Detecting and Identifying Applications by Job Signatures

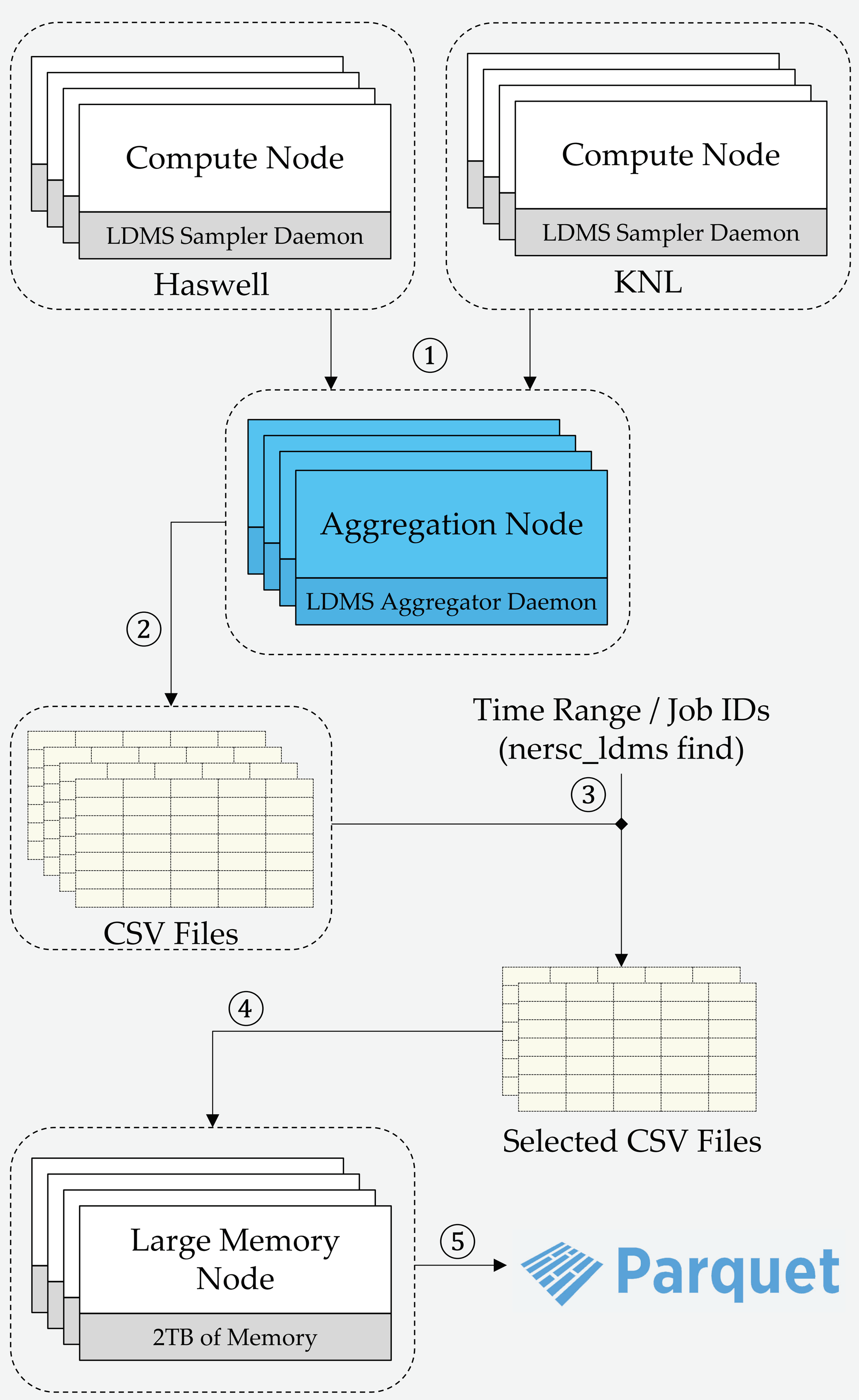
Jie Li[1], Brandon Cook[2], Yong Chen[1]

[1]Texas Tech University, [2]Lawrence Berkeley National Laboratory

ABSTRACT

Knowing the application of a job running on an HPC system is a non-trivial task if the user does not specify the job name indicating its application in the job submission script. This research aims to detect and identify applications by job signatures built upon monitoring traces obtained from the LDMS[1] monitoring infrastructure on Cori. In addition, this research will explore the use of job signatures to study workloads by computation motif.

LDMS WORKFLOW



① Aggregation nodes collect metrics sampled from compute nodes. ② These monitoring metrics are saved as large CSV files. ③ *nersc_ldms*[2] finds corresponding CSV files by time range or job IDs. ④ Selected CSV files are then submitted to large memory node for postprocessing. ⑤ Postprocessed data are saved in parquet format.

DATA PREPARATION

The dataset is generated by the following steps:

1. Two months (April and May 2021) of job metadata are retrieved from the MySQL database, where the application names are labeled.
2. An HPC expert selects the applications with explicit meaningful names.
3. The job IDs of the selected applications are used with *nersc_ldms* to obtain the corresponding monitoring metrics.
4. These metrics are post-processed and the raw metrics, sampled metrics and extracted features are saved in an HDF file for each job. There are 18703 job samples.

FEATURE EXTRACTION

LDMS collects more than 50 different metrics, and among these available metrics, we choose *power(W)*, *IPC* and *MemUsed* to build job signatures, as shown in Table 1. These three metrics are applied several statistical functions (listed in the right column of Table 2) to generate features (listed in the left column of Table 2) .

JOB SIGNATURES: the pattern within performance metrics of a job.

Metric	Sampler
power(W)	cray_aries_sampler
IPC (Instruction Per Cycle)	syspapi_Hsw64, syspapi_Knl272
PAPI_TOT_INS/ PAPI_TOT_CYC	
MemUsed	meminfo
MemTotal - MemFree	

Table 1

Feature	Function
Median	np.median
Std	np.std
Skewness	scipy.stats.skew
Kurtosis	scipy.stats.kurtosis
Percentile (5%)	lambda: x: np.percentile(x, 5)
Percentile (25%)	lambda: x: np.percentile(x, 25)
Percentile (50%)	lambda: x: np.percentile(x, 50)
Percentile (75%)	lambda: x: np.percentile(x, 75)
Percentile (95%)	lambda: x: np.percentile(x, 95)

Table 2

CLASSIFICATION

To detect and identify applications, we train a machine learning model using the labelled dataset. Specifically, we use *Random Forest*[3] as the classifier; the criterion parameter is set to "entropy"; the rest of the parameters are left unchanged.

EVALUATION

Figure 1 shows the precision-recall curve; it plots the precision (y-axis) and the recall (x-axis) of application classification for different thresholds. The random forest classifier performs the best for the *WRF* application and the worst for the *NWCHEM* application.

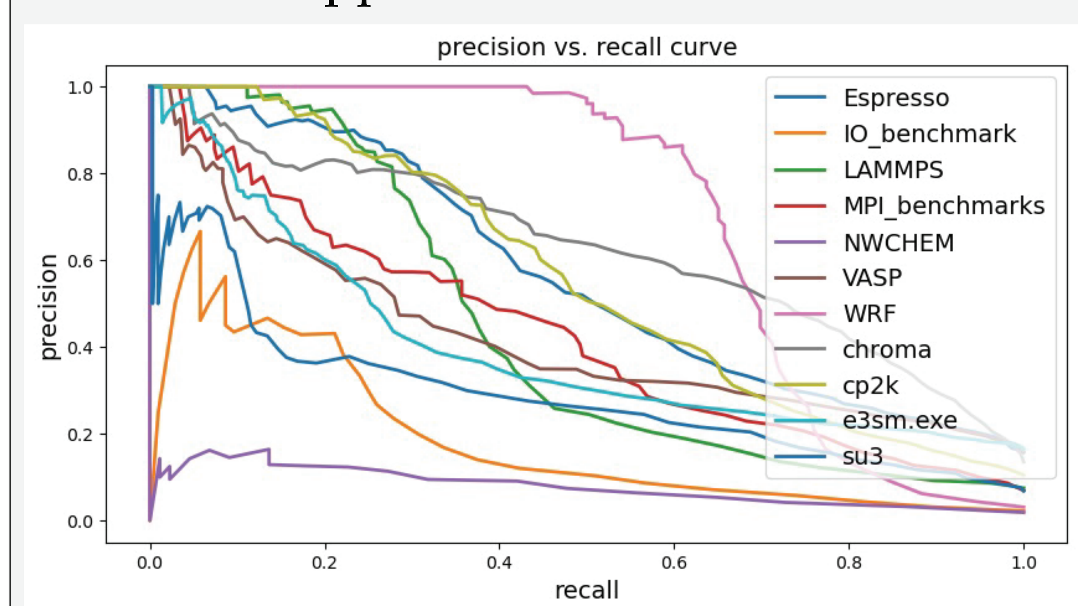


Figure 1. Precision-recall Curve

The SHAP[4] summary plot in Figure 2 shows the feature importance with feature effects: *std_MemUsed* (Standard deviation of Memory Used) contributes the most to the predictions of the model, followed by *min_MemUsed* and *min_PAPI_IPC*.

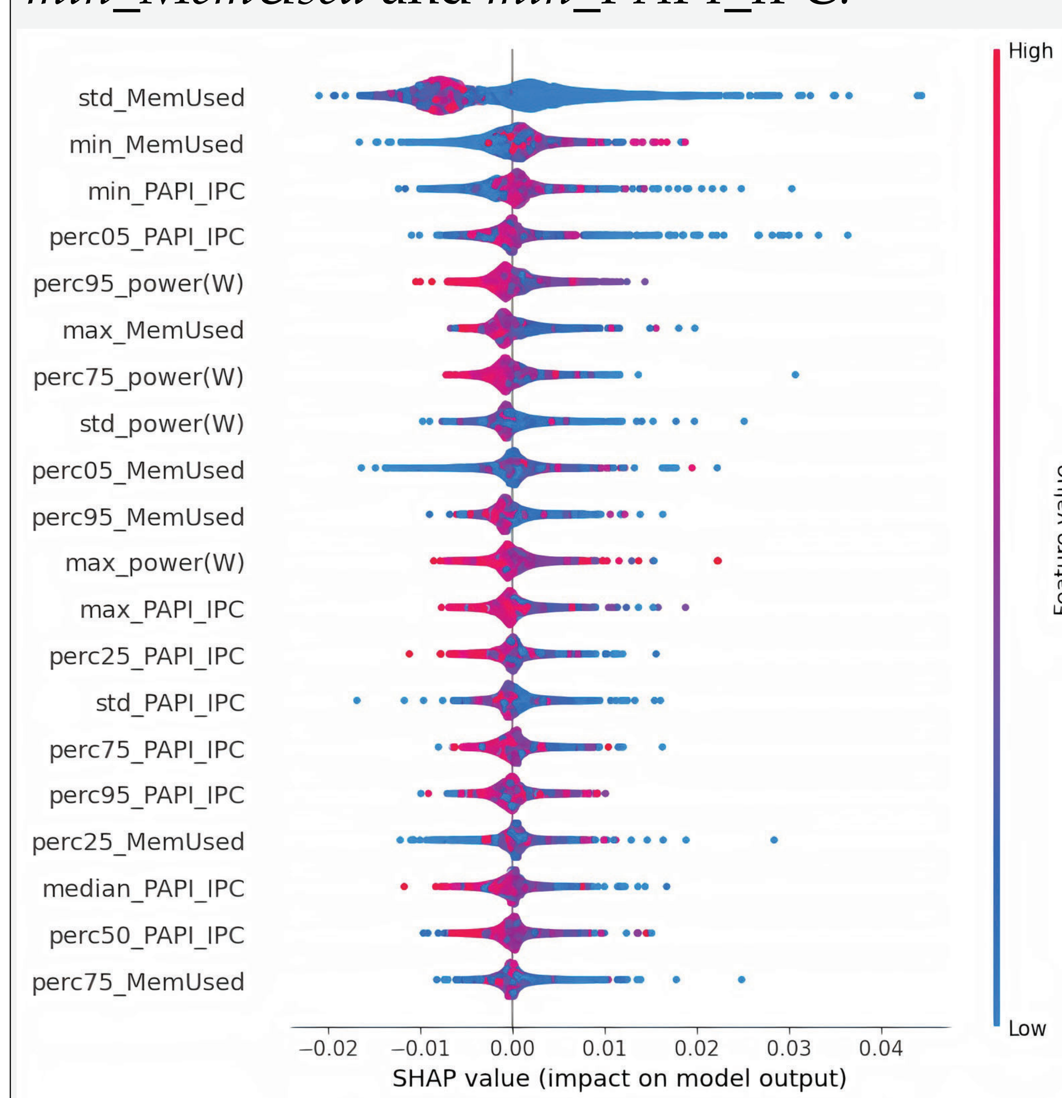


Figure 2. SHAP Summary Plot

FUTURE WORK

1. We will add more features from the available metrics to improve the classification performance and compare the performance of different classifiers.
2. It is worth exploring using the entire monitoring trace as a signature (as shown in Figure 3) and classifying the time series data with deep learning.
3. Another interesting topic we would like to explore is whether we can detect and identify job applications while the job is still running, i.e., using partial of the monitoring metrics for classification.

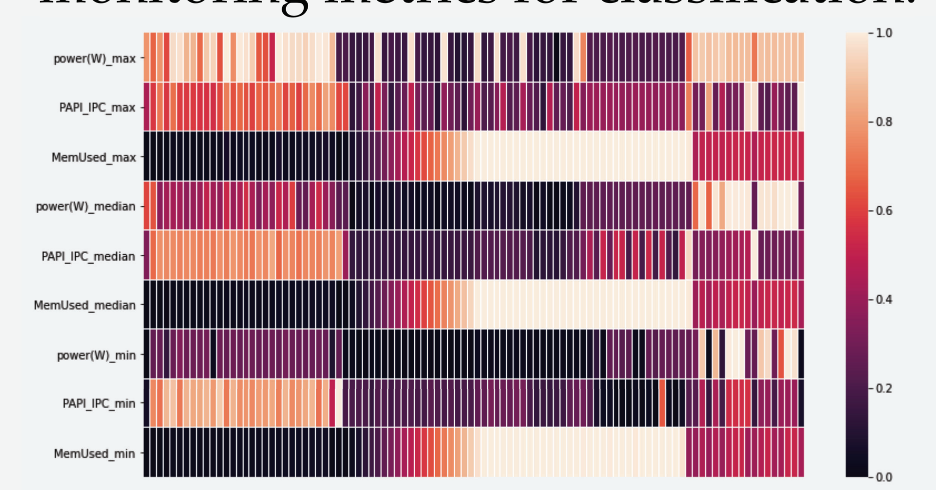


Figure 3. Time Series Signature

REFERENCES

[1] Agelastos, Anthony, Benjamin Allan, Jim Brandt, Paul Cassella, Jeremy Enos, Joshi Fullop, Ann Gentile et al. "The lightweight distributed metric service: a scalable infrastructure for continuous monitoring of large-scale computing systems and applications." In SC'14: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 154-165. IEEE, 2014.

[2] NERSC LDMS, https://software.nersc.gov/ldms/nersc-ldms

[3] Svetnik, Vladimir, Andy Liaw, Christopher Tong, J. Christopher Culberson, Robert P. Sheridan, and Bradley P. Feuston. "Random forest: a classification and regression tool for compound classification and QSAR modeling." Journal of chemical information and computer sciences 43, no. 6 (2003): 1947-1958.

[4] Lundberg, Scott M., and Su-In Lee. "A unified approach to interpreting model predictions." In Proceedings of the 31st international conference on neural information processing systems, pp. 4768-4777. 2017.

ACKNOWLEDGMENTS

This research used resources of the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility located at Lawrence Berkeley National Laboratory, operated under Contract No. DE-AC02-05CH11231.