Predicting Cross-Platform Relative Performance with Deep Generative Models

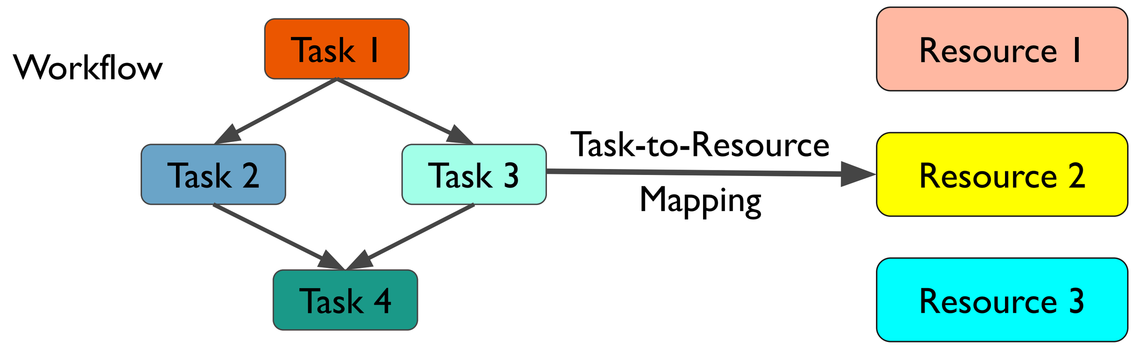
Daniel Nichols[1], Jae-Seung Yeom[2], Abhinav Bhatele[1]
[1]University of Maryland, College Park, [2]Lawrence Livermore National Laboratory

Abstract

Applications can experience significant performance differences when run on different architectures. For example, GPUs are often utilized to accelerate an application over its CPU implementation. Understanding how performance changes across platforms is vital to the design of hardware, systems software, and performance critical applications. However, modelling the relationship between systems and performance is difficult as run time data needs to be collected on each platform. In this poster, we present a methodology for predicting the relative performance of an application across multiple systems using profiled performance counters.

Motivation

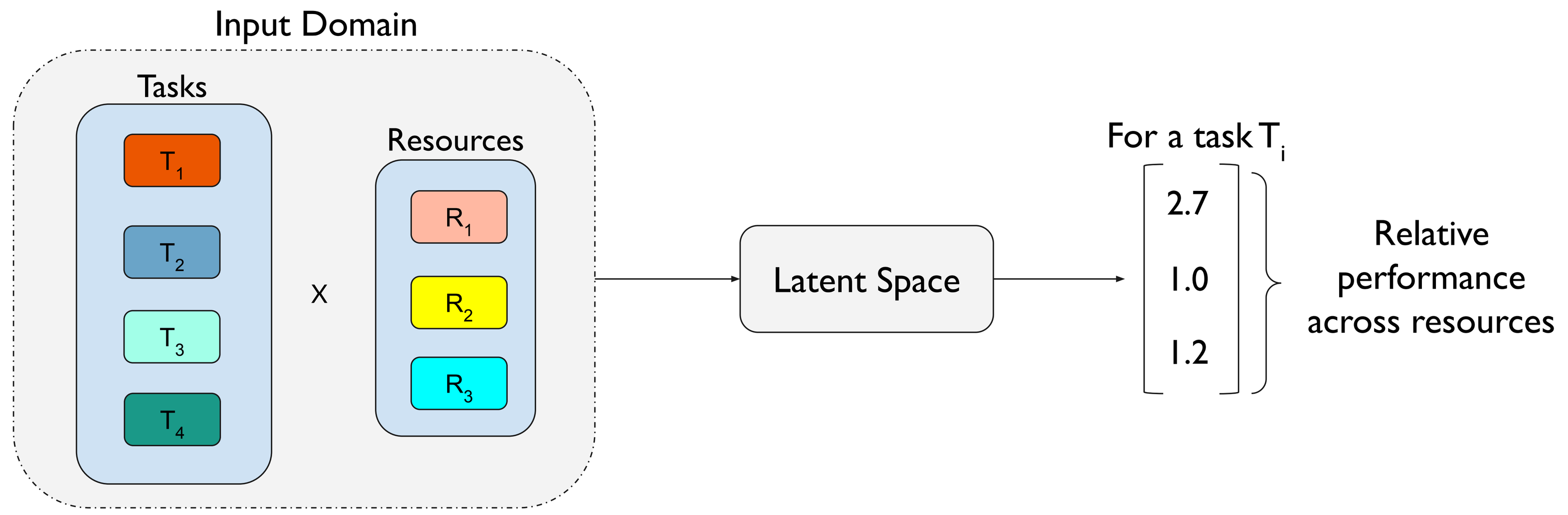
Predicting relative performance enables easier use of performance modelling results for downstream tasks. A motivating example of this is multi-resource job scheduling.



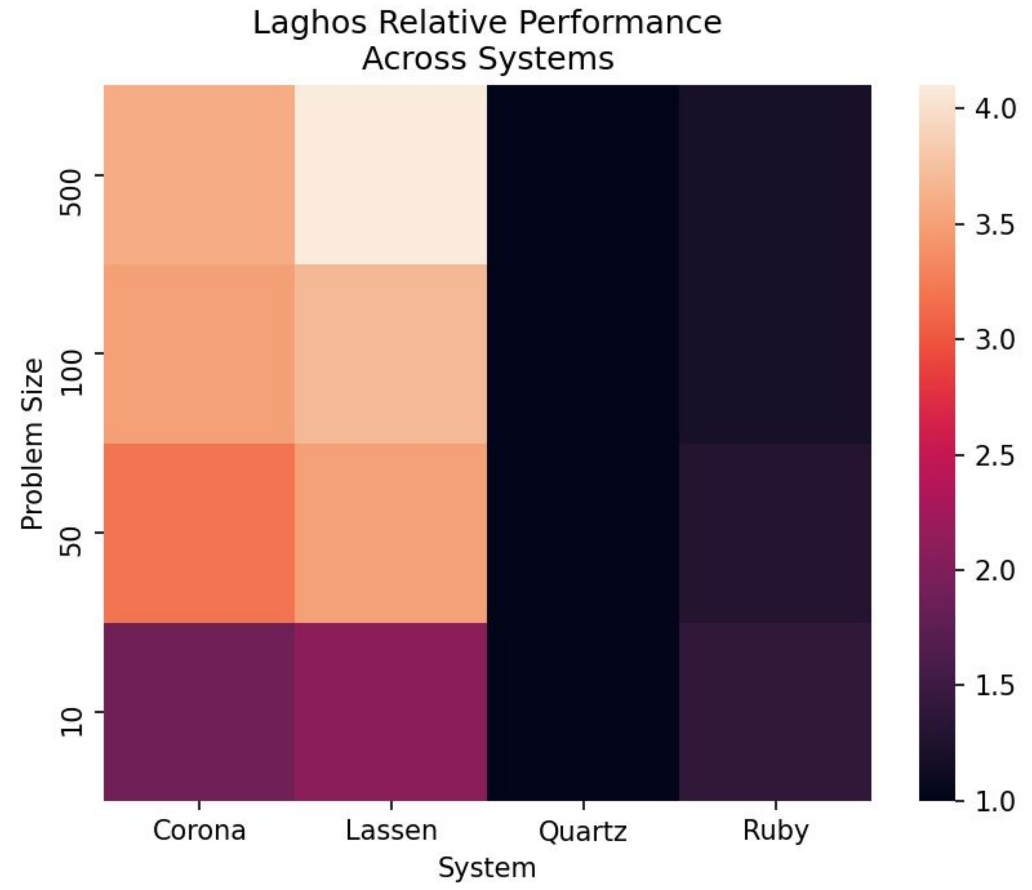
Performance predictions can be used to schedule jobs across different clusters, hardware, and cloud resources.

Problem Overview

Goal: Learn a latent space for a fixed set of resources that can map tasks to their relative performance across those resources.



Data Set



Example relative performance vectors for the Laghos proxy application across different problem sizes

Recorded counters for E4S[1], ECP[2], and coral2 applications on 4 LLNL systems:

Quartz	Ruby	Corona	Lassen
Intel Xeon E5-2695 v4	Intel Xeon CLX-8276L	AMD MI50	NVIDIA V100

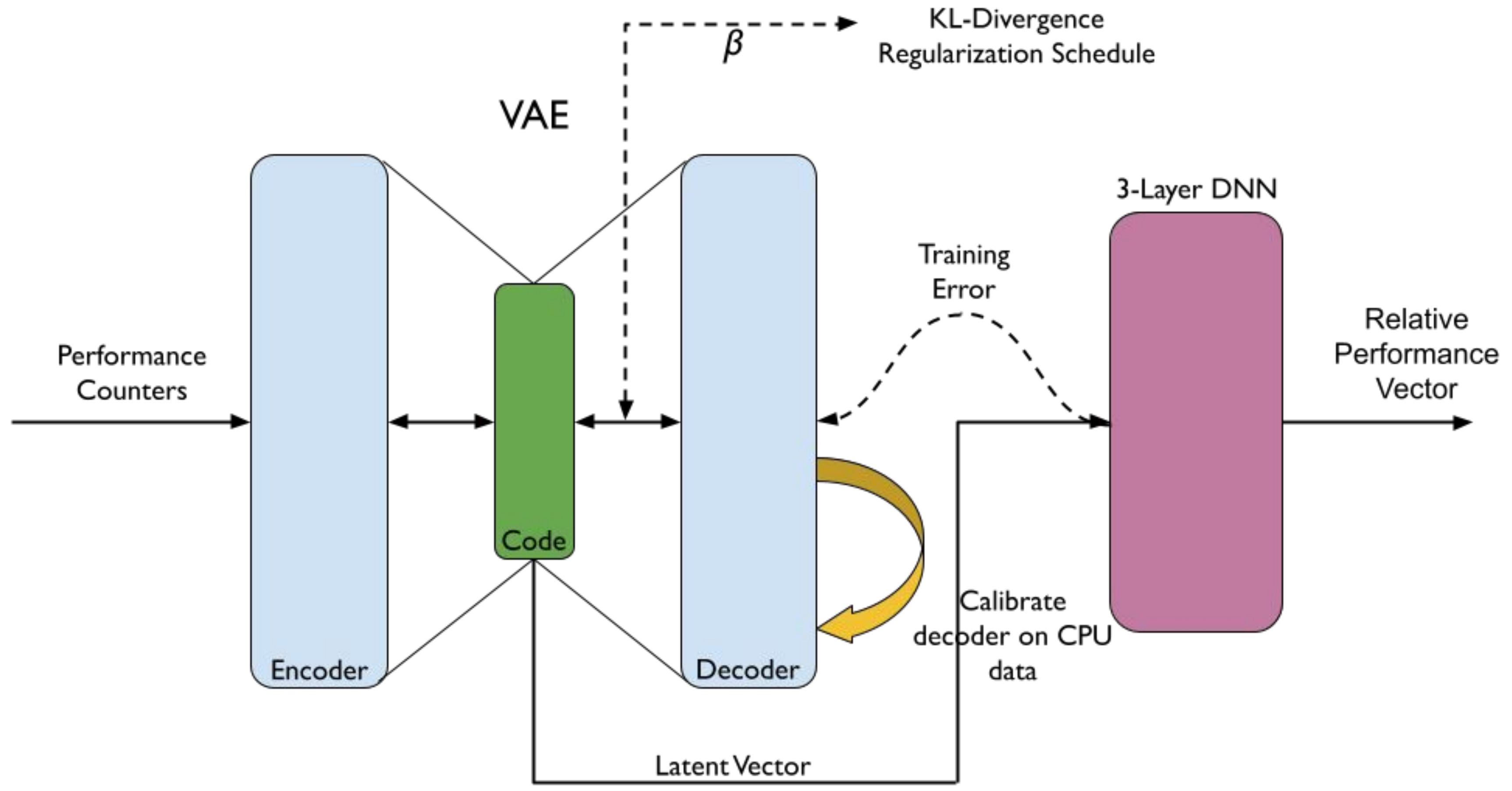
Data locality & memory
- Measure data intensivity
- Locality is good measure of potential GPU performance

Control flow & parallelism
- Measure instruction level parallelism
- Divergent behavior is bad for GPU performance

IO
- Relative amount and frequency of IO actions

Counters
Instructions
Ratio Int Arith. to Total Instr.
Ratio Float Arith. to Total Instr
Ratio Mem. to Total Instr.
Ratio Caches Misses to Hits
Page Levels
Page Faults
GPU2CPU & CPU2GPU Bandwidth
Ratio Control to Total Instr.
Ratio Indep. Instr. to Total Instr.
Ratio Branch Misses to Branch Instr.
Threads / Streams
Warp Blocks
IO Bytes Read/Written
IO File Descriptors Opened

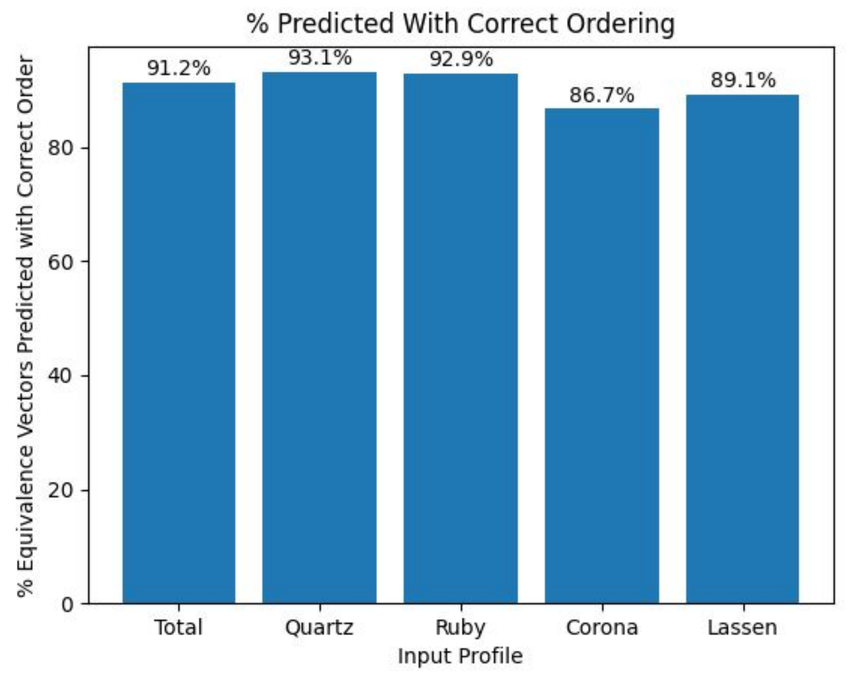
Model

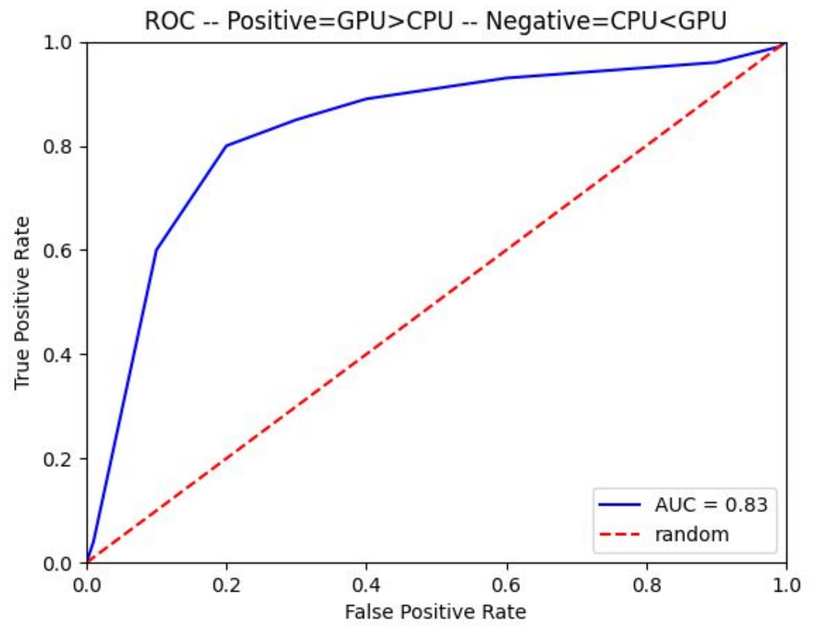


- Performance counter data is used to train the VAE
- Decoder is *calibrated*[3] on CPU only data set first for better starting weights
- Update parameter weighted by KL-divergence regularization schedule
- Downstream task of generating relative performance vector is accomplished with 3-layer DNN
- Decoder loss is weighted by DNN training error for some epochs
- Output values are normalized to integer values

Training Results

- Models are trained with 80-20 cross-validation split
- DNN regressor has R^2 value of ≈0.81. Baseline RandomForest regressor has R^2 value of ≈0.68
- ≈91% of predicted relative performance vectors are in the correct order
- Predictions are often better using CPU profiles





- ≈80% of data set has GPU as faster than CPU
- Distinguishing between relative performance vectors where GPUs are faster than CPUs and vice-versa we can see the model still maintains high sensitivity.
- The red line shows the naive baseline of random guessing

Conclusion and Future Work

We are able to train a deep learning model to generate relative performance vectors for a set of resources based on profiled counter data.
In future work we will:
- Explore transfer learning to retrain the model for new resource sets with few samples
- Include resource property counters in input so model can handle unseen architectures
- Utilize performance modelling in multi-system job scheduler

Acknowledgements

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344 (LLNL-POST-838517)

References

[1] https://e4s-project.github.io/
[2] https://proxyapps.exascaleproject.org/
[3] Rybkin, Oleh, *et al*, "Simple and Effective VAE Training with Calibrated Decoders," in MLR '21.