AutoMon: Automatic Distributed Monitoring for Arbitrary Multivariate Functions

HADAR SIVAN
MOSHE GABEL
ASSAF SCHUSTER

SIGMOD 2022
Warn users seconds before earthquake:

- Continuously collect accelerometer statistics.
  - \( x^1, \ldots, x^k \) dynamic statistics vectors of size \( d \)

- Aggregate data
  - \( \bar{x} = \frac{1}{k} \sum_{j=1}^{k} x^j \) global average of local vectors

- Run through a neural network:
  - \( f_{nn}(\bar{x}) = W_3 \cdot \text{tanh} (W_2 \cdot \text{tanh} (W_1 \cdot \bar{x} + b_1) + b_2) + b_3 \)
  - \( W_i, b_i \) network weights
  - tanh activation
Computing $f_{nn}$ in Centralized Settings

- Straightforward in centralized settings:

  ```python
  def f_nn(x, W1, b1, W2, b2, W3, b3):
      return W3 @ tanh(W2 @ tanh(W1 @ x + b1) + b2) + b3
  ```

- Data is not static!
  - What will we do when $x^1 \ldots x^k$ change?
$f_{nn}$ Over Geo-distributed Streams

- Can’t centralize all updates
  - Limited battery, bandwidth
  - Communication costs x1000s more energy than computation!
    [Anastasi et al., Ad hoc networks, 2009; Pottie et al., CACM, 2000]
  - Could overwhelm local datacenter

- $f_{nn}(x^j)$ does not reflect $f_{nn}(\bar{x})$

- Need a communication-efficient algorithm for $f_{nn}$
AutoMon

The first approach for monitoring that is automatic and general:

- Given *source code* for computing $f$ from data...
- ...*automatically* implements a communication-efficient distributed approximation protocol for $f(\bar{x})$
- Reduces communication by up to $\times 50$
- Works on complicated, non-convex $f$
- No need for math
How AutoMon Works?

➢ Setting
  ▶ $n$ nodes with data streams
  ▶ Nodes communicate with coordinator

➢ AutoMon’s Input:
  ▶ Source code for computing $f$ from $\bar{x}$
  ▶ Desired approximation error $\varepsilon$

```python
def f_inner_product(x):
    n = x.shape[0] // 2
    u = x[:n]
    v = x[n:]
    return u @ v
```
AutoMon Protocol Overview

The geometric monitoring protocol
[Alfassi et al., ICDE, 2021; Gabel et al., SIGKDD, 2017; Keren et al., TKDE, 2012]

1. Sync
2. Monitor
3. Report

➢ Collect data, $x_0 = \frac{1}{n} \sum x^j$, and compute current approximation $f_0 = f(x_0)$

➢ **Intuition:** send data and update $f_0$ only if $x^j$ changed “enough”
AutoMon Protocol Overview

- Need constraint that guarantees $|f(\bar{x}) - f_0| \leq \epsilon$.
- Find a **safe zone**: convex set $C$ such that:

$$\bar{x} \in C \implies L \leq f(\bar{x}) \leq U$$

1. Sync
2. Monitor
3. Report
Protocol in Action

From convexity, the average $\bar{x}$ is in $C$.

Coordinator

1. Sync
2. Monitor
3. Report
Protocol in Action

1. Sync
2. Monitor
3. Report

- node 1
- node 2
- node 3
- average

- $x^1$
- poll local data
- new $f_0$, $\mathcal{C}$
- violation! (plus $x^3$)
- $x^3$ is outside $\mathcal{C}$
- average $\bar{x}$ also no longer in $\mathcal{C}$
The Core of AutoMon

➢ During sync, $C$ must be tailored to $f$ such that
\[ \bar{x} \in C \implies L \leq f(\bar{x}) \leq U \]

➢ The magic of AutoMon:
finding a good $C$ automatically, using only $f$’s source code.

1. Sync
2. Monitor
3. Report
AutoMon’s Safe Zone

1. DC DECOMPOSITION

- Convex difference:
  \[ f(x) = \tilde{g}(x) - \tilde{h}(x) \]
  [Lazerson et al., 2018]

- Concave difference:
  \[ f(x) = \hat{g}(x) - \hat{h}(x) \]

2. SAFE ZONE

Finding DC decomposition automatically!

\[ C = \left\{ x \left| \begin{array}{c}
\tilde{g}(x) \leq U \\
\tilde{h}(x) \leq f(x_0) + \nabla f(x_0)^T (x - x_0) - L
\end{array} \right. \right\} \]

\[ C = \left\{ x \left| \begin{array}{c}
\hat{h}(x) \geq f(x_0) + \nabla f(x_0)^T (x - x_0) - U \\
\hat{g}(x) \geq L
\end{array} \right. \right\} \]
Automatic DC Decomposition (ADCD)

1. Numerical optimization $\rightarrow \lambda_X$ (eXtreme eigenvalue)
2. Quadratic component: $h = -\frac{1}{2} \lambda_X \| x - x_0 \|^2 \rightarrow f + h = g$
3. DC decomposition: $f = g - h$
ADCD-X

\[ f \rightarrow \tilde{g} \]

\( f \) : the function we monitor
ADCD-X

\[
f \rightarrow \tilde{g}
\]

\[
f : \text{the function we monitor}
\]

\[
\tilde{h} : \text{quadratic component, based on } \lambda_X
\]

\[
\tilde{g} = f + \tilde{h}
\]
Example: ADCD to Safe-Zone

\[ f(x) = \tilde{g}(x) - \tilde{h}(x) \quad \rightarrow \quad C = \left\{ x \mid \begin{array}{l} \tilde{g}(x) \leq U \\ \tilde{h}(x) \leq f(x_0) + \nabla f(x_0)^T (x - x_0) - L \end{array} \right\} \]
Example: ADCD to Safe-Zone

- $f = \sin(x)$
- Gray area: $L \leq f(x) \leq U$

\[
f(x) = \tilde{g}(x) - \tilde{h}(x) \quad \mathcal{C} = \left\{ x \left| \begin{array}{l}
\tilde{g}(x) \leq U \\
\tilde{h}(x) \leq f(x_0) + \nabla f(x_0)^T (x - x_0) - L
\end{array} \right. \right\}
\]
Example: ADCD to Safe-Zone

- \( f = \sin(x) \)
- Gray area: \( L \leq f(x) \leq U \)

- Start with \( \tilde{g}(x) \)

\[
f(x) = \tilde{g}(x) - \tilde{h}(x) \quad \mathcal{C} = \left\{ x \mid \begin{array}{l}
\tilde{g}(x) \leq U \\
\tilde{h}(x) \leq f(x_0) + \nabla f(x_0)^T(x - x_0) - L
\end{array} \right\}
\]
Example: ADCD to Safe-Zone

- \( f = \sin(x) \)
  - Gray area: \( L \leq f(x) \leq U \)

- Start with \( \tilde{g}(x) \)
- Show area where \( \tilde{g}(x) \leq U \)
  - Partial safe zone \( \mathcal{C}_U \)

\[
f(x) = \tilde{g}(x) - \tilde{h}(x) \quad \mathcal{C} = \left\{ x \mid \begin{align*} \tilde{g}(x) &\leq U \\ \tilde{h}(x) &\leq f(x_0) + \nabla f(x_0)^T (x - x_0) - L \end{align*} \right\}
\]
Example: ADCD to Safe-Zone

- $f = \sin(x)$
- Gray area: $L \leq f(x) \leq U$

- Now $\tilde{h}(x)$

\[ f(x) = \tilde{g}(x) - \tilde{h}(x) \quad \Rightarrow \quad e = \left\{ x \left| \begin{array}{l} \tilde{g}(x) \leq U \\ \tilde{h}(x) \leq f(x_0) + \nabla f(x_0)^T (x - x_0) - L \end{array} \right. \right\} \]
Example: ADCD to Safe-Zone

1. \( f = \sin(x) \)
   - Gray area: \( L \leq f(x) \leq U \)

2. Now \( \tilde{h}(x) \) and tangent minus \( L \)
   \[
   f(x_0) + \nabla f(x_0)^T (x - x_0) - L
   \]

\[
\begin{align*}
  f(x) &= \tilde{g}(x) - \tilde{h}(x) \\
  c &= \left\{ x \mid \tilde{g}(x) \leq U, \tilde{h}(x) \leq f(x_0) + \nabla f(x_0)^T (x - x_0) - L \right\}
\end{align*}
\]
Example: ADCD to Safe-Zone

- \( f = \sin(x) \)
  - Gray area: \( L \leq f(x) \leq U \)

- Now \( \tilde{h}(x) \) and tangent minus \( L \)
  - \( f(x_0) + \nabla f(x_0)^T (x - x_0) - L \)

- The area where \( \tilde{h}(x) \leq \text{tangent} \)

\[
f(x) = \bar{g}(x) - \tilde{h}(x) \quad \mathcal{C} = \left\{ x \mid \bar{g}(x) \leq U \right. \\
\left. \tilde{h}(x) \leq f(x_0) + \nabla f(x_0)^T (x - x_0) - L \right\}
\]
Example: ADCD to Safe-Zone

- \( f = \sin(x) \)
- Gray area: \( L \leq f(x) \leq U \)

- Both areas

\[
f(x) = \tilde{g}(x) - \tilde{h}(x) \\
\mathcal{C} = \left\{ x \mid \begin{array}{l}
\tilde{g}(x) \leq U \\
\tilde{h}(x) \leq f(x_0) + \nabla f(x_0)^T (x - x_0) - L
\end{array} \right\}
\]
Example: ADCD to Safe-Zone

- $f = \sin(x)$
  - Gray area: $L \leq f(x) \leq U$

- Both areas

- Final $\mathcal{C}$ is intersection $\mathcal{C}_L \cap \mathcal{C}_U$

\[
f(x) = \tilde{g}(x) - \tilde{h}(x) \quad \mathcal{C} = \left\{ x \left| \begin{array}{l}
\tilde{g}(x) \leq U \\
\tilde{h}(x) \leq f(x_0) + \nabla f(x_0)^T (x - x_0) - L
\end{array} \right. \right\}
\]
Convex vs. Concave Difference

\[ L \leq f(x) \leq U \]

Convex difference
Convex vs. Concave Difference

\[ \sin(x) \quad f(x_0) \quad U \quad L \]

Convex difference

Concave difference

\[ C \]
Convex vs. Concave Difference

The DC Heuristic

Convex difference

Concave difference
The DC Heuristic

Convex difference

Concave difference

✓ Reduces safe zone violations by up to 30%
Optimizations and Correctness Guarantees

➢ Approximation correctness guarantees:
  ❑ For constant $H$ and convex/concave functions
  ❑ For others, not guaranteed → in practice error is small

➢ “Lazy” violation resolution avoids extra syncing

➢ Optimization on neighborhood of $x_0$
  ❑ Tradeoff parameter that affects safe zone effectiveness
  ❑ Automatic tuning algorithm
Evaluation Setup

- Different applications
- One coordinator
- $n$ nodes (5 to 1000)
- Vector size $d$ (10 to 200)
- 1K to 311K datapoints
Applications (Functions & Datasets)

**INTRUSION DETECTION + DNN**
- KDDCup-99 network data
- 5-layer DNN [512, 64, 32, 16, 8] with ReLU activation

**POLLUTION MONITORING + KLD**
- Beijing air-quality dataset [Zhang et al, ’17]
- KLD, $D_{KL}(P\|Q)$, of PM10 and PM2.5

(we also have inner product, quadratic form, MLP, Rozenbrock, ...)

H. Sivan

AutoMon, SIGMOD 2022
Results: Error-Communication Tradeoff

- One run with specific $\epsilon$
  - X axis – total sent messages
  - Y axis – max error across run

lower-left is better (less error with fewer messages)
Results: Error-Communication Tradeoff

- Test on a range of $\varepsilon$
- **AutoMon**’s trade-off curve of on this data and function

![Graph showing the trade-off curve between max error and number of messages](image-url)
Results:
Error-Communication Tradeoff

➢ Test on a range of $\epsilon$

➢ **AutoMon**

➢ **Centralization:**
  just send all data updates.
  - No error
  - State-of-the-art for sketches (they reduce message size, not number of messages)
Results:
Error-Communication Tradeoff

- Test on a range of $\epsilon$
- **AutoMon**
- **Centralization**
- **Periodic**: send every N updates
  - Non-adaptive
  - The common approach
Results:
Error-Communication Tradeoff

Value changes **slowly:**

- Periodic wastes messages
- AutoMon is adaptive and communication-efficient
- **2% comm** with low error
Results: Error-Communication Tradeoff

DNN (intrusion detection)

Value changes slowly:
- Periodic wastes messages
- AutoMon is adaptive and communication-efficient
- 2% comm with low error

KLD (pollution monitoring)

Value changes gradually:
- AutoMon performance similar to Periodic
- AutoMon guarantees error, and is adaptive
Results: Error-Communication Tradeoff

Value changes slowly:
- Periodic wastes messages
- AutoMon is adaptive and communication-efficient
- 2% comm with low error

Value changes gradually:
- AutoMon performance similar to Periodic
- AutoMon guarantees error, and is adaptive

Value changes quickly:
- Large error in Periodic
- AutoMon is adaptive: smooth, superior tradeoff

DNN (intrusion detection)
- Value changes slowly:
  - AutoMon is communication-efficient
  - 2% comm with low error

KLD (pollution monitoring)
- Value changes gradually:
  - AutoMon guarantees error, and is adaptive
- Value changes quickly:
  - AutoMon is adaptive: smooth, superior tradeoff

H. Sivan
AutoMon, SIGMOD 2022
Results:
Error-Communication Tradeoff

AutoMon:
✓ provides equivalent or superior tradeoff to current approaches ...
✓ automatically from source code.
Error-Bandwidth Tradeoff

AWS experiments:
- Coordinator on us-west-2
- Nodes on us-east-2
- Round trip time = 56ms

Investigate:
1) Validity of simulation
2) Error-bandwidth tradeoff
3) Estimate reduction in traffic
DNN Results: Error-Bandwidth Tradeoff

✓ Similar reduction in messages (0 to 16% error)
DNN Results: Error-Bandwidth Tradeoff

- Same reduction in messages (0 to 16% error)
- Error-BW tradeoff agrees with error-messages tradeoff
DNN Results: Error-Bandwidth Tradeoff

- Same reduction in messages (0 to 16% error)
- Error-BW tradeoff agrees with error-messages tradeoff
- Traffic reduced to 1/3 on average, and up to 98%
DNN Results: Error-Bandwidth Tradeoff

- Same reduction in messages (0 to 16% error)
- Error-BW tradeoff agrees with error-messages tradeoff
- Traffic reduced to 1/3 on average, and up to 98%
Additional Experiments

Scalability

Approximation guarantees and empirical errors

Automatic parameter tuning

Neighborhood size tradeoffs
Related Work

Sampling
- Send only some updates
- Delay warnings
- Miss transients

Distributed Dataflow, Query Planning
- Combine operators
- No built-in operator for $f_{nn}$

Sketching
- Compress updates, approximate $f_{nn}$
- No sketch for $f_{nn}$
- Reduce bandwidth, not number of messages

Geometric Monitoring
- Avoid sending updates
- No bound for $f_{nn}$
- Devs are not PhDs

H. Sivan
AutoMon vs. Universal Sketches

- **Liu et al., One Sketch to Rule Them All: Rethinking Network Flow Monitoring with UnivMon, SIGMOD ‘16**

- Universal sketch: just implement a function \( g(x_i) \)

- Limited to Stream-PolyLog functions in the turnstile model:
  - \( f(x) = \sum g(x_i) \)
  - \( x_i \) are frequencies
  - \( g \) monotonic, bounded by \( O(x_i^2) \)

- AutoMon supports wider variety of \( x \) and \( f \)
Summary

➢ AutoMon: first truly automatic distributed monitor
  ✓ Automatic: Arbitrary functions of $\bar{x}$.
  ✓ Accessible: Works from source code.
  ✓ Efficient: Superior error-communication tradeoff to existing methods.

➢ Allows difficult functions without hand-crafted solution
  ❑ For example, for DNN, reduces communication by up to $\times 50$.

➢ Open source: [https://github.com/hsivan/automon](https://github.com/hsivan/automon)

hadarivan@cs.technion.ac.il  mgabel@cs.toronto.edu  assaf@technion.ac.il