LSTM Forward Pass

$$
\begin{aligned}
\mathbf{z}^t &= g(\mathbf{W}_z \mathbf{x}^t + \mathbf{R}_z \mathbf{y}^{t-1} + \mathbf{b}_z) \\
\mathbf{i}^t &= \sigma(\mathbf{W}_i \mathbf{x}^t + \mathbf{R}_i \mathbf{y}^{t-1} + \mathbf{p}_i \odot \mathbf{c}^{t-1} + \mathbf{b}_i) \\
\mathbf{f}^t &= \sigma(\mathbf{W}_f \mathbf{x}^t + \mathbf{R}_f \mathbf{y}^{t-1} + \mathbf{p}_f \odot \mathbf{c}^{t-1} + \mathbf{b}_f) \\
\mathbf{c}^t &= \mathbf{i}^t \odot \mathbf{z}^t + \mathbf{f}^t \odot \mathbf{c}^{t-1} \\
\mathbf{o}^t &= \sigma(\mathbf{W}_o \mathbf{x}^t + \mathbf{R}_o \mathbf{y}^{t-1} + \mathbf{p}_o \odot \mathbf{c}^t + \mathbf{b}_o) \\
\mathbf{y}^t &= \mathbf{o}^t \odot h(\mathbf{c}^t)
\end{aligned}
$$

The Deepest Learning

Jan Koutník

IDSIA

CYNAR

Istituto Dalle Molle di studi sull'intelligenza artificiale (IDSIA)

Istituto Dalle Molle di studi sull'intelligenza artificiale (IDSIA)

- Swiss AI Institute in Lugano

Istituto Dalle Molle di studi sull'intelligenza artificiale (IDSIA)

- Swiss AI Institute in Lugano
- Jürgen Schmidhuber & Luca Gambardella

Istituto Dalle Molle di studi sull'intelligenza artificiale (IDSIA)

- Swiss AI Institute in Lugano
- Jürgen Schmidhuber & Luca Gambardella
- Long Short-Term Memory (LSTM; 1997)

Istituto Dalle Molle di studi sull'intelligenza artificiale (IDSIA)

- Swiss AI Institute in Lugano
- Jürgen Schmidhuber & Luca Gambardella
- Long Short-Term Memory (LSTM; 1997)
- Universal Optimal AI Theory (AIXI; 2001)

Istituto Dalle Molle di studi sull'intelligenza artificiale (IDSIA)

- Swiss AI Institute in Lugano
- Jürgen Schmidhuber & Luca Gambardella
- Long Short-Term Memory (LSTM; 1997)
- Universal Optimal AI Theory (AIXI; 2001)
- Deep Convolutional Neural Networks on GPUs (2009)

Istituto Dalle Molle di studi sull'intelligenza artificiale (IDSIA)

- Swiss AI Institute in Lugano
- Jürgen Schmidhuber & Luca Gambardella
- Long Short-Term Memory (LSTM; 1997)
- Universal Optimal AI Theory (AIXI; 2001)
- Deep Convolutional Neural Networks on GPUs (2009)
- Natural Evolution Strategy (NES; 2010)

Istituto Dalle Molle di studi sull'intelligenza artificiale (IDSIA)

- Swiss AI Institute in Lugano
- Jürgen Schmidhuber & Luca Gambardella
- Long Short-Term Memory (LSTM; 1997)
- Universal Optimal AI Theory (AIXI; 2001)
- Deep Convolutional Neural Networks on GPUs (2009)
- Natural Evolution Strategy (NES; 2010)
- Fast Scanning Convolutional NNs (2011)

Istituto Dalle Molle di studi sull'intelligenza artificiale (IDSIA)

- Swiss AI Institute in Lugano
- Jürgen Schmidhuber & Luca Gambardella
- Long Short-Term Memory (LSTM; 1997)
- Universal Optimal AI Theory (AIXI; 2001)
- Deep Convolutional Neural Networks on GPUs (2009)
- Natural Evolution Strategy (NES; 2010)
- Fast Scanning Convolutional NNs (2011)
- **http://www.idsia.ch**

What is Deep Learning?

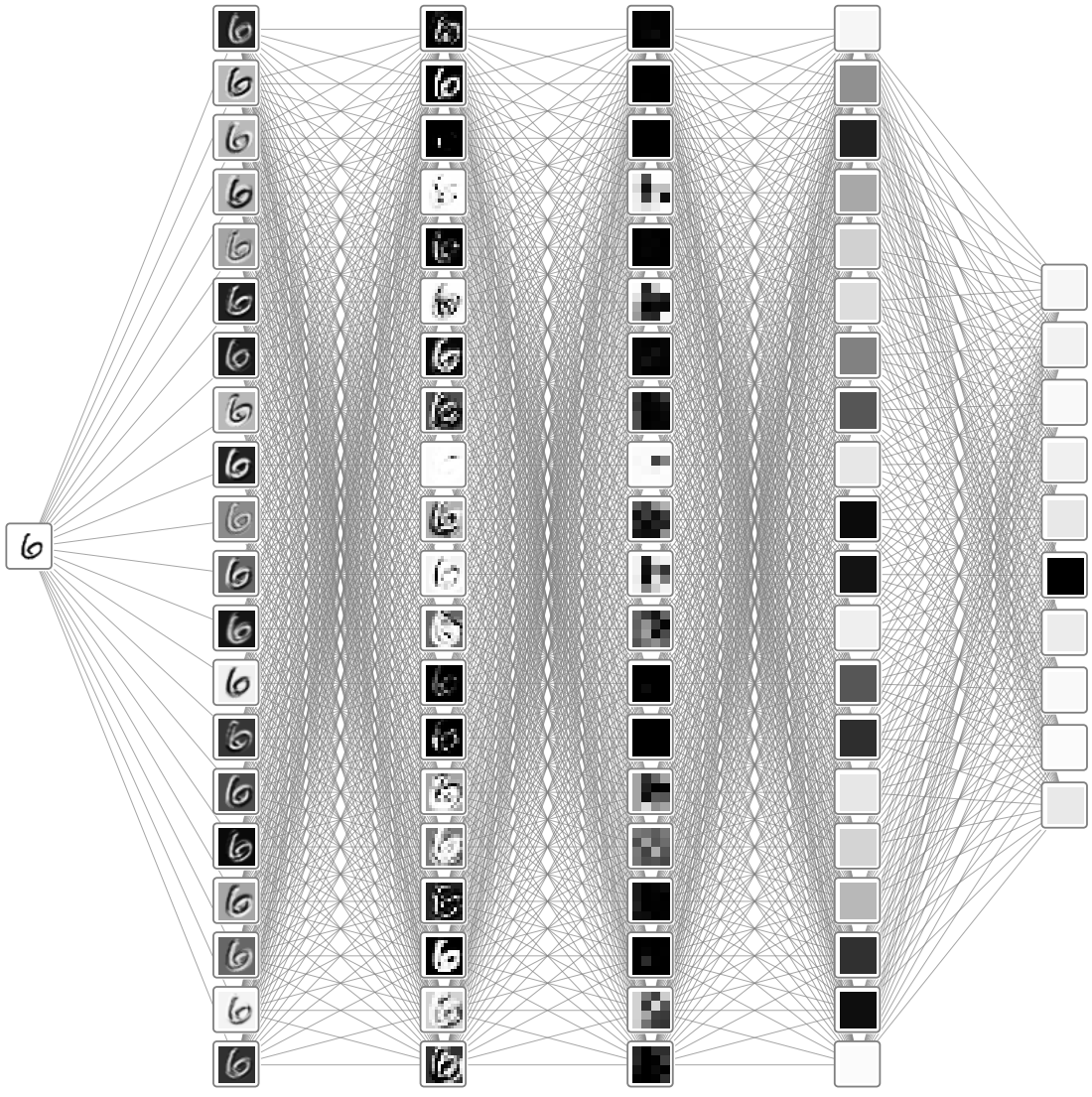
- A fancy *marketing* name for Artificial Neural Networks (NN)
- Namely Convolutional Neural Networks used for supervised perception tasks
- A few centuries old term
- Brought to machine learning in 1986[1]
- Used in NN context in 1999[2]
- The depth can be measured by *credit assignment path* length in NN
- ... number of layers

---

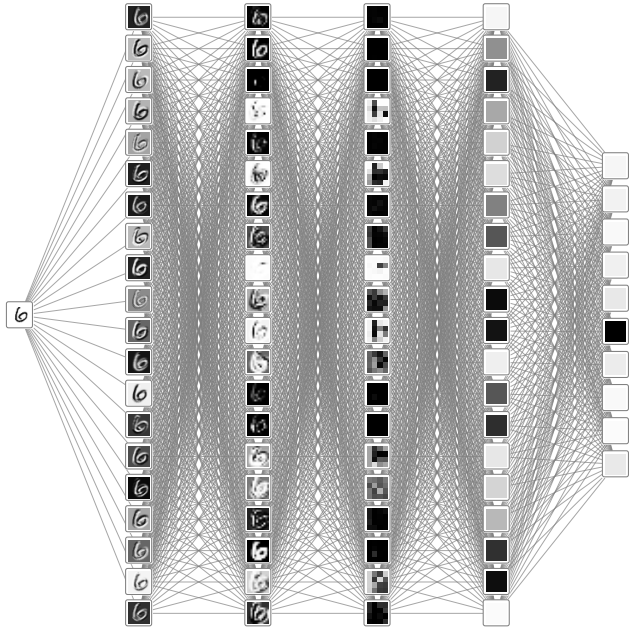
[1] R. Dechter, "Learning while searching in constraint-satisfaction-problems," in *Proceedings of the 5th National Conference on Artificial Intelligence. Philadelphia, PA, August 11-15, 1986. Volume 1: Science.*, T. Kehler, Ed., Morgan Kaufmann, 1986, pp. 178–185. [Online]. Available: http://www.aaai.org/Library/AAAI/1986/aaai86-029.php.

[2] I. N. Aizenberg, "Neural networks based on multi-valued and universal binary neurons: theory, application to image processing and recognition," in *Computational Intelligence, Theory and Applications, International Conference, 6th Fuzzy Days, Dortmund, Germany, May 25-28, 1999, Proceedings*, B. Reusch, Ed., ser. Lecture Notes in Computer Science, vol. 1625, Springer, 1999, pp. 306–316, ISBN: 3-540-66050-X. DOI: 10.1007/3-540-48774-3_36. [Online]. Available: http://dx.doi.org/10.1007/3-540-48774-3_36.

Convolutional Neural Networks



Convolutional Neural Networks



- Used for image classification (typical usage)
- Input filtered using convolutions in order to provide features
- Downsample the image in order to reduce dimension
- Features classified by a classifier (e.g. neural network)
- Features are not hand-crafted but learnt from the data

- Deep Maxpooling CNNs, MLP on top
- 1 year on CPU = 1 week on GPU
- 5×10^9 weight updates/s
- First human-competitive result (0.2% error)[3]
- After a decade of 0.4% error

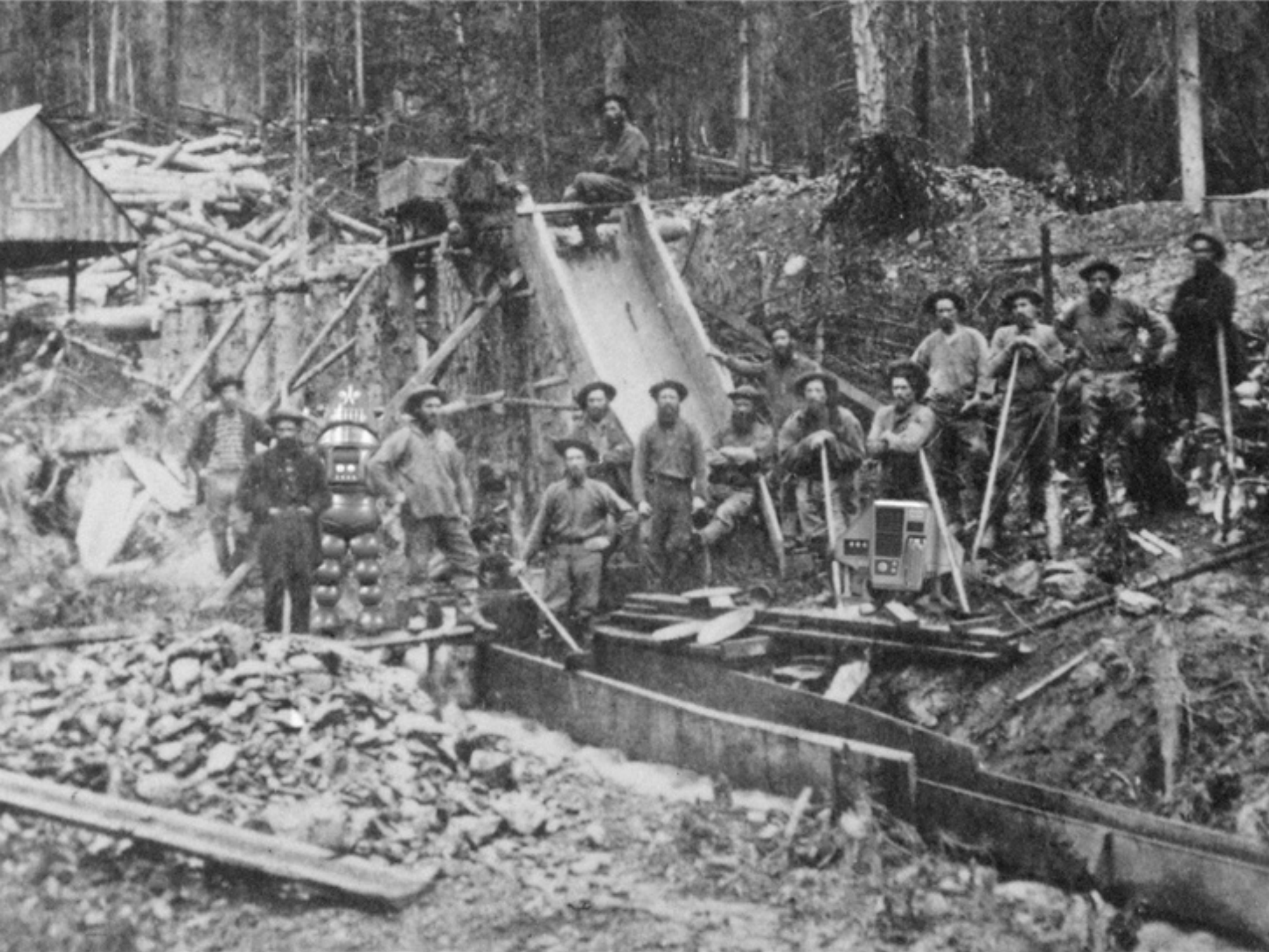
[3] D. C. Ciresan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *IEEE Conference on Computer Vision and Pattern Recognition CVPR 2012*, Long preprint arXiv:1202.2745v1 [cs.CV], 2012.

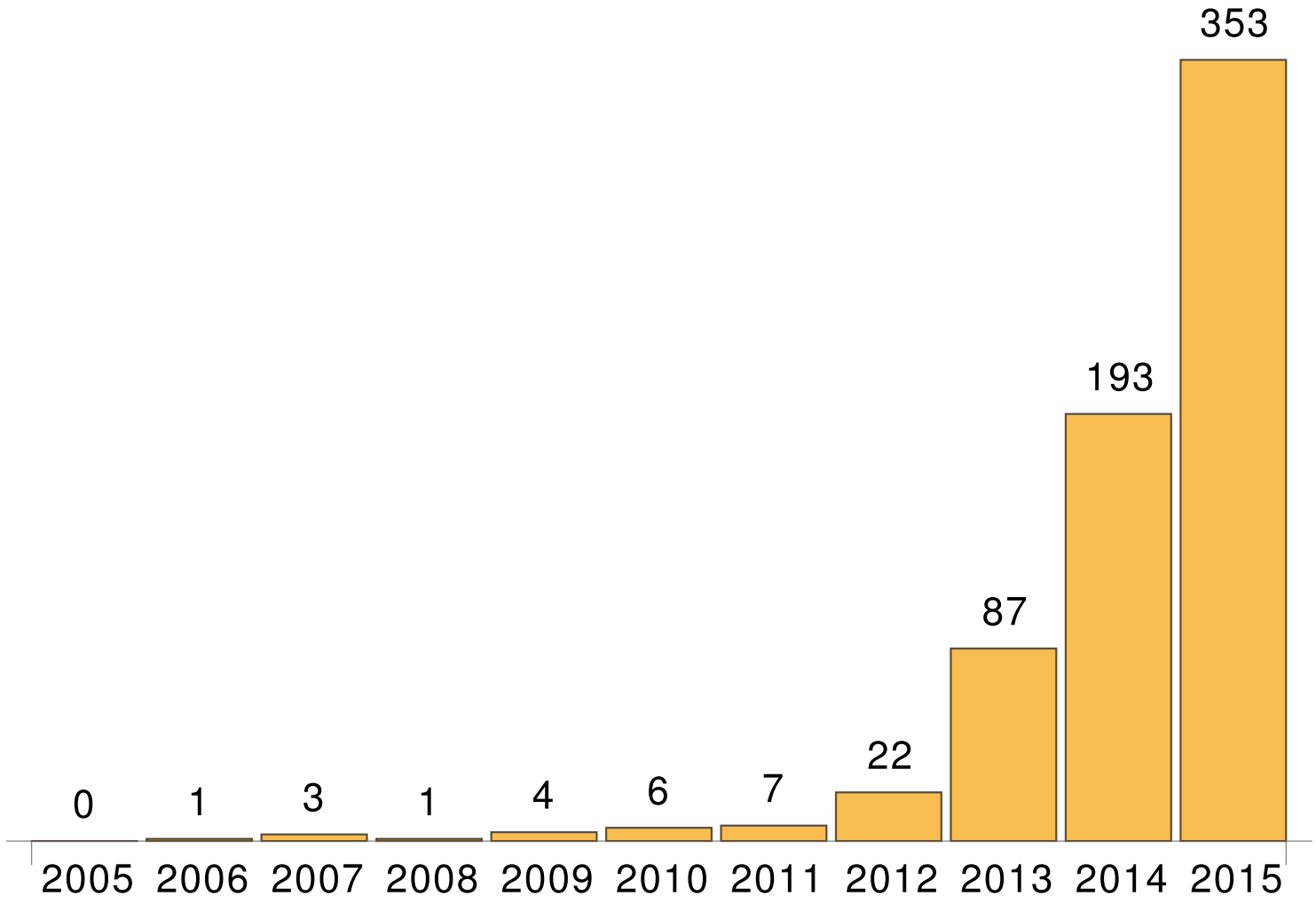
- IJCNN 2011, on site traffic sign recognition
- 1^{st} 0.56%[4,5]
- 2^{nd} 1.16% (Humans)
- 3^{rd} 1.69%
- 4^{th} 3.86%

[4] D. C. Ciresan, U. Meier, J. Masci, *et al.*, "A committee of neural networks for traffic sign classification," in *International Joint Conference on Neural Networks (IJCNN)*, 2011, pp. 1918–1921.

[5] D. C. Ciresan, U. Meier, J. Masci, *et al.*, "Multi-column deep neural network for traffic sign classification," *Neural Networks*, vol. 32, pp. 333–338, 2012.

since 2012 ...

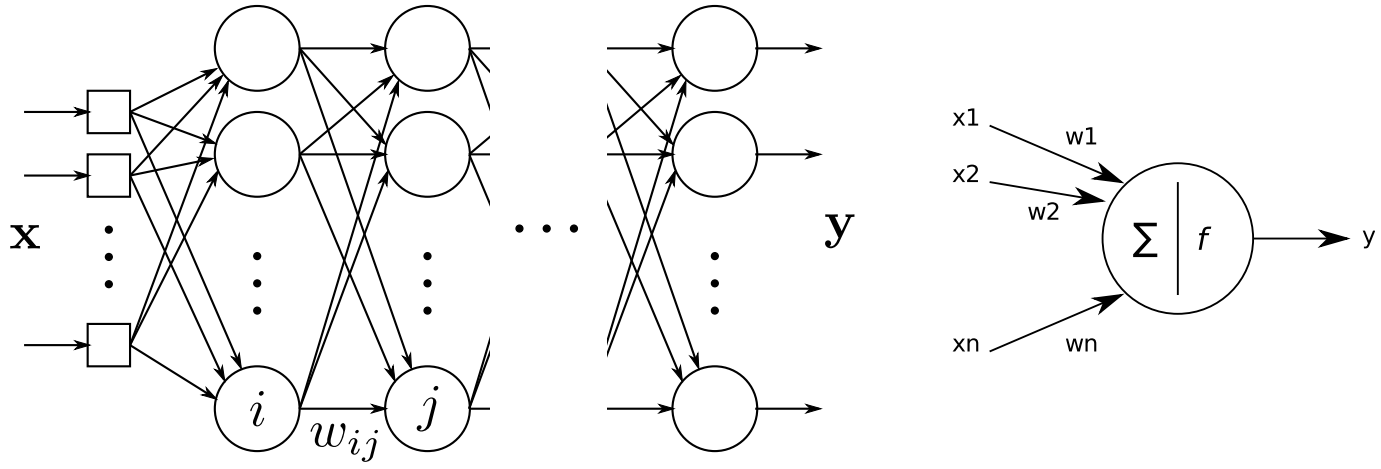




- Search for "Deep Learning" in abstracts on ArXiv[6]

[6] http://arxiv.org

Feed-Forward Artificial Neural Networks



- Neuron: $f(\mathbf{w} \cdot \mathbf{x})$, nonlinearity $f(\cdot)$
- Each connection has a weight w
- Training set consist of N pairs: input $\mathbf{x}_n$, output $\mathbf{z}_n$
- Training changes w in order to match all $\mathbf{y}_n$ to $\mathbf{z}_n$ with a minimal error E
- Optimization problem, gradient defined $\rightarrow$ gradient descent

Gradient Descent

- Define objective function:
  $Q(w) = \sum_{i=1}^{N} Q_i(w)$, for N training examples
- Optimize weights by gradient descent:
  $$w(t+1) = w(t) - \alpha \nabla Q(w(t)) = w(t) - \alpha \sum_{i=1}^{N} \nabla Q_i(w(t))$$
- Stochastic gradient descent (SGD):
  $w(t+1) = w(t) - \alpha \nabla Q_i(w(t))$
- Alternatively, use smaller batch:
  $$w(t+1) = w(t) - \alpha \sum_{i=1}^{B} \nabla Q_i(w(t)),\ B < N$$

Gradient Descent

- Minimize error of N output neurons: $E = \sum_j \frac{1}{2}(z_j - y_z)^2$
- Calculate partial with respect to each weight $\frac{\partial E}{\partial w_{ij}}$
- For output j and input weight w_i: $\frac{\partial E}{\partial w_i} = \frac{\partial(\frac{1}{2}(z-y)^2)}{\partial w_i}$
- Use *chain rule*: $= \frac{\partial(\frac{1}{2}(z-y)^2)}{\partial y}\frac{\partial y}{\partial w_i}$

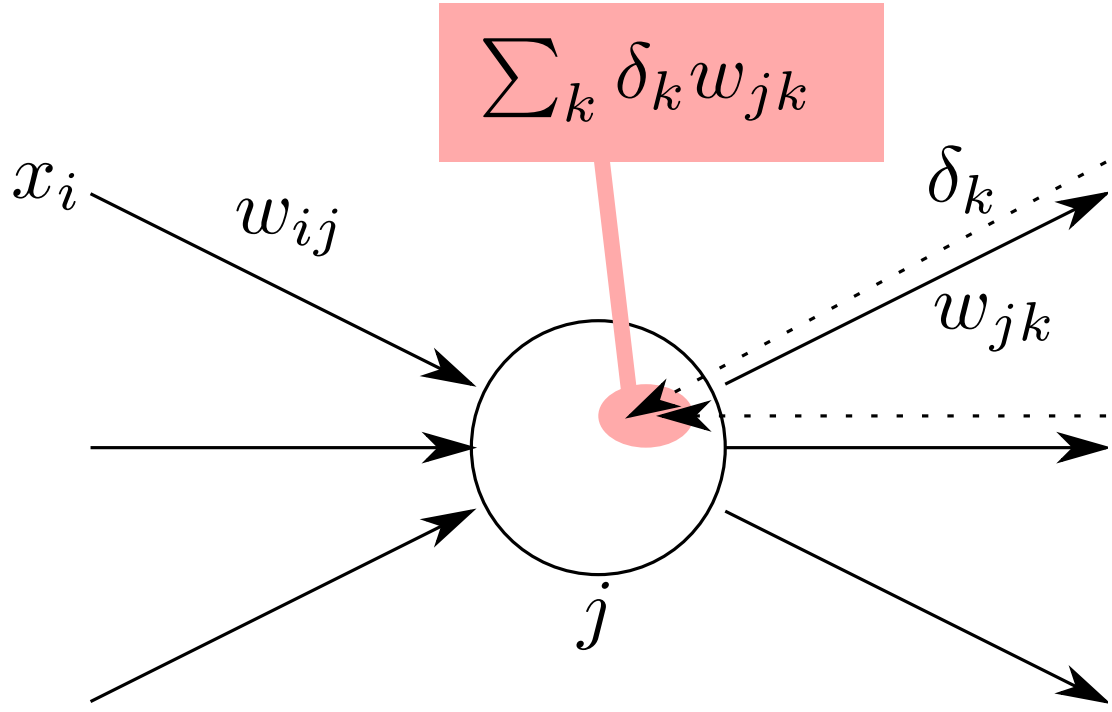
  $= -(z-y)\frac{\partial y}{\partial h}\frac{\partial h}{\partial w_i}$

  $= -(z-y)f'(h)\frac{\partial \mathbf{x}\cdot\mathbf{w}}{\partial w_i}$

  $= -(z-y)f'(h)x_i$
- Change weight by a learning rate α: $\Delta wi = \alpha \underbrace{(z-y)}_{\delta} f'(h)x_i$
- called Delta Rule[7]

[7] B. Widrow and M. Hoff, "Associative storage and retrieval of digital information in networks of adaptive neurons," *Biological Prototypes and Synthetic Systems*, vol. 1, p. 160, 1962.

Multiple Layers



- Each connection weight participates proportionally on the error
- Propagate and sum δ backwards from output nodes k
- $\Delta w_{ij} = \alpha x_i \delta_j = \alpha x_i f'(h_j) \sum_k \delta_k w_{jk}$

Error Backpropagation in Neural Networks

- Euler-LaGrange calculus + dynamic programming (1962)[8]
- Simplified version using chain rule (1962)[9]
- BP in sparse, discrete graphs (1970)[10], first MLP-like BP
- Automatic differentiation (1980)[11]
- BP for MLP (1981)[12][13]
- BP for *recurrent* NNs (1986)[14]

---

[8] A. E. Bryson Jr., "A steepest ascent method for solving optimum programming problems," *Applied Mechanics*, vol. 29, no. 2, p. 247, 1962.

[9] S. E. Dreyfus, "The numerical solution of variational problems," *Journal of Mathematical Analysis and Applications*, vol. 5(1), pp. 30–45, 1962.

[10] S. Linnainmaa, "The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors," Master's thesis, Univ. Helsinki, 1970.

[11] B. Speelpenning, "Compiling fast partial derivatives of functions given by algorithms," PhD thesis, Department of Computer Science, University of Illinois, Urbana-Champaign, Jan. 1980.

[12] P. J. Werbos, "Applications of advances in nonlinear sensitivity analysis," in *Proceedings of the 10th IFIP Conference, 31.8 - 4.9, NYC*, 1981, pp. 762–770.

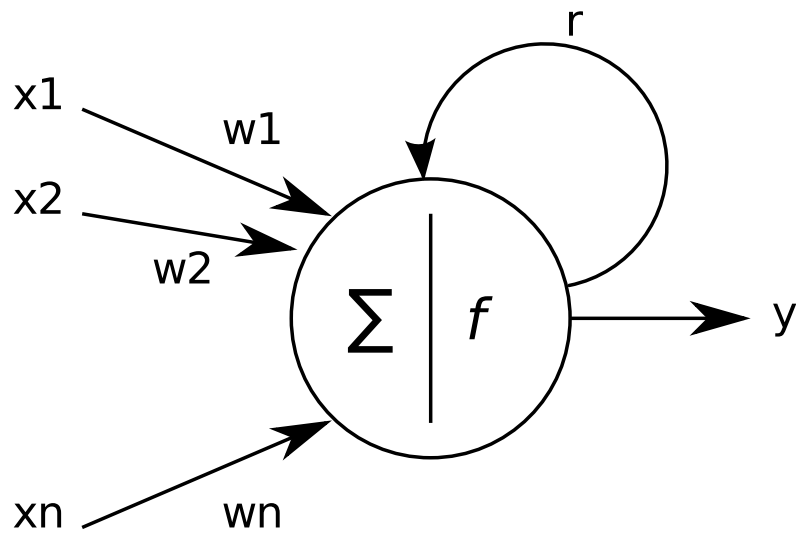
[13] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing*, D. E. Rumelhart and J. L. McClelland, Eds., vol. 1, MIT Press, 1986, pp. 318–362.

[14] D. E. Rumelhart and D. Zipser, "Feature discovery by competitive learning," in *Parallel Distributed Processing*, MIT Press, 1986, pp. 151–193.

Sequence Learning

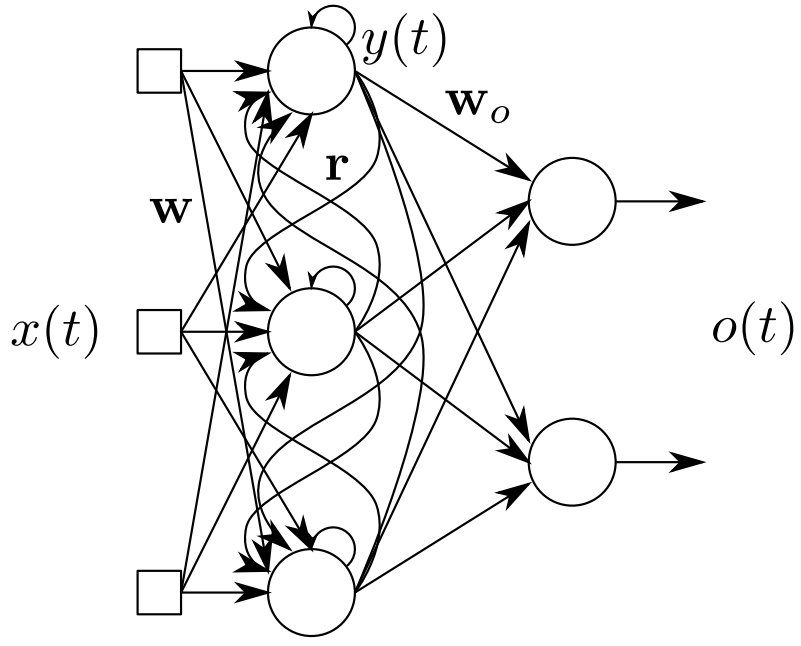
- Up to now, we looked at static mappings:
  $\mathbf{y}(t) = f(\mathbf{x}(t)), \forall t$
- Time t had no meaning
- Used only for the order of presenting inputs $\mathbf{x}$
- It does not matter when $\mathbf{x}$ was presented (except SGD)
- Output $\mathbf{y}$ depends on more than just immediate output
- $\mathbf{y} = f(\mathbf{s}(t)) = F(\mathbf{x}(t), \mathbf{x}(t-1), \ldots, \mathbf{x}(1))$
- where $\mathbf{s}$ is a state changing at every t:
  $\mathbf{s}(t) = g(\mathbf{s}(t-1), \mathbf{x}(t))$
- State is the memory
- Implemented by feedback, recurrent connections

Recurrent Neuron



- $y(t) = f\big(\sum_{i=1}^{I} w_i x_i + r_n y_n(t-1)\big) = f(\mathbf{w} \cdot \mathbf{x} + r \cdot y(t-1))$

Simple Recurrent Network



- $\mathbf{y}(t) = f\big(\sum_{i=1}^{I} w_i x_i + \sum_{n=1}^{N} r_n y_n(t-1)\big) = f(\mathbf{w} \cdot \mathbf{x} + \mathbf{r} \cdot \mathbf{y})$
- $\mathbf{o}(t) = f(\mathbf{w_o} \cdot \mathbf{y}(t))$

Why study sequences?

Natural data is inherently sequential in many cases:

- Speech
- Video
- Natural language
- DNA

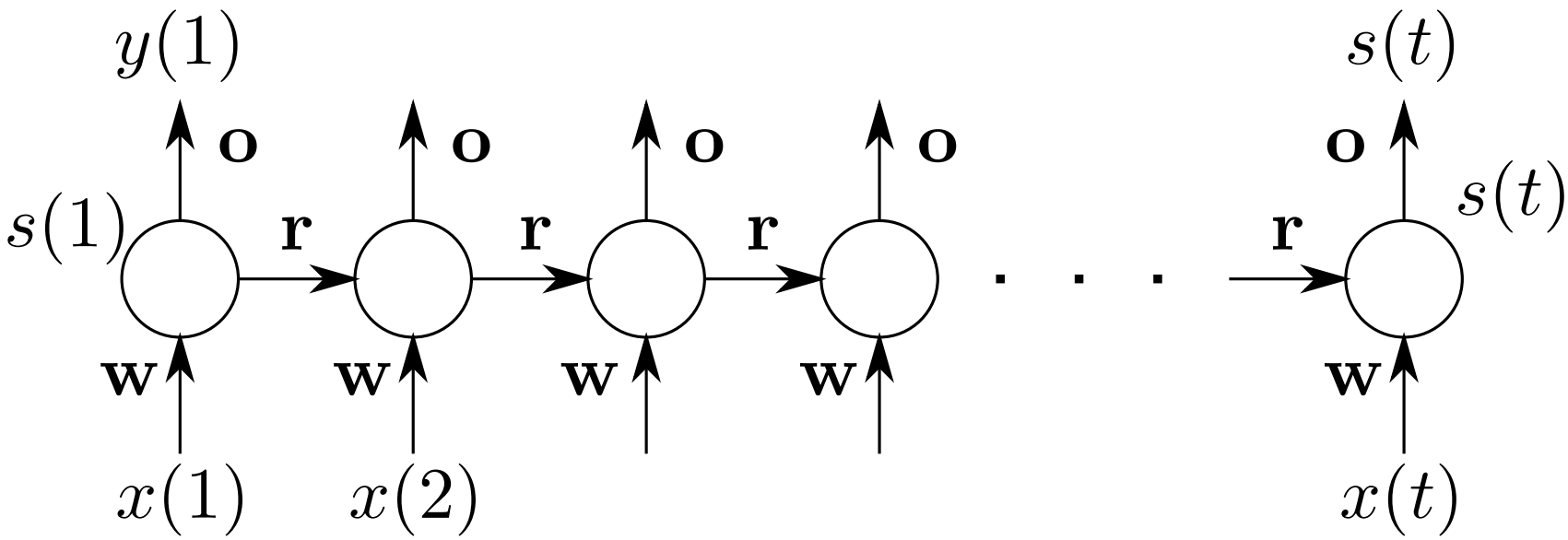
In many cases the memory in important, e.g. for sequential decision making.

- RNNs are general computers[15,16]

[15] H. T. Siegelmann and E. D. Sontag, "Turing computability with neural nets," *Applied Mathematics Letters*, vol. 4, no. 6, pp. 77–80, 1991.

[16] J. Schmidhuber, *Dynamische neuronale Netze und das fundamentale raumzeitliche Lernproblem.* (Dynamic neural nets and the fundamental spatio-temporal credit assignment problem.) *Dissertation, Inst. f. Inf., Tech. Univ. Munich*, 1990.

Backpropagation Through Time



- Unfolding in time[17,18,19]
- Deep Feed-forward network with shared weights across layers

---

[17] R. J. Williams, "Complexity of exact gradient computation algorithms for recurrent neural networks," Boston: Northeastern University, College of Computer Science, Tech. Rep. Technical Report NU-CCS-89-27, 1989.

[18] A. J. Robinson and F. Fallside, "The utility driven dynamic error propagation network," Cambridge University Engineering Department, Tech. Rep. CUED/F-INFENG/TR.1, 1987.

[19] P. J. Werbos, "Generalization of backpropagation with application to a recurrent gas market model," *Neural Networks*, vol. 1, 1988.

Vanishing Gradient Problem

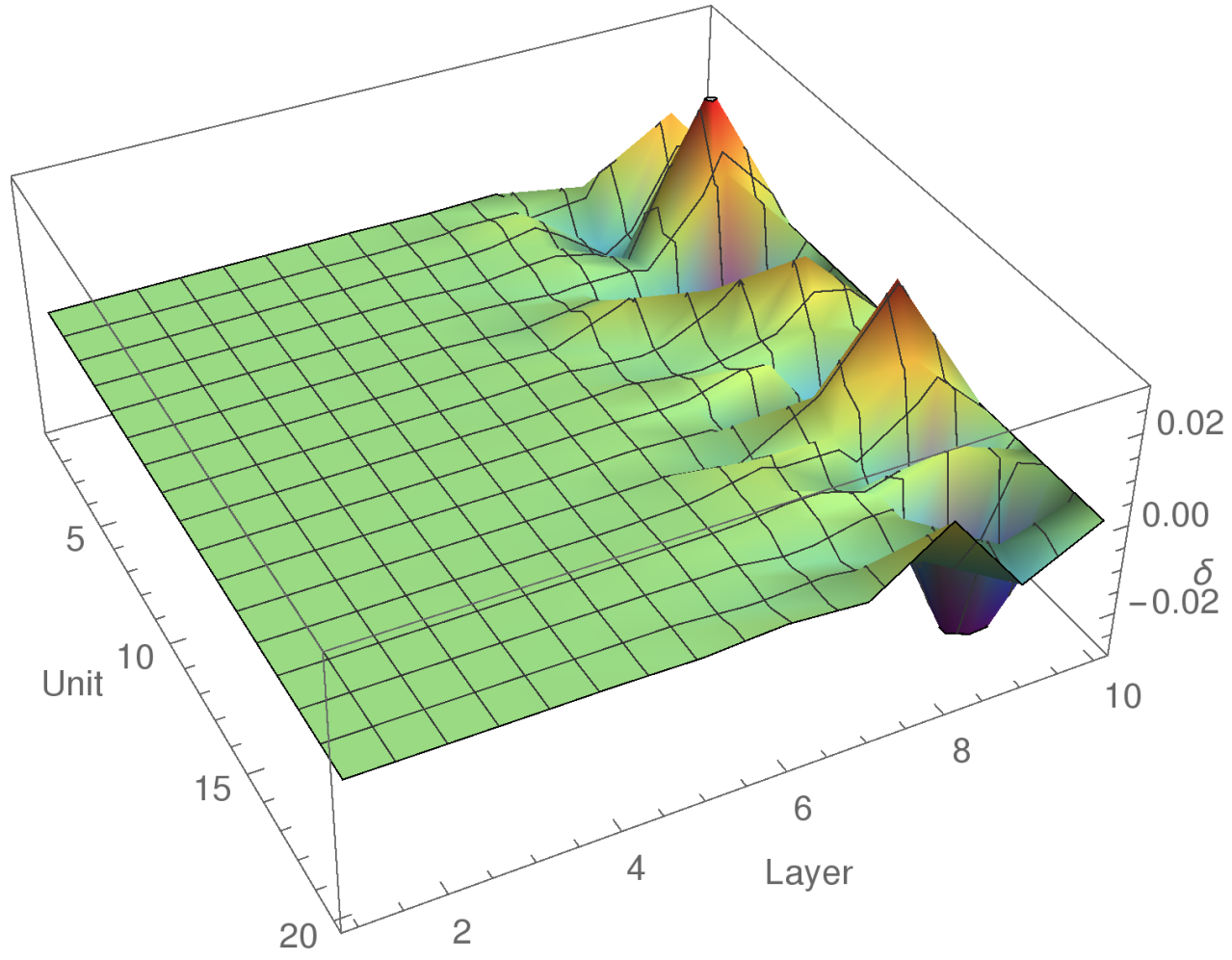
- Training RNNs suffers from dimensionality
- Sequences that span across more than 10 time steps become difficult to train[20,21,22]
- Gradient propagated through non-linearities either vanishes or explodes
- Weights cannot be adjusted to events from the distant past

[20] S. Hochreiter, *Untersuchungen zu dynamischen neuronalen Netzen. Diploma thesis, Institut für Informatik, Lehrstuhl Prof. Brauer, Technische Universität München*, Advisor: J. Schmidhuber, 1991.

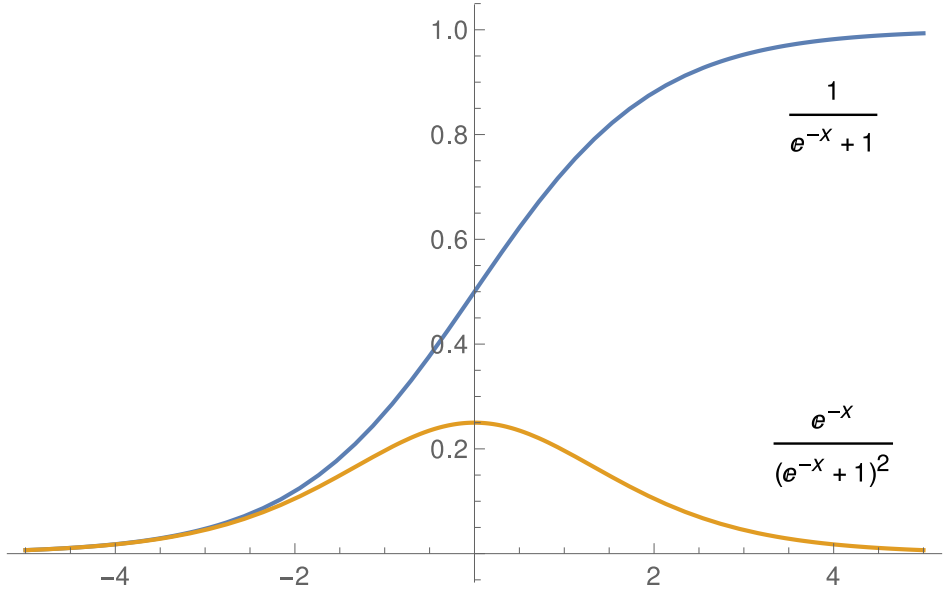
[21] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.

[22] S. Hochreiter, Y. Bengio, P. Frasconi, *et al.*, "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies," in *A Field Guide to Dynamical Recurrent Neural Networks*, S. C. Kremer and J. F. Kolen, Eds., IEEE Press, 2001.

Vanishing Gradient Problem



Vanishing Gradient Problem



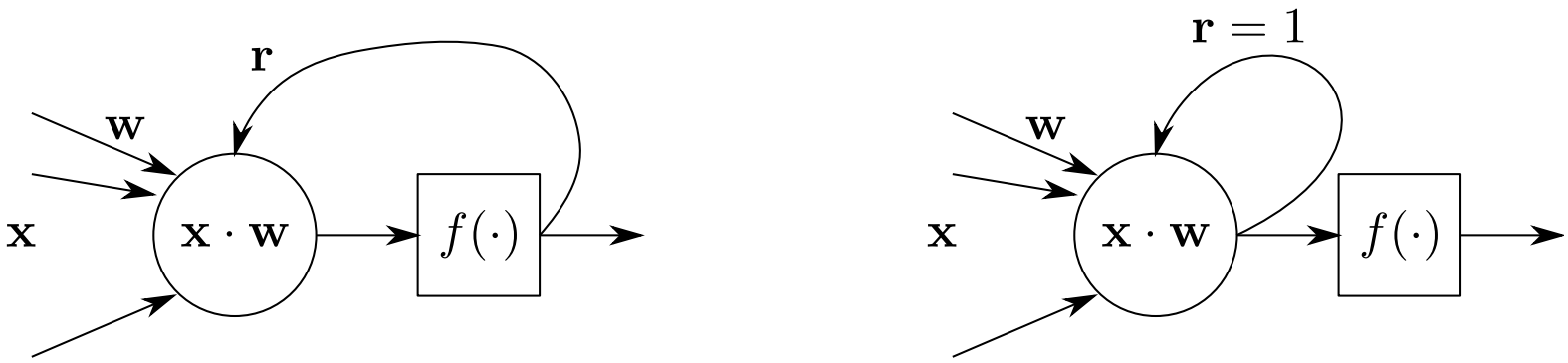
- Error blows up or decays exponentially
- Error tends to vanish when absolute weight values < 4.0
- Large weights do not help (sigmoid derivative goes to zero faster than the weights can grow)
- Increasing number of units does not increase the error flow
- The error decreases even with linear (non-sigmoid) units

Vanishing Gradient Problem

- Recurrent weights $\rightarrow$ matrix power
- If not identity matrix (or a matrix with spectral radius of 1), then
- It either explodes or vanishes
- Simple solution $\rightarrow$ initialize recurrent matrices with identity[23]
- Not as good as LSTM

[23] Q. V. Le, N. Jaitly, and G. E. Hinton, "A simple way to initialize recurrent networks of rectified linear units," *CoRR*, vol. abs/1504.00941, 2015. [Online]. Available: http://arxiv.org/abs/1504.00941.

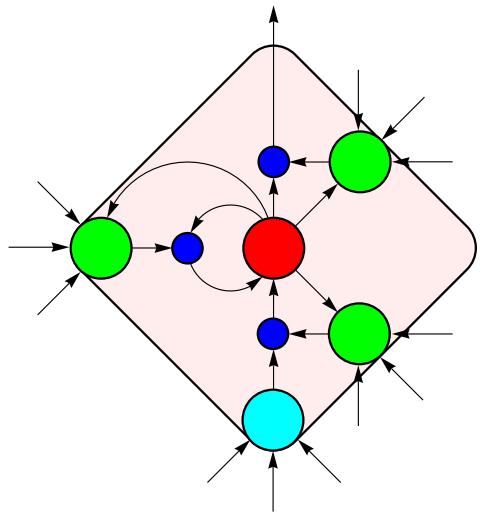
Long Short-Term Memory



- Central idea: **place the recurrent connection before the non-linear function**
- Constant Error Carousel (CEC) - linear cell
- Surround with gating units (multiply the signal with the gate output)[24]

[24] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: continual prediction with LSTM," , IDSIA, Lugano, Lugano, CH, Tech. Rep. IDSIA-01-99, 1999.

Long Short-Term Memory



- Long Short-Term Memory (LSTM)[25,26]
- Architectural solution vanishing gradient problem
- Detailed analysis in the *Odyssey* paper[27]

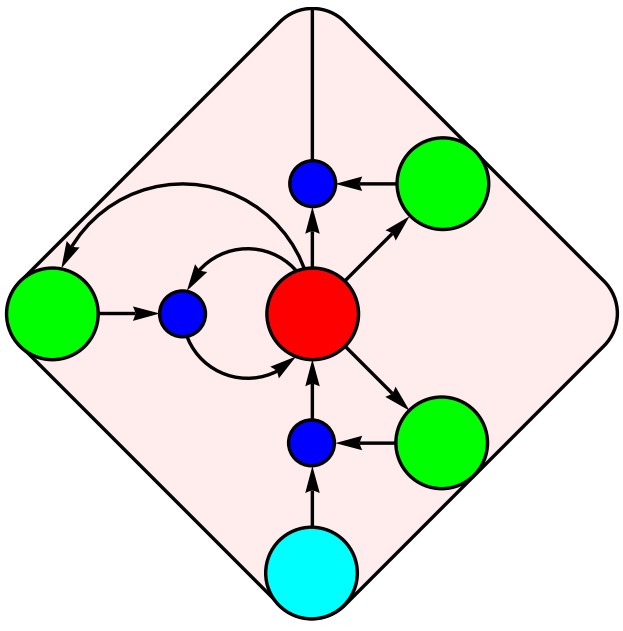
---

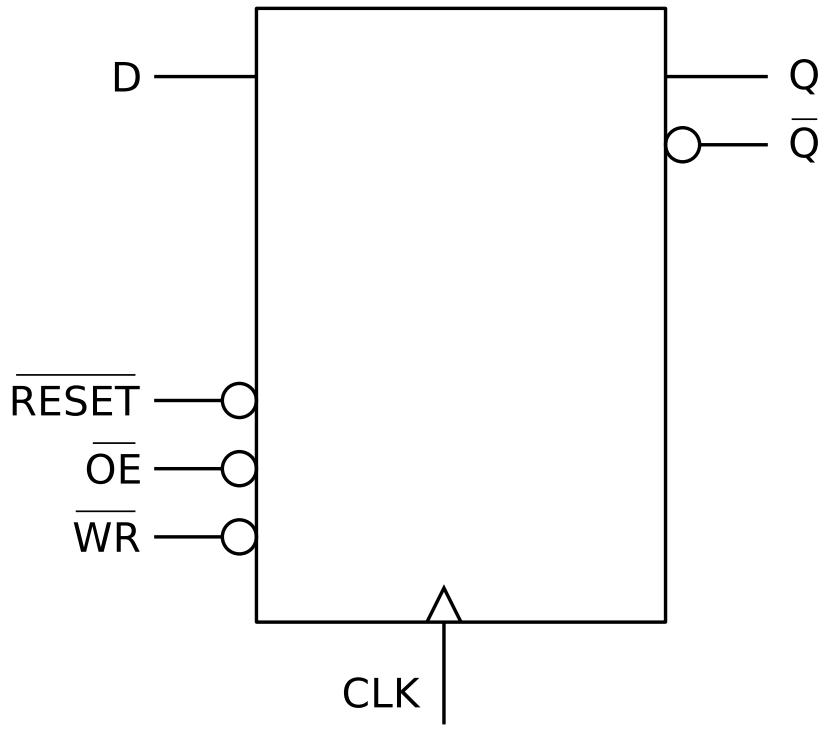
[25] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Fakultät für Informatik, Technische Universität München, Tech. Rep. FKI-207-95, 1995, Revised 1996 (see www.idsia.ch/~juergen, www7.informatik.tu-muenchen.de/~hochreit).

[26] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

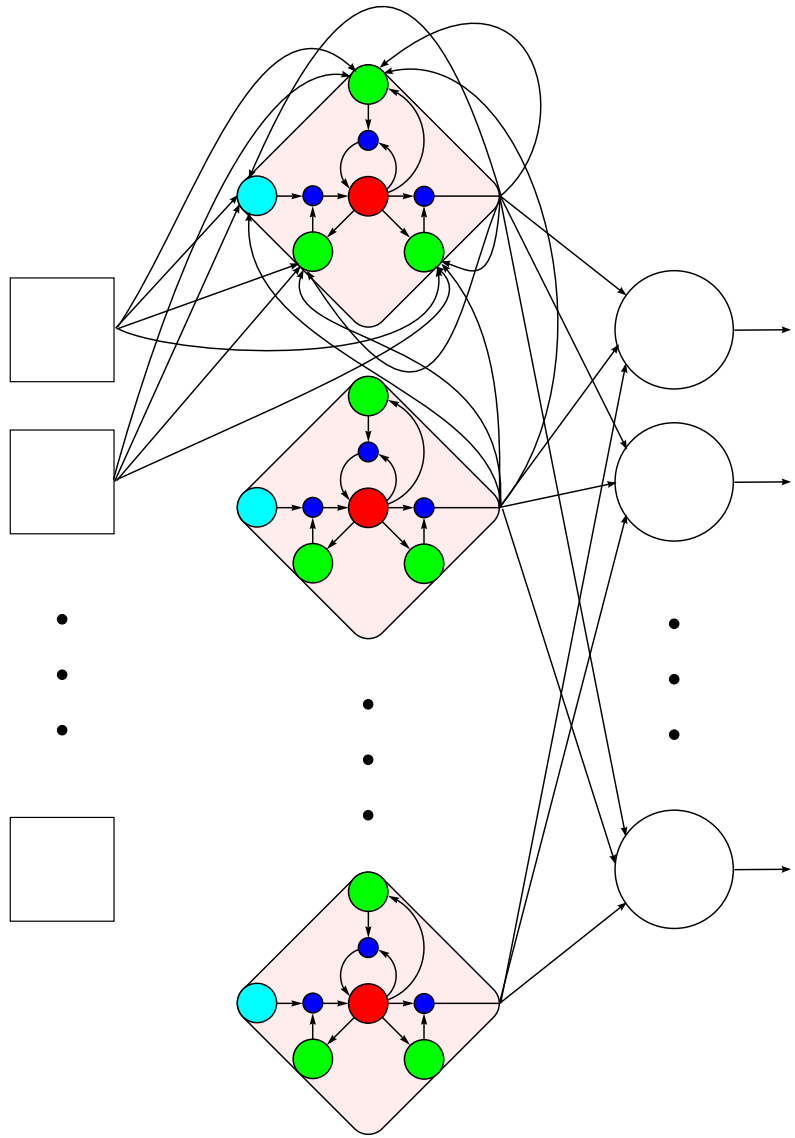
[27] K. Greff, R. K. Srivastava, J. Koutník, *et al.*, "LSTM: A search space odyssey," *CoRR*, vol. abs/1503.04069, 2015. [Online]. Available: `http://arxiv.org/abs/1503.04069`.

Long Short-Term Memory

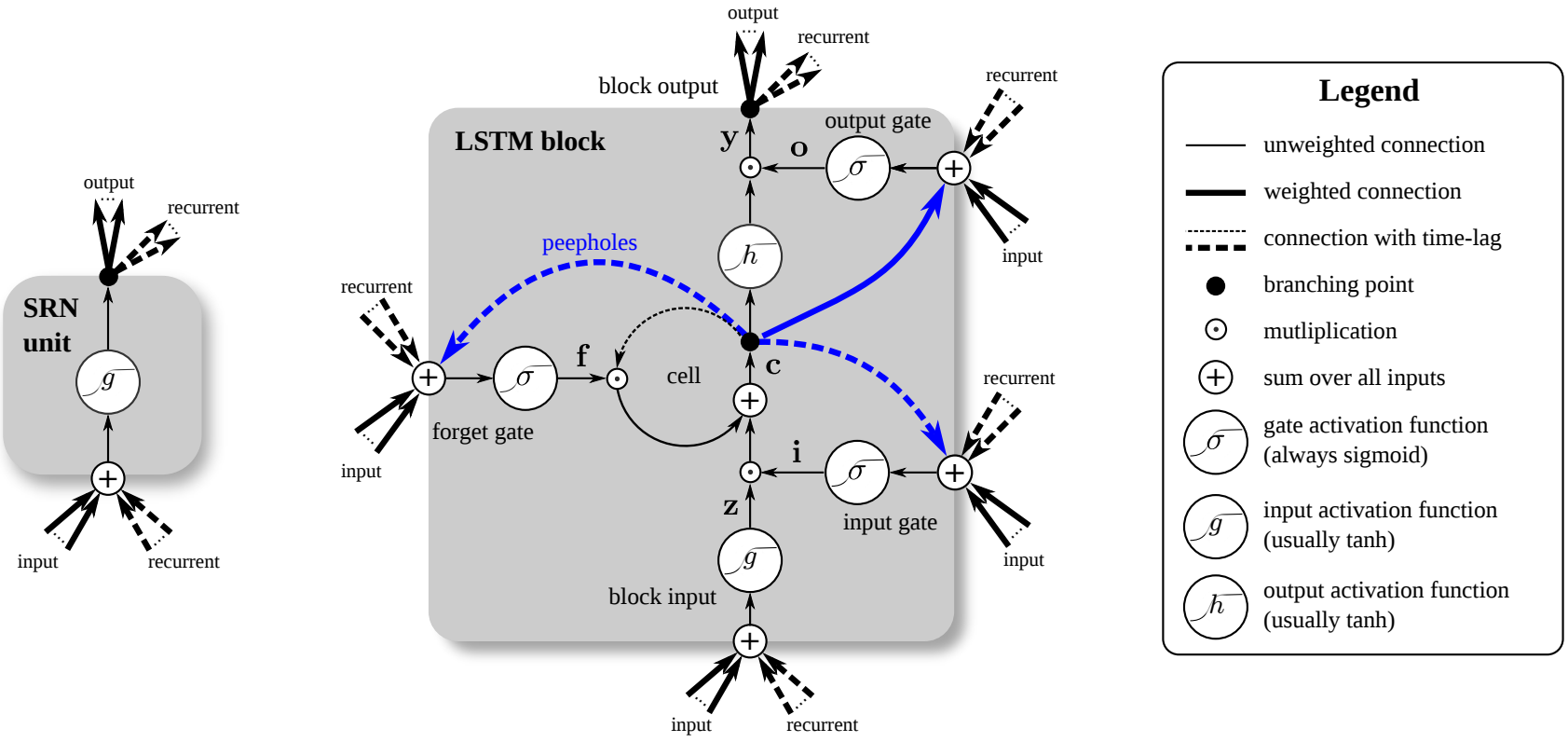




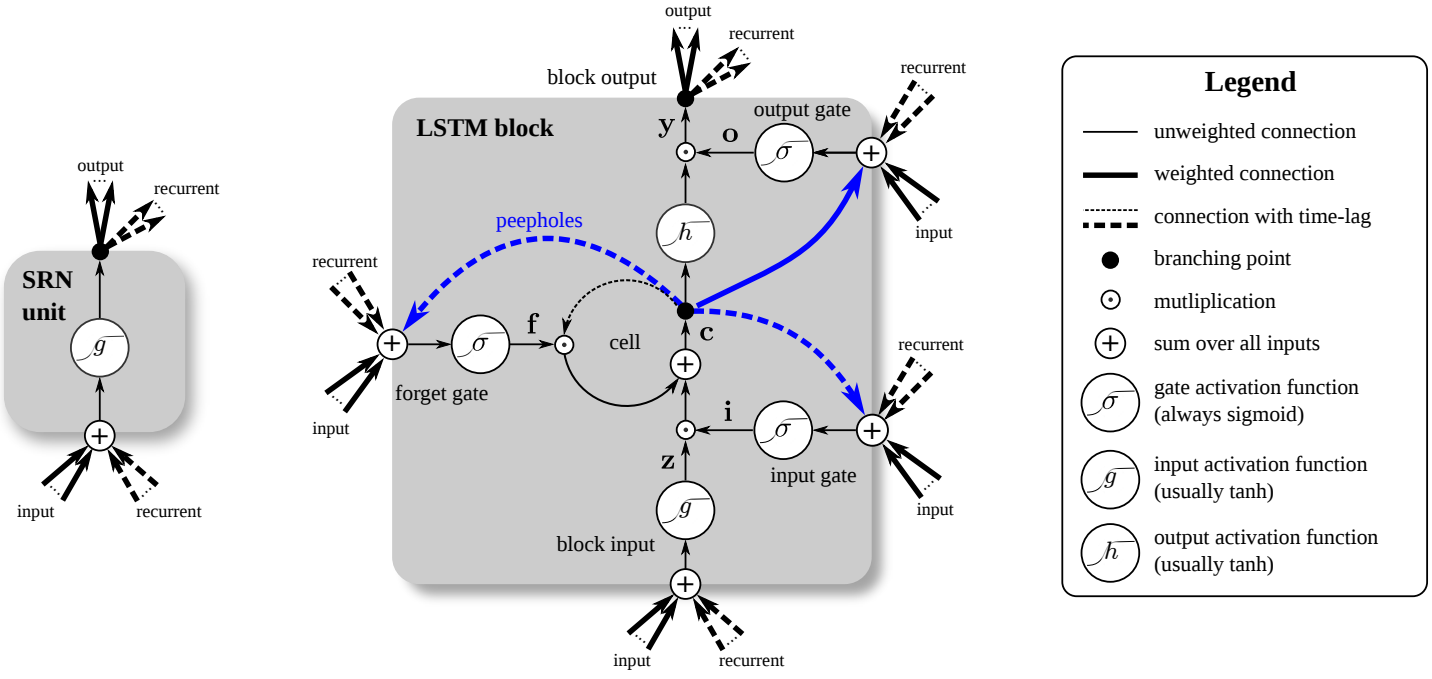
Long Short-Term Memory



LSTM Forward Pass



LSTM Forward Pass



$$\bar{\mathbf{z}}^t = \mathbf{W}_z \mathbf{x}^t + \mathbf{R}_z \mathbf{y}^{t-1} + \mathbf{b}_z \qquad \mathbf{z}^t = g(\bar{\mathbf{z}}^t) \qquad \textit{block input}$$

$$\bar{\mathbf{i}}^t = \mathbf{W}_i \mathbf{x}^t + \mathbf{R}_i \mathbf{y}^{t-1} + \mathbf{p}_i \odot \mathbf{c}^{t-1} + \mathbf{b}_i \qquad \mathbf{i}^t = \sigma(\bar{\mathbf{i}}^t) \qquad \textit{input gate}$$

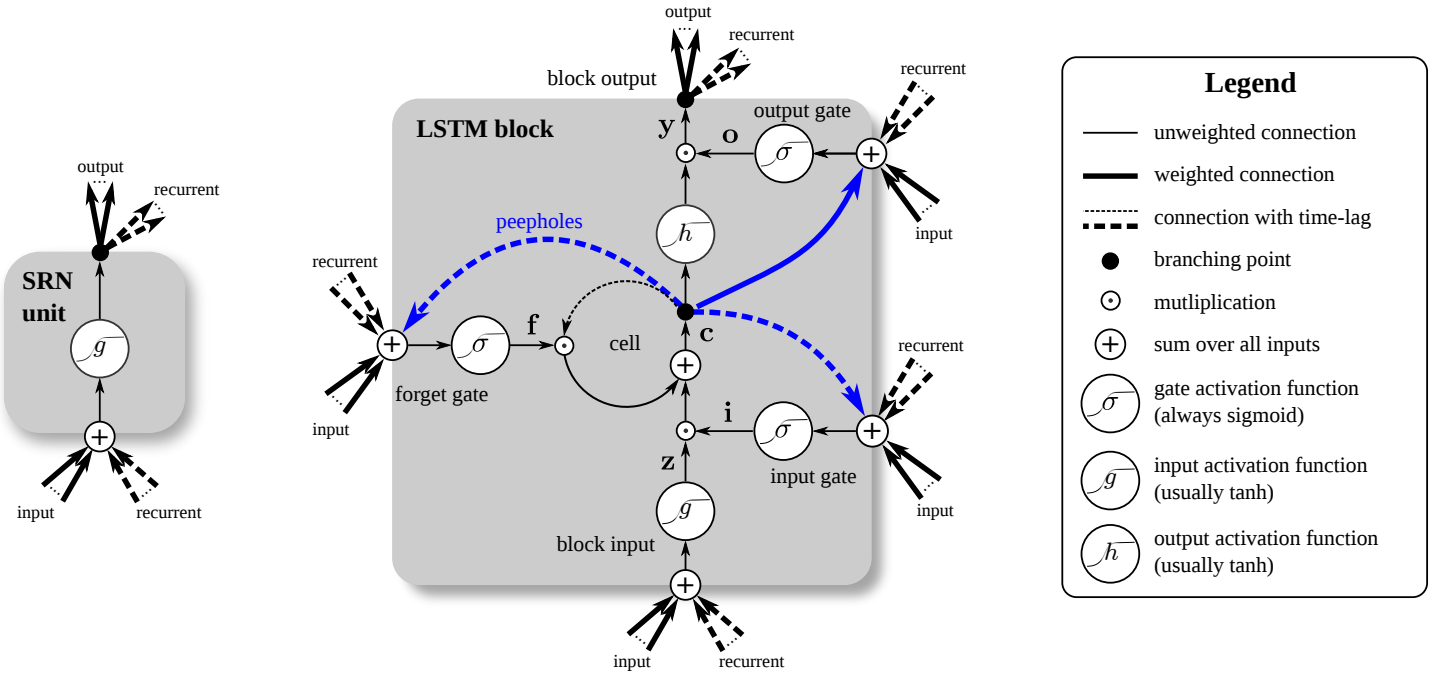
$$\bar{\mathbf{f}}^t = \mathbf{W}_f \mathbf{x}^t + \mathbf{R}_f \mathbf{y}^{t-1} + \mathbf{p}_f \odot \mathbf{c}^{t-1} + \mathbf{b}_f \qquad \mathbf{f}^t = \sigma(\bar{\mathbf{f}}^t) \qquad \textit{forget gate}$$

$$\mathbf{c}^t = \mathbf{z}^t \odot \mathbf{i}^t + \mathbf{c}^{t-1} \odot \mathbf{f}^t \qquad \textit{cell}$$

$$\bar{\mathbf{o}}^t = \mathbf{W}_o \mathbf{x}^t + \mathbf{R}_o \mathbf{y}^{t-1} + \mathbf{p}_o \odot \mathbf{c}^t + \mathbf{b}_o \qquad \mathbf{o}^t = \sigma(\bar{\mathbf{o}}^t) \qquad \textit{output gate}$$

$$\mathbf{y}^t = h(\mathbf{c}^t) \odot \mathbf{o}^t \qquad \textit{block output}$$

LSTM Backward Pass



$$\delta \mathbf{y}^t = \Delta^t + \mathbf{R}_z^T \delta \mathbf{z}^{t+1} + \mathbf{R}_i^T \delta \mathbf{i}^{t+1} + \mathbf{R}_f^T \delta \mathbf{f}^{t+1} + \mathbf{R}_o^T \delta \mathbf{o}^{t+1}$$

$$\delta \mathbf{o}^t = \delta \mathbf{y}^t \odot h(\mathbf{c}^t) \odot \sigma'(\bar{\mathbf{o}}^t)$$

$$\delta \mathbf{c}^t = \delta \mathbf{y}^t \odot \mathbf{o}^t \odot h'(\mathbf{c}^t) + \mathbf{p}_o \odot \delta \mathbf{o}^t + \mathbf{p}_i \odot \delta \mathbf{i}^{t+1} + \mathbf{p}_f \odot \delta \mathbf{f}^{t+1} + \delta \mathbf{c}^{t+1} \odot \mathbf{f}^{t+1}$$

$$\delta \mathbf{f}^t = \delta \mathbf{c}^t \odot \mathbf{c}^{t-1} \odot \sigma'(\bar{\mathbf{f}}^t)$$

$$\delta \mathbf{i}^t = \delta \mathbf{c}^t \odot \mathbf{z}^t \odot \sigma'(\bar{\mathbf{i}}^t)$$

$$\delta \mathbf{z}^t = \delta \mathbf{c}^t \odot \mathbf{i}^t \odot g'(\bar{\mathbf{z}}^t)$$

Gradient Calculation

$$\delta \mathbf{W}_\star = \sum_{t=0}^{T} \langle \delta\star^t, \mathbf{x}^t \rangle \qquad \delta \mathbf{p}_i = \sum_{t=0}^{T-1} \mathbf{c}^t \odot \delta \mathbf{i}^{t+1}$$

$$\delta \mathbf{R}_\star = \sum_{t=0}^{T-1} \langle \delta\star^{t+1}, \mathbf{y}^t \rangle \qquad \delta \mathbf{p}_f = \sum_{t=0}^{T-1} \mathbf{c}^t \odot \delta \mathbf{f}^{t+1}$$

$$\delta \mathbf{b}_\star = \sum_{t=0}^{T} \delta\star^t \qquad \delta \mathbf{p}_o = \sum_{t=0}^{T} \mathbf{c}^t \odot \delta \mathbf{o}^t$$

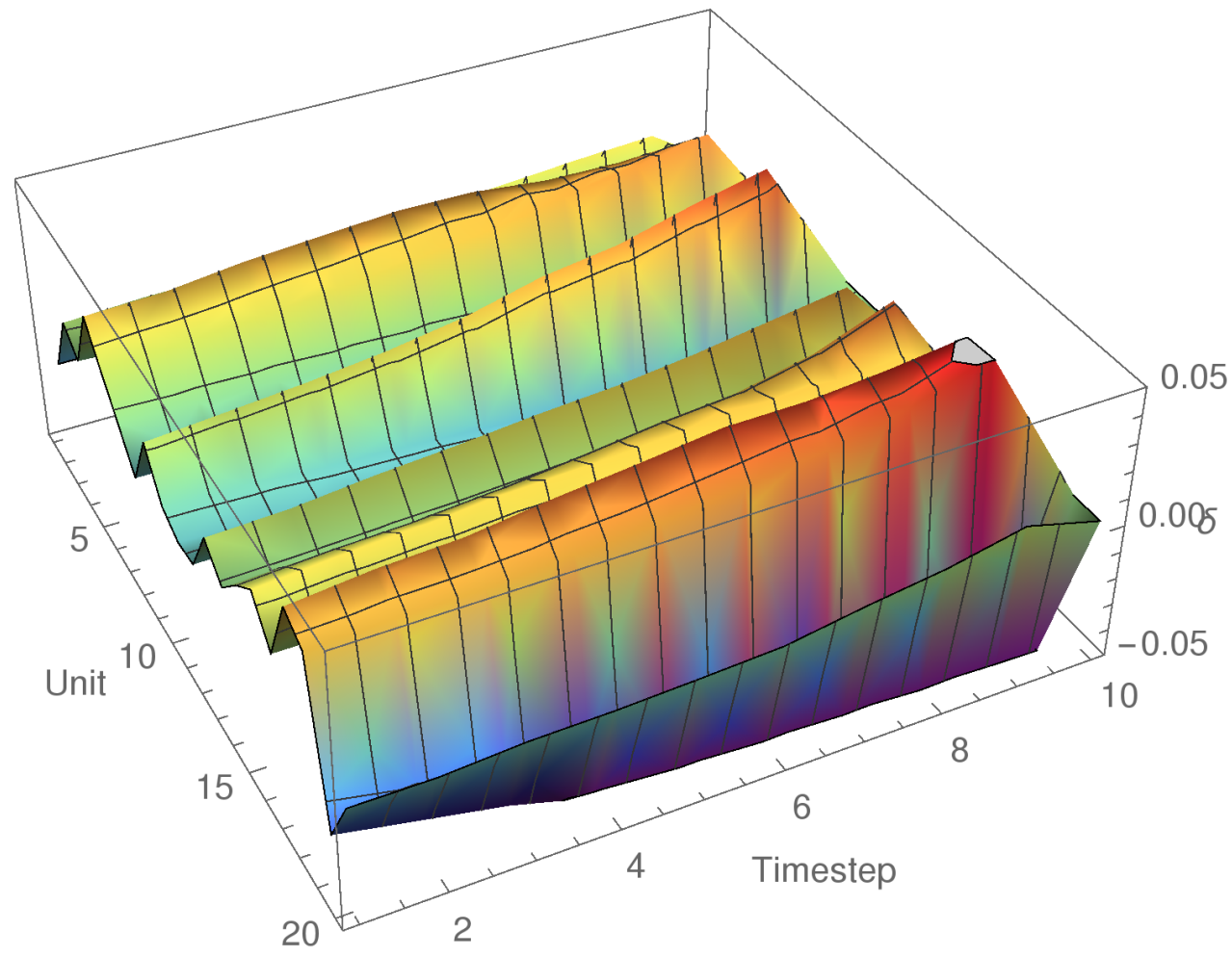
LSTM Properties

- Learn order from separated events in noisy environments
- Robust storage of high precision numbers across time intervals
- Recall information separated by long temporal lags
- Can learn regular and context free languages[28]
- Biologically plausible[29]

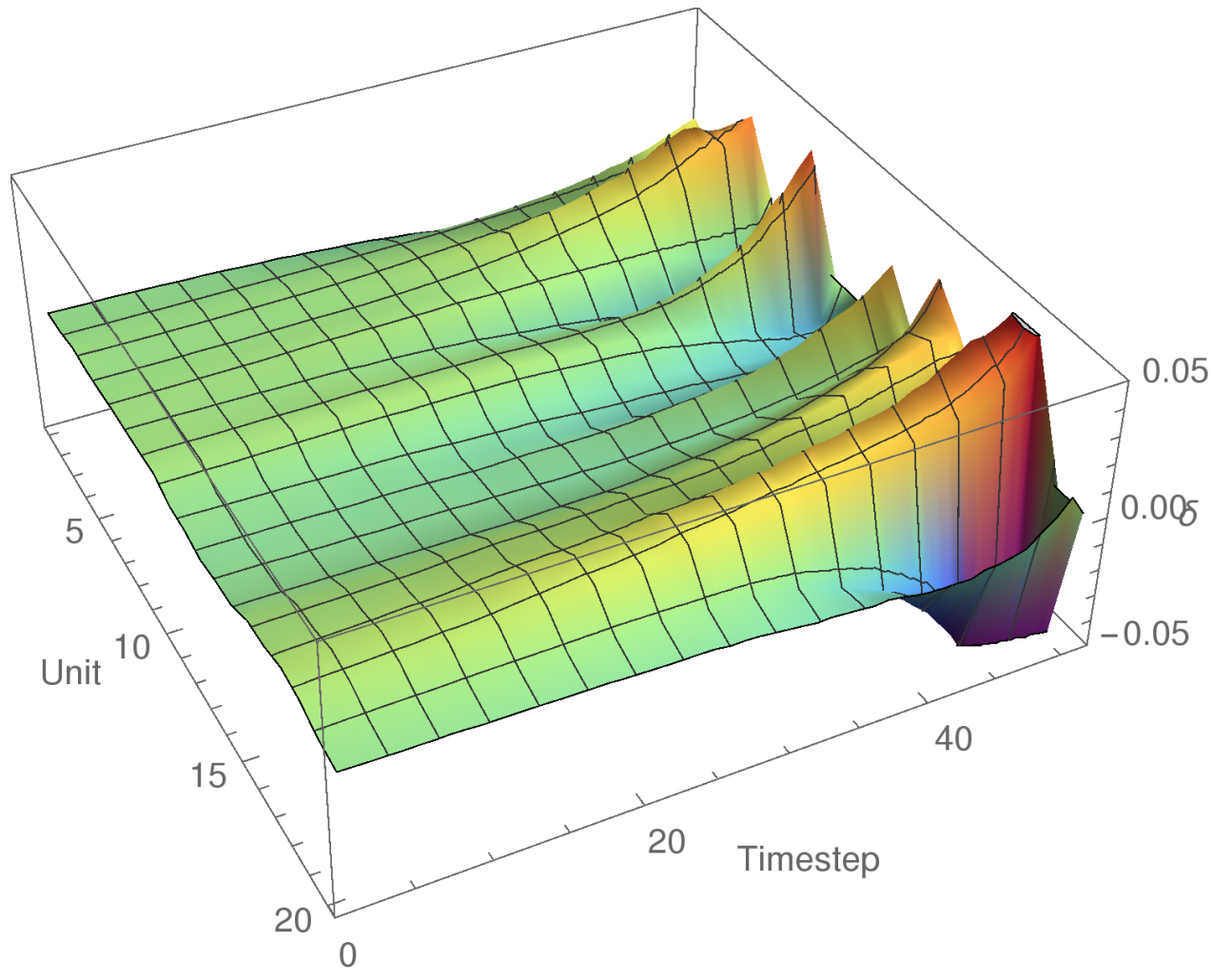
[28] F. A. Gers and J. Schmidhuber, "Long Short-Term Memory learns context free and context sensitive languages," in *Proceedings of the ICANNGA 2001 Conference*, K. et./ al./, Ed., vol. 1, Wien, NY: Springer, 2001, pp. 134–137.

[29] R. O'Reilly, "Making working memory work: a computational model of learning in the prefrontal cortex and basal ganglia," ICS, Tech. Rep. ICS-03-03, Jun. 2003.

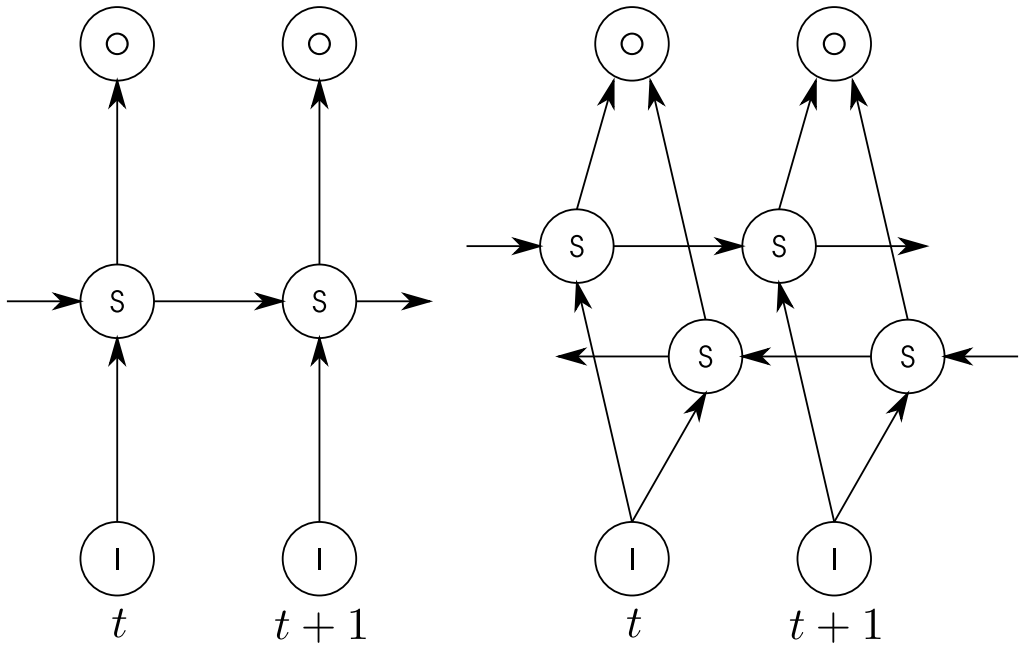
Gradient Flow in LSTM Cells



Gradient Flow in LSTM Cells



Bidirectional RNNs

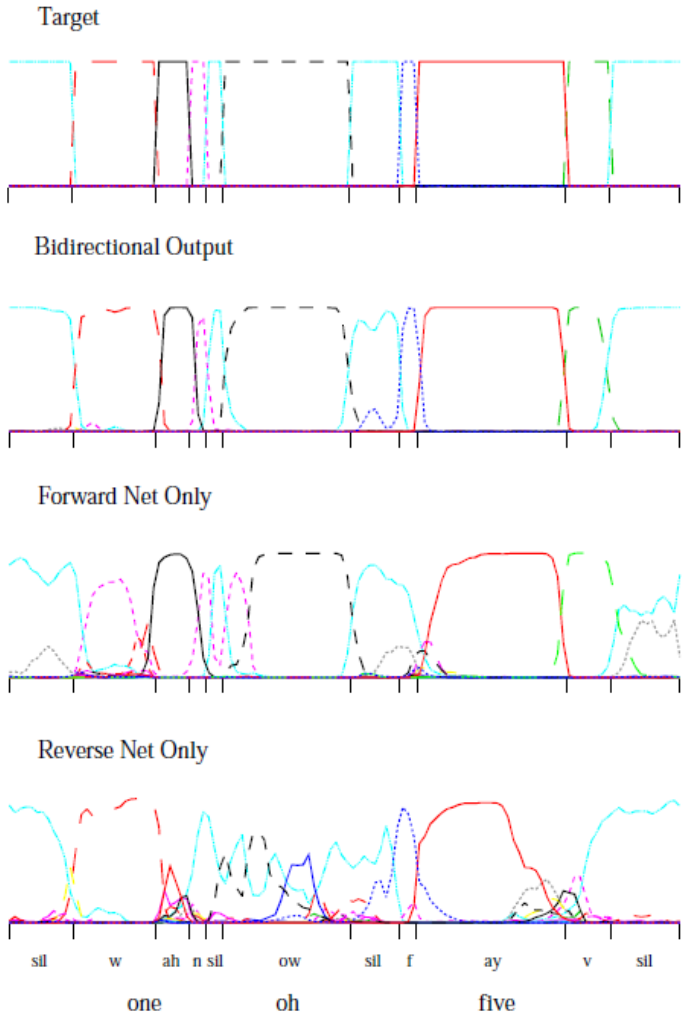


- Time flows in both directions[30,31]
- At each t context from past and future is available

[30] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, pp. 2673–2681, Nov. 1997.

[31] P. Baldi, S. Brunak, P. Frasconi, *et al.*, "Exploiting the past and the future in protein secondary structure prediction," *Bioinformatics*, vol. 15, pp. 937–946, 1999.

Bidirectional LSTM



- Phoneme classification in speech data[32]

[32] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Networks*, vol. 18, no. 5-6, pp. 602–610, 2005.

Sequences of Different Lengths

- Connectionist Temporal Classification (CTC)[33]
- Encoder—Decoder[34,35,36]
- Grid LSTM[37]

---

[33] A. Graves, S. Fernandez, F. J. Gomez, *et al.*, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural nets," in *ICML'06: Proceedings of the 23rd International Conference on Machine Learning*, 2006, pp. 369–376.

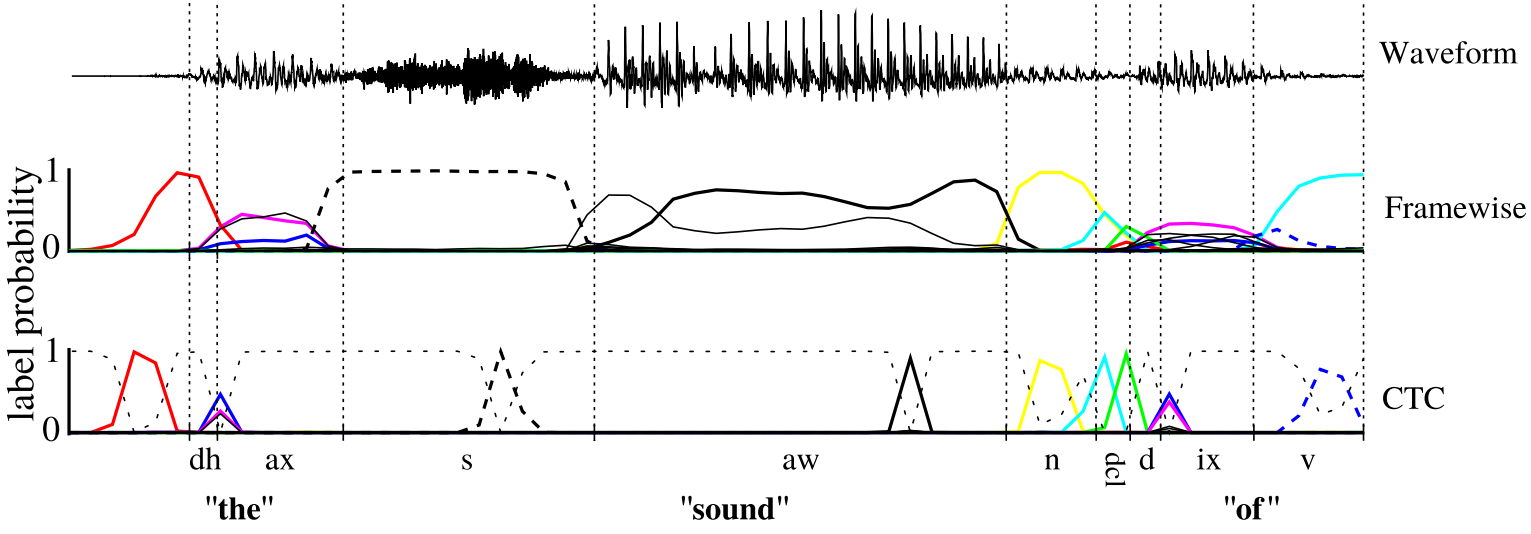
[34] K. Cho, B. van Merrienboer, D. Bahdanau, *et al.*, "On the properties of neural machine translation: encoder-decoder approaches," *CoRR*, vol. abs/1409.1259, 2014. [Online]. Available: http://arxiv.org/abs/1409.1259.

[35] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *CoRR*, vol. abs/1409.3215, 2014. [Online]. Available: http://arxiv.org/abs/1409.3215.

[36] K. Cho, B. van Merrienboer, Ç. Gülçehre, *et al.*, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *CoRR*, vol. abs/1406.1078, 2014. [Online]. Available: http://arxiv.org/abs/1406.1078.

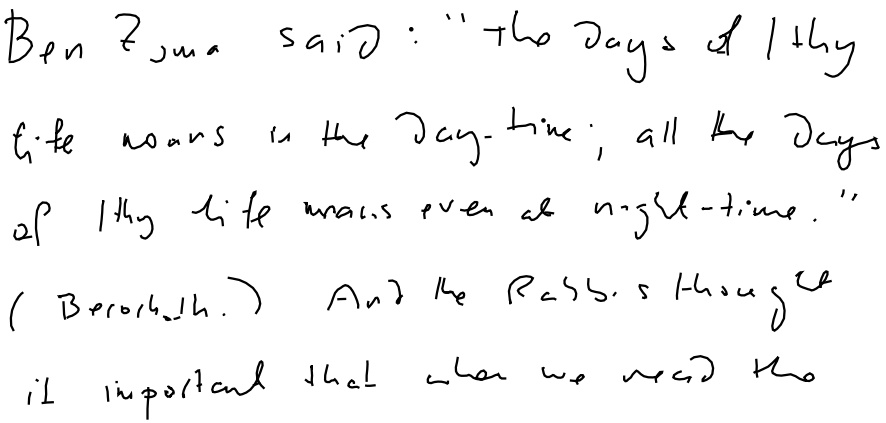
[37] N. Kalchbrenner, I. Danihelka, and A. Graves, "Grid long short-term memory," *CoRR*, vol. abs/1507.01526, 2015. [Online]. Available: http://arxiv.org/abs/1507.01526.

Connectionist Temporal Classification



- Automatically transforms frame labels to classification segments
- Provides labels for all time steps
- Allows to back-propagate the error from sparse labels

Bidirectional LSTM



Ben Zoma said: "The days of 1thy
life means in the day-time; all the days
of 1thy life means even at night-time ."
(Berochoth .) And the Rabbis thought
it important that when we read the

- Online handwriting[38]
- 11.5% Character error rate with extensive preprocessing in 2009[39]
- State-of-the-art results in 2015 (9.26%)[40]

---

[38] M. Liwicki and H. Bunke, "IAM-OnDB – an on-line English sentence database acquired from handwritten text on a whiteboard," in *Proc. 8th Int. Conf. on Document Analysis and Recognition*, vol. 2, 2005, pp. 956–961.

[39] A. Graves, S. Fernandez, M. Liwicki, *et al.*, "Unconstrained on-line handwriting recognition with recurrent neural networks," in *Advances in Neural Information Processing Systems (NIPS) 20*, J. Platt, D. Koller, Y. Singer, *et al.*, Eds., Cambridge, MA: MIT Press, 2008, pp. 577–584.

[40] K. Greff, R. K. Srivastava, J. Koutník, *et al.*, "LSTM: A search space odyssey," *CoRR*, vol. abs/1503.04069, 2015. [Online]. Available: http://arxiv.org/abs/1503.04069.

2014 – The year of LSTM

Best results achieved using LSTM

1. Large vocabulary speech recognition (Sak et al., Google, Interspeech)
2. English to French translation (Sutskever et al., Google, NIPS)
3. Text-to-speech synthesis (Fan et al., Microsoft, Interspeech)
4. Prosody contour prediction (Fernandez et al., IBM, Interspeech)
5. Language identification (Gonzalez-Dominguez et al., Google, Interspeech)
6. Medium vocabulary speech recognition (Geiger et al., Interspeech)
7. Audio onset detection (Marchi et al., ICASSP)
8. Social signal classification (Brueckner & Schulter, ICASSP)
9. Arabic handwriting recognition (Bluche et al., DAS)
10. Image caption generation (Vinyals et al., Google)
11. Video to textual description (Donahue et al.)

2015 is no Exception

1. Image caption generation (Xu et al.)
2. Scene labeling (Byeon et al., CVPR)
3. Subcellular localization of proteins (Sonderby et al.)
4. Language understanding (Peng et al.)
5. ...

[41] http://googleresearch.blogspot.ch/2015/09/google-voice-search-faster-and-more.html

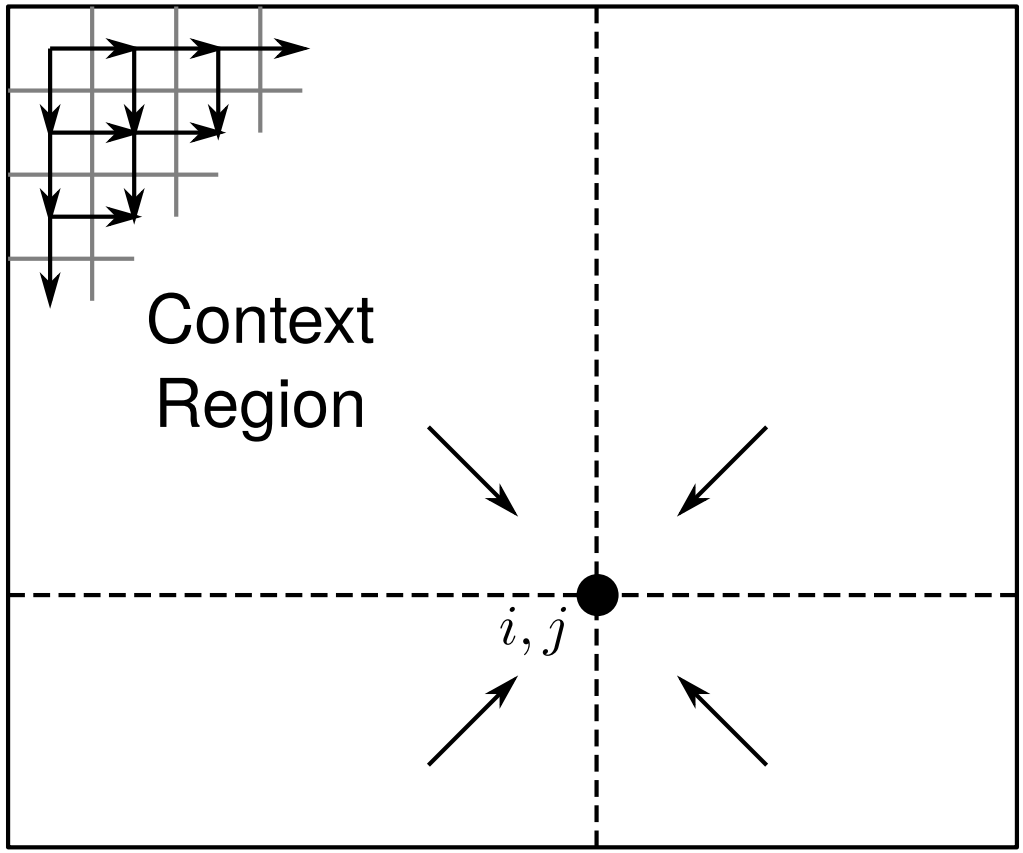
2015 is no Exception

1. Image caption generation (Xu et al.)
2. Scene labeling (Byeon et al., CVPR)
3. Subcellular localization of proteins (Sonderby et al.)
4. Language understanding (Peng et al.)
5. ...

- LSTM&CTC replaces HMMs in Google Voice[41] speech recognition on Android devices (1B+ installations), September 2015.

[41] http://googleresearch.blogspot.ch/2015/09/google-voice-search-faster-and-more.html

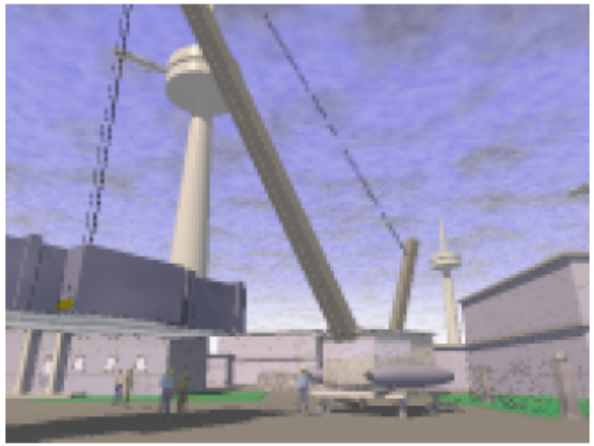
Multidimensional RNNs

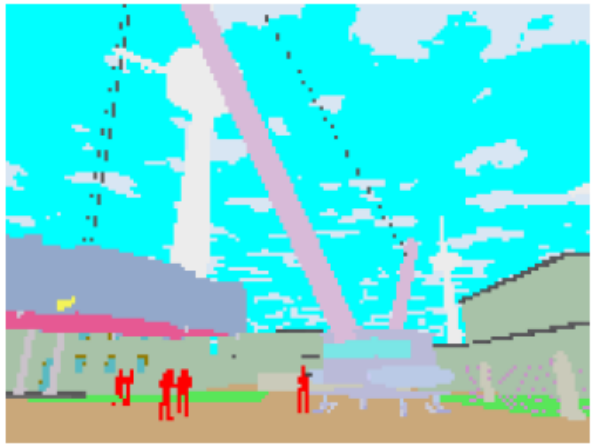


- Input array (image) treated as N-dimensional signal[42]
- 2^N contexts available at each *time* (point)

[42] A. Graves, S. Fernández, and J. Schmidhuber, "Multi-dimensional recurrent neural networks," in *Proceedings of the 17th International Conference on Artificial Neural Networks*, Sep. 2007.

Multidimensional LSTM





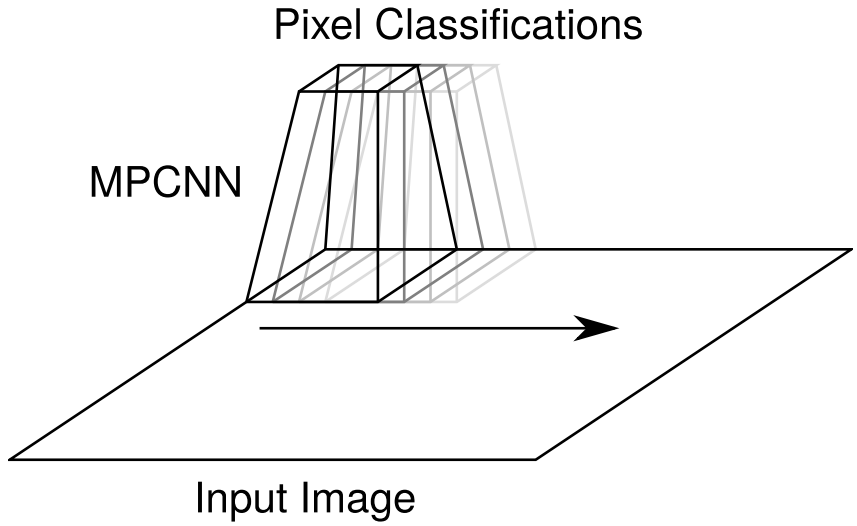
- Recurrent neuron replaced by LSTM unit
- Offline handwriting recognition[43]
- Image segmentation[44]
- Better than scanning MPCNN

---

[43] A. Graves and J. Schmidhuber, "Offline handwriting recognition with multidimensional recurrent neural networks," in *Advances in Neural Information Processing Systems (NIPS) 21*, Cambridge, MA: MIT Press, 2009, pp. 545–552.

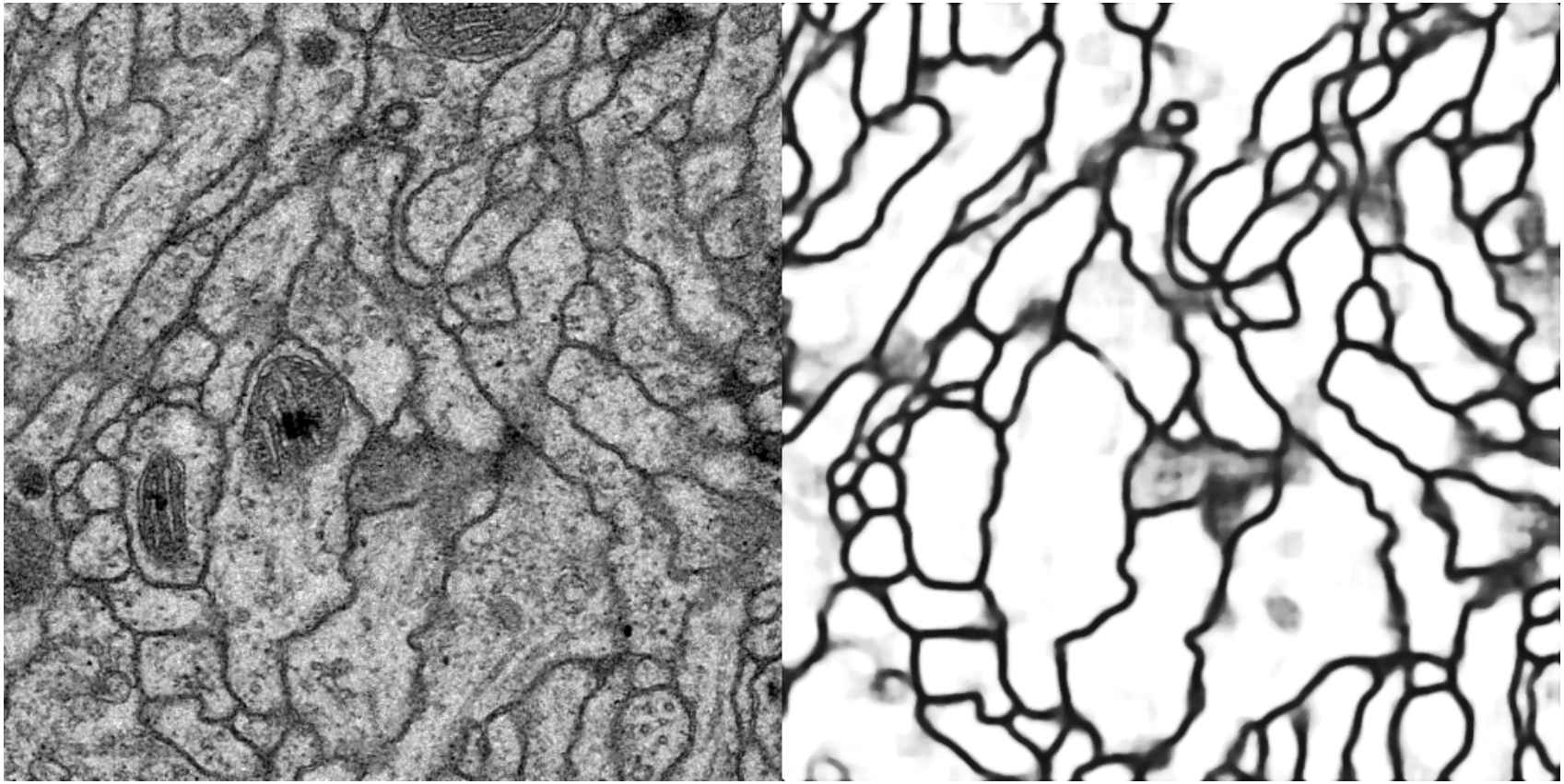
[44] A. Graves, "Supervised sequence labelling with recurrent neural networks," PhD thesis, Technische Universität München, 2008.

Reminder: Fast Scanning MPCNN



- How to segment an image?
- Evaluate trained MPCNN for shifted receptive fields (slow)
- Share already computed convolutions
- Problem of pooling layers

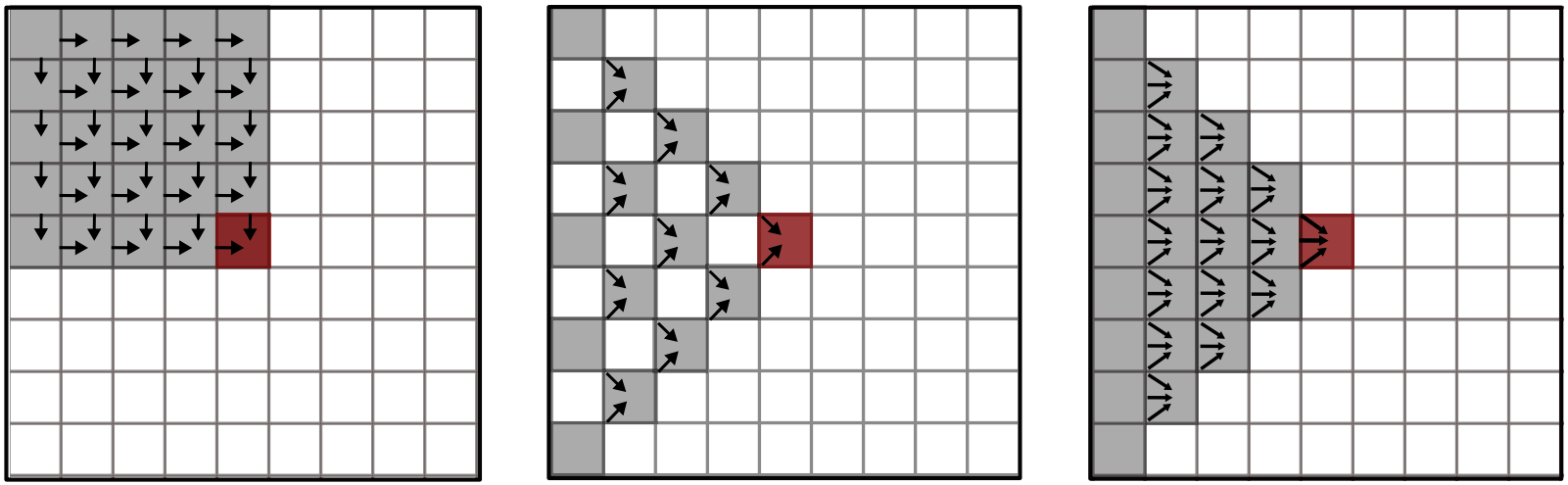
Microscopy Image Segmentation with MPCNN



- ISBI 2012 Segmentation Challenge[45]
- 780× speedup on 512 × 512 images over shifting MPCNN

[45] D. C. Ciresan, A. Giusti, L. M. Gambardella, *et al.*, "Neural networks for segmenting neuronal structures in EM stacks," in *ISBI Segmentation Challenge Competition: Abstracts*, 2012.

PyraMiD LSTM

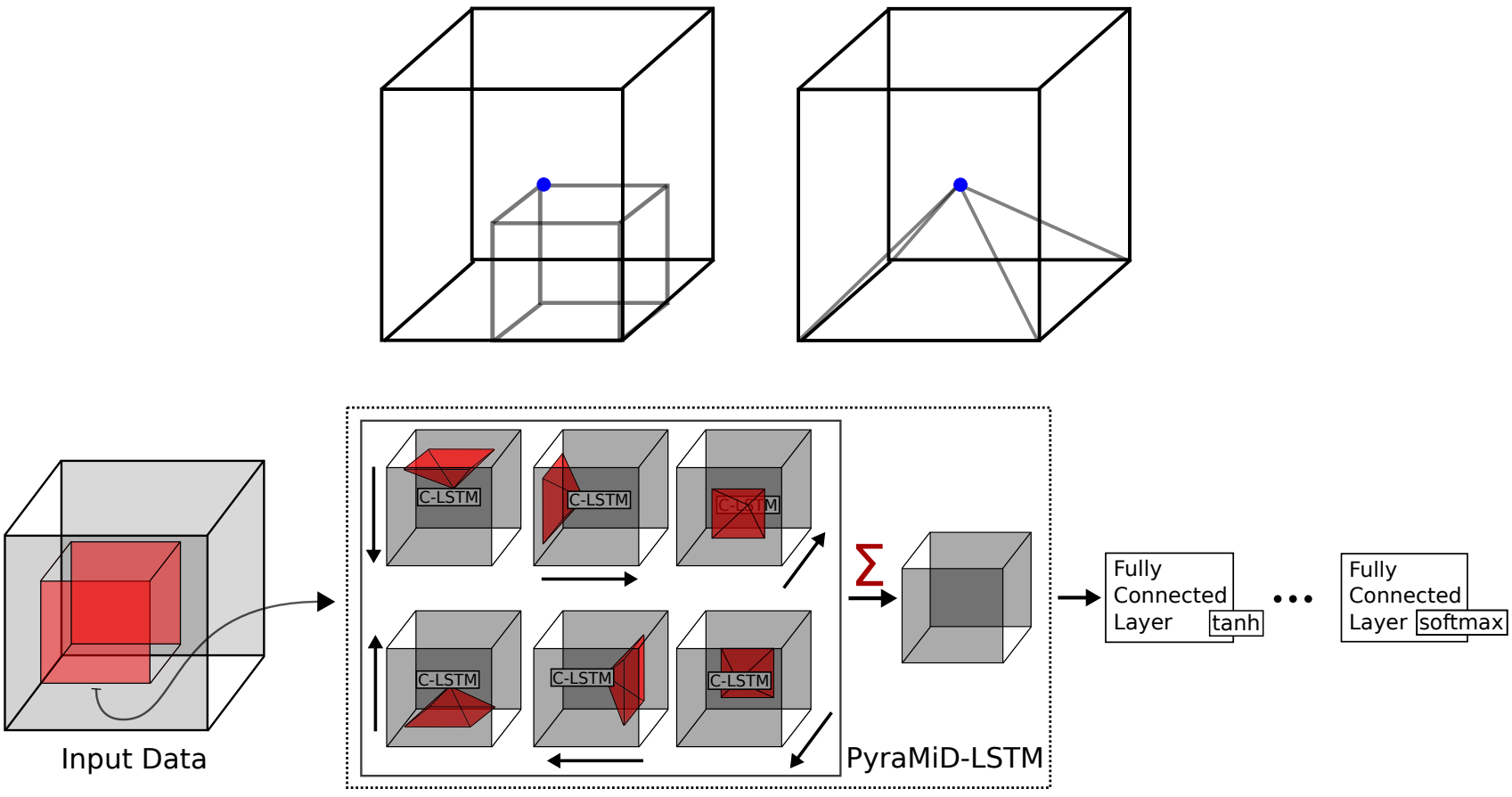


(a) Standard MD-LSTM (b) 'Turned' MD-LSTM (c) PyraMiD LSTM

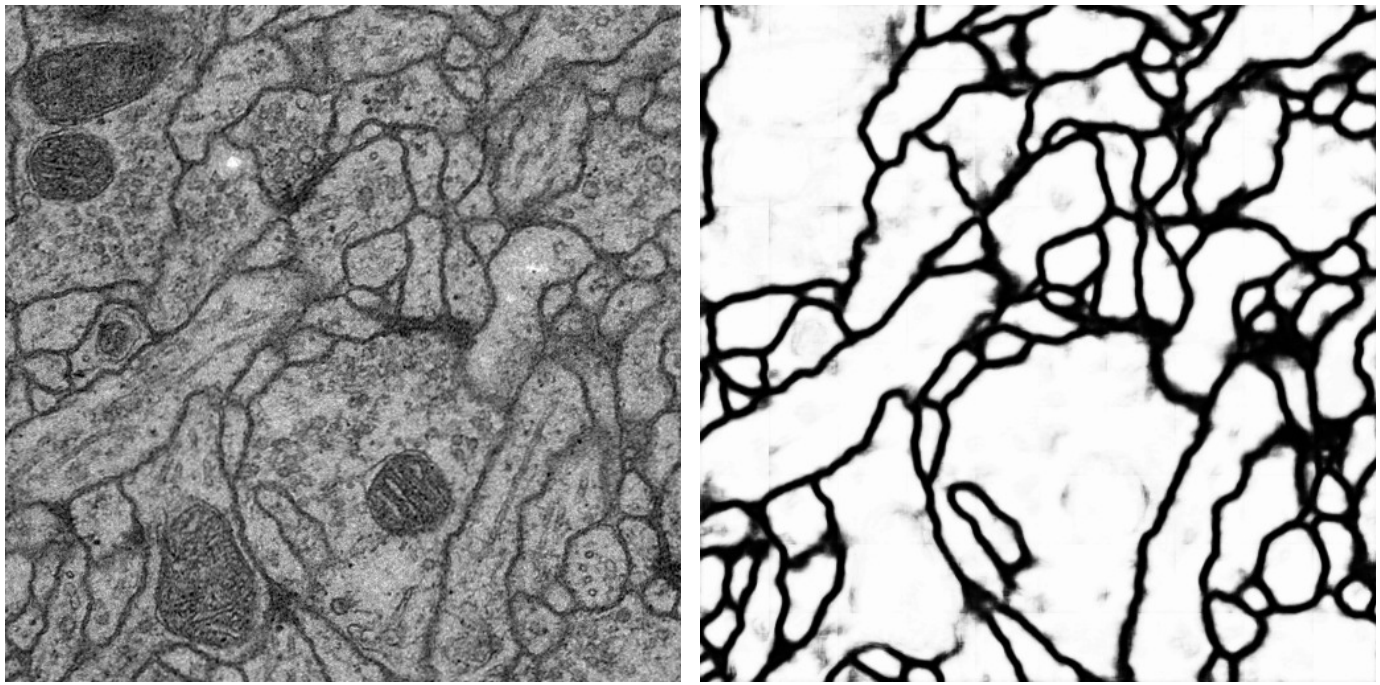
- PyraMiD LSTM[46]
- The context is rotated by 45 degrees
- Fast parallelization on GPUs

[46] M. Stollenga, W. Byeon, M. Liwicki, *et al.*, "Parallel multi-dimensional lstm, with application to fast biomedical volumetric image segmentation," *CoRR*, vol. abs/1506.07452, 2015. [Online]. Available: http://arxiv.org/abs/1506.07452.

PyraMiD LSTM



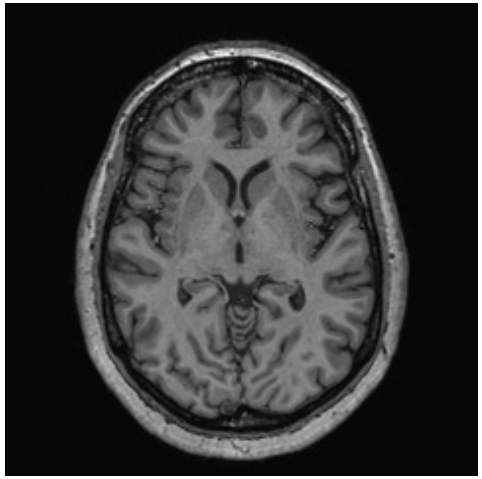
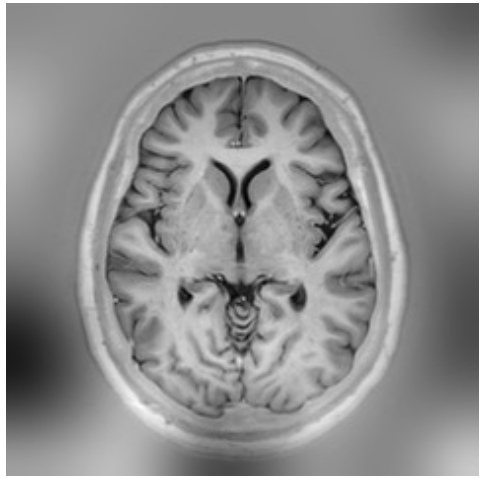
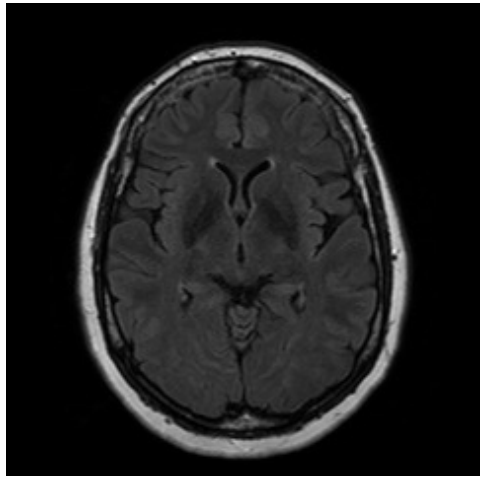
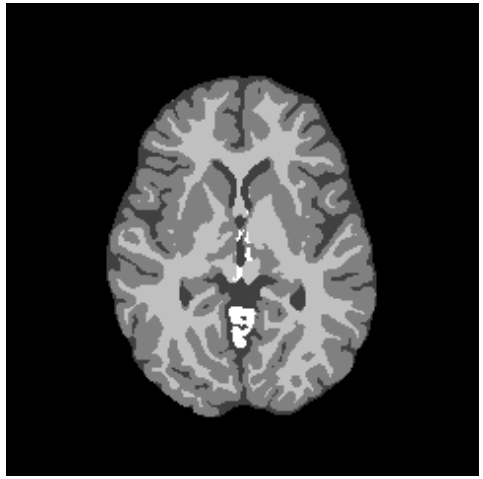
Segmentation with PyraMiD LSTM



- Electron microscopy (EM) dataset[47]

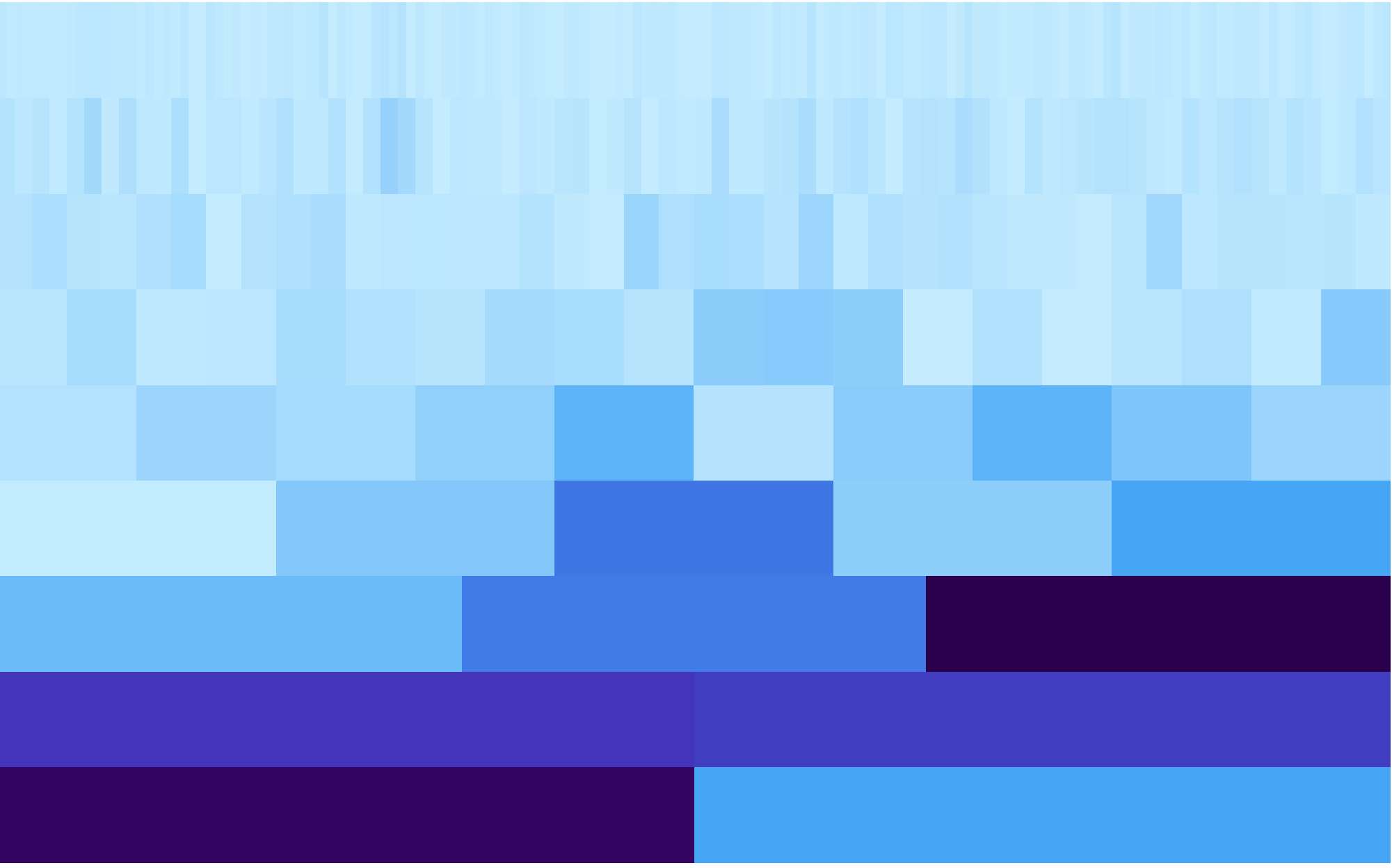
[47] A. Cardona, S. Saalfeld, S. Preibisch, *et al.*, "V.: an integrated micro- and macroarchitectural analysis of the drosophila brain by computer-assisted serial section electron microscopy," Tech. Rep., 2010.

Segmentation with PyraMiD LSTM

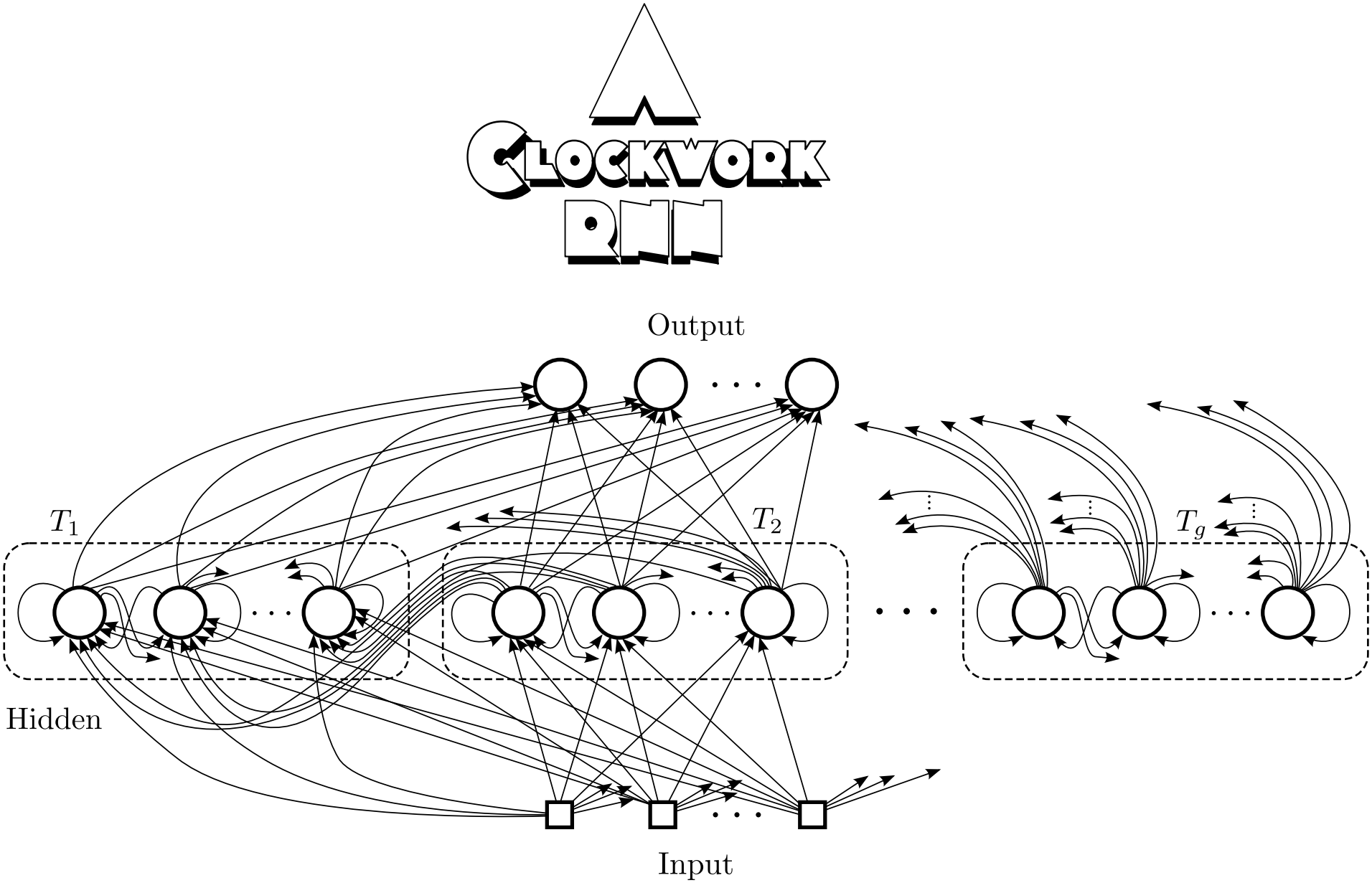





- MR dataset[48]

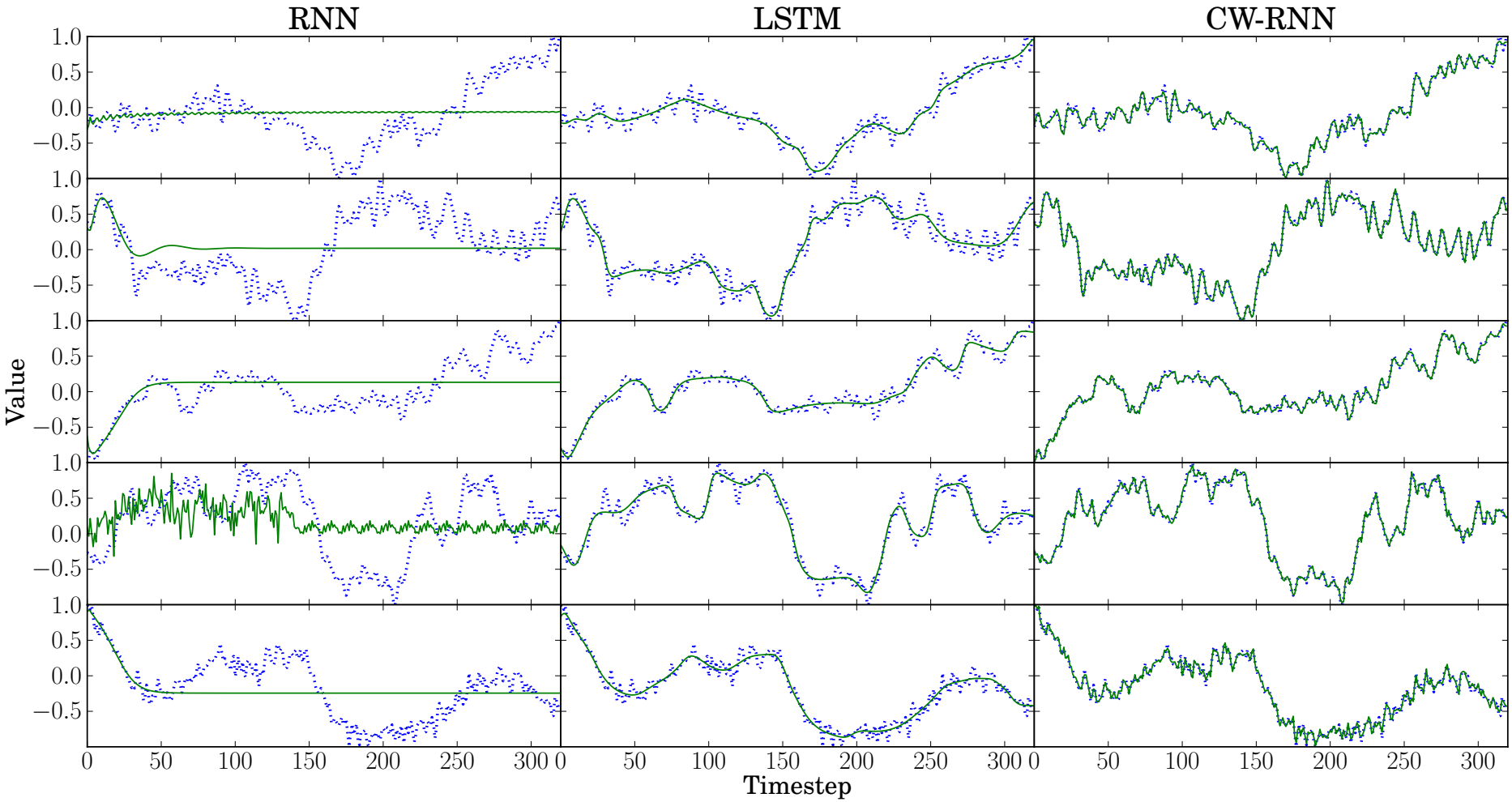
[48] A. Mendrik, K. Vincken, H. Kuijf, *et al.*, "MRBrainS Challenge: Online Evaluation Framework for Brain Image Segmentation in 3T MRI Scans," *Computational Intelligence and Neuroscience, special issue on Simulation and Validation in Brain Image Analysis*, vol. in press, Sep. 2015.



A Clockwork RNN



Sequence Generation with Clockwork RNN



- Better than LSTM in speech keyword recognition[49]
- Precise sequence generation

[49] J. Koutník, K. Greff, F. Gomez, *et al.*, "A Clockwork RNN," , in *Proceedings of the 31th International Conference on Machine Learning (ICML)*, arXiv:1402.3511 [cs.NE], vol. 32, 2014, pp. 1845–1853.

How to start?

- Brainstorm[50]
  - Python interface to c++/CUDA backends
- Theano[51]
  - Automated gradient calculation
  - Keras[52], Lasagne[53]
- TensorFlow[54]
  - Parallel training on clusters

- `http://caffe.berkeleyvision.org`
- `https://velesnet.ml`
- `http://marvin.is`
- `https://github.com/autumnai/leaf`
- `http://www.dmtk.io/document.html`

---

[50] `https://github.com/IDSIA/brainstorm`

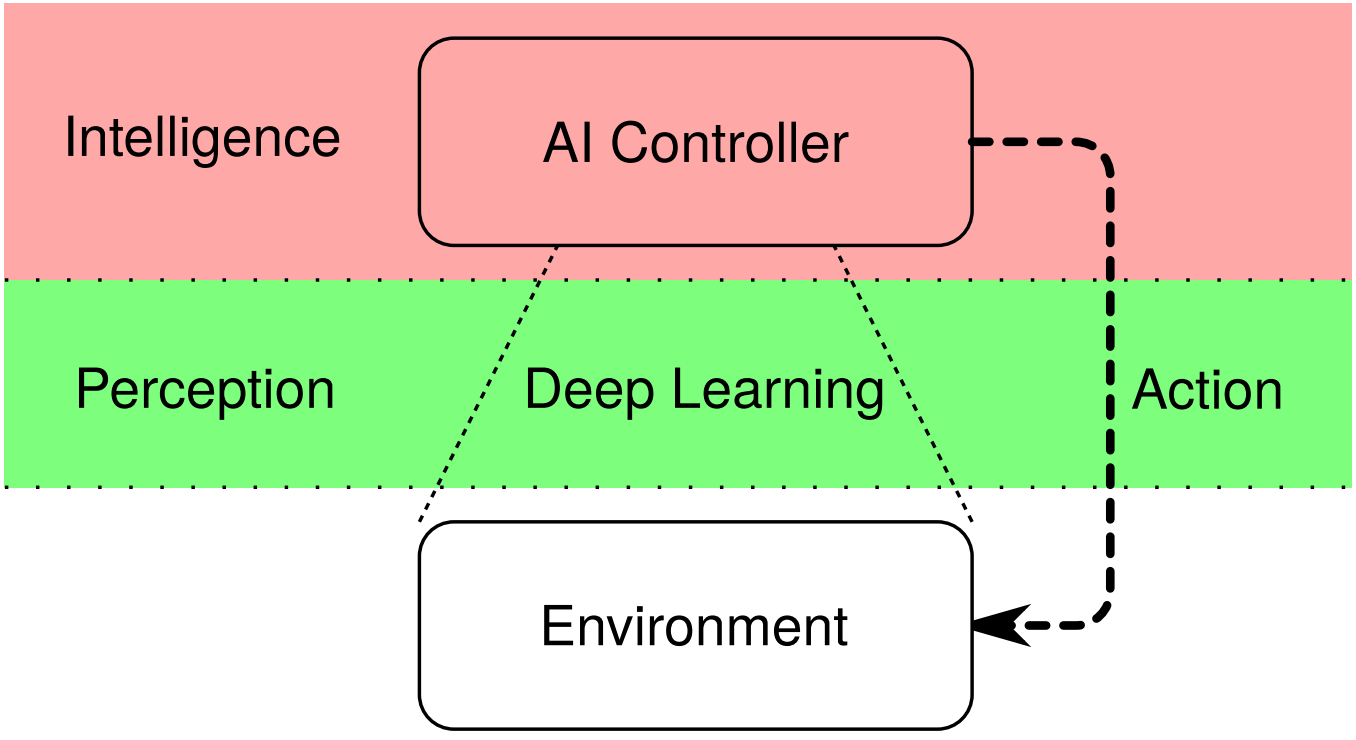
[51] `http://deeplearning.net/software/theano`

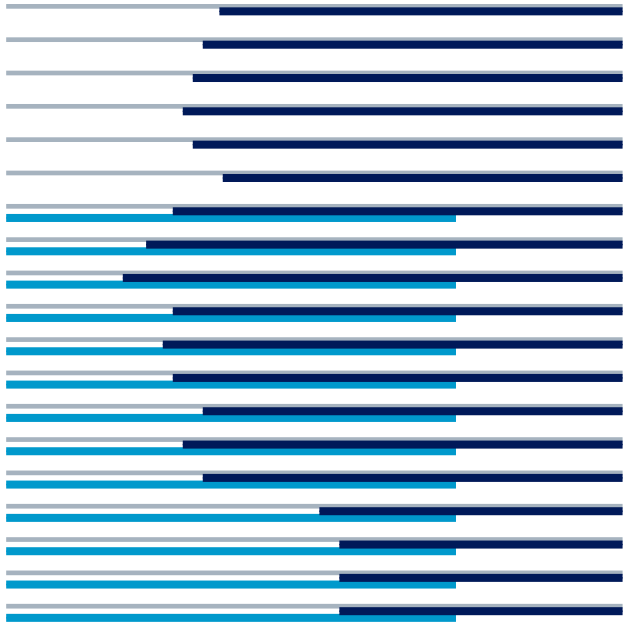
[52] `http://keras.io`

[53] `https://github.com/Lasagne/Lasagne`

[54] `http://www.tensorflow.org/`

The Real Challenge





http://www.idsia.ch