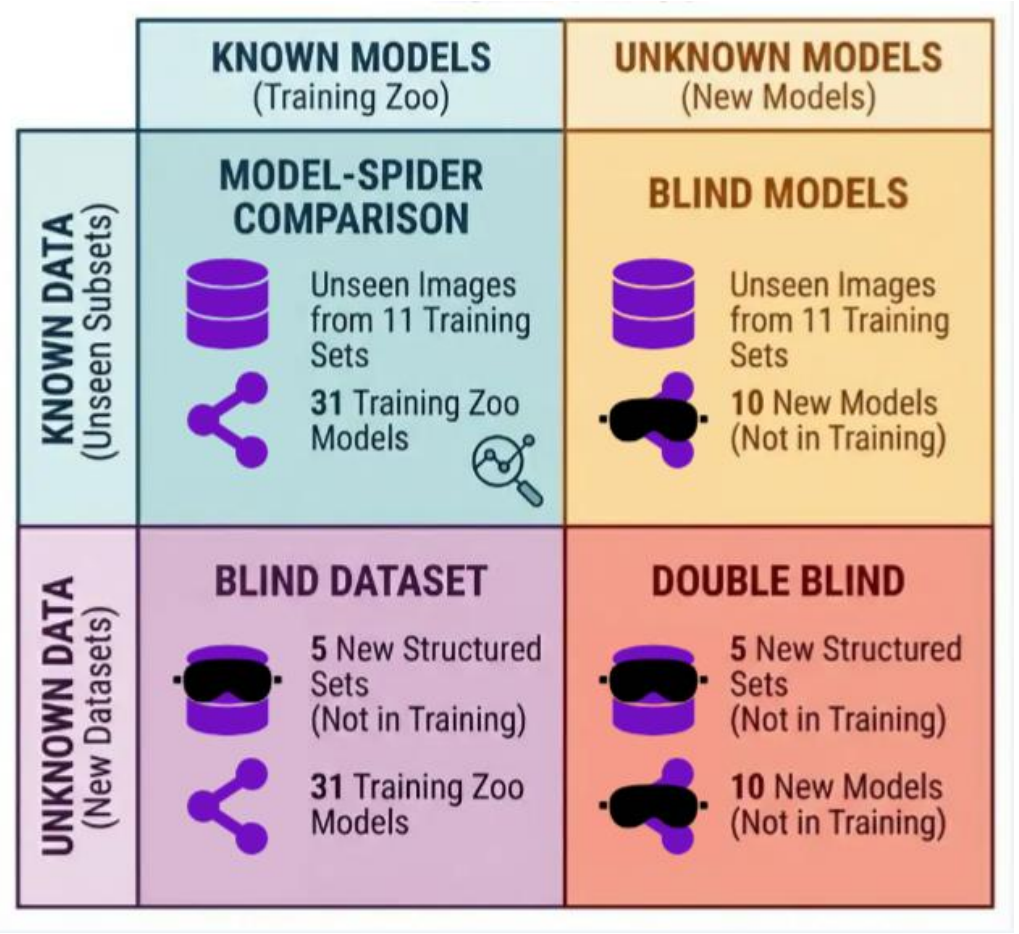
$$\ell_{\text{rank}}(\hat{\boldsymbol{t}}, \boldsymbol{t}) = \sum_{m=1}^{M} -\log\left(\frac{\exp\left(\hat{\boldsymbol{t}}_{\text{dsc}(m)}\right)}{\sum_{l=m}^{M}\exp\left(\hat{\boldsymbol{t}}_{\text{dsc}(l)}\right)}\right)$$

- **Ground Truth Ranking**
  - $a_{dt,mk}$ – Self reported score on the training dataset
  - $\sigma(d_t, d_b)$ – Similarity between training and target dataset (manually initialized)
  - S_k – ground truth score for k^{th} model.
  - π – Ground truth ranking computed with scores and small noise (ε) as a tie breaker.
- **Loss Function**
  - Goal: Make whole order of the predicted ranks close to the ground-truth rankings
  - Directly use the score outputs and use actual rank to arrange them in the sum, allowing the gradients to flow during training.

Preliminary Experimental Setup

For testing:

- Model-Spider Comparison:
  - Unseen Images from 11 Dataset used for training
  - 31 Models in the training Model Zoo
- Blind Dataset:
  - 5 Labeled dataset not used for training
  - 31 Models in the training Model Zoo
- Blind Models:
  - Unseen Images from 11 Dataset used for training
  - 10 Models not used for training
- Double Blind:
  - 5 Labeled dataset not used for training
  - 10 Models not used for training



5. Cross-sight

C) Results and comparisons

PURDUE UNIVERSITY®

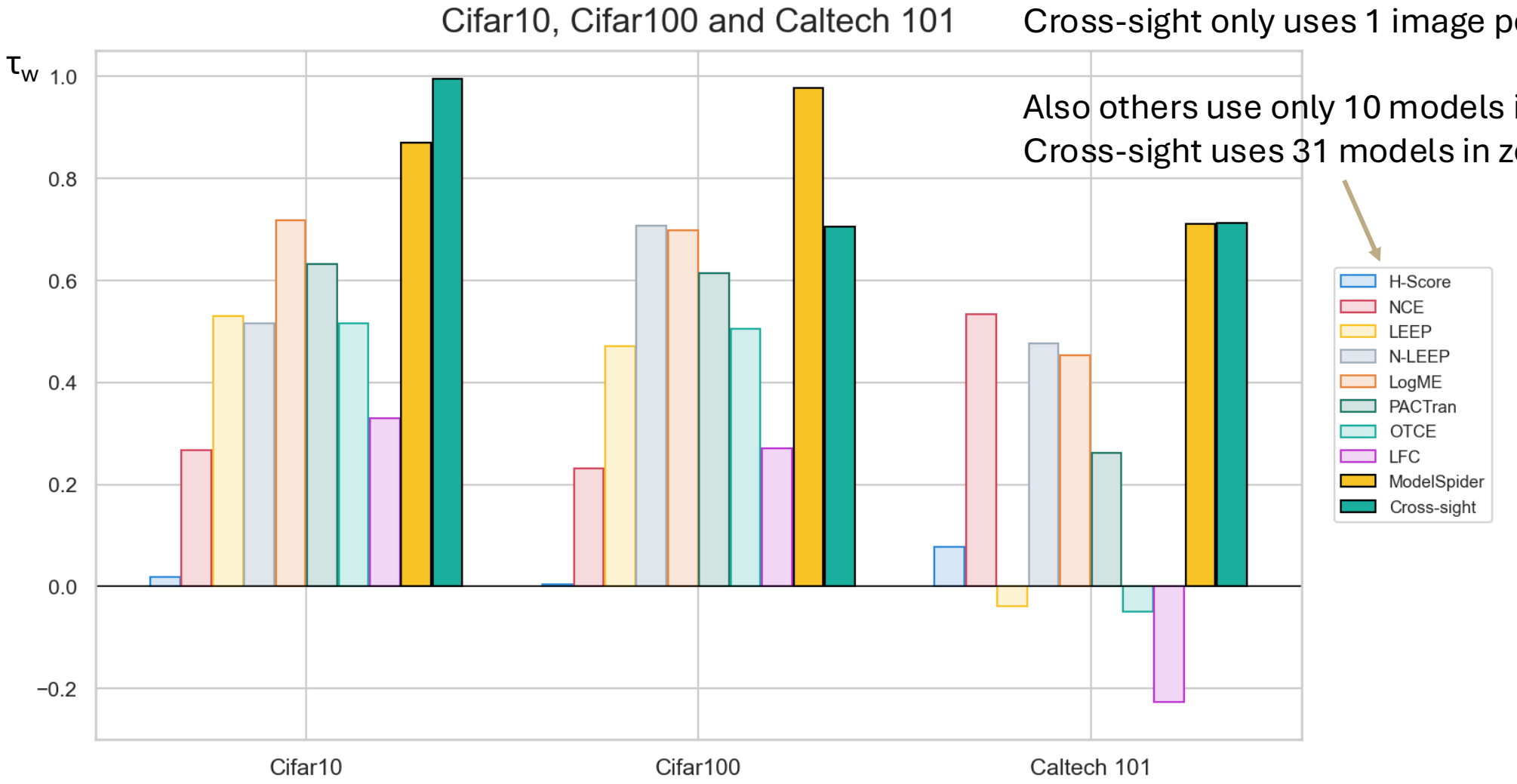
Kendall's Rank Correlation

- This is the Metric used for comparing the ranking results
- Kendall's Tau (τ) is a statistic used to measure the ordinal association between two measured quantities
- The Metric Range: -1 to 1
  - Kendall's Tau is normalized. It counts the number of pairs that agree in order **(Concordant) (+1)** versus those that disagree **(Discordant) (-1).**
- Weighted Tau (τ_w):
  - This variant of Kendall's Tau is used to compare results.
  - applies a hyperbolic weighting function. Differences at the top of the list (Rank 0, 1, 2) contribute significantly to the score, while differences deep in the list have negligible impact.

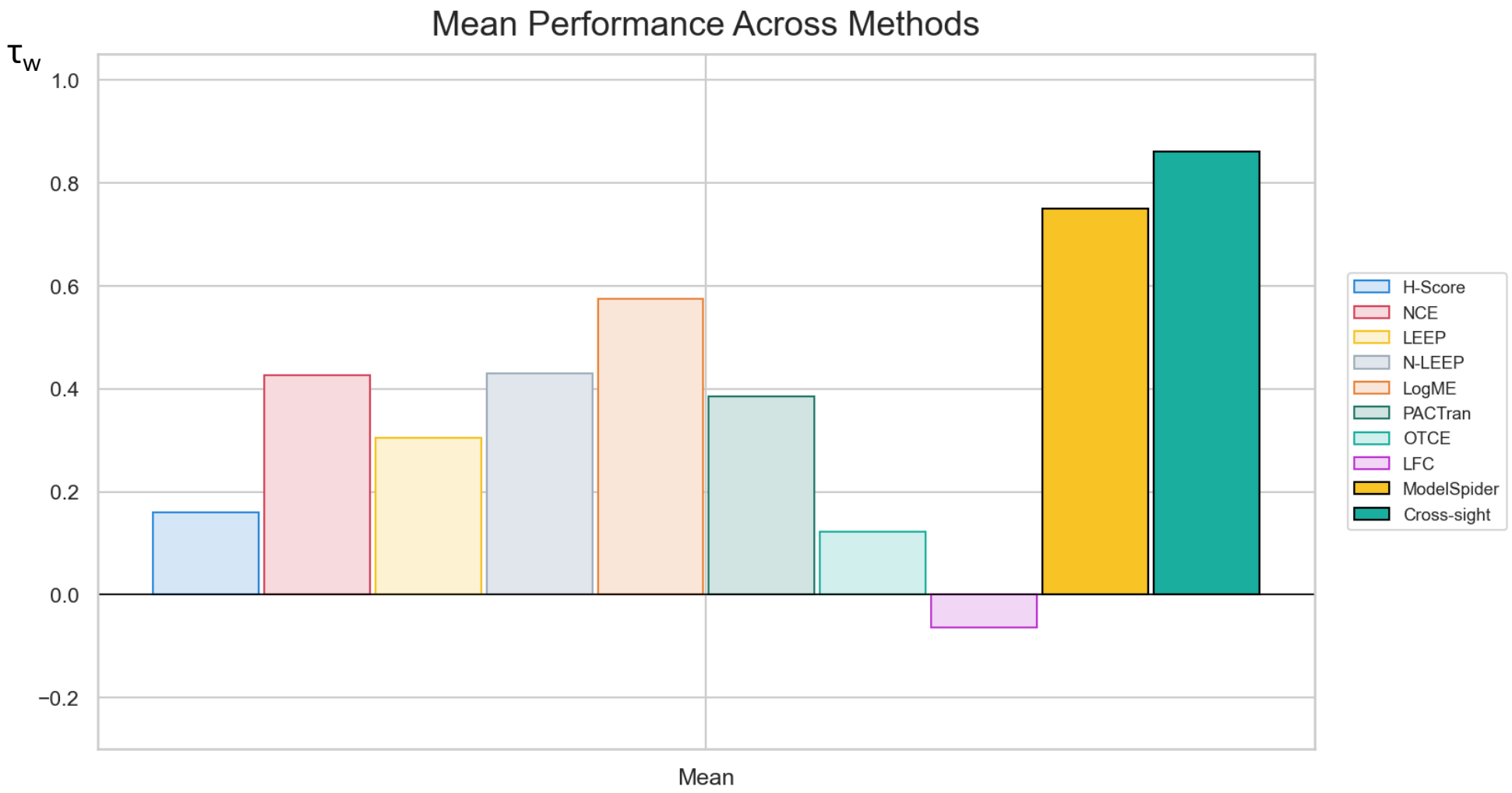
Model-Spider Comparison

All approaches expect Cross-Sight
Uses 10 images per class per Dataset
Cross-sight only uses 1 image per class

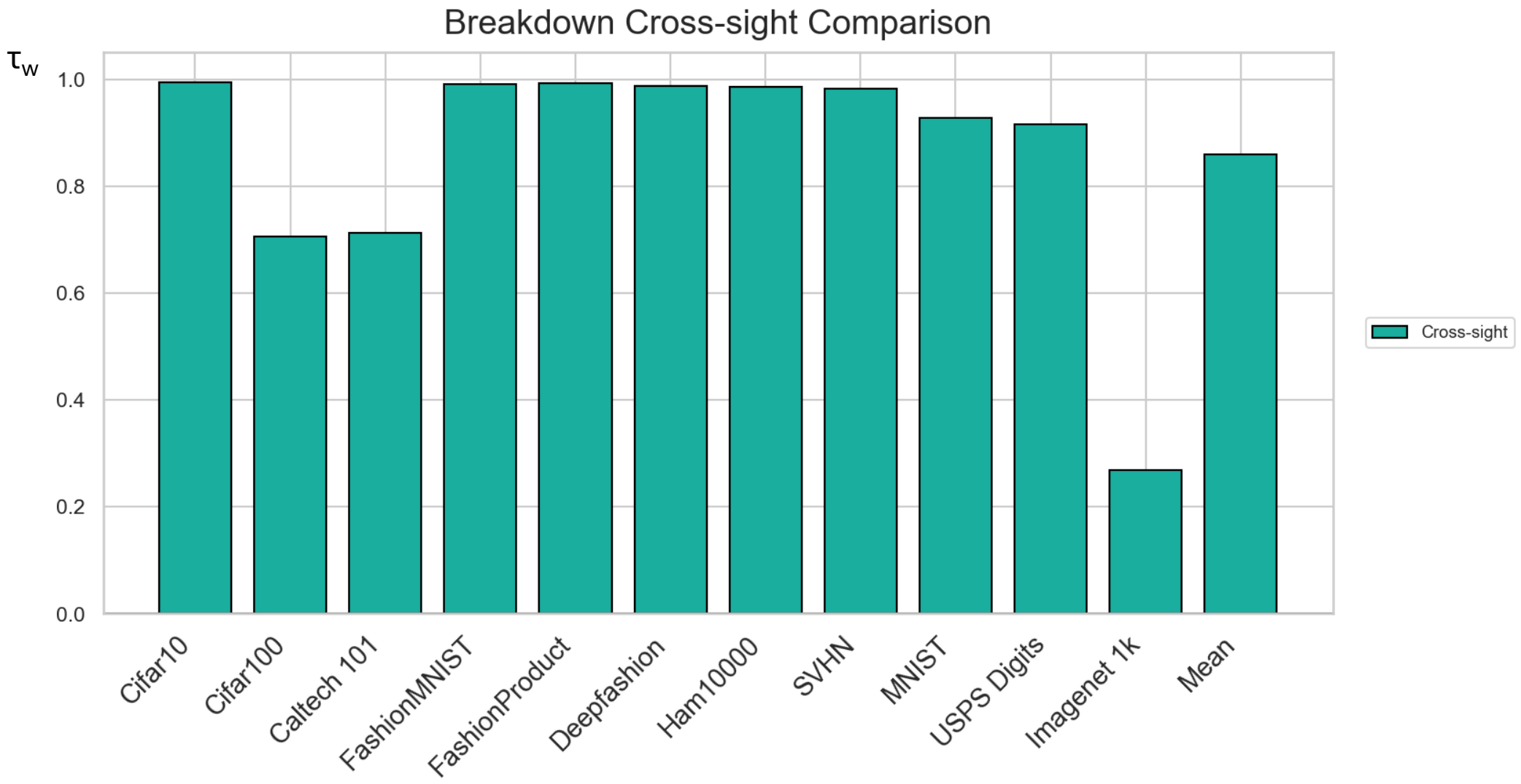
Also others use only 10 models in their zoo
Cross-sight uses 31 models in zoo



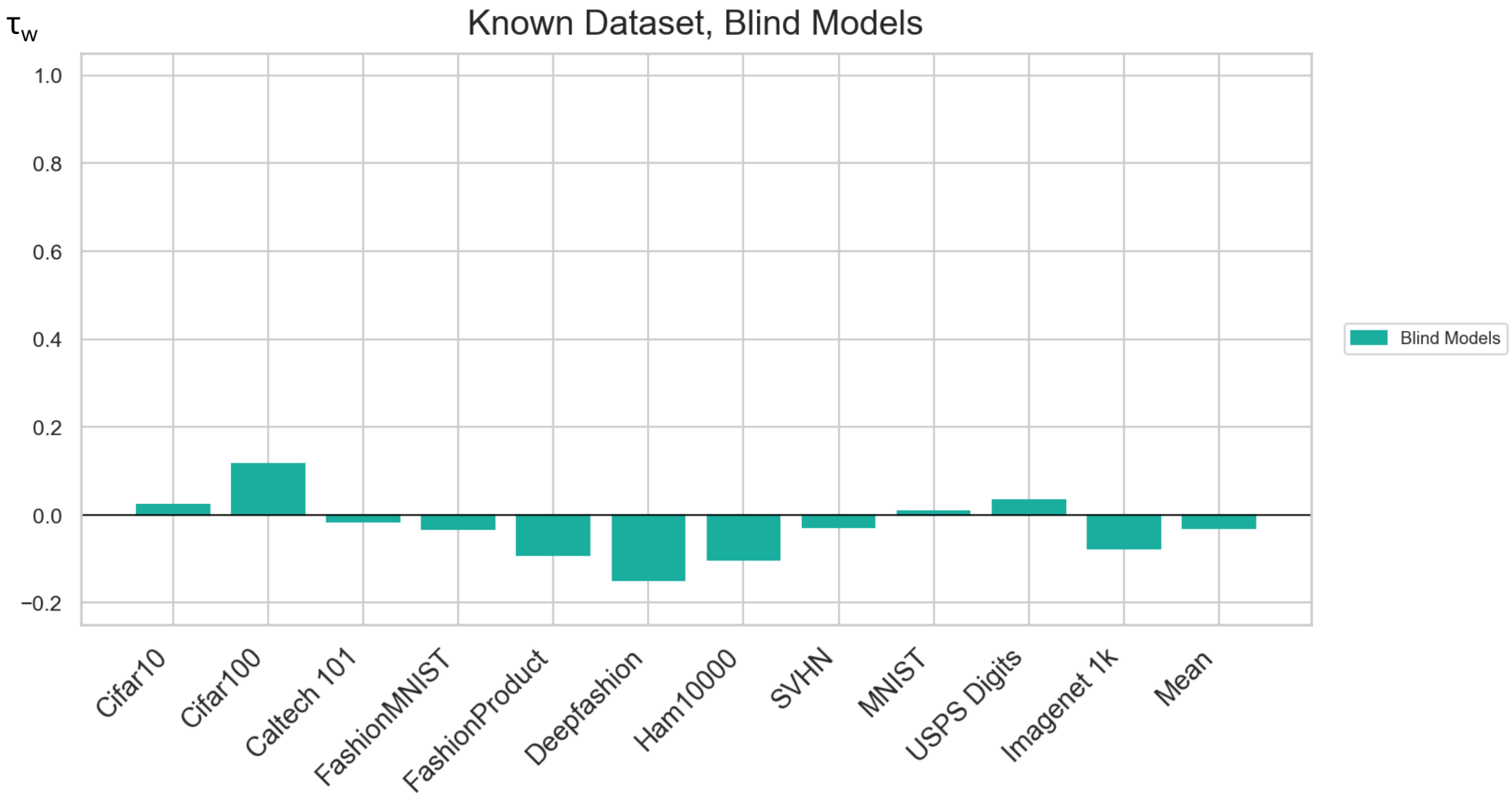
Model-Spider Comparison



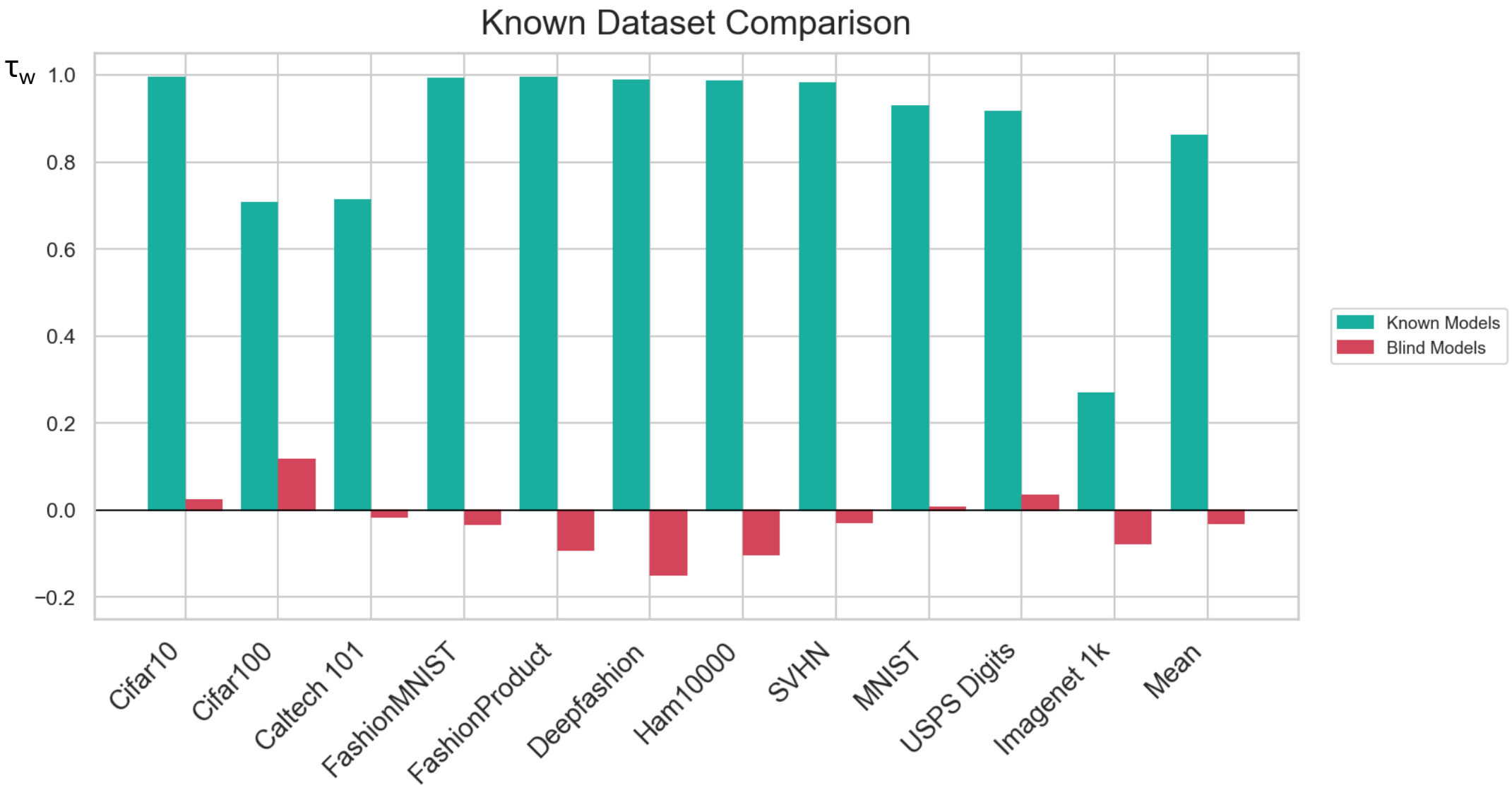
Model-Spider Comparison



Blind Models

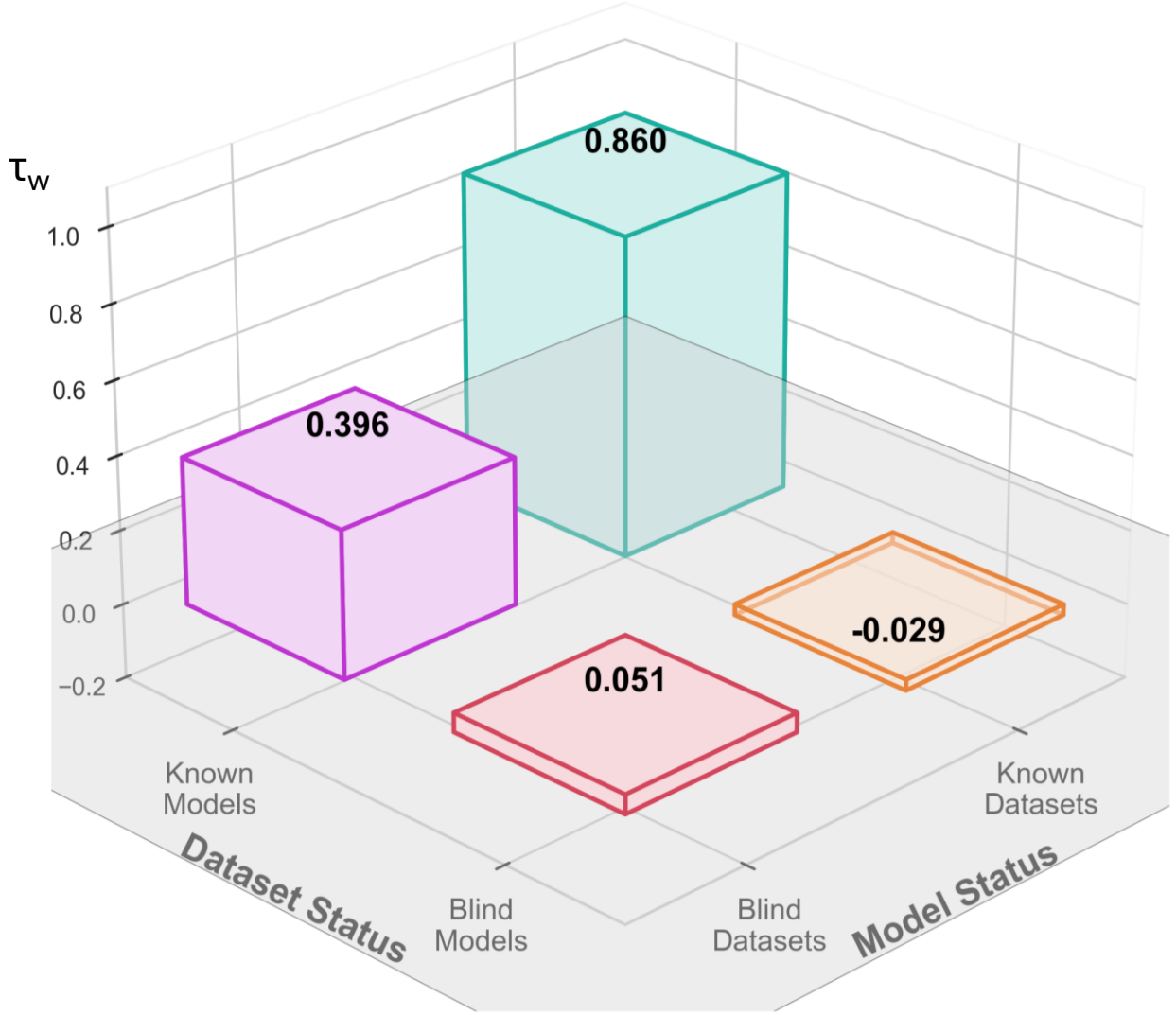


Known Datasets



Blind Dataset

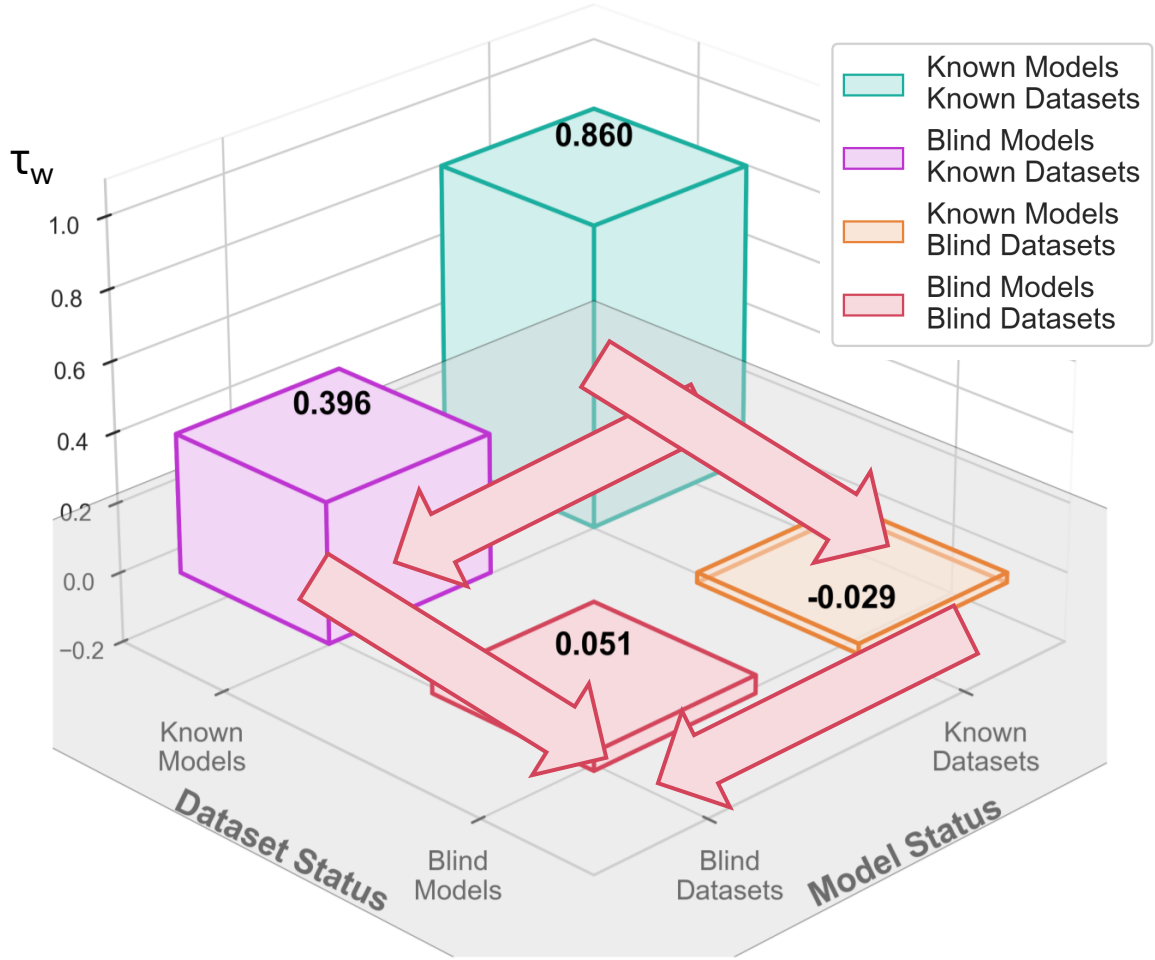
Overall Performance of Cross-Sight



Note: All the results are based on the estimated ground truth ranking, which rely on reported accuracies

Conclusions

- H1:
  - Using cross attention did increase the performance
  - Even performs decent on unseen dataset.
- H2:
  - Able to use the generated model token as a replacement for learnable token.
  - But the pipeline doesn't generalize for unseen models.



6. Limitations and Future Work

PURDUE UNIVERSITY®

Limitations and Future Work

- Improvements to the Model-To-Vector pipeline
  - Introduce encoding model structure in model token
  - Incorporate Model Card info and Meta-Data
  - Ablate the encoder (not just autoencoder)
- End to End (Co-Learning):
  - Train both Encoder and Transformer together, so the ranking loss can flow to train the model encoder as well

Limitations and Future Work

- Introduce additional ranking dimensions
  - Introduce proposed pipelines Model Spider Fusion and Model Spider Shadow to study ranking based on different constraints.
- Train on Ground-Truth Rankings: Create Dataset
  - Collect actual fine-tune performance
  - Collect performance information for specific hardware
- Analyze how the Model lineage affect generalization
  - If a PTM share same architecture with one in Model Zoo, can we rank it correctly?

7. Conclusion and Discussion

PURDUE UNIVERSITY®

Conclusion and Discussion

- This work put forth a broad vision for the future of PTM recommendation. The preliminary experiments shows significant promise for the proposed Cross-sight approach.

- Questions?

Thank you

PURDUE UNIVERSITY®

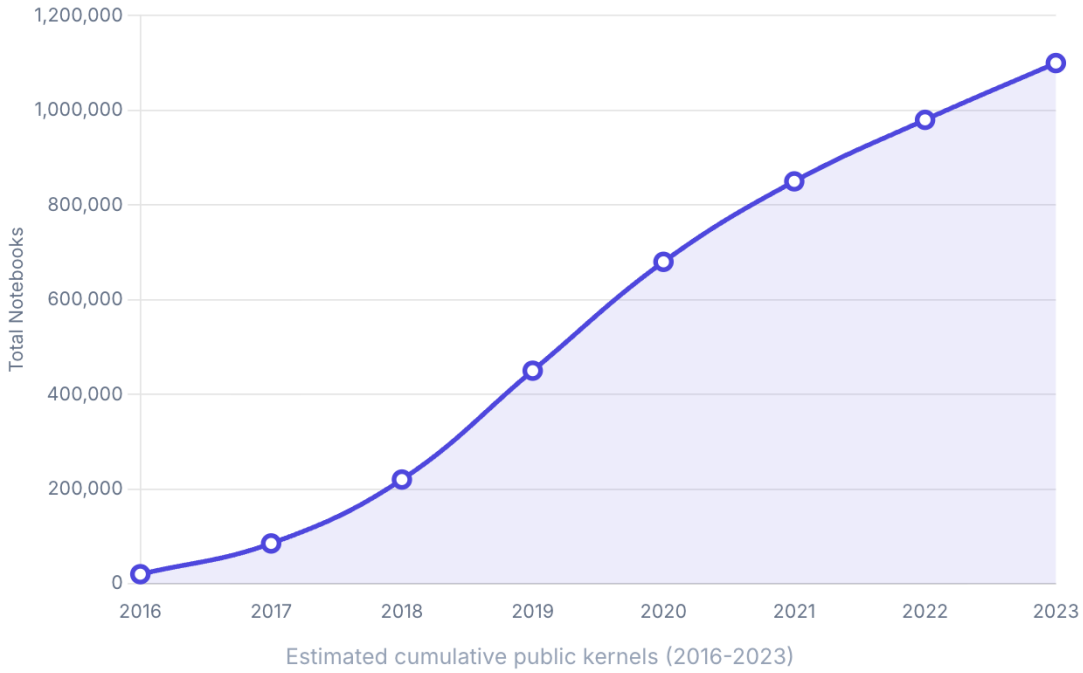
Appendix

PURDUE UNIVERSITY®

1. Number of Kaggle notebooks

The data points for the "Cumulative Public Kernels (Est.)"are estimated based on analysis of the official Meta Kaggle Dataset and high-level growth figures released in platform updates.

Year	Estimated Cumulative Public Kernels (Used in Chart)	Primary Data Source Context
2016	20,000	Early growth figures often referenced in Kaggle's blog posts following major user milestones.
2017	85,000	Reflects acceleration after the introduction of free GPU/TPU environments.
2018	220,000	Post-acquisition growth and maturity of the Kernels feature (then "Scripts").
2019	450,000	Strong community engagement and adoption of advanced frameworks (TensorFlow/PyTorch).
2020	680,000	Accelerated growth due to global remote work trends and increased interest in data science.
2021	850,000	Steady platform growth.
2022	980,000	Approaching the 1 Million mark.
2023	1,100,000+	Official milestone confirmed by Kaggle platform announcements and tracking via the public dataset.



Example Heuristic-Based: LogME

- Computes the Logarithm of Maximum Evidence (LogME) of labels given features extracted by each PTM
- Workflow:
  - Extract features from the target data via each PTM (forward pass)
  - Compute LogME score per model
  - Rank models by descending LogME for selection

Background

- Compares current model recommendation approaches and their requirements.
- Model Spider is notably efficient, not requiring a forward pass on the target dataset
- Though none address hardware constraints

	Recommendation Approach	Requires			Hardware Aware
		Forward Pass	*Source Labels*	*Target Labels*	
Heuristic - based	NCE [6]	No	-	-	✗
	H-Score [7]	No	-	-	✗
	OTCE [17]	-	-	-	✗
	LEEP [8]	-	No	-	✗
	N-LEEP [9]	-	No	-	✗
	LogME [10]	-	-	No	✗
	PACTran [11]	-	-	-	✗
	GBC [12]	-	No	-	✗
	LFC [13]	-	No	-	✗
Learning	Model Spider [14]	No	No	No	✗
	EMMS [15]	No	No	-	✗
	Fennec [16]	No	No	-	✗
Ours	MS Fusion	No	No	No	✓
	MS Shadow	No	No	No	✓

Hardware aware PTM Recommendation

	Metric	Description	Sample
Hardware	Execution Time	Time for one forward pass	90 ms
	Memory Utilization	Memory used during execution	3 GB
	Power Consumption	Total energy consumed	5 W
	CPU Temperature	Temperature of the CPU	75°C
	Carbon Footprint [20]	Environmental impact	10.9 units
Model	Accuracy	% correct predictions	92%
	Precision	identified / predicted positives	88%
	Recall (Sensitivity)	identified / actual positives	85%
	F1 Score	Harmonic mean of last two	86%

Hardware aware PTM Recommendation

$$\max_{\alpha \in A} f(\alpha) \cdot \sum_i \left(\frac{\mathrm{HW}_i(\alpha)}{T_i} \right)^{w_i}$$

Proposed approach to combine rankings for **Shadow** Approach

ω_i - weight assigned to a metric rank
$f(\alpha)$ - model performance for configuration α
$HW_i(\alpha)$ - hardware performance for configuration α
T_i - Normalization term for i-th hardware metric (HW_i)

Ranking Loss details

Ranking Loss Overview:

Given: $\mathbf{s} = [s_1, \ldots, s_M] \in \mathbb{R}^M$ (predicted scores), $\boldsymbol{\pi} = [\pi_1, \ldots, \pi_M]$ (true ranking indices)

where π_1 is best model, π_2 is second-best, etc.

Loss Computation:

$$\mathcal{L}_{\text{rank}}(\mathbf{s}, \boldsymbol{\pi}; \tau) = \sum_{m=1}^{M} \left[\log \sum_{j=m}^{M} \exp\left(\frac{s_{\pi_j}}{\tau}\right) - \frac{s_{\pi_m}}{\tau} \right]$$

$$= \sum_{m=1}^{M} -\log \frac{\exp(s_{\pi_m}/\tau)}{\sum_{j=m}^{M} \exp(s_{\pi_j}/\tau)}$$

$$= -\sum_{m=1}^{M} \log P(\text{model } \pi_m \text{ is ranked } m\text{-th} \mid \text{remaining models})$$

where $\tau > 0$ is temperature parameter (default $\tau = 1$)

Dataset Similarity

	MNIST	SVHN	USPS Digits	Fashion MNIST	Deepfashion Inshop	Fashion Product	ImageNet 1k	CIFAR-10	CIFAR-100	Caltech-101	HAM10000	Aircraft	SUN397	Cars	DTD	Pets
MNIST	1	0.6	0.85	0.4	0.1	0.1	0.05	0.2	0.1	0.15	0.05	0.15	0.15	0.15	0.1	0.1
SVHN	0.6	1	0.65	0.3	0.1	0.1	0.05	0.25	0.15	0.2	0.05	0.2	0.2	0.2	0.1	0.1
USPS Digits	0.85	0.65	1	0.35	0.1	0.1	0.05	0.15	0.1	0.15	0.05	0.15	0.15	0.15	0.1	0.1
Fashion MNIST	0.4	0.3	0.35	1	0.4	0.45	0.15	0.25	0.2	0.35	0.2	0.3	0.25	0.3	0.35	0.25
Deepfashion Inshop	0.1	0.1	0.1	0.4	1	0.8	0.35	0.3	0.25	0.45	0.25	0.4	0.35	0.4	0.45	0.3
Fashion Product	0.1	0.1	0.1	0.45	0.8	1	0.4	0.35	0.3	0.5	0.3	0.45	0.4	0.45	0.5	0.3
ImageNet 1k	0.05	0.05	0.05	0.15	0.35	0.4	1	0.65	0.8	0.6	0.3	0.75	0.75	0.7	0.7	0.75
CIFAR-10	0.2	0.25	0.15	0.25	0.3	0.35	0.65	1	0.75	0.55	0.25	0.6	0.55	0.55	0.6	0.65
CIFAR-100	0.1	0.15	0.1	0.2	0.25	0.3	0.8	0.75	1	0.65	0.25	0.7	0.65	0.65	0.65	0.7
Caltech-101	0.15	0.2	0.15	0.35	0.45	0.5	0.6	0.55	0.65	1	0.35	0.8	0.7	0.75	0.65	0.8
HAM10000	0.05	0.05	0.05	0.2	0.25	0.3	0.3	0.25	0.25	0.35	1	0.25	0.25	0.2	0.3	0.2
Aircraft	0.15	0.2	0.15	0.3	0.4	0.45	0.75	0.6	0.7	0.8	0.25	1	0.65	0.8	0.6	0.6
SUN397	0.15	0.2	0.15	0.25	0.35	0.4	0.75	0.55	0.65	0.7	0.25	0.65	1	0.6	0.75	0.55
Cars	0.15	0.2	0.15	0.3	0.4	0.45	0.7	0.55	0.65	0.75	0.2	0.8	0.6	1	0.5	0.6
DTD	0.1	0.1	0.1	0.35	0.45	0.5	0.7	0.6	0.65	0.65	0.3	0.6	0.75	0.5	1	0.55
Pets	0.1	0.1	0.1	0.25	0.3	0.3	0.75	0.65	0.7	0.8	0.2	0.6	0.55	0.6	0.55	1

Creating Ground Truth Rankings

- Step 1: Load Raw Data

$$\mathcal{P} = (d, m, a_{d,m}) \text{ where } a_{d,m} \in [0, 1] \text{ is model accuracy}$$

$$\mathcal{S} = (d_i, d_j, \sigma_{ij}) \text{ where } \sigma_{ij} \in [0, 1] \text{ is dataset similarity}$$

- Step 2: Compute Weighted Scores

$$\text{score}(m_k, d_t) = \begin{cases} a_{d_t, m_k} + 1 & \text{if model } m_k \text{trained on } d_t \\ a_{d_b, m_k} \times \sigma(d_t, d_b) & \text{if model } m_k \text{trained on } d_b \neq d_t \end{cases}$$

- Step 3: Build Score Vector

$$\mathbf{s}(d_t) = [\text{score}(m_1, d_t), \text{score}(m_2, d_t), \ldots, \text{score}(m_M, d_t)] \in \mathbb{R}^M$$

- Step 4: Add Tie-Breaking Epsilon

$$\tilde{\mathbf{s}} = \mathbf{s} + \boldsymbol{\epsilon} \text{ where} \boldsymbol{\epsilon}_i = i \times 10^{-6} \text{ for } i = 0, 1, \ldots, M - 1$$

- Step 5: Argsort to Get True Ranks

$$\boldsymbol{\pi} = \text{argsort}(\tilde{\mathbf{s}}, \text{descending=True}) = [\pi_1, \pi_2, \ldots, \pi_M]$$

where π_1 is the index of the best-performing model

Training Model AutoEncoder

rithm 1 Training the Model Autoencoder (ArchToVec)

ıire: Set of extracted layer weights $\mathcal{W}$, Regularization weight λ_{reg}, Learning rate

ıitialize Encoder f_{enc} and Decoder f_{dec}

/hile not converged **do**

Sample batch of weight vectors $\mathbf{w} \sim \mathcal{W}$

Normalize inputs: $\mathbf{w}' \leftarrow (\mathbf{w} - \mu)/(\sigma + \epsilon)$

Forward pass (Encoder): $\mathbf{z} \leftarrow f_{enc}(\mathbf{w}')$

Forward pass (Decoder): $\hat{\mathbf{w}}' \leftarrow f_{dec}(\mathbf{z})$

Compute Reconstruction Loss: $\mathcal{L}_{rec} \leftarrow \text{SmoothL1}(\hat{\mathbf{w}}', \mathbf{w}')$

Compute Sparsity Penalty: $\mathcal{L}_{reg} \leftarrow \lambda_{reg} \cdot \text{mean}(\texttt{\textbackslash mathbf\{z\}})$

Total Loss: $\mathcal{L}_{total} \leftarrow \mathcal{L}_{rec} + \mathcal{L}_{reg}$

Update parameters: $\theta_{AE} \leftarrow \theta_{AE} - \eta \nabla \mathcal{L}_{total}$

nd while

eturn Trained Encoder f_{enc} to generate Model Tokens θ_m

Training Cross Attention Transformer

Algorithm 2 Training the Cross-Attention Transformer

Require: Dataset tokens $\{\tau(D_i)\}$, Model tokens Θ_{zoo}, Ground truth ranks $\{\boldsymbol{t}_i\}$

Require: Temperature Schedule $T(step)$, Learning rate η

Initialize Transformer $\mathcal{T}_{cross}$

for epoch $= 1$ to N_{epochs} **do**

  for each batch $(\tau(D), m_{zoo}, \boldsymbol{\pi})$ in DataLoader **do**

    Get current temperature: $T \leftarrow \text{Scheduler}(step)$

    Forward Pass: For 6 layers

    Encode Dataset: $K, V \leftarrow \text{Encoder}(\tau(D))$

    Decode Models: $\hat{\boldsymbol{s}}_{zoo} \leftarrow \text{Decoder}(m_{zoo}, \text{context} = (K, V))$

    Compute Loss:

    Normalize Scores $\hat{y}_{zoo} = \sigma(\hat{s}_{zoo})$

    Calculate listwise ranking loss $\mathcal{L}_{rank}(\hat{\boldsymbol{y}}, \boldsymbol{\pi}, T)$

    Optimization:

    Update parameters: $\theta_{\mathcal{T}} \leftarrow \theta_{\mathcal{T}} - \eta \nabla \mathcal{L}_{rank}$

  end for

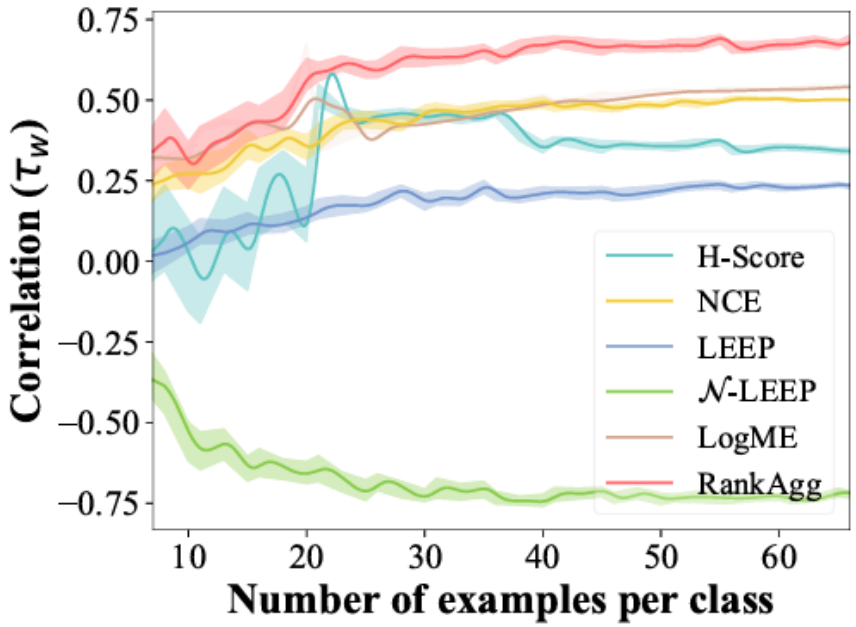
end for

Model Spider Results for increasing models

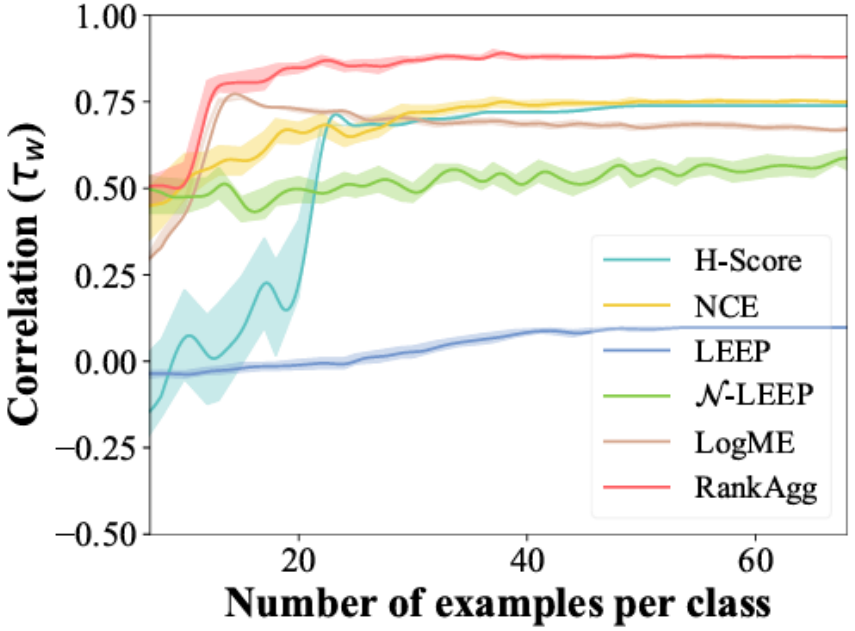
Table 5: **Ablation studies** on the performance of MODEL SPIDER when the pre-trained model repository grows dynamically.

MODEL SPIDER	Aircraft	Caltech101	Cars	CIFAR10	CIFAR100	DTD	Pets	SUN397	**Mean**
When the number of PTMs increases									
w/ number of 3	0.545	1.000	1.000	1.000	0.182	1.000	1.000	1.000	0.841
increase to 6	0.573	0.627	0.818	0.905	0.839	0.445	0.888	0.336	0.679
increase to 10	0.568	0.637	0.576	0.797	0.695	0.796	0.573	0.436	0.635

Variation of number of samples per classes

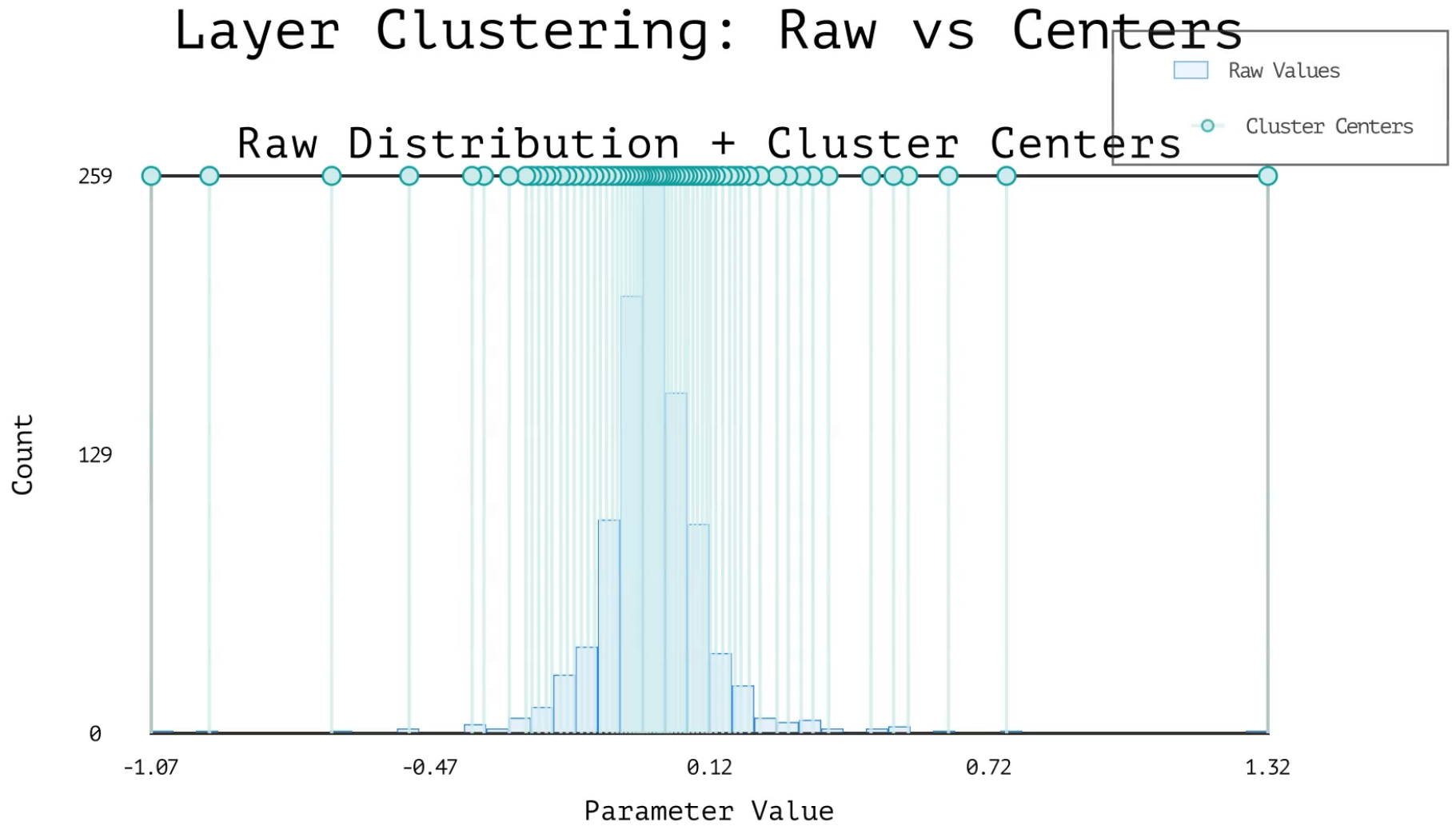


(a) on Aircraft

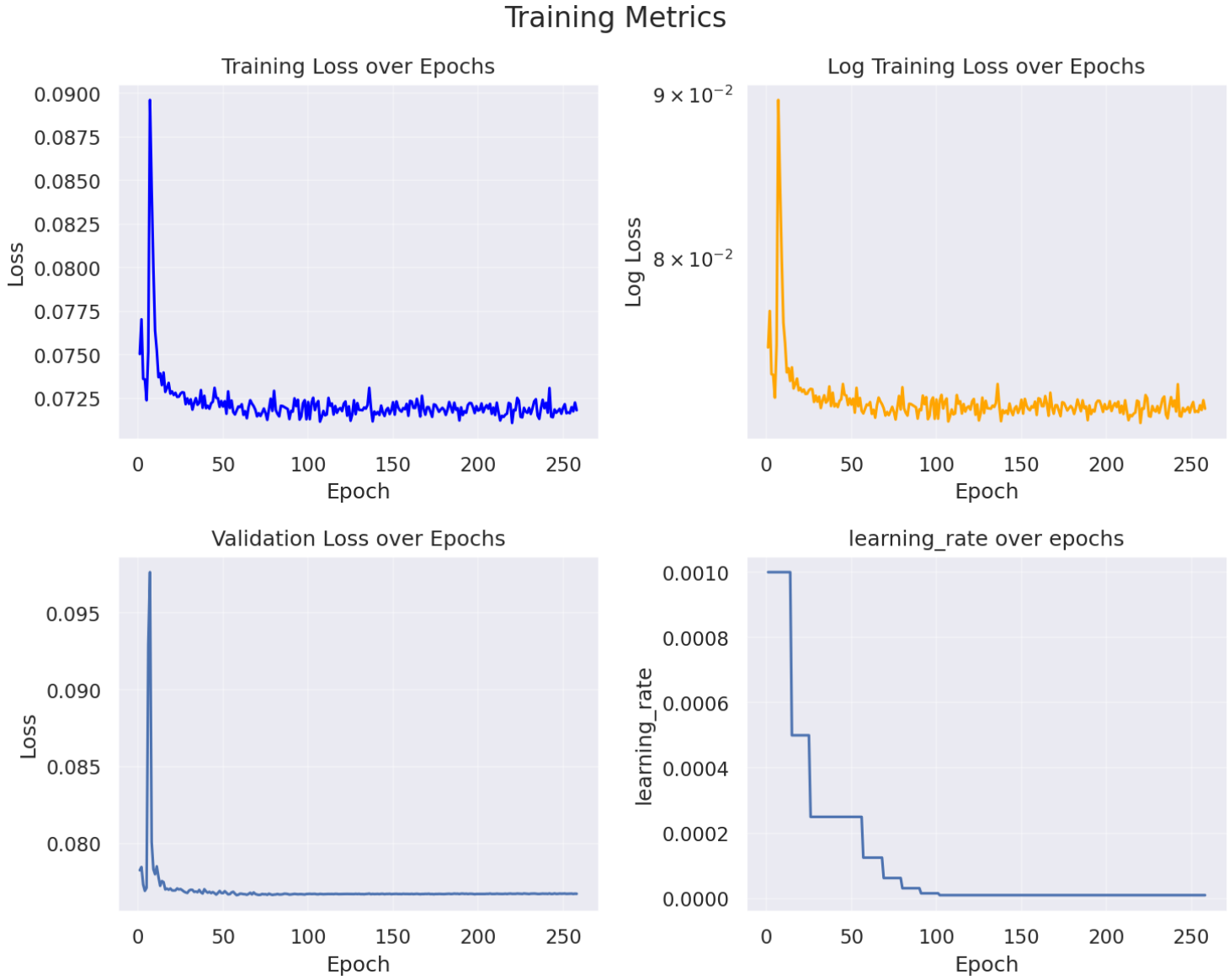


(b) on Caltech101

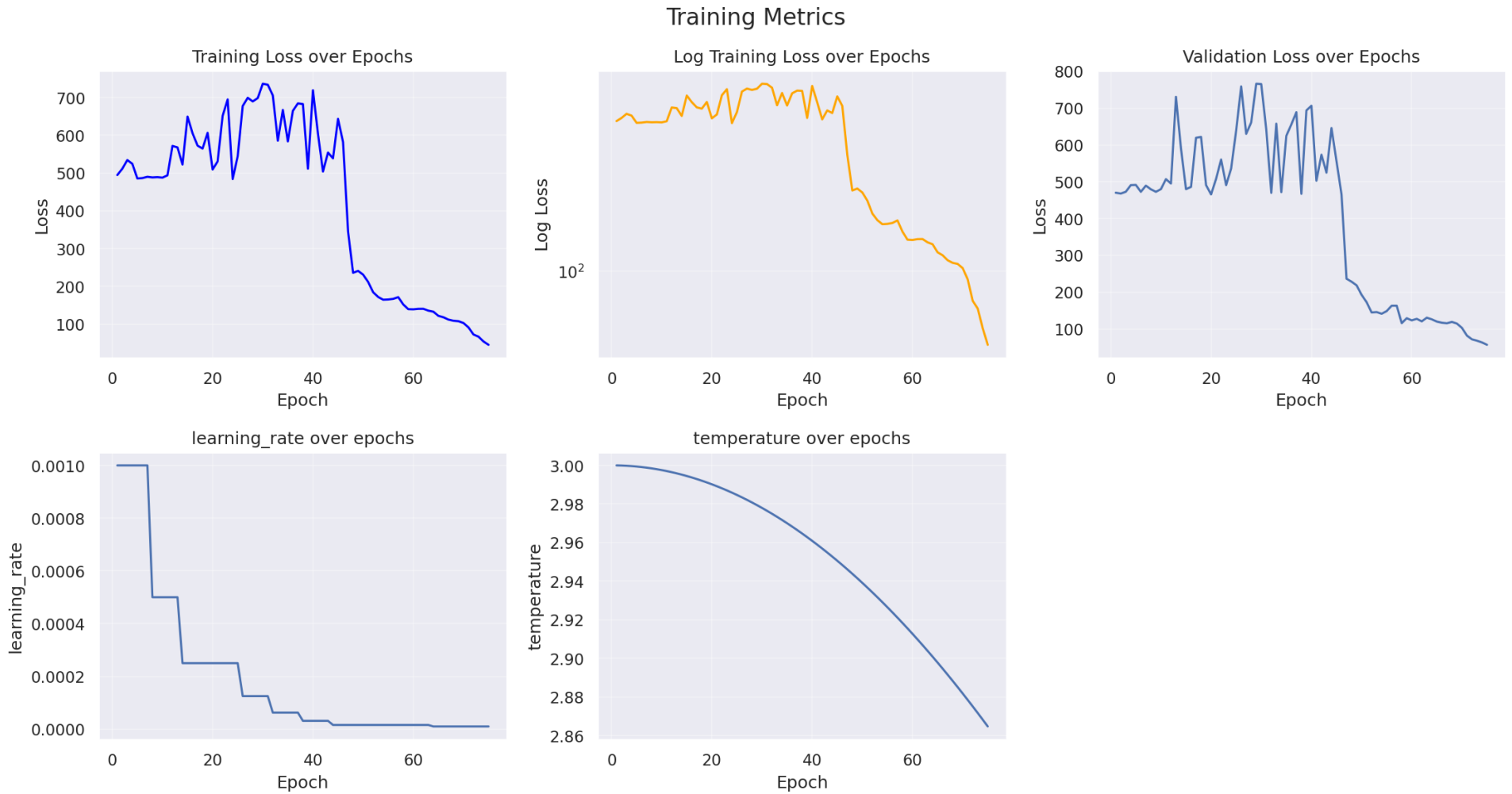
Clustering for Model Layer



Training Metrics for Model Auto Encoder



Training Metrics for Cross-Sight



Training for Cross-Sight

- Trained on Purdue's Gautschi.
- CPU: 64 core
- GPU: NVIDIA L40S 40GB
- Training Time: ~10 hours
- Total steps per epoch: 37904
- Total epochs: 300 ~ 400∫

Comparison with Hardware performance predictors

- NeuSight, a forecasting framework to predict the performance of a diverse range of deep learning models, for both training and inference, on unseen GPUs
- Key difference only comment on Pre-trained performance and not on fine-tuned performance.
- But this can be used generate Hardware Task token in the new Proposed approaches Model Spider Fusion and Shadow

Comparison with Hardware performance predictors

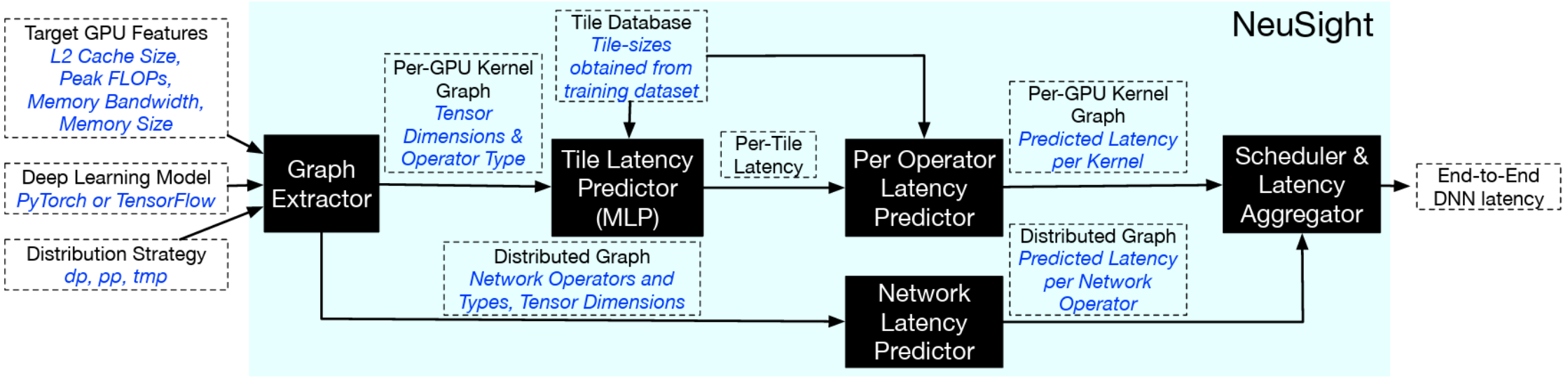


Figure 6. Overall workflow of NeuSight. The Tile Latency Predictor uses an MLP to determine the utilization and predicts the kernel latency. This prediction is aggregated for per-device and distributed execution latency forecasting.