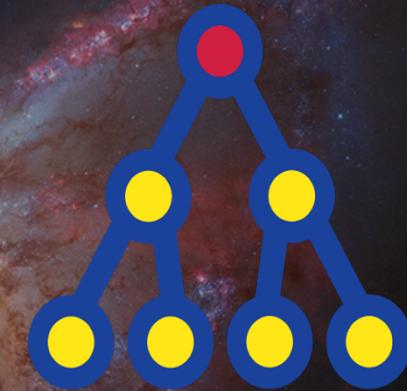


# Numen ta Platform for Intelli gent Computing

(NuPIC)

[numenta.org](http://numenta.org)



Matt Taylor  
OS Community Flag-Bearer  
[matt@numenta.org](mailto:matt@numenta.org)  
[twitter.com/rhyolight](https://twitter.com/rhyolight)

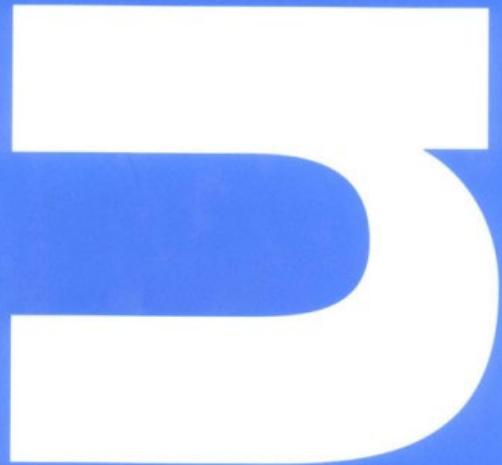
Scott Purdy  
NuPIC Engineer  
[scott@numenta.org](mailto:scott@numenta.org)  
[twitter.com/scottmpurdy](https://twitter.com/scottmpurdy)

## Agenda

- THEORY
- SOFTWARE
- COMMUNITY

Copyrighted Material

HOW A NEW  
UNDERSTANDING OF  
THE BRAIN WILL LEAD  
TO THE CREATION OF  
TRULY INTELLIGENT  
MACHINES



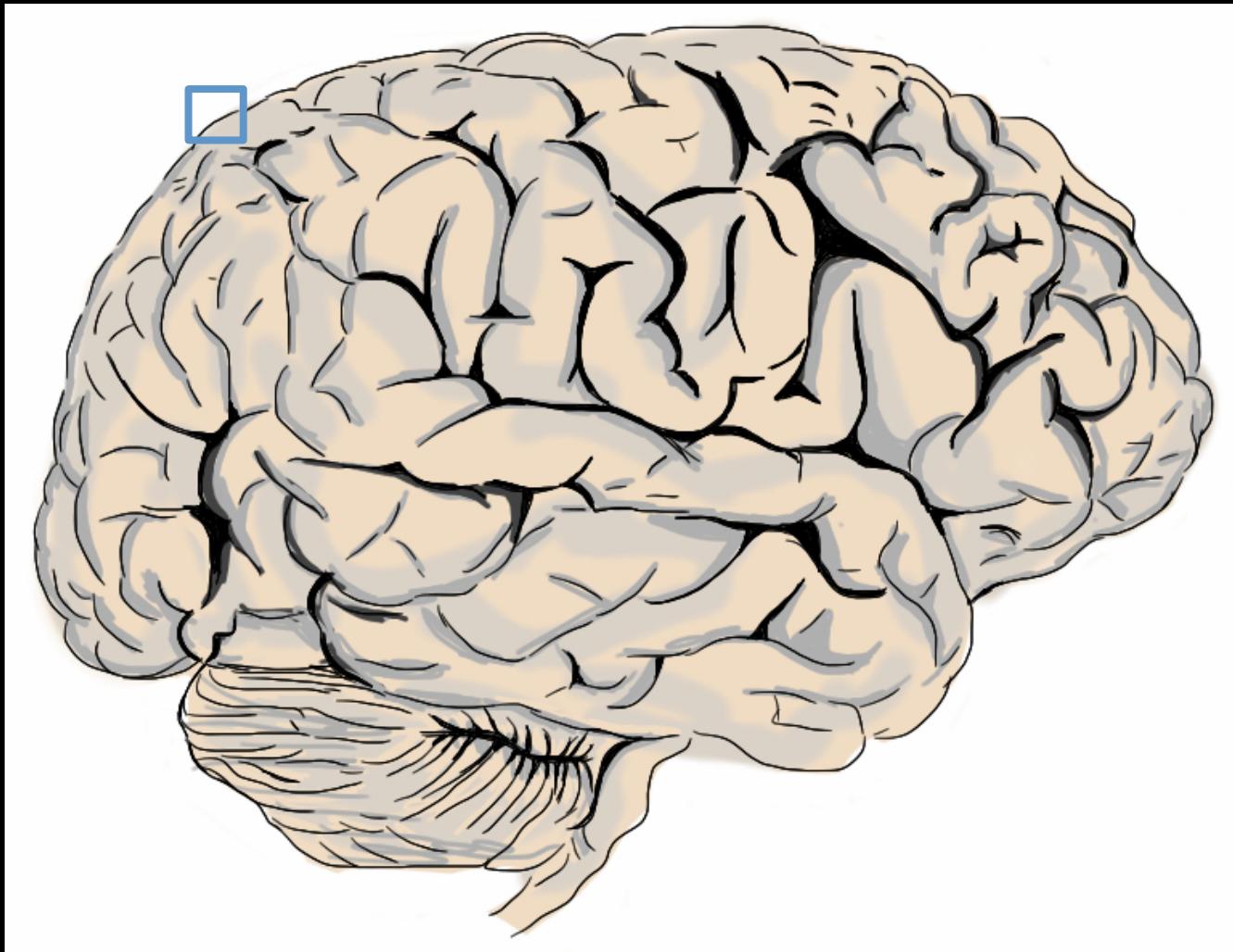
FROM THE  
INVENTOR  
OF THE  
ORIGINAL  
PALMPilot

# INTELLIGENCE

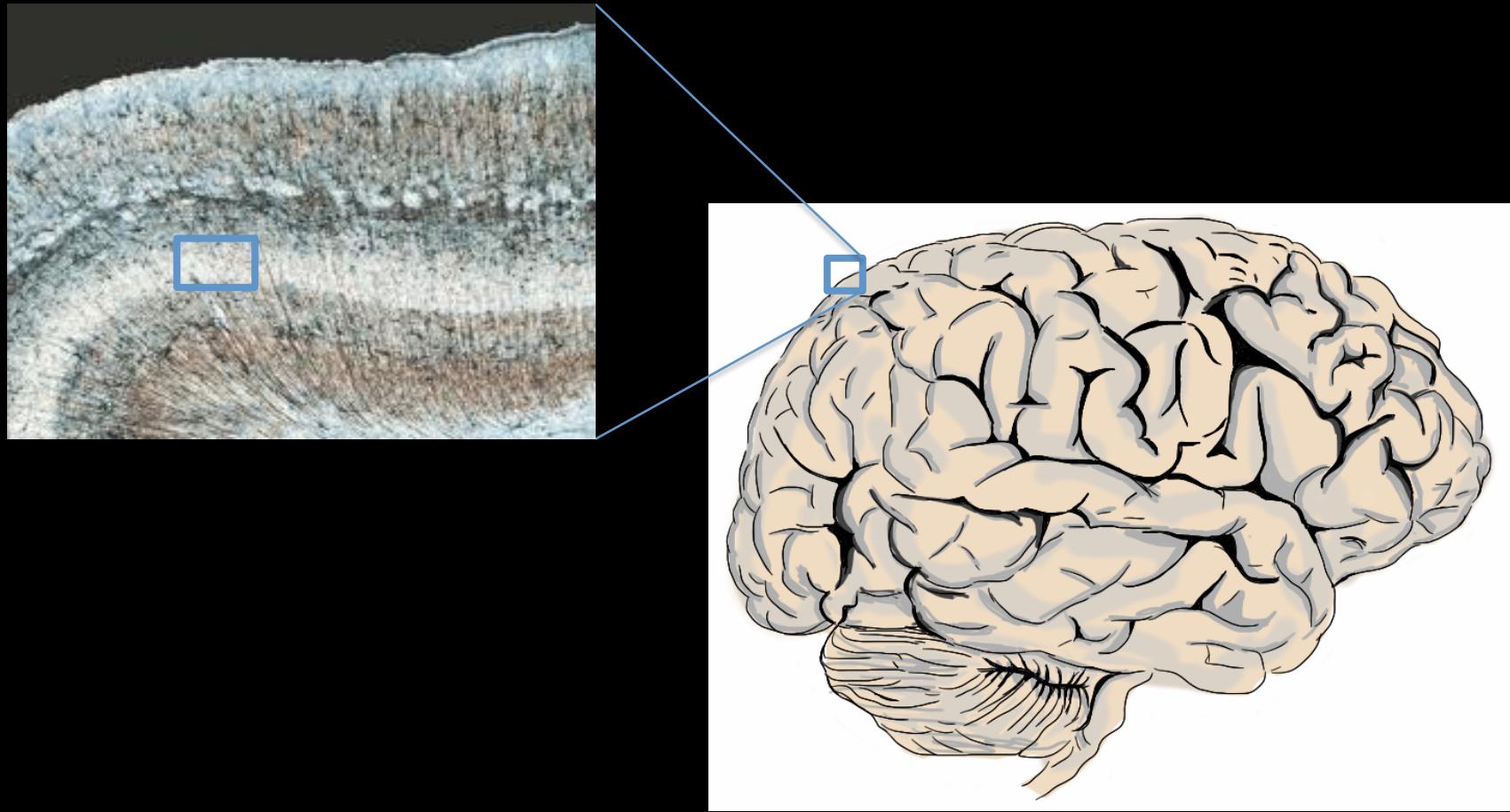
JEFF HAWKINS

with Sandra Blakeslee

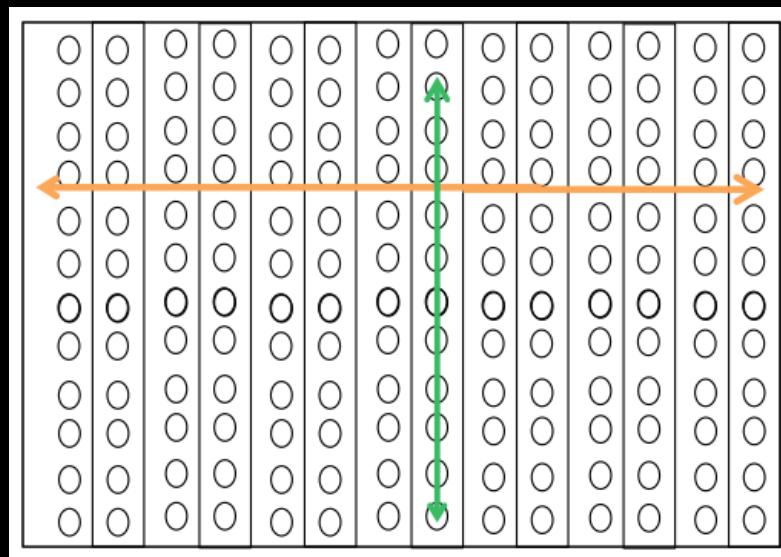
# Cortical Learning Algorithm



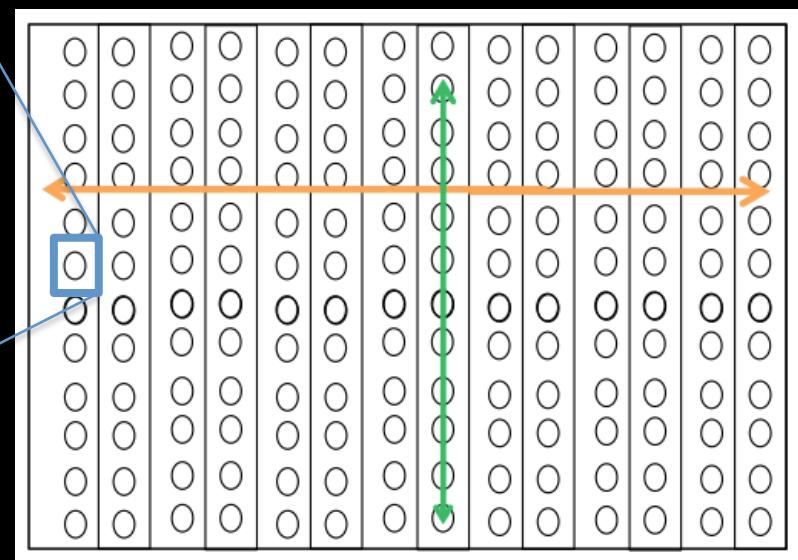
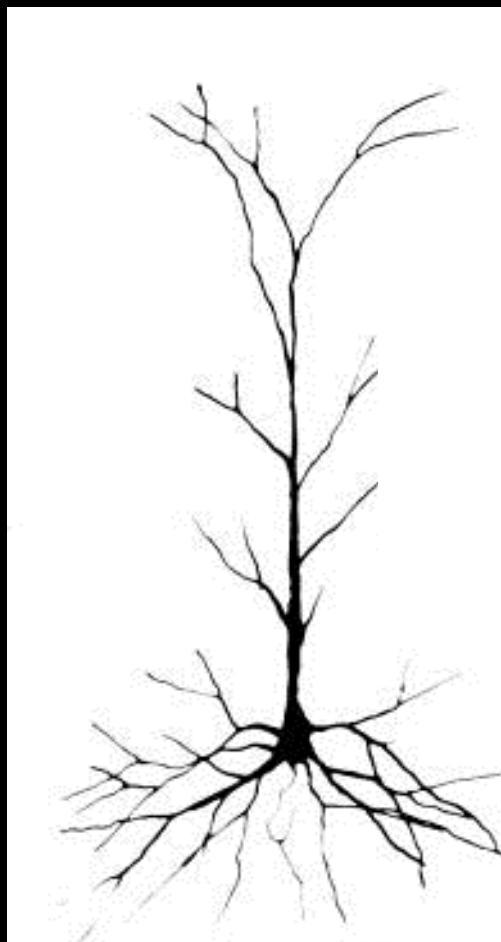
# Cortical Learning Algorithm



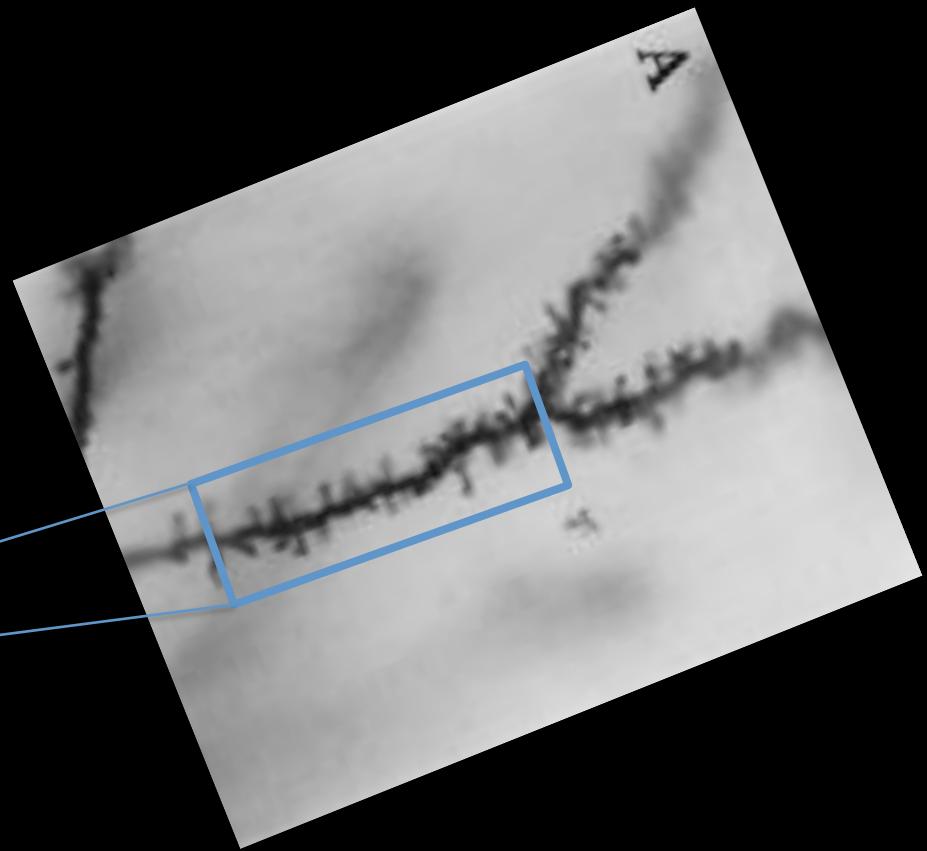
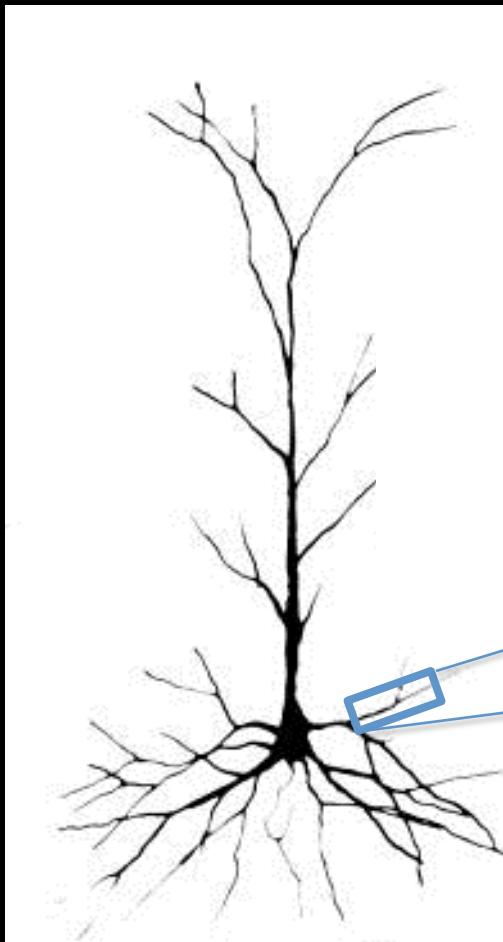
# Cortical Learning Algorithm



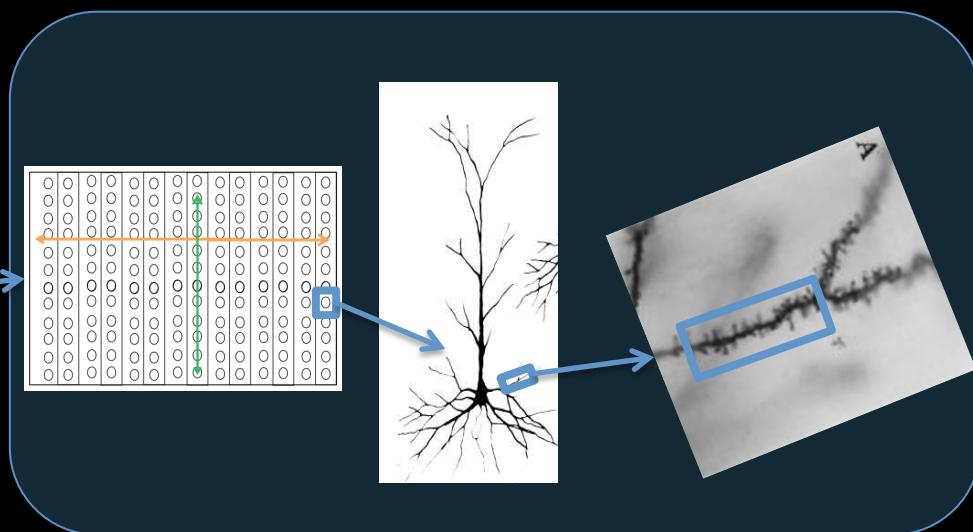
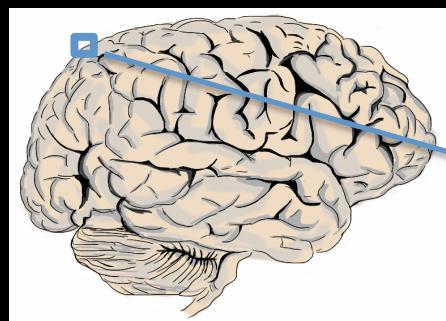
# Cortical Learning Algorithm



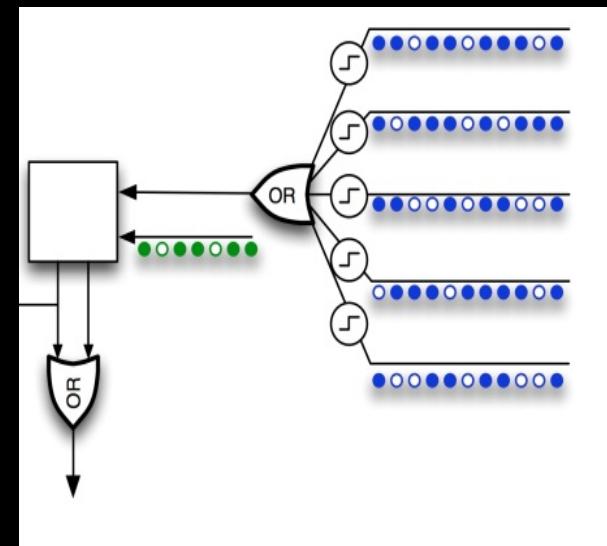
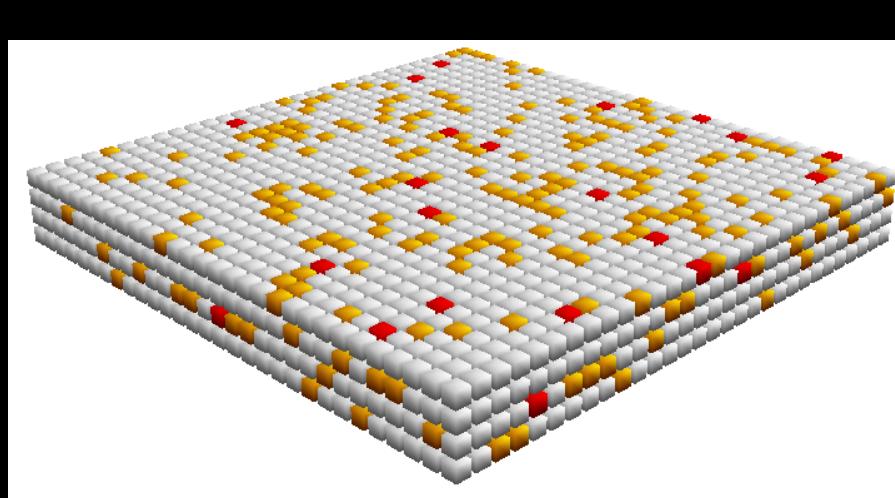
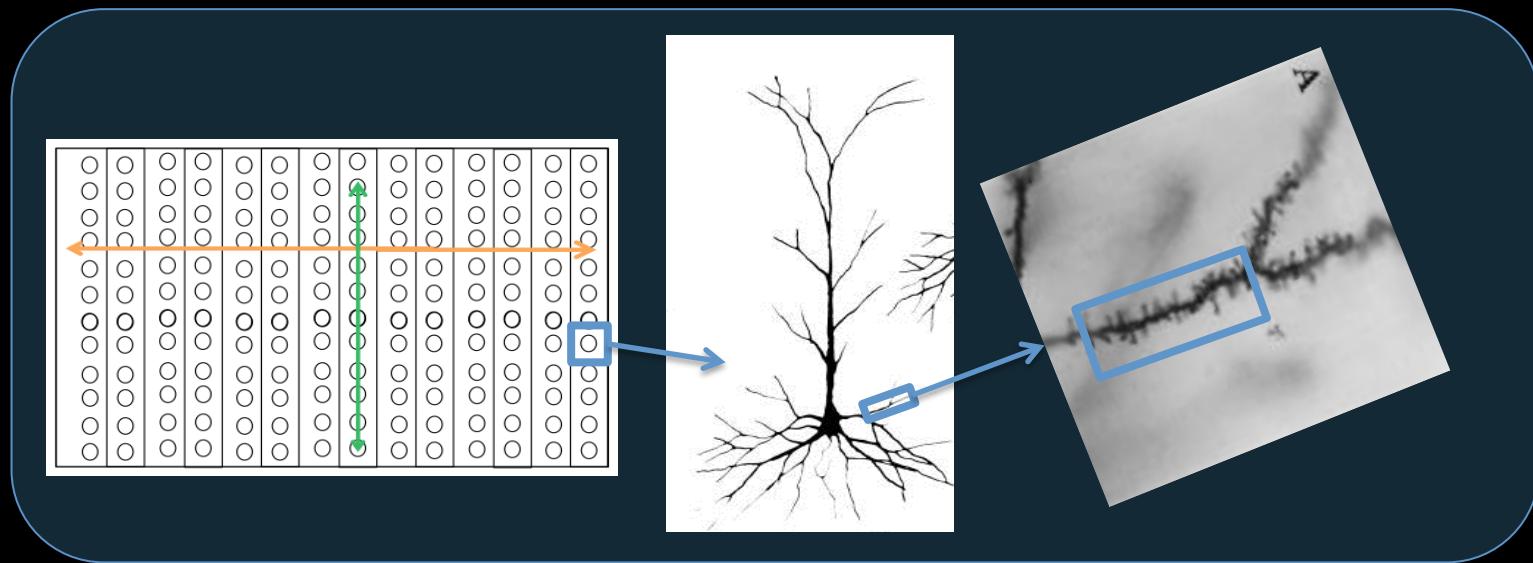
# Cortical Learning Algorithm



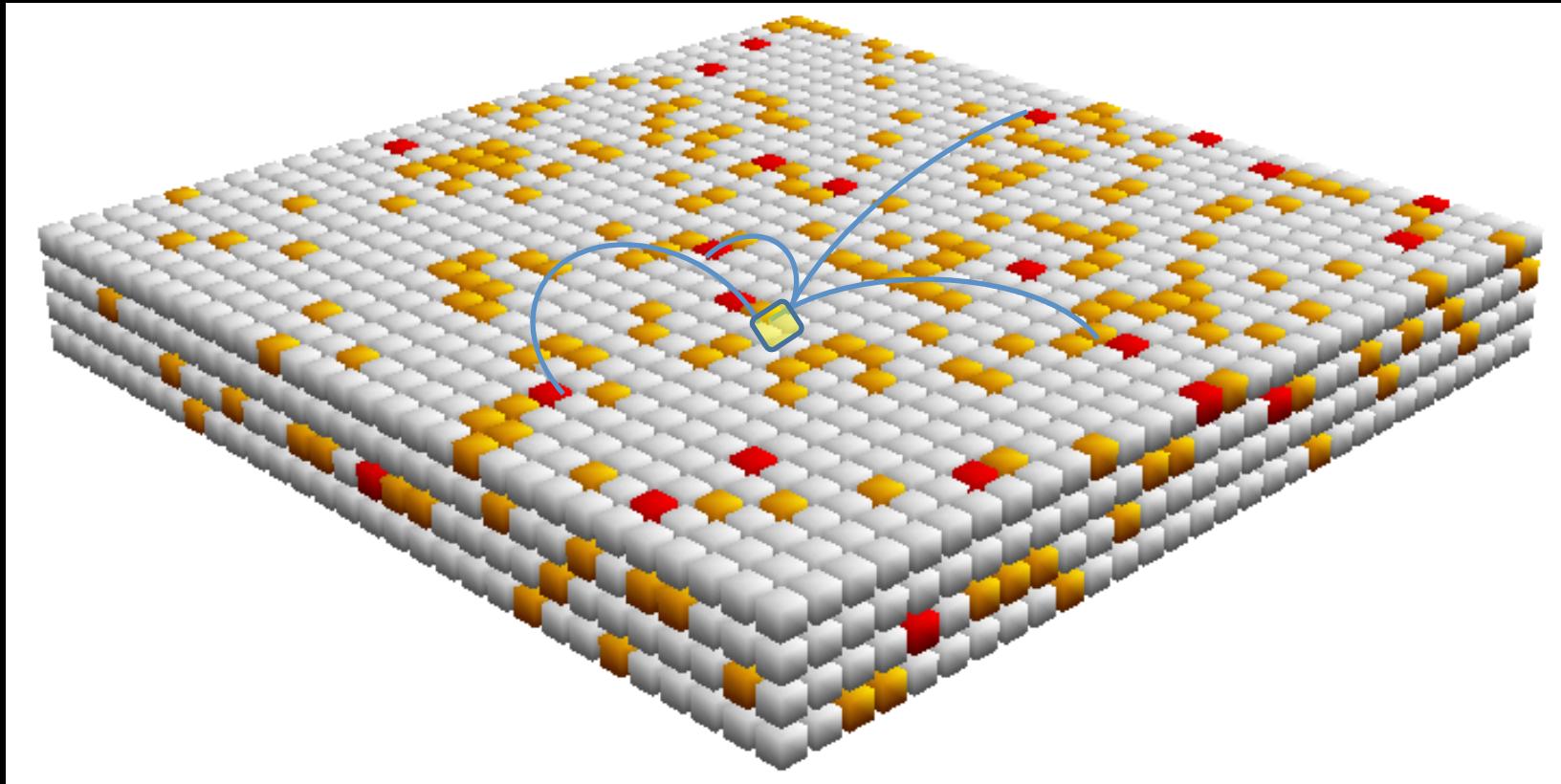
# Cortical Learning Algorithm



# Cortical Learning Algorithm



# Cortical Learning Algorithm



Variable order sequence memory  
Multiple simultaneous predictions  
High capacity

Online learning  
Distributed, fault tolerant  
Semantic generalization

# What does it do?

- accepts streams of data
- learns spatial and temporal patterns
- creates a model of the data
- inference (pattern recognition)
  - predictions
  - anomaly detection

# A taste of the details

- **Sparse Distributed Representations**  
`000001000100000000000000010000`
- **very realistic neuron modeling**
- **layers and connections**

Go to [numenta.org](http://numenta.org)  
and download the  
CLA White Paper 

achieves the same functionality, though in biology there is no equivalent to a dendrite segment attached to a column.

**Synapses**

HTM synapses have binary weights. Biological synapses have varying weights but they are also permanent, meaning a biological neuron can have many precise weights. The use of distributed representations in HTMs and our model of dendrite operation allows us to assign binary weights to HTM synapses with no effect. To model the forming and un-forming of synapses we use two additional concepts of neurons that you may not be familiar with. One is the concept of "potential synapse". This represents all the ones that are close enough to a dendrite segment that they could potentially form a synapse. The second is called "permanence". This is a scalar value assigned to each potential synapse. The permanence of a synapse represents the range of connections between a dendrite and a dendrite segment; the range can go from completely unconnected, to starting to form a synapse but not connected yet, to a minimally connected synapse, to a fully formed synapse. The permanence of a synapse is a scalar value ranging from 0.0 to 1.0. Learning involves incrementing and decrementing a synapse's permanence. When a synapse's permanence is above a threshold, it is connected with a weight of "1". When it is below the threshold, it is unconnected with a weight of "0".

**Overview**

Imagine that you are a region of an HTM. Your input consists of thousands or tens of thousands of bits. These input bits may represent sensory data or they may come from other regions within the hierarchy. They are turning on and off in complex ways. What do you suppose to do with this?

We already have discussed the answer in its simplest form. Each HTM region looks for common patterns in its input and then learns sequences of those patterns. From its memory of sequences, each region makes predictions. That high level description makes it sound easy, but in reality there is a lot going on. Let's break it down a little further into the following three steps:

- 1) Form a sparse distributed representation of the input
- 2) Form a representation of the input in the context of previous inputs
- 3) Form a prediction based on the current input in the context of previous inputs

We will discuss each of these steps in more detail.

**1) Form a sparse distributed representation of the input**  
When you imagine an input to a region, think of it as a large number of bits. In a brain these would be axons from neurons. At any point in time some of these input bits will be active (value 1) and others will be inactive (value 0). The percentage of input bits that are active vary, say from 0% to 60%. The first thing an HTM region does is to convert this input into a new representation that is sparse. For example, the input might have 40% of its bits "on" but the new representation has just 2% of its bits "on".

An HTM region is logically comprised of a set of columns. Each column is comprised of one or more cells. Columns may be logically arranged in a 2D array but this is not a requirement. Each column in a region is connected to a unique subset of the input bits (possibly overlapping with other columns but never exactly the same subset of input bits). As a result, different columns receive different patterns of activation of the columns. The columns with the strongest activation inhibit, or deactivate, the columns with weaker activation. (The inhibition occurs within a radius that can span from very small to the entire region.) The sparse representation of the input is encoded by which columns are active and which are inactive after inhibition. The inhibition function is defined to achieve a relatively constant percentage of columns to be active, even when the number of input bits that are active varies significantly.

Figure 2.1: An HTM region consists of columns of cells. Only a small portion of a region is shown. Each column of cells receives activation from a unique subset of the input. Columns with the strongest activation inhibit columns with weaker activation. The result is a sparse pattern that is encoded by which columns are active and which are inactive after inhibition. The inhibition function is defined to achieve a relatively constant percentage of columns to be active, even when the number of input bits that are active varies significantly.

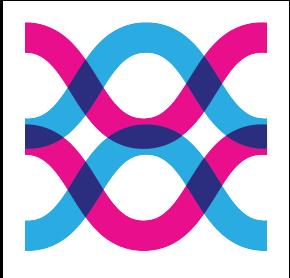
Imagine now that the input pattern changes. If only a few input bits change, some columns will receive a few more or a few less inputs in the "on" state, but the set of active columns will not likely change much. Thus similar input patterns (ones that have a significant number of active bits in common) will map to a relatively stable set of active columns. How stable the encoding depends greatly on what inputs

© Numenta 2011

Page 20

© Numenta 2011

Page 21

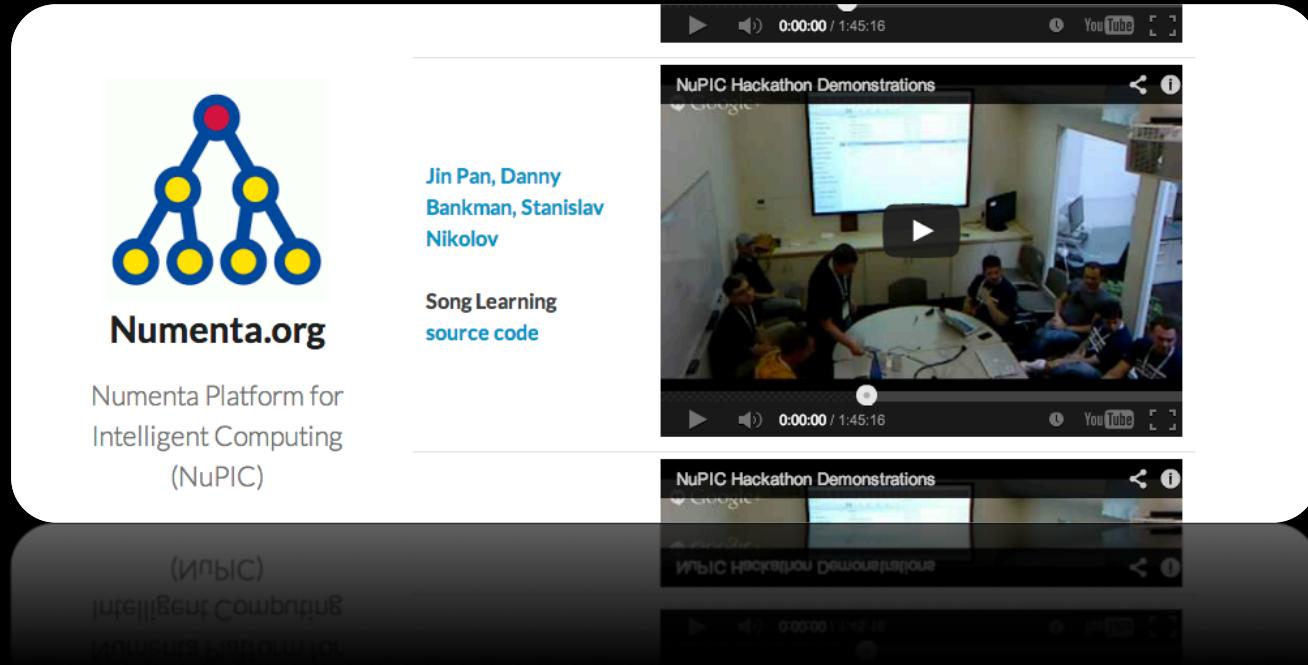


## Current Application: **GROK**

- **SaaS** product
- High velocity, streaming input data
- NuPIC at the core
- Focus on
  - Industrial automation
  - IT applications

# Possible Future Applications

- Vision
- Robotics
- Audio
- Natural language processing
- Fraud detection
- Supply / Demand
- IT
- Automated actions



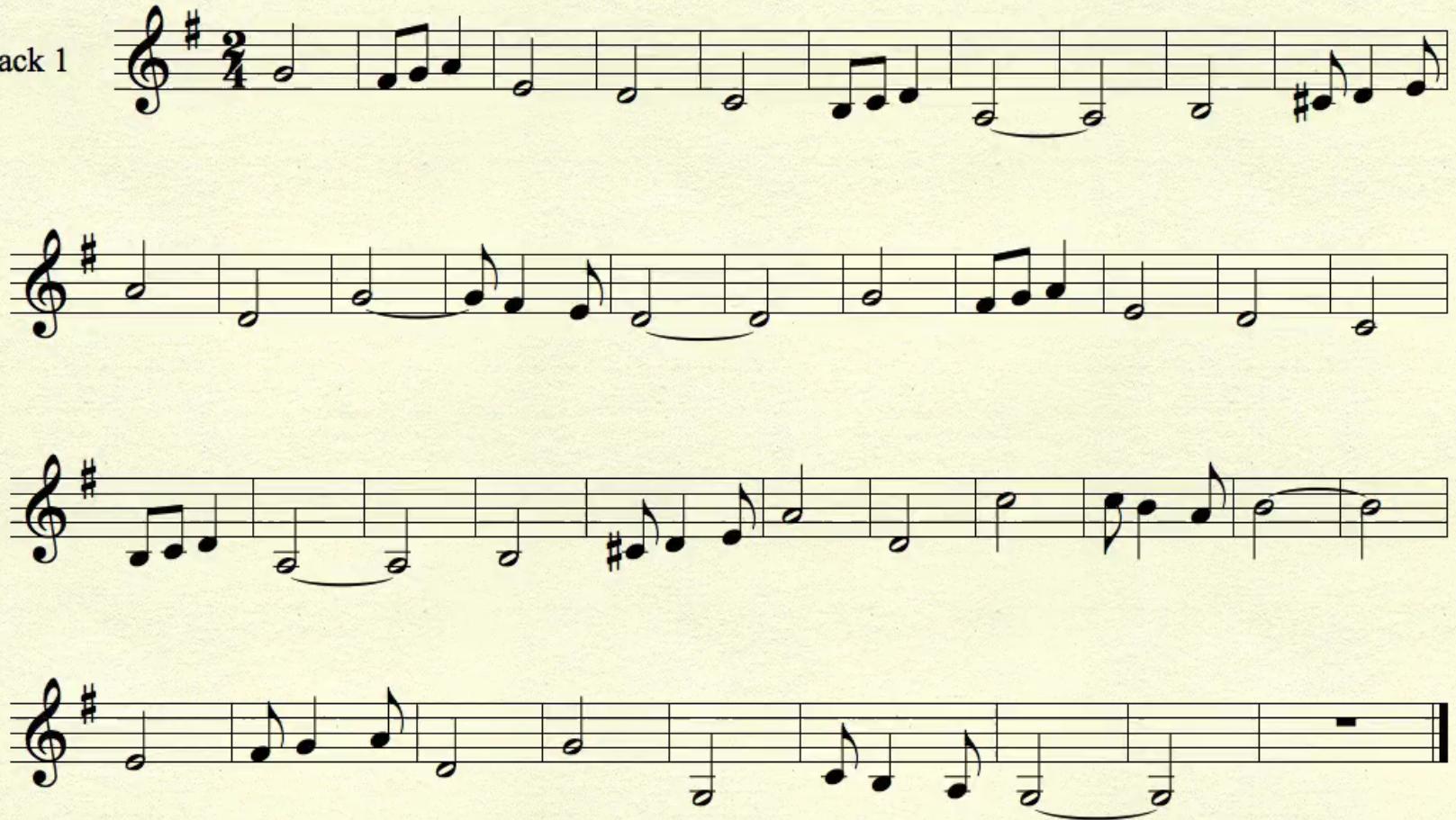
# demonstration

# ONLINE LEARNING

Danny Bankman  
Stanislav Nikolov  
Jin Pan

# The Original Song

Track 1



# After 10 listens

Track 1

A musical score for 'Track 1' in 2/4 time with a key signature of one sharp. The score consists of five staves of music. The first four staves contain measures of eighth and sixteenth notes with various slurs and ties. The fifth staff is a single blank staff ending with a double bar line.

# After 20 listens

Track 1

A musical score for 'Track 1' in G major, 3/4 time. The score consists of five staves of music. The first four staves contain the melody, while the fifth staff is a blank ending bracket. The music features various note values including eighth and sixteenth notes, and includes several slurs and grace notes.

# After 25 listens

Track 1



# Powerful Sequence Memory

- online learning
- temporal patterns
- common phrases
- just like your brain

Scott

**NUTS AND BOLTS**

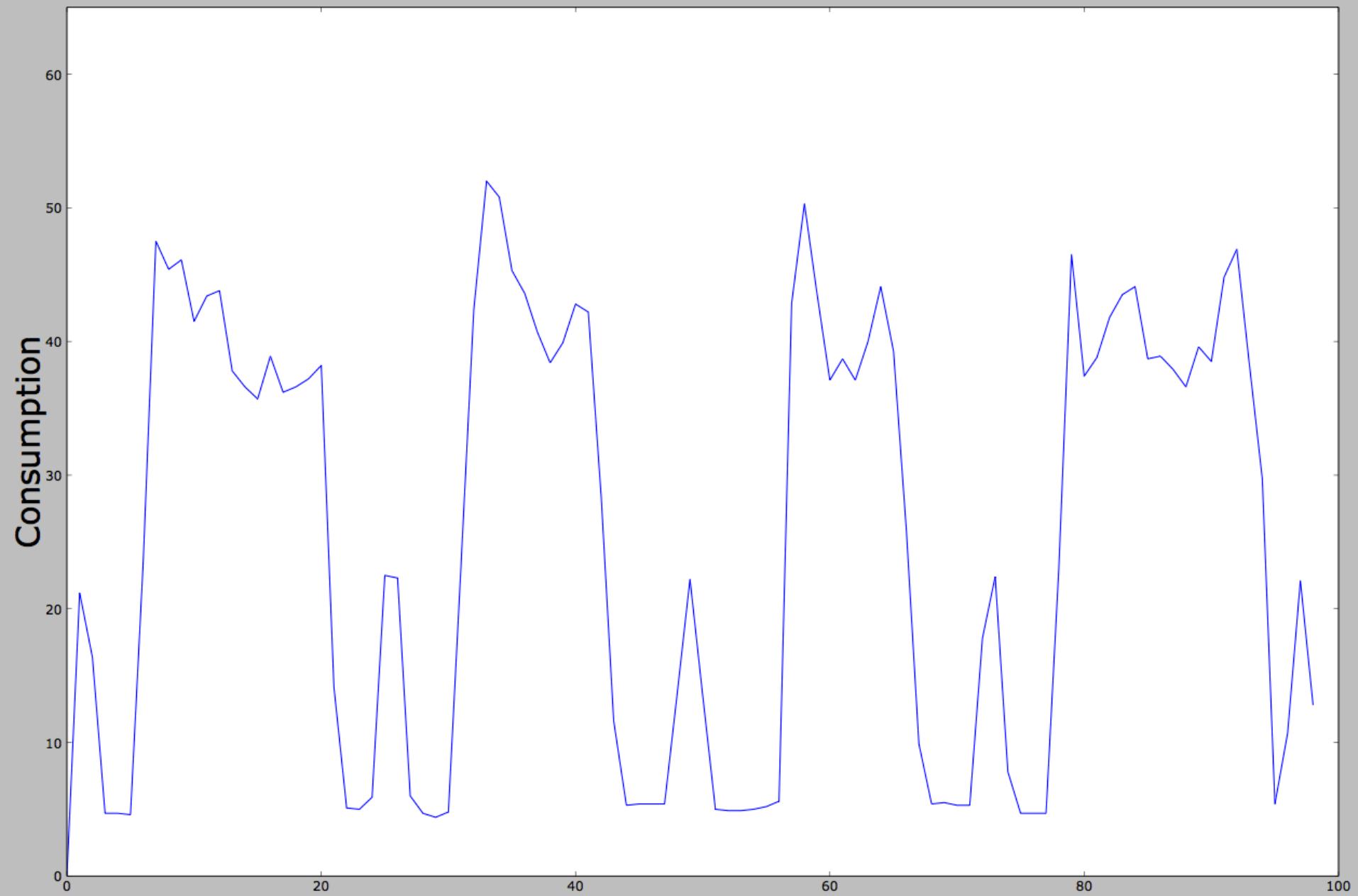
# NuPIC

- Python API (some C++)
- Dual-license with GPL v3 option
- Code on Github
- Travis CI + Github = ❤



# Getting Started Walkthrough

# Building Energy Consumption



# Create the model

```
# Create a model
model = ModelFactory.create(model_params.MODEL_PARAMS)
model.enableInference({'predictedField': 'consumption'})
```

# Model Parameters

```
'sensorParams': {  
    'encoders': {  
        'consumption': {  
            'type': 'AdaptiveScalarEncoder',  
            'n': 100,  
            'w': 21  
        },  
        },  
    },  
    'tpParams': {  
        'columnCount': 2048,  
        'cellsPerColumn': 32,  
  
        'permanenceInc': 0.1,  
        'permanenceDec' : 0.1,  
    },  
    'clParams': {  
        'steps': '1,5',  
    },
```

# Iterate over the data

```
# Create a model
model = ModelFactory.create(model_params.MODEL_PARAMS)
model.enableInference({'predictedField': 'consumption'})

reader = csv.reader(open(_DATA_PATH))
headers = reader.next()

# Iterate over the data file
for i, record in enumerate(reader, start=1):
```

# Pass the data to the model

```
# Create a model
model = ModelFactory.create(model_params.MODEL_PARAMS)
model.enableInference({'predictedField': 'consumption'})

reader = csv.reader(open(_DATA_PATH))
headers = reader.next()

# Iterate over the data file
for i, record in enumerate(reader, start=1):
    # Create a dictionary to pass in to the model
    modelInput = dict(zip(headers, record))
    modelInput["consumption"] = float(modelInput["consumption"])
    modelInput["timestamp"] = datetime.datetime.strptime(
        modelInput["timestamp"], "%m/%d/%y %H:%M")

    # Run the model on a single record and retrieve the result
    result = model.run(modelInput)
```

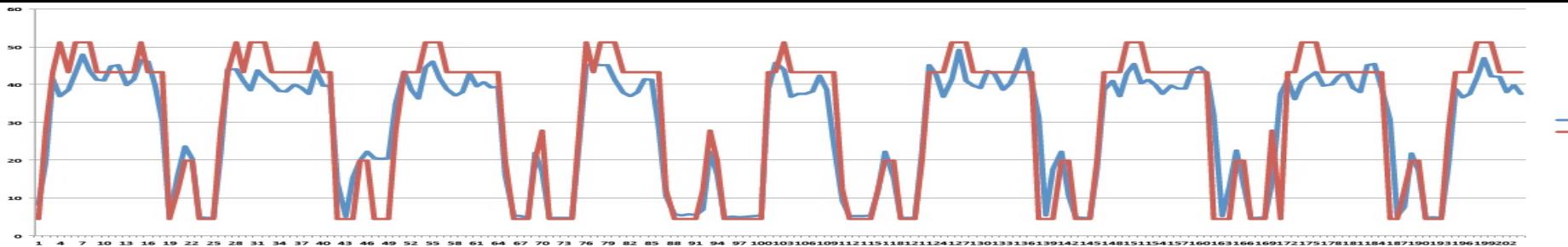
# ModelResult

```
ModelErrort(
    inferences={
        'multiStepPredictions': {
            1: {
                5.2825868514199987: 0.69999516634971859,
                10.69999999999999: 0.07601257054965195,
                22.100000000000001: 0.055294648127235196,
                22.89999999999999: 0.052690624183750749,
            },
            5: {
                38.188079999999999: 0.2275438176777452,
                47.35999999999992: 0.19538808382423584,
                37.39999999999999: 0.12597931862094047,
                45.39999999999999: 0.099123261272031596,
                37.08999999999996: 0.082913215936932752,
                39.280000000000001: 0.077935781935515161,
                43.62999999999995: 0.076405289164189288
            }
        },
        'multiStepBestPredictions': {1: 5.2825868514199987, 5: 38.188079999999999},
    }
    metrics={...}
    predictedFieldIdx=0
    predictedFieldName=consumption
)
```

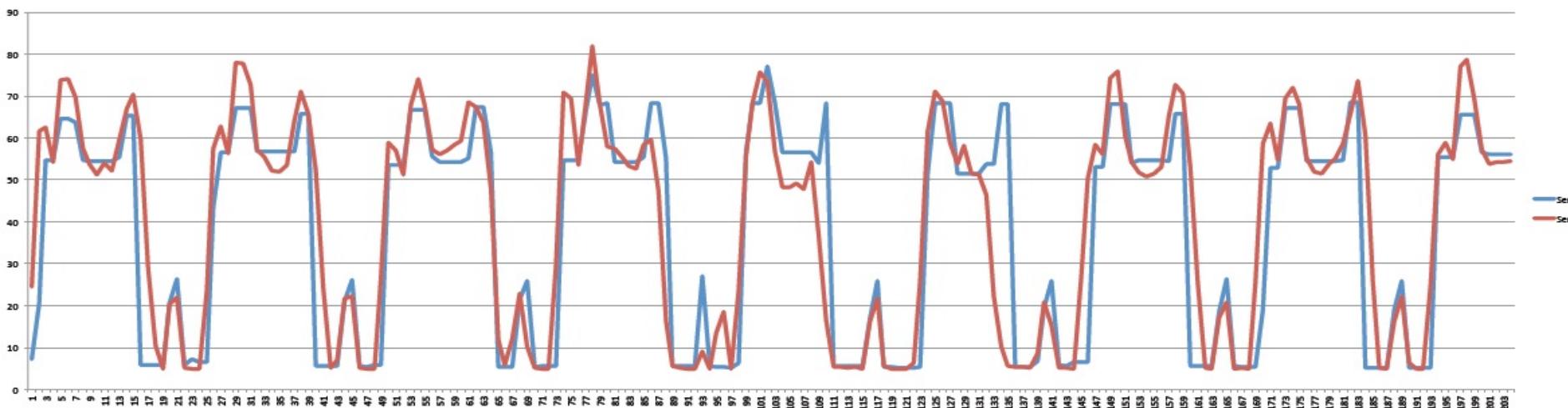
# Live Demo

# Changing patterns

Time epoch 1



Time epoch 2



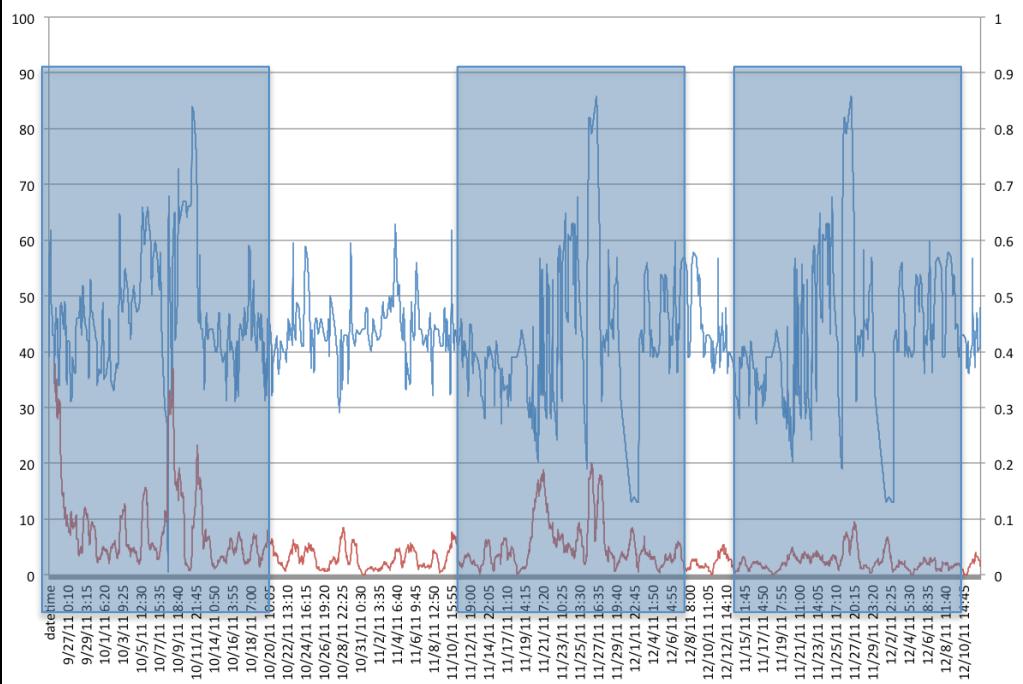
# Anomaly Detection

CLA used to detect anomalies in gear bearing temperature

Detects anomalies based on temporal characteristics

Thresholds are not sufficient

Reduces downtime and maintenance costs



Gear bearing temperature & anomaly score

# Quick Review

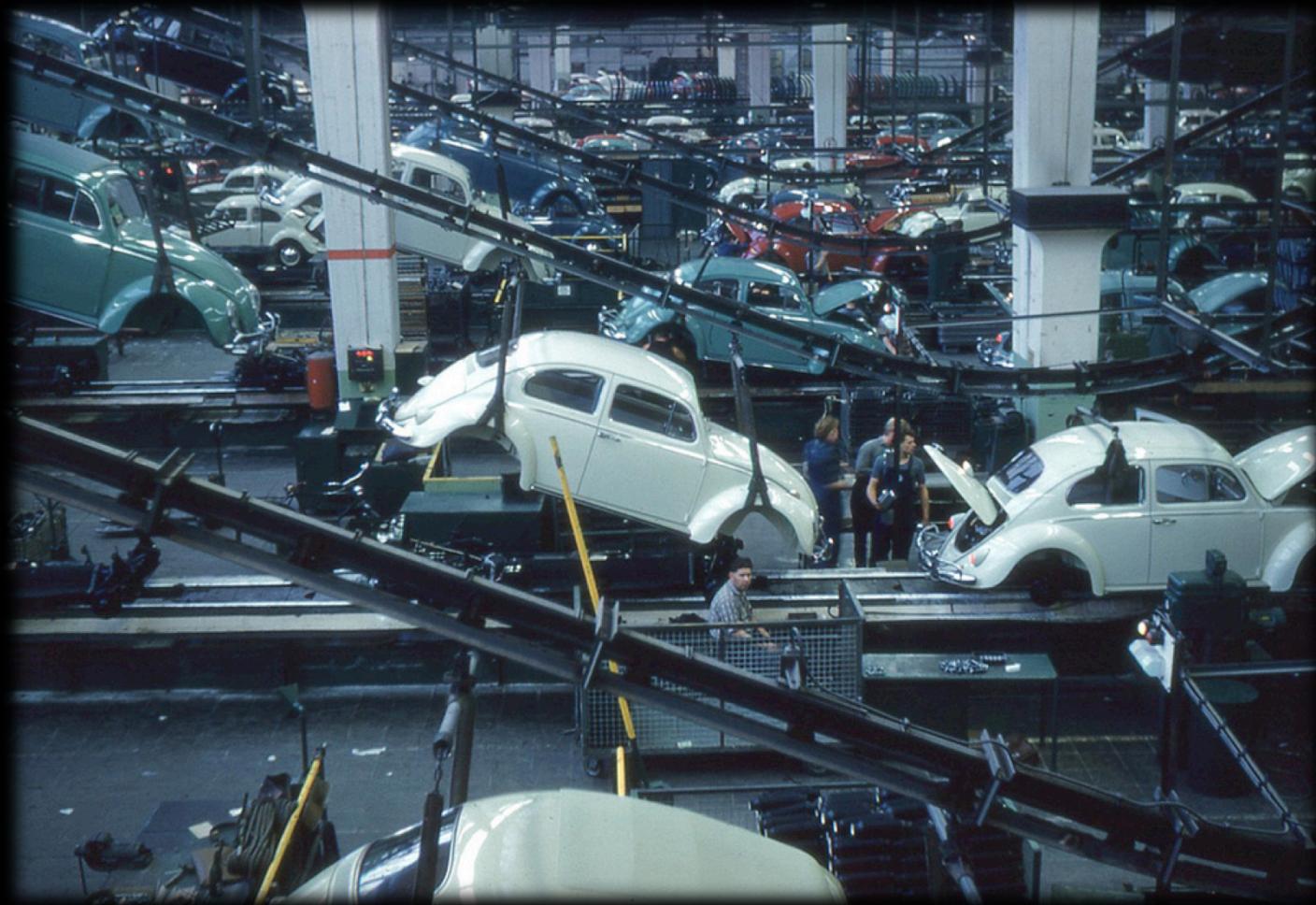
- streamed energy consumption data into CLA
- it learned spatial and temporal patterns in data
  - created a model of the data
- inference (pattern recognition)
  - predictions (1 hour, 5 hours)
  - anomalies

# Ideas to inspire

Matt

**OPEN SOURCE COMMUNITY**

# Why this OS project is different



# Why open source the tool...

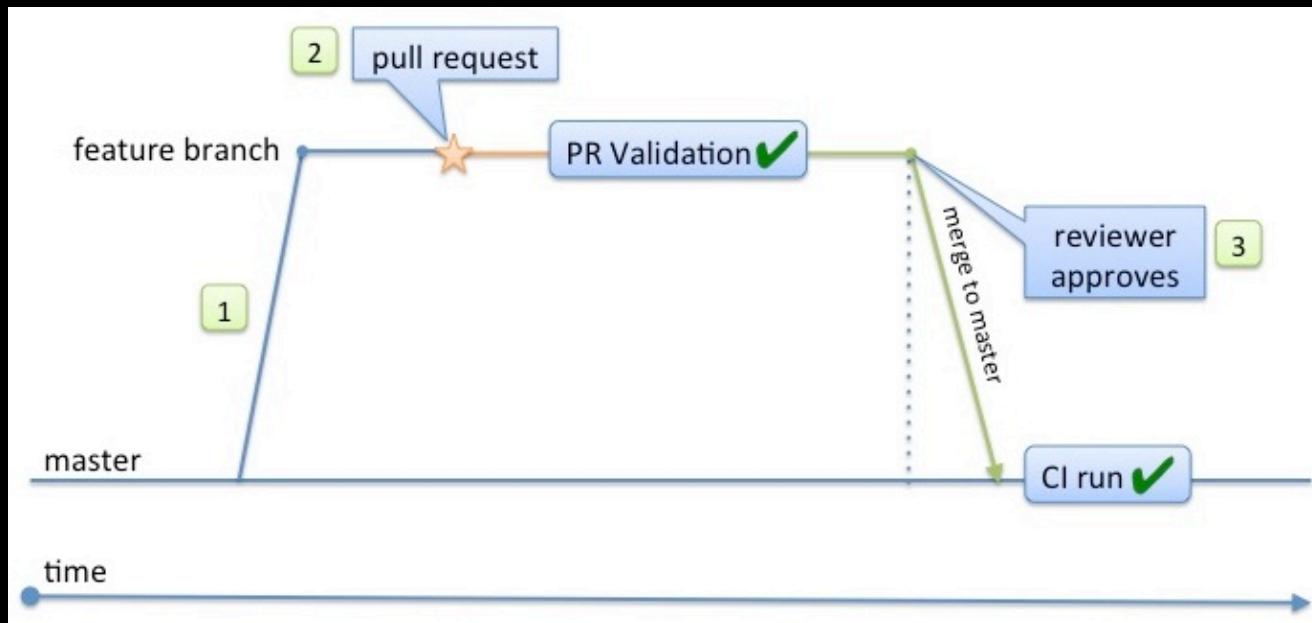


# When you can open source the engine?



# Single Source Tree

- Grok builds directly from NuPIC
- master is always green
- testing occurs within Pull Requests



# Github / Travis-CI

The image shows a dual-monitor setup. The left monitor displays the GitHub repository page for 'numenta/nupic'. The right monitor displays the Travis-CI build history for the same repository.

**GitHub Repository Page (Left Monitor):**

- URL: <https://github.com/numenta/nupic>
- Repository: numenta / nupic
- Public
- 314 commits, 2 branches
- Branch: master
- Merge pull request #125 from scottpurdy/examples
- rhylight authored 7 days ago
- Everything.xcodeproj: Squashing code from internal repo.
- build\_system: Clean up the clean function (so meta).
- conf: Adds logging configuration file and include
- examples: Remove unused metrics code from cpu e
- external: Merge branch 'master' into recent-libs
- githooks: Squashing code from internal repo.
- lang: Merge branch 'master' into python27-sup
- nta: gcc/clang: fix template lookup
- py: Merge branch 'master' of git://github.com/
- qa: Removes qa/shared\_data/plugins
- tests: NUP-2097 #resolve Update C++ CLAClass
- .gitignore: gitignore: remove nonexistent file referen
- .travis.yml: tests: ignore on test due to missing dep.
- LICENSE.txt: Squashing code from internal repo.
- Makefile.ami: Revert "build: proper make clean target"
- README.md: Update the README with wiki links and r
- build.sh: Changes build scripts to stop on first failu
- cleanbuild.sh: Changes build scripts to stop on first failu
- env.sh: fix bash operator
- run\_tests.sh: Trying to run\_tests.sh with bash instead o
- trunk.sln: Squashing code from internal repo.

**Travis-CI Build History (Right Monitor):**

- URL: <https://travis-ci.org/numenta/nupic/builds>
- Repository: numenta/nupic
- Builds (Listed):
  - 280: Merge pull request #125 from scottpurdy/examples (3c0226a (master))
  - 277: Merge pull request #119 from glisho/examples (fb4cfef (master))
  - 275: Merge pull request #120 from ischeinman/clRegionName\_asci (c5d9f7d (master))
  - 273: Merge pull request #121 from marianetti/nup-2031 (916a324 (master))
  - 257: Merge pull request #118 from scottpurdy/buildfailure (265102d (master))
  - 254: Merge pull request #115 from scottpurdy/hotgym (a06ac02 (master))
  - 249: Merge pull request #111 from scottpurdy/clean (12b1de3 (master))
  - 245: Merge pull request #108 from breznak/recent-libs (6216e2d (master))
  - 243: Merge pull request #85 from breznak/python27-support (0972016 (master))
  - 241: Merge pull request #113 from scottpurdy/master (a373648 (master))
  - 237: Merge pull request #89 from breznak/compilation (cfa3925 (master))
  - 231: Merge pull request #110 from marianetti/spatialClassification (bf39070 (master))
  - 229: Merge pull request #109 from scottpurdy/streaming (1f08f97 (master))
  - 226: Merge pull request #106 from scottpurdy/load\_model (6d16e09 (master))
  - 223: Merge pull request #107 from scottpurdy/cleanbuild (0d9e795 (master))
  - 219: Merge pull request #104 from scottpurdy/master (f4e860c (master))
  - 209: Merge pull request #105 from breznak/travis-run-examples (c970387 (master))
  - 202: Merge pull request #103 from octopush/link-to-issues (8784dc6 (master))
  - 200: Merge pull request #102 from subutai/master (3fc6db6 (master))
  - 198: Merge pull request #101 from subutai/master (4083c97 (master))
  - 197: Merge pull request #100 from scottpurdy/master (6a70fd (master))
  - 191: Merge pull request #99 from scottpurdy/master (6552d9c (master))
  - 186: Merge pull request #91 from octopush/redundant\_validationerror\_exception (4473142 (master))
  - 180: Merge pull request #87 from numenta/updatedSpatialClassification (6bf8871 (master))
  - 178: Merge branch 'master' into updatedSpatialClassification (8b23f71 (updatedSpatialClassification))
- Build History Tab
- Pull Requests Tab
- Branch Summary Tab

# September 2013 Archives by thread

- Messages sorted by: [\[ subject \]](#) [\[ author \]](#) [\[ date \]](#)
- [More info on this list...](#)

Starting: Sun Sep 1 05:10:25 EDT 2013

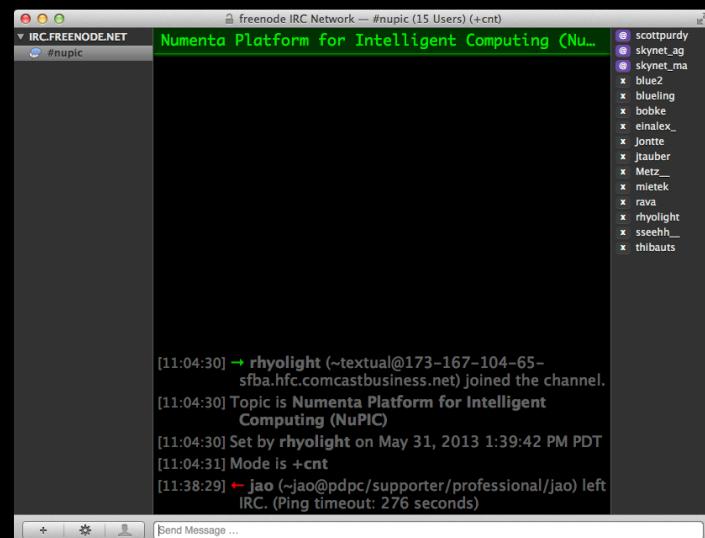
Ending: Tue Sep 24 17:52:29 EDT 2013

Messages: 321

- [\[nupic-dev\] Some comments on](#) *Tim McNamara*
  - [\[nupic-dev\] Some comments on](#) *Matthew Taylor*
    - [\[nupic-dev\] Some comments on](#) *Tim McNamara*
    - [\[nupic-dev\] Some comments on](#) *Austin Marshall*
      - [\[nupic-dev\] Some comments on](#) *Tim McNamara*
        - [\[nupic-dev\] Some comments on](#) *Matthew Taylor*
  - [\[nupic-dev\] Some comments on](#) *Erik Blas*
- [\[nupic-dev\] Inter-layer plumbing](#) *Patrick Higgins*
  - [\[nupic-dev\] Inter-layer plumbing](#) *Jeff Hawkins*
    - [\[nupic-dev\] Inter-layer plumbing](#) *Tim Boudreau*
  - [\[nupic-dev\] Inter-layer plumbing](#) *Tim Boudreau*
    - [\[nupic-dev\] Inter-layer plumbing](#) *Jeff Hawkins*
      - [\[nupic-dev\] Inter-layer plumbing](#) *Chetan Surpur*
      - [\[nupic-dev\] Inter-layer plumbing](#) *Fergal Byrne*
      - [\[nupic-dev\] Inter-layer plumbing](#) *Jeff Hawkins*
      - [\[nupic-dev\] Inter-layer plumbing](#) *Chetan Surpur*
      - [\[nupic-dev\] Inter-layer plumbing](#) *Jeff Hawkins*
      - [\[nupic-dev\] Inter-layer plumbing](#) *Chetan Surpur*
      - [\[nupic-dev\] Inter-layer plumbing](#) *Fergal Byrne*
  - [\[nupic-dev\] Inter-layer plumbing](#) *Ivan Sytenko*
- [\[nupic-dev\] Brain Protein](#) *may2003*
- [\[nupic-dev\] NuPIC for Model Predictive Control](#) *Ralph Dratman*
  - [\[nupic-dev\] NuPIC for Model Predictive Control](#) *Ian Danforth*
    - [\[nupic-dev\] NuPIC for Model Predictive Control](#) *Matthew Taylor*
      - [\[nupic-dev\] NuPIC for Model Predictive Control](#) *Pedro Tabacof*
      - [\[nupic-dev\] NuPIC for Model Predictive Control](#) *Chetan Surpur*
      - [\[nupic-dev\] NuPIC for Model Predictive Control](#) *Chetan Surpur*
      - [\[nupic-dev\] NuPIC for Model Predictive Control](#) *Chetan Surpur*
      - [\[nupic-dev\] NuPIC for Model Predictive Control](#) *Ian Danforth*
      - [\[nupic-dev\] NuPIC for Model Predictive Control](#) *Erik Blas*
  - [\[nupic-dev\] Documenting models](#) *Tim McNamara*
    - [\[nupic-dev\] Documenting models](#) *Matthew Taylor*
  - [\[nupic-dev\] Python 2.7 build with 2.6 Error](#) *Matthew Taylor*
    - [\[nupic-dev\] Python 2.7 build with 2.6 Error](#) *Marek Otahal*
      - [\[nupic-dev\] Python 2.7 build with 2.6 Error](#) *Matthew Taylor*
      - [\[nupic-dev\] Python 2.7 build with 2.6 Error](#) *Marek Otahal*
      - [\[nupic-dev\] Python 2.7 build with 2.6 Error](#) *Sinon Galvin*
    - [\[nupic-dev\] Python 2.7 build with 2.6 Error](#) *Joshua Auerbach*
      - [\[nupic-dev\] Python 2.7 build with 2.6 Error](#) *Erik Blas*
      - [\[nupic-dev\] Python 2.7 build with 2.6 Error](#) *Sinon Galvin*

# Community

- 63 contributors
- 227 list members
- 10 daily messages
- [public issue tracker](#)
- Bi-weekly sprints
- IRC #nupic



# June 2013 Hackathon

- Robot movement
- IR Sensors
- Weather
- Human movement
- Tic Tac Toe
- CPU usage

Screenshot of a web browser showing the Numenta.org news page for the June 2013 Hackathon Outcome. The page features a header with the Numenta logo and navigation links for Home, NuPIC, Community, Events, Licenses, News, and FAQ. Below the header, there is a section titled "Demonstrations" with three video thumbnails, each showing a group of people in a conference room with a whiteboard and a large screen displaying data. The first video is for Erik Blas's "IR sensor predictions" demo, the second for Kevin Archie's "Prediction Charting" demo, and the third for Alexander Van Dijk and Sam Gateau's "Streaming Weather" demo.

**Demonstrations**

The demonstrations in their entirety are [here](#), but it might be easier for you to navigate by project. I've linked each demo individually below.

**Erik Blas**  
IR sensor predictions  
[source code](#)

**Kevin Archie**  
Prediction Charting  
[source code](#)

**Alexander Van Dijk,  
Sam Gateau**  
Streaming Weather  
[source code](#)

# Getting Involved

- Community Page on [numenta.org](http://numenta.org)
- Public JIRA for ticket tracking
- You can always tell what we're doing
- Newbie tasks
- Hackathon in SF this fall
  - Nov 2 - 3
  - [numenta.org/events.html](http://numenta.org/events.html)

# Next Steps for you

- Read the CLA White Paper
- Read *On Intelligence*
- Join our mailing list
  - [nupic@lists.numenta.org](mailto:nupic@lists.numenta.org)
- Videos: [numenta.org/media.html](http://numenta.org/media.html)

# Contacts

[matt@numenta.org](mailto:matt@numenta.org)

[scott@numenta.org](mailto:scott@numenta.org)

[jhawkins@numenta.org](mailto:jhawkins@numenta.org)

<http://numenta.org>

We'll be outside after

