Surrogate ML/AI Model Benchmarking for FAIR Principles' Conformance

Piotr Luszczek
EECS, University of Tennessee
Knoxville, TN, USA

Cade Brown
EECS, University of Tennessee
Knoxville, TN, USA

2022 IEEE High Performance Extreme Computing Conference (HPEC) | 978-1-6654-9786-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/HPEC55821.2022.9926401

***Abstract*—We present benchmarking platform for surrogate ML/AI models that enables the essential properties for open science and allow them to be findable, accessible, interoperable, and reusable. We also present a use case of cloud cover modeling, analysis, and experimental testing based on a large dataset of multi-spectral satellite sensor data. We use this particular evaluation to highlight the plethora of choices that need resolution for the life cycle of supporting the scientific workflows with data-driven models that need to be first trained to satisfactory accuracy and later monitored during field usage for proper feedback into both computational results and future data model improvements. Unlike traditional testing, performance, or analysis efforts, we focus exclusively on science-oriented metrics as the relevant figures of merit.**

I. Introduction and Background Information

Four foundational principles were initially announced for scientific data management and stewardship. Specifically, they were: Findability, Accessibility, Interoperability, and Reusability [1]. Soon they became the rallying cry that spurred a number of reproducibility initiatives across the scientific community [2] with community-wide development of accompanying services for supporting researchers in providing better exposure.

We present Surrogate AI Benchmarking Applications' Testing Harness (SABATH) that abides by the principles of findable, accessible, interoperable, and reusable (FAIR) manifesto [1]. Our software platform combines the four foundational principles with benchmarking functionality to test surrogate machine learning (ML) models, that were designed as a state-of-the-art approach of enhancing *ab initio* scientific models with data-driven models and part of Surrogate Benchmarking Initiative (SBI). Unlike other performance-focused evaluation efforts, we select our metrics of merit so that they are much closer aligned with the applications' scientific area and get as close as possible to quantitative research discovery goals. Thus, we consider metrics like FLOPs or bytes-transferred-per-second to be too low-level. In other words, they represent hardware-level view of performance but we aim at the science-level evaluation metrics. At the same time, the solver- or method-specific measurements are very useful intermediaries but they still do not reflect end-goals of the research field. An example from the Finite Element Method (FEM) would be the rate of elements-per-second but this may still be somewhat detached from the users' needs because the exact meaning of an element depends, e.g., on the order of the discretization scheme. Instead, we use such metrics as mid-way proxies leading towards physical world quantities such as days-per-nano-second of Molecular Dynamics (MD) simulation time or square kilo-meters of cloud cover identified per second. We describe in detail the latter example as our sample use case data set and model presented in Section §II. Our platform's design and implementation details are included in Section §III.

Finally, we are driven by the following goals for our SABATH platform:

- *Conform* with the FAIR principles for open science and data.
- *Allow* flexibility to support wide range of science domains and AI/ML models they require.
- *Provide* exhaustive reporting about the collected solutions including the science domain, models, data sets, results, software stack, and hardware platform.

II. Test Case: Satellite Cloud Cover Analysis

SMCEFR: Sentinel-3 Satellite is a dataset consisting of 1024×1024 red-green-blue (RGB) images generated from the Sentinel-3 satellite via the Ocean and Land Color Instrument (OLCI). In order to facilitate the competition challenge we applied simplifications to the original and reduced the data volume. This preprocessing created a subset of the data and produced RGB images that are easier to analyze with model prototypes using tools like Python and NumPy/Pillow modules or Tensorflow or OpenCV in C++ and many other modern image processing software. Our goal was to give the users the ability to rapidly test and visualize their algorithms. Furthermore, we posed challenge questions to guide the potential research directions for the users could explore to present their own insights. In particular, we encourage approaches originating from computer vision, numerical programming, and machine learning.

A. Multi-Spectral Sensor Data Source

Satellite data is important for many environmental sciences, as they give scientists a bird's eye view of large areas of the Earth. Modern orbital sensor allow them to collect additional research input remotely with high precision in order to employ data-intensive analysis workflows. For example, the Sentinel-3 satellite collects images that are used by scientists for a wide variety of research tasks, such as monitoring ocean and land surface temperatures, bootstrapping models for weather

This work was supported by the U.S. Department of Energy, Office of Science, ASCR under Award Number DE-SC0021419.

forecasts or atmospheric conditions' prediction, and modeling and predicting climate change.

However, in its original form, the data size and the format are unwieldy for rapid data exploration. Our goal was to remedy this by simplifying the dataset. To this end, it has been prepossessed to use a more approachable format described next.

The original data for this dataset came from the Copernicus Hub [3], which hosts large-scale historical records allowing access to multi-sensor data. However, the hub's access policy requires a complicated workflow to query, filter, download, and pre-process the available datasets. We recognize the need of the full applications to have access to the extra geospatial and spectral information. While this could be extremely useful, for the purposes of a data challenge competition with a broad appeal, a smaller and more manageable dataset is desirable. In the following, we summarize the specifications that guided the creation of this dataset, which we will call SMC's Earth Full Resolution (SMCEFR):

- Data from the Ocean Land and Color Instrument (OLCI) on the Sentinel-3 was queried.
- Data was considered from the OLCI's Earth Full Resolution (EFR) data product [4].
- Only the scans visualizing a part of the North or Sourth American continents were used, to provide a more consistent dataset.
- Data from the selected time periods and only in the past 3 years were used.
- All images are center-cropped to the exact 1024×1024 resolution, or were discarded if only a smaller image was available. This naturally avoided the need for normalization and resizing.
- Multispectral data (20+ bands) are reduced to standard visible RGB representation to reduce the dimensionality.
- The images with the most extreme cases of sparsity or low-entropy were discarded. For example, images that are completely white or blue which represented dense cloud cover or ocean water, respectively.

In the end, we produced smaller scale datasets in specific sizes, and encoded as traditional RGB images (PNG image format) with a 1024×1024 size. Figure 1 presents a sample of 18 images from our reduced data set. These are available from our GitHub release page [5], and are meant to be easily accessible using any of the following software packages or workflows:

- Python with NumPy/SciPy, Pillow, Pandas modules.
- Python with OpenCV module.
- Python with Tensorflow or PyTorch.

B. Benchmarking Targets

To motivate the potential analysis methods for the dataset, we present below sample challenge questions and directions that explore the prospective ideas in data science, computer vision, and machine learning.

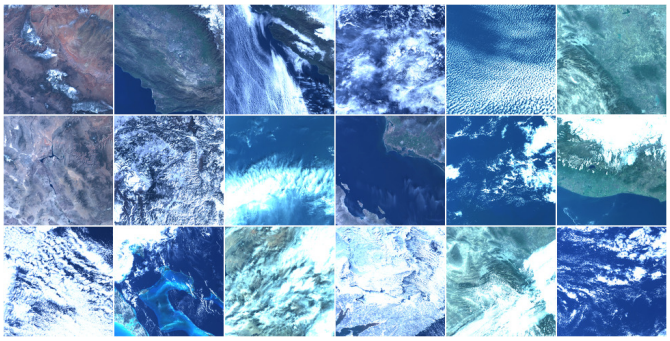

Fig. 1. A sample of 18 images from the `smcefr-mini` dataset.

1) Cloud Identification: What methods can be used to segment and identify the regions of the images that are partially or completely obstructed by the cloud cover?

Potential approaches include:

- Computer Vision: thresholding, K-means clustering, and edge detection should allow for a straightforward identification of clouds as a mask or segmentation map.
- Machine Learning: many methods could be used, for example, the U-Net architecture [6]. Most methods would require some ground truth dataset, i.e., using supervised machine learning. Also, unsupervised clustering followed by a programmer-assisted classification on the learned clustering patterns could be used as well. i.e., using unsupervised machine learning methods [7].

2) Sensor Noise Removal: Consider a case where the sensor data is incomplete, of varied quality, or totally degraded. This may involve finding a way to take partially damaged and/or noisy image data in order to reconstruct something "closer" to the original sensor reading. Clearly, various measures of "closeness" need to be carefully to considered. For example, whether useful existing metrics, including Peak Signal to Noise Ratio (PSNR) and Mean Square Error (MSE), can potentially be used to compare performance of various processing methods.

For this task, we suggest creating copies of the *ground truth* dataset, and then adding a random amount of noise, followed by processing the copy as input.

- Computer Vision: standard denoising techniques such as Non-local Means Denoising, available in OpenCV [8], would produce usable results quite quickly.
- Machine Learning: using unsupervised learning techniques, with the noise-augmented copies, one could use any number of optimization approaches. One example using Tensorflow that uses the concepts of autoencoders and latent space is available online [9]–[12]. Note that optimizing neural networks by directly maximizing its Evidence Lower Bound (ELBO) turns an Autoencoder into a Variational Autoencoder (VAE) [13], which treats representation learning as an inference problem. In addition to compressing the data and learning representation, VAE can be used to generate data, and have shown success in generating several kinds of such data

978-1-6654-9786-2/22/$31.00 ©2022 IEEE

including music [14], speech and handwriting [15], arithmetic expressions and molecular structures [16], and sentences [17].

3) Image Compression: To preserve data integrity, the dataset is provided as PNG, in the lossless data compression format mode. However, for a variety of purposes, it would be useful to allow a small amount of error in exchange for a substantial size reduction[1]. What can research methods be used to save space while keeping the best quality possible?

- Numerical Programming: software libraries exist for HPC and scientific workloads, for example `zfp` [18] or `sz` [19], FFT, cosine transform, and wavelet-based methods [20]–[22].
- Machine Learning: Using Generative Adversarial Networks (GANs), some models have been trained to perform image compression $\approx 4\times$ times better than JPEG, such as HIFIC [23], which comes with included pre-trained models and is freely available to download. For code samples and their documentation, see the Tensorflow Compression code [24]. Additionally, a custom autoencoder could be trained specifically for the task, so that the compression is optimized for this dataset, and potentially allowing for even more fine-tuned performance in terms of the compression ratio.

III. SABATH Platform Details

The Cloud Cover use case presented in Section §II served as practical example to elucidate the broad range of choices that FAIR-conforming platform like SABATH has to address in a comprehensive manner. On the one hand, it needs to show the commitment to its founding principles. And on the other hand, it should remain flexible enough to address the variety of ML/AI methods. This flexibility is essential to keep up with the rapid evolution of the ML/AI field: from basic unsupervised ML such as probably approximately correct (PAC) methods [25] to DNNs or CNNs (neural networks) to attention-based transformers to deep reinforcement learning with ever changing policy representations.

A. Design Goals and Their Practical Realizations

SABATH's website serves as a public repository for browsing and search of scientific ML/AI models, datasets, and performance runs' results. In practical settings, this requires rich meta-data collections associated with the collected artifacts for both visual display but also to provide semantically relevant search functionality. The expected outcome is to go beyond enabling a simple website search (either directly or indirectly with externally crawled indexes either through commercially-oriented services or non-profit projects such as Way Back Machine) for, say, model features, authors affiliations, etc. Also, simple keyword-based search is insufficient and we initially provide both semantic and topic modeling augmented with multi-dimensional view based on tags and labels, with further down the line integrating deep Natural Language Processing (NLP) language models distilled for the purposes of expressing semantics of scientific models.

The unspoken rule followed by modern ML/AI models is that the their underlying frameworks are the ease to use: they support high-level languages like Python or R. This translates directly into SABATH's goal of quick-and-easy installation requiring ideally none or at most minimal instructions for model's install-test-deploy cycle. This requires a single entry-point into a particular functionality, e.g., a single locator for model's data access or a single function or script for invoking an inference with a given model on the user's input data. Also, direct integration with users' environments is part of the user-perceived ease-of-use: simplicity of accessing the platform and obtaining the relevant results is essential. We further expand on the multi-framework support and integration below. In terms of the installation process, SABATH both looks and feels as yet another ML/AI model running on a dataset of choice inside the context of a typical scientific ML environment: both at scale but also with a familiar interface of a script (be it a Unix shell, a Python interpreter, Julia's function, or MATLAB window) downloaded on demand with a single click for a particular test run or generated just-in-time for a new combination of input data, inference model, and hardware configuration. These details of the run are encapsulated inside a single file, e.g., `dataset-model-hardware.sh` or `dataset-model-hardware.py`. We provide typical parameters to differentiate operational modes of the user's system. Thus allow a more staged execution of tests especially when they are requested explicitly for the impromptu invocation, e.g., `python3 model.py [ setup | fetch | train | run | benchmark | verify | report ]` and so on. This may change to accommodate the particular model's vagaries: some may need training while others are only meant for inference.

Combining the goal of ease-of-use with the goal of *portable performance* became much harder to achieve. Still, the stated goal is for our platform to work on the relevant hardware systems including mainline CPUs and accelerators such as GPUs, DPUs, TPUs and FPGAs. To some extent, these are deployed in industrial or academic settings for ML/AI workflows and/or computational science and may be combined to cover a wide variety of applications. We achieved this goal through delegation of complexity down the software stack to leverage builtin capacity of modern ML/AI framework stacks. For example, accessing a TPU accelerator [26], a Google-specific processing hardware, is done with code generated by JAX [27] module invoked by Tensorflow [28], [29] for efficient usage of the processing units. We proceed along a similar usage path to access CUDA's cuDNN [30] to enable executing a highly optimized code on NVIDIA's GPU accelerators. This approach allowed us to also support HPC' supercomputer resources and AMD HIP/ROCm middleware and Intel's DPC++ and oneAPI libraries' ecosystem. Unfortunately, some models may be framework-specific and this would naturally limit their portability range but this is beyond the scope of our project due

[1]For example, serving images on the web that are compressed with a lossy method enables more responsive websites.

978-1-6654-9786-2/22/$31.00 ©2022 IEEE

to a large effort required to re-write and we defer to automated transpilation efforts to bridge the disparate software stacks. Those might still be lacking at the moment. Cross-framework efforts such as Keras may offer some relief in this respect and are likely improve in the near future.

Related to performance portability is *data portability*, which postulates the following goal for SABATH: provide broad support for data formats that are used for the relevant models' structure, their inputs, and output results either from training or inference iterations. This fulfills the accessibility requirement from the FAIR principles. In a similar fashion to performance, we leverage the existing software frameworks and their respective data conversion capabilities that enable for the majority of models' data to be easily converted into a common format and subsequently translated to the format that is required for proper working of the tested model.

IV. Meta-Data: Schema Considerations

Full definition of the SABATH schema is out of scope for this writing. We only give a generic overview that is enforced by the initial JSON schema meta-document. To address one of SABATH's main goals, the schema enables extensive reporting space about the science problems, collected solutions, training, verification, and validation. The main sections of the meta-data schema includes the science domain, models, data sets, results, software stack, and hardware platform.

V. Profiling and Logging Interface Options

Profiling interface, while not required, it needs to be possible, and better yet, ease to use and accessible in a portable manner. To that end, Figure 2 shows a sample integration with TensorBoard – a preeminent graphical profiling of TensorFlow ML models and we show how profiling is enabled for one of the SABATH's models based on TensorFlow.

In general, SABATH is not limited by a single ML/AI software stack. To that end, we envision a much broader support for profiling, logging, and tracing and we included the flexibility support a variety of related data formats including human readable and binary ones. The former includes simple JSON schema and the latter is exemplified by self-describing HDF5 storage format [31] or OTF2 [32] geared specifically towards large scale tracing that may produce large data sets of events. We also specify a minimum requirements for logging relevant information so that the behavior of a model can be recorded during the benchmark run. This will enable easy manipulations with packages such as Pandas [33].

VI. Conclusions and Future Work

We presented the SABATH benchmarking platform whose main design goal is to enable extensive evaluation of surrogate ML/AI models, that proliferate in order to assist computational science. Our platform is meant to follow closely the essential properties that are critical for open science and allow them to be findable, accessible, interoperable, and reusable as specified in the FAIR manifesto [1]. We also presented a use case model for cloud cover modeling, classification, analysis, and

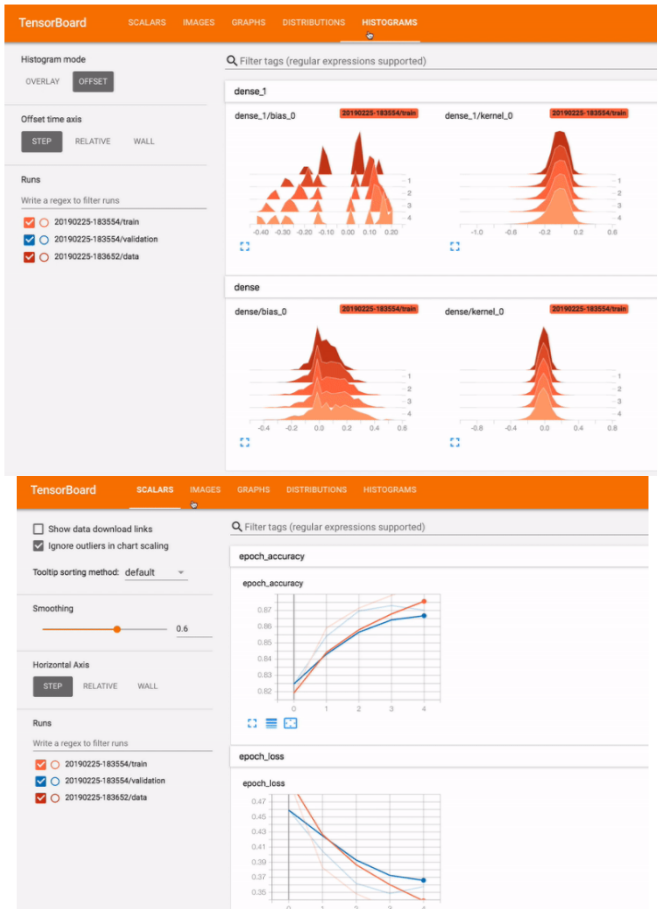

Fig. 2. TensorBoard window showing bias values during a profiling of SABATH (top) and model's accuracy during the training epochs (bottom).

experimental testing, which uses a large dataset consisting of multi-spectral satellite sensor readings. We leveraged this particular use case's evaluation to highlight the large space of potential choices faced in designing our surrogate benchmarking effort. With the flexible resolution of this selection space of supported formats, libraries, and hardware systems, we obtained a useful testing life cycle while supporting the important scientific workflows enhanced with data-driven models. Even if they need to be first trained to a satisfactory levels of accuracy and later monitored during field usage for proper feedback into both computational results and future data model improvements. We managed to go beyond the traditional testing, performance, or analysis efforts, and we focused exclusively on science-oriented metrics as the relevant figures of merit.

Our goal is to continue including additional models and test SABATH to ensure compatibility across the usage scenarios relevant for ML/AI-enriched computational science. This will help us to increase the number of available models that then may be considered compliant with FAIR principles and thus better serve science goals.

978-1-6654-9786-2/22/$31.00 ©2022 IEEE

REFERENCES

[1] M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.-W. Boiten, L. B. da Silva Santos, P. E. Bourne, J. Bouwman, A. J. Brookes, T. Clark, M. Crosas, I. Dillo, O. Dumon, S. Edmunds, C. T. Evelo, R. Finkers, A. Gonzalez-Beltran, A. J. Gray, P. Groth, C. Goble, J. S. Grethe, J. Heringa, P. A. 't Hoen, R. Hooft, T. Kuhn, R. Kok, J. Kok, S. J. Lusher, M. E. Martone, A. Mons, A. L. Packer, B. Persson, P. Rocca-Serra, M. Roos, R. van Schaik, S.-A. Sansone, E. Schultes, T. Sengstag, T. Slater, G. Strawn, M. A. Swertz, M. Thompson, J. van der Lei, E. van Mulligen, J. Velterop, A. Waagmeester, P. Wittenburg, K. Wolstencroft, J. Zhao, and B. Mons, "The FAIR guiding principles for scientific data management and stewardship," *Scientific Data*, vol. 3, no. 160018, Mar. 2016. [Online]. Available: https://doi.org/10.1038/sdata.2016.18

[2] W. Keyrouz and M. Mascagni, "Numerical reproducibility at exascale (nre2015)," 2015, part of SC15, Austin, TX; www.nist.gov/news-events/events/2015/11/numerical-reproducibility-exascale-nre2015.

[3] "Copernicus open access hub," 2021, https://scihub.copernicus.eu/.

[4] "Sentinel-3 OLCI EFR data product description," 2021, https://sentinels.copernicus.eu/web/sentinel/technical-guides/sentinel-3-olci/level-1/fr-or-rr-toa-radiances.

[5] "Github dataset download of `smcefr-mini` (1GB)," 2022, https://github.com/cadebrown/smcefr/releases/download/v1.0.0/smcefr-mini.tar.gz.

[6] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: convolutional networks for biomedical image segmentation," *LNCS*, vol. 9351, pp. 234–241, 2015, https://arxiv.org/abs/1505.04597.

[7] D. Greene, P. Cunningham, and R. Mayer, "Unsupervised learning and clustering," in *Machine Learning Techniques for Multimedia: Case Studies on Organization and Retrieval, Chapter: 3*, M. Cord and P. Cunningham, Eds. Springer, January 2008, pp. 51–90.

[8] "OpenCV: image denoising," https://docs.opencv.org/3.4/d5/d69/tutorial_py_non_local_means.html.

[9] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio, "Why does unsupervised pre-training help deep learning?" *Journal of Machine Learning Research*, vol. 11, pp. 625–660, 2010.

[10] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *CoRR*, vol. abs/1502.03167, no. 1502.03167, 2015. [Online]. Available: arxiv.org/abs/1502.03167

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, no. 1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[12] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *CoRR*, no. 1312.6114, 2013. [Online]. Available: arxiv.org/abs/1312.6114

[13] ——, "Auto-encoding variational Bayes," in *2nd International Conference on Learning Representations, ICLR, Conference Track Proceedings*, Banff, AB, Canada, April 2014.

[14] O. Fabius, J. R. van Amersfoort, and D. P. Kingma, "Variational recurrent autoencoders," in *3rd International Conference on Learning Representations, ICLR, Workshop Track Proceedings*, San Diego, CA, USA, May 2015.

[15] J. Chung, K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio, "A recurrent latent variable model for sequential data," in *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems*, Montreal, Quebec, Canada, December 2015, pp. 2980–2988.

[16] M. J. Kusner, B. Paige, and J. M. Hernández-Lobato, "Grammar variational autoencoder," in *Proceedings of the 34th International Conference on Machine Learning, ICML*, vol. 70, Sydney, NSW, Australia, August 2017, proceedings of Machine Learning Research, pp. 1945-1954, PMLR.

[17] S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Józefowicz, and S. Bengio, "Generating sentences from a continuous space," in *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL*, Berlin, Germany, August 2016, pp. 10–21, ACL.

[18] P. Lindstrom, "Fixed-rate compressed floating-point arrays," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 2674–2683, December 2014. [Online]. Available: https://computing.llnl.gov/projects/zfp

[19] S. Di and F. Cappello, "Fast error-bounded lossy HPC data compression with SZ," in *IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2016, pp. 730–739.

[20] M. Dorobantu and B. Engquist, "Wavelet-based numerical homogenization," *SIAM Journal on Numerical Analysis*, vol. 35, no. 2, pp. 540–559, 1998.

[21] H. F. Bucher and L. C. Wrobel, "A novel approach to applying fast wavelet transforms in boundary element method," *Electronic Journal of Boundary Elements*, vol. 2, pp. 187–195, 01 2002.

[22] L. Ebrahimnejad and R. Attarnejad, "A novel way of using fast wavelet transforms to solve dense linear systems arising from boundary element methods," *Engineering Computations*, vol. 26, no. 5, pp. 483–499, 2009. [Online]. Available: doi.org/10.1108/02644400910970167

[23] F. Mentzer, G. D. Toderici, M. Tschannen, and E. Agustsson, "High-fidelity generative image compression," in *Advances in Neural Information Processing Systems*, vol. 33, 2020. [Online]. Available: https://hific.github.io/

[24] "Tensorflow source code: Compression," 2020. [Online]. Available: github.com/tensorflow/compression/tree/master/models/hific

[25] L. G. Valiant, "A theory of the learnable," *Communications of the ACM*, vol. 27, pp. 1134–1142, 1984.

[26] N. P. Jouppi, C. Young, N. Patil, D. A. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers, R. Boyle, P. Cantin, C. Chao, C. Clark, J. Coriell, M. Daley, M. Dau, J. Dean, B. Gelb, T. V. Ghaemmaghami, R. Gottipati, W. Gulland, R. Hagmann, R. C. Ho, D. Hogberg, J. Hu, R. Hundt, D. Hurt, J. Ibarz, A. Jaffey, A. Jaworski, A. Kaplan, H. Khaitan, A. Koch, N. Kumar, S. Lacy, J. Laudon, J. Law, D. Le, C. Leary, Z. Liu, K. Lucke, A. Lundin, G. MacKean, A. Maggiore, M. Mahony, K. Miller, R. Nagarajan, R. Narayanaswami, R. Ni, K. Nix, T. Norrie, M. Omernick, N. Penukonda, A. Phelps, J. Ross, A. Salek, E. Samadiani, C. Severn, G. Sizikov, M. Snelham, J. Souter, D. Steinberg, A. Swing, M. Tan, G. Thorson, B. Tian, H. Toma, E. Tuttle, V. Vasudevan, R. Walter, W. Wang, E. Wilcox, and D. H. Yoon, "In-datacenter performance analysis of a tensor processing unit," *CoRR*, vol. abs/1704.04760, 2017. [Online]. Available: http://arxiv.org/abs/1704.04760

[27] R. Frostig, M. Johnson, and C. Leary, "Compiling machine learning programs via high-level tracing," in *SysML Conference*, Stanford, CA, USA, February 15-16, 2018.

[28] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, and et al., "Tensorflow: a system for large-scale machine learning," in *OSDI*, vol. 16, 2016, pp. 265–283.

[29] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[30] "Cudnn." [Online]. Available: https://developer.nvidia.com/cudnn

[31] M. Folk, G. Heber, Q. Koziol, E. Pourmal, and D. Robinson, "An overview of the HDF5 technology suite and its applications," in *Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases*. ACM, 2011, pp. 36–47.

[32] D. Eschweiler, M. Wagner, M. Geimer, A. Knüpfer, W. E. Nagel, and F. Wolf, "Open Trace Format 2: The next generation of scalable trace formats and support libraries," in *Paralle Computing Conference (PARCO)*, vol. 22, 2011, pp. 481–490.

[33] W. McKinney, "Pandas: a foundational Python library for data analysis and statistics," *Python for High Performance and Scientific Computing*, vol. 14, 2011.

978-1-6654-9786-2/22/$31.00 ©2022 IEEE