Real-Time Anomaly Detection from Edge to HPC-Cloud

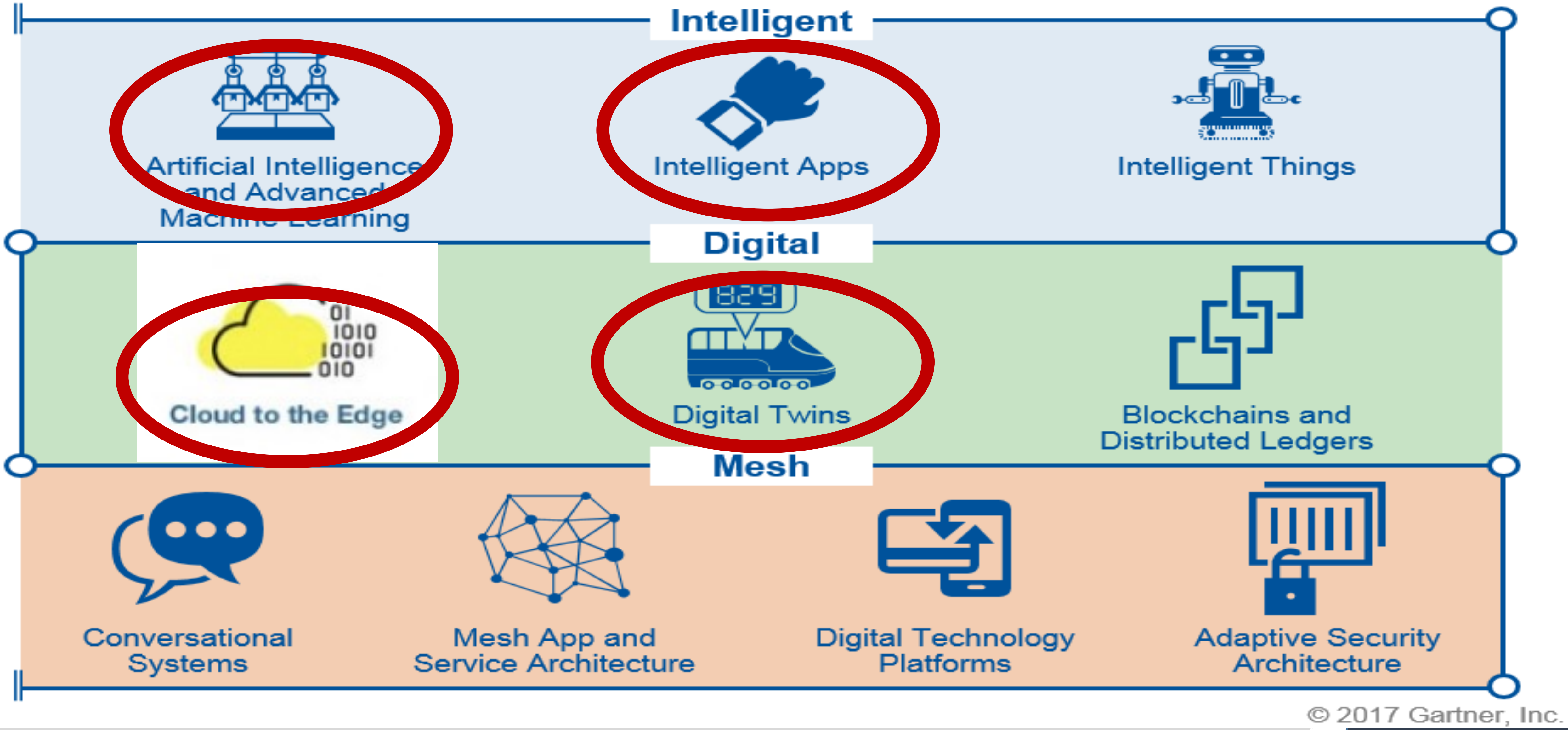
Intro & Preliminary Results

November 14, 2018

Digital Science Center

Judy Qiu, Bo Peng, Supun Kamburu, Sahil Tyagi

Intelligent Systems Engineering Department
Indiana University
Email: xqiu@indiana.edu


Intelligent
Artificial Intelligence and Advanced Machine Learning
Intelligent Apps
Intelligent Things
Digital
Cloud to the Edge
Digital Twins
Blockchains and Distributed Ledgers
Mesh
Conversational Systems
Mesh App and Service Architecture
Digital Technology Platforms
Adaptive Security Architecture
© 2017 Gartner, Inc.

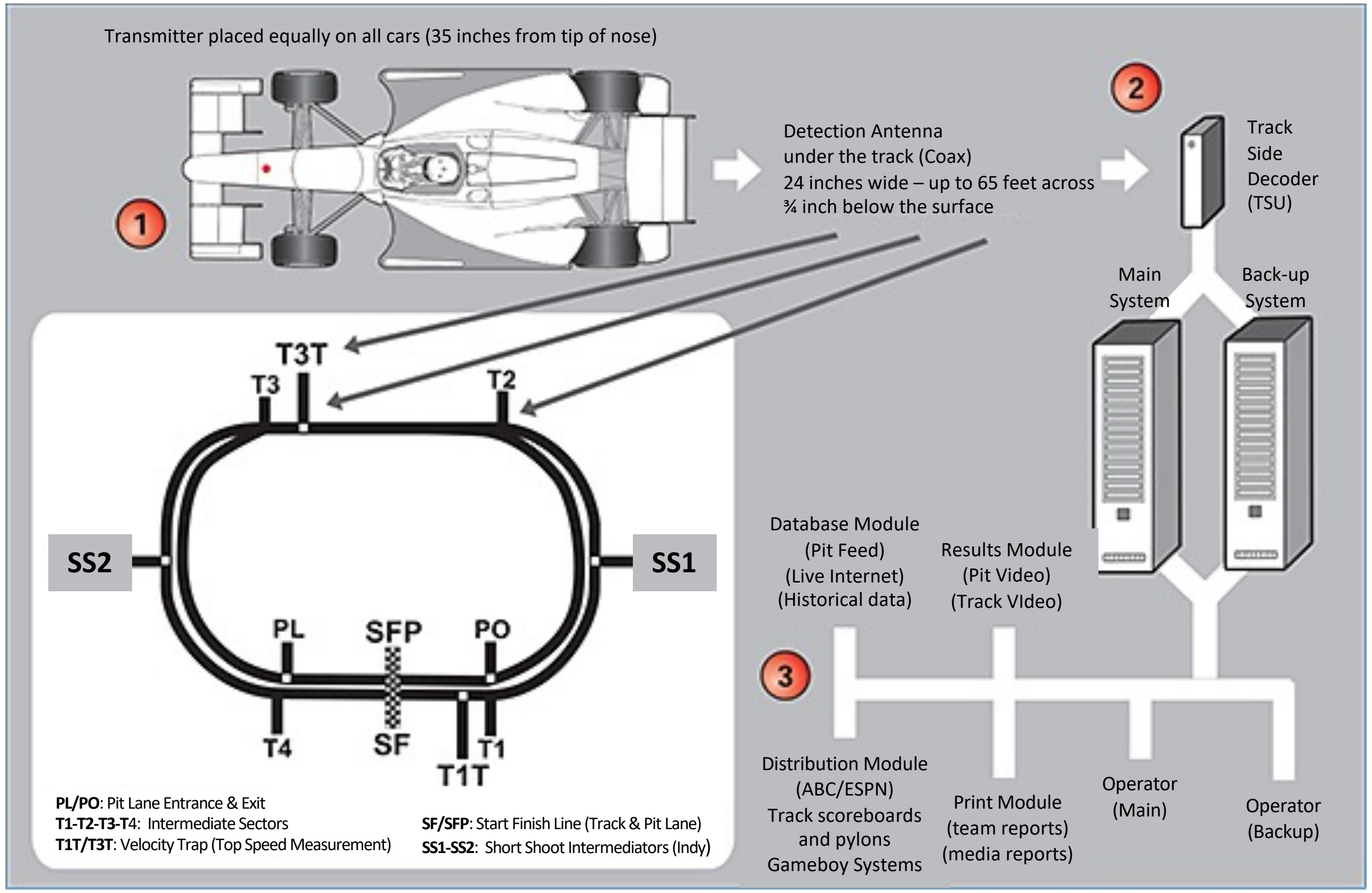
Gartner’s report on Strategic Technology Trend for 2017-2018

- The IndyCar Series is a major open-wheel racing format in North America. The series' premier event is the Indianapolis 500, held each May.
- Computing Systems and Data analytics is critical to the sport, both in improving the performance of the team to make it faster and in helping the race control to make it safer.

IndyCar

https://youtu.be/d8EFRIfls3Q

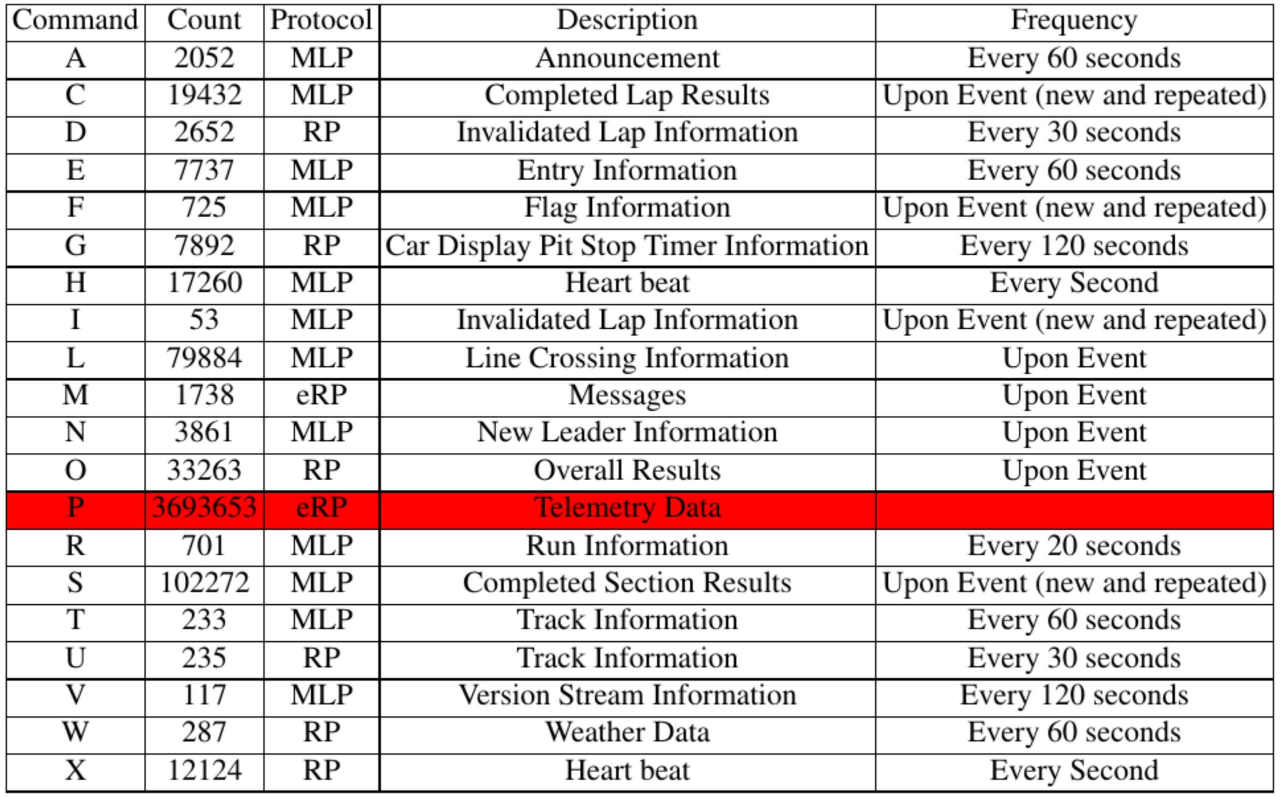
Indianapolis 500 Car Racing, May 2017


Transmitter placed equally on all cars (35 inches from tip of nose)
1
Detection Antenna under the track (Coax) 24 inches wide – up to 65 feet across ¾ inch below the surface
2
Track Side Decoder (TSU)
Main System
Back-up System
T3T
T3
T2
SS2
SS1
PL
SFP
PO
T4
SF
T1
T1T
PL/PO: Pit Lane Entrance & Exit
T1-T2-T3-T4: Intermediate Sectors
T1T/T3T: Velocity Trap (Top Speed Measurement)
SF/SFP: Start Finish Line (Track & Pit Lane)
SS1-SS2: Short Shoot Intermediators (Indy)
3
Database Module (Pit Feed) (Live Internet) (Historical data)
Results Module (Pit Video) (Track VIdeo)
Distribution Module (ABC/ESPN) Track scoreboards and pylons Gameboy Systems
Print Module (team reports) (media reports)
Operator (Main)
Operator (Backup)

- Sensors in the cars and under the track.
- Antenna and communication system.
- Telemetry data (including the many performance information of the cars like speed, gear, brake, throttle,etc) stream into the on-site computer system in a real-time fasion.

Timing and Score Data

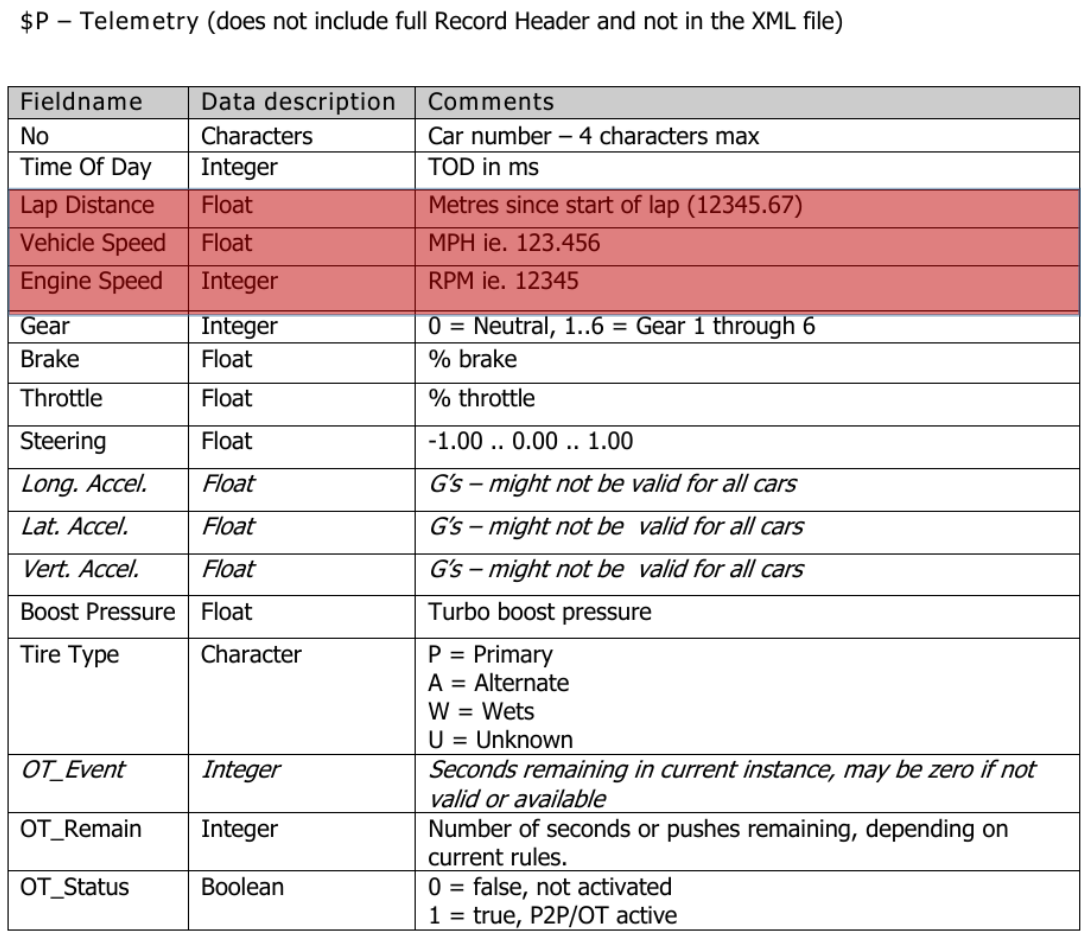
Dataset



Command	Count	Protocol	Description	Frequency
A	2052	MLP	Announcement	Every 60 seconds
C	19432	MLP	Completed Lap Results	Upon Event (new and repeated)
D	2652	RP	Invalidated Lap Information	Every 30 seconds
E	7737	MLP	Entry Information	Every 60 seconds
F	725	MLP	Flag Information	Upon Event (new and repeated)
G	7892	RP	Car Display Pit Stop Timer Information	Every 120 seconds
H	17260	MLP	Heart beat	Every Second
I	53	MLP	Invalidated Lap Information	Upon Event (new and repeated)
L	79884	MLP	Line Crossing Information	Upon Event
M	1738	eRP	Messages	Upon Event
N	3861	MLP	New Leader Information	Upon Event
O	33263	RP	Overall Results	Upon Event
P	3693653	eRP	Telemetry Data	
R	701	MLP	Run Information	Every 20 seconds
S	102272	MLP	Completed Section Results	Upon Event (new and repeated)
T	233	MLP	Track Information	Every 60 seconds
U	235	RP	Track Information	Every 30 seconds
V	117	MLP	Version Stream Information	Every 120 seconds
W	287	RP	Weather Data	Every 60 seconds
X	12124	RP	Heart beat	Every Second

- The INDYCAR Timing system supports retrieving timing data from the primary timing system – serial or sequential data feed for live data and report querying for historical or archived data.
- The Results Protocol is designed to deliver more detailed results information through the use of a single record command.
- Example: one Indianapolis 500 car race on the 28th of May 2017 which contained 750 MB of data and total of 3986170 records.

Telemetry Data



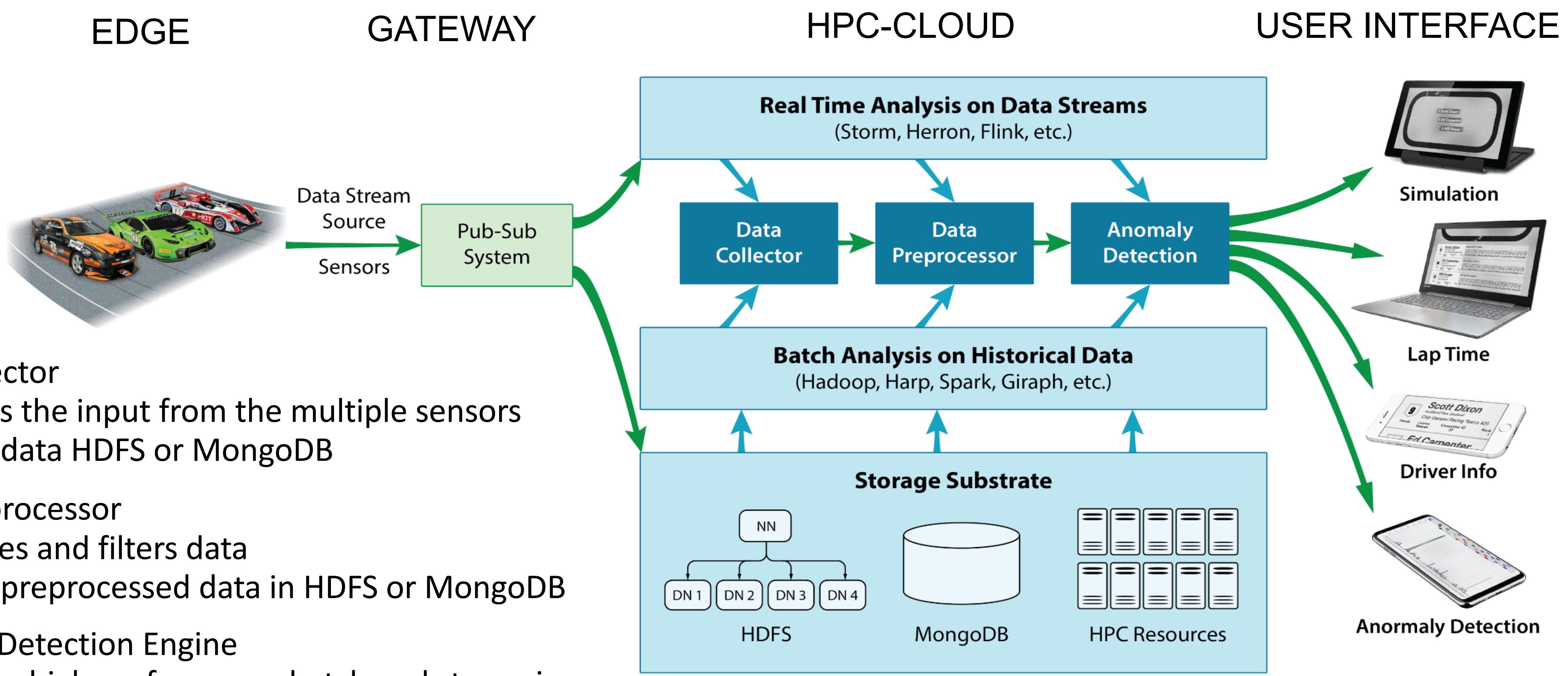
$P – Telemetry (does not include full Record Header and not in the XML file)

Fieldname	Data description	Comments
No	Characters	Car number – 4 characters max
Time Of Day	Integer	TOD in ms
Lap Distance	Float	Metres since start of lap (12345.67)
Vehicle Speed	Float	MPH ie. 123.456
Engine Speed	Integer	RPM ie. 12345
Gear	Integer	0 = Neutral, 1..6 = Gear 1 through 6
Brake	Float	% brake
Throttle	Float	% throttle
Steering	Float	-1.00 .. 0.00 .. 1.00
Long. Accel.	*Float*	*G's – might not be valid for all cars*
Lat. Accel.	*Float*	*G's – might not be valid for all cars*
Vert. Accel.	*Float*	*G's – might not be valid for all cars*
Boost Pressure	Float	Turbo boost pressure

Tire Type	Character	P = Primary
		A = Alternate
		W = Wets
		U = Unknown

OT_Event	*Integer*	*Seconds remaining in current instance, may be zero if not valid or available*
OT_Remain	Integer	Number of seconds or pushes remaining, depending on current rules.
OT_Status	Boolean	0 = false, not activated 1 = true, P2P/OT active

- Telemetry is a radio device that relays information such as engine, tire, steering and throttle performance to team engineers in the pit box. The team can monitor car and driver activity to ensure the car is performing properly.
- During a race, IndyCar Series teams use telemetry to gather data live from their cars as they formulate their race strategy. In the IndyCar Series, teams receive their data from their own on-board systems as well as from the league's Timing & Scoring operation.



Data collector

- collects the input from the multiple sensors
- stores data HDFS or MongoDB

Data Preprocessor

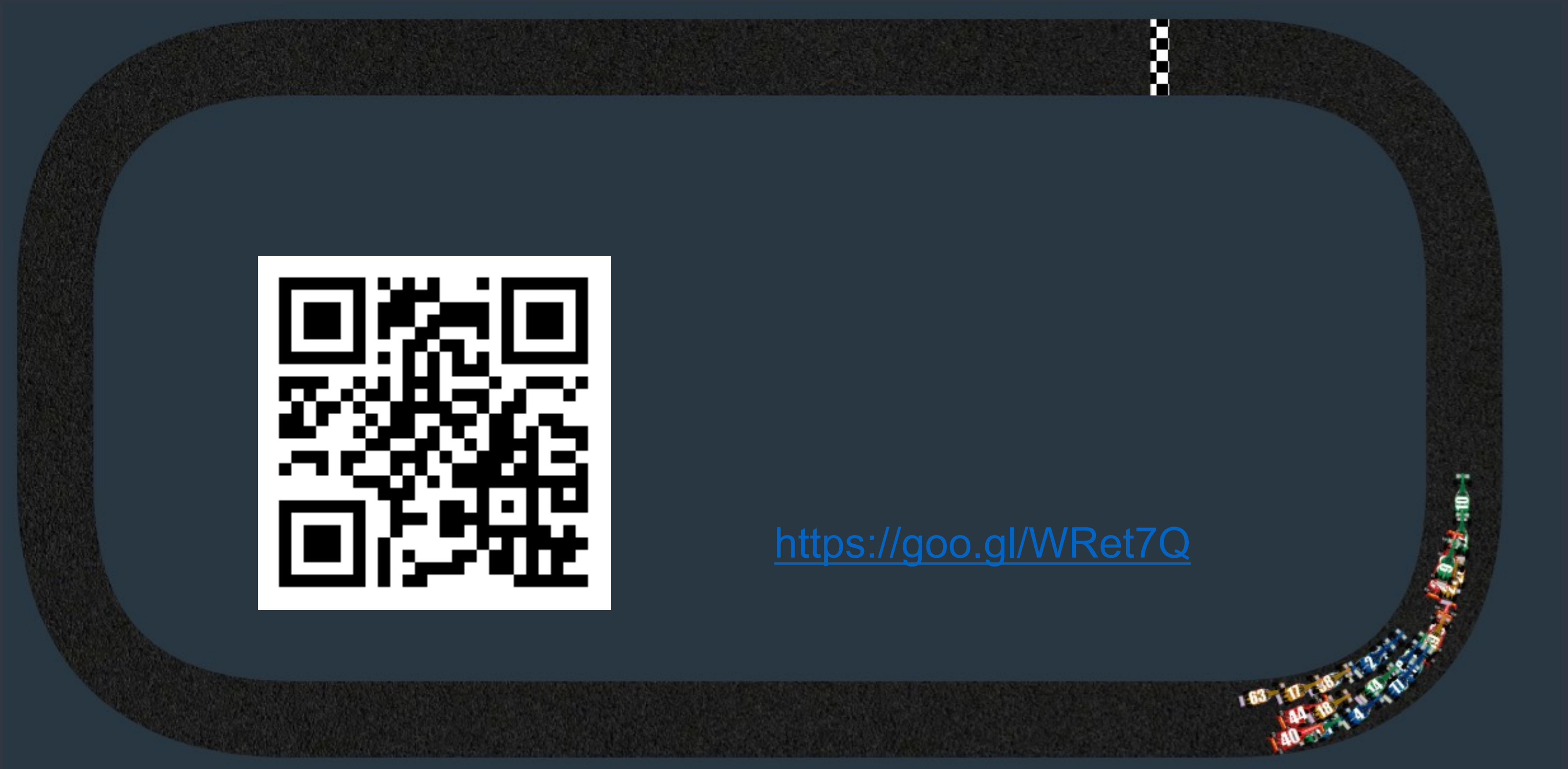
- accesses and filters data
- stores preprocessed data in HDFS or MongoDB

Anomaly Detection Engine

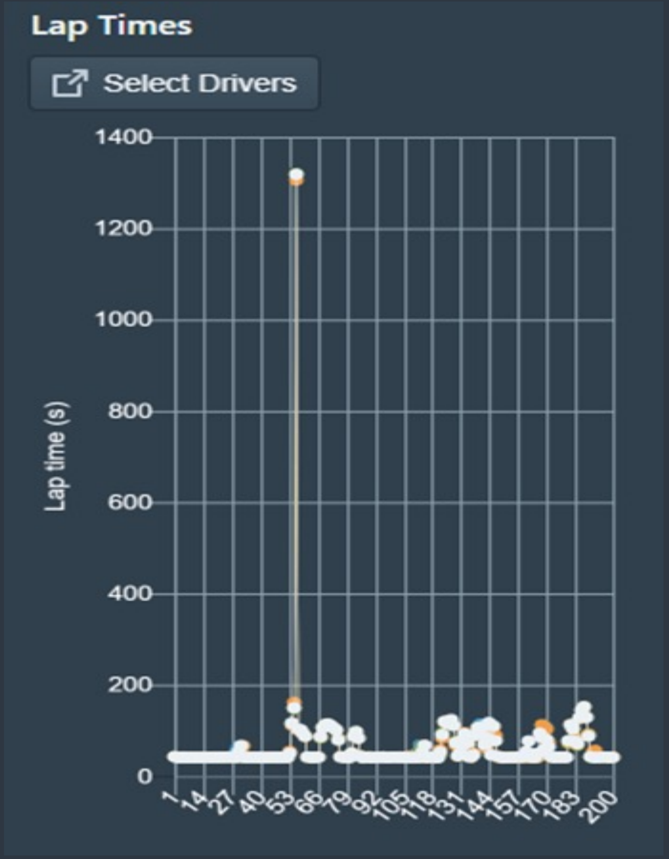
- involves high performance batch and streaming data processing frameworks
- detects anomalies from preprocessed data pipeline

System Architecture

SIMULATION



https://goo.gl/WRet7Q



Ed Carpenter

20

Indianapolis IN

Ed Carpenter Racing

Chevy

Veteran

2C0

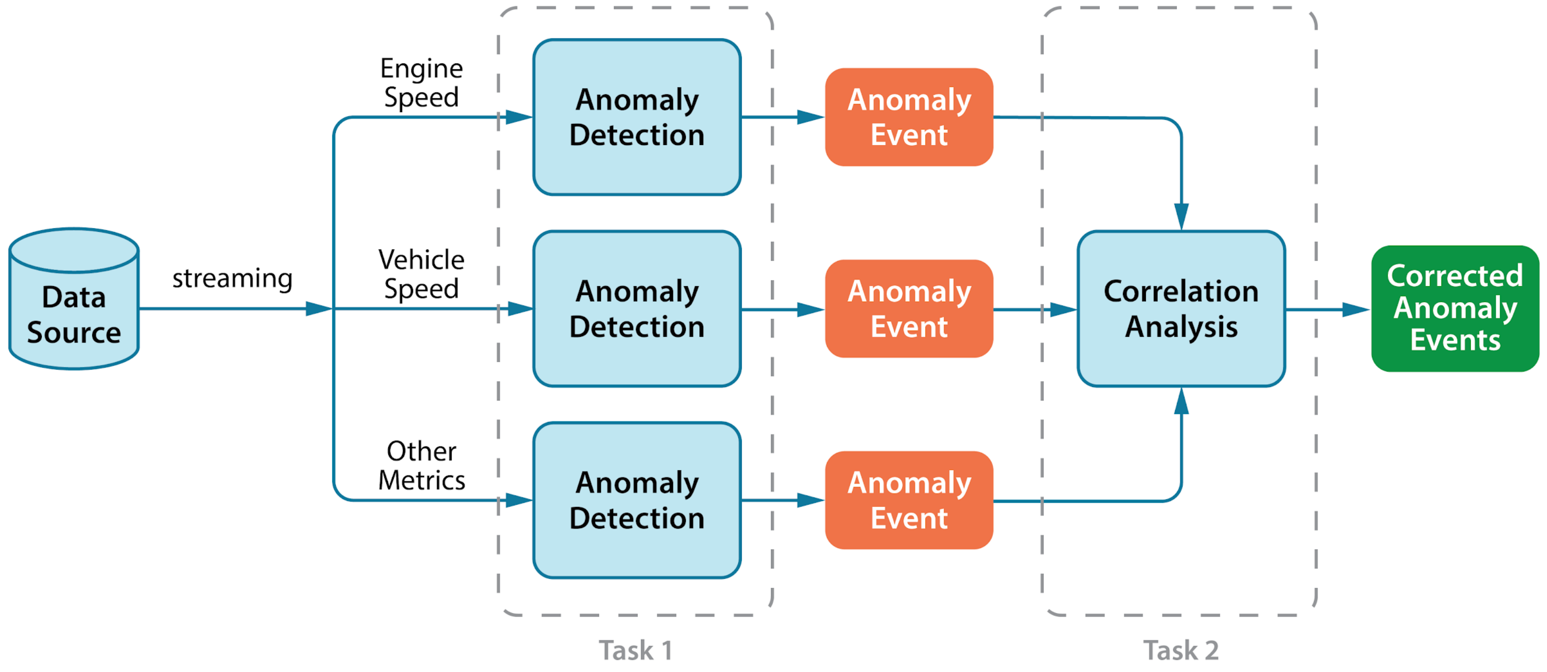
#1

Real-time data analysis is in need

"We want to know if it’s an expected event or a minor deviation that we need to be worried about. It helps race control people. "

" And we want to know the data corresponding to the anomaly; when car got into problems what kind of event it is and what is causing it."

Problem

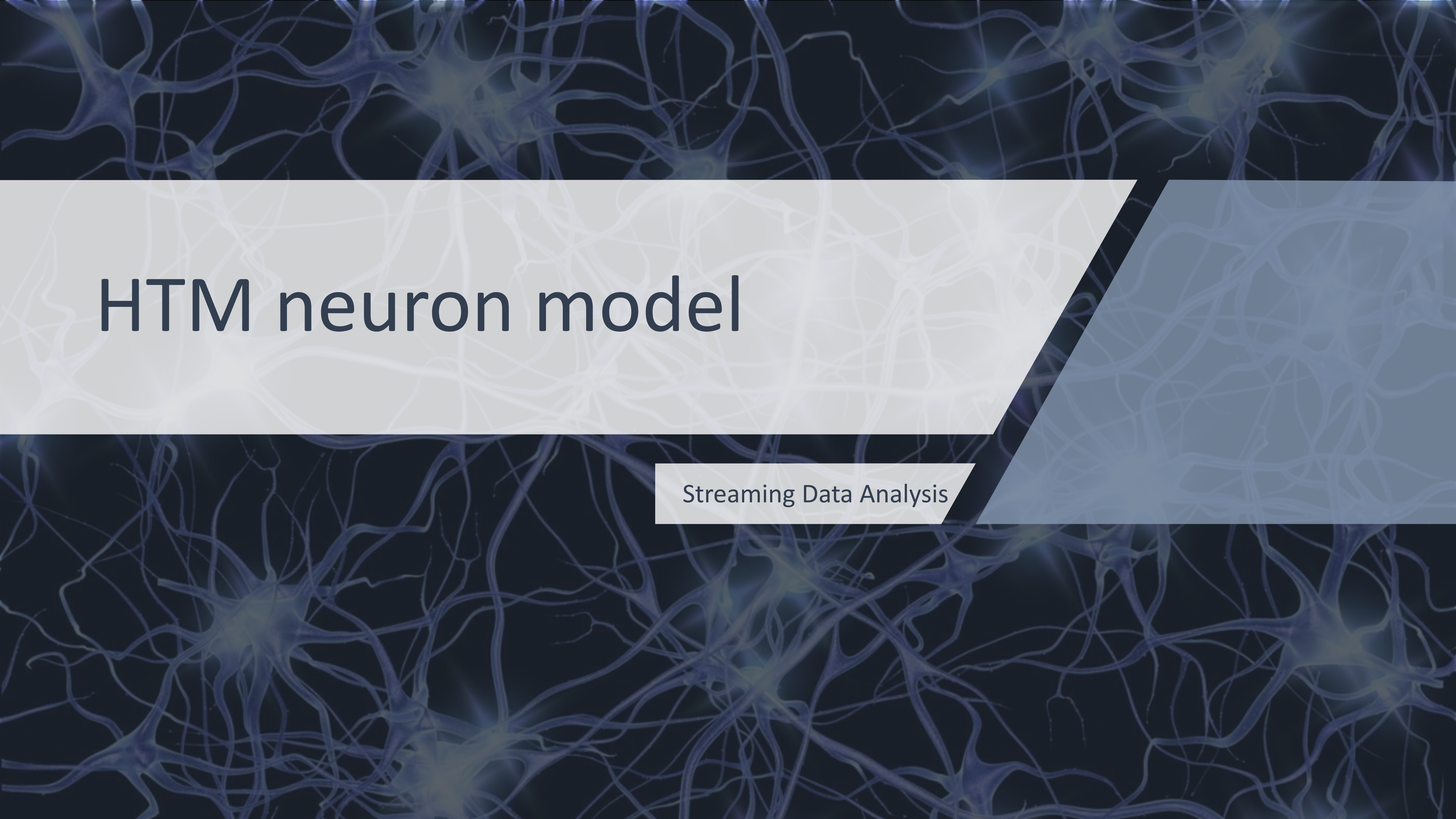


- Anomaly Detection: Learning algorithms themselves can only find the abnormal pattern in the data with best efforts under predefined assumption of what is “normal”.
- Correlation Analysis: Learning algorithms can find out the “events” from data and mine correlation relationship among the events.

Tasks

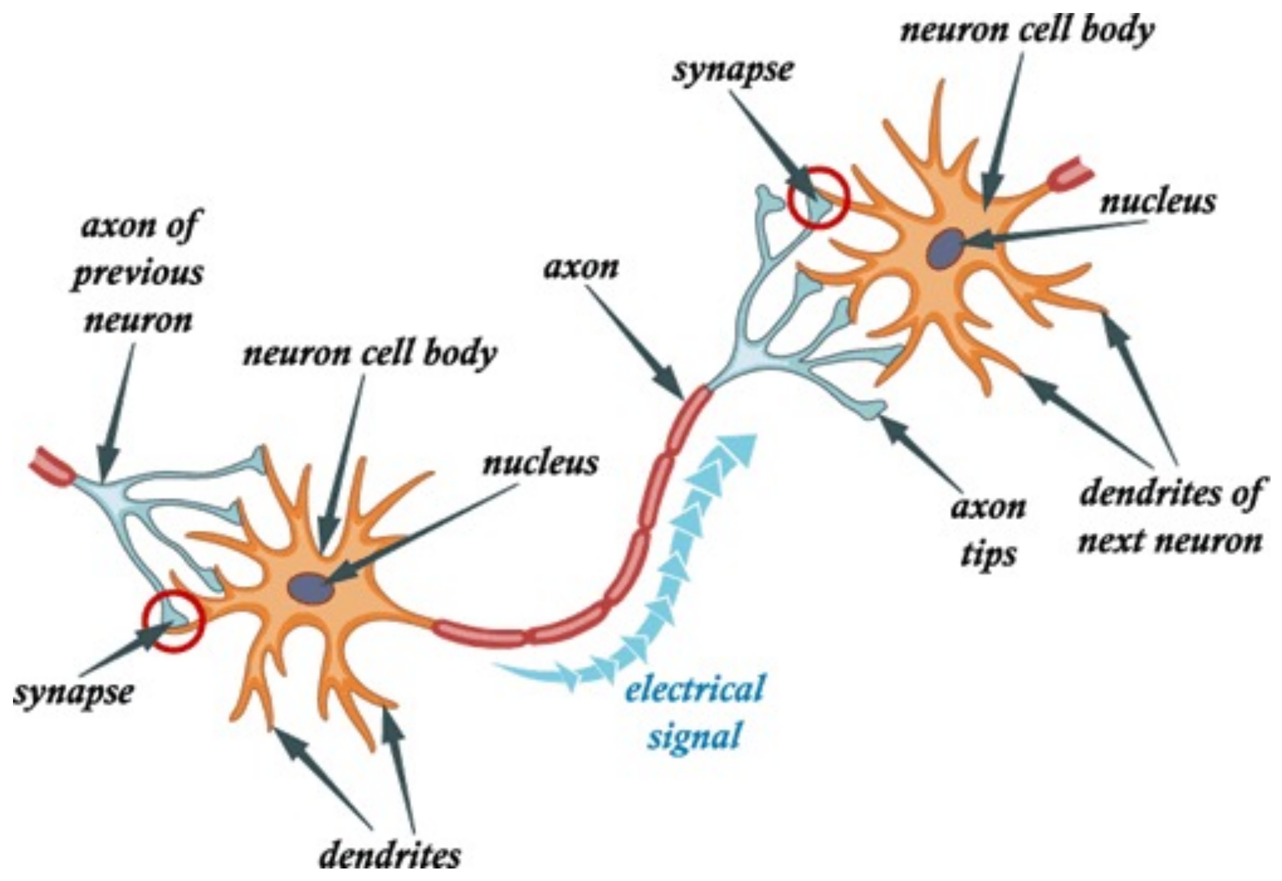
- Dataset
  - One of the Indianapolis 500 car races on the 28th of May 2017, which consists of 750 MB of data and a total of 3986170 records in the log file.
- Approach
  - Hierarchical Temporal Memory (HTM) algorithm, a state-of-the-art online algorithm to detect anomaly in real-time data source.

Preliminary Results



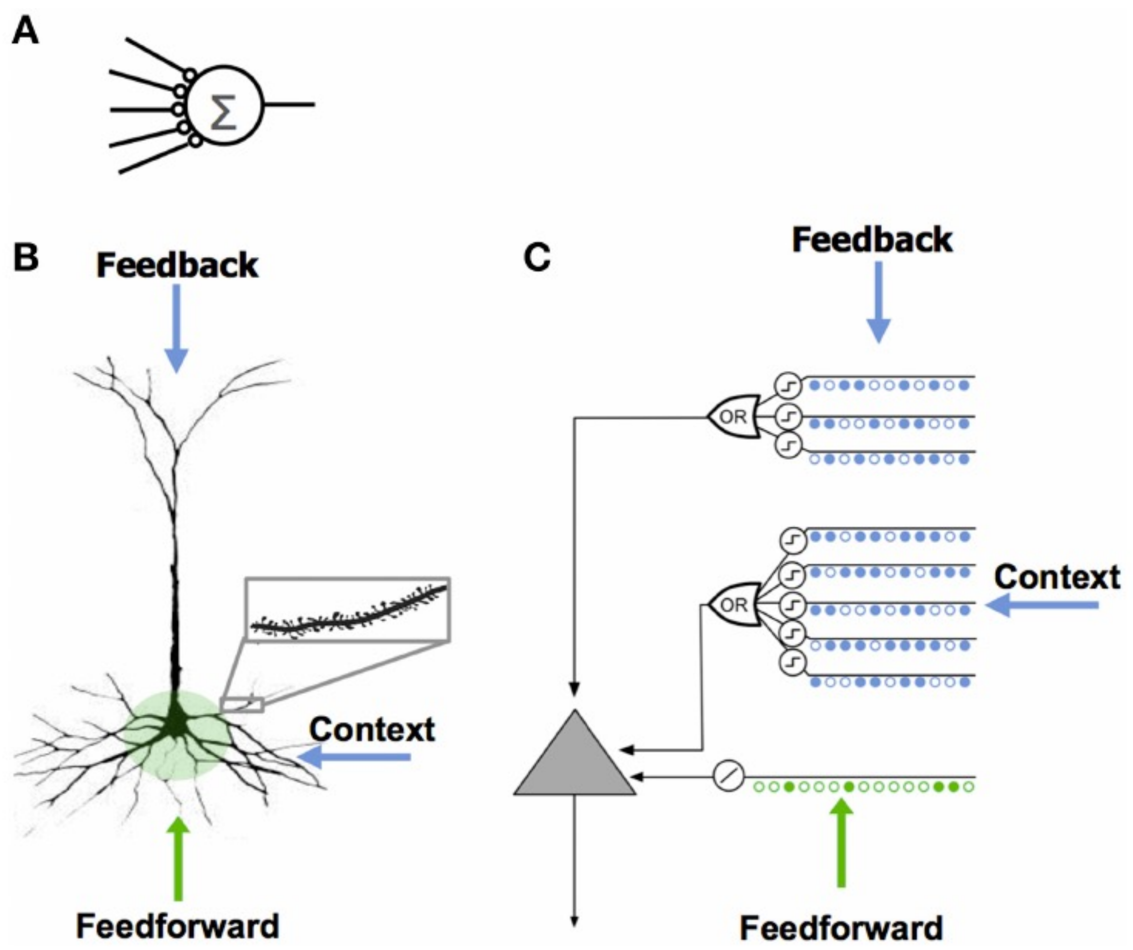
HTM neuron model

Streaming Data Analysis



- Neurons have specialized projections called dendrites and axons.
- Dendrites bring information to the cell body and axons take information away from the cell body.
- Information from one neuron flows to another neuron across a synapse. The synapse contains a small gap separating neurons.

Axon, Dendrite and Synapse



A. The neuron model used in most artificial neural networks has a few synapses and no dendrites
B. Different source of input: feedforward, feedback and context
C. HTM sequence memory models dendrites with an array of coincident detectors each with a set of synapses

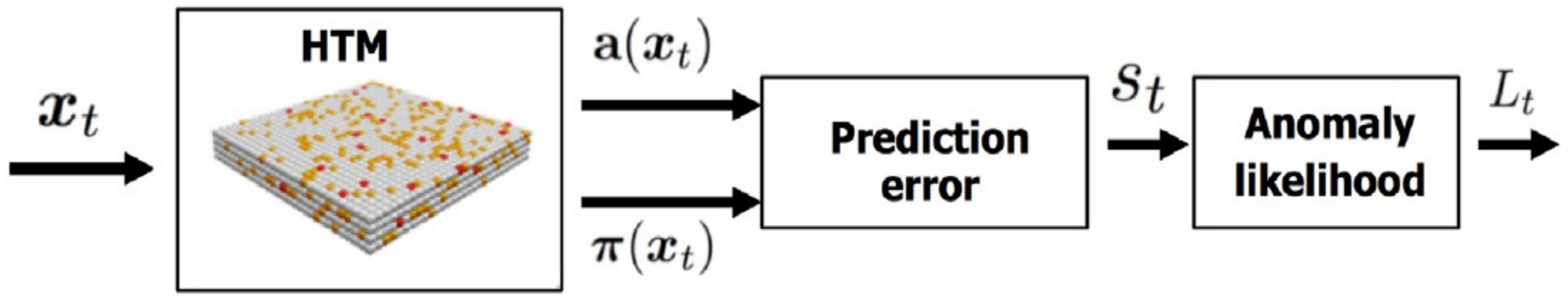
Hierarchical Temporal Memory(HTM)

- Hierarchical Temporal Memory (HTM) is a machine learning technology that aims to capture the structural and algorithmic properties of the neocortex.

- HTM models neurons, which are arranged in columns, in layers, in regions, and in a hierarchy. In this regard HTMs are a new form of neural network.
  - HTM consists of a layer of HTM neurons organized into a set of columns.
  - It models high-order sequences (sequences with long-term dependencies) using a composition of two separate sparse representations(spatial and temporal).
  - When receiving the next input, the network uses the difference between predicted input and the actual input to update its synaptic connections.

- HTMs are online learning and prediction machines that can be applied to many types of problems.

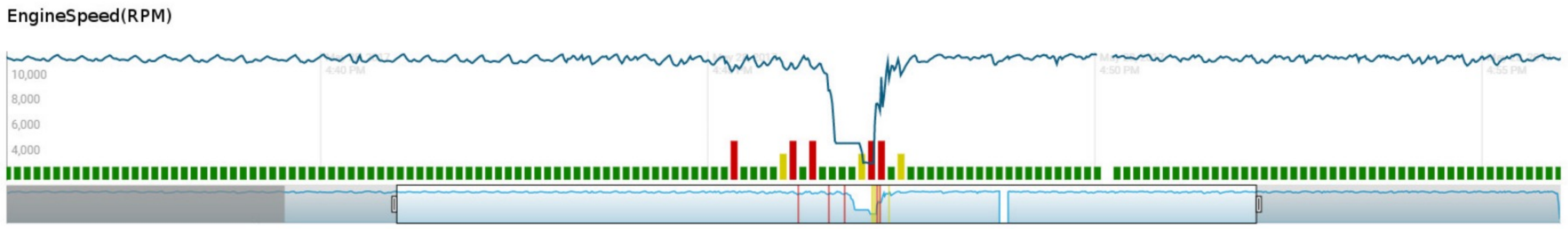
HTM



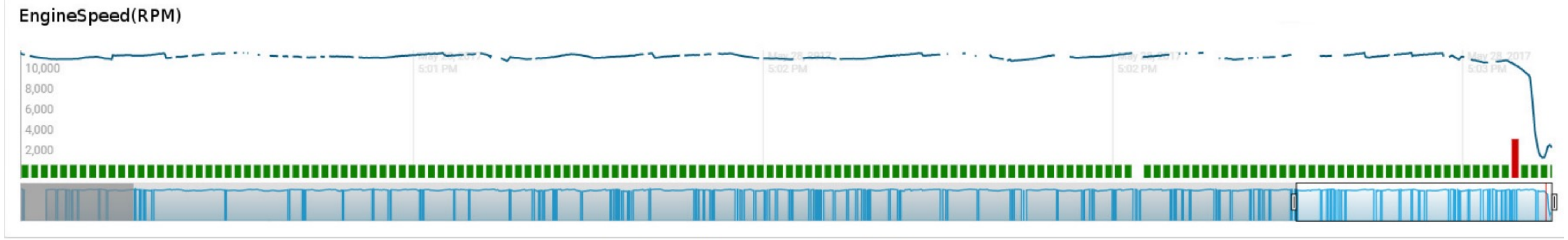
- The input time series x_t are fed to the HTM component. It models temporal patterns in $a(x_t)$ and output a prediction in $\pi(x_t)$.

- Then by building a statistical model on the prediction error, $\pi(x_t) - a(x_{t-1})$, anomaly likelihood score can be calculated on x_t.

Anomaly Detection Based on HTM



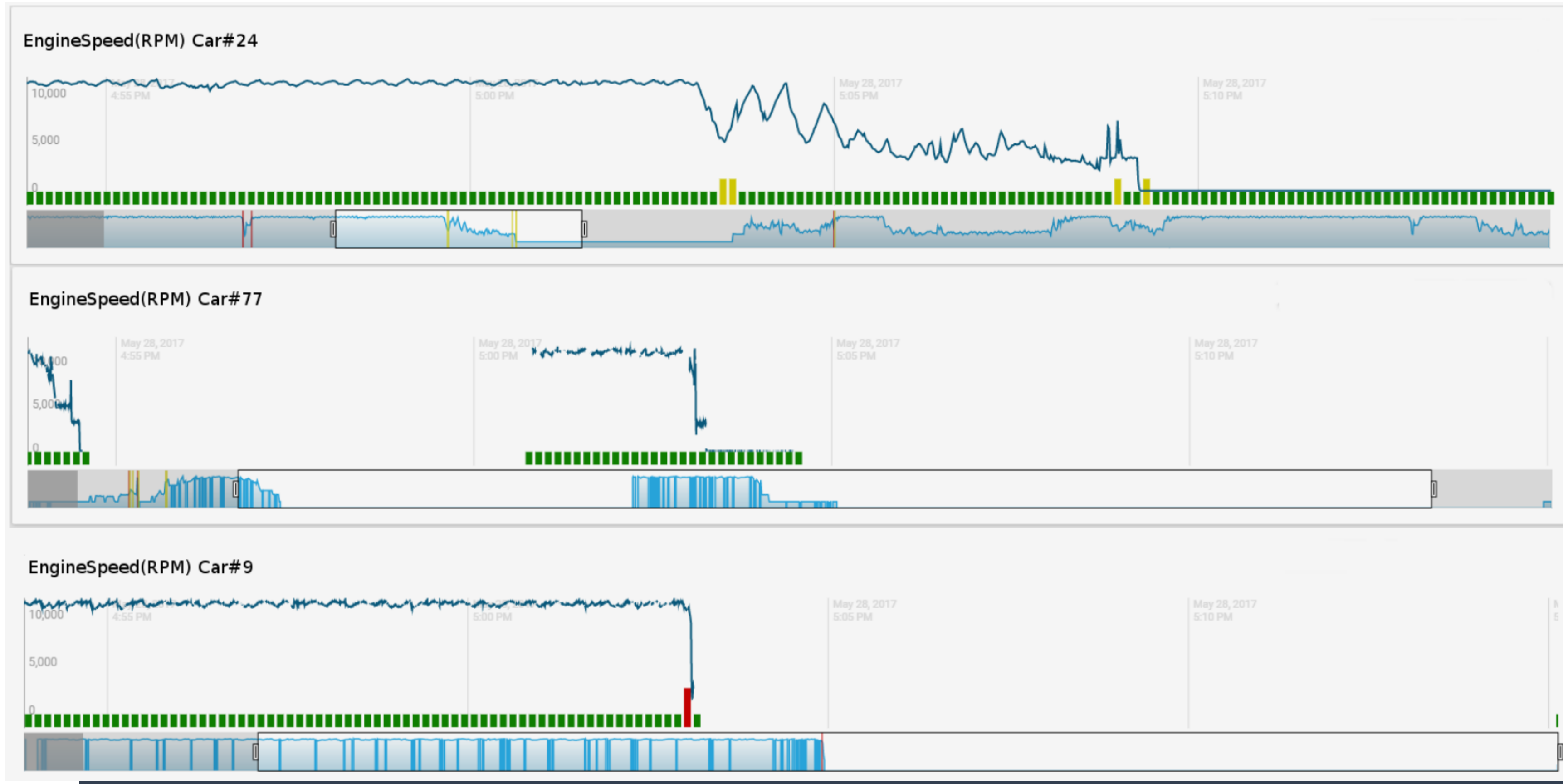
(a). Car#9 in middle of the race



(b). Car#9 at end of the race

- Detection algorithm used has the capability to detect certain type of anomaly in few seconds ahead of the time.

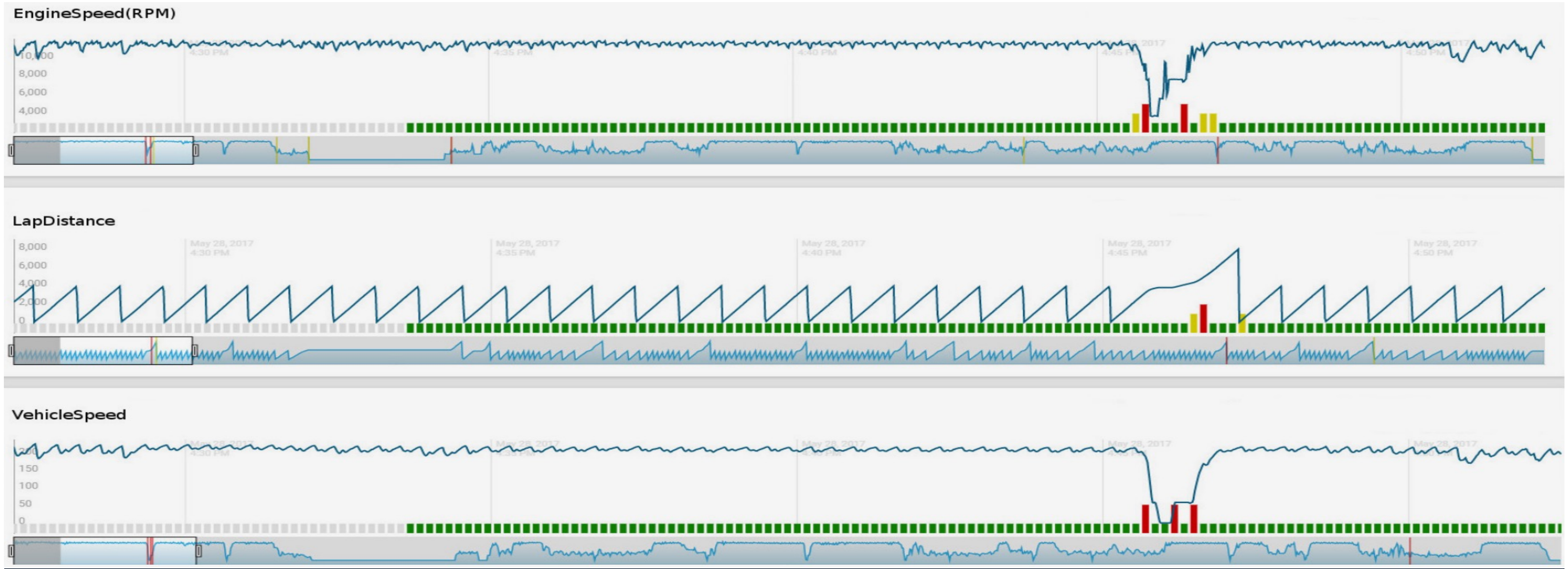
Anomaly Detection



Correlation of “events” among different cars: #9 #77 and #24

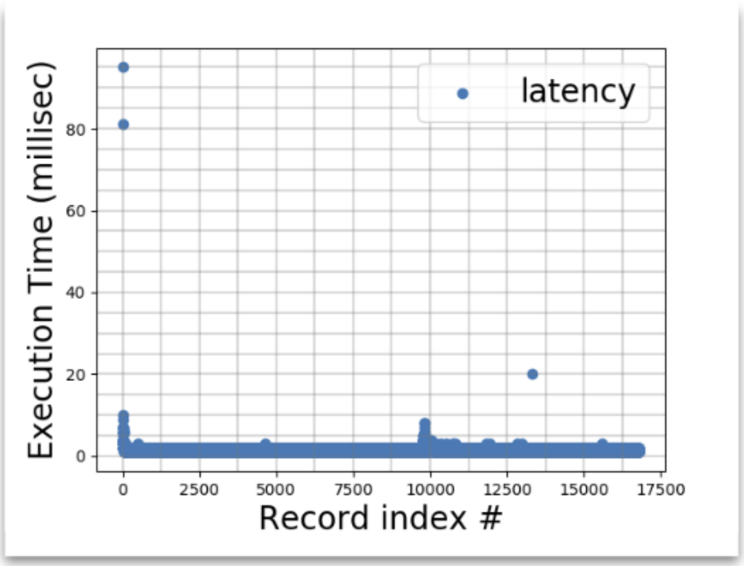
- The anomaly occurs around 15:03 pm, where the RPM of car #9 totally disappeared. In fact, car #9 got totaled due to a collision with car #77. The others cars, including car #24, all slowed down after the crash.

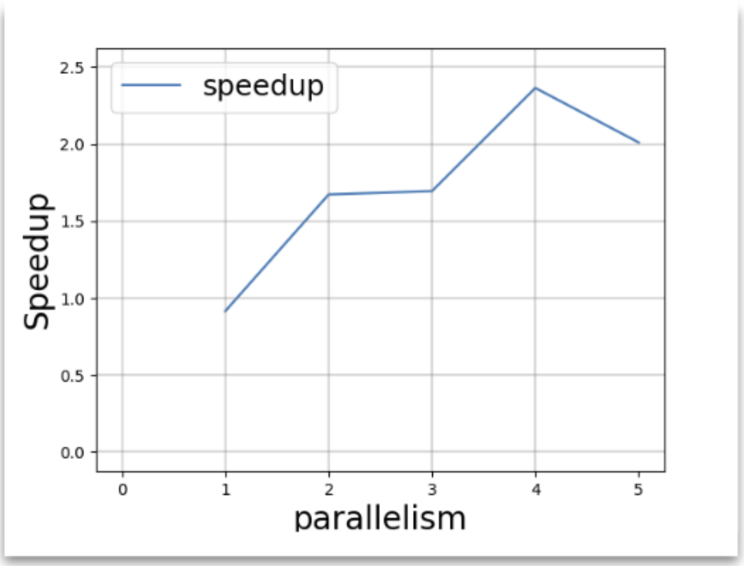
Correlation Analysis

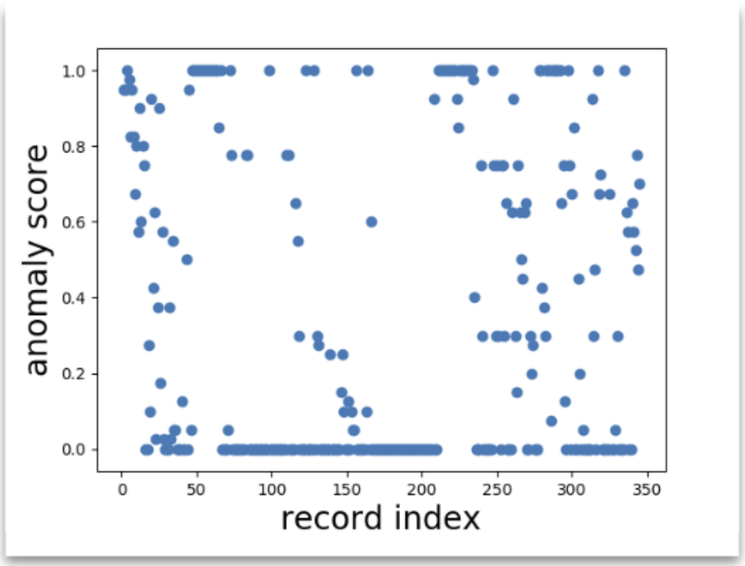


Correlation of “metrics” for one car #11.

Correlation Analysis







- The preliminary tests were done on the eRP data for car #9 (Scott Dixon), with ~17,300 records.
- The first plot shows the execution latency (time taken to predict anomaly on a single input record) to process each record in car #9 data on Apache Storm framework with parallelism of 1. Average execution time to predict an anomaly ~ 1.43 milliseconds.
- Plot 2 shows time speedup gain on increasing the parallelism in Storm to process car #9 telemetry. Speedup is the ratio of time to predict anomalies on a serial process to the time taken by running Storm at various parallelism levels on same data.
- To establish a sense of ground truth on labels and validate our approach, we deliberately inject anomalies at 2% data fraction (~ 345 data points) at known indexes. We inject speeds of 0.00 mph (absolute vehicle halt) at various indexes. The anomaly scores of each of the 345 data points is shown in plot 3. Anomaly score of 0.0 implies a normal event.

Validation

Future Work

DATASET

Collaborate with the domain experts, to build the anomaly dataset by human annotation.

ACCURACY & EFFICIENCY

Real-time anomaly detection is challenging, both accuracy and efficiency need to be improved.

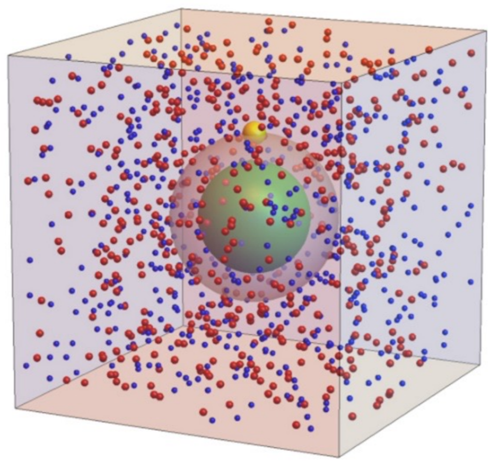
INTERPRETATION

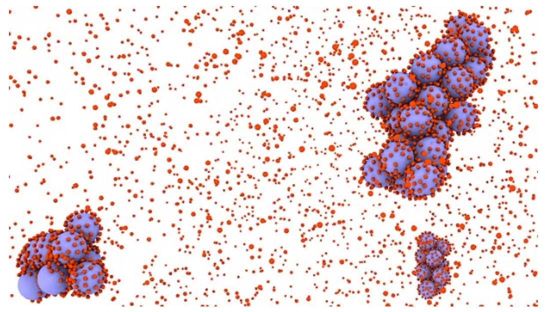
Correlation analysis provide a list of 'related' events. However, correlation does not imply casual relation. We need to collaborate with domain experts to interpret the results.

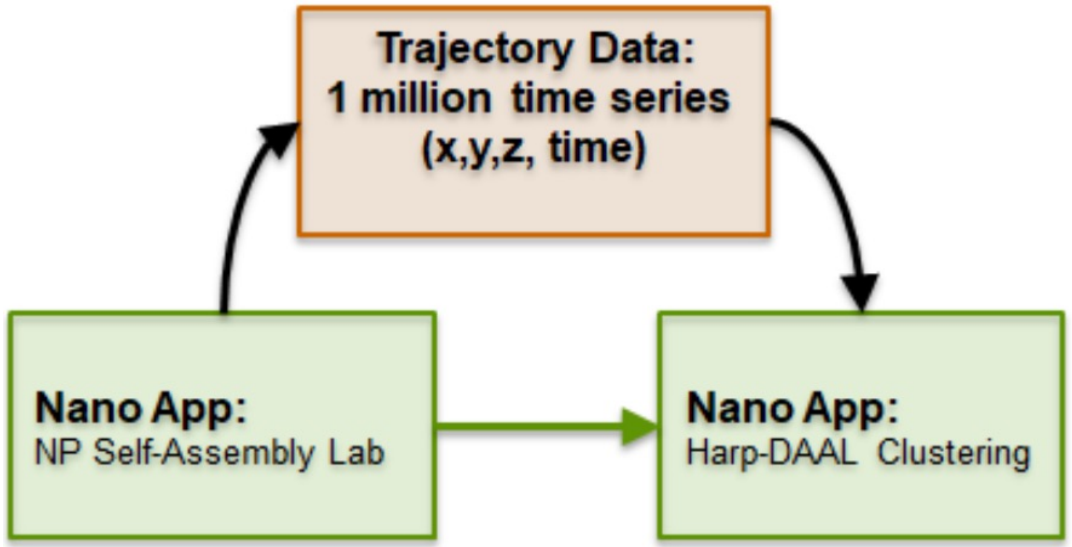
Software Tools: Harp-DAAL

For High Performance Data Analytics and Machine Learning

Nanoparticle Trajectories & Time Series

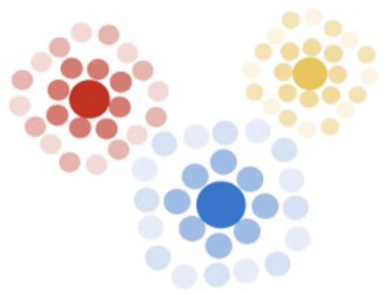


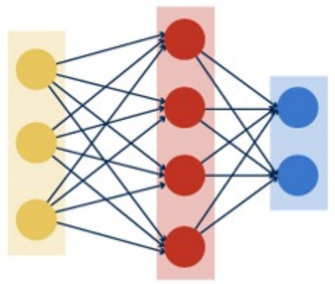


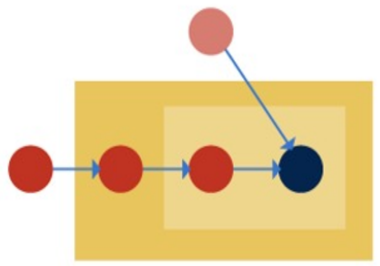


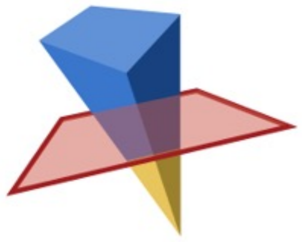
- These Harp-DAAL HPC kernels could well serve in accelerating the process of clustering the nanoBIO simulation results to obtain different nanoparticle trajectories.
- NP Self-Assembly Lab simulates assembly of nanoparticles and generates 1 million high dimensional trajectory data. Harp-DAAL takes the input data and runs clustering over different nanoparticle trajectories that can be visualized for NP distribution.

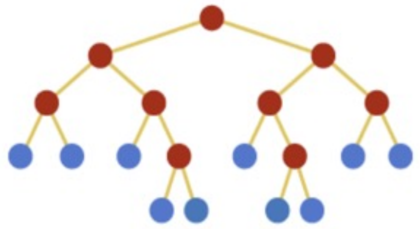
Data Analytics and Machine Learning

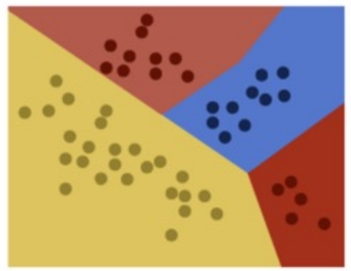

K-Means


Neural Networks

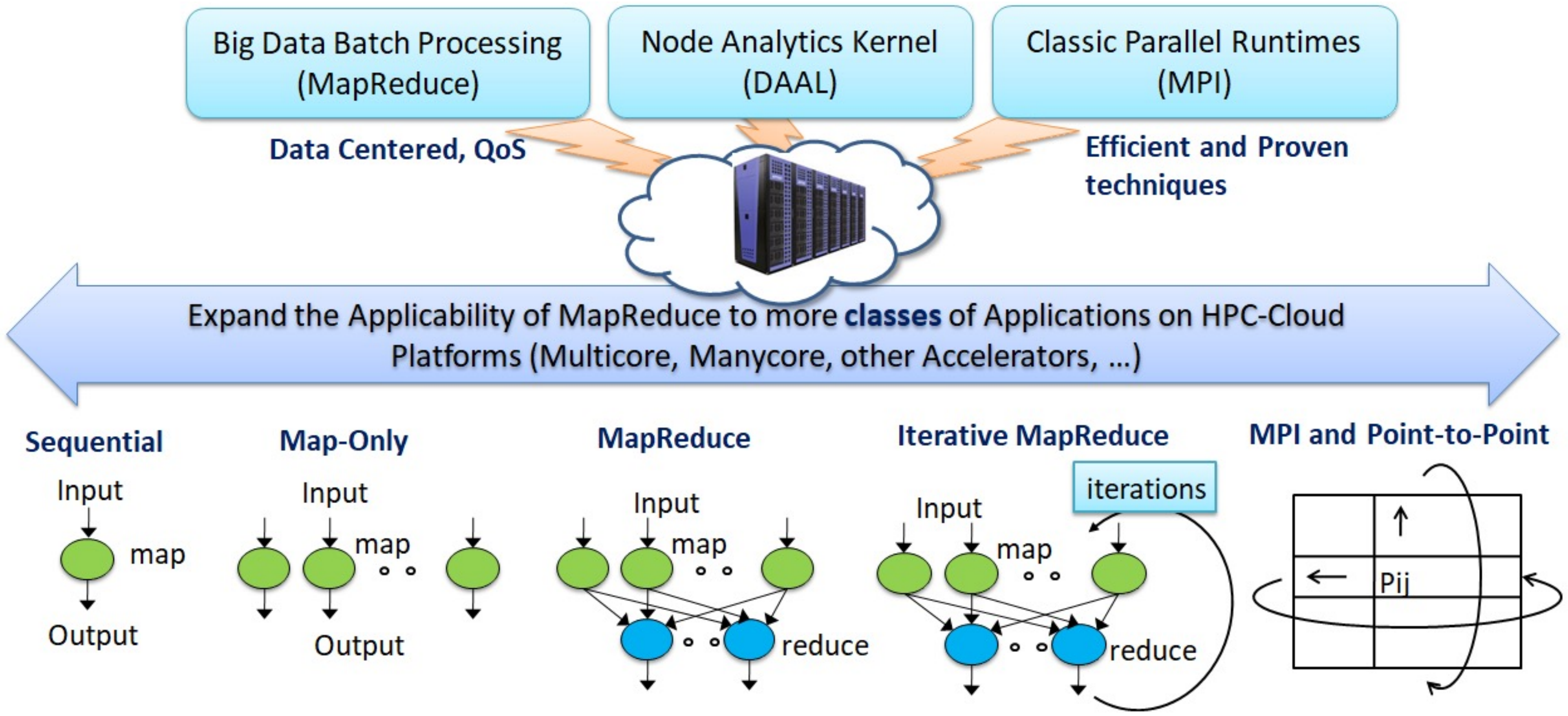

Latent Dirichlet Allocation


Support Vector Machine


RandomForest


Multi-Class Logistic Regression

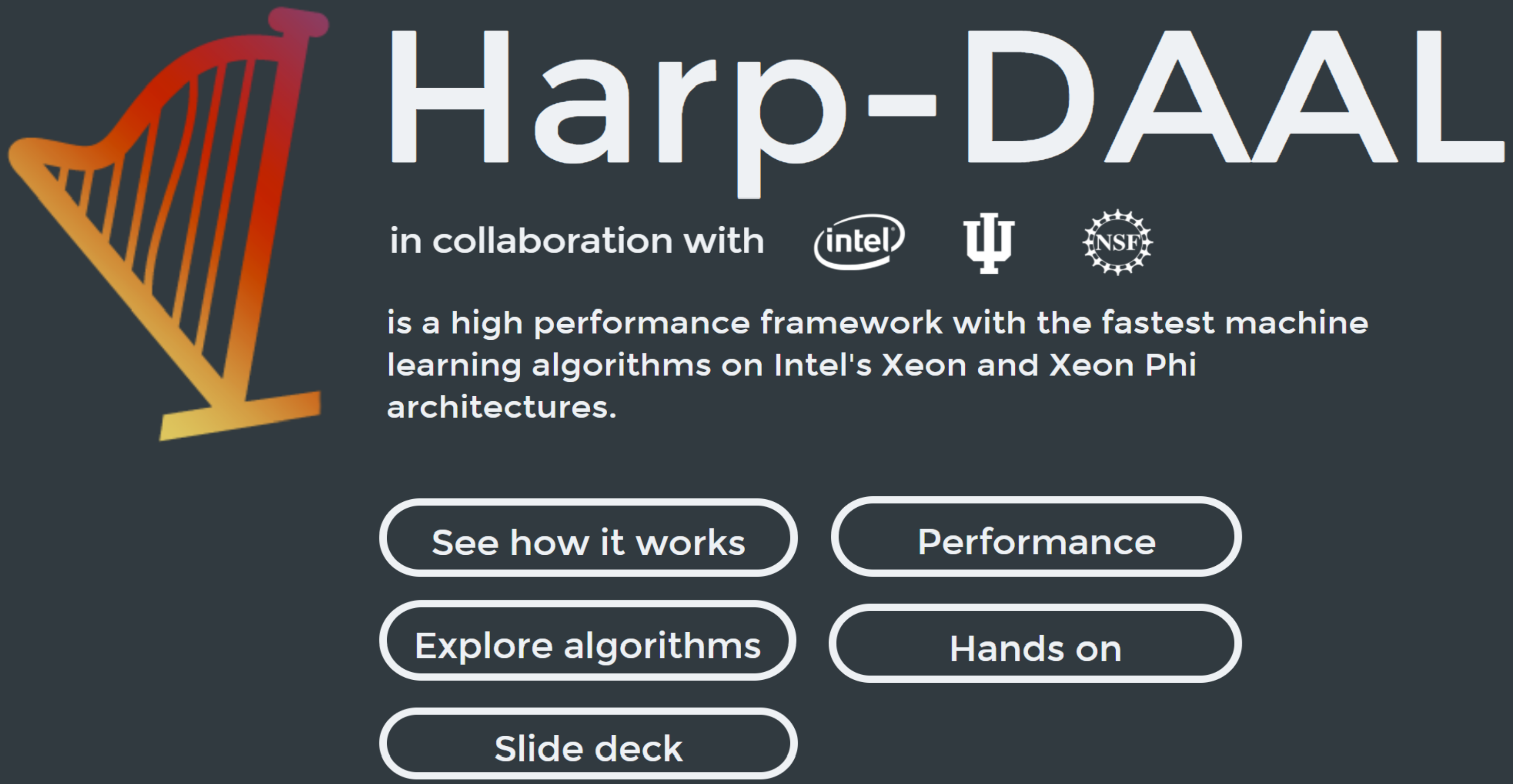
Harp/Harp-DAAL is an HPC-Cloud convergence framework that aims to automate ML as a service for both ease of use and scalability. Harp is designed to cover a full range of data-intensive computation from pleasingly parallel to machine learning and simulation.


Big Data Batch Processing (MapReduce)
Node Analytics Kernel (DAAL)
Classic Parallel Runtimes (MPI)
Data Centered, QoS
Efficient and Proven techniques
Expand the Applicability of MapReduce to more **classes** of Applications on HPC-Cloud Platforms (Multicore, Manycore, other Accelerators, …)
Sequential
Input
map
Output
Map-Only
Input
map
Output
MapReduce
Input
map
reduce
Iterative MapReduce
iterations
Input
map
reduce
MPI and Point-to-Point
Pij

Harp and Harp-DAAL allow our data analytics to be scalable and interoperable across a range of computing systems, including clouds (Azure, Amazon), clusters (Haswell, Knights Landing) and supercomputers (IU Big Red II).

Harp-DAAL | Cloud-HPC interoperable software for High Performance Data Analytics

Open Source Github Website (https://dsc-spidal.github.io/harp/)



Harp-DAAL

in collaboration with intel

is a high performance framework with the fastest machine learning algorithms on Intel's Xeon and Xeon Phi architectures.

We gratefully acknowledge support from NSF, IU and Intel Parallel Computing Center (IPCC) Grant.

Langshi Cheng , Bo Peng , Supun Kamburugamuve, Chathura Widanage, Sahil Tyagi, Miao Jiang, Selahattin Akkas, Andrey Nikolaev, Egor Smirnov, Dayle Smith, Lisa Smith

Intelligent Systems Engineering
School of Informatics and Computing
Indiana University

Acknowledgements