Fundamentals of Machine Learning for Machine Translation

Dr. John D. Kelleher
ADAPT Centre for Digital Content Technology[1]
Dublin Institute of Technology, Ireland

Translating Europe Forum 2016
Brussels, Charlemagne Building, Rue de la Loi 170
27^{th} October 2016

[1]The ADAPT Centre is funded under the SFI Research Centres Programme (Grant 13/RC/2106) and is co-funded under the European Regional Development Fund

Outline

What is a function?

A function maps a set of inputs (numbers) to an output (number)

$$sum(2, 5, 4) \rightarrow 11$$

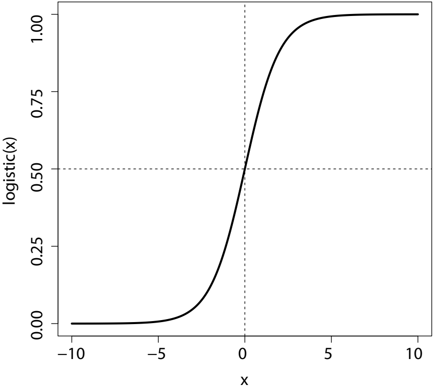
What is a WEIGHTEDSUM function?

$$\textsc{weightedSum}(\underbrace{[n_1, n_2, \ldots, n_m]}_{\text{Input Numbers}}, \underbrace{[w_1, w_2, \ldots, w_m]}_{\textit{Weights}})$$
$$= (n_1 \times w_1) + (n_2 \times w_2) + \cdots + (n_m \times w_m)$$

$$\begin{aligned}\textsc{weightedSum}([3, 9], [-3, 1]) &\\ &= (3 \times -3) + (9 \times 1) \\ &= -9 + 9 \\ &= 0\end{aligned}$$

What is an ACTIVATION function?

An ACTIVATION function takes the output of our WEIGHTEDSUM function and applies another mapping to it.



What is an ACTIVATION function?

$$\text{ACTIVATION} =$$

$$\text{LOGISTIC}(\text{WEIGHTEDSUM}((\underbrace{[n_1, n_2, \ldots, n_m]}_{\text{Input Numbers}}, \underbrace{[w_1, w_2, \ldots, w_m]}_{\textit{Weights}})))$$

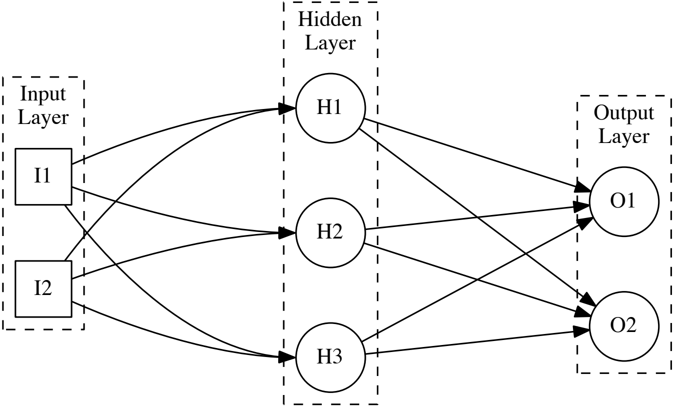
$$\begin{aligned} \text{LOGISTIC}(\text{WEIGHTEDSUM}([3, 9], [-3, 1])) &= \text{LOGISTIC}((3 \times -3) + (9 \times 1)) \\ &= \text{LOGISTIC}(-9 + 9) \\ &= \text{LOGISTIC}(0) \\ &= 0.5 \end{aligned}$$

What is a NEURON?

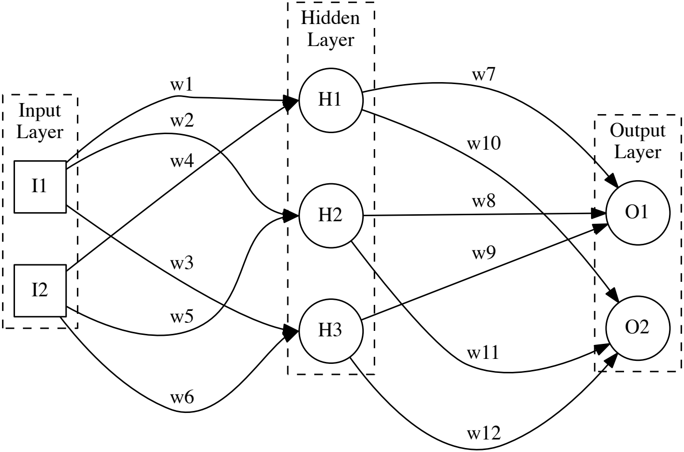
The simple list of operations that we have just described defines the fundamental building block of a neural network: the NEURON.

$$\textsc{Neuron} = \textsc{activation}(\textsc{weightedSum}((\underbrace{[n_1, n_2, \ldots, n_m]}_{\text{Input Numbers}}, \underbrace{[w_1, w_2, \ldots, w_m]}_{\textit{Weights}})))$$

What is a Neural Network?



Where do the WEIGHTS come from?



Training a Neural Network

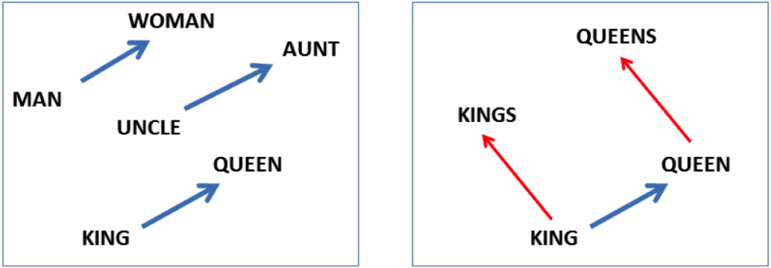
- We train a neural network by iteratively updating the weights
- We start by randomly assigning weights to each edge
- We then show the network examples of inputs and expected outputs and update the weights using Backpropogation so that the network outputs match the expected outputs
- We keep updating the weights until the network is working the way we want

Word Embeddings

Each word is represented by a vector of numbers that positions the word in a multi-dimensional space, e.g.:

$$
\begin{aligned}
king &=< 55, -10, 176, 27 > \\
man &=< 10, 79, 150, 83 > \\
woman &=< 15, 74, 159, 106 > \\
queen &=< 60, -15, 185, 50 >
\end{aligned}
$$

Word Embeddings



$$vec(King) - vec(Man) + vec(Woman) \approx vec(Queen)^2$$

[2] Linguistic Regularities in Continuous Space Word Representations (Mikolov et al., 2013)

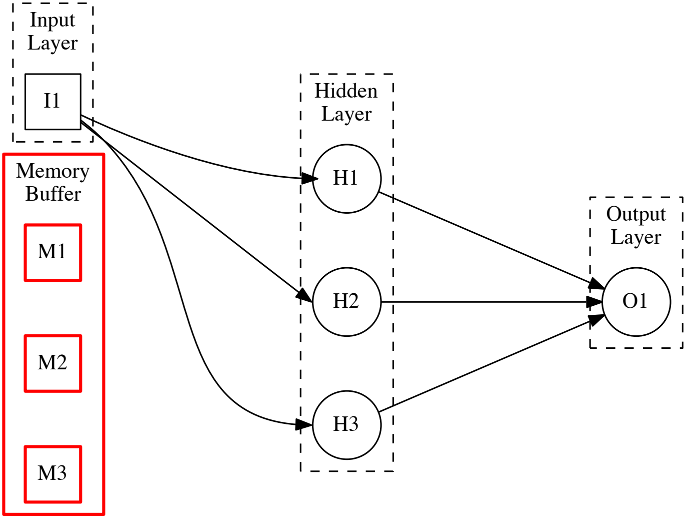
Recurrent Neural Networks

A particular type of neural network that is useful for processing sequential data (such as, language) is a RECURRENT NEURAL NETWORK.

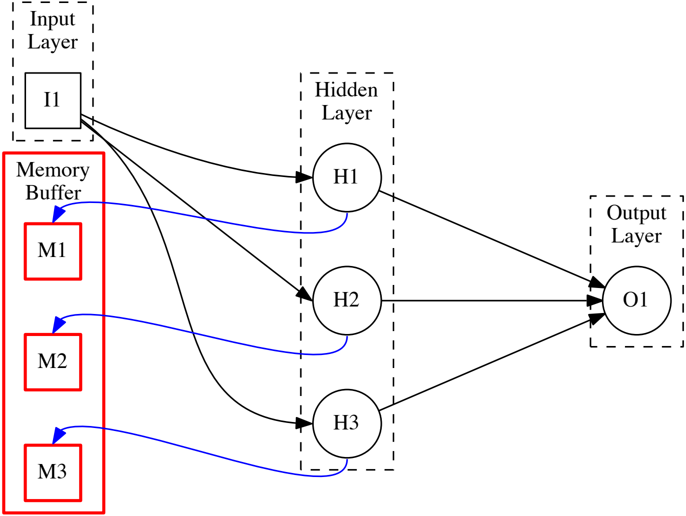
Using an RNN we process our sequential data one input at a time.

In an RNN the outputs of some of the neurons for one input are feed back into the network as part the next input.

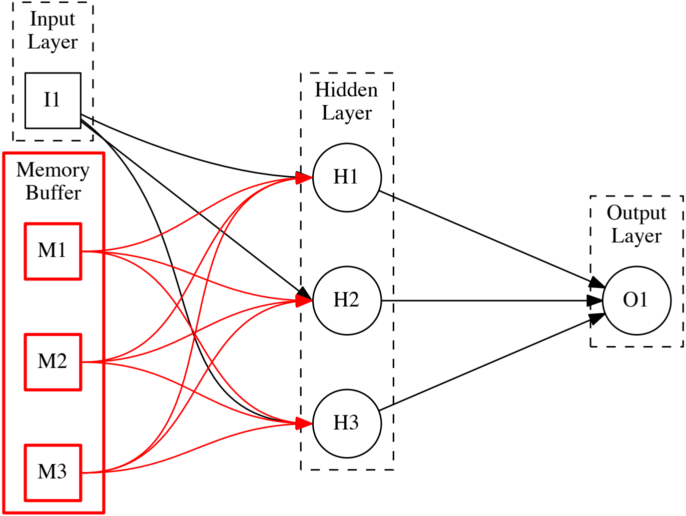
Recurrent Neural Networks



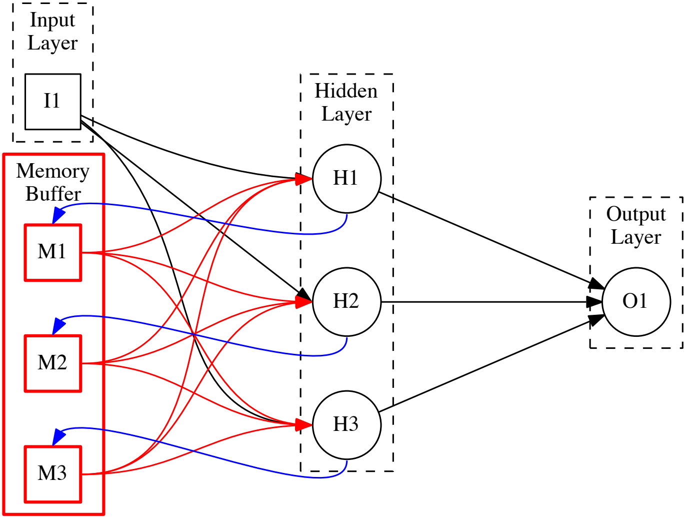
Recurrent Neural Networks



Recurrent Neural Networks



Recurrent Neural Networks



Recurrent Neural Networks

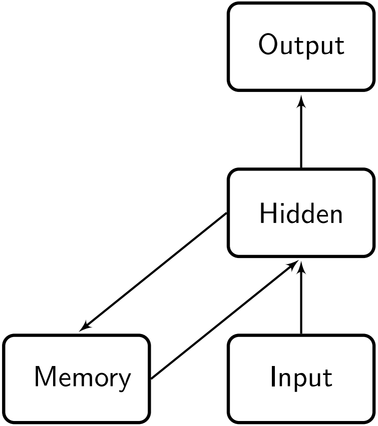


Figure: Recurrent Neural Network

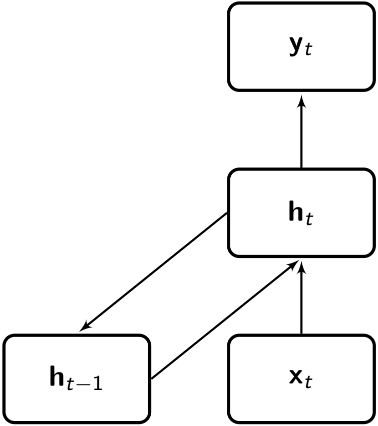


Figure: Recurrent Neural Network

Recurrent Neural Networks

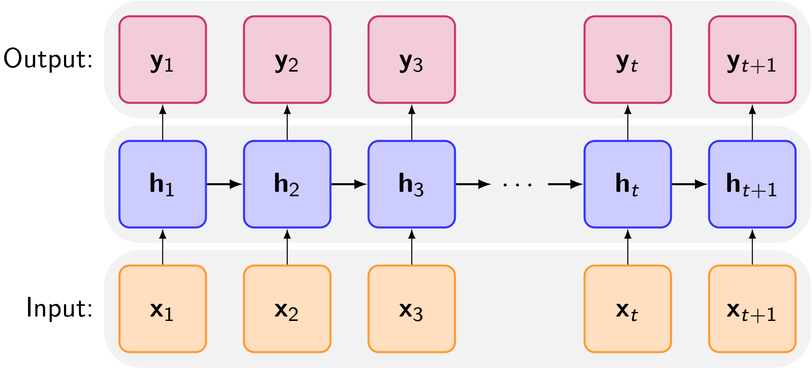


Figure: RNN Unrolled Through Time

Recurrent Neural Networks

1. RNN Encoders
2. RNN Language Models

Encoders

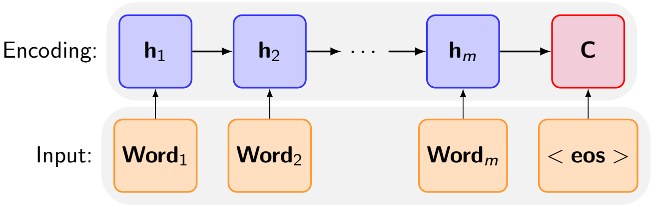


Figure: Using an RNN to Generate an Encoding of a Word Sequence

Language Models

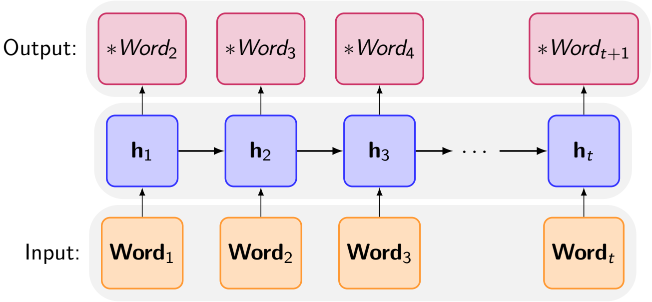


Figure: RNN Language Model Unrolled Through Time

Decoder

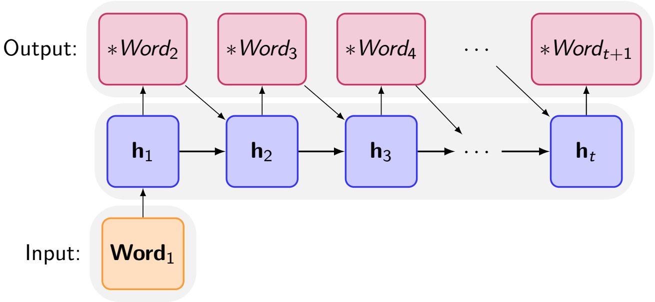


Figure: Using an RNN Language Model to Generate (Hallucinate) a Word Sequence

Encoder-Decoder Architecture

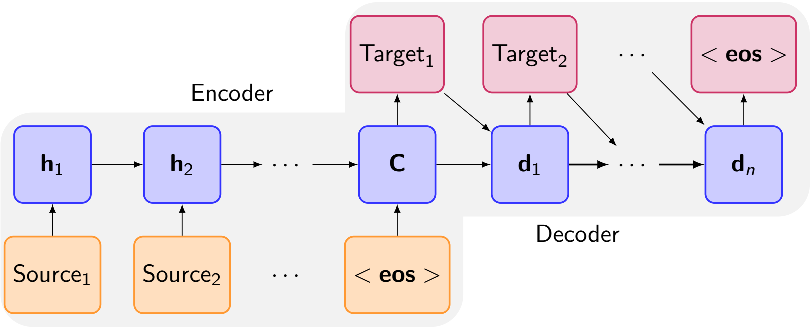


Figure: Sequence to Sequence Translation using an Encoder-Decoder Architecture

Neural Machine Translation

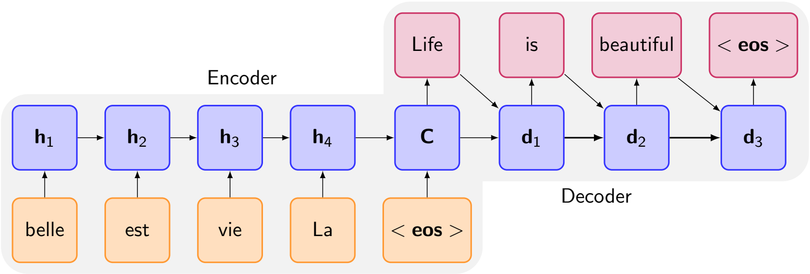


Figure: Example Translation using an Encoder-Decoder Architecture

Conclusions

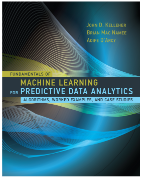
- An advantage of the ENCODER-DECODER architecture is that the system processes the entire input before it starts translating
- This means that the decoder can use what it has already generated and the entire source sentence when generating the next word in the translation

Conclusions

- There is ongoing research on what is the best way to present the source sentence to the encoder
- There is also ongoing research on giving the decoder the ability to attend to different parts of the input during translation
- There is also interesting work on improving how these systems handle idiomatic language

Thank you for your attention

john.d.kelleher@dit.ie
@johndkelleher
www.comp.dit.ie/jkelleher
www.machinelearningbook.com



Acknowledgements: The ADAPT Centre is funded under the SFI Research Centres Programme (Grant 13/RC/2106) and is co-funded under the European Regional Development Fund.