Knowledge Processing
with Neural Networks

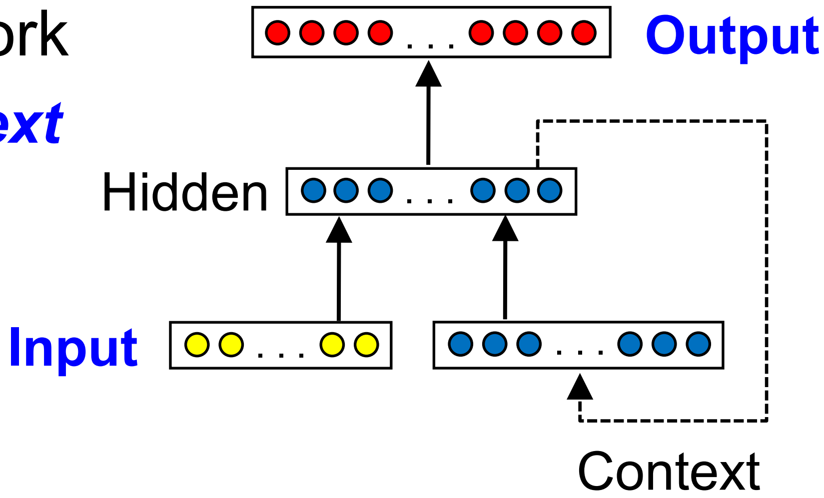
Lecture 11: Advanced Recurrent Neural Architectures

http://www.informatik.uni-hamburg.de/WTM/

Recurrent Neural Networks

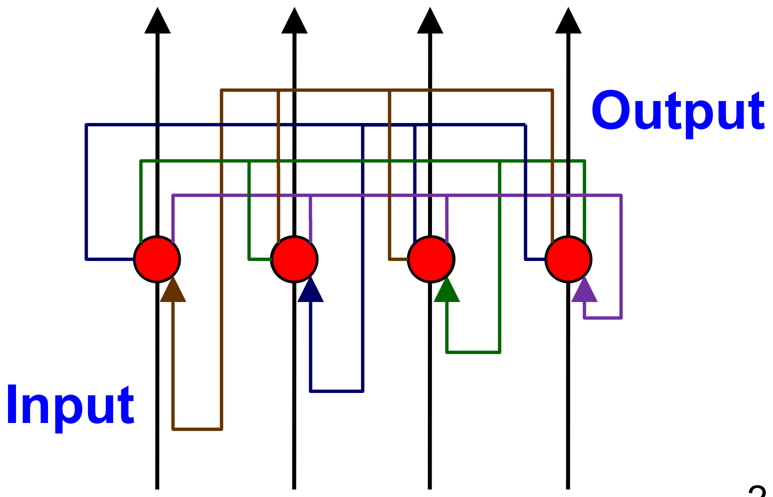
- Simple Recurrent Neural Network
  - Previous activation adds ***context*** to the current activation

  Examples:
  - Elman Network
  - Jordan Network



- Fully Connected Neural Network
  - Often called ***auto-associator***

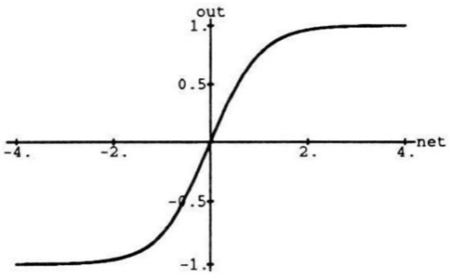
  Examples:
  - Hopfield Network (binary)
  - Boltzmann machine (stochastic)



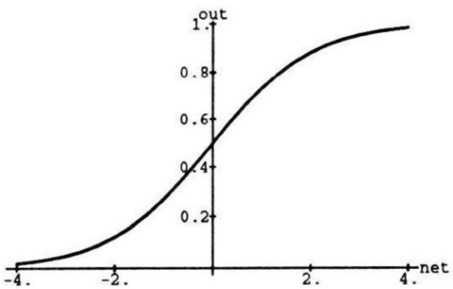
Revision: Learning in Neural Networks

- Known gradient descent method for MLPs:
  the ***back-propagation*** algorithm:
  - Forward pass: Calculate ***activation forward*** through the net
  - Determine error at output nodes
  - Backward pass: Propagate ***error backward*** through the net
  - Update weights by cumulated deltas

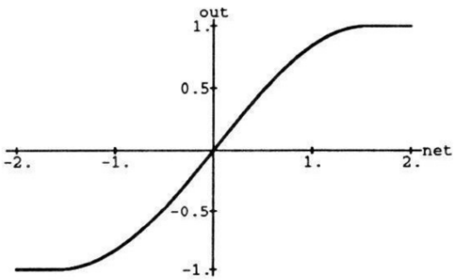
- Important: employ a ***differentiable threshold*** function



tanh(x)



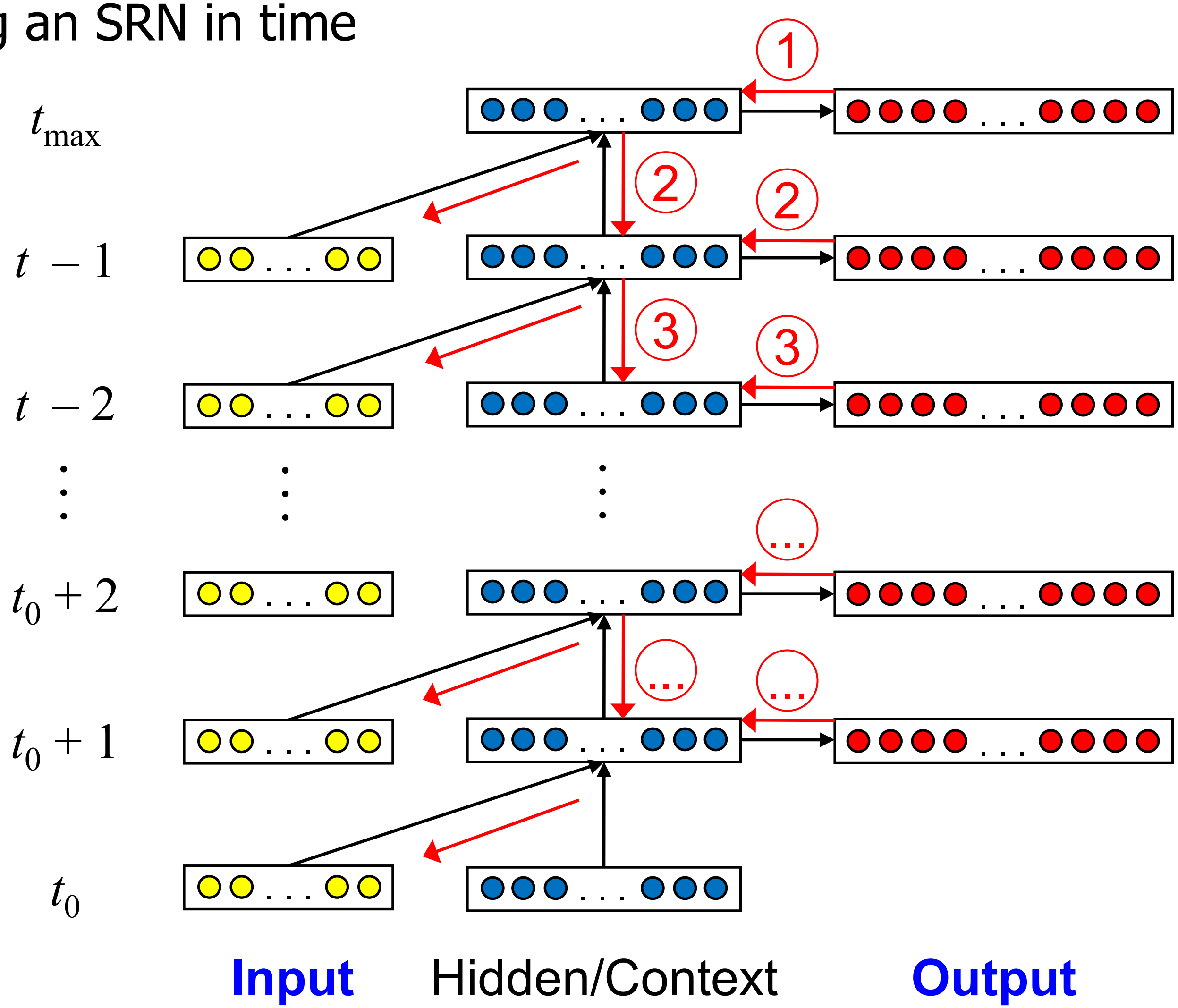
logistic 1/(1+exp(-x))



sin(x) till saturation

Training for Recurrent Neural Networks

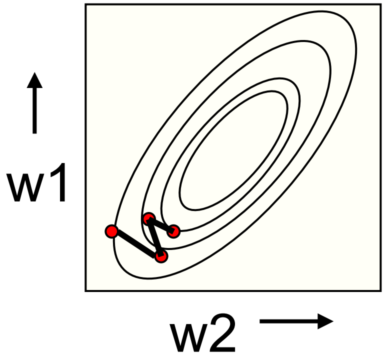
Unfolding an SRN in time

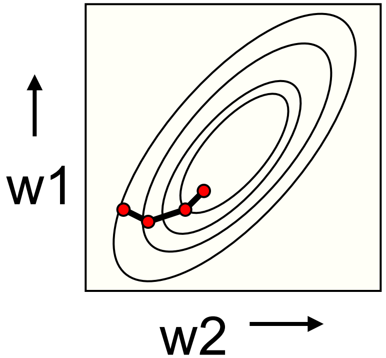


Training for Recurrent Neural Networks (cont.)

- Idea: ***unfold network*** in time and use a backprop variant:
  - **B**ack-**p**ropagation **t**hrough **t**ime
  - **T**runcated **b**ack-**p**ropagation **t**hrough **t**ime
  - **R**eal-**t**ime **r**ecurrent **l**earning
  - ...

- Also important: Online vs. Batch learning
  - Online: More exploration; more dynamic
  - Batch: Faster convergence to a minimum; steepest descent ≠ global minimum
  - Good Idea: ***Mini batches***





RNN applications and issues

- Recurrent Neural Networks are ***efficiently applicable*** to
  - Sequence and stock market prediction
  - Handwriting and speech recognition
  - Attentive vision and keywords spotting
  - Music composition ... *and much more!*
- Advantages – so far:
  - Bio-inspired method to problem solving ***using*** some ***context***...
  - ... that is still ***deterministic*** and can be analysed
- Issues:
  - Time leaks or disturbations in the sequences are destructive
  - Often uses only a fraction of the available information

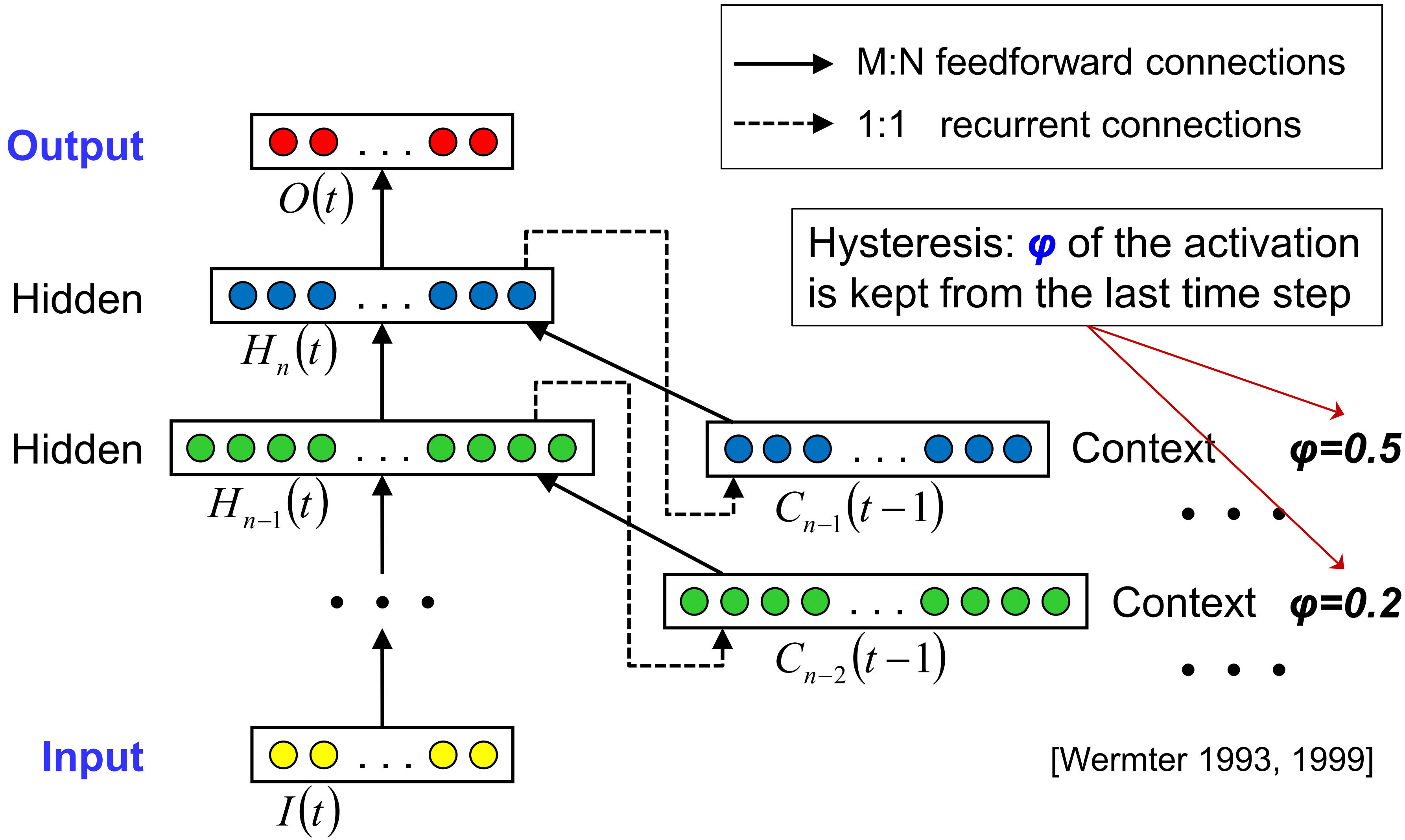
1. RNN with multiple context layers

- Characteristics
  - Arbitrary number of hidden & context layers
  - Every context layer memorises a ***different*** degree of ***dynamics***

  Example:
  - Recurrent Plausibility Networks

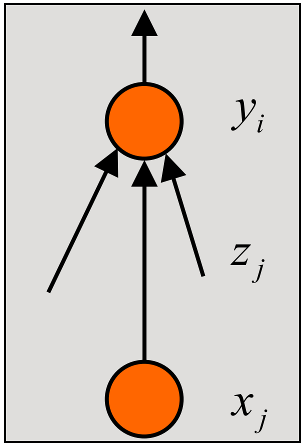
- Advantage
  - Architecture reflects short-context and larger-context memory
  - Very ***robust*** against noise
  - Can be trained with backprop

Recurrent Plausibility Networks (RPN)



[Wermter 1993, 1999]

RPN: Activation and learning



- Units of context layers perform ***time-averaging***

$$C_{n,i}(t) = (1-\varphi_n)H_{n,i}(t-1) + \varphi_n C_{n,i}(t-1)$$

hysteresis value

- Learning: Employ ***back propagation in RPN***:

$$\Delta w_{ij}(t) = \begin{cases} \big(d_j(t) - y_j(t)\big) \cdot f'\big(z_j(t)\big) y_i(t) & \text{if } i \in H(t),\ j \in O(t) \\ \left(\sum_k \delta_k(t) w_{jk}\right) \cdot f'\big(z_j(t)\big) y_i^*(t) & \text{otherwise} \end{cases}$$

$$\text{with } y_i^*(t) = \begin{cases} y_i(t) & \text{if } i \in I(t) \\ y_i(t-1) & \text{if } i \in C(t) \\ \ldots & \ldots \\ y_i(t-t_h) & \text{if } i \in C(t-t_h) \end{cases}$$

for an arbitrary number h of recurrent context layers

$z_j(t) = \sum_l \sum_i w_{ij} y_i(t-l)$, for $l \in (0, \ldots, t_h)$, t_h is maximal time step

Feedforward and SRN Networks are Special Cases of Recurrent Plausibility Network (RPN)

For an arbitrary number n of recurrent context layers:

$z_j(t) = \sum_l \sum_i w_{ij} y_i(t-l)$, for $l \in (0,\ldots,t_h)$, t_h is maximal time step

for context layers for H_h, Unit $j \in L_h$ and $h > 0$, then it holds:

$$\Delta w_{ij}(t) = \begin{cases} \left(d_j(t) - y_j(t)\right) \cdot f'\left(z_j(t)\right) y_i(t) & \text{if } i \in H_{n-1}(t),\ j \in H_n(t) \\ \left(\sum_k \delta_k(t) w_{jk}\right) \cdot f'\left(z_j(t)\right) y_i^*(t) & \text{otherwise} \end{cases}$$

$$\text{with } y_i^*(t) = \begin{cases} y_i(t) & \text{if } i \in H_{h-1}(t) & \text{FF Net} \\ y_i(t-1) & \text{if } i \in C_{h-1}(t-1) & \text{SRN-Net} \\ \ldots & \ldots & \\ y_i(t-t_h) & \text{if } i \in C_{h-1}(t-t_h) & \text{RPN-Net} \end{cases}$$

RPN experiment (Arevian 2007)

- Classification on the Reuters-21578 Corpus
  - **Task**: determine a ***category*** of a news title
  - Dataset of 21578 news with 118 categories
  - *Example*:


```
<REUTERS TOPICS=''YES'' LEWISSPLIT=''TRAIN''
CGISPLIT=''TRAINING-SET'' OLDID=''12981'' NEWID=''798''>
<DATE> 2-MAR-1987 16:51:43 42</DATE>
<TOPICS><D>livestock</D><D>hog</D></TOPICS>
<TITLE>AMERICAN PORK CONGRESS KICKS OFF TOMORROW</TITLE>
<DATELINE> CHICAGO, March 2 - </DATELINE><BODY>The American Pork
Congress kicks off tomorrow, March 3, in Indianapolis with 160
                         ...
trade show, in conjunction with the congress, will feature
the latest in technology in all areas of the industry, the NPPC
added. Reuter
\&\#3;</BODY></TEXT></REUTERS>
```


- Experiment:
  - Train on a sub-set (1,040 titles)
  - Test on 9,663 titles

RPN experiment: Classification results

Method	Mean performance for 50 networks (%)		
	Recall	Precision	F_1 Measure
Randomised	92.72	92.12	92.42
Original Corpus	92.59	91.73	92.16
Reversed Original	92.26	91.39	91.83

$$precision = \frac{tp}{tp + fp} \qquad recall = \frac{tp}{tp + fn}$$

$$F_{score} = \frac{(1 + N^2) \cdot precision \cdot recall}{precision + (N^2 \cdot recall)}$$

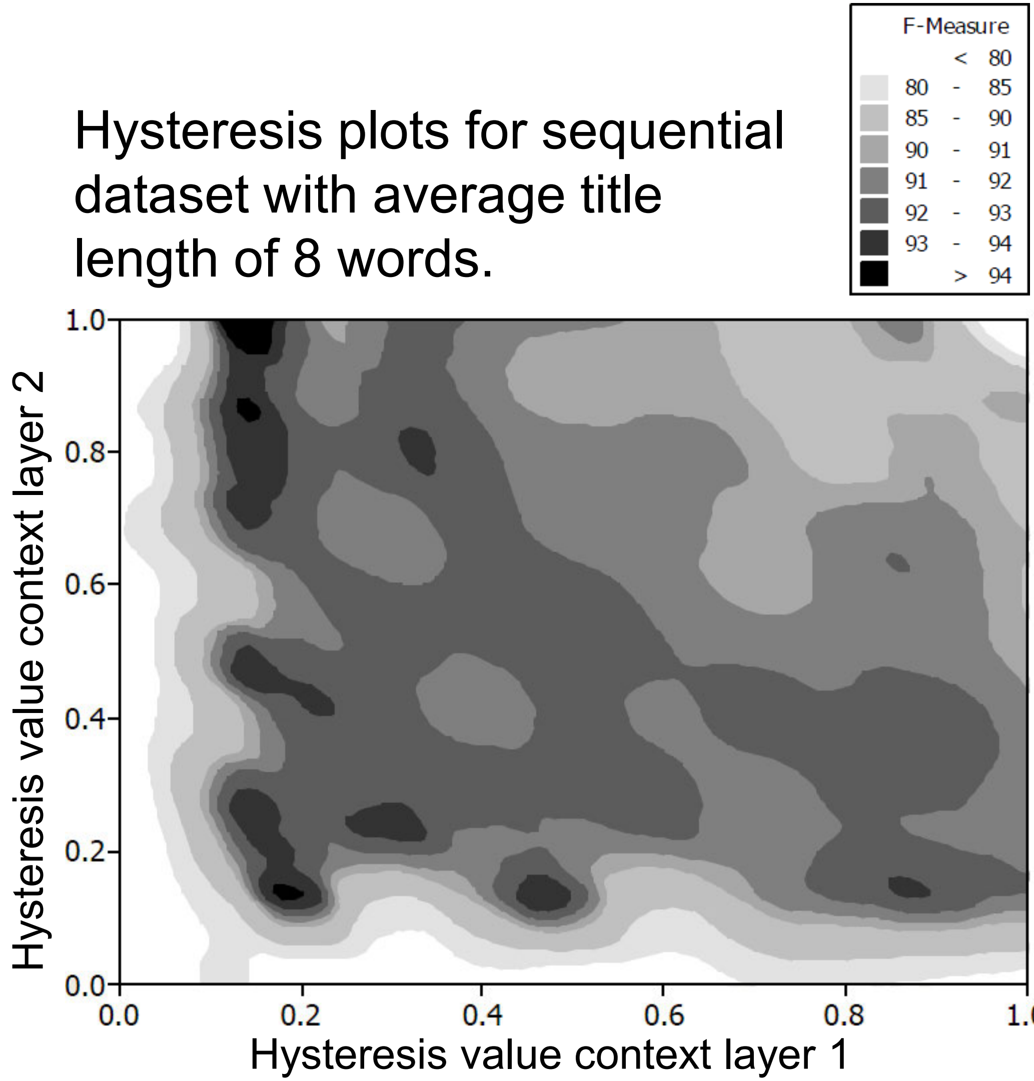
F_1 Measure : F_{score} with $N = 1$

How to determine Hysteresis parameters?

- Depending on the problem!
  - Good choice: ***smaller*** values for ***first context*** layers, ***higher*** values for ***second context*** layers

- On Reuters corpus on average:
  - $\varphi(C_1) = 0.2$
  - $\varphi(C_2) = 0.7$

Hysteresis plots for sequential dataset with average title length of 8 words.



RPN experiment: Adding some noise

Method	Mean performance for 50 networks (%)		
	Recall	Precision	F_1 Measure
Randomised	92.72	92.12	92.42
Original Corpus	92.59	91.73	92.16
Reversed Original	92.26	91.39	91.83
Noise Factor 2	92.39	91.63	92.01
Noise Factor 4	91.28	90.37	90.82
Noise Factor 6	86.40	85.63	86.01

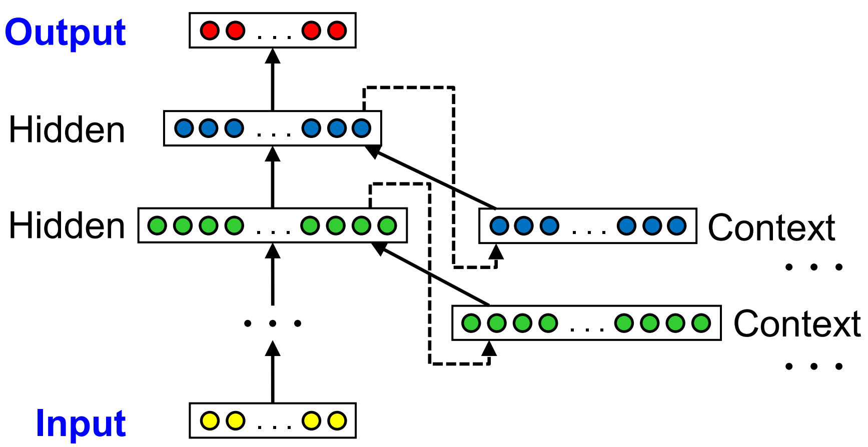
Noise: Introduce ***stop-words*** at ***random*** ⇨ Increase length of titles from e.g. 8 words to 16 (x2), 32 (x4) or 48 (x6) words

Adding noise leads to graceful degradation!

RPN experiment summary

- Recurrent Plausibility Networks (RPN) can better capture the important context no matter whether relevant words occur in the beginning or the end of a sequence
⇨ Local context / local word order is less important

- Noise robustness leads to good classification results for potentially disturbed sequences



- Hysteresis values can tune the reach-out of the context units

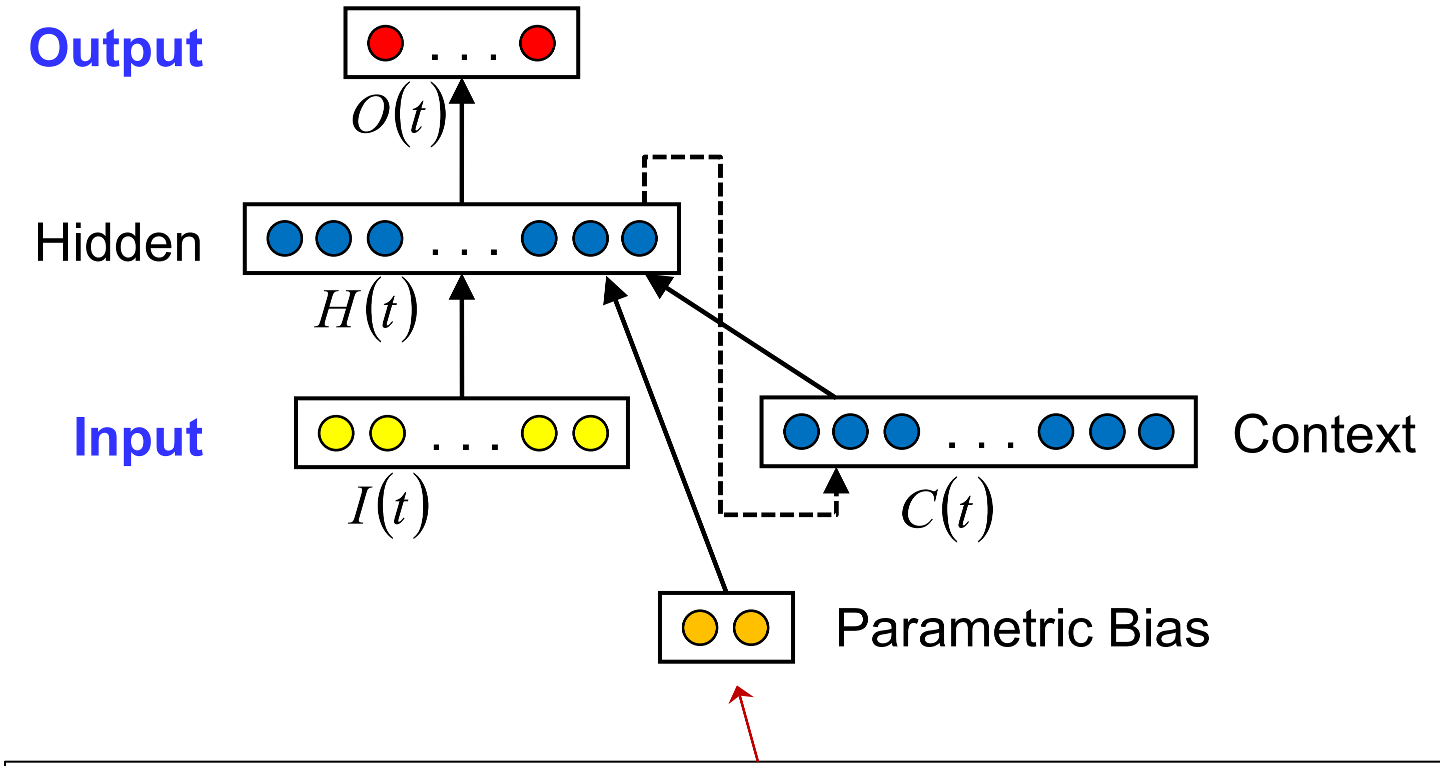
2. RNN Extension: Parametric Bias

- Characteristics
  - Additional nodes which self organise a ***bias*** for a sequence
  - ***Continuous input*** to the network for all time steps

  Example:
  - Recurrent Neural Network with Parametric Bias (PB)

- Advantage
  - Using alternated PB values can generate alternative but still meaningful sequences
  - Network generates ***nonlinear mappings*** between the parametric ***bias*** and corresponding ***sequences***

Recurrent Neural Network with Parametric Bias



[Tani 2003]

The same ***bias*** is influencing the activation in every time step, but is ***self-organised*** by backprop for every sequence.

RNNPB learning

- Extended ***back propagation through time*** (BPTT)
  - Weights are updated based on determined deltas:

$$w_{ij}^{(n+1)} = w_{ij}^{(n)} + \eta \cdot \Delta w_{ij} \qquad \Delta w_{ij} = \sum_t x_{t,j} \cdot \delta_{i,j}$$

  - The PB nodes are updated based on the accumulated gradient over a time window l

$$\rho_i^{(n+1)} = \rho_i^{(n)} + \gamma \cdot \sum_{k=t-\frac{l}{2}}^{t+\frac{l}{2}} \delta_{i,k}^{(\mathrm{PB})}$$

large time windows: ***general characteristic*** of a sequence, *small window:* ***repetitive characteristics*** of a sequence

  - Learning rate can be fixed or adaptive

$$\gamma_i \propto \frac{1}{l} \cdot \left\| \sum_{k=t-\frac{l}{2}}^{t+\frac{l}{2}} \delta_{i,k}^{(\mathrm{PB})} \right\|$$

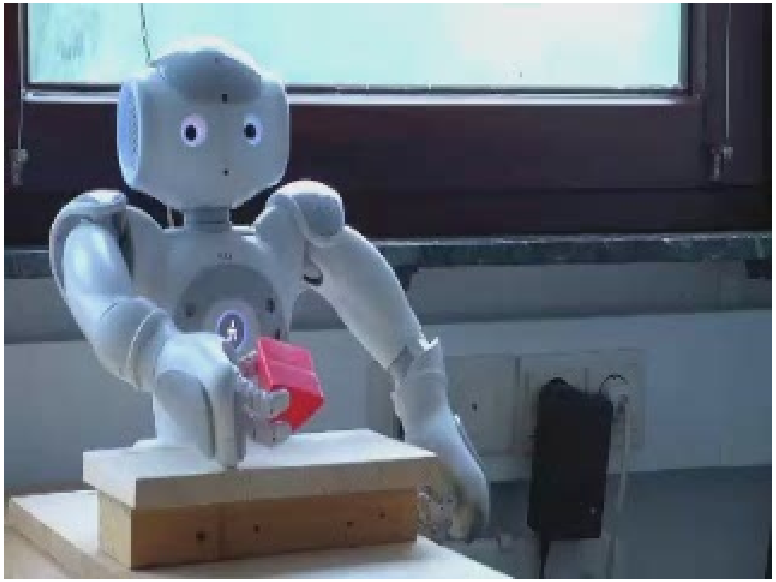
Adaptive learning rate: scaled proportional to the absolute mean gradient

RNNPB experiment (Kleesiek 2011)

- Humanoid Robot NAO ***perceive actively*** different objects

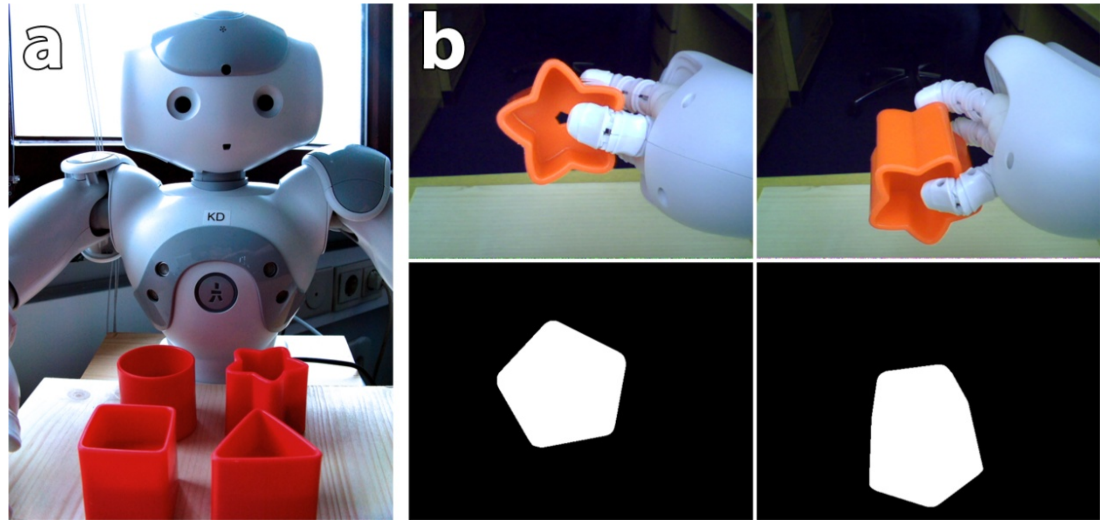
- **Task**: identify the object held in the hand
- Approach: ***experience*** the visual and sensori-motor experience ***over time***.

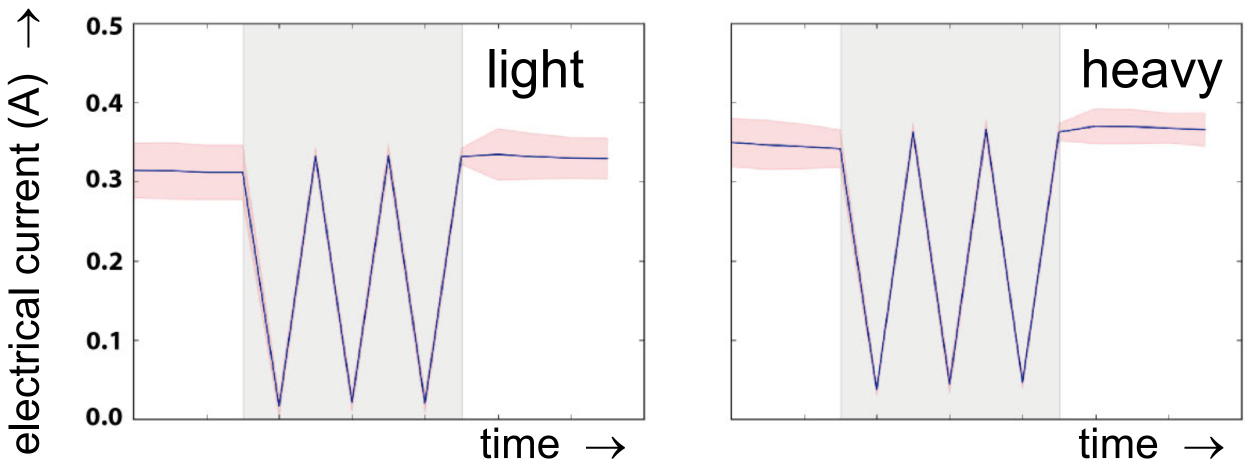
- Mean time series used for training the RNNPB to ***recognise*** and ***generate*** experience



RNNPB experiment: Data aquisition

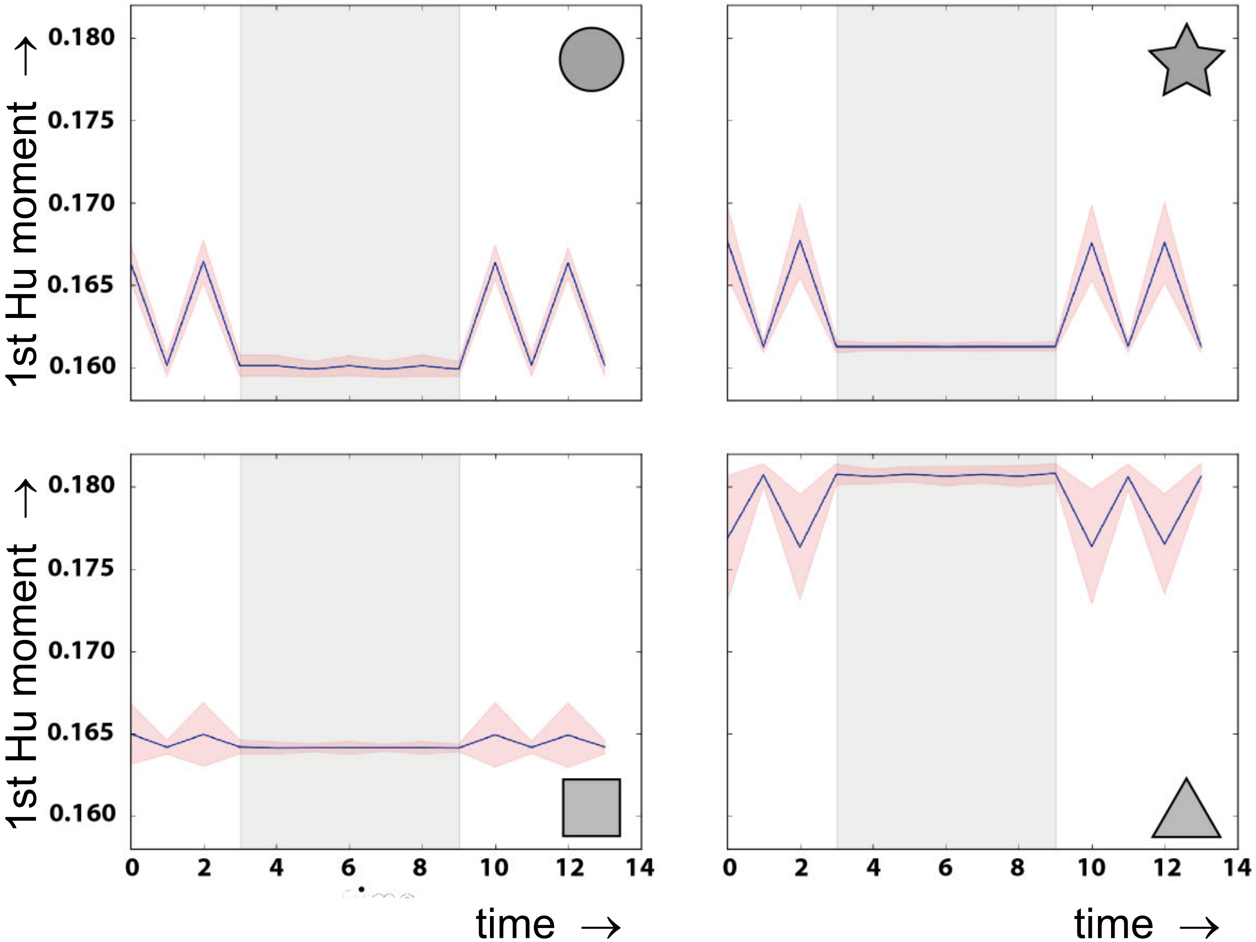
- 8 objects:
  - 4 different shapes
  - 2 different weights





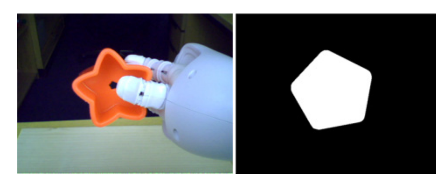
Weight is experienced with the arm (current on the servo)

RNNPB experiment: Data acquisition (cont.)



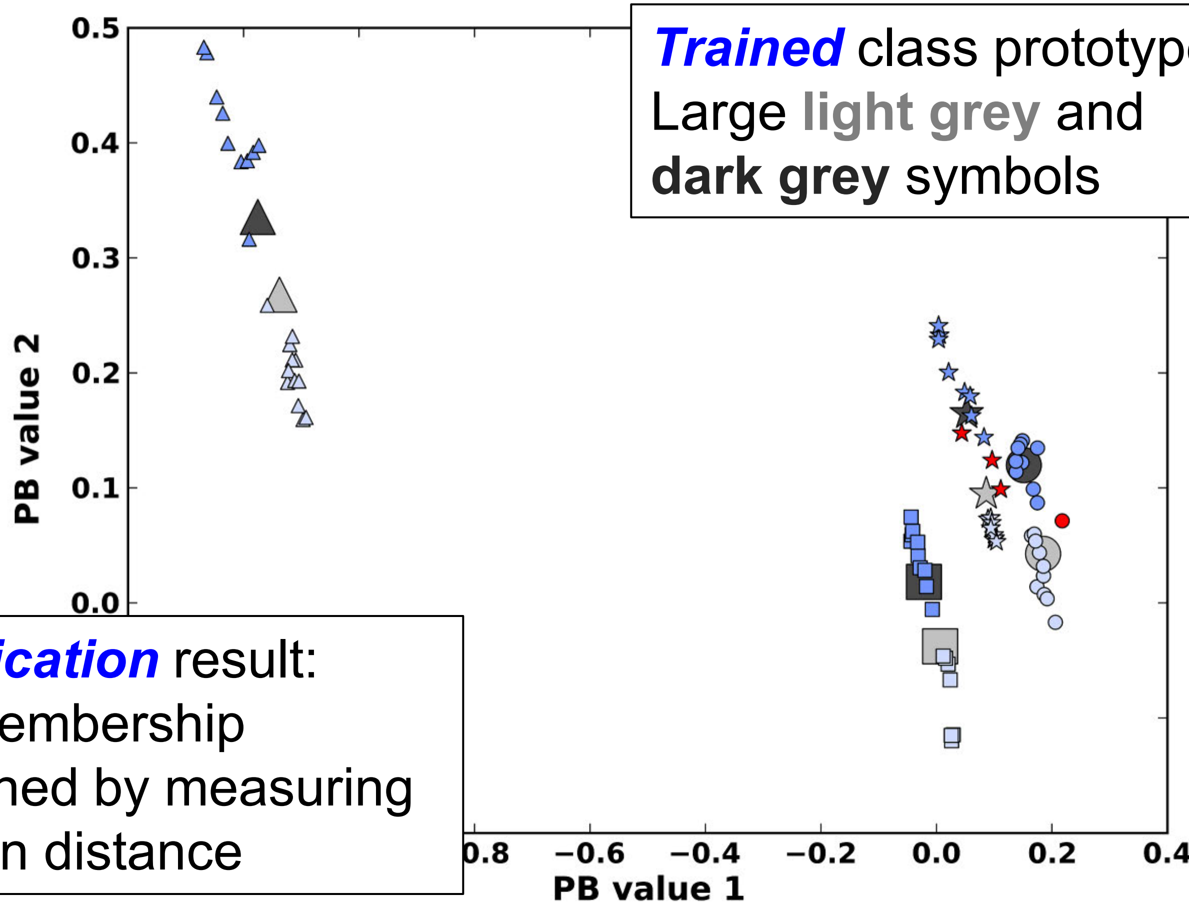
Appearances experienced with the camera:

- Thresholding
- Determining convex hull
- Extracting contour: First Hu moment



RNNPB experiment: Results

- Experiment 1:
  Classification using all object categories for training:

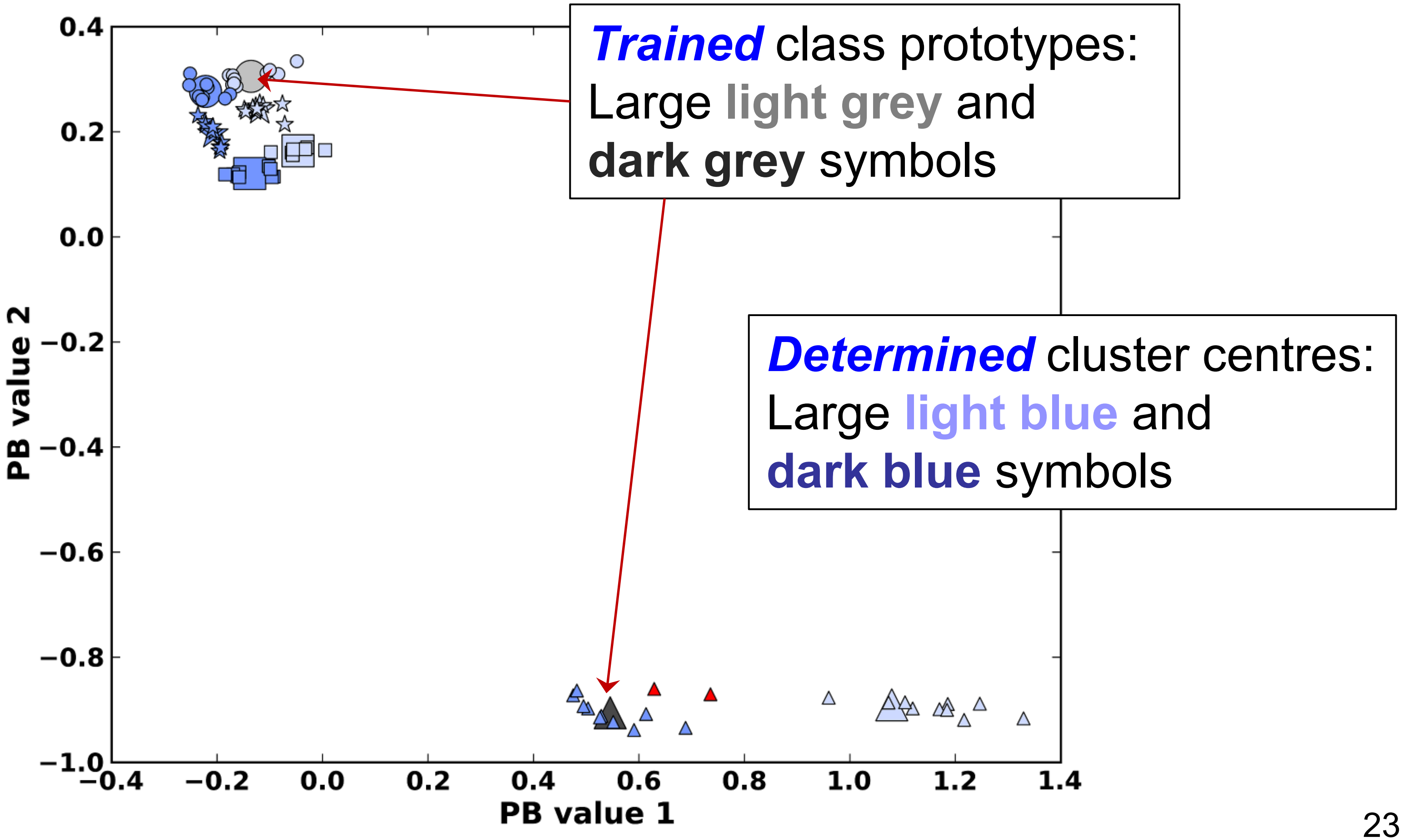


Trained class prototypes:
Large **light grey** and
dark grey symbols

Classification result:
class-membership
determined by measuring
Euclidian distance

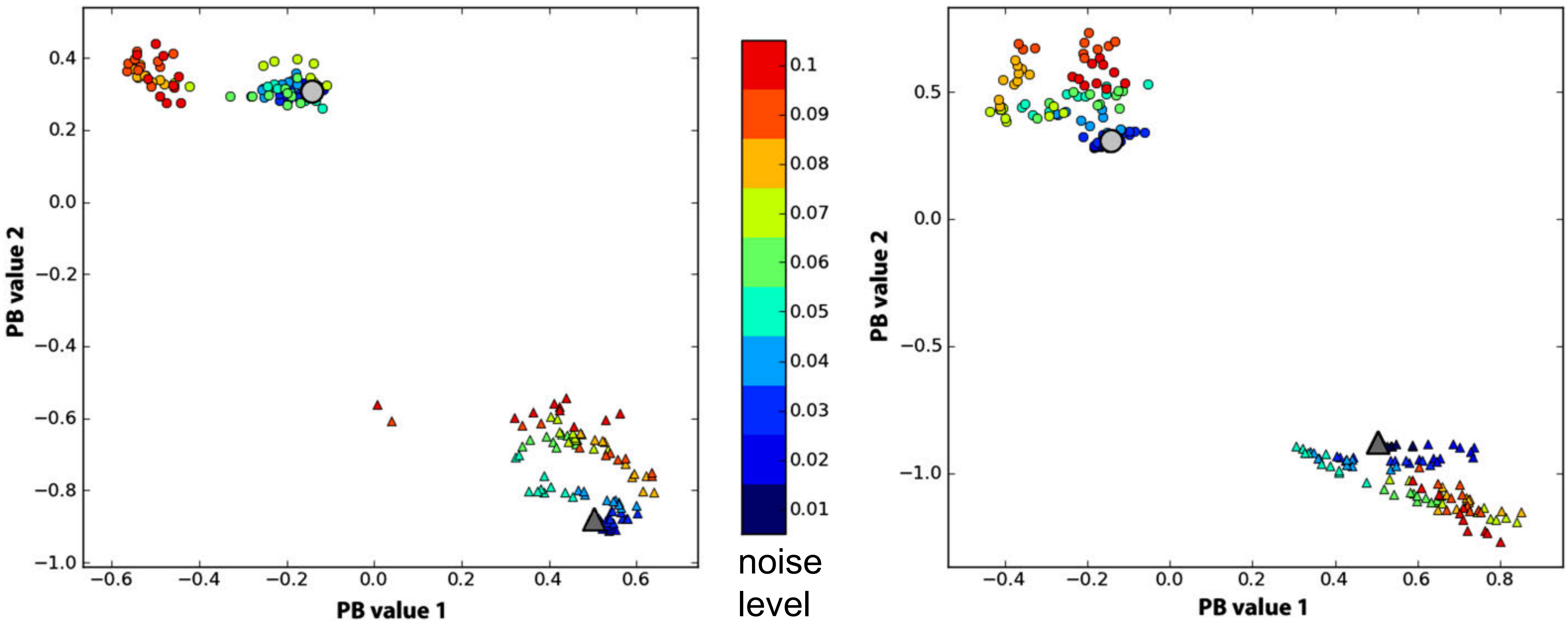
RNNPB experiment: Results

- Experiment 2: Classification using only the light circular-shaped and the heavy triangular-shaped object for training:



RNNPB experiment: Results

- Analysis: Noise tolerance within and across modalities:

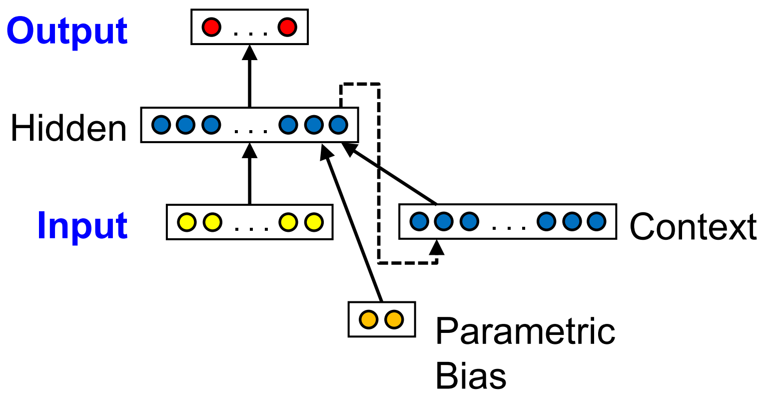


Uni-modal noise tolerance (only vision time series)

Bi-modal noise tolerance (vision and sensori-motor time series)

RNNPB experiment summary

- Parametric bias plays a role of ***behaviour modulator*** sent by a higher level as keys for behaviour
- PB nodes can capture small repetitive parts or general characteristics of a sequence
- The network with PB nodes is very robust against noise in time series after learning



3. RNN Extension: Multiplicity of time

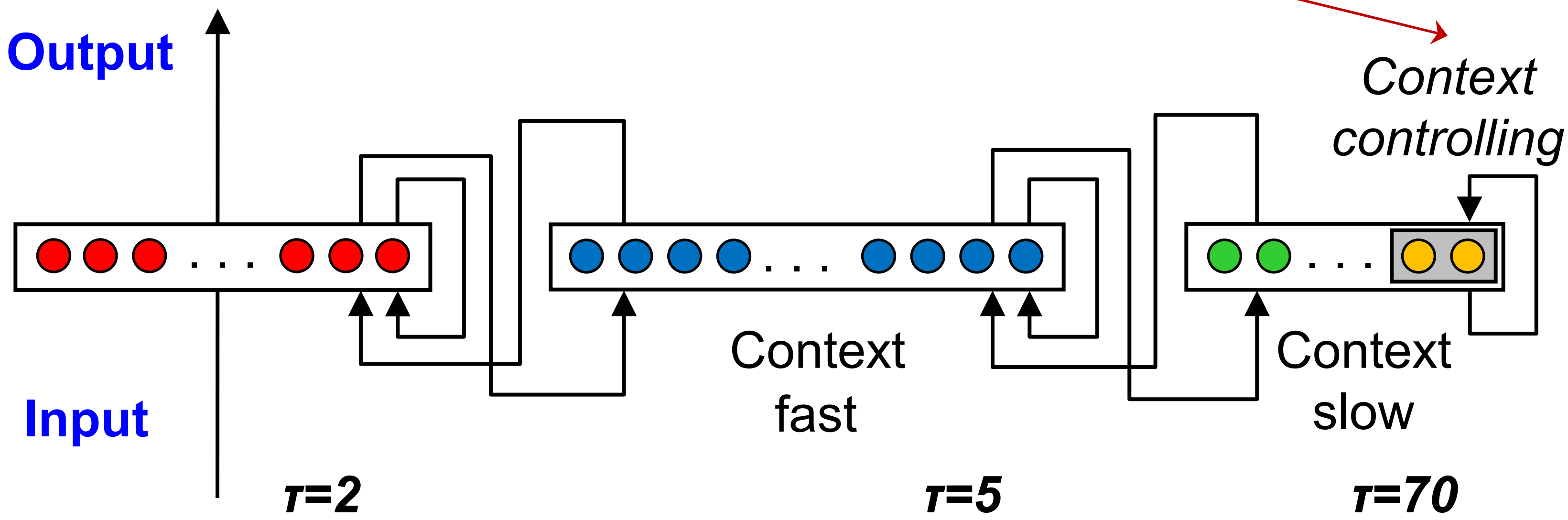
- Characteristics:
  - Multiple context layers with ***different timescales***
  - *Context controlling* nodes that bias the sequence

  Example:
  - Multiple Timescale Recurrent Neural Network
  - Very related to PRN with hysteresis concept (!)

- Advantages:
  - Self organising of different aspects of the sequences
  - Hierarchy of dynamics can emerge
  - Similar to PB nodes, the *context controlling* nodes can generate a whole sequence

Multiple Timescale Recurrent Neural Network

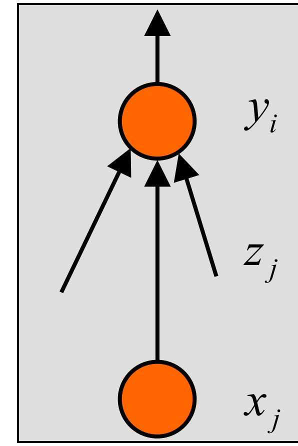
Context controlling nodes (time step **0**) can determine the whole sequence ⇨ ***bias***



[Yamashita 2008]

Timescale: ***1/τ*** of the recurrent activation is taken into account while (1-1/τ) is kept from the last timestep

MTRNN activation functions



- Activation value of the *i*th neuron at step t:

output activity

$$y_{t,i} = \begin{cases} \dfrac{\exp(z_{t,i} + b_i)}{\sum_{j \in I_{\mathrm{IO}}} \exp(z_{t,j} + b_j)} & i \in \text{Input layer } I_{\mathrm{IO}} \\ \dfrac{1}{1 + \exp(-(z_{t,i} + b_i))} & i \notin I_{\mathrm{IO}} \end{cases}$$

softmax function

logistic function

summed input

$$z_{t,i} = \begin{cases} 0 & \text{if } t = 0 \wedge i \notin I_{\mathrm{CSC}} \\ \mathrm{Csc}_{0,i} & \text{if } t = 0 \wedge i \in I_{\mathrm{CSC}} \\ \dfrac{1}{\tau_i} \sum_{j \in I_{\mathrm{all}}} w_{ij} \cdot x_{t,j} + \left(1 - \dfrac{1}{\tau_i}\right) z_{t-1,j} & \text{otherwise} \end{cases}$$

Initial state of *context controlling* nodes

time constant τ

input activity

$$x_{t,i} = \begin{cases} (1-\psi) y_{t-1,i} + (\psi) d_{t-1,i} & t \geq 1 \wedge i \in I_{\mathrm{IO}} \\ y_{t-1,i} & t \geq 1 \wedge i \notin I_{\mathrm{IO}} \end{cases}$$

teacher forcing

MTRNN learning algorithm

- Variant of ***real-time back propagation through time*** (BPTT)
  - Determine the deltas based on the partial derivatives:

$$\Delta w_{ij} = \frac{1}{\tau_i} \cdot \sum_t x_{t,j} \frac{\partial E}{\partial z_{t,i}}$$

error derivate

  - Weights and biases are updated as usual:

$$w_{ij}^{(n+1)} = w_{ij}^{(n)} - \eta \cdot \Delta w_{ij}$$

$b_i^{(n+1)}$ analog

  - Update also the initial state of *context controlling* nodes at time step **0**

learning rates

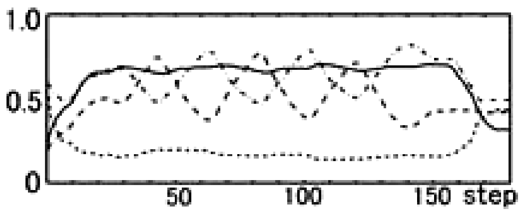
$$Csc_{0,i}^{(n+1)} = Csc_{0,i}^{(n)} - \varsigma \cdot \Delta Csc_i \qquad \Delta Csc_i = \frac{\partial E}{\partial z_{\mathbf{0},i}}$$

Self organising of *context controlling* nodes

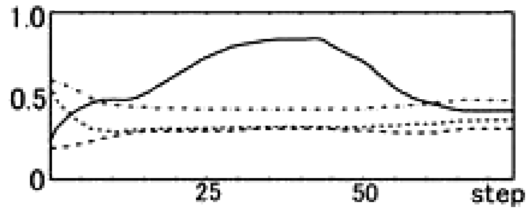
MTRNN experiment A (Yamashita 2008)

- Trained ***motor sequences***
  and test for the ***emergence*** of a hierarchy
  - Learn ***primitives***:

backward-forward

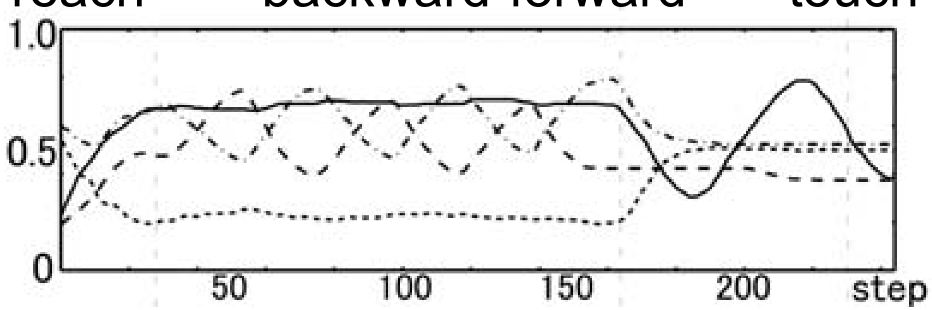


touch



  - Learn ***combinations*** of primitives

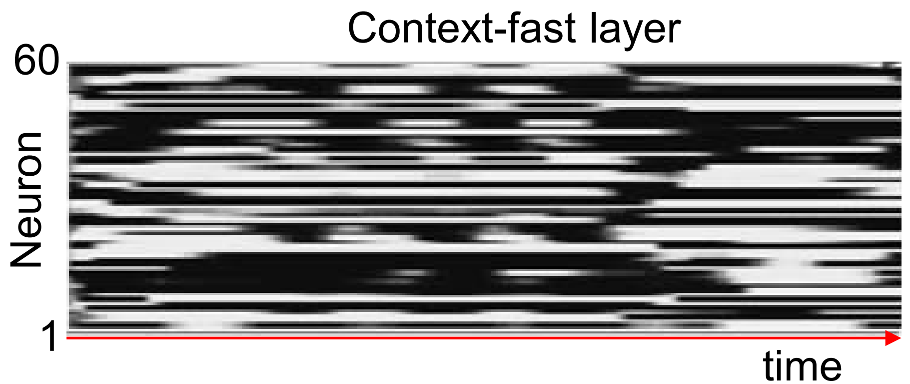
reach backward-forward touch



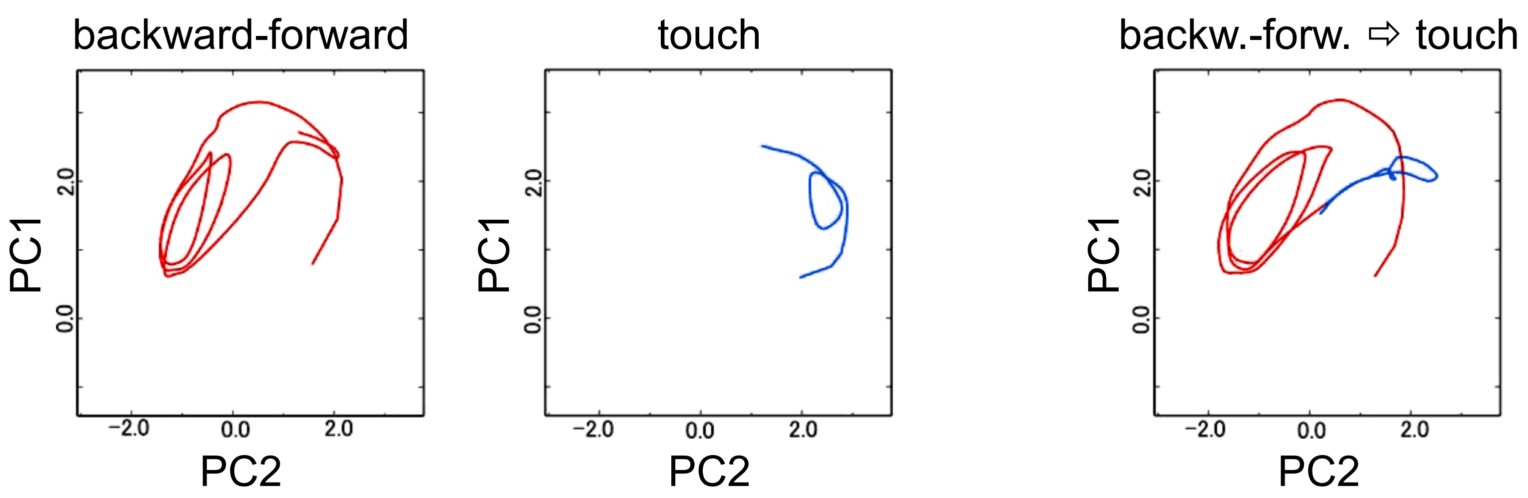
Figures:
[Yamashita et al., 2008]

MTRNN experiment A: Analysis

- Question: How does the network ***self-organize***?
- Approach: Run a ***Principle Component Analysis*** on the neural activity



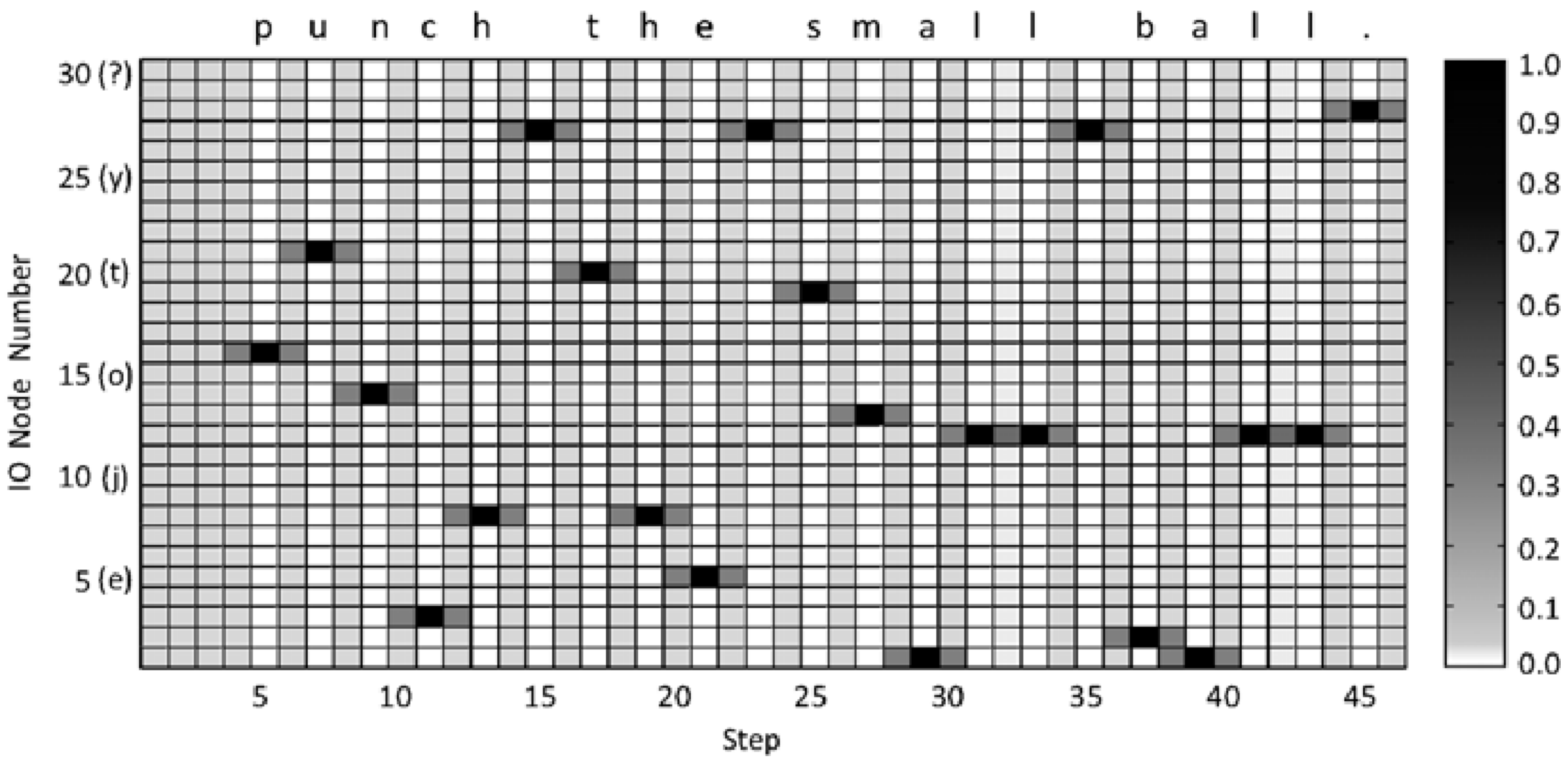
- Result: Emergence of a ***functional hierarchy***



MTRNN experiment B (Hinoshita 2011)

- Trained ***sentences*** and tested for emulation and recognition

- Sentence represented as a ***sequence*** of ***symbols***:



MTRNN experiment B: The language

Lexicon.

Category	Nonterminal symbol	Words
Verb (intransitive)	V_I	jump, run, walk
Verb (transitive)	V_T	kick, punch, touch
Noun	N	ball, box
Article	ART	a, the
Adverb	ADV	quickly, slowly
Adjective (size)	ADJ_S	big, small
Adjective (color)	ADJ_C	blue, red, yellow

Regular grammar.

S $\rightarrow$ V_I	NP $\rightarrow$ ART N	ADJ $\rightarrow$ ADJ_S
S $\rightarrow$ V_I ADV	NP $\rightarrow$ ART ADJ N	ADJ $\rightarrow$ ADJ_C
S $\rightarrow$ V_T NP		ADJ $\rightarrow$ ADJ_S ADJ_C
S $\rightarrow$ V_T NP ADV		

List of sentences.

Number	Sentence
001	"jump slowly."
002	"punch the small ball."
003	"run quickly."
004	"punch the ball quickly."

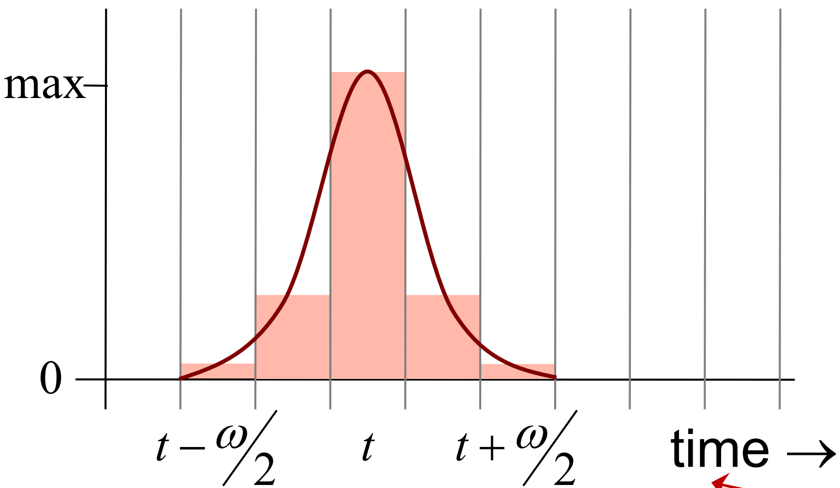
MTRNN experiment B: Sentence encoding

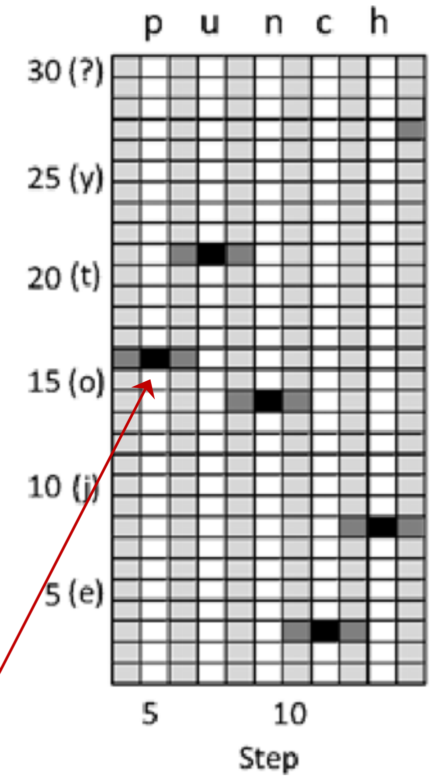
- Every symbol represents an IO activation

desired output activity

$$y^*_{t,i} = \frac{\exp(z^*_{t,i})}{\sum_{j \in I_{IO}} \exp(z^*_{t,j})}$$

desired summed input $z^*_{t,i}$:





$$g = \lambda \cdot \exp\left(\frac{-r^2}{2\sigma^2}\right)$$

Input symbols are fed in with a ***Gaussian*** g over several time steps ω scaled by λ to [0.0,max]

MTRNN experiment B: Results

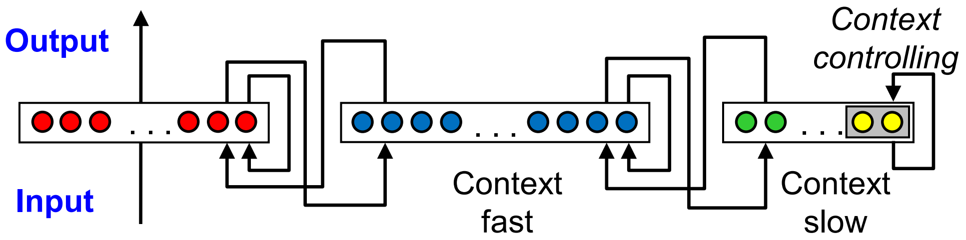
- From a learned network sentences can be
  - ***generated*** (emulated)
    Using a Csc_0 state, calculated from the desired sentence
  - ***corrected***
    Using a Csc_0 state, calculated from the corrupted sentence

Results of cross validation.

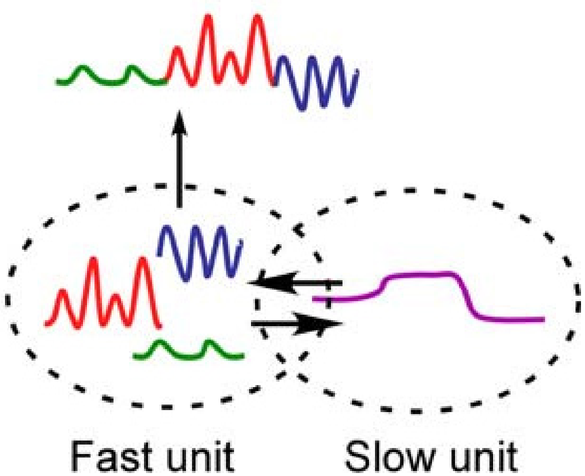
Sentences for validation	Emulation task	Correction task
001–020	95/100	84/100
021–040	100/100	82/100
041–060	96/100	78/100
061–080	96/100	81/100
081–100 (above-mentioned)	98/100	83/100

MTRNN experiment B: Analysis

Interesting observation:

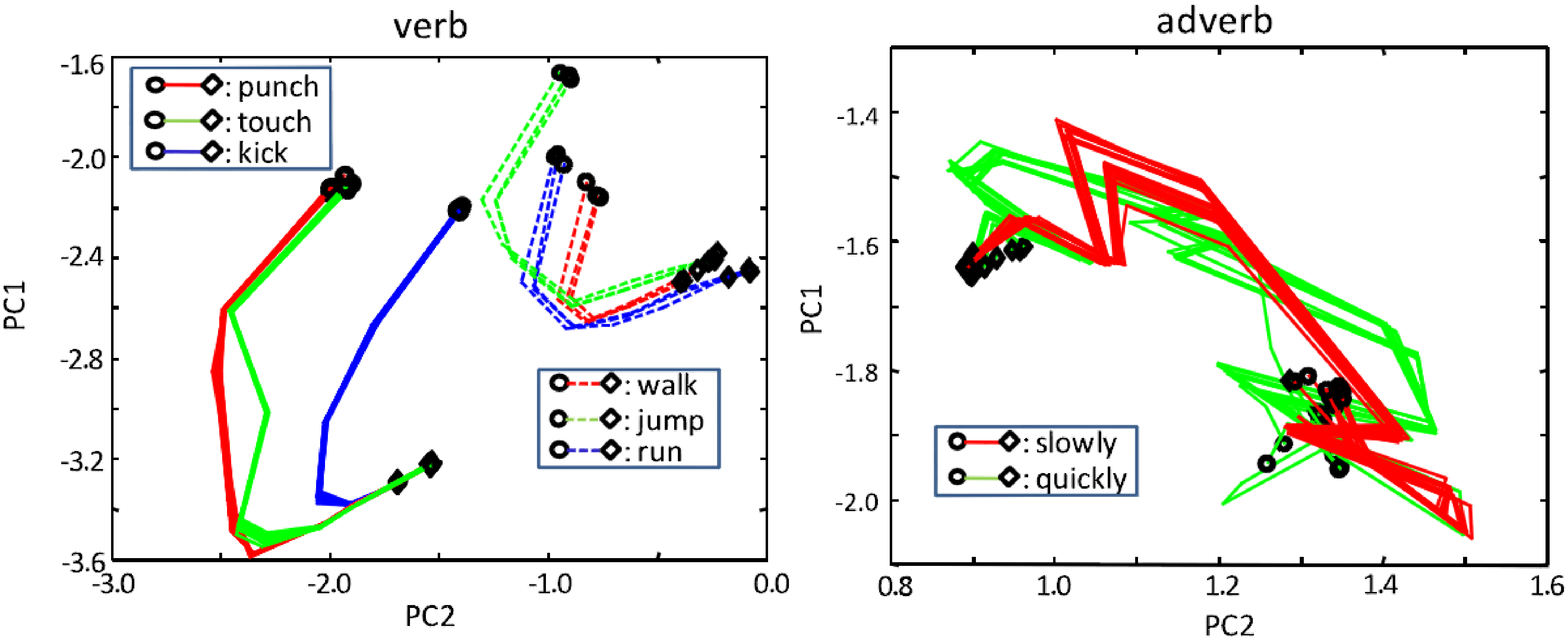


- Linguistic ***hierarchy*** emerges in the network:
  - Word representations in the Cf
  - Sentence representations in the Cs
- Linguistic structure can produce sentences from the ***inferred*** grammar.
  - Even if they where not learned explicitly!



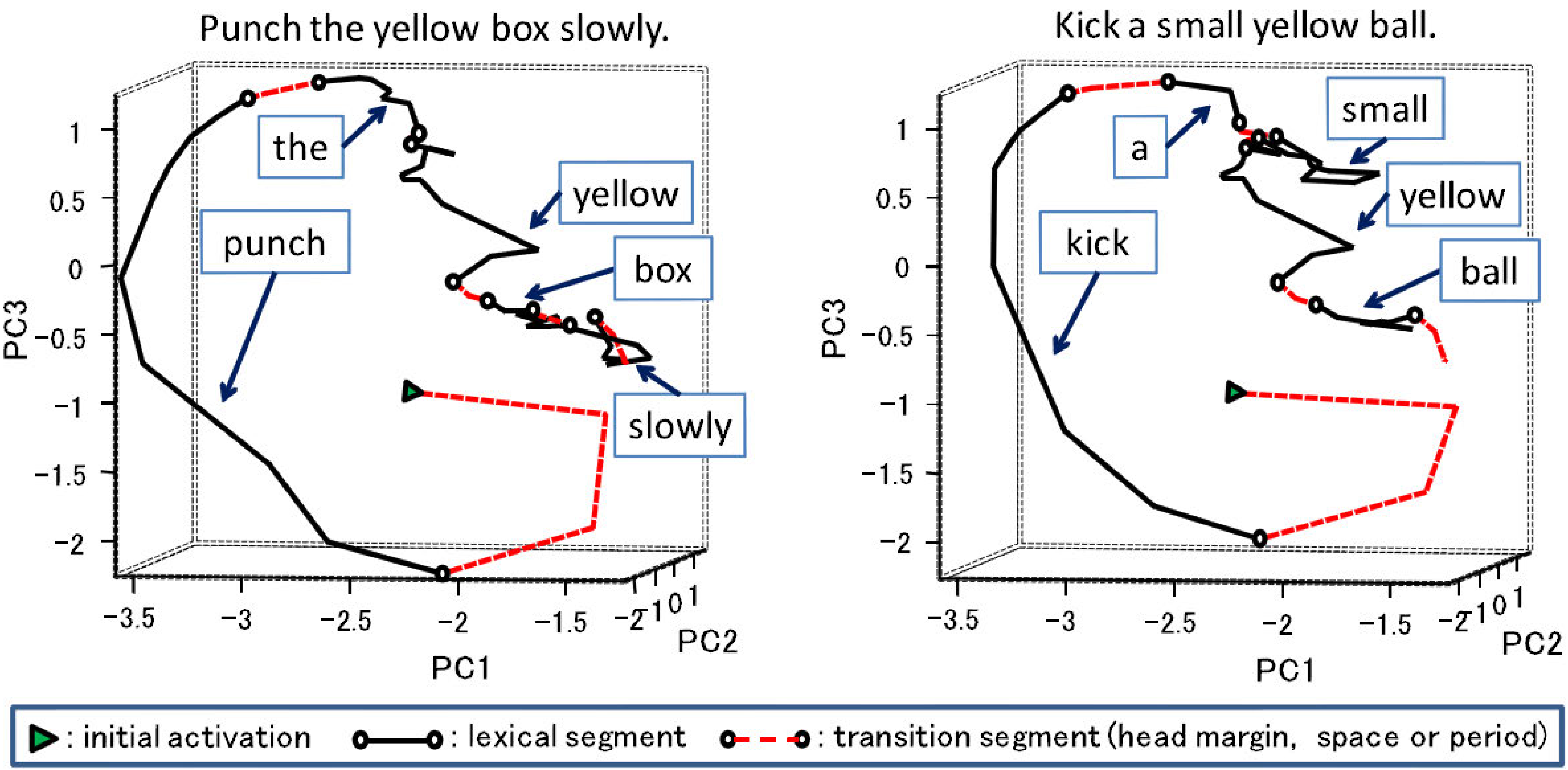
MTRNN analysis example: Words

- Trajectories over time with Principal Component Analysis



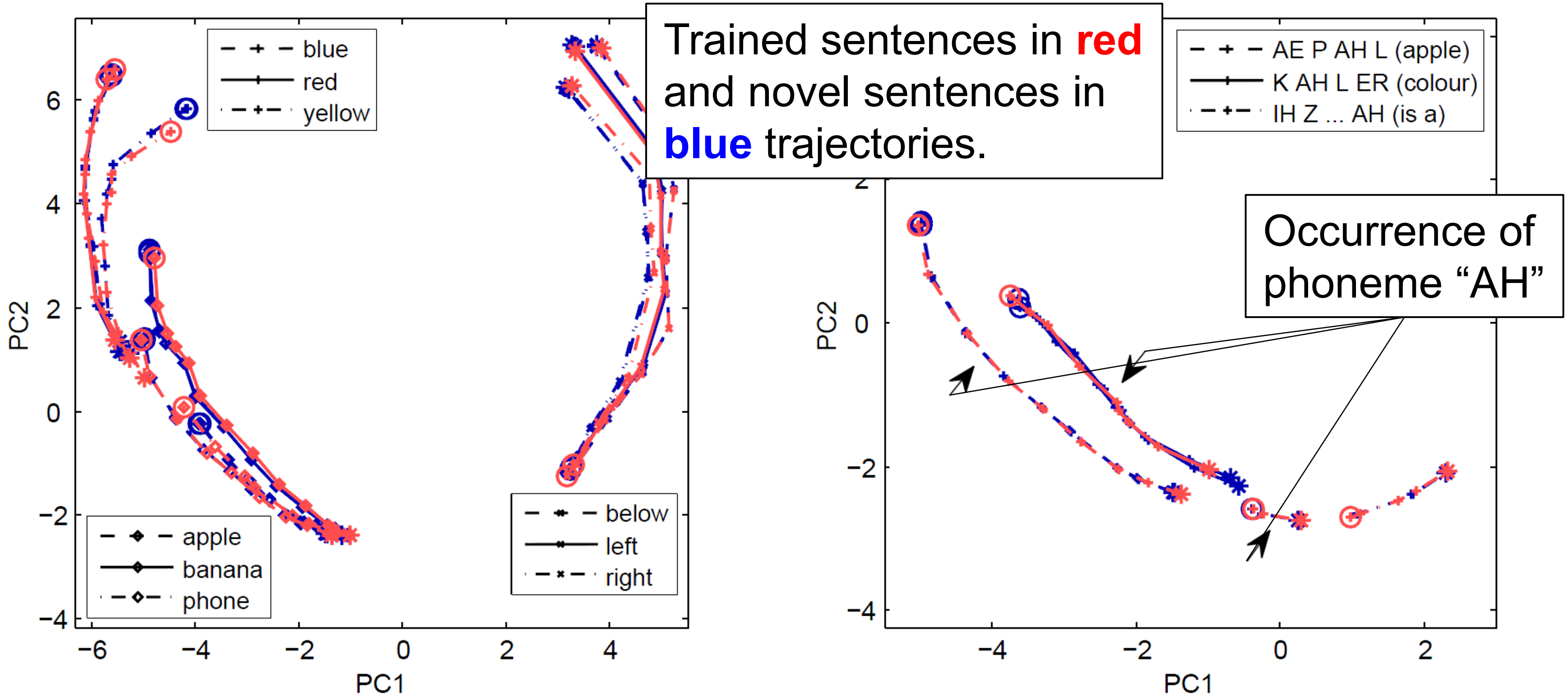
  - Same words have nearly identical trajectories
  - Words in the same categories have similar trajectories

MTRNN analysis example: Sentences



Extended MTRNN analysis: Generalisation

- Tested with novel sentences that stem from the same grammar

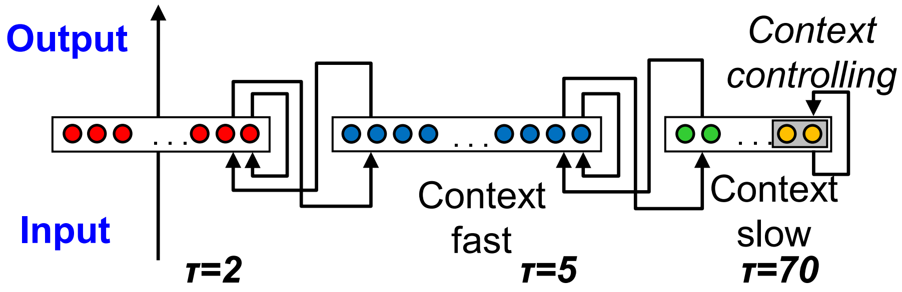


  - Similar pattern for same words in trained and untrained sentences
  - No similar pattern for worlds with similar phonetic representation

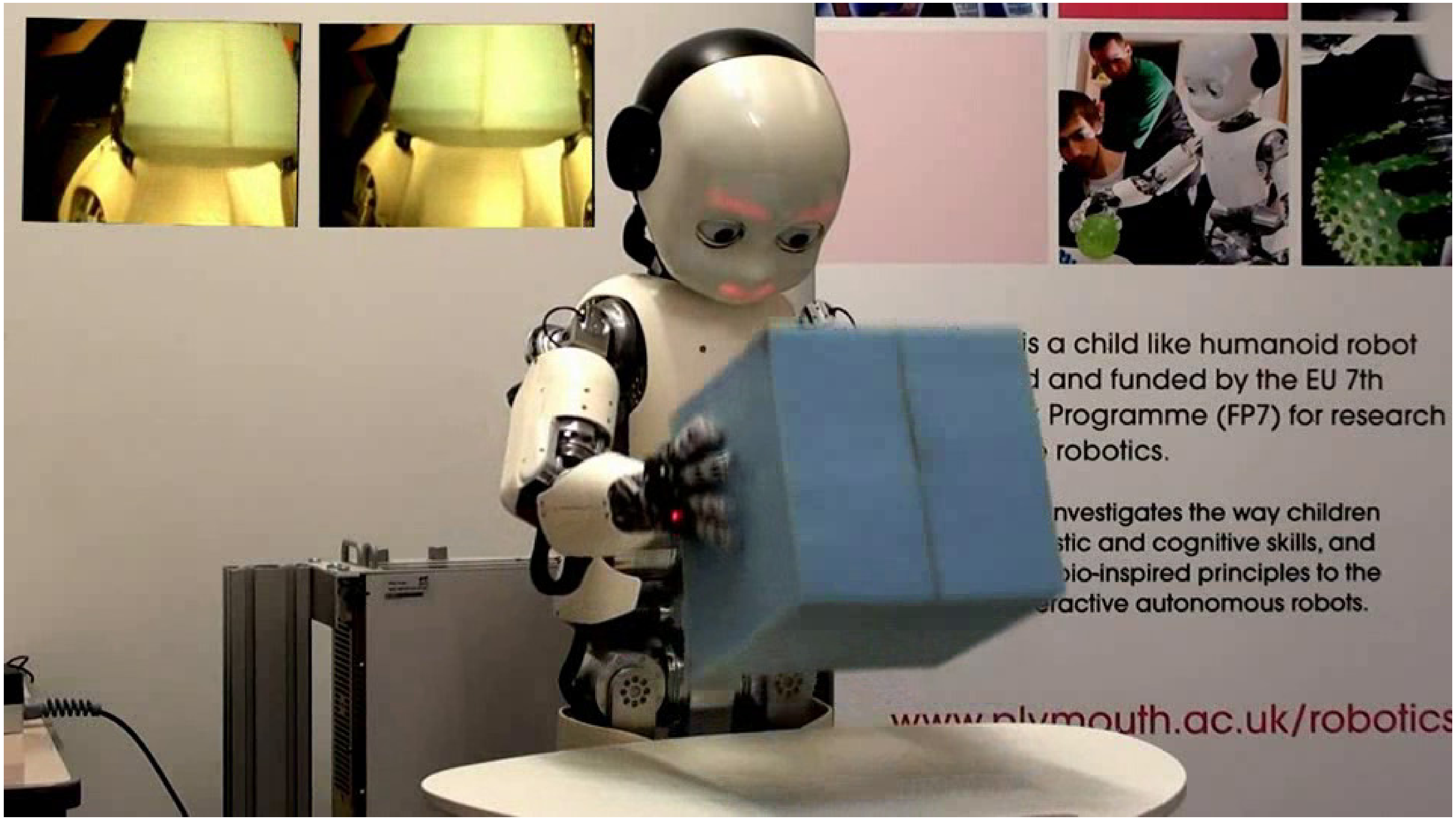
[Heinrich et al., ICANN 2013]

MTRNN experiment summary

- ***Deterministic*** recurrent neural network
- Can recognize, generate and correct sequences
- ***Self-organizing*** internal hierarchical ***structure***
- Uses fast and slow adapting context nodes
- Issue: BPTT difficult to calculate in real time
- Advantages:
  - Can correct substitution errors in sentences
  - Linguistic structure can emerge



Recap: Multiple Timescale RNN for movements on an ICub humanoid robot



http://www.italkproject.com

Summary

- Recurrent Neural Networks with different extensions are ***efficient neural methods*** for various tasks and
  - Can make use of ***more context*** information
  - Can approximate key patterns of time-series/sequences
  - Can ***self organise*** to inherent hierarchies of the information
- Advanced RNNs can be trained with adaptations of well researched algorithms, e.g. back-propagation
- Offer high degree of ***noise robustness*** – even to significant disturbations in the sequences
- Allow ***general neural architectures*** to be developed

Further reading

- Wermter S., Panchev C. Arevian G. Hybrid Neural Plausibility Networks for News Agents. *Proceedings of the National Conference on Artificial Intelligence,* AAAI*.* pp. 93-98, Orlando, USA, July 1999.
- Arevian, G. Recurrent Neural Networks for Robust Real-World Text Classification. *IEEE/WIC/ACM International Conference on Web Intelligence WI07,* pp. 326-329, 2007.
- Kleesiek, J., Badde, S., Wermter, S., Engel, A.K. What Do Objects Feel Like? - Active Perception for a Humanoid Robot. *Proceedings of the 4th International Conference on Agents and Artificial Intelligence (ICAART 2012)*, Vol. 1, pp. 64-73, Vilamoura, Portugal, January 2012.
- Hinoshita, W., Arie, H., Tani, J., Okuno, H.G. & Ogata, T. Emergence of hierarchical structure mirroring linguistic composition in a recurrent neural network. *Neural Networks* 24, pp. 311-320, 2011.