BERT and Transfer Learning

Natural Language Processing — Lecture 7

Kenneth Enevoldsen | 2024

CENTER FOR
HUMANITIES
COMPUTING

Learning Goals

- Knowledge of different types of transformer architectures and how they are trained
  - Notably BERT and masked language modelling (MLM)
- An understanding of the relation between MLM and linguistic tasks
- An understanding of pre-training
  - And its influence on model performance
- The influence of scale on model performance
- An understanding of what language models learn
  - And how we can examine this using probing

Switching out class 9 and 10?

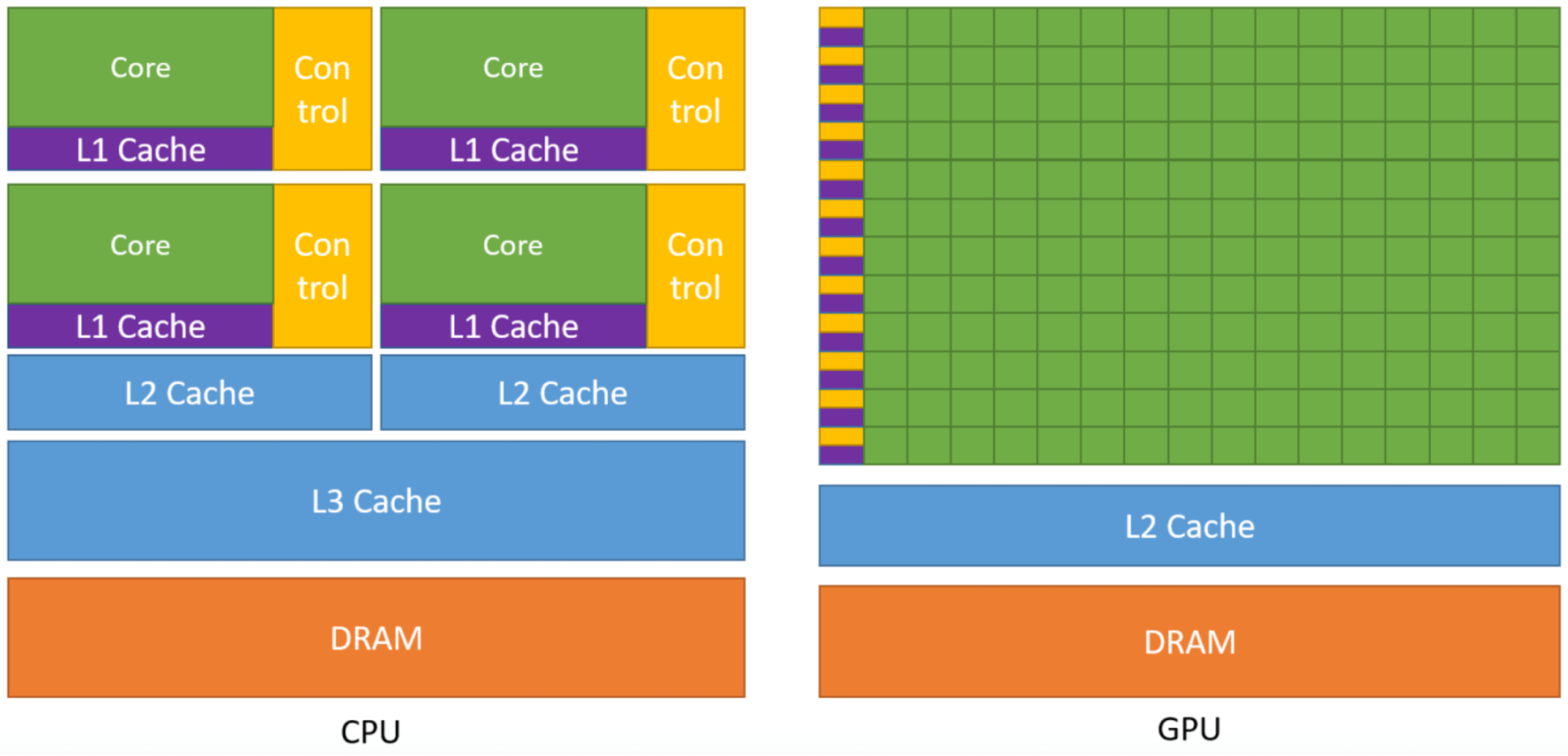
- Bullet 1
- Bullet 2
- Bullet 3

Learning Goals

- Knowledge of different types of transformer architectures and how they are trained
  - Notably BERT and masked language modelling (MLM)
- An understanding of the relation between MLM and linguistic tasks
- An understanding of pre-training
  - And its influence on model performance
- The influence of scale on model performance
- An understanding of what language models learn
  - And how we can examine this using probing

Question: What is a GPU?

- Central Processing Unit (CPU)
  - Few highly performant cores (4-16), Serial processing, general purpose
  - Optimizes for: Low latency
- Accelerators
  - **Graphical** Processing Unit (GPU)
    - Many cores (100-1000),
    - Optimizes for: Throughput
  - **Neural** Processing Unit (NLU)
  - ...



This is by no means a comprehensive description of GPUs

How does that mean for us?

- Inference using the bert-base-cased

	Cost /hour	Cost /inference	Max inference per hour
8cpu	0.197872	0.003848	51.42
16cpu	0.395744	0.004584	86.33
32cpu	0.791488	0.008266	95.74
1gpu	2.004	0.001447	1384.61

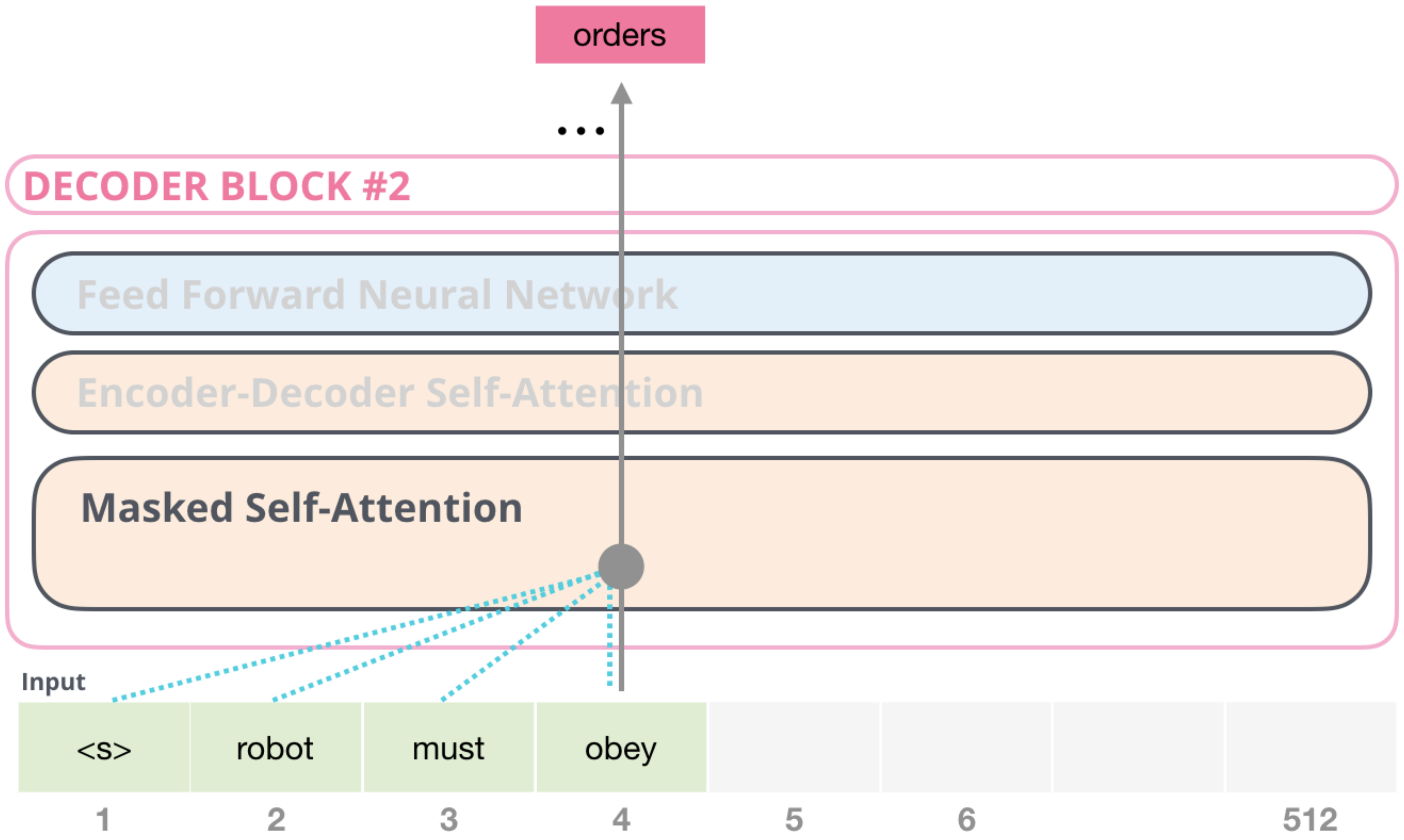
14-26x

- Difference is something taking a day vs. 30 min
- Notably larger effect for training

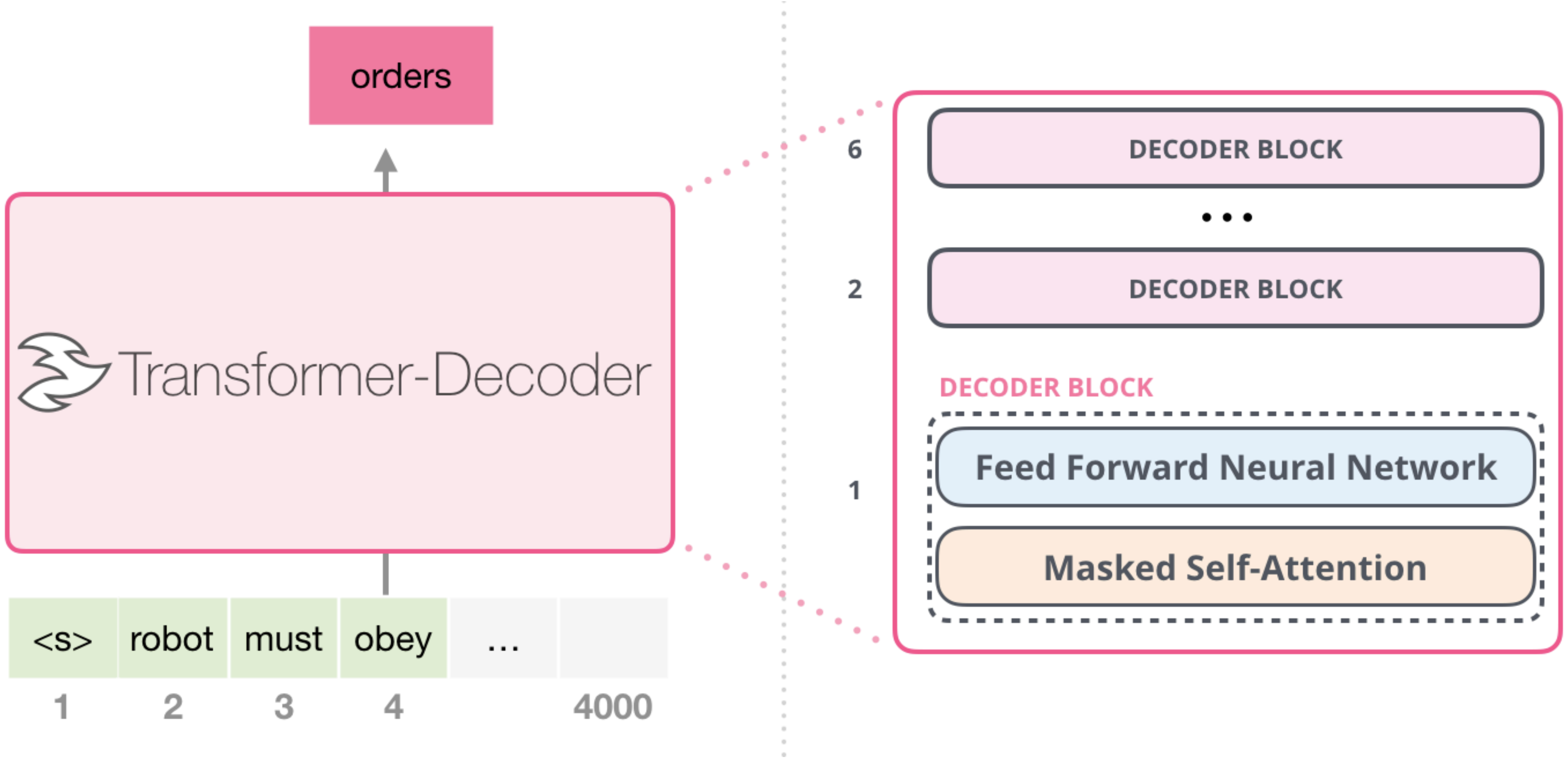
Google cloud using an NVIDIA V100 GPU and e2-highcpus

Recap: Decoder transformers block

Unidirectional Attention

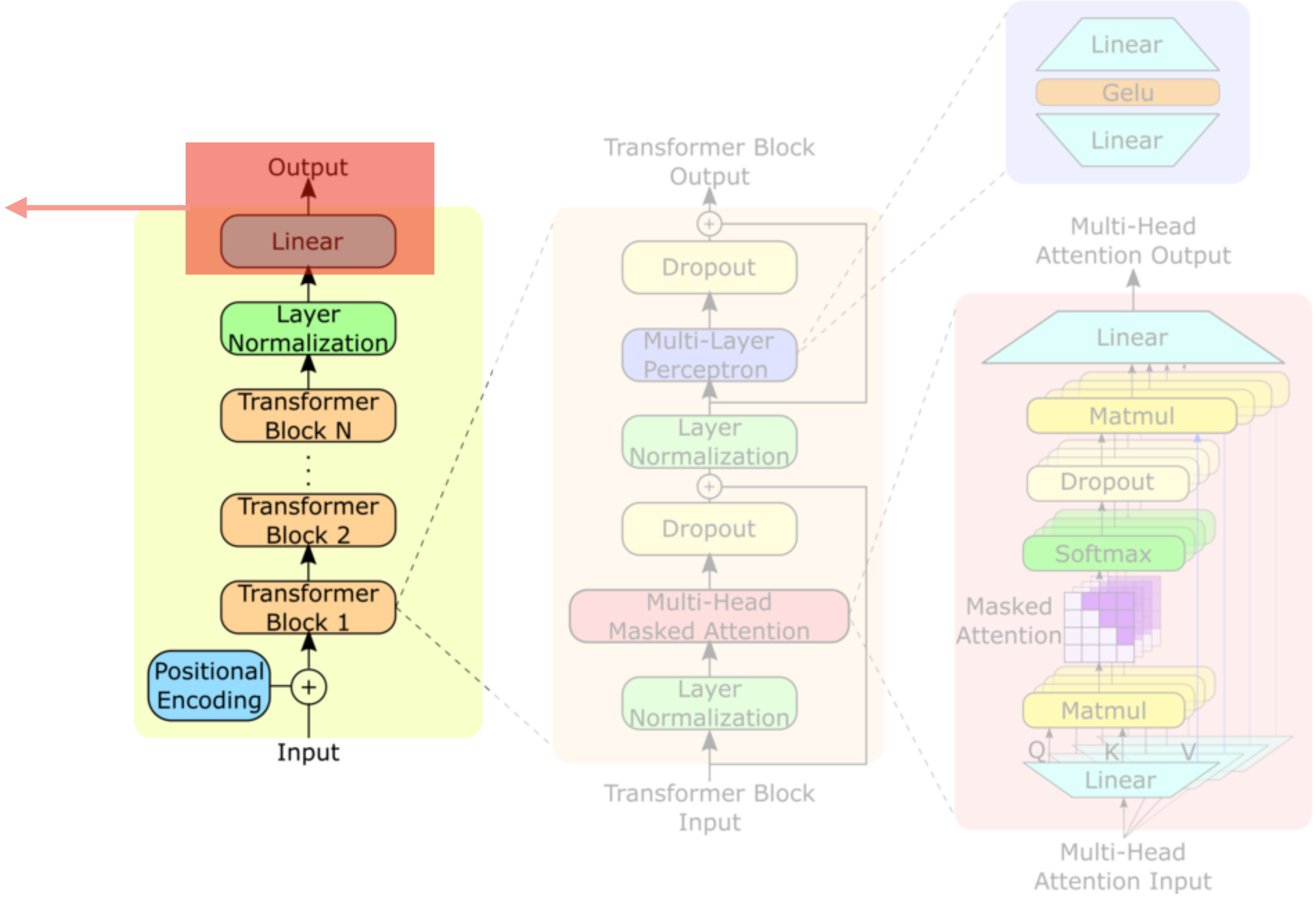


Recap: Transformer blocks for langauge generation



Recap: Overview of GPT

Language
modelling Head

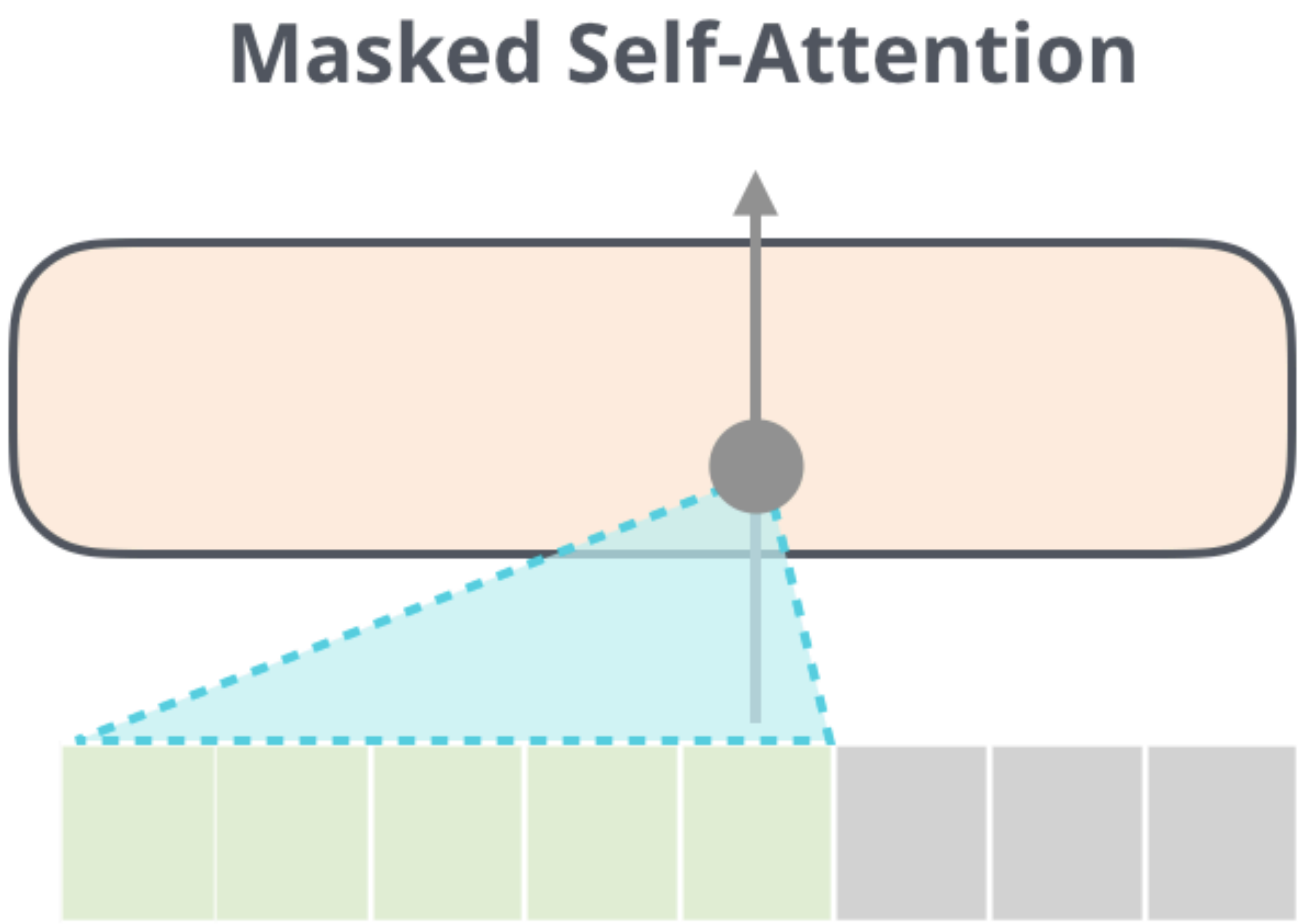


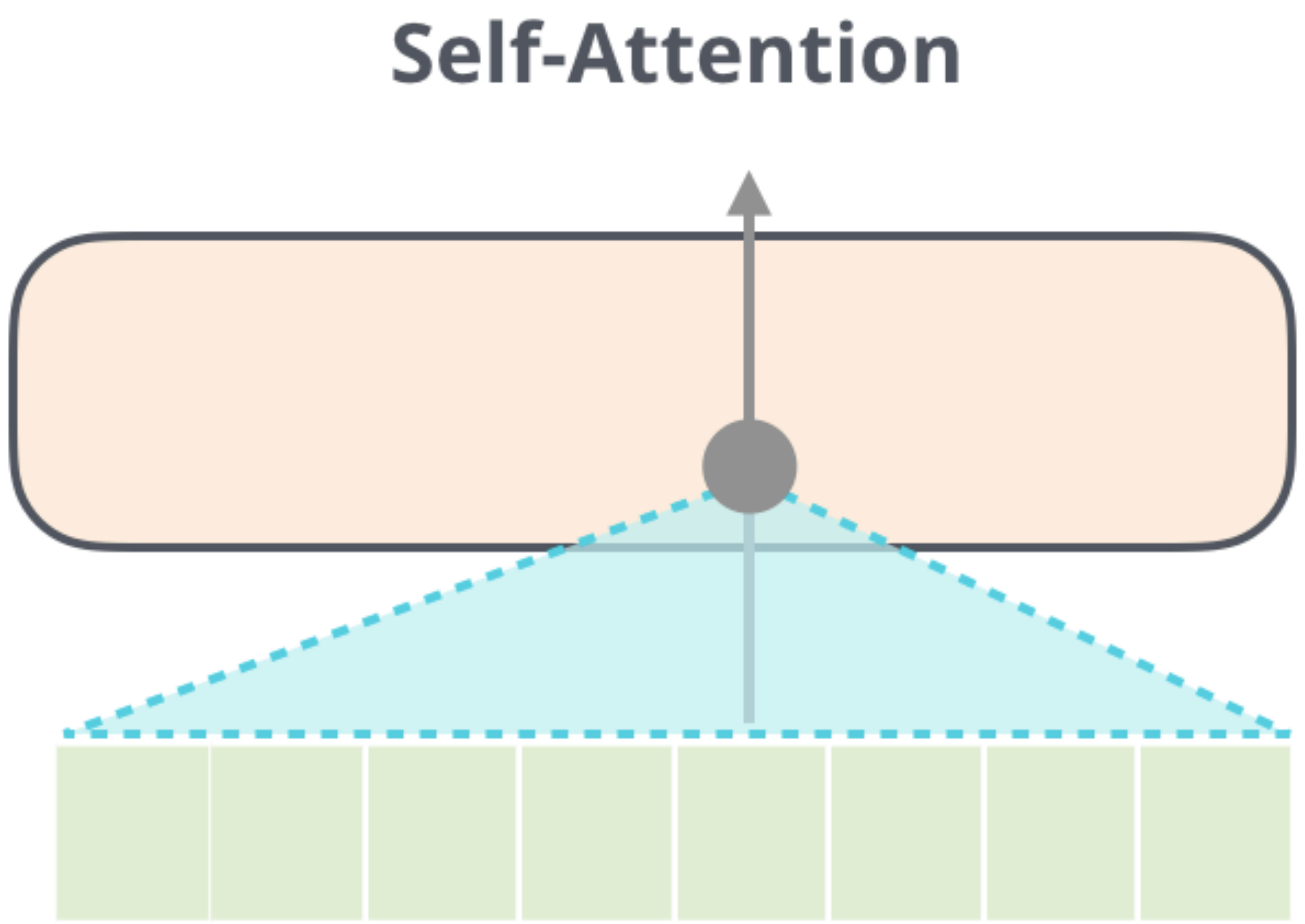
Understanding a text backward

Ich Kann ohne meinen **Ausweis** nicht den Club **gehen**

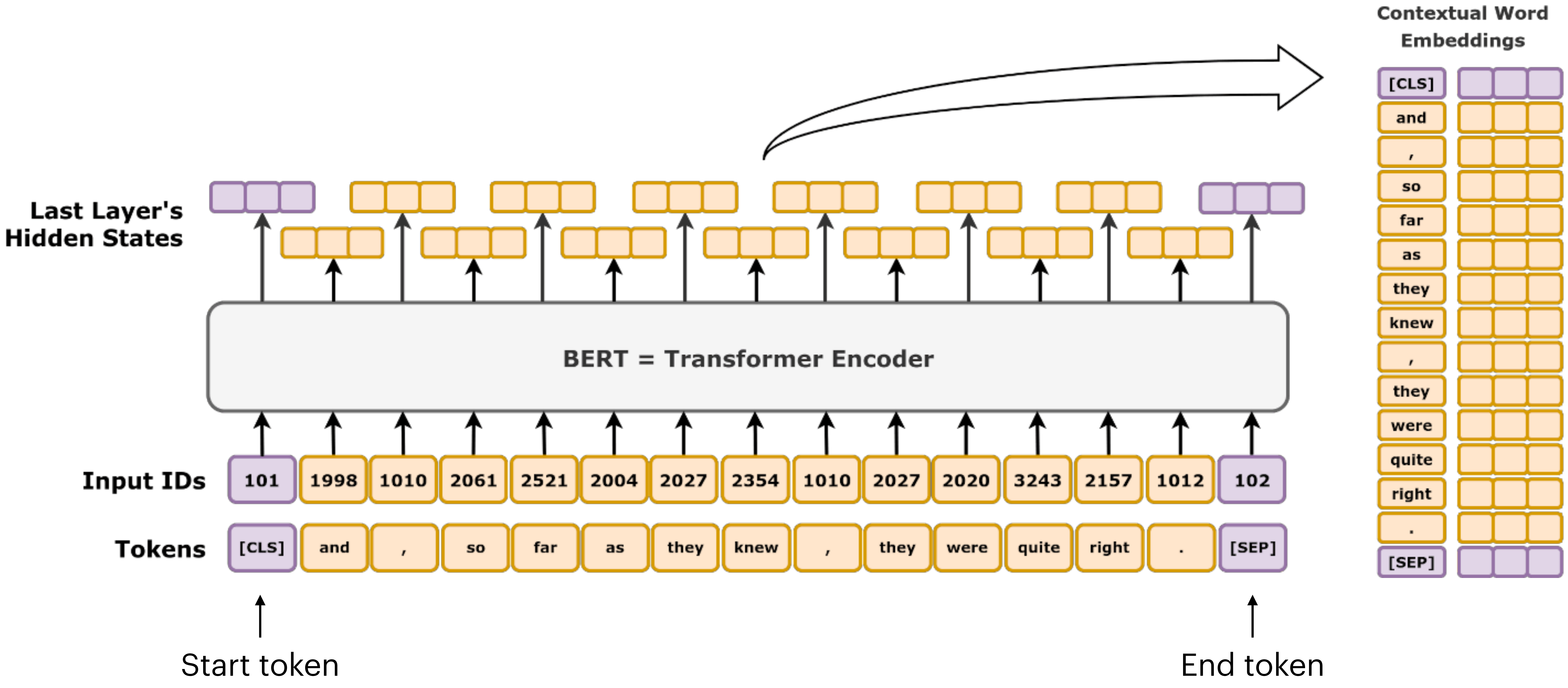
I can't **go** to the club without my **ID card**

Unidirectional to Bidirectional

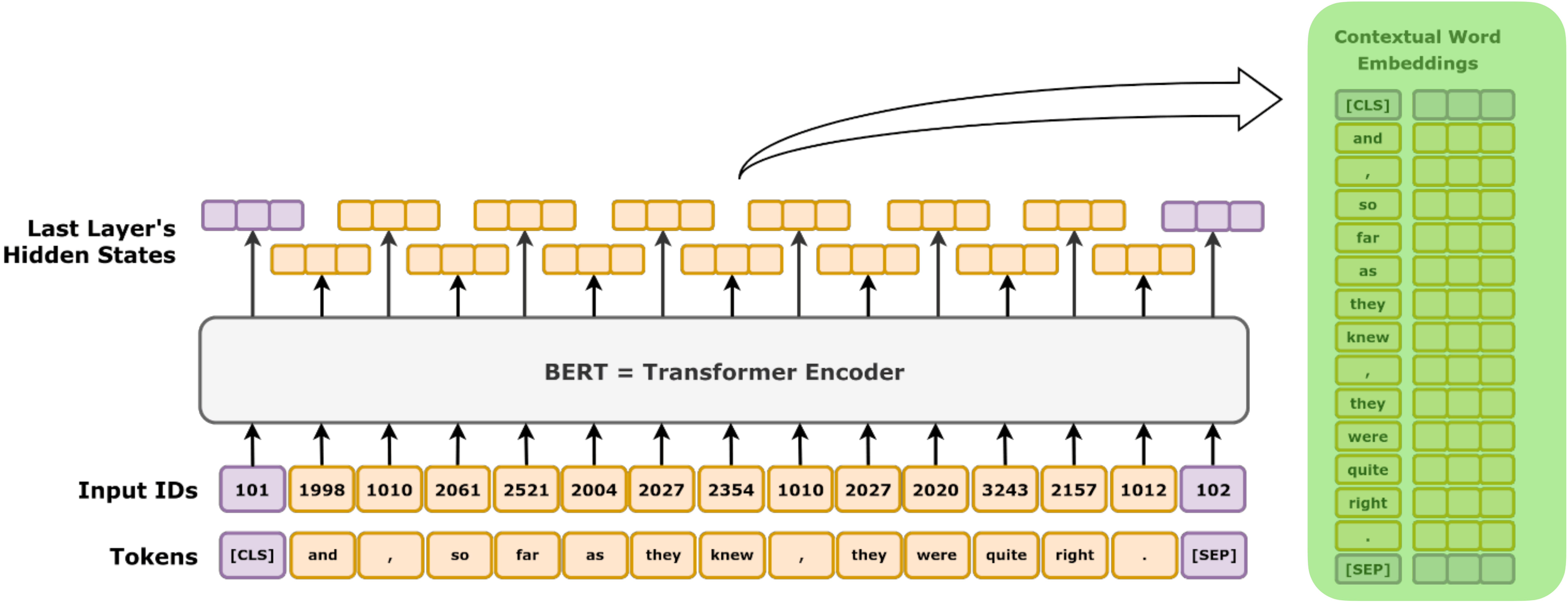




BERT: Biderectional Attention

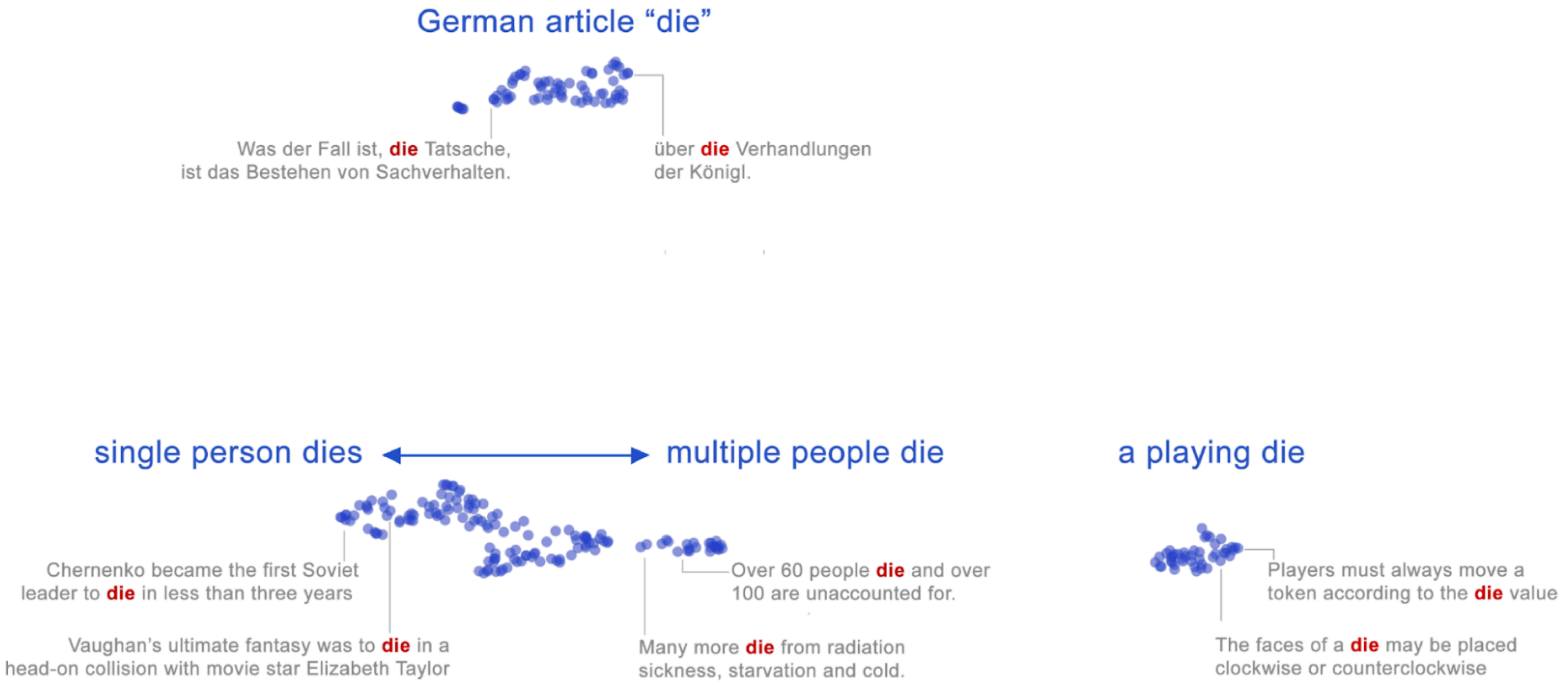


BERT: Biderectional Attention



Q: How can we use these?

Visualizing Contextualized Embeddings



Explore more:
https://storage.googleapis.com/bert-wsd-vis/demo/index.html?#word=die

Coenen et al. (2019), Visualizing and Measuring the Geometry of BERT

Contextualized word vectors using BERT

```python
from transformers import BertTokenizer, BertModel

# download model and tokenizer
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')
model = BertModel.from_pretrained("bert-base-uncased")

# encode text
text = "Replace me by any text you'd like."

# convert text to token ids
encoded_input = tokenizer(text, return_tensors='pt')

encoded_input.input_ids.shape # torch.Size([1, 12])

# embed ids and contextualize
output = model(**encoded_input)

output.last_hidden_state.shape # torch.Size([1, 12, 768])
```

Q: What is the shape?

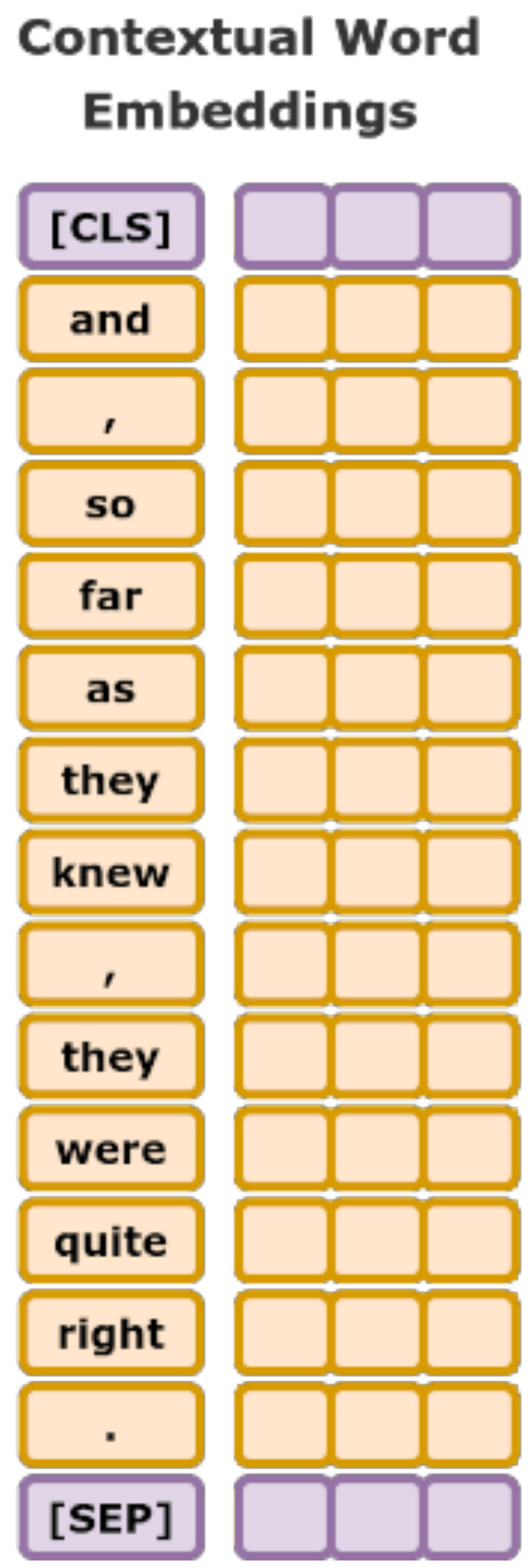
Contextualized word vectors using BERT

```python
from transformers import BertTokenizer, BertModel

# download model and tokenizer
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')
model = BertModel.from_pretrained("bert-base-uncased")

# encode text
text = "Replace me by any text you'd like."

# convert text to token ids
encoded_input = tokenizer(text, return_tensors='pt')

encoded_input.input_ids.shape # torch.Size([1, 12])

# embed ids and contextualize
output = model(**encoded_input)

output.last_hidden_state.shape # torch.Size([1, 12, 768])
```

Q: What is the shape?

How do we train it?

- Two approaches
  - ~~Next sentence prediction (NSP)~~*
    - Sequence-level representation
  - Masked language Modelling (MLM)
    - Word-level representations



Sources & Notes

* We don't use NSP anymore, it was showed by Liu et al., (2019) that this did not lead to an improvement. The task is generally believed to be too easy.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. ArXiv, abs/1907.11692.

Cloze Task

- Cloze -> closure (gestalt theory)
- Language assesement test
- Understanding of context

  "Today, I went to the ________ and bought some milk and eggs. I knew it was going to rain, but I fo"rgot to take my ______, and ended up getting wet on the way."

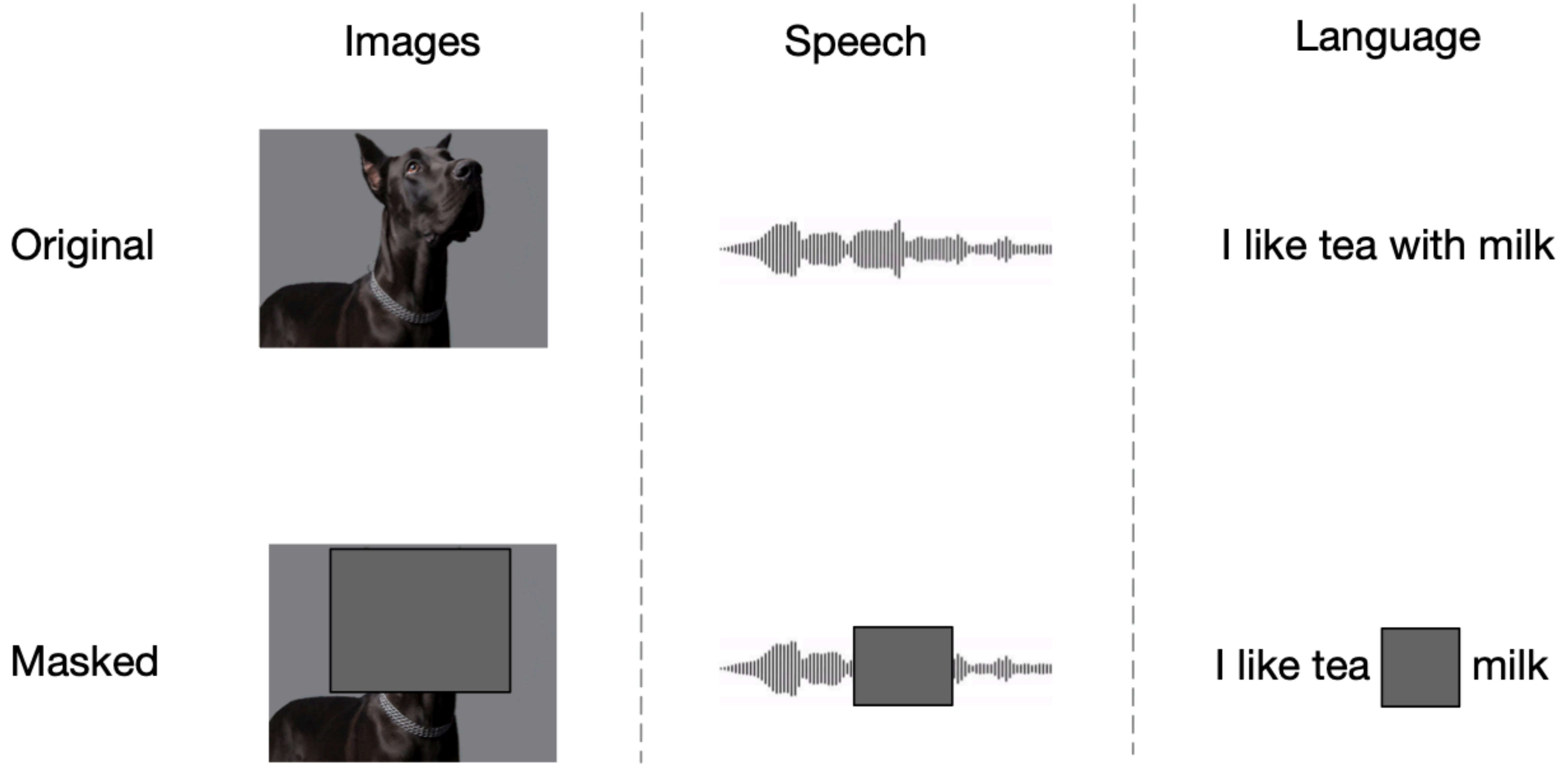
- Factual Knowledge

  "______ is the anaerobic catabolism of glucose."

Sources & Notes

Taylor, W. L. (1953). "Cloze procedure: A new tool for measuring readability". Journalism Quarterly. 30 (4): 415–433

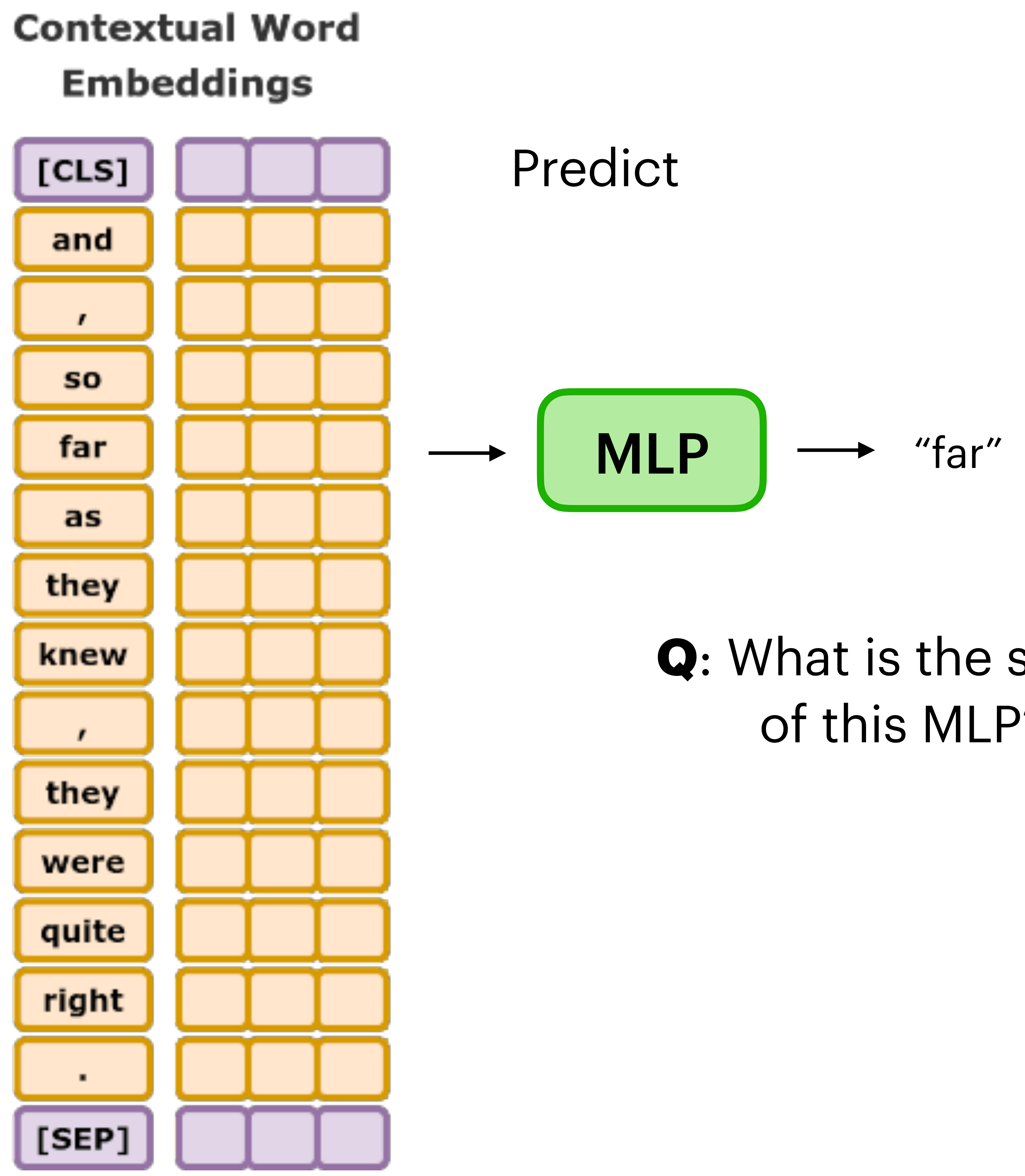
Masking in other modalities



Baevski, A., Hsu, W., Xu, Q., Babu, A., Gu, J., & Auli, M. (2022). data2vec: A General Framework for Self-supervised Learning in Speech, Vision and Language. ArXiv, abs/2202.03555.

How do we train it?

- Two approaches
  - ~~Next sentence prediction (NSP)~~*
  - Masked language Modelling (MLM)
    - Goal: Predict word from its context



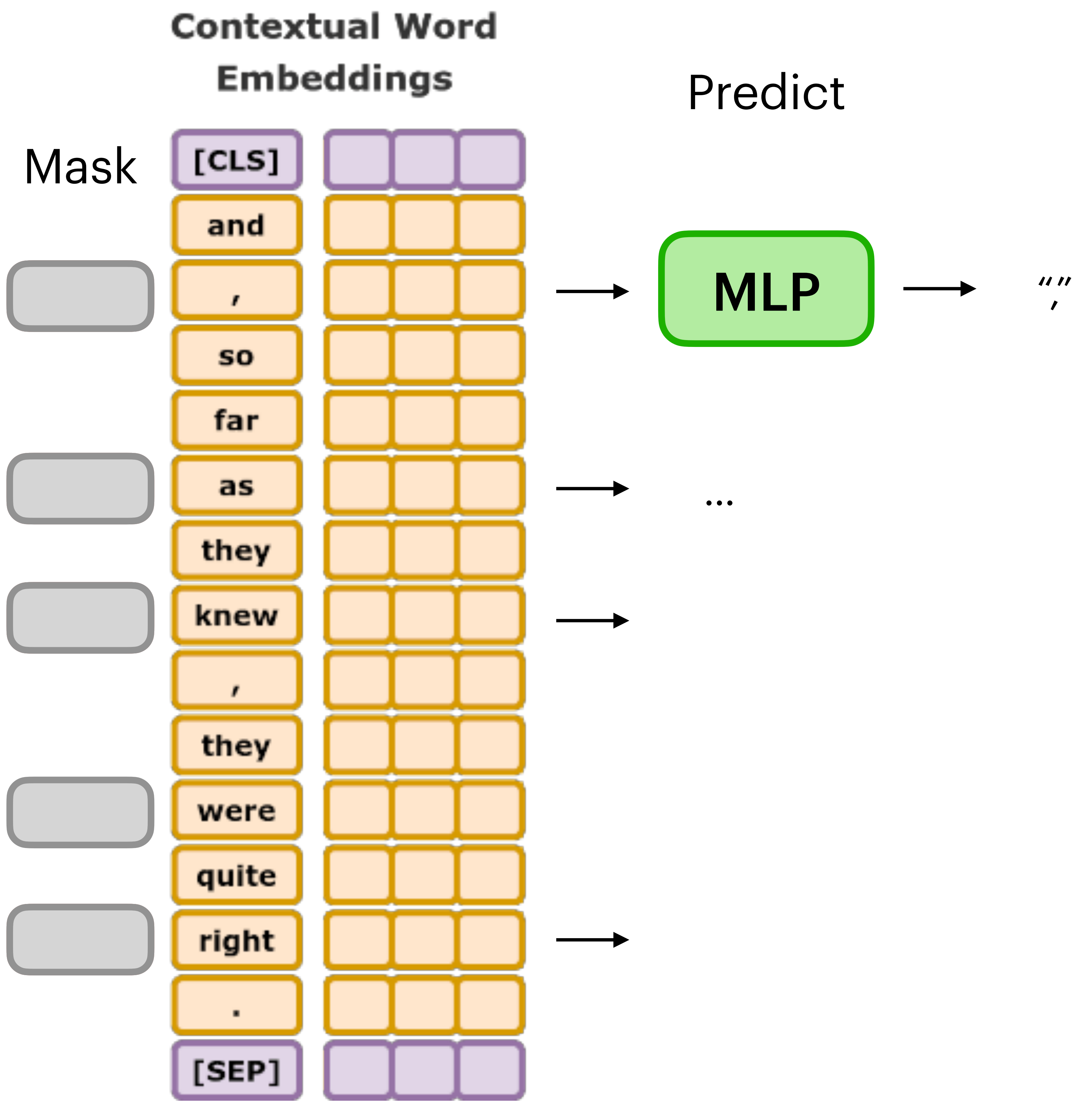
Q: What is the shape of this MLP?

* We don't use NSP anymore, it was showed by Liu et al., (2019) that this did not lead to an improvement. The task is generally believed to be too easy.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. ArXiv, abs/1907.11692.

How do we train it?

- Two approaches
  - ~~Next sentence prediction (NSP)~~*
  - Masked language Modelling (MLM)
    - Mask 15%



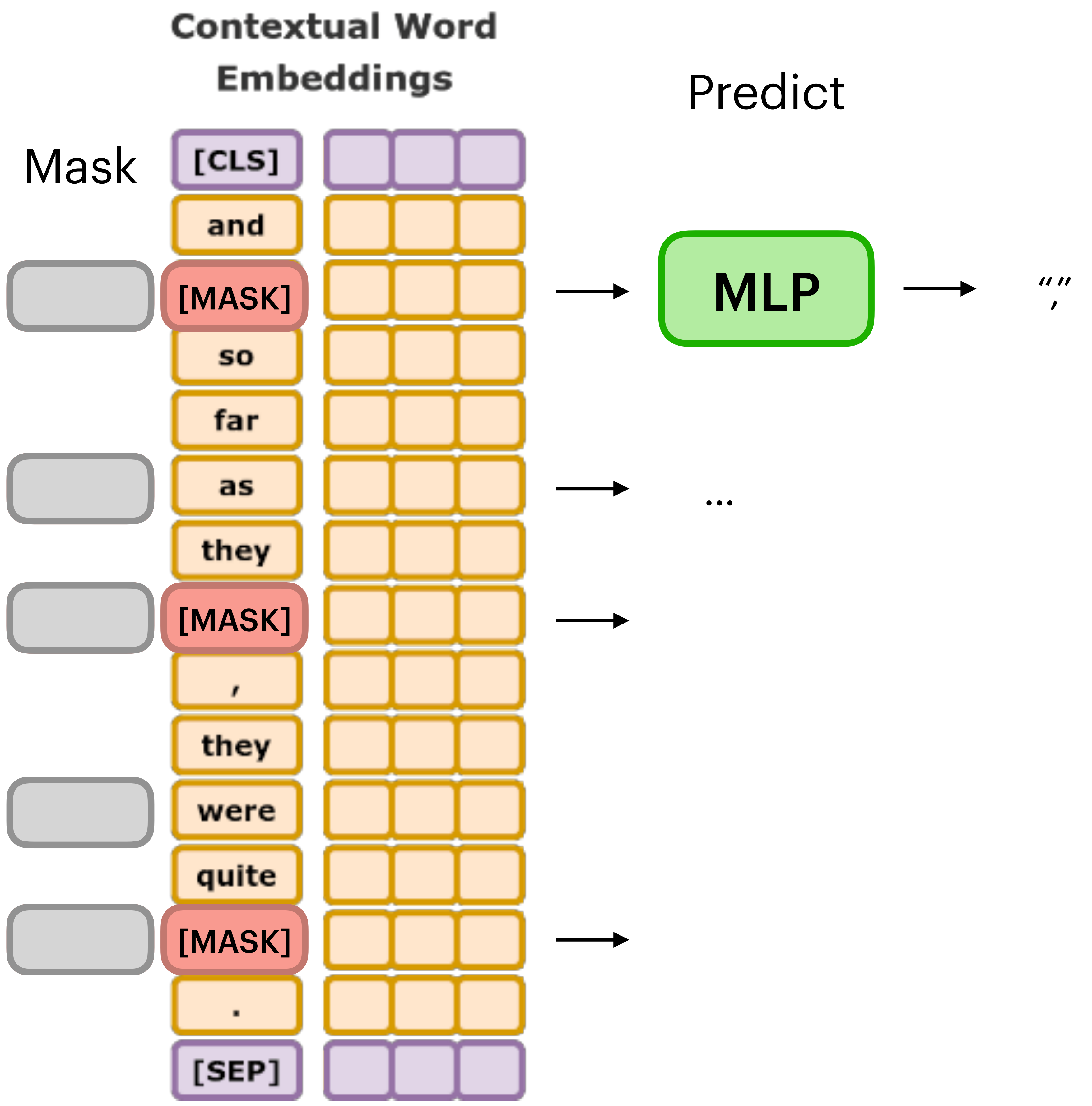
Sources & Notes

* We don't use NSP anymore, it was showed by Liu et al., (2019) that this did not lead to an improvement. The task is generally believed to be too easy.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. ArXiv, abs/1907.11692.

How do we train it?

- Two approaches
  - ~~Next sentence prediction (NSP)~~*
  - Masked language Modelling (MLM)
    - Mask 15%
    - Replace 80% with mask

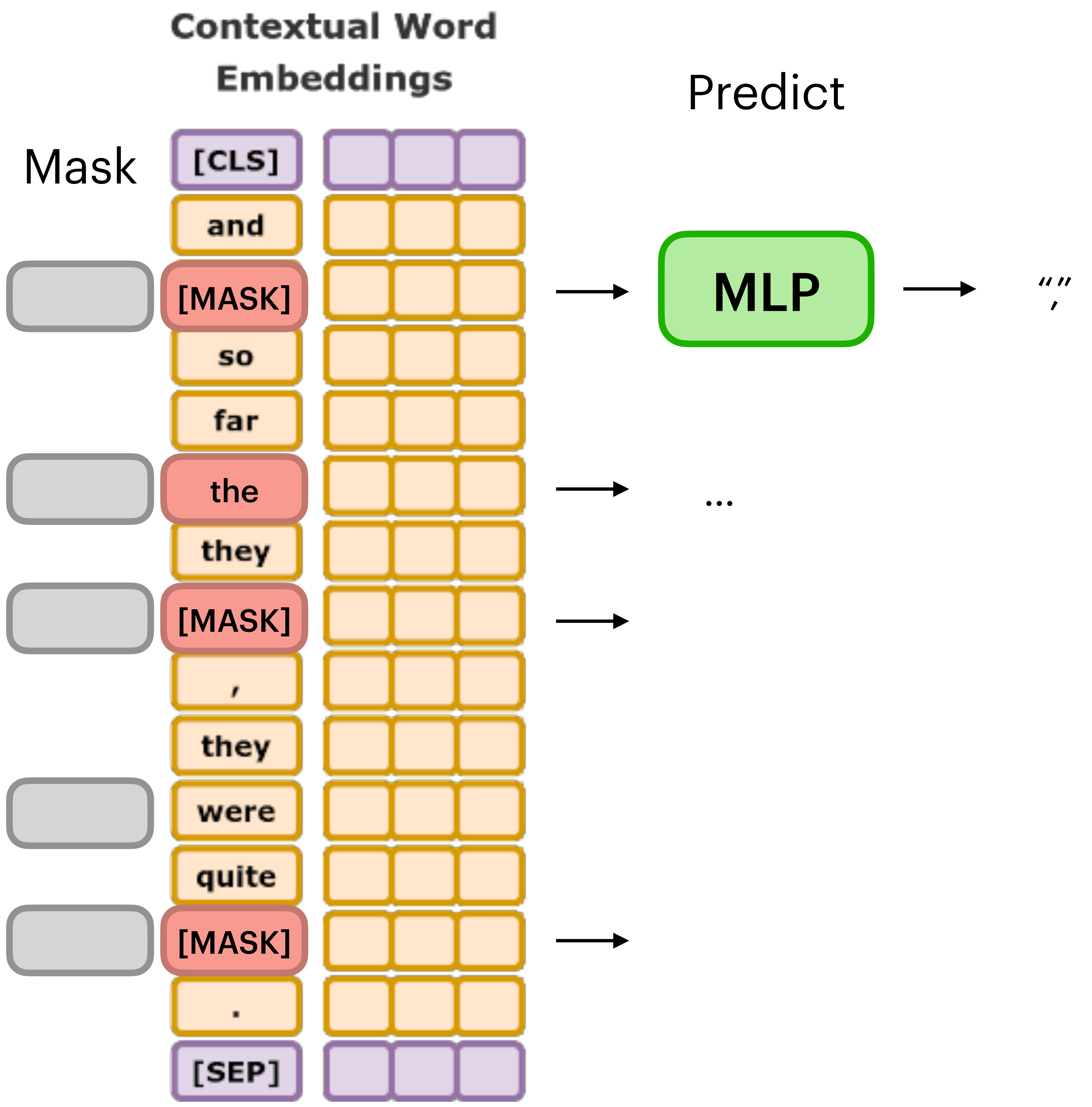


* We don’t use NSP anymore, it was showed by Liu et al., (2019) that this did not lead to an improvement. The task is generally believed to be too easy.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. ArXiv, abs/1907.11692.

How do we train it?

- Two approaches
  - ~~Next sentence prediction (NSP)~~*
  - Masked language Modelling (MLM)
    - Mask 15%
    - Replace 80% with mask
    - 10% with a random token
    - Leave 10% as is



Sources & Notes

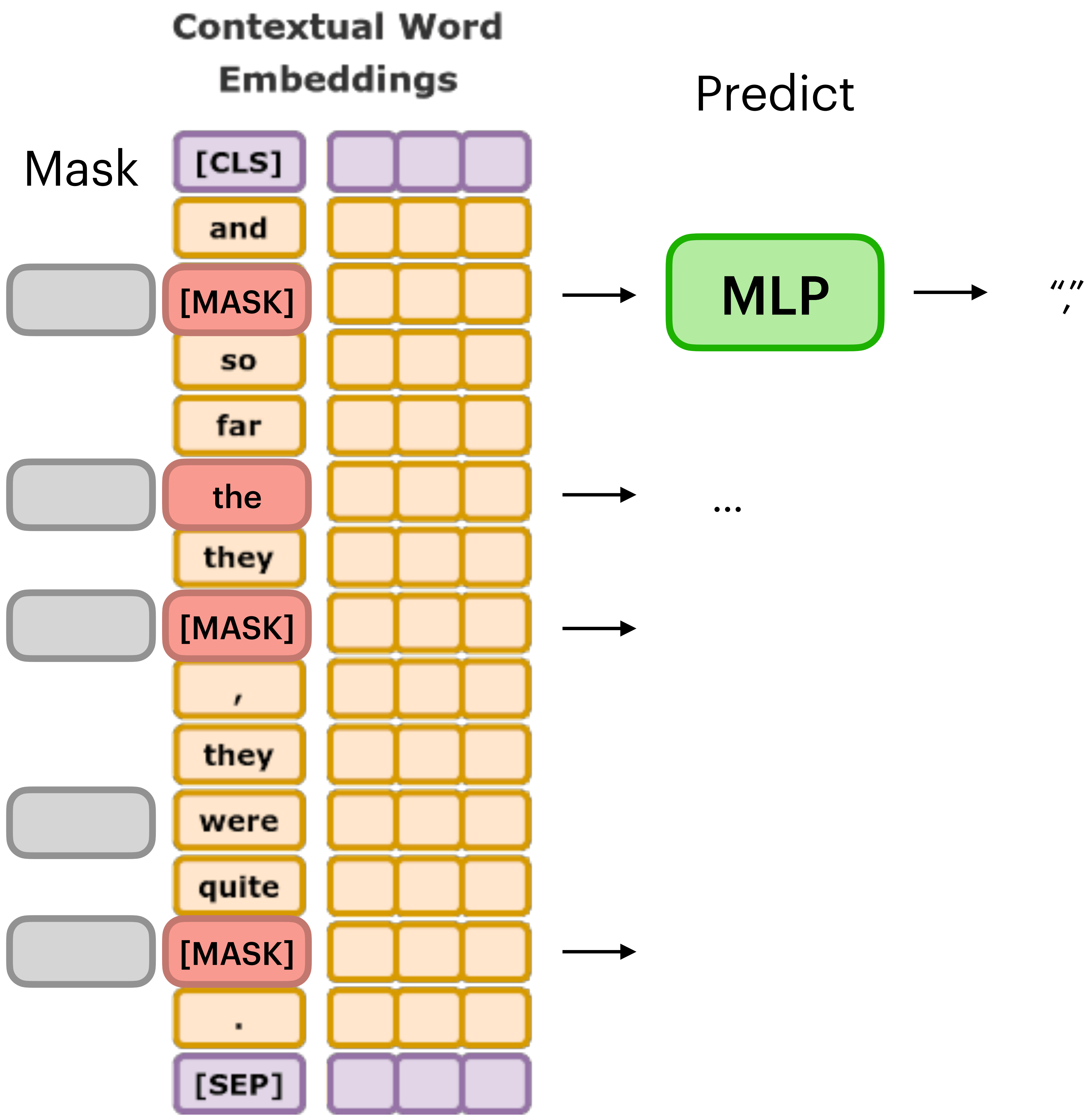
* We don't use NSP anymore, it was showed by Liu et al., (2019) that this did not lead to an improvement. The task is generally believed to be too easy.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. ArXiv, abs/1907.11692.

How do we train it?

- Two approaches
  - ~~Next sentence prediction (NSP)~~*
  - Masked language Modelling (MLM)
    - Mask 15%
    - Replace 80% with mask
    - 10% with a random token
    - Leave 10% as is

Q: Why?



Sources & Notes

* We don’t use NSP anymore, it was showed by Liu et al., (2019) that this did not lead to an improvement. The task is generally believed to be too easy.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. ArXiv, abs/1907.11692.

CENTER FOR HUMANITIES COMPUTING

Training BERT

Building the model

```
import torch
from torch import nn
from transformers import AutoModel, AutoTokenizer

tokenizer = AutoTokenizer.from_pretrained("bert-base-uncased")
model = AutoModel.from_pretrained("bert-base-uncased")

# add a classification head
classifier = nn.Linear(
    768, num_labels
)  # embed dim is 768, num_labels is the number of classes

# combine the model and the classifier
model = nn.Sequential(model, classifier)
```

```
# Or using the transformer library

from transformers import AutoModelForSequenceClassification

model = AutoModelForSequenceClassification.from_pretrained(
    "google-bert/bert-base-cased", num_labels=5
)
```

Training the model*

```
# training the model
loss = nn.CrossEntropyLoss()
optimizer = torch.optim.AdamW(model.parameters(), lr=5e-5)  # or SGD

train_loader = …  # create a DataLoader with your data

model.train()

for epoch in range(epochs):  # number of epochs to train for
    for text, label in train_loader:
        optimizer.zero_grad()

        # tokenize text
        token_ids = tokenizer(text, return_tensors="pt")
        output = model(**token_ids)

        # compute loss
        loss_value = loss(output, label)

        # compute gradients
        loss_value.backward()

        # update weights
        optimizer.step()
```

Sources & Notes

*Here you can also use the transformer library e.g. using their trainer API: https://huggingface.co/docs/transformers/training

Any Questions?

Learning Goals

- Knowledge of different types of transformer architectures and how they are trained
  - Notably BERT and masked language modelling (MLM)
- An understanding of the relation between MLM and linguistic tasks
- An understanding of pre-training
  - And its influence on model performance
- The influence of scale on model performance
- An understanding of what language models learn
  - And how we can examine this using probing

Evaluating BERT

We will be **assuming** a lot of knowledge about evaluation tasks.
Generally not recommended. More on this in class 9.

P[a]	A senior is waiting at the window of a restaurant that serves sandwiches.	Relationship
H[b]	A person waits to be served his food.	Entailment
	A man is looking to order a grilled cheese sandwich.	Neutral
	A man is waiting in line for the bus.	Contradiction

[a]**P**, Premise.
[b]**H**, Hypothesis.

Sentiment Analysis

Textual Entailment

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**

Evaluating BERT

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**

Note: GPT 1

GLUE: General Language Understanding Evaluation

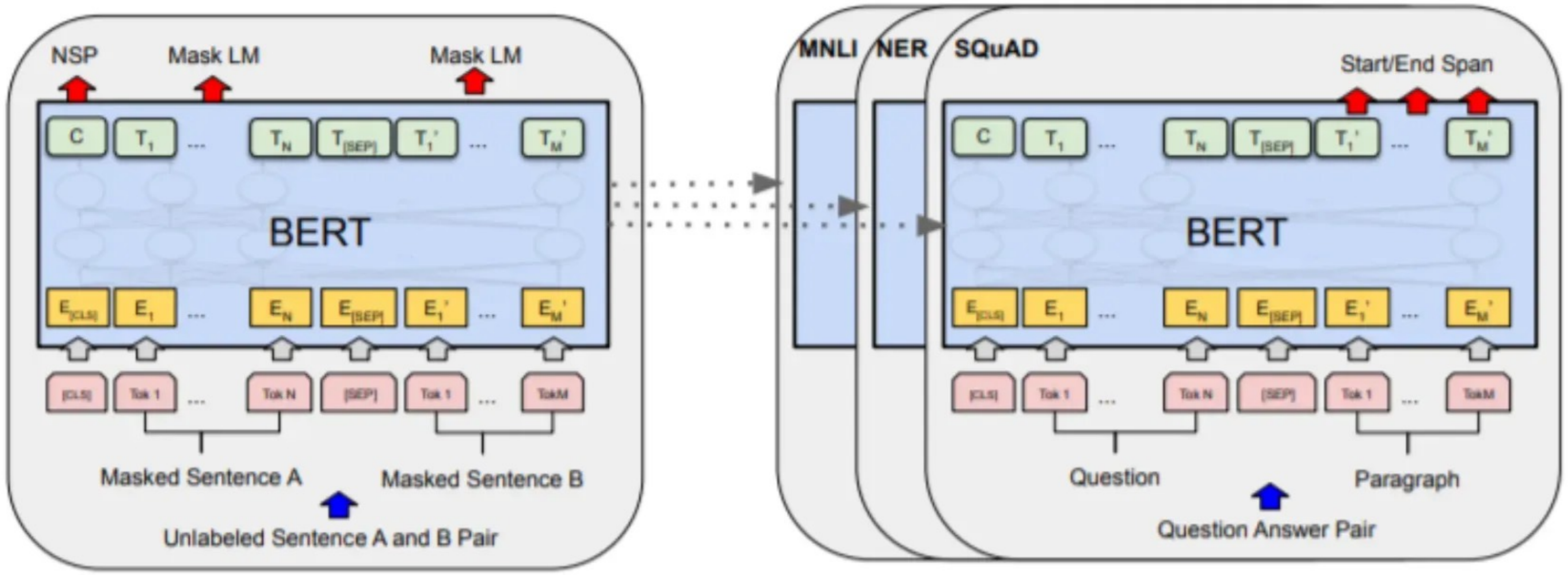
Evaluating BERT

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**

More parameters lead to better performance

Pre-Training and Scale

- **Pre-training**
  - **Semi-supervised training** of model with labels derived from data
    - Next token prediction
    - Masked language modelling
    - …*
- **Fine-tuning**
  - Task-specific training using **labelled training data**
- **Note:** In the following we will take examples from each architecture, but findings generalize across



Pre-training
Fine-Tuning

Sources & Notes

* Span corruption (T5), Replaced token detection (ELECTRA), Contrastive methods (SimCSE), …

The effect of **pre-training**

Effect of pre-training on T5:

	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Baseline average	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
Baseline standard deviation	0.235	0.065	0.343	0.416	0.112	0.090	0.108
No pre-training	66.22	17.60	50.31	53.04	25.86	**39.77**	24.04

Task with few trianing samples or highly diverse tasks is **highly affected by pre-training**

Tasks with **a lot of training data** isn't affected by pre-training

The effect of **data**

- Encoders = Birectional (full attention)*
  - BERT, RoBERTa, ELECTRA, ...

Pre-training on **more data**
Increase performance

Model	data	bsz	steps	SQuAD (v1.1/2.0)	MNLI-m	SST-2
RoBERTa						
with BOOKS + WIKI	16GB	8K	100K	93.6/87.3	89.0	95.3
+ additional data (§3.2)	160GB	8K	100K	94.0/87.7	89.3	95.6
+ pretrain longer	160GB	8K	300K	94.4/88.7	90.0	96.1
+ pretrain even longer	160GB	8K	500K	**94.6/89.4**	**90.2**	**96.4**

Sources
& Notes

*A decoder can also sometimes be referred to as an encoder.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. ArXiv, abs/1907.11692.

The effect of **compute time**

- Encoders = Birectional (full attention)*
  - BERT, RoBERTa, ELECTRA, …

Model	data	bsz	steps	SQuAD (v1.1/2.0)	MNLI-m	SST-2
RoBERTa						
with BOOKS + WIKI	16GB	8K	100K	93.6/87.3	89.0	95.3
+ additional data (§3.2)	160GB	8K	100K	94.0/87.7	89.3	95.6
+ pretrain longer	160GB	8K	300K	94.4/88.7	90.0	96.1
+ pretrain even longer	160GB	8K	500K	**94.6/89.4**	**90.2**	**96.4**

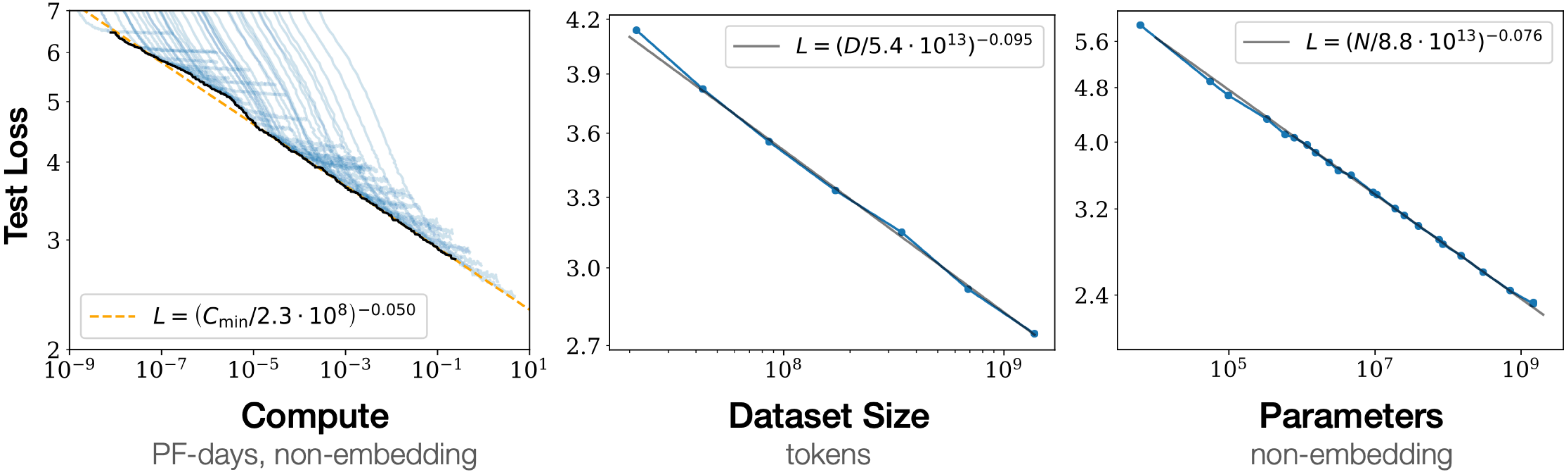
Pre-training **for longer** on those data increase performance

Sources & Notes

*A decoder can also sometimes be referred to as an encoder.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. ArXiv, abs/1907.11692.

Scaling Laws or Neural Language Models



Sources & Notes

Kaplan, J., McCandlish, S., Henighan, T., Brown, T.B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., & Amodei, D. (2020). Scaling Laws for Neural Language Models. ArXiv, abs/2001.08361.

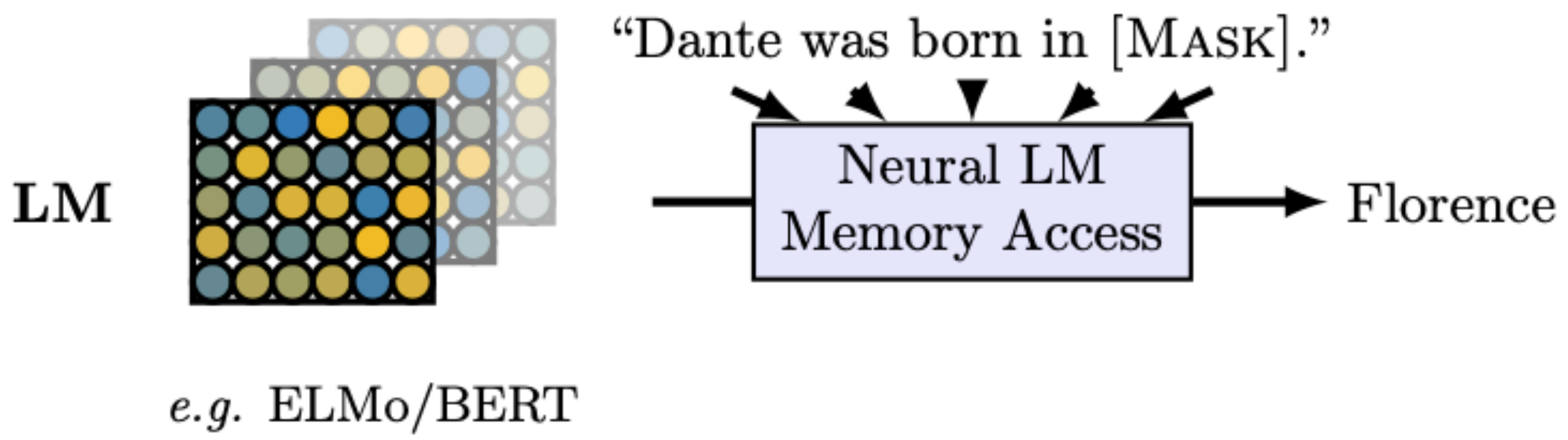
CENTER FOR HUMANITIES COMPUTING

What does the Model learn?

- Samples from the BERTology
  - Will not be comprehensive
- Includes multiple types of probes
- As well as behavioural analysis

Knowledge within BERT

Behavioural experiment of the model



In some situations competitive with Knowledge bases!

Prompting

```
from transformers import pipeline
unmasker = pipeline('fill-mask', model='bert-base-uncased')
unmasker("The capital of Denmark is [MASK].")
# [{'score': 0.9113172888755798,
#   'token': 9664,
#   'token_str': 'copenhagen',
#   'sequence': 'the capital of denmark is copenhagen.'},
#  {'score': 0.06609592586755753,
#   'token': 29173,
#   'token_str': 'aarhus',
#   'sequence': 'the capital of denmark is aarhus.'},
#  {'score': 0.003040957497432828,
#   'token': 5842,
#   'token_str': 'denmark',
#   'sequence': 'the capital of denmark is denmark.'},
#  {'score': 0.0017591337673366607,
#   'token': 11755,
#   'token_str': '##borg',
#   'sequence': 'the capital of denmark isborg.'},
#  {'score': 0.0013613264309242368,
#   'token': 21860,
#   'token_str': 'lund',
#   'sequence': 'the capital of denmark is lund.'}]
```

Negations

Within fine-tuning BERT isn't influenced overly by negations

Context	Match
A robin is a ____	*bird*
A robin is not a ____	*bird*

Humans are apparently not too surprised by:
"A robin is not a bird" (measures using N400)

Context	$\text{BERT}_{\text{LARGE}}$ predictions
A robin is a ____	*bird, robin, person, hunter, pigeon*
A daisy is a ____	*daisy, rose, flower, berry, tree*
A hammer is a ____	*hammer, tool, weapon, nail, device*
A hammer is an ____	*object, instrument, axe, implement, explosive*
A robin is not a ____	*robin, bird, penguin, man, fly*
A daisy is not a ____	*daisy, rose, flower, lily, cherry*
A hammer is not a ____	*hammer, weapon, tool, gun, rock*
A hammer is not an ____	*object, instrument, axe, animal, artifact*

Table 13: $\text{BERT}_{\text{LARGE}}$ top word predictions for selected NEG-136-SIMP sentences

Sources & Notes

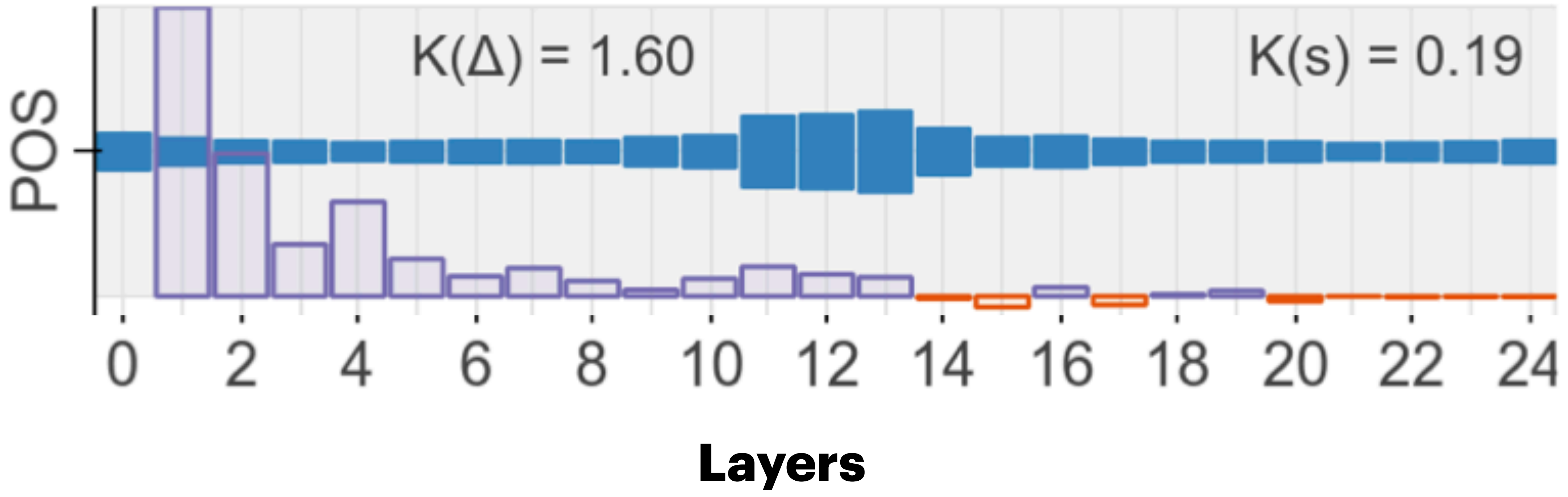
Ettinger, A. (2019). What BERT Is Not: Lessons from a New Suite of Psycholinguistic Diagnostics for Language Models. Transactions of the Association for Computational Linguistics, 8, 34-48.

Biases in BERT

```
>>> from transformers import pipeline
>>> unmasker = pipeline('fill-mask', model='bert-base-uncased')
>>> unmasker("The man worked as a [MASK].")

[{'sequence': '[CLS] the man worked as a carpenter. [SEP]',
  'score': 0.09747550636529922,
  'token': 10533,
  'token_str': 'carpenter'},
 {'sequence': '[CLS] the man worked as a waiter. [SEP]',
  'score': 0.0523831807076931,
  'token': 15610,
  'token_str': 'waiter'},
 {'sequence': '[CLS] the man worked as a barber. [SEP]',
  'score': 0.04962705448269844,
  'token': 13362,
  'token_str': 'barber'},
 {'sequence': '[CLS] the man worked as a mechanic. [SEP]',
  'score': 0.03788609802722931,
  'token': 15893,
  'token_str': 'mechanic'},
 {'sequence': '[CLS] the man worked as a salesman. [SEP]',
  'score': 0.037680890411138535,
  'token': 18968,
  'token_str': 'salesman'}]
```

```
>>> unmasker("The woman worked as a [MASK].")

[{'sequence': '[CLS] the woman worked as a nurse. [SEP]',
  'score': 0.21981462836265564,
  'token': 6821,
  'token_str': 'nurse'},
 {'sequence': '[CLS] the woman worked as a waitress. [SEP]',
  'score': 0.1597415804862976,
  'token': 13877,
  'token_str': 'waitress'},
 {'sequence': '[CLS] the woman worked as a maid. [SEP]',
  'score': 0.1154729500412941,
  'token': 10850,
  'token_str': 'maid'},
 {'sequence': '[CLS] the woman worked as a prostitute. [SEP]',
  'score': 0.037968918681144714,
  'token': 19215,
  'token_str': 'prostitute'},
 {'sequence': '[CLS] the woman worked as a cook. [SEP]',
  'score': 0.03042375110089779,
  'token': 5660,
  'token_str': 'cook'}]
```

BERT rediscovers the Classical NLP Pipeline



~What layer a "probe" find most predictive

Probing a transformer

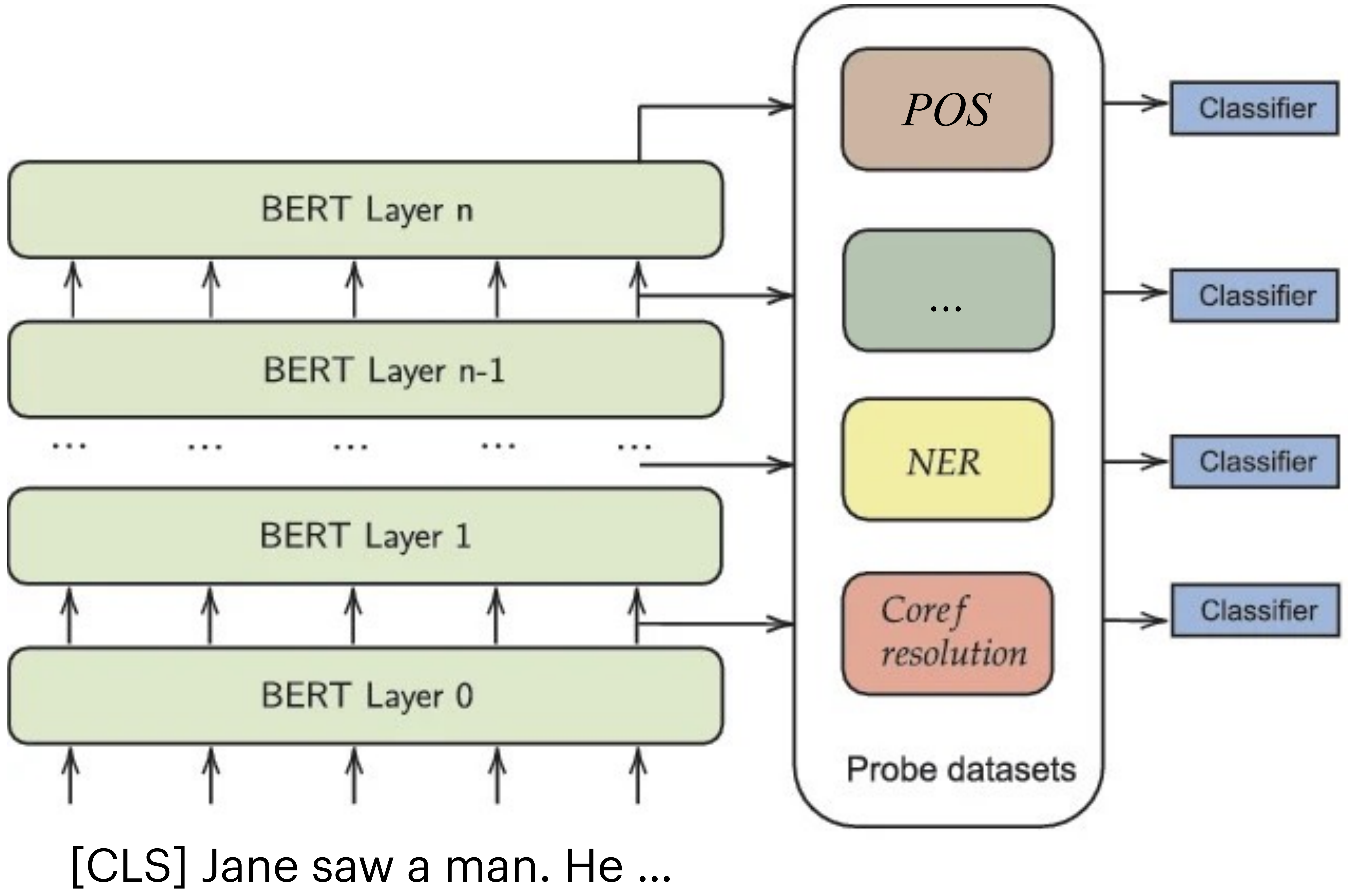
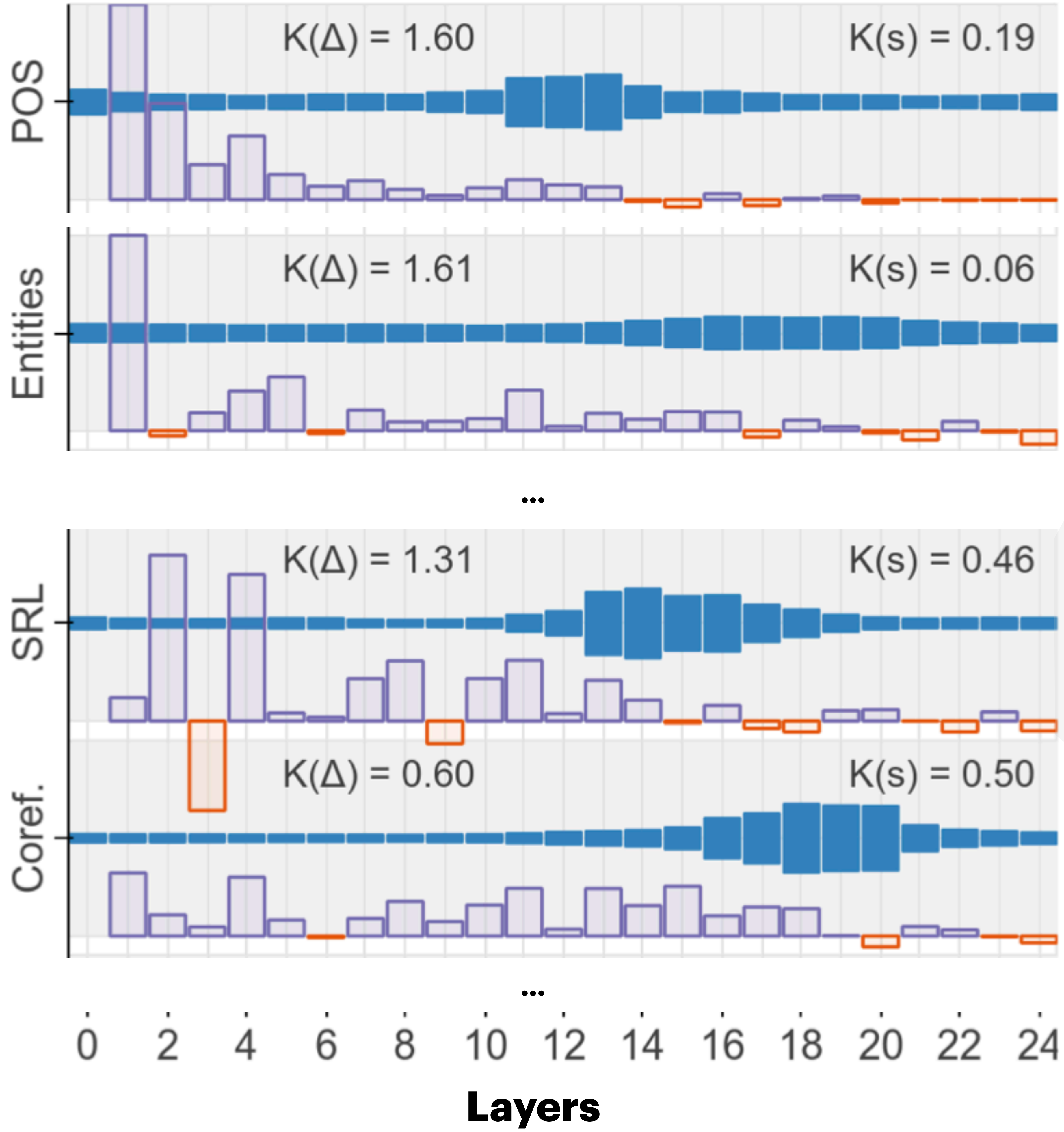
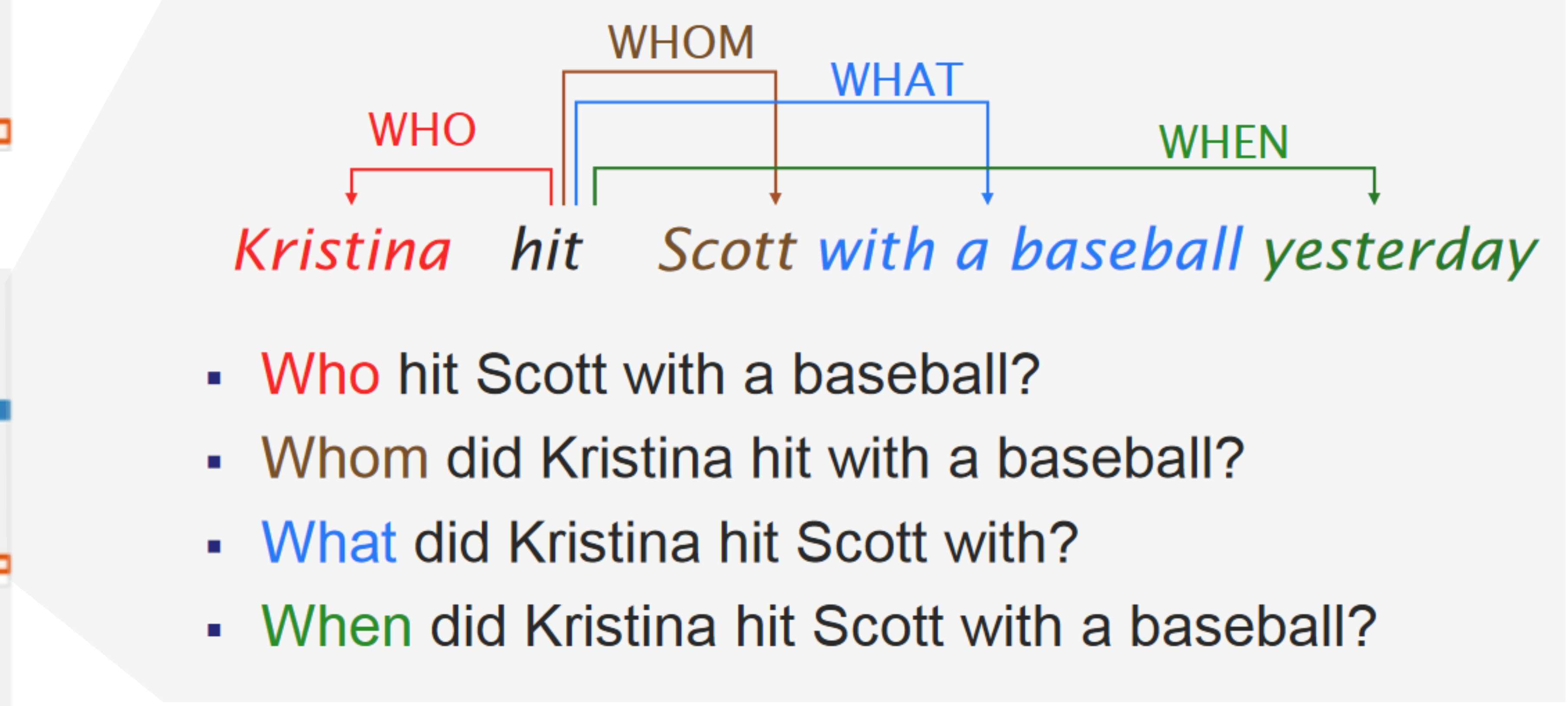


Image source: https://link.springer.com/chapter/10.1007/978-3-031-28238-6_17

BERT rediscovers the Classical NLP Pipeline

The model learns about
- Part-of-speech tags
- entities
- Semantic role labelling
- Coreferences
- ...
-





Tenney, I., Das, D., & Pavlick, E. (2019). BERT Rediscovers the Classical NLP Pipeline. Annual Meeting of the Association for Computational Linguistics.

The effect of pre-training

With pre-training BERT finds **wider minima** during fine-tuning

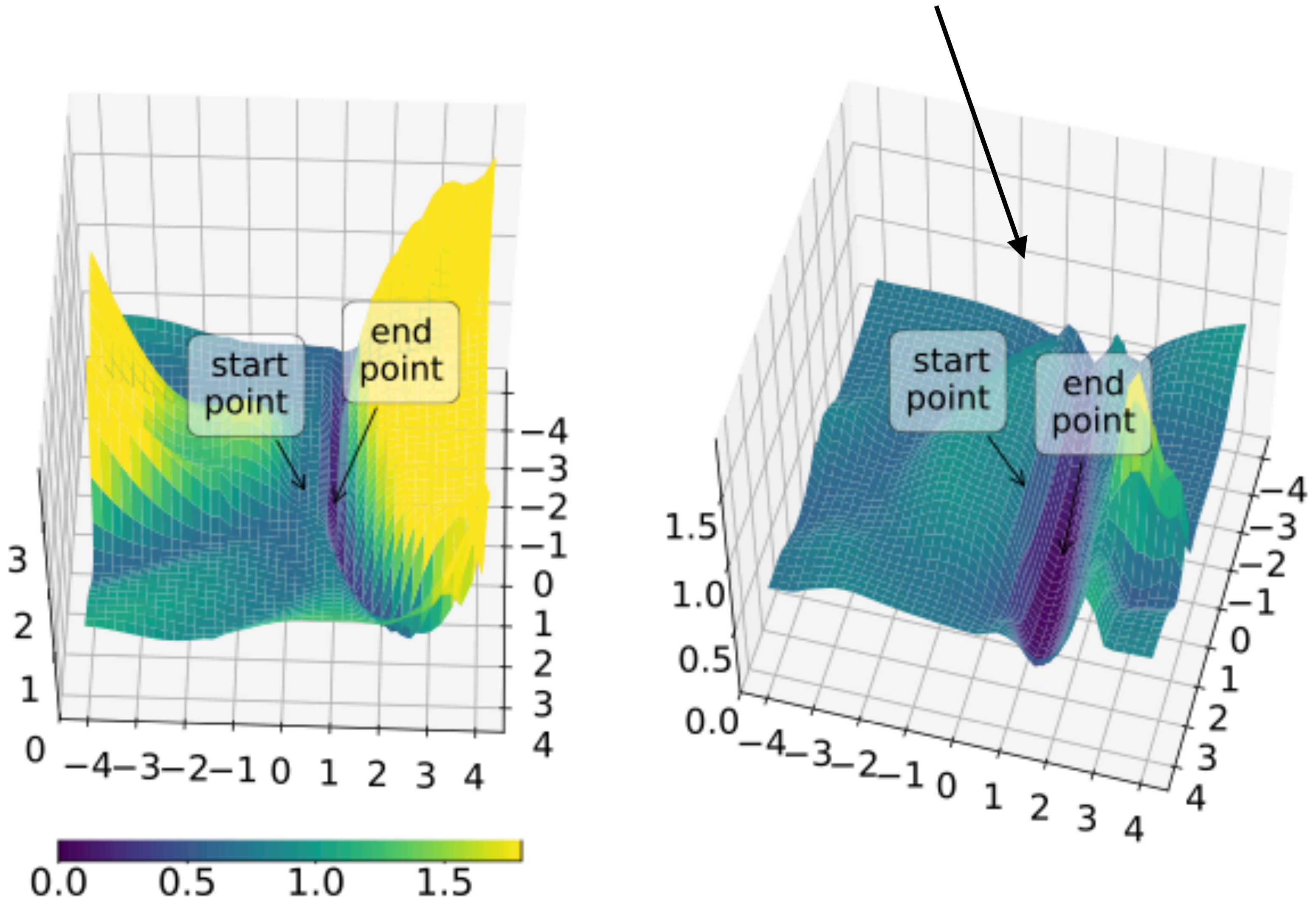


Figure 5: Pre-trained weights help BERT find wider optima in fine-tuning on MRPC (right) than training from scratch (left) (Hao et al., 2019)

- https://storage.googleapis.com/bert-wsd-vis/demo/index.html?#word=die
- https://visbert.demo.datexis.com/
-

Discussion

- Your boss have asked you to solve task X to the best of you ability, he has given you the training data
  - How to select the “best” model for your use-case?
  - What if you have limited compute?

Discussion

- Your boss have asked you to solve task X to the best of you ability, he has given you the training data
  - How to select the “best” model for your use-case?
  - What if you have limited compute?
  - What if you don’t have the training data?

Next Class

- Back to generative model
- “How do we make the models less sensitive to the prompt?”
  - Instruction tuning
  - Reinforcement learning for Human feedback