Snap ML:
Accelerated Machine Learning for Business

Haris Pozidis, Ph.D.
Principal Scientist, IBM Research

Outline

AI in Industry Applications
Snap Machine Learning
Business Use Cases
Data Discovery
Roadmap
AI Automation

AI in Business

Opportunities

Reduced **operational costs** thanks to process automation.

Increased **revenues** thanks to better productivity and enhanced user experiences.

Better **compliance** and reinforced **security**.

In general, the more data you feed, the more accurate are the results.

Business sectors have big data on transactions, customers, bills, money transfers, etc.

That is a perfect fit for machine learning!

Challenges

AI landscape is complex today and new capabilities are being offered daily.

Data scientists are scarce and demand increases.

Data **volume** is large & increasing. Must be able to **handle big data** to reap benefits.

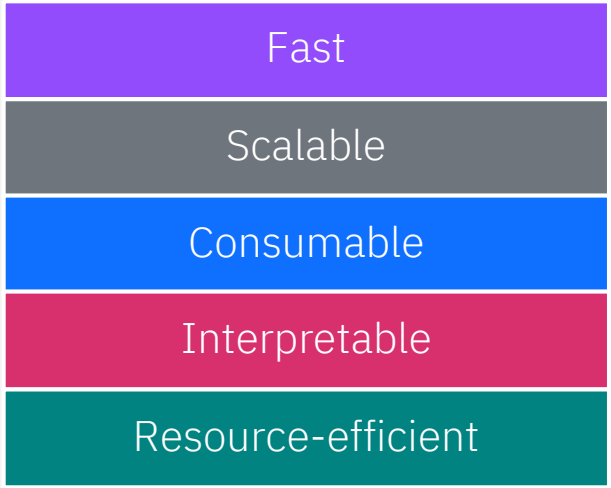
Data **velocity** is high & increases with amount of transactions. Need to react **fast** and accurately.

AI brings automation, less labor work, but should be **efficient** to offer cost benefits.

Snap ML: Accelerating Machine Learning

Snap ML is a set of compute libraries that transparently accelerate open source frameworks for training Machine Learning (ML) Models

It's main characteristics are:



Why Fast?
Performance matters for:
- online re-training of models
- model selection and hyper-parameter tuning
- fast adaptability to changes

Why Large-Scale?
Large datasets arise in business-critical applications: recommendation, credit fraud, advertising, space exploration, weather, etc.

Why Resource-Savvy?
Increased Resource Utilization. Less idle time. Less usage means savings, higher profit margin.

Why Interpretable?
Necessary feature for regulated industries where accountability is critical.

Snap ML Features

WMLCE 1.6.0 (1Q19)

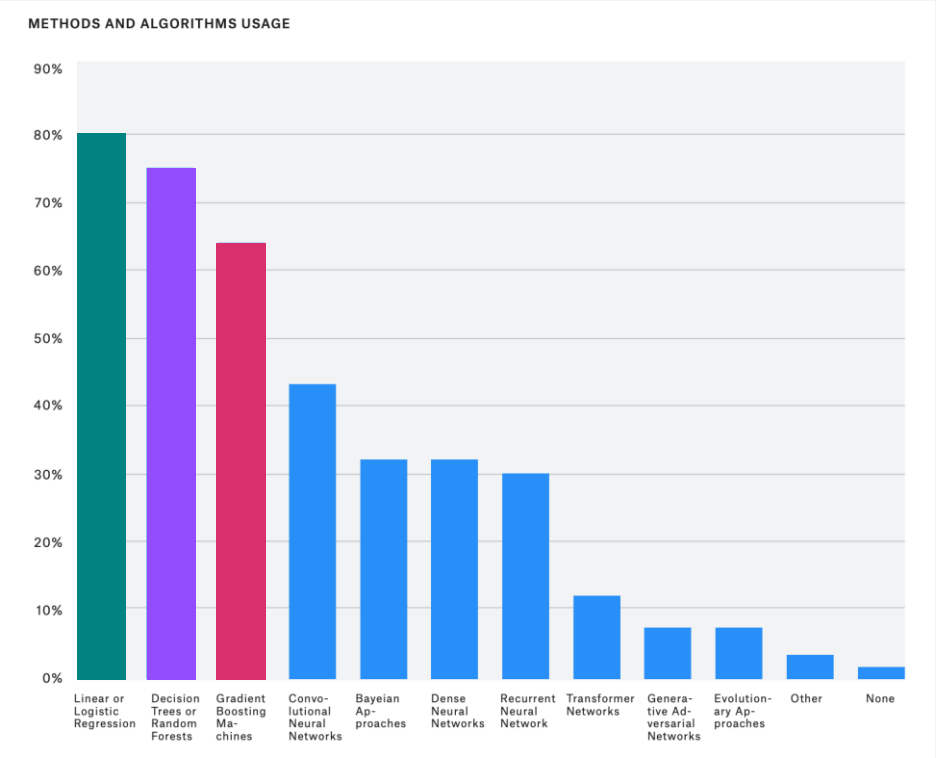
- Linear Regression
- Logistic Regression
- SVM

WMLCE 1.6.1 (2Q19)

- Decision Trees
- Random Forest

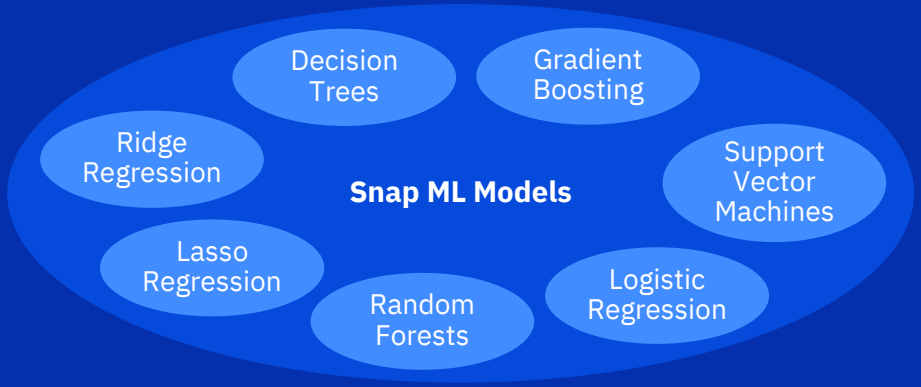
WMLCE 1.6.2 (4Q19)

- Boosting Machine



Source: Kaggle ML & DS Survey (Nov. 2019)

Why are GLMs/Trees useful?



Fast Training

GLMs can scale to datasets with billions of examples and/or features

Less tuning

Algorithms for training linear models involve much less parameters than more complex models (GBMs, NNs)

Interpretability

Linear models are naturally interpretable since they explicitly assign an importance to each input feature.

Tree models are interpretable as they explicitly illustrate the path to a decision.

Building blocks for Complex Models

More expressive models such as Gradient Boosting Machines heavily use Decision Trees as basic models.

Accelerating Decision Tree models naturally leads to faster GBM models as well.

Key Research Innovations

Distributed Training

Adaptive Distributed Newton (**ADN**) algorithm for scaling out GLMs across a cluster of CPUs/GPUs

ICML 2018
NeurIPS 2018, 2019

Multi-threaded Training

State-of-the-art solvers for training GLMs on multi-core, multi-socket CPUs.

MLSys 2018

GPU Acceleration

Twice-parallel, asynchronous stochastic coordinate descent (**TPA-SCD**) for training linear models on GPUs.

Duality-gap based heterogenous learning (**DuHL**) for when the dataset does not fit in GPU memory.

NIPS 2017, FGCS 2018

Decision Tree, Boosting Machine

Memory-efficient breadth-first search algorithm for training of decision trees, random forests and gradient boosting machines.

New boosting algorithm based on stochastic selection of base learner.

MLSys 2019

Usage Example

```
#Load data
from sklearn.datasets import load_svmlight_file
X, y = load_svmlight_file(filename)

# Train/Test split
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.25)

#Create a logistic regression model
#from sklearn.linear_model import LogisticRegression
from pai4sk.linear_model import LogisticRegression
lr = LogisticRegression(solver='liblinear')

#Training
lr.fit(X_train, y_train)

#inference
probs_test = lr.predict_proba(X_test)

#Evaluate logloss on test data
from sklearn.metrics import log_loss
test_loss = logloss(y_test, proba_test)
```

Use scikit-learn to train Logistic Regression

Snap ML is packaged as an IBM-extension to scikit-learn

"*Plug and play*" to achieve orders of magnitude faster training

Usage Example

```
#Load data
from sklearn.datasets import load_svmlight_file
X, y = load_svmlight_file(filename)

# Train/Test split
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.25)

#Create a logistic regression model
#from sklearn.linear_model import LogisticRegression
from pai4sk.linear_model import LogisticRegression
lr = LogisticRegression(use_gpu = True, device_ids = [0,1])

#Training
lr.fit(X_train, y_train)

#inference
probs_test = lr.predict_proba(X_test)

#Evaluate logloss on test data
from sklearn.metrics import log_loss
test_loss = logloss(y_test, proba_test)
```

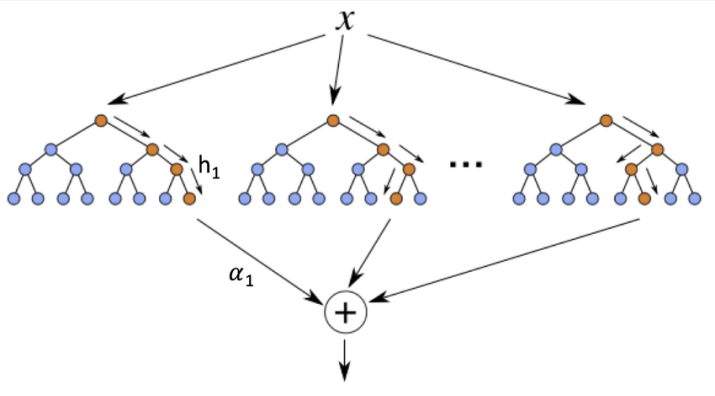
Use Snap ML to train Logistic Regression

Usage Example

```
#Load data
from sklearn.datasets import load_svmlight_file
X, y = load_svmlight_file(filename)

# Train/Test split
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.25)

#Create a logistic regression model
#from sklearn.linear_model import LogisticRegression
from pai4sk.linear_model import LogisticRegression
lr = LogisticRegression(use_gpu = True, device_ids = [0,1])

#Training
lr.fit(X_train, y_train)

#inference
probs_test = lr.predict_proba(X_test)

#Evaluate logloss on test data
from sklearn.metrics import log_loss
test_loss = logloss(y_test, proba_test)
```

- ✓ minor changes required to accelerate your scikit-learn code
- ✓ reuse additional functionalities of scikit-learn

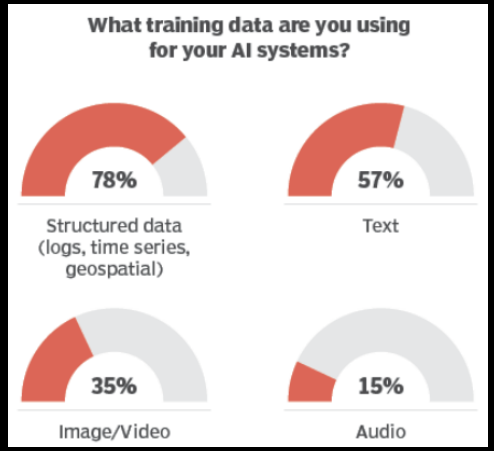
Gradient Boosting



GB models comprise an ensemble of decision trees, similar to a random forest (RF)

Unlike RF, they are trained, using stochastic gradient descent in a functional space

It is this training procedure that enables GB to generalize to unseen data exceptionally well



Majority of data in business is **Structured data**

Source: The State of AI in the Enterprise, O'Reilly, 2019

Deep neural networks achieve state-of-the-art accuracy on image, audio and NLP tasks

However, on **structured** datasets GB usually out-performs all other models in terms of accuracy

Snap ML Gradient Boosting targets **high accuracy** by a stochastic combination of base learners

Technical Innovation

Snap ML introduces a new algorithm for implementing Gradient Boosting

It also uses Decision Trees as base learners

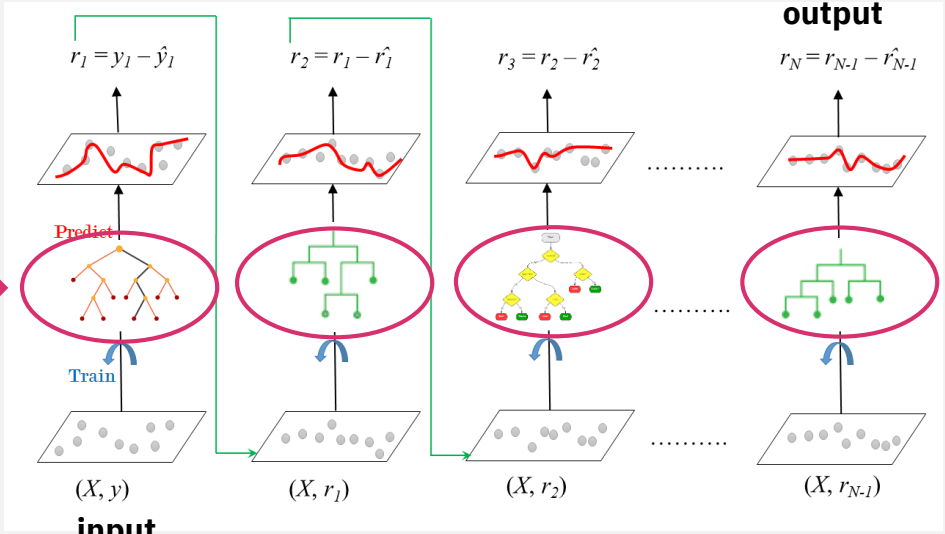
However, instead of fixing the tree structure, it selects from a **distribution of tree structures**

At each step, the tree structure is selected at random (other strategies are possible)

The main objective is to increase the generalization accuracy, while not sacrificing on performance

Tree structure can be different at each step

Principle of Gradient Boosting in Snap ML



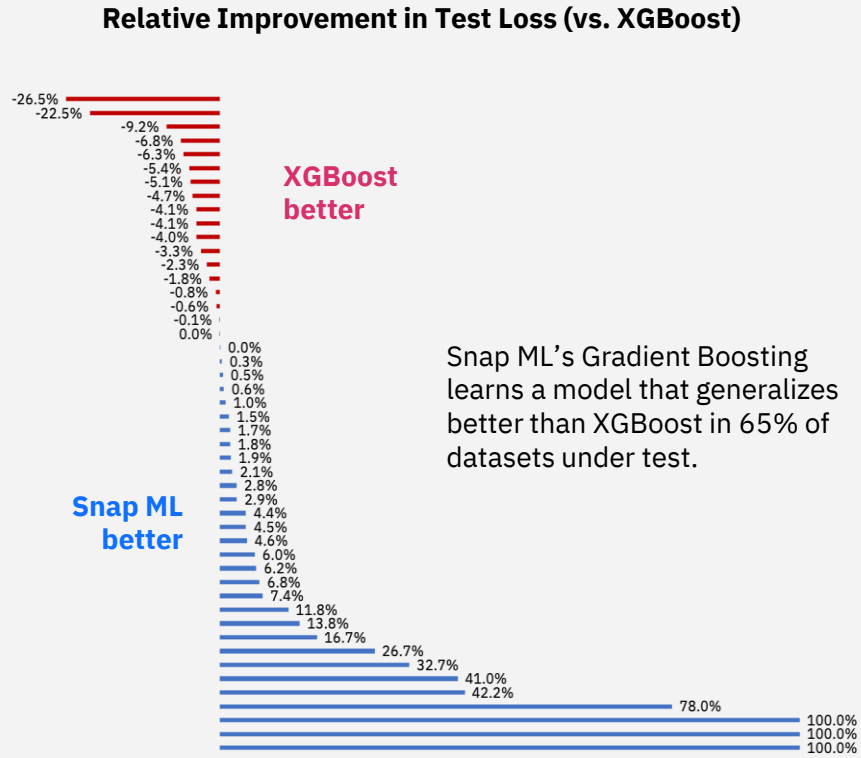
OpenML Benchmark

OpenML (www.openml.org) is a platform for collaborative data science.

We benchmarked **Snap ML's Gradient Boosting** against **XGBoost** and **LightGBM** using 48 binary classification datasets from OpenML.

Hyper-parameter tuning and generalization estimation was performed using nested cross-validation.

Snap ML provides **best-in-class accuracy** for a majority (65%) of datasets.



Snap ML's Gradient Boosting learns a model that generalizes better than XGBoost in 65% of datasets under test.

Use Cases

Financial Fraud Detection

Millions of financial transactions every hour, only a tiny fraction of which are fraudulent

Fraud detection requires connecting multiple transactions, often over several days

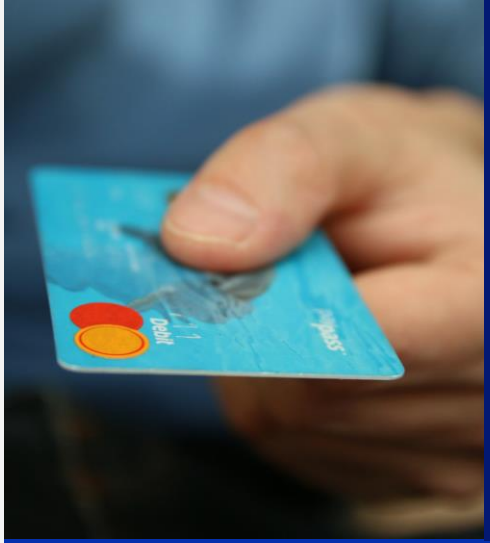
Opportunity:

- Preventing fraud *before* it occurs avoids costly efforts to reclaim lost capital
- Credit card fraud losses equal $22.8 B a year
  (Nilson Report 2017)

Challenges:

- Data **volume** and **velocity** is large
- Fraud must be detected **quickly** and **accurately**

Data source: https://www.kaggle.com/mlg-ulb/creditcardfraud

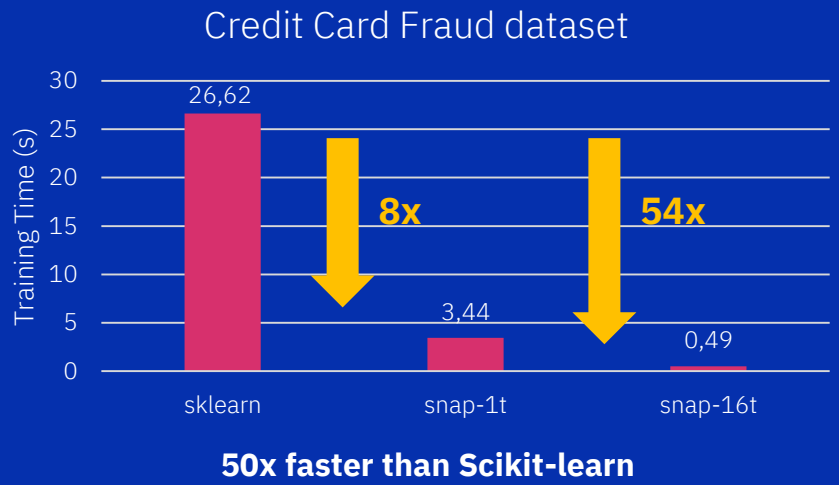


Model:
Logistic Regression

Dataset:
Credit card fraud (10x)

2,136,050 examples
28 features

Hardware:
Power9 CPU



50x faster than Scikit-learn

Price Optimization

Traditional pricing methods are insufficient to compete with online competitors who can better capture profits through careful price management.

AI is ideal for situations where a retailer needs to optimize across a wide assortment of items based on a variety of factors.

Opportunity:

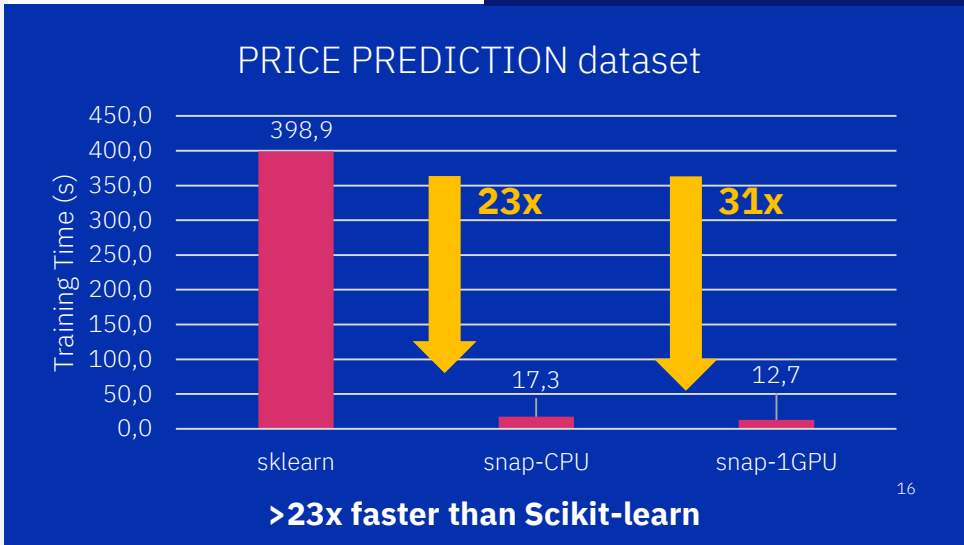
- AI can show retailers likely outcomes of different pricing strategies leading to better promotional offers, and higher sales.
- AI models can be used to determine the best price for each item using data on seasonality along with real-time inputs on inventory levels.

Model:
Ridge Regression

Dataset:
Price Prediction

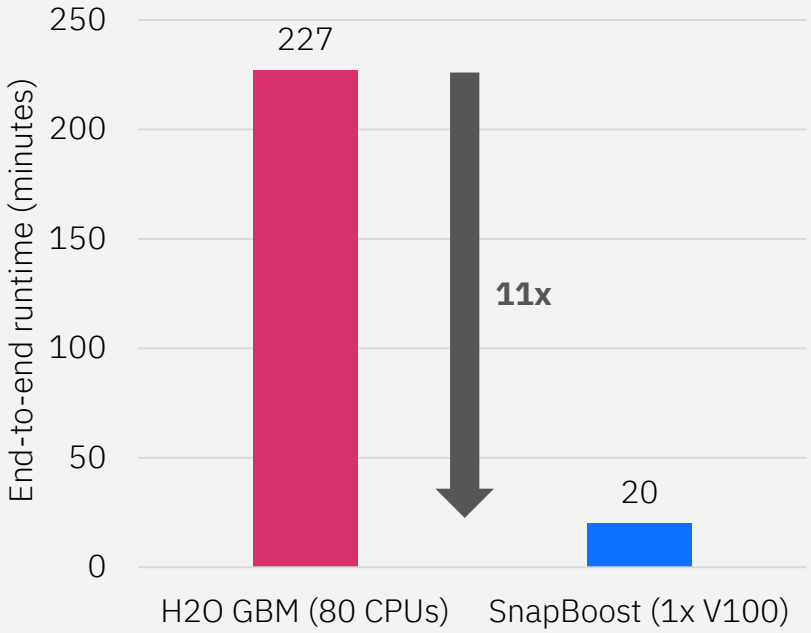
50M examples,
17k features

Hardware:
AC922 server,
V100 NVIDIA GPUs



Data source: https://www.kaggle.com/saitosean/mercari

Financial Services Application



Large FSS customer with a GBM workload

They are using GBM from H2O

Dataset / learning task are proprietary

The workload involves extensive hyper-parameter tuning

The customer evaluated **Snap ML Boosting** and found it delivered:

11x end-to-end speed-up

3% better accuracy

High-Energy Physics

Discovering exotic particles involves making numerous measurements at high energy particle colliders

Finding rare particles requires solving difficult signal-versus-background classification problems, hence machine-learning approaches are often used

Task:

- Distinguish between a signal process which produces supersymmetric particles and a background process which does not.

Challenges:

- Data **volume** and **velocity** can be very large

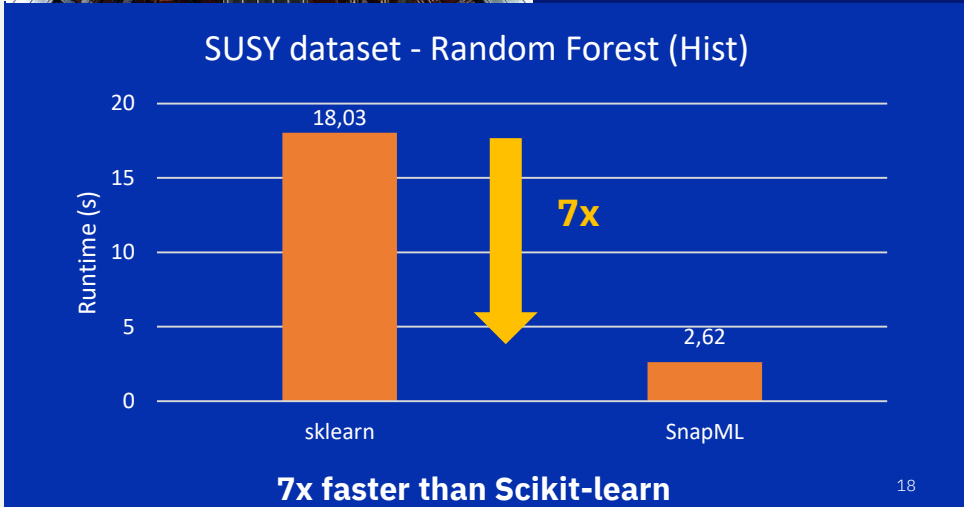
Data source: https://archive.ics.uci.edu/ml/datasets/SUSY

Model:
Random Forest

Dataset:
SUSY

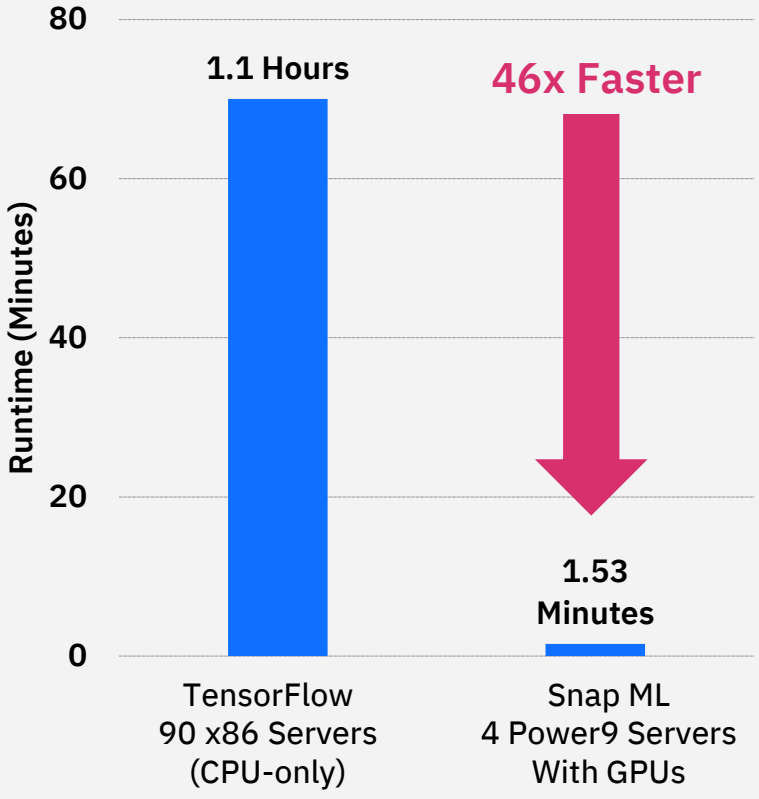
5,000,000 examples
18 features

Hardware:
Power9 CPU (sklearn)
Nvidia GPU (SnapML)



7x faster than Scikit-learn

Handling Terabyte-scale Datasets



For huge data-sets that do not fit into CPU memory, scikit-learn cannot be used for training.

Instead, one must turn to distributed frameworks like TensorFlow or Spark MLlib

Snap ML applications can be seamlessly distributed across a cluster **without any code change**

Dataset: Criteo Terabyte Click Logs (http://labs.criteo.com/2013/12/download-terabyte-click-logs/)
4 billion training examples, 1 million features
Model: Logistic Regression: TensorFlow vs Snap ML
Test LogLoss: 0.1293 (Google using Tensorflow), 0.1292 (Snap ML)
Platform: 89 CPU-only machines in Google using Tensorflow versus 4 AC922 servers (each 2 Power9 CPUs + 4 V100 GPUs) for Snap ML
Google data from this Google blog

Interpretability of Decisions

```
>> # Import data
>> from sklearn.datasets import load_svmlight_file
>> X_train, y_train = load_svmlight_file("mortgage-data")

>> # Import the LogisticRegression classifier from snap_ml
>> from snap_ml import LogisticRegression
>> lr = LogisticRegression(penalty = "l1", dual = False, use_gpu = True)

>> # Training
>> lr.fit(X_train, y_train)

>> # Model Analysis
>> important_features = lr.support

>> print(important_features)

[2,7,13,16]
```

Features {2,7,13,16} were the most important factors leading to a model decision

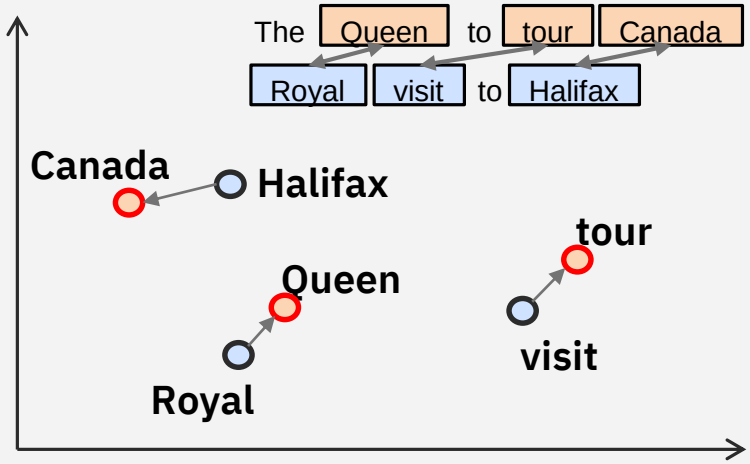
Illustrated here for a “Mortgage” dataset and a sparse logistic regression model

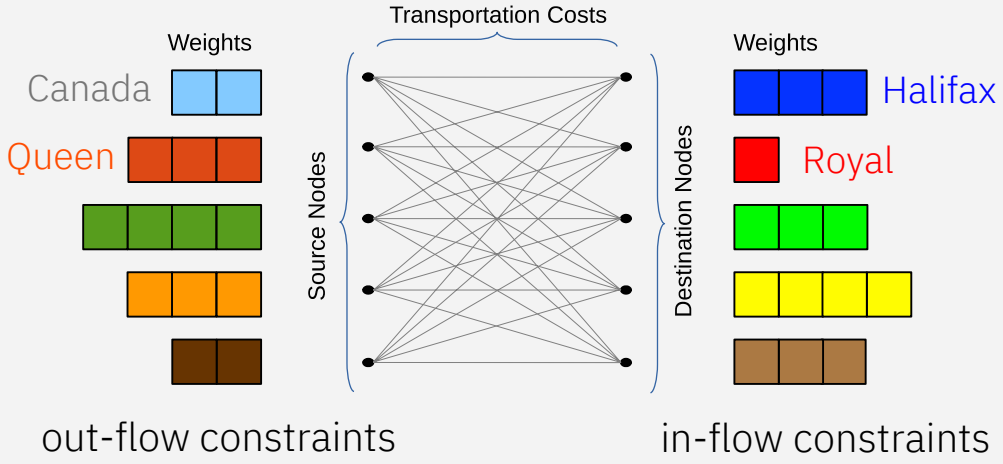
Snap ML helps the user to interpret the learned model

The “support” property informs the user which were the critical features leading to a decision

Data Discovery

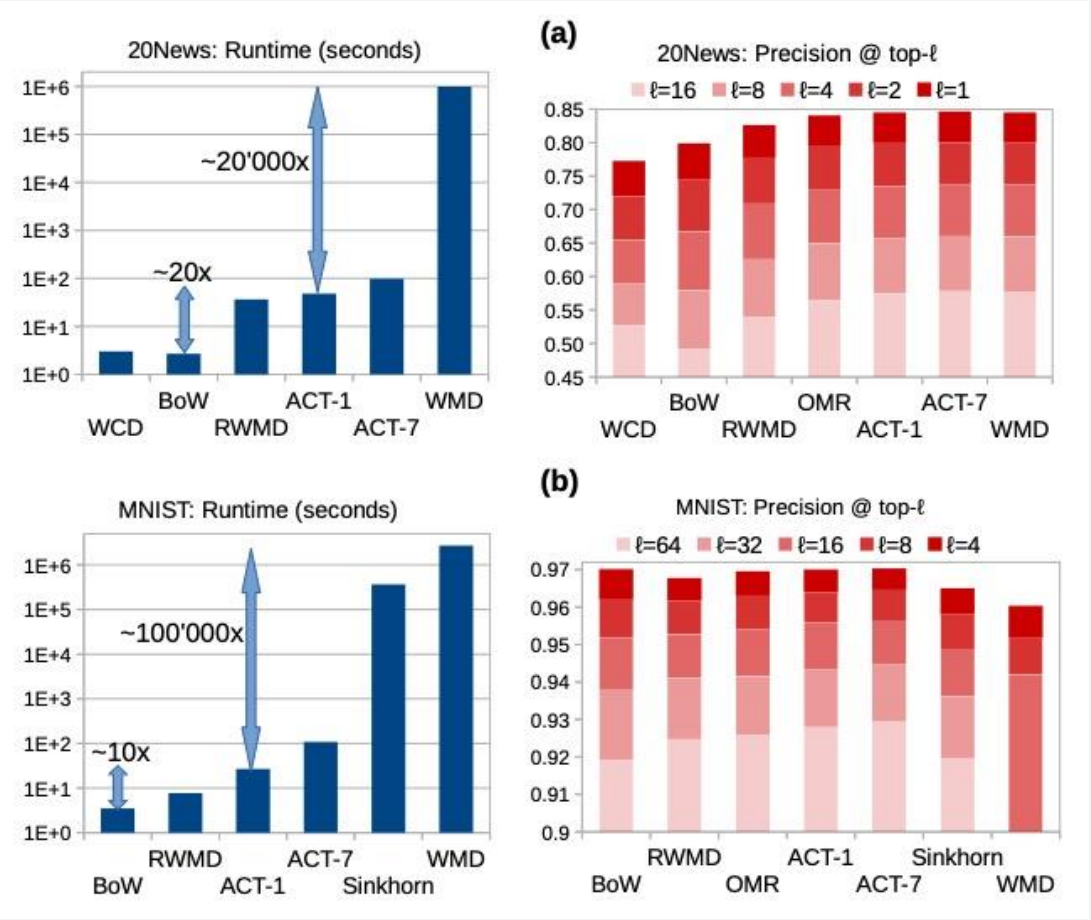
Earth/Word Mover's Distance: Discrete Wasserstein Distance





	Search Accuracy	Complexity	GPU friendly	Optimality
EMD/WMD	Very high	$h^3 \log h$	No	Yes
Sinkhorn	Very high	$(h^2 \log h)\ /\epsilon^2$	Yes	Within ϵ
RWMD	High	h	Yes	No
Our Work	Very high	hk	Yes	No

Experiments: Runtime vs Nearest-Neighbors-Search Accuracy



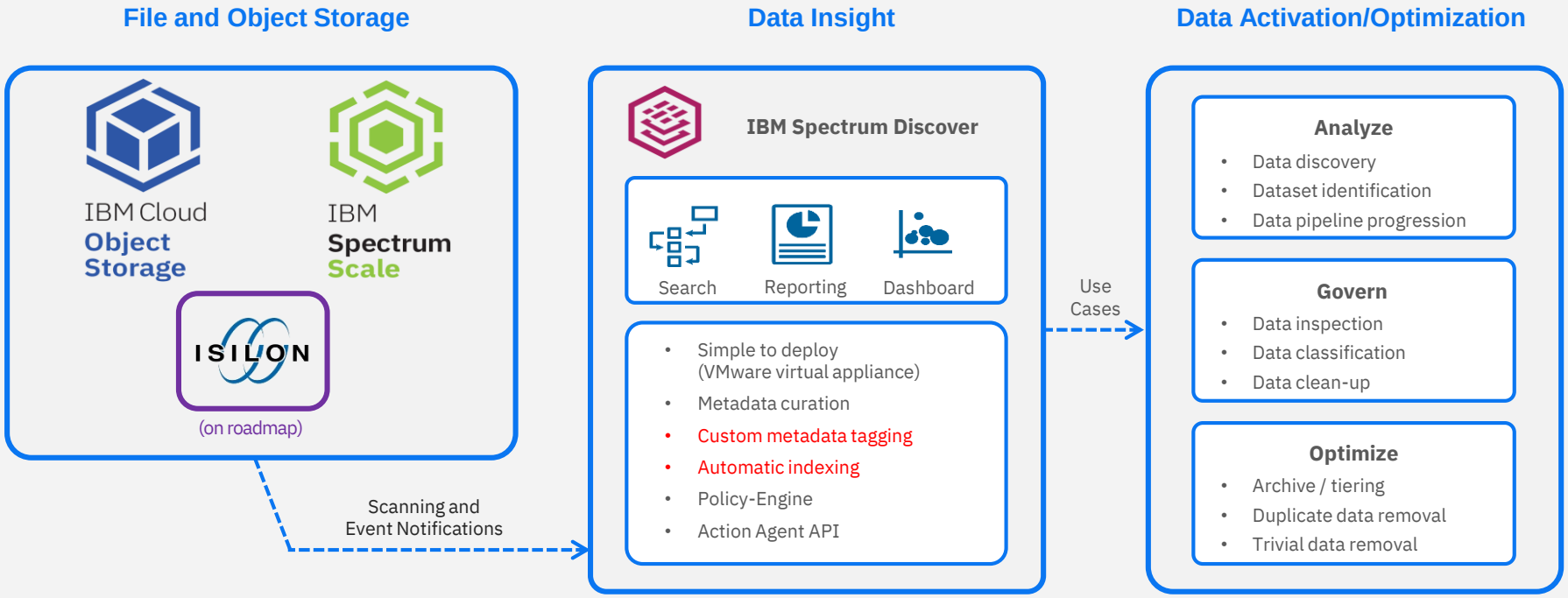
- ✓ ACT effective on sparse as well as dense, low- as well as high-dimensional datasets
- ✓ 20’000 faster than WMD and matches its search accuracy on 20 Newsgroups
- ✓ 10’000 faster and offers a slightly higher search accuracy than Sinkhorn on MNIST

20News: high-dimensional, sparse histograms
MNIST: two-dimensional, dense histograms

WCD: Word centroid distance (Euclidean)
BoW: Bag-of-Words (Cosine similarity)
WMD: Word Mover’s Distance (Kusner et al.)
RWMD: Relaxed Word Mover’s Distance

OMR and **ACT-k**: the new algorithms

IBM Spectrum Discover Overview



Use case – Search based on query document

Query Document:

QUERY document: Federer breezes into semi-finals
Label: Tennis

Roger Federer reached the last four of the Qatar Open with an easy 6-1 6-2 win over seventh seed Feliciano Lopez... Russian Nikolay Davydenko... meanwhile, upset French third seed Sebastien Grosjean 2-6 6-3 6-2. Fabrice Santoro ... France when he was forced to retire when 6-2 3-0 down to Albert Costa.. Ivan Ljubicic after the sixth seed beat Rafael Nadal 6-2 6-7 (3/7) 6-3.

User: show me the k most similar documents to my query document

Neighbor Documents:

NEIGHBOR document: Johansson takes Adelaide victory
Label: Tennis

... top contender at the Australian Open, which starts on 17 January. "I believe men's tennis is all about holding serve and if he's playing like that on his own serve I don't see how guys are going to break him...

NEIGHBOR document: Federer claims Dubai crown
Label: Tennis

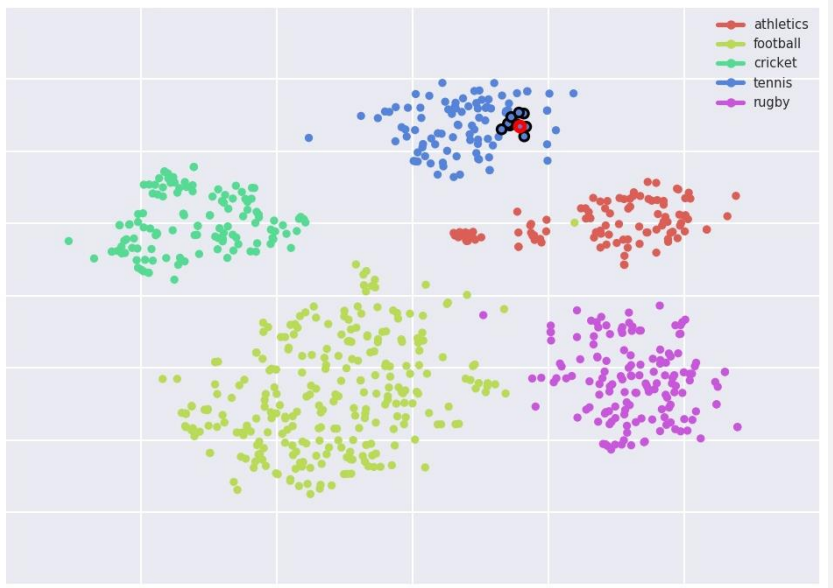
Federer claims Dubai ... Ivan Ljubicic.Top seed Federer looked to be on course for a easy victory when he thumped the eighth seed 6-1 in the first set. But Ljubicic, who beat Tim Henman in the last ...

NEIGHBOR document: Clijsters hope on Aussie Open
Label: Tennis

...has pulled out of January's Australian Open... Kim 17 January, Clijsters... French Open, is another absentee because of an injured left foot.

NEIGHBOR document: Officials respond in court row
Label: Tennis

Australia's Geoff Pollard rejected his comments... The tournament starts on 17 January... beat Roger Federer on in the Davis Cup in 2003."



Offering & Roadmap

Snap ML Evolution

1.5.2 06/18
- GLMs: Logistic regression, SVM, linear regression
- Local (single-node), MPI, Spark Python APIs
- Multi-GPU training, multi-threaded CPU training

1.5.3 09/18
- GLMs: sparse linear models
- Multiclass classification
- Visualization extensions

1.5.4 11/18
- CPU performance optimizations
- Fast (multi-threaded) inference
- Fast SVM Light data loader
- Seemless acceleration of scikit-learn : <pai4sk>

1.6.0 03/19
- Decision Tree: CPU single-node
- Random Forest: CPU single-node
- RAPIDS cuDF integration for fast pre-processing
- RAPIDS cuML integration

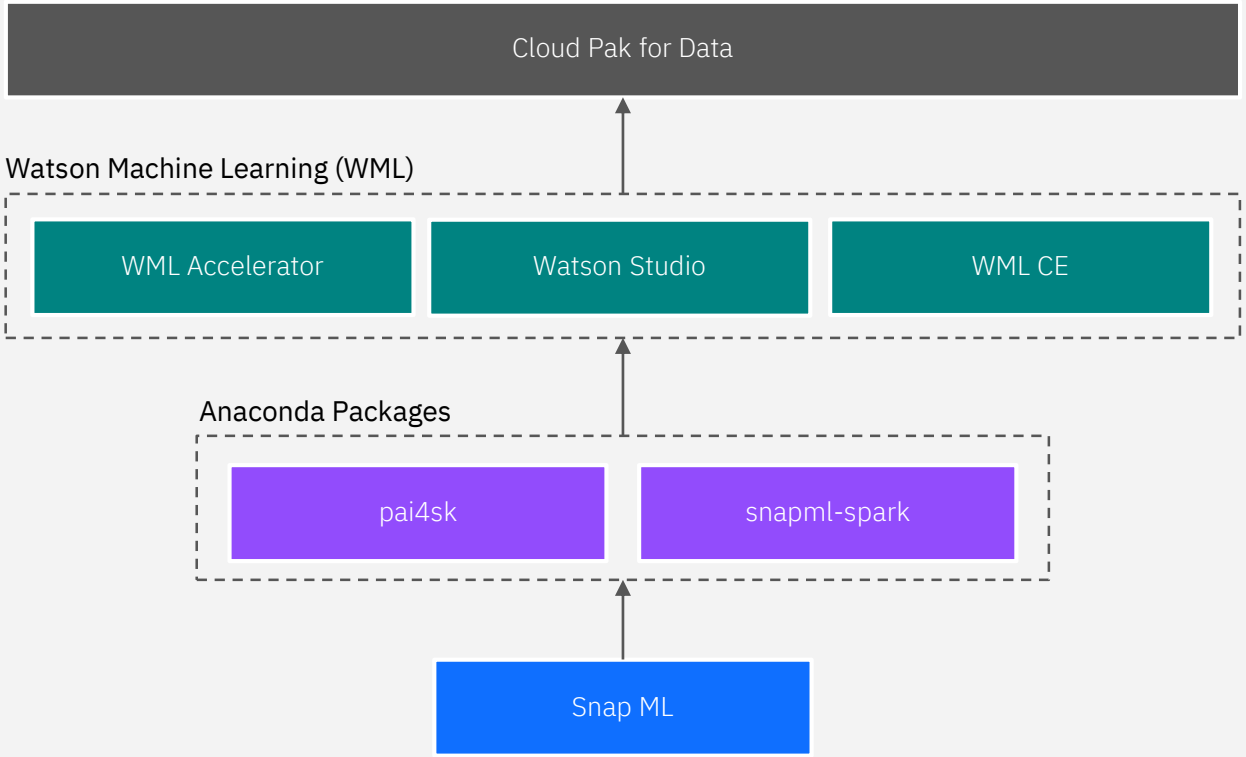
1.6.1 06/19
- Dmlc XGBoost Tech Preview
- Deep integration with Spark for Data Movement
- Nvidia RAPIDS updates: cuDF and cuML 0.7x

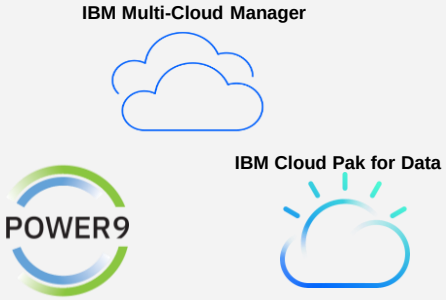
1.6.2 10/19
- SnapBoost Tech Preview (CPU solver)
- GPU-accelerated SnapBoost Tech Preview
- Sample weighting for imbalanced datasets (Decision Tree, Random Forest)

Roadmap

- Gradient boosting continuous improvement
- Distributed tree-based models
- GPU-accelerated tree-based models
- Sparse data support for tree models
- Distributed data pre-processing

Snap ML Integration





Where to get / How to try Snap ML

Available through WML-CE 1.6.2

- Download it for free from https://developer.ibm.com/linuxonpower/deep-learning-powerai/releases/
- Available in Power Systems currently (on-prem & in the cloud)
- Try it on now on Power servers through Cirrascale: http://snapml.cirrascale.com NEW
- Delivery through Conda packaging
- Public Documentation: https://ibmsoe.github.io/snap-ml-doc/index.html
- Blogs and articles:
  https://medium.com/@sumitg_16893/snap-ml-2x-faster-machine-learning-than-scikit-learn-c3529a1a6172
  https://www.ibm.com/blogs/systems/power-snapml-watson-machine-learning/
  https://developer.ibm.com/linuxonpower/2018/12/02/running-snapml-applications-with-ibm-powerai-enterprise-1-1-2/
  https://developer.ibm.com/series/learn-watson-machine-learning-accelerator/
  https://developer.ibm.com/series/snapml-on-powerai/

In a Nutshell

Snap ML is a new framework for training of popular ML models.

It is:

Fast
Scalable to TB-scale datasets
Resource-efficient
Interpretable

Snap ML models can leverage:

- GPUs for acceleration
- single-node and multi-node environments to learn from big data
- fast interconnects like NVLINK for streaming, out-of-core performance.

Snap ML is available as part of IBM PowerAI (WML)

Project www: https://www.zurich.ibm.com/snapml/

Snap ML characteristics:

It can train logistic regression on a **TB-scale** dataset in 90 seconds.

Trains random forests 4-8x faster than sklearn.

Interpretable decisions for regulated industries

Applicable to **business/scientific** use cases

- ✓ Fraud detection
- ✓ Credit risk assessment
- ✓ Stock volatility prediction
- ✓ Sales forecasting
- ✓ Price optimization
- ✓ … and many more

Core publication: https://arxiv.org/abs/1803.06333

Publications

Publications

1. D. Sarigiannis, T. Parnell, H. Pozidis, “Weighted Sampling for Combined Model Selection and Hyperparameter Tuning,” **AAAI, 2020.**
2. N. Ioannou, C. Mendler-Dünner, T. Parnell, “SySCD: A System-Aware Parallel Coordinate Descent Algorithm,” **NeurIPS, 2019 (Spotlight).**
3. K. Atasu, T. Mittelholzer, “Linear-Complexity Data-Parallel Earth Mover's Distance Approximations,” **ICML, 2019.**
4. C. Dünner, T. Parnell, D. Sarigiannis, N. Ioannou, A. Anghel, M. Kandasamy, G. Ravi and H. Pozidis, “SnapML: A Hierarchical Framework For Machine Learning,” **NeurIPS, 2018.**
5. C. Dünner, M. Gargiani, A. Lucchi, A. Bian, T. Hofmann and M. Jaggi, “A Distributed Second-Order Algorithm You Can Trust,” **ICML, 2018.**
6. T. Parnell, C. Dünner, K. Atasu, M. Sifalakis and H. Pozidis, “Tera-Scale Coordinate Descent on GPUs,” **FGCS, Elsevier, 2018**.
7. C. Dünner, T. Parnell and M. Jaggi, “Efficient Use of Limited-Memory Accelerators for Linear Learning on Heterogeneous Systems,” **NIPS, 2017**.
8. C. Dünner, T. Parnell, K. Atasu, M. Sifalakis and H. Pozidis, “Understanding and Optimizing the Performance of Distributed Machine Learning Applications on Apache Spark,” **IEEE Big Data, 2017**.
9. K. Atasu,. T. Parnell, C. Dünner, M. Vlachos and H. Pozidis, “High-Performance Recommender System Training using Co-Clustering on CPU/GPU Clusters”, **ICPP, 2017**.
10. T. Parnell, C. Dünner, K. Atasu, M. Sifalakis and H. Pozidis, “Large-Scale Stochastic Learning using GPUs,” **IPDPSW ParLearning, 2017**.
11. K. Atasu, T. Parnell, C. Dünner, M. Sifalakis, H. Pozidis, V. Vasileiadis, M. Vlachos, C. Berrospi and A. Labbi. “Linear-Complexity Word Mover’s Distance with GPU Acceleration”, **IEEE BigData, 2017**.
12. C. Dünner, S. Forte, M. Takac and M. Jaggi, “Primal-Dual Rates and Certificates,” **ICML, 2016**.

Workshops and Presentations

- 7 workshop papers at NeurIPS 2018, NeurIPS 2019
- Numerous invited talks and presentations, e.g. IBM THINK 2018, THINK 2019, TechU Prague 2019, SC’19


AI Automation
IBM

AI Automation for scalable AI adoption

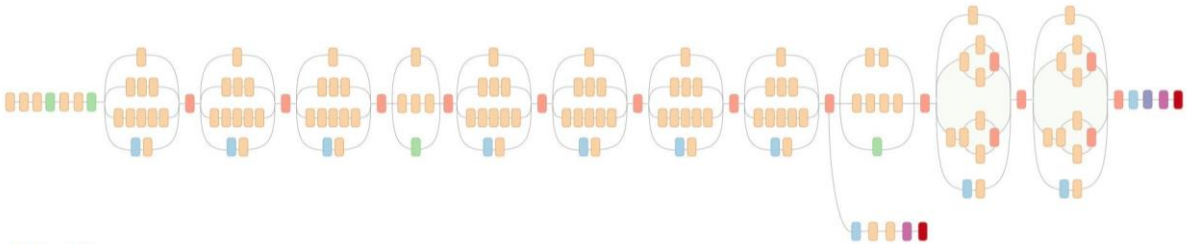
From automation of neural network architecture search, towards the acceleration of the full data science process in one click

Cristiano Malossi
IBM Research – Zurich
acm@zurich.ibm.com

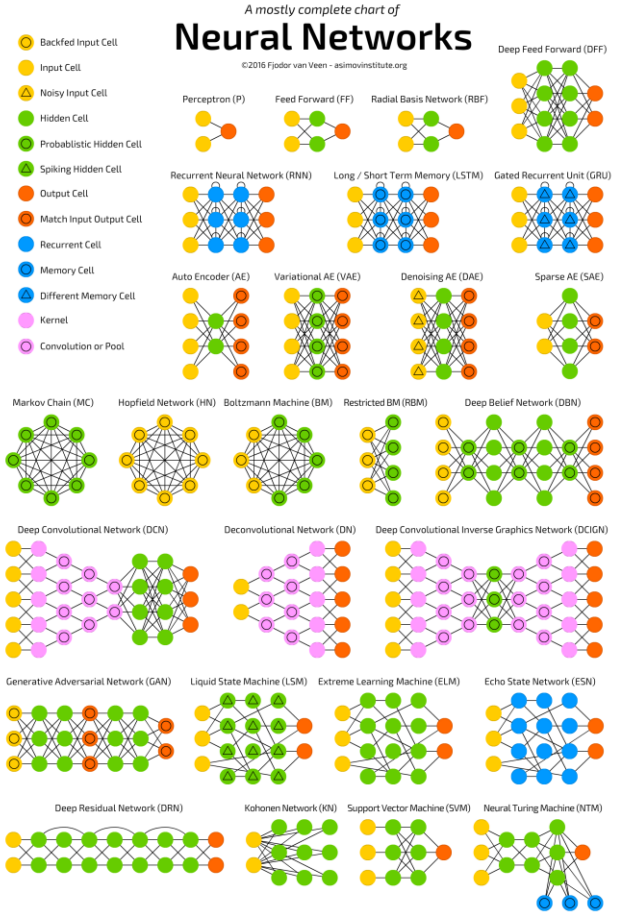
IBM

The need for automation: millions of params to tweak

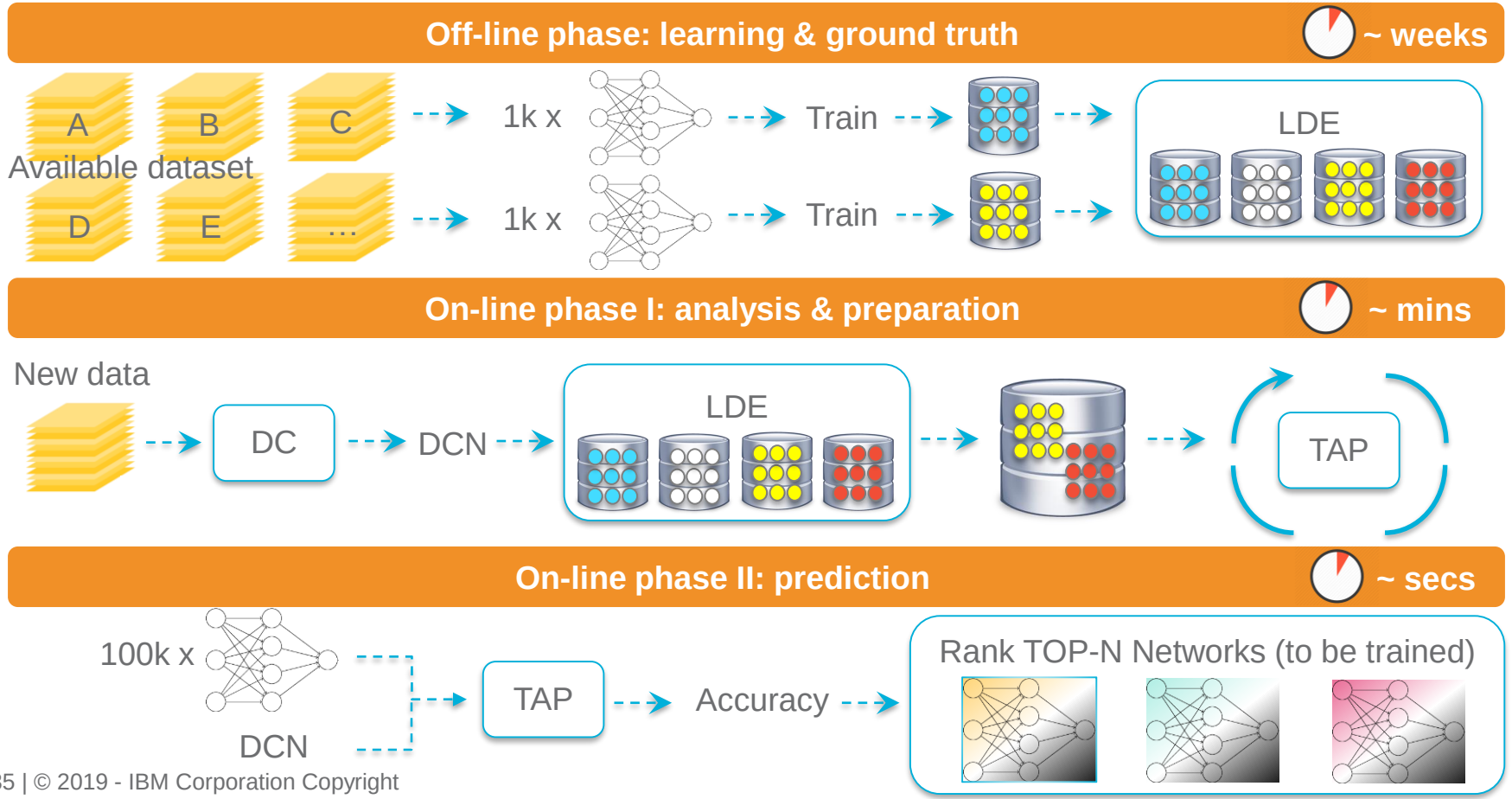
- AI models development is a manual, empirical process that requires highly skilled data-scientists
- Each hand-crafted model takes weeks to months of work
- Models must be adapted and re-trained for new tasks, even when very similar
- Size and complexity of AI models grow every year!



Inception V3 architecture:
C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna. Rethinking the Inception Architecture for Computer Vision, 2015.

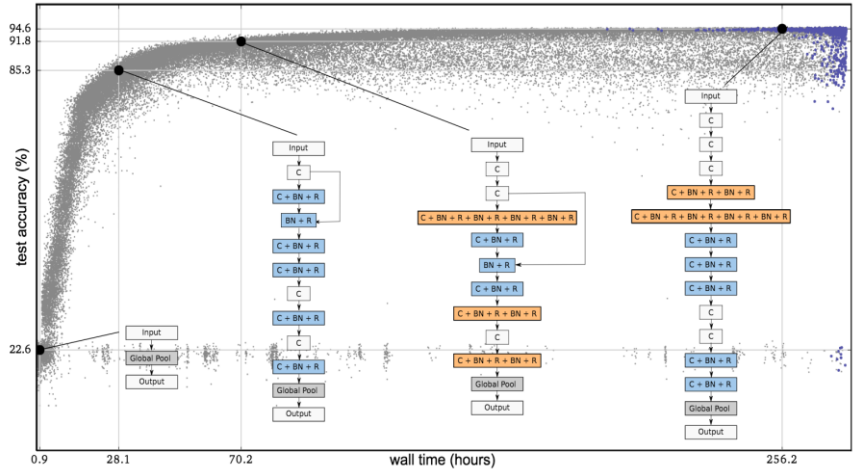


Train-Less Accuracy Predictor for Architecture Search



Classical LSE

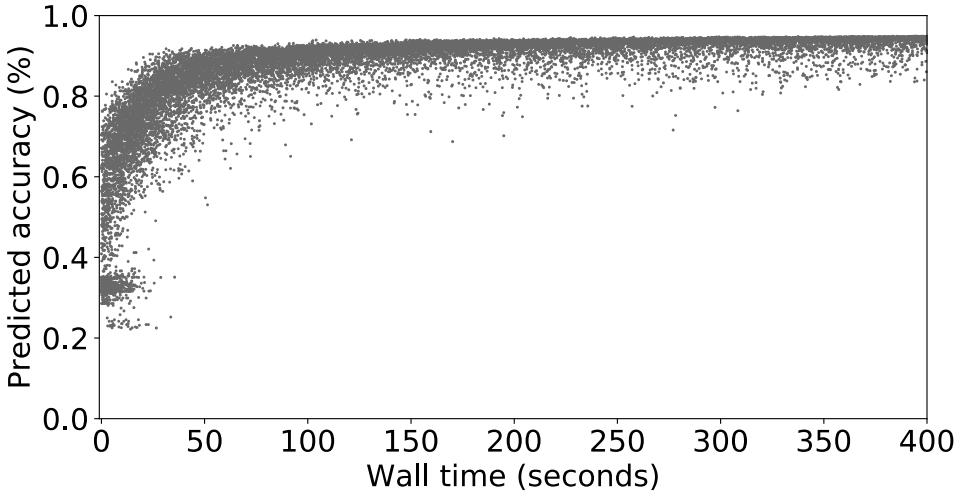
(Real et al. 2017)



- Search time: 256h
- Resources: 250 workers (~250-1000 GPUs)
- CIFAR-10 top accuracy: 94.6%
- CIFAR-100 top accuracy: 77%

TAPAS-NeuNetS

(Istrate et al. 2018)



- Search time: 400s (2300x faster)
- Resources: 1 GPU (250-1000x less expensive)
- CIFAR-10 top accuracy: 93.67% (1% worse)
- CIFAR-100 top accuracy: 81.01% (4% better)

SPEED UP ORDER: 1M (architecture discovery) – 100k (trained model)

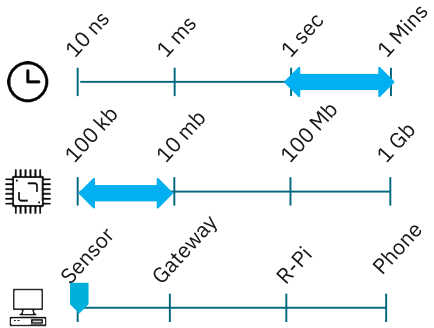
Constrained model synthesis for IoT applications

USER INPUT

Dataset

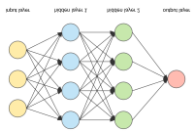
Images, Sensors, Audio

IoT Budget & Requirements

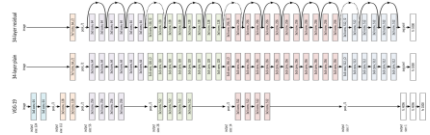


AUTOMATED ML MODEL SYNTHESIS FOR GIVEN EDGE DEVICE

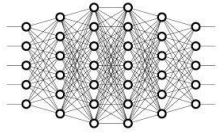
CASE #1



CASE #2



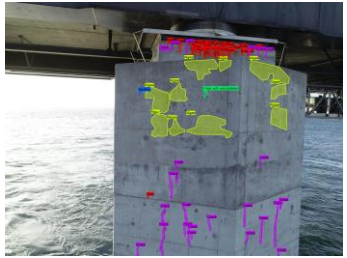
CASE #3



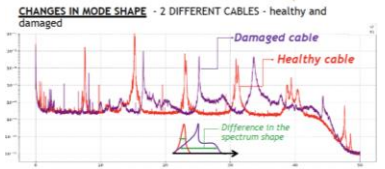
...

OUTPUT

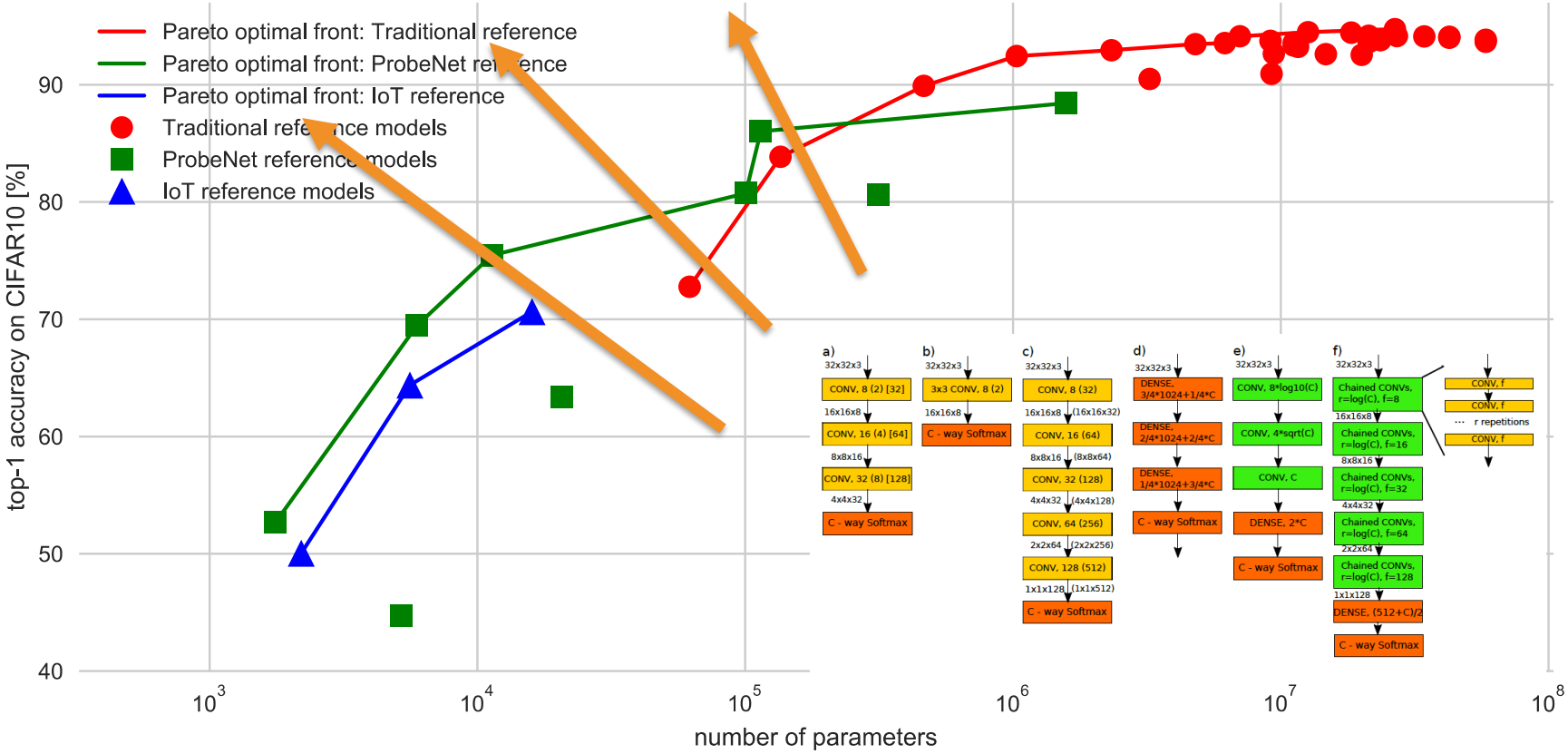
Visual inspection



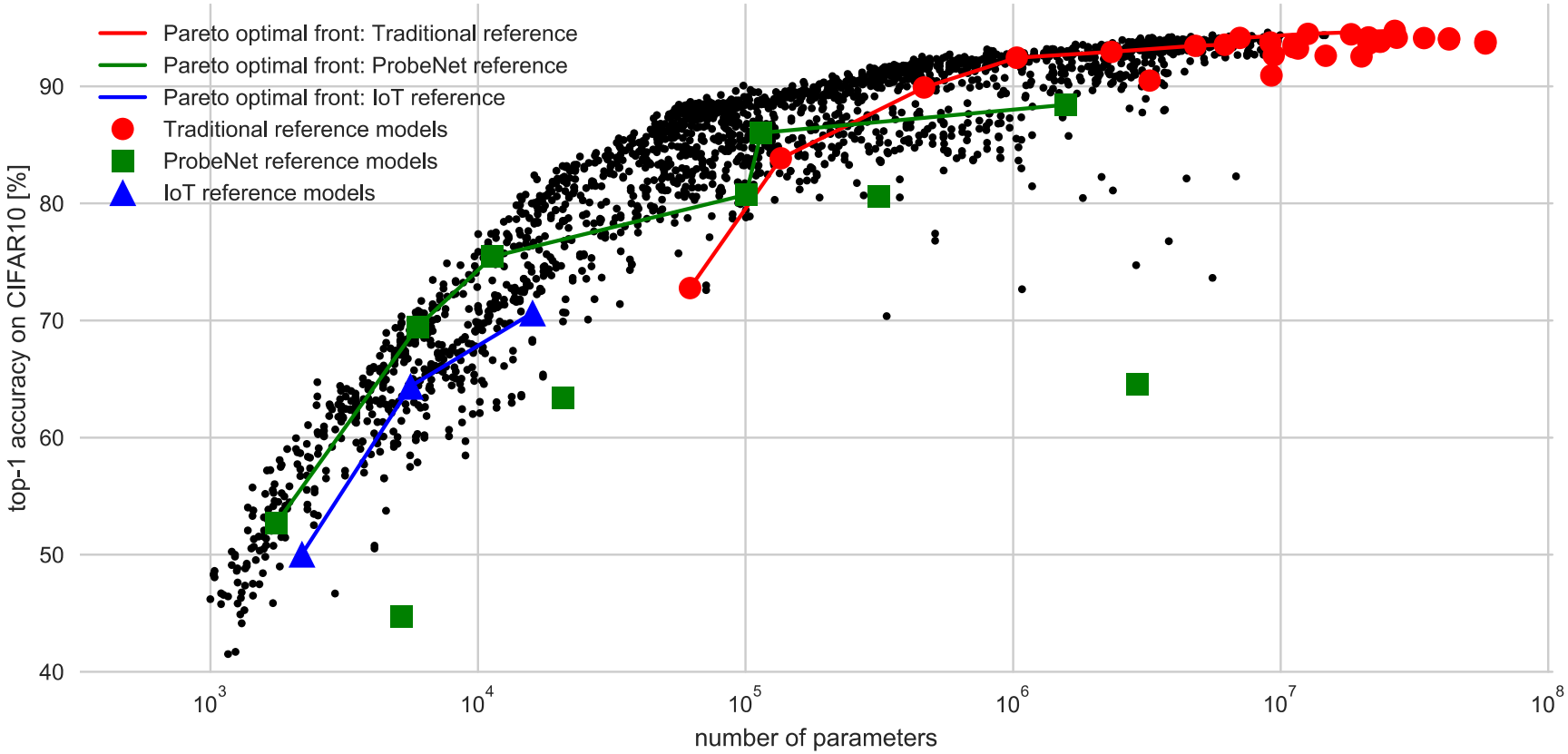
Anomaly detection



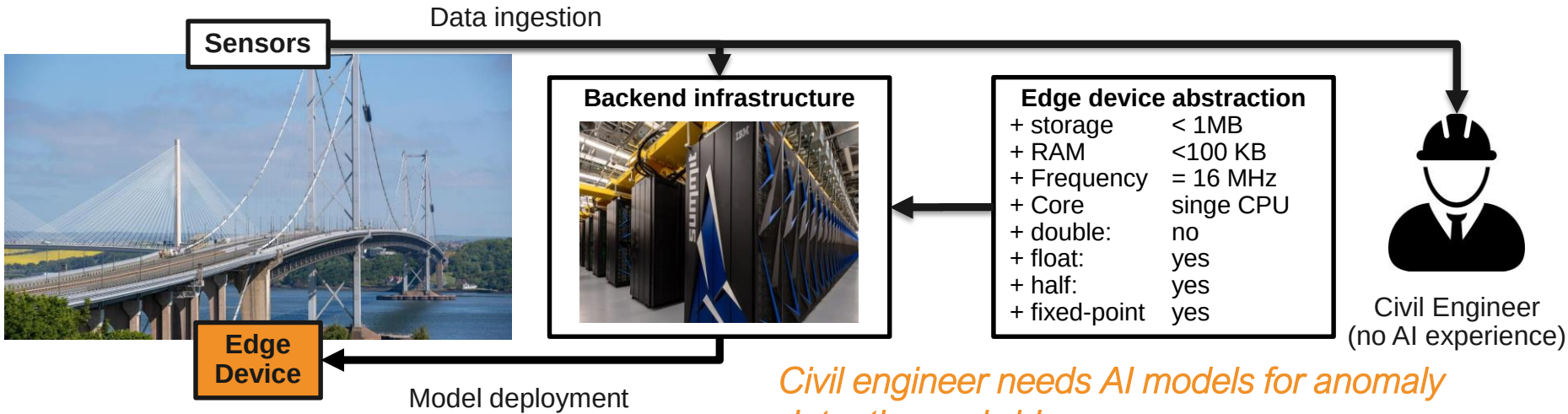
Constrained model synthesis: size/accuracy trade-off

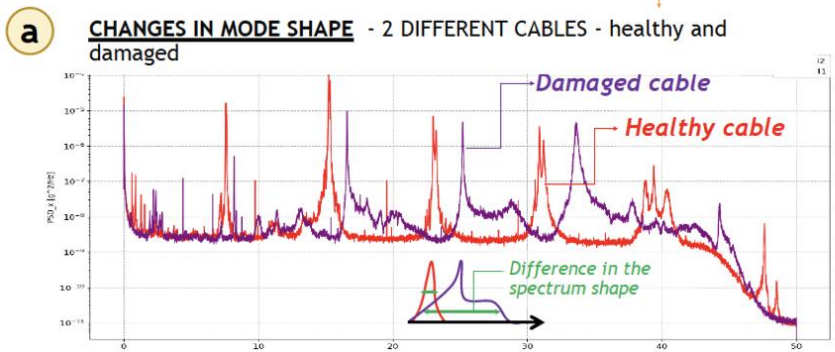


Constrained model synthesis: with automated search



Application to Civil Engineering: anomaly detection





Civil engineer needs AI models for anomaly detection on bridges:

- Civil engineer can annotate data (domain expert)
- Civil Engineer knows Edge device characteristics
- *NeuNetS* creates & deploys models
- Civil engineer can analyze results and provide feedback to the system (augment data and improves over time)

Thank You!

Haris Pozidis
Cloud Storage & Analytics
IBM Research – Zurich