Natural Language Processing

CSE 447 / 547 M

Pre-training

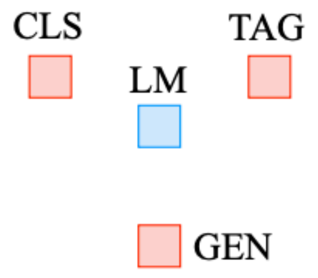
Lecturer: Kabir Ahuja

Slides adapted from Liwei Jiang, John Hewitt, Anna Goldie

Major Paradigms in NLP

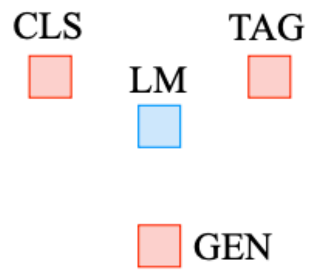
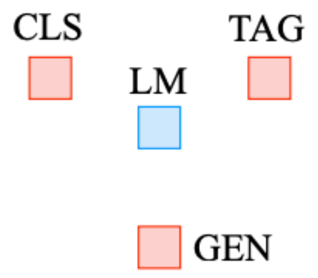
Liu et al. 2021

Major Paradigms in NLP

Paradigm	Engineering	Task Relation
a. Fully Supervised Learning (Non-Neural Network)	Features (e.g. word identity, part-of-speech, sentence length)	

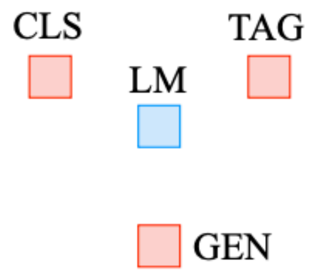
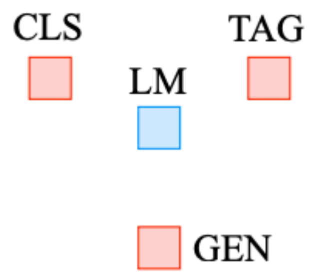
Liu et al. 2021

Major Paradigms in NLP

Paradigm	Engineering	Task Relation
a. Fully Supervised Learning (Non-Neural Network)	Features (e.g. word identity, part-of-speech, sentence length)	
b. Fully Supervised Learning (Neural Network)	Architecture (e.g. convolutional, recurrent, self-attentional)	

Liu et al. 2021

Major Paradigms in NLP

Paradigm	Engineering	Task Relation
a. Fully Supervised Learning (Non-Neural Network)	Features (e.g. word identity, part-of-speech, sentence length)	
b. Fully Supervised Learning (Neural Network)	Architecture (e.g. convolutional, recurrent, self-attentional)	

Liu et al. 2021

Major Paradigms in NLP

Paradigm	Engineering	Task Relation
a. Fully Supervised Learning (Non-Neural Network)	Features (e.g. word identity, part-of-speech, sentence length)	CLS TAG LM GEN
b. Fully Supervised Learning (Neural Network)	Architecture (e.g. convolutional, recurrent, self-attentional)	CLS TAG LM GEN

Pre 2017

Liu et al. 2021

Major Paradigms in NLP

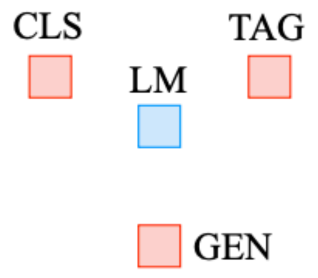
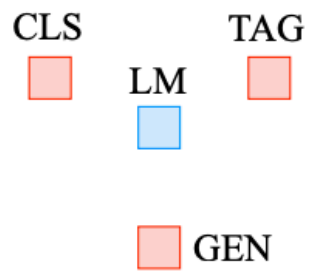
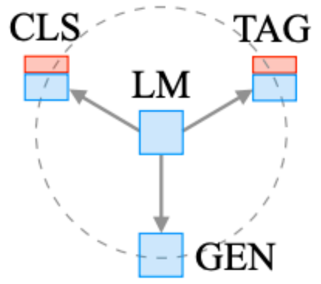
Paradigm	Engineering	Task Relation
a. Fully Supervised Learning (Non-Neural Network)	Features (e.g. word identity, part-of-speech, sentence length)	CLS LM TAG GEN
b. Fully Supervised Learning (Neural Network)	Architecture (e.g. convolutional, recurrent, self-attentional)	CLS LM TAG GEN

Pre 2017

What we have seen so far

Liu et al. 2021

Major Paradigms in NLP

Paradigm	Engineering	Task Relation
a. Fully Supervised Learning (Non-Neural Network)	Features (e.g. word identity, part-of-speech, sentence length)	
b. Fully Supervised Learning (Neural Network)	Architecture (e.g. convolutional, recurrent, self-attentional)	
c. Pre-train, Fine-tune	Objective (e.g. masked language modeling, next sentence prediction)	

Pre 2017

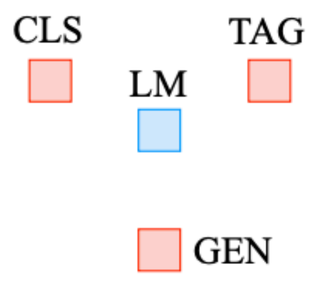
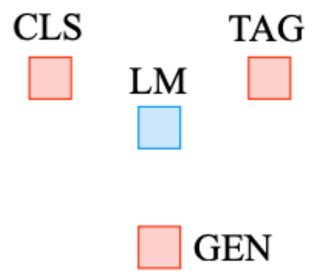
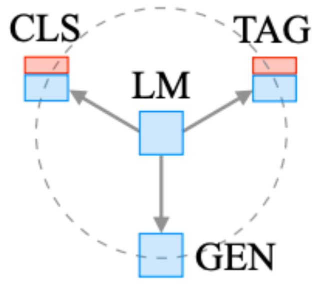
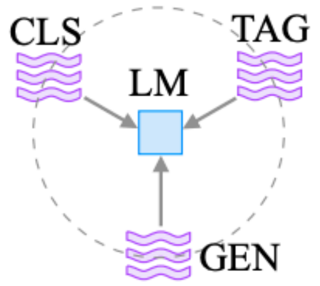
2017-2019

What we have seen so far

Liu et al. 2021

Major Paradigms in NLP

What we have seen so far

Paradigm	Engineering	Task Relation
a. Fully Supervised Learning (Non-Neural Network)	Features (e.g. word identity, part-of-speech, sentence length)	
b. Fully Supervised Learning (Neural Network)	Architecture (e.g. convolutional, recurrent, self-attentional)	
c. Pre-train, Fine-tune	Objective (e.g. masked language modeling, next sentence prediction)	
d. Pre-train, Prompt, Predict	Prompt (e.g. cloze, prefix)	

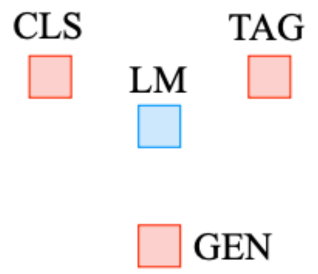
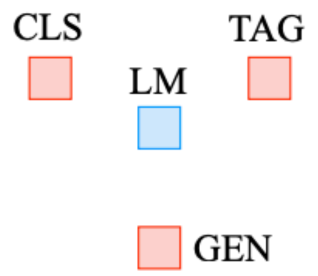
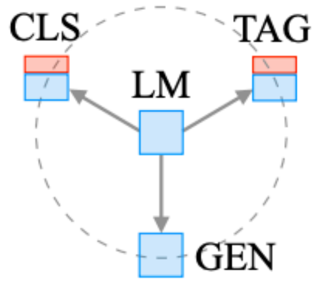
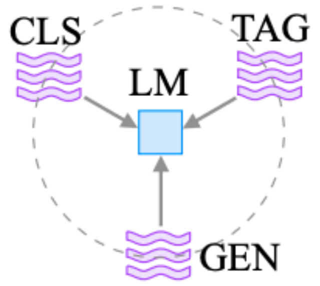
Pre 2017

2017-2019

2021- Present?

Liu et al. 2021

Major Paradigms in NLP

Paradigm	Engineering	Task Relation	
a. Fully Supervised Learning (Non-Neural Network)	Features (e.g. word identity, part-of-speech, sentence length)		Pre 2017
b. Fully Supervised Learning (Neural Network)	Architecture (e.g. convolutional, recurrent, self-attentional)		
c. Pre-train, Fine-tune	Objective (e.g. masked language modeling, next sentence prediction)		2017-2019
d. Pre-train, Prompt, Predict	Prompt (e.g. cloze, prefix)		2021- Present?
e. Pre-train, Alignment, (Fine-tune), Predict			2022- Present

What we have seen so far

Liu et al. 2021

Major Paradigms in NLP

Paradigm	Engineering	Task Relation
a. Fully Supervised Learning (Non-Neural Network)	Features (e.g. word identity, part-of-speech, sentence length)	CLS TAG LM GEN
b. Fully Supervised Learning (Neural Network)	Architecture (e.g. convolutional, recurrent, self-attentional)	CLS TAG LM GEN
c. Pre-train, Fine-tune	Objective (e.g. masked language modeling, next sentence prediction)	CLS TAG LM GEN
d. Pre-train, Prompt, Predict	Prompt (e.g. cloze, prefix)	CLS TAG LM GEN
e. Pre-train, Alignment, (Fine-tune), Predict		

What we have seen so far

Pre 2017

2017-2019

2021- Present?

2022- Present

Liu et al. 2021

Major Paradigms in NLP

Paradigm	Engineering	Task Relation
a. Fully Supervised Learning (Non-Neural Network)	Features (e.g. word identity, part-of-speech, sentence length)	CLS TAG LM GEN
b. Fully Supervised Learning (Neural Network)	Architecture (e.g. convolutional, recurrent, self-attentional)	CLS TAG LM GEN
c. Pre-train, Fine-tune		
d. Pre-train, Prompt, Predict	Prompt (e.g. cloze, prefix)	CLS TAG LM GEN
e. Pre-train, Alignment, (Fine-tune), Predict		

Liu et al. 2021

What we have seen so far

Pre 2017

Pre-training common across all major paradigms post 2017

2017-2019

2021- Present?

2022- Present

Major Paradigms in NLP

Paradigm	Engineering	Task Relation
a. Fully Supervised Learning (Non-Neural Network)	Features (e.g. word identity, part-of-speech, sentence length)	CLS TAG LM GEN
b. Fully Supervised Learning (Neural Network)	Architecture (e.g. convolutional, recurrent, self-attentional)	CLS TAG LM GEN
c. Pre-train, Fine-tune		
d. Pre-train, Prompt, Predict	Prompt (e.g. cloze, prefix)	CLS TAG LM GEN
e. Pre-train, Alignment, (Fine-tune), Predict		

Liu et al. 2021

Pre 2017

2017-2019

2021- Present?

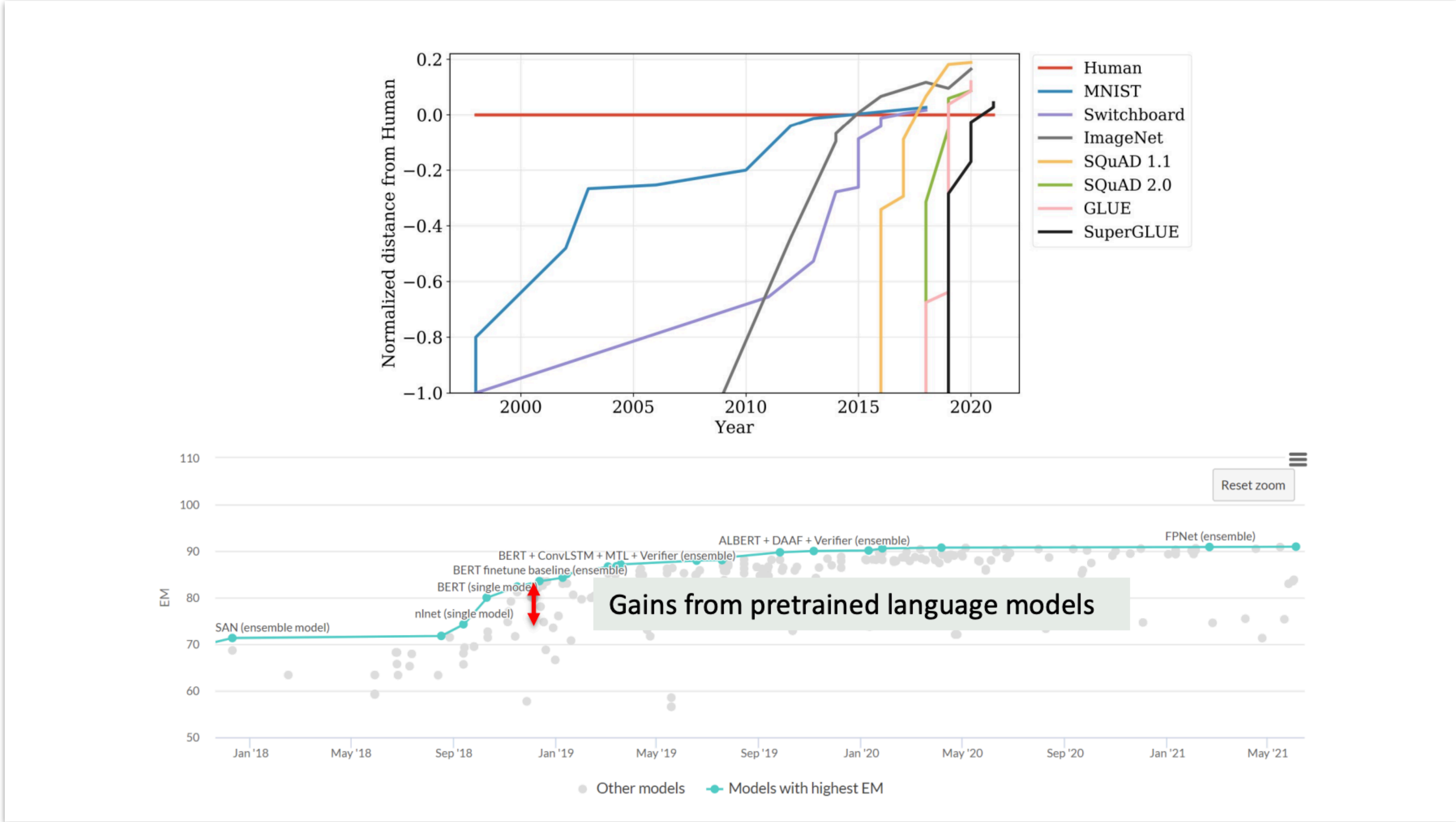
2022- Present

What we have seen so far

Pre-training common across all major paradigms post 2017

What we will see in the coming lectures

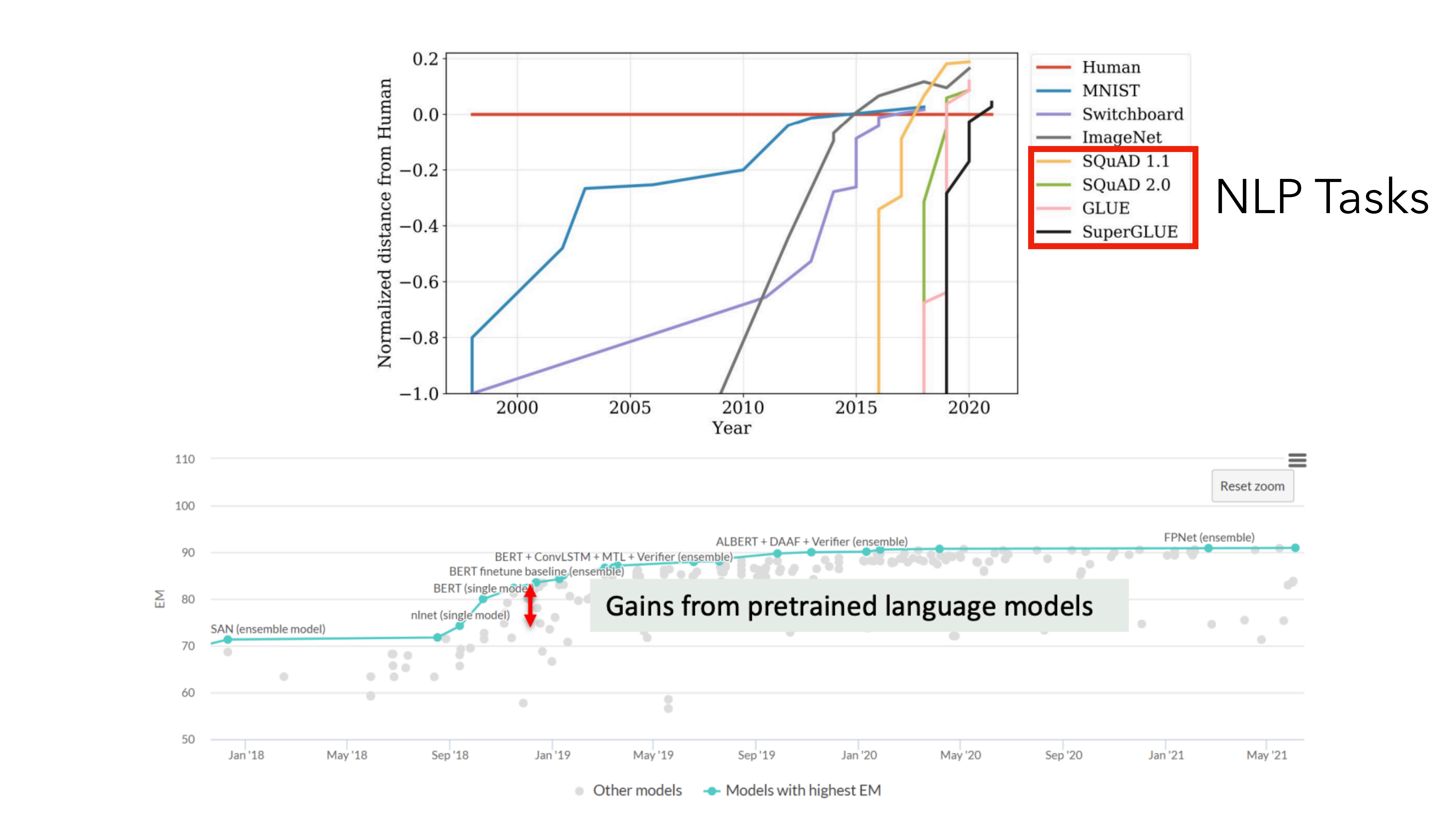
The Pre-training Revolution



Pre-training has had a major, tangible impact on how well NLP systems work

Slide from Chris Manning, Lecture 9: Pre-training, CS224n Spring 2024

The Pre-training Revolution



Pre-training has had a major, tangible impact on how well NLP systems work

Slide from Chris Manning, Lecture 9: Pre-training, CS224n Spring 2024

Lecture Outline

1. Motivating Pre-training, aka Self-supervised Learning
2. Pre-training Architectures and Training Objectives
    1. Encoders
    2. Encoder-Decoders
    3. Decoder

Lecture Outline

1. Motivating Pre-training, aka Self-supervised Learning
2. Pre-training Architectures and Training Objectives
    1. Encoders
    2. Encoder-Decoders
    3. Decoder

Issues with Fully Supervised Learning Approaches

Food Review: "I recently had the pleasure of dining at Fusion Bites, and the experience was nothing short of spectacular. The menu boasts an exciting blend of global flavors, and each dish is a masterpiece in its own right."

Say that we are given a dataset of 100K food reviews with sentiment labels, **how do we train a model to perform sentiment analysis over unseen food reviews?**

Issues with Fully Supervised Learning Approaches

Food Review: "I recently had the pleasure of dining at Fusion Bites, and the experience was nothing short of spectacular. The menu boasts an exciting blend of global flavors, and each dish is a masterpiece in its own right."

Say that we are given a dataset of 100K food reviews with sentiment labels, **how do we train a model to perform sentiment analysis over unseen food reviews?**

We can directly train a randomly initialized model to take in food review texts and output "positive" or "negative" sentiment labels.

Issues with Fully Supervised Learning Approaches

Food Review: “I recently had the pleasure of dining at Fusion Bites, and the experience was nothing short of spectacular. The menu boasts an exciting blend of global flavors, and each dish is a masterpiece in its own right.”

Movie Review: "The narrative unfolds with a steady pace, showcasing a blend of various elements. While the performances are competent, and the cinematography captures the essence of the story, the overall impact falls somewhere in the middle."

If we are instead given **movie reviews** to classify, can we use the same system trained from food reviews to predict the sentiment?

Issues with Fully Supervised Learning Approaches

Food Review: "I recently had the pleasure of dining at Fusion Bites, and the experience was nothing short of spectacular. The menu boasts an exciting blend of global flavors, and each dish is a masterpiece in its own right."

Movie Review: "The narrative unfolds with a steady pace, showcasing a blend of various elements. While the performances are competent, and the cinematography captures the essence of the story, the overall impact falls somewhere in the middle."

If we are instead given **movie reviews** to classify, can we use the same system trained from food reviews to predict the sentiment?

May NOT generalize well due to distributional shift!

Issues with Fully Supervised Learning Approaches

Food Review: “I recently had the pleasure of dining at Fusion Bites, and the experience was nothing short of spectacular. The menu boasts an exciting blend of global flavors, and each dish is a masterpiece in its own right.”

Movie Review: "The narrative unfolds with a steady pace, showcasing a blend of various elements. While the performances are competent, and the cinematography captures the essenc

somewhere in the middle."

Fully Supervised Learning

Collect a labeled dataset for movie reviews and train a model from scratch on this new dataset

If we are instead given **movie**

system trained from food reviews t

May NOT generalize well due to distributional shift!

Transfer Learning: A History Lesson from Computer Vision

- Instead of training a randomly initialized neural network every time we encounter a new task or domain,
  - can we re-use the learned representations from one task/domain for another?

Image from Lecture 7 CS231n slides by Fei-Fei Li, Ehsan Adeli, Zane Durante

Transfer Learning: A History Lesson from Computer Vision

- Instead of training a randomly initialized neural network every time we encounter a new task or domain,
  - can we re-use the learned representations from one task/domain for another?

Idea: Train a **(very) deep neural network** on a **large-scale dataset** and re-use the learned representations from this network to adapt to new tasks

Image from Lecture 7 CS231n slides by Fei-Fei Li, Ehsan Adeli, Zane Durante

Transfer Learning: A History Lesson from Computer Vision

- Instead of training a randomly initialized neural network every time we encounter a new task or domain,
  - can we re-use the learned representations from one task/domain for another?

Idea: Train a **(very) deep neural network** on a **large-scale dataset** and re-use the learned representations from this network to adapt to new tasks

ImageNet Challenge

IMAGENET

- 1,000 object classes (categories).
- Images:
  - 1.2 M train
  - 100k test.

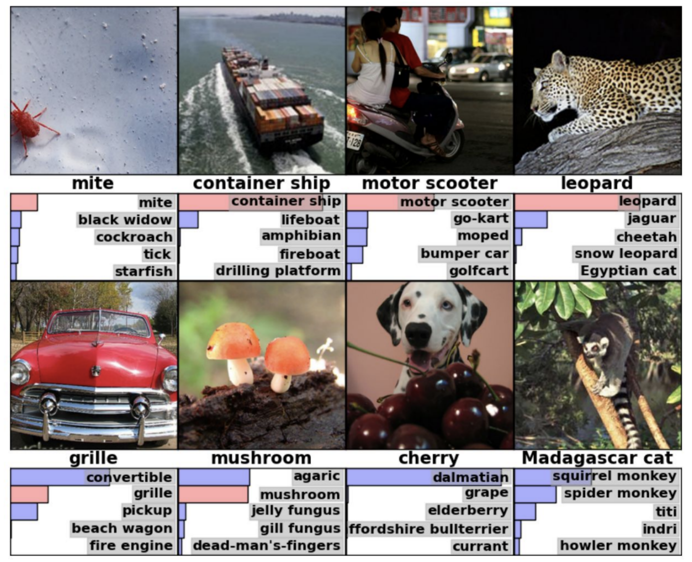


Image from Lecture 7 CS231n slides by Fei-Fei Li, Ehsan Adeli, Zane Durante

Transfer Learning: A History Lesson from Computer Vision

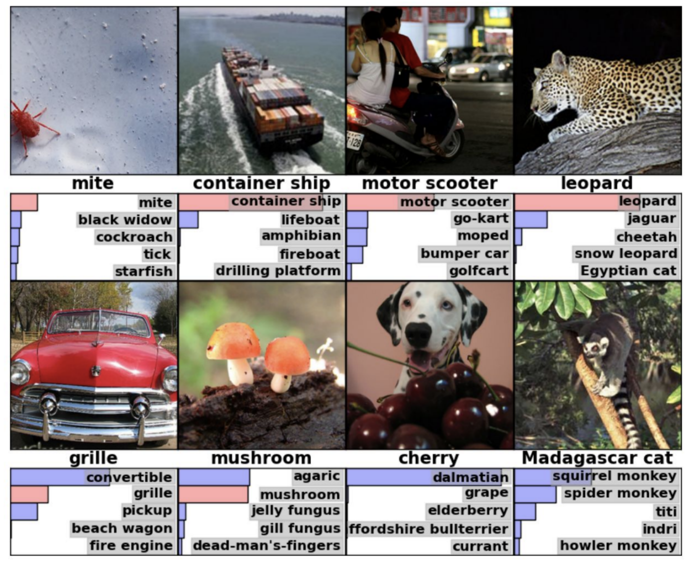
- Instead of training a randomly initialized neural network every time we encounter a new task or domain,
  - can we re-use the learned representations from one task/domain for another?

Idea: Train a **(very) deep neural network** on a **large-scale dataset** and re-use the learned representations from this network to adapt to new tasks

ImageNet Challenge

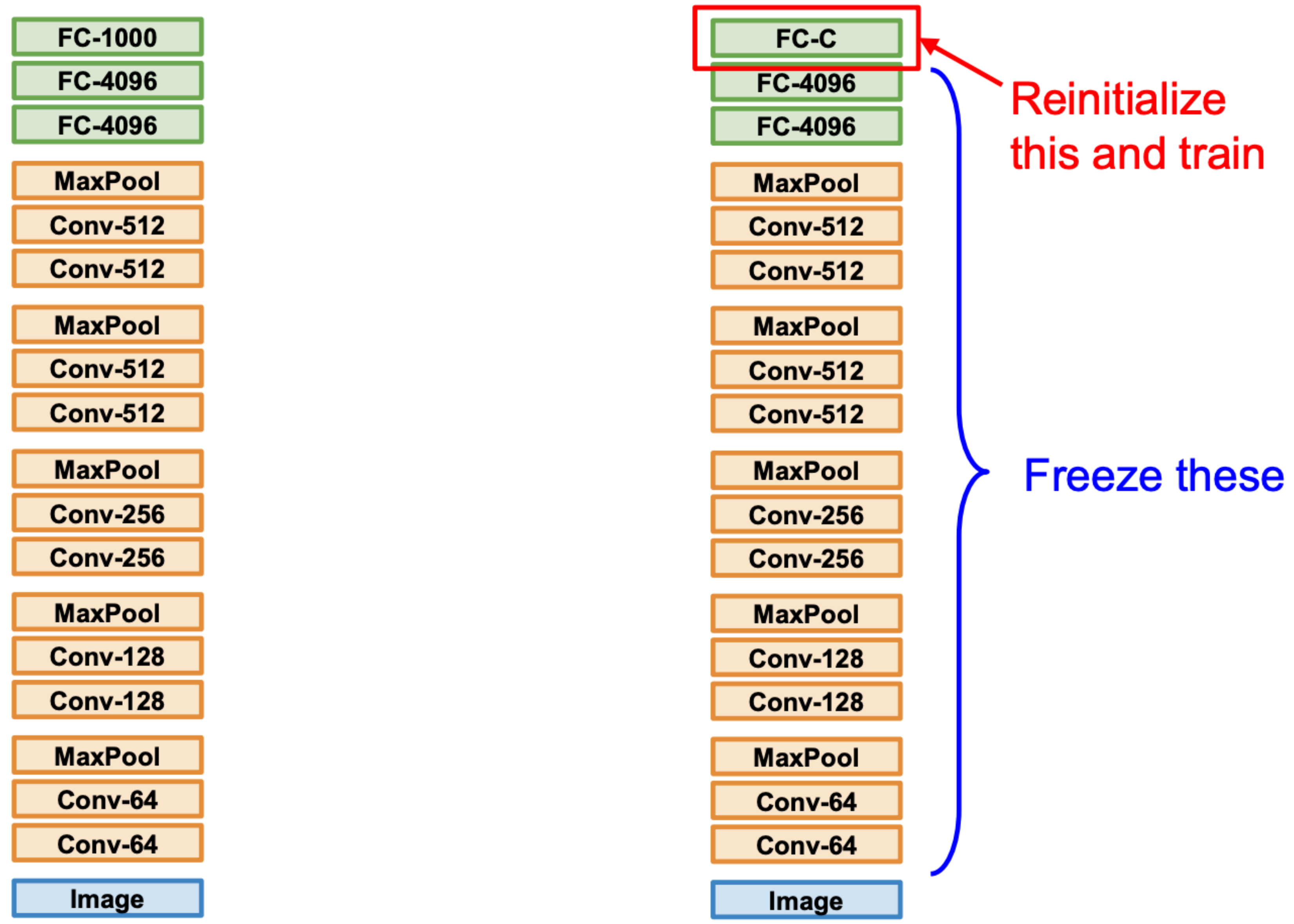
IMAGENET

- 1,000 object classes (categories).
- Images:
  - 1.2 M train
  - 100k test.

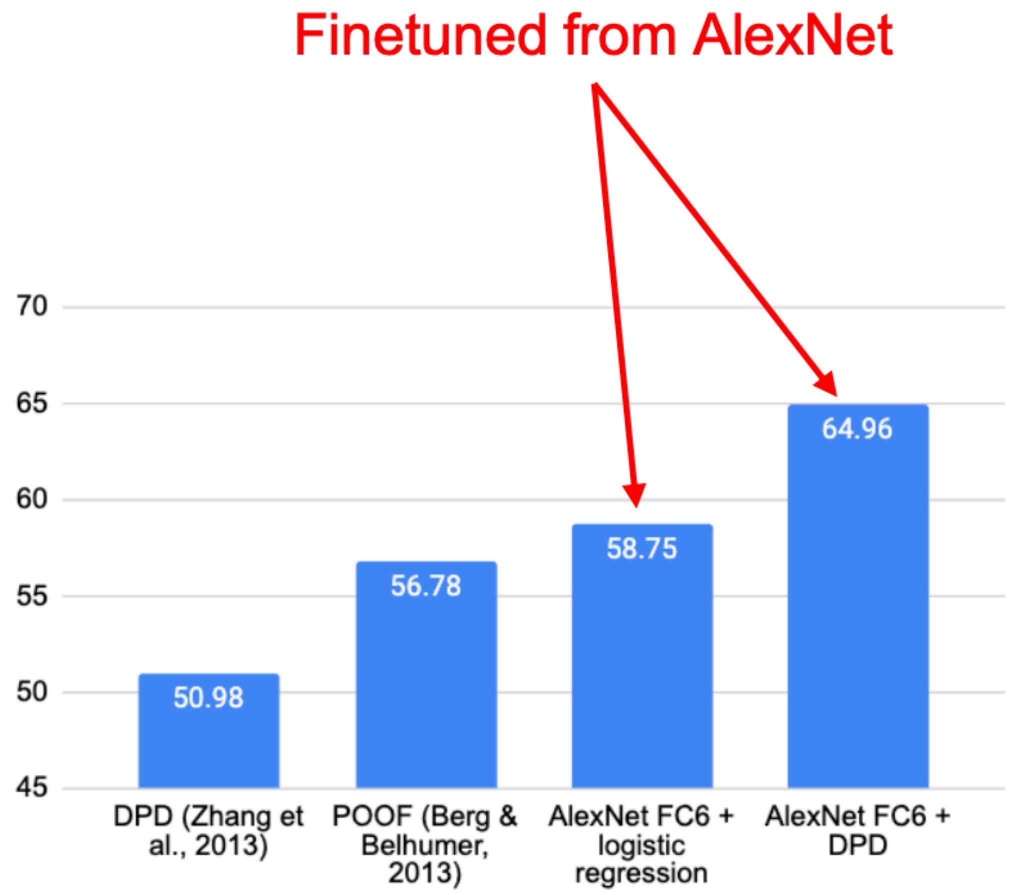


1. Train on Imagenet

2. Small Dataset (C classes)



This is called Fine-tuning!



Donahue et al, "DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition", ICML 2014

Image from Lecture 7 CS231n slides by Fei-Fei Li, Ehsan Adeli, Zane Durante

Transfer Learning: A History Lesson from Computer Vision

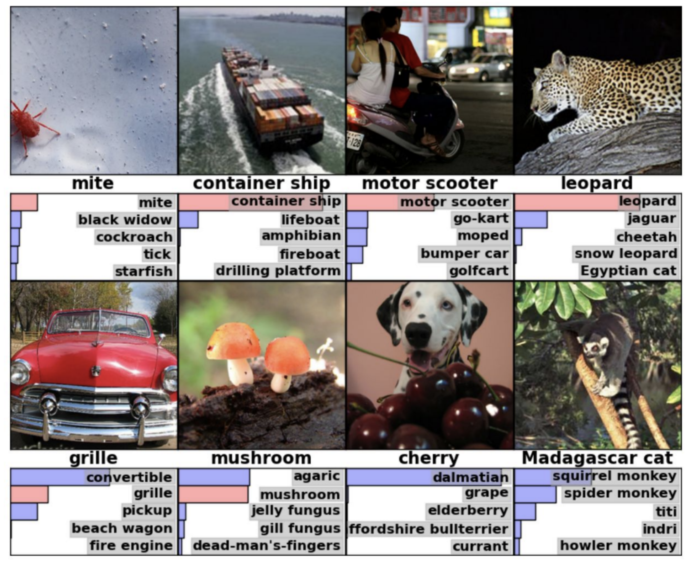
- Instead of training a randomly initialized neural network every time we encounter a new task or domain,
  - can we re-use the learned representations from one task/domain for another?

Idea: Train a **(very) deep neural network** on a **large-scale dataset** and re-use the learned representations from this network to adapt to new tasks

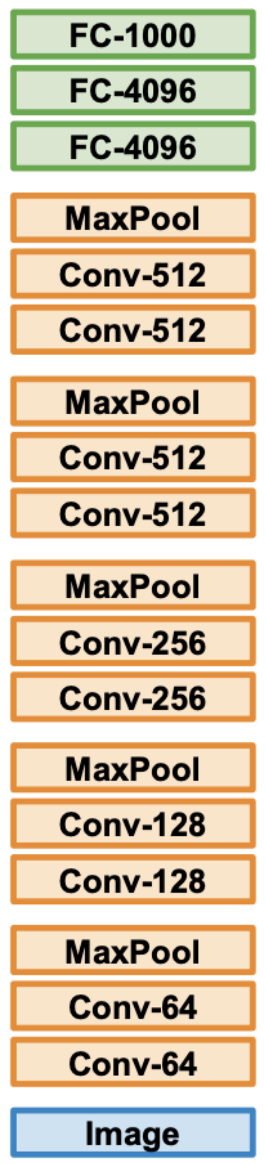
ImageNet Challenge

IMAGENET

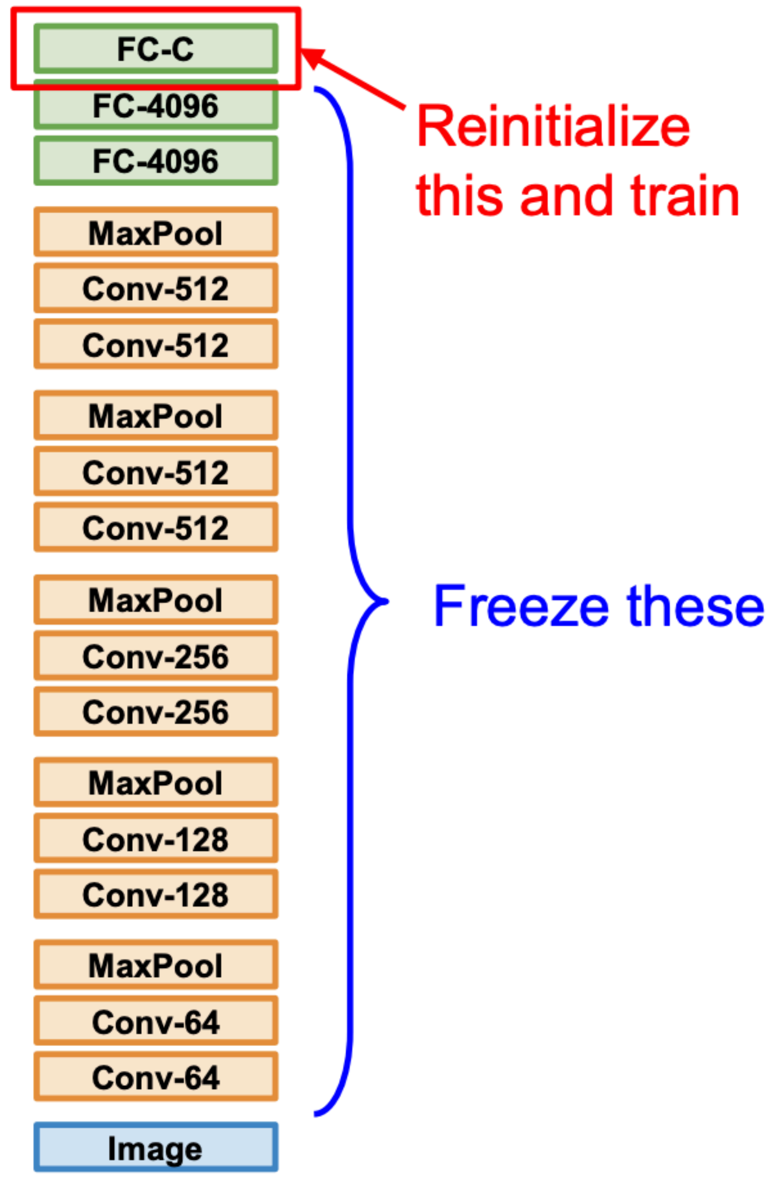
- 1,000 object classes (categories).
- Images:
  - 1.2 M train
  - 100k test.



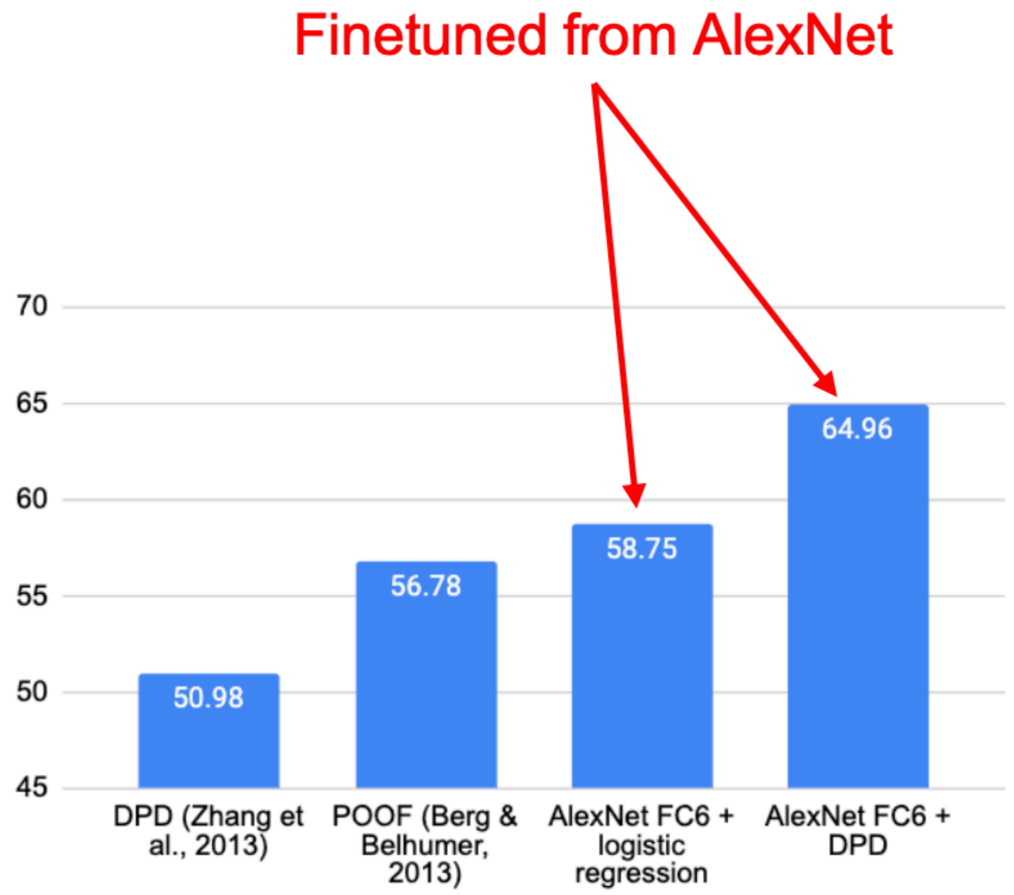
1. Train on Imagenet



2. Small Dataset (C classes)



This is called Fine-tuning!



Donahue et al, "DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition", ICML 2014

- A very successful recipe for adapting to different vision tasks like object detection, semantic segmentation, pose estimation, etc.
- Also, reduced the reliance on large training datasets to achieve good performance

Image from Lecture 7 CS231n slides by Fei-Fei Li, Ehsan Adeli, Zane Durante

Why it took so long for NLP?

- Since 2014, it had become common practice in the Computer Vision community to download a pre-trained (on Image Net) deep neural network model and "fine-tune" it on the problem at hand instead of starting from scratch.

- This wasn't the case in NLP till late 2017s.

- It was common to use pre-trained word vectors like word2vec, GloVe for NLP tasks, and while those would help boost performance, most often it was a marginal improvement.

Why it took so long for NLP?

- Since 2014, it had become common practice in the Computer Vision community to download a pre-trained (on Image Net) deep neural network model and "fine-tune" it on the problem at hand instead of starting from scratch.

- This wasn't the case in NLP till late 2017s.

- It was common to use pre-trained word vectors like word2vec, GloVe for NLP tasks, and while those would help boost performance, most often it was a marginal improvement.

You might have seen this already while attempting HW2

Why it took so long for NLP?

Why it took so long for NLP?

We can mostly boil down this delay to two factors

Why it took so long for NLP?

We can mostly boil down this delay to two factors

1. Lack of a large-scale general dataset

    1. It wasn’t clear what would be a suitable NLP task most representative of the space of NLP tasks (classification, QA, NLI, Parsing, Language Modeling?). Getting high-quality label at such a large scale was also a challenge.

Why it took so long for NLP?

We can mostly boil down this delay to two factors

1. Lack of a large-scale general dataset
    1. It wasn't clear what would be a suitable NLP task most representative of the space of NLP tasks (classification, QA, NLI, Parsing, Language Modeling?). Getting high-quality label at such a large scale was also a challenge.
2. Neural Network Models for NLP were usually very shallow
    1. Pre-2017, dominant models used in NLP were recurrent neural networks e.g. LSTMs
    2. These models were usually 1-2 hidden layers, and scaling them to a large number of layers was non-trivial as these models were notoriously hard to train

Why it took so long for NLP?

What changed starting from 2017?

We can mostly boil down this delay to two factors

1. Lack of a large-scale general dataset
    1. It wasn't clear what would be a suitable NLP task most representative of the space of NLP tasks (classification, QA, NLI, Parsing, Language Modeling?). Getting high-quality label at such a large scale was also a challenge.
2. Neural Network Models for NLP were usually very shallow
    1. Pre-2017, dominant models used in NLP were recurrent neural networks e.g. LSTMs
    2. These models were usually 1-2 hidden layers, and scaling them to a large number of layers was non-trivial as these models were notoriously hard to train

Why it took so long for NLP?

What changed starting from 2017?

We can mostly boil down this delay to two factors

1. Lack of a large-scale general dataset **Self-supervised Learning**
    1. It wasn't clear what would be a suitable NLP task most representative of the space of NLP tasks (classification, QA, NLI, Parsing, Language Modeling?). Getting high-quality label at such a large scale was also a challenge.
2. Neural Network Models for NLP were usually very shallow
    1. Pre-2017, dominant models used in NLP were recurrent neural networks e.g. LSTMs
    2. These models were usually 1-2 hidden layers, and scaling them to a large number of layers was non-trivial as these models were notoriously hard to train

Part of input data itself provides labels instead of requring external labels. What SSL model have we already seen?

for NLP?

What changed starting from 2017?

We can ... to two factors

Self-supervised Learning

1. Lack of a large-scale general dataset
    1. It wasn't clear what would be a suitable NLP task most representative of the space of NLP tasks (classification, QA, NLI, Parsing, Language Modeling?). Getting high-quality label at such a large scale was also a challenge.
2. Neural Network Models for NLP were usually very shallow
    1. Pre-2017, dominant models used in NLP were recurrent neural networks e.g. LSTMs
    2. These models were usually 1-2 hidden layers, and scaling them to a large number of layers was non-trivial as these models were notoriously hard to train

Part of input data itself provides labels instead of requring external labels. What SSL model have we already seen?
Language Models!

for NLP?

What changed starting from 2017?

We can ... to two factors

Self-supervised Learning

1. Lack of a large-scale general dataset
    1. It wasn't clear what would be a suitable NLP task most representative of the space of NLP tasks (classification, QA, NLI, Parsing, Language Modeling?). Getting high-quality label at such a large scale was also a challenge.
2. Neural Network Models for NLP were usually very shallow
    1. Pre-2017, dominant models used in NLP were recurrent neural networks e.g. LSTMs
    2. These models were usually 1-2 hidden layers, and scaling them to a large number of layers was non-trivial as these models were notoriously hard to train

Part of input data itself provides labels instead of requring external labels. What SSL model have we already seen?

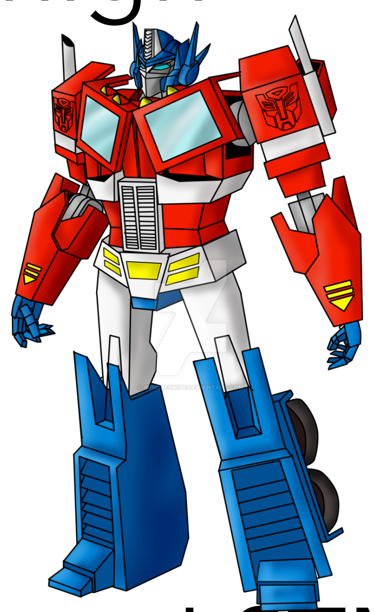
Language Models!

for NLP?

What changed starting from 2017?

We ca… to two factors

1. Lack of a large-scale general dataset **Self-supervised Learning**
    1. It wasn't clear what would be a suitable NLP task most representative of the space of NLP tasks (classification, QA, NLI, Parsing, Language Modeling?). Getting high-quality label at such a large scale was also a challenge.
2. Neural Network Models for NLP were usually very shallow **Transformers**

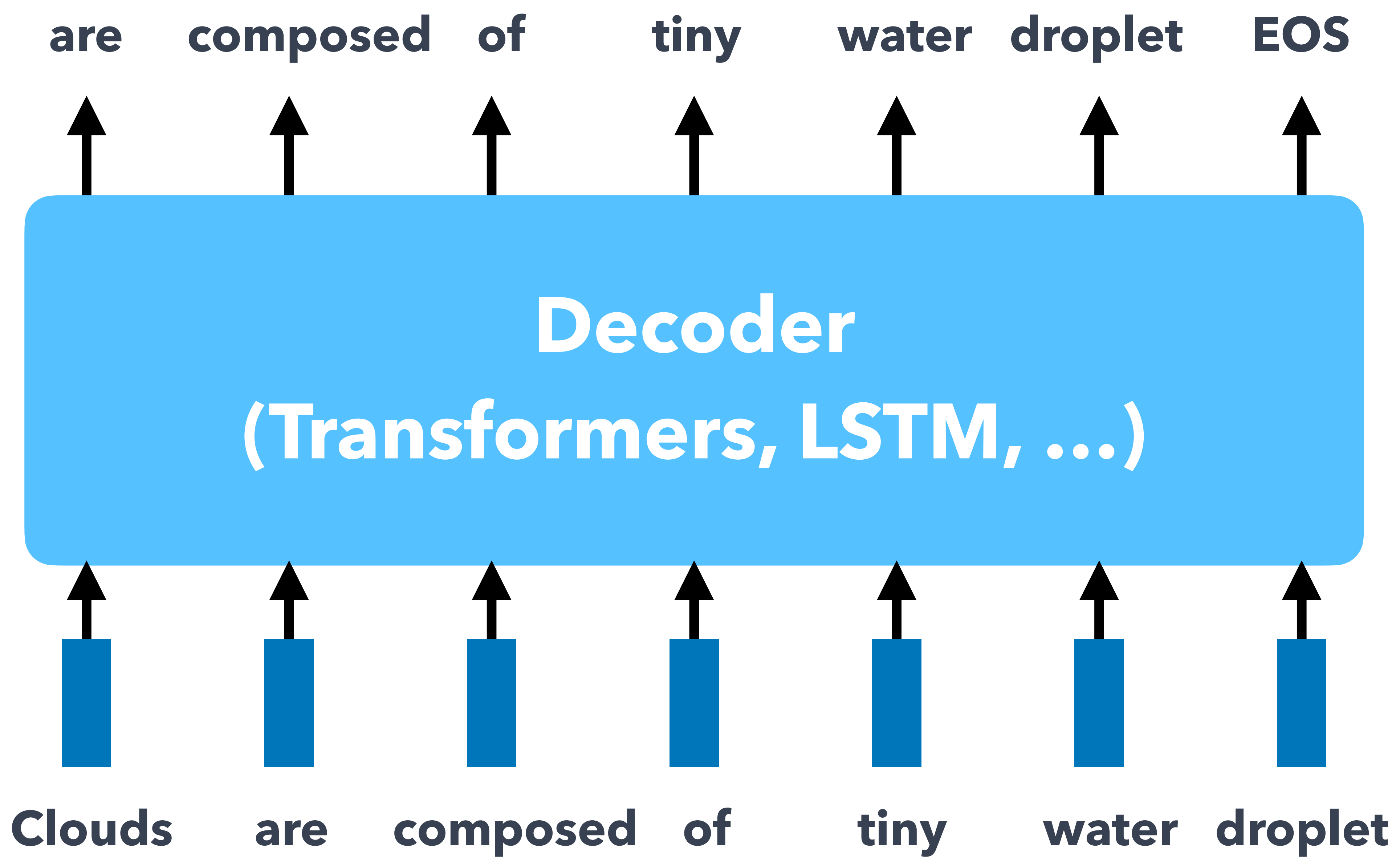


    1. Pre-2017, dominant models used in NLP were recurrent neural networks e.g. LSTMs
    2. These models were usually 1-2 hidden layers, and scaling them to a large number of layers was non-trivial as these models were notoriously hard to train

Self-supervised Pre-training for Learning Underlying Patterns, Structures, and Semantic Knowledge

Self-supervised Pre-training for Learning Underlying Patterns, Structures, and Semantic Knowledge

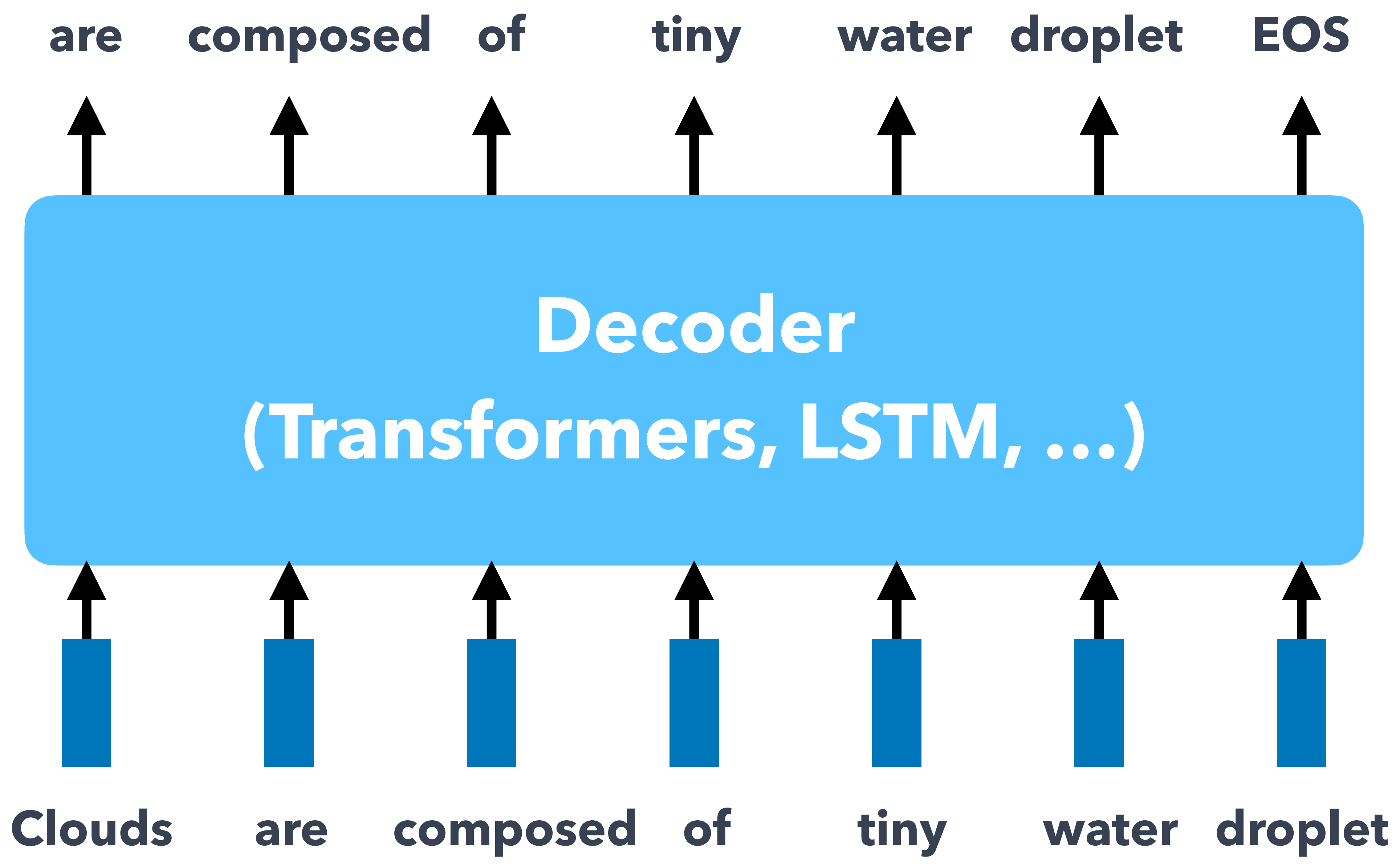
- Pre-training through **language modeling** [Dai and Le, 2015]
  - Model $P_\theta(w_t \mid w_{1:t-1})$, the probability distribution of the next word given previous contexts.
  - **There's lots of (English) data for this!** E.g., books, websites.
  - **Self-supervised** training of a neural network to perform the language modeling task with massive raw text data.
  - Save the network parameters to reuse later.



Self-supervised Pre-training for Learning Underlying Patterns, Structures, and Semantic Knowledge

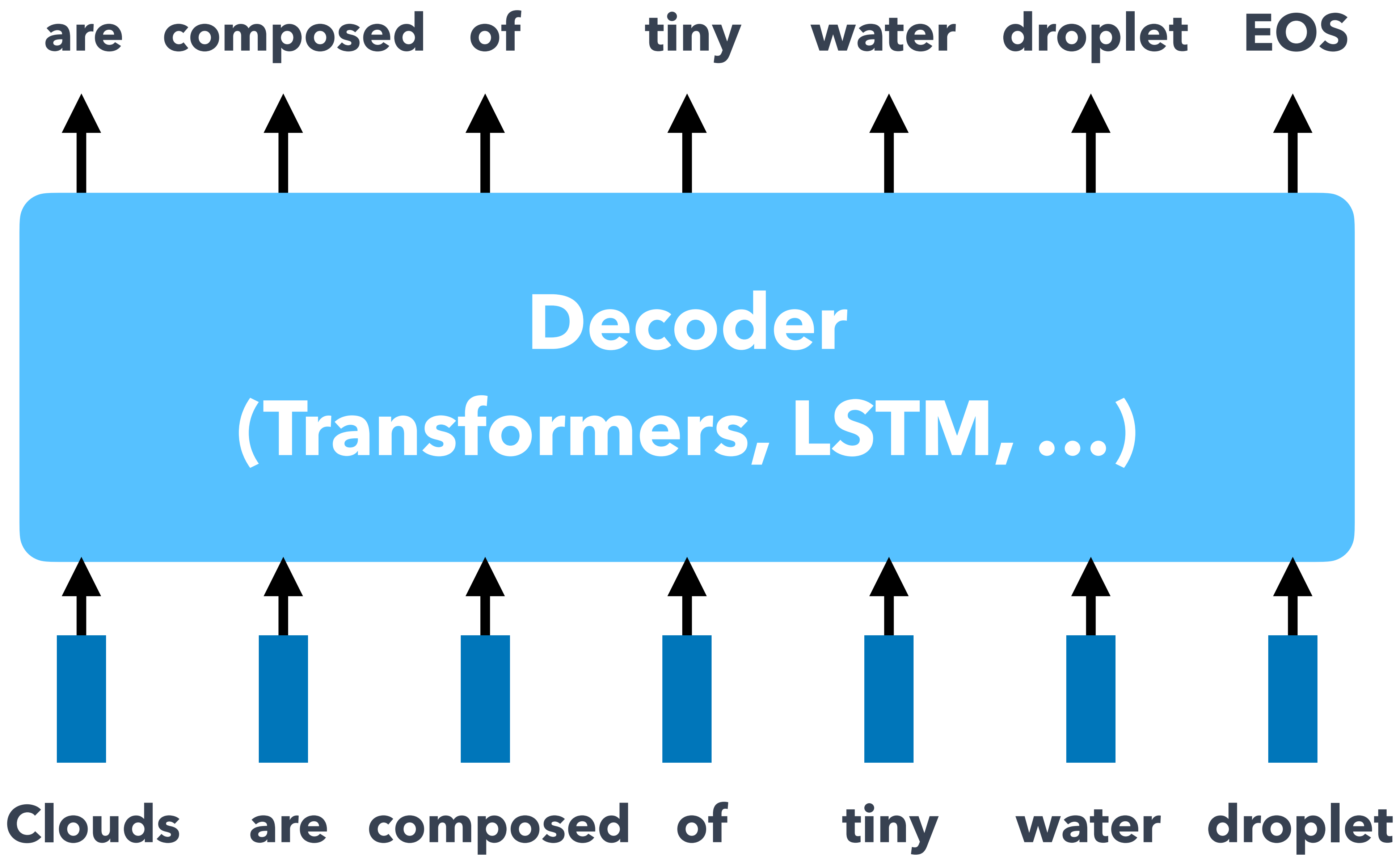
- Pre-training through **language modeling** [Dai and Le, 2015]
  - Model $P_\theta(w_t \mid w_{1:t-1})$, the probability distribution of the next word given previous contexts.
  - **There's lots of (English) data for this!** E.g., books, websites.
  - **Self-supervised** training of a neural network to perform the language modeling task with massive raw text data.
  - Save the network parameters to reuse later.

Why is this called self-supervised?
The labels come from the input data itself!



Supervised Fine-tuning for Specific Tasks

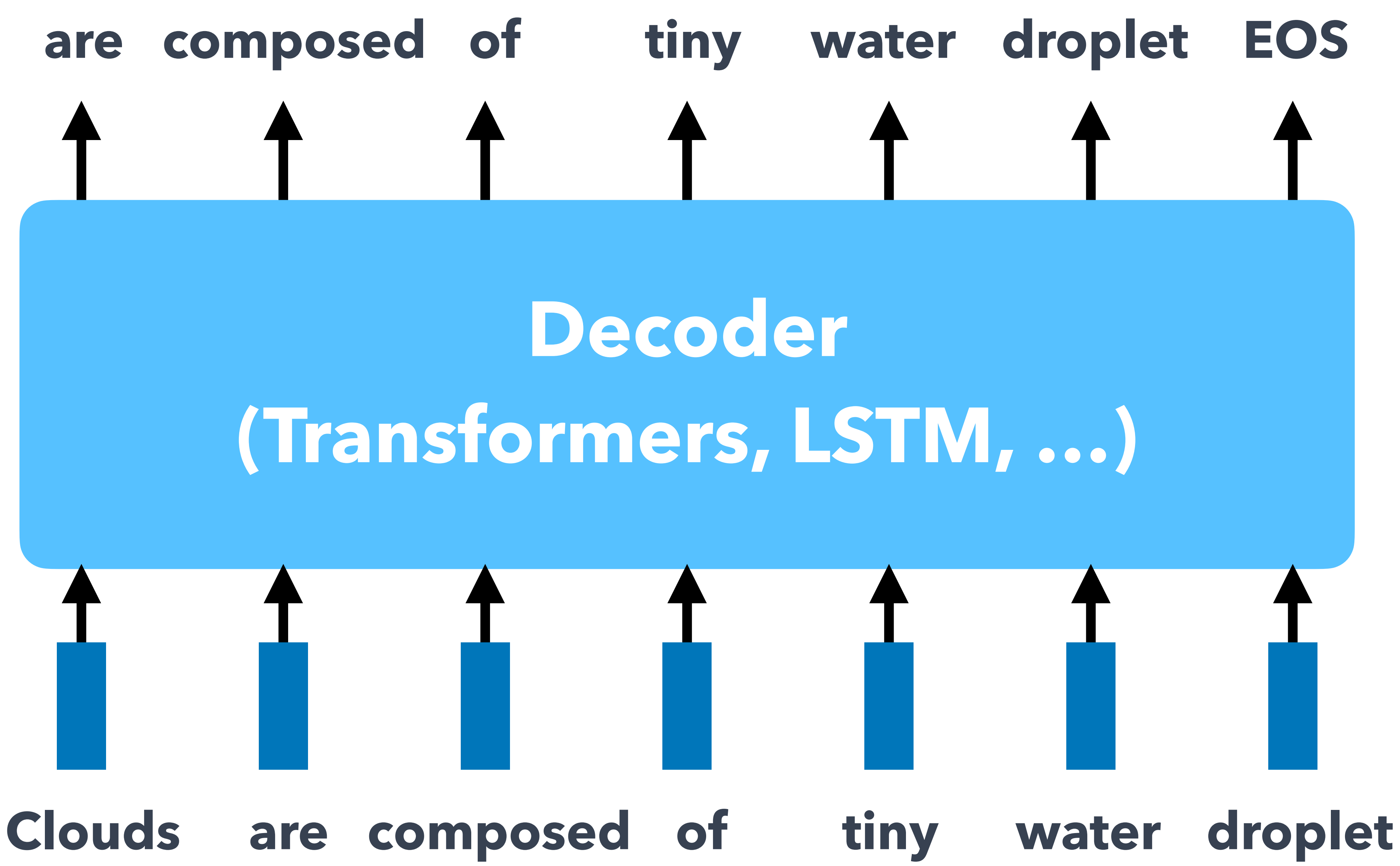
Step 1:
Pre-training



Abundant data; learn general language

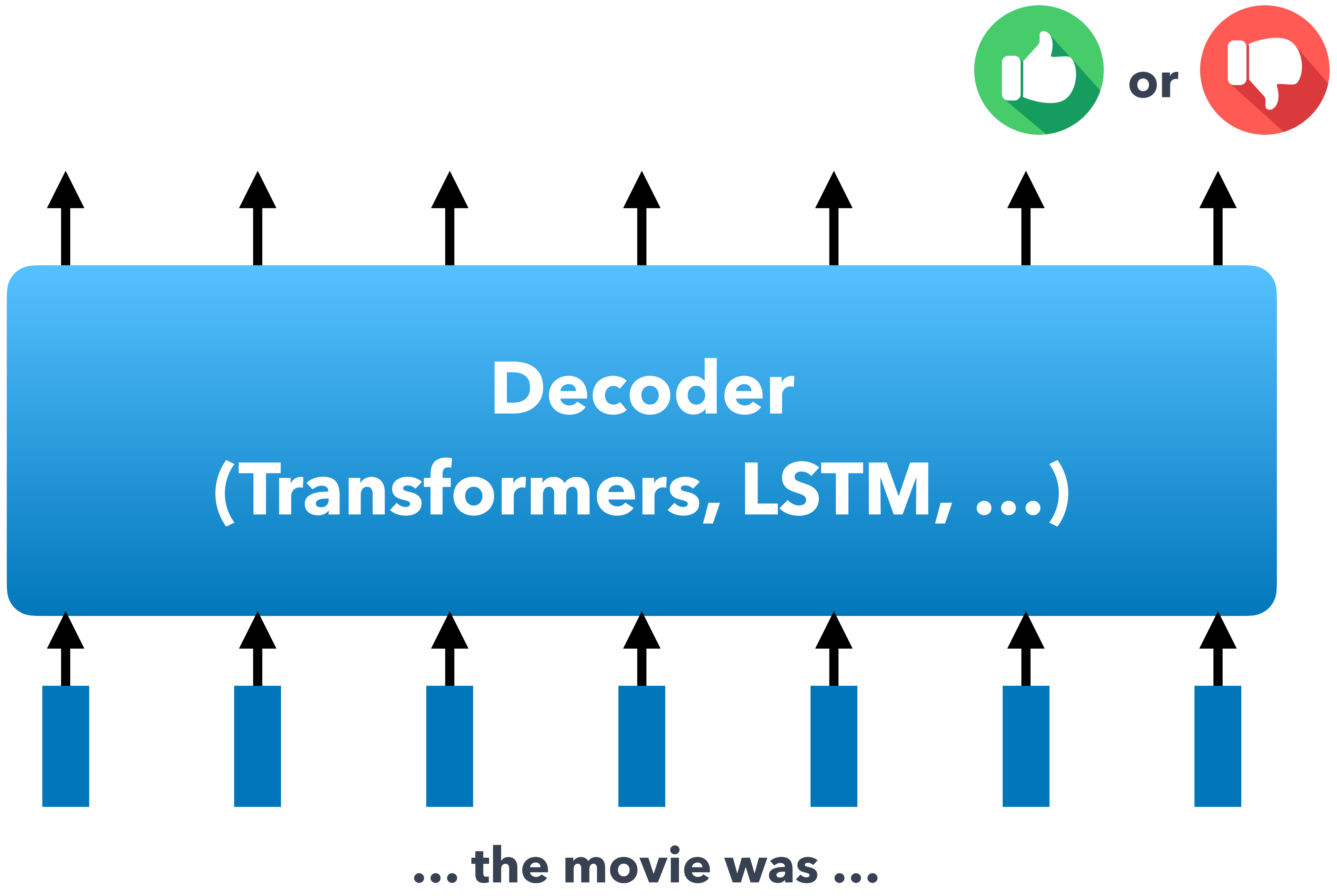
Supervised Fine-tuning for Specific Tasks

Step 1:
Pre-training



Abundant data; learn general language

Step 2:
Fine-tuning

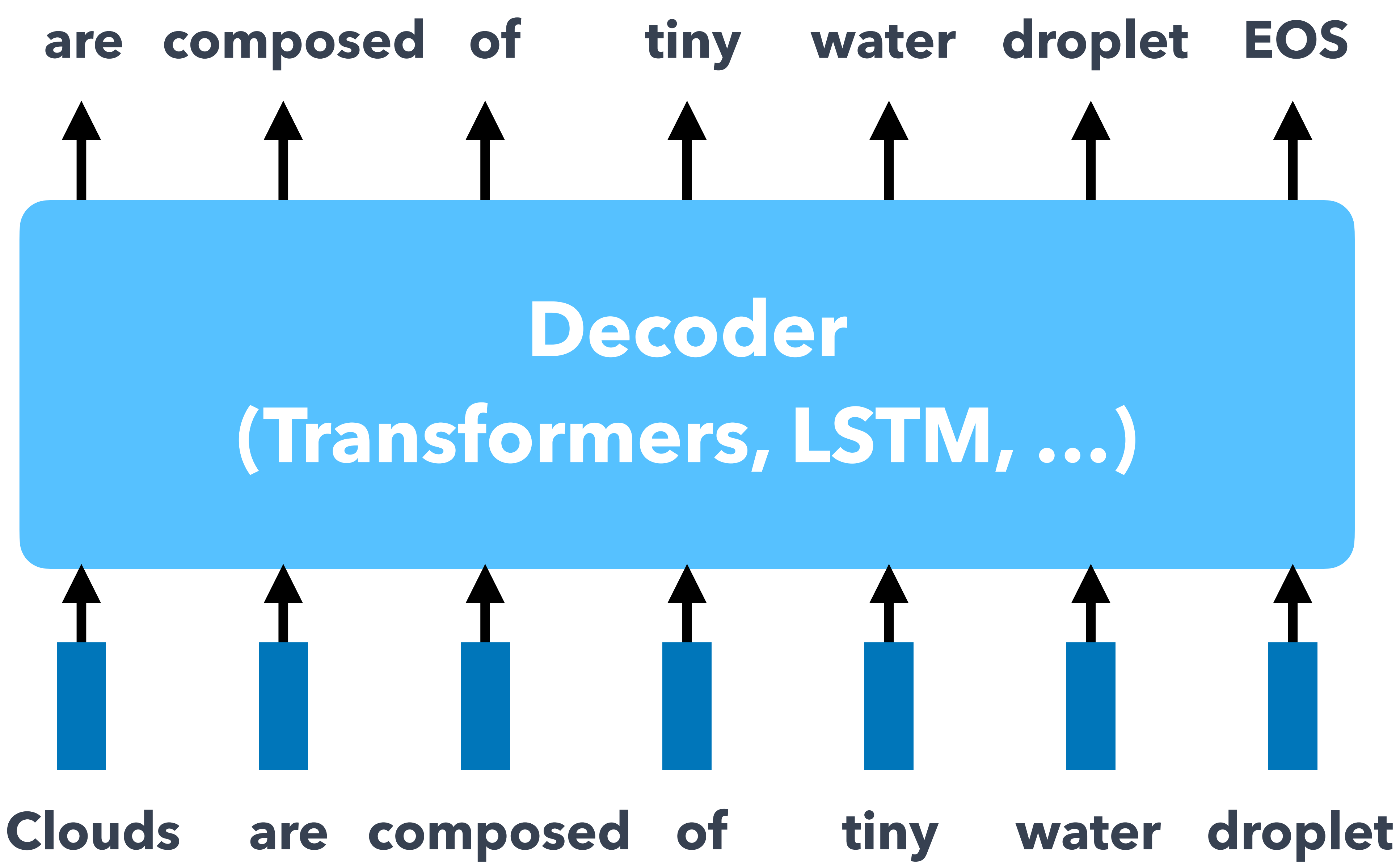


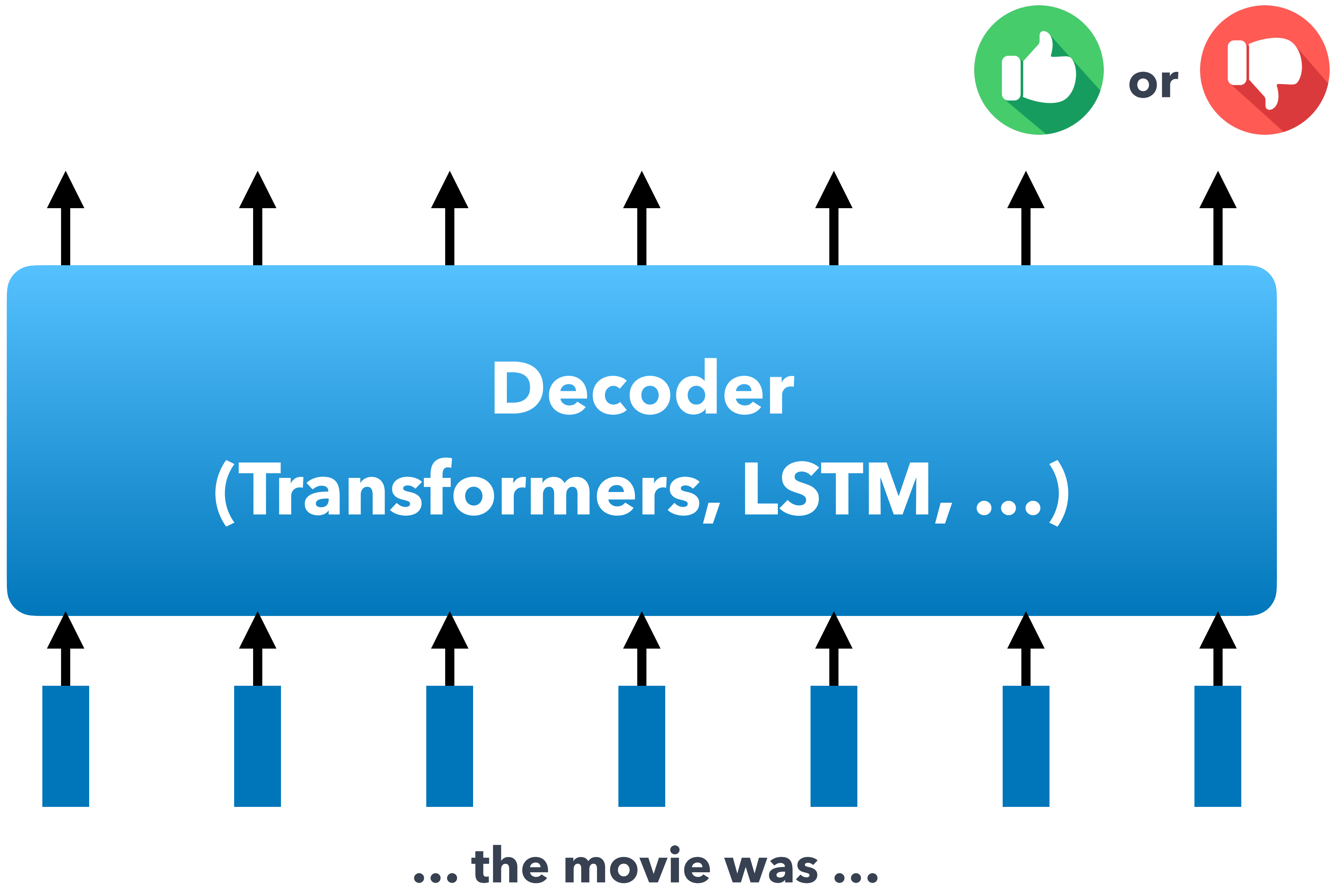
Limited data; adapt to the task

Supervised Fine-tuning for Specific Tasks

Step 1:
Pre-training

Step 2:
Fine-tuning

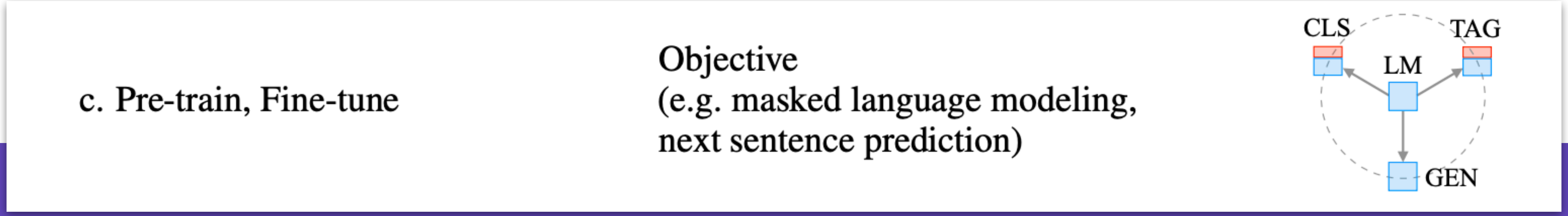




Abundant data; learn general language

Limited data; adapt to the task

Remember this is paradigm 3 from before



Why this works?

Lots of Information in Raw Texts

I went to Hawaii for snorkeling, hiking, and whale __________.

I walked across the street, checking for traffic ________ my shoulders.

I use __________ and fork to eat steak.

Ruth Bader Ginsburg was born in __________.

University of Washington is located at __________, Washington.

I was thinking about the sequence that goes 1, 1, 2, 3, 5, 8, 13, 21, ______.

Sugar is composed of carbon, hydrogen, and __________.

Lots of Information in Raw Texts

Verb

I went to Hawaii for snorkeling, hiking, and whale _watching_.

I walked across the street, checking for traffic ________ my shoulders.

I use __________ and fork to eat steak.

Ruth Bader Ginsburg was born in __________.

University of Washington is located at __________, Washington.

I was thinking about the sequence that goes 1, 1, 2, 3, 5, 8, 13, 21, ______.

Sugar is composed of carbon, hydrogen, and __________.

Lots of Information in Raw Texts

Verb I went to Hawaii for snorkeling, hiking, and whale _watching_.

Preposition I walked across the street, checking for traffic __over__ my shoulders.

I use __________ and fork to eat steak.

Ruth Bader Ginsburg was born in __________.

University of Washington is located at __________, Washington.

I was thinking about the sequence that goes 1, 1, 2, 3, 5, 8, 13, 21, ______.

Sugar is composed of carbon, hydrogen, and __________.

Lots of Information in Raw Texts

Verb

I went to Hawaii for snorkeling, hiking, and whale _watching_.

Preposition

I walked across the street, checking for traffic __over__ my shoulders.

Commonsense

I use ___knife___ and fork to eat steak.

Ruth Bader Ginsburg was born in __________.

University of Washington is located at __________, Washington.

I was thinking about the sequence that goes 1, 1, 2, 3, 5, 8, 13, 21, ______.

Sugar is composed of carbon, hydrogen, and __________.

Lots of Information in Raw Texts

Verb	I went to Hawaii for snorkeling, hiking, and whale _watching_.
Preposition	I walked across the street, checking for traffic __over__ my shoulders.
Commonsense	I use ___knife___ and fork to eat steak.
Time	Ruth Bader Ginsburg was born in ___1933___.
	University of Washington is located at __________, Washington.
	I was thinking about the sequence that goes 1, 1, 2, 3, 5, 8, 13, 21, ______.
	Sugar is composed of carbon, hydrogen, and __________.

Lots of Information in Raw Texts

Verb	I went to Hawaii for snorkeling, hiking, and whale _watching_.
Preposition	I walked across the street, checking for traffic __over__ my shoulders.
Commonsense	I use ___knife___ and fork to eat steak.
Time	Ruth Bader Ginsburg was born in ___1933___.
Location	University of Washington is located at __Seattle__, Washington.
	I was thinking about the sequence that goes 1, 1, 2, 3, 5, 8, 13, 21, _______.
	Sugar is composed of carbon, hydrogen, and __________.

Lots of Information in Raw Texts

Verb	I went to Hawaii for snorkeling, hiking, and whale _watching_.
Preposition	I walked across the street, checking for traffic __over__ my shoulders.
Commonsense	I use ___knife___ and fork to eat steak.
Time	Ruth Bader Ginsburg was born in ___1933___.
Location	University of Washington is located at __Seattle__, Washington.
Math	I was thinking about the sequence that goes 1, 1, 2, 3, 5, 8, 13, 21, __34__.
	Sugar is composed of carbon, hydrogen, and __________.

Lots of Information in Raw Texts

Verb	I went to Hawaii for snorkeling, hiking, and whale _watching_.
Preposition	I walked across the street, checking for traffic __over__ my shoulders.
Commonsense	I use ___knife___ and fork to eat steak.
Time	Ruth Bader Ginsburg was born in ___1933___.
Location	University of Washington is located at __Seattle__, Washington.
Math	I was thinking about the sequence that goes 1, 1, 2, 3, 5, 8, 13, 21, __34__.
Chemistry	Sugar is composed of carbon, hydrogen, and _oxygen__.

Lots of Information in Raw Texts

Verb	I went to Hawaii for snorkeling, hiking, and whale _watching_.
Preposition	I walked across the street, checking for traffic __over__ my shoulders.
Commonsense	I use ___knife___ and fork to eat steak.
Time	Ruth Bader Ginsburg was born in ___1933___.
Location	University of Washington is located at __Seattle__, Washington.
Math	I was thinking about the sequence that goes 1, 1, 2, 3, 5, 8, 13, 21, __34__.
Chemistry	Sugar is composed of carbon, hydrogen, and _oxygen__.

...

The Stochastic Gradient Descent Angle

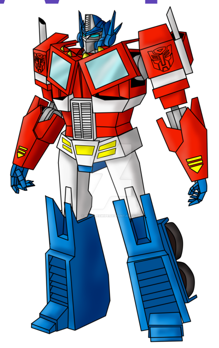
Why should pre-training and then fine-tuning help?

- Providing parameters $\hat{\theta}$ by approximating the pre-training loss, $\min_{\theta} \mathscr{L}_{\text{pretrain}}(\theta)$.
- Then, starting with parameters $\hat{\theta}$, approximating fine-tuning loss, $\min_{\theta} \mathscr{L}_{\text{finetune}}(\theta)$.
- **Stochastic gradient descent sticks (relatively) close to $\hat{\theta}$ during fine-tuning.**
  - So, maybe the fine-tuning local minima near $\hat{\theta}$ tend to generalize well!
  - And/or, maybe the gradients of fine-tuning loss near $\hat{\theta}$ propagate nicely!

Advantages of Pre-training & Fine-tuning

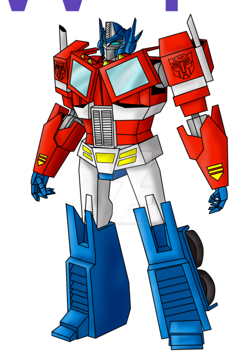
- **Leveraging rich underlying information** from abundant raw texts.
- **Reducing the reliance of task-specific labeled data** that is difficult or costly to obtain.
- **Initializing model parameters** for more **generalizable** NLP applications.
- **Saving training cost** by providing a reusable model checkpoints.
- **Providing robust representation** of language contexts.

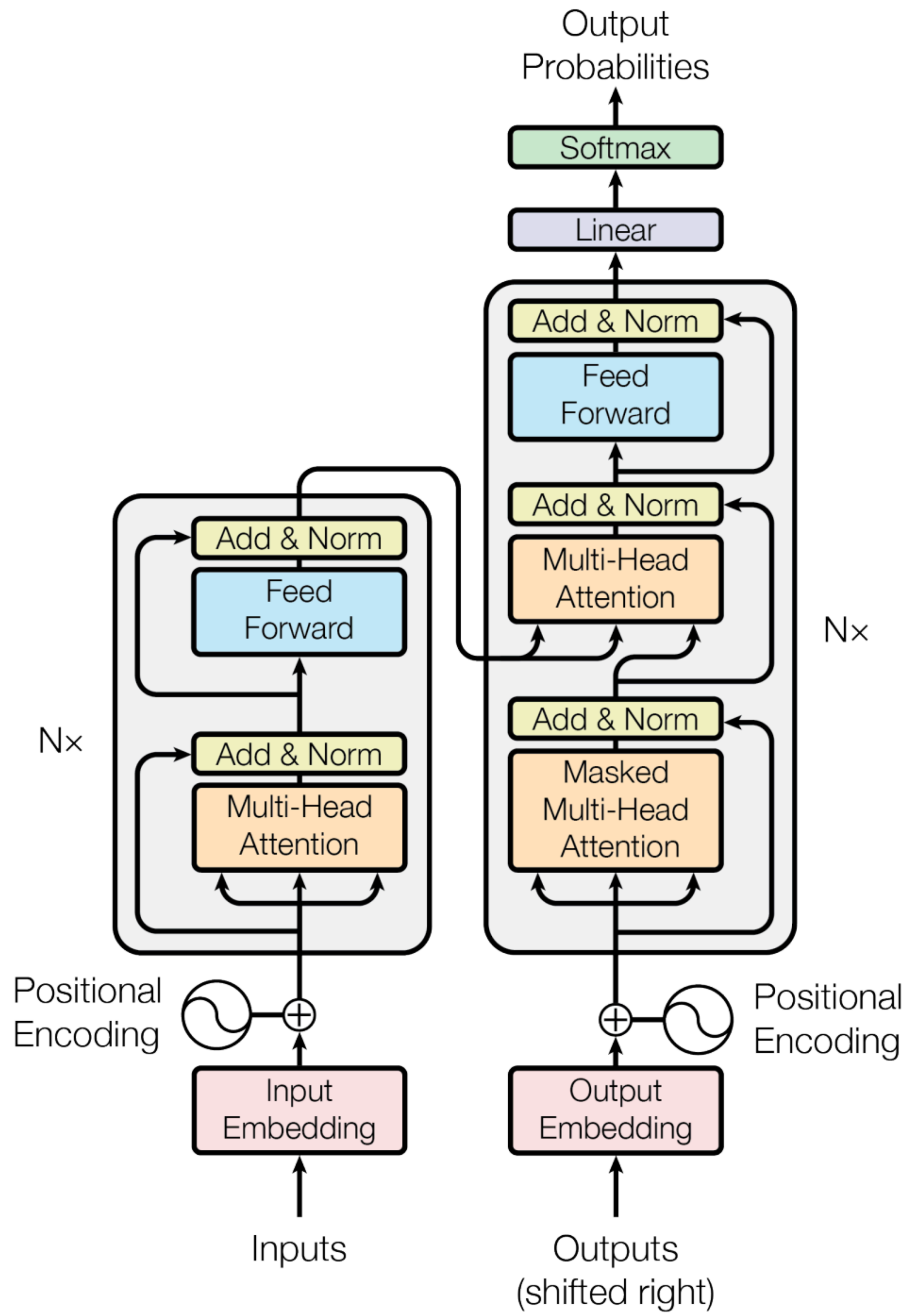
Solving Shallow Networks Problem in NLP: **Enter Transformers**



Attention is all You Need. 2017.

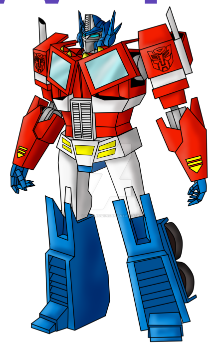
Solving Shallow Networks Problem in NLP: **Enter Transformers**

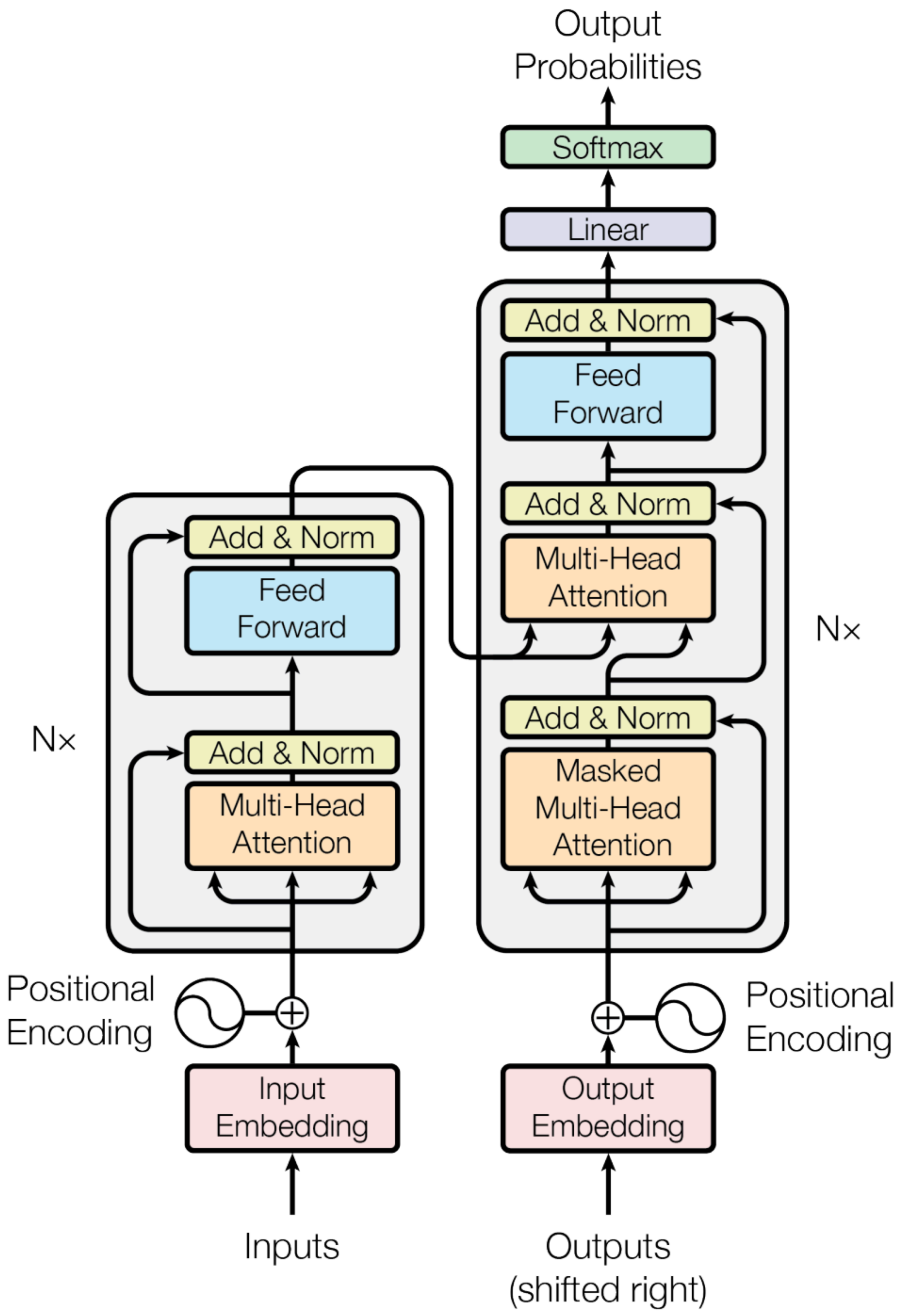


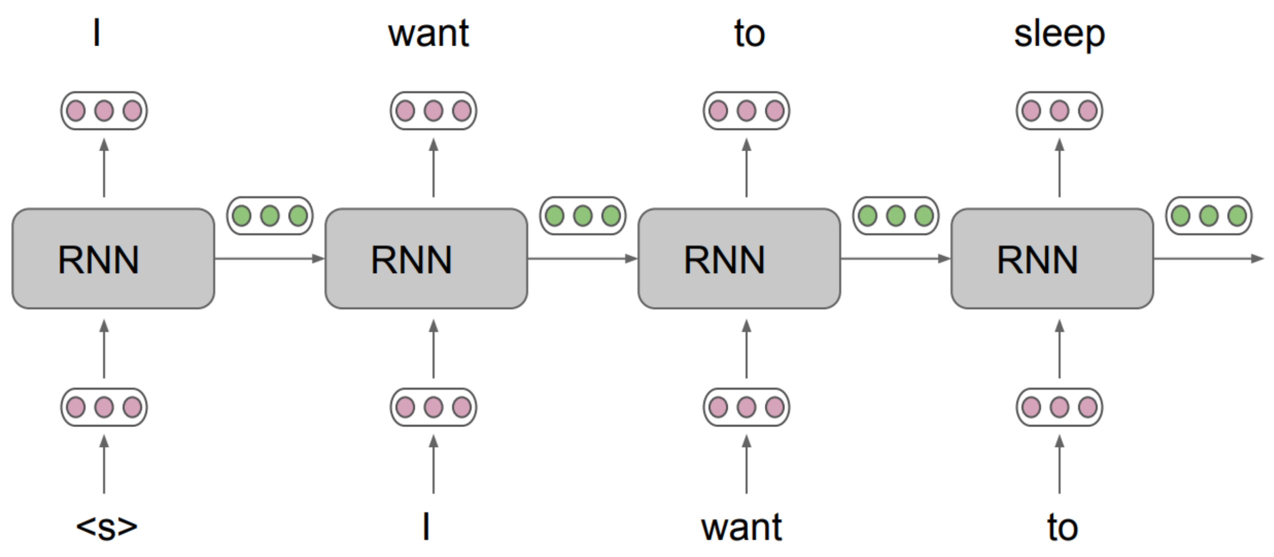


Attention is all You Need. 2017.

Solving Shallow Networks Problem in NLP: **Enter Transformers**

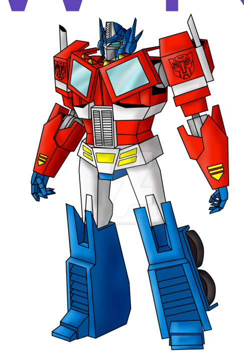




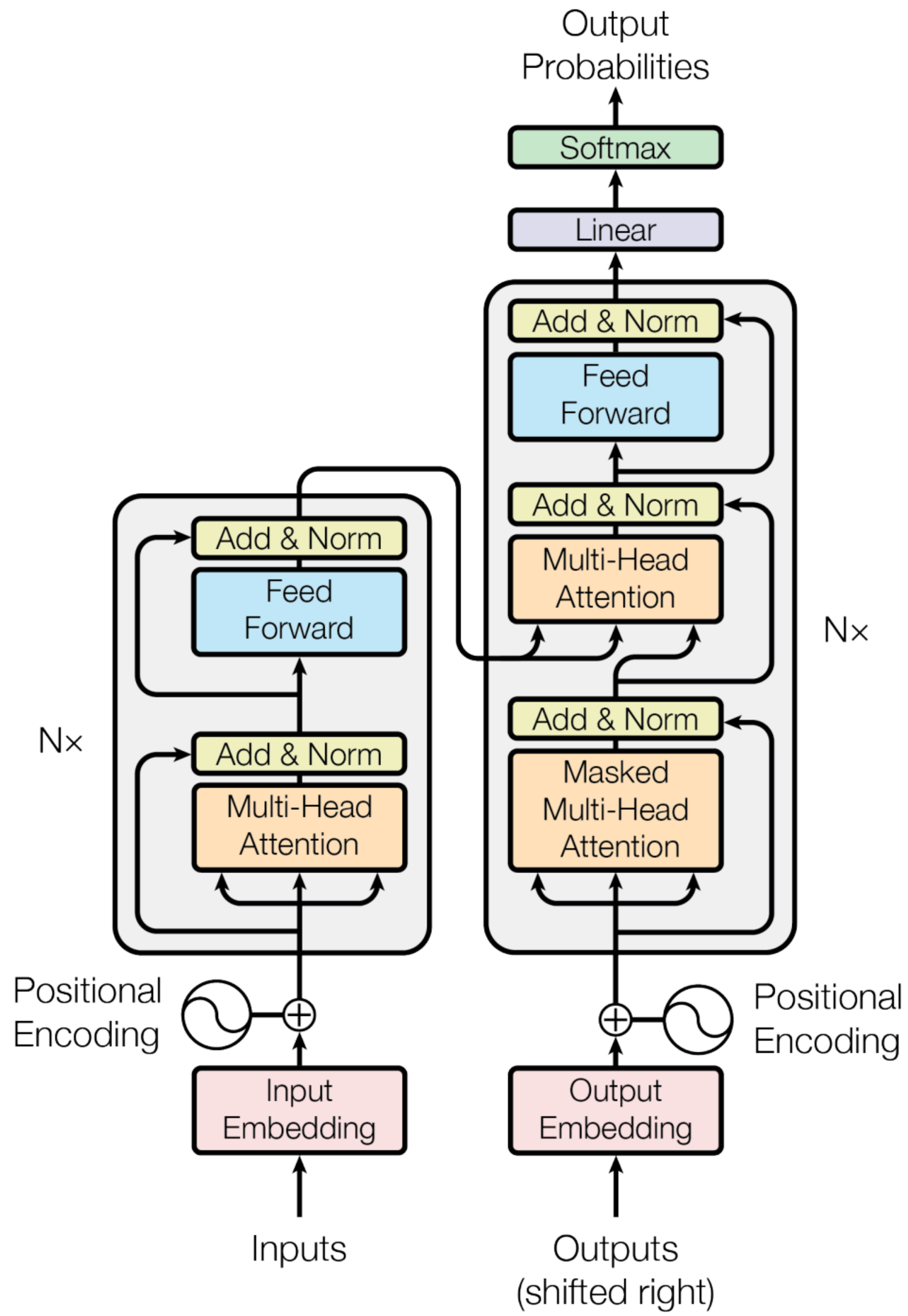


Attention is all You Need. 2017.

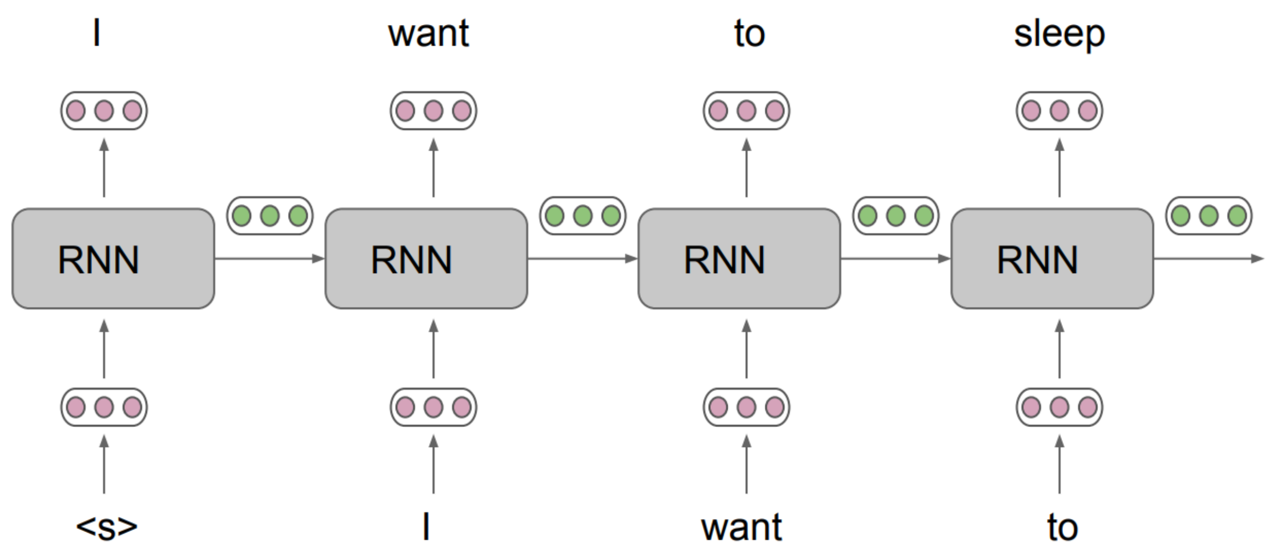
Solving Shallow Networks Problem in NLP: **Enter Transformers**



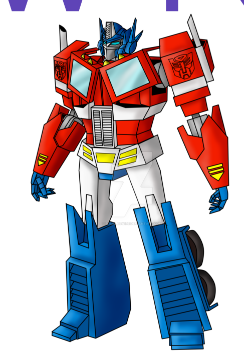
Transformers managed to avoid the two major problems that made Recurrent Neural Networks hard to scale on larger compute and depths:



Attention is all You Need. 2017.

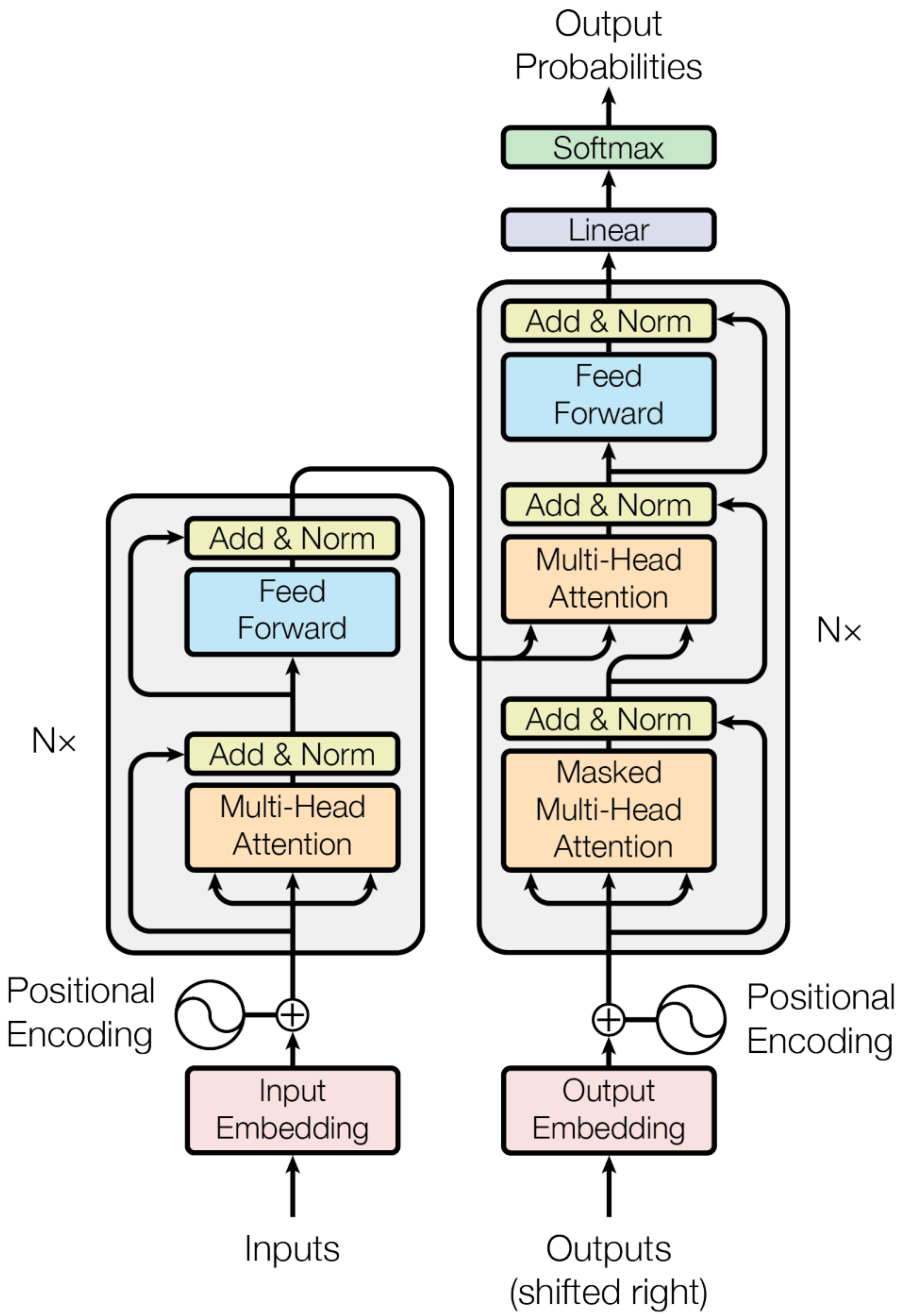


Solving Shallow Networks Problem in NLP: **Enter Transformers**

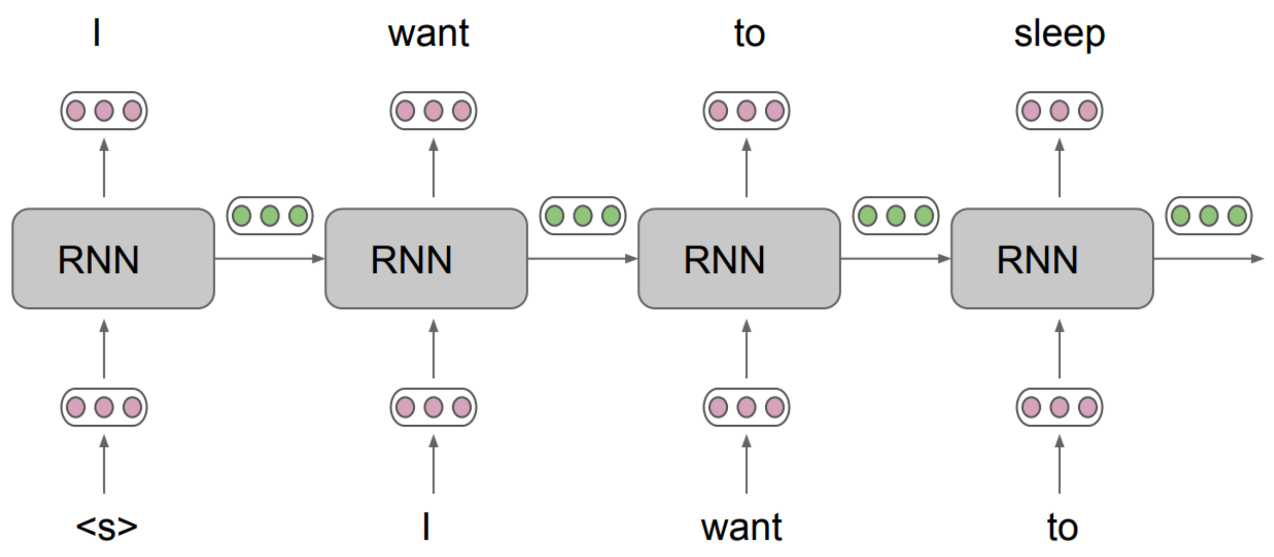


Transformers managed to avoid the two major problems that made Recurrent Neural Networks hard to scale on larger compute and depths:

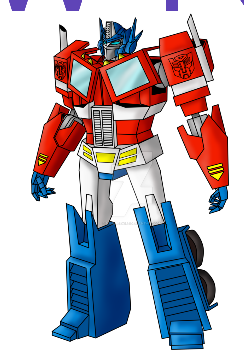
- **<u>Highly Parallelizable During Training</u>**: Need not wait for the computation at the previous time step to complete to execute the next step



Attention is all You Need. 2017.



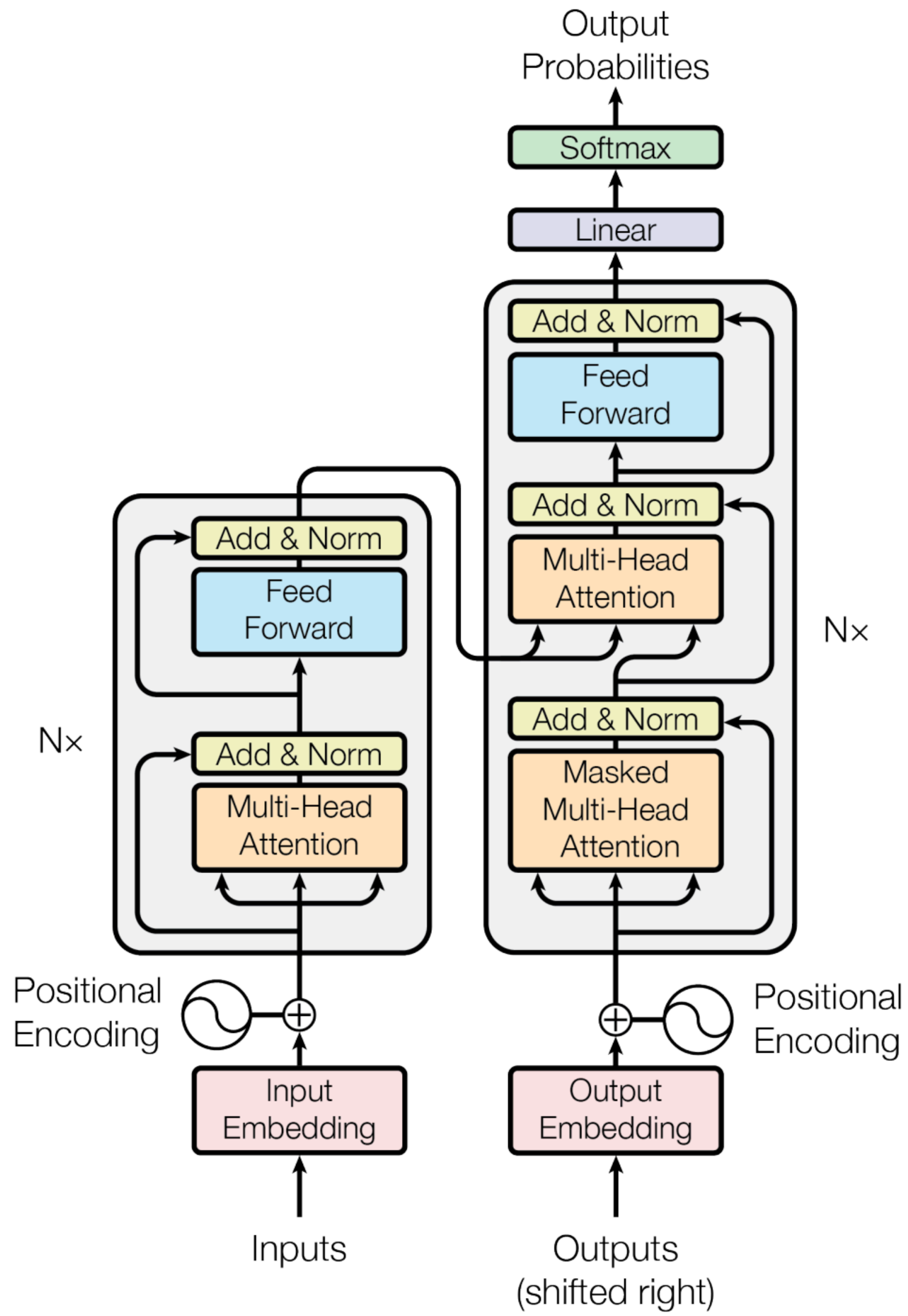
Solving Shallow Networks Problem in NLP: **Enter Transformers**



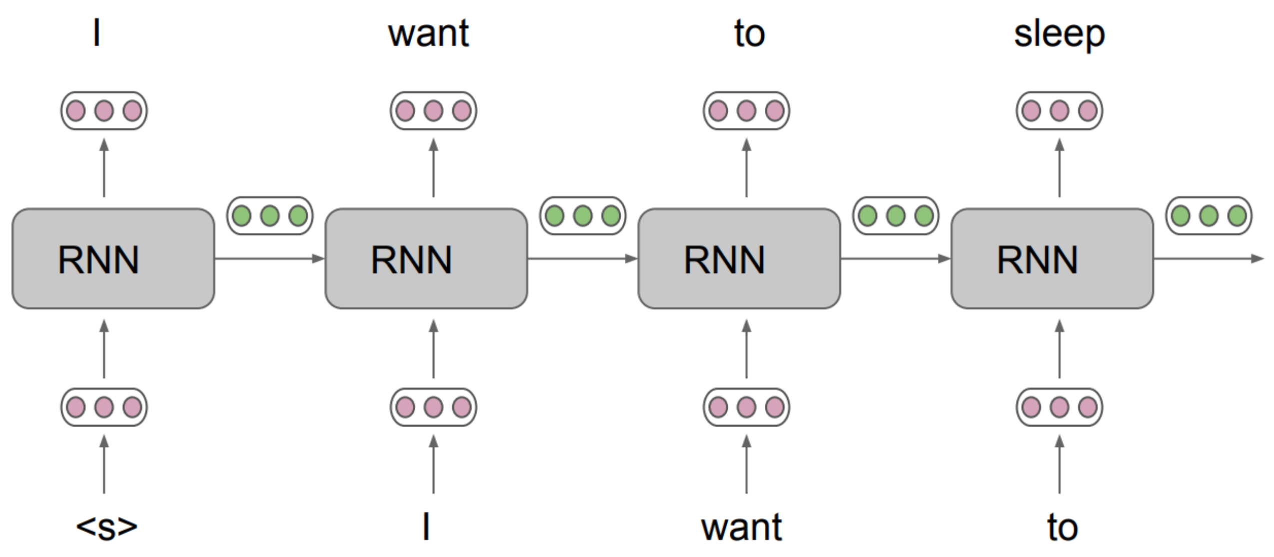
Transformers managed to avoid the two major problems that made Recurrent Neural Networks hard to scale on larger compute and depths:

- **<u>Highly Parallelizable During Training</u>**: Need not wait for the computation at the previous time step to complete to execute the next step

- **<u>Avoids Training Complications like Vanishing Gradients:</u>** Unlike RNNs, which have a fixed state that gets updated repeatedly, transformers have dynamic memory, which also avoids issues such as vanishing gradients



Attention is all You Need. 2017.



Lecture Outline

1. Motivating Pre-training, aka Self-supervised Learning
2. Pre-training Architectures and Training Objectives
    1. Encoders
    2. Encoder-Decoders
    3. Decoder

3 Pre-training Paradigms/Architectures

Encoder

- E.g., BERT, RoBERTa, DeBERTa, …
- **Autoencoder** model
- **Masked** language modeling

Encoder-Decoder

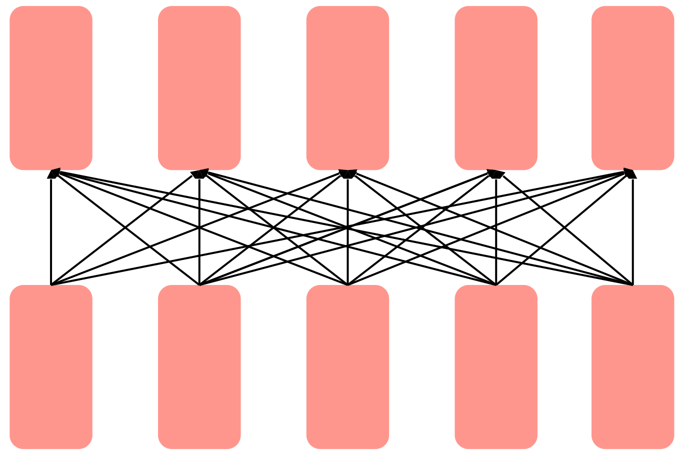
- E.g., T5, BART, …
- **seq2seq** model

Decoder

- E.g., GPT, GPT2, GPT3, …
- **Autoregressive** model
- **Left-to-right** language modeling

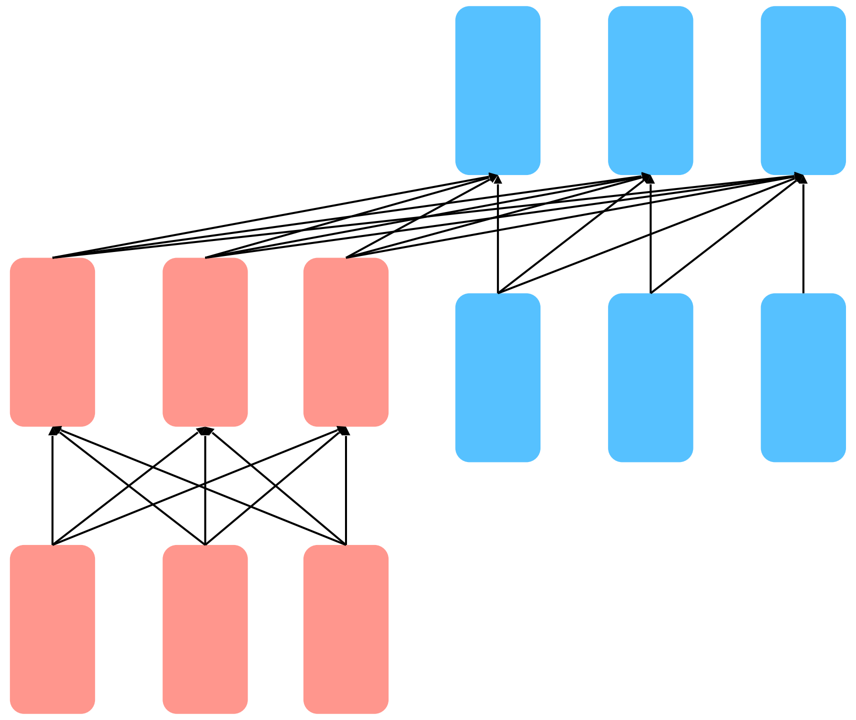
3 Pre-training Paradigms/Architectures

Encoder



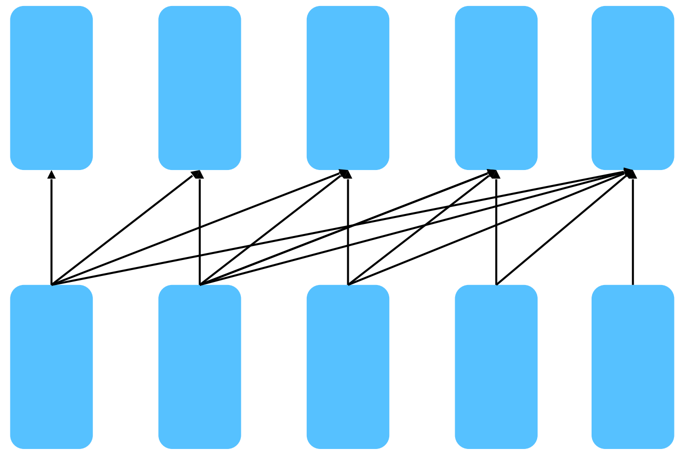
- Bidirectional; can condition on the future context

Encoder-Decoder



- Map two sequences of different length together

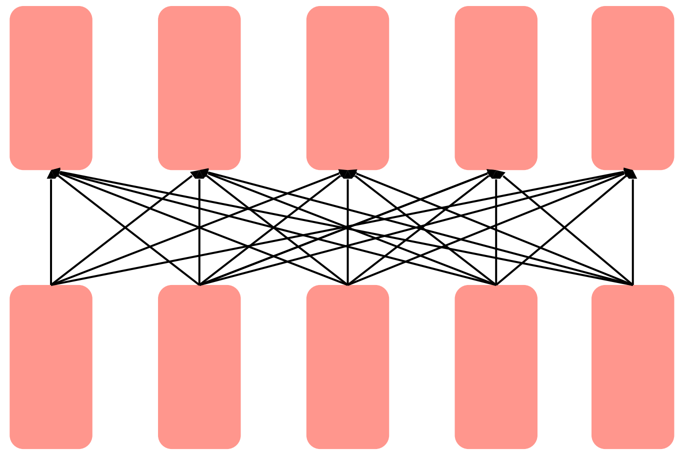
Decoder



- Language modeling; can only condition on the past context

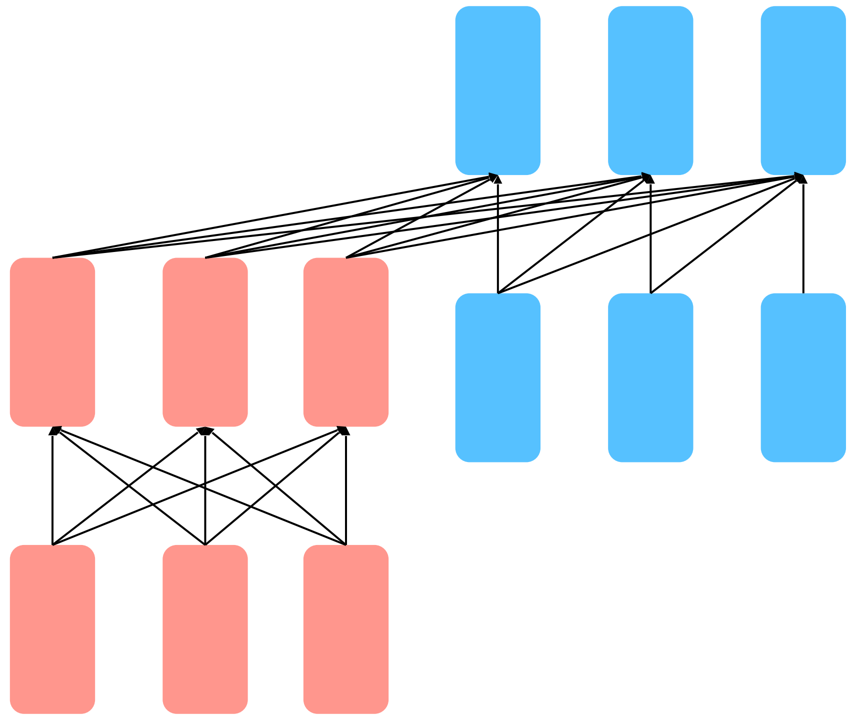
3 Pre-training Paradigms/Architectures

Encoder



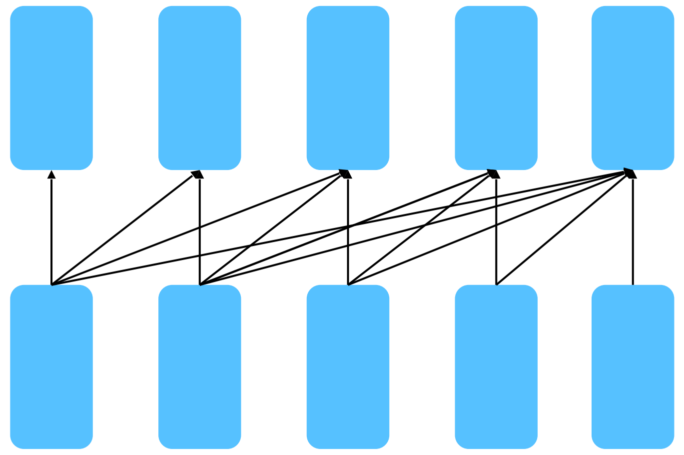
- Bidirectional; can condition on the future context

Encoder-Decoder



- Map two sequences of different length together

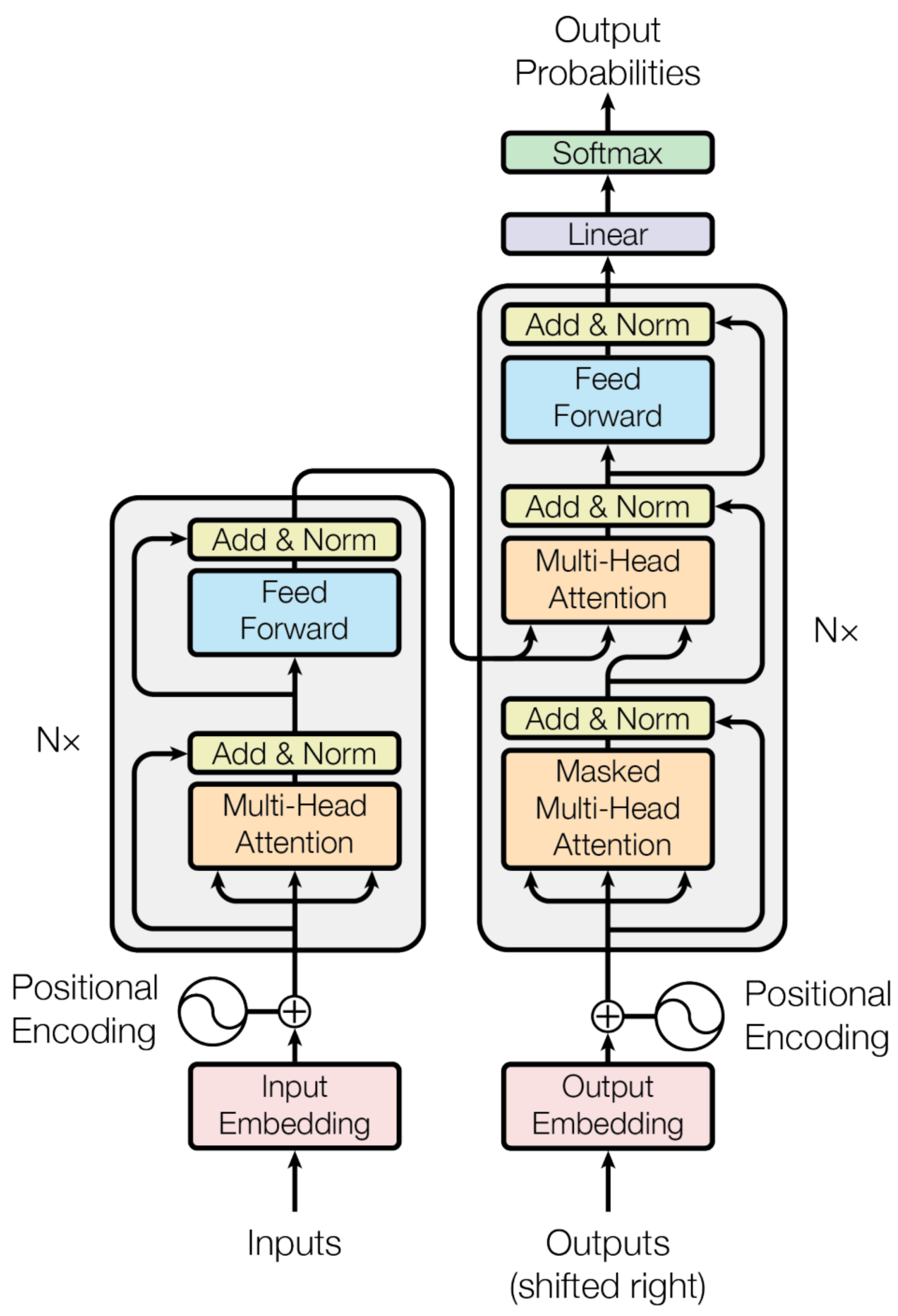
Decoder



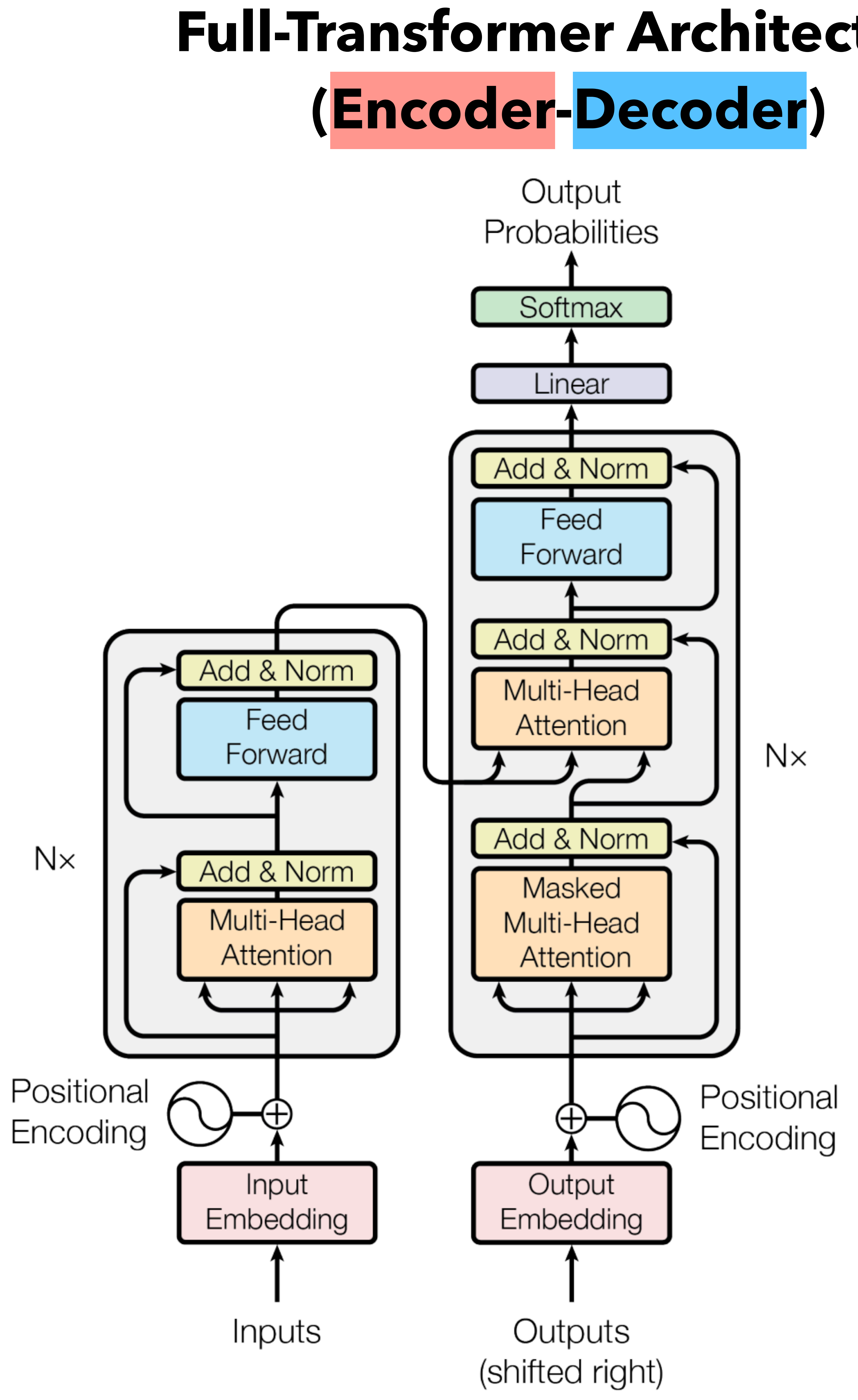
- Language modeling; can only condition on the past context

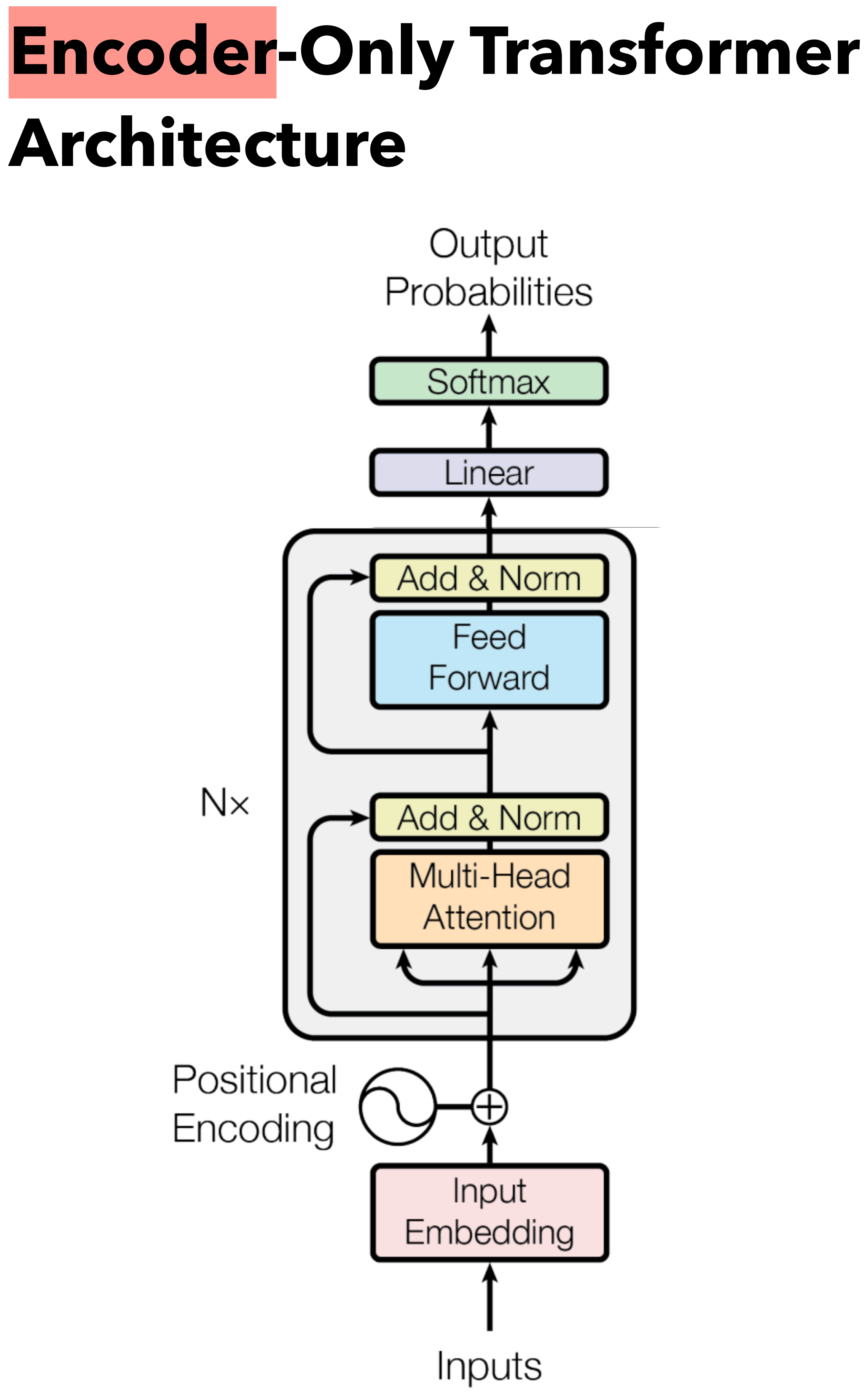
Encoder: Architecture

Full-Transformer Architecture
(Encoder-Decoder)



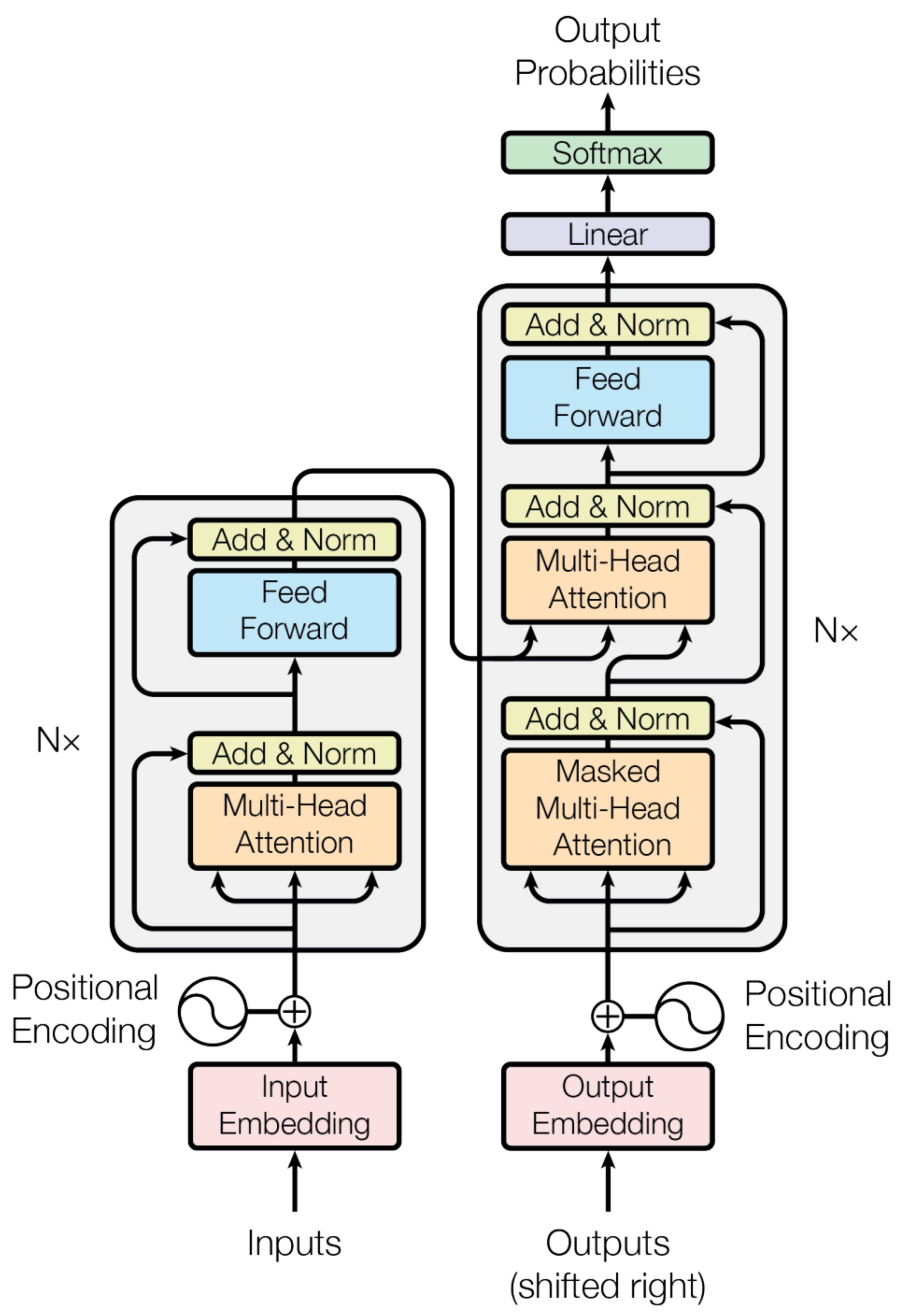
Encoder: Architecture



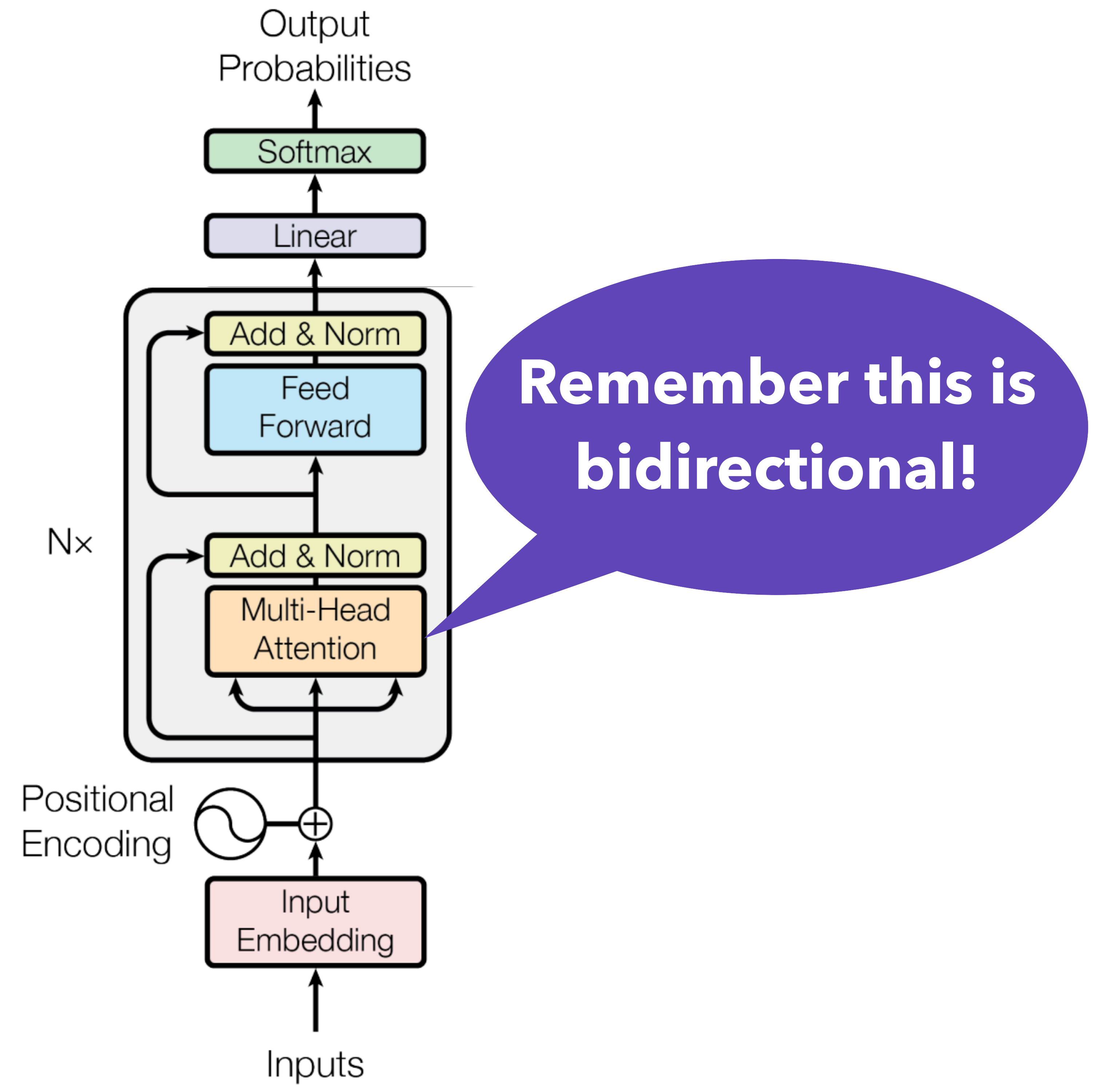


Encoder: Architecture

Full-Transformer Architecture (Encoder-Decoder)

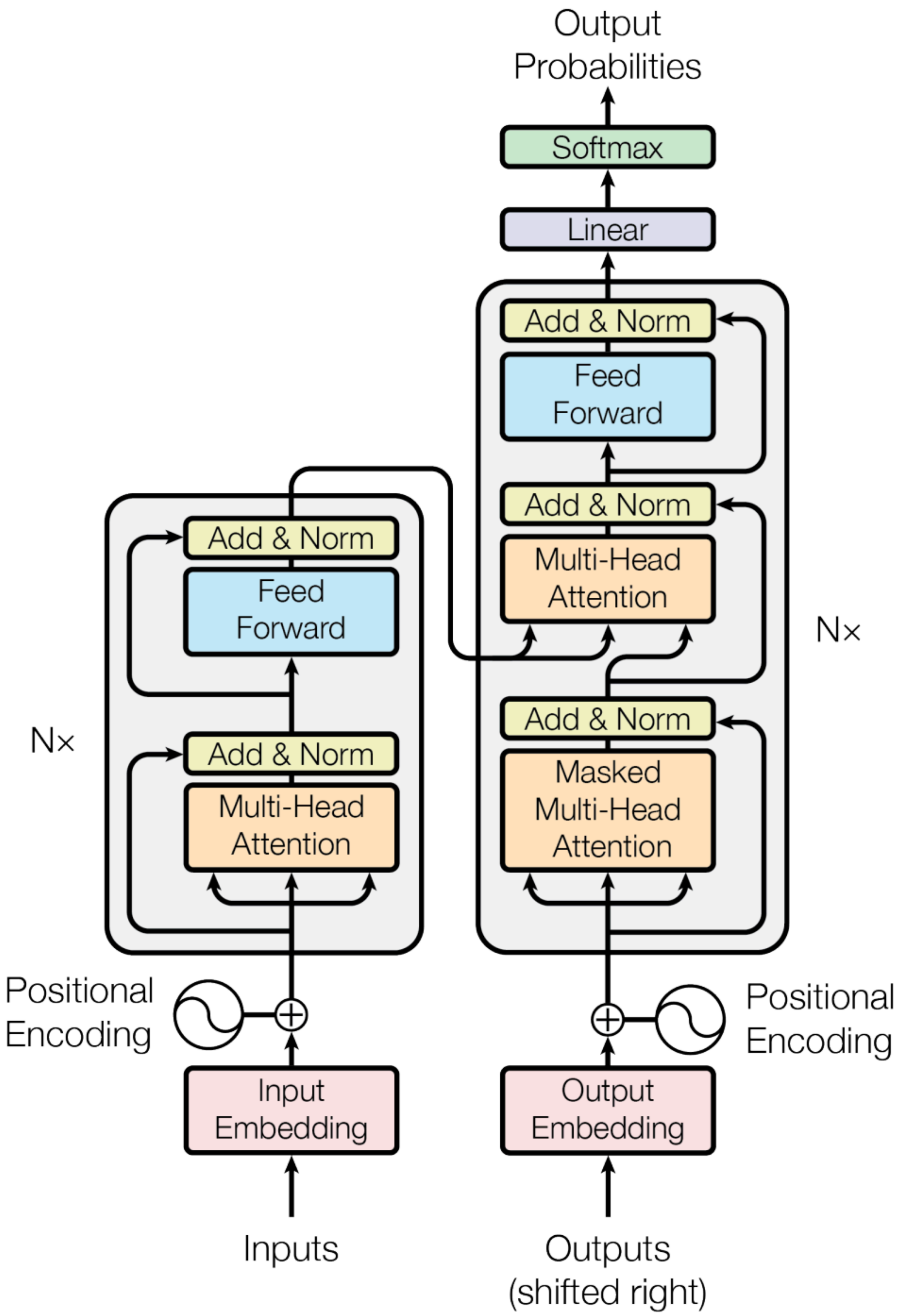


Encoder-Only Transformer Architecture

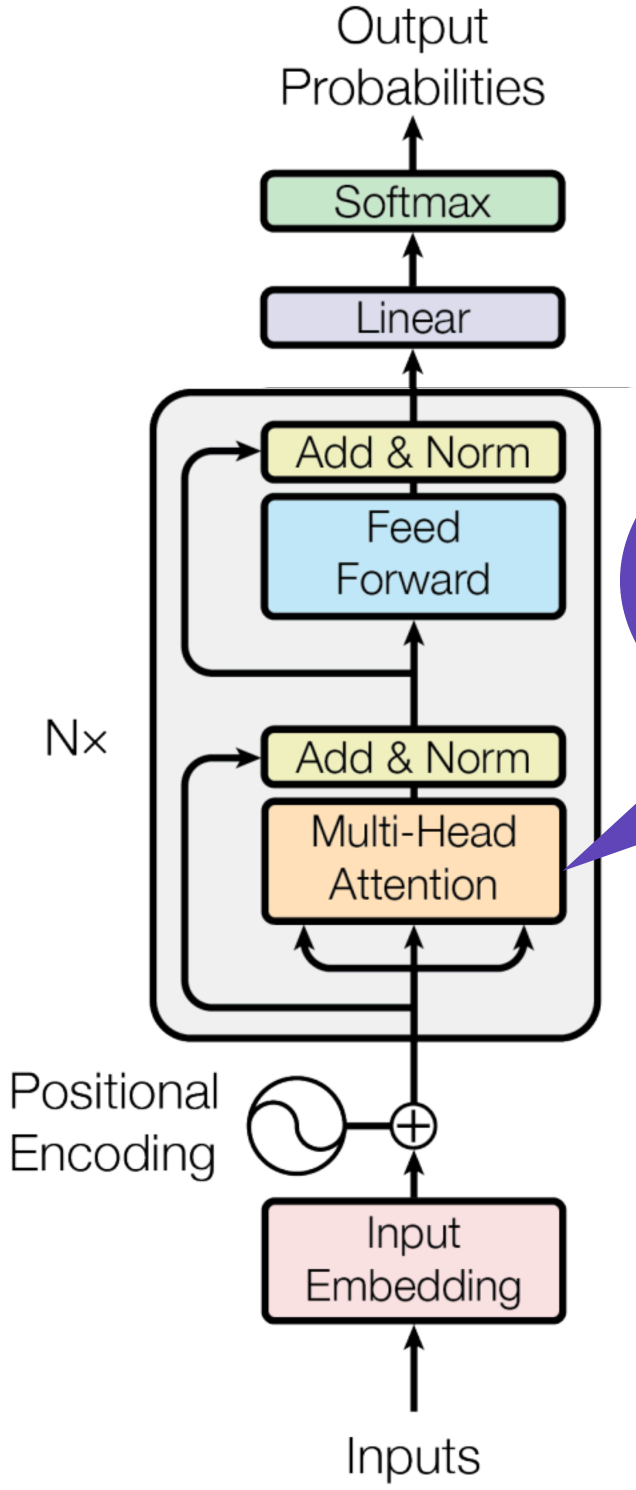


Encoder: Architecture

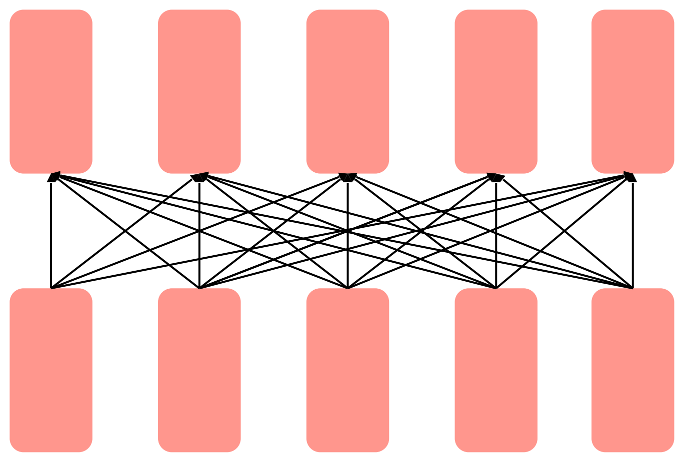
Full-Transformer Architecture (Encoder-Decoder)



Encoder-Only Transformer Architecture



Remember this is bidirectional!

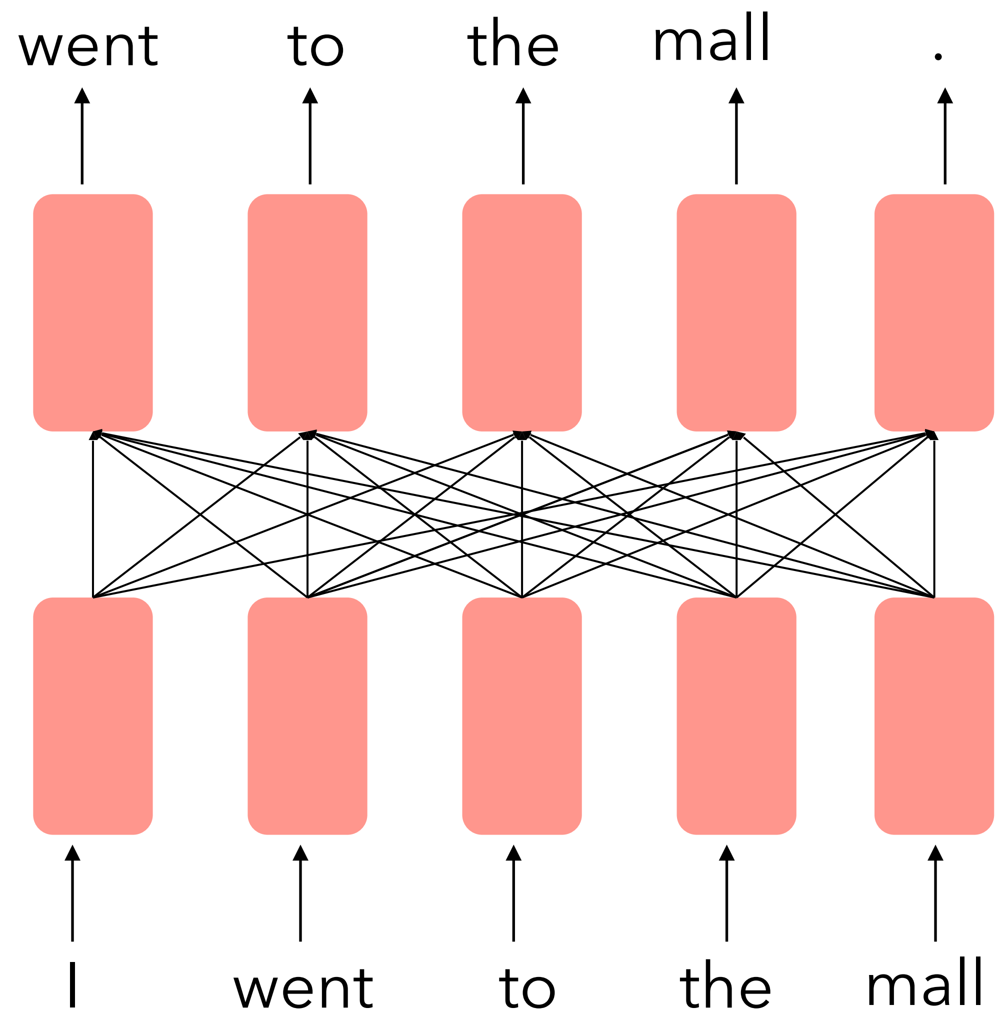


Encoder: Training Objective

- So far, we've looked at language modeling for pre-training.
- Language Model Pretraining is problematic for encoders
- Why?
  - **Encoders get bidirectional contexts**
  - The model can cheat by just looking at the next token when predicting it without actually learning anything about language!

Encoder: Training Objective

- So far, we've looked at language modeling for pre-training.
- Language Model Pretraining is problematic for encoders
- Why?
  - **Encoders get bidirectional contexts**
  - The model can cheat by just looking at the next token when predicting it without actually learning anything about language!



Encoder: Training Objective

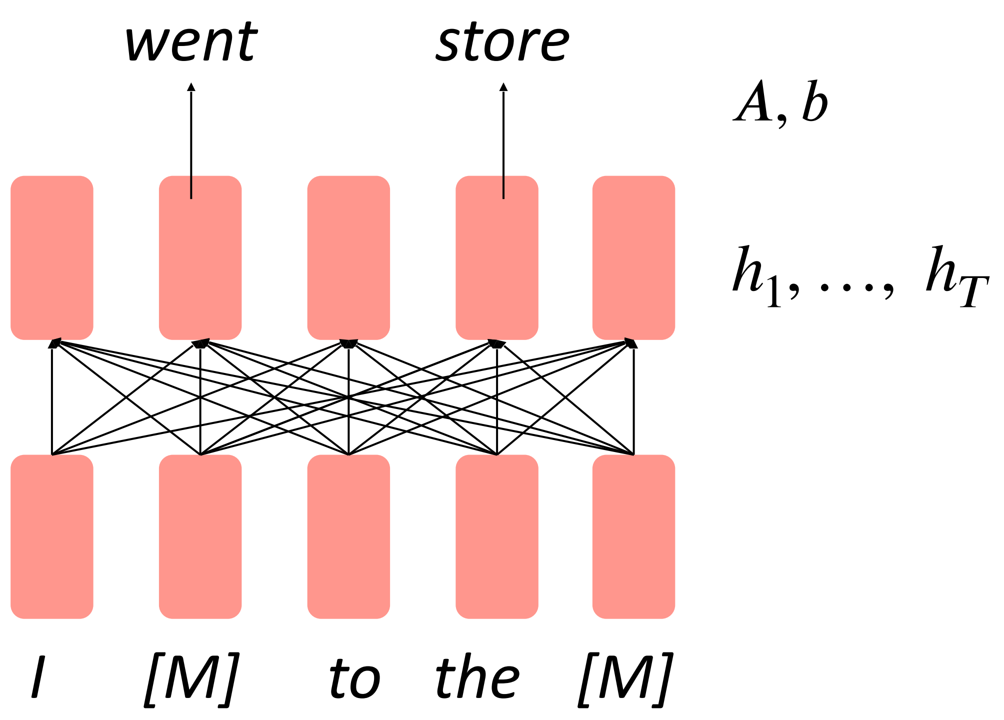
[Devlin et al., 2018]

- How to encode information from both **bidirectional** contexts?
- General Idea: **text reconstruction!**
  - Your time is [MASK], so don't [MASK] it living someone else's life. Don't be trapped by [MASK], which is [MASK] with the results of other [MASK]'s thinking. – [MASK] Jobs

Encoder: Training Objective

[Devlin et al., 2018]

- How to encode information from both **bidirectional** contexts?
- General Idea: **text reconstruction!**
  - Your time is [MASK], so don't [MASK] it living someone else's life. Don't be trapped by [MASK], which is [MASK] with the results of other [MASK]'s thinking. – [MASK] Jobs



$$h_1, \ldots, h_T = \text{Encoder}(w_1, \ldots, w_T)$$

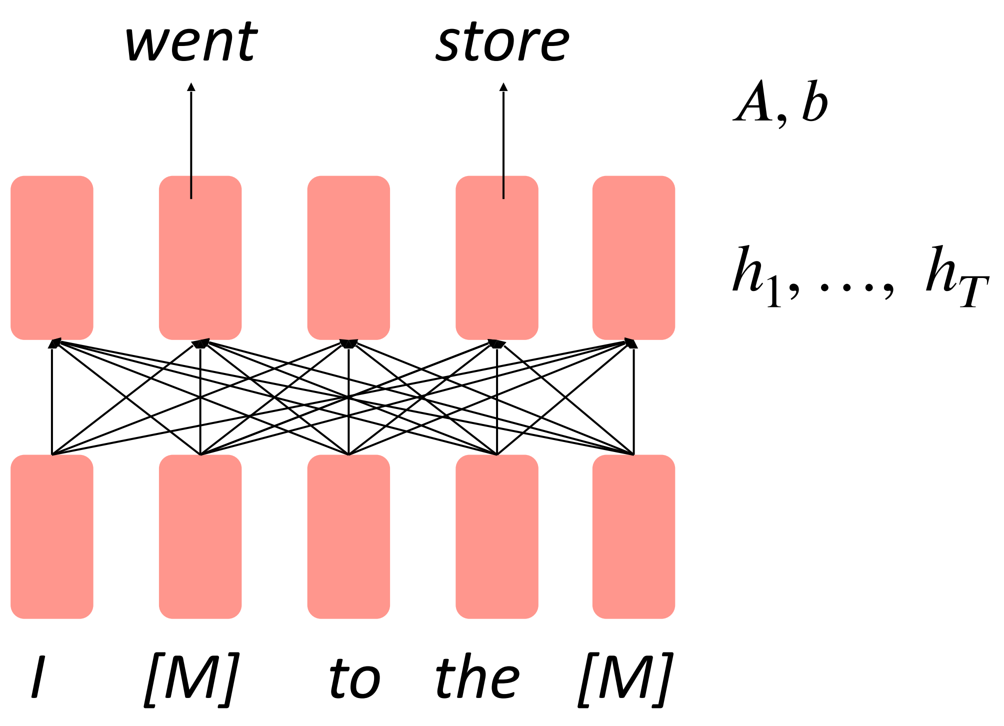
$$y_i \sim Aw_i + b$$

Only add loss terms from the masked tokens. If $\tilde{x}$ is the masked version of x, we're learning $p_\theta(x \mid \tilde{x})$. Called **Masked Language model (MLM)**.

Encoder: Training Objective

[Devlin et al., 2018]

- How to encode information from both **bidirectional** contexts?
- General Idea: **text reconstruction!**
  - Your time is limited so don't [MASK] it living someone else's life. Don't be trapped by [MASK], which is [MASK] with the results of other [MASK]'s thinking. – [MASK] Jobs



$$h_1, \ldots, h_T = \text{Encoder}(w_1, \ldots, w_T)$$

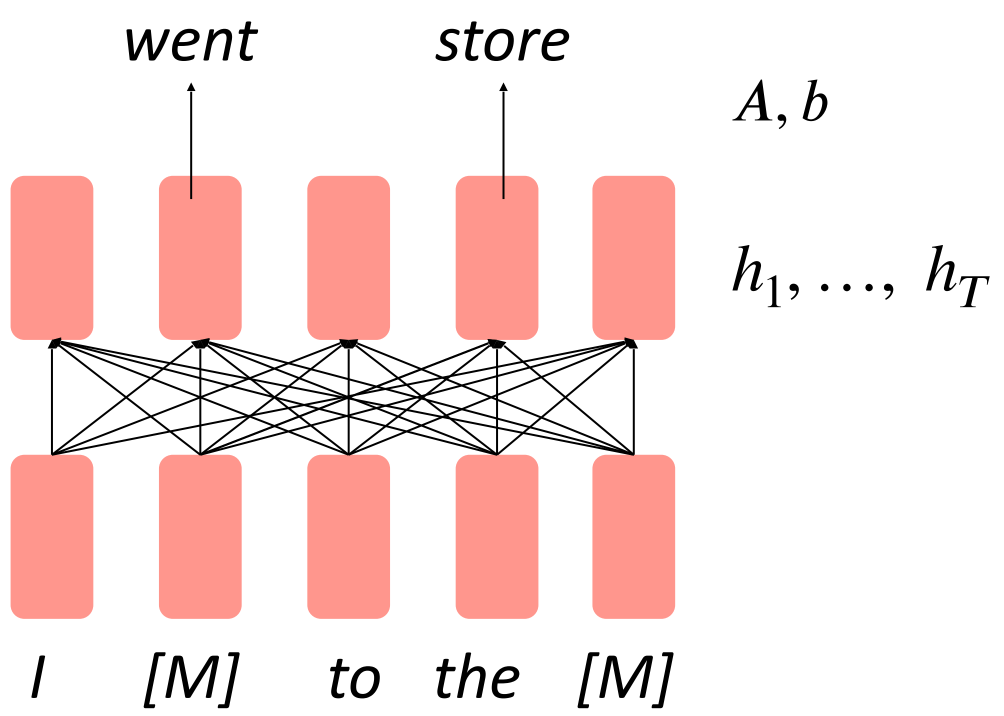
$$y_i \sim Aw_i + b$$

Only add loss terms from the masked tokens. If $\tilde{x}$ is the masked version of x, we're learning $p_\theta(x \mid \tilde{x})$. Called **Masked Language model (MLM)**.

Encoder: Training Objective

[Devlin et al., 2018]

- How to encode information from both **bidirectional** contexts?
- General Idea: **text reconstruction!**
  - Your time is limited so don't waste it living someone else's life. Don't be trapped by [MASK], which is [MASK] with the results of other [MASK]'s thinking. – [MASK] Jobs



$$h_1, \ldots, h_T = \text{Encoder}(w_1, \ldots, w_T)$$

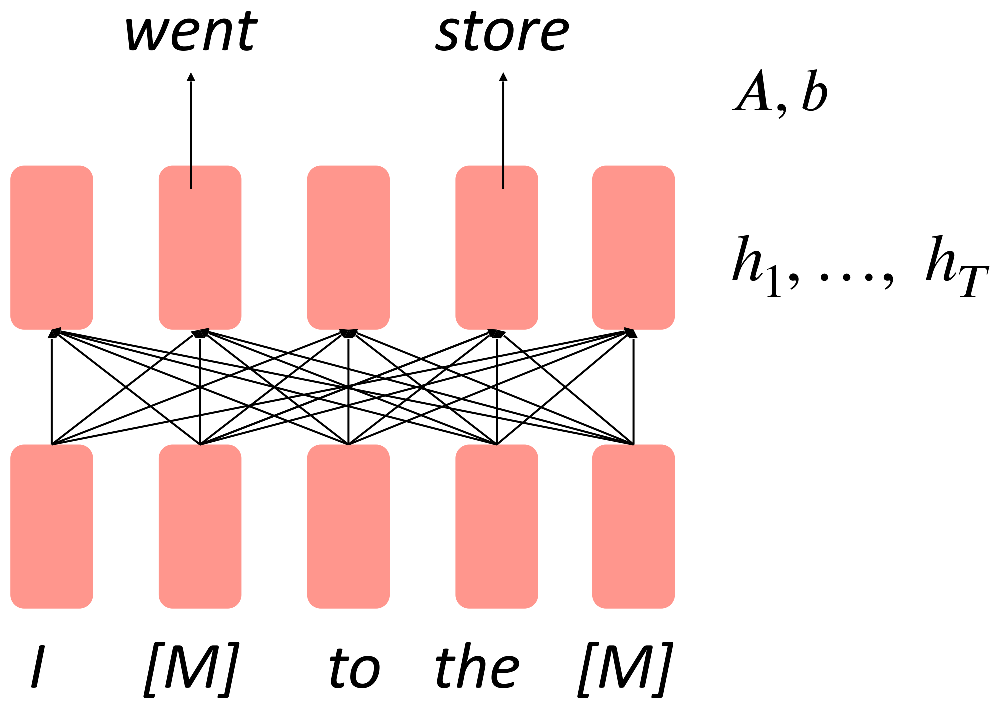
$$y_i \sim Aw_i + b$$

Only add loss terms from the masked tokens. If $\tilde{x}$ is the masked version of x, we're learning $p_\theta(x \mid \tilde{x})$. Called **Masked Language model (MLM)**.

Encoder: Training Objective

[Devlin et al., 2018]

- How to encode information from both **bidirectional** contexts?
- General Idea: **text reconstruction!**
  - Your time is limited so don't waste it living someone else's life. Don't be trapped by dogma which is [MASK] with the results of other [MASK]'s thinking. – [MASK] Jobs



$$h_1, \ldots, h_T = \text{Encoder}(w_1, \ldots, w_T)$$

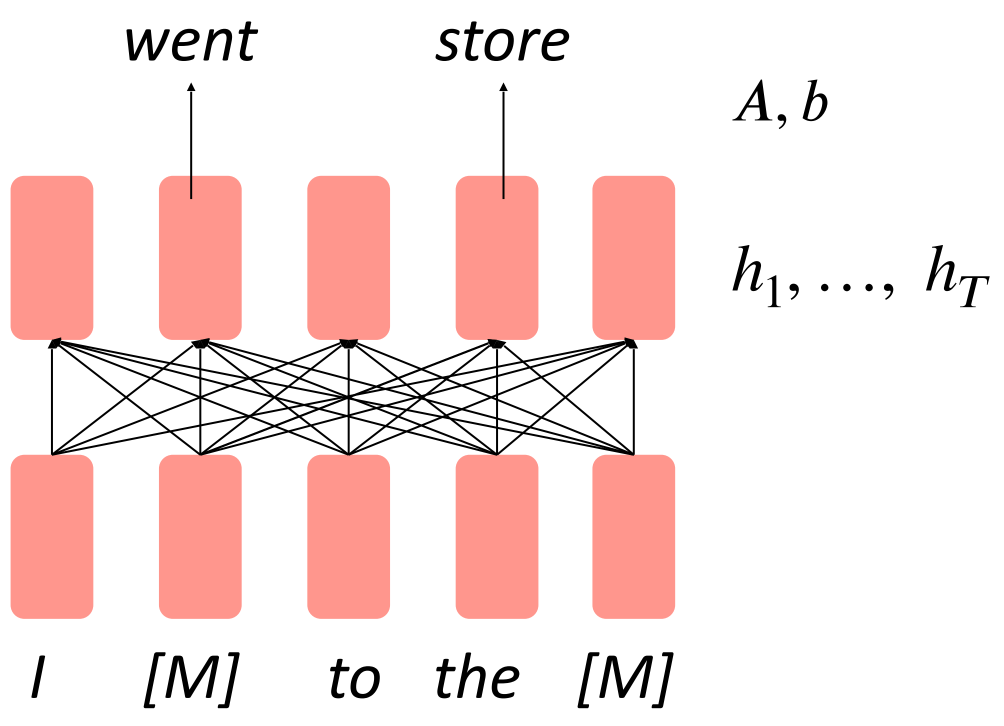
$$y_i \sim Aw_i + b$$

Only add loss terms from the masked tokens. If $\tilde{x}$ is the masked version of x, we're learning $p_\theta(x \mid \tilde{x})$. Called **Masked Language model (MLM)**.

Encoder: Training Objective

[Devlin et al., 2018]

- How to encode information from both **bidirectional** contexts?
- General Idea: **text reconstruction!**
  - Your time is limited so don't waste it living someone else's life. Don't be trapped by dogma which is living with the results of other [MASK]'s thinking. – [MASK] Jobs



$$h_1, \ldots, h_T = \text{Encoder}(w_1, \ldots, w_T)$$

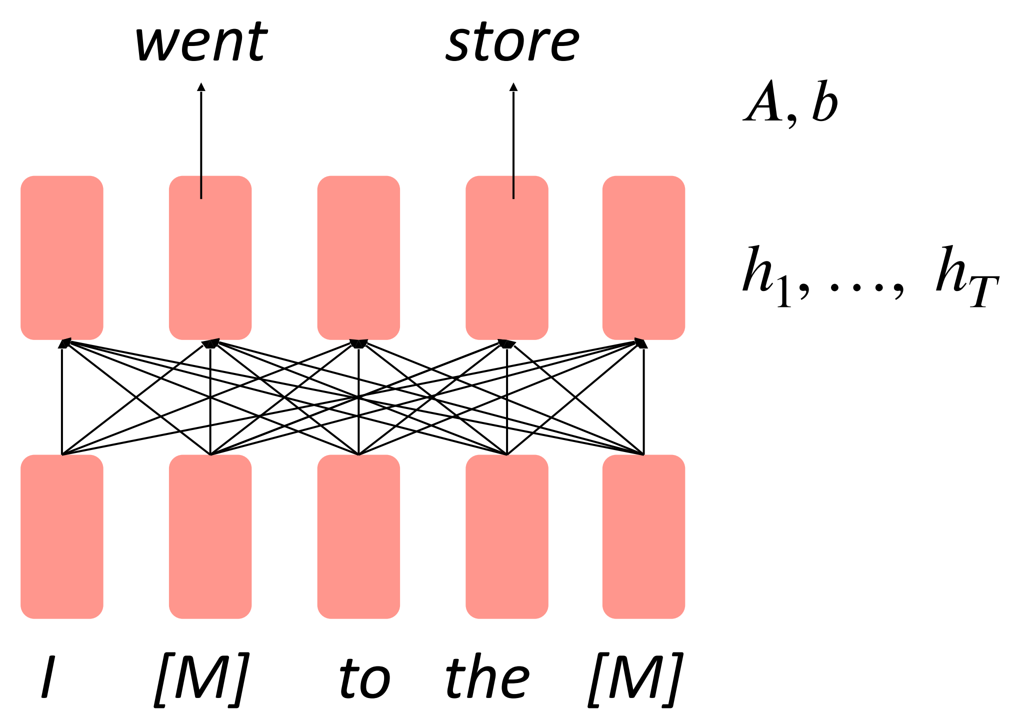
$$y_i \sim Aw_i + b$$

Only add loss terms from the masked tokens. If $\tilde{x}$ is the masked version of x, we're learning $p_\theta(x \mid \tilde{x})$. Called **Masked Language model (MLM)**.

Encoder: Training Objective

[Devlin et al., 2018]

- How to encode information from both **bidirectional** contexts?
- General Idea: **text reconstruction!**
  - Your time is limited so don't waste it living someone else's life. Don't be trapped by dogma which is living with the results of other people's thinking. – [MASK] Jobs



$$h_1, \ldots, h_T = \text{Encoder}(w_1, \ldots, w_T)$$

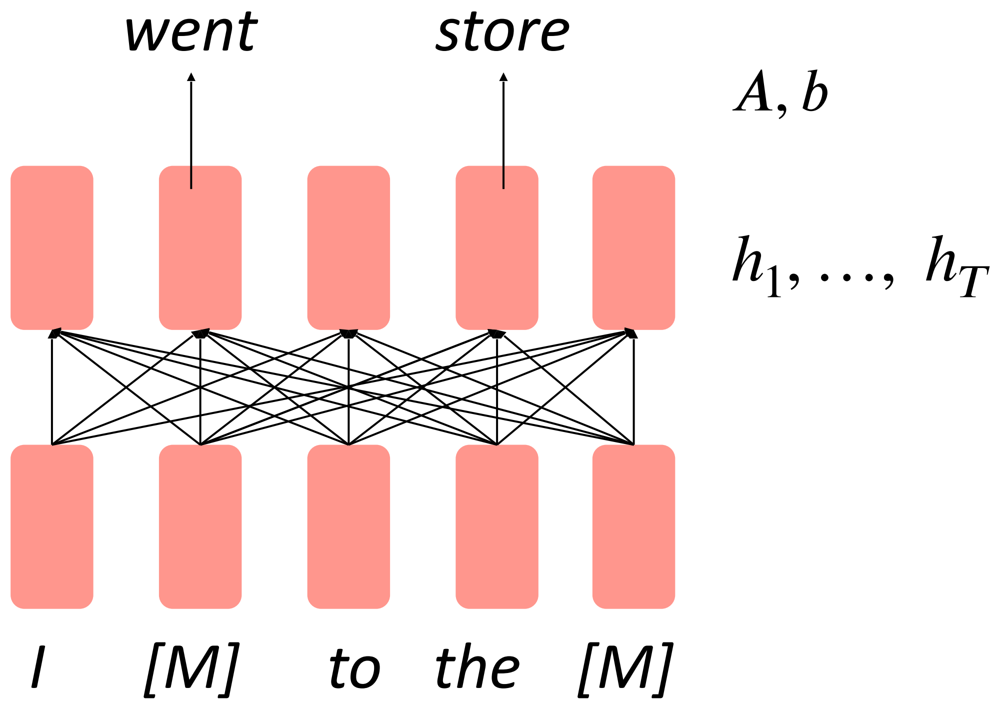
$$y_i \sim Aw_i + b$$

Only add loss terms from the masked tokens. If $\tilde{x}$ is the masked version of x, we're learning $p_\theta(x \mid \tilde{x})$. Called **Masked Language model (MLM)**.

Encoder: Training Objective

[Devlin et al., 2018]

- How to encode information from both **bidirectional** contexts?
- General Idea: **text reconstruction!**
  - Your time is limited so don't waste it living someone else's life. Don't be trapped by dogma which is living with the results of other people's thinking. – Steve Jobs



$$h_1, \ldots, h_T = \text{Encoder}(w_1, \ldots, w_T)$$

$$y_i \sim Aw_i + b$$

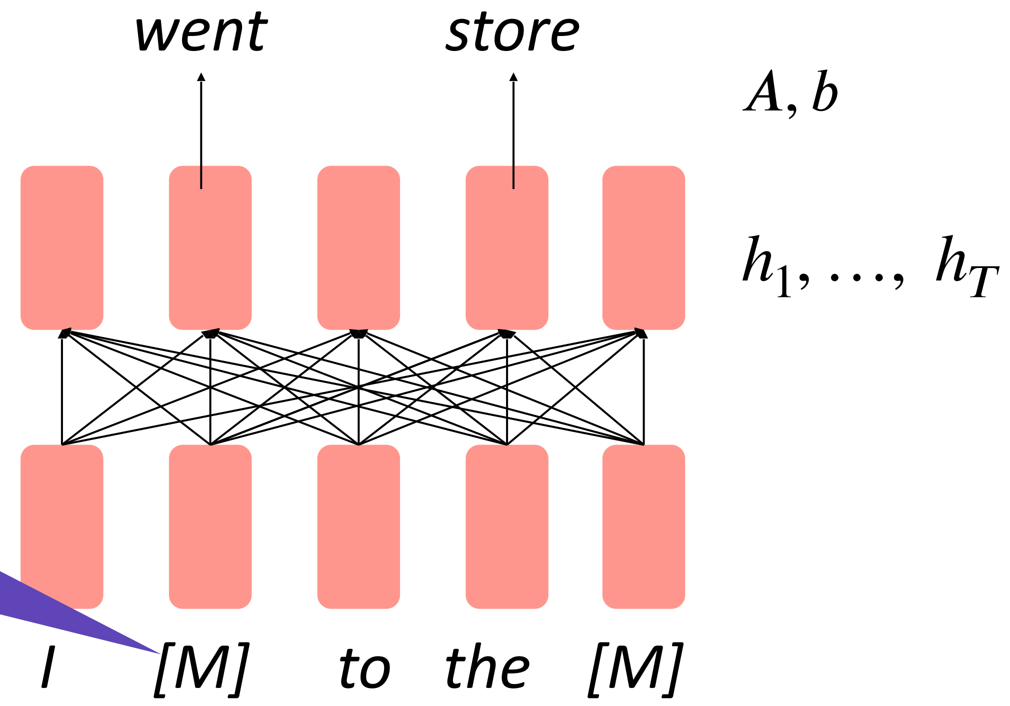
Only add loss terms from the masked tokens. If $\tilde{x}$ is the masked version of x, we're learning $p_\theta(x \mid \tilde{x})$. Called **Masked Language model (MLM)**.

Encoder: Training Objective

[Devlin et al., 2018]

- How to encode information from both **bidirectional** contexts?
- General Idea: **text reconstruction!**
  - Your time is limited so don't waste it living someone else's life. Don't be trapped by dogma which is living with the results of other people's thinking. – Steve Jobs

Since the identity of the word is masked the model can no longer cheat



$$h_1, \ldots, h_T = \text{Encoder}(w_1, \ldots, w_T)$$

$$y_i \sim Aw_i + b$$

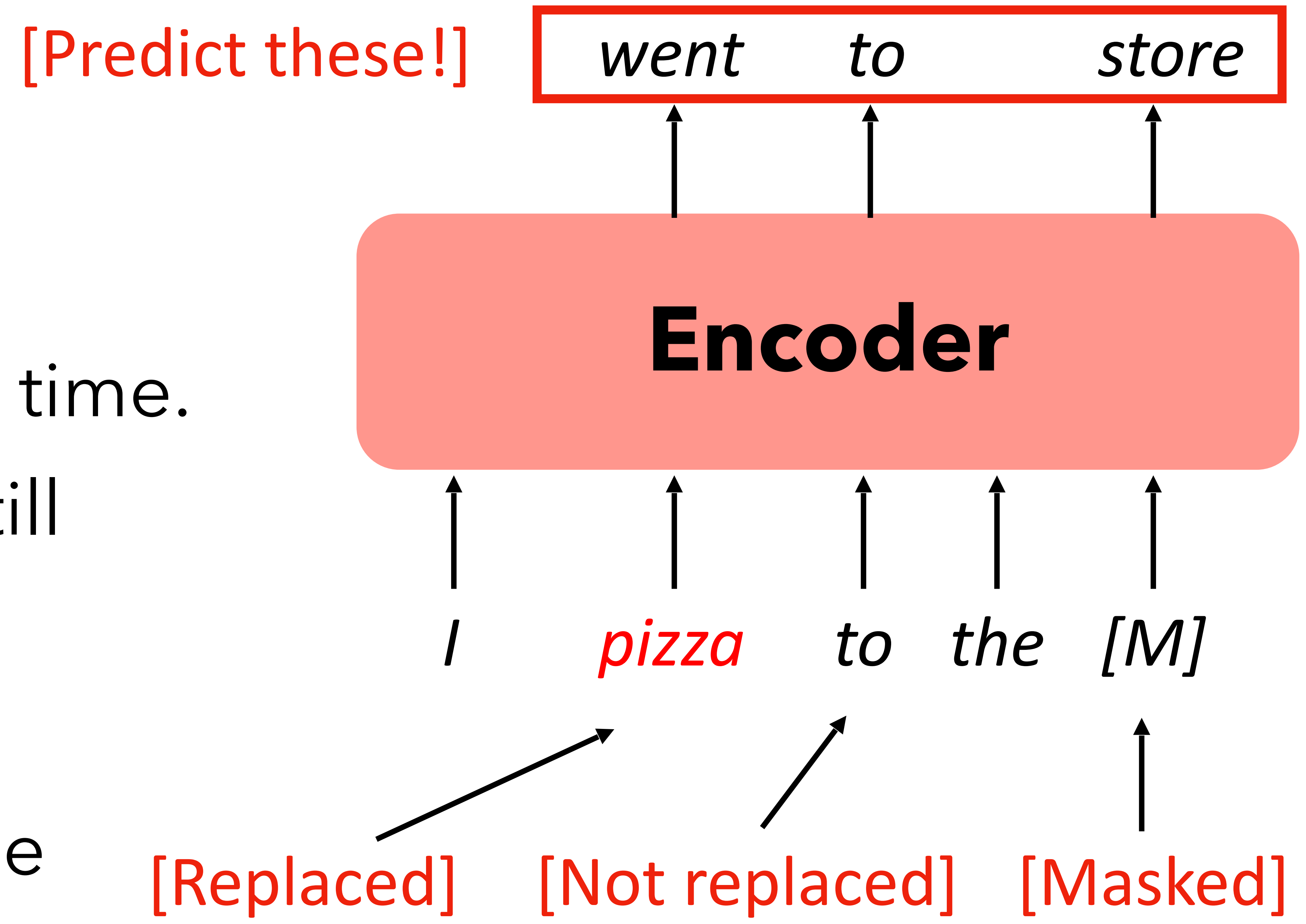
Only add loss terms from the masked tokens. If $\tilde{x}$ is the masked version of x, we're learning $p_\theta(x \mid \tilde{x})$. Called **Masked Language model (MLM)**.

Encoder: BERT

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

[Devlin et al., 2018]

- **2 Pre-training Objectives:**
  - **Masked LM: Choose a random 15% of tokens to predict.**
    - For each chosen token:
      - Replace it with **[MASK]** 80% of the time.
      - Replace it with a **random token** 10% of the time.
      - Leave it **unchanged** 10% of the time (but still predict it!).
  - **Next Sentence Prediction (NSP)**
    - 50% of the time two adjacent sentences are in the correct order.
    - **This actually hurts model learning based on later work!**



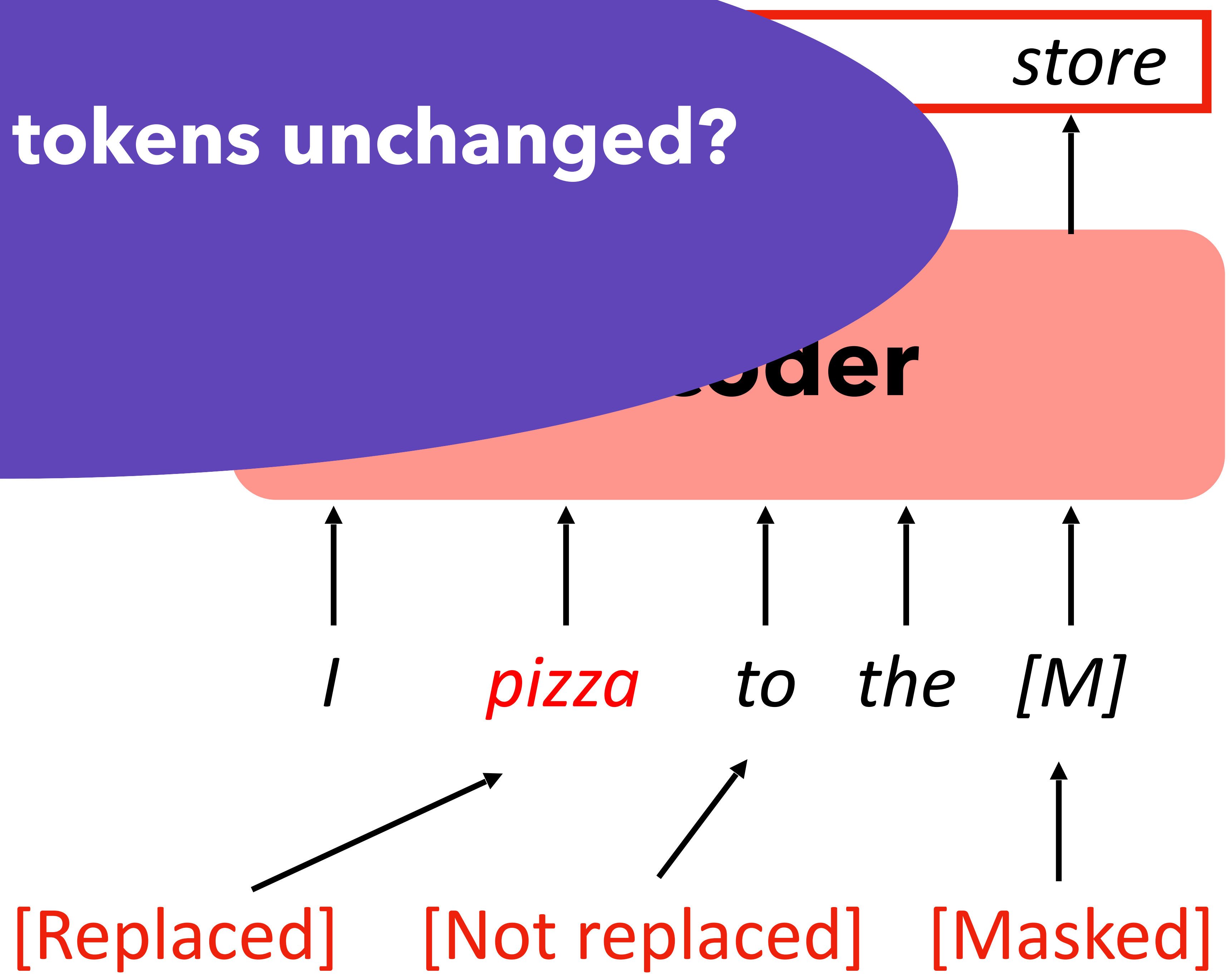
Encoder: BERT

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

[Devlin et al., 2018]

- **2 Pre-training Objectives:**
  - **Masked LM: Choose a random 15% of tok predict.**
    - For each chosen token:
      - Replace it with **[MASK]**
      - Replace it with a **random to**
      - Leave it **unchanged** 10% of the time (but still predict it!).
  - **Next Sentence Prediction (NSP)**
    - 50% of the time two adjacent sentences are in the correct order.
    - **This actually hurts model learning based on later work!**

WHY keeping some tokens unchanged?



Encoder: BERT

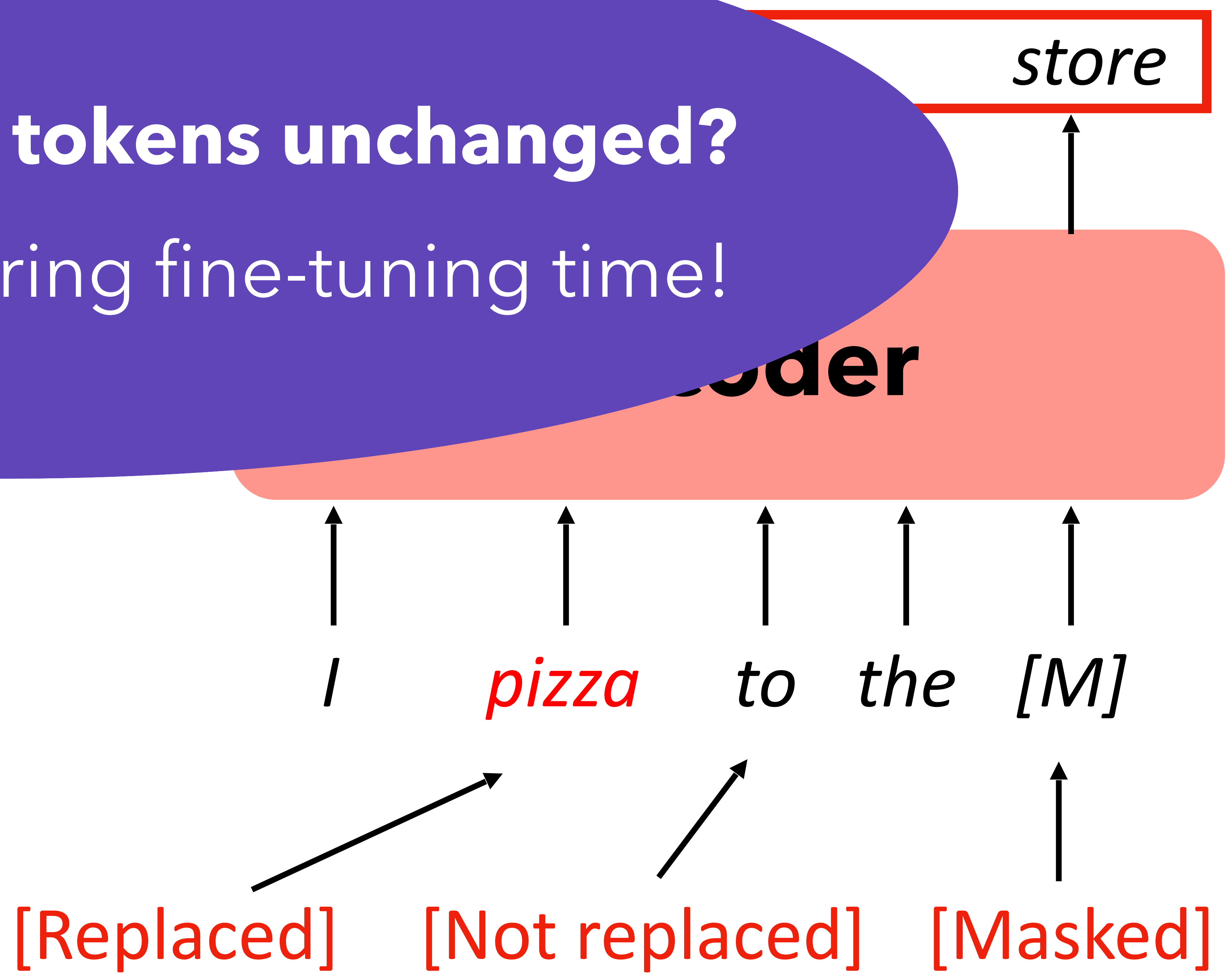
Bidirectional **E**ncoder **R**epresentations from **T**ransformers

[Devlin et al., 2018]

- **2 Pre-training Objectives:**
  - **Masked LM: Choose a random 15% of tok**
    predict.
    - For each chosen token:
      - Replace it with **[MASK]**
      - Replace it with a **random to**
      - Leave it **unchanged** 10% of the time (but still predict it!).
  - **Next Sentence Prediction (NSP)**
    - 50% of the time two adjacent sentences are in the correct order.
    - **This actually hurts model learning based on later work!**

WHY keeping some tokens unchanged?

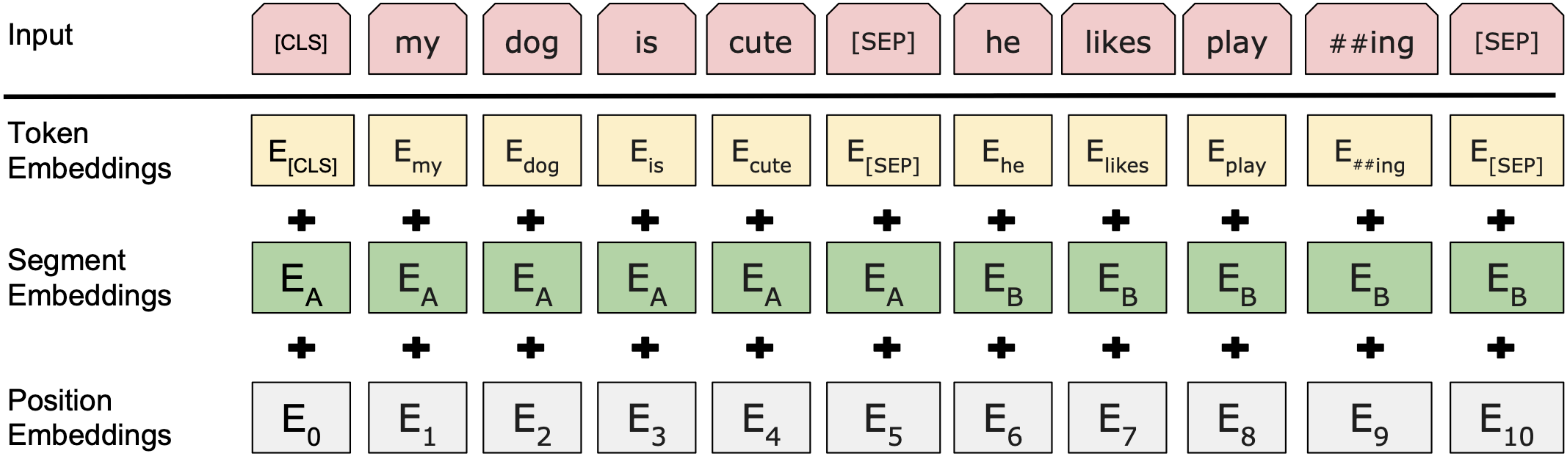
There's no [MASK] during fine-tuning time!



Encoder: BERT

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

[Devlin et al., 2018]

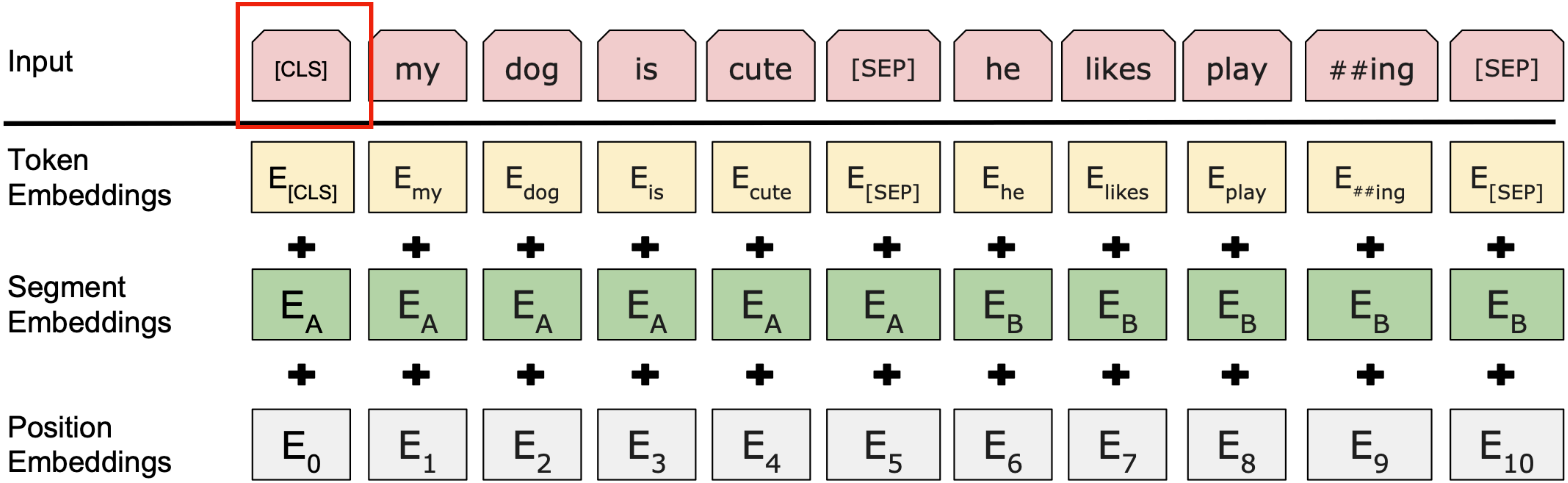


Encoder: BERT

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

[Devlin et al., 2018]

Special token added to the beginning of each input sequence



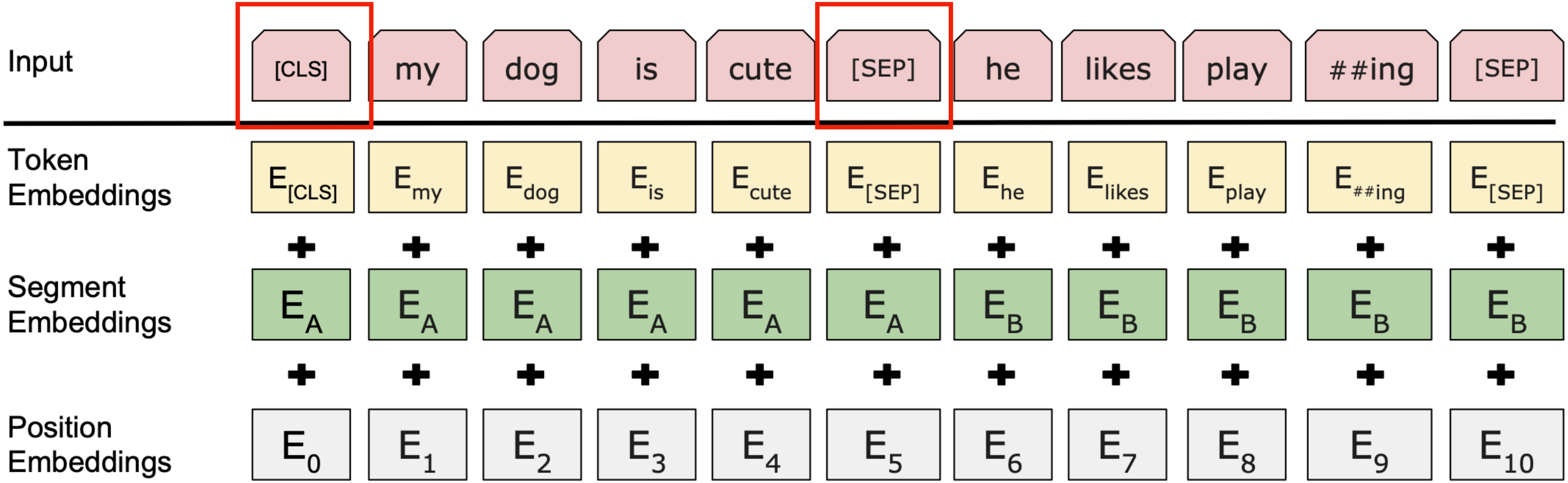
Encoder: BERT

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

[Devlin et al., 2018]

Special token added to the beginning of each input sequence

Special token to separate sentence A/B



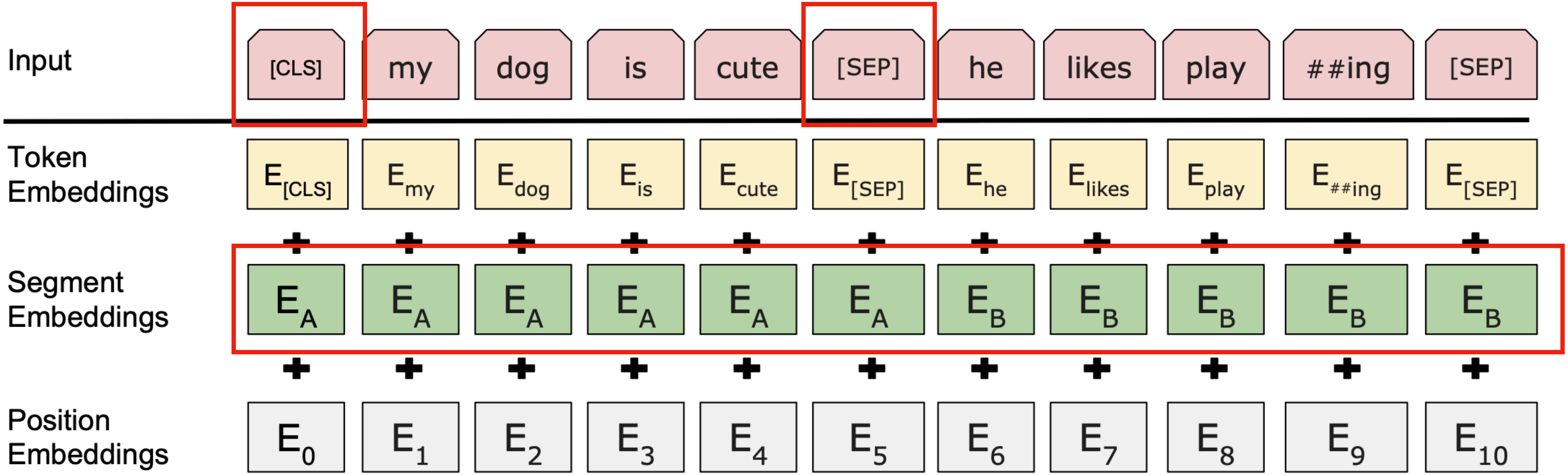
Encoder: BERT

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

[Devlin et al., 2018]

Special token added to the beginning of each input sequence

Special token to separate sentence A/B



Learned embedding to every token indicating whether it belongs to sentence A or sentence B

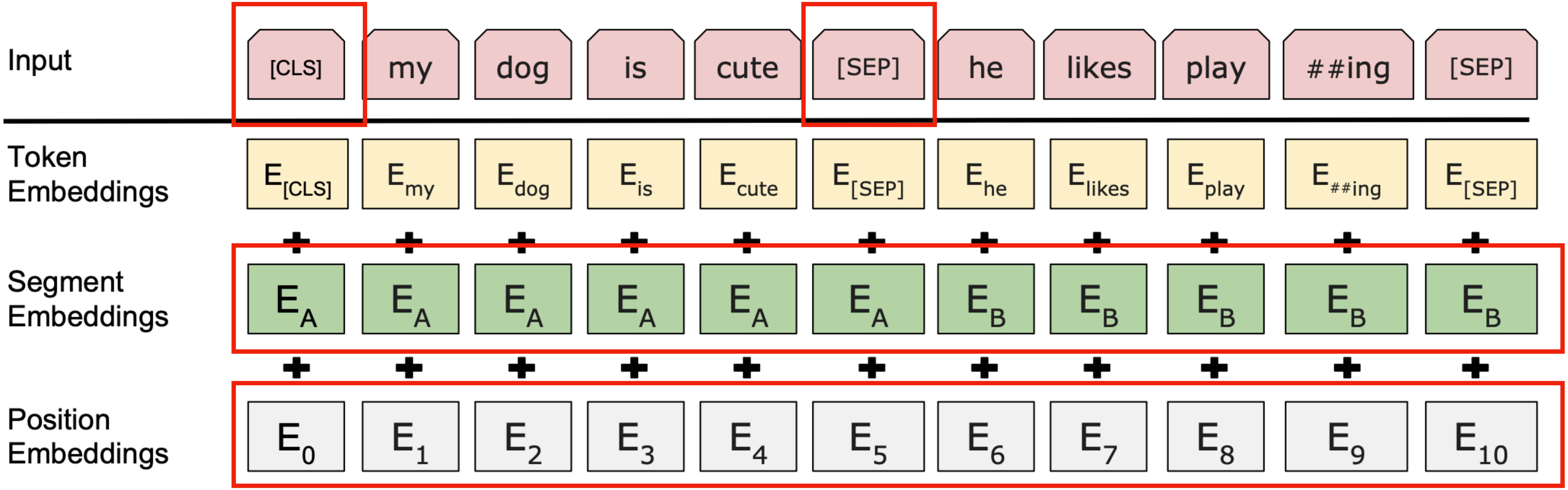
Encoder: BERT

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

[Devlin et al., 2018]

Special token added to the beginning of each input sequence

Special token to separate sentence A/B



Learned embedding to every token indicating whether it belongs to sentence A or sentence B

Position of the token in the entire sequence

Encoder: BERT

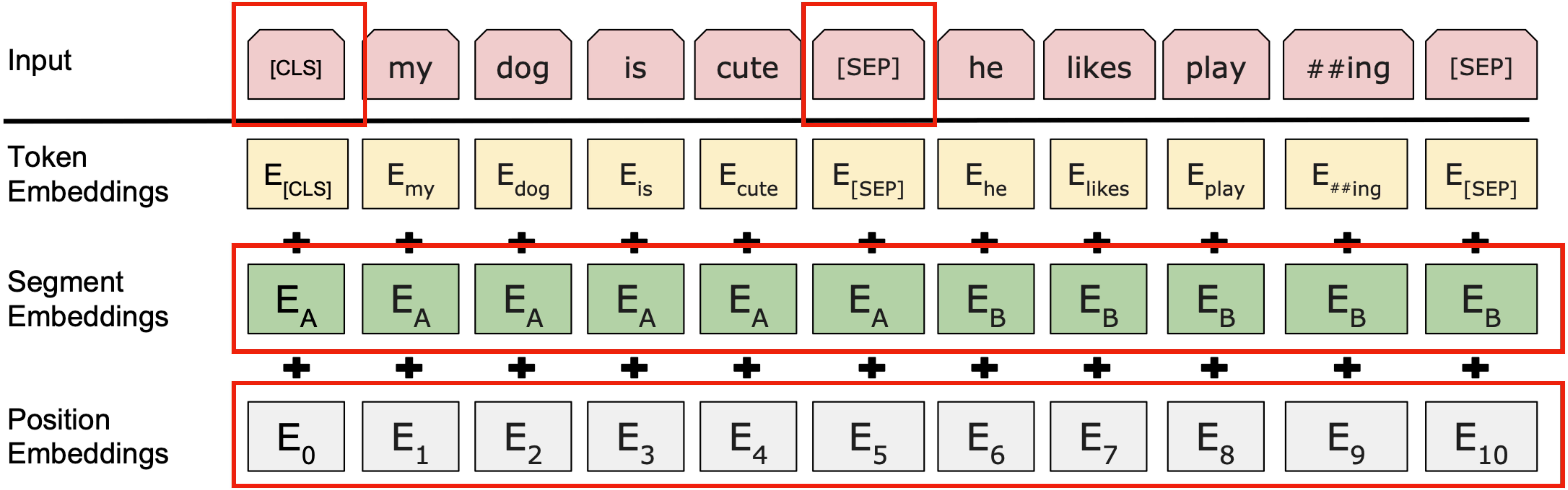
Bidirectional **E**ncoder **R**epresentations from **T**ransformers

[Devlin et al., 2018]

Special token added to the beginning of each input sequence

Special token to separate sentence A/B

Final embedding is the sum of all three!



Learned embedding to every token indicating whether it belongs to sentence A or sentence B

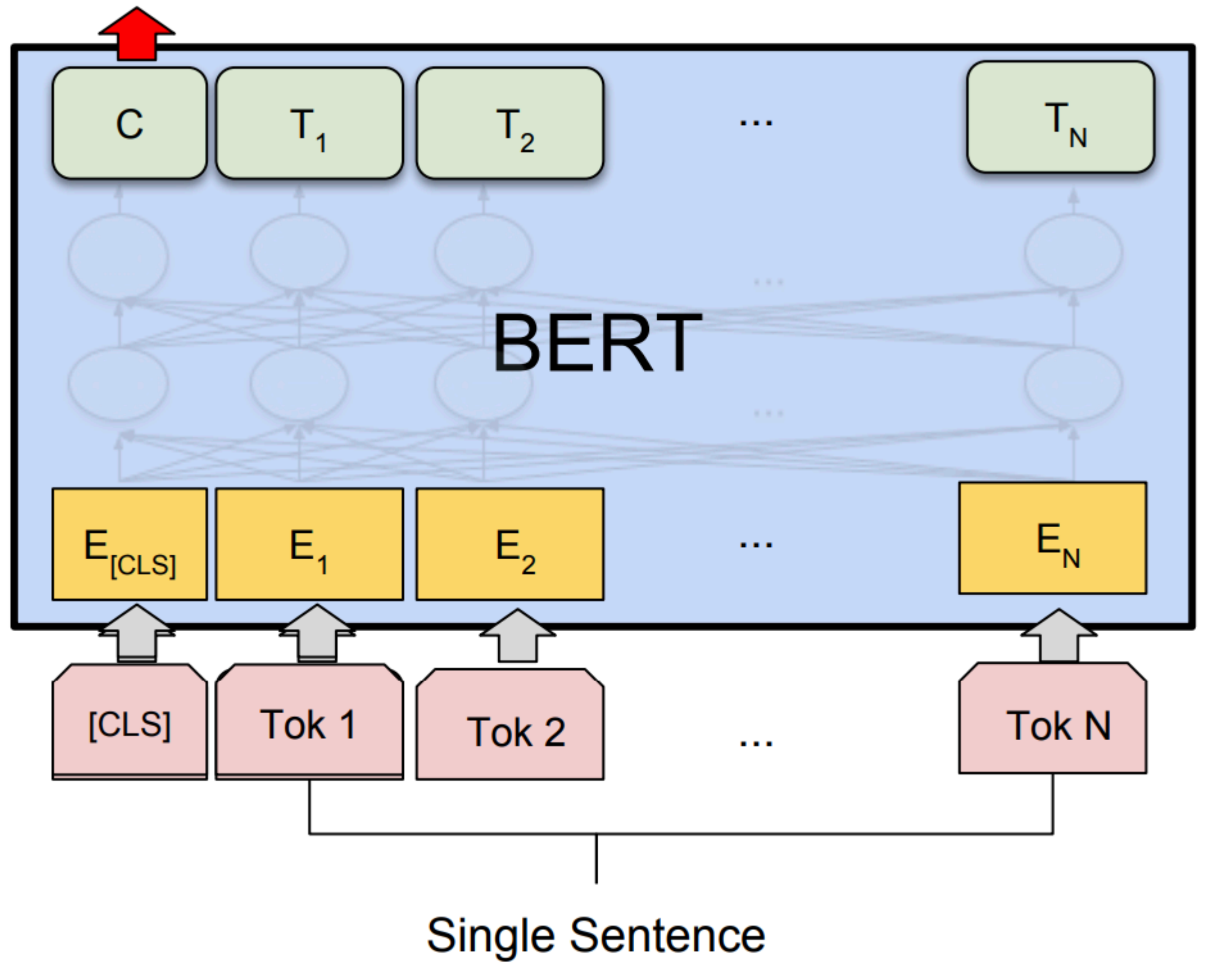
Position of the token in the entire sequence

Encoder: BERT (Fine-tuning)

Encoder: BERT (Fine-tuning)

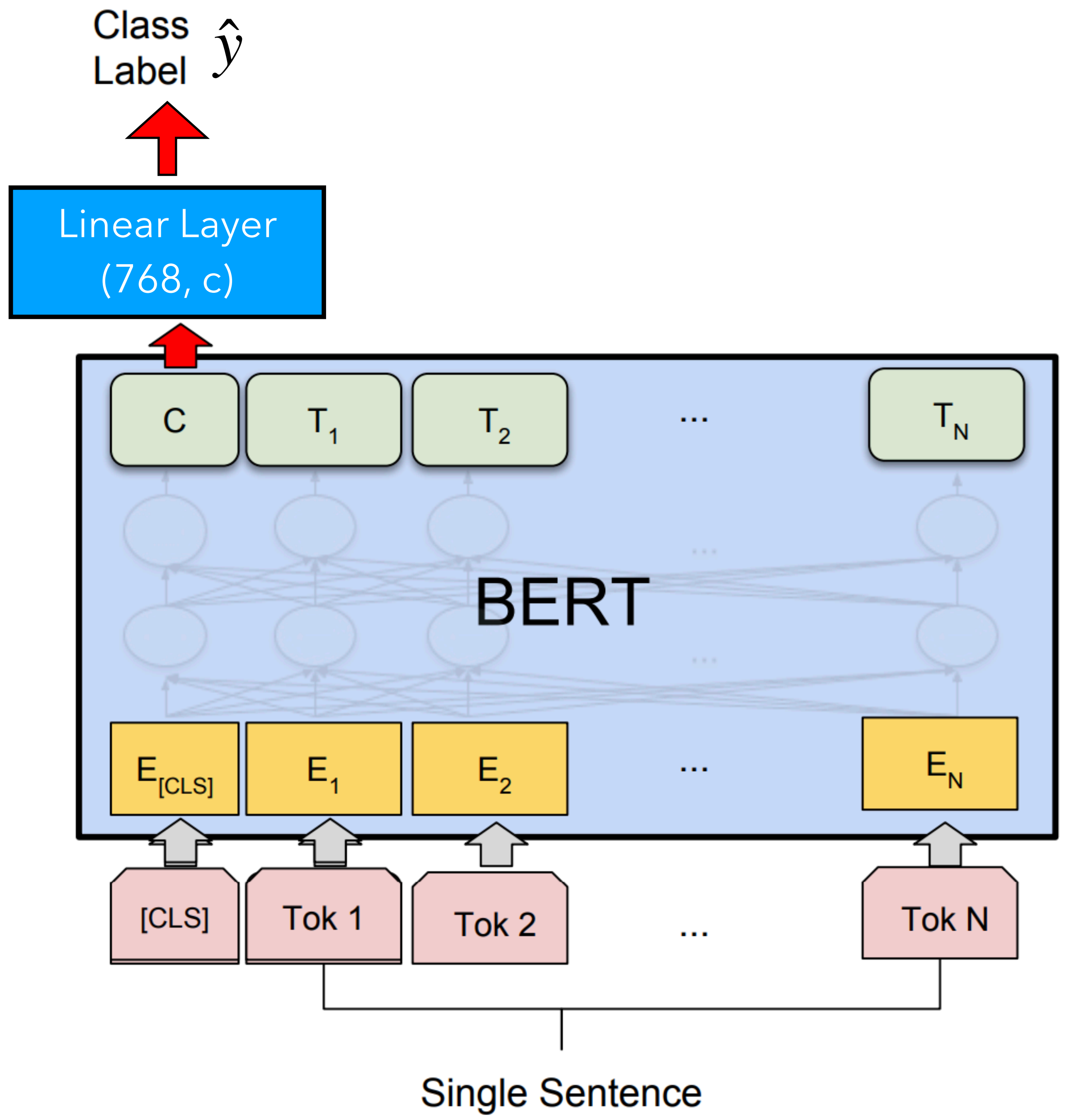
Single-Sentence Tasks like
SST-2 (Sentiment Analysis)

Encoder: BERT (Fine-tuning)



Single-Sentence Tasks like SST-2 (Sentiment Analysis)

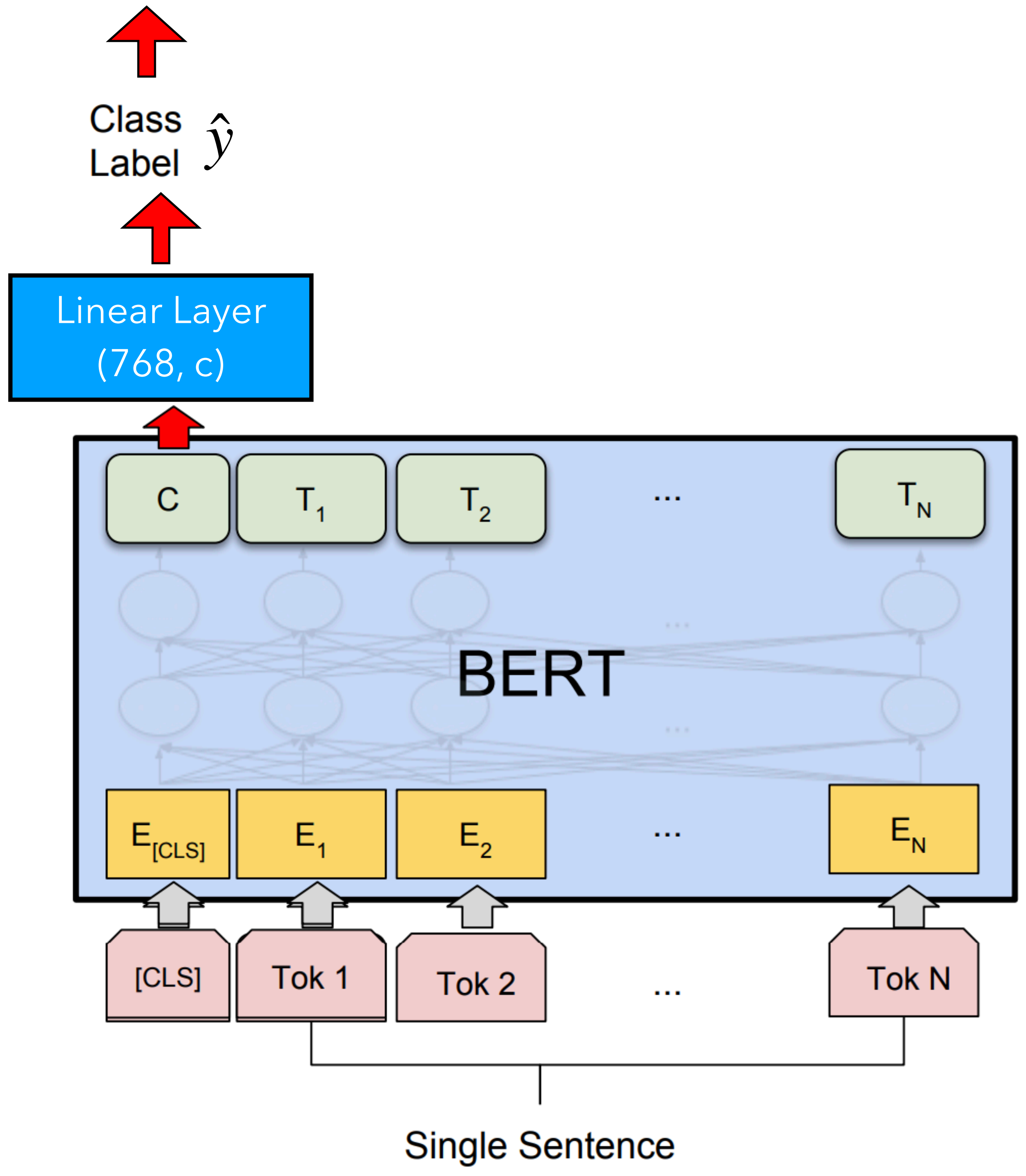
Encoder: BERT (Fine-tuning)



Single-Sentence Tasks like
SST-2 (Sentiment Analysis)

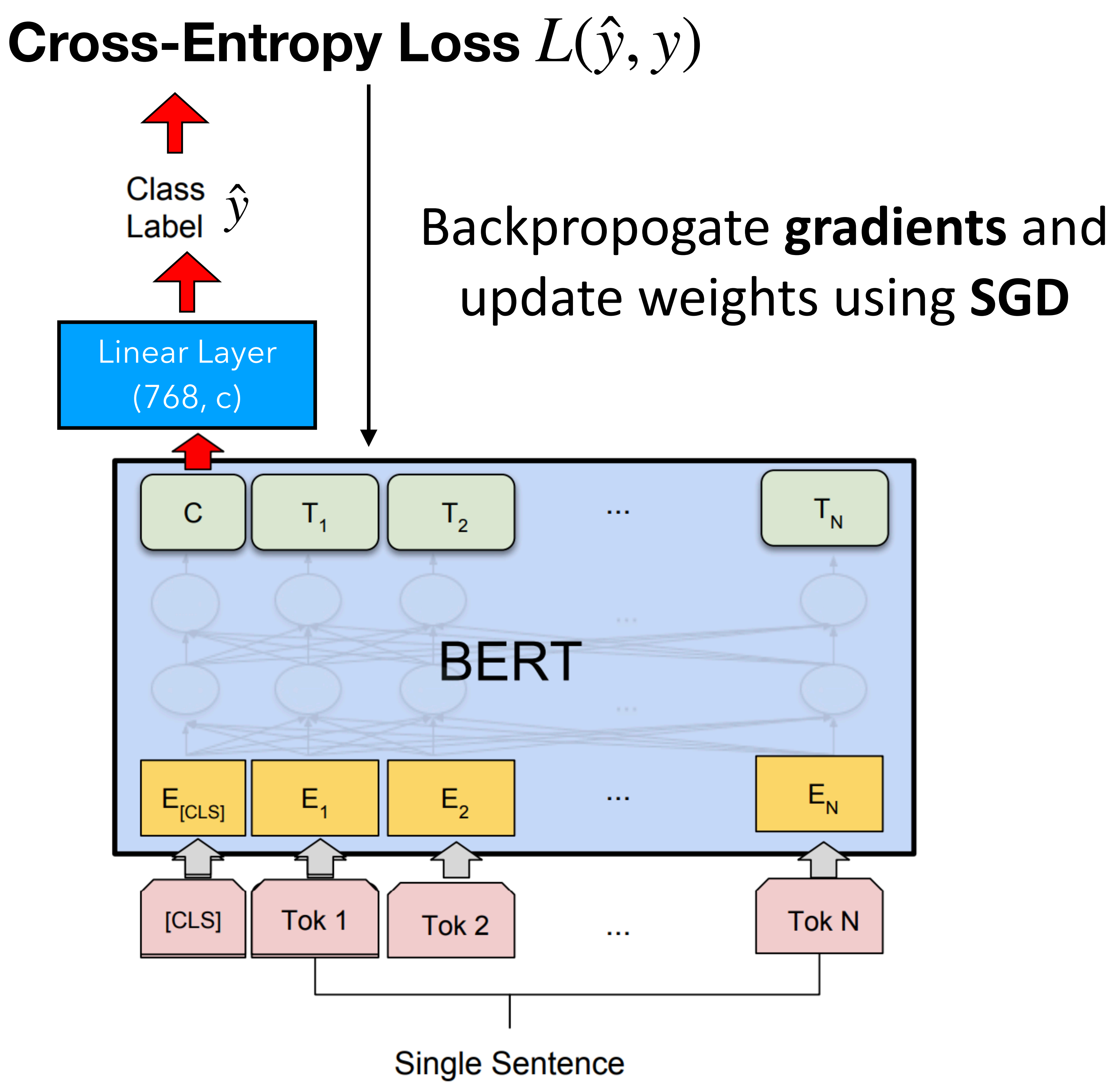
Encoder: BERT (Fine-tuning)

Cross-Entropy Loss $L(\hat{y}, y)$



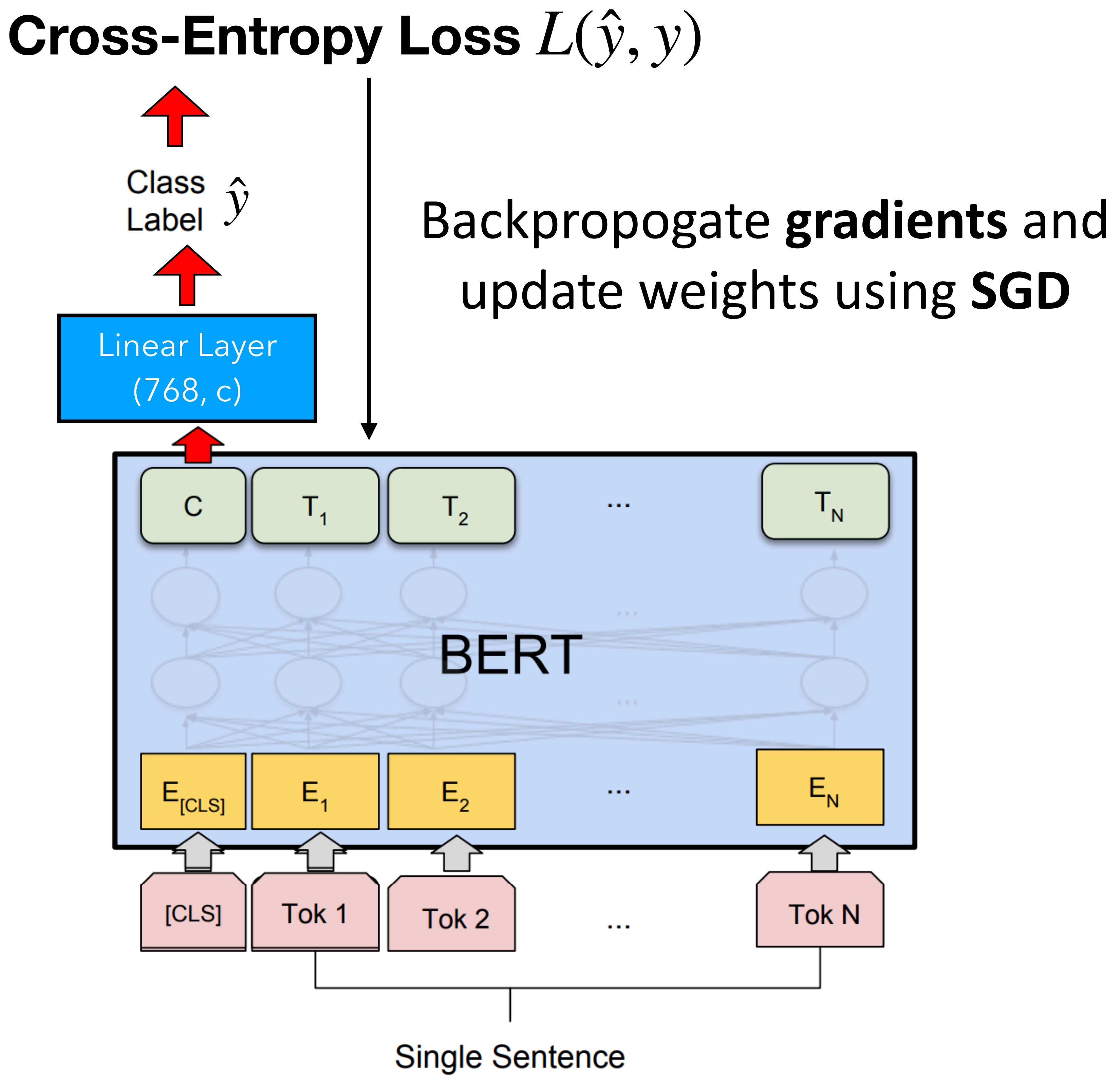
Single-Sentence Tasks like SST-2 (Sentiment Analysis)

Encoder: BERT (Fine-tuning)



Single-Sentence Tasks like SST-2 (Sentiment Analysis)

Encoder: BERT (Fine-tuning)



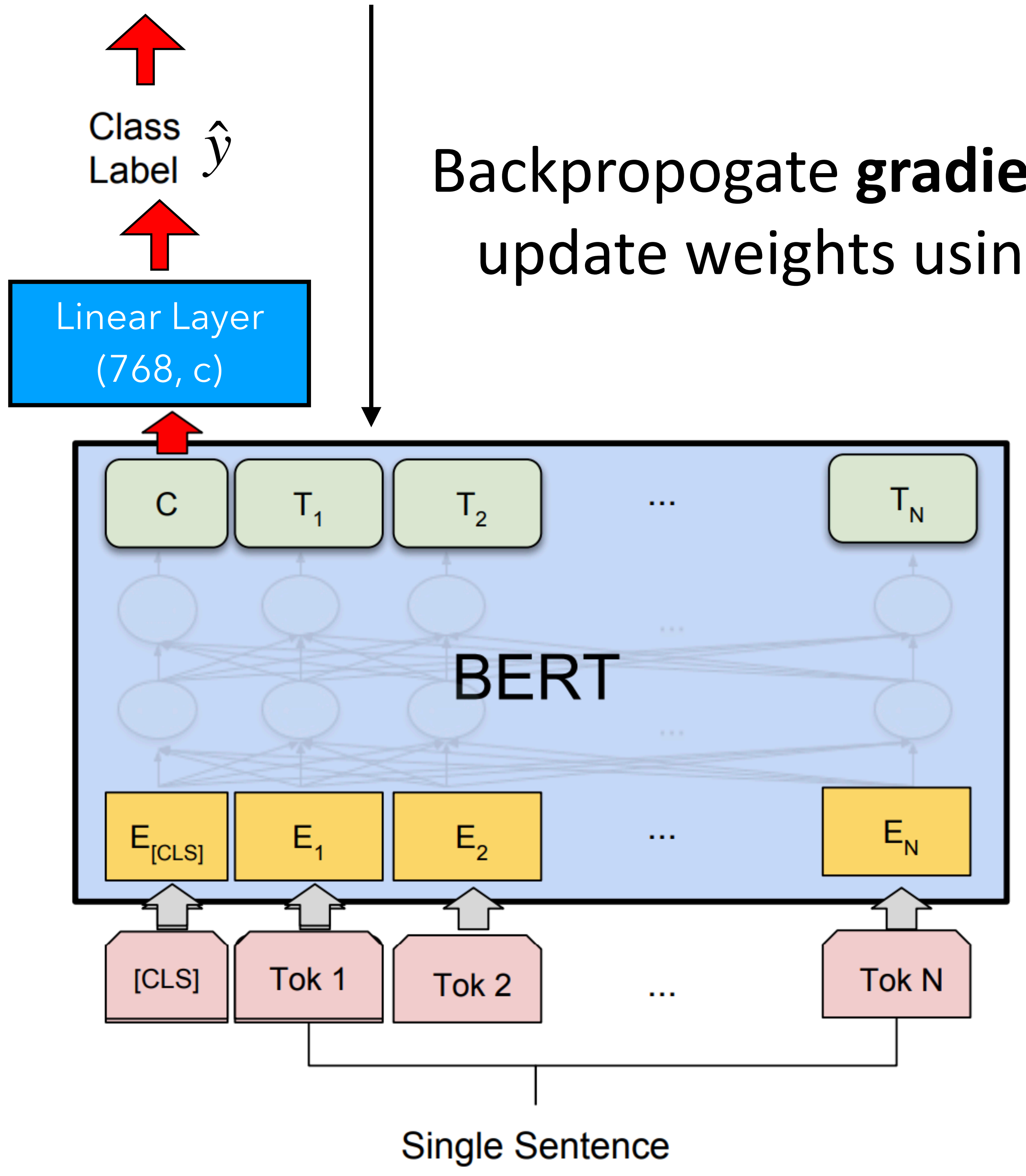
Single-Sentence Tasks like SST-2 (Sentiment Analysis)

Sentence Pair Classification Tasks like Natural Language Inference

Encoder: BERT (Fine-tuning)

Cross-Entropy Loss $L(\hat{y}, y)$

Backpropogate **gradients** and update weights using **SGD**

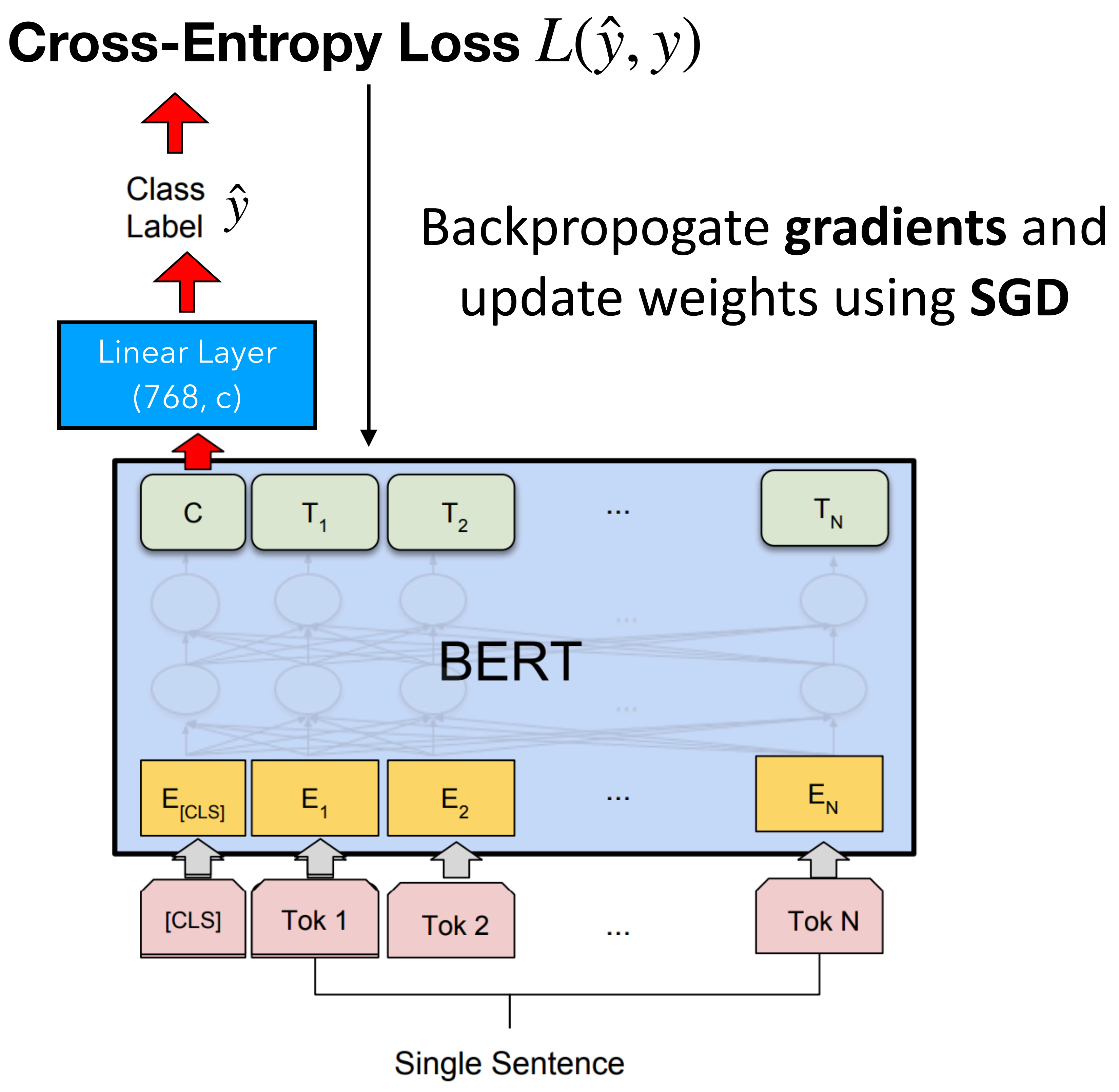


Single-Sentence Tasks like SST-2 (Sentiment Analysis)

Input:
Premise: A soccer game with multiple males playing
Hypothesis: Some men are playing a sport
Label: **Entailment** / Neutral / Contadiction

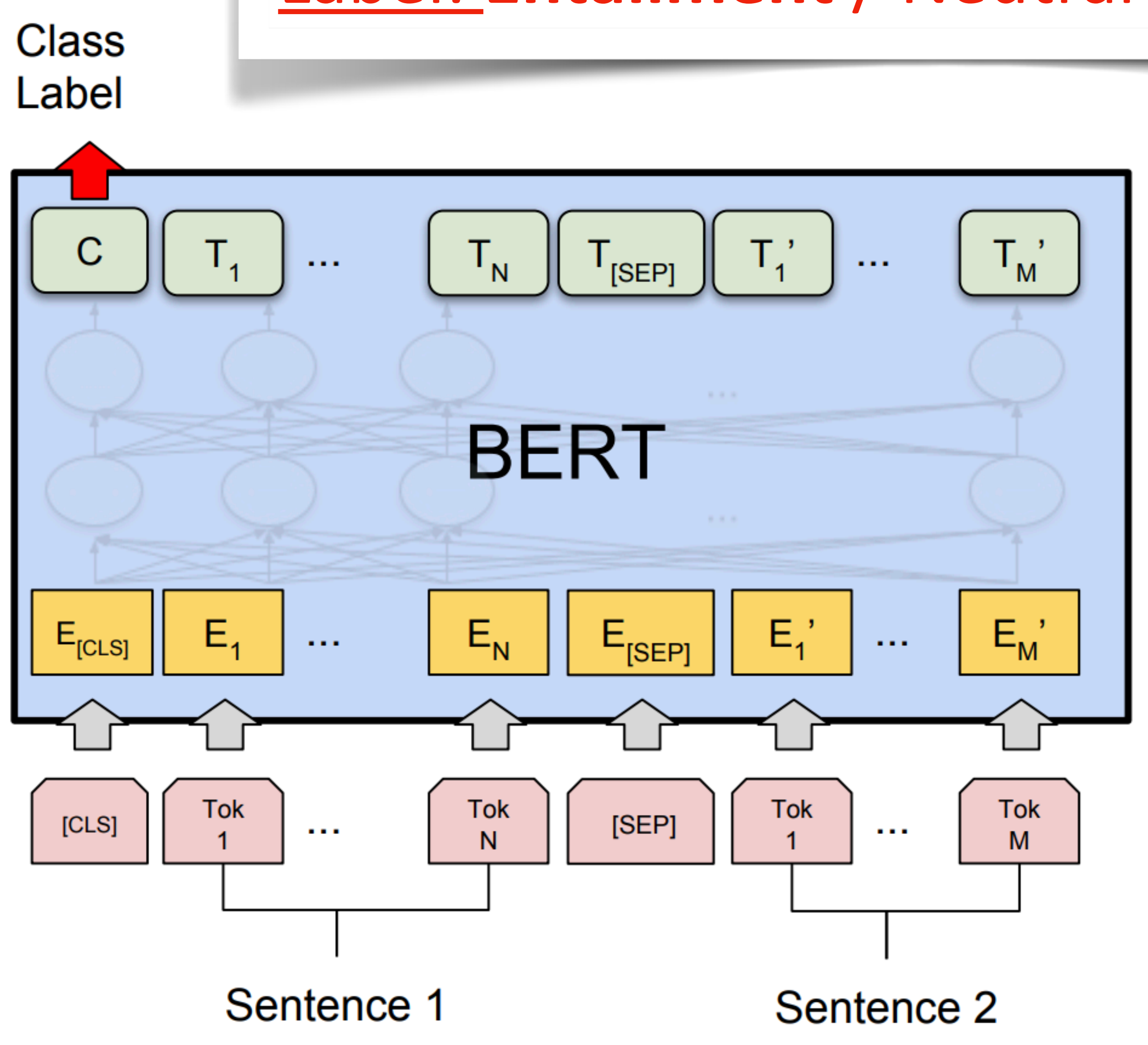
Sentence Pair Classification Tasks like Natural Language Inference

Encoder: BERT (Fine-tuning)



Single-Sentence Tasks like SST-2 (Sentiment Analysis)

Input:
Premise: A soccer game with multiple males playing
Hypothesis: Some men are playing a sport
Label: **Entailment** / Neutral / Contadiction



Sentence Pair Classification Tasks like Natural Language Inference

Encoder: BERT

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

[Devlin et al., 2018]

- **SOTA at the time on a wide range of tasks after fine-tuning!**

System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Average**
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

- **QQP:** Quora Question Pairs (detect paraphrase questions)
- **QNLI:** natural language inference over question answering data
- **SST-2:** sentiment analysis
- **CoLA:** corpus of linguistic acceptability (detect whether sentences are grammatical.)
- **STS-B:** semantic textual similarity
- **MRPC:** microsoft paraphrase corpus
- **RTE:** a small natural language inference corpus

Encoder: BERT

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

[Devlin et al., 2018]

- **SOTA at the time on a wide range of tasks after fine-tuning!**

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	**Average** -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

- **QQP:** Quora Question Pairs (detect paraphrase questions)
- **QNLI:** natural language inference over question answering data
- **SST-2:** sentiment analysis
- **CoLA:** corpus of linguistic acceptability (detect whether sentences are grammatical.)
- **STS-B:** semantic textual similarity
- **MRPC:** microsoft paraphrase corpus
- **RTE:** a small natural language inference corpus

Encoder: BERT

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

[Devlin et al., 2018]

Encoder: BERT

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

[Devlin et al., 2018]

SWAG (Commonsense inference task)

System	Dev	Test
ESIM+GloVe	51.9	52.7
ESIM+ELMo	59.1	59.2
OpenAI GPT	-	78.0
$\text{BERT}_{\text{BASE}}$	81.6	-
$\text{BERT}_{\text{LARGE}}$	**86.6**	**86.3**
Human (expert)†	-	85.0
Human (5 annotations)†	-	88.0

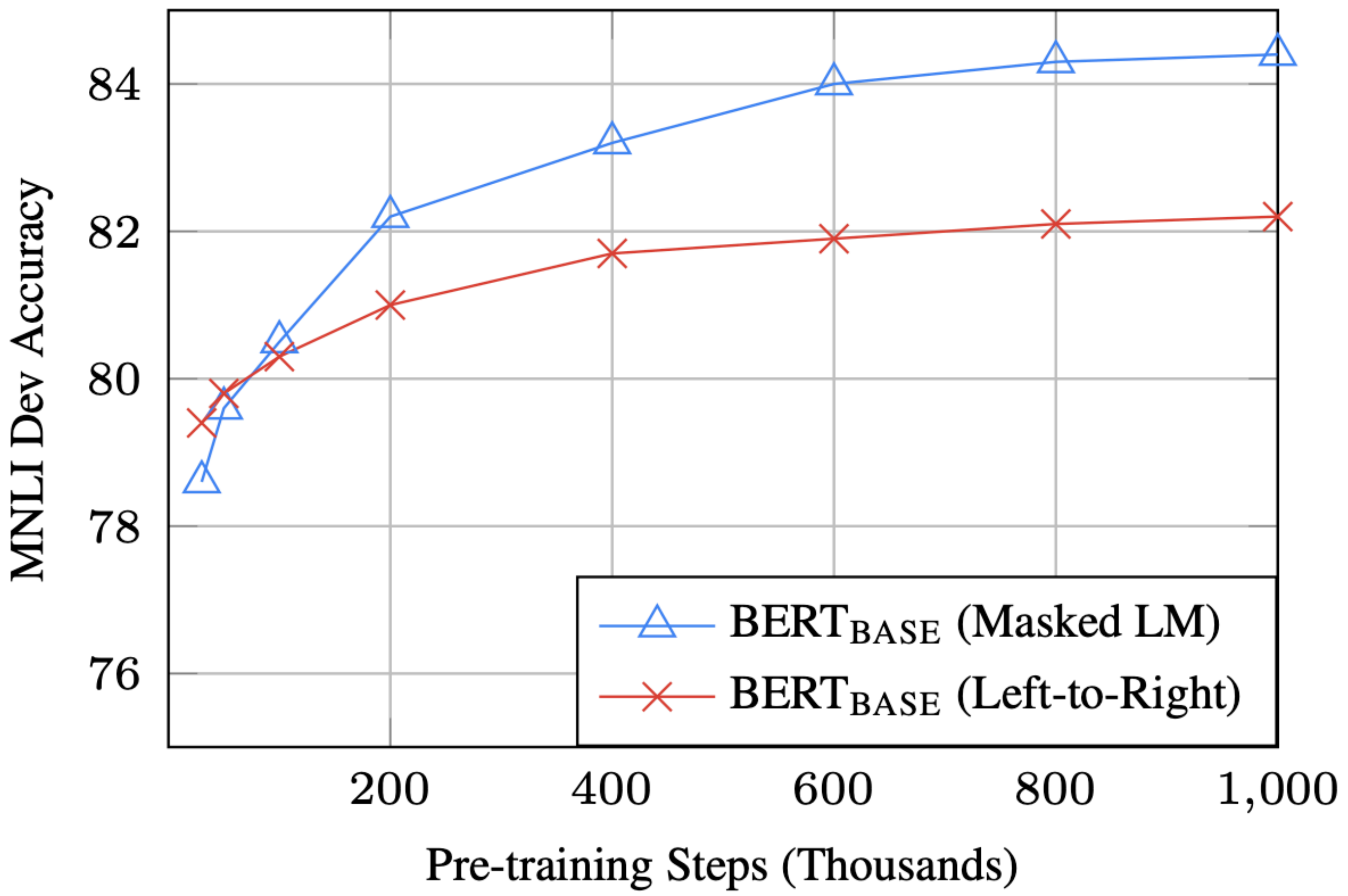
- **Two Sizes of Models**
  - **Base:** 110M, 4 Cloud TPUs, 4 days
  - **Large:** 340M, 16 Cloud TPUs, 4 days
  - Both models can be fine-tuned with single GPU
  - The larger the better!

Encoder: BERT

Bidirectional **E**ncoder **R**epresentations from **T**ransformers [Devlin et al., 2018]

SWAG (Commonsense inference task)

System	Dev	Test
ESIM+GloVe	51.9	52.7
ESIM+ELMo	59.1	59.2
OpenAI GPT	-	78.0
$BERT_{BASE}$	81.6	-
$BERT_{LARGE}$	**86.6**	**86.3**
Human (expert)†	-	85.0
Human (5 annotations)†	-	88.0



- **Two Sizes of Models**
  - **Base:** 110M, 4 Cloud TPUs, 4 days
  - **Large:** 340M, 16 Cloud TPUs, 4 days
  - Both models can be fine-tuned with single GPU
  - The larger the better!
- MLM converges slower than Left-to-Right at the beginning, but out-performers it eventually

Encoder: RoBERTa

[Liu et al., 2019]

- **Original BERT is significantly undertrained!**
- More data (16G => 160G)
- Pre-train for longer
- Bigger batches
- Removing the next sentence prediction (NSP) objective
- Training on longer sequences
- Dynamic masking, randomly masking out different tokens
- A larger byte-level BPE vocabulary containing 50K sub-word units

Encoder: RoBERTa

[Liu et al., 2019]

- **Original BERT is significantly undertrained!**
- More data (16G => 160G)
- Pre-train for longer
- Bigger batches
- Removing the next sentence prediction (NSP) objective
- Training on longer sequences
- Dynamic masking, randomly masking out different tokens
- A larger byte-level BPE vocabulary containing 50K sub-word units

All around better than BERT!

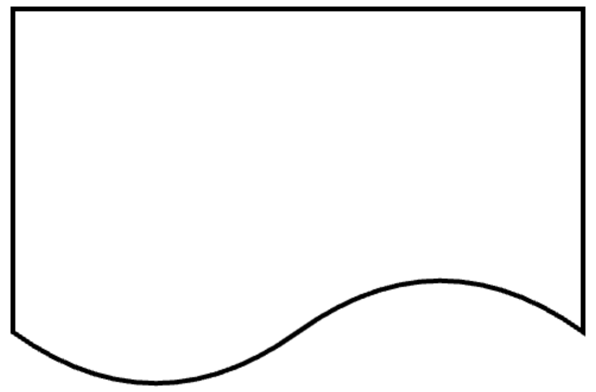
Encoders for Information Retrieval

Encoders for Information Retrieval

Retrieve the set of relevant documents given a query

Encoders for Information Retrieval

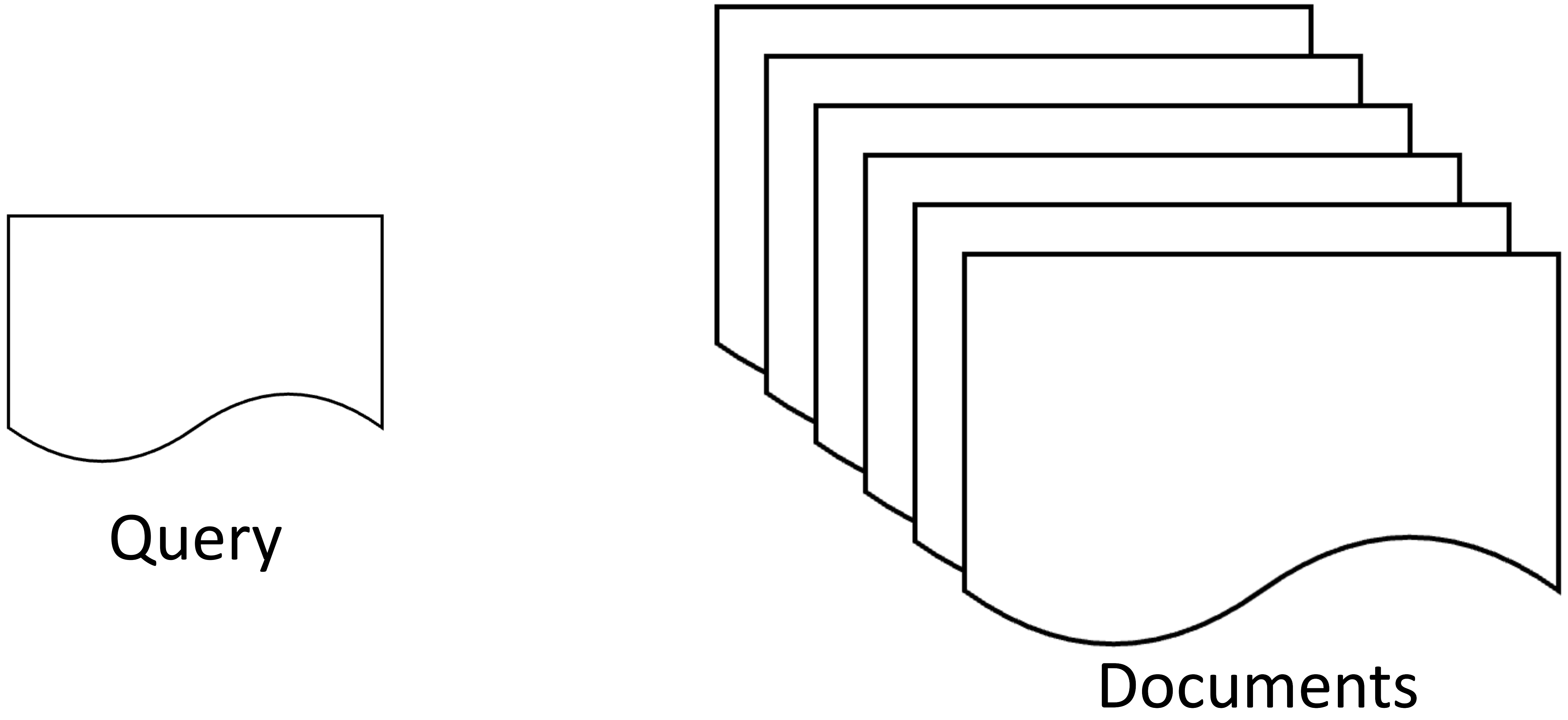
Retrieve the set of relevant documents given a query



Query

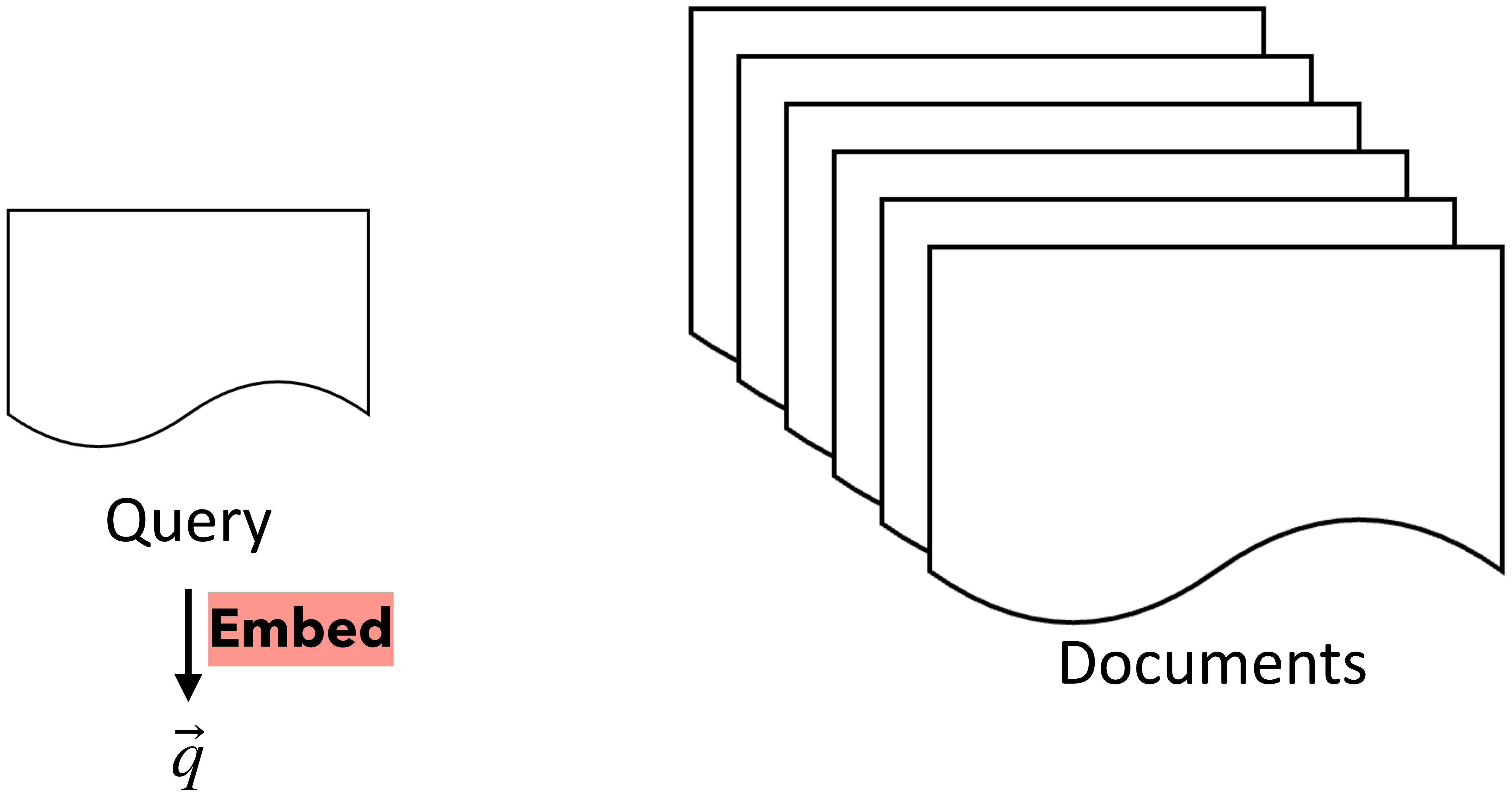
Encoders for Information Retrieval

Retrieve the set of relevant documents given a query



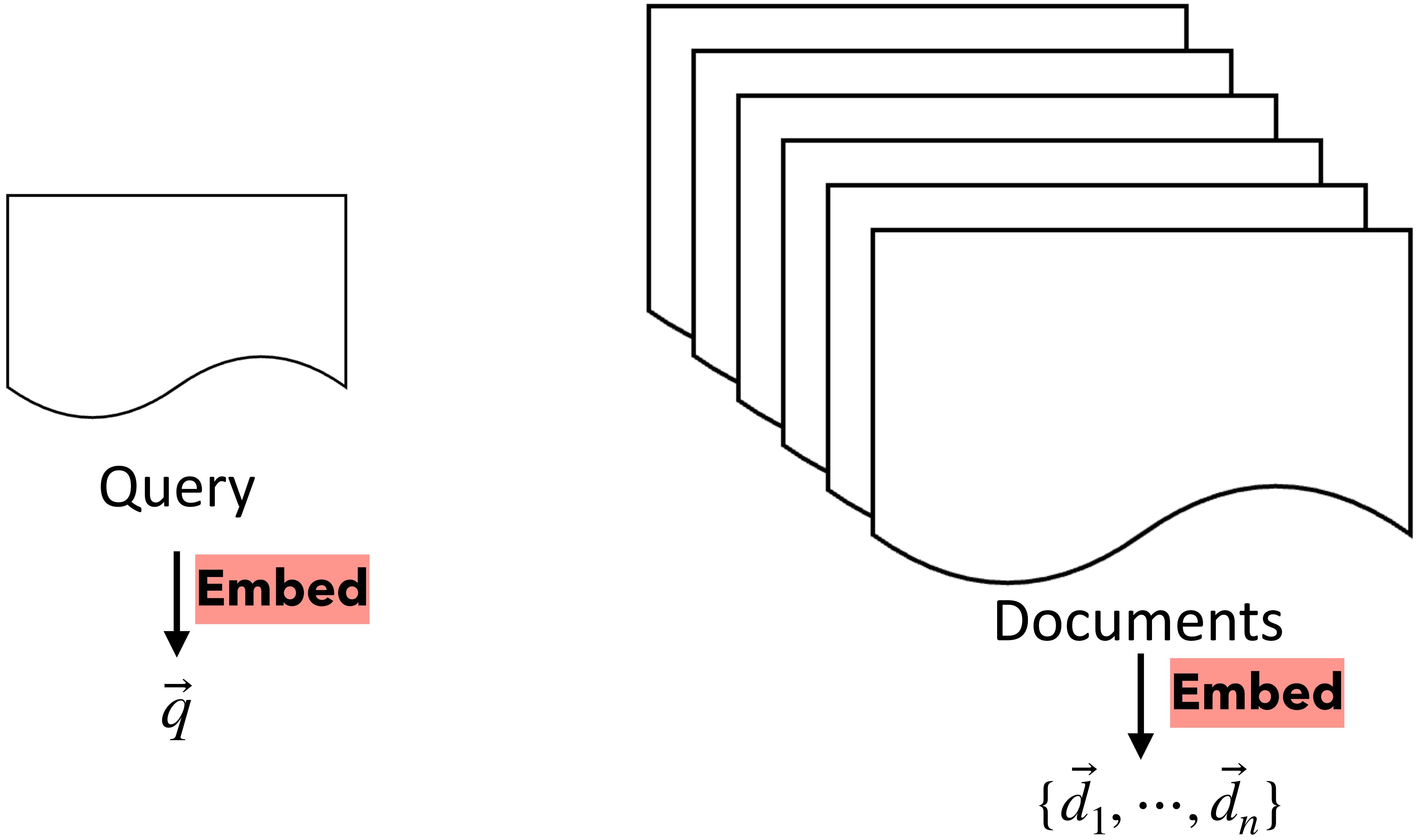
Encoders for Information Retrieval

Retrieve the set of relevant documents given a query



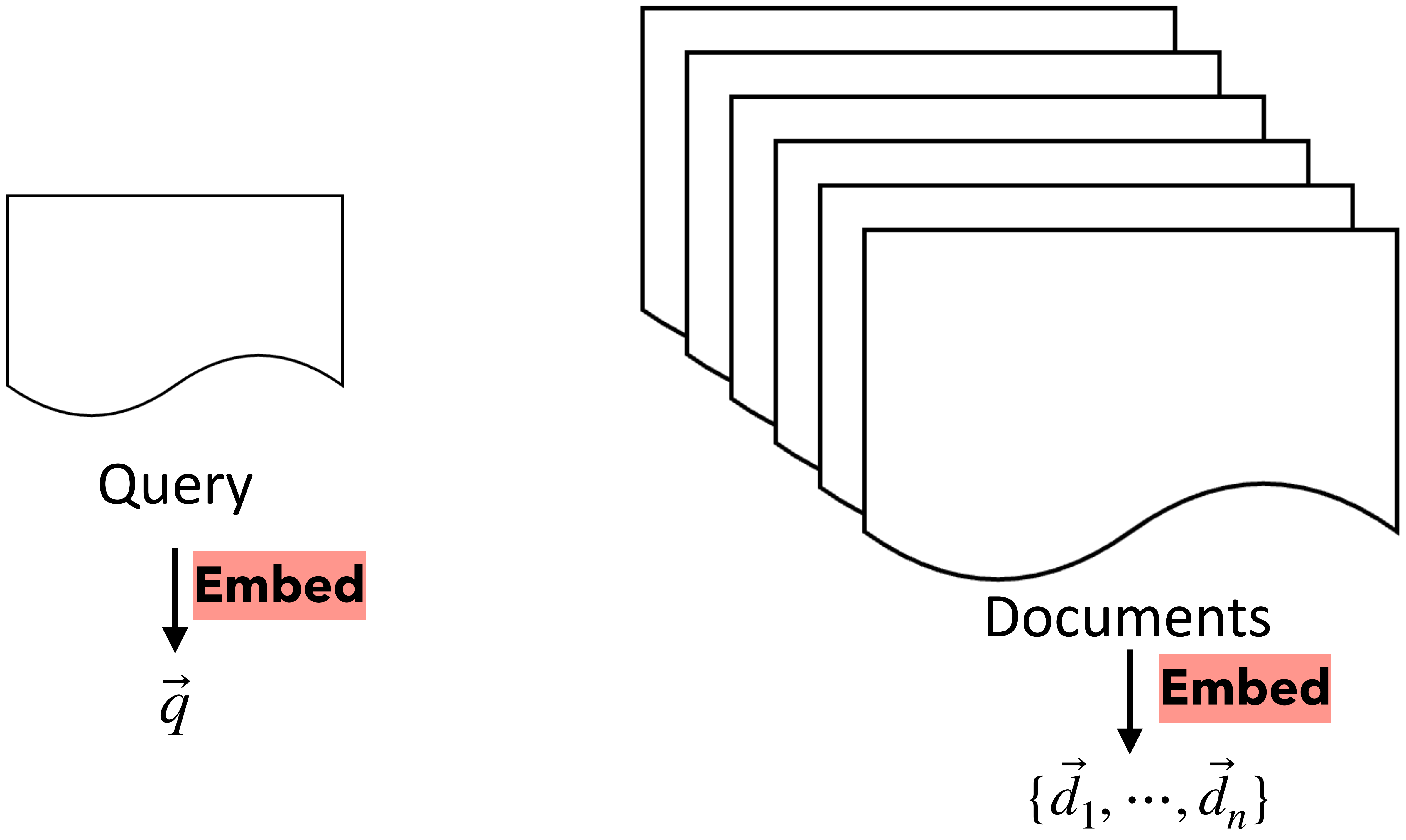
Encoders for Information Retrieval

Retrieve the set of relevant documents given a query



Encoders for Information Retrieval

Retrieve the set of relevant documents given a query

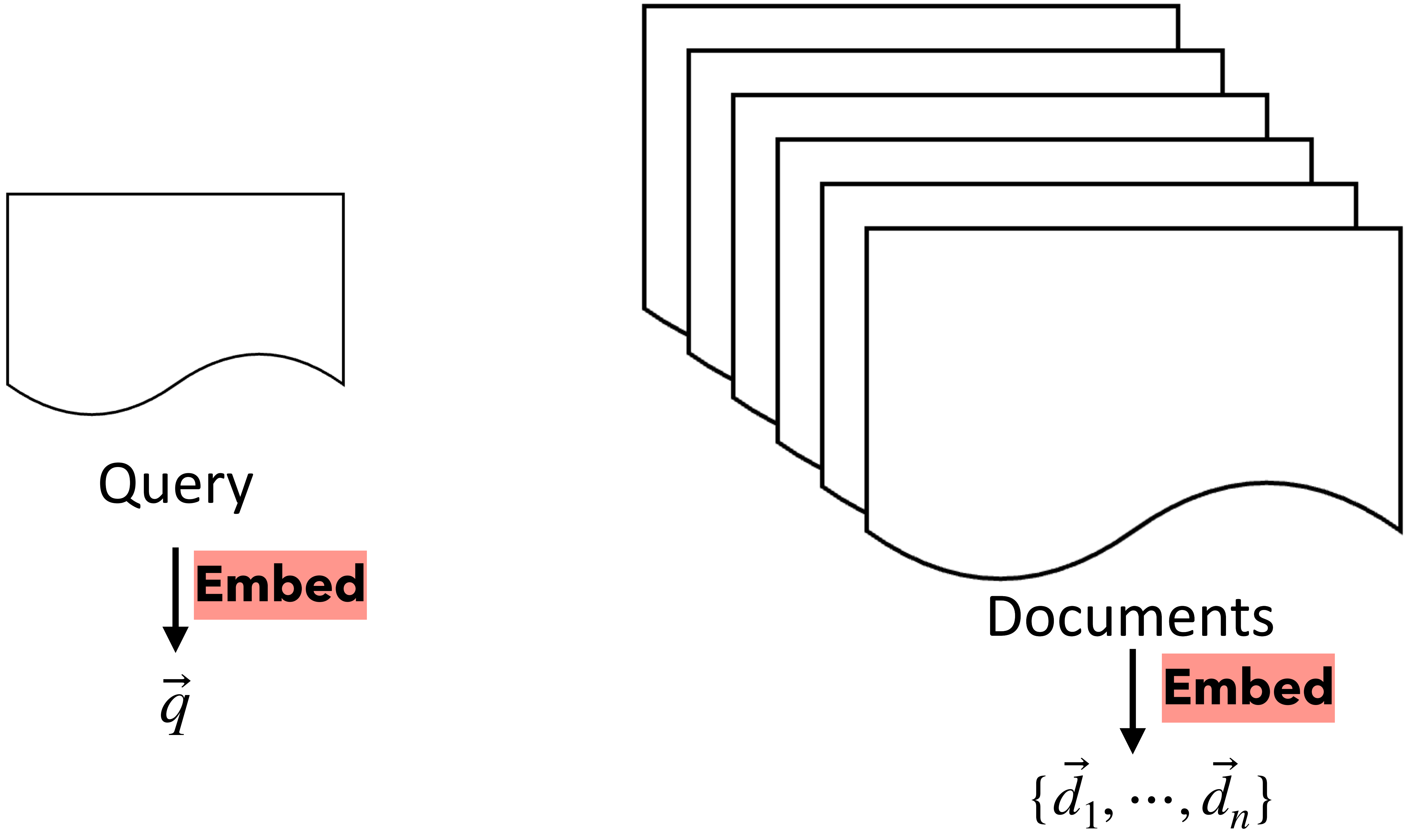


Score document relevance by, e.g., computing cosine similarity between the query and the document

relevance-score$(d \mid q) = \cos(\hat{q}, \hat{d})$

Encoders for Information Retrieval

Retrieve the set of relevant documents given a query



HW2!

Score document relevance by, e.g., computing cosine similarity between the query and the document

relevance-score$(d \mid q) = \cos(\hat{q}, \hat{d})$

Encoders for Information Retrieval

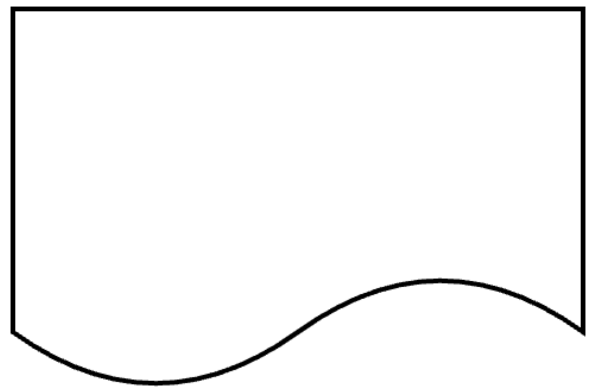
Retrieve the set of relevant documents given a query

Applications:
- **Search Engines (This is how google works!)**
- **Retrieval Augmented Language Models**

HW2!

Score document relevance by, e.g., computing cosine similarity between the query and the document

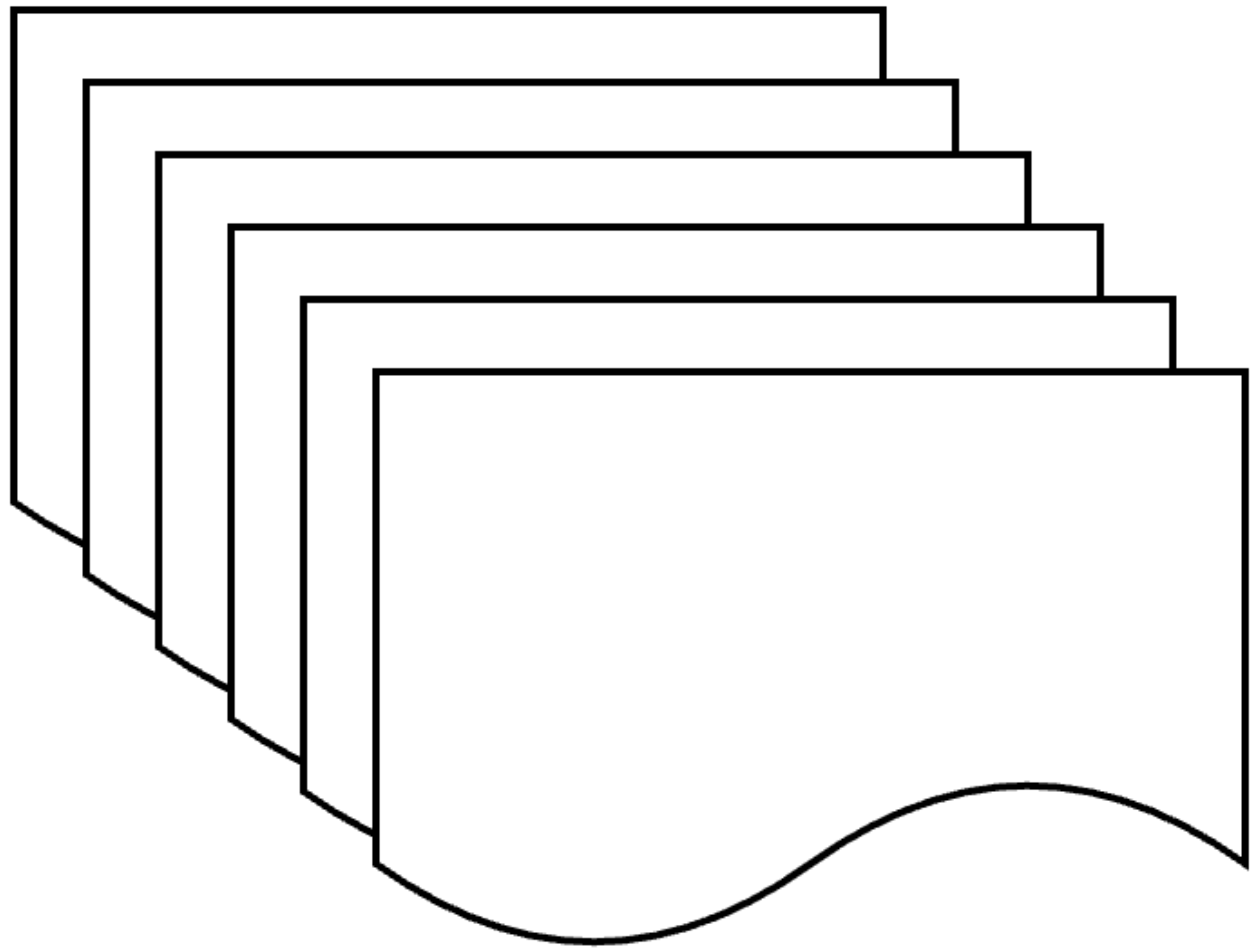
relevance-score$(d \mid q) = \cos(\hat{q}, \hat{d})$



Query

Embed

$\vec{q}$



Documents

Embed

$\{\vec{d}_1, \cdots, \vec{d}_n\}$

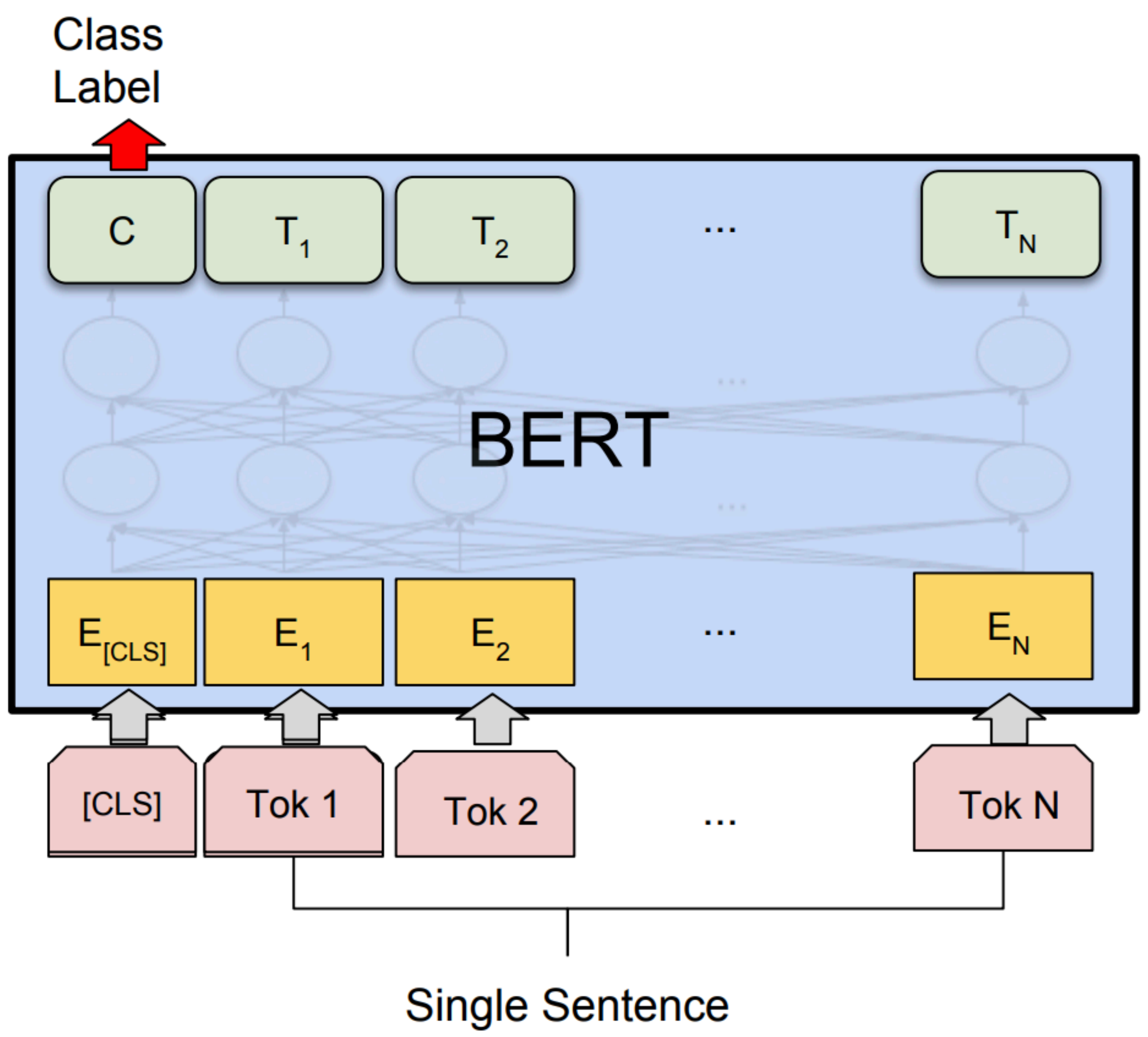
Encoders for Information Retrieval

Encoders for Information Retrieval

How do we get sentence embeddings from an encoder-based model like BERT?

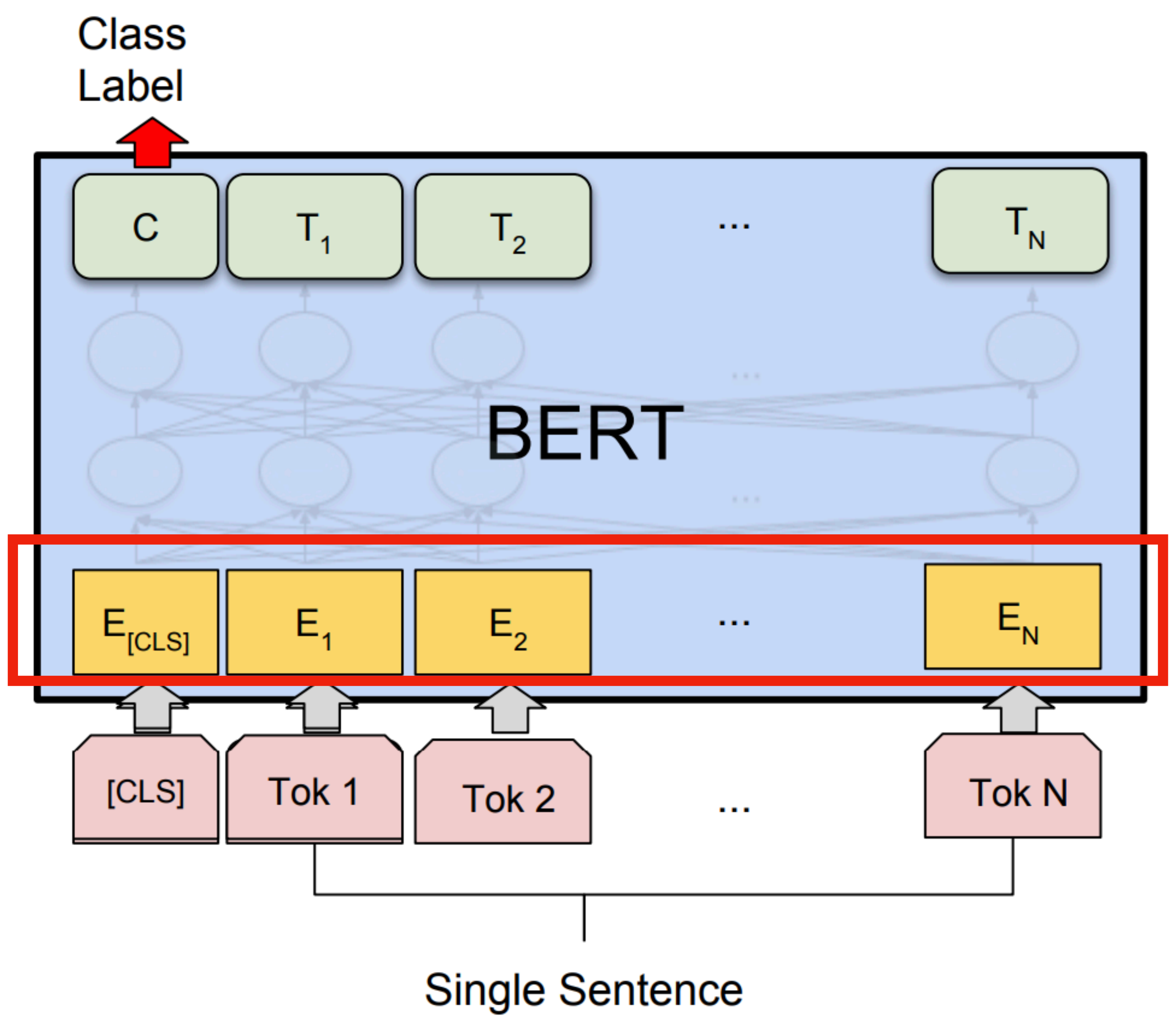
Encoders for Information Retrieval

How do we get sentence embeddings from an encoder-based model like BERT?



Encoders for Information Retrieval

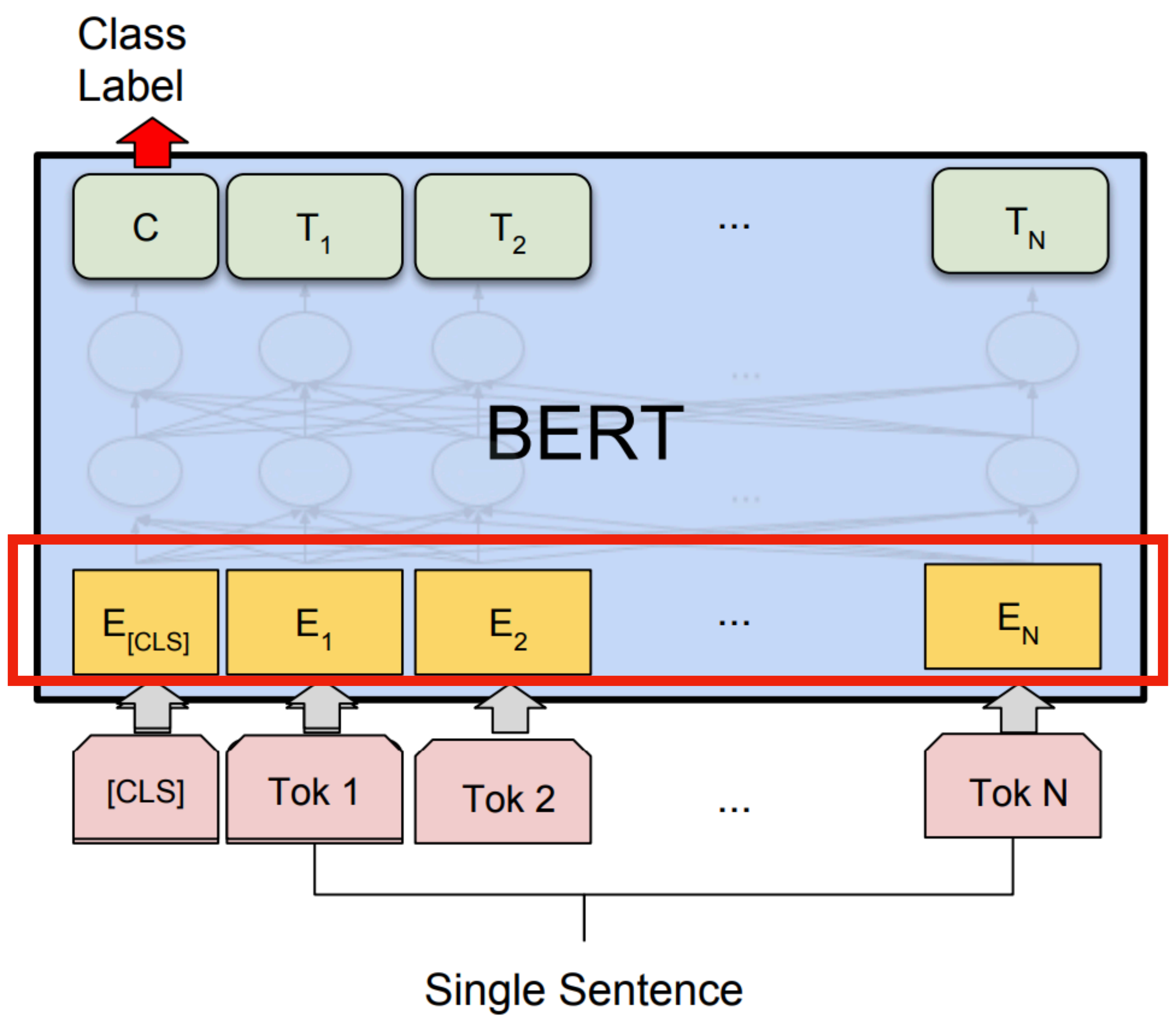
How do we get sentence embeddings from an encoder-based model like BERT?



Option 1: Average learned word embeddings

Encoders for Information Retrieval

How do we get sentence embeddings from an encoder-based model like BERT?

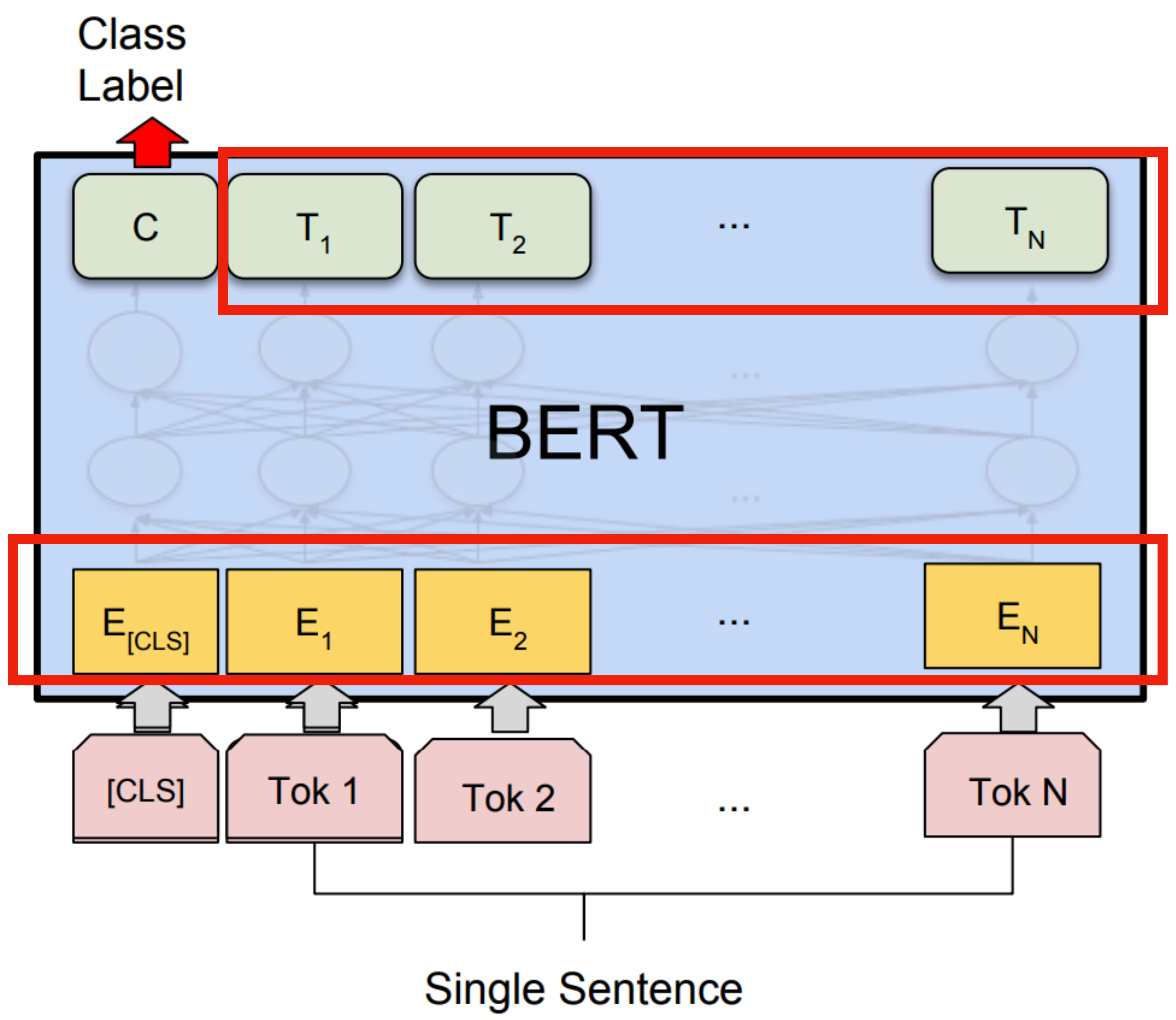


Option 1: Average learned word embeddings

Problem:
Representations not contextual!
Equivalent to using GloVe vectors

Encoders for Information Retrieval

How do we get sentence embeddings from an encoder-based model like BERT?



Option 2: Average learned **contexual** word embeddings

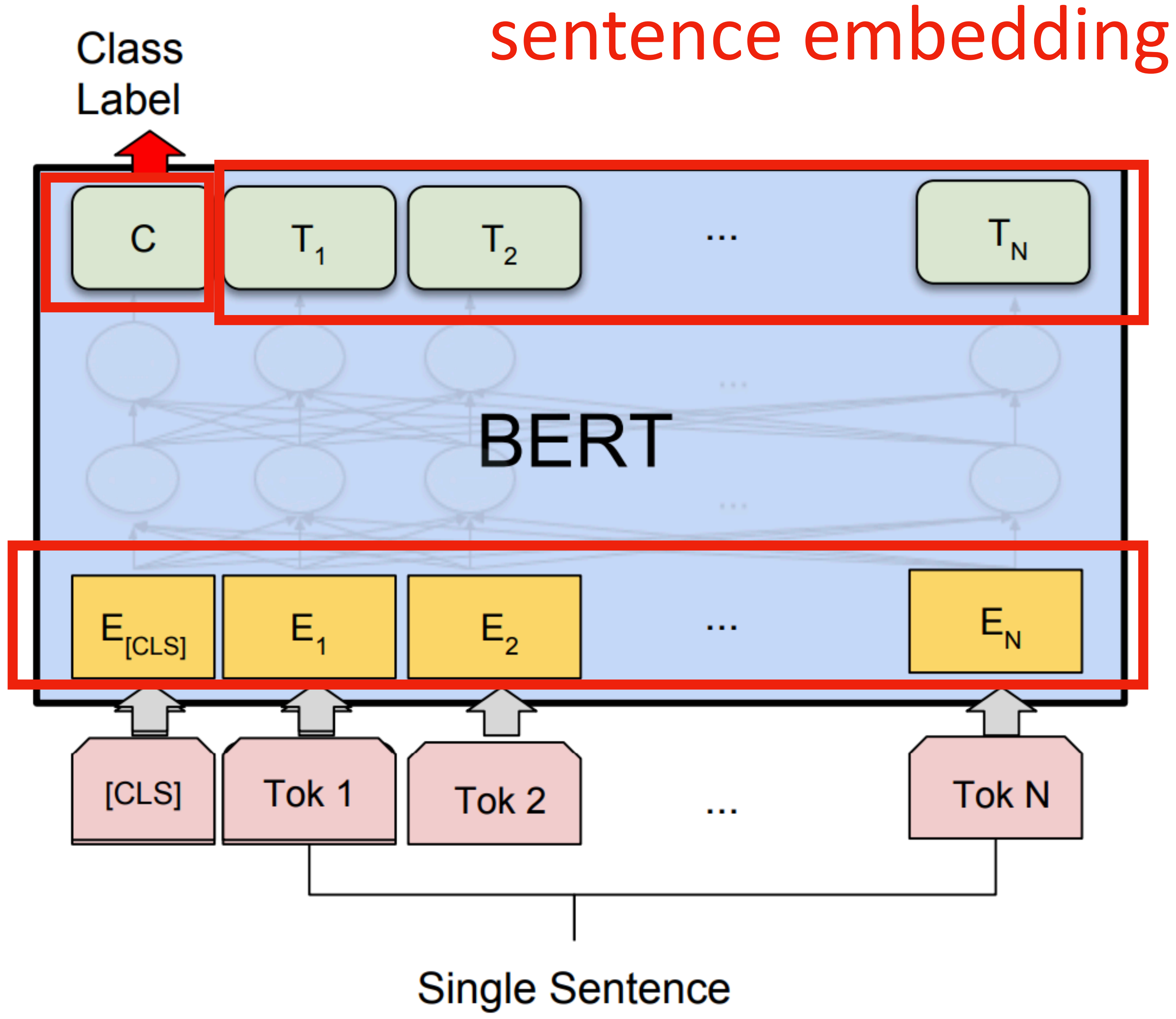
Option 1: Average learned word embeddings

Problem:
Representations not contextual!
Equivalent to using GloVe vectors

Encoders for Information Retrieval

How do we get sentence embeddings from an encoder-based model like BERT?

Option 3: Use representations of CLS token for sentence embedding

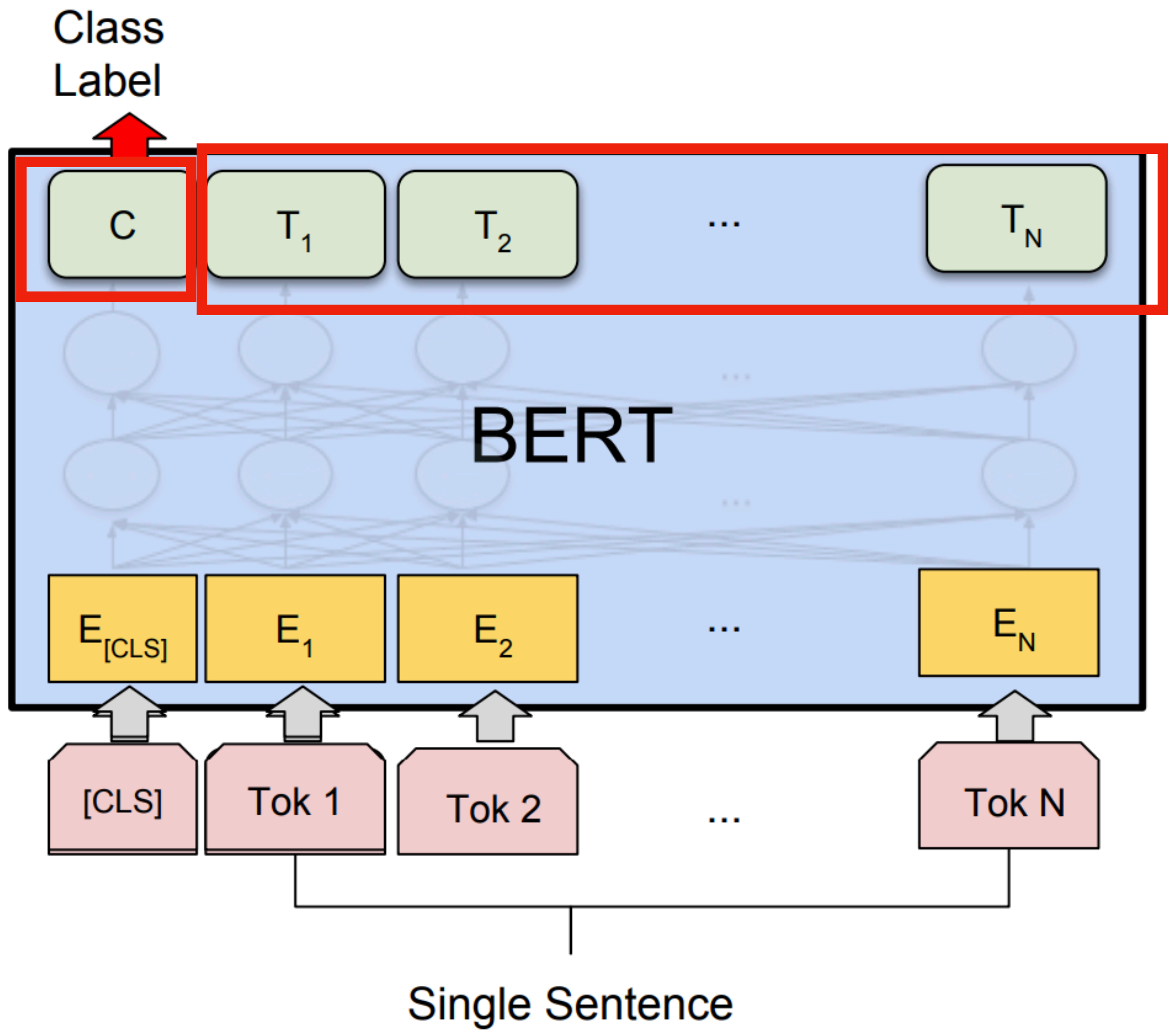


Option 2: Average learned **contexual** word embeddings

Option 1: Average learned word embeddings

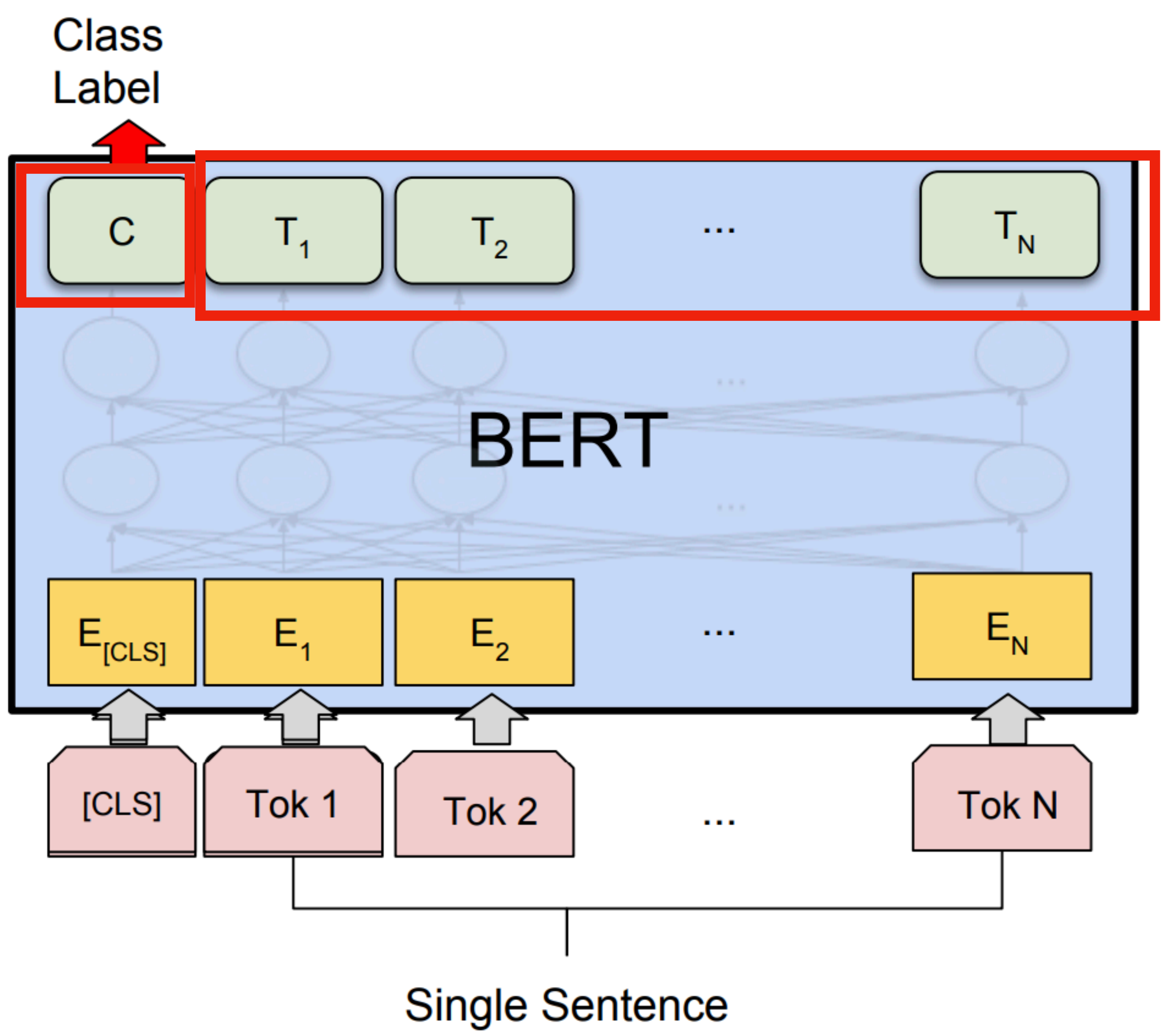
Problem:
Representations not contextual!
Equivalent to using GloVe vectors

Encoders for Information Retrieval



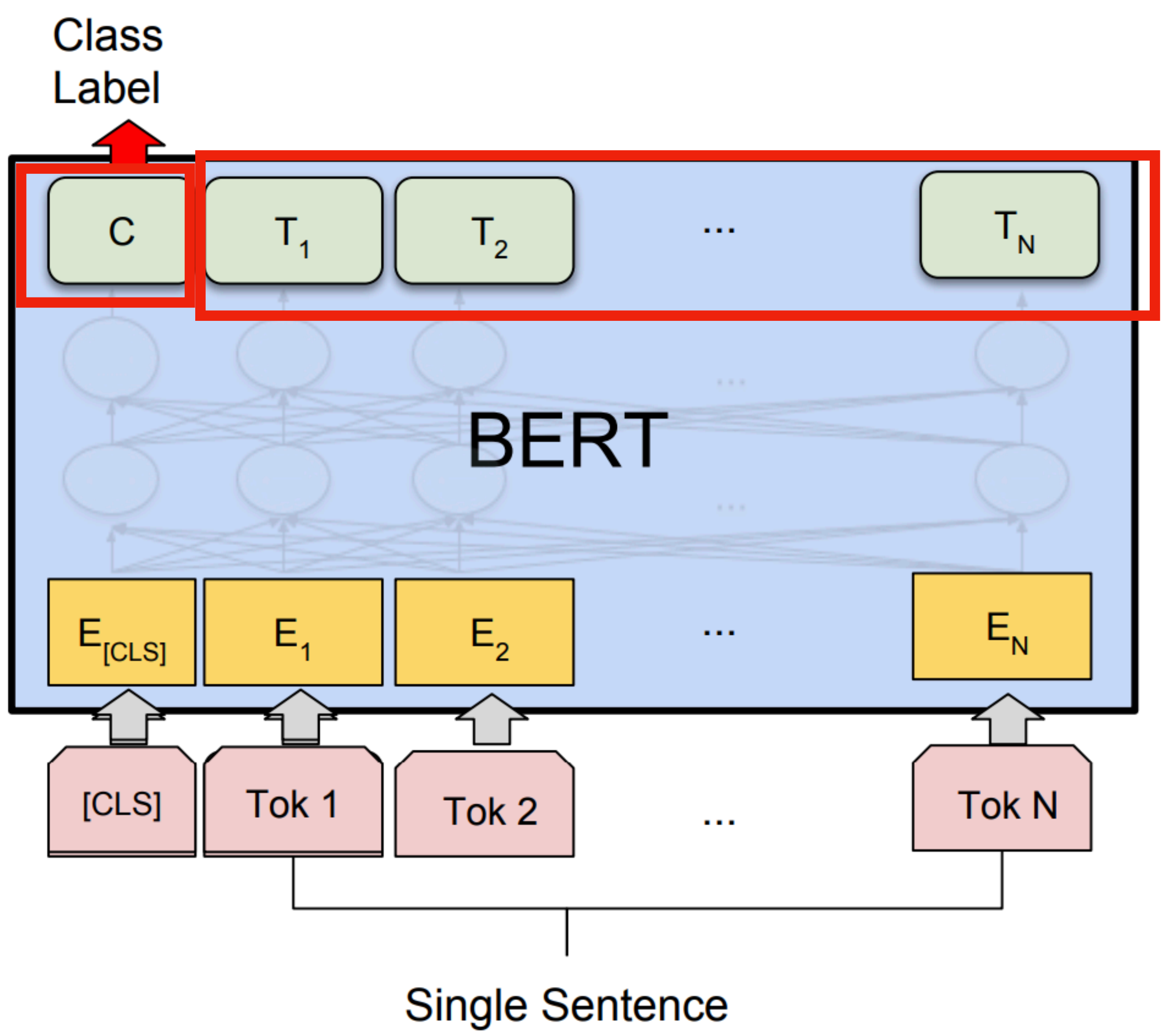
Encoders for Information Retrieval

Out of the box even contextual representations are not very good for retrieval!



Encoders for Information Retrieval

Out of the box even contextual representations are not very good for retrieval!

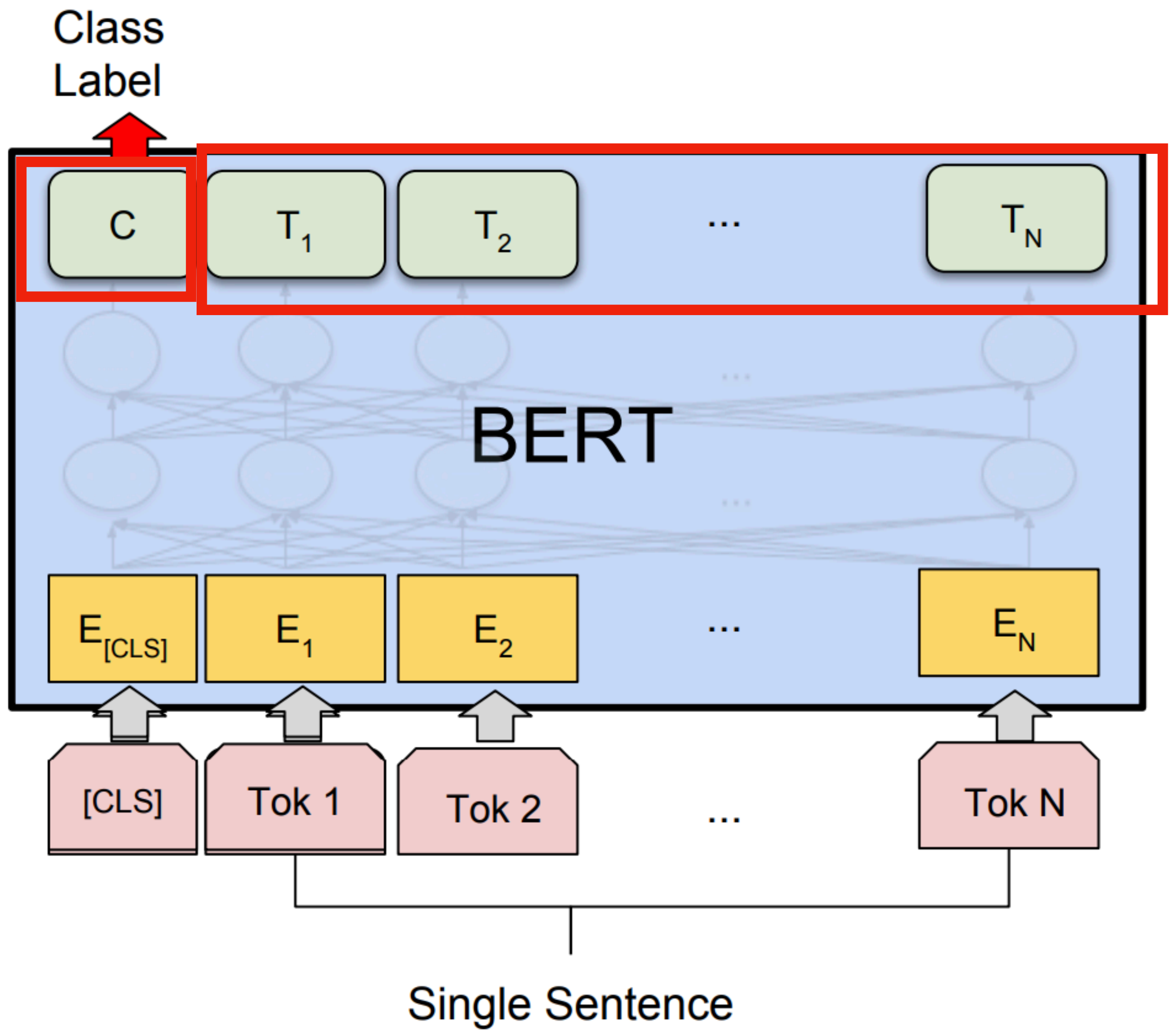


Model	STS12	STS13	STS14	STS15	STS16	STSb	SICK-R	Avg.
Avg. GloVe embeddings	55.14	70.66	59.73	68.25	63.66	58.02	53.76	61.32
Avg. BERT embeddings	38.78	57.98	57.98	63.15	61.06	46.35	58.40	54.81
BERT CLS-vector	20.16	30.01	20.09	36.88	38.08	16.50	42.63	29.19
InferSent - Glove	52.86	66.75	62.15	72.77	66.87	68.03	65.65	65.01
Universal Sentence Encoder	64.49	67.80	64.61	76.83	73.18	74.92	**76.69**	71.22

Spearman correlations for Textual Similarity (STS) tasks
(higher is better)

Encoders for Information Retrieval

Out of the box even contextual representations are not very good for retrieval!

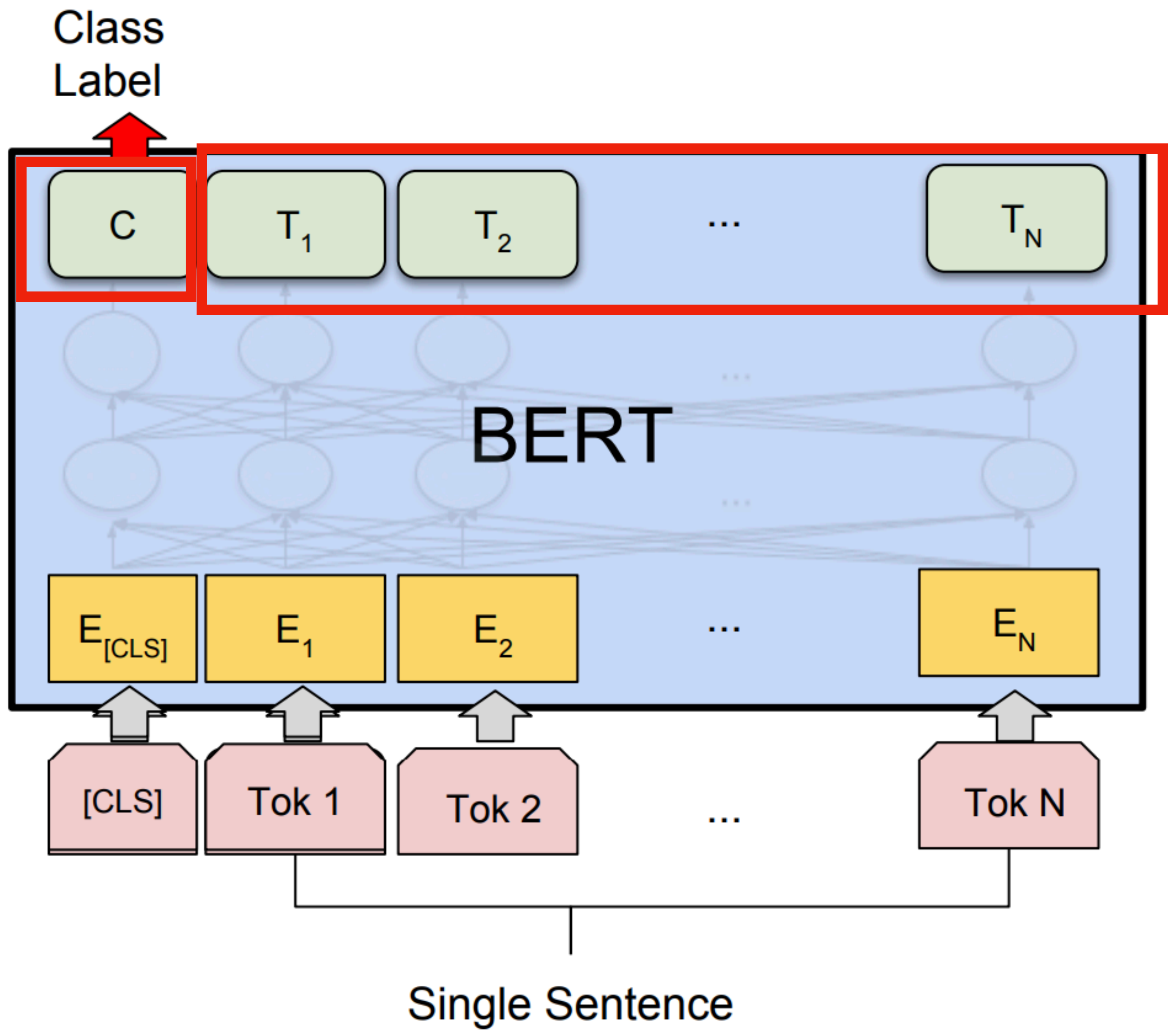


	Model	STS12	STS13	STS14	STS15	STS16	STSb	SICK-R	Avg.
	Avg. GloVe embeddings	55.14	70.66	59.73	68.25	63.66	58.02	53.76	61.32
Option 2	Avg. BERT embeddings	38.78	57.98	57.98	63.15	61.06	46.35	58.40	54.81
Option 3	BERT CLS-vector	20.16	30.01	20.09	36.88	38.08	16.50	42.63	29.19
	InferSent - Glove	52.86	66.75	62.15	72.77	66.87	68.03	65.65	65.01
	Universal Sentence Encoder	64.49	67.80	64.61	76.83	73.18	74.92	**76.69**	71.22

Spearman correlations for Textual Similarity (STS) tasks
(higher is better)

Encoders for Information Retrieval

Out of the box even contextual representations are not very good for retrieval!

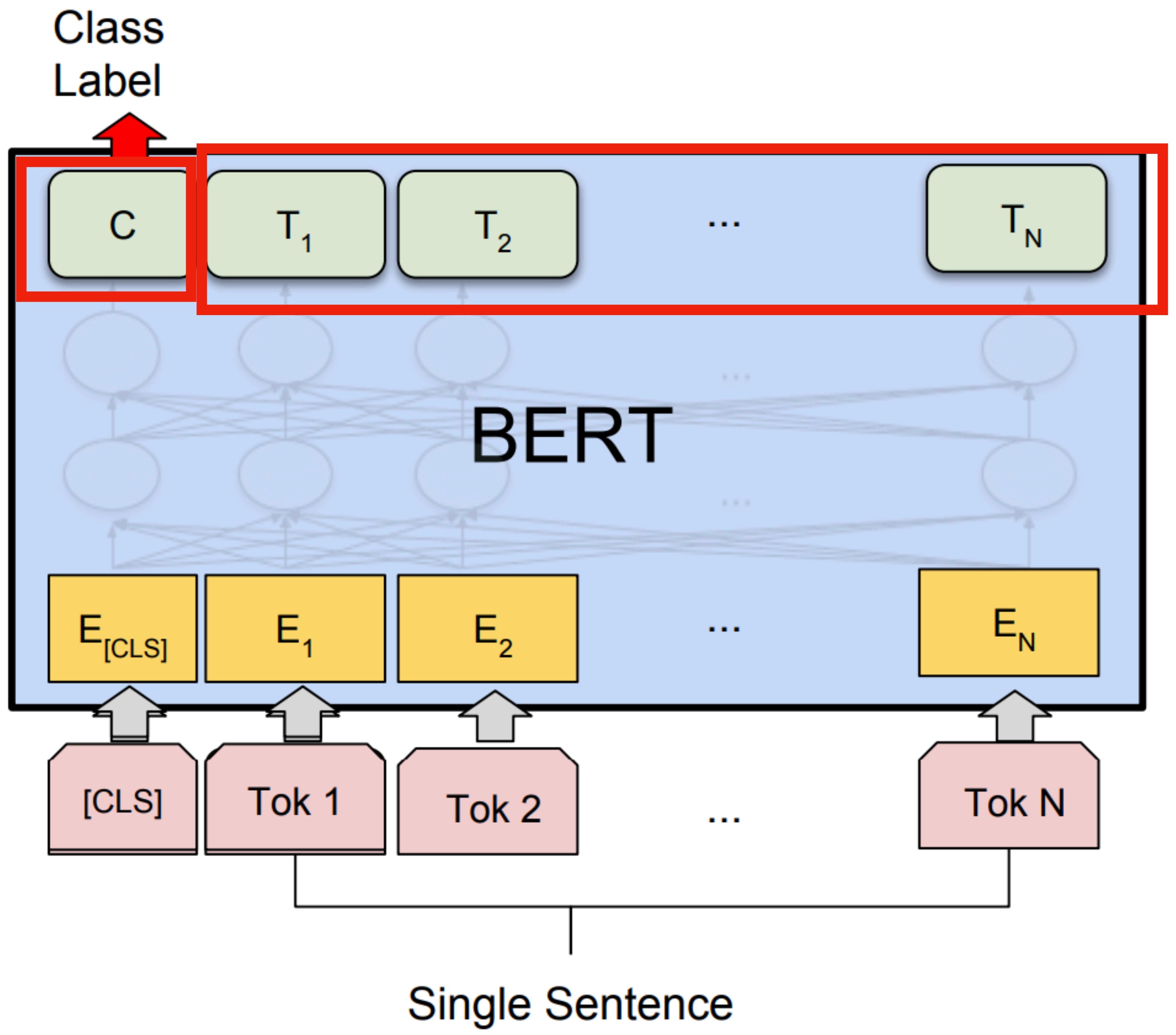


Model	STS12	STS13	STS14	STS15	STS16	STSb	SICK-R	Avg.
Avg. GloVe embeddings	55.14	70.66	59.73	68.25	63.66	58.02	53.76	61.32
Option 2 Avg. BERT embeddings	38.78	57.98	57.98	63.15	61.06	46.35	58.40	54.81
Option 3 BERT CLS-vector	20.16	30.01	20.09	36.88	38.08	16.50	42.63	29.19
InferSent - Glove	52.86	66.75	62.15	72.77	66.87	68.03	65.65	65.01
Universal Sentence Encoder	64.49	67.80	64.61	76.83	73.18	74.92	**76.69**	71.22

Spearman correlations for Textual Similarity (STS) tasks
(higher is better)

Encoders for Information Retrieval

Out of the box even contextual representations are not very good for retrieval!



Performance is even worse than averaging word embeddings!

	Model	STS12	STS13	STS14	STS15	STS16	STSb	SICK-R	Avg.
	Avg. GloVe embeddings	55.14	70.66	59.73	68.25	63.66	58.02	53.76	61.32
Option 2	Avg. BERT embeddings	38.78	57.98	57.98	63.15	61.06	46.35	58.40	$\boxed{54.81}$
Option 3	BERT CLS-vector	20.16	30.01	20.09	36.88	38.08	16.50	42.63	$\boxed{29.19}$
	InferSent - Glove	52.86	66.75	62.15	72.77	66.87	68.03	65.65	65.01
	Universal Sentence Encoder	64.49	67.80	64.61	76.83	73.18	74.92	**76.69**	71.22

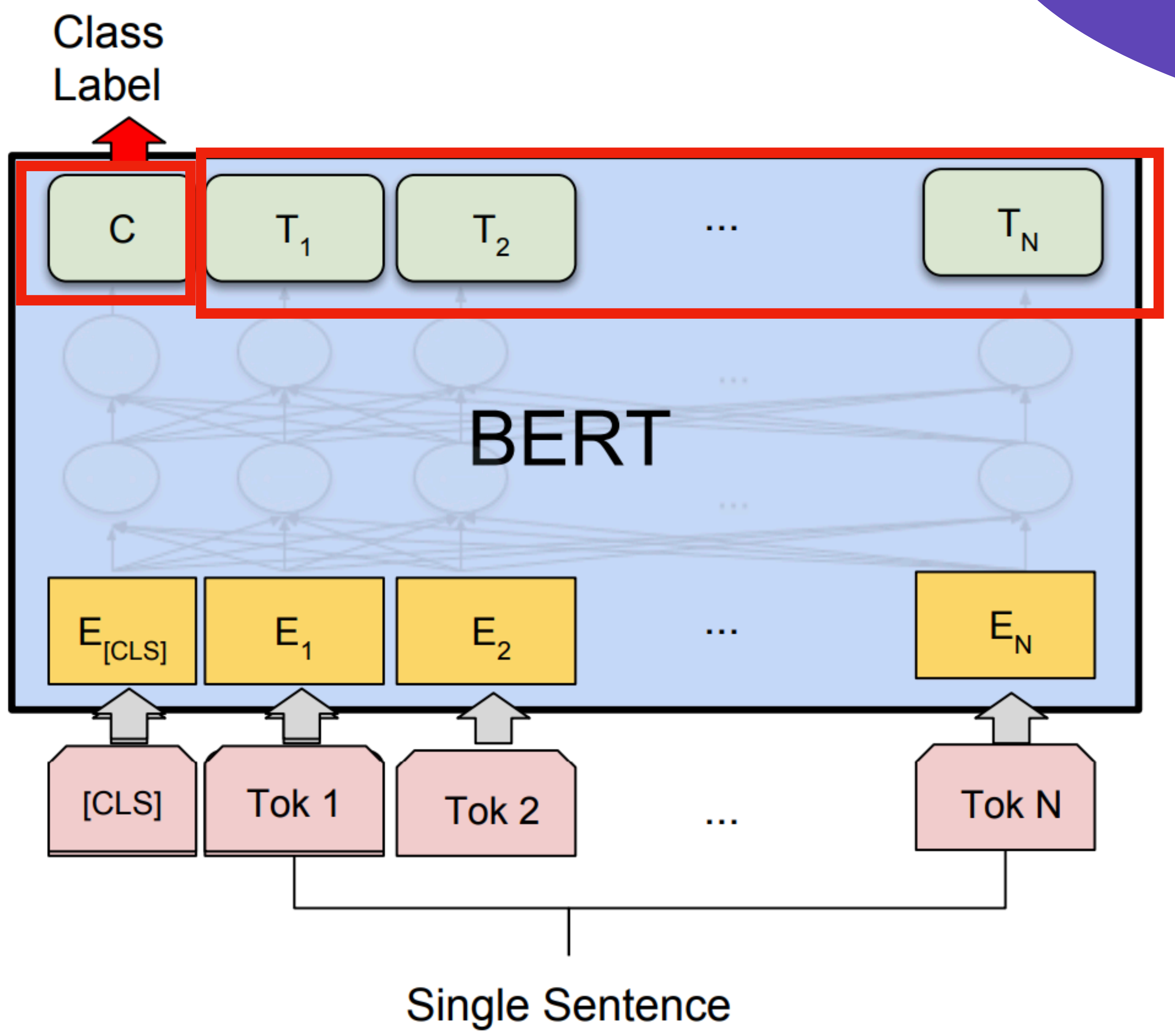
Spearman correlations for Textual Similarity (STS) tasks
(higher is better)

Encoders for Information Retrieval

Out of the box even contextual representations are not very good for retrieval!

Why?

Performance is even worse than averaging word embeddings!



Model	STS12	STS13	STS14	STS15	STS16	STSb	SI[illegible]-R	Avg.
Avg. GloVe embeddings	55.14	70.66	59.73	68.25	63.66	58.02	53.7[illegible]	61.32
Option 2 Avg. BERT embeddings	38.78	57.98	57.98	63.15	61.06	46.35	58.40	54.81
Option 3 BERT CLS-vector	20.16	30.01	20.09	36.88	38.08	16.50	42.63	29.19
InferSent - Glove	52.86	66.75	62.15	72.77	66.87	68.03	65.65	65.01
Universal Sentence Encoder	64.49	67.80	64.61	76.83	73.18	74.92	**76.69**	71.22

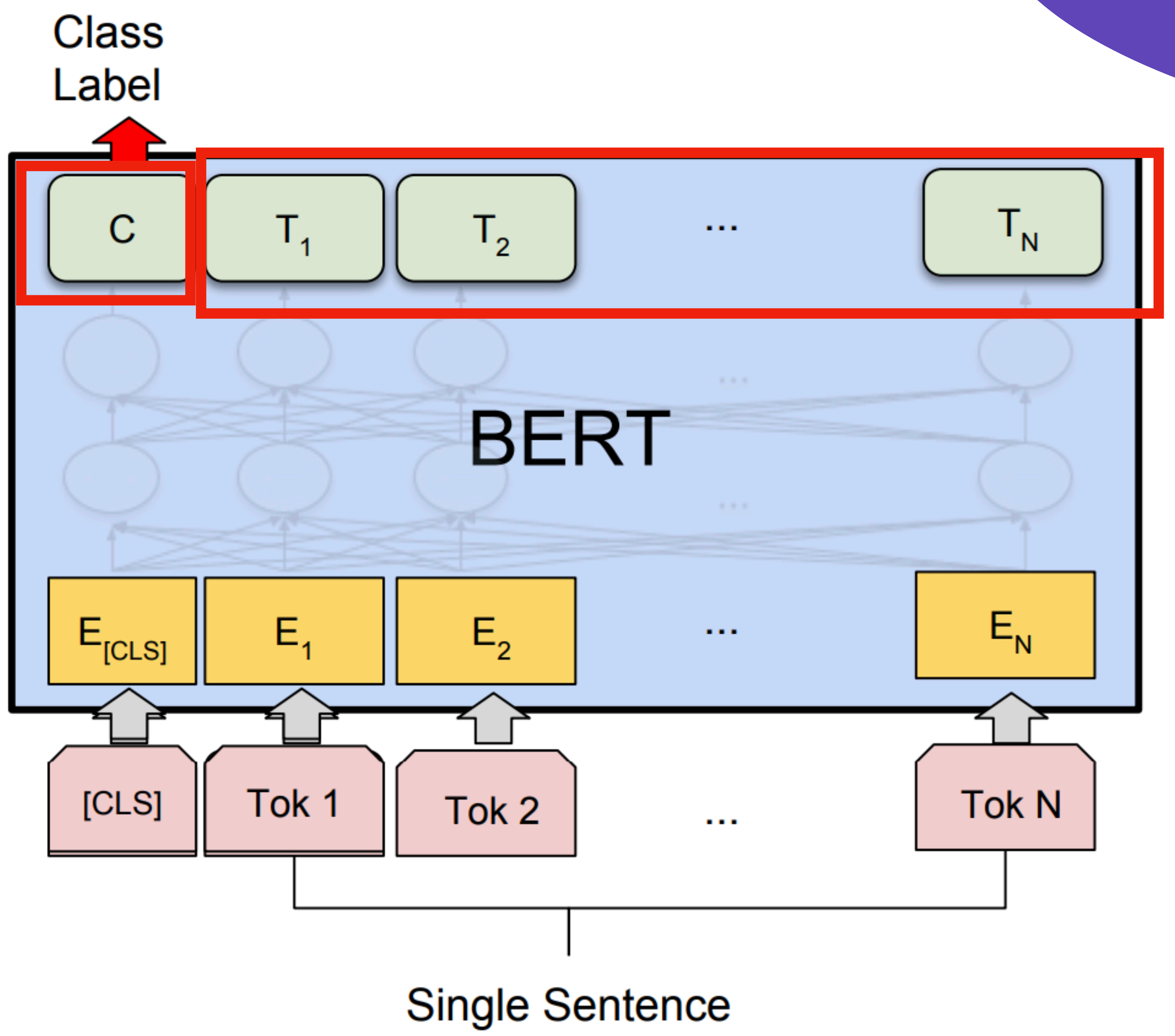
Spearman correlations for Textual Similarity (STS) tasks (higher is better)

Encoders for Information Retrieval

Out of the box even contextual representations are not very good for retrieval!

Why?
Pre-training objective unlike for word2vec is not aligned with the objective of placing similar sentences closer in embedding space

Performance is even worse than averaging word embeddings!



	Model	STS12	STS13	STS14	STS15	STS16	STSb	SICK-R	Avg.
	Avg. GloVe embeddings	55.14	70.66	59.73	68.25	63.66	58.02	53.76	61.32
Option 2	Avg. BERT embeddings	38.78	57.98	57.98	63.15	61.06	46.35	58.40	54.81
Option 3	BERT CLS-vector	20.16	30.01	20.09	36.88	38.08	16.50	42.63	29.19
	InferSent - Glove	52.86	66.75	62.15	72.77	66.87	68.03	65.65	65.01
	Universal Sentence Encoder	64.49	67.80	64.61	76.83	73.18	74.92	**76.69**	71.22

Spearman correlations for Textual Similarity (STS) tasks
(higher is better)

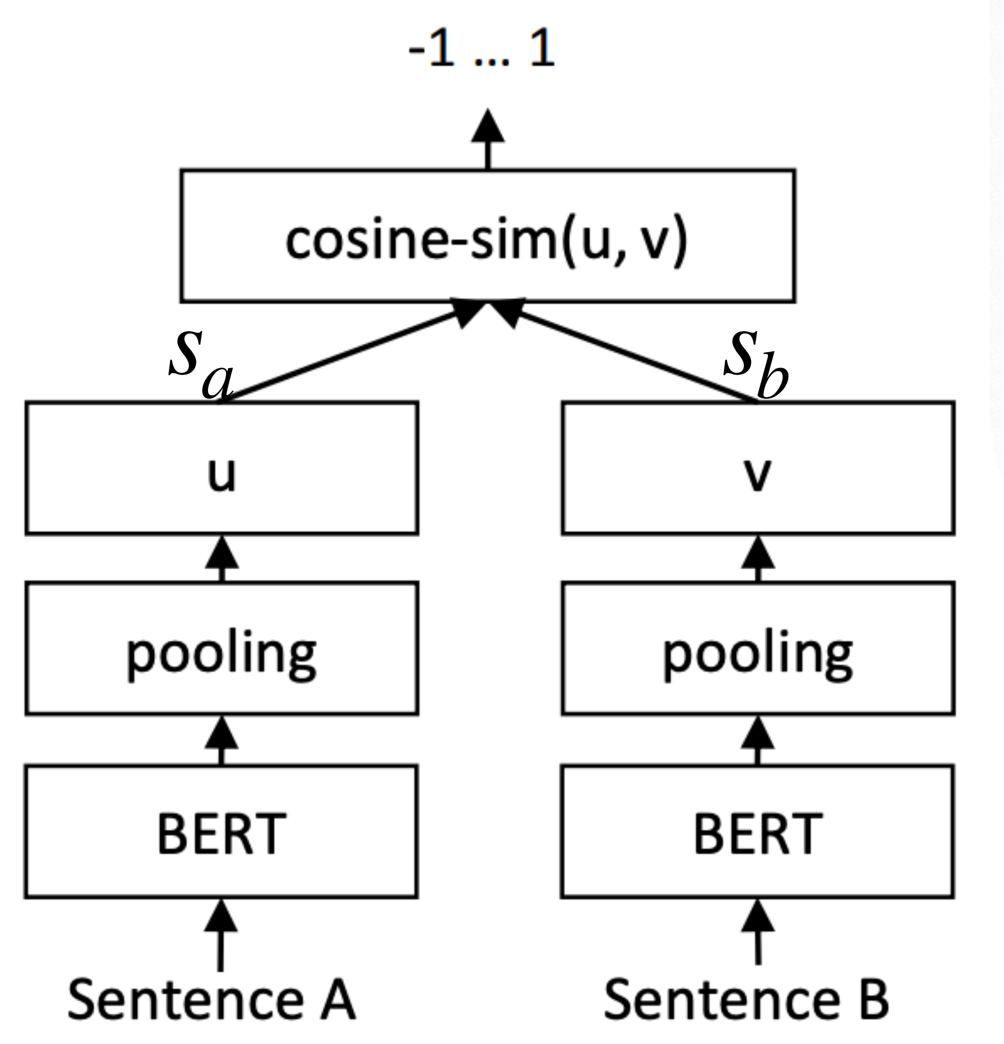
Encoders for Information Retrieval: Sentence BERT (S-BERT)

Encoders for Information Retrieval: Sentence BERT (S-BERT)

- Finetune BERT / RoBERTa to learn sentence level representations such that similar sentences are located closer in the embedding space

Encoders for Information Retrieval: Sentence BERT (S-BERT)

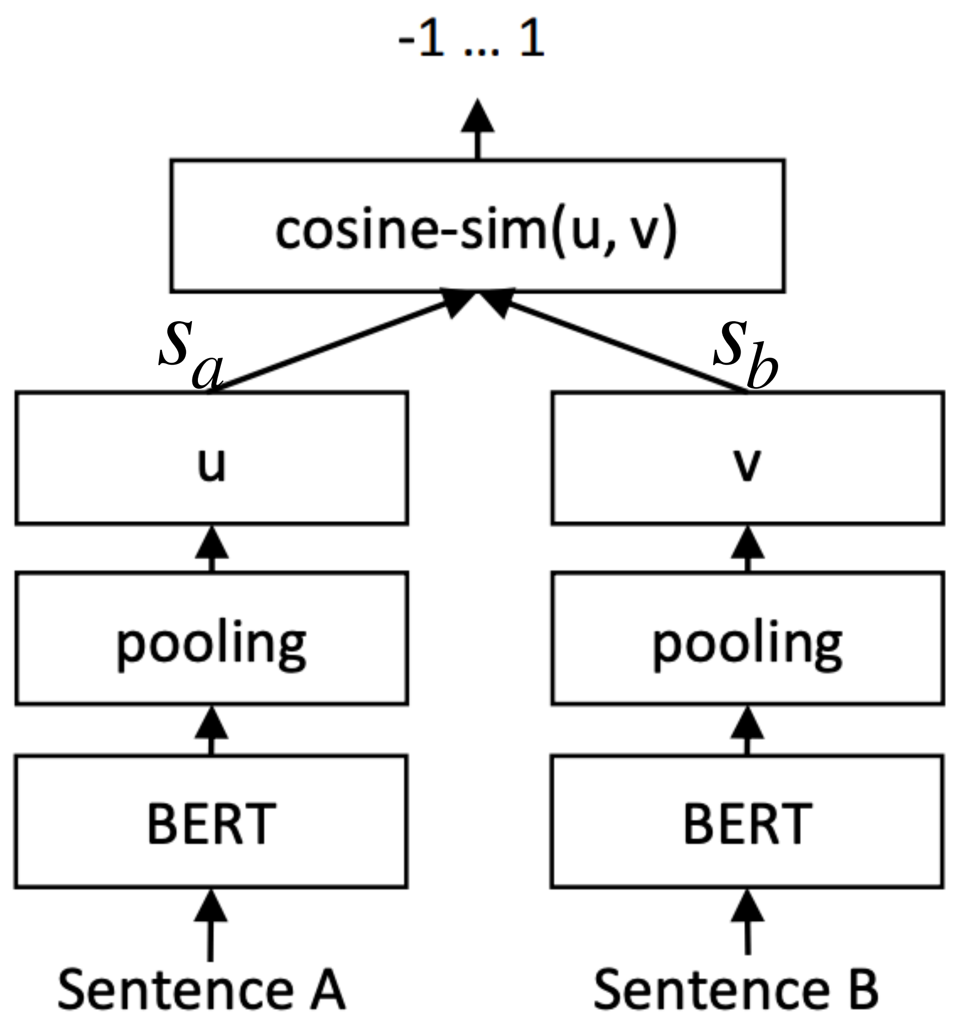
- Finetune BERT / RoBERTa to learn sentence level representations such that similar sentences are located closer in the embedding space



Encoders for Information Retrieval: Sentence BERT (S-BERT)

- Finetune BERT / RoBERTa to learn sentence level representations such that similar sentences are located closer in the embedding space

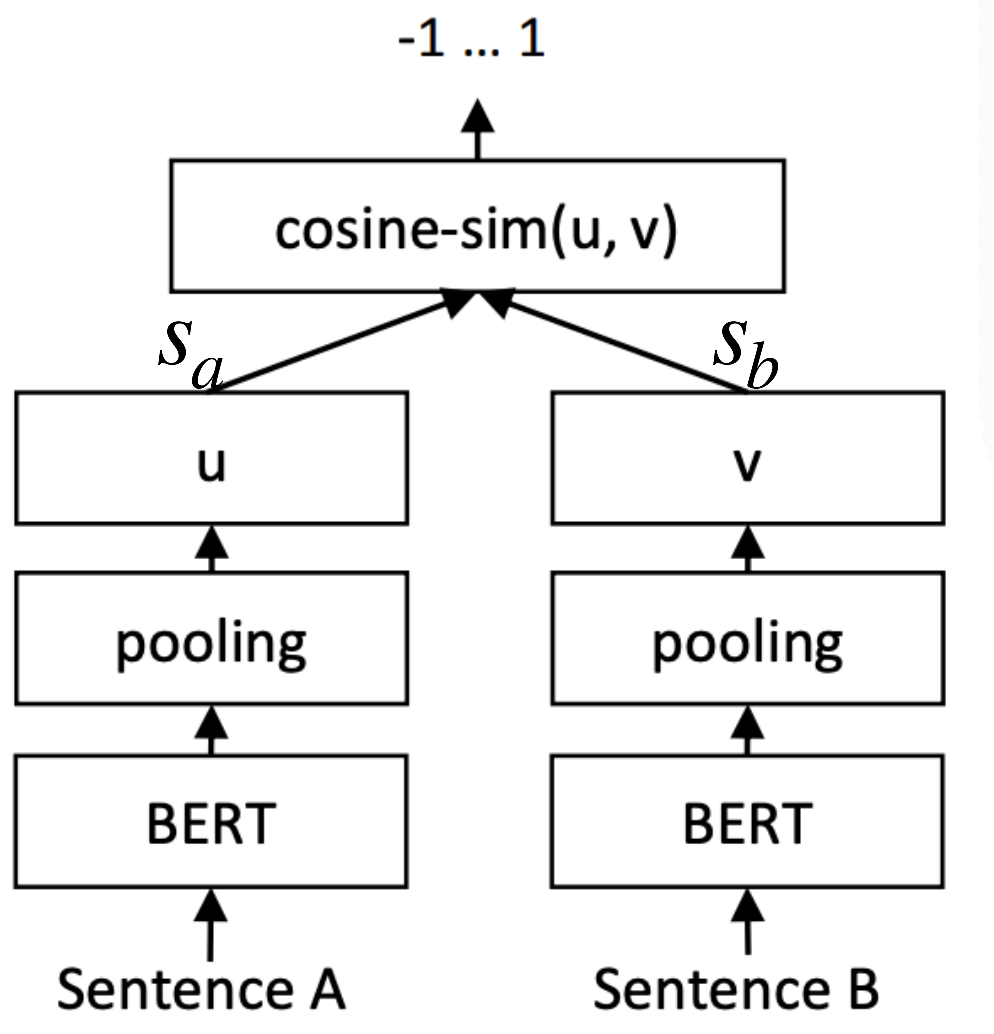
- Uses a triplet objective function – Given an anchor sentence a, a positive sentence p, and a negative sentence n, triplet loss tunes the network such that the distance between a and p is smaller than the distance between a and n.



Encoders for Information Retrieval: Sentence BERT (S-BERT)

- Finetune BERT / RoBERTa to learn sentence level representations such that similar sentences are located closer in the embedding space

- Uses a triplet objective function – Given an anchor sentence a, a positive sentence p, and a negative sentence n, triplet loss tunes the network such that the distance between a and p is smaller than the distance between a and n.



Triplet objective function

$$\max(\|s_a - s_p\| - \|s_a - s_n\| + \epsilon, 0)$$

Encoders for Information Retrieval: Sentence BERT (S-BERT)

Encoders for Information Retrieval: Sentence BERT (S-BERT)

Model	STS12	STS13	STS14	STS15	STS16	STSb	SICK-R	Avg.
Avg. GloVe embeddings	55.14	70.66	59.73	68.25	63.66	58.02	53.76	61.32
Avg. BERT embeddings	38.78	57.98	57.98	63.15	61.06	46.35	58.40	54.81
BERT CLS-vector	20.16	30.01	20.09	36.88	38.08	16.50	42.63	29.19
InferSent - Glove	52.86	66.75	62.15	72.77	66.87	68.03	65.65	65.01
Universal Sentence Encoder	64.49	67.80	64.61	76.83	73.18	74.92	**76.69**	71.22
SBERT-NLI-base	70.97	76.53	73.19	79.09	74.30	77.03	72.91	74.89
SBERT-NLI-large	72.27	**78.46**	**74.90**	80.99	76.25	**79.23**	73.75	76.55
SRoBERTa-NLI-base	71.54	72.49	70.80	78.74	73.69	77.77	74.46	74.21
SRoBERTa-NLI-large	**74.53**	77.00	73.18	**81.85**	**76.82**	79.10	74.29	**76.68**

Spearman correlations for Textual Similarity (STS) tasks
(higher is better)

Encoders for Information Retrieval: Sentence BERT (S-BERT)

Model	STS12	STS13	STS14	STS15	STS16	STSb	SICK-R	Avg.
Avg. GloVe embeddings	55.14	70.66	59.73	68.25	63.66	58.02	53.76	61.32
Avg. BERT embeddings	38.78	57.98	57.98	63.15	61.06	46.35	58.40	54.81
BERT CLS-vector	20.16	30.01	20.09	36.88	38.08	16.50	42.63	29.19
InferSent - Glove	52.86	66.75	62.15	72.77	66.87	68.03	65.65	65.01
Universal Sentence Encoder	64.49	67.80	64.61	76.83	73.18	74.92	**76.69**	71.22
SBERT-NLI-base	70.97	76.53	73.19	79.09	74.30	77.03	72.91	74.89
SBERT-NLI-large	72.27	**78.46**	**74.90**	80.99	76.25	**79.23**	73.75	76.55
SRoBERTa-NLI-base	71.54	72.49	70.80	78.74	73.69	77.77	74.46	74.21
SRoBERTa-NLI-large	**74.53**	77.00	73.18	**81.85**	**76.82**	79.10	74.29	**76.68**

Spearman correlations for Textual Similarity (STS) tasks
(higher is better)

Encoders for Information Retrieval: Sentence BERT (S-BERT)

Model	STS12	STS13	STS14	STS15	STS16	STSb	SICK-R	Avg.
Avg. GloVe embeddings	55.14	70.66	59.73	68.25	63.66	58.02	53.76	61.32
Avg. BERT embeddings	38.78	57.98	57.98	63.15	61.06	46.35	58.40	54.81
BERT CLS-vector	20.16	30.01	20.09	36.88	38.08	16.50	42.63	29.19
InferSent - Glove	52.86	66.75	62.15	72.77	66.87	68.03	65.65	65.01
Universal Sentence Encoder	64.49	67.80	64.61	76.83	73.18	74.92	**76.69**	71.22
SBERT-NLI-base	70.97	76.53	73.19	79.09	74.30	77.03	72.91	74.89
SBERT-NLI-large	72.27	**78.46**	**74.90**	80.99	76.25	**79.23**	73.75	76.55
SRoBERTa-NLI-base	71.54	72.49	70.80	78.74	73.69	77.77	74.46	74.21
SRoBERTa-NLI-large	**74.53**	77.00	73.18	**81.85**	**76.82**	79.10	74.29	**76.68**

Spearman correlations for Textual Similarity (STS) tasks
(higher is better)

Sentence-BERT / RoBERTa performs remarkably better than the existing approaches!

Encoders for Information Retrieval: Sentence BERT (S-BERT)

SBERT.net

Sentence Transformers Library. Very handy for using pre-trained Sentence-BERT-like models

Sentence-BERT / RoBERTa performs remarkably better than the existing approaches!

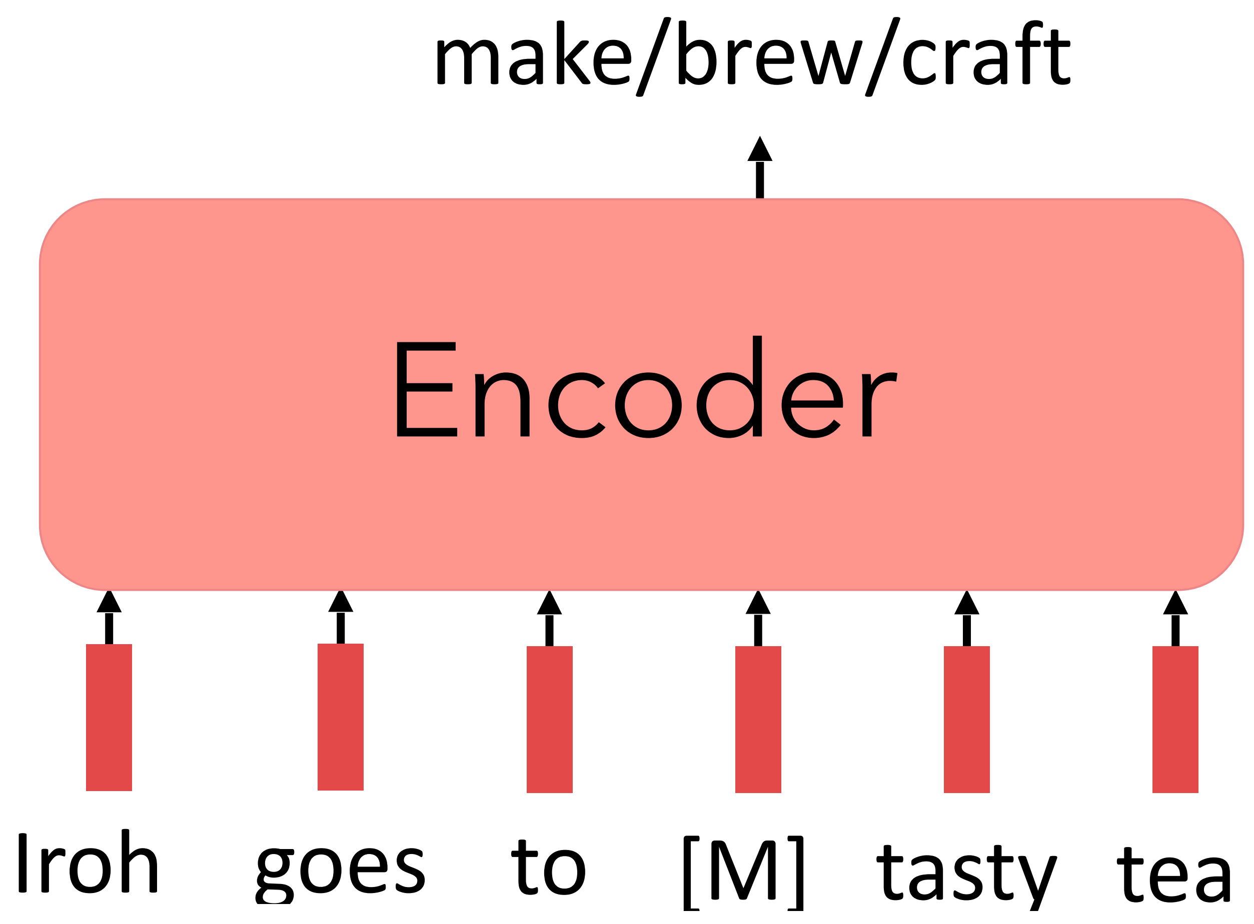
Model	STS12	STS13	STS14	STS15	STS16	STSb	SICK-R	Avg.
Avg. GloVe embeddings	55.14	70.66	59.73	68.25	63.66	58.02	53.76	61.32
Avg. BERT embeddings	38.78	57.98	57.98	63.15	61.06	46.35	58.40	54.81
BERT CLS-vector	20.16	30.01	20.09	36.88	38.08	16.50	42.63	29.19
InferSent - Glove	52.86	66.75	62.15	72.77	66.87	68.03	65.65	65.01
Universal Sentence Encoder	64.49	67.80	64.61	76.83	73.18	74.92	**76.69**	71.22
SBERT-NLI-base	70.97	76.53	73.19	79.09	74.30	77.03	72.91	74.89
SBERT-NLI-large	72.27	**78.46**	**74.90**	80.99	76.25	**79.23**	73.75	76.55
SRoBERTa-NLI-base	71.54	72.49	70.80	78.74	73.69	77.77	74.46	74.21
SRoBERTa-NLI-large	**74.53**	77.00	73.18	**81.85**	**76.82**	79.10	74.29	**76.68**

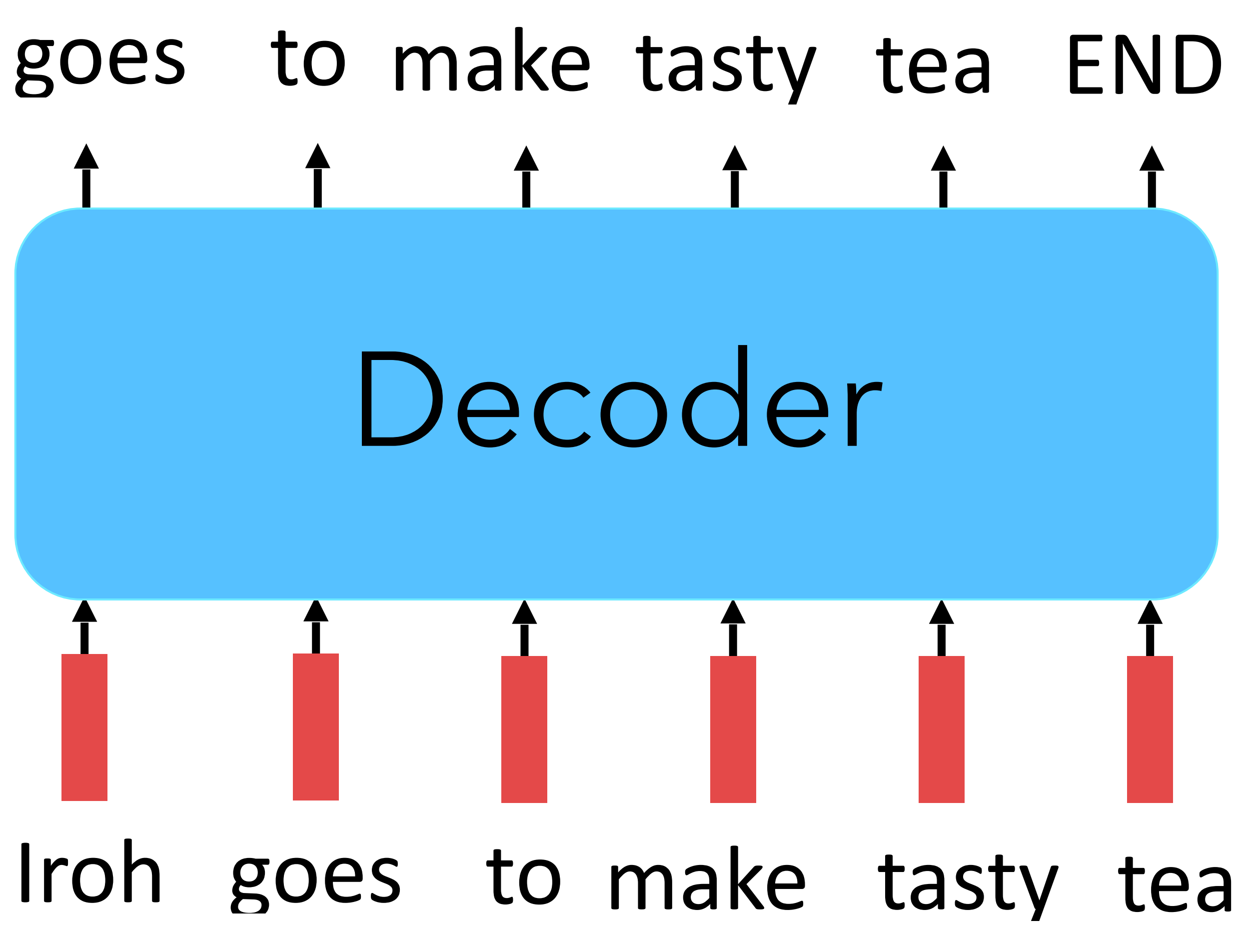
Spearman correlations for Textual Similarity (STS) tasks (higher is better)

Encoder: Pros & Cons

- Consider both left and right context
- Capture intricate contextual relationships

- Not good at generating open-text from left-to-right, one token at a time





Thank you!