

Deep Learning - COSC2779/2972

Modelling Sequential (Time Series) Data

Dr. Ruwan Tennakoon



Semester 2, 2022

Reference: *Chapter 10: Ian Goodfellow et. al., "Deep Learning", MIT Press, 2016.*



We started out with the intention of creating an AI that could complete Mozart's unfinished composition *Lacrimosa* —the eighth sequence of the Requiem—which was written only till the eighth bar at the time of his passing.

We ended up creating a deep neural network called maia—play on the term 'music ai'—that can generate original piano solo compositions by learning patterns of harmony, rhythm, and style from a corpus of classical music by composers like Mozart, Chopin, and Bach.

Lacrimosa

(from the *Requiem* in D minor)

Wolfgang Mozart (1756-1791)
completed by maia (2018)

A musical score for piano solo. It consists of two staves: a treble staff and a bass staff. The score is in 12/8 time and D minor. The piano keys are shown in a light beige color, and the musical notes are represented by black dots on the staff lines. The score begins with a dynamic marking 'p' (piano).

MAIA - AI for music creation

Automated Image Captioning with ConvNets and Recurrent Nets

Andrej Karpathy, Fei-Fei Li



a young boy is holding a baseball bat



a group of people sitting at a table with wine glasses

Automated Image Captioning with ConvNets and Recurrent Nets

Andrej Karpathy, Fei-Fei Li



a baby laying on a bed with a stuffed bear



a horse is standing in the middle of a road



a toilet with a seat up in a bathroom

NeuralTalk Sentence Generation Results

Outline

- 1 Sequential Data
- 2 Recurrent Neural Networks (RNN)
- 3 Variants of RNN
- 4 Gated Recurrent Units Networks
- 5 Long Short Term Memory networks
- 6 Bi-Directional and Deep RNN

Revision

A look back at what we have learned:

- Deep neural network Building blocks:
 - **Week2:** Feed forward NN Model and cost functions.
 - **Week3:** Optimising deep models challenges and solutions.
 - **Week4:** Convolution neural network: for data with spatial structure.
 -
- Case study:
 - **Week5:** Famous networks for computer vision applications
- Putting things together:
 - **Week6:** Practice methodology

A look back at what we have learned:

- Deep neural network Building blocks:
 - **Week2:** Feed forward NN Model and cost functions.
 - **Week3:** Optimising deep models challenges and solutions.
 - **Week4:** Convolution neural network: for data with spatial structure.
 -
- Case study:
 - **Week5:** Famous networks for computer vision applications
- Putting things together:
 - **Week6:** Practice methodology



Image: Unsplash

A look back at what we have learned:

- Deep neural network Building blocks:
 - **Week2:** Feed forward NN Model and cost functions.
 - **Week3:** Optimising deep models challenges and solutions.
 - **Week4:** Convolution neural network: for data with spatial structure.
 - **Week7-8:** Recurrent neural network: for data with sequential structure.
- Case study:
 - **Week5:** Famous networks for computer vision applications
- Putting things together:
 - **Week6:** Practice methodology

Type of NN?

- Supervised learning with fixed-size vectors: Deep Feed-forward models
- Input has topological structure: CNN.
- Input or Output is a sequence: LSTM or GRU (will be discussed in future).

Objectives of this lecture

- Understand the main building blocks of RNNs designed to handle sequential data.
- Improvements to basic structure and the intuition.

Outline

- 1 Sequential Data
- 2 Recurrent Neural Networks (RNN)
- 3 Variants of RNN
- 4 Gated Recurrent Units Networks
- 5 Long Short Term Memory networks
- 6 Bi-Directional and Deep RNN

Why do we need another model type?

Sentiment analysis:

“My experience so far has been fantastic” → *Positive*

“Your support team is useless” → *Negative*

How can we represent this data (for NN)?

Why do we need another model type?

Sentiment analysis:

“My experience so far has been fantastic” → *Positive*

“Your support team is useless” → *Negative*

How can we represent this data (for NN)?

- Character level: Each character mapped to a number.

Why do we need another model type?

Sentiment analysis:

“My experience so far has been fantastic” → *Positive*

“Your support team is useless” → *Negative*

How can we represent this data (for NN)?

- Character level: Each character mapped to a number.
- Word level

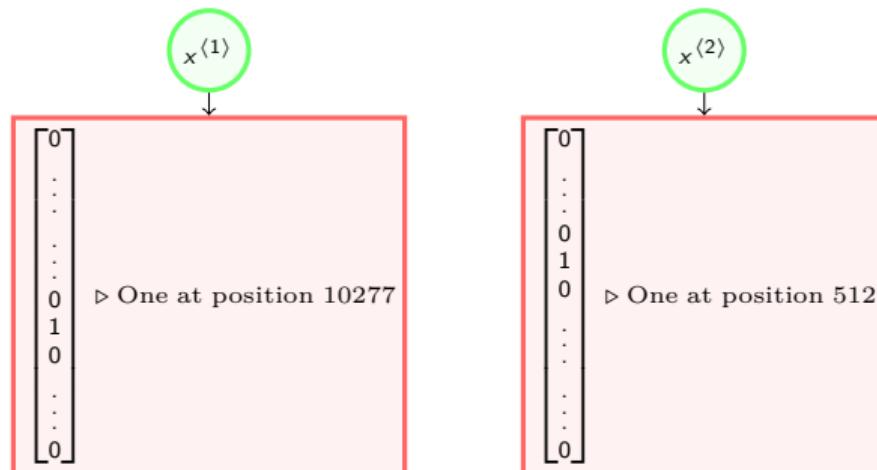
	My	experience	so	far	has	been	fantastic
	10277	512	12011	611	854	325	625
$x^{(i)} =$	$x^{(i)\langle 1 \rangle}$	$x^{(i)\langle 2 \rangle}$	$x^{(i)\langle 3 \rangle}$	$x^{(i)\langle 4 \rangle}$	$x^{(i)\langle 5 \rangle}$	$x^{(i)\langle 6 \rangle}$	$x^{(i)\langle 7 \rangle}$

Index	Word
1	a
2	ability
3	able
...	...
325	been
...	...
512	experience
...	...
611	far
...	...
625	fantastic
...	...
854	has
...	...
10277	My
...	...
12011	So
...	...

Vocabulary: Assume
20k Words

One-Hot Representation

	My	experience	so	far	has	been	fantastic
	10277	512	12011	611	854	325	625
$\mathbf{x}^{(i)} =$	$x^{(i)\langle 1 \rangle}$	$x^{(i)\langle 2 \rangle}$	$x^{(i)\langle 3 \rangle}$	$x^{(i)\langle 4 \rangle}$	$x^{(i)\langle 5 \rangle}$	$x^{(i)\langle 6 \rangle}$	$x^{(i)\langle 7 \rangle}$



$\mathbf{x}^{(i)}$ is a matrix with dimensions $20,000 \times 7$.

Index	Word
1	a
2	ability
3	able
...	...
325	been
...	...
512	experience
...	...
611	far
...	...
625	fantastic
...	...
854	has
...	...
10277	My
...	...
12011	So
...	...

Vocabulary: Assume
20k Words

Why do we need another model type?

Sentiment analysis:

“My experience so far has been fantastic” → *Positive*

“Your support team is useless” → *Negative*

$$\mathbf{x}^{(i)} = [x^{(1)}, \quad x^{(2)}, \quad x^{(3)}, \quad x^{(4)}, \quad x^{(5)}, \quad x^{(6)}, \quad x^{(7)}]$$

My experience so far has been fantastic
 $x^{(1)}$, $x^{(2)}$, $x^{(3)}$, $x^{(4)}$, $x^{(5)}$, $x^{(6)}$, $x^{(7)}$

We have now represented a sentence as a vector, Can we use FF-NN?

- Variable length vector.
- Context of data (elements) matters
 - You were *right*.
 - Make a *right* turn at the light.

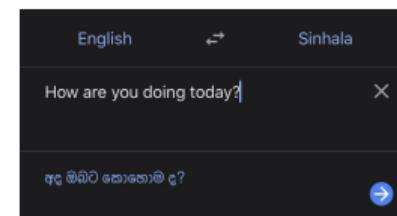
Sequential Data: Examples

Speech recognition:

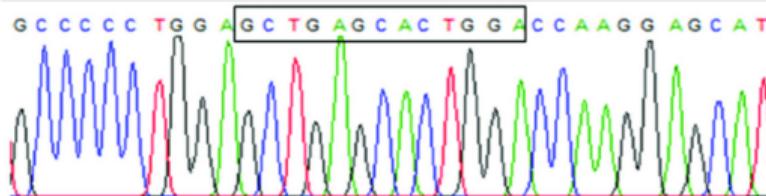


→ “How are you”

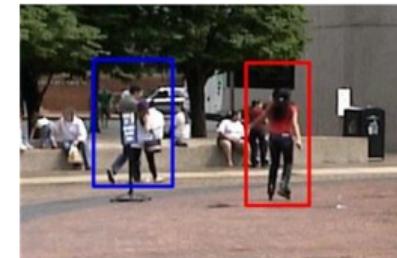
Machine translation:



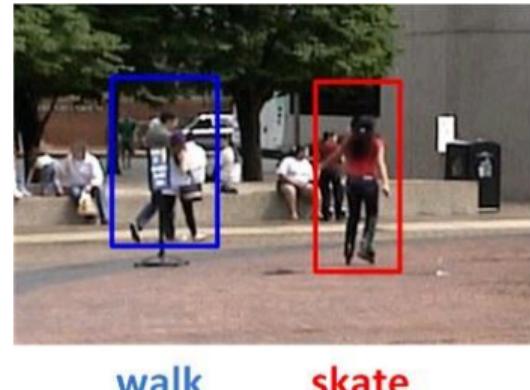
DNA sequence analysis:



Video Activity recognition:



- Data points with variable length .
- Order of the data matters. Order can be:
 - Time related: Video analysis
 - Not related to time: DNA sequence
- Shared feature across time is useful - context.



- Why is word based representation preferred over character based representation in NLP?
- For a simple NLP problem, what will be the one-hot representation for a word with index 3, If the vocabulary size is 10?
- What are the main properties of sequential data?

Outline

- 1 Sequential Data
- 2 Recurrent Neural Networks (RNN)
- 3 Variants of RNN
- 4 Gated Recurrent Units Networks
- 5 Long Short Term Memory networks
- 6 Bi-Directional and Deep RNN

“Automatically find information units like names, including person, organization and location names, and numeric expressions including time, date, money and percent expressions from unstructured text.”

Albert Einstein PER Albert Einstein was born in Ulm LOC in
Germany LOC on March 14, 1879. Six weeks later the family moved to
Munich LOC, where he later on began his schooling at the Luitpold
Gymnasium ORG. In 1896 he entered the Swiss Federal Polytechnic
School ORG in Zurich LOC to be trained as a teacher in physics and
mathematics.

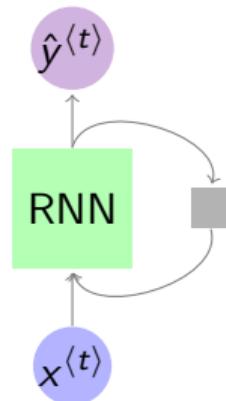
Simplified example: Only person names.

The	retired	English	test	cricketer	Mark	Butcher	is
$x^{(1)}$	$x^{(2)}$	$x^{(3)}$	$x^{(4)}$	$x^{(5)}$	$x^{(6)}$	$x^{(7)}$	$x^{(8)}$
0	0	0	0	0	1	1	0
$y^{(1)}$	$y^{(2)}$	$y^{(3)}$	$y^{(4)}$	$y^{(5)}$	$y^{(6)}$	$y^{(7)}$	$y^{(8)}$

Dictionary size: 20k

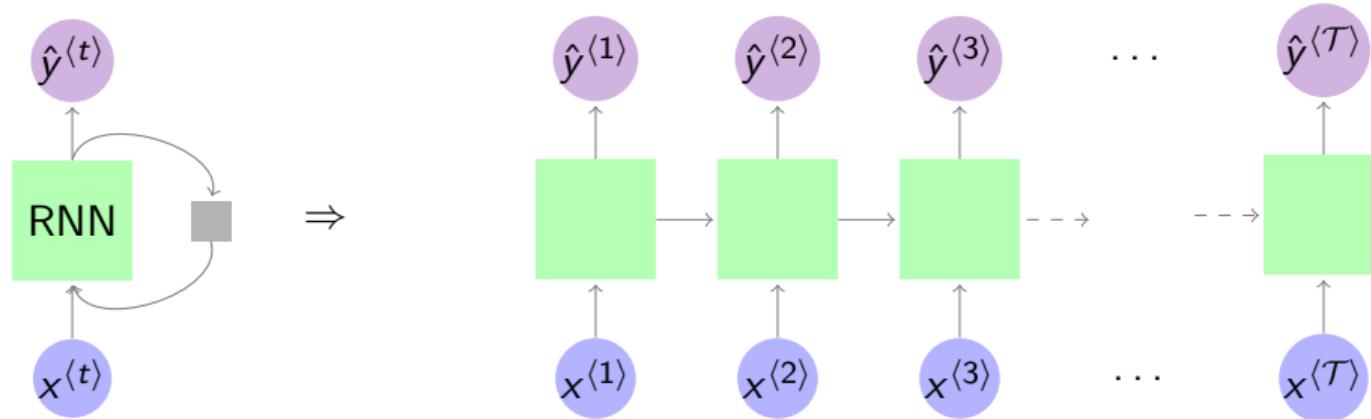
Simple solution: Represent each word as one-hot vector, predict name/not-name with FF-NN. Does not work for words like “Mark”, “Butcher”.

Recurrent neural networks have loops in them, allowing information to persist.



The	retired	English	Test	cricketer	Mark	Butcher	is
$x^{(1)}$	$x^{(2)}$	$x^{(3)}$	$x^{(4)}$	$x^{(5)}$	$x^{(6)}$	$x^{(7)}$	$x^{(8)}$
0	0	0	0	0	1	1	0

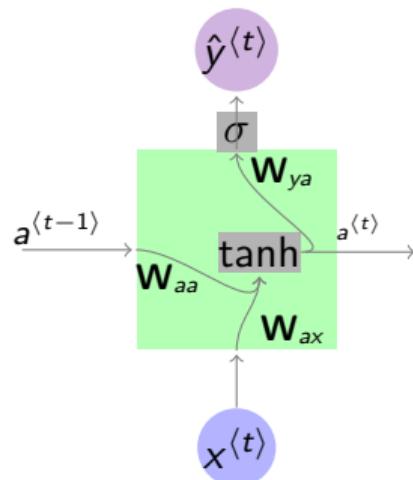
Recurrent neural networks have loops in them, allowing information to persist.



Unrolled Recurrent Neural Network

Same cell repeated. Not multiple cells.

Recurrent Neural Network: Cell



$$a^{(t)} = g_1 \left(W_{aa}a^{(t-1)} + W_{ax}x^{(t)} + b_a \right)$$

$$\hat{y}^{(t)} = g_2 \left(W_{ya}a^{(t)} + b_y \right)$$

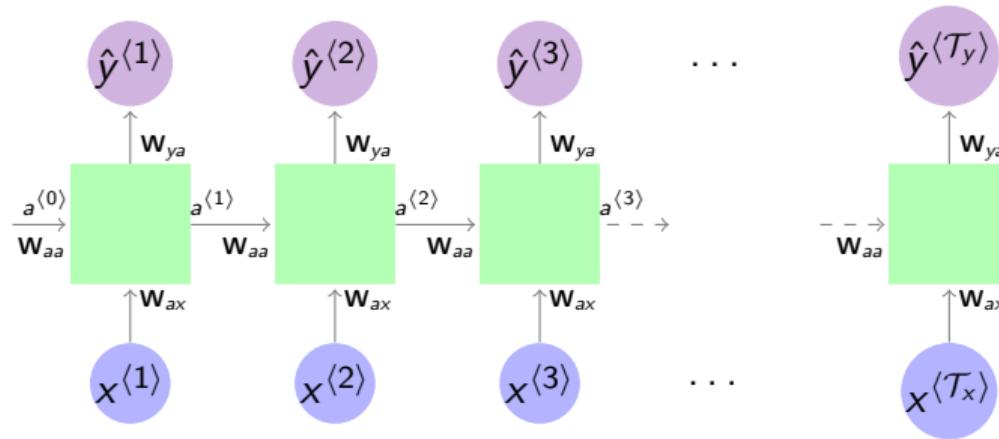
Assume:

- $x^{(t)}$ is 20000-dimensional (Vocab size)
- $a^{(t)}$ is 100-dimensional (user defined)
- $y^{(t)}$ is 1-dimensional (output dimension)

Then weight matrices (Learned):

- W_{ax} is 100×20000
- W_{aa} is 100×100
- W_{ya} is 1×100
- b_a is 100×1
- b_y is 1×1

Recurrent Neural Networks: Forward Propagation



All W_{aa} , W_{ax} , W_{ya} are shared across RNN cells

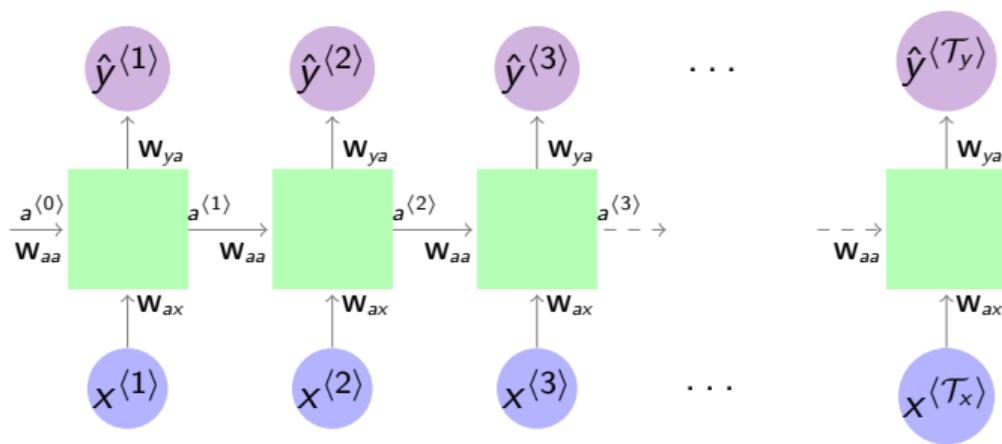
For a data point (i) :

- T_x Number of elements in the input
- T_y Number of elements in the output
- $a^{(0)}$ Initial state can be all zero vector.

$$a^{(t)} = g_1 \left(W_{aa}a^{(t-1)} + W_{ax}x^{(t)} + b_a \right)$$

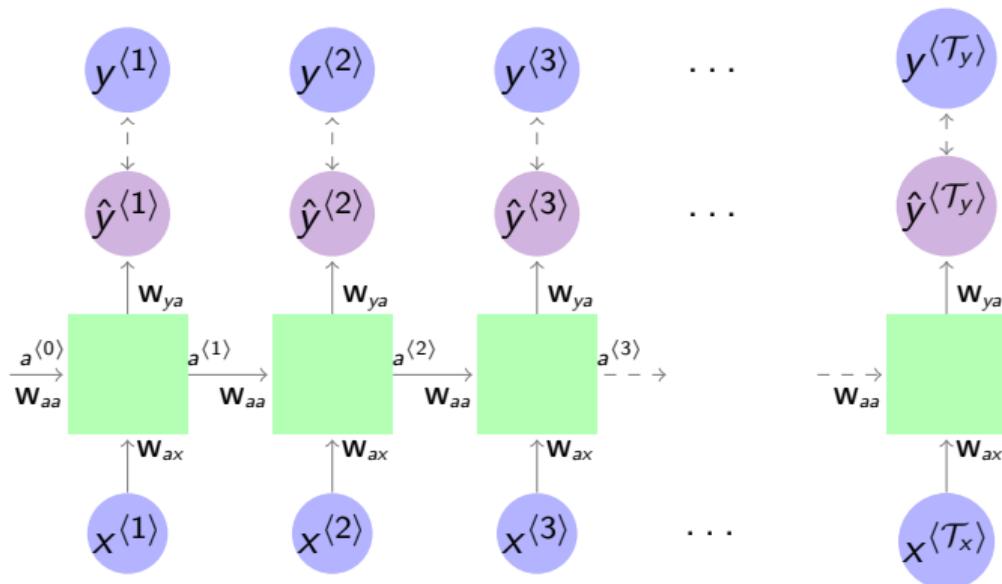
$$\hat{y}^{(t)} = g_2 \left(W_{ya}a^{(t)} + b_y \right)$$

Recurrent Neural Network: Back-propagation



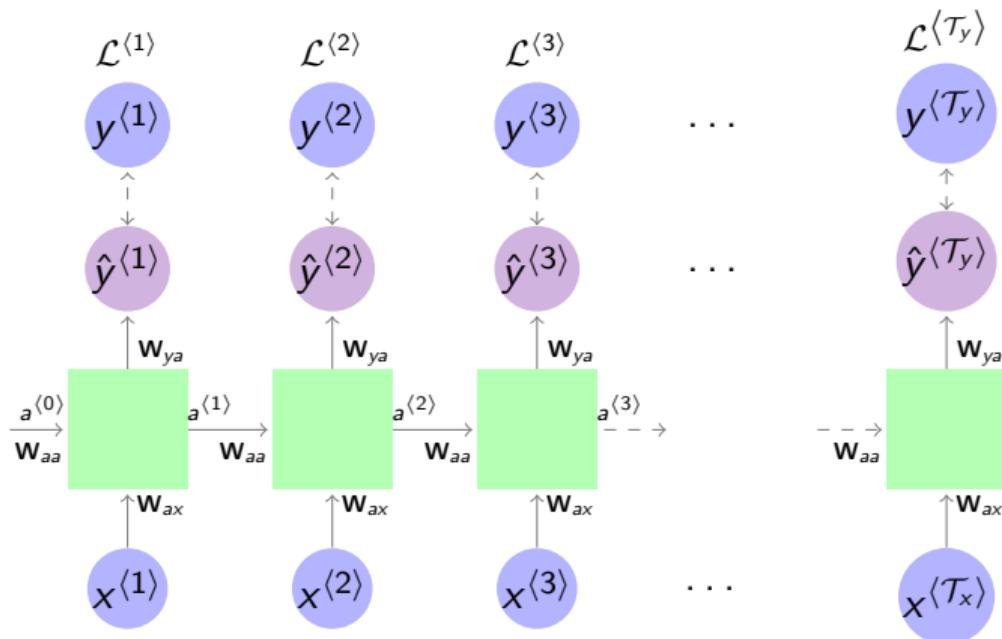
All W_{aa} , W_{ax} , W_{ya} are shared across RNN cells

Recurrent Neural Network: Back-propagation



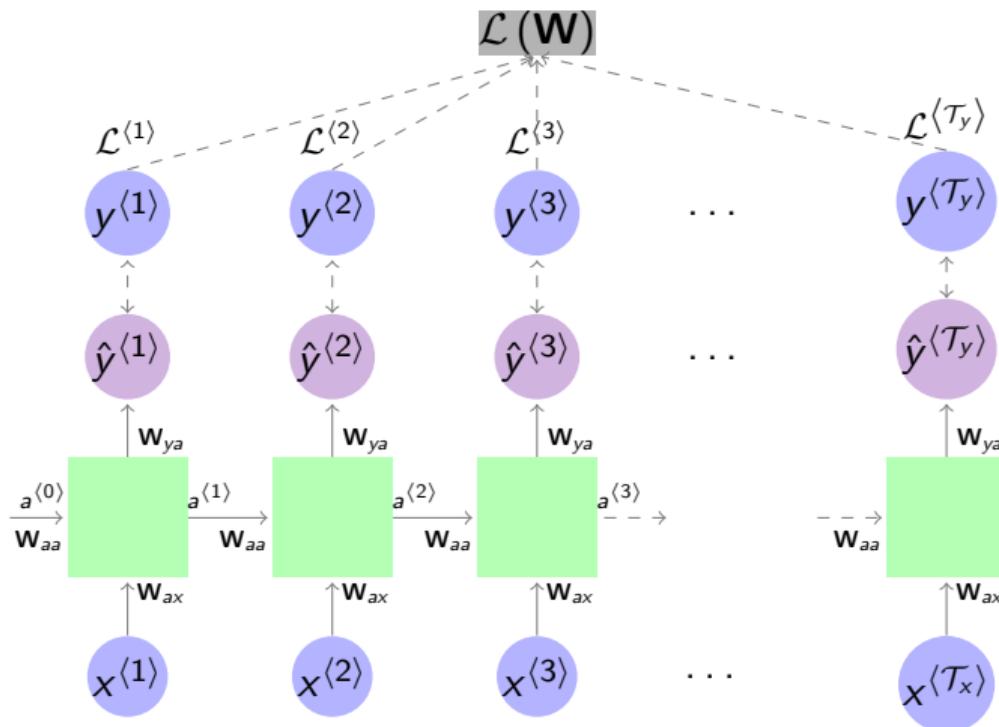
All $\mathbf{W}_{aa}, \mathbf{W}_{ax}, \mathbf{W}_{ya}$ are shared across RNN cells

Recurrent Neural Network: Back-propagation



All $\mathbf{W}_{aa}, \mathbf{W}_{ax}, \mathbf{W}_{ya}$ are shared across RNN cells

Recurrent Neural Network: Back-propagation



$$\mathcal{L}(\mathbf{W}) = \sum_{t=1}^{\mathcal{T}_y} \mathcal{L}^{(t)}(\mathbf{W})$$

All W_{aa}, W_{ax}, W_{ya} are shared across RNN cells

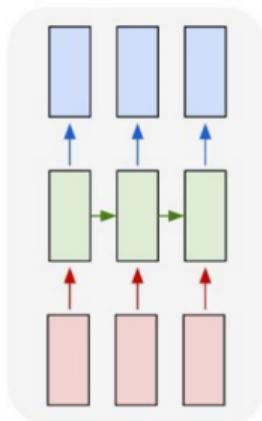
- What values are usually used for $a^{(0)}$?
- What will be an appropriate function for output activation (g_2) of a RNN if the task is regression?

Outline

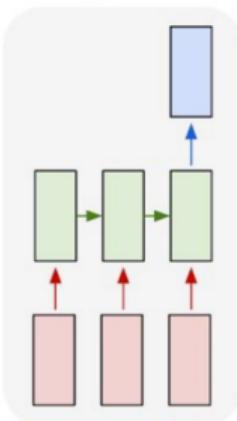
- 1 Sequential Data
- 2 Recurrent Neural Networks (RNN)
- 3 Variants of RNN
- 4 Gated Recurrent Units Networks
- 5 Long Short Term Memory networks
- 6 Bi-Directional and Deep RNN

Variants of the Simple RNN Structure

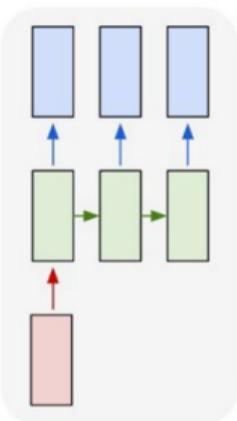
There are several variants of the basic RNN depending on the shapes of inputs and outputs.



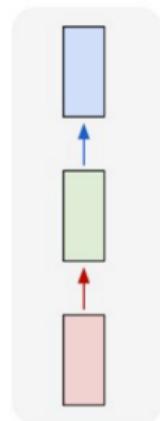
Many-to-Many



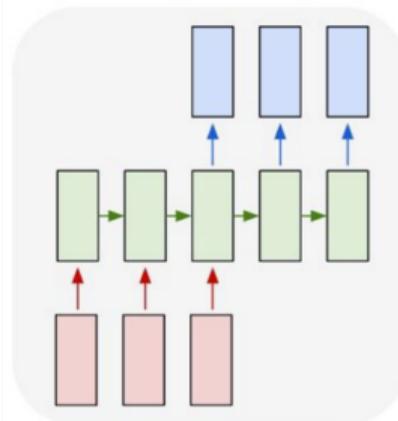
Many-to-One



One-to-Many



One-to-One



Many-to-Many $T_y \neq T_x$

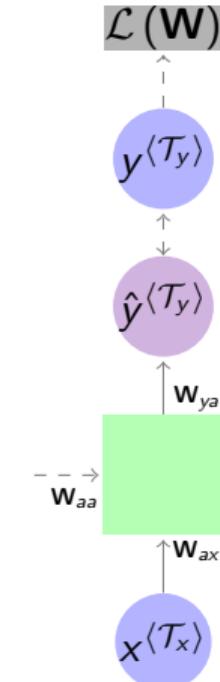
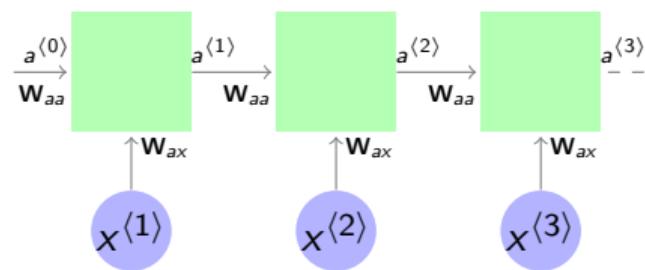
Image: The Unreasonable Effectiveness of Recurrent Neural Networks - Andrej Karpathy

Many-to-Many ($T_y = T_x$): We have already discussed this - e.g. Named Entity Recognition.

Sentiment analysis:

- “My experience so far has been fantastic” → *Positive*
- “Your support team is useless” → *Negative*

Video Action Classification:



All W_{aa}, W_{ax}, W_{ya} are shared across RNN cells

Generating Text/Music.

E.g Generating text similar to Shakespeare's writing.

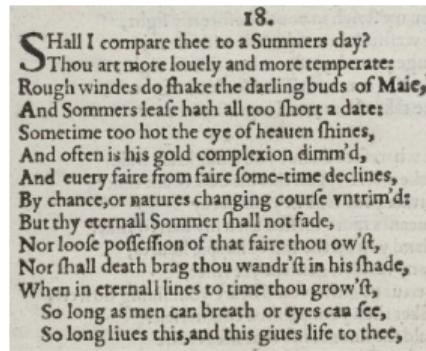


Image: Sonnet 18 in the 1609 Quarto of Shakespeare's sonnets.

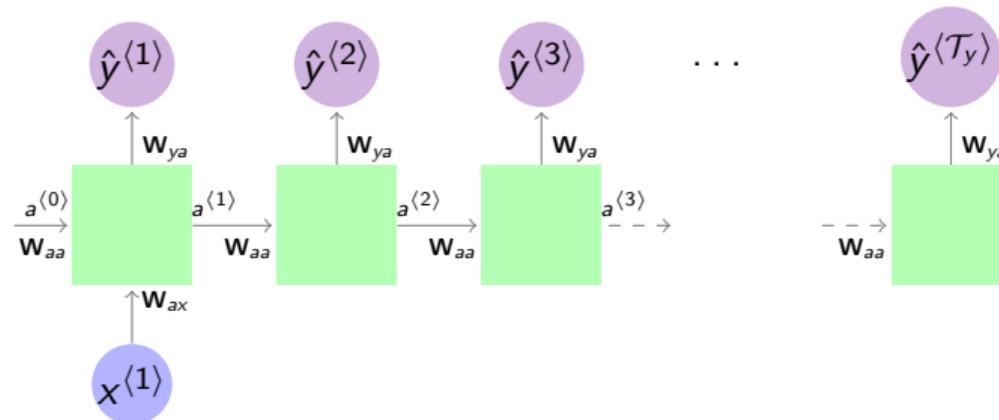
Given some text from Shakespeare's writing generate novel sentences that look similar.

One-to-Many

Generating Shakespeare's writing.

$x^{(t)}$ is a one-hot with size equal to number of characters.

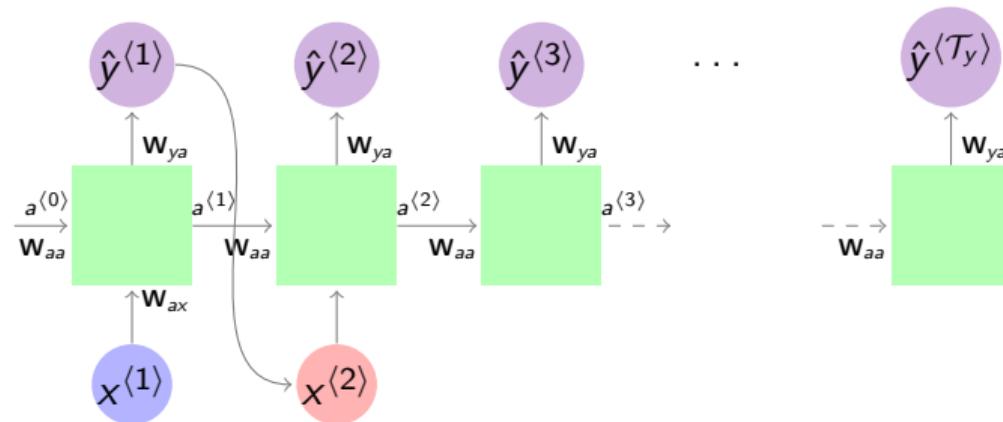
$\hat{y}^{(t)}$ is a soft-max-out with size equal to number of characters.



All W_{aa} , W_{ax} , W_{ya} are shared across RNN cells

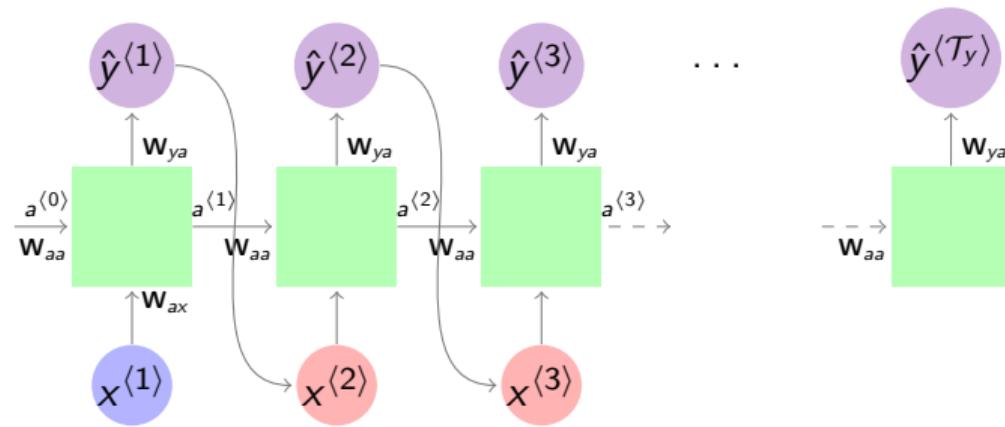
One-to-Many

Inference: Convert $\hat{y}^{(t)}$ to one-hot by sampling and input as $x^{(t+1)}$



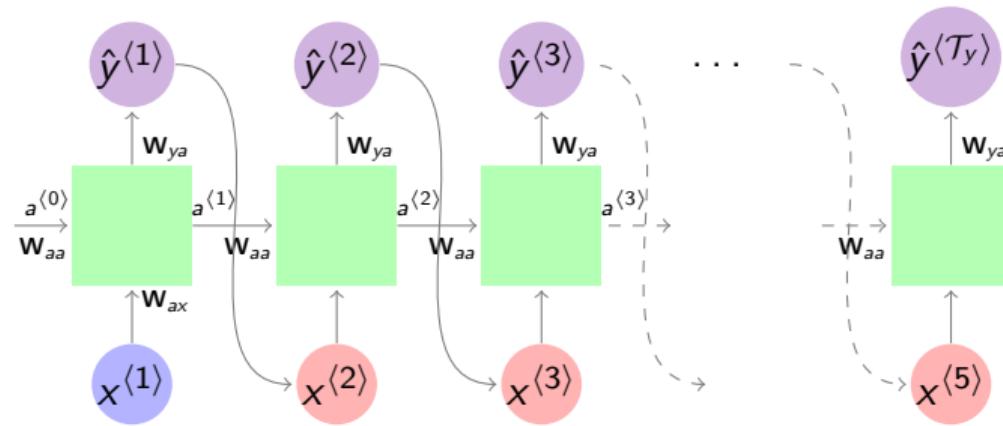
All W_{aa} , W_{ax} , W_{ya} are shared across RNN cells

One-to-Many



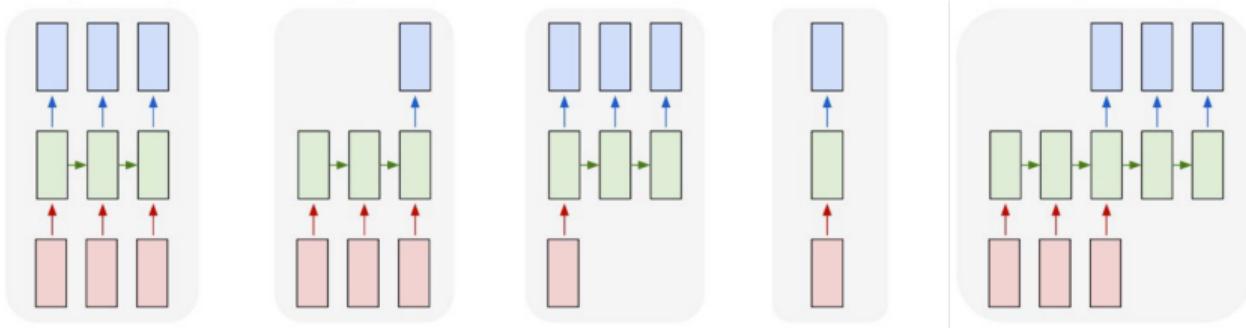
All W_{aa}, W_{ax}, W_{ya} are shared across RNN cells

One-to-Many



All W_{aa} , W_{ax} , W_{ya} are shared across RNN cells

Many-to-Many $\mathcal{T}_y \neq \mathcal{T}_x$



Many-to-Many Many-to-One One-to-Many One-to-One Many-to-Many $\mathcal{T}_y \neq \mathcal{T}_x$

There are any situations where X, Y does not have one to one relationship.

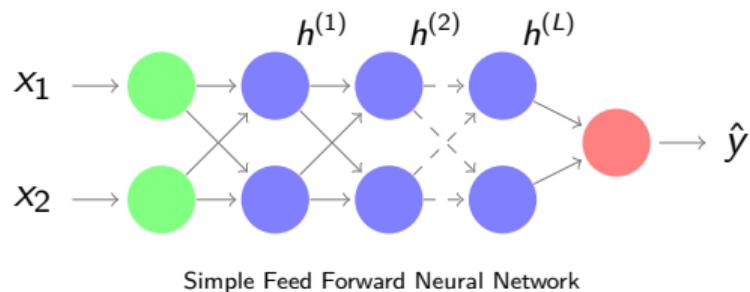
E.g French to English translation, video captioning.

- The sequence lengths are not the same.
- Word to word translation does not work.

- You are given the task of predicting the genre of short music clips (not fixed length). What type of RNN is suitable for this task? (many-to-many, one-to-many, many-to-one, many-to-many).
- You are given the task of generating image captions. What type of RNN is suitable for this task?
- You are given the task of designing an automated exam marking system. How would RNN help you in this task?

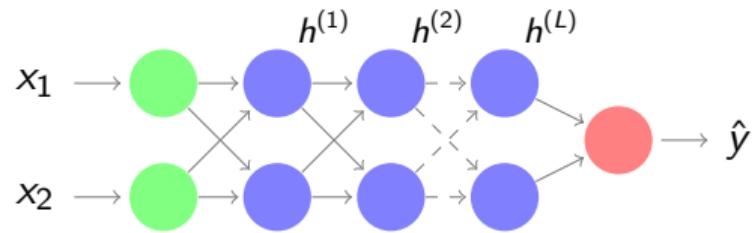
Outline

- 1 Sequential Data
- 2 Recurrent Neural Networks (RNN)
- 3 Variants of RNN
- 4 Gated Recurrent Units Networks
- 5 Long Short Term Memory networks
- 6 Bi-Directional and Deep RNN

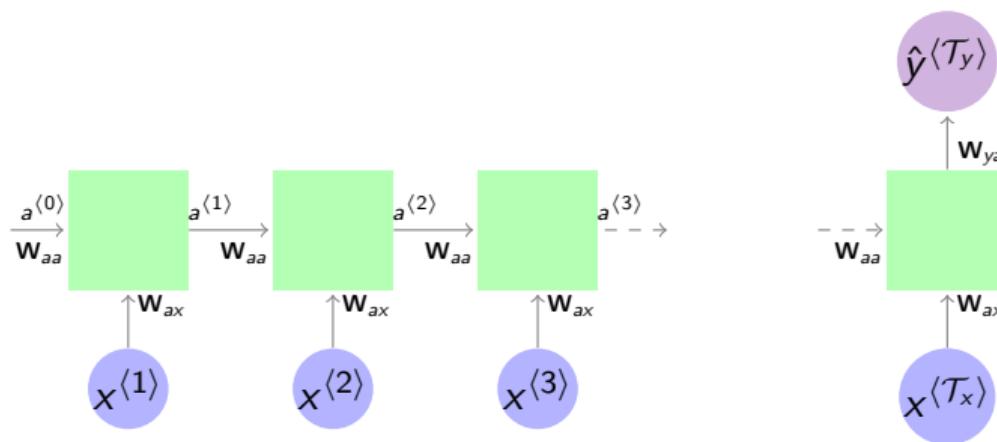


- Vanishing gradient: use Skip connections, BN, ReLU, etc.
- Exploding gradient: use gradient clipping.

Vanishing Gradients

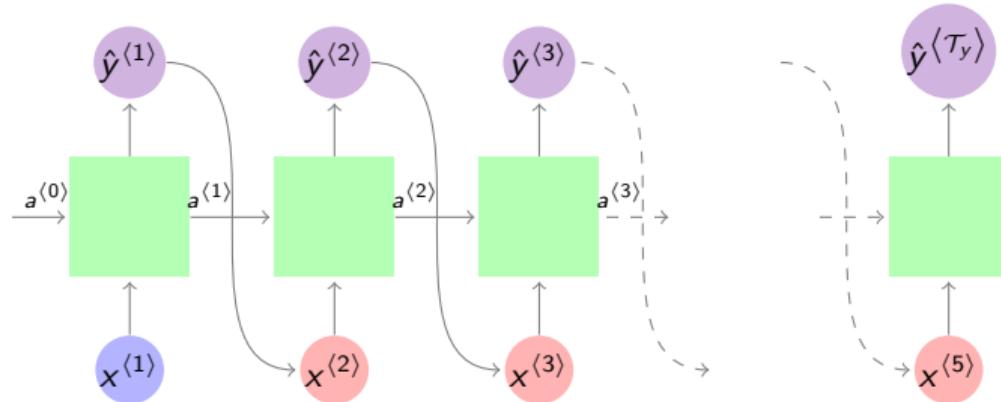


Simple Feed Forward Neural Network

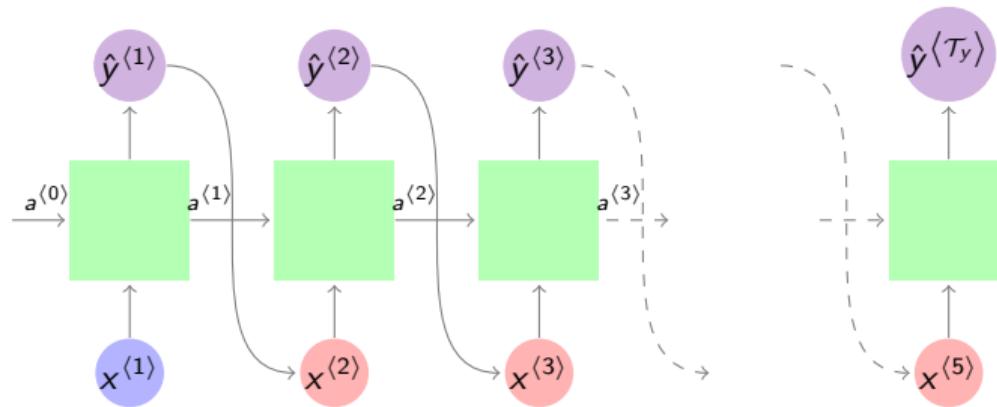


- Vanishing gradient: use Skip connections, BN, ReLU, etc.
- Exploding gradient: use gradient clipping.

Long Range Dependencies



The *cat, who ate . . . , was/were full.*



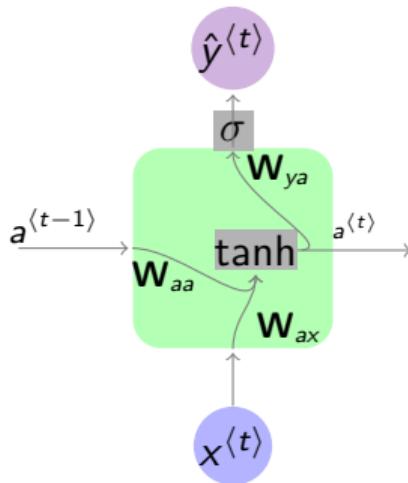
The *cat*, who ate ..., was full.

The *cats*, who ate ..., were full.

In theory, RNNs are absolutely capable of handling such “long-term dependencies”. In practice, RNNs don’t seem to be able to learn them.

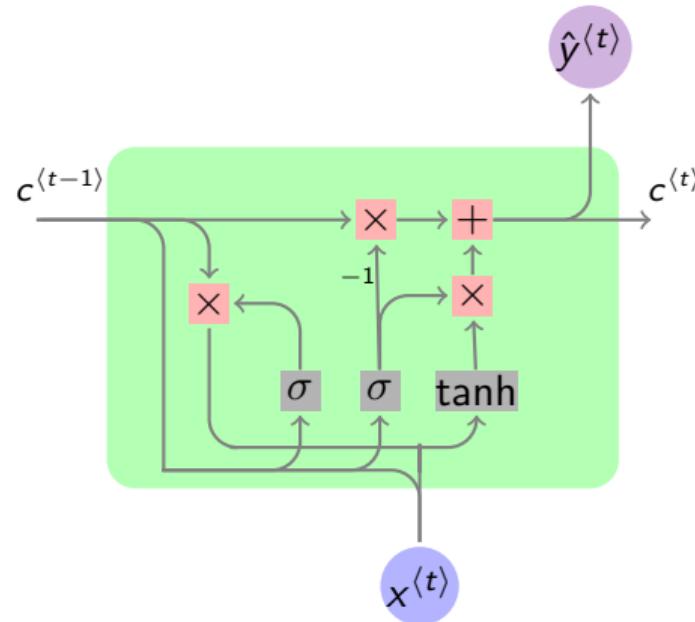
Gated Recurrent Units (GRU)

Change the structure of the units



$$a^{(t)} = g_1 (W_{aa}a^{(t-1)} + W_{ax}x^{(t)} + b_a)$$

$$\hat{y}^{(t)} = g_2 (W_{ya}a^{(t)} + b_y)$$

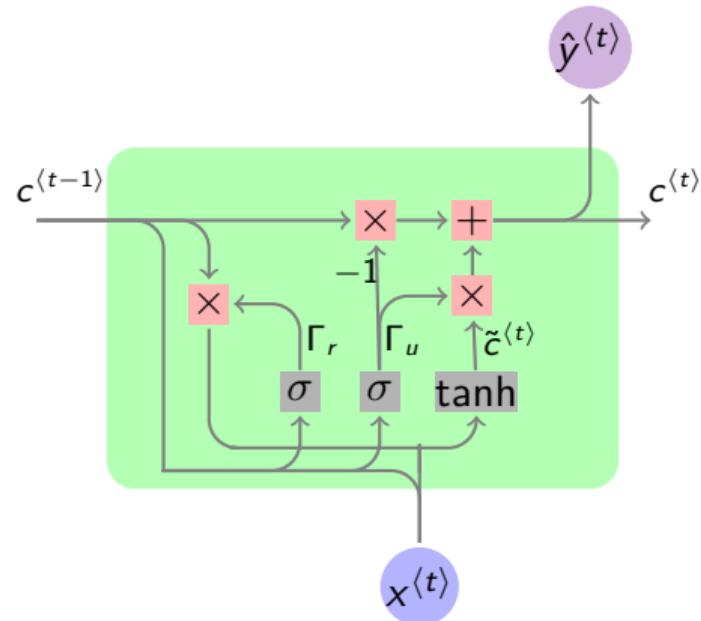


On the Properties of Neural Machine Translation: Encoder-Decoder Approaches
 Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling

Gated Recurrent Units (GRU)

$$\tilde{c}^{(t)} = \tanh(\mathbf{W}_c [c^{(t-1)}, x^{(t)}] + b_c) \quad \triangleright \text{Cell State}$$

$$c^{(t)} = \tilde{c}^{(t)}$$



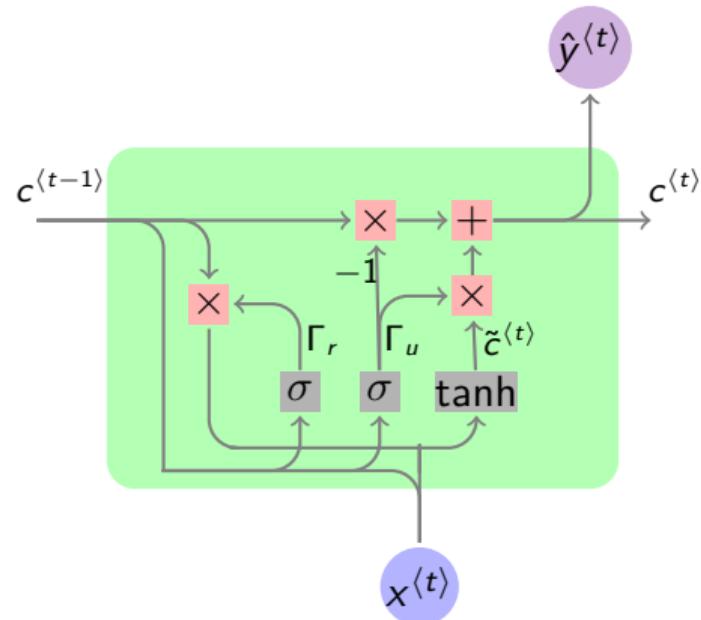
On the Properties of Neural Machine Translation: Encoder-Decoder Approaches
 Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling

Gated Recurrent Units (GRU)

$$\tilde{c}^{(t)} = \tanh(\mathbf{W}_c [c^{(t-1)}, x^{(t)}] + b_c) \quad \triangleright \text{Cell State}$$

$$c^{(t)} = (1 - \Gamma_u) \odot c^{(t-1)} + \Gamma_u \odot \tilde{c}^{(t)}$$

$$\Gamma_u = \sigma(\mathbf{W}_u [c^{(t-1)}, x^{(t)}] + b_u) \quad \triangleright \text{Update gate}$$



On the Properties of Neural Machine Translation: Encoder-Decoder Approaches
 Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling

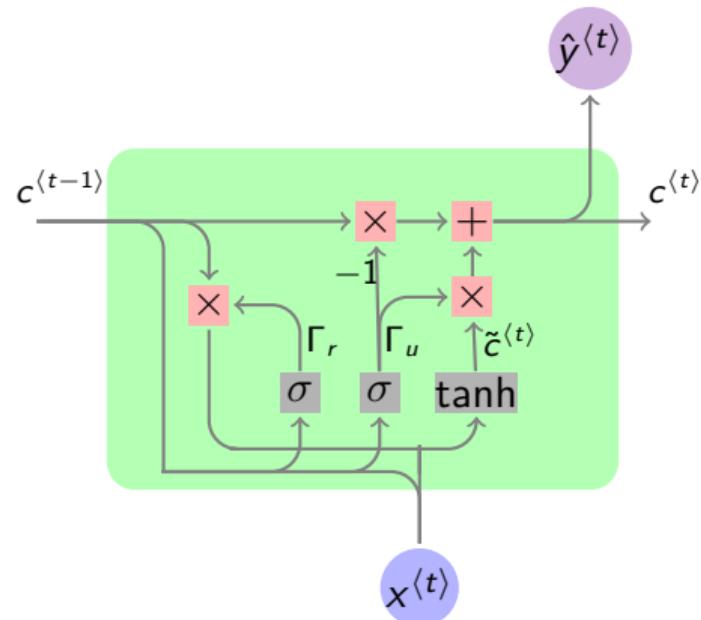
Gated Recurrent Units (GRU)

$$\tilde{c}^{(t)} = \tanh(\mathbf{W}_c [\Gamma_r \odot c^{(t-1)}, x^{(t)}] + b_c) \quad \triangleright \text{Cell State}$$

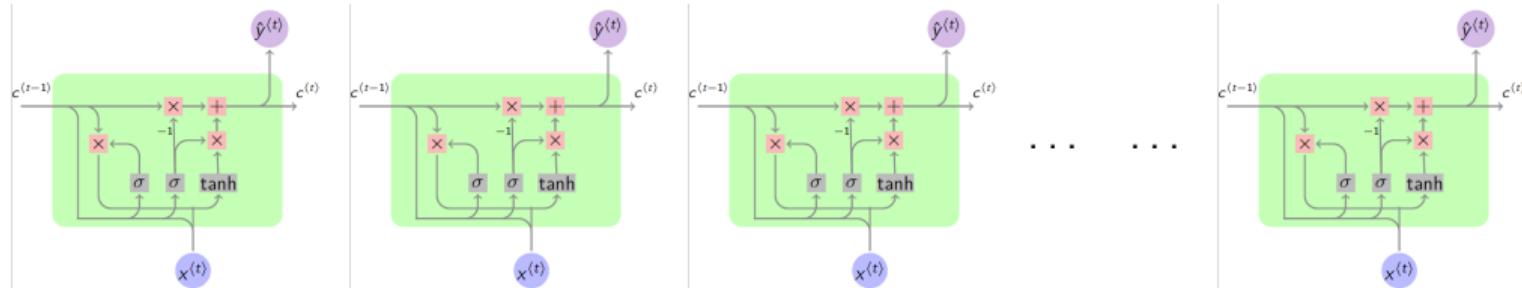
$$c^{(t)} = (1 - \Gamma_u) \odot c^{(t-1)} + \Gamma_u \odot \tilde{c}^{(t)}$$

$$\Gamma_u = \sigma(\mathbf{W}_u [c^{(t-1)}, x^{(t)}] + b_u) \quad \triangleright \text{Update gate}$$

$$\Gamma_r = \sigma(\mathbf{W}_r [c^{(t-1)}, x^{(t)}] + b_r) \quad \triangleright \text{Relevance gate}$$



On the Properties of Neural Machine Translation: Encoder-Decoder Approaches
 Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling

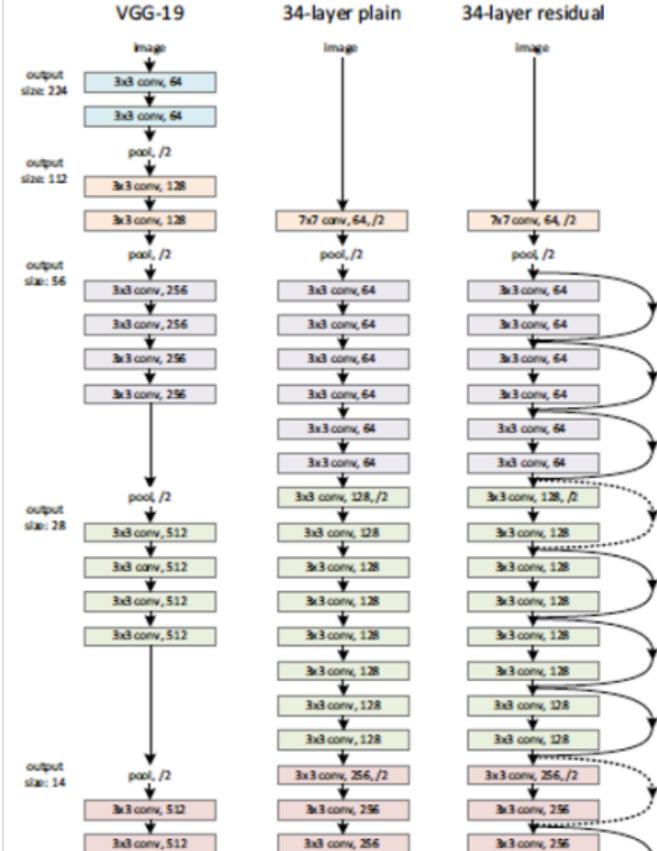


Unrolled GRU network is identical to simple-RNN except for the cell type.

Note the **horizontal line** running through the top of the diagram. like a conveyor belt, it runs straight down the entire chain, with only some minor linear interactions. It is very easy for information to just flow along it unchanged (recall ResNet and skip connections).

Gates are a way to optionally let information through.

GRU Network



Outline

- 1 Sequential Data
- 2 Recurrent Neural Networks (RNN)
- 3 Variants of RNN
- 4 Gated Recurrent Units Networks
- 5 Long Short Term Memory networks
- 6 Bi-Directional and Deep RNN

Long Short Term Memory (LSTM) Cell

$$\Gamma_u = \sigma (\mathbf{W}_u [a^{(t-1)}, x^{(t)}] + b_u) \quad \triangleright \text{Update gate}$$

$$\Gamma_f = \sigma (\mathbf{W}_f [a^{(t-1)}, x^{(t)}] + b_f) \quad \triangleright \text{Forget gate}$$

$$\Gamma_o = \sigma (\mathbf{W}_o [a^{(t-1)}, x^{(t)}] + b_o) \quad \triangleright \text{Output gate}$$

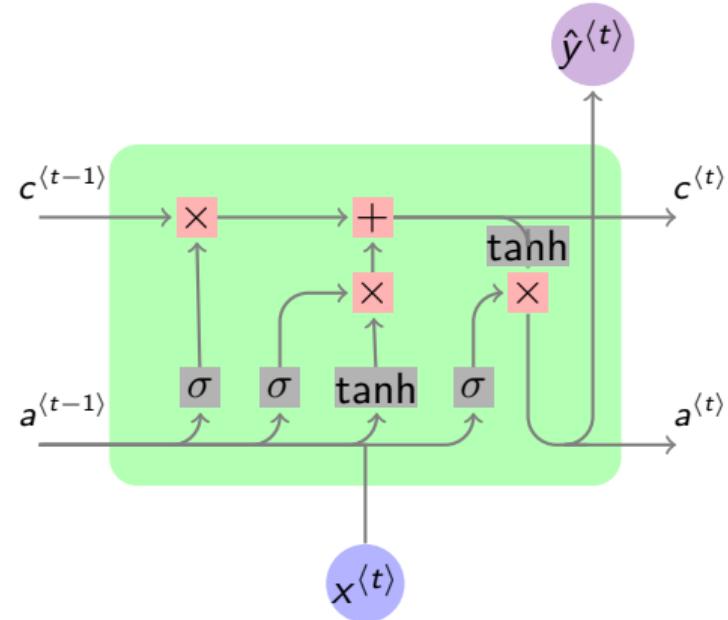
$$\tilde{c}^{(t)} = \tanh (\mathbf{W}_c [a^{(t-1)}, x^{(t)}] + b_c) \quad \triangleright \text{Cell State}$$

$$c^{(t)} = \Gamma_f \odot c^{(t-1)} + \Gamma_u \odot \tilde{c}^{(t)}$$

$$a^{(t)} = \Gamma_o \odot \tanh (c^{(t)})$$

Uses two gates (Γ_u, Γ_f) to make the update compared to single Γ_u in GRU.

Long-short term memory



LSTM was developed first, GRU is a simplification of the LSTM cell.

How to decide GRU vs. LSTM:

- No universally applicable rule to select one over other for a given application.
- GRU: simpler and allow much larger models.
- LSTM: More powerful and flexible - Much older than GRU and historically proven.

Either GRU or LSTM will enable the model to capture long range dependencies.

Outline

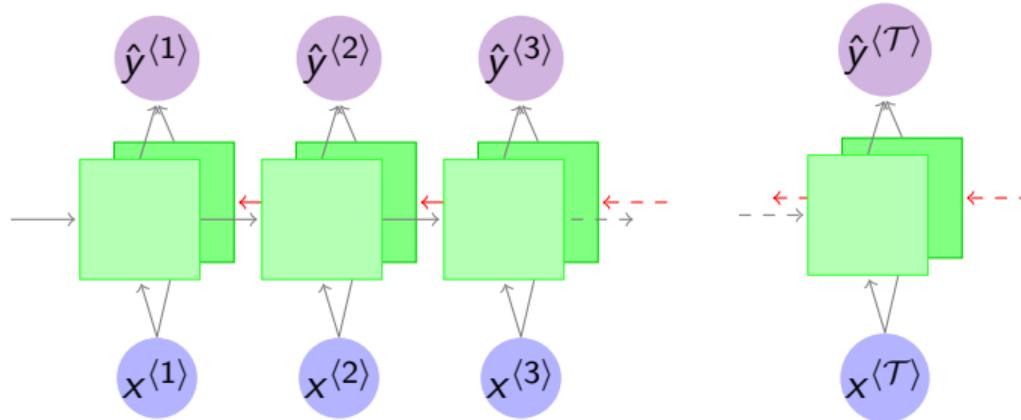
- 1 Sequential Data
- 2 Recurrent Neural Networks (RNN)
- 3 Variants of RNN
- 4 Gated Recurrent Units Networks
- 5 Long Short Term Memory networks
- 6 Bi-Directional and Deep RNN

RNNs can only use information from the past to make predictions at time t.

In some cases, past information is not enough to make prediction. Future information can be helpful.

Mark	Butcher	is	a	retired	English	Test	cricketer
$x^{(1)}$	$x^{(2)}$	$x^{(3)}$	$x^{(4)}$	$x^{(5)}$	$x^{(6)}$	$x^{(7)}$	$x^{(8)}$
1	1	0	0	0	0	0	0

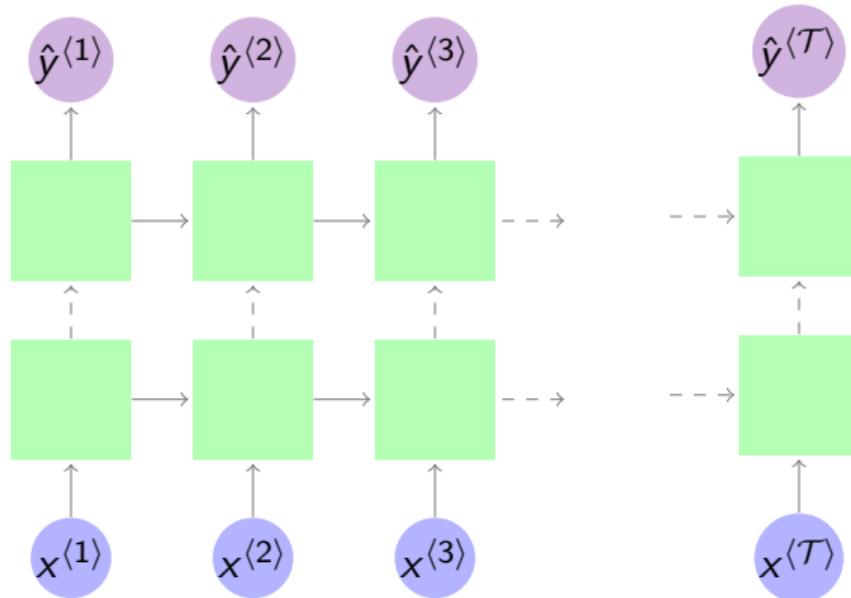
“Mark my word . . . ”



Unrolled Bi-Directional Recurrent Neural Network

Two RNN. One goes from first word to last, the other goes from last to first.

Not suitable for real time systems as the future is not available for such systems.



Unrolled Recurrent Neural Network

Stack RNN cells on top of each other.

Can be expensive in terms of memory. Not as many layers as CNN.

Summary

- Sequential data: variable length, order matters, context is important.
- Recurrent neural networks have loops in them, allowing information to persist.
- Modifications to simple RNN:
 - GRU/LSTM: capture Long range dependencies.
 - Bi-Directional: Use historical and future information.
 - Deep: More complex models.

Next week: Applications of sequential data modelling

Lab: Getting started with RNN - Sentiment analysis