Investigating the Brain's Computational Paradigm

December 17, 2014

J E Smith

jes.at.ece.wisc.edu@gmail.com

Introduction

There Is Only One Grand Challenge in Computing

Discover and emulate the brain's computational paradigm

We know *what* the brain can do...

How does the brain do it?

Can we *construct* hardware that follows the same paradigm?

Topics

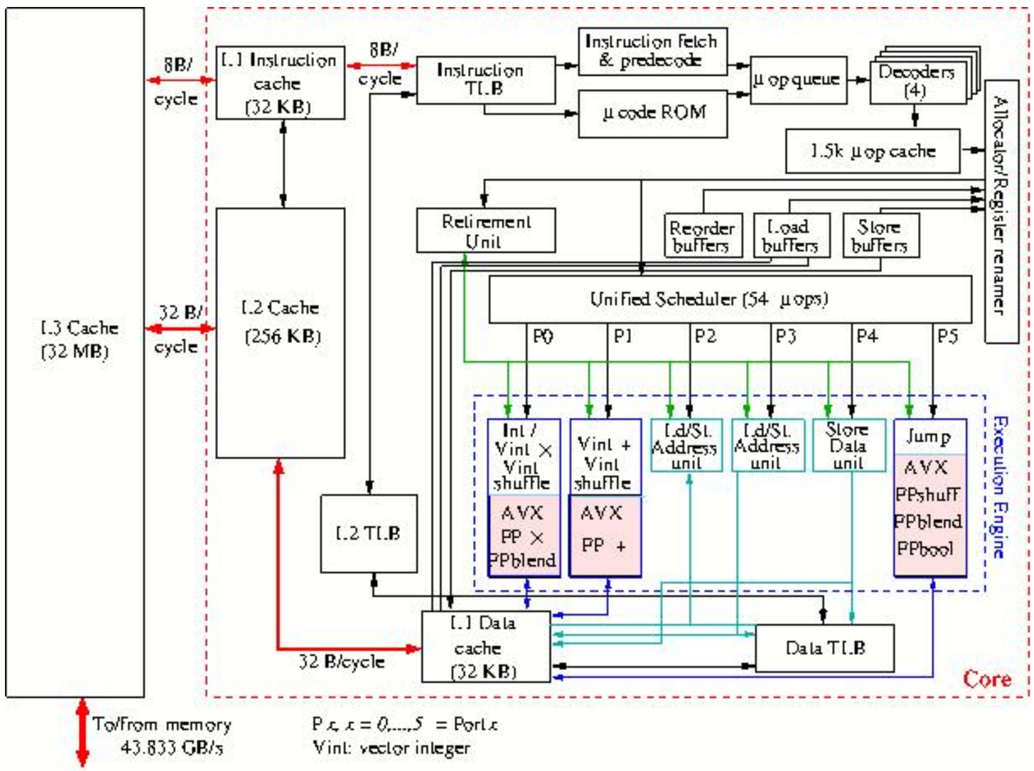
- Computational paradigms
  - Discuss via a very familiar example
  - This will be a guiding analogy
- A neuron model to support computation
  - Basic elements and operation
- Temporal computation and communication
  - What is it?
  - How it is different from conventional computational models?
  - How it is different from traditional neural networks?
- An architecture under development
  - Consider it to be a case study for tackling the problem
  - Progress to date

Initial Comment

- Everyone speaks with authority, but no one knows
  - Preface everything I say with “In my opinion”

Imagine...

- An advanced civilization which computes in an entirely different way than we
- Say this civilization comes across one of our high-end computers
    - Cores are multi-way out-of-order superscalar processors
    - Multiple processors with complex memory hierarchy



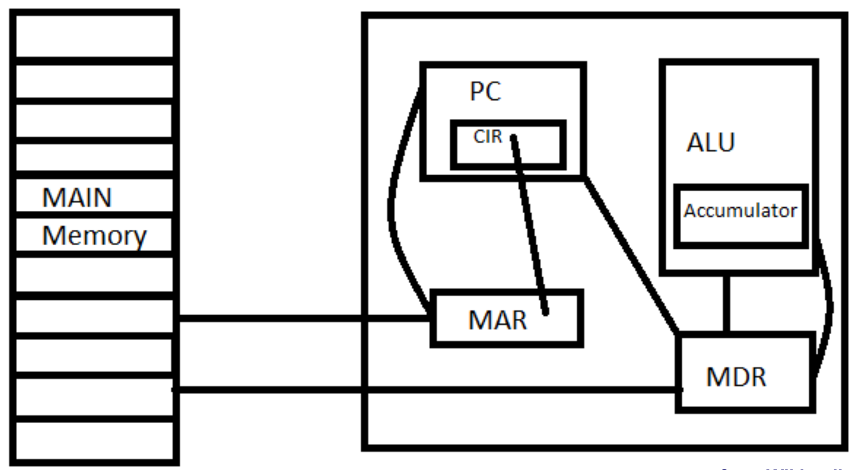
http://www.euroben.nl/reports/web12/xeon.php

Discovering the Computational Paradigm

- *Breakthrough 1: Binary communication is discovered*
  - After a lot of work, analyzing voltages on wires
  - Exact voltage levels aren't important, only 0s and 1s !
- *Breakthrough 2: Logic gates*
  - A small set of basic building blocks are used everywhere
  - Signals flow from inputs to outputs
- *Breakthrough 3: Combinational Logic*
  - Understanding the operation of a fixed point adder would be a triumph!
  - Purely combinational logic would be extremely useful
    - The advanced civilization could eventually develop systems with thousands of combinational logic levels
    - These could perform very useful computations
    - Without even knowing the complete computing paradigm
- *Breakthrough 4: The function of feedback and clocking*

Eventually...

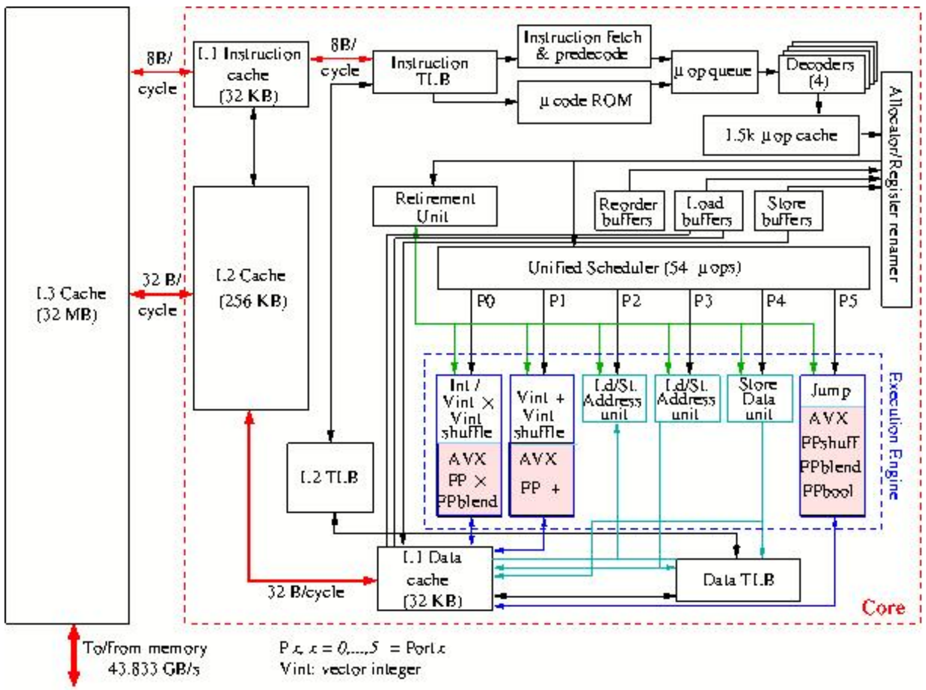
- Someone might eventually discover the paradigm
  - But it is very much obscured – mostly by performance enhancements
  - Eg., find *the* PC in a superscalar processor



from Wikipedia

What Is *Not* Part of the Paradigm?

- Short answer: almost everything!
- All physical properties of CMOS ckts
- Performance enhancements
  - Branch prediction
  - Memory hierarchies
  - n-Way issue
- Power savings
  - Clock gating
  - Voltage scaling
- Reliability
  - ECC in memories
  - Redundancy at server level
- Etc.
  - Buffering state for precise traps



from Wikipedia

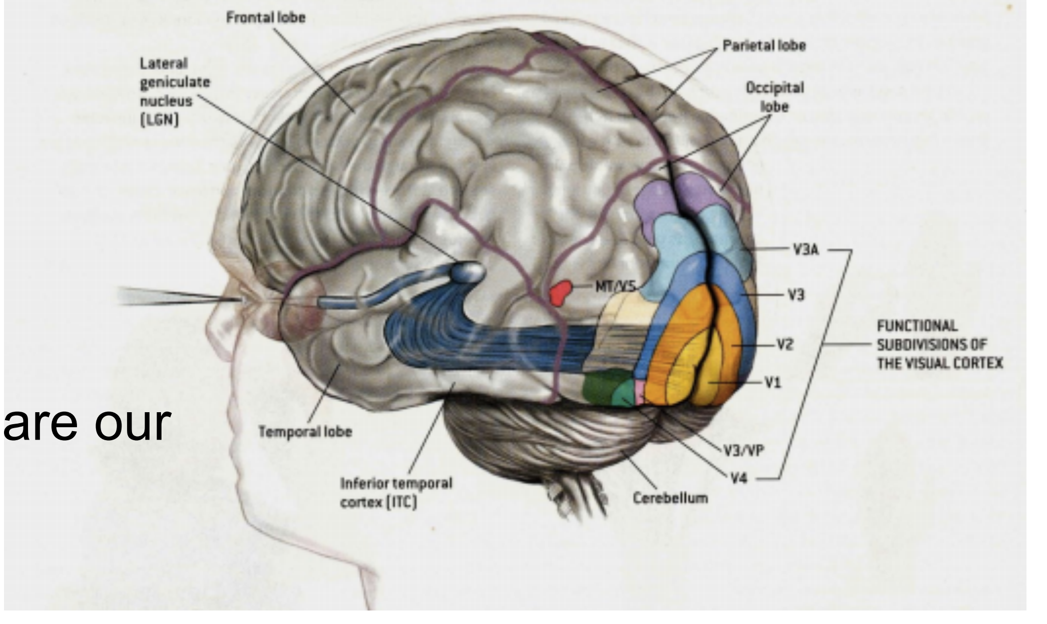
There is a lot of stuff in the brain that isn't part of its paradigm, as well!

Biological Overview

Neocortex

- Thin sheet of neurons
  - Area of about 2500 cm^2
    - Folds increase surface area
  - 2 to 4 mm thick.
  - Approx 100 billion total neurons (human)
- Hierarchical Structure
  - Neurons
  - Columns
  - Macro-Columns
  - Regions
- Neuron and column levels are our focus

from wiki.bethanycrane.com

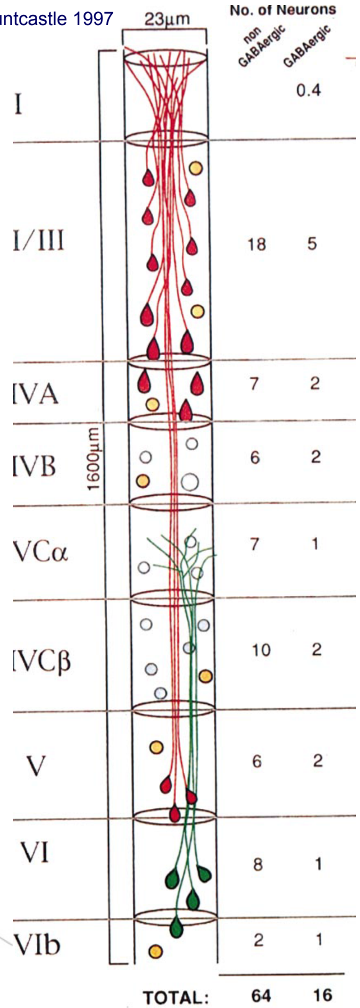


Hierarchy

from Hill et al. 2012

Neuron

from Mountcastle 1997

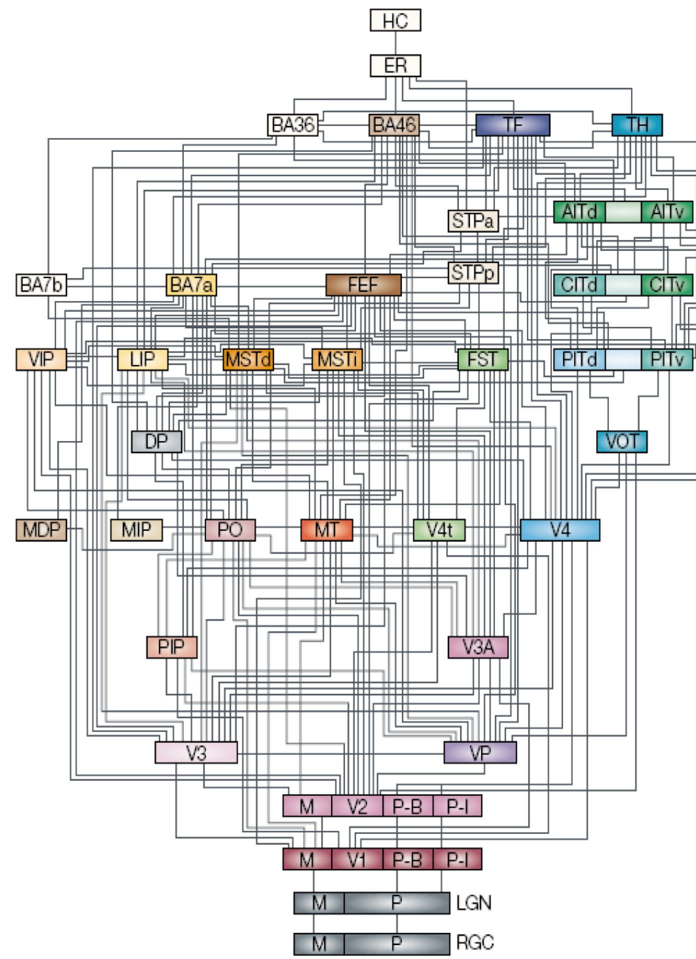


Column
O(100) neurons

from Ramon y Cajal (wikipedia)

Macro-Column
O(100) columns

from Felleman and Van Essen (1997)



Regions
Many Macro-Columns

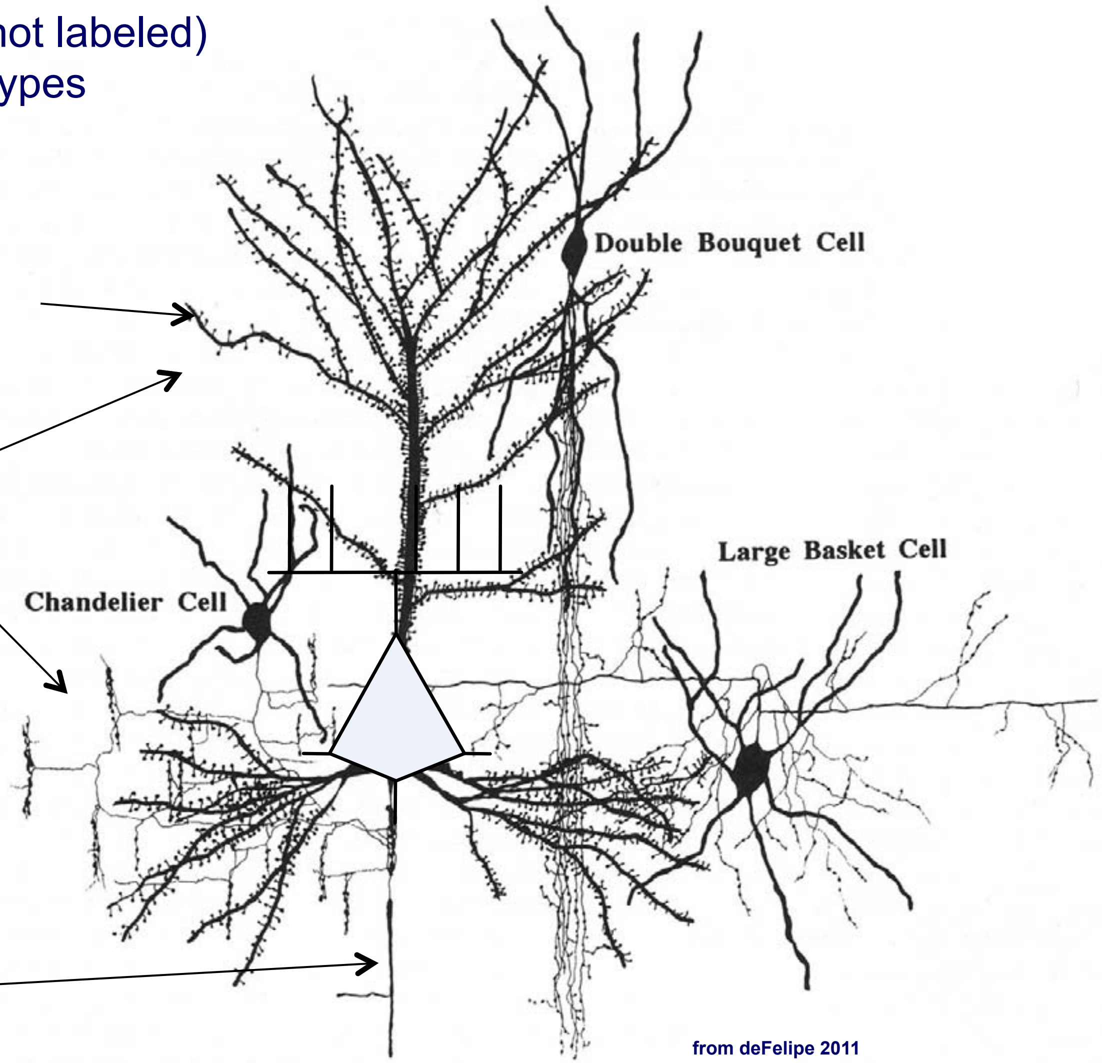
Biological Neurons

pyramid cell (center, not labeled) surrounded by three types inhibitory cells

tiny dots are synapses (connection points)

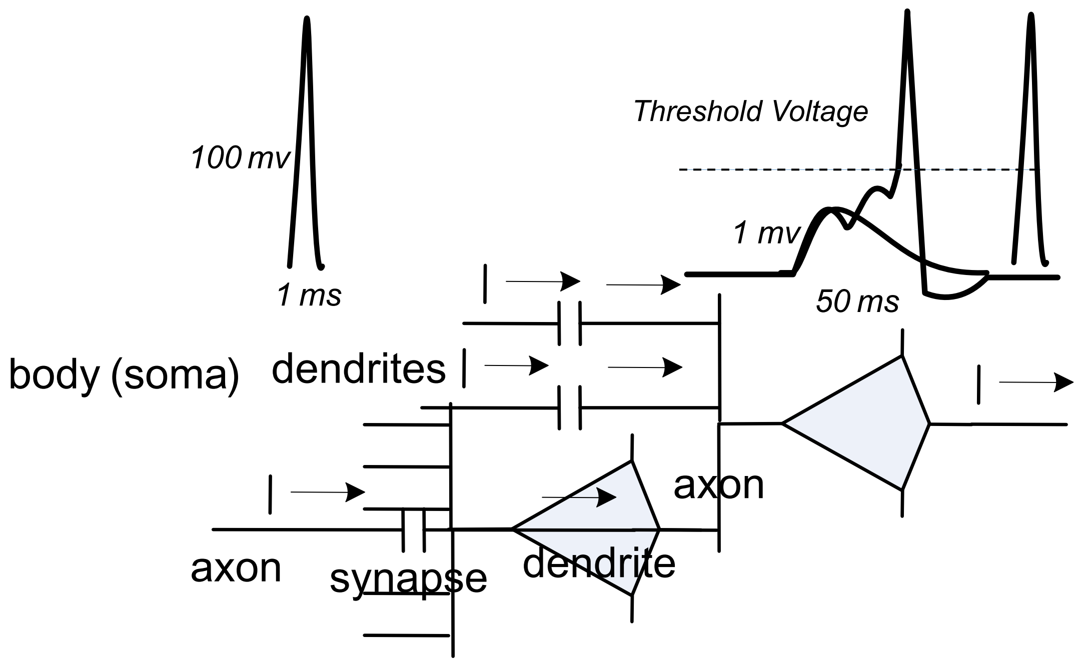
Dendrites (Inputs)

Axon (Output)



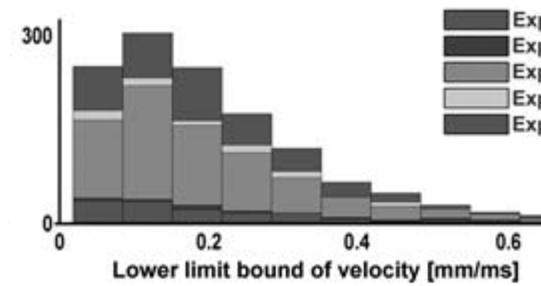
from deFelipe 2011

Neuron Operation

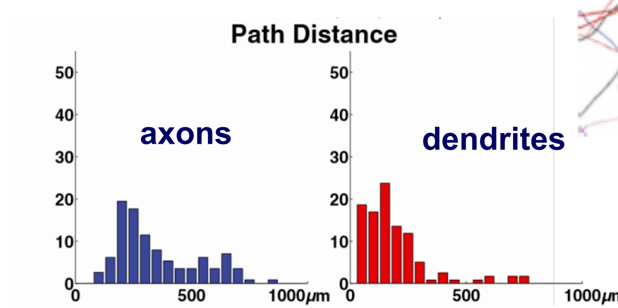


Important Properties (Neocortex)

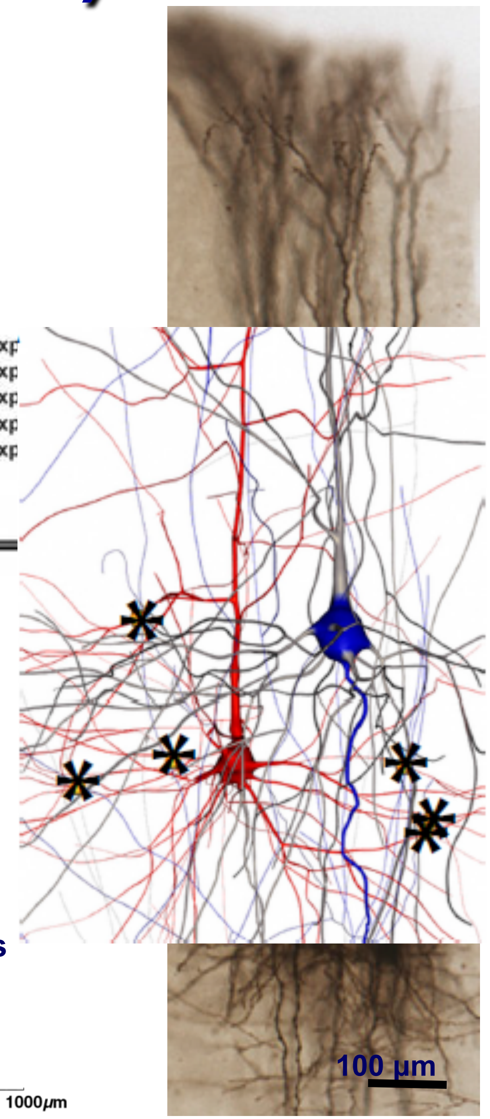
- Total neurons – 100 billion
- Synapses per neuron – 10 thousand
  - 90% or more are “silent” at any given time
- Neuron latency
  - 1’s of milliseconds
- “Transmission” delay
  - 1’s of milliseconds
  - Path lengths order (100s um) or more
  - Prop. delay order (100s um per ms)
- Multiple synapses per neuron pair
  - On the order of 10



from Bakkum et al. 2008



from Hill et al. 2012 – Markram group

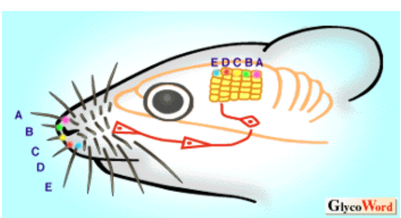


Spiking Neurons: Precision

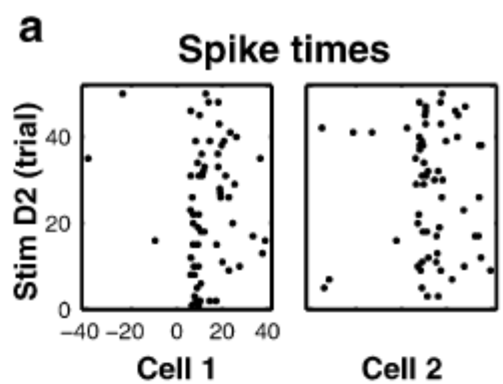
- Mainen and Sejnowski (1995)

  "stimuli with fluctuations resembling synaptic activity produced spike trains with timing reproducible to less than 1 millisecond"

- Petersen, Panzeri, and Diamond (2001)
  - Somatosensory cortex of rat (barrel columns)
  - Two cells in same column
  - Figure shows latency to first spike in response to whisker stimulation (.1 ms resolution)
  - There is some information content in immediately following spikes, but it is minor



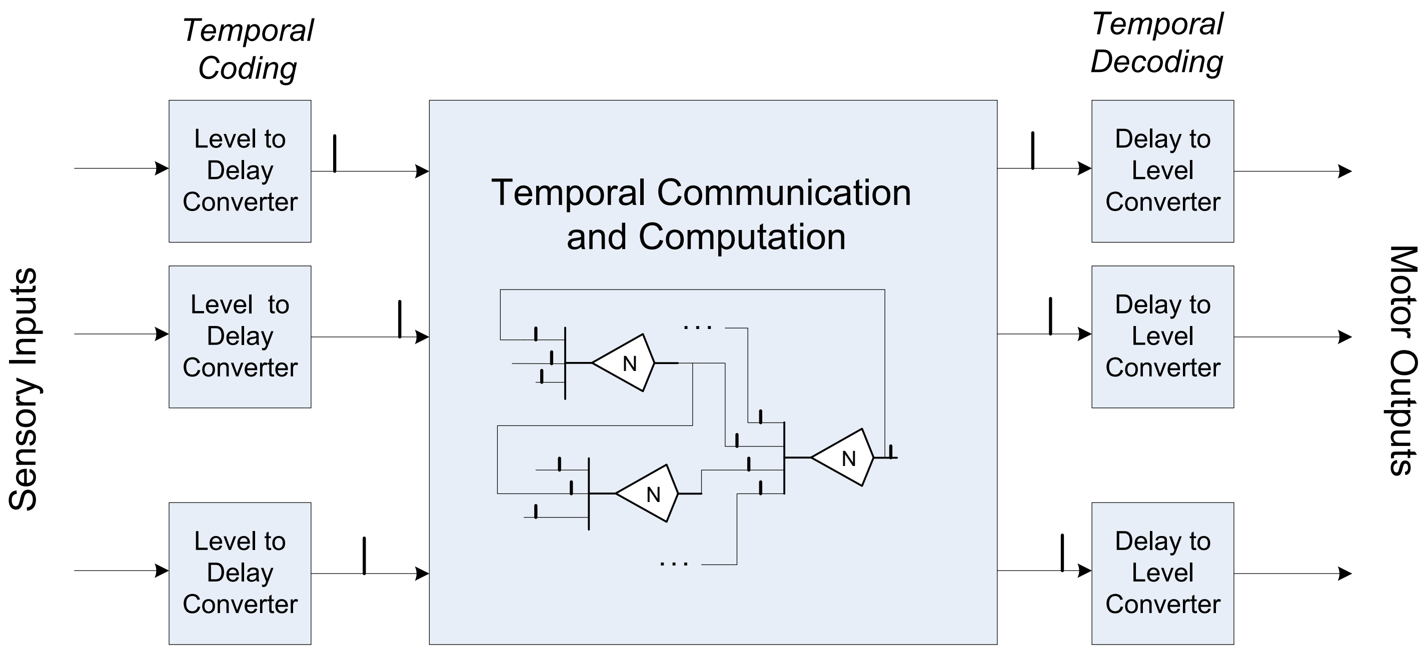
http://www.glycoforum.gr.jp/science/word/proteoglycan/PGA07E.html



units are .1 msec

Your Brain (Neocortex)

- A massive, asynchronous, locally-self-timed network built of unreliable components
- Information is encoded via precise spike timing relationships (“precise” = 1 decimal digit @ 1 msec)



Biological Plausibility

- Goal: Discover a computational paradigm such that all the significant features embodied in the paradigm are supported via biological experiment
  - But not everything in the biology must be represented in the paradigm!
    - A lot of it won't be
  - The way the brain *computes* is not the same as the way the brain *works*
- In the brain, the paradigm is obscured by mechanisms that implement a large asynchronous machine built with unreliable components
  - The vast majority of the elements and variety may not be part of the basic paradigm
  - Rather, they are there to provide a reliable substrate for computation
  - Brain scientists and computer scientists have little (or no) appreciation for this
    - "What is all the feedback for?"

Modeling Neurons

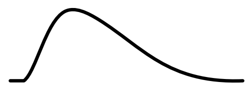
Neuron Model: Integration

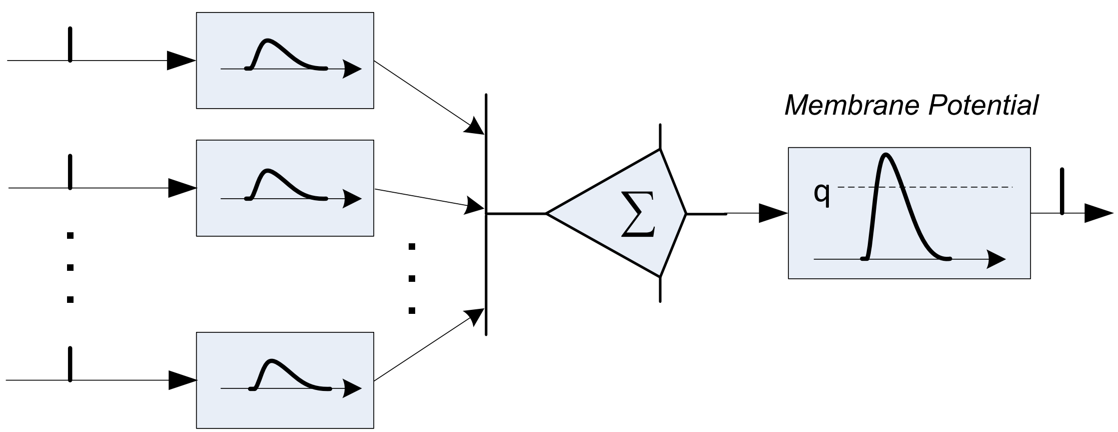
- Spike generates response function
  - Bi-exponential Excitatory Post Synaptic Potential (EPSP)
  - Responses are summed linearly
  - This models the neuron’s membrane (body) potential
- When potential exceeds threshold value (θ)
  - Generate output spike
  - Then reset potential to zero (not illustrated)
  - Wait for refractory time interval (not illustrated)

EPSP = K ($e^{-t/\tau M} - e^{-t/\tau E}$)

$e^{-t/\tau E}$: Synaptic “gate” closing

$e^{-t/\tau M}$: Membrane leakage

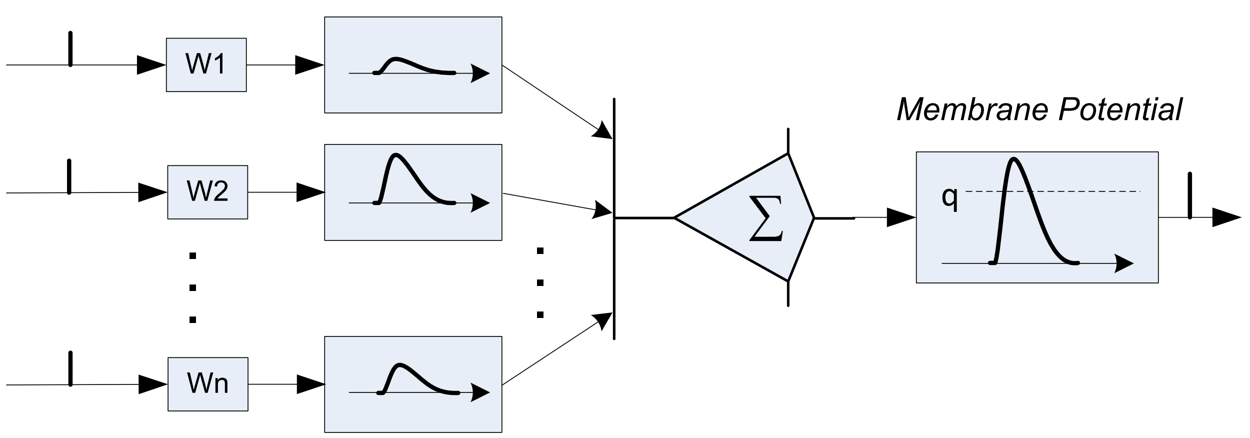




Neuron Model: Synapses

- Synapses have an associated efficacy or "weight"
  - The larger the weight, the higher the *EPSP's* amplitude
  - Typically modeled as a value between 0 and 1

EPSP from strong synapse **EPSP from weak synapse**

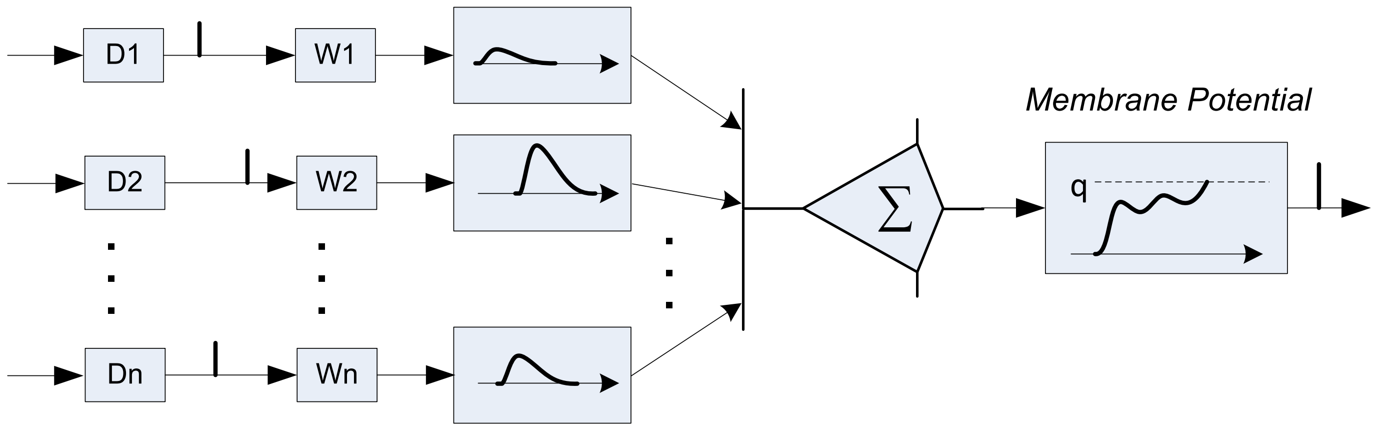


Neuron Model: Synaptic Plasticity

- Synapse weights are adaptable or *plastic*
  - This allows the customization of individual neurons
  - Achieved via *training*
- Spike Time Dependent Plasticity (STDP)
  - If an input spike closely precedes an output spike, the associated synapse is *strengthened*
  - If an input spike closely follows an output spike, the associated synapse is *weakened*
- Training
  - Present patterns of input spikes for the neuron to learn
  - Synapses adjust weights to those patterns
  - After training, the learned patterns (*as well as similar patterns*) will cause an output spike
- Training is localized, unsupervised, proceeds from inputs to outputs
  - All good features to have for very fast, very efficient training

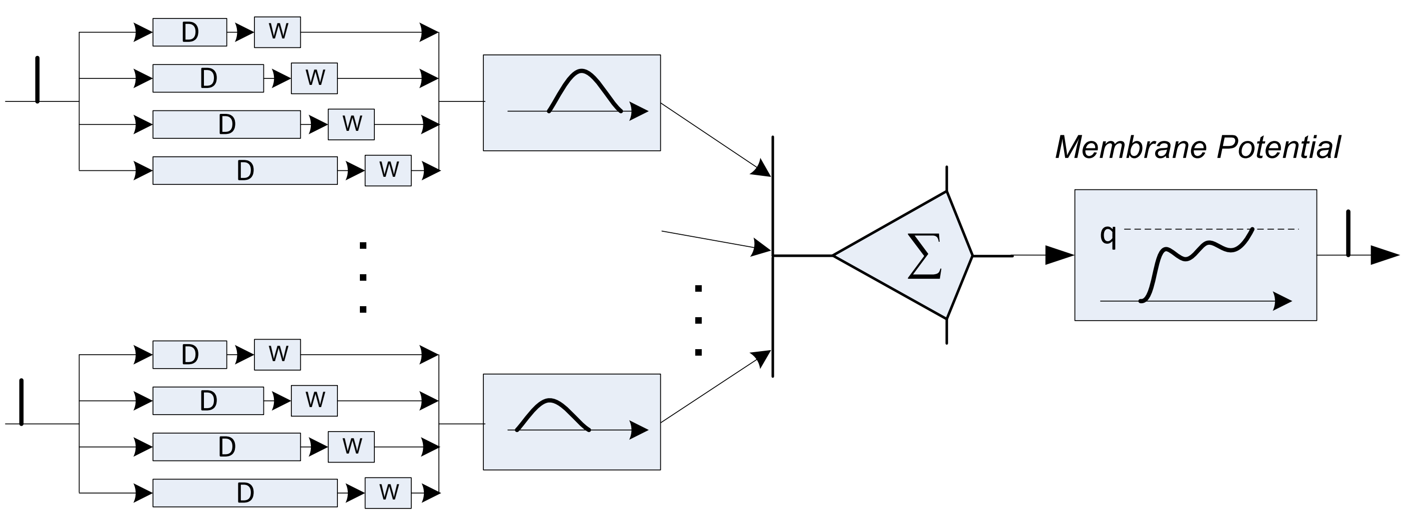
Neuron Model: Delays

- The inter-neuron delay is on the same order as the neuron's computational latency
  - Delays cannot be ignored
  - They are a basic computational component
  - A few theoreticians take this into consideration



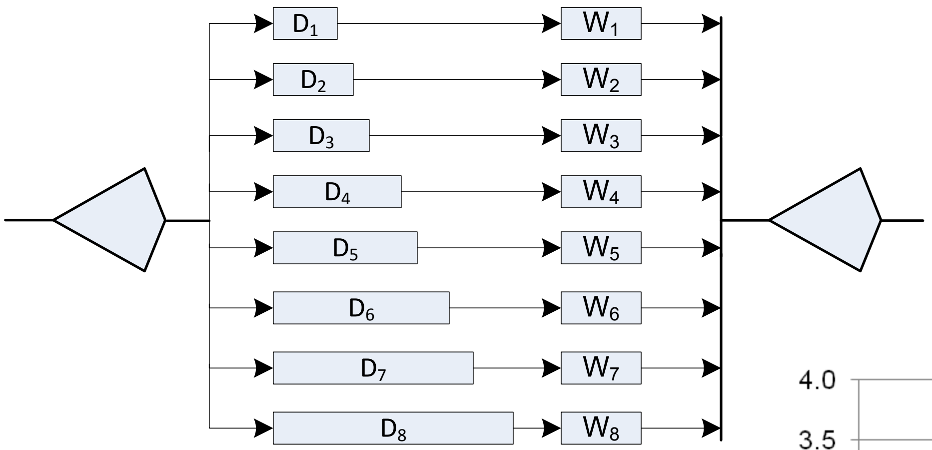
Neuron Model: Multisynapse Connections

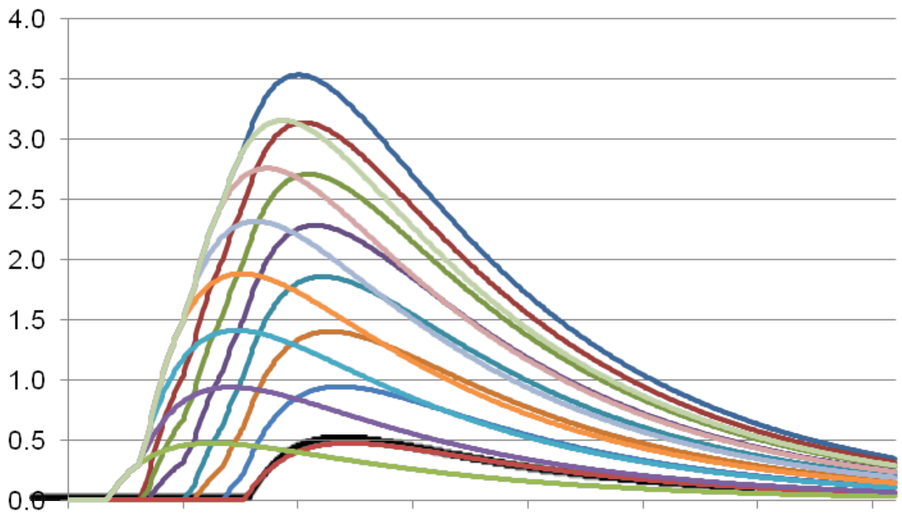
- A connection between two neurons is not just a single path
  - There may be several paths
  - Each with a different delay
- This is also a key part of the computational process
  - *Almost* entirely ignored by theoreticians
  - The range of weights defines a rich set of unary EPSP transformations



Multi-Path EPSPs

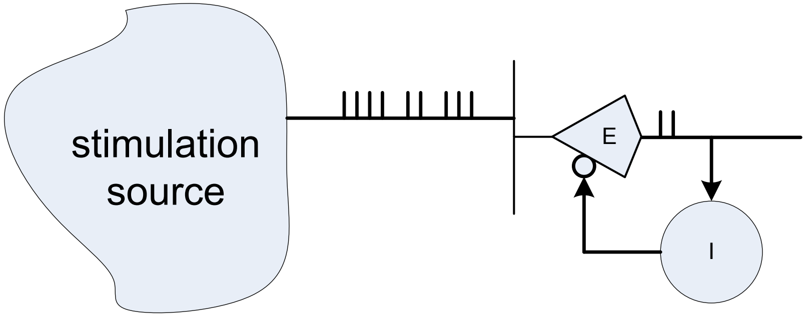
- Example: Eight paths w/ different delays
- Use some selected 0/1 weights to illustrate possibilities





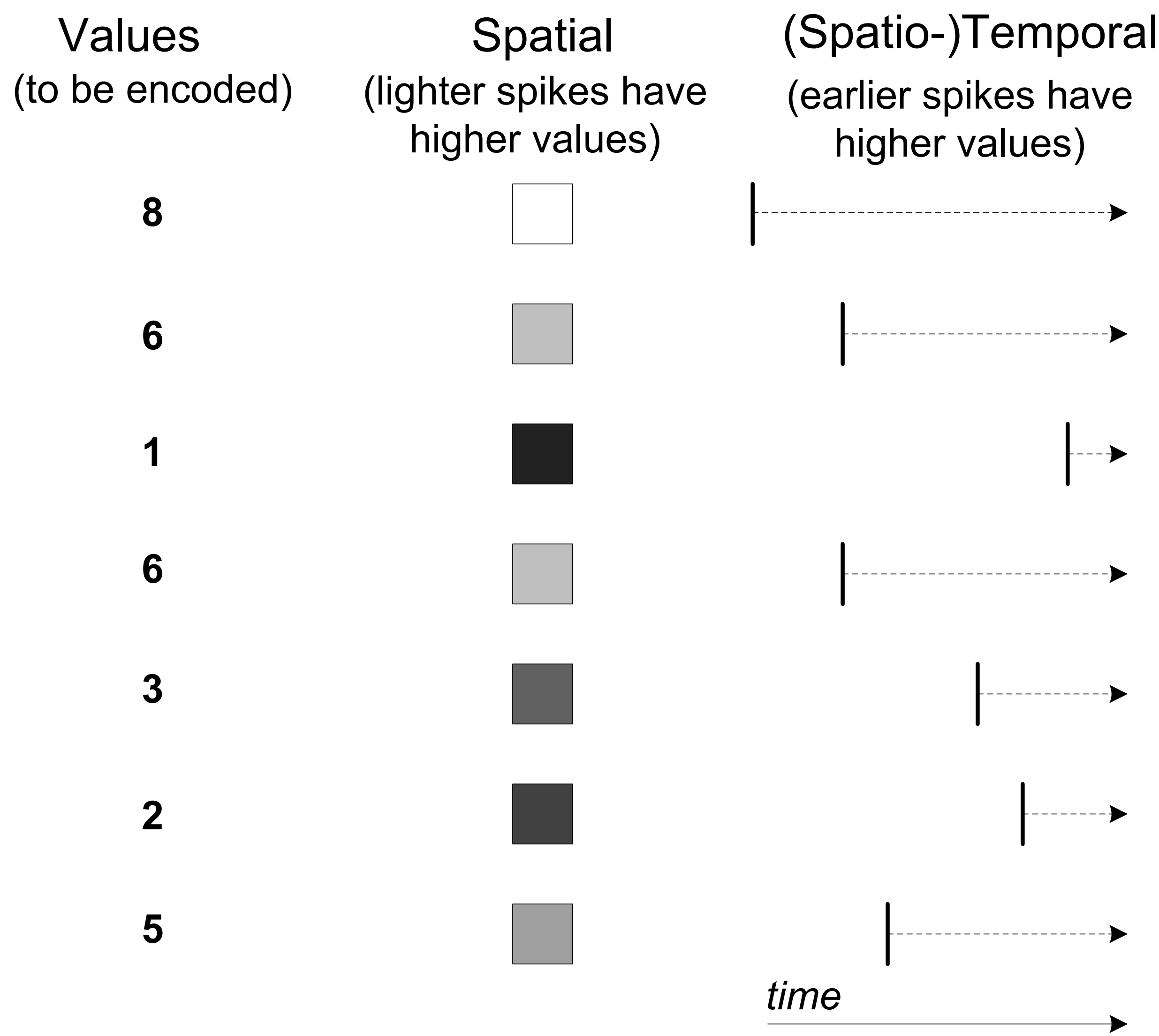
Role of Inhibition

- Inhibition is not computationally symmetric wrt excitation
- Inhibitory neurons:
  - Sharpen/tune
  - Provide localized moderating/throttling
    - Save energy
  - Allow dynamic adjustment of effective threshold
- Inhibitory neuron properties
  - Only 15-25 % of neurons
  - Outputs are typically used only locally (interneurons)
  - *Can be modeled as a population*

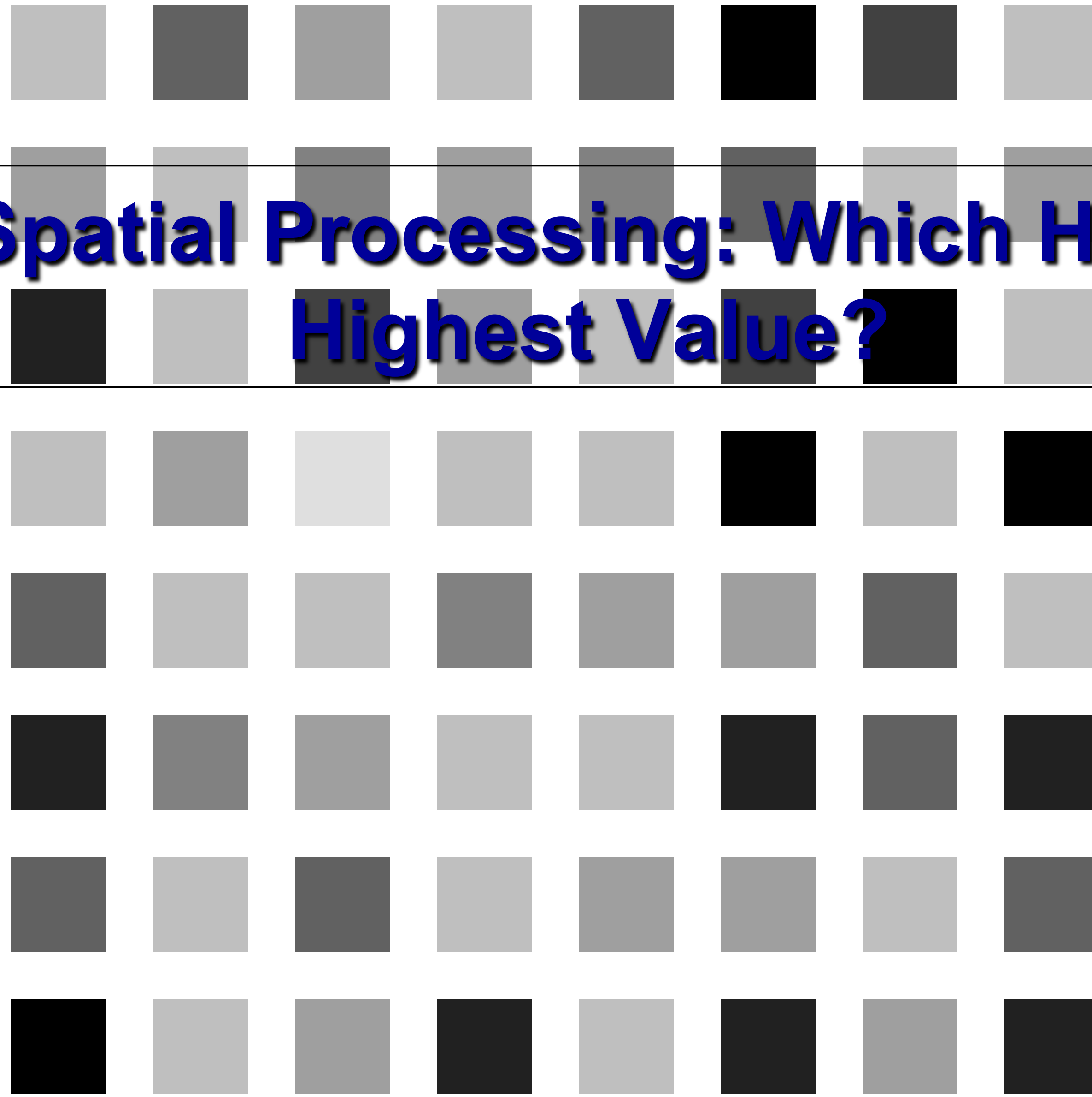


Temporal Computation

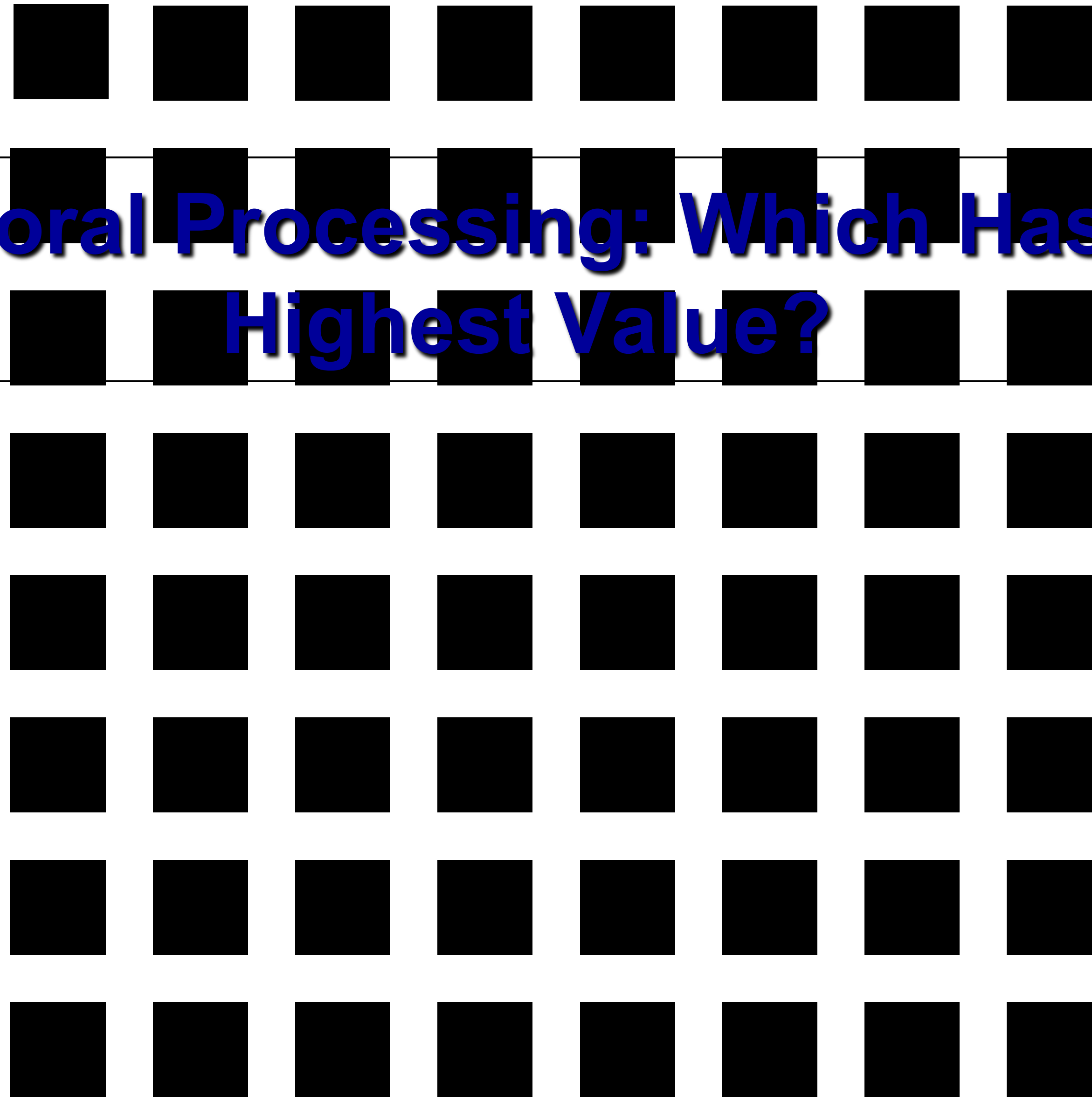
Temporal vs. Spatial Coding



Spatial Processing: Which Has Highest Value?

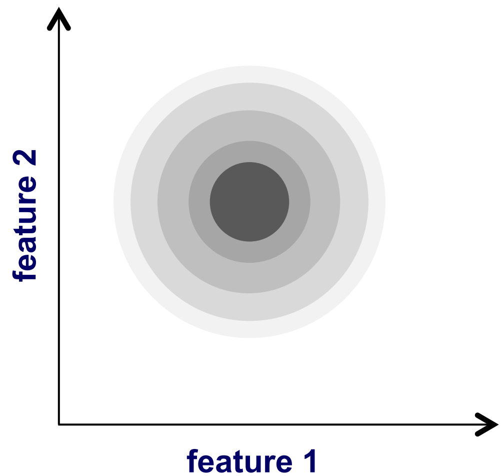


Temporal Processing: Which Has the Highest Value?



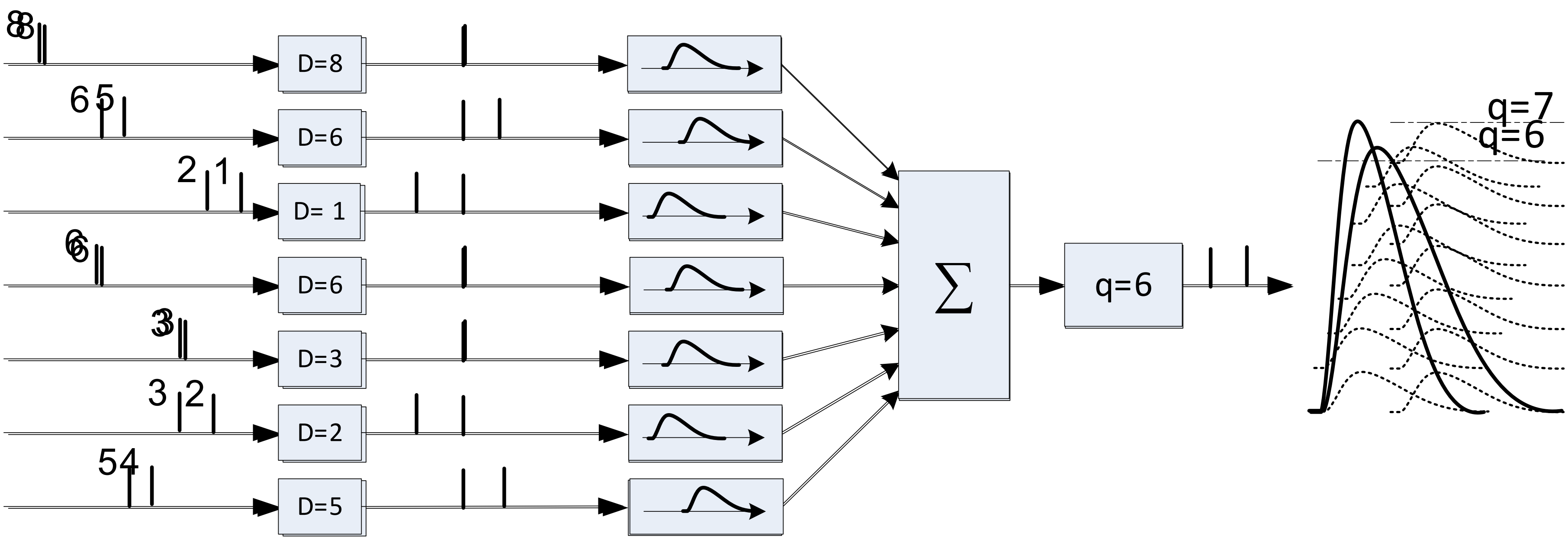
Example: Classification

- Class
  - A collection of similar patterns, based on a number of features or properties
  - Degrees of class membership may be quantified
- Train using some members of the class
- Then, detect (classify) input patterns that are *similar* to the training patterns
  - But which may never have been used during training
  - These form a class
- Temporal pattern: a sequence of spikes from multiple neurons
  - Example: a class defined by two features, or properties



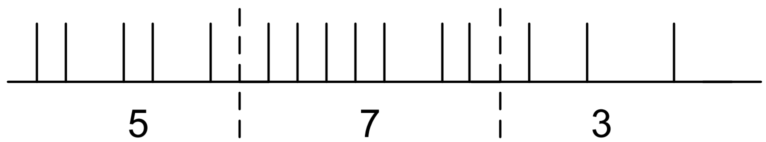
A Spiking Neuron is a Natural Classifier

- An example with seven quantified features (inputs)
  - For simplicity, assume unit weights
- Establish class "center" as point where all input spikes align in time
  - After input delays are applied
  - Note: Delays are computational components
- Adjust threshold level to establish limits of class



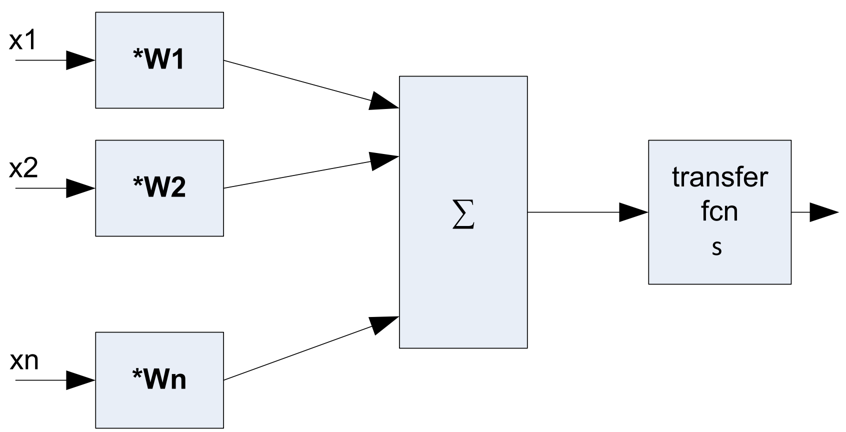
Conventional Artificial Neural Nets

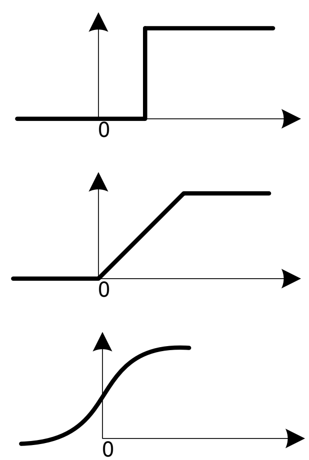
- Traditionally, information is carried in spike *rates*



- Encode rates as a range of values
- Operate on values with Perceptrons
  - Form sum of weighted inputs
  - Apply transfer function:

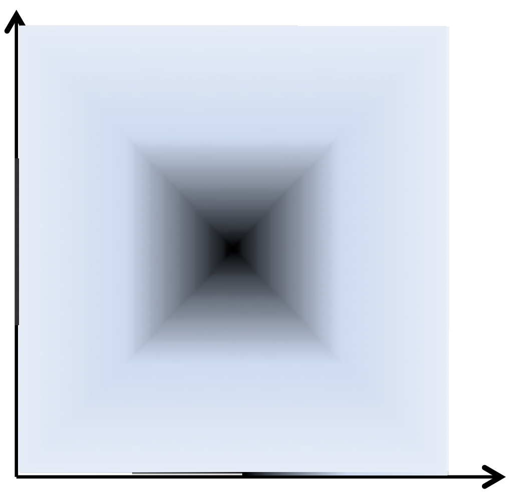
    threshold, sigmoid --$1/(1-e^{-x})$, *tanh* -- or saturating piecewise linear function

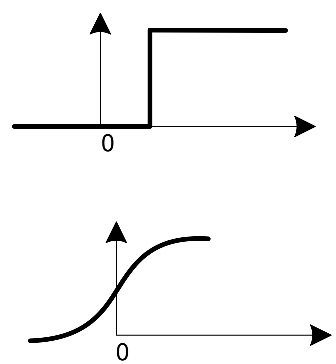




Example: Defining a Class

- In two-dimensional space
  - *Two perceptrons in 1st dimension define open convex region (threshold fcn)*
  - *Use sigmoid function to add gradation*
  - *Two perceptrons in second dimension define second open convex region*
  - *Adding sigmoid function and combining with first convex region yields approx. class*
- Requires O(N) perceptrons for N dimension
  - There can be hundreds of dimensions





Developing a Paradigm

Approach

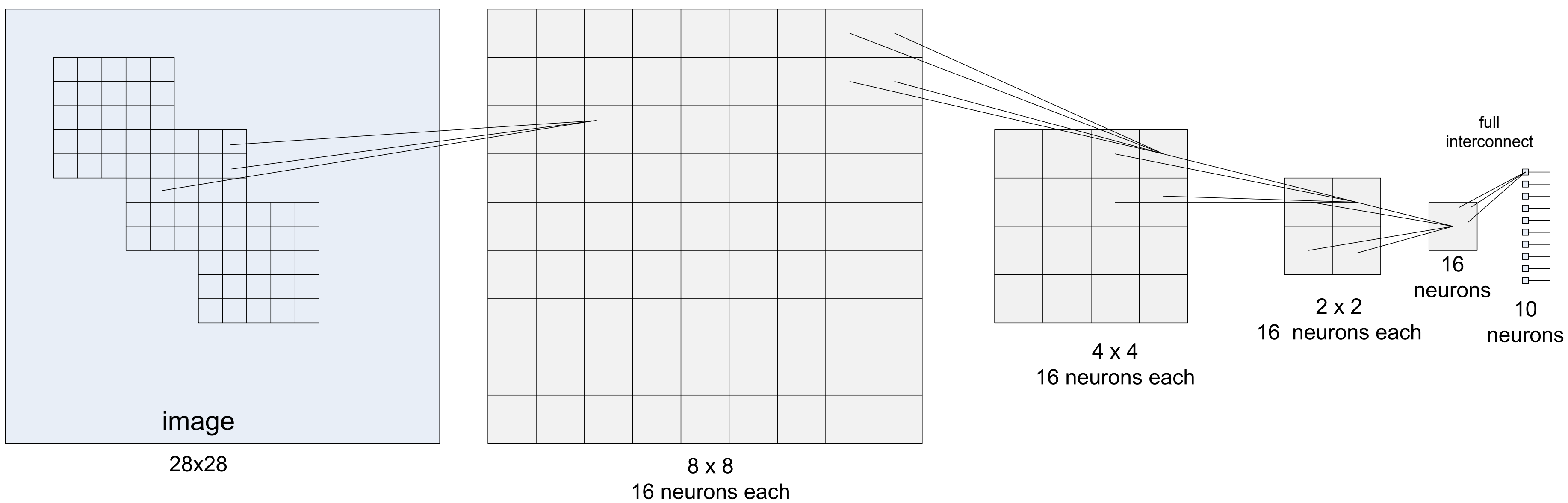
- Networks studied
  - Feedforward
  - Separate training and application
  - Single volley of spikes at a time
- Standard machine learning benchmark(s)
  - This could morph into a machine learning project
    - With a far more efficient training method
- Network Simulation Model
  - Written in Matlab
  - Frontend simulates synaptic weights, delays, and plasticity
  - Backend simulates neuron body
- Abstract Functional Model
  - Written in Matlab
  - Abstracts spike volleys
  - Uses direct function evaluation
    - Avoids time step simulation
- Run on a laptop

MNIST Benchmark

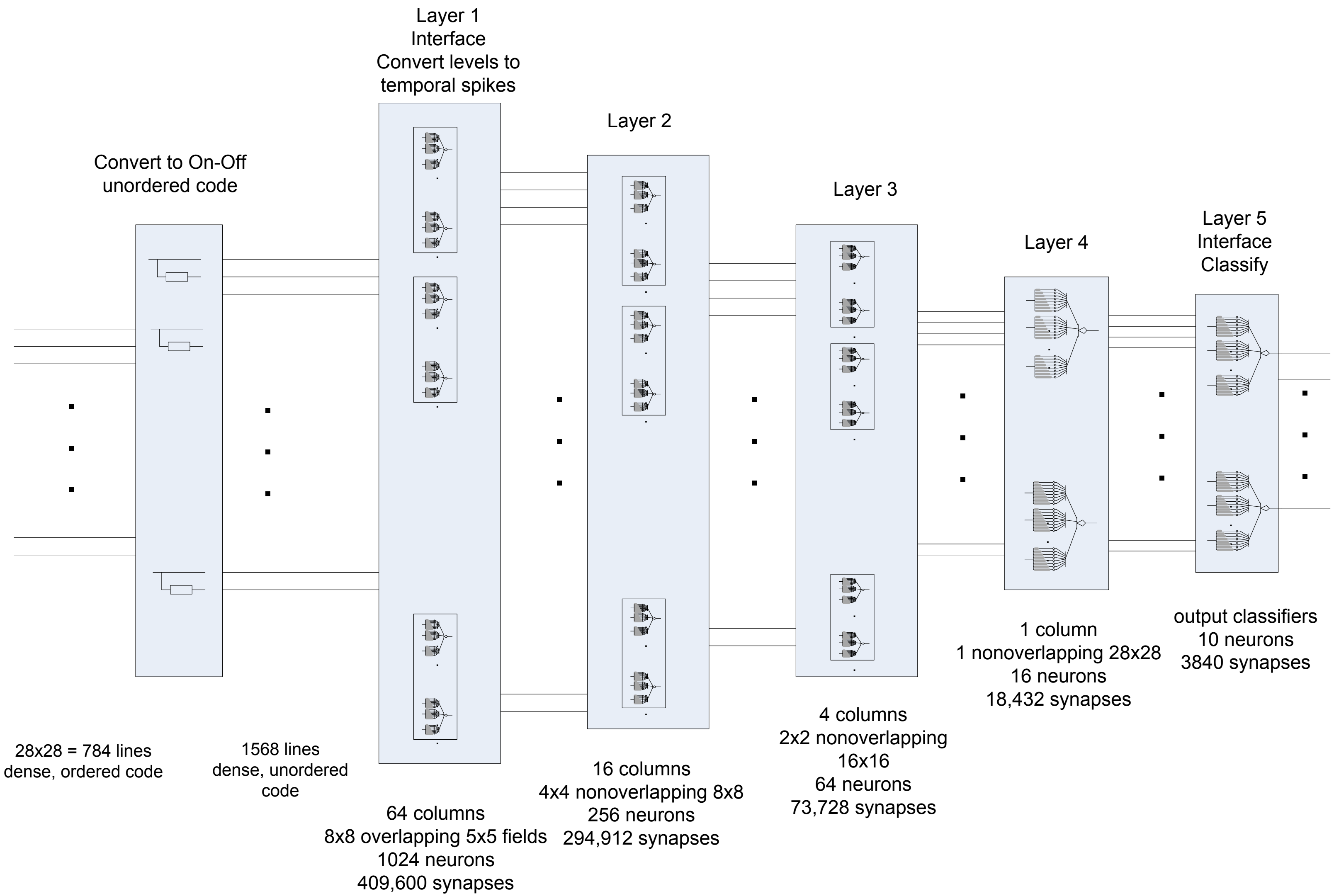
- What it is:
  - Tens of thousands of 28 x 28 grayscale images of written numerals 0-9
  - Pixelized w/ interpolation from B&W images
- Accuracy
  - The best machine learning implementations have an accuracy of about 99.5%
- Goal: similar accuracy to best machine learning implementations
  - Training time several orders of magnitude faster

Overview

- Use hierarchy as in deep learning approaches
  - First level: processing; conversion to temporal, sparse unordered coding
  - Second level: 64 to 16 processing; 8x8 columns, 16 neurons each
  - Third level : 16 to 4 processing; 4x4 columns, 16 neurons each
  - Fourth level: 4 to 1 processing; 2x2 columns, 16 neurons each
  - Final level: 10 classifier outputs operating on single 16 neuron column



System Architecture



- Layer 1: A single column processes a 5x5 receptive field
  - There are 8x8 = 64 overlapping RFs in all
- Layer 2: A single column processes four of the overlapping 5 x 5 fields
  - Covers an 8x8 in the original images

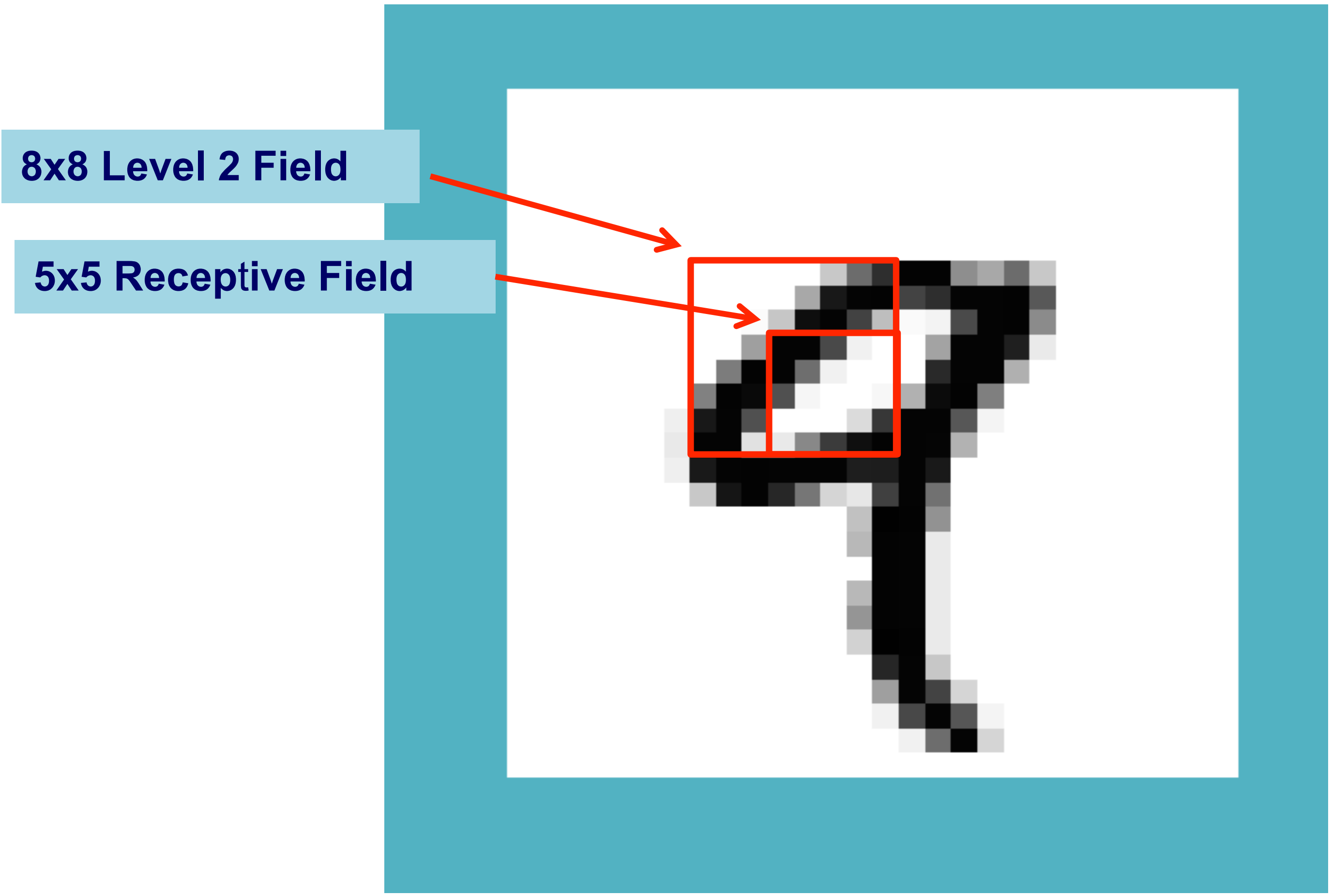
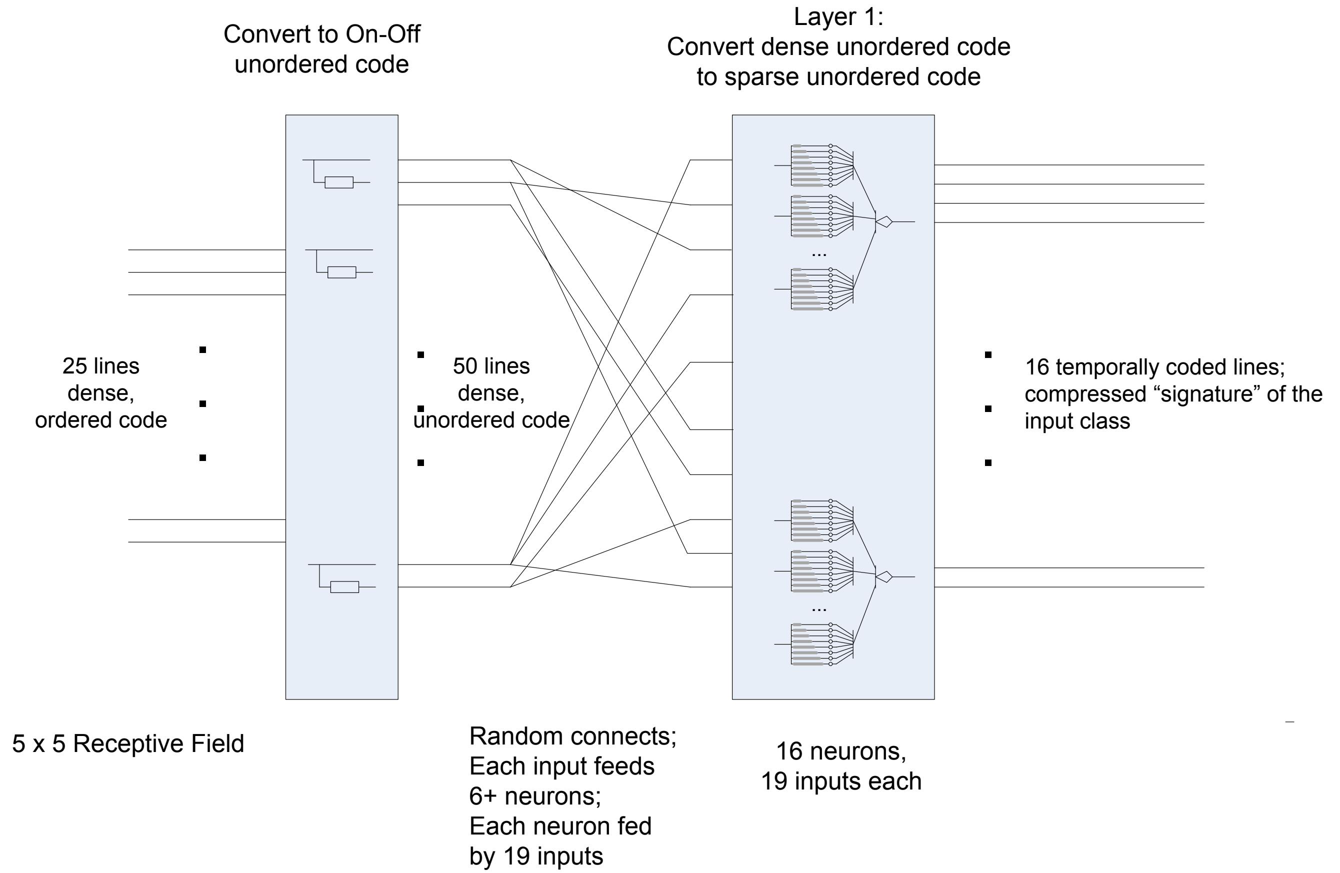


Image #5 from training set

Layer 1 Column Architecture

- One column per 5x5 Receptive Field
- 16 neurons
- All input weights fixed at 11110000
  - Discovered to work well after much experimentation

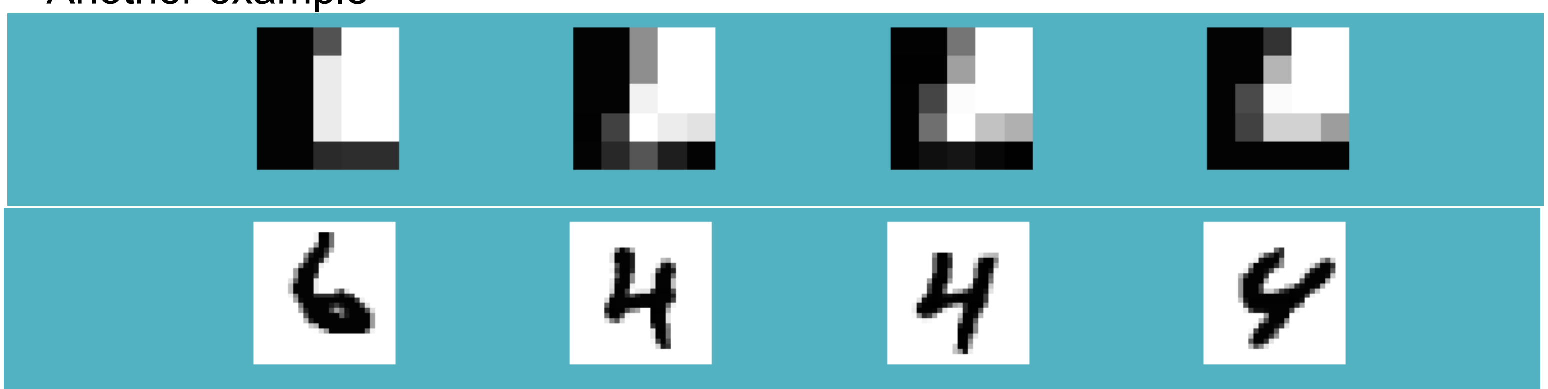


Layer 1 Example Classes

- Use 10,000 train and test images
- Classes are identified by first six ordered spikes (only)
  - This ignores specific temporal information which can further distinguish images
- Example of images in the same class:

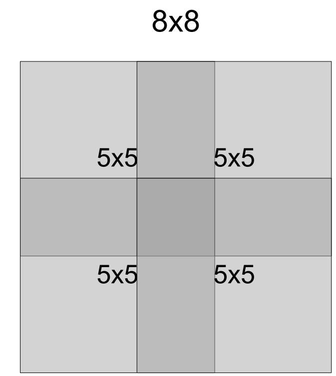
- Full images:

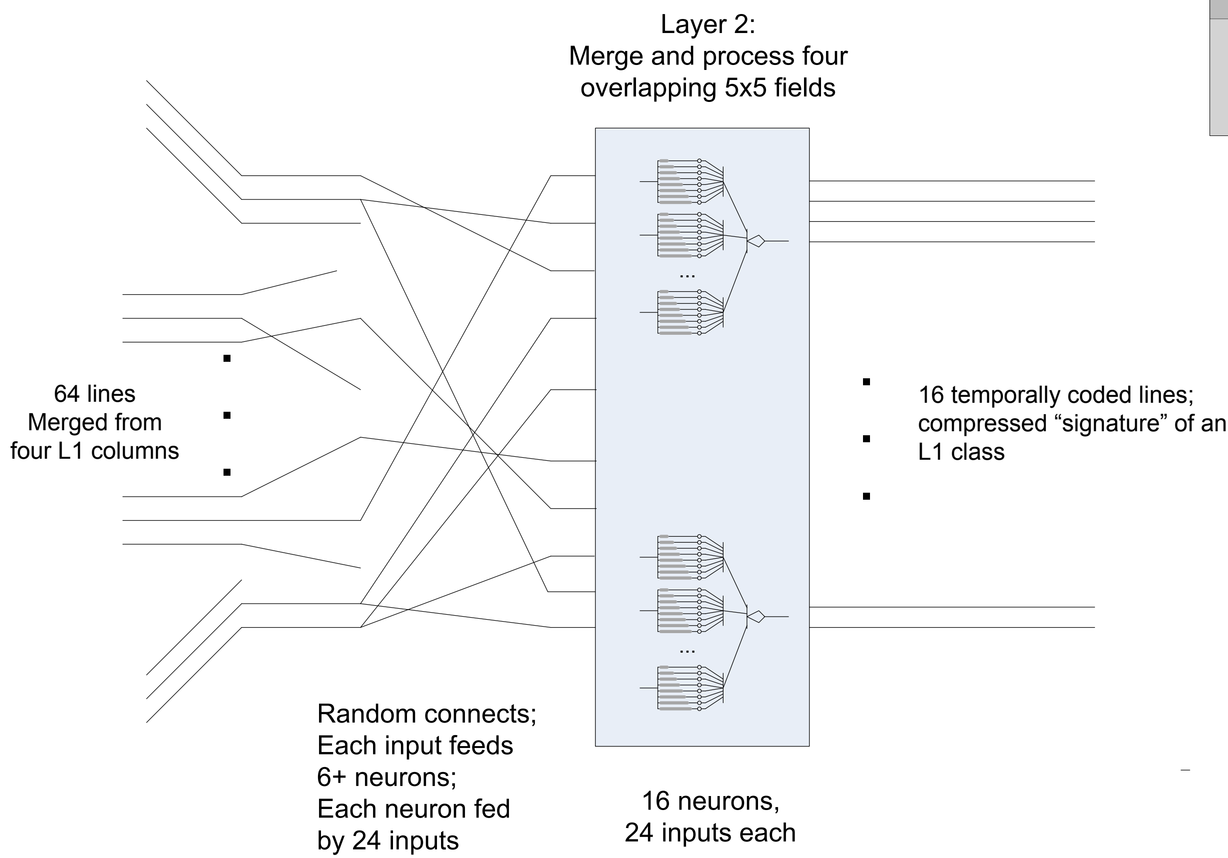
- Another example



Layer 2 Column Architecture

- Outputs of 4 Layer1 columns are merged
- These are inputs to Layer 2 column





Layer 2 Example Classes

- At layer 2
  - These are four overlapped 5 x 5s which form an 8 x 8
- Example of images with same first six spikes in same order:

- Full size images

- Another example: same first four spikes in same order

Conclusions

Conclusions (Opinions)

- Mechanisms for keeping a large asynchronous machine afloat are much more prominent than the paradigm, itself
  - Which may explain why the paradigm has been so hard to find
- Delays are a critical computing component
  - Communication and computation are temporal
  - Which may explain why the paradigm has been so hard to find
- Very efficient training is a key part of the paradigm
  - And this sets it (far) apart from conventional machine learning
- There are many opportunities for researchers with a good engineering sense
  - Engineering practicality and biological plausibility are first cousins

Background (Where to Look)

- The following are *example* sources of information that I have found very useful
  - There are many others
- All have a very strong interest in discovering the brain’s computational paradigm
- They all seem to be asking the right questions

Biology

Just about any text, Scholarpedia, or Wikipedia

Experimental work

Markram group

Hill,, al. et, Statistical connectivity provides a sufficient foundation for specific functional connectivity in neocortical neural microcircuits, *Proc. of the National Academy of Sciences* (2012)

Modeling

Gerstner group

Morrison, Abigail, Markus Diesmann, and Wulfram Gerstner. "Phenomenological models of synaptic plasticity based on spike timing." *Biological cybernetics* 98, no. 6 (2008): 459-478.

Theory

Maass

Maass, Wolfgang, Networks of spiking neurons: the third generation of neural network models, *Neural networks* 10.9 (1997): 1659-1671

Neuron level design

Bohte

Bohte, Sander M., Han La Poutré, and Joost N. Kok. "Unsupervised clustering with spiking neurons by sparse temporal coding and multilayer RBF networks," *IEEE Transactions on Neural Networks*, 13, no. 2 (2002): 426-435.

Column level design (also spike coding)

Thorpe group

Guyonneau, Rudy, Rufin VanRullen, and Simon J. Thorpe. "Temporal codes and sparse representations: A key to understanding rapid processing in the visual system." *Journal of Physiology-Paris* 98 (2004): 487-497.