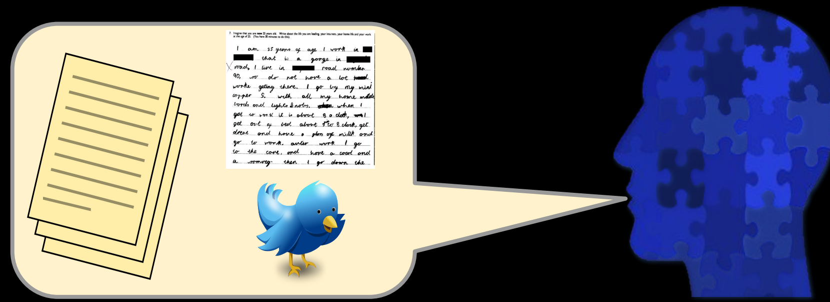
Transformer Sequence Models and Sequence Applications

(Machine Translation, Speech Recognition)

CSE392 - Spring 2019

Special Topic in CS

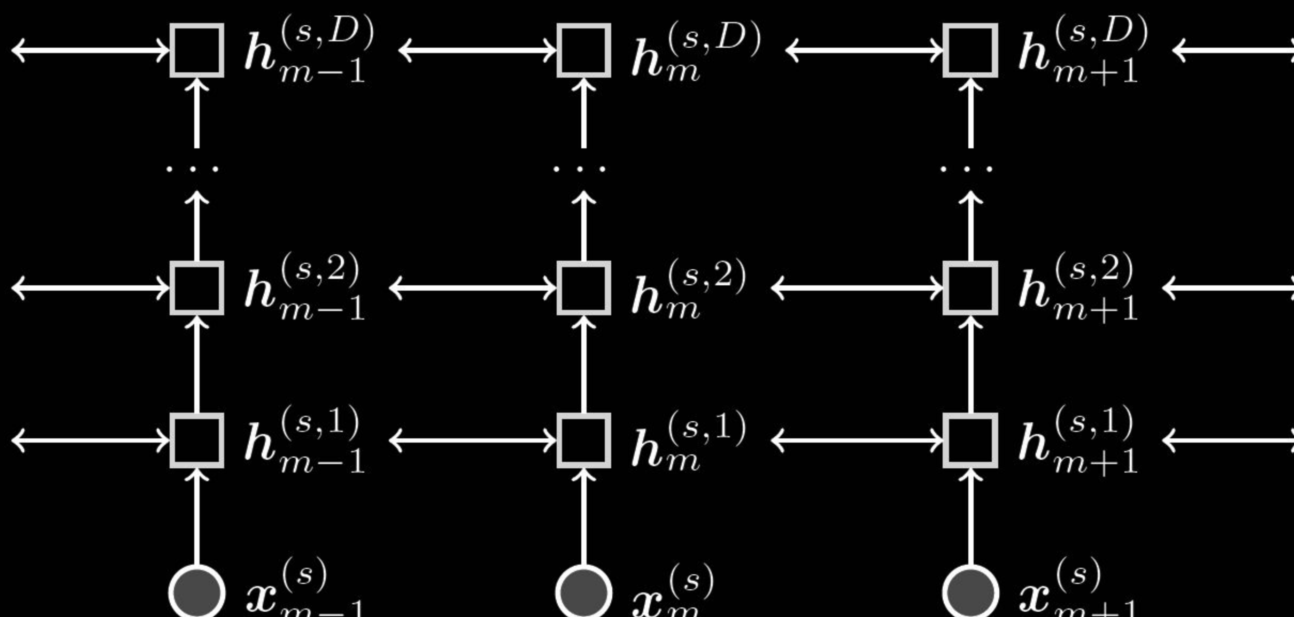


Most NLP Tasks. E.g.

- Sequence Tasks
  - Language Modeling
  - Machine Translation
  - Speech Recognition

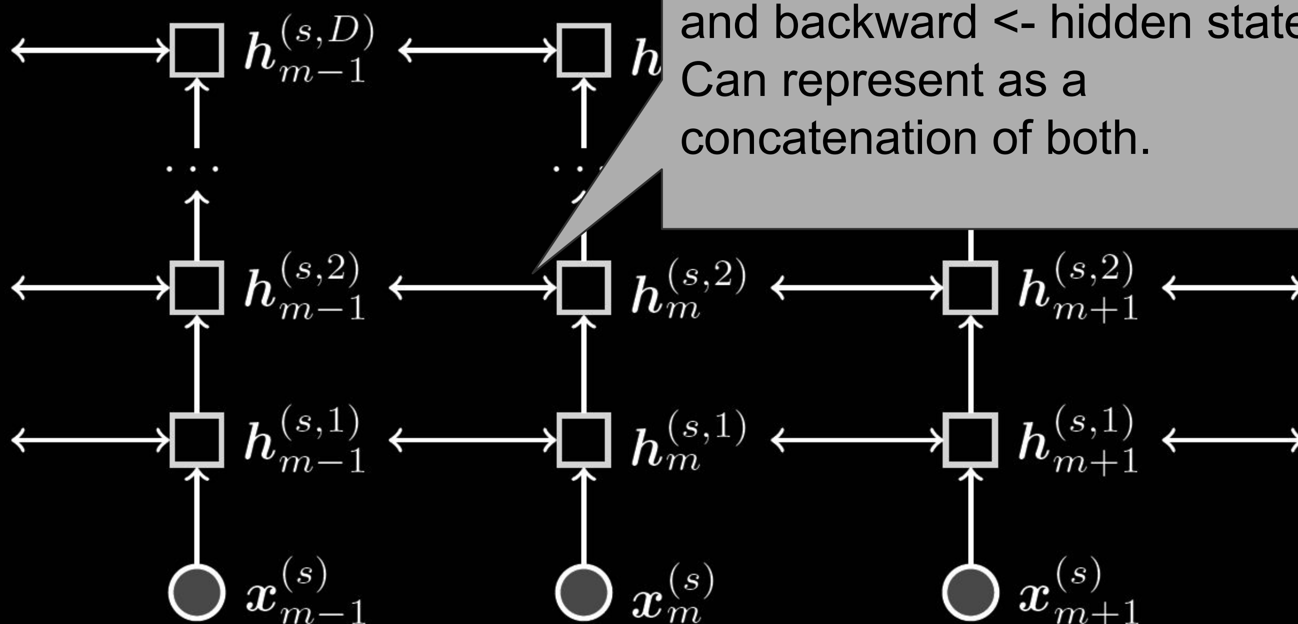
- Transformer Networks
  - Transformers
  - BERT

Multi-level bidirectional RNN (LSTM or GRU)



(Eisenstein, 2018)

Multi-level bidirectional RNN (LSTM or GRU)

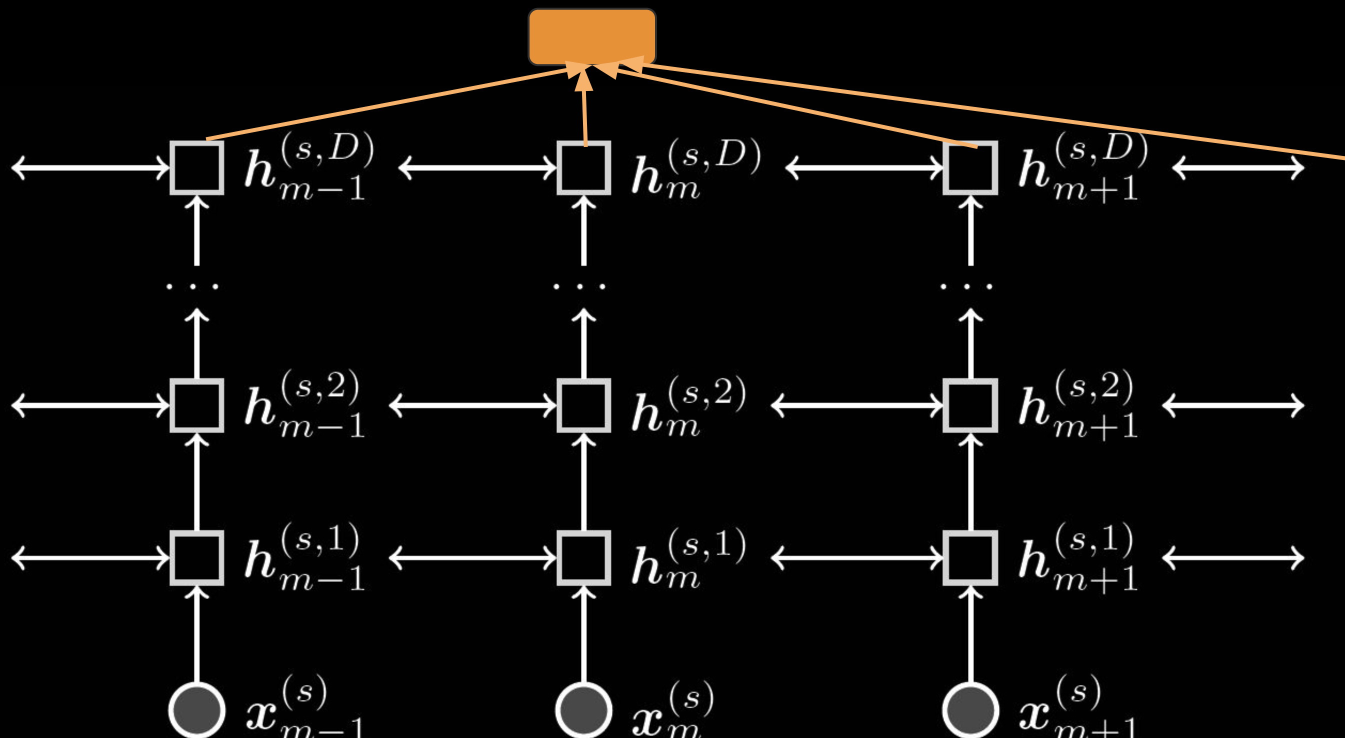


Each node has a forward ->
and backward <- hidden state:
Can represent as a
concatenation of both.

(Eisenstein, 2018)

Multi-level bidirectional RNN (LSTM or GRU)

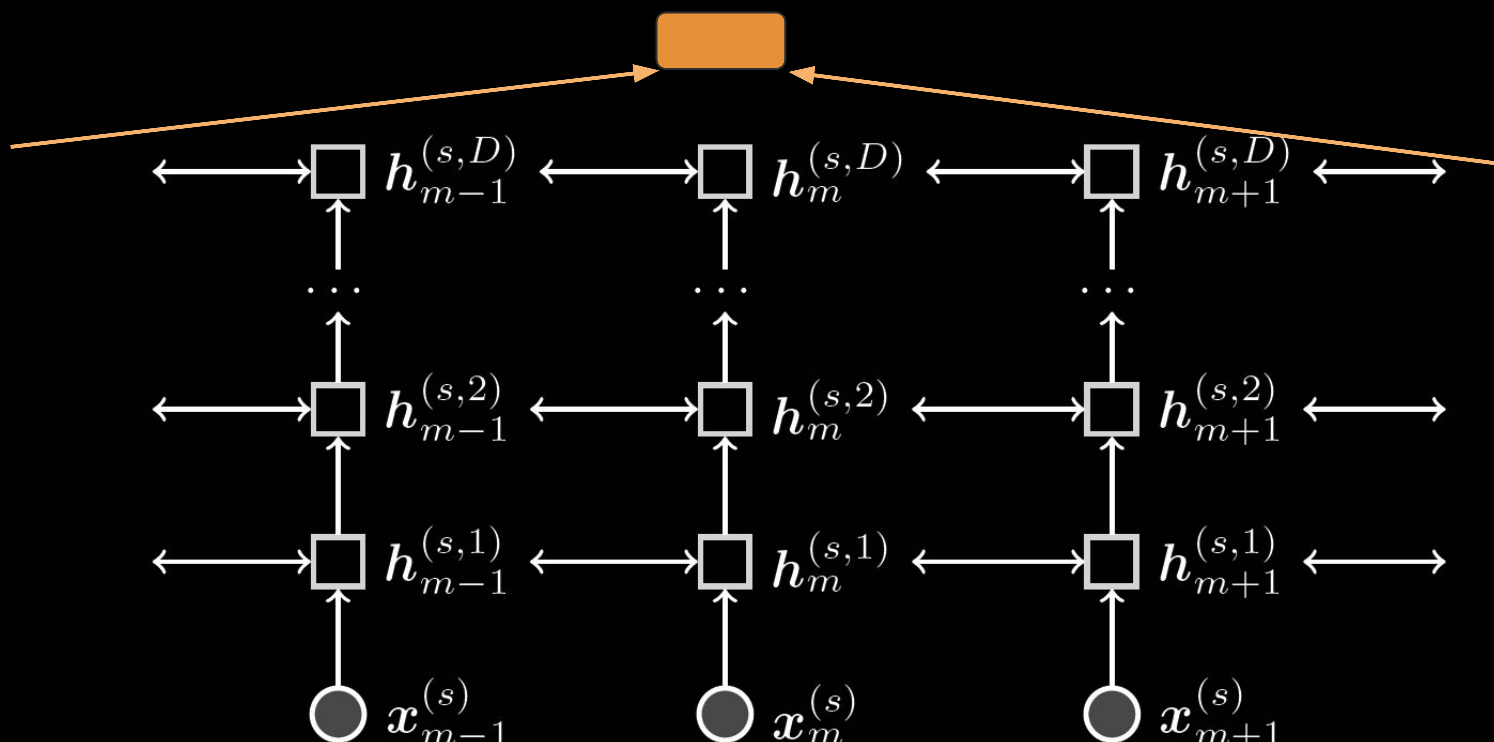
Average of top layer is an embedding (average of concated vectors)



(Eisenstein, 2018)

Multi-level bidirectional RNN (LSTM or GRU)

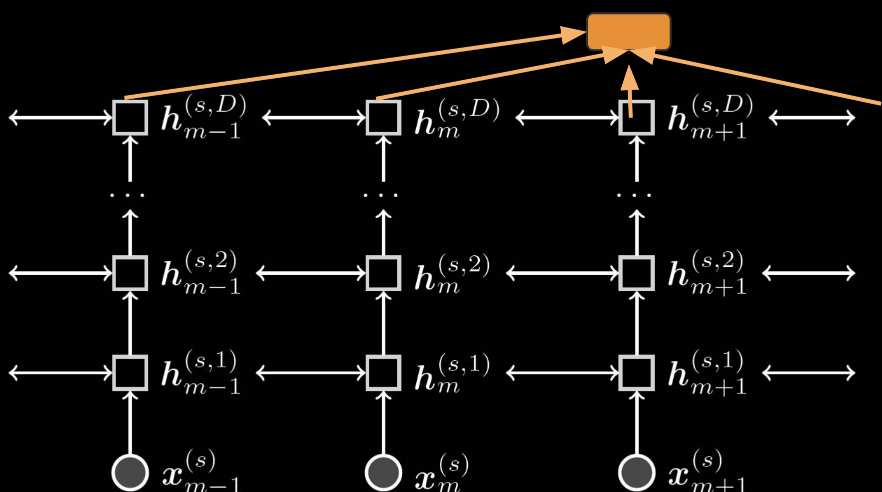
Sometimes just use left-most and right-most hidden state instead



(Eisenstein, 2018)

Encoder

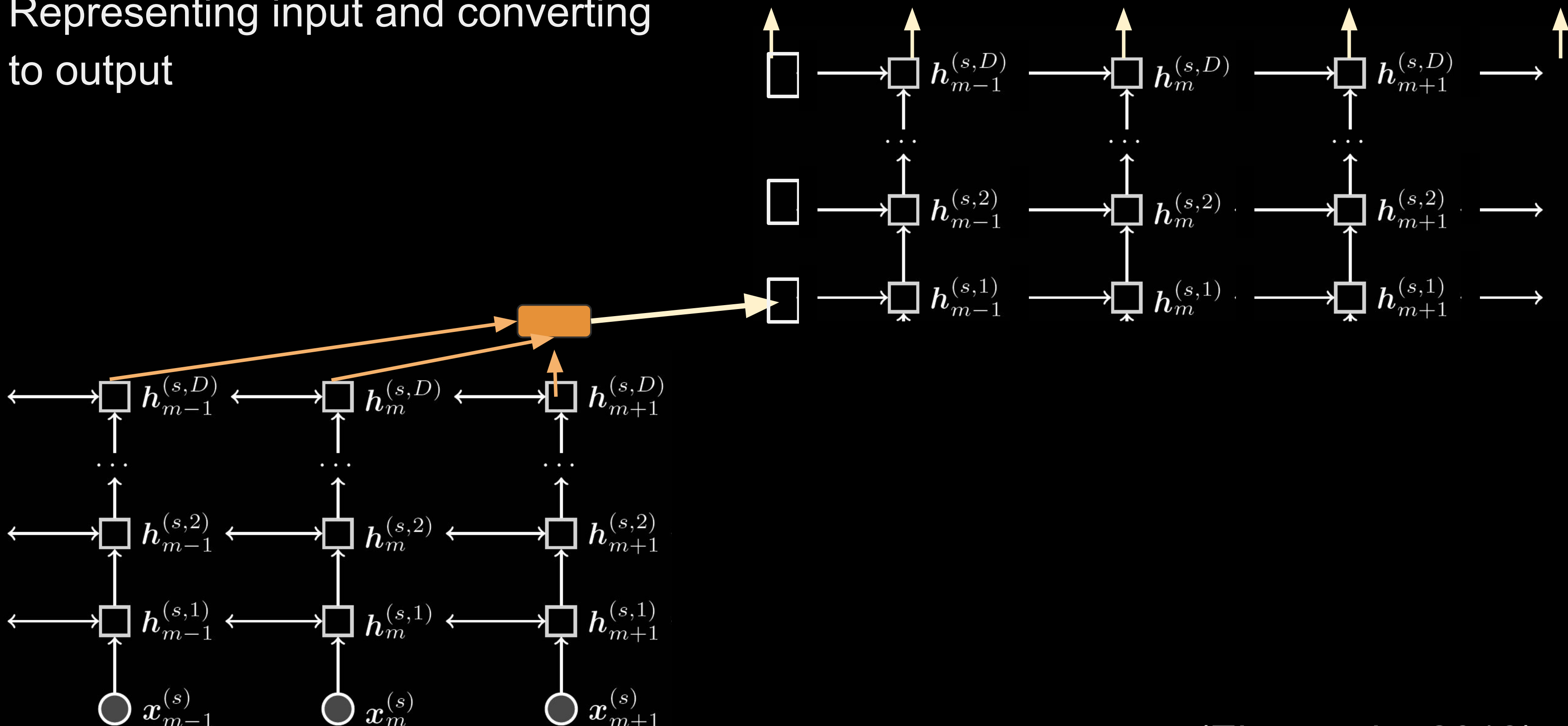
A representation of input.



(Eisenstein, 2018)

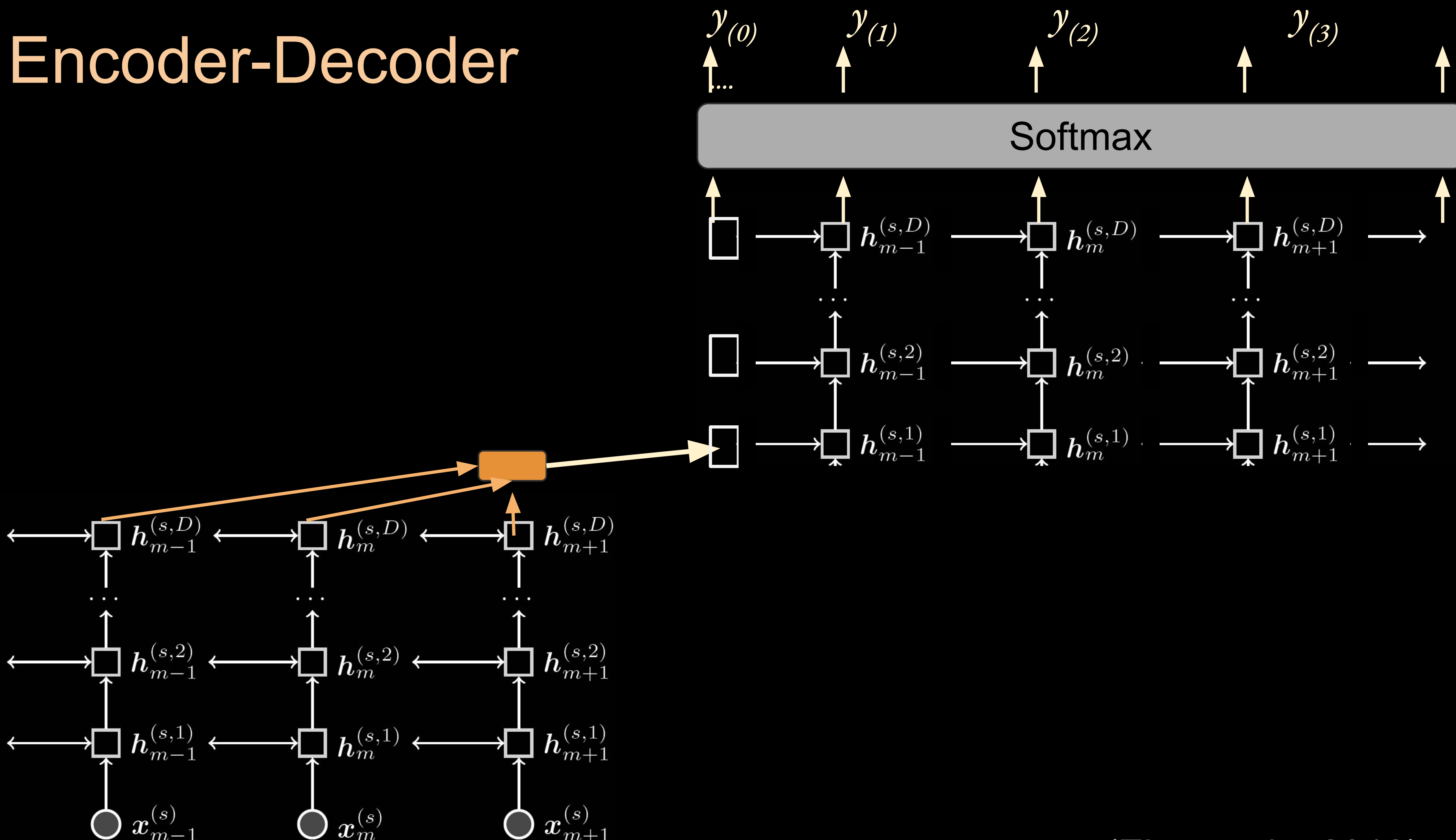
Encoder-Decoder

Representing input and converting to output



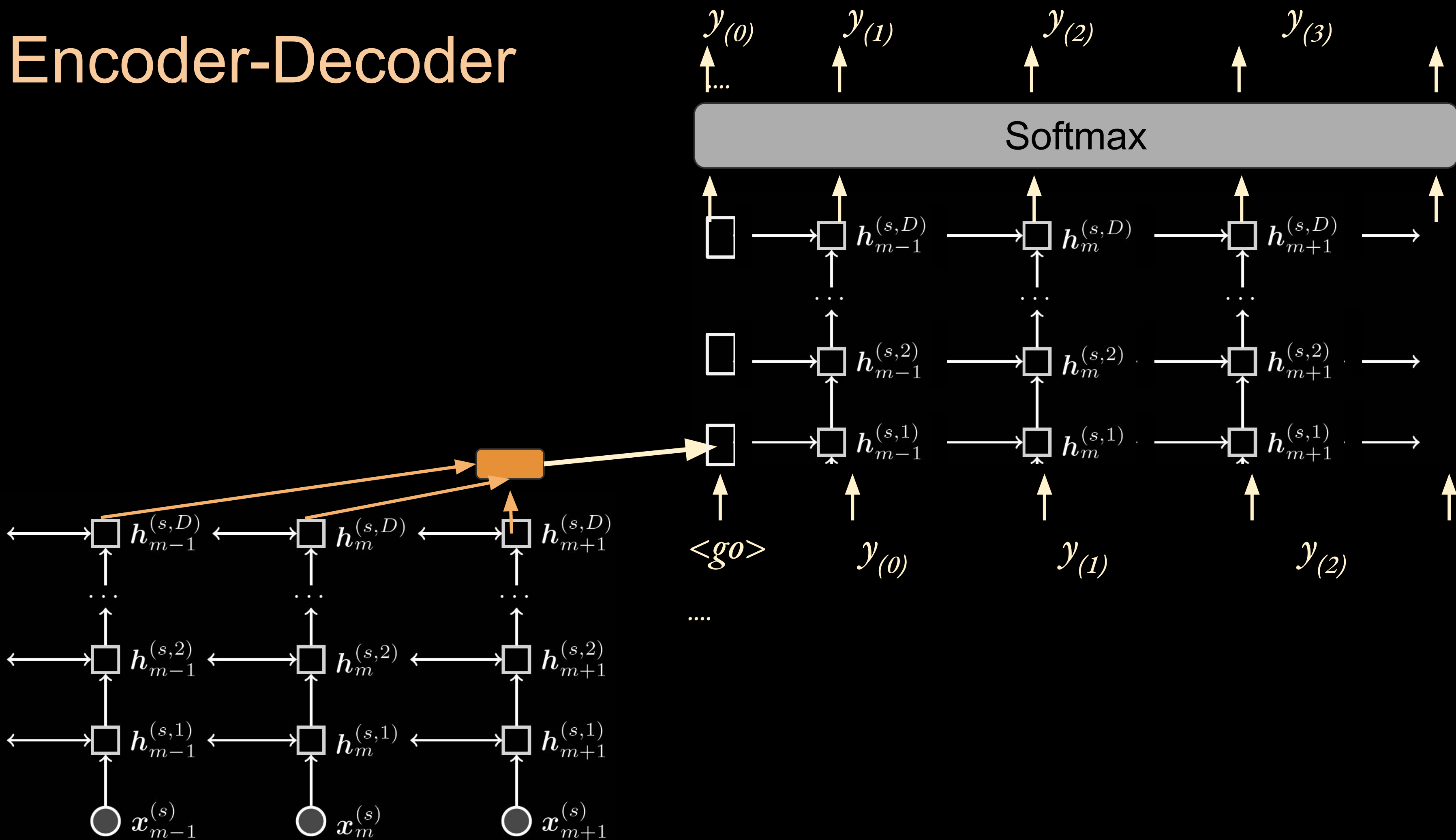
(Eisenstein, 2018)

Encoder-Decoder



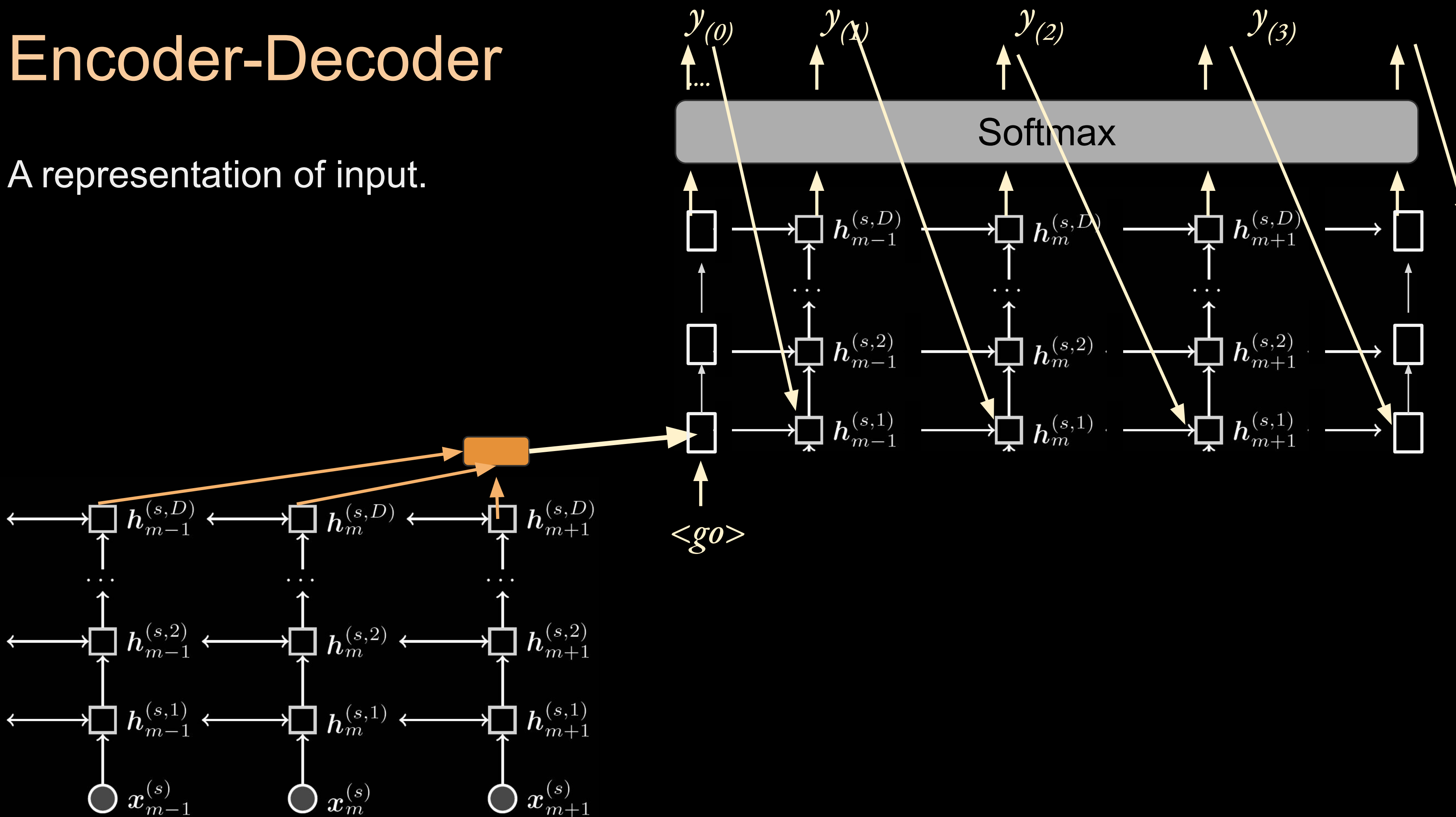
(Eisenstein, 2018)

Encoder-Decoder



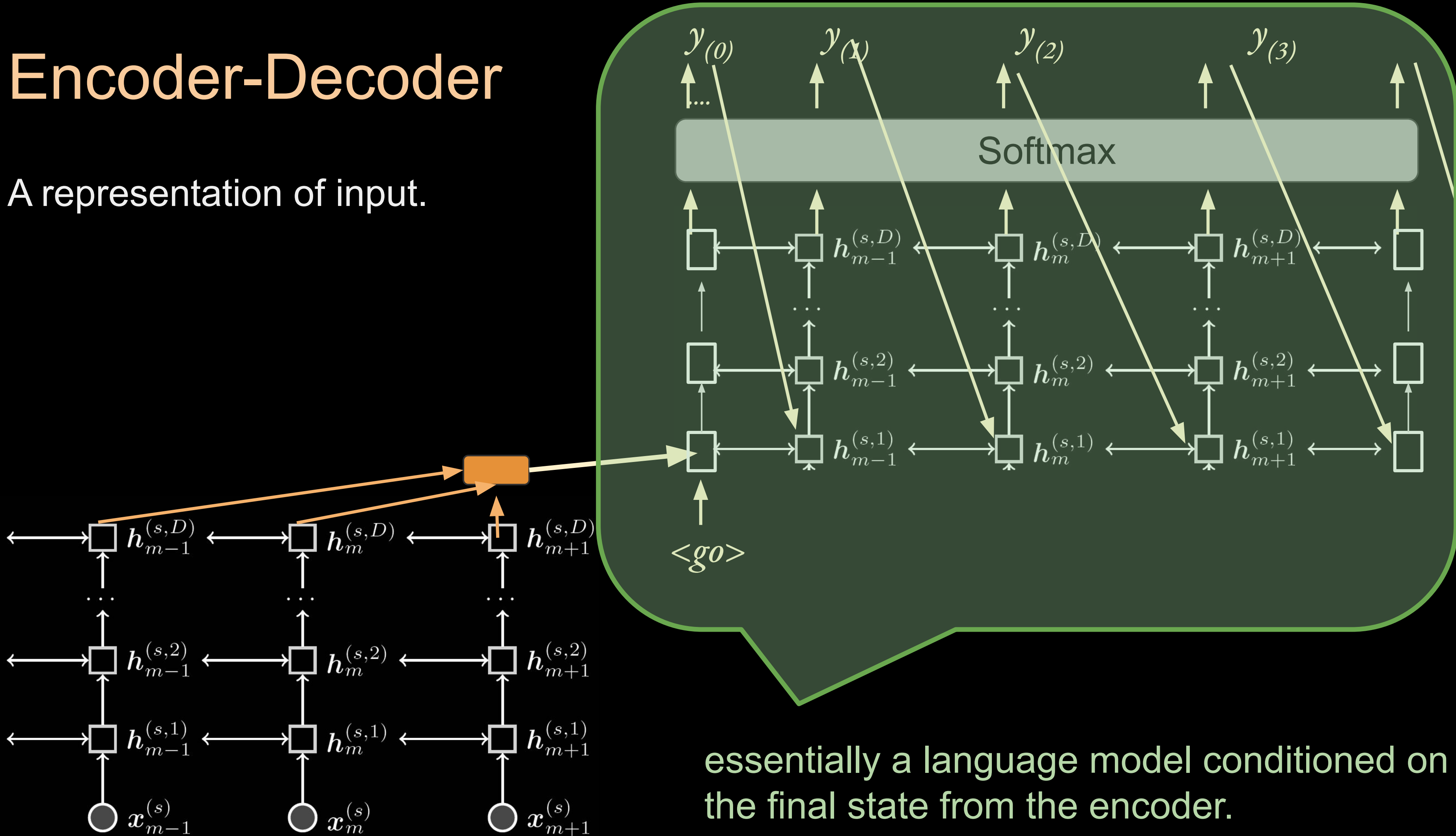
Encoder-Decoder

A representation of input.



Encoder-Decoder

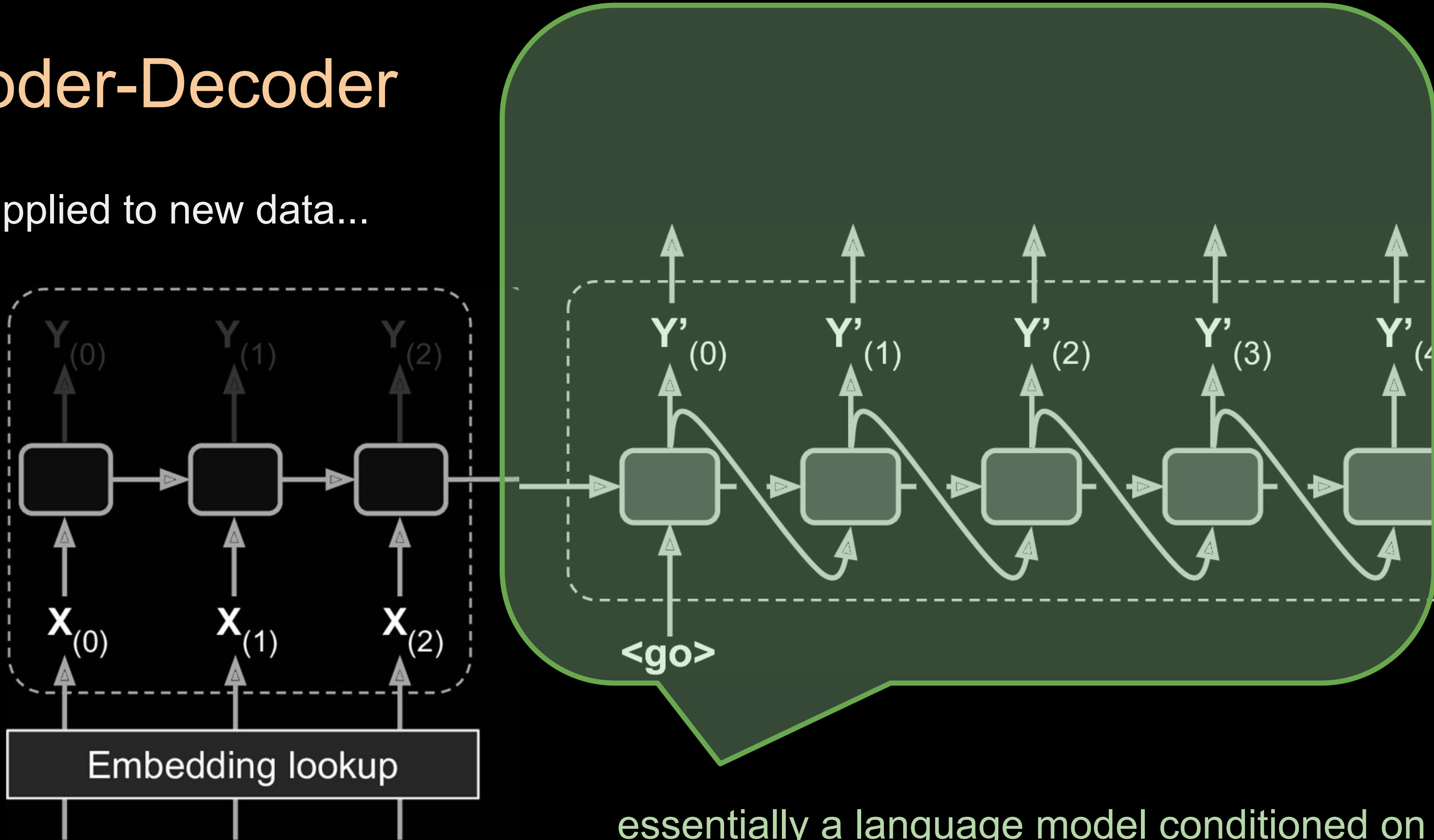
A representation of input.



essentially a language model conditioned on the final state from the encoder.

Encoder-Decoder

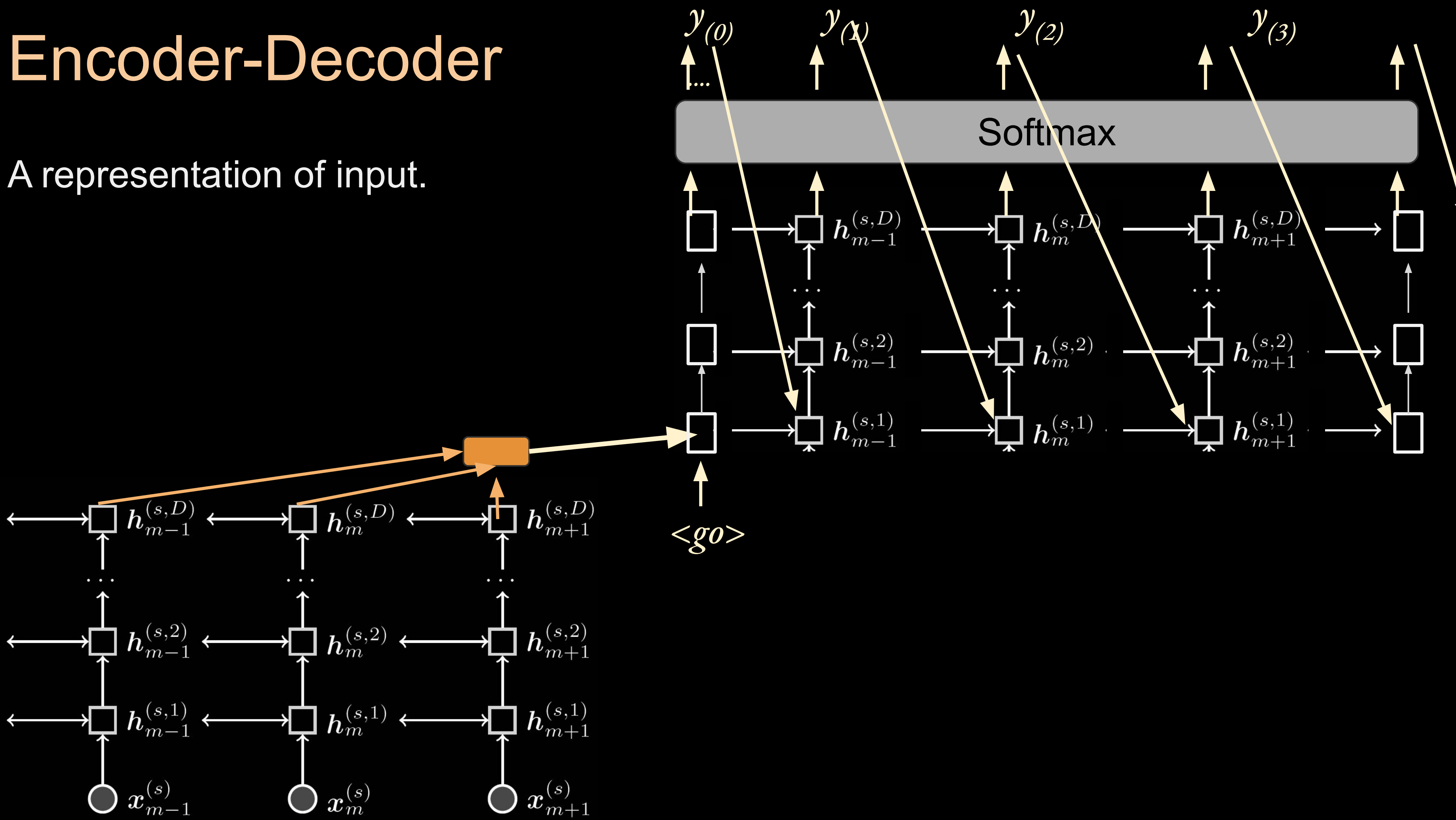
When applied to new data...



essentially a language model conditioned on the final state from the encoder.

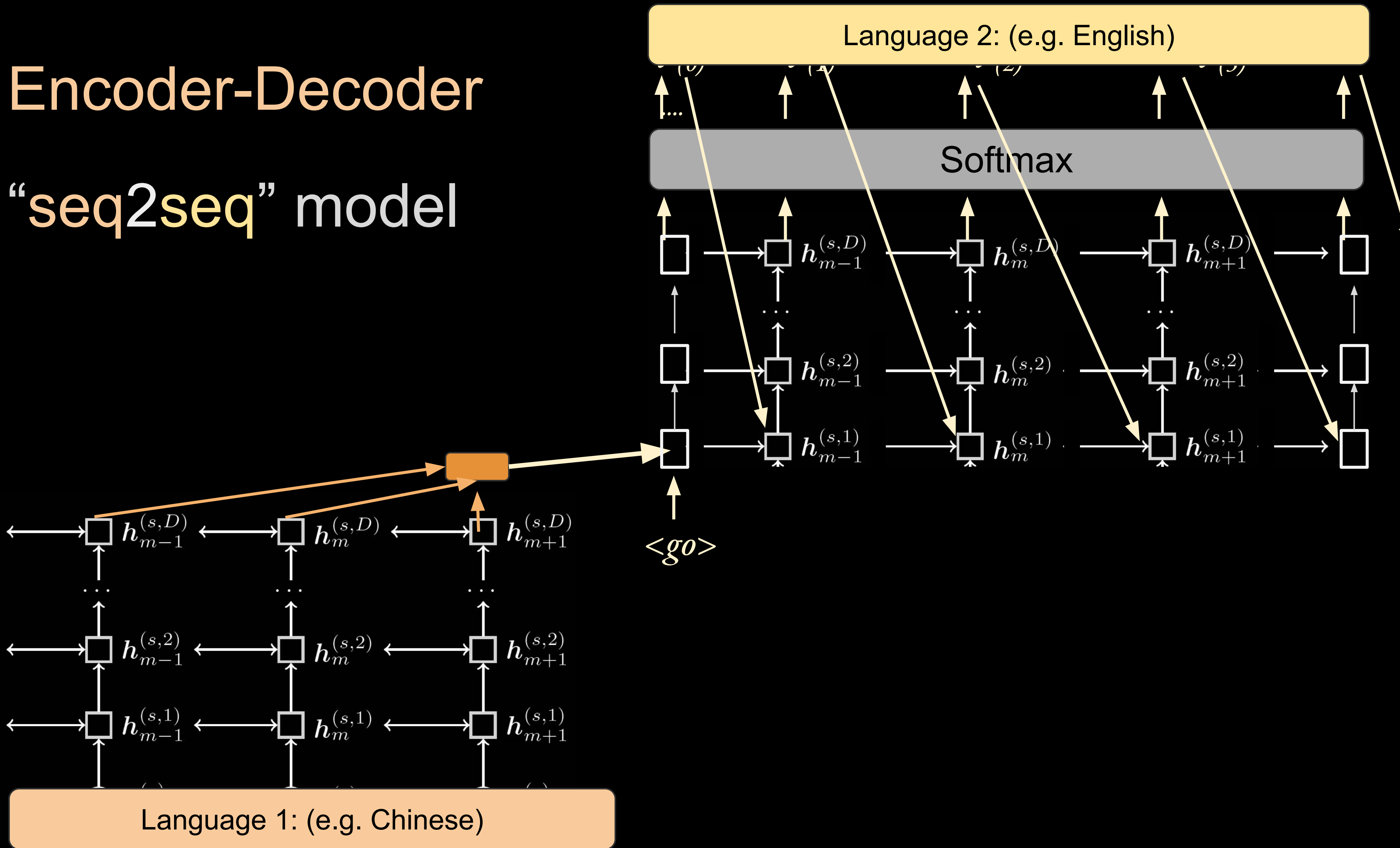
Encoder-Decoder

A representation of input.



Encoder-Decoder

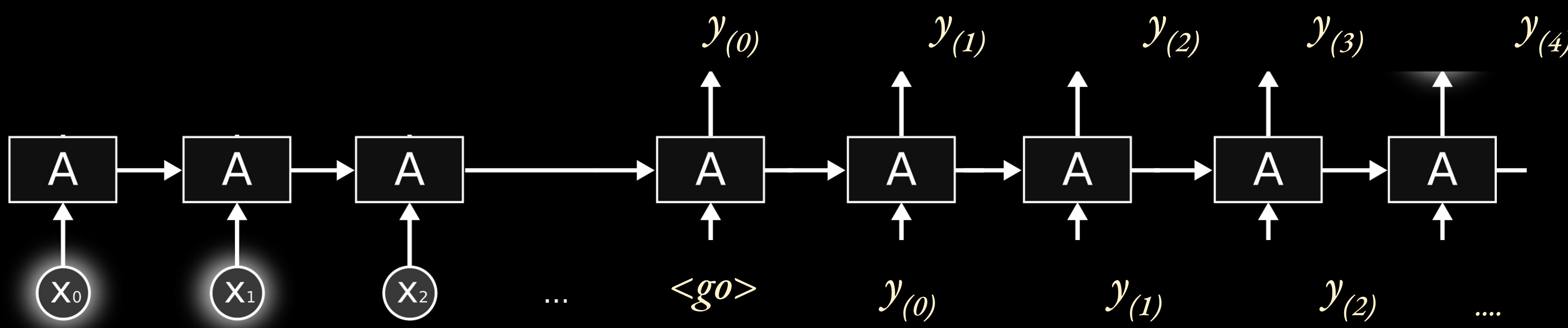
"seq2seq" model



Encoder-Decoder

Challenge:

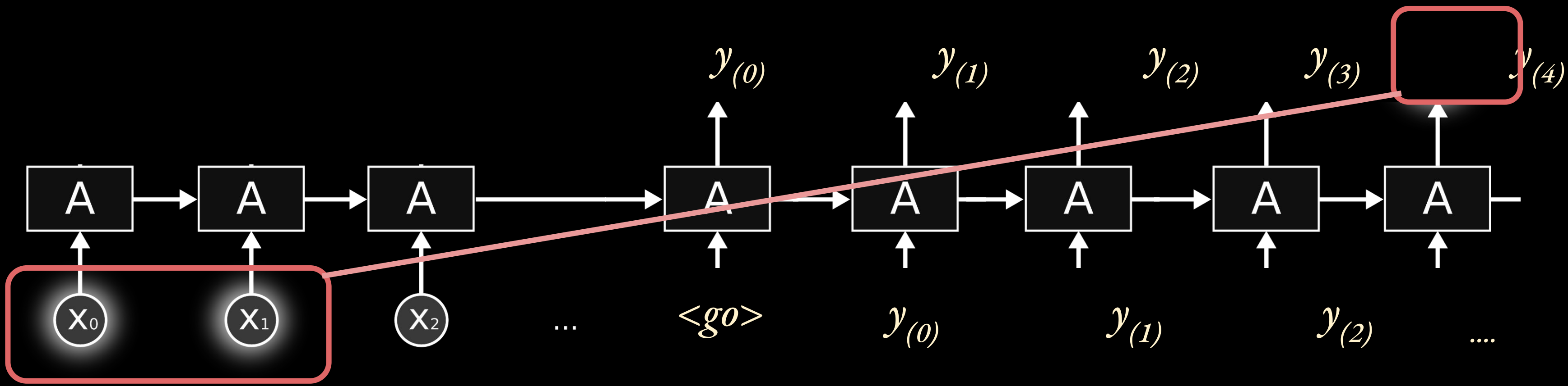
- Long distance dependency when translating:



Encoder-Decoder

Challenge:

- Long distance dependency when translating:

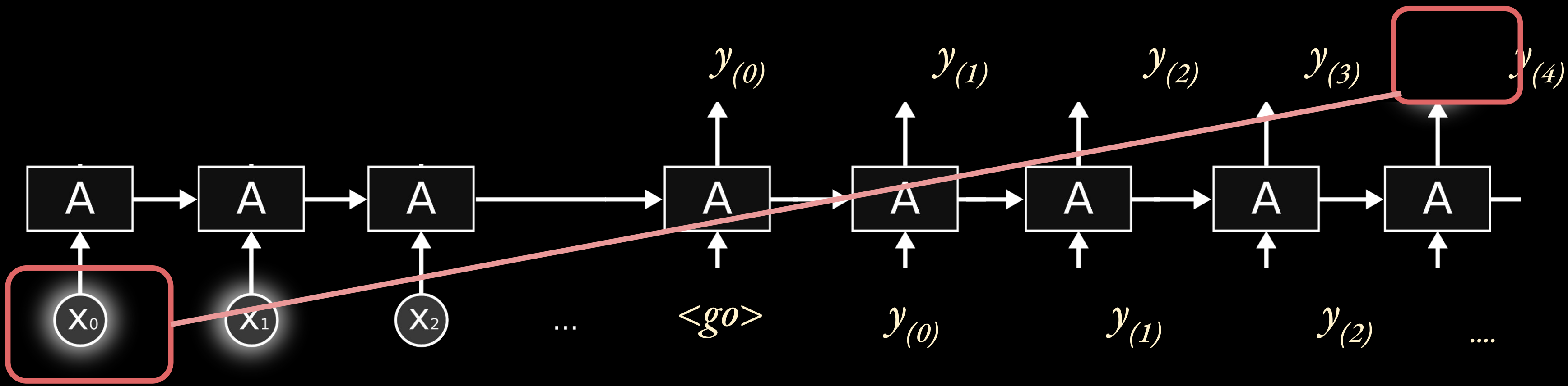


Encoder-Decoder

Challenge:

- Long distance dependency when translating:

The ball was kicked by kayla.



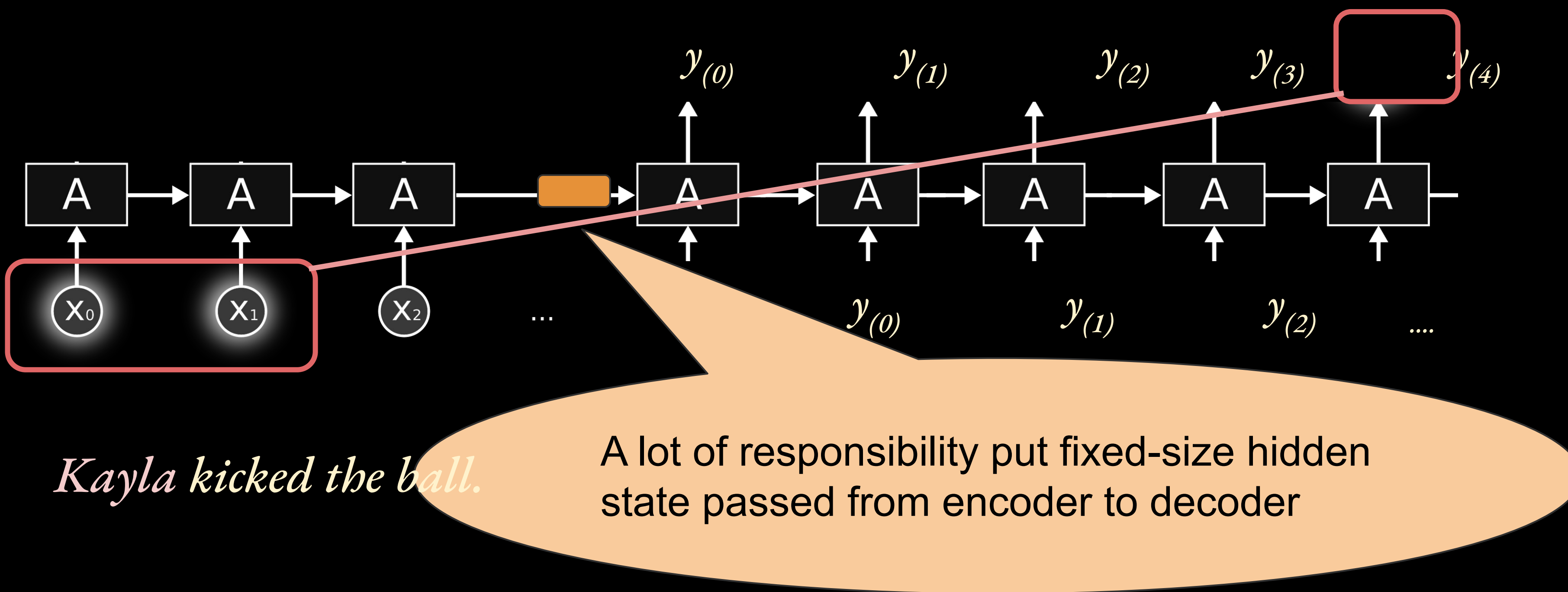
Kayla kicked the ball.

Encoder-Decoder

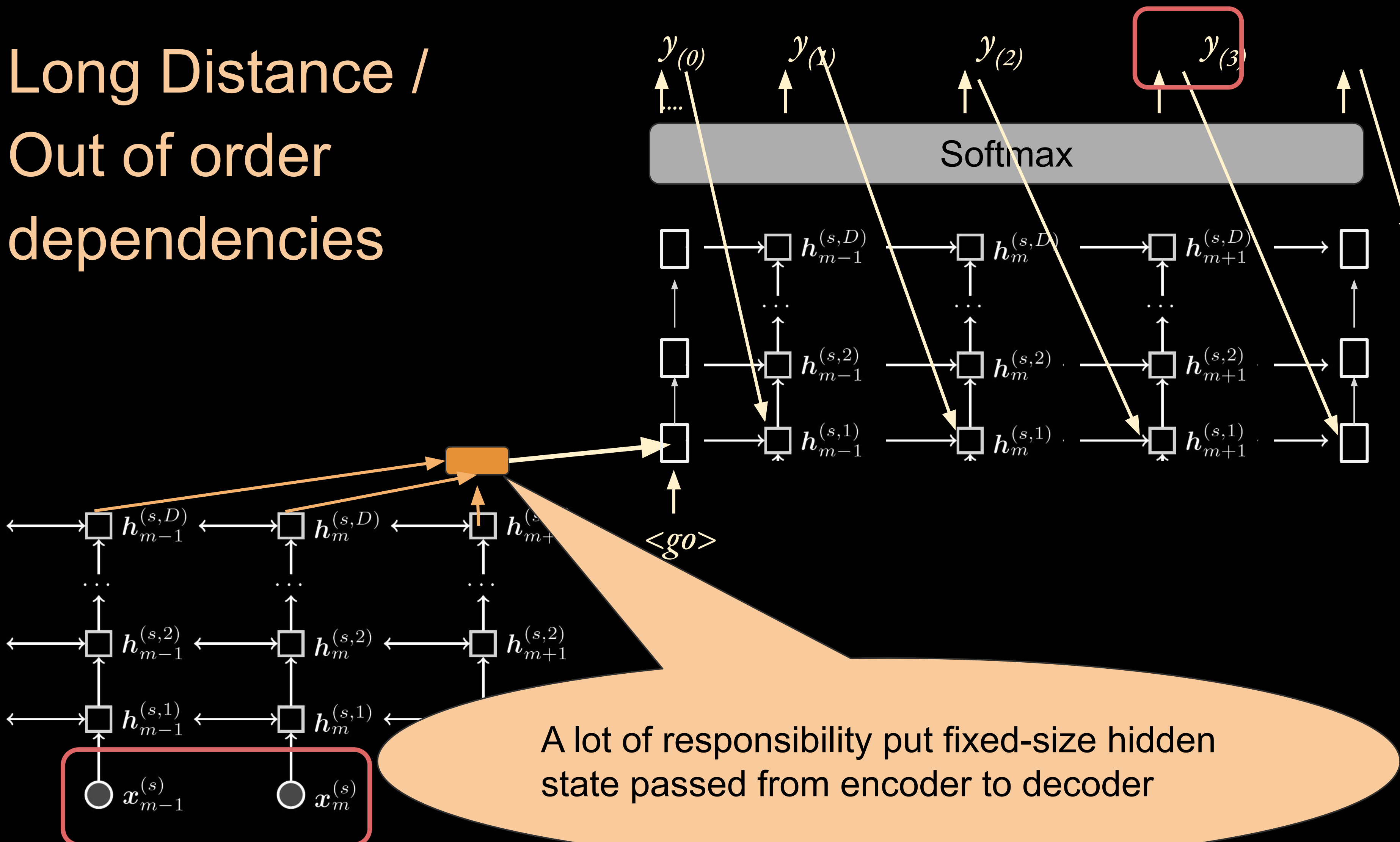
Challenge:

The ball was kicked by kayla.

- Long distance dependency when translating:

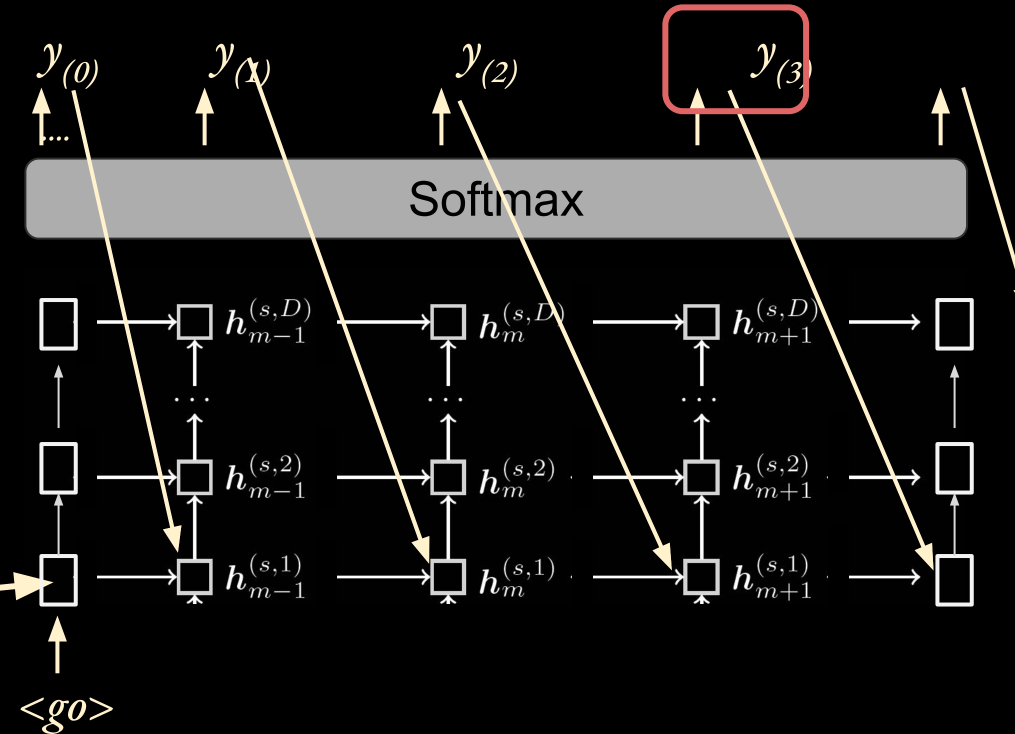
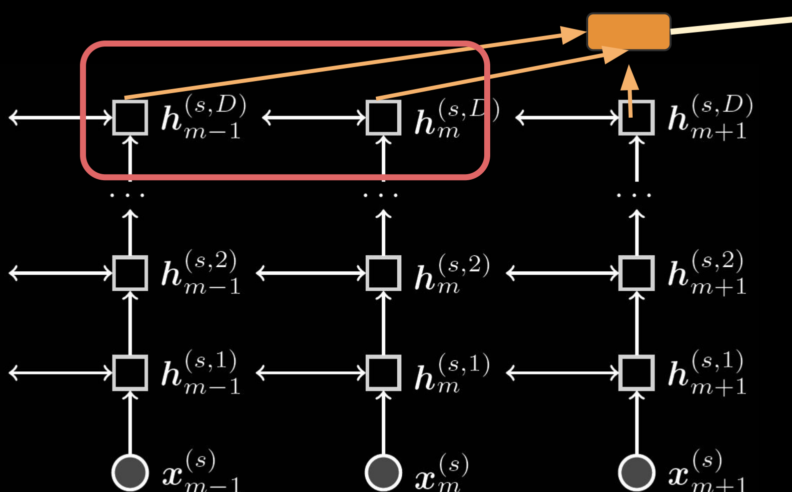


Long Distance / Out of order dependencies

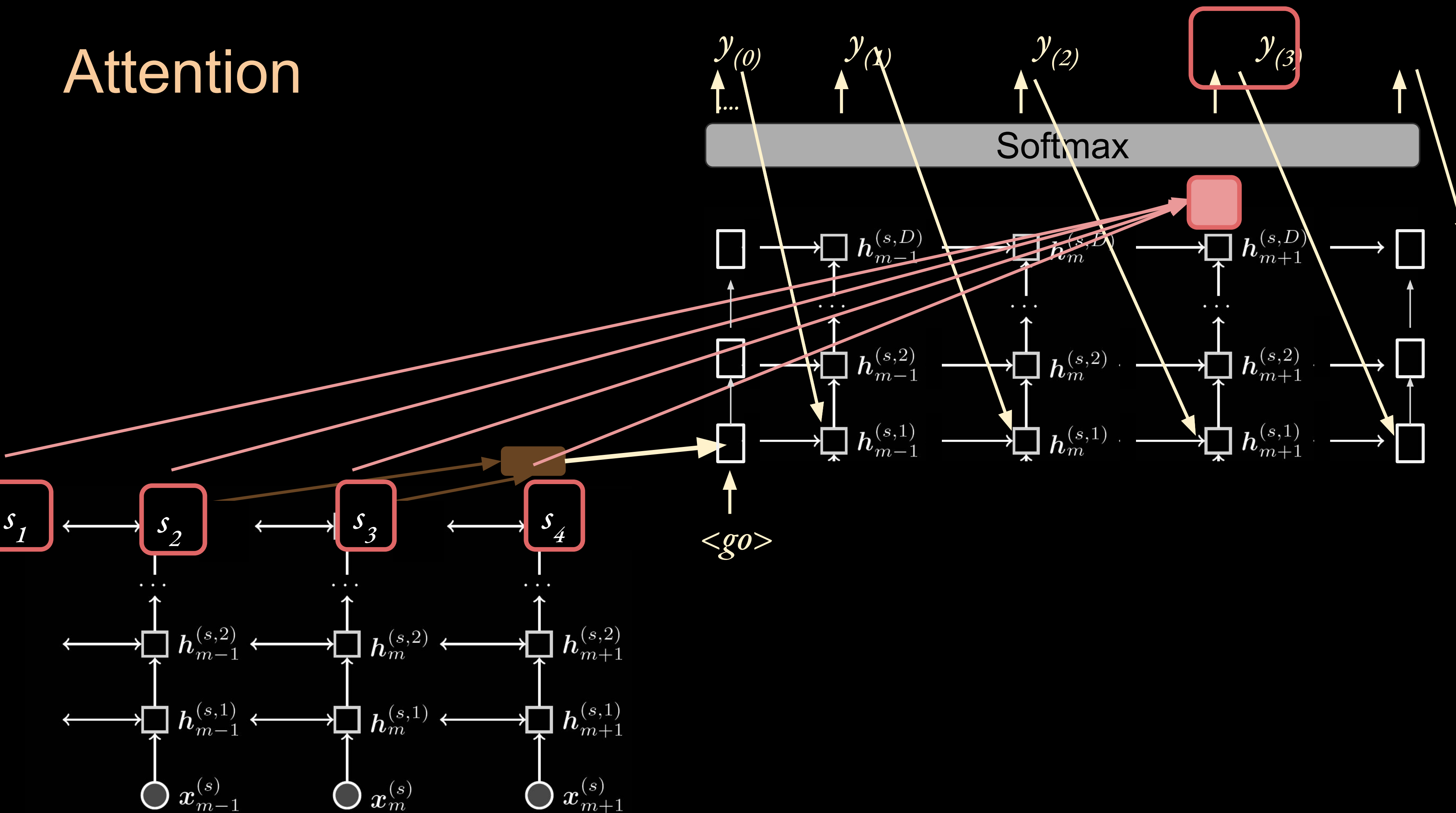


A lot of responsibility put fixed-size hidden state passed from encoder to decoder

Long Distance / Out of order dependencies

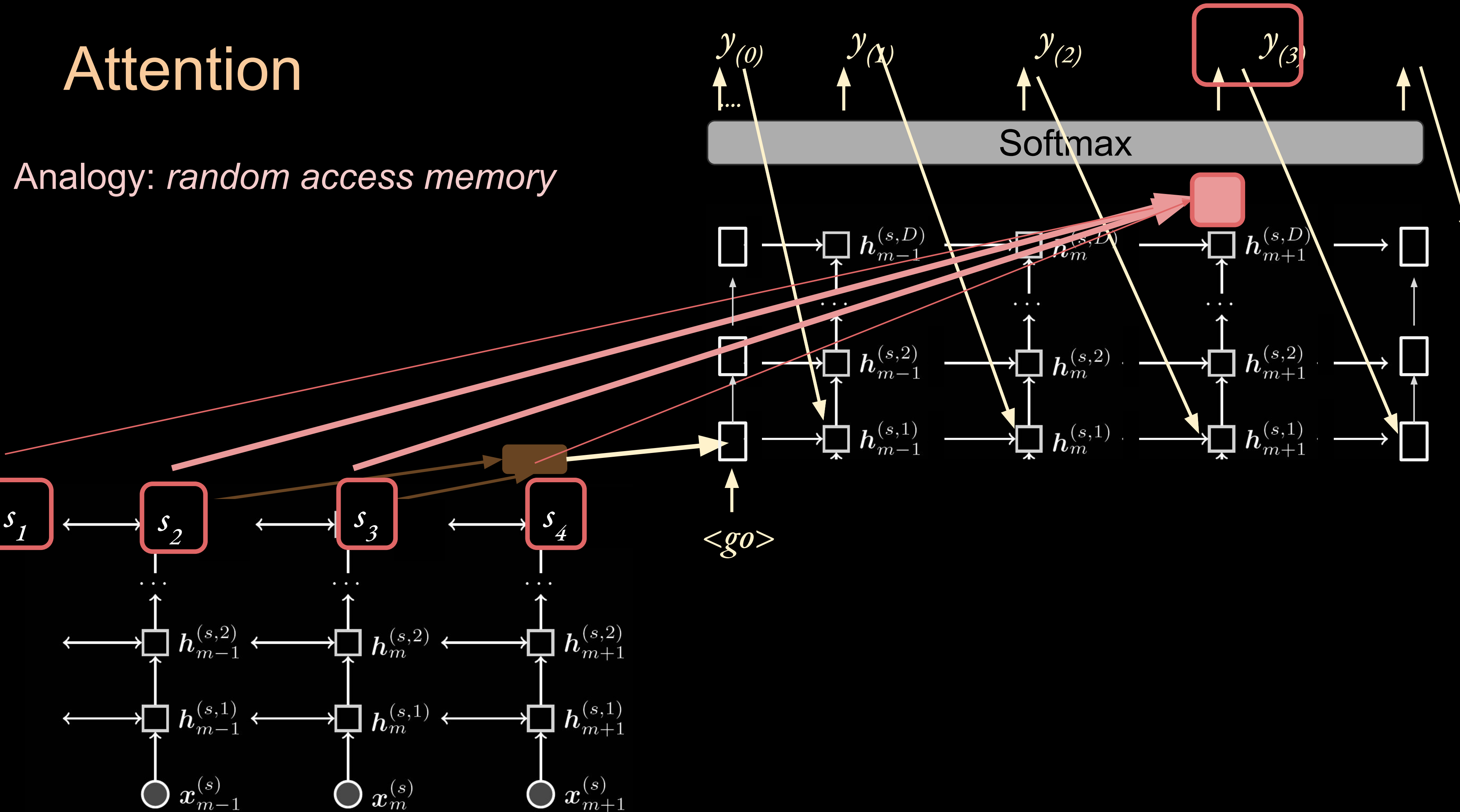
Attention



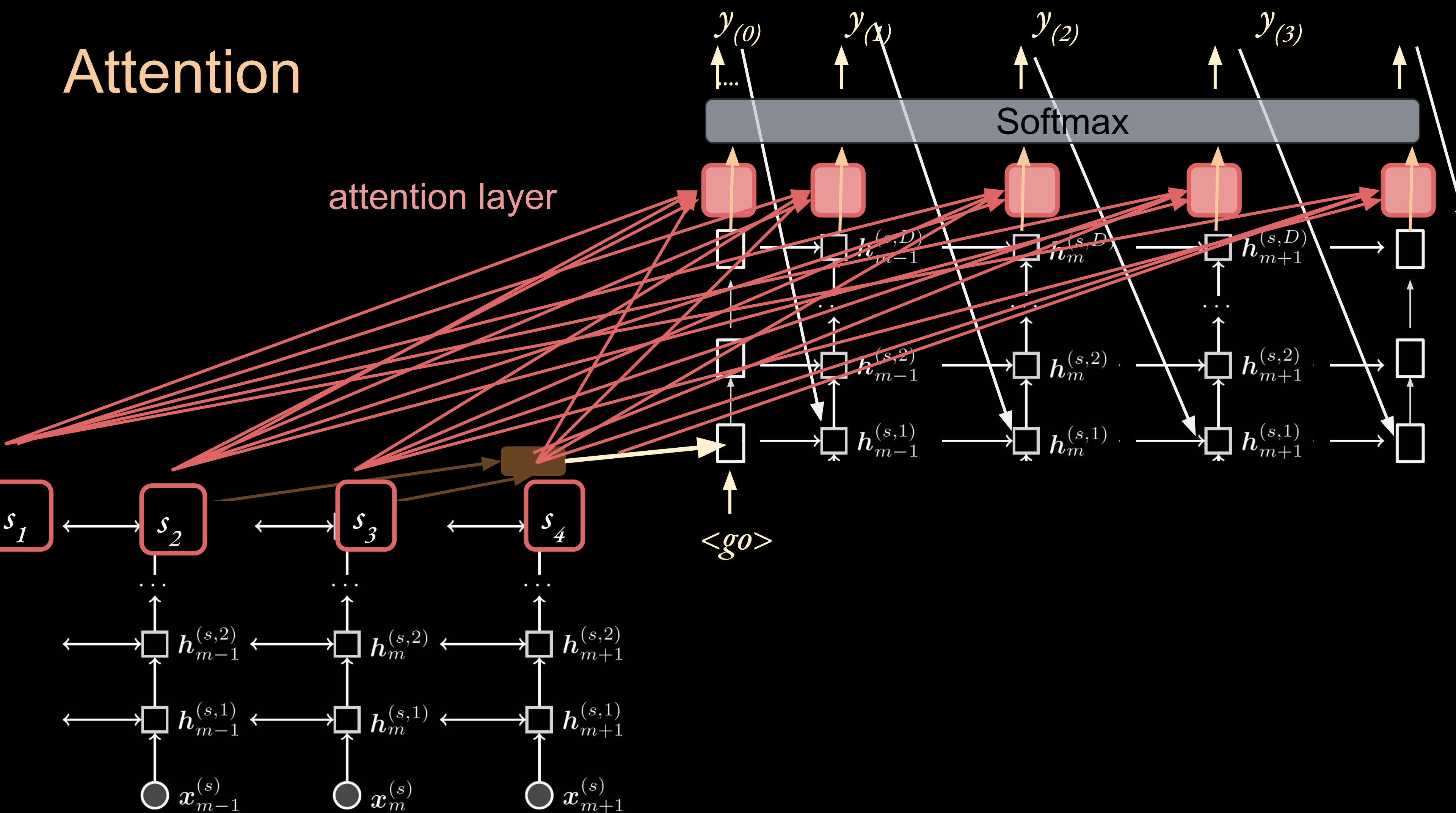
$y_{(0)}$
$y_{(1)}$
$y_{(2)}$
$y_{(3)}$
Softmax
$h_{m-1}^{(s,D)}$
$h_{m}^{(s,D)}$
$h_{m+1}^{(s,D)}$
$h_{m-1}^{(s,2)}$
$h_{m}^{(s,2)}$
$h_{m+1}^{(s,2)}$
$h_{m-1}^{(s,1)}$
$h_{m}^{(s,1)}$
$h_{m+1}^{(s,1)}$
<go>
s_1
s_2
s_3
s_4
$h_{m-1}^{(s,2)}$
$h_{m}^{(s,2)}$
$h_{m+1}^{(s,2)}$
$h_{m-1}^{(s,1)}$
$h_{m}^{(s,1)}$
$h_{m+1}^{(s,1)}$
$x_{m-1}^{(s)}$
$x_{m}^{(s)}$
$x_{m+1}^{(s)}$

Attention

Analogy: *random access memory*

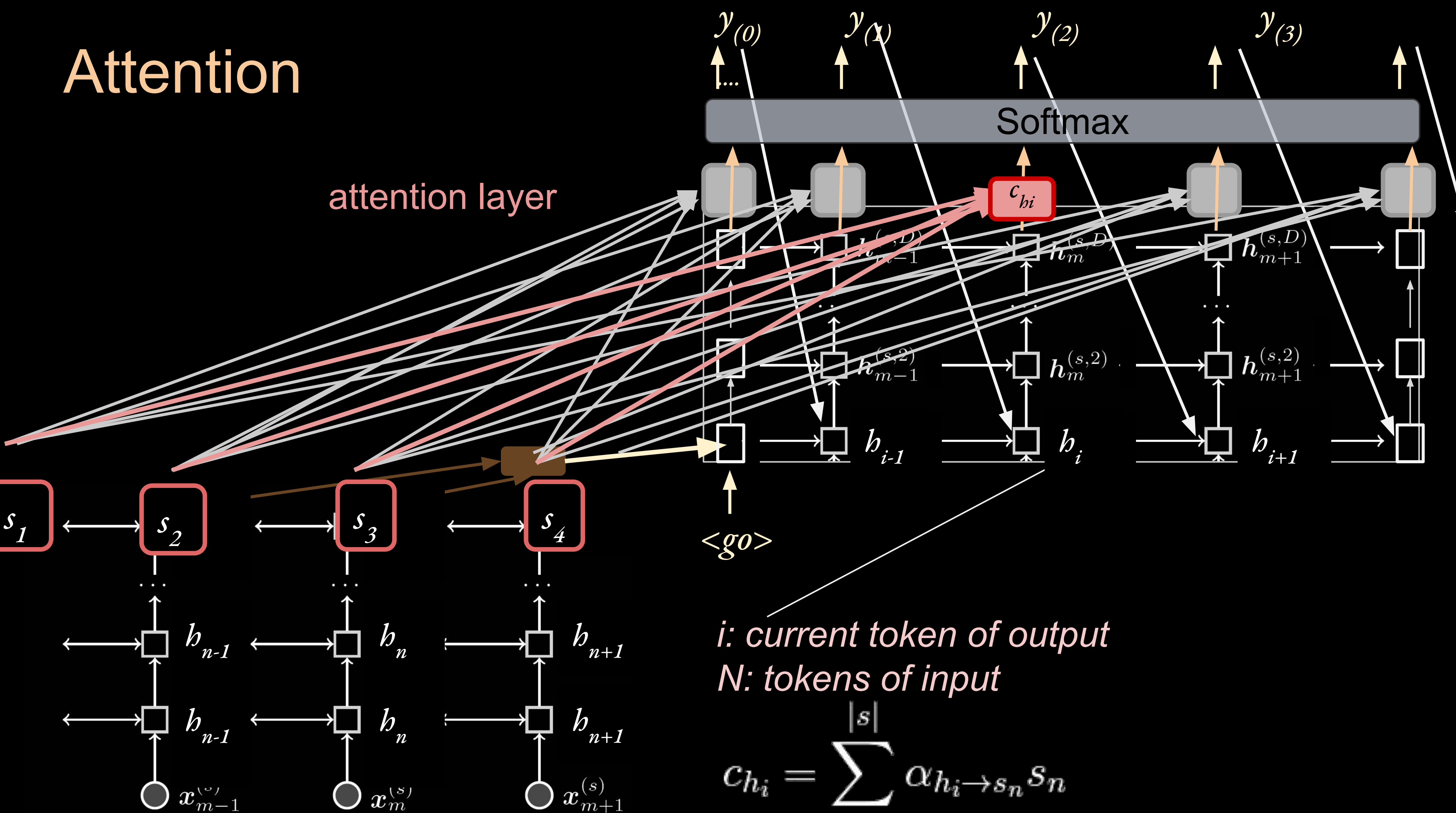


Attention



$y_{(0)}$
$y_{(1)}$
$y_{(2)}$
$y_{(3)}$
Softmax
attention layer
$h_{m-1}^{(s,D)}$
$h_{m}^{(s,D)}$
$h_{m+1}^{(s,D)}$
$h_{m-1}^{(s,2)}$
$h_{m}^{(s,2)}$
$h_{m+1}^{(s,2)}$
$h_{m-1}^{(s,1)}$
$h_{m}^{(s,1)}$
$h_{m+1}^{(s,1)}$
<go>
s_1
s_2
s_3
s_4
$x_{m-1}^{(s)}$
$x_{m}^{(s)}$
$x_{m+1}^{(s)}$

Attention

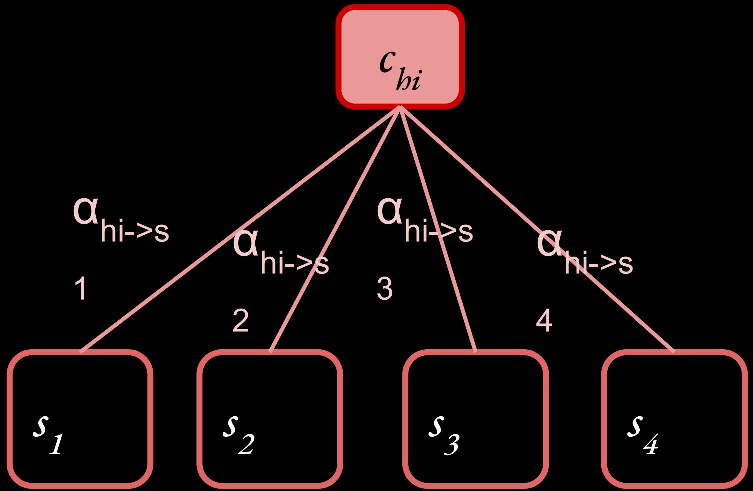


i: current token of output

N: tokens of input

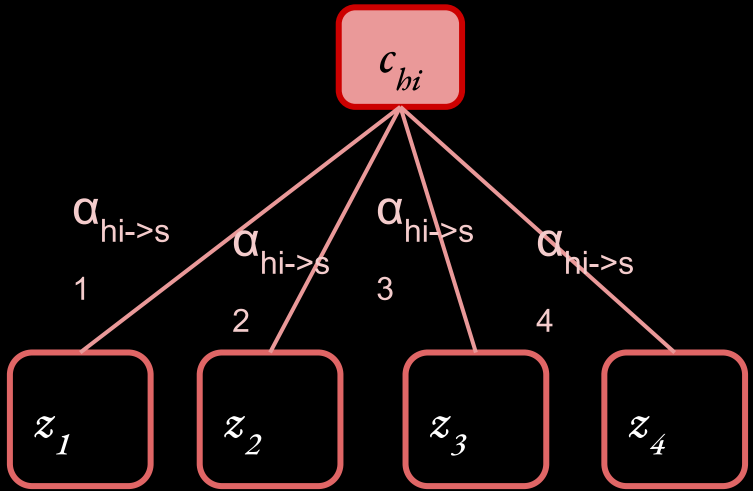
$$c_{h_i} = \sum_{n=1}^{|s|} \alpha_{h_i \to s_n} s_n$$

Attention



$$c_{h_i} = \sum_{n=1}^{|s|} \alpha_{h_i \rightarrow s_n} s_n$$

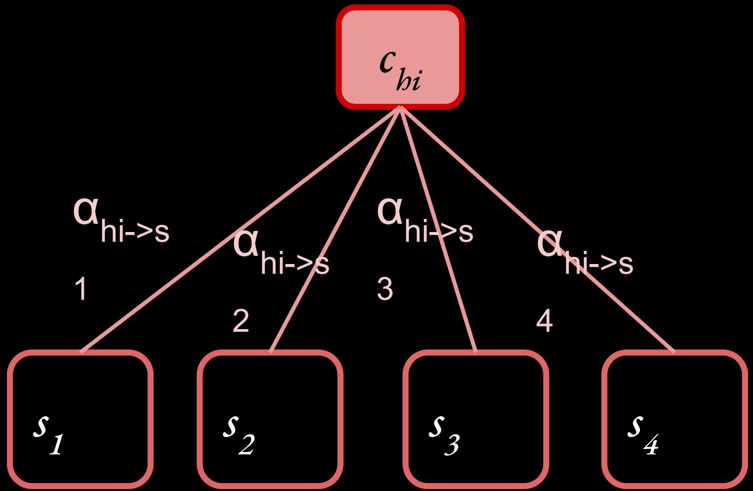
Attention



Z is the vector to be attended to (the value in memory). It is typically hidden states of the input (i.e. s_n) but can be anything.

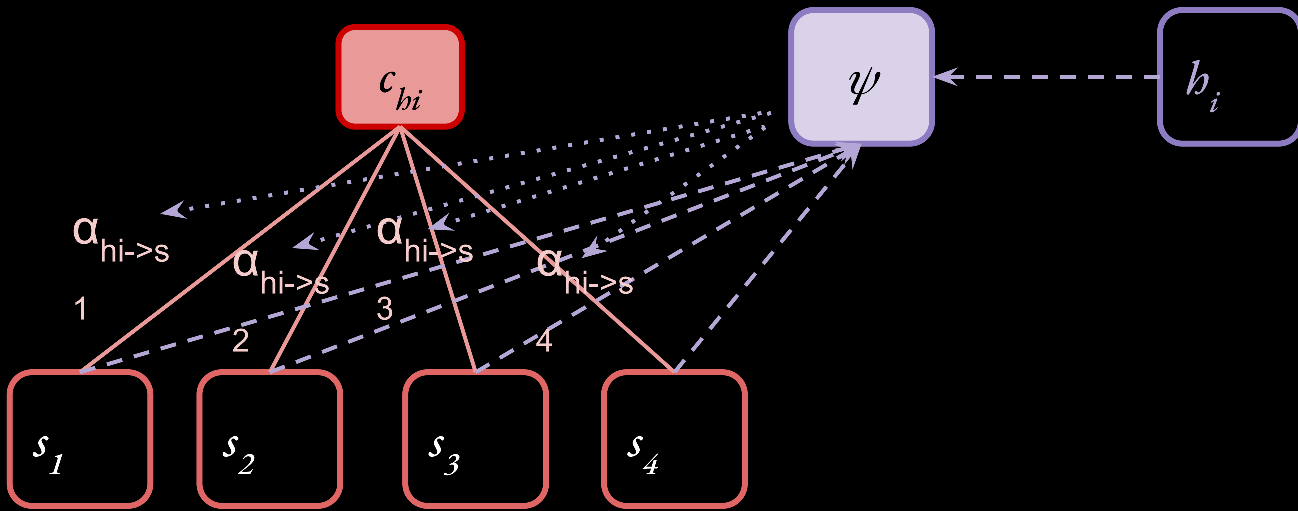
$$c_{h_i} = \sum_{n=1}^{|s|} \alpha_{h_i \to s_n} z_n$$

Attention



$$c_{h_i} = \sum_{n=1}^{|s|} \alpha_{h_i \to s_n} s_n$$

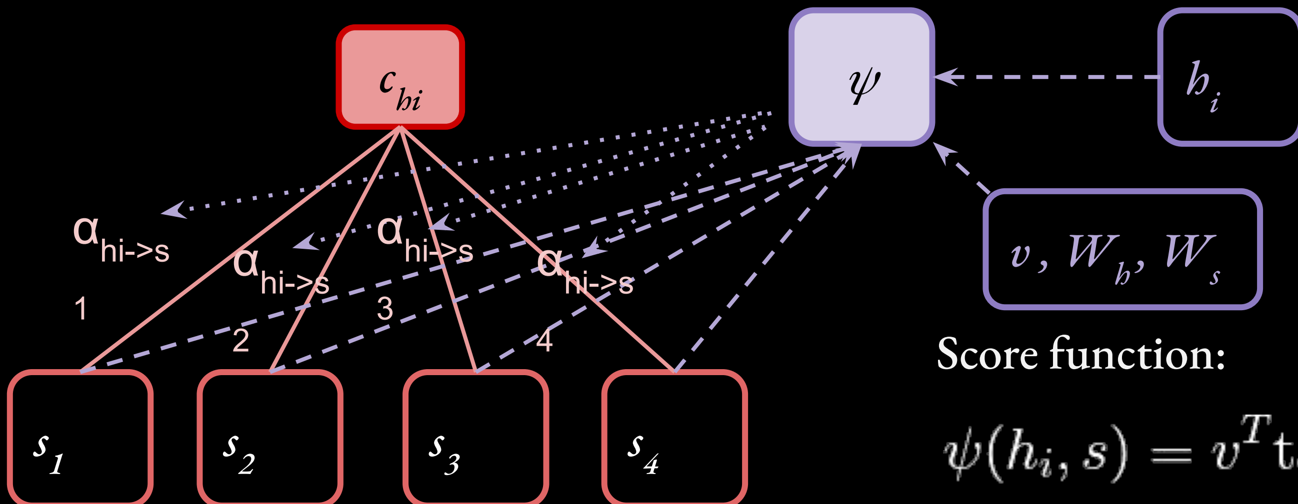
Attention



$$\alpha_{h_i \to s} = \mathrm{softmax}(\psi(h_i, s))$$

$$c_{h_i} = \sum_{n=1}^{|s|} \alpha_{h_i \to s_n} s_n$$

Attention



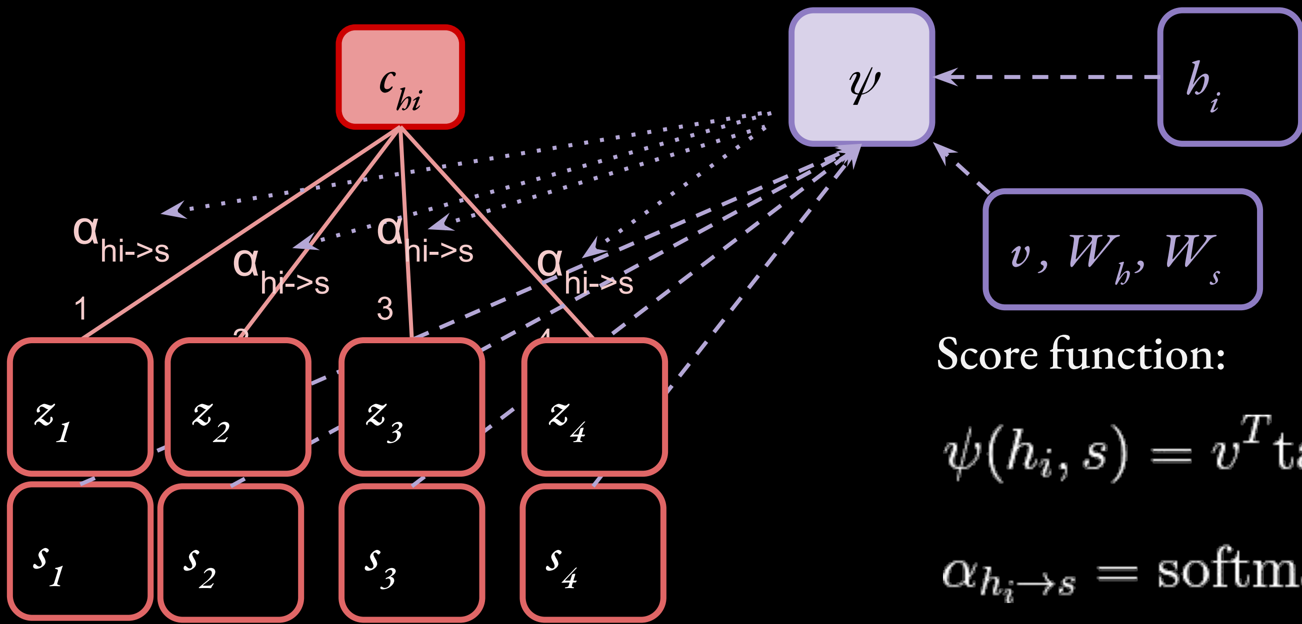
Score function:

$$\psi(h_i, s) = v^T \tanh(W_h h_i + W_s s])$$

$$\alpha_{h_i \to s} = \mathrm{softmax}(\psi(h_i, s))$$

$$c_{h_i} = \sum_{n=1}^{|s|} \alpha_{h_i \to s_n} s_n$$

Attention



Score function:

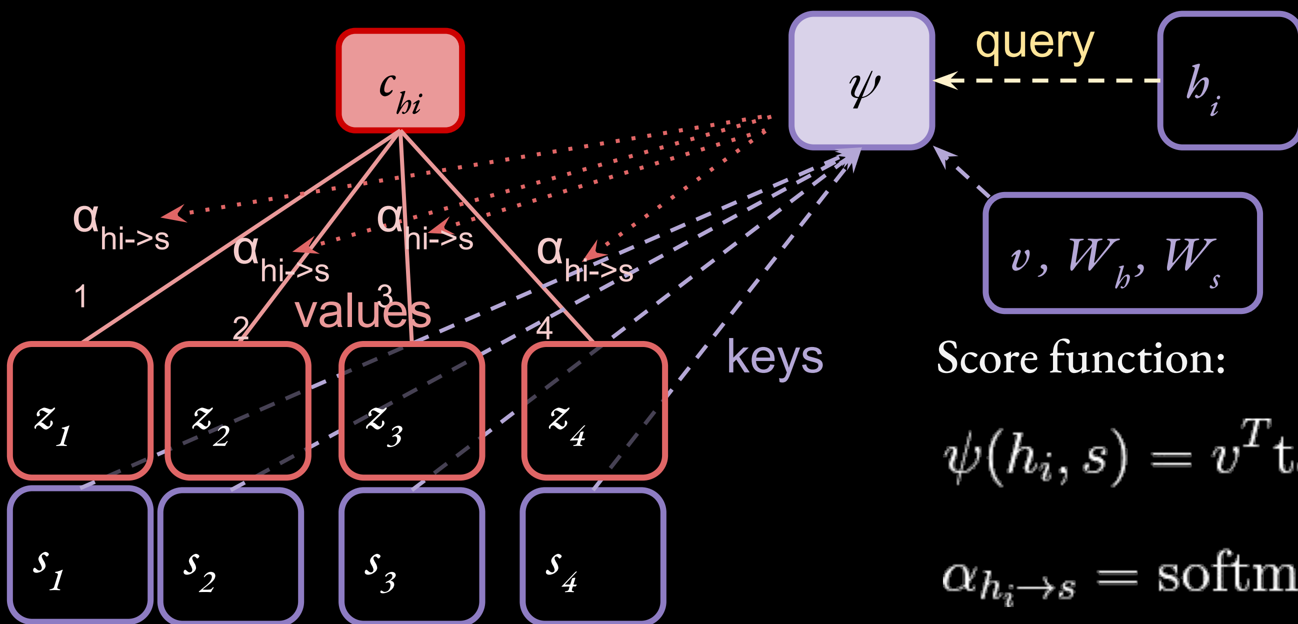
$$\psi(h_i, s) = v^T \tanh(W_h h_i + W_s s])$$

$$\alpha_{h_i \to s} = \mathrm{softmax}(\psi(h_i, s))$$

$$c_{h_i} = \sum_{n=1}^{|s|} \alpha_{h_i \to s_n} z_n$$

A useful abstraction is to make the vector attended to (the "value vector", *Z*) separate than the "key vector" (s).

Attention



A useful abstraction is to make the vector attended to (the "value vector", *Z*) separate than the "key vector" (s).

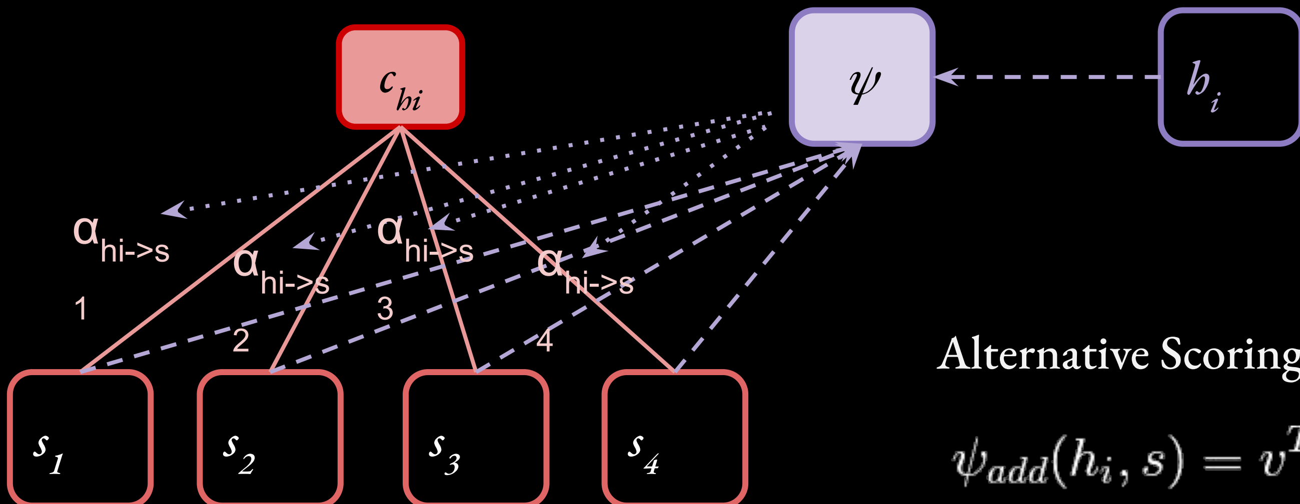
Score function:

$$\psi(h_i, s) = v^T \tanh(W_h h_i + W_s s])$$

$$\alpha_{h_i \to s} = \text{softmax}(\psi(h_i, s))$$

$$c_{h_i} = \sum_{n=1}^{|s|} \alpha_{h_i \to s_n} z_n$$

Attention



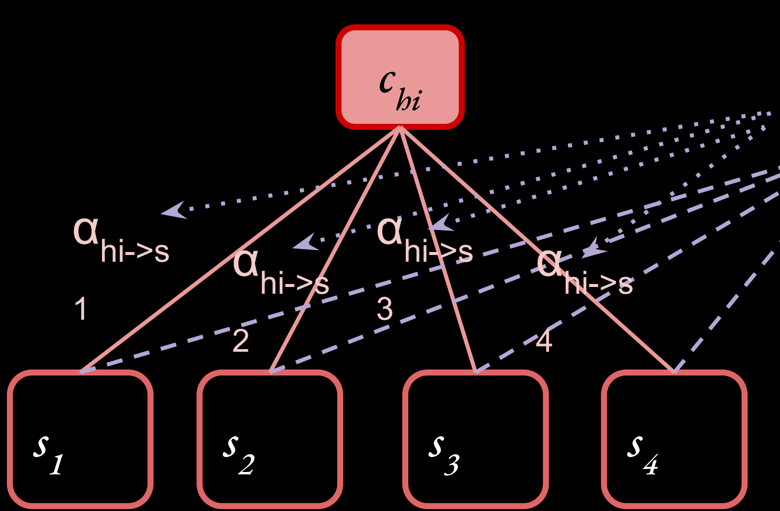
Alternative Scoring Functions

$$\psi_{add}(h_i, s) = v^T \tanh(W_h h_i + W_s s])$$

$$\psi_{dp}(h_i, s) = s^T h_i$$

$$\psi_{mult}(h_i, s) = s^T W h_i$$

Attention



If variables are standardized, matrix multiply produces a similarity score.

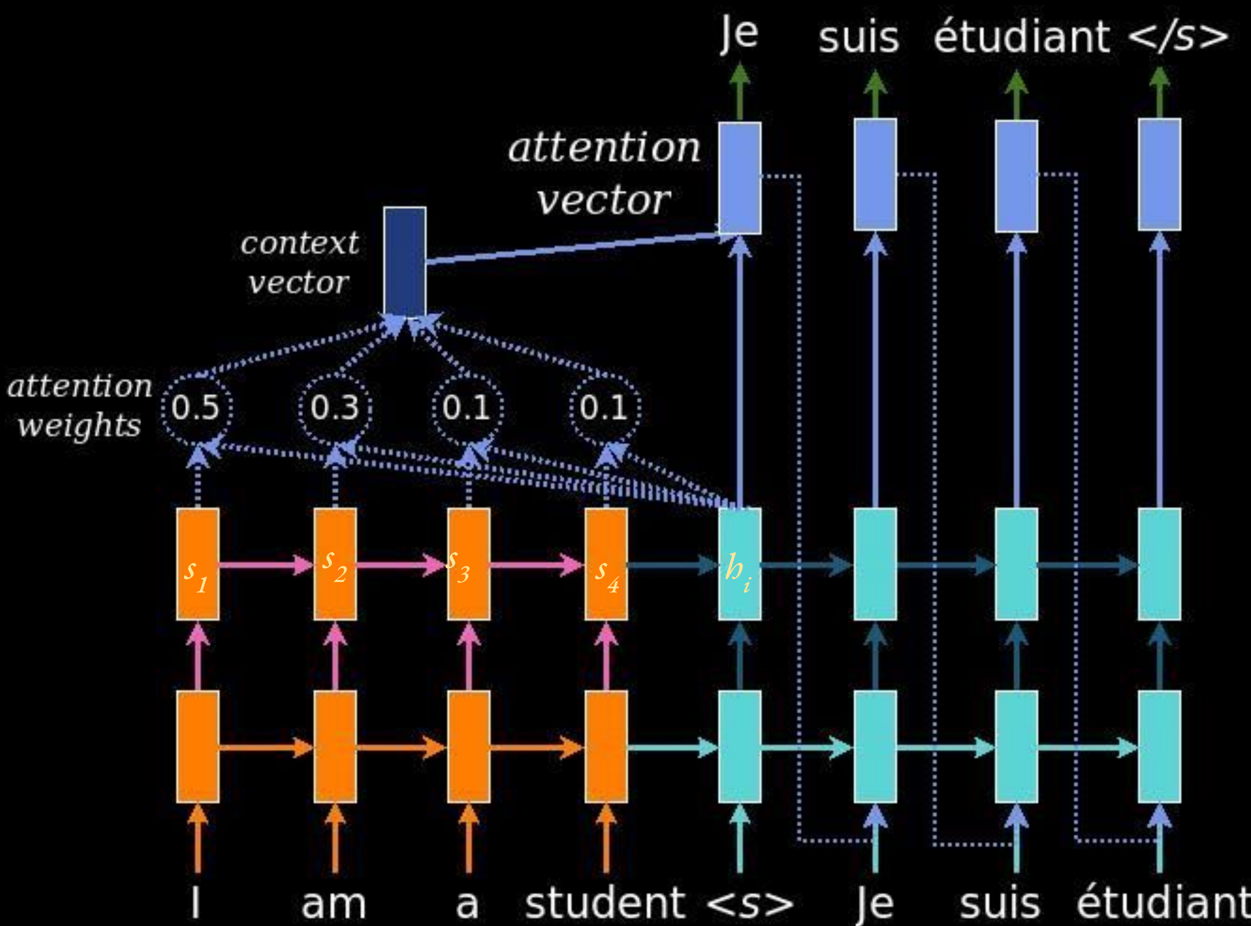
Alternative Scoring Functions

$$\psi_{add}(h_i, s) = v^T \tanh(W_h h_i + W_s s])$$

$$\psi_{dp}(h_i, s) = s^T h_i$$

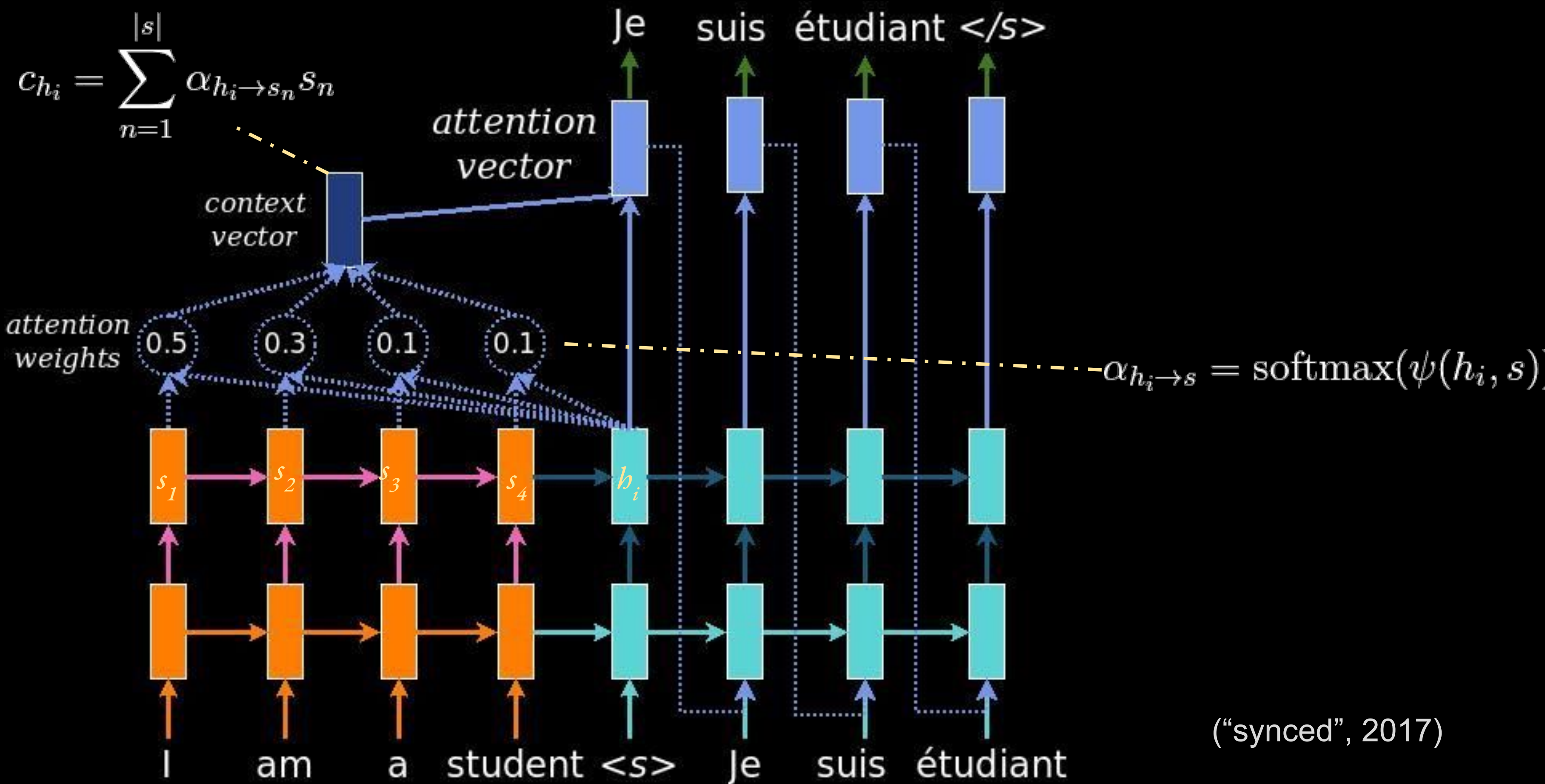
$$\psi_{mult}(h_i, s) = s^T W h_i$$

Attention



("synced", 2017)

Attention

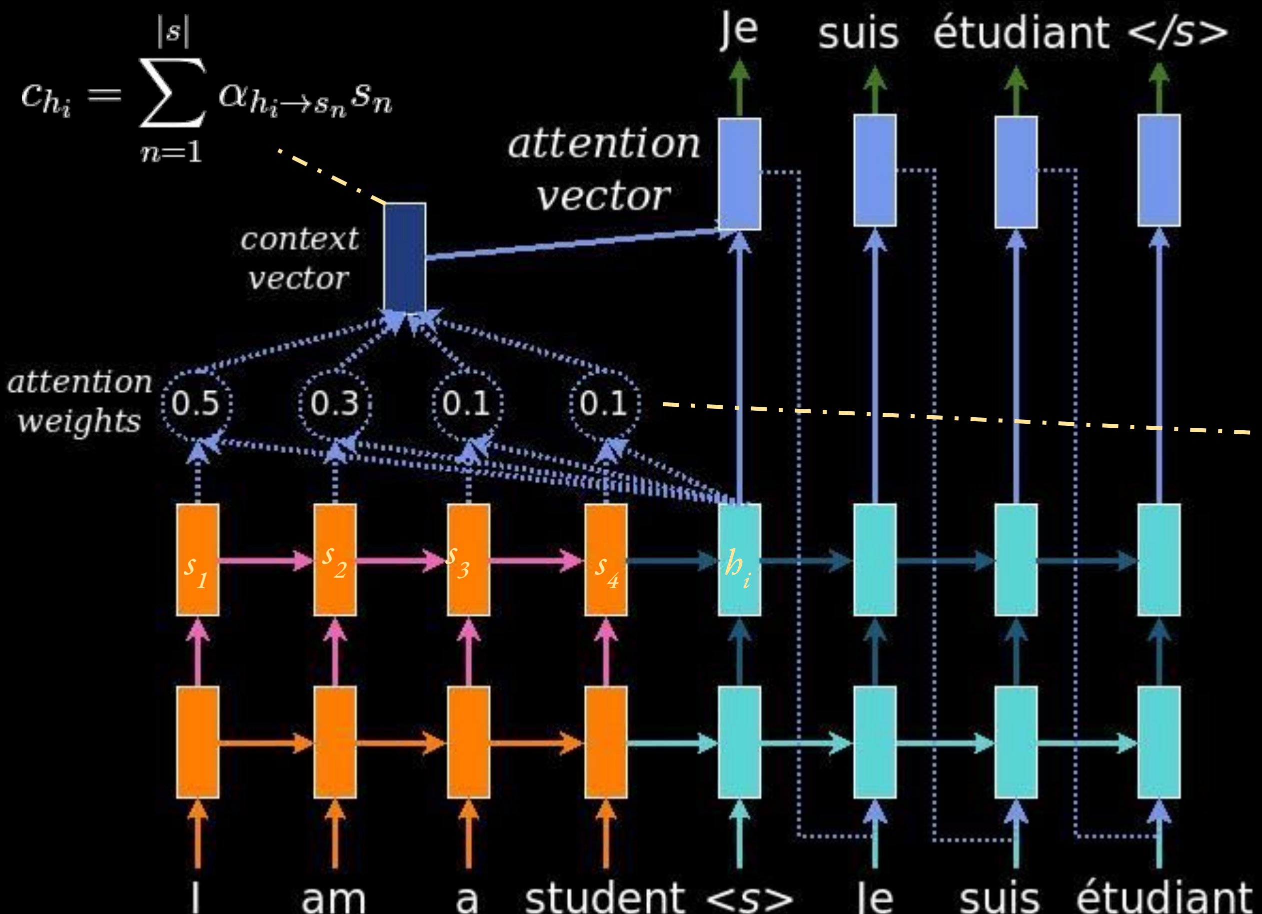


(“synced”, 2017)

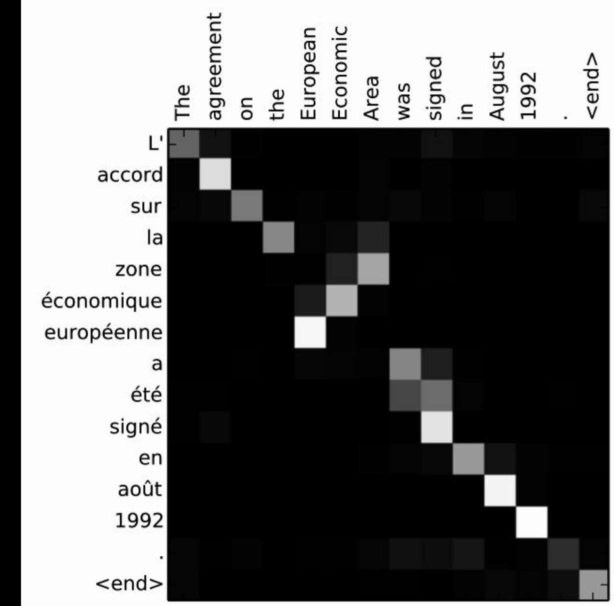
Attention

(Bahdanau et al., 2015)

$$c_{h_i} = \sum_{n=1}^{|s|} \alpha_{h_i \to s_n} s_n$$



$$\alpha_{h_i \to s} = \mathrm{softmax}(\psi(h_i, s))$$

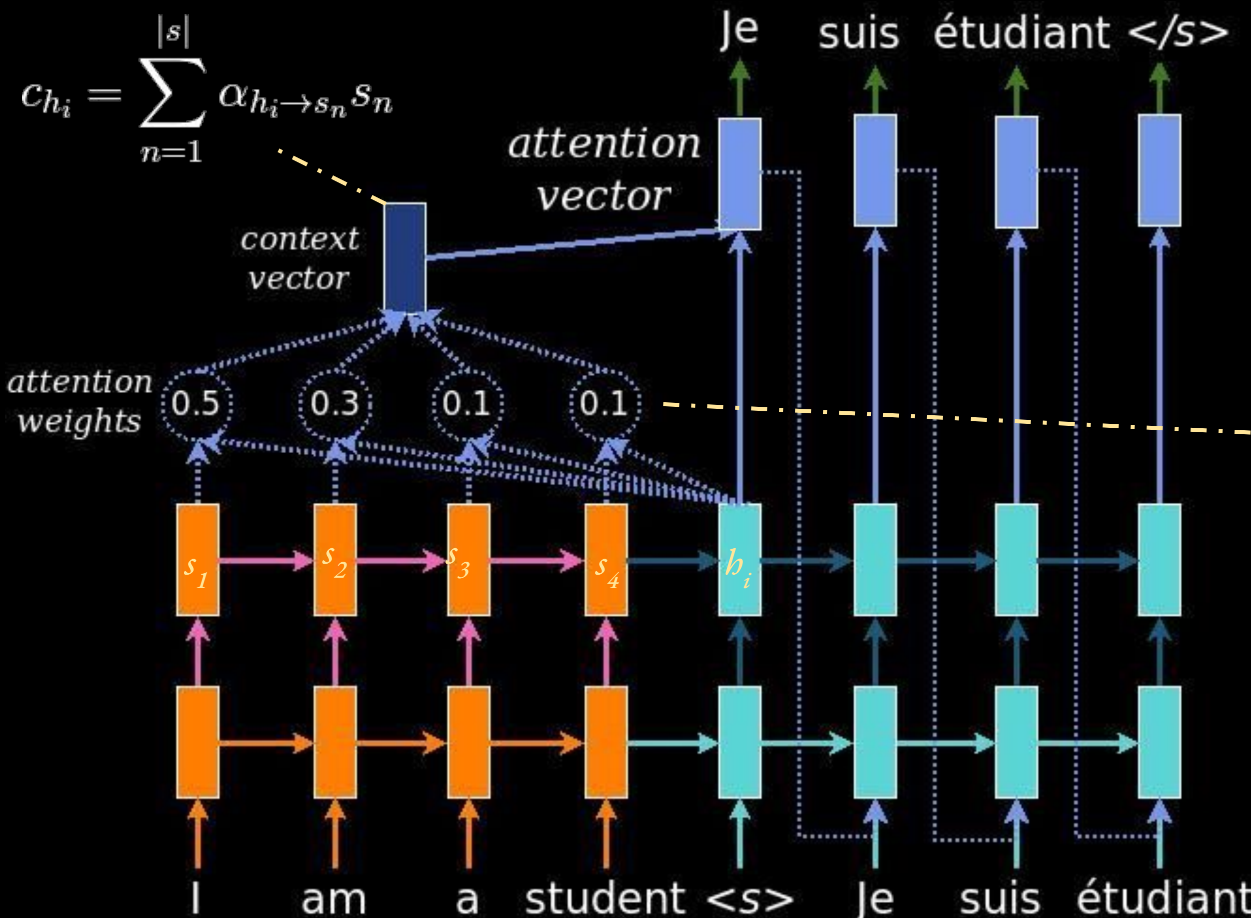


(“synced”, 2017)

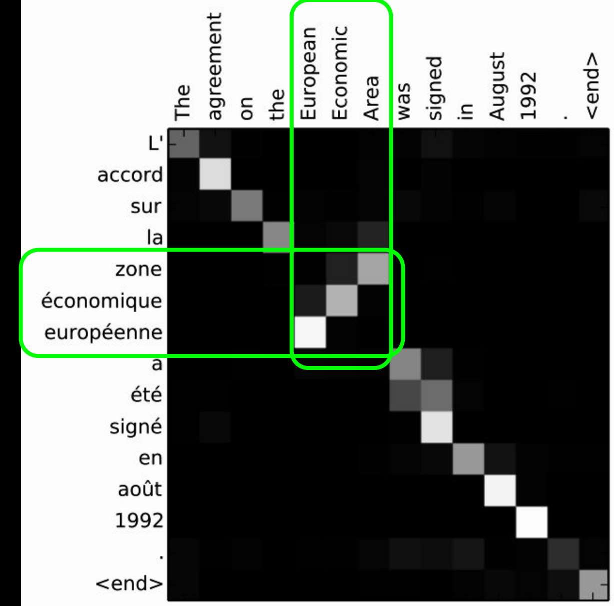
Attention

(Bahdanau et al., 2015)

$$c_{h_i} = \sum_{n=1}^{|s|} \alpha_{h_i \to s_n} s_n$$



$$\alpha_{h_i \to s} = \mathrm{softmax}(\psi(h_i, s))$$

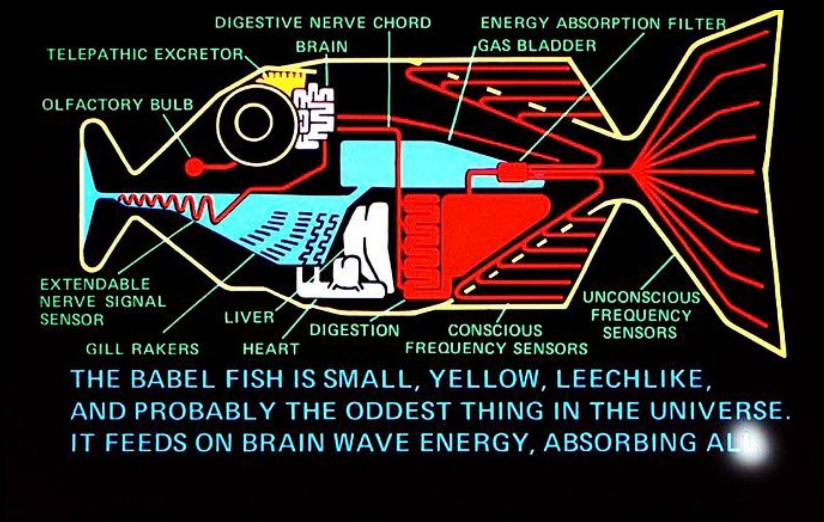


(“synced”, 2017)

Machine Translation

Why?

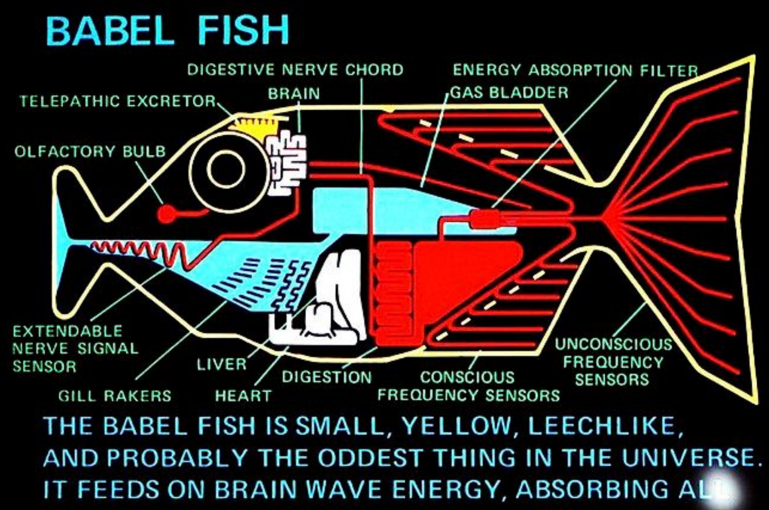
- $40billion/year industry
- A center piece of many genres of science fiction
- A fairly “universal” problem:
  - Language understanding
  - Language generation
- Societal benefits of inter-cultural communication



Machine Translation

Why?

- $40billion/year industry
- A center piece of many genres of science fiction
- A fairly "universal" problem:
  - Language understanding
  - Language generation
- Societal benefits of inter-cultural communication



(Douglas Adams)

Machine Translation

Why Neural Network Approach works? (Manning, 2018)

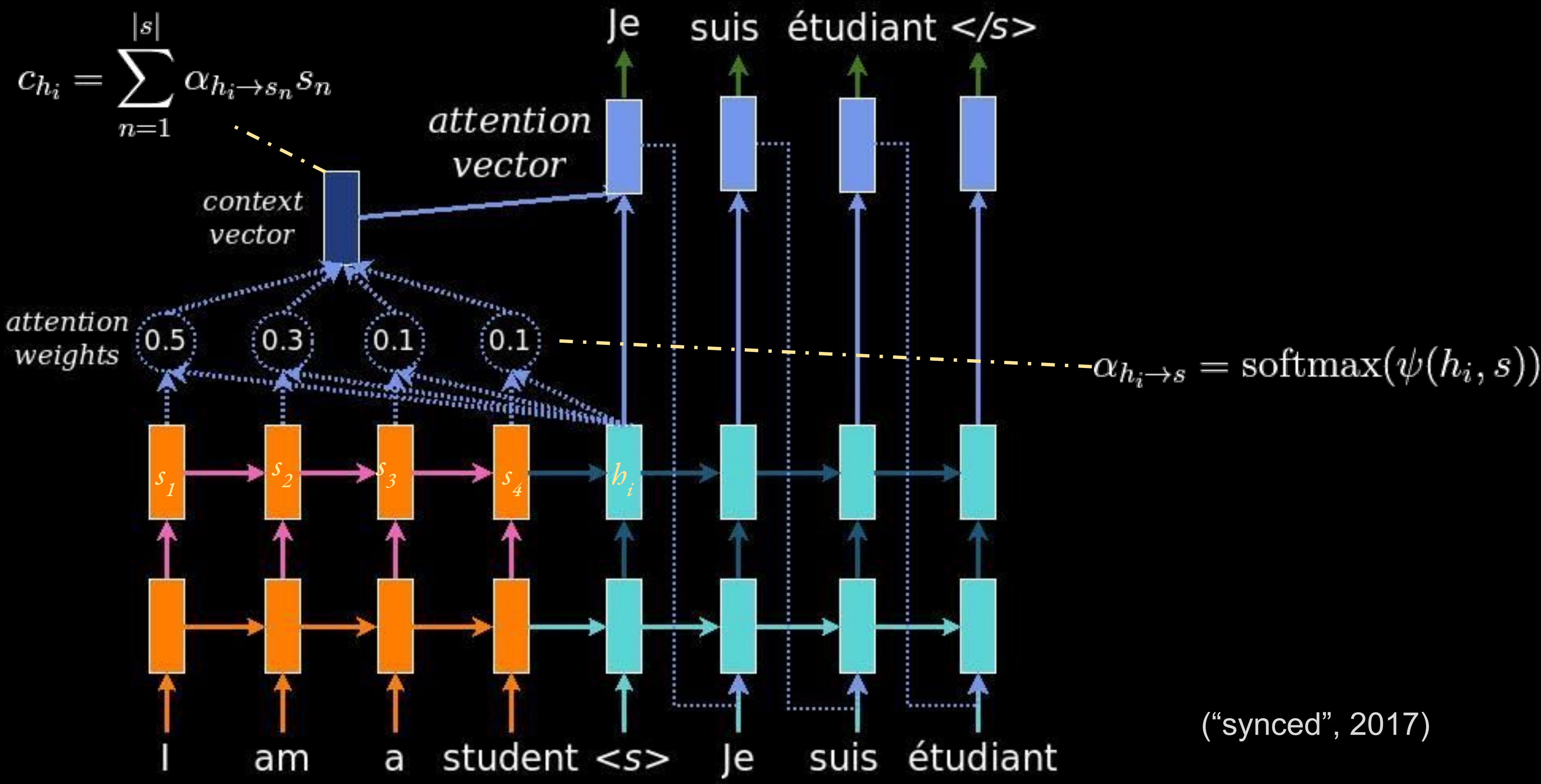
- Joint end-to-end training: learning all parameters at once.
- Exploiting distributed representations (embeddings)
- Exploiting variable-length context
- High quality generation from deep decoders - stronger language models (even when wrong, make sense)

Machine Translation

As an optimization problem (Eisenstein, 2018):

$$\hat{\boldsymbol{w}}^{(t)} = \underset{\boldsymbol{w}^{(t)}}{\operatorname{argmax}} \Psi(\boldsymbol{w}^{(s)}, \boldsymbol{w}^{(t)})$$

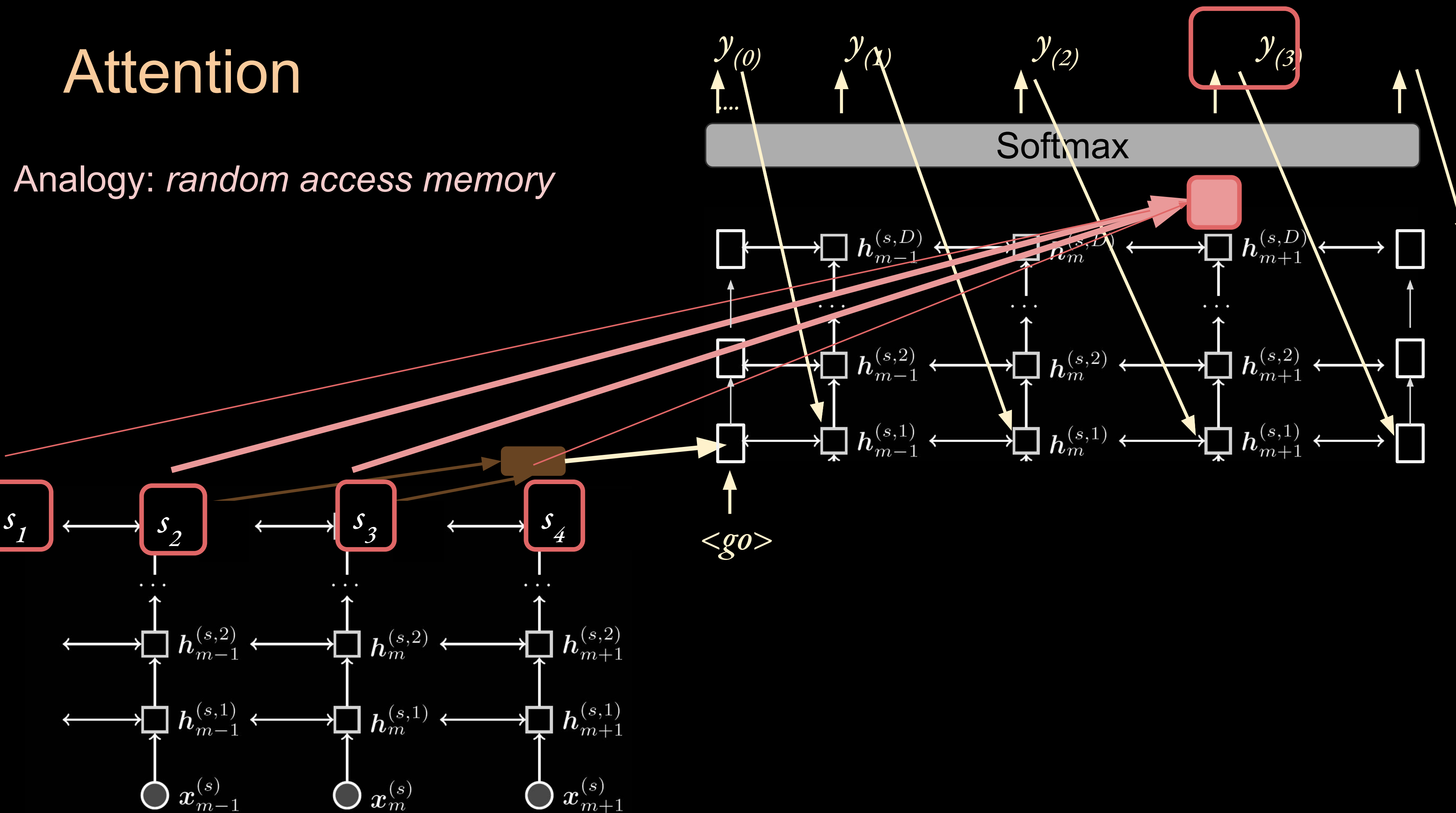
Attention



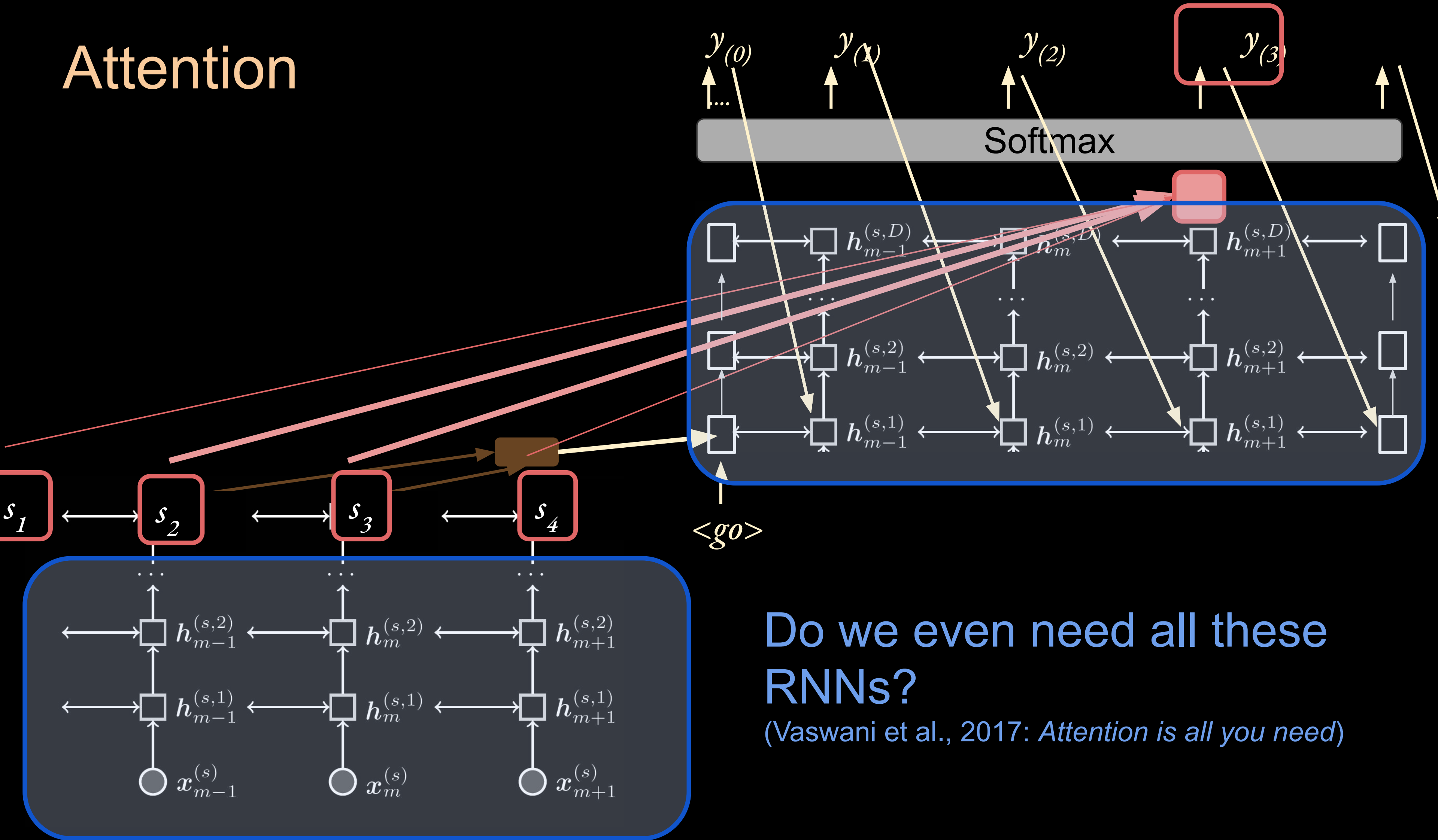
("synced", 2017)

Attention

Analogy: *random access memory*



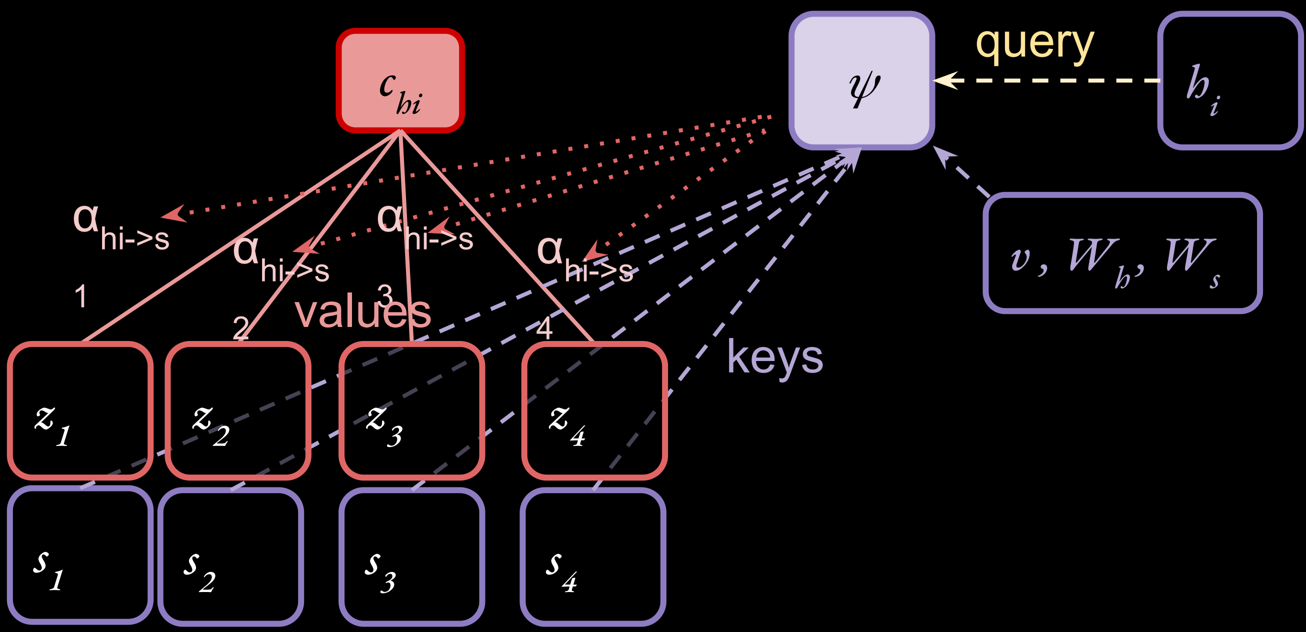
Attention



Do we even need all these RNNs?

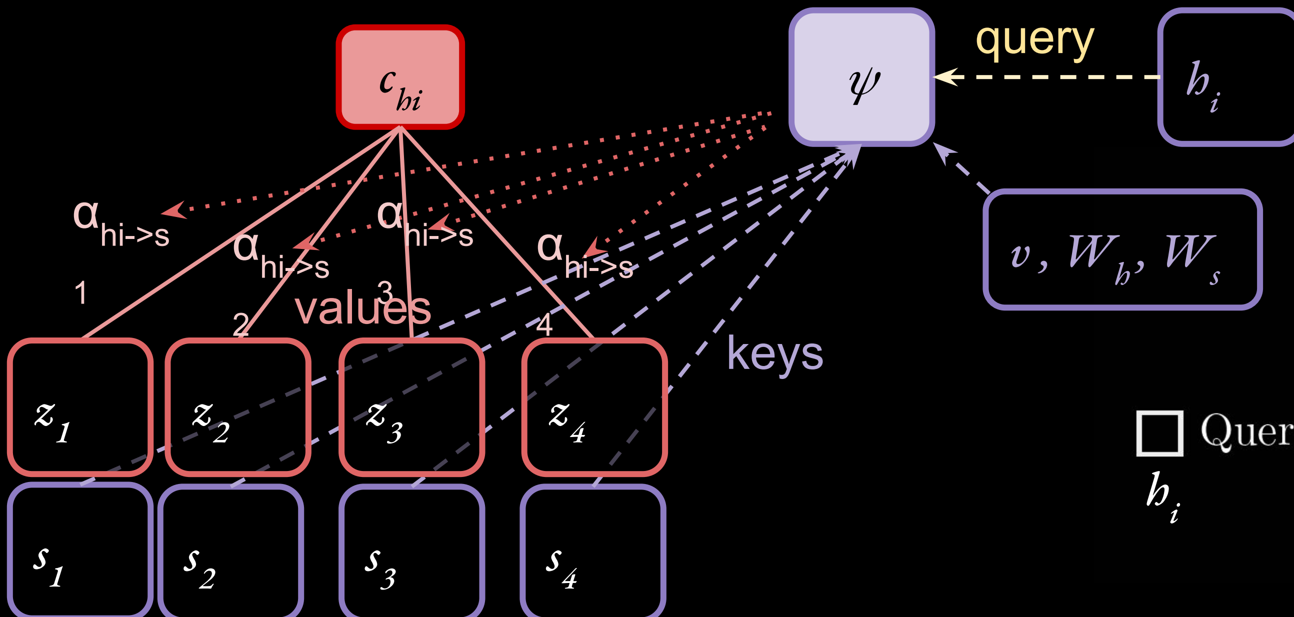
(Vaswani et al., 2017: *Attention is all you need*)

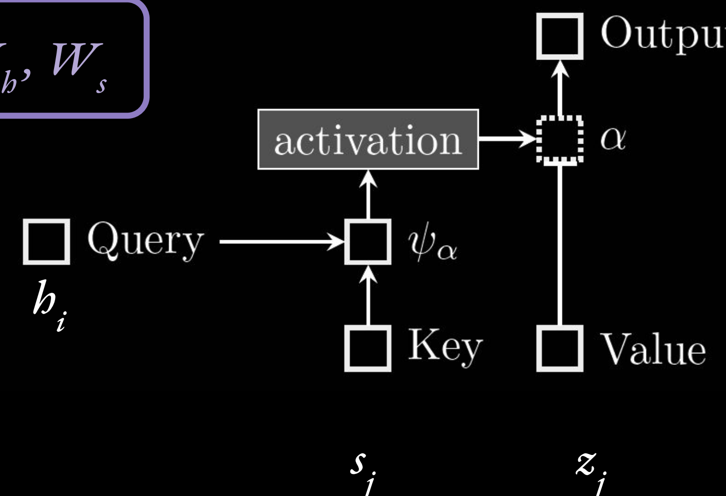
Attention



A useful abstraction is to make the vector attended to (the "value vector", Z) separate than the "key vector" (s).

Attention



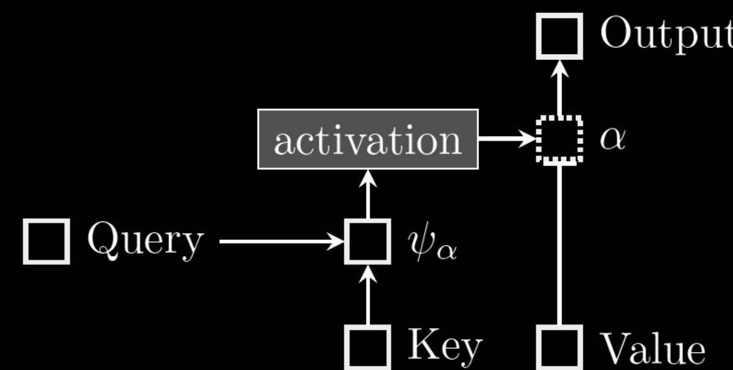


A useful abstraction is to make the vector attended to (the "value vector", Z) separate than the "key vector" (s).

(Eisenstein, 2018)

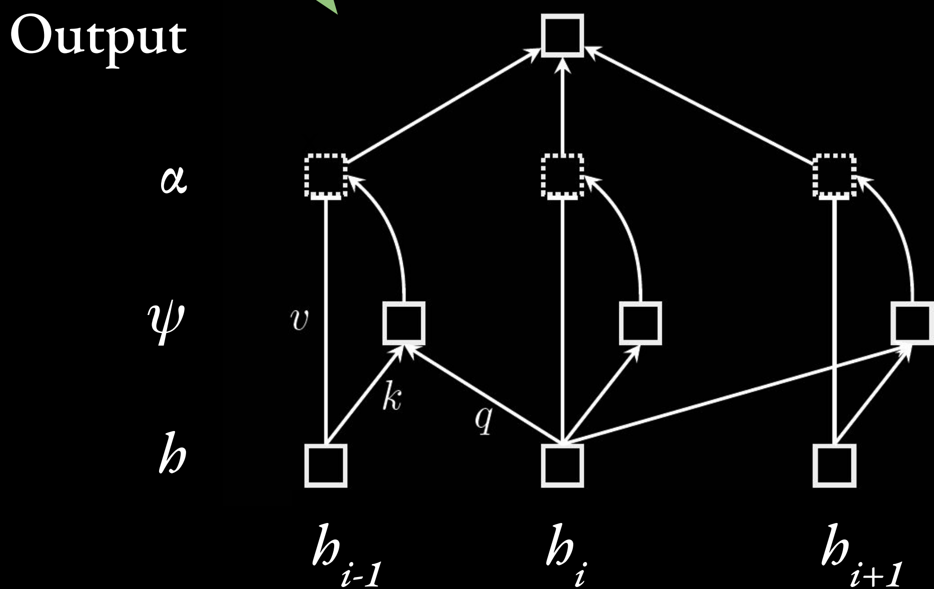
The Transformer: “Attention-only” models

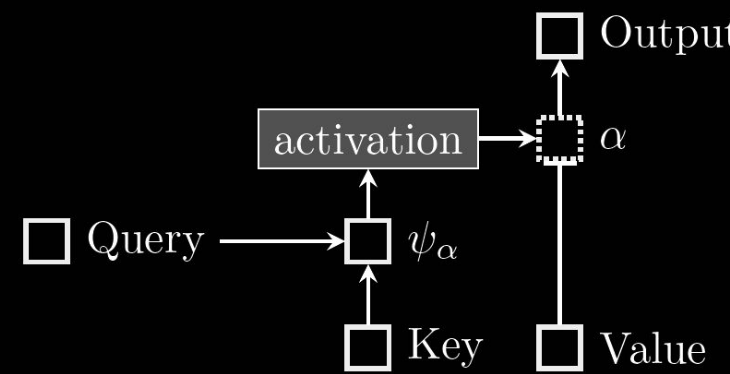
Attention as weighting a value based on a query and key:



(Eisenstein, 2018)

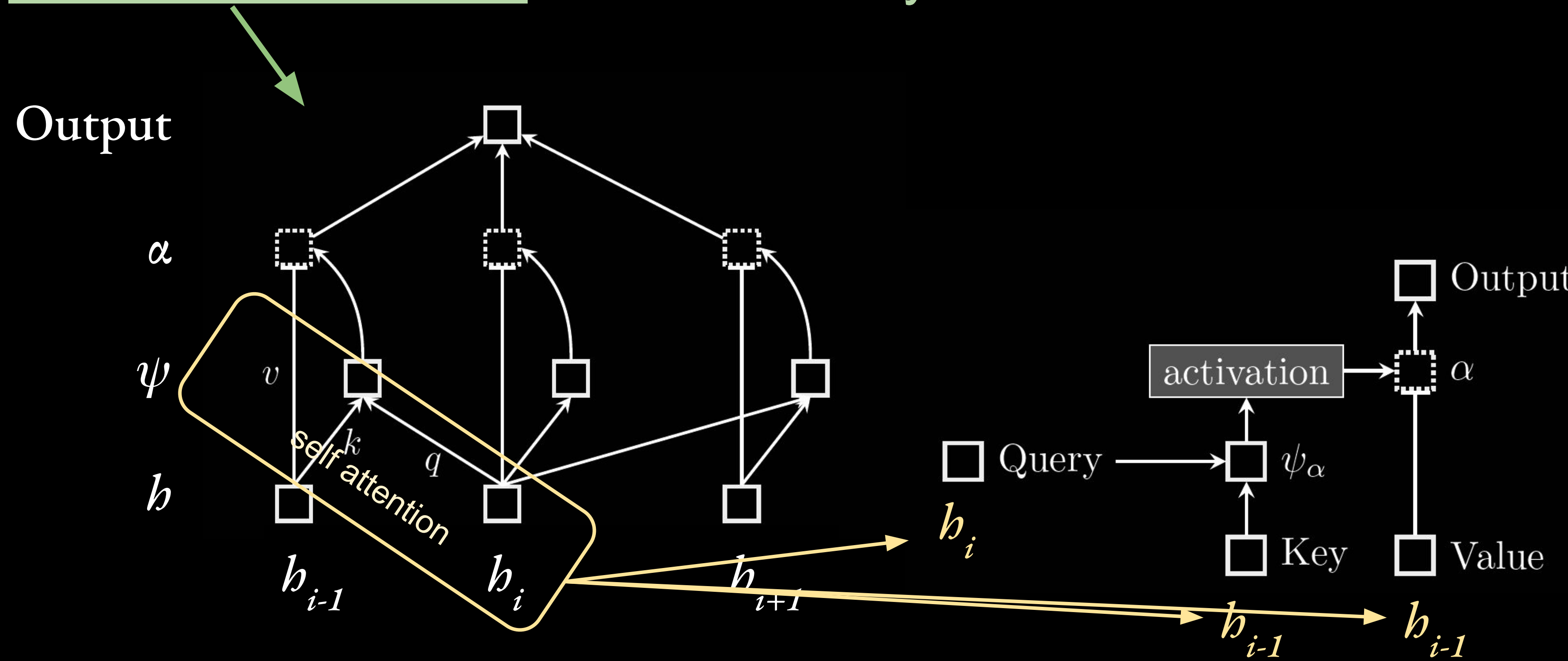
The Transformer: “Attention-only” models





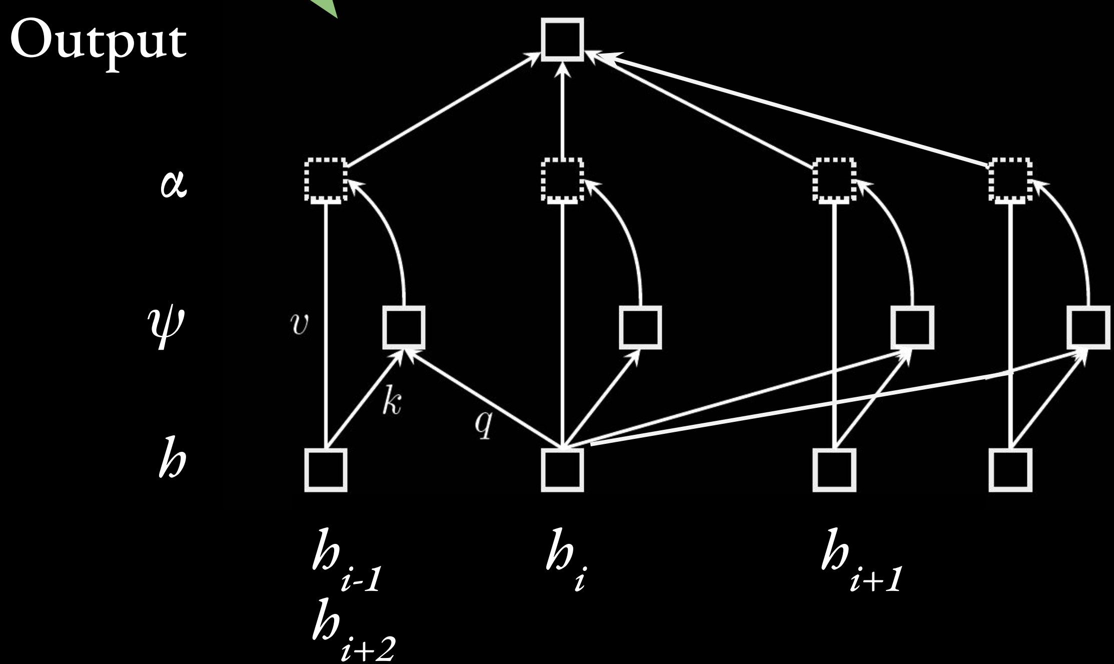
(Eisenstein, 2018)

The Transformer: "Attention-only" models

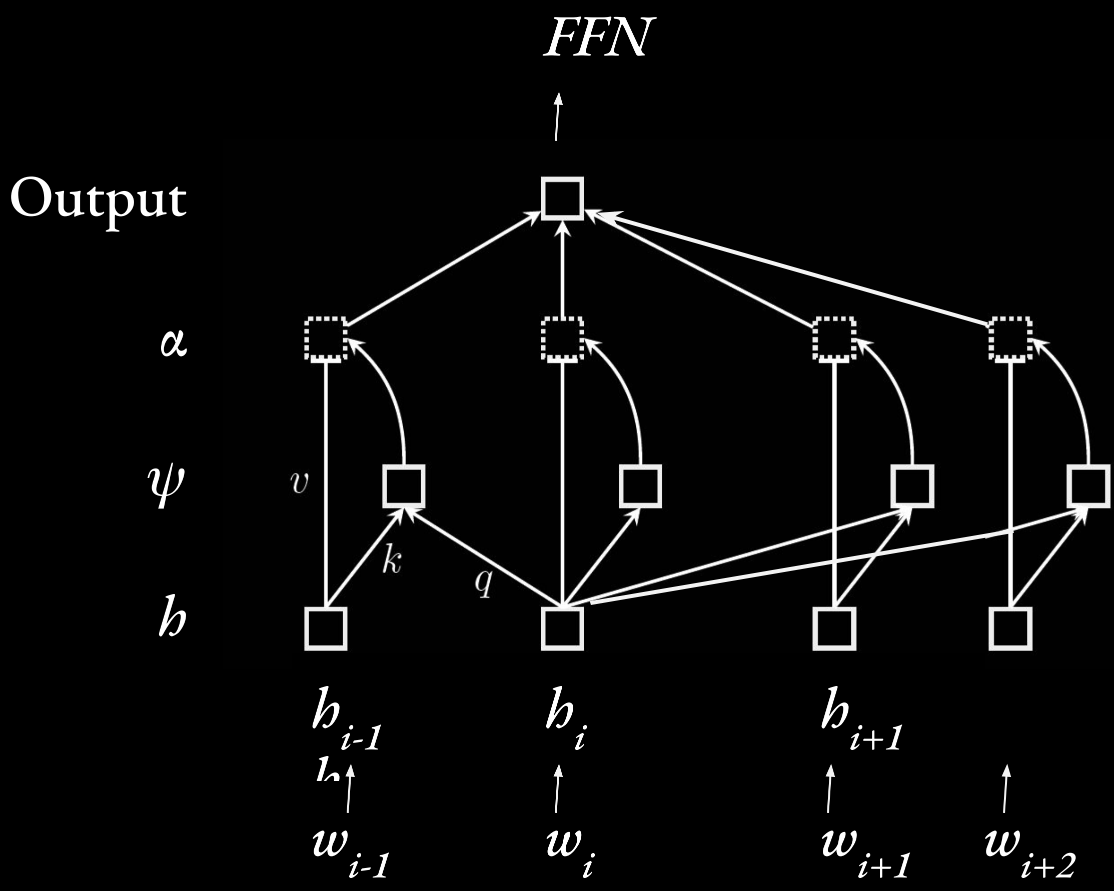


(Eisenstein, 2018)

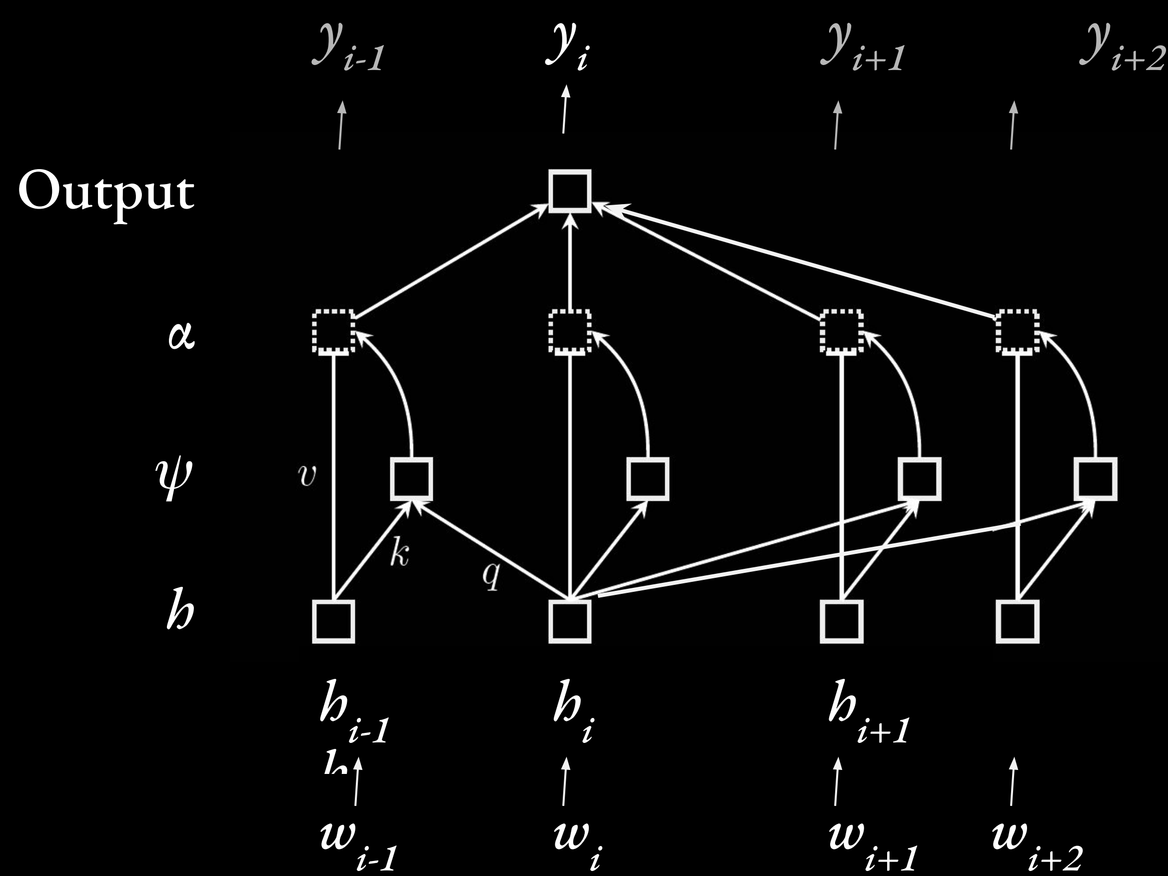
The Transformer: "Attention-only" models



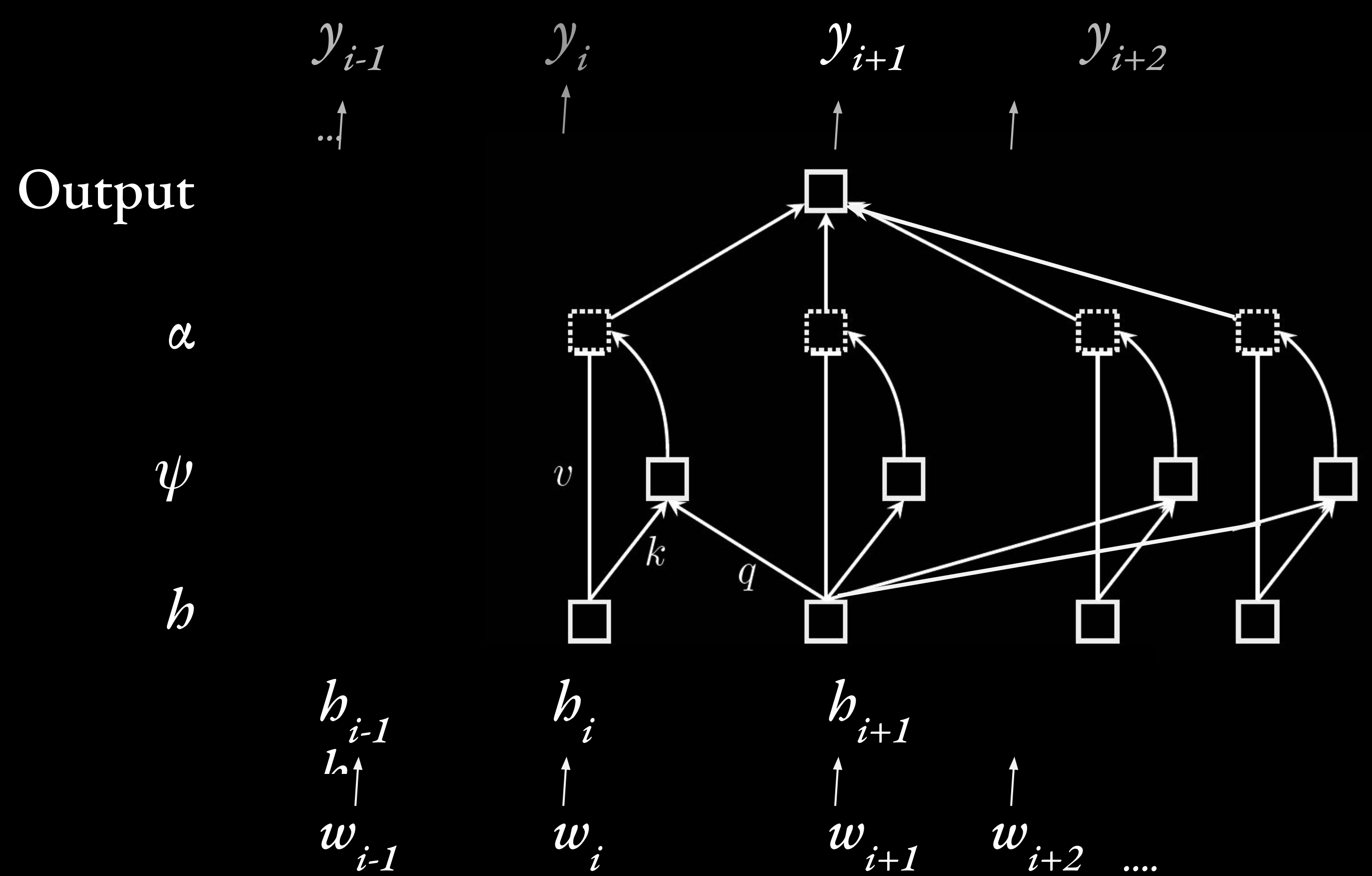
The Transformer: “Attention-only” models



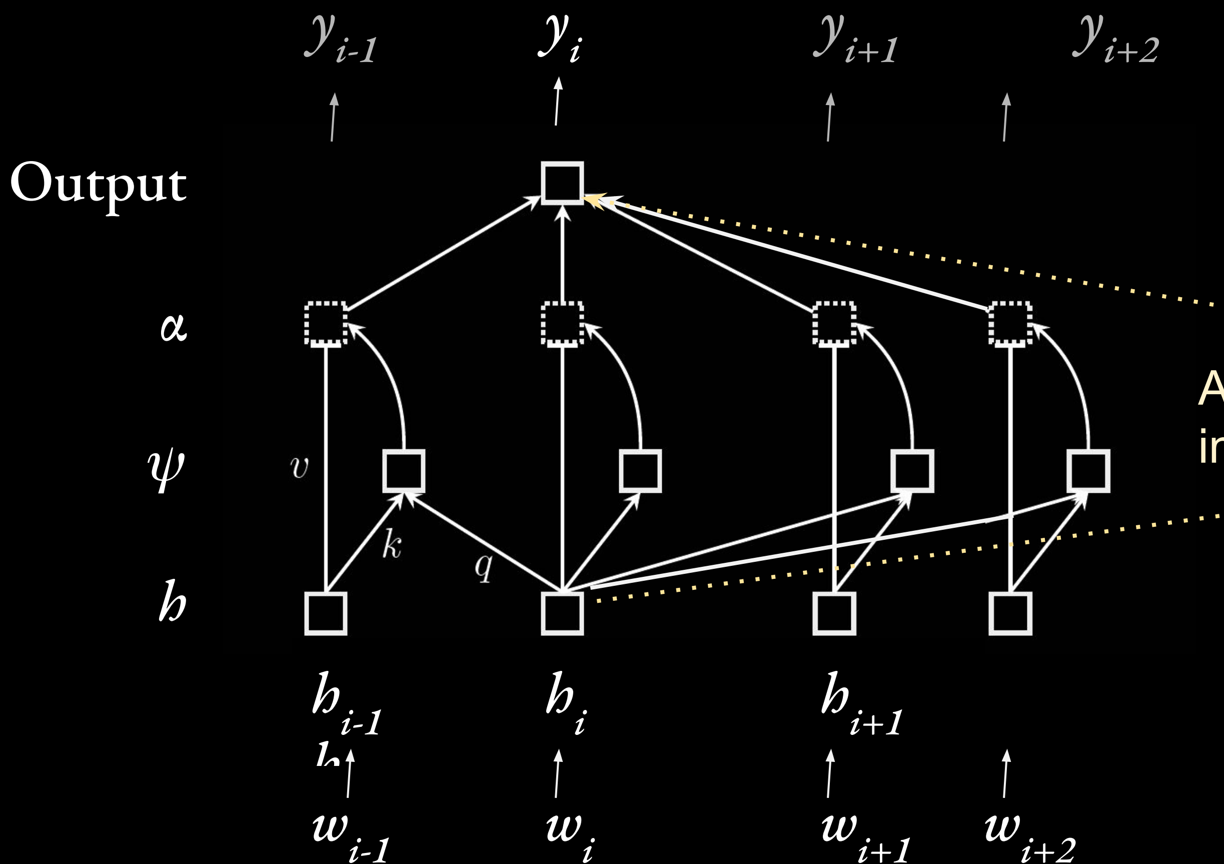
The Transformer: "Attention-only" models



The Transformer: "Attention-only" models

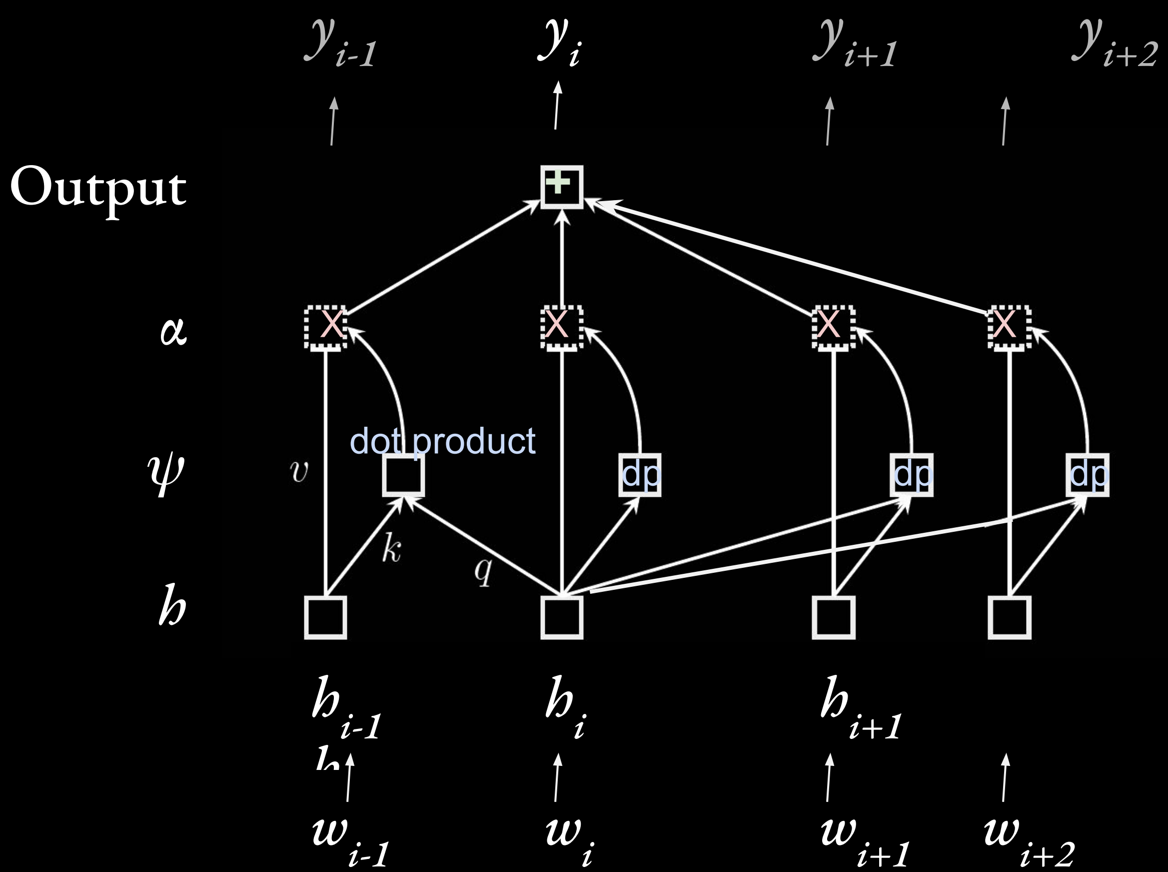


The Transformer: "Attention-only" models



Attend to all hidden states in your "neighborhood".

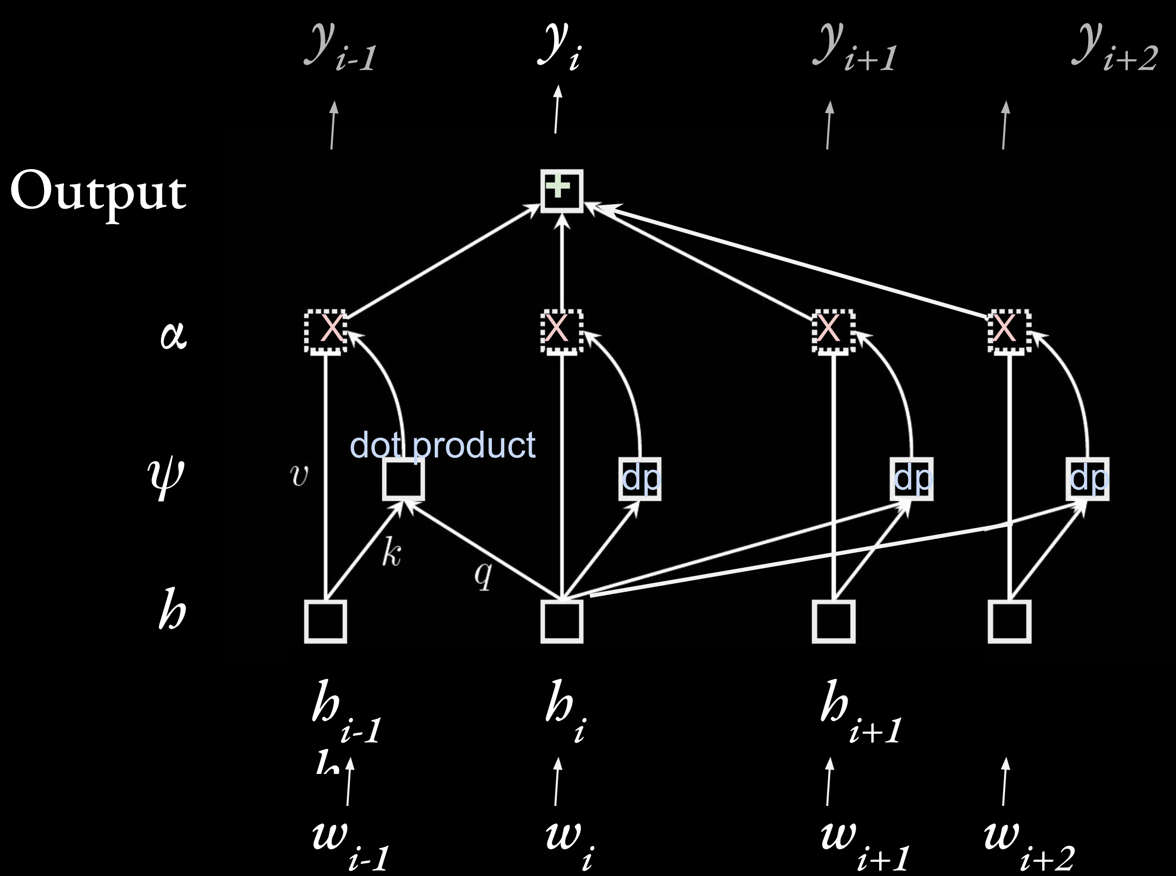
The Transformer: "Attention-only" models



$$\psi_{dp}(h_i, s) = s^T h_i$$

$k^t q$

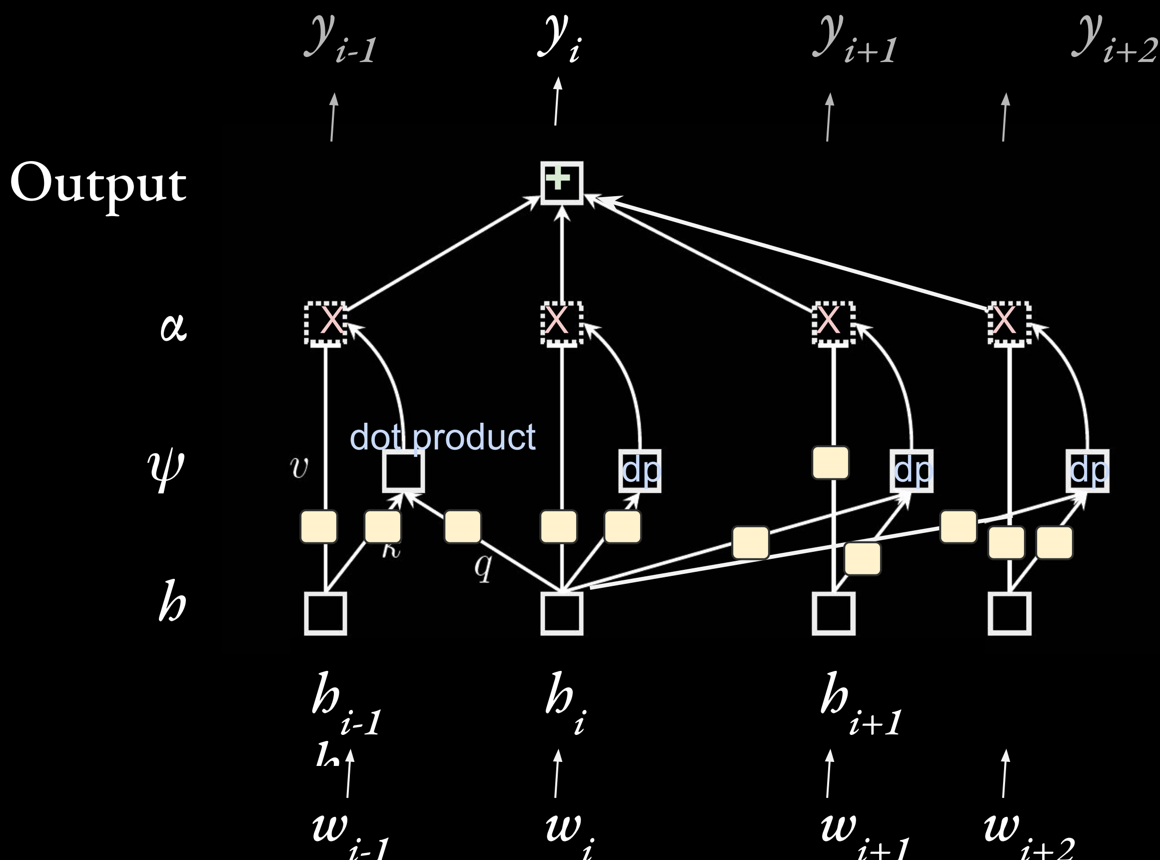
The Transformer: "Attention-only" models



scaling parameter

$$\psi_{dp}(k,q) = (k^t q)\,\sigma$$

The Transformer: "Attention-only" models



$$\psi_{dp}(k,q) = (k^t q)\,\sigma$$

Linear layer:
$W^T X$

One set of weights for each of for K, Q, and V

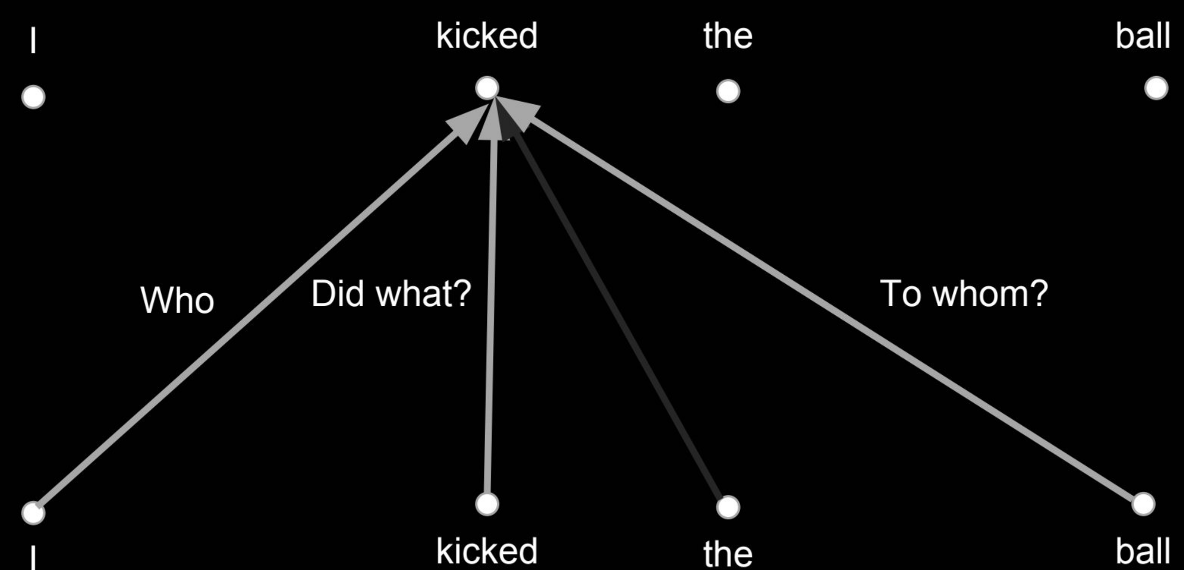
The Transformer: “Attention-only” models

Why?

- Don’t need complexity of LSTM/GRU cells
- Constant num edges between words (or input steps)
- Enables “interactions” (i.e. adaptations) between words
- Easy to parallelize -- don’t need sequential processing.

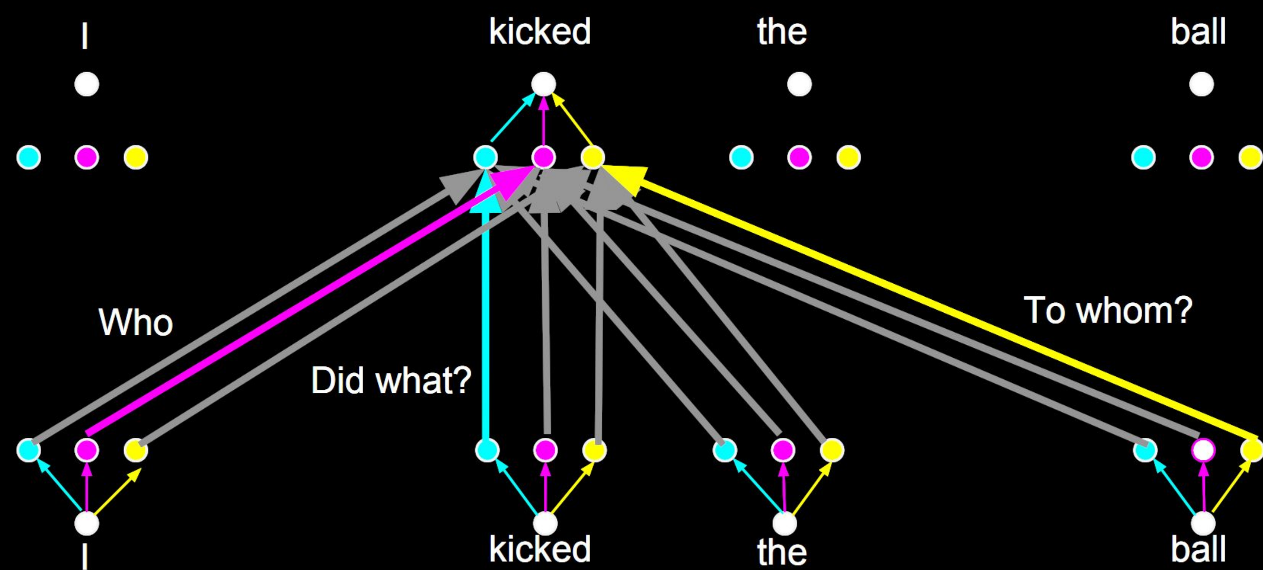
The Transformer

Limitation (thus far): Can't capture multiple types of dependencies between words.

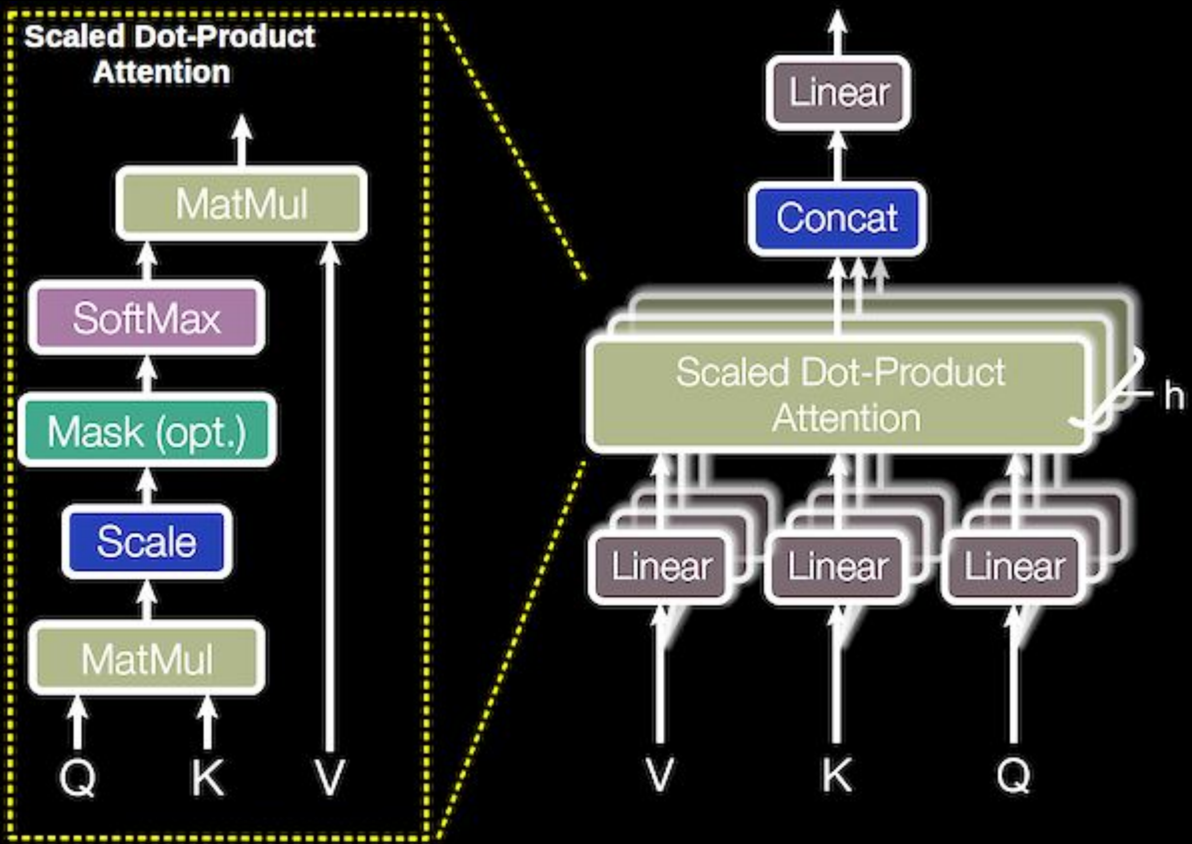


The Transformer

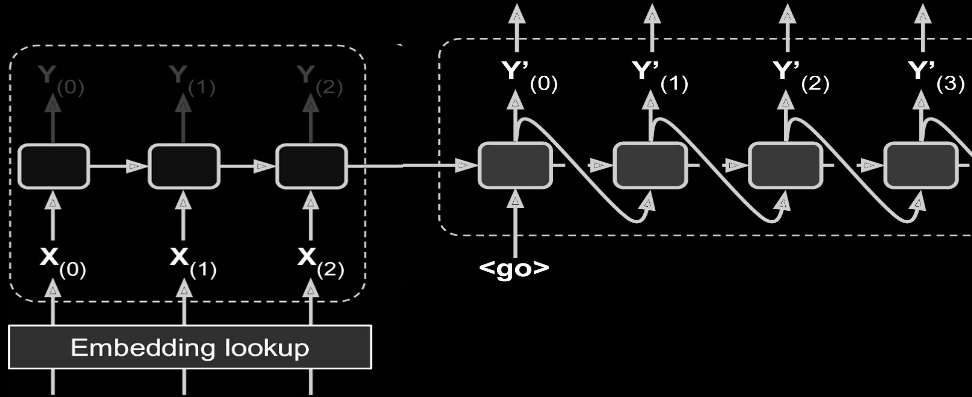
Solution: Multi-head attention

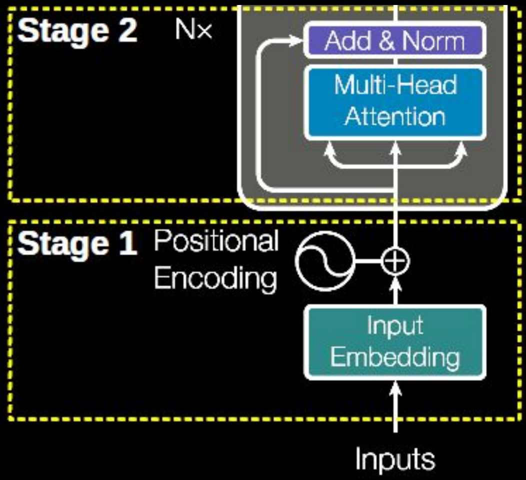


Multi-head Attention

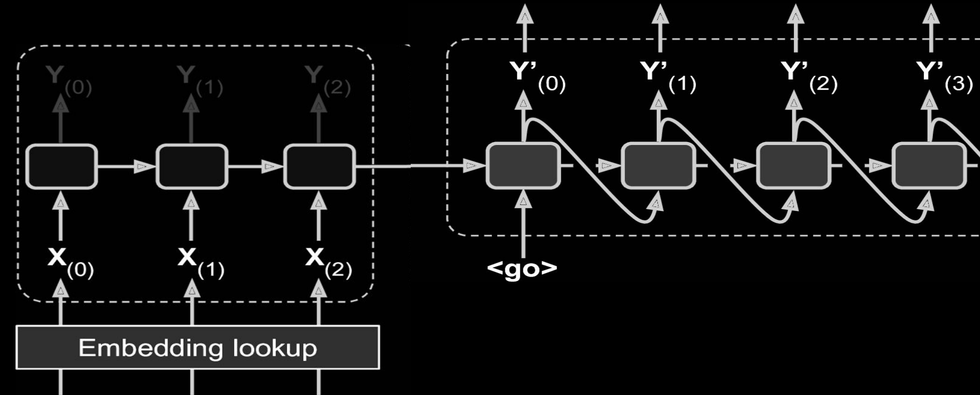


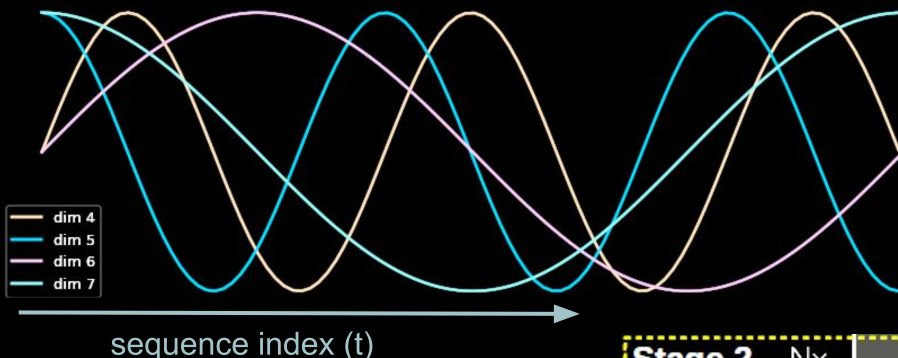
Transformer for Encoder-Decoder

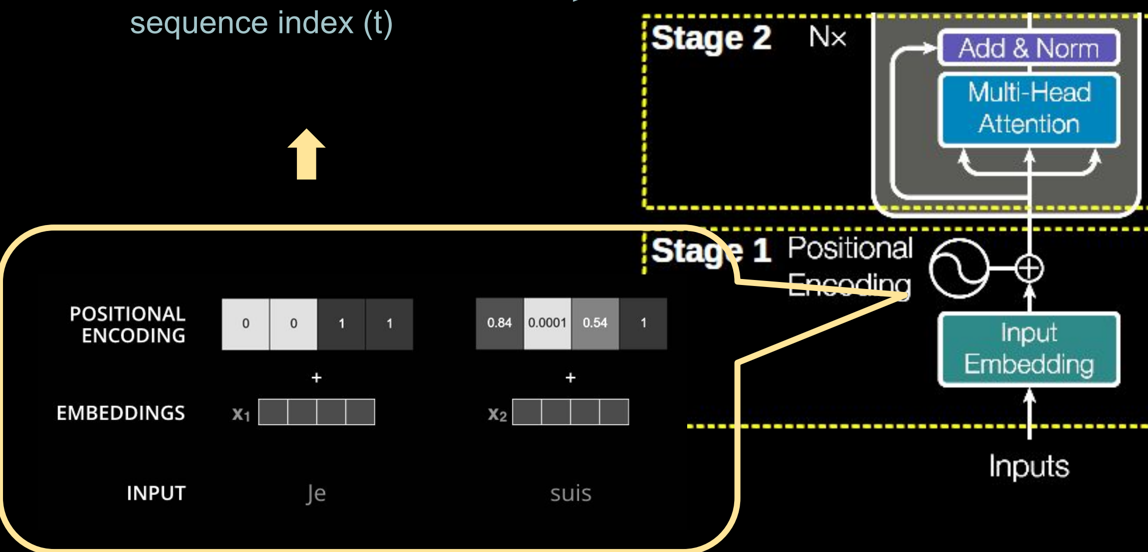




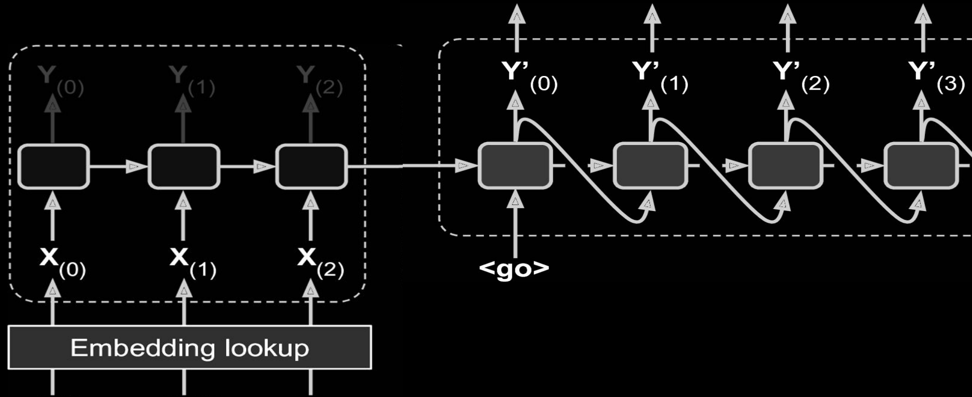
Transformer for Encoder-Decoder

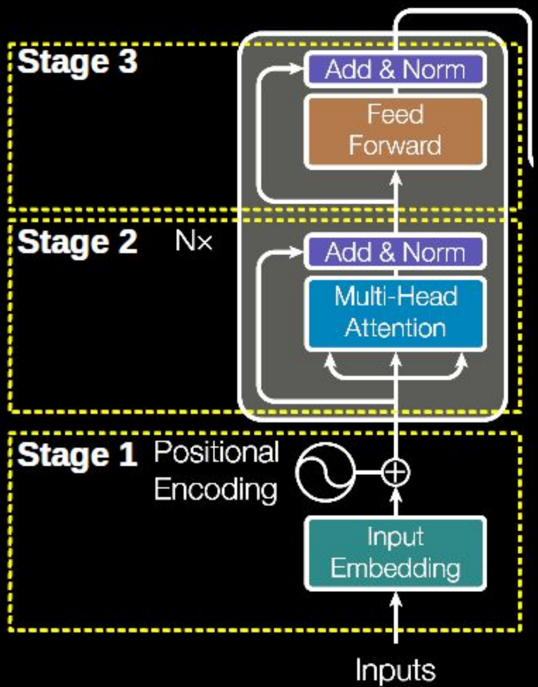




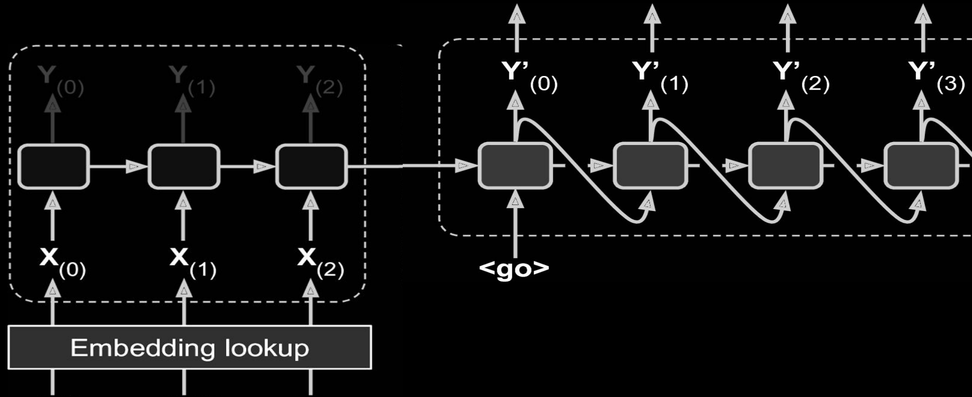


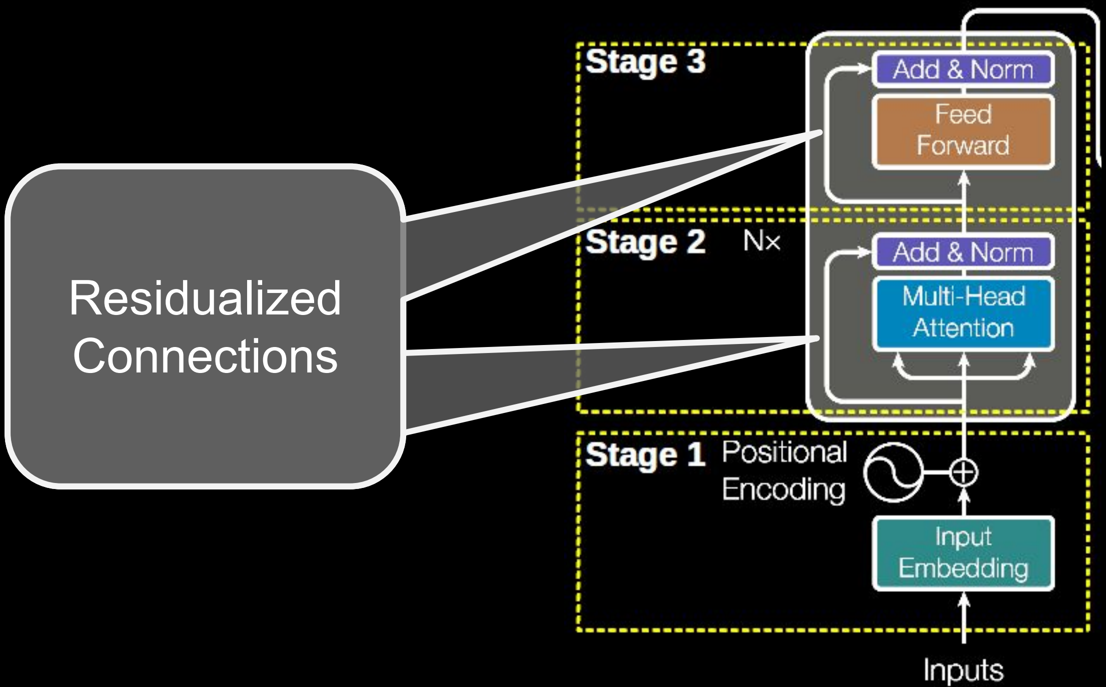
Transformer for Encoder-Decoder



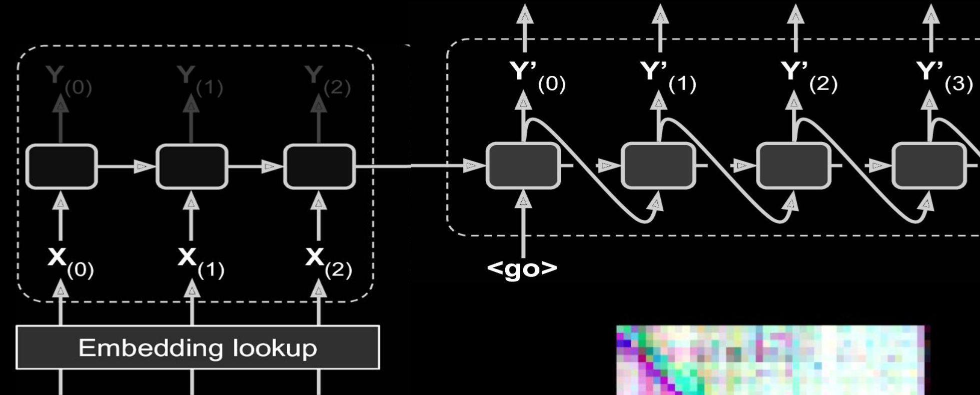


Transformer for Encoder-Decoder

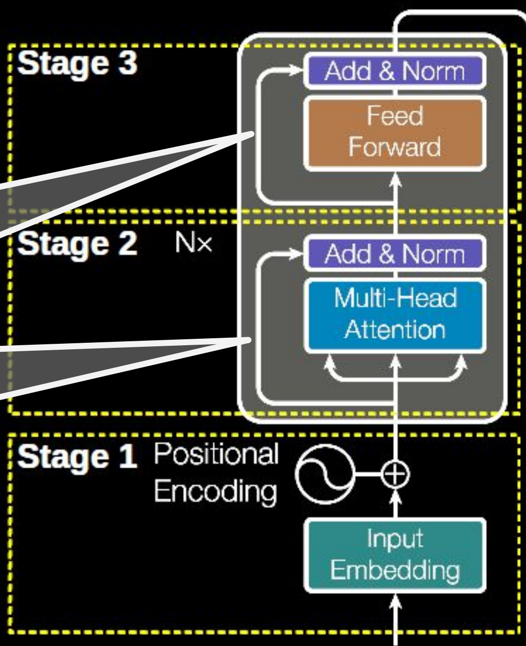




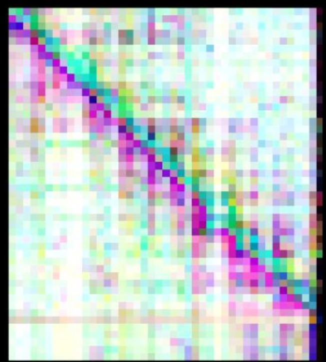
Transformer for Encoder-Decoder



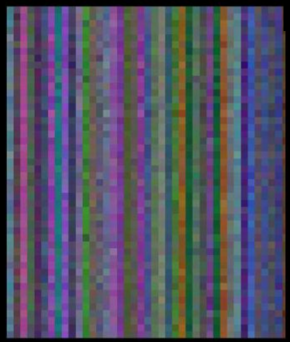
Residualized Connections



residuals enable positional information to be passed along

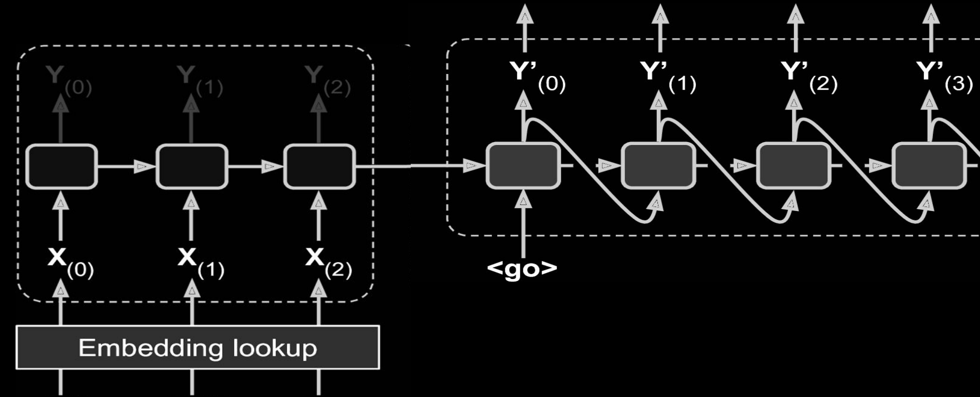


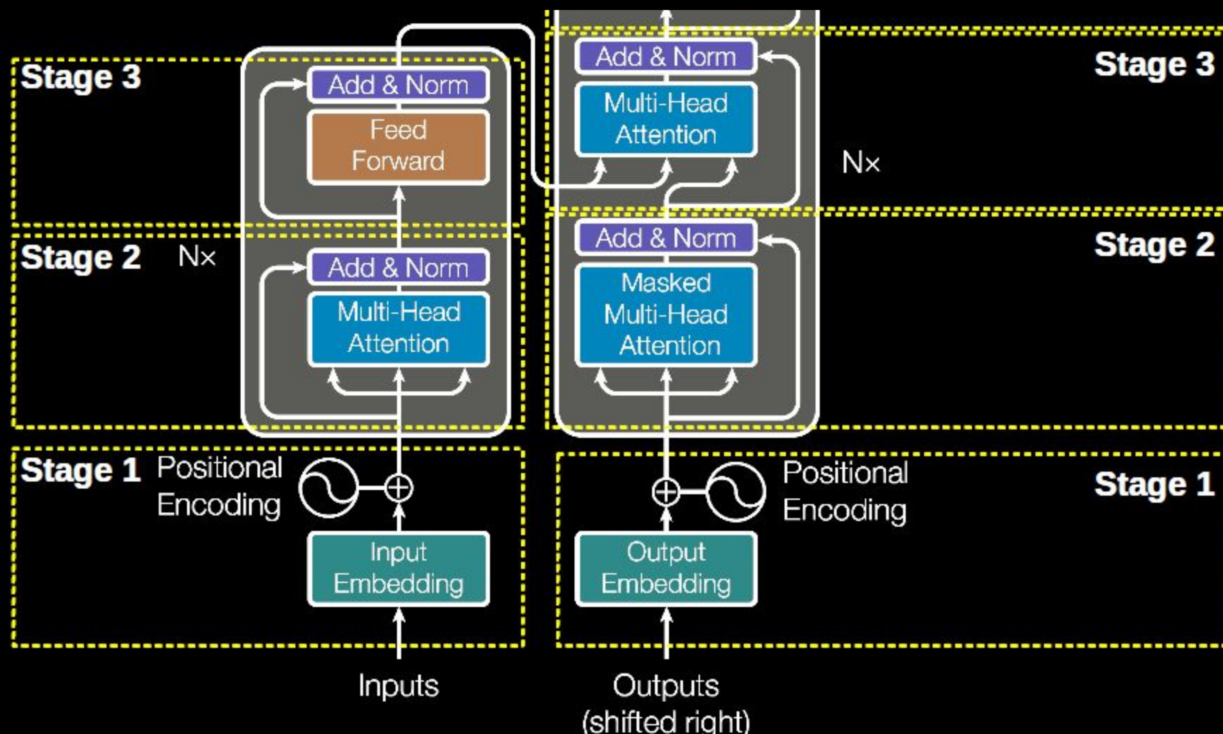
With residuals



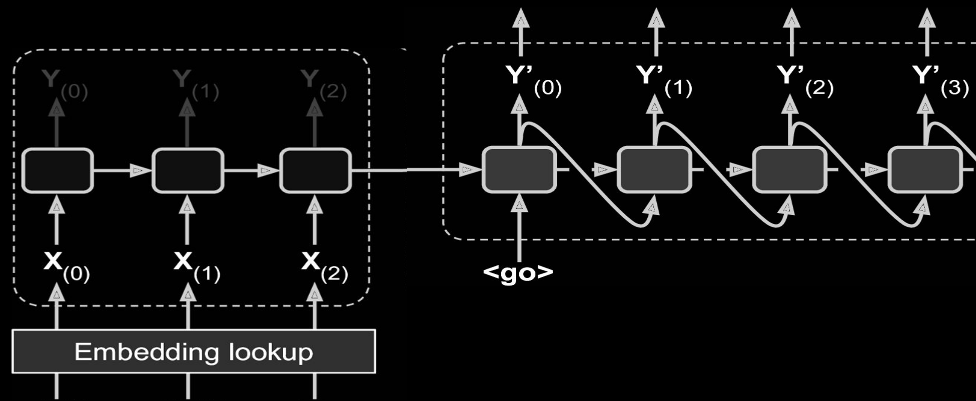
Without residuals

Transformer for Encoder-Decoder

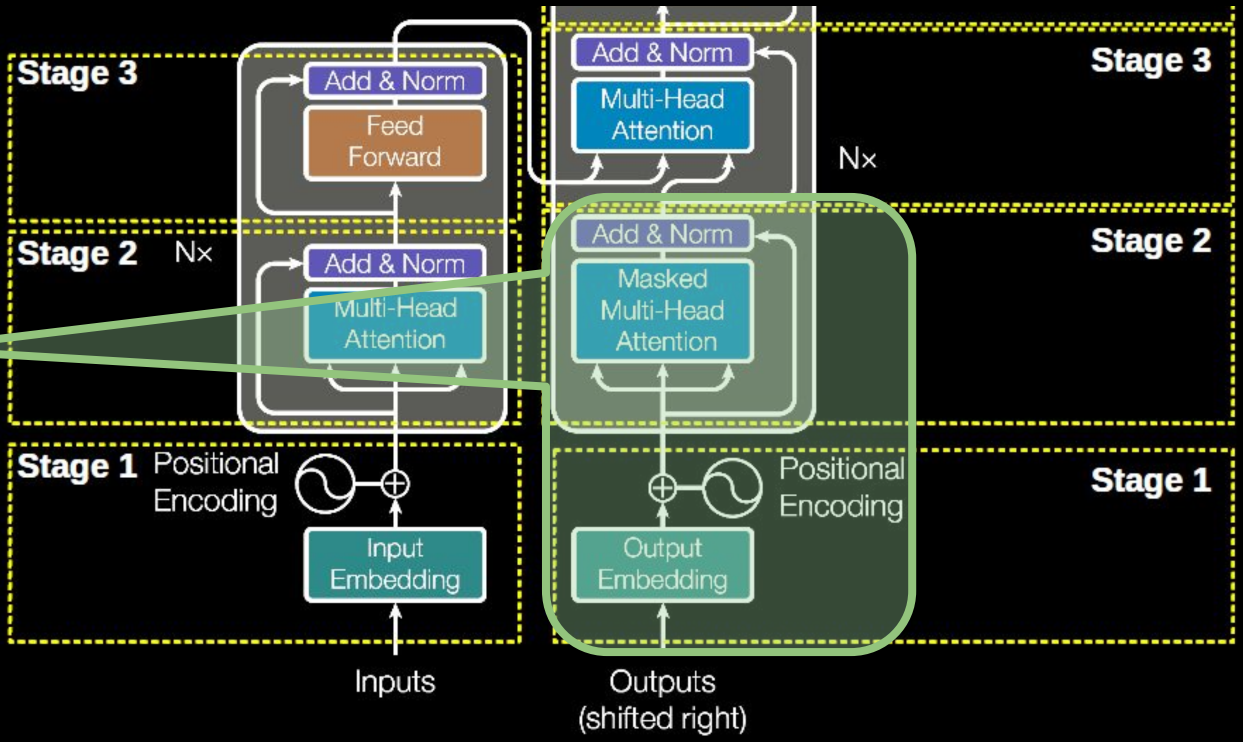




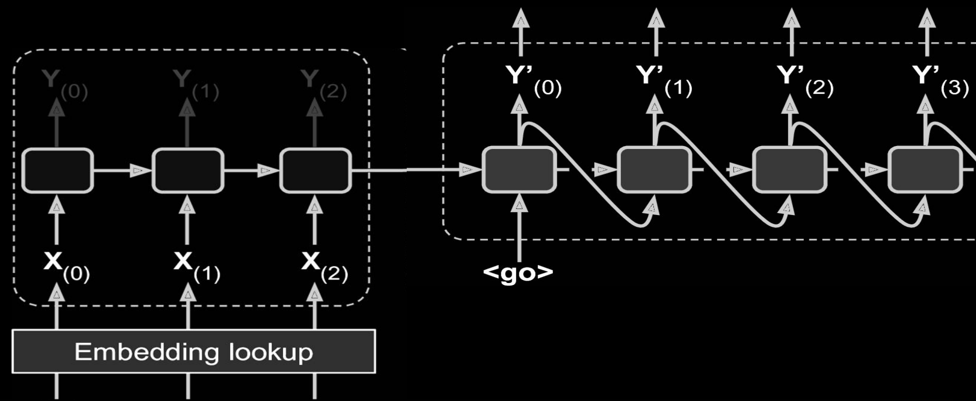
Transformer for Encoder-Decoder



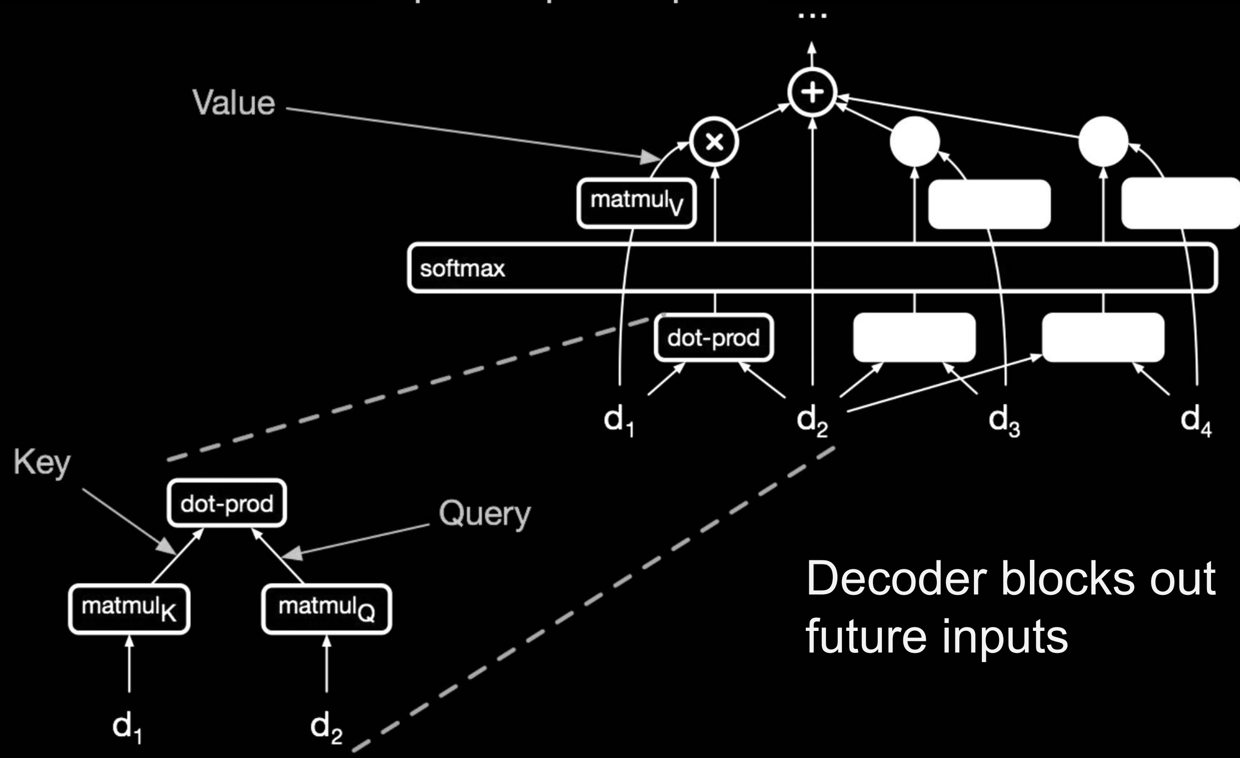
essentially, a language model



Transformer for Encoder-Decoder



essentially, a language model

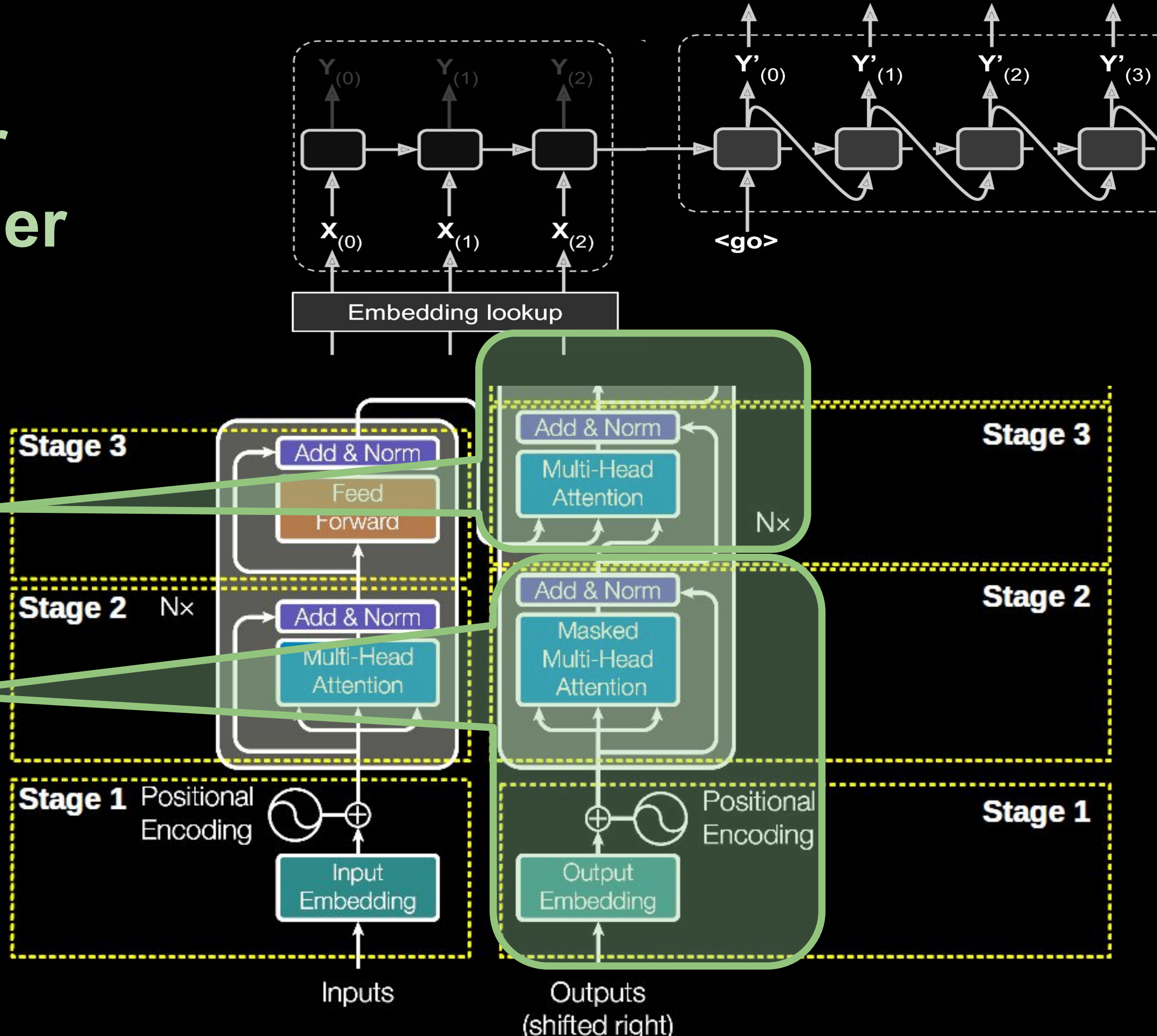


Decoder blocks out future inputs

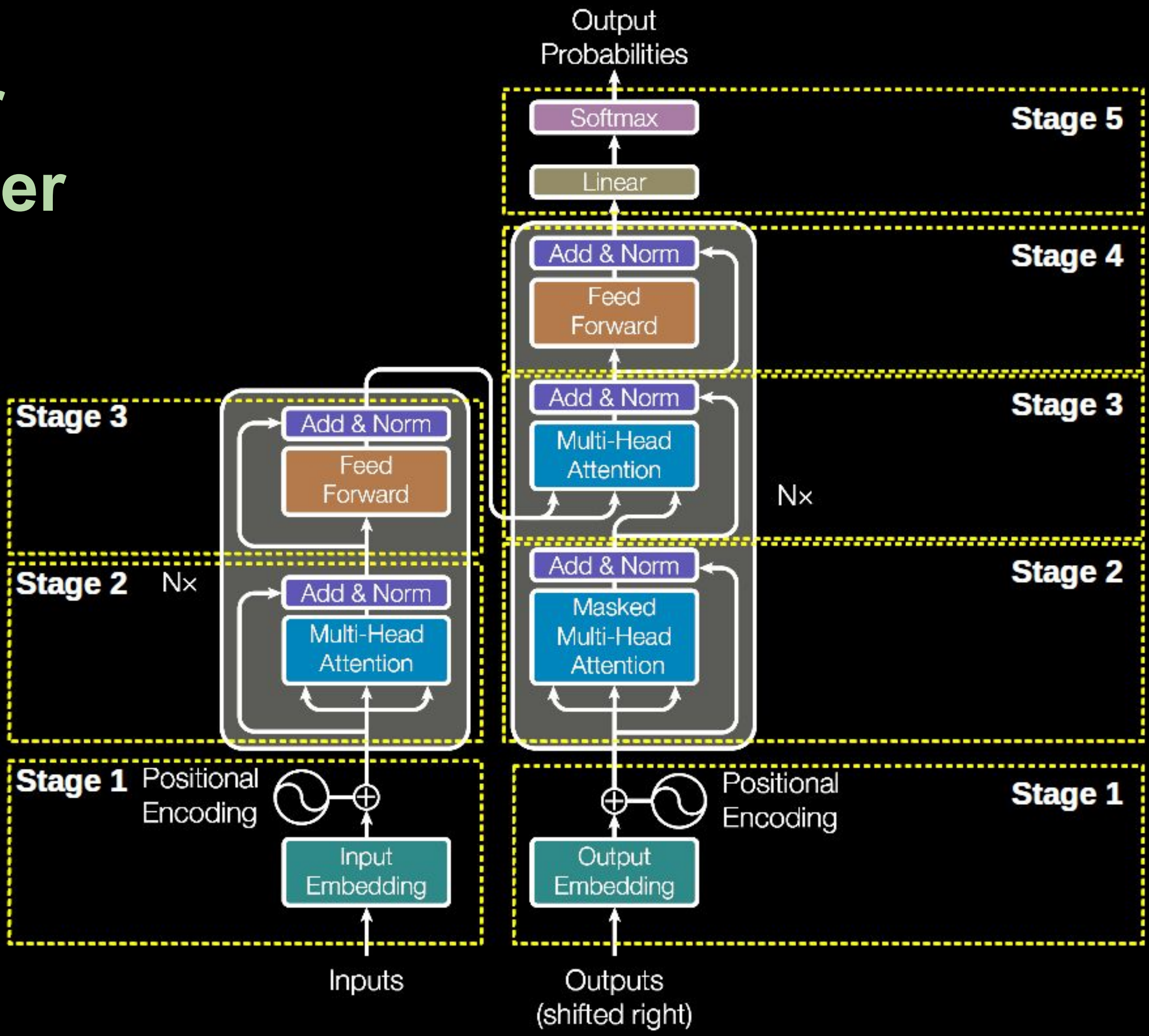
Transformer for Encoder-Decoder

Add conditioning of the LM based on the encoder

essentially, a language model



Transformer for Encoder-Decoder



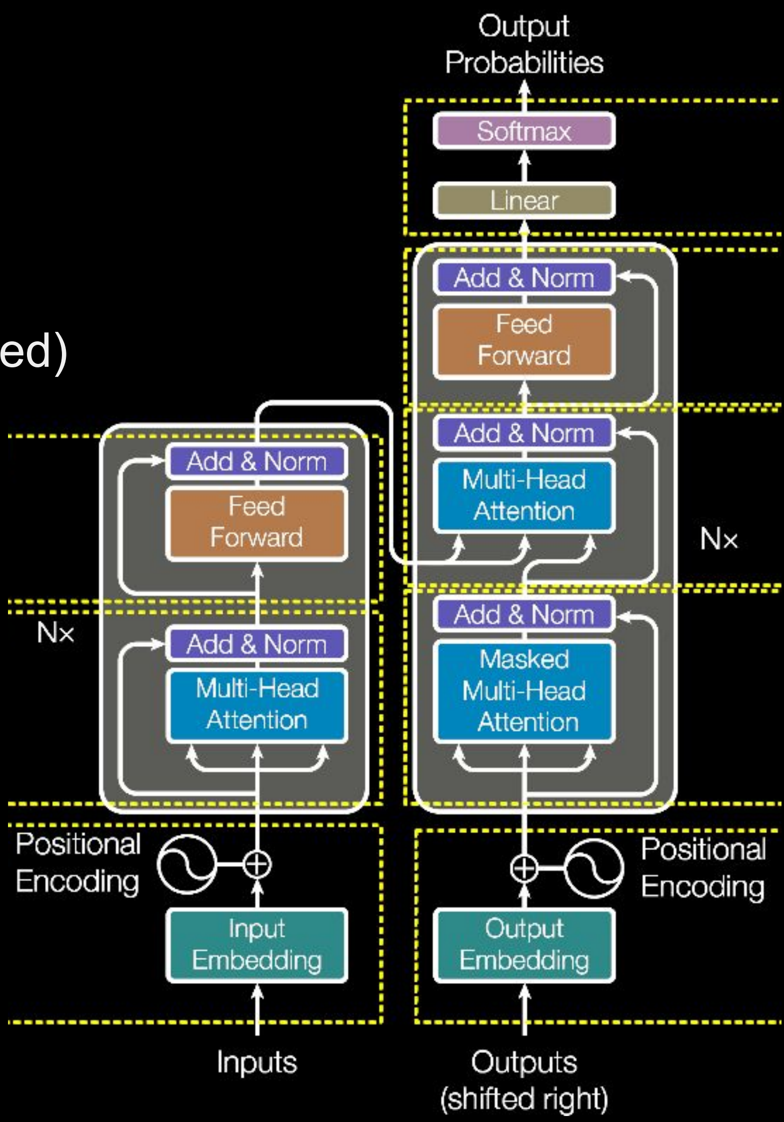
Transformer (as of 2017)

"WMT-2014" Data Set. BLEU scores:

	EN-DE	EN-FR
GNMT (orig)	24.6	39.9
ConvSeq2Seq	25.2	40.5
Transformer*	**28.4**	**41.8**

Transformer

- Utilize Self-Attention
- Simple att scoring function (dot product, scaled)
- Added linear layers for Q, K, and V
- Multi-head attention
- Added positional encoding
- Added residual connection
- Simulate decoding by masking



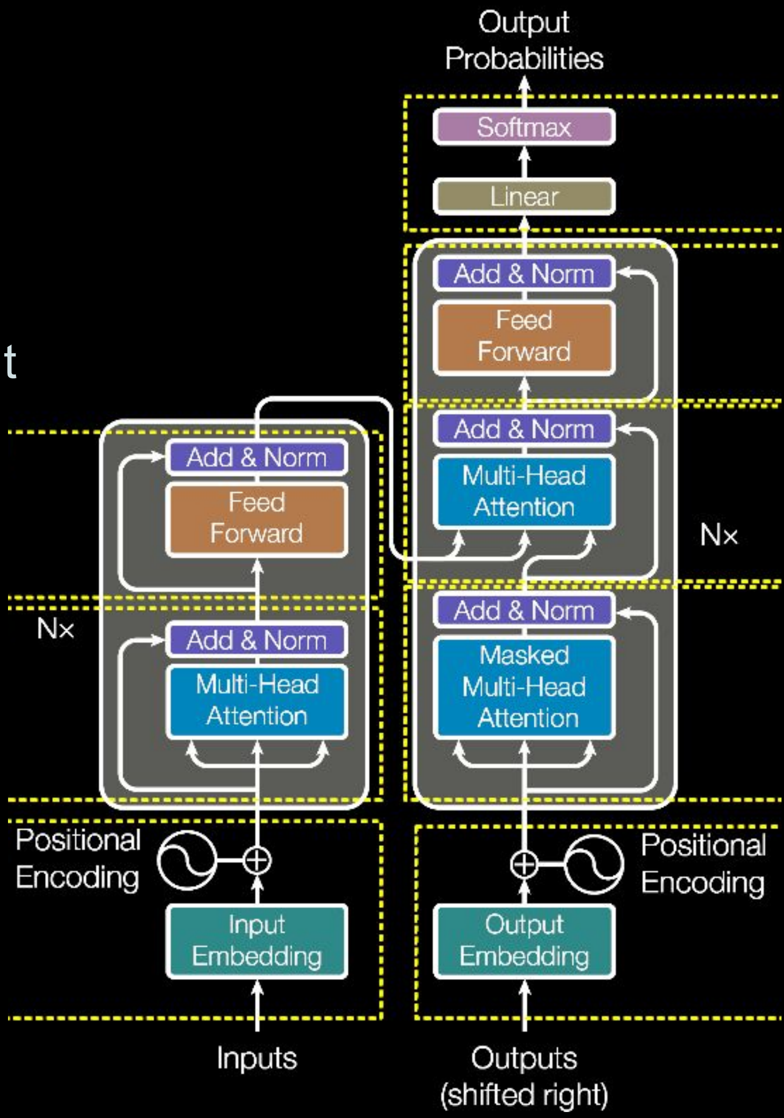
Transformer

Why?

- Don’t need complexity of LSTM/GRU cells
- Constant num edges between words (or input steps)
- Enables “interactions” (i.e. adaptations) between words
- Easy to parallelize -- don’t need sequential processing.

Drawbacks:

- Only unidirectional by default
- Only a “single-hop” relationship per layer (multiple layers to capture multiple)



BERT

Bidirectional **E**ncoder **R**epresentations from **T**ransformers

Produces contextualized embeddings
(or pre-trained contextualized encoder)

Drawbacks of Vanilla Transformers:

- Only unidirectional by default
- Only a “single-hop” relationship per layer
  (multiple layers to capture multiple)

BERT

Bidirectional **E**ncoder **R**epresentations from **T**ransformers

Produces contextualized embeddings
(or pre-trained contextualized encoder)

- Bidirectional context by “masking” in the middle
- A lot of layers, hidden states, attention heads.

Drawbacks of Vanilla Transformers:

- Only unidirectional by default
- Only a “single-hop” relationship per layer
(multiple layers to capture multiple)

BERT

Bidirectional **E**ncoder **R**epresentations from **T**ransformers

Produces contextualized embeddings
(or pre-trained contextualized encoder)

- **Bidirectional context by “masking” in the middle**
- A lot of layers, hidden states, attention heads.

She saw the man on the hill with the telescope.

She [mask] *the man on the hill* [mask] *the telescope.*

BERT

Bidirectional **E**ncoder **R**epresentations from **T**ransformers

Produces contextualized embeddings
(or pre-trained contextualized encoder)

- **Bidirectional context by "masking" in the middle**
- A lot of layers, hidden states, attention heads.

She saw the man on the hill with the telescope.

She [mask] *the man on the hill* [mask] *the telescope.*

Mask 1 in 7 words:

- Too few: expensive, less robust
- Too many: not enough context

BERT

Bidirectional **E**ncoder **R**epresentations from **T**ransformers

Produces contextualized embeddings
(or pre-trained contextualized encoder)

- Bidirectional context by “masking” in the middle
- **A lot of layers, hidden states, attention heads.**
    - `BERT-Base, Cased`:
      12-layer, 768-hidden, 12-heads , 110M parameters

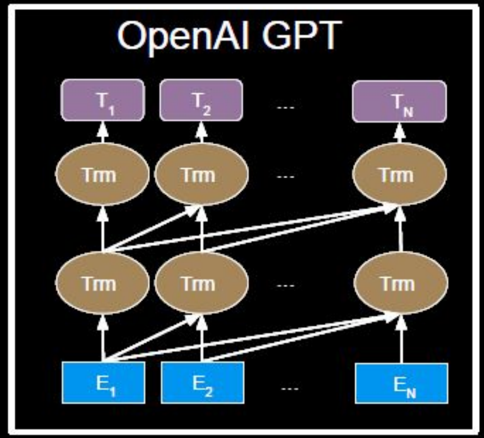
BERT

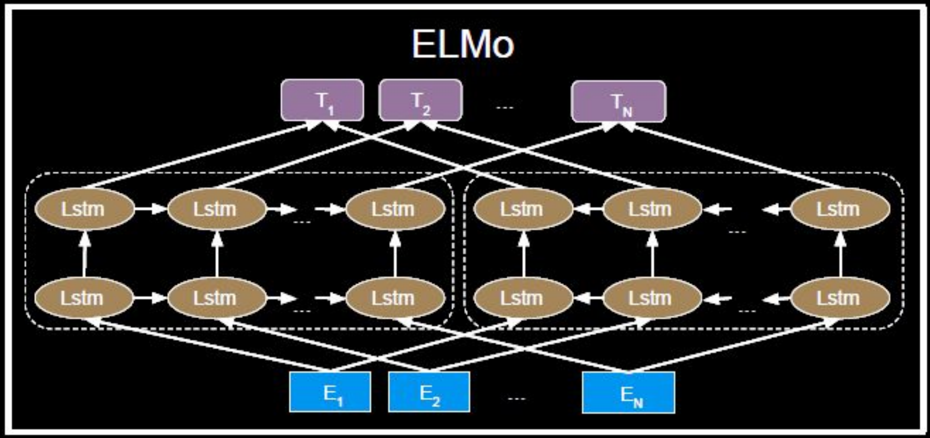
Bidirectional **E**ncoder **R**epresentations from **T**ransformers

Produces contextualized embeddings
(or pre-trained contextualized encoder)

- Bidirectional context by “masking” in the middle
- **A lot of layers, hidden states, attention heads.**
    - `BERT-Base, Cased`:
      12-layer, 768-hidden, 12-heads , 110M parameters
    - `BERT-Large, Cased`:
      24-layer, 1024-hidden, 16-heads, 340M parameters
    - `BERT-Base, Multilingual Cased`:
      104 languages, 12-layer, 768-hidden, 12-heads, 110M parameters

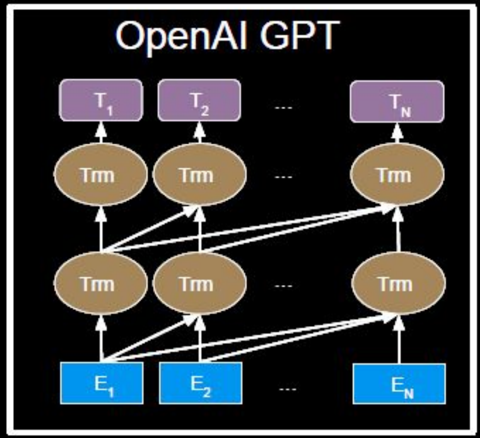
BERT

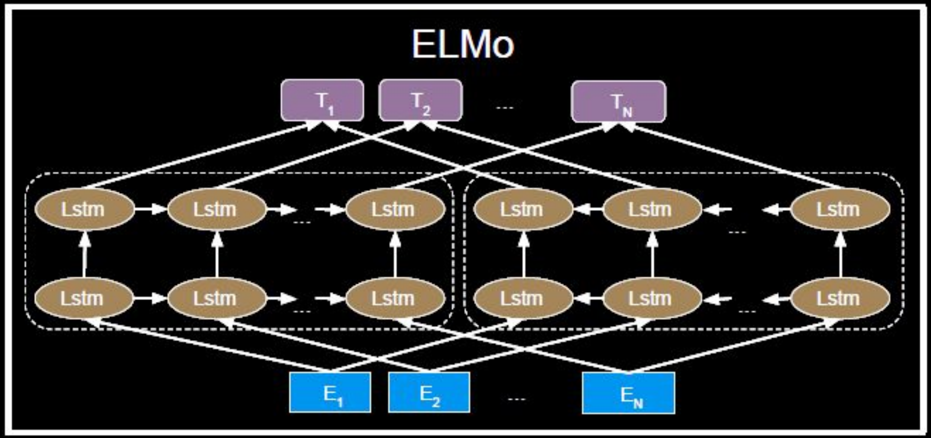


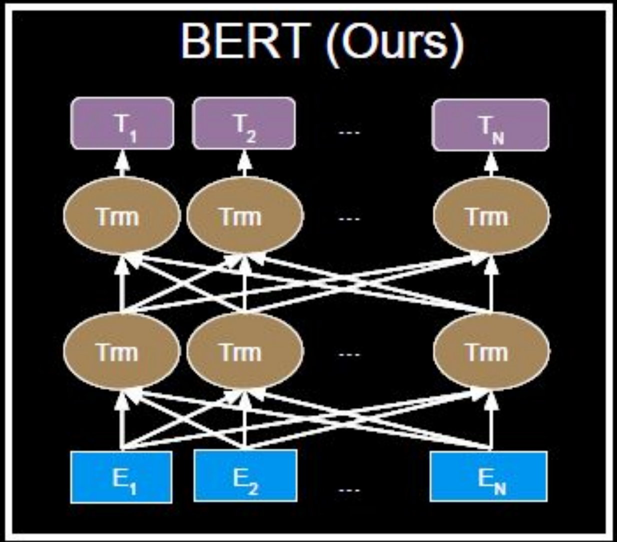


(Devlin et al., 2019)

BERT





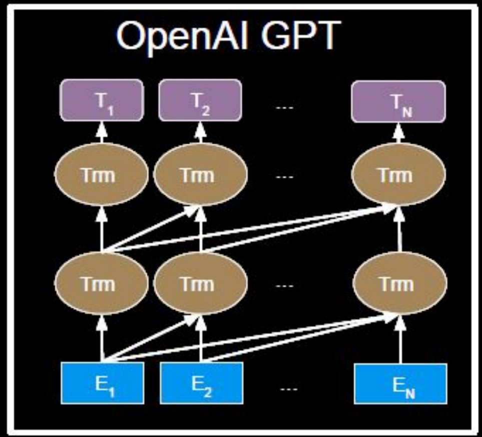


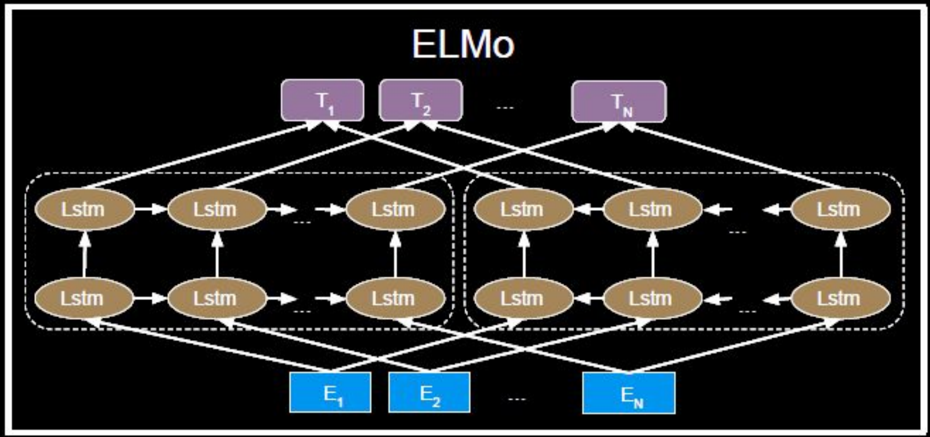
Differences from previous state of the art:

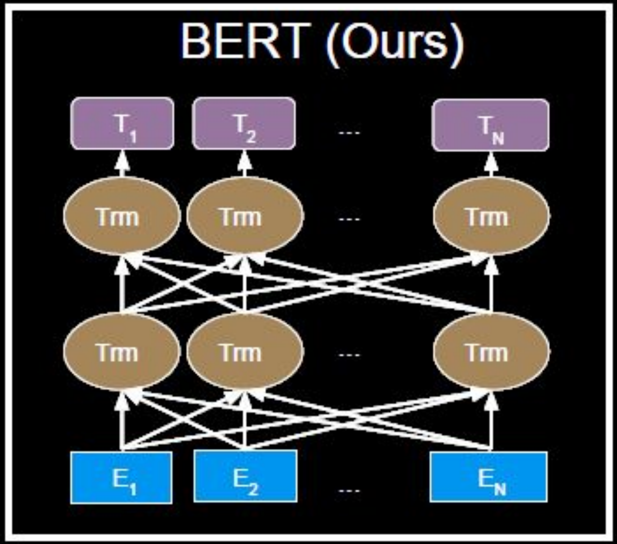
- Bidirectional transformer (through masking)
- Directions jointly trained at once.

(Devlin et al., 2019)

BERT







Differences from previous state of the art:

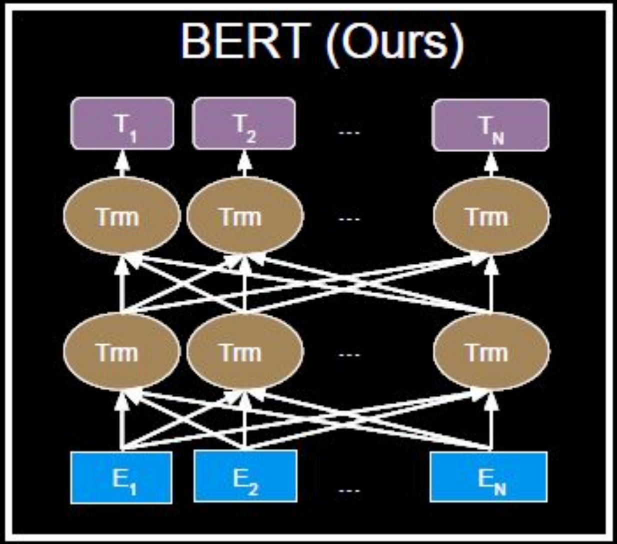
- Bidirectional transformer (through masking)
- Directions jointly trained at once.
- **Capture sentence-level relations**

(Devlin et al., 2019)

BERT

Sentence A = The man went to the store.
Sentence B = He bought a gallon of milk.
Label = IsNextSentence

Sentence A = The man went to the store.
Sentence B = Penguins are flightless.
Label = NotNextSentence



Differences from previous state of the art:

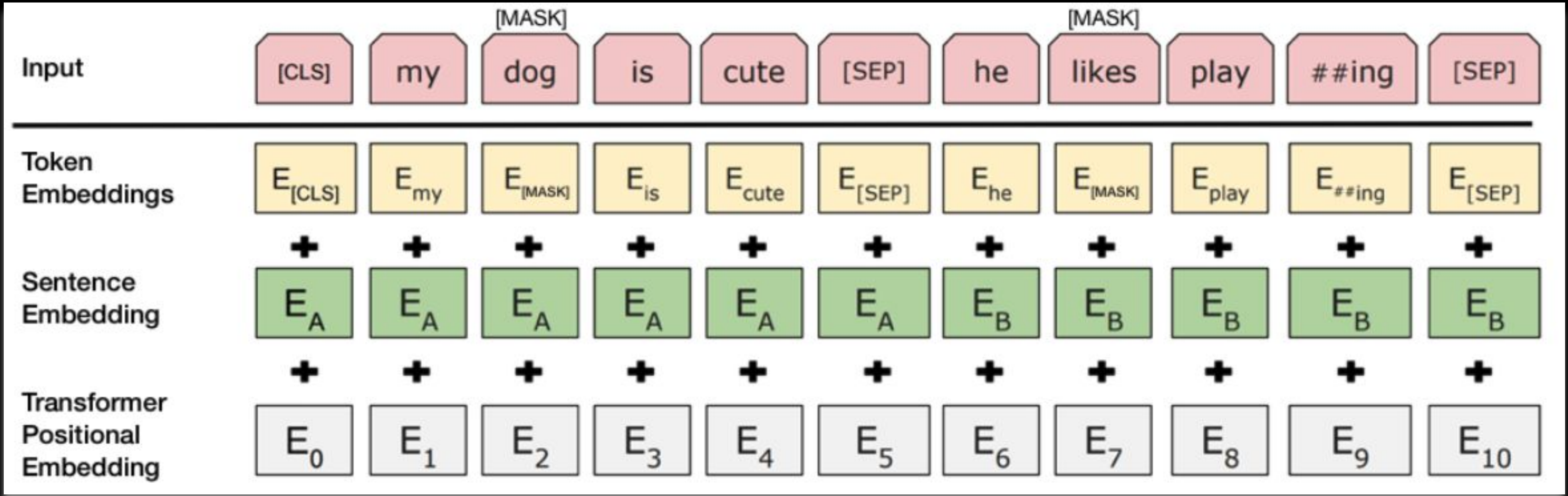
- Bidirectional transformer (through masking)
- Directions jointly trained at once.
- **Capture sentence-level relations**

(Devlin et al., 2019)

BERT

Sentence A = `The man went to the store.`
Sentence B = `He bought a gallon of milk.`
Label = `IsNextSentence`

Sentence A = `The man went to the store.`
Sentence B = `Penguins are flightless.`
Label = `NotNextSentence`

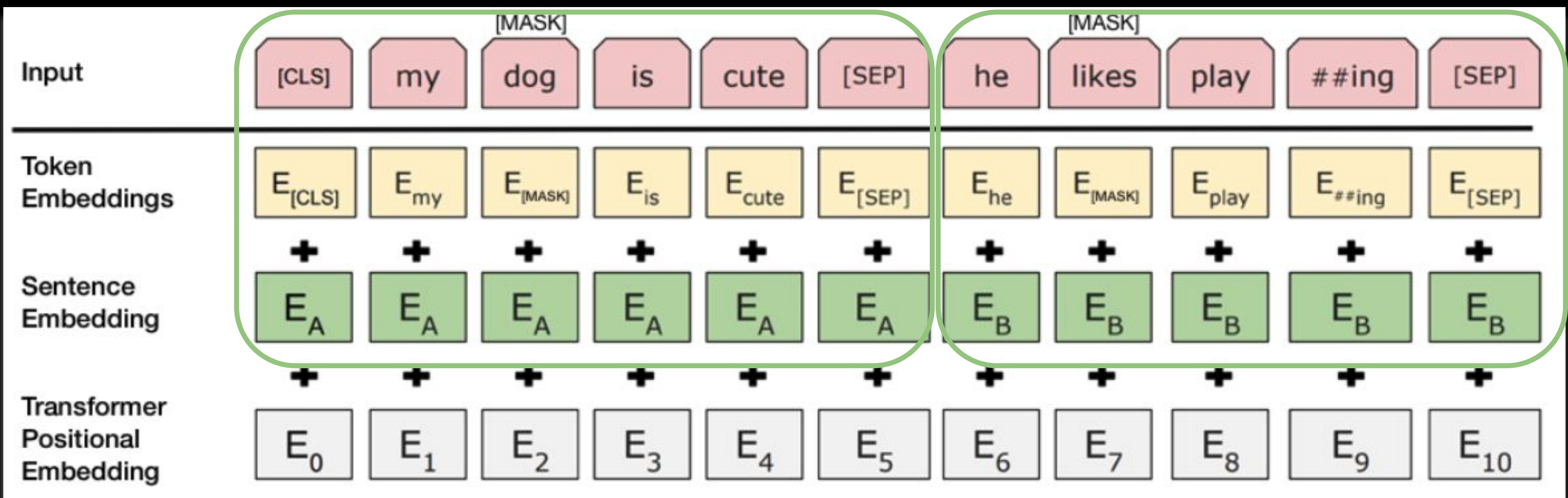


(Devlin et al., 2019)

BERT

Sentence A = The man went to the store.
Sentence B = He bought a gallon of milk.
Label = IsNextSentence

Sentence A = The man went to the store.
Sentence B = Penguins are flightless.
Label = NotNextSentence



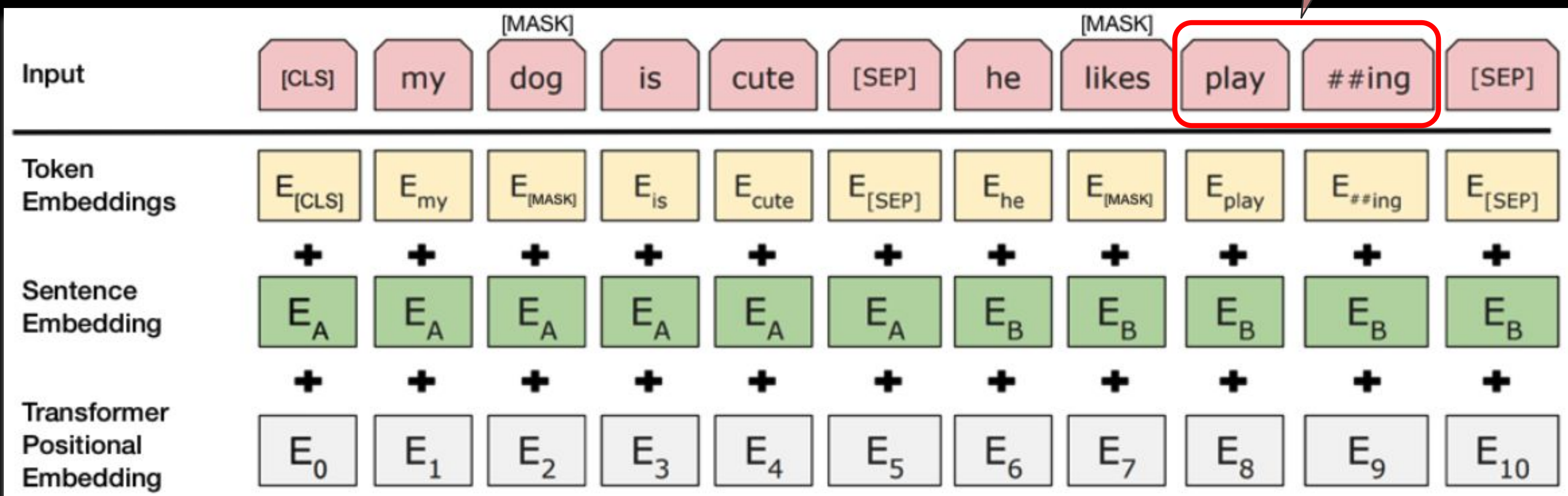
(Devlin et al., 2019)

BERT

tokenize into “word pieces”

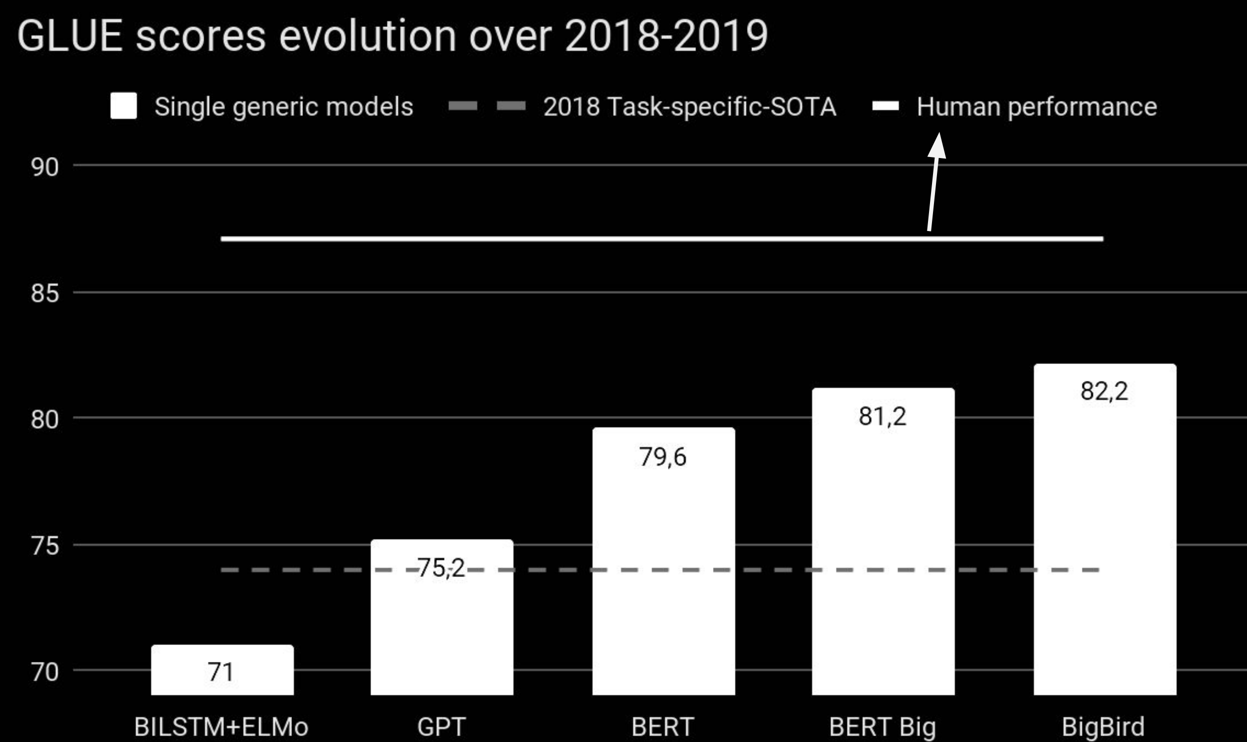
Sentence A = The man went to the store.
Sentence B = He bought a gallon of milk.
Label = IsNextSentence

Sentence A = The man went to the store.
Sentence B = Penguins are flightless.
Label = NotNextSentence



(Devlin et al., 2019)

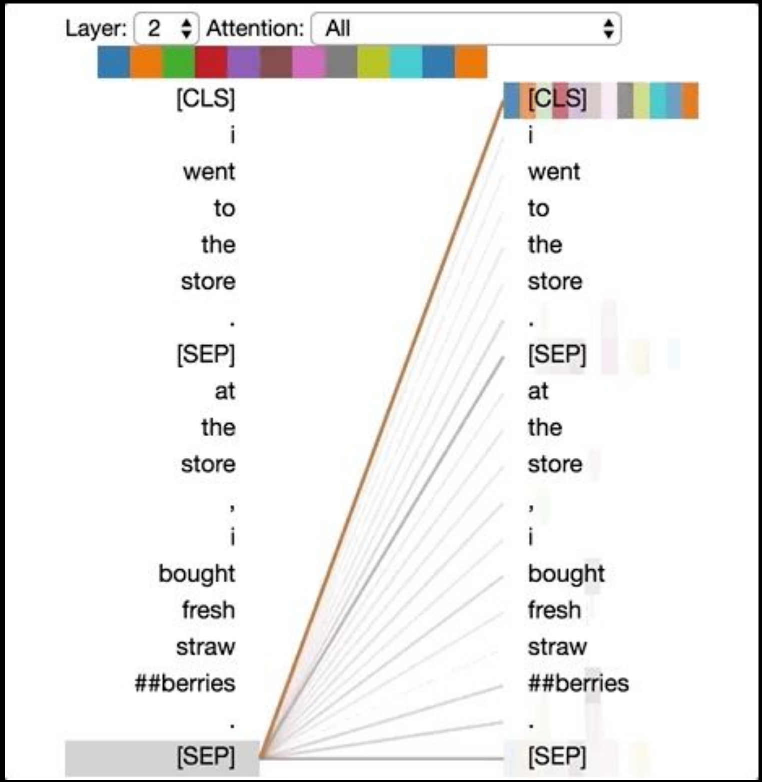
BERT Performance: e.g. Question Answering



https://rajpurkar.github.io/SQuAD-explorer/

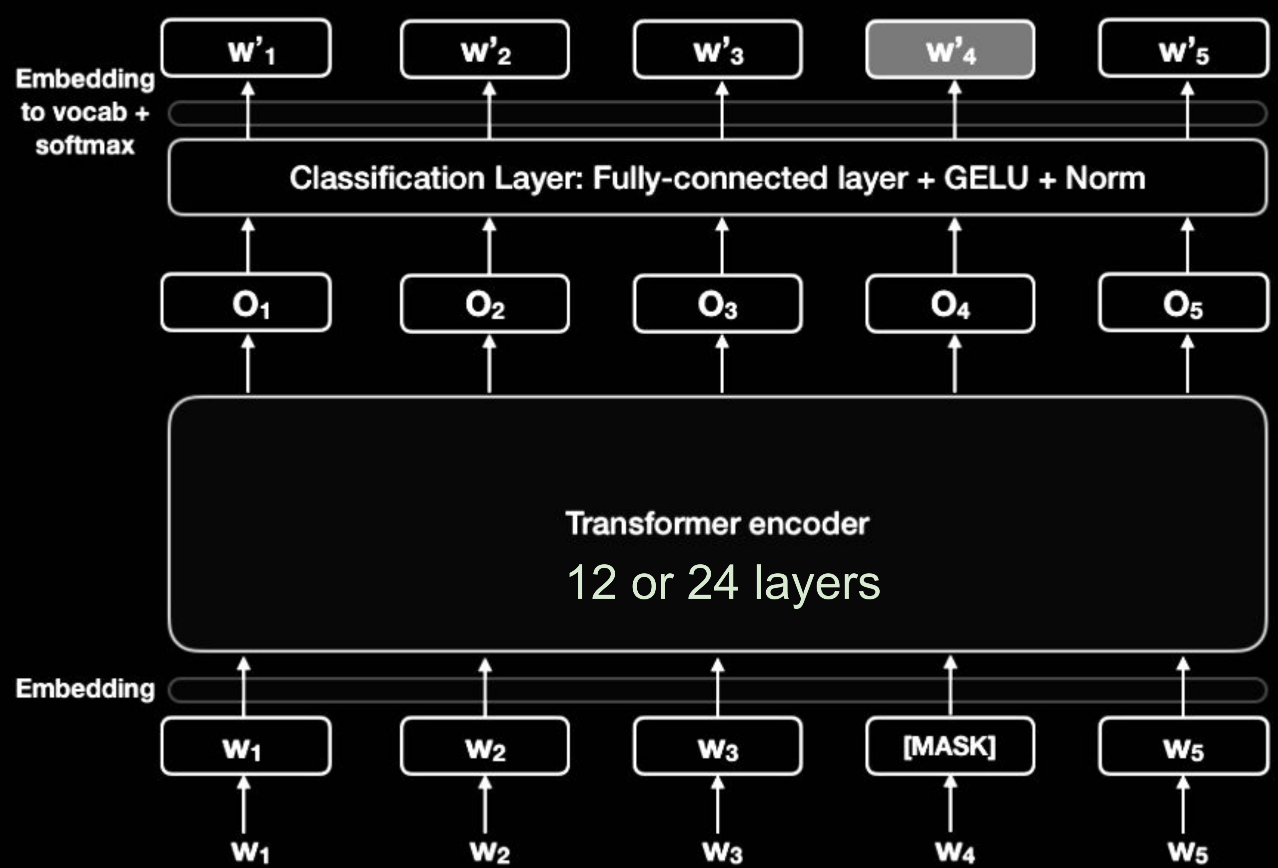
Bert: Attention by Layers

https://colab.research.google.com/drive/1vlOJ1IhdujVjfH857hvYKIdKPTD9Kid8

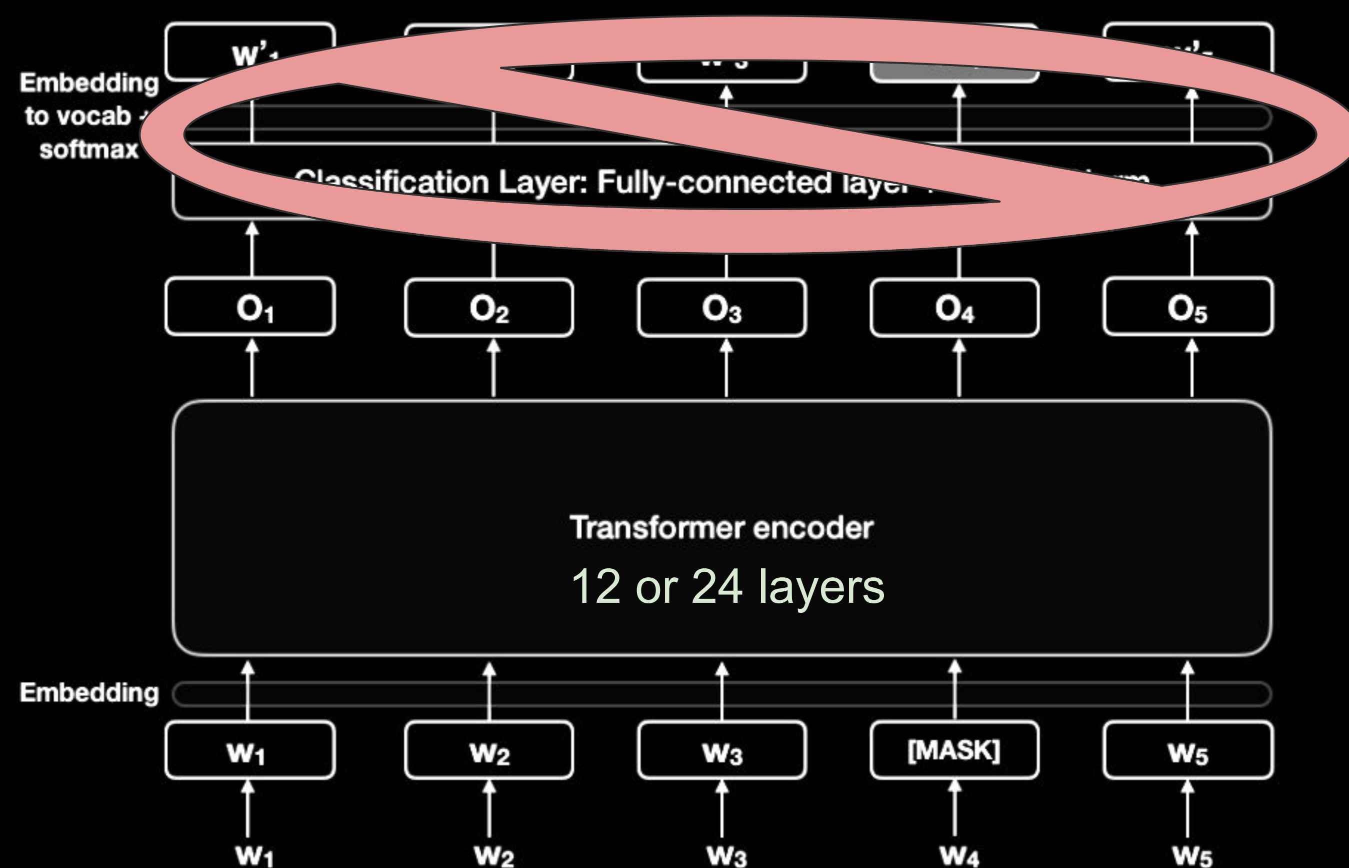


(Vig, 2019)

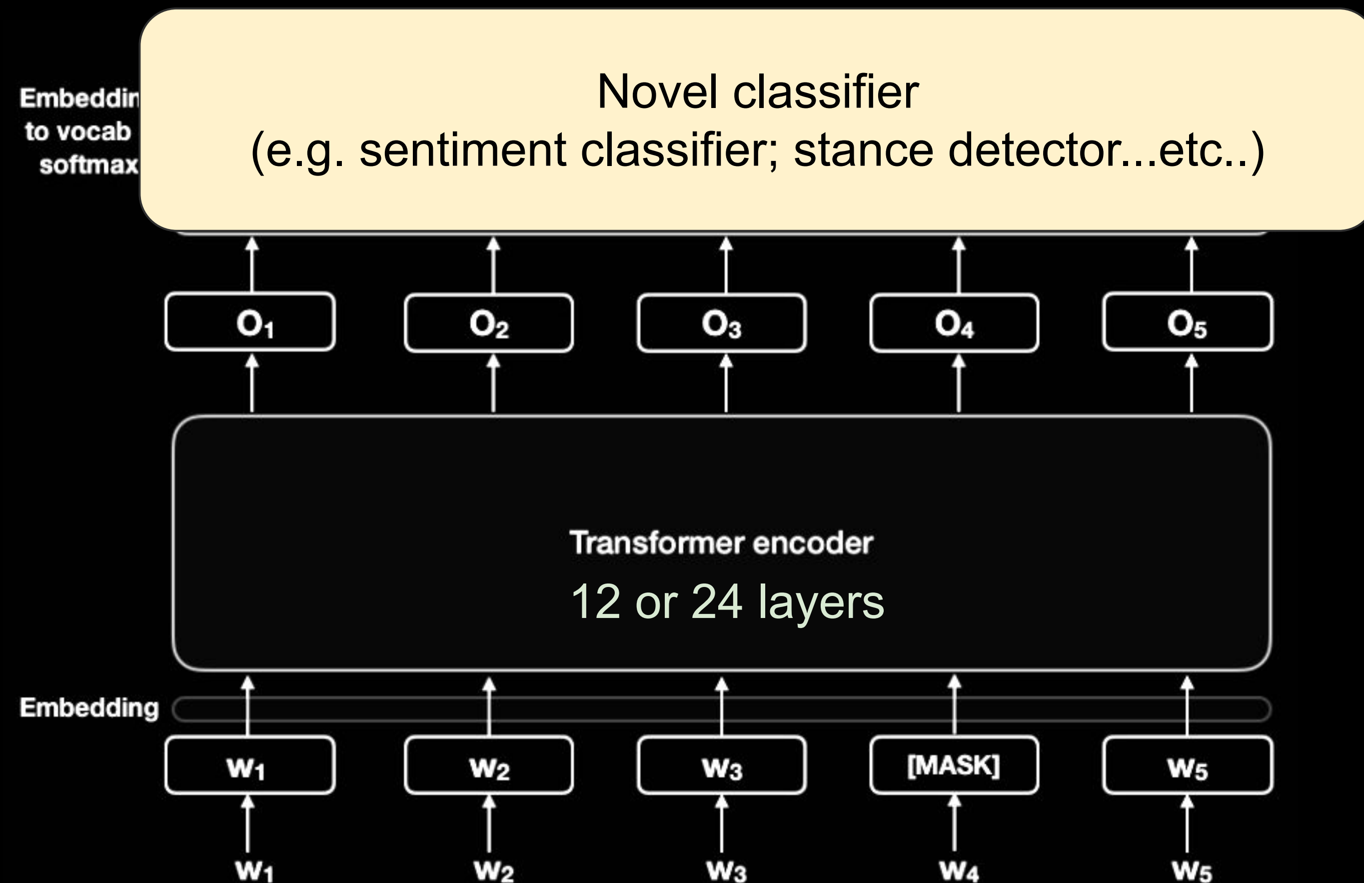
BERT: Pre-training; Fine-tuning



BERT: Pre-training; Fine-tuning

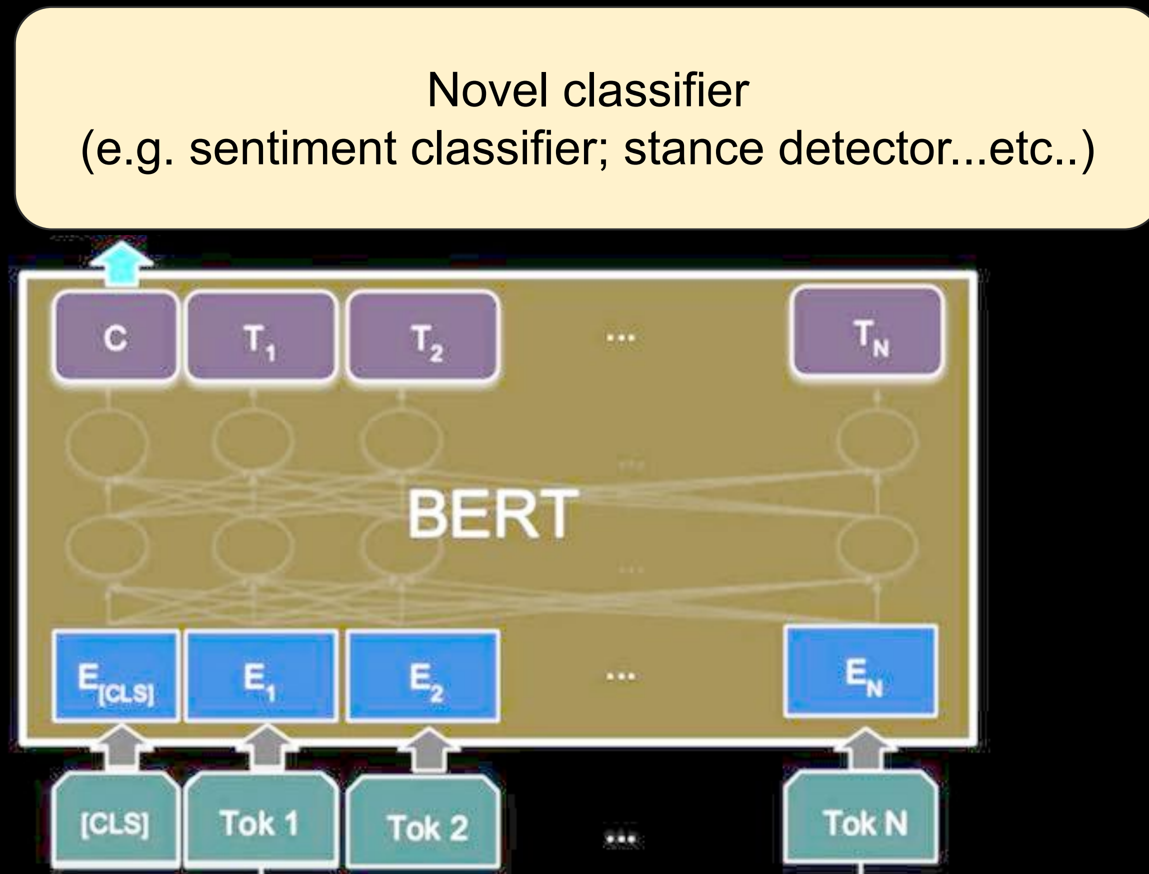


BERT: Pre-training; Fine-tuning



BERT: Pre-training; Fine-tuning

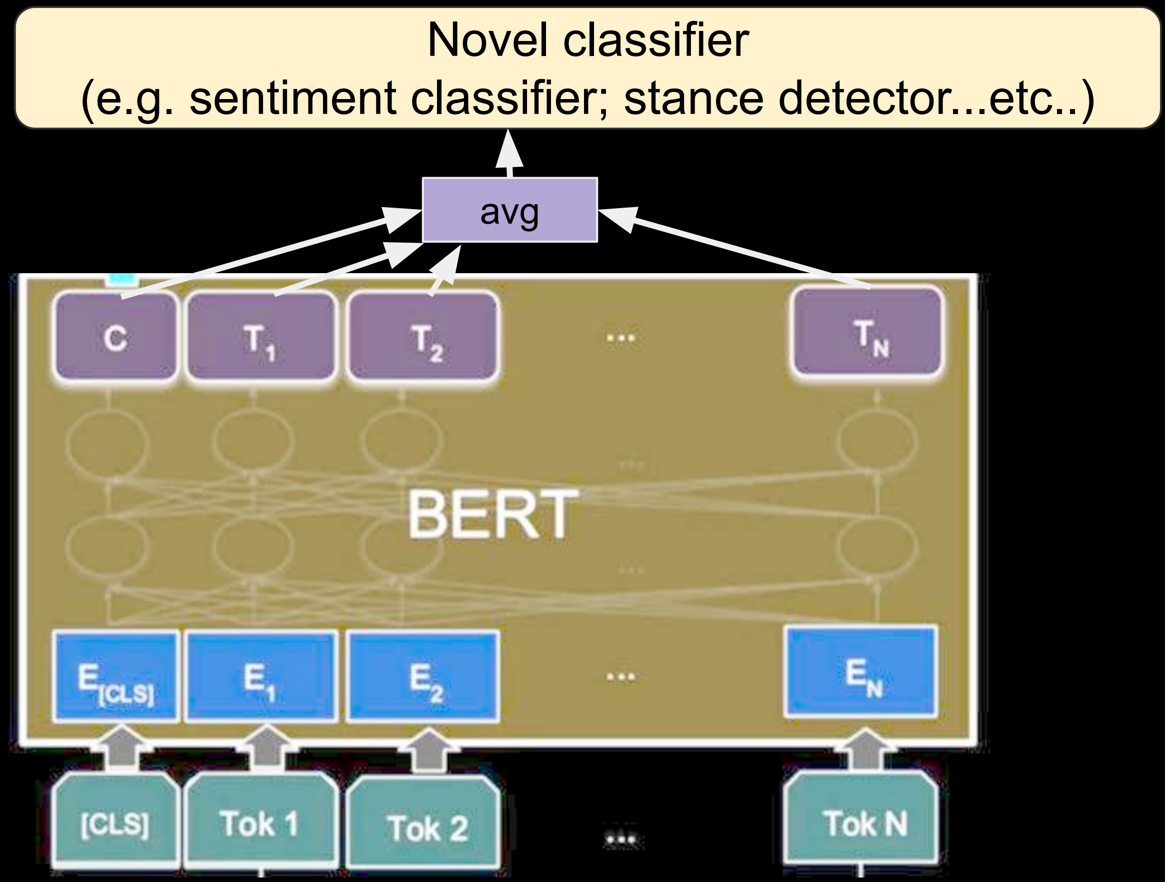
[CLS] vector at start is supposed to capture meaning of whole sequence.



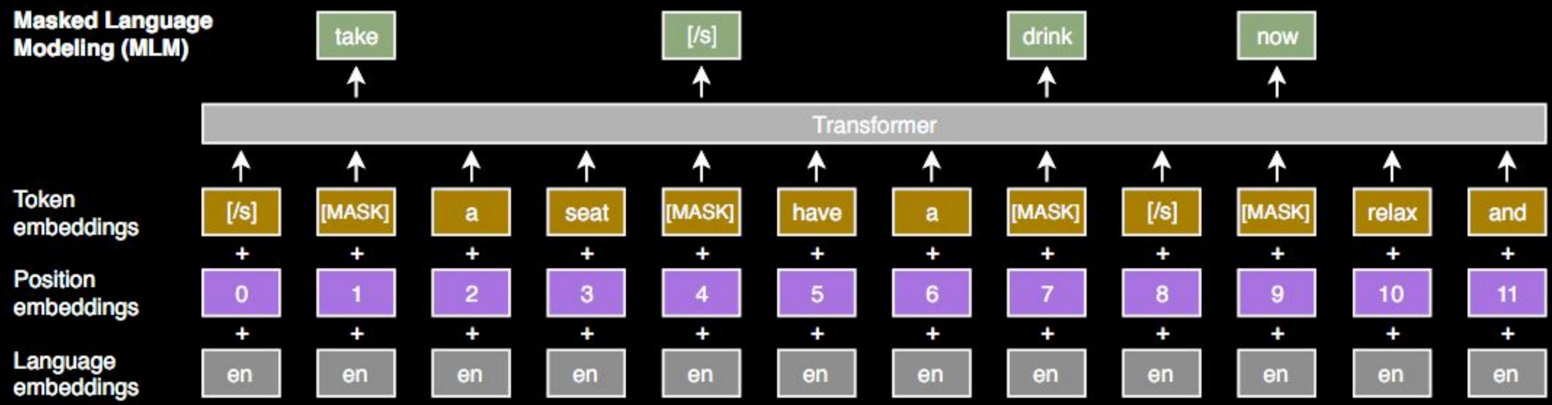
BERT: Pre-training; Fine-tuning

[CLS] vector at start is supposed to capture meaning of whole sequence.

Average of top layer (or second to top) also often used.

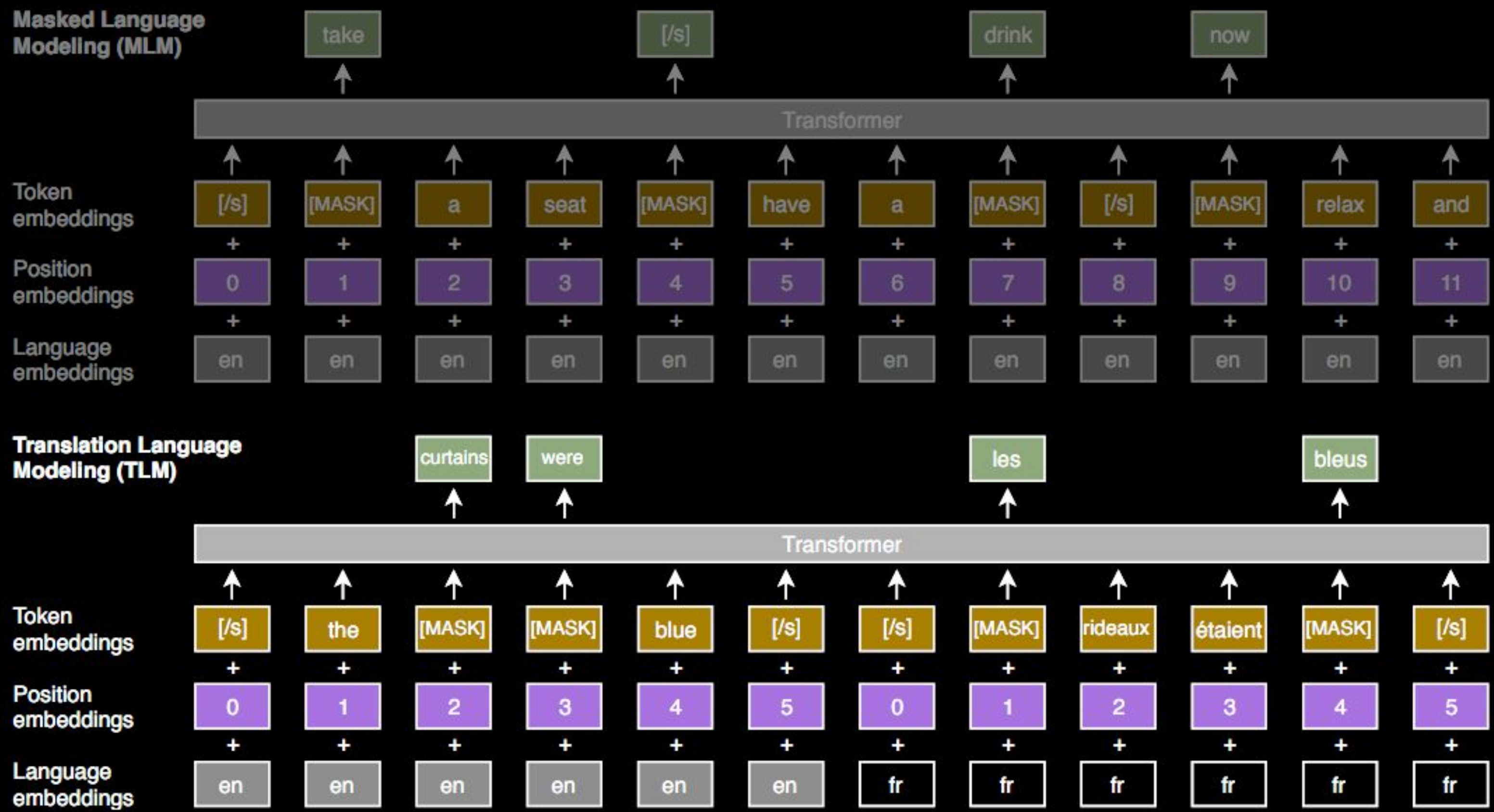


BERT for Machine Translation:



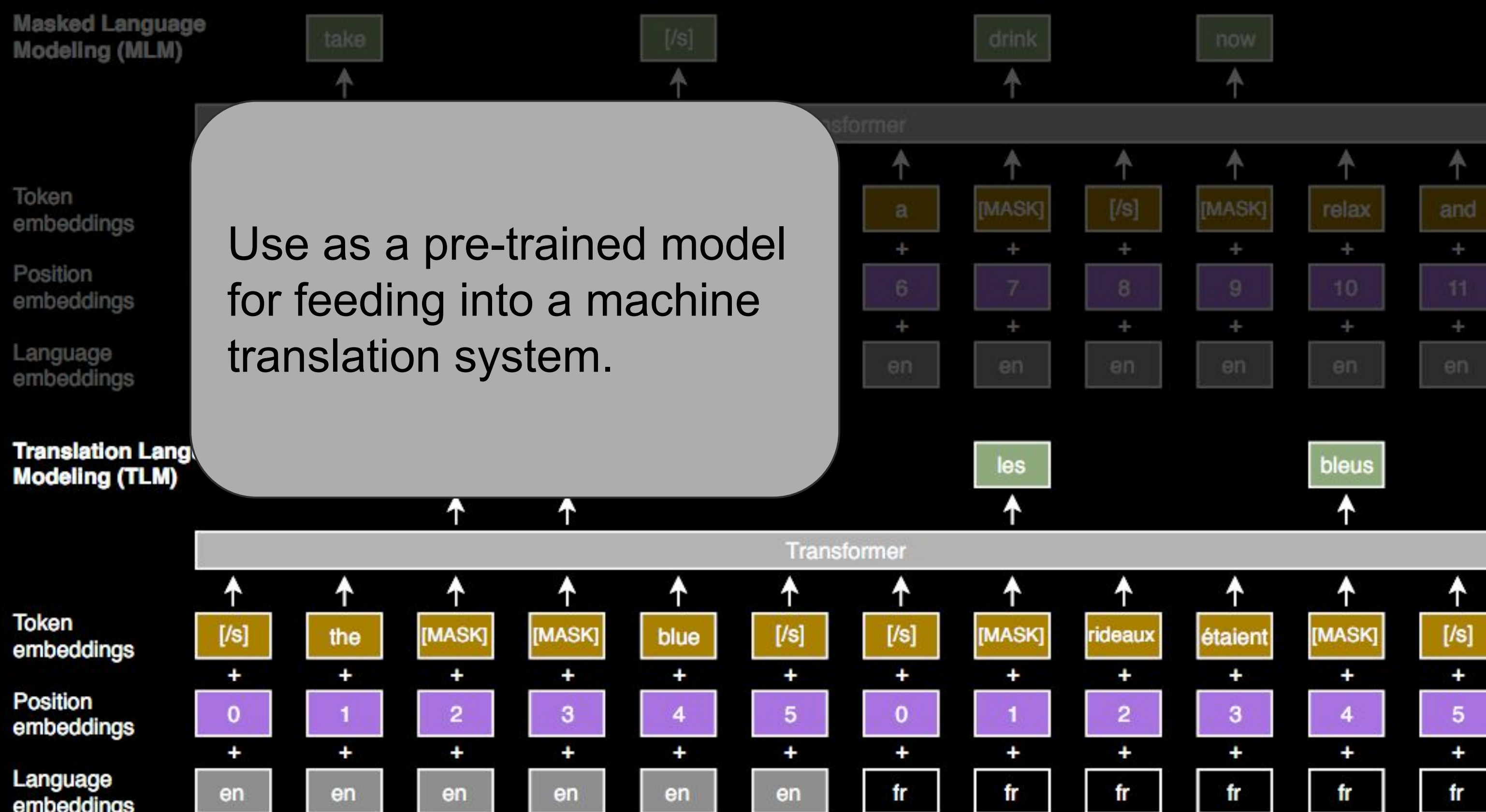
(Lample & Conneau, Facebook, 2019)

BERT for Machine Translation:



(Lample & Conneau, Facebook, 2019)

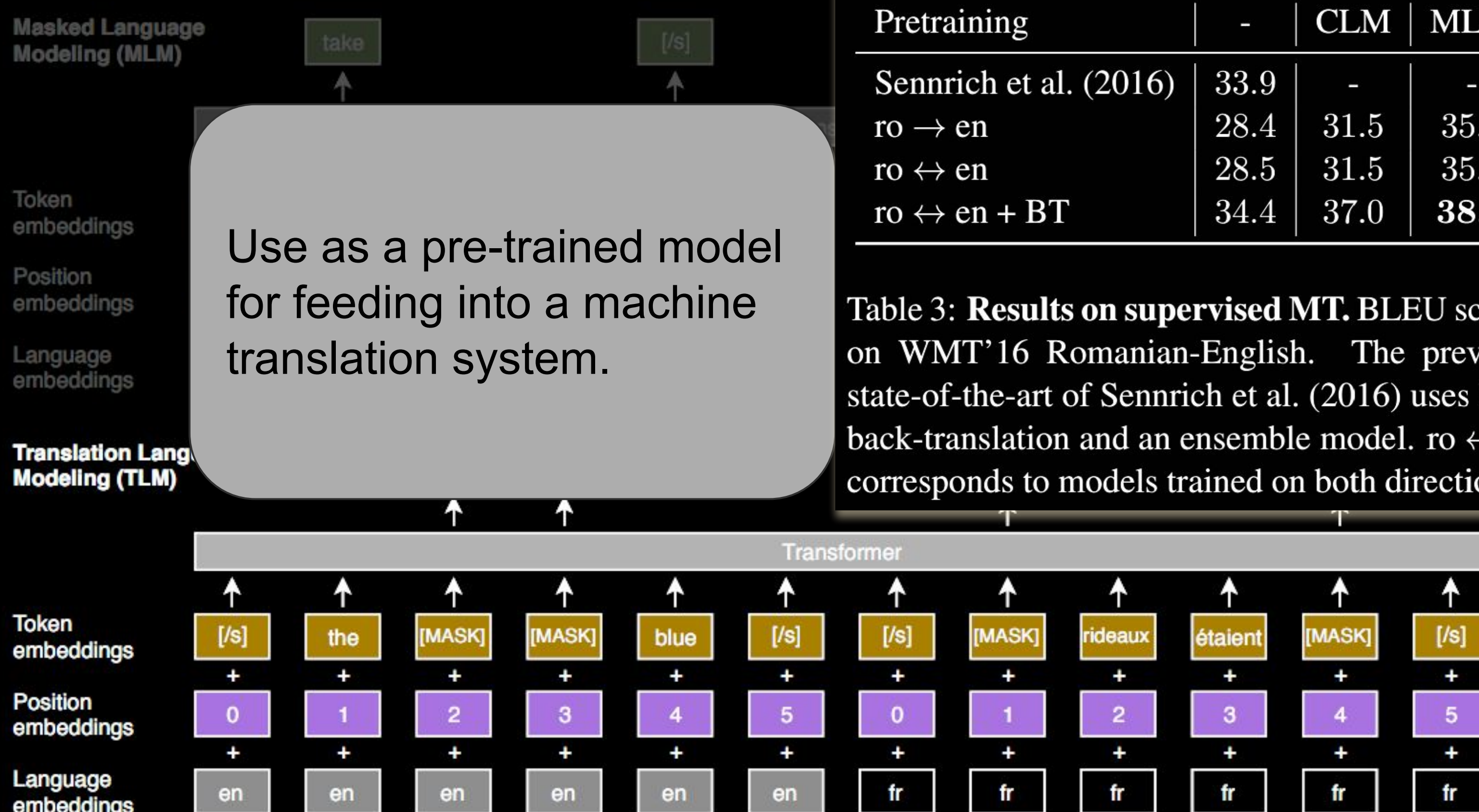
BERT for Machine Translation:



Use as a pre-trained model for feeding into a machine translation system.

(Lample & Conneau, Facebook, 2019)

BERT for Machine Translation:



Use as a pre-trained model for feeding into a machine translation system.

Pretraining	-	CLM	MLM
Sennrich et al. (2016)	33.9	-	-
ro $\rightarrow$ en	28.4	31.5	35.3
ro $\leftrightarrow$ en	28.5	31.5	35.6
ro $\leftrightarrow$ en + BT	34.4	37.0	**38.5**

Table 3: **Results on supervised MT.** BLEU scores on WMT'16 Romanian-English. The previous state-of-the-art of Sennrich et al. (2016) uses both back-translation and an ensemble model. ro $\leftrightarrow$ en corresponds to models trained on both directions.

(Lample & Conneau, Facebook, 2019)

Neural Machine Translation

Where does neural approach fall short? (Manning, 2018)

- Translation process is mostly a black box -- can’t answer “why” for reordering, word choice decisions
- No direct use of semantic or syntactic structures
- Not modeling discourse structure -- only rough sense of how sentences relate to each other. Doesn’t model long distance anaphora.