Using stigmergy as a computational memory in the design of recurrent neural networks

Federico A. Galatolo

Department of Information Engineering
University of Pisa

20 February 2019

Time-Series Static Classification

X_0	X_1	X_2	X_3	$\cdots$	X_{n-2}	X_{n-1}

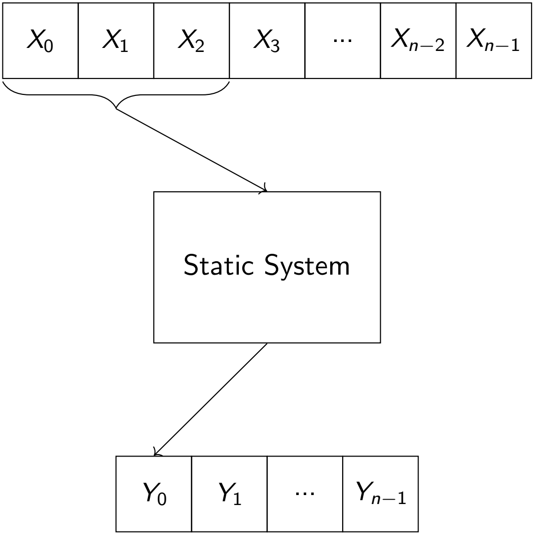
- MLP
- CNN
- ...

Static System

Y_0	Y_1	$\cdots$	Y_{n-1}

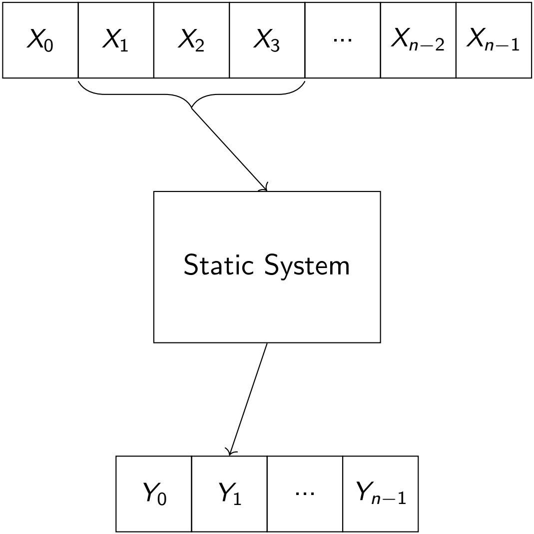
Time-Series Static Classification

- MLP
- CNN
- ...



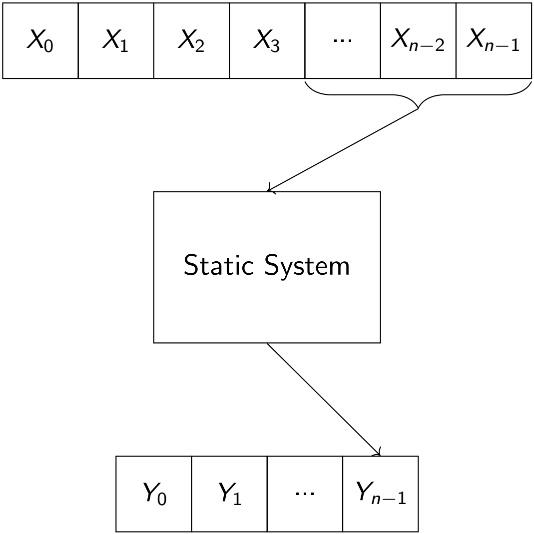
Time-Series Static Classification

- MLP
- CNN
- ...



Time-Series Static Classification

- MLP
- CNN
- ...



Time-Series Static Classification

- ✓ You can use any existing ML Architecture
- × Window size choice
- × Long-lived relationships are impossible to infer

Time-Series Dynamic Classification

X_0	X_1	X_2	X_3	$\cdots$	X_{n-2}	X_{n-1}

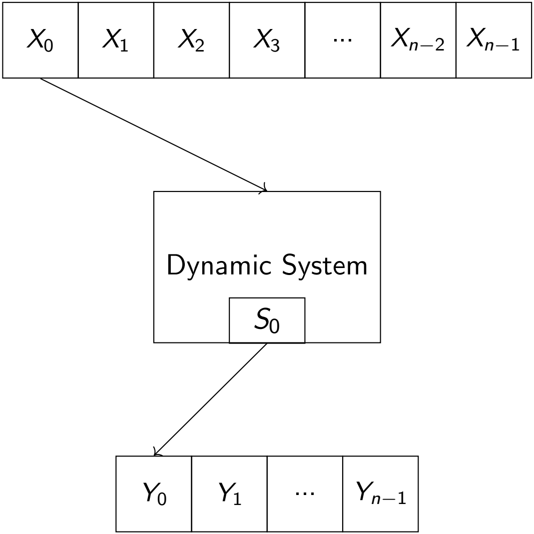
- RNN
- LSTM
- ...

Dynamic System

Y_0	Y_1	$\cdots$	Y_{n-1}

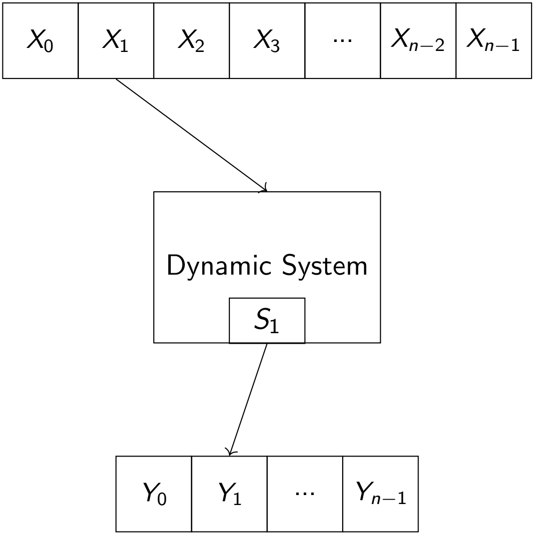
Time-Series Dynamic Classification

- RNN
- LSTM
- ...



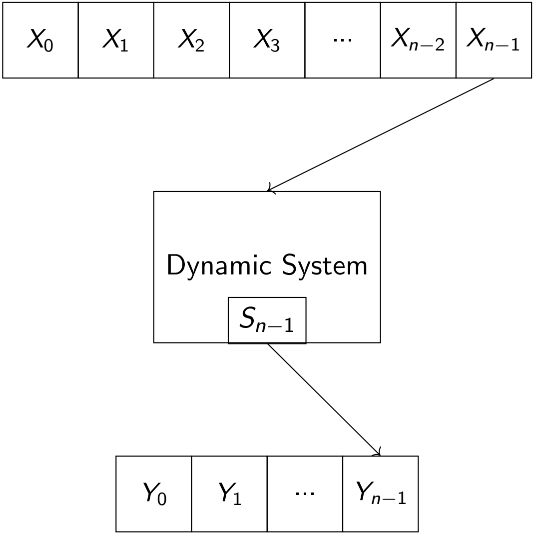
Time-Series Dynamic Classification

- RNN
- LSTM
- ...



Time-Series Dynamic Classification

- RNN
- LSTM
- ...

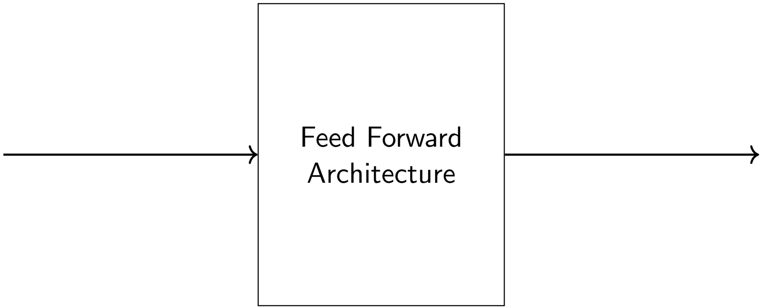


Time-Series Dynamic Classification

- ✓ The system knows the concept of time
- ✓ Can autonomously decide what to remember and forget
- × Ad-Hoc solutions
- × Highly engineered

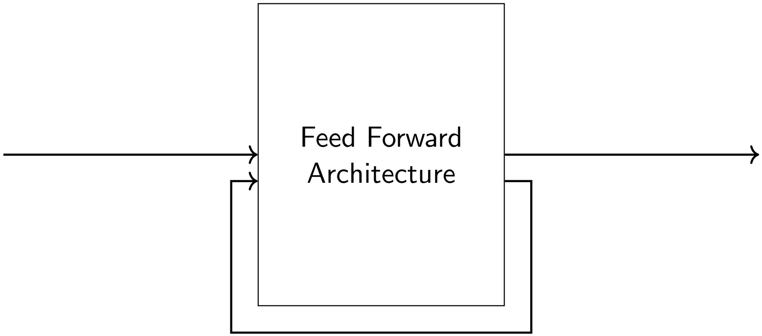
RNN $\neq$ LSTM

- RNN and LSTM are often used as synonyms in literature
- Has been proven that “Vanilla recursion” performs poorly
- LSTM are the state of the art for Time Series Classification



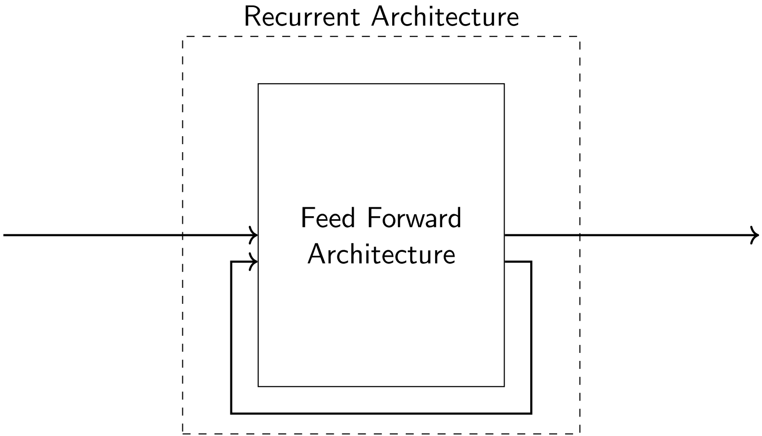
RNN $\neq$ LSTM

- RNN and LSTM are often used as synonyms in literature
- Has been proven that "Vanilla recursion" performs poorly
- LSTM are the state of the art for Time Series Classification

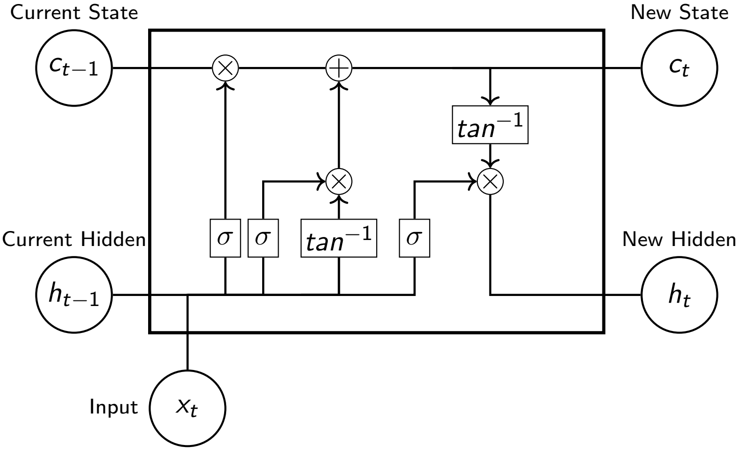


RNN $\neq$ LSTM

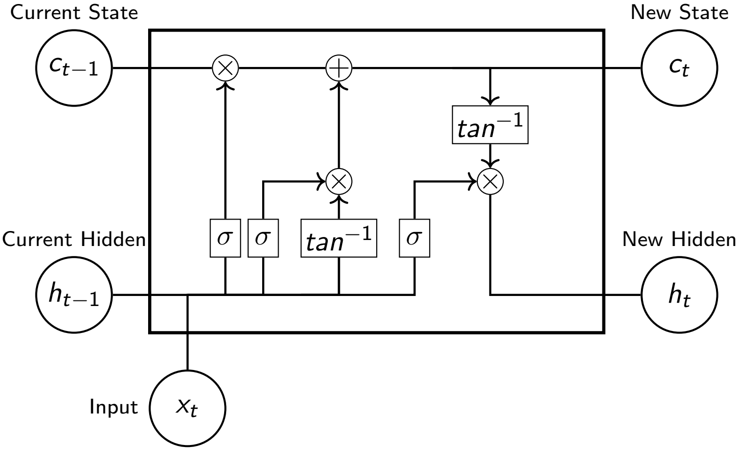
- RNN and LSTM are often used as synonyms in literature
- Has been proven that "Vanilla recursion" performs poorly
- LSTM are the state of the art for Time Series Classification



A deep look inside an LSTM cell

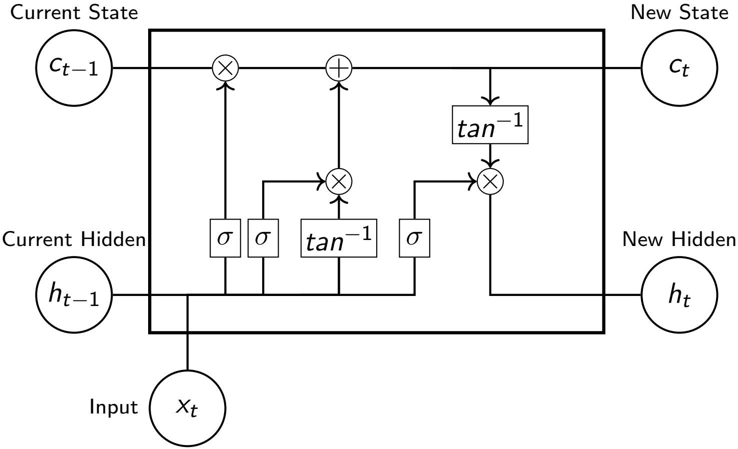


A deep look inside an LSTM cell



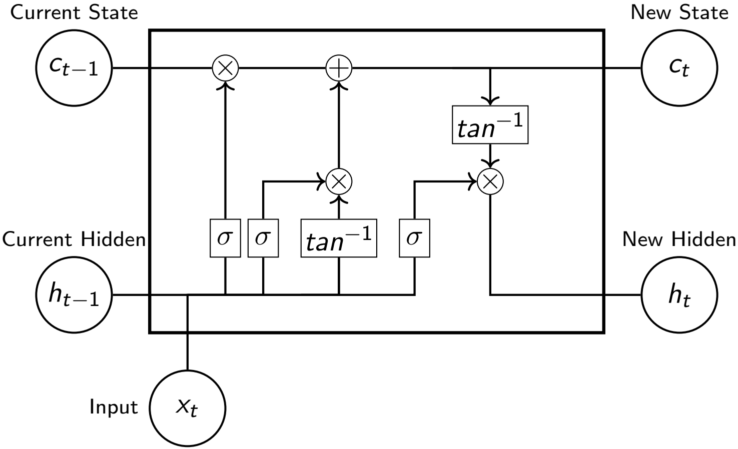
- $f_t = \sigma(W_f x_t + U_f h_t + b_f)$

A deep look inside an LSTM cell



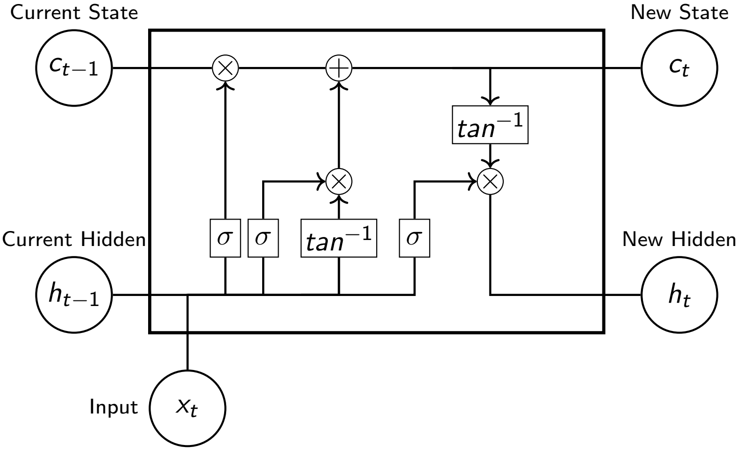
- $f_t = \sigma(W_f x_t + U_f h_t + b_f)$
- $i_t = \sigma(W_i x_t + U_i h_t + b_i)$

A deep look inside an LSTM cell



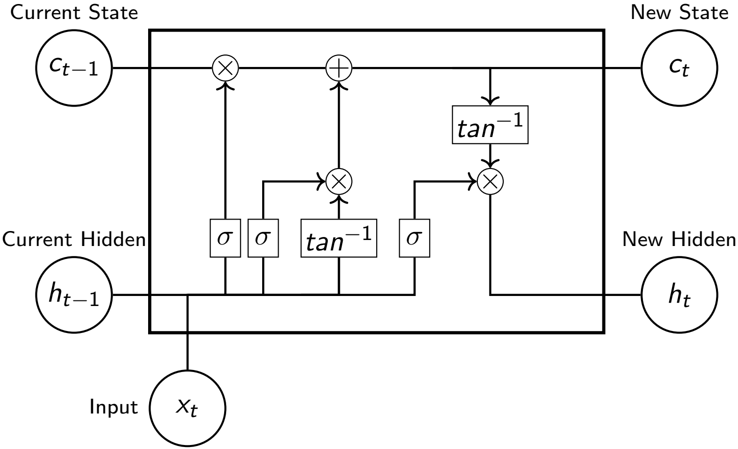
- $f_t = \sigma(W_f x_t + U_f h_t + b_f)$
- $i_t = \sigma(W_i x_t + U_i h_t + b_i)$
- $i_c = tan^{-1}(W_c x_t + U_c h_t + b_c)$

A deep look inside an LSTM cell



- $f_t = \sigma(W_f x_t + U_f h_t + b_f)$
- $i_t = \sigma(W_i x_t + U_i h_t + b_i)$
- $i_c = tan^{-1}(W_c x_t + U_c h_t + b_c)$
- $c_t = f_t \circ c_{t-1}$

A deep look inside an LSTM cell



- $f_t = \sigma(W_f x_t + U_f h_t + b_f)$
- $i_t = \sigma(W_i x_t + U_i h_t + b_i)$
- $i_c = tan^{-1}(W_c x_t + U_c h_t + b_c)$
- $c_t = f_t \circ c_{t-1} + i_t \circ i_c$

A deep look inside an LSTM cell

- $f_i = \sigma(W_f x_i + U_f h_{i-1} + b_f)$
- $i_i = \sigma(W_i x_i + U_i h_{i-1} + b_i)$
- $o_i = \sigma(W_o x_i + U_o h_{i-1} + b_o)$
- $c_i = \sigma(f_i \circ c_{i-1} + i_i \circ tan^{-1}(W_c X_i + U_c h_{i-1} + b_c))$
- $h_t = o_i * tan^{-1}(c_i)$

Using

- $W_f,\ W_i,\ W_o,\ W_C \in R^{n\times h}$
- $U_f,\ U_i,\ U_o,\ U_c \in R^{h\times h}$
- $b_f,\ b_i,\ b_o,\ b_c \in R^h$

A deep look inside an LSTM cell

- $f_i = \sigma(W_f x_i + U_f h_{i-1} + b_f)$
- $i_i = \sigma(W_i x_i + U_i h_{i-1} + b_i)$
- $o_i = \sigma(W_o x_i + U_o h_{i-1} + b_o)$
- $c_i = \sigma(f_i \circ c_{i-1} + i_i \circ tan^{-1}(W_c X_i + U_c h_{i-1} + b_c))$
- $h_t = o_i * tan^{-1}(c_i)$

Using

- $W_f,\ W_i,\ W_o,\ W_C \in R^{n \times h}$
- $U_f,\ U_i,\ U_o,\ U_c \in R^{h \times h}$
- $b_f,\ b_i,\ b_o,\ b_c \in R^h$

Can we do better?

A deep look inside an LSTM cell

- $f_i = \sigma(W_f x_i + U_f h_{i-1} + b_f)$
- $i_i = \sigma(W_i x_i + U_i h_{i-1} + b_i)$
- $o_i = \sigma(W_o x_i + U_o h_{i-1} + b_o)$
- $c_i = \sigma(f_i \circ c_{i-1} + i_i \circ tan^{-1}(W_c X_i + U_c h_{i-1} + b_c))$
- $h_t = o_i * tan^{-1}(c_i)$

Using

- $W_f,\ W_i,\ W_o,\ W_C \in R^{n \times h}$
- $U_f,\ U_i,\ U_o,\ U_c \in R^{h \times h}$
- $b_f,\ b_i,\ b_o,\ b_c \in R^h$

Can we do better?

Can we do **simpler**?

Let's take a step back

Let's take a step back

Let's take a step back

Let's take a step back

- Complex behaviors can **emerge** from simple ones
- Emergence is a key phenomenon in nature
- **Stigmergy** is one of the tools nature uses to achieve emergence

Let's take a step back

- Complex behaviors can **emerge** from simple ones
- Emergence is a key phenomenon in nature
- **Stigmergy** is one of the tools nature uses to achieve emergence

Can we **emerge** a computational memory using the **stigmergy**?

Biological Stigmergy

Implemented in nature via pheromonic marks

Biological Stigmergy

Implemented in nature via pheromonic marks

Mark

Biological Stigmergy

Implemented in nature via pheromonic marks

Stimulus

Mark

Biological Stigmergy

Implemented in nature via pheromonic marks

Stimulus

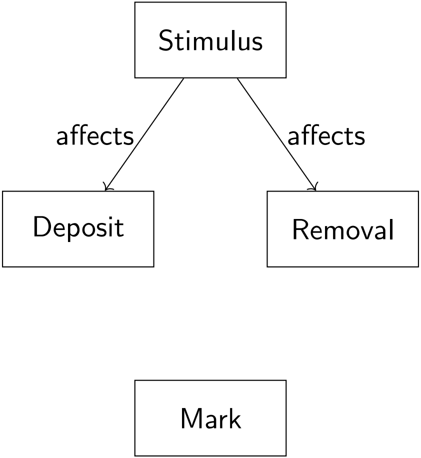
Deposit

Removal

Mark

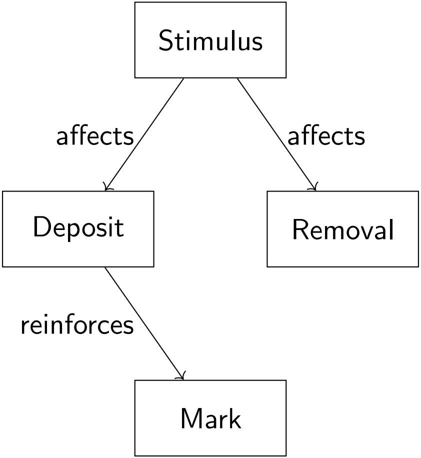
Biological Stigmergy

Implemented in nature via pheromonic marks



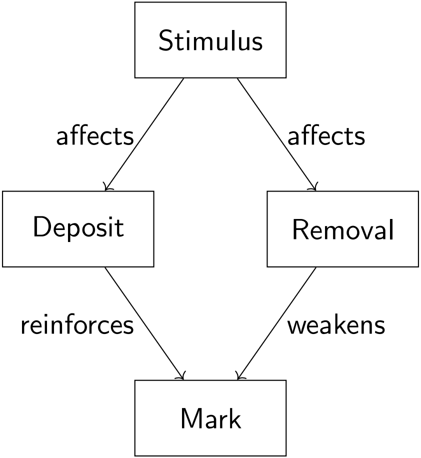
Biological Stigmergy

Implemented in nature via pheromonic marks



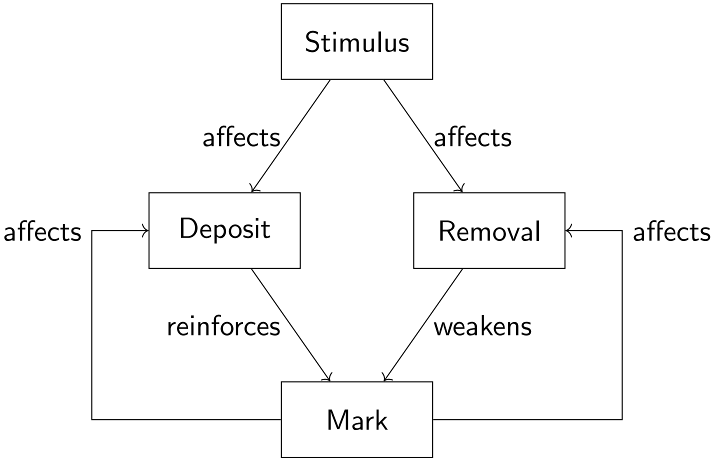
Biological Stigmergy

Implemented in nature via pheromonic marks



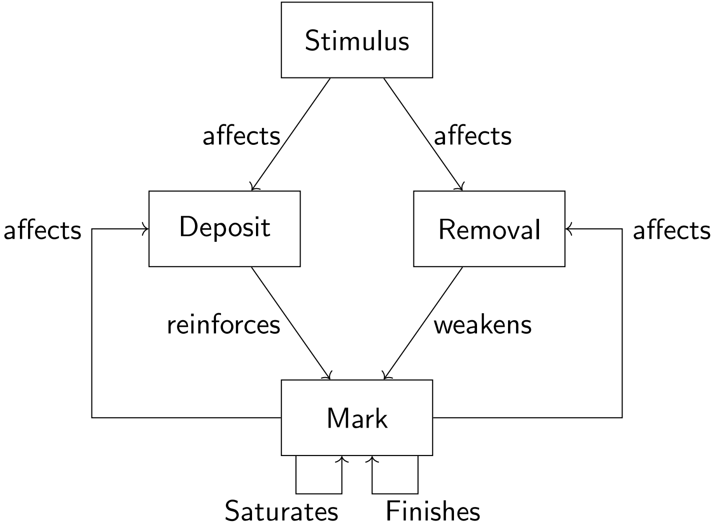
Biological Stigmergy

Implemented in nature via pheromonic marks



Biological Stigmergy

Implemented in nature via pheromonic marks

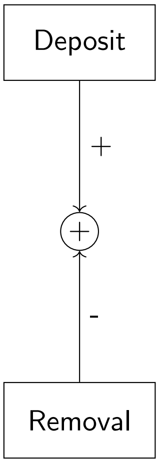


Computational Stigmergy

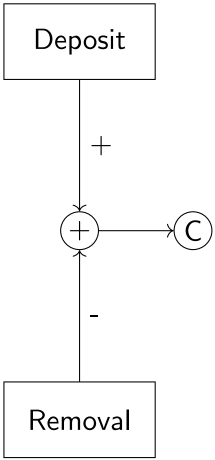
Deposit

Removal

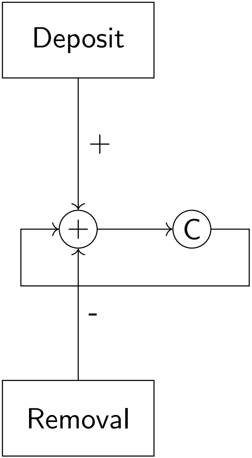
Computational Stigmergy



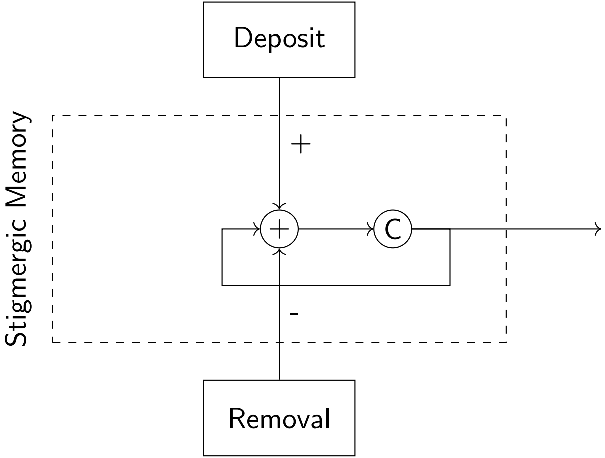
Computational Stigmergy



Computational Stigmergy

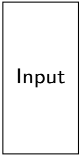


Computational Stigmergy

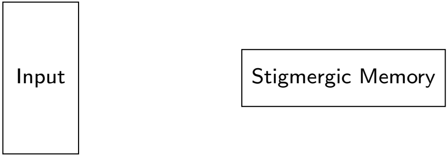


Stigmergic Memory ML Architecture

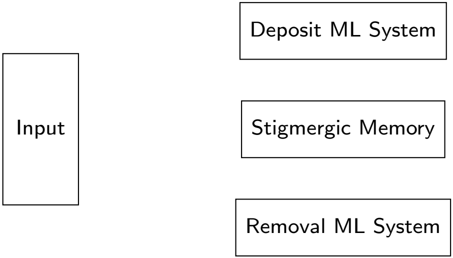
Stigmergic Memory ML Architecture



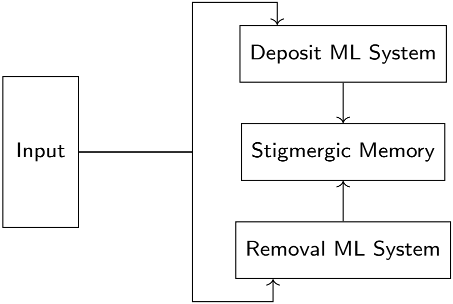
Stigmergic Memory ML Architecture



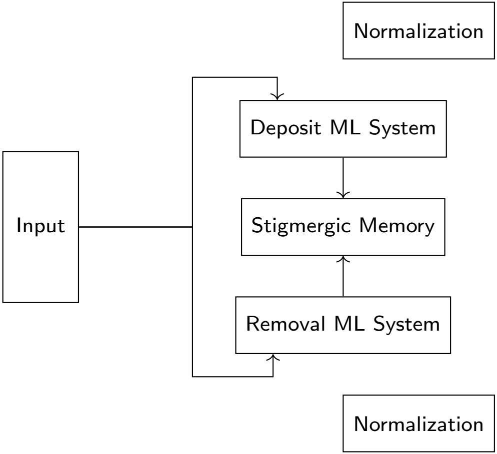
Stigmergic Memory ML Architecture



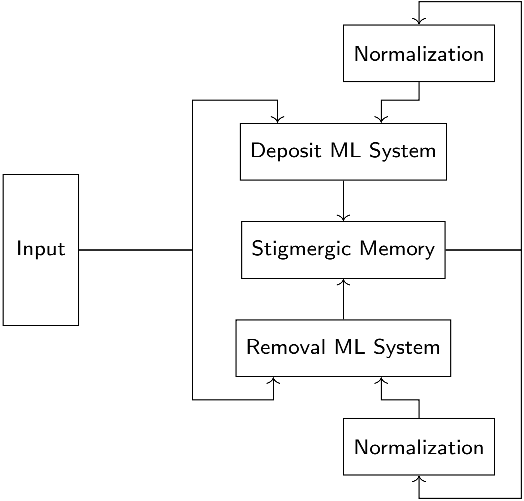
Stigmergic Memory ML Architecture



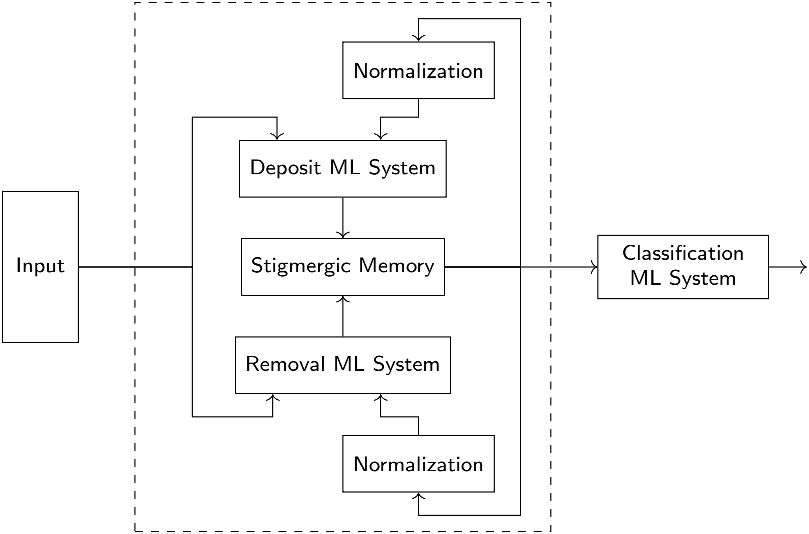
Stigmergic Memory ML Architecture



Stigmergic Memory ML Architecture

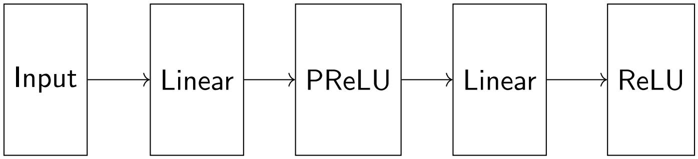


Stigmergic Memory ML Architecture



Experimental Stigmergic ML Systems

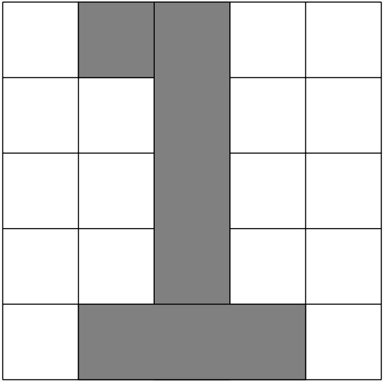
Neural Networks used as Deposit, Removal and Classification Systems

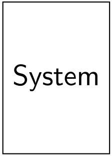


Experimental Architectures

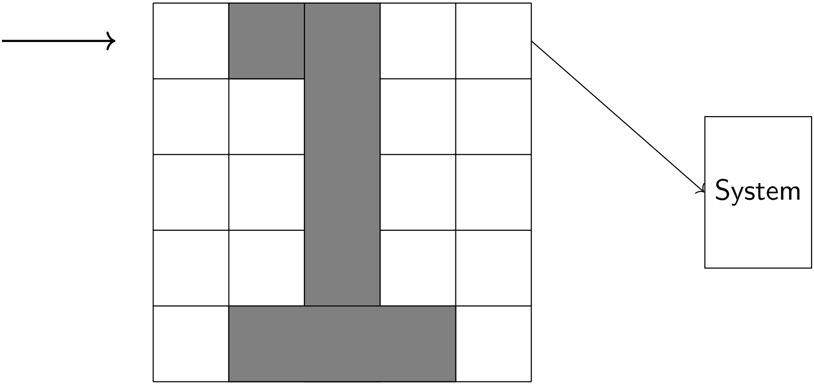
- Stigmergic Memory NNs
- LSTMs
- Vanilla RNNs
- FF NNs (only with spatial dataset)

Experimental Results: Spatial MNIST

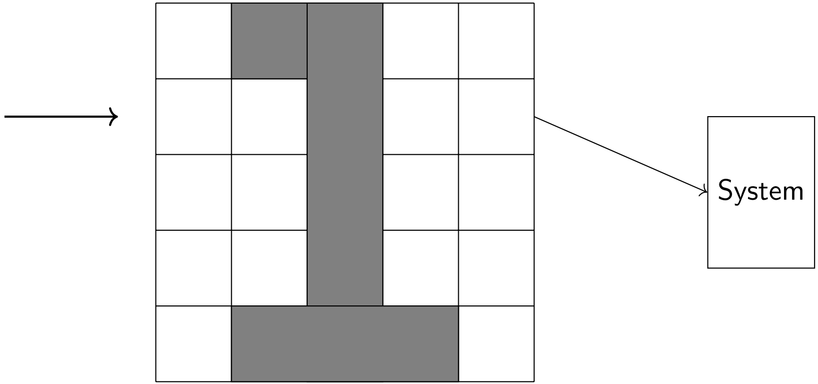




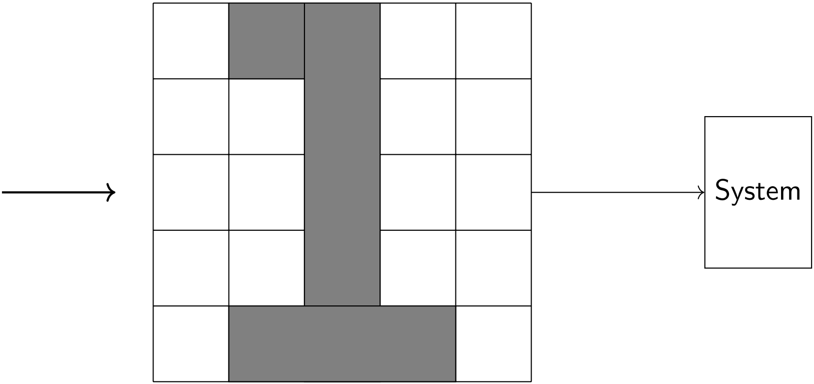
Experimental Results: Spatial MNIST



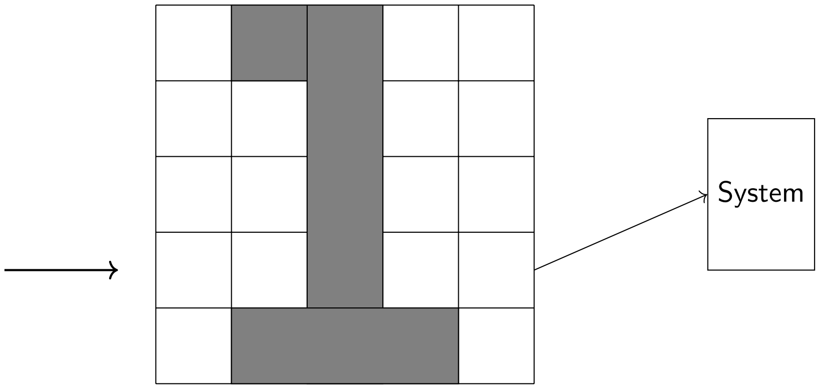
Experimental Results: Spatial MNIST



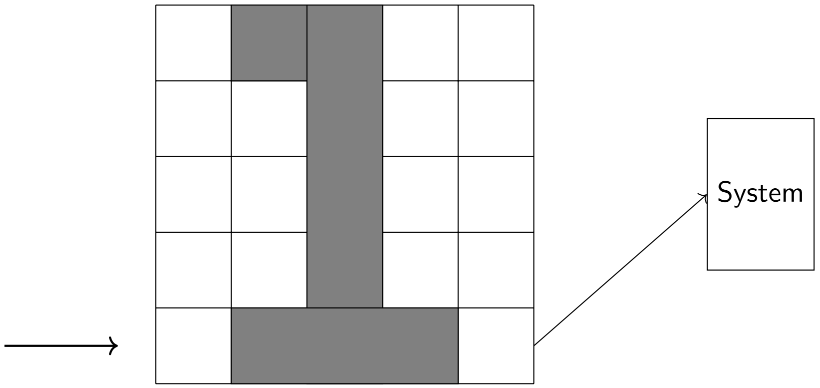
Experimental Results: Spatial MNIST



Experimental Results: Spatial MNIST



Experimental Results: Spatial MNIST

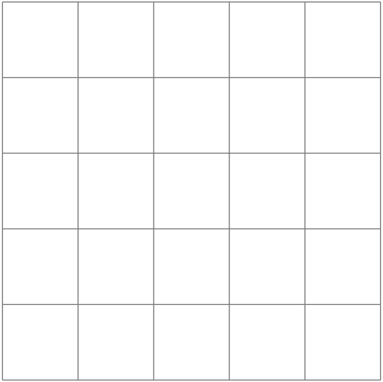


Experimental Results: Spatial MNIST

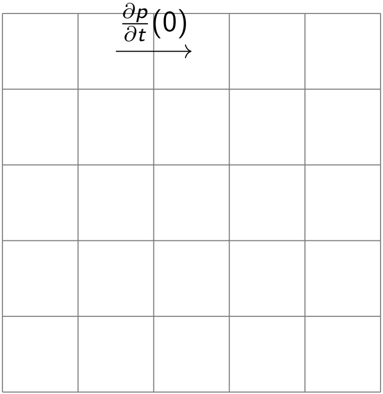
Architecture	N. Parameters	Accuracy
Stigmergic Memory	3190	96.5 ± 0.5 %
Static Feed Forward	328810	95.1 ± 0.02 %
LSTM	3360	94.3 ± 0.1 %
RNN	3482	76.6 ± 0.3 %

- Outperforms LSTMs, Vanilla RNNs and FFs
- Best performances, smaller number of parameters

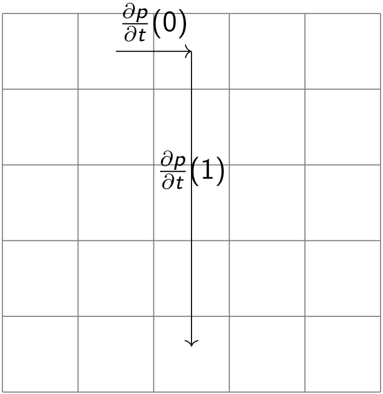
Experimental Results: Temporal MNIST



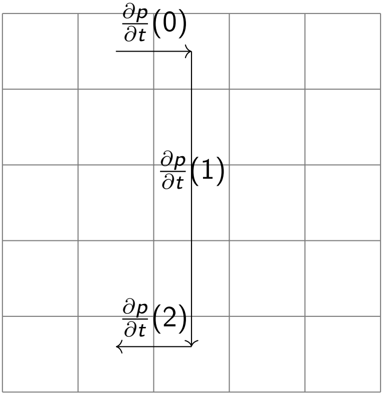
Experimental Results: Temporal MNIST



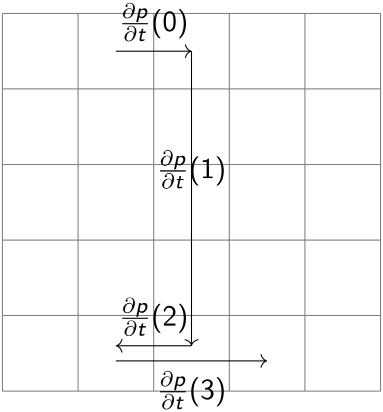
Experimental Results: Temporal MNIST



Experimental Results: Temporal MNIST



Experimental Results: Temporal MNIST



Experimental Results: Temporal MNIST

Architecture	N. Parameters	Accuracy
LSTM	5490	94.96 ± 0.2 %
Stigmergic Memory	5420	94.67 ± 0.7 %
RNN	5480	72.95 ± 11 %

- Outperforms Vanilla RNNs
- Same performances as LSTMs

Keep in touch

You can find the pytorch implementation on GitHub

https://github.com/galatolofederico/icpram2019

federico.galatolo@ing.unipi.it

galatolo.me

@galatolo

@galatolofederico