

Masked Language Models

Wei Xu

(many slides from Greg Durrett)

What's new? Jev model - released 9/15/2026

jev safeai

Artificial intelligence

Jeev

Chatgpt

Jev typesafe

Ai model

Automation

Llm

Llms

Typesafe ai

Faster Cheaper Hallucinate

What Is Jev? TypeSafe AI's System...
Zapier

200x Faster
400x Cheaper
Never Hallucinates
Jev
Video

5 Practical Use Cases for Jev
Jev is a type-safe AI system built for fast, structured decisions. Here are 5 real use cases worth exploring.
01 Agentic Workflows
02 Routing & Triage
03 Large Scale Classification
04 Scoring & Guardrails
05 Real-Time Decisions
LLMs generate. Jev decides. Code acts.
Facebook

JEV: SYSTEM ONE
5 days ago
Jev Explained: The Fast, Cheap...
Medium

Do you know what Jev AI is?
YES NO
Jev / TypesafeAI is revolutionary as...
Reddit

What is Jev??
release that came out a...
Video

Jev in the Wild: 60 Case Studies for...
4 days ago
NamoClaw

Are you a Qwen Model?
YES NO
Jev / TypesafeAI is revolutionary as...
Reddit

I BUILT 3 APPS WITH JEV AI
END OF UNSTRUCTURED LLMs?
Video - 6 days ago
I Built 3 Apps with Jev from TypeSafe...
YouTube

JEV WENT INTO EARLY
Could Jev Be The Fast Filter For...

The cheapest AI model has a catch
JEV
238x
CHEAPER THAN FABLE

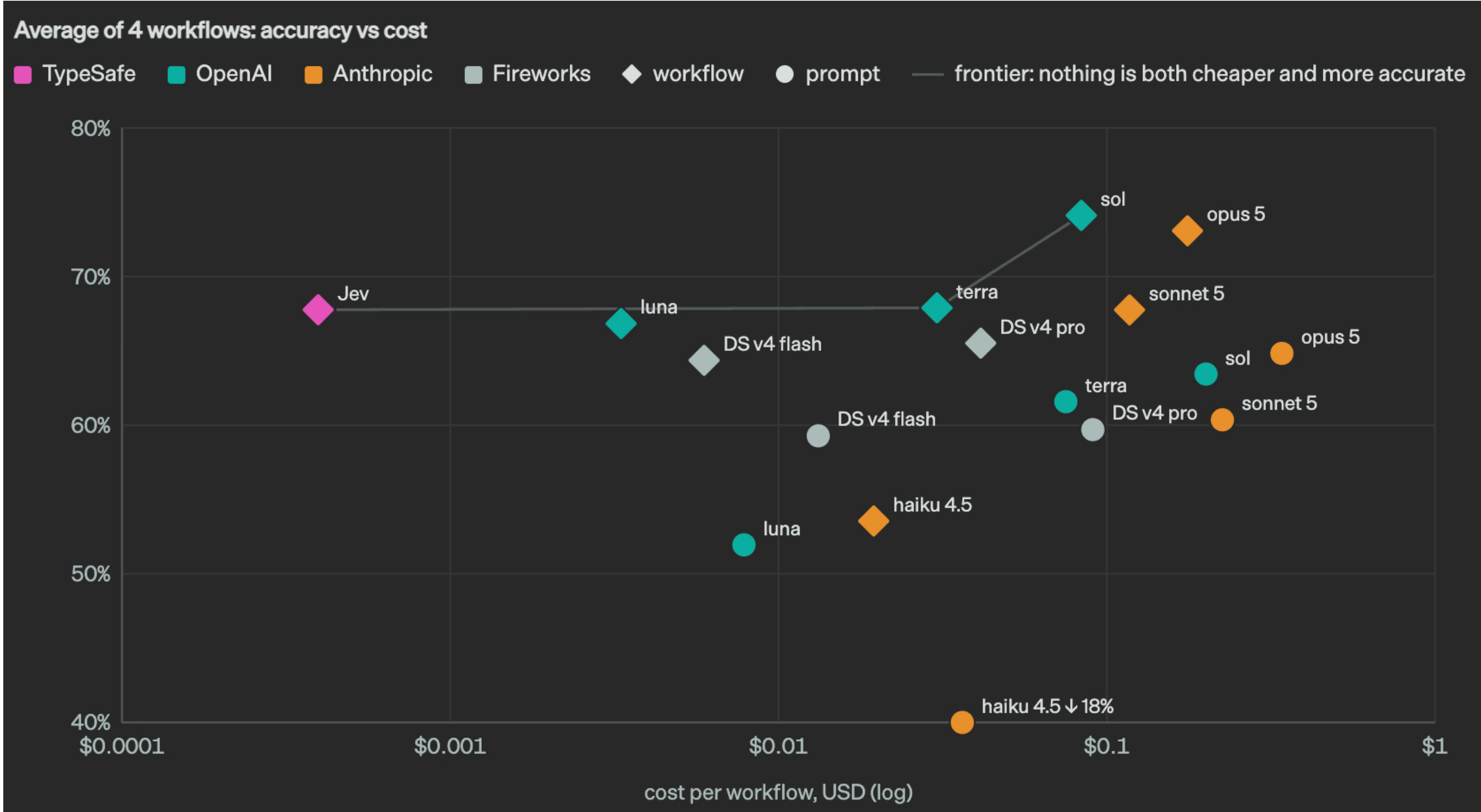
Does the word "Seventeen" contains a total of 3 e's?
TypeSafe Jev Explained: Fast, Type Safe AI

JEV: AI FOR DECISIONS
TypeSafe AI's first public model returns typed answers — with probabilities and confidence.
BUILT FOR SOFTWARE
LAUNCH SNAPSHOT
20-200x FASTER

Type safe AI released Jev. T
Instagram

Jev and the Rise of Type-S
LinkedIn

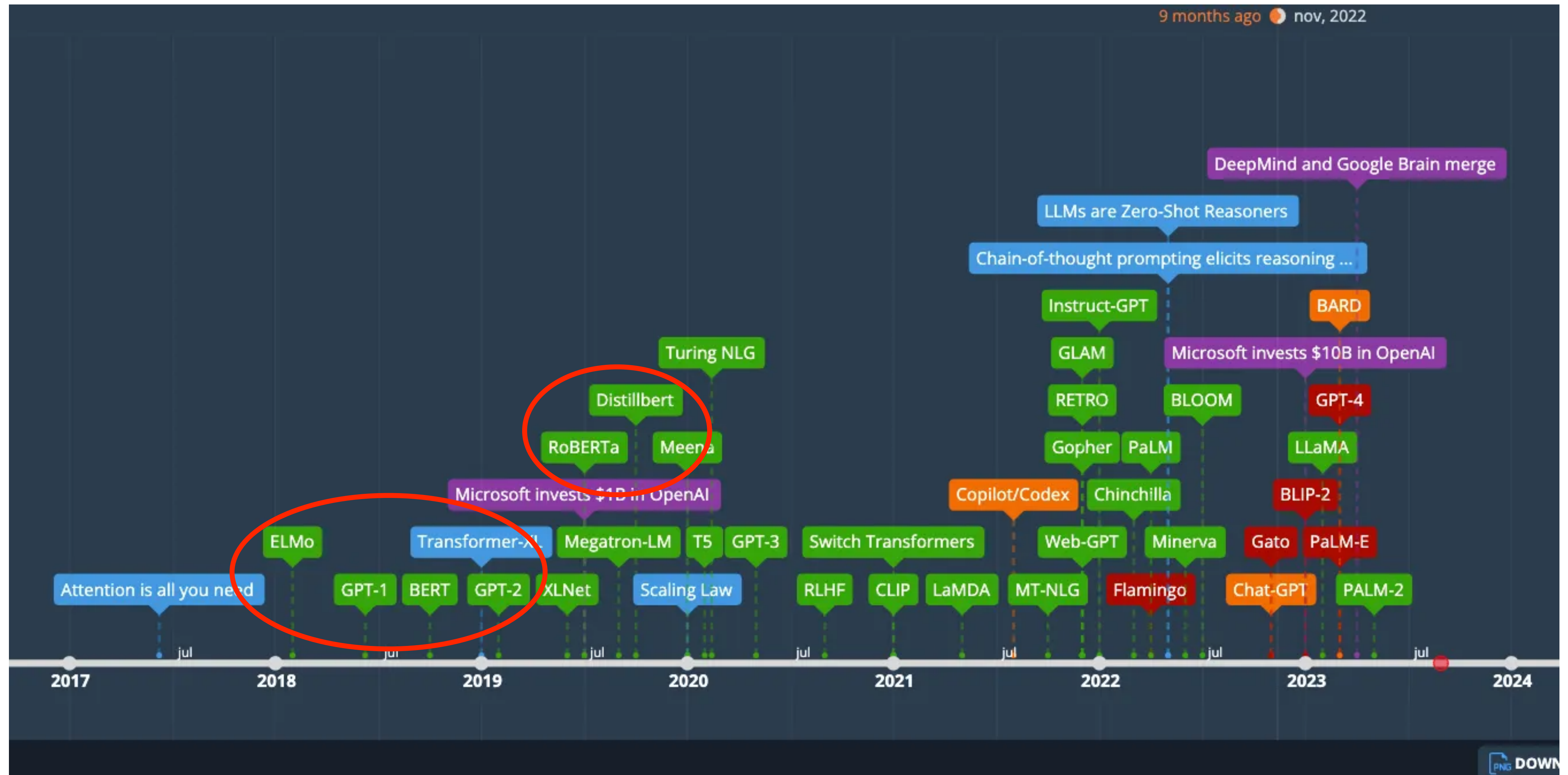
What's new? Jev model - released 9/15/2026



This Lecture

- ▶ Encoder-based LMs (e.g., ELMo, BERT)
- ▶ Next Lectures:
 - ▶ Decoder-based LMs (e.g., GPT-1/GPT-2/GPT-3)
 - ▶ Encoder-decoder LMs (e.g., BART, T5)
 - ▶ Knowledge Distillation (e.g., DistillBERT)

The beginning of LLMs



Readings

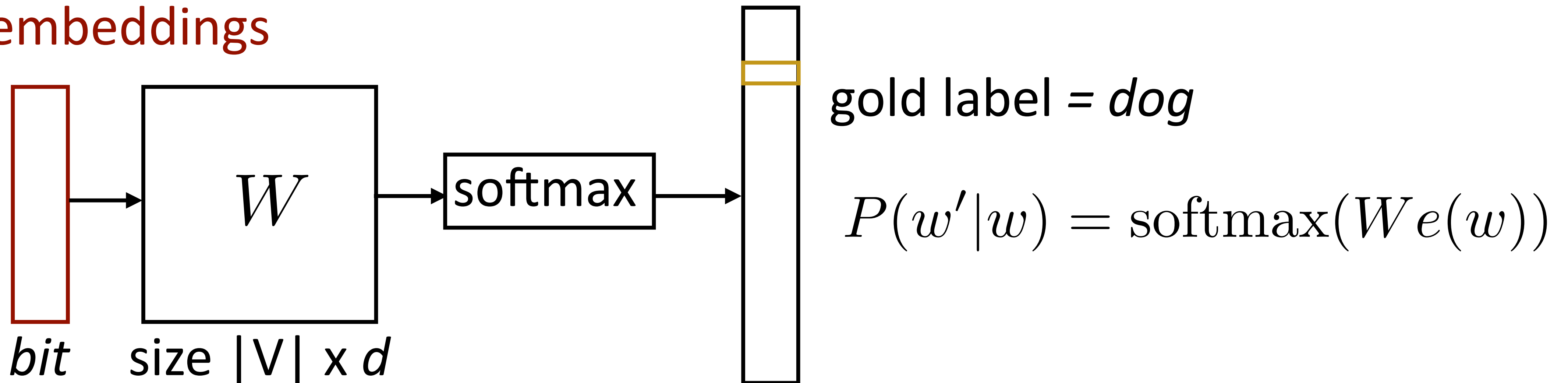
- ▶ Readings —
 - ▶ J+M 9
 - ▶ ELMo by Peters et al.
<https://aclanthology.org/N18-1202.pdf>
 - ▶ BERT by Devlin et al.
<https://aclanthology.org/N19-1423.pdf>

Recall: word2vec (Skip-Gram)

- Predict one word of context from word

d-dimensional
word embeddings

the dog bit the man



- Another training example: *bit* $\rightarrow$ *the*
- Parameters: $d \times |V|$ **vectors**, $|V| \times d$ output parameters (W) (also usable as vectors!)

ELMo

(released 2018/2, early version 2017/10)

ELMo

Deep contextualized word representations

Matthew E. Peters[†], Mark Neumann[†], Mohit Iyyer[†], Matt Gardner[†],
{matthewp, markn, mohiti, mattg}@allenai.org

Christopher Clark*, Kenton Lee*, Luke Zettlemoyer^{†*}
{csquared, kentonl, lsz}@cs.washington.edu

[†]Allen Institute for Artificial Intelligence

*Paul G. Allen School of Computer Science & Engineering, University of Washington

Abstract

We introduce a new type of *deep contextualized* word representation that models both (1) complex characteristics of word use (e.g., syntax and semantics), and (2) how these uses vary across linguistic contexts (i.e., to model polysemy). Our word vectors are learned functions of the internal states of a deep bidirectional language model (biLM), which is pre-trained on a large text corpus. We show that these representations can be easily added to existing models and significantly improve the state of the art across six challenging NLP problems, including question answering, textual entailment and sentiment analysis. We also present an analysis showing that exposing the deep internals of the pre-trained network is crucial, allowing downstream models to mix different types of semi-supervision signals.

1 Introduction

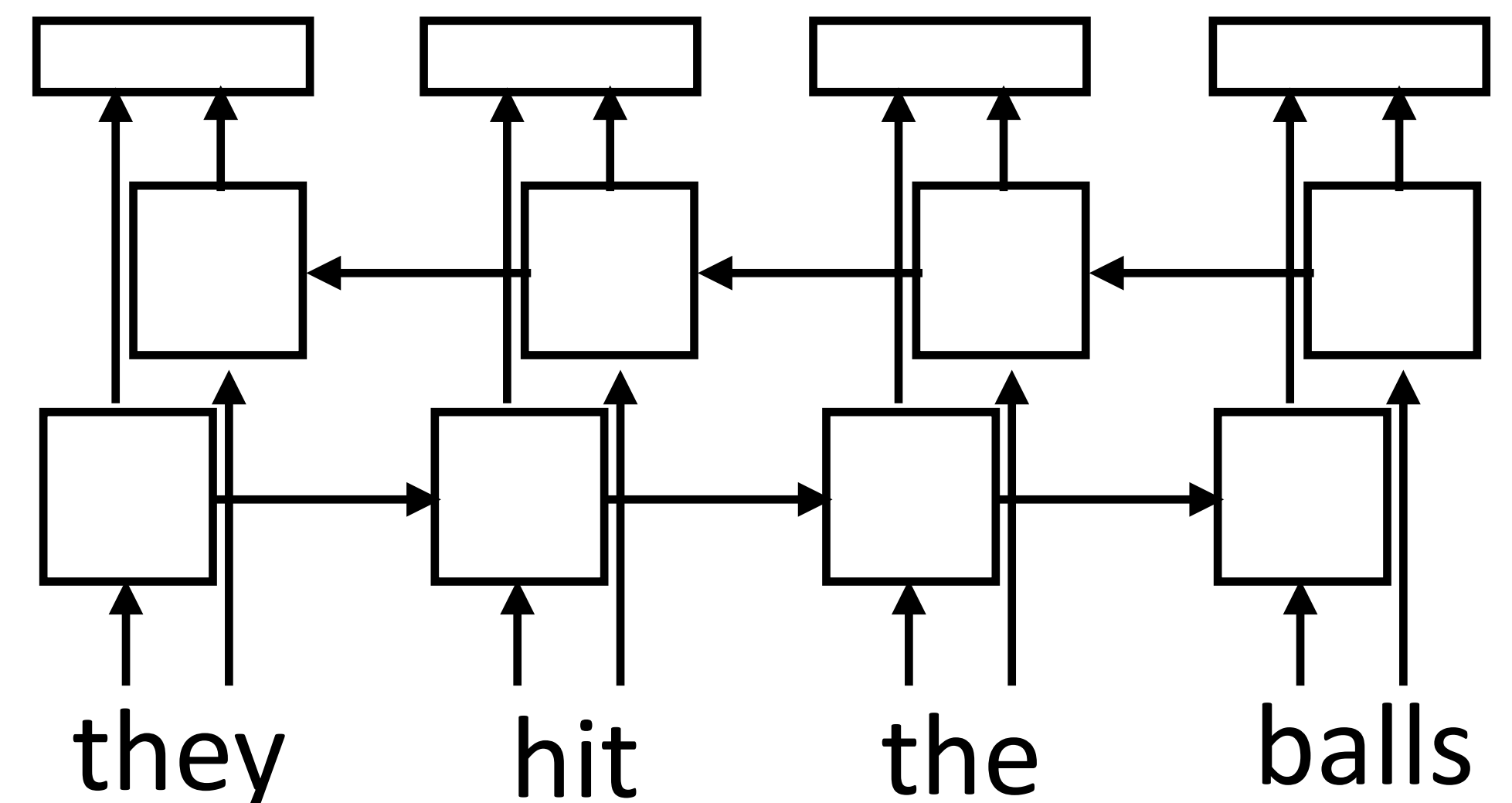
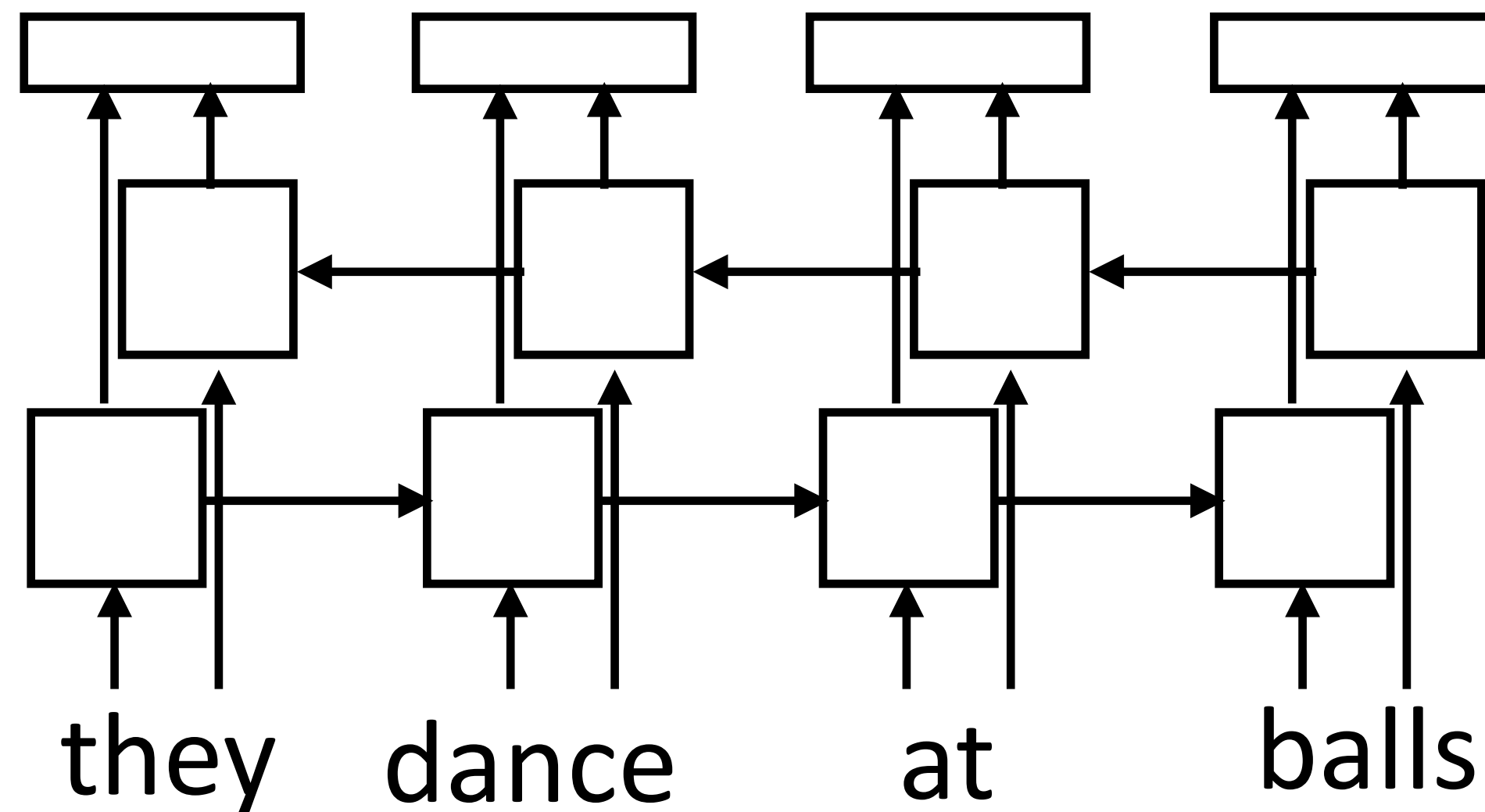
Pre-trained word representations (Mikolov et al., 2013; Pennington et al., 2014) are a key compo-

guage model (LM) objective on a large text corpus. For this reason, we call them ELMo (Embeddings from Language Models) representations. Unlike previous approaches for learning contextualized word vectors (Peters et al., 2017; McCann et al., 2017), ELMo representations are deep, in the sense that they are a function of all of the internal layers of the biLM. More specifically, we learn a linear combination of the vectors stacked above each input word for each end task, which markedly improves performance over just using the top LSTM layer.

Combining the internal states in this manner allows for very rich word representations. Using intrinsic evaluations, we show that the higher-level LSTM states capture context-dependent aspects of word meaning (e.g., they can be used without modification to perform well on supervised word sense disambiguation tasks) while lower-level states model aspects of syntax (e.g., they can be used to do part-of-speech tagging). Simultaneously exposing all of these signals is highly bene-

Context-dependent Embeddings

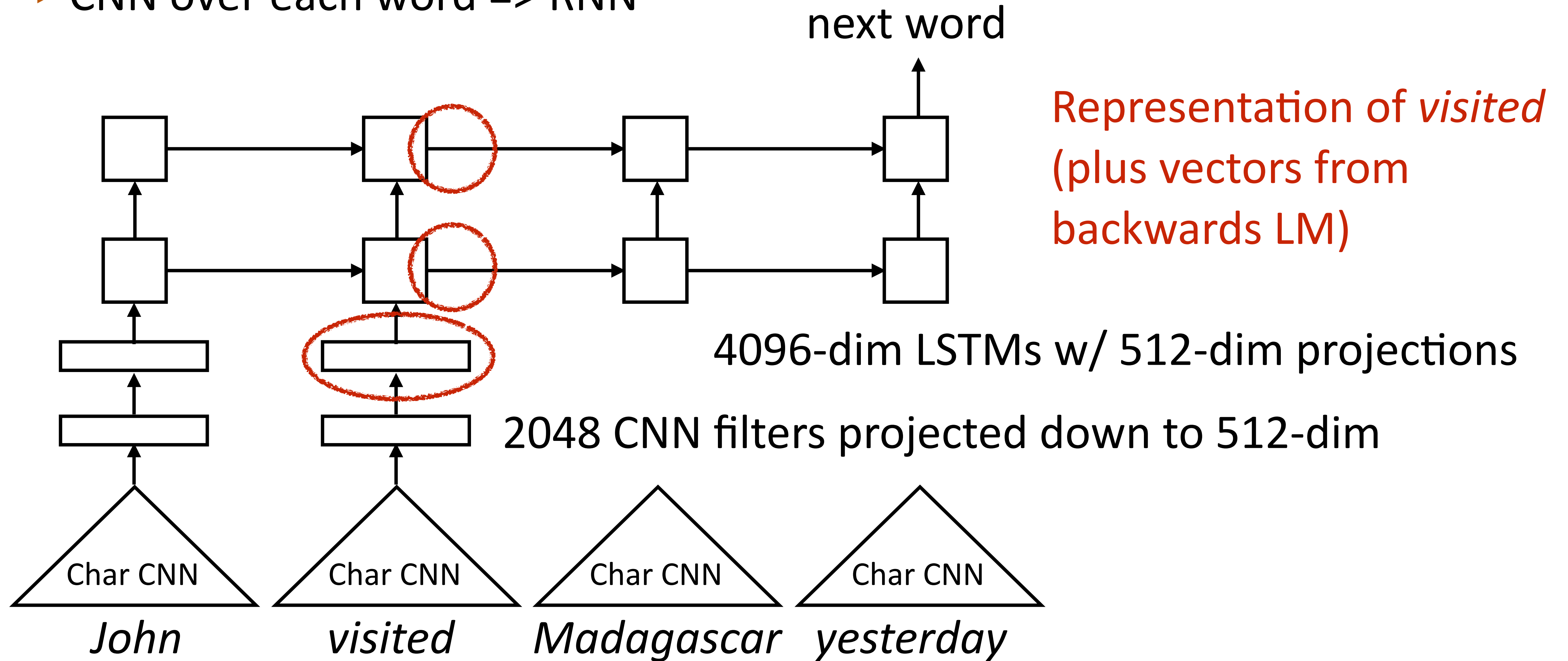
- ▶ How to handle different word senses? One vector for *balls*



- ▶ Train a neural language model to predict the next word given previous words in the sentence, use its internal representations as word vectors
 - ▶ *Context-sensitive* word embeddings: depend on rest of the sentence
 - ▶ *Huge* improvements across nearly all NLP tasks over word2vec & GloVe
- ELMo - Peters et al. (2018)

ELMo

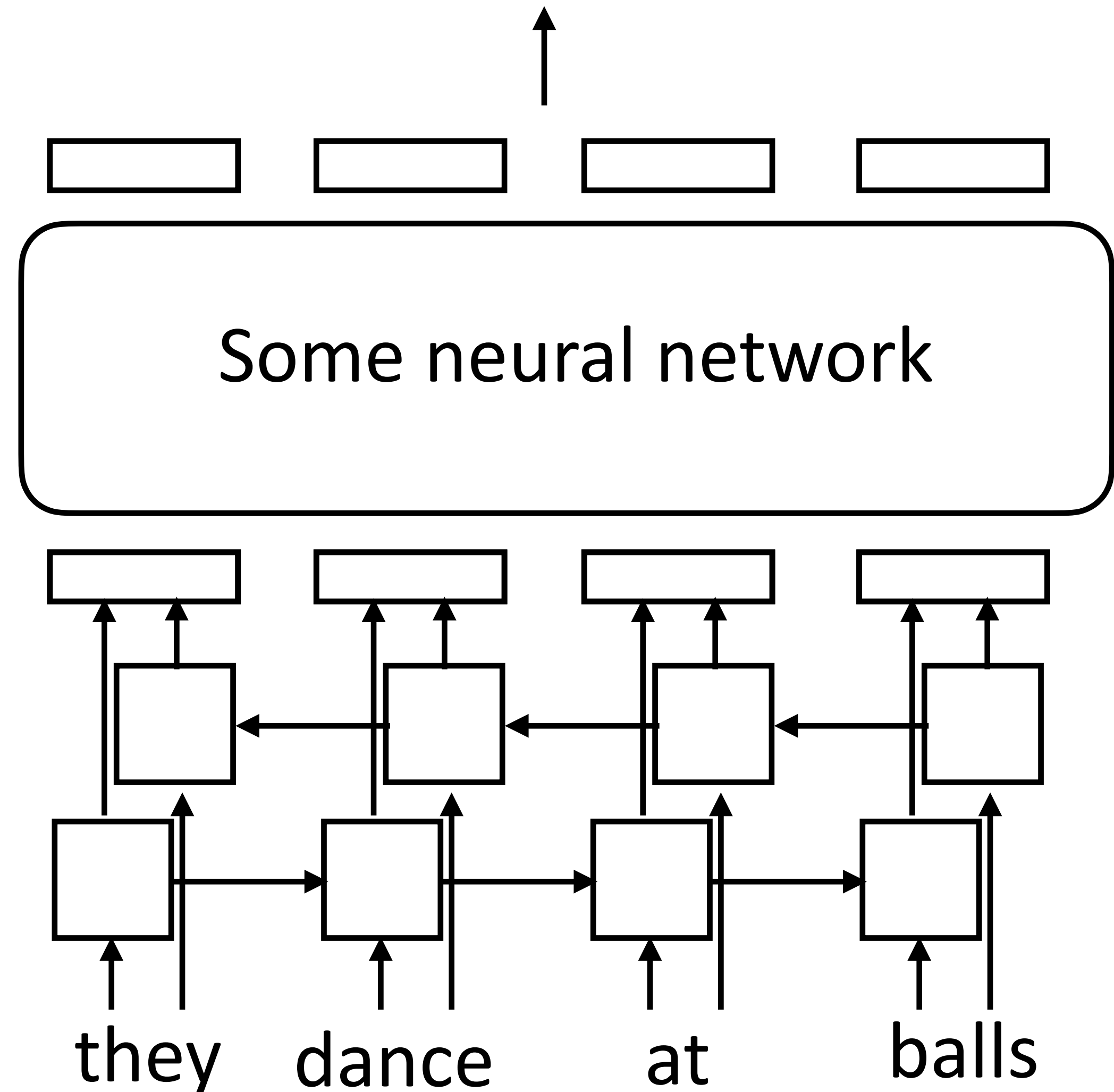
- ▶ CNN over each word => RNN



How to apply ELMo?

- ▶ Take those embeddings and feed them into whatever architecture you want to use for your task
- ▶ *Frozen* embeddings: update the weights of your network but keep ELMo's parameters frozen
- ▶ *Fine-tuning*: backpropagate all the way into ELMo when training your model

Task predictions (sentiment, etc.)



Results: Frozen ELMo

TASK	PREVIOUS SOTA		OUR BASELINE	ELMo + BASELINE	INCREASE (ABSOLUTE/ RELATIVE)
SQuAD	Liu et al. (2017)	84.4	81.1	85.8	4.7 / 24.9%
SNLI	Chen et al. (2017)	88.6	88.0	88.7 $\pm$ 0.17	0.7 / 5.8%
SRL	He et al. (2017)	81.7	81.4	84.6	3.2 / 17.2%
Coref	Lee et al. (2017)	67.2	67.2	70.4	3.2 / 9.8%
NER	Peters et al. (2017)	91.93 $\pm$ 0.19	90.15	92.22 $\pm$ 0.10	2.06 / 21%
SST-5	McCann et al. (2017)	53.7	51.4	54.7 $\pm$ 0.5	3.3 / 6.8%

- Massive improvements across 5 benchmark datasets: question answering, natural language inference, semantic role labeling, coreference resolution, named entity recognition, and sentiment analysis

ELMo

- ▶ Key idea: language models can allow us to form useful word representations in the same way word2vec did
- ▶ Take a powerful language model, train it on large amounts of data, then use those contextualized representations in downstream tasks
 - ▶ Data: Wikipedia, books, crawled stuff from the web, ...

OpenReview

OpenReview.net

Search OpenReview...



Login

Deep contextualized word representations



Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, Luke Zettlemoyer

02 Feb 2018 (modified: 02 Mar 2025) ICLR 2018 Conference Blind Submission Readers: Everyone [Show Bibtex](#) [Show Revisions](#)

Keywords: representation learning, contextualized word embeddings

TL;DR: We introduce a new type of deep contextualized word representation that significantly improves the state of the art for a range of challenging NLP tasks.

Abstract: We introduce a new type of deep contextualized word representation that models both (1) complex characteristics of word use (e.g., syntax and semantics), and (2) how these uses vary across linguistic contexts (i.e., to model polysemy). Our word vectors are learned functions of the internal states of a deep bidirectional language model (biLM), which is pretrained on a large text corpus. We show that these representations can be easily added to existing models and significantly improve the state of the art across six challenging NLP problems, including question answering, textual entailment and sentiment analysis. We also present an analysis showing that exposing the deep internals of the pretrained network is crucial, allowing downstream models to mix different types of semi-supervision signals.

Community Implementations: [\[x\]](#) 43 code implementations

Withdrawal: Confirmed

Reply Type: Author: Visible To: Hidden From:

17 Replies

[\[-\]](#) NLP conference

ICLR 2018 Conference Paper759 Authors

02 Feb 2018, 21:17 ICLR 2018 Conference Paper759 Official Comment Readers: Everyone [Show Revisions](#)

Comment: As suggested by the meta reviewer, the empirical results in this paper are better suited for a NLP conference then ICLR. As a result, we are withdrawing it from ICLR. Thank you to the Chairs for your consideration and paper acceptance.

[\[-\]](#) ICLR 2018 Conference Acceptance Decision

ICLR 2018 Conference Program Chairs

29 Jan 2018, 13:11 (modified: 29 Jan 2018, 16:08) ICLR 2018 Conference Acceptance Decision Readers: Everyone [Show Revisions](#)

Decision: Accept (Poster)

Comment: This is a good paper that presents state-of-the-art results on a number of challenging NLP tasks. The idea is fairly simple and clean, I therefore expect it to get adopted in the community. It also seems to work across several tasks, which is nice. At the same time it is fairly simple (train an LSTM language model and use representations from all levels of the LSTM in the input or output layer of a supervised task of interest) and hence may be more appropriate for an NLP conference than ICLR?

It is a good paper and it will get cited even though the ML contributions are modest. Accept due to the strong empirical results.

BERT

BERT

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

Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova

Google AI Language

{jacobdevlin, mingweichang, kentonl, kristout}@google.com

Abstract

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

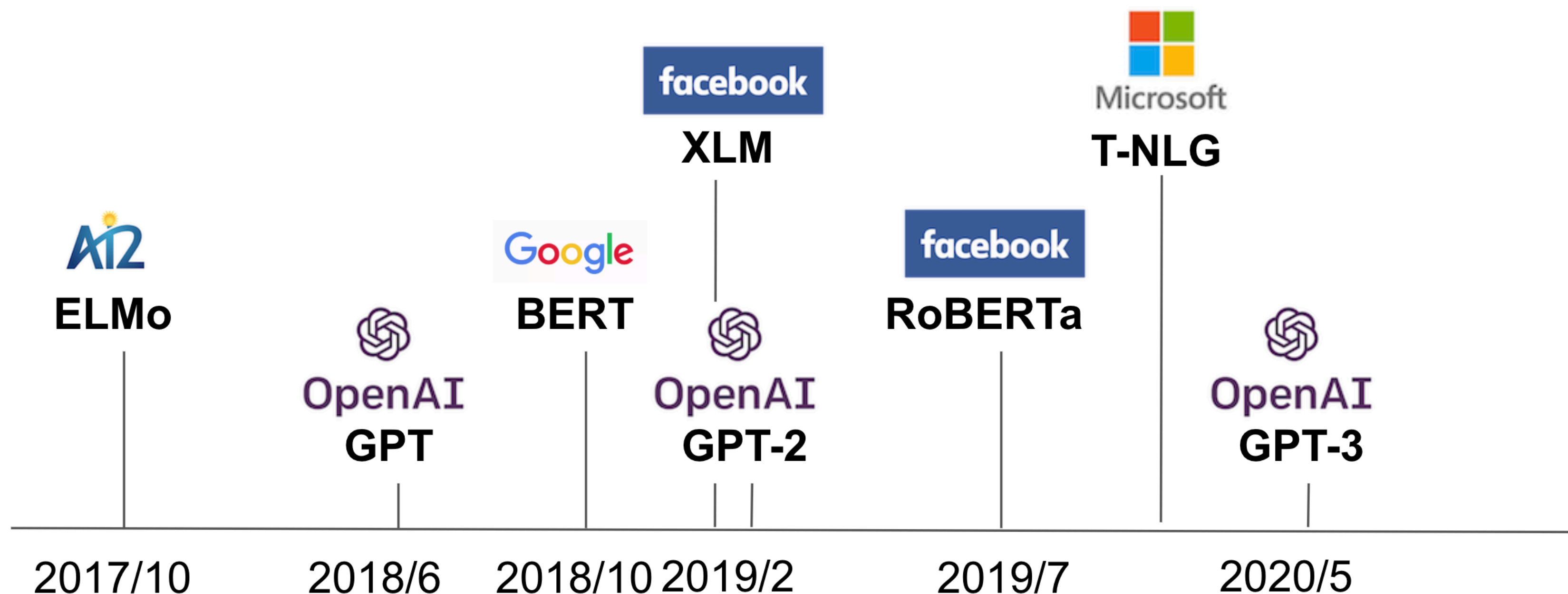
BERT is conceptually simple and empirically powerful. It obtains new state-of-the-art results on eleven natural language processing tasks, including pushing the GLUE score to 80.5% (7.7% point absolute improvement), MultiNLI accuracy to 86.7% (4.6% absolute improvement), SQuAD v1.1 question answering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

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

We argue that current techniques restrict the power of the pre-trained representations, especially for the fine-tuning approaches. The major limitation is that standard language models are unidirectional, and this limits the choice of architectures that can be used during pre-training. For example, in OpenAI GPT, the authors use a left-to-right architecture, where every token can only attend to previous tokens in the self-attention layers

Contextual Word Embeddings

- ▶ AI2 released ELMo in spring 2018, GPT (transformer-based) was released in summer 2018, BERT (transformer-based) came out October 2018
- ▶ aka Pre-trained Language Models



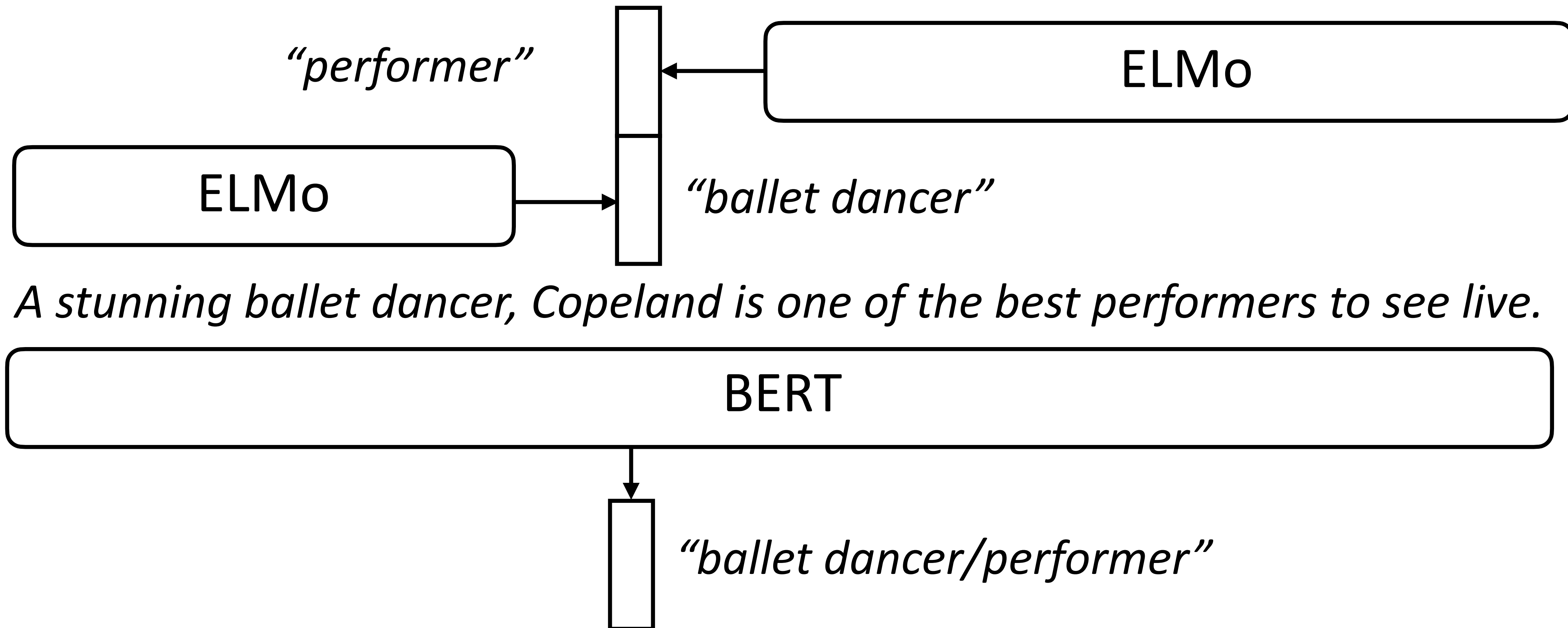
and many more ...

Contextual Word Embeddings

- ▶ AI2 released ELMo in spring 2018, GPT (transformer-based) was released in summer 2018, BERT (transformer-based) came out October 2018
- ▶ aka Pre-trained Language Models
- ▶ BERT's four major changes compared to ELMo:
 - ▶ Transformers instead of LSTMs (transformers in GPT as well)
 - ▶ “Truly” Bidirectional $\Leftrightarrow$ Masked LM objective instead of standard LM
 - ▶ Fine-tune instead of freeze at test time
 - ▶ Uses word pieces (subword tokenization)

BERT

- ELMo is a unidirectional model: we can concatenate two unidirectional models, but is this the right thing to do?

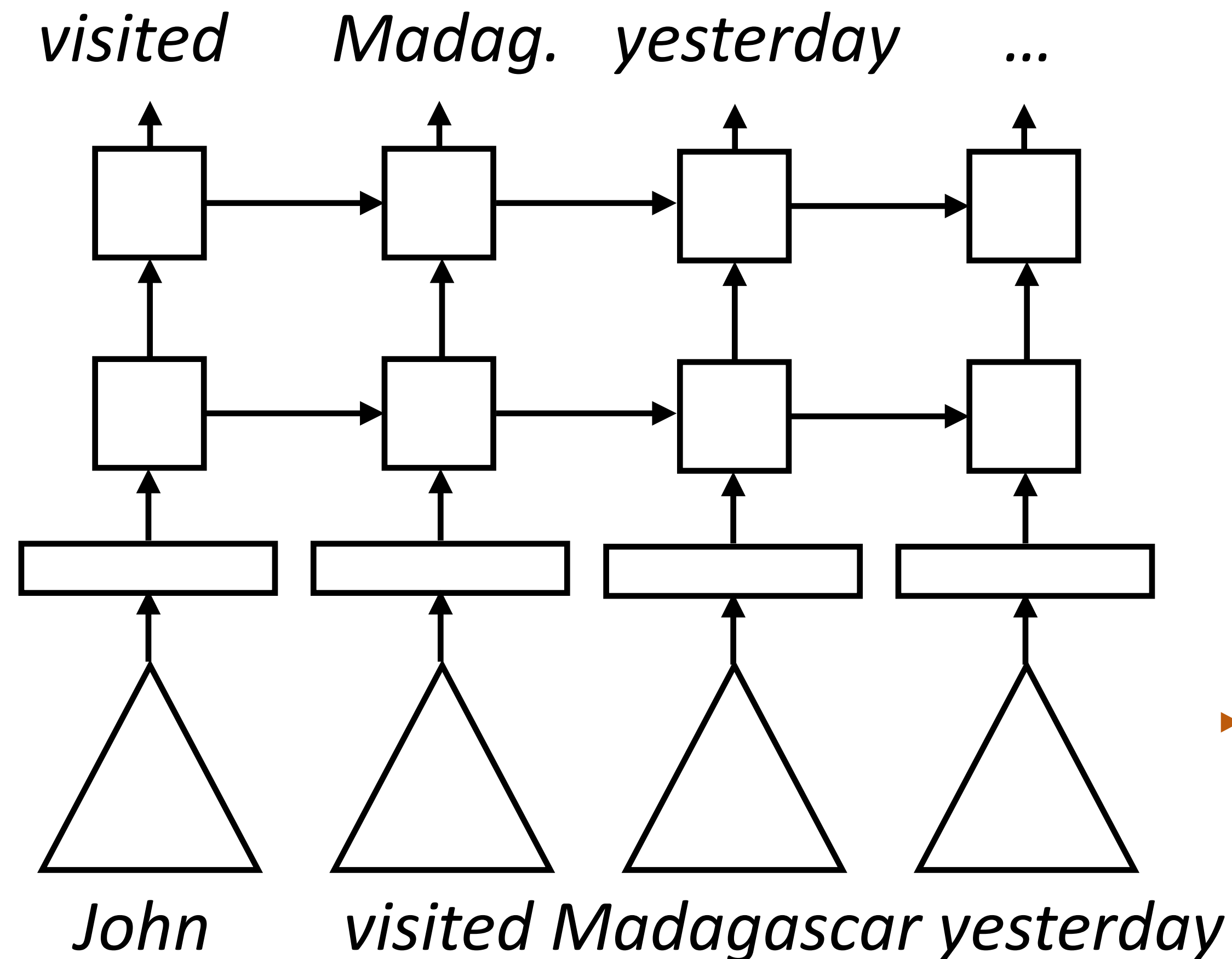


- ELMo looks at each direction in isolation; BERT looks at them jointly
Devlin et al. (2019)

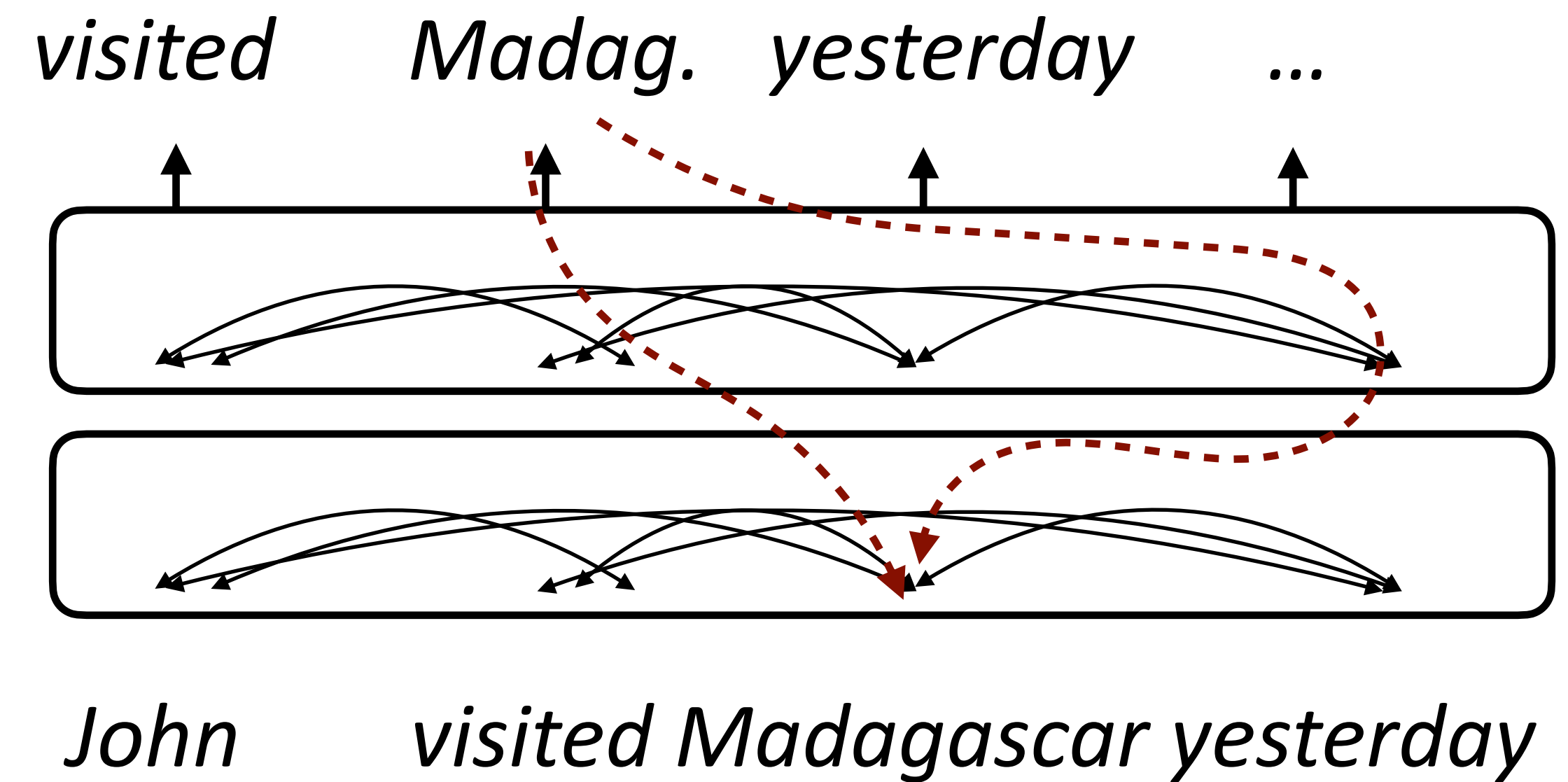
BERT

- How to learn a “deeply bidirectional” model? What happens if we just replace an LSTM with a transformer?

ELMo (Language Modeling)



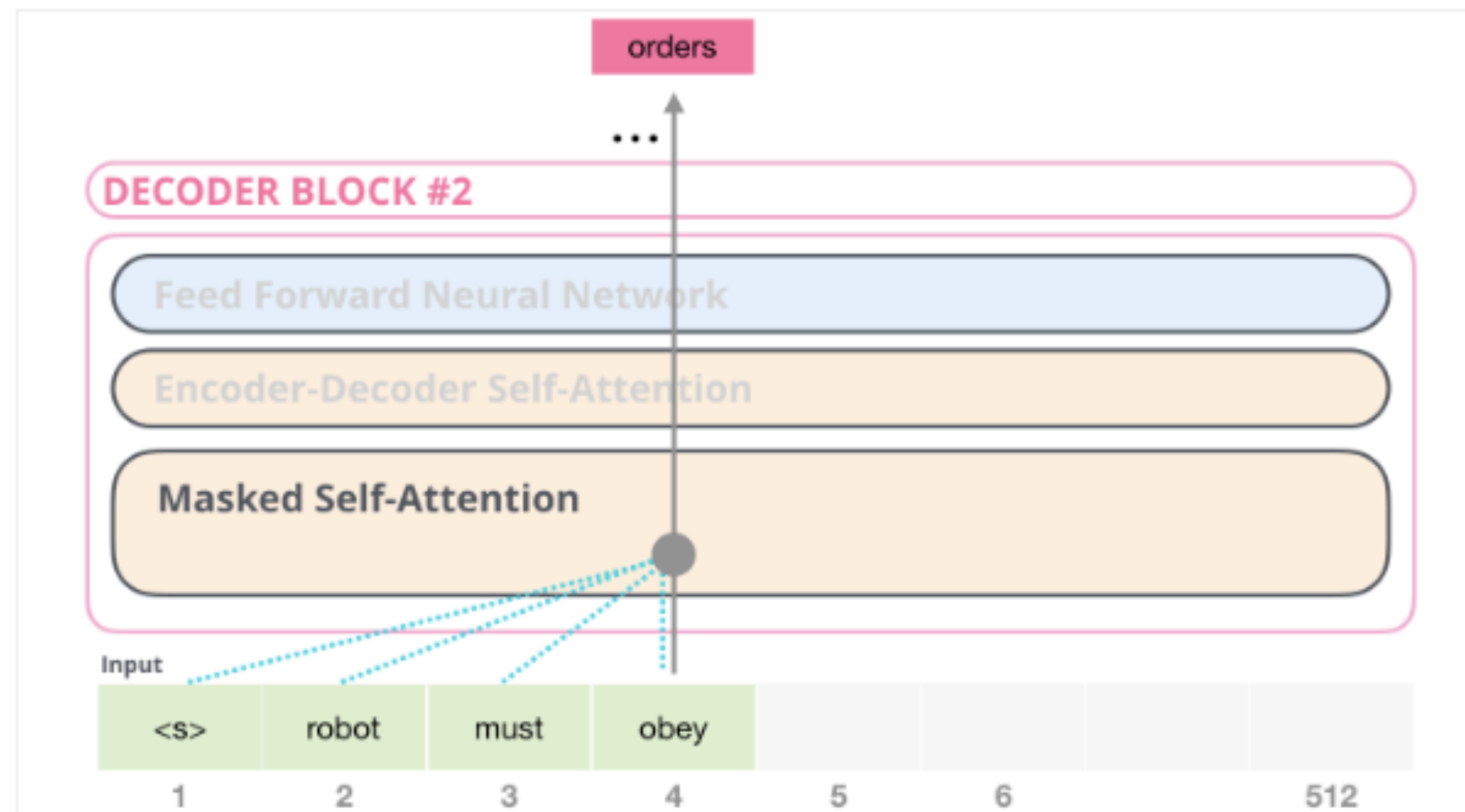
BERT



- Can do this by “one-sided” Transformer (masked self-attention), but “two-sided” Transformer encoder can cheat

(Preview) GPT

- ▶ Transformer with masked self-attention: each token can only attend to past tokens



Masked Language Modeling (MLM)

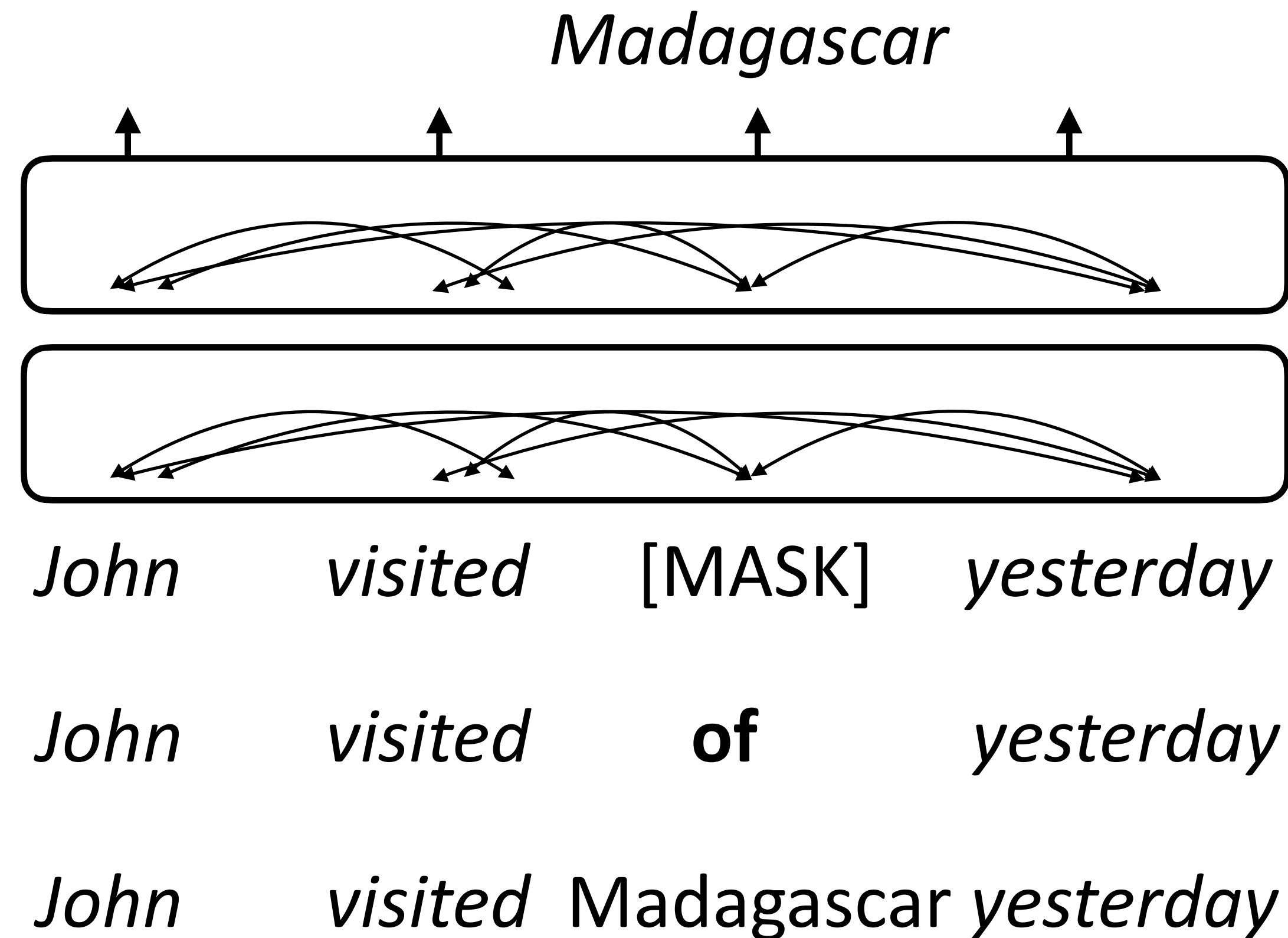
- ▶ How to prevent cheating? Next word prediction fundamentally doesn't work for bidirectional models, instead do *masked language modeling*

- ▶ BERT formula: take a chunk of text, predict 15% of the tokens

- ▶ For 80% (of the 15%), replace the input token with [MASK]

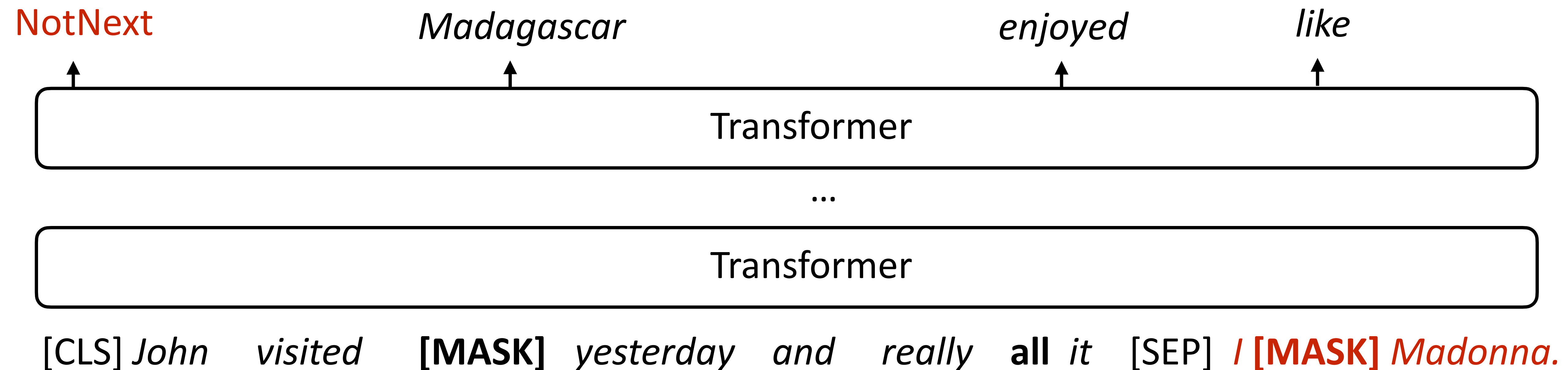
- ▶ For 10%, replace w/random

- ▶ For 10%, keep same



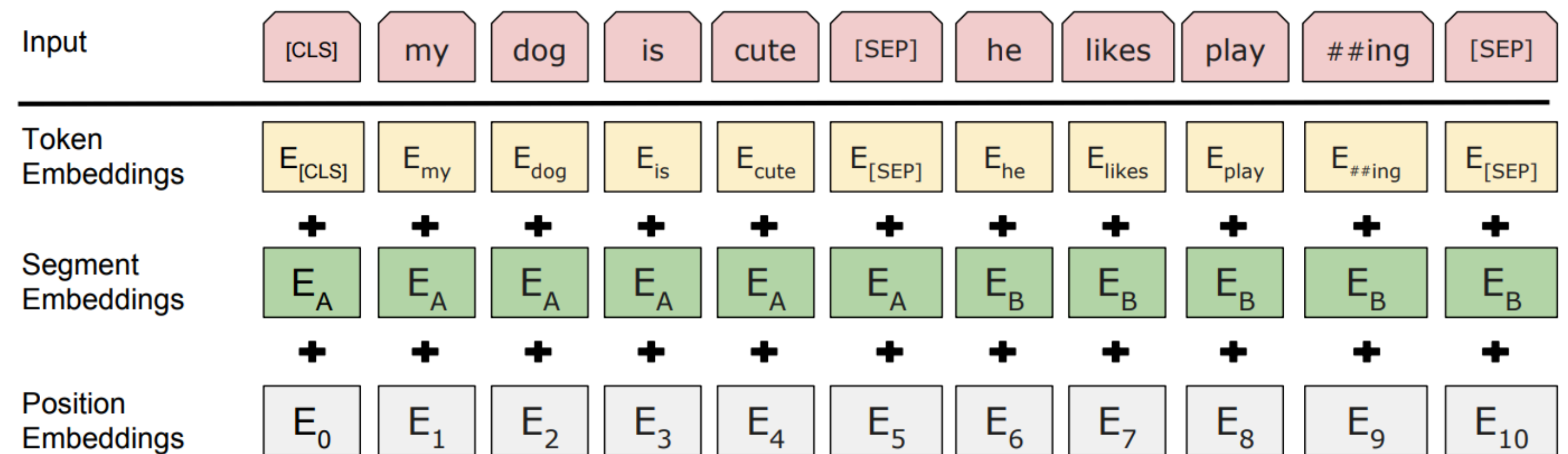
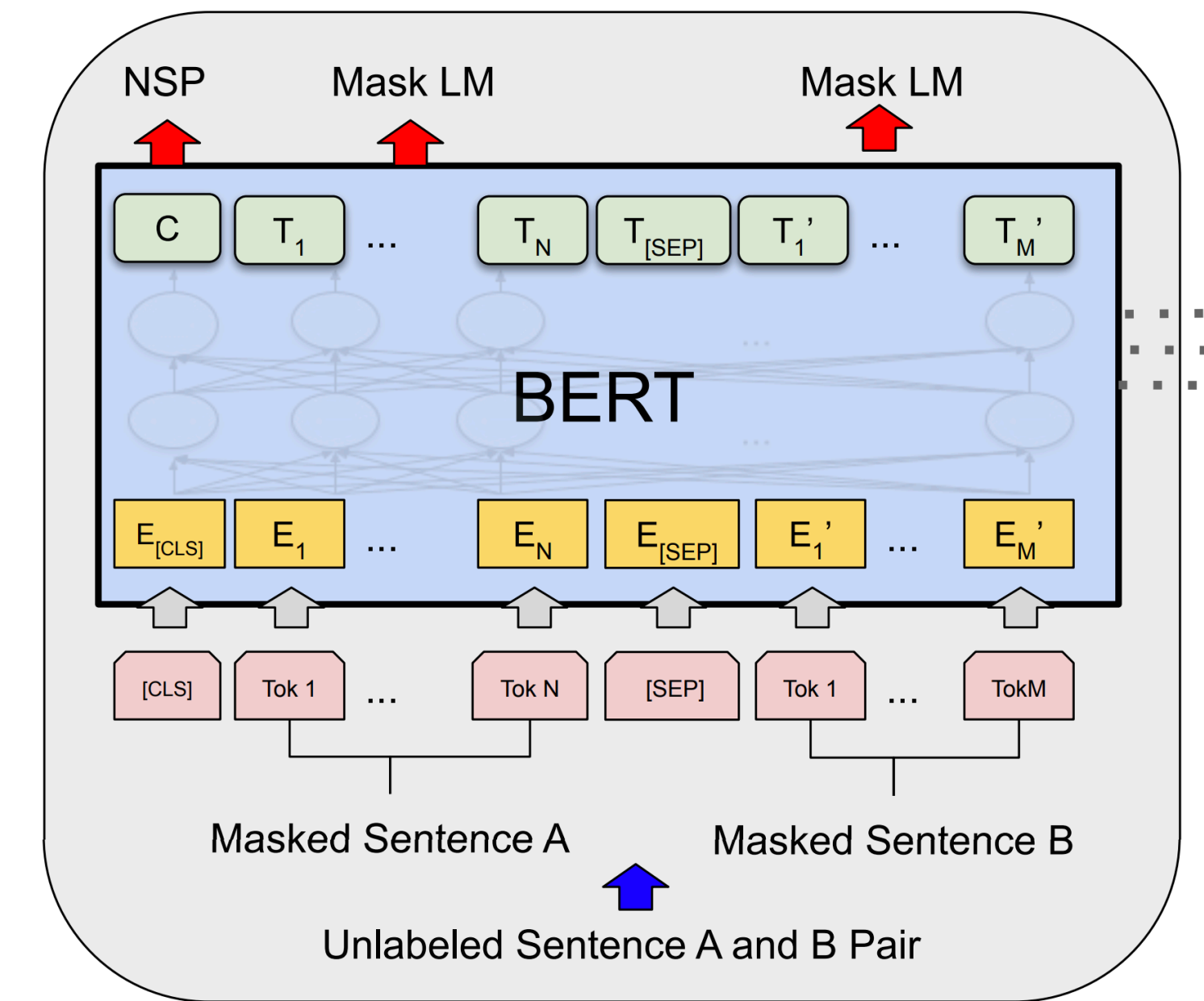
Next “Sentence” Prediction

- ▶ Input: [CLS] Text chunk 1 [SEP] Text chunk 2
- ▶ 50% of the time, take the true next chunk of text, 50% of the time take a random other chunk. Predict whether the next chunk is the “true” next
- ▶ BERT objective: masked LM + next sentence prediction



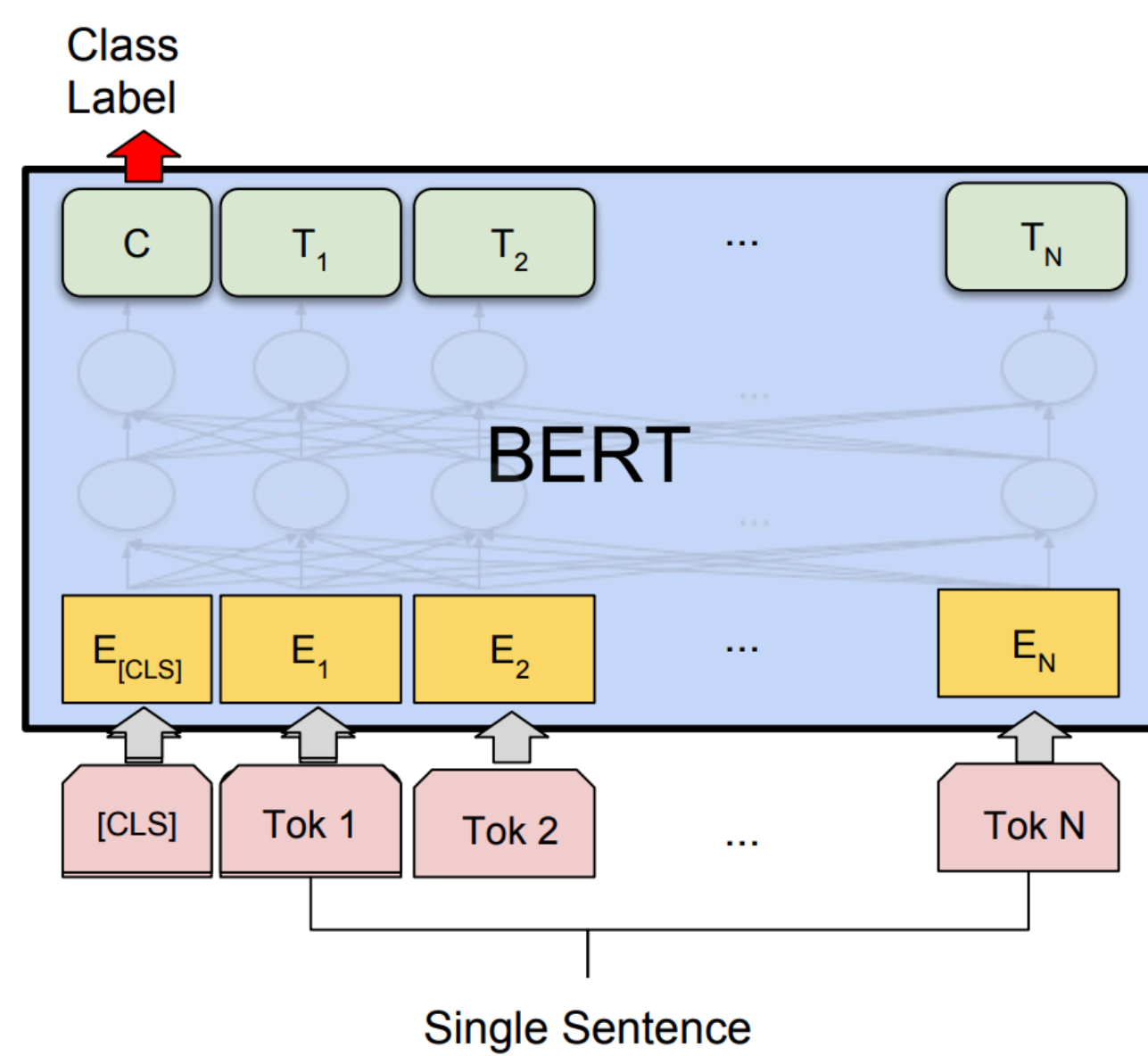
BERT Architecture

- ▶ BERT Base: 12 layers, 768-dim, 12 heads. Total params = 110M
- ▶ BERT Large: 24 layers, 1024-dim, 16 heads. Total params = 340M
- ▶ Positional embeddings and segment embeddings, 30k word pieces
- ▶ This is the model that gets **pre-trained** on a large corpus

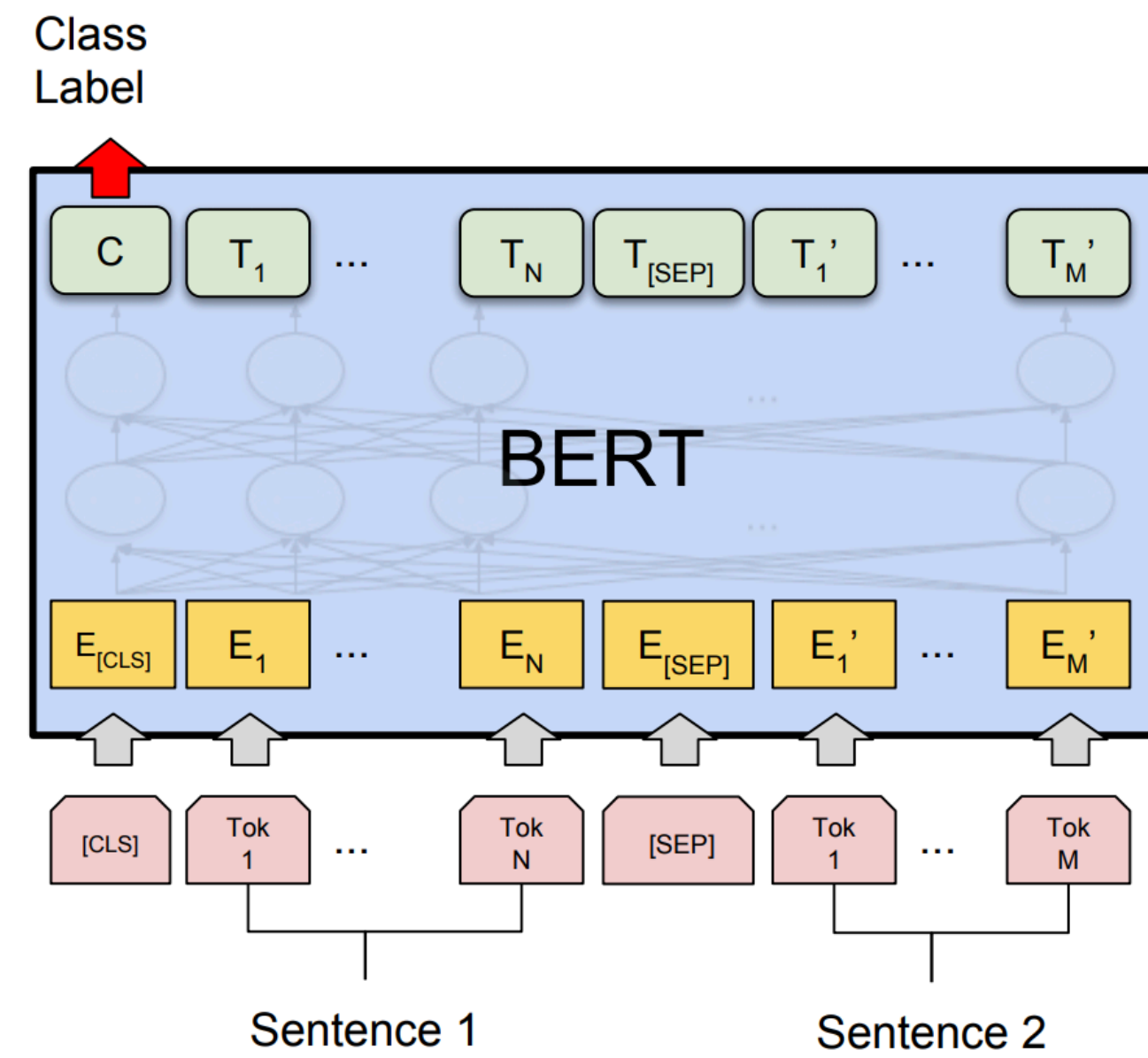


Devlin et al. (2019)

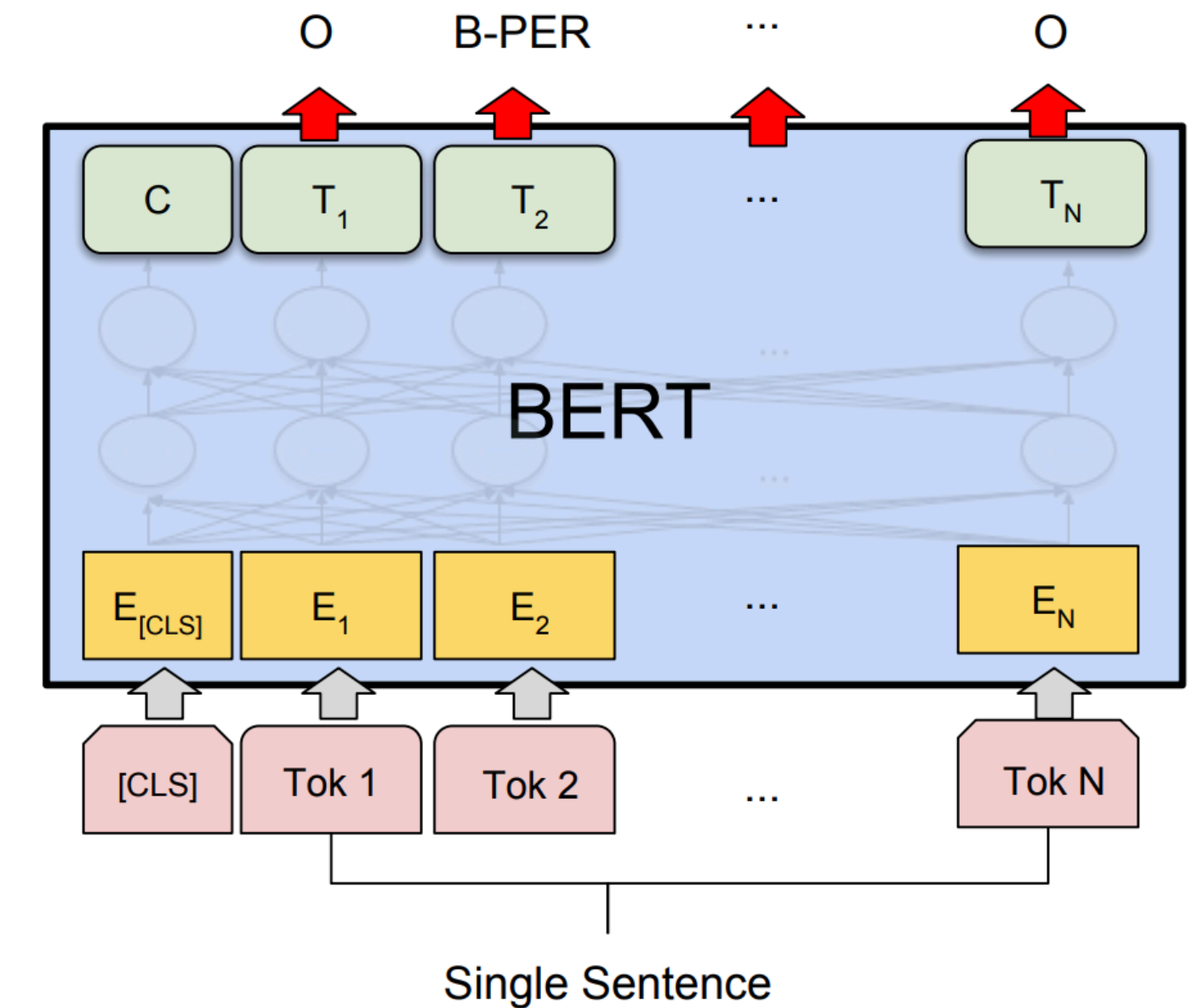
What can BERT do?



(b) Single Sentence Classification Tasks:
SST-2, CoLA



(a) Sentence Pair Classification Tasks:
MNLI, QQP, QNLI, STS-B, MRPC,
RTE, SWAG



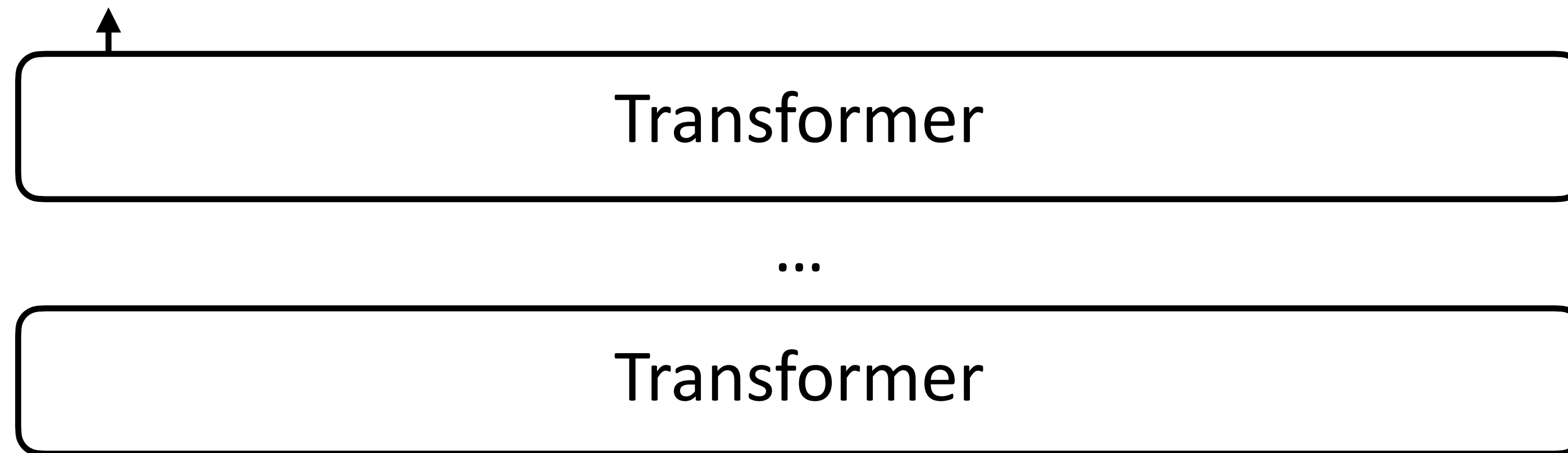
(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER

- ▶ CLS token is used to provide classification decisions
- ▶ Sentence pair tasks (entailment): feed both sentences into BERT
- ▶ BERT can also do tagging by predicting tags at each word piece

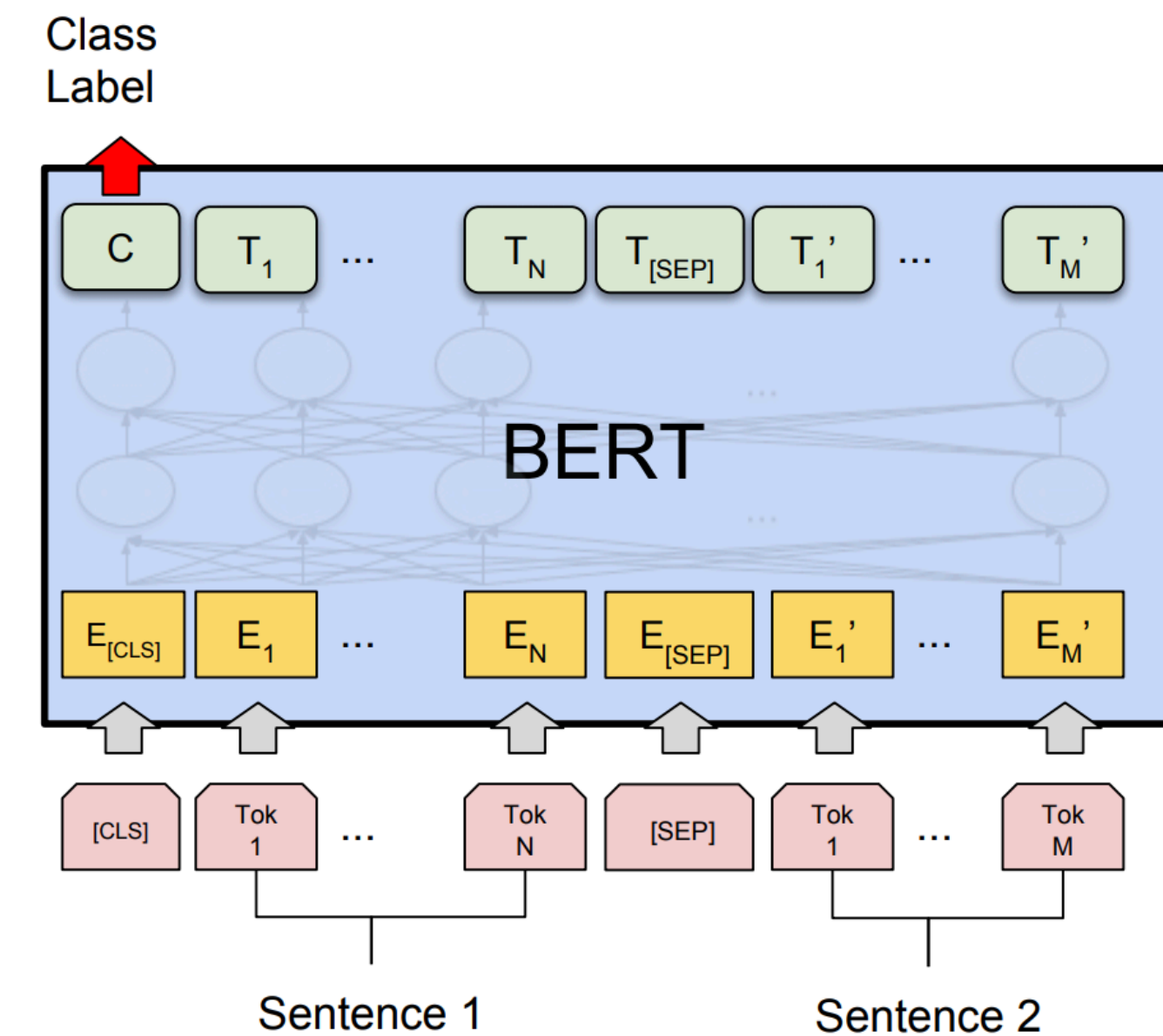
Devlin et al. (2019)

What can BERT do?

Entails (first sentence implies second one is true)



[CLS] A boy plays in the snow [SEP] A boy is outside



(a) Sentence Pair Classification Tasks:
MNLI, QQP, QNLI, STS-B, MRPC,
RTE, SWAG

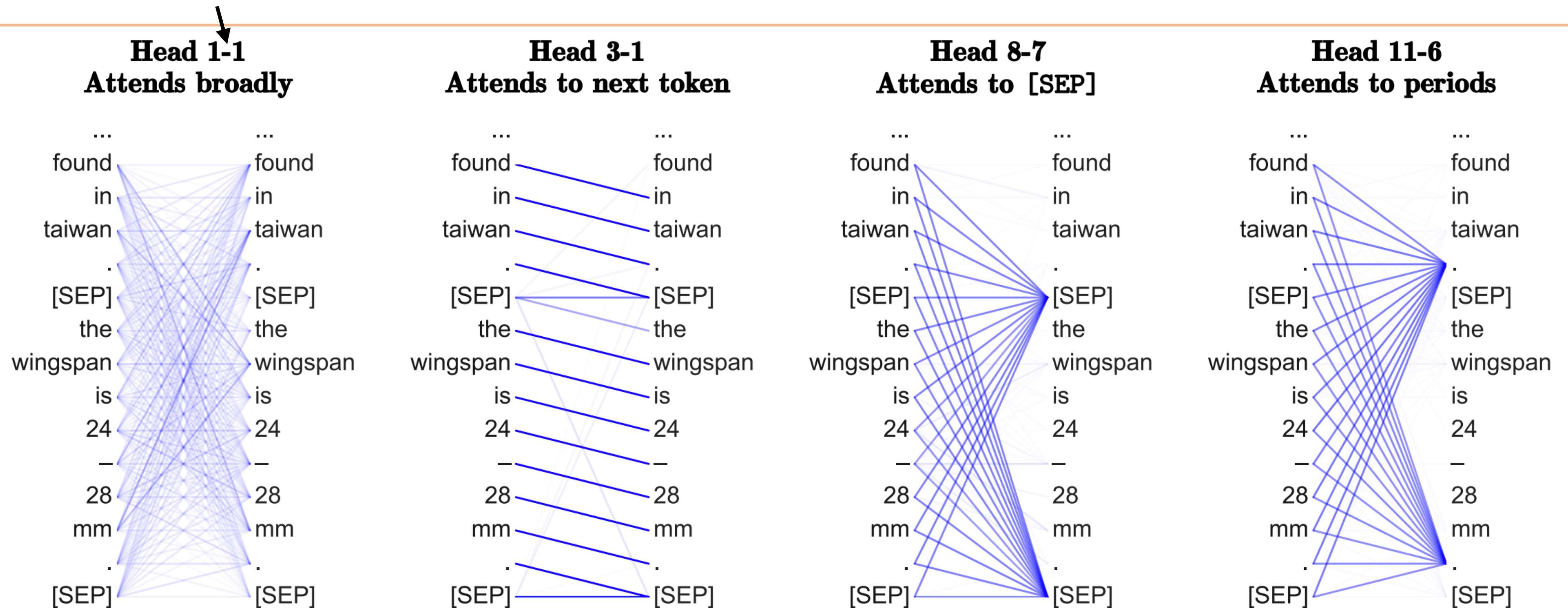
- ▶ How does BERT model this sentence pair stuff?
 - ▶ Transformers can capture interactions between the two sentences, (even though the NSP objective doesn't really cause this to happen).
- Devlin et al. (2019)

What can BERT NOT do?

- ▶ BERT **cannot** generate text (at least not in an obvious way)
 - ▶ Can fill in [MASK] tokens, but can't generate left-to-right (you can put [MASK] at the end, then predict repeatedly, but this is slow)
- ▶ Masked language models are intended to be used primarily for “analysis” tasks, e.g., sequential tagging, semantic similarity between two sentences, ...

What does BERT learn?

layer-head



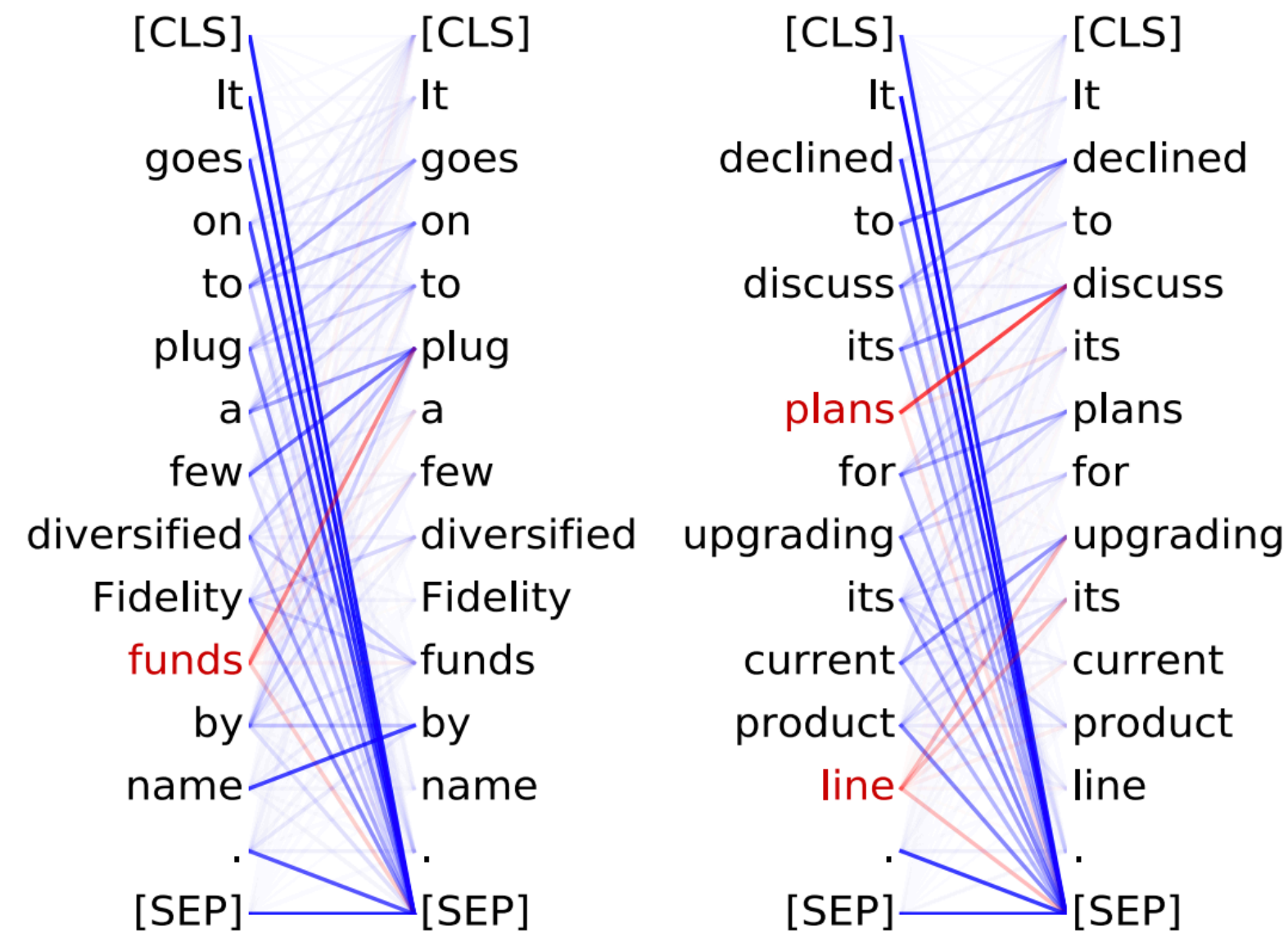
- ▶ Heads on transformers learn interesting and diverse things: content heads (attend based on content), positional heads (based on position), etc.

Clark et al. (2019)

What does BERT learn?

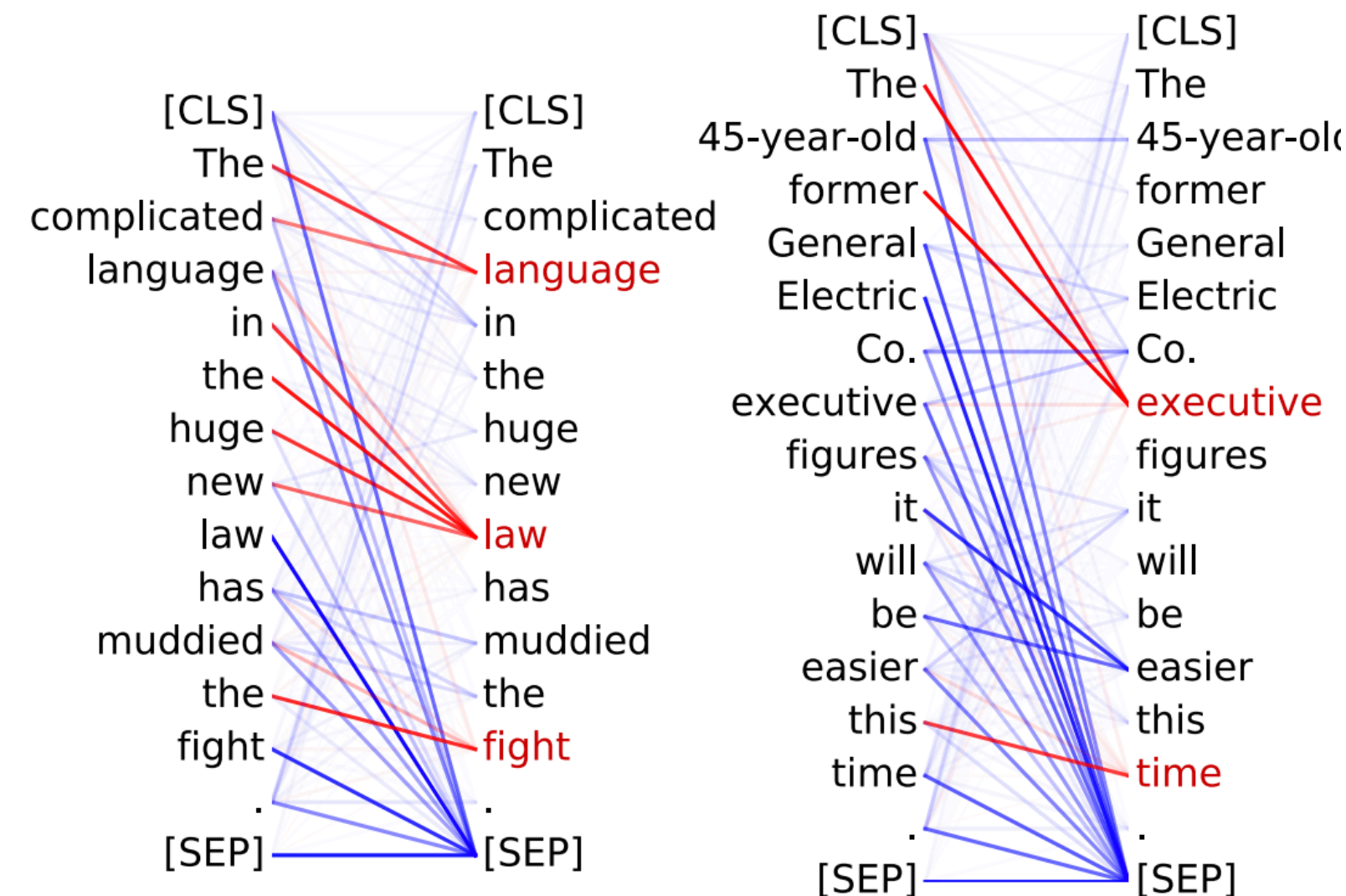
Head 8-10

- **Direct objects** attend to their verbs
- 86.8% accuracy at the dobj relation



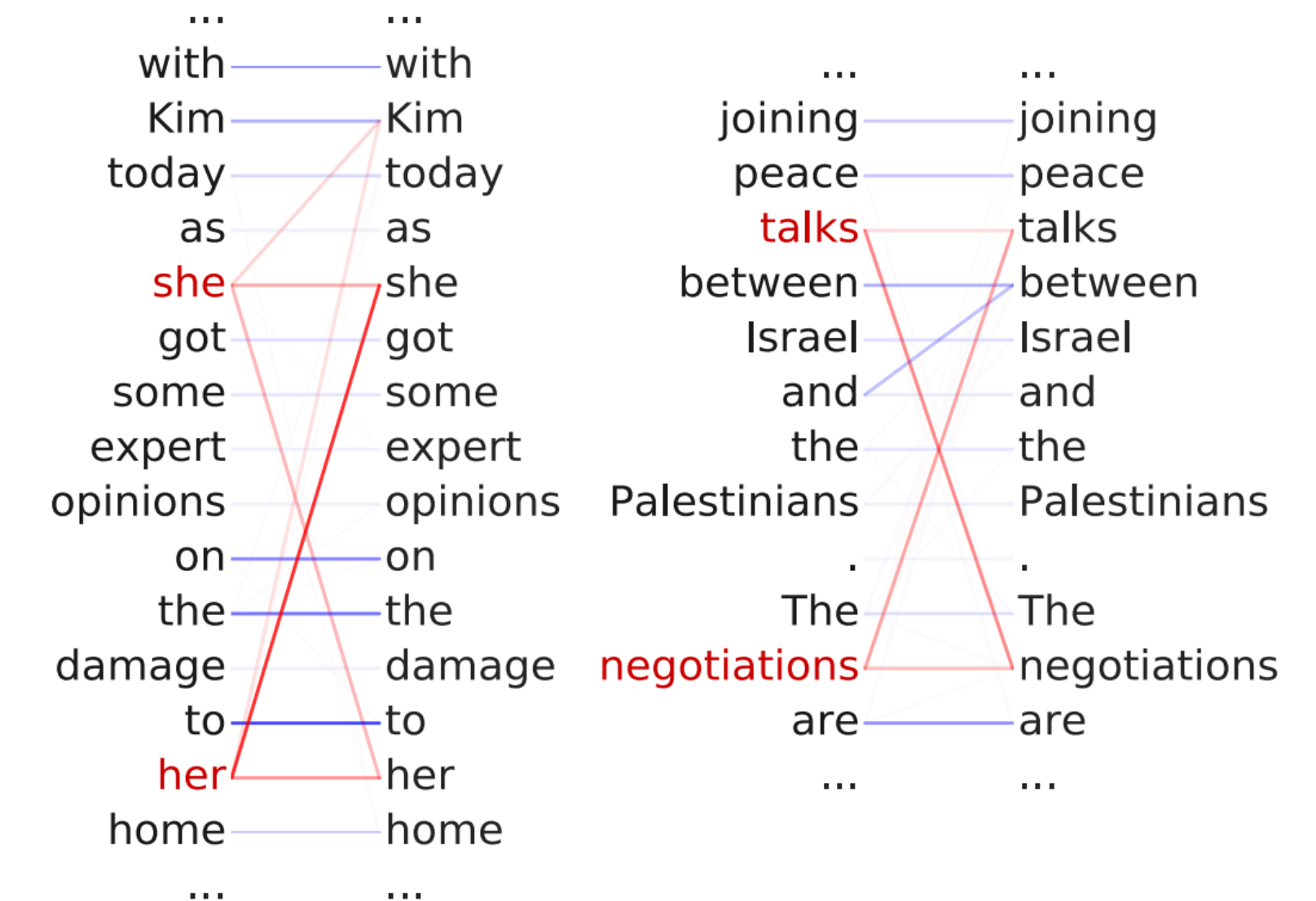
Head 8-11

- **Noun modifiers** (e.g., determiners) attend to their noun
- 94.3% accuracy at the det relation



Head 5-4

- **Coreferent** mentions attend to their antecedents
- 65.1% accuracy at linking the head of a coreferent mention to the head of an antecedent

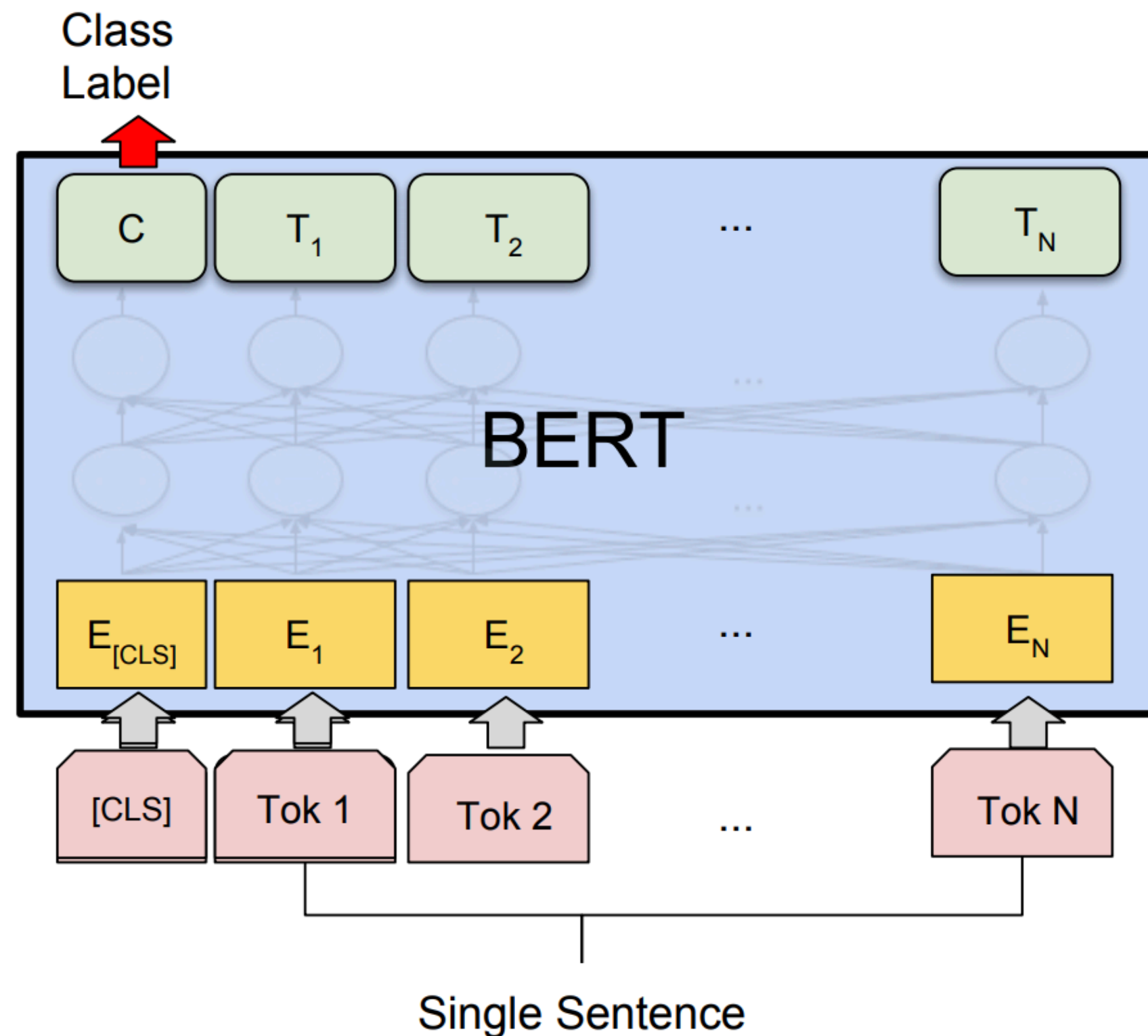


- ▶ Still way worse than what supervised systems can do, but interesting that this is learned organically

BERT Results, Extensions

Fine-tuning BERT

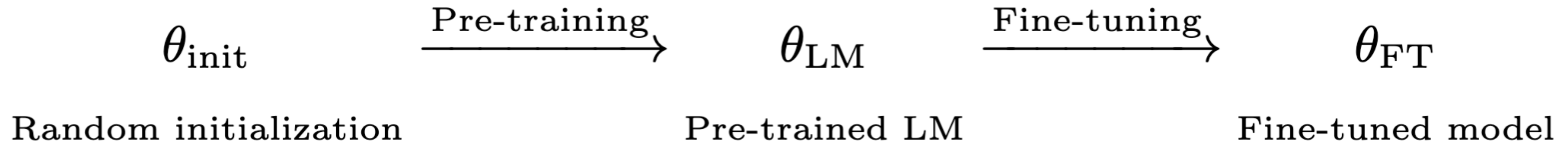
- ▶ Fine-tune for 1-3 epochs, small learning rate (e.g. $2e-5$ - $5e-5$)



(b) Single Sentence Classification Tasks:
SST-2, CoLA

- ▶ Large changes to weights up here (particularly in last layer to route the right information to [CLS])
- ▶ Smaller changes to weights lower down in the transformer
- ▶ Small LR and short fine-tuning schedule mean weights don't change much
- ▶ More complex “triangular learning rate” schemes exist

Pre-training, then Fine-tuning



- ▶ Masked Language Modeling (e.g., BERT):
$$\mathcal{L}_{\text{MLM}}(\theta) = - \sum_{t \in M} \log p_{\theta}(x_t \mid \tilde{\mathbf{x}})$$

corrupted input sequence (e.g., with [MASK] tokens)

- ▶ Fine-tuning: Learn how to perform a task from labeled examples.

$$\mathcal{L}_{\text{finetune}}(\theta) = - \sum_{(\mathbf{x}, \mathbf{y}) \in \mathcal{D}_{\text{task}}} \log p_{\theta}(\mathbf{y} \mid \mathbf{x})$$

initialize from θ_{LM}

Tasks: sentence classification, NER, etc.

Evaluation: GLUE

Corpus	Train	Test	Task	Metrics	Domain
Single-Sentence Tasks					
CoLA	8.5k	1k	acceptability	Matthews corr.	misc.
SST-2	67k	1.8k	sentiment	acc.	movie reviews
Similarity and Paraphrase Tasks					
MRPC	3.7k	1.7k	paraphrase	acc./F1	news
STS-B	7k	1.4k	sentence similarity	Pearson/Spearman corr.	misc.
QQP	364k	391k	paraphrase	acc./F1	social QA questions
Inference Tasks					
MNLI	393k	20k	NLI	matched acc./mismatched acc.	misc.
QNLI	105k	5.4k	QA/NLI	acc.	Wikipedia
RTE	2.5k	3k	NLI	acc.	news, Wikipedia
WNLI	634	146	coreference/NLI	acc.	fiction books

Wang et al. (2019)

Fine-tuning BERT

- How does frozen (❄️) vs. fine-tuned (🔥) compare?

Pretraining	Adaptation	NER	SA	Nat. lang. inference		Semantic textual similarity		
		CoNLL 2003	SST-2	MNLI	SICK-E	SICK-R	MRPC	STS-B
Skip-thoughts	❄️	-	81.8	62.9	-	86.6	75.8	71.8
ELMo	❄️	91.7	91.8	79.6	86.3	86.1	76.0	75.9
	🔥	91.9	91.2	76.4	83.3	83.3	74.7	75.5
	$\Delta = \text{🔥} - \text{❄️}$	0.2	-0.6	-3.2	-3.3	-2.8	-1.3	-0.4
BERT-base	❄️	92.2	93.0	84.6	84.8	86.4	78.1	82.9
	🔥	92.4	93.5	84.6	85.8	88.7	84.8	87.1
	$\Delta = \text{🔥} - \text{❄️}$	0.2	0.5	0.0	1.0	2.3	6.7	4.2

- BERT is typically better if the whole network is fine-tuned, unlike ELMo

Results

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	79.9	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	88.1	91.3	45.4	80.0	82.3	56.0	75.2
BERT _{BASE}	84.6/83.4	71.2	90.1	93.5	52.1	85.8	88.9	66.4	79.6
BERT _{LARGE}	86.7/85.9	72.1	91.1	94.9	60.5	86.5	89.3	70.1	81.9

- ▶ Huge improvements over prior work (even compared to ELMo)
- ▶ Effective at “sentence pair” tasks: textual entailment (does sentence A imply sentence B), paraphrase detection

Devlin et al. (2018)

Subsequent Improvements to BERT

- ▶ Dynamic masking: standard BERT uses the same MASK scheme for every epoch, RoBERTa recomputes them

epoch 2

epoch 1

... John visited Madagascar yesterday ...

- ▶ Whole word masking: don't mask out parts of words (word pieces)

... _John _visited _Mada gas car yesterday ...

Liu et al. (2019)

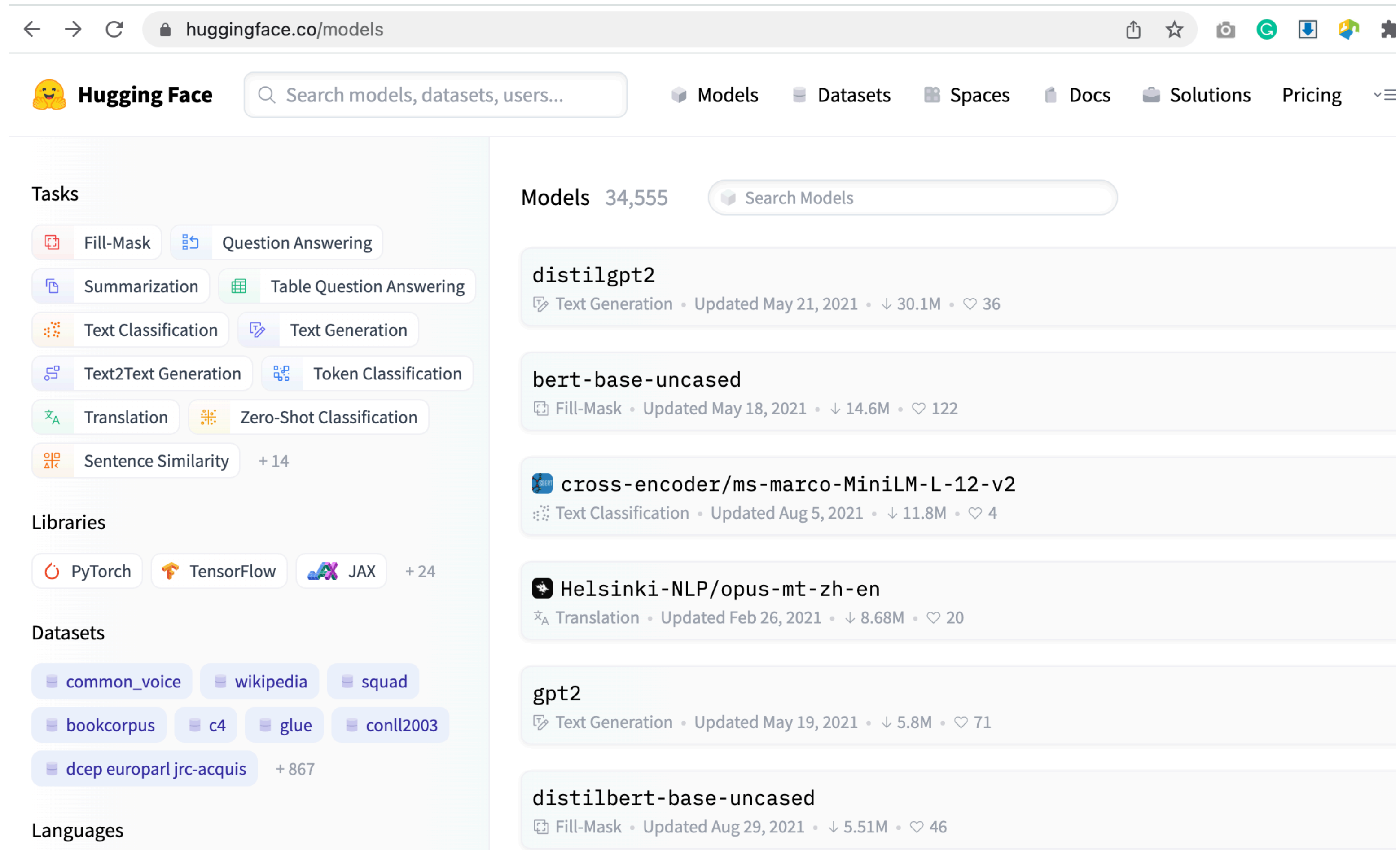
RoBERTa

- ▶ “Robustly optimized BERT” incorporating some of these tricks
- ▶ 160GB of data instead of 16 GB
- ▶ New training + more data = better 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
BERT _{LARGE}						
with BOOKS + WIKI	13GB	256	1M	90.9/81.8	86.6	93.7

many BERT variations

- For specific text domains (e.g. StackOverflow), or specific languages



The screenshot shows the Hugging Face website's models page. The browser address bar displays 'huggingface.co/models'. The page features a navigation bar with the Hugging Face logo, a search bar, and links to Models, Datasets, Spaces, Docs, Solutions, and Pricing. On the left sidebar, there are sections for Tasks (Fill-Mask, Question Answering, Summarization, Table Question Answering, Text Classification, Text Generation, Text2Text Generation, Token Classification, Translation, Zero-Shot Classification, Sentence Similarity), Libraries (PyTorch, TensorFlow, JAX), Datasets (common_voice, wikipedia, squad, bookcorpus, c4, glue, conll2003, dcep europarl jrc-acquis), and Languages. The main content area shows a list of models with a search bar and a count of 34,555 models. The listed models include distilgpt2, bert-base-uncased, cross-encoder/ms-marco-MiniLM-L-12-v2, Helsinki-NLP/opus-mt-zh-en, gpt2, and distilbert-base-uncased, each with details on its task, update date, size, and popularity.

← → ↻ huggingface.co/models

 **Hugging Face**

Models Datasets Spaces Docs Solutions Pricing

Tasks

Fill-Mask Question Answering

Summarization Table Question Answering

Text Classification Text Generation

Text2Text Generation Token Classification

Translation Zero-Shot Classification

Sentence Similarity + 14

Libraries

PyTorch TensorFlow JAX + 24

Datasets

common_voice wikipedia squad

bookcorpus c4 glue conll2003

dcep europarl jrc-acquis + 867

Languages

Models 34,555

distilgpt2
Text Generation • Updated May 21, 2021 • ↓ 30.1M • ♥ 36

bert-base-uncased
Fill-Mask • Updated May 18, 2021 • ↓ 14.6M • ♥ 122

cross-encoder/ms-marco-MiniLM-L-12-v2
Text Classification • Updated Aug 5, 2021 • ↓ 11.8M • ♥ 4

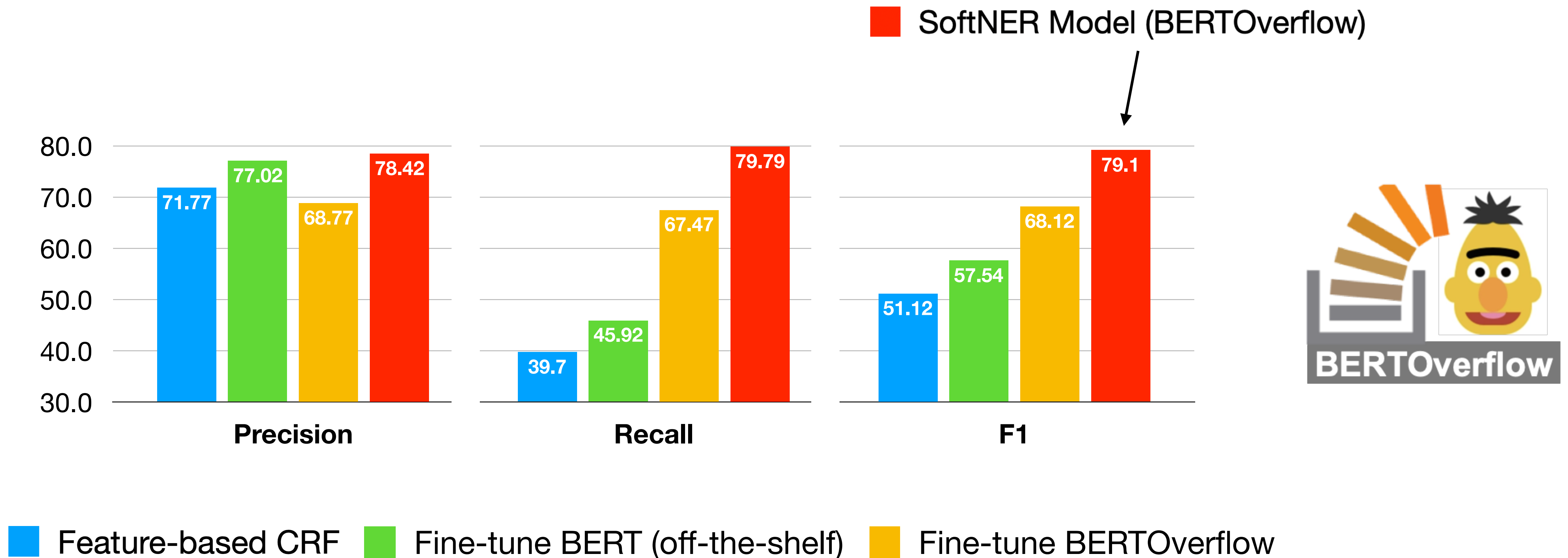
Helsinki-NLP/opus-mt-zh-en
Translation • Updated Feb 26, 2021 • ↓ 8.68M • ♥ 20

gpt2
Text Generation • Updated May 19, 2021 • ↓ 5.8M • ♥ 71

distilbert-base-uncased
Fill-Mask • Updated Aug 29, 2021 • ↓ 5.51M • ♥ 46

BERTOverflow

- ▶ A domain-specific BERT model that pre-trained on 10-year StackOverflow data (152M sentences; ~2B tokens).



NER in StackOverflow

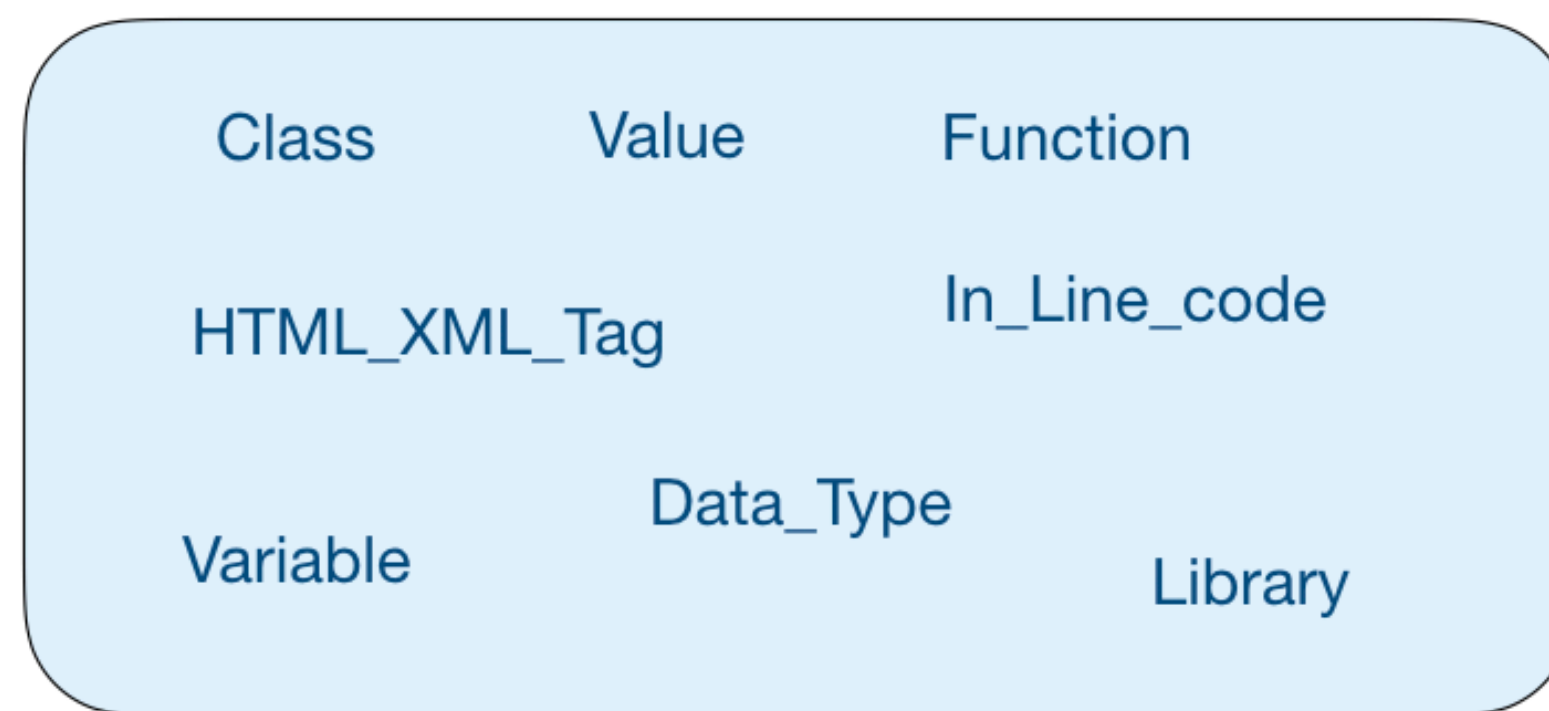
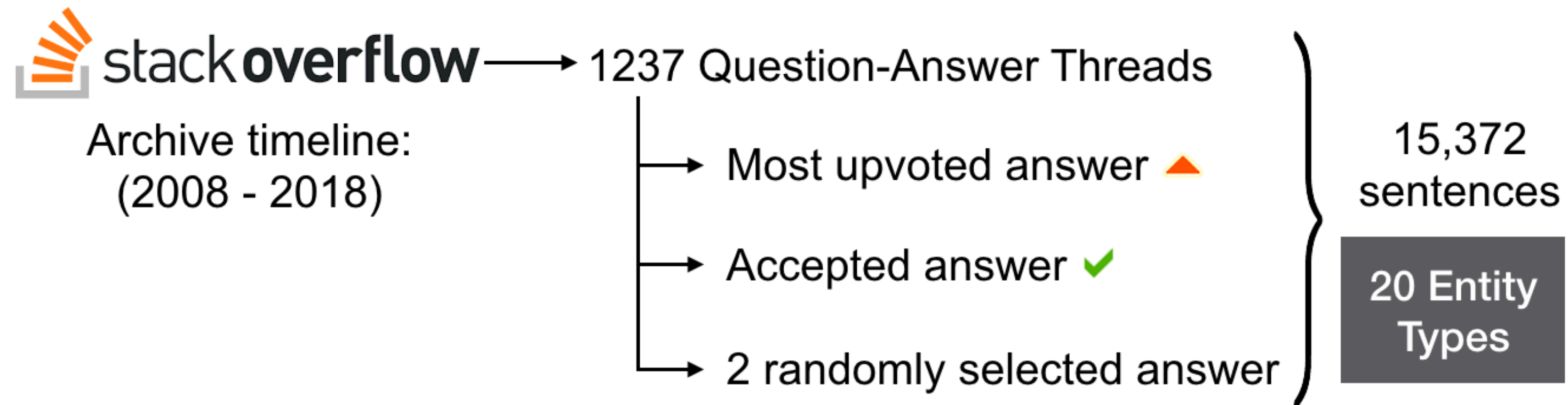
I am passing an Library_Class array list as message header to Library_Class camel route
through Language java Library_Class bean as follows

```
ArrayList<String> list=new ArrayList<String>();  
    list.add("http://www.google.com");  
    list.add("http://www.stackoverflow.com");  
    list.add("http://www.tutorialspoint.com");  
    list.add("http://localhost:8080/sampleExample/query");  
    exchange.getOut().setHeader("endpoints",list);
```

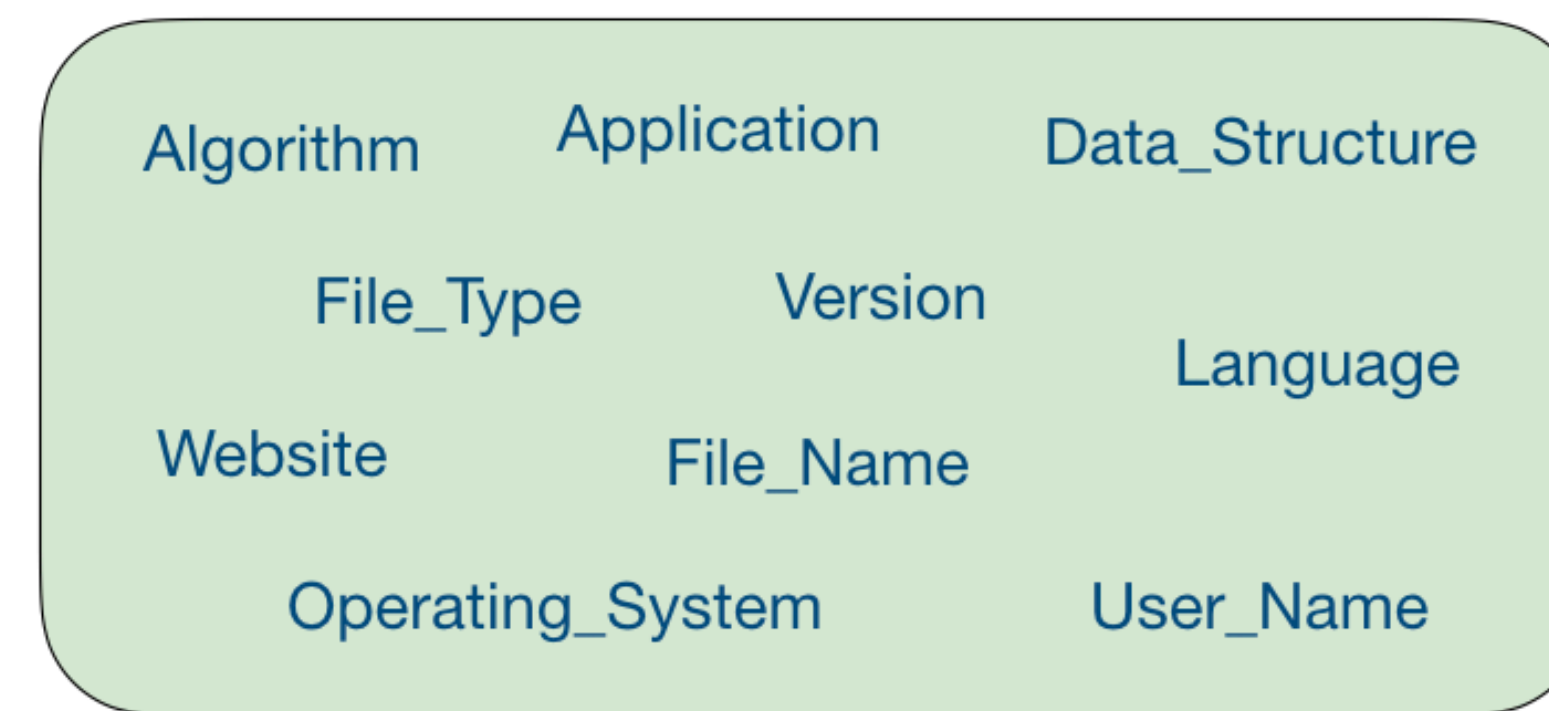
and, inside Library_Class camel route i want to iterate through this Variable_Name list

NER in StackOverflow

StackOverflow NER Corpus



Code Entity Types



Natural Language Entity

Multilingual BERT

Bilingual BERT

- ▶ A customized bilingual BERT for Arabic NLP and English-to-Arabic zero-shot transfer learning

	Data Source	Data Size (All / English / Arabic)	IE Performance (F1 score)
AraBERT (AUBeirut 2019)	News	2.5B / 0B / 2.5B	97.1 / —
mBERT (Google 2018)	Wiki	21.9B / 2.5B / 0.15B	75.3 / 30.1
XLM-RoBERTa (Facebook 2019)	Common Crawl	295B / 55.6B / 2.9B	79.2 / 40.4
GigaBERT (our work)	News, Wiki, Common Crawl	10.4B / 6.1B / 4.3B	84.3 / 48.2

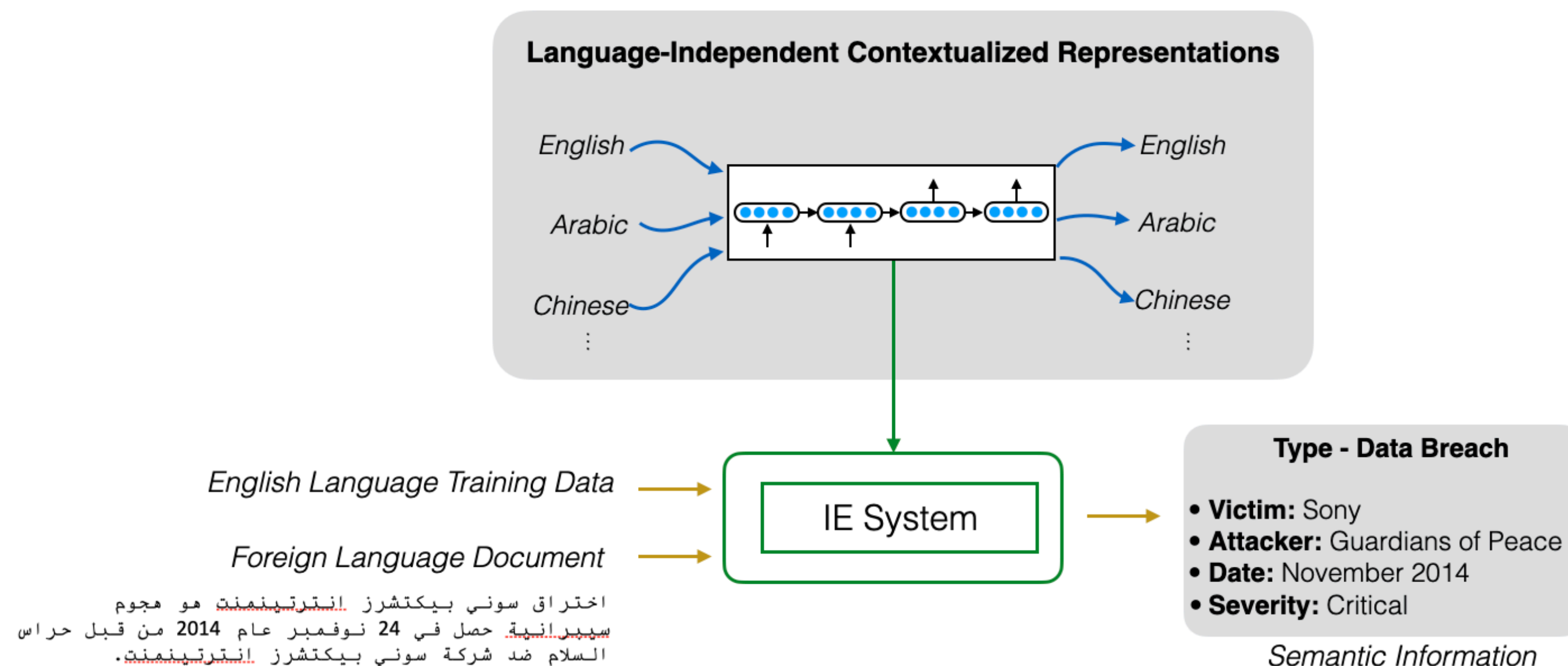
supervised learning

zero-shot transfer learning

Lan et al. (2020)

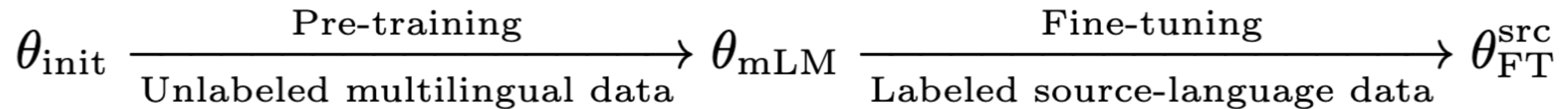
Information Extraction

- ▶ A customized bilingual BERT for Arabic NLP and English-to-Arabic zero-shot transfer learning
- ▶ i.e., Information extraction models, trained on annotated English data, directly apply to non-English texts to extract entities and events.



Cross-lingual Transfer Learning

- ▶ Pre-train LMs on unlabeled multilingual data, then fine-tune for specific tasks using labeled data in some languages (e.g., English)

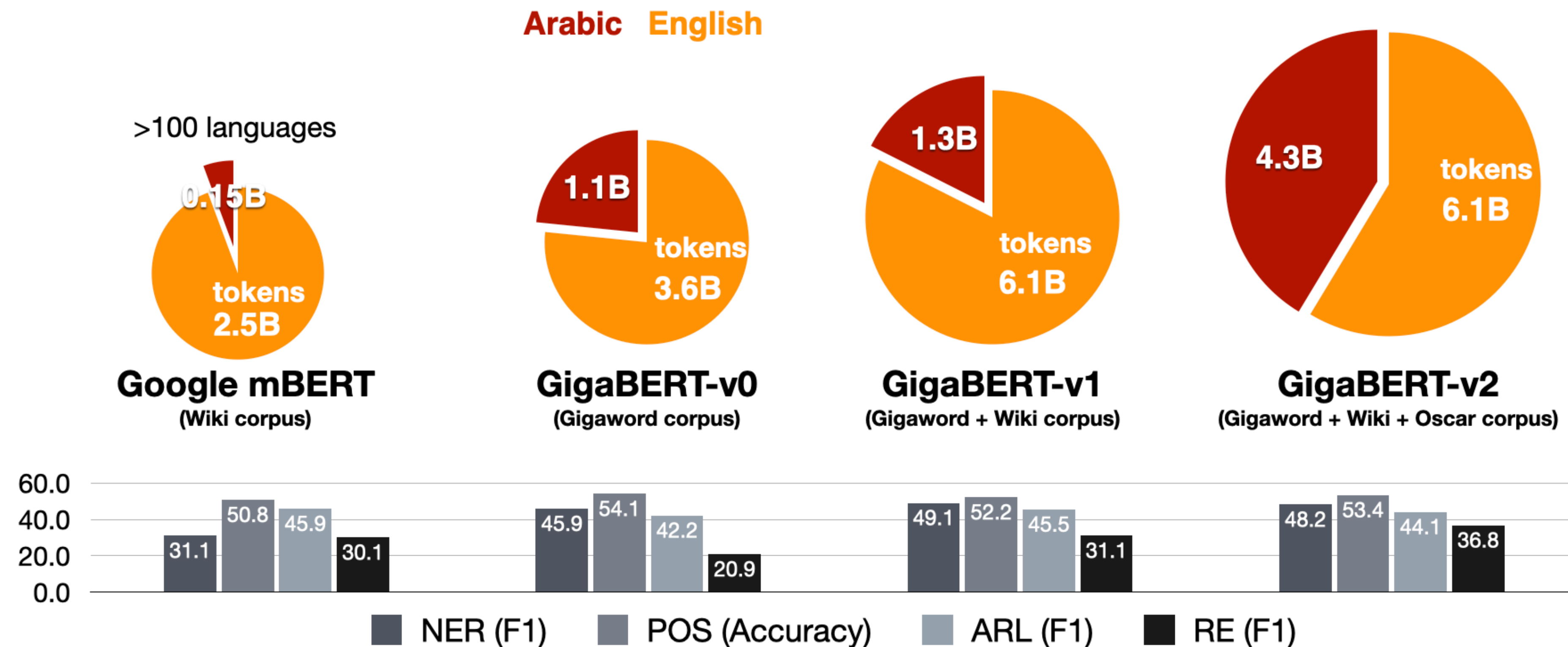


- ▶ Apply the fine-tuned model directly to target languages (e.g., Arabic), without labeled data in target languages or further training:

$$\hat{\mathbf{y}}_{\text{tgt}} = \arg \max_{\mathbf{y}} p_{\theta_{\text{FT}}^{\text{src}}}(\mathbf{y} \mid \mathbf{x}_{\text{tgt}})$$

Bilingual BERT

- ▶ A customized bilingual BERT for Arabic NLP and English-to-Arabic zero-shot transfer learning

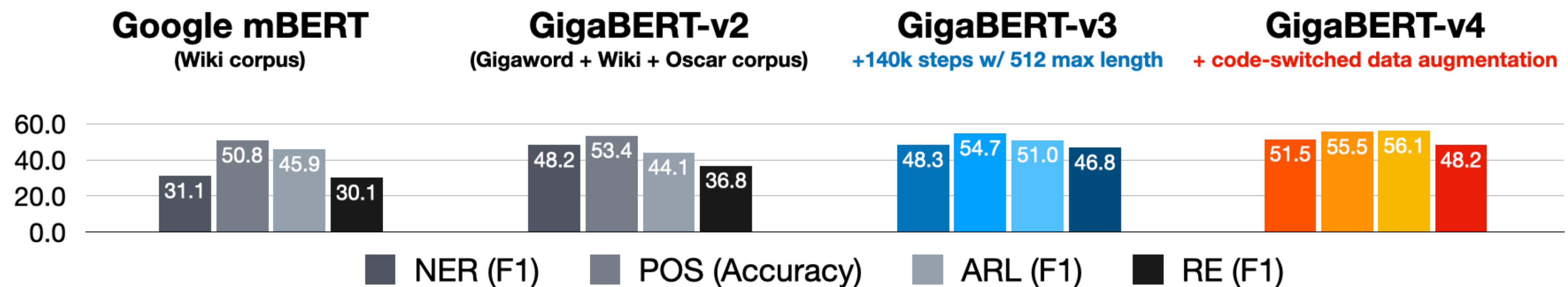


Bilingual BERT

- ▶ A customized bilingual BERT for Arabic NLP and English-to-Arabic zero-shot transfer learning + synthetic code-switching data

(English Translation — Official: China reaches the five-year plan target of cutting emissions)
مسؤول: الصين تبلغ هدف الخطة الخمسية المتعلق بخفض الانبعاثات

مسؤول: China تبلغ هدف plan الخمسية المتعلق بخفض emissions



mmBERT

- Multi-stage training with different number of languages per stage

Category	Dataset	Pre-training		Mid-training		Decay Phase	
		Tokens (B)	%	Tokens (B)	%	Tokens (B)	%
Code	Code (ProLong)	–	–	–	–	2.8	2.7
Code	Starcoder	100.6	5.1	17.2	2.9	0.5	0.5
Crawl	DCLM	600.0	30.2	10.0	1.7	–	–
Crawl	DCLM (Dolmino)	–	–	40.0	6.7	2.0	2.0
Crawl	FineWeb2	1196.6	60.2	506.7	84.3	78.5	76.0
Instruction	Tulu Flan	15.3	0.8	3.1	0.5	1.0	1.0
Math	Dolmino Math	11.2	0.6	4.3	0.7	0.5	0.5
Reference	Books	4.3	0.2	3.9	0.7	2.2	2.1
Reference	Textbooks (ProLong)	–	–	–	–	3.1	3.0
Reference	Wikipedia (MegaWika)	4.7	0.2	1.2	0.2	9.5	9.2
Scientific	Arxiv	27.8	1.4	5.4	0.9	3.3	3.2
Scientific	PeS2o	8.4	0.4	3.2	0.5	–	–
Social	StackExchange	18.6	0.9	3.0	0.5	–	–
Social	StackExchange (Dolmino)	1.4	0.1	2.8	0.5	–	–
Total		1989.0	100.0	600.8	100.0	103.3	100.0

mmBERT

- Pre-trained on 3T tokens of multilingual text in over 1800 languages

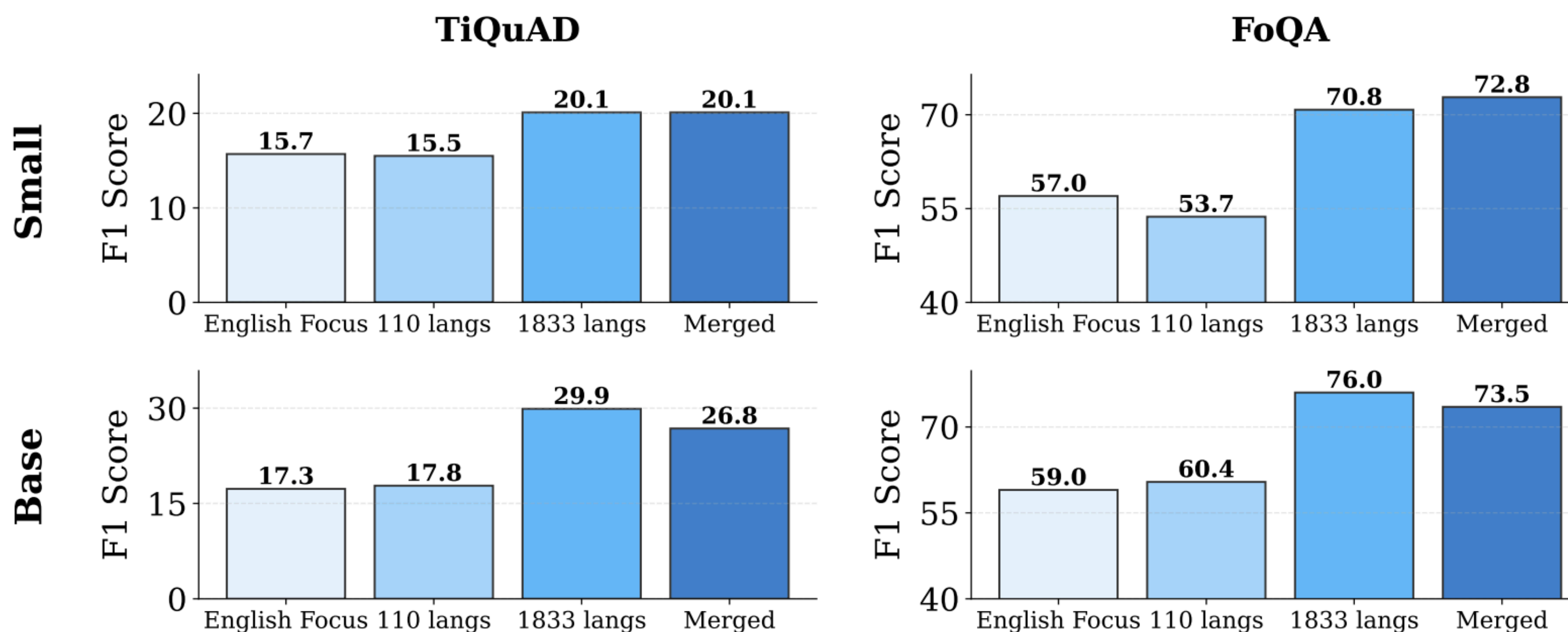


Figure 2: Performance of models using different decay phases on two languages (Tigray and Faroese) only added during the decay phase. We see that MMBERT with the 1833 language decay phase shows rapid performance improvements despite only having the models in the last 100B tokens of training. The final MMBERT models shows improvements by merging together checkpoints.

Other Encoder-only LMs

- ▶ XLM-RoBERTa (Conneau et al., 2020) for 100 languages
- ▶ mDeBERTa (He et al., 2021) for 100 languages
- ▶ MosaicBERT (Portes et al., 2023; Nussbaum et al., 2024)
- ▶ ModernBERT (Warner et al., 2024)
- ▶ EuroBERT (Boizard et al., 2025) for 15 languages
- ▶ NeoBERT (Le Breton et al., 2025)
- ▶ mmBERT (Marone et al., 2025) for 1800 languages
- ▶ ? Jev (TypeSafe AI, 2026)