The Practical Machine Learning
PIRE-GEMADARC Lecture Series

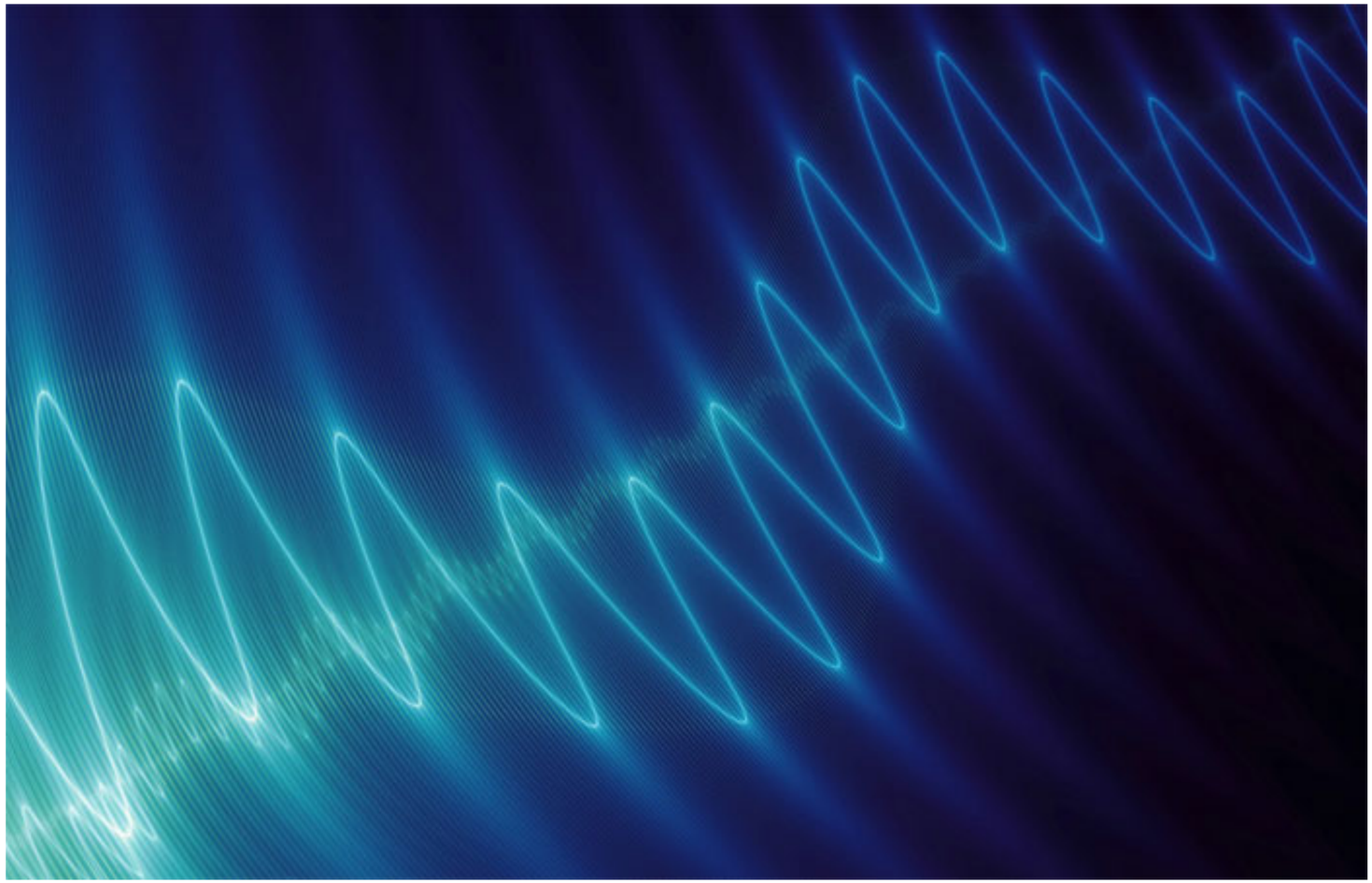
Lecture III: Time Series I

AOBO LI, 7/29/21

Office Hour After Lecture (11:00AM to 12:00 AM EST)

Outline

- What is time series?
- Traditional algorithm for time series:
  - Recurrent Neural Network
  - Long Short Term Memory
- Attention based algorithm:
  - Attention
  - Why Attention Works
- Conclusion

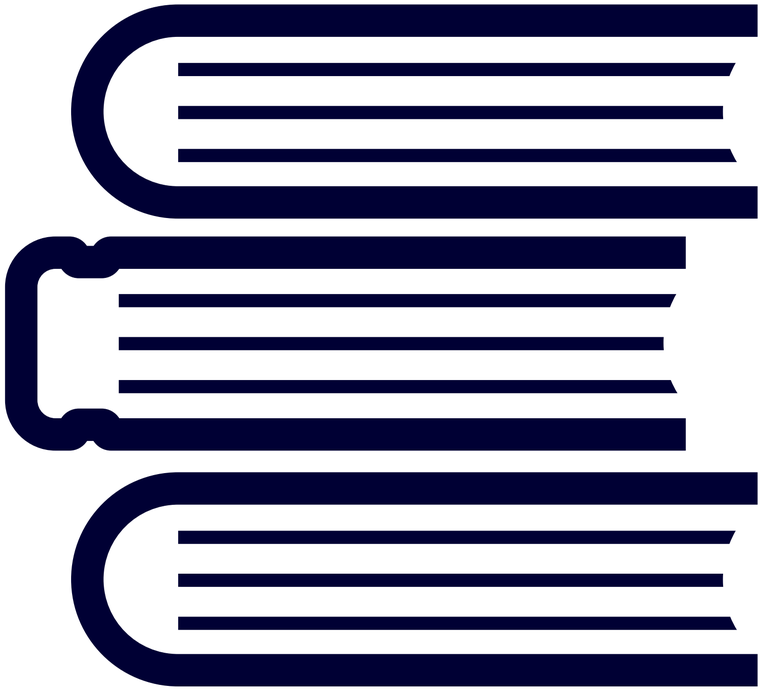


What is Time Series?

- Time series is a series of data points indexed by time
  - Data is in the form of $[t_i, \overrightarrow{x}_i]_{i=0}^{n}$
  - The length of $\overrightarrow{x}_i$ is called the number of expected features in time series
  - $\overrightarrow{x}_i$ could be scalar (one expected feature per time index) or vector (many expected features per time index)
- Most popular time series: the stock market
  - t_i is the past/current time
  - $\overrightarrow{x}_i$ could be Dow Jones/Nasdaq/Individual stock price, or a linear combination of them

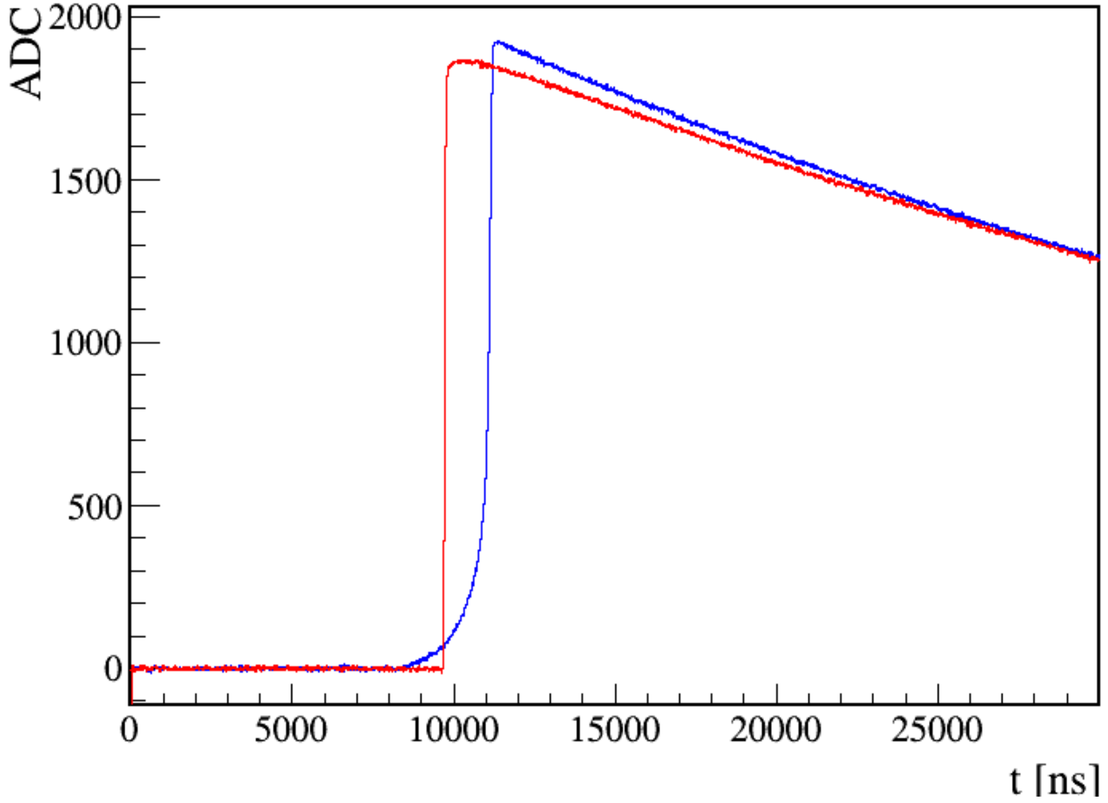
Time Series Data

- Language can also be treated as a type of time series:
  - One-hot encoding of each word:
    - I: [1 0 0 0]
    - Love: [0 1 0 0]
    - Machine: [0 0 1 0]
    - Learning: [0 0 0 1]
  - Concatenate each word into sentence and phrase
    - I love machine learning: [[1 0 0 0], [0 1 0 0], [0 0 1 0], [0 0 0 1]]
- Text data follows $[t_i, \overrightarrow{x}_i]_{i=0}^{n}$ structure
  - t_i is a bit ill-defined, but a sentence is still ordered
- The machine learning field dealing with text and sentences is called Natural Language Processing (NLP)
- Most time series algorithm can be applied to NLP and vice versa
- In this lecture, we will focus on numerical time series, and occasionally use NLP to help the understanding of concepts



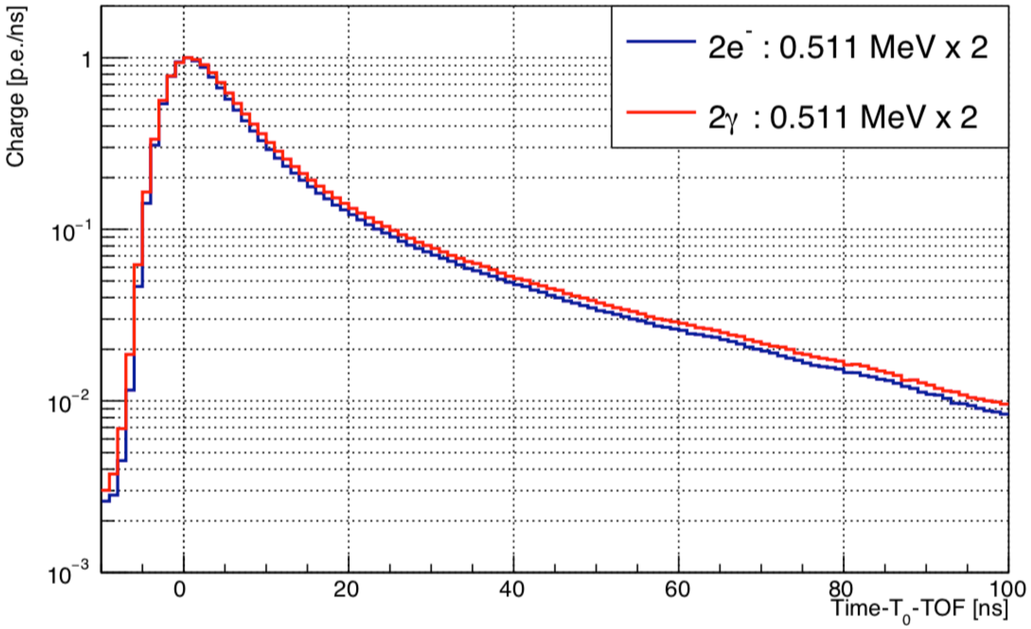
Time Series Data in Physics

- **Ge Detectors:**
  - t_i is the time sample, x_i is the corresponding ADC counts/Voltage
  - Usually we feed in waveforms one-by-one, so the number is expected features is 1
  - However, we can increase this by performing waveform reshape:
    - Equivalent to downsampling the waveform $(3000,1) \rightarrow (1000,3)$
    - Long time series are usually expensive for the RNN models we will introduce today



Time Series Data in Physics

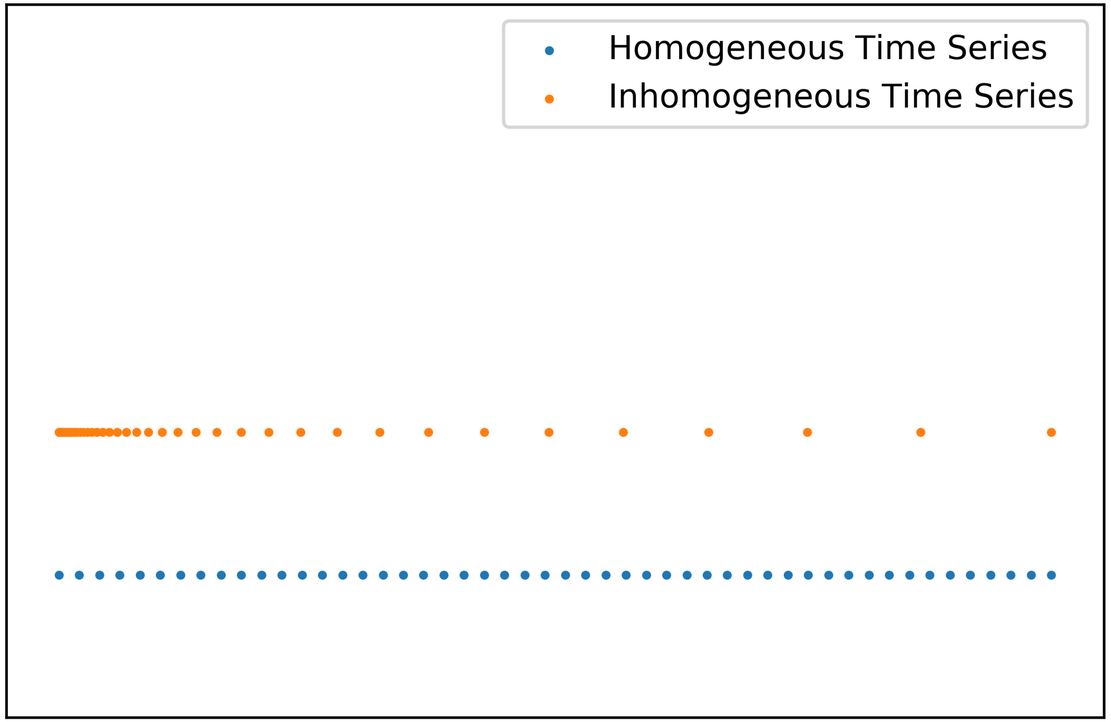
- **Liquid Scintillator/Water Cherenkov Detectors:**
  - Stacking PMT time and charge into a scintillation time profile
  - Using RNN to analyze this scintillation time profile
  - Cons: this approach gives up the positional information of all PMTs
- **LHC:**
  - Treating each particle track/jet as a ‘word’, and use RNN to backtrack the interaction within collider



A. Takeuchi, Neutrino 2020

Type of Time Series

- **Homogeneous Time Series**:
  - The interval between t_i and t_{i+1} is always the same
- **Inhomogeneous Time Series**:
  - The interval between t_i and t_{i+1} varies
- In ML, there's no particularly good way to deal with inhomogeneous time series
  - The usual approach is interpolating it into a homogeneous time series
- Thanks to the constant sampling rate, our waveform is a homogeneous time series



Two Important Features of Time Series

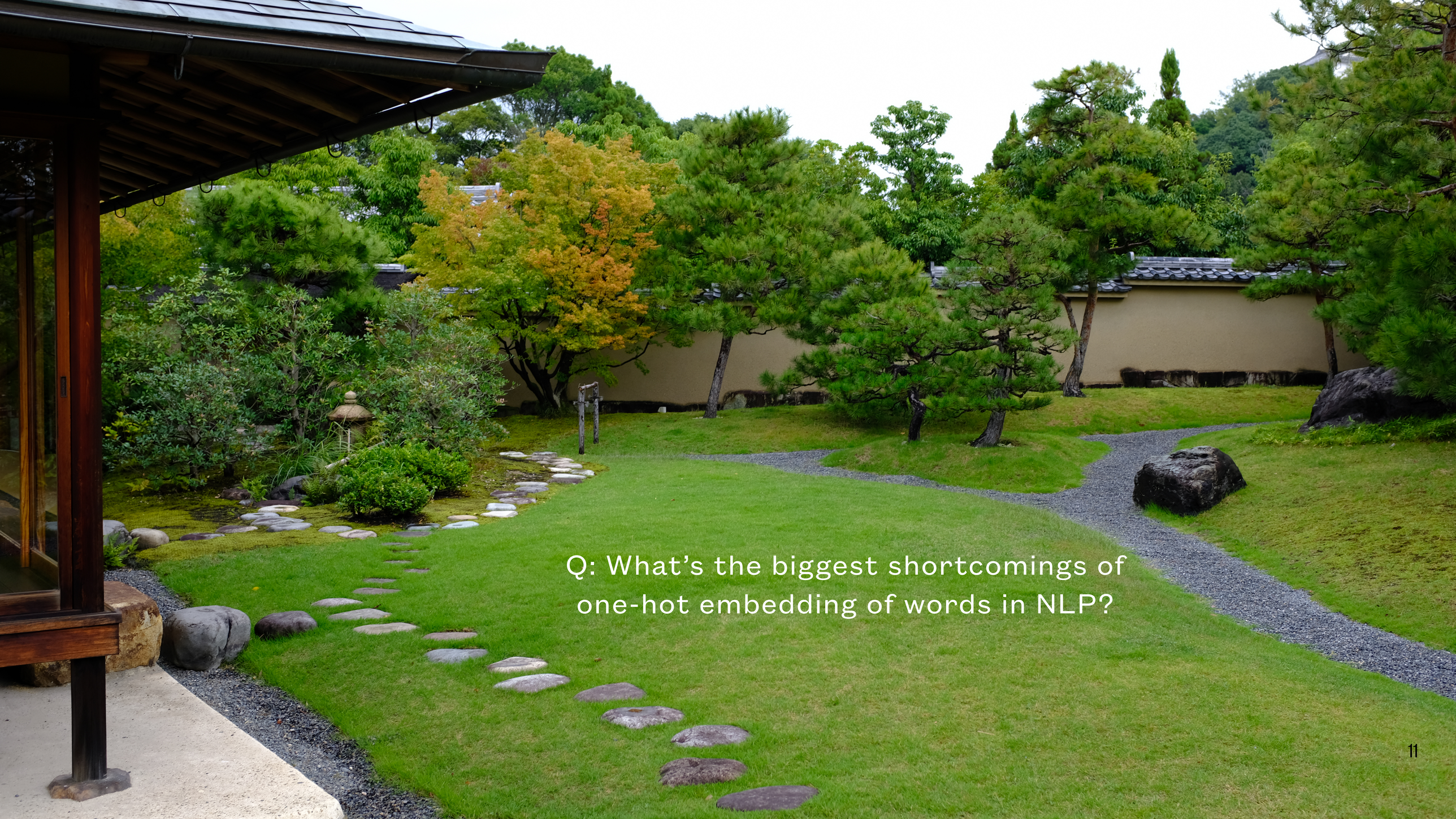
- **Order**:
  - Order contains extra information than merely the values
  - Even if two $\overrightarrow{x}_i$ contains exactly the same numbers/words, they are different if the order is different
  - Consider these 2 sentences:
    - “Woman dances well, only for a little while”
    - “For a while, only little woman dances well”

Two Important Features of Time Series

- **Long range correlations**:
  - “Love looks not with the eyes” and “blind” are at either end of the sentence, but they are highly correlated.
  - We human beings are used to handle long range correlations, but ML models are not.
- A good algorithm for time series should handle both order and long range correlations properly

LOVE LOOKS NOT WITH THE EYES, BUT WITH THE MIND; AND THEREFORE IS WINGED CUPID PAINTED BLIND.

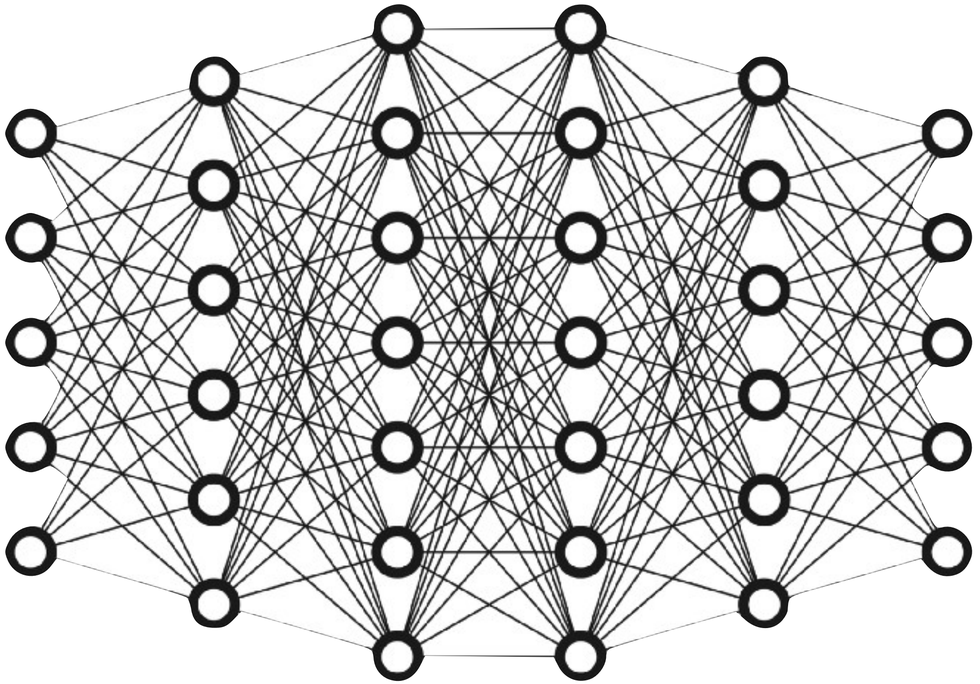
——WILLIAM SHAKESPEARE



Q: What’s the biggest shortcomings of one-hot embedding of words in NLP?

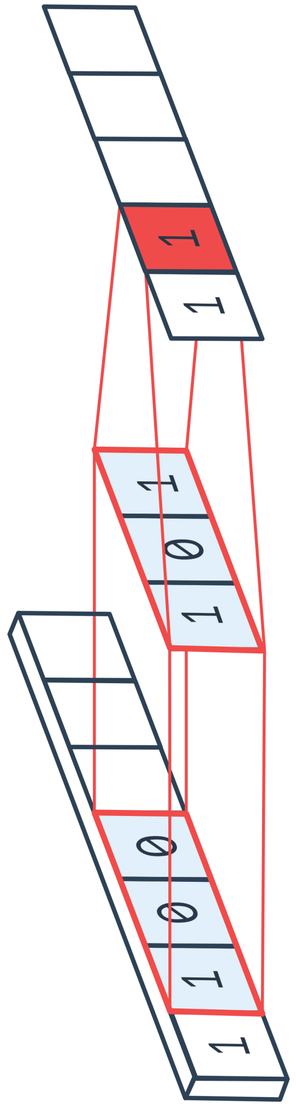
Possible Models for Time Series Data

- Fully connected neural networks:
  - For time series data, using fully connected neural network is not as bad as for image data
  - In Ge detectors, the data size is usually $(3000,1)$, which is a reasonable input for FC neural networks
  - FC networks does not particularly handle the order of data
  - It's hard to train FC network to gain long range correlations



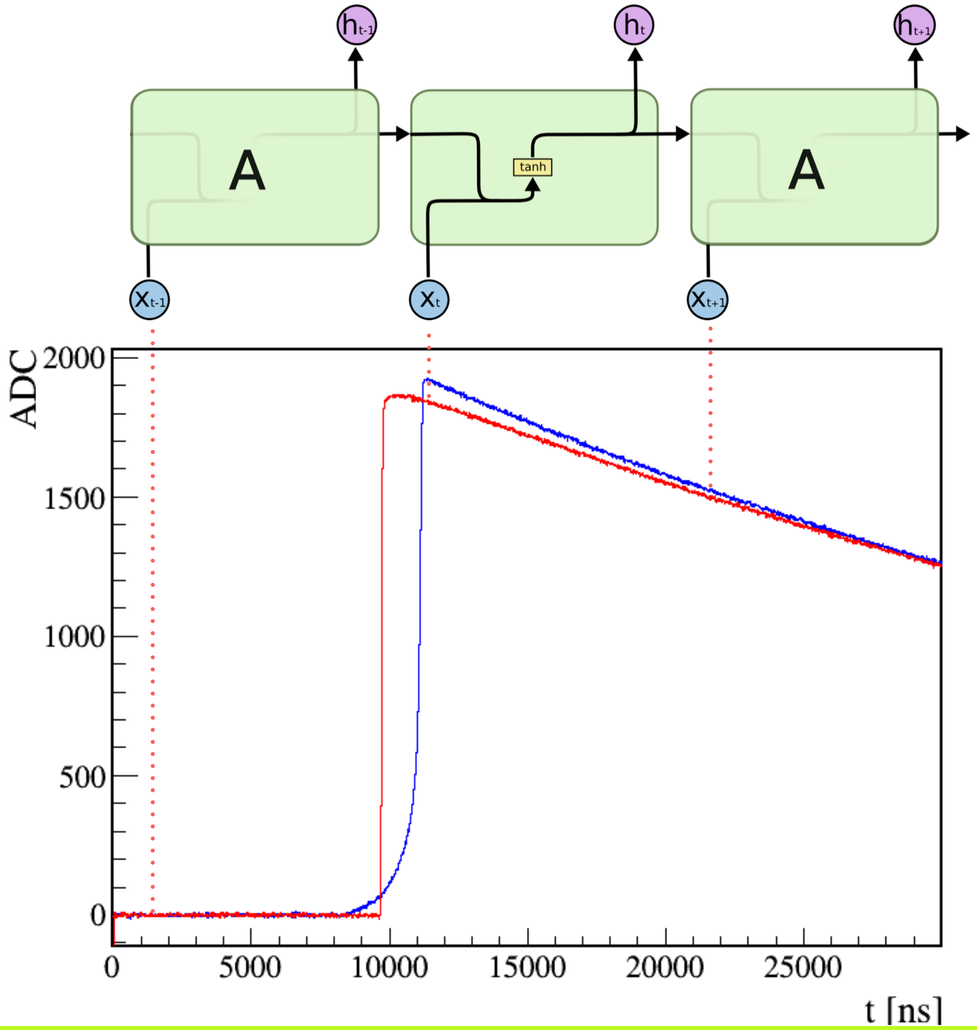
Convolutional Neural Network

- 1D convolutional neural network handles long range correlations
  - One convolutional layer calculate the spatial correlation within the filter/kernel
  - By stacking many convolutional layers (and optionally using pooling layer), we increase the reception field of convolutional layers
  - The stacked CNN will be able to probe the entire time series
  - Tips: large kernel size works well for waveform data
- CNN is robust & easy to build, but the order information is still missing
  - CNN can be enhanced to handle order information using positional encoding (discussed in Time Series II)



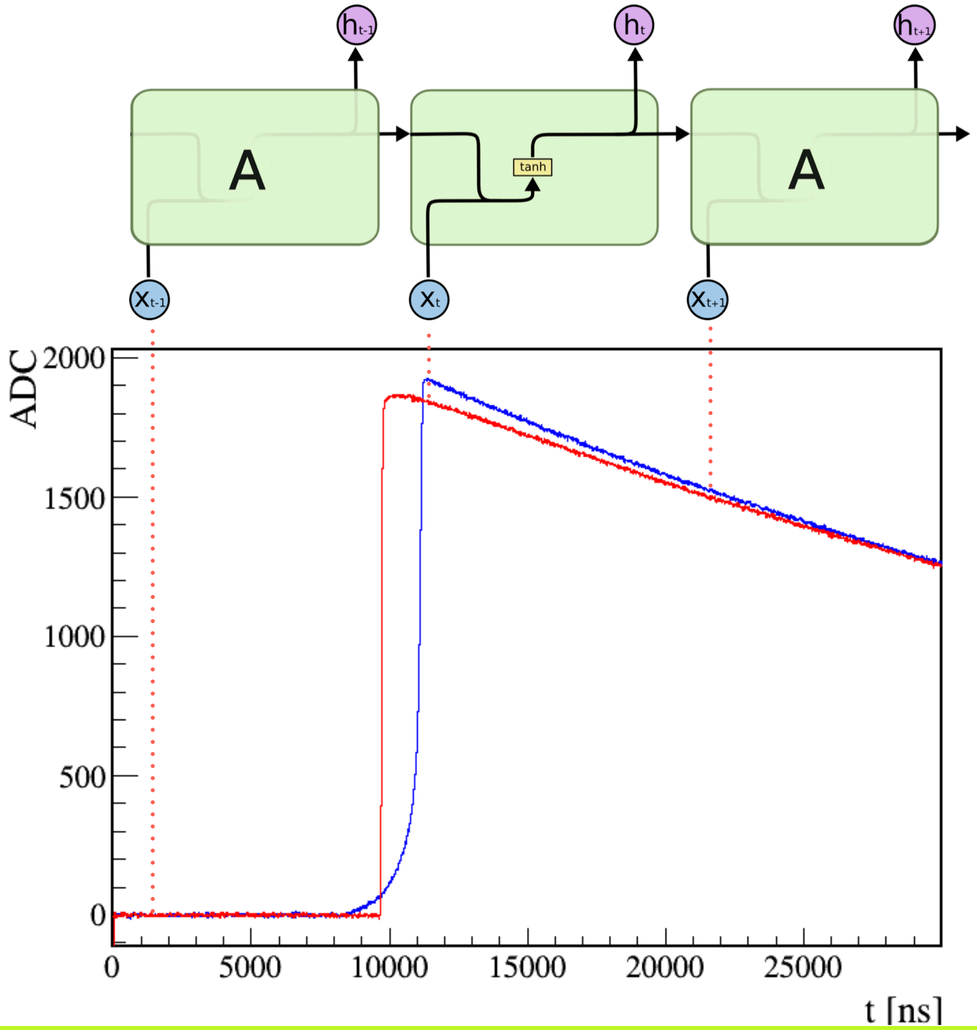
Recurrent Neural Network

- The canonical model for time series and NLP
- Each $\overrightarrow{x}_i$ of the time series $[t_i, \overrightarrow{x}_i]_{i=0}^{n}$ is fed sequentially into the recurrent unit
- Recurrent unit contains a **hidden state** $\overrightarrow{h}_i$ which stores information from all previous inputs
  - What information to store in $\overrightarrow{h}_i$ is decided by training
- The kernel of RNN is the pairs of weights W_{input}, W_{hidden}
  - W_{input} is the kernel for new input $\overrightarrow{x}_i$, and W_{hidden} is the kernel for $\overrightarrow{h}_{i-1}$



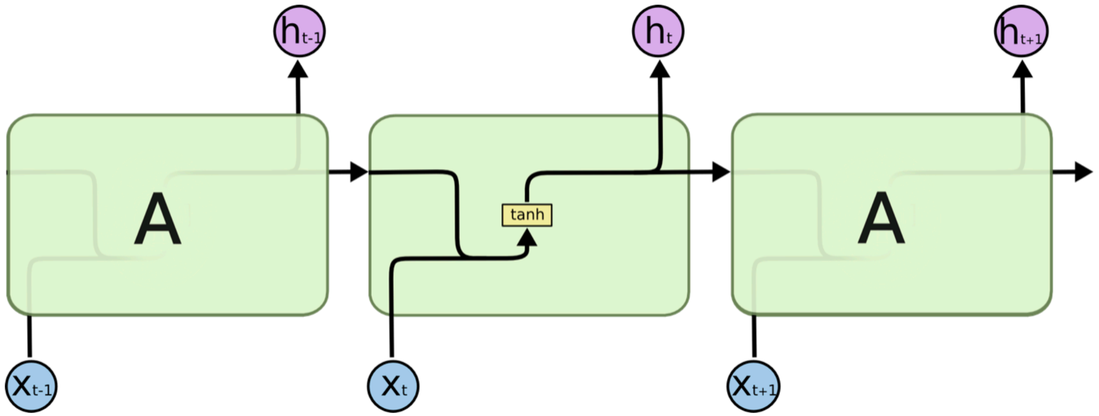
Recurrent Neural Network

- When the recurrent unit goes to the current point $\vec{x}_i$:
  - Kernel is multiplied to the current input $\vec{x}_i$ and hidden state from previous steps $\vec{h}_{i-1}$
    - $\vec{h}'_i = W_{input}\vec{x}_i + W_{hidden}\vec{h}_{i-1} + \text{Bias}$
  - Information from previous state $\vec{h}_{i-1}$ and $\vec{x}_i$ are analyzed together
  - The output $\vec{h}_i$ is produced by $\vec{h}_i = \tanh(\vec{h}'_i)$
  - $\vec{h}_i$ serves as both the intermediate hidden state output and the hidden state to the next input $\vec{x}_{i+1}$
- Iteration goes until the end of series, and the last hidden state output $\vec{h}_n$ is the output of network.



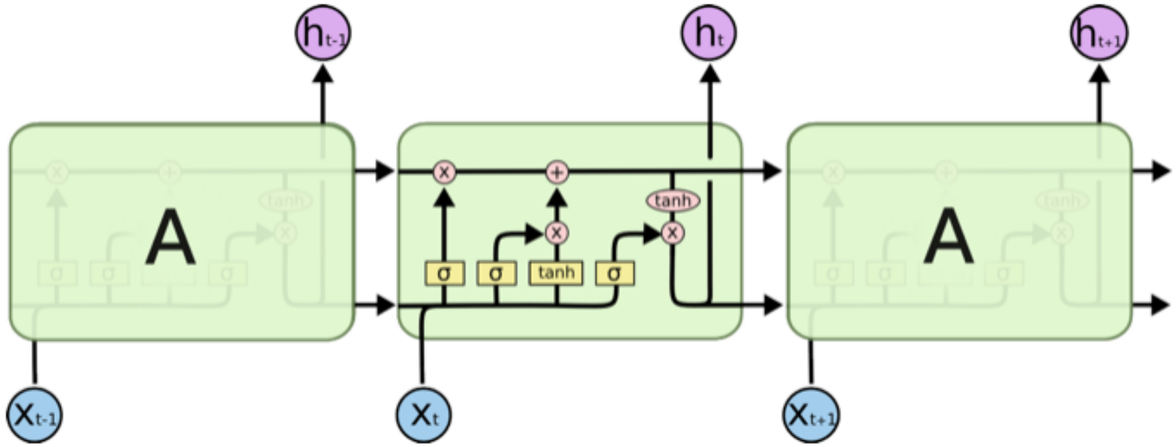
Recurrent Neural Network

- RNN is recurrent because the output of previous step is fed into the current step
- RNN handles order well, but has the following shortcomings:
  - **Long range correlation** is not handled well:
    - Technically speaking, $\overrightarrow{h}_i$ contains information from all previous $i-1$ indices
    - $\overrightarrow{h}_n$ contains information of the entire time series
    - However, longer-term information is flooded by near term information from $\overrightarrow{x}_{i-1}, \overrightarrow{x}_{i-2}$
  - **Non-parallelizable**, each step requires the input from previous step



Long Short Term Memory Network

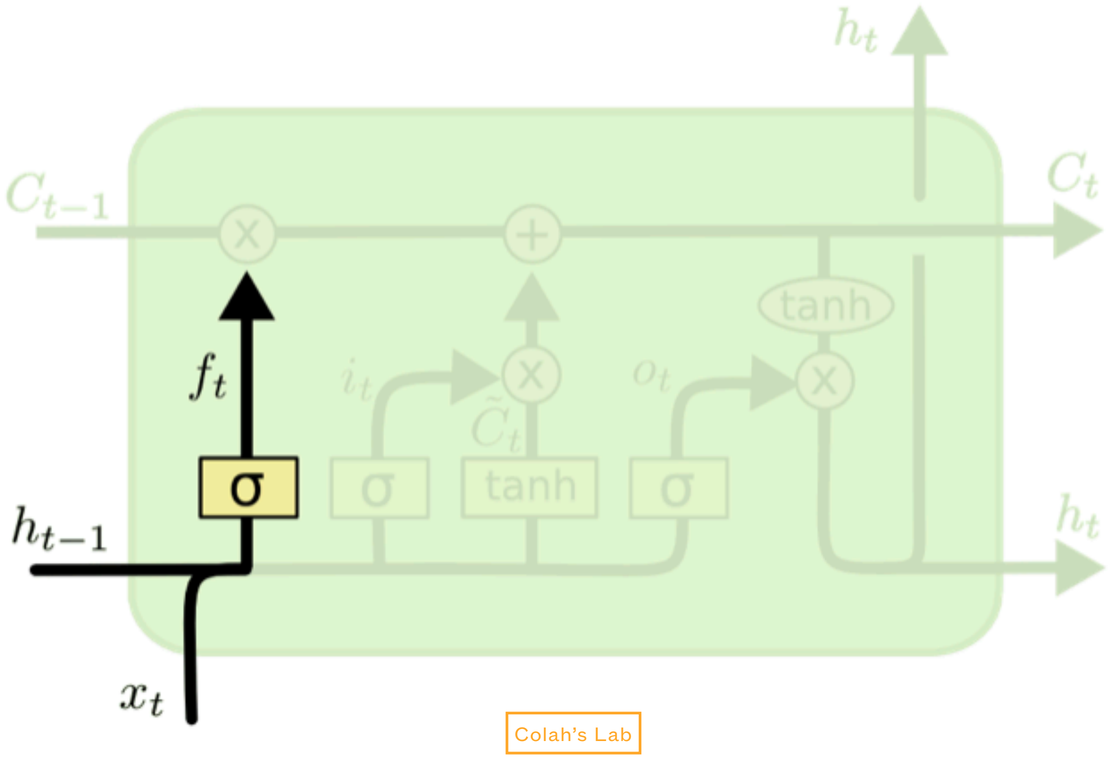
- LSTM is proposed to embody RNN with long range correlation.
- Instead of having a single hidden state $\overrightarrow{h}_i$, LSTM contain a **hidden state** $\overrightarrow{h}_i$ and a **cell state** $\overrightarrow{c}_i$
  - $\overrightarrow{h}_i$: Short term memory
  - $\overrightarrow{c}_i$.: Long term memory
- Thus, LSTM also have 4×2 kernels:
  - One pair of kernel (W_{input}, W_{hidden}) for input, forget, cell input and output gate
  - No kernel for cell state because we don't want to directly modify cell state
- Short term memory is similar to the hidden state $\overrightarrow{h}_i$ in vanilla RNN
- Long term memory is an isolated information channel
  - It only accept/reject informations indirectly through gate operation, to preserve long range correlation



Colah's Lab

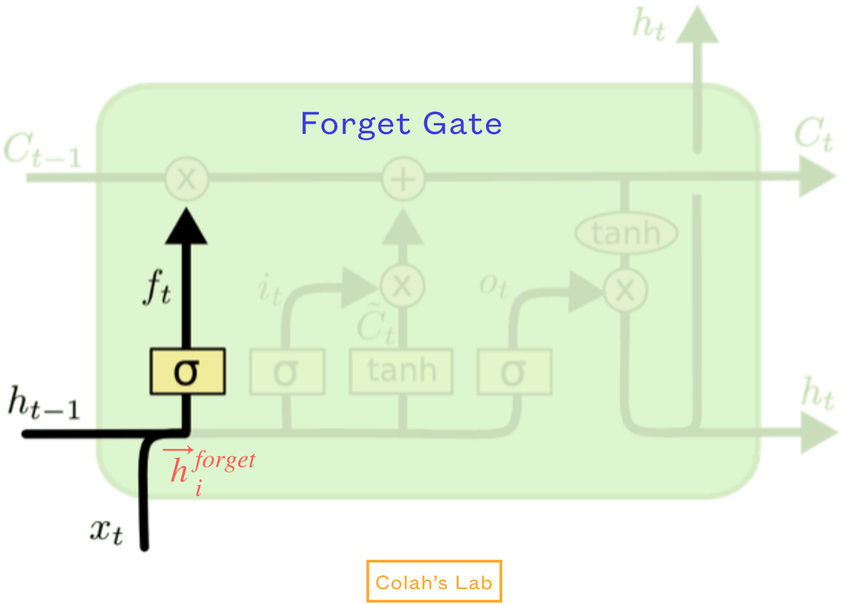
Long Short Term Memory Network

- **Gate operation** is a structure to control information flow between short and long term memories
  - The essence of gate operation is sigmoid activation function
  - Sigmoid function outputs value between [0,1]
  - Multiplying sigmoid output to preserve/reject informations:
    - If output 0, information at this position should be completely erased
    - If output 1, information at this position should be completely preserved
    - Fractional output is allowed, meaning that we want to partially preserve this information



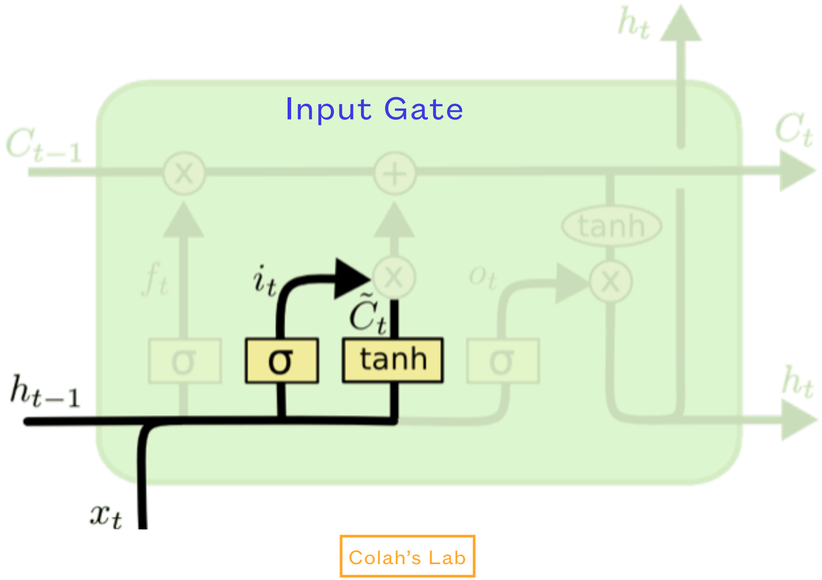
LSTM Network

- Define the following symbols:
  - $\overrightarrow{h}_i^{type} = W_{type:input}\overrightarrow{x}_i + W_{type:hidden}\overrightarrow{h}_{i-1} + \text{Bias}$
  - $\overrightarrow{h}_i^{forget} = W_{forget:input}\overrightarrow{x}_i + W_{forget:hidden}\overrightarrow{h}_{i-1} + \text{Bias}$
- **Forget Gate:**
  - In forget gate, $\overrightarrow{h}_i^{forget}$ acts as the gate to erase information from $\overrightarrow{c}_{i-1}$
  - The current input and previous hidden state jointly decides what information can be preserved/ removed in the long term memory



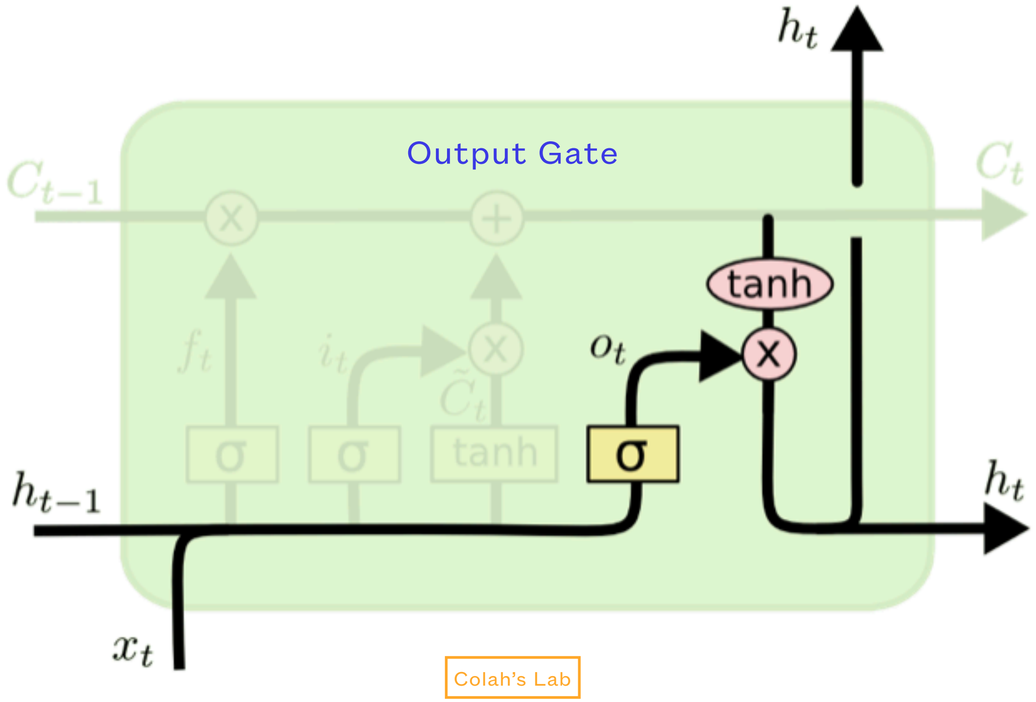
LSTM Network

- **Input Gate**:
  - In input gate, $\overrightarrow{h}_i^{input}$ acts as the gate to control $\overrightarrow{h}_i^{\text{cell input}}$
  - $\tanh \overrightarrow{h}_i^{\text{cell input}}$ produces values between -1 and 1
  - $\sigma(\overrightarrow{h}_i^{input})$ produce the gate values between 0 and 1
  - Thus, $\sigma(\overrightarrow{h}_i^{input}) \times \tanh \overrightarrow{h}_i^{\text{cell input}}$ will decides the values to add/ subtract to the cell state, at corresponding locations
  - Note that $\overrightarrow{h}_i^{input}$ and $\overrightarrow{h}_i^{\text{cell input}}$ are different since their corresponding kernels are different
  - Up to this point the cell state $\overrightarrow{c}_t$ for current time stamp has been produced. It will remain unchanged until the next recurrence



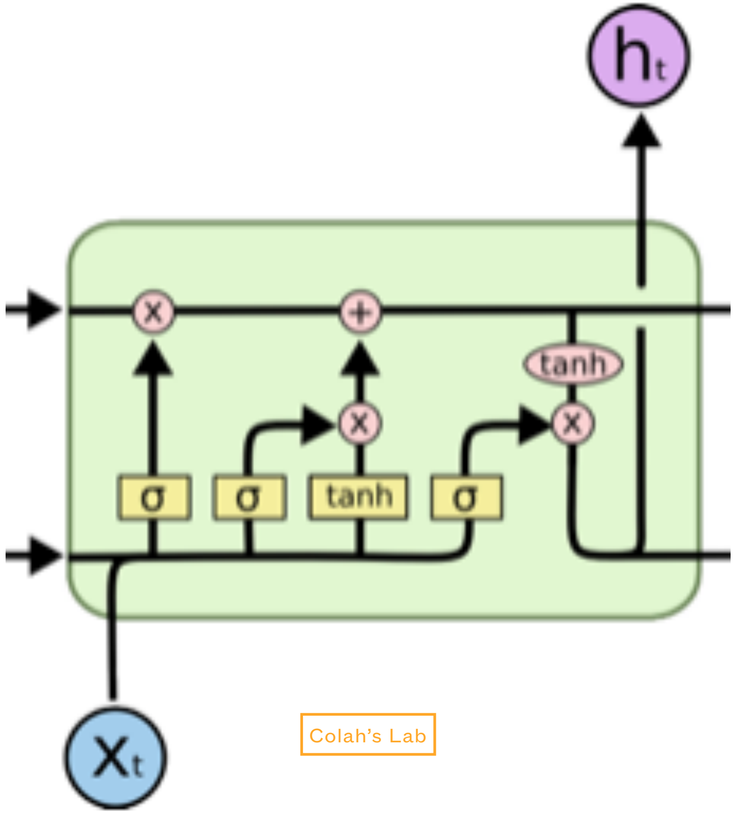
LSTM Network

- **Output Gate:**
  - The structure is similar to input gate, but now the input is calculated from the cell state
  - $\tanh \overrightarrow{c}_t$ produces values between -1 and 1
  - $\sigma(\overrightarrow{h}_i^{output})$ produce the gate values between 0 and 1
  - $\sigma(\overrightarrow{h}_i^{input}) \times \tanh \overrightarrow{h}_i^{\text{cell input}}$ will decides the values to add/subtract to the intermediate hidden state output



LSTM Network

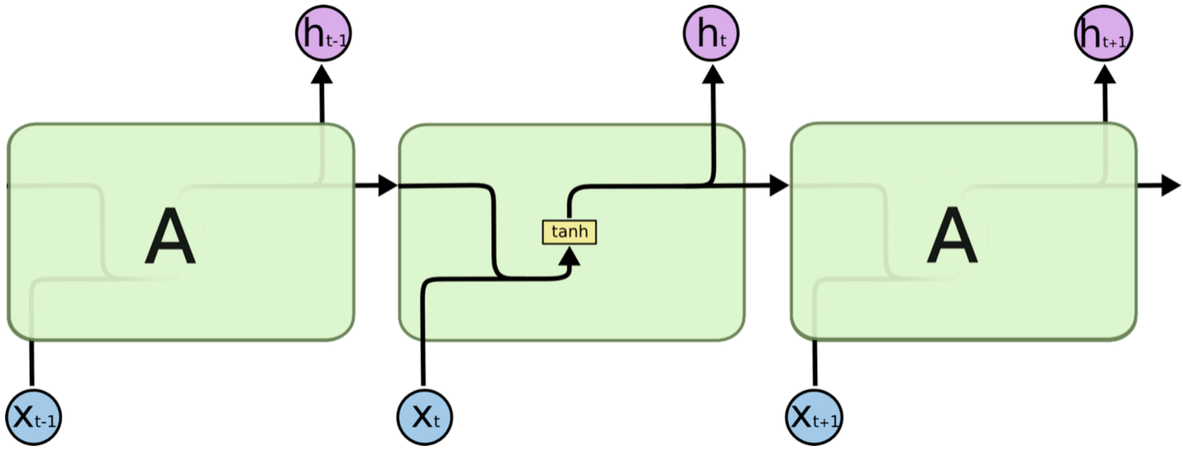
- LSTM preserve long range correlations because:
  - The cell state (long term memory) is isolated from the recurrent process, its information can only be indirectly added/removed via gate operation:
    - The forget gate throws away unimportant informations from the cell state, avoid flooding of information
    - The input gate only allow selected information to flow into cell state
  - The intermediate hidden state output is calculated from cell state, and only indirectly affected by the input/hidden state via gate operation
- For more info, I recommend this blog article:
  - https://colah.github.io/posts/2015-08-Understanding-LSTMs/




Questions?

A Quick Review of RNN

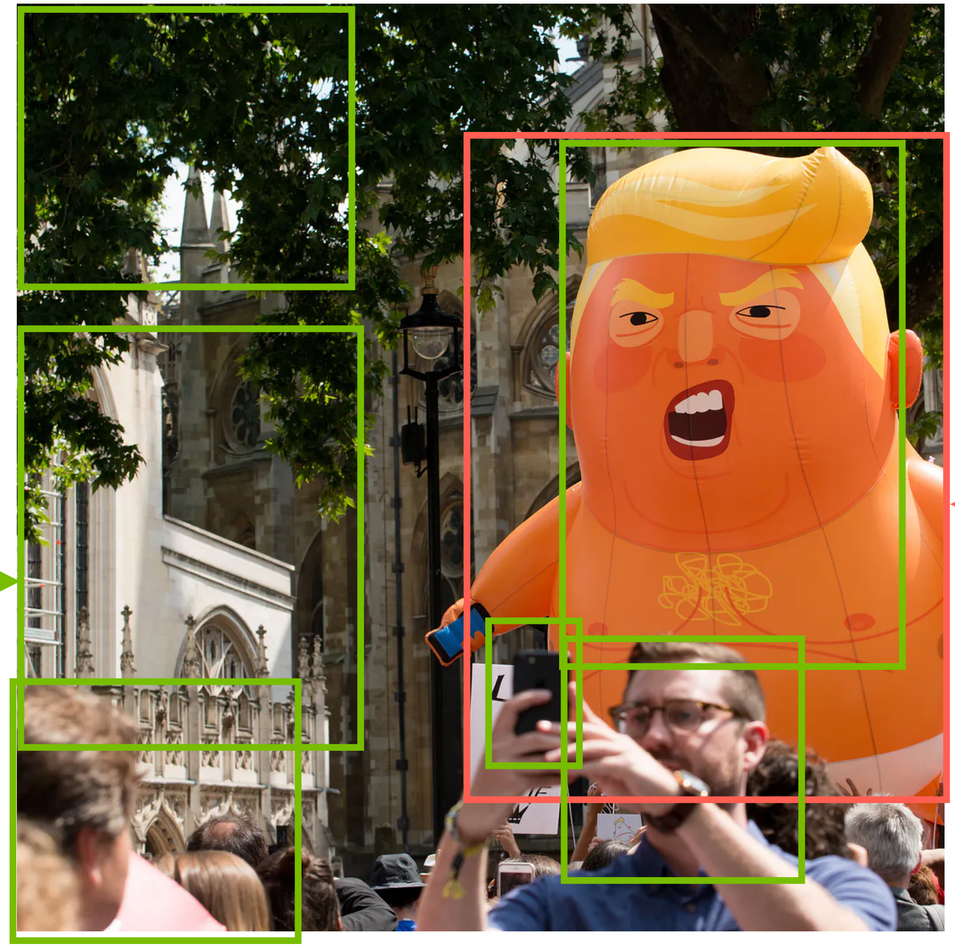
- Recurrent Neural Network (RNN) takes sequential inputs
- Exhibit a hidden state containing information from all previous steps
- Each step output an intermediate hidden state
- When moving one step forward:
  - The previous intermediate hidden state h_{t-1} and current input x_t are jointly analyzed
- The final hidden state h_n contains information from the entire time sequence
- LSTM is a more complicated version of RNN, but the structure of intermediate hidden state and final hidden state still preserve



Human vs. Machine

WHAT THE MACHINE SEE

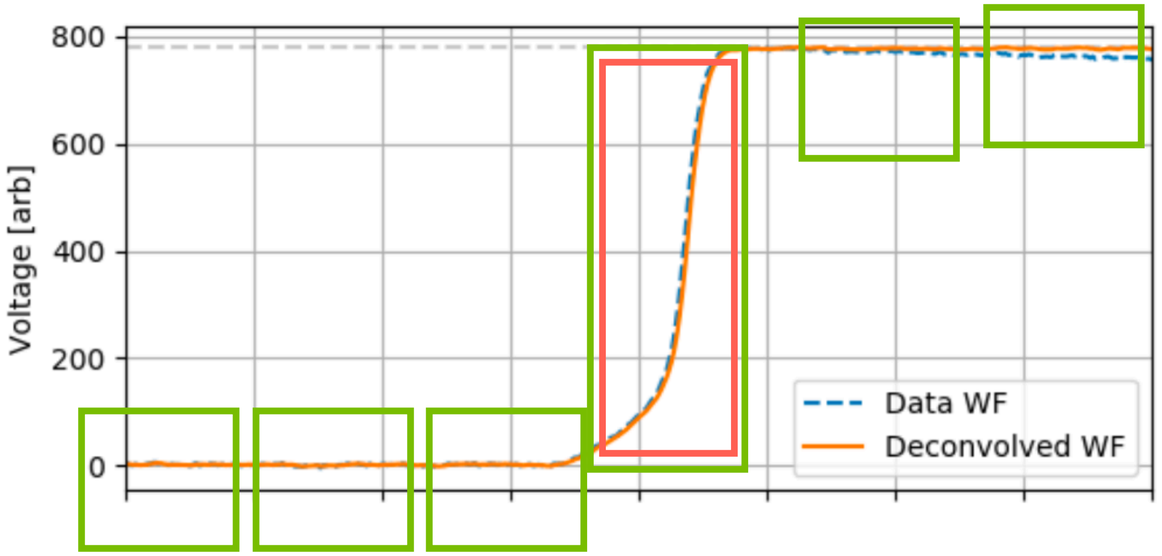
- The machine scans every single pixel of the image
- Detects many objects
- The machine does not know which object is the most important for this image, in other word, the machine does not know where to put its **Attention** on

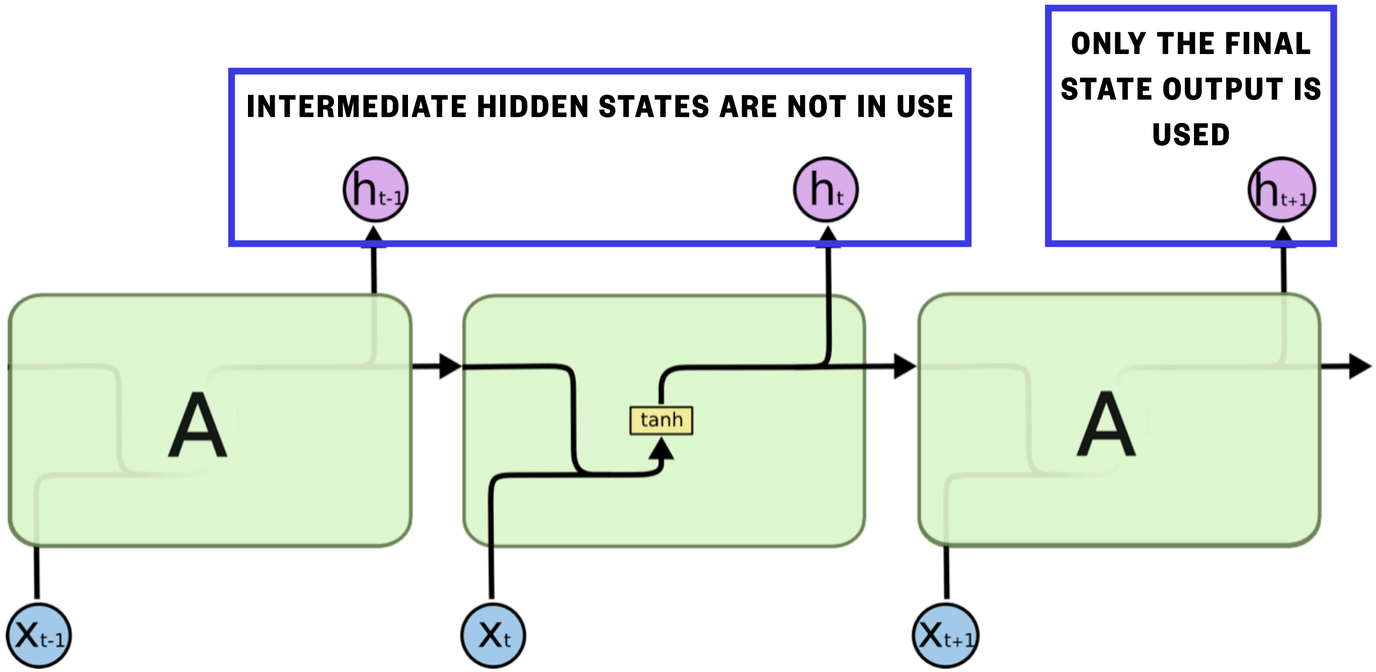


WHAT WE SEE

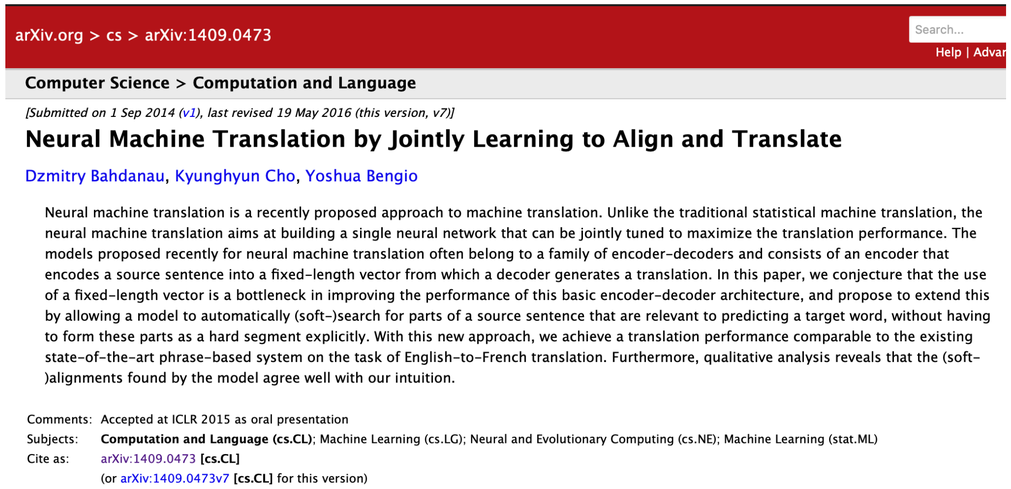
- We are “well trained” to know the theme of photo with a first glance.

Long Range Information Loss



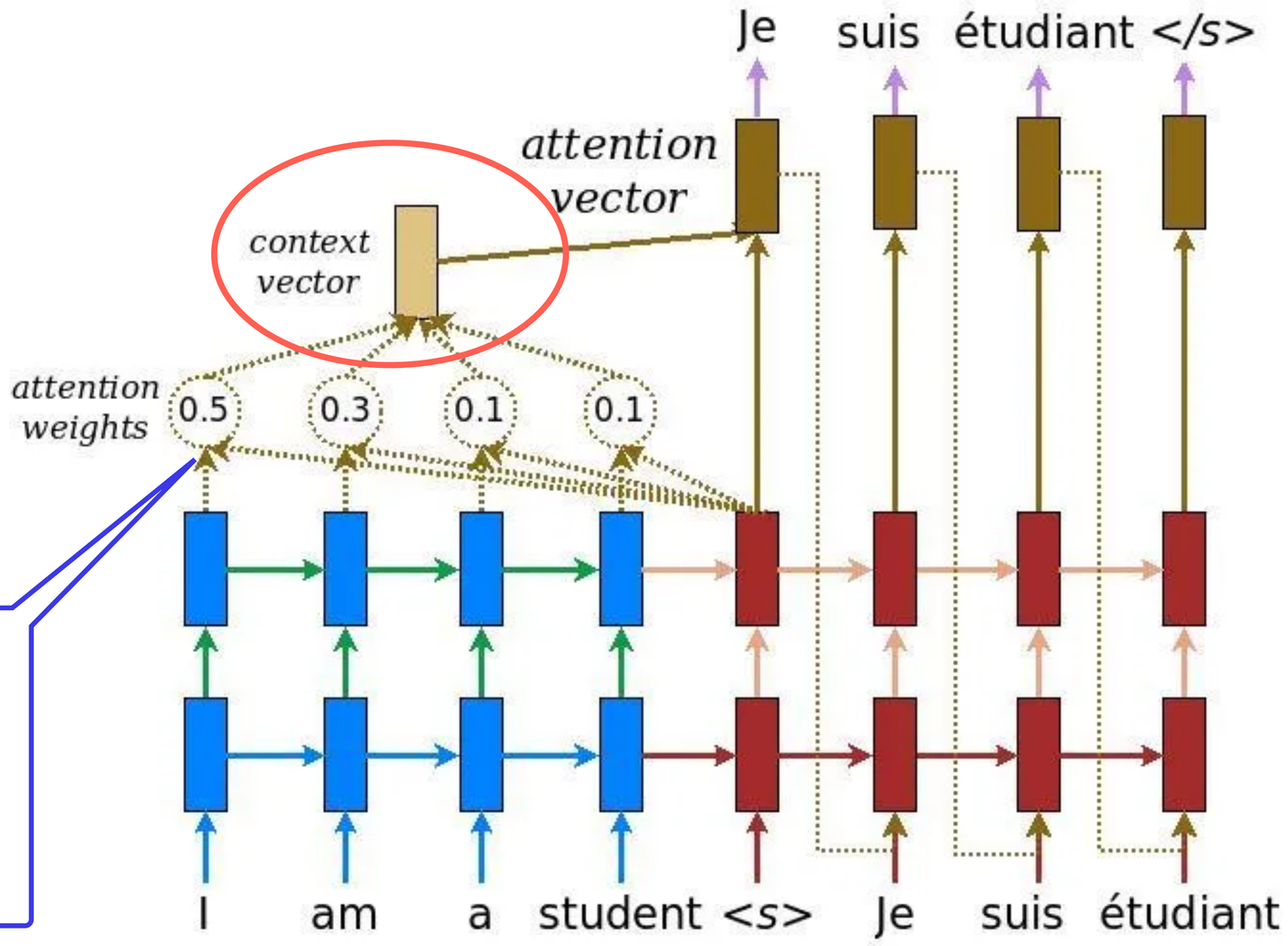


Attention Mechanism



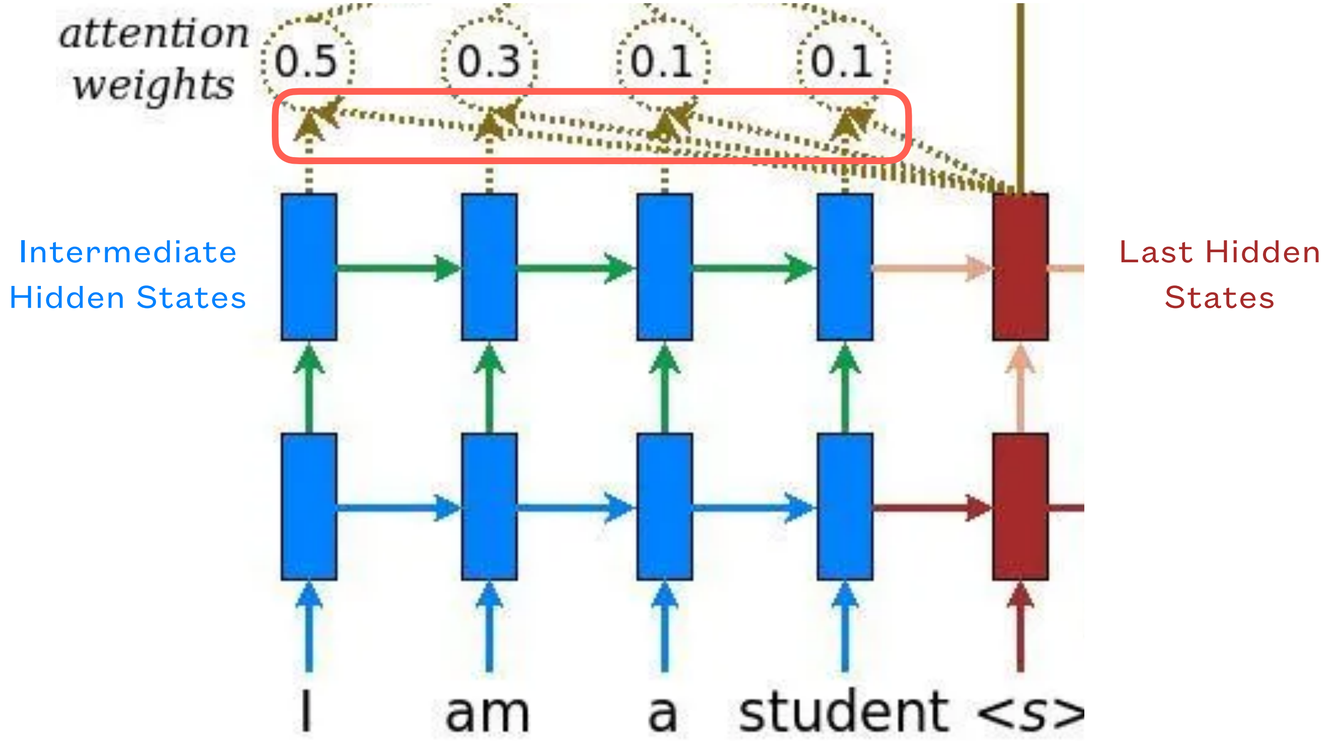
Yoshua Bengio, 2014

- Calculate the similarity of each hidden states of RNN to the final hidden state
- Different similarity metric defines different types of attention
- Weighted sum the concatenation result, softmax to produce an attention score



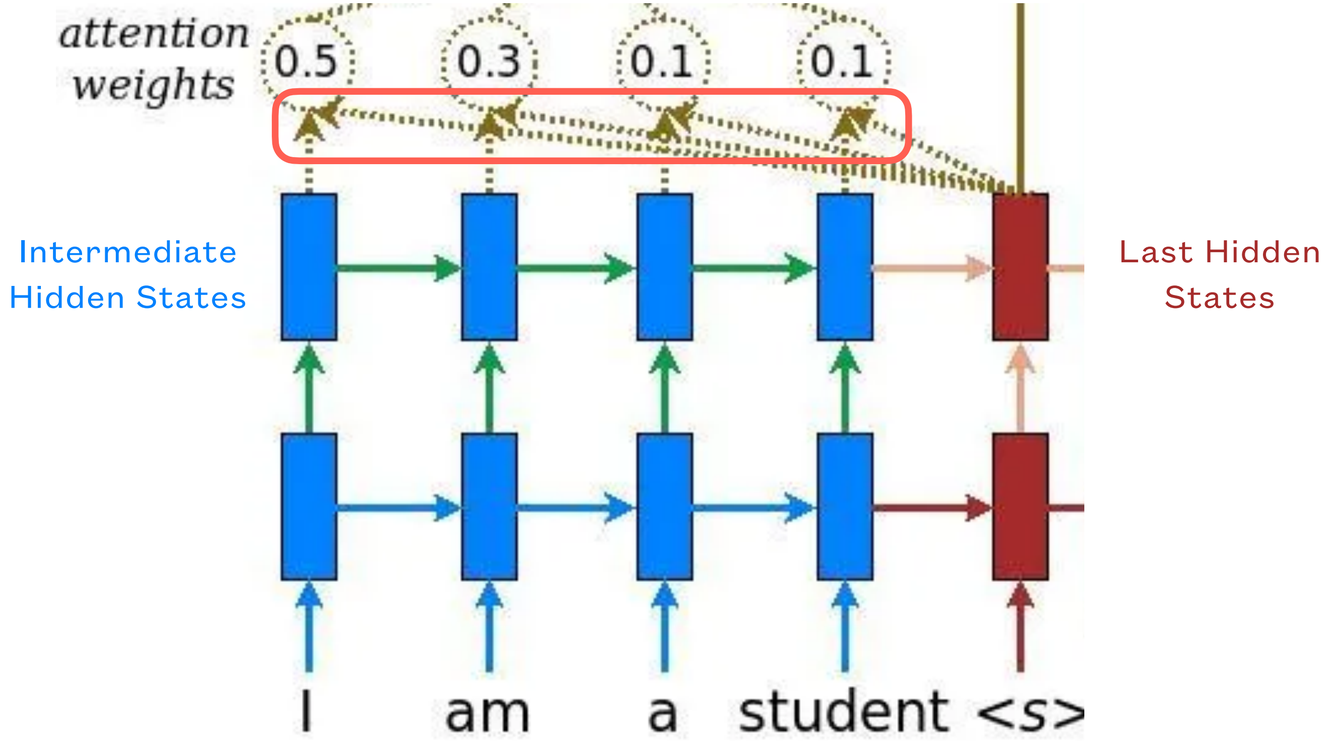
Types of Attention

- Given each hidden state vectors $\overrightarrow{h}_{i=0...n-1}$ and the final hidden state vector $\overrightarrow{h}_n$, we want to produce a scalar quantity to measure the similarity between pairs of them:
  - Dot Product Attention: $s(h_i, h_n) = h_i^T h_n$
  - Cosine Similarity: $s(h_i, h_n) = \frac{h_i^T h_n}{||h_i|| \cdot ||h_n||}$
  - Weight Kernel Concatenation:
    $s(h_i, h_n) = h_i^T W h_n$
    - W is a self-defined kernel tensor, its value is learned during training



Attention Score

- Given each hidden state vectors $\overrightarrow{h}_{i=0...n-1}$ and the final hidden state vector $\overrightarrow{h}_n$, we want to produce a scalar quantity to measure the similarity between pairs of them:
  - Dot Product Attention: $s(h_i, h_n) = h_i^T h_n$
  - Cosine Similarity: $s(h_i, h_n) = \frac{h_i^T h_n}{||h_i|| \cdot ||h_n||}$
  - Weight Kernel Concatenation:
    $s(h_i, h_n) = h_i^T W h_n$
    - W is a self-defined kernel tensor, its value is learned during training



Types of attention

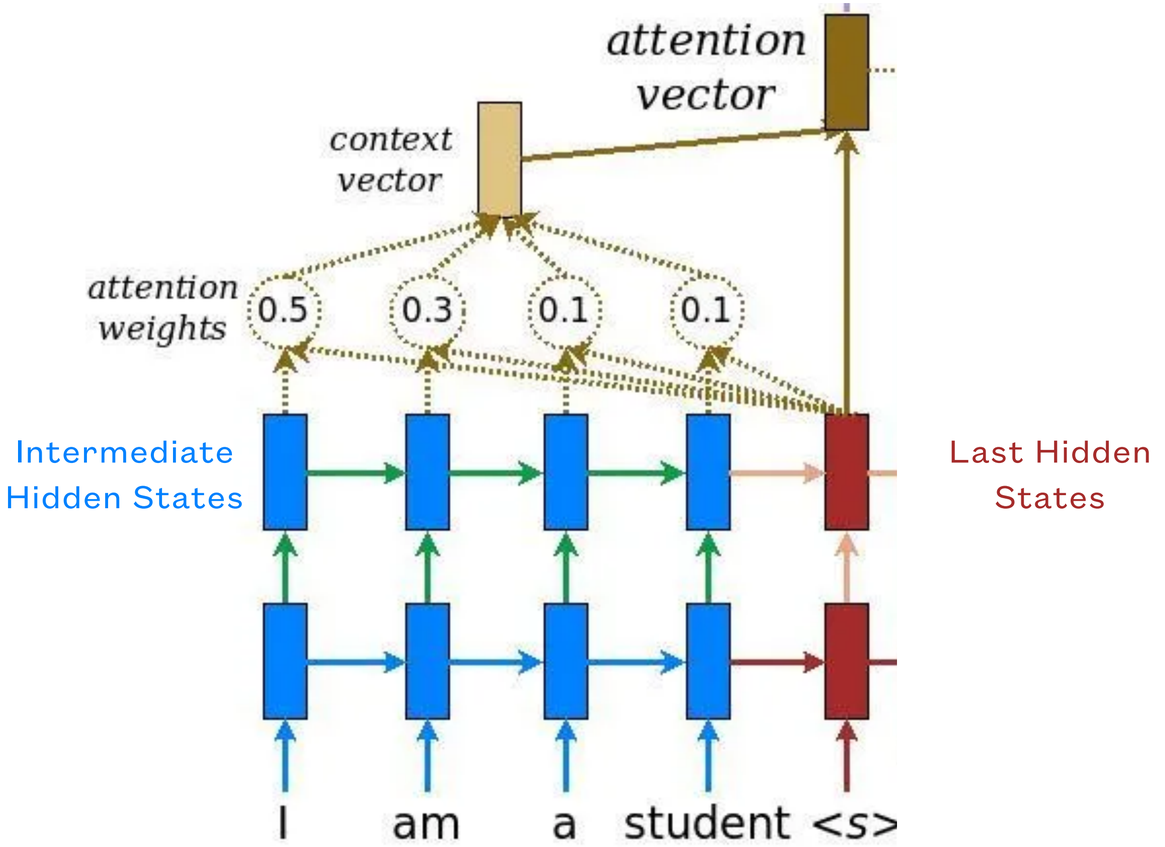
- The Attention score is produced by:

  $\vec{a} = Softmax([s_0, s_1, \ldots s_n])$

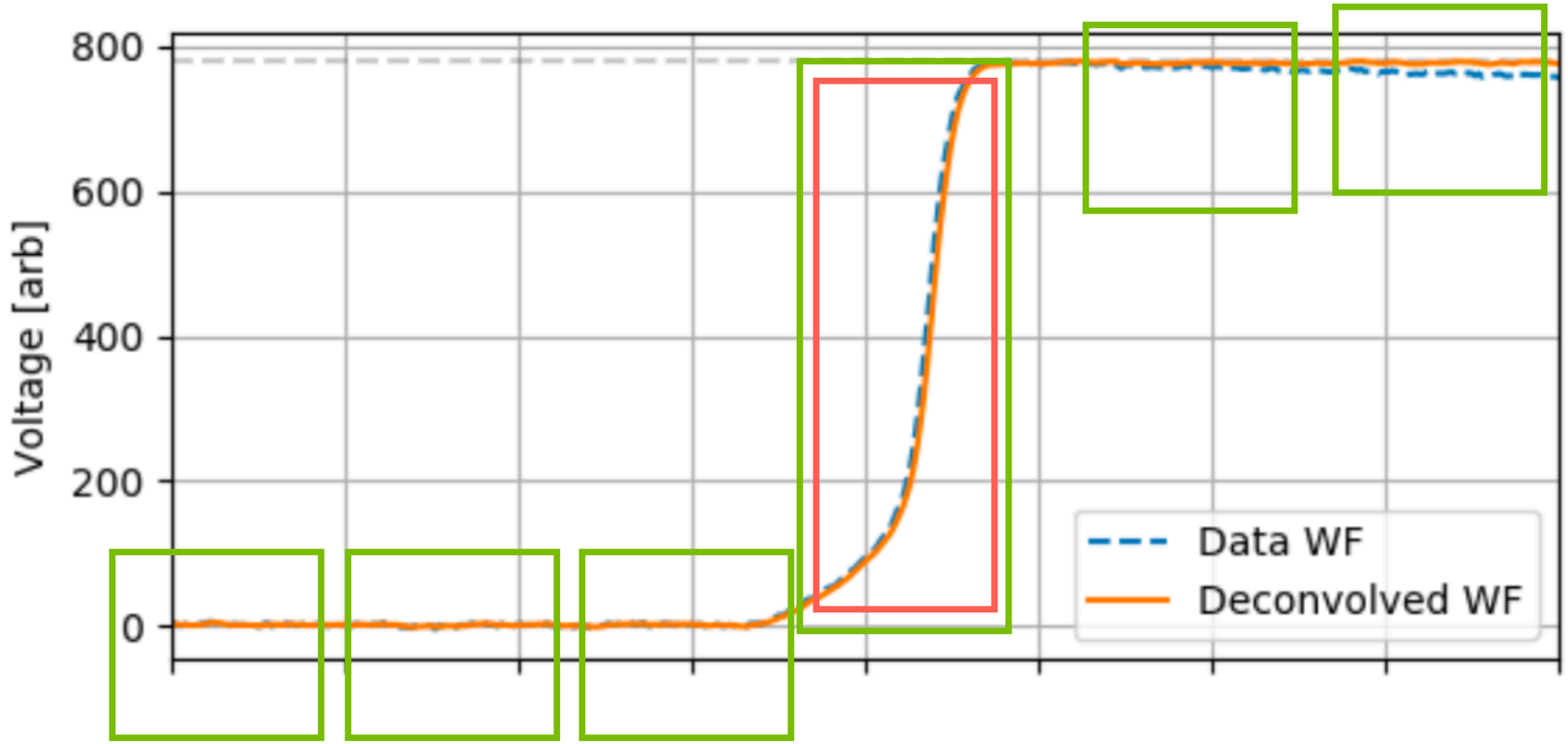
  - $\sum \vec{a} = 1$, so attention score can also be treated as a relative weight of each intermediate hidden state

- Context vector is a weighted sum of attention score and its corresponding intermediate hidden states ($\sum a_i h_i$)

- Attention vector is obtained by concatenating context vector and last hidden state together



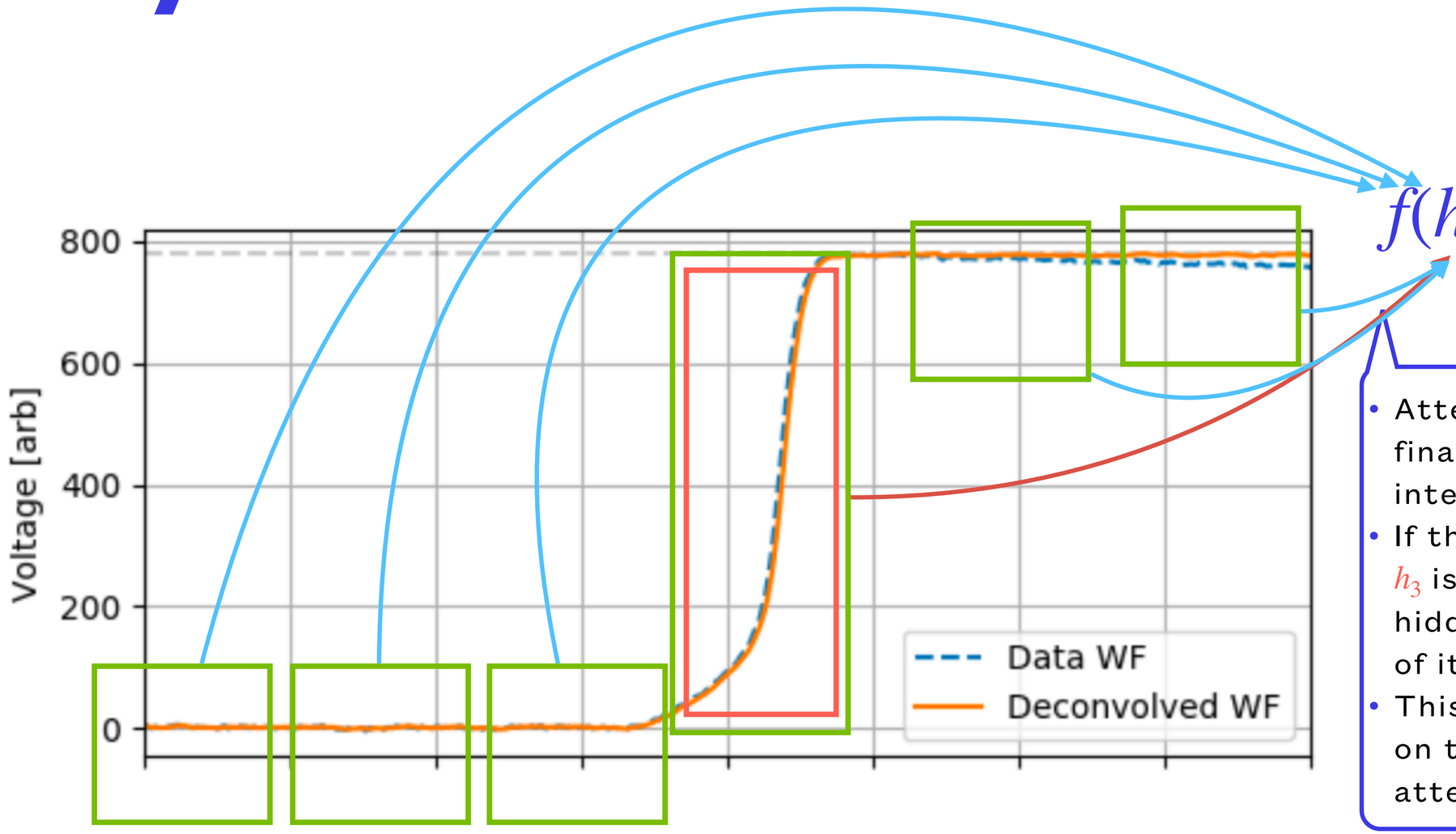
Why Attention Mechanism Works?



h_n

- Ideally, h_n should have already contained all information we need about the waveform
- But because of information over flood, it often loses focus to the most critical part of time series, but instead putting its attention on random noise

Why Attention Mechanism Works?

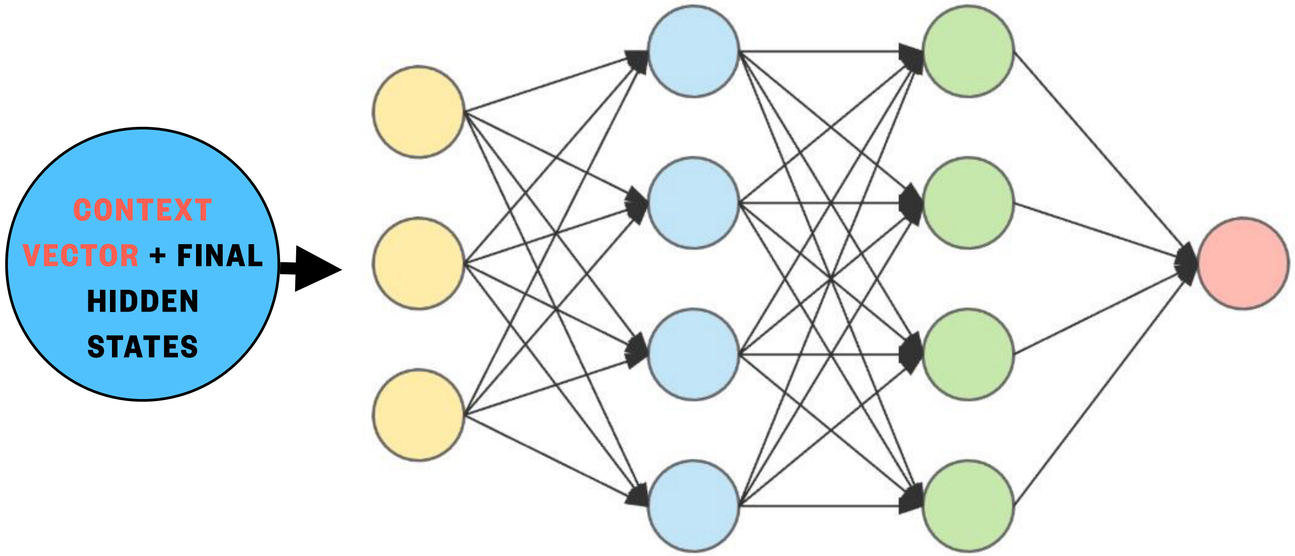


$f(h_i, h_n)$

- Attention Mechanism allows the final hidden state to “talk” to each intermediate hidden states
- If the feature in one hidden state h_3 is important, then the final hidden state should preserve most of its info
- This lead to high attention score on this state $s(h_3, h_n)$ and low attention score on other states

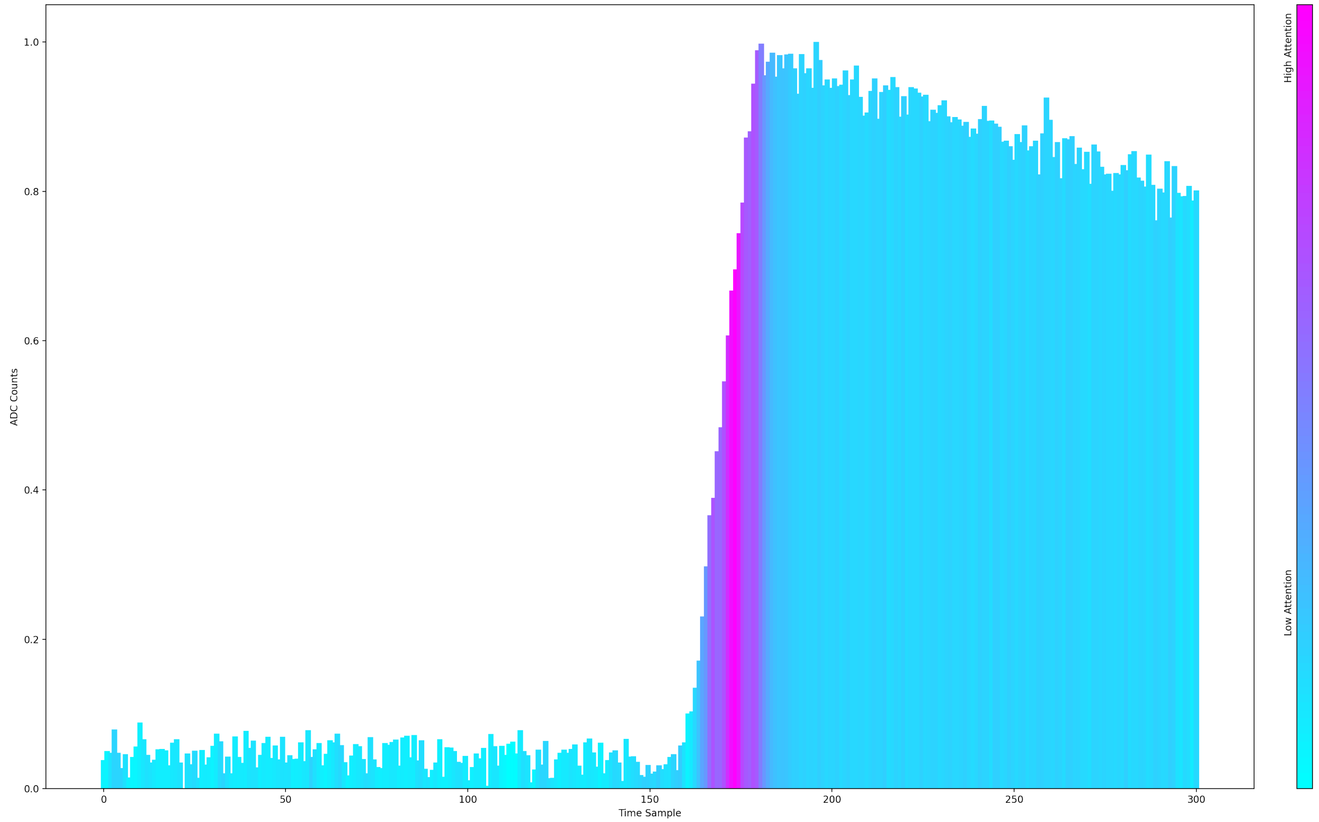
Proceed with Context Vector

- Concatenate context vector and final hidden state into a 1D vector
- Feed the 1D vector into fully connected neural network for:
  - Classification
  - Sequence-to-Sequence model for machine translation



Interpretable Machine Learning

- Attention score is an important tools to interpret the machine learning models
  - An attention score is assigned to each one or few time samples of the input time series
  - Higher attention score means this time segment is more important than others
- Plotting attention score of a trained waveform classifier on a fake Ge waveform, we see that it correctly put attention on the rising edge
  - The baseline/falling edge gets less attention, since they are indifferent from waveform to waveform



Summary

- What is time series?
  - Types of time series
  - Time series data in physics
  - Features of time series
- Time series models:
  - Vanilla RNN
  - LSTM
- Intro to Attention Mechanism
- Next time, we will discuss more about attention algorithm