Using Machine Learning to Predict Effective Compression Algorithms for Heterogeneous Datasets

Brandon Alexander Burtchell* Martin Burtscher

Department of Computer Science

Texas State University

Motivation

- Heterogeneous datasets are prevalent in big-data (e.g., IoT[1], medicine[2])
- Data compression is necessary on large datasets
- Using a single compression algorithm per file is suboptimal
  - Compression algorithms tend to exploit patterns that are unique to a data
- But exhaustively considering many algorithms per file is infeasible

[1] Cios and Moore, "Uniqueness of medical data mining".
[2] Wang, "Heterogeneous Data and Big Data Analytics".

Motivation

- Heterogeneous datasets are prevalent in big-data (e.g., IoT[1], medicine[2])
- Data compression is necessary on large datasets
- Using a single compression algorithm per file is suboptimal
  - Compression algorithms tend to exploit patterns that are unique to a data
- But exhaustively considering many algorithms per file is infeasible

Problem Statement

How can we predict an effective lossless compression algorithm for each file in a heterogeneous dataset?

[1] Cios and Moore, "Uniqueness of medical data mining".

[2] Wang, "Heterogeneous Data and Big Data Analytics".

Introduction

- We call our approach "MLcomp"
- Offloads computation by **training** a nearest-neighbor (1NN) model off-line
- The compression ratios (CRs) of simple compression algos make effective **features**
- A few features (4) **sufficiently distinguish** files in a heterogeneous dataset
- We **reduce** a search space of over 100,000 algos to one well-performing algo for any input
- On our evaluation dataset, MLcomp reaches **97.8%** of the CR achieved when exhaustively searching our library

Background: CRUSHER[4]

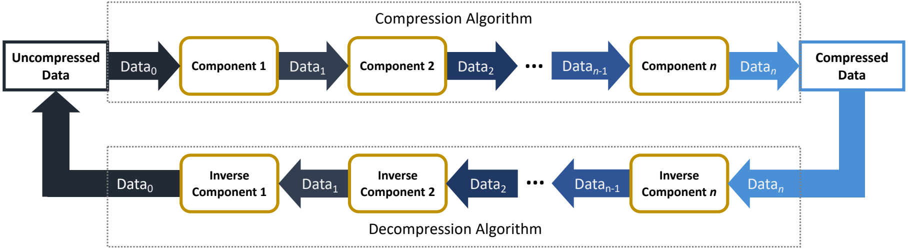

Figure 1: CRUSHER Compression and Decompression Pipeline Flow

- CRUSHER generates $56 \times 56 \times 33 = 103{,}488$ target pipelines
- CRUSHER generates $57 \times 33 = 1{,}881$ feature pipelines
  - We use sequential feature selection (SFS)[3] to greedily choose the 4 best features

[3] Ferri et al., "Comparative study of techniques for large-scale feature selection".

[4] Burtscher et al., "Real-Time Synthesis of Compression Algorithms for Scientific Data".

MLcomp Walkthrough: Setup

- Suppose:
  - 12 heterogeneous files to learn to compress: $\{f_0, f_1, f_2, ..., f_{11}\}$
  - 10 CRUSHER components: $\{c_0, c_1, c_2, ..., c_9\}$

1. Split dataset
   - Training: $\{f_0, f_1, f_2, f_3\}$
   - Validation: $\{f_4, f_5, f_6, f_7\}$
   - Testing: $\{f_8, f_9, f_{10}, f_{11}\}$
2. Generate CRUSHER pipelines:
   - Features (length 1): $\{c_0, c_1, c_2, ..., c_9\}$
   - Targets (length 2): $\{c_0 c_0, c_0 c_1, c_0 c_2, ..., c_9 c_9\}$

MLcomp Walkthrough: Setup

- Suppose:
  - 12 heterogeneous files to learn to compress: $\{f_0, f_1, f_2, ..., f_{11}\}$
  - 10 CRUSHER components: $\{c_0, c_1, c_2, ..., c_9\}$

1. Split dataset
   - Training: $\{f_0, f_1, f_2, f_3\}$
   - Validation: $\{f_4, f_5, f_6, f_7\}$
   - Testing: $\{f_8, f_9, f_{10}, f_{11}\}$
2. Generate CRUSHER pipelines:
   - Features (length 1): $\{c_0, c_1, c_2, ..., c_9\}$
   - Targets (length 2): $\{c_0c_0, c_0c_1, c_0c_2, ..., c_9c_9\}$

MLcomp Walkthrough: Setup

- Suppose:
  - 12 heterogeneous files to learn to compress: $\{f_0, f_1, f_2, ..., f_{11}\}$
  - 10 CRUSHER components: $\{c_0, c_1, c_2, ..., c_9\}$

1. Split dataset
   - Training: $\{f_0, f_1, f_2, f_3\}$
   - Validation: $\{f_4, f_5, f_6, f_7\}$
   - Testing: $\{f_8, f_9, f_{10}, f_{11}\}$
2. Generate CRUSHER pipelines:
   - Features (length 1): $\{c_0, c_1, c_2, ..., c_9\}$
   - Targets (length 2): $\{c_0 c_0, c_0 c_1, c_0 c_2, ..., c_9 c_9\}$

MLcomp Walkthrough: Training

1. Compute features and identify target pipelines
   - i.e., for each training file, run each feature and target pipeline
2. Perform SFS to reduce features to size n
3. Train 1NN model with reduced feature vector

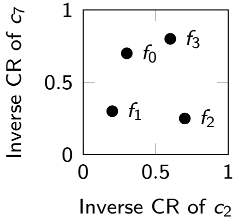

Figure 2: 1NN Feature Space

Training File	Target Pipeline
f_0	$c_2\,c_3$
f_1	$c_4\,c_7$
f_2	$c_6\,c_1$
f_3	$c_2\,c_3$

Table 1: Target Pipeline Lookup

MLcomp Walkthrough: Prediction

1. Compute feature vector of input file
   - i.e., run each feature pipeline on f_8
2. Find nearest neighbor (f_2)
3. Compress with neighbor's target pipeline ($c_6 c_1$)

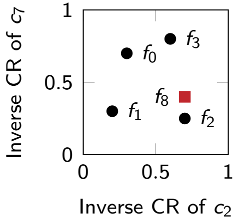

Figure 2: 1NN Feature Space

Training File	Target Pipeline
f_0	$c_2 c_3$
f_1	$c_4 c_7$
f_2	$c_6 c_1$
f_3	$c_2 c_3$

Table 1: Target Pipeline Lookup

Evaluation Methodology

- Data is from the THEMIS-B spacecraft[5]
  - 27 distinct data packet types sent to Earth daily
  - THEMIS-B assigns compressors according to packet type
- Dataset splits:
  - Training: January and February 2013 (1,406 files)
  - Validation: March 2013 (775 files)
  - Testing: All data packets from 2014 (8,916 files)
- Final MLcomp model stats:
  - 4 feature pipelines selected from 1,881 (length 2)
  - 90 target pipelines identified from 103,488 (length 3)

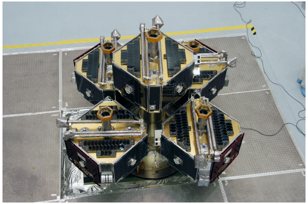

Figure 3: THEMIS Satellites

[5] Angelopoulos, "The THEMIS Mission".

Evaluation Methodology

- Data is from the THEMIS-B spacecraft[5]
  - 27 distinct data packet types sent to Earth daily
  - THEMIS-B assigns compressors according to packet type
- Dataset splits:
  - Training: January and February 2013 (1,406 files)
  - Validation: March 2013 (775 files)
  - Testing: All data packets from 2014 (8,916 files)
- Final MLcomp model stats:
  - 4 feature pipelines selected from 1,881 (length 2)
  - 90 target pipelines identified from 103,488 (length 3)

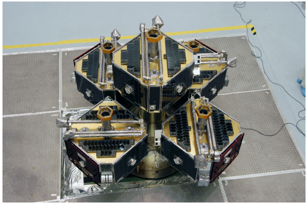

Figure 3: THEMIS Satellites

[5] Angelopoulos, "The THEMIS Mission".

Evaluation Methodology

- Data is from the THEMIS-B spacecraft[5]
  - 27 distinct data packet types sent to Earth daily
  - THEMIS-B assigns compressors according to packet type
- Dataset splits:
  - Training: January and February 2013 (1,406 files)
  - Validation: March 2013 (775 files)
  - Testing: All data packets from 2014 (8,916 files)
- Final MLcomp model stats:
  - 4 feature pipelines selected from 1,881 (length 2)
  - 90 target pipelines identified from 103,488 (length 3)

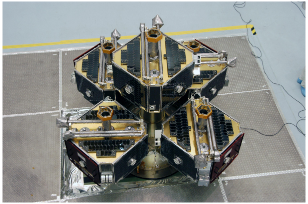

Figure 3: THEMIS Satellites

[5] Angelopoulos, "The THEMIS Mission".

Results: Compression Ratio

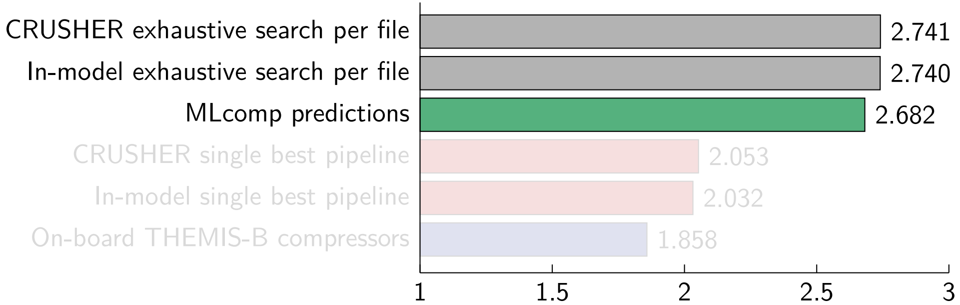

Figure 4: Geometric-mean Compression Ratio of MLcomp and Baselines

- MLcomp nearly achieves our upper bounds
- Compressing with a single pipeline (even the best!) is sub-optimal
- MLcomp surpasses THEMIS-B despite withholding the packet type label

Results: Compression Ratio

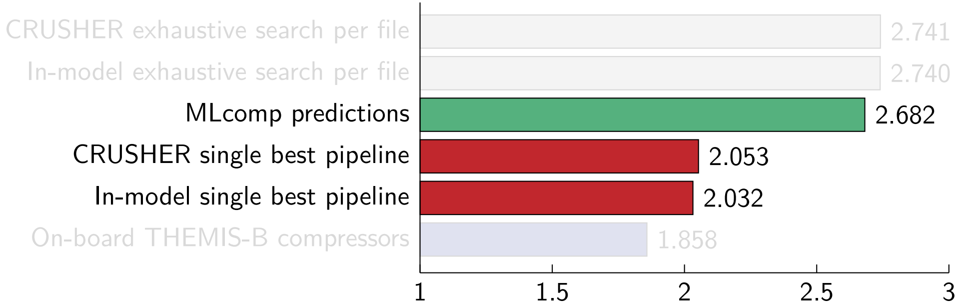

Figure 4: Geometric-mean Compression Ratio of MLcomp and Baselines

- MLcomp nearly achieves our upper bounds
- Compressing with a single pipeline (even the best!) is sub-optimal
- MLcomp surpasses THEMIS-B despite withholding the packet type label

Results: Compression Ratio

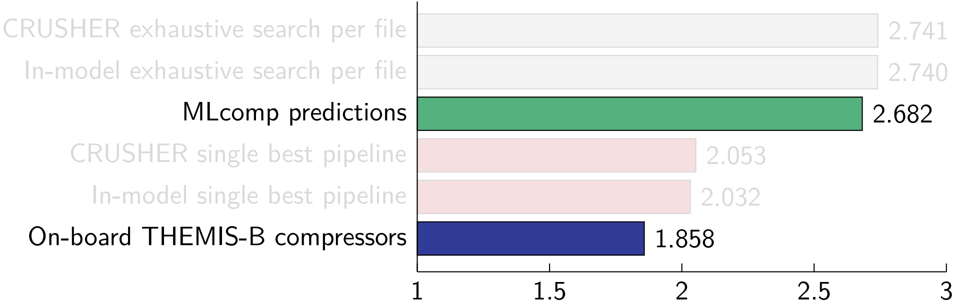

Figure 4: Geometric-mean Compression Ratio of MLcomp and Baselines

- MLcomp nearly achieves our upper bounds
- Compressing with a single pipeline (even the best!) is sub-optimal
- MLcomp surpasses THEMIS-B despite withholding the packet type label

Results: Correlation between Packet Type and Predicted Pipeline

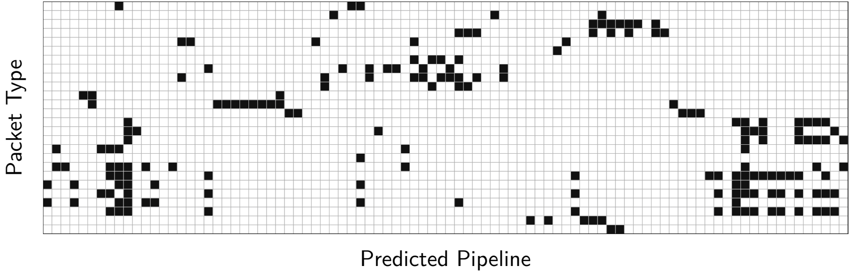

Figure 5: Correlation between Packet Type and Predicted Compression Pipeline

- Discreteness exhibits MLcomp's lack of bias towards a few pipelines
- Some packet types have similar sets of predicted pipelines
  - Likely collected by the same instrument in different modes

Results: Comparison with THEMIS-B Compressors per Packet Type

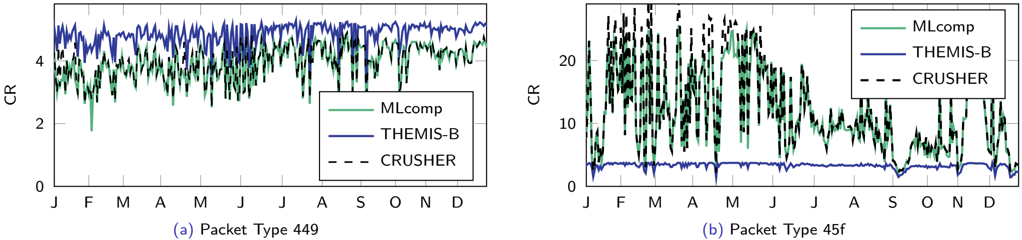

Figure 6: Compression Ratio of Packet Type across Test Set

- 449: THEMIS-B beats MLcomp by highest factor (1.2×)
  - Due to limitations of CRUSHER, not MLcomp
- 45f: MLcomp beats THEMIS-B by highest factor (3.0×)
  - MLcomp predicted 15 distinct pipelines for 45f
  - Adapts to heterogeneity within packet type

Summary & Conclusion

- An ML approach is useful for heterogeneous datasets
  - Using a single algorithm results in poor CRs
  - But exhaustively searching per file is too slow
- Training a model offloads computation, so prediction is relatively quick
- MLcomp yields near-optimal CR on 8,916 unseen heterogeneous packets
- We hope this inspires others to explore ML to improve data compression

Further Questions?

burtchell@txstate.edu

TEXAS STATE UNIVERSITY

ECL Efficient Computing Laboratory