State Preservation for Deep Learning Applications: The Case of DataStates

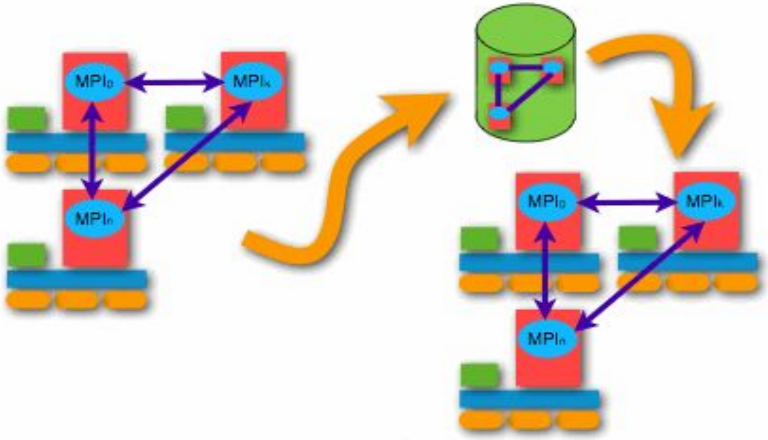
Bogdan Nicolae
Mathematics and Computer Science
Argonne National Laboratory

Argonne NATIONAL LABORATORY

From Checkpointing to State Preservation

More than checkpointing, state preservation maintains a searchable history of intermediate data snapshots (states) and the relationships between them. It aims to make data states ***FAIR*** (findable, accessible, interoperable, reusable).

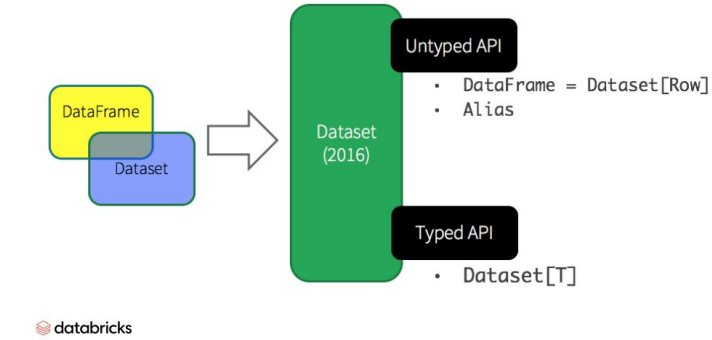
- **Defensive:**
  - Fault tolerance based on checkpoint-restart
- **Administrative:**
  - Suspend-resume (e.g. make room for higher priority jobs)
  - Migration
  - Debugging
- **Productive (where FAIR matters most)**
  - Share intermediate data
  - Find and revisit and/or revert to interesting previous intermediate data
  - Provenance and evolution of data
  - Reproducibility: verify intermediate data

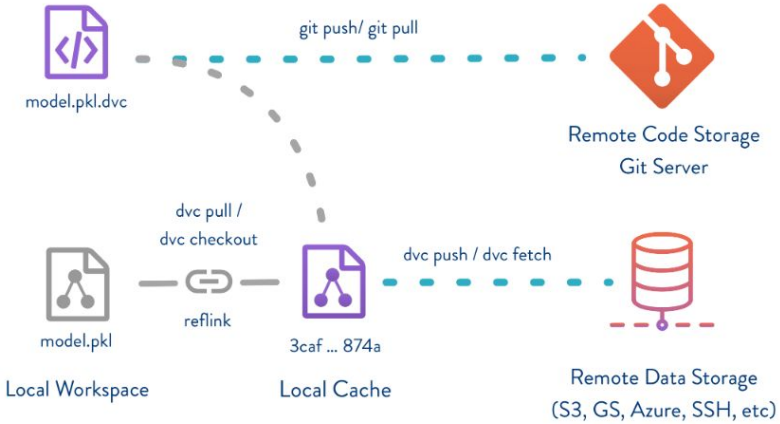


Argonne NATIONAL LABORATORY

State of Art

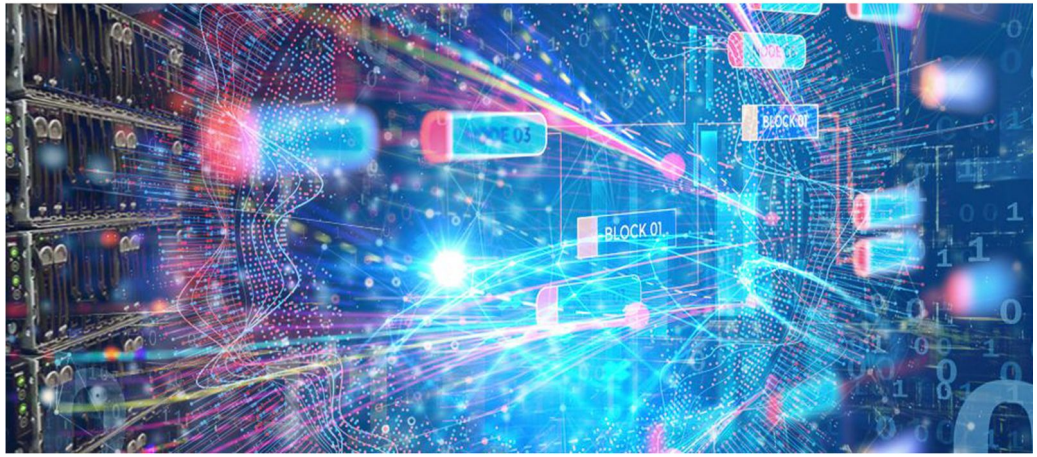
- Checkpointing, KV Stores, Workflow Engines
  - VELOC (Very Low Overhead Checkpointing)
  - ADIOS, DataSpaces, etc.
    - Largely put/get API
    - Supports HPC heterogeneous storage
    - No lineage or FAIR-ness
- Big Data Ecosystem
  - Spark RDDs/DataSets
    - Compute-centric lineage
    - No FAIR-ness (needs DB engines)
    - Limited HPC support
- Versioning
  - Git: source code
  - BlobSeer, DVC: binary data
    - Limited FAIR-ness (id based)
  - DLHub: feature-rich (containers, etc.)
    - External repository (heavyweight)





DataStates: New Project Award

- Works with in-memory annotated distributed datasets
- Annotations as metadata:
  - Identifying: search criteria
  - Actionable: hints, constraints
- Data states as native constructs: datasets are “tagged” to capture immutable snapshots



- **Automated FAIR-ness:**
  - Snapshots automatically recorded in a lineage
  - Versioning semantics
  - Search and navigation capability
- **Performance portability:**
  - Runtime responsible to compose I/O strategy to satisfy hints and constraints
  - Hidden complexity of heterogeneous HPC storage
- **Lightweight and data-centric:**
  - Close to in-memory data structures (reduced interaction with external repositories)
  - Clone, reshape

DataStates Architecture

Applications (direct access)

Workflow Engines (e.g. Decaf)

External repositories (e.g. DLHub)

DataStates

Lineage

- Metadata management
- Search and discovery
- Garbage collection / pruning

Performance/scalability

- Multi-objective optimization
- Composable asynchronous actions
- Space efficiency and reduction

Management of heterogeneous storage

PMDK
(Persistent Memory Developer Toolkit)

Mochi

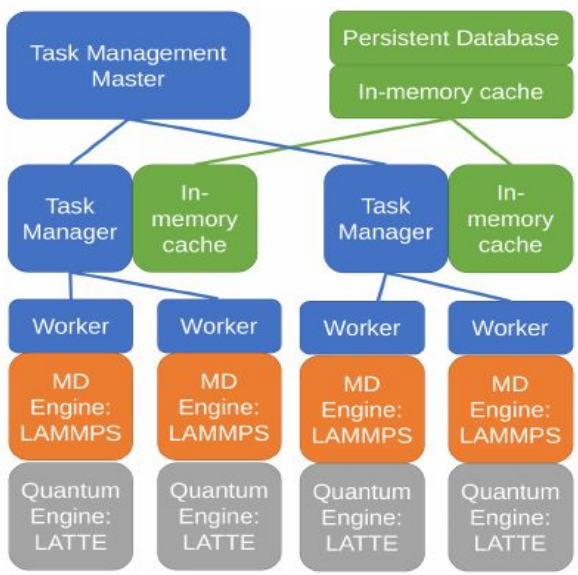
VELOC

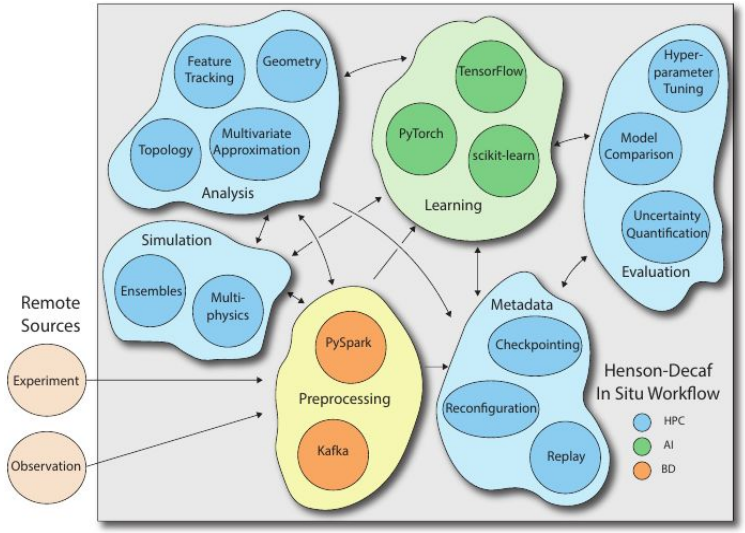
ECP
EXASCALE COMPUTING PROJECT

DataStates Ecosystem

- Beyond checkpointing for “traditional” ECP applications:
  - HACC: coupled simulations + analytics
  - EXAALT: caching and reusing partial results
- Braid: Data Flow Automation for Scalable and FAIR Science (Co-PI)
  - Distributed caching
  - Locality-aware sharing
  - Replay-aware streaming
- Advanced in-situ workflow tools (Co-PI)
  - RRR (Robustness, Reconfigurability, Reproducibility) for triple convergence: AI+DL+HPC
- Many “productive” use cases in deep learning

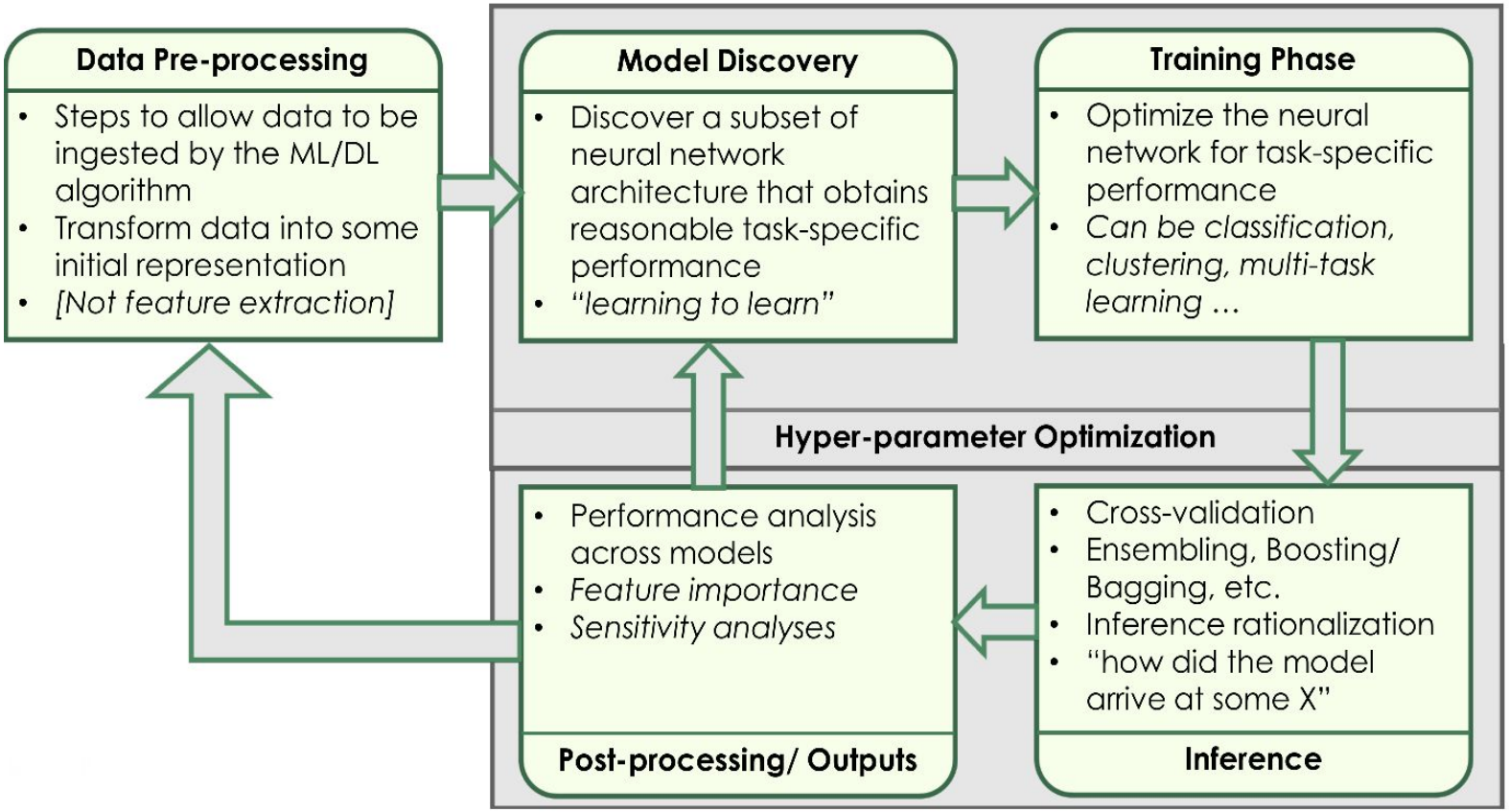
EXAALT: reuse partial results





Argonne NATIONAL LABORATORY

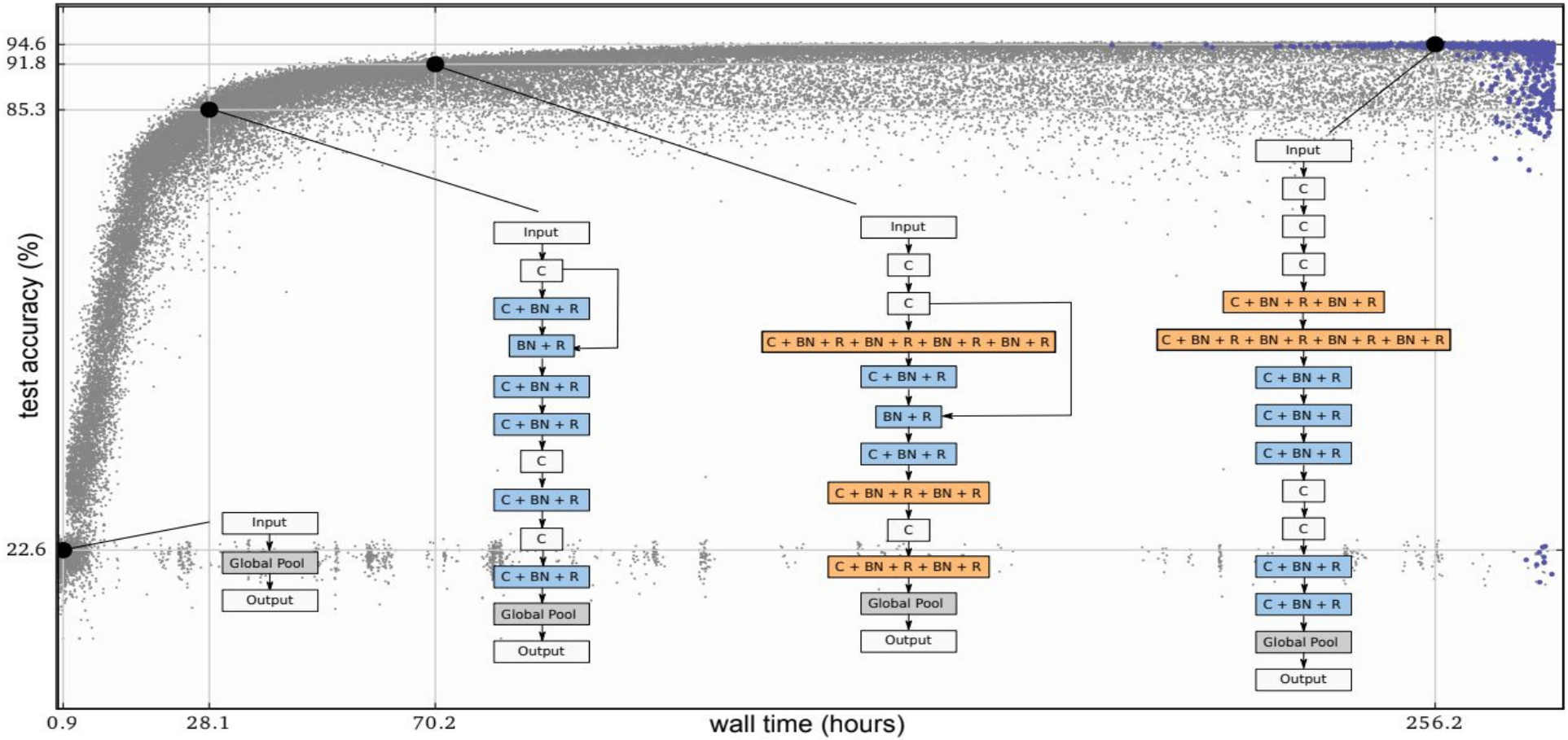
State Preservation for AI and Deep Learning



- Deep learning applications have complex workflows: more than just training and inference
- Model discovery: explore a large number of alternative DNN architectures / hyper-params
- Sensitivity analysis + explainability: explore a large number of alternative training paths
- The notion of "alternative" introduces the need to capture and replicate/revert to previous data states that represent training data and/or models

Argonne NATIONAL LABORATORY

Example 1: Evolutionary Model Discovery

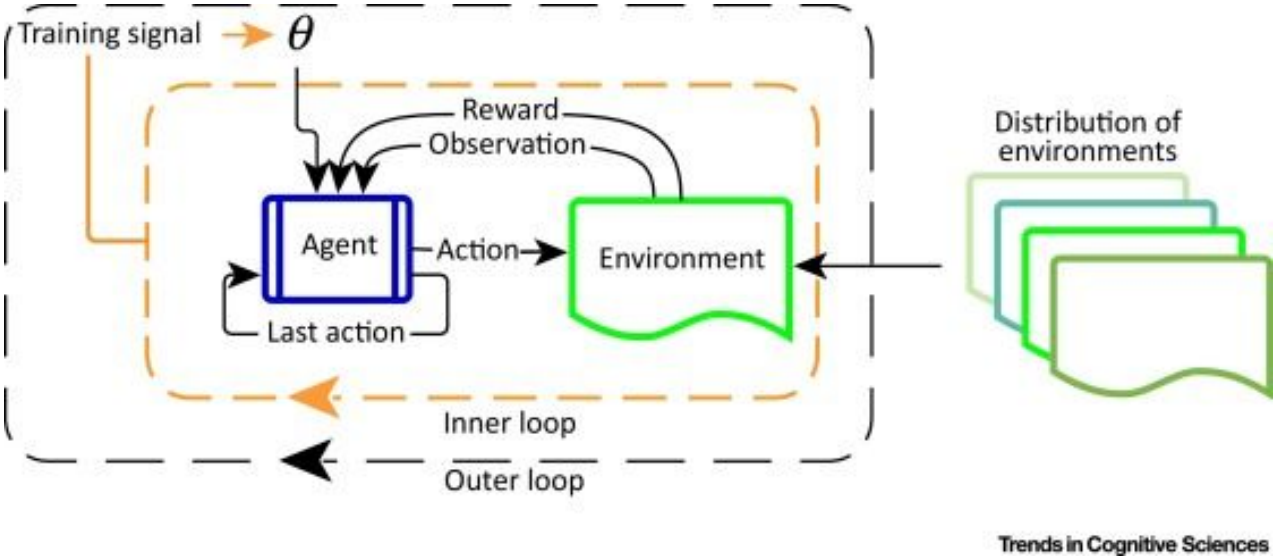


(Esteban Real et al. Large Scale Evolution of Image Classifiers. In ICML'17)

- Constructing DNN architectures manually is tedious and their quality depends on expertise
- Automated approaches include evolutionary algorithms (e.g. GA)
- To avoid a local optimum, a large population of promising individuals (model states) is mutated and/or paired with the aim of obtaining better individuals (blue: active, grey: ancestors)
- Ancestors are needed too in order to study the evolution

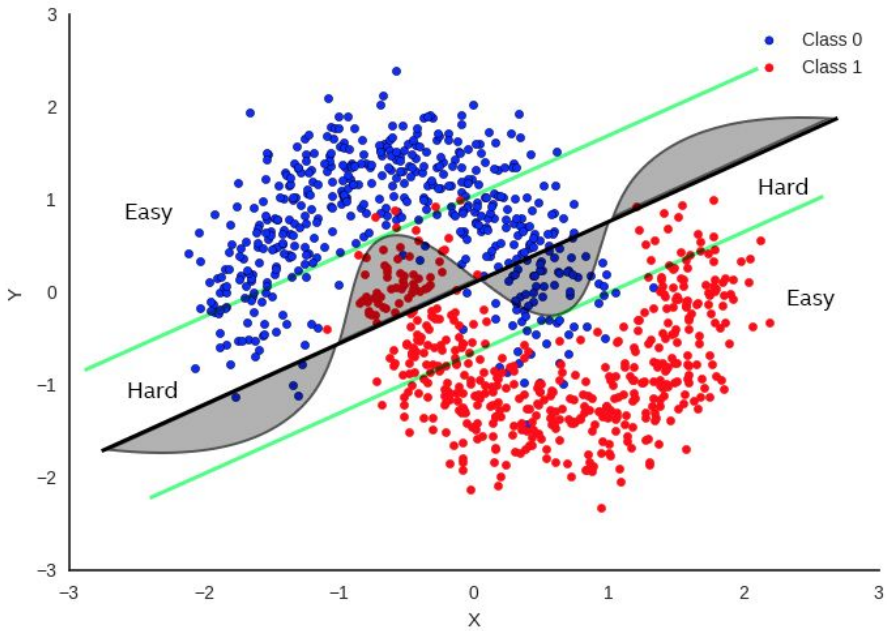
Argonne NATIONAL LABORATORY

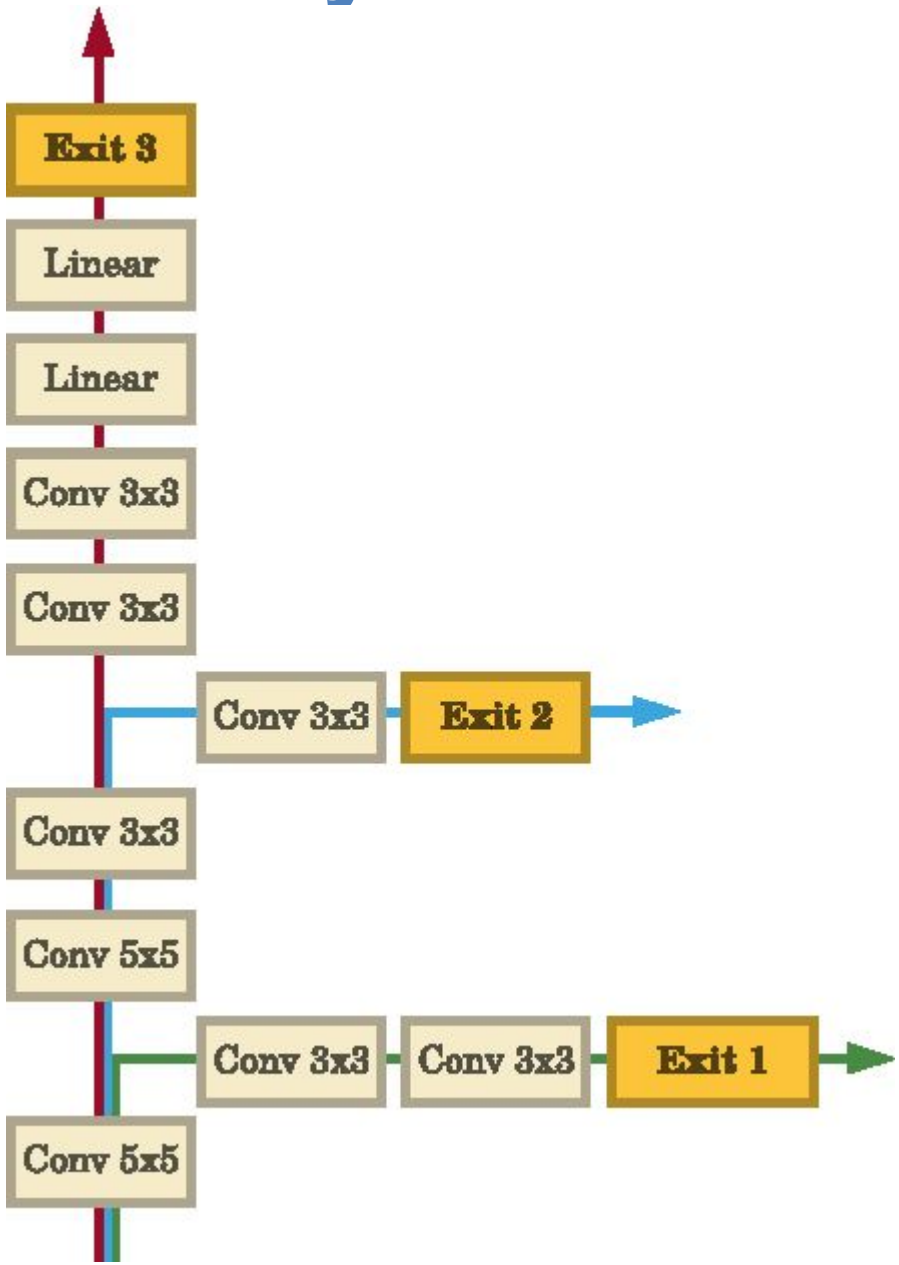
Example 2: Reinforcement Learning



- Used to solve problems for which the ground truth is not known but the quality of the solution can be evaluated
- Modeled as an agent (often a DNN) that receives rewards from its environment for performing actions
- Needs to find and revisit previous states:
  - Sample episodes from a pool of representative environments
  - Go back and try a different action at the same moment in time
  - Reinforce past experience to avoid critical forgetting (overfitting the present)

Example 3: Inference with Early Exit

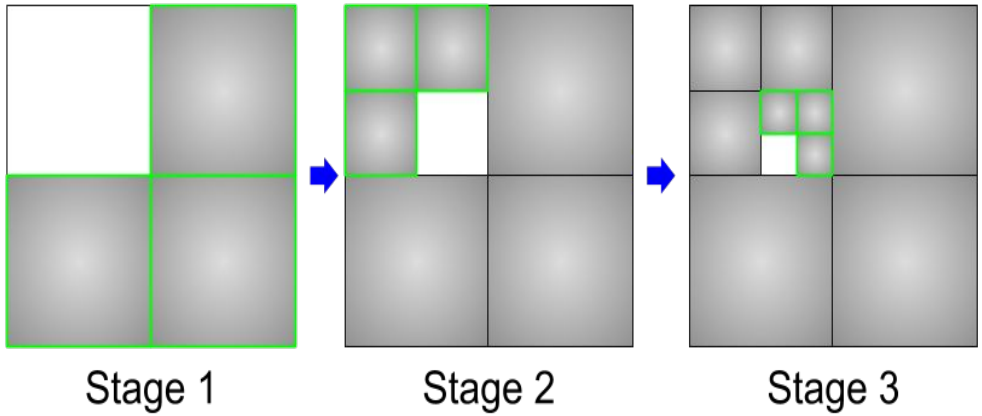


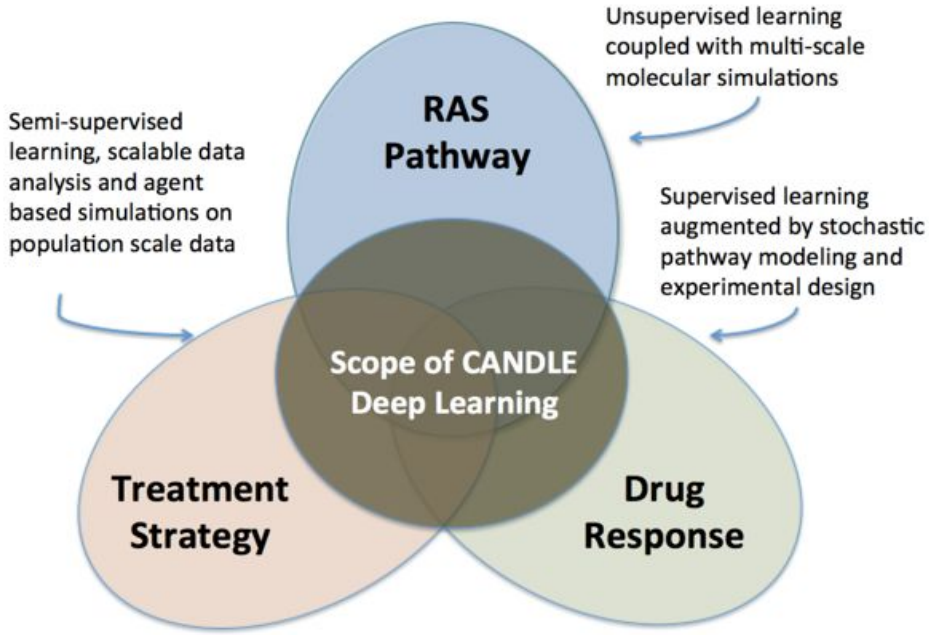


- Applies for non-linear models with branches that trade-off complexity/depth for accuracy
- Ideal for problems with non-uniform difficulty
- Try easy path first
- If result is not good enough, go back and try the harder path

Example 4: Sensitivity Analysis

- Case study: CANDLE (Cancer Deep Learning Environment)
- Sensitivity analysis: important step towards explainability
- Allows for investigations into data quality and learning patterns:
  - Training sample outlier detection
  - DNN model tolerance to outliers
  - Potential for generalization
  - Correlations between sample outliers and learning outliers
- One possible solution: fork models recursively at each step to exclude a different training subset
- Could also boost performance by partial training
- Runs at large scale on ORNL Summit

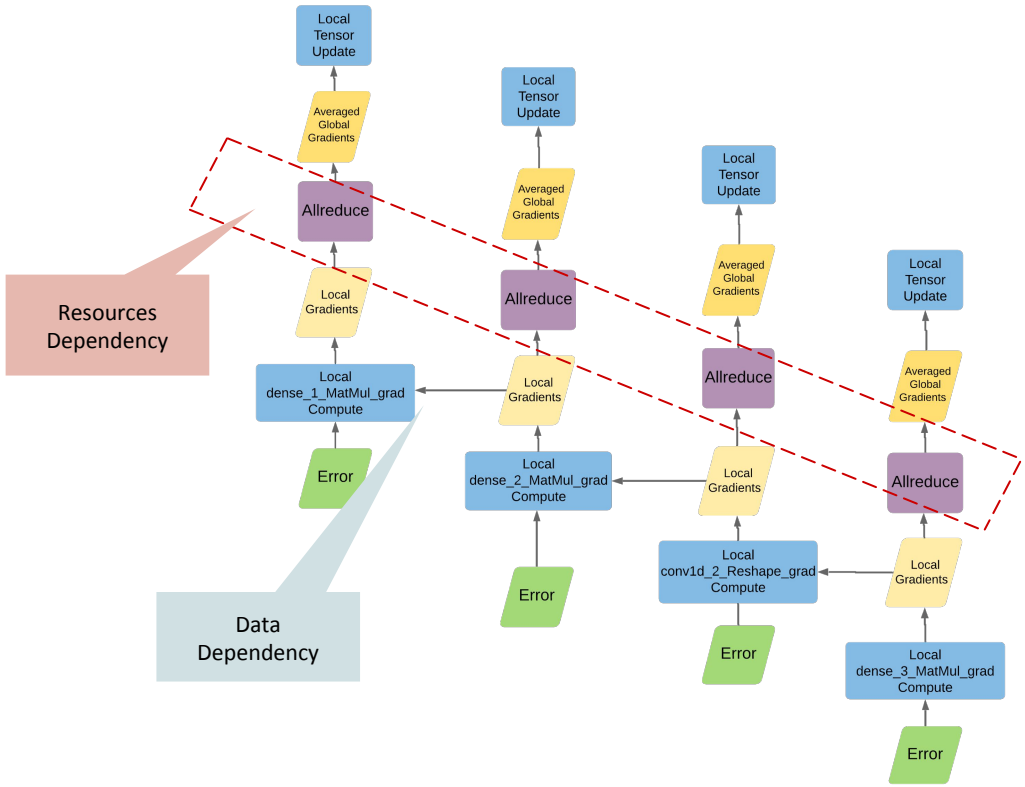




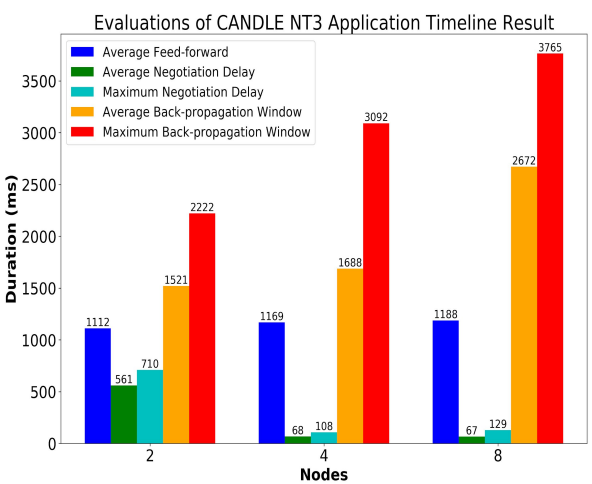
Check out ECP CANDLE slides for more details: http://tiny.cc/zwpojz

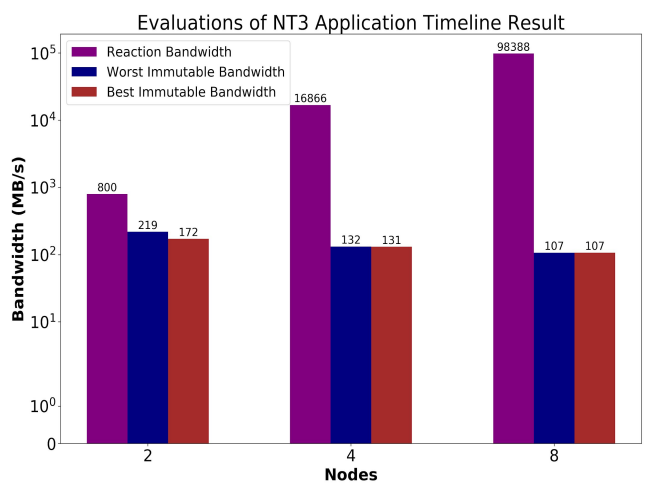
Argonne NATIONAL LABORATORY

Case Study: ECP CANDLE - NT3 Benchmark



- Data-parallel training:
  - Multiple model replicas
  - Each looks at different training samples
  - Gradients are averaged (all-reduce)
- Fine-grain parallelism of back-propagation for data-parallel training
  - Reference implementation: Horovod
  - Resource dependency: all-reduce serialized
  - Data dependency: Wait for lower layers
- Study of fine-grain parallelism
  - Introduced metrics and scripts to process Horovod timeline
  - Results show significant opportunity to embed asynchronous operations into the pipeline (e.g., after local tensor update)
  - Helps understand bottlenecks and system-level aspects related to data-parallel training





Jiali Li, Bogdan Nicolae, Justin Wozniak, and George Bosilca. Understanding scalability and fine-grain parallelism of synchronous data parallel training. **Presented at MLHPC@SC'19**

Argonne NATIONAL LABORATORY

DeepFreeze: Checkpointing for Deep Learning

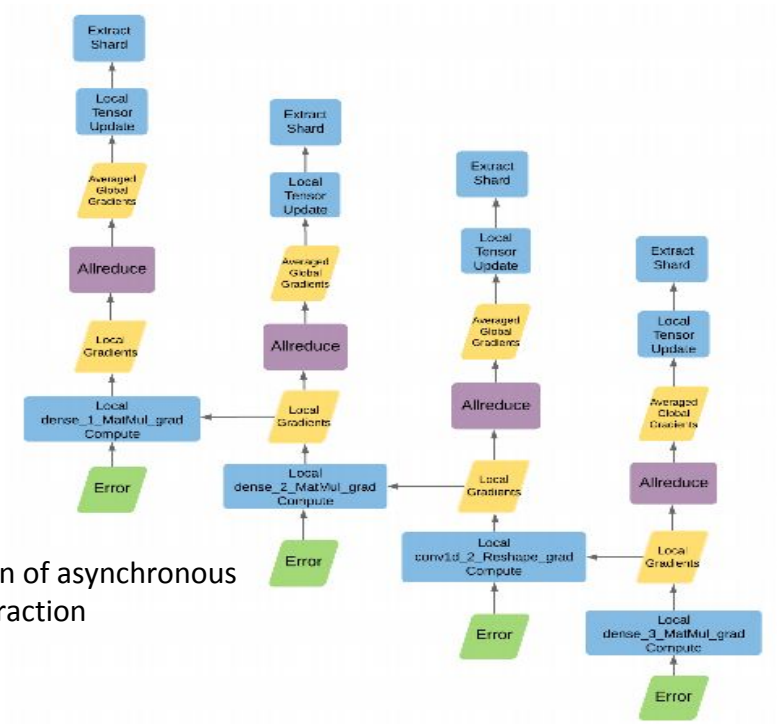
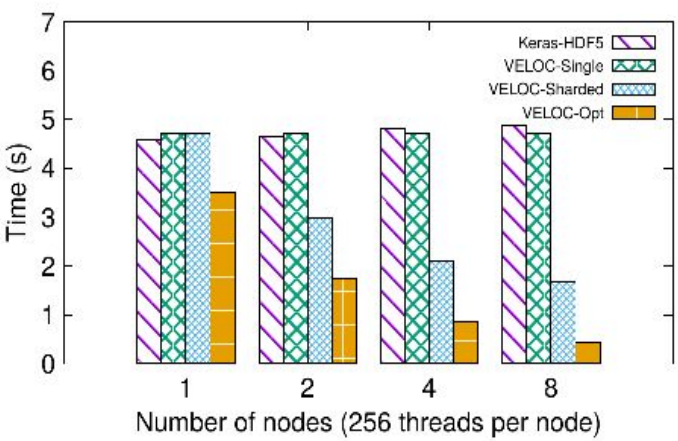


Illustration of asynchronous shard extraction

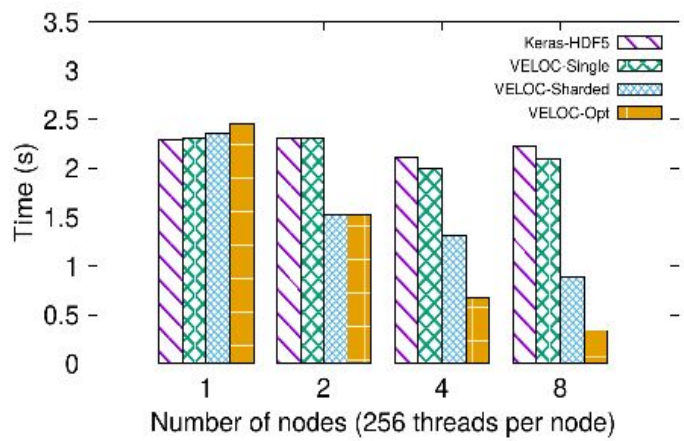
Efficient checkpointing techniques that exploit fine-grain parallelism:

- Efficient serialization to local storage
- Sharding: each model replica flushes a different piece of the layer
- Asynchronous shard extraction: serialization of the pieces is embedded in the Tensorflow graph for maximum parallelism

Results: 10x less checkpointing overhead and 5x less runtime overhead over standard Keras+HDF5 checkpoints (blocking)



(b) Blocking phase (lower is better).



(c) Runtime overhead (lower is better).

Bogdan Nicolae, Jiali Li, Justin Wozniak, George Bosilca, Franck Cappello. DeepFreeze: Towards scalable checkpoinitng of deep learning models.

Accepted at CCGrid'20
To be presented at CCGrid'21

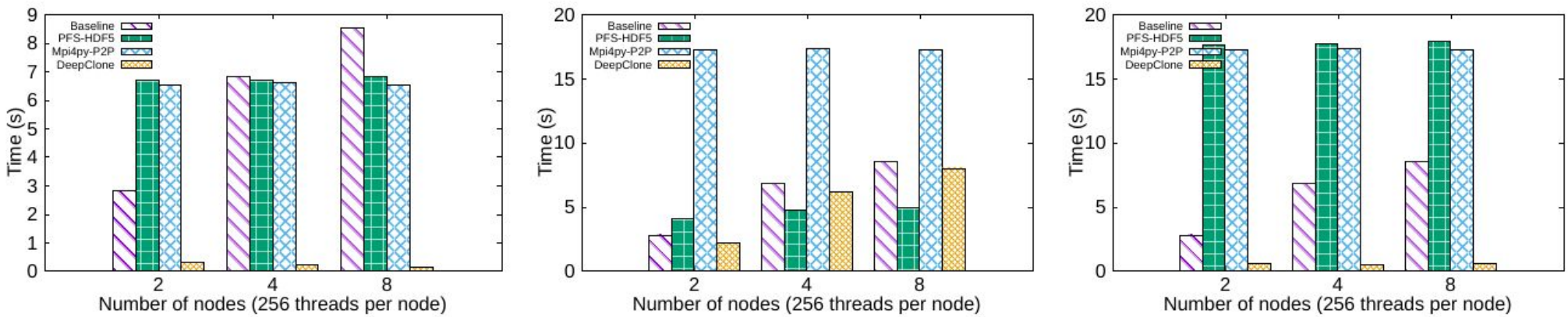
Argonne NATIONAL LABORATORY

DeepClone: Efficient Fork of DNN Models

Efficient forking of data-parallel training of deep learning models in alternative directions:

- Goal: create a clone of the model such that: (1) overhead on source is minimized; (2) stand-by time on destination is minimized; (3) destination continues training as soon as possible
- The clone may be itself a data-parallel training!
- Different from checkpointing: state must not only be captured but also simultaneously duplicated
- Related to live migration
- Principles: (1) zero-copy transfer of the model state; (2) large tensor sharding and reconstruction; (3) augmentation of execution graph for async transfers

Results: up to 25x less runtime overhead (on source) and 35x less readiness duration (time until destination ready to continue)



(a) Runtime overhead (lower is better). (b) Standby duration(lower is better). (c) Readiness duration (lower is better).

Fig. 4: CANDLE-NT3: Scalability of cloning for NT3.C (1.7 GiB model size) for increasing number of nodes.

To be presented at CLUSTER'20 (Sept 17th): https://clustercomp.org/2020/timetable/

Argonne NATIONAL LABORATORY

Conclusions

- State preservation is an important evolution of checkpointing
- Deep learning makes extensive use of state preservation
- DataStates is a new data model specifically designed to address state preservation
  - Automated FAIR-ness
  - I/O performance portability for HPC heterogeneous storage
  - Lightweight and data-centric

- Positions available
  - PhD internships, Postdoc
  - Immediate opening: http://bit.ly/3lcKCPM
- Promising initial results
  - Upcoming talk: **DeepClone - CLUSTER'20 (Sept 17th):** https://clustercomp.org/2020/timetable/

Acknowledgements: Justin Wozniak, Matthieu Dorier, Franck Cappello (ANL); Adam Moody, Kathryn Mohror (LLNL), JiaLi Li, George Bosilca (UTK), Shu-Mei Tseng, Aparna Chandramowlishwaran (UCI)

Funding: DataStates ECRP, ECP VELOC, ANL LDRD, ASCR Braid, ASCR Triple Convergence

Contact: bogdan.nicolae@acm.org
Web: www.bnicolae.net