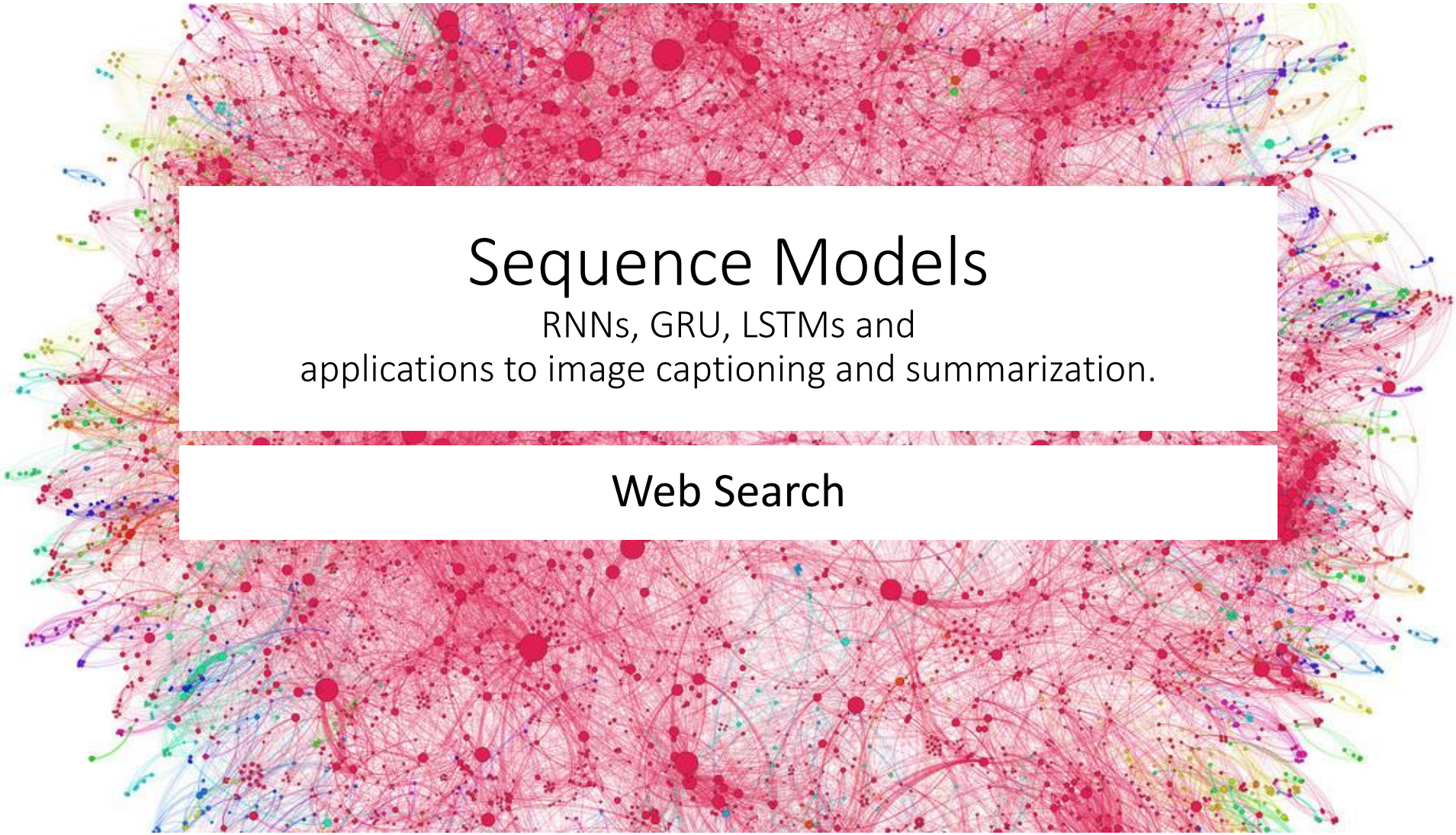


Sequence Models

RNNs, GRU, LSTMs and
applications to image captioning and summarization.

Web Search

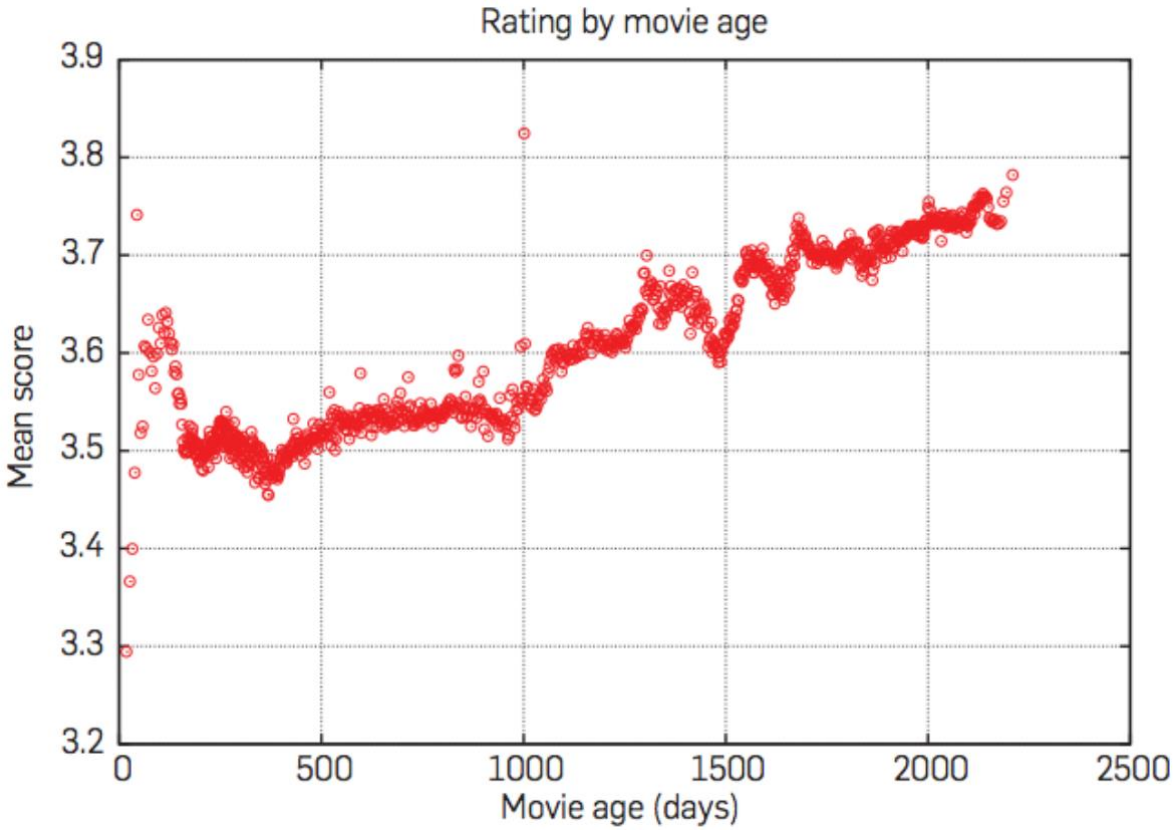
From static data to sequence data

- There are many domains where data samples have a dynamic nature or unknown size.

- CNNs and traditional multi-layer networks cope well with fixed-size input and output.

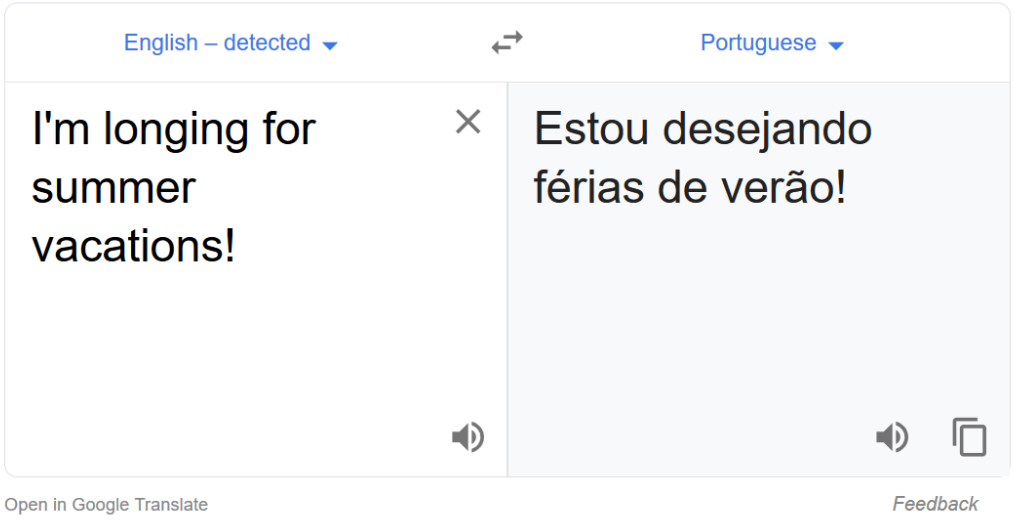
- However, there are many domains where:
  - Input data is a sequence
  - Output is a sequence

Sequence problems: trend analysis



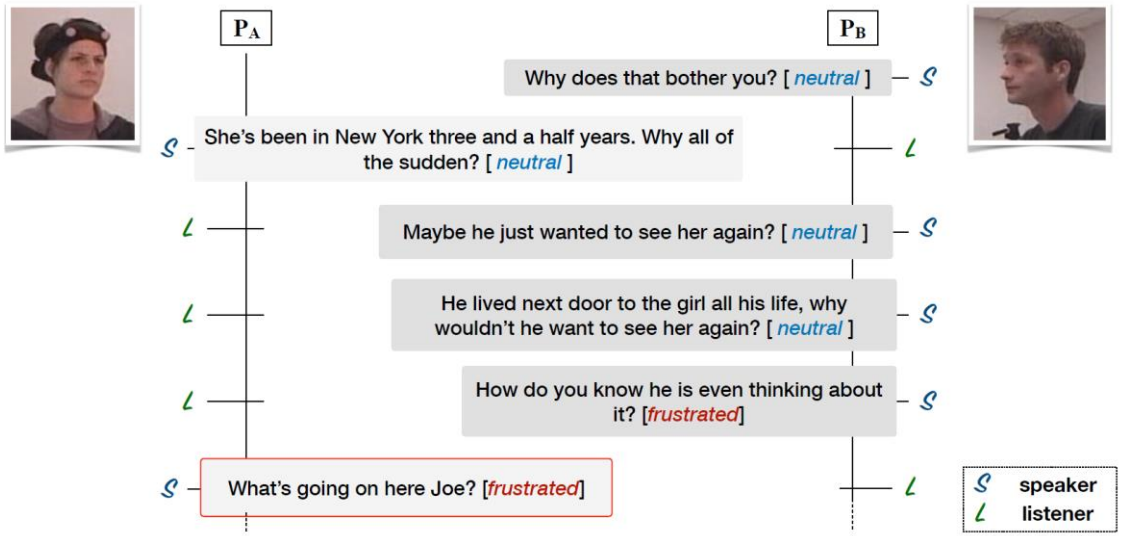
Sequence problems: Machine translation

- Translating a sentence from one language to another language.



Sequence problems: Sentiment analysis

- Detection of excitement, depression, frustration, etc.



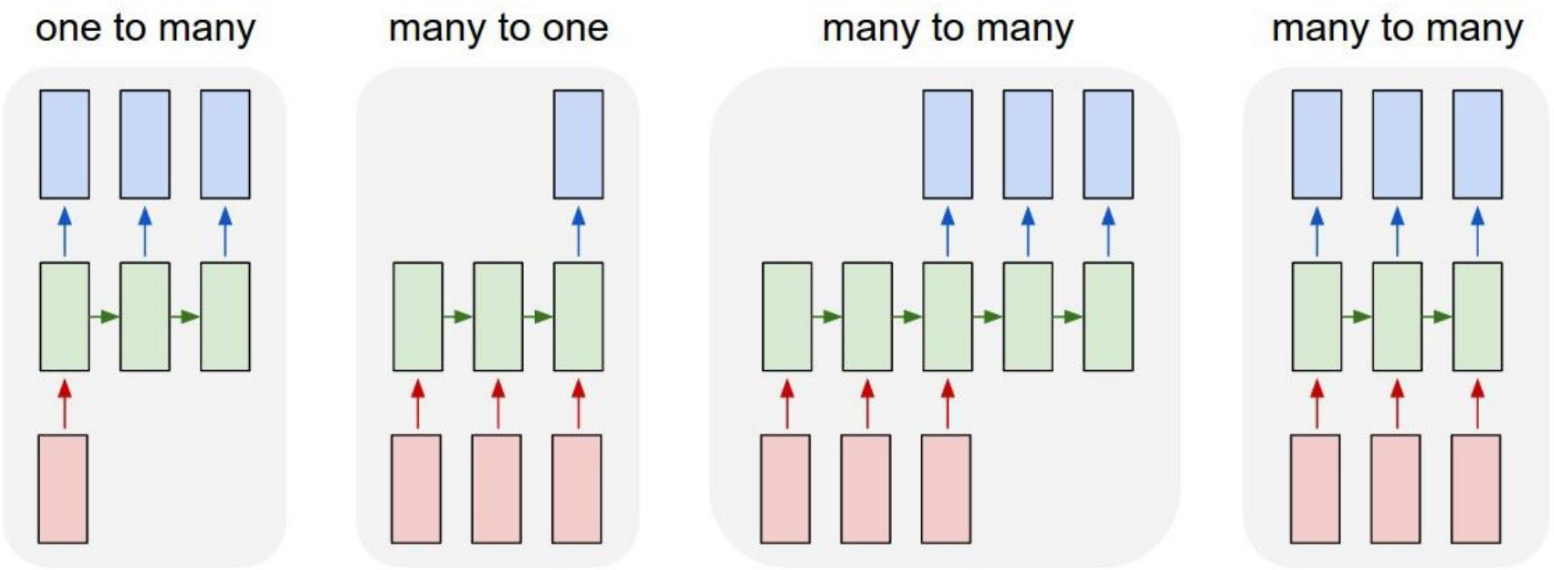
Majumder, Navonil, Soujanya Poria, Devamanyu Hazarika, Rada Mihalcea, Alexander Gelbukh, and Erik Cambria. "Dialoguernn: An attentive rnn for emotion detection in conversations." *AAAI* 2019.

Tang, Duyu, Bing Qin, and Ting Liu. "Aspect level sentiment classification with deep memory network." *arXiv* 2016.

Usually, data is not

Independent and Identically Distributed (IID).

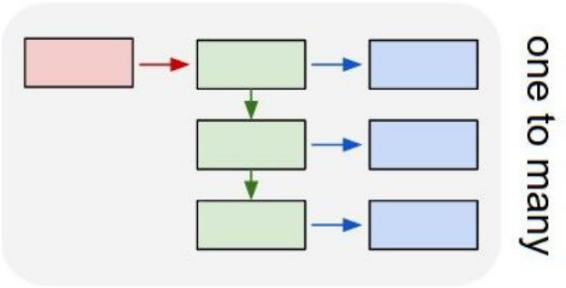
There are several possible architectures



Web data sequence modeling tasks

Input **Output**

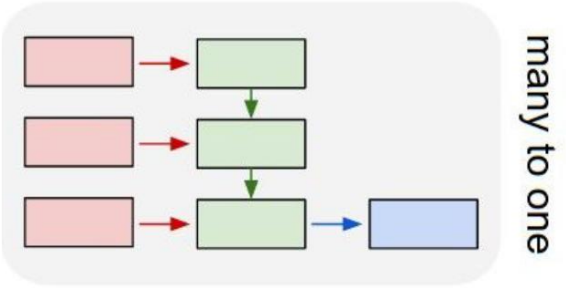
Image Captioning



A person riding a motorbike on dirt road

Sentiment Analysis

RNNs are awesome.



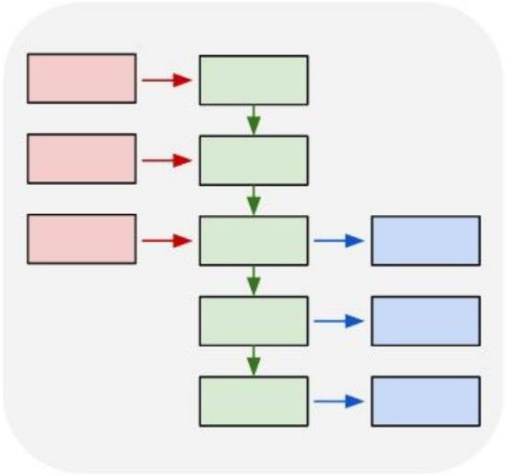
Positive

Web data sequence modeling tasks

Input **Output**

Machine Translation

Happy birthday!



ਜਨਮਦਿਨ ਮੁਬਾਰਕ

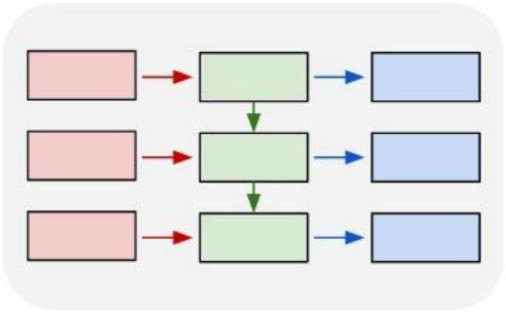
Janamadina mubāraka

Information Extraction

Word 1

Word 2

Word 3



ADJ

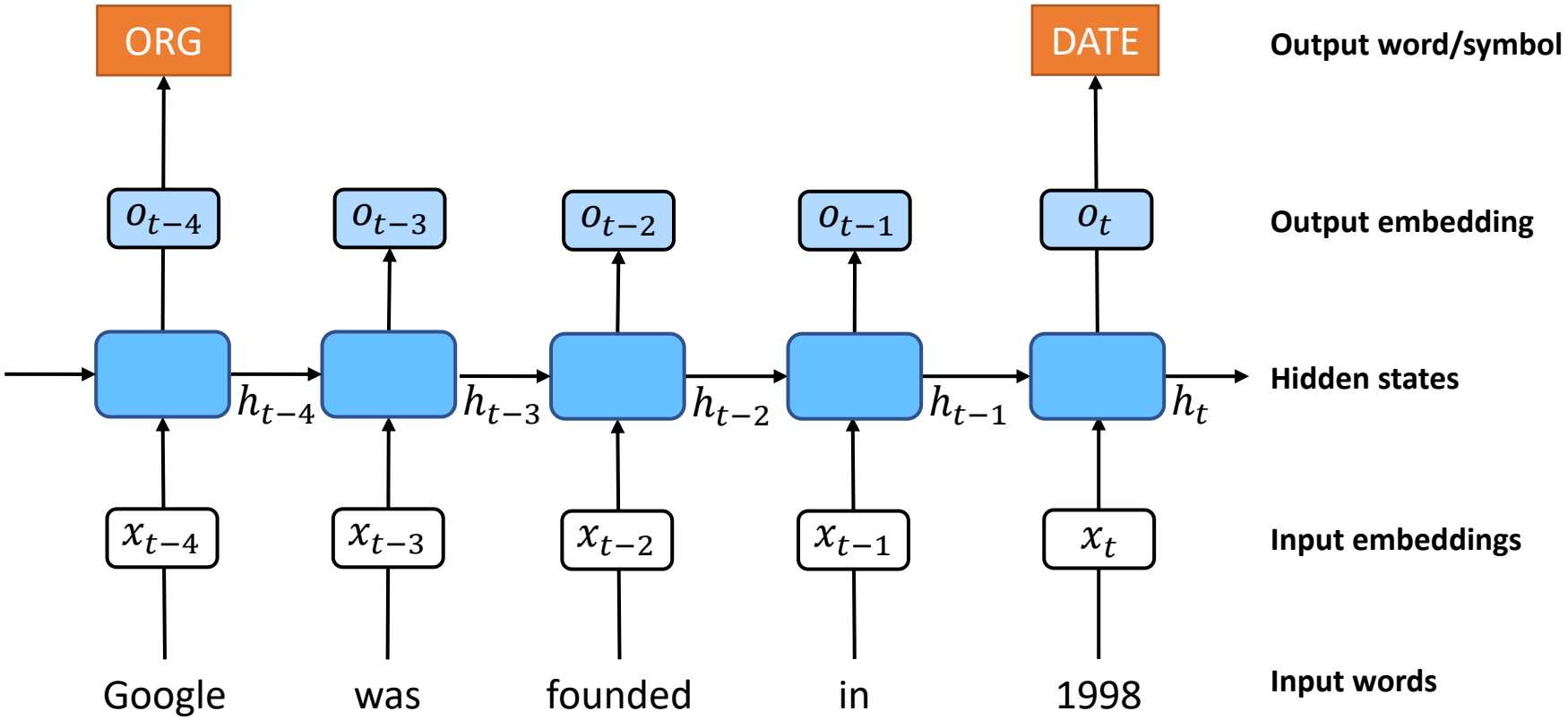
NOUN

Sure! Here you are ….

Information extraction with Spacy

When Sebastian Thrun **PERSON** started working on self-driving cars at Google **ORG** in 2007 **DATE**, few people outside of the company took him seriously.

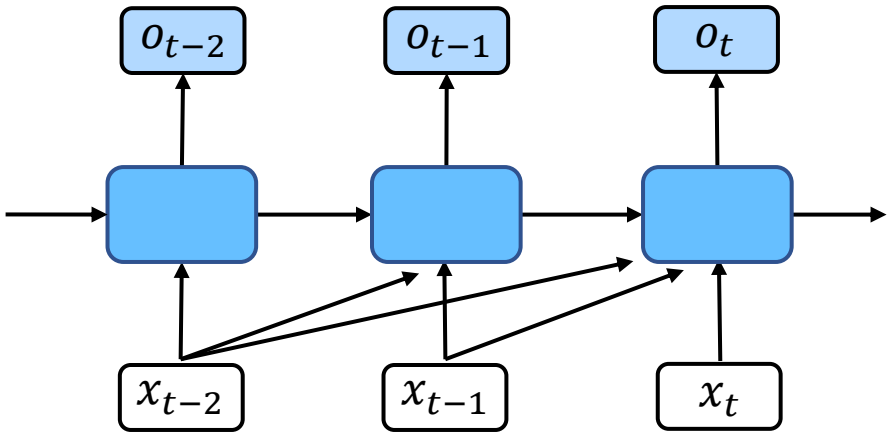
Information extraction



Auto regressive models

- In auto-regressive models the current output depends on the current input and a limited span of past inputs.

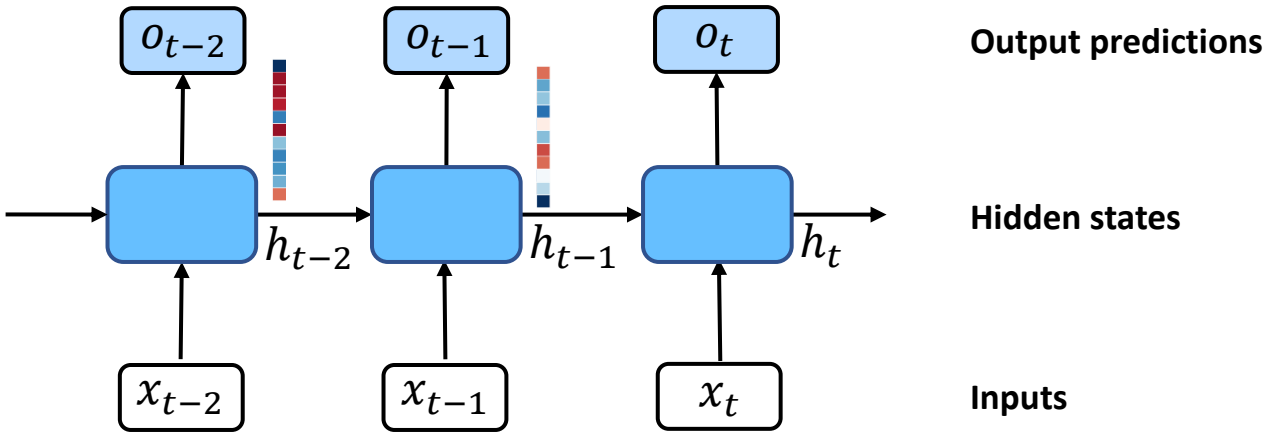
$$o_t = p(x_t \,|x_{t-1}, x_{t-2}, \ldots, x_{t-\tau})$$



Latent auto regressive models (RNNs)

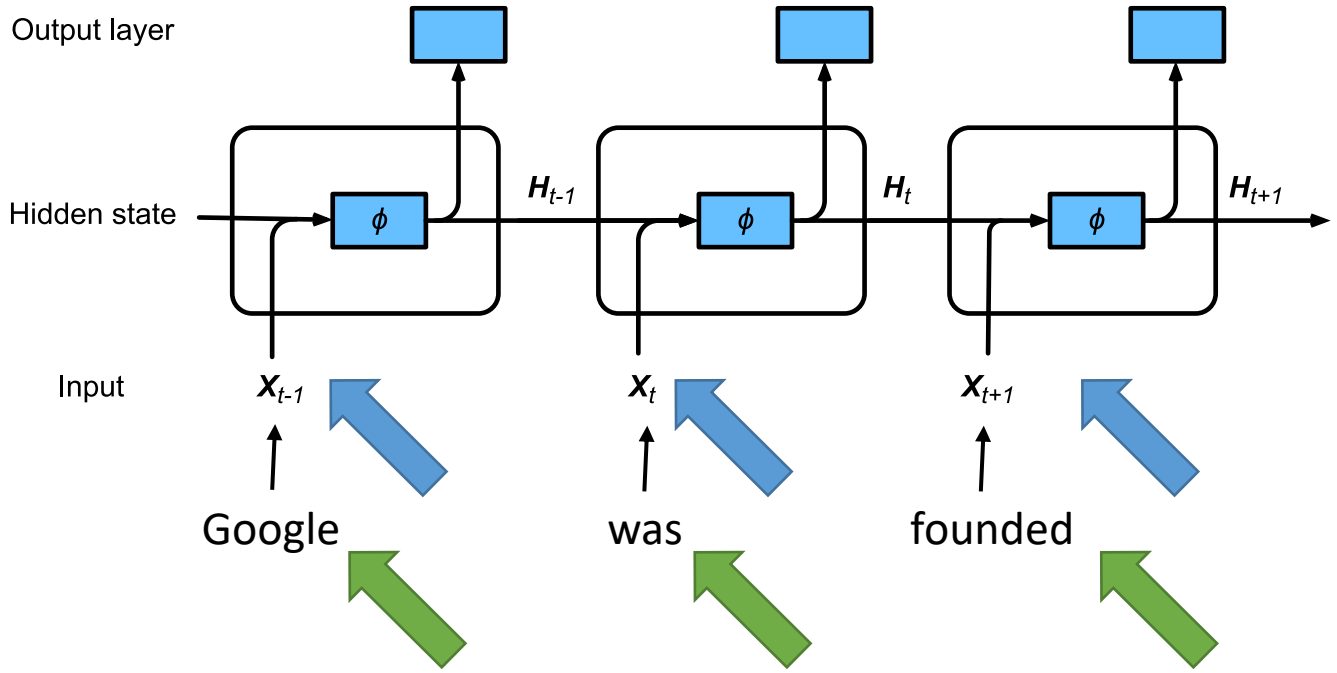
- In latent auto-regressive models, the model depends on the current input and a hidden state, capturing the past inputs:

$$o_t =\sim p(x_t|x_{t-1}, h_t)$$



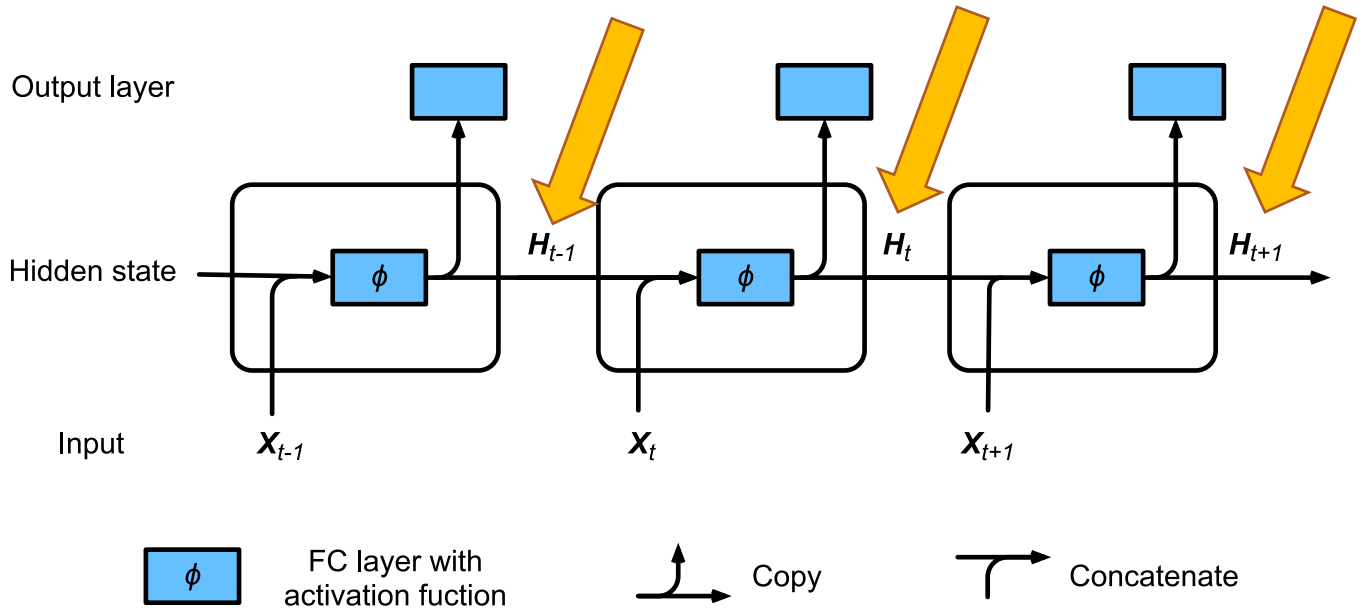
Input symbol and embedding

- **Input symbol** can be a word, any other symbol, or it may not exist because it is a direct measure.
- **Input embedding** is the vectorial representation of the symbol.



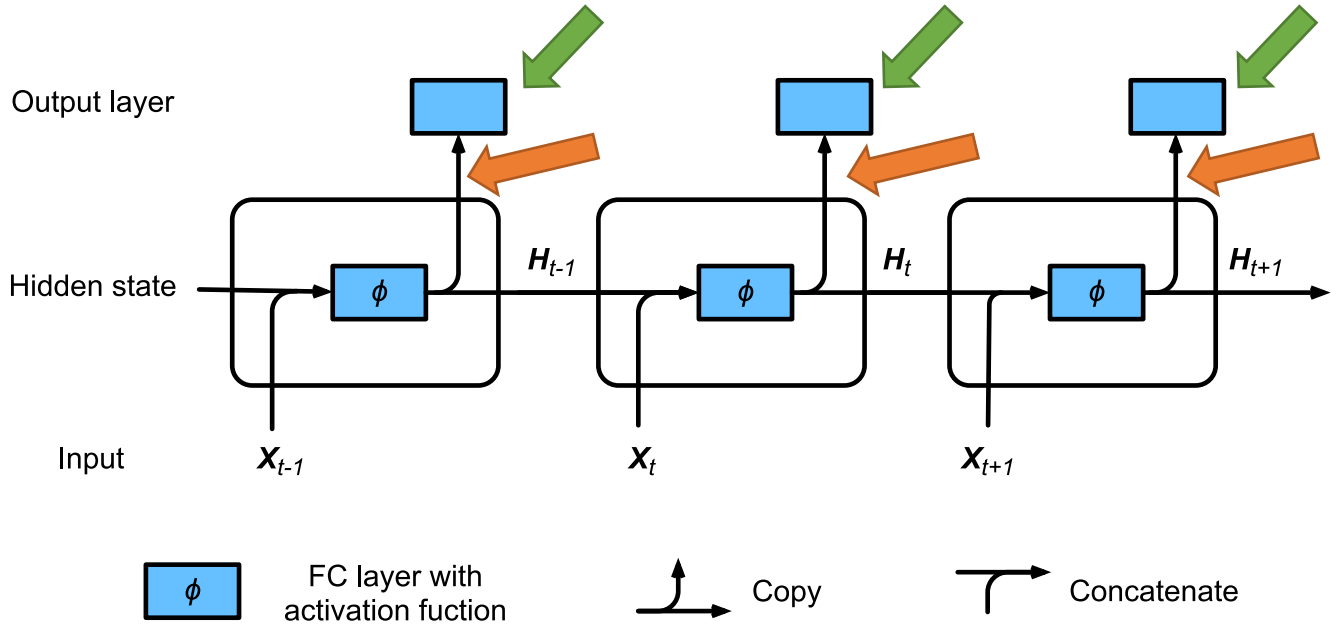
Hidden state

- **The hidden state**:
  - is propagated from state to state, and
  - it works as a memory to help decisions in later parts of the sequence.



Output embedding and symbol

- **Output embedding** the RNN prediction in the output space
- **Output symbol** is obtained by applying the softmax to the embedding to compute the most likely word



Any type of RNN can be used in any sequence task

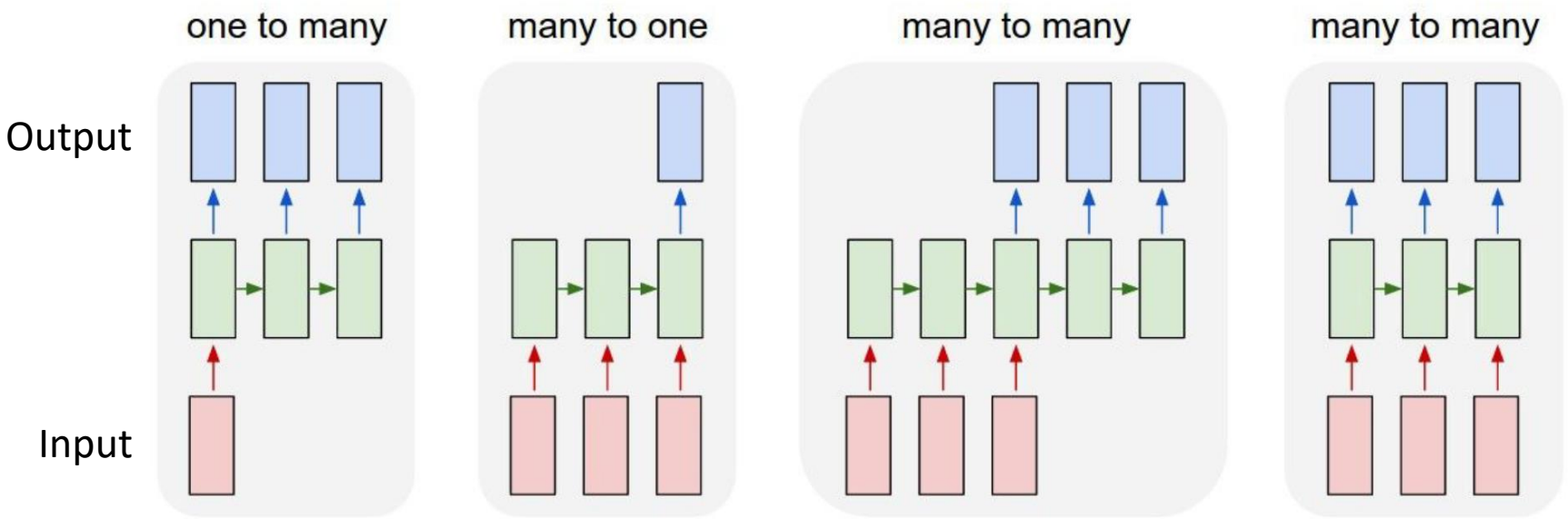


Image captioning example

This layer captures a good high level embedding of the image semantic content.

Image captioning example

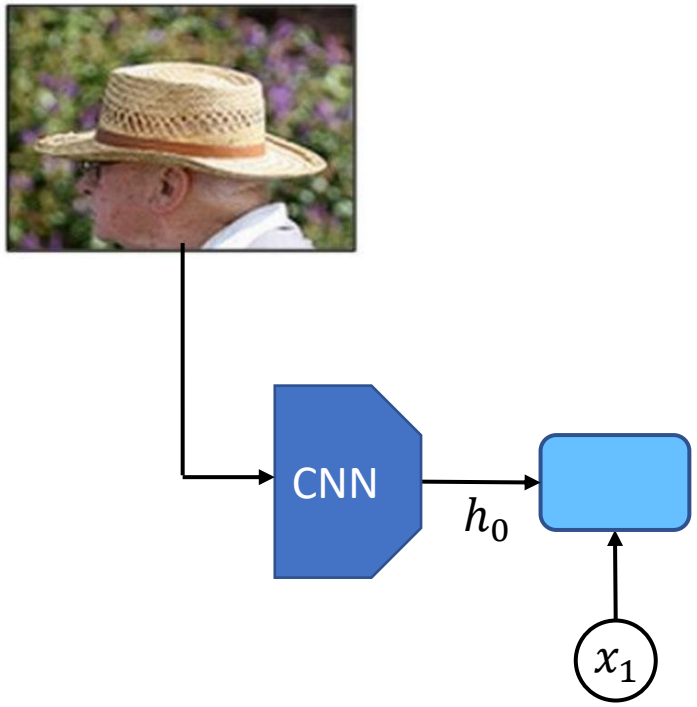


Image captioning example

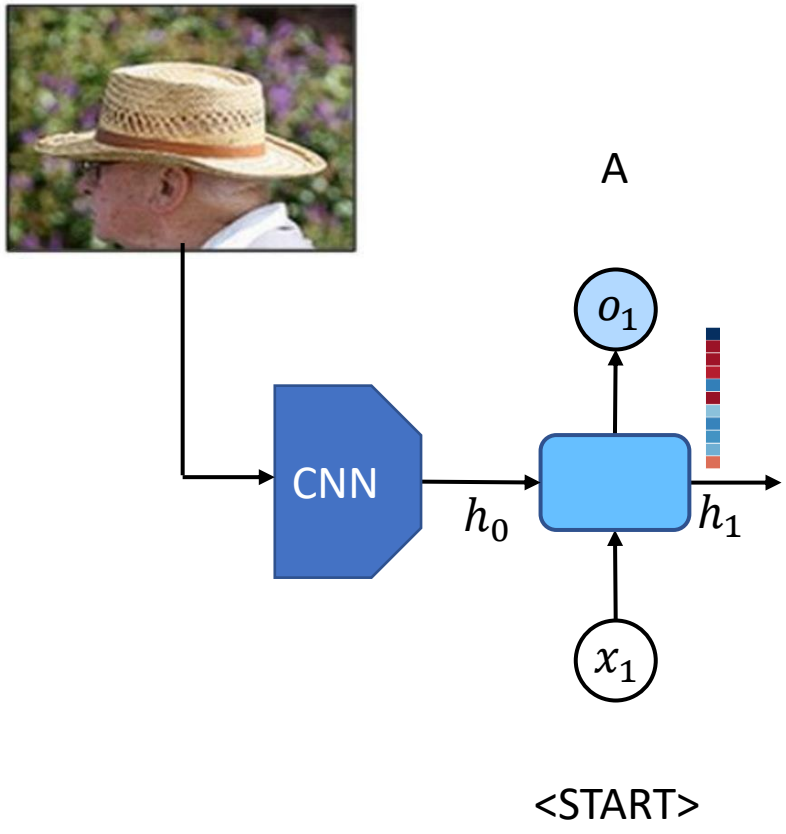


Image captioning example

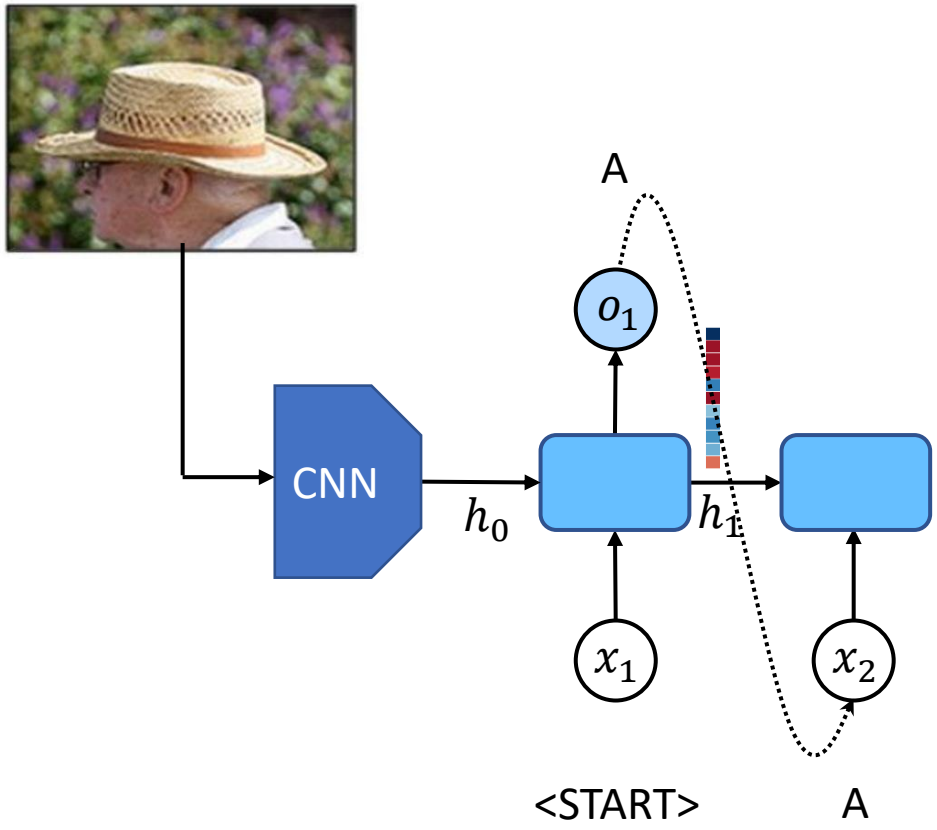


Image captioning example

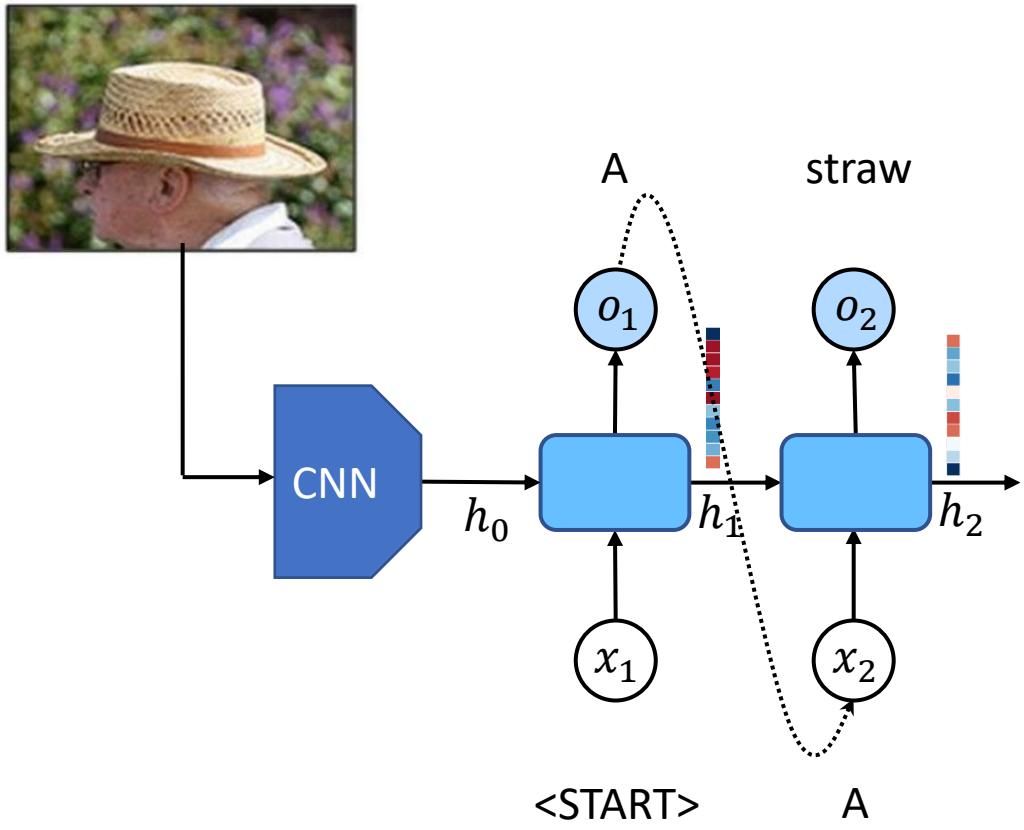


Image captioning example

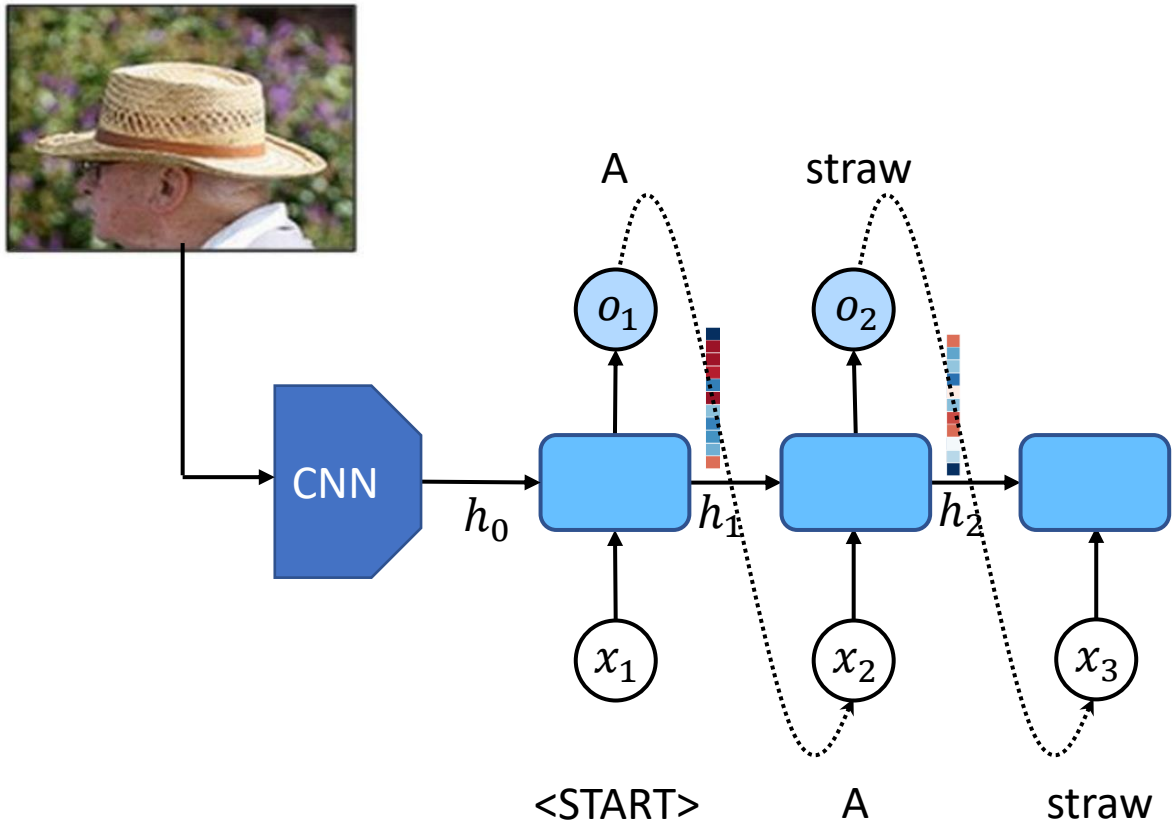


Image captioning example

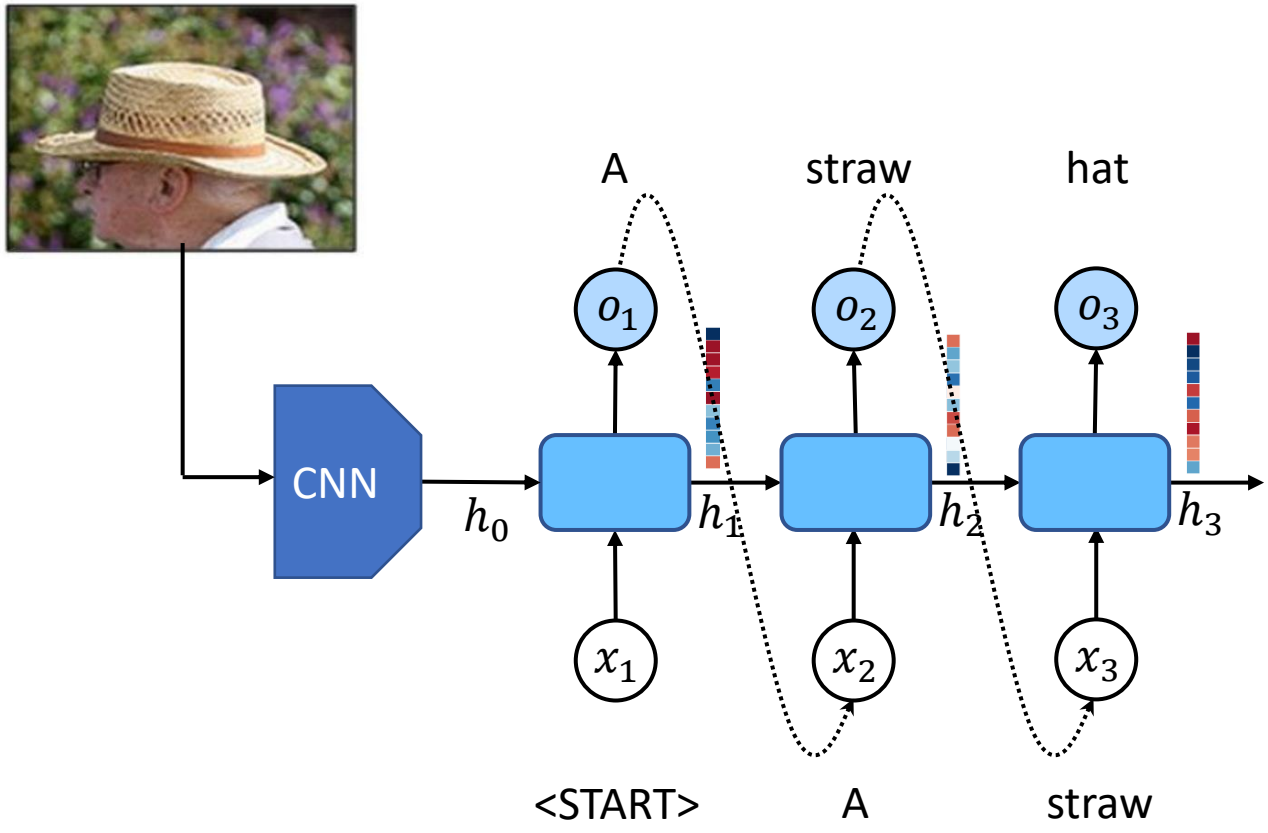


Image captioning example

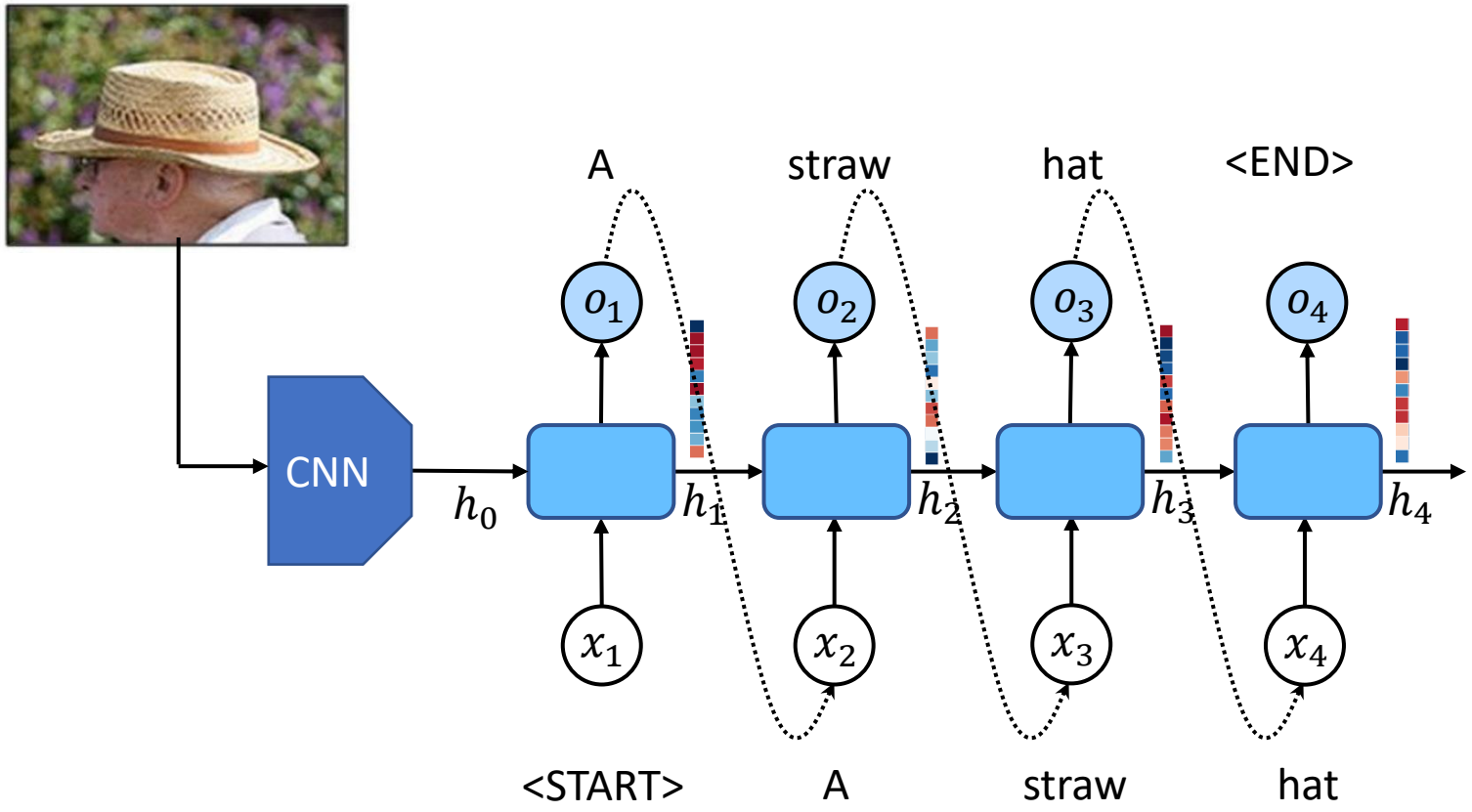


Image captioning examples

A cat sitting on a suitcase on the floor

A cat is sitting on a tree branch

A dog is running in the grass with a frisbee

A white teddy bear sitting in the grass

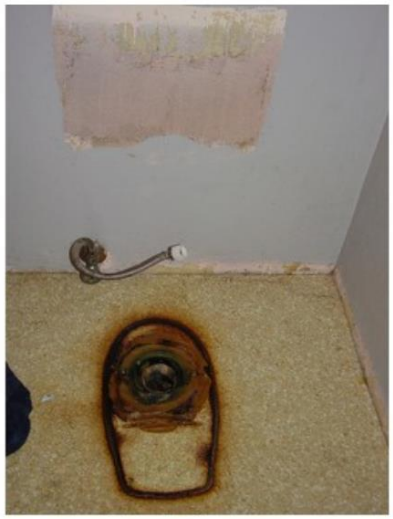
Two people walking on the beach with surfboards

A tennis player in action on the court

Two giraffes standing in a grassy field

A man riding a dirt bike on a dirt track

Image captioning (bad) examples

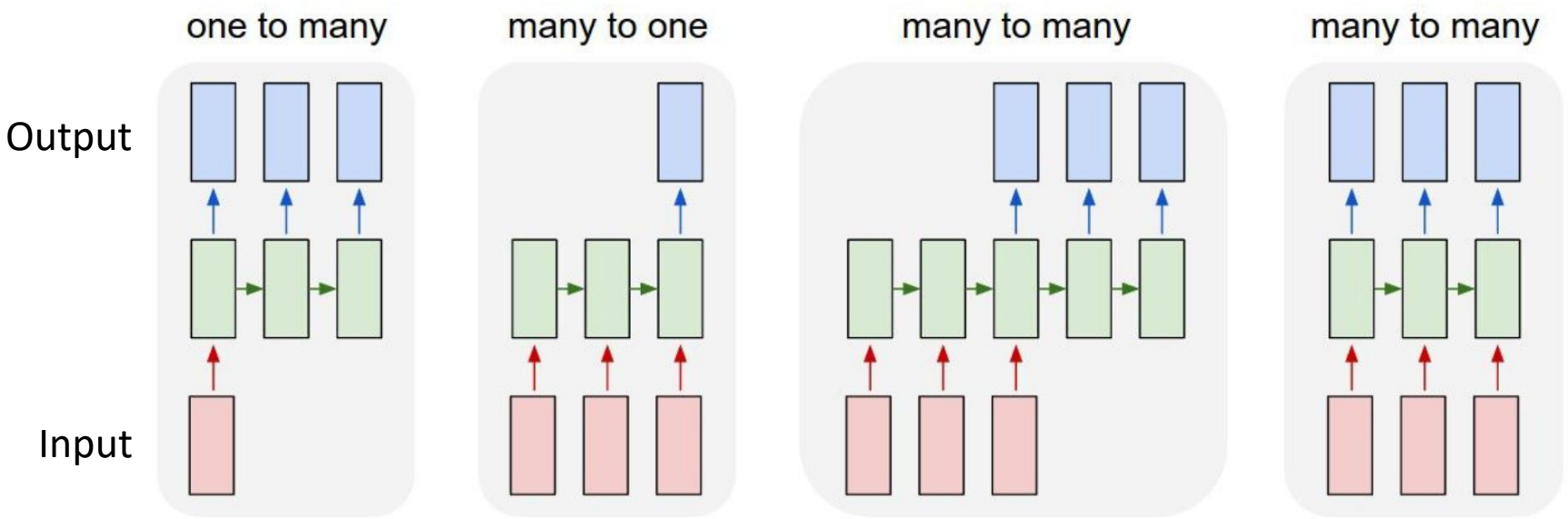


a toilet with a seat up in a bathroom
logprob: -13.44

a woman holding a teddy bear in front of a mirror
logprob: -9.65

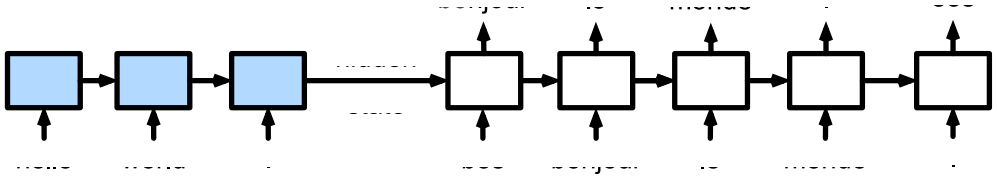
a horse is standing in the middle of a road
logprob: -10.34

Any type of RNN can be used in any sequence task



Sequence to sequence

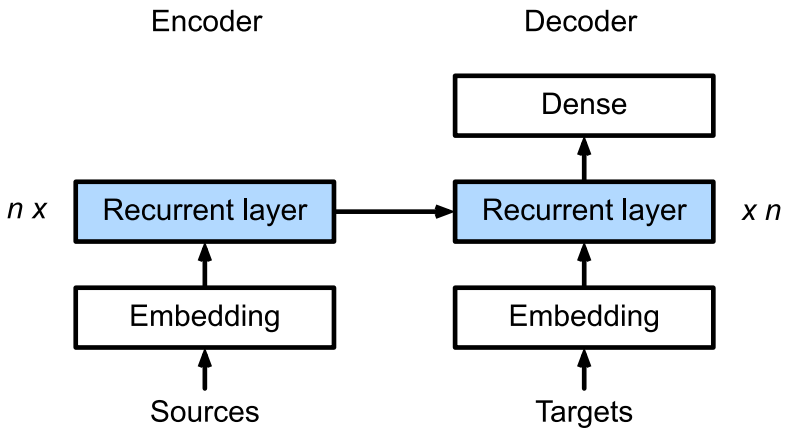
- The sequence to sequence (seq2seq) model is based on the encoder-decoder architecture to generate a sequence output for a sequence input



- Both the encoder and the decoder use RNNs to handle sequence inputs of variable length.
- The hidden state of the encoder is used directly to initialize the decoder hidden state.

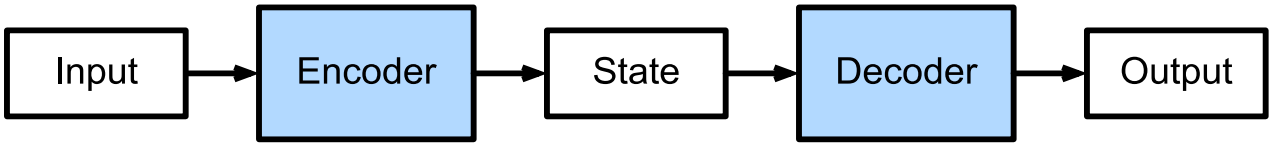
Sequence to sequence

- Sequence to sequence models are a special case of encoder-decoder architectures.
- The hidden state of the encoder is used directly to initialize the decoder hidden state to pass information from the encoder to the decoder.



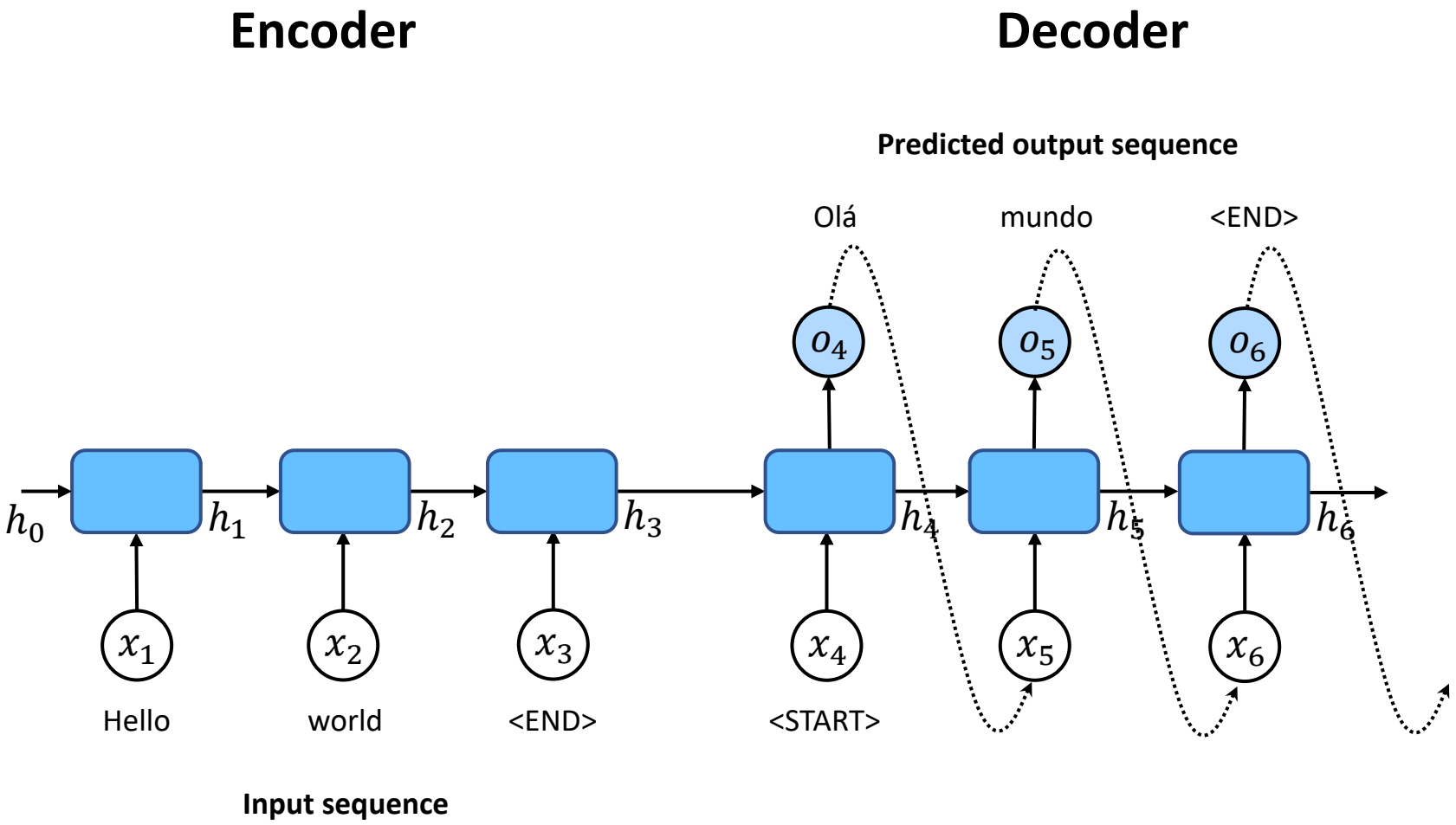
Encoder-decoder

- The encoder-decoder is a design pattern.
- The encoder's role is to encode the inputs into state, which often contains several tensors.
- Then the state is passed into the decoder to generate the outputs.

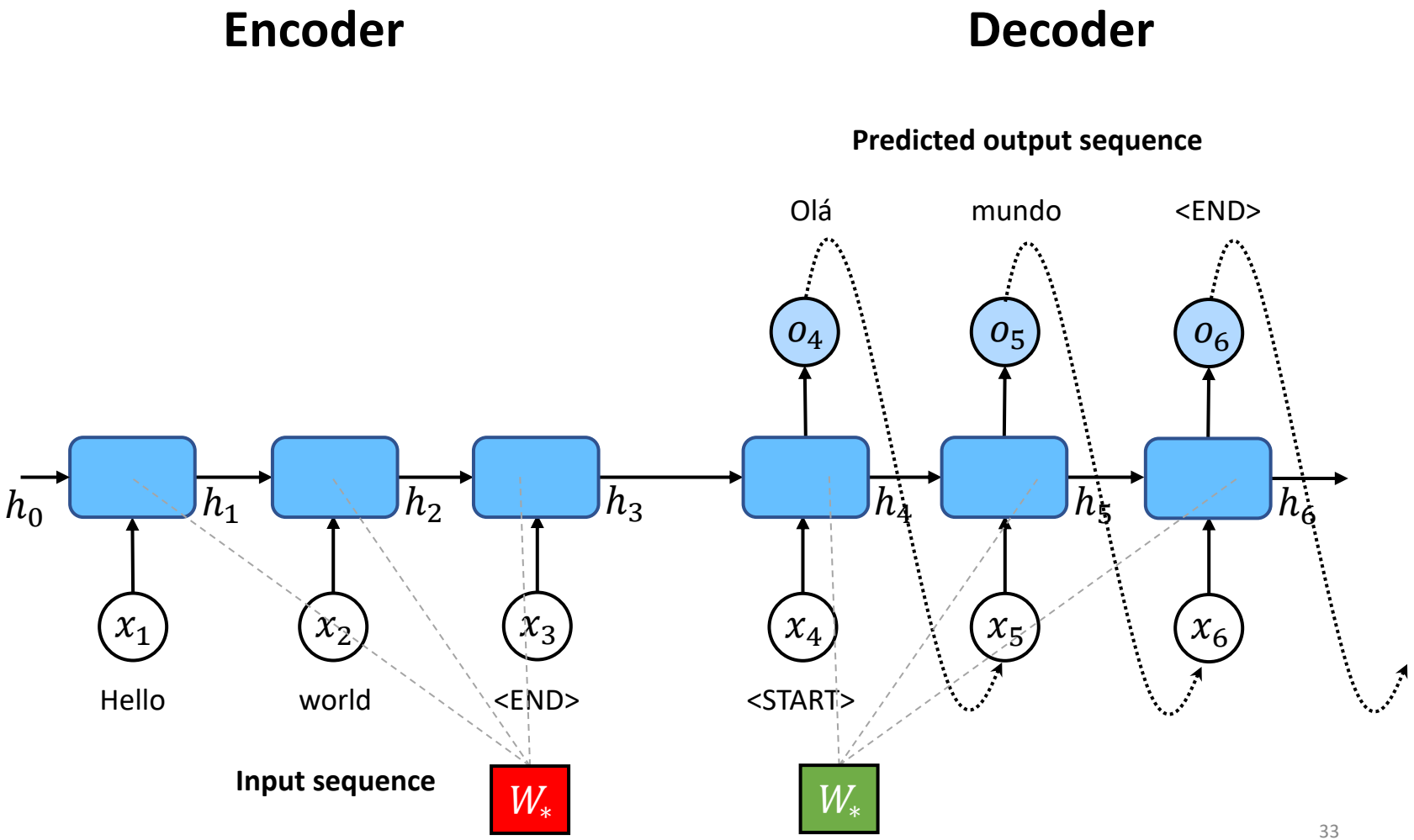


- The encoder and the decoder can have different architectures, e.g., in image captioning the encoder is a CNN and the decoder is an RNN.

Feedforward in sequence-to-sequence models



Encoder and decoder have different parameters

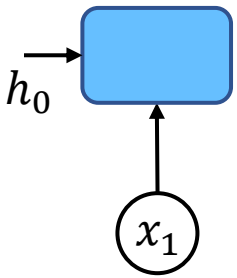


Types of RNNs

- Recurrent Neural Networks
- Gated Recurrent Neural Network
- Long-term Short-Term Memory

  - All three types of RNNs, try to tackle the memory problem of RNNs, also known as vanishing or exploding gradient problem.
  - There are many variations of each one of these three types of RNNs.

Feedforward

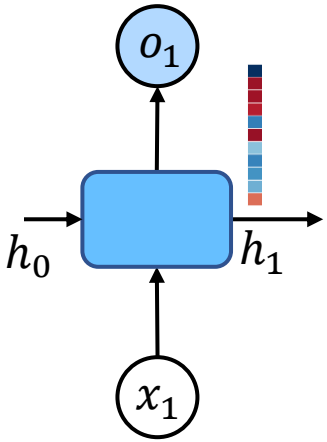


Output predictions

States

Inputs

Feedforward

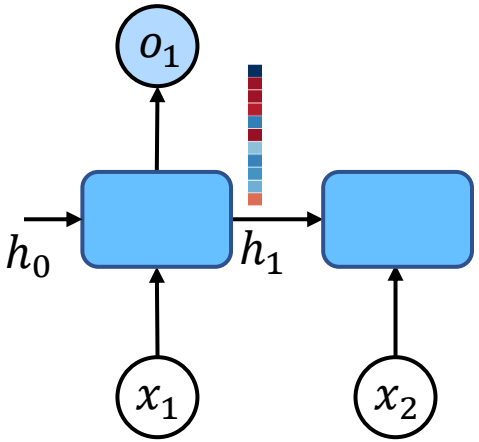


Output predictions

States

Inputs

Feedforward

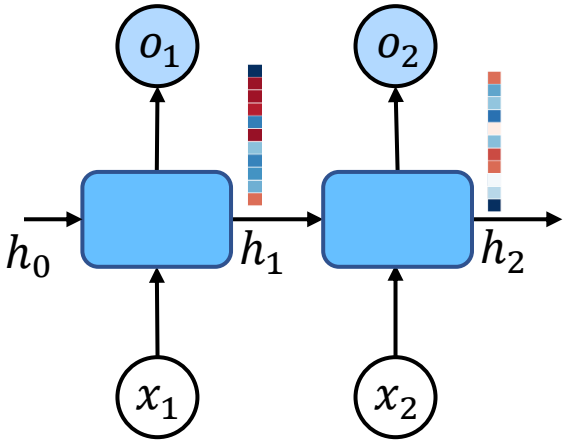


Output predictions

States

Inputs

Feedforward

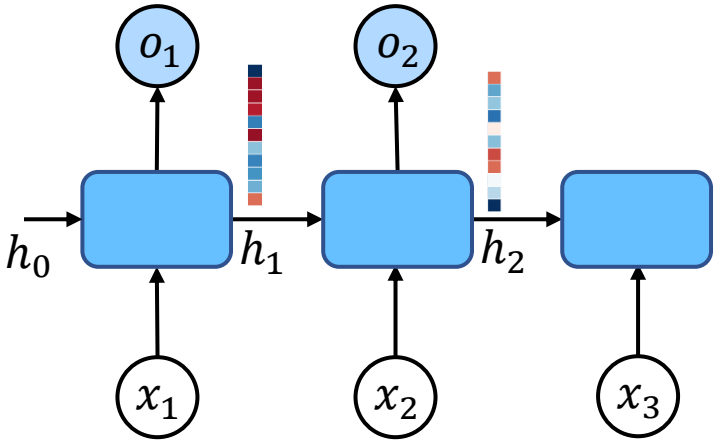


Output predictions

States

Inputs

Feedforward

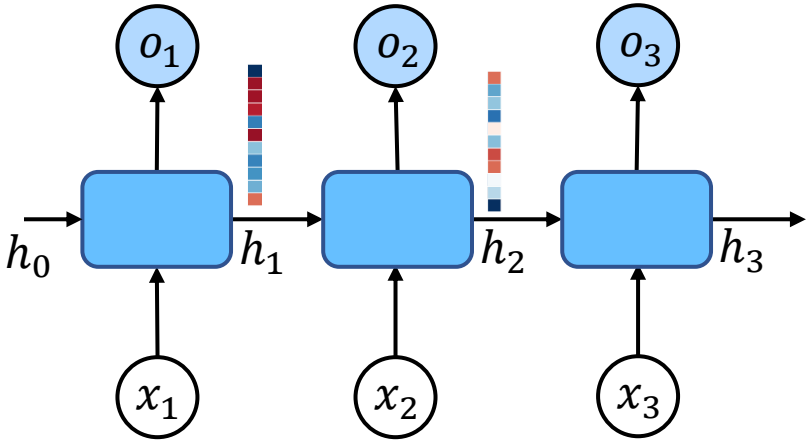


Output predictions

States

Inputs

Feedforward

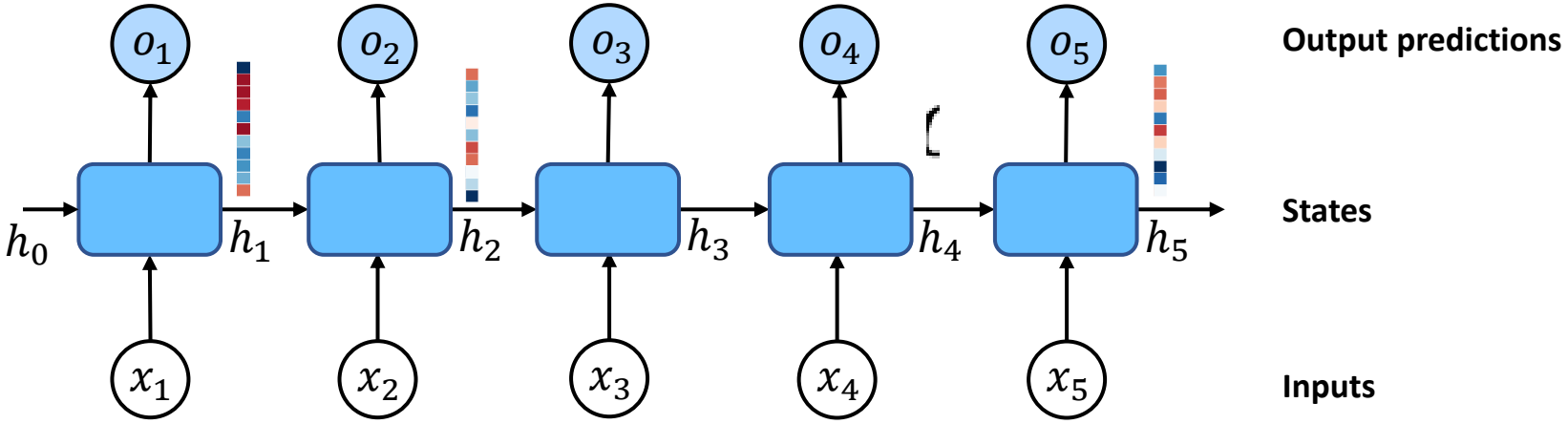


Output predictions

States

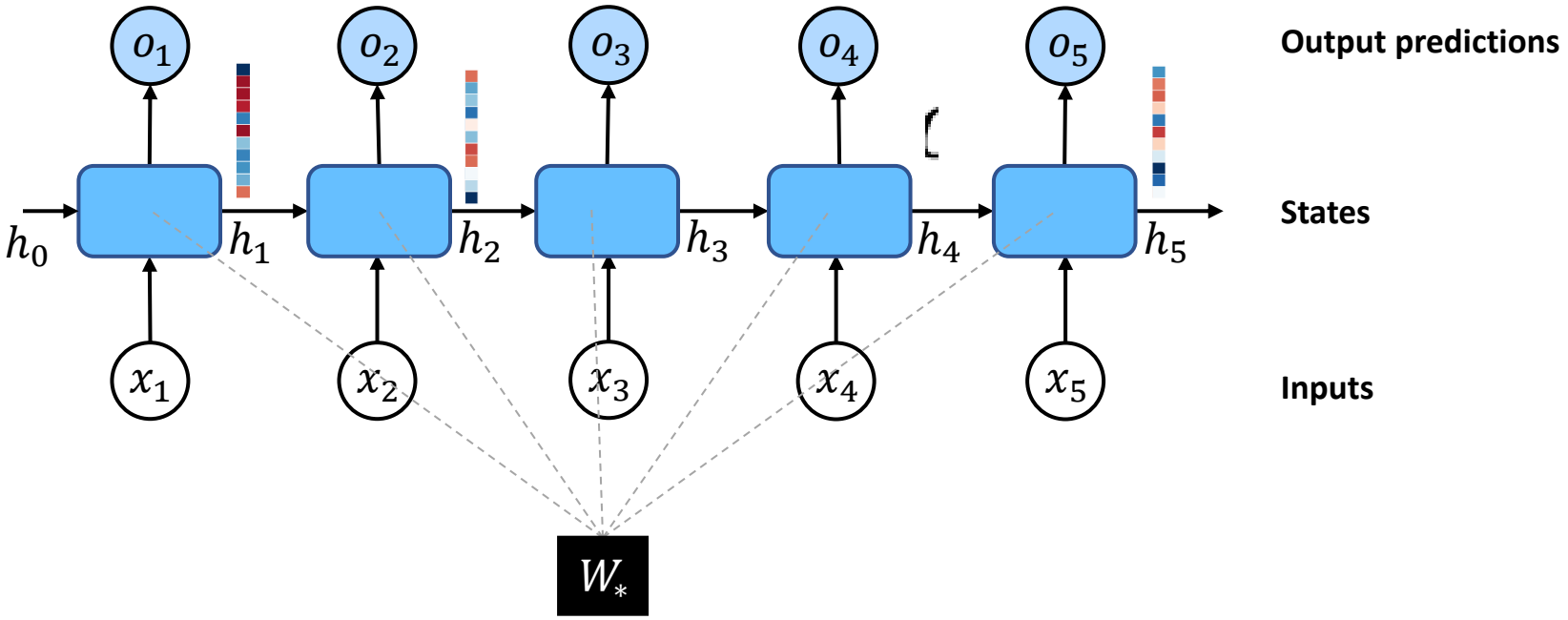
Inputs

Feedforward



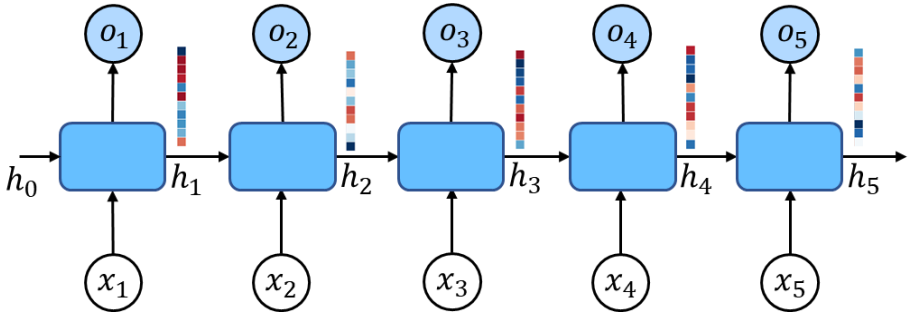
Parameter sharing

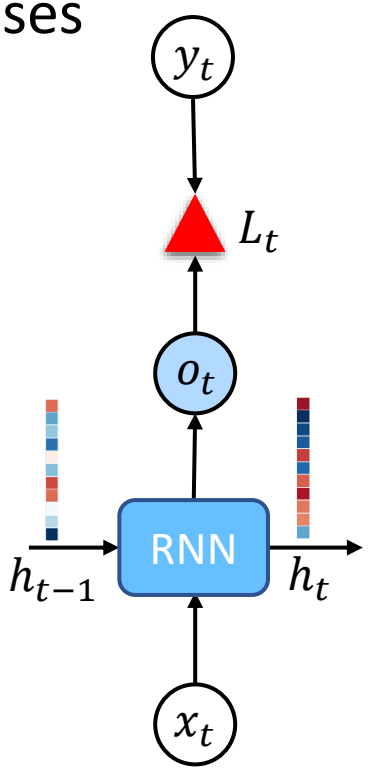
- There is only one RNN that runs through the sequence
- It has only one set of parameters



Elman's Recurrent Neural Network

- Elman's Recurrent Neural Network (RNN) processes sequences of data by processing each step t.
- The RNN unit is a recurrent function.
- The goal is to predict the output y at each step t.





Elman's Recurrent Neural Network

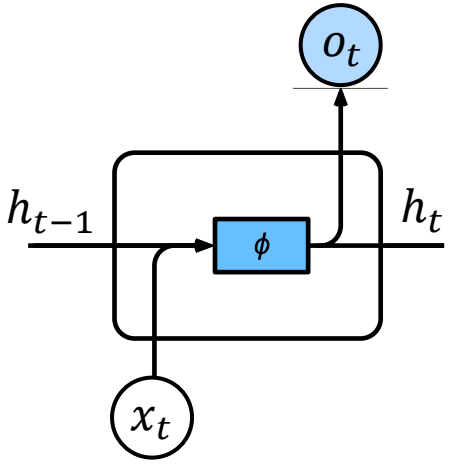
- The RNN unit preserves information from:
  - previous state h_{t-1};
  - current input data input x_t.

- The RNN unit is a recurrent function:

$$h_t = \phi\left([W_{hh}\ W_{xh}] \cdot \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix} + b_h\right)$$

  - W_{xh} and W_{hh} are the RNN parameters matrix
  - The output at each step t is:

$$o_t = h_t \cdot W_{hq} + b_q$$

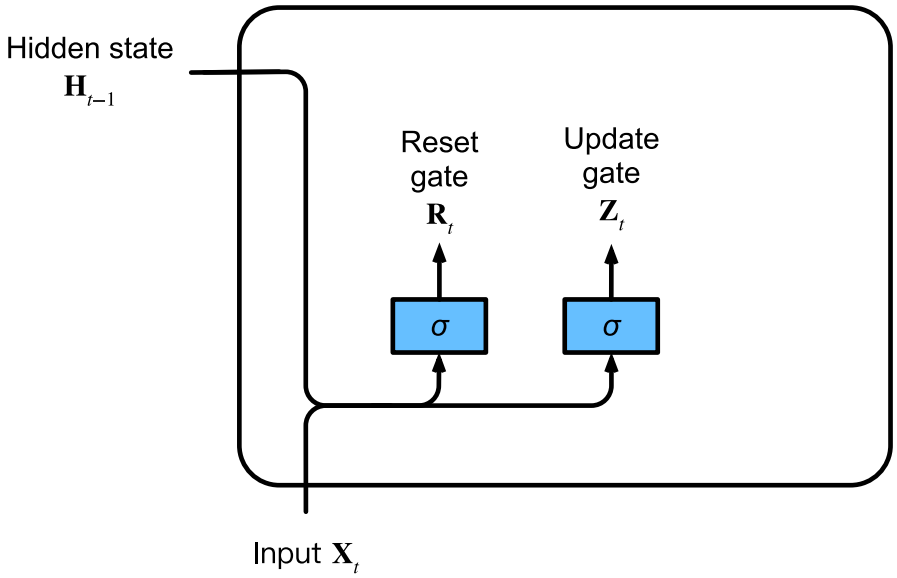


State and output are too tightly connected

- In Elman's RNN the output and the state are derived from the same variable.
- A unit's state should give more emphasis to the current data or the previous state.
- State passing mechanism should be able to control the amount of information from data and/or previous state that is encoded in a state
  - A better approach is to make a stronger separation between state and output.
- GRUs and LSTMs are the best examples of such idea.

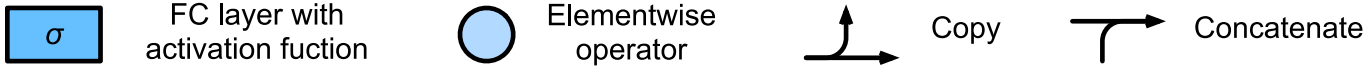
Gated Recurrent Unit: Old state information

- Introduces a reset memory and update state functions



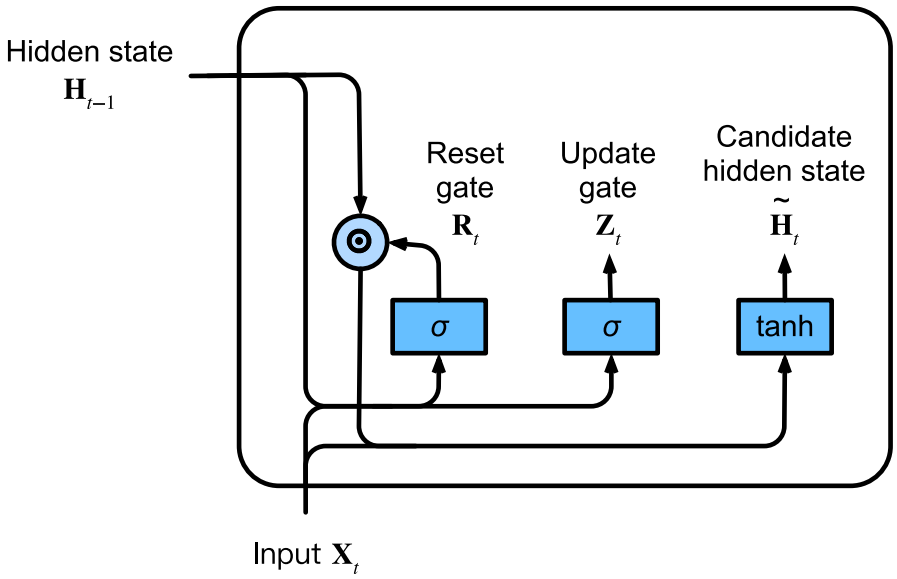
$$R_t = \sigma_r\left(W_r \cdot \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix}\right)$$

$$Z_t = \sigma_t\left(W_u \cdot \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix}\right)$$



Gated Recurrent Unit: New candidate state

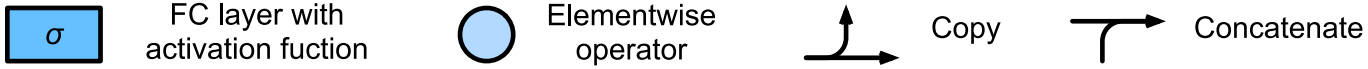
- Introduces a reset memory and update state functions
- Computes a candidate hidden state



$$R_t = \sigma_r\left(W_r \cdot \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix}\right)$$

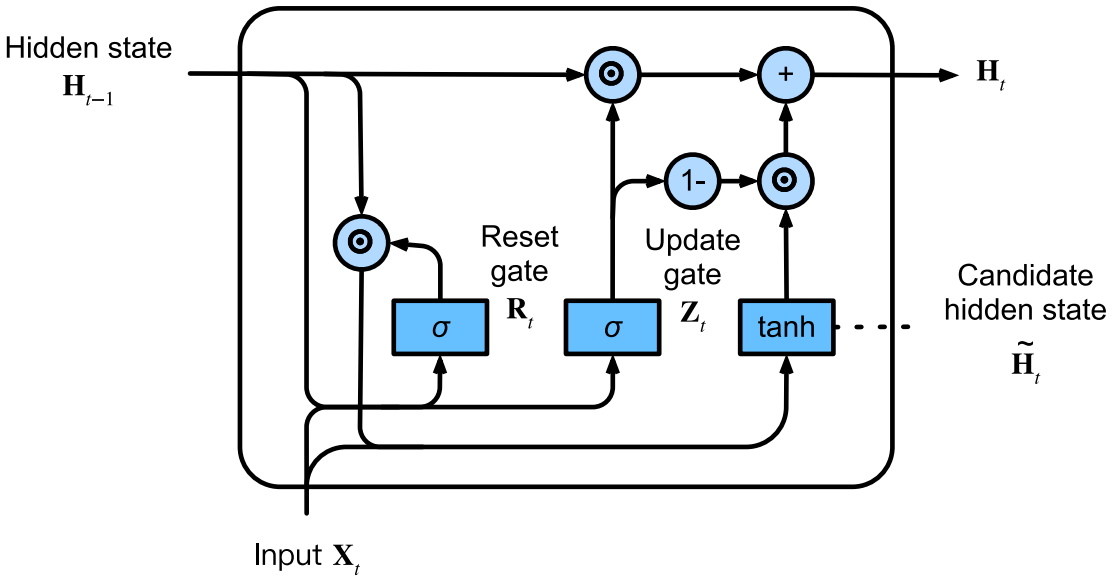
$$Z_t = \sigma_t\left(W_u \cdot \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix}\right)$$

$$\tilde{h}_t = tanh\left(W_c \cdot \begin{bmatrix} R_t \odot h_{t-1} \\ x_t \end{bmatrix}\right)$$



Gated Recurrent Unit

- The new hidden state is a mixture of the candidate hidden state and the previous hidden state.

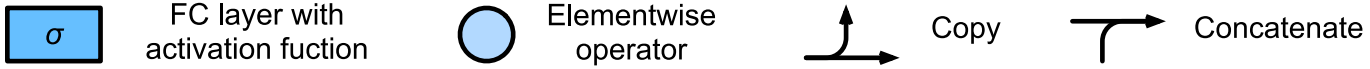


$$R_t = \sigma_r\left(W_r \cdot \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix}\right)$$

$$Z_t = \sigma_t\left(W_u \cdot \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix}\right)$$

$$\tilde{h}_t = tanh\left(W_c \cdot \begin{bmatrix} R_t \odot h_{t-1} \\ x_t \end{bmatrix}\right)$$

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



Gated Recurrent Unit

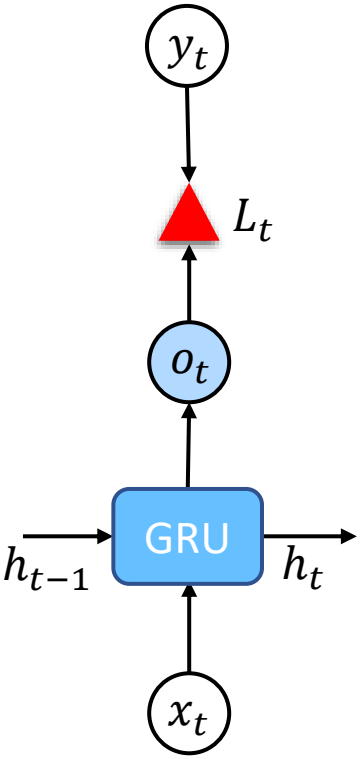
- The reset gate controls the **information that is deleted** from the previous state:

$$R_t = \sigma_r\left(W_r \cdot \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix}\right) \qquad \tilde{h}_t = tanh\left(W_c \cdot \begin{bmatrix} R_t \odot h_{t-1} \\ x_t \end{bmatrix}\right)$$

- The update gate controls the **information that is preserved** from the previous state:

$$h_t = (1 - Z_t) \odot \tilde{h}_t + Z_t \odot h_{t-1} \qquad Z_t = \sigma_t\left(W_u \cdot \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix}\right)$$

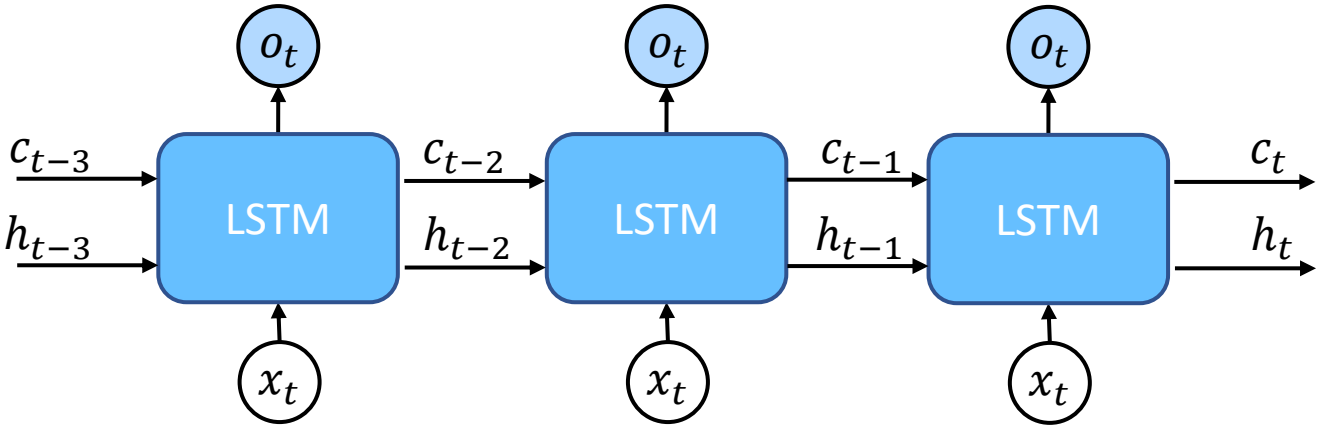
- The output is computed from the cell memory: $o_t = g_y(c_t)$



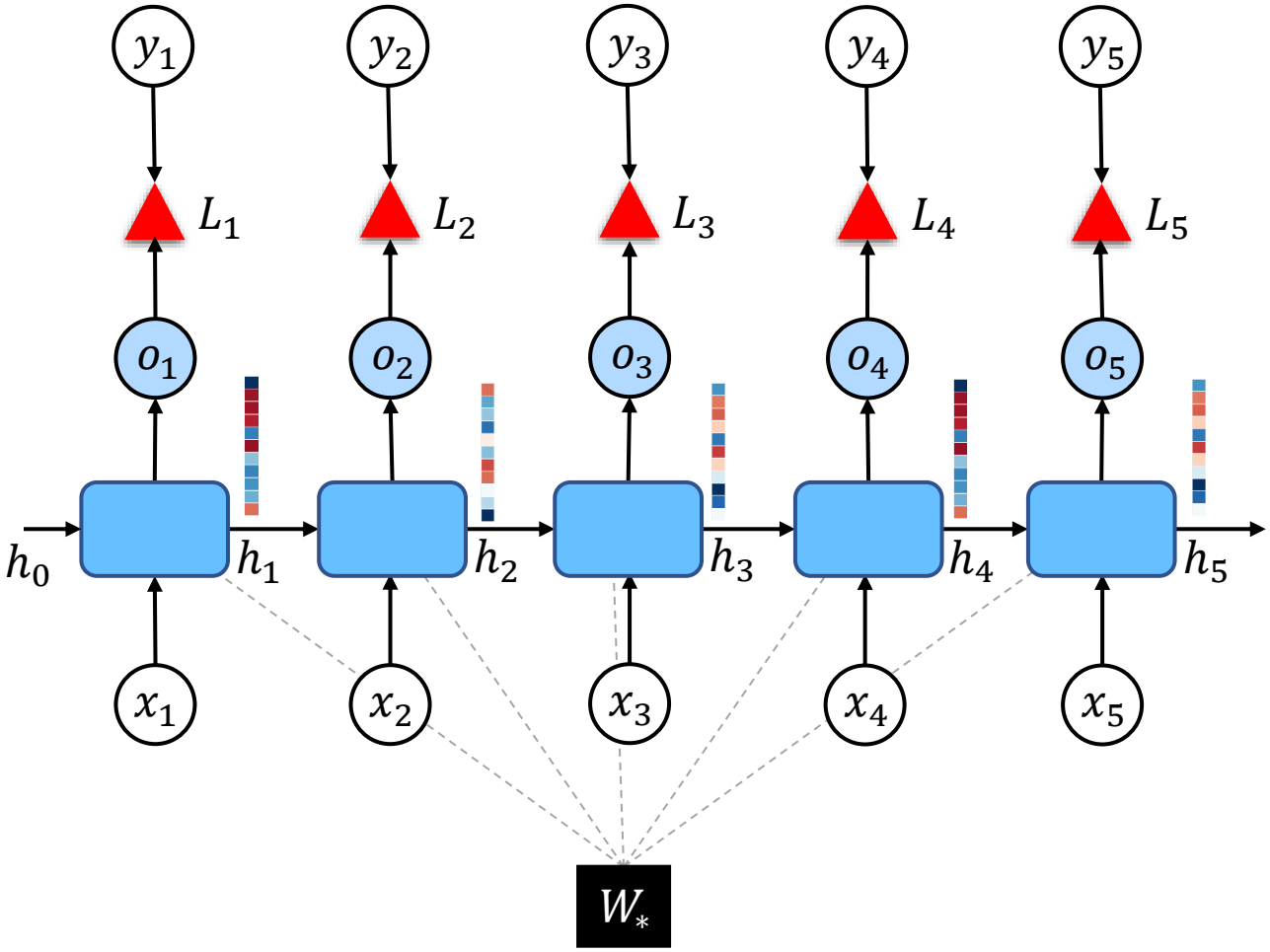
Long-Short Term Memory

- **Key idea 1:** Separate **state** from **memory**.
  - This allows to better preserve memory from past states and still generate the correct output from the hidden state.

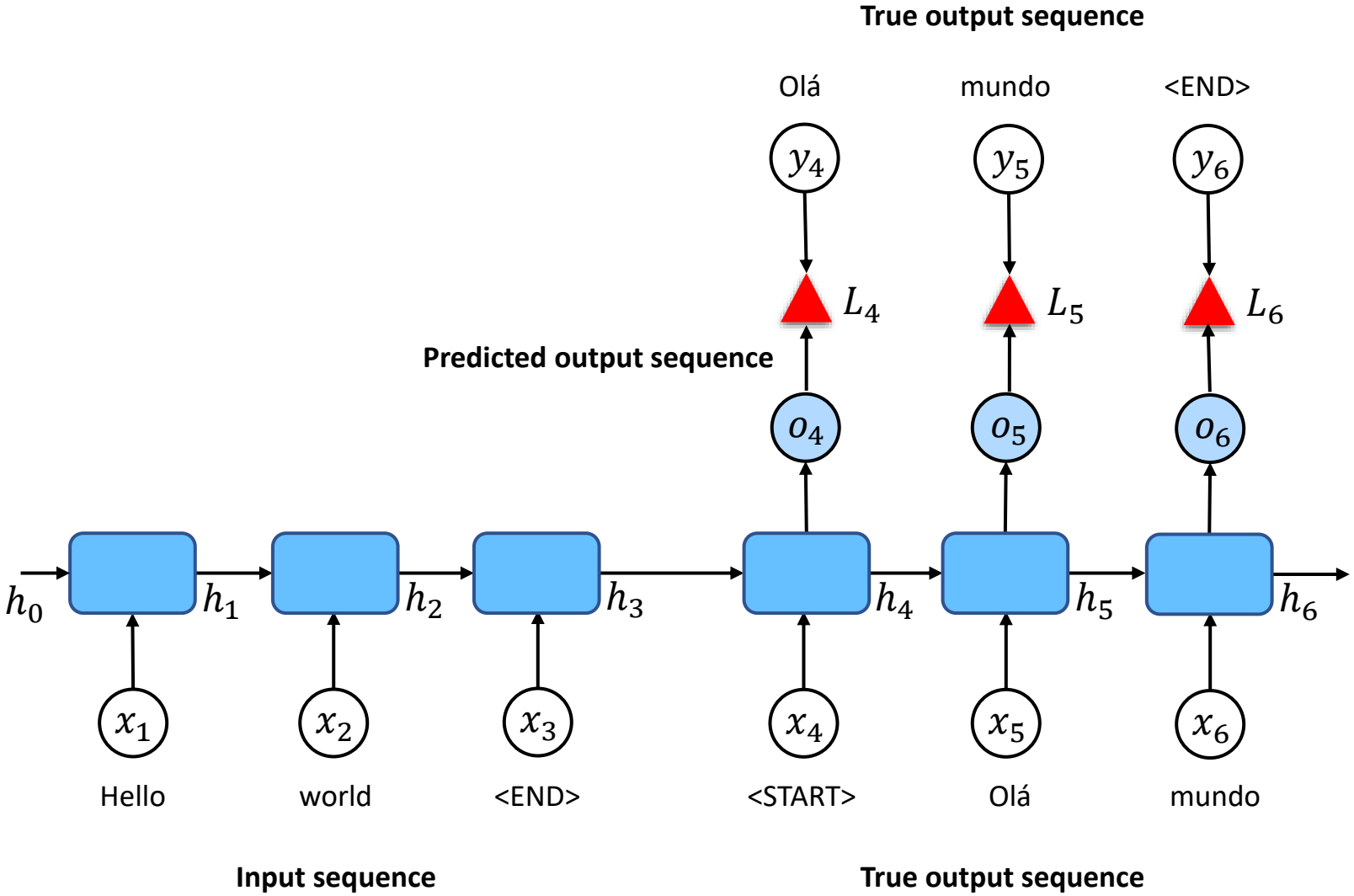
- **Key idea 2:** Put the memory along an **uninterrupted path**.
  - It avoids the vanishing gradient problem and lets information propagate backwards.



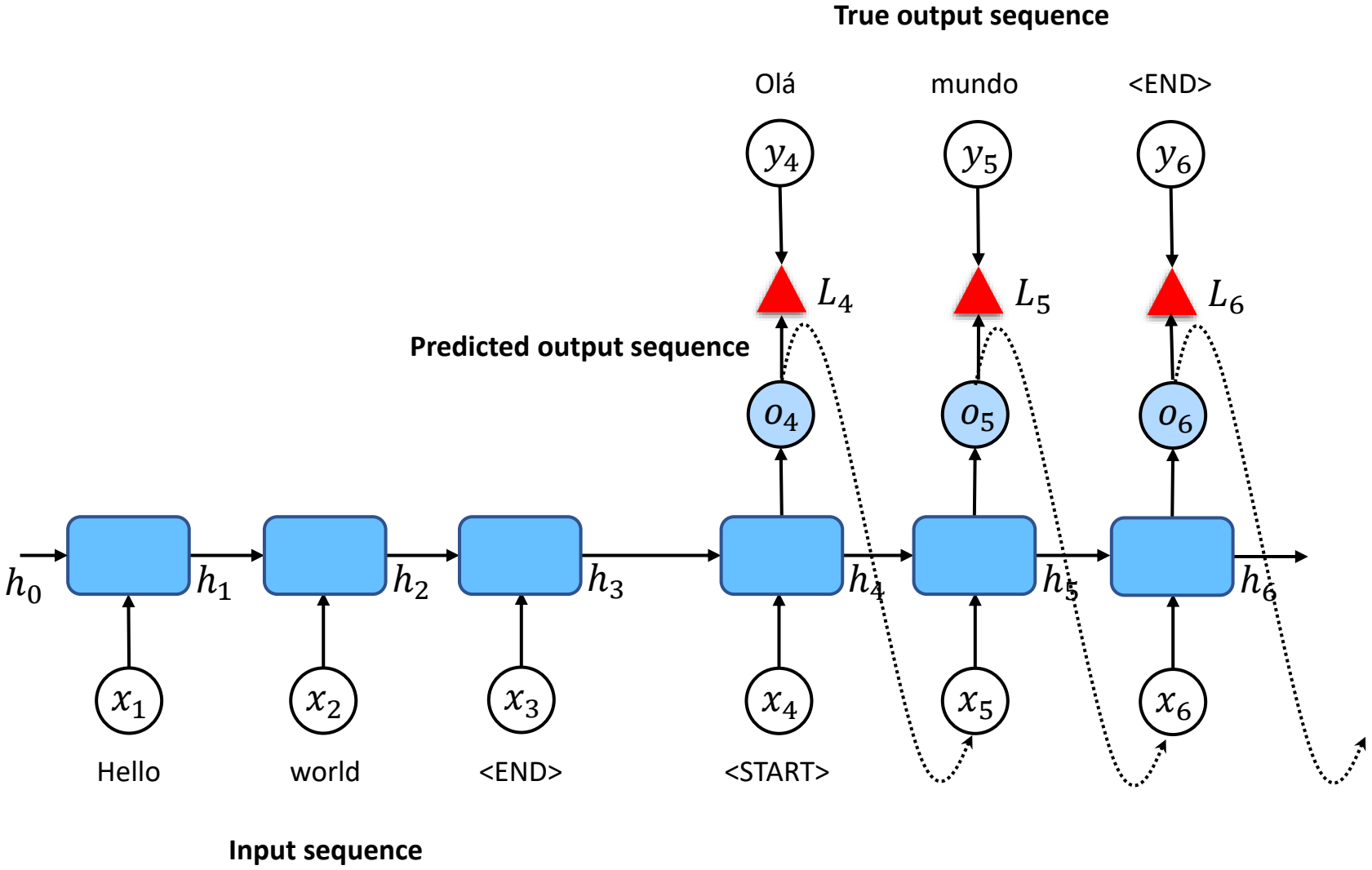
Training loss



Training with teacher forcing



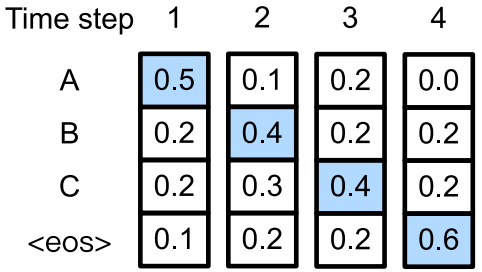
Training with reinjection



Inference – Greedy Search

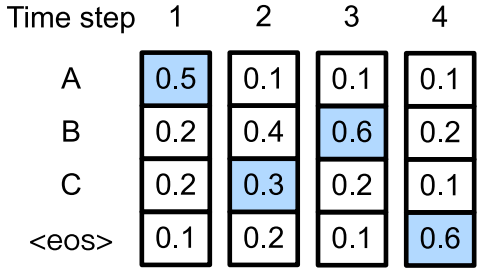
- The conditional probability of this output sequence is

  $0.5 \times 0.4 \times 0.4 \times 0.6 = 0.048$

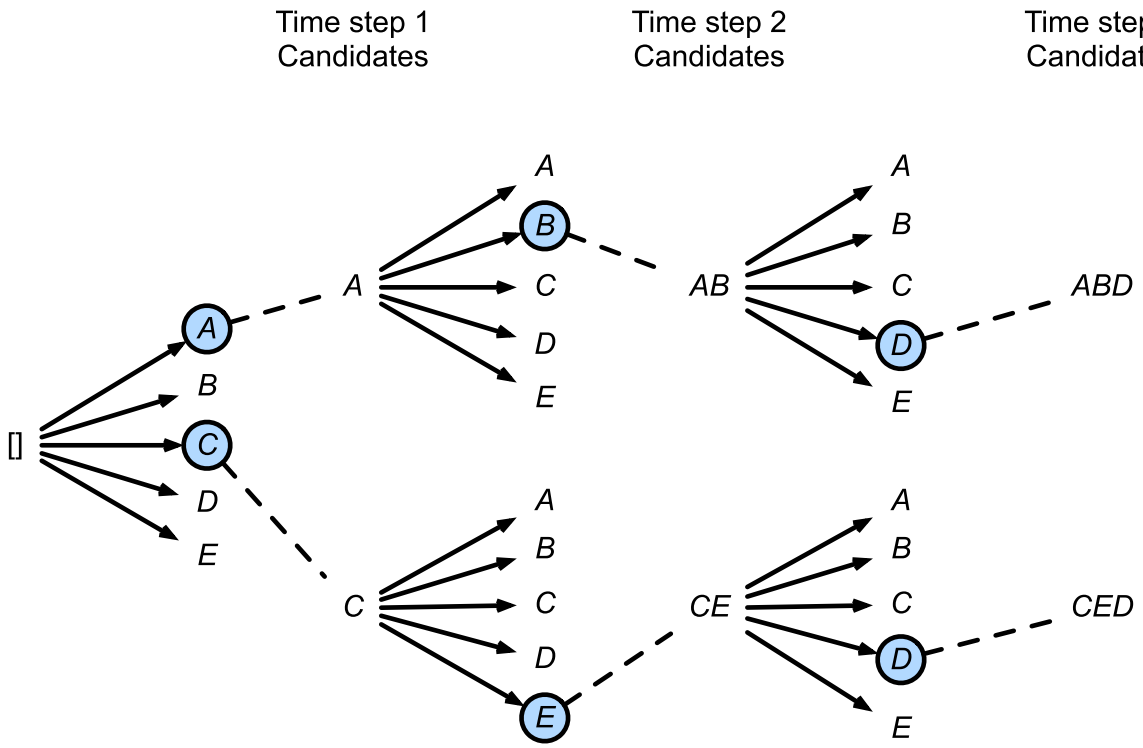


- The conditional probability of the output sequence “A”, “C”, “B”, and “<eos>” is

  $0.5 \times 0.3 \times 0.6 \times 0.6 = 0.054$

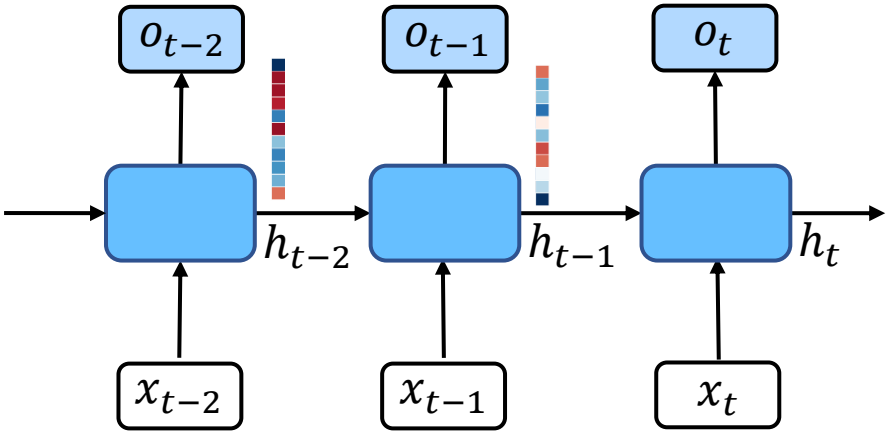


Inference – Beam Search



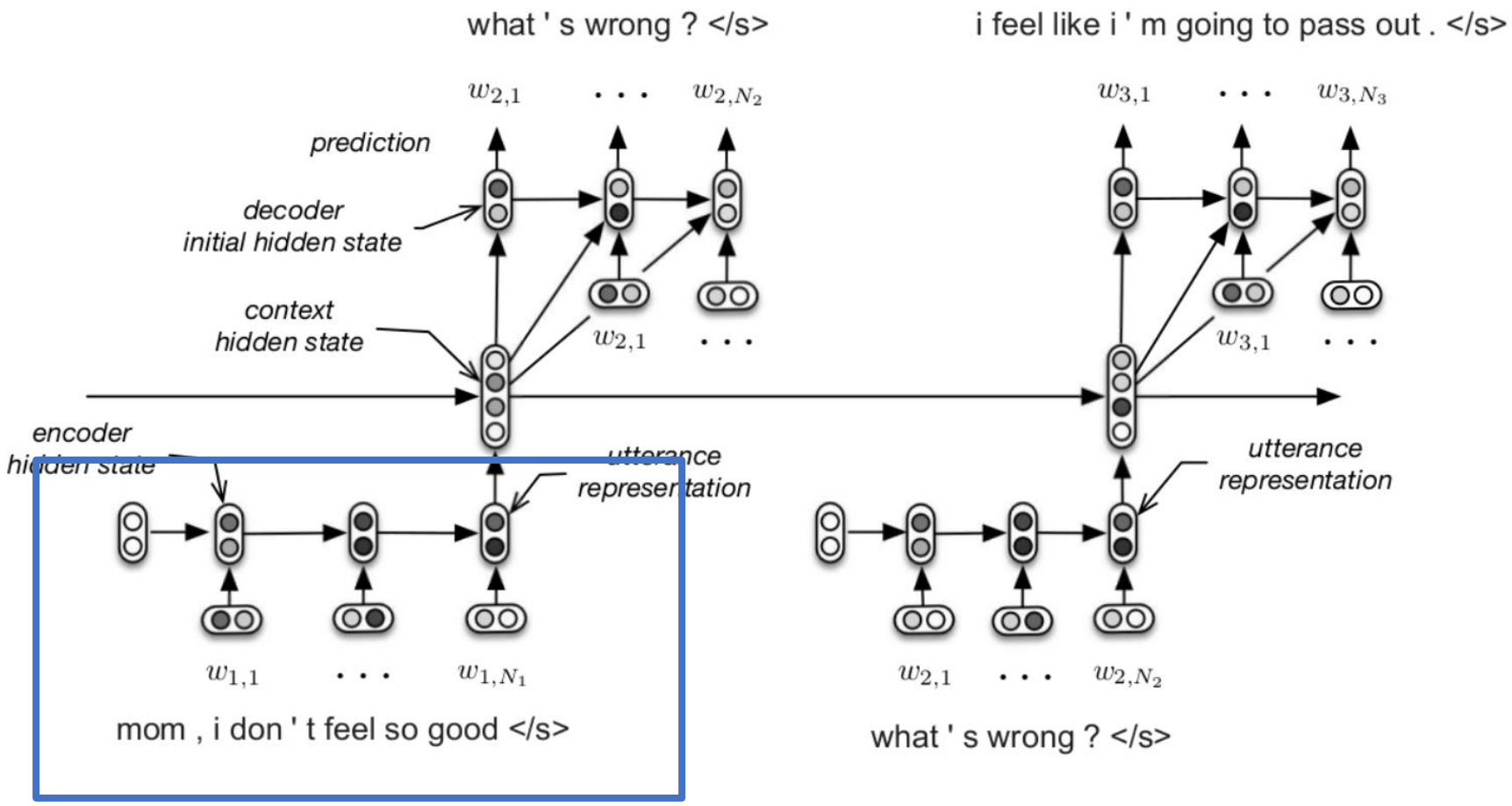
RNN-based dialog state tracking

- A first RNN is used to track the state of the conversation.
- It's input data are the utterances of the conversation.
- Agent utterances can be generated from the conversation state.



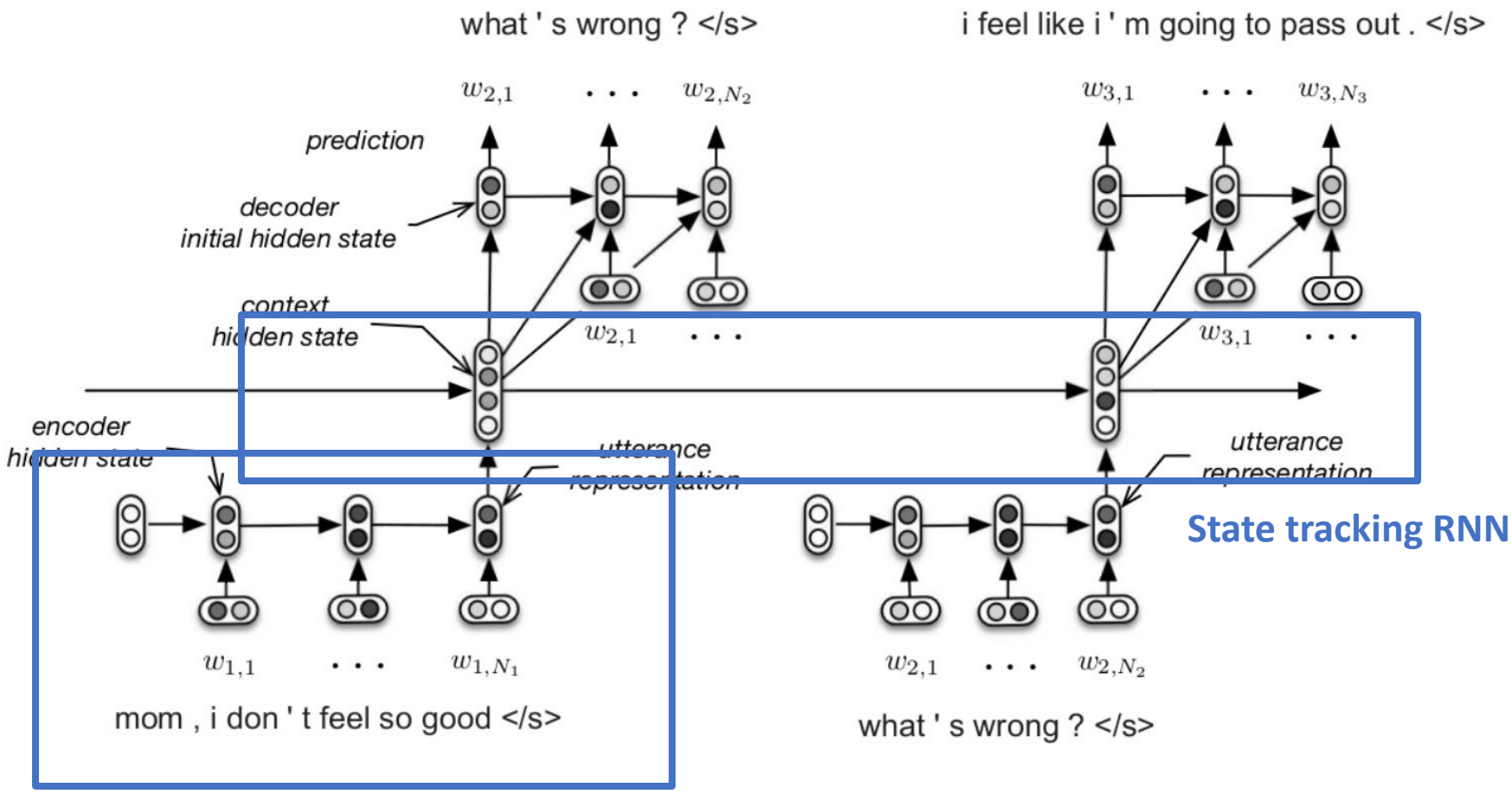
Serban, Iulian V., Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2015. "Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models.
http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/download/11957/12160

Hierarchical Recurrent Encoder-Decoder

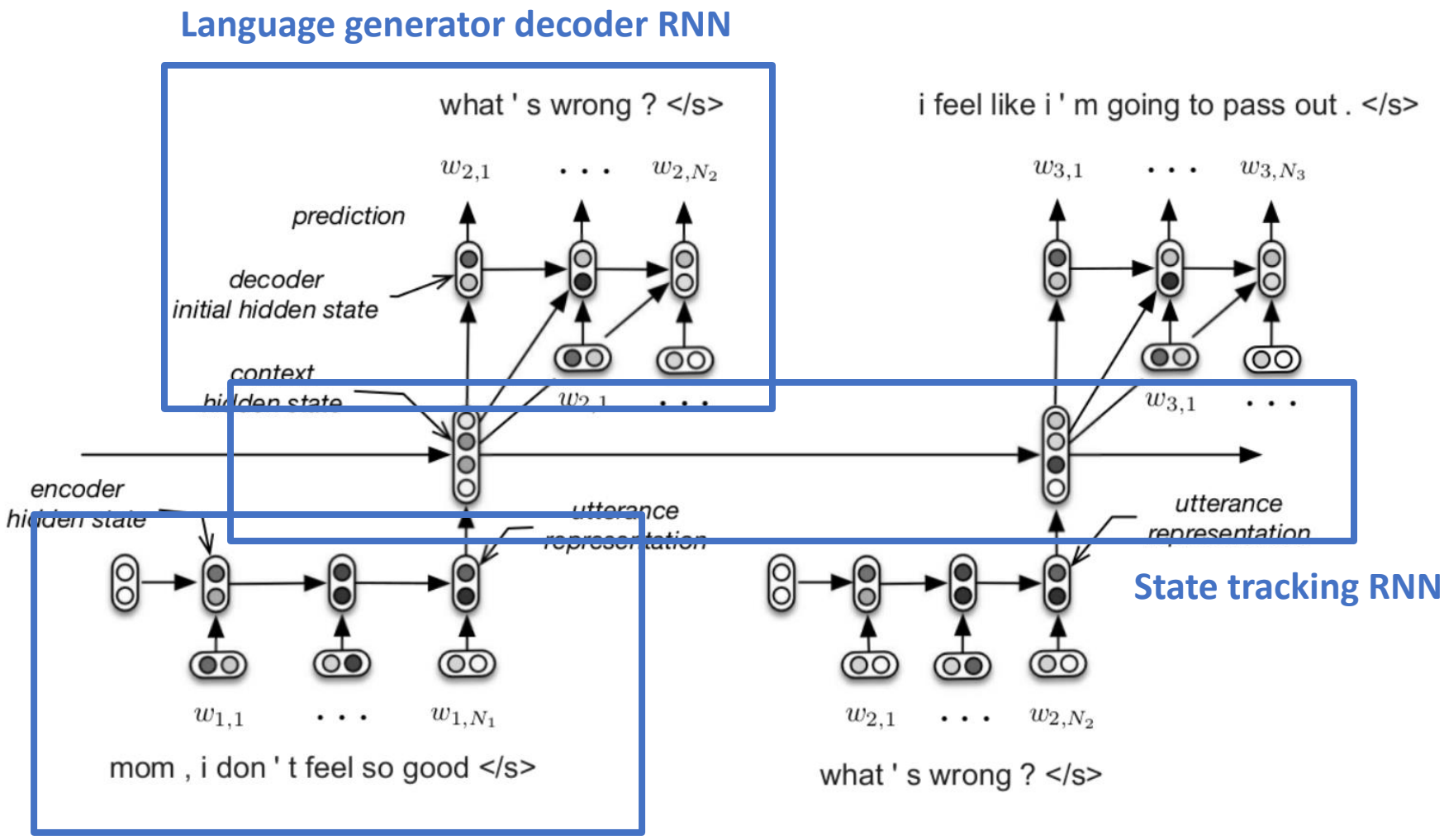


Utterance's encoder RNN

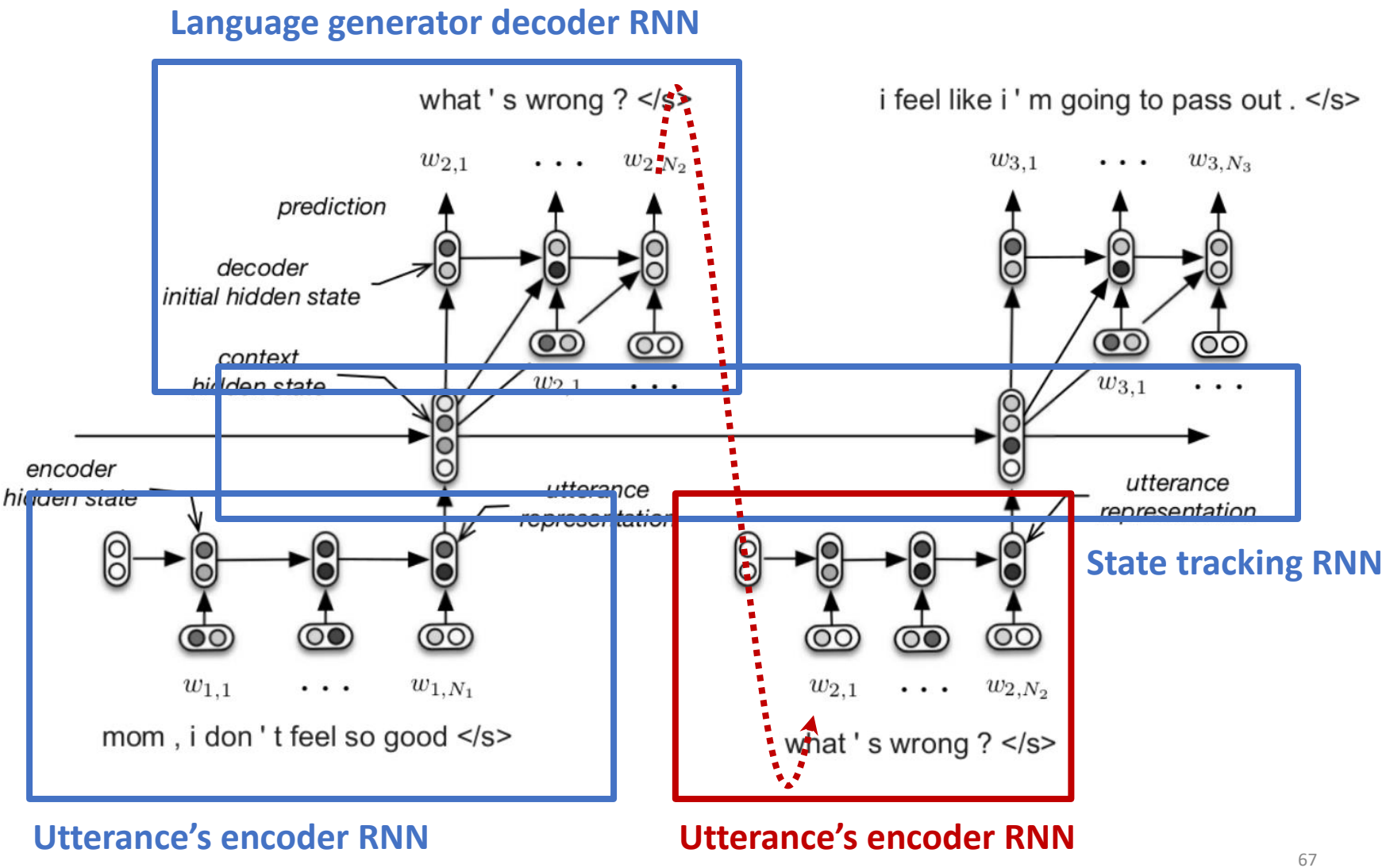
Hierarchical Recurrent Encoder-Decoder



Hierarchical Recurrent Encoder-Decoder



Hierarchical Recurrent Encoder-Decoder



Initialization

- Word embeddings are initialized from word2vec
- Utterance encoders and decoders are initialized with QA from movie subtitles

Sample neural chatbot output

Input	tell me ... how long have you had this falling sickness ?
Vanilla-SEQ2SEQ	i'm not a doctor.
Adversarial	a few months, i guess .
Input	so i had the doctors test sammy 's response to conditioning .
Vanilla-SEQ2SEQ	sammy wrote the test sammy wrote the test .
Adversarial	so he took the pills .
Input	they didn 't have much success with this problem commander .
Vanilla-SEQ2SEQ	they 're not the only ones who have been in the system .
Adversarial	can we find someone else ?

Summary

- Sequence models:
  - RNNs can nicely model sequence data.
  - GRUs and LSTMs overcome some of the memory limitations.
- Architectures:
  - Deep archictures to capture complex interactions
  - Bi-direction architectures to capture long-term dependencies
  - Encoder-decoder
  - Sequence to sequence
- **Tasks**: machine translation, image captioning, summarization.
- Dive into Deep Learning chapters 8 and 9:
  - http://d2l.ai/chapter_recurrent-neural-networks/index.html
  - http://d2l.ai/chapter_recurrent-modern/index.html