Pretrained Representations:
GPT, ELMo, BERT, BART & T5

Antoine Bosselut

EPFL

Today's Outline

- **Lecture**
  - **Pretraining Transformers:** GPT
  - **Bidirectional Pretraining:** ELMo + BERT
  - **Sequence-to-sequence Pretraining:** BART + T5

Some slides adapted from Danqi Chen, Greg Durrett

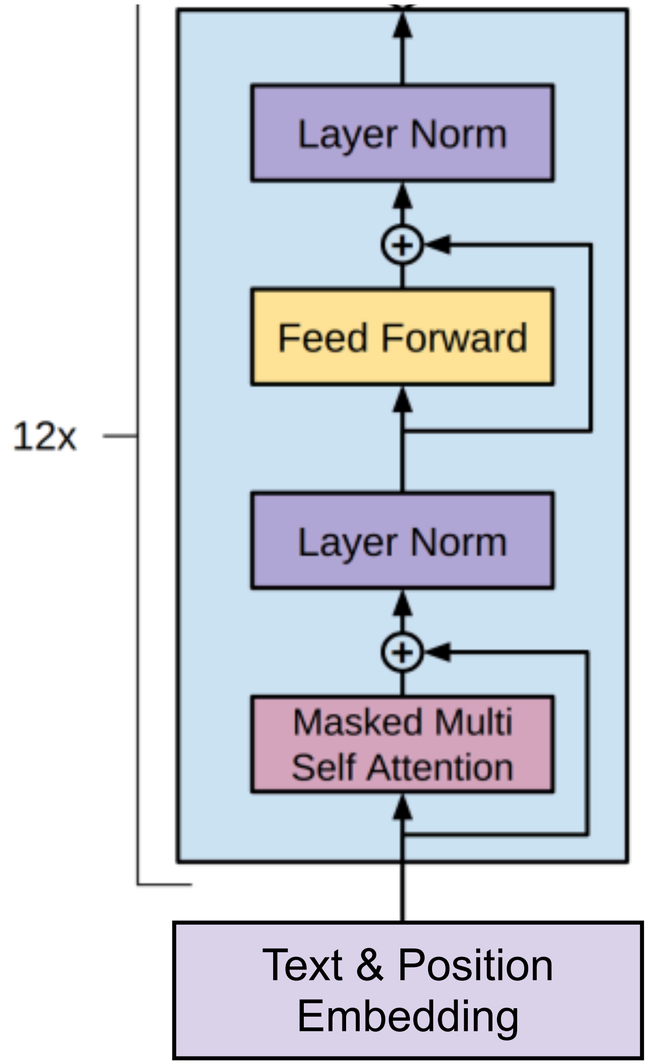
Pretrained Representations: **GPT**, **GPT-2**

Antoine Bosselut

EPFL

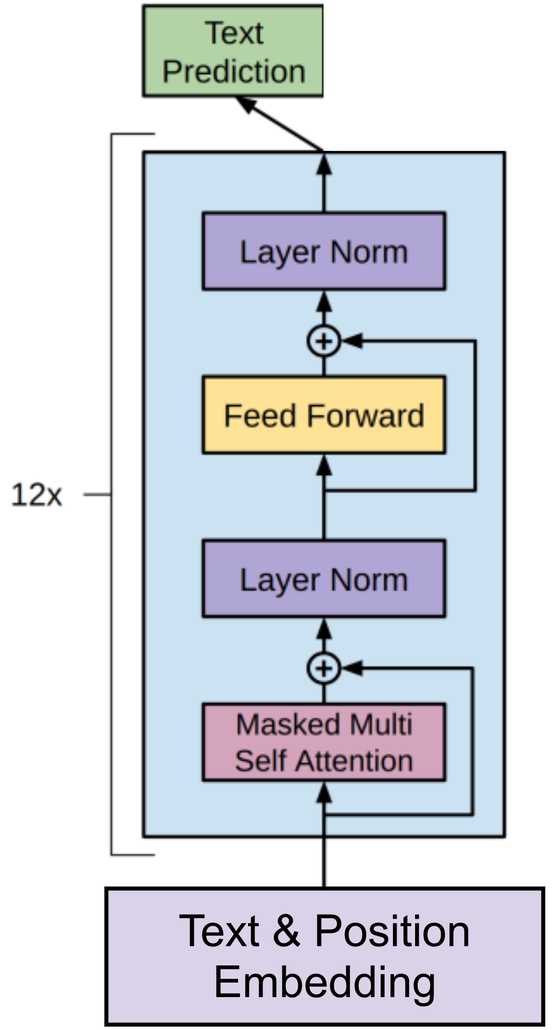
GPT: Generative Pretrained Transformer

- Called a *decoder* transformer
- **But**, actual GPT block mixes design of encoder and decoder from original transformer
- Uses masked multi-headed self-attention (**decoder**)
  - Token at each step can't attend to future tokens
- No cross-attention; only computes a self-attention over its history (**encoder**)



Training GPT

- Pretrained on TorontoBooks corpus: 7000 unpublished books (~13 GB)
- Corpus segmented broken up into windows of 512 tokens
  - can model long-range context during pretraining up to 512 tokens
- **Pretraining task:** next word prediction (i.e., causal language modelling)



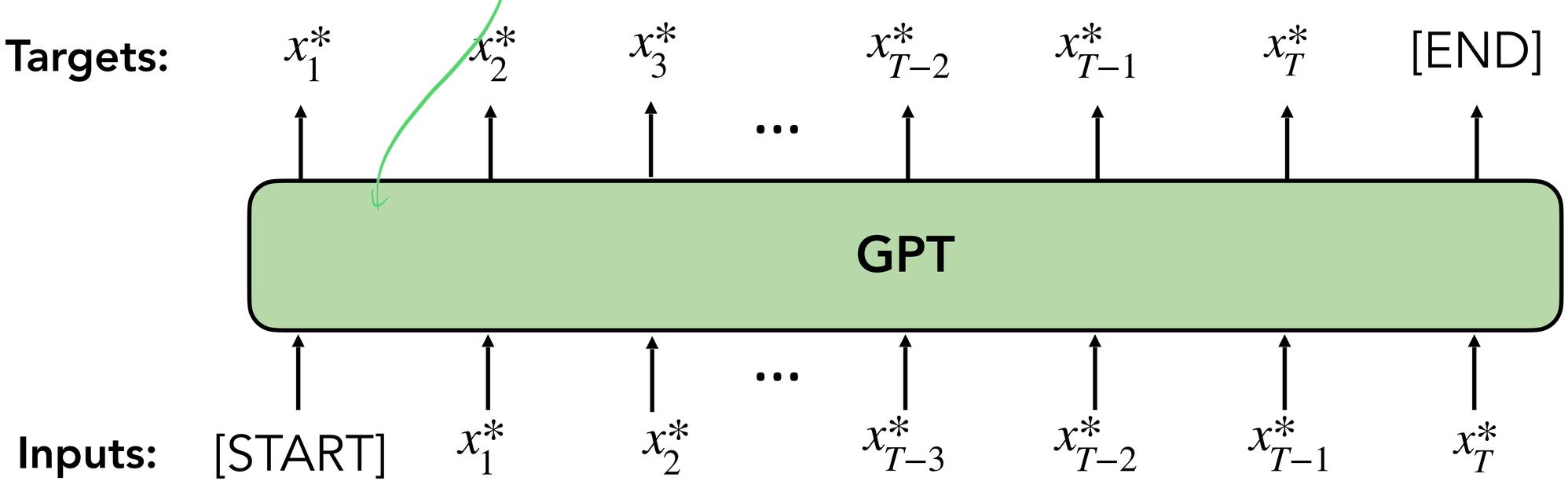
can be parallelized

Pretraining

- Minimize the negative log probability of the **gold*** sequences in your dataset

$$\mathscr{L} = -\sum_{t=1}^{T} \log P\left(x_t^* \mid \{x_s^*\}_{s<t}\right)$$

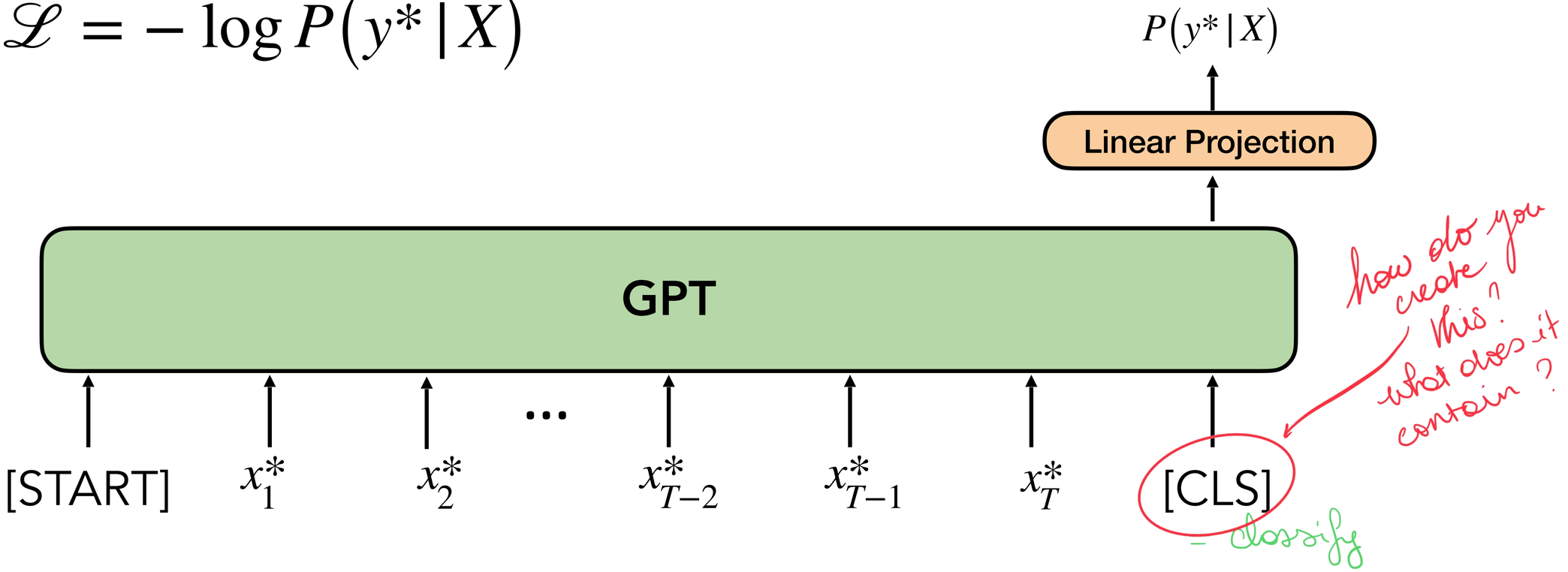
outputs the dist of all tokens that can be produced



Finetuning

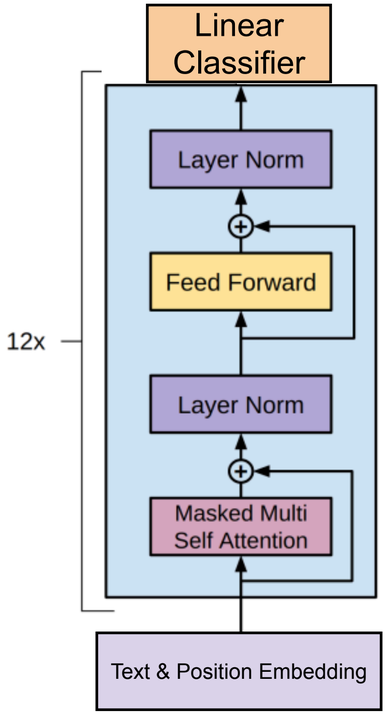
- Minimize the cross-entropy of the **gold** label of examples in your dataset

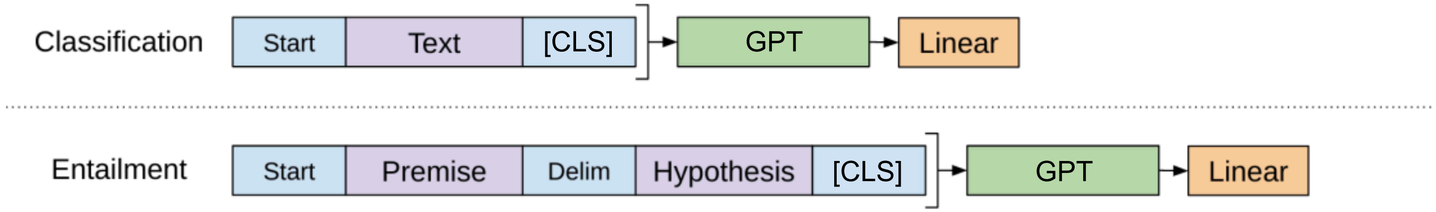
$$\mathscr{L} = -\log P\left(y^* \mid X\right)$$



Fine-tuning

- After pre-training, model can be fine-tuned by training on individual datasets
- Pretrained model used as initialisation for training on individual tasks





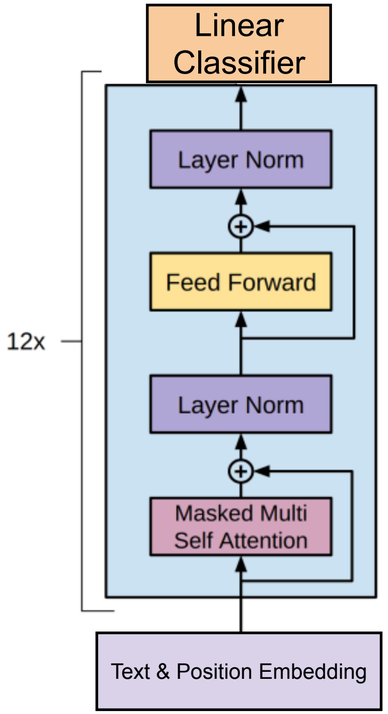
Premise: Antoine was giving a lecture.

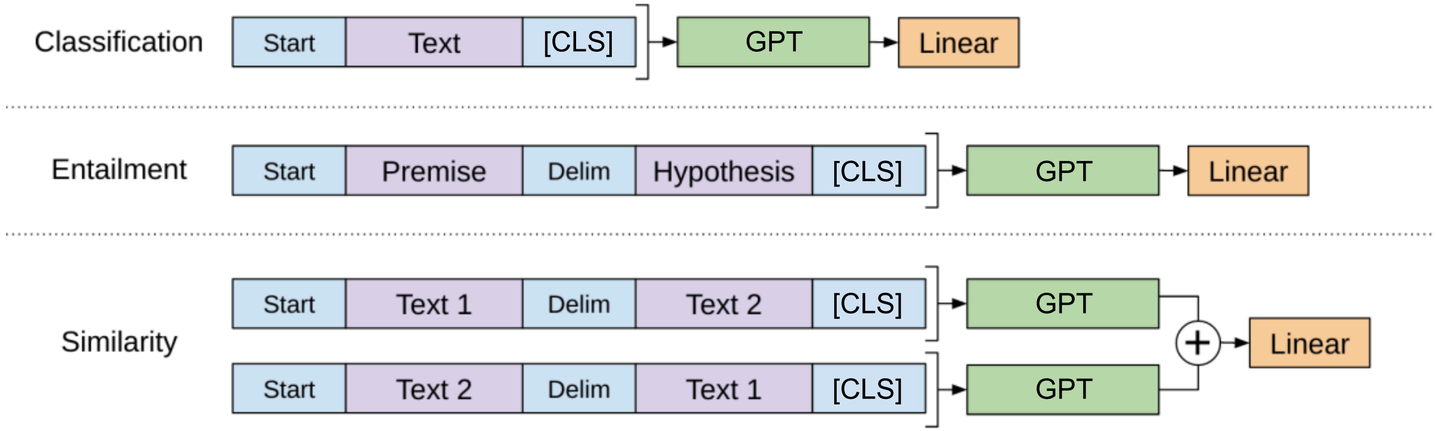
Hypothesis: The students were learning.

Entailed ? Contradicted ? Neutral ?

Fine-tuning

- After pre-training, model can be fine-tuned by training on individual datasets
- Pretrained model used as initialisation for training on individual tasks





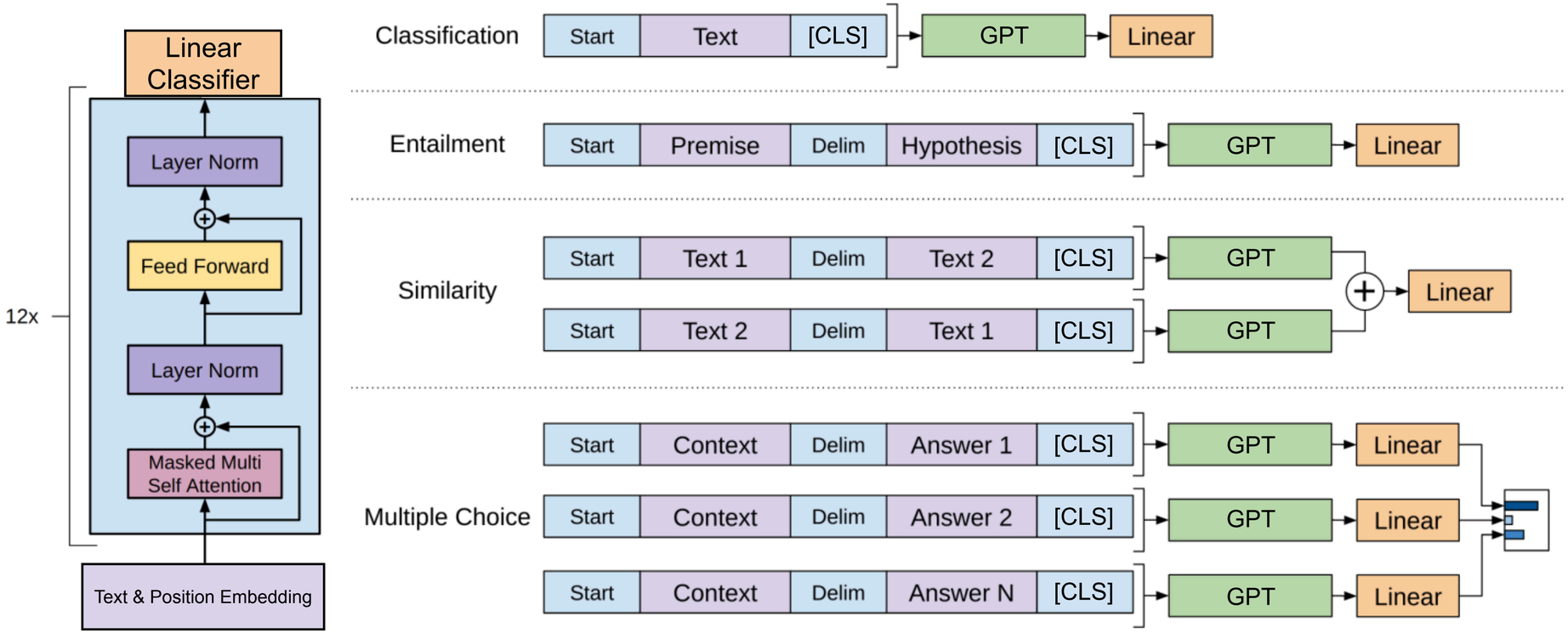
Statement 1: Antoine was giving a lecture.

Statement 2: Antoine was teaching.

Similar ? Not Similar ?

Fine-tuning

- After pre-training, model can be fine-tuned by training on individual datasets
- Pretrained model used as initialisation for training on individual tasks



Massive Improvements (back then)

Dataset	Task	SOTA	Ours
SNLI	Textual entailment	89.3	89.9
MNLI matched	Textual entailment	80.6	82.1
MNLI mismatched	Textual entailment	80.1	81.4
SciTail	Textual entailment	83.3	88.3
QNLI	Textual entailment	82.3	88.1
RTE	Textual entailment	61.7	56.0
STS-B	Semantic similarity	81.0	82.0
QQP	Semantic similarity	66.1	70.3
MRPC	Semantic similarity	86.0	82.3
RACE	Reading comprehension	53.3	59.0
ROCStories	Commonsense reasoning	77.6	86.5
COPA	Commonsense reasoning	71.2	78.6
SST-2	Sentiment analysis	93.2	91.3
CoLA	Linguistic acceptability	35.0	45.4
GLUE	Multi task benchmark	68.9	72.8

https://openai.com/research/language-unsupervised

Paradigm Shift

1️⃣ Pretraining Phase (General Training)

Goal: Learn general language patterns by predicting the next token in large amounts of text.

How it Works:

1. **Input Processing:**
   - Large amounts of text (e.g., books, articles, web pages) are tokenized into subwords.
   - Each input sequence gets a **start token** but **no CLS token** (CLS is used in BERT, not GPT).
2. **Causal Masked Self-Attention:**
   - GPT uses **causal masking**, meaning at each step, it can **only attend to previous tokens** (not future tokens).
   - This forces the model to learn **sequential text generation** rather than bidirectional understanding.
3. **Next-Token Prediction Objective:**
   - The model is trained to **predict the next token** given previous tokens.
   - Example:
     - Input: `"The cat sat on the"`
     - Target Output: `"mat"`
     - The model learns to output `"mat"` based on `"The cat sat on the"`
4. **Loss Function:**
   - Cross-entropy loss is used to compare the predicted next token with the actual next token.
   - The model updates its weights via backpropagation.

Outcome:

- After pretraining, GPT has learned **general language structures**, common word associations, and grammatical patterns.
- However, it **hasn't specialized** in specific tasks (e.g., translation, summarization, coding).

4. **(Optional) Reinforcement Learning with Human Feedback (RLHF):**
   - Used in models like **ChatGPT** to make outputs more aligned with human preferences.
   - Involves:
     - Humans ranking model responses.
     - A reward model learning these preferences.
     - Reinforcement learning adjusting GPT's behavior.

Outcome:

- GPT now **specializes** in the fine-tuning task.
- It still **retains general pretraining knowledge** but adapts to new patterns.

2️⃣ Fine-Tuning Phase (Task-Specific Training)

Goal: Adapt the pretrained model to perform a **specific task (e.g., summarization, Q&A, chatbot dialogue).**

How it Works:

1. **New Dataset:**
   - The model is trained on a smaller, **task-specific** dataset.
   - Example: If fine-tuning GPT for sentiment analysis, we train it on labeled reviews ("positive" or "negative").
2. **Fine-Tuning with Supervised Learning:**
   - Unlike pretraining, where the model **just predicts the next word**, fine-tuning can involve:
     - **Custom formatting** (e.g., adding instruction prompts).
     - **Supervised learning** (model output is compared against labeled data).
     - **Loss function modifications** (depending on the task).
3. **Parameter Updates:**
   - The pretrained GPT weights are **further updated** based on the new dataset.
   - This allows GPT to specialize in the new task while **retaining general language understanding**.

- **Fundamental change:** No more training architectures (e.g., LSTM, GRU) initialized with pretrained embeddings (Word2vec, GloVe, fastText, etc.).

- **Finetuning pretrained full models**

Example: Pretraining vs. Fine-Tuning in Action

Pretrained GPT-3 (General Model)

💬 **Input:** `"The capital of France is"`
🤖 **GPT-3:** `"Paris"` ✅ *(General knowledge from pretraining)*

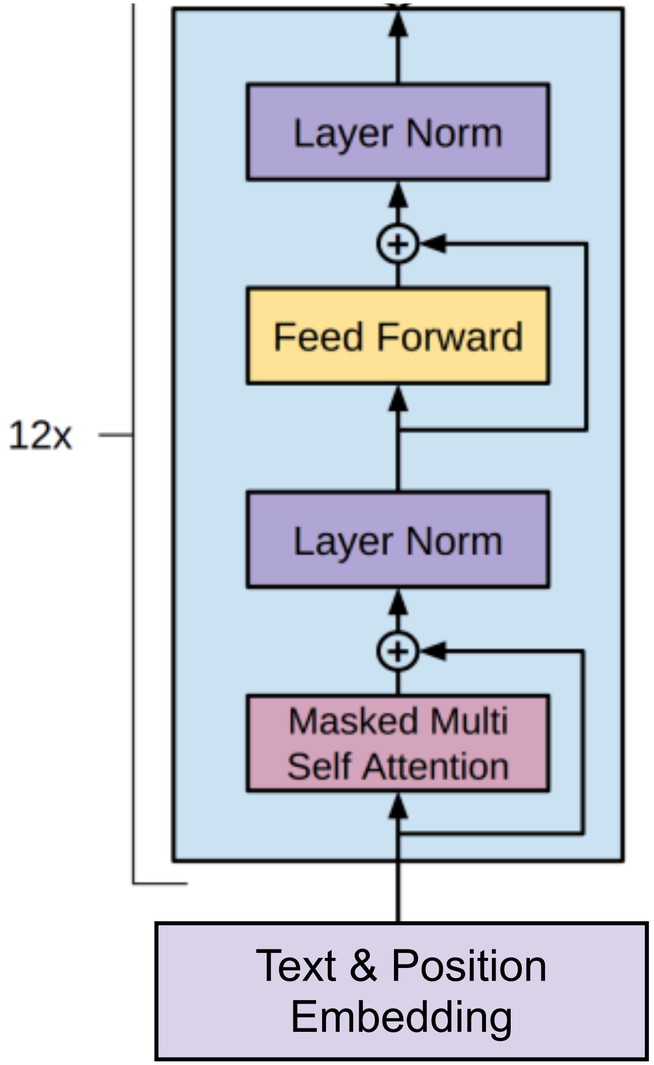
Fine-Tuned GPT-3 (QA System)

💬 **Input:** `"Question: What is the capital of France? Answer:"`
🤖 **Fine-tuned GPT:** `"The capital of France is Paris."` ✅ *(More structured response because of fine-tuning)*

GPT2

- Roughly the same architecture as GPT
- Trained on 40 GB of data: OpenWebText
  - Articles linked to from reddit threads
- Four model sizes:

Parameters	Layers	d_{model}
117M	12	768
345M	24	1024
762M	36	1280
1542M	48	1600



GPT2

Dataset	Metric	Our result	Previous record	Human
Winograd Schema Challenge	accuracy (+)	70.70%	63.7%	92%+
LAMBADA	accuracy (+)	63.24%	59.23%	95%+
LAMBADA	perplexity (–)	8.6	99	~1–2
Children's Book Test Common Nouns (validation accuracy)	accuracy (+)	93.30%	85.7%	96%
Children's Book Test Named Entities (validation accuracy)	accuracy (+)	89.05%	82.3%	92%
Penn Tree Bank	perplexity (–)	35.76	46.54	unknown
WikiText-2	perplexity (–)	18.34	39.14	unknown
enwik8	bits per character (–)	0.93	0.99	unknown
text8	bits per character (–)	0.98	1.08	unknown
WikiText-103	perplexity (–)	17.48	18.3	unknown

Less focus on fine-tuning; more focus on zero-shot prediction

https://openai.com/research/better-language-models

GPT2

SYSTEM PROMPT (HUMAN-WRITTEN)	*Miley Cyrus was caught shoplifting from Abercrombie and Fitch on Hollywood Boulevard today.*
MODEL COMPLETION (MACHINE-WRITTEN, SECOND TRY)	The 19-year-old singer was caught on camera being escorted out of the store by security guards. The singer was wearing a black hoodie with the label ‘Blurred Lines’ on the front and ‘Fashion Police’ on the back. Scroll down for video Shoplifting: Miley Cyrus was caught shoplifting from Abercrombie and Fitch on Hollywood Boulevard today (pictured) The singer was wearing a black hoodie with the label ‘Blurred Lines’ on the front and ‘Fashion Police’ on the back The singer was also wearing a pair of black-rimmed glasses, a black jacket, black jeans and black sandals. She was carrying a pair of black and white striped gloves and a small black bag.

Increase in Scale

- GPT (Jun 2018): 117 million parameters, trained on 13GB of data (~1 billion tokens)
- GPT2 (Feb 2019): 1.5 billion parameters, trained on 40GB of data
- GPT3 (July 2020): 175 billion parameters, ~500GB data (~300 billion tokens)
- GPT4 (March 2023): 1.8T parameters (rumoured), ????? data

More to come in Weeks 7, 11!

Pretrained Representations:
ELMo, BERT

Antoine Bosselut

EPFL

Question

Was GPT the first large-scale pretrained neural representation?

Word Embeddings: word2vec!

Question

What's an issue with word embeddings?

1) Chico Ruiz made a spectacular **play** on Alusik's grounder {. . . }

2) Olivia De Havilland signed to do a Broadway **play** for Garson {. . . }

3) Kieffer was commended for his ability to hit in the clutch , as well as his all-round excellent **play** {. . . }

4) {. . . } they were actors who had been handed fat roles in a successful **play** {. . . }

5) Concepts **play** an important role in all aspects of cognition {. . . }

Question

Words have different meanings in different contexts!

1) Chico Ruiz made a spectacular **play** on Alusik's grounder {. . . }

2) Olivia De Havilland signed to do a Broadway **play** for Garson {. . . }

3) Kieffer was commended for his ability to hit in the clutch , as well as his all-round excellent **play** {. . . }

4) {. . . } they were actors who had been handed fat roles in a successful **play** {. . . }

5) Concepts **play** an important role in all aspects of cognition {. . . }

Question

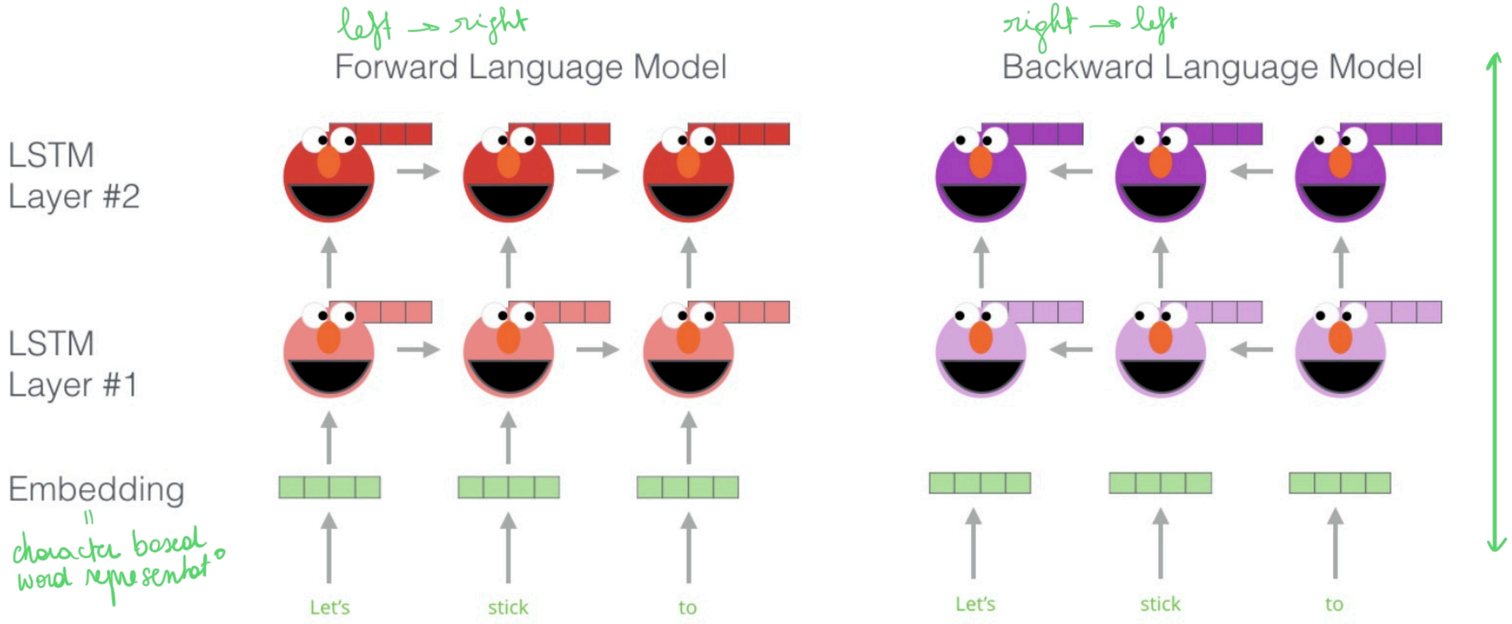
How might we integrate contextual information into word embeddings?

2018: Use bidirectional LSTMs!

ELMo

≠ word2vec, GloVe = static word representat°
= dynamic, context-dependent embeddings
provides ≠ embeddings for same word in ≠ contexts

- Train two-layer LSTM-based language model on a **large** corpus
- Use **hidden states** of the LSTMs for each token to compute an embedding of each word
- LSTM should be bidirectional



OUTPUT = final word embedding is a WEIGHTED SUM of these layers where weights are learnable params

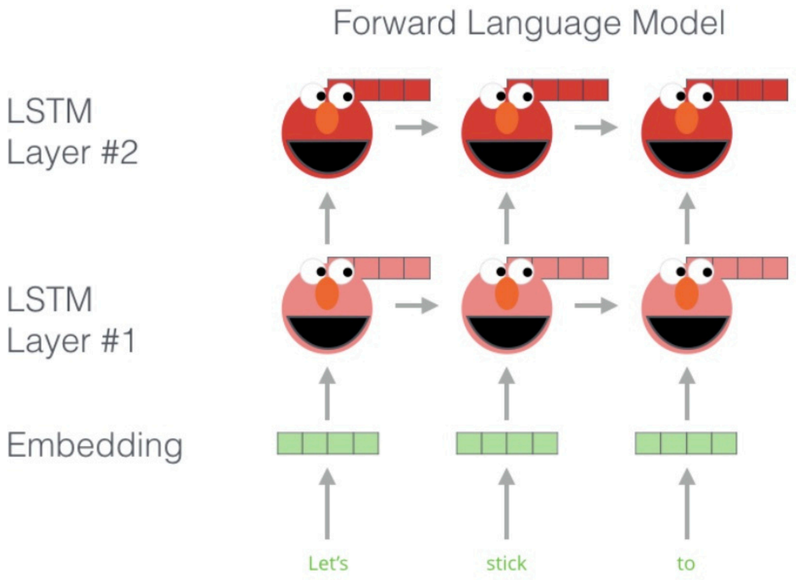
learns how much to use from each layer dynamically depending on task

Peters et al. (2018)

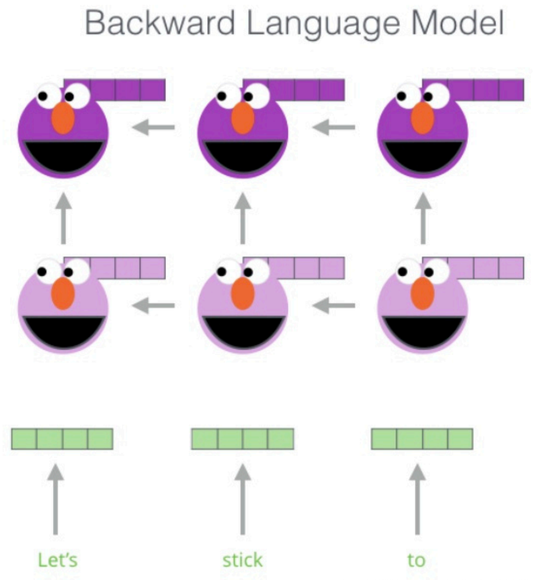
ELMo

1) PRETRAINED on large text corpus
2) FINETUNED for specific task by plugging them into models for sentiment analysis, qanswering ...

- Train two-layer LSTM-based language model on a **large** corpus
- Use **hidden states** of the LSTMs for each token to compute an embedding of each word
- LSTM should be bidirectional



$$\sum_{k=1}^{N} \Big(\log p(t_k \mid t_1, \ldots, t_{k-1}; \Theta_x, \overrightarrow{\Theta}_{LSTM}, \Theta_s) + \log p(t_k \mid t_{k+1}, \ldots, t_N; \Theta_x, \overleftarrow{\Theta}_{LSTM}, \Theta_s) \Big)$$



Peters et al. (2018)

ELMo

- Train two-layer LSTM-based language model on a **large** corpus
- Use **hidden states** of the LSTMs for each token to compute an embedding of each word
- LSTM should be bidirectional
- Use 1B word benchmark (**single sentences** — why might this be a problem?)

Input Embeddings Forward LSTM

$$\sum_{k=1}^{N} \Big(\log p(t_k \mid t_1, \ldots, t_{k-1}; \Theta_x, \overrightarrow{\Theta}_{LSTM}, \Theta_s) + \log p(t_k \mid t_{k+1}, \ldots, t_N; \Theta_x, \overleftarrow{\Theta}_{LSTM}, \Theta_s) \Big)$$

Vocab Projection

Backward LSTM

Peters et al. (2018)

Using ELMo Embeddings

L = # of layers (typically 2)
h_j = representation from layer j
s_j = learned weight for layer j

At layer 0, this is just word embeddings

$$\mathbf{ELMo}_k^{task} = E(R_k; \Theta^{task}) = \gamma^{task} \sum_{j=0}^{L} s_j^{task} \mathbf{h}_{k,j}^{LM}$$

At higher layers, concatenation of biLSTM hidden states

- γ^{task}: allows the task model to scale the entire ELMo vector
- s_j^{task}: softmax-normalized weights across layers

Learn both of these parameters

Why average the representation at each layer as opposed to the final one?
For different tasks, useful representations may be at different layers

Peters et al. (2018)

ELMo Improvements

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 ± 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 ± 0.19	90.15	92.22 ± 0.10	2.06 / 21%
SST-5	McCann et al. (2017)	53.7	51.4	54.7 ± 0.5	3.3 / 6.8%

Across the board improvements over SOTA when introduced

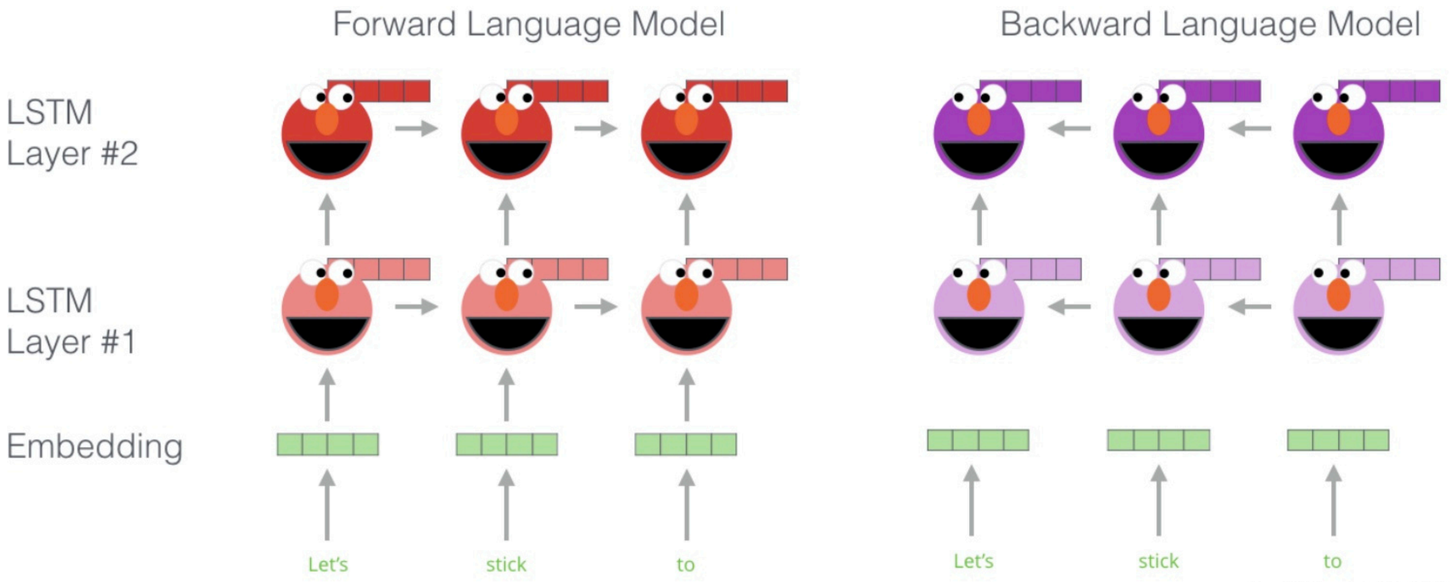
Peters et al. (2018)

Question

What's a problem with ELMo's notion of bidirectionality?

ELMo Issue

- ELMo (and bidirectional LSTMs / RNNs, in general) are **unidirectional** models masquerading as **bidirectional**
- Separate language model encodes the forward and backward sequence (and the representations are concatenated)

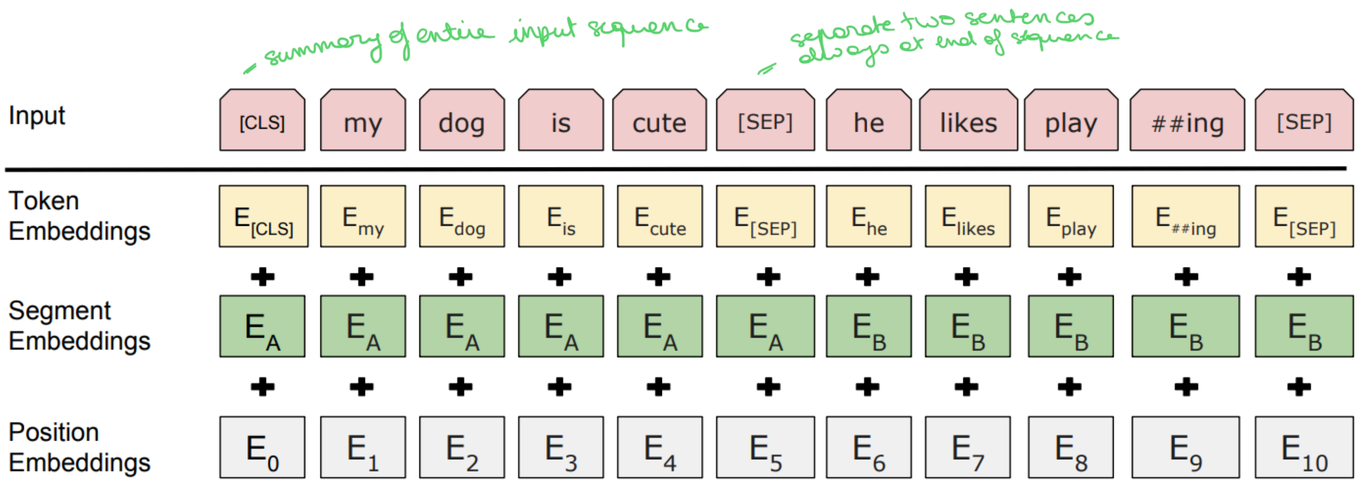


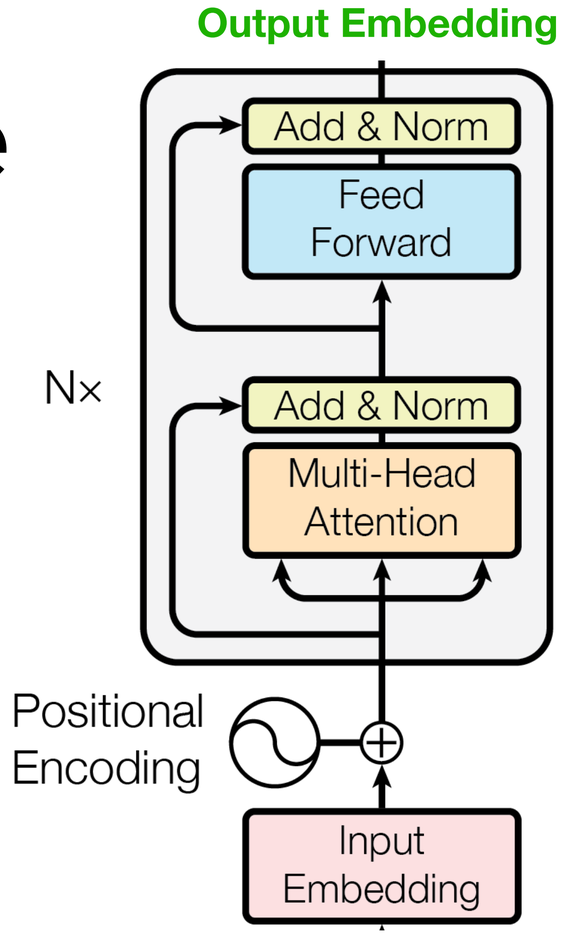
Peters et al. (2018)

What is BERT  ?

BERT Architecture

- **Transformer Encoder as we saw previously!**
- BERT-Base: 12 layers, *d*=768, 12 heads.
- Total params = **110M**
- BERT-Large: 24 layers, *d*=1024, 16 heads.
  Total params = **340M**





- Input embeddings for V = 30k word pieces
- Positional & segment embeddings

Devlin et al. (2019)

How is BERT trained?

Pretraining: Before

(Causal, Left-to-right) Language Modeling

I really enjoyed the movie we watched on ____

OpenAI

(Radford et al., 2018, 2019, many others)

Pretraining: Two Approaches

(Causal, Left-to-right) Language Modeling

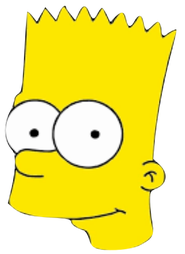
I really enjoyed the movie we watched on ____

OpenAI

(Radford et al., 2018, 2019, many others)

Masked Language Modeling

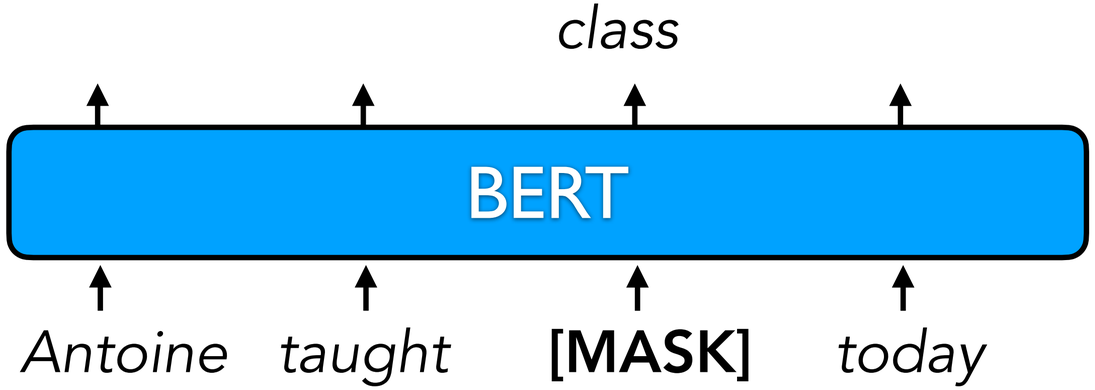
I really enjoyed the ____ we watched on Saturday!



(Devlin et al., 2018; Liu et al., 2020)

Masked Language Modeling (BERT)

- **Pretraining (self-supervised learning)**:
  - Done at scale on natural occurring sequences of text (any large corpus of raw text)

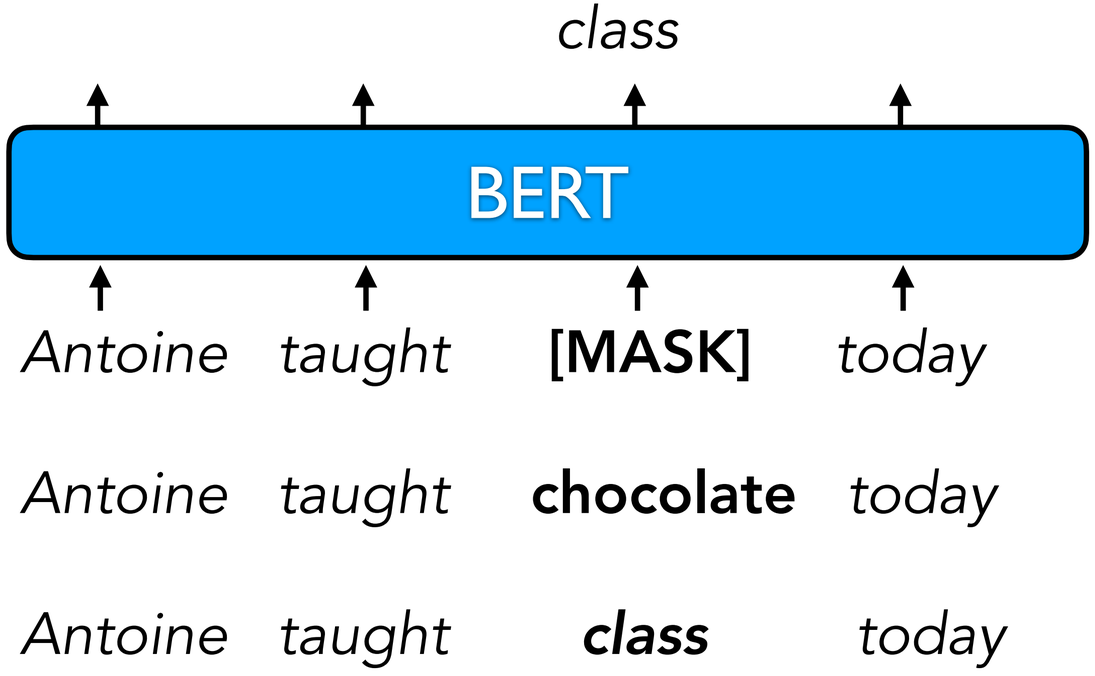


Devlin et al. (2019)

Masked Language Modeling (BERT)

- **Pretraining**: take a sequence of text, and predict 15% of the tokens

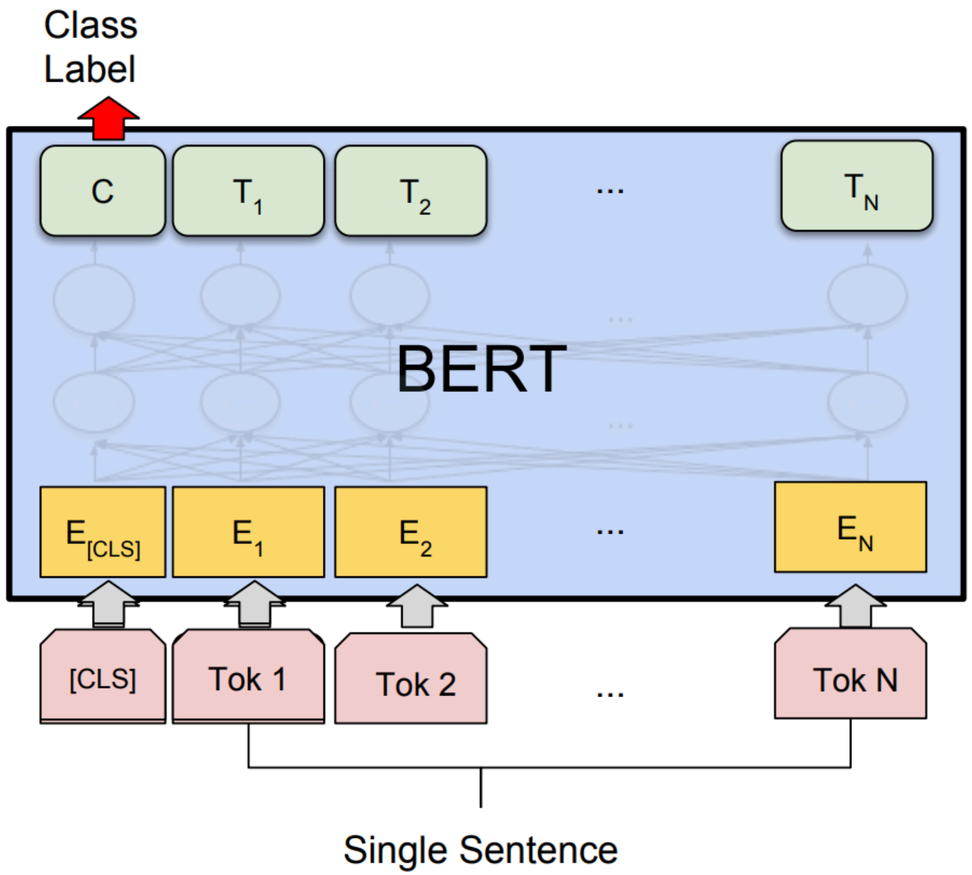
- **For 15% of tokens:**
  - Replace input token with [MASK] (80% of predictions)
  - Replace input token with a random token (10% of predictions)
  - Keep the same input token (10% of predictions)



Devlin et al. (2019)

Fine-tuning BERT

Fine-tuning BERT for classification



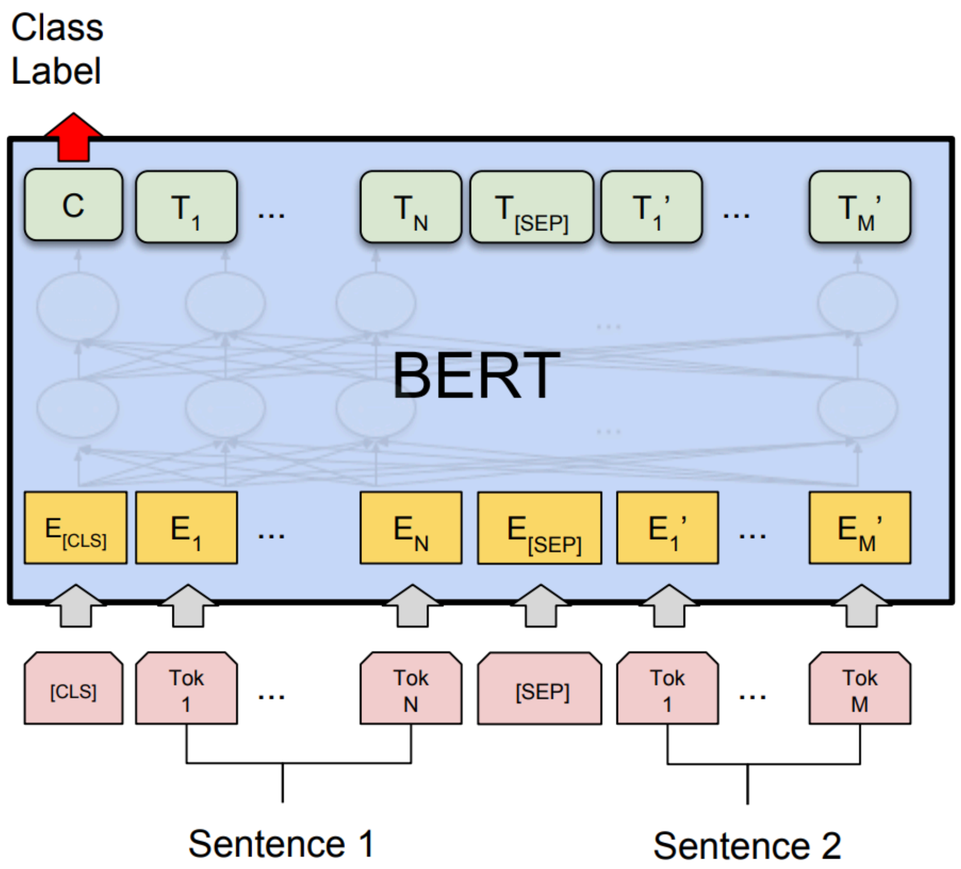
- **Fine-tuning:** Done after BERT has been pretrained (no more pretraining objectives)
- **During fine-tuning**, we update the parameters of the BERT model to learn the task
- A contextual embedding is outputted for each token in the input
- Prepend a special token **[CLS]** to the front of the sequence
- **Learn to** classify the output embedding for this token

How do we classify the output embedding for this token? Logistic Regression!

Devlin et al. (2019)

Fine-tuning BERT for classification

After pretraining

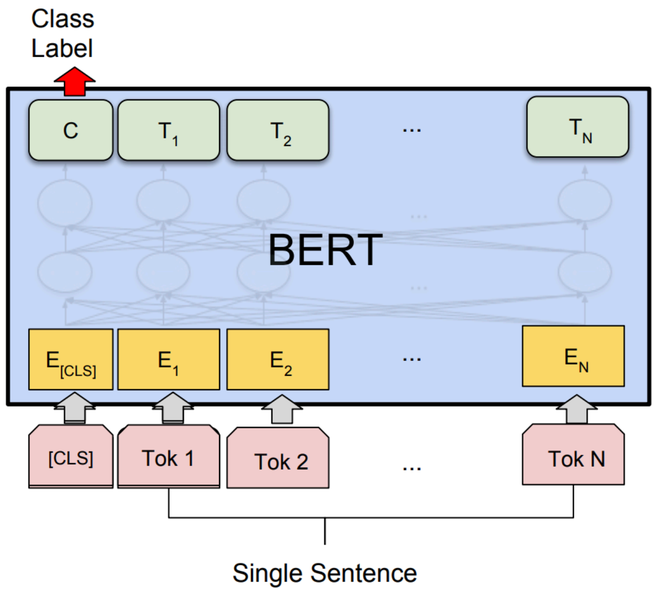


- A contextual embedding is outputted for each token in the input
- Prepend a special token **[CLS]** to the front of the sequence
- **Learn to** classify the output embedding for this token
- Separate sequences with special **[SEP]** token

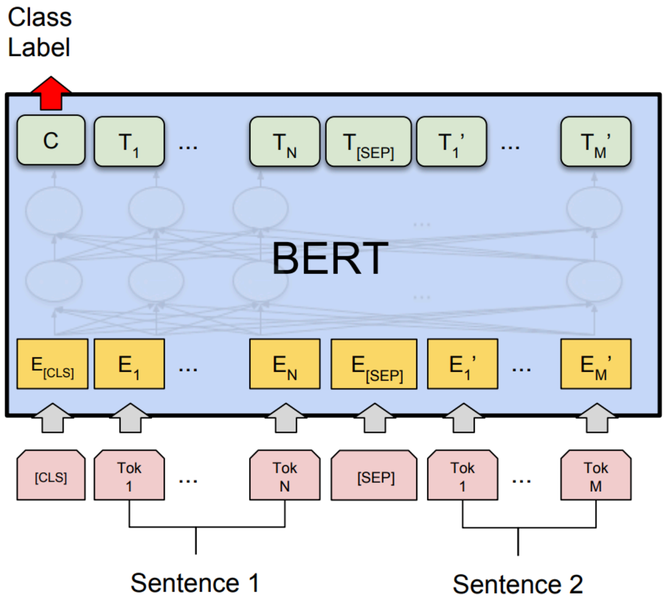
Can add special meta-tokens your vocabulary when they're needed!

Devlin et al. (2019)

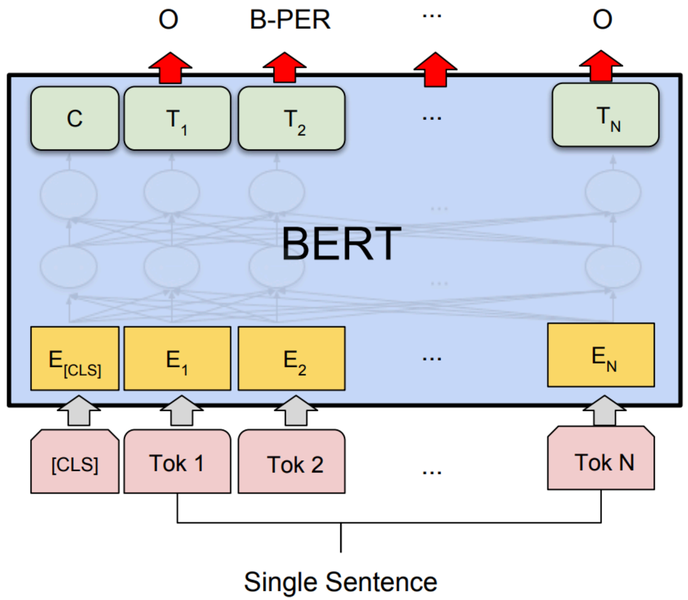
Single model starting point for many tasks



(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

- Re-using the same pretrained BERT model for fine-tuning on many tasks:
  - **Classification**: Take [CLS] output embedding as input features to classification model
  - **Sequence labeling**: Take output embedding for each token and classify individually

Devlin et al. (2019)

Question

Why do we put the [CLS] token at the front of the sequence?

Easiest place to put it. Bidirectionality ensures it attends to representations of all other tokens

GLUE: Prototypical NLU Benchmark

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)

BERT on GLUE

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
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.1	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**91.1**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**81.9**

For each of these tasks, a different BERT model is fine-tuned on the task data

Not the same fine-tuned BERT model that gets the same performance

Devlin et al. (2019)

BERT on GLUE

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
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.1	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**91.1**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**81.9**

Performance increases across the board

Devlin et al. (2019)

BERT on GLUE

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
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.1	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**91.1**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**81.9**

Performance increases across the board

Big increase over OpenAI GPT highlights importance of bidirectionality

Devlin et al. (2019)

Ablation Studies

Tasks	Dev Set MNLI-m (Acc)	QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
$\text{BERT}_{\text{BASE}}$	84.4	88.4	86.7	92.7	88.5
No NSP	83.9	84.9	86.5	92.6	87.9
LTR & No NSP	82.1	84.3	77.5	92.1	77.8
+ BiLSTM	82.1	84.1	75.7	91.6	84.9

Table 5: Ablation over the pre-training tasks using the $\text{BERT}_{\text{BASE}}$ architecture. "No NSP" is trained without the next sentence prediction task. "LTR & No NSP" is trained as a left-to-right LM without the next sentence prediction, like OpenAI GPT. "+ BiLSTM" adds a randomly initialized BiLSTM on top of the "LTR + No NSP" model during fine-tuning.

Importance of bidirectionality

Hyperparams				Dev Set Accuracy		
#L	#H	#A	LM (ppl)	MNLI-m	MRPC	SST-2
3	768	12	5.84	77.9	79.8	88.4
6	768	3	5.24	80.6	82.2	90.7
6	768	12	4.68	81.9	84.8	91.3
12	768	12	3.99	84.4	86.7	92.9
12	1024	16	3.54	85.7	86.9	93.3
24	1024	16	3.23	86.6	87.8	93.7

Table 6: Ablation over BERT model size. #L = the number of layers; #H = hidden size; #A = number of attention heads. "LM (ppl)" is the masked LM perplexity of held-out training data.

More parameters = better!

Devlin et al. (2019)

Question

Should we use BERT to generate embeddings or fine-tune the full model?

Fine-tuning vs. Embeddings

Pretraining	Adaptation	NER CoNLL 2003	SA SST-2	Nat. lang. inference MNLI	Nat. lang. inference SICK-E	Semantic textual similarity SICK-R	Semantic textual similarity MRPC	Semantic textual similarity 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
	Δ=🔥-❄️	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**
	Δ=🔥-❄️	0.2	0.5	0.0	1.0	2.3	6.7	4.2

- BERT outputs embeddings that can be frozen and provided to a different model, but BERT performs better when fully fine-tuned

Peters et al., 2019

Transfer Learning

Pretraining

Learn embeddings that can be used to seed a downstream model (ELMo)

-or-

Learn a model that can be fine-tuned for many downstream tasks (GPT, BERT)

Fine-tuning

Design a new model architecture whose embeddings are initialised with pretrained embeddings. Train this model on a task of interest

- or -

Take a pretrained model and train it further on data from a task of interest

Transfer Learning

Pretraining

Learn embeddings that can be used to seed a downstream model (ELMo)

-or-

Learn a model that can be fine-tuned for many downstream tasks (GPT, BERT)

Fine-tuning

Design a new model architecture whose embeddings are initialised with pretrained embeddings. Train this model on a task of interest

- or -

Take a pretrained model and train it further on data from a task of interest

Transfer Learning

Pretraining

Learn embeddings that can be used to seed a downstream model (ELMo)

-or-

Learn a model that can be fine-tuned for many downstream tasks (GPT, BERT)

Fine-tuning

Design a new model architecture whose embeddings are initialised with pretrained embeddings. Train this model on a task of interest

- or -

Take a pretrained model and train it further on data from a task of interest

Transfer Learning

Pretraining

Uses simple training objectives

Requires tons of data

Resultant model often not useful yet

Slow & expensive; can often only do once

Fine-tuning

Done on smaller datasets

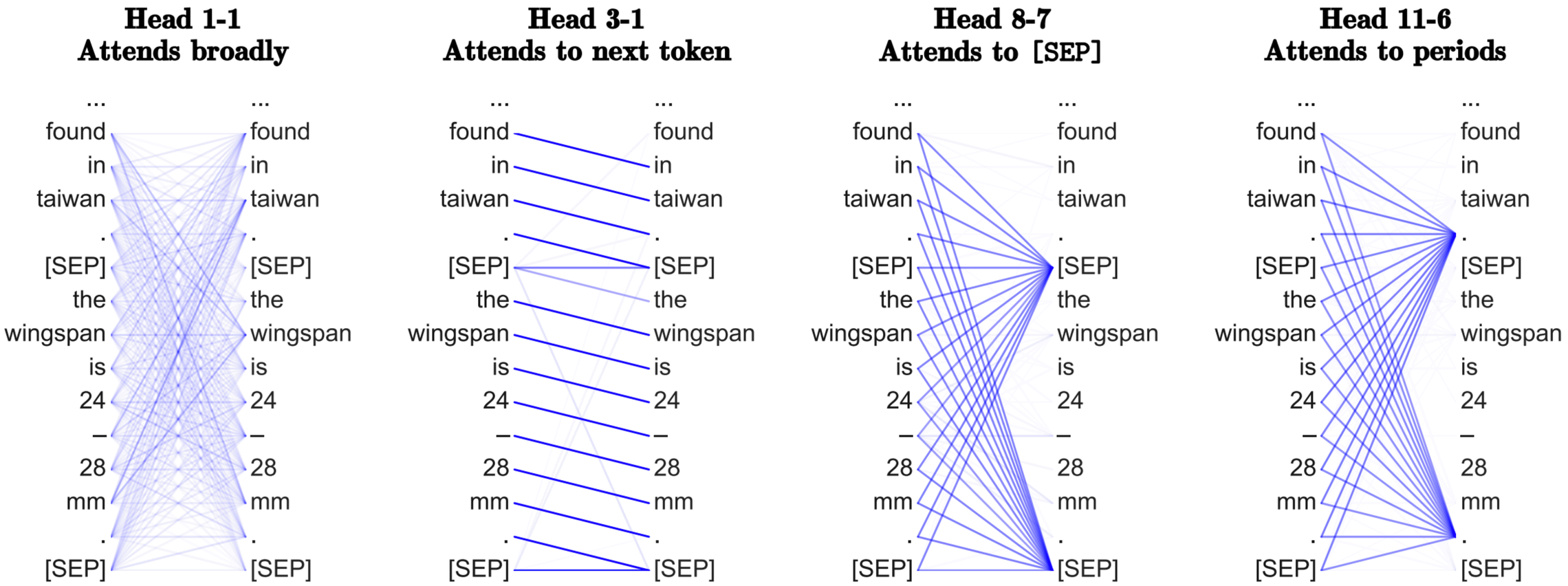
Trained on data with a more complex structure

Resultant model applied to task of interest

Typically cheaper; can afford multiple runs, hyper parameter tuning, etc.

What does BERT Learn?

What does BERT learn?



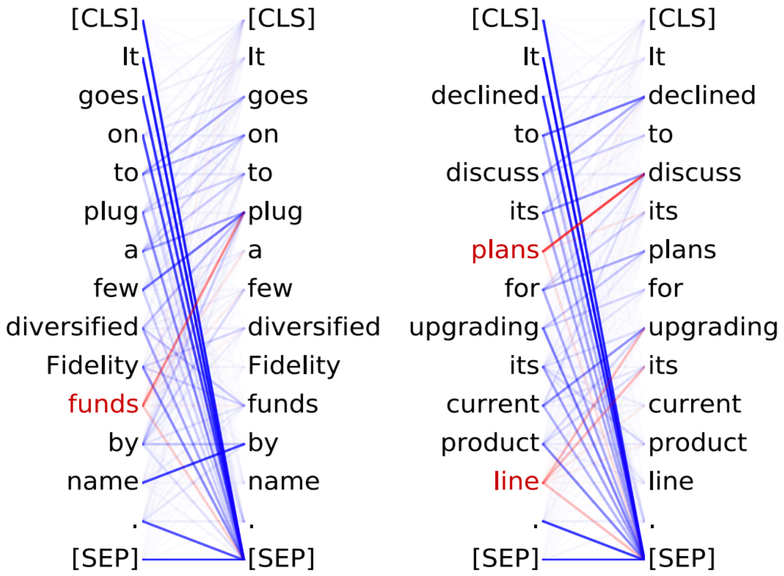
Transformer heads learn diverse concepts that map to positional, semantic, and syntactic relationships

Clark et al. (2019)

What does BERT learn?

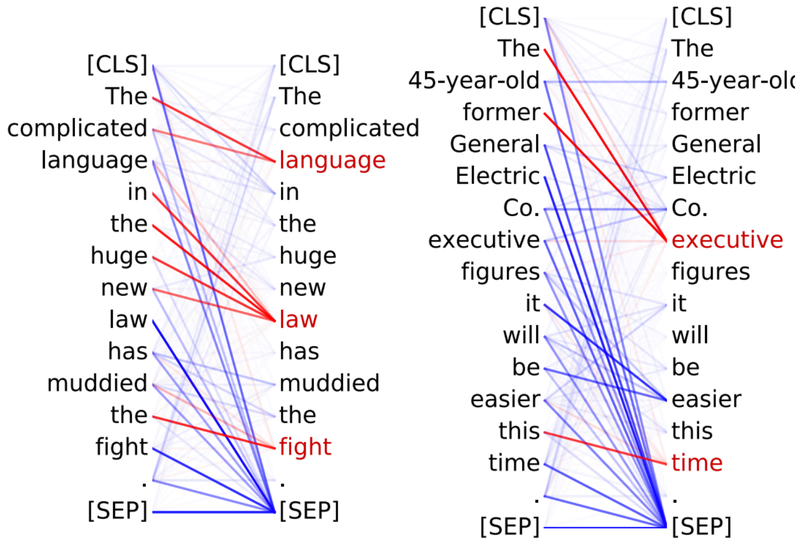
Head 8-10

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



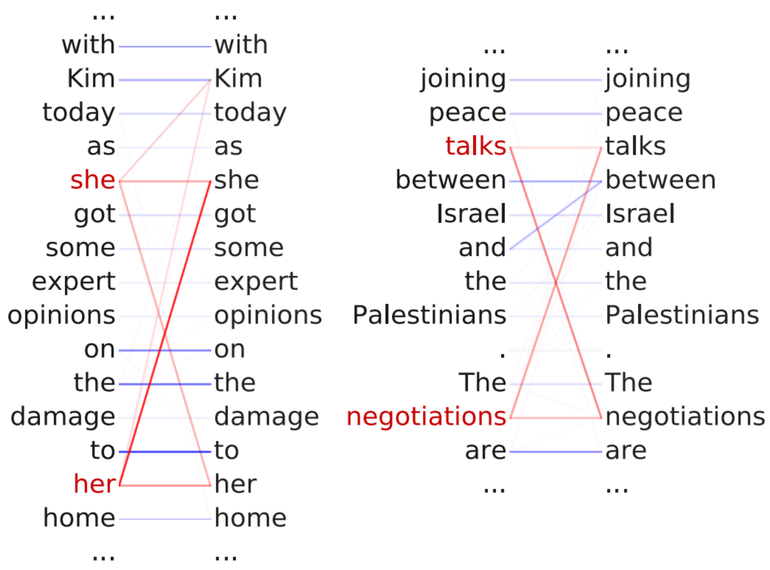
Head 8-11

- **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



Transformer heads learn diverse concepts that map to positional, semantic, and syntactic relationships

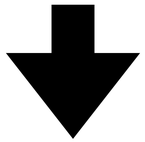
Clark et al. (2019)

Improvements to BERT

Feature	ELMo	BERT	GPT
Architecture	biLSTM	Transformer (encoder)	Transformer (decoder)
Contextualized Embeddings?	✅ Yes	✅ Yes	✅ Yes
Directionality	Bidirectional (LSTM)	Bidirectional (Self-Attention)	Unidirectional (Left-to-Right)
Pretraining Task	Bi-directional LM	Masked LM (MLM) + NSP	Causal LM (Next-word prediction)
Output Type	Word embeddings for each token, useful for downstream tasks	Sentence representation for classification & per-token embeddings	Continuously generates text token by token

Whole word masking

Obama was the president of the United States in 2010



[MASK] _**bama** _was _the _president _of _the _United _States _in _2010

vs.

[MASK] [MASK] _was _the _president _of _the _United _States _in _2010

Why might whole word masking be important?

Too easy to predict masked subwords if rest of word is in context — model doesn't learn as well

RoBERTa

- "Robustly Optimised BERT" — a collection of improvements to BERT
- Same architecture as BERT
- 160 GB of training data, rather than only 13 GB in 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

Liu et al. (2019)

DistilBERT

How Does DistilBERT Work?

Instead of training a new model from scratch, **DistilBERT "compresses" BERT** using a technique called **knowledge distillation**:

1. **Teacher-Student Learning:**
   - The full **BERT (teacher model)** trains a smaller **DistilBERT (student model)**.
   - DistilBERT **learns to mimic BERT's outputs**, making it more efficient.
2. **Removes Next Sentence Prediction (NSP):**
   - BERT uses **Masked Language Modeling (MLM) + NSP**, but DistilBERT **only uses MLM** (since NSP isn't that useful).
3. **Reduces Model Size:**
   - Uses **half the number of layers** (12 → 6).
   - Keeps **attention weights similar** to BERT's.

- Do we need all parameters of BERT, which require lots
- What if BERT was a much smaller?
- Train with distillation over soft target probabilities of BERT (and MLM)

$$\mathcal{L}_{distil} = -P_{BERT}\left(y_t^* \mid [\mathrm{M}], \{y_s^*\}_{s \neq t}\right) \log P_{dbert}\left(y_t^* \mid [\mathrm{M}], \{y_s^*\}_{s \neq t}\right)$$

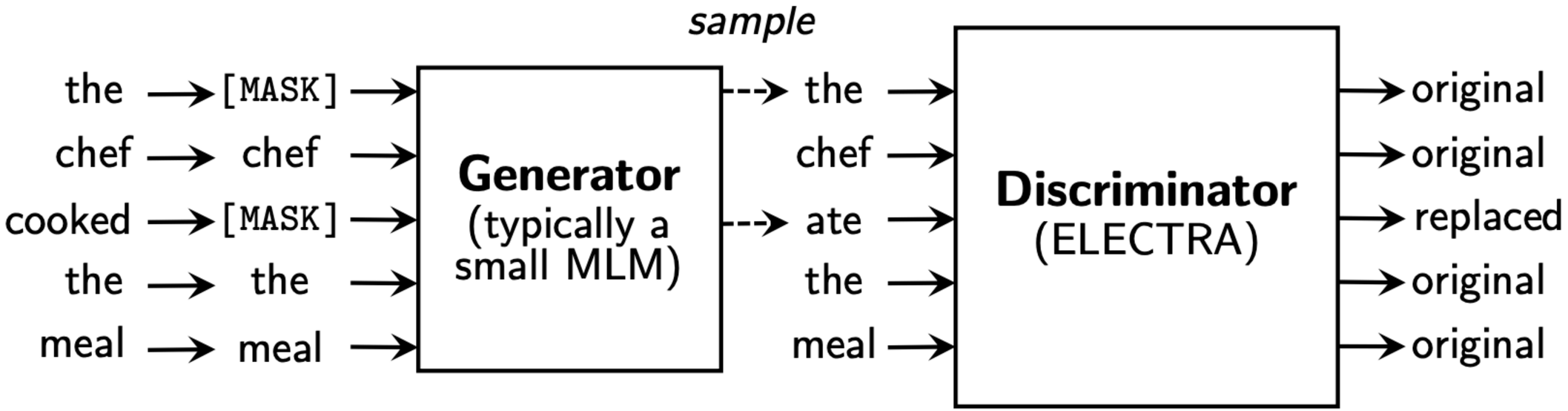
$$\mathcal{L}_{mlm} = -\log P_{dbert}\left(y_t^* \mid [\mathrm{M}], \{y_s^*\}_{s \neq t}\right)$$

$$\mathcal{L}_{tot} = \gamma_1 \mathcal{L}_{distil} + \gamma_1 \mathcal{L}_{mlm}$$

- Allows you to train much smaller DistilBERT with ~97% performance of BERT

Sanh et al. (2019)

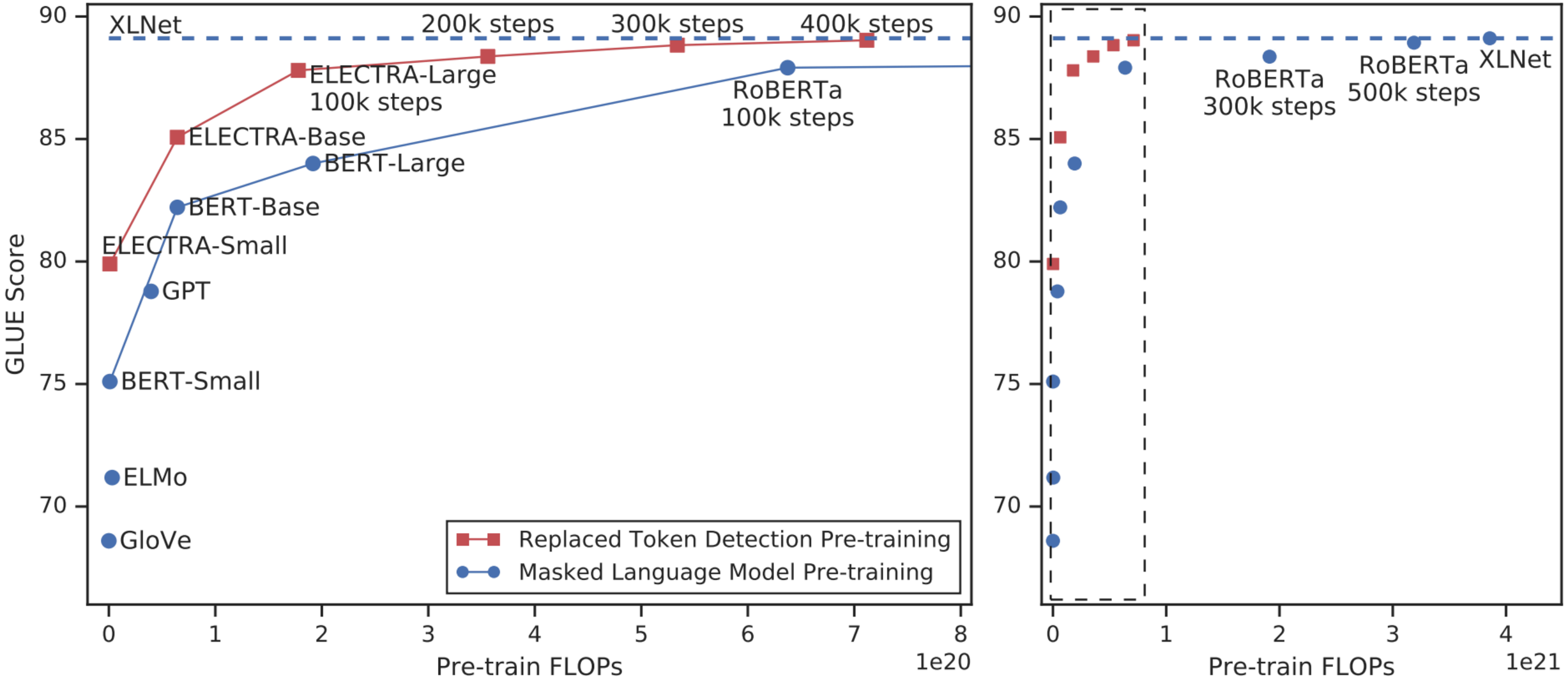
ELECTRA



- **Recall:** BERT only learns from 15% of masked tokens **(quite inefficient!)**
- Instead, predict whether a token is corrupted on the discriminator
- Learning from all tokens drastically speeds up training

Clark et al. (2019)

ELECTRA


90
85
80
75
70
GLUE Score
XLNet
200k steps
300k steps
400k steps
ELECTRA-Large
100k steps
ELECTRA-Base
BERT-Large
BERT-Base
ELECTRA-Small
GPT
BERT-Small
ELMo
GloVe
RoBERTa
100k steps
Replaced Token Detection Pre-training
Masked Language Model Pre-training
0
1
2
3
4
5
6
7
8
1e20
Pre-train FLOPs
RoBERTa
300k steps
RoBERTa
500k steps
XLNet
0
1
2
3
4
1e21
Pre-train FLOPs

Question

What was the main improvement of BERT-style models over GPT?

Bidirectionality allows for more expressive representations to be learned

Question

What can BERT NOT due as a result of its masked LM training objective?

Generate text!

Recap

- **Contextualised embeddings:** Let us model words and sequences conditioned on the context around them
- **ELMo:** One of the first models for contextualized embeddings based on bidirectional LSTMs good for pretrained embeddings
- **GPT**: First model for contextualised embeddings using transformer models good for generating text as a language model
- **BERT**: Improving transformer-based contextual representations using masked language models and bidirectional encoding good for classificat° + sequence labelling
- Many variants of BERT in recent years!
- Much work on analysing information learned in contextual representations (Week 13)

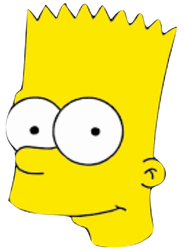
Pretrained Representations:
BART & T5

Antoine Bosselut

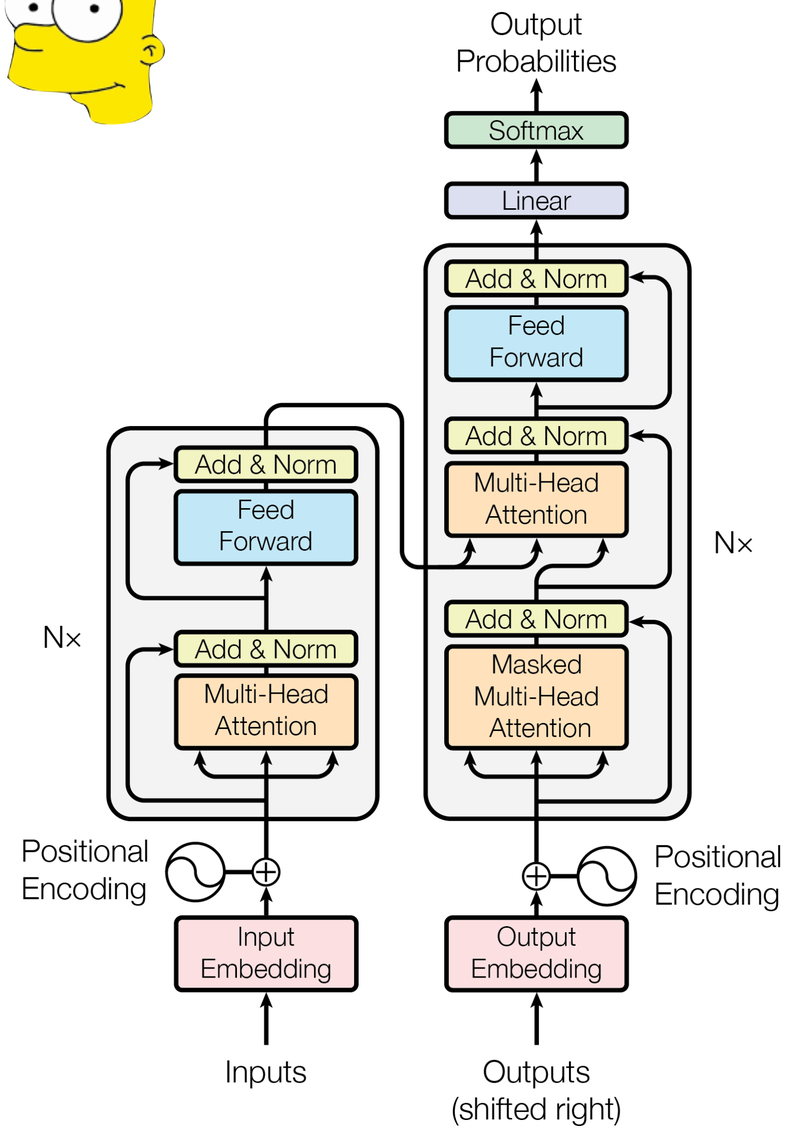
EPFL

How should we pretrain sequence-to-sequence models?

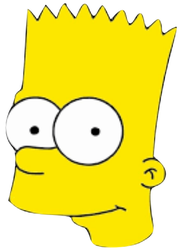
BART



- Classic transformer architecture
- Bidirectional encoder feeds into autoregressive decoder
- Cross-attention layers in decoder are back!
- **BART-base**: 6-layers each in encoder and decoder; 140M parameters
- **BART-large**: 12 layers each in encoder and decoder; 400M parameters

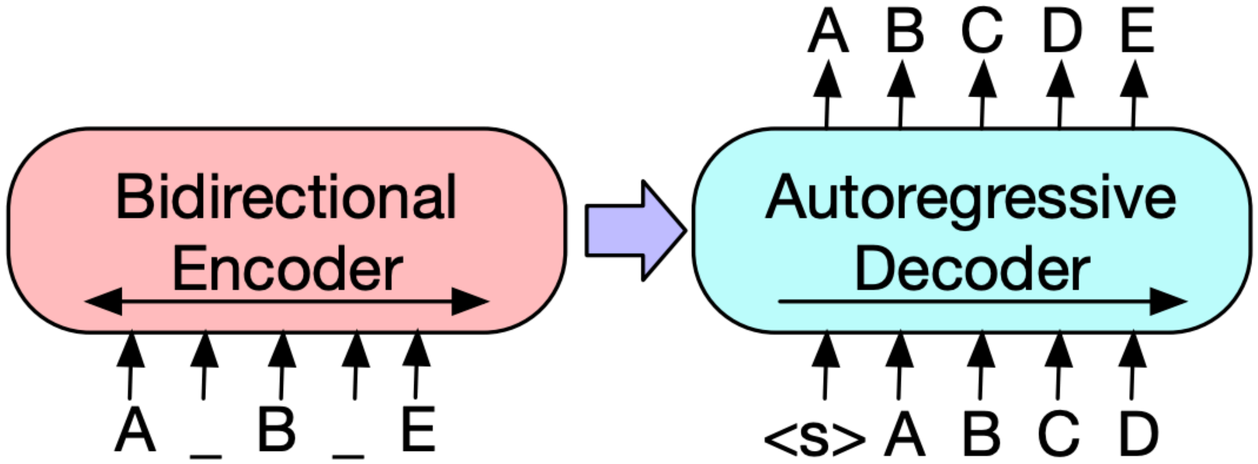


Lewis et al. (2019)

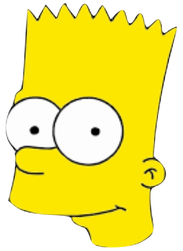


BART Pretraining

- Pretraining BART combines elements of BERT and GPT!
- **BERT-style:** input texts corrupted before they are passed to bidirectional encoder
- **GPT-style:** model is trained with a language modelling objective in the decoder: predict the next word!



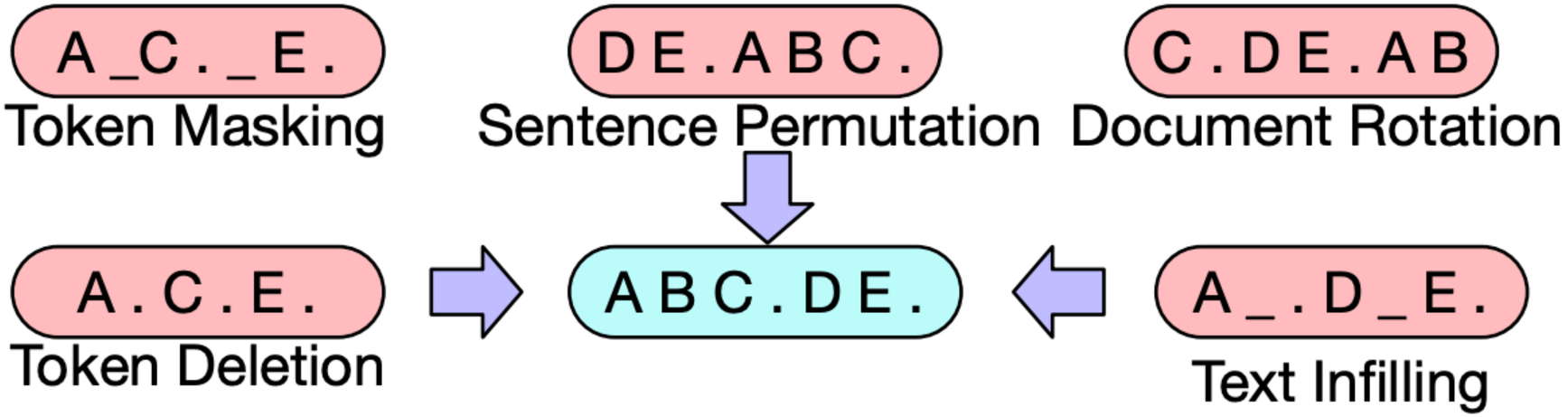
Lewis et al. (2019)



BART Pretraining

- We're not reconstructing the input the same way as BERT, so can we corrupt the input in different ways?

- Many corruption strategies can be used on the encoder side



Lewis et al. (2019)

Can do all the same tasks

Fine tuning = works well for text generat°, summarizat°, machine translat°, text complet°

- BART can also do all the tasks that BERT does!

- **Classification:**
  - Give input to both encoder AND decoder (input the sequence twice)
  - Append [CLS] token to decoder sequence and classify its output

- **Sequence Labeling:**
  - Give input to both encoder AND decoder (input the sequence twice)
  - Classify decoder output representations for each token

Can do all the same tasks

	SQuAD 1.1 EM/F1	**SQuAD 2.0** EM/F1	**MNLI** m/mm	**SST** Acc	**QQP** Acc	**QNLI** Acc	**STS-B** Acc	**RTE** Acc	**MRPC** Acc	**CoLA** Mcc
BERT	84.1/90.9	79.0/81.8	86.6/-	93.2	91.3	92.3	90.0	70.4	88.0	60.6
UniLM	-/-	80.5/83.4	87.0/85.9	94.5	-	92.7	-	70.9	-	61.1
XLNet	**89.0**/94.5	86.1/88.8	89.8/-	95.6	91.8	93.9	91.8	83.8	89.2	63.6
RoBERTa	88.9/**94.6**	**86.5/89.4**	**90.2/90.2**	96.4	92.2	94.7	**92.4**	86.6	**90.9**	**68.0**
BART	88.8/**94.6**	86.1/89.2	89.9/90.1	**96.6**	**92.5**	**94.9**	91.2	**87.0**	90.4	62.8

Almost as good as RoBERTa

Way better than BERT! Why ?

Trained on way more data!

Results: Summarization

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.
PG&E stated it scheduled the blackouts in response to forecasts for high winds amid dry conditions. The aim is to reduce the risk of wildfires. Nearly 800 thousand customers were scheduled to be affected by the shutoffs which were expected to last through at least midday tomorrow.	Power has been turned off to millions of customers in California as part of a power shutoff plan.

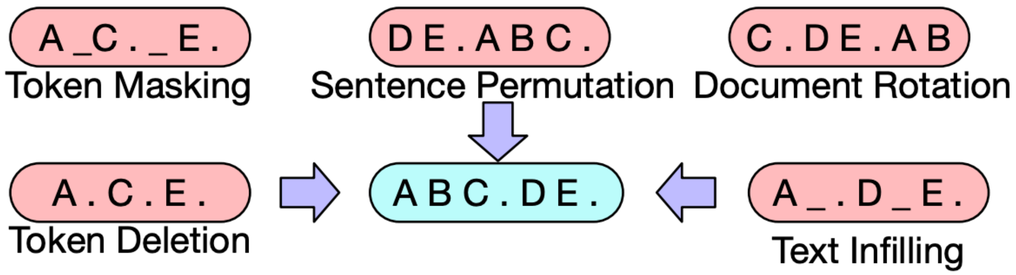
However, BART can do generation tasks too

Decoder is autoregressive!

Lewis et al. (2019)

Which encoder-side corruption?

Model	SQuAD 1.1 F1	MNLI Acc	ELI5 PPL	XSum PPL	ConvAI2 PPL	CNN/DM PPL
BART Base						
w/ Token Masking	90.4	84.1	25.05	7.08	11.73	6.10
w/ Token Deletion	90.4	84.1	24.61	6.90	11.46	5.87
w/ Text Infilling	**90.8**	84.0	24.26	**6.61**	**11.05**	5.83
w/ Document Rotation	77.2	75.3	53.69	17.14	19.87	10.59
w/ Sentence Shuffling	85.4	81.5	41.87	10.93	16.67	7.89
w/ Text Infilling + Sentence Shuffling	**90.8**	83.8	24.17	6.62	11.12	**5.41**

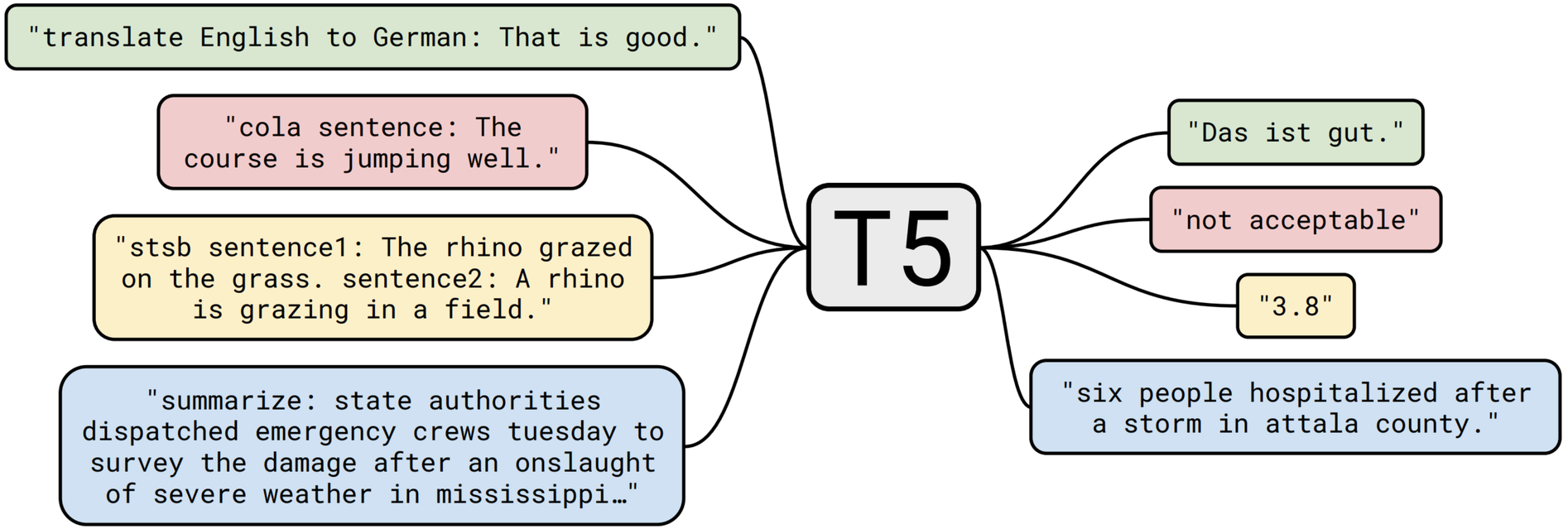


- Different corruption better for transfer to different tasks
- **Use combination of text infilling + sentence permutation**

T5

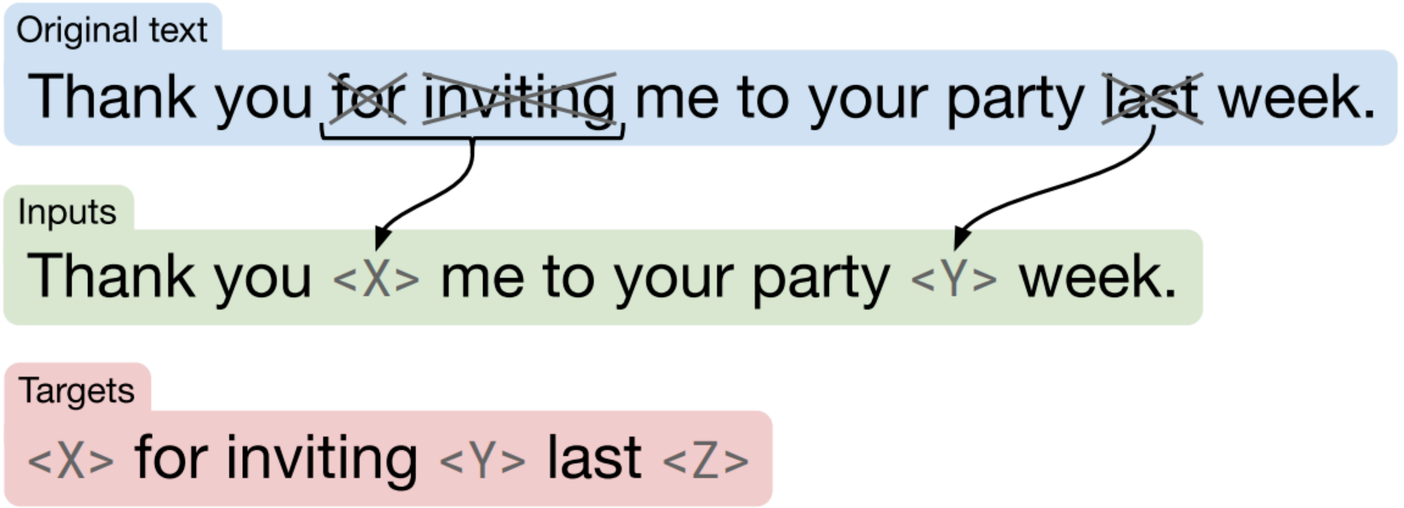
Finetuning = all tasks are converted into text generat° problems

- **Similar idea as BART:** Any problem can be cast as sequence-to-sequence



Raffel et al. (2019)

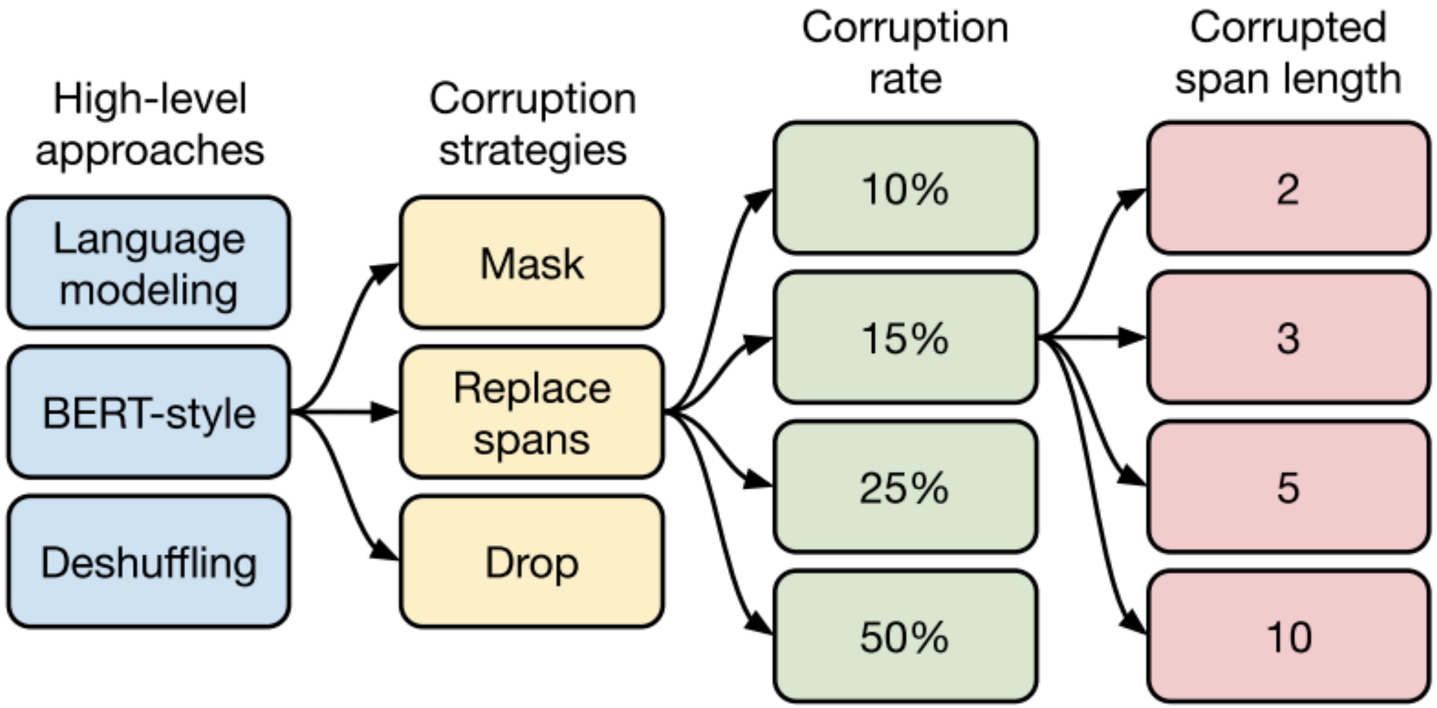
T5 Pretraining



- Similar to BART
- Uses the infilling objective where tokens are reconstructed from underspecified mask corruptions

Raffel et al. (2019)

T5 Pretraining Decisions



- Explored many dimensions of pretraining in se2seq framework
- Took findings to train much larger model — 11B parameters!

Raffel et al. (2019)

Recap

- **Contextual representations:** Let us model words and sequences conditioned on the context around them
- **ELMo:** Based on bidirectional LSTMs. **Good for pretrained embeddings.**
- **GPT**: Uses a transformer decoder. **Good for generating text as a language model.**
- **BERT**: Uses a transformer encoder. **Good for classification and sequence labelling.**
- **BART + T5:** Pretraining sequence-to-sequence transformer models. **Extendable to all task types!**

References

- Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training.
- Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, *1*(8), 9.
- Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep Contextualized Word Representations. *North American Chapter of the Association for Computational Linguistics*.
- Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.
- Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.
- Clark, K., Luong, M. T., Le, Q. V., & Manning, C. D. (2020). Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.
- Clark, K., Khandelwal, U., Levy, O., & Manning, C. D. (2019). What does bert look at? an analysis of bert's attention. *arXiv preprint arXiv:1906.04341*.