+ Why sequence Models?

Audio clips would be sequence data

Music Generation Too.

DNA sequences Too.

Video activity recognition

+ Notation

$X^{\langle T \rangle}$ where T is the position in the sequence

T_x = length of sequence.

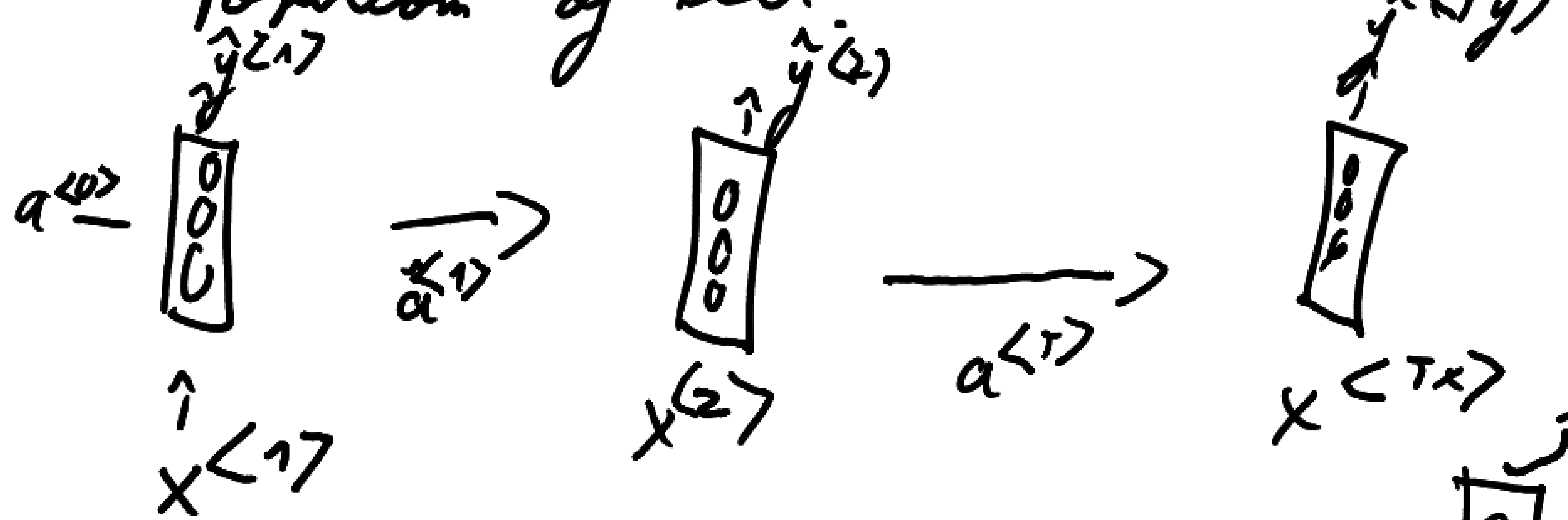
$y^{\langle T \rangle}$, T_y

$T_x^{(i)}$ = length of ith Training example.

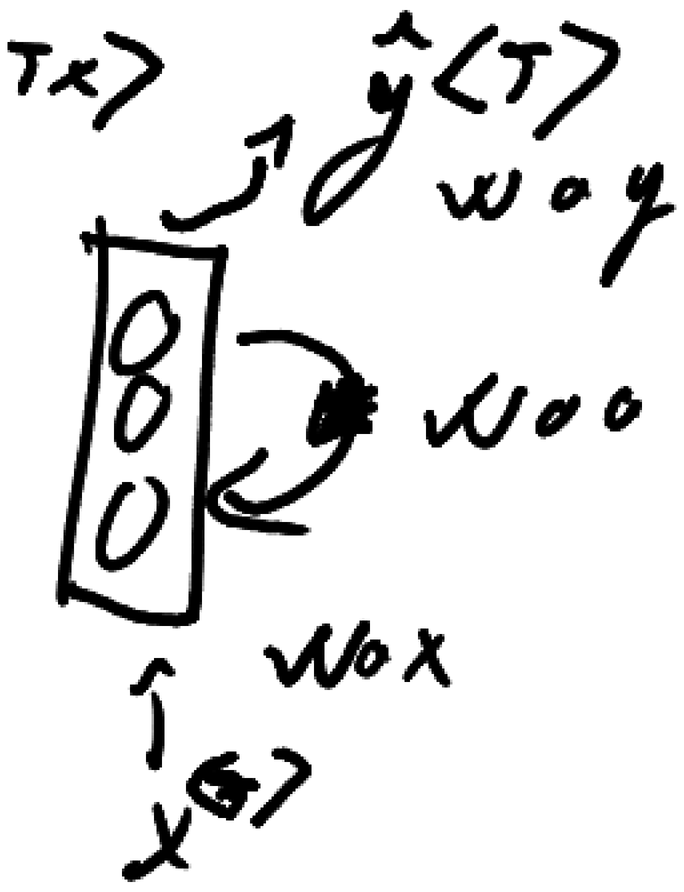
+ Recurrent Neural Network Model

Problem with Traditional NN architectures

- Inputs, outputs can be different lengths in different examples.
- Doesn't share features across different position of text



Other way to draw it:



$$a^{<t>} = g(W_{aa}\, a^{<t-1>} + W_{ax}\, x^{<t>} + b_a$$

$$\hat{y}^{<t>} = g(W_{ya}\, a^{<t>} + b_y)$$

$$100 \updownarrow [\underbrace{W_{aa}}_{100}, \underbrace{W_{ax}}_{10000}] = W_a \Rightarrow$$

$$(100, 10\,100)$$

$$\Rightarrow [a^{<t-1>}, x^{<t>}] = \begin{bmatrix} a^{<t-1>} \\ \hline x^{<t>} \end{bmatrix} \begin{matrix} \updownarrow 100 \\ \updownarrow 10.000 \end{matrix} \quad \updownarrow 10.100$$

+ Back propagation Through Time ::

$$\mathcal{L}(\hat{y}, y) = \sum_{T=1} \mathcal{L}^{<T>}\left(\hat{y}^{<T>}, y^{<T>}\right)$$

$$a^{(0)} \underset{\Longleftarrow}{\longrightarrow} a^{(T)}$$

+ Different Types of RNN'S :

- Many To Many ($T_x = T_y$)
- Sentiment classification => Many To one.
- One TO One (less used)

Music generation => One To many

Machine Translation (encoder - Decoder)

Many-To many

+ Language Model and Sequence Generation:

P(sentence) is what we are looking for.

- Training set: Large corpus of English text.

Tokenize => $y^{<1>}$ $y^{<2>}$ .... <EOS> End of sentence.

<UNK> = Unknown word.

+ Sampling Novel Sequences

Not quite good enough.

+ Main issue is a Vanishing gradient.

After forward prop, back prop has a very hard time with it

Local influences are common in RNN's i.e words next to each other hold more meaning.

- Exploding gradients can also be an issue
  - Solution is gradient clipping.

+ Gated Recurrent Unit (GRU)

Much better at capturing long range conection

c = memory cell. => $c^{<t>} = a^{<t>}$

$\tilde{c}^{<t>} = \tanh(W_c[c^{<t-1>}, x^{<t>}] + b_c)$

Γ_u = update gage. = $\sigma(W_u[c^{<t-1>}, x^{<t>}] + b_u)$

Γ → Gate

$c^{<t>} = \underbrace{\Gamma_u * \tilde{c}^{<t>} + (1-\Gamma_u)}_{\text{element wise}} * c^{<t-1>}$

Full GRU.

$\tilde{h}$: $\tilde{c}^{<t>} = \tanh(W_c[\Gamma_r * c^{<t-1>}, x^{<t>}] + b_c)$

u) $\Gamma_u = \sigma(W_u[c^{<t-1>}, x^{<t>}] + b_u)$

r) $\Gamma_r = \sigma(W_r[c^{<t-1>}, x^{<t>}] + b_c)$

h $c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1-\Gamma_u) * c^{<t-1>}$

+Long Short Term Memory. (LSTM)

Same but Two different update Terms

Γ_f / Γ_u

$\tilde{c}^{<t>} = \tanh(W_c [a^{<t-1>}, x^{<t>}] + b_c)$

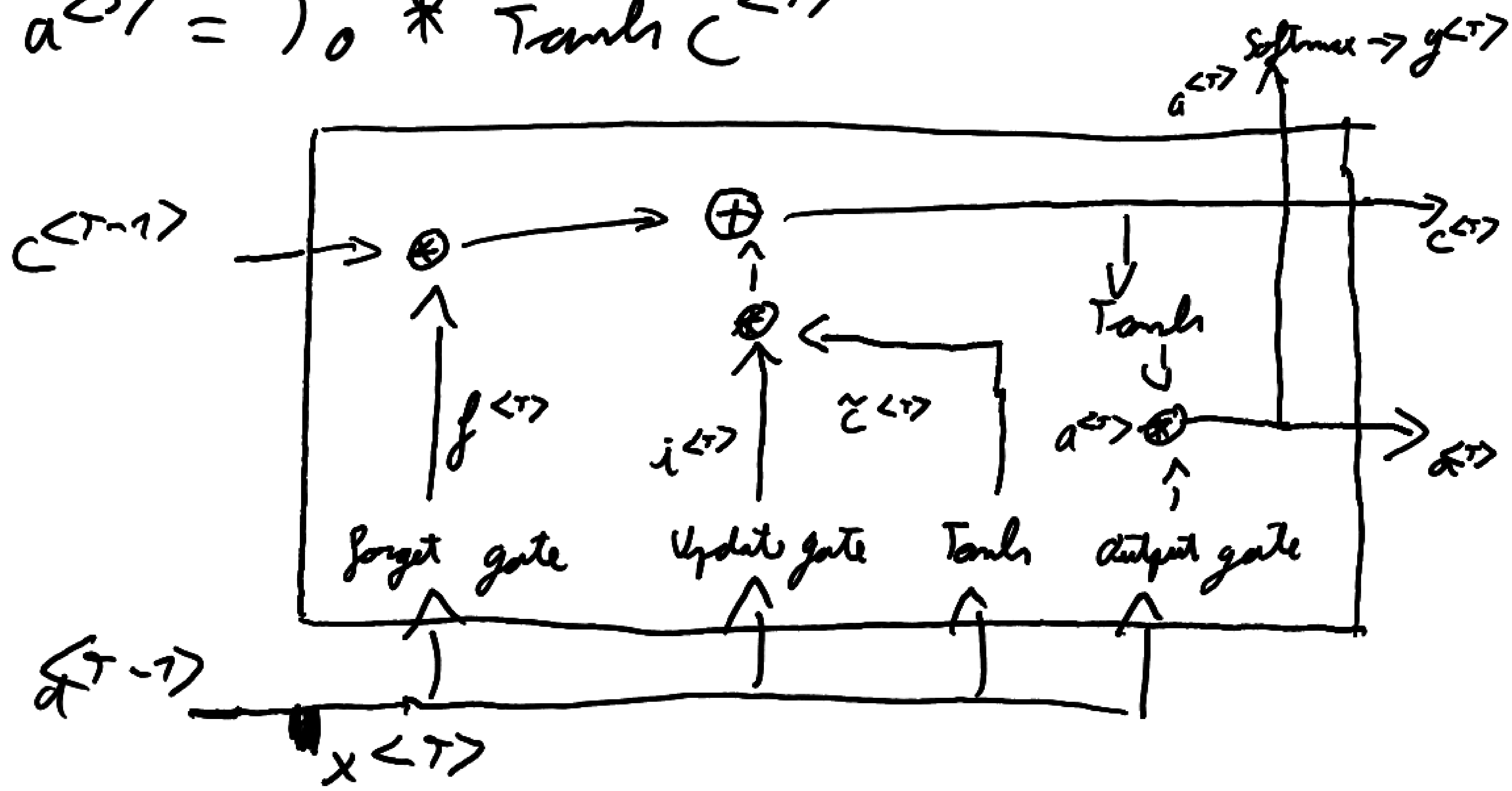
$\Gamma_u = \sigma(W_u [a^{<t-1>}, x^{<t>}] + b_u) \Rightarrow$ update

$\Gamma_f = \sigma(W_f [a^{<t-1>}, x^{<t>}] + b_f) \Rightarrow$ forget

$\Gamma_o = \sigma(W_o [a^{<t-1>}, x^{<t>}] + b_o) \Rightarrow$ output

$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + \Gamma_f$

$a^{<t>} = \Gamma_o * \tanh c^{<t>}$



+ Bidirectional RNN.

A modification to all previous models, that take info from the entire sequence of data, and you need to wait for the entire RNN

+ Deep RNN example