Refactoring TEZip: Integrating Python-Based Predictive Compression into an HPC C++/LibTorch Environment

Mina Yousef †1, Amarjit Singh †2, Kento Sato †2

†1: CIS - Nile University, †2: RIKEN Center for Computational Science

1. BACKGROUND

- **TEZip [1]:** Data compression tool by AI predicting future frames and storing only the delta
  - **Supported AI models**: PredNet and ConvLSTM
- **Need for Speed:**
  - HPC requires **efficient I/O** and **data reduction**
  - **TEZip's Python-based implementation**: TensorFlow, later moved to PyTorch [2], but faced performance bottlenecks (Python's GIL, repeated data conversions)

How can we make TEZip faster ?

2. APPROACH

TEZip: AI-based Data Compression Tool

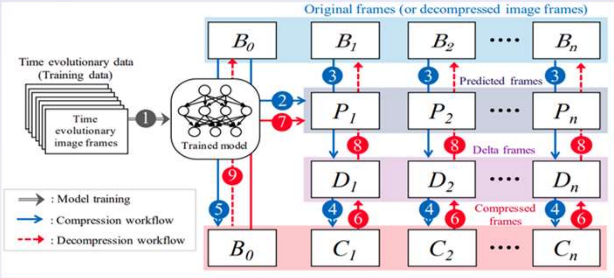

Figure 1 TEzip Architecture

- **Key stages:**
  - **Data creation:** Converts raw images into HDF5/HKL files for training and validation
  - **Model training:** Trains a predictive neural network to accurately predict future frames
  - **Compression:** Predict frames and stores only the difference
  - **Decompression:** Restore original frames by adding deltas back to the predicted frames
- **Error-bound Function**
  - TEZip provides user-specified threshold controlling how much each pixel can deviate from its original value

We accelerates key stages in TEZip by LibTorch

- **Use of LibTorch:** C++ based PyTorch implementation -- free of Python overhead to achieve performance improvement
- **C++/LibTorch Refactor:** Eliminates Python's GIL and overhead, running training, delta storage, and reconstruction on optimized GPU kernels.
- **Seamless Integration:** Seamlessly deployable on multi-core CPU/GPU clusters and remains model-agnostic for additional deep models while maintaining

3. EVALUATION

Benchmark Datasets

Dataset	Domain	Image Size	Train+Val #	Test #	Size
KITTI	Road Footage	375×1242	752	187	813 MB
NYX	Cosmology	512×512	6144	250	2.7 GB
Hurricane Isabel	Weather Simulation	500×500	680	34	1.25 GB

Table 3.1 Datasets: KITTI[4], NYX[5], and Hurricane Isabel[5]

Summary: LibTorch eliminates Python Overhead !!

Category	Hurricane ISABEL	KITTI	NYX
Whole Training Time	4x	1.1x	1.6x
Total Compression Time	11x	13x	13.7x
Total Decompression Time	3.7x	5x	4x

Table 3.2 Performance Summary for each dataset

Key stages in TEZip are accelerated by LibTorch

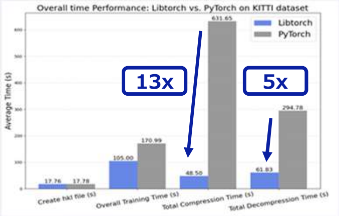

Figure 3.3 Overall Time on KITTI

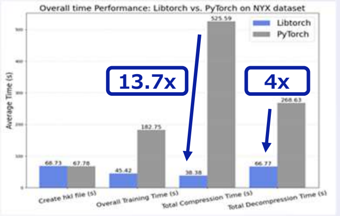

Figure 3.4 Overall Time on NYX

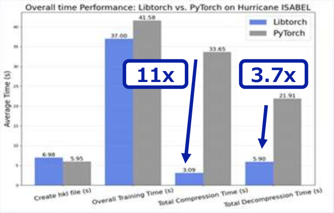

Figure 3.5 Overall Time on Hurricane

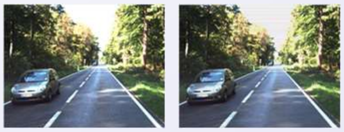

Figure 3.6 Original image (left) vs. Predicted (right) image with 10% error-bound in KITTI

Break-down analysis with error-bound in TEZip

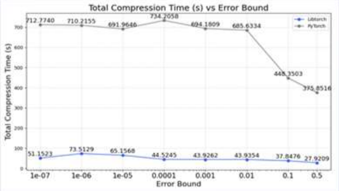

Figure 3.7 Compression Time vs. Error-bound on KITTI

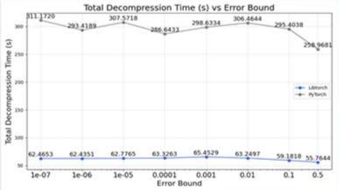

Figure 3.8 Decompression Time vs. Error-bound on KITTI

4. CONCLUSION AND FUTURE WORK

- **Conclusion**
  - **Refactoring TEZip** with C++/LibTorch
  - **Highly efficient**: Training up to **4x** / Compression **~13x** / Decompression **~4x** by eliminating Python overhead

- **Future work:**
  - Evaluate TEZip on additional scientific datasets and new predictive models

Please find Mina's CV at this QR code

[1] Rupak Roy et al., "Compression of time evolutionary image data through predictive deep neural networks", IEEE/ACM CCGrid2021, doi: 10.1109/CCGrid51090.2021.00014, 2021

[2] Akshay Nambudiripad et al "Development of TEZip in PyTorch: Integrating new prediction models into an existing compression framework", SC '24 (Poster), 2024

[3] William Lotter et al., "Deep Predictive Coding Networks for Video Prediction and Unsupervised Learning, URL https://arxiv.org/abs/1605.08104, 2017

[4] Sun, Haodong et al., "Tourism Demand Forecasting of Multi-Attractions with Spatiotemporal Grid: a Convolutional Block Attention Module Model", Information Technology & Tourism , 2023

[5] Andreas Geiger et al, "Are We Ready for Autonomous Driving? The KITTI Vision Benchmark Suite", In 2012 IEEE Conference on Computer Vision and Pattern Recognition, doi:10.1109/CVPR.2012. 6248074.

Acknowledgement: This work has been supported by the COE research grant in computational science from Hyogo Prefecture and Kobe City through Foundation for Computational Science.
This work ("AI for Science" supercomputing platform project) was supported by the RIKEN TRIP initiative (System software for AI for Science).