Report on the Results of the Paper Reproduction: "Maximizing Insights, Minimizing Data: I/O Time Prediction Using Transfer Learning"

Dlyaver Djebarov

Data Landscape

Blue Waters

The Blue Waters dataset was downloaded using Globus[1], resulting in numerous archives containing Darshan logs from September 2017 to December 2019. Parsing these files required installation of Darshan version 3.4.1. The original script for organizing the logs into separate CSV files did not delete unpacked files after processing, which led to high memory usage. The script was modified to remove intermediate files after writing the logs to CSV, thereby reducing storage demands.

After merging all CSV files into a single dataset, the data was prepared for training. The final dataset contained 851,195 raw logs, compared to 822,790 logs in the dataset used in the original study (3.5% more). Due to differences in the filtering methods used by Voss et al. and Povaliaiev et al., separate Jupyter notebooks were created to implement each approach:

1. Blue_Waters_filter_data.ipynb: Removes unnecessary columns, rows containing only zeros, and applies IQR filtering. Used to replicate the preprocessing of Povaliaiev et al.
2. Blue_Waters_filter_data_by_nprocs.ipynb: Filters out records with numbers of processes other than 4, 16, 48, 64, 144, or 240. Used to replicate Povaliaiev et al.
3. Blue_Waters_filter_data_Voss.ipynb: Removes unnecessary columns, extracts application names, eliminates rows with only zeros, and applies negative outlier filtering.
4. Blue_Waters_remove_time.ipyn: Removes all time-related columns except for `POSIX_TOTAL_TIME`.
5. Blue_Waters_remove_dups.ipynb: Identifies duplicate rows.
6. Blue_Waters_compute_concurr_procs.ipynb: Detects processes running in parallel.
7. Blue_Waters_compute_MAE.ipynb: Computes MAE.

After processing, 820,701 logs remained, while Voss et al. reported 792,311 logs (3.5% fewer). The study presents a scatter plot of `POSIX_TOTAL_TIME` versus `POSIX_READ` and `POSIX_WRITES`, with color indicating `POSIX_BYTES_READ` and `POSIX_BYTES_WRITTEN`; darker colors represent larger values. The same plots were reproduced using the current dataset (Figure 1, Figure 2, Figure 3), and the results appear nearly identical. This confirms the highly similar nature of the training data, despite a slight difference in dataset size.

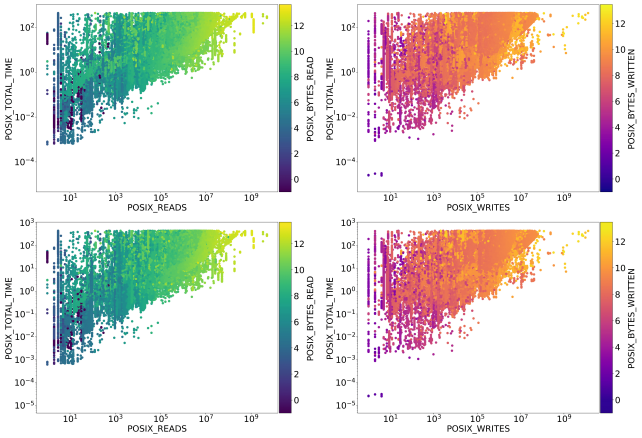


Figure 1: IQR filtering. Top: original study. Bottom: reproduced from the available dataset.

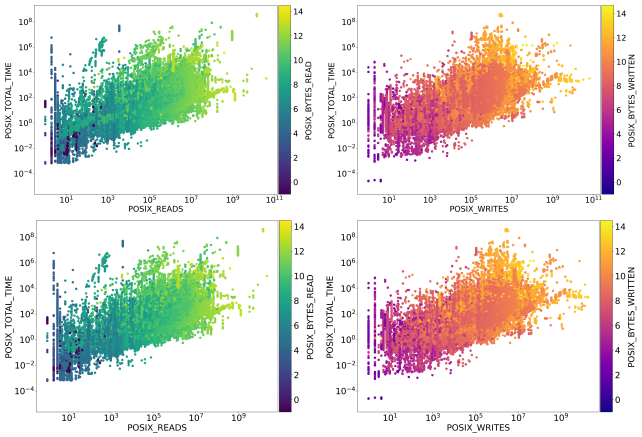


Figure 2: No IQR filtering. Top: original study. Bottom: reproduced from the available dataset.

[1] https://bluewaters.ncsa.illinois.edu/data-sets

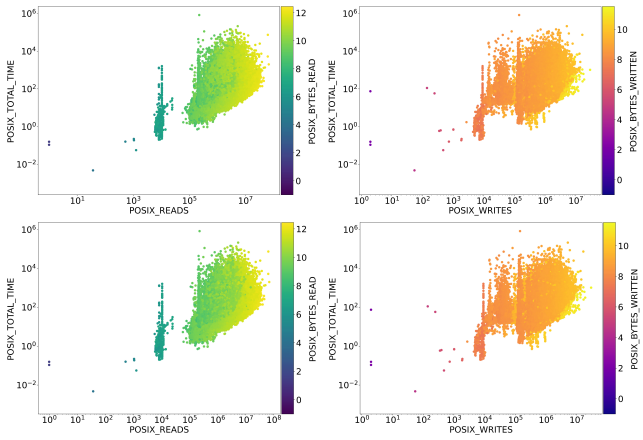


Figure 3: No IQR filtering, restricted to SpEC benchmark application. Top: original study. Bottom: reproduced from the available dataset.

Theta

According to Voss' thesis, the Theta dataset from 2021, divided into 12 parts, was used as the dataset for fine-tuning. It is likely that each part corresponds to one month. The dataset was located in the ALCF Data Catalog[2], however, it contained 187 columns, and the feature names differed significantly from those in the Blue Waters dataset. Multiple attempts to reformat the dataset to match the required structure were unsuccessful. Consequently, a dataset from the Voss GitLab repository was used instead, containing 218,111 records. For comparison, the CSV extracted from the ALCF Data Catalog contained 226,689 records, only 3.8% more. Given the small difference, the dataset from the Voss GitLab repository was used. After removing unnecessary columns (such as mean, error, and `POSIX_TOTAL_TIME_predicted`) a clean dataset was obtained.

Duplicate Sets and Lower MAE Bound

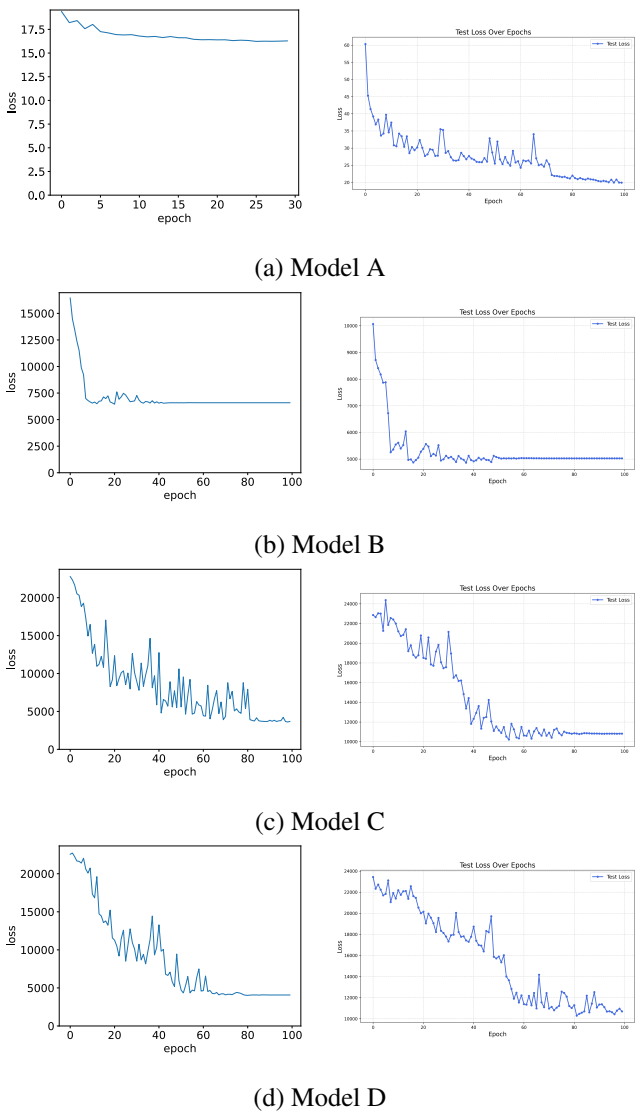
The MAE calculation resulted in 14.96% for Blue Waters and 15.71% for Theta, compared to 19.97% and 15.71% reported by Voss et al., respectively.

Base Model Training Results

The experiments were repeated using the same configurations as in the original study. Figure 4 compares the test loss curves from the original study and the reproduced experiments. Additionally, Table 1 summarizes the training results. At this stage, a noticeable difference between the lower-bound MAE and the actual MAE can be observed. In Voss et al., this difference ranged from 0.38% to 6.93% depending on the model. In the current experiments, the range was 3.61% to 5.01%.

As in the original study, models A, B, and C converged within 100 epochs. However, model D required extended training up to 120 epochs to fully converge, achieving a final test loss of 10648.36.

[2]https://reports.alcf.anl.gov/data/



(a) Model A

(b) Model B

(c) Model C

(d) Model D

Figure 4: Test loss over training epochs for each model. Left: from Voss et al. Right: reproduced results.

Transfer Learning Outcomes

To replicate the base model training outcomes, the same scripts as in the original study were used. The result for the electronics structure application is shown in Figure 5a. For this application, 93.75% of the predictions satisfy the order-of-magnitude criterion, compared to 87.25% in the original study.

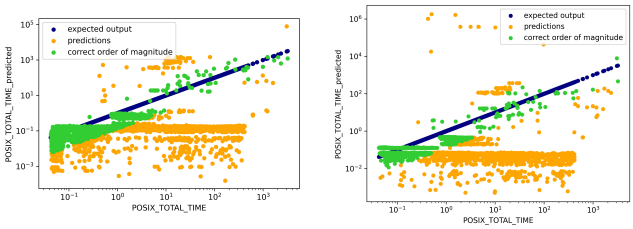
As shown in Figure 5b, the physics application does not satisfy the order-of-magnitude criterion at this stage. None of the predictions matched the correct order of magnitude, whereas the original study reported 1.5%.

The model was then fine-tuned on Theta data. According to Voss, only 2 logs associated with `cp2k.psmp` (physics application) were used for pre-training. Repeating this setup yielded 1.57% correct order-of-magnitude predictions. Increasing the number of logs to 4 improved the percentage to

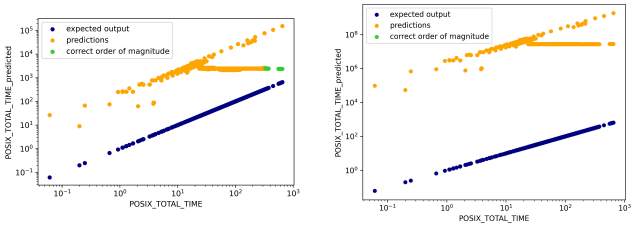
Table 1: Training results on the Blue Waters dataset.

Note: V = results from Voss et al.; D = reproduced results.

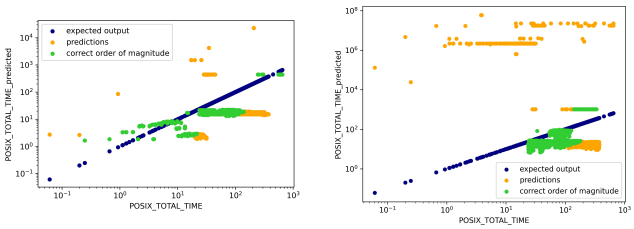
	IQR	Scaler	SpEC	Batch Size	Layers	Loss (V)	Loss (D)	MAE (V)	MAE (D)
A	yes	Standard	all	2048	2048-512-128	16.28	19.93	-	-
B	no	Robust	all	2048	2048-512-128	6589.23	5028.59	21.03%	19.25%
C	no	Robust	50,000	2048	2048-512-128	3675.80	10813.29	26.90%	18.57%
D	no	Robust	50,000	2048	512-256-128	4070.28	10704.26	20.35%	19.99%



(a) Electronics structure application



(b) Physics application without fine-tuning



(c) Physics application with fine-tuning

Figure 5: Predictions $F(x)$ of Model D versus true outputs y for `POSIX_TOTAL_TIME`. Acceptable predictions are shown in green; all others in orange. Blue points indicate perfect predictions ($F(x) = y$). Left: results from Voss et al. Right: reproduced results.

82%, still 5% lower than reported in the original study. The results are shown in Figure 5c.

These records were excluded from the test dataset after fine-tuning, so they were not reused during model evaluation, even though they accounted for less than 0.002% of the total test dataset. A possible explanation for the remaining discrepancy in the physics application results is the randomness in record selection during fine-tuning. This variability can significantly influence the outcome, with prediction accuracy ranging up to 93.71%.

During the evaluation of the fine-tuned model on the Blue Waters dataset, it was also observed that the prediction accuracy for the electronics structure application dropped to 0%, as shown in Figure 6.

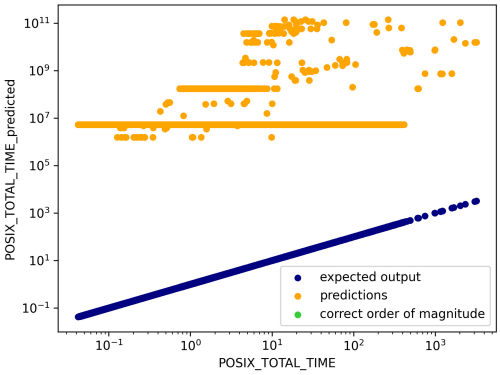


Figure 6: Electronics structure application after fine-tuning.

Model Explainability

Feeding all-zero input values into the model results in a `POSIX_TOTAL_TIME` prediction of 0.8811 for model D and 12.2331 for the fine-tuned model D. These values are inconsistent with the result of 3.0789 reported in the original study.

Initially, difficulties arose because the notebook from the Voss GitLab repository used the model file `Small_net_SpEC_sampling_no_IQR.tar` for Captum analysis. However, the configuration for this model could not be found. As a result, model D was used instead to analyze the contribution of input feature values.

As shown in Figure 7, the range of attribution values is narrow, in contrast to the results from the original study. This indicates that in the reproduced model, `POSIX_READS` has a weaker influence on the output. The model recognizes `POSIX_READS` as relevant but does not treat it as a dominant feature. Its contribution is limited—possibly due to feature scaling or the presence of other more influential features.

Figure 8 compares Integrated Gradients and DeepLIFT. In contrast to the original study, where attribution values ranged between large positive and negative extremes (e.g., $[-263.81, 273.16]$ for `POSIX_STATS`), the reproduced values are strictly non-negative and span several orders of magnitude. Specifically, `POSIX_STATS` ranges from approximately 0 to 6×10^7, and `POSIX_BYTES_WRITTEN` from

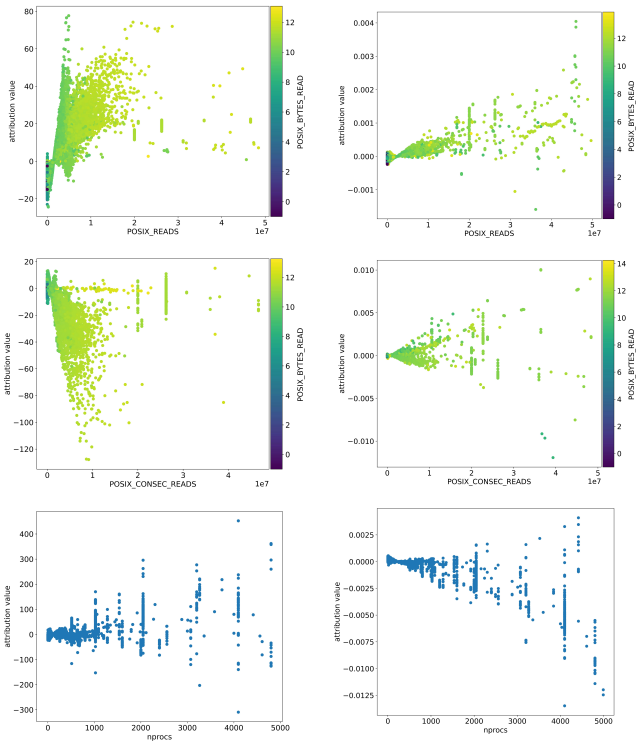


Figure 7: Importance values attributed to a feature as computed by Integrated Gradients, plotted against input values for `POSIX_READS`, `POSIX_CONSEC_READS`, and `nprocs`. Darker coloring indicates higher data density. Left: reproduced results. Right: results from Voss et al.

0 to 9×10^{13}. Nonetheless, both algorithms capture the same patterns as observed in the original study.

Conclusion and Future work

During the reproduction process, a GitHub repository[3] was created containing a collection of notebooks, which makes the workflow transparent and easier to follow. The repository also includes data stored using Git LFS, allowing others to run their own experiments and verify the results.

Overall, the reproduction can be considered successful up to the fine-tuning stage. At that point, differences in the data become more pronounced, indicating that further investigation is required. A potential direction for future work is to convert the available dataset from the ALCF Data Catalog for fine-tuning. However, this is unlikely to significantly impact the results, since according to Voss' thesis, only two logs were used for fine-tuning the provided examples.

The model explainability results also differ from those presented in the original study. This discrepancy is likely due to incorrect data scaling and highlights the need for further experimentation with the available scripts.

It was also observed that there were attempts to use agglomerative clustering to reduce input dimensionality. Since the original study does not mention this technique, it was

[3]https://github.com/arcturus5340/IOTransferLearning

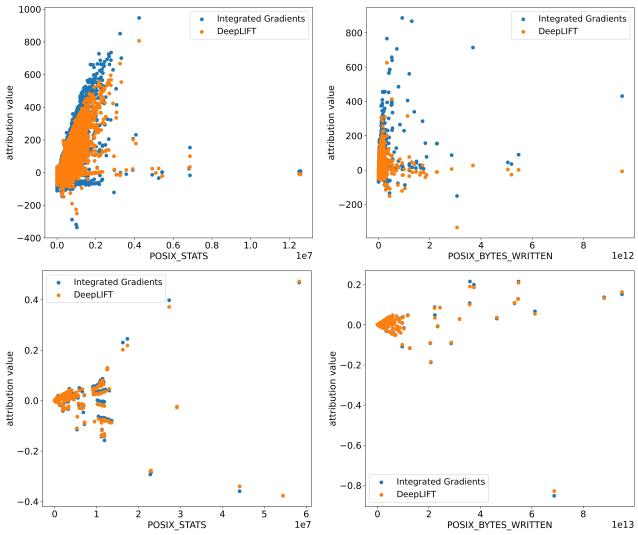


Figure 8: Importance values attributed to a feature as computed by Integrated Gradients (blue) and DeepLIFT (orange), plotted against input values for `POSIX_STATS` and `POSIX_BYTES_WRITTEN`. Top: results from Voss et al. Bottom: reproduced results.

not tested here. Nonetheless, it may offer potential improvements and requires further investigation.