Institut
des algorithmes
d'apprentissage
de Montréal

Mila

Recurrent Neural Networks

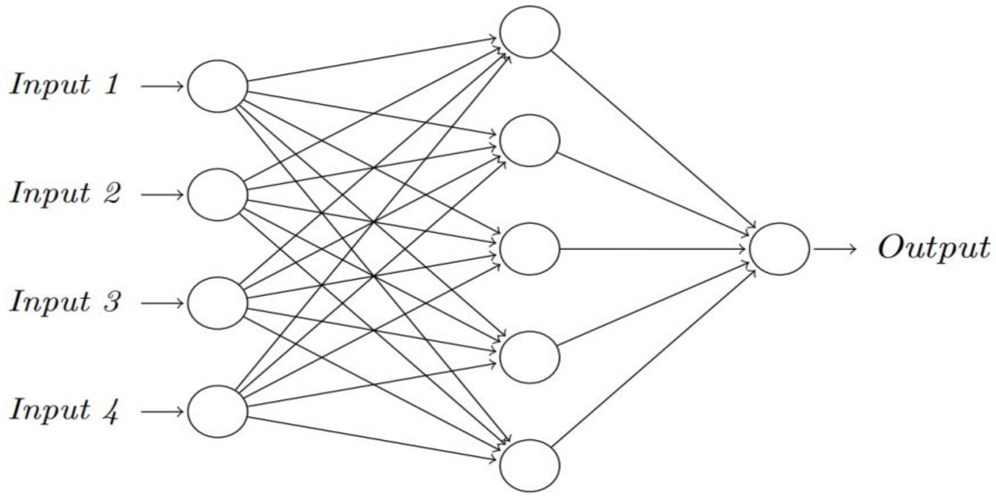
Nan Rosemary Ke

Plan

- Motivation
- Introduction to recurrent neural networks (RNNs)
- Training RNNs
- Difficulties for training
- RNN architectures
- Teacher forcing
- Different variations of RNNs

Motivation

- Seen how to train model with fixed size data

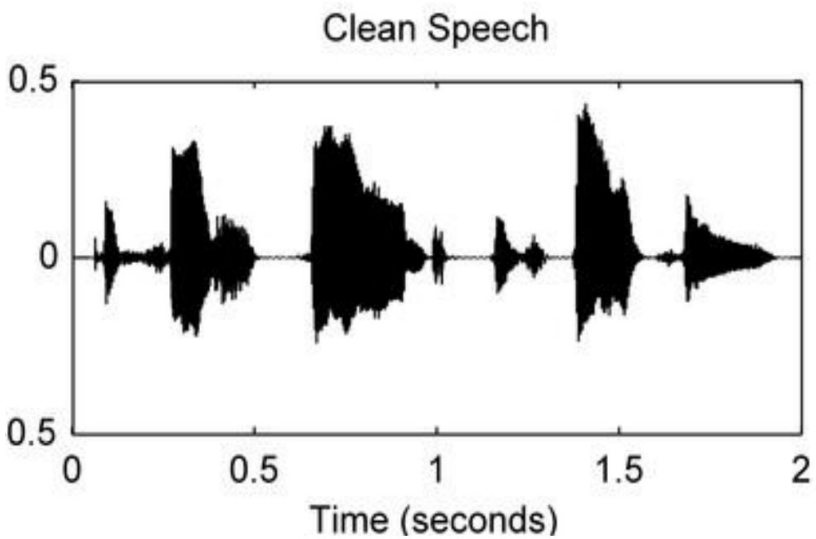


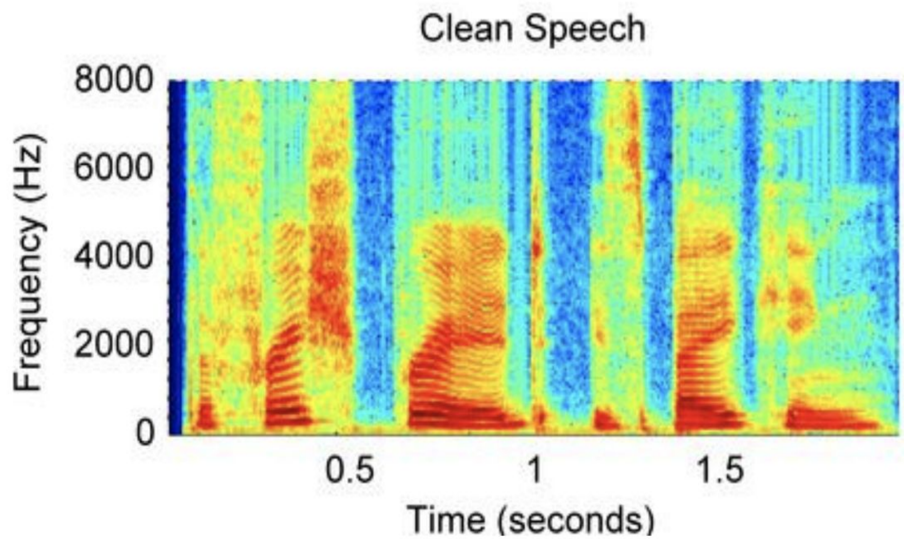
- What about sequential data?

Images from Cedar Laurent's slides

Sequential data

- Speech recognition
  - Audio to text

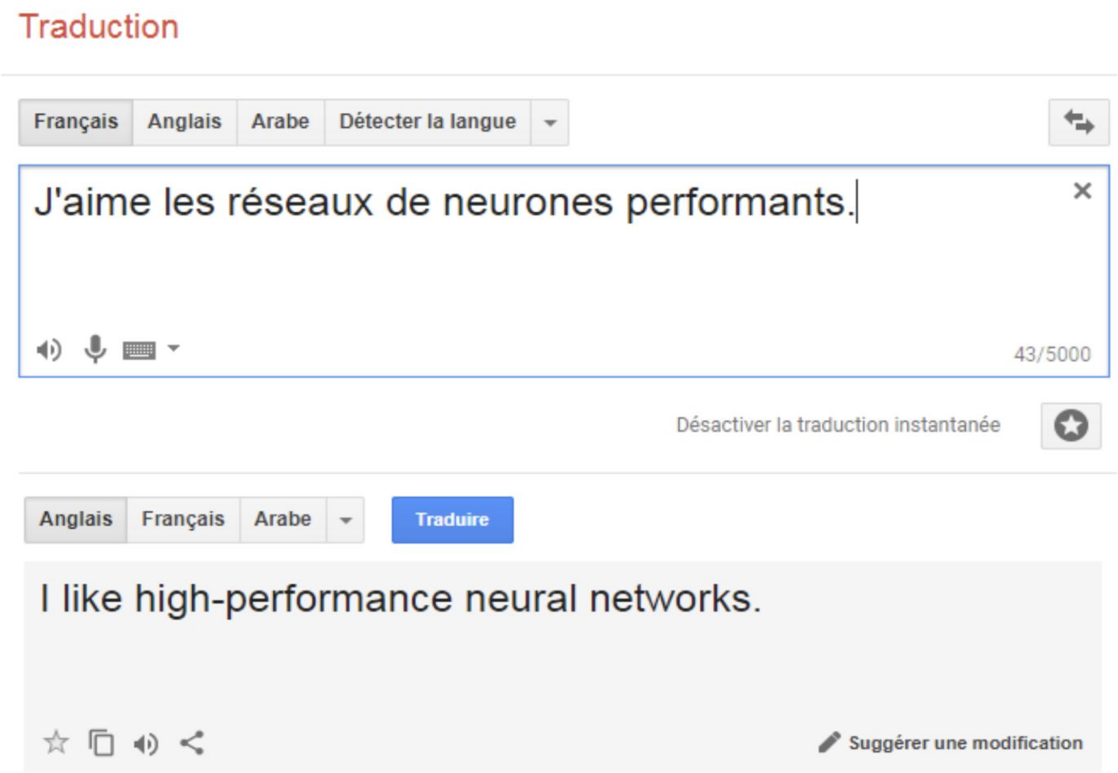




Images from Cedar Laurent's slides

Sequential data

- Machine translation
  - Text to text



Images from Cedar Laurent's slides

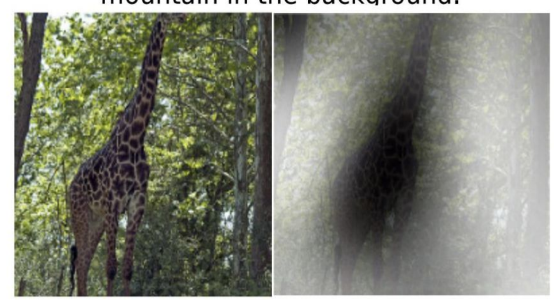
Sequential data

- Image captioning
  - Image to text

A woman is throwing a frisbee in a park.

A stop sign is on a road with a mountain in the background.

A group of people sitting on a boat in the water.



A giraffe standing in a forest with trees in the background.

Images from Cedar Laurent's slides

Sequential data

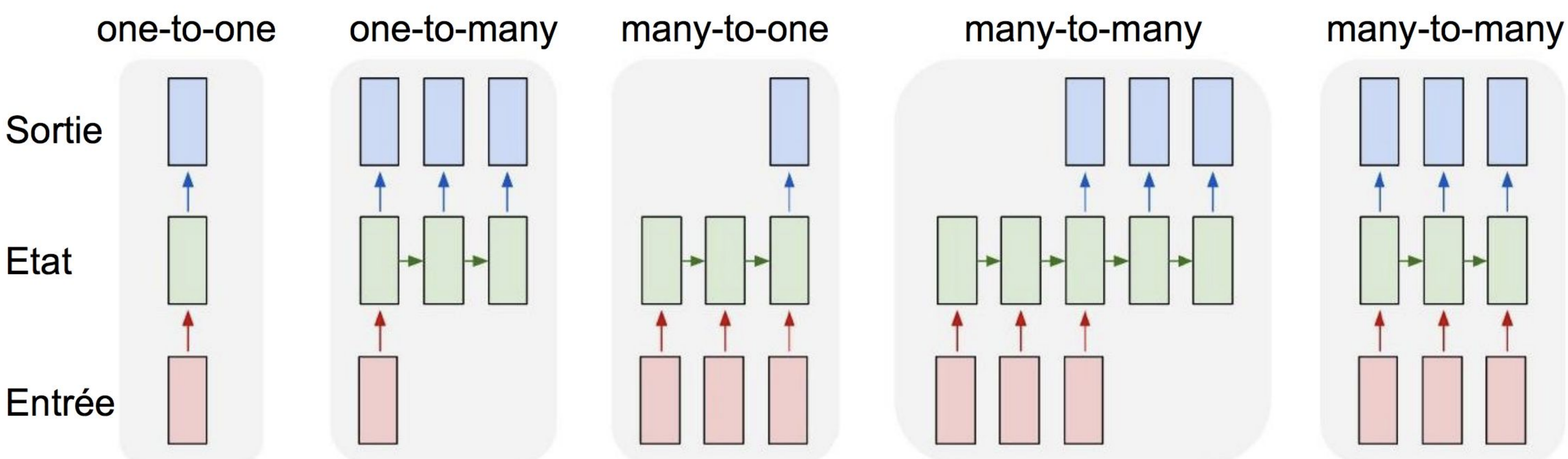
- More examples
  - Text (language modeling)
  - Video (video generation, video understanding)
  - Biological data
    - Medical imaging
    - DNA sequences

Recurrent Neural Networks

- For handling sequential data
  - Variable length input
  - Variable order
    - “I visited Paris in 2014”
    - “In 2014, I visited Paris”
- Use shared parameters across time

Sequence modeling

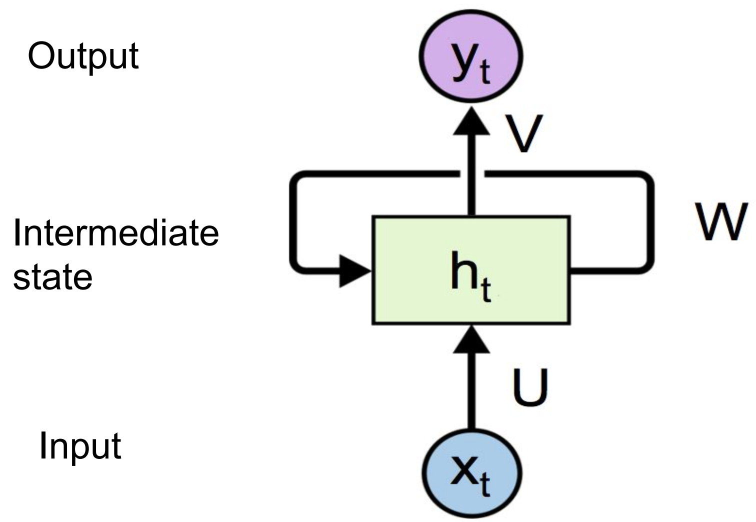
- Different applications



Object classification | Image captioning | Sentiment analysis | Machine translation | Speech recognition

Introduction to recurrent neural networks (RNNs)

- Input: $x_0, x_1, \ldots, x_T$
- Output: $y_0, y_1, \ldots, y_{T,}$
- Intermediate state: $h_0, h_1, \ldots, h_{T,}$



Images from Chris Olah's slides

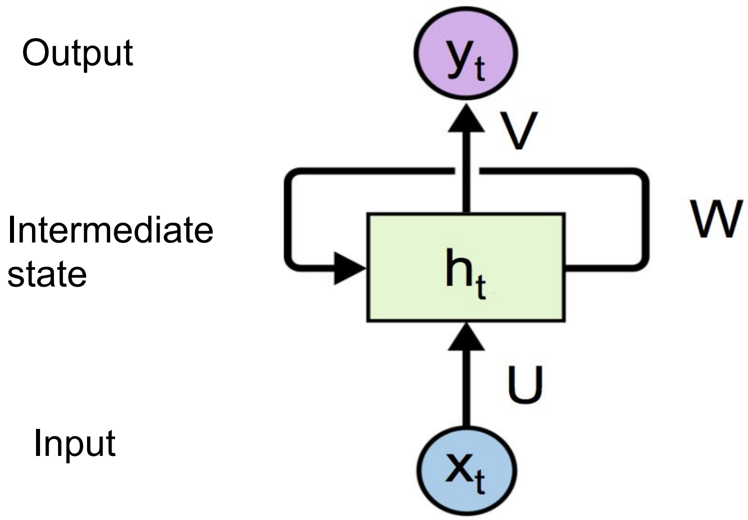
Recurrent neural networks

Vanilla recurrent neural networks

$$h_t = \tanh(Ux_t + Wh_{t-1})$$
$$y_t = f(Vh_t)$$

Parameters of the network

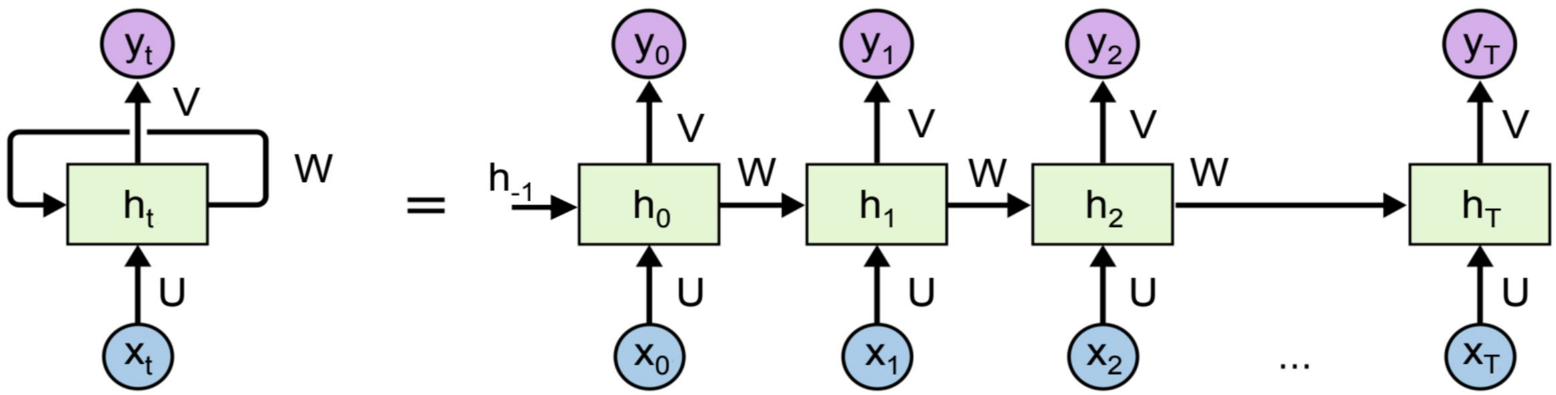
- ➢ U, W, V
- ➢ Shared across time steps



Images from Chris Olah's slides

Recurrent neural networks

Parameters unrolled across time



Images from Chris Olah's slides

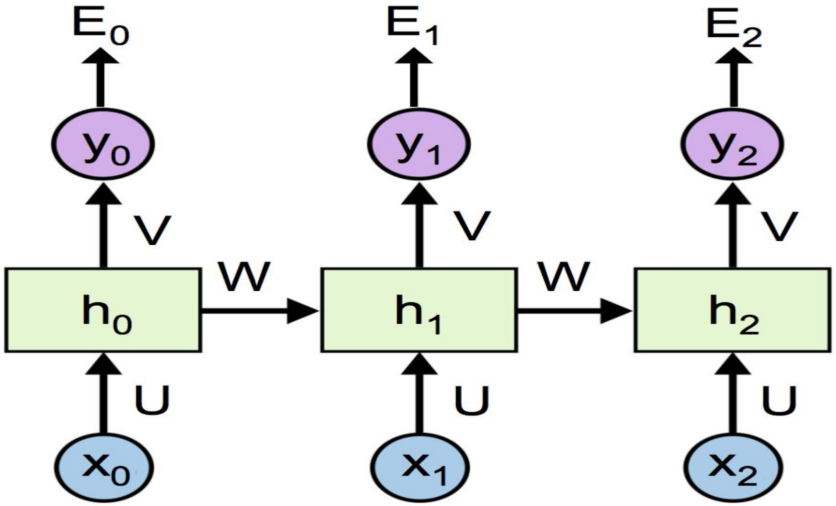
Training RNNs

- All losses

$$E = \sum_{t=0}^{T} E_t$$

- Regular backprop

$$\frac{\partial E}{\partial U} = \sum_{t=0}^{T} \frac{\partial E_t}{\partial U}$$



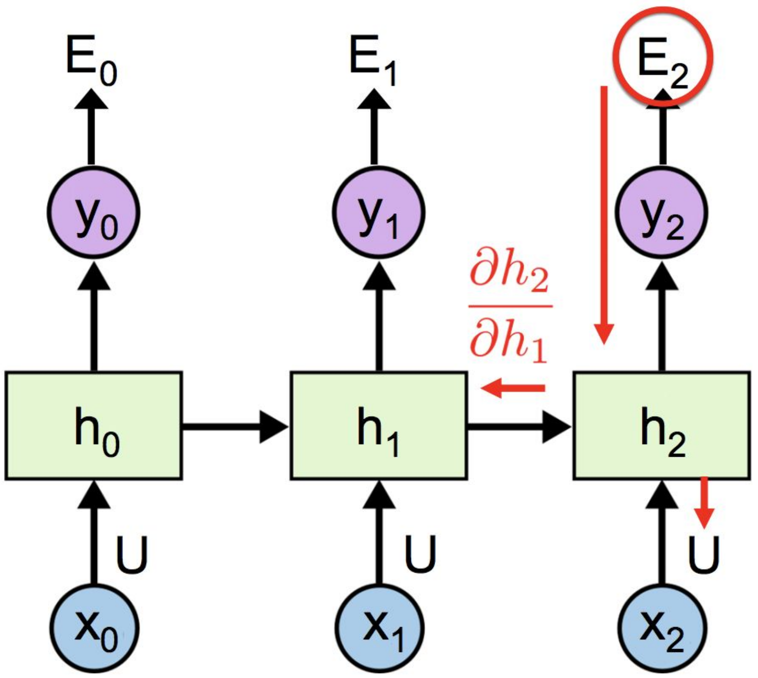
Images from Chris Olah's slides

Mila

Backpropagation through time (BPTT)

$$\frac{\partial E_2}{\partial U} = \frac{\partial E_2}{\partial h_2}\left(x_2^T + \frac{\partial h_2}{\partial h_1}\left(\cdots\right.\right.$$

Image from Christopher Olah's blog

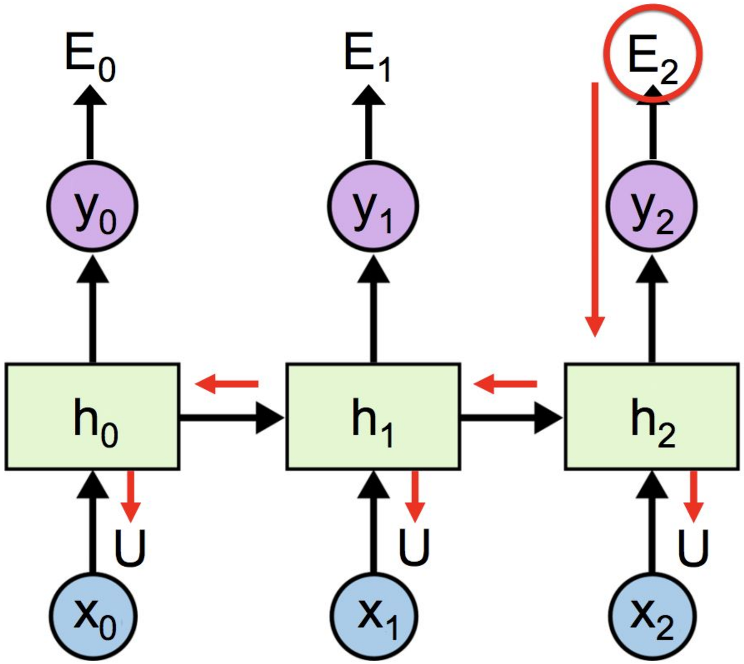


Images from Chris Olah's slides

Backpropagation through time (BPTT)

$$\frac{\partial E_2}{\partial U} = \frac{\partial E_2}{\partial h_2}\left(x_2^T + \frac{\partial h_2}{\partial h_1}\left(x_1^T + \frac{\partial h_1}{\partial h_0}x_0^T\right)\right)$$

Image from Christopher Olah's blog



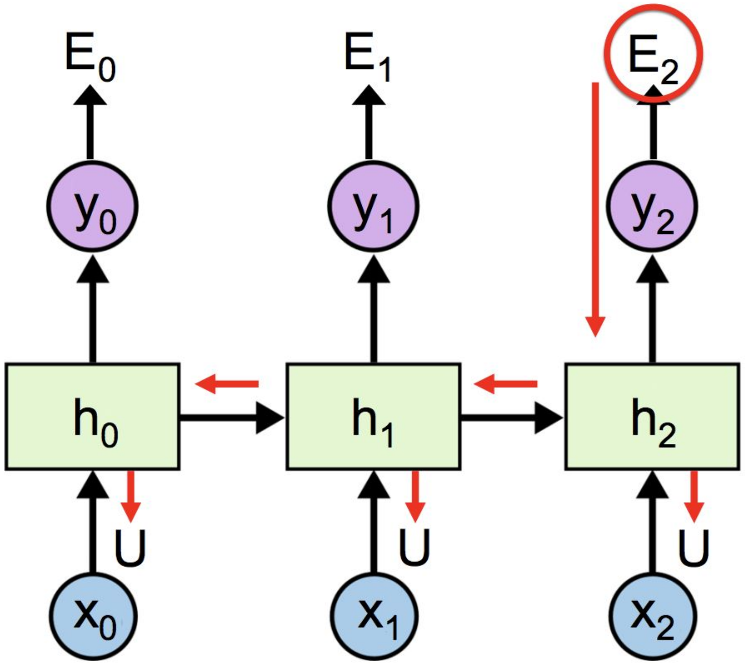
Images from Chris Olah's slides

Mila

Backpropagation through time (BPTT)

$$\frac{\partial E_2}{\partial U} = \frac{\partial E_2}{\partial h_2}\left(x_2^T + \frac{\partial h_2}{\partial h_1}\left(x_1^T + \frac{\partial h_1}{\partial h_0}x_0^T\right)\right)$$

Image from Christopher Olah's blog

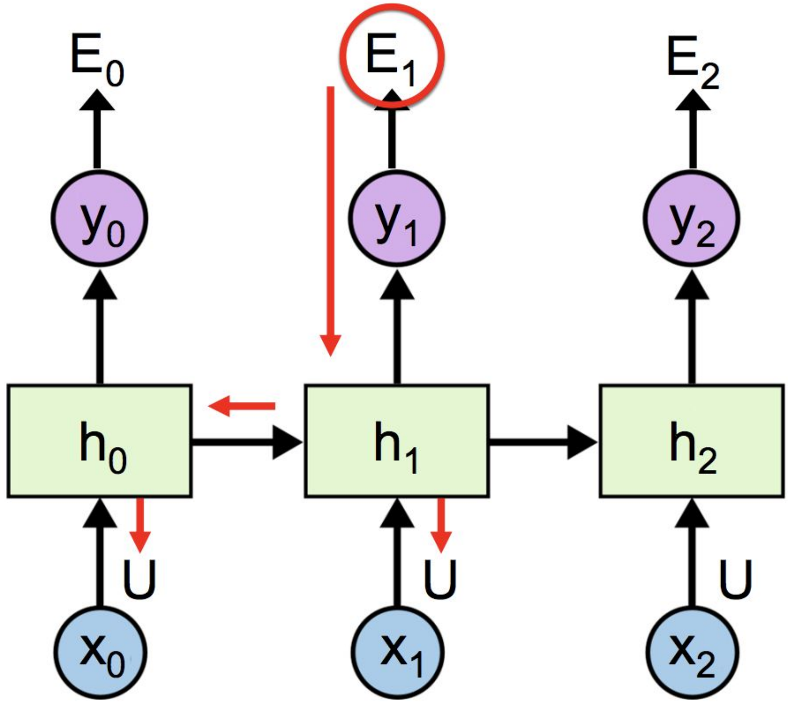


Images from Chris Olah's slides

Backpropagation through time (BPTT)

Same procedure

1. Compute dE_1/dh_1
2. Compute dh_1/dU at time 1
3. Compute dh_0/dU



Images from Chris Olah's slides

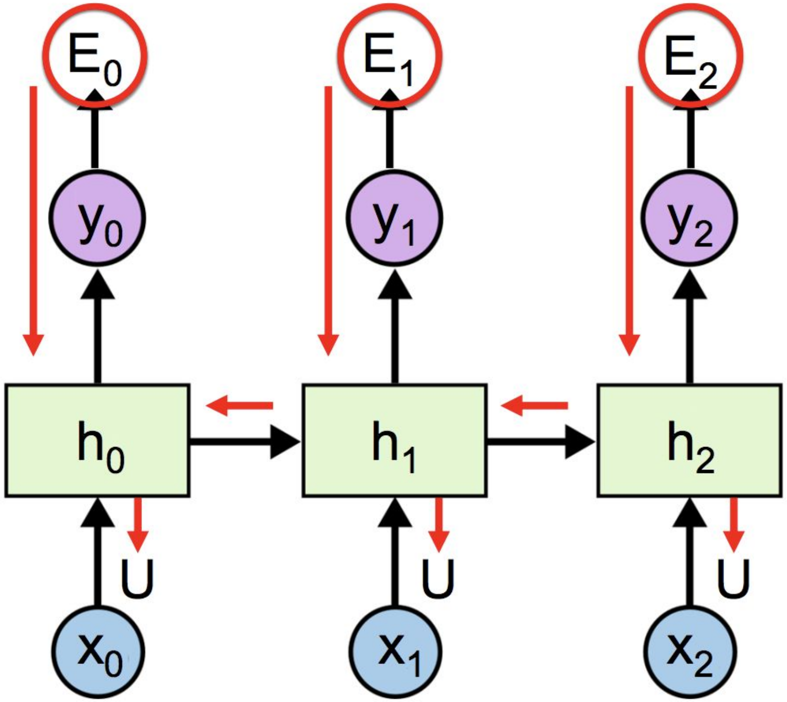
Backpropagation through time (BPTT)

All gradients are summed

$$\frac{\partial E}{\partial U} = \sum_{t=0}^{T} \frac{\partial E_t}{\partial U}$$

$$\frac{\partial E}{\partial W} = \sum_{t=0}^{T} \frac{\partial E_t}{\partial W}$$

$$\frac{\partial E}{\partial W} = \sum_{t=0}^{T} \frac{\partial E_t}{\partial W}$$



Images from Chris Olah's slides

RNN Training

- Trained to predict future from the past
  - h_t is a lossy summary of past
  - Depending on training criteria, h_t decides to keep certain information
  - Difficulty: if y_t depends on the distant past, so h_t has to keep information from many timesteps ago.

    Long term dependencies

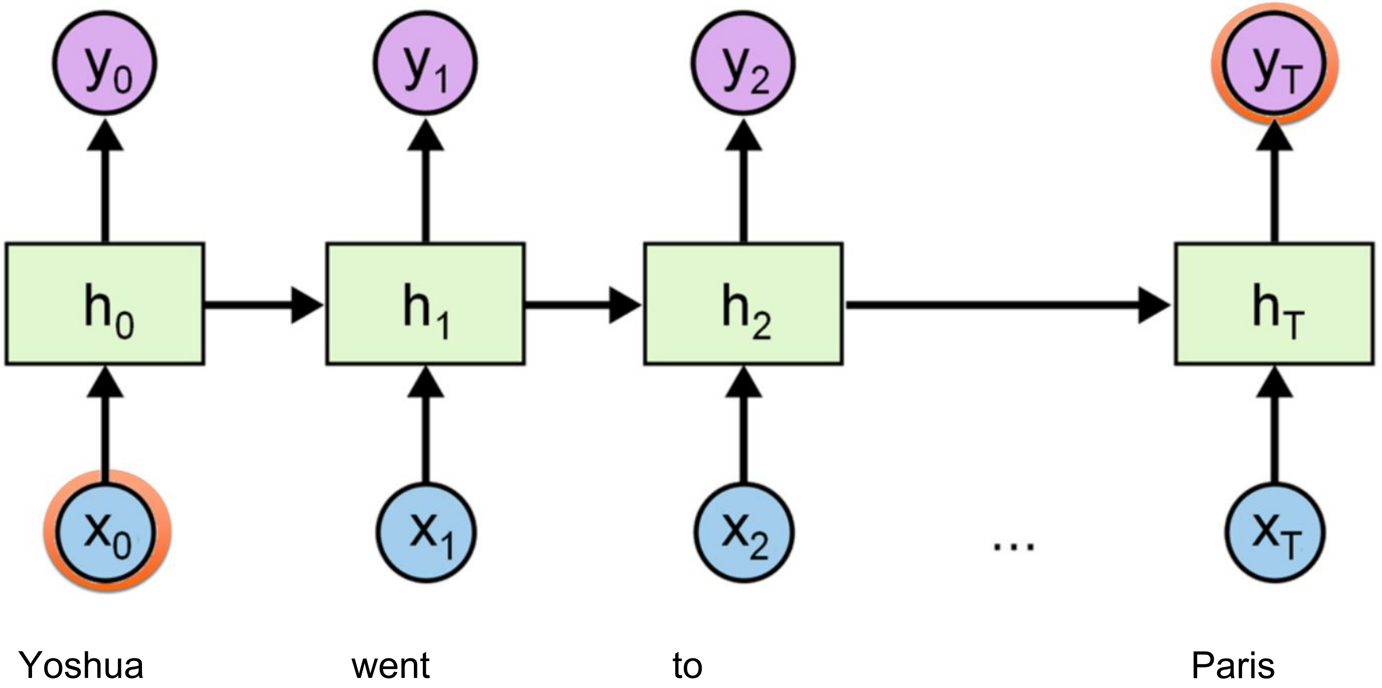
Challenges in RNN training

- Parameters are shared across time
  - # of parameters do not change with longer sequences
  - Consequence:
    - ➢ Optimization issue
    - ➢ Assumption: same param can be used for different time. Conditional probability distribution over variables are same at t+1 compared to t
    - ➢ Exploding/ vanishing gradients

Challenges in RNN training

Long term dependencies

Who went to Paris?



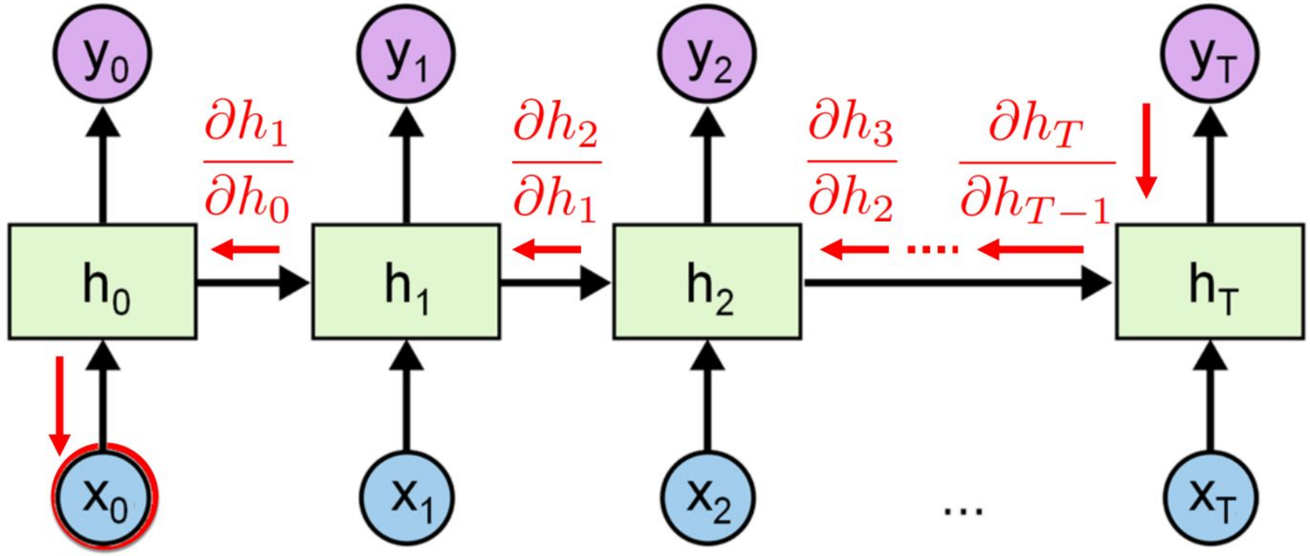
Images from Chris Olah's slides

Mila

Challenges in RNN training

Long term dependencies

- Gradients going far back in time



Images from Chris Olah's slidesc

Mila

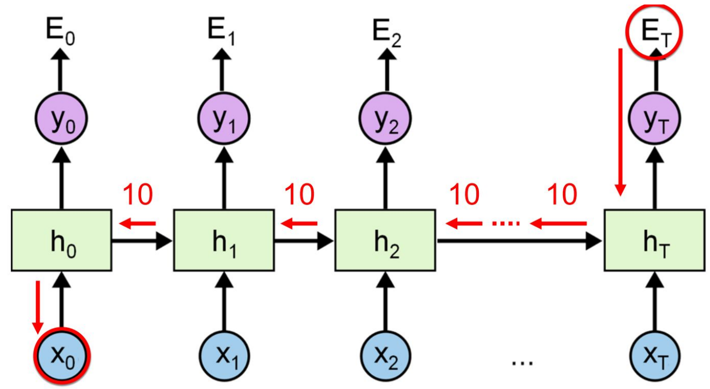
Challenges in RNN training

Long term dependencies

- Gradients going far back in time

$$\frac{\partial y_T}{\partial x_0} = \frac{\partial y_T}{\partial h_T}\frac{\partial h_T}{\partial h_{T-1}}\cdots\frac{\partial h_3}{\partial h_2}\frac{\partial h_2}{\partial h_1}\frac{\partial h_1}{\partial h_0}\frac{\partial h_0}{\partial x_0}$$

- Gradients can become unstable



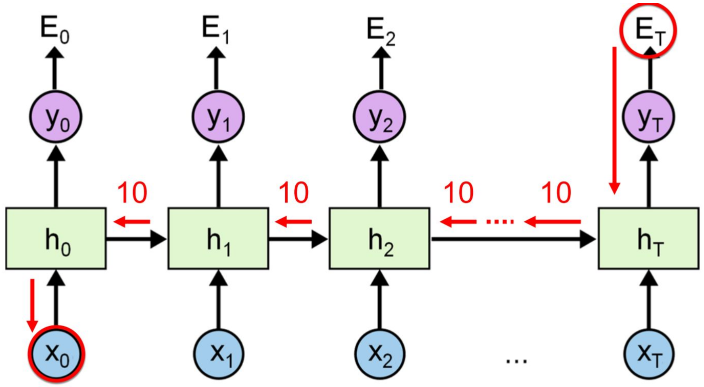
Images from Chris Olah's slides

Exploding gradients

- Gradients going far back in time

$$\frac{\partial y_T}{\partial x_0} = \frac{\partial y_T}{\partial h_T}\frac{\partial h_T}{\partial h_{T-1}} \cdots \frac{\partial h_3}{\partial h_2}\frac{\partial h_2}{\partial h_1}\frac{\partial h_1}{\partial h_0}\frac{\partial h_0}{\partial x_0}$$

- Gradients magnified at each time → exploding gradient!



Images from Chris Olah's slides

Exploding gradients

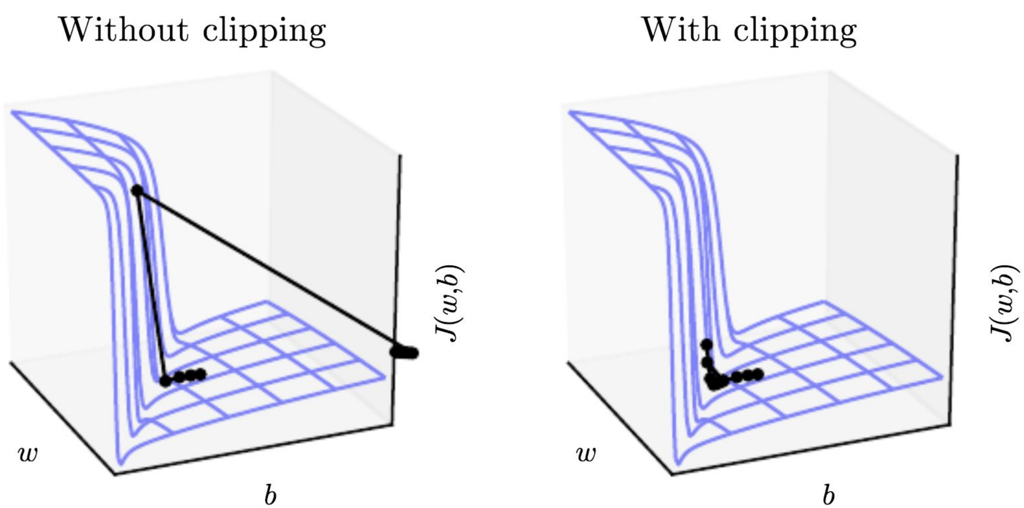
Solution:

- Gradient clipping

$$g = \frac{\partial E}{\partial W}$$

If $\quad ||g|| >= threshold$

then $\quad g = \frac{threshold}{||g||} g$



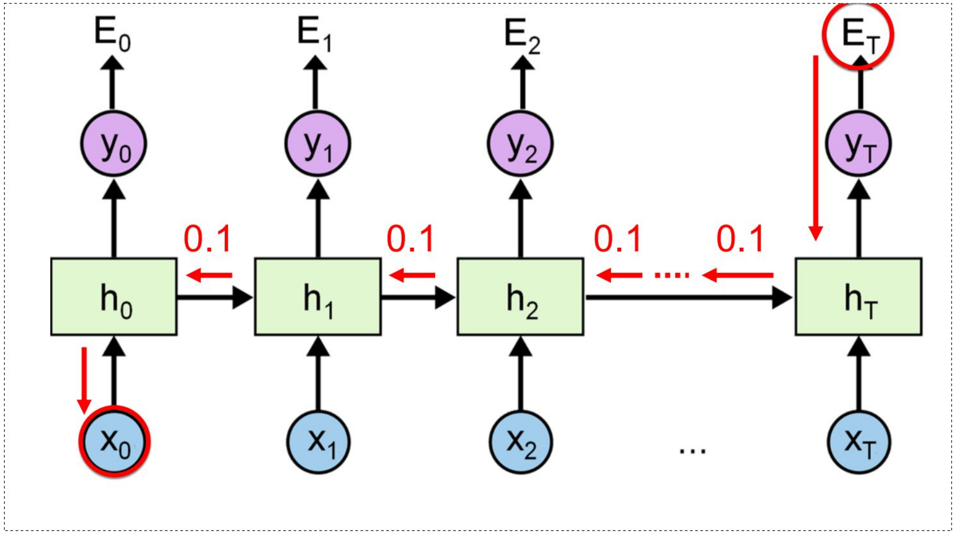
Images from the deep learning book

Vanishing gradients

- Gradients going far back in time

$$\frac{\partial y_T}{\partial x_0} = \frac{\partial y_T}{\partial h_T} \frac{\partial h_T}{\partial h_{T-1}} \cdots \frac{\partial h_3}{\partial h_2} \frac{\partial h_2}{\partial h_1} \frac{\partial h_1}{\partial h_0} \frac{\partial h_0}{\partial x_0}$$

- Gradients contracts at each time → vanishing gradient!



Vanishing gradients

- Gradients going far back in time

$$\frac{\partial y_T}{\partial x_0} = \frac{\partial y_T}{\partial h_T}\frac{\partial h_T}{\partial h_{T-1}} \cdots \frac{\partial h_3}{\partial h_2}\frac{\partial h_2}{\partial h_1}\frac{\partial h_1}{\partial h_0}\frac{\partial h_0}{\partial x_0}$$

- Parameters are stable
  - No explosion
- Can not learn long-term dependencies
  - No easy solution
  - Make architecture changes

Solutions to long-term dependencies

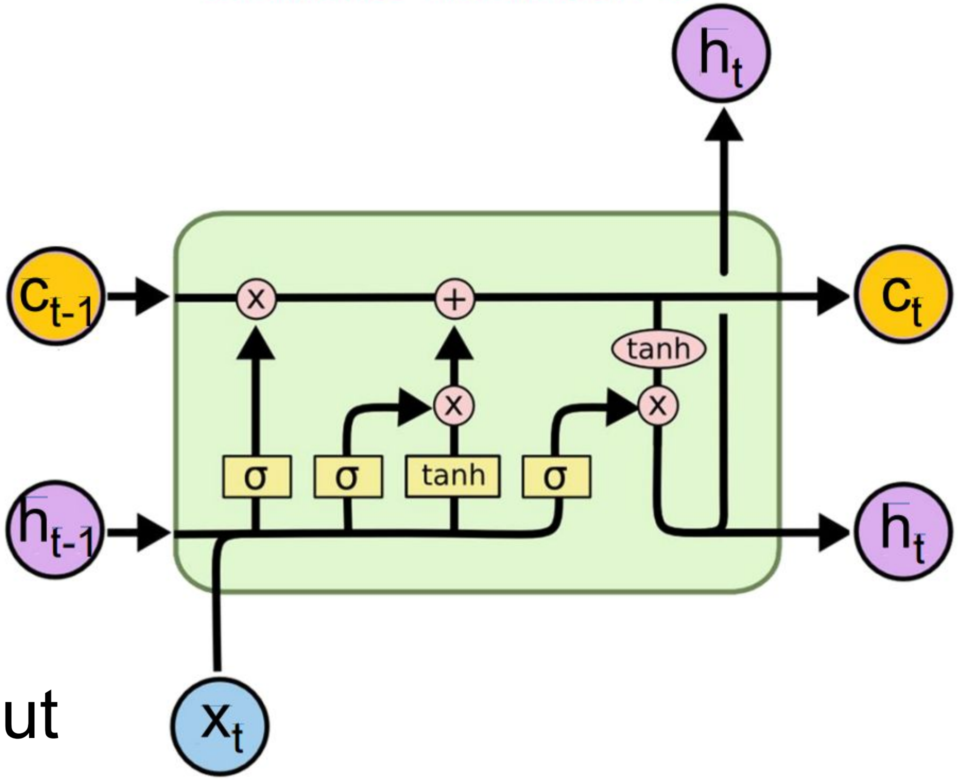
- Gated recurrent neural networks
  - Self-loop for gradients to flow for many steps
  - Model can learn what to forget
  - Long-short term memory (LSTM)
  - Gated recurrent neural networks (GRU)

LSTM

Memory cell c_t

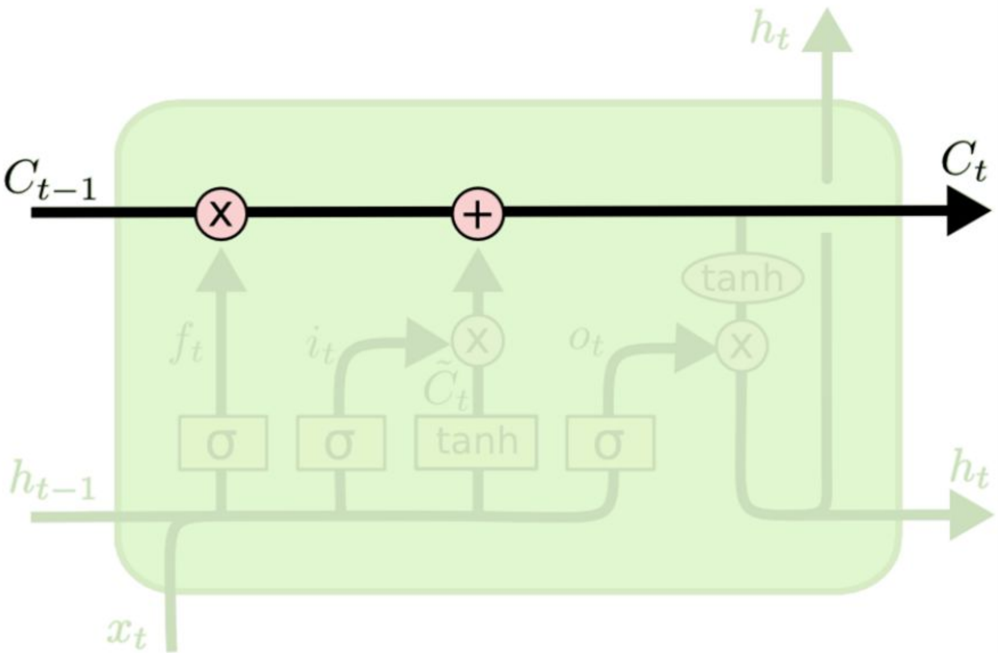
Internal state

Sequence input



Long short term memory (LSTM)

Memory cell is critical



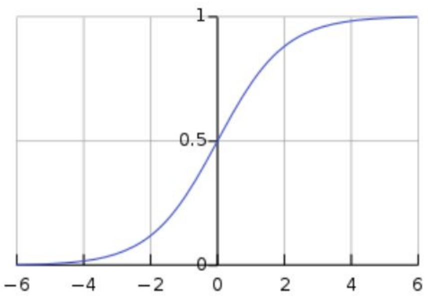
Images from Chris Olah’s slides

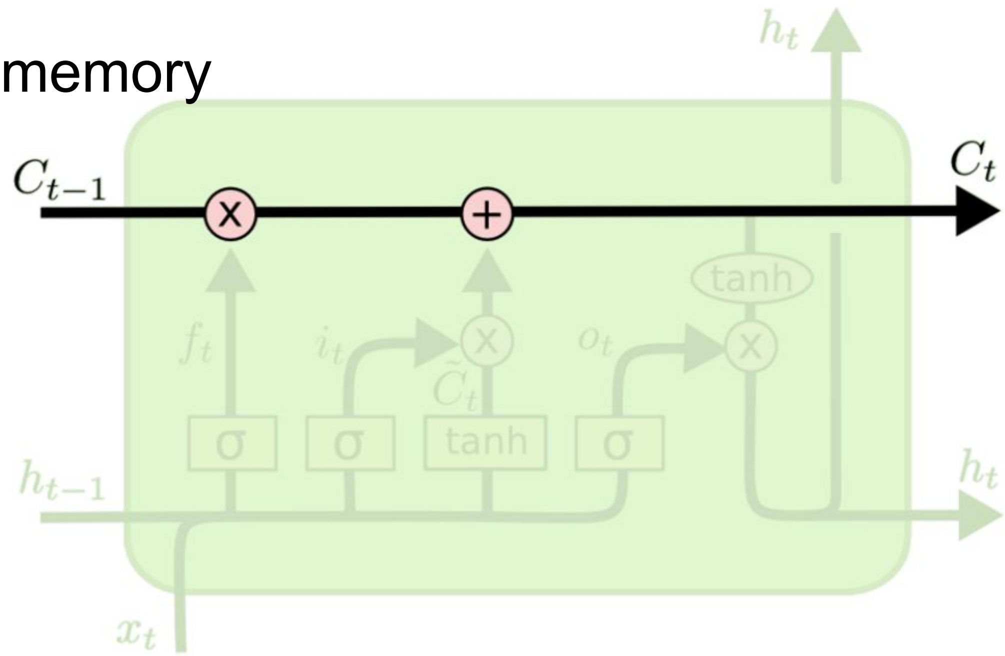
LSTM- forget gates

$$f_t = \sigma(U_f x_t + W_f h_{t-1} + b_f)$$

Decides what to forget in memory

Sigmoid function





Images from Chris Olah’s slides

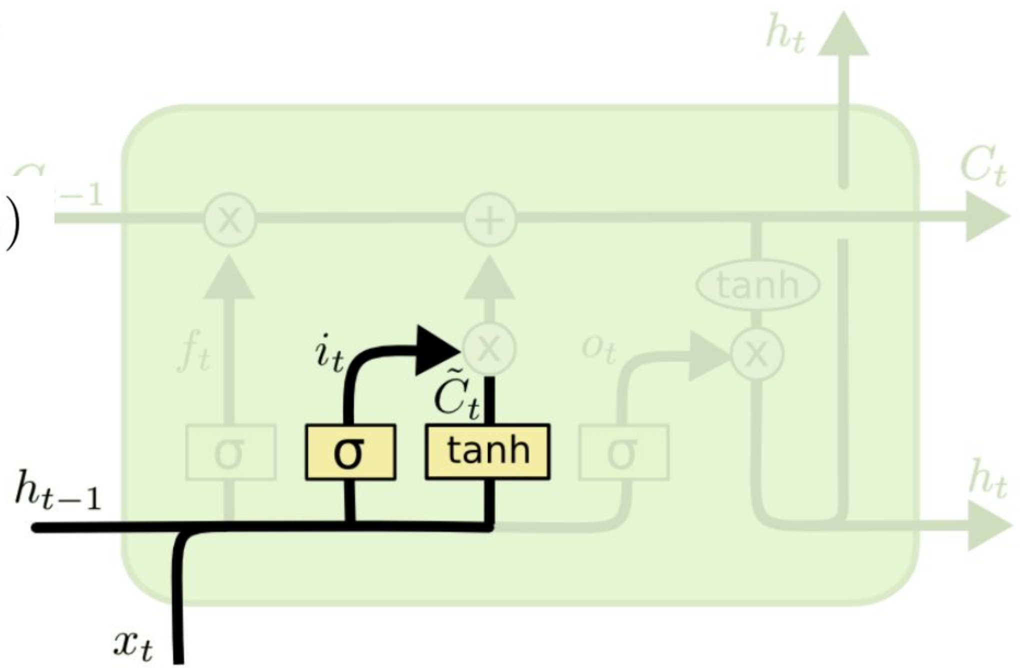
Mila

LSTM- input gates

Controls how much/ what in memory cell

$$i_t = \sigma(U_i x_t + W_i h_{t-1} + b_i)$$

$$g_t = \tanh(U_g x_t + W_g h_{t-1} + b_g)$$



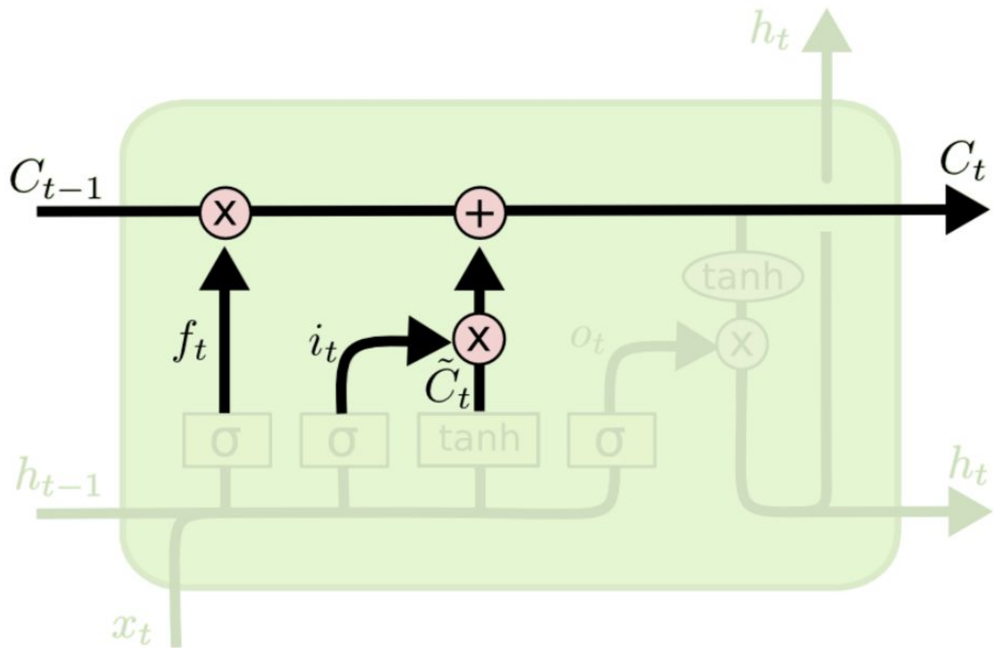
Images from Chris Olah’s slides

LSTM- input gates

$$c_t = i_t \odot g_t + f_t \odot c_{t-1}$$

$\odot$ Element-wise multiply

Input gates allow to add to cell
Forget gates allows to forget



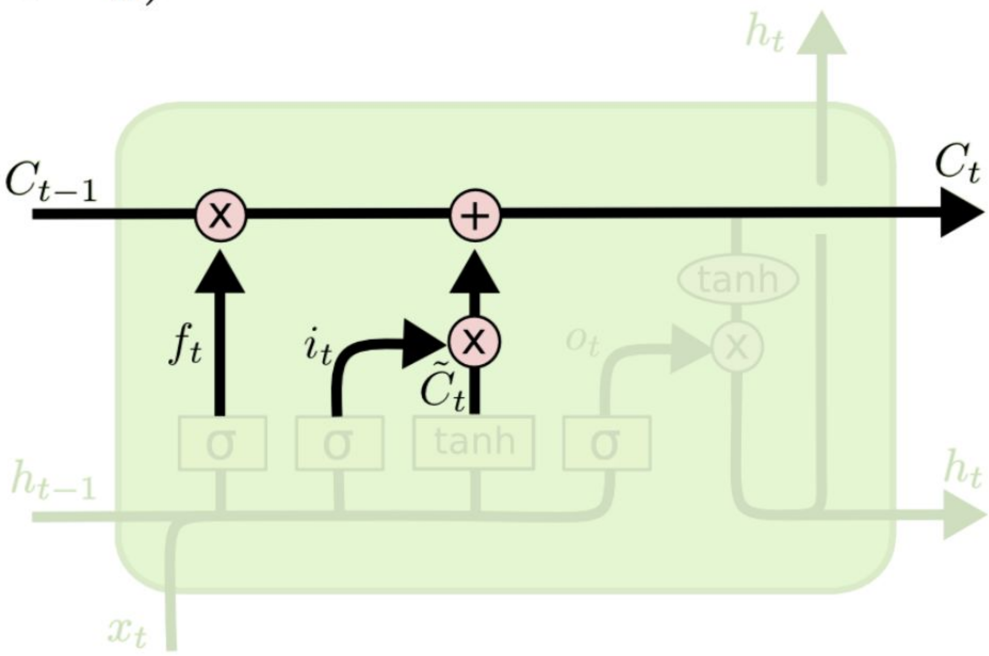
Images from Chris Olah's slides

LSTM - output

$$o_t = \sigma(U_o x_t + W_o h_{t-1} + b_o)$$

Controls what goes out of memory cell

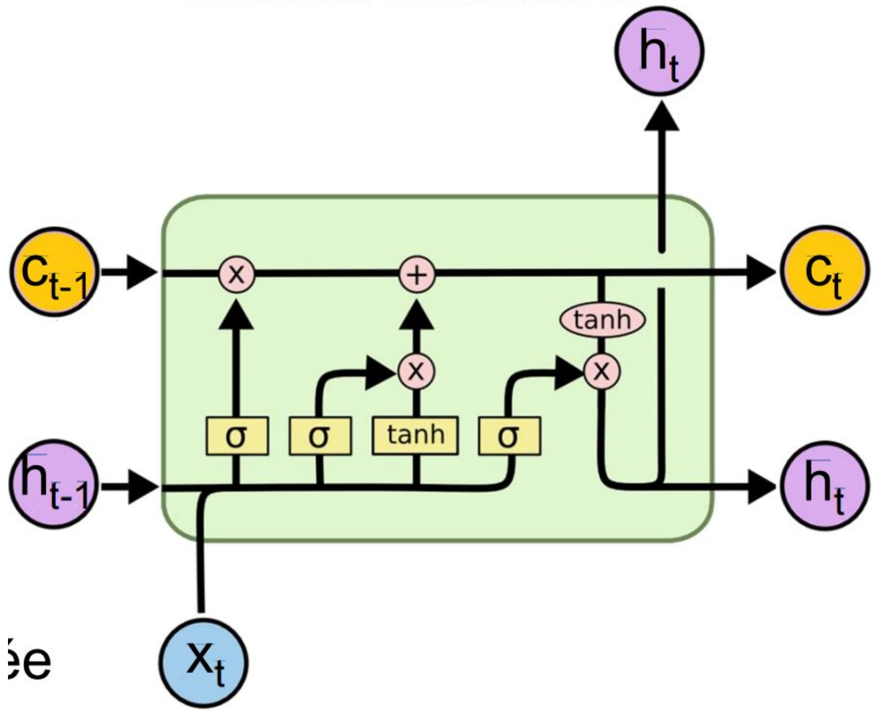
$$h_t = o_t \odot \tanh(c_t)$$



Images from Chris Olah’s slides

LSTM

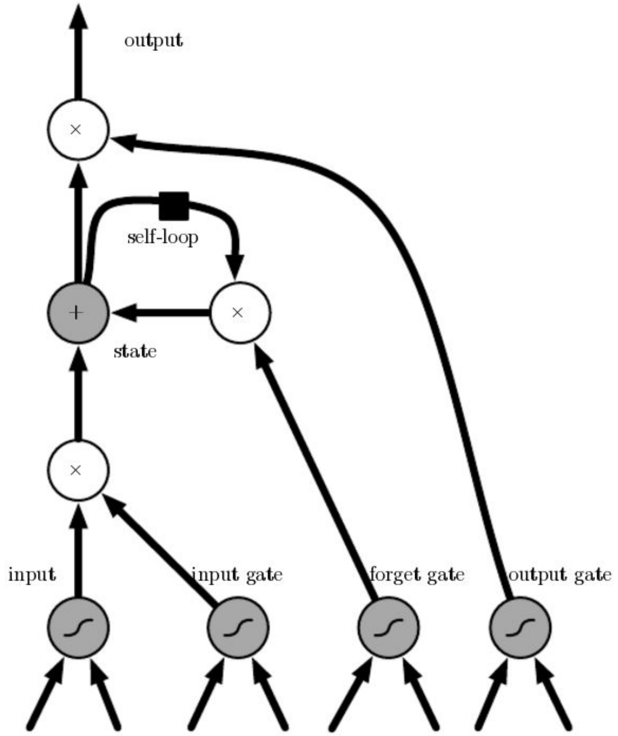
$$i_t = \sigma(U_i x_t + W_i h_{t-1} + b_i)$$
$$f_t = \sigma(U_f x_t + W_f h_{t-1} + b_f)$$
$$o_t = \sigma(U_o x_t + W_o h_{t-1} + b_o)$$
$$g_t = \tanh(U_g x_t + W_g h_{t-1} + b_g)$$
$$c_t = i_t \odot g_t + f_t \odot c_{t-1}$$
$$h_t = o_t \odot \tanh(c_t)$$



Images from Chris Olah’s slides

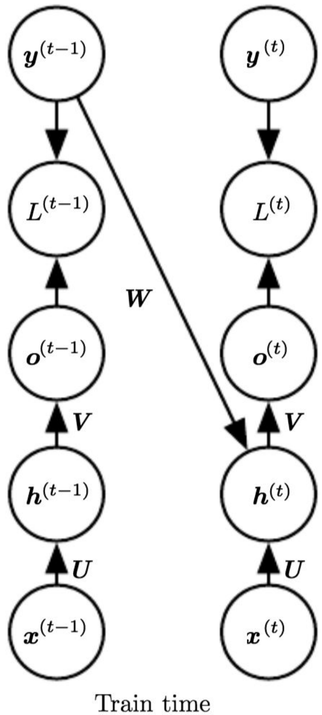
LSTM

- Long-short term memories (LSTM)

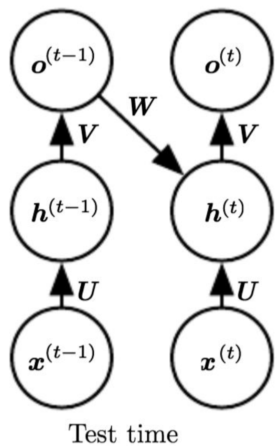


Teacher forcing

- Network uses its output at previous timestep as input



Teacher forcing



Free-running

Images from the “Deep Learning book”

Teacher forcing/ Free-running

- Derived from maximum likelihood criteria

$$\begin{aligned} &logp(y_2, y_1 | x_2, x_1) \\ &= logp(y_2 | y_1, x_2, x_1) + logp(y_1 | x_2, x_1) \end{aligned}$$

- Pass in ground truth y_1 at time step 2
- Used for generative tasks
  - ➢ Language modeling

Issues with teacher forcing

- Difference between train and test
  - During test time, model makes small mistakes
  - Errors accumulate, models has never seen mistakes
  - Low error during training, high during test

Solutions for teacher forcing

No easy solutions

- Scheduled sampling
  - Fed in either free-running or ground truth input during training
- Professor forcing
  - Use a discriminator to match distributions during teacher forcing and free-running

RNNs conditioned on context

For example

- Conditioned language generation
    - ➢ Image captioning

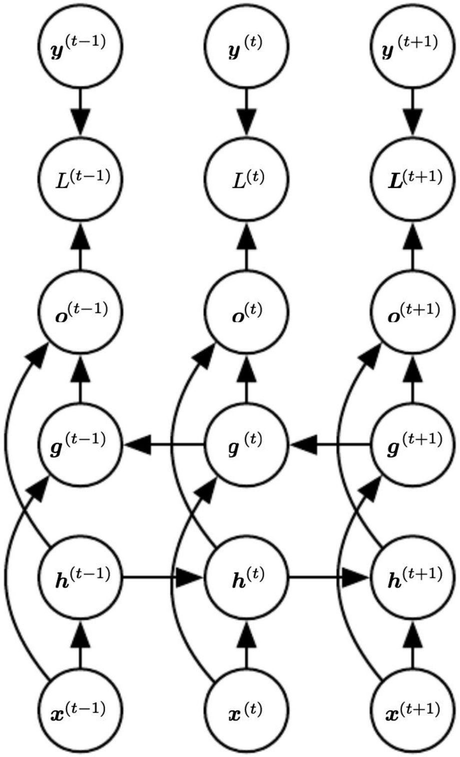
How to pass extra input:

- As an extra input at every step
- As an initial hidden state h_0
- both

Bidirectional RNN

So far, all outputs are conditioned on the past, what about outputs that are conditioned both on the past and the future?

- Speech recognition
  - ➢ Depends on both past and future

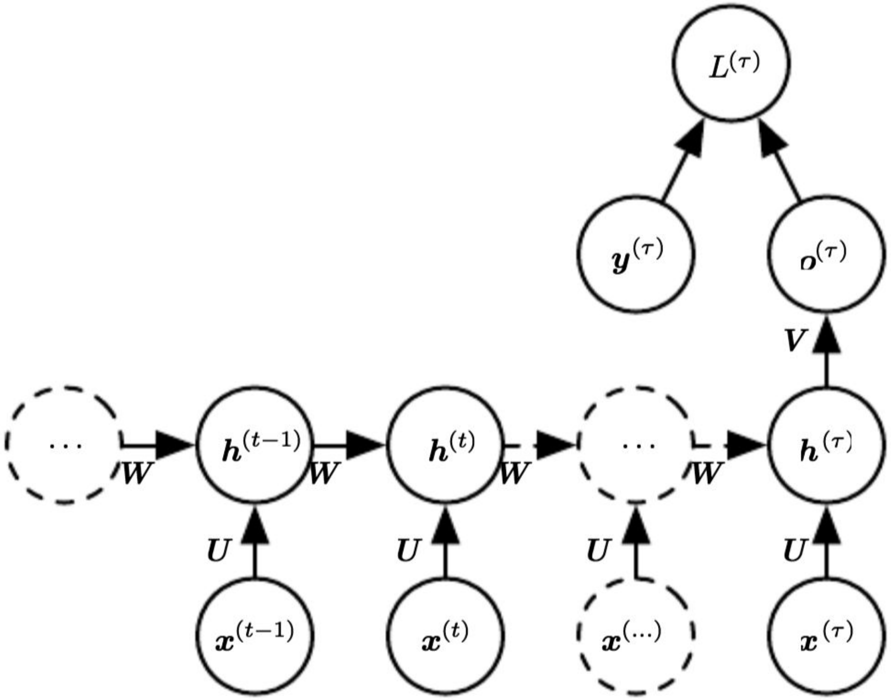


RNN with single output

Image classification

- MNIST recognition
  - ➢ Mapping all input

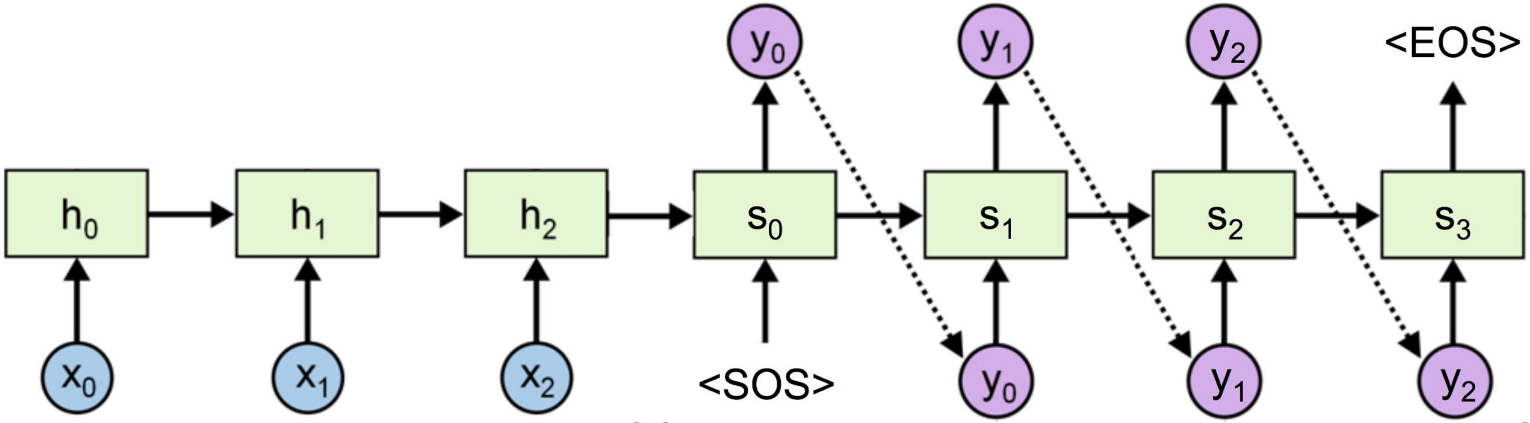
    To single output



Encoder - Decoder RNN

Sequence to sequence

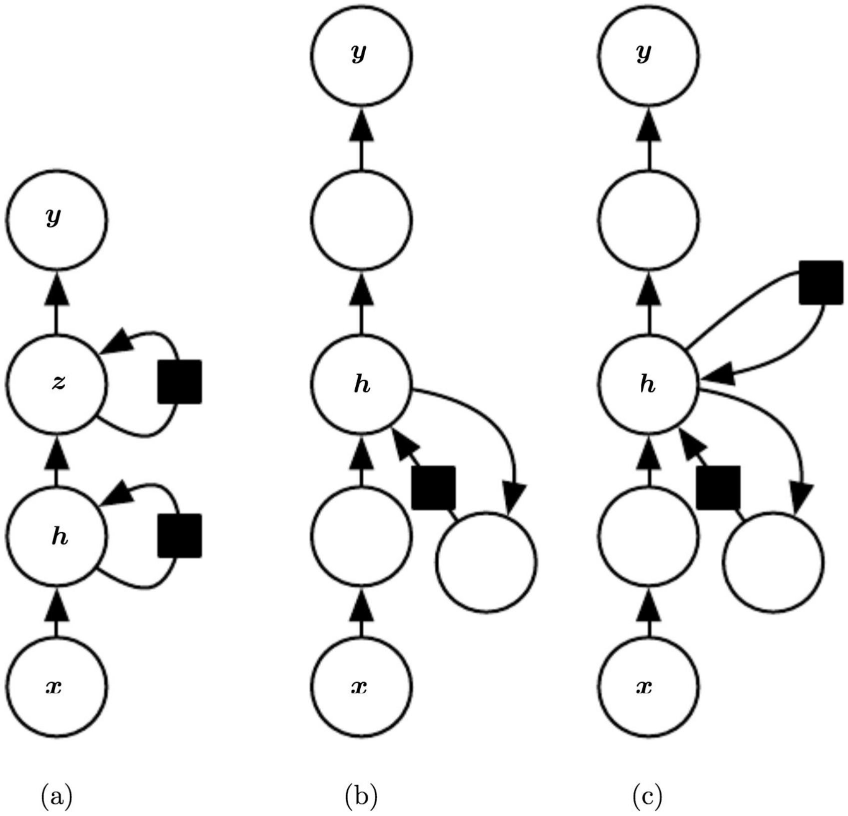
- Encoder
  - ➢ maps inputs to a single vector
- Decoder
  - ➢ a conditional RNN



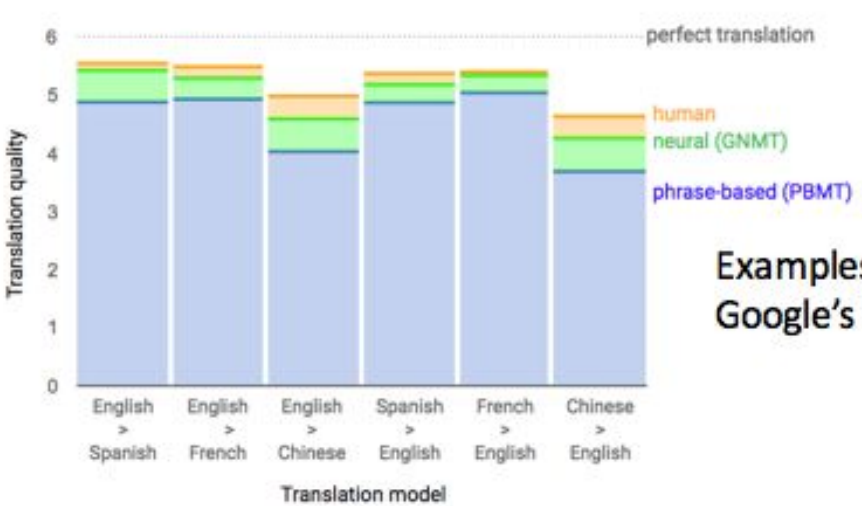
Images from Chris Olah's slides

Deep recurrent networks

- Does depth help?
  - ➢ Does help
  - ➢ Harder to optimize
- How to make it deep?
  - ➢ Stacked RNN
  - ➢ Extra MLP
  - ➢ Skip connections



Comparing Google’s Neural Machine Translation (GNMT) with previous Phrase-Based method (PBMT)



Examples from Google’s system

Input sentence:
李克強此行將啟動中加總理年度對話機制，與加拿大總理杜魯多舉行兩國總理首次年度對話。

Translation (PBMT):	***Translation (GNMT):***	***Translation (human):***
Li Keqiang premier added this line to start the annual dialogue mechanism with the Canadian Prime Minister Trudeau two prime ministers held its first annual session.	Li Keqiang will start the annual dialogue mechanism with Prime Minister Trudeau of Canada and hold the first annual dialogue between the two premiers.	Li Keqiang will initiate the annual dialogue mechanism between premiers of China and Canada during this visit, and hold the first annual dialogue with Premier Trudeau of Canada.

The New York Times

SCIENCE

Researchers Announce Advance in Image-Recognition Software

By JOHN MARKOFF NOV. 17, 2014

MOUNTAIN VIEW, Calif. — Two groups of scientists, working independently, have created artificial intelligence software capable of **recognizing and describing the content of photographs and videos** with far greater accuracy than ever before, sometimes even mimicking human levels of understanding.

Captioned by Human and by Google's Experimental Program

Human: "A young hockey player playing in the ice rink."
Computer model: "Two hockey players are fighting over the puck."

2 of 6

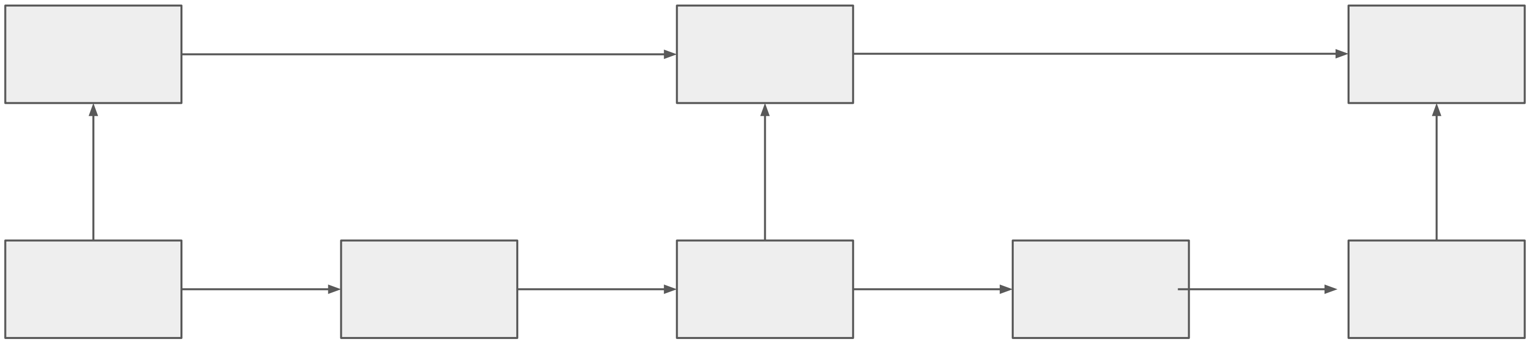
Skip connections through time

Helps with exploding and vanishing gradients across time

- Gradients can propagate over longer spans through skip connections

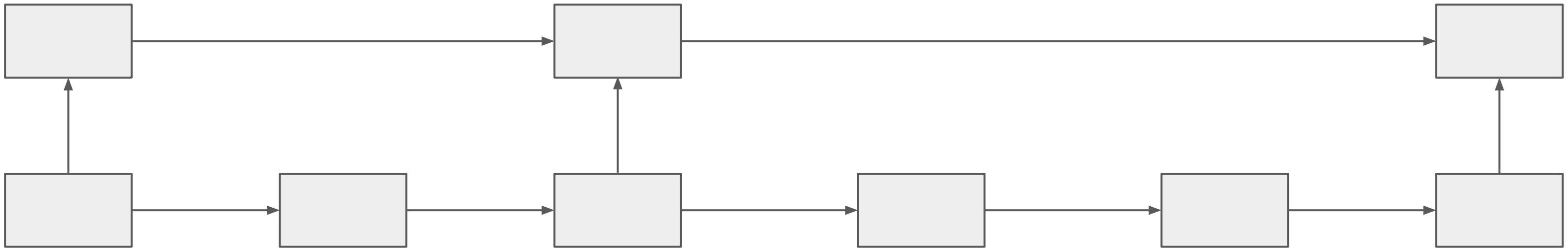
Learning hierarchical representation in RNNs?

- Updates higher layers less frequently.
- Clockwork RNN (Bengio et. al, 94, Schmidhuber et. al, 14)
  - Updates at fixed time steps
- Gradients can propagate over longer spans through slow time-scale paths



Learning hierarchical representation in RNNs?

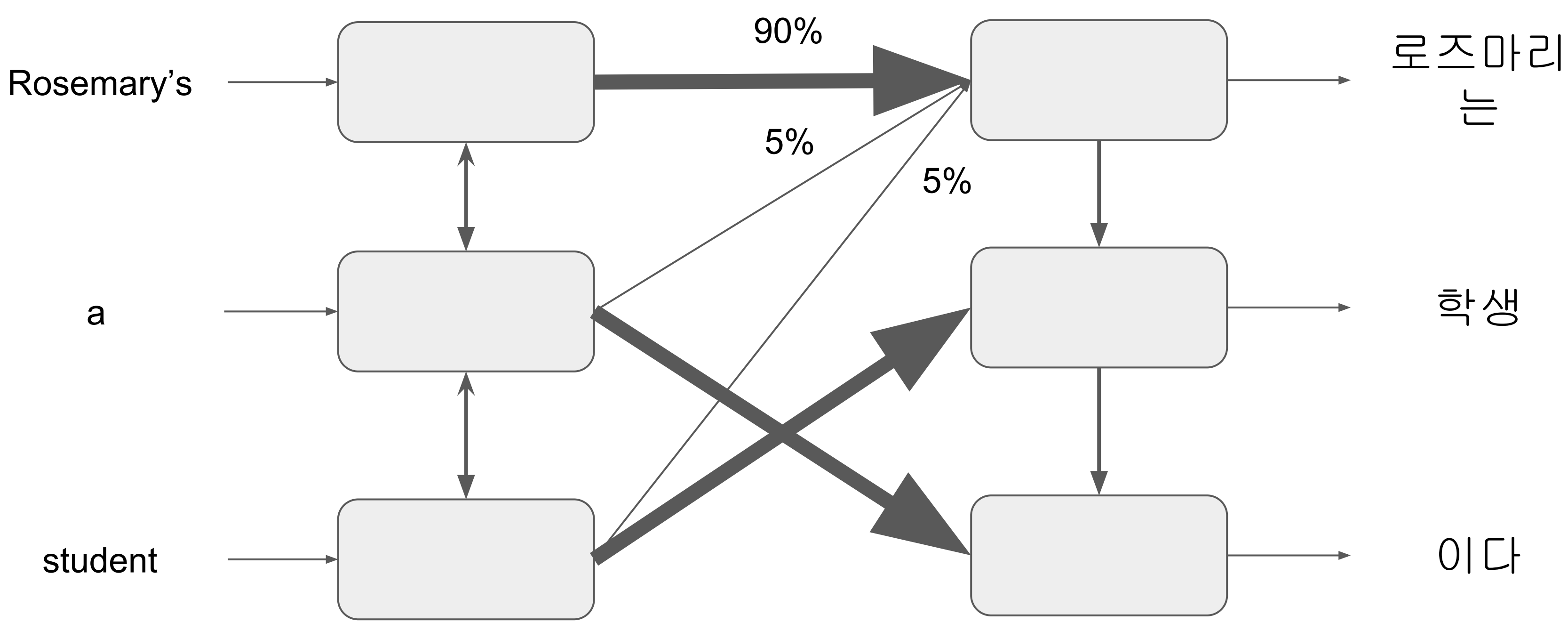
- Previous work: Hierarchical Multiscale Recurrent Neural Network (Chung et. al, 16)
  - Learns when to update.
- Hierarchical Encoder Network (our method)
  - Learns when to update conditionally.



Neural networks with attention

- Encoder-decoder
  - Last hidden state contains all information for inputs
  - Bottleneck state
  - How to make it better?
- Attention
  - $$a_j = \frac{e^{A(z_i, h_j)}}{\sum_{j'} e^{A(z_i, h_{j'})}}$$
    $$r = \sum_j a_j h_j$$

RNNs with attention



Open research questions

- How to efficiently assign credit in RNNs
  - Is there credit diffusion? How to handle that
  - Do we have to perform BPTT?
- How to regularize RNN for better longer term planning
- How to build models that are better adjusted to the difference between training (teacher forcing) and test (free-running) mode?

Sparse Attentive Backtracking
Temporal credit assignment through reminding

(NIPS 2018 spotlight)

Nan Rosemary Ke, Anirudh Goyal, Olexa Blanik

Jonathan Binas, Chris Pal, Mike Mozer, Yoshua Bengio

MILA

How credit assignment through time is done in RNNS?

Usually Backpropagation Through Time (BPTT)

- Detailed reminding of all past events
- Assign soft credit to almost all past events
- Diffusion of credit?

Credit assignment through time and memory

- Humans have memory recall
- Automatic reminding
  - Triggered by contextual features
  - Can serve a useful computation role in ongoing cognition
  - What about credit assignment?

Credit assignment in time and automatic reminding

Failing a class

- Did badly in the exam
- Did not study during the semester
- Change the parameters/ weights of your model, so next time you will study for a class

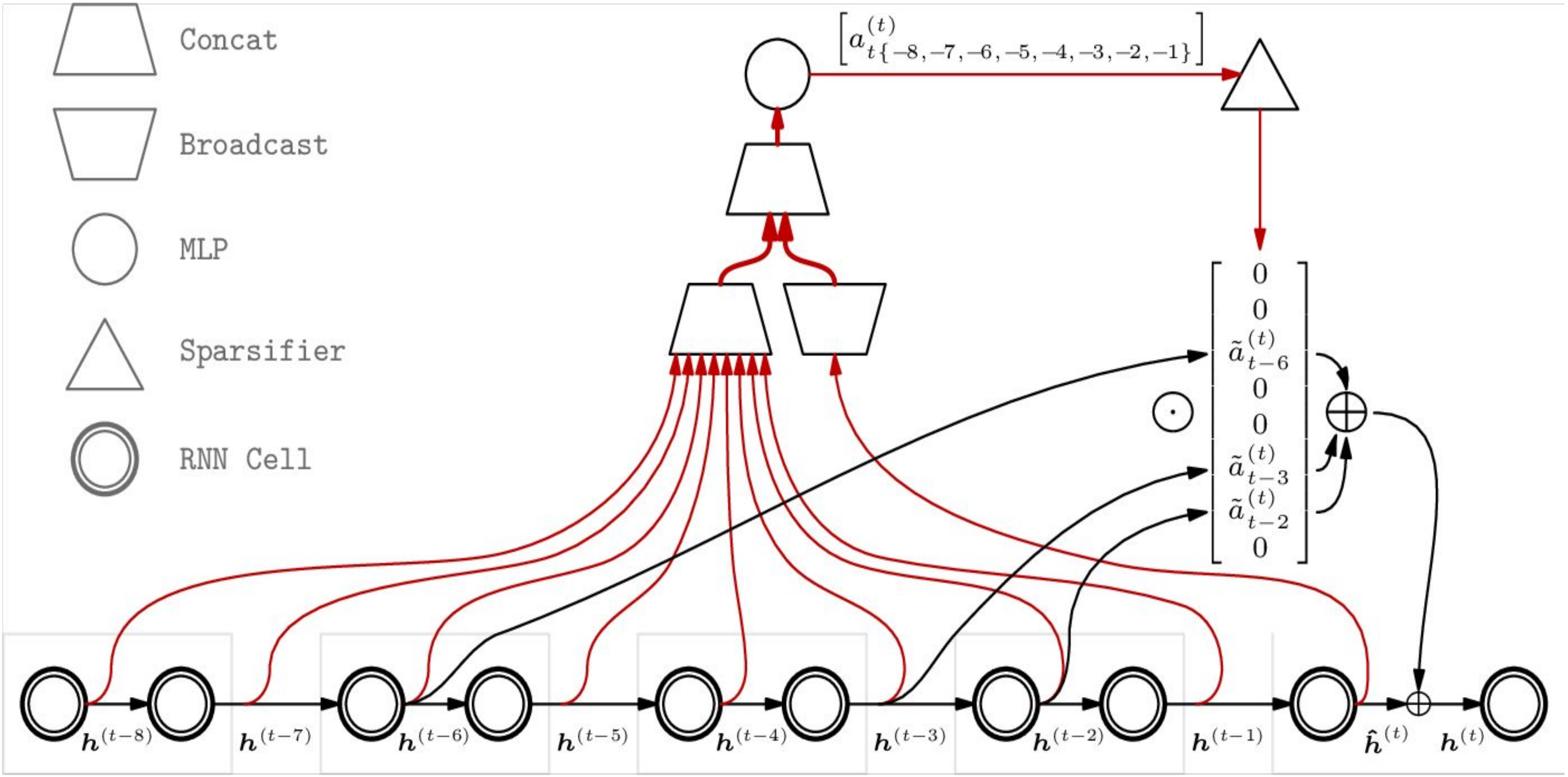
Can credit assignment through a few states work?

- Can we assign credit through only a few states, instead of all states.
- How to pick the states to assign credit to?

Sparse Attentive Backtracking

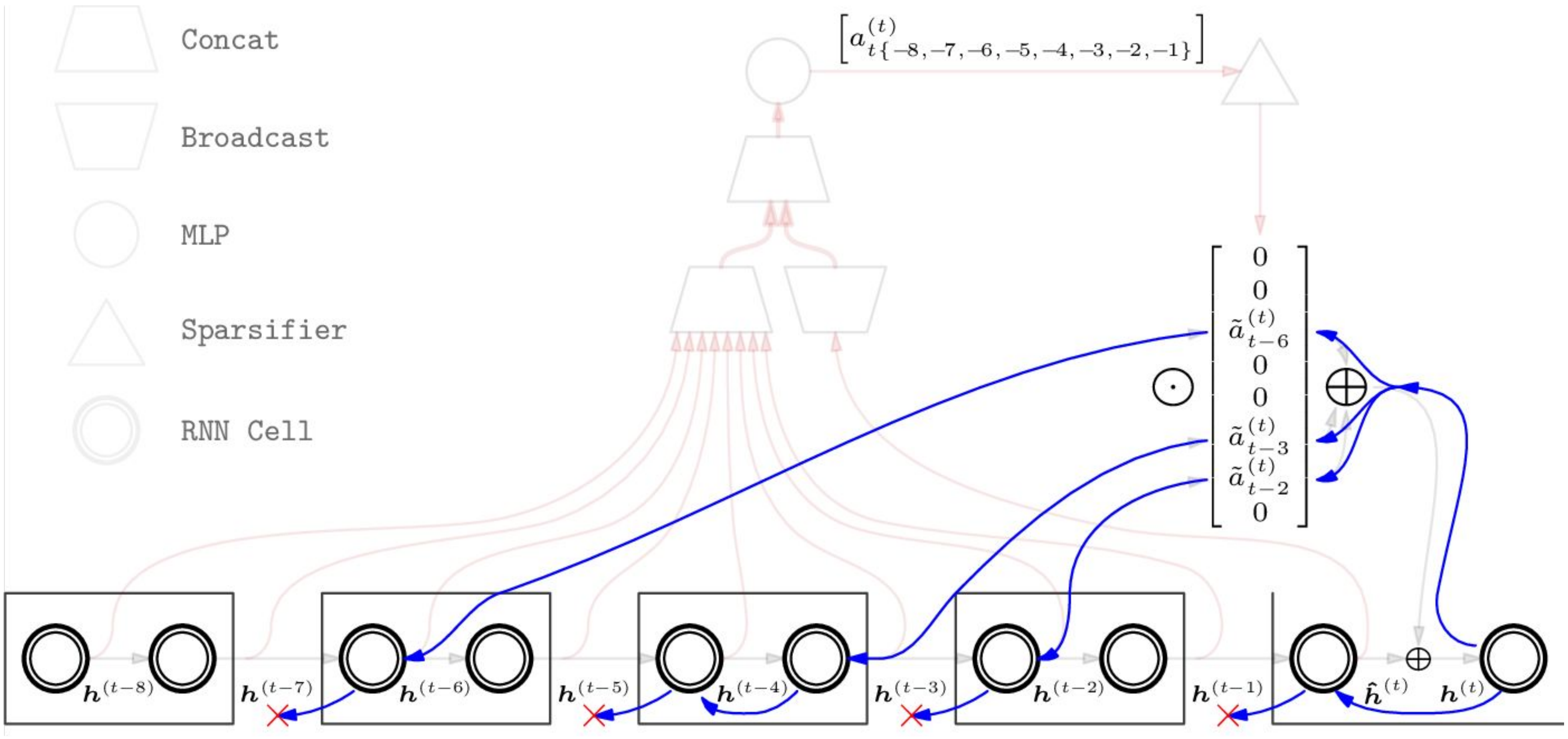
- Use Attention mechanism to select previous time steps to do local backprop - reminding
  - Local backprop - truncated BPTT
  - Select previous hidden states - sparsely

Forward pass



MILA

Backward pass



MILA

Sparse Attentive Backtracking

- **RNN models** Not support such operations on their past.
  - Make some architectural additions.
- **Forward Pass**
  - Selects microstates from the macrostate
  - Summarizes them
  - Incorporates this summary into the next hidden state
- **Backward Pass**: gradient flows through
  - (Truncated) master chain linking consecutive hidden states
  - + Microstates selected in the forward pass

MILA

Macro-state and Micro-state

- Ever-growing macro states
  - Adaptive, dynamic connections
  - Past micro-states linked to current hidden states through dynamic decisions.
- **Skip connections** - Propagate Information over very long sequences.
  - Natural for learning long-term dependencies

Sparse Attention Process

- Linear transformation computes scalar attention weight
  - $$a_{i_1}^{(t)} = \boldsymbol{w}_1^\top \boldsymbol{m}^{(i)} + \boldsymbol{w}_2^\top \hat{\boldsymbol{h}}^{(t)}$$
  - $$a_i^{(t)} = \boldsymbol{w}_3^\top \tanh(\boldsymbol{a}_{i_1}^{(t)})$$
- Mask out all but top K attention weights
  - $$\tilde{a}_i^{(t)} = \mathrm{ReLU}\left(a_i^{(t)} - a_{k\mathrm{top}}^{(t)}\right)$$
- Summarization of micro states
  - $$\boldsymbol{s}^{(t)} = \sum_{\boldsymbol{m}^{(i)} \in \mathcal{M}} \tilde{a}_i^{(t)} \boldsymbol{m}^{(i)}$$
- Hidden state

$$\boldsymbol{h}^{(t)} = \hat{\boldsymbol{h}}^{(t)} + \boldsymbol{s}^{(t)}$$

Copying task

	k_{trunc}	k_{top}	Copying (T=100) acc.	Copying (T=100) CE_{10}	Copying (T=100) CE	Copying (T=200) acc.	Copying (T=200) CE_{10}	Copying (T=200) CE	Copying (T=300) acc.	Copying (T=300) CE_{10}	Copying (T=300) CE
LSTM	*full BPTT*		99.8	0.030	0.002	56.0	1.07	0.046	35.9	0.197	0.047
LSTM	*full self-attn.*		100.0	0.0008	0.0000	100.0	0.001	0.000	100.0	0.002	7.5e-5
LSTM	1	-	20.6	1.984	0.165				14.0	2.077	0.065
LSTM	5	-	31.0	1.737	0.145	17.1	2.03	0.092			
LSTM	10	-	29.6	1.772	0.148	20.2	1.98	0.090			
LSTM	20	-	30.5	1.714	0.143	35.8	1.61	0.073	25.7	1.848	0.197
LSTM	150	-	-	-	-	35.0	1.596	0.073	24.4	1.857	0.058
SAB	1	1	57.9	1.041	0.087	39.9	1.516	0.069	43.1	0.231	0.045
SAB	1	5	**100.0**	**0.001**	**0.000**				89.1	0.383	0.012
SAB	5	5	**100.0**	**0.000**	**0.000**	**100.0**	**0.000**	**0.000**	**99.9**	**0.007**	**0.001**
SAB	10	10	**100.0**	**0.000**	**0.001**	**100.0**	**0.000**	**0.000**			

Table 1: Test accuracy and cross-entropy (CE) loss performance on the copying task with sequence lengths of T=100, 200, and 300. Accuracies are given in percent for the last 10 characters. CE_{10} corresponds to the CE loss on the last 10 characters. These results are with mental updates; Compare with Table 3 for without.

Adding task

Adding			T=200	T=400
	k_{trunc}	k_{top}	CE	CE
LSTM	*full BPTT*		4.59e-6	1.554e-7
	full self-attn.		5.541e-8	4.972e-7
	20	-	1.1e-3	
	50	-	3.0e-4	
	100	-		6.8e-4
SAB	5	5	4.26e-5	
	5	10		2.30e-4
	10	10	**2.0e-6**	1.001e-5

Language Modeling

Language	k_{trunc}	k_{top}	k_{att}	PTB BPC	Text8 BPC
LSTM	*full BPTT*			1.36	1.42
LSTM	1	-	-	1.47	1.56
LSTM	5	-	-	1.44	
LSTM	20	-	-	1.40	
SAB	10	5	10	1.42	1.47
SAB	10	10	10	1.40	1.45
SAB	20	5	20	1.39	1.45
SAB	20	10	20	1.37	1.44

Classification + Comparison to Transformer

- Outperforms Transformers on CIFAR10

Image class.				pMNIST	CIFAR10
	k_{trunc}	k_{top}	k_{att}	acc.	acc.
LSTM	*full BPTT*			90.3	58.3
	300	-	-		51.3
SAB	20	5	20	89.8	
	20	10	20	90.9	
	50	10	50	**94.2**	
	16	10	16		**64.5**
Transformer (Vasvani'17)				**97.9**	62.2

Table 4: Test accuracy for the permutated MNIST and CIFAR10 classification tasks.

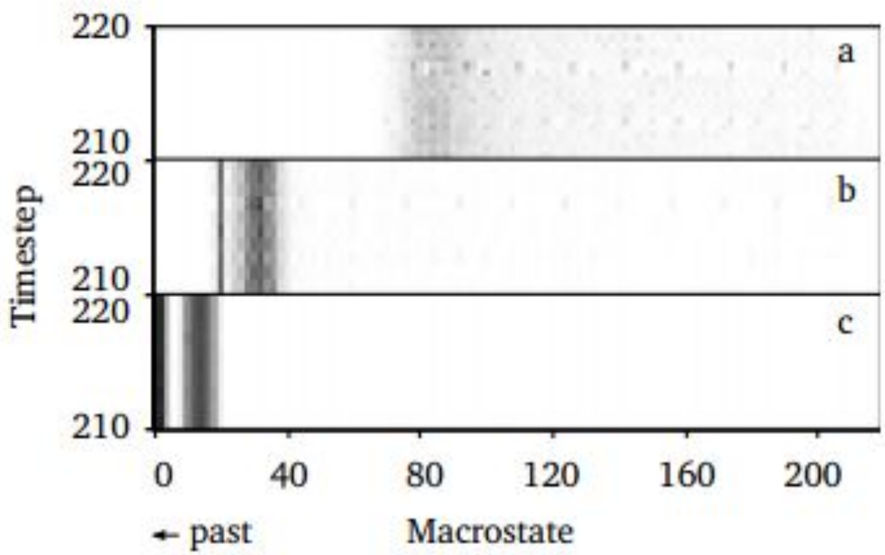
MILA

Ablation Study

Ablation			Copying, T=100			Adding, T=200
	k_{trunc}	k_{top}	acc.	$CE_{\text{last 10}}$	CE	CE
no MU	1	1	49.0	1.252	0.104	
	5	5	98.3	0.042	0.0036	
	10	10	99.6	0.022	0.0018	2.171e-6
	5	all	40.5	1.529	0.127	

MILA

Sparse Attention



Generalization results

Transfer Learning Results

Copy len. (T)	LSTM	LSTM +self-a.	SAB
100	99%	100%	99%
200	34%	52%	**95%**
300	25%	28%	**83%**
400	21%	20%	**75%**
2000	12%	12%	**47%**
5000	12%	OOM	**41%**

Table 5: Transfer performance (Accuracy for last 10 digits) for models trained on $T = 100$ copy memory task. Comparisons to LSTM and LSTM with full self-attention trained with BPTT.

Future Work

- **Content-based rule to retrieve memory**
    - Humans show a systematic dependence on many content: salient, extreme, unusual, and unexpected experiences are more likely to be stored and subsequently remembered
- **Model Based RL** : Ability to plan many steps ahead in time leveraging the past.