CS7015 (Deep Learning) : Lecture 16

Encoder Decoder Models, Attention Mechanism

Mitesh M. Khapra

Department of Computer Science and Engineering
Indian Institute of Technology Madras

Module 16.1: Introduction to Encoder Decoder Models

- We will start by revisiting the problem of language modeling

- We will start by revisiting the problem of language modeling
- Informally, given '$t-i$' words we are interested in predicting the t^{th} word

- We will start by revisiting the problem of language modeling
- Informally, given '$t-i$' words we are interested in predicting the t^{th} word
- More formally, given $y_1, y_2, ..., y_{t-1}$ we want to find

$$y^* = argmax\ P(y_t|y_1, y_2, ..., y_{t-1})$$

- We will start by revisiting the problem of language modeling
- Informally, given '$t-i$' words we are interested in predicting the t^{th} word
- More formally, given $y_1, y_2, ..., y_{t-1}$ we want to find

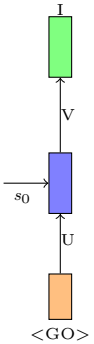
$$y^* = argmax\ P(y_t|y_1, y_2, ..., y_{t-1})$$

- Let us see how we model $P(y_t|y_1, y_2...y_{t-1})$ using a RNN

- We will start by revisiting the problem of language modeling
- Informally, given '$t-i$' words we are interested in predicting the t^{th} word
- More formally, given $y_1, y_2, ..., y_{t-1}$ we want to find

$$y^* = argmax\ P(y_t|y_1, y_2, ..., y_{t-1})$$

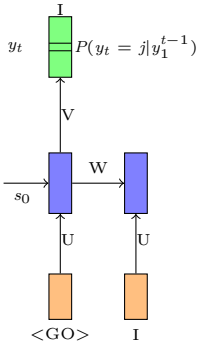
- Let us see how we model $P(y_t|y_1, y_2...y_{t-1})$ using a RNN
- We will refer to $P(y_t|y_1, y_2...y_{t-1})$ by shorthand notation: $P(y_t|y_1^{t-1})$



- We will start by revisiting the problem of language modeling
- Informally, given '$t-i$' words we are interested in predicting the t^{th} word
- More formally, given $y_1, y_2, ..., y_{t-1}$ we want to find

$$y^* = argmax\ P(y_t|y_1, y_2, ..., y_{t-1})$$

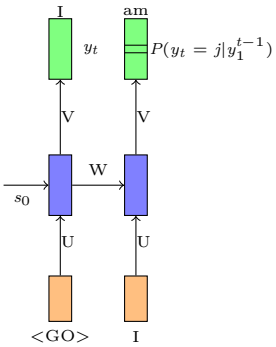
- Let us see how we model $P(y_t|y_1, y_2...y_{t-1})$ using a RNN
- We will refer to $P(y_t|y_1, y_2...y_{t-1})$ by shorthand notation: $P(y_t|y_1^{t-1})$



- We will start by revisiting the problem of language modeling
- Informally, given '$t-i$' words we are interested in predicting the t^{th} word
- More formally, given $y_1, y_2, ..., y_{t-1}$ we want to find

$$y^* = argmax\ P(y_t|y_1, y_2, ..., y_{t-1})$$

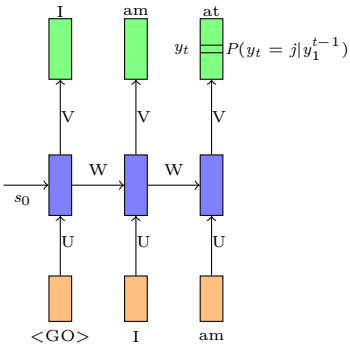
- Let us see how we model $P(y_t|y_1, y_2...y_{t-1})$ using a RNN
- We will refer to $P(y_t|y_1, y_2...y_{t-1})$ by shorthand notation: $P(y_t|y_1^{t-1})$



- We will start by revisiting the problem of language modeling
- Informally, given '$t-i$' words we are interested in predicting the t^{th} word
- More formally, given $y_1, y_2, ..., y_{t-1}$ we want to find

$$y^* = argmax\ P(y_t|y_1, y_2, ..., y_{t-1})$$

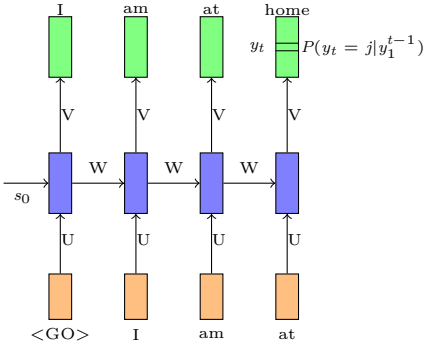
- Let us see how we model $P(y_t|y_1, y_2...y_{t-1})$ using a RNN
- We will refer to $P(y_t|y_1, y_2...y_{t-1})$ by shorthand notation: $P(y_t|y_1^{t-1})$



- We will start by revisiting the problem of language modeling
- Informally, given '$t-i$' words we are interested in predicting the t^{th} word
- More formally, given $y_1, y_2, ..., y_{t-1}$ we want to find

$$y^* = argmax\ P(y_t|y_1, y_2, ..., y_{t-1})$$

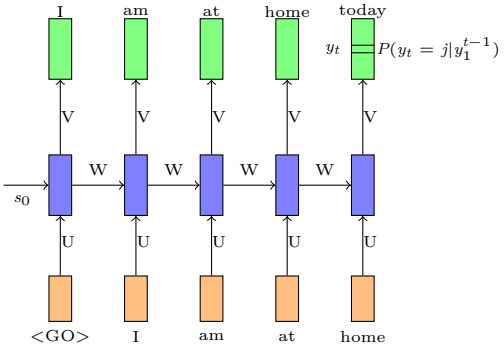
- Let us see how we model $P(y_t|y_1, y_2...y_{t-1})$ using a RNN
- We will refer to $P(y_t|y_1, y_2...y_{t-1})$ by shorthand notation: $P(y_t|y_1^{t-1})$



- We will start by revisiting the problem of language modeling
- Informally, given '$t-i$' words we are interested in predicting the t^{th} word
- More formally, given $y_1, y_2, ..., y_{t-1}$ we want to find

$$y^* = argmax\ P(y_t|y_1, y_2, ..., y_{t-1})$$

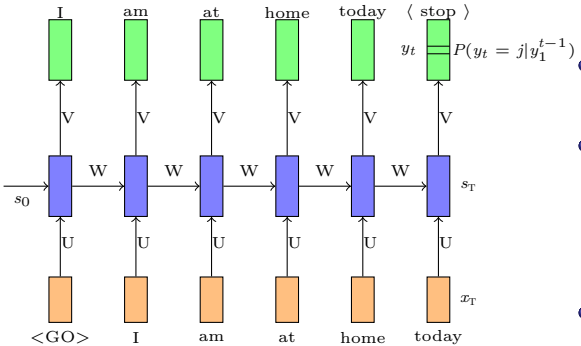
- Let us see how we model $P(y_t|y_1, y_2...y_{t-1})$ using a RNN
- We will refer to $P(y_t|y_1, y_2...y_{t-1})$ by shorthand notation: $P(y_t|y_1^{t-1})$



- We will start by revisiting the problem of language modeling
- Informally, given '$t-i$' words we are interested in predicting the t^{th} word
- More formally, given $y_1, y_2, ..., y_{t-1}$ we want to find

$$y^* = argmax\ P(y_t|y_1, y_2, ..., y_{t-1})$$

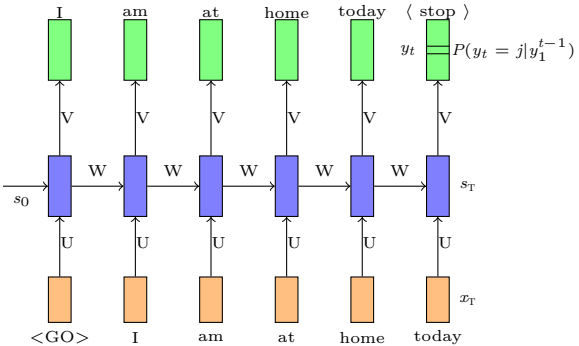
- Let us see how we model $P(y_t|y_1, y_2...y_{t-1})$ using a RNN
- We will refer to $P(y_t|y_1, y_2...y_{t-1})$ by shorthand notation: $P(y_t|y_1^{t-1})$



- We will start by revisiting the problem of language modeling
- Informally, given '$t - i$' words we are interested in predicting the t^{th} word
- More formally, given $y_1, y_2, ..., y_{t-1}$ we want to find

$$y^* = argmax\ P(y_t|y_1, y_2, ..., y_{t-1})$$

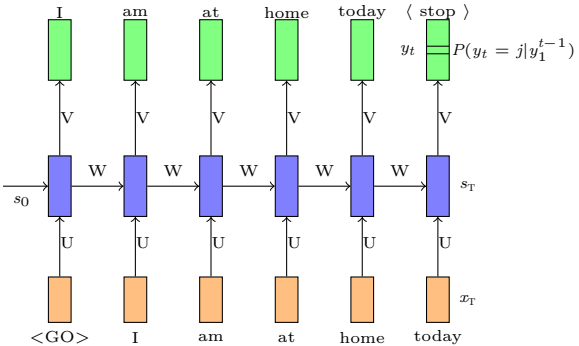
- Let us see how we model $P(y_t|y_1, y_2...y_{t-1})$ using a RNN
- We will refer to $P(y_t|y_1, y_2...y_{t-1})$ by shorthand notation: $P(y_t|y_1^{t-1})$



- We are interested in

$$P(y_t = j | y_1, y_2 \ldots y_{t-1})$$

where $j \in V$ and V is the set of all vocabulary words



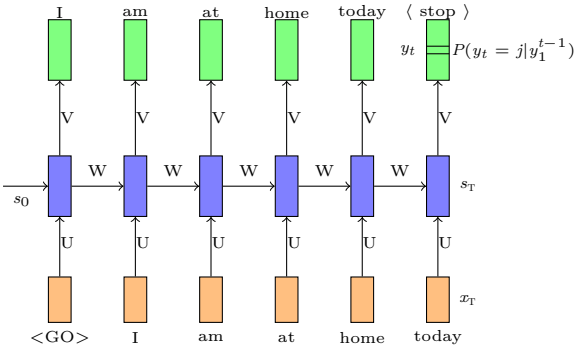
- We are interested in

$$P(y_t = j|y_1, y_2 \ldots y_{t-1})$$

where $j \in V$ and V is the set of all vocabulary words

- Using an RNN we compute this as

$$P(y_t = j|y_1^{t-1}) = softmax(Vs_t + c)_j$$



- We are interested in

$$P(y_t = j|y_1, y_2 \ldots y_{t-1})$$

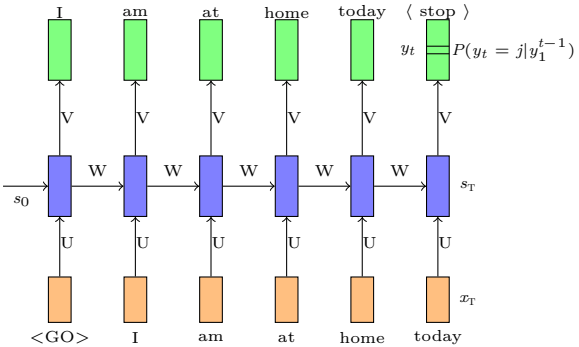
where $j \in V$ and V is the set of all vocabulary words

- Using an RNN we compute this as

$$P(y_t = j|y_1^{t-1}) = softmax(Vs_t + c)_j$$

- In other words we compute

$$\begin{aligned} P(y_t = j|y_1^{t-1}) &= P(y_t = j|s_t) \\ &= softmax(Vs_t + c)_j \end{aligned}$$



- We are interested in

$$P(y_t = j|y_1, y_2 \ldots y_{t-1})$$

where $j \in V$ and V is the set of all vocabulary words

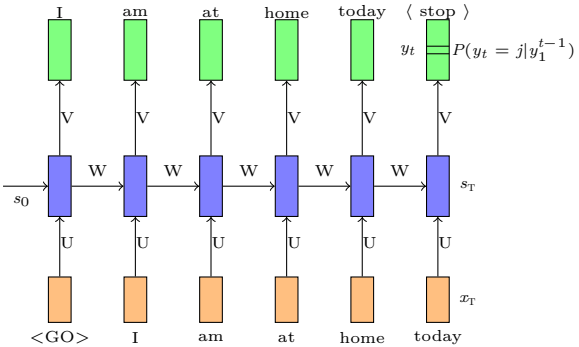
- Using an RNN we compute this as

$$P(y_t = j|y_1^{t-1}) = softmax(Vs_t + c)_j$$

- In other words we compute

$$\begin{aligned} P(y_t = j|y_1^{t-1}) &= P(y_t = j|s_t) \\ &= softmax(Vs_t + c)_j \end{aligned}$$

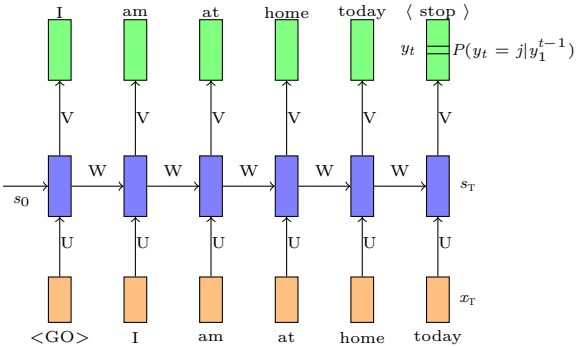
- Notice that the recurrent connections ensure that s_t has information about y_1^{t-1}



- **Data:** All sentences from any large corpus (say wikipedia)

Data:

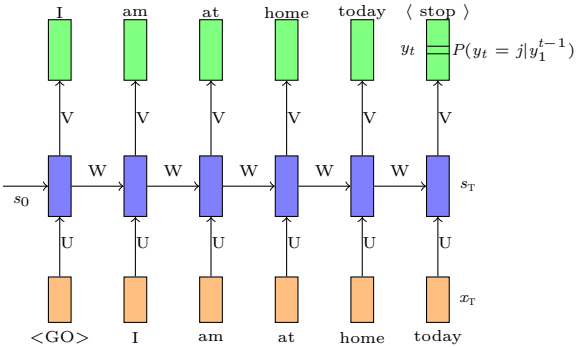
India, officially the Republic of India, is a country in South Asia. It is the seventh-largest country by area, ...



- **Data:** All sentences from any large corpus (say wikipedia)
- **Model:**

Data:

India, officially the Republic of India, is a country in South Asia. It is the seventh-largest country by area, ...



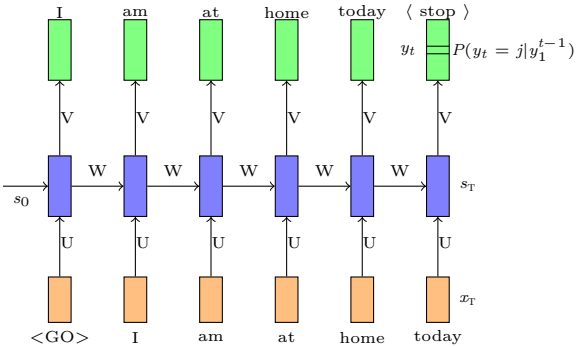
- **Data:** All sentences from any large corpus (say wikipedia)
- **Model:**

$$s_t = \sigma(Ws_{t-1} + Ux_t + b)$$

$$P(y_t = j|y_1^{t-1}) = softmax(Vs_t + c)_j$$

Data:

India, officially the Republic of India, is a country in South Asia. It is the seventh-largest country by area, ...



- **Data:** All sentences from any large corpus (say wikipedia)
- **Model:**

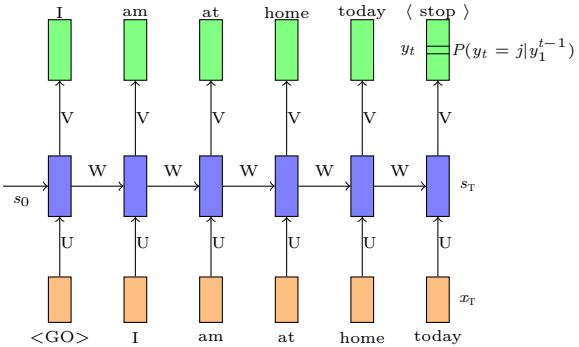
$$s_t = \sigma(Ws_{t-1} + Ux_t + b)$$

$$P(y_t = j|y_1^{t-1}) = softmax(Vs_t + c)_j$$

- **Parameters:** U, V, W, b, c

Data:

India, officially the Republic of India, is a country in South Asia. It is the seventh-largest country by area, ...



- **Data:** All sentences from any large corpus (say wikipedia)
- **Model:**

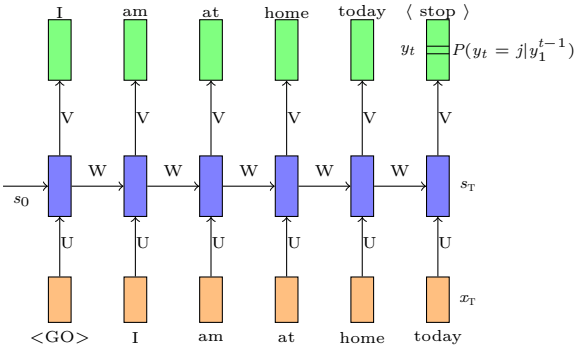
$$s_t = \sigma(Ws_{t-1} + Ux_t + b)$$

$$P(y_t = j|y_1^{t-1}) = softmax(Vs_t + c)_j$$

- **Parameters:** U, V, W, b, c
- **Loss:**

Data:

> India, officially the Republic of India, is a country in South Asia. It is the seventh-largest country by area, ...



Data:

India, officially the Republic of India, is a country in South Asia. It is the seventh-largest country by area, ...

- **Data:** All sentences from any large corpus (say wikipedia)
- **Model:**

$$s_t = \sigma(Ws_{t-1} + Ux_t + b)$$

$$P(y_t = j|y_1^{t-1}) = softmax(Vs_t + c)_j$$

- **Parameters:** U, V, W, b, c
- **Loss:**

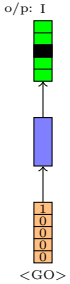
$$\mathscr{L}(\theta) = \sum_{t=1}^{T} \mathscr{L}_t(\theta)$$

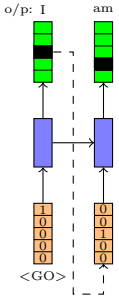
$$\mathscr{L}_t(\theta) = -\log P(y_t = \ell_t|y_1^{t-1})$$

where ℓ_t is the true word at time step t

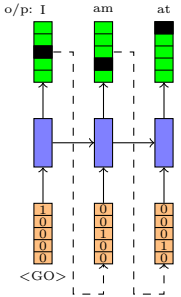
- What is the input at each time step?

- What is the input at each time step?

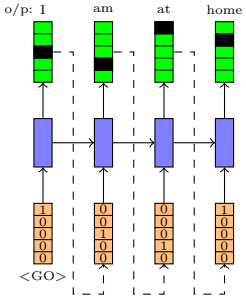




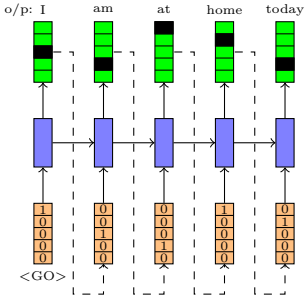
- What is the input at each time step?
- It is simply the word that we predicted at the previous time step



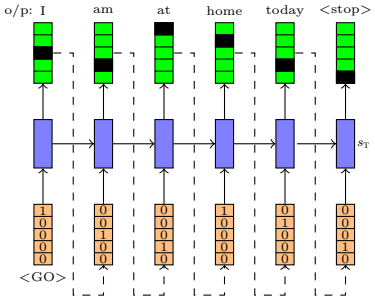
- What is the input at each time step?
- It is simply the word that we predicted at the previous time step



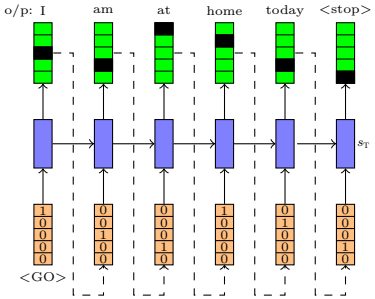
- What is the input at each time step?
- It is simply the word that we predicted at the previous time step



- What is the input at each time step?
- It is simply the word that we predicted at the previous time step

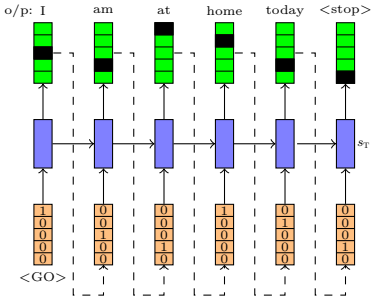


- What is the input at each time step?
- It is simply the word that we predicted at the previous time step

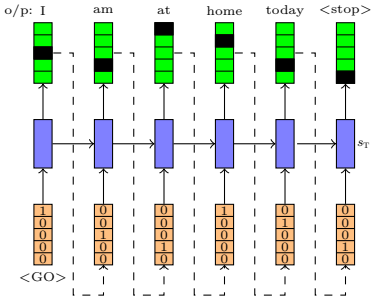


- What is the input at each time step?
- It is simply the word that we predicted at the previous time step
- In general

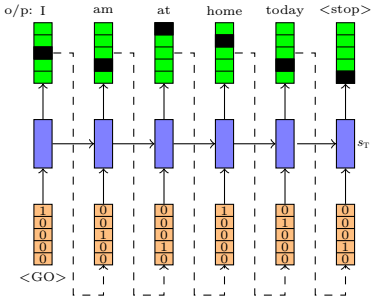
$$s_t = RNN(s_{t-1}, x_t)$$



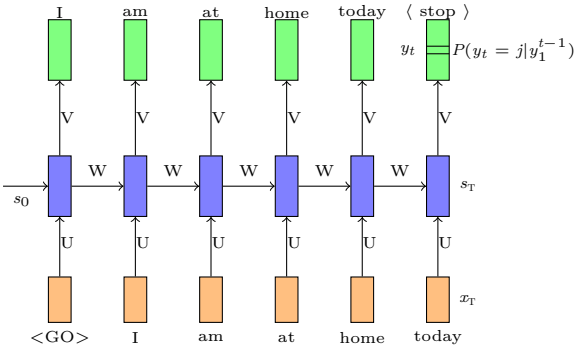
- What is the input at each time step?
- It is simply the word that we predicted at the previous time step
- In general
$$s_t = RNN(s_{t-1}, x_t)$$
- Let j be the index of the word which has been assigned the max probability at time step $t-1$
$$x_t = e(v_j)$$



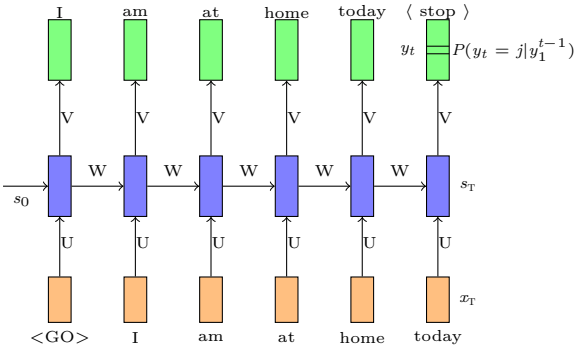
- What is the input at each time step?
- It is simply the word that we predicted at the previous time step
- In general
$$s_t = RNN(s_{t-1}, x_t)$$
- Let j be the index of the word which has been assigned the max probability at time step $t-1$
$$x_t = e(v_j)$$
- x_t is essentially a one-hot vector $(e(v_j))$representing the j^{th} word in the vocabulary



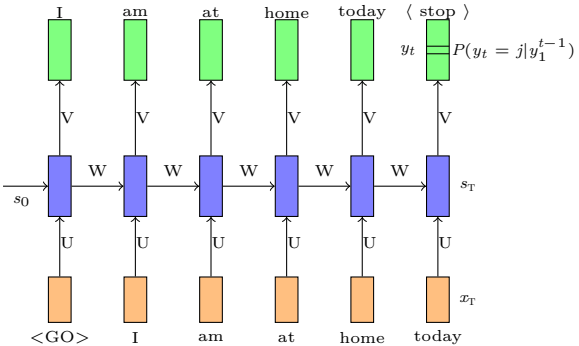
- What is the input at each time step?
- It is simply the word that we predicted at the previous time step
- In general
$$s_t = RNN(s_{t-1}, x_t)$$
- Let j be the index of the word which has been assigned the max probability at time step $t-1$
$$x_t = e(v_j)$$
- x_t is essentially a one-hot vector $(e(v_j))$representing the j^{th} word in the vocabulary
- In practice, instead of one hot representation we use a pre-trained word embedding of the j^{th} word



- Notice that s_0 is not computed but just randomly initialized



- Notice that s_0 is not computed but just randomly initialized
- We learn it along with the other parameters of RNN (or LSTM or GRU)



- Notice that s_0 is not computed but just randomly initialized
- We learn it along with the other parameters of RNN (or LSTM or GRU)
- We will return back to this later

- Before moving on we will see a compact way of writing the function computed by RNN, GRU and LSTM

$$s_t = \sigma(U\ x_t + W s_{t-1} + b)$$

- Before moving on we will see a compact way of writing the function computed by RNN, GRU and LSTM

$$s_t = \sigma(U\ x_t + W s_{t-1} + b)$$

$$s_t = \text{RNN}(\ s_{t-1}, x_t)$$

- Before moving on we will see a compact way of writing the function computed by RNN, GRU and LSTM

$$s_t = \sigma(U\ x_t + W s_{t-1} + b) \qquad \tilde{s}_t = \sigma(W(o_t \odot\ s_{t-1}) + U x_t + b)$$
$$s_t = i_t \odot\ s_{t-1} + (1 - i_t) \odot \tilde{s}_t$$

$$s_t = \text{RNN}(\ s_{t-1}, x_t)$$

- Before moving on we will see a compact way of writing the function computed by RNN, GRU and LSTM

$$s_t = \sigma(U\ x_t + W s_{t-1} + b) \qquad \tilde{s}_t = \sigma(W(o_t \odot\ s_{t-1}) + U x_t + b)$$
$$s_t = i_t \odot\ s_{t-1} + (1 - i_t) \odot \tilde{s}_t$$

$$s_t = \text{RNN}(\ s_{t-1}, x_t) \qquad s_t = \text{GRU}(\ s_{t-1}, x_t)$$

- Before moving on we will see a compact way of writing the function computed by RNN, GRU and LSTM

$$s_t = \sigma(U\ x_t + W s_{t-1} + b)$$

$$\tilde{s}_t = \sigma(W(o_t \odot\ s_{t-1}) + U x_t + b)$$
$$s_t = i_t \odot\ s_{t-1} + (1 - i_t) \odot \tilde{s}_t$$

$$\tilde{s}_t = \sigma(W\ h_{t-1} + U x_t + b)$$
$$s_t = f_t \odot\ s_{t-1} + i_t \odot \tilde{s}_t$$
$$h_t = o_t \odot \sigma(s_t)$$

$$s_t = \text{RNN}(\ s_{t-1}, x_t)$$

$$s_t = \text{GRU}(\ s_{t-1}, x_t)$$

- Before moving on we will see a compact way of writing the function computed by RNN, GRU and LSTM

$$s_t = \sigma(U\ x_t + W s_{t-1} + b)$$

$$\tilde{s}_t = \sigma(W(o_t \odot\ s_{t-1}) + U x_t + b)$$
$$s_t = i_t \odot\ s_{t-1} + (1 - i_t) \odot \tilde{s}_t$$

$$\tilde{s}_t = \sigma(W\ h_{t-1} + U x_t + b)$$
$$s_t = f_t \odot\ s_{t-1} + i_t \odot \tilde{s}_t$$
$$h_t = o_t \odot \sigma(s_t)$$

$$s_t = \text{RNN}(\ s_{t-1}, x_t)$$

$$s_t = \text{GRU}(\ s_{t-1}, x_t)$$

$$h_t, s_t = \text{LSTM}(\ h_{t-1}, s_{t-1}, x_t)$$

- Before moving on we will see a compact way of writing the function computed by RNN, GRU and LSTM

$$s_t = \sigma(U\ x_t + W s_{t-1} + b)$$

$$\tilde{s}_t = \sigma(W(o_t \odot\ s_{t-1}) + U x_t + b)$$
$$s_t = i_t \odot\ s_{t-1} + (1 - i_t) \odot \tilde{s}_t$$

$$\tilde{s}_t = \sigma(W\ h_{t-1} + U x_t + b)$$
$$s_t = f_t \odot\ s_{t-1} + i_t \odot \tilde{s}_t$$
$$h_t = o_t \odot \sigma(s_t)$$

$$s_t = \text{RNN}(\ s_{t-1}, x_t)$$

$$s_t = \text{GRU}(\ s_{t-1}, x_t)$$

$$h_t, s_t = \text{LSTM}(\ h_{t-1}, s_{t-1}, x_t)$$

- Before moving on we will see a compact way of writing the function computed by RNN, GRU and LSTM
- We will use these notations going forward

- So far we have seen how to model the conditional probability distribution $P(y_t|y_1^{t-1})$

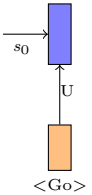
- So far we have seen how to model the conditional probability distribution $P(y_t|y_1^{t-1})$
- More informally, we have seen how to generate a sentence given previous words

- So far we have seen how to model the conditional probability distribution $P(y_t|y_1^{t-1})$
- More informally, we have seen how to generate a sentence given previous words
- What if we want to generate a sentence given an image?

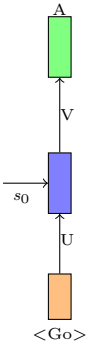
- So far we have seen how to model the conditional probability distribution $P(y_t|y_1^{t-1})$
- More informally, we have seen how to generate a sentence given previous words
- What if we want to generate a sentence given an image?

A man throwing
a frisbee in a park

- So far we have seen how to model the conditional probability distribution $P(y_t|y_1^{t-1})$
- More informally, we have seen how to generate a sentence given previous words
- What if we want to generate a sentence given an image?

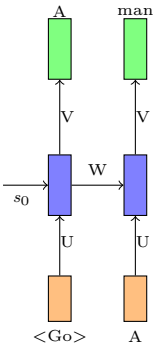


A man throwing
a frisbee in a park



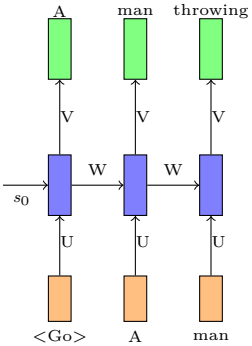
A man throwing
a frisbee in a park

- So far we have seen how to model the conditional probability distribution $P(y_t|y_1^{t-1})$
- More informally, we have seen how to generate a sentence given previous words
- What if we want to generate a sentence given an image?



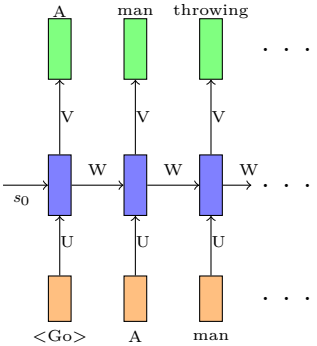
A man throwing a frisbee in a park

- So far we have seen how to model the conditional probability distribution $P(y_t|y_1^{t-1})$
- More informally, we have seen how to generate a sentence given previous words
- What if we want to generate a sentence given an image?



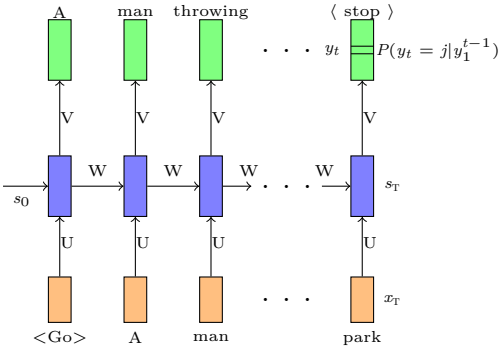
A man throwing a frisbee in a park

- So far we have seen how to model the conditional probability distribution $P(y_t|y_1^{t-1})$
- More informally, we have seen how to generate a sentence given previous words
- What if we want to generate a sentence given an image?



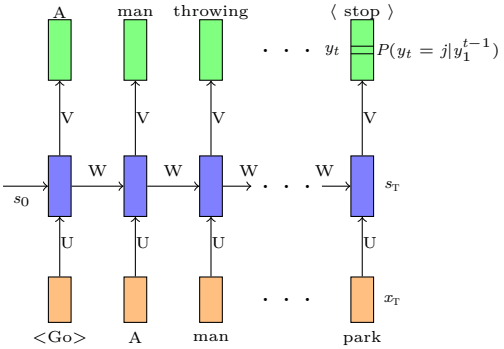
A man throwing
a frisbee in a park

- So far we have seen how to model the conditional probability distribution $P(y_t|y_1^{t-1})$
- More informally, we have seen how to generate a sentence given previous words
- What if we want to generate a sentence given an image?



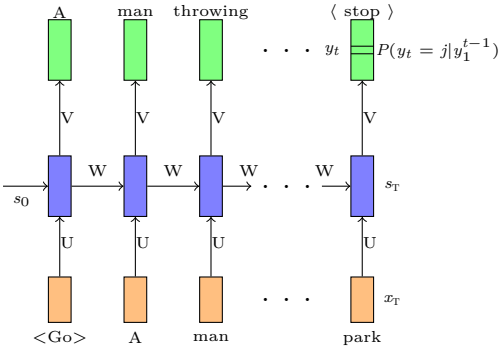
A man throwing
a frisbee in a park

- So far we have seen how to model the conditional probability distribution $P(y_t|y_1^{t-1})$
- More informally, we have seen how to generate a sentence given previous words
- What if we want to generate a sentence given an image?



A man throwing a frisbee in a park

- So far we have seen how to model the conditional probability distribution $P(y_t|y_1^{t-1})$
- More informally, we have seen how to generate a sentence given previous words
- What if we want to generate a sentence given an image?
- We are now interested in $P(y_t|y_1^{t-1}, I)$ instead of $P(y_t|y_1^{t-1})$ where I is an image

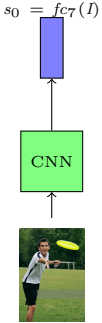


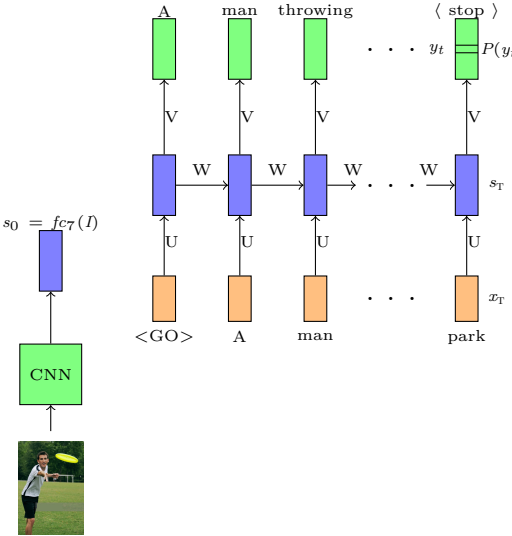
A man throwing a frisbee in a park

- So far we have seen how to model the conditional probability distribution $P(y_t|y_1^{t-1})$
- More informally, we have seen how to generate a sentence given previous words
- What if we want to generate a sentence given an image?
- We are now interested in $P(y_t|y_1^{t-1}, I)$ instead of $P(y_t|y_1^{t-1})$ where I is an image
- Notice that $P(y_t|y_1^{t-1}, I)$ is again a conditional distribution

- Earlier we modeled $P(y_t|y_1^{t-1})$ as

$$P(y_t|y_1^{t-1}) = P(y_t = j|s_t)$$

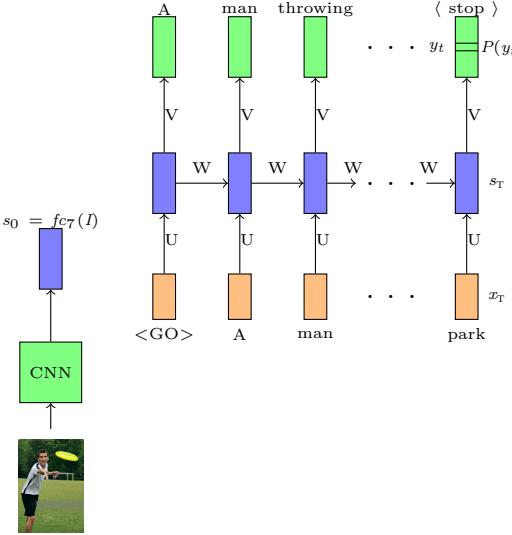




- Earlier we modeled $P(y_t|y_1^{t-1})$ as

$$P(y_t|y_1^{t-1}) = P(y_t = j|s_t)$$

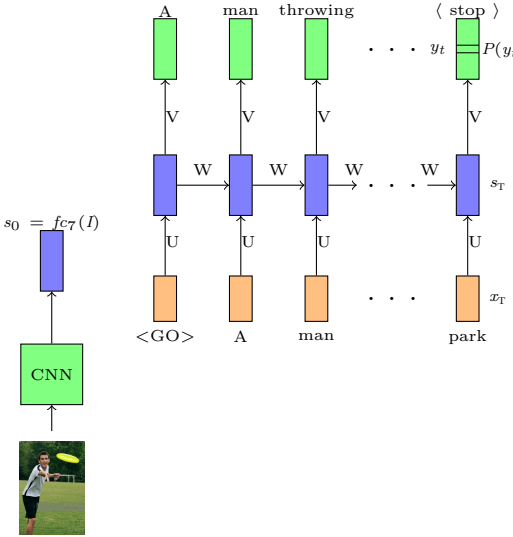
- Where s_t was a state capturing all the previous words



- Earlier we modeled $P(y_t|y_1^{t-1})$ as

$$P(y_t|y_1^{t-1}) = P(y_t = j|s_t)$$

- Where s_t was a state capturing all the previous words
- We could now model $P(y_t = j|y_1^{t-1}, I)$ as $P(y_t = j|s_t, f_{c7}(I))$



- Earlier we modeled $P(y_t|y_1^{t-1})$ as

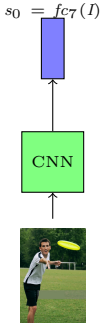
$$P(y_t|y_1^{t-1}) = P(y_t = j|s_t)$$

- Where s_t was a state capturing all the previous words
- We could now model $P(y_t = j|y_1^{t-1}, I)$ as $P(y_t = j|s_t, f_{c_7}(I))$
- where $fc_7(I)$ is the representation obtained from the fc_7 layer of an image

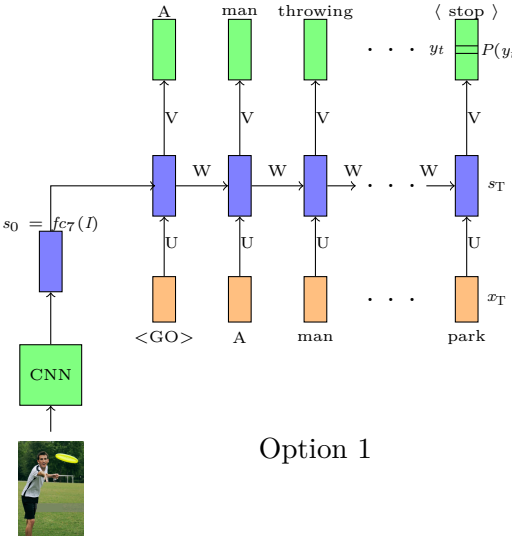
- There are many ways of making $P(y_t = j)$ conditional on $f_{c_7}(I)$

- There are many ways of making $P(y_t = j)$ conditional on $f_{c_7}(I)$
- Let us see two such options

- **Option 1:** Set $s_0 = f_{c_7}(I)$

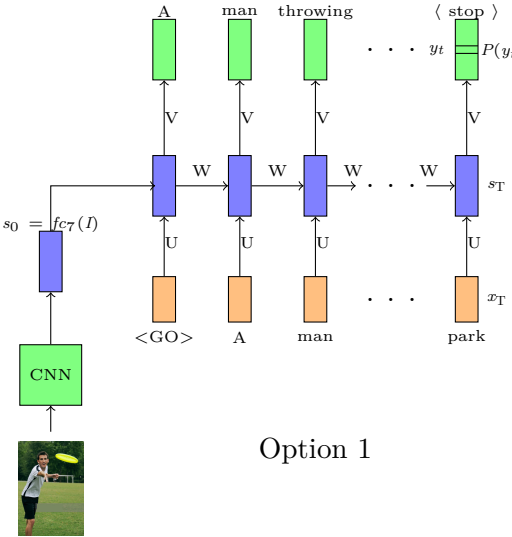


Option 1



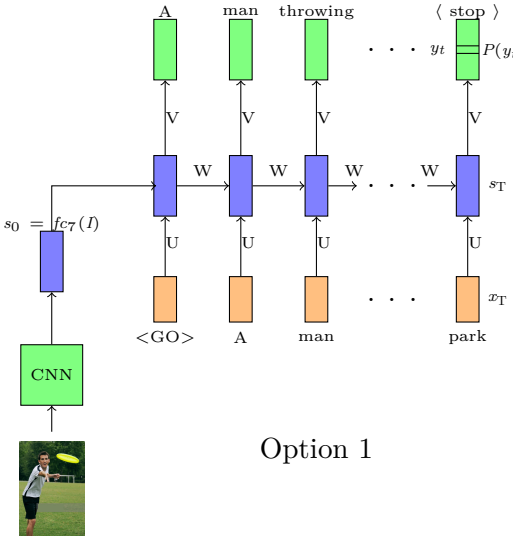
Option 1

- **Option 1:** Set $s_0 = f_{c_7}(I)$
- Now s_0 and hence all subsequent s_t's depend on $f_{c_7}(I)$



Option 1

- **Option 1:** Set $s_0 = f_{c_7}(I)$
- Now s_0 and hence all subsequent s_t's depend on $f_{c_7}(I)$
- We can thus say that $P(y_t = j)$ depends on $f_{c_7}(I)$

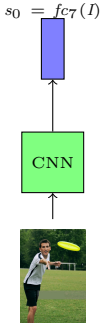


Option 1

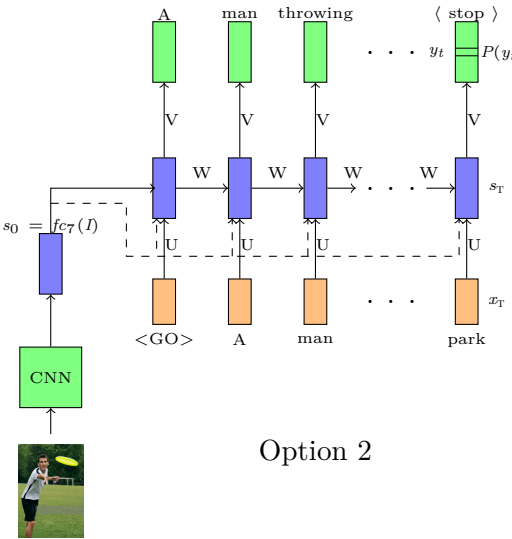
- **Option 1:** Set $s_0 = f_{c_7}(I)$
- Now s_0 and hence all subsequent s_t's depend on $f_{c_7}(I)$
- We can thus say that $P(y_t = j)$ depends on $f_{c_7}(I)$
- In other words, we are computing $P(y_t = j|s_t, f_{c_7}(I))$

- **Option 2:** Another more explicit way of doing this is to compute

$$s_t = RNN(s_{t-1}, [x_t, f_{c_7}(I)])$$



Option 2

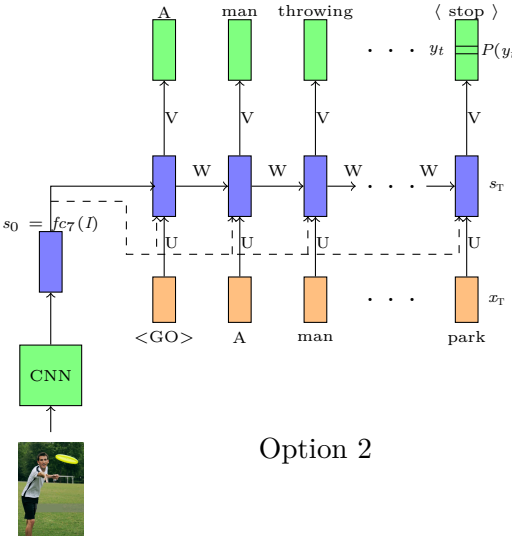


Option 2

- **Option 2:** Another more explicit way of doing this is to compute

$$s_t = RNN(s_{t-1}, [x_t, f_{c_7}(I)])$$

- In other words we are explicitly using $f_{c_7}(I)$ to compute s_t and hence $P(y_t = j)$



Option 2

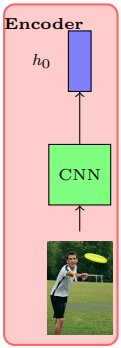
- **Option 2:** Another more explicit way of doing this is to compute

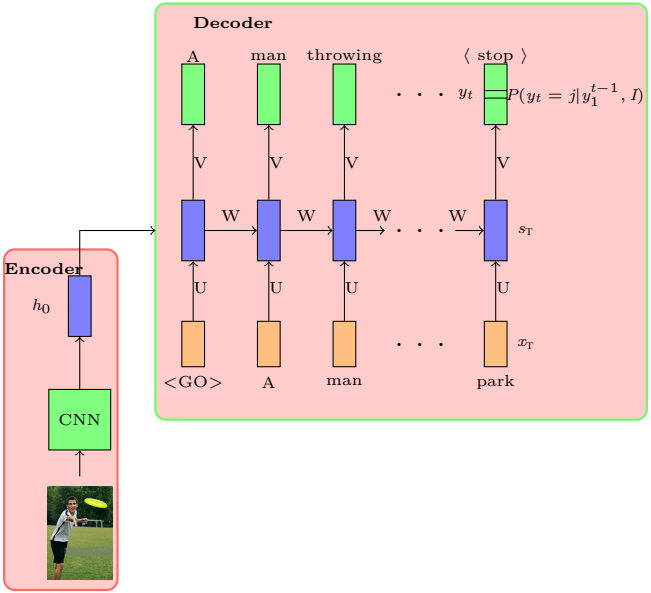
$$s_t = RNN(s_{t-1}, [x_t, f_{c_7}(I)])$$

- In other words we are explicitly using $f_{c_7}(I)$ to compute s_t and hence $P(y_t = j)$
- You could think of other ways of conditioning $P(y_t = j)$ on f_{c_7}

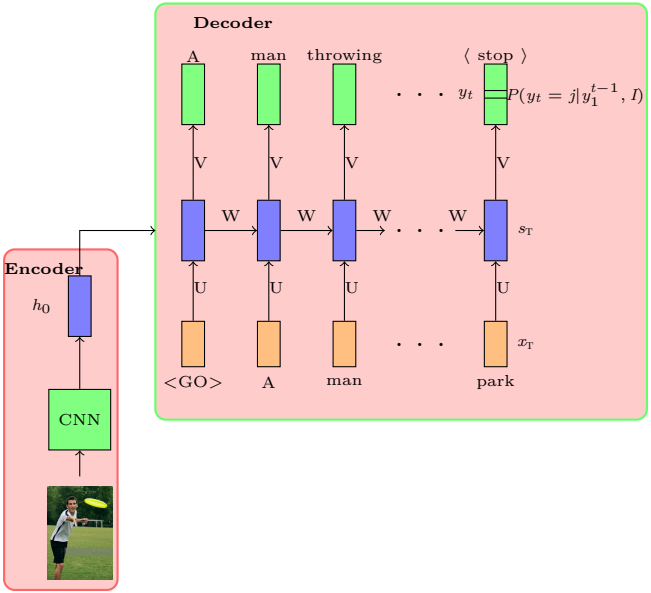
- Let us look at the full architecture

- Let us look at the full architecture
- A CNN is first used to **encode** the image

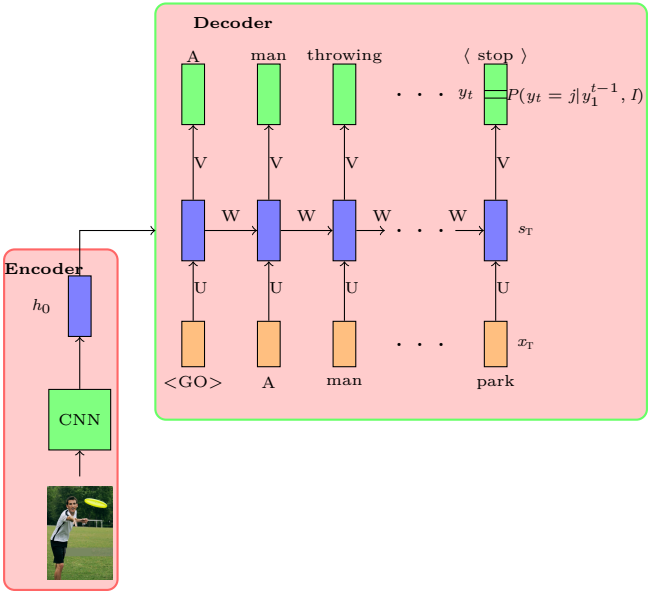




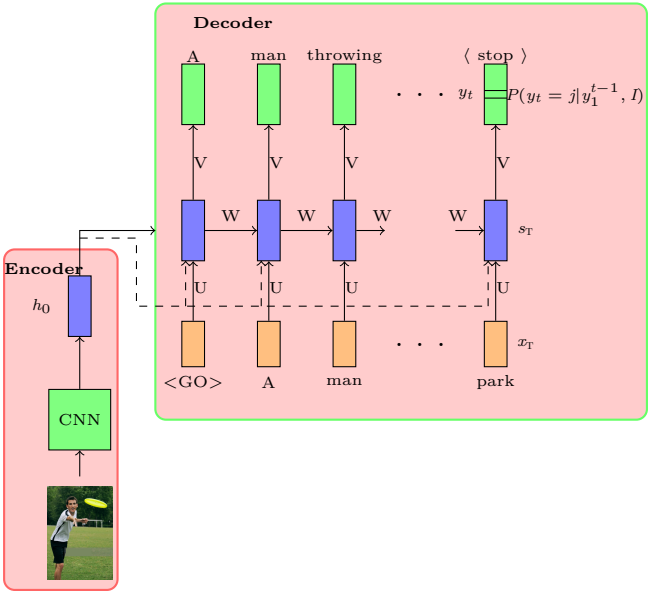
- Let us look at the full architecture
- A CNN is first used to **encode** the image
- A RNN is then used to decode (generate) a sentence from the encoding



- Let us look at the full architecture
- A CNN is first used to **encode** the image
- A RNN is then used to decode (generate) a sentence from the encoding
- This is a typical **encoder decoder architecture**



- Let us look at the full architecture
- A CNN is first used to **encode** the image
- A RNN is then used to decode (generate) a sentence from the encoding
- This is a typical **encoder decoder architecture**
- Both the encoder and decoder use a neural network



- Let us look at the full architecture
- A CNN is first used to **encode** the image
- A RNN is then used to decode (generate) a sentence from the encoding
- This is a typical **encoder decoder architecture**
- Both the encoder and decoder use a neural network
- Alternatively, the encoder's output can be fed to every step of the decoder