Executive UPDATE
CUTTER CONSORTIUM
Access to the Experts
DATA ANALYTICS & DIGITAL TECHNOLOGIES

IoT Time Series Data: The Smarter Path to Solutions

by Sean Lorenz

The Internet of Things (IoT) — like gamification or big data — is now a commonplace term in the technology industry. And just like big data, the IoT can mean something different to just about everyone. The one thing we all seem to agree on is that the IoT is bound to be big, so now's the time to create an IoT strategy that gives your company a competitive edge.

The problem is, however, that companies are creating connected products just for the sake of putting them on display at *CES* (*Consumer Electronics Show*). Consumers are responding to this approach with ambivalence to many of these new IoT-enabled products. On the B2B side of the coin, enterprises are amassing data from sensors to maximize operational efficiencies yet the output of these efforts often end up unused in simple business intelligence reporting tools. So what is the solution? The promise of the IoT centers on *what you do with the data* — not connecting and collecting it.

In this *Executive Update,* we explore methods for creating actionable intelligence from time series–based sensor data in order to solve specific business problems.

Ingest and Organize

Application programming interfaces (APIs) are the lifeblood of the IoT. APIs can often be RESTful due to software engineering's familiarity with this format from Web application development. However, publish-subscribe protocols such as MQTT or CoAP are becoming de facto methods for shuttling data to and from edge devices, local hubs, cloud infrastructure, Web apps, and mobile apps. The vast majority of IoT APIs pack information into JSON-formatted data objects before sending data back and forth.

Most IoT projects today use Amazon Web Services (AWS), Microsoft Azure, or Google Cloud Engine to manage real-time ingestion and organization of IoT data. One of the most crucial design decisions you can make comes at this stage. How do you plan on storing/accessing all that data efficiently for analysis and decision making?

Certainly, we don't want all this data becoming stagnant in some giant data lake. Our goal is to collect, organize, store, and retrieve aggregated data as efficiently as possible since customers and employees often use applications that require real-time responses.

Since the IoT deals with detecting changes happening out in the real world, much of the data we analyze is inherently made of time series. One of the most well-used methods for dealing with time series data in Python is open source pandas (the Python Data Analysis Library), which does an excellent job at handling data objects with timestamps attached to them so that you can easily analyze data at different time intervals. Apache Cassandra is another open source distributed database management system built more for those familiar with SQL who want to add time series support to their application.

In addition, Apache has introduced a newcomer to the mix: Spark, which is quickly gaining momentum. Spark has its own data frame format along with useful migration tools for Hadoop, pandas, and Cassandra users who want to make the switch.

The Importance of Time

So what makes time series data different than text, images, or structured records? The key differentiator (for our purposes) is simply finding the appropriate predictive algorithms to use on temporal datasets. Unlike images, keyword searches, or database field queries that can be analyzed independent of time, real-world IoT data can rarely be isolated when building actionable intelligence into an application.

As a computational neuroscientist by training, I tend to look at the IoT as a giant brain. Our brains take in massive amounts of motor and sensory data from "edge devices" in a process similar to that of nonbiological sensors. The commonality between the two is their reliance on time series inputs to produce relevant real-time outputs. Table 1 illustrates a biological (human brain) example of the process involved in picking up a coffee cup versus a nonbiological (medical IoT) example outlining the process of predicting sudden oxygen desats in neonatal intensive care unit (NICU) babies.

This entire process of predicting either the best path for picking up a cup or whether a NICU baby will stop breathing is completely dependent on the precise timing across a number of sensors. As our world becomes more and more connected, the need for precisely timed predictive sensor fusion algorithms is essential for business success. In our NICU example, the ability to forecast apnea and desats can not only save the lives of babies, but also reduce the time a baby stays in a NICU. Sudden desats take a toll on a baby's lungs and often mean longer hospital stays, which equates to higher hospital bills. Healthcare systems transitioning to value-based care are eager to implement systems like this in order to reduce costs.

Table 1 — Biological Example of the Process Involved in Picking Up a Coffee Cup vs. Nonbiological Example of Predicting Sudden Oxygen Desats in NICU Babies

Sensor Processing Steps	Biological Example	Nonbiological Example
1. Ingest sensor data	These include sight, touch, smell, temperature, and force.	These include O2 levels, pulse oximetry, bed mat restlessness, feeding tube delivery, and medications.
2. Preprocess and organize at the edge	Sight data gets routed from eyes and nose through the thalamus to the primary visual cortex; haptic touch and temp inputs travel from fingers up the spinal cord in the peripheral nervous system.	Data from each medical device is transmitted (e.g., BLE, Wi-Fi, ZigBee) to a local gateway hub where it homogenizes data from different formats into a single data structure while performing downsampling or other data munging to optimize prediction and performance.
3. Send data to central location	Visual line and edge data is sent to higher-level vision areas to let you know that what you're looking at is indeed a "cup" while the steam from the cup prompts a prior understanding of "hot."	Data at the local gateway is now sent to the cloud (or local server with greater number-crunching capability) where time series analysis can be performed based on existing predictive models.
4. Combine data from multiple centrally aggregated locations	"Cup" inputs along with prior motor control paths (e.g., hand position, distance from cup, heat detection) for "pick up hot beverage" are sent to the posterior parietal cortex for combing visual, motor, and other sensory input plans.	Individually processed O2, pulse, medication timing and baby restlessness predictions and/or higher-level data abstractions (e.g., frequency bins, medication clusters) are combined.
5. Send fused data to decision-making locations	Prefrontal cortex and premotor cortex make execution path decisions for optimal motor planning and cognitive control, resulting in the best way to pick up a hot coffee without spilling or burning yourself.	After sensor fusion, the highest level in the predictive model decides whether an alert that a baby is showing signs of ceasing to breathe in the very near future should be sent to nurses.

Reliving the 1980s

The problem of managing time series data to perform an intelligent action is nothing new. Backpropagation and adaptive resonance theory (ART) artificial neural networks have been used in industrial manufacturing settings to predict part failure on the factory floor for over 30 years. What we now call "Industrial IoT" or "Industry 4.0" was once called machine-to-machine (M2M) back in the days of George Michael and Alf. And just like the return of 1980s fashion and music, the predictive algorithms used for M2M applications — namely backpropagation — are back *en vogue*.

So why the sudden reemergence of backpropagation? Thanks to massively scalable cloud computing and access to far larger datasets, backprop gets better simply by way of brute force. In other words, the Internet and mobile devices have created unprecedented amounts of log, text, image, location, user behavior, and video data. In the 1980s, it took days to create a model from training a backprop algorithm on a few hundred dog images and produced relatively low classification accuracy. Today, we have access to millions of dog photos and can not only distinguish cats from dogs, but also predict a dog's breed.

The same scenario is true with regards to sensor data. Artificial neural networks hit an adoption wall and went out of vogue in the 1990s when only a small set of manufacturing floor use cases showed success. It wasn't until recently that Geoffrey Hinton's work on deep convolutional neural networks (good for image classification) and Chris Manning's work on natural language processing and deep learning caught the eye of major Silicon Valley tech titans. These machine learning (ML) methods are now ubiquitous for Web and mobile applications, but that was just the beginning. Videos are just images strung together in a sequence, and text is more than just keywords. These items need context before and after an event in order to take prediction further. In this way, text and images also now have a time series problem.

Time and Recurrence

Like backprop, another ML method from the 1980s has popped up to tackle time: recurrent neural networks (RNNs). RNNs are similar to backpropagation neural networks with the addition of sequence processing. In other words, they can account for what happened in the past in order to predict something about the future. This also means that instead of updating neurons in a feedforward manner, like most neural networks, RNNs have a notion of recurrence whereby complex temporal and spatial neural network layer patterns "unroll" over time to learn a sequence better. MIT student Nikhil Buduma presents probably the best breakdown of how RNNs have an advantage over traditional backprop networks. RNNs match nicely with the needs of IoT actionable intelligence for a few reasons:

- RNNs use feedback connections to change the neural network's "synaptic weights," making them especially suited to process time series data.
- RNNs retain a "memory" of previous events and utilize this memory when making decisions.
- RNNs learn more complex temporal patterns as a result.

Diving into the rabbit hole just a bit deeper, there is a variant of RNNs that has proven to be especially effective for IoT sensor anomaly detection and actionable intelligence tasks. These are called long short-term memory RNNs (or "LSTMs" for short). The long and short of it (wink, wink) is that LSTMs add a level of gating complexity inside the hidden layer nodes in order to more stably store inputs and compare them to past inputs before allowing them to pass to the next time step. LSTMs can become quite complicated to model, but thankfully there are several open source data science packages that handle much of the computational and mathematical complexity for you.

Stitching It All Together

Now that we've discussed IoT data and methods for building smarter, more adaptive intelligence applications, let's put the pieces all together. At my company, Senter, we combine data from off-the-shelf consumer health IoT products (e.g., scales, wearables, bed mats, motion sensors, blood pressure cuffs) to predict if extremely sick individuals may end up back in the hospital. For example, patients with chronic heart failure (CHF) often gain weight quickly at home due to the common symptom of retaining fluids. CHF patients also become very weak and urinate less when symptoms exacerbate.

Senter aggregates data from various products via their RESTful APIs, then stores that data in the cloud after it has been organized into a pandas data store. Every night, we update an RNN LSTM network model that predicts, for example, whether or not a CHF patient's symptoms are worsening to the point of imminent hospitalization. Once the model is trained, new data enters the cloud via a real-time stream processing application and then runs through the trained LSTM classifier to determine if that patient should see a doctor immediately. As we collect more data and see what historical time series data led to hospitalization, we can retrain our model to hopefully prevent ER visits in the future.

From here, it is easy to extrapolate and see how we can apply these types of predictive intelligence algorithms to just about any industry wanting to solve a problem that couldn't have been solved before access to sensor data. This means smarter planning of when to turn on your sprinklers to save on the water bill. It means smarter energy efficiency management in commercial building by automating blinds, lights, and HVAC systems at every minute. It also means smarter supply chain optimization by tracking pallet whereabouts and contents as well as warehouse floor production to reduce operational costs. Once organizations realize both the limitations and possibilities of predictive analytic tools such as RNNs, they will be able to design better systems that solve business problems for both customers and employees.

About the Author

Sean Lorenz is the founder and CEO of Senter, a startup creating a smart home health hub for healthy aging, as well as CTO for the Aging Well Institute. Dr. Lorenz was the Director of IoT Market Strategy for LogMeIn's IoT platform and a founding member of the TechStars robotics startup Neurala. He has shaped business models and product strategies in several emerging markets, including IoT, robotics, AI, and healthcare. Dr. Lorenz holds a PhD in cognitive and neural systems from Boston University and has extensive knowledge in digital health, natural language processing, brain-computer interfaces, adaptive systems, neuroscience-based computational algorithms, and context-aware computing. He can be reached at sean.d.lorenz@gmail.com.