Neural Networks and Deep Learning

Recurrent Networks II

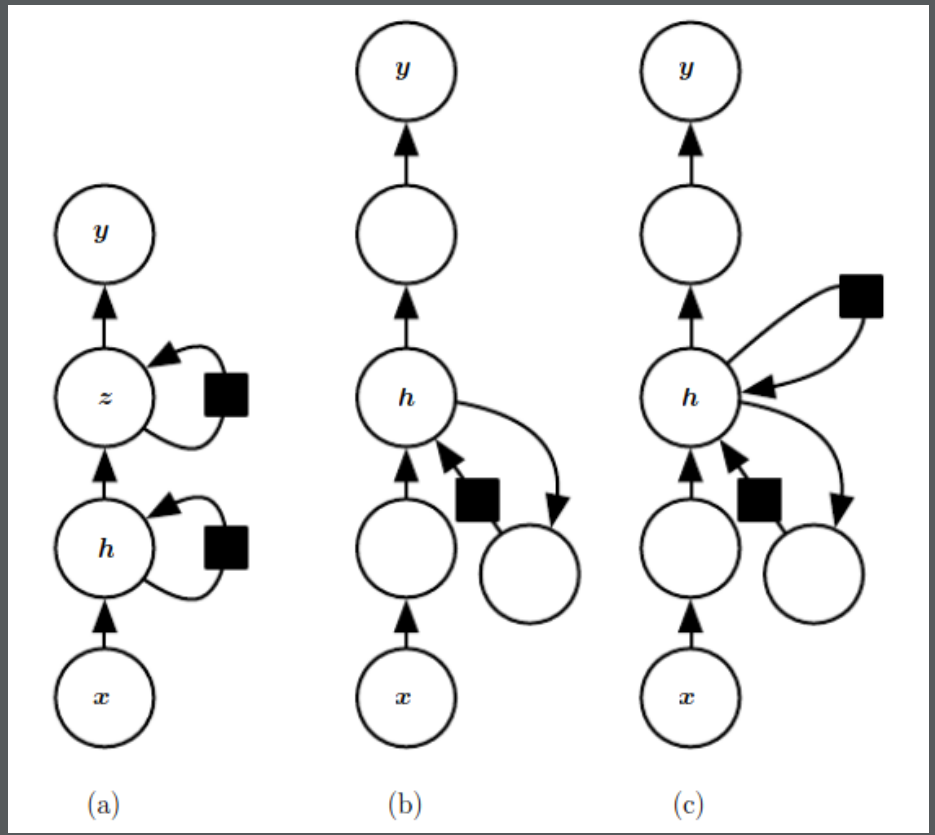
Nicholas Dronen

Department of Computer Science
dronen@colorado.edu

February 20, 2019

University of Colorado **Boulder**

An RNN can be made deep in several ways. (a) The hidden recurrent state can be broken into groups organized hierarchically. (b) Deeper computational modules (e.g., an MLP) can be introduced in a variety of places. (c) Skip connections can be added to ease optimization. Source: http://www.deeplearningbook.org.



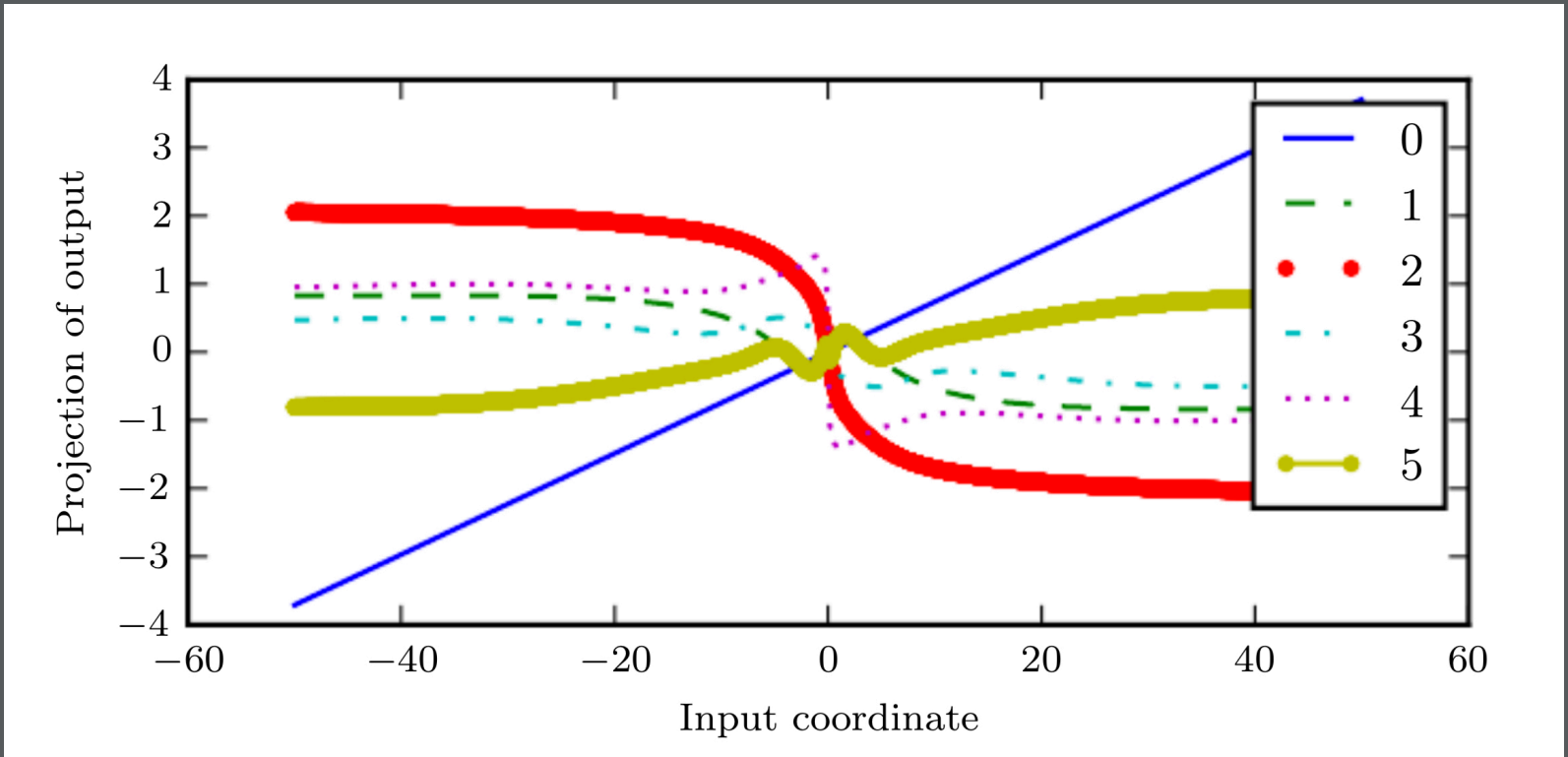
Difficulty of Learning Depends on Temporal Distance of Dependencies

"In practice, the experiments in Bengio et al. (1994) show that as we increase the span of the dependencies that need to be captured, gradient-based optimization becomes increasingly difficult, with the probability of successful training of a traditional RNN via SGD rapidly reaching 0 for sequences of only length x or y."

Difficulty of Learning Depends on Temporal Distance of Dependencies

"In practice, the experiments in Bengio et al. (1994) show that as we increase the span of the dependencies that need to be captured, gradient-based optimization becomes increasingly difficult, with the probability of successful training of a traditional RNN via SGD rapidly reaching 0 for sequences of only length 10 or 20."

Nonlinearities and Instability



Repeatedly applying a non-linear function (here, tanh) results in many small derivatives, some large ones, and many changes in the sign of derivatives.

Axes show a linear projection of 100-dimensional state down to 1 dimension.

Vanishing/Exploding Gradients and Recurrent Networks

Consider the recurrence of an RNN without respect to the input

$$\mathbf{h}^{(t)} = \mathbf{W}^T \mathbf{h}^{(t-1)}$$

This reduces to

$$\mathbf{h}^{(t)} = (\mathbf{W}^{(t)})^T \mathbf{h}^{(0)}$$

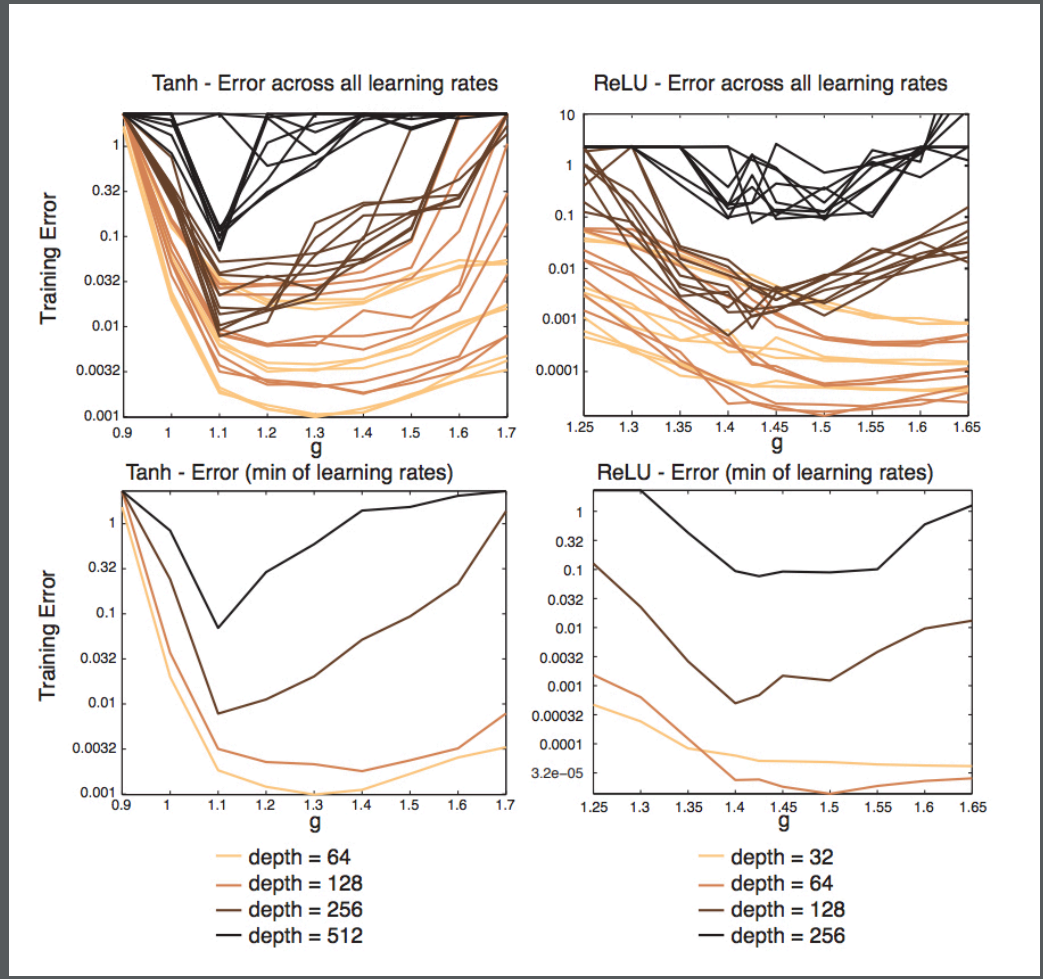
If $\mathbf{W}$ lends itself to eigendecomposition s.t. $\mathbf{W} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T$, with $\mathbf{Q}$ being orthogonal, then $\mathbf{h}^{(t)}$ can be written as

$$\mathbf{h}^{(t)} = \mathbf{Q}^T \mathbf{\Lambda}^t \mathbf{Q} \mathbf{h}^{(0)}$$

What is the implication of $\mathbf{\Lambda}^t$ for the training of RNNs?

Vanishing/Exploding Gradients and Feed-Forward Networks

Assume a scalar weight w. With an initial state of 1, the state at time t is $\prod_t w^{(t)}$. If $w_1, \ldots, w_t$ are i.i.d. with zero mean and unit variance, then judicious sampling of $w_1, \ldots, w_t$ can yield the desired variance for the product neither to vanish nor explode.

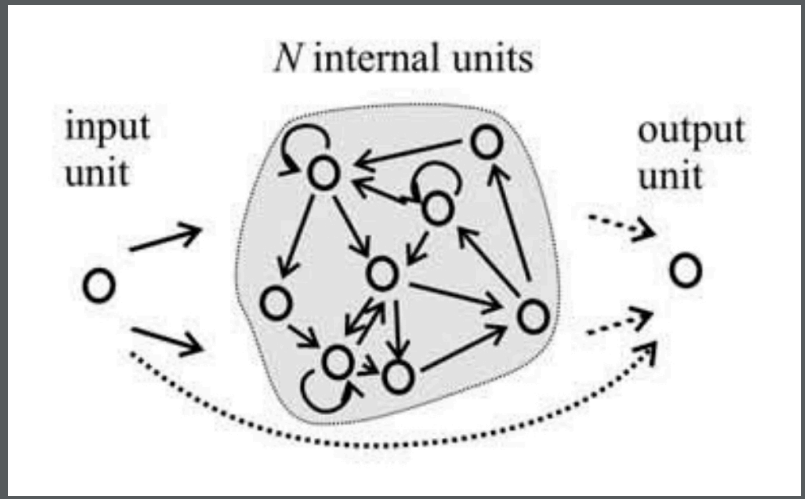


Source: Random Walk Initialization for Training Very Deep Feedforward Networks, Sussillo and Abbott, 2014

One way to overcome the difficulty of training the recurrent layer of an RNN is to … punt.

- Echo State Networks (ESN)
- Liquid State Machines (like ESNs, with spiking activations)

Set the recurrent weights $\mathbf{W}$ such that the spectral radius of $\mathbf{W}$ is e.g. 3, keep them fixed during training, and train only output layer.



Solid arrows are fixed weights, dashed are trainable. Source: Adaptive Nonlinear System Identification with Echo State Networks, Jaeger, 2003

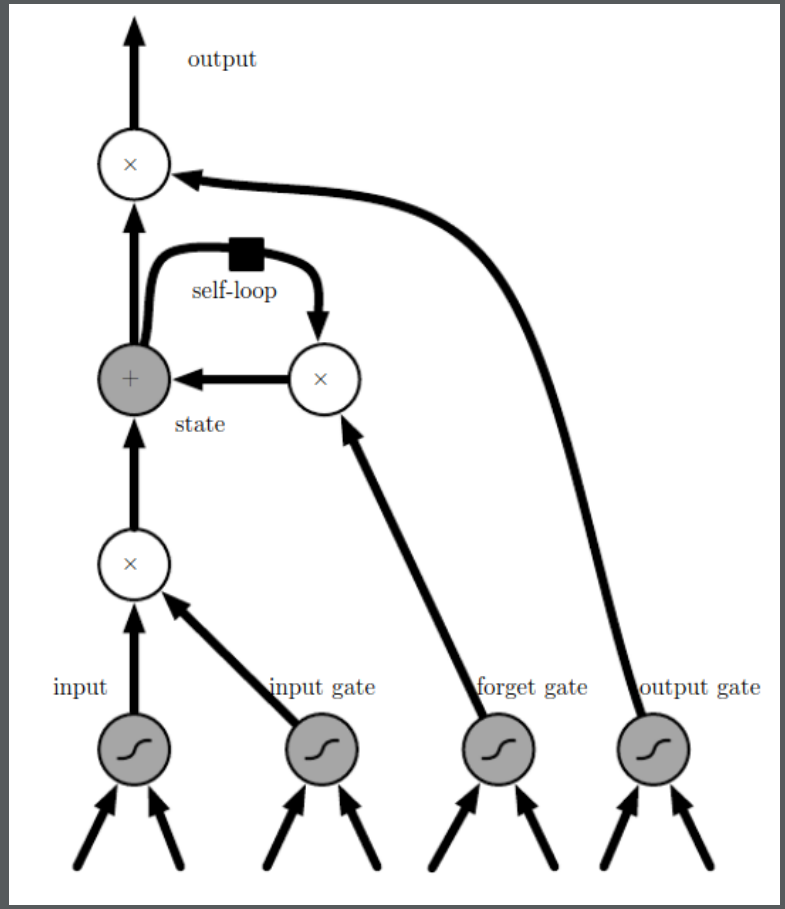
Strategies for Accounting for Multiple Time Scales

- Add skip connections through time with delay d (gradients diminish like τ/d rather than τ)
- Removing connections

Leaky Units

- Obtaining paths on which the product of derivatives is near one can be done by having units with linear self-connections and a weight α near one on these connections.
- In a moving running average μ_t of some value v_t (e.g. via $\mu_t \leftarrow \alpha\mu_{t-1} + (1-\alpha)v_t$), the α parameter acts as a linear self-connection from μ_{t-1} to μ_t, because it preserves the past.
- Leaky units – hidden units with linear self-connections – can behave similarly to moving averages.

Long Short-Term Memory (LSTM)

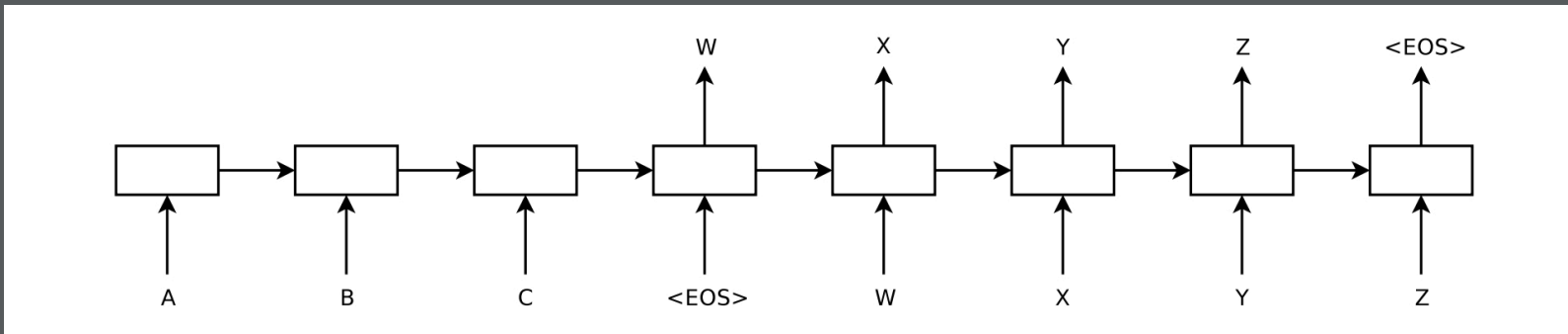


Block diagram of the LSTM recurrent network "cell." Cells are connected recurrently to each other, replacing the usual hidden units of ordinary recurrent networks

Long Short-Term Memory (LSTM)

Sutskever et al, 2014 trained a encoder-decoder LSTM to perform machine translation.

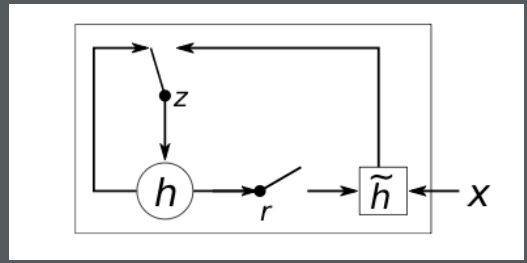
- Separate encoder and decoder allows for simultaneous training of multiple language pairs
- Encoder and decoder each had four layers (four layers of recurrences via LSTM cells)
- Sentences were input to the encoder in reverse - "greatly boost(ed) the performance of the LSTM"



The encoder-decoder LSTM reads an input sentence "ABC" and produces "WXYZ" as the output sentence. Source: Sutskever et al, 2014

Gated Recurrent Units (GRU)

- The update gate z selects whether the hidden state is to be updated with a new hidden state $\tilde{h}$.
- The reset gate r decides whether the previous hidden state is ignored.



Gated Recurrent Unit (GRU).
Source: Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. Cho et al

$$h_i^{(t)} = z_i^{(t)} h_i^{(t)} + (1 - z_i^{(t)}) \sigma(b_i + \sum_j U_{i,j} x_j^{(t)} + \sum_j W_{i,j} r_j^{(t)} h_j^{(t-1)})$$

where the update gate (z) and reset gate (r) are defined as usual

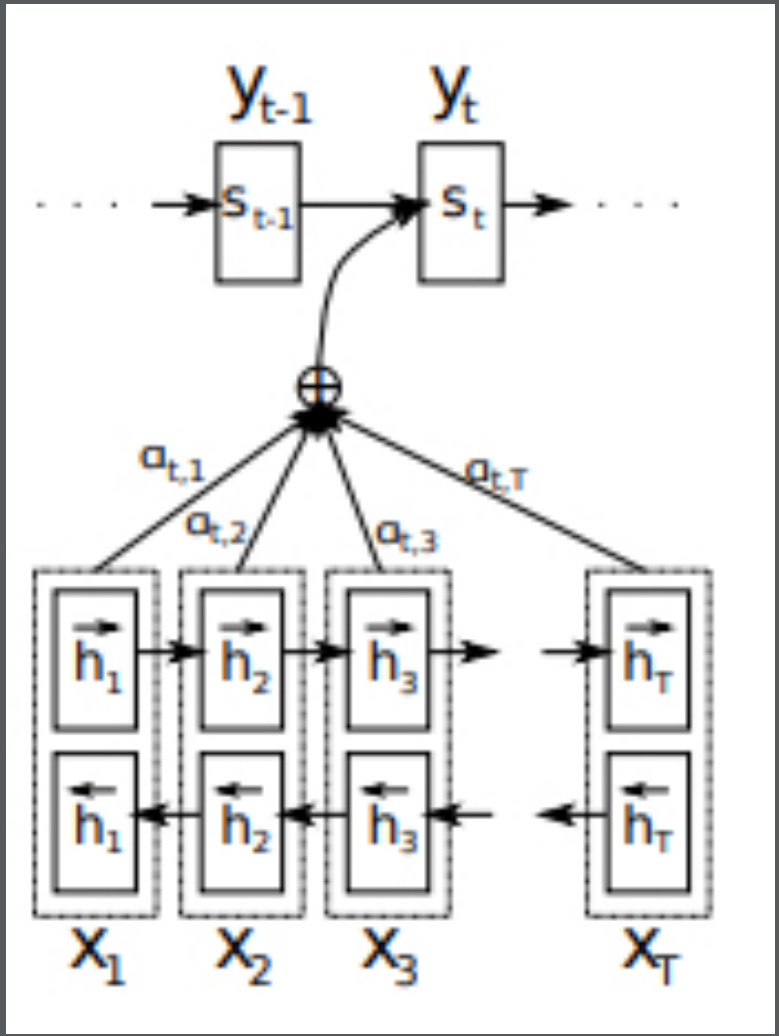
$$z_i^{(t)} = \sigma(b_i^z + \sum_j U_{i,j}^z x_j^{(t)} + \sum_j W_{i,j}^z h_j^{(t-1)})$$

$$r_i^{(t)} = \sigma(b_i^r + \sum_j U_{i,j}^r x_j^{(t)} + \sum_j W_{i,j}^r h_j^{(t-1)})$$

Attention with a Bidirectional RNN

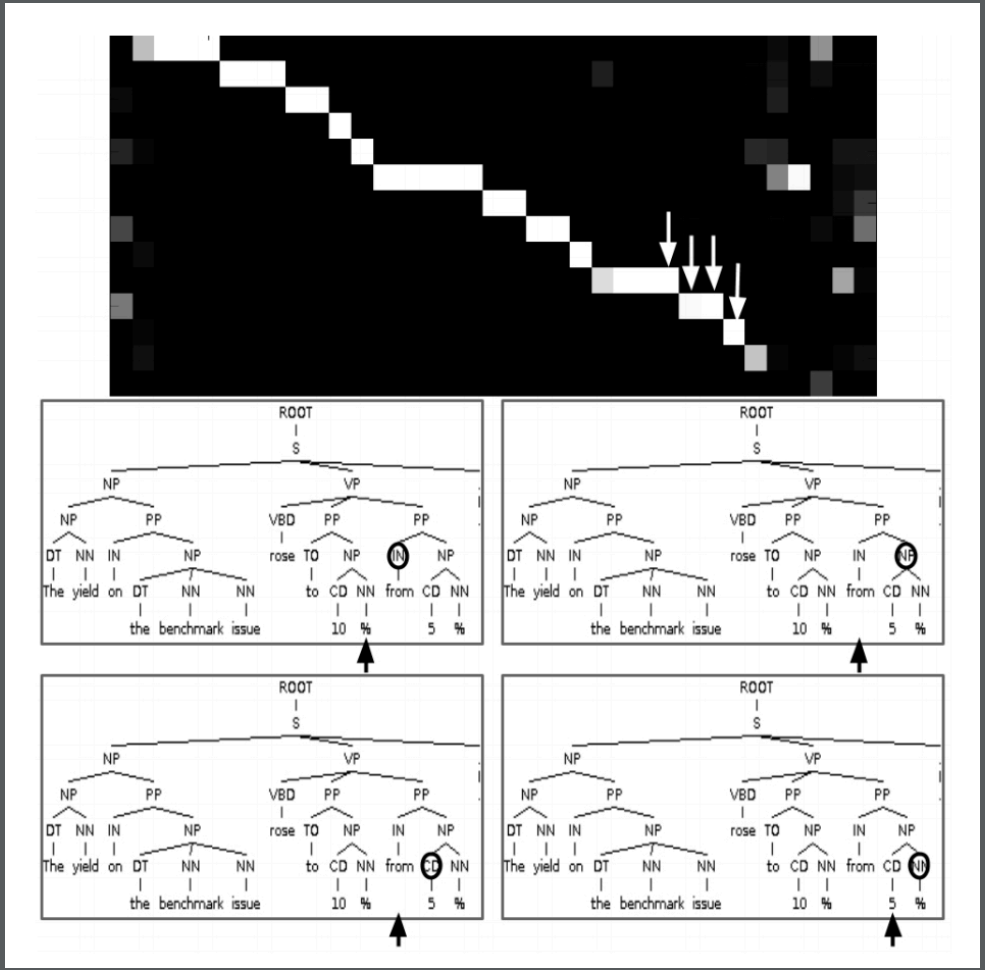
Bahdanau et al, 2014 apply a feed-forward network and softmax to the encoder hidden state of each time step of the input ($\oplus$ the previous decoder hidden state).

This computes a separate "soft attention" weighting $\alpha_{t,j}$ for each encoder hidden state at t, which increases the quality of the context vector during each step of the decoder.



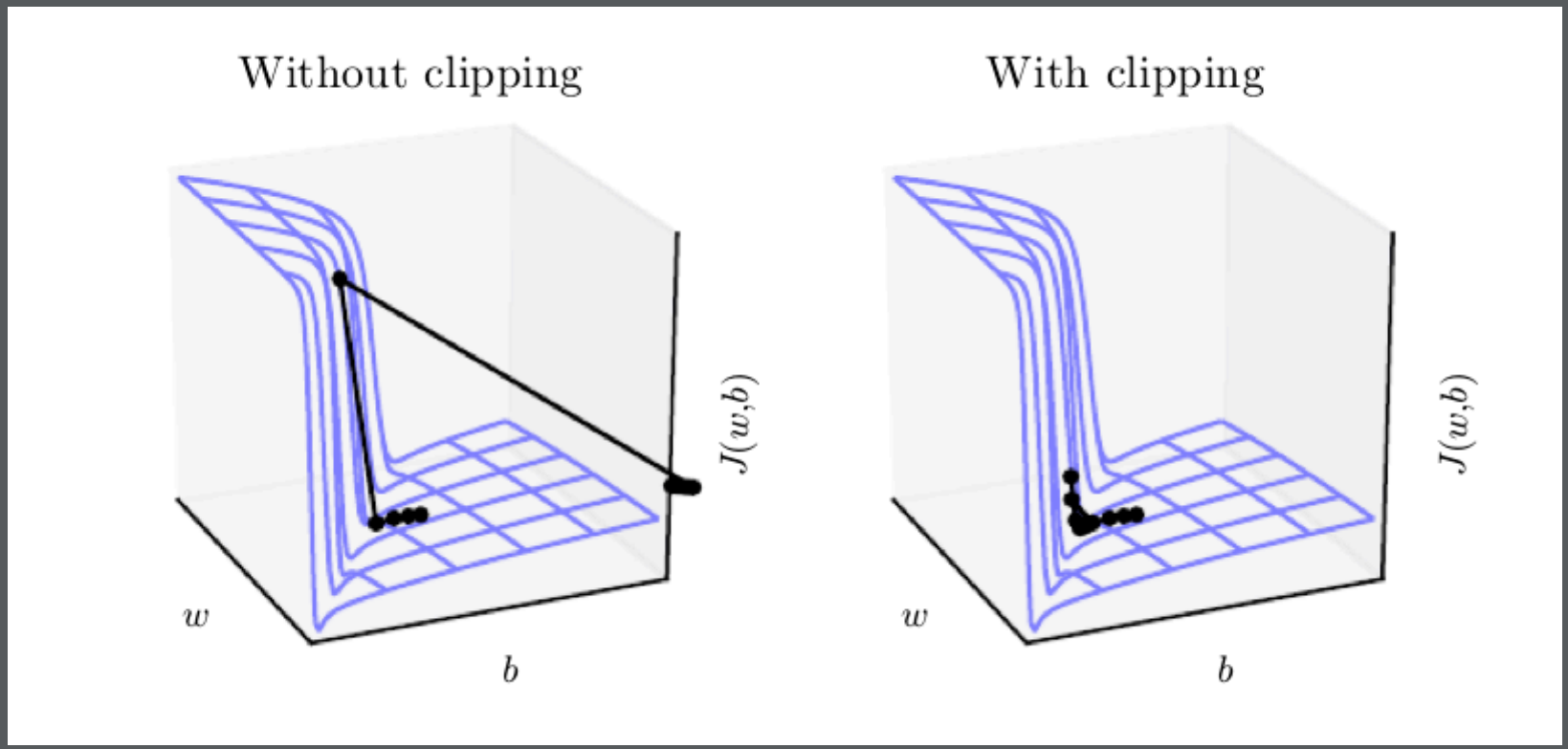
Attention with an LSTM

Combines LSTM encoder-decoder architecture of Sutskever et al, 2014 and soft attention mechanism of Bahdanau et al, 2014 to learn to predict the output of a constituency parser.



Source: Grammar as a Foreign Language, Vinyals et al, 2014

Gradient Clipping



$$if\ ||g|| > v\ \textbf{do}\ g \leftarrow \frac{gv}{||g||}$$

Source: www.deeplearningbook.org

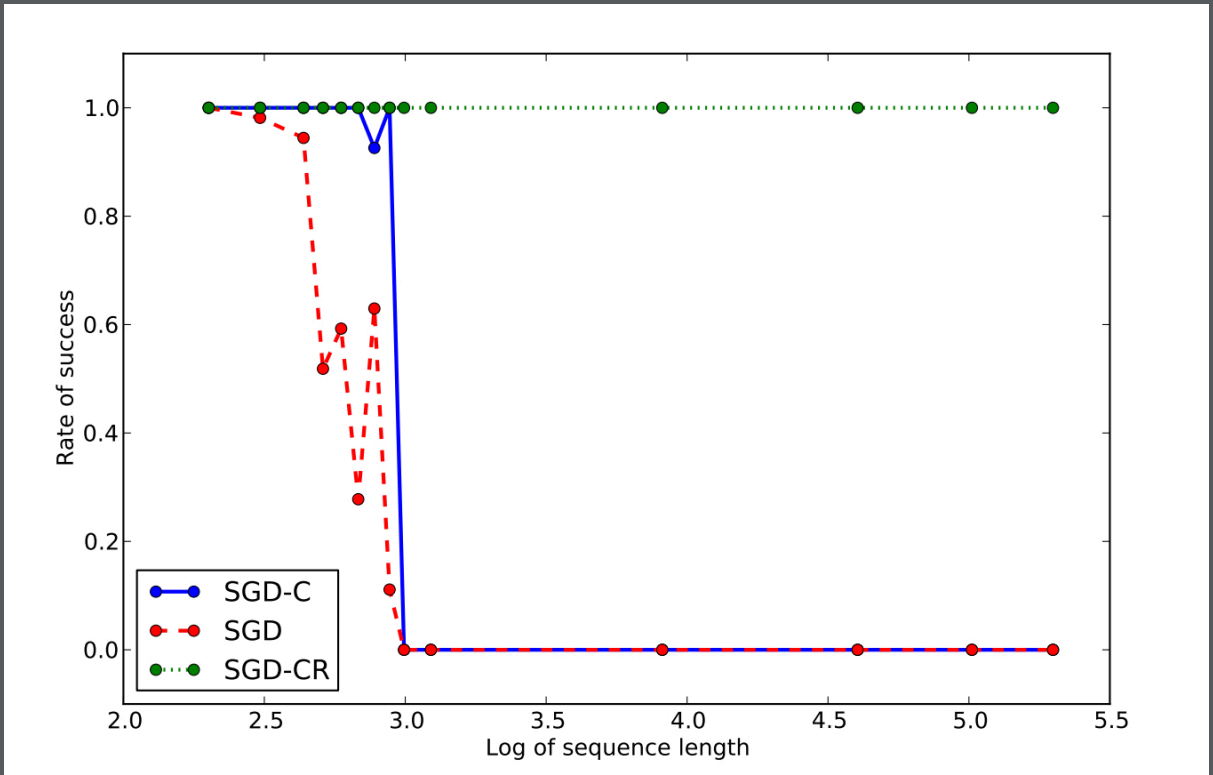
Regularizing to Encourage Information Flow

Pascanu et al (2013) proposed a regularization term to encourage the flow of gradients during BPTT.

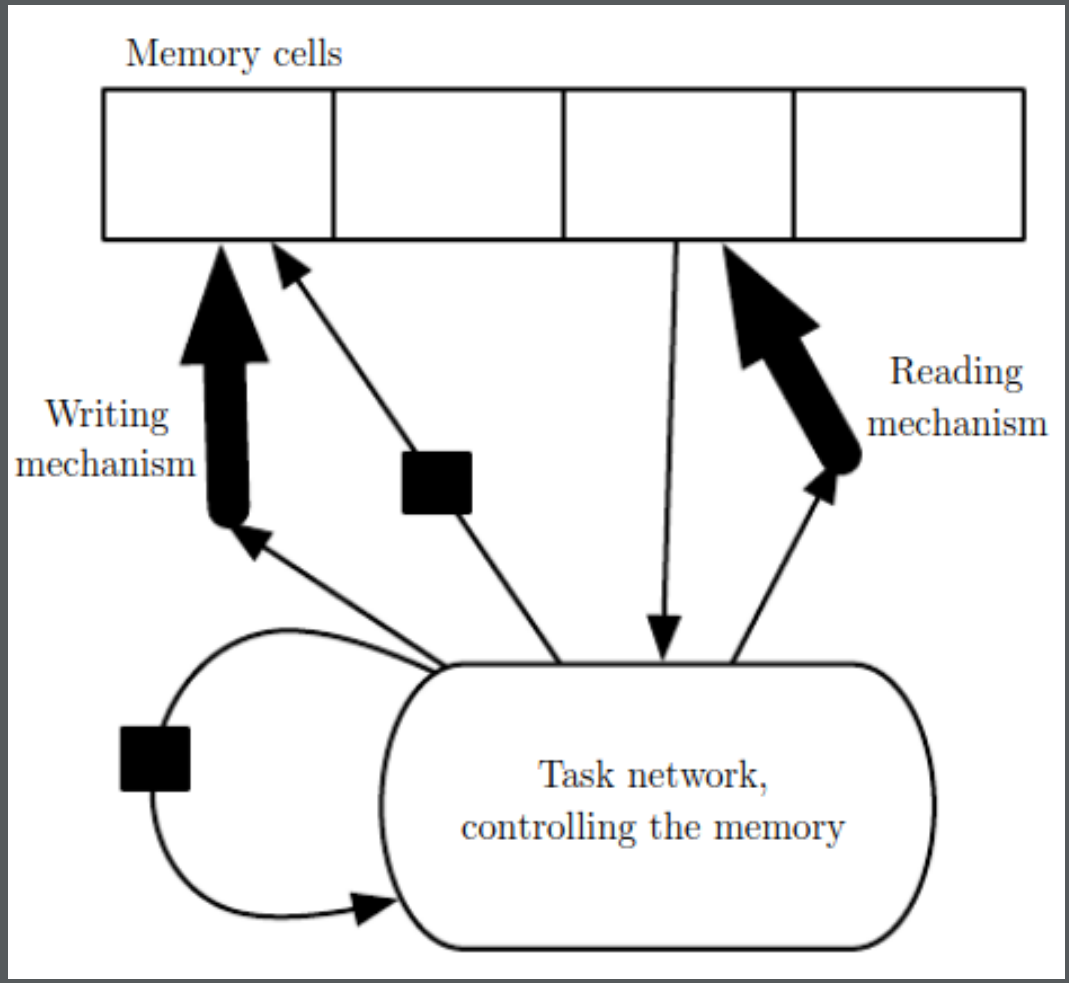
$$\Omega = \sum_t \left(\frac{||(\nabla_{\mathbf{h}^{(t)}} L) \frac{\partial \mathbf{h}^{(t)}}{\partial \mathbf{h}^{(t-1)}}||}{||\nabla_{\mathbf{h}^{(t)}} L||} - 1 \right)^2$$

What does this do?

Clipping Gradients and Encouraging Information Flow



Rate of success for solving the temporal order problem versus log of sequence length. *SGD*: Vanilla SGD, *SGD-C*: SGD with gradient clipping, *SGD-CR*: SGD with clipping and information flow regularization Source: On the Difficulty of Training Recurrent Neural Networks, Pascanu et al, 2013


Memory cells
Writing mechanism
Reading mechanism
Task network, controlling the memory

Memory Networks

Weston et al 2014 introduced Memory Networks.

Method	F1
(Fader et al., 2013)	0.54
(Bordes et al., 2014b)	0.73
MemNN (embedding only)	0.72
MemNN (with BoW features)	0.82

Memory Network versus previous models on question-answering task

	Difficulty 1		Difficulty 5	
Method	actor w/o before	actor w/o before + object	actor w/o before	actor w/o before + object
RNN	100%	58%	29%	17%
MemNN $k = 1$	90%	9%	46%	21%
MemNN $k = 1$ (+time)	100%	73%	100%	73%
MemNN $k = 2$ (+time)	100%	99.95%	100%	99.4%

Memory Network versus an RNN on synthetic sentence dataset (precursor to bAbI?)

bAbI Tasks Dataset

Weston et al, 2016 introduced a dataset of 20 tasks designed to evaluate the ability of an agent to understand and reason with natural language. [download]

Task 1: Single Supporting Fact
Mary went to the bathroom.
John moved to the hallway.
Mary travelled to the office.
Where is Mary? A:office

Task 2: Two Supporting Facts
John is in the playground.
John picked up the football.
Bob went to the kitchen.
Where is the football? A:playground

Task 3: Three Supporting Facts
John picked up the apple.
John went to the office.
John went to the kitchen.
John dropped the apple.
Where was the apple before the kitchen? A:office

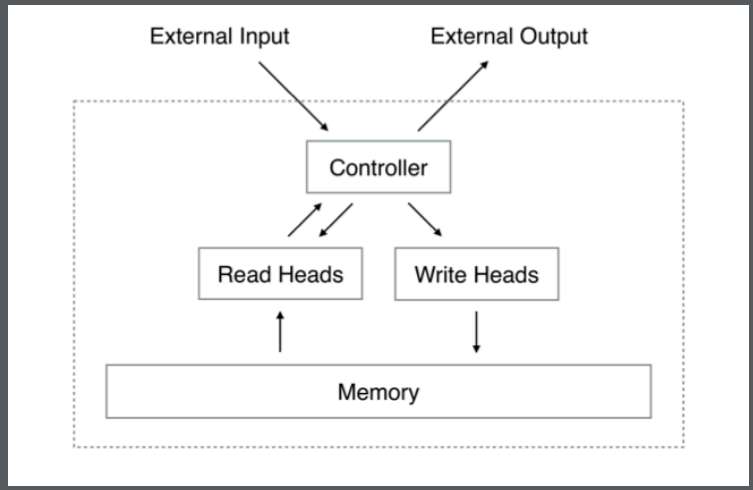
Task 4: Two Argument Relations
The office is north of the bedroom.
The bedroom is north of the bathroom.
The kitchen is west of the garden.
What is north of the bedroom? A: office
What is the bedroom north of? A: bathroom

Examples of bAbI tasks

Neural Turing Machines

During each update cycle, the controller network receives inputs from an external environment and emits outputs in response. It also reads to and writes from a memory matrix via a set of parallel read and write heads. The dashed line in the figure below indicates the division between the NTM circuit and the outside world.

NTMs have been shown to learn simple algorithms such as sorting, copying and associative recall based on input and output examples.



Neural Turing Machine

Source: Neural Turing Machines. Graves et al

Differentiable Neural Computer (DNC)

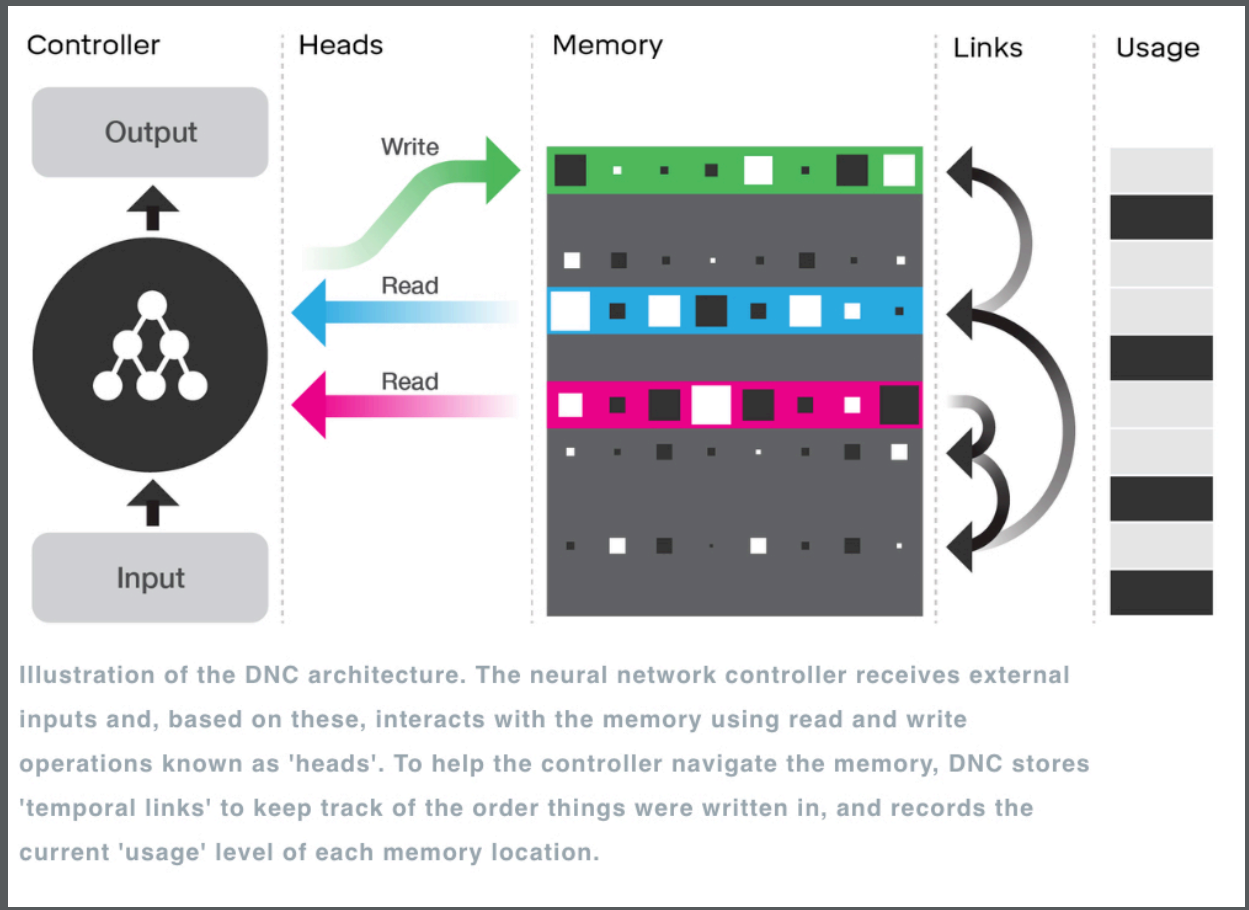


Illustration of the DNC architecture. The neural network controller receives external inputs and, based on these, interacts with the memory using read and write operations known as 'heads'. To help the controller navigate the memory, DNC stores 'temporal links' to keep track of the order things were written in, and records the current 'usage' level of each memory location.

DNC memory supports selective writes, unlike Memory Networks and Pointer Networks. Source: https://deepmind.com/blog/differentiable-neural-computers/