ε.δε.μ^2
ΕΠΙΣΤΗΜΗ ΔΕΔΟΜΕΝΩΝ & ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ

STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

Recurrent Neural Networks (RNN)

1. Memory Models
2. Hopfield Recurrent Networks
3. Long Short-Term Memory (LSTM) Networks

Large Language Models (LLM)

Prof. Vasilis Maglaris

maglaris@netmode.ntua.gr

www.netmode.ntua.gr

Room 002, New ECE Building

Wednesday May 21, 2025

STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

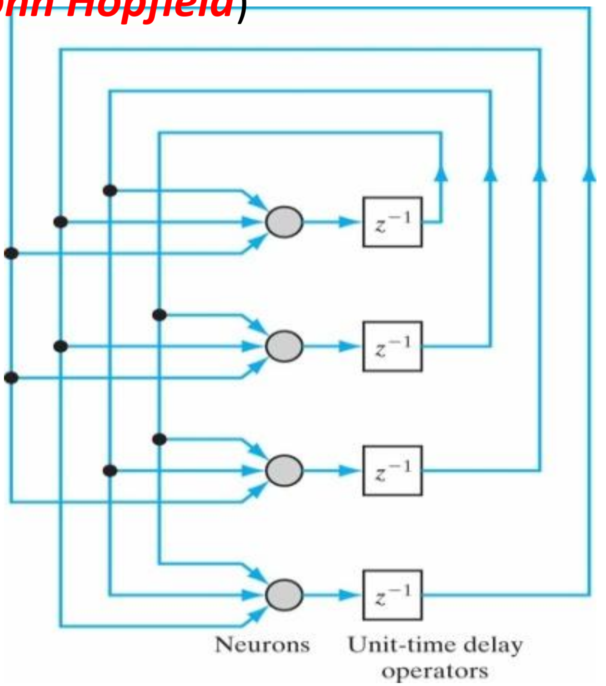
Associative Memory - Content Addressable Memory (CAM)

https://www.doc.ic.ac.uk/~ae/papers/Hopfield-networks-15.pdf

- **Computer Memory Organization**
  - Traditional ***address-based*** models
  - Newer models based on pattern storage: ***Associative Memory*** or ***Content Addressable Memory*** (***CAM***) correlating new (and/or distorted) elements with pre-stored patterns
- **Human Memory Organization**
  - Pattern storage based on related characteristics (***associative memory***) and not in specific brain location
- **Neurophysiological Learning**
  - Tuning of neural ***synapses*** and their approximate storage, e.g. based on ***Hebbian*** rules: Amplify synaptic weights among simultaneously active neurons, drop to zero weights among non-synchronized neurons
- **Neural Memory Models: *Hopfield*** Networks
  - 1982, ***John Hopfield*** https://www.pnas.org/doi/pdf/10.1073/pnas.79.8.2554
  - It operates as an ***attractor*** that assigns sample elements to stored patterns with minimum ***Gibbs*** "energy"
  - It learns and stores patterns via ***supervised learning*** using ***Hebb***'s rules

STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

Hopfield Deterministic Neural Network (1982, *John Hopfield*)

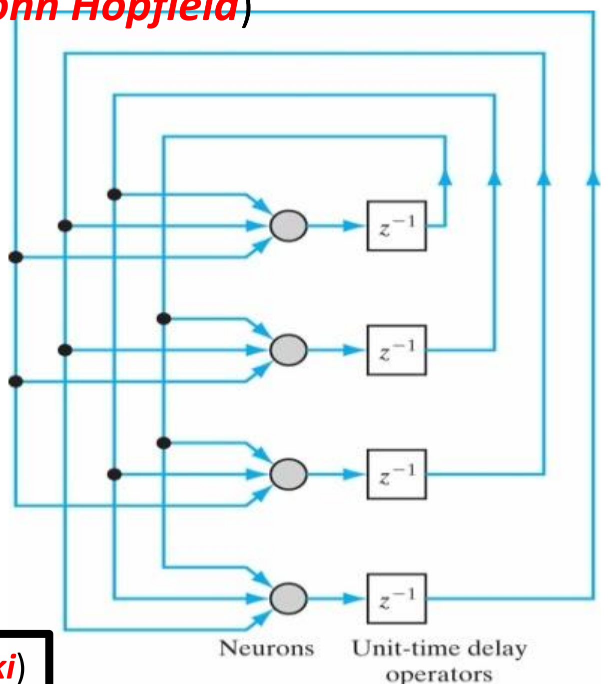


- Binary non-stochastic neurons with recurrent synapses, threshold activation and ***Hebbian supervised learning*** to determine $w_{ji} = w_{ij}, w_{ii} = 0$ in (local) minimum of ***system energy***. Application in pattern classification – recognition of images
- Input sample elements converge to output ***fixed points***, ***target patterns*** stored in w_{ji} during ***supervised*** training
- First application in pattern recognition (e.g. distinction of hand-written decimal numbers from distorted input elements, retrieved from the ***MNIST*** database)

STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

Hopfield Deterministic Neural Network (1982, *John Hopfield*)

- Binary non-stochastic neurons with recurrent synapses, threshold activation and ***Hebbian supervised learning*** to determine $w_{ji} = w_{ij}, w_{ii} = 0$ in (local) minimum of ***system energy***. Application in pattern classification – recognition of images
- Input sample elements converge to output ***fixed points***, ***target patterns*** stored in w_{ji} during ***supervised*** training
- First application in pattern recognition (e.g. distinction of hand-written decimal numbers from distorted input elements, retrieved from the ***MNIST*** database)



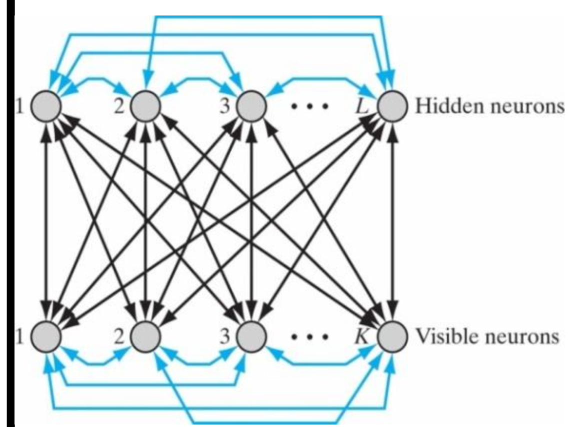
Stochastic Extension: ***Boltzmann Machine*** (1985, ***Geoffrey Hinton*** & ***Terry Sejnowski***)

A ***Boltzmann Machine*** (**BM**) is a ***Stochastic Recurrent Network*** with 2 layers of neurons:

- K **Visible,** L **Hidden** binary state ***Stochastic Neurons***, with state probabilities assigned via ***unsupervised*** learning
- **Symmetric Synapses** $i \to j$: $w_{ji} = w_{ij}, w_{ii} = 0$ amongst **all** neurons

The **BM** converges via ***unsupervised*** learning to ***Markov Random Field*** "thermal" equilibrium:

- Binary **training vectors** are clamped to ***Visible Nodes***; via a **gradient ascent algorithm** synaptic weights converge and final states of **both** ***Visible*** & ***Hidden Neurons*** are determined
- A new **input** vector (**test**) is inserted In ***Visible Nodes***. The **BM** generates via ***Gibbs sampling*** its **output** image as an update in the ***Visible Nodes***, statistically conforming to training sample vectors The **BM** converges via ***unsupervised learning*** to equilibrium probabilities of a ***Markov Random Field***:



STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

Hopfield Networks & Neural Memory (1/2)

https://towardsdatascience.com/hopfield-networks-neural-memory-machines-4c94be821073

- Binary ***deterministic*** state s_i of neuron i, ***recurrent*** symmetric synapses $i \leftrightarrow j$: $w_{ji} = w_{ij}$, $w_{ii} = 0$, ***binary threshold activation*** ± 1 (sgn)
- Output of Neuron i: $y_i = \mathrm{sgn}\{\sum_j y_j w_{ij} + b_j\}$, b_j ***bias***
- State of Neuron i: $s_i = \begin{cases} +1, y_i > 0 \\ -1, y_i < 0 \end{cases}$

1st Phase: Storage of Patterns via Supervised Learning

- Storage of a ***target pattern*** according to ***Hebbian*** rules: $w_{ij} = s_i s_j$ in equilibrium (minimum energy similarly to the ***Ising Network Model***)
- Simultaneous storage of M ***target patterns*** $\mu = 1,2,..,M$: The neural states are vectors with coordinates $s_i^{(\mu)}$ with synaptic weights abiding by the ***generalized Hebbian rule***:

$$w_{ij} = \frac{1}{M} \sum_{\mu=1}^{M} s_i^{(\mu)} s_j^{(\mu)}$$

- Alternatively, determining w_{ij} via ***back-propagation supervised learning*** requires very large training sets

2nd Phase: Retrieval Process for New Test Element

- Iterations in discrete steps to ***update*** neuron states, one at a time in random order based on the relation $y_i = \mathrm{sgn}\{\sum_j y_j w_{ij} + b_j\}$

STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

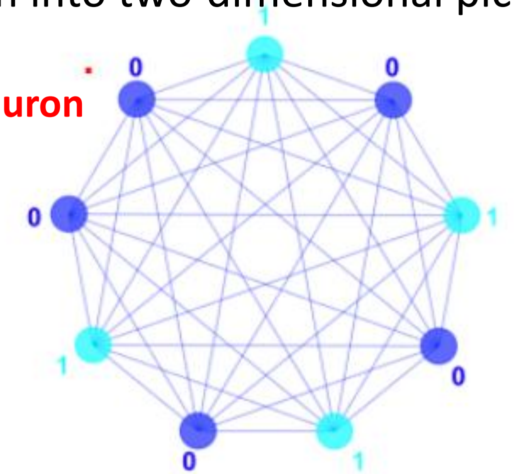
Hopfield Nnetworks & Neural Memory (2/2)

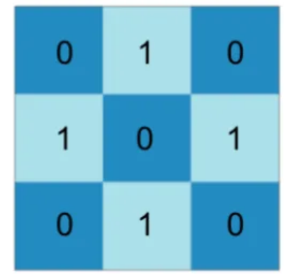
https://towardsdatascience.com/hopfield-networks-neural-memory-machines-4c94be821073

Example of Using a Hopfield Network for Pattern Recognition

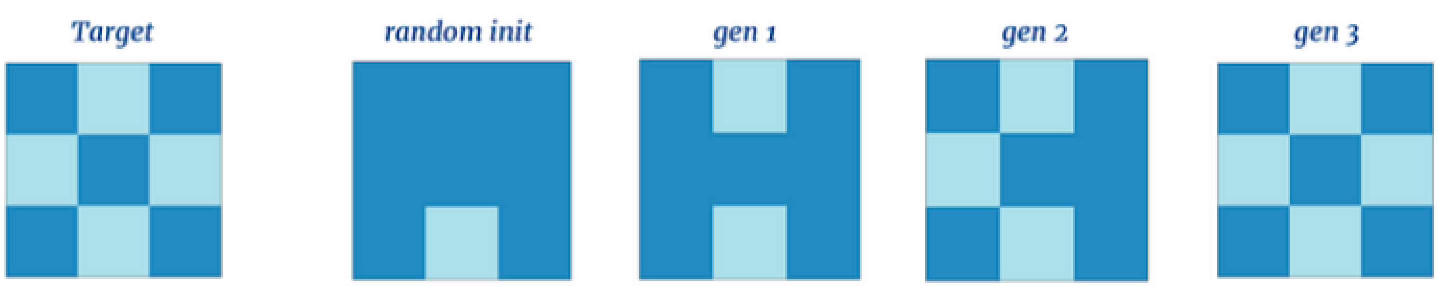
- Pattern $[0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0]^{\mathrm{T}}$, vector of 9 binary digits $\{0,1\}$
- Transformation of pattern into two-dimensional picture:

Initial neuron



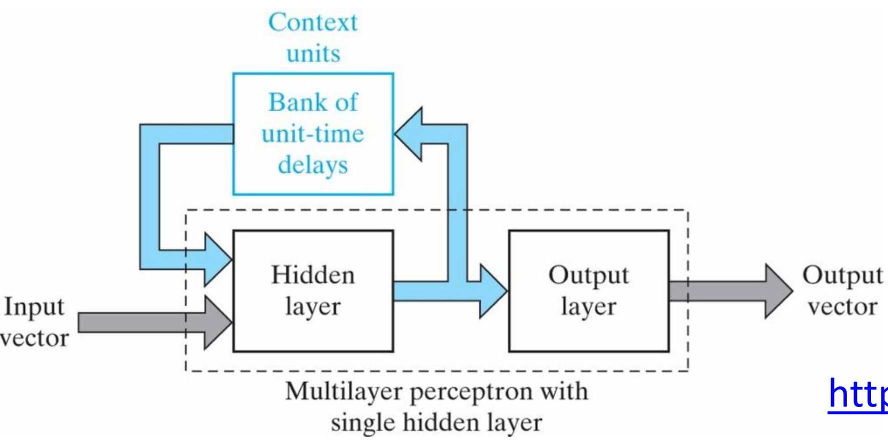


- Patterns are stored in a ***Hopfield*** Network of 9 neurons of states $\{0,1\}$ (instead of ± 1)
- The ***Hopfield*** nets are referred to as ***attractor networks*** since new ***test patterns*** are attracted (converge) to a pre-stored ***target pattern***. In the example, the process starts at a random initial state and converges to the target in just 3 ***updates***:

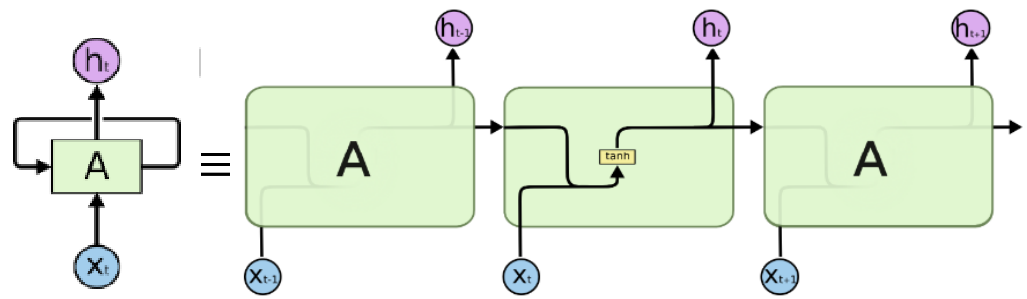


Recurrent Neural Networks (RNN) (1/2)

Simple Recurrent Network (SRN)



Expansion of SRN into an Equivalent Sequence of Single-layer Perceptrons im Tandem



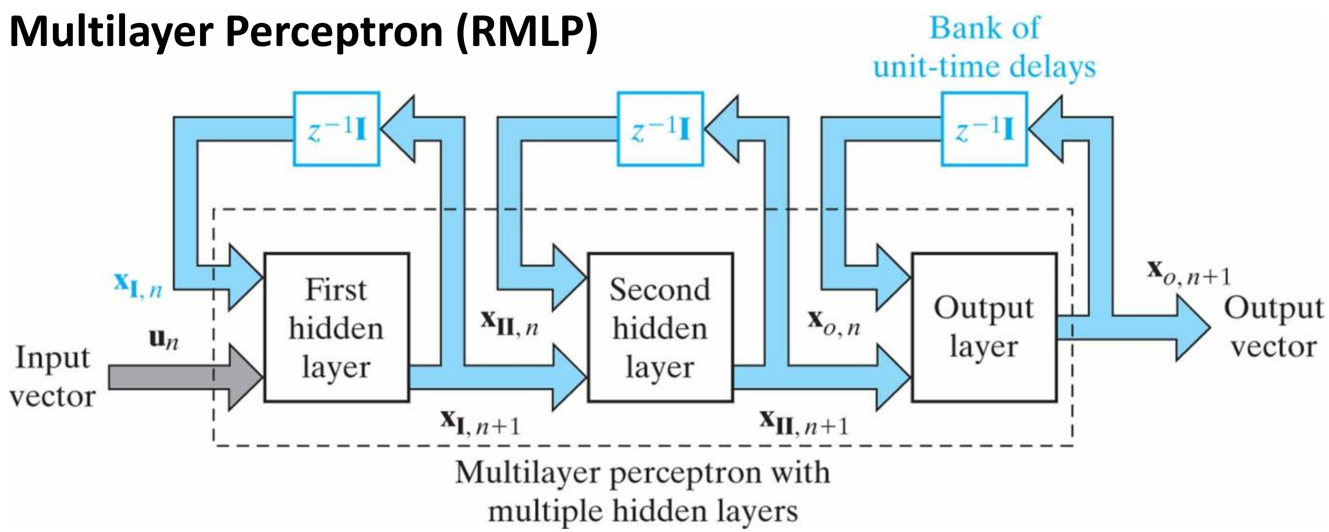
http://colah.github.io/posts/2015-08-Understanding-LSTMs/

A : Hidden *Perceptron* layer with *Activation Function* tanh and *output* $\in [-1, +1\}$

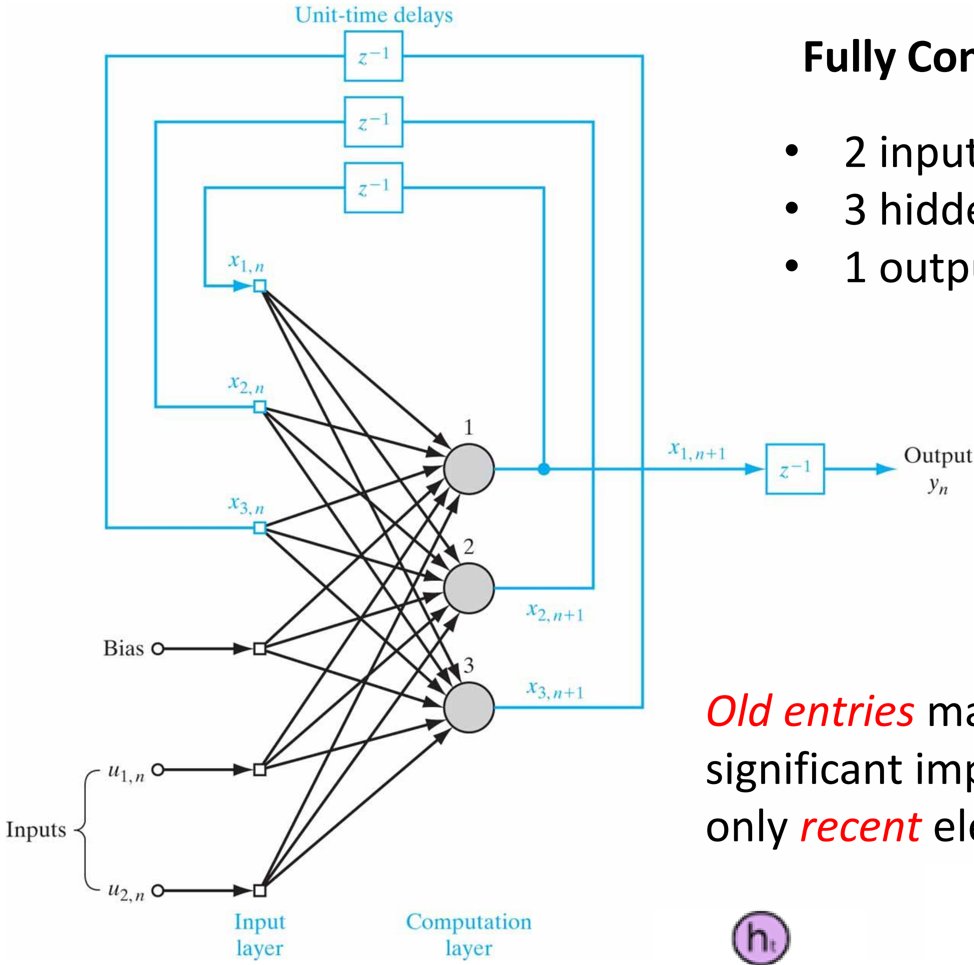
$\mathbf{x}_t$: *Input* element to ***A*** in time period t

$\boldsymbol{h}_t$: Hidden output of ***A*** in time period t

Recurrent Multilayer Perceptron (RMLP)



Αναδρομικά Νευρωνικά Δίκτυα - Recurrent Neural Networks RNN (2/2)

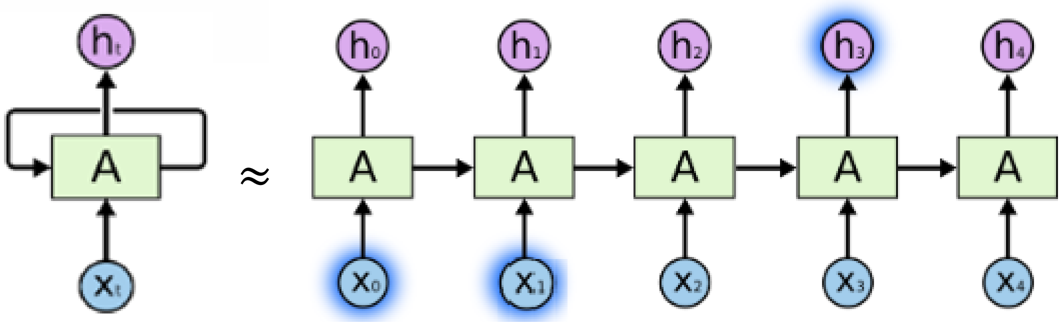


Fully Connected SRN

- 2 input nodes
- 3 hidden neurons
- 1 output node

Long-Term Dependencies

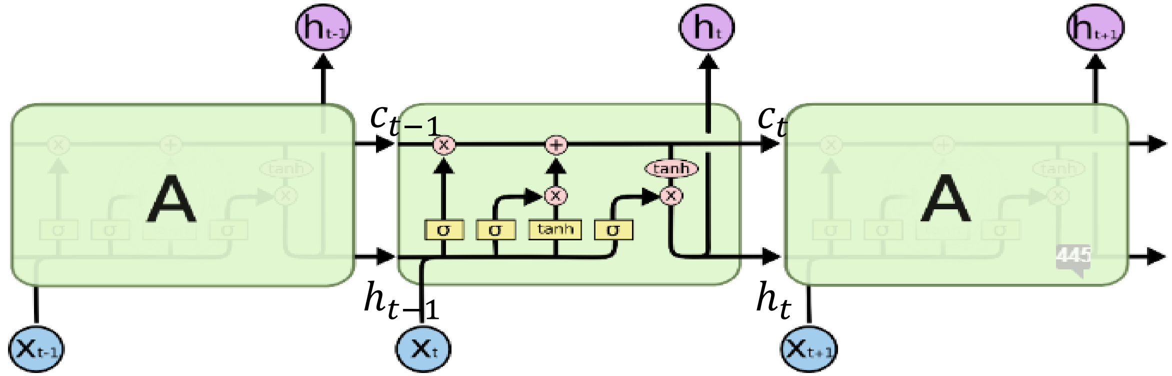
Old entries may hinder learning without having a significant impact to the *output* ⇒ need to consider only *recent* elements (e.g. 5 time periods)



Long Short-Term Memory (LSTM) (1/3)

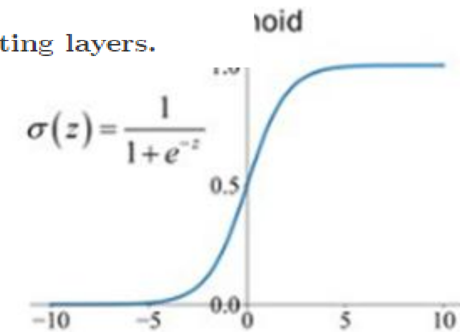
http://colah.github.io/posts/2015-08-Understanding-LSTMs/

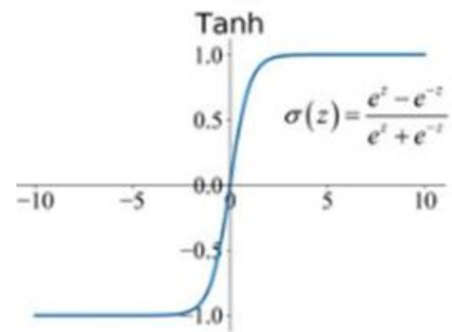
Equivalent LSTM Expansion with Cells in Tandem



The repeating module in an LSTM contains four interacting layers.

x_t:	Input vector of cell t
c_t:	***Cell State*** vector ($c_{t-1} \rightarrow c_t$ recursively, depending on h_{t-1}, x_t if allowed by ***Control Gates***)
h_t:	Hidden output vector of cell t
σ:	Sigmoid activation function $\sim\{0,1\}$ in 3 ***Gates*** (***forget***, ***input***, ***output***) that control the flow of information e.g. the ***forget gate*** cuts ***obsolete*** information
tanh:	Hyperbolic tan activation, tuning the ***Cell State*** to $[-1,1]$ values

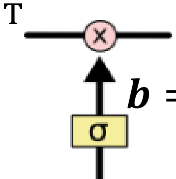




Control Gate Strucure

Hadamard (point-wise) product

$\boldsymbol{v} = [v_1 \dots v_k]^{\mathrm{T}}$



$$\boldsymbol{v} \circ \boldsymbol{b} = [v_1 \times b_1 \dots v_k \times b_k]^{\mathrm{T}}$$

$$\boldsymbol{b} = [b_1 \dots b_{\mathrm{k}}]^{\mathrm{T}} = [\sigma(a_1) \dots \sigma(a_{\mathrm{k}})]^{\mathrm{T}} = \left[{}^{1}/_{(1+e^{-a_1})} \dots {}^{1}/_{(1+e^{-a_k})}\right]^{\mathrm{T}}$$

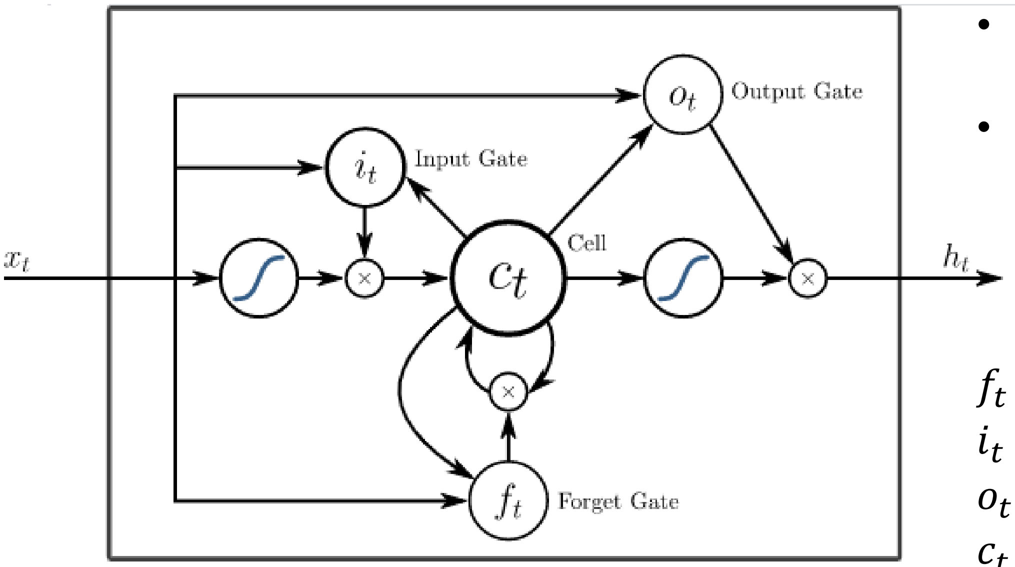
$\boldsymbol{a} = [a_1 \dots a_{\mathrm{k}}]^{\mathrm{T}}$

STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

Long Short-Term Memory (LSTM) (2/3)

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

S. Hochreiter & ***J. Schmidhuber***, 1997
https://www.bioinf.jku.at/publications/older/2604.pdf



Architecture of LSTM Units

- Regulatory functionality: ***Input***, ***Output***, ***Forget Gates*** (usually neurons with sigmoid activation)
- Memory ***Cell*** (storage of time-dependent factors within the remembrance range, regulated by the *Forget Gate*)

Input/Output Operations of Functional Units

$$f_t = \sigma_g(W_f x_t + U_f c_{t-1} + b_f)$$
$$i_t = \sigma_g(W_i x_t + U_i c_{t-1} + b_i)$$
$$o_t = \sigma_g(W_o x_t + U_o c_{t-1} + b_o)$$
$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + b_c)$$
$$h_t = \sigma_h(o_t \circ c_t),$$

$\circ$: *Hadamard* (point-wise) product

where

$\sigma_g(a) = \frac{1}{1+e^{-a}}$, *sigmoid function,* in the limit $\{0,1\}$

$\sigma_c(a) = \tanh(a)$, *hyperbolic tangent*, in the limit ± 1

$\sigma_c(a) = \tanh(a)$, *hyperbolic tangent* or $\sigma_h(a) = a$

Definition of Input/Output Variables

d: number of input features

h : number of hidden units

$x_t \in \mathbb{R}^d$: input vector to the LSTM unit

$f_t \in \mathbb{R}^h$: forget gate's activation vector

$i_t \in \mathbb{R}^h$: input/update gate's activation vector

$o_t \in \mathbb{R}^h$: output gate's activation vector

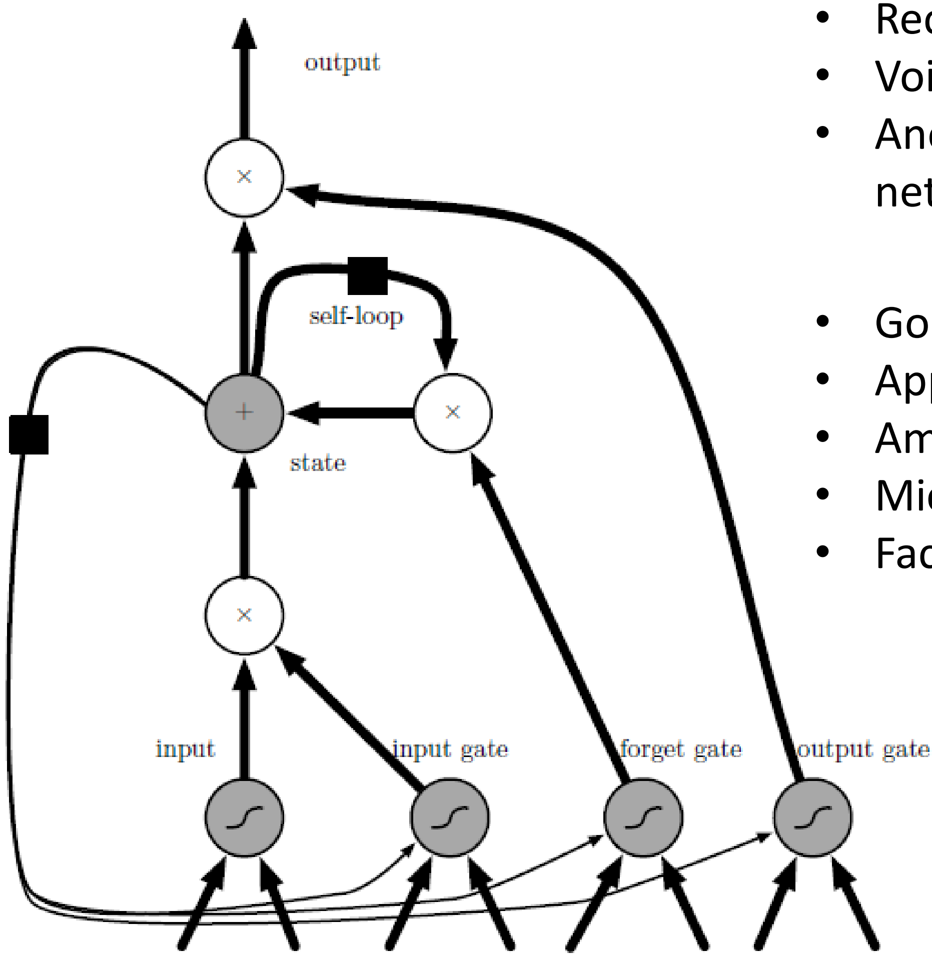
$h_t \in \mathbb{R}^h$: output vector of the LSTM unit (hidden state vector)

$c_t \in \mathbb{R}^h$: cell state vector

$W \in \mathbb{R}^{h\times d}$, $U \in \mathbb{R}^{h\times h}$, $b \in \mathbb{R}^h$: weights & bias parameters, tuned via ***supervised learning***

Long Short-Term Memory (LSTM) (3/3)

I. Goodfellow, Y. Bengio, A. Courville, "Deep Learning", Ch.10



Applications of LSTM

- Recognition of hand-written text
- Voice recognition
- Anomaly detection in information systems and networks - Intrusion Detection Systems (IDS)

Commercial Applications

- Google (*Smartphone*, *Translate*,…)
- Apple (*QuickTime*, *iPhone*, *Siri*…)
- Amazon (*Alexa*,…)
- Microsoft (*Switchboard*…)
- Facebook (Automatic translation)

Control of State Storage

- Dynamic determination of time-window via the ***Forget Gate***
- Possibility of cell-state access from other modules (***Input***. ***Output Gates***): ***Peephole LSTM***
- Cell parameter tuning via Supervised Learning from ***Labeled Datasets***

STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

Large Language Models (LLM)

https://en.wikipedia.org/wiki/Large_language_model

- **Current hype** involving combination of ***Natural Language Processing*** (**NLP**) and ***Artificial Intelligence*** (**AI**) - ***Machine Learning*** (**ML**) fields
- Builds on years R&D in **NLP** (e.g. BERT Models, ***Search Engines***, ***Automatic Translation***, ***Chatboxes***…) and **ML** (***Deep Learning***, ***Generative Models***, ***Autoencoders***, ***Transformers***…)
- Tuning of ***billions*** of parameters (***synaptic*** weights, NLP ***tokens***…)
- Use of ***Unsupervised***, ***Supervised***, ***Self-supervised*** pre-training and ***Reinforcement Learning*** algorithms
- Deployment of extensive ***data-centers***, with very high ***energy*** requirements
- Need extensive resources, usually offered to end-users by ***Computing Clouds*** as ***S***oftware-***a***s-***a***-***S***ervice (**AaaS**), with downloading options
- Very lengthy pre-training for corpus (***foundation***) model, possible customization for specific use-cases
- Raised legal ***regulatory*** matters (property rights, confidentiality, openness), ***ethical*** & ***geopolitical*** concerns, far-reaching effects of ***labor realignment***, challenges that humanity faced in previous industrial and technological revolutions (reminiscent of violent ***Luddism*** reactions in the early 19th century against proliferation of looming machines etc.)

For a comprehensive review see the **2025** book "***Foundations of LLMs***" by ***Tong Xiao*** & ***Jingbo Zhu***, NLP Labs, Northeastern University – China, https://arxiv.org/pdf/2501.09223)