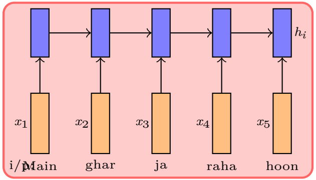
Module 16.3: Attention Mechanism

o/p : I am going home

- Let us motivate the task of attention with the help of MT

i/p : Main ghar ja raha hoon

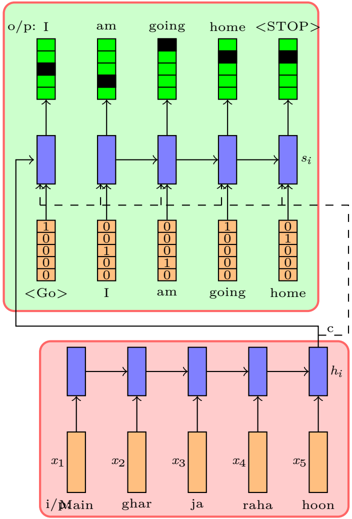
o/p : I am going home



i/p : Main ghar ja raha hoon

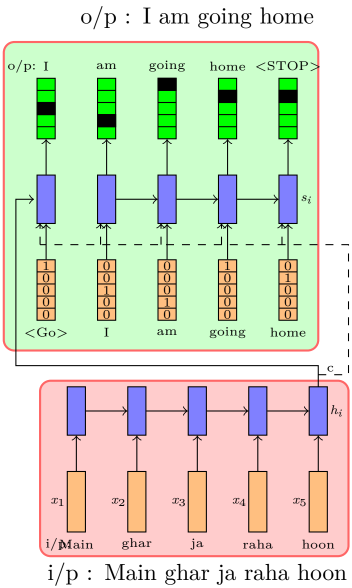
- Let us motivate the task of attention with the help of MT
- The encoder reads the sentences only once and encodes it

o/p : I am going home



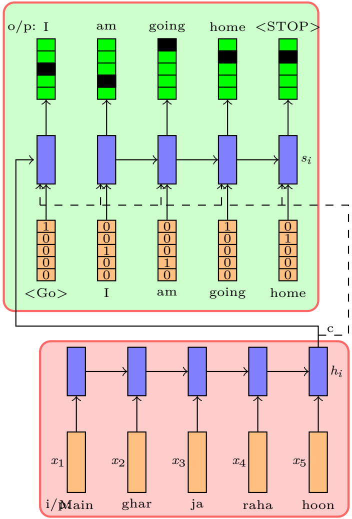
i/p : Main ghar ja raha hoon

- Let us motivate the task of attention with the help of MT
- The encoder reads the sentences only once and encodes it
- At each timestep the decoder uses this embedding to produce a new word



- Let us motivate the task of attention with the help of MT
- The encoder reads the sentences only once and encodes it
- At each timestep the decoder uses this embedding to produce a new word
- Is this how humans translate a sentence ?

o/p : I am going home



i/p : Main ghar ja raha hoon

- Let us motivate the task of attention with the help of MT
- The encoder reads the sentences only once and encodes it
- At each timestep the decoder uses this embedding to produce a new word
- Is this how humans translate a sentence ? Not really!

o/p : I am going home

i/p : Main ghar ja raha hoon

- Humans try to produce each word in the output by focusing only on certain words in the input

o/p : I am going home

i/p : Main ghar ja raha hoon

- Humans try to produce each word in the output by focusing only on certain words in the input
- Essentially at each time step we come up with a distribution on the input words

o/p : I am going home

t_1 : [1 0 0 0 0]

i/p : Main ghar ja raha hoon

- Humans try to produce each word in the output by focusing only on certain words in the input
- Essentially at each time step we come up with a distribution on the input words

o/p : I am going home

t_1 : [1 0 0 0 0]

t_2 : [0 0 0 0 1]

i/p : Main ghar ja raha hoon

- Humans try to produce each word in the output by focusing only on certain words in the input
- Essentially at each time step we come up with a distribution on the input words

o/p : I am going home

t_1 : [1 0 0 0 0]

t_2 : [0 0 0 0 1]

t_3 : [0 0 0.5 0.5 0]

i/p : Main ghar ja raha hoon

- Humans try to produce each word in the output by focusing only on certain words in the input
- Essentially at each time step we come up with a distribution on the input words

o/p : I am going home

t_1 : [1 0 0 0 0]

t_2 : [0 0 0 0 1]

t_3 : [0 0 0.5 0.5 0]

t_4 : [0 1 0 0 0]

i/p : Main ghar ja raha hoon

- Humans try to produce each word in the output by focusing only on certain words in the input
- Essentially at each time step we come up with a distribution on the input words

o/p : I am going home

t_1 : [1 0 0 0 0]

t_2 : [0 0 0 0 1]

t_3 : [0 0 0.5 0.5 0]

t_4 : [0 1 0 0 0]

i/p : Main ghar ja raha hoon

- Humans try to produce each word in the output by focusing only on certain words in the input
- Essentially at each time step we come up with a distribution on the input words
- This distribution tells us how much attention to pay to each input words at each time step

o/p : I am going home

t_1 : [1 0 0 0 0]

t_2 : [0 0 0 0 1]

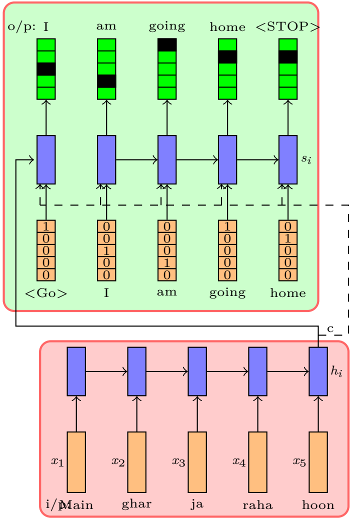
t_3 : [0 0 0.5 0.5 0]

t_4 : [0 1 0 0 0]

i/p : Main ghar ja raha hoon

- Humans try to produce each word in the output by focusing only on certain words in the input
- Essentially at each time step we come up with a distribution on the input words
- This distribution tells us how much attention to pay to each input words at each time step
- Ideally, at each time-step we should feed only this relevant information (i.e. encodings of relevant words) to the decoder

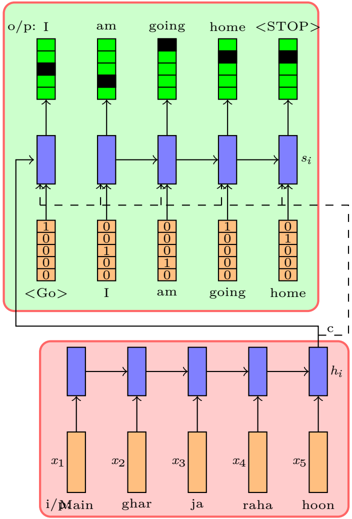
o/p : I am going home



i/p : Main ghar ja raha hoon

- Let us revisit the decoder that we have seen so far

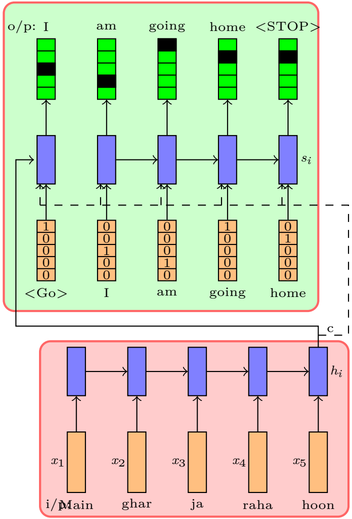
o/p : I am going home



i/p : Main ghar ja raha hoon

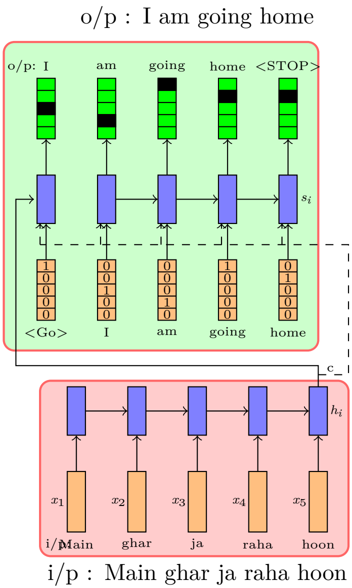
- Let us revisit the decoder that we have seen so far
- We either feed in the encoder information only once(at s_0)

o/p : I am going home

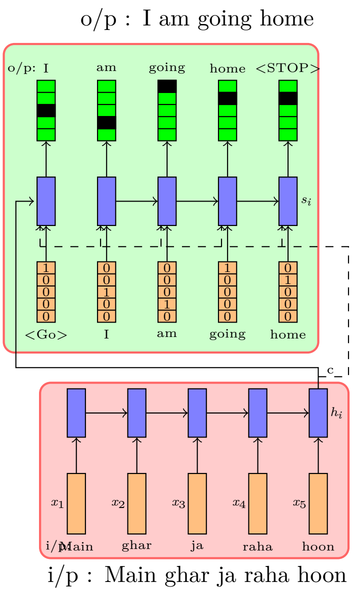


i/p : Main ghar ja raha hoon

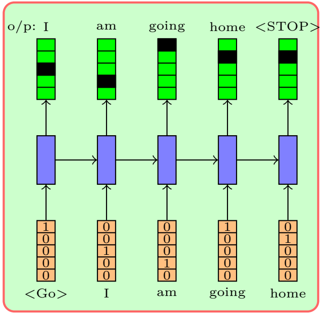
- Let us revisit the decoder that we have seen so far
- We either feed in the encoder information only once(at s_0)
- Or we feed the same encoder information at each time step

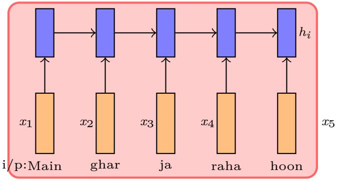


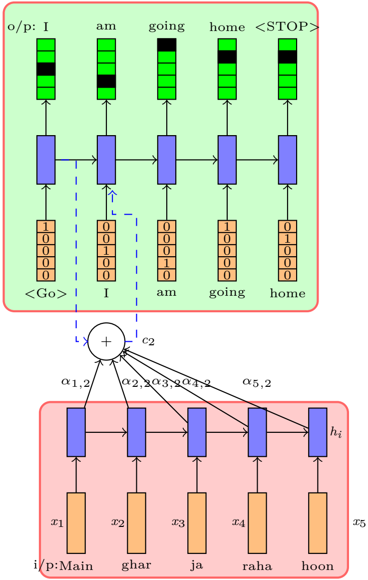
- Let us revisit the decoder that we have seen so far
- We either feed in the encoder information only once(at s_0)
- Or we feed the same encoder information at each time step
- Now suppose an oracle told you which words to focus on at a given time-step t



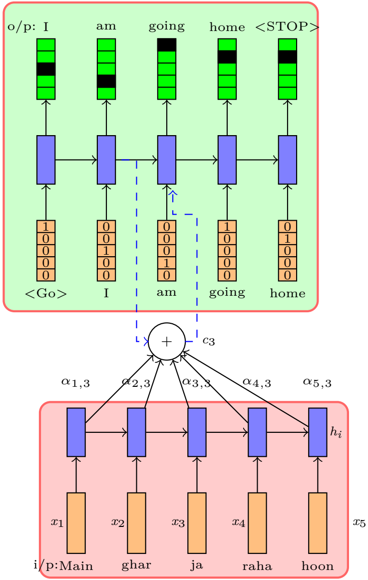
- Let us revisit the decoder that we have seen so far
- We either feed in the encoder information only once(at s_0)
- Or we feed the same encoder information at each time step
- Now suppose an oracle told you which words to focus on at a given time-step t
- Can you think of a smarter way of feeding information to the decoder?


o/p: I
am
going
home
<STOP>
<Go>
I
am
going
home

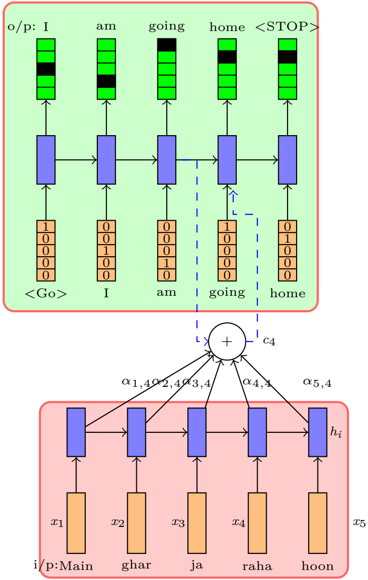

h_i
x_1
x_2
x_3
x_4
x_5
i/p:Main
ghar
ja
raha
hoon



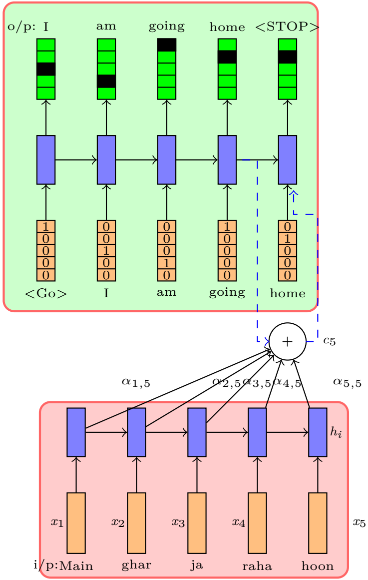
- We could just take a weighted average of the corresponding word representations and feed it to the decoder



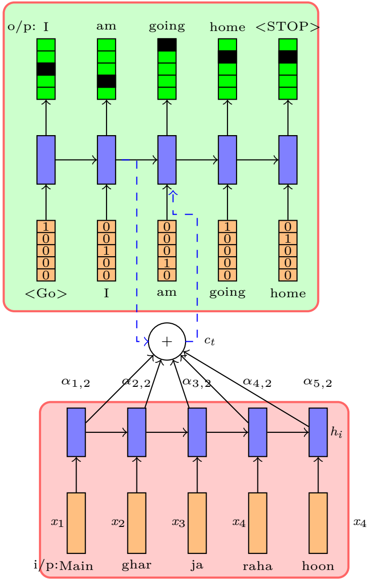
- We could just take a weighted average of the corresponding word representations and feed it to the decoder
- For example at timestep 3, we can just take a weighted average of the representations of 'ja' and 'raha'

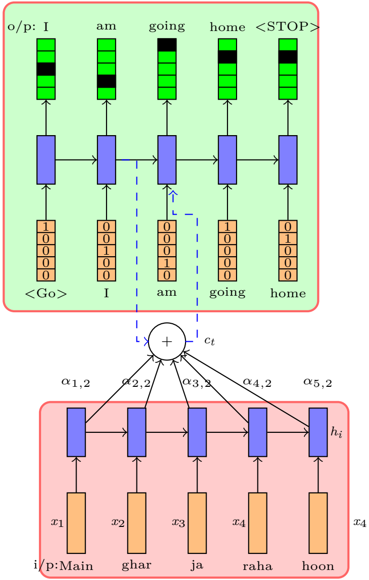


- We could just take a weighted average of the corresponding word representations and feed it to the decoder
- For example at timestep 3, we can just take a weighted average of the representations of 'ja' and 'raha'
- Intuitively this should work better because we are not overloading the decoder with irrelevant information (about words that do not matter at this time step)

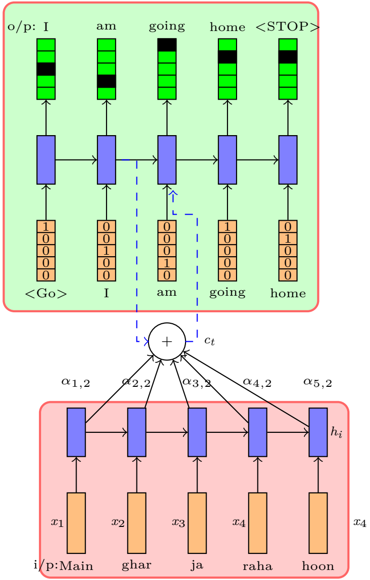


- We could just take a weighted average of the corresponding word representations and feed it to the decoder
- For example at timestep 3, we can just take a weighted average of the representations of 'ja' and 'raha'
- Intuitively this should work better because we are not overloading the decoder with irrelevant information (about words that do not matter at this time step)
- How do we convert this intuition into a model ?

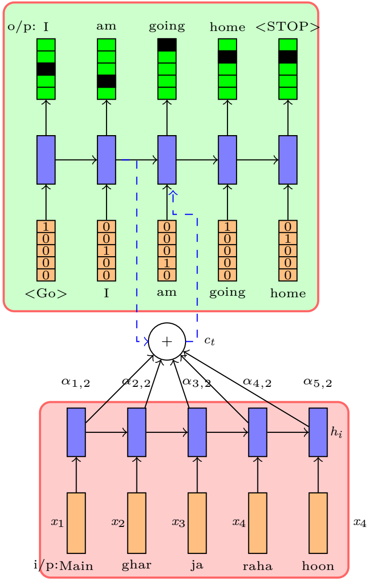

o/p: I
am
going
home
<STOP>
1 0 0 0 0
0 0 1 0 0
0 0 0 1 0
1 0 0 0 0
0 1 0 0 0
<Go>
I
am
going
home
c_t
$\alpha_{1,2}$
$\alpha_{2,2}$
$\alpha_{3,2}$
$\alpha_{4,2}$
$\alpha_{5,2}$
h_i
x_1
x_2
x_3
x_4
x_4
i/p:Main
ghar
ja
raha
hoon



- Of course in practice we will not have this oracle

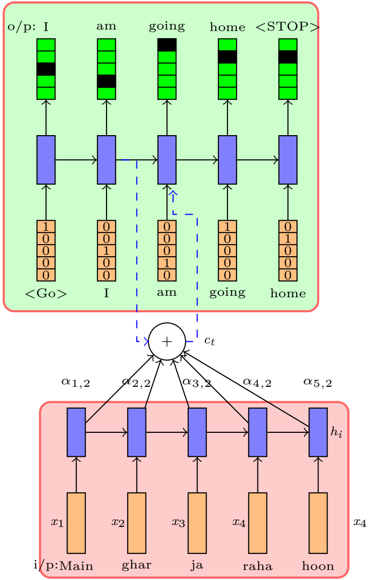


- Of course in practice we will not have this oracle
- The machine will have to learn this from the data



- Of course in practice we will not have this oracle
- The machine will have to learn this from the data
- To enable this we define a function

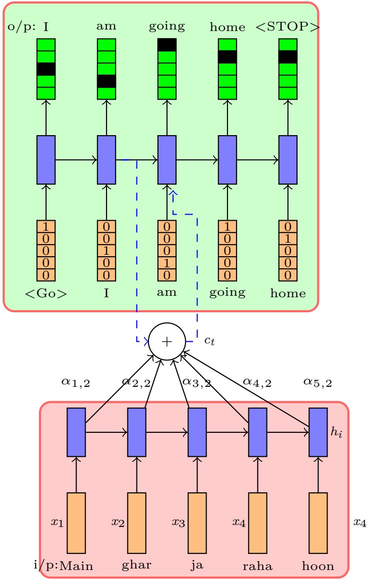
$$e_{jt} = f_{ATT}(s_{t-1}, c_j)$$



- Of course in practice we will not have this oracle
- The machine will have to learn this from the data
- To enable this we define a function

$$e_{jt} = f_{ATT}(s_{t-1}, c_j)$$

- This quantity captures the importance of the j^{th} input word for decoding the t^{th} output word (we will see the exact form of f_{ATT} later)

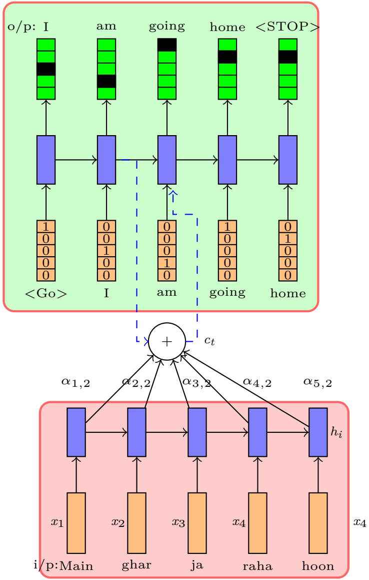


- Of course in practice we will not have this oracle
- The machine will have to learn this from the data
- To enable this we define a function

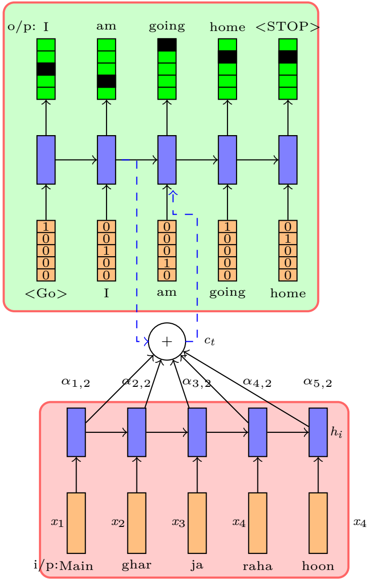
$$e_{jt} = f_{ATT}(s_{t-1}, c_j)$$

- This quantity captures the importance of the j^{th} input word for decoding the t^{th} output word (we will see the exact form of f_{ATT} later)
- We can normalize these weights by using the softmax function

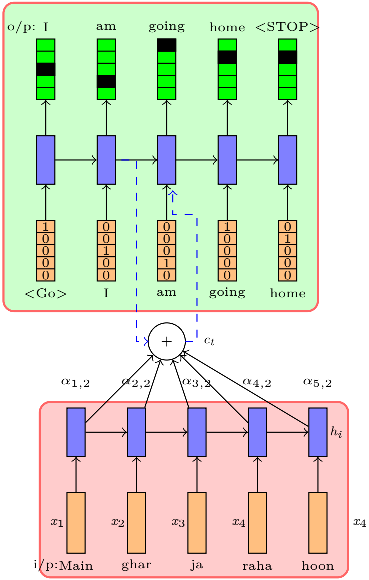
$$\alpha_{jt} = \frac{exp(e_{jt})}{\sum_{j=1}^{M} exp(e_{jt})}$$



o/p: I
am
going
home
<STOP>
1 0 0 0 0
0 0 1 0 0
0 0 0 1 0
1 0 0 0 0
0 1 0 0 0
<Go>
I
am
going
home
+
c_t
$\alpha_{1,2}$
$\alpha_{2,2}$
$\alpha_{3,2}$
$\alpha_{4,2}$
$\alpha_{5,2}$
h_i
x_1
x_2
x_3
x_4
x_4
i/p:Main
ghar
ja
raha
hoon

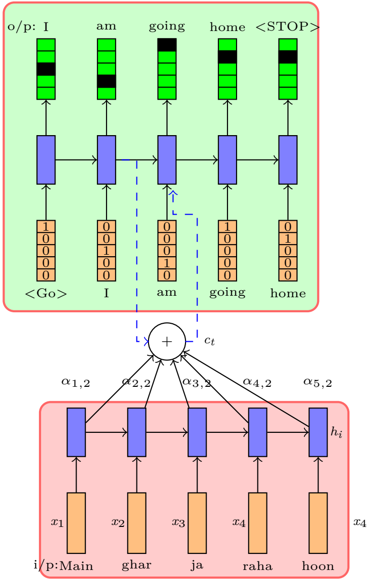


$$\alpha_{jt} = \frac{exp(e_{jt})}{\sum_{j=1}^{M} exp(e_{jt})}$$



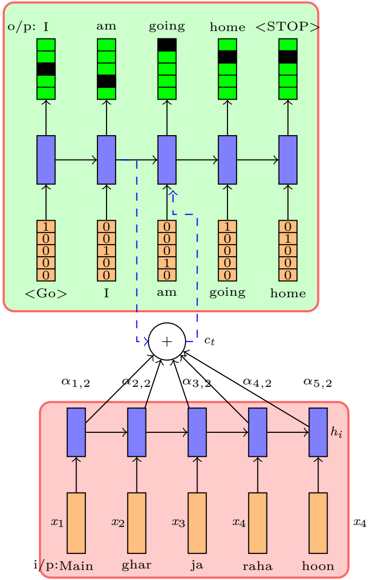
$$\alpha_{jt} = \frac{exp(e_{jt})}{\sum_{j=1}^{M} exp(e_{jt})}$$

- α_{jt} denotes the probability of focusing on the j^{th} word to produce the t^{th} output word



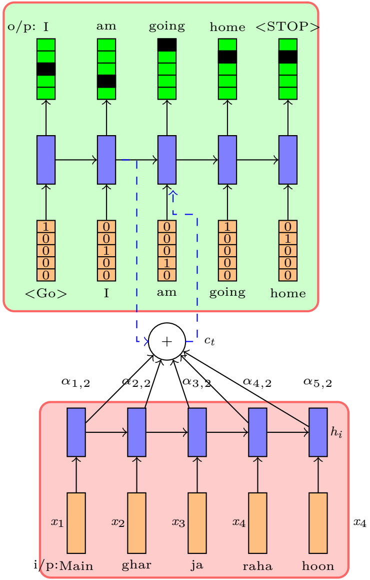
$$\alpha_{jt} = \frac{exp(e_{jt})}{\sum_{j=1}^{M} exp(e_{jt})}$$

- α_{jt} denotes the probability of focusing on the j^{th} word to produce the t^{th} output word
- We are now trying to learn the α's instead of an oracle informing us about the α's



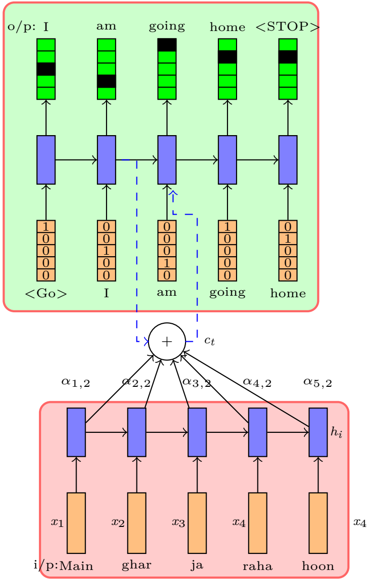
$$\alpha_{jt} = \frac{exp(e_{jt})}{\sum\limits_{j=1}^{M} exp(e_{jt})}$$

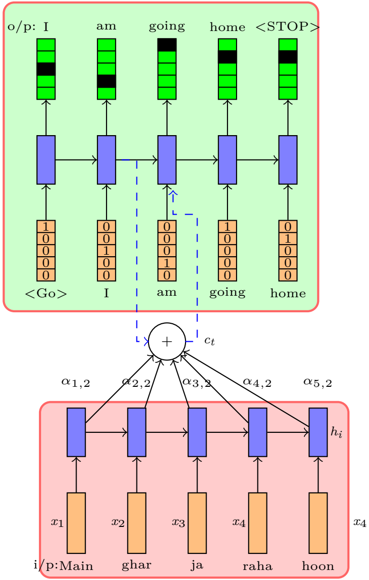
- α_{jt} denotes the probability of focusing on the j^{th} word to produce the t^{th} output word
- We are now trying to learn the α's instead of an oracle informing us about the α's
- Learning would always involve some parameters



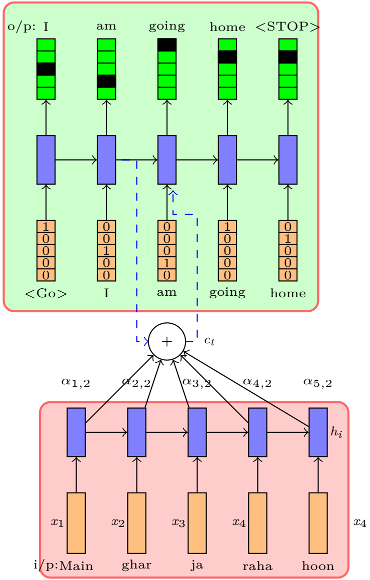
$$\alpha_{jt} = \frac{exp(e_{jt})}{\sum_{j=1}^{M} exp(e_{jt})}$$

- α_{jt} denotes the probability of focusing on the j^{th} word to produce the t^{th} output word
- We are now trying to learn the α's instead of an oracle informing us about the α's
- Learning would always involve some parameters
- So let's define a parametric form for α's


o/p: I
am
going
home
<STOP>
1 0 0 0 0
0 0 1 0 0
0 0 0 1 0
1 0 0 0 0
0 1 0 0 0
<Go>
I
am
going
home
c_t
$\alpha_{1,2}$
$\alpha_{2,2}$
$\alpha_{3,2}$
$\alpha_{4,2}$
$\alpha_{5,2}$
h_i
x_1
x_2
x_3
x_4
x_4
i/p:Main
ghar
ja
raha
hoon

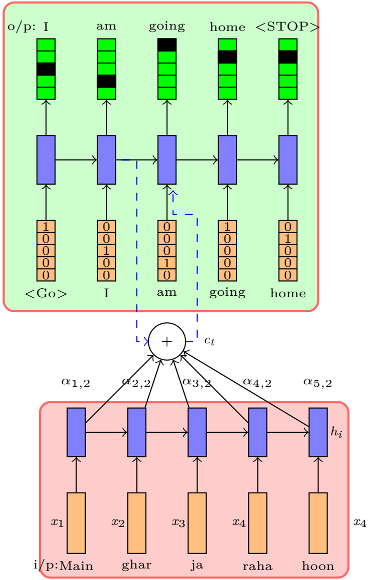


- From now on we will refer to the decoder RNN's state at the t-th timestep as s_t and the encoder RNN's state at the j-th time step as c_j



- From now on we will refer to the decoder RNN's state at the t-th timestep as s_t and the encoder RNN's state at the j-th time step as c_j
- Given these new notations, one (among many) possible choice for f_{ATT} is

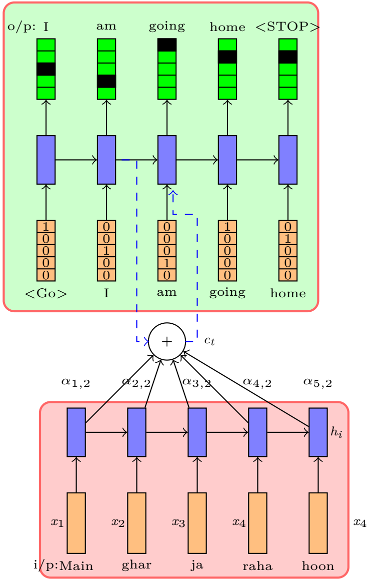
$$e_{jt} = V_{att}^T \tanh(U_{att} s_{t-1} + W_{att} c_j)$$



- From now on we will refer to the decoder RNN's state at the t-th timestep as s_t and the encoder RNN's state at the j-th time step as c_j
- Given these new notations, one (among many) possible choice for f_{ATT} is

$$e_{jt} = V_{att}^T \tanh(U_{att} s_{t-1} + W_{att} c_j)$$

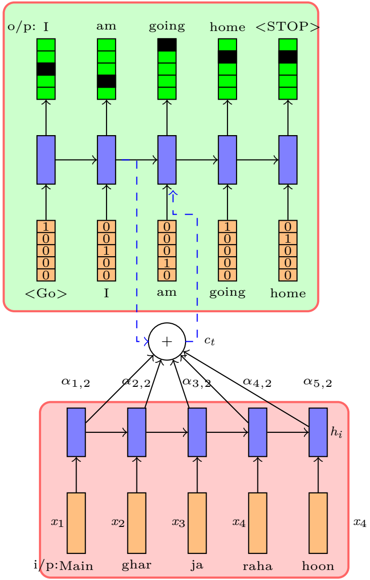
- $V_{att} \in \mathbb{R}^d$, $U_{att} \in \mathbb{R}^{d \times d}$, $W_{att} \in \mathbb{R}^{d \times d}$ are additional parameters of the model

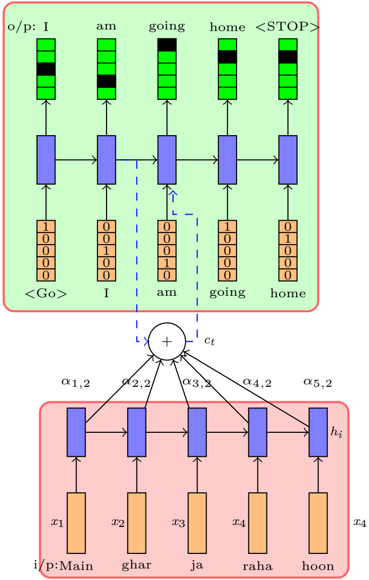


- From now on we will refer to the decoder RNN's state at the t-th timestep as s_t and the encoder RNN's state at the j-th time step as c_j
- Given these new notations, one (among many) possible choice for f_{ATT} is

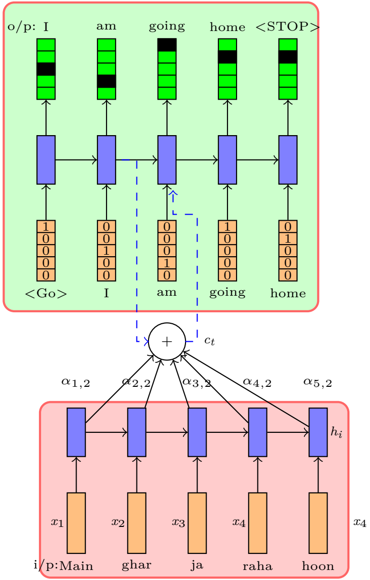
$$e_{jt} = V_{att}^T \tanh(U_{att} s_{t-1} + W_{att} c_j)$$

- $V_{att} \in \mathbb{R}^d$, $U_{att} \in \mathbb{R}^{d \times d}$, $W_{att} \in \mathbb{R}^{d \times d}$ are additional parameters of the model
- These parameters will be learned along with the other parameters of the encoder and decoder


o/p: I
am
going
home
<STOP>
<Go>
I
am
going
home
c_t
$\alpha_{1,2}$
$\alpha_{2,2}$
$\alpha_{3,2}$
$\alpha_{4,2}$
$\alpha_{5,2}$
h_i
x_1
x_2
x_3
x_4
x_4
i/p:Main
ghar
ja
raha
hoon



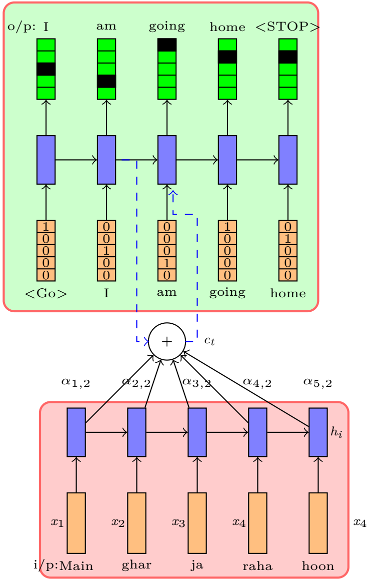
- Wait a minute !



- Wait a minute !
- This model would make a lot of sense if were given the true α's at training time

$$\alpha_{tj}^{true} = [0, 0, 0.5, 0.5, 0]$$

$$\alpha_{tj}^{pred} = [0.1, 0.1, 0.35, 0.35, 0.1]$$

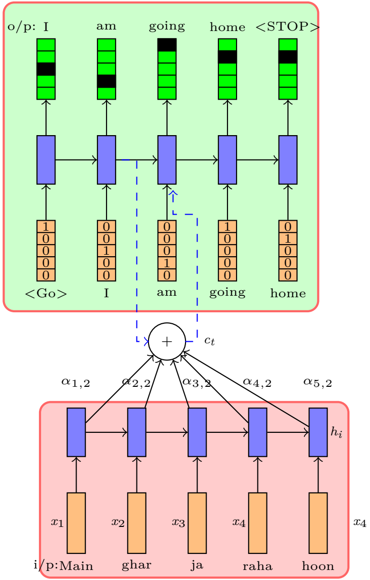


- Wait a minute !
- This model would make a lot of sense if were given the true α's at training time

$$\alpha_{tj}^{true} = [0, 0, 0.5, 0.5, 0]$$

$$\alpha_{tj}^{pred} = [0.1, 0.1, 0.35, 0.35, 0.1]$$

- We could then minimize $\mathscr{L}(\alpha^{true}, \alpha^{pred})$ in addition to $\mathscr{L}(\theta)$ as defined earlier

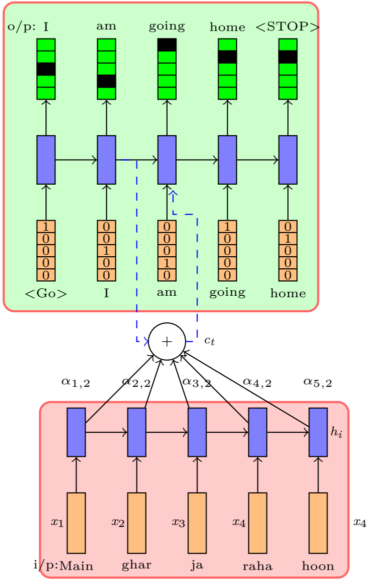


- Wait a minute !
- This model would make a lot of sense if were given the true α's at training time

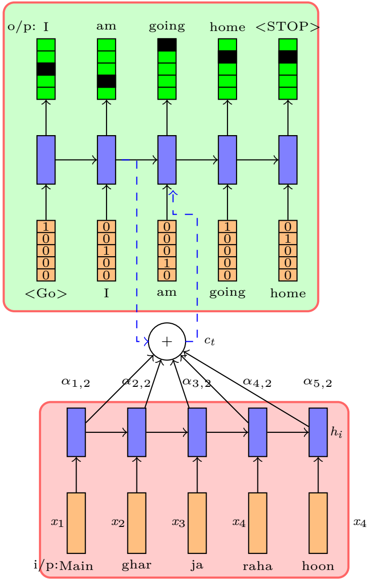
$$\alpha_{tj}^{true} = [0, 0, 0.5, 0.5, 0]$$

$$\alpha_{tj}^{pred} = [0.1, 0.1, 0.35, 0.35, 0.1]$$

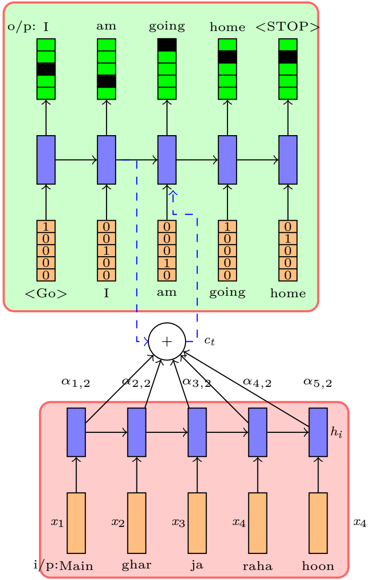
- We could then minimize $\mathscr{L}(\alpha^{true}, \alpha^{pred})$ in addition to $\mathscr{L}(\theta)$ as defined earlier
- But in practice it is very hard to get α^{true}



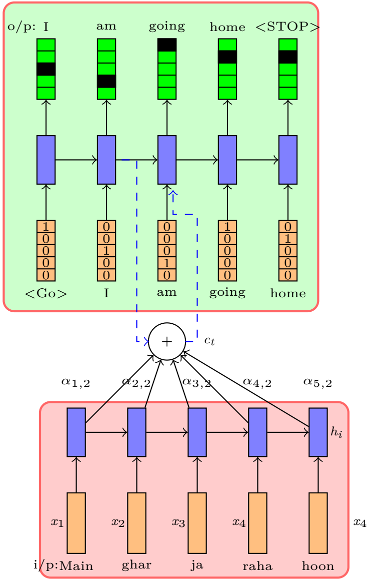
- For example, in our translation example we would want someone to manually annotate the source words which contribute to every target word

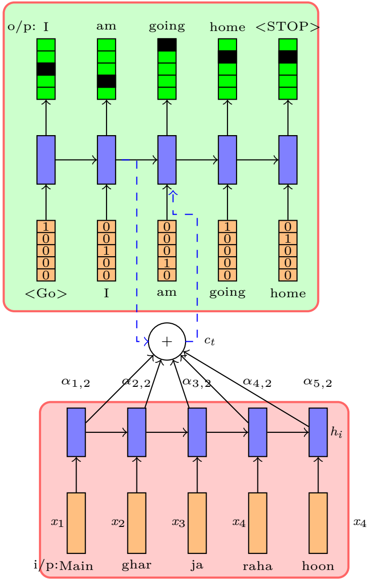


- For example, in our translation example we would want someone to manually annotate the source words which contribute to every target word
- It is hard to get such annotated data

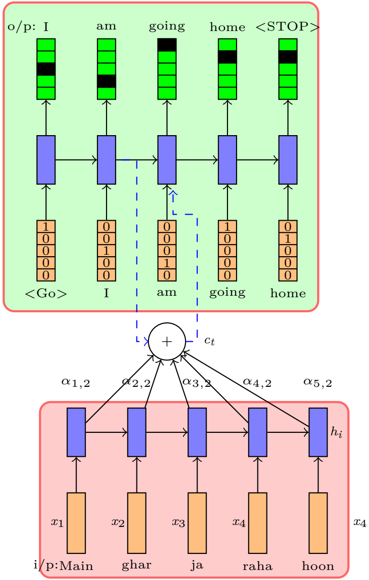


- For example, in our translation example we would want someone to manually annotate the source words which contribute to every target word
- It is hard to get such annotated data
- Then how would this model work in the absence of such data ?

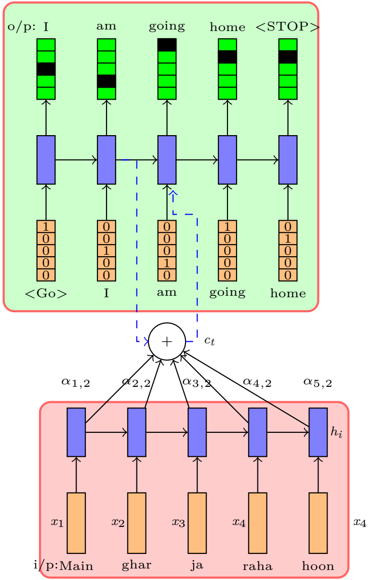

o/p: I
am
going
home
<STOP>
<Go>
I
am
going
home
c_t
$\alpha_{1,2}$
$\alpha_{2,2}$
$\alpha_{3,2}$
$\alpha_{4,2}$
$\alpha_{5,2}$
h_i
x_1
x_2
x_3
x_4
x_4
i/p:Main
ghar
ja
raha
hoon



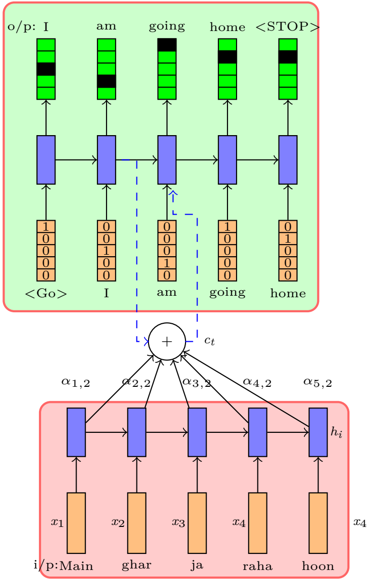
- It works because it is a better modeling choice



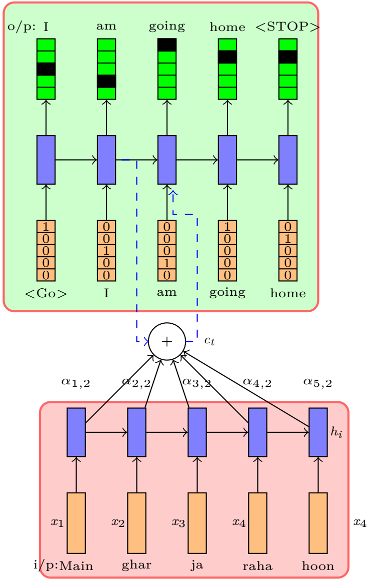
- It works because it is a better modeling choice
- This is a more informed model



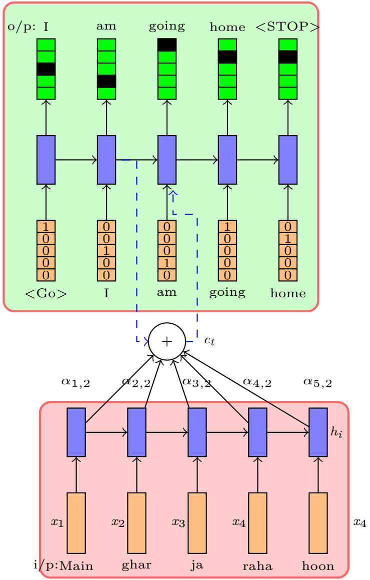
- It works because it is a better modeling choice
- This is a more informed model
- We are essentially asking the model to approach the problem in a better (more natural) way



- It works because it is a better modeling choice
- This is a more informed model
- We are essentially asking the model to approach the problem in a better (more natural) way
- Given enough data it should be able to learn these attention weights just as humans do



- It works because it is a better modeling choice
- This is a more informed model
- We are essentially asking the model to approach the problem in a better (more natural) way
- Given enough data it should be able to learn these attention weights just as humans do
- That's the hope (and hope is a good thing)



- It works because it is a better modeling choice
- This is a more informed model
- We are essentially asking the model to approach the problem in a better (more natural) way
- Given enough data it should be able to learn these attention weights just as humans do
- That's the hope (and hope is a good thing)
- And in practice indeed these models work better than the vanilla encoder decoder models