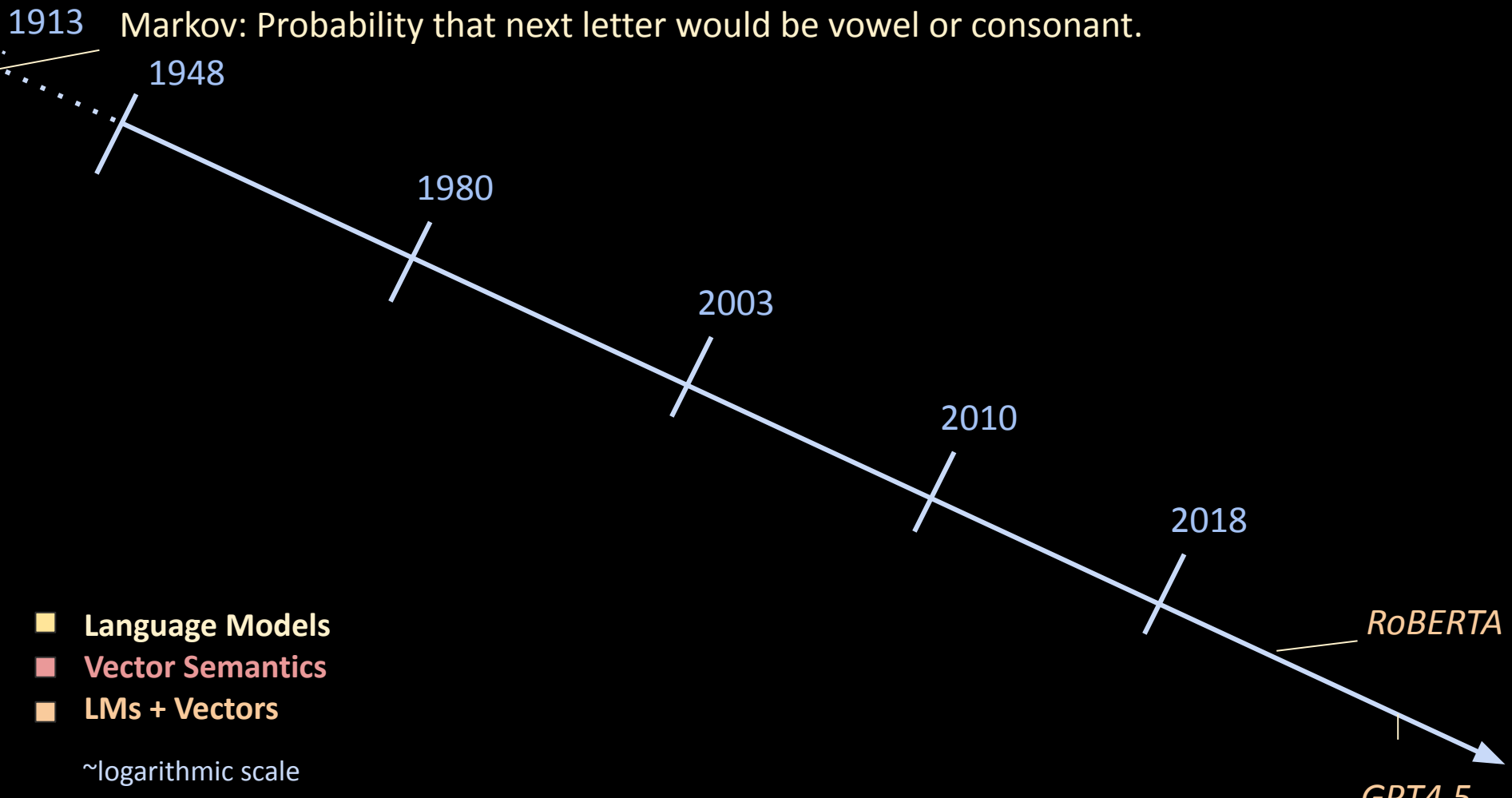
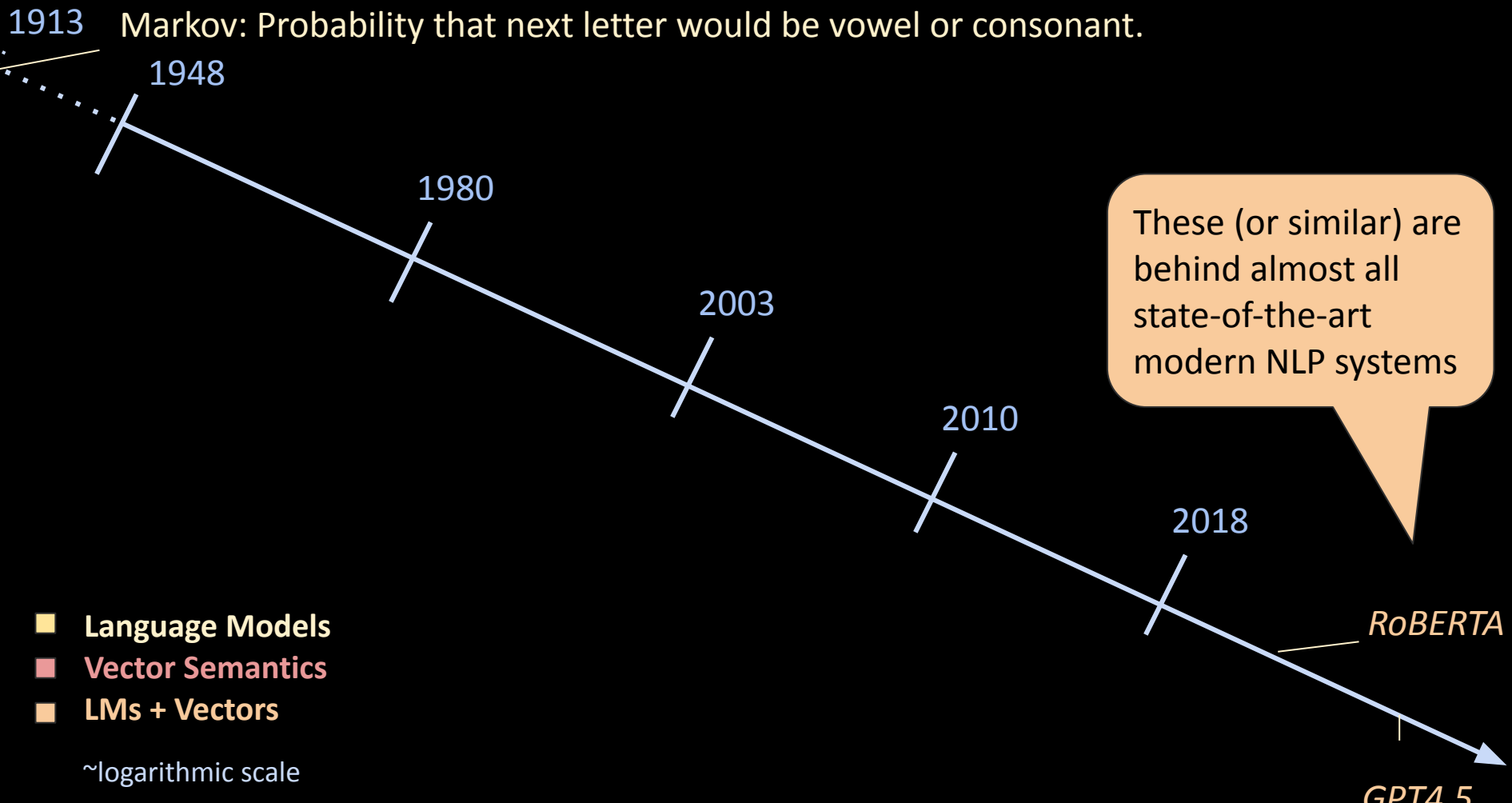
Attention & Transformer LMs

CSE538 - Spring 2025

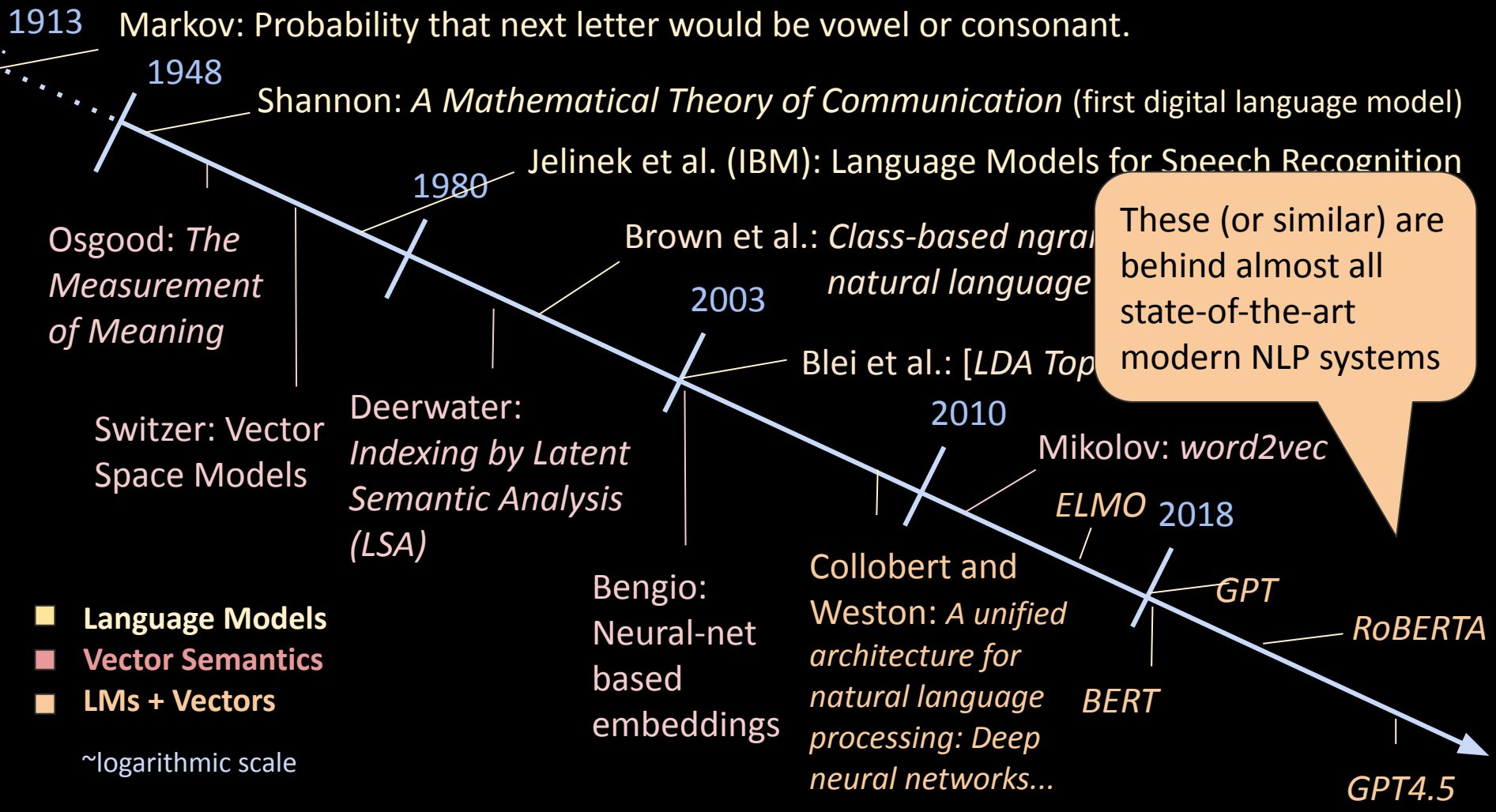
Timeline: *Language Modeling* and *Vector Semantics*



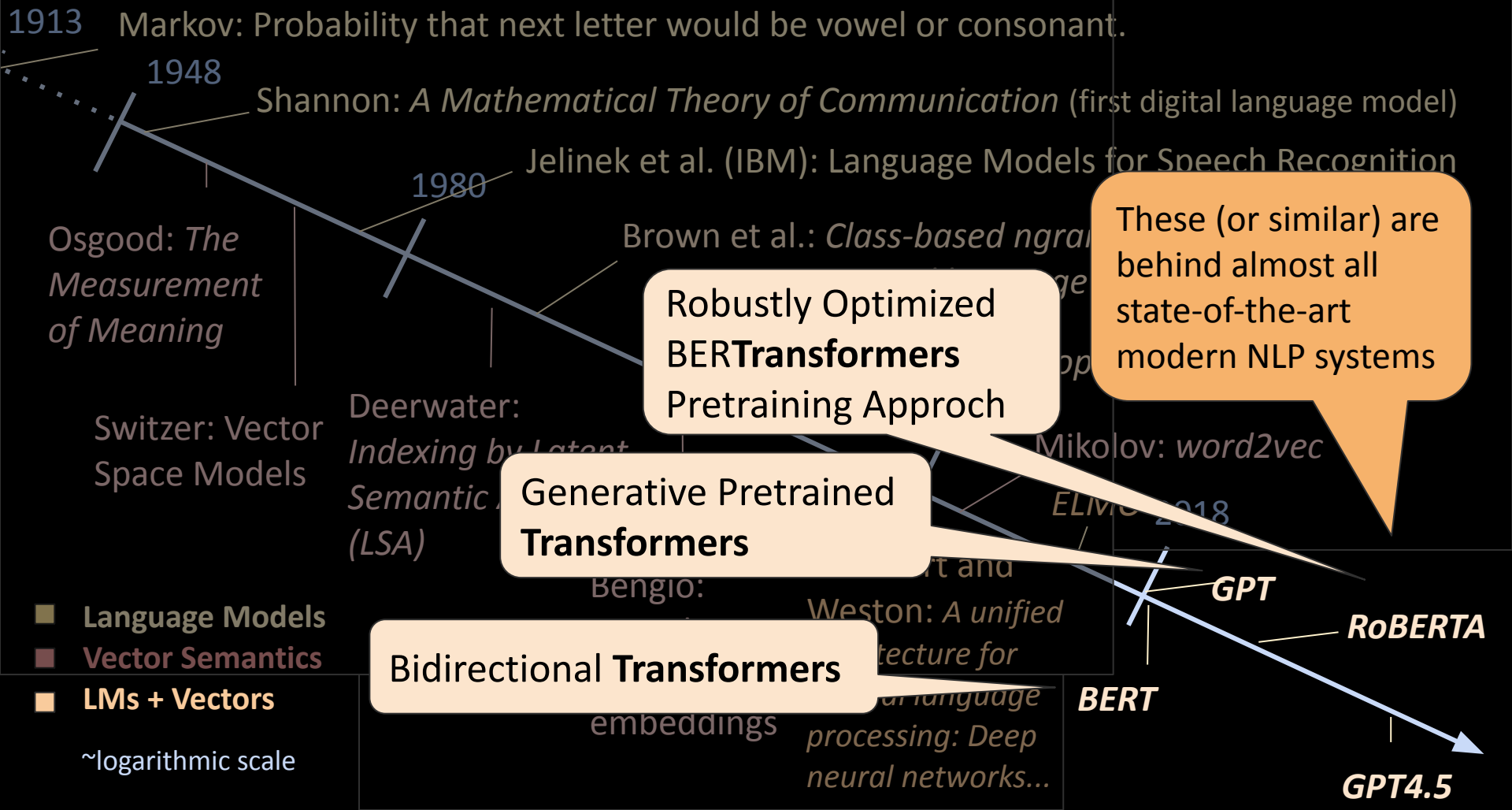
Timeline: *Language Modeling* and *Vector Semantics*



Timeline: *Language Modeling* and *Vector Semantics*

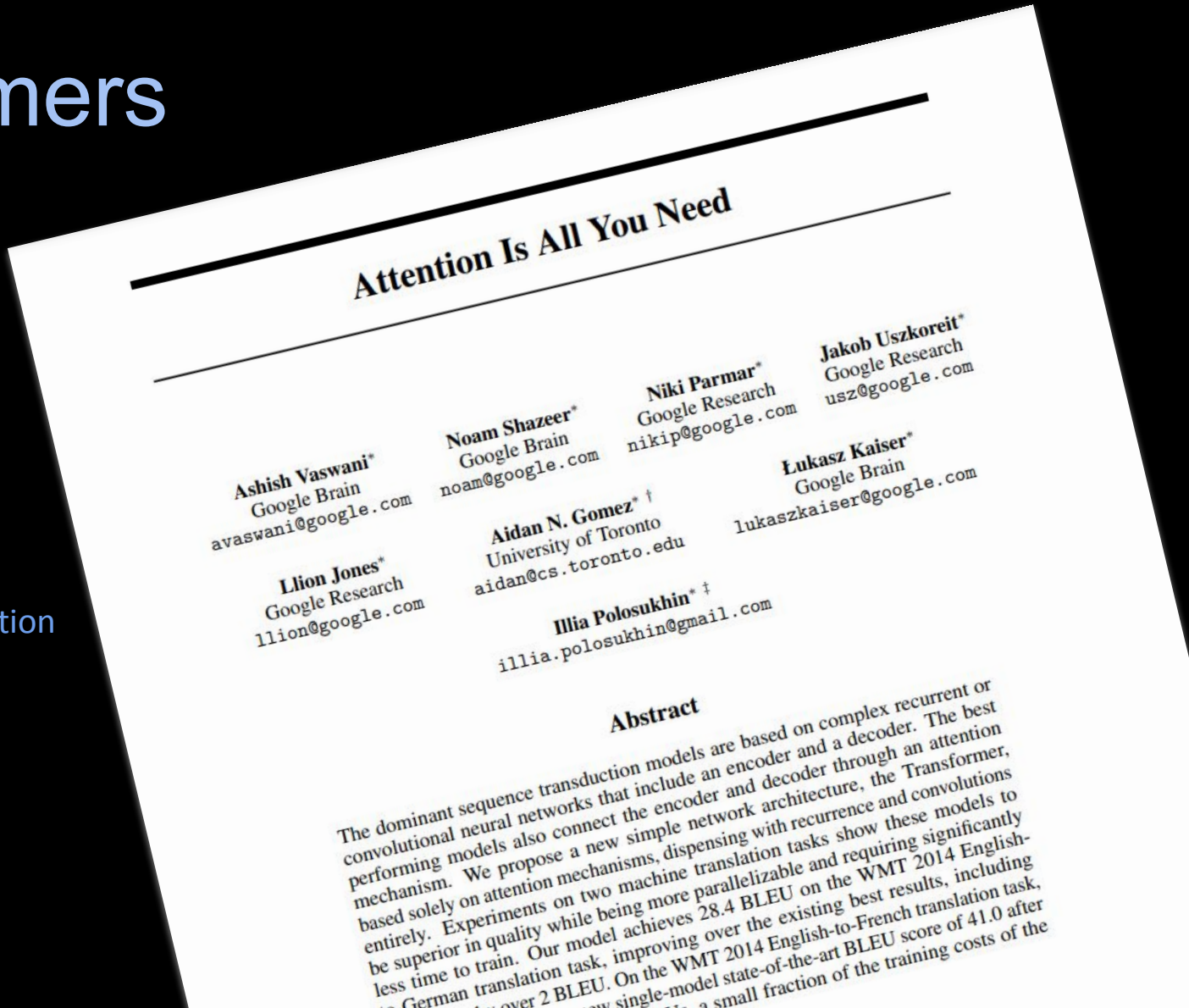


Timeline: *Language Modeling* and *Vector Semantics*



Transformers

(Advances in neural information processing systems, 2017)



Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

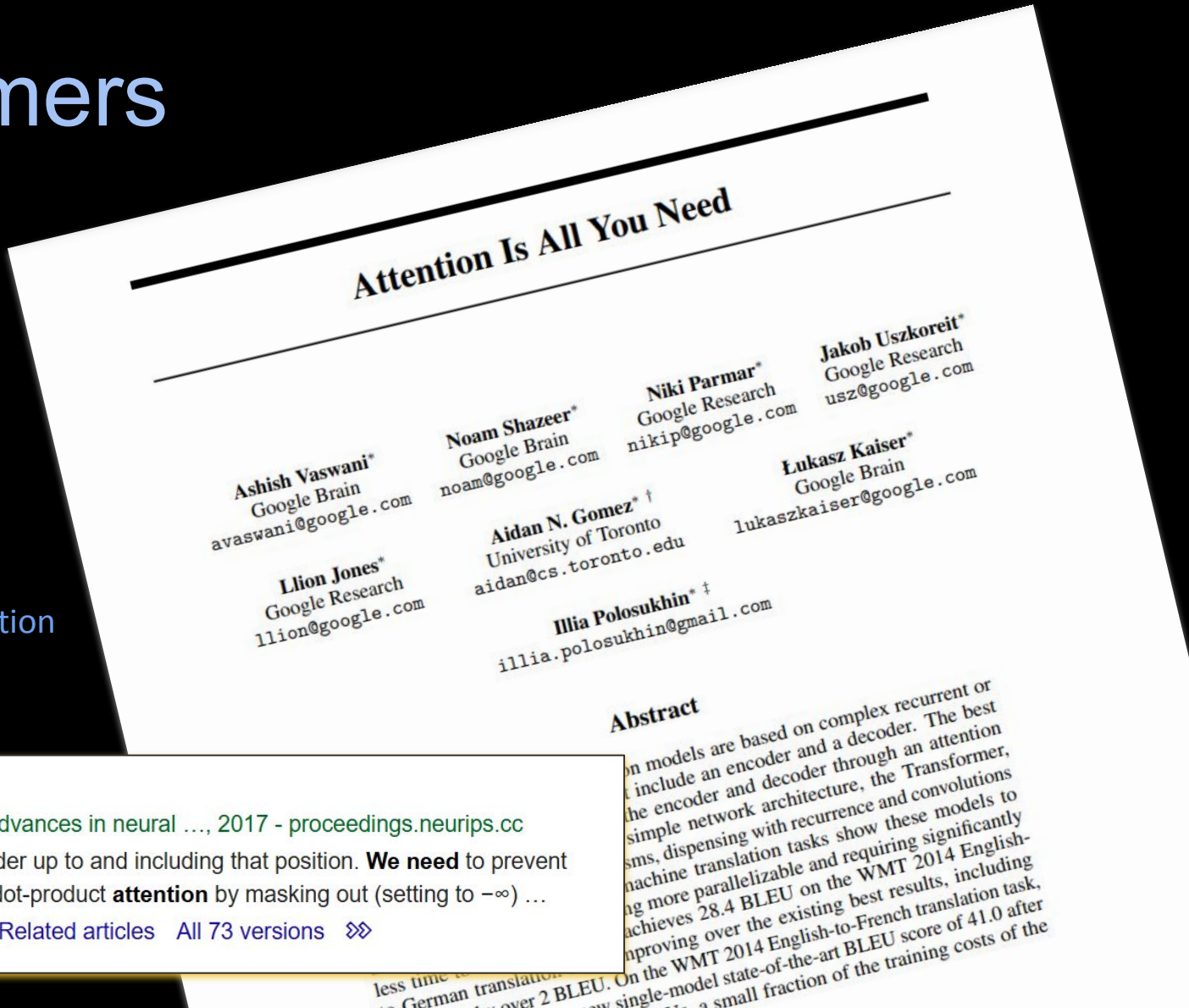
Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including over 2 BLEU. On the WMT 2014 English-to-French translation task, our single-model state-of-the-art BLEU score of 41.0 after a small fraction of the training costs of the

Transformers

(Advances in neural information processing systems, 2017)



Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Llion Jones* Google Research llion@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

…n models are based on complex recurrent or …t include an encoder and a decoder. The best …he encoder and decoder through an attention …simple network architecture, the Transformer, …sms, dispensing with recurrence and convolutions …achine translation tasks show these models to …g more parallelizable and requiring significantly …achieves 28.4 BLEU on the WMT 2014 English- …mproving over the existing best results, including …on the WMT 2014 English-to-French translation task, …single-model state-of-the-art BLEU score of 41.0 after …a small fraction of the training costs of the …less time …German translation … over 2 BLEU. On the …

Attention is **all you need**

A Vaswani, N Shazeer, N Parmar… - Advances in neural …, 2017 - proceedings.neurips.cc

… to attend to **all** positions in the decoder up to and including that position. **We need** to prevent … **We** implement this inside of scaled dot-product **attention** by masking out (setting to −∞) …

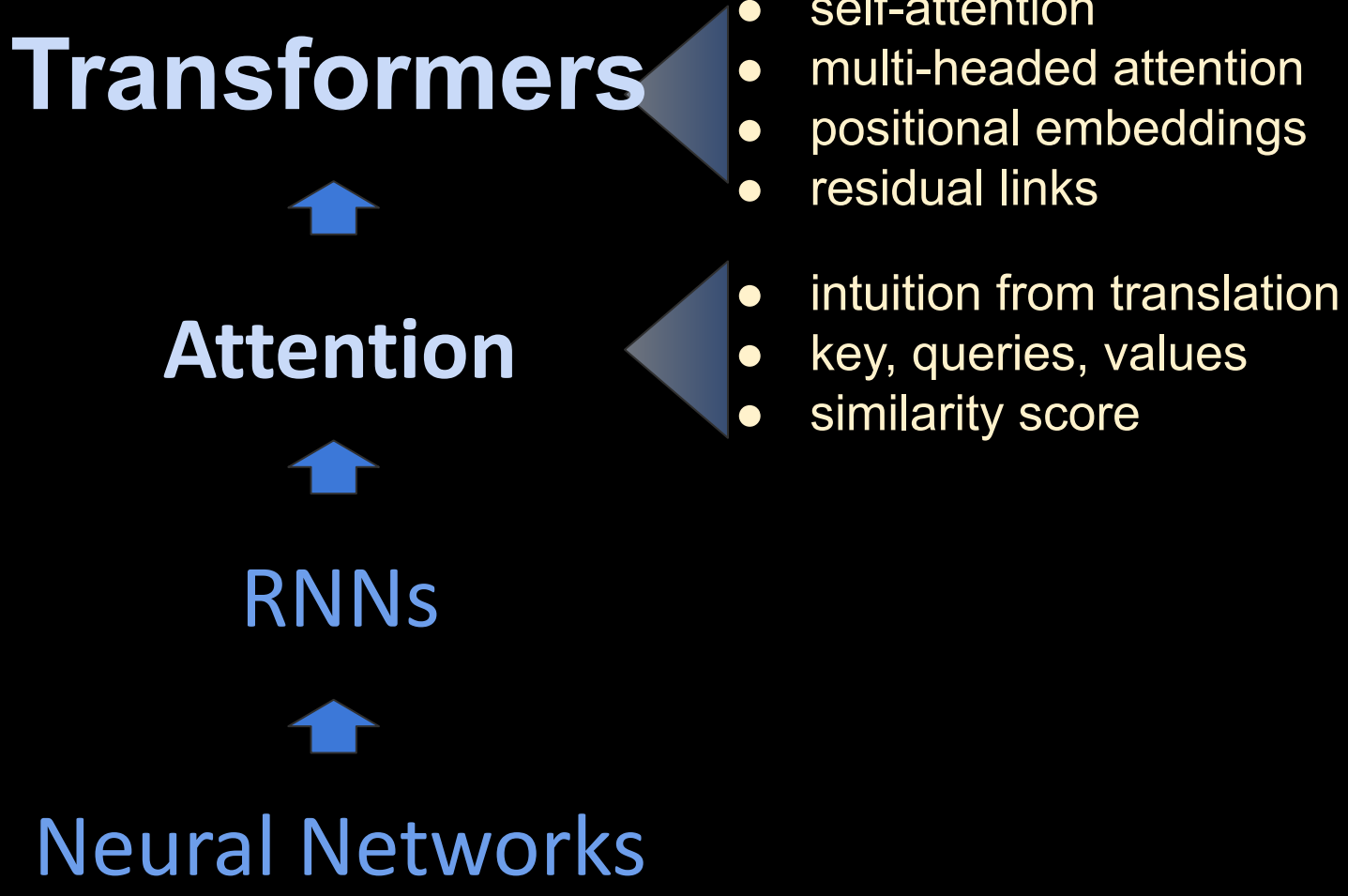
☆ Save Cite Cited by 173107 Related articles All 73 versions »

Transformers

Attention

RNNs

Neural Networks


Transformers
self-attention
multi-headed attention
positional embeddings
residual links
Attention
intuition from translation
key, queries, values
similarity score
RNNs
Neural Networks

Attention: Motivated from Translation

Machine Translation

As an optimization problem (Eisenstein, 2018):

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

Machine Translation

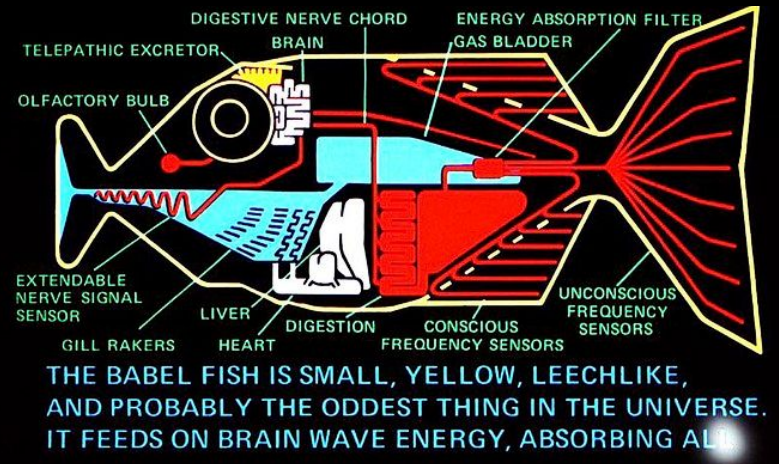
Why?

- $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?

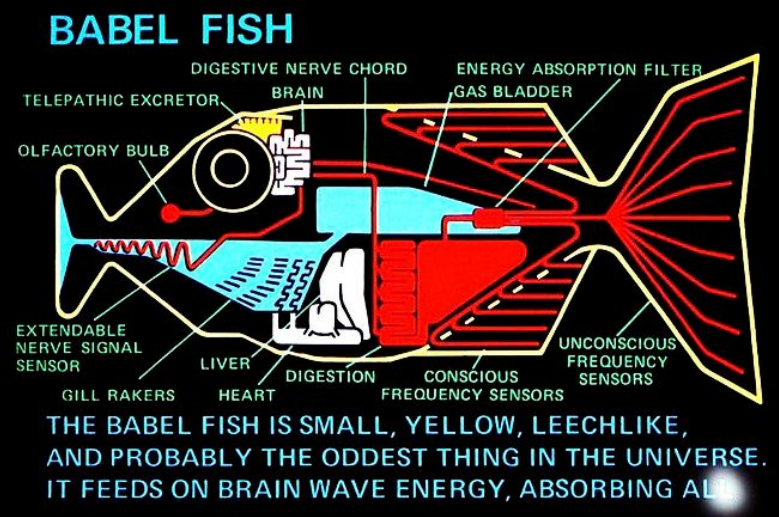
- $40billion/year industry
- A center piece of many genres of science fiction
- A fairly “universal” problem:
  - Language understanding
  - Language generation
- Societal benefits of inter-cultural communication



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



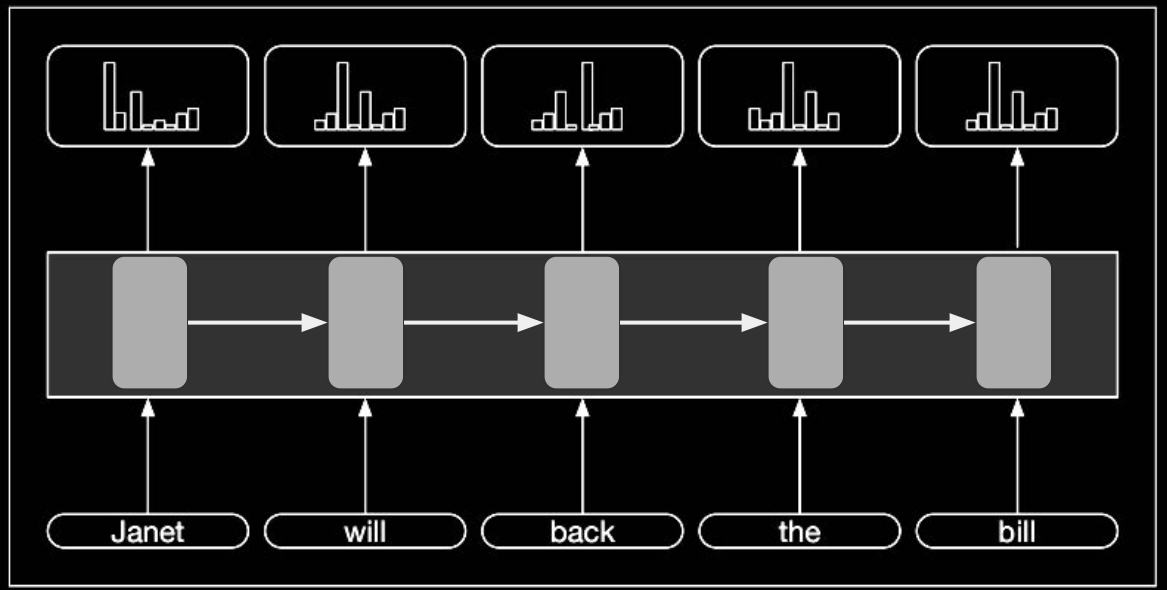
(Douglas Adams)

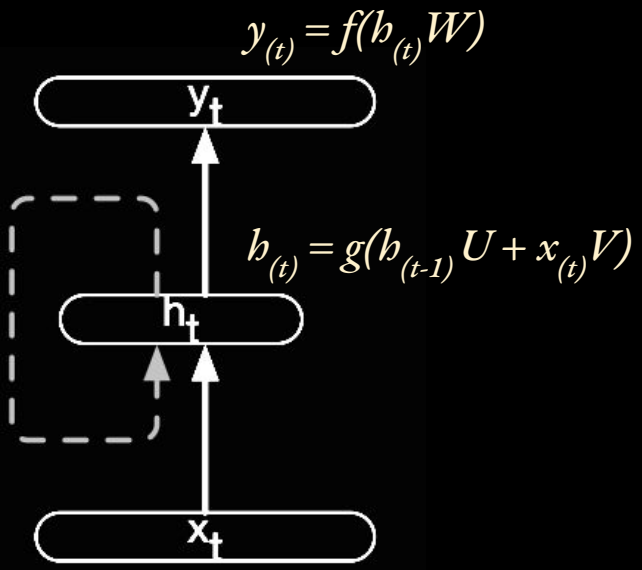
Machine Translation

Why Neural Network Approach works? (Manning, 2018)

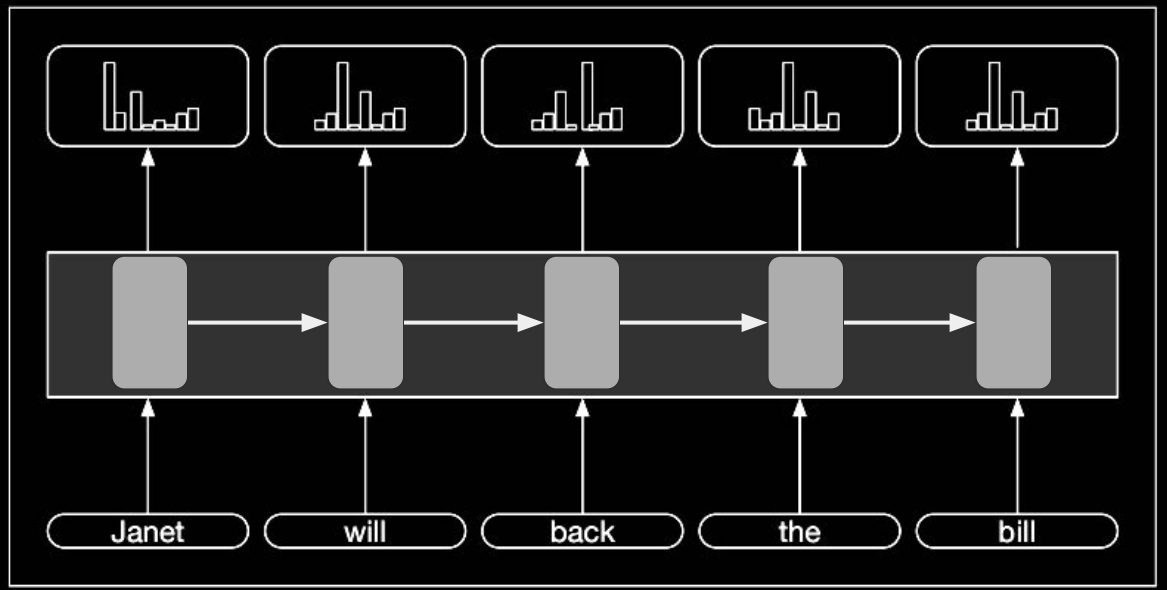
- Joint end-to-end training: learning all parameters at once.
- Exploiting distributed representations (embeddings)
- Exploiting variable-length context
- High quality generation from deep decoders - stronger language models (even when wrong, make sense)

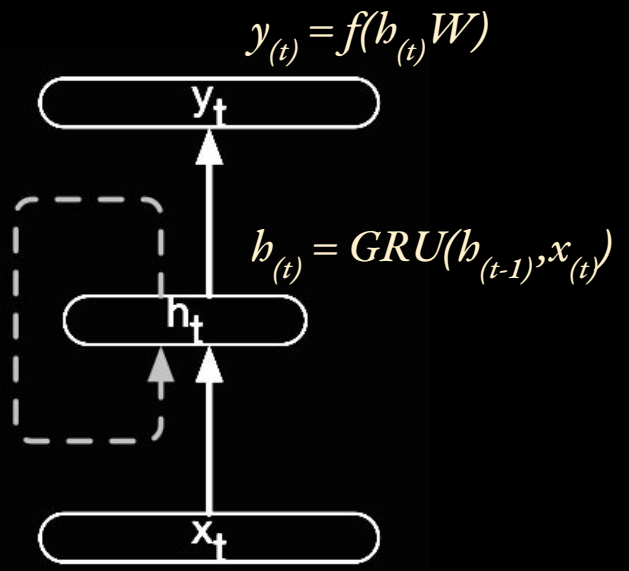
Recurrent Neural Network



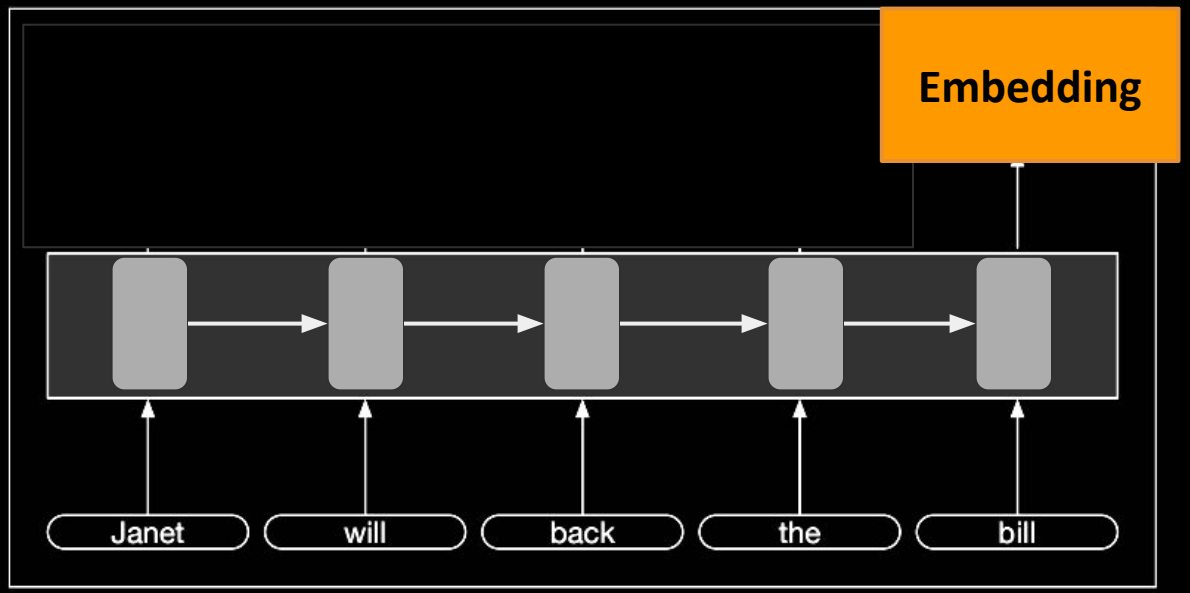


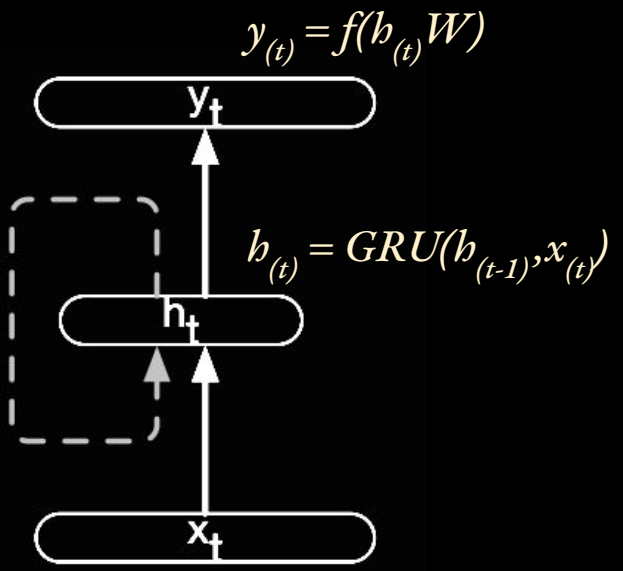
Recurrent Neural Network



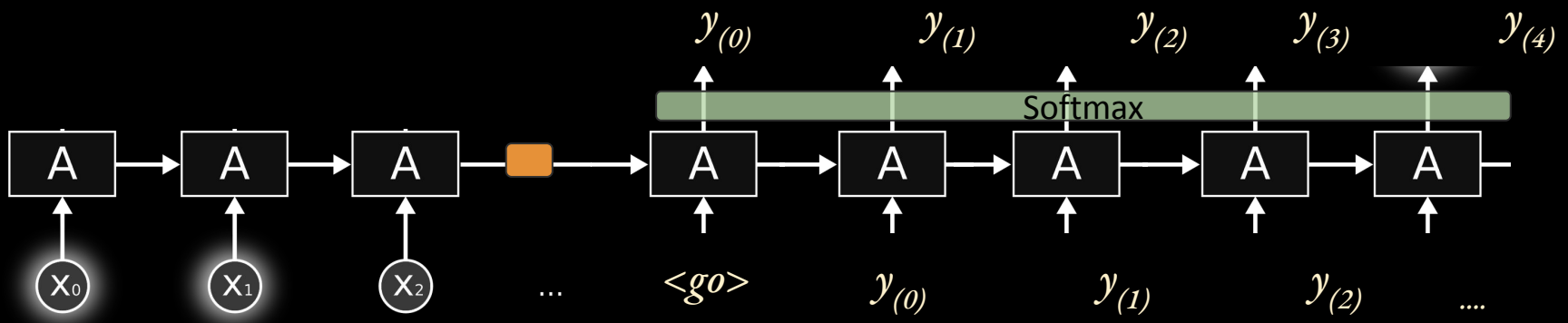


RNN: Encoder

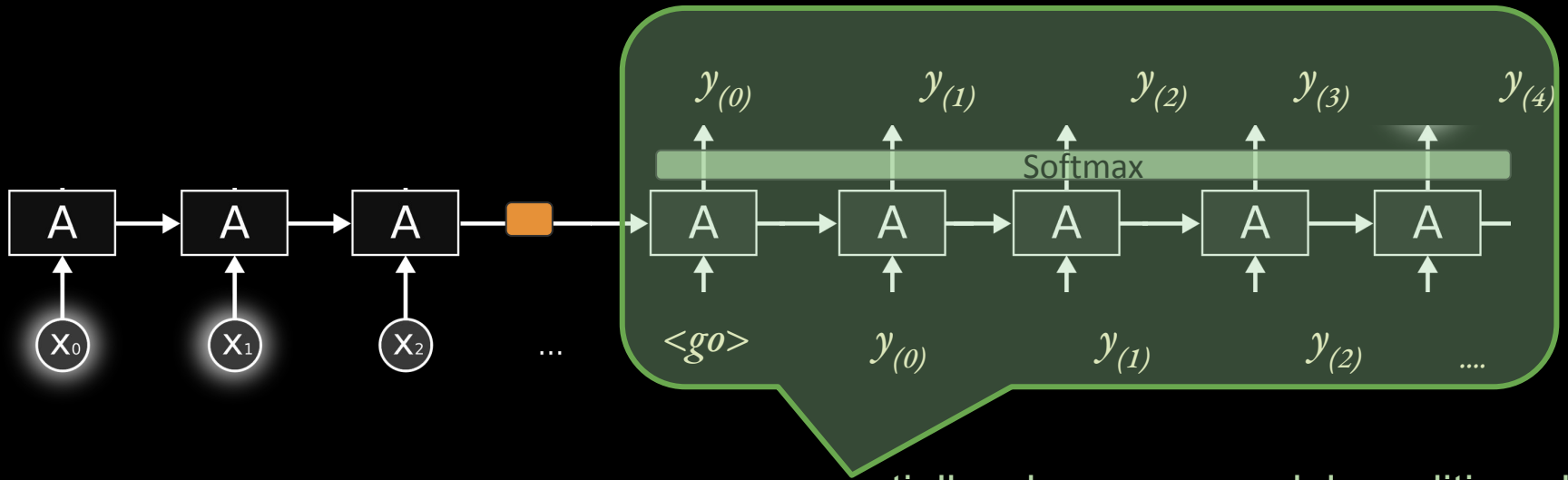

Embedding
Janet
will
back
the
bill


$y_{(t)} = f(h_{(t)}W)$
y_t
$h_{(t)} = GRU(h_{(t-1)}, x_{(t)})$
h_t
x_t

Encoder-Decoder (Simplified Representation)

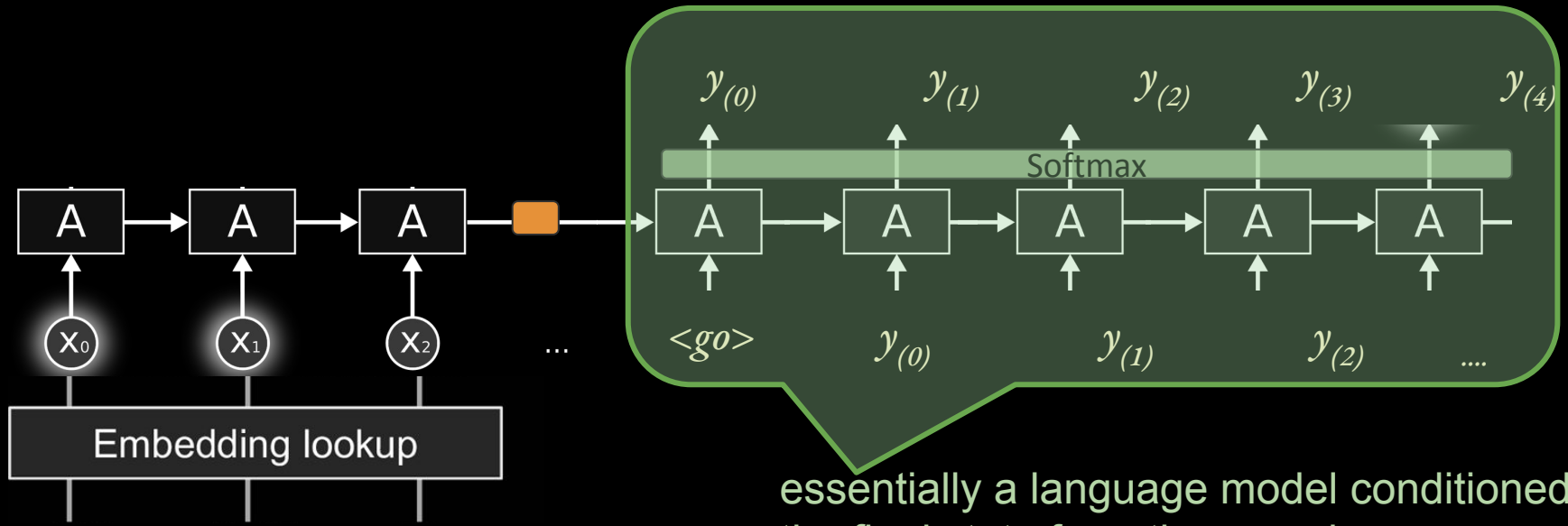


Encoder-Decoder (Simplified Representation)



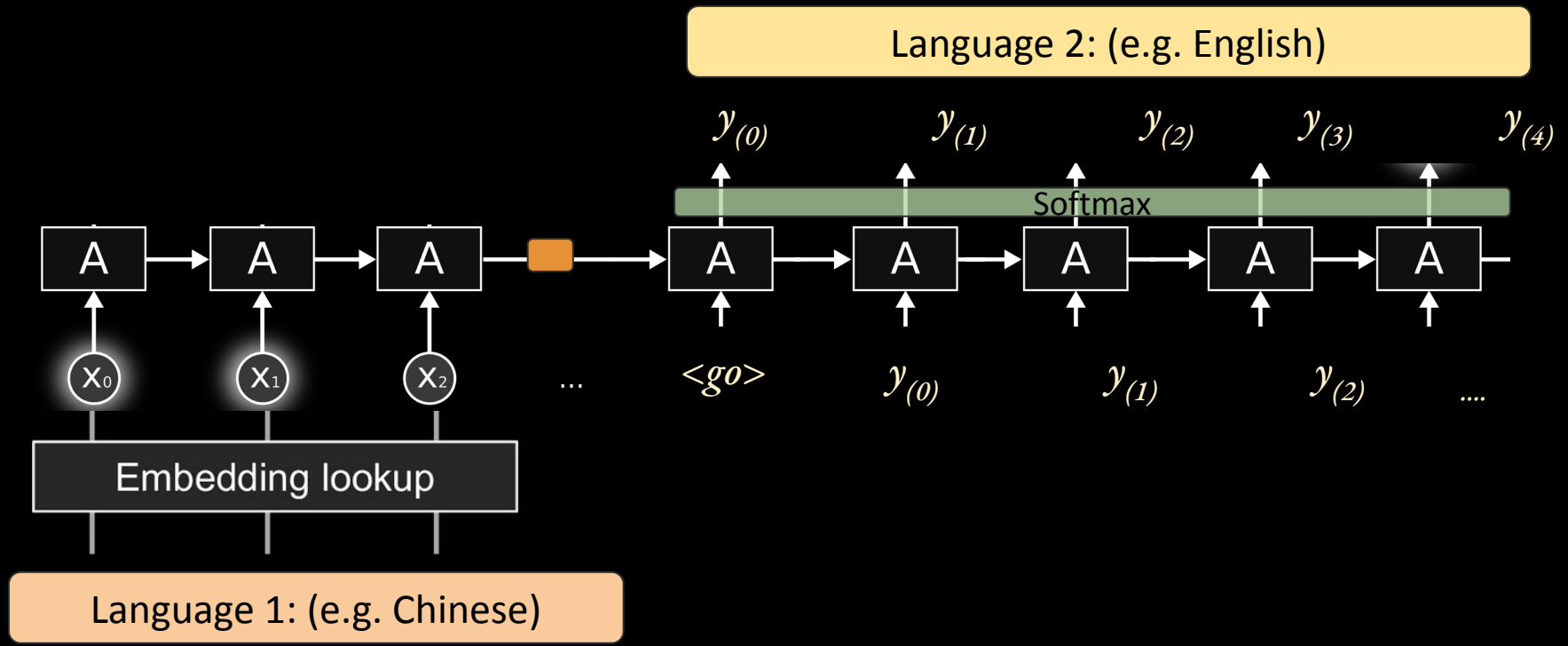
essentially a language model conditioned on the final state from the encoder.

Encoder-Decoder (Simplified Representation)



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

Encoder-Decoder (Simplified Representation)

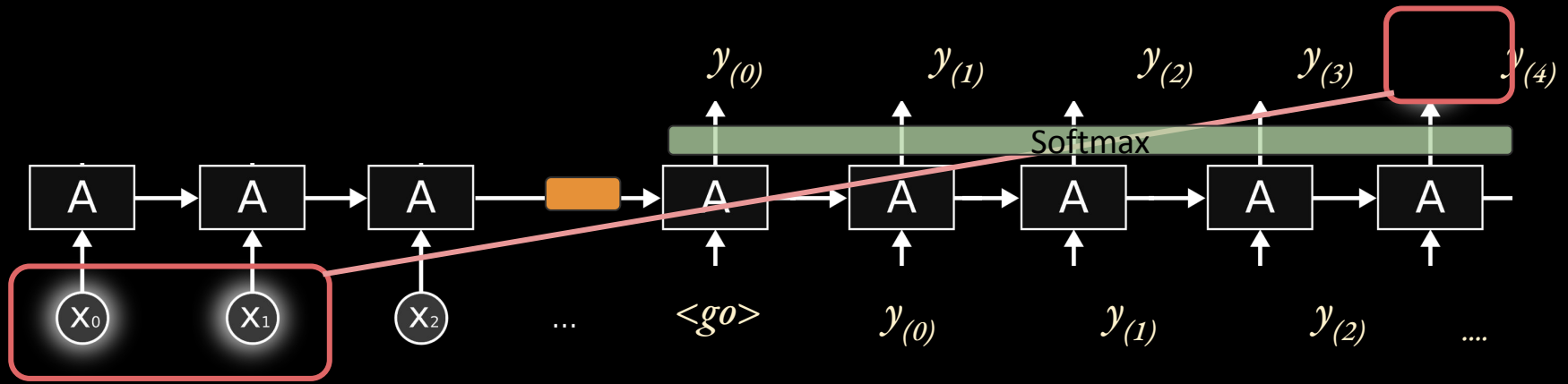


Encoder-Decoder

Challenge:

The ball was kicked by kayla.

- Long distance dependency when translating:



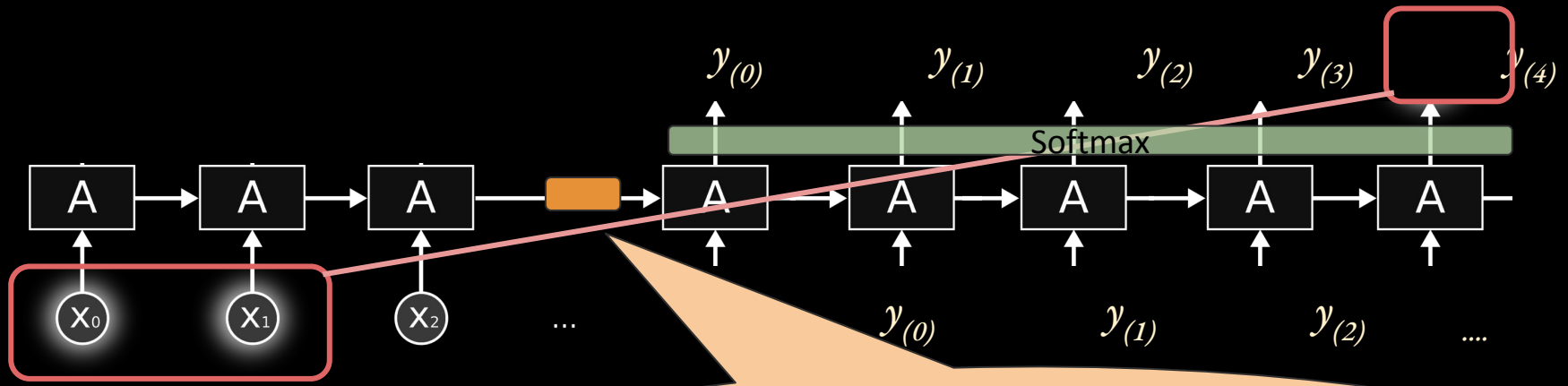
Kayla kicked the ball.

Encoder-Decoder

Challenge:

The ball was kicked by kayla.

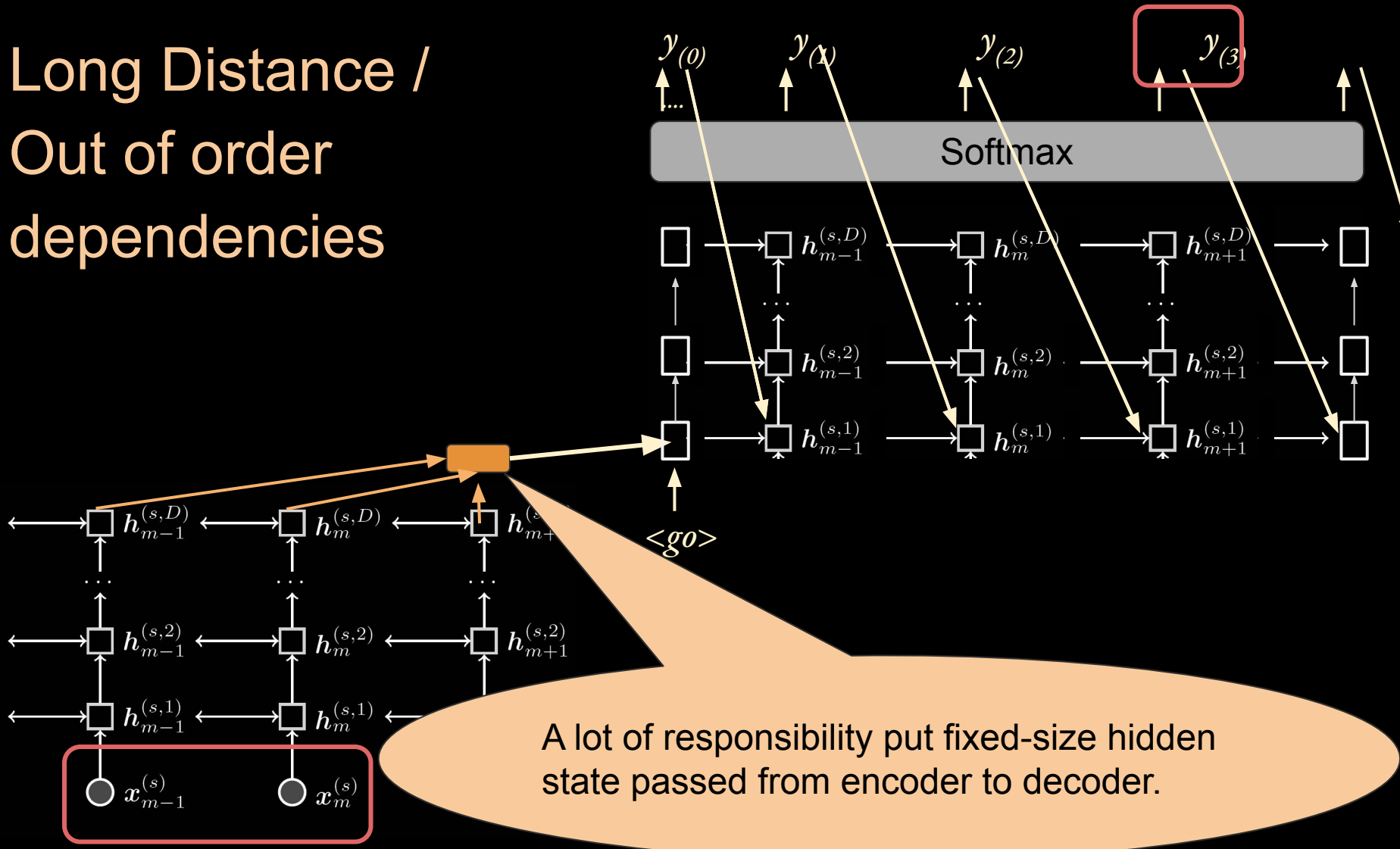
- Long distance dependency when translating:



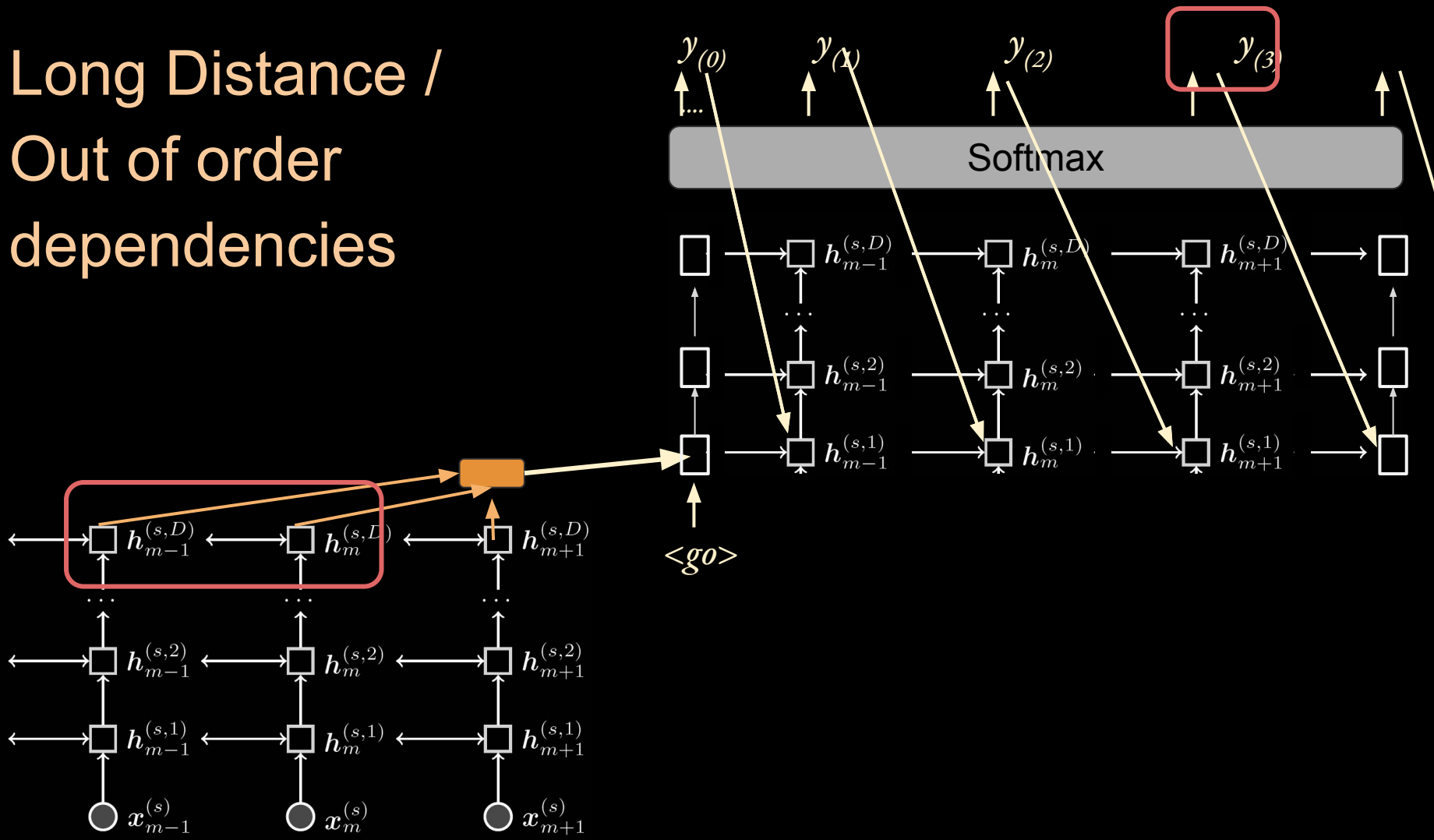
Kayla kicked the ball.

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

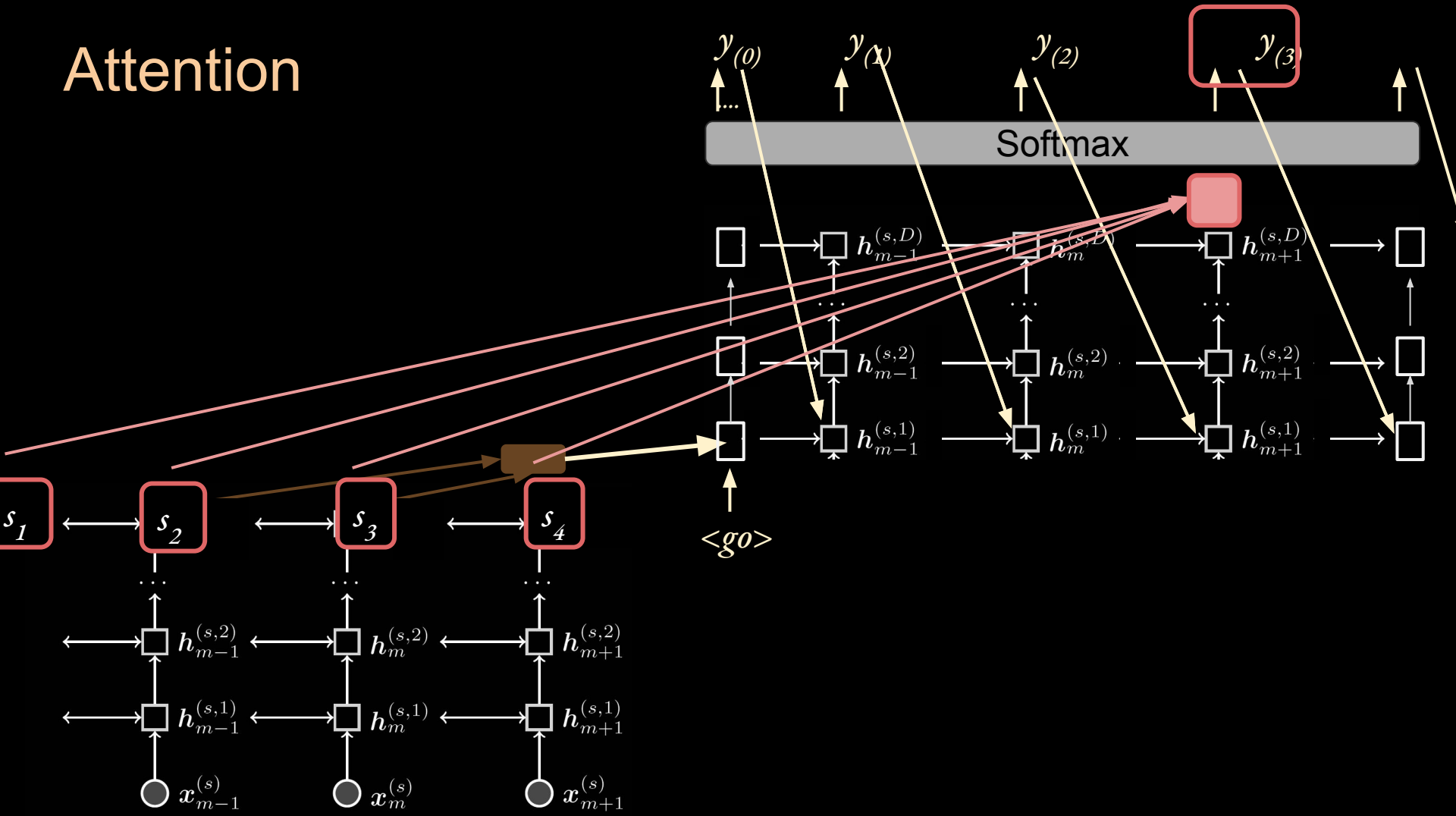
Long Distance / Out of order dependencies



Long Distance / Out of order dependencies

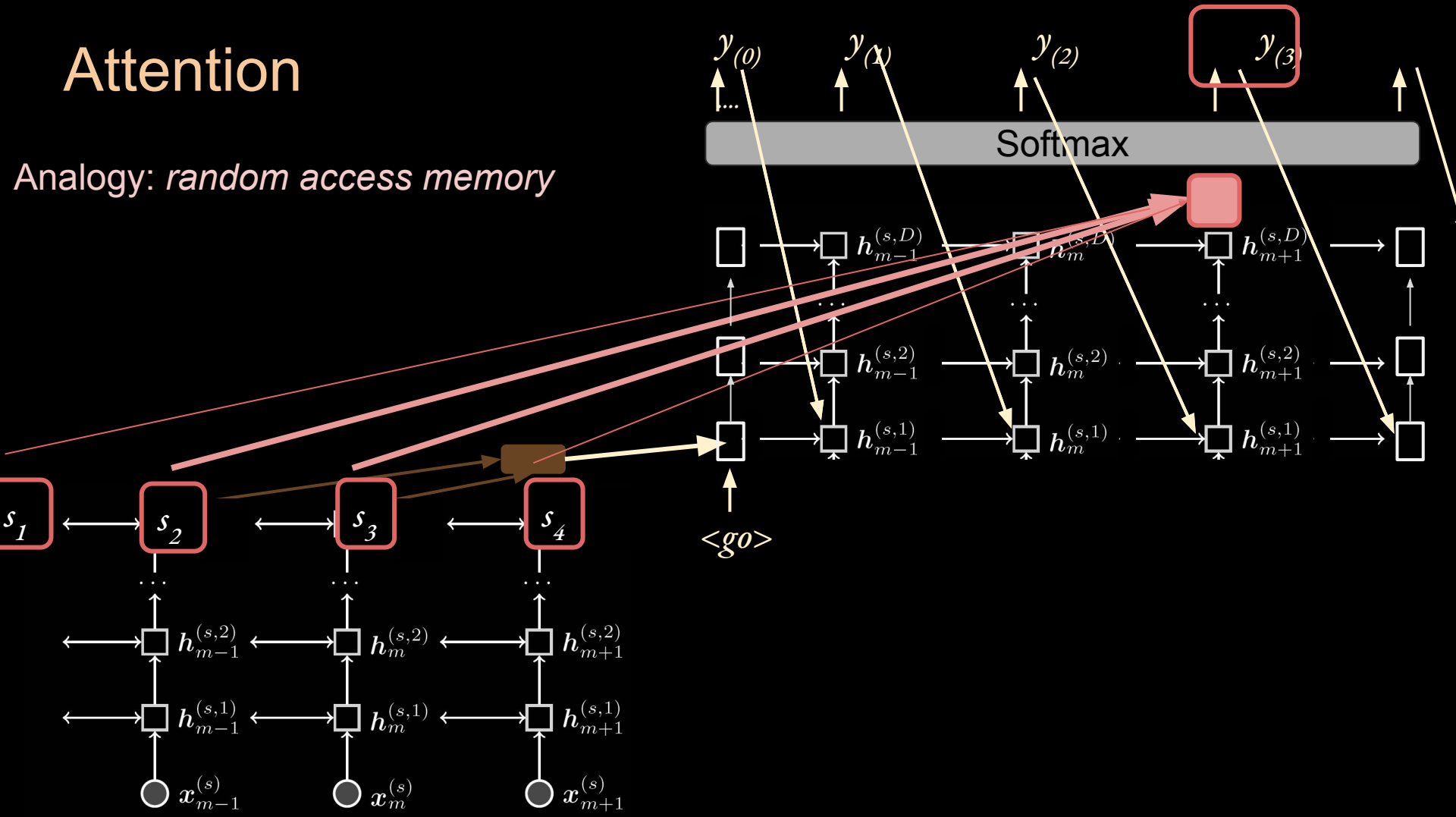


Attention

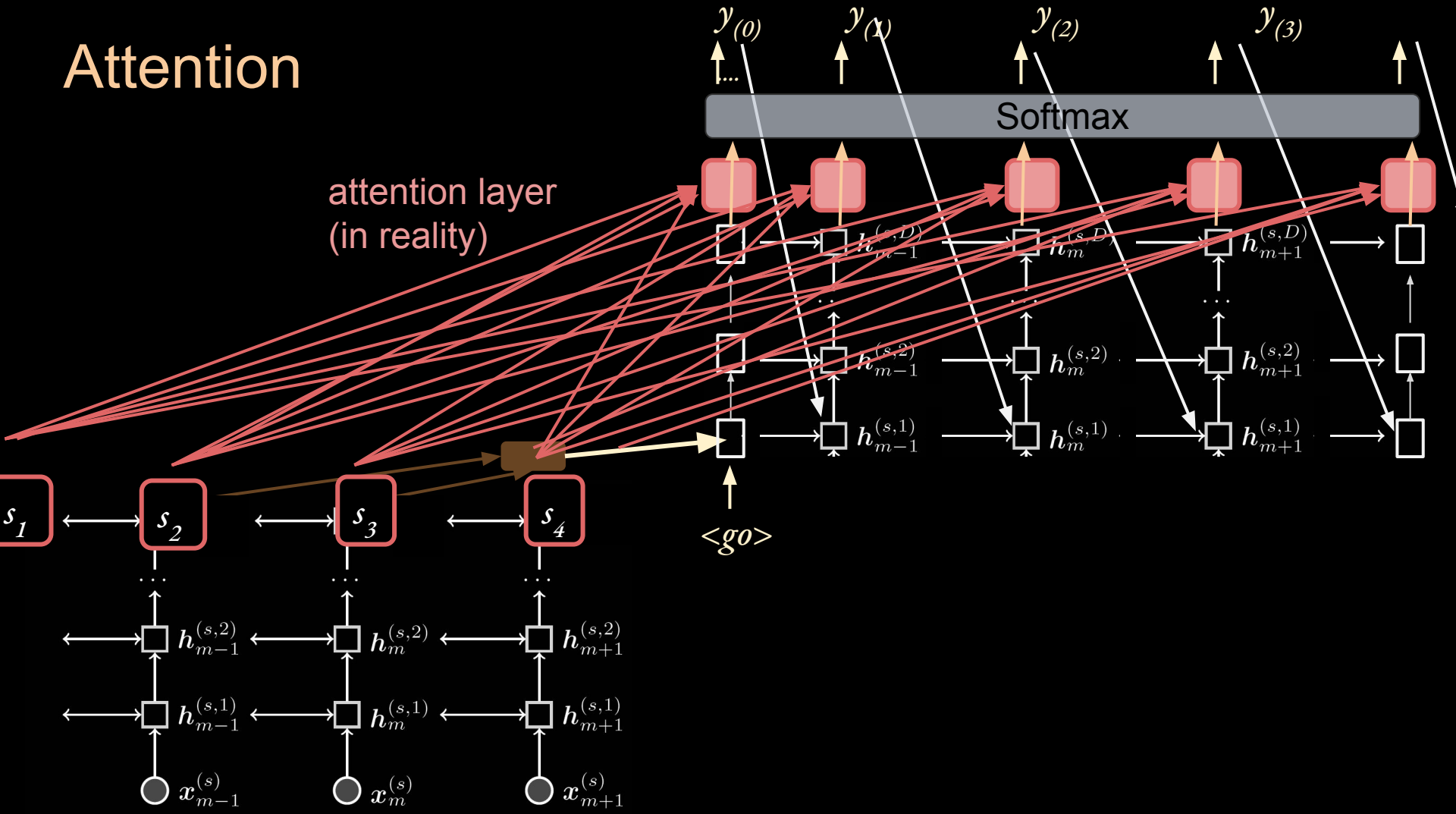

$y_{(0)}$
$y_{(1)}$
$y_{(2)}$
$y_{(3)}$
Softmax
$h_{m-1}^{(s,D)}$
$h_m^{(s,D)}$
$h_{m+1}^{(s,D)}$
$h_{m-1}^{(s,2)}$
$h_m^{(s,2)}$
$h_{m+1}^{(s,2)}$
$h_{m-1}^{(s,1)}$
$h_m^{(s,1)}$
$h_{m+1}^{(s,1)}$
s_1
s_2
s_3
s_4
<go>
$x_{m-1}^{(s)}$
$x_m^{(s)}$
$x_{m+1}^{(s)}$

Attention

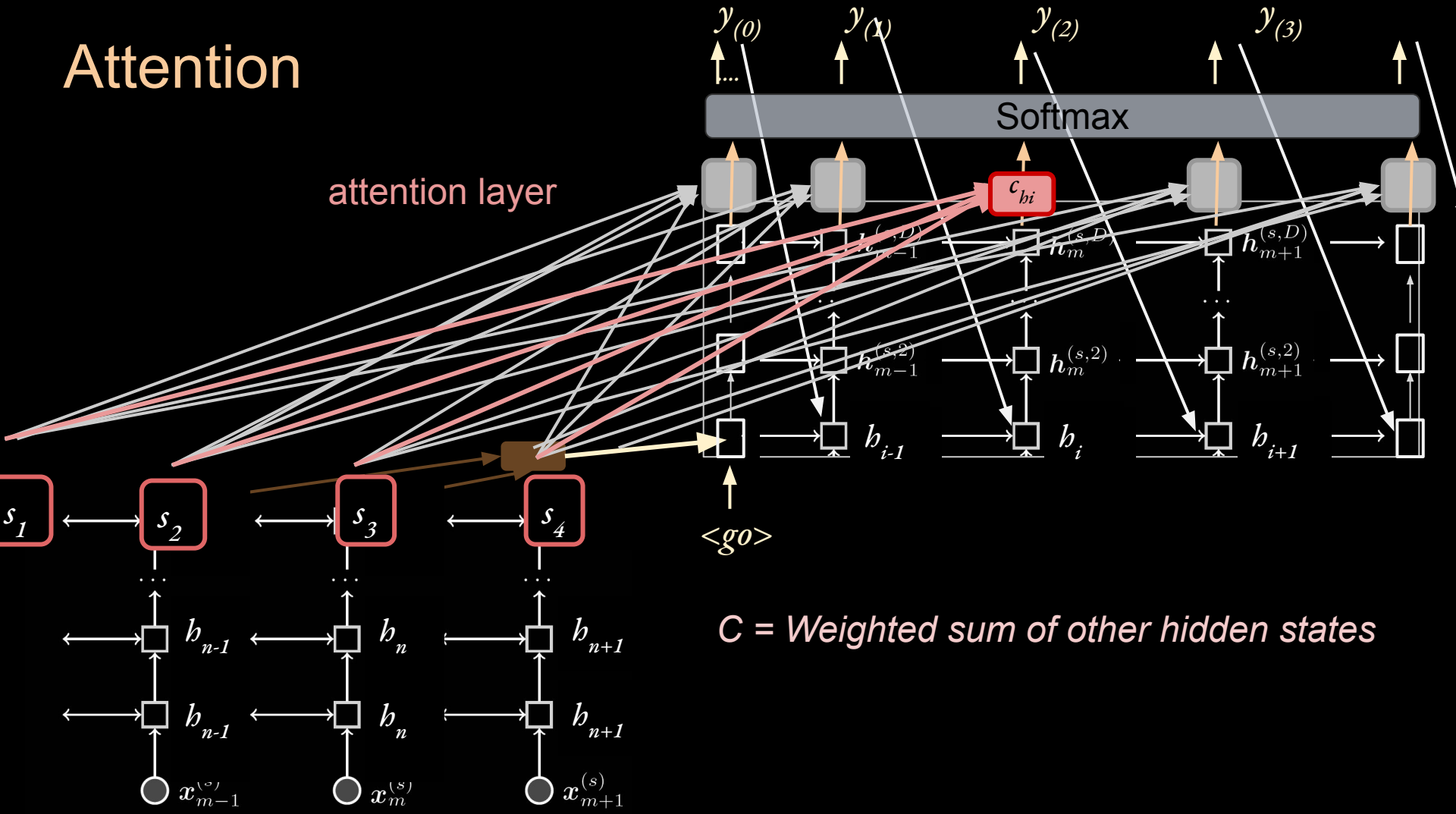
Analogy: *random access memory*



Attention

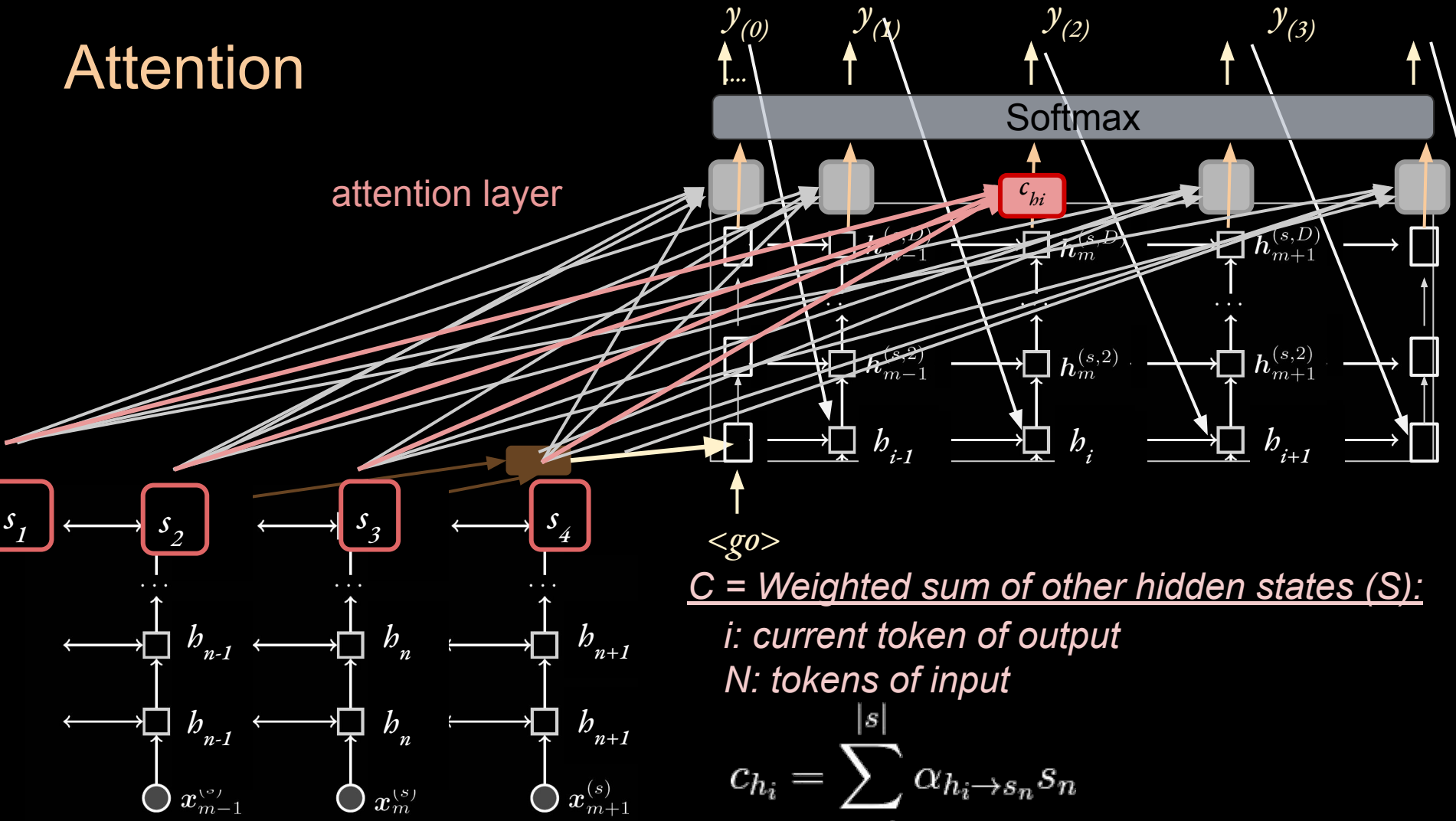

$y_{(0)}$
$y_{(1)}$
$y_{(2)}$
$y_{(3)}$
Softmax
attention layer
(in reality)
$h_{m-1}^{(s,D)}$
$h_m^{(s,D)}$
$h_{m+1}^{(s,D)}$
$h_{m-1}^{(s,2)}$
$h_m^{(s,2)}$
$h_{m+1}^{(s,2)}$
$h_{m-1}^{(s,1)}$
$h_m^{(s,1)}$
$h_{m+1}^{(s,1)}$
s_1
s_2
s_3
s_4
<go>
$x_{m-1}^{(s)}$
$x_m^{(s)}$
$x_{m+1}^{(s)}$

Attention



C = Weighted sum of other hidden states

Attention



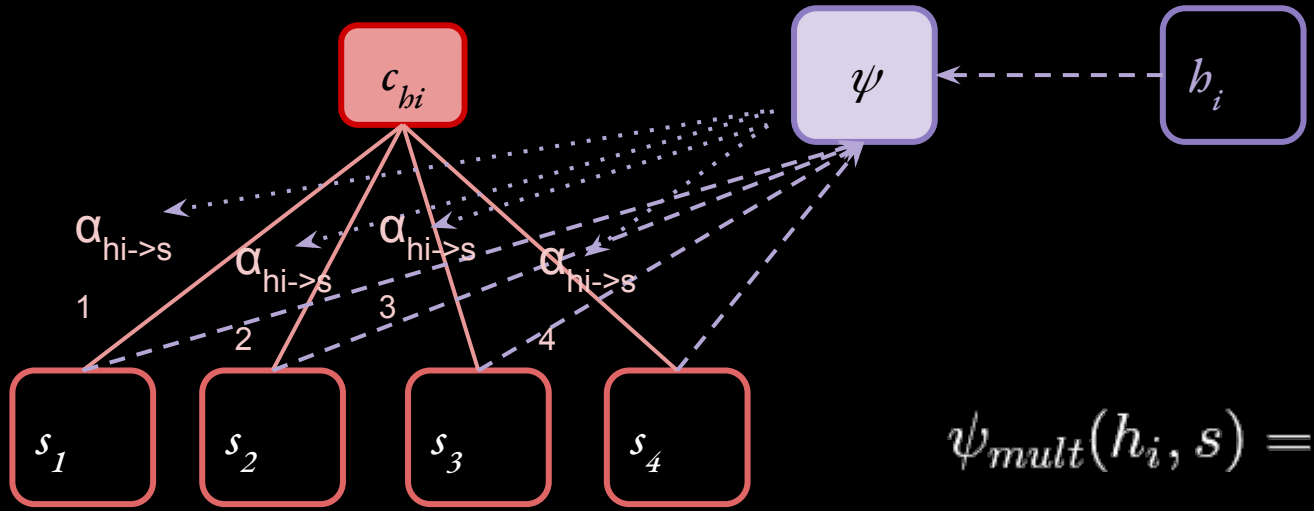
C = Weighted sum of other hidden states (S):

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

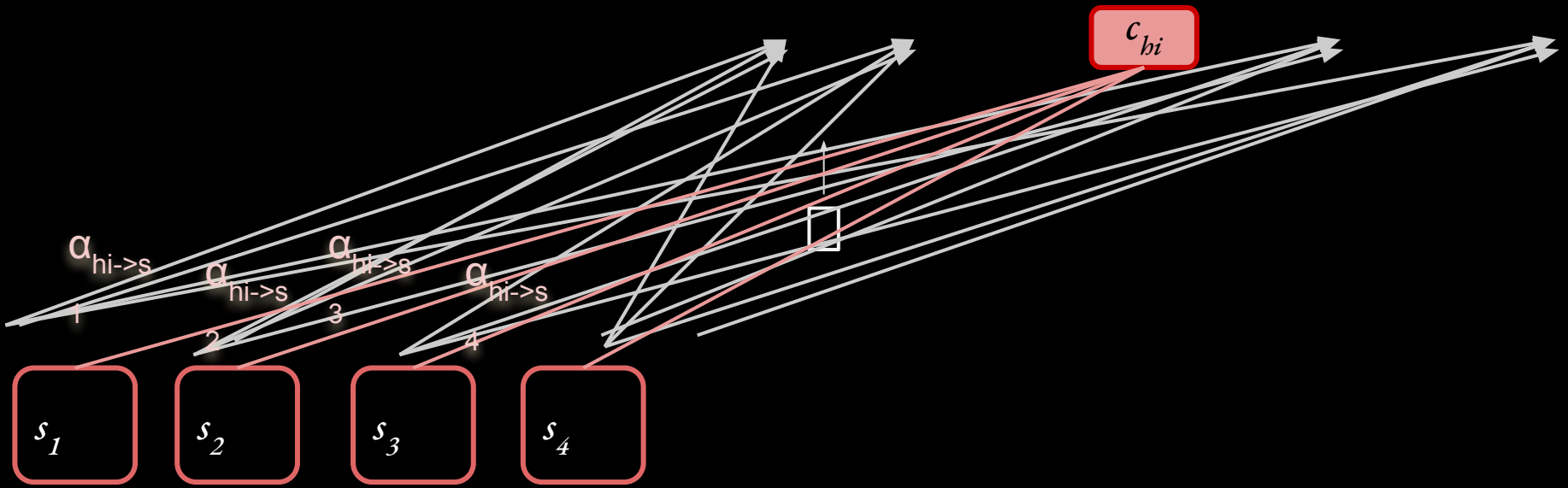


$$\psi_{mult}(h_i, s) = s^T W h_i$$

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

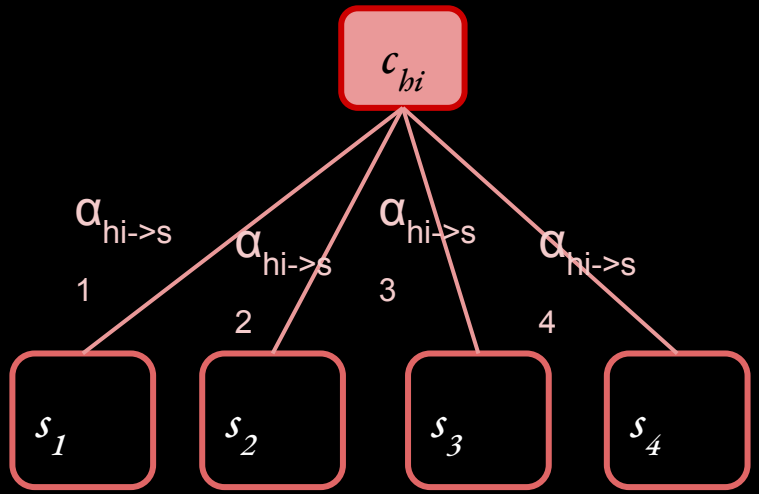
$$c_{h_i} = \sum_{n=1}^{|s|} \alpha_{h_i \to s_n} s_n$$

Attention



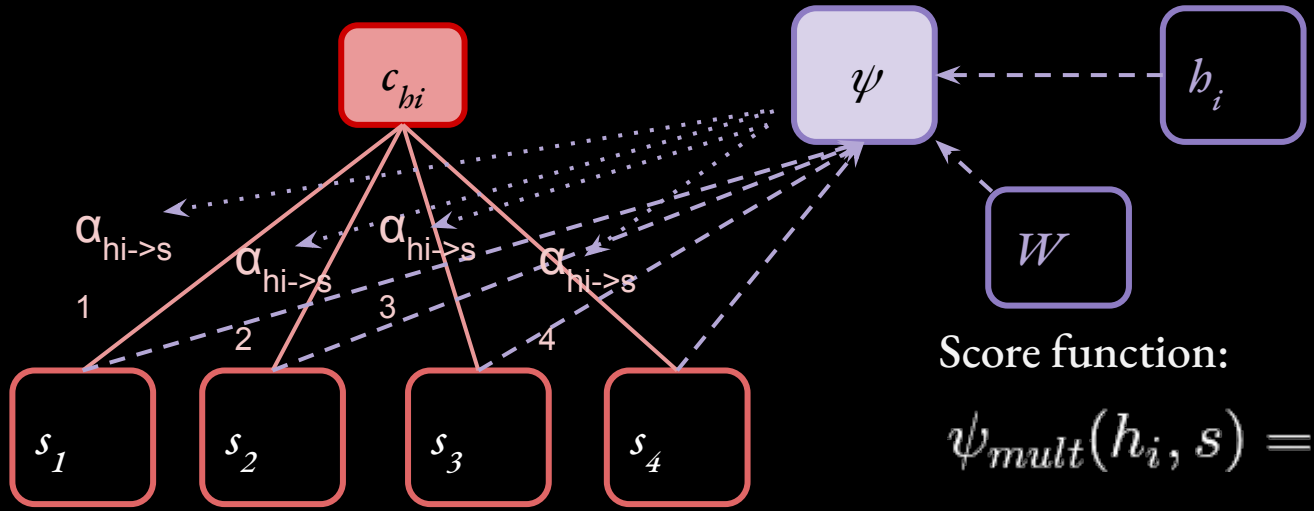
$$c_{h_i} = \sum_{n=1}^{|s|} \alpha_{h_i \to s_n} s_n$$

Attention



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

Attention



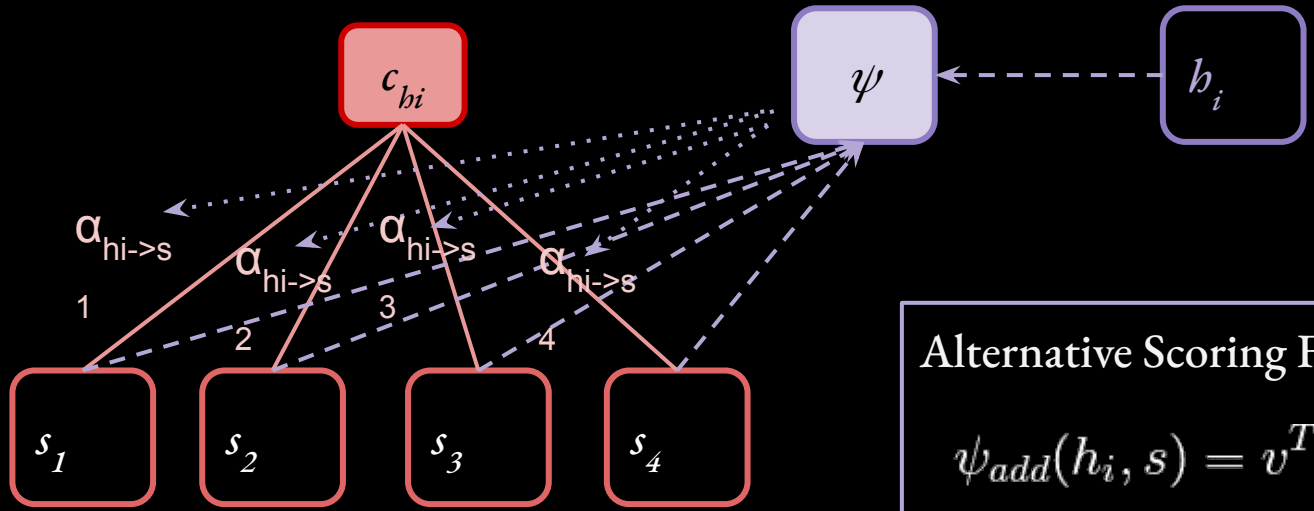
Score function:

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

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

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

Attention



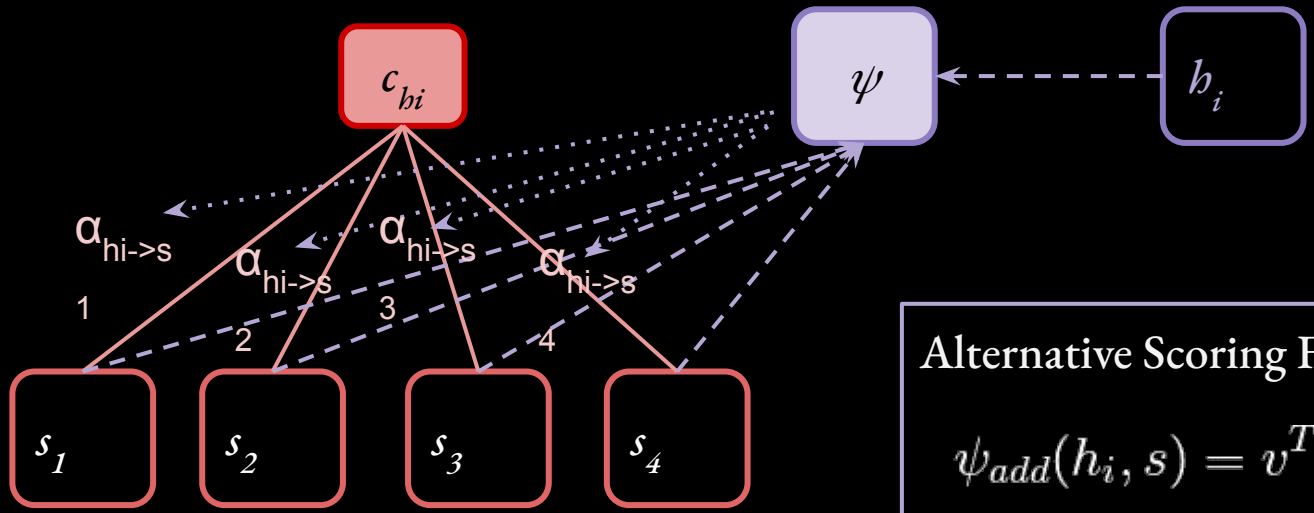
Alternative Scoring Functions

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

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

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

Attention



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

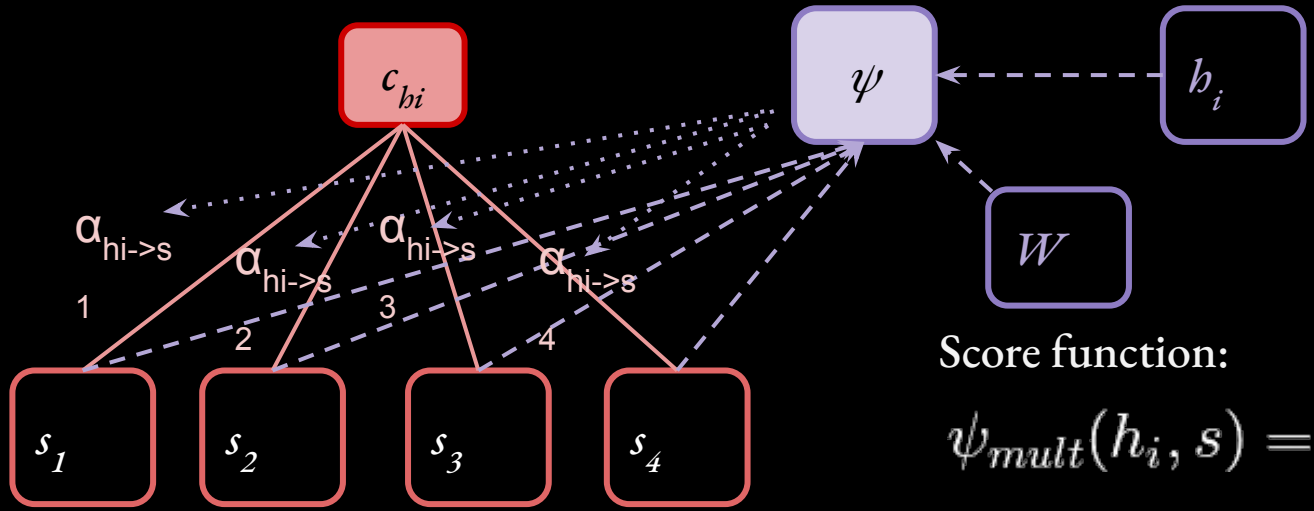
Alternative Scoring Functi

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

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

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

Attention



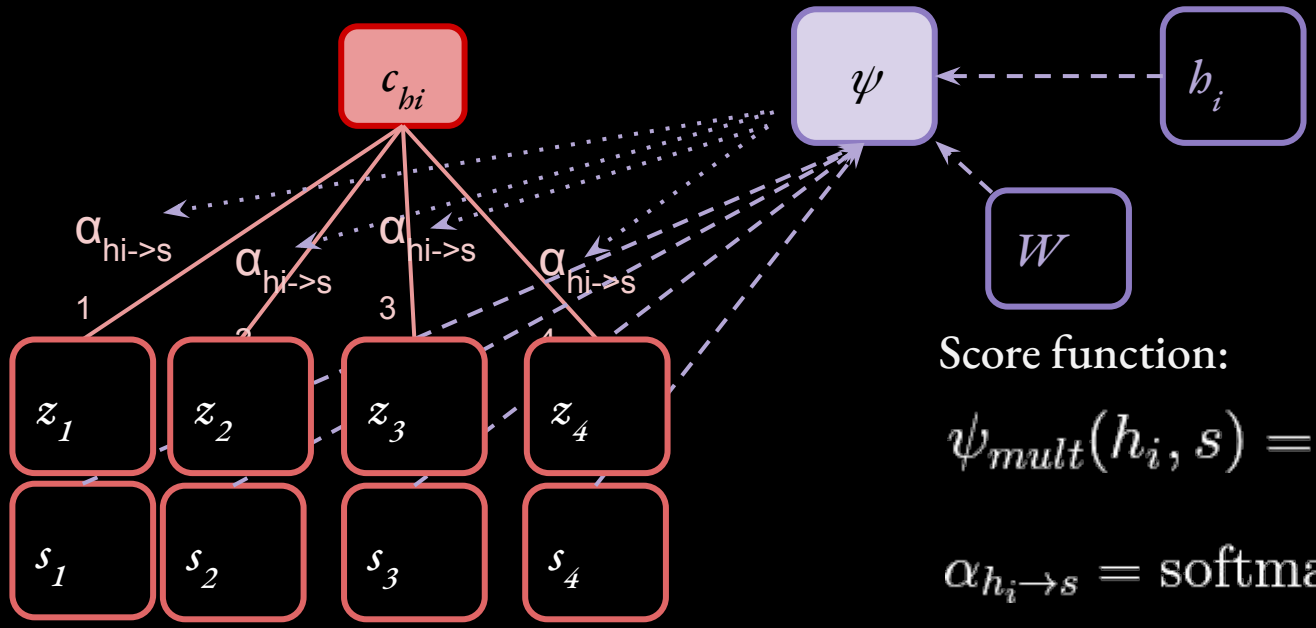
Score function:

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

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

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

Attention



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

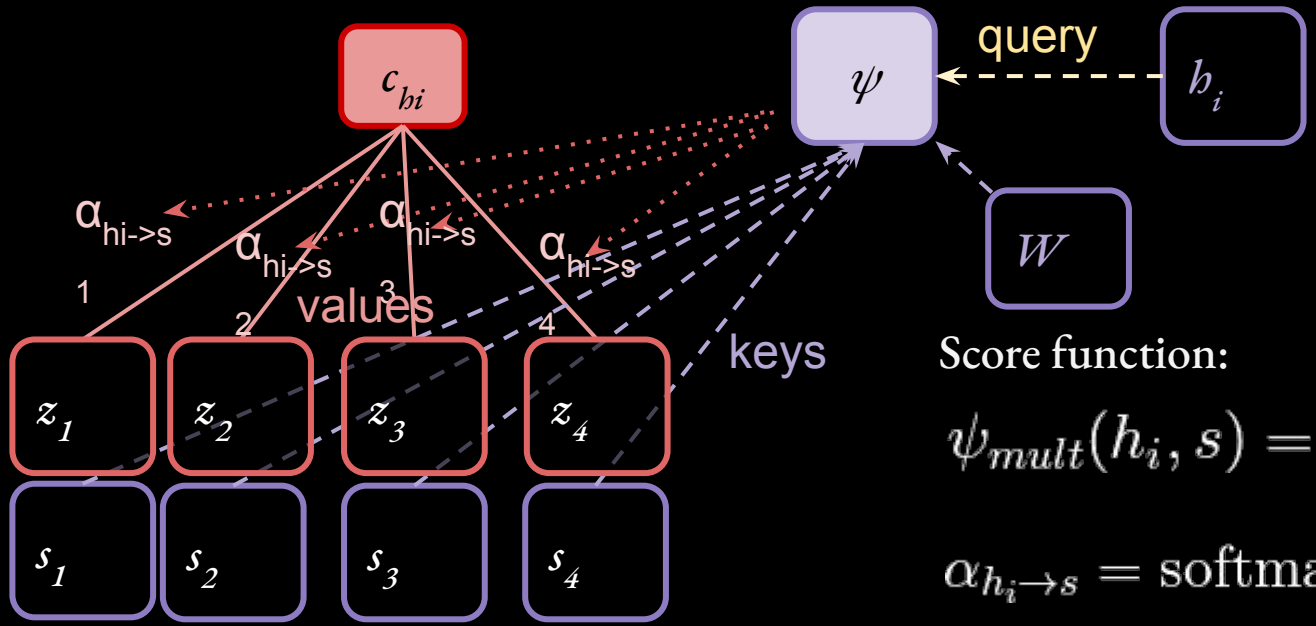
Score function:

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

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

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

Attention



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

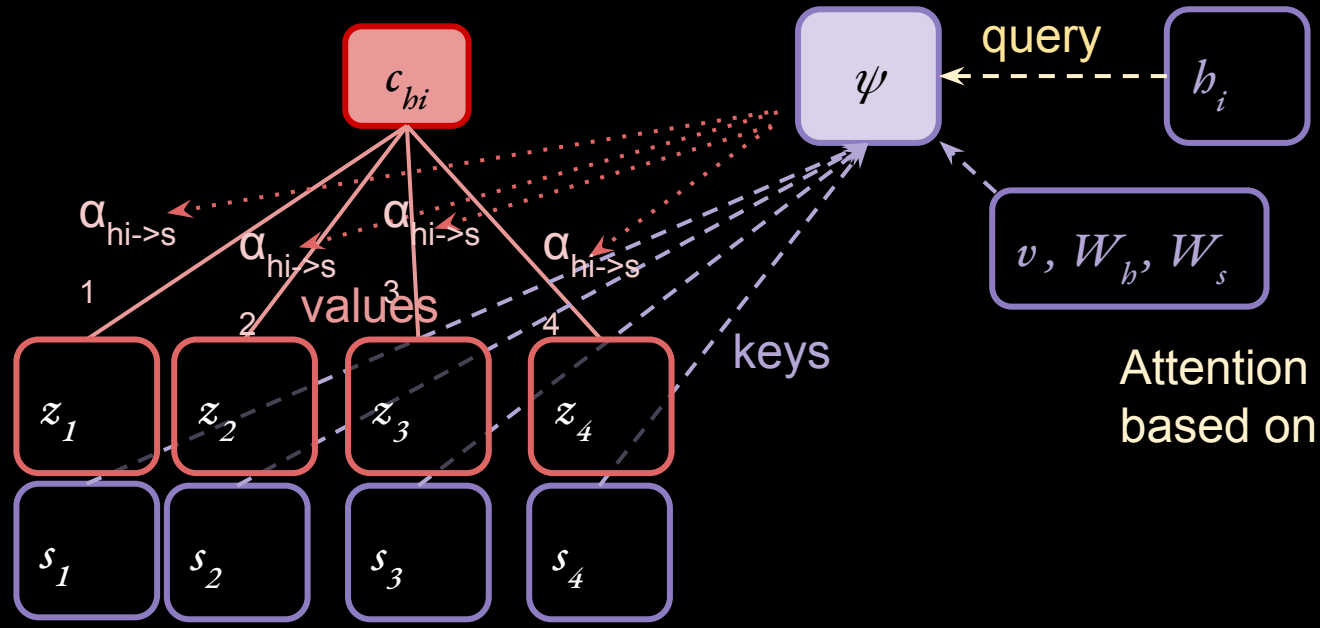
Score function:

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

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

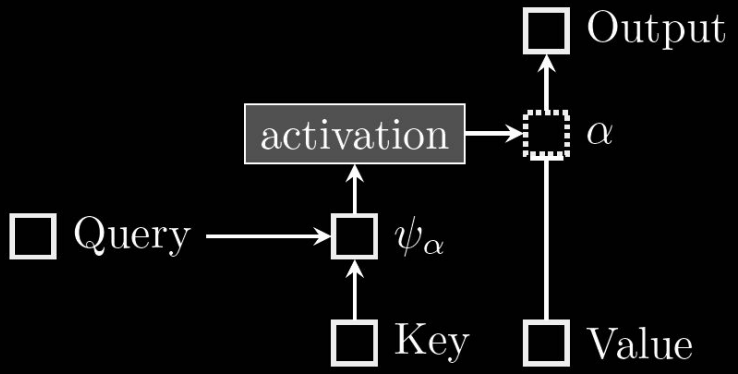
$$c_{h_i} = \sum_{n=1}^{|s|} \alpha_{h_i \to s_n} z_n$$

Attention

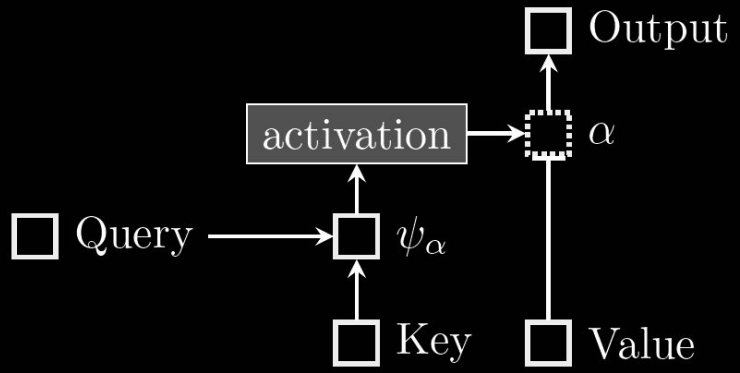


(Eisenstein, 2018)

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

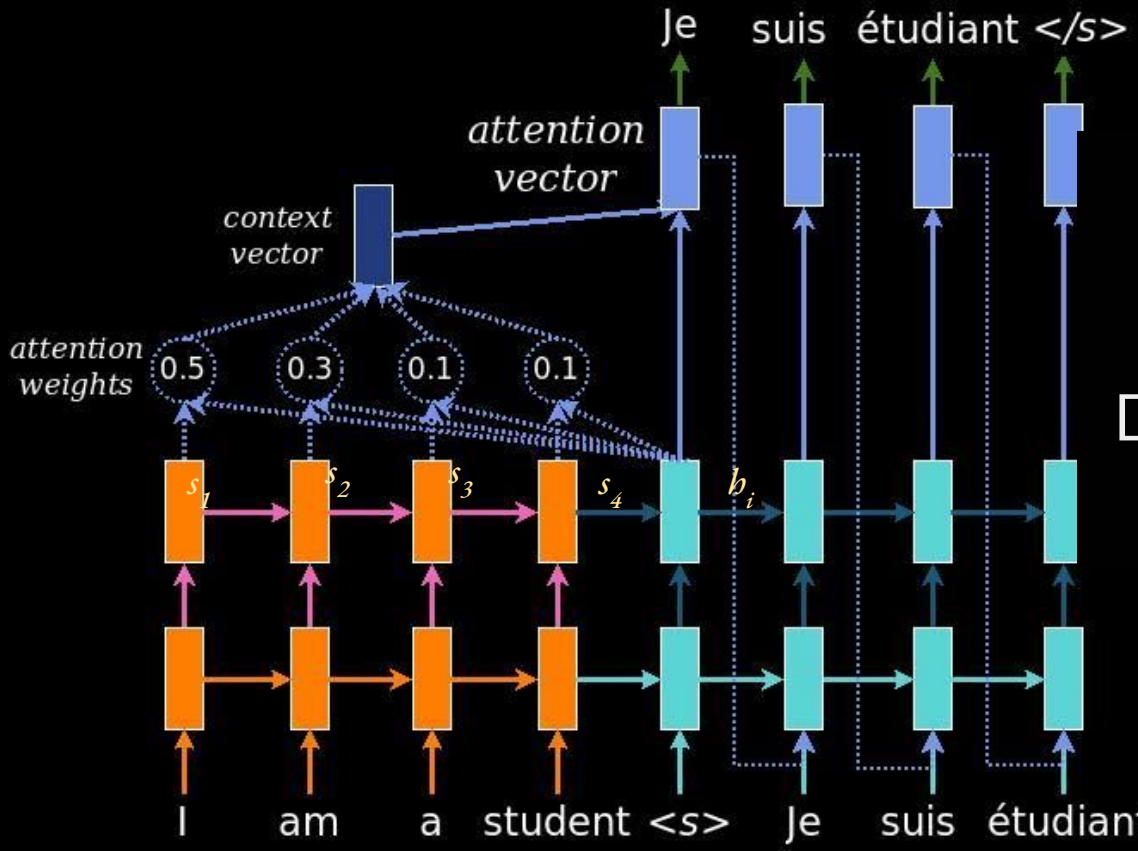


Attention as weighting a value based on a query and key:

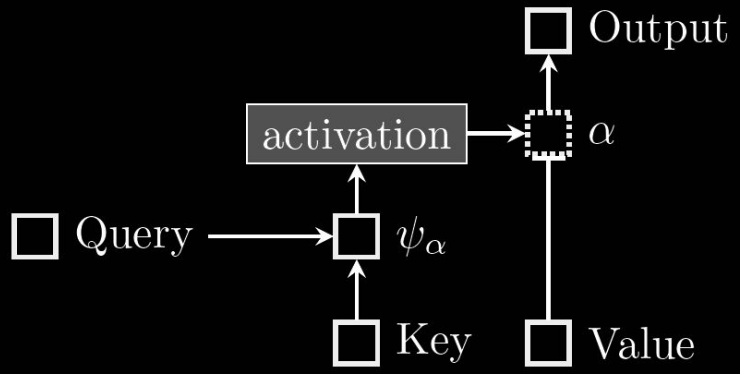


(Eisenstein, 2018)

Attention

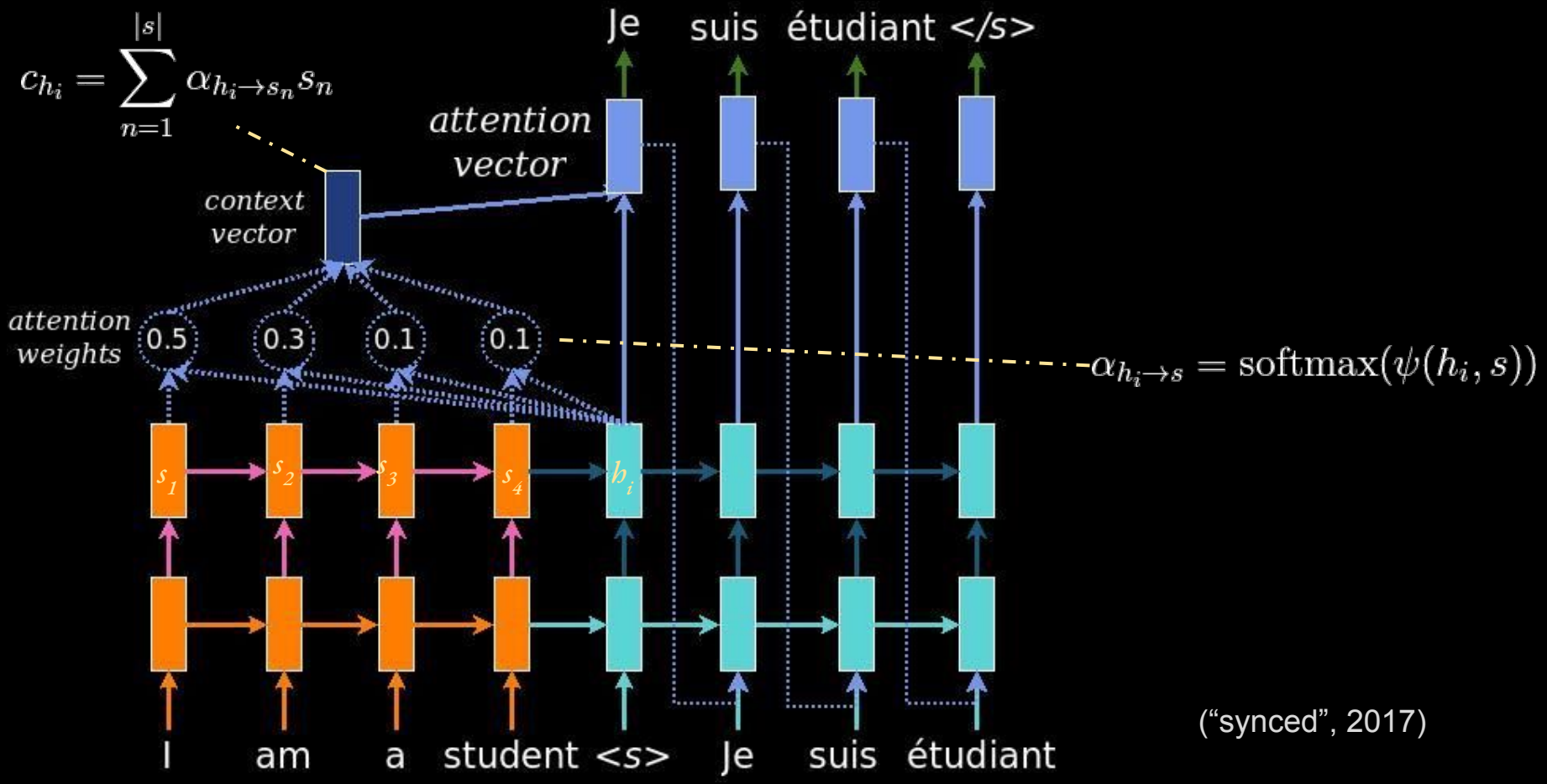


Attention as weighting a value based on a query and key:



("synced", 2017)

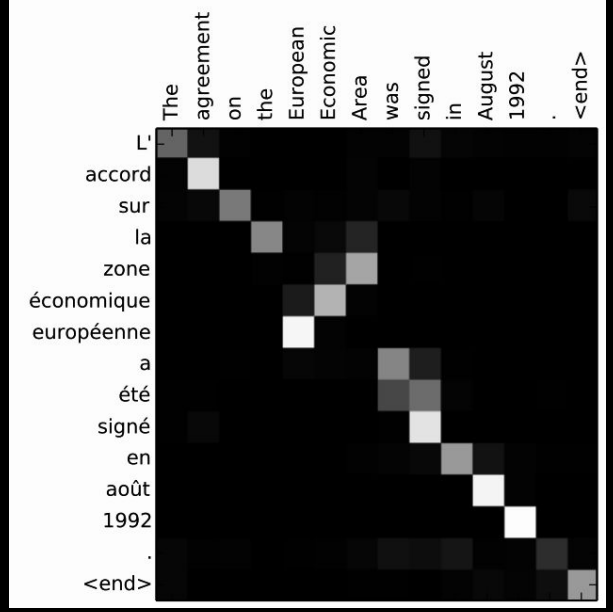
Attention



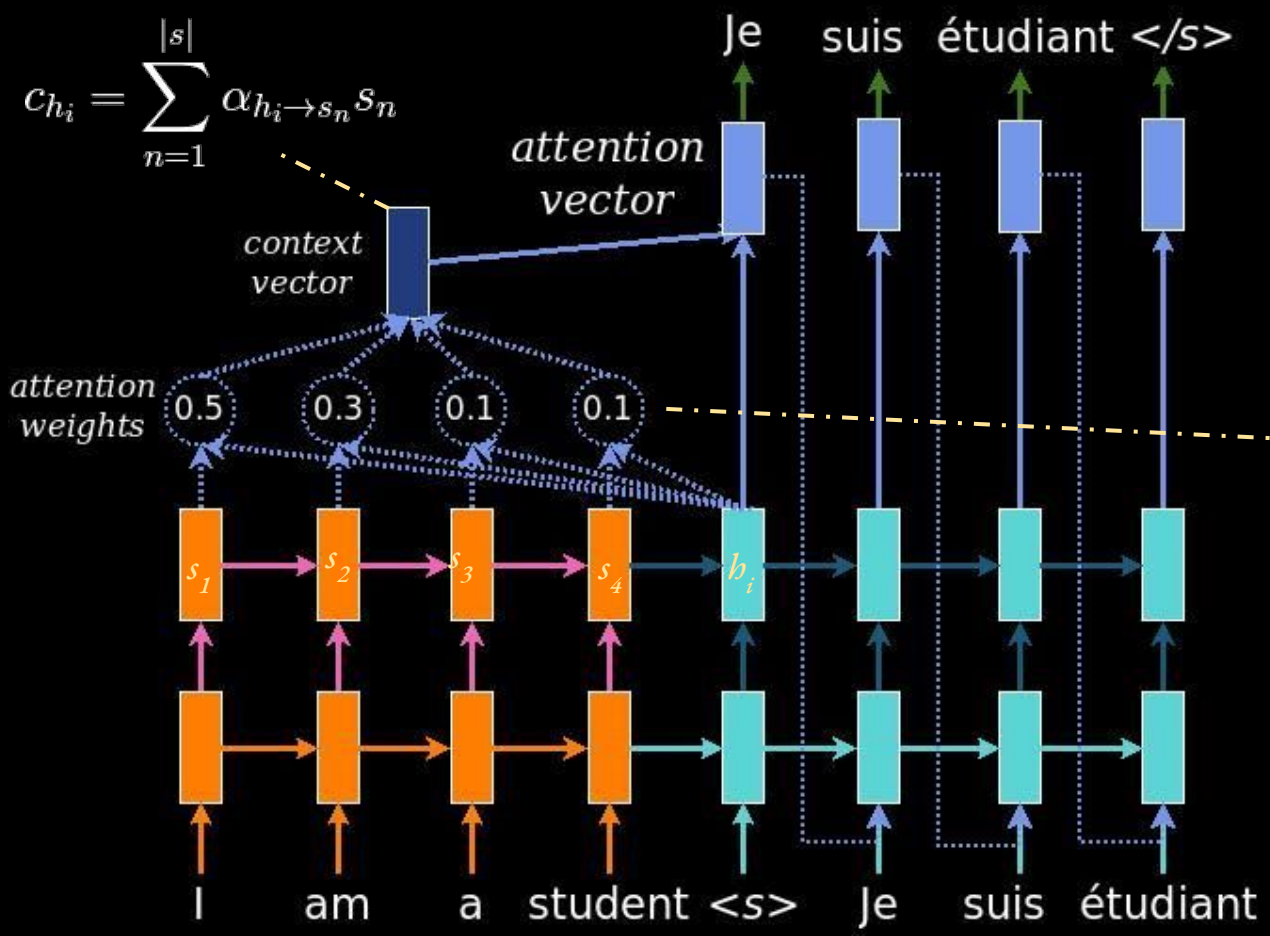
("synced", 2017)

Attention

(Bahdanau et al., 2015)



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

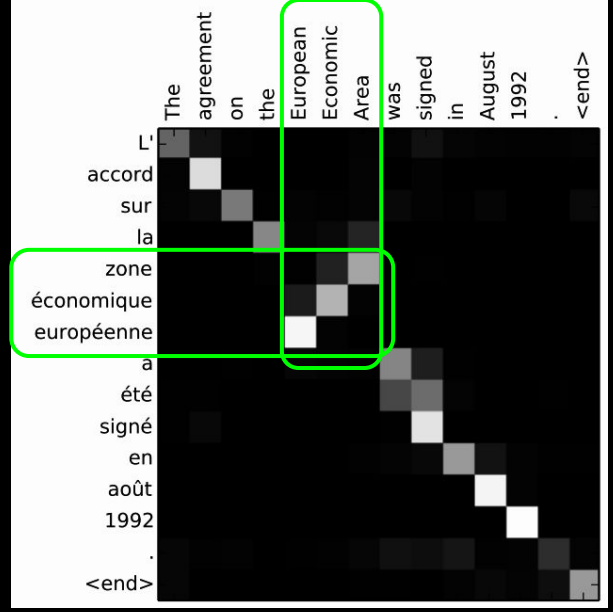


$$\alpha_{h_i \to s} = \mathrm{softmax}(\psi(h_i, s))$$

("synced", 2017)

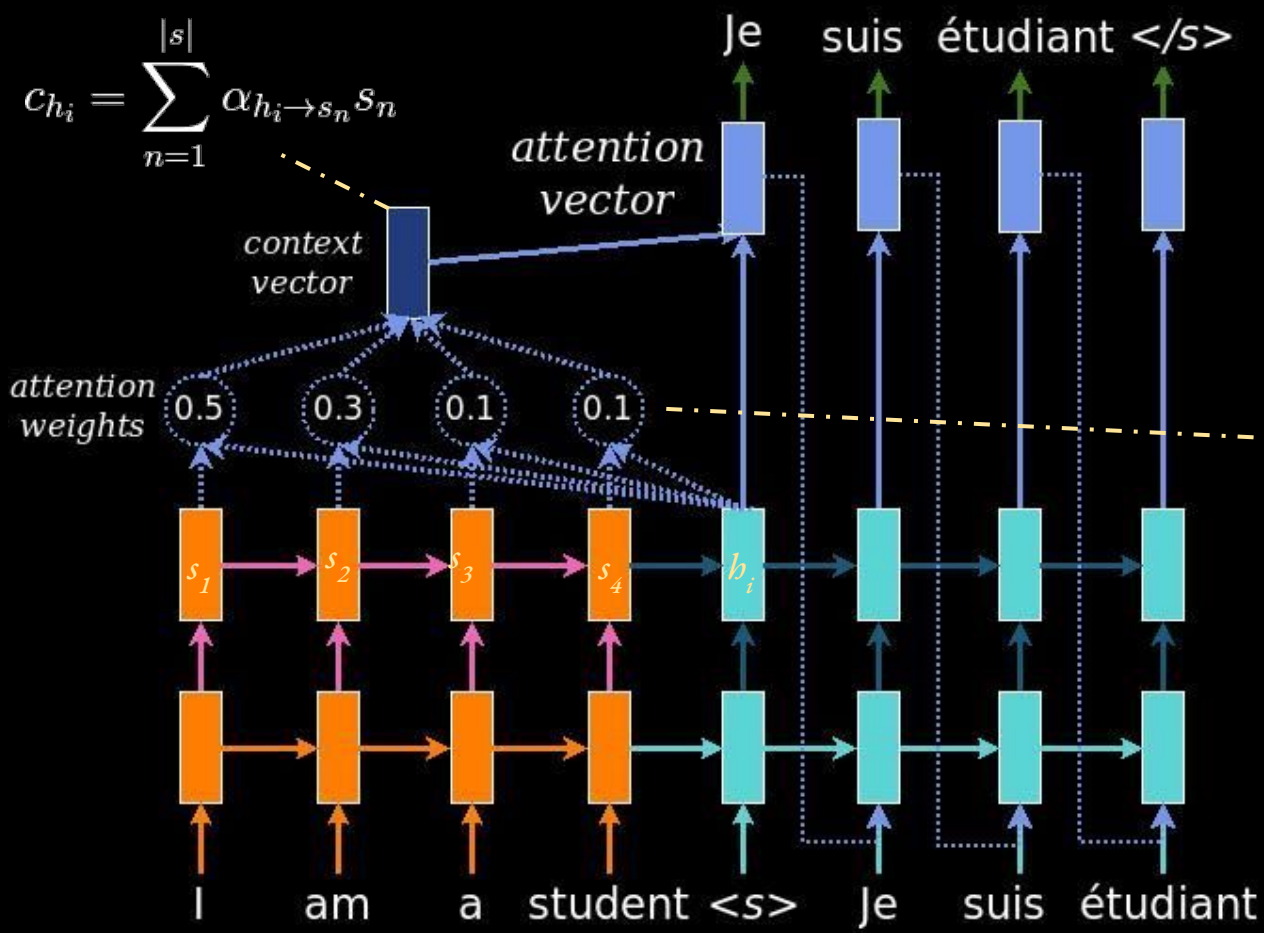
Attention

(Bahdanau et al., 2015)



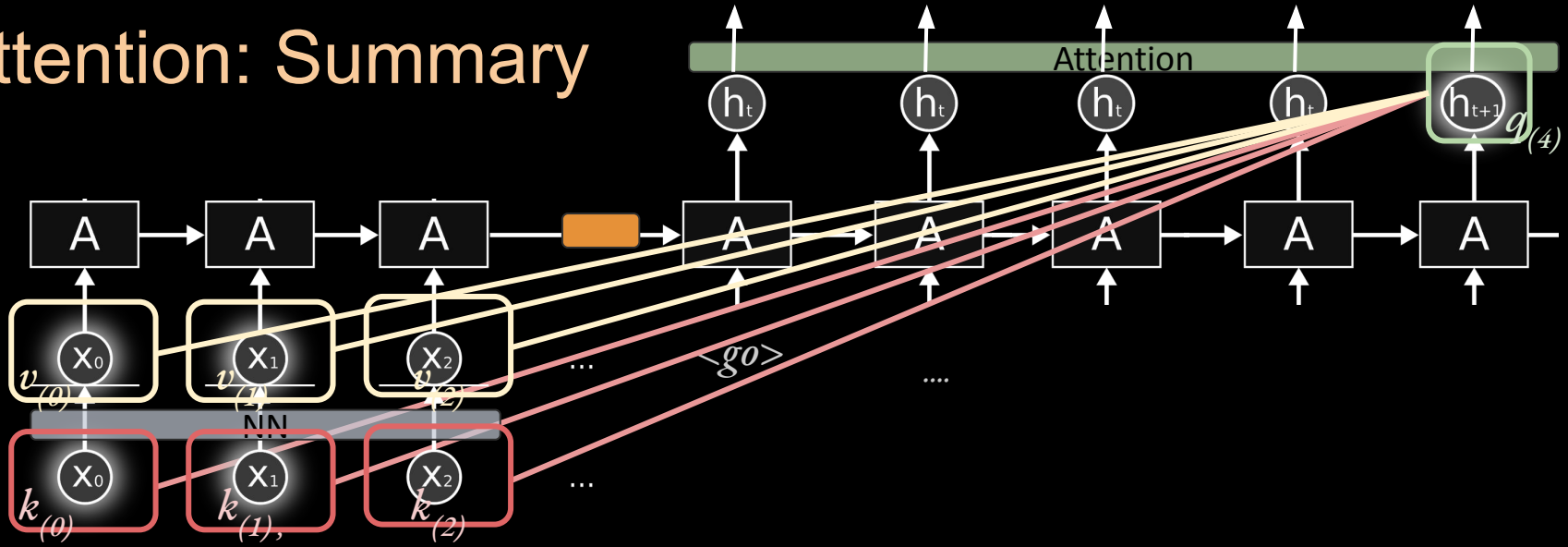
$$c_{h_i} = \sum_{n=1}^{|s|} \alpha_{h_i \to s_n} s_n$$

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

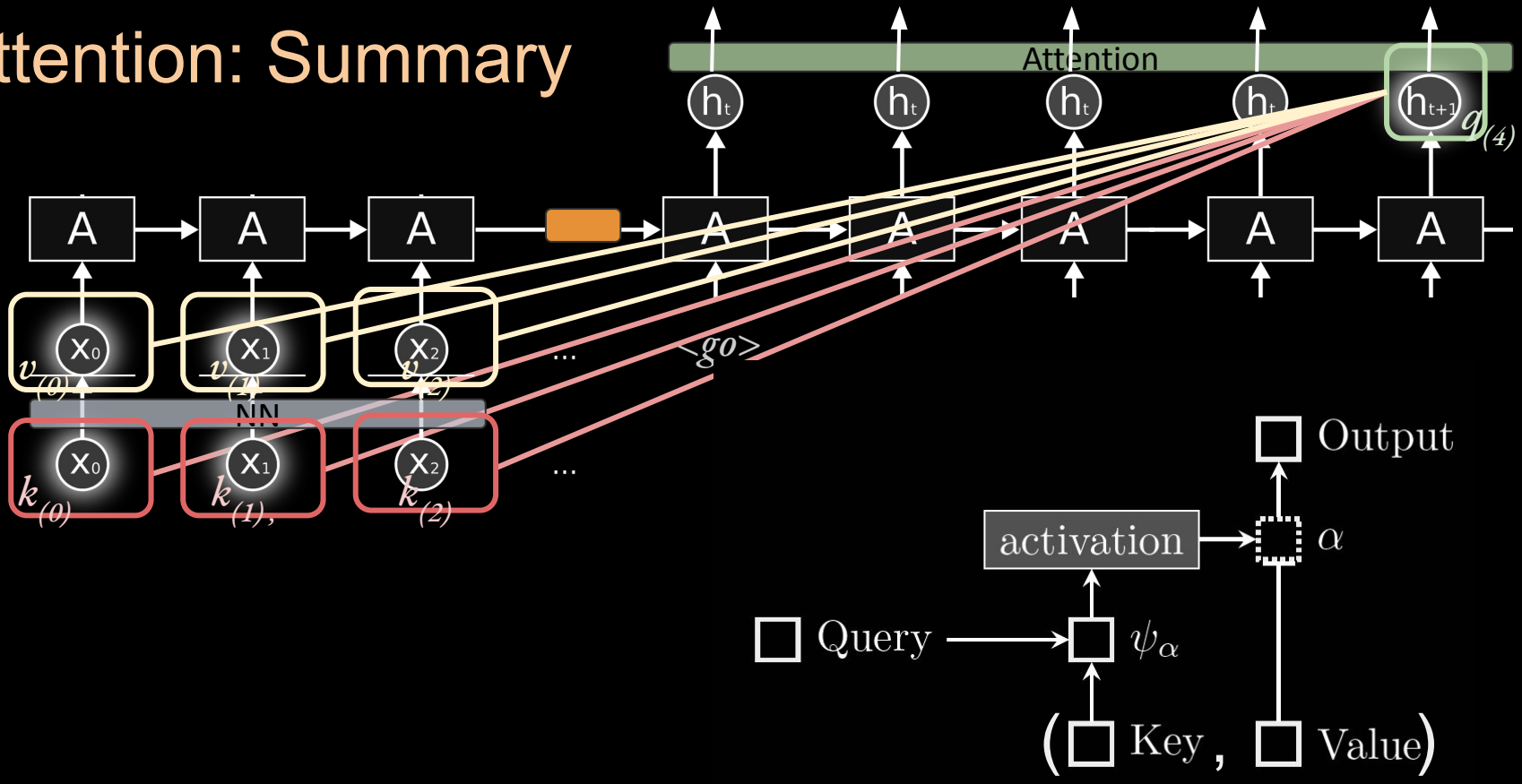


("synced", 2017)

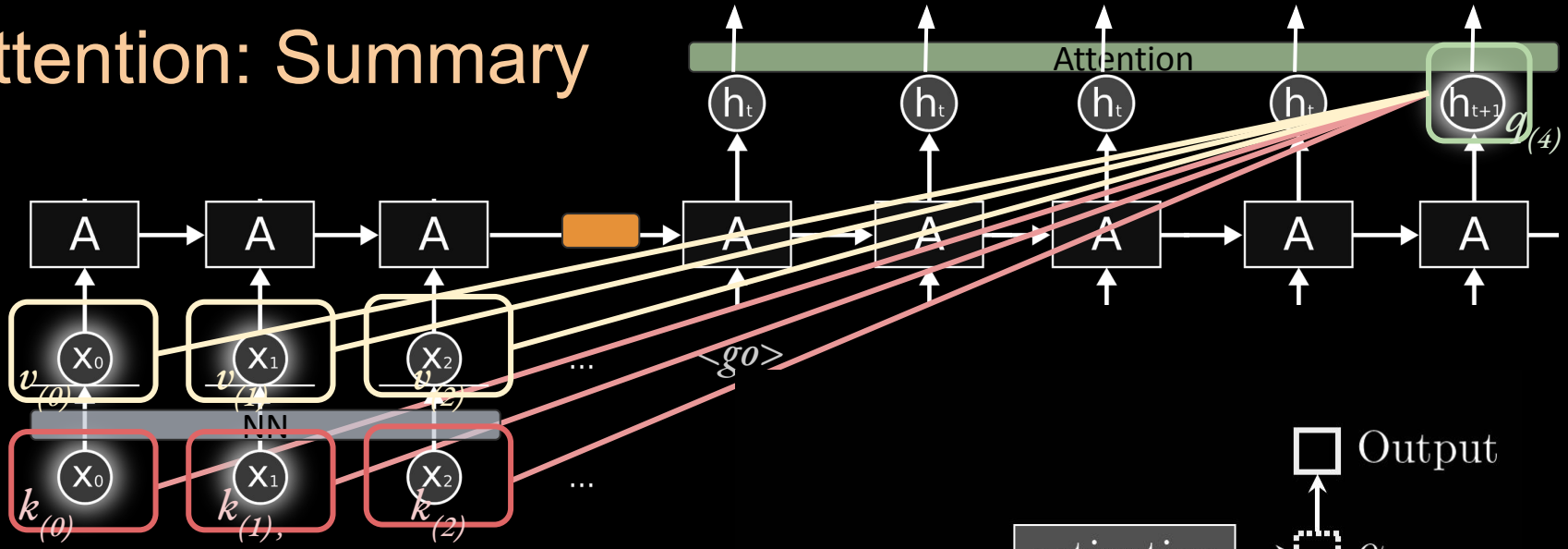
Attention: Summary

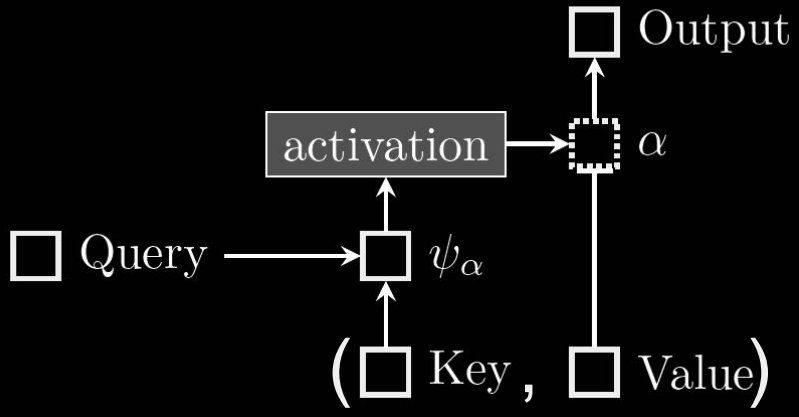


Attention: Summary



Attention: Summary





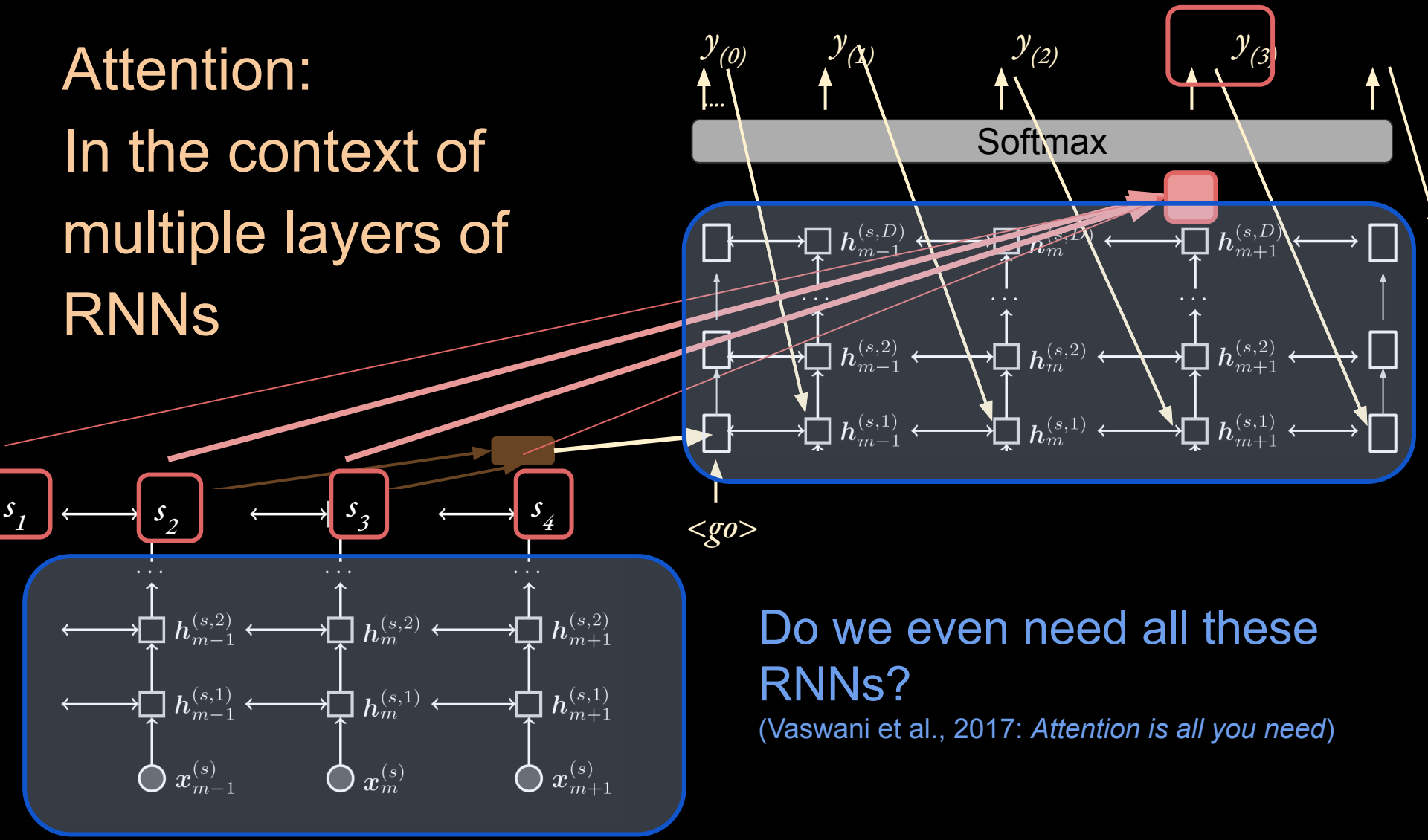
all equations with dot-product score:

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

$$\alpha_{q_i \to k} = softmax(\psi_{dp}(q_i, k))$$

$$o_{q_i} = \sum_{j=1}^{|k|} \alpha_{q_i \to k_j} v_j$$

Attention:
In the context of multiple layers of RNNs



Do we even need all these RNNs?

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

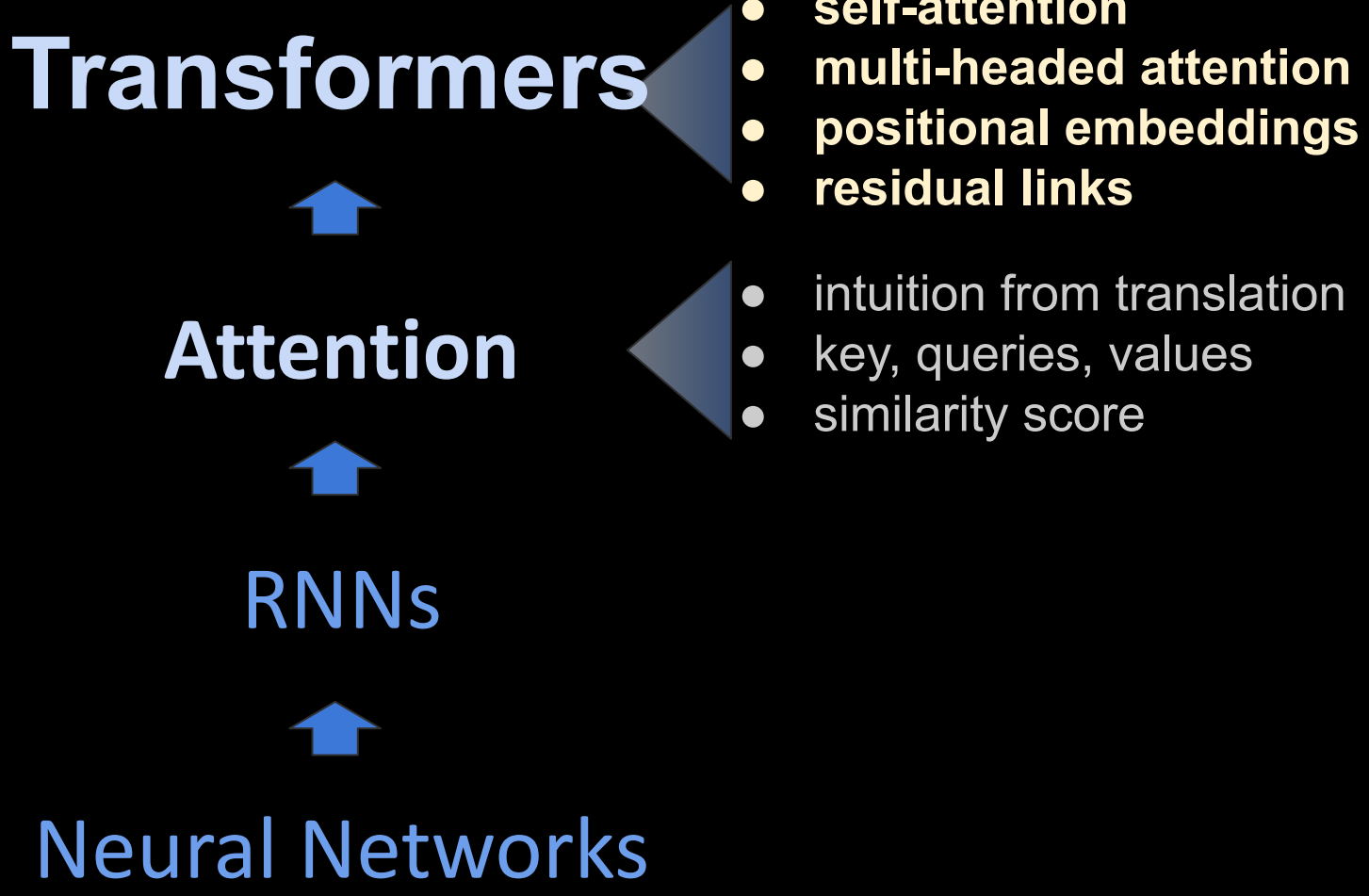
Evolution of Sequence Modeling

RNNs

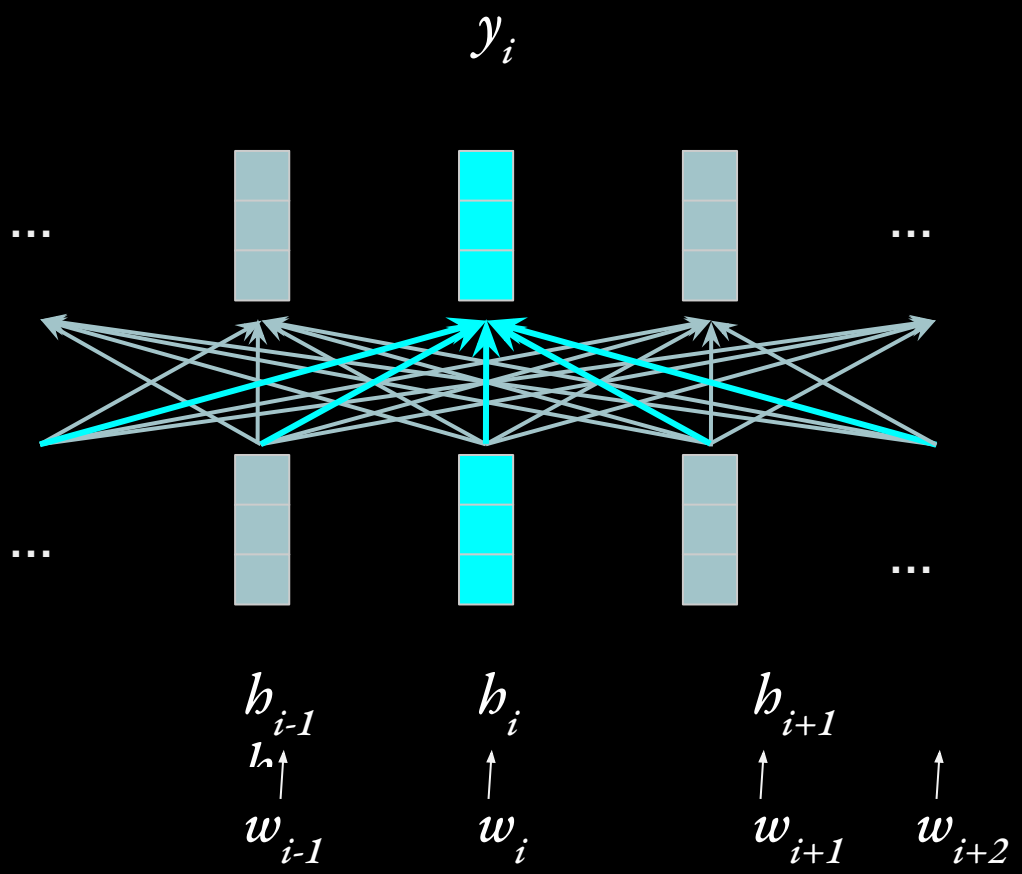
-> GRU/LSTMs-RNN

-> LSTMS with Attention

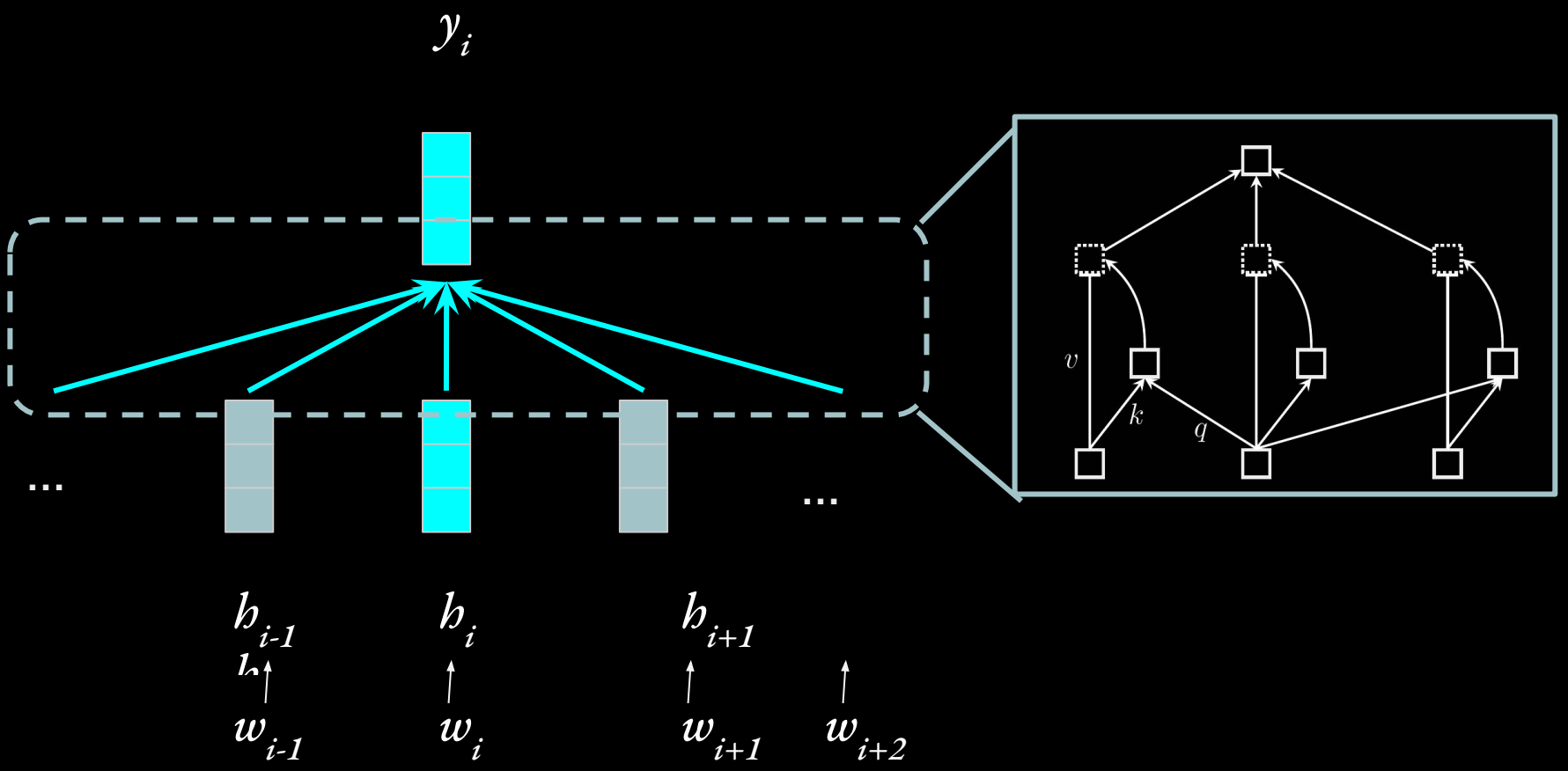
-> Attention without RNN


Transformers
self-attention
multi-headed attention
positional embeddings
residual links
Attention
intuition from translation
key, queries, values
similarity score
RNNs
Neural Networks

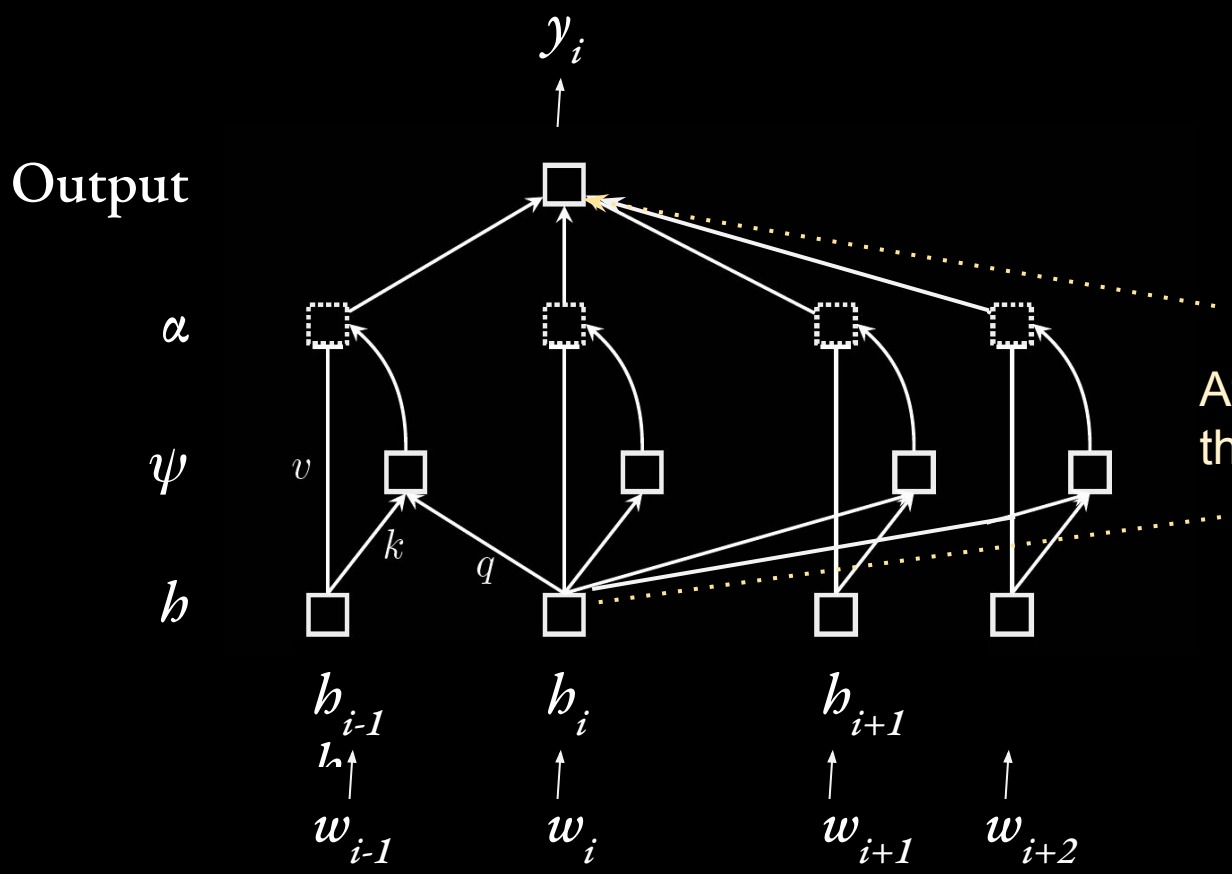
Self-Attention



Self-Attention

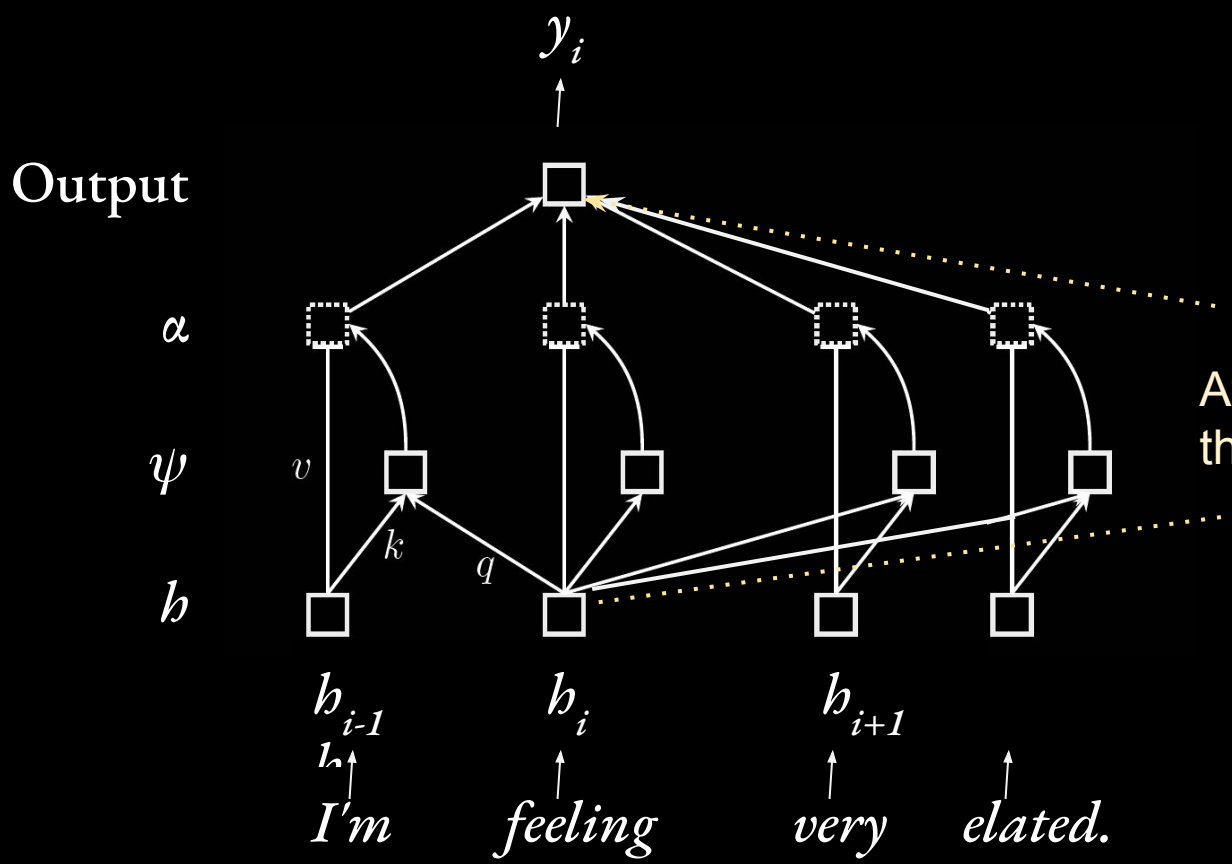


Self-Attention



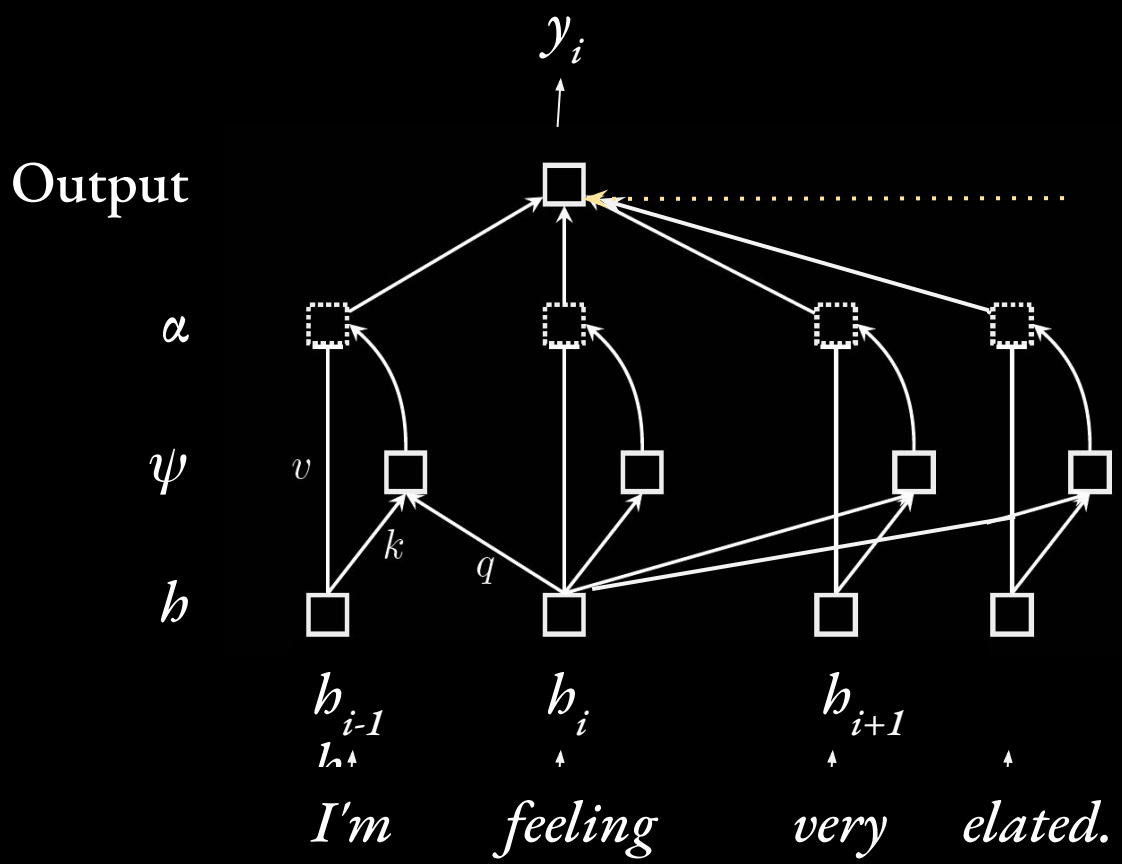
Attend to all other words in the sequence

Self-Attention



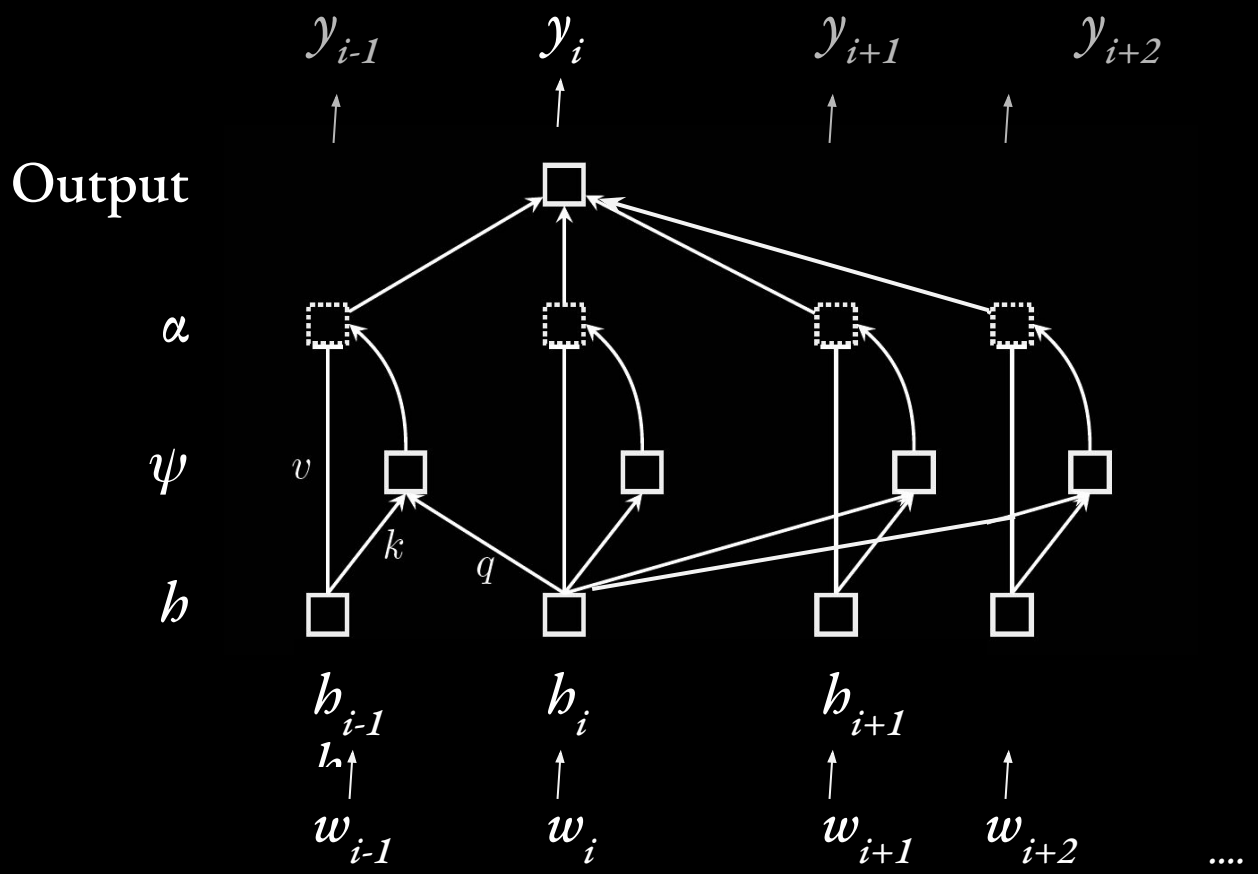
Attend to all other words in the sequence

Self-Attention

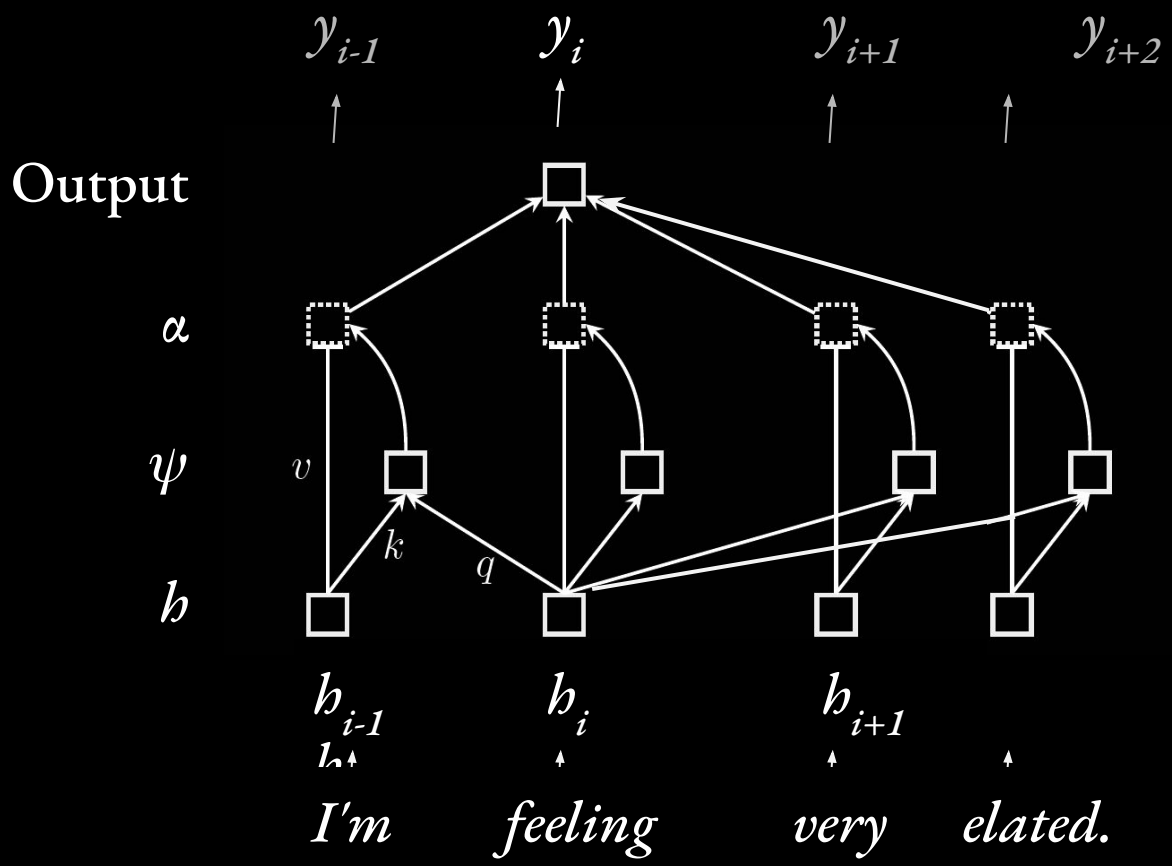


A weighted combination of other words' vectors.

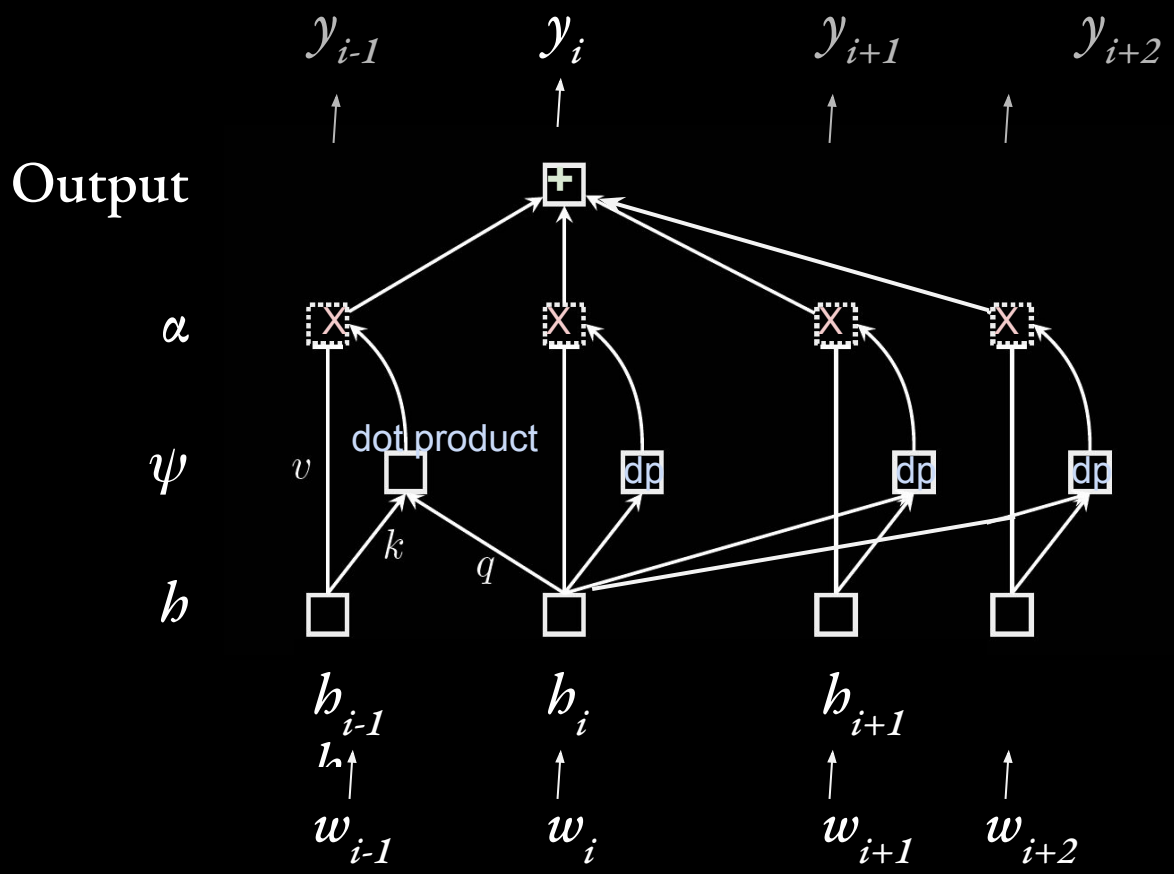
Self-Attention



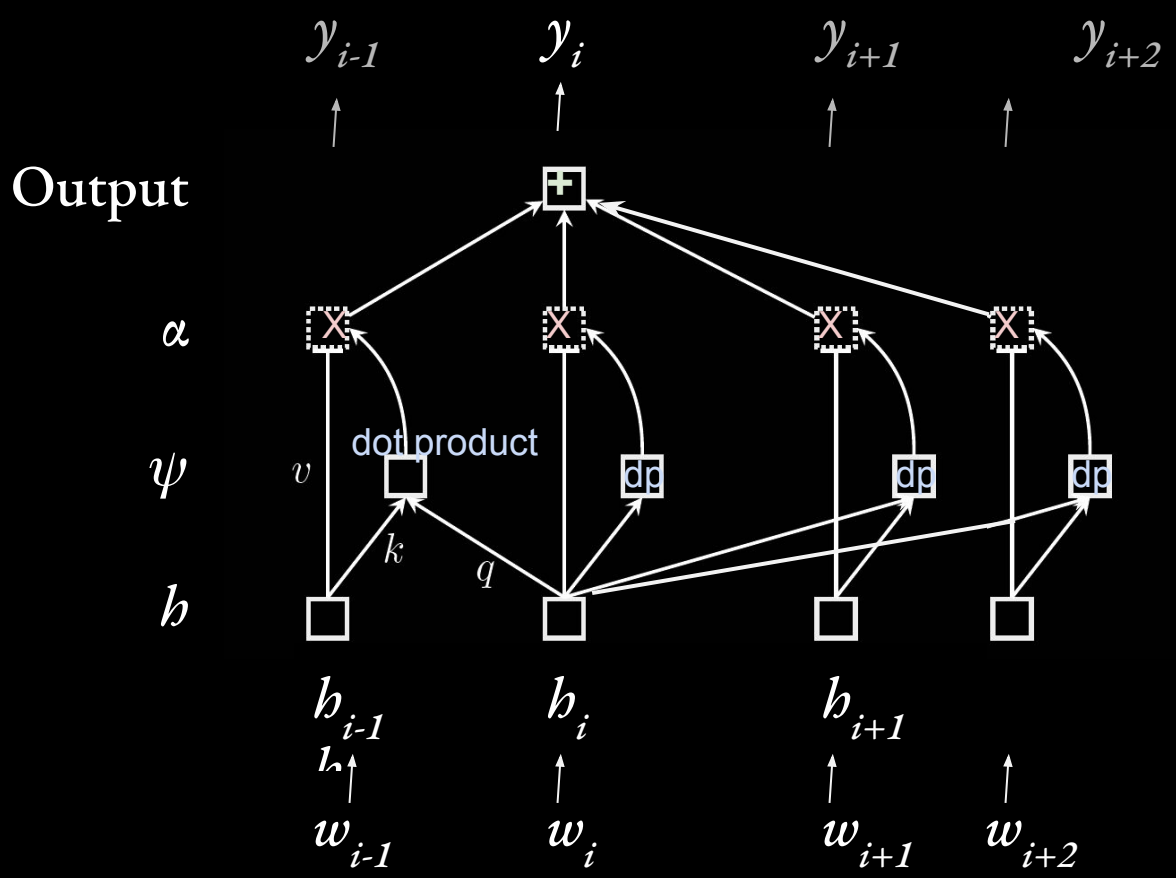
The Transformer's Heart: Self-Attention



Self-Attention



Self-Attention

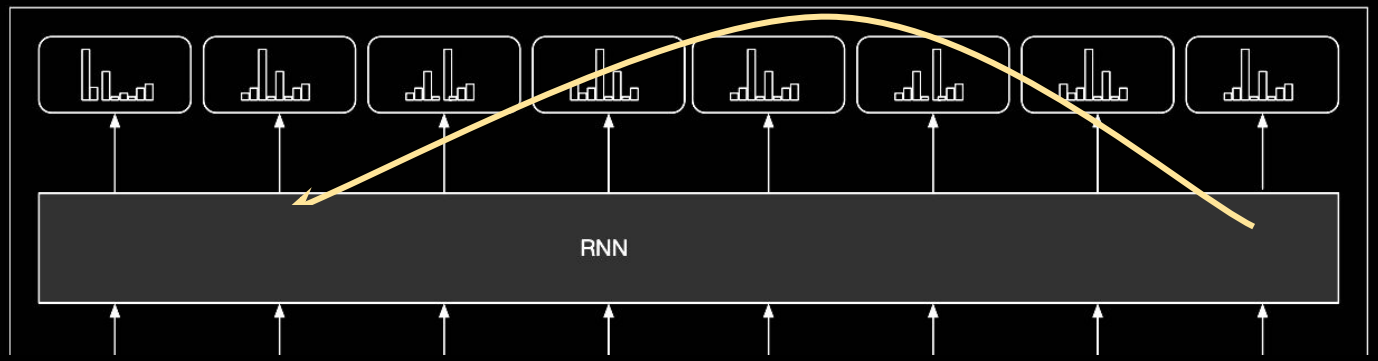


scaling parameter

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

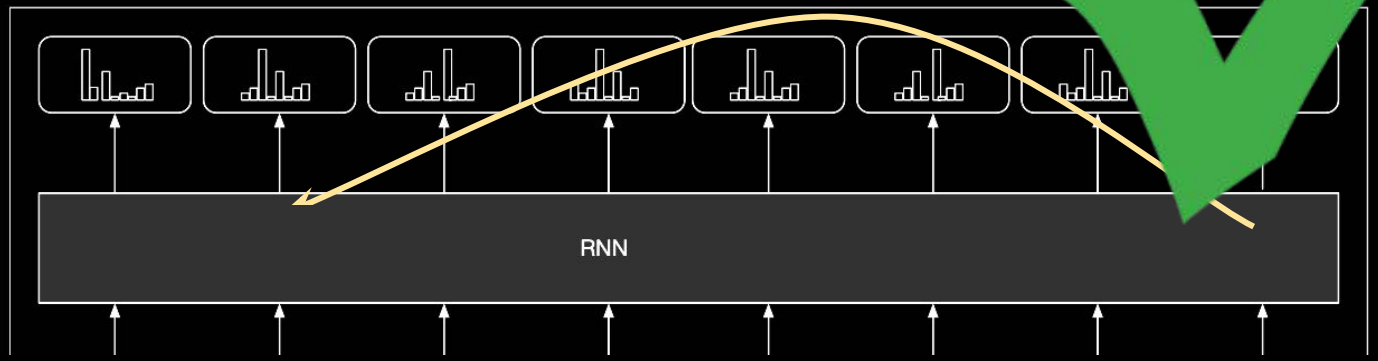
RNN Limitation:
Losing Track of Long Distance Dependencies

The horse which was raced past the barn tripped .

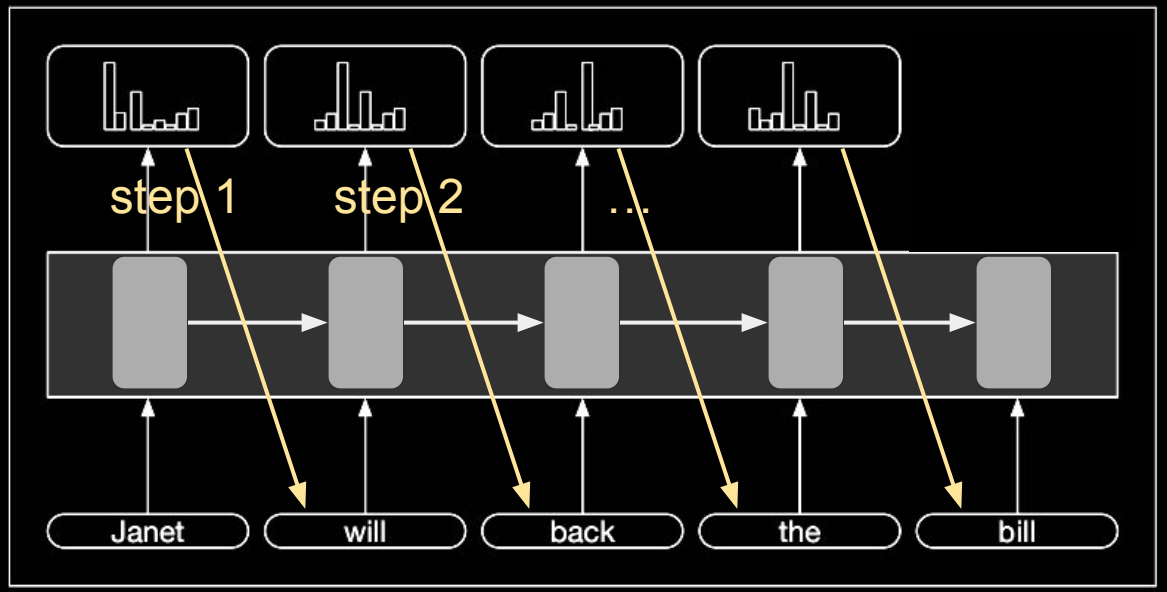


RNN Limitation:
Losing Track of Long Distance Dependencies

The horse which was raced past the barn tripped .



RNN: Limitation: Not parallelizable



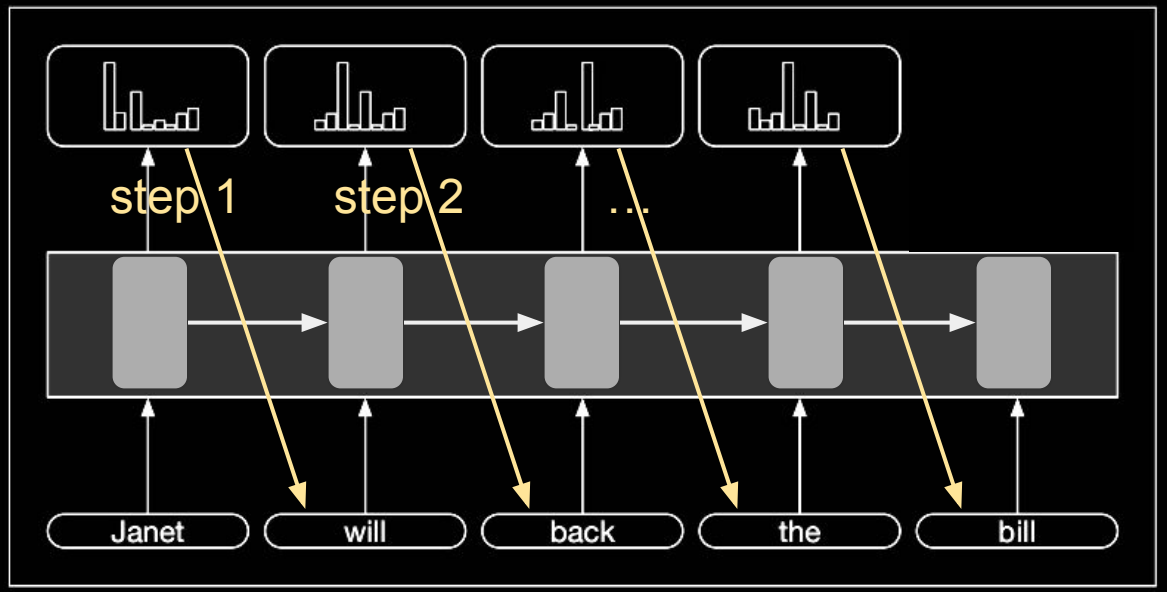
Language modeling with an RNN

Contextual Word Vectors

	Person A	Person B
How are you?	I feel ***fine*** –even ***great***!	My life is a ***great*** mess! I'm having a very hard time being happy.
What is going on?	Earlier, I ***played*** the ***game*** Yahtzee with my ***partner.*** I could not get that ***die*** to roll a 1! Now I'm ***lying*** on my bed for a ***rest***.	My business ***partner*** was ***lying*** to me. He was trying to ***game*** the system and ***played*** me. I think I am going to ***die*** –he left and now I have to pay the ***rest*** of his ***fine.***

(Kjell, Kjell, and Schwartz, 2023)

RNN: Limitation: Not parallelizable



Language modeling with an RNN

The Transformer: Motivation

- Capture long-distance dependencies
- Preserving sequential distances / periodicity
- Capture multiple relationships
- Easy to parallelize -- don’t need sequential processing.

Introducing the Transformer

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.0 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature.

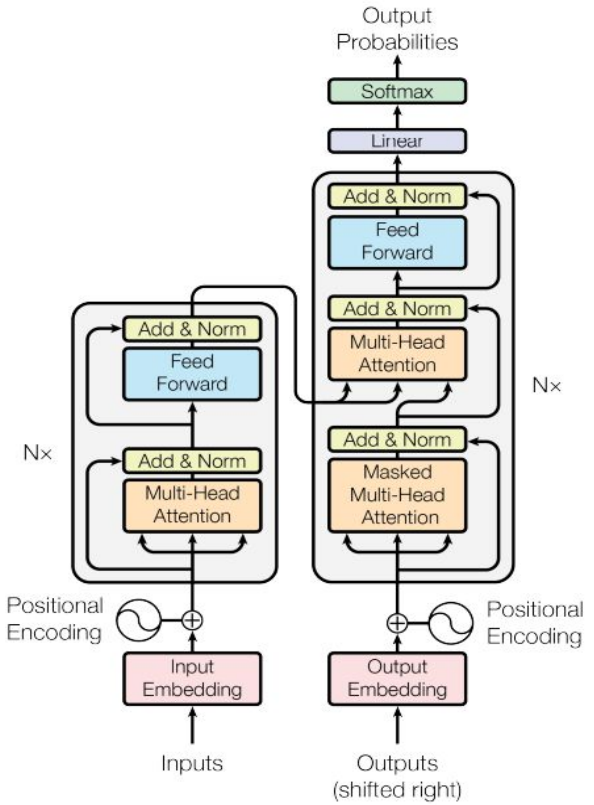


Figure 1: The Transformer - model architecture.

Introducing the Transformer

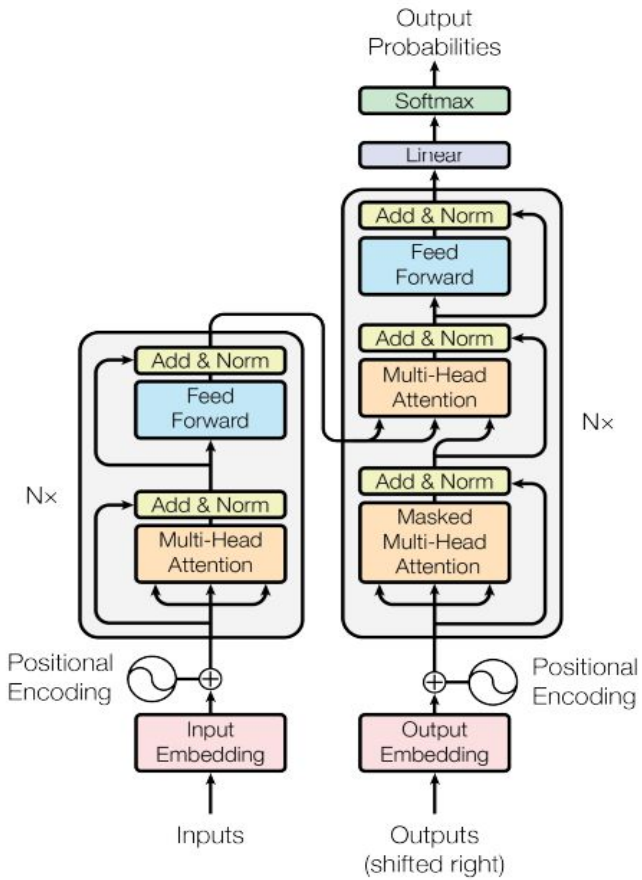
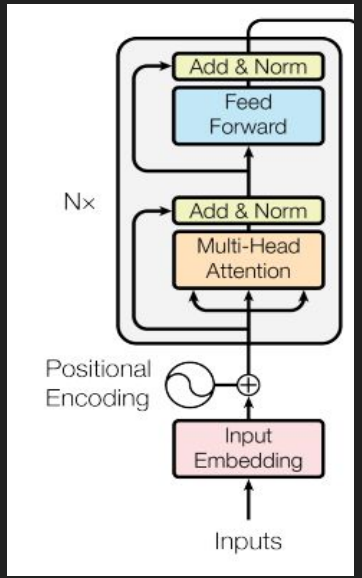
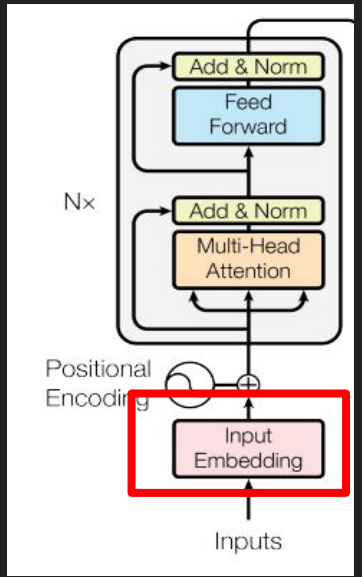


Figure 1: The Transformer - model architecture.

Encoder


Add & Norm
Feed Forward
Nx
Add & Norm
Multi-Head Attention
Positional Encoding
Input Embedding
Inputs

Encoder: Input Embedding



Input Embedding

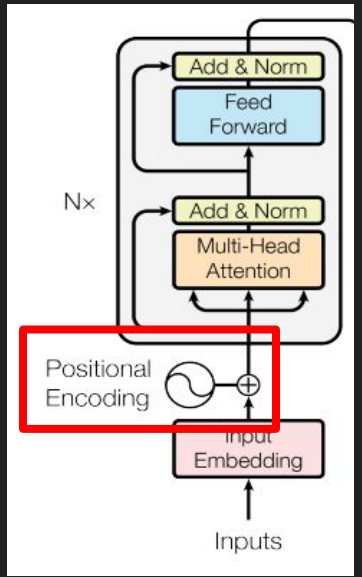
Original Sentence

Tokenization

Input IDs
(embedding lookup: position in the vocab - FIXED)

Embeddings
(vector of size d_{model} = 512 or 1024 or … LEARNED)

Encoder: Positional Encoding



Positional Encoding

Original Sentence
(tokens)

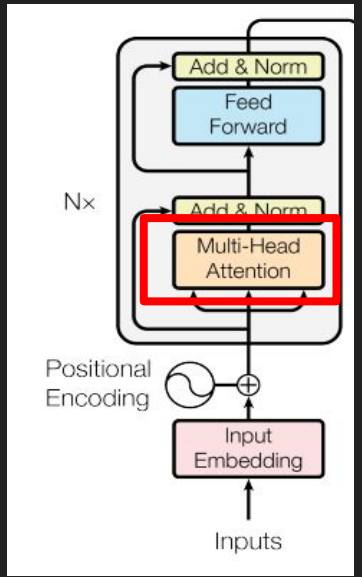
Embeddings
(vector of size d_{model}= 512 or 1024 or …
Learned)

Positional Embedding
(vector of size d_{model}= 512 or 1024 or …
Can be Learned or FIxed)

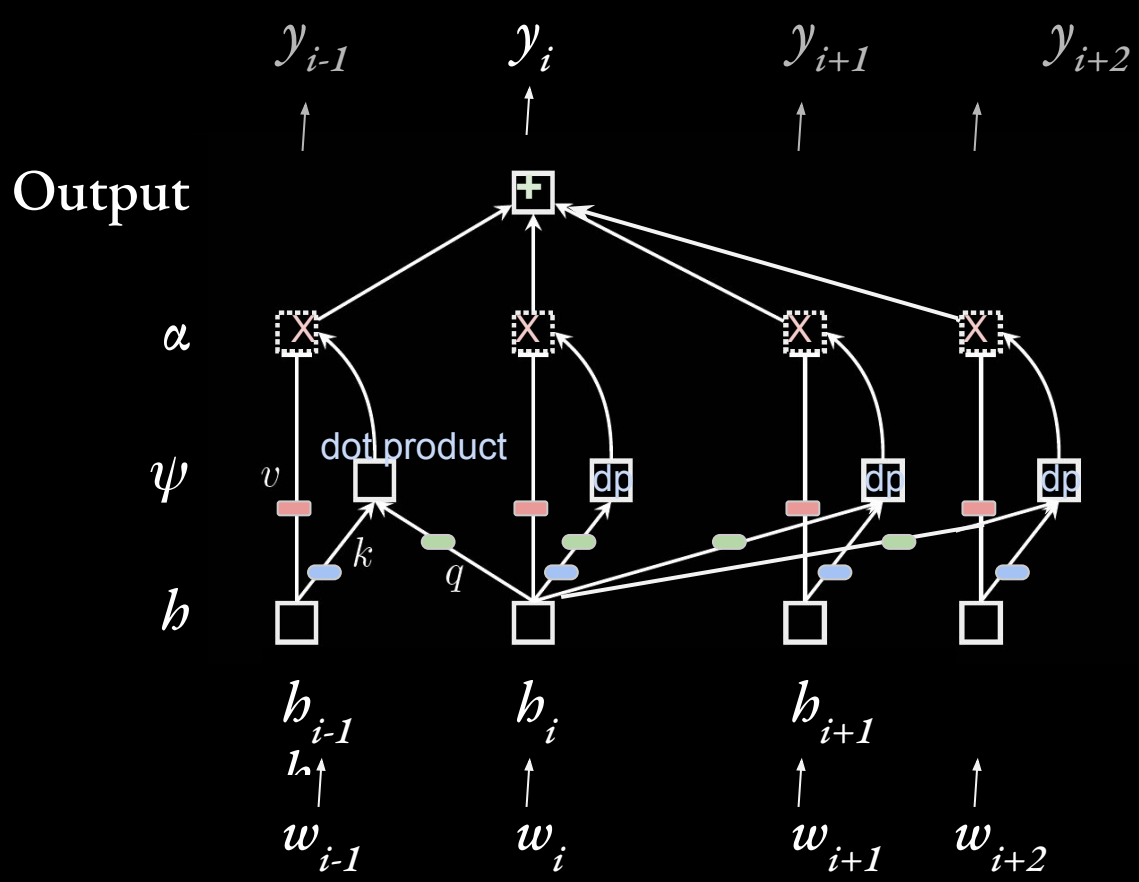
e.g.

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

Encoder: Multi-Head Attention



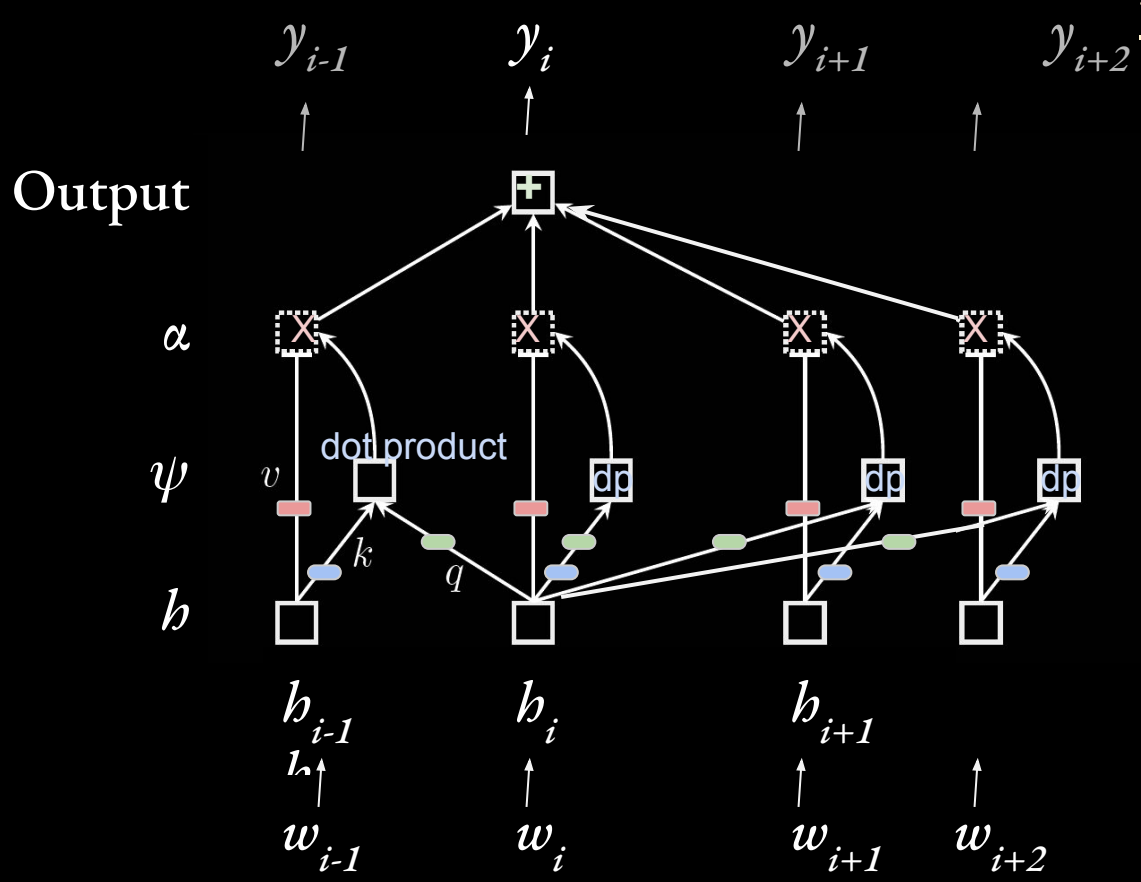
Specs for Transformer's Self-Attention



Two Additions:

1. Linearly parameterized Q, K, and V:
   - $q = W_q^T h_i$
   - $k = W_k^T h_i$
   - $v = W_v^T h_i$
2.

Specs for Transformer's Self-Attention



Two Additions:

1. Linearly parameterized Q, K, and V:
   - q = $W_q^T h_i$
   - k = $W_k^T h_i$
   - v = $W_v^T h_i$
2. Scaling parameter for score

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

$$\text{where } \sigma = \frac{1}{\sqrt{d_k}}$$

$$\psi_{dp}(q,k) = \frac{(qk^t)}{\sqrt{d}}$$

Specs for Transformer's Self-Attention

$$\text{attn\_head} = \text{attn}(W_q^t h, W_k^t h, W_v^t h)$$

$$\text{attn}(q, k, v) = \text{softmax}\left(\frac{qk^t}{\sqrt{d_k}}\right) v$$

Two Additions:

1. Linearly parameterized Q, K, and V:
   - q = $W_q^T h_i$
   - k = $W_k^T h_i$
   - v = $W_v^T h_i$
2. Scaling parameter for score

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

$$\text{where } \sigma = \frac{1}{\sqrt{d_k}}$$

$$\psi_{dp}(q, k) = \frac{(qk^t)}{\sqrt{d}}$$

Self-Attention in PyTorch

```
import nn.functional as f
class SelfAttention(nn.Module):
    def __init__(self, h_dim:int):
        self.Q = nn.Linear(h_dim, h_dim) #1 head
        self.K = nn.Linear(h_dim, h_dim)
        self.V = nn.Linear(h_dim, h_dim)


    def forward(hidden_states:torch.Tensor):
        v = self.V(hidden_states)
        k = self.K(hidden_states)
        q = self.Q(hidden_states)
        attn_scores = torch.matmul(q, k.T)
        attn_probs = f.Softmax(attn_scores)

        context = torch.matmul(attn_probs, v)
        return context
```

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

Linear layer:
$W^T X$

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

Self-Attention in PyTorch

```
import nn.functional as f
class SelfAttention(nn.Module):
    def __init__(self, h_dim:int):
        self.Q = nn.Linear(h_dim, h_dim) #1 head
        self.K = nn.Linear(h_dim, h_dim)
        self.V = nn.Linear(h_dim, h_dim)
        self.dropout = nn.dropout(p=0.1)

    def forward(hidden_states:torch.Tensor):
        v = self.V(hidden_states)
        k = self.K(hidden_states)
        q = self.Q(hidden_states)
        attn_scores = torch.matmul(q, k.T)
        attn_probs = f.Softmax(attn_scores)
        attn_probs = self.dropout(attn_probs)
        context = torch.matmul(attn_probs, v)
        return context
```

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

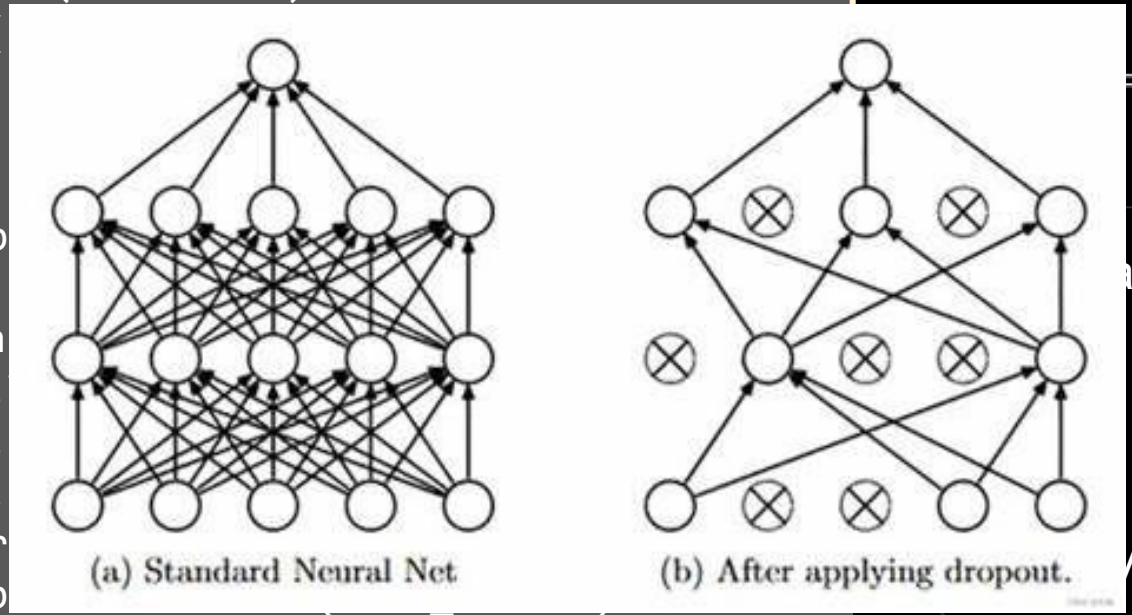
Linear layer:
$W^T X$

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

Self-Attention in PyTorch

```
import nn.functional as f
class SelfAttention(nn.Module):
    def __init__(
        self.Q =
        self.K =
        self.V =
        self.drop

    def forward(h
        v = self.
        k = self.
        q = self.
        attn_scor
        attn_prob
        attn_probs = self.dropout(attn_probs)
        context = torch.matmul(attn_probs, v)
        return context
```



(a) Standard Neural Net

(b) After applying dropout.

$= (qk^t)\,\sigma$

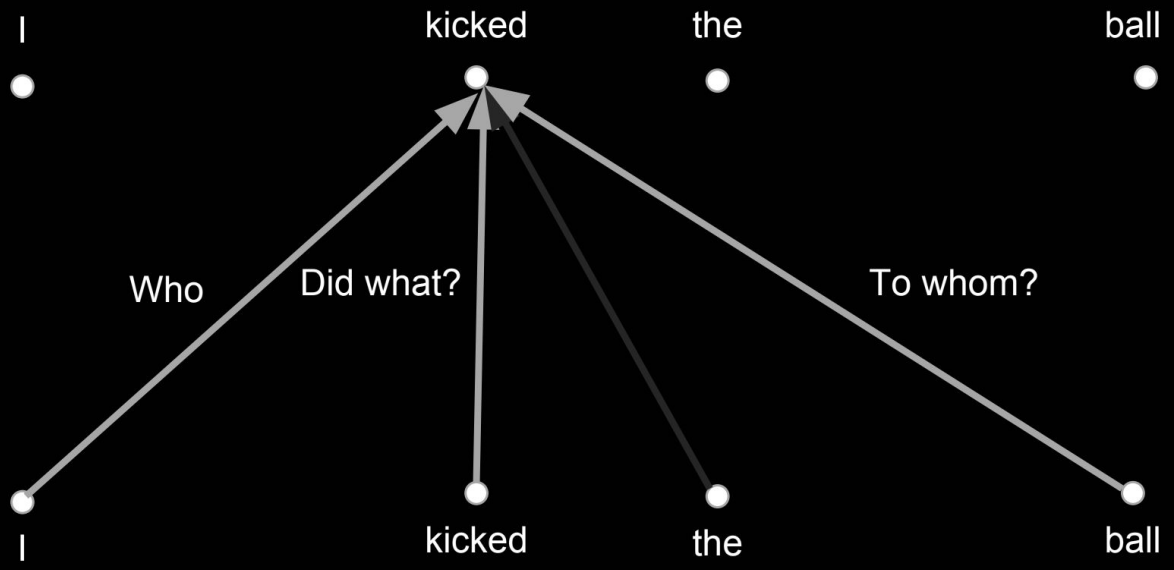
yer:

of weights
of for K,

Multi-headed Attention

Single-headed (standard self-attention)

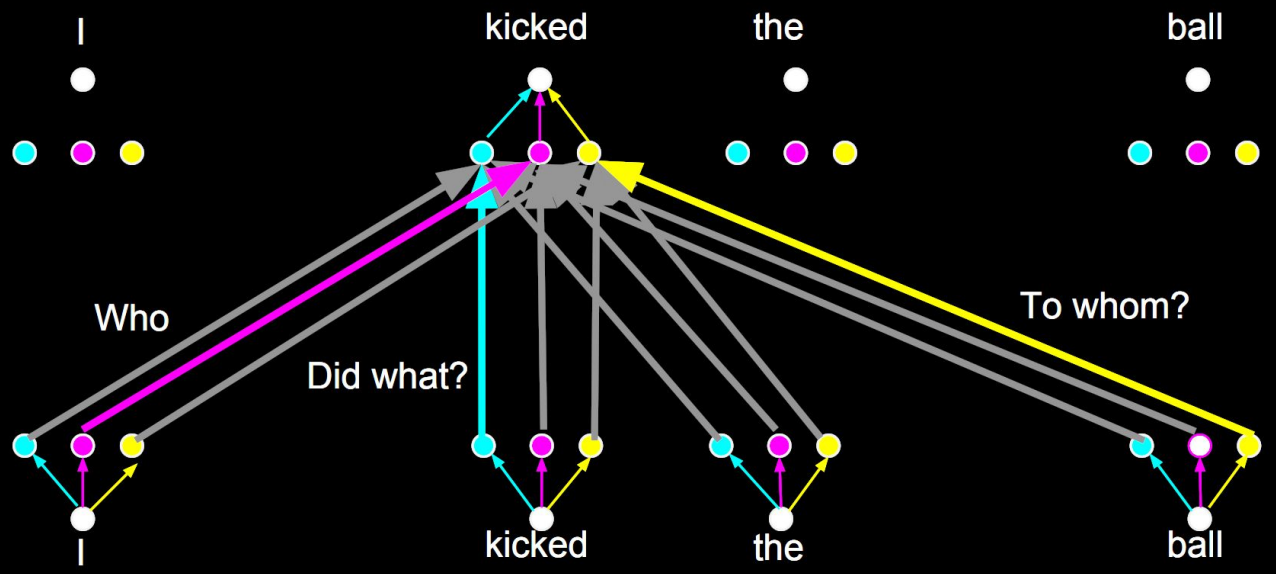
Limitation (thus far): Can't capture multiple types of dependencies between words.



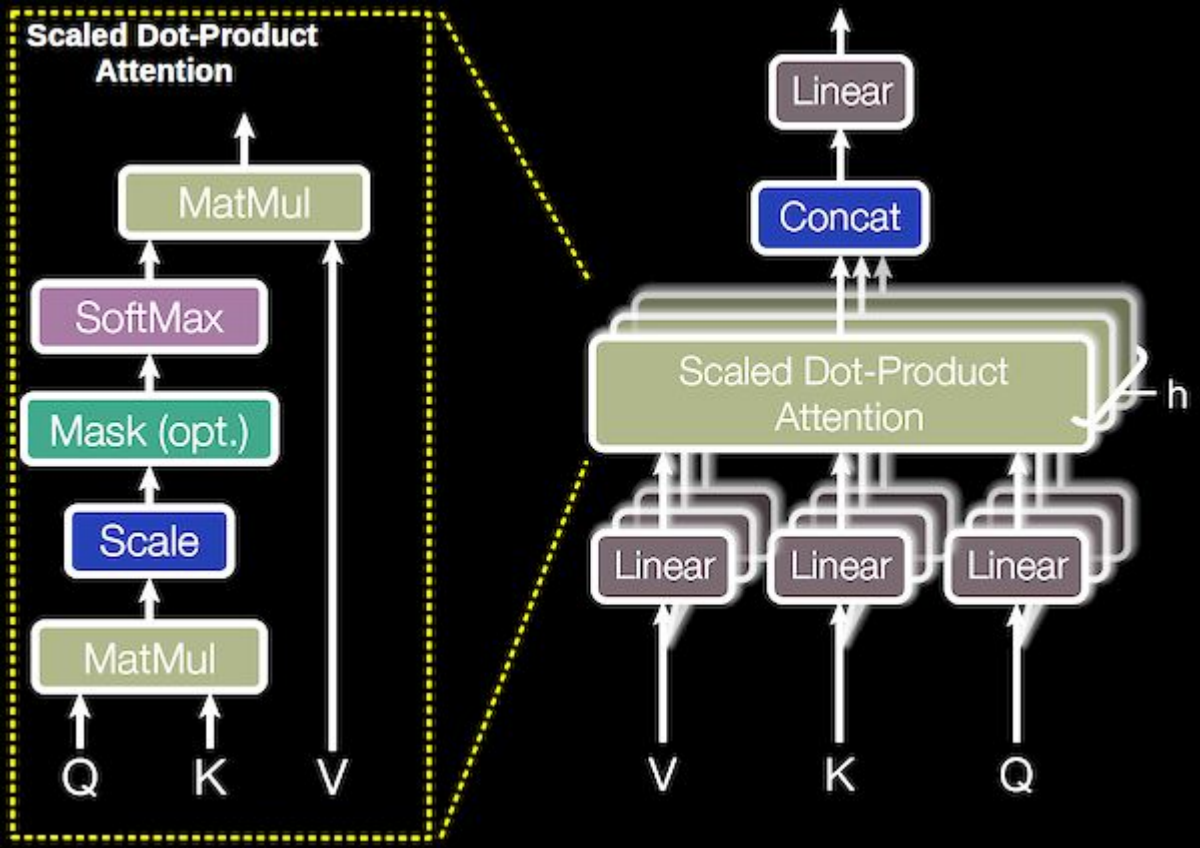
Multi-headed Attention

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

Solution: Multi-head attention



The Transformer: Multi-headed Attention



Transformer's Multi-headed Attention

$$\text{MHAttn}(h) = Concat(\text{attn\_head}_1(h[:d], \text{attn\_head}_2(h[(d+1):2d]), \ldots, \text{attn\_head}_h(h[-d:]))W_o$$

$d = dims / num\_hs$

Builds on weighted, scaled dot-product

1. Linearly parameterized Q, K, and V:
   - $q = W_q^T h_i$
   - $k = W_k^T h_i$
   - $v = W_v^T h_i$
2. Scaling parameter for score

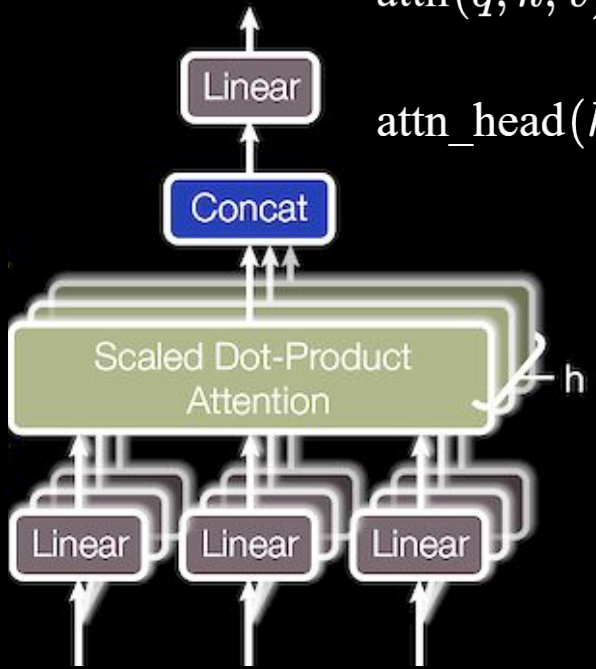
$$\psi_{dp}(q,k) = (qk^t)\sigma,$$

$$\text{where } \sigma = \frac{1}{\sqrt{d_k}}$$

$$\psi_{dp}(q,k) = \frac{(qk^t)}{\sqrt{d}}$$

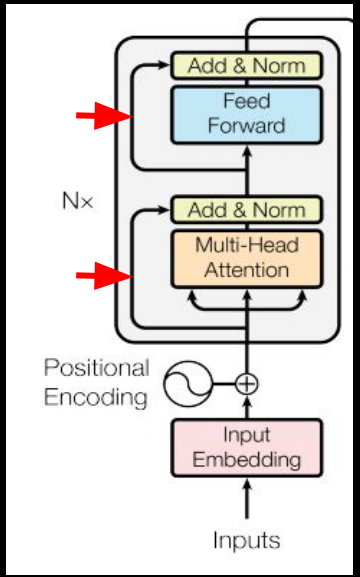
$$\text{attn}(q,k,v) = \text{softmax}\left(\frac{qk^t}{\sqrt{d_k}}\right)v$$

$$\text{attn\_head}(h) = attn(W_q^t h, W_k^t h, W_v^t h)$$



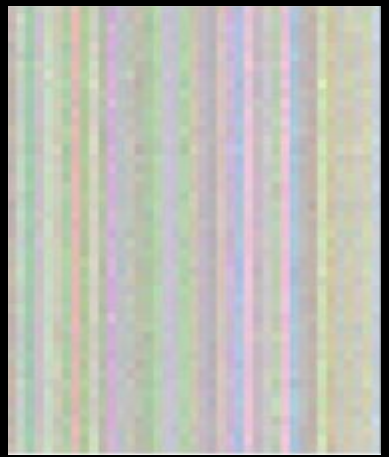
update "h"

Transformer's Residual Links



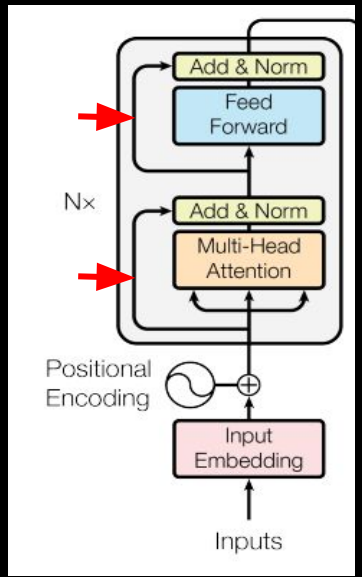
Transformer's Residual Links

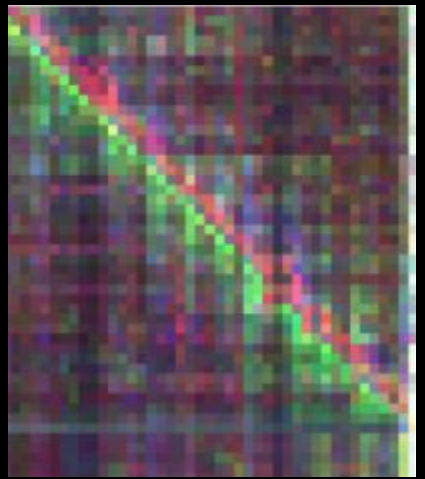
Heat matrix of attention weights from one layer (rows) to the next (cols).



without residuals

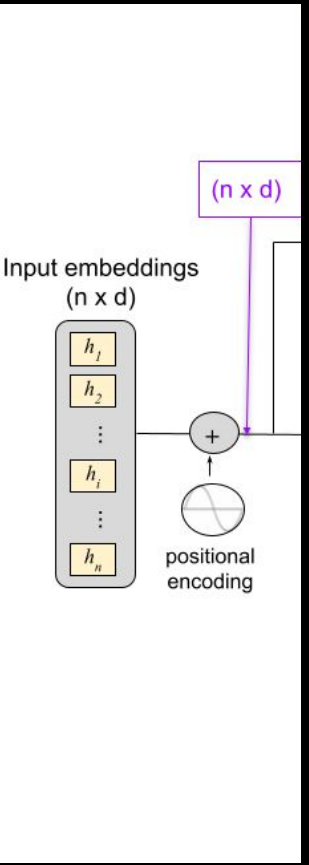
Output=LayerNorm(h+MHAtten(h))



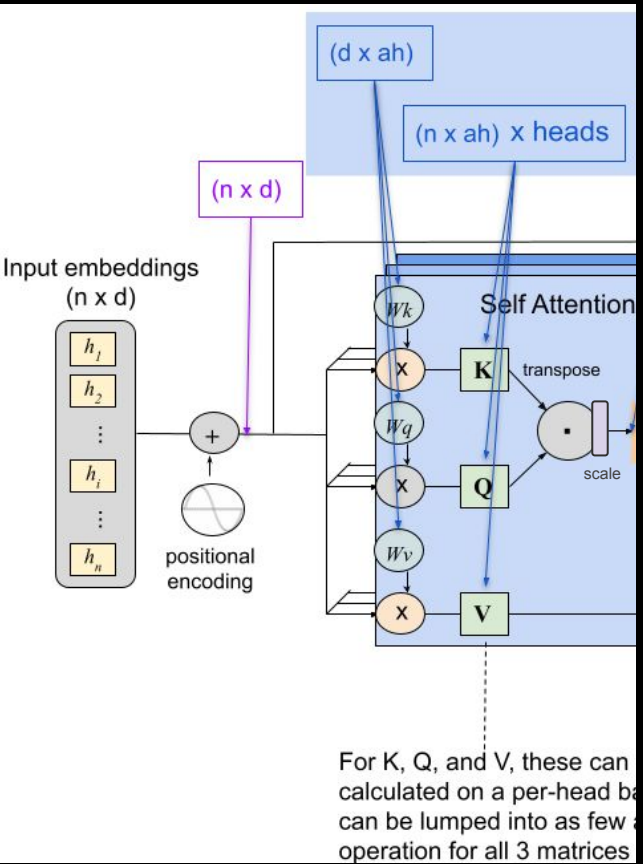


with residuals

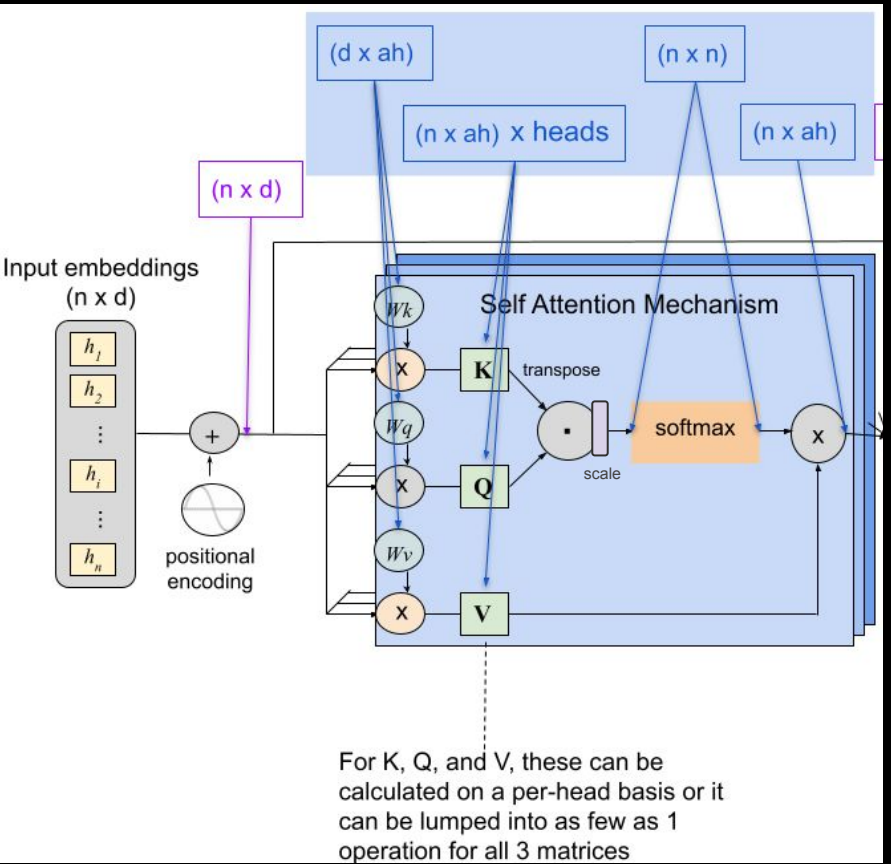
Detailed Overview of (HuggingFace) Transformer Matrices and Computation



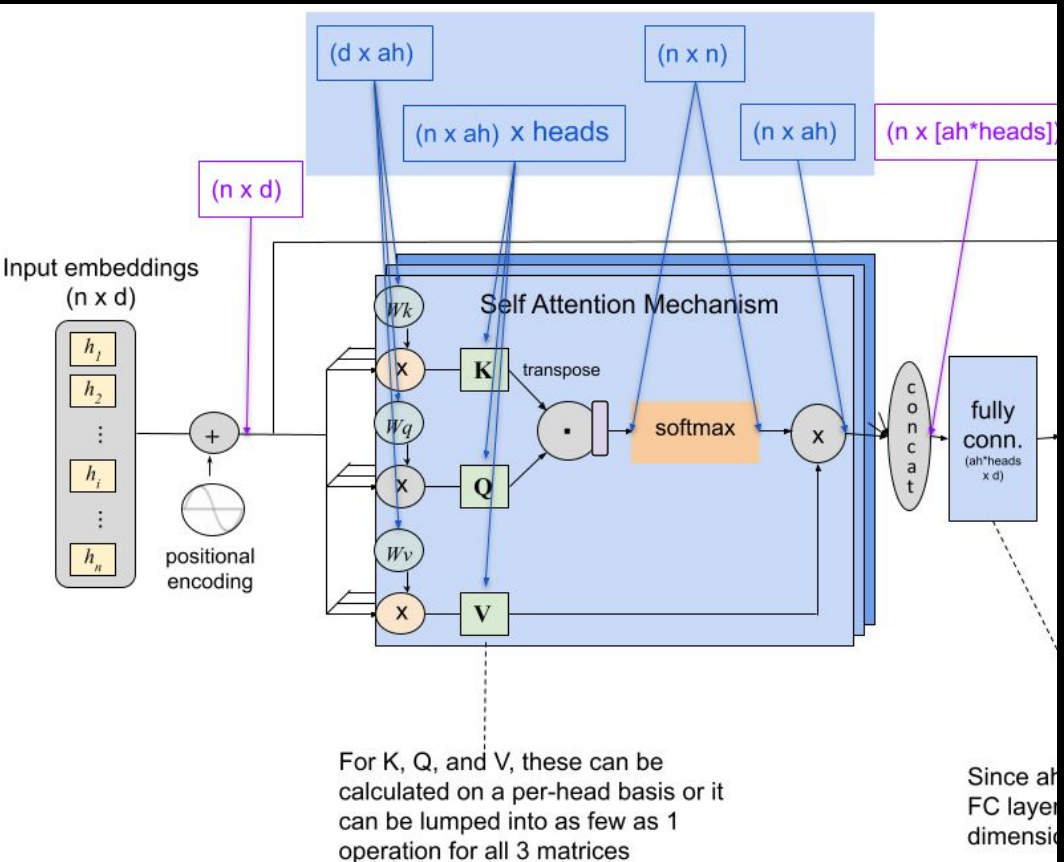
Detailed Overview of (HuggingFace) Transformer Matrices and Computation



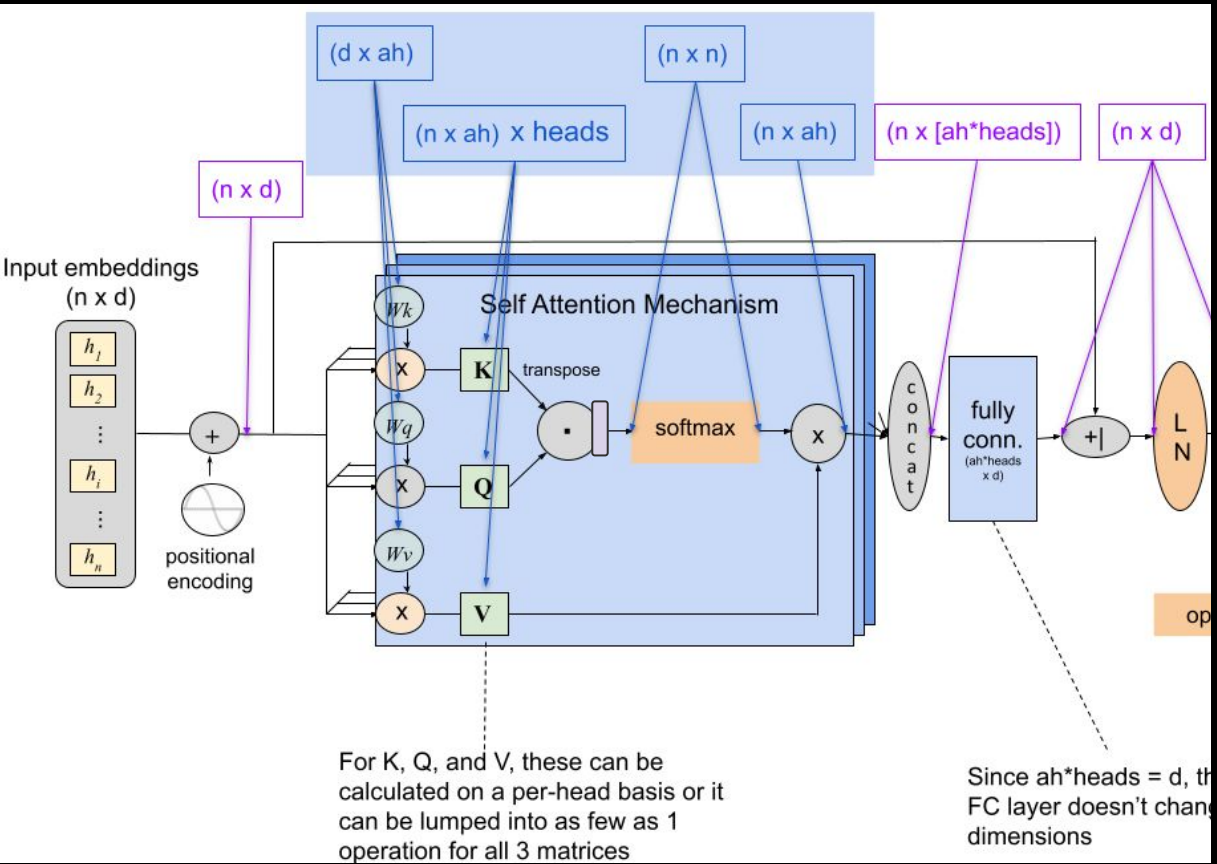
Detailed Overview of (HuggingFace) Transformer Matrices and Computation



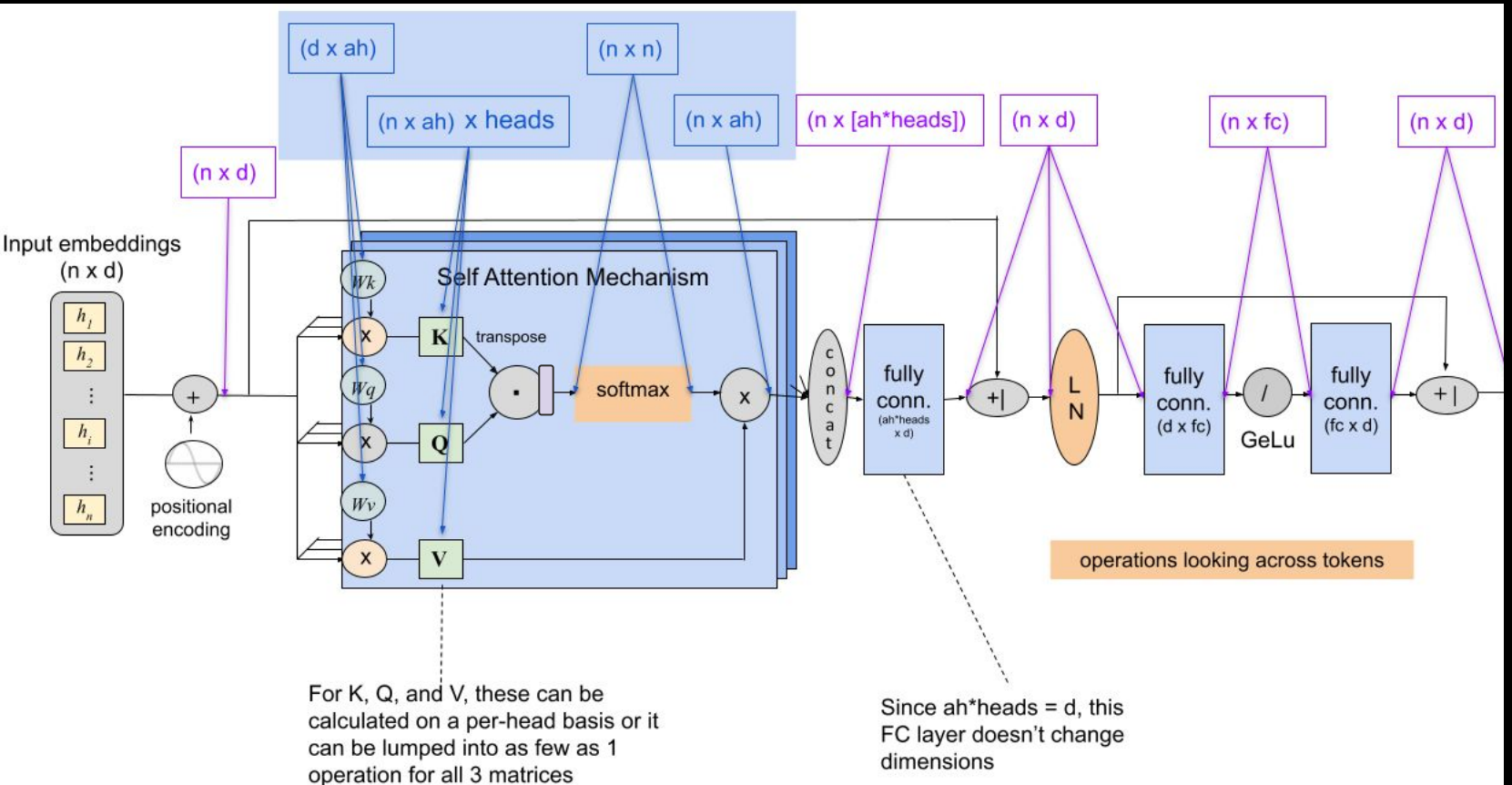
Detailed Overview of (HuggingFace) Transformer Matrices and Computation



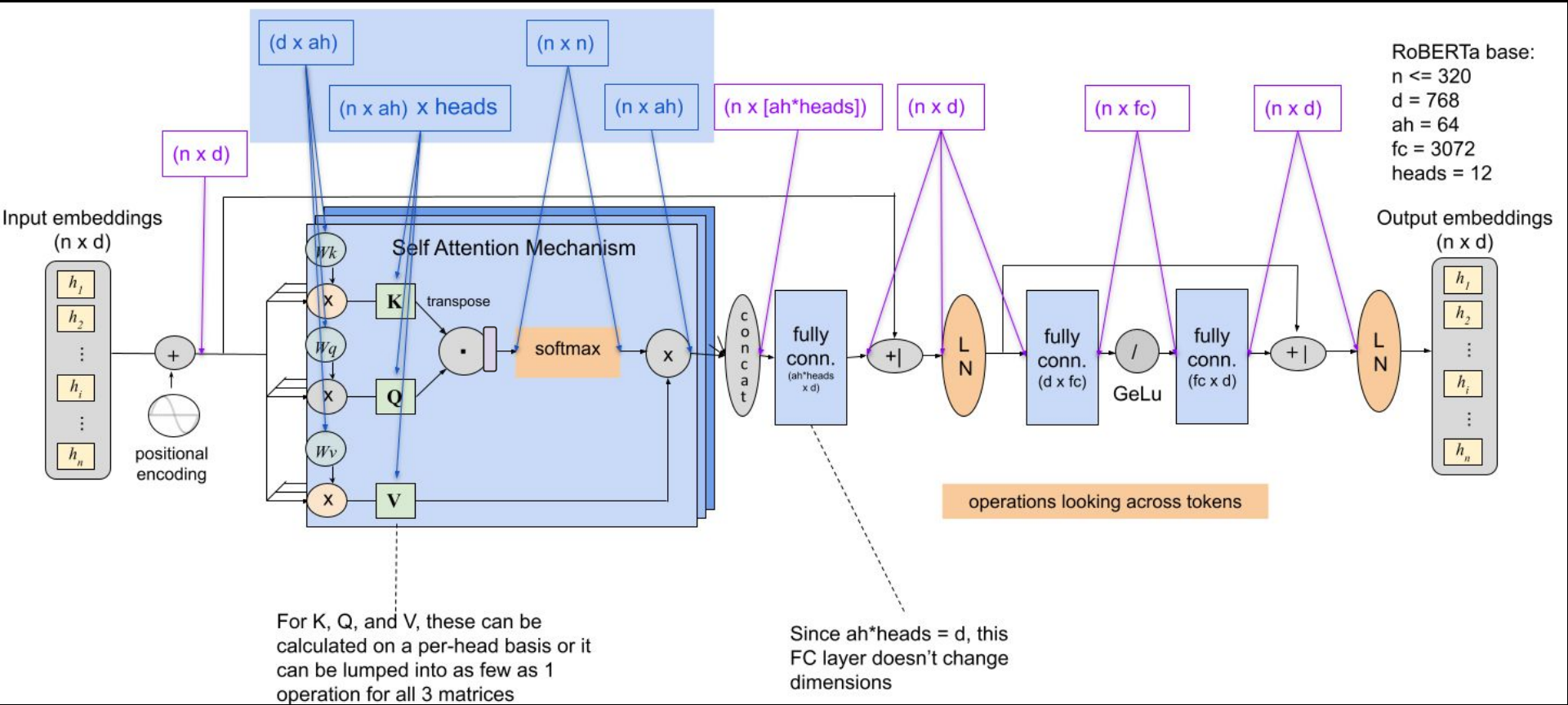
Detailed Overview of (HuggingFace) Transformer Matrices and Computation



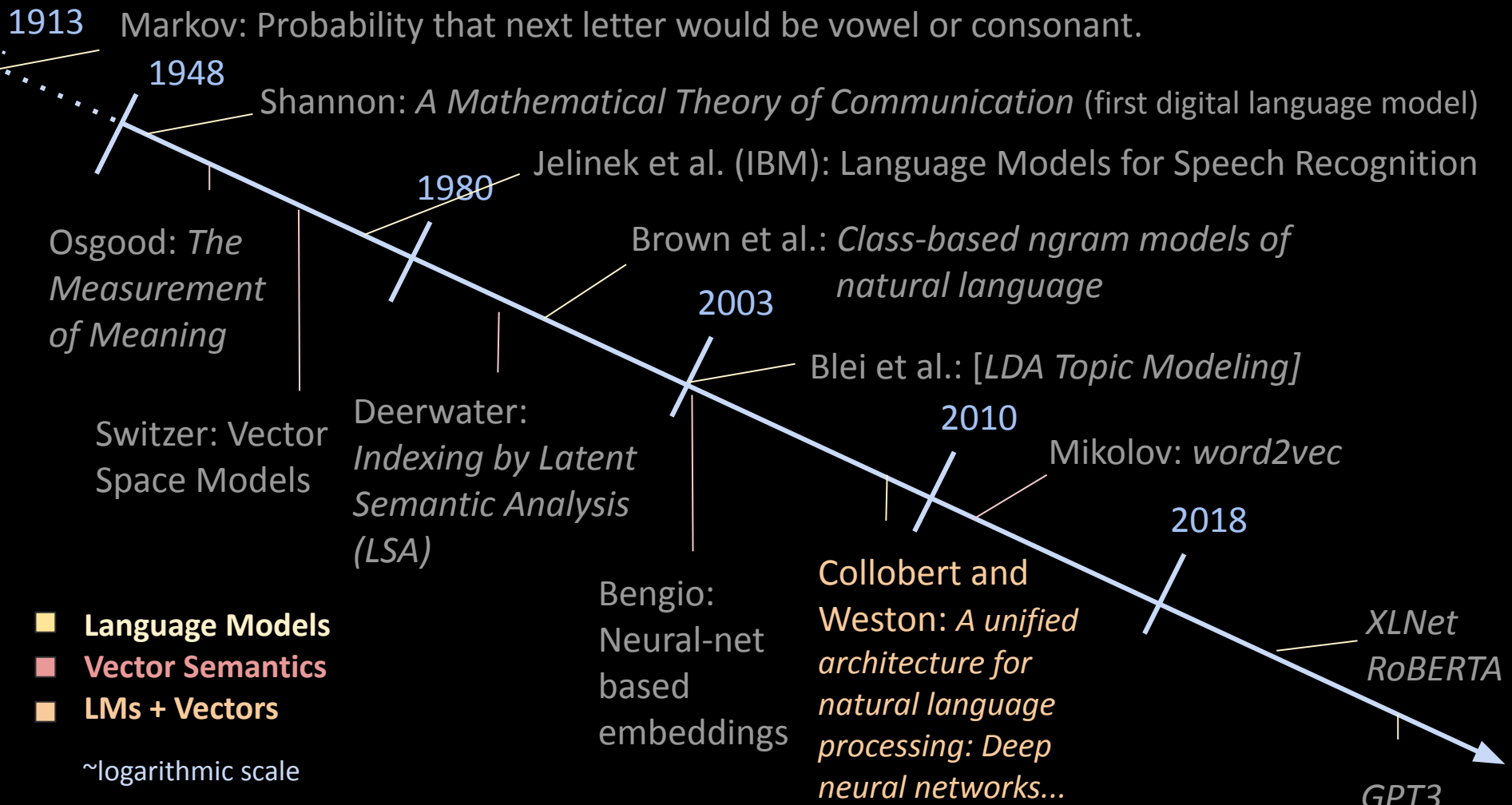
Detailed Overview of (HuggingFace) Transformer Matrices and Computation



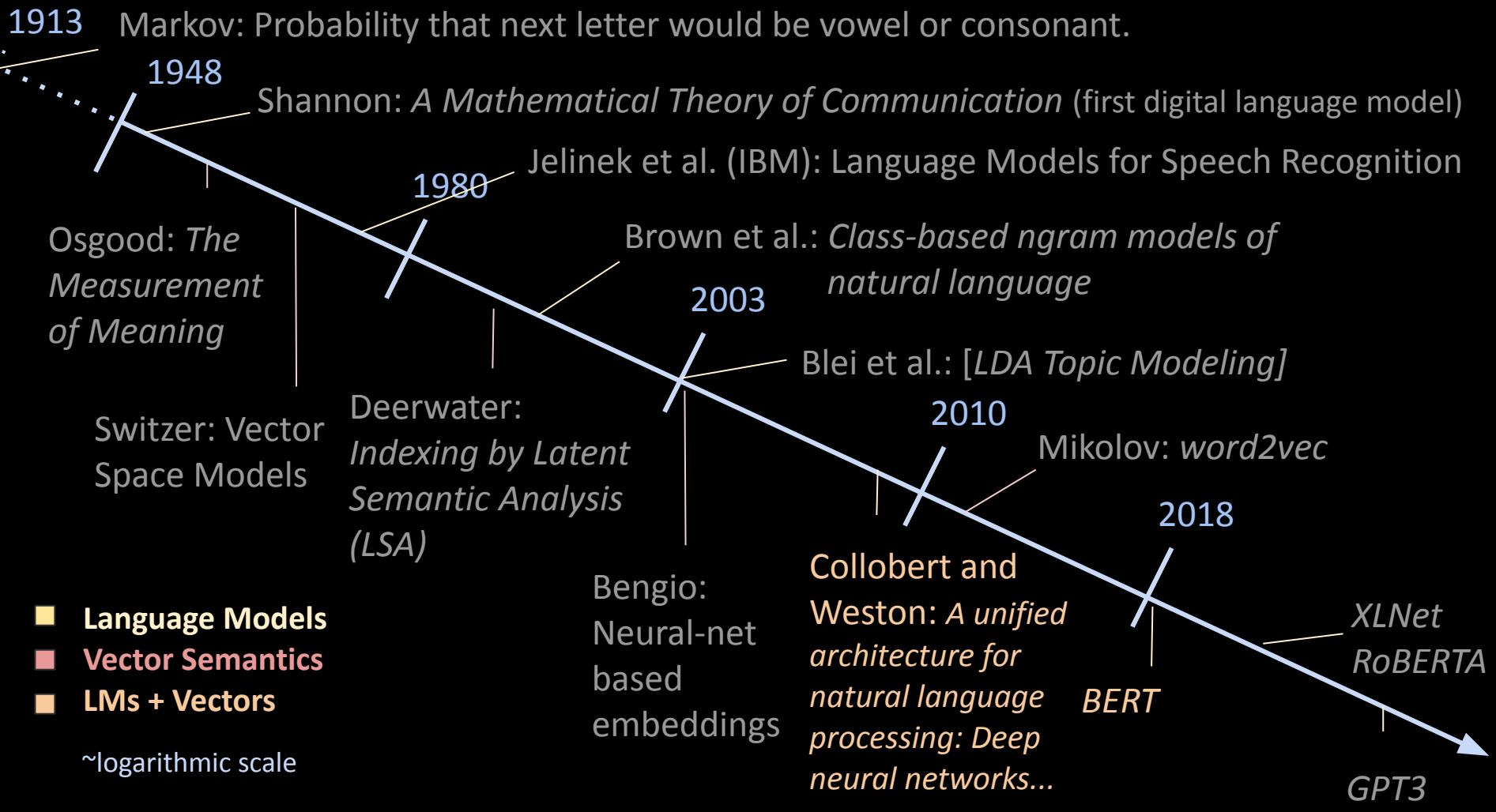
Detailed Overview of (HuggingFace) Transformer Matrices and Computation



Timeline: *Language Modeling* and *Vector Semantics*



Timeline: *Language Modeling* and *Vector Semantics*

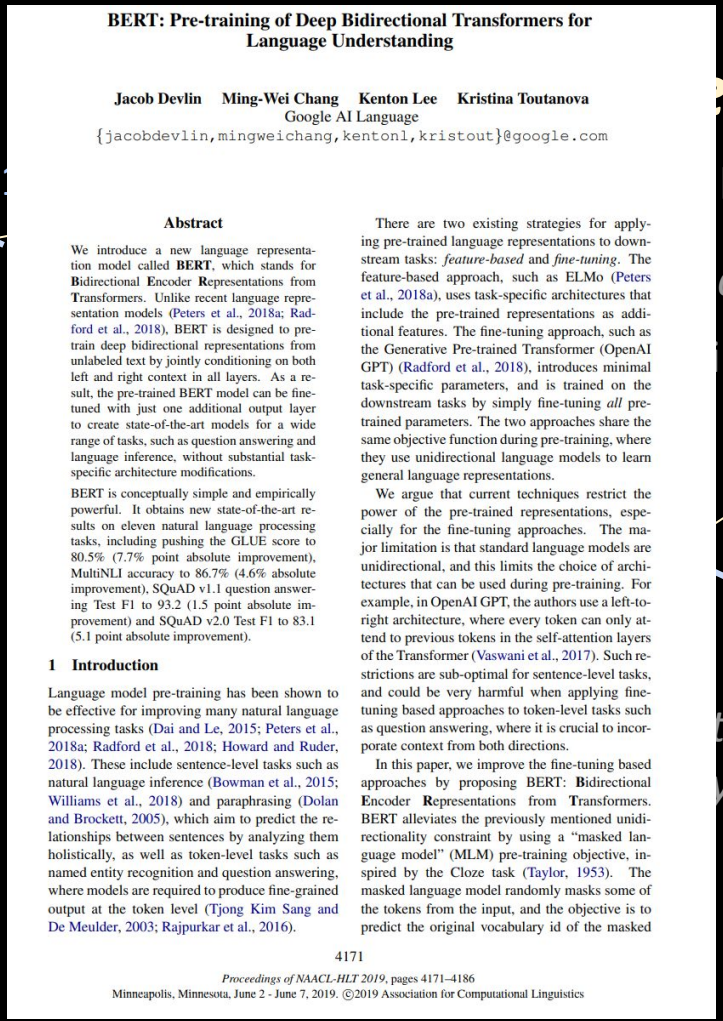


e Modeling and *Vector Semantics*

letter would be vowel or consonant.

atical Theory of Communication (first digital language model)

inek et al. (IBM): Language Models for Speech Recognition



BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin **Ming-Wei Chang** **Kenton Lee** **Kristina Toutanova**
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.com

Abstract

We introduce a new language representation model called **BERT**, which stands for **B**idirectional **E**ncoder **R**epresentations from **T**ransformers. Unlike recent language representation models (Peters et al., 2018a; Radford et al., 2018), BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications.

BERT is conceptually simple and empirically powerful. It obtains new state-of-the-art results on eleven natural language processing tasks, including pushing the GLUE score to 80.5% (7.7% point absolute improvement), MultiNLI accuracy to 86.7% (4.6% absolute improvement), SQuAD v1.1 question answering Test F1 to 93.2 (1.5 point absolute improvement) and SQuAD v2.0 Test F1 to 83.1 (5.1 point absolute improvement).

1 Introduction

Language model pre-training has been shown to be effective for improving many natural language processing tasks (Dai and Le, 2015; Peters et al., 2018a; Radford et al., 2018; Howard and Ruder, 2018). These include sentence-level tasks such as natural language inference (Bowman et al., 2015; Williams et al., 2018) and paraphrasing (Dolan and Brockett, 2005), which aim to predict the relationships between sentences by analyzing them holistically, as well as token-level tasks such as named entity recognition and question answering, where models are required to produce fine-grained output at the token level (Tjong Kim Sang and De Meulder, 2003; Rajpurkar et al., 2016).

There are two existing strategies for applying pre-trained language representations to downstream tasks: *feature-based* and *fine-tuning*. The feature-based approach, such as ELMo (Peters et al., 2018a), uses task-specific architectures that include the pre-trained representations as additional features. The fine-tuning approach, such as the Generative Pre-trained Transformer (OpenAI GPT) (Radford et al., 2018), introduces minimal task-specific parameters, and is trained on the downstream tasks by simply fine-tuning *all* pre-trained parameters. The two approaches share the same objective function during pre-training, where they use unidirectional language models to learn general language representations.

We argue that current techniques restrict the power of the pre-trained representations, especially for the fine-tuning approaches. The major limitation is that standard language models are unidirectional, and this limits the choice of architectures that can be used during pre-training. For example, in OpenAI GPT, the authors use a left-to-right architecture, where every token can only attend to previous tokens in the self-attention layers of the Transformer (Vaswani et al., 2017). Such restrictions are sub-optimal for sentence-level tasks, and could be very harmful when applying fine-tuning based approaches to token-level tasks such as question answering, where it is crucial to incorporate context from both directions.

In this paper, we improve the fine-tuning based approaches by proposing BERT: **B**idirectional **E**ncoder **R**epresentations from **T**ransformers. BERT alleviates the previously mentioned unidirectionality constraint by using a "masked language model" (MLM) pre-training objective, inspired by the Cloze task (Taylor, 1953). The masked language model randomly masks some of the tokens from the input, and the objective is to predict the original vocabulary id of the masked

Proceedings of NAACL-HLT 2019, pages 4171–4186
Minneapolis, Minnesota, June 2 - June 7, 2019.

Brown et al.: *Class-based ngram models of natural language*

2003

Blei et al.: [*LDA Topic Modeling*]

2010

Mikolov: *word2vec*

2018

tent
ysis

Bengio: Neural-net based embeddings

Collobert and Weston: *A unified architecture for natural language processing: Deep neural networks...*

BERT

XLNet
RoBERTA

GPT3

- **LMs + Vectors**

~logarithmic scale

BERT: Pre-training of Deep Bidirectional Transformers for

BERT Rediscovers the Classical NLP Pipeline

Ian Tenney[1] Dipanjan Das[1] Ellie Pavlick[1,2]
[1]Google Research [2]Brown University
{iftenney,dipanjand,epavlick}@google.com

Abstract

Pre-trained text encoders have rapidly advanced the state of the art on many NLP tasks. We focus on one such model, BERT, and aim to quantify where linguistic information is captured within the network. We find that the model represents the steps of the traditional NLP pipeline in an interpretable and localizable way, and that the regions responsible for each step appear in the expected sequence: POS tagging, parsing, NER, semantic roles, then coreference. Qualitative analysis reveals that the model can and often does adjust this pipeline dynamically, revising lower-level decisions on the basis of disambiguating information from higher-level representations.

of the network directly, to assess whether there exist localizable regions associated with distinct types of linguistic decisions. Such work has produced evidence that deep language models can encode a range of syntactic and semantic information (e.g. Shi et al., 2016; Belinkov, 2018; Tenney et al., 2019), and that more complex structures are represented hierarchically in the higher layers of the model (Peters et al., 2018b; Blevins et al., 2018).

We build on this latter line of work, focusing on the BERT model (Devlin et al., 2019), and use a suite of probing tasks (Tenney et al., 2019) derived from the traditional NLP pipeline to quantify where specific types of linguistic information are

lationships between sentences by analyzing them holistically, as well as token-level tasks such as named entity recognition and question answering, where models are required to produce fine-grained output at the token level (Tjong Kim Sang and De Meulder, 2003; Rajpurkar et al., 2016).

rectionality constraint by using a "masked language model" (MLM) pre-training objective, inspired by the Cloze task (Taylor, 1953). The masked language model randomly masks some of the tokens from the input, and the objective is to predict the original vocabulary id of the masked

Proceedings of NAACL-HLT 2019, pages 4171–4186
Minneapolis, Minnesota, June 2 - June 7, 2019.

and *Vector Semantics*

owel or consonant.

ommunication (first digital language model)

anguage Models for Speech Recognition

lass-based ngram models of atural language

i et al.: [*LDA Topic Modeling]*

2010

Mikolov: *word2vec*

2018

Bengio: Neural-net based embeddings

Collobert and Weston: *A unified architecture for natural language processing: Deep neural networks...*

BERT

XLNet
RoBERTA

GPT3

- **LMs + Vectors**

~logarithmic scale

ng and *Vector Semantics*

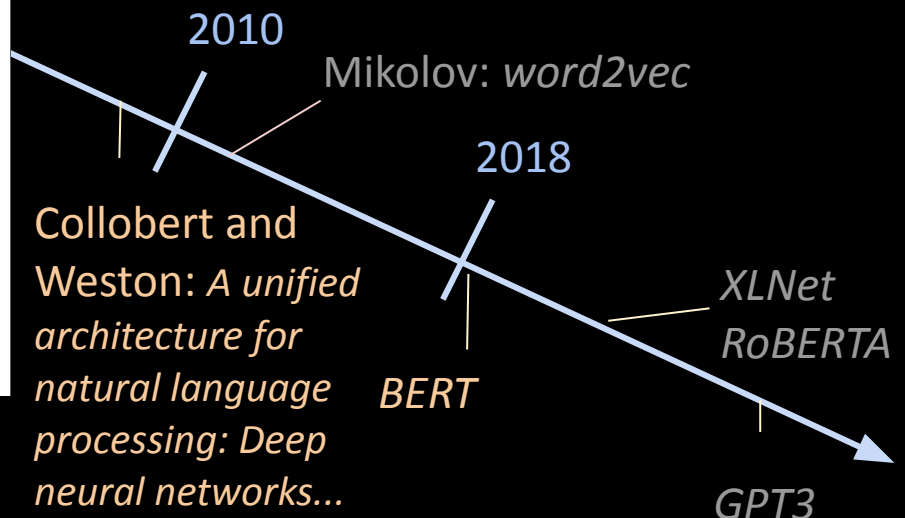
e vowel or consonant.

of Communication (first digital language model)

M): Language Models for Speech Recognition

l.: *Class-based ngram models of natural language*

Blei et al.: [*LDA Topic Modeling*]



BERT: Pre-trair

Jacob Devlin Mi

{jacobdevlin,mi

Abstract

We introduce a new languag tion model called BERT, whi Bidirectional Encoder Represe Transformers. Unlike recent la sentation models (Peters et al. ford et al., 2018), BERT is de train deep bidirectional represe unlabeled text by jointly conditi left and right context in all lay sult, the pre-trained BERT mod tuned with just one additiona to create state-of-the-art mode range of tasks, such as question language inference, without su specific architecture modificatio

BERT is conceptually simple a powerful. It obtains new state sults on eleven natural langua tasks, including pushing the G 80.5% (7.7% point absolute MultiNLI accuracy to 86.7% (improvement), SQuAD v1.1 qu ing Test F1 to 93.2 (1.5 point provement) and SQuAD v2.0 T (5.1 point absolute improvemen

1 Introduction

Language model pre-training ha be effective for improving many processing tasks (Dai and Le, 2(2018a; Radford et al., 2018; Ho 2018). These include sentence-le natural language inference (Bow Williams et al., 2018) and para and Brockett, 2005), which aim lationships between sentences by holistically, as well as token-lev named entity recognition and qu where models are required to pro output at the token level (Tjong De Meulder, 2003; Rajpurkar et a

Pro

Minneapolis, Minnesota, J

Journalism Quarterly

DEVOTED TO RESEARCH STUDIES IN THE FIELD OF MASS COMMUNICATIONS

FALL 1953

"Cloze Procedure": A New Tool For Measuring Readability

BY WILSON L. TAYLOR*

Here is the first comprehensive statement of a research method and its theory which were introduced briefly during a workshop at the 1953 AEJ convention. Included are findings from three pilot studies and two experiments in which "cloze procedure" results are compared with those of two readability formulas.

"CLOZE PROCEDURE" IS A NEW PSYchological tool for measuring the effectiveness of communication. The method is straightforward; the data are easily quantifiable; the findings seem to stand up.

At the outset, this tool was looked on as a new approach to "readability." It was so used in three pilot studies and two experiments, the main findings of which are reported here.

First, the results of the new method were repeatedly shown to conform with the results of the Flesch and Dale-Chall devices for estimating readability. Then the scope broadened, and cloze procedure was pitted against those standard formulas.

If future research substantiates the results so far, this tool seems likely to have a variety of applications, both theoretical and practical, in other fields involving communication functions.

THE "CLOZE UNIT"

At the heart of the procedure is a functional unit of measurement tentatively dubbed a "cloze." It is pronounced like the verb "close" and is derived from "closure." The last term is one gestalt psychology applies to the human tendency to complete a familiar but not-quite-finished pattern—to "see" a broken circle as a whole one, for example, by mentally closing up the gaps.

*The writer is particularly obligated to Prof. Charles E. Osgood, University of Illinois, and Melvin R. Marks, Personnel Research Section, A.G.O., Department of the Army, for instigating and assisting in the series of efforts that yielded the notion of "cloze procedure." Both are experimental psychologists. Among others who have advised, encouraged or otherwise aided are these of the University of Illinois: Prof. Lee J. Cronbach, educational psychologist and statistician; Dean Wilbur Schramm, Division of Communications; Prof. Charles E. Swanson, Institute of Communications Research, and George R. Klare, psychologist, both of whom have authored articles on readability; and several journalism teachers who lent their classes. Kalmer E. Stordahl and Clifford M. Christensen, until recently research associates of the Institute, also contributed.

415

■ LMs + Vectors

embeddings

~logarithmic scale

Masked Language Modeling

Task: Estimate $P(w_i | w_1, \ldots w_{i-1}, w_{i+1} \ldots w_n)$

:P(masked word given context)

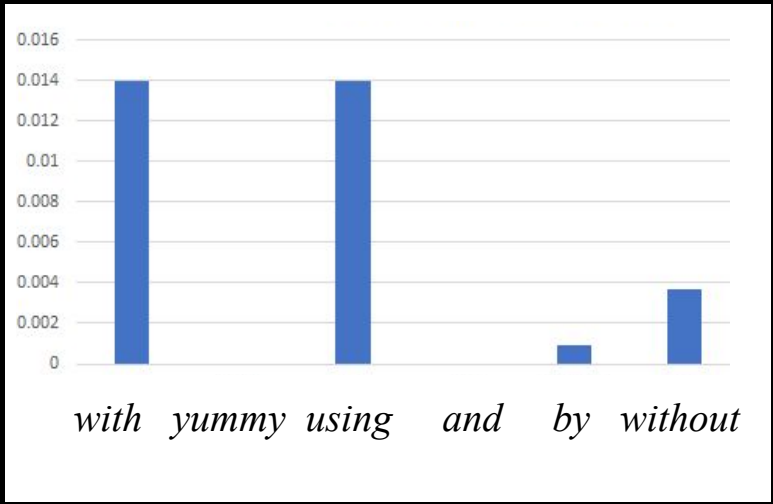
P(with | He ate the cake <M> the fork) = ?

Masked Language Modeling

Task: Estimate $P(w_i | w_1, \ldots w_{i-1}, w_{i+1} \ldots w_n)$

:P(masked word given context)

P(with | He ate the cake <M> the fork) = ?

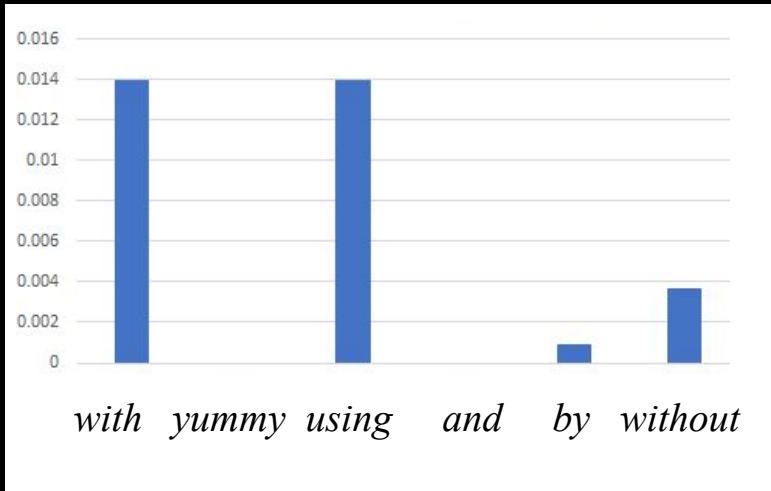


Masked Language Modeling

Task: Estimate $P(w_i | w_1, \ldots w_{i-1}, w_{i+1} \ldots w_n)$

:P(masked word given context)

P(with | He ate the cake <M> the fork) = ?



Masked Language Modeling

Task: Estimate $P(w_i | w_1, \ldots w_{i-1}, w_{i+1} \ldots w_n)$

:P(masked word given context)

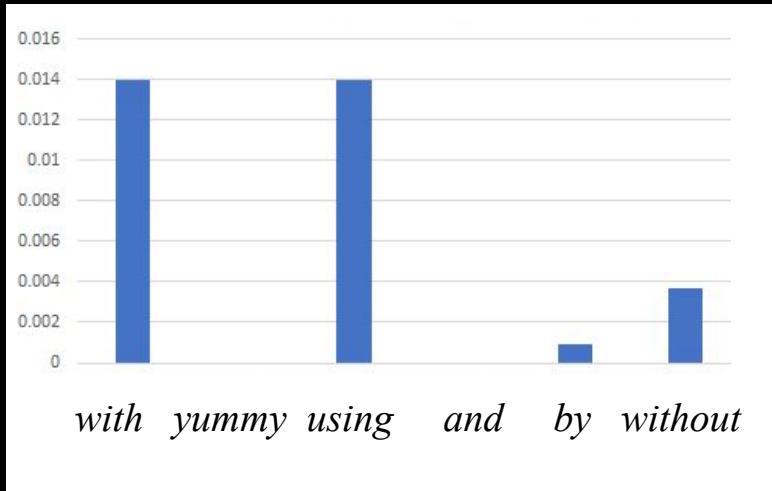
P(with | He ate the cake <M> the fork) = ?

Sequence

(He, at, the, cake,<MASK>, the, fork)

→ Neural Network →

What is the masked word in the sequence?



Transformer Language Models: Uses multiple layers of a **transformer**

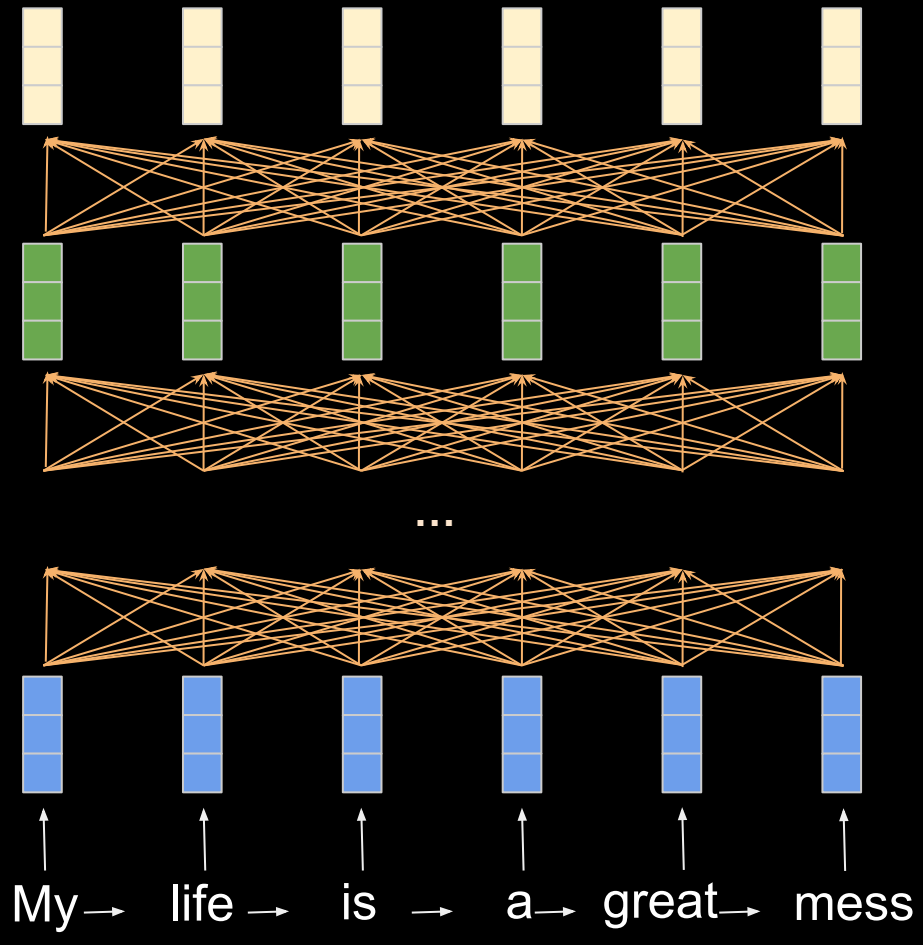
layer k:
(used for language modeling)

layer k-1:
(taken as contextual embedding)

layers 1 to k-2:
(compose embeddings with context)

layer 0:
(input: word-type embeddings)

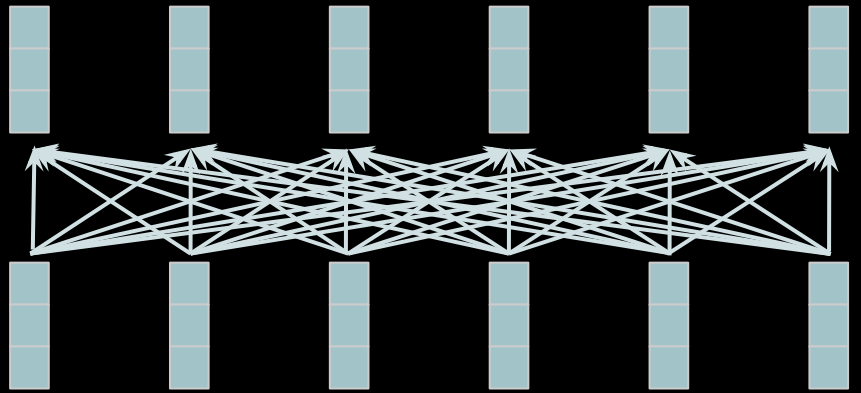
sentence (sequence) input:



(Kjell, Kjell, and Schwartz, 2023)

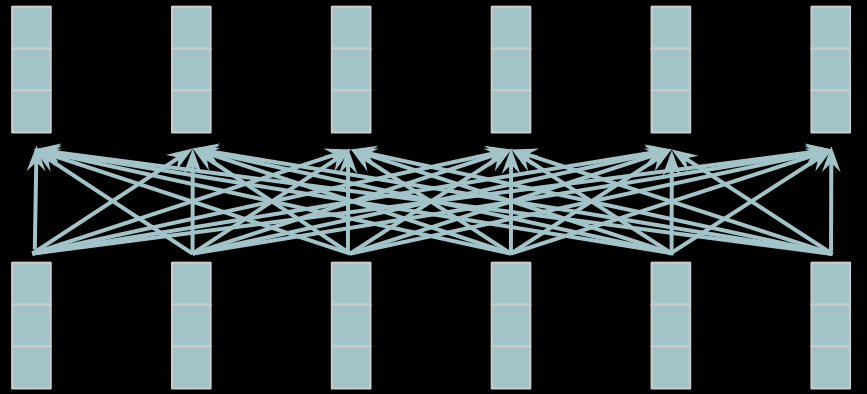
Auto-encoder (MLM):

- Connections go both directions.
- Task is predict word in middle:
  p(wi| …, pwi-2, wi-1, wi+1, wi+2…)
- Better for:
  - embeddings
  - fine-tuning (transfer learning)



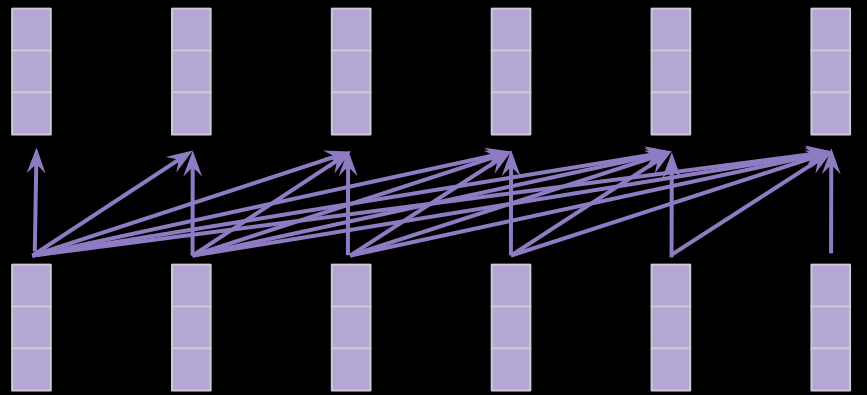
Auto-encoder (MLM):

- Connections go both directions.
- Task is predict word in middle:
  p(wi| …, pwi-2, wi-1, wi+1, wi+2…)
- Better for:
  - embeddings
  - fine-tuning (transfer learning)



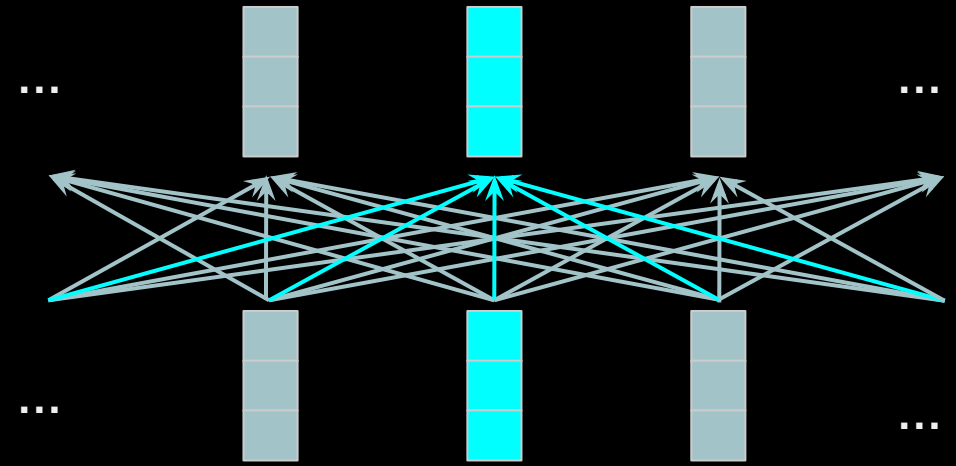
Auto-regressor *(generator):*

- Connections go forward only
- Task is predict word next word:
  p(wi| wi-1, wi-2, …)
- Better for:
  - generating text
  - zero-shot learning



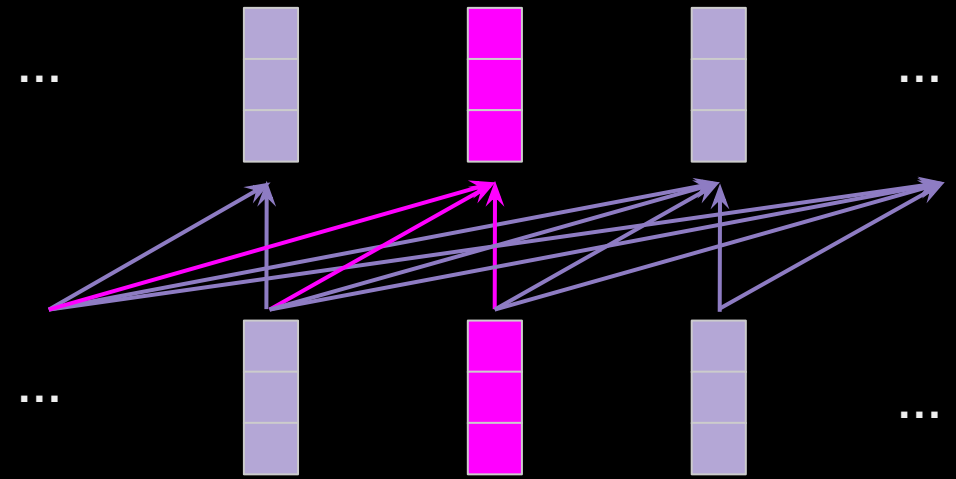
Auto-encoder (MLM):

- Connections go both directions.
- Task is predict word in middle:
  p(wi| …, pwi-2, wi-1, wi+1, wi+2…)
- Better for:
  - embeddings
  - fine-tuning (transfer learning)



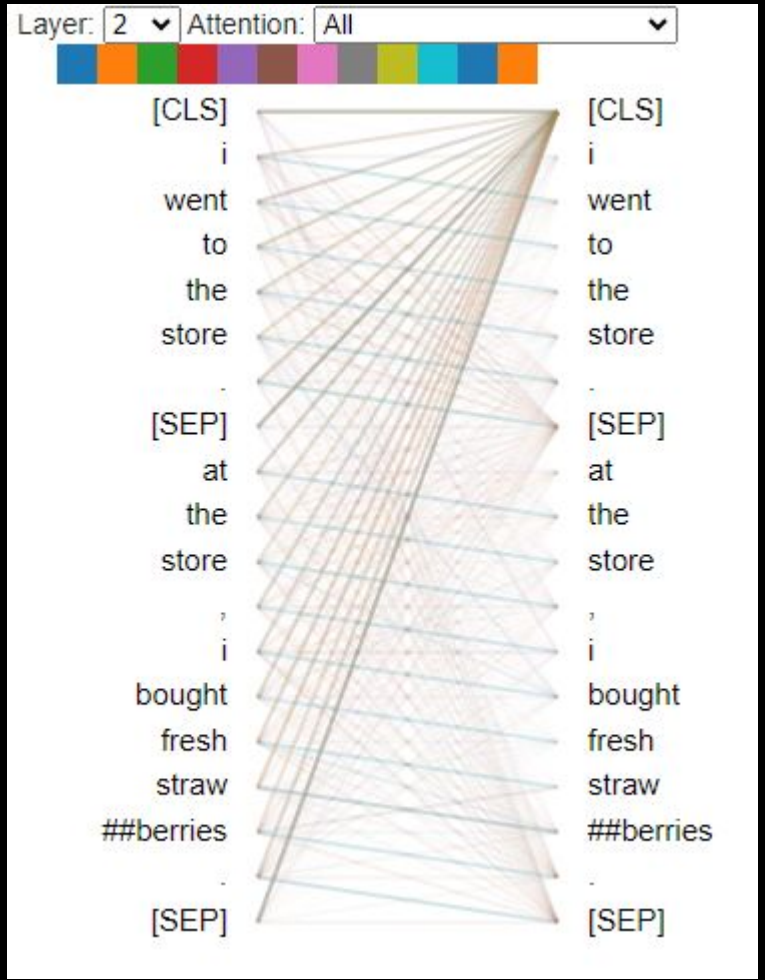
Auto-regressor *(generator):*

- Connections go forward only
- Task is predict word next word:
  p(wi| wi-1, wi-2, …)
- Better for:
  - generating text
  - zero-shot learning



Bert: Attention by Layers

https://colab.research.google.com/drive/1vlOJ1lhdujVjfH857hvYKIdKPTD9Kid8



(Vig, 2019)

Hugging Face or AllenNLP

https://github.com/huggingface/transformers

```
#example for getting embeddings
from transformers import BertModel, PreTrainedTokenizerFast, pipeline

bert_tokenizer = PreTrainedTokenizerFast.from_pretrained('google-bert/bert-base-uncased')
bert_model = BertModel.from_pretrained('google-bert/bert-base-uncased')
pipe = pipeline('feature-extraction', model=bert_model, tokenizer=bert_tokenizer)
emb = pipe(text)
print(emb[0][0])
```

https://docs.allennlp.org/v2.10.1/api/modules/transformer/transformer_module/

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**

Transformers as of 2024

General Language Understanding Evaluations:

https://gluebenchmark.com/leaderboard

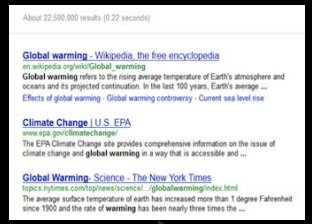
https://super.gluebenchmark.com/leaderboard/

Transformers as of 2023

Machine Translation

absolutamente me gustaría ir de excursión

Web Search

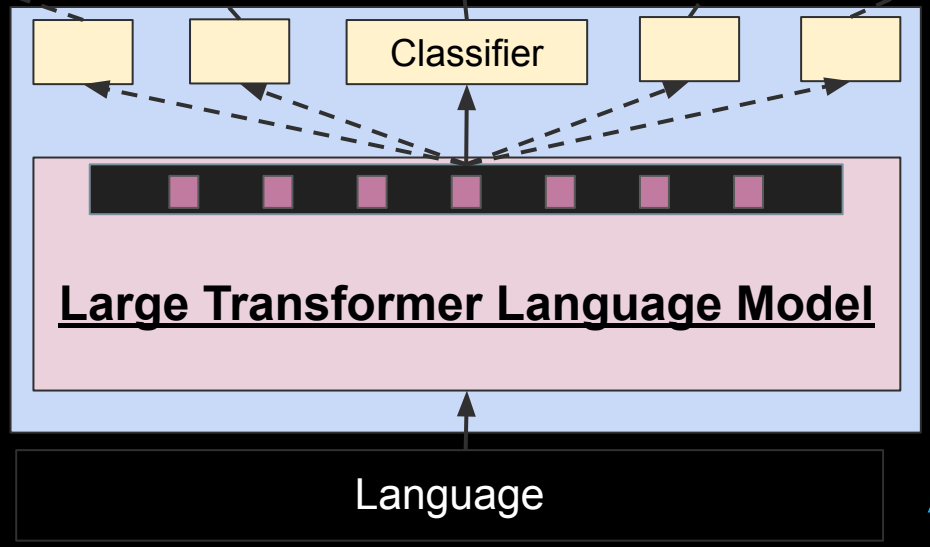


Sentiment Analysis

Document Classification

Assistant, QA

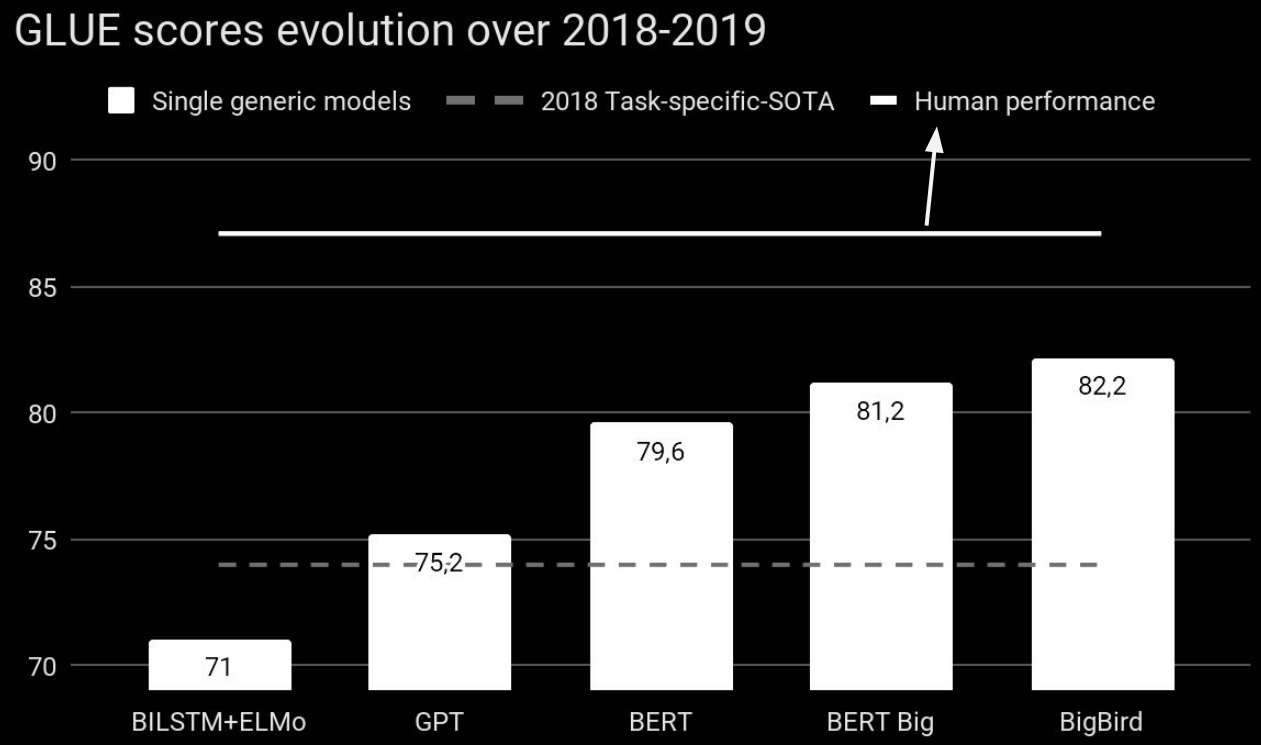
...



(NLP System)

Soni, N., Matero, M., Balasubramanian, N., & Schwartz, H. (2022, May). Human Language Modeling. In *Findings of the Association for Computational Linguistics: ACL 2022* (pp. 622-636).

BERT Performance: e.g. Question Answering



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

The Transformer: Take Away

Challenges to sequential representation learning

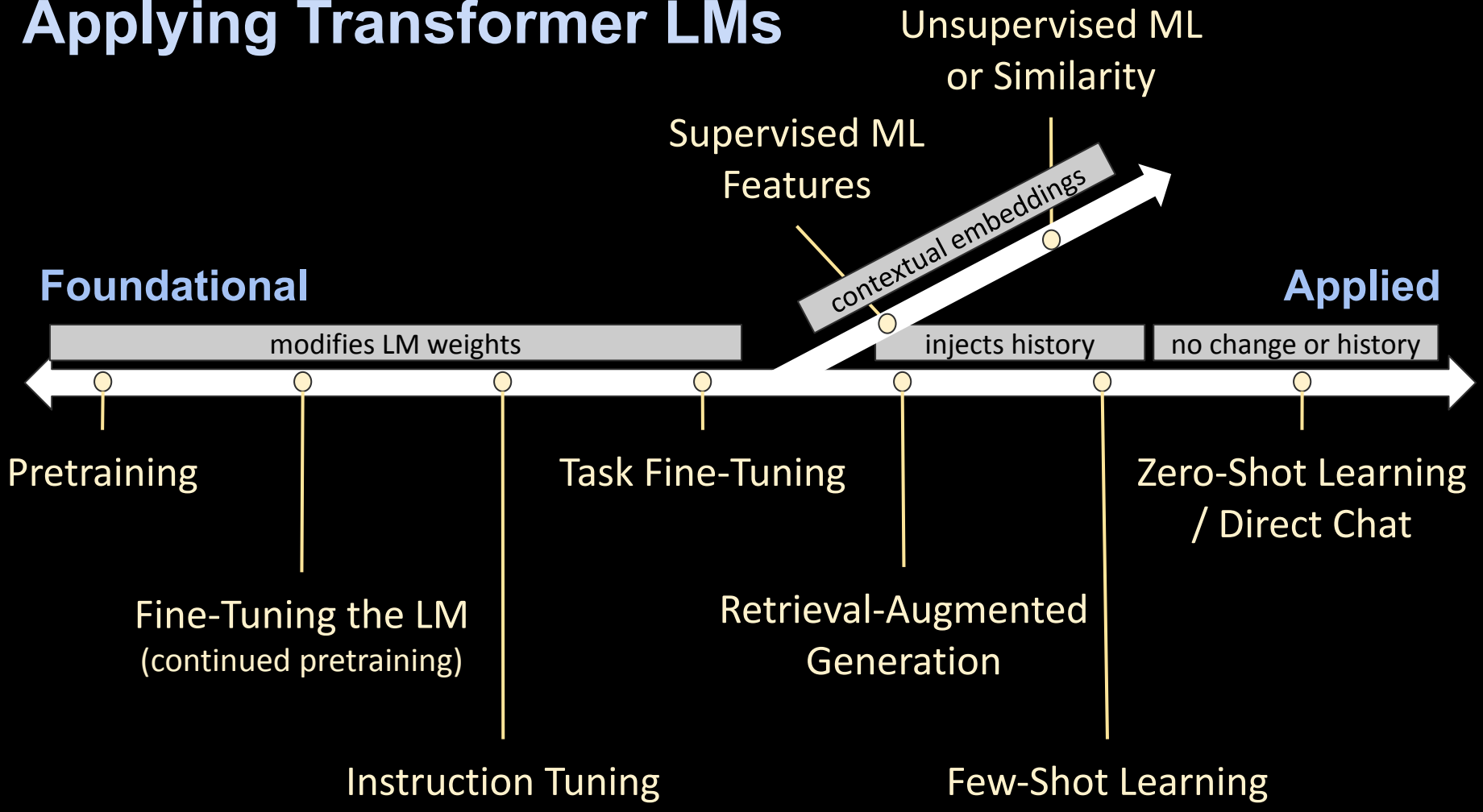
- Capture long-distance dependencies
  Self-attention *treats far away words similar to those close.*
- Preserving sequential distances / periodicity
  Positional embeddings *encode distances/periods.*
- Capture multiple relationships
  Multi-headed attention *enables multiple compositions.*
- Easy to parallelize -- don’t need sequential processing.
  Entire layer can be computed at once. Is only matrix multiplications + standardizing.

Part 3: Applying Transformer LMs

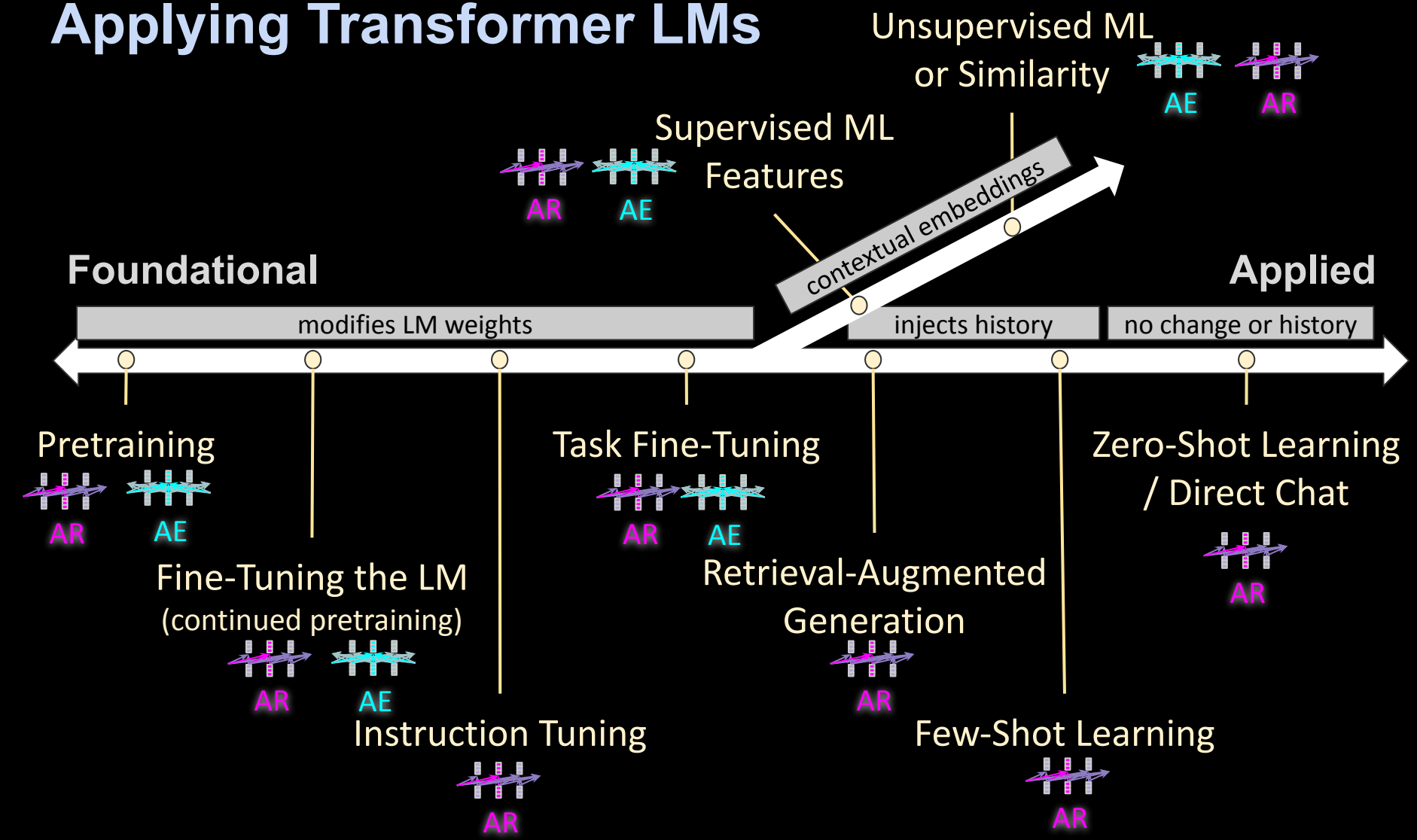
Foundational

Applied

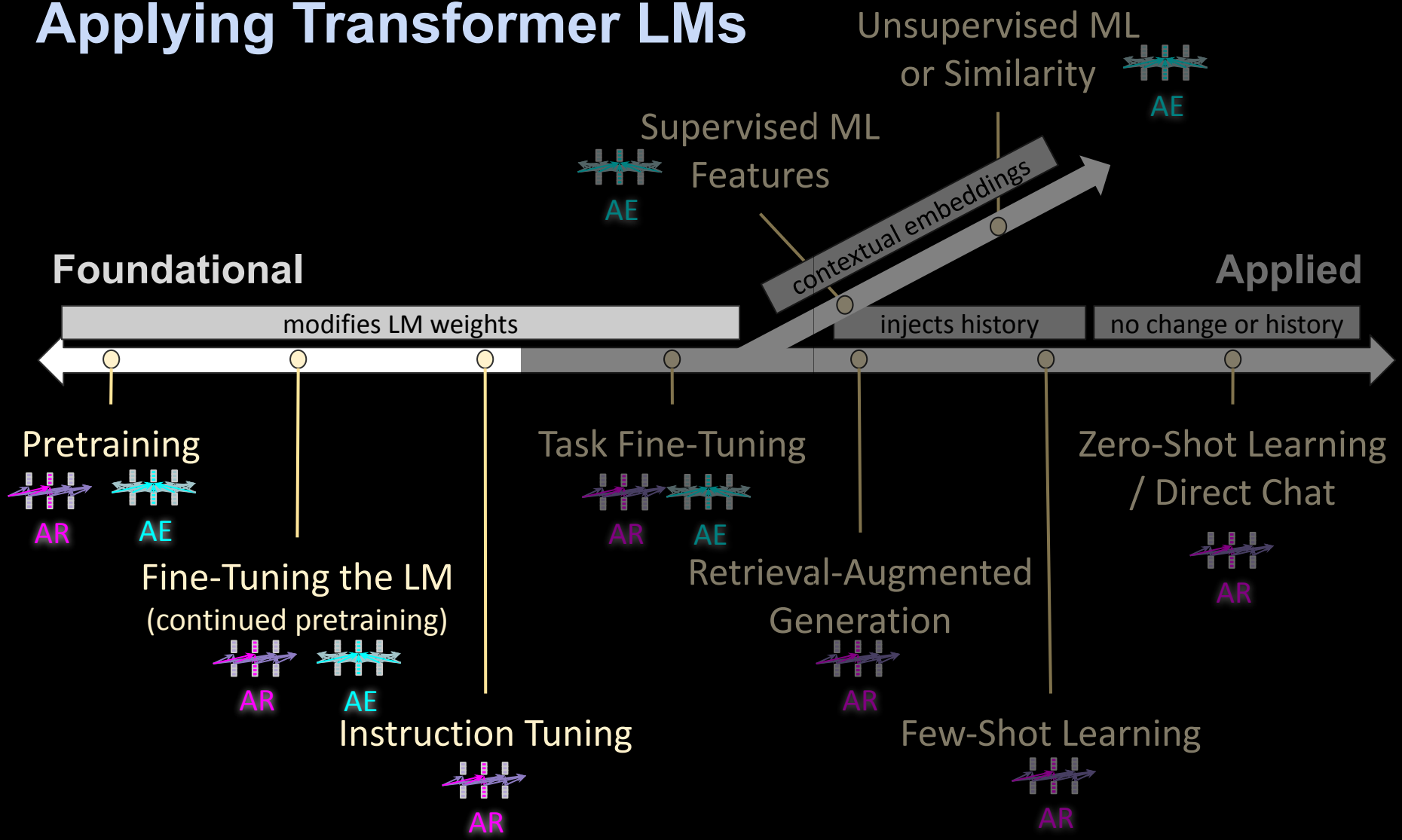
Applying Transformer LMs



Applying Transformer LMs



Applying Transformer LMs



Pretraining; FTing the LM; Instruction Tuning

softmax for LM:

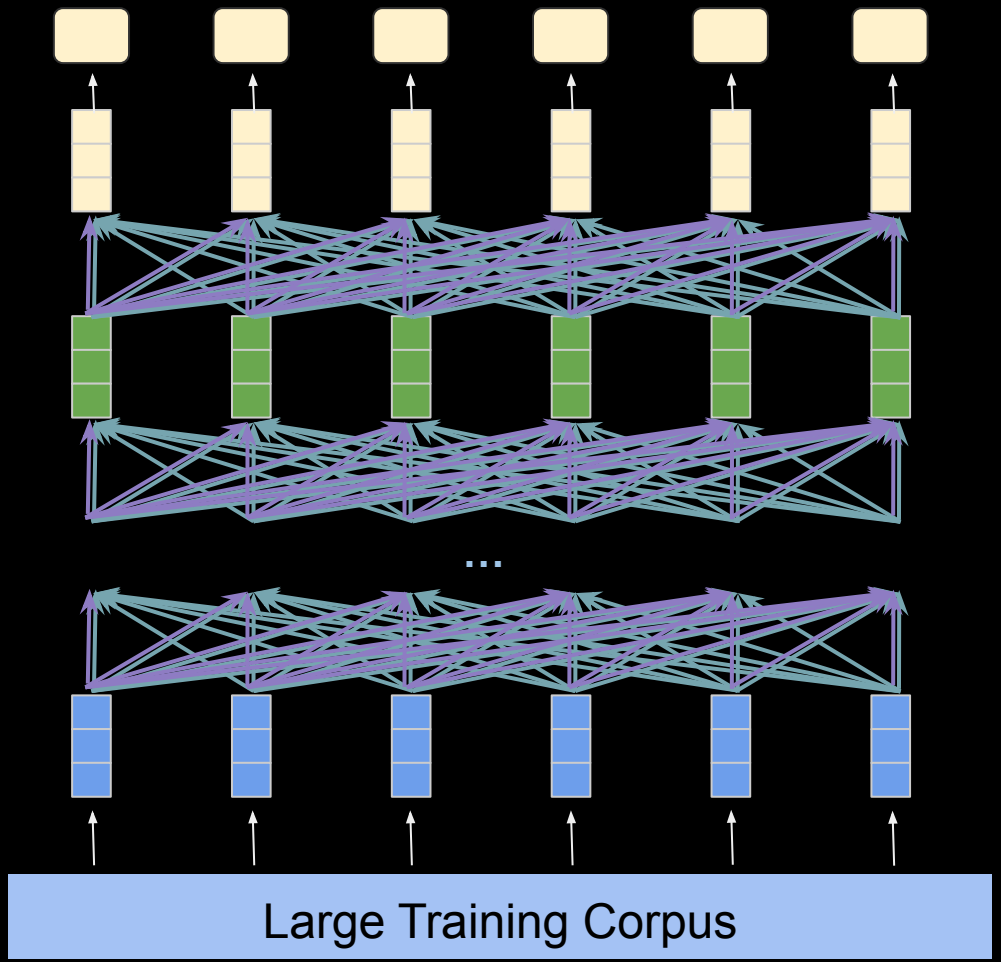
layer k:
(used for language modeling)

layer k-1:
(taken as contextual embedding)

layers 1 to k-2:
(compose embeddings with context)

layer 0:
(input: word-type embeddings)

sentence (sequence) input:



(Kjell, Kjell, and Schwartz, 2023)

Pretraining; **FTing the LM**; Instruction Tuning

softmax for LM:

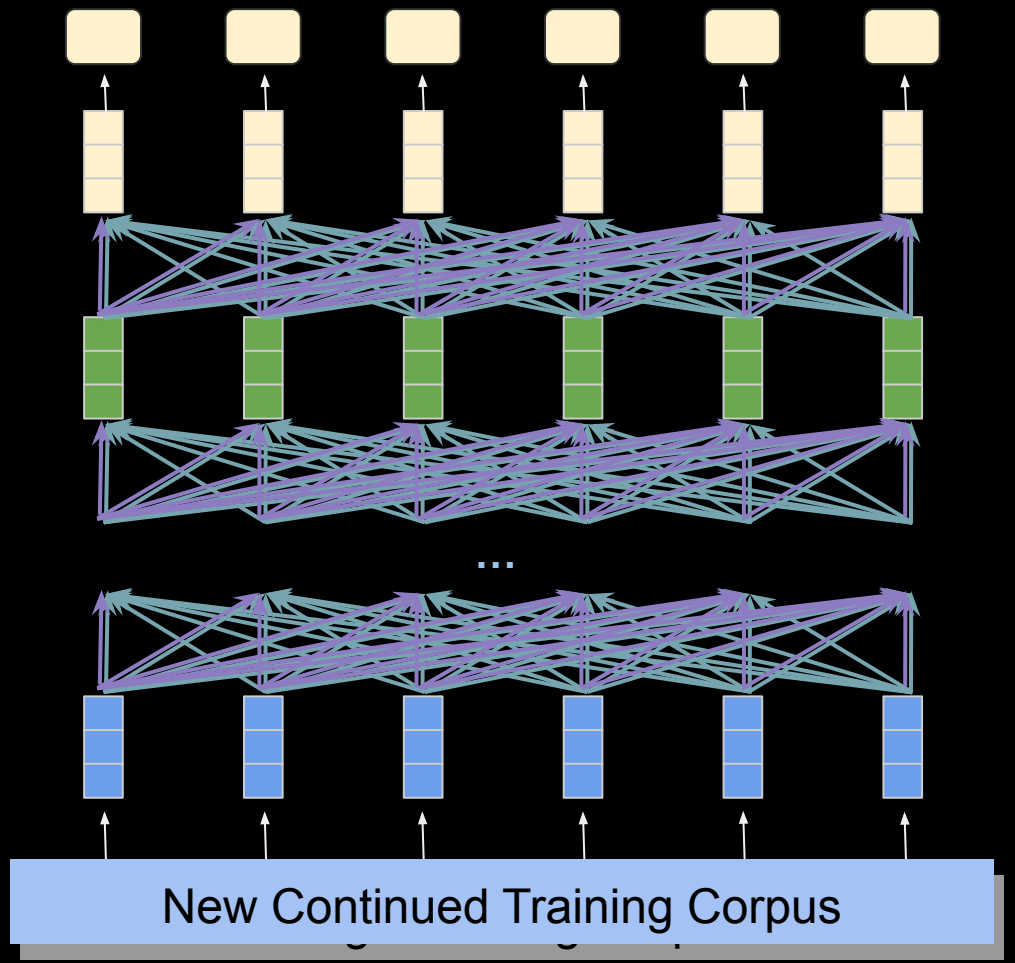
layer k:
(used for language modeling)

layer k-1:
(taken as contextual embedding)

layers 1 to k-2:
(compose embeddings with context)

layer 0:
(input: word-type embeddings)

sentence (sequence) input:



Pretraining; FTing the LM; **Instruction Tuning**

softmax for LM:

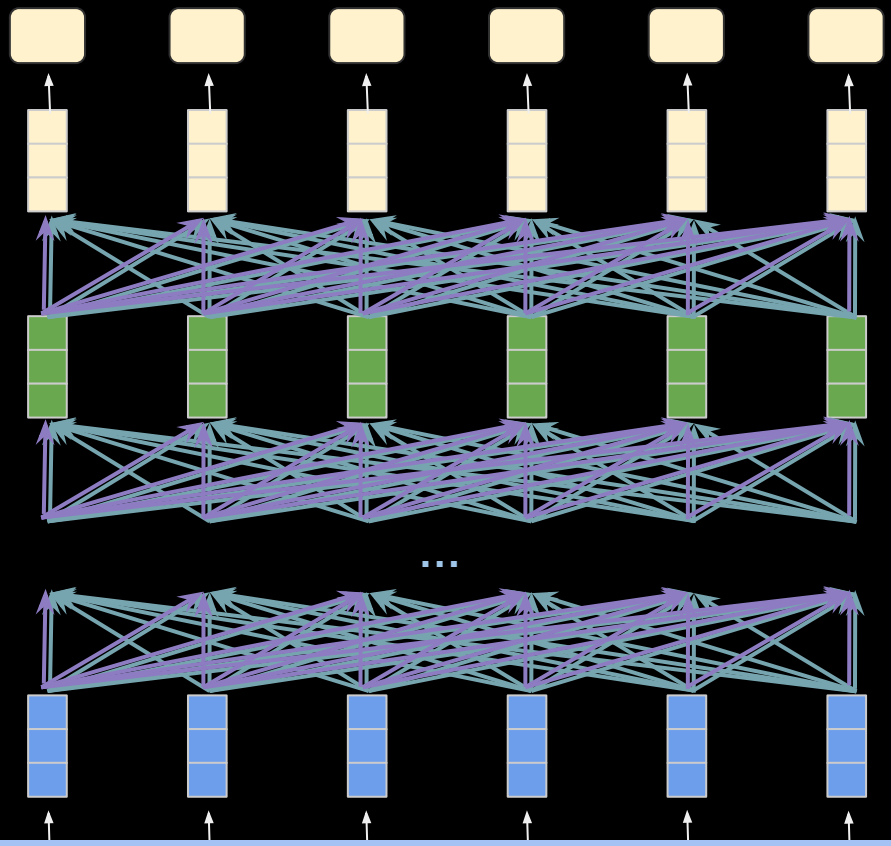
layer k:
(used for language modeling)

layer k-1:
(taken as contextual embedding)

layers 1 to k-2:
(compose embeddings with context)

layer 0:
(input: word-type embeddings)

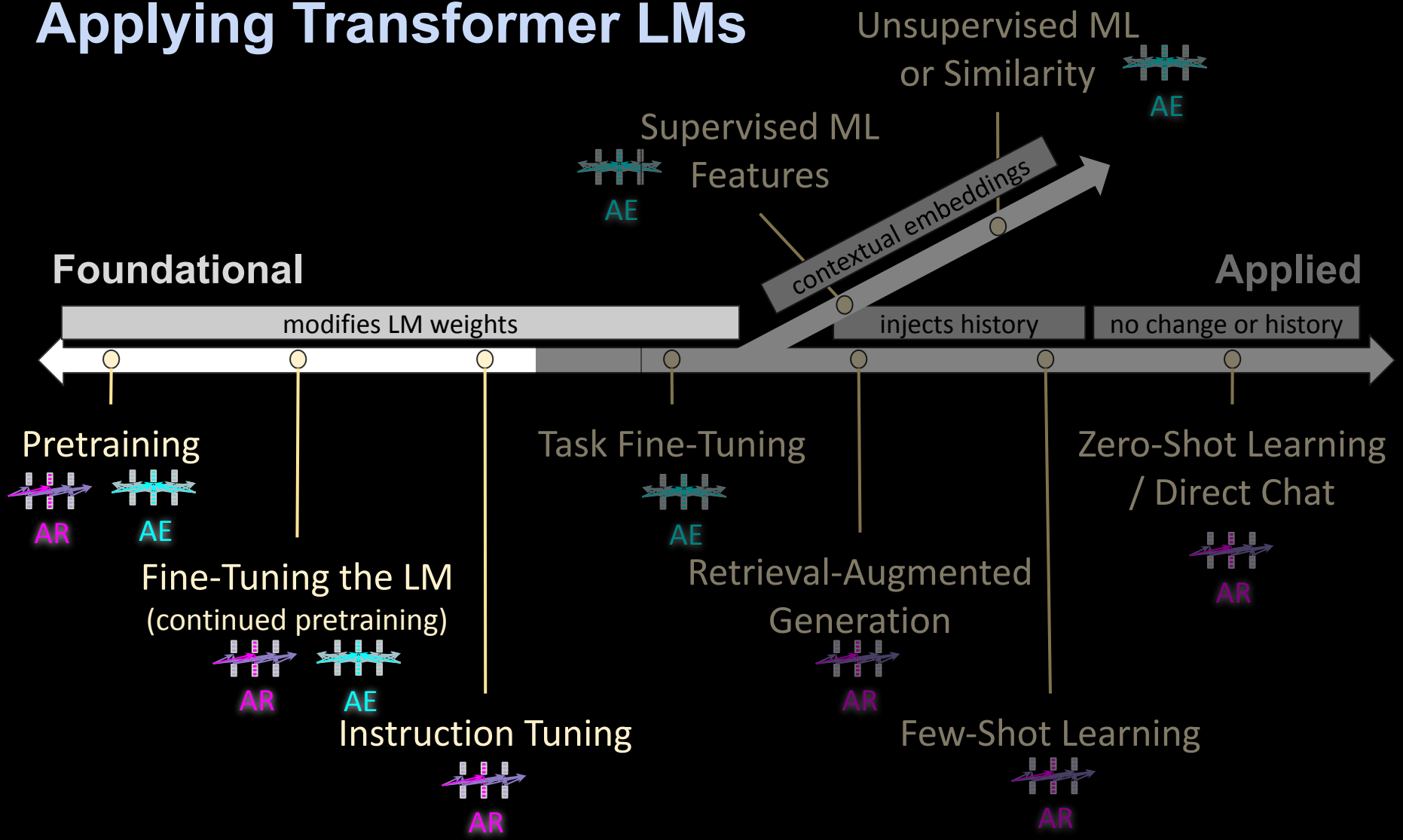
sentence (sequence) input:



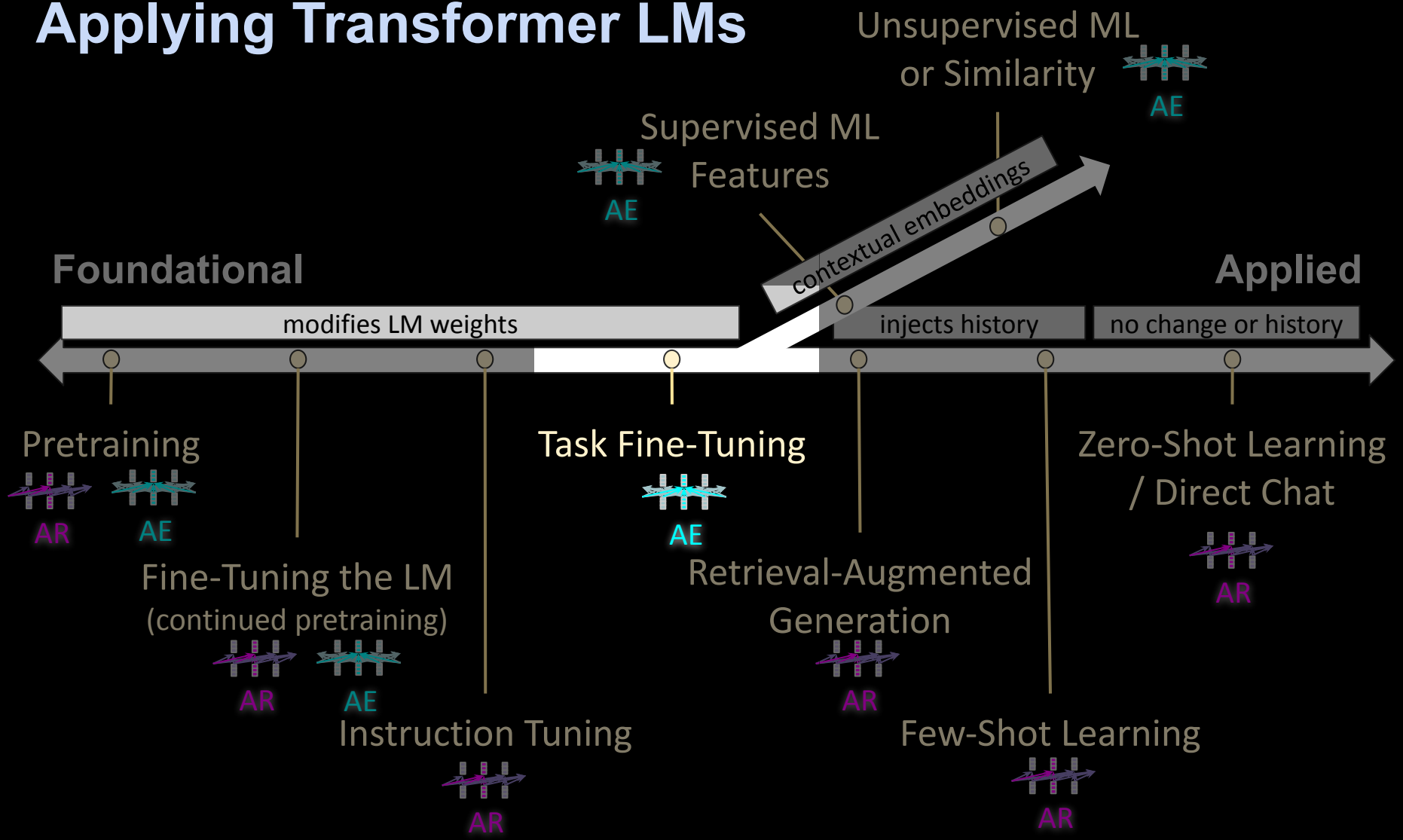
Task Prompts

e.g. *What topic is this about? "Last night, the Seawolves won the game." answer: sports*

Applying Transformer LMs



Applying Transformer LMs



Task Fine-Tuning

~~softmax for LM:~~

~~layer k~~

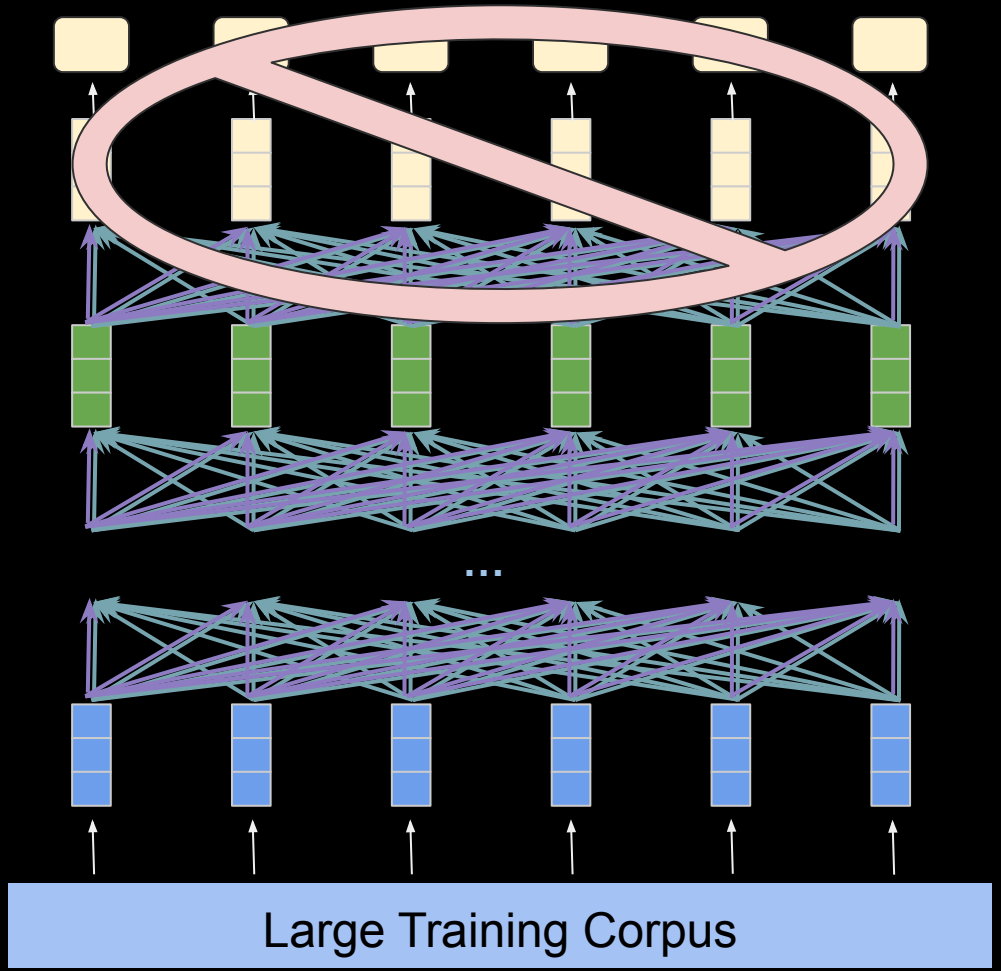
~~(used for language modeling)~~

layer k-1:
(taken as contextual embedding)

layers 1 to k-2:
(compose embeddings with context)

layer 0:
(input: word-type embeddings)

sentence (sequence) input:



(Kjell, Kjell, and Schwartz, 2023)

Task Fine-Tuning

classifier or regressor:
(e.g. sentiment, topic classification, etc.)

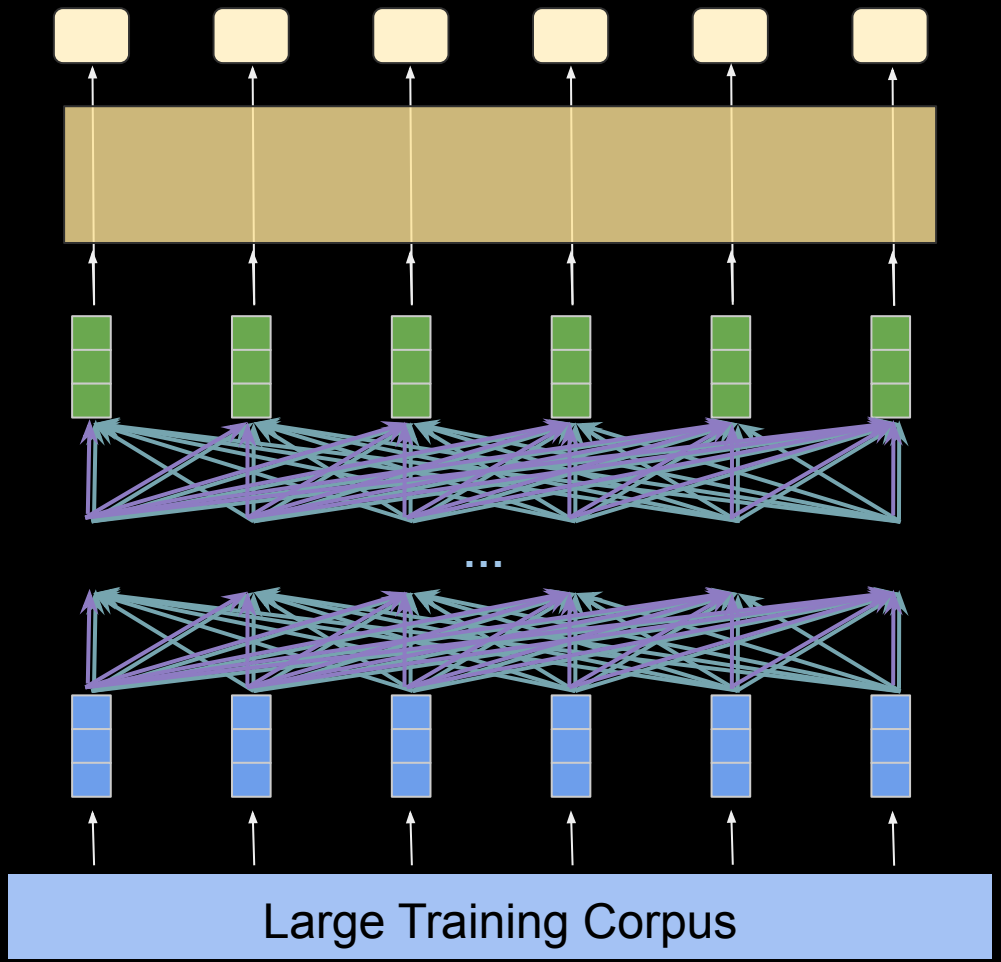
optional layer(s) for task:

layer k-1:
(taken as contextual embedding)

layers 1 to k-2:
(compose embeddings with context)

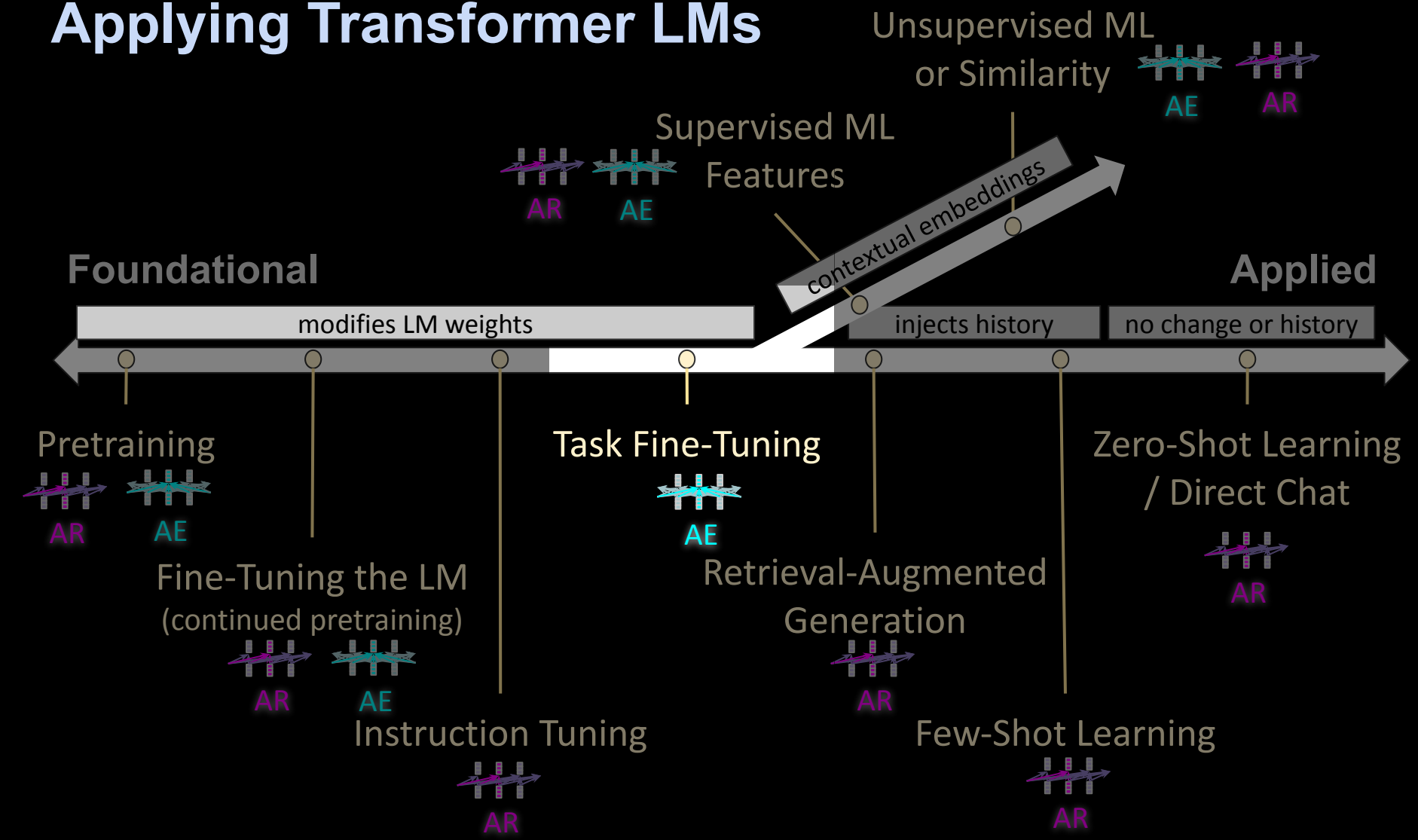
layer 0:
(input: word-type embeddings)

sentence (sequence) input:

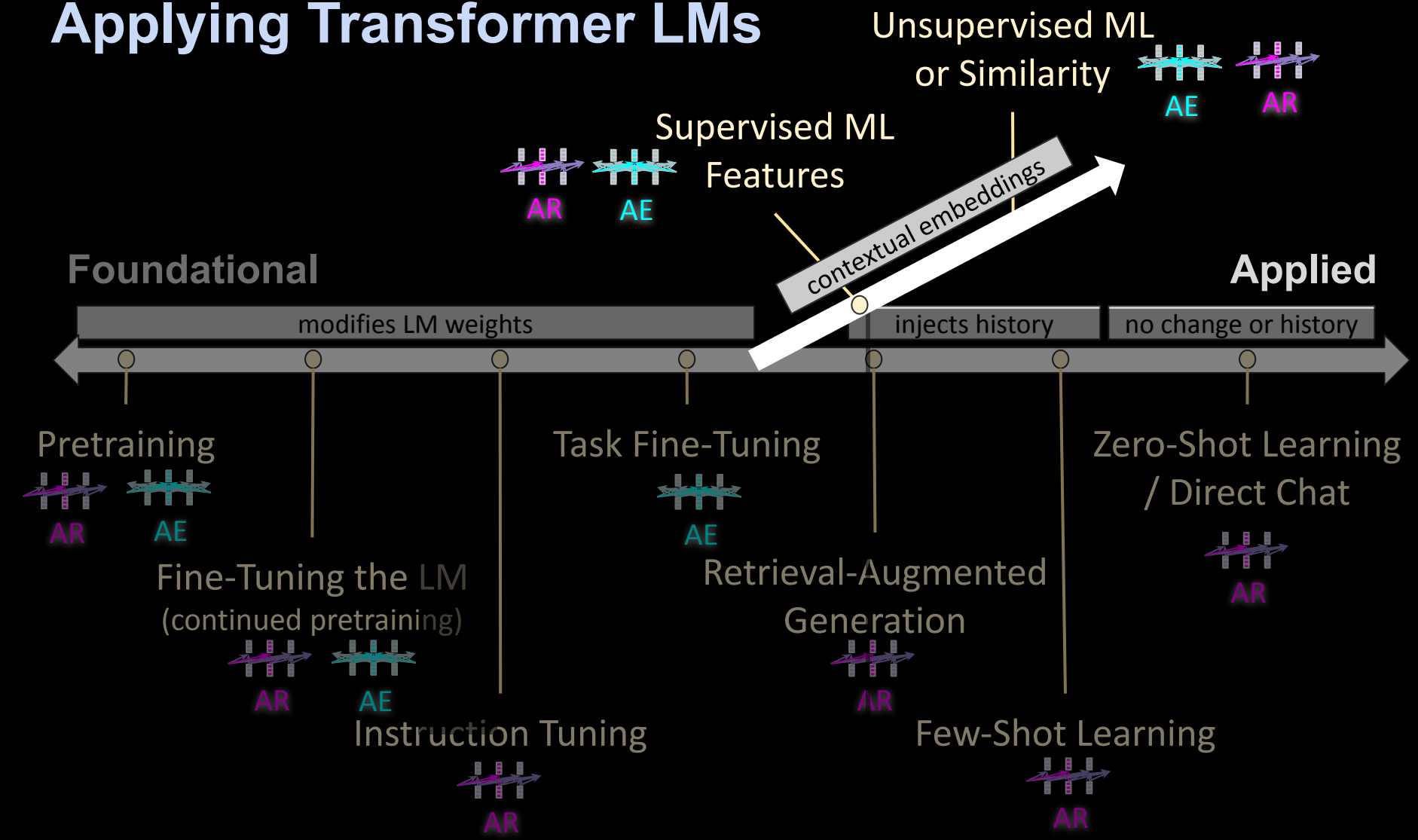


(Kjell, Kjell, and Schwartz, 2023)

Applying Transformer LMs



Applying Transformer LMs



Contextual Embeddings: for Supervised ML; for Similarity (unsup)

softmax for LM:

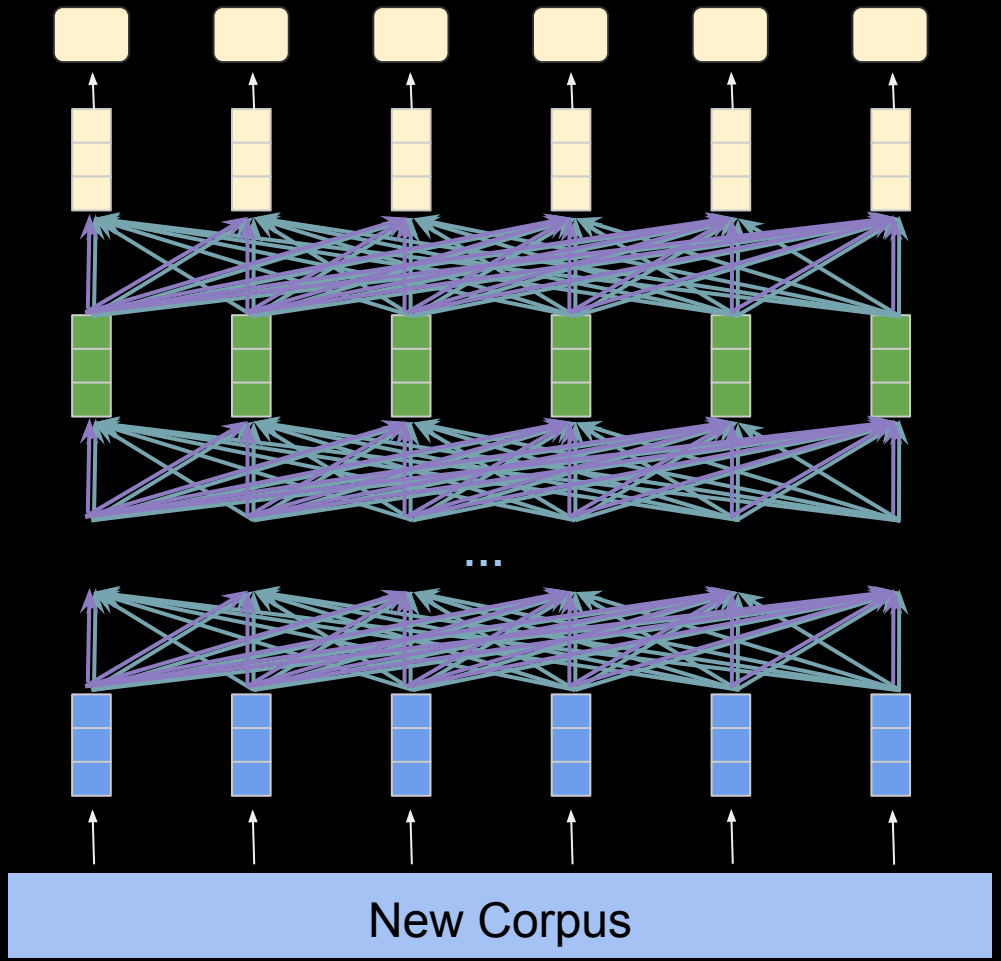
layer k:
(used for language modeling)

layer k-1:
(taken as contextual embedding)

layers 1 to k-2:
(compose embeddings with context)

layer 0:
(input: word-type embeddings)

sentence (sequence) input:



(Kjell, Kjell, and Schwartz, 2023)

Contextual Embeddings: for Supervised ML; for Similarity (unsup)

~~softmax for LM:~~

~~layer k~~

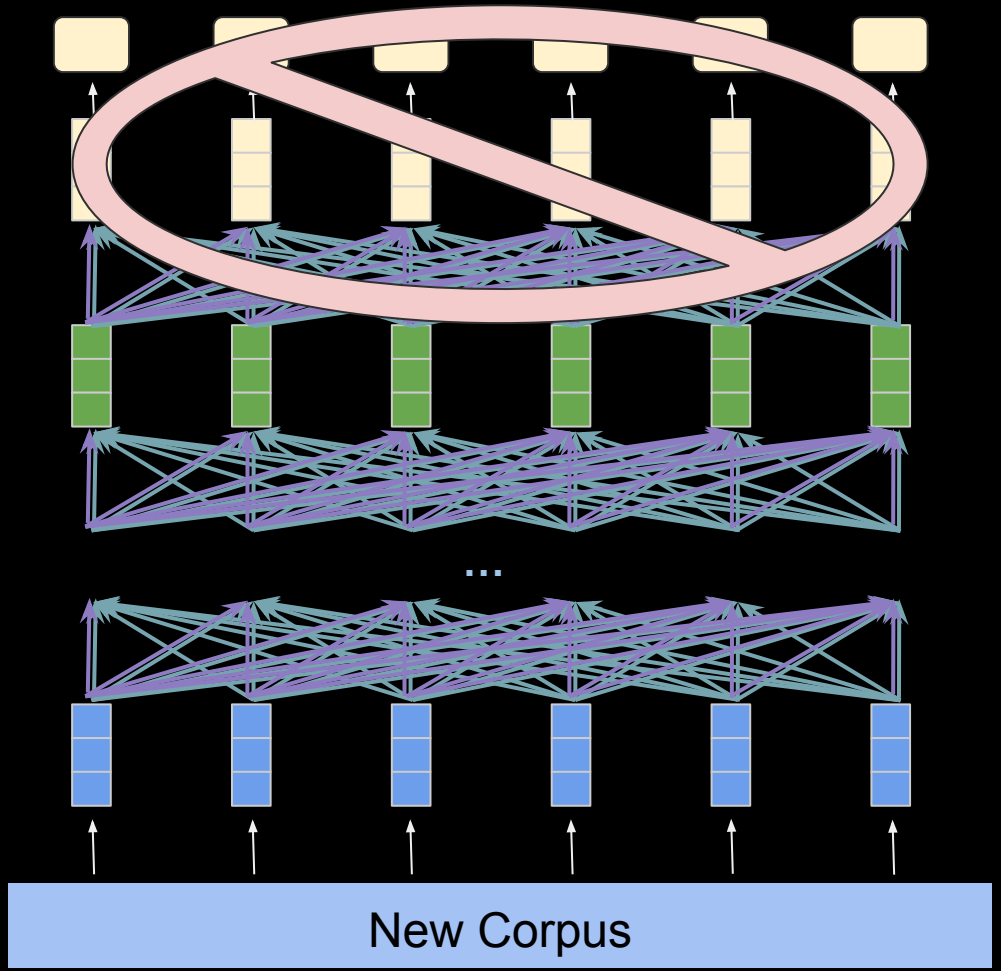
(used for language modeling)

layer k-1:
(taken as contextual embedding)

layers 1 to k-2:
(compose embeddings with context)

layer 0:
(input: word-type embeddings)

sentence (sequence) input:



(Kjell, Kjell, and Schwartz, 2023)

Contextual Embeddings: for Supervised ML

classifier or regressor:
(e.g. sentiment, topic classification, etc.)

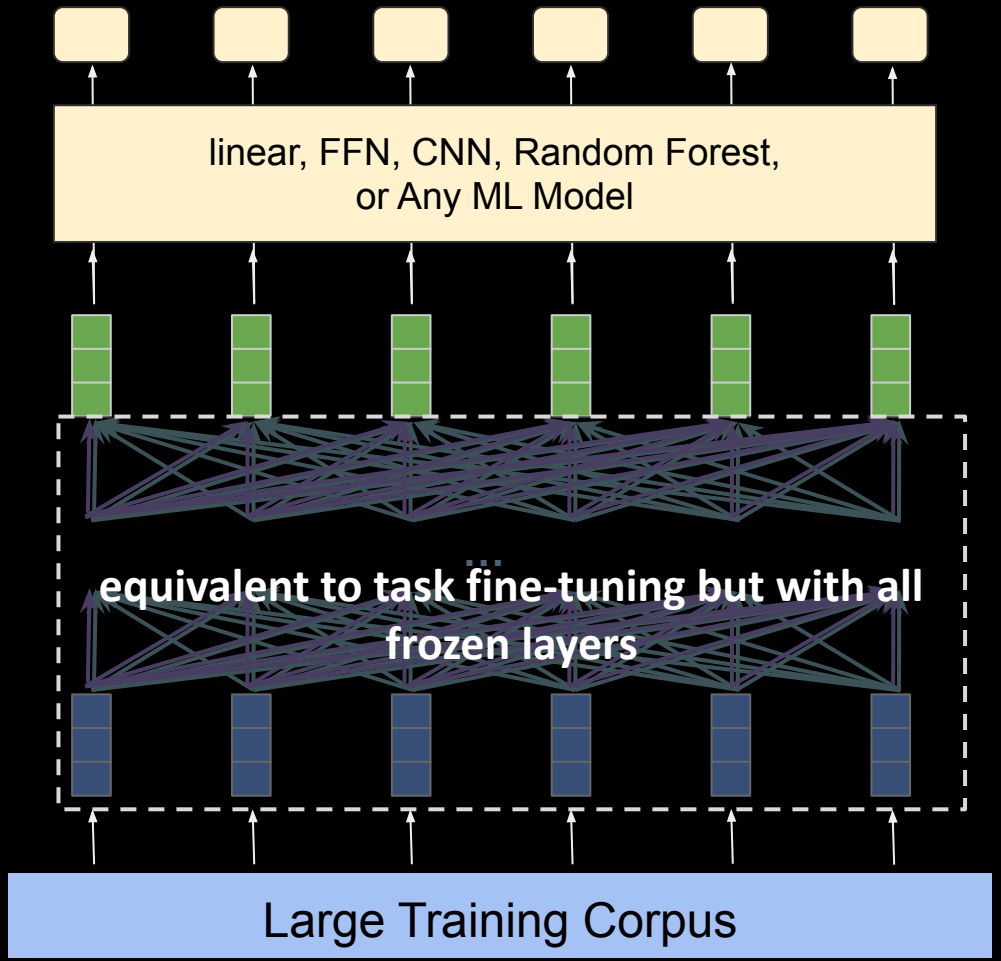
layer(s) for task:

layer k-1:
(taken as contextual embedding)

layers 1 to k-2:
(compose embeddings with context)

layer 0:
(input: word-type embeddings)

sentence (sequence) input:



(Kjell, Kjell, and Schwartz, 2023)

Contextual Embeddings: for Similarity (unsup)

classifier or regressor:
(e.g. sentiment, topic classification, etc.)

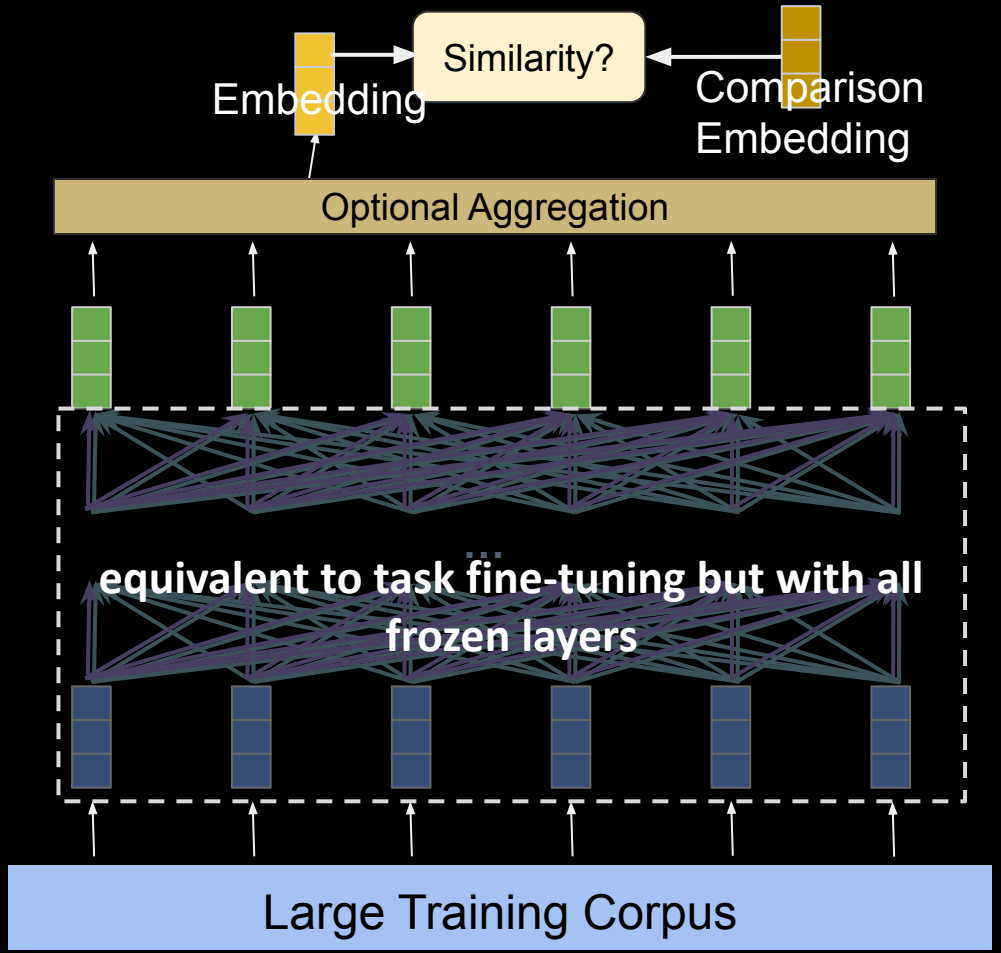
layer(s) for task:

layer k-1:
(taken as contextual embedding)

layers 1 to k-2:
(compose embeddings with context)

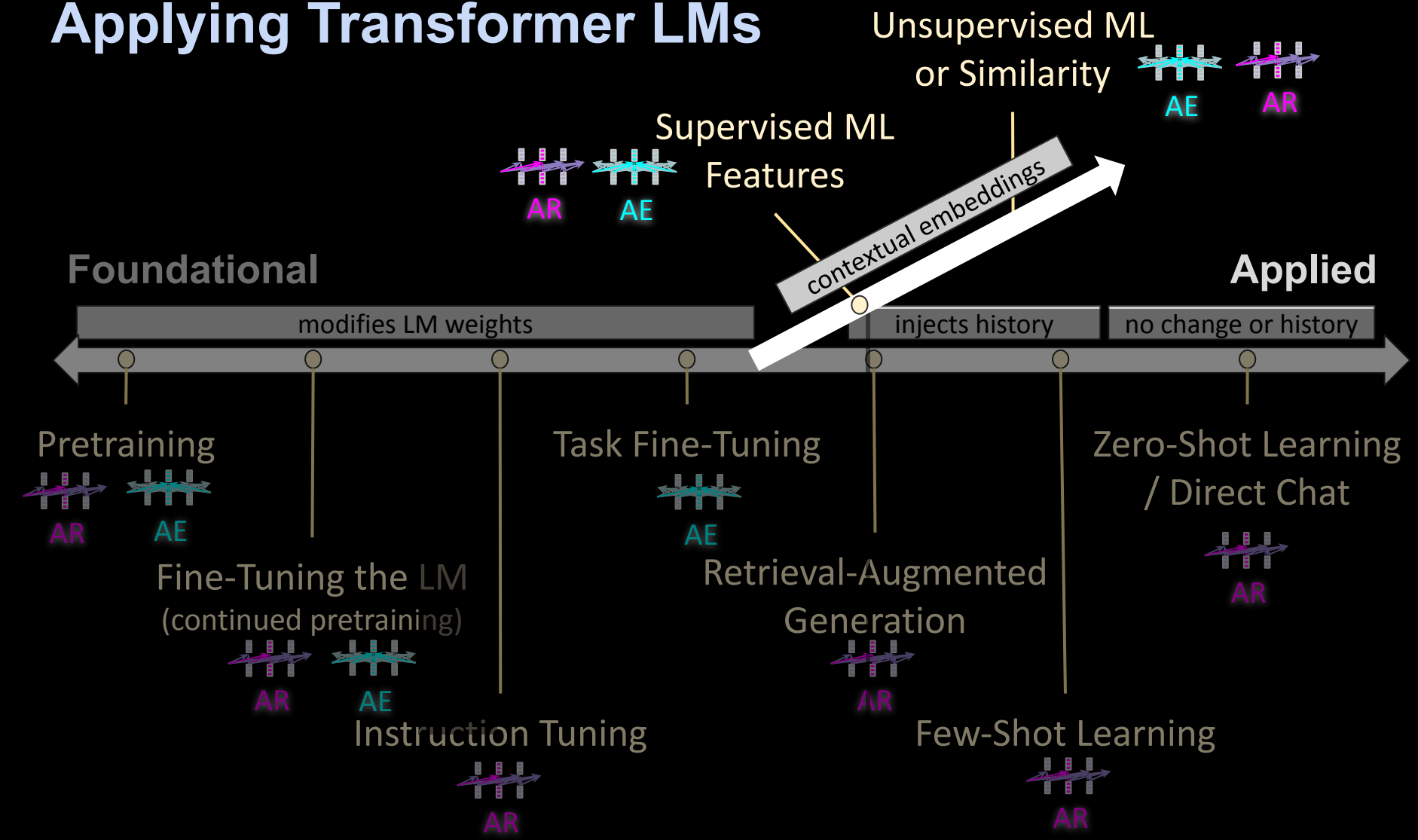
layer 0:
(input: word-type embeddings)

sentence (sequence) input:

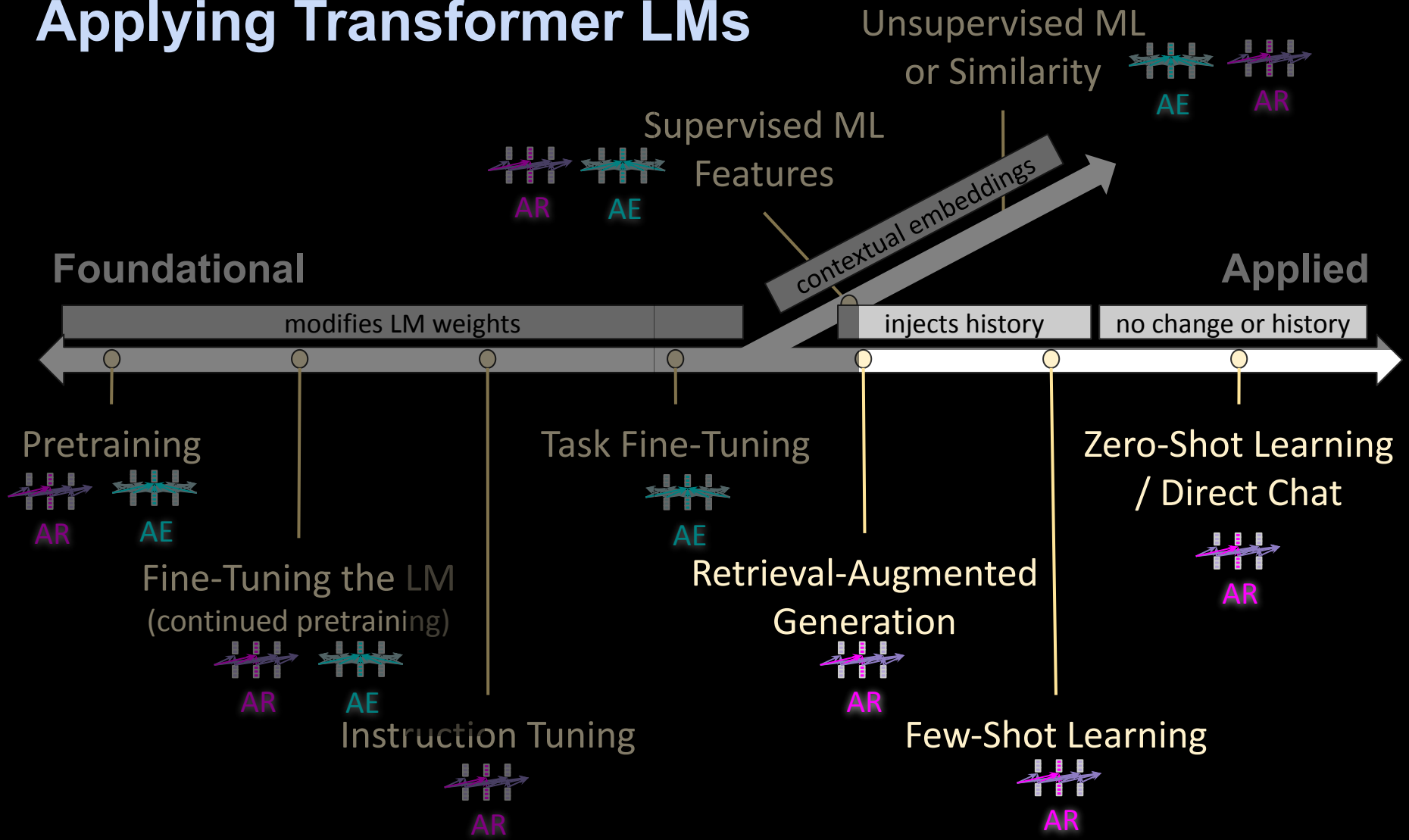


(Kjell, Kjell, and Schwartz, 2023)

Applying Transformer LMs



Applying Transformer LMs



RAG, Few-Shot, Zero-Shot

softmax for LM:

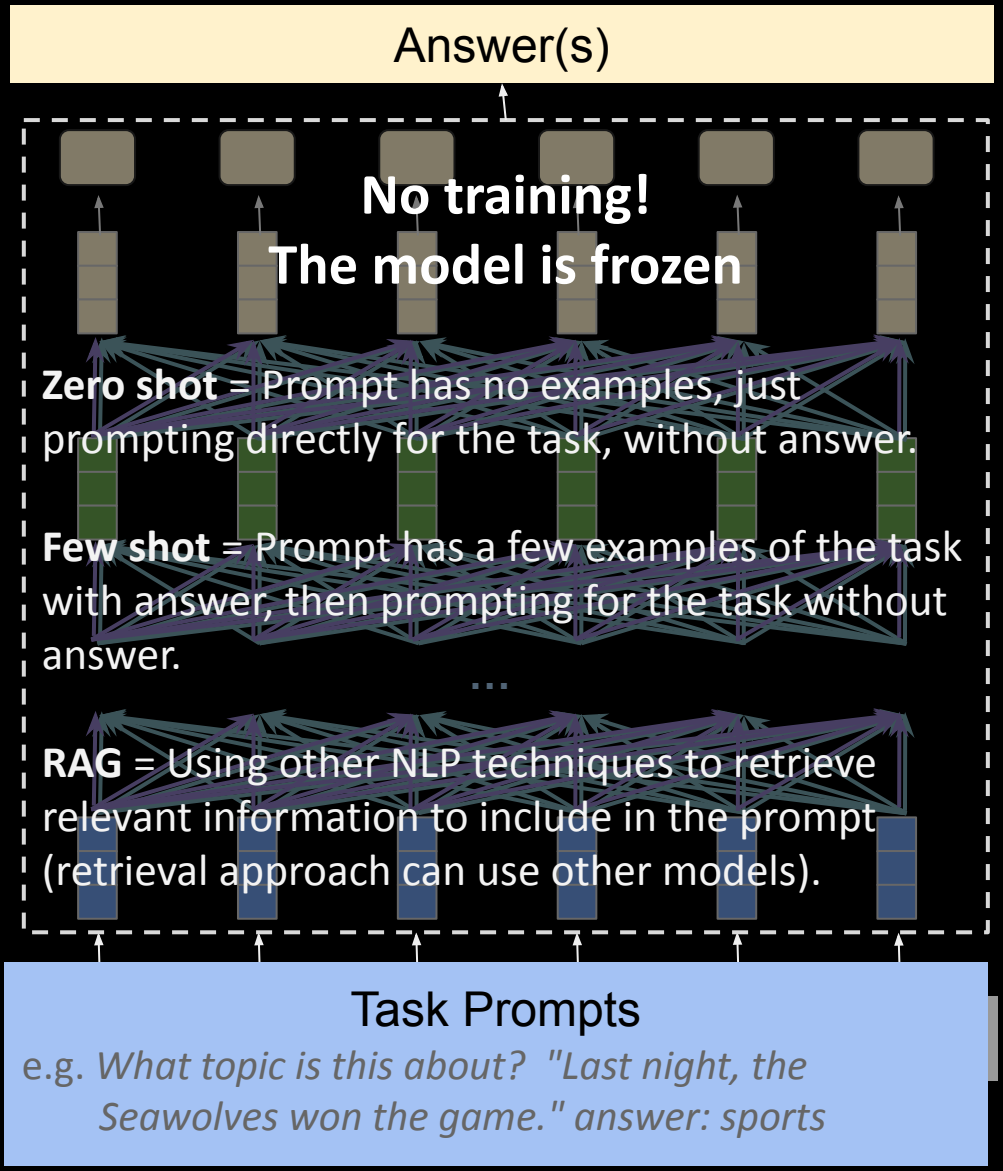
layer k:
(used for language modeling)

layer k-1:
(taken as contextual embedding)

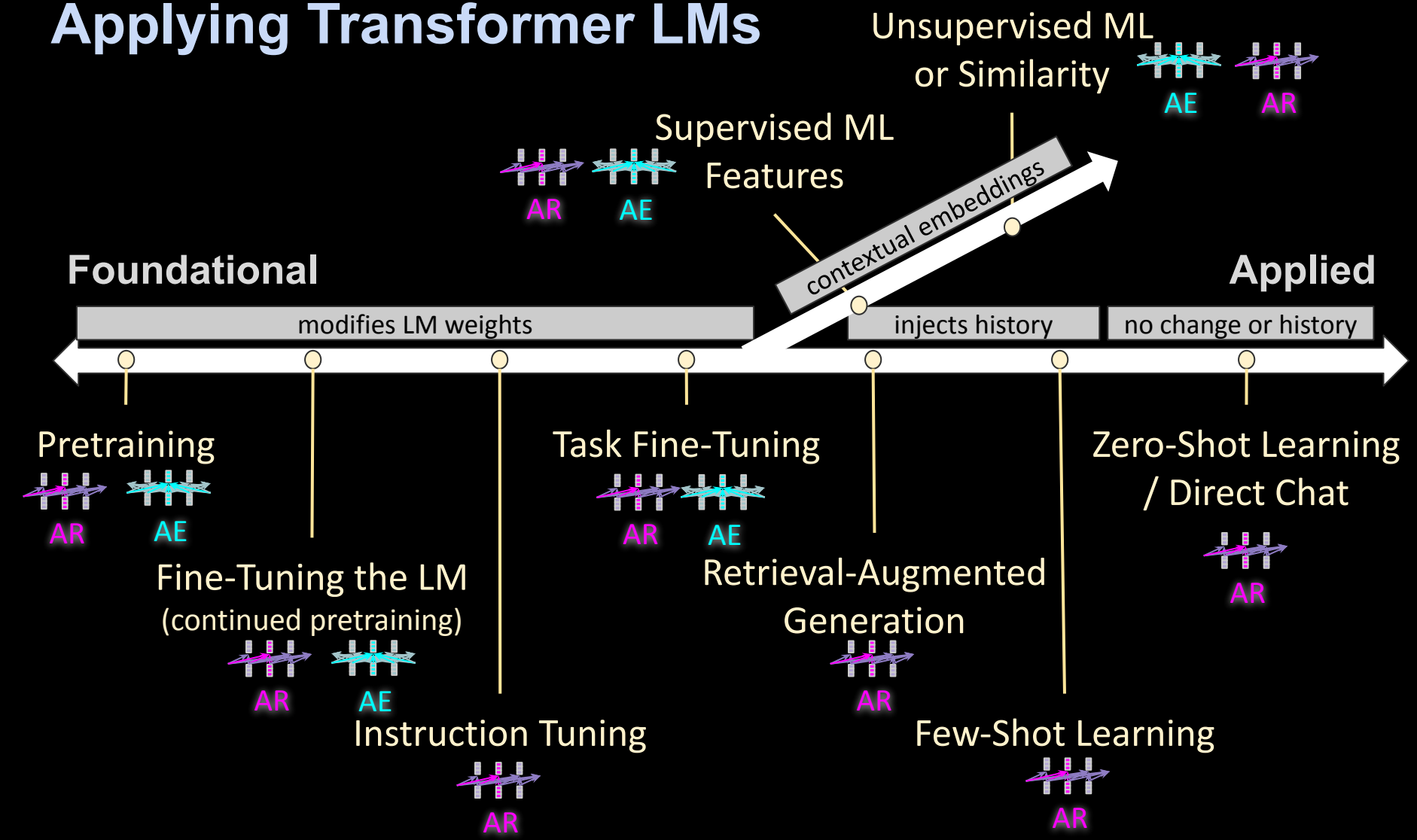
layers 1 to k-2:
(compose embeddings with context)

layer 0:
(input: word-type embeddings)

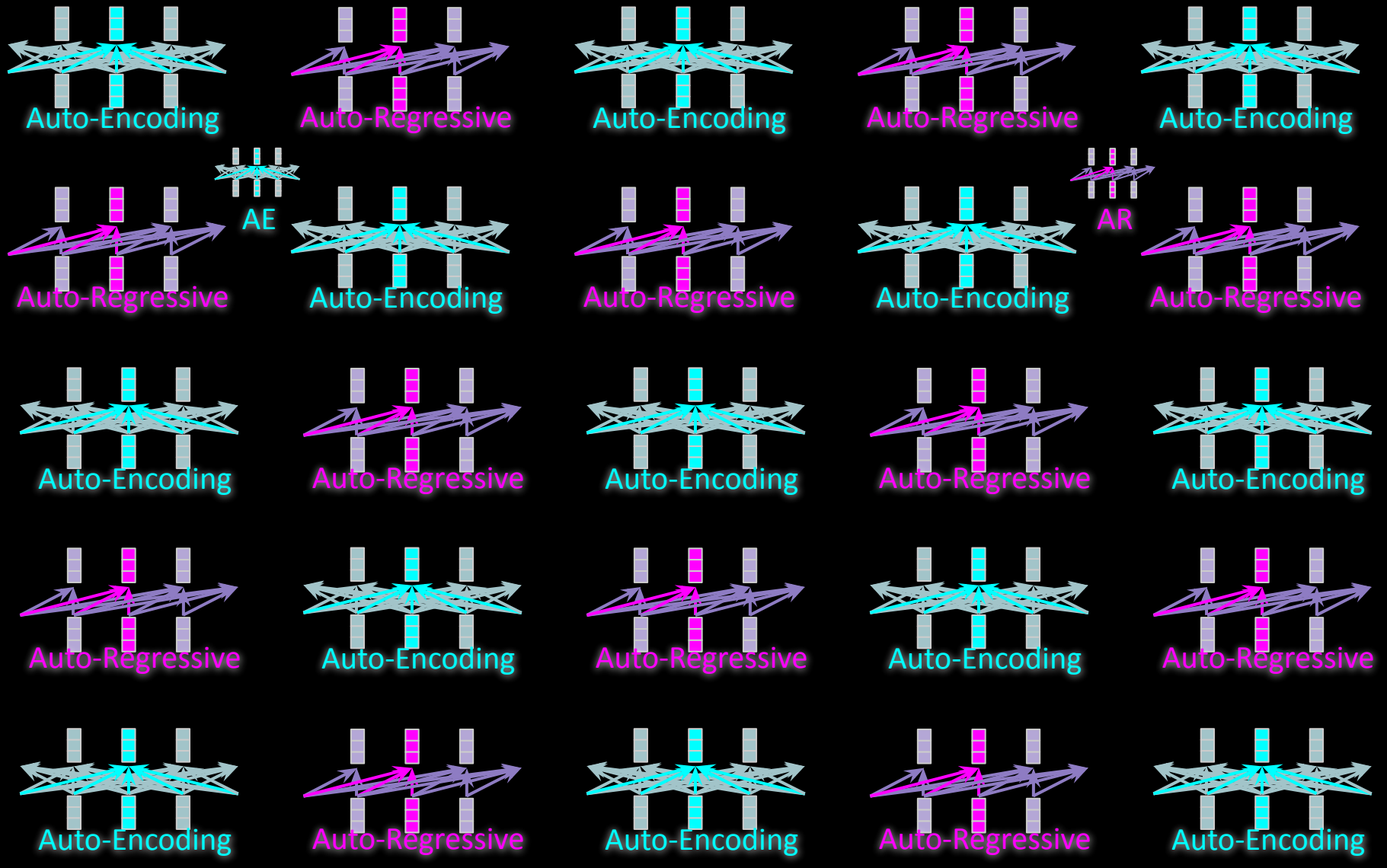
sentence (sequence) input:



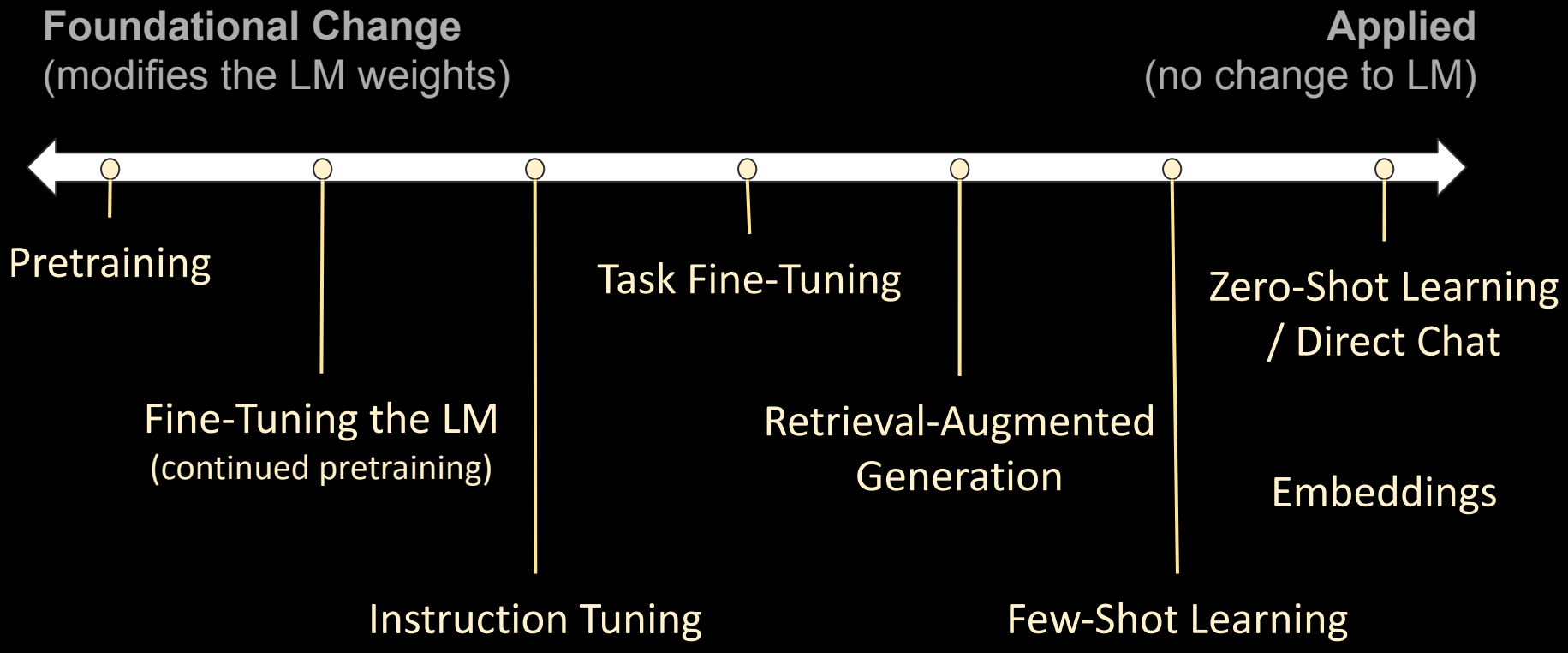
Applying Transformer LMs



Supplemental Review Material


Auto-Encoding
Auto-Regressive
Auto-Encoding
Auto-Regressive
Auto-Encoding
AE
AR
Auto-Regressive
Auto-Encoding
Auto-Regressive
Auto-Encoding
Auto-Regressive
Auto-Encoding
Auto-Regressive
Auto-Encoding
Auto-Regressive
Auto-Encoding
Auto-Regressive
Auto-Encoding
Auto-Regressive
Auto-Encoding
Auto-Regressive
Auto-Encoding
Auto-Encoding
Auto-Regressive
Auto-Encoding
Auto-Regressive
Auto-Encoding

simpler version



Pretraining; FTing the LM; Instruction Tuning

softmax for LM:

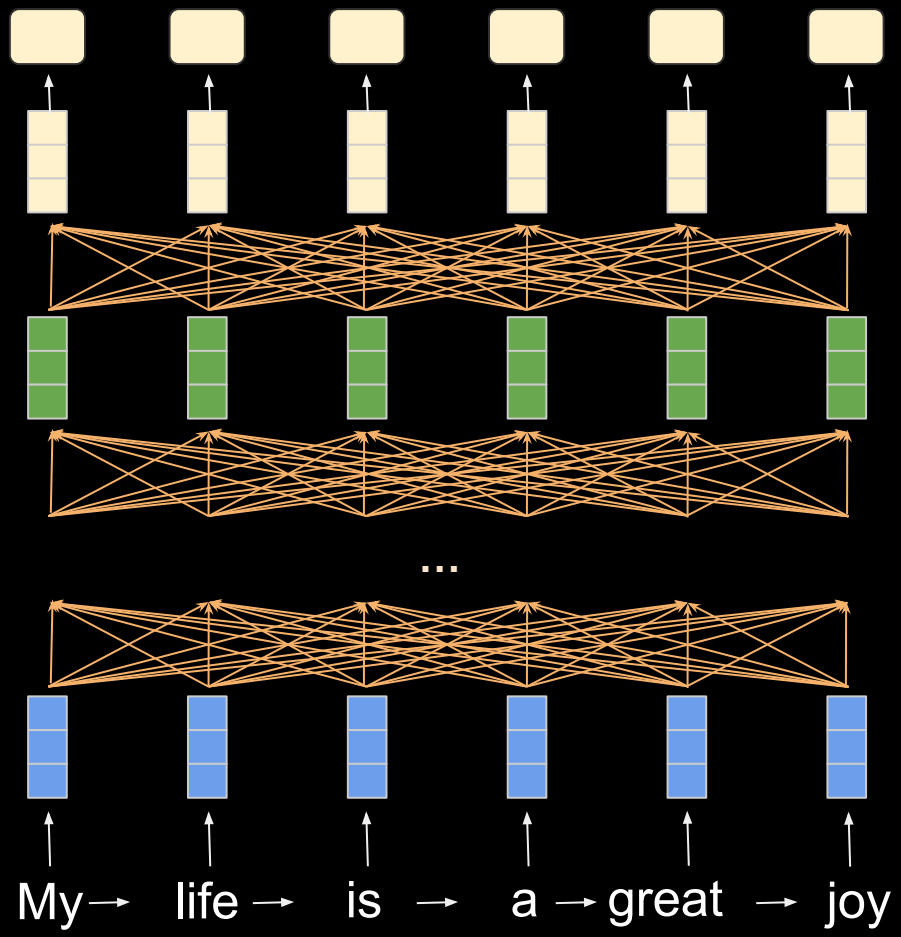
layer k:
(used for language modeling)

layer k-1:
(taken as contextual embedding)

layers 1 to k-2:
(compose embeddings with context)

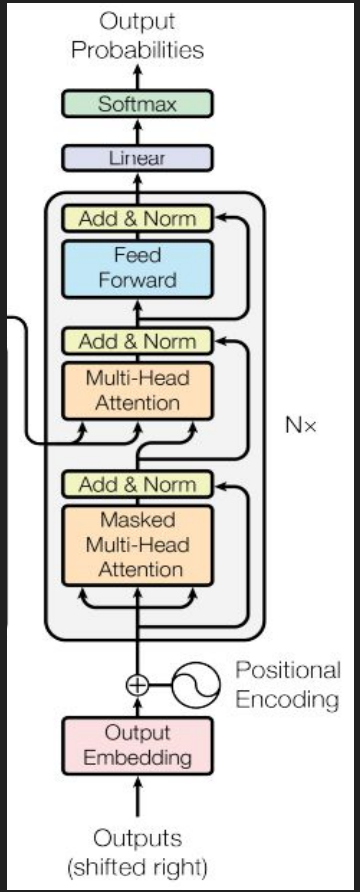
layer 0:
(input: word-type embeddings)

sentence (sequence) input:

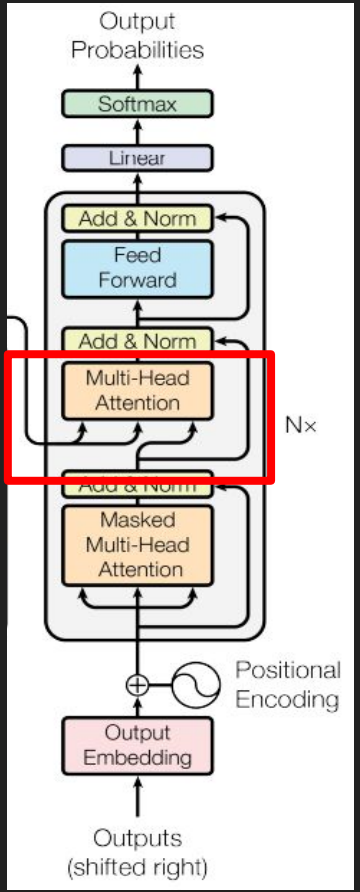


(Kjell, Kjell, and Schwartz, 2023)

Decoder


Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Positional Encoding
Output Embedding
Outputs (shifted right)

Decoder: Cross Attention



Decoder: Masked Multi-Head Attention