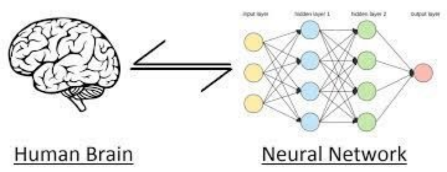
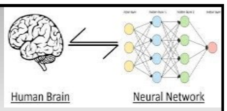
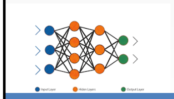


Artificial Neural Network and Deep Learning



Agenda

- ❑ Generative Adversarial Network (GAN)
- ❑ Recurrent Neural Network (RNN)
  - ❑ LSTM
  - ❑ GRU
- ❑ Attention Mechanism
- ❑ Transformers

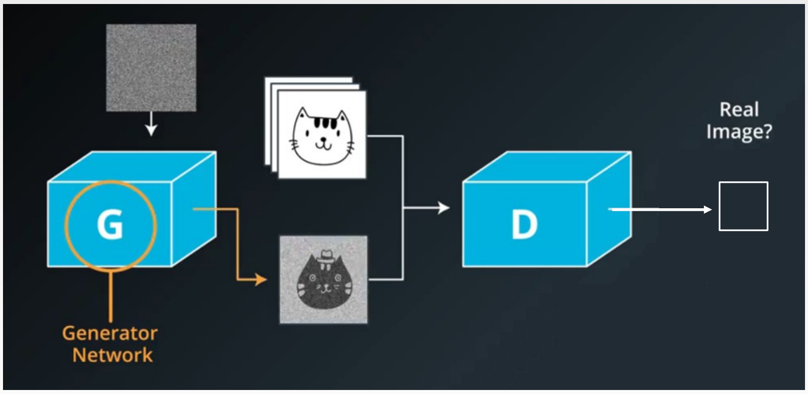


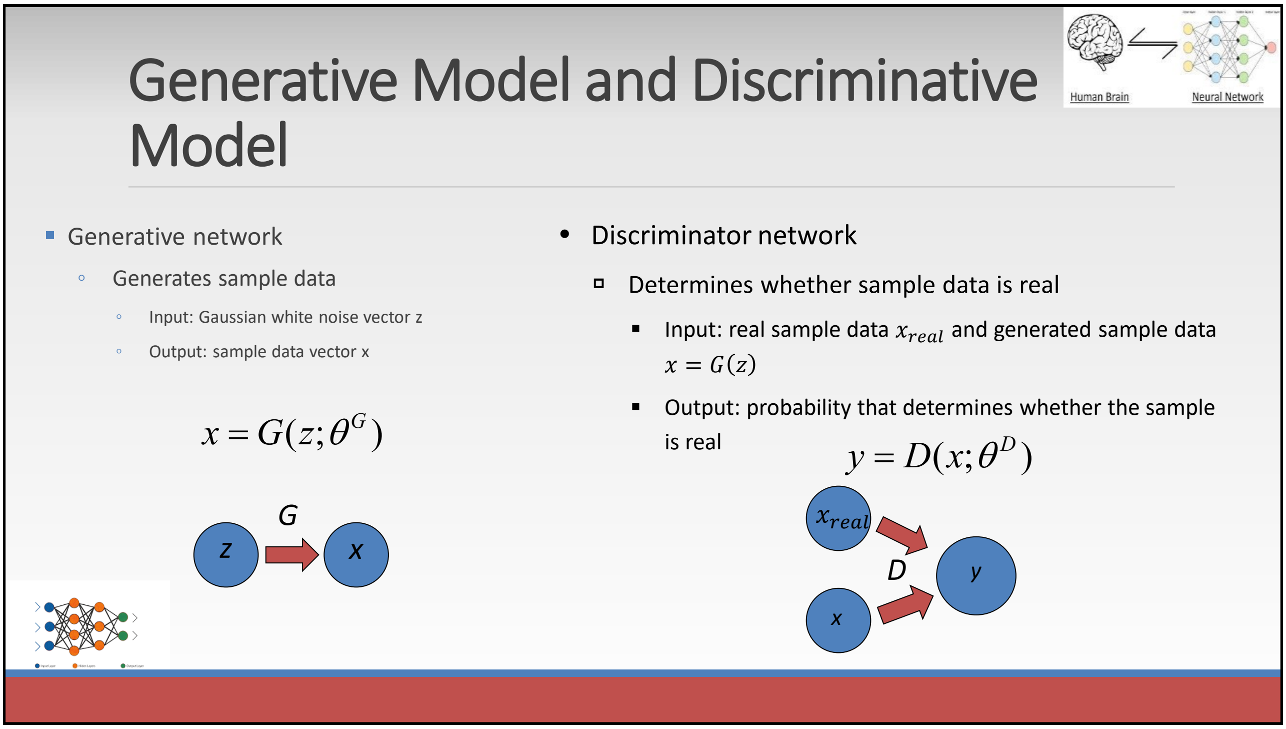
Generative Adversarial Network (GAN)

- Generative Adversarial Network is a **unsupervised generative** framework that trains generator G and discriminator D through the adversarial process. Through the adversarial process, the discriminator can tell whether the sample from the generator is fake or real. GAN adopts a mature BP algorithm.
- **Generator G**: The **input is noise z**, which complies with manually selected prior probability distribution, such as even distribution and Gaussian distribution.
  - The generator adopts the network structure of the **multilayer perceptron (MLP)**, uses maximum likelihood estimation (MLE) parameters to represent the derivable mapping G(z), and maps the input space to the sample space.
- **Discriminator D**: The **input is the real sample** x **and** the **fake sample** G(z), which are tagged as real and fake respectively.
  - The network of the discriminator can use the MLP carrying parameters. The output is the probability D(G(z)) that determines whether the sample is a real or fake sample.
- GAN can be applied to **scenarios** such as **image generation, text generation, speech enhancement, image super-resolution.**

GAN Architecture

- Generator/Discriminator




Human Brain
Neural Network
G
z
x
x_{real}
D
y
x

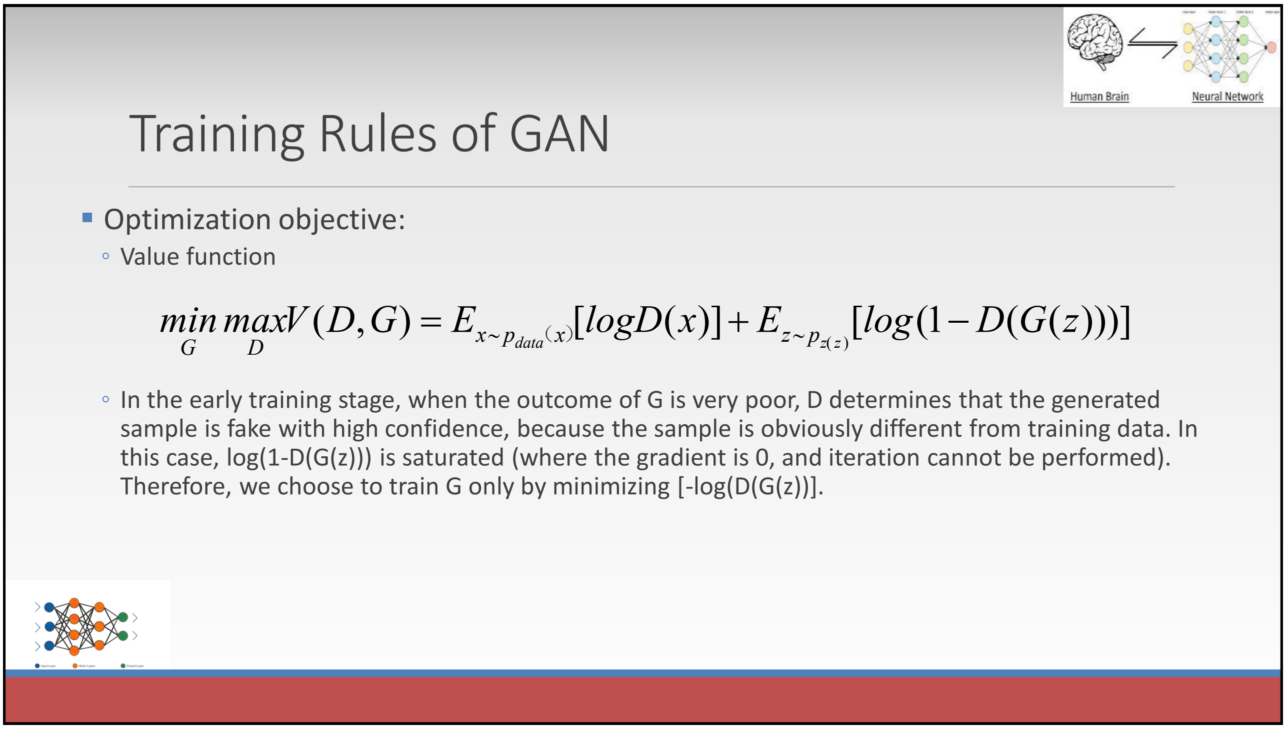
Generative Model and Discriminative Model

- Generative network
  - Generates sample data
    - Input: Gaussian white noise vector z
    - Output: sample data vector x

$$x = G(z;\theta^G)$$

- Discriminator network
  - Determines whether sample data is real
    - Input: real sample data x_{real} and generated sample data $x = G(z)$
    - Output: probability that determines whether the sample is real

$$y = D(x;\theta^D)$$


Human Brain
Neural Network

Training Rules of GAN

- Optimization objective:
  - Value function

$$\underset{G}{min}\,\underset{D}{max}V(D,G) = E_{x\sim p_{data}(x)}[logD(x)] + E_{z\sim p_{z}(z)}[log(1 - D(G(z)))]$$

  - In the early training stage, when the outcome of G is very poor, D determines that the generated sample is fake with high confidence, because the sample is obviously different from training data. In this case, log(1-D(G(z))) is saturated (where the gradient is 0, and iteration cannot be performed). Therefore, we choose to train G only by minimizing [-log(D(G(z))].

Recurrent Neural Network

- The Recurrent Neural Network (RNN) is a neural network that **captures dynamic information in sequential data through periodical connections of hidden layer nodes**. It can classify sequential data.
- Unlike other forward neural networks, the **RNN can keep a context state and even store, learn, and express related information in context windows of any length**. Different from traditional neural networks, it is not limited to the space boundary, but also **supports time sequences**. In other words, there is a side between the hidden layer of the current moment and the hidden layer of the next moment.
- The RNN is widely used in **scenarios related to sequences**, such as **videos consisting of image frames, audio consisting of clips, and sentences consisting of words.**

Recurrent Neural Network Architecture

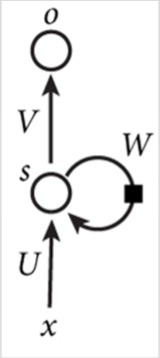
- X_t is the input of the input sequence at time t.
- S_t is the memory unit of the sequence at time t and caches previous information.

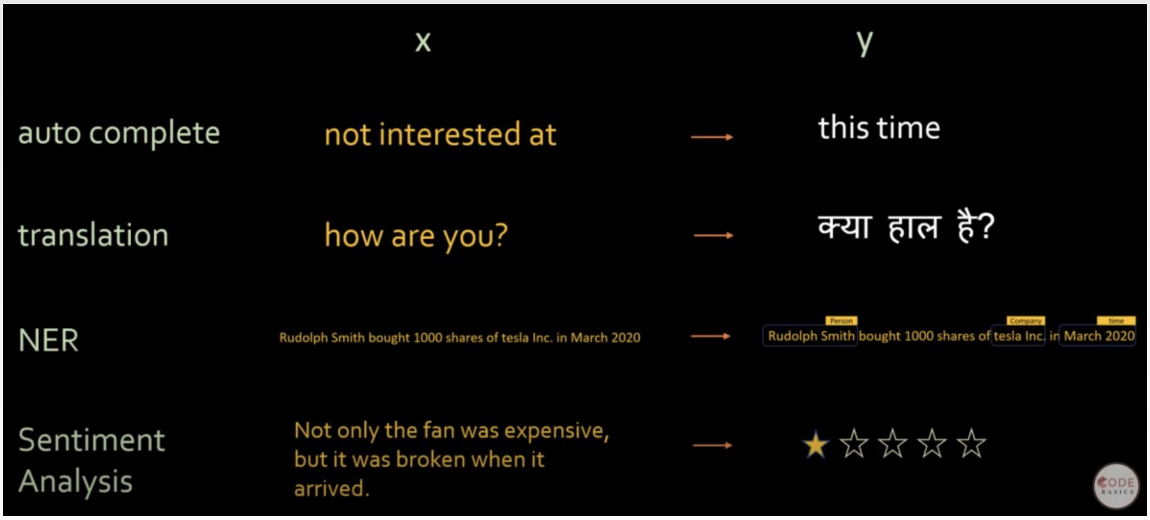
$$S_t = tanh(UX_t + WS_{t-1}).$$

- O_t is the output of the hidden layer of the sequence at time t.

$$O_t = tanh(VS_t)$$

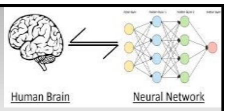
- O_t after through multiple hidden layers, it can get the final output of the sequence at t.





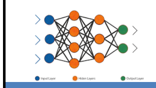
	x		y
auto complete	not interested at	→	this time
translation	how are you?	→	क्या हाल है?
NER	Rudolph Smith bought 1000 shares of tesla Inc. in March 2020	→	Person Company time Rudolph Smith bought 1000 shares of tesla Inc. in March 2020
Sentiment Analysis	Not only the fan was expensive, but it was broken when it arrived.	→	★☆☆☆☆

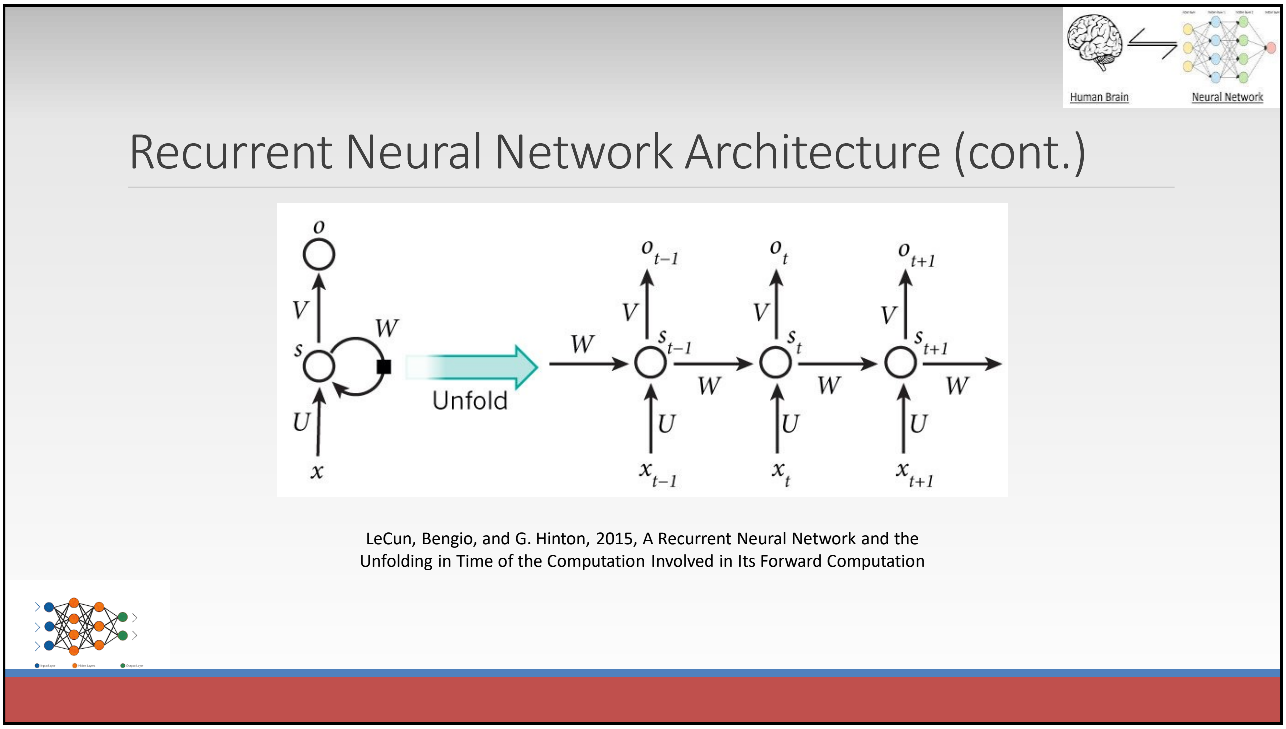
CODE BASICS



Why can’t we use a simple neural network?

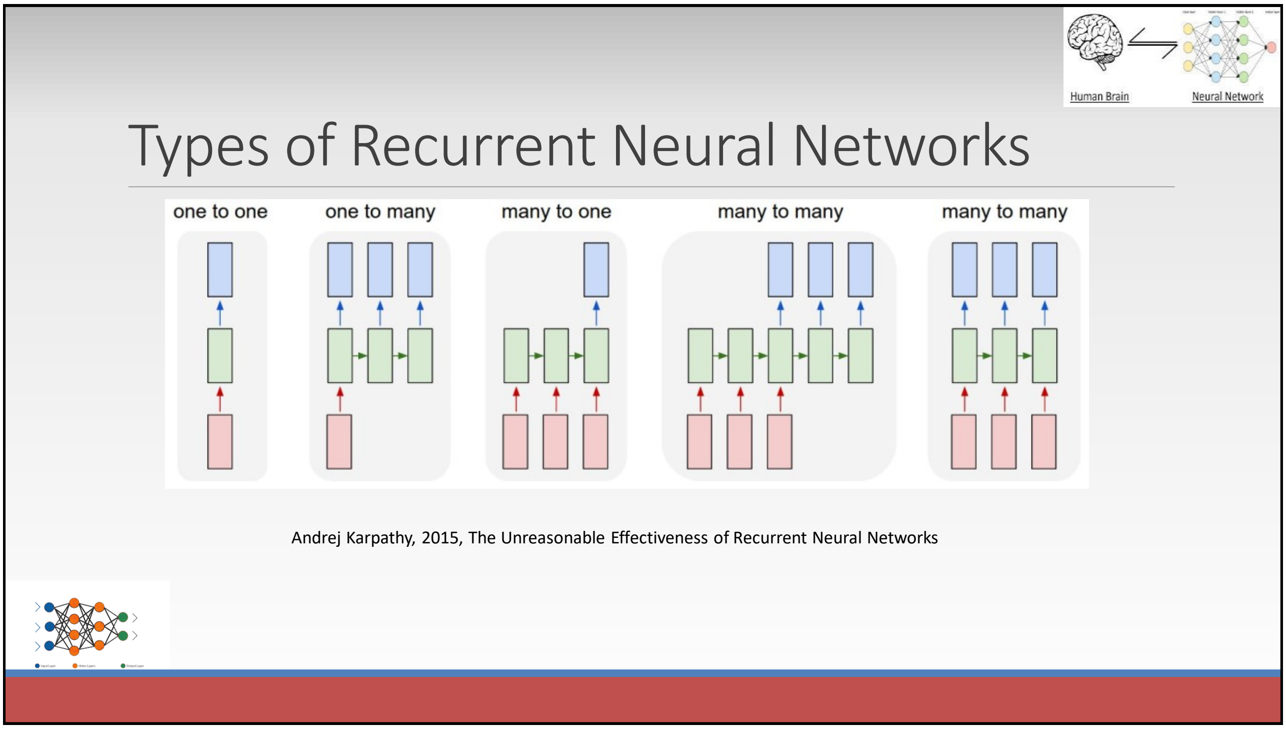
- **The sequence is important**. In a simple ANN, the order of the inputs doesn’t make a difference in the prediction while but in sequence modelling problems (e.g. NLP) sequence is important, “how are you” is different from “how you are”.
- **No fixed sizes of neuron**. Your input sentence could be of any size and you cannot decide a fixed input size.
- **Too much computation** if processed the same way as ANN. Because each word has to be converted into a vector of 1s and 0s. For example, if the English vocab has 2500 words, each word will be represented as 1 in a vector of 2500 numbers (one hot encoding).





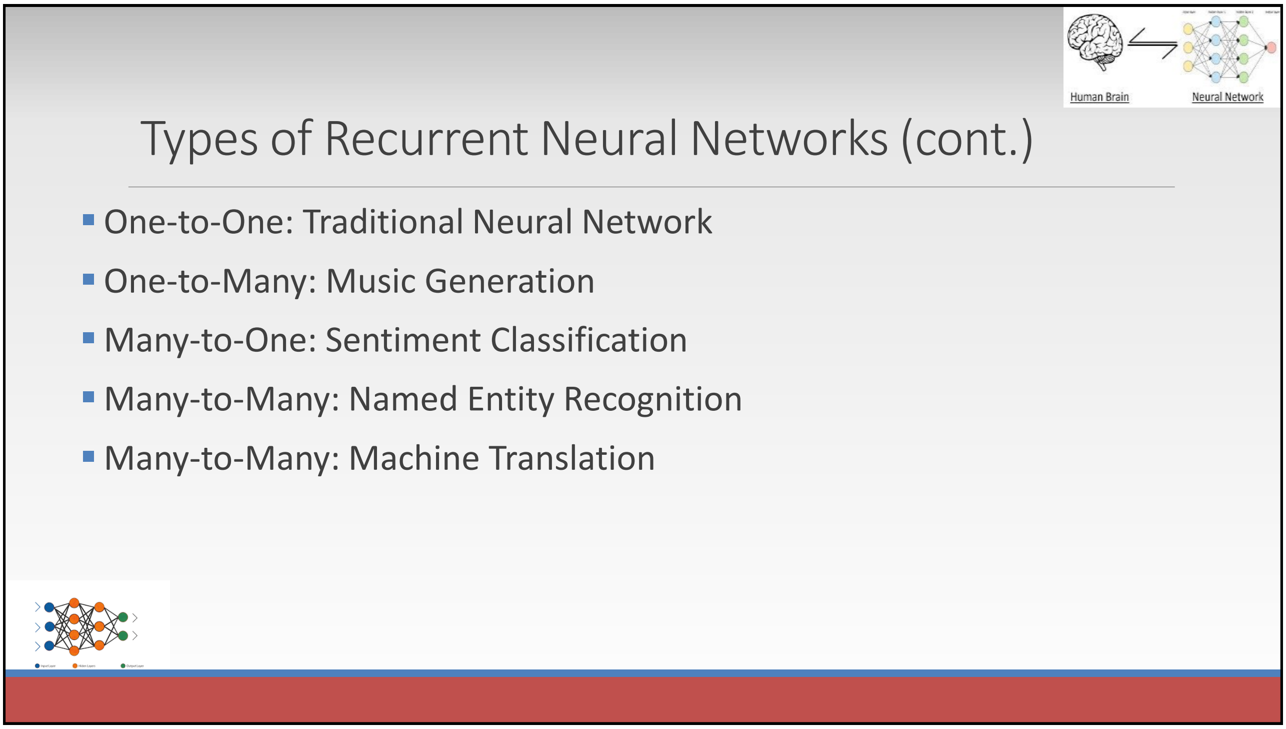
Recurrent Neural Network Architecture (cont.)

LeCun, Bengio, and G. Hinton, 2015, A Recurrent Neural Network and the Unfolding in Time of the Computation Involved in Its Forward Computation



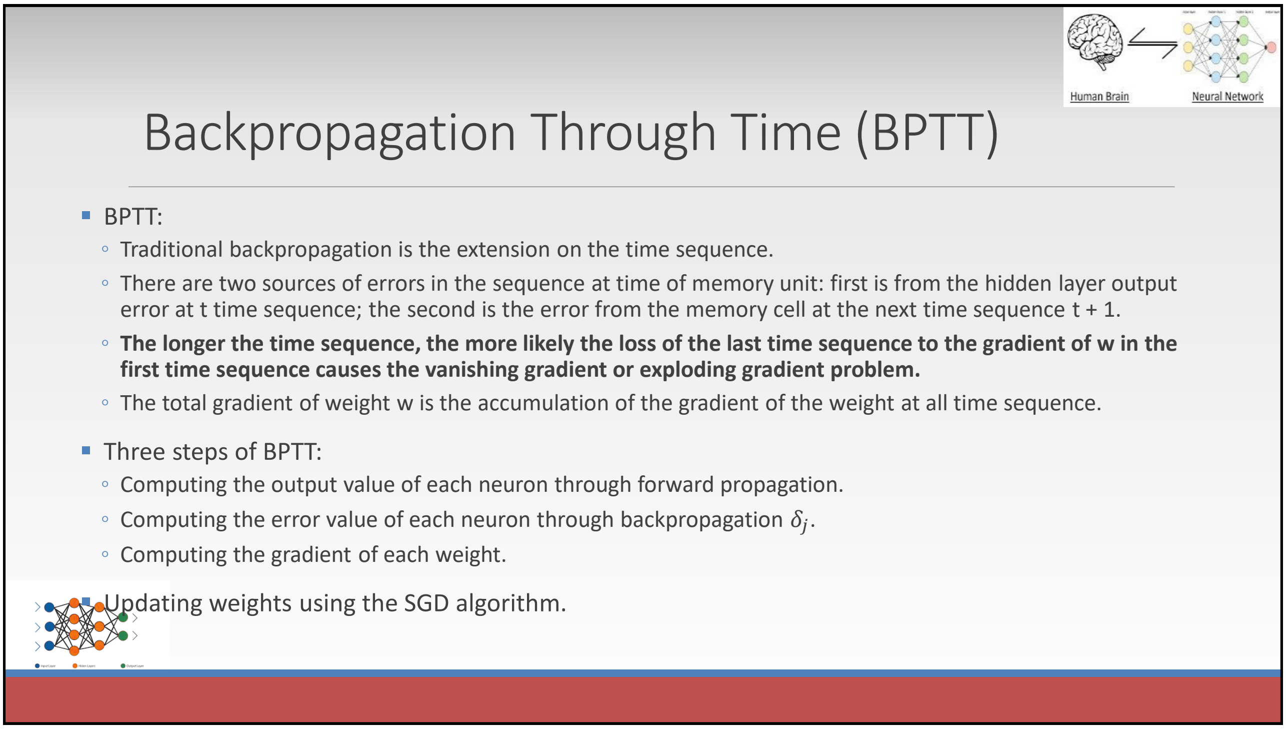
Types of Recurrent Neural Networks

Andrej Karpathy, 2015, The Unreasonable Effectiveness of Recurrent Neural Networks



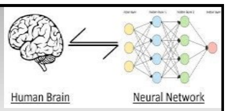
Types of Recurrent Neural Networks (cont.)

- One-to-One: Traditional Neural Network
- One-to-Many: Music Generation
- Many-to-One: Sentiment Classification
- Many-to-Many: Named Entity Recognition
- Many-to-Many: Machine Translation



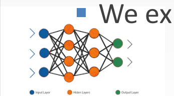
Backpropagation Through Time (BPTT)

- BPTT:
  - Traditional backpropagation is the extension on the time sequence.
  - There are two sources of errors in the sequence at time of memory unit: first is from the hidden layer output error at t time sequence; the second is the error from the memory cell at the next time sequence t + 1.
  - **The longer the time sequence, the more likely the loss of the last time sequence to the gradient of w in the first time sequence causes the vanishing gradient or exploding gradient problem.**
  - The total gradient of weight w is the accumulation of the gradient of the weight at all time sequence.
- Three steps of BPTT:
  - Computing the output value of each neuron through forward propagation.
  - Computing the error value of each neuron through backpropagation δ_j.
  - Computing the gradient of each weight.
- Updating weights using the SGD algorithm.

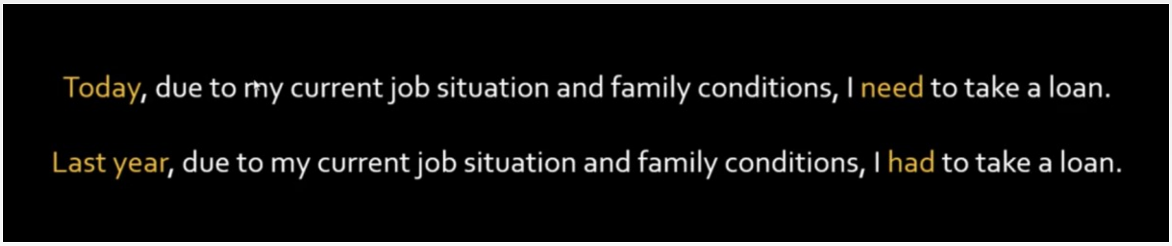


Recurrent Neural Network Problem

- $S_t = \sigma(UX_t + WS_{t-1})$ is extended on the time sequence.
- $S_t = \sigma\left(UX_t + W\left(\sigma\left(UX_{t-1} + W\left(\sigma\left(UX_{t-2} + W(\ldots)\right)\right)\right)\right)\right)$
- Despite that the standard RNN structure solves the problem of information memory, the information attenuates during long-term memory.
- Information needs to be saved long time in many tasks. For example, a hint at the beginning of a speculative fiction may not be answered until the end.
- **The RNN may not be able to save information for long due to the limited memory unit capacity**.
- We expect that memory units can remember key information.

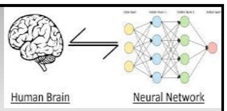


In English language the words at the beginning has an effect on the words at the end, traditional RNN have a short memory give more weight to closer hidden layers than further layers due to the vanishing gradient issue. There are two solutions for this, the GRU and LSTM.



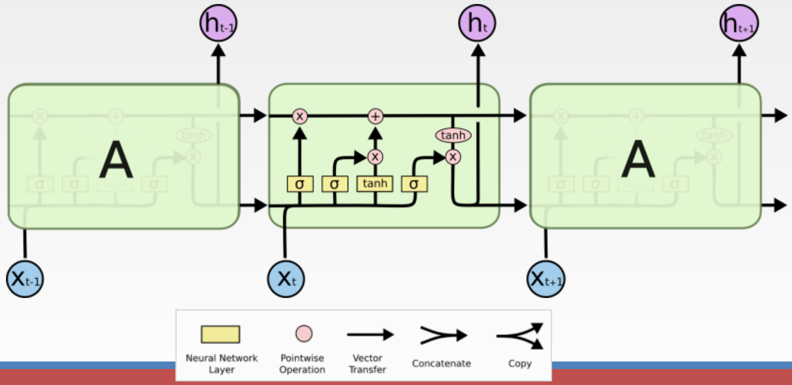
Today, due to my current job situation and family conditions, I need to take a loan.

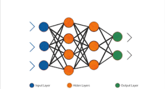
Last year, due to my current job situation and family conditions, I had to take a loan.



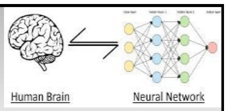
Long Short-term Memory Network

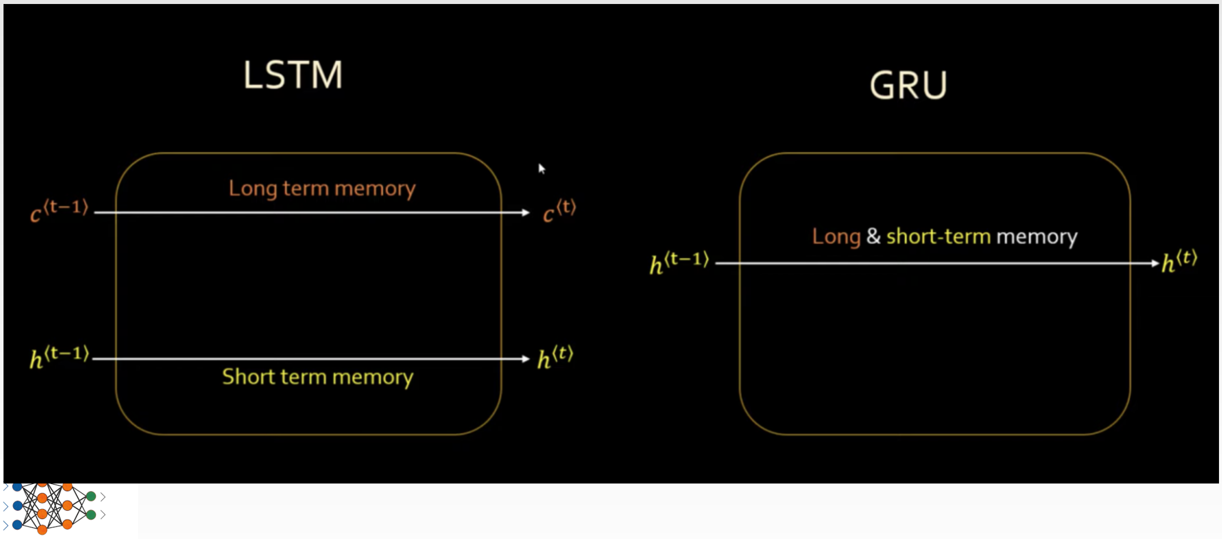
- Long short-term memory (LSTM) applies to the scenario with a **large gap between the relevant information and the point where it is needed.**
- LSTM can **connect previous information for long periods of time** to the present task.

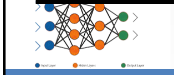




Colah, 2015, Understanding LSTMs Networks

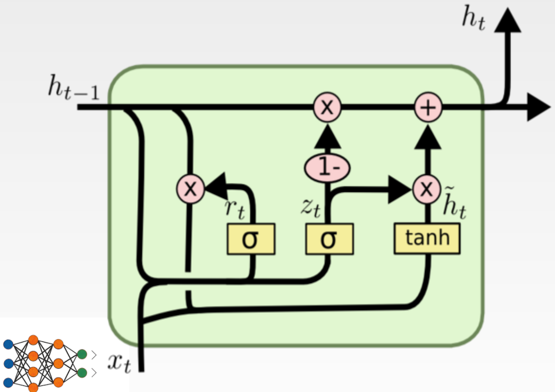






Gated Recurrent Unit (GRU)

- GRU model is simpler than LSTM. GRU combines the Forget Gate and the Input Gate into a single Update Gate.



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

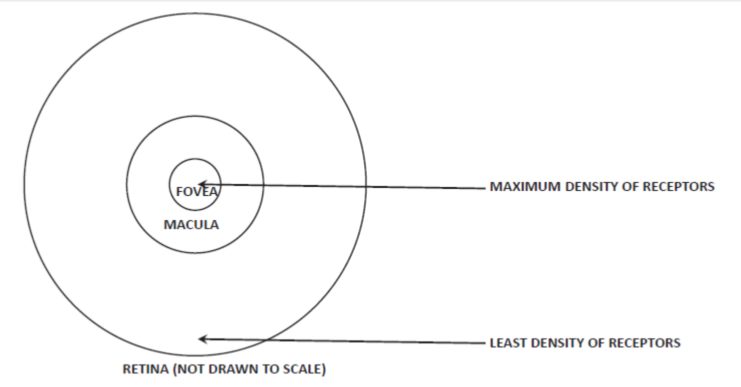
LSTM	GRU
Has short-term memory and long-term memory	Combines short-term memory and long-term memory
Has three gates: forget gate, input gate and output gate	Has two gates: reset gate and update get
More computation	Less computation
Invented in 1995-1997	Invented in 2014 (gaining popularity)

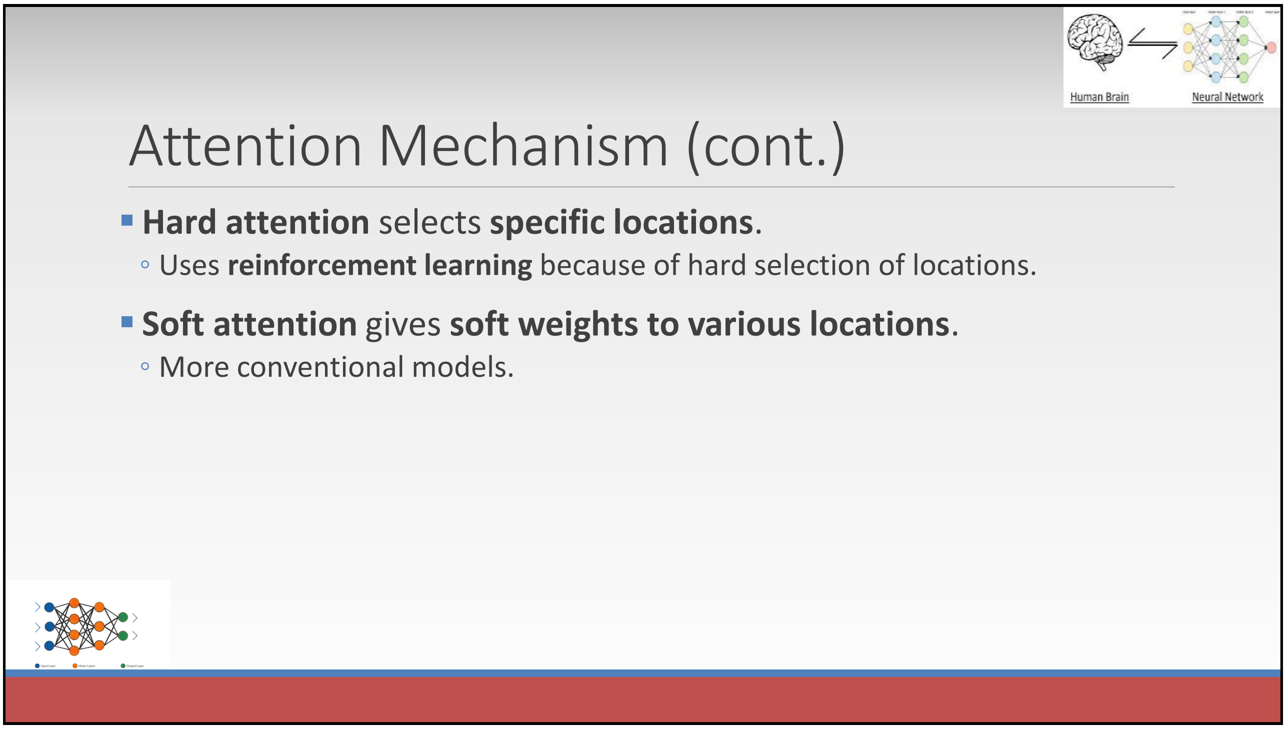
Attention Mechanism

- Humans **do not actively use all the information available** to them from the environment at any given time.
- Rather, they **focus on specific portions of the data** that are relevant to the task at hand.
- This biological notion is referred to as that of **attention**.
- Similarly, **neural networks with attention focus on smaller portions of the data** that are relevant to the task at hand.

Attention Mechanism (cont.)

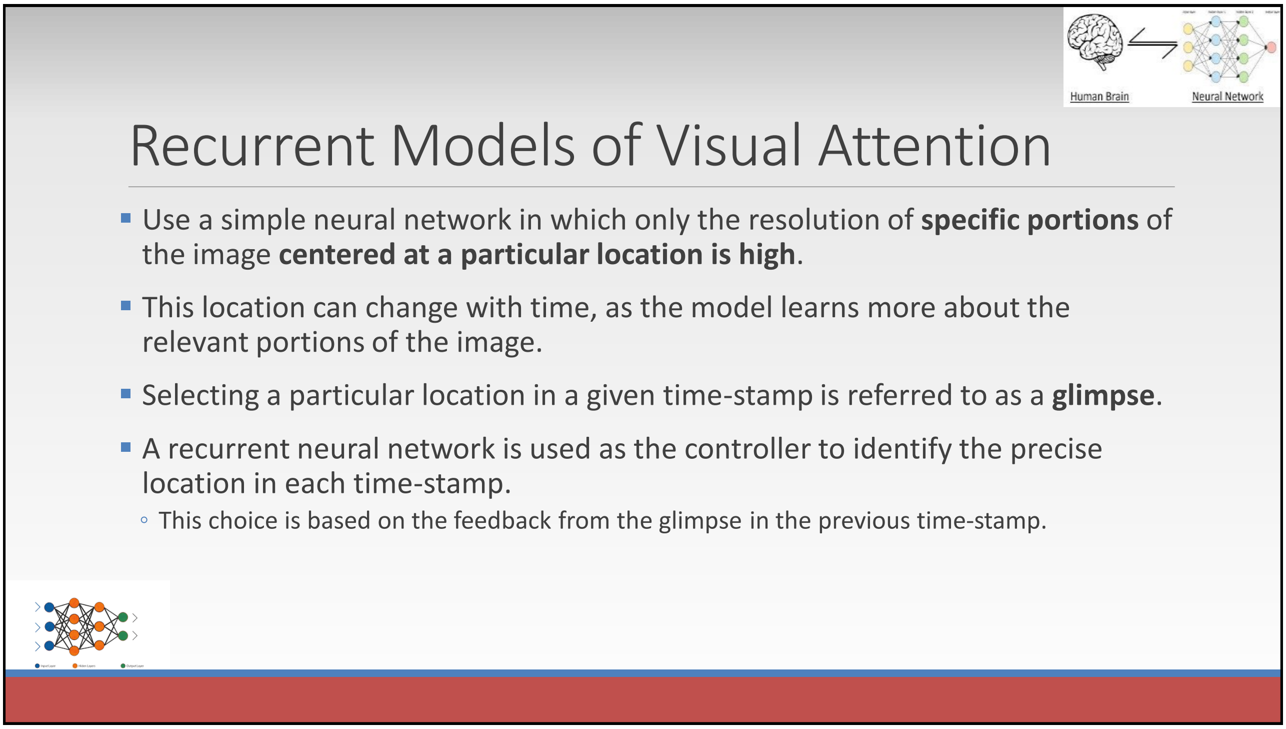
- The retina often has an image of a broader scene, although one rarely focuses on the full image.
- One pays greater *attention* to the *relevant* parts of the image
- Only a small portion of the image in the retina is carried in high resolution.


FOVEA
MACULA
MAXIMUM DENSITY OF RECEPTORS
LEAST DENSITY OF RECEPTORS
RETINA (NOT DRAWN TO SCALE)



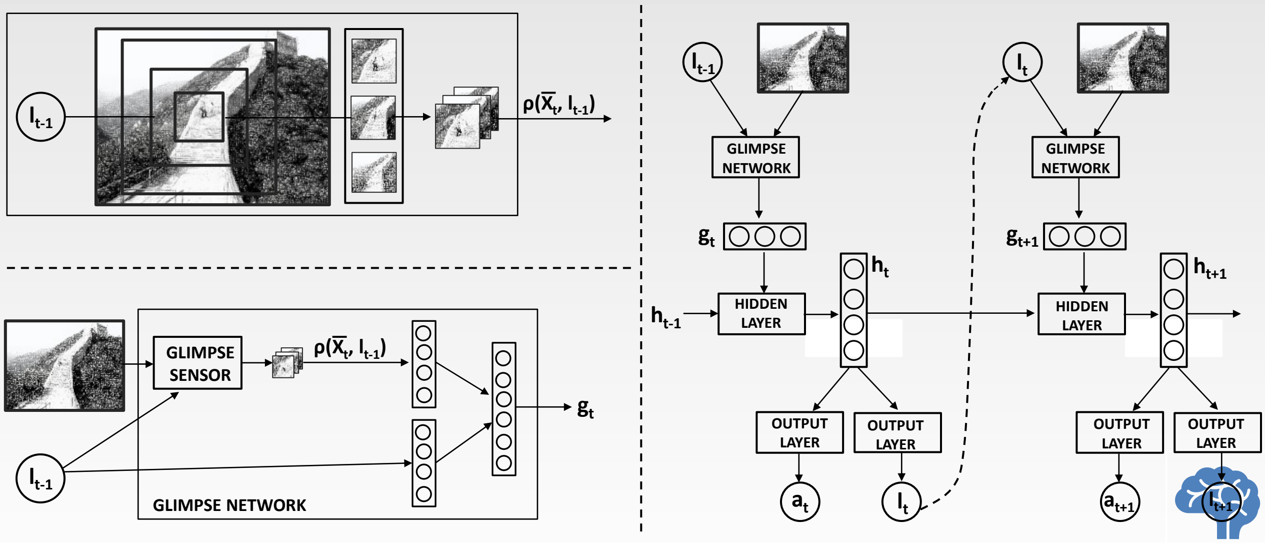
Attention Mechanism (cont.)

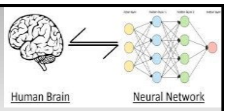
- **Hard attention** selects **specific locations**.
  - Uses **reinforcement learning** because of hard selection of locations.
- **Soft attention** gives **soft weights to various locations**.
  - More conventional models.



Recurrent Models of Visual Attention

- Use a simple neural network in which only the resolution of **specific portions** of the image **centered at a particular location is high**.
- This location can change with time, as the model learns more about the relevant portions of the image.
- Selecting a particular location in a given time-stamp is referred to as a **glimpse**.
- A recurrent neural network is used as the controller to identify the precise location in each time-stamp.
  - This choice is based on the feedback from the glimpse in the previous time-stamp.





Recurrent Models of Visual Attention (cont.)

- **Glimpse Sensor**: Creates a retina-like representation $\rho(X_t, l_{t-1})$ of the image X_t based on location l_{t-1}.
  - The **glimpse** sensor is conceptually assumed to **not have full access to the image** (because of band width constraints), and is able to access only a small portion of the image in high-resolution, which is centered at l_{t-1}.
- **Glimpse Network**: The glimpse network contains the glimpse sensor and encodes both the glimpse location l_{t-1} and the glimpse representation $\rho(X_t, l_{t-1})$ into hidden spaces.
  - Key image-processing component that is much simpler than a convolutional neural network.
- **Recurrent Neural Network**: The recurrent neural network outputs locations l_t for the next time stamp.
- **Important result**: The relatively simple glimpse network is able to **outperform a convolutional neural network because of the attention mechanism.**

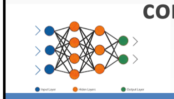
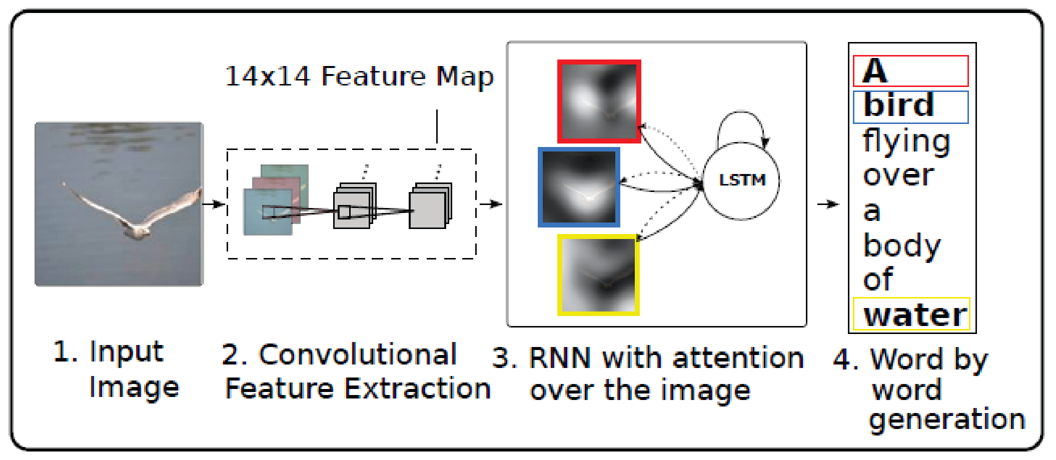
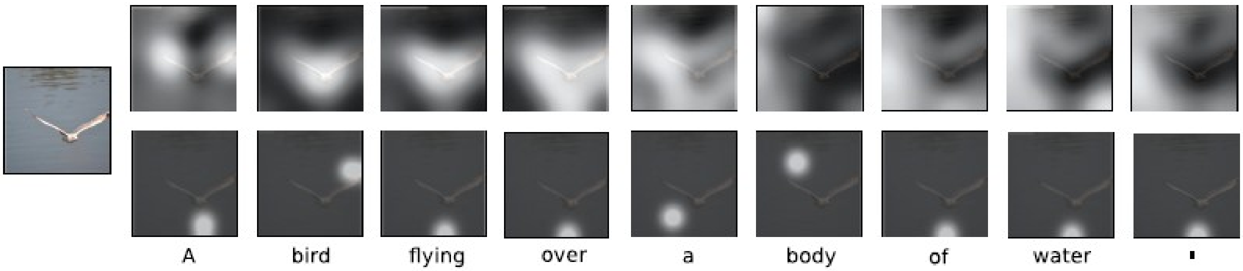


Image Captioning

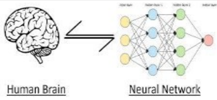
- Modified version of classification framework.
- Instead of using glimpse sensor outputting location l_t, we use L preprocessed variants centered at different positions.
- Reinforcement learning selects one of these L actions (discrete output).
- The output at the next time stamp is the subsequent word in the image caption.
- Reward based on caption prediction accuracy.


14x14 Feature Map
LSTM
A
bird
flying
over
a
body
of
water
1. Input Image
2. Convolutional Feature Extraction
3. RNN with attention over the image
4. Word by word generation

K. Xu *et al.* Show, attend, and tell: Neural image caption generation with visual attention. *International Conference on Machine Learning*, 2015.

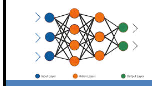


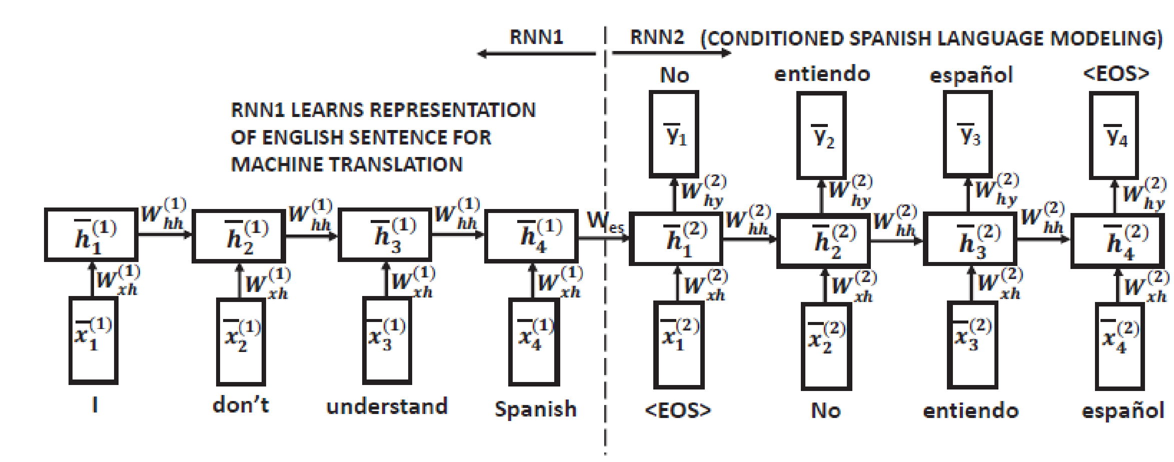
K. Xu *et al.* Show, attend, and tell: Neural image caption generation with visual attention. *International Conference on Machine Learning*, 2015.



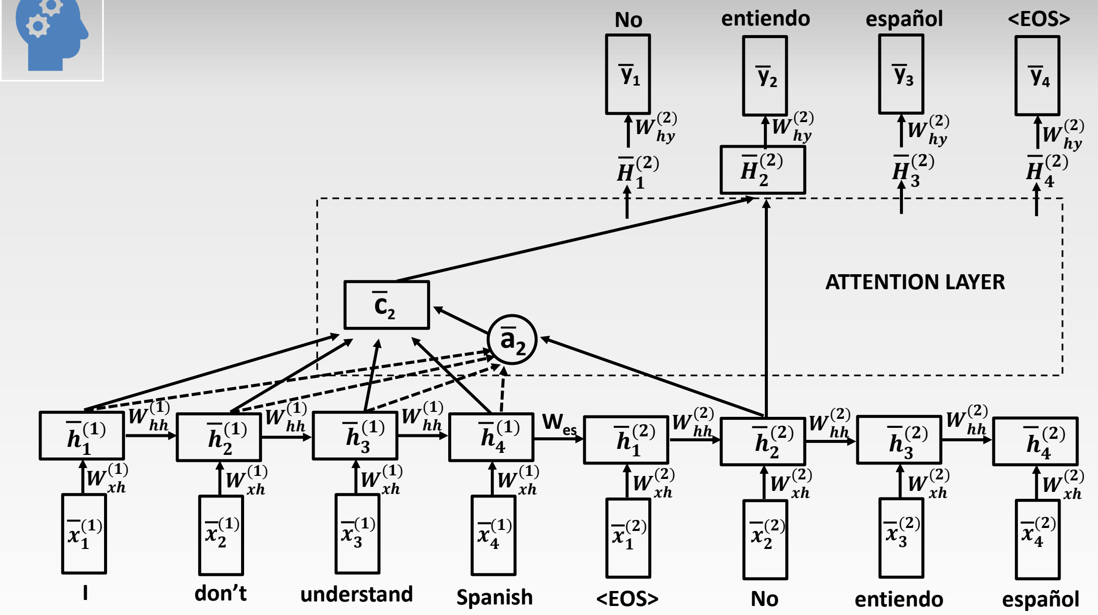
Application to Machine Translation

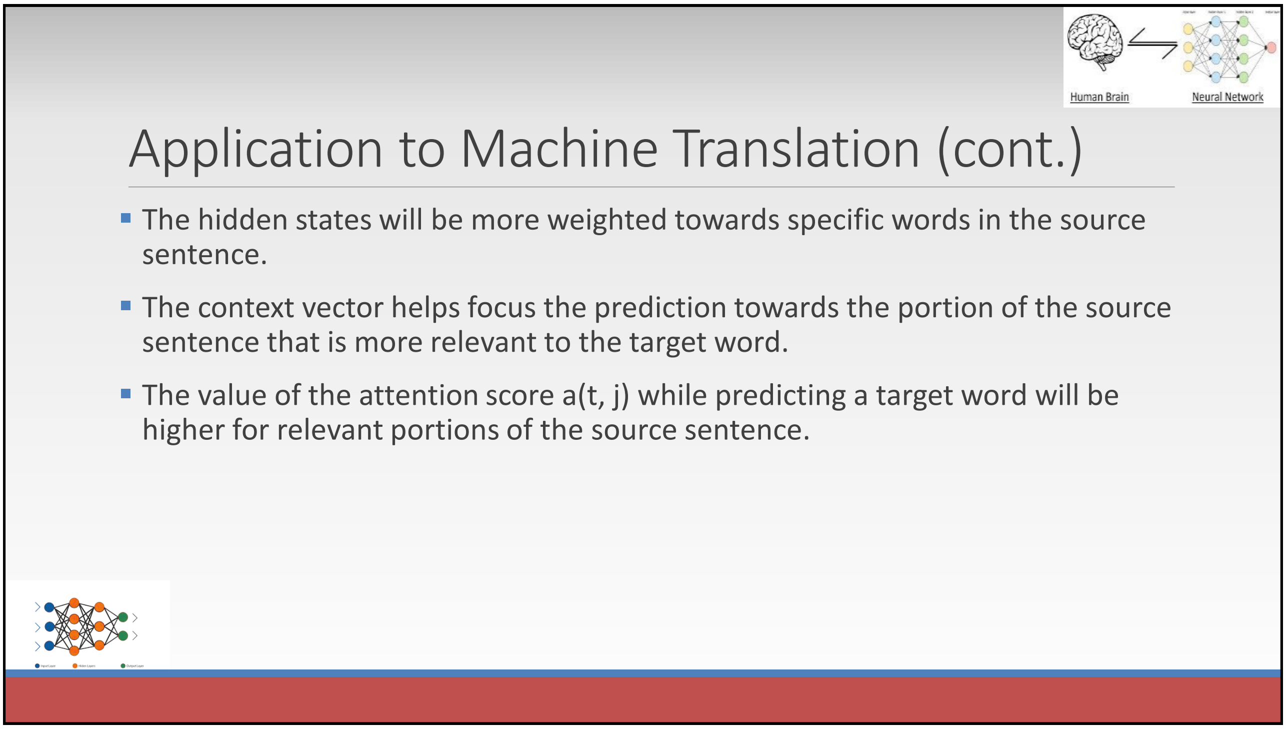
- A basic machine translation model hooks up two recurrent neural networks.
  - Typically, an advanced variant like LSTM is used.
  - Show basic version for simplicity.
- An attention model focuses on small portions of the sentence while translating a word.
- Use soft attention model in which the individual words are weighted.




RNN1
RNN2 (CONDITIONED SPANISH LANGUAGE MODELING)
No
entiendo
español
<EOS>
$\bar{y}_1$ $\bar{y}_2$ $\bar{y}_3$ $\bar{y}_4$
$W_{hy}^{(2)}$
$W_{hh}^{(1)}$
$W_{hh}^{(2)}$
W_{es}
$\bar{h}_1^{(1)}$ $\bar{h}_2^{(1)}$ $\bar{h}_3^{(1)}$ $\bar{h}_4^{(1)}$
$\bar{h}_1^{(2)}$ $\bar{h}_2^{(2)}$ $\bar{h}_3^{(2)}$ $\bar{h}_4^{(2)}$
$W_{xh}^{(1)}$
$W_{xh}^{(2)}$
$\bar{x}_1^{(1)}$ $\bar{x}_2^{(1)}$ $\bar{x}_3^{(1)}$ $\bar{x}_4^{(1)}$
$\bar{x}_1^{(2)}$ $\bar{x}_2^{(2)}$ $\bar{x}_3^{(2)}$ $\bar{x}_4^{(2)}$
I
don't
understand
Spanish
<EOS>
No
entiendo
español

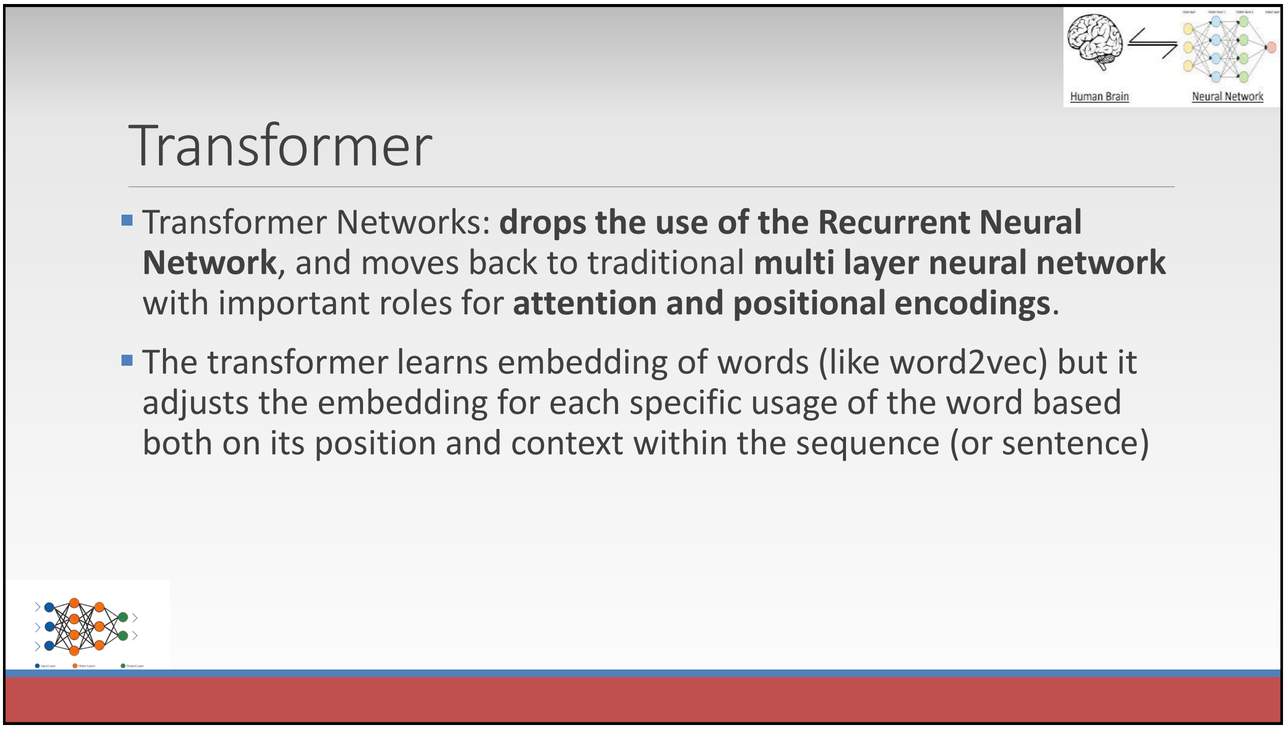
RNN1 LEARNS REPRESENTATION OF ENGLISH SENTENCE FOR MACHINE TRANSLATION


No
entiendo
español
<EOS>
$\bar{y}_1$ $\bar{y}_2$ $\bar{y}_3$ $\bar{y}_4$
$W_{hy}^{(2)}$
$\bar{H}_1^{(2)}$ $\bar{H}_2^{(2)}$ $\bar{H}_3^{(2)}$ $\bar{H}_4^{(2)}$
ATTENTION LAYER
$\bar{c}_2$
$\bar{a}_2$
$W_{hh}^{(1)}$
$W_{hh}^{(2)}$
W_{es}
$\bar{h}_1^{(1)}$ $\bar{h}_2^{(1)}$ $\bar{h}_3^{(1)}$ $\bar{h}_4^{(1)}$
$\bar{h}_1^{(2)}$ $\bar{h}_2^{(2)}$ $\bar{h}_3^{(2)}$ $\bar{h}_4^{(2)}$
$W_{xh}^{(1)}$
$W_{xh}^{(2)}$
$\bar{x}_1^{(1)}$ $\bar{x}_2^{(1)}$ $\bar{x}_3^{(1)}$ $\bar{x}_4^{(1)}$
$\bar{x}_1^{(2)}$ $\bar{x}_2^{(2)}$ $\bar{x}_3^{(2)}$ $\bar{x}_4^{(2)}$
I
don't
understand
Spanish
<EOS>
No
entiendo
español



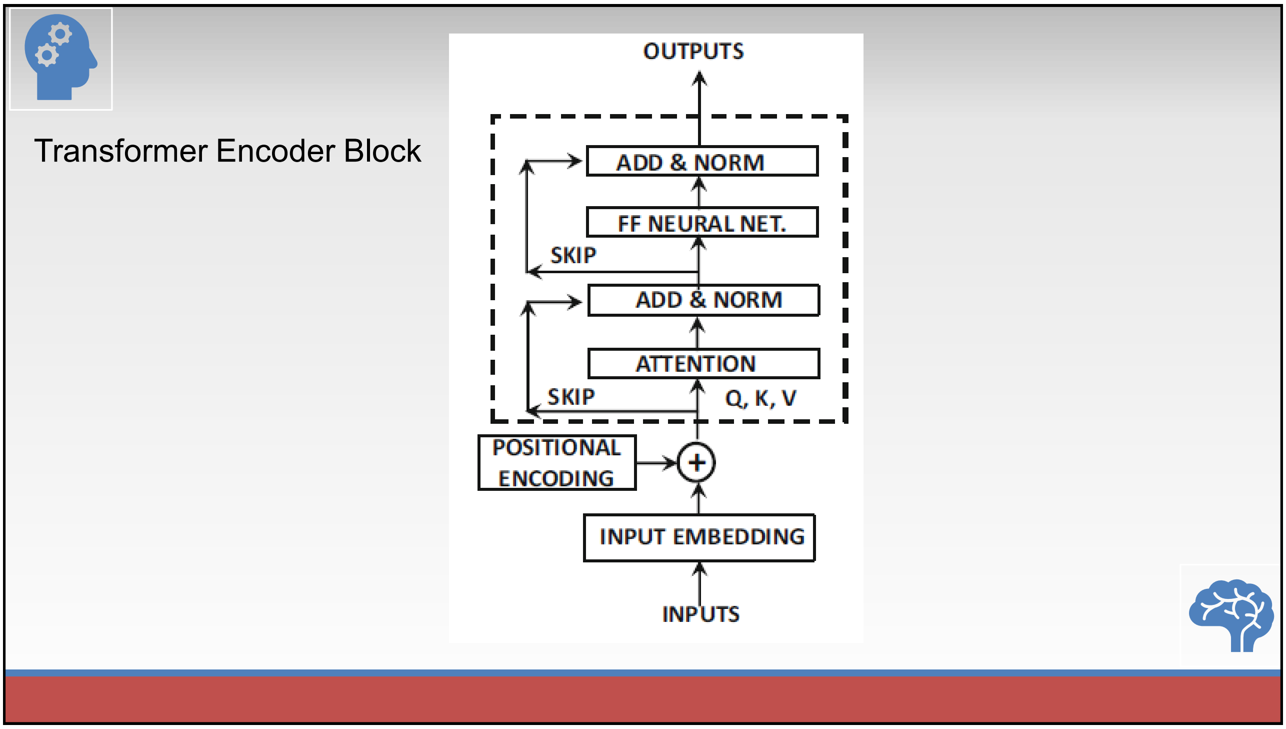
Application to Machine Translation (cont.)

- The hidden states will be more weighted towards specific words in the source sentence.
- The context vector helps focus the prediction towards the portion of the source sentence that is more relevant to the target word.
- The value of the attention score a(t, j) while predicting a target word will be higher for relevant portions of the source sentence.



Transformer

- Transformer Networks: **drops the use of the Recurrent Neural Network**, and moves back to traditional **multi layer neural network** with important roles for **attention and positional encodings**.
- The transformer learns embedding of words (like word2vec) but it adjusts the embedding for each specific usage of the word based both on its position and context within the sequence (or sentence)



Transformer Encoder Block

OUTPUTS

ADD & NORM

FF NEURAL NET.

SKIP

ADD & NORM

ATTENTION

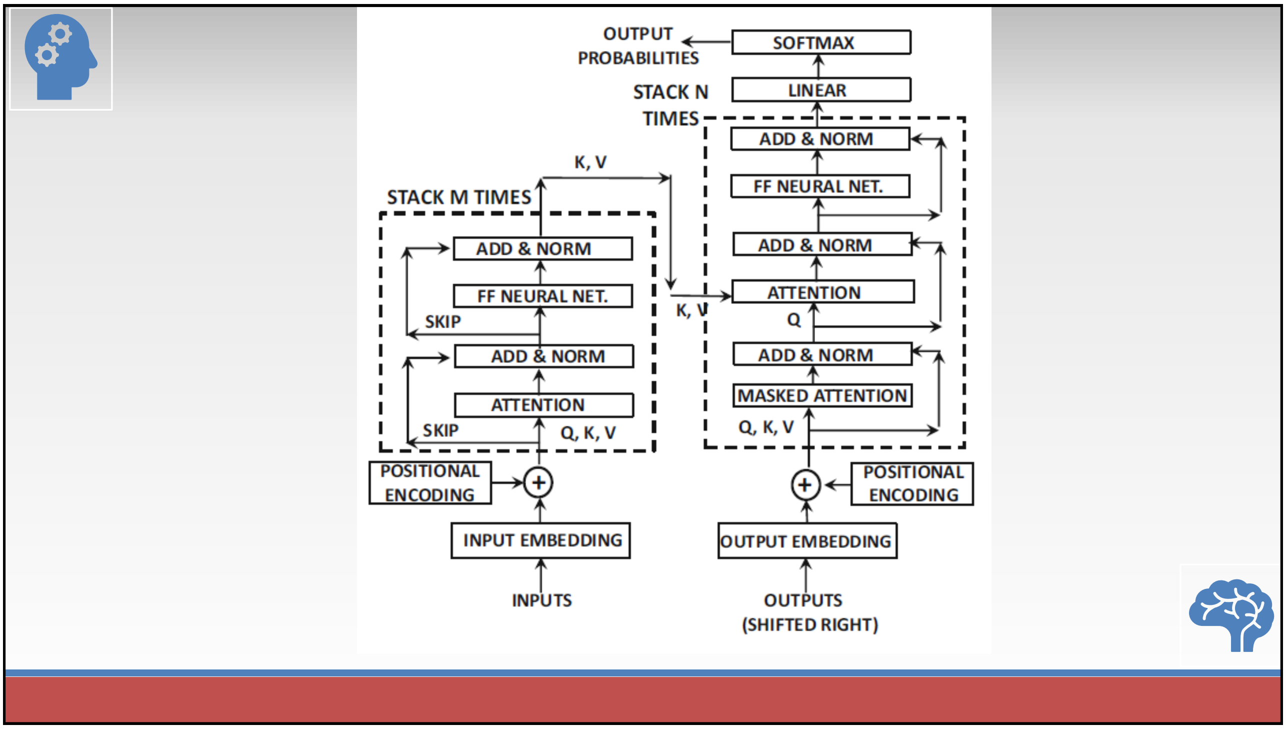
SKIP

Q, K, V

POSITIONAL ENCODING

INPUT EMBEDDING

INPUTS



OUTPUT PROBABILITIES

SOFTMAX

LINEAR

STACK N TIMES

ADD & NORM

FF NEURAL NET.

ADD & NORM

ATTENTION

Q

K, V

ADD & NORM

MASKED ATTENTION

Q, K, V

POSITIONAL ENCODING

OUTPUT EMBEDDING

OUTPUTS (SHIFTED RIGHT)

K, V

STACK M TIMES

ADD & NORM

FF NEURAL NET.

SKIP

ADD & NORM

ATTENTION

SKIP

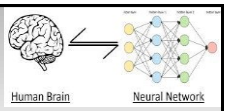
Q, K, V

POSITIONAL ENCODING

INPUT EMBEDDING

INPUTS

Thank You



References

- Aggarwal, Charu C. "Neural networks and deep learning." Springer.
- Huawei HCIA-AI