TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät für Elektro- und Informationstechnik, Professur für Grundlagen der Elektrotechnik und Elektronik

FliK Modul 2020

GAN and RNN

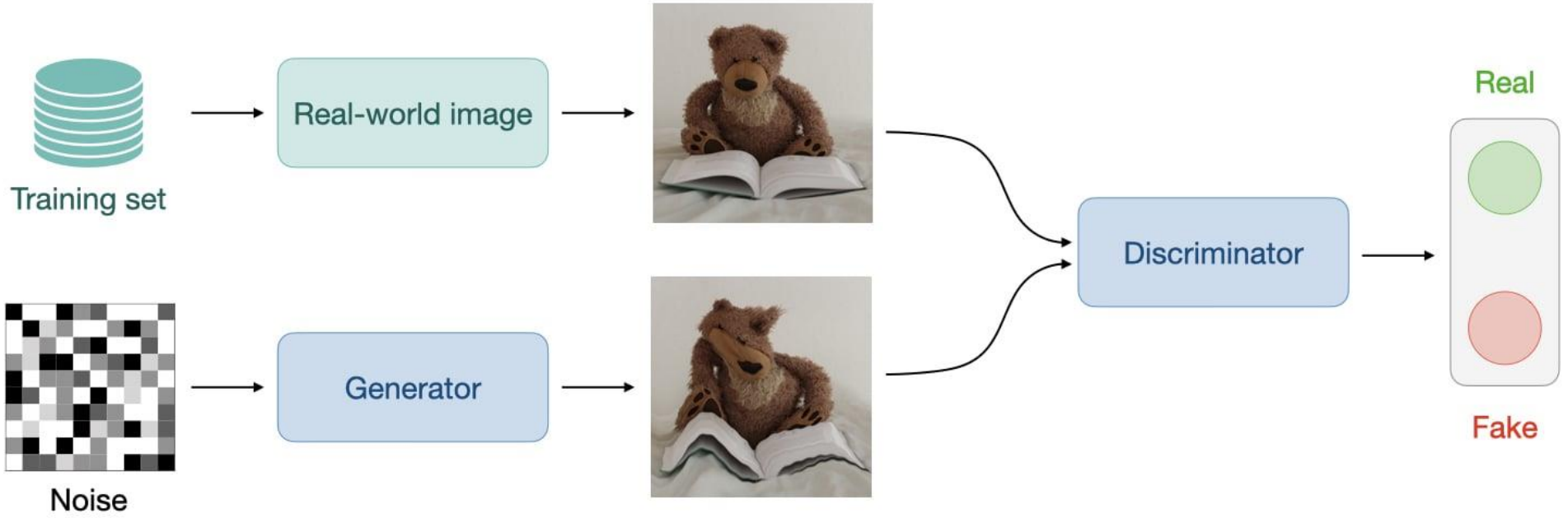
Steffen Seitz, Marvin Arnold & Markus Fritzsche

Prof. Ronald Tetzlaff

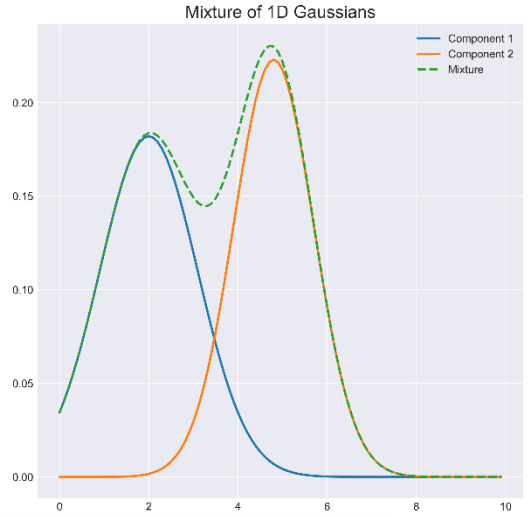
Dresden, 19-23.10.

Generative Adversial Networks (GAN)

Like VAE a Generative Adversial Network is a **generative learning** approach, hence you try to model a **distribution** (instead of a probability) as close as possible to your data to sample from it. Training a GAN is **completly different** from what we have seen so far.



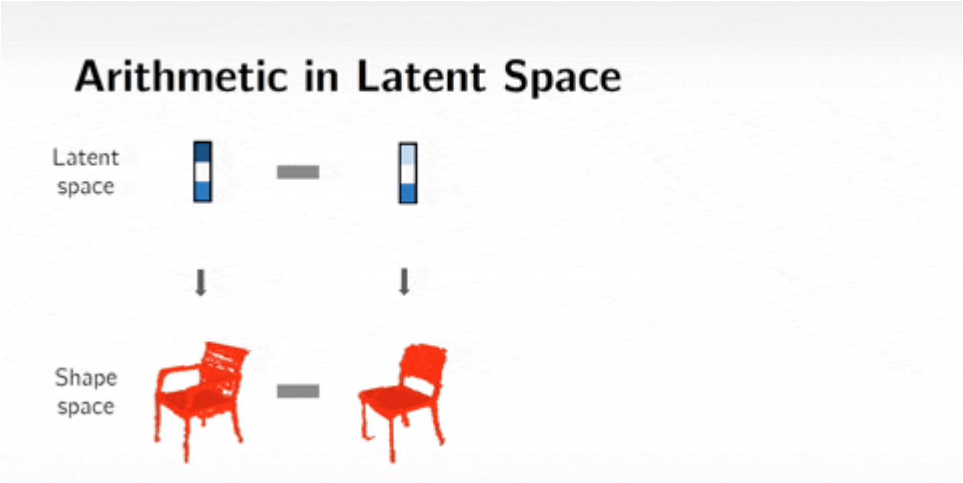
Generative Adversial Networks (GAN)



Since GAN models a **distribution**, we can use this to do **latent space arithmetics** (similar of adding gausian distributions) and sample from this new continuous space, with some funny results.

Generative Adversial Networks (GAN)

We can also do **style transfer**!



Style transfer with Ron Swanson (Deepfakes)

12. Exercise

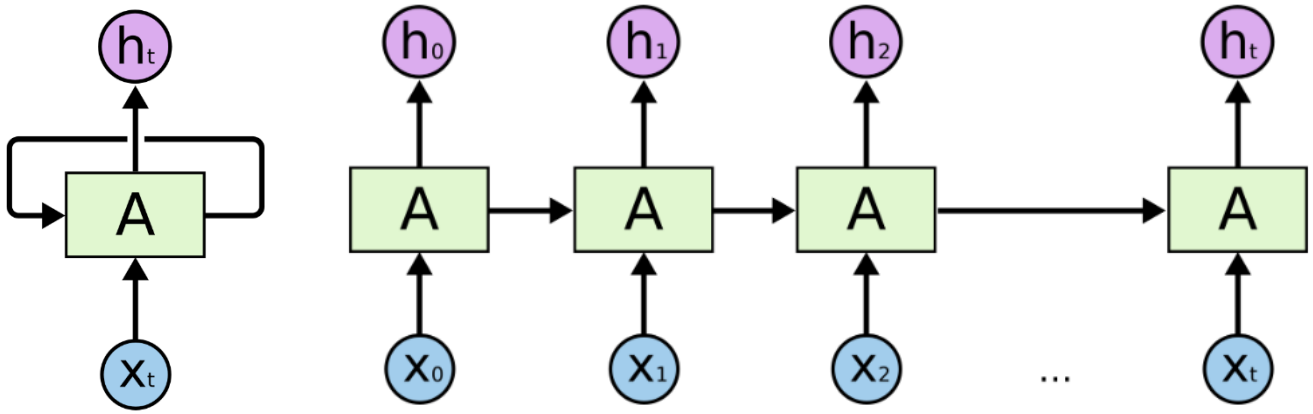
Let‘s train our first GAN!

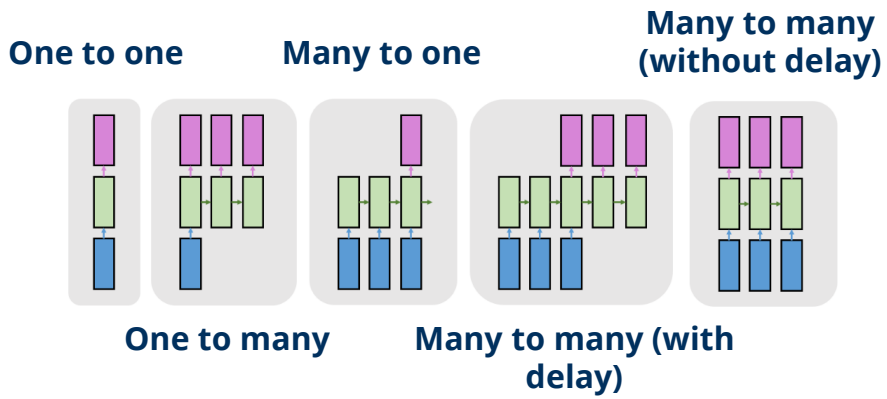
Recurrent Neural Networks

$$h_t = f_W(h_{t-1}, x_t)$$

New State / Some Transferfunction with Parameters W — Old State — Input Data at Timestep t

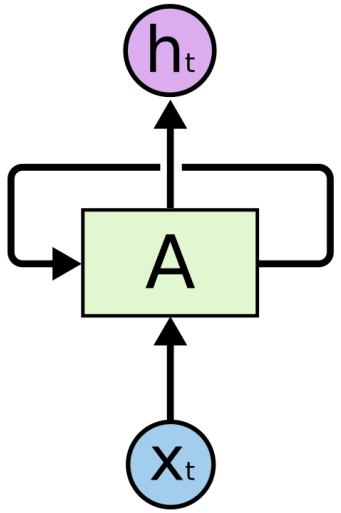
Humans don't start their thinking from scratch every second. As you read this text, **you understand each word based on your understanding of previous words**. You don't throw everything away and start thinking from scratch again. Your thoughts have **persistence**.

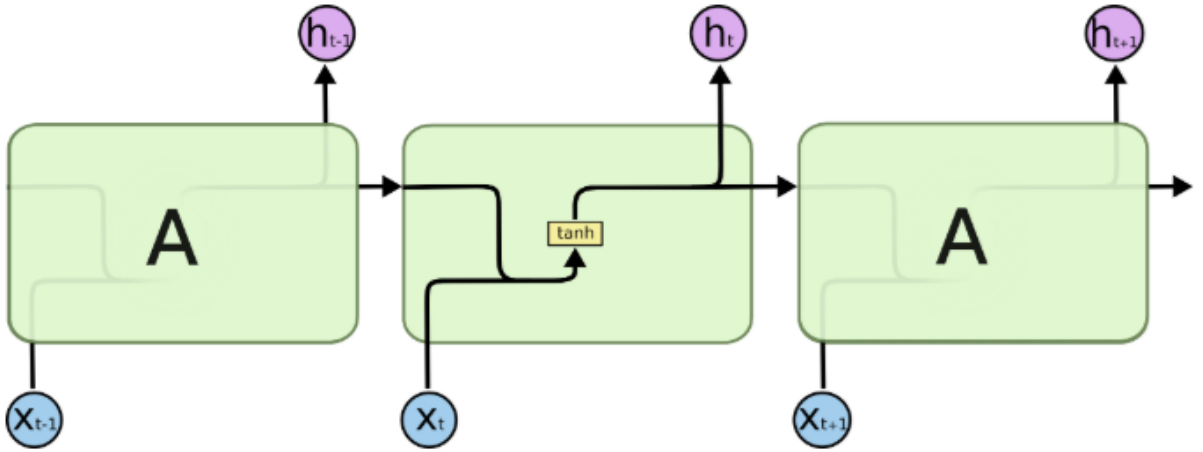




Recurrent Neural Networks

```
model = Sequential()
model.add(RNN(hidden_size, input_shape=(1, look_back), return_sequences=True)
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam')
model.fit(trainX, trainY, epochs=100, batch_size=1, verbose=2)
```





$$h_t = f_W(h_{t-1}, x_t)$$

New State — Old State — Input Data at Timestep t

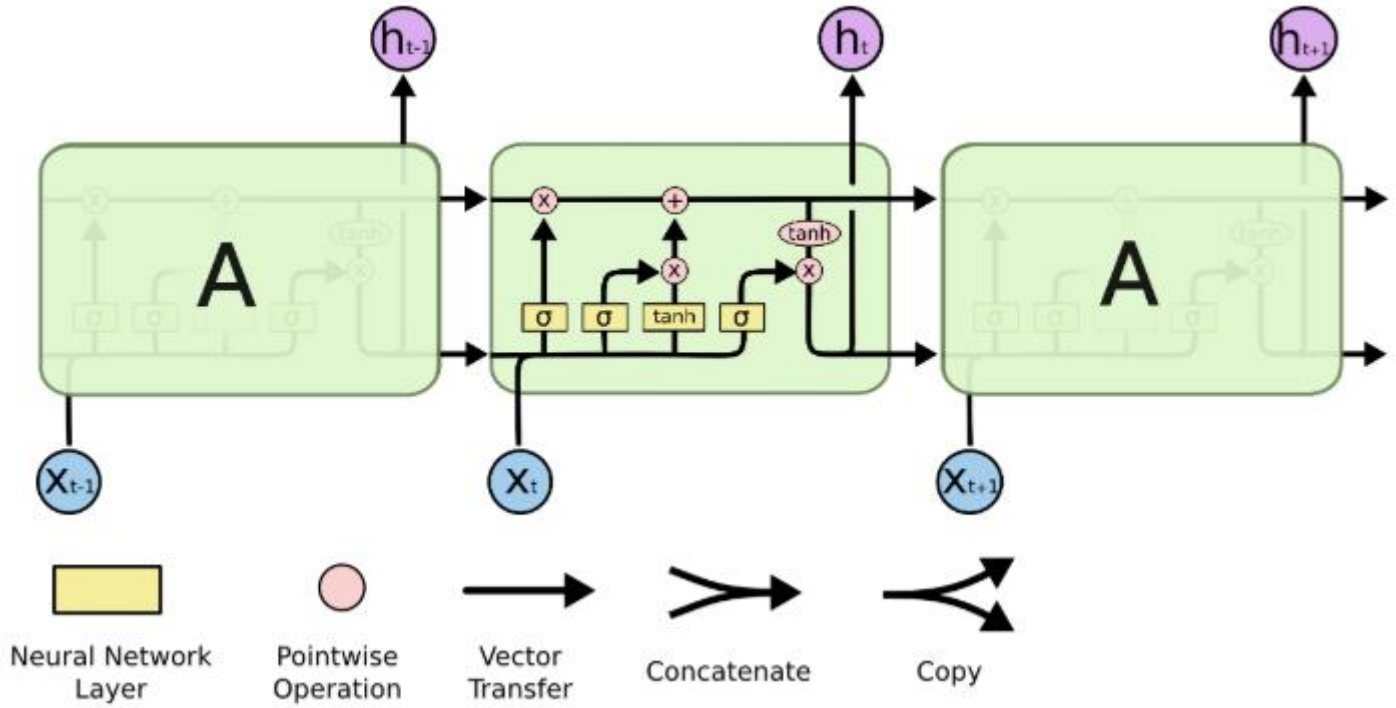
Some Transferfunction with Parameters W

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

$$= \tanh\left((W_{hh} \quad W_{hx})\begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$

$$= \tanh\left(W\begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$

LSTM



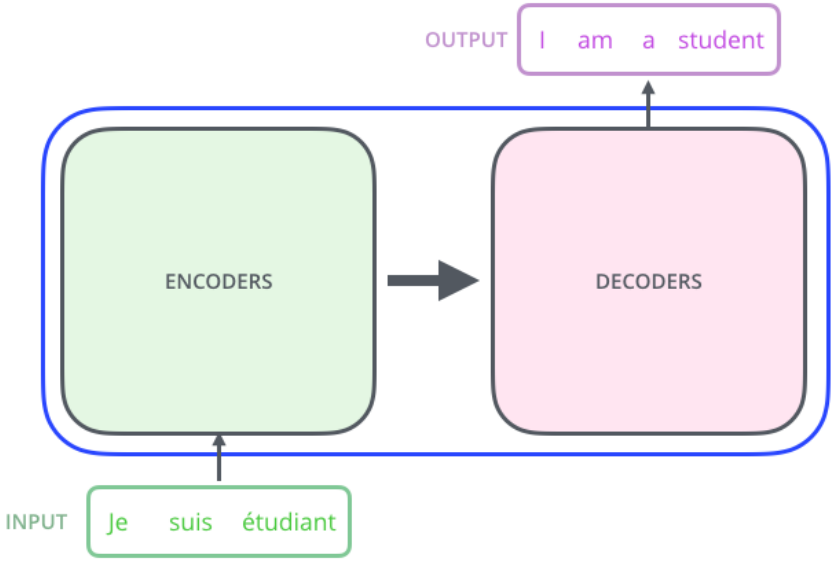
$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

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

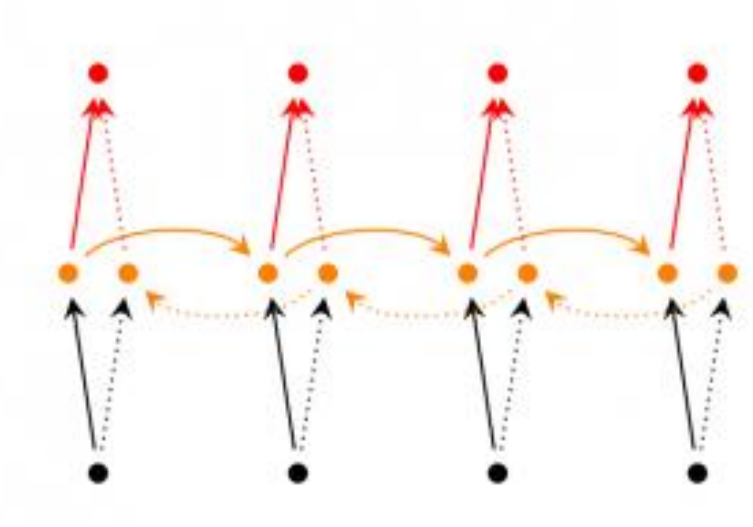
LSTM introduces a „gating“ structure that implements **a gradient highway** (like Resnet) to make them **resillient** to vanishing gradient

BidirectionalRNN/Sequence to Sequence Models



Translation models are natural **sequence to sequence** use cases!

Bidirectional RNN

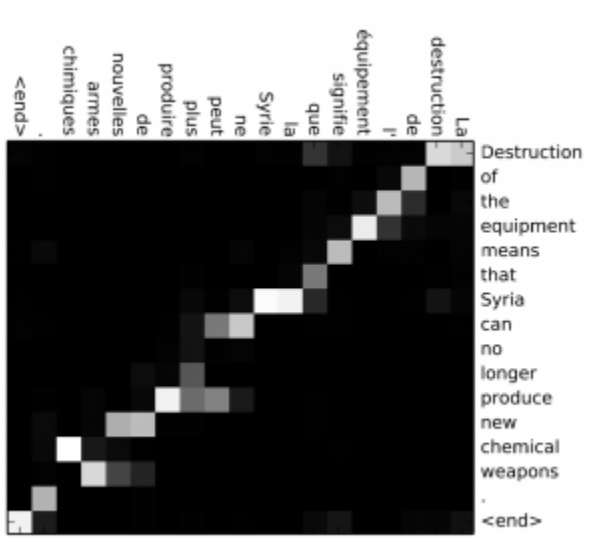


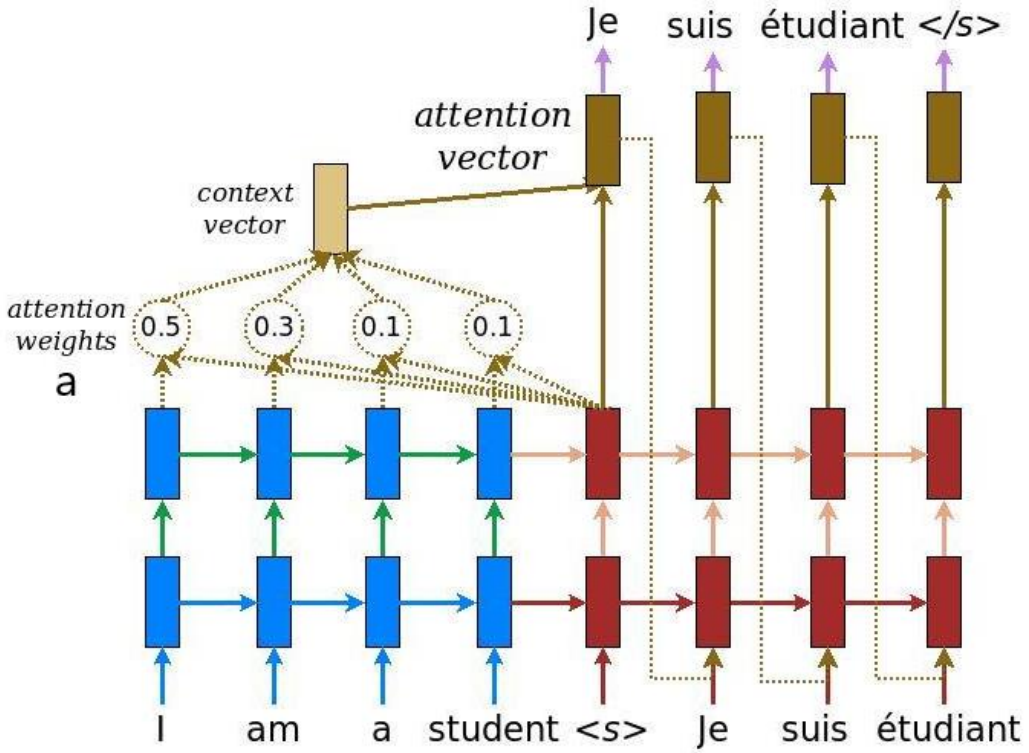
Bidirectional RNN are **two seperate** RNN receiving reversed outputs, but their output is **added together** in the end.

Attention

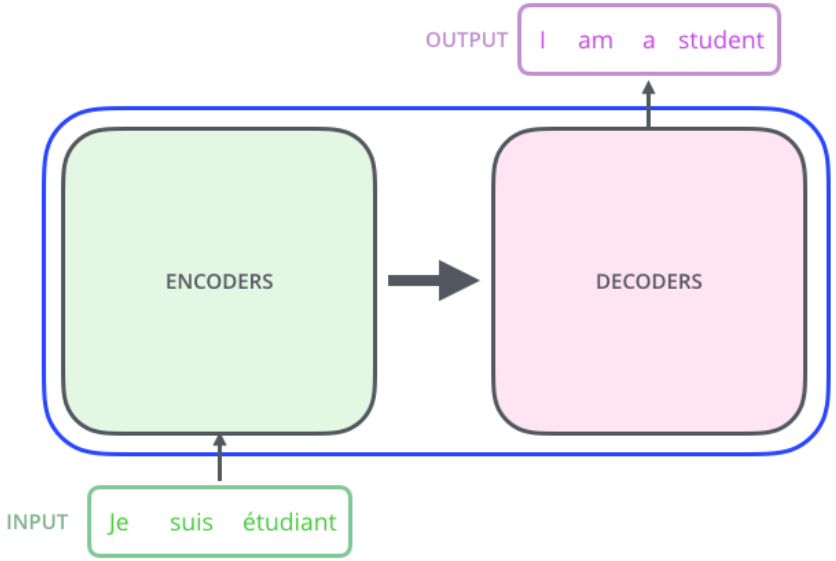
Attention is a layer on top of an RNN that detemines the importance of each input to the output of the network.

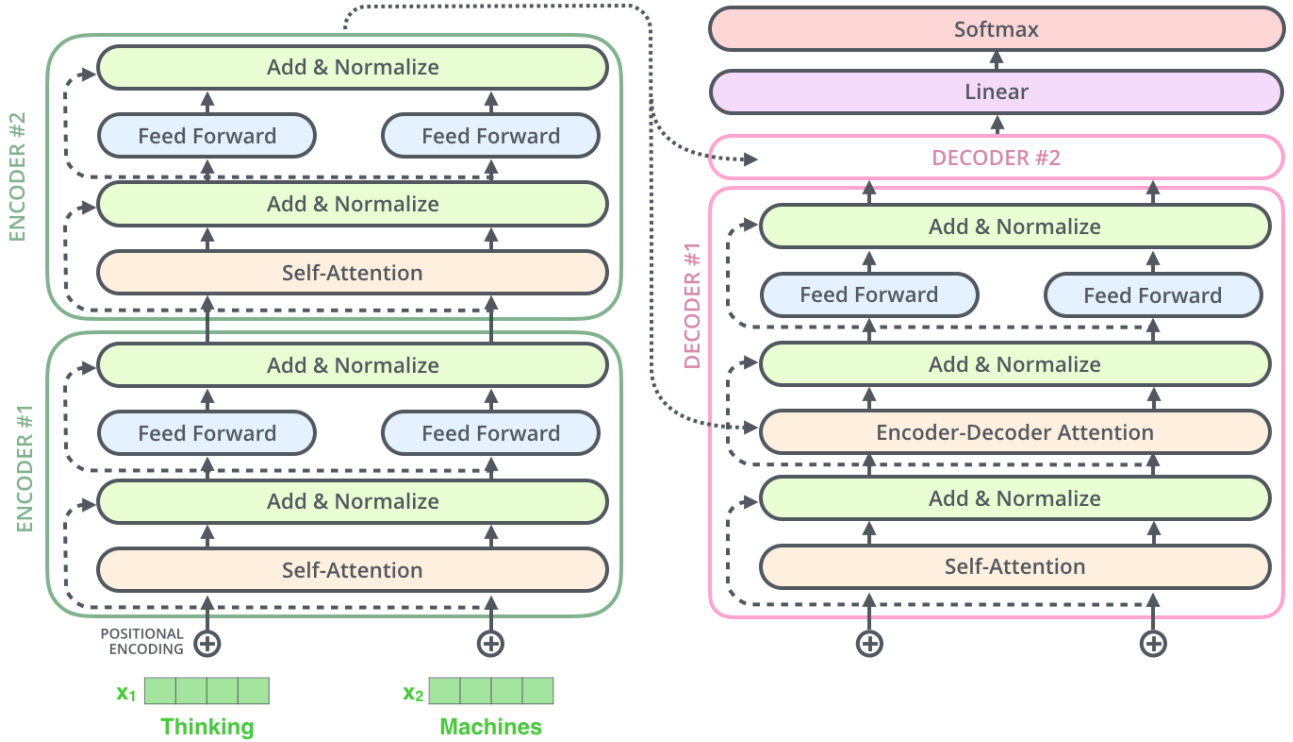
With attention it is possible to **plot** the **cross dependencies** of specific words of an input.





Transformer
Attention is all you need





Transformers are state of the art models for many translation tasks!

13. Exercise

Let's train our first RNN on IMDB!