# Recurrent Neural Networks (RNNs).

Alvaro Soto

Computer Science Department, PUC

RNNs: neural networks models for processing sequential data (Rumelhart et al., 1986a).

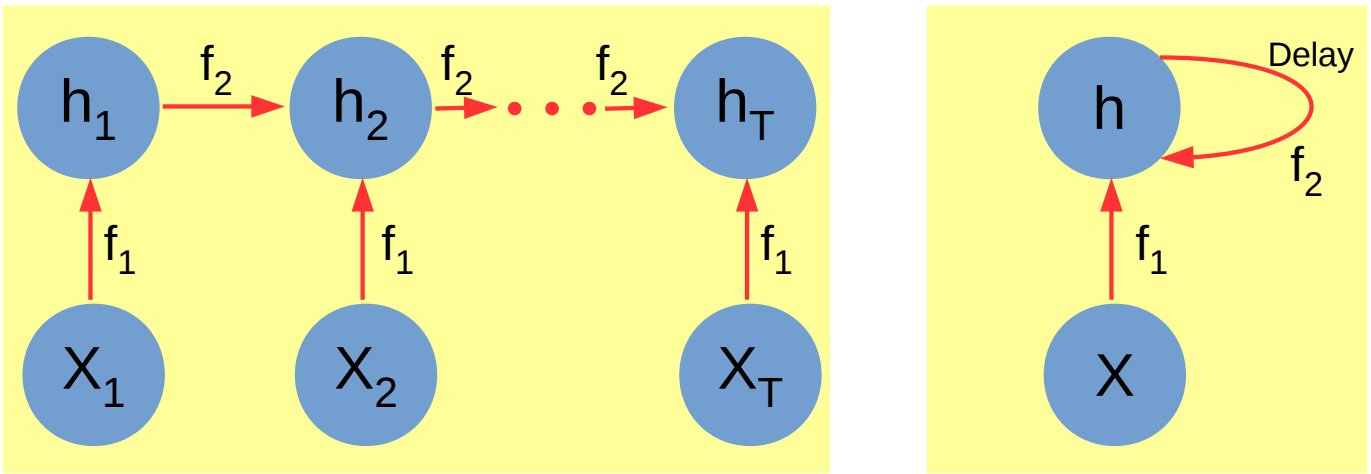
Sequential Data? 

Yeah!, they are highly useful, our big data world is full of sequences like text, audio, and video datasets.

RNNs: neural networks models for processing sequential data (Rumelhart et al., 1986a).

## Common Configuration.

$$h_t = f(h_{t-1}, x_t), \quad \text{t=[1,...,T]}$$

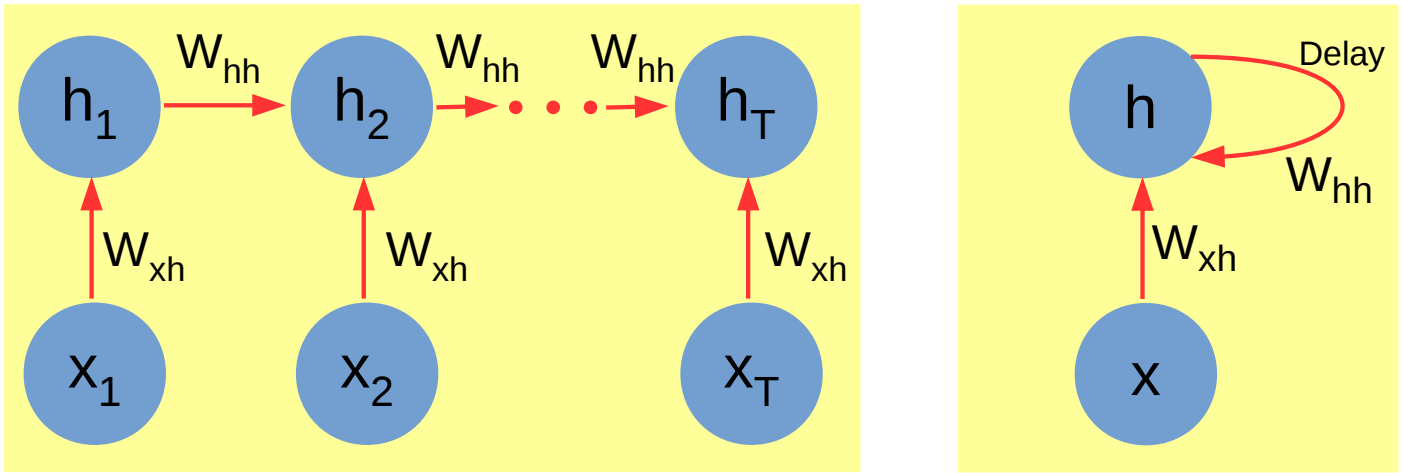



- ***t*** represents a step in the sequence. It usually means temporal steps, but it can also represent spatial order or other dimension.
- Functions $f_1$ and $f_2$ can take different forms, usually they correspond to linear models.

RNNs: neural networks models for processing sequential data (Rumelhart et al., 1986a).

Common Configuration:
Lineal models and sigmoid activations.

$$h_t = \sigma(W_{hh}\, h_{t-1} + W_{xh}\, x_t)$$

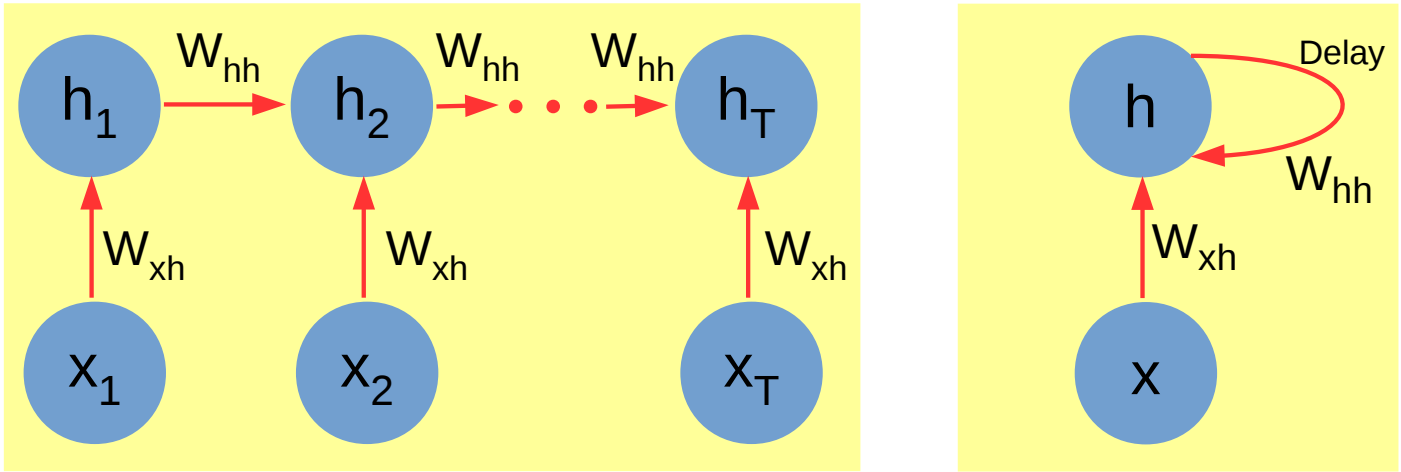



$$x \in \mathbb{R}^{d_x}, h \in \mathbb{R}^{d_h}, W_{xh} \in \mathbb{R}^{d_h \times d_x}, W_{hh} \in \mathbb{R}^{d_h \times d_h}.$$

RNNs: neural networks models for processing sequential data (Rumelhart et al., 1986a).

Common Configuration:
Lineal models and sigmoid activations.

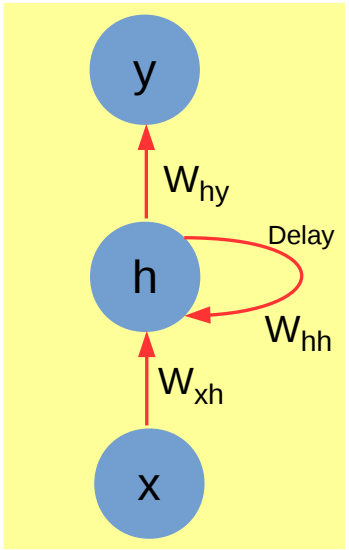
$$h_t = \sigma(W_{hh}\, h_{t-1} + W_{xh}\, x_t)$$




- Interestingly, this generates a deep architecture without a substantial increase in the number of parameters, why?.
- Great!, by using recursive or recurrent convolutional layers, we can obtain longer data dependencies while controlling model capacity.

Most applications needs to incorporate the coding of an output, why?.

$$h_t = \sigma(W_{hh}\, h_{t-1} + W_{xh}\, x_t)$$
$$y_t = W_{hy}\, h_t$$

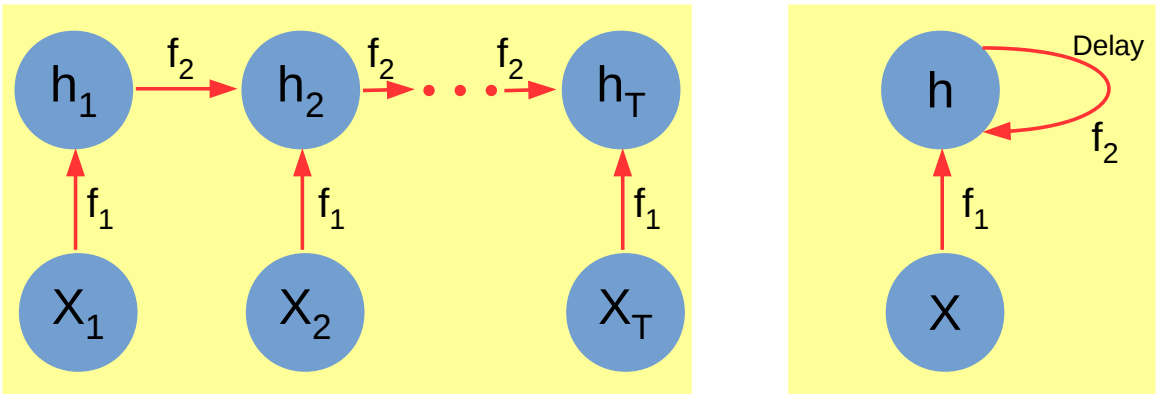



- Notice that this configuration leads to a RNN that maps an input sequence to an output sequence of the same length.
- However, as we will discuss, RNNs are very flexible and they can accomodate to model sequences of different size.

RNNs can have different configurations. Key idea is to provide deep parameter sharing by including cycles.

## Typical Configuration.

$$h_t = f(h_{t-1}, x_t)$$

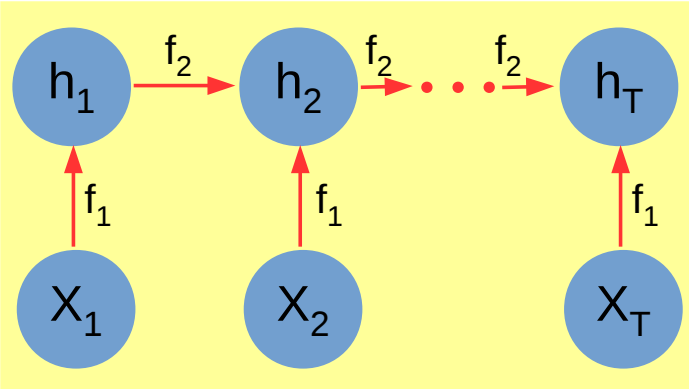



Key step is the cycle or recurrent connection that provides some degree of "memory".

RNNs can have different configurations. Key idea is to provide deep parameter sharing by including cycles.

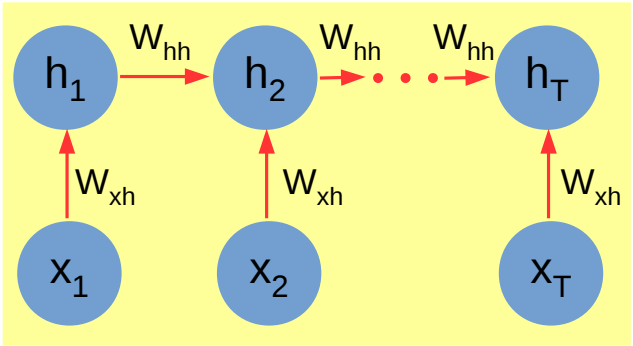
## Typical Configuration.

$$h_t = f(h_{t-1}, x_t)$$




- The network learns to use the hidden state $h_t$ as a lossy summary of the inputs until step $t$.
- This allows us to model sequence order (similar to convolution masks in CNNs) (key property 1).

RNNs can have different configurations. The key idea is to provide deep parameter sharing by including cycles.




Parameter sharing:

- Provides a suitable mechanism that allows us to model sequences with different length (key property 2)
- Actually, an unfolded RNNs can be pictured as a deep feedforward network where all the layers share the same weights (why?).

Deep parameter sharing:

- While CNN parameter sharing is shallow (convolution mask), RNN sharing is deep (Why?).
- Deep parameter sharing allows us to model interactions that are far away (deeper) (key property 3).

RNNs can have different configurations. The key idea is to provide deep parameter sharing by including cycles.

Deep parameter sharing:

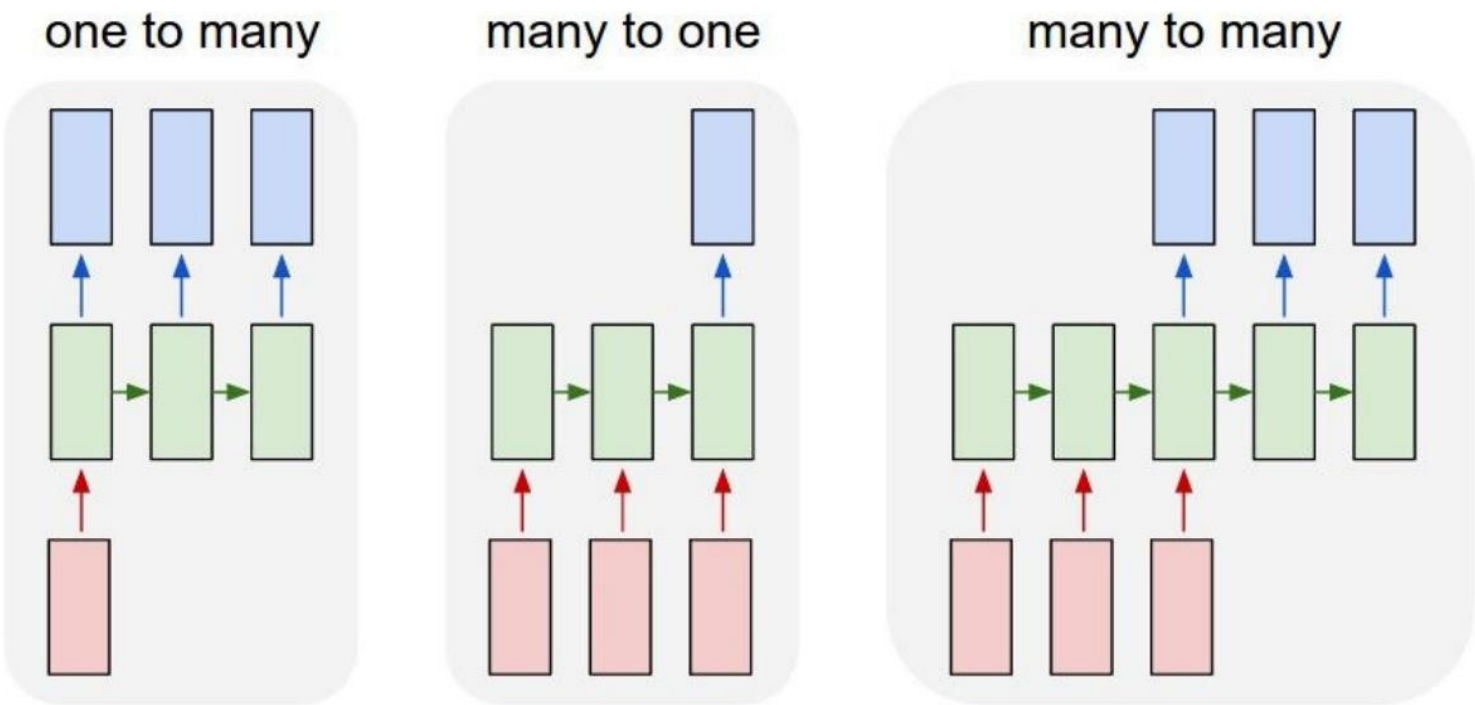
- It is important to understand the relevance of parameter sharing to model data of different dimensionality.
- CNNs can only be applied to problems whose inputs and targets can be encoded with vectors of fixed dimensionality.
- This is a significant limitation, since many important problems are best expressed as sequences whose lengths are not known a-priori. E.g., speech recognition, machine translation, Q&A, etc.
- RNNs learns to map an input sentence of variable length into a fixed-dimensional vector representation.
- Additionally, output vectors can be generated sequentially, so one can obtain an output sequence of variable length.

RNNs can have different configurations. The key idea is to provide deep parameter sharing by including cycles.

Deep parameter sharing:

- It is also important to understand the relevance of modelling far interactions among data.
- CNNs are esentially hierarchical compositional models, while this is great and follows the pattern generation of a large list of domains, CNNs do not incorporate an explicit mechanism to model distant interactions among input data.
- Several applications need these types of models, such as text understanding, Q&A systems, end-to-end dialog agents, among many others.
- In general, the ability to distinguish distance context or attention mechanims is key to create intelligent agents.

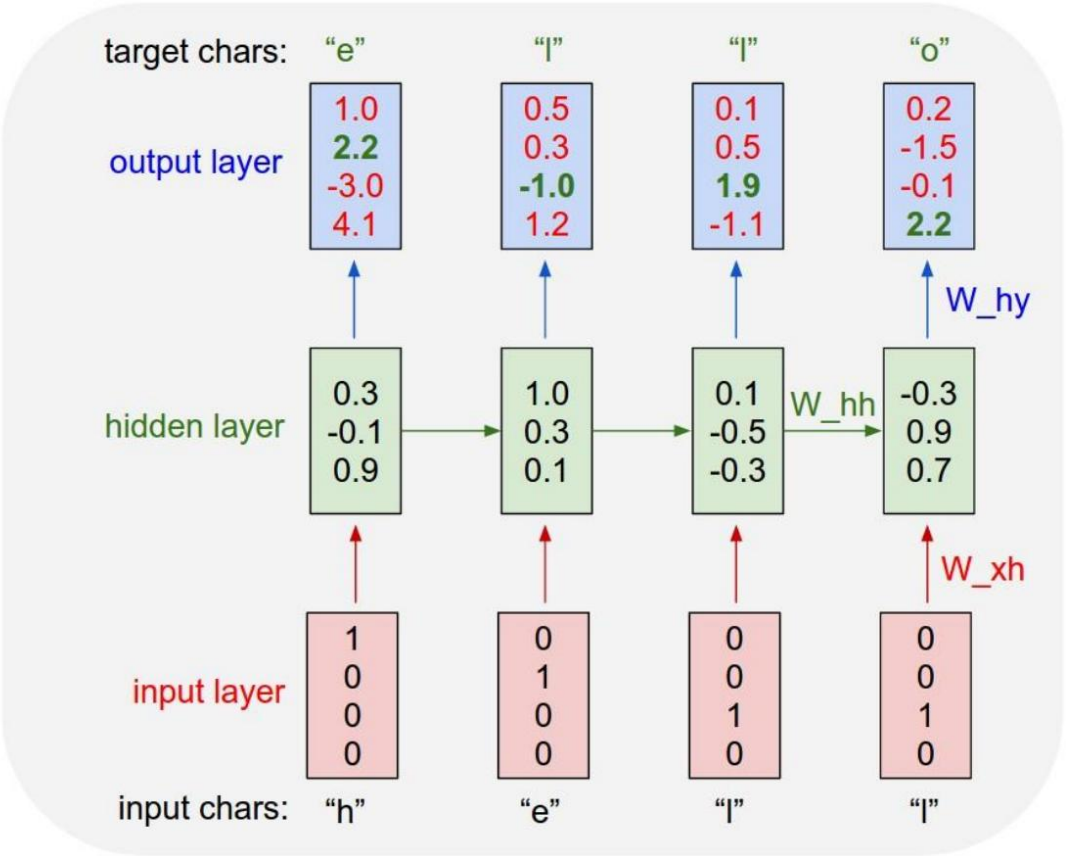
Depending of the application, we can define several possible configurations.



**Example: Character-level Language Model**

Vocabulary:
[h,e,l,o]

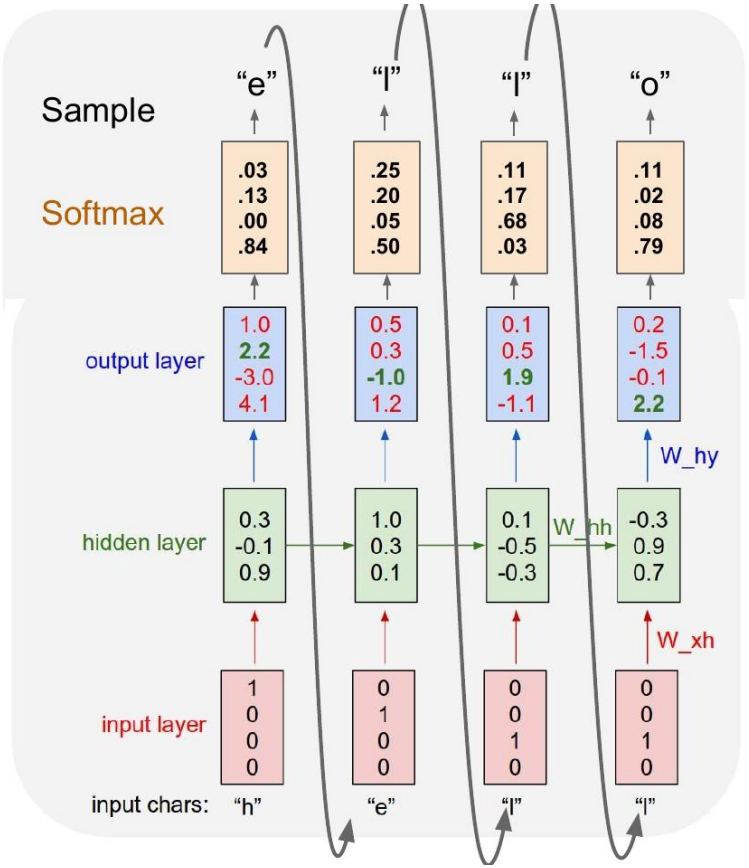
Example training sequence:
**“hello”**



**Example: Character-level Language Model Sampling**

Vocabulary:
[h,e,l,o]

At test-time sample characters one at a time, feed back to model



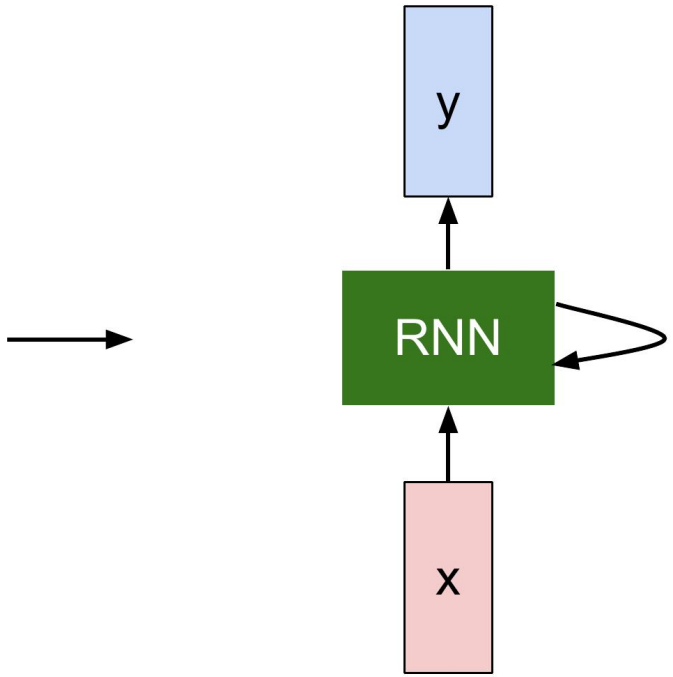
# Ex. Text generation

## THE SONNETS

by William Shakespeare

From fairest creatures we desire increase,
That thereby beauty's rose might never die,
But as the riper should by time decease,
His tender heir might bear his memory:
But thou, contracted to thine own bright eyes,
Feed'st thy light's flame with self-substantial fuel,
Making a famine where abundance lies,
Thyself thy foe, to thy sweet self too cruel:
Thou that art now the world's fresh ornament,
And only herald to the gaudy spring,
Within thine own bud buriest thy content,
And tender churl mak'st waste in niggarding:
Pity the world, or else this glutton be,
To eat the world's due, by the grave and thee.

When forty winters shall besiege thy brow,
And dig deep trenches in thy beauty's field,
Thy youth's proud livery so gazed on now,
Will be a tatter'd weed of small worth held:
Then being asked, where all thy beauty lies,
Where all the treasure of thy lusty days;
To say, within thine own deep sunken eyes,
Were an all-eating shame, and thriftless praise.
How much more praise deserv'd thy beauty's use,
If thou couldst answer 'This fair child of mine
Shall sum my count, and make my old excuse,'
Proving his beauty by succession thine!
This were to be new made when thou art old,
And see thy blood warm when thou feel'st it cold.



# Ex. Text generation

```
tyntd-iafhatawiaoihrdemot  lytdws  e ,tfti, astai f ogoh eoase rrranbyne 'nhthnee e
plia tklrgd t o idoe ns,smtt   h ne etie h,hregtrs nigtike,aoaenns lng
```

train more

```
"Tmont thithey" fomesscerliund
Keushey. Thom here
sheulke, anmerenith ol sivh I lalterthend Bleipile shuwy fil on aseterlome
coaniogennc Phe lism thond hon at. MeiDimorotion in ther thize."
```

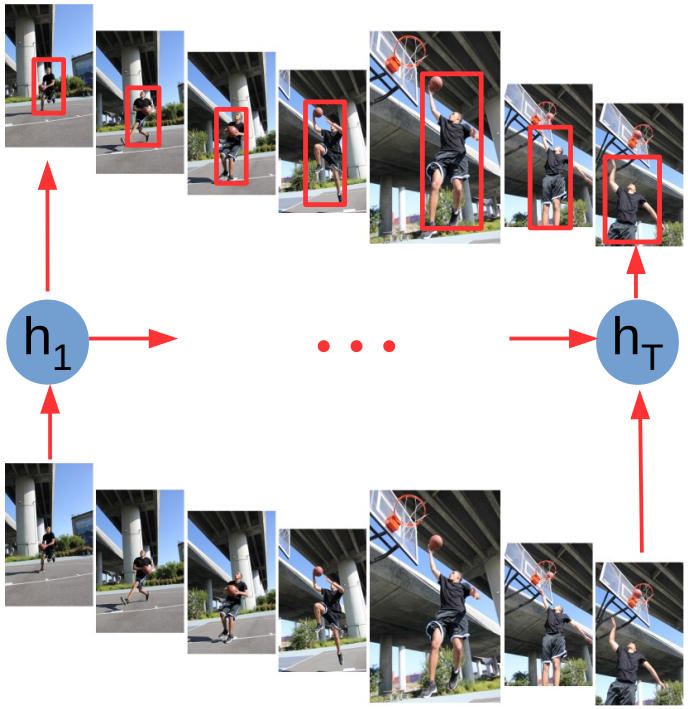
train more

```
Aftair fall unsuch that the hall for Prince Velzonski's that me of
her hearly, and behs to so arwage fiving were to it beloge, pavu say falling misfort
how, and Gogition is so overelical and ofter.
```

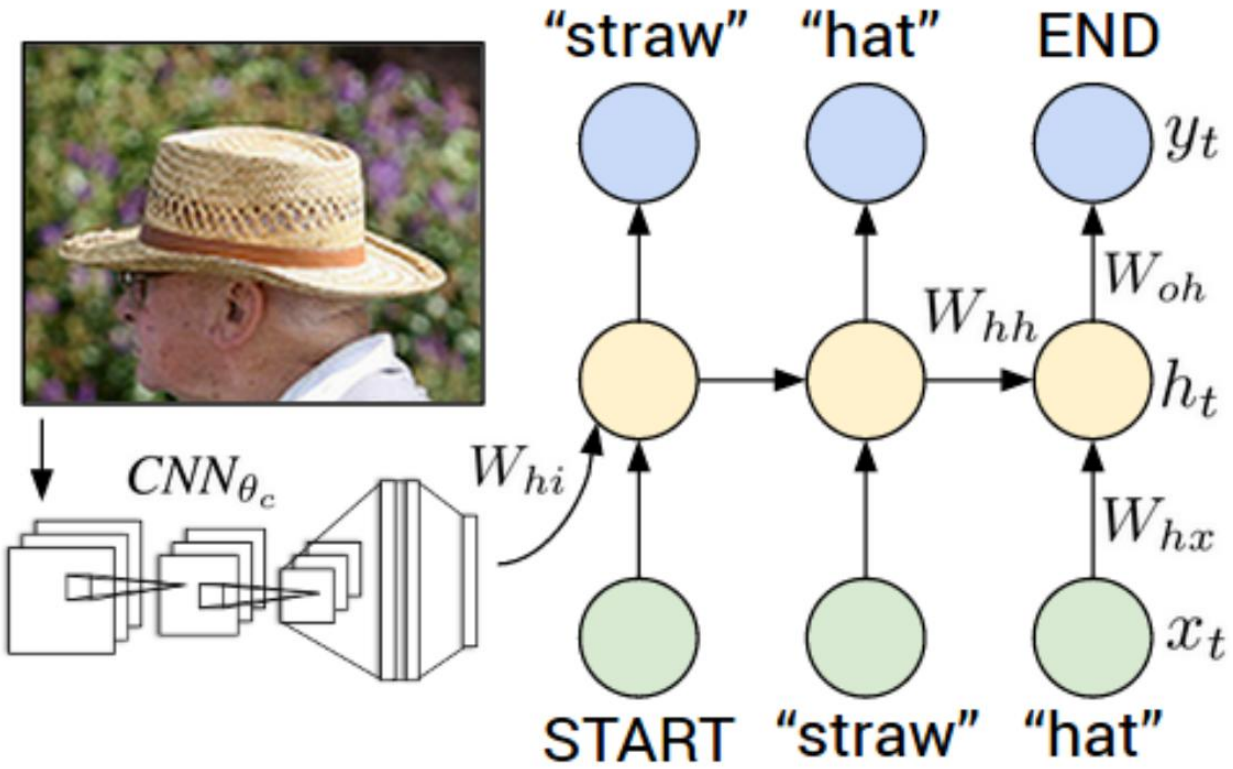
train more

```
"Why do what that day," replied Natasha, and wishing to himself the fact the
princess, Princess Mary was easier, fed in had oftened him.
Pierre aking his soul came to the packs and drove up his father-in-law women.
```

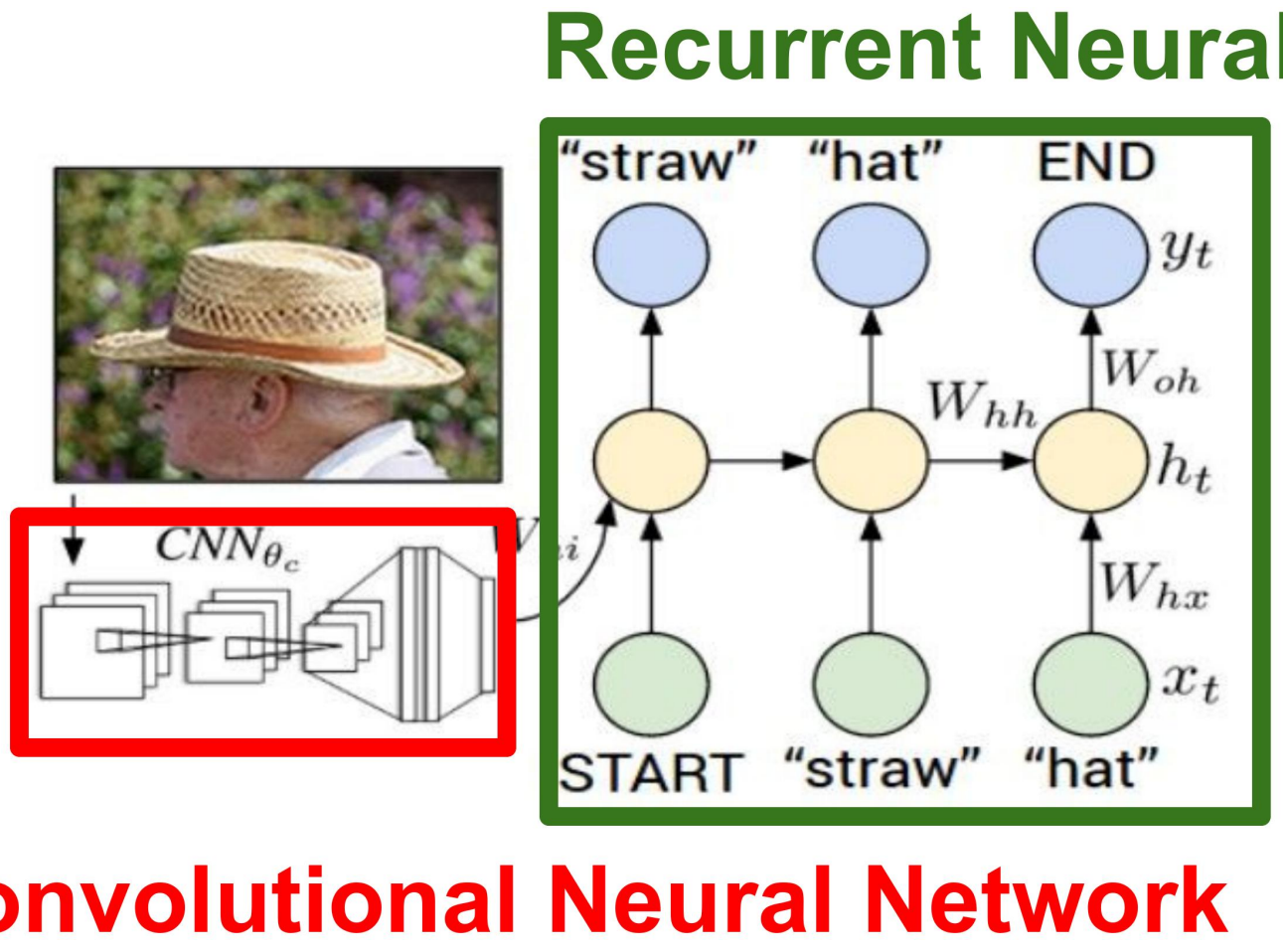
# Ex. Video Classification.



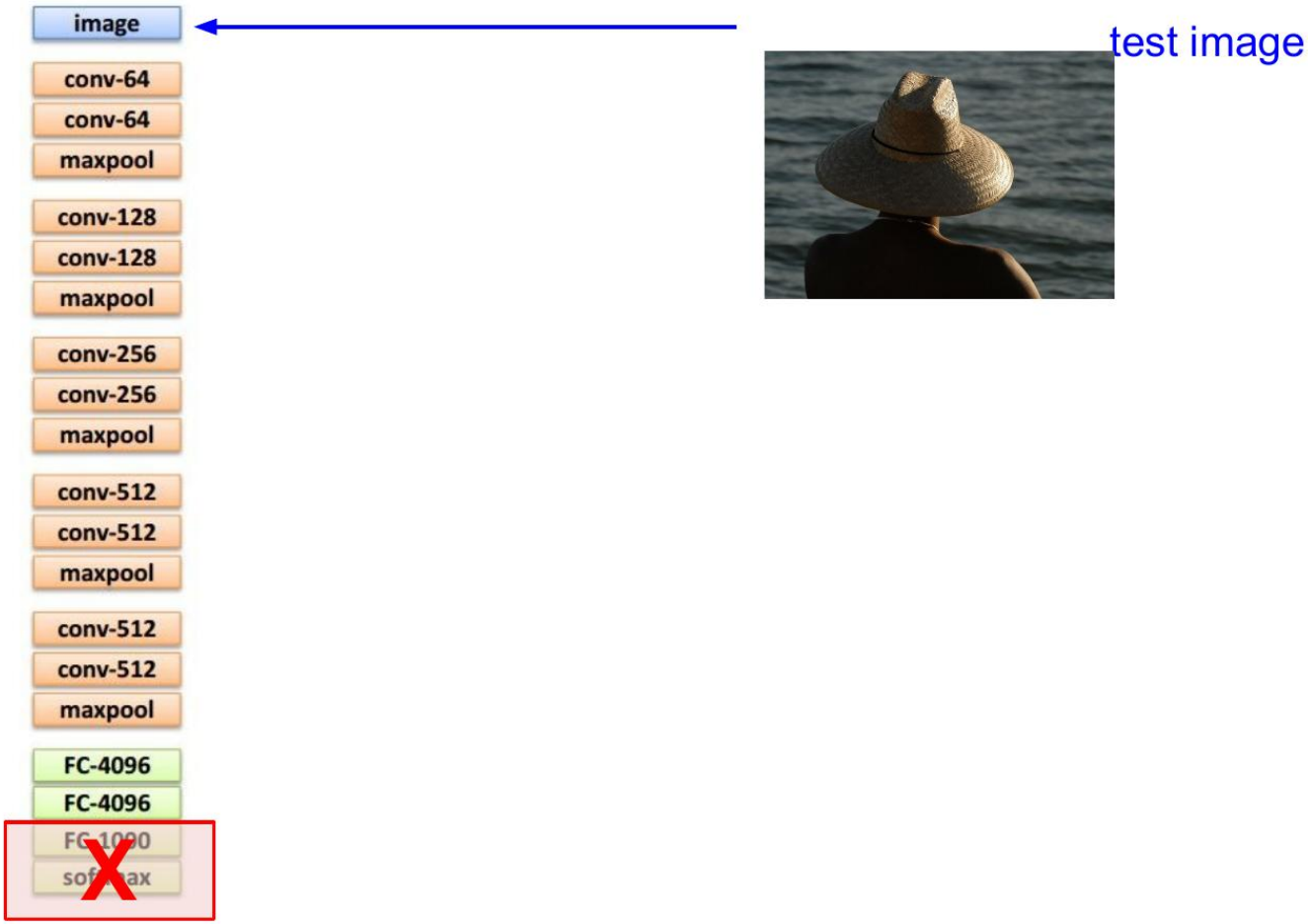
Ex. Image captioning.



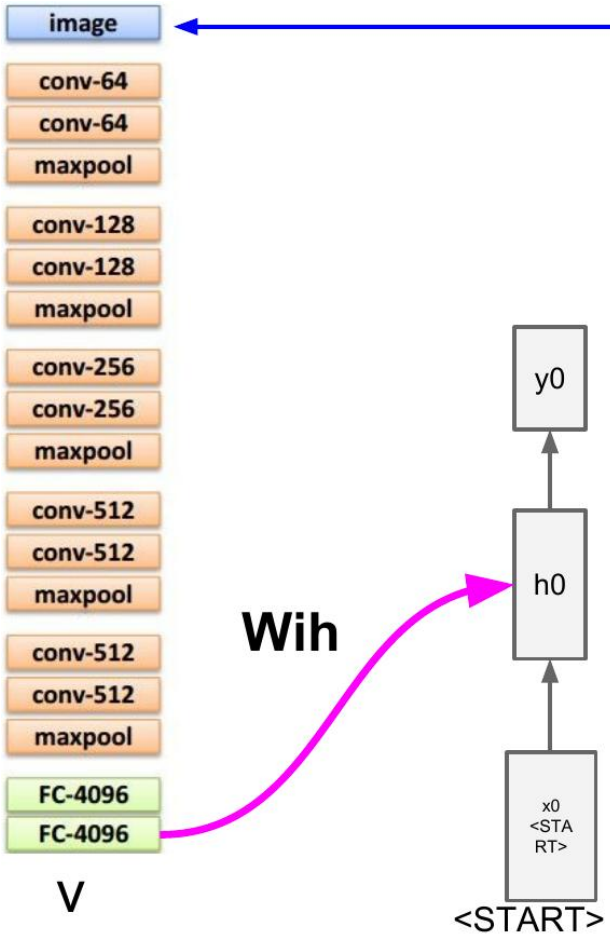
Ex. Image captioning.



# Ex. Image captioning.



# Ex. Image captioning.




test image

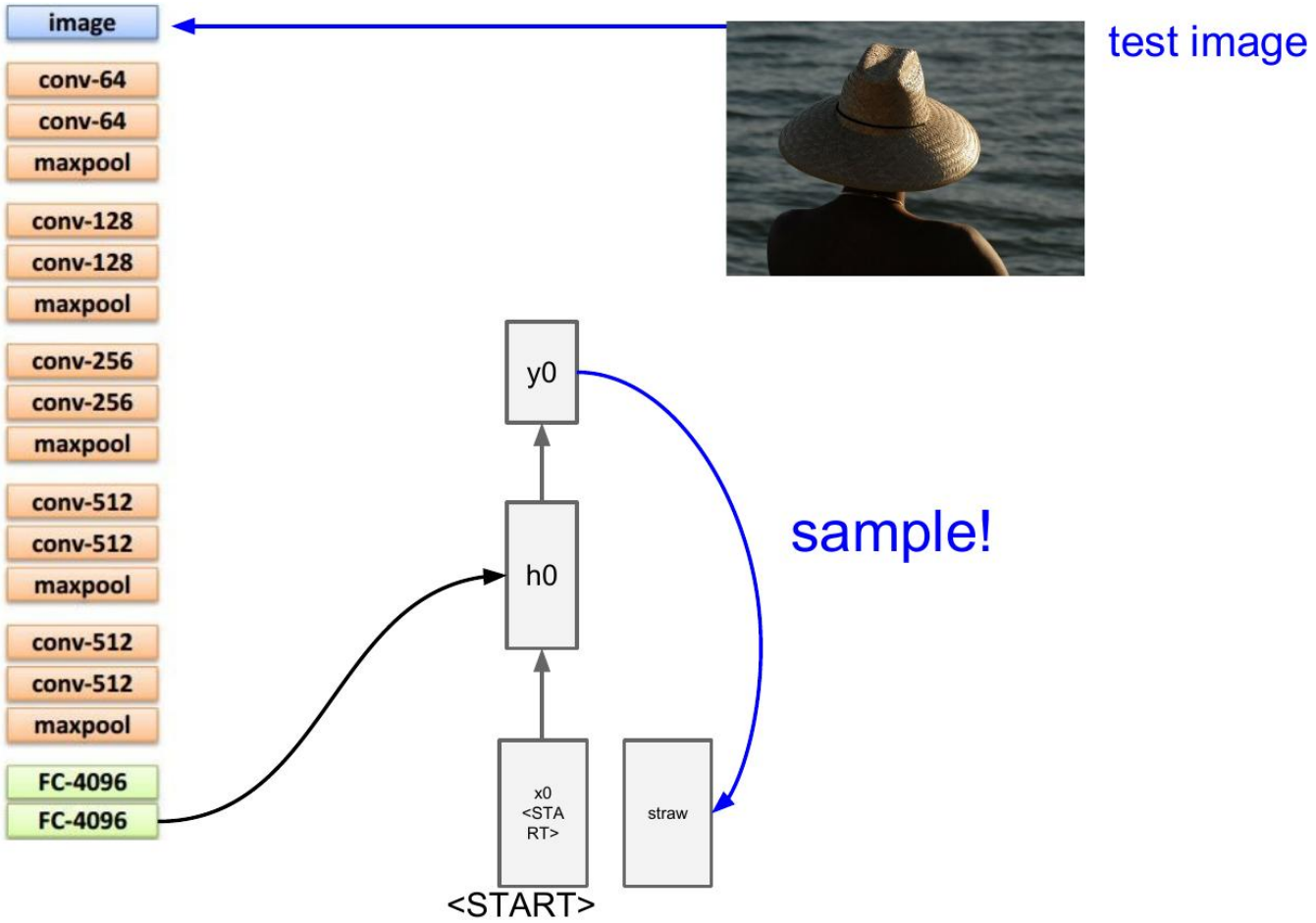
**before:**

$h = \tanh(W_{xh} * x + W_{hh} * h)$

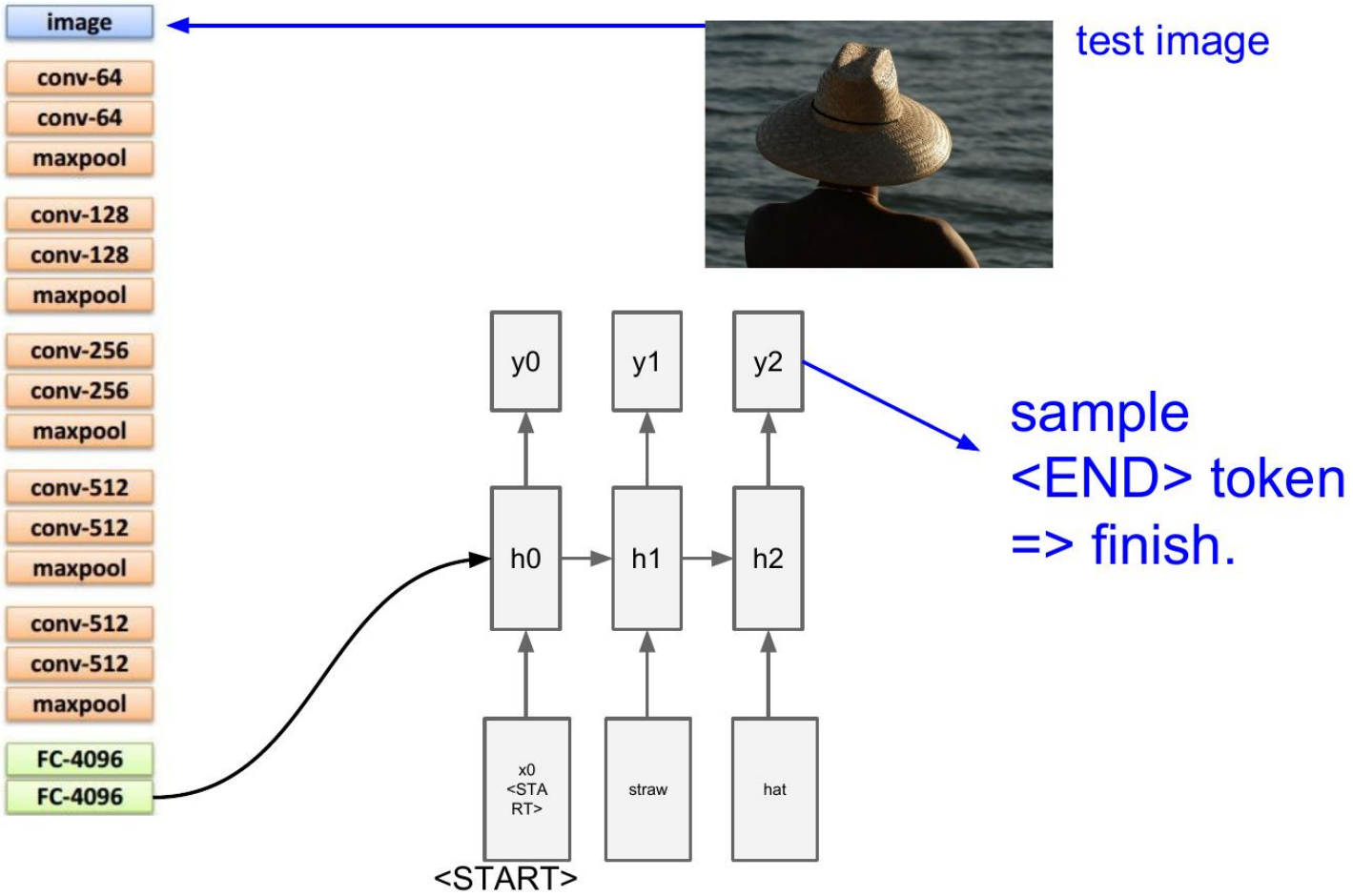
**now:**

$h = \tanh(W_{xh} * x + W_{hh} * h + \mathbf{W_{ih}} * \mathbf{v})$

# Ex. Image captioning.



# Ex. Image captioning.



# Ex. Image captioning: Results

*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*

*Two people walking on the beach with surfboards*

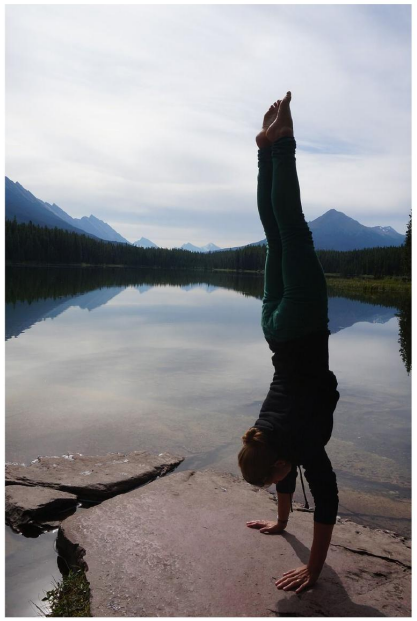

*A tennis player in action on the court*

*Two giraffes standing in a grassy field*

# Ex. Image captioning: Failure Cases

*A woman is holding a cat in her hand*

*A person holding a computer mouse on a desk*



*A woman standing on a beach holding a surfboard*

*A bird is perched on a tree branch*

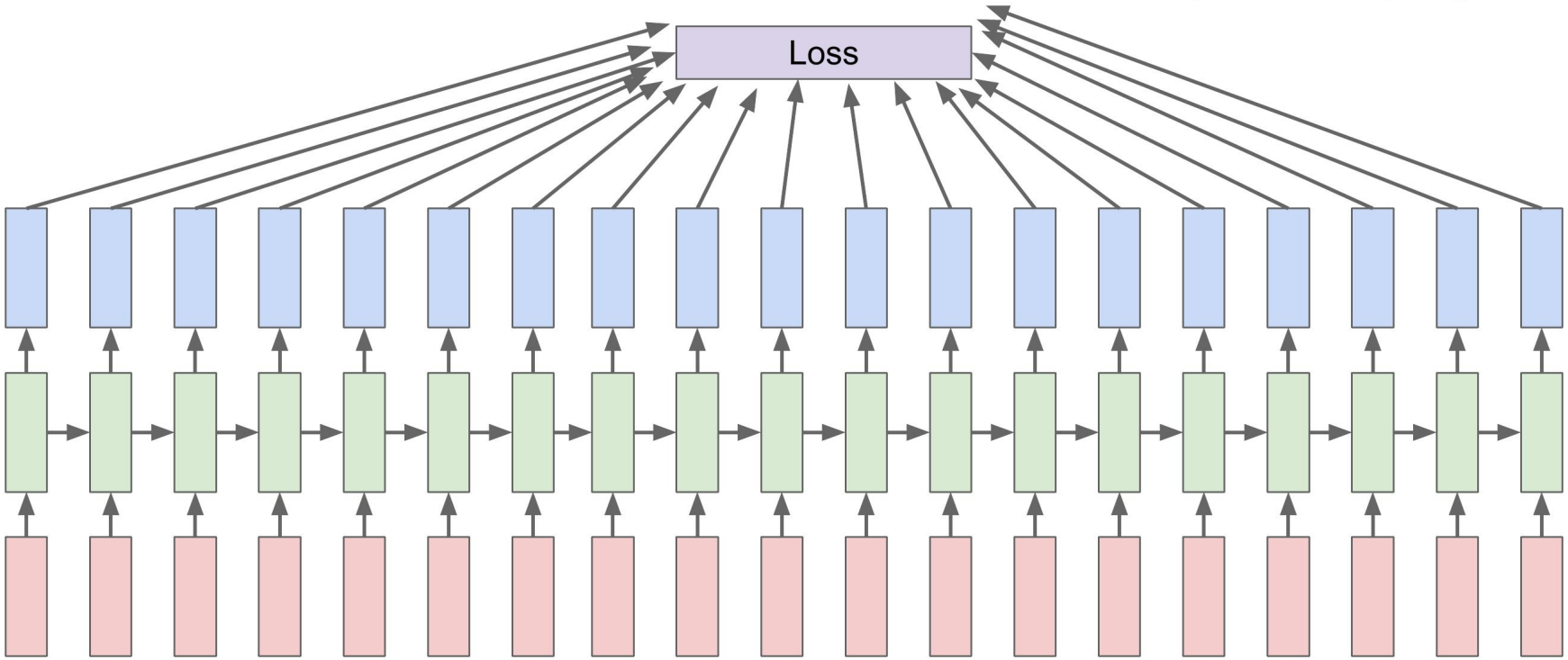
*A man in a baseball uniform throwing a ball*

# RNN Training

- Computation in most RNNs can be decomposed into three blocks of parameters and associated transformations:
  1. From input to hidden state $W_{xh}$,
  2. From previous hidden state to next hidden state $W_{hh}$, and
  3. From hidden state to output $W_{hy}$.
- Training requires to establish a suitable loss function that relates RNN outputs to desired labeled sequences. Ex. cross entropy, ranking loss, etc.
- Minimization of this loss function is usually performed through mini-batch sthocastic gradient steps.
- Computing the gradient through a RNN is straightforward. One simply applies the generalized backpropagation algorithm to the unrolled computational graph (see relevant math in appendix).
- The use of backpropagation on the unrolled graph is called the backpropagation through time (BPTT) algorithm.

# Backpropagation through time

Forward through entire sequence to compute loss, then backward through entire sequence to compute gradient



# The Challenge of Long-Term Dependencies

## Backpropagation through time

- Vanishing gradient problem
- Exploding gradient problem

## Vanilla RNN

$$h_t = \tanh\left(W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$

## LSTM

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

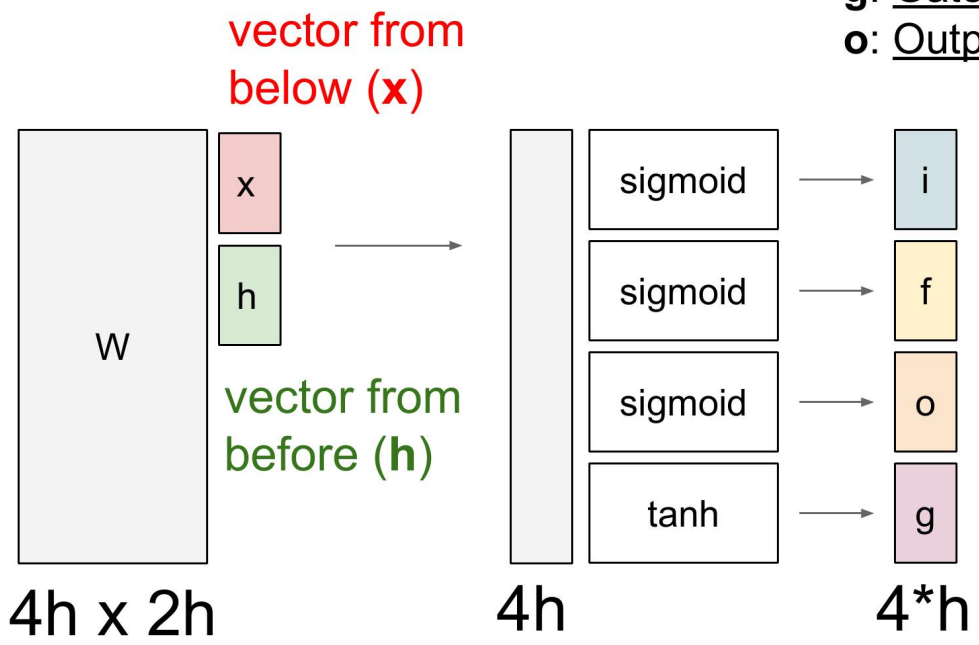
$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$

# Long Short Term Memory (LSTM)

*[Hochreiter et al., 1997]*

**f**: Forget gate, Whether to erase cell
**i**: Input gate, whether to write to cell
**g**: Gate gate (?), How much to write to cell
**o**: Output gate, How much to reveal cell




$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$

- Backpropagation from $c$ to $c_{t-1}$ only elementwise multiplication by f, no matrix multiplication by W.
- In general, vanishing and exploding gradient problems depends on the largest singular value of W. (>1 exploding gradients; <1 vanishing gradients).

# Encoder-Decoder Approach (Cho et al., 2014; Sutskever et al., 2014)

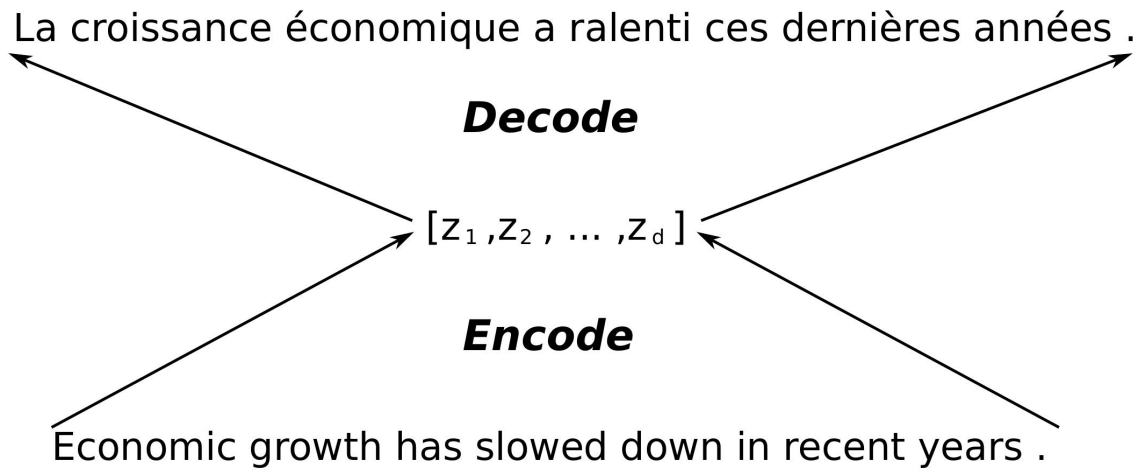
- Several ML applications can be cast as sequence-to-sequence transformation. Ex. speech recognition, machine translation, or question answering, among many others.
- The encoder-decoder approach is a fruitful model to handle these cases.

Core Idea:

1. An encoder or input RNN processes the input sequence, and emits an intermediate fixed-length vector representations, usually as a simple function of its final hidden state.
2. A decoder or output RNN is conditioned on the intermediate vector to generate a suitable output sequence.

Core Idea:

1. An encoder or input RNN processes the input sequence, and emits an intermediate fixed-length vector representations, usually as a simple function of its final hidden state.
2. A decoder or output RNN is conditioned on the intermediate vector to generate a suitable output sequence.




As a main novelty, the intermediate representation $[z_1, \ldots, z_d]$ acts as a latent space, such that the input and output sequences can have different sizes.

We can think about the mid-level representation as a semantic embedding space that encodes relevant semantic of the input instances.

We can think about the mid-level representation as a semantic embedding space that encodes relevant semantic of the input instances.

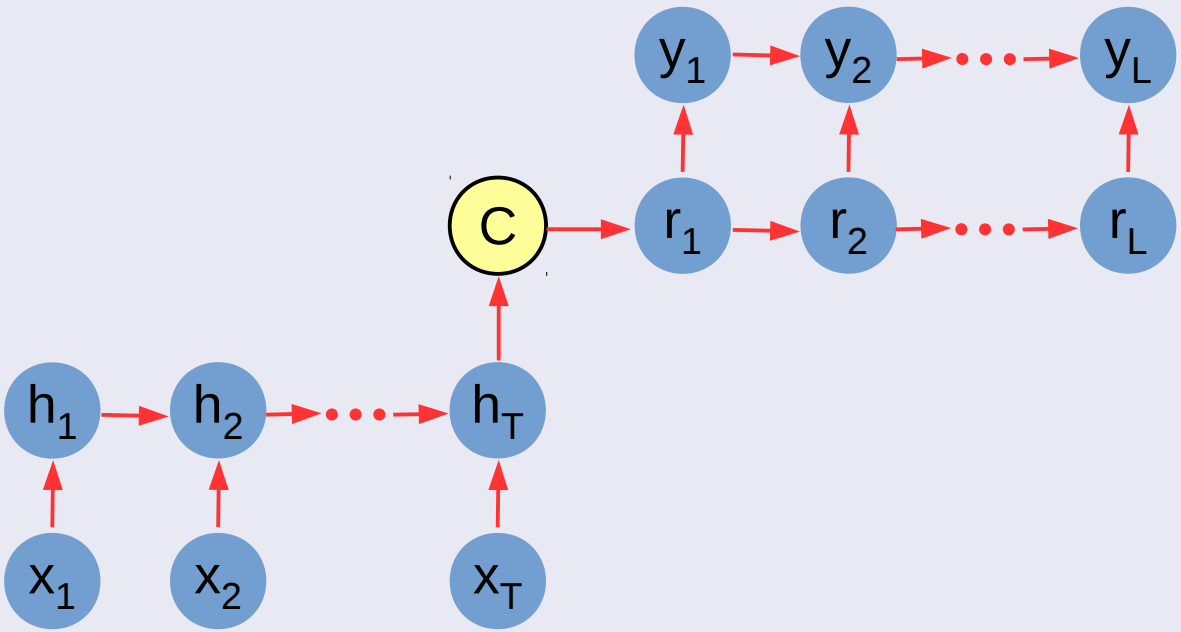
As an example, in the case of text inputs:

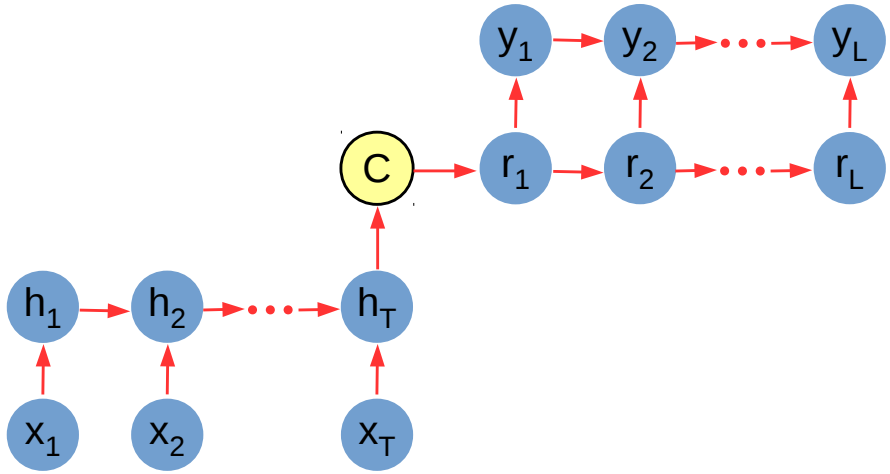
- The intermediate embedded space maps sentences with similar meaning close to each other, and far from unrelated sentences.
- Furthermore, due to the sequential construction, the encoding is sensitive to word order, a capability not possible in BoW models.

- Encoder and decoder RNNs are jointly trained to maximize the average conditional log probability over all pairs of training sequences $(x_i, y_i) \in TS$, i.e., we maximize:

$$\frac{1}{|TS|} \sum_{(x_i, y_i) \in TS} log\ P(y_i|x_i)$$

- One of the simplest configuration considers an encoder that outputs a context vector C that is then transformed to a sequence by a decoder network:






In this case, the model is given by:

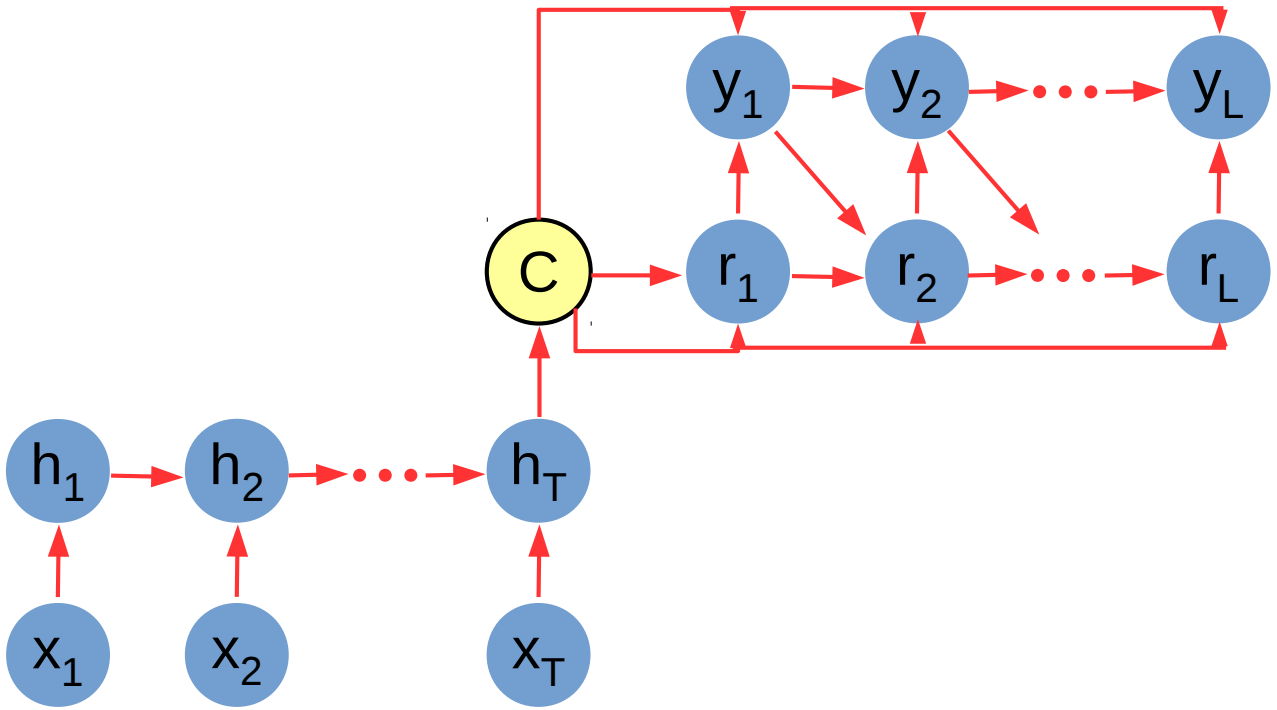
1. Encoder RNN

$$h_t = W_{xh} x_t + W_{hh} h_{t-1}$$

2. Decoder RNN

$$\begin{aligned}
&if(t > 1) \\
&\quad y_t = W_{ry} r_t + W_{yy} y_{t-1} \\
&\quad r_t = W_{rr} r_{t-1} \\
&if(t = 1) \\
&\quad y_1 = W_{ry} r_1 \\
&\quad r_1 = W_{rr} C \\
&\quad C = W_{hC} h_T
\end{aligned}$$

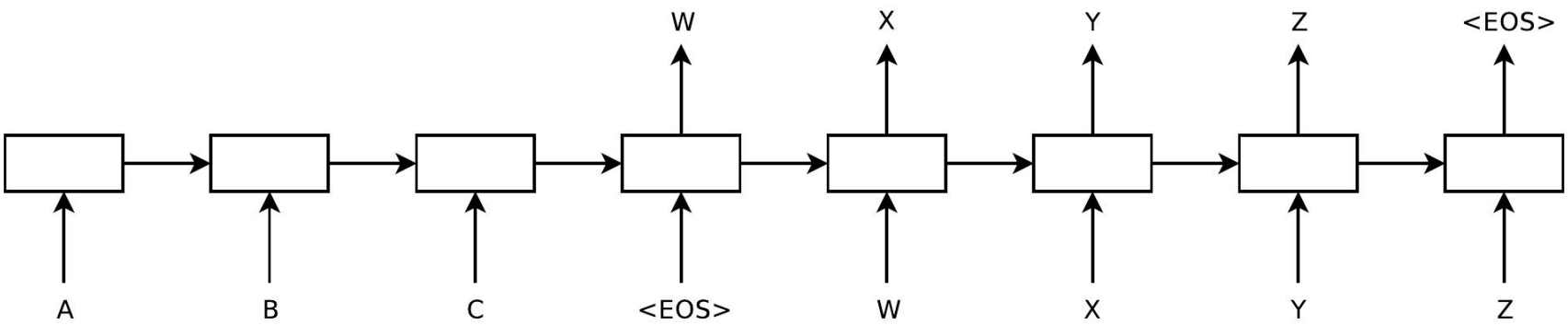
More complex configurations can also be used.



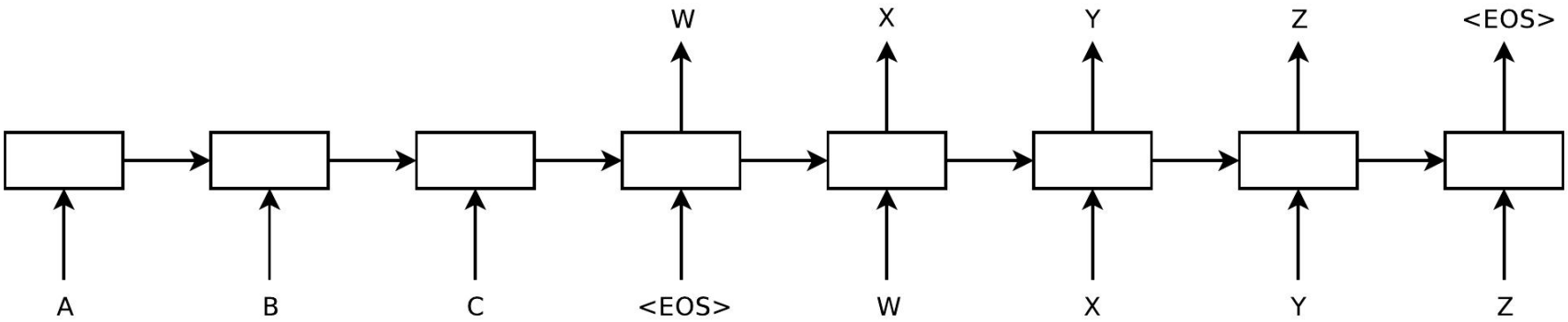
- RNNs allow a lot of flexibility in architecture design
- Vanilla RNNs are simple but donât work very well
- Common to use LSTM or GRU: their additive interactions improve gradient flow
- Backward flow of gradients in RNN can explode or vanish. Exploding is controlled with gradient clipping. Vanishing is controlled with additive interactions (LSTM)
- Better/simpler architectures are a hot topic of current research
- Better understanding (both theoretical and empirical) is needed.

# Ex. from Sutskever et al.,2014

- Textual translation application.
- Model reads an input sentence (ex. "ABC") and produces an output sentence (ex."WXYZ").
- They model RNNs using deep LSTM models (4 layers deep).
- C is last hidden state of the encoder LSTM.
- A special end-of-sentence symbol "$< EOS >$" is used.
- During training, ground truth outputs are used as inputs to the decoder RNN.



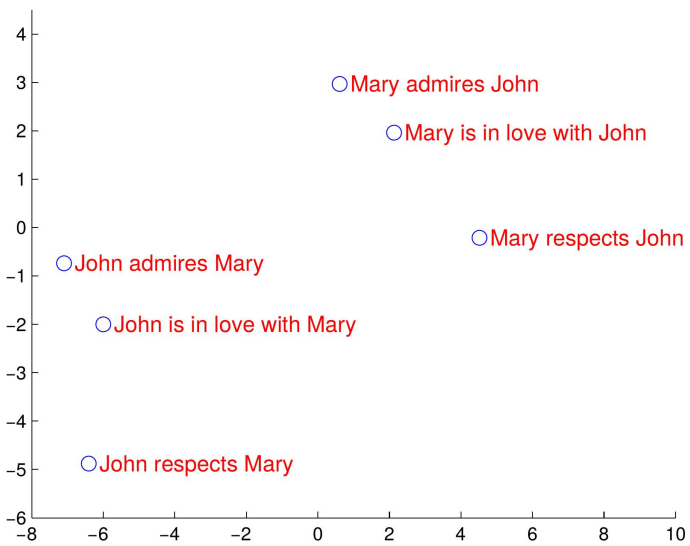
# Ex. from Sutskever et al.,2014

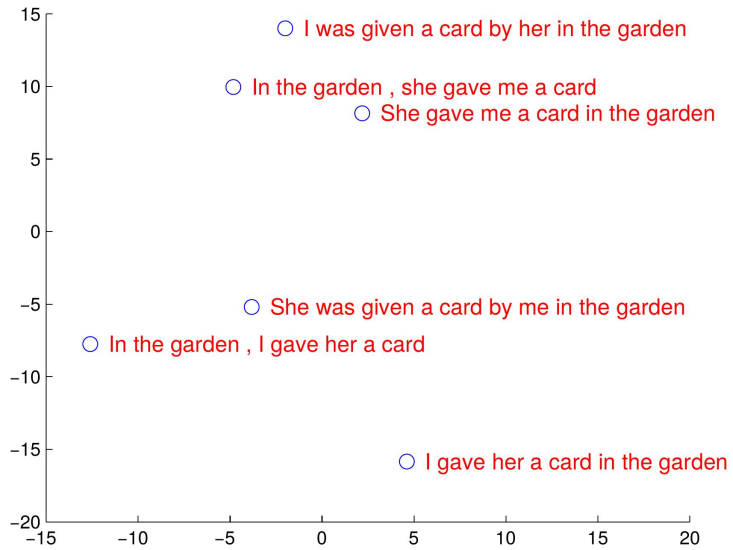



- Interestingly, they obtain better results by reading the input sentence in reverse order, i.e., "CBA" instead of "ABC".
- They argue that this facilitates the optimization process by increasing the number of shorter term dependencies in the data.
- In other words, while reversing the order of the source sentence does not change the average distance between corresponding words in source/target sentences, it decreases the distance between the first few words. Thus, backpropagation has an easier time “establishing communication” between initial parts of source and target sentences.

# Ex. from Sutskever et al.,2014

- At the intermediate level, the model encodes each input sentence into a vector of fixed dimensionality.
- Figure shows a 2D PCA projection of some of these intermediate vectors.




- Notice that phrases are clustered by meaning, considering word order. This type of representation would be difficult to capture with a BoW model.

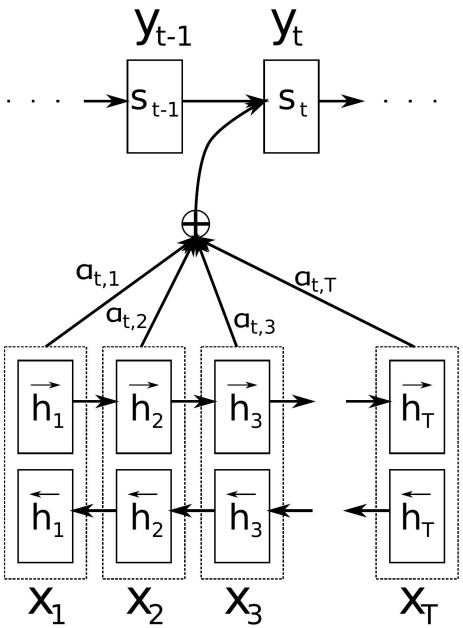
# Attention models

- Under the previous scheme, the input RNN encodes a source sentence into a fixed-length vector. Afterwards, the output RNN uses this vector as its initial hidden state.
- As a relevant problem, the intermediate layer needs to compress all the necessary information from the source sentence into a fixed-length vector.
- An alternative is to adaptively identify the part of the input sentence that is relevant to generate the next output in the sequence.
- This is equivalent to introduce an attention or alignment scheme, where the output is aligned with the corresponding (relevant) input information.
- Consider the following example:

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

# Attention models (Bahdanau et al., ICLR 2015)

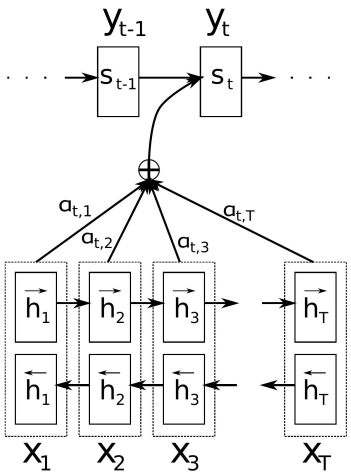



Output $y_t$ is generated according to an adaptive context $C_t$.

$$y_t = W_{yy} y_{t-1} + W_{sy} s_t + W_{cy} C_t$$
$$s_t = W_{ss} s_{t-1} + W_{cs} C_t$$
$$C_t = \sum_{i=1}^{T} \alpha_{ti} h_i \quad (1)$$

- Factors $\alpha_{tj}$ are adaptively estimated. They weight (filter) the relevance of the corresponding hidden state $h_j$ to generate the current output $y_t$.
- The weighting in Eq. 1 implies an expected attention over all the possible alignments (soft attention mechanism). Using this, the model is able to focus on different parts of the source sentence when generating each translated words.
- The use of a bidirectional RNN allows the context variable $C_t$ to potentially depend on the whole input sequence.
- Checking understanding: Is an attention mechanism more important for short or long input sequences?

# Attention models (Bahdanau et al., ICLR 2015)



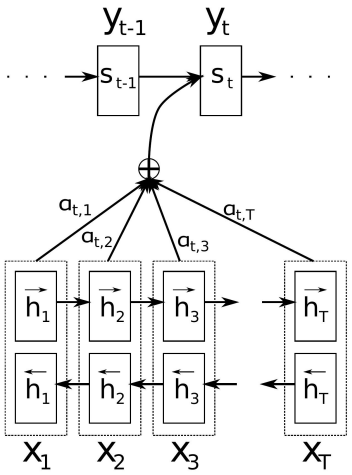

- Key part of the model is the estimation of the attention coefficients $\alpha_{tj}$.
- In Bahdanau et al., $\alpha_{tj}$ estimates how previous output context $s_{t-1}$ and current input context $h_t$ affects the context $s_t$ that is used to generate output $y_t$.
- In general, coeffs $\alpha_{tj}$ can be estimated using other sources, even external knowledge (we will talk about this later).

Bahdanau et al. estimates $\alpha_{tj}$ using a softmax function:

$$\alpha_{ti} = \frac{\beta_{ti}}{\sum_j \beta_{tj}}, \quad \text{where} : \beta_{tj} = V_c^T \sigma(W_c s_{t-1} + U_c h_j), V_c \in \mathbb{R}^n, W_c \in \mathbb{R}^{nxn}, U_c \in \mathbb{R}^{nx2n}$$

- Notice that $U_c \in \mathbb{R}^{nx2n}$ because the bidirectional RNN concatenates the forward and backward hidden vectors.
- In summary, the decoder uses the attention coefficients to find which hidden states from the encoding are most useful for outputting the next translated word.

# Attention models (Bahdanau et al., ICLR 2015)




- Key part of the model is the estimation of the attention coefficients $\alpha_{tj}$.
- In Bahdanau et al., $\alpha_{tj}$ estimates how previous output context $s_{t-1}$ and current input context $h_t$ affects the context $s_t$ that is used to generate output $y_t$.
- In general, coeffs $\alpha_{tj}$ can be estimated using other sources, even external knowledge (we will talk about this later).

Bahdanau et al. estimates $\alpha_{tj}$ using a softmax function:

$$\alpha_{tj} = \frac{\beta_{tj}}{\sum_k \beta_{tk}}, \quad \text{where} : \beta_{tj} = V_c^T \sigma(W_c s_{t-1} + U_c h_j), V_c \in \mathbb{R}^n, W_c \in \mathbb{R}^{nxn}, U_c \in \mathbb{R}^{nx2n}$$

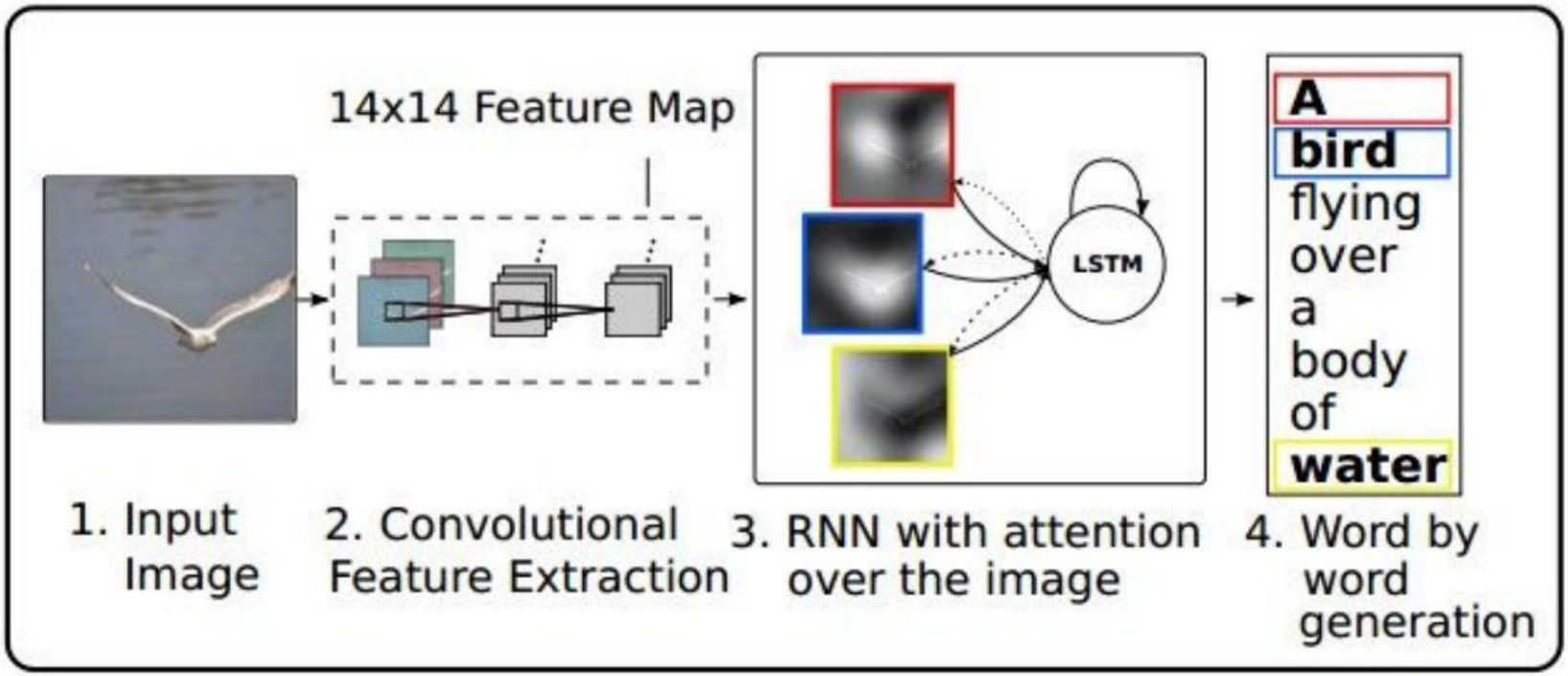
- It is important to note than the attention mechanism proposed by this model acts over the context vectors $h_j$ provided by the encoder. Also, they are estimated by an additive model that combines embeddings of previous output context and current input context.
- As we will discuss, alternative attention models based on other operations (tensors, bilinear) can act directly over the input features $x_t$'s or (and) over the output context vectors $s_t$'s.

# Ex. Attention models: Image captioning

- In general, one can frame a modality transformation as a language translation problem. In the case of image captioning, a transformation from a visual to a textual representation (caption). Therefore, we can use the previous model (Bahdanau et al., ICLR 2015).
- In contrast to textual inputs, in the visual world we have two complications: i) List of valid words (dictionary) is unknown, and ii) Spatial location of relevant structures (word order) is not predefined.

- i) In terms of a relevant dictionary, most works use an image descriptor provided by one of the last layers of a CNN (ex. 4096D from Alex or VGG nets).
- ii) In terms of spatial location, several works use different ideas: a detector of potential object areas (objectness detector) (Shih et al., CVPR-16), regular grid of patches from the output of a CNN (Xu, ICML-15), and a bank of attribute detectors (visual classifier of words) (You et al., CVPR 2016).

# Ex. Attention models: Image captioning (Xu et al., ICML 2015)



# Ex. Attention models: Image captioning (Xu et al., ICML 2015)

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.


A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.