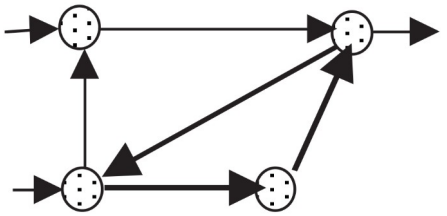
Recurrent neural networks

Vlad Gladkikh

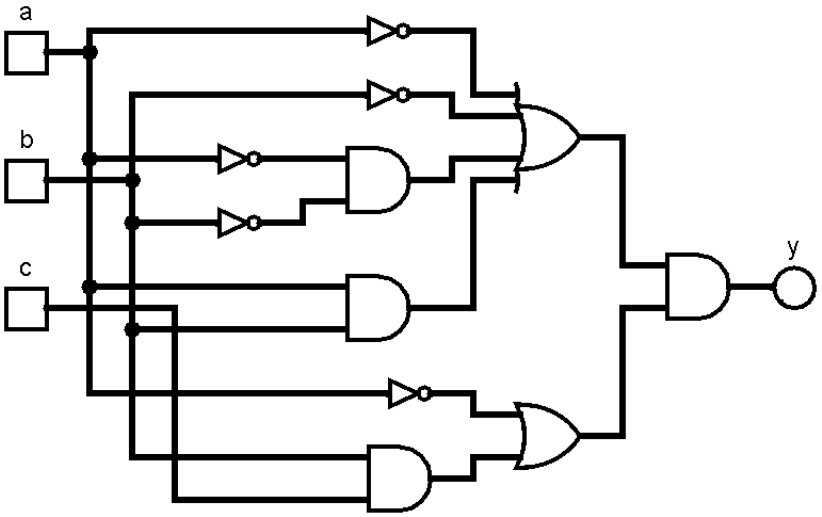
IBS CMCM

Recurrent neural networks are networks that contains cycles.



Similar concepts in digital electronics:

Two types of digital circuits



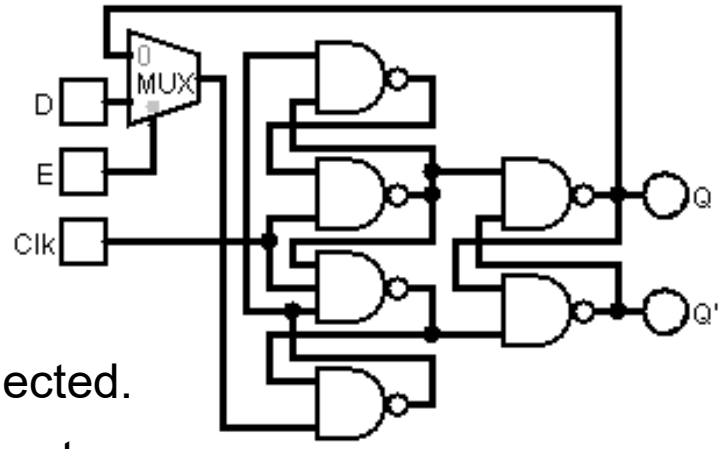
Combinational circuits: the outputs of the circuit depend only on the current inputs.

– do not need to know the history of past inputs

– do not require any memory elements

Sequential circuits: outputs depend on not only the current inputs, but also on all of the past inputs.

– must contain memory elements in order to remember the history of past input values



Both combinational and sequential circuits consist of the same building blocks – the AND, OR, and NOT gates, or just NAND gates.

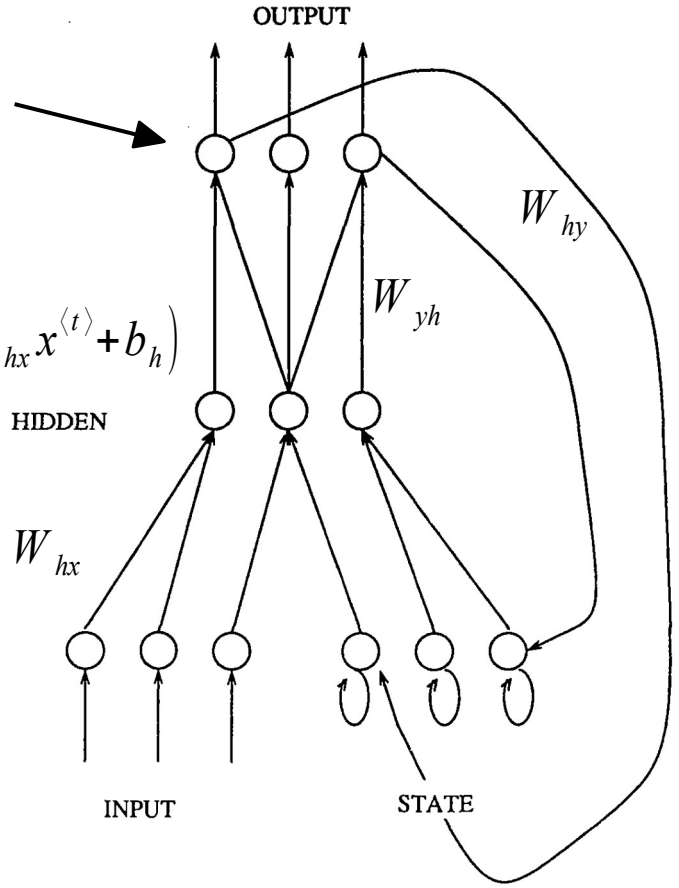
What makes them different is in the way the gates are connected.

Sequential circuits have backward connections: output → input

Jordan network

$$\hat{y}^{\langle t \rangle} = \sigma_y\left(W_{yh} h^{\langle t \rangle} + b_y\right)$$

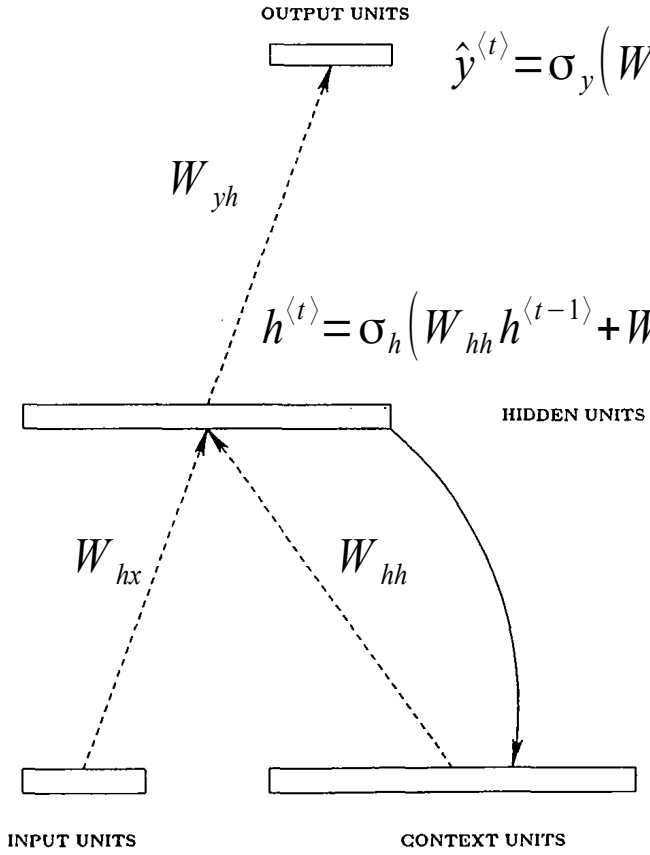
$$h^{\langle t \rangle} = \sigma_h\left(W_{hy} y^{\langle t-1 \rangle} + W_{hx} x^{\langle t \rangle} + b_h\right)$$



Elman network

$$\hat{y}^{\langle t \rangle} = \sigma_y\left(W_{yh} h^{\langle t \rangle} + b_y\right)$$

$$h^{\langle t \rangle} = \sigma_h\left(W_{hh} h^{\langle t-1 \rangle} + W_{hx} x^{\langle t \rangle} + b_h\right)$$

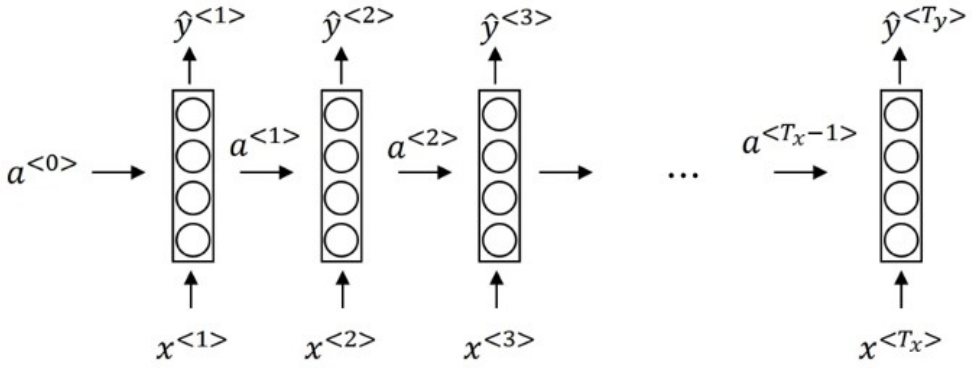


Dotted lines: trainable connections.

Elman, Jeffrey L. (1990). "Finding Structure in Time". Cognitive Science. 14 (2): 179–211. doi:10.1016/0364-0213(90)90002-E

https://datascience.stackexchange.com/questions/82416/difference-between-jordan-elman-and-normal-rnn

https://ai.stackexchange.com/questions/8190/where-can-i-find-the-original-paper-that-introduced-rnns/



$y^{(i)\langle t\rangle}$ – element t of output sample i

$i = 1, \ldots \langle T_y \rangle$

$x^{(i)\langle t\rangle}$ – element t of input sample i

$i = 1, \ldots \langle T_x \rangle$

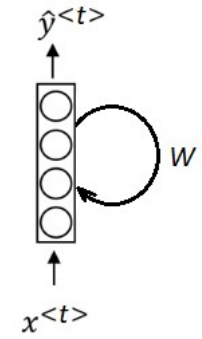
time

$$a^{\langle t\rangle} = g_a\left(W_{aa}\, a^{\langle t-1\rangle} + W_{ax}\, x^{\langle t\rangle} + b_a\right) = g_a\left(W_a\,[a^{\langle t-1\rangle}, x^{\langle t\rangle}] + b_a\right)$$

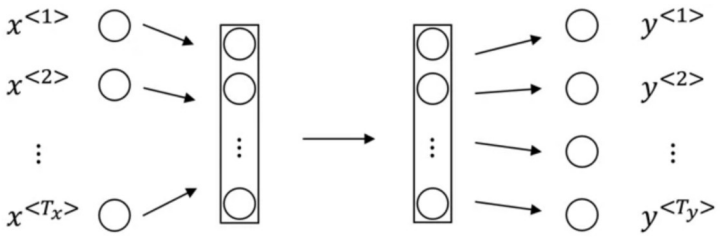
$$W_a = [W_{aa}, W_{ax}]$$

$$\hat{y}^{\langle t\rangle} = g_y\left(W_{ya}\, a^{\langle t\rangle} + b_y\right) = g_y\left(W_y\, a^{\langle t\rangle} + b_y\right)$$

unfolded in time



Why not a feed-forward NN?



– Inputs and outputs can be different lengths in different examples

– No sharing of features learned across different positions in the sequence

$[0\,1\,1\,1\,0\,0\,0\,0\,0]$ $[1\,0\,0]$

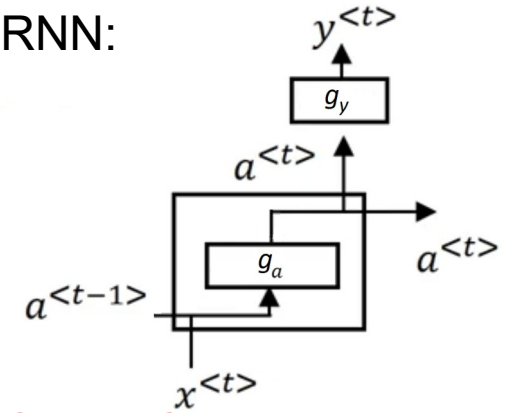
$[0\,0\,0\,1\,1\,1\,0\,0\,0]$ $[0\,1\,0]$

$[0\,0\,1]$

Keras TensorFlow

https://www.coursera.org/learn/nlp-sequence-models/

RNN:



GRU and LSTM are RNN that can learn over long sequences.

https://www.coursera.org/learn/nlp-sequence-models/

$$a^{\langle t\rangle}=g_a\left(W_a[a^{\langle t-1\rangle},x^{\langle t\rangle}]+b_a\right)$$
$$\hat{y}^{\langle t\rangle}=g_y\left(W_y a^{\langle t\rangle}+b_y\right)$$

GRU - Gated Recurrent Unit

$$\tilde{c}^{\langle t\rangle}=g_c\left(W_c[\Gamma_r\circ c^{\langle t-1\rangle},x^{\langle t\rangle}]+b_c\right)$$
$$\Gamma_r=g_r\left(W_r[c^{\langle t-1\rangle},x^{\langle t\rangle}]+b_r\right)$$
$$\Gamma_u=g_u\left(W_u[c^{\langle t-1\rangle},x^{\langle t\rangle}]+b_u\right)$$
$$c^{\langle t\rangle}=\Gamma_u\circ\tilde{c}^{\langle t\rangle}+(1-\Gamma_u)\circ c^{\langle t-1\rangle}$$
$$a^{\langle t\rangle}=c^{\langle t\rangle}$$
$$\hat{y}^{\langle t\rangle}=g_y\left(W_y a^{\langle t\rangle}+b_y\right)$$

c - cell

r - relevance

u - update

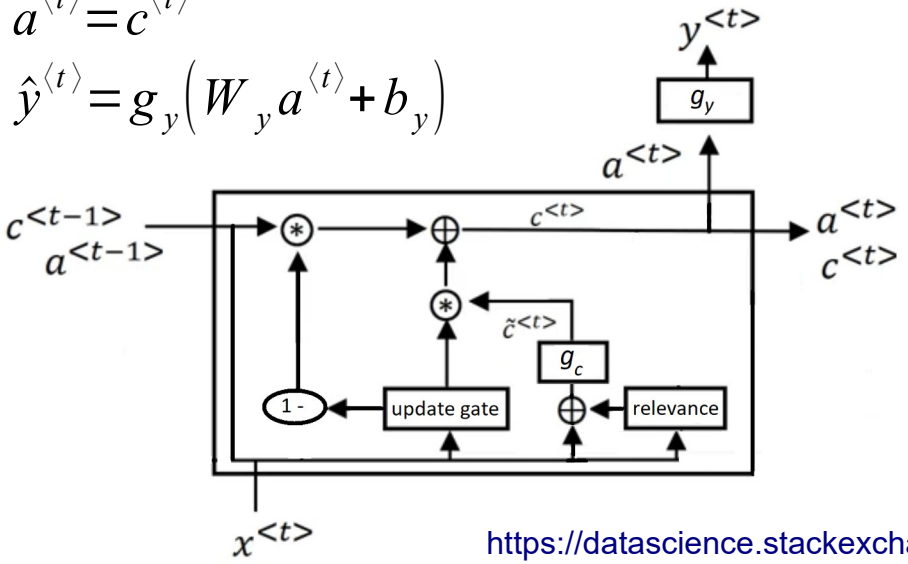
f - forget

o - output

○ - element-wise product

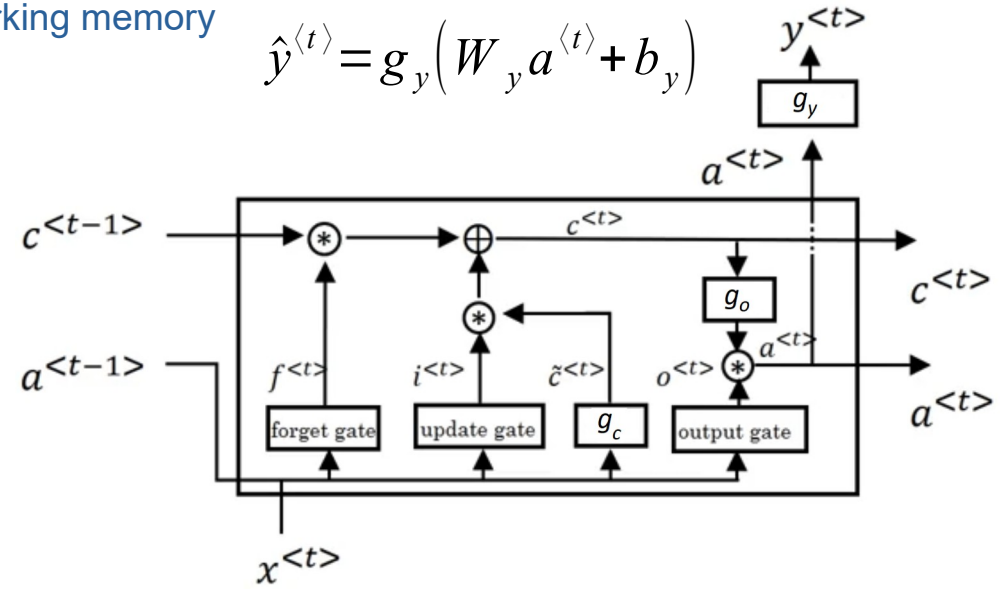
Cell state: Long term memory

Hidden state: Working memory

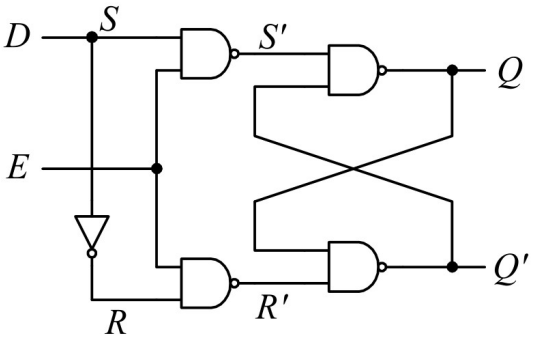


LSTM - Long Short-Term Memory

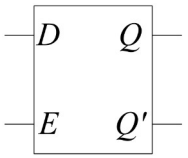
$$\tilde{c}^{\langle t\rangle}=g_c\left(W_c[a^{\langle t-1\rangle},x^{\langle t\rangle}]+b_c\right)$$
$$\Gamma_u=g_u\left(W_u[a^{\langle t-1\rangle},x^{\langle t\rangle}]+b_u\right)$$
$$\Gamma_f=g_f\left(W_f[a^{\langle t-1\rangle},x^{\langle t\rangle}]+b_f\right)$$
$$\Gamma_o=g_o\left(W_o[a^{\langle t-1\rangle},x^{\langle t\rangle}]+b_o\right)$$
$$c^{\langle t\rangle}=\Gamma_u\circ\tilde{c}^{\langle t\rangle}+\Gamma_f\circ c^{\langle t-1\rangle}$$
$$a^{\langle t\rangle}=\Gamma_o\circ g_o\left(c^{\langle t\rangle}\right)$$
$$\hat{y}^{\langle t\rangle}=g_y\left(W_y a^{\langle t\rangle}+b_y\right)$$

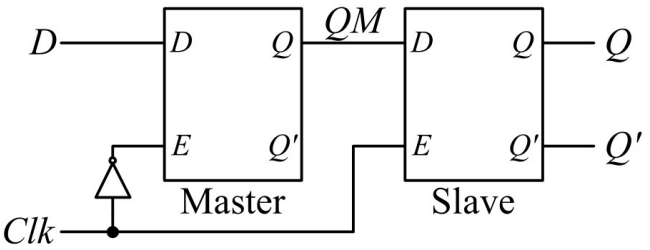


https://datascience.stackexchange.com/questions/82808/difference-between-cell-state-and-hidden-state

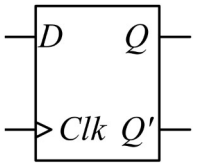


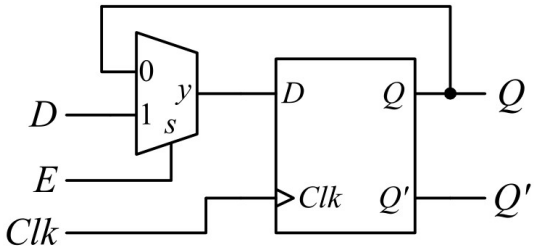
D Latch



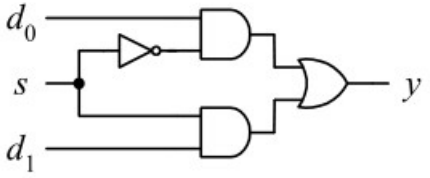


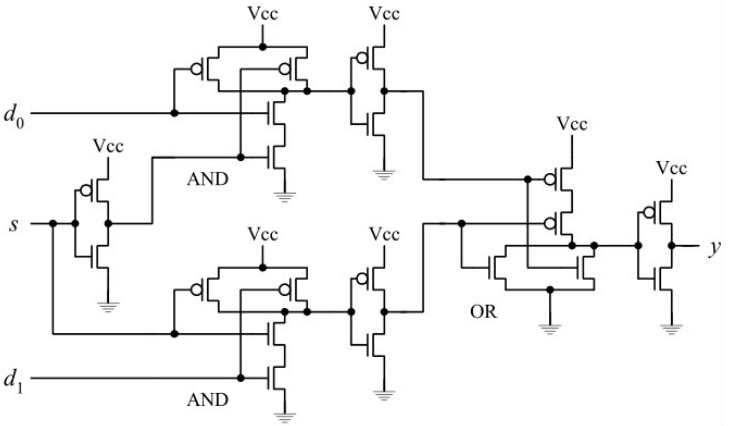
D Flip-Flop

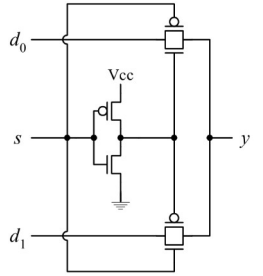




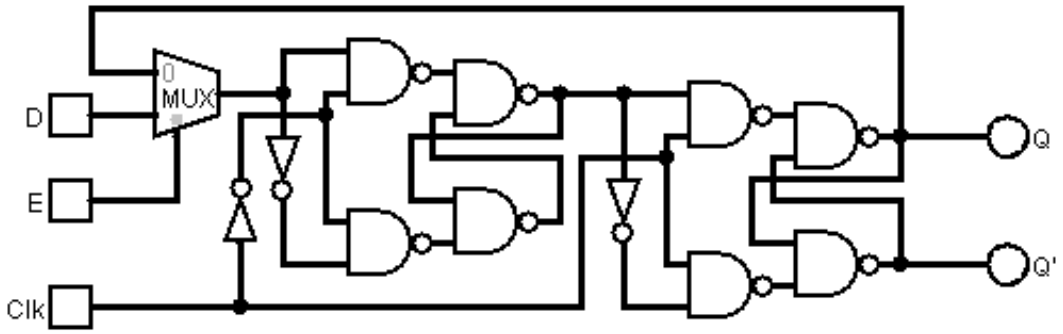
Naive implementation

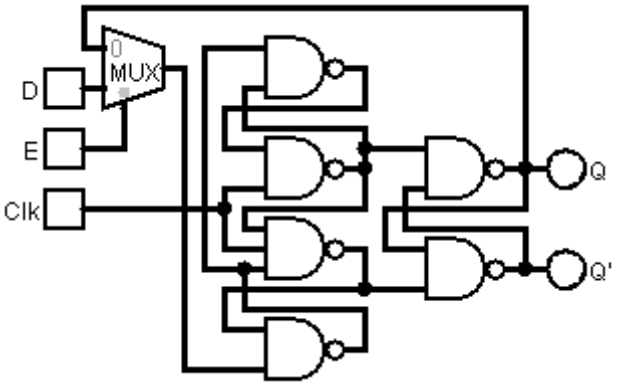






A smarter implementation





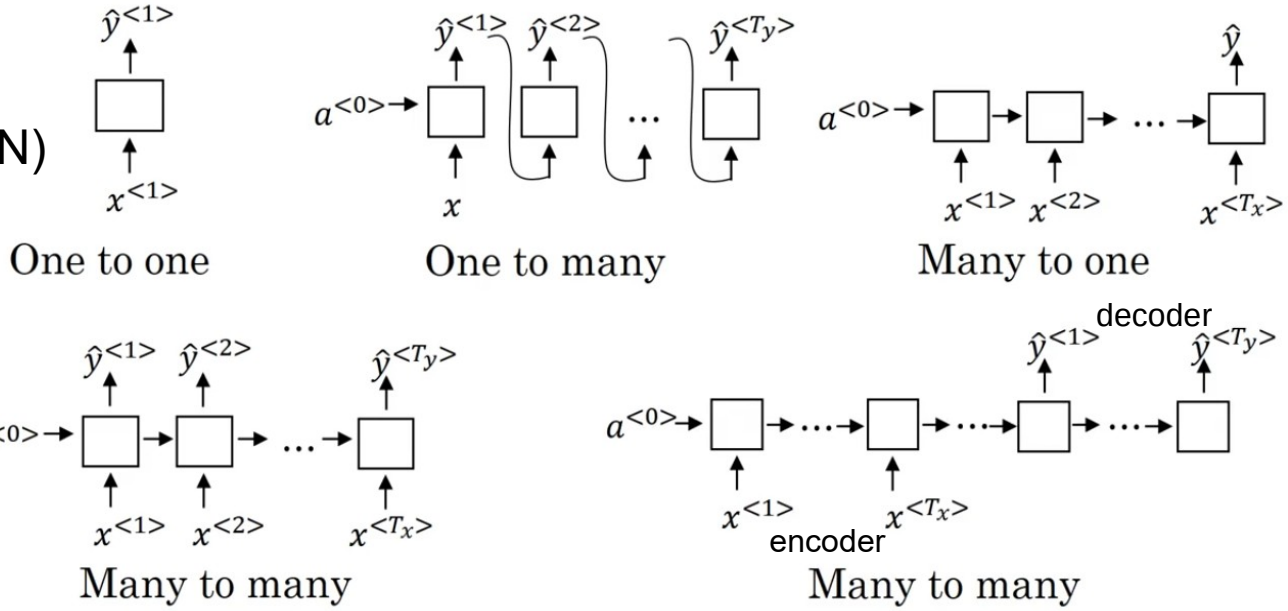
RNN can operate over sequences of vectors

https://www.coursera.org/learn/nlp-sequence-models/

Sequences can be in the input, the output, or both

No constraints on the lengths sequences since the recurrent transformation is fixed and can be applied many times.

(FF-NN)



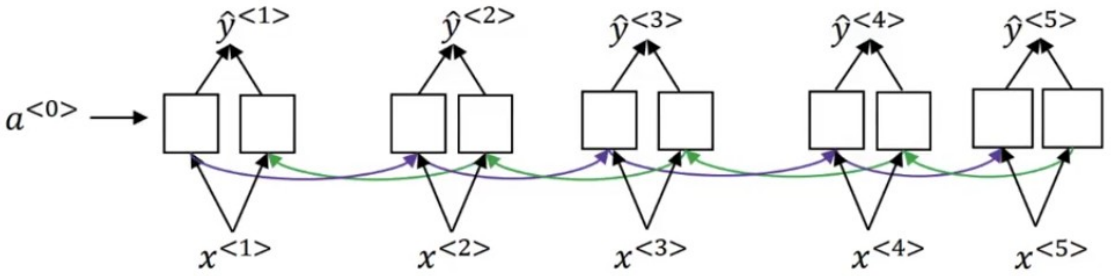
Implicit assumptions (wrong for some problems):

1) Only the preceding elements are relevant for the decision

2) The transformative relationship between the input, output and the passed-on state is the same for all elements in the chain, and doesn't change over time

https://ai.stackexchange.com/questions/16226/why-use-a-recurrent-neural-network-over-a-feedforward-neural-network-for-sequenc
https://stats.stackexchange.com/questions/514611/do-recurrent-neural-networks-assume-stationarity-or-just-a-general-kind-of-seque

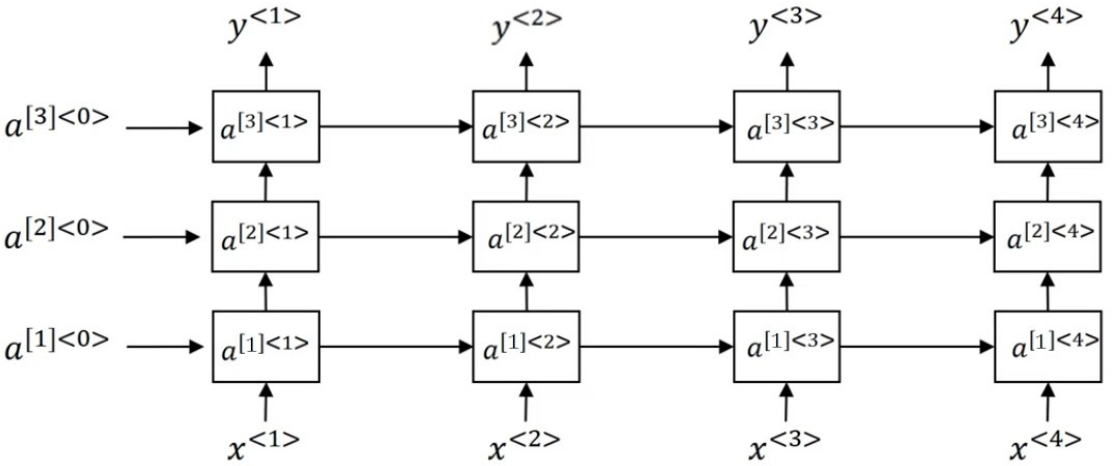
Bidirectional RNN



He said, "Teddy Roosevelt was a great President."

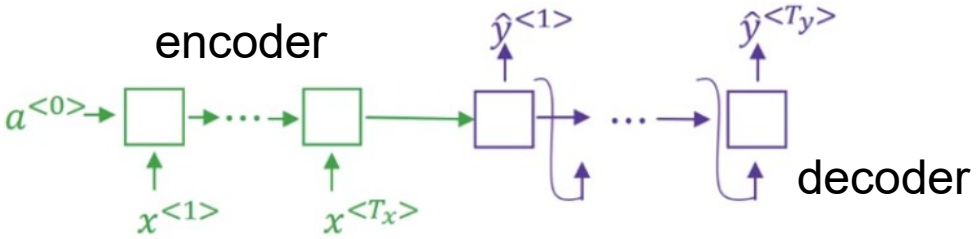
He said, "Teddy bears are on sale!"

Deep RNN



https://www.coursera.org/learn/nlp-sequence-models/

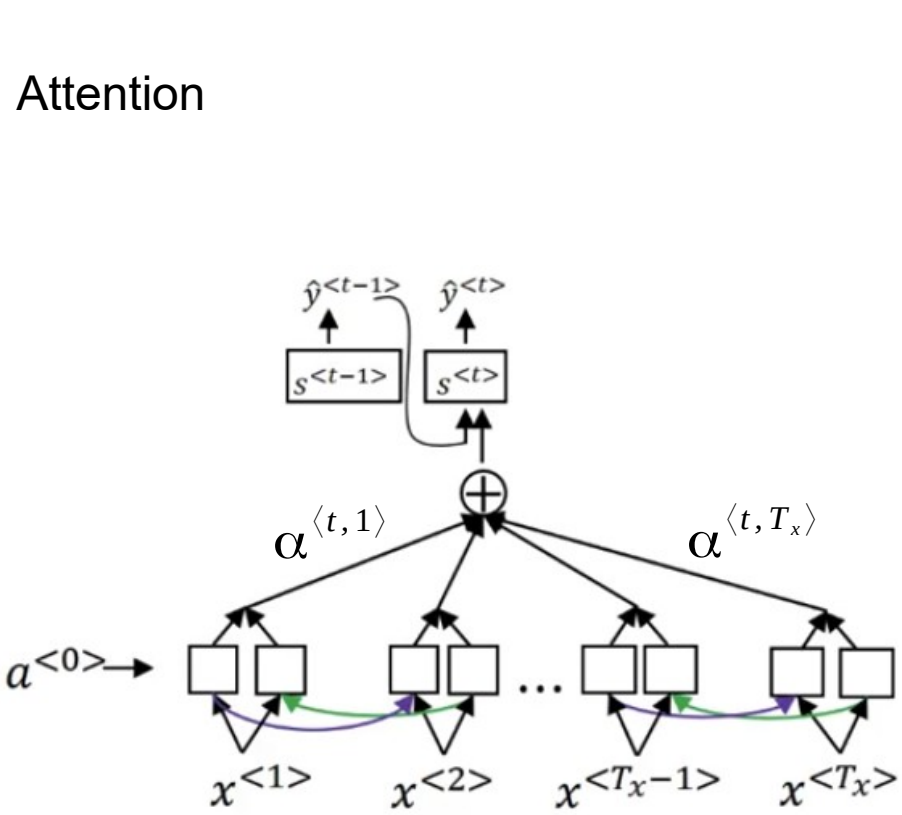
Sequence to sequence

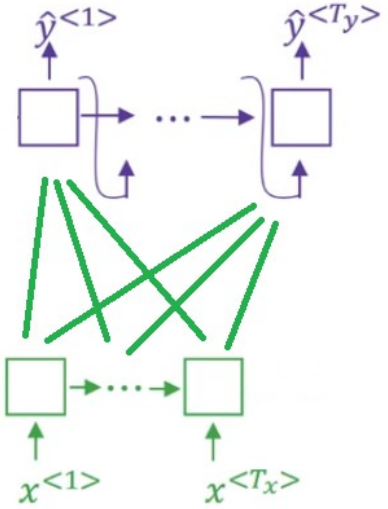


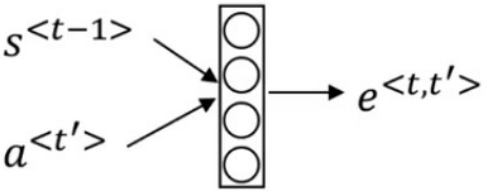
Jane s'est rendue en Afrique en septembre dernier, a apprécié la culture et a rencontré beaucoup de gens merveilleux; elle est revenue en parlant comment son voyage était merveilleux, et elle me tente d'y aller aussi.

Jane went to Africa last September, and enjoyed the culture and met many wonderful people; she came back raving about how wonderful her trip was, and is tempting me to go too.

Attention





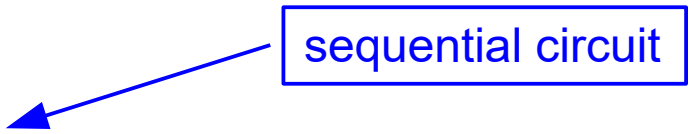


$$\alpha^{<t,t'>} = \frac{\exp(e^{<t,t'>})}{\sum_{t'=1}^{T_x} \exp(e^{<t,t'>})}$$

$\alpha^{\langle t,t' \rangle}$ – amount of attention $y^{\langle t \rangle}$ should pay to $a^{\langle t' \rangle}$

https://www.coursera.org/learn/nlp-sequence-models/

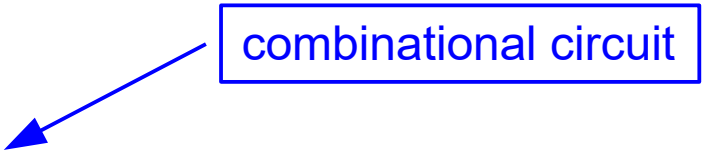
Parts of sequential circuits



State memory

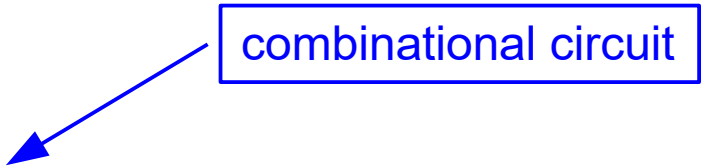
– keeps the complete history of inputs up to the current time.

The output of the state memory at one particular instance of time is called the **state** of the system at that time.



Output logic circuit

– generates the output of the whole circuit based on the current state, and may or may not be dependent on the current inputs.



Next-state logic circuit

– responsible for determining what next state to go to based on the current state that the system is in (i.e., the past inputs), and the current inputs

---

A sequential circuit operates by transitioning from one state to the next, generating different output signals.

Word representation

https://www.coursera.org/learn/nlp-sequence-models/

V = [a, aaron, ..., zulu, <UNK>]

1-hot representation

Man (5391)	Woman (9853)	King (4914)	Queen (7157)	Apple (456)	Orange (6257)
$\begin{bmatrix}0\\0\\0\\0\\\vdots\\1\\\vdots\\0\\0\end{bmatrix}$	$\begin{bmatrix}0\\0\\0\\0\\\vdots\\1\\\vdots\\0\end{bmatrix}$	$\begin{bmatrix}0\\0\\0\\\vdots\\1\\\vdots\\0\\0\\0\end{bmatrix}$	$\begin{bmatrix}0\\0\\0\\0\\\vdots\\1\\\vdots\\0\end{bmatrix}$	$\begin{bmatrix}0\\\vdots\\1\\\vdots\\0\\0\\0\\0\end{bmatrix}$	$\begin{bmatrix}0\\0\\0\\0\\\vdots\\1\\\vdots\\0\end{bmatrix}$

No inherent relationship to one another

Similar words are likely to have similar vectors

Featurized representation: word embedding

	Man (5391)	Woman (9853)	King (4914)	Queen (7157)	Apple (456)	Orange (6257)
Gender	-1	1	-0.95	0.97	0.00	0.01
300 Royal	0.01	0.02	0.93	0.95	-0.01	0.00
Age	0.03	0.02	0.7	0.69	0.03	-0.02
Food	0.04	0.01	0.02	0.01	0.95	0.97
⋮ size cost alive verb	⋮	⋮				

e_{5391} e_{9853}

I want a glass of orange juice.

I want a glass of apple juice.

Andrew Ng

Language modeling

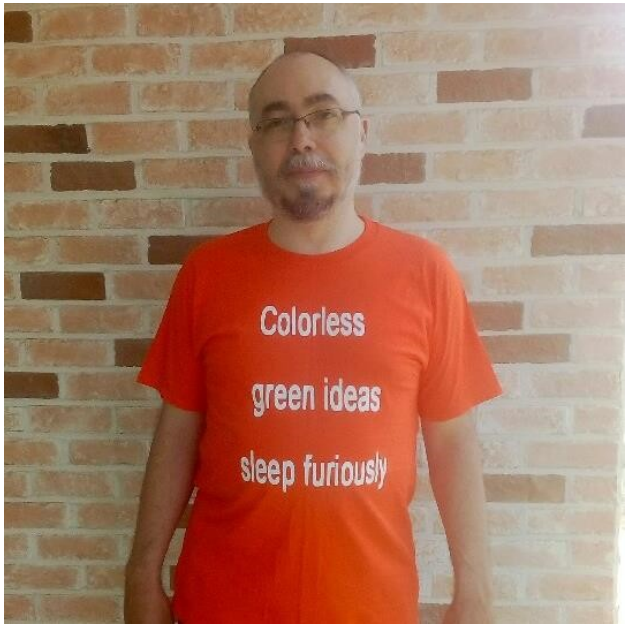
Given a sentence, what is the probability of that sentence?

E.g.

$$P(\text{The apple and pair salad}) = 3.2 \times 10^{-13}$$

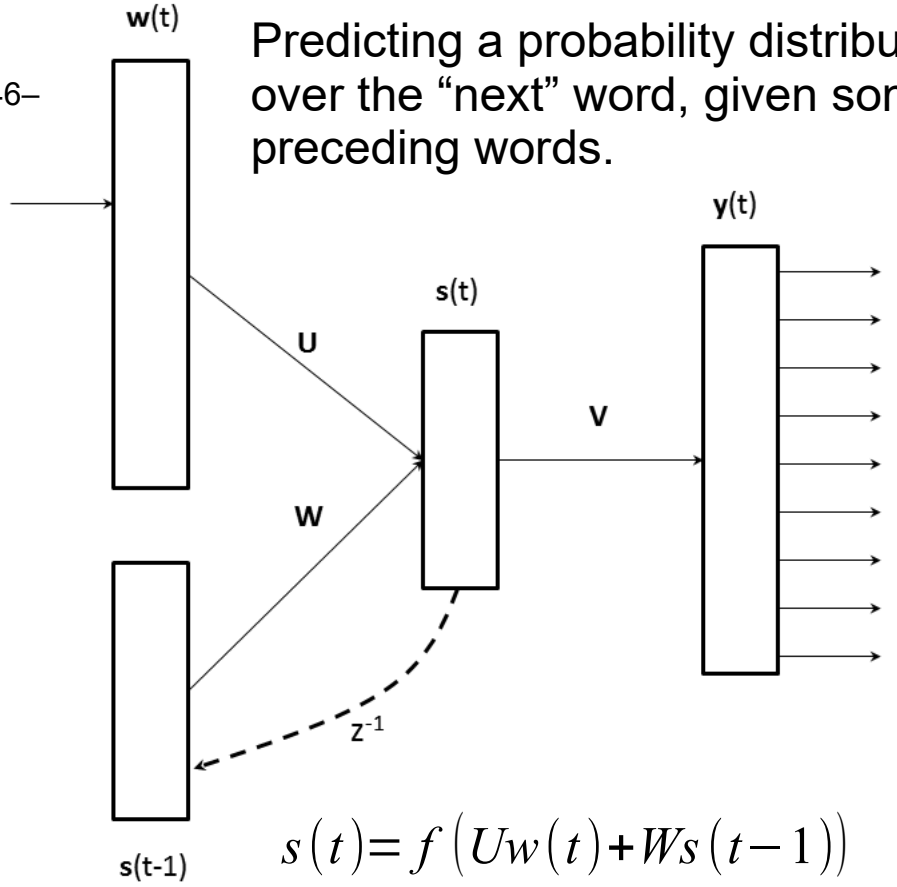
$$P(\text{The apple and pear salad}) = 5.7 \times 10^{-10}$$

$$\frac{P(\text{Colorless green ideas sleep furiously.})}{P(\text{Furiously sleep ideas green colorless.})} = 2 \times 10^{5}$$



Tomas Mikolov, Wen-tau Yih, Geoffrey Zweig. Linguistic Regularities in Continuous Space Word Representations. Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2013, 746–751, https://www.aclweb.org/anthology/N13-1090

Predicting a probability distribution over the "next" word, given some preceding words.



By training a neural network language model, one obtains not just the model itself, but also the learned word representations, which may be used for other, potentially unrelated, tasks.

$w(t)$ – input word at time *t* encoded using 1-of-*N* coding

$y(t)$ – a probability distribution over words

$s(t)$ maintains a representation of the sentence history

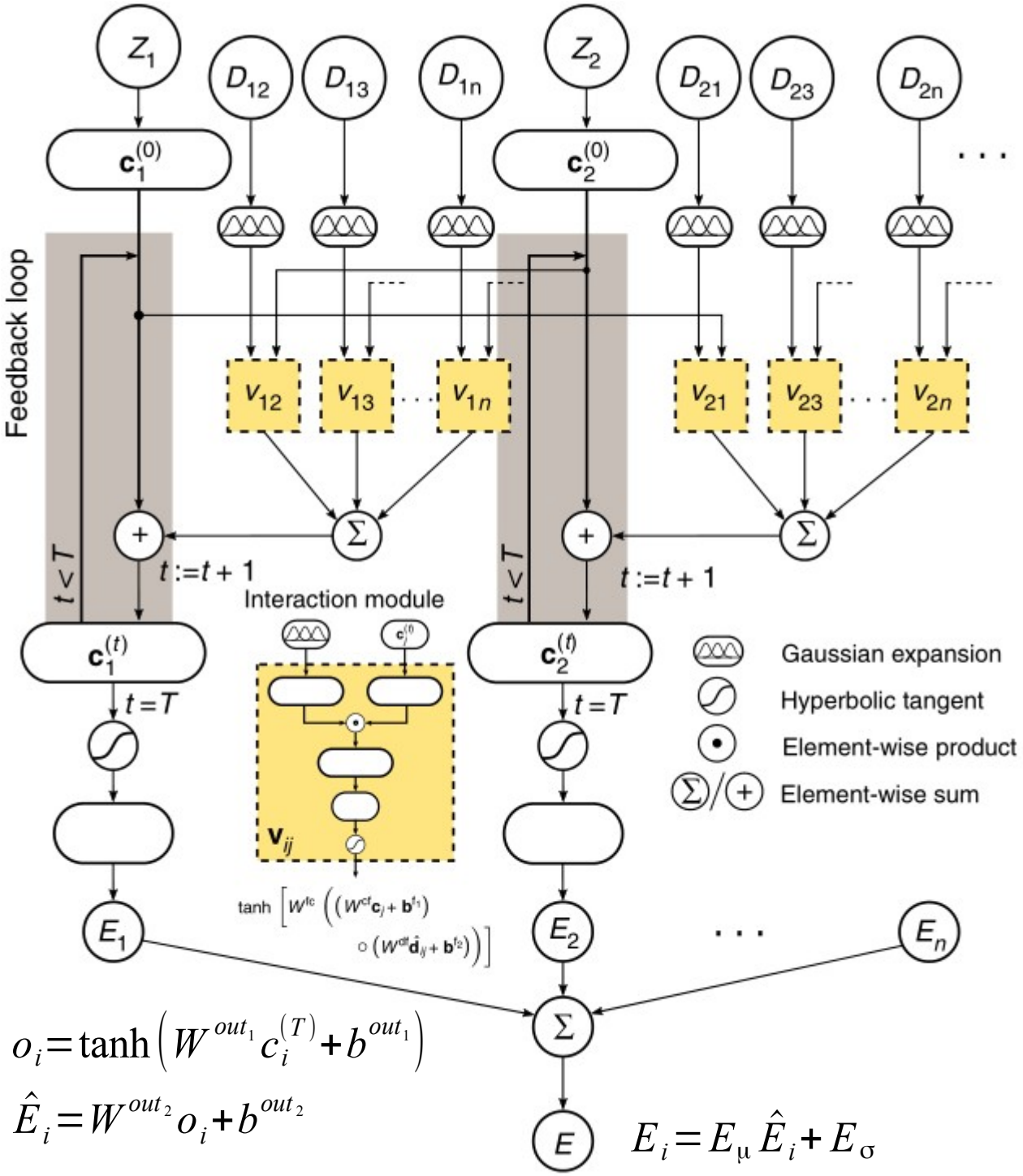
$w(t), y(t)$ have dimensionality of the vocabulary

$$s(t) = f\left(Uw(t) + Ws(t-1)\right)$$

$$y(t) = g\left(Vs(t)\right)$$

$$f(z) = \frac{1}{1+e^{-z}} \qquad g(z_m) = \frac{e^{z_m}}{\sum_k e^{z_k}}$$

The word representations are found in the columns of U , with each column representing a word.

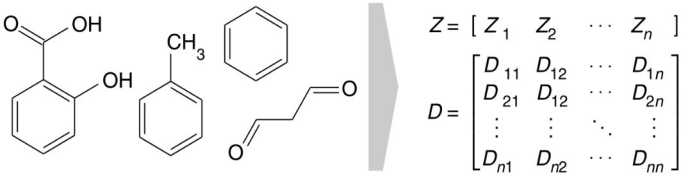


$$o_i = \tanh\left(W^{out_1} c_i^{(T)} + b^{out_1}\right)$$

$$\hat{E}_i = W^{out_2} o_i + b^{out_2}$$

$$E_i = E_\mu \hat{E}_i + E_\sigma$$

K. T. Schütt et al. Quantum-Chemical Insights from Deep Tensor Neural Networks, Nat. Commun. 8, 13890, 2017.
https://www.nature.com/articles/ncomms13890



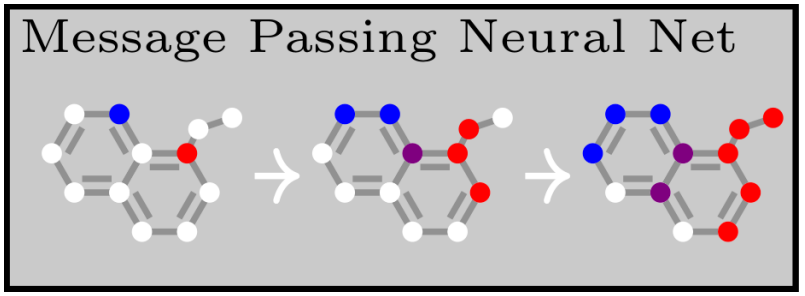
The distances are expanded in a Gaussian basis, yielding a feature vector $\hat{d}_{ij} \in \mathbb{R}^G$

Each atom *i* is represented by a coefficient vector $c \in \mathbb{R}^B$

B : # of basis functions (features)

This atomic expansion is repeatedly refined by pairwise interactions with the surrounding atoms

$$c_i^{(t+1)} = c_i^{(t)} + \sum_{j \neq i} v_{ij}$$



Learning features from (molecular) graphs

x_v node features

e_{vw} edge features

h_v^t hidden states for vertices

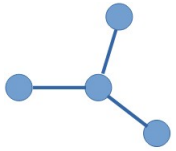
h_{vw}^t hidden states for edges

Gilmer et al Neural Message Passing for Quantum Chemistry https://arxiv.org/abs/1704.01212

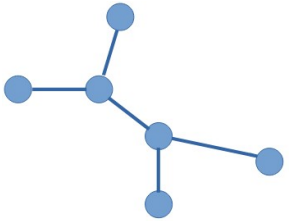
Two phases:

1) Message passing phase

T time steps



message functions



$$m_v^{t+1} = \sum_{w \in N(v)} M_v^t(h_v^t, h_w^t, e_{vw}) \qquad m_{vw}^{t+1} = \sum_{e \in N(e)} M_{vw}^t(h_{vw}^t, e, h_v^t, h_w^t)$$

$$h_v^{t+1} = U_v^t(h_v^t, m_v^{t+1}) \qquad h_{vw}^{t+1} = U_{vw}^t(h_{vw}^t, m_{vw}^{t+1})$$

update functions

2) Readout phase

$$\hat{y} = R(\{h_v^T, h_{vw}^T | v, w \in G\})$$

readout function, must be invariant to permutations of the node states

E.g. DTNN

$$M_v^t = \tanh\left(W^{fc}((W^{cf} h_w^t + b_1) \odot (W^{df} e_{vw} + b_2))\right)$$

$$U_v^t = h_v^t + m_v^{t+1}$$

$$R = \sum_v NN(h_v^T)$$

Swanson et al. Deep learning for automated classification and characterization of amorphous materials. Soft Matter, 2020,16, 435-446
https://doi.org/10.1039/C9SM01903K