CS11-711 Advanced NLP

Pre-training for Fine-tuning

Daniel Fried

Carnegie Mellon University
Language Technologies Institute

Site
https://cmu-anlp.github.io/

(w/ slides by Greg Durrett, Antonis Anastasopoulos, Graham Neubig, and Lucio Dery)

Multi-task Learning

Overview

Terminology

- **Multi-task learning** is a general term for training on multiple tasks

- **Transfer learning** is a type of multi-task learning where we only really care about one of the tasks

- **Pre-training** is a type of transfer learning where one objective is used first

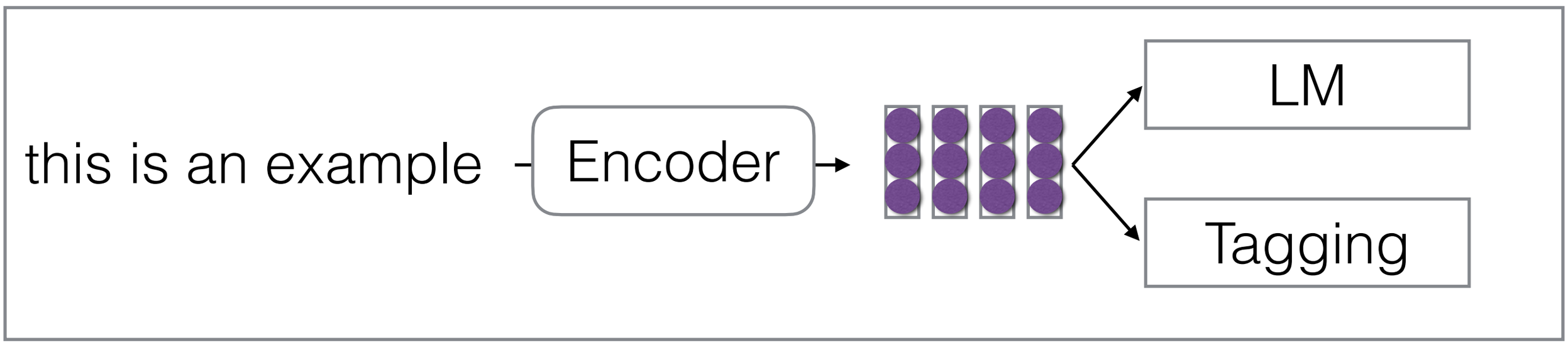
- **Few-shot, zero-shot learning** indicates learning to perform a task with very few, or zero labeled examples for that task

Plethora of Tasks in NLP

- In NLP, there are a plethora of tasks, each requiring different varieties of data
  - **Only text:** e.g. language modeling
  - **Naturally occurring inputs and outputs:** e.g. machine translation
  - **Hand-labeled outputs:** e.g. most analysis tasks
- And each in many languages, many domains!

Standard Multi-task Learning

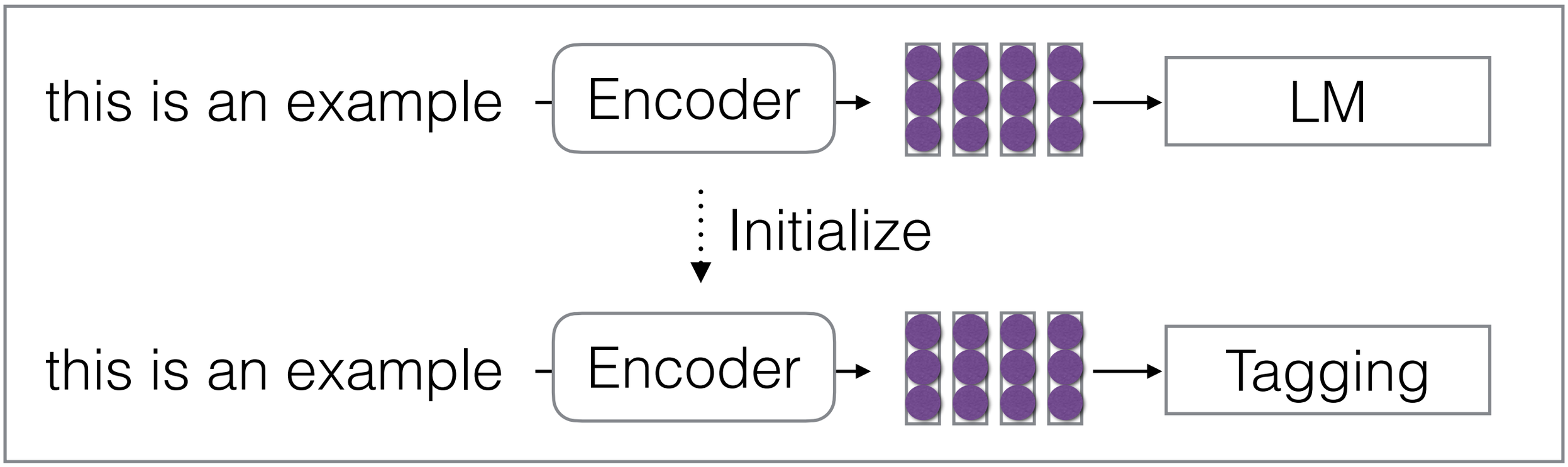
- Train representations to do well on multiple tasks at once



- Often as simple as randomly choosing minibatch from one of multiple tasks

Pre-train and Fine-Tune

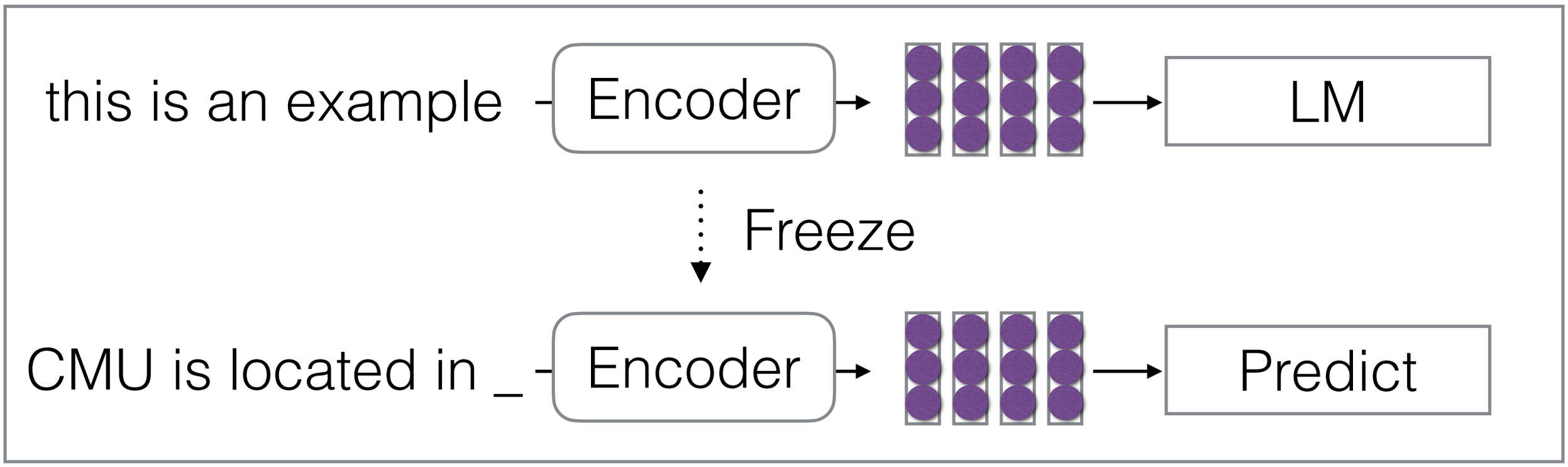
- First train on one task, then train on another



This is our main focus today.

Prompting

- Train on language generation task, make predictions in textualized tasks



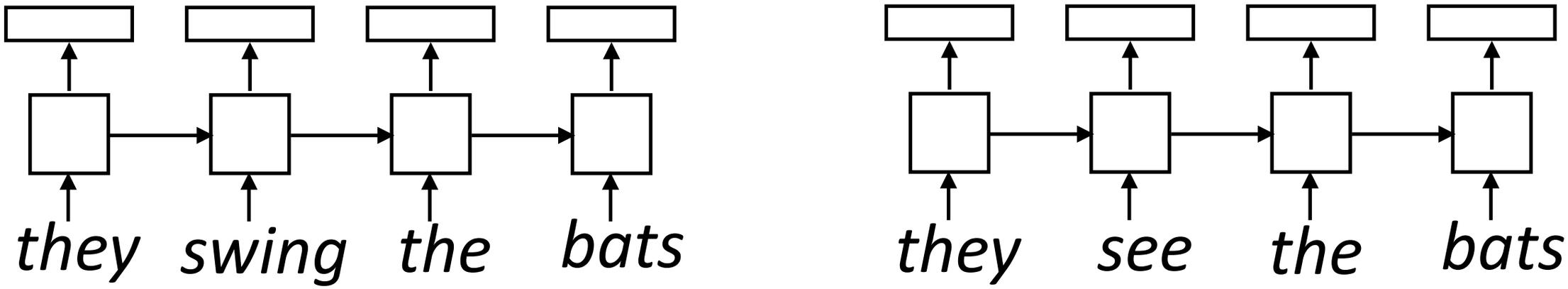
We'll mostly cover this next week.

Thinking about Pre-trained Models

- Many pre-trained models have names like BERT, RoBERTa, GPT-3, PaLM
- These often refer to a combination of
  - **Architecture:** Usually Transformer-based, but details vary & can be underspecified
  - **Data:** Often webpages, Wikipedia, books…
  - **Training objective:** Typically one of
    - **Masked Language Modeling** (e.g. BERT)
    - **Seq-to-seq De-noising** (e.g. BART, T5)
    - **Autoregressive Language Modeling** (e.g. GPT)

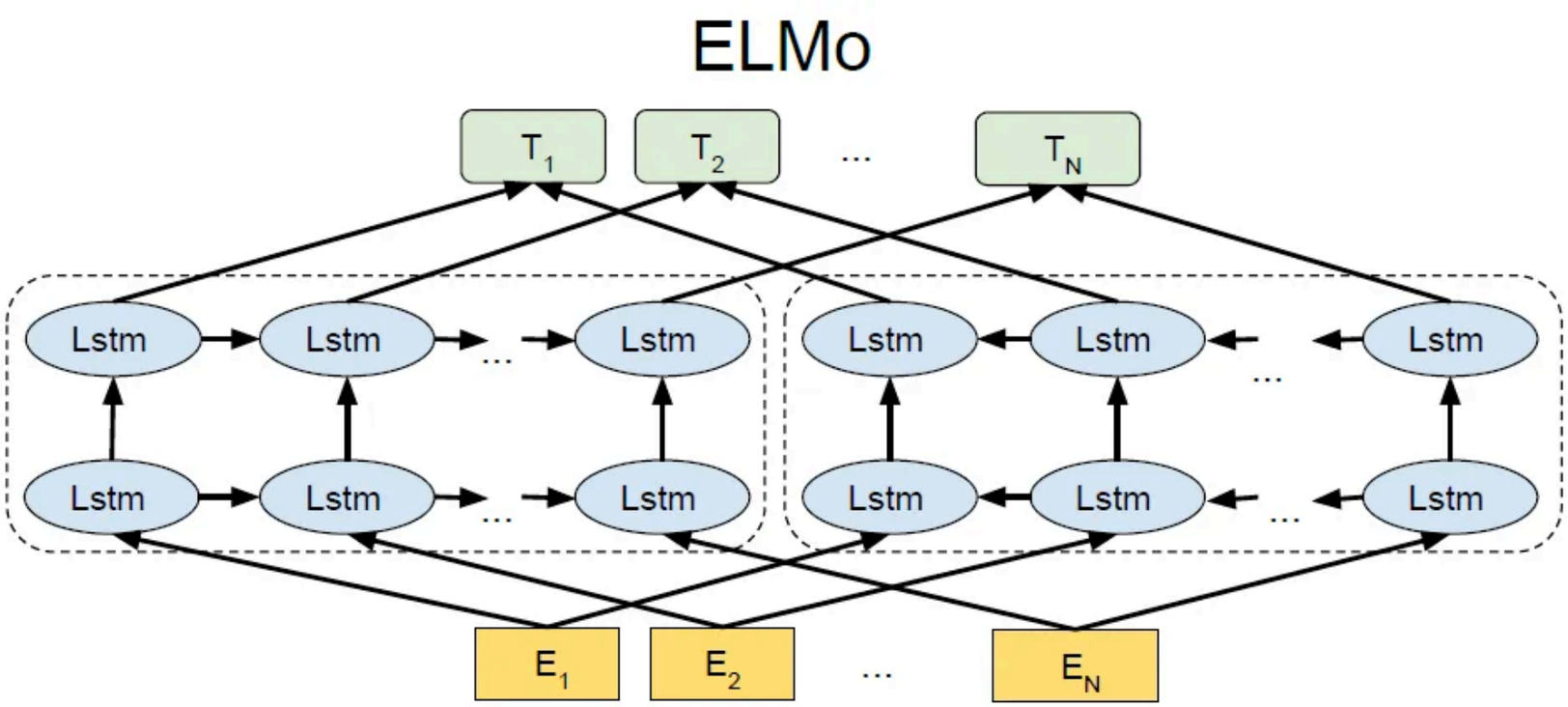
(Masked) Language Modeling

Context-dependent Embeddings



- Train a neural language model to predict the next word given previous words in the sentence, use the hidden states (output) at each step *as word embeddings*

ELMo: Embeddings from Language Models



- Once ELMo is pre-trained, keep it “frozen” and use the representations (“embeddings”, **T**) in down-stream tasks.
- Huge gains across many high-profile tasks: NER, question answering, semantic role labeling, etc.

Image from Devlin et al. 2018

Peters et al. (2018)

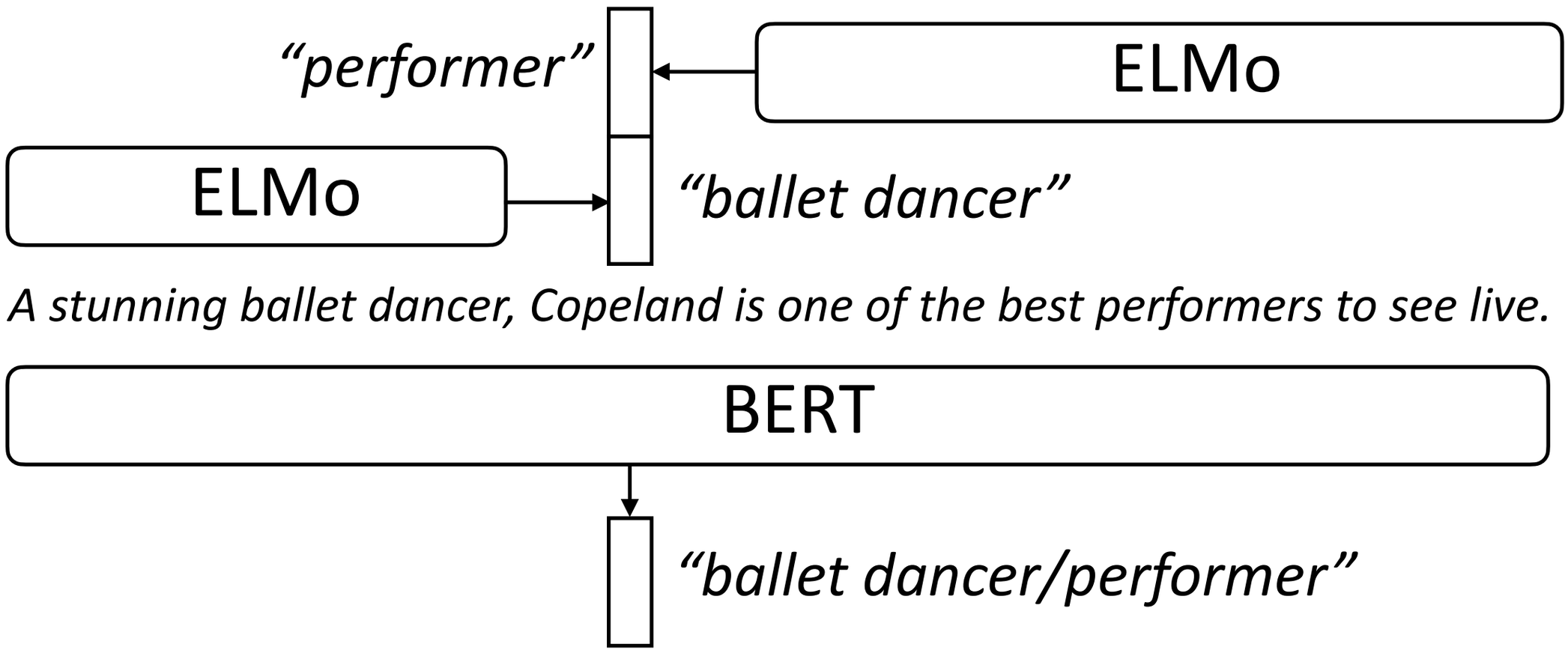
BERT: Bidirectional Embedding Representations from Transformers

- AI2 made ELMo in spring 2018, GPT (transformer-based ELMo) was released in summer 2018, BERT came out October 2018

- Four major changes compared to ELMo:
  - Transformers instead of LSTMs
  - Bidirectional model with “Masked LM” objective instead of standard LM
  - Fine-tune the model when transferring to tasks, instead of freezing
  - Operates over word pieces (byte pair encoding)

BERT

- ELMo is a unidirectional model (as is GPT): we can concatenate two unidirectional models, but is this the right thing to do?
- ELMo reprs look at each direction in isolation; BERT looks at them jointly

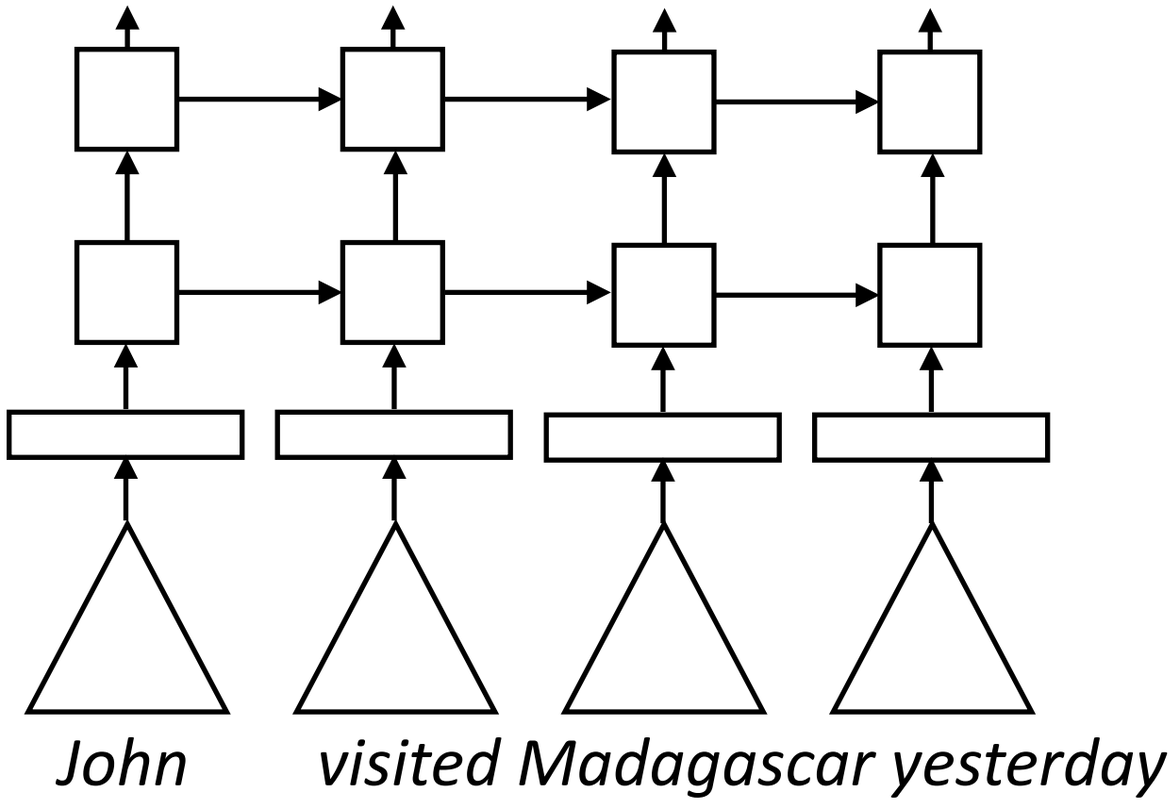


Devlin et al. (2019)

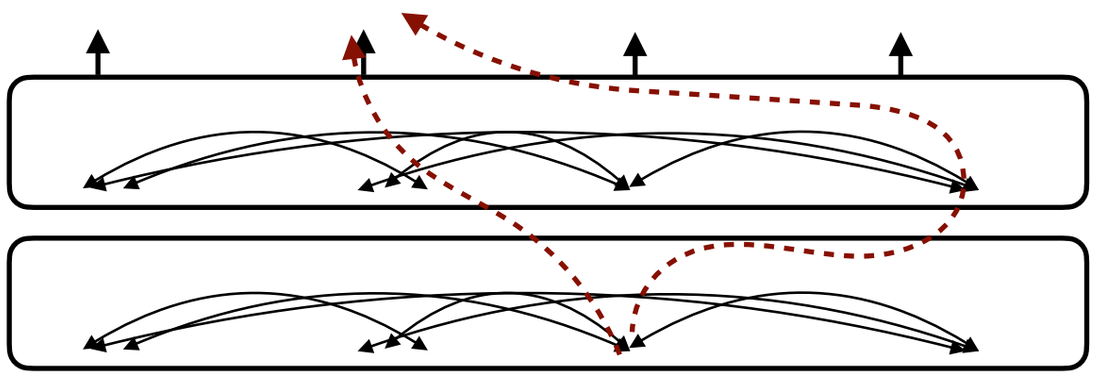
BERT

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

ELMo (Language Modeling)



BERT

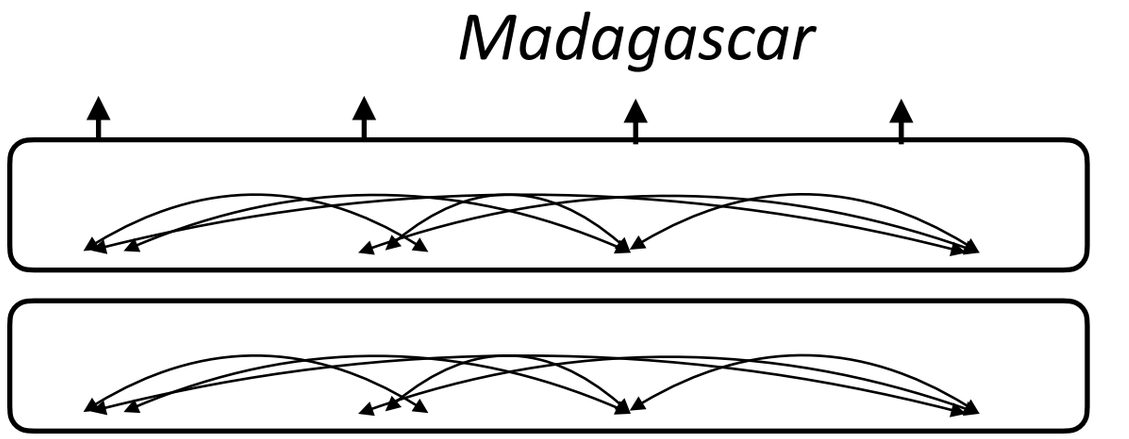


- You could do this with a “one-sided” transformer, but this “two-sided” model can cheat

Devlin et al. (2019)

Masked Language Modeling

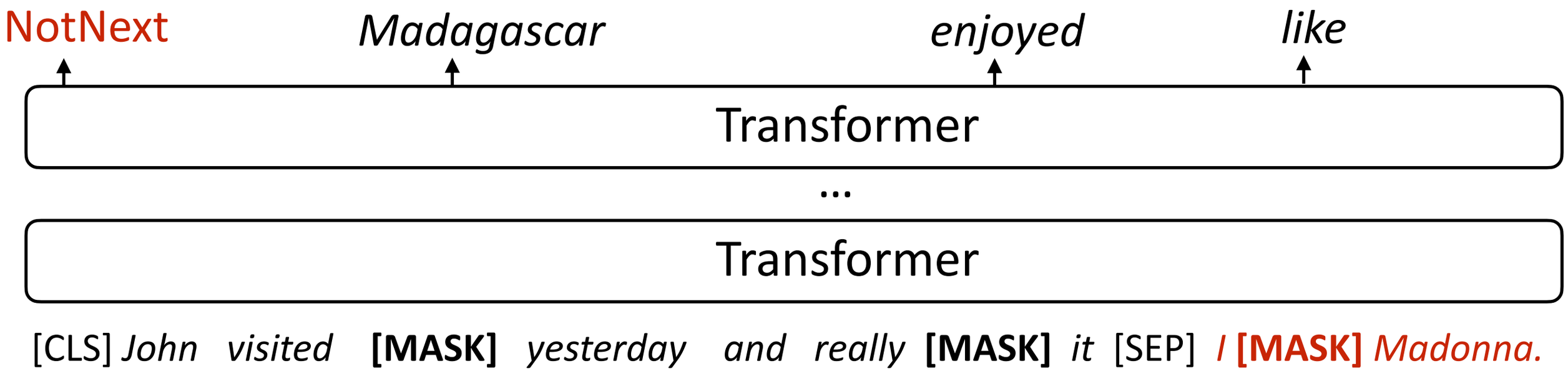
- How to prevent cheating? Next word prediction fundamentally doesn't work for bidirectional models, instead do *masked language modeling*
- BERT formula: take a chunk of text, mask out 15% of the tokens, and try to predict them
- Optimize P(*Madagascar* | *John visited [MASK] yesterday*)



Devlin et al. (2019)

Next “Sentence” Prediction

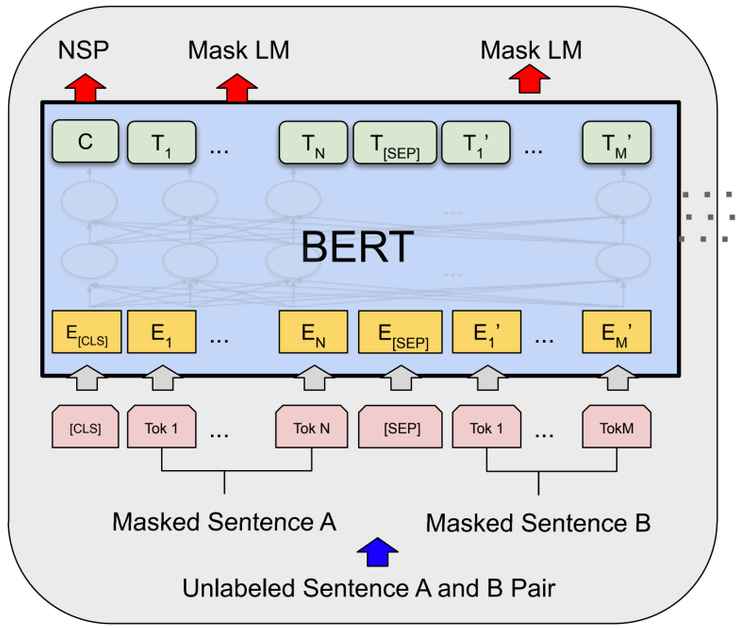
- Input: [CLS] Text chunk 1 [SEP] Text chunk 2
- 50% of the time, take the true next chunk of text, 50% of the time take a random other chunk. Predict whether the next chunk is the “true” next
- **Why might this be a good idea?**
- BERT objective: masked LM + next sentence prediction

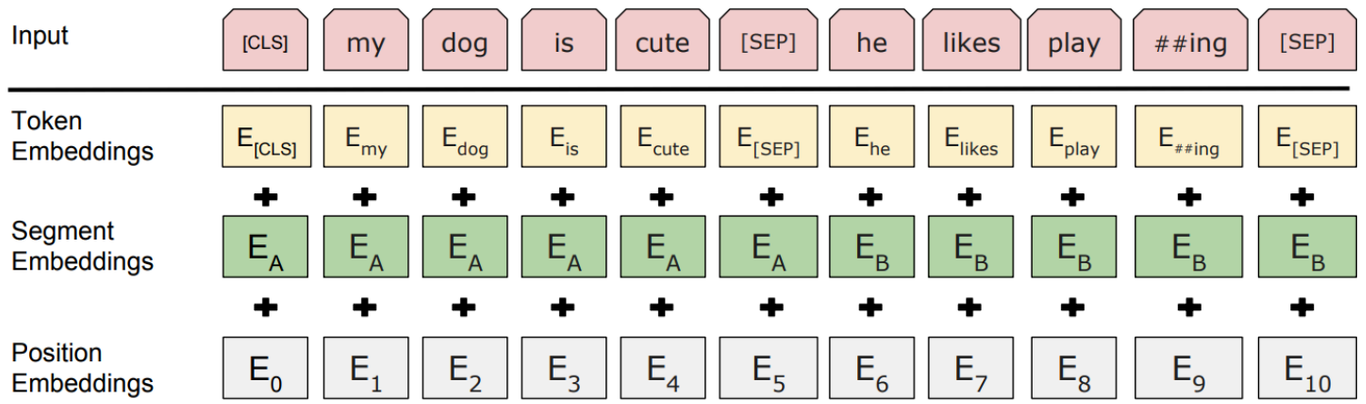


Devlin et al. (2019)

BERT Architecture

- BERT Base: 12 layers, 768-dim per wordpiece token, 12 heads. Total params = 110M
- BERT Large: 24 layers, 1024-dim per wordpiece token, 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

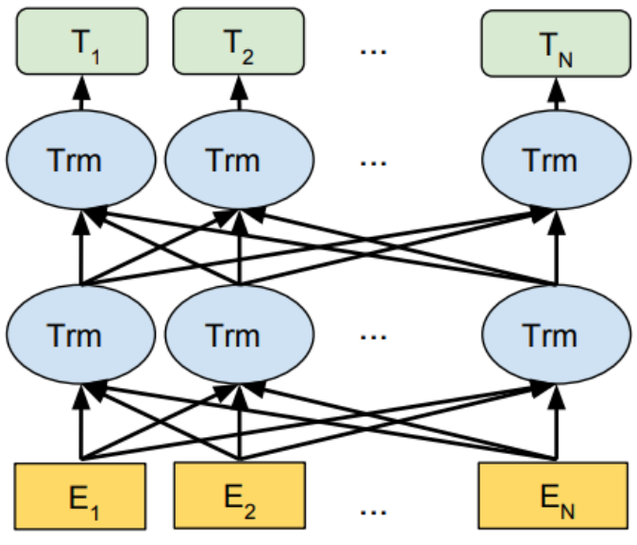


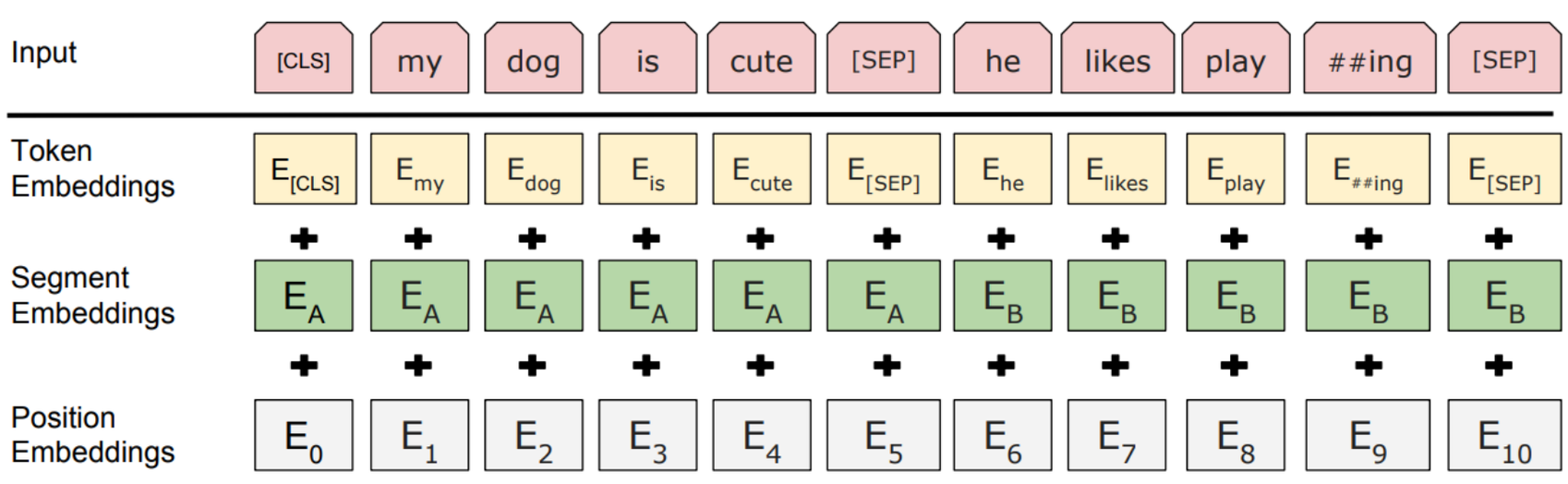


Devlin et al. (2019)

BERT: All Together

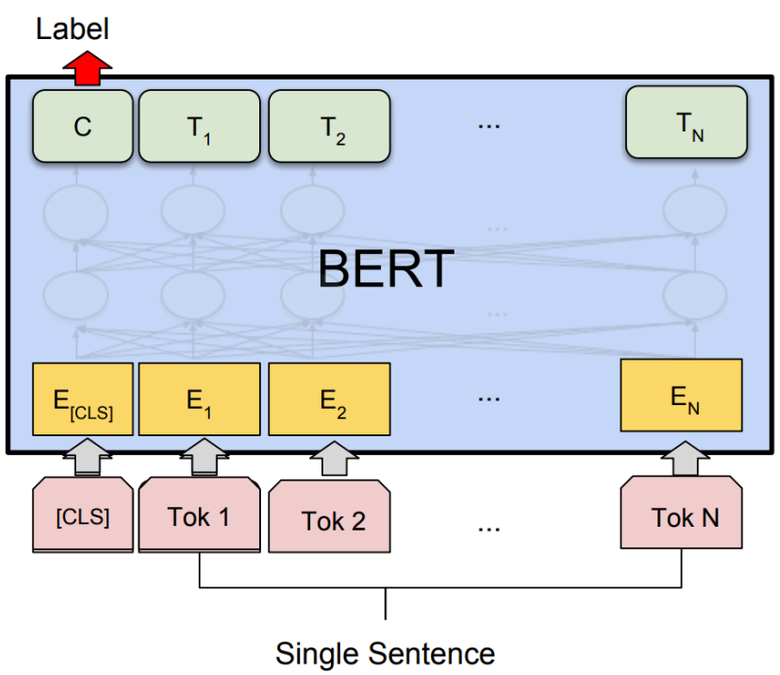
- **Model:** Multi-layer self-attention. Input sentence or pair, w/ [CLS] token, subword representation. Up to 340M parameters



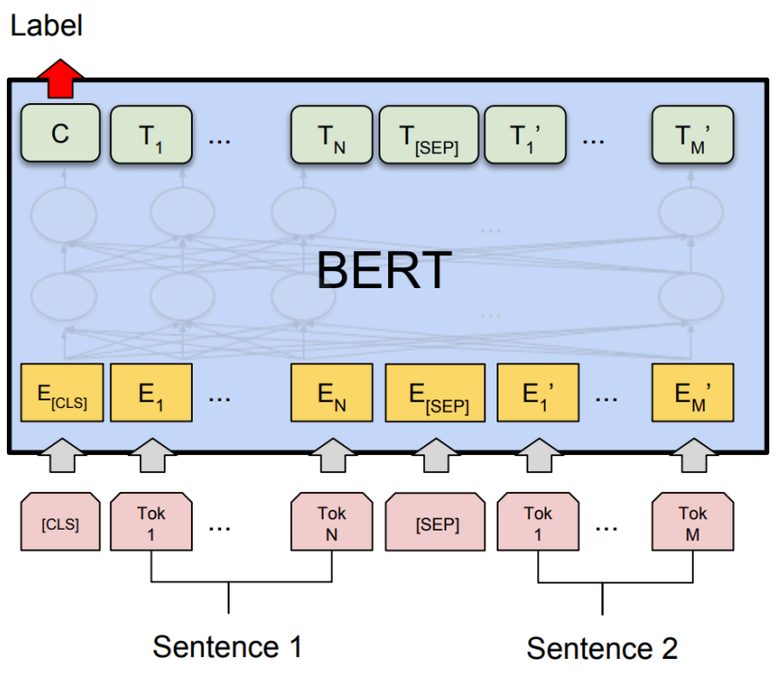


- **Objective:** Masked word prediction + next-sentence prediction
- **Data:** BooksCorpus + English Wikipedia (16GB)

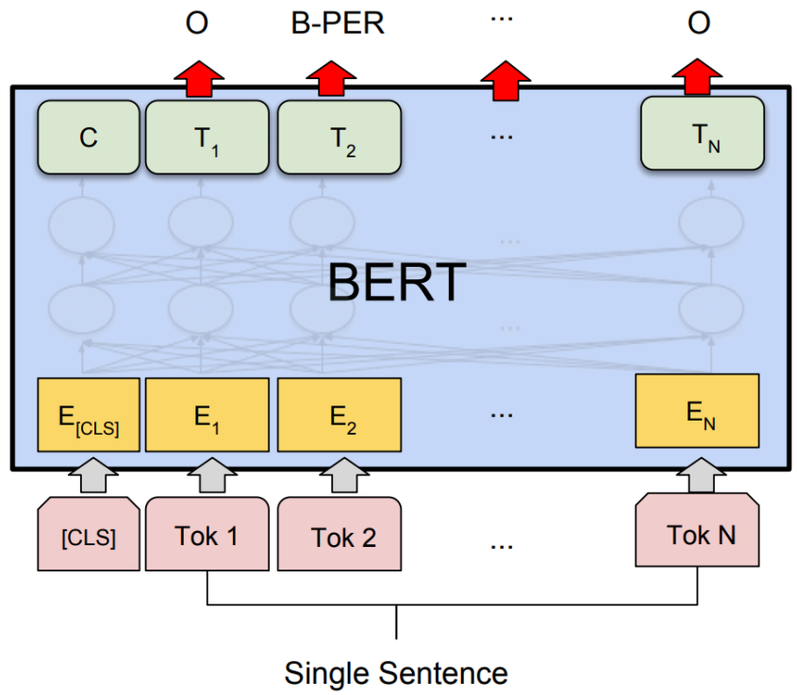
What can BERT do?



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



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



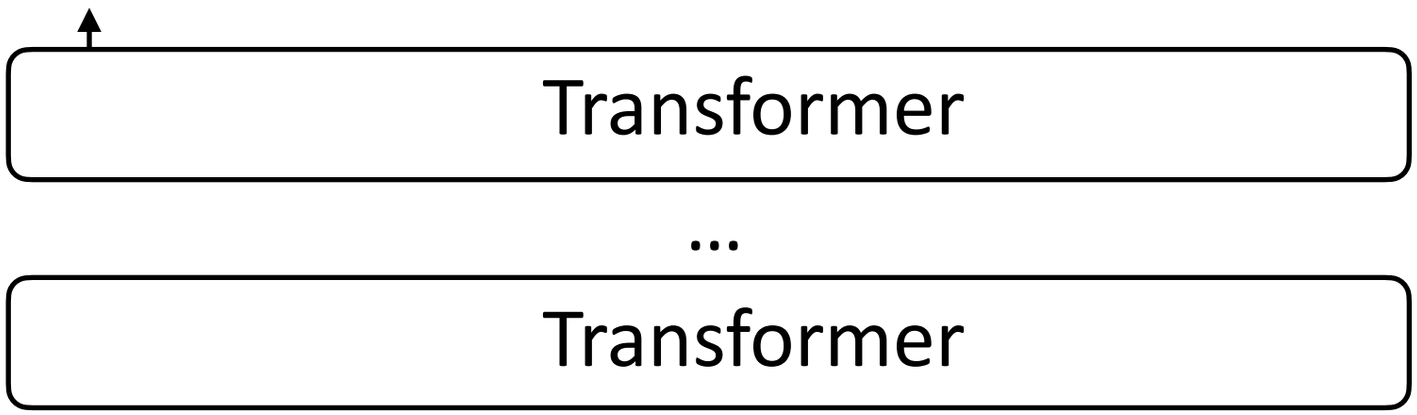
(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

- Artificial [CLS] token is used as the vector to do classification from
- Sentence pair tasks (entailment): feed both sentences into BERT
- BERT can also do tagging by predicting tags at each word piece

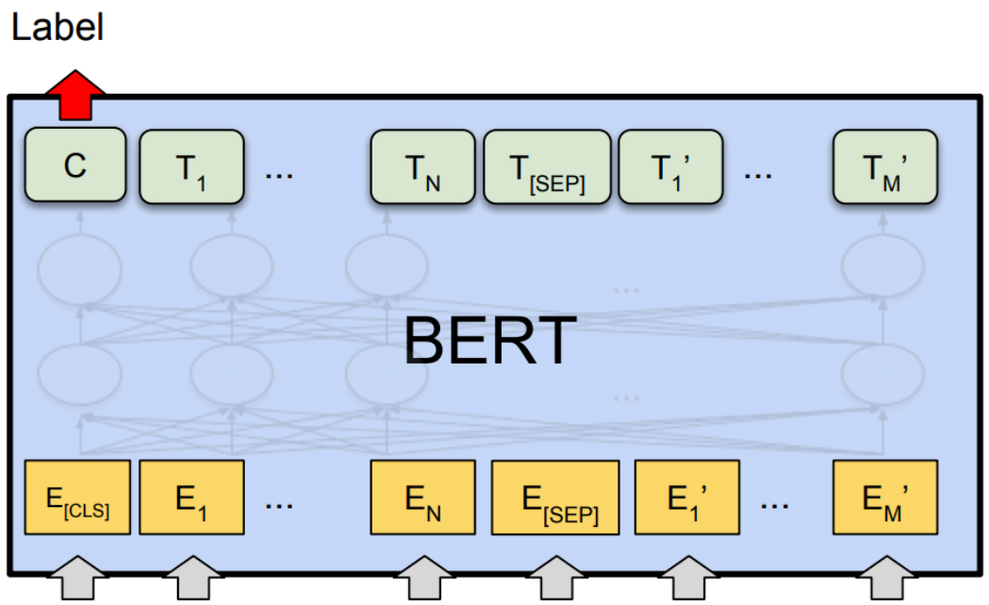
Devlin et al. (2019)

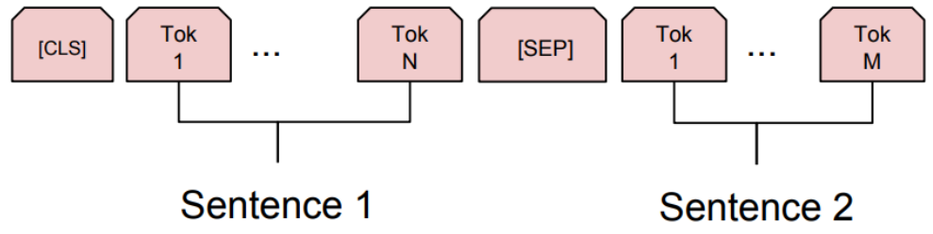
What can BERT do?

Entails (first sentence implies second is true)



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



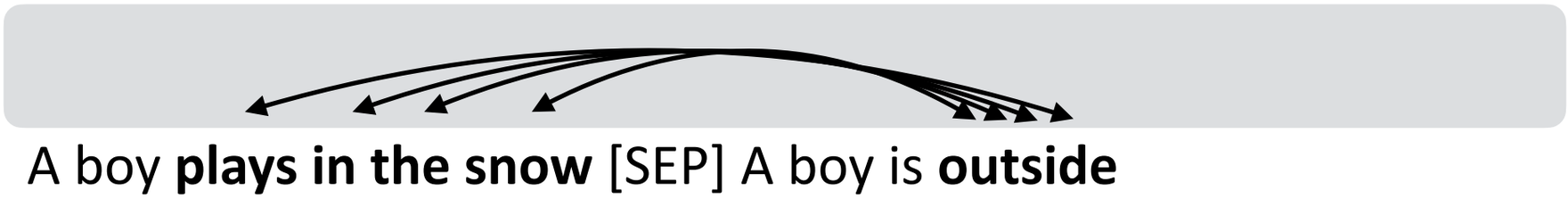


(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

- How does BERT model sentence pairs?
- Transformers can capture interactions between the two sentences, even though the NSP objective doesn't really cause this to happen

How do models do it?

A **man** is eating a sandwich [SEP] A **person** is eating a sandwich



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

- Transformers can easily learn to spot words or short phrases that are transformed
- **But**, models are often overly sensitive to lexical overlap

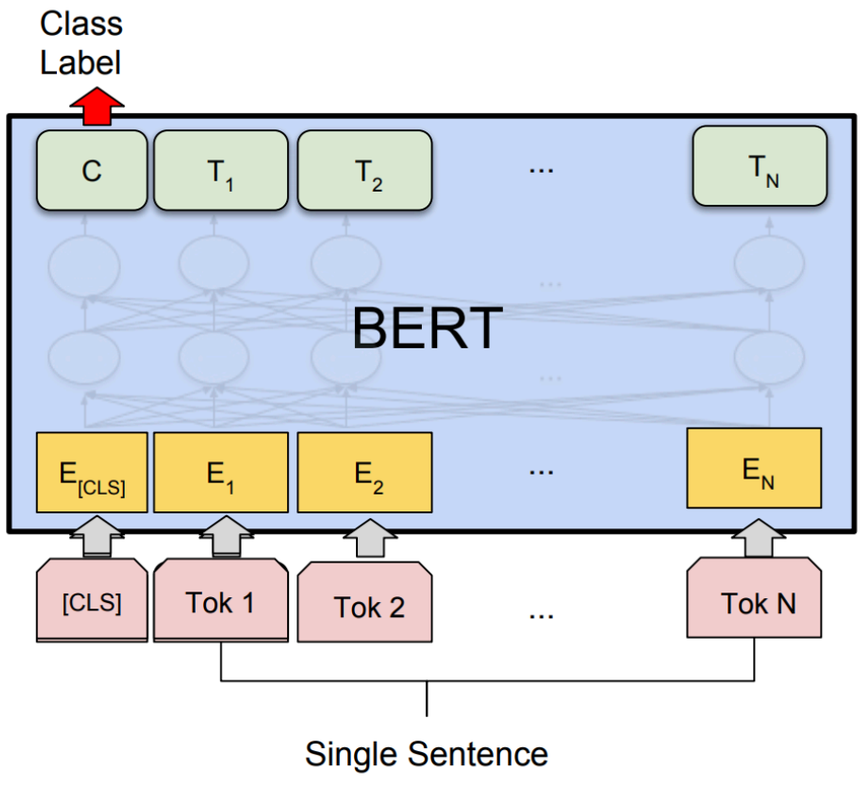
Williams et al. (2018)

What can BERT NOT do?

- BERT **cannot** easily generate text
  - Can fill in MASK tokens, but can't generate left-to-right (well, you could put MASK at the end repeatedly, but this is slow)
- Masked language models are intended to be used primarily for "analysis" tasks

Fine-tuning BERT

- Fine-tune for 1-3 epochs, batch size 2-32, learning rate 2e-5 - 5e-5



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

- Large changes to weights at top (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
- Often requires tricky learning rate schedules ("triangular" learning rates with warmup periods)

Hyperparameter Optimization/Data (RoBERTa)

(Liu et al. 2019)

- **Model:** Same as BERT (bidirectional encoder with up to 340M params)
- **Objective:** Same as BERT, but *train longer,* with *bigger batches,* run on *full paragraphs,* and *drop sentence prediction* objective
- **Data:**
  - **BERT corpus:** BooksCorpus + Wikipedia (16GB)
  - **Additional data**: CC-News + OpenWebText + Stories (~140GB)
- **Results:** are empirically much better than BERT

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**
$\text{BERT}_{\text{LARGE}}$						
with BOOKS + WIKI	13GB	256	1M	90.9/81.8	86.6	93.7

DeBERTa

(He et al. 2021)

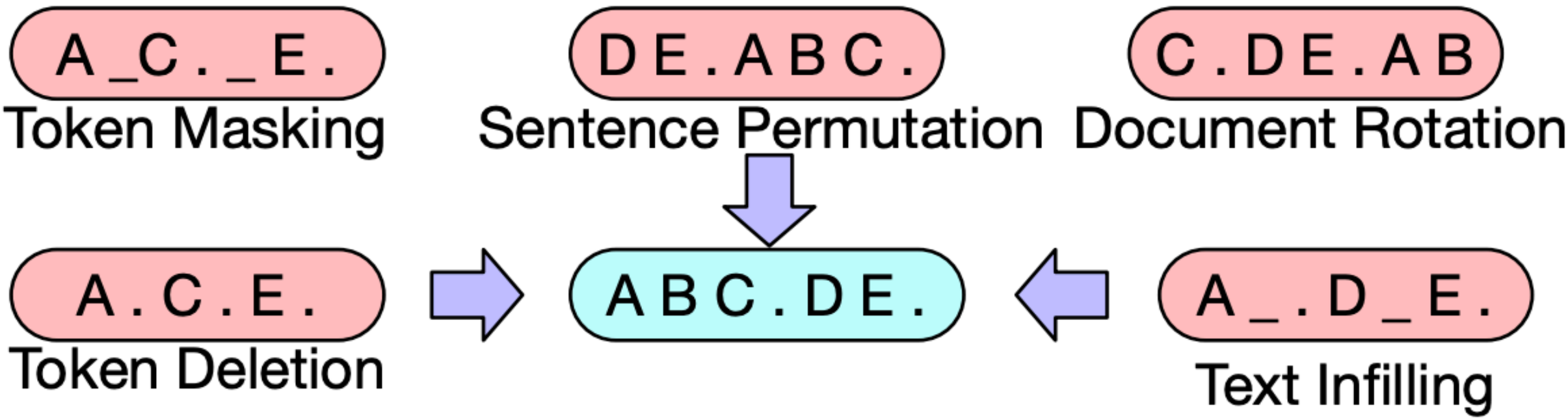
- **Model:** Transformer model with
  - "disentangled attention" treating relative position and content separately
  - absolute positional embeddings added at end of model
- **Objective:** Masked language modeling (w/ regularization by perturbing input embeddings)
- **Data:** 78GB Wikipedia, Reddit, and Subset of Common Crawl

Seq-to-Seq De-noising

How do we pre-train seq2seq models?

- LMs P(**w**): trained unidirectionally
- Masked LMs: trained bidirectionally but with masking
- How can we pre-train a model for P(**y**|**x**)?
- Well, why was BERT effective?
  - Predicting a mask requires some kind of text “understanding”.
- What would it take to do the same for sequence prediction?
- Requirements: (1) should use unlabeled data; (2) should force a model to attend from **y** back to **x**

BART



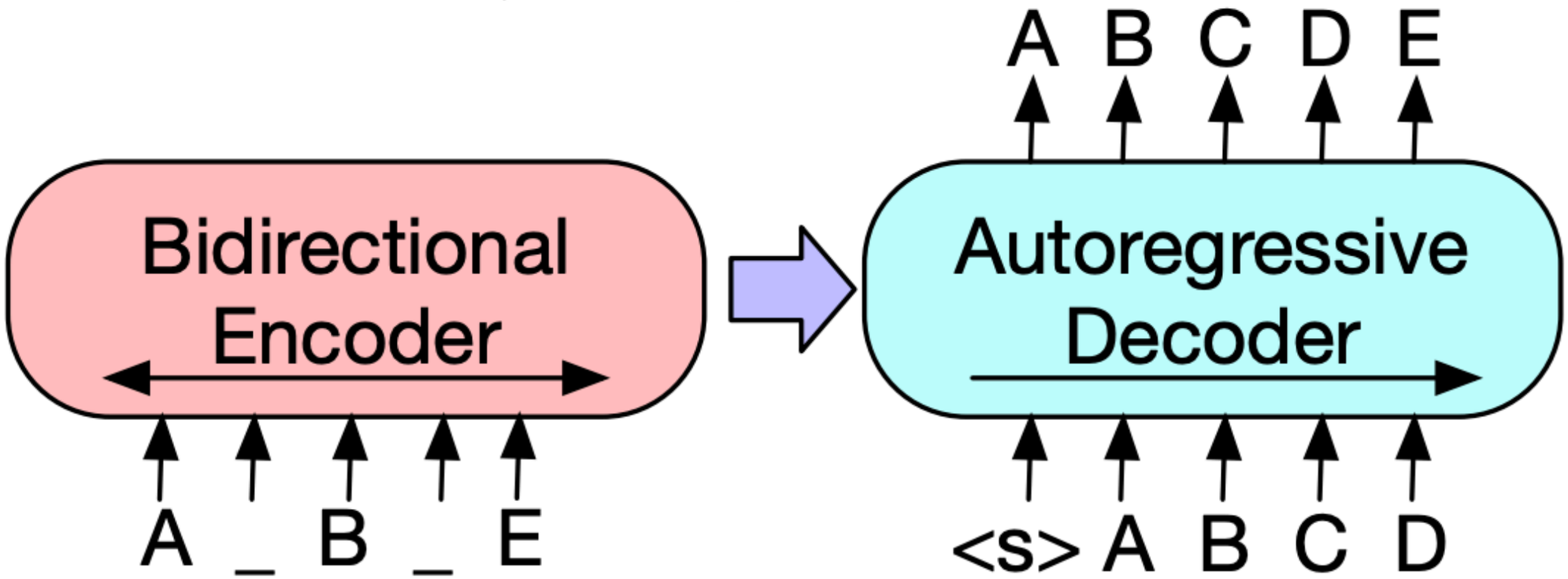
Infilling is longer spans than masking

- Several possible strategies for corrupting a sequence are explored in the BART paper

Lewis et al. (2019)

BART

- **Model & Objective**: Sequence-to-sequence Transformer trained on this data: permute/make/delete tokens, then predict full sequence autoregressively



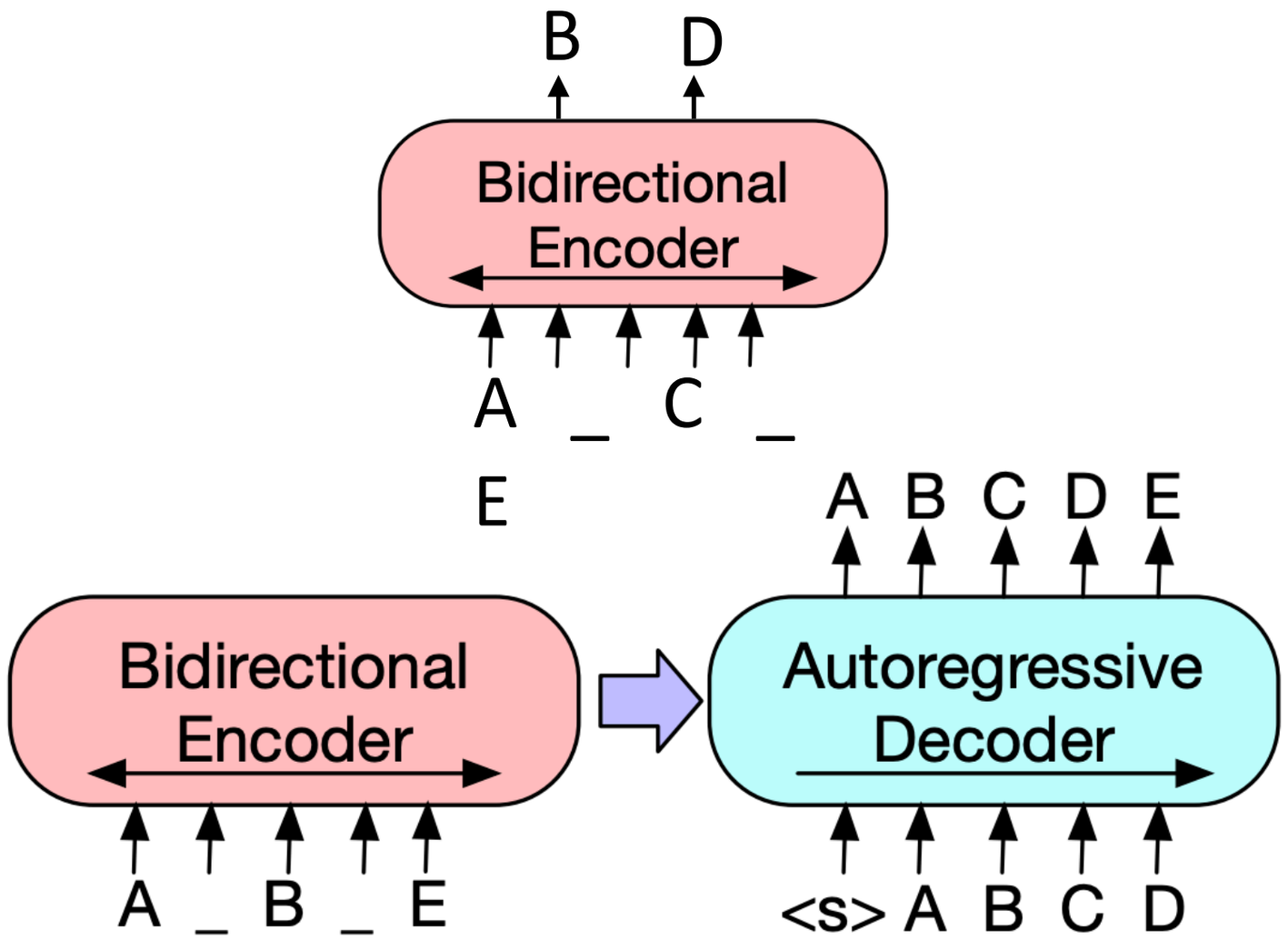
- **Data**: Same as RoBERTa; 160 GB of text

Lewis et al. (2019)

BERT vs. BART

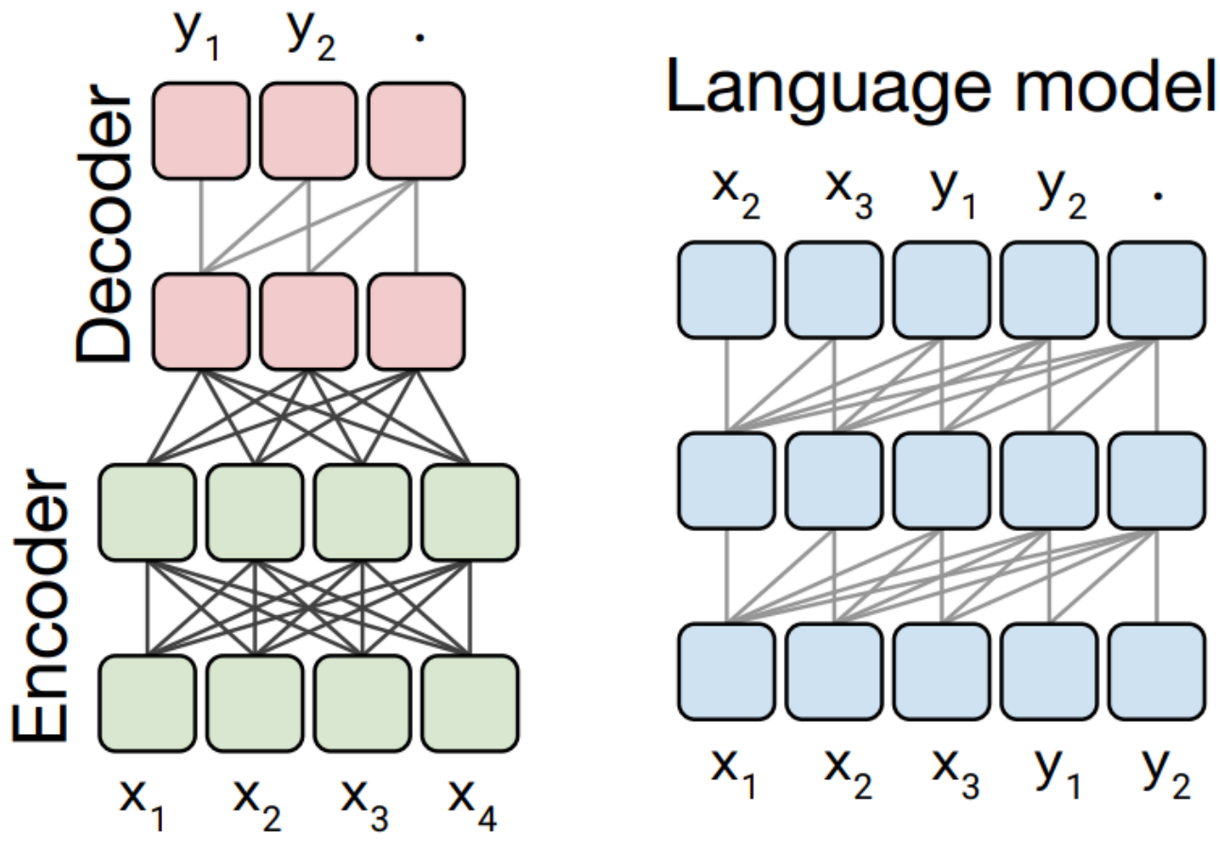
- BERT: only parameters are an encoder, trained with masked language modeling objective. Cannot generate text or do seq2seq tasks

- BART: both an encoder and a decoder. Can also use just the encoder wherever we would use BERT



Lewis et al. (2019)

Transformer Seq2seq Architecture



- Encoder-decoder model is structurally similar to your language model
- Modification: decoder now attends back to the input. But the input doesn't change, so this just needs to be encoded once

Image: Raffel et al. 2019

BART for Summarization

- **Pre-train** on the BART task: take random chunks of text, noise them according to the schemes described, and try to “decode” the clean text
- **Fine-tune** on a summarization dataset: a news article is the input and a summary of that article is the output (usually 1-3 sentences depending on the dataset)
- Can achieve good results even with **few summaries to fine-tune on**, compared to basic seq2seq models which require 100k+ examples to do well

Lewis et al. (2019)

BART for Summarization: Outputs

This is the first time anyone has been recorded to run a full marathon of 42.195 kilometers (approximately 26 miles) under this pursued landmark time. It was not, however, an officially sanctioned world record, as it was not an ”open race” of the IAAF. His time was 1 hour 59 minutes 40.2 seconds. Kipchoge ran in Vienna, Austria. It was an event specifically designed to help Kipchoge break the two hour barrier.

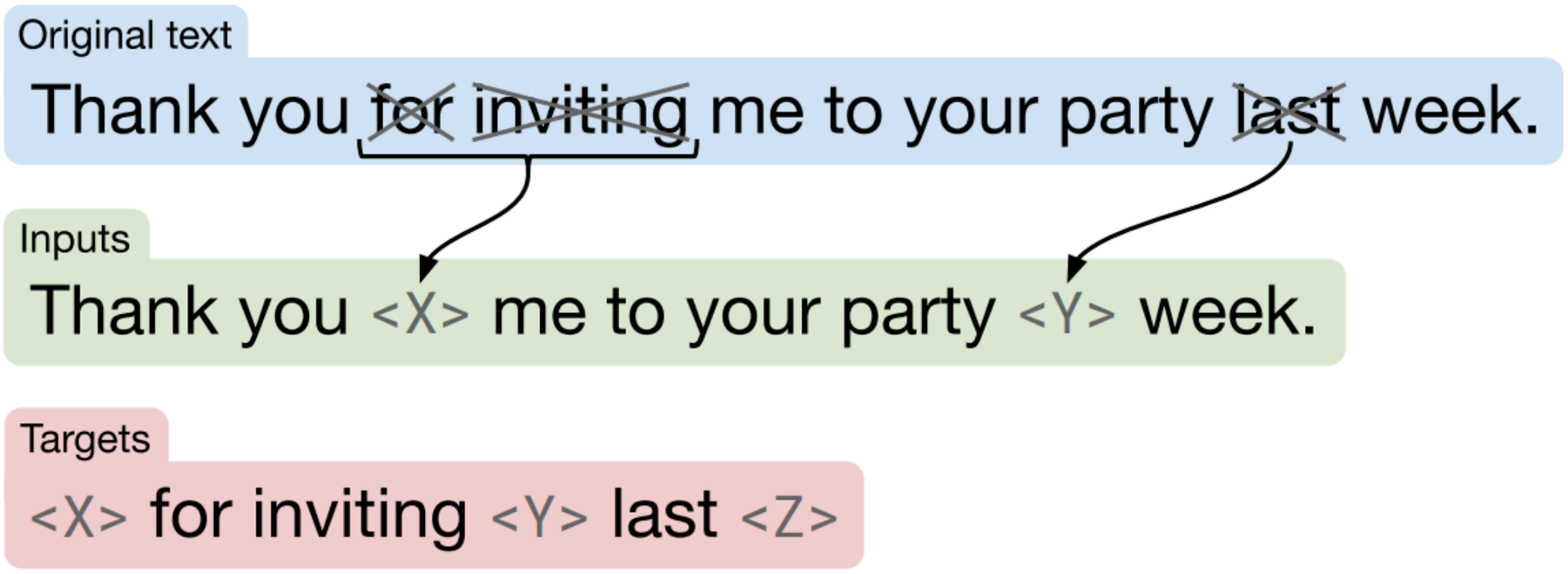
↓

Kenyan runner Eliud Kipchoge has run a marathon in less than two hours.

Lewis et al. (2019)

T5

- **Objective**: similar denoising scheme to BART (they were released within a week of each other in fall 2019).
- Input: text with gaps. Output: a series of phrases to fill those gaps.
- Lower computational cost compared to BART: predicts fewer tokens.



Raffel et al. (2019)

T5

	Number of tokens	Repeats	GLUE	CNNDM	EnDe	EnFr	EnRo
★	Full dataset	0	**83.28**	**19.24**	**26.98**	**39.82**	**27.65**
	2^{29}	64	**82.87**	**19.19**	**26.83**	**39.74**	**27.63**
	2^{27}	256	82.62	**19.20**	**27.02**	**39.71**	27.33
	2^{25}	1,024	79.55	18.57	26.38	39.56	26.80
	2^{23}	4,096	76.34	18.33	26.37	38.84	25.81
				summarization	machine translation		

- Colossal Cleaned Common Crawl: 750 GB of text
- We still haven't hit the limit of bigger data being useful for pre-training: here we see stronger MT results from the biggest data
- Models: larger than BART; up to 11B parameters

Raffel et al. (2019)

Successes of T5

- How can we handle a task like QA by framing it as a seq2seq problem?

Dataset	SQuAD 1.1
Input	At what speed did the turbine operate? \n (Nikola_Tesla) On his 50th birthday in 1906, Tesla demonstrated his 200 horsepower (150 kilowatts) 16,000 rpm bladeless turbine. ...
Output	16,000 rpm

- Format: *Question \n Passage* —> *Answer*
  encoder decoder

Raffel et al. (2019)

UnifiedQA

AB	**Dataset**	NarrativeQA
	Input	What does a drink from narcissus's spring cause the drinker to do? \n Mercury has awakened Echo, who weeps for Narcissus, and states that a drink from Narcissus's spring causes the drinkers to ``Grow dotingly enamored of themselves.'' ...
	Output	fall in love with themselves

Abstractive question, requires generating *free-form answer*

- Past work: different architectures for every QA formulation. (Span selection, answer generation, multiple choice, …)
- Now: one 11B parameter T5 model

Khashabi et al. (2020)

UnifiedQA

Multiple choice

Yes/no

MC	**Dataset**	MCTest
	Input	Who was Billy? \n (A) The skinny kid (B) A teacher (C) A little kid (D) The big kid \n Billy was like a king on the school yard. A king without a queen. He was the biggest kid in our grade, so he made all the rules during recess. ...
	Output	The big kid
YN	**Dataset**	BoolQ
	Input	Was America the first country to have a president? \n (President) The first usage of the word president to denote the highest official in a government was during the Commonwealth of England ...
	Output	no

- Past work: different architectures for every QA formulation. (Span selection, answer generation, multiple choice, …)
- Now: one 11B parameter T5 model

Khashabi et al. (2020)

Takeaways

- BART and T5 are useful for all sorts of seq2seq tasks involving language — so if you were going to use a seq2seq model, use one of these.
  (Caveat: need specialized models for language-to-code, like PLBART and CodeT5; multi-lingual tasks like mT5)
- UnifiedQA suggests that big generative models are good at generalizing across tasks and even to new tasks (although QA results have a long way to go)
- If we have a strong enough pre-trained model and train on enough tasks, can we generalize to new tasks?
- How do we specify those new tasks if they're not close to tasks we've already run on?
- Answer: **prompting**. But to do that well, we'll need to scale up further

Practicals of large pre-trained models

Impacts of Transfer Learning

- **Downstream performance:** Improved downstream task performance

- **Faster convergence:** Fewer epochs to reach same level of performance

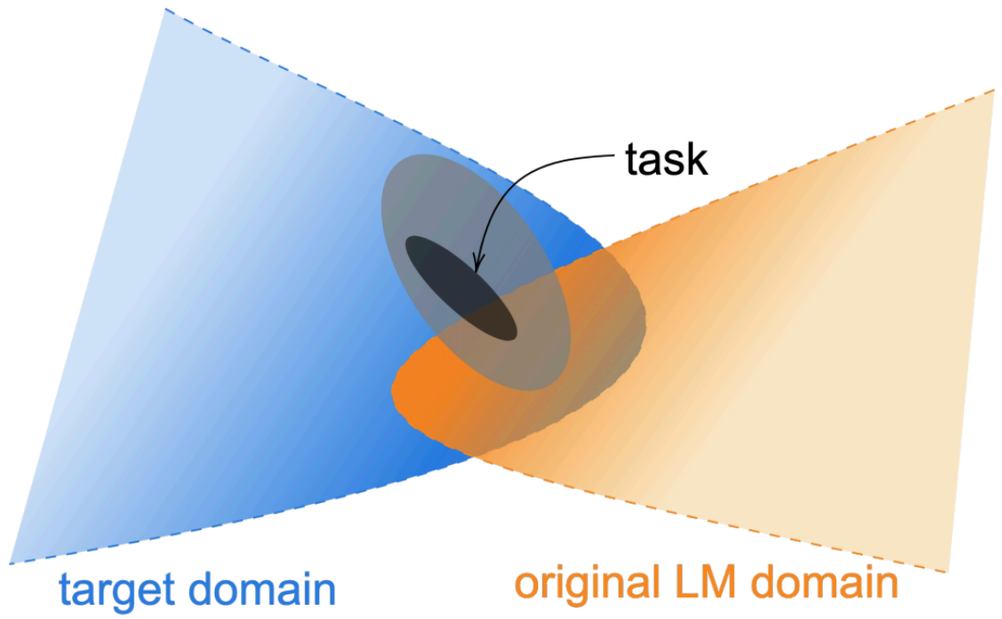
- **Data-efficiency:** Fewer datapoints required to achieve good performance

Is Pre-train then Fine-tune always appropriate ?

Pros	**Cons**
• One model for all downstream tasks • Amortize compute burden • All the benefits of transfer learning	• Good pre-training performance does not imply good downstream perf • No free lunch - one pre-training objective cannot perform well across all end tasks • No clear way to cross-validate pre-training stage

Continued Pre-training

- What can you do if you have lots of unlabeled text in your task domain?
  - Take RoBERTa and continue to pre-train with the MLM objective on your (large) unlabeled text
- Then, fine-tune with your (small) labeled data



(Guruangan et al. 2020)

Continued Pre-training

Domain	Pretraining Corpus	# Tokens	Size	$\mathcal{L}_{\text{ROB.}}$	$\mathcal{L}_{\text{DAPT}}$
BIOMED	2.68M full-text papers from S2ORC (Lo et al., 2020)	7.55B	47GB	1.32	0.99
CS	2.22M full-text papers from S2ORC (Lo et al., 2020)	8.10B	48GB	1.63	1.34
NEWS	11.90M articles from REALNEWS (Zellers et al., 2019)	6.66B	39GB	1.08	1.16
REVIEWS	24.75M AMAZON reviews (He and McAuley, 2016)	2.11B	11GB	2.10	1.93
ROBERTA (baseline)	see Appendix §A.1	N/A	160GB	‡1.19	-

Dom.	Task	ROBA.	DAPT
BM	CHEMPROT	$81.9_{1.0}$	$\mathbf{84.2}_{0.2}$
	†RCT	$87.2_{0.1}$	$\mathbf{87.6}_{0.1}$
CS	ACL-ARC	$63.0_{5.8}$	$\mathbf{75.4}_{2.5}$
	SCIERC	$77.3_{1.9}$	$\mathbf{80.8}_{1.5}$
NEWS	HYP.	$86.6_{0.9}$	$\mathbf{88.2}_{5.9}$
	†AGNEWS	$\mathbf{93.9}_{0.2}$	$\mathbf{93.9}_{0.2}$
REV.	†HELPFUL.	$65.1_{3.4}$	$\mathbf{66.5}_{1.4}$
	†IMDB	$95.0_{0.2}$	$\mathbf{95.4}_{0.2}$

(Guruangan et al. 2020)

Pre-training design choices

Objective	Data ($\mathcal{D}$)	Transform ($\mathcal{T}$)	Representation ($\mathcal{R}$)	Output ($\mathcal{O}$)
BERT	**Out-of-domain**	**BERT-Op**	**Bidirectional**	**Denoise Token**
TAPT	**Task data**	**BERT-Op**	**Bidirectional**	**Denoise Token**
DAPT	**In-domain**	**BERT-Op**	**Bidirectional**	**Denoise Token**
ELMO	**Out-of-domain**	**No-Op**	**Left-to-Right** and **Right-to-Left**	**Next Token**
GPT	**Out-of-domain**	**No-Op**	**Left-To-Right**	**Next Token**
XLNet	**Out-of-domain**	**No-Op**	**Random factorized**	**Next Token**
Electra	**Neural LM Data**	**Replace**	**Bidirectional**	**Real / Synthetic**
...	...	...	...	...

- We can generate many more objectives by taking this view
- Let the end-task choose which objectives are most useful

(Dery et al. 2021, Dery et al. 2022)

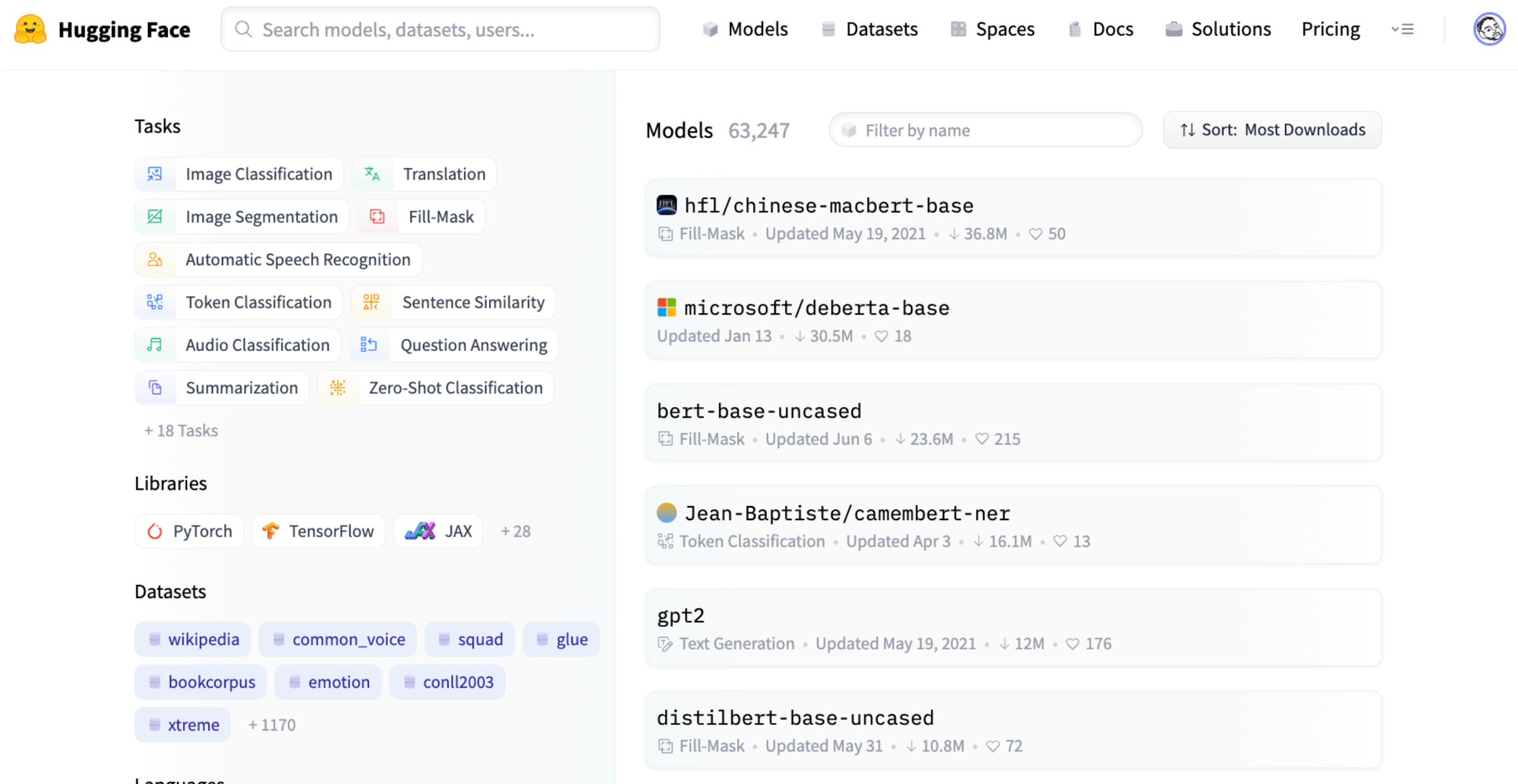
Practicals of using large pre-trained models

- Gradient accumulation
    - Fitting large batches lead to OOMs - run several smaller batches and back-prop to gather gradients before optimizer step
- Selective finetuning
    - Top few layers -> layer-norm layers -> Every thing else

HuggingFace Model Hub

- The HF model hub is one go-to source for models

https://huggingface.co/models



Questions?