ECE382N.23:
Embedded Systems Design and Modeling

ML-Based Workload Forecasting

Guest: Erika Susana Alcorta

Andreas Gerstlauer
Electrical and Computer Engineering
University of Texas at Austin
`gerstl@ece.utexas.edu`

The University of Texas at Austin
Chandra Department of Electrical and Computer Engineering
Cockrell School of Engineering

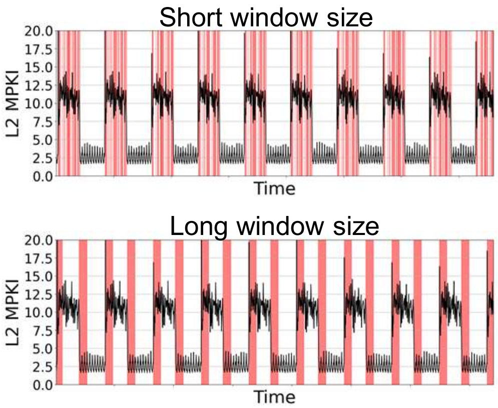
Outline

- **Workload Forecasting**
  - Background
  - Proactive runtime management
  - Phase-Aware Workload Forecasting
- **Long-term predictions**
  - Phase classification
  - Phase prediction
  - Evaluation
- **Short-term predictions**
  - Phase-aware workload forecasting
  - Evaluation
- **Summary**
- **Future Work and Project Ideas**

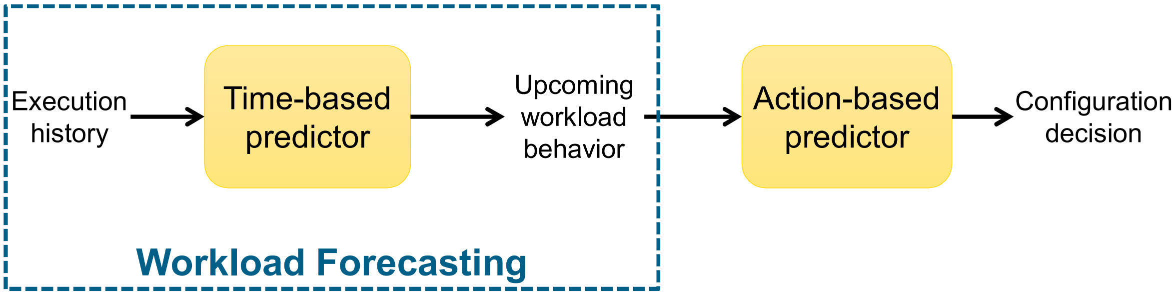
Background

- **Runtime management**
  - Dynamic optimization opportunities
  - Examples: frequency scaling, cache partitioning, prefetcher throttling, task scheduling
  - Most existing solutions are reactive
- **Proactive management**
  - Map future workload behavior into an action
  - Unlocks optimization opportunities
    - We observe it can improve IPC gains by 1.5x with prefetcher selection
    - Up to 45% energy savings with heterogeneous task scheduling[Roeder'21]
- **Requires accurately predicting the future**
  - ➢ Machine learning (ML) is a promising approach
  - ➢ Workload forecasting

Reactive decisions examples



Proactive Runtime Management



Execution history → Time-based predictor → Upcoming workload behavior → Action-based predictor → Configuration decision

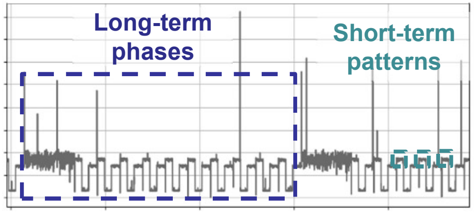
Workload Forecasting

Workload Forecasting

- **Workload behaviors are repetitive**
  - Learn from the past to predict the future
- **Long-term predictions**
  - Phases are periods of similar behavior
  - Detect phases and learn phase patterns
  - ➢ Phase classification and prediction
- **Short-term predictions**
  - Short-term variations within phases
  - Regression and time series forecasting
  - ➢ Short-term workload forecasting

➢ **Only studied separately**

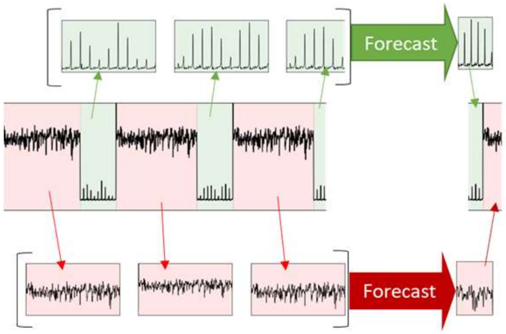
Snippet of CPI trace



Phase-Aware Workload Forecasting

- **Forecasting models struggle to predict long-term patterns**
- **Long-term phase classification and prediction**

➢ **Phase-aware workload forecasting**



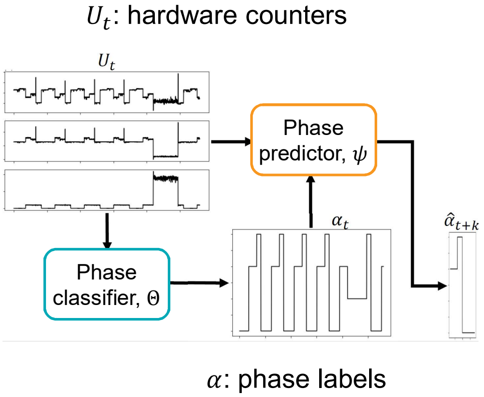
Questions?

NEXT: LONG-TERM PREDICTIONS

Long-Term Predictions

- **Phase classification**
  - Find periods of execution with similar behaviors
  - Requires selecting inputs and clustering method
  - ➢ Hardware counters as inputs
  - ➢ Explore clustering methods
- **Phase prediction**
  - Learns patterns between phases
  - Foretells future phases
  - ➢ Explore prediction strategies

➢ **Best combination?**



Phase Classification Models

Classifier	Inputs	Description	Hyperparameters
table [Sembrant '11]	~~Branch instr ptrs.~~ HW counters	•Incremental clustering •Leader-follower algorithm	•Distance •Threshold
pcakmeans [Khanna '12]	HW counters	•Iterative clustering •Reduce dimensionality with PCA •Cluster new dimensions with K-Means	•Dimensionality •# of clusters
gmm [Chiu '18]	~~Annotated branches~~ HW counters	•Iterative clustering •Assumes gaussian models distribution in clusters •Considers cluster variance in addition to the mean	•# of clusters
2kmeans [Khoshbakht '17]	HW counters	•Two-level iterative clustering •Level 1: clusters each sample with k-means •Level 2: clusters frequency vectors of fixed-size windows of 1st level clusters with k-means	•Window size •# of clusters

Phase Prediction Models

Prediction strategy		Description	Inputs	Previous models	Proposed models
Window-based prediction		Determine the **phase ID** of an **upcoming** fixed size **window of samples**	• History of phases • History of pre-classified data	Decision trees [Chiu '18]	LSTM, SVM
Phase change prediction	Next phase prediction	Predict the phase **ID** of the **next phase transition**	• History of phase transitions	GHT [Zhang '15]	LSTM, SVM
	Phase duration prediction	Predict the **duration** of the current (target) phase	• History of durations of all phases • History of durations of the target phase	Linear adaptive filter [Srinivasan '13]	SVM, MLP, LR, LV

LV: last value. MLP: multi-layer perceptron

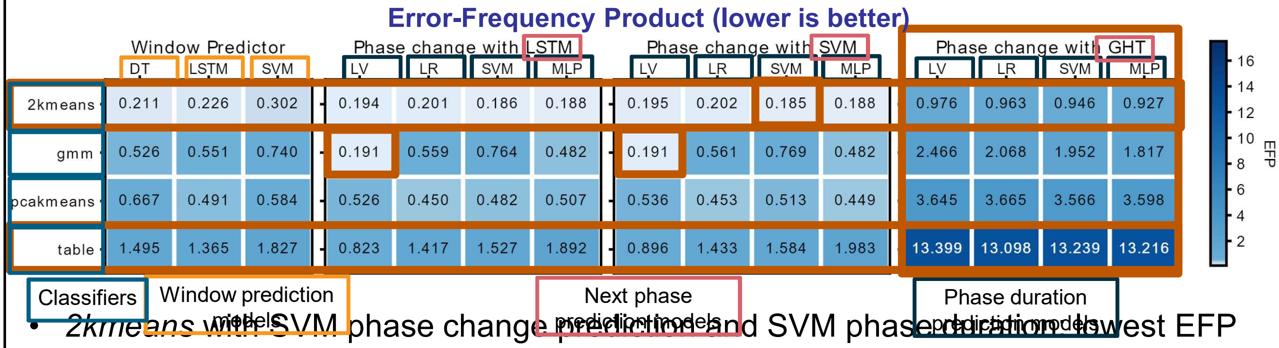
Long-Term Models Study

- **What is the best combination?**
  - Phase classifiers and predictors have only been studied in specific combinations in prior work
- **Does the classifier find long-term phases?**
- **Can the phase predictor learn the generated phase patterns?**
- **Methodology**
  - Evaluate all combinations of phase classification and prediction
  - Error-frequency product (EFP)
    - Penalize prediction error and high transition frequency

Experimental Setup

- **Data collection**
  - Platform: Intel i9-9900K
  - Workloads
    - SPEC CPU 2017 (single and multi-core experiments)
    - Parsec 3.0 (multi-core experiments)
  - Performance monitoring counters:
    - L2 misses, main memory accesses, mis-predicted branches, retired FP operations
  - Period: 10 ms
- **Implementation**
  - Variable of interest: CPI
  - Train-test split: 70%-30%
  - Hyperparameter tuning with training set

Single Core Results



Error-Frequency Product (lower is better)

	Window Predictor			Phase change with LSTM				Phase change with SVM				Phase change with GHT			
	DT	LSTM	SVM	LV	LR	SVM	MLP	LV	LR	SVM	MLP	LV	LR	SVM	MLP
2kmeans	0.211	0.226	0.302	0.194	0.201	0.186	0.188	0.195	0.202	0.185	0.188	0.976	0.963	0.946	0.927
gmm	0.526	0.551	0.740	0.191	0.559	0.764	0.482	0.191	0.561	0.769	0.482	2.466	2.068	1.952	1.817
pcakmeans	0.667	0.491	0.584	0.526	0.450	0.482	0.507	0.536	0.453	0.513	0.449	3.645	3.665	3.566	3.598
table	1.495	1.365	1.827	0.823	1.417	1.527	1.892	0.896	1.433	1.584	1.983	13.399	13.098	13.239	13.216

- *2kmeans* with SVM phase change prediction and SVM phase duration lowest EFP
- *2kmeans* is the best classifier for 13 out of 15 predictors, *gmm* is the best for the other 2
  - ➢ **Key difference: two-level clustering**
- Worst classifier: *table,* worst predictor: GHT
  - ➢ **ML-based models are better than table-based (ML generalizes better)**

Multi-Core Phases

- **How would you design the models to classify and predict multi-core phases?**

Multi-core Phase Classification

- **Counter traces per core in multi-core platform**
  - N traces, $(U_{1,t}, \dots, U_{N,t})$, one per core
  - Should phases be **global (G)** for all cores or **local (L)** for each core?
  - Should local classifiers **share (L-S)** cluster centers?
- **Multi-core settings for phase classification**
  - G: a classifier, θ_p, outputs one system-wide phase for all cores per timestep, α_t
  - L: a classifier per core with its own parameters, θ_{p_n}, outputs phases for its core, $\alpha_{n,t}$, $1 \leq n \leq N$
  - L-S: L classifiers share trainable parameters ρ, i. e., $p_n = \rho$

Global

$U_{1,t}$ … $U_{N,t}$

θ_p

α_t

Local

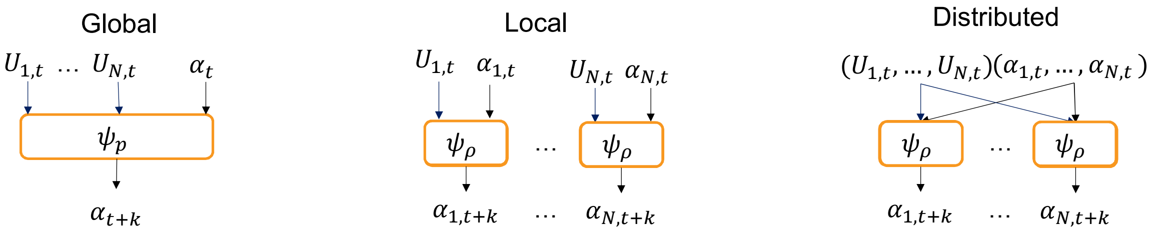
$U_{1,t}$ $U_{N,t}$

θ_ρ … θ_ρ

$\alpha_{1,t}$ $\alpha_{N,t}$

Multi-core Phase Prediction

- **Global phases**
  - Use a **global (G)** predictor, ψ_p, that predicts system-wide phases, α_{t+k}
- **Local phases**
  - Use a **local (L)** predictor per core, ψ_{p_n}, that predicts phases for one core, $\alpha_{n,t+k}$
  - Account for other cores activity with a **distributed (D)** predictor
  - Support shared variants, for local **L-S** and distributed **D-S**



Global

$U_{1,t}$ … $U_{N,t}$ α_t

ψ_p

α_{t+k}

Local

$U_{1,t}$ $\alpha_{1,t}$ $U_{N,t}$ $\alpha_{N,t}$

ψ_ρ … ψ_ρ

$\alpha_{1,t+k}$ … $\alpha_{N,t+k}$

Distributed

$(U_{1,t}, \dots, U_{N,t})(\alpha_{1,t}, \dots, \alpha_{N,t})$

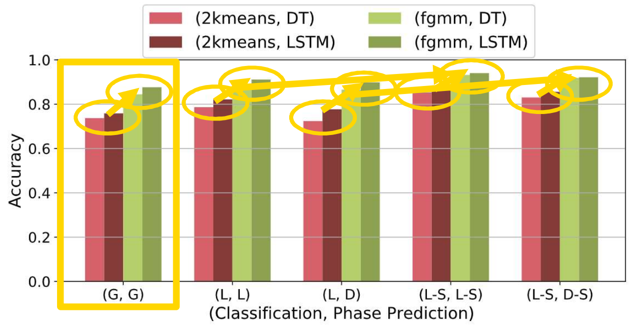
ψ_ρ … ψ_ρ

$\alpha_{1,t+k}$ … $\alpha_{N,t+k}$

Multi-core Evaluation

- **Multi-core combinations: (classification, prediction)**
  - (G, G), (L, L), (L, D), (L-S, L-S), (L-S, D-S)
- **Phase classifiers**
  - ***2kmeans*** was best in combination with phase prediction
    - Has a filtering step with an extra level of clustering
    - Windows for its frequency vectors caused high variance within phases
  - *gmm* was best in isolation
    - Add a filtering step
    - ***fgmm***
- **Phase predictors**
  - Phase change prediction struggled to learn multi-threaded patterns[Alcorta'22]
  - Window-based prediction: **DT** and **LSTM** close winners in single-core evaluation

Multi-core Phase Prediction Results



- **(L-S, L-S) with (*fgmm*, LSTM) has the best average accuracy**
- **Local phases are better than global**
  - G, G tends to have the lowest accuracy
- **Sharing trainable parameters helps prediction accuracy**
  - L-S and D-S predictors have higher accuracy than their L and D counterparts
- **Both DT and LSTM with *fgmm* have consistently higher accuracy than *2kmeans***
- **LSTM is better than DT**

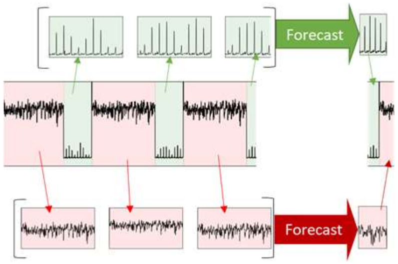
Questions?

NEXT: SHORT-TERM PREDICTIONS

Short-Term Predictions

- **Phase prediction overlooks short-term variations**
- **Forecast workload metrics**
  - Time series forecasting
  - ➢ Struggle to predict long-term effects

➢ **Phase-aware forecasting**



Basic Forecasting

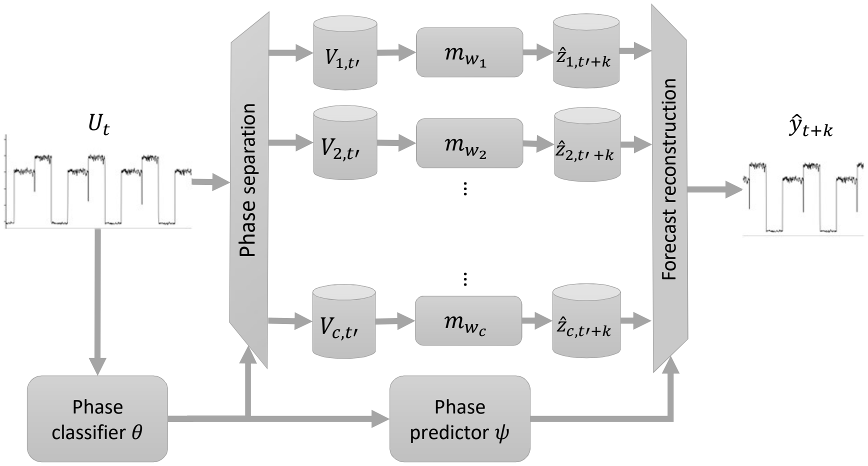
- **Workload forecasting formulation**

$$(\hat{\boldsymbol{y}}_{t+1}, \dots, \hat{\boldsymbol{y}}_{t+k}) = \boldsymbol{m}_{w}(\boldsymbol{U}_{t-h+1}, \dots, \boldsymbol{U}_{t})$$

  - U_t: vector of hardware counters at time t
  - y_t: counter variable of interest, e.g., CPI, at time t
  - $\hat{y}_{t+i}$: prediction of y_{t+i} made at time t
  - m_w: model function with given model parameters w, e.g., ML model
  - h: input history size
  - k: forecast horizon

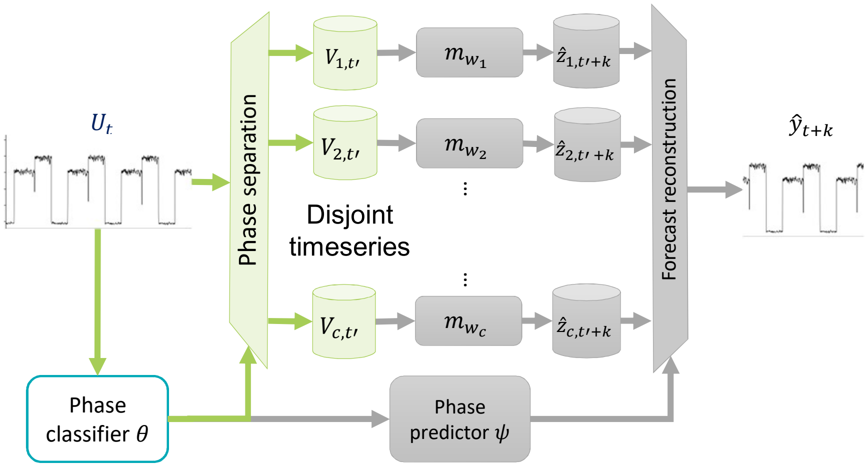
Phase-Aware Forecasting

- **Overview**



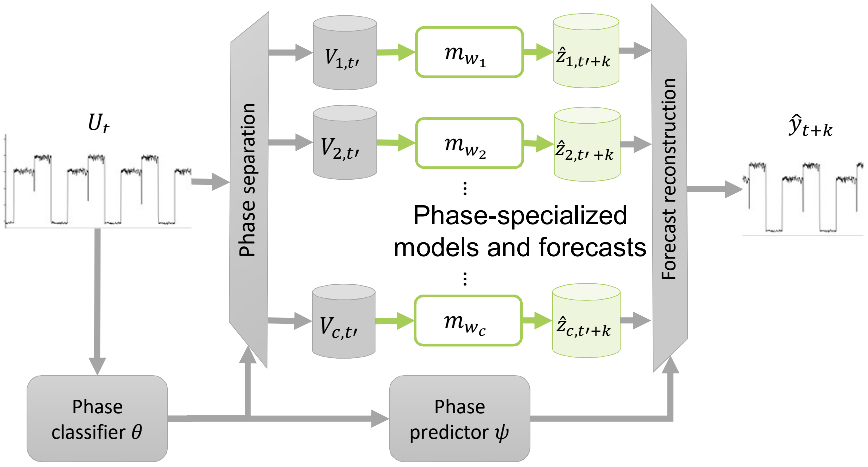
Phase-Aware Forecasting

1. Phase classification and separation



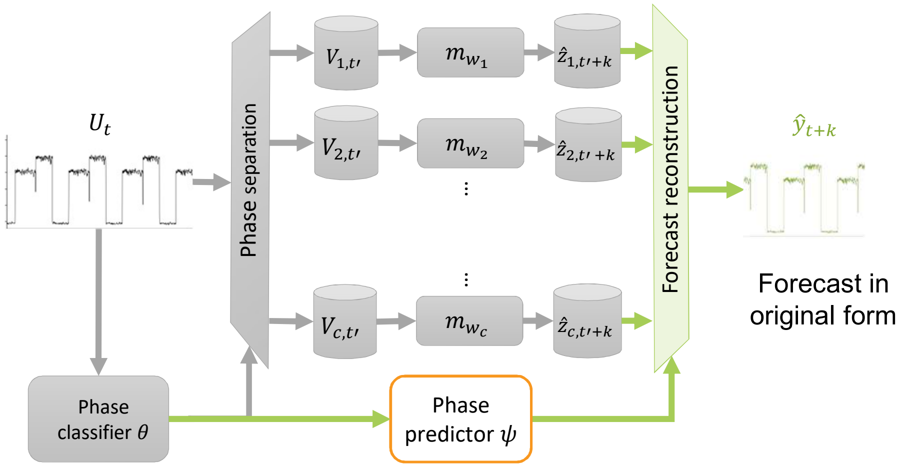
Phase-Aware Forecasting

2. Phase-based forecasting



Phase-Aware Forecasting

3. Phase prediction and forecast reconstruction



Forecast in original form

Forecasting Models

- **Support Vector Machines (SVM)**
  - Minimizes an error bound instead of residuals
  - Commonly used with non-linear transformations
- **Long-Short Term Memory (LSTM)**
  - Recurrent neural network
  - Popular for handling time-dependent data
- **Dynamic Linear Model (DLM)***
  - Dynamically regressive
  - State-space model representation similar to Kalman filter
- **Matrix Profile (MP)***
  - Finds a subsequence in time series history closest to the most recent window
  - Predicts history repeats exactly the same

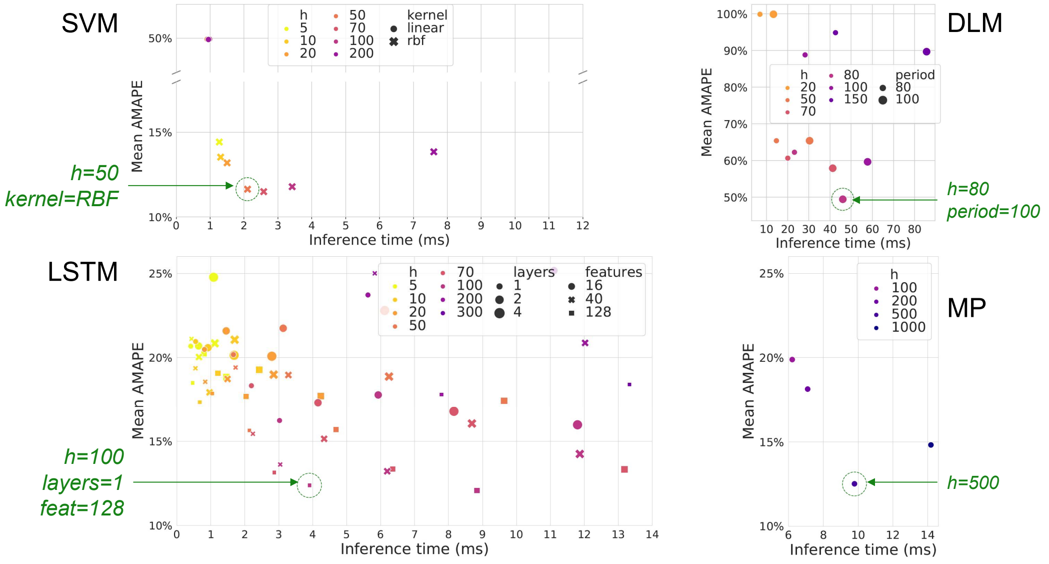
* Not used for CPU workload forecasting in the past

Experimental Setup

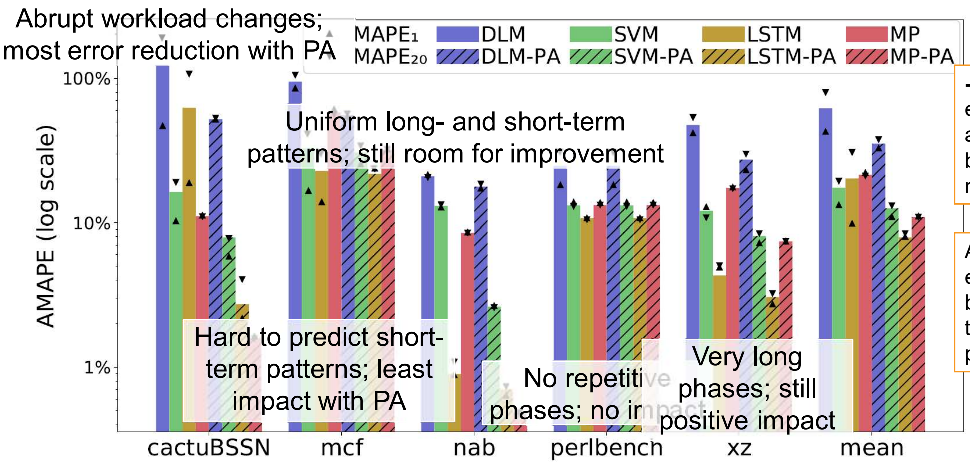
- **Data collection**
  - SPEC CPU 2017
  - Parsec-3.0 (multi-core experiments)
  - Variable of interest: CPI
  - Train-test split: 70%-30%
- **Data preprocessing for SVM and LSTM**
  - PCA
  - Median filter
- **Metrics:**
  - Mean absolute percentage error (MAPE)
  - Average MAPE (AMAPE) across forecasts horizon
- **Case study**
  - Oracle phase classification and prediction
  - Phase-unaware vs phase-aware

Single-threaded benchmarks	**Samples**	**No. of phases**	**Avg. ph. length**	**Phase repetition behavior**
cactuBSSN	202,179	5	167	abrupt transitions
mcf	52,673	5	599	hard to predict
nab	170,251	5	231	uniform pattern
perlbench	16,462	1	-	single execution
xz	126,669	4	7,037	long phases

Hyperparameter Tuning



Case Study



Abrupt workload changes; most error reduction with PA

Uniform long- and short-term patterns; still room for improvement

Hard to predict short-term patterns; least impact with PA

No repetitive phases; no impact

Very long phases; still positive impact

-PA reduced error by 14% across benchmarks and models

All **-PA** reduced error across benchmarks that go through phases

Best: **MP-PA**

Best: **LSTM-PA**

Best: **MP-PA**

Best: **LSTM/ LSTM-PA**

Best: **LSTM-PA**

Best on average: **LSTM-PA**

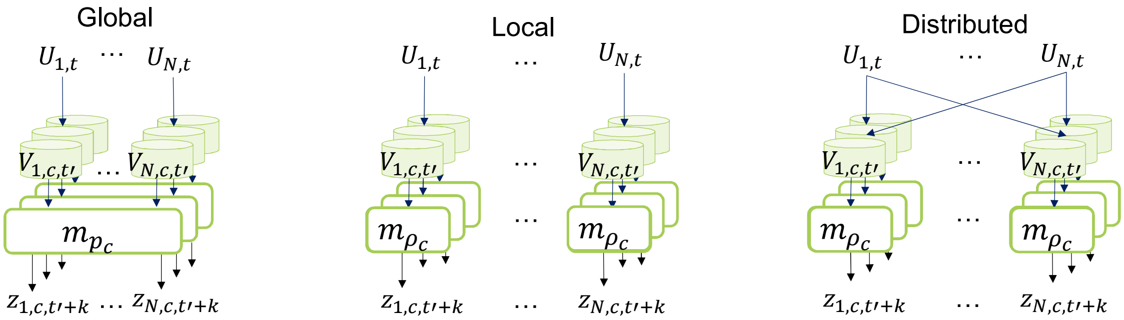
Best phase unaware: **SVM**

Multi-Core Workload Forecasting

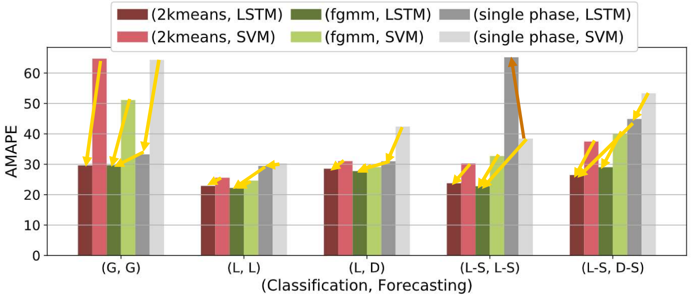
- **How would you design multi-core workload forecasting models?**

Multi-Core Phase-Based Forecasting

- **Phase-based models with per-core counter traces**
  - Global (**G**): one model per global phase outputs per-core forecasts
  - Local (**L**): one model per local phase per core outputs its core's forecasts
  - Distributed (**D**): local models that access other cores' hardware counters
  - Shared (**L-S**, **D-S**): models of the same phase share trainable parameters across cores

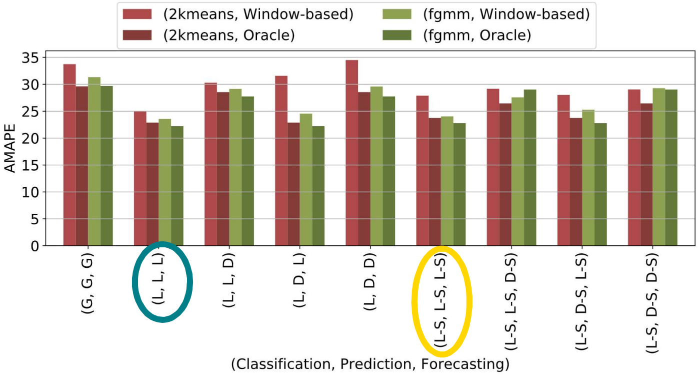


Multi-Core Phase-Based Forecasting Results



- **Best phase-based is L,L with *fgmm* and LSTM and average forecast error of 21%**
- **Best phase-unaware is also L with average forecast error of 29.8%**
- **Phase-based (2kmeans and fgmm) models outperform phase-unaware (single phase)**
  - ➢ **Phase specialized forecasts are more accurate**
- **Local forecasting (L and L-S) is generally better than distributed (D, and D-S) and global (G)**
  - ➢ **No need to access hardware counters across cores**
- **LSTM yields lower forecasting error than SVM with an exception for single-phase L-S**
- **Not significant difference between *fgmm* and *2kmeans***

Multi-core Phase-aware Forecasting Results



- **Best phase-aware: (L, L, L) with *fgmm*, LSTM window-based phase prediction and LSTM phase-based forecasting with 22.3% average forecast error**
  - Other with very close accuracy require accessing data from other cores (sharing parameters)
- ➢ **AMAPE improved from 29.8% to 22.3% on average over phase-unaware**

Summary

- **Workload forecasting**
  - Model dynamic workload behaviors
  - Guides proactive runtime management
  - ML-based forecasting improves accuracy
  - Long-term workload phases pose a challenge for basic forecasting

- **Phase-aware workload forecasting**
  - Combine short-term forecasting with long-term phase prediction
  - Evaluate multiple combinations
  - Best multi-core combination (L, L, L) with *fgmm*, window-based LSTM and phase-based LSTM
  - Reduces forecast error by 12% and 7% on single-core and multi-core experiments, respectively

Questions?

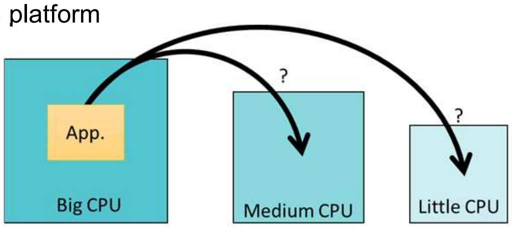
NEXT: FUTURE WORK

Workload Forecasting Future Work

- **What assumptions need to be addressed to polish workload forecasting work?**
  - Implementation in hardware vs software
  - Optimizing model inference latency
  - Reduce training time and/or training set size

- **What can you use workload forecasting for?**
  - Release data set
  - ➢ Combine with runtime management applications

Proactive Runtime Management



- **Heterogeneous Scheduling**
  - Decide where to schedule tasks in a heterogeneous platform
  - Cross-platform performance estimations
  - Task migration cost
- **Prefetcher Selection**
  - Decide when to enable/disable prefetchers
  - Combine with workload forecasting
- **Power Management**
  - Dynamic voltage frequency scaling (DVFS)
  - Minimize power without compromising performance
- **Cache reconfiguration**
  - Turn off cache lines
  - Partition cache ways across cores