

# Silicon Learning Machines

Gert Cauwenberghs

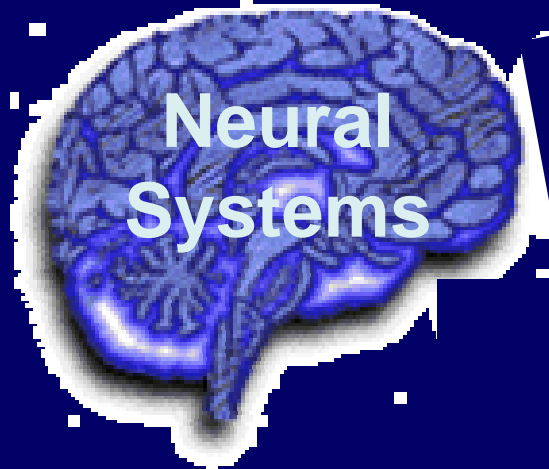
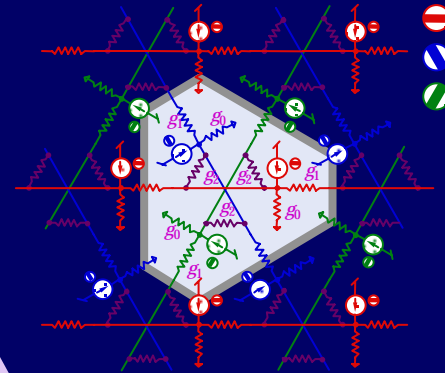
*gert@ucsd.edu*

# Neuromorphic Engineering

*"in silico" neural systems design*

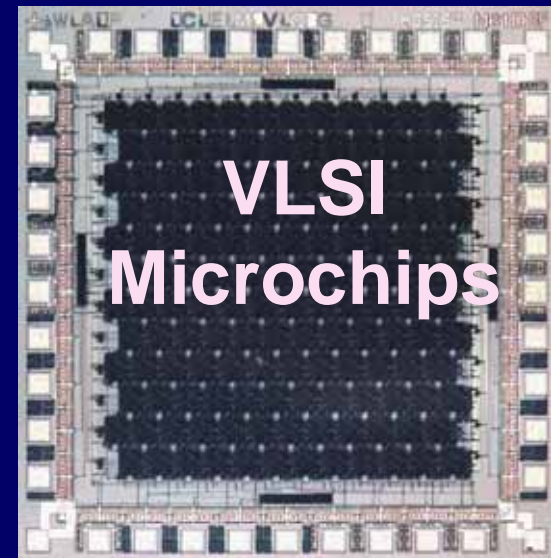


**Neuromorphic  
Engineering**



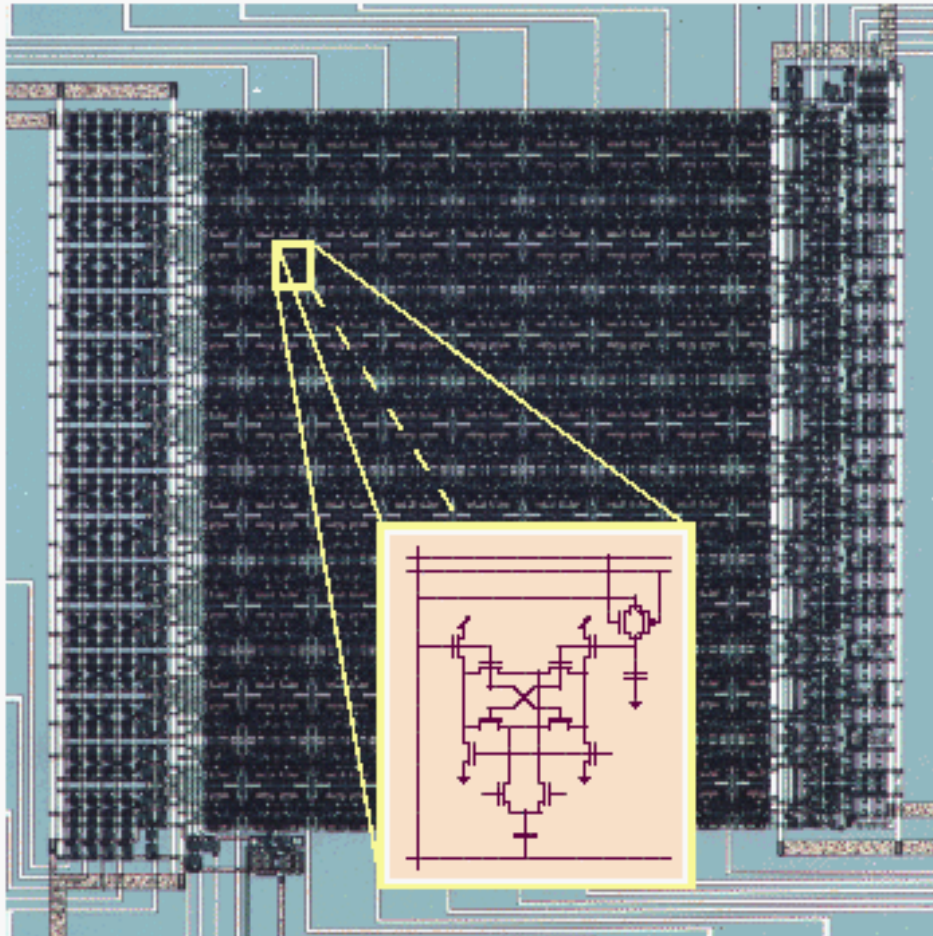
**Neural  
Systems**

**Learning  
&  
Adaptation**



**VLSI  
Microchips**

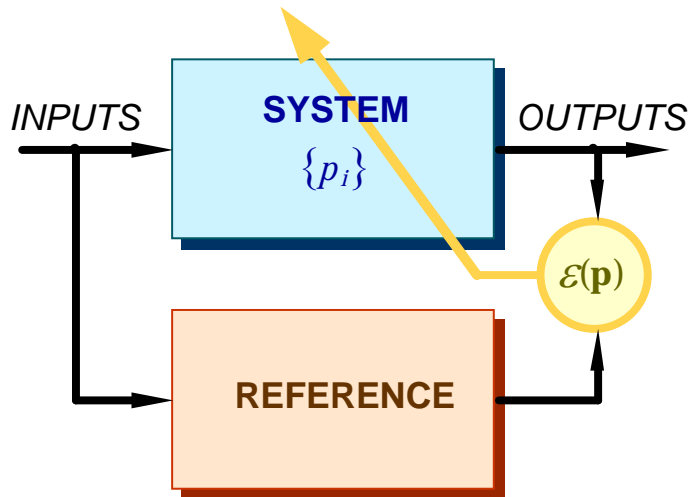
# Large-Scale Mixed-Signal Sensory Computation



*Example: VLSI Analog-to-digital vector quantizer  
(Cauwenberghs and Pedroni, 1997)*

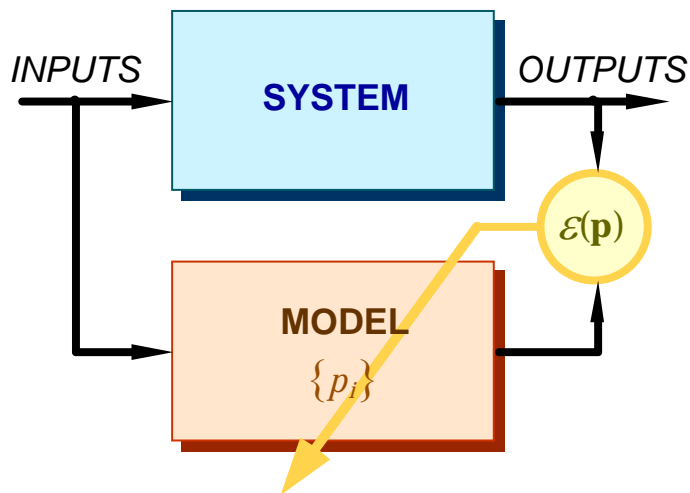
- **Massive Parallelism**
  - distributed representation
  - local memory and adaptation
  - analog sensory interface
  - physical computation
  - analog accumulation on single wire
- **Scalable**
  - silicon area and power scale linearly with throughput
- **Highly Efficient**
  - factor 100 to 10,000 less energy/operation than DSP
- **Limited Precision**
  - analog mismatch and nonlinearity (WYDINWYG)
  - fix: adaptation in redundancy

# Learning on Silicon



## Adaptation:

- necessary for robust performance under variable conditions and in unpredictable environments
- also compensates for imprecision in analog computation
- avoids ad-hoc programming, tuning, and manual parameter adjustment



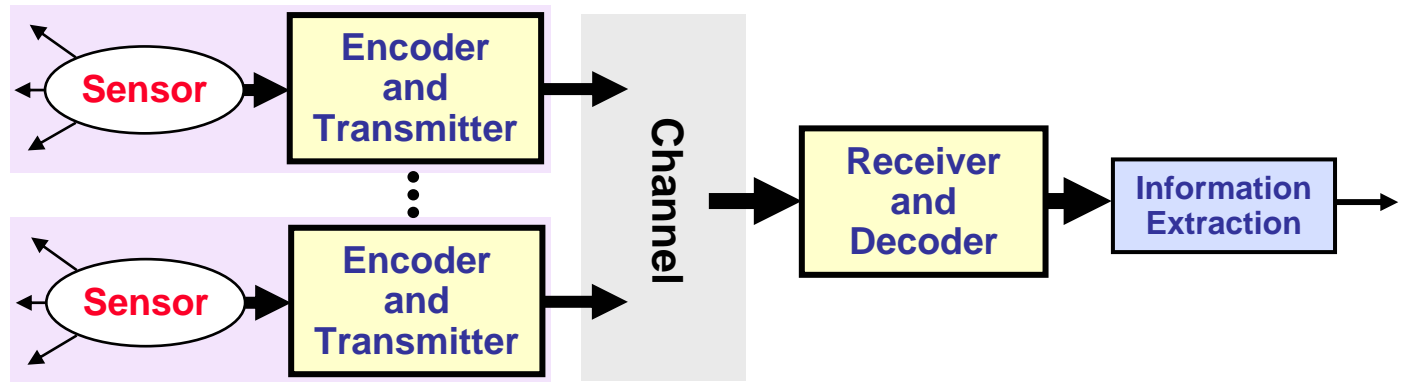
## Learning:

- generalization of output to previously unknown, although similar, stimuli
- system identification to extract relevant environmental parameters

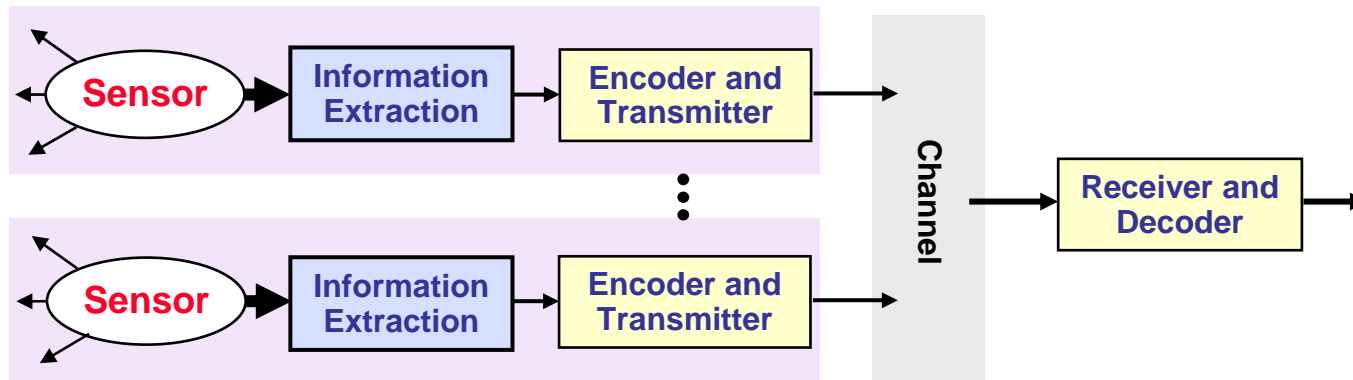
Cauwenberghs & Bayoumi, Eds., *Learning on Silicon*, Kluwer 1999.

# Integrated “Smart” Sensors

- **Sensor networks:**



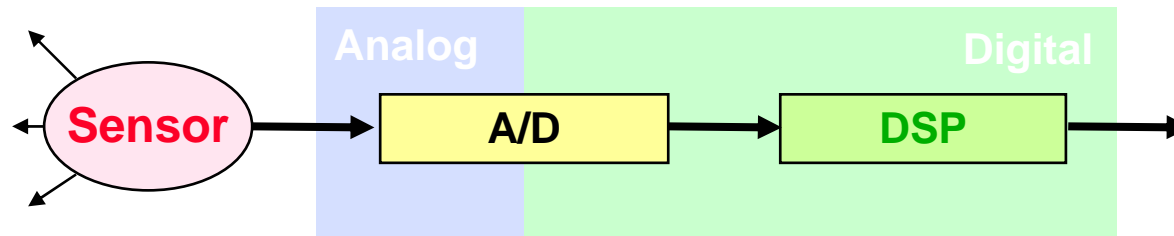
- **Sensor Integration and Distributed Intelligence:**



- reduced bandwidth requirements
- reduced power dissipation and form factor
- trade-off between precision and complexity/power of integrated processing

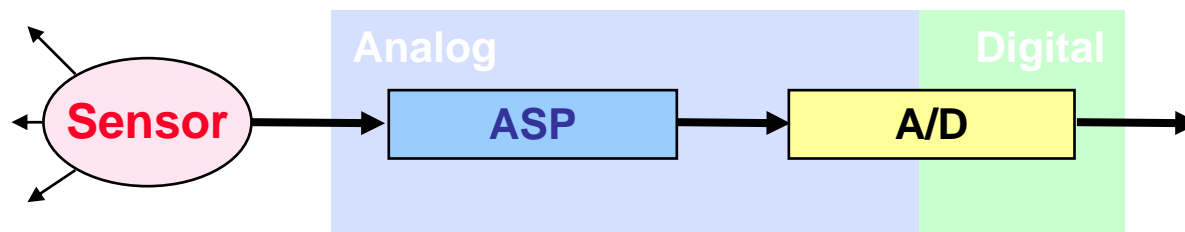
# Pushing the Analog-Digital Boundary

- **Digital Sensory Processing:**



- General-purpose
- High precision (limited by A/D)

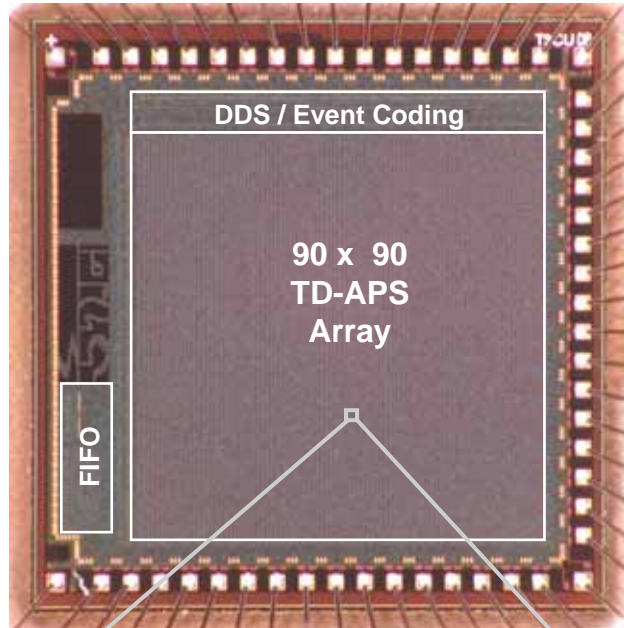
- **Analog and Mixed-Signal Sensory Processing:**



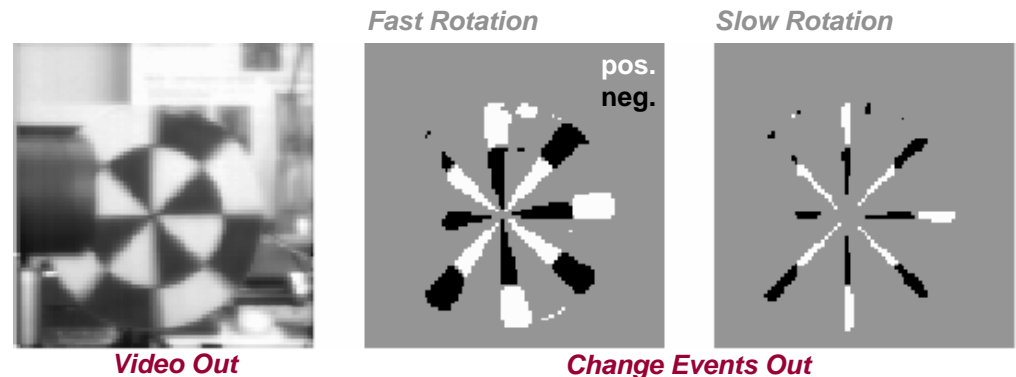
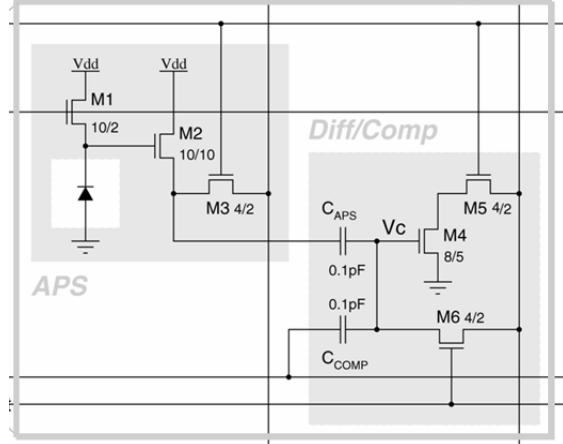
- “Smart” A/D
- Low power
- Low complexity

# Example: Change Threshold Detection APS CMOS Imager

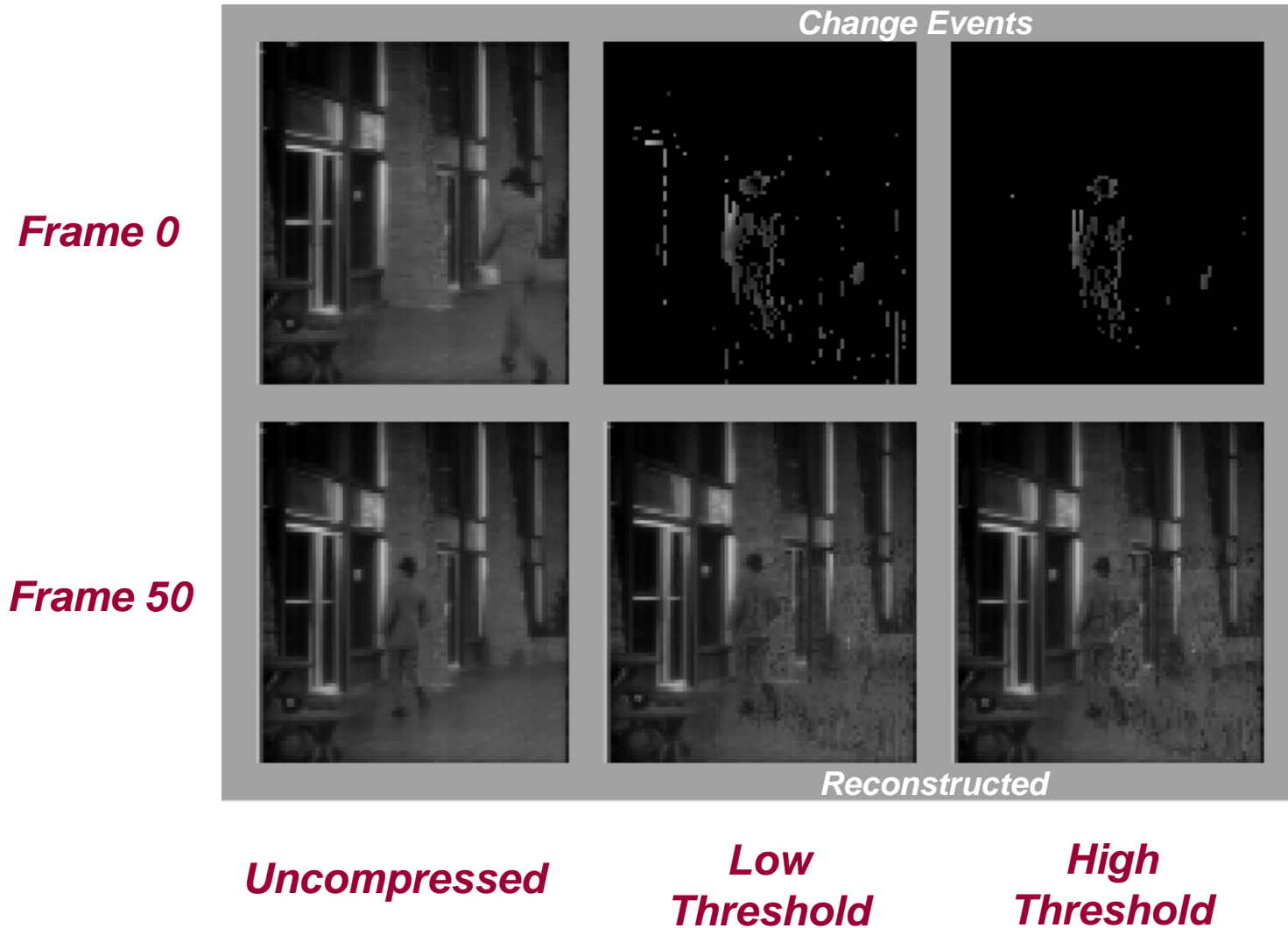
*Mallik, Clapp, Choi, Cauwenberghs and Etienne-Cummings (ISSCC'2005)*



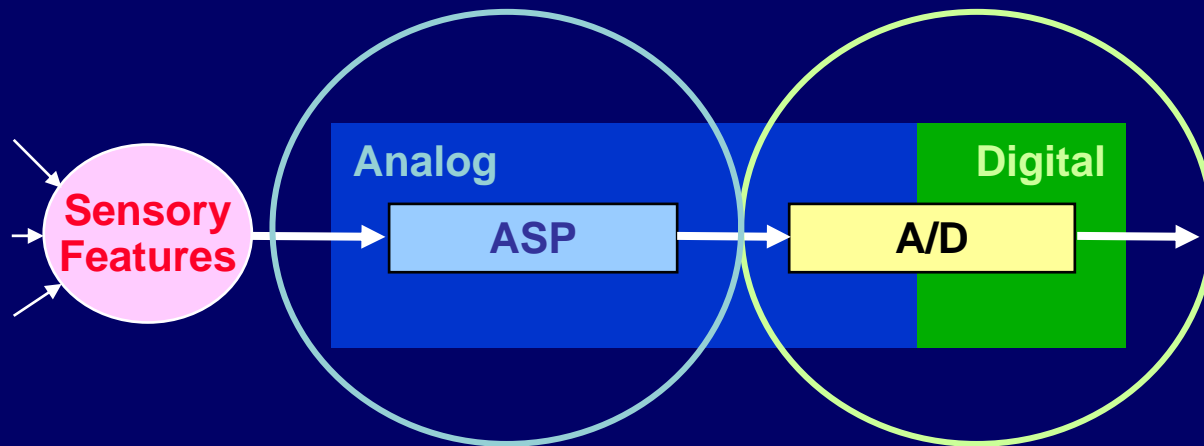
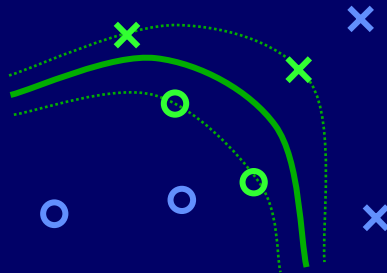
- Event-driven video compression
  - Address, polarity and intensity of change detection events are asynchronously communicated through FIFO
- Change detection and threshold encoding on the focal plane
  - $25\mu\text{m} \times 25\mu\text{m}$ , nMOS only pixel
  - 17% fill factor
  - 0.5% FPN
- 3mm X 3mm in  $0.5\mu\text{m}$  2P3M CMOS
- 4.3mW power at 3V and 30fps



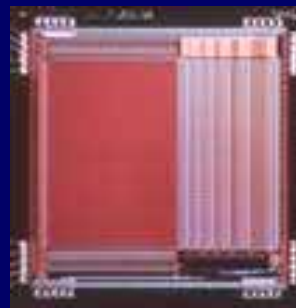
# Change Detection APS: Compression and Reconstruction



# SVM Pattern Recognition



**Large-Margin Kernel  
Regression**

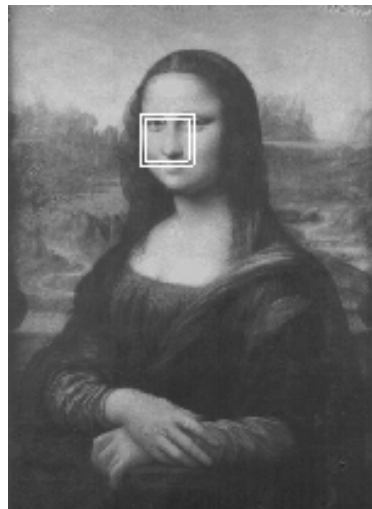
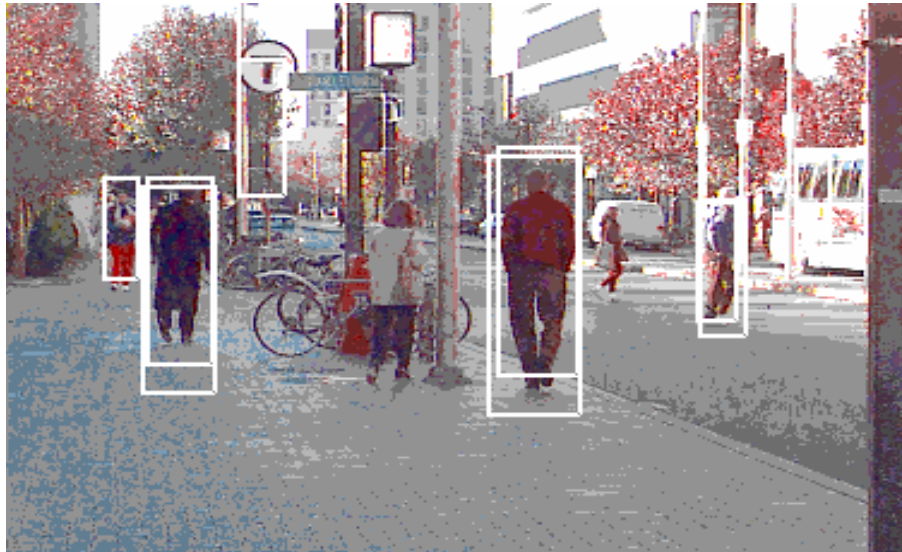


**Class Identification**

*Kerneltron:*  
massively parallel  
support vector  
"machine" in silicon  
(ESSCIRC'2002)

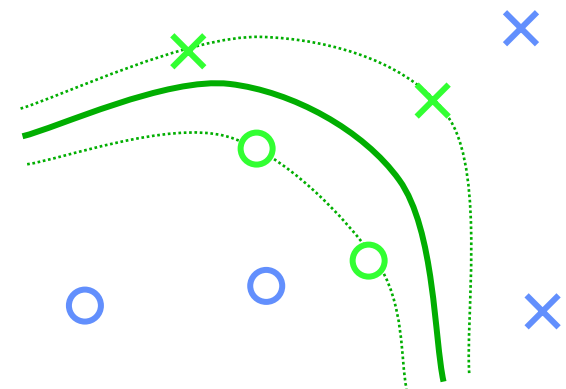
# Trainable Modular Vision Systems: The SVM Approach

Papageorgiou, Oren, Osuna and Poggio, 1998



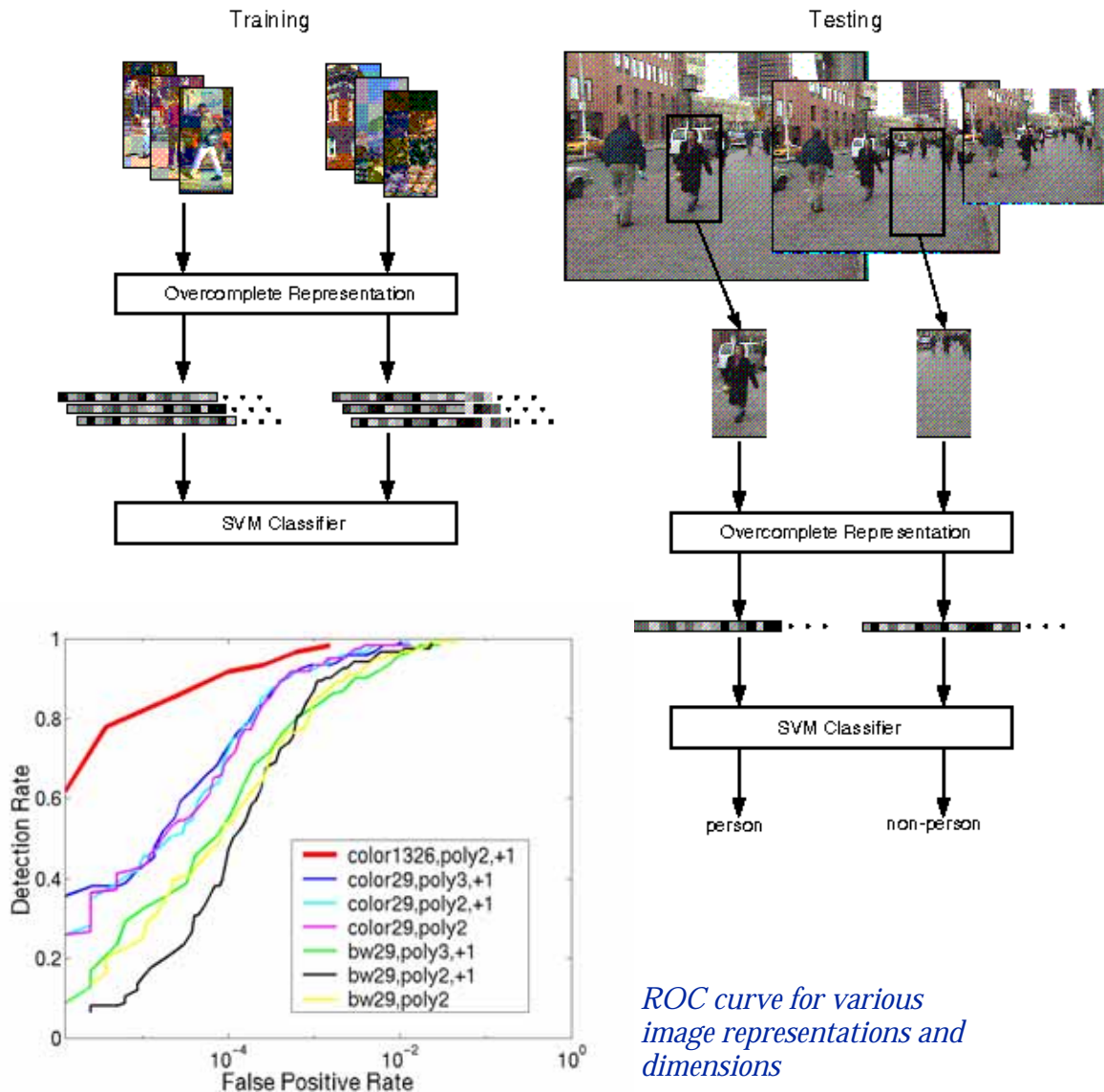
*SVM classification for pedestrian and face object detection*

- Strong mathematical foundations in *Statistical Learning Theory* (Vapnik, 1995)
- The training process selects a small fraction of prototype *support vectors* from the data set, located at the *margin* on both sides of the classification boundary (e.g., barely faces vs. barely non-faces)



# Trainable Modular Vision Systems: The SVM Approach

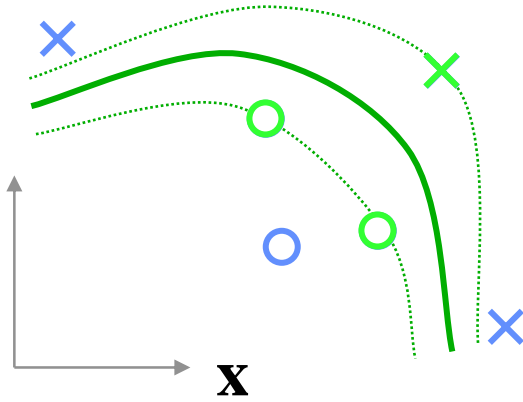
Papageorgiou, Oren, Osuna and Poggio, 1998



- The number of support vectors, in relation to the number of training samples and the vector dimension, determine the generalization performance
- Both training and run-time performance are severely limited by the computational complexity of evaluating kernel functions

# Kernels and Support Vector Machines

Mercer, 1909; Aizerman et al., 1964  
Boser, Guyon and Vapnik, 1992

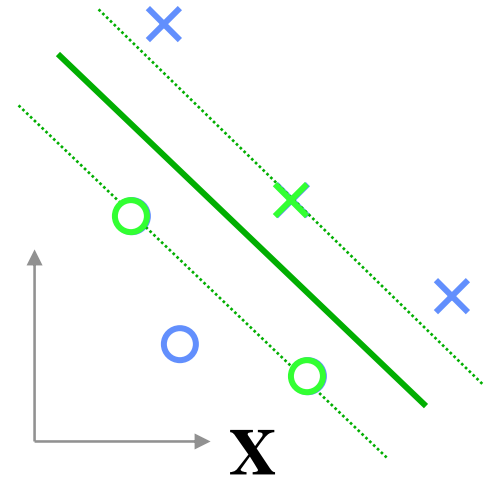


$\Phi(\cdot)$



$$\mathbf{X}_i = \Phi(\mathbf{x}_i)$$

$$\mathbf{X} = \Phi(\mathbf{x})$$



$$\mathbf{X}_i \cdot \mathbf{X} = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x})$$



$$y = \text{sign}\left(\sum_{i \in S} \alpha_i y_i \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}) + b\right)$$

$$y = \text{sign}\left(\sum_{i \in S} \alpha_i y_i \mathbf{X}_i \cdot \mathbf{X} + b\right)$$

$K(\cdot, \cdot)$



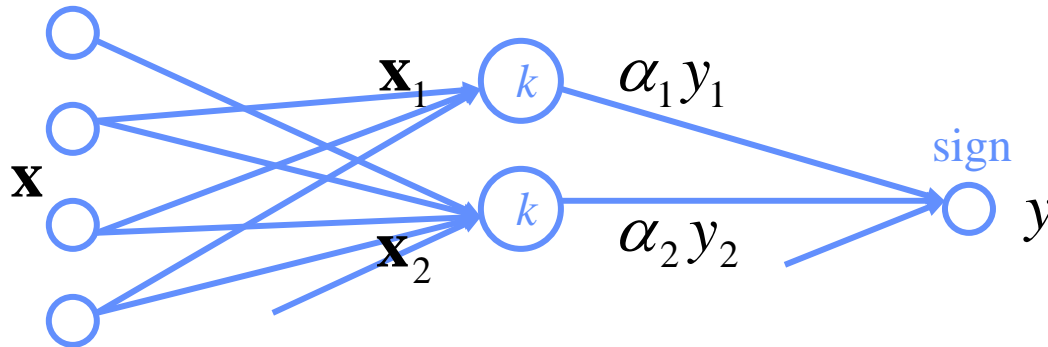
$$\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}) = K(\mathbf{x}_i, \mathbf{x})$$

Mercer's Condition

$$y = \text{sign}\left(\sum_{i \in S} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b\right)$$

# Kernel Machines

$$y = \text{sign}\left(\sum_{i \in S} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b\right)$$



- Gaussian (Radial Basis Function Networks)

$$K(\mathbf{x}_i, \mathbf{x}) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}\|^2}{2\sigma^2}\right) \propto \exp\left(\frac{\mathbf{x}_i \cdot \mathbf{x}}{\sigma^2}\right)$$

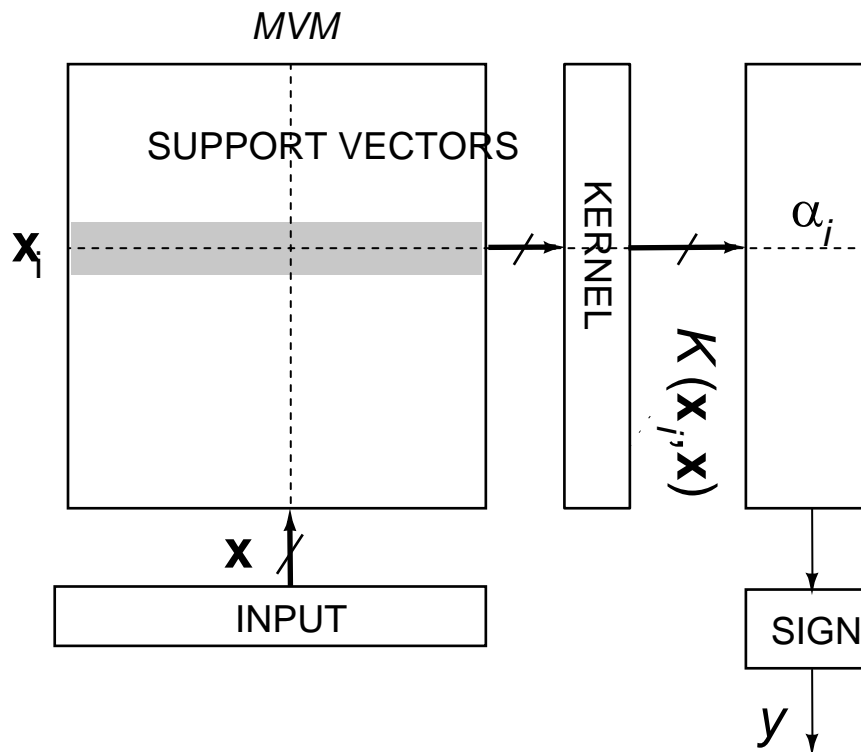
- Sigmoid (Two-Layer Perceptron)

$$K(\mathbf{x}_i, \mathbf{x}) = \tanh(L + \mathbf{x}_i \cdot \mathbf{x}) \quad \text{only for certain } L$$

- Polynomial (Splines etc.)

$$K(\mathbf{x}_i, \mathbf{x}) = (1 + \mathbf{x}_i \cdot \mathbf{x})^\nu$$

# Parallel SVM Architecture

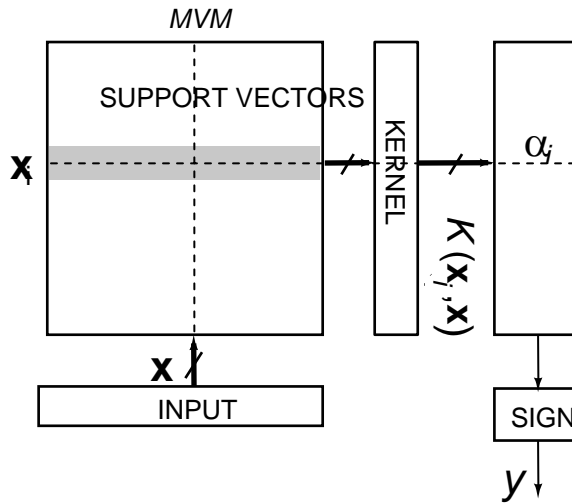


$$y = \text{sign}\left(\sum_{i \in S} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b\right)$$

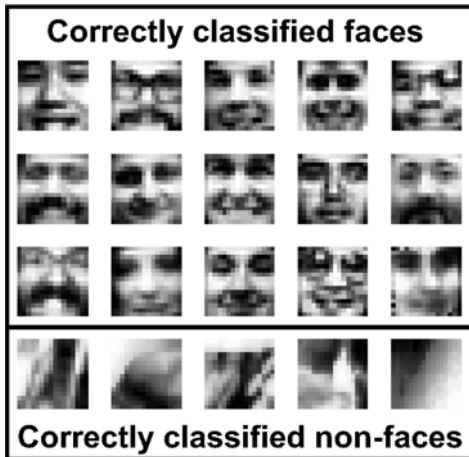
- Kernel inner-products are implemented by parallel matrix-vector multiplication (MVM).
- Silicon area and power dissipation are proportional to number of support vectors, favoring sparse SVM solutions.
- Sparsity in SVM training also guarantees proper generalization performance (Vapnik, 1995).

# Kerneltron III: Adiabatic Support Vector “Machine”

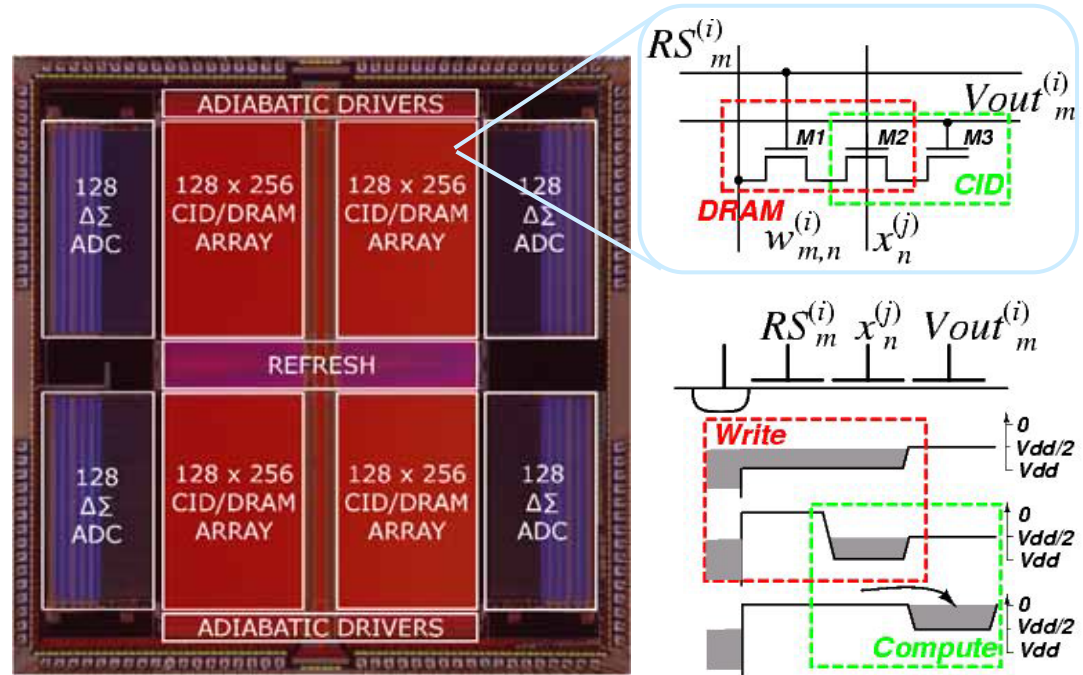
Karakiewicz, Genov, and Cauwenberghs, VLSI'2006; CICC'2007



$$y = \text{sign}\left(\sum_{i \in S} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b\right)$$



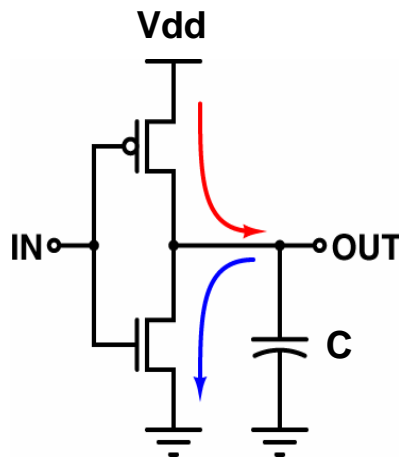
Classification results on MIT CBCL face detection data



- **1.2 TMACS / mW**

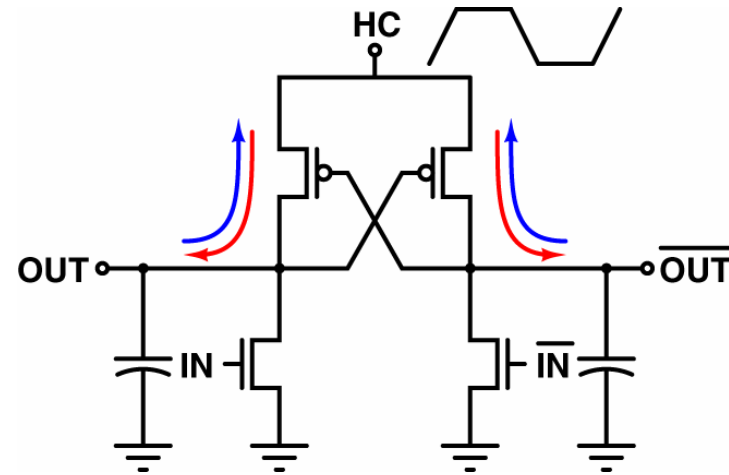
- *adiabatic resonant clocking conserves charge energy*
- *energy efficiency on par with human brain ( $10^{15}$  SynOP/S at 15W)*

# CMOS Logic vs. Adiabatic Computing



CMOS logic

- *Dynamic energy dissipation*  
$$E_{diss.} = CVdd^2$$

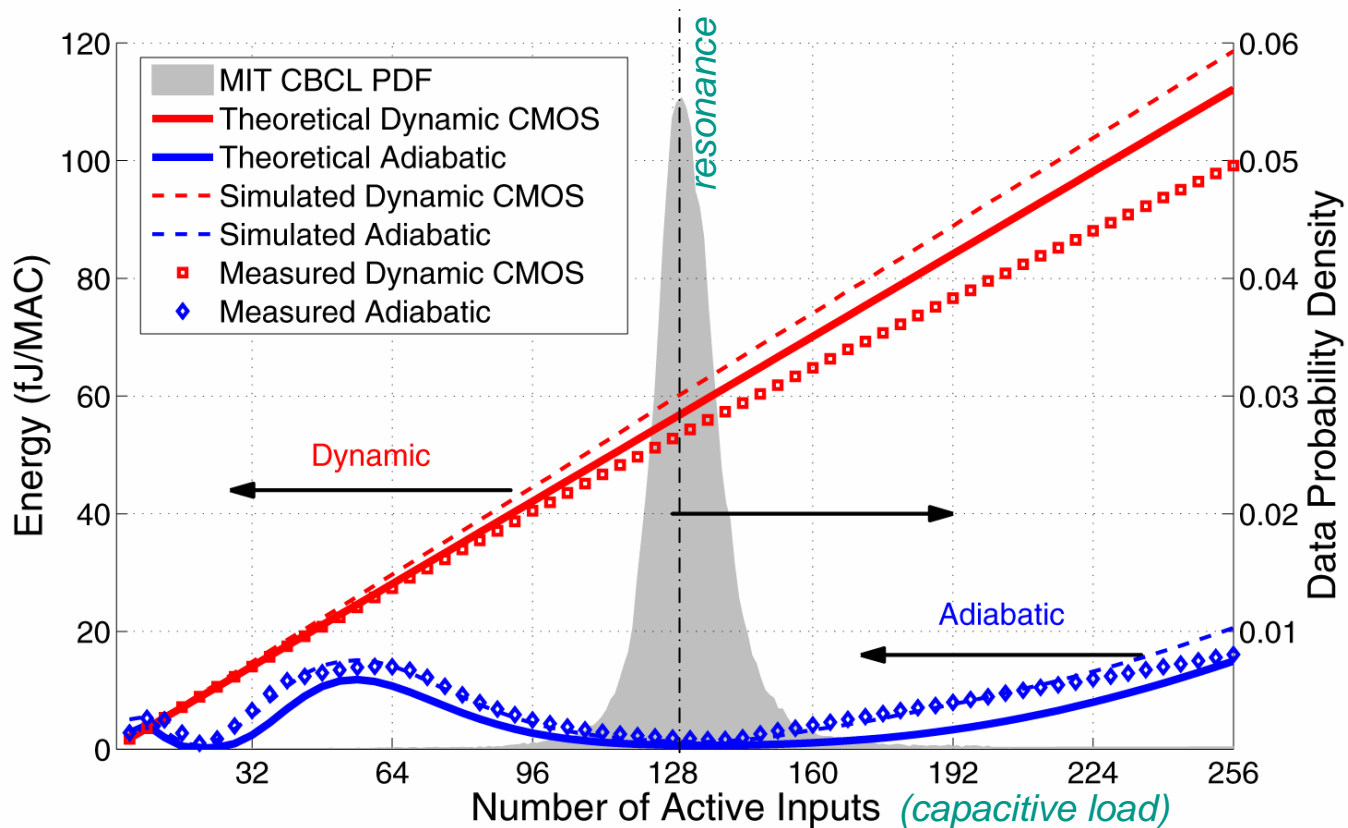
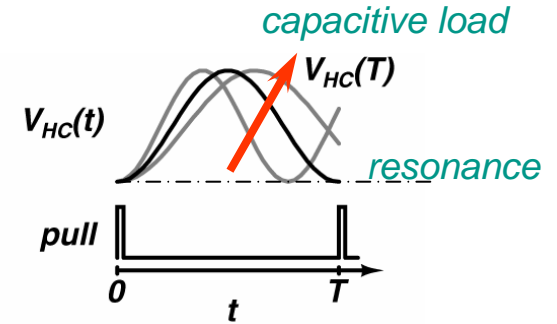
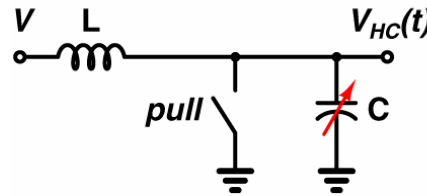
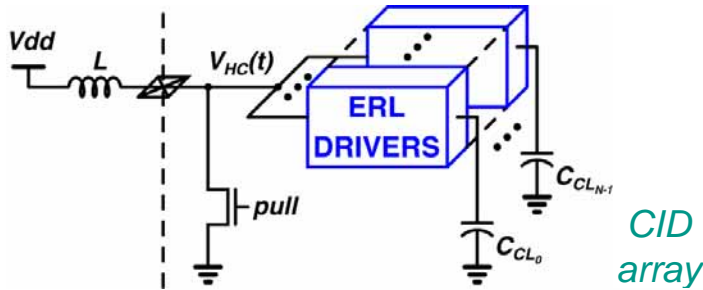


Energy recovery logic (ERL)  
(Y. Moon, JSSC'96)

- *'Hot clock' recycles energy*  
– *LC tank resonant clock*
- *Reversible computation*

# Resonant Charge Energy Recovery

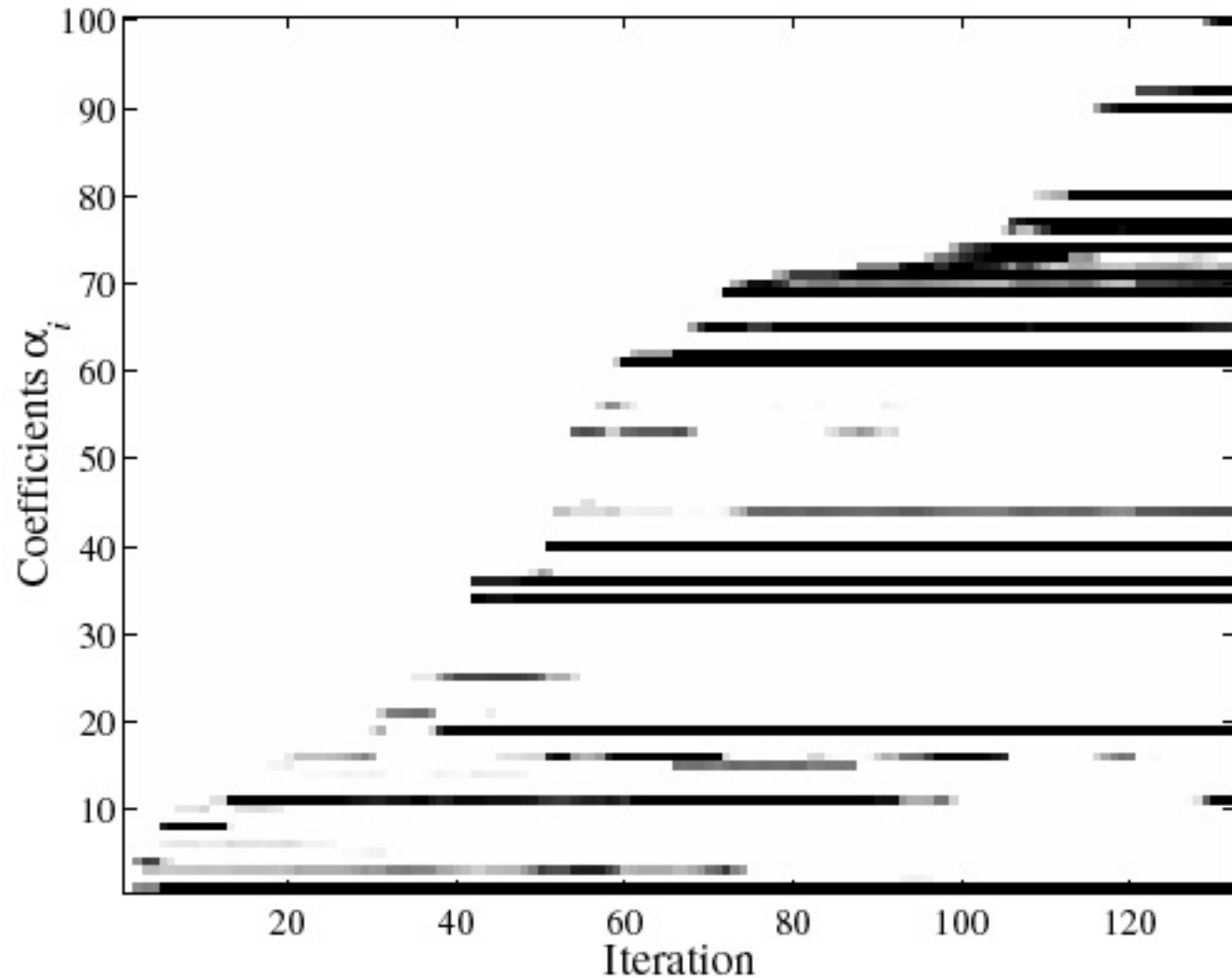
Karakiewicz, Genov, and Cauwenberghs, IEEE JSSC, 2007



# Incremental and Decremental SVM Learning

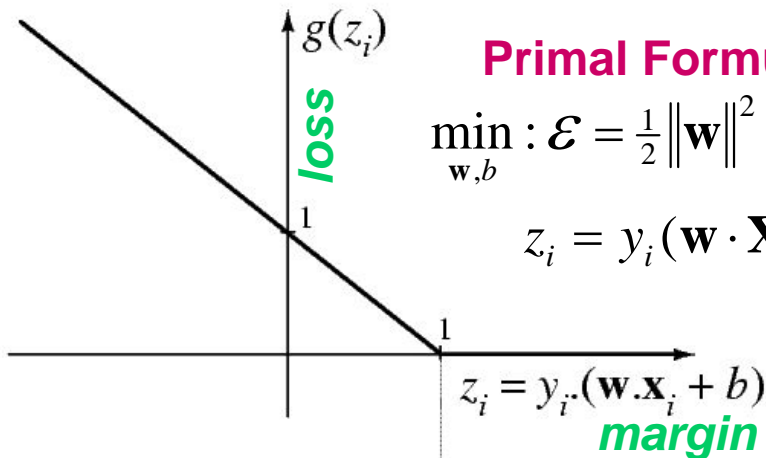
Cauwenberghs and Poggio, 2001

- Support Vector Machine training requires solving a linearly constrained quadratic programming problem in a number of coefficients equal to the number of data points.
- An incremental version, training one data point at a time, is obtained by solving the QP problem in recursive fashion, without the need for QP steps or inverting a matrix.
  - *On-line learning is thus feasible, with no more than  $L^2$  state variables, where  $L$  is the number of margin (support) vectors.*
  - *Training time scales approximately linearly with data size for large, low-dimensional data sets.*
- Decremental learning (adiabatic reversal of incremental learning) allows to directly evaluate the exact leave-one-out generalization performance on the training data.
- When the incremental inverse jacobian is (near) ill-conditioned, a direct L1-norm minimization of the  $\alpha$  coefficients yields an optimally sparse solution.



Trajectory of coefficients  $\alpha_i$  as a function of time during incremental learning, for 100 data points in the non-separable case, and using a Gaussian kernel.

# SVM Learning Revisited



## Primal Formulation

$$\min_{\mathbf{w}, b} : \mathcal{E} = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_i g(z_i)$$

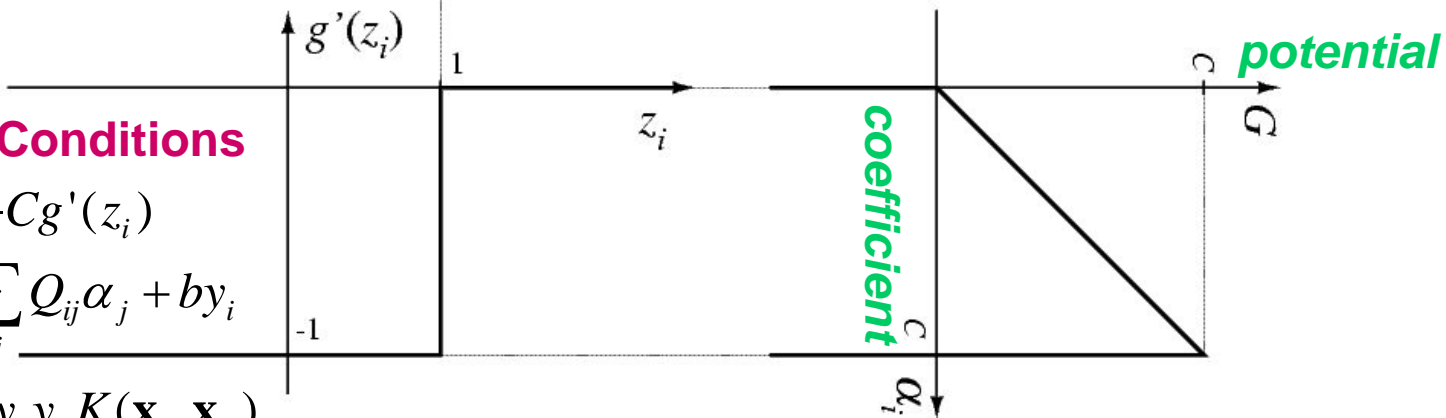
$$z_i = y_i(\mathbf{w} \cdot \mathbf{X}_i + b)$$

## KKT Conditions

$$\alpha_i = -Cg'(z_i)$$

$$z_i = \sum_j Q_{ij} \alpha_j + by_i$$

$$Q_{ij} = y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$



## Dual Formulation

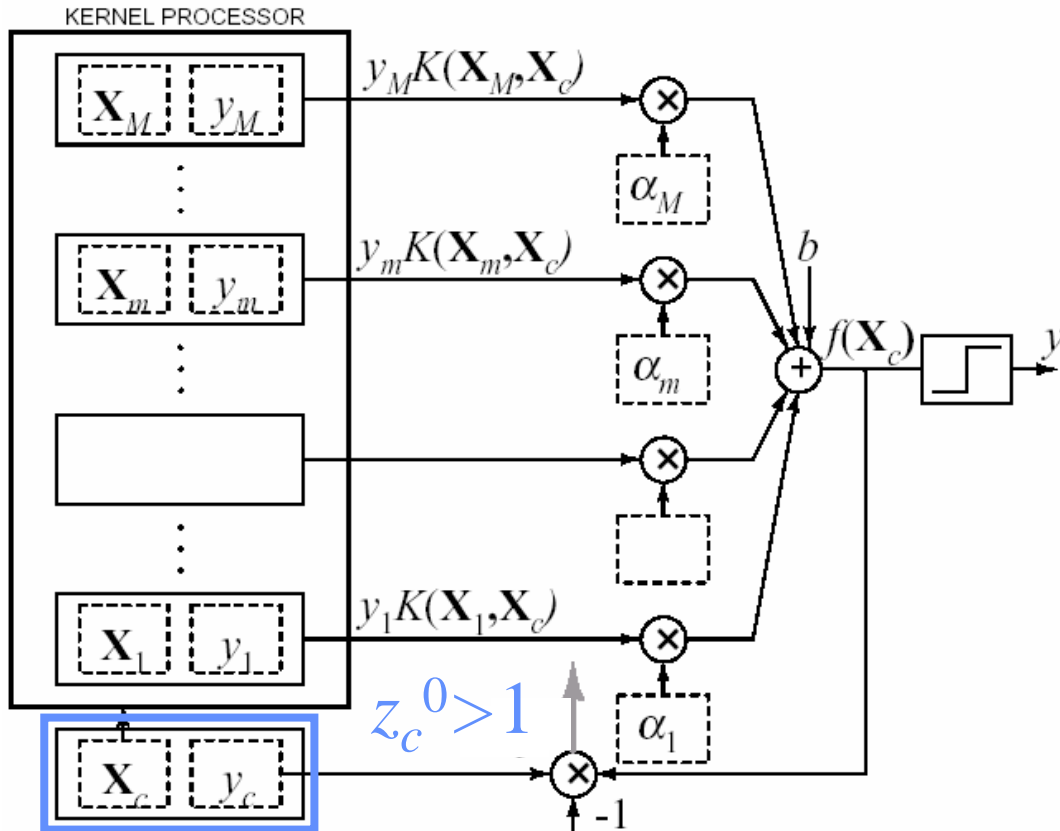
$$\min_{\mathbf{w}, b} : \mathcal{E}_2^{SVM} = \frac{1}{2} \sum_i \sum_j \alpha_i Q_{ij} \alpha_j - \sum_i \alpha_i$$

$$\text{subject to: } \sum_i y_i \alpha_i \equiv 0 \text{ and } 0 \leq \alpha_i \leq C, \forall i$$

Soft-Margin SVM Classification  
(Cortes and Vapnik, 1995)

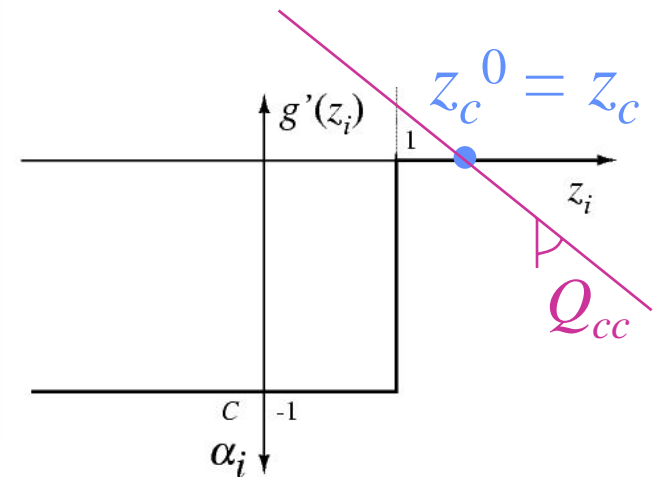
# Sequential On-Line SVM Learning

Chakrabarty, Genov and Cauwenberghs, 2003



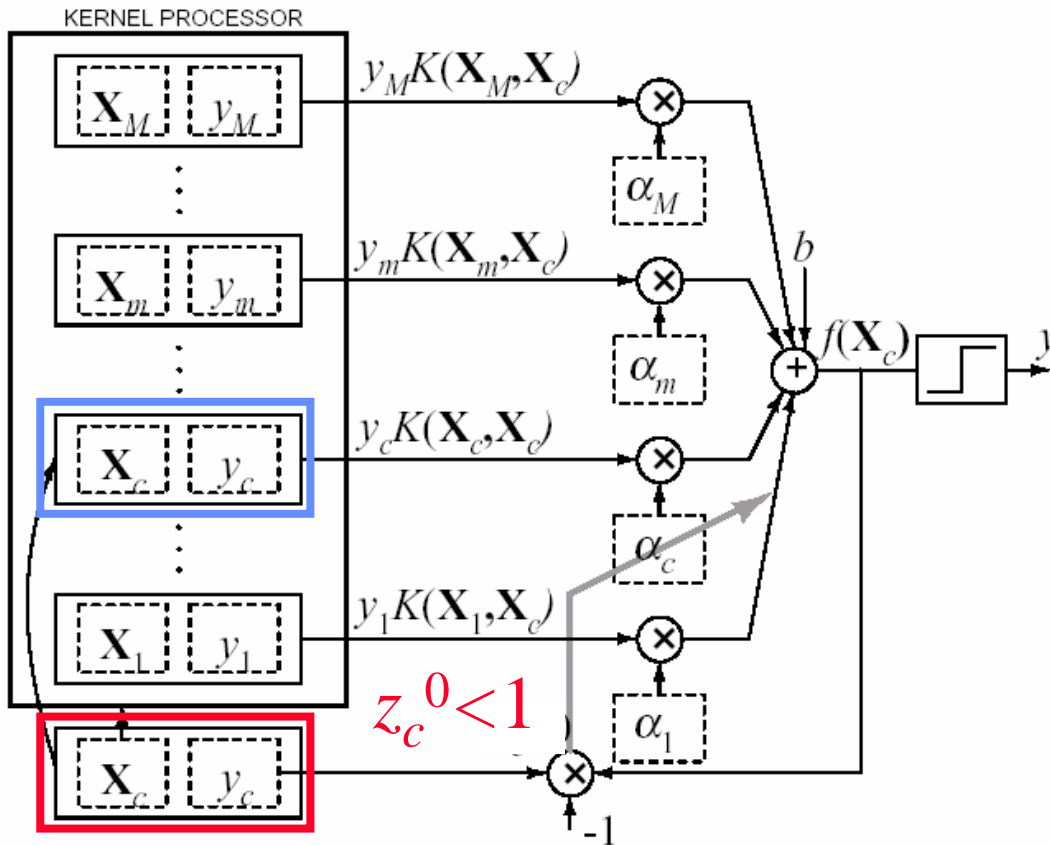
$$\alpha_c = -Cg'(z_c)$$

$$z_c \approx z_c^0 + Q_{cc} \alpha_c$$



# Sequential On-Line SVM Learning

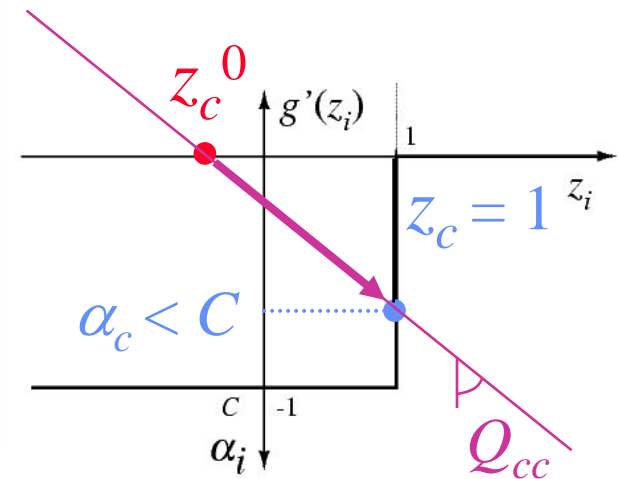
Chakrabarty, Genov and Cauwenberghs, 2003



*margin support vector*

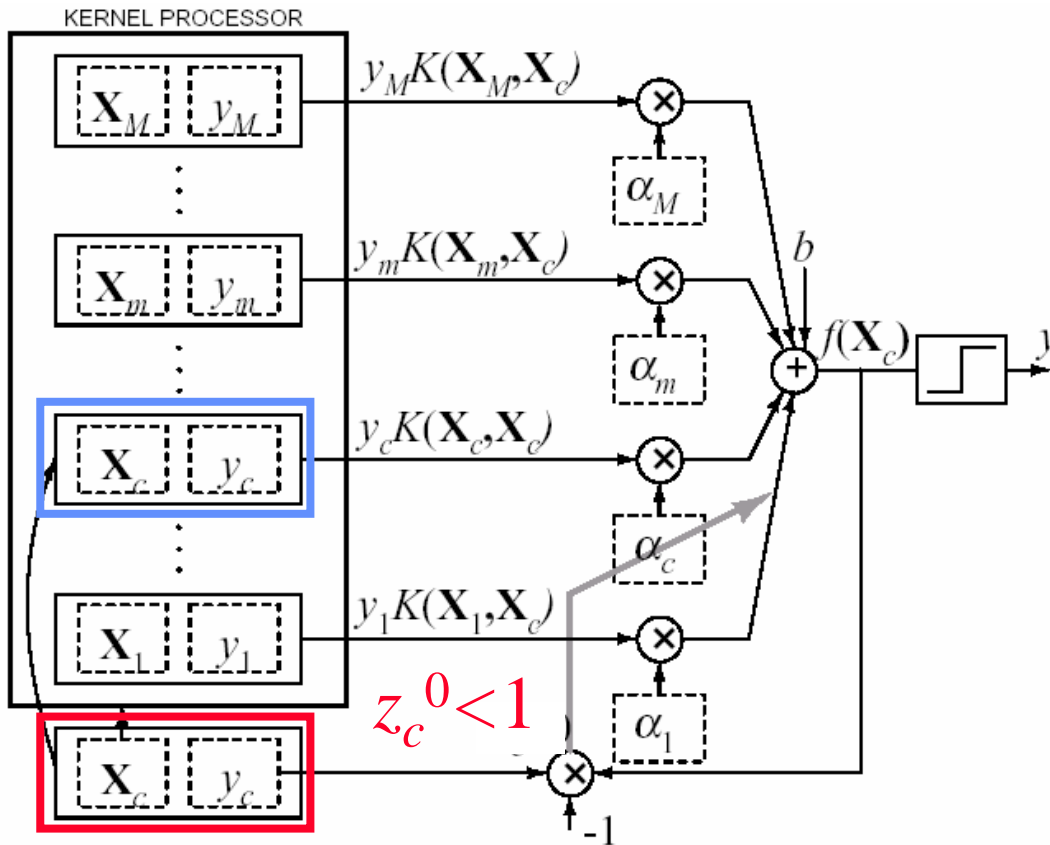
$$\alpha_c = -Cg'(z_c)$$

$$z_c \approx z_c^0 + Q_{cc} \alpha_c$$



# Sequential On-Line SVM Learning

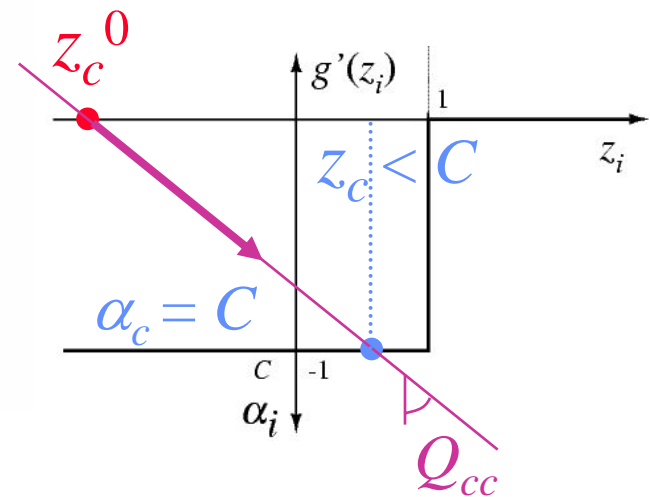
Chakrabarty, Genov and Cauwenberghs, 2003



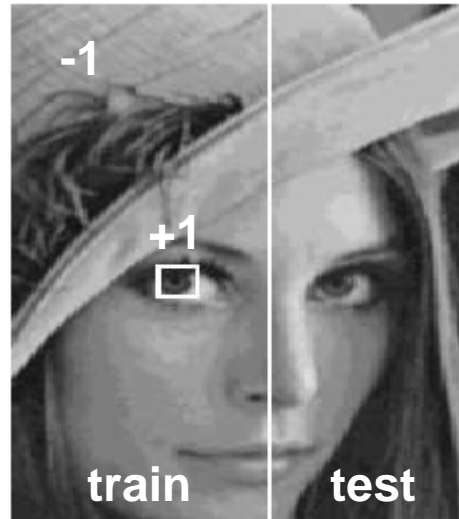
*error support vector*

$$\alpha_c = -Cg'(z_c)$$

$$z_c \approx z_c^0 + Q_{cc} \alpha_c$$

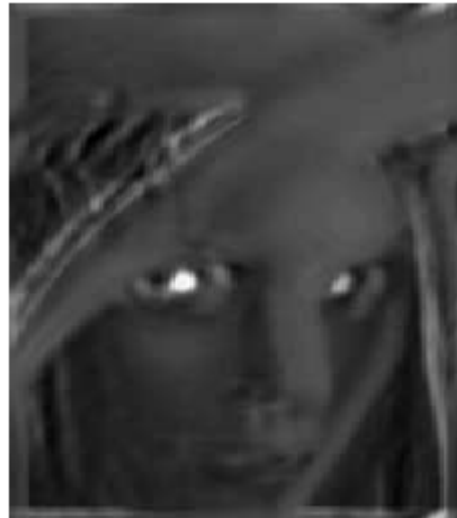


# Effects of Sequential On-Line Learning and Finite Resolution



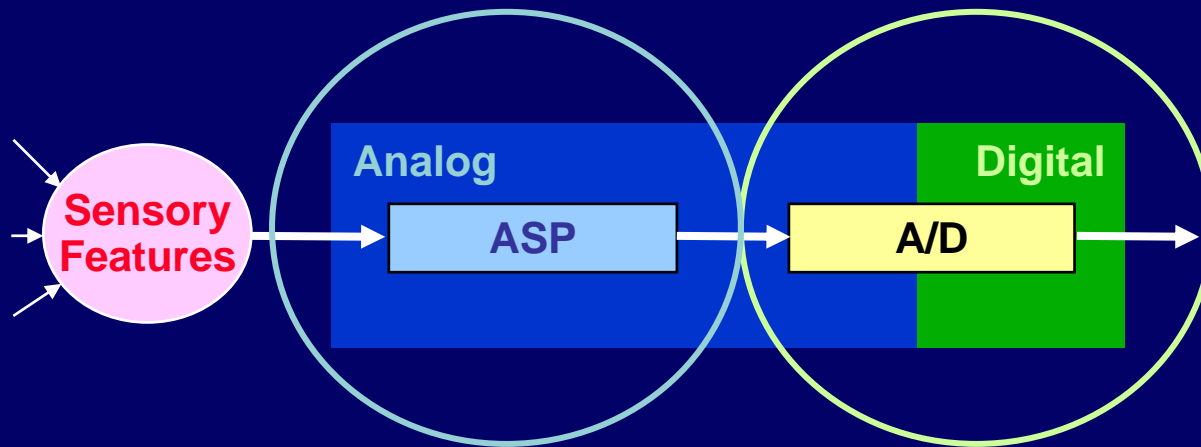
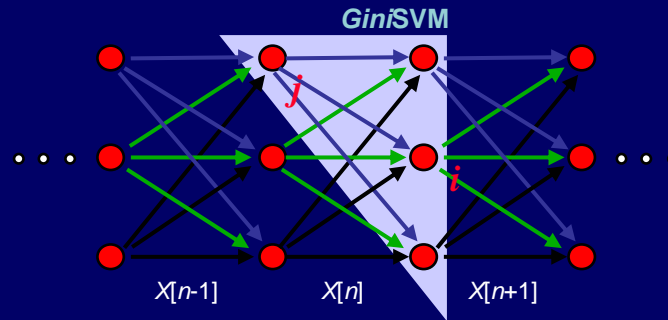
- *Matched Filter Response*

- *Batch Training*
- *Floating-Point Resolution*

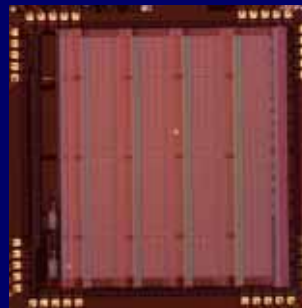


- *On-Line Sequential Training*
- *Kerneltron II*

# SVM Sequence Estimation



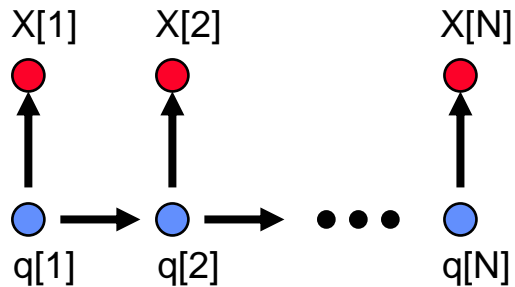
**MAP Forward  
Decoding**



**Sequence Identification**

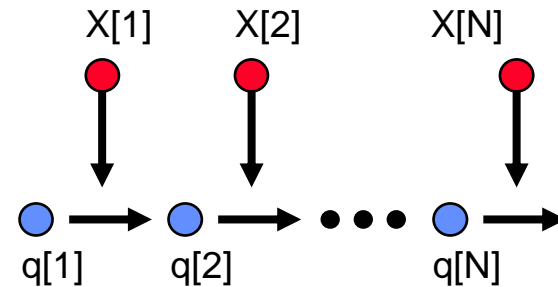
Sub-microwatt  
speaker verification  
and phoneme  
recognition  
(*NIPS'2004*)

# MLE vs. MAP Sequence Estimation



**Generative (MLE)**  
*HMM*

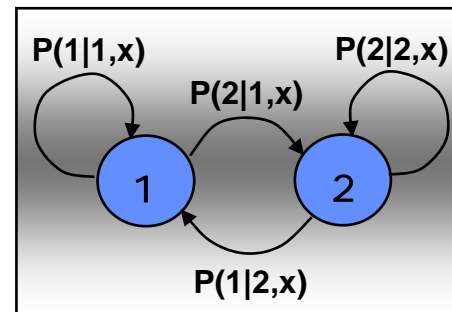
Density models (such as mixtures of Gaussians) require vast amounts of training data to reliably estimate parameters.



**Discriminative (MAP)**  
*FDKM*

Transition-based speech recognition  
*(H. Boullard and N. Morgan, 1994)*

MAP forward decoding



Transition probabilities generated by large margin probability regressor

# Forward Decoding Kernel Machines (FDKM)

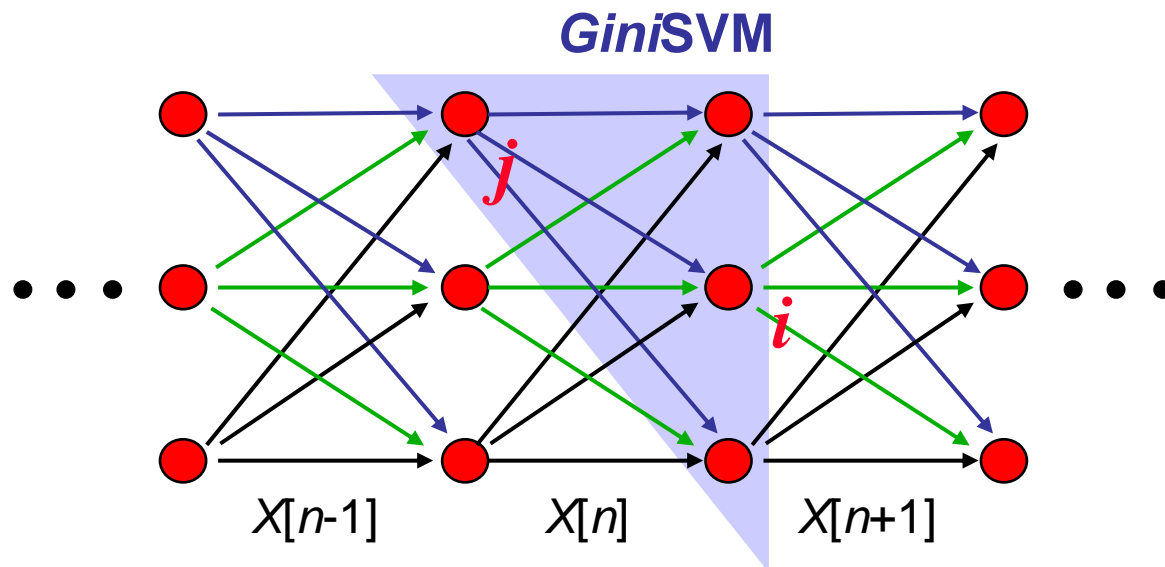
*Chakrabarty and Cauwenberghs (NIPS'2002)*

- Forward decoding of posterior probabilities  $\alpha_i$

$$\alpha_i[n] = \sum_j P_{ij}[n] \alpha_j[n-1]$$

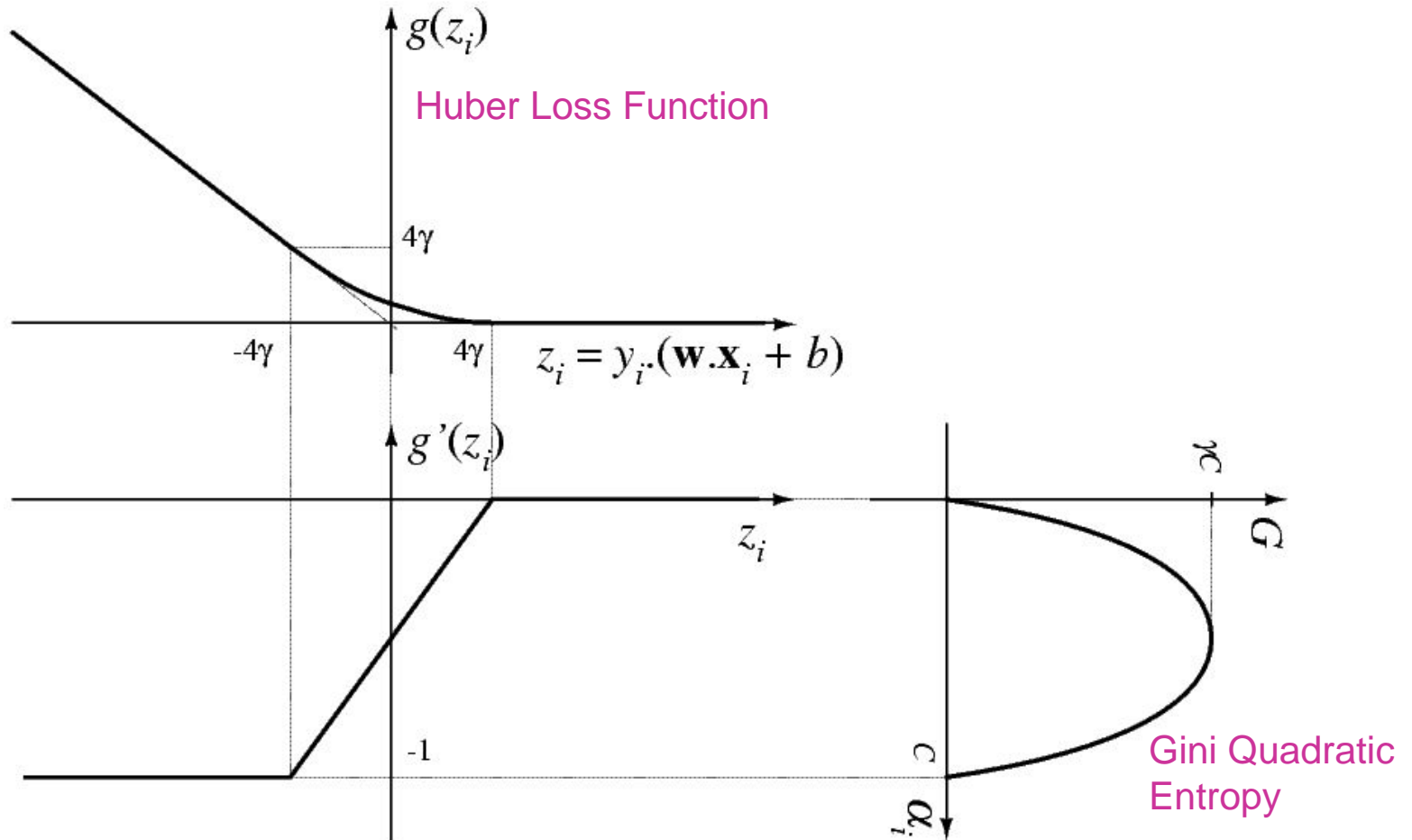
- Transition probabilities  $P_{ij}$  generated by SVM conditioned on input data  $X$

$$P_{ij}[n] = P(i | j, X[n]) \propto f_{ij}(X[n])$$



# GiniSVM Sparse Probability Regression

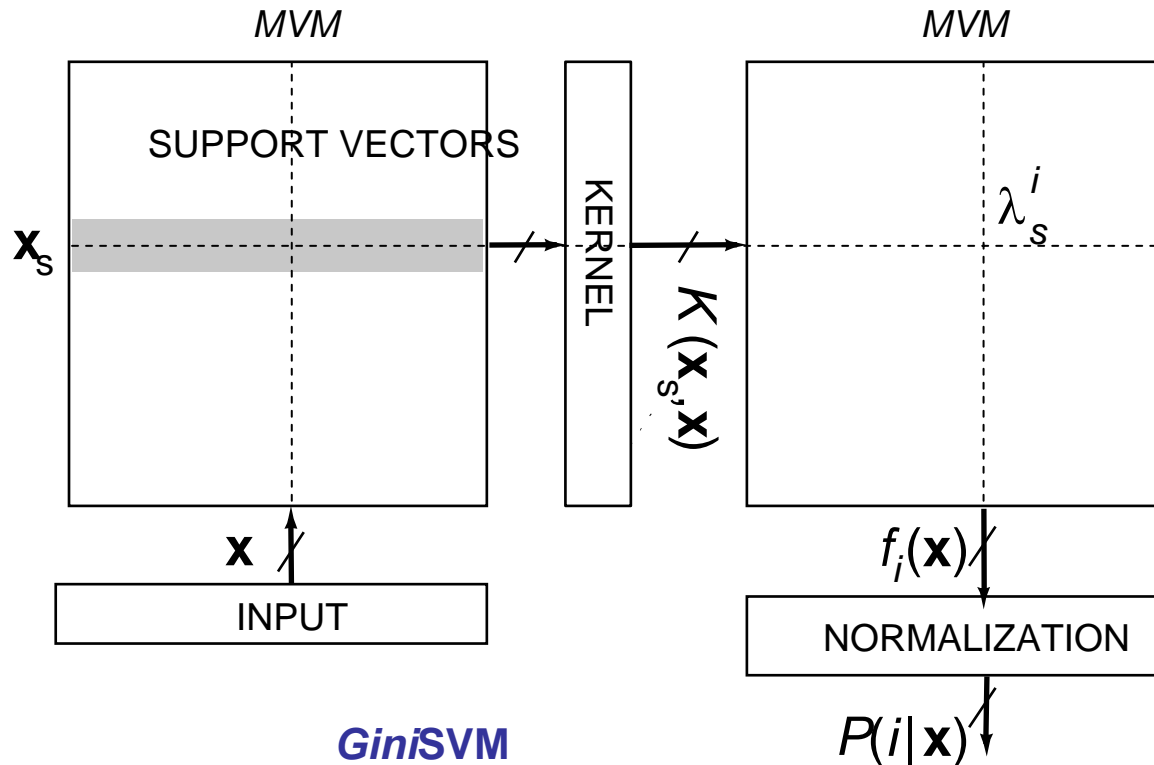
Chakrabarty and Cauwenberghs, JMLR 2007



$$\min_{\mathbf{w}, b} : \mathcal{E}_2^{kGini} = \frac{1}{2} \sum_i \sum_j \alpha_i Q_{ij} \alpha_j - C \sum_i H_{Gini} \left( \frac{\alpha_i}{C} \right)$$

$$\text{subject to : } \sum_i y_i \alpha_i \equiv 0 \text{ and } 0 \leq \alpha_i \leq C, \text{ with } H_{Gini}(a) = 4\gamma(1-a)a$$

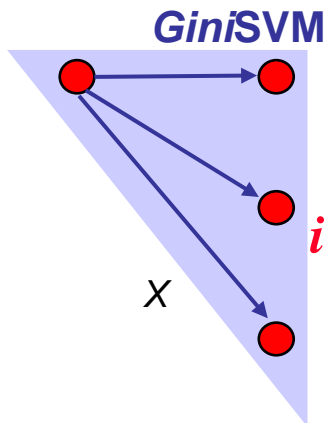
# GiniSVM Probability Regression



*Gini quadratic entropy in SVM training leads to sparse, large-margin regression of class probabilities (Chakrabartty et al. 2002)*

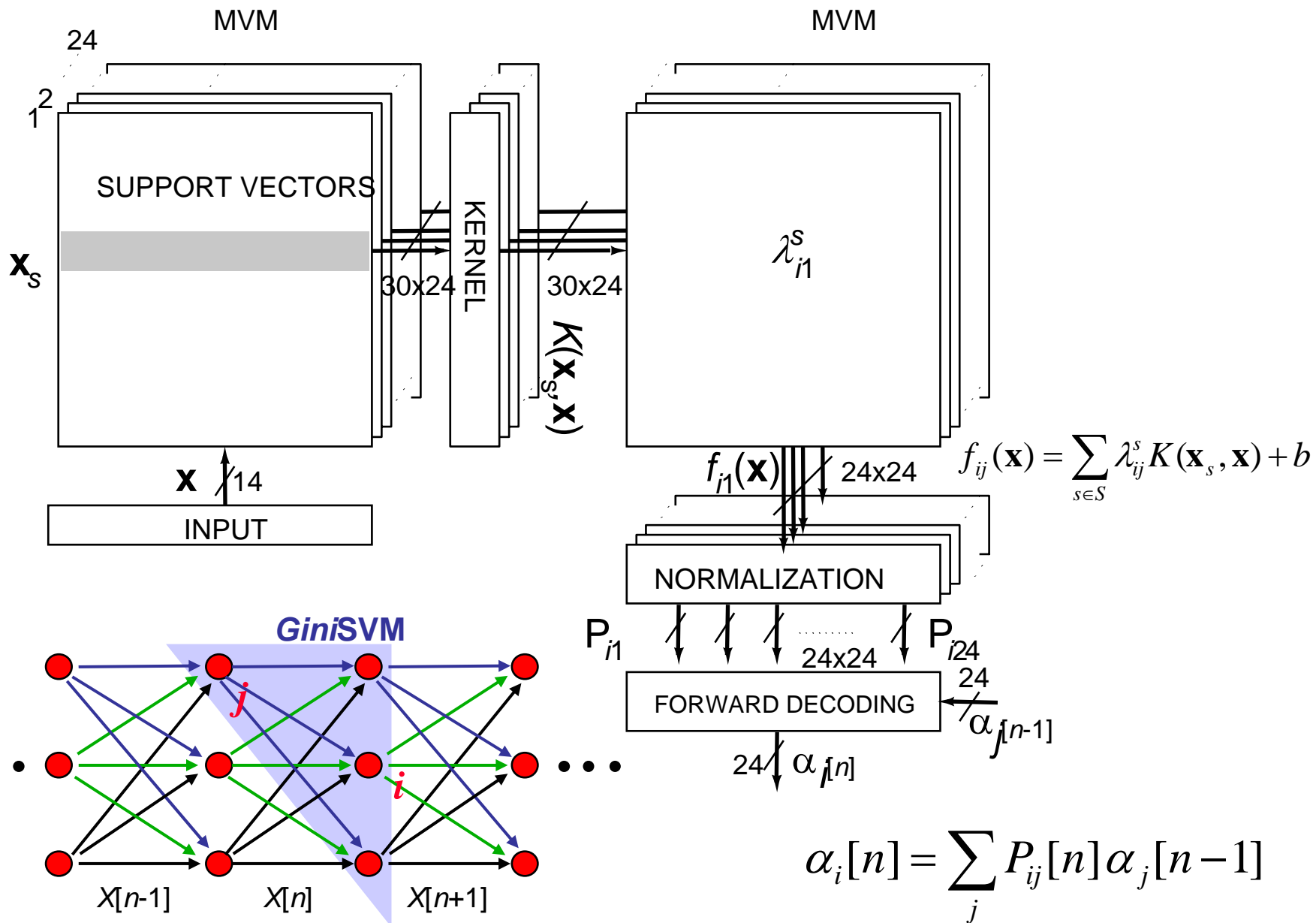
$$\sum_i P(i | \mathbf{x}) = 1$$

$$0 \leq P(i | \mathbf{x}) \leq 1$$



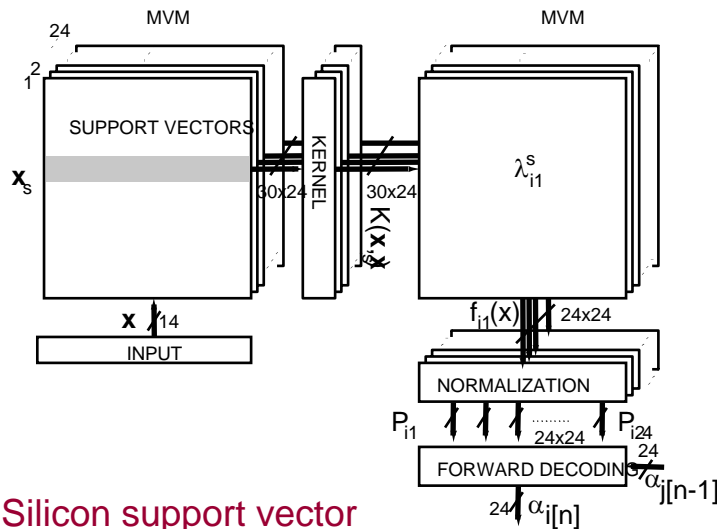
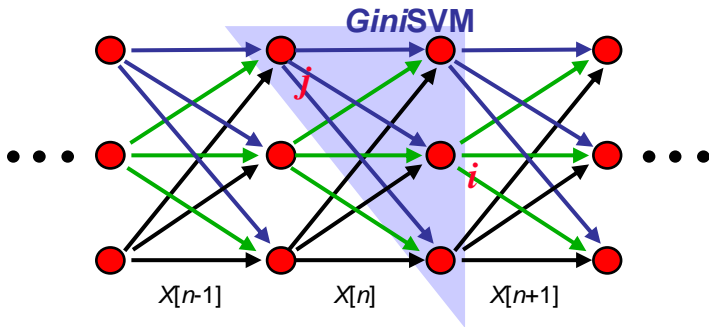
$$P(i | \mathbf{x}) \propto f_i(\mathbf{x}) = \sum_{s \in \mathcal{S}} \lambda_s^i y_i K(\mathbf{x}_s, \mathbf{x}) + b$$

# FDKM Architecture

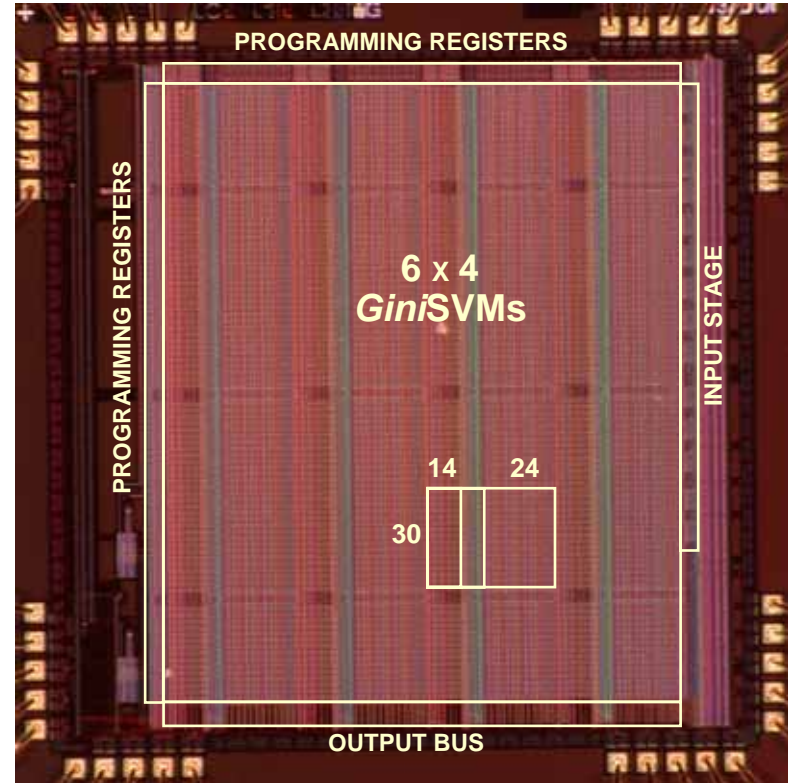


# GiniSVM/FDKM Processor

Chakrabarty and Cauwenberghs (NIPS'2004)



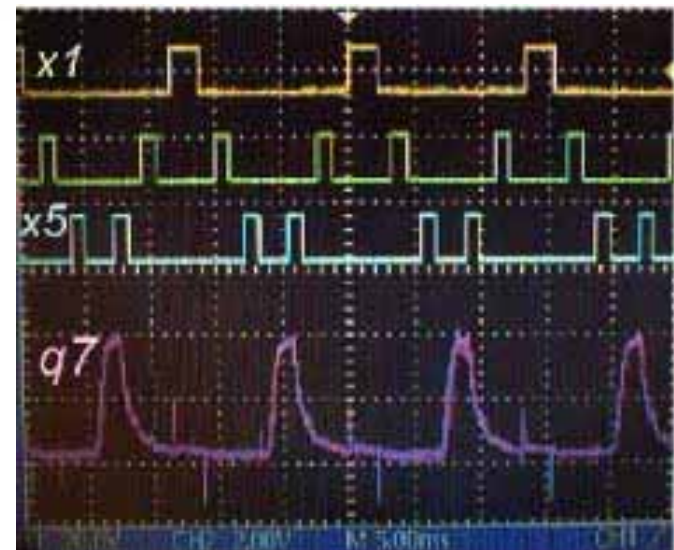
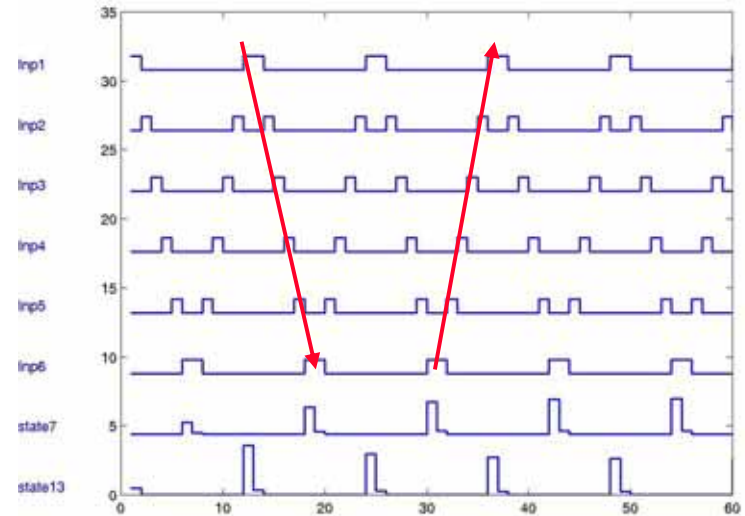
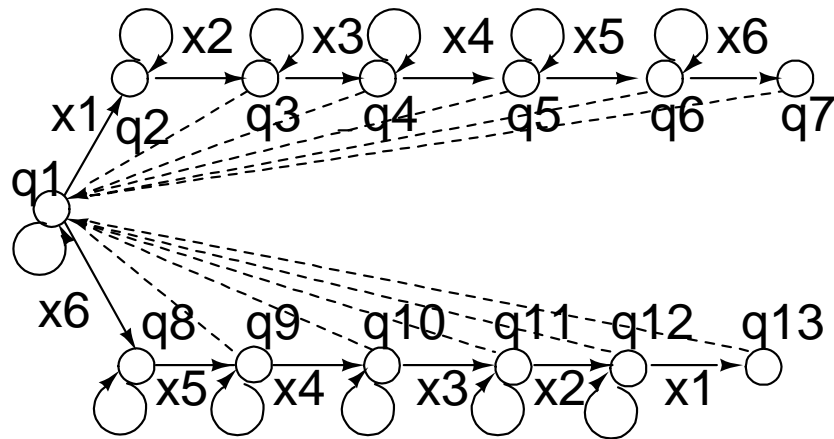
Silicon support vector machine (SVM) and forward decoding kernel machine (FDKM)



- **Sub-Microwatt Power**
- **Subthreshold translinear MOS circuits**
- **Programmable with floating-gate non-volatile analog storage**

# FDKM Dynamic Sequence Detection (80 nW)

*Chakrabarty and Cauwenberghs (NIPS'2004)*



# FDKM Training Formulation

Chakrabarty and Cauwenberghs, 2002

- Large-margin training of state transition probabilities, using regularized cross-entropy on the posterior state probabilities:

$$H = C \sum_{n=0}^{N-1} \sum_{i=0}^{S-1} y_i[n] \log \alpha_i[n] - \frac{1}{2} \sum_{j=0}^{S-1} \sum_{i=0}^{S-1} |w_{ij}|^2$$

- Forward Decoding Kernel Machines (FDKM) decompose an upper bound of the regularized cross-entropy (by expressing concavity of the logarithm in forward recursion on the previous state):

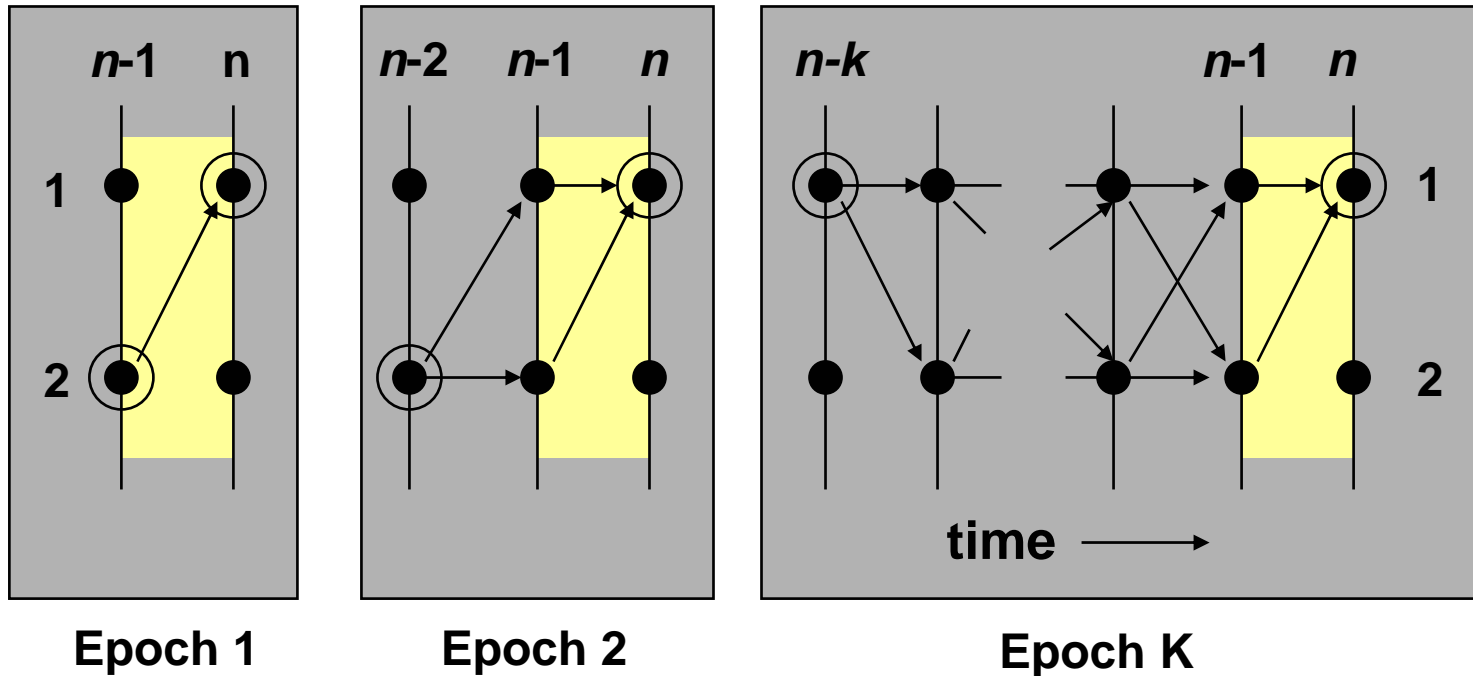
$$H \geq \sum_{j=0}^{S-1} H_j$$

which then reduces to  $S$  independent regressions of conditional probabilities, one for each outgoing state:

$$H_j = \sum_{n=0}^{N-1} C_j[n] \sum_{i=0}^{S-1} y_i[n] \log P_{ij}[n] - \frac{1}{2} \sum_{i=0}^{S-1} |w_{ij}|^2$$

$$C_j[n] = C \alpha_j[n-1]$$

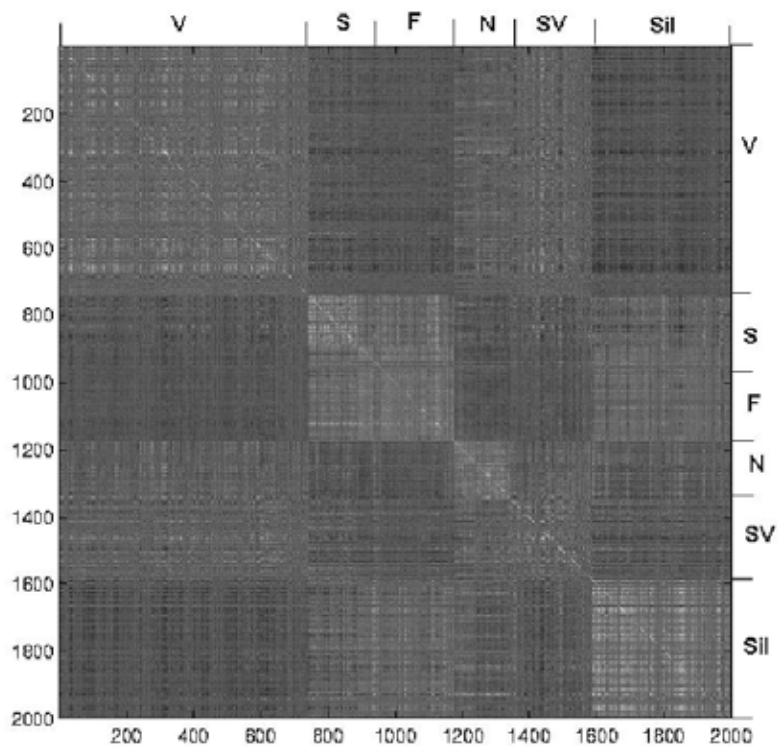
# Recursive MAP Training of FDKM



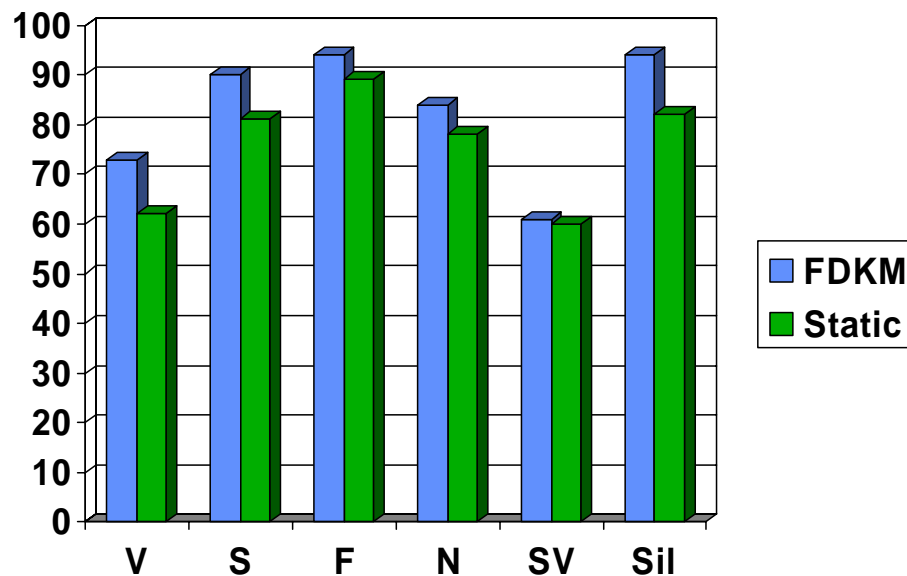
# Phonetic Experiments (TIMIT)

Chakrabarty and Cauwenberghs, 2002

Features: cepstral coefficients for *Vowels*, *Stops*, *Fricatives*, *Semi-Vowels*, and *Silence*



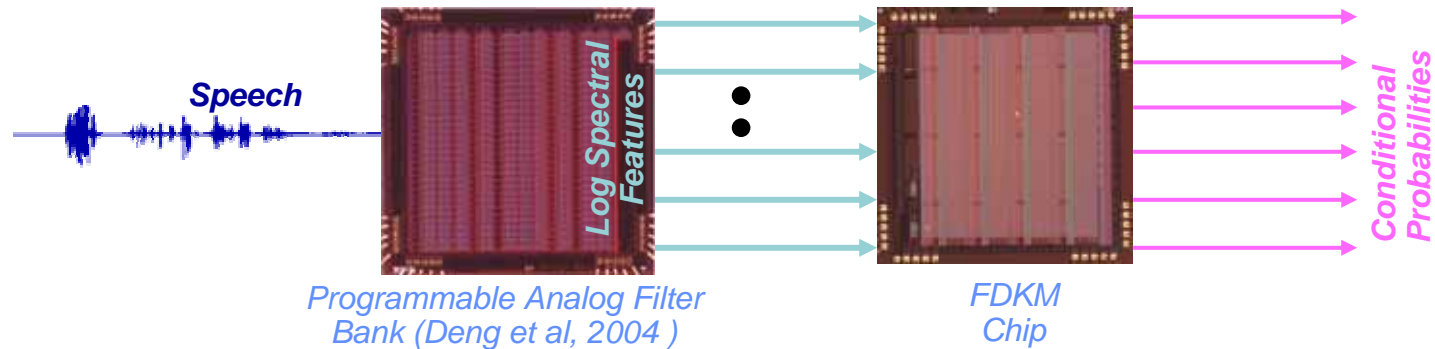
Kernel Map



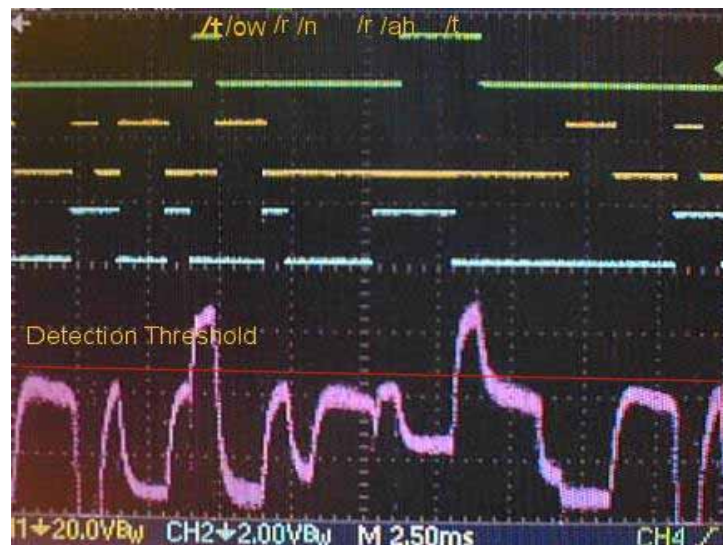
Recognition Rate

# On-Chip TIMIT Phone Recognition

*Chakrabarty and Cauwenberghs (NIPS'2004)*



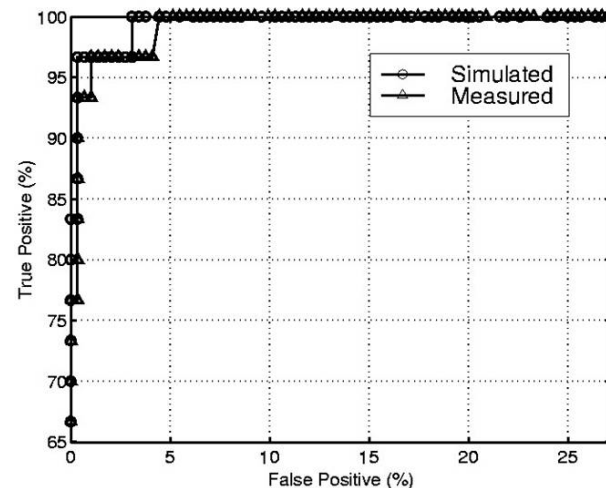
- 6 phones /t/n/r/ow/ah/eh/ from TIMIT corpus
- Thresholded Mel-cepstral features from log-compressed analog filterbank



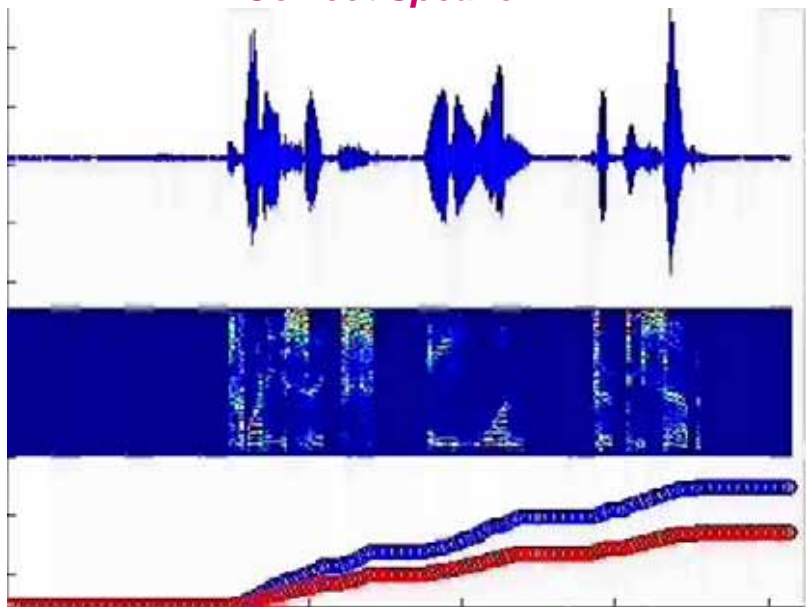
# On-Chip Speaker Verification (840nW)

*Chakrabarty and Cauwenberghs (NIPS'2004)*

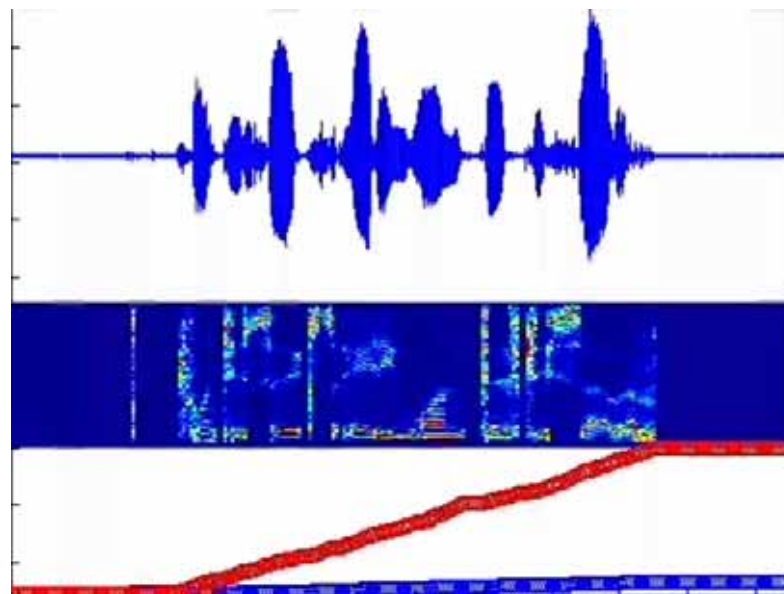
- 1 speaker and 10 imposters from YOHO dataset
- 92% recognition accuracy on 48 true and 432 imposter out-of-sample utterances
- 352 support vectors (47% FDKM chip capacity)
- 840 nW power at 25msec frame rate



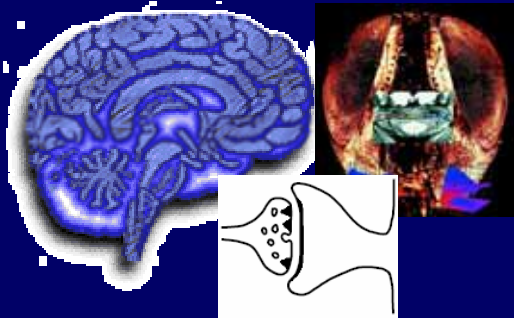
*Correct Speaker*



*Imposter*

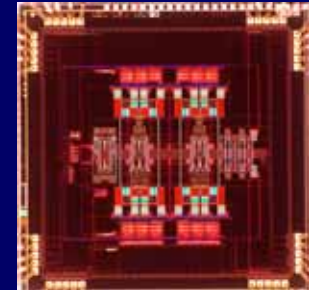


# Closing the Loop: Interactive Neural/Artificial Intelligence



*Neuromorphic Engineering*

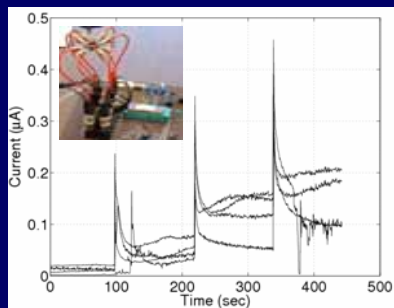
Adaptive Sensory Feature Extraction and Pattern Recognition



**Neuro Bio**

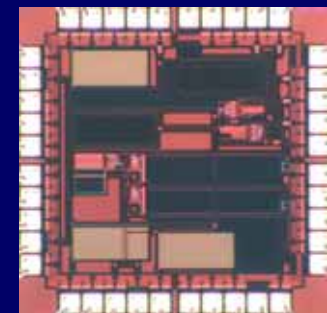
Learning & Adaptation

**Micropower Mixed-Signal VLSI**



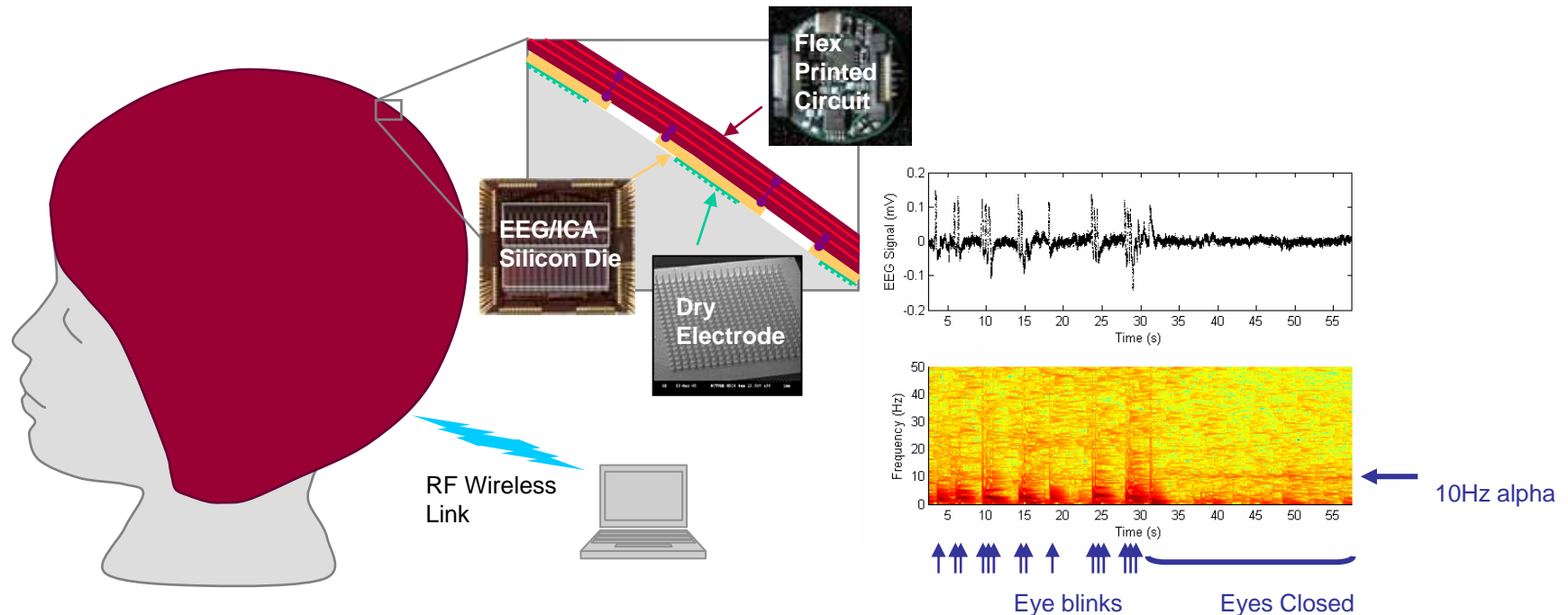
*Neurosystems Engineering*

Biosensors, Neural Prostheses and Brain Interfaces



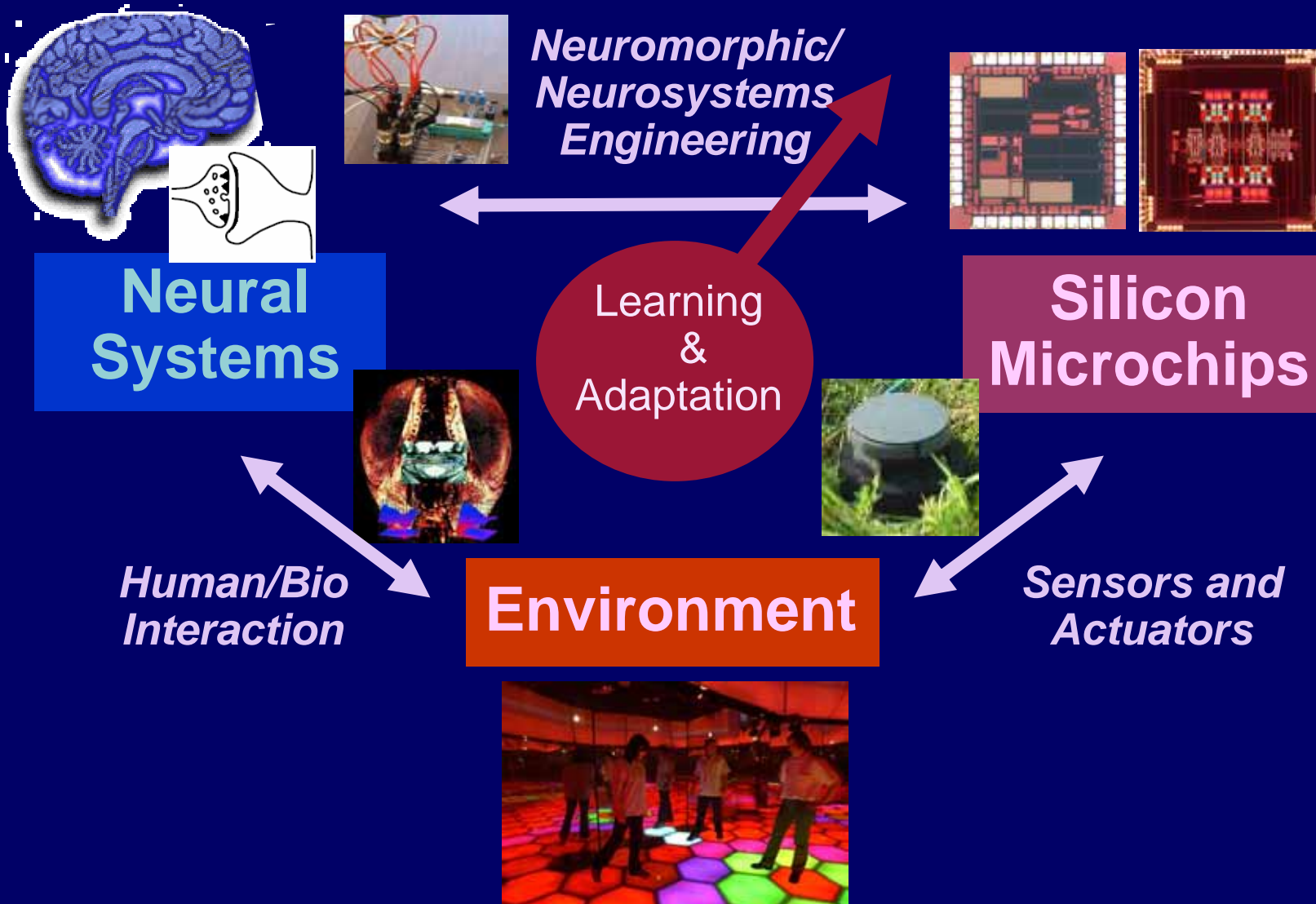
# Wireless EEG/ICA Neurotechnology

with Tom Sullivan, Steve Deiss, Tzyy-Ping Jung and Scott Makeig, 2007  
supported by DARPA DSO



- **Integrated EEG/ICA wireless EEG recording system**
  - Scalable towards 1000+ channels
  - Dry contact electrodes
  - Wireless, lightweight
  - Integrated, distributed independent component analysis (ICA)

# Towards Large-Scale Distributed, Pervasive Computational Intelligence



# References

## Support Vector Machines and Kernel Methods

- M. Aizerman, E. Braverman, and L. Rozonoer, "Theoretical foundations of the potential function method in pattern recognition learning," *Automation and Remote Control*, vol. **25**, pp. 821-837, 1964.
- B.E. Boser, I.M. Guyon and V.N. Vapnik, "A training algorithm for optimal margin classifiers," *Proc. 5th ACM Workshop on Computational Learning Theory (COLT)*, ACM Press, pp. 144-152, July 1992.
- C. Cortes and V. Vapnik, "Support vector networks," *Machine Learning*, vol. **20**, pp. 273-297, 1995.
- J. Mercer, "Functions of positive and negative type and their connection with the theory of integral equations," *Philos. Trans. Royal Society London, A*, vol. **209**, pp. 415-446, 1909.
- V. Vapnik, *The Nature of Statistical Learning Theory*, Springer, 1995; 2nd Ed., 2000.

## Feature Coding

- S. Chakrabarty, Y. Deng and G. Cauwenberghs, "*Robust Speech Feature Extraction by Growth Transformation in Reproducing Kernel Hilbert Space*," *IEEE Trans. Audio, Speech, and Language Processing*, vol. **15** (6), pp. 1842-1849, 2007.
- Y. Deng, S. Chakrabarty and G. Cauwenberghs, "*Analog Auditory Perception Model for Robust Speech Recognition*," *Proc. IEEE Int. Joint Conf. Neural Networks (IJCNN'2004)*, Budapest Hungary, July 25-29, 2004.
- C.P. Papageorgiou, M. Oren and T. Poggio, "*A general framework for object detection*," in *Proceedings of International Conference on Computer Vision*, 1998.

# References (cont'd)

## Incremental and On-Line SVM Learning

- G. Cauwenberghs and T. Poggio, "[Incremental and Decremental Support Vector Machine Learning](#)," *Adv. Neural Information Processing Systems (NIPS\*2000)*, Cambridge, MA: MIT Press, vol. **13**, 2001.
- C.P. Diehl and G. Cauwenberghs, "[SVM Incremental Learning, Adaptation and Optimization](#)," *Proc. IEEE Int. Joint Conf. Neural Networks (IJCNN'2003)*, Portland OR, July 20-23, 2003.
- R. Genov, S. Chakrabartty and G. Cauwenberghs, "[Silicon Support Vector Machine with On-Line Learning](#)," *Int. J. Pattern Recognition and Artificial Intelligence*, vol. **17** (3), pp. 385-404, 2003.

## GiniSVM and Forward Decoding Kernel Machines

- S. Chakrabartty and G. Cauwenberghs, "[Forward Decoding Kernel Machines: A Hybrid HMM/SVM Approach to Sequence Recognition](#)," *Proc. SVM'2002*, Lecture Notes in Computer Science, vol. **2388**, pp. 278-292, 2002.
- S. Chakrabartty and G. Cauwenberghs, "[Forward-Decoding Kernel-Based Phone Sequence Recognition](#)," *Adv. Neural Information Processing Systems (NIPS'2002)*, Cambridge: MIT Press, vol. **15**, 2003.
- S. Chakrabartty and G. Cauwenberghs, "[Gini-Support Vector Machine: Quadratic Entropy Based Multi-class Probability Regression](#)," *J. Machine Learning Research*, vol. **8** (4), pp. 813-839, 2007.

# References (cont'd)

## Analog VLSI Adaptive and Sensory Microsystems

- G. Cauwenberghs and M. Bayoumi, Eds., *Learning on Silicon: Adaptive VLSI Microsystems*, Kluwer Academic (Springer), 1999.
- G. Cauwenberghs and V. Pedroni, "A Low-Power CMOS Analog Vector Quantizer," *IEEE Journal of Solid-State Circuits*, vol. **32** (8), pp. 1278-1283, 1997.
- S. Chakrabartty and G. Cauwenberghs, "Sub-Microwatt Analog VLSI Support Vector Machine for Pattern Classification and Sequence Estimation," *Adv. Neural Information Processing Systems (NIPS'2004)*, Cambridge: MIT Press, **17**, 2005.
- S. Chakrabartty and G. Cauwenberghs, "Sub-Microwatt Analog VLSI Trainable Pattern Classifier," *IEEE J. Solid-State Circuits*, vol. **42** (5), pp. 1169-1179, 2007.
- R. Genov, G. Cauwenberghs, G. Mulliken and F. Adil, "A 5.9mW 6.5GMACS CID/DRAM Array Processor," *Proc. European Solid-State Circuits Conference (ESSCIRC'2002)*, Florence Italy, Sept. 24-26, 2002.
- R. Genov and G. Cauwenberghs, "Kerneltron: Support Vector Machine in Silicon," *IEEE Trans. Neural Networks*, vol. **14** (5), pp. 1426-1434, 2003.
- U. Mallik, M. Clapp, E. Choi, G. Cauwenberghs and R. Etienne-Cummings, "Temporal Change Threshold Detection Imager," *Proc. IEEE Int. Solid-State Circuits Conf. (ISSCC'2005)*, San Francisco, Febr. 6-10, 2005.
- T.J. Sullivan, S.R. Deiss, G. Cauwenberghs, and T.P. Jung, "A Low-Noise, Low-Power EEG Acquisition Node for Scalable Brain-Machine Interfaces," *SPIE Proc. Bioengineered and Bioinspired Systems*, vol. 6592, DOI:10.1117/12.724019, 2007.

# References (cont'd)

## Charge Recovery and Adiabatic Computation

- R. Karakiewicz, R. Genov, A. Abbas and G. Cauwenberghs, "175 GMACS/mW Charge-Mode Adiabatic Mixed-Signal Array Processor," *Proc. IEEE 2006 Symp. VLSI Circuits*, Honolulu HI, Jun 13-17, 2006.
- R. Karakiewicz, R. Genov, and G. Cauwenberghs, "480-GMACS/mW Resonant Adiabatic Mixed-Signal Processor Array for Charge-Based Pattern Recognition," *IEEE J. Solid-State Circuits*, vol. **42** (11), pp. 2573-2584, 2007.
- R. Karakiewicz, R. Genov, and G. Cauwenberghs, "1.1 TMACS/mW load-balanced resonant charge-recycling array processor," *IEEE Custom Integrated Circuits Conference (CICC'2007)*, 2007.
- Y. Moon and D. K. Jeong, "An efficient charge recovery logic circuit," *IEEE J. of Solid-State Circuits*, SC-31(4): 514-522, April 1996.