

Natural Language Processing

AIGC 5501

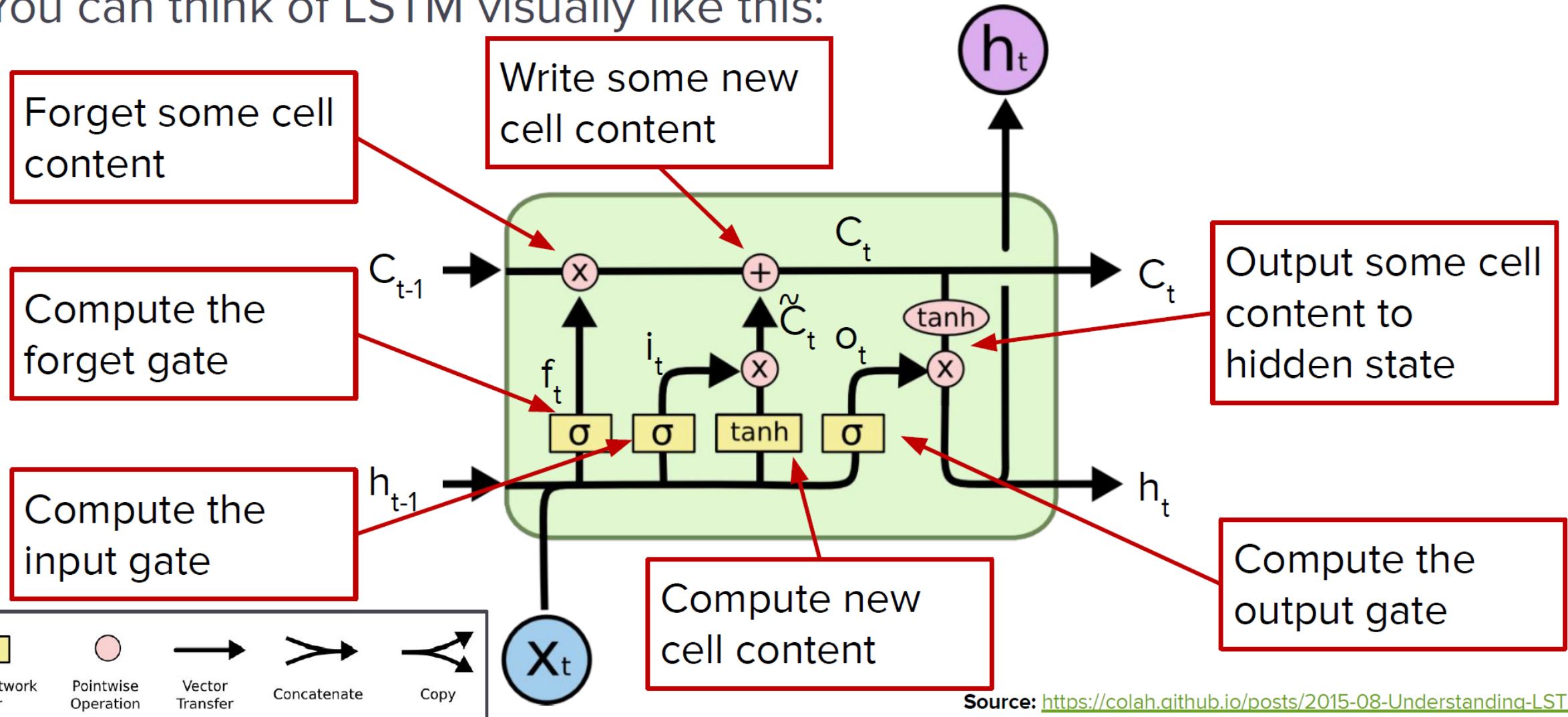
Encoder-Decoders
Attention

Instructor: Ritwick Dutta
Email: ritwick.dutta@humber.ca

Last Class

Long Short-Term Memory RNNs (LSTMs)

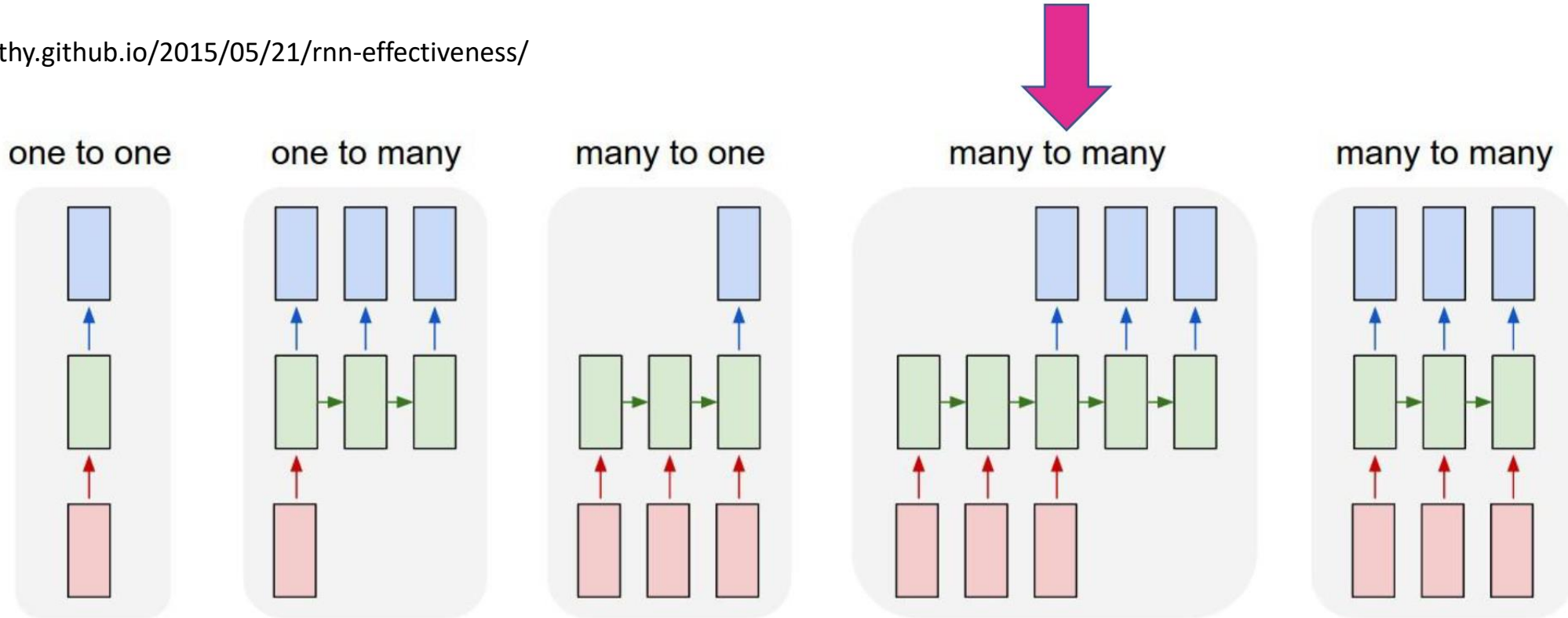
You can think of LSTM visually like this:



Encoder-decoder models

- RNNs for sequence-to-sequence tasks

<https://karpathy.github.io/2015/05/21/rnn-effectiveness/>



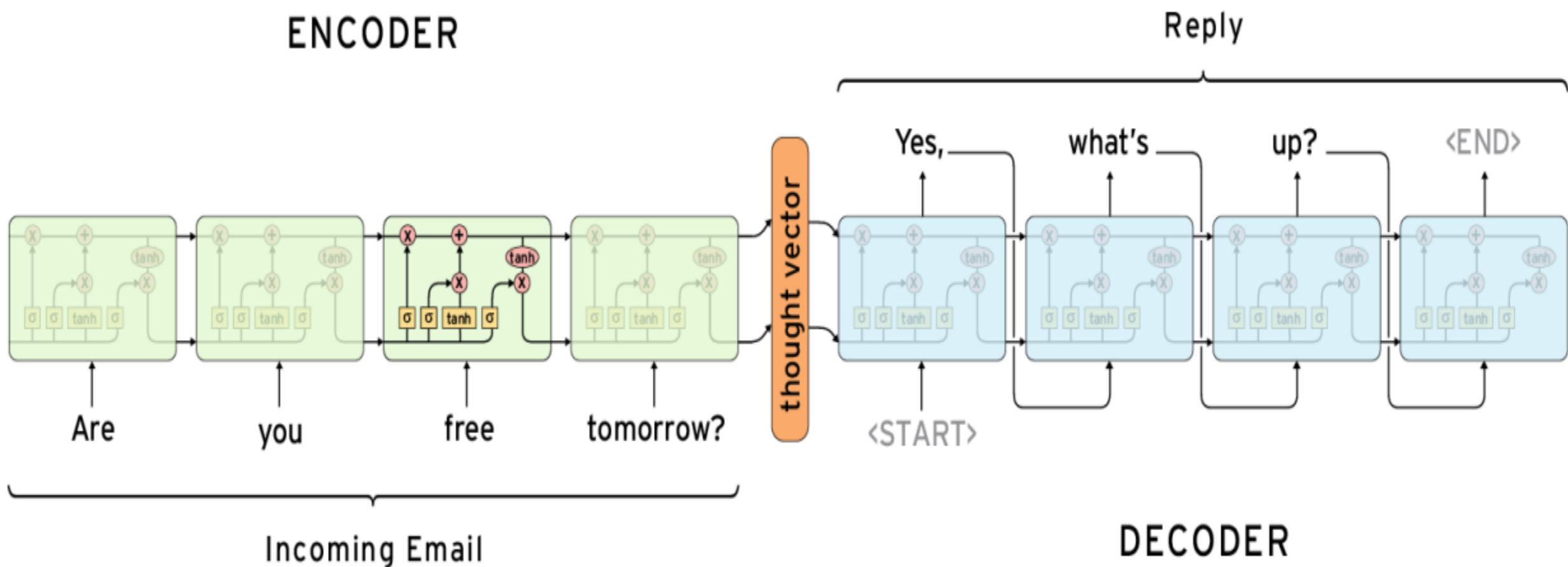
Each rectangle is a vector and arrows represent functions (e.g. matrix multiply). Input vectors are in red, output vectors are in blue and green vectors hold the RNN's state (more on this soon).

From left to right: **(1) Vanilla mode** of processing without RNN, from fixed-sized input to fixed-sized output (e.g. image classification). **(2) Sequence output** (e.g. image captioning takes an image and outputs a sentence of words). **(3) Sequence input** (e.g. sentiment analysis where a given sentence is classified as expressing positive or negative sentiment). **(4) Sequence input and sequence output** (e.g. Machine Translation: an RNN reads a sentence in English and then outputs a sentence in French). **(5) Synced sequence input and output** (e.g. video classification where we wish to label each frame of the video).

Key idea

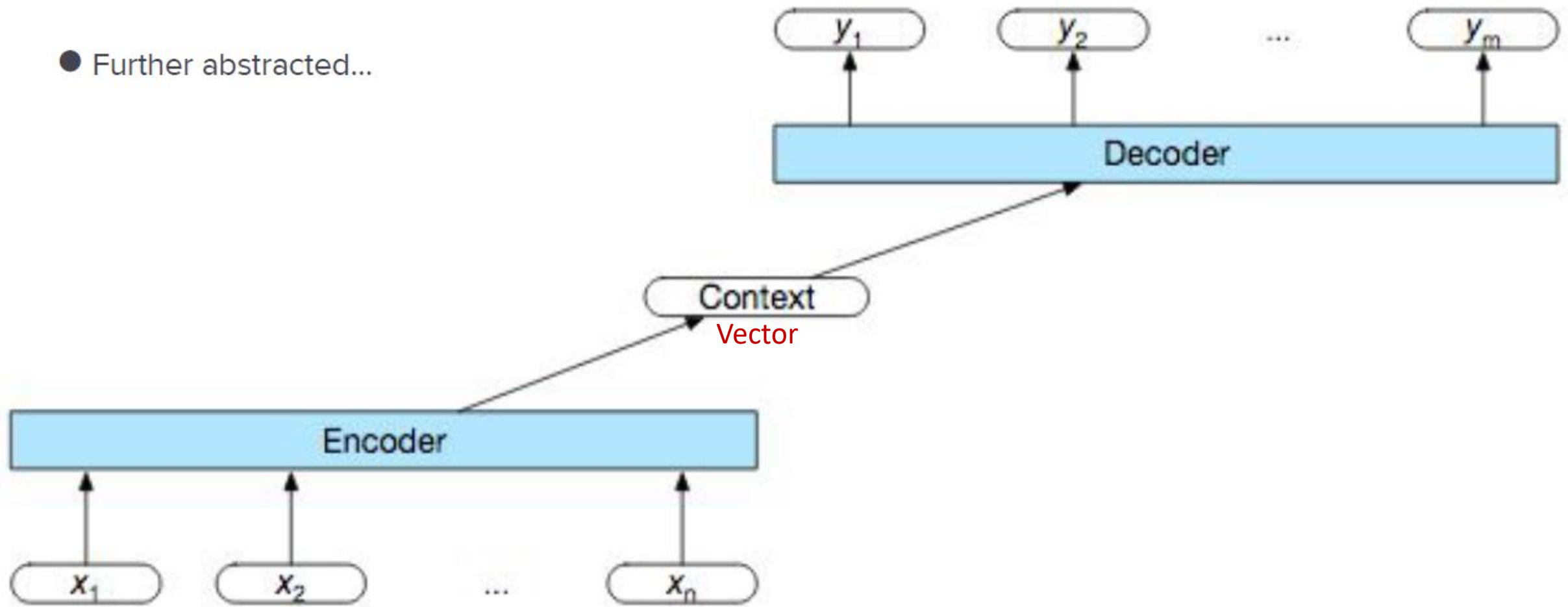
- An **encoder** network takes an input sequence and creates a contextualized representation of it
- This representation is then passed to a **decoder** which generates a task-specific output sequence
- Capable of generating contextually appropriate, arbitrary length, output sequences.
- Pipeline architecture that can be **trained end-to-end** rather than in two separate stages

General RNN-based encoder-decoder architecture



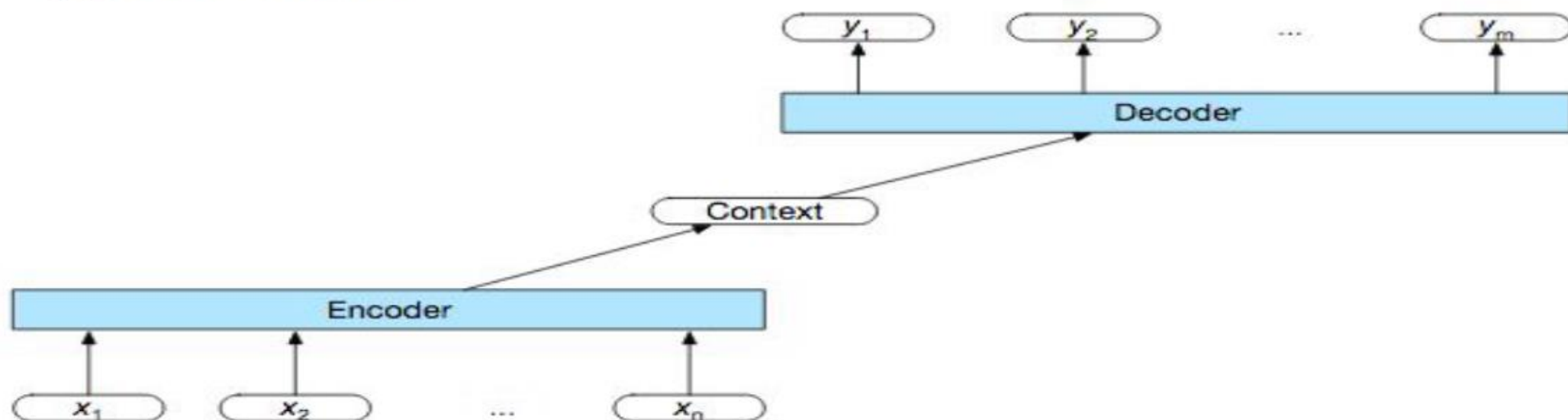
General RNN-based encoder-decoder architecture

- Further abstracted...



General RNN-based encoder-decoder architecture


1. An **encoder** that accepts an input sequence, x_1^n , and generates a corresponding sequence of contextualized representations, h_1^n .
2. A **context vector**, c , which is a function of h_1^n , and conveys the essence of the input to the decoder.
3. And a **decoder**, which accepts c as input and generates an arbitrary length sequence of hidden states h_1^m , from which a corresponding sequence of output states y_1^m , can be obtained.



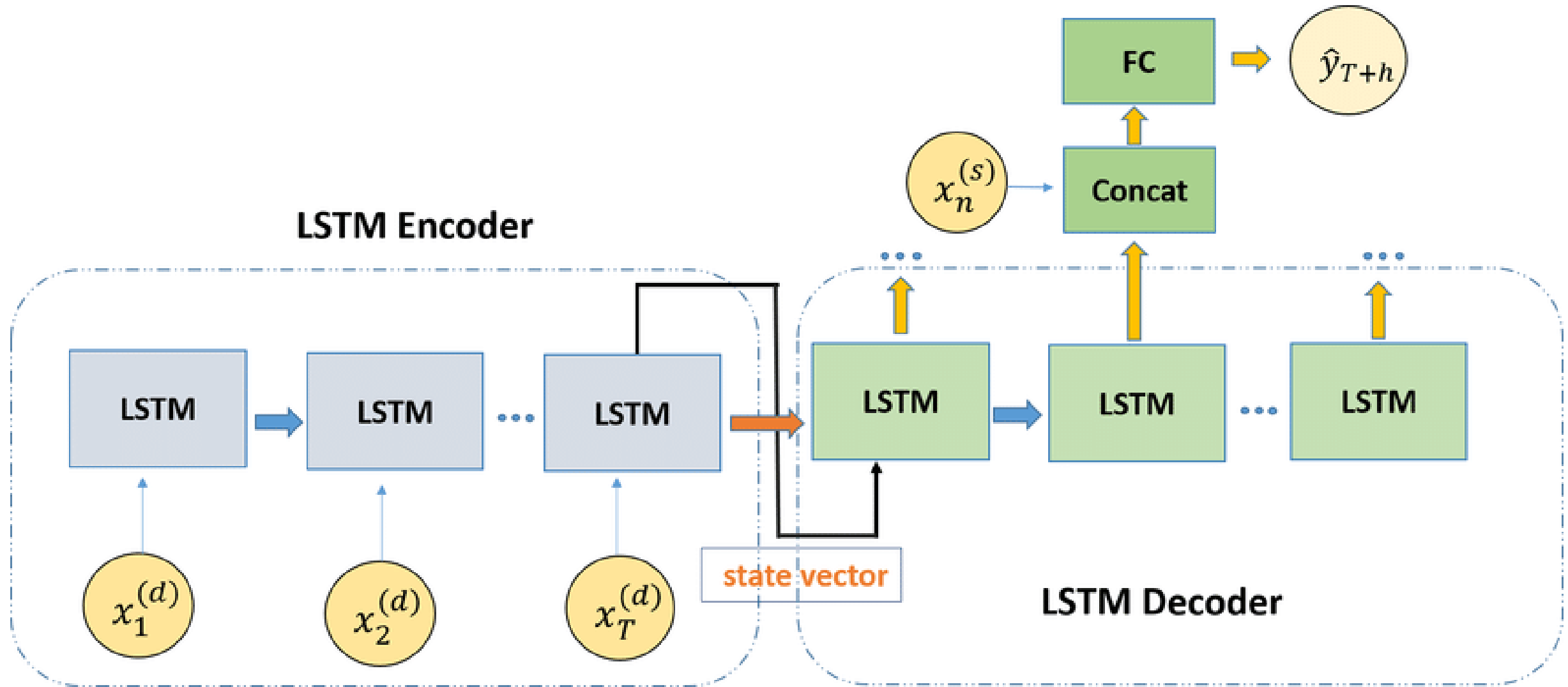
Role of Encoder

- Can be simple RNNs, LSTMs, GRUs, convolutional networks (not covered), transformer networks (not covered yet)
- Typically are multi-layer (stacked)
- Final representation derived from the **top layer** of the stack
- Common architecture
 - Stacked Bi-LSTMs
 - Final contextualized representation is concatenated hidden states from final time step of top layer of forward and backward pass
- For training the encoder
 - Input to output layer is concatenated hidden states from top layer of forward and backward pass **at the current time step**

Passed to the decoder



Encoder Decoder Architecture



Training an Encoder-Decoder

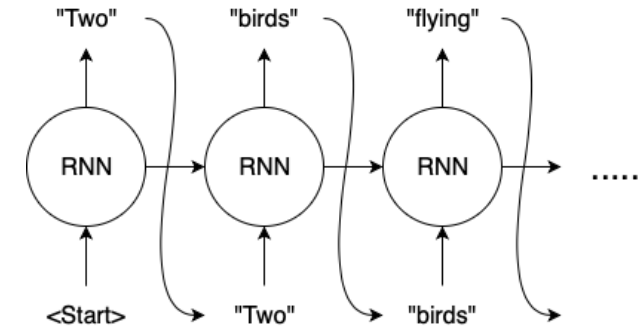
- Can train encoder and decoder separately
 - Encoder usually a language model
 - Decoder can be a language model
 - Or decoder can generate word-level predictions
 - Needs joint **fine-tuning**
- Can train encoder-decoder as a single pipeline, end-to-end
- Uses teacher-forcing

Pros:

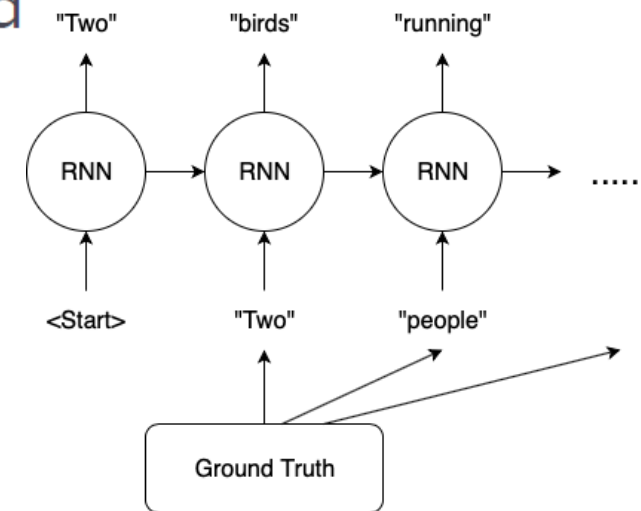
Training with Teacher Forcing **converges faster**.

Cons:

There is a discrepancy between training and inference, and this might lead to poor model performance and instability. This is known as **Exposure Bias** in literature.



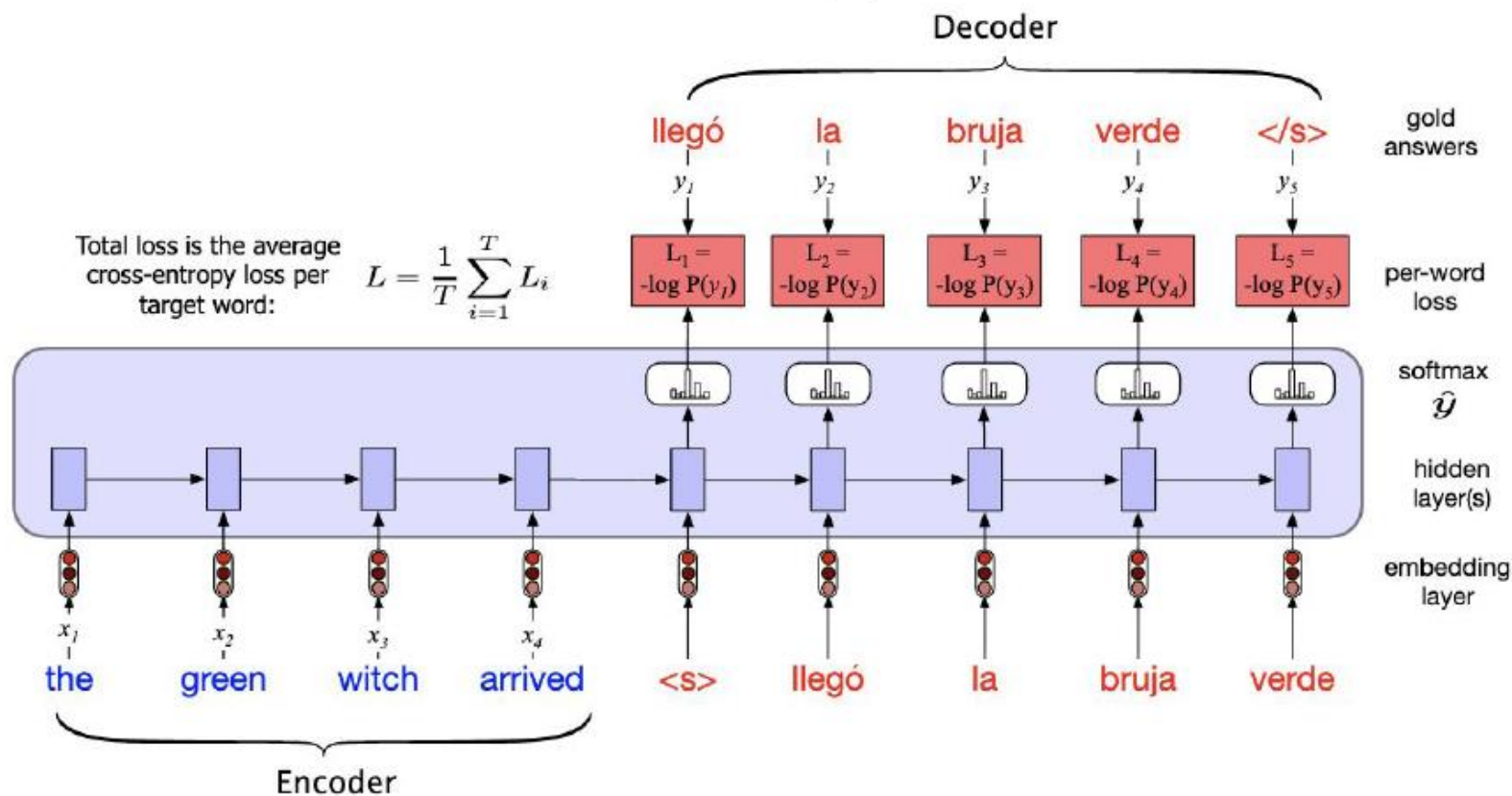
Without Teacher Forcing



With Teacher Forcing

Training an encoder-decoder

- Use **teacher forcing** in the decoder, i.e., force the system to use the gold target token from training as the next input x_{t+1}
- Speeds up training



Encoder-decoder applications

Email Response Generator

- Using RNNs to produce one sequence as a response to another
 - E.g., predict responses for incoming emails
 - **Encoder:** represents the “essence” of the email as a feature vector
 - **Decoder:** autoregressively generates the response

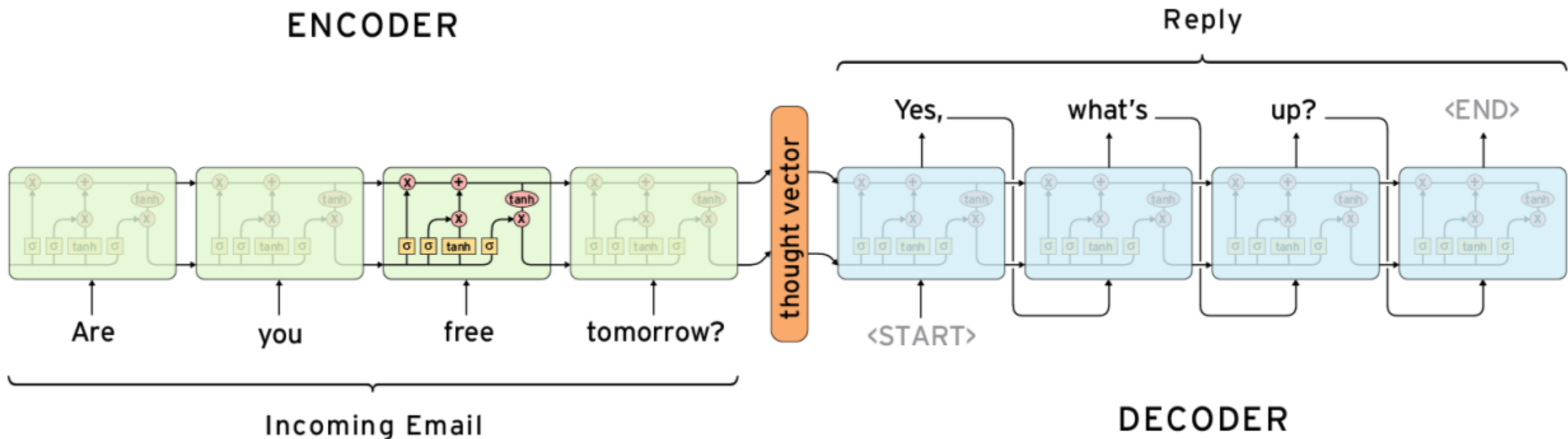


Image Captioning

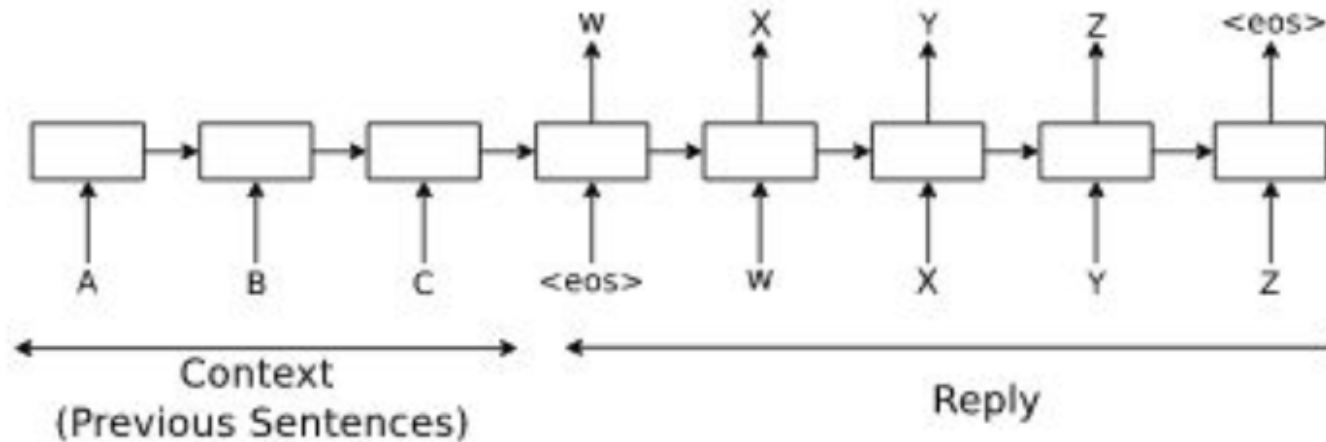
- Generation of text captions for images



A woman is throwing a frisbee in a park.

Chatbots

- Systems that parry the user's input with a response that keeps the conversation going
- **Encoder:** converts the user's input into a vector
- **Decoder:** produces a sequence of words as a response



Trained on IT Helpdesk dataset

Machine: *hi*

Human: *hello*

Machine: *could you please let me know what are the operating systems you are using ?*

Human: *i am using linux*

Machine: *ok it was in the ssl vpn on which you stated it was broken*

Machine: *please check if you are logged in to corp network*

Machine: *what is the error that you are running please*

Human: *i am seeing an error related to vpn*

Machine: *what is the error message that you are getting when connecting to vpn using network connect ?*

Human: *connection refused or something like that*

Machine: *may i know the version of network connect you connect ?*

Human: *i am not sure i know that*

Machine: *is the network connect prompting for any user certificate*

Machine: *?*

Human: *yes*

Machine: *are you at home or at work at the moment ?*

Machine: *office ?*

Human: *i am at home*

Machine: *try this*

Machine: *goto <URL> page and enter your name and password*

Human: *done*

Machine: *try logging in with <NAME> and check if its working fine now*

Human: *yes , now it works !*

Machine: *great*

Machine: *anything else that i can help*

Human: *no , that was very useful*

Machine: *you are other than <NAME> :)*

Machine: *have a nice night*

Trained on OpenSubtitles

Conversation 2: Simple Q&A (contexts and multiple choice)

Human: *my name is david . what is my name ?*

Machine: *david .*

Human: *my name is john . what is my name ?*

Machine: *john .*

Human: *are you a leader or a follower ?*

Machine: *i 'm a leader .*

Human: *are you a follower or a leader ?*

Machine: *i 'm a leader .*

Conversation 4: Philosophical Q&A

Human: *what is the purpose of life ?*

Machine: *to serve the greater good .*

Human: *what is the purpose of living ?*

Machine: *to live forever .*

Human: *what is the purpose of existence ?*

Machine: *to find out what happens when we get to the planet earth .*

Human: *where are you now ?*

Machine: *i 'm in the middle of nowhere .*

Human: *what is the purpose of dying ?*

Machine: *to have a life .*

Human: *what is the purpose of being intelligent ?*

Machine: *to find out what it is .*

Human: *what is the purpose of emotions ?*

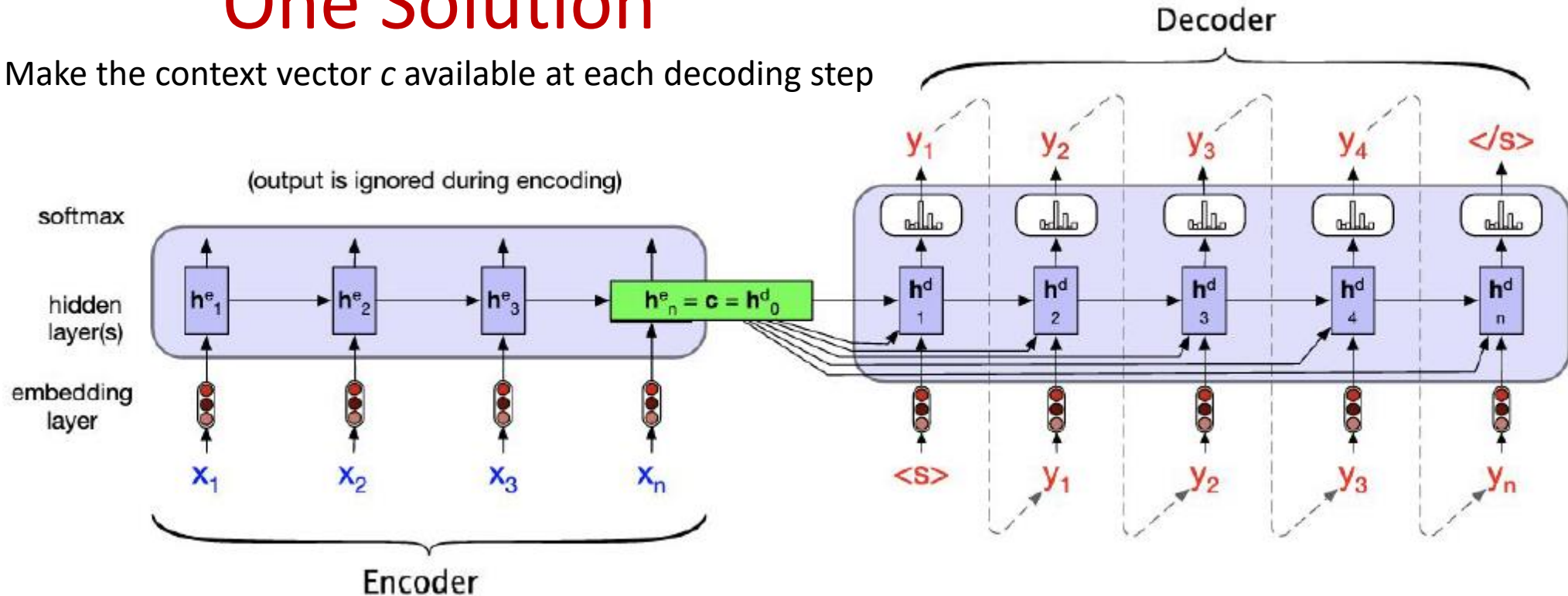
Machine: *i don 't know .*

Primary weakness of the encoder-decoder model

- The context vector, c , is only directly available at the beginning of the decoding process
- Its influence wanes as the output sequence is generated

One Solution

Make the context vector c available at each decoding step



Another weakness

- **Difficult for the decoder to keep track of what has already been generated** and what has not

One solution

- Condition the **output** on the newly generated hidden state, the output generated at the previous state, and the encoder context.

$$h_t^d = g(\hat{y}_{t-1}, h_{t-1}^d, c)$$

$$z_t = f(h_t^d)$$

$$y_t = \text{softmax}(z_t)$$

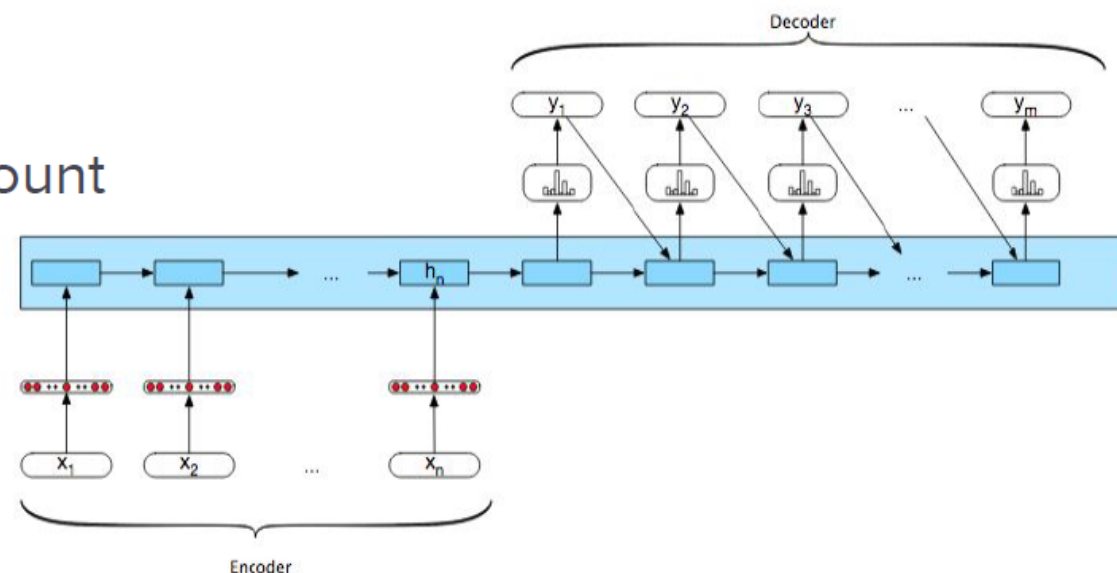
FFNN



$$y_t = \text{softmax}(\hat{y}_{t-1}, z_t, c)$$

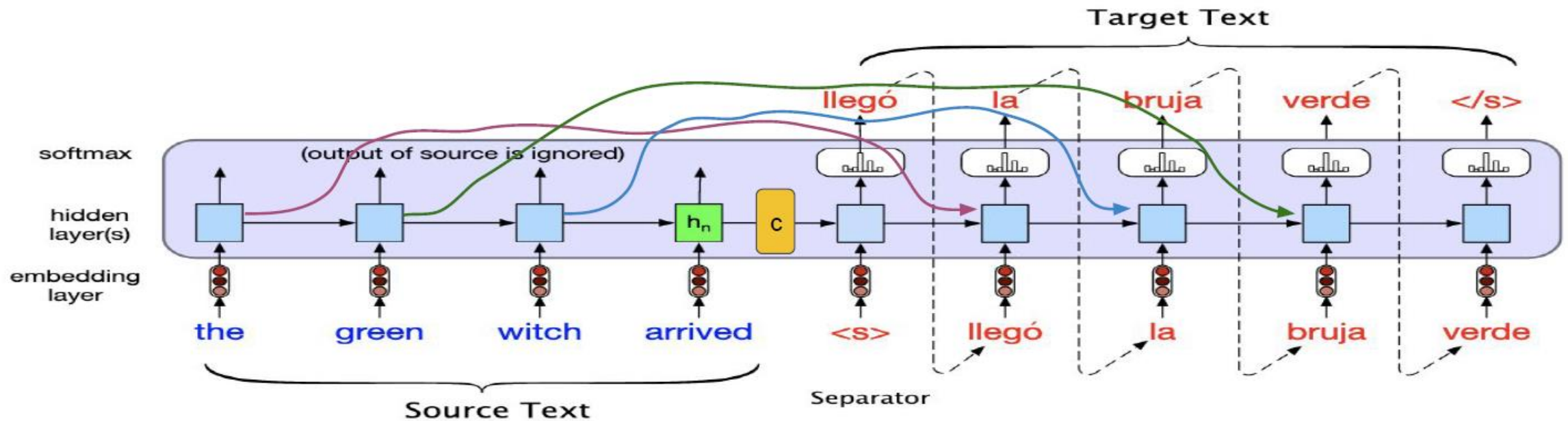
Final(!) additional problem with context vector c

- Context vector c is a function of the hidden states of the encoder
- Forced to **encode** a **variable number** of hidden states
- Loses useful information about each of the individual encoder states that may prove useful in decoding
- We need a method that can:
 - Take entire encoder context into account
 - Dynamically update during decoding
 - Be embodied in fixed size vector



Possible solution

If we knew a relevant encoder state $h_{t,enc}$ to use, we could use that instead of c_{enc}



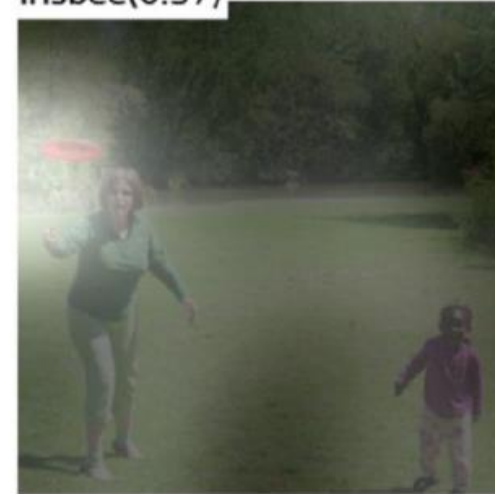
Attention to the rescue!

- Intuition
 - When translating a sentence, we pay special attention to the word or phrase that we are translating.

woman(0.54)



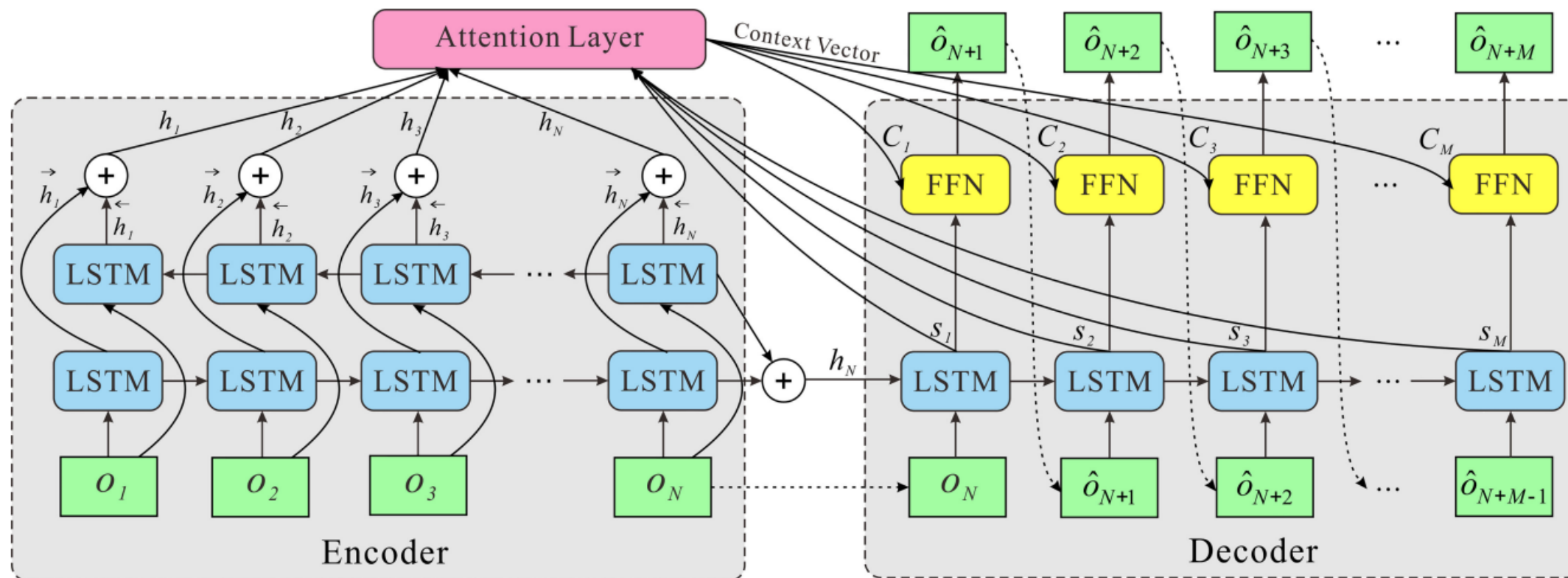
frisbee(0.37)



Attention mechanisms in RNNs do the same thing!

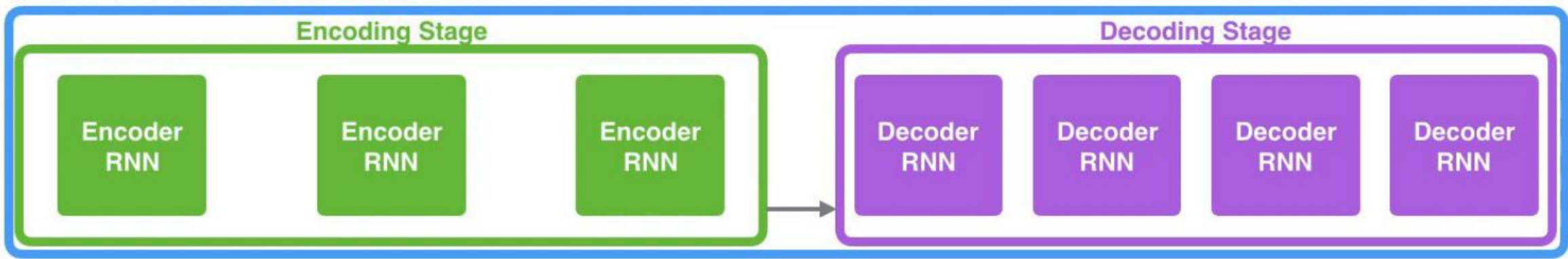
Attention: the Basic Idea

- Dynamically derive a context vector, c , from encoder hidden states at each step, i , during decoding; refer to each as c_i
- Take all of the encoder hidden states into account
- Condition the computation of the current decoder state on c_i (and prior hidden state and previous output)



Encoder-Decoder without Attention

Neural Machine Translation
SEQUENCE TO SEQUENCE MODEL



Now if we pay attention....

Neural Machine Translation
SEQUENCE TO SEQUENCE MODEL WITH ATTENTION



Attention: Computing c_i

Attention at time step 4

1. Prepare inputs



h_1



h_2



h_3

Encoder
hidden
states



Decoder hidden
state at time step 4

2. Score each hidden state

13	9	9
----	---	---

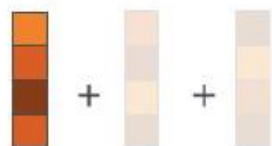
scores
Attention weights for
decoder time step #4

3. Softmax the scores

0.96	0.02	0.02
------	------	------

softmax scores

4. Multiply each vector by
its softmaxed score



=

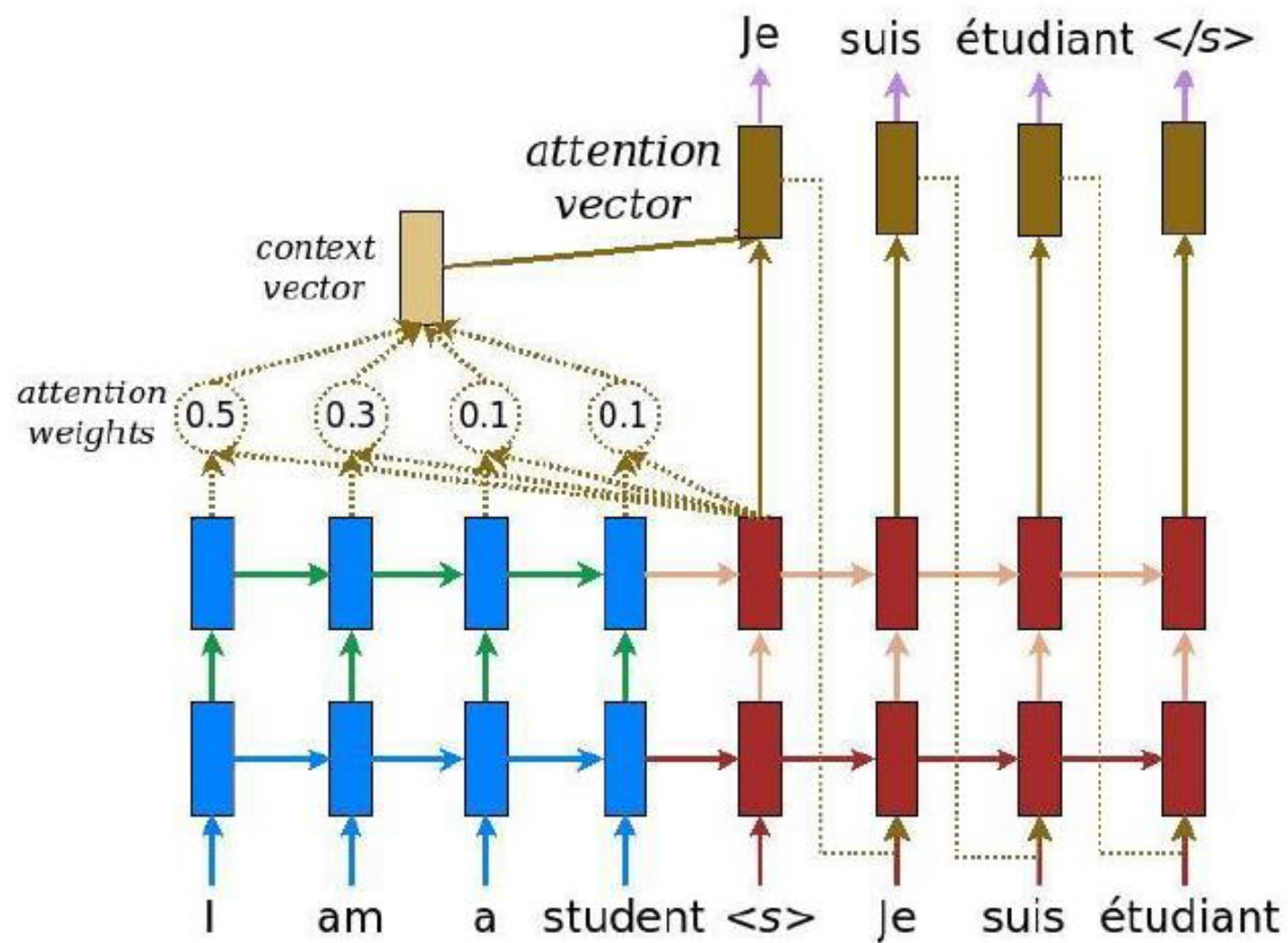
5. Sum up the weighted
vectors



Context vector for
decoder time step #4

Source: <https://jalammar.github.io/>

Attention – for MT

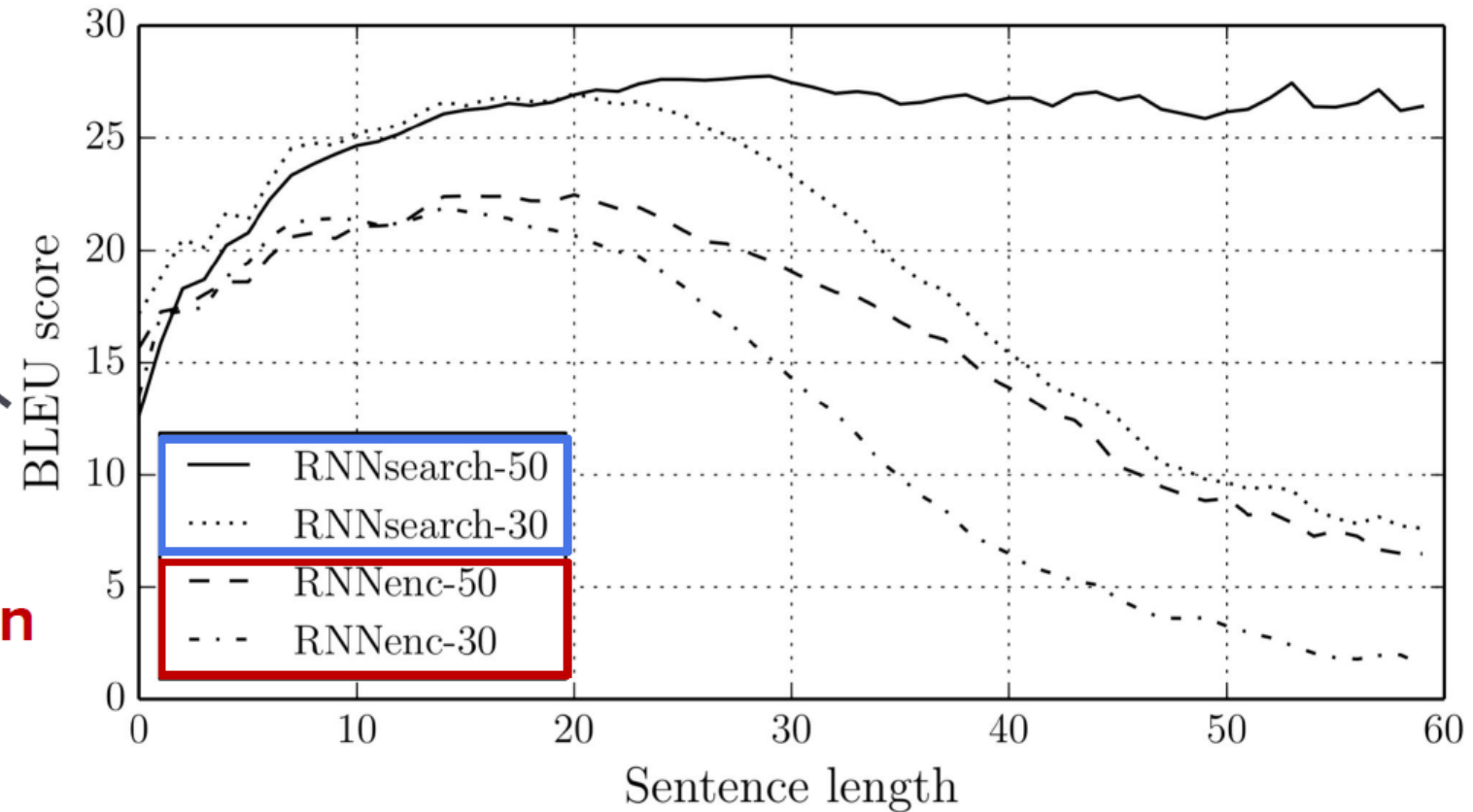


Machine translation

Higher score is better

With Attention

Without Attention



Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," ICLR, 2015

BLEU (BiLingual Evaluation Understudy) is a metric for automatically evaluating machine-translated text. The BLEU score is a number between zero and one that measures the similarity of the machine-translated text to a set of high quality reference translations.

Ref: <https://towardsdatascience.com/foundations-of-nlp-explained-bleu-score-and-wer-metrics-1a5ba06d812b>

Attention Model Watch

<https://www.youtube.com/watch?v=PSs6nxngL6k>

Lab - 10

Exercise 1: English sentences to French translation using LSTM sequence-to-sequence (Content): <https://blog.keras.io/a-ten-minute-introduction-to-sequence-to-sequence-learning-in-keras.html>

Code: https://keras.io/examples/nlp/lstm_seq2seq/

Exercise 2: Tutorial to demonstrate how to train a sequence-to-sequence model for Spanish-to-English translation roughly based on Effective Approaches to Attention-based Neural Machine Translation (Luong et al., 2015).

https://www.tensorflow.org/text/tutorials/nmt_with_attention

Final Projects

Thanks for submitting the group information

Choose one from the below list

- a) Chatbot for Humber - students association assistant (Eng + French support)
- b) Creative assistant for Humber marketing team (Eng + French support)
- c) Document summarizer - PDF / word / reports (Eng + French support)