EBOOK

GridGain

Using In-Memory Computing for Continuous Machine and Deep Learning

Part 1: A Machine and Deep Learning Primer

To many software developers, machine learning is like rocket science: too complicated and not living up to its full potential. For those who don't understand the underlying principles, the subject seems too complex to tackle. For those who do try to get started, it's not clear which tools to use. For those who have already implemented some form of machine learning, the concept of continuous learning, which is what we humans do as a matter of course in our daily lives, seems out of reach. The reason is that it takes too long to manage, move, and (re)train models, given the amount of data needed to create a reasonably accurate model.

But continuous machine and deep learning is attainable. This machine learning eBook series is designed to give developers a basic understanding of machine and deep learning, hands-on experience with Apache Ignite to get up and running quickly with continuous learning, and tips to help avoid some of the more common challenges.

This eBook Series

The series is broken into five parts:

- **Part 1:** A Machine and Deep Learning Primer
- **Part 2:** Hands-on Machine Learning Using Apache Ignite
- **Part 3:** Hands-on Machine Learning for Fraud Detection at Scale Using Apache Ignite
- **Part 4:** Hands-on Deep Learning Using the Apache Ignite Genetic Algorithm
- **Part 5:** Hands-on Deep Learning Using TensorFlow with Apache Ignite

The series is designed to be flexible and useful. Start with the primer now for a basic understanding of the concepts. Then continue with the hands-on topics as you need them, in any order.

GridGain

A MACHINE AND DEEP LEARNING PRIMER

The opportunities to use machine learning are everywhere. The enormous quantities of data now available, driven by our online lives and constant digital connectivity, has made the turn to machine learning possible. But it is the recent growth in computing capabilities, along with advances in the machine learning technologies, that have made that turn inevitable.

With machine learning, statistical algorithms are used with massive amounts of data to discover patterns and insights that enable us to better predict probable outcomes in a variety of domains. Machine learning algorithms help programs learn and adapt, adjusting themselves as needed to provide better and more accurate answers. This is the wave of the future, one that will either help augment or replace most applications. The turn to machine learning and deep learning will change the world as we know it.

WHAT IS MACHINE LEARNING?

In traditional programming, software architects model tasks down to the smallest detail, and then programmers implement each task and every possibility in code. While there is some flexibility through configuration, in general each program cannot change what it does without changing code.

With machine learning, programs can change their own behavior after having been trained to find the significant and relevant patterns in a set of data. Machine learning algorithms find "instructions"—that is, the weights to apply to the formulas—based on what they discover in the data they've been fed. Software developers or analysts gather and prepare the data, ensuring that it is tidy and accurate, without any misleading biases that might distort the findings. They determine which algorithm or algorithms are best suited for the task at hand, and run them on a percentage of the dataset designed to "train" the algorithm to create a model. They then usually test the model against different datasets to determine whether what the model has learned in training is accurate enough.

A fully trained model is the goal. A trained model can be applied to new data (of the sort it has been trained to work with), at any time, to do whatever it was trained to do: predict probable values; classify data or images; or cluster objects based on similar characteristics.

Machine learning is driving innovation in every industry, and every domain. The promise of machine learning is to replace basic human decision-making with automated decision-making, and to help guide more complicated decisions at scale, to deliver quick-turnaround answers in real time, at a rate no human could ever manage on their own.

Today we can see machine learning influencing many parts of our daily lives:

- An ecommerce site that provides recommendations based on previous user selections
- Mail programs that detect and flag potential spam
- Increasingly, programs that calculate our credit scores
- Property sites that calculate the optimum price for a given house in a given market
- Applications that help identify diseases in difficult-to-diagnose cases

But existing uses of machine learning have focused on relatively simple problems where the data and the outcomes are relatively constant. For machine learning to make a big impact in our daily lives, it needs to adapt quickly to ever-changing conditions and criteria. It must support continuous learning based on new data, so as to stay nimble and relevant. The challenge is that continuous learning is hard when massive amounts of data must be moved around or streamed, processed, and reflected in the models in real time.

WHAT IS DEEP LEARNING?

Deep learning is a subset of machine learning. Machine learning is divided into two types: shallow and deep. The so-called shallow machine learning is the traditional machine learning pipeline:

- Supervised learning: The data that the algorithms work on is labeled with the correct outcomes
- Unsupervised learning: The data is entirely unlabeled, and the algorithms find the patterns
- Reinforcement learning: Uses a reward-based feedback loop

In deep learning, there are many more algorithms at work, and the work they perform proceeds in a kind of black box, with little explanation for how the algorithms arrive at the answers they do. These are the "hidden layers" of the deep neural networks. Deep learning architecture was an attempt to create a structure that imitates the human brain, although the structure and processing are statistical and mathematical in nature.

THE BIG CHALLENGES WITH MACHINE LEARNING

It ought not to come as a surprise that the biggest challenge with machine learning is not which algorithms to choose, but the training of those algorithms. In other words, it's the data, or more precisely, the amount of data and the current state it represents. To become accurate and reliable, the predictive models of machine learning must crunch their way through massive amounts of data—sometimes terabytes or more. In addition, the data must accurately reflect the current state of whatever is happening. The algorithm creates an approximate model based on the behaviors and conditions of that time, both of which can and often do change. So models need to be continually tested for accuracy, and retrained whenever the estimates are not good enough. Some call this approach continuous learning. This is what we do as people, so it shouldn't be surprising that as we try to automate decisions currently made by people, that it's what we want out of machine learning.

Unfortunately, it's hard to implement continuous learning. One reason is that it takes too long to train and retrain models on traditional infrastructure. The first big challenge is that data must be "ETL"ed – Extracted, Transformed, and Loaded. Most machine learning is conducted on specialized infrastructure, not directly where the data is stored. So first, the data must be moved across a network, which can take hours. A typical corporate 10 GigE network can only move roughly a gigabyte a second fully loaded, or 3.6 terabytes an hour, without any network collisions, and then only if it were dedicated to just one task. The data may also need to be transformed, cleansed, and enriched before being loaded, which can also take time.

Second, most machine learning infrastructure, while designed to run machine learning algorithms, does not scale well in practice when dealing with terabytes or petabytes of data. It can take hours to train or retrain models even once the data is loaded. While a delayed time frame may be sufficient for some applications where behavior doesn't change that often, like a manufacturing line, it does not work well when behavior changes more frequently, such as anything involving people, or even the weather. With any moving targets like these, it's important to be continuously retesting and retraining the model. And in order to continuously retrain, that means the latest data must be processed and models retrained in real time, similar to the way streaming analytics works. Ideally, testing and retraining happens fast enough to impact each interaction and decision as the changes are detected, before the first bad decision happens. This is exactly how people avoid bad decisions. They operate based on the latest information and adapt in real time.

If you understand these points, then you probably will arrive at the same conclusion others have: that there is one clear way to implement continuous machine learning.

1. To process this much data in real time, machine learning must be horizontally scalable and run against data already in RAM. This is possible because many machine learning algorithms and data are parallelizable.
2. Since the data cannot be moved fast enough to do the training, the machine learning algorithms must run in place, where the data is stored.
3. In order to react fast enough during operations, the infrastructure must be collocated with any transaction or stream processing that is using the machine learning results.

At first glance, this may not seem possible without rewriting both the applications and the machine learning infrastructure. Luckily, companies have already been adding technologies to existing systems and newer applications, APIs, and real-time analytics that satisfy all three criteria: in-memory computing.

THE SOLUTION: IN-MEMORY COMPUTING

In-memory computing has been used for the last decade to add speed and scale to existing applications, to ingest massive datasets in real time, and to perform real-time analytics and high-performance computing. These problems have very similar characteristics to machine and deep learning.

In-memory computing adds speed by moving data from hard drives (HDDs) into memory (RAM). While HDD media speeds are measured in milliseconds, RAM speeds can be measured in nanoseconds, a million times faster. In-memory computing adds linear, horizontal scalability by partitioning data across a cluster of machines, or nodes. To avoid moving large datasets over the network, in-memory computing also moves the code—whether it's SQL, Java, .NET or C++—to each node and then aggregates the results, similar to the way Hadoop works.

Once applications and streaming analytics are running on in-memory computing, adding machine and deep learning becomes straightforward. You implement algorithms as code that can be distributed across an in-memory computing cluster. The cluster can then be configured and further scaled to run model testing, training, and retraining continuously in

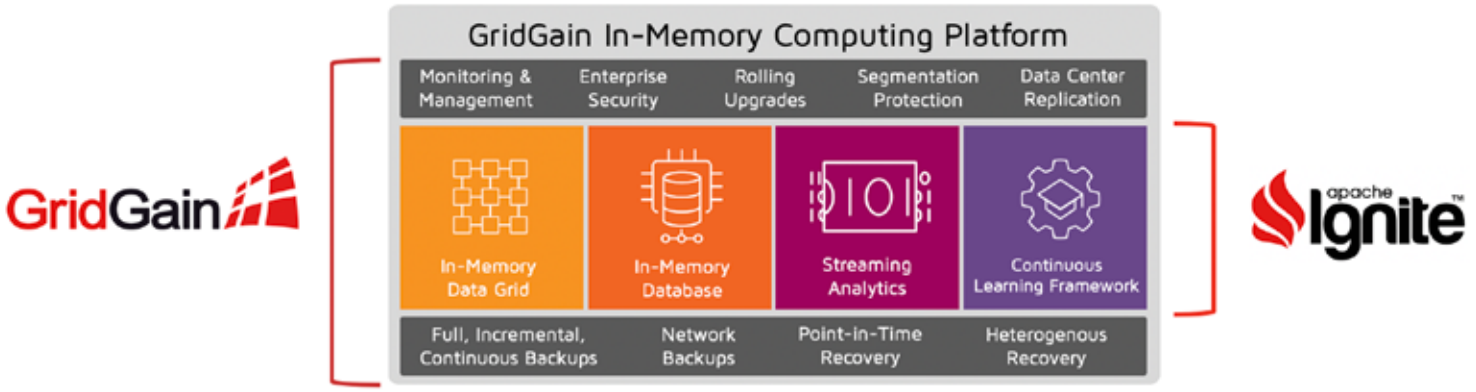

Figure 1. Apache® Ignite™ and GridGain® In-Memory Computing Platform

place, in memory, and in parallel with decision automation. By adding in-memory speed and unlimited horizontal scale, in-memory computing enables models to adapt to new conditions before a bad decision is made.

GRIDGAIN® AND APACHE® IGNITE™: IN-MEMORY COMPUTING AND ENHANCED SUPPORT FOR CONTINUOUS LEARNING

Apache Ignite (see Figure 1) is one of the top five Apache Software Foundation (ASF) projects and the leading ASF project for in-memory computing. Ignite is an in-memory computing platform that includes an in-memory data grid (IMDG), in-memory database (IMDB), support for streaming analytics, and a continuous learning framework for machine and deep learning. It provides in-memory speed and unlimited horizontal scalability to:

- New or existing online transaction processing (OLTP) or online analytical processing (OLAP) applications
- New or existing hybrid transactional/analytical processing (HTAP) applications
- Streaming analytics
- Continuous learning use cases involving machine or deep learning

Continuous learning was added last, in part because those applications that already required speed and scale also needed speed and scale for any decision automation using machine or deep learning, which meant running continuous learning in place.

GridGain (see Figure 1) provides the only enterprise-grade, commercially supported version of the Apache Ignite open source project, and it is the only company that provides commercial support for Apache Ignite open source-based deployments as well. GridGain Systems contributed the code that became Ignite to the Apache Software Foundation and continues to be the project's lead contributor. GridGain is 100% compatible with Ignite. GridGain focuses on adding improved connectivity and integration, security, deployment, management and monitoring capabilities, and on hardening and patching Ignite for business-critical deployments.

When GridGain originally contributed the code to the Apache Ignite project, Ignite was primarily used as an in-memory data grid (IMDG). At the heart of the Ignite architecture is memory-centric storage spread across a cluster of nodes, and a compute grid that distributes code to each node, runs it in parallel across the nodes, and then aggregates the results, similar to the way MapReduce works. This is referred to as massively parallel processing, or MPP. Over time, as companies expanded their use of in-memory computing, the Apache Ignite project added distributed SQL, an in-memory database (IMDB), streaming analytics, and machine learning on top of Ignite's core architecture. Streaming analytics includes the broadest integration with Apache Spark when compared to other in-memory computing technologies. Ignite became a natural fit to provide in-memory data management for Apache Spark where the Hadoop Distributed File System (HDFS) falls short as disk-based storage, as well as a platform to "push down" certain computing to the compute grid and run at scale. As Apache Spark processing capabilities evolved to support machine and deep learning, so too did Apache Ignite. Apache Ignite machine and deep learning is a set of simple, scalable, and efficient algorithms implemented on top of the Ignite compute grid that allow the building of predictive machine learning models in place, without costly ETL data transfers over the network.

The GridGain editions include all of the capabilities of Apache

Ignite, including machine learning and deep learning. So in this eBook, when we refer to Ignite, we mean both Ignite and GridGain.

CONTINUOUS MACHINE LEARNING

The machine learning capabilities of Apache Ignite enable the building of predictive models directly within Ignite. This allows users to achieve scale and performance without costly ETL or data transfer (see Figure 2).

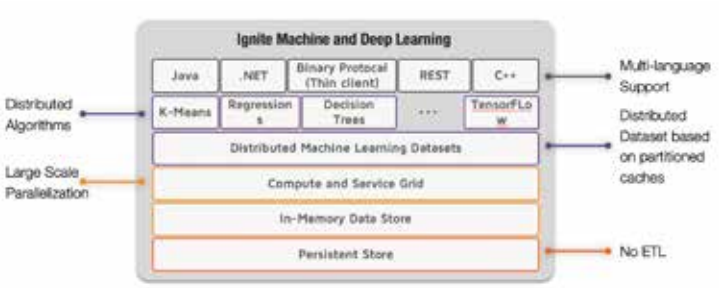

Figure 2. Ignite Machine and Deep Learning

Previously, machine learning models had to be trained and deployed on different systems. For example, data would need to be moved to specialized infrastructure, where training would be performed using one set of tools. Then models would be deployed into production systems using a second set of tools. This previous approach had several drawbacks:

- There was a lengthy ETL process, particularly for very large datasets, that could take hours to move, cleanse, and enrich. Datasets can be hundreds of gigabytes or terabytes in size.
- Datasets often pushed the capacity of a single server on systems that only supported scaling up. It made scaling more expensive and led to longer training times.
- ETL meant the developer was taking a snapshot of the data, not using the live data, which meant the models were effectively out of date.
- It required retraining the models each time they changed, which involved long wait times.

Ignite solves these problems by moving the machine and deep learning directly to the decision-making systems and the data:

- Ignite can work on the data in place, avoiding costly ETL.
- Ignite can scale to petabytes of data in any combination of RAM, NVRAM, and disk across thousands of nodes for the best combination of speed and scale.
- Ignite can ingest and process streaming data in real time to update models, and use model updates midstream during transactions also running on Ignite.
- Data models in Ignite can be used for runtime decision automation within the same cluster as the operational systems. You don't need a second set of tools.

ZERO NETWORK-BASED ETL, IN-MEMORY SPEED, AND HORIZONTAL SCALABILITY

Apache Ignite machine and deep learning is built on Ignite's memory-centric storage, which eliminates the need for network-based data movement as part of the ETL process for machine and deep learning by bringing the learning to the data and running it with in-memory speed and massive horizontal scalability. Ignite provides a host of data preprocessing, machine and deep learning algorithms that are optimized to leverage Ignite's MPP capabilities. These implementations are optimized to run in place at once across large clusters and massive datasets, or incrementally against incoming data streams. You can also implement your own algorithms on top of the Ignite compute grid in Java, .NET, or C++, or even leverage SQL. This support for both full and incremental training enables continuous learning that can improve decisions based on the latest data as it arrives in real time.

The built-in algorithms and use of the compute grid makes it easy to add machine and deep learning to existing Ignite deployments that already support transactional systems. Ignite can also easily support new data being loaded or streamed as needed. Because Ignite scales horizontally, additional workloads including machine and deep learning can be run against the same data without impacting the performance of the existing applications. Additionally, APIs can be hosted on Ignite that use any models to help automate decisions. These APIs can be collocated with any related computing to deliver the fastest results.

FAULT TOLERANCE AND CONTINUOUS LEARNING

Once you move to continuous learning, you need to make sure your systems do not go down. It's not just that your models become out of date. When you move to delivering real-time support to a customer, you need to be able to handle exceptions in real time, which means stream processing cannot go down.

Apache Ignite machine learning has a host of features that help maximize availability. For example, it can keep multiple copies of the same data, and reroute requests automatically in the case of a node failure. It supports having clusters span multiple data centers. GridGain makes multi-data-

center management and disaster recovery easy to manage. All recovery procedures are transparent to the applications. Existing processes, including machine and deep learning along with any use of the models in production, continue to run without interruption. All of these features are automatically leveraged by machine and deep learning.

PARTITION-BASED DATASETS

Horizontal scalability is easy to say and hard to do. The data must be partitioned across the cluster in such a way that when the code executes on each node, it has all the data it needs without having to fetch more over the network. Once data has to move over the network, it can bring machine and deep learning performance to a grinding halt.

Ignite solves this problem by supporting various predefined and user-defined partitioning algorithms. Partitioning provides an abstraction layer that sits between a machine learning algorithm and the storage and compute be applied to the Key of a Key-Value (K-V) pair to determine where the Value is stored in a cluster. The Value is stored in a partition. Partitions are atomic. It helps ensure the data is sent to the right node in advance to help minimize network traffic during execution. Ignite preprocesses data on each node, creates a new cache for training data, and then executes the model training. Once the algorithms have executed, the results are then aggregated using MapReduce. Partitioning can create a master and multiple copies across the cluster for maximum performance and availability in the event of individual node failures.

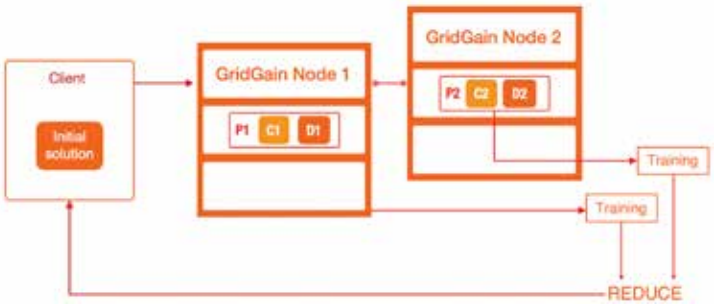

Figure 3. Partition-based Dataset and Context

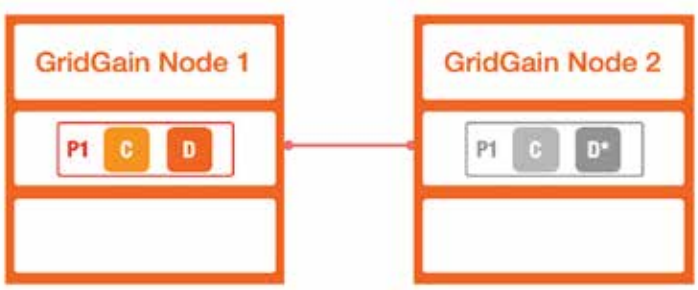

Figure 4. Node Grid

For example, if you have a two-node cluster there could be two corresponding partitions P1 with data D1 and P2 with data D2 (see Figure 3). The machine learning algorithms will have a separate cache to manage the context of the machine and deep learning with its own partitions on each node, P3 with C1, and P4 with C2. The model training results from each node are then combined and become the initial model for a client doing the model execution.

In the event of a node failure, Ignite can recover the partition and context (see Figure 4). For example, node 2 could have a backup of data and context for node 1 and become active whenever node 2 fails. Context is automatically backed up. Training data is usually recovered from the cluster by re-running preprocessing routines and ETL (marked D*).

	CLASSIFICATION	REGRESSION	CLUSTERING
Description	Identify to which category a new observation belongs, on the basis of a training set of data	Modeling the relationship between a scalar dependent variable y and one or more explanatory variables x	Grouping a set of objects in such a way that objects in the same group are more similar to each other than to those in other groups
Applicability	spam detection, image recognition, credit scoring, disease identification	Drug response, stock prices, supermarket revenue	Customer segmentation, grouping experiment outcomes, grouping shopping items
Algorithms	SVM, nearest neighbor, decision tree classification, neural network	Linear regression, decision tree regression, nearest neighbor, neural network	K-means

Table 1. Machine learning algorithms

TYPES OF MACHINE AND DEEP LEARNING ALGORITHMS

Now that you have a basic understanding of machine and deep learning, you need to start to understand the different algorithms. Ignite supports a host of different types of machine learning algorithms (see Table 1): classification, to identify a type of object or data point using supervised learning; regression, to calculate a result from various inputs; and clustering, to try to identify groups using unsupervised learning. For other algorithms, you can either turn to the Ignite community for examples or implement your own. Ignite also supports preprocessing the data to prepare data for these various types of learning.

Regarding deep learning, there are some prebuilt algorithms, including a multilayer perceptron and a genetic algorithm. But most people are interested in TensorFlow. Ignite 2.7 added support for TensorFlow, so people could use it directly against data in Ignite for speed and scale.

WHERE TO GO NEXT

The next two eBooks, Parts 2 and 3, give hands-on examples on how to use these different types of machine learning. Part 4 covers how to use the genetic algorithm. Part 5 focuses on using TensorFlow.

You can always go online and get started now on the machine and deep learning resource center.

Contact GridGain Systems

To learn more about how GridGain can help your business, please email our sales team at sales@gridgain.com, call us at +1 (650) 241-2281 (US) or +44 (0)208 610 0666 (Europe), or complete the form at https://www.gridgain.com/contact to have us contact you.

About GridGain Systems

GridGain Systems is revolutionizing real-time data access and processing with the GridGain in-memory computing platform built on Apache® Ignite™. GridGain and Apache Ignite are used by tens of thousands of global enterprises in financial services, fintech, software, e-commerce, retail, online business services, healthcare, telecom and other major sectors, with a client list that includes ING, Raymond James, American Express, Societe Generale, Finastra, IHS Markit, ServiceNow, Marketo, RingCentral, American Airlines, Agilent, and UnitedHealthcare. GridGain delivers unprecedented speed and massive scalability to both legacy and greenfield applications. Deployed on a distributed cluster of commodity servers, GridGain software can reside between the application and data layers (RDBMS, NoSQL and Apache® Hadoop®), requiring no rip-and-replace of the existing databases, or it can be deployed as an in-memory transactional SQL database. GridGain is the most comprehensive in-memory computing platform for high-volume ACID transactions, real-time analytics, web-scale applications, continuous learning and hybrid transactional/analytical processing (HTAP). For more information on GridGain products and services, visit www.gridgain.com.