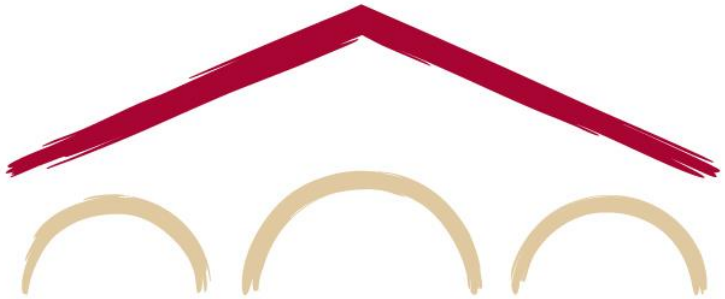
Natural Language Processing with Deep Learning
CS224N/Ling284

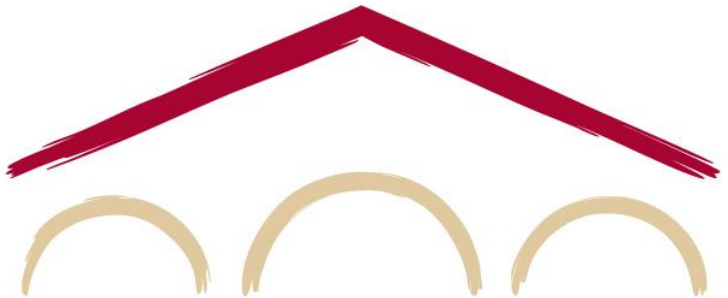


Diyi Yang

Lecture 6: Pretraining

自然语言处理与深度学习

CS224N/Ling284



Diyi Yang

第 6 讲：预训练

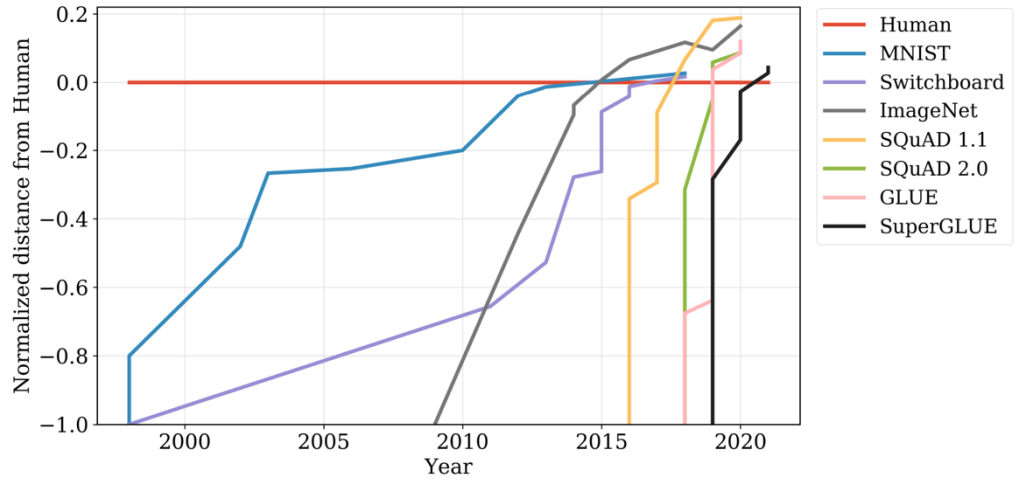
Lecture Plan

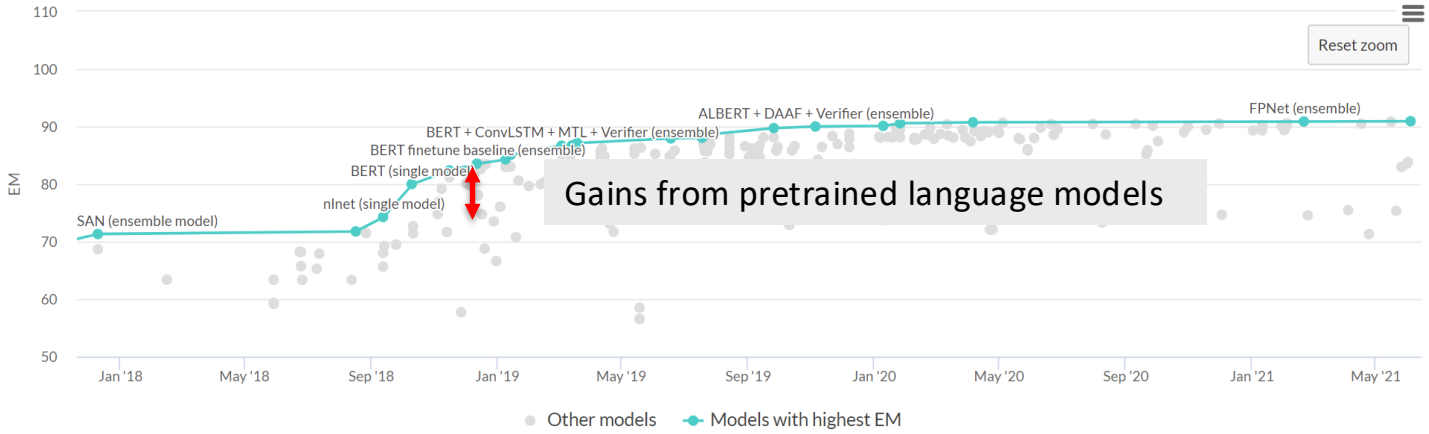
1. Pretraining – motivation (10 mins)
2. Subword modeling (10 mins)
3. Motivating model pretraining from word embeddings (10 mins)
4. Model pretraining three ways (25 mins)
   - Decoders
   - Encoders
   - Encoder-Decoders
5. Interlude: what do we think pretraining is teaching? (10 mins)
6. Very large models and in-context learning (10 mins)

课程计划

1. 预训练 —— 动机（１０分钟）
2. 子词建模（１０分钟）
3. 从词 ｅｍｂｅｄｄｉｎｇ 出发理解模型预训练的动机（１０分钟）
4. Model pretraining three ways (25 mins)
   - Ｄｅｃｏｄｅｒ 类
   - Ｅｎｃｏｄｅｒ 类
   - Ｅｎｃｏｄｅｒ-Ｄｅｃｏｄｅｒ 类
5. 插曲：我们认为预训练在教什么？（１０分钟）
6. 非常大的模型和 ｉｎ-ｃｏｎｔｅｘｔ ｌｅａｒｎｉｎｇ（１０分钟）

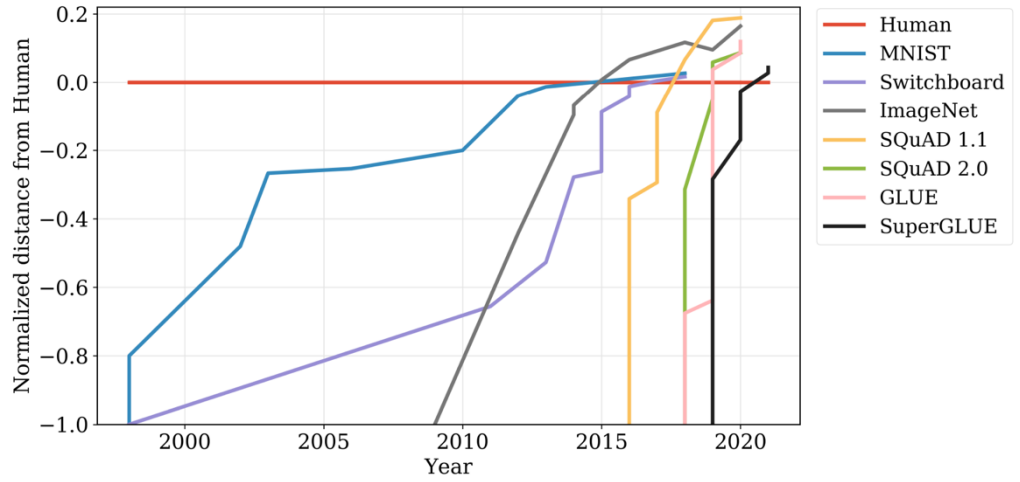
The pretraining revolution

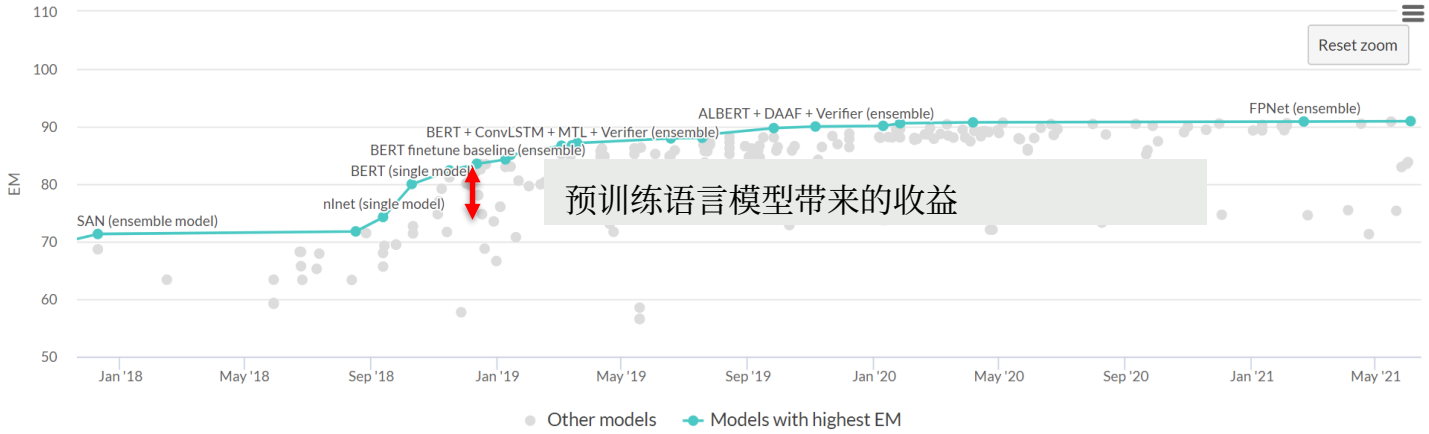




Pretraining has had a major, tangible impact on how well NLP systems work

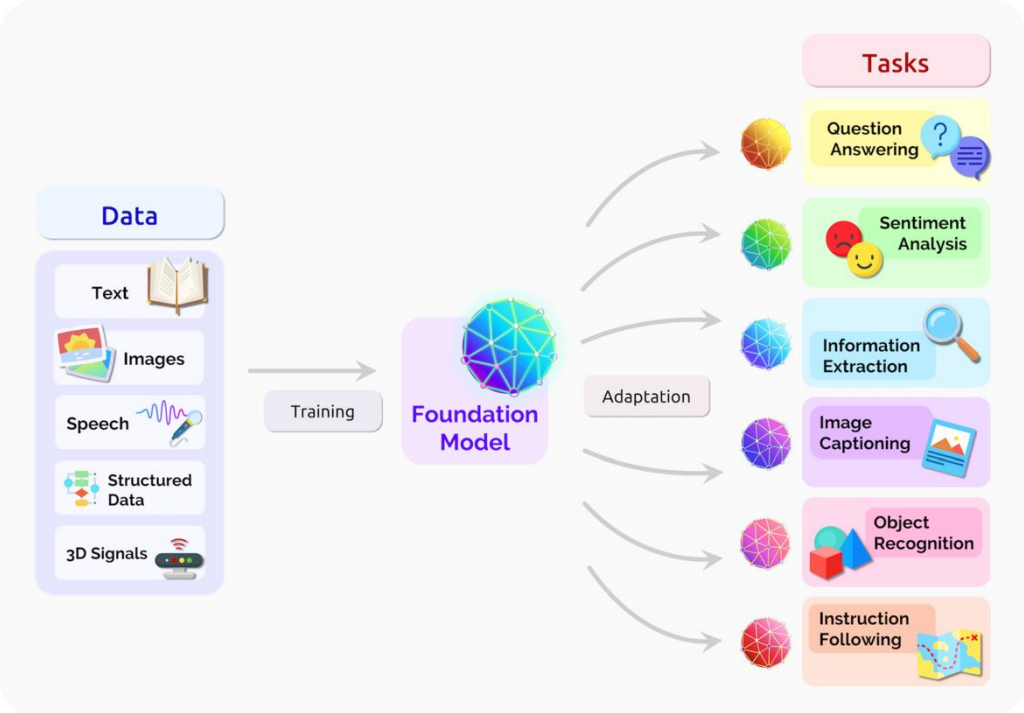
预训练革命





预训练对　ＮＬＰ　系统的效果产生了重大而切实的影响

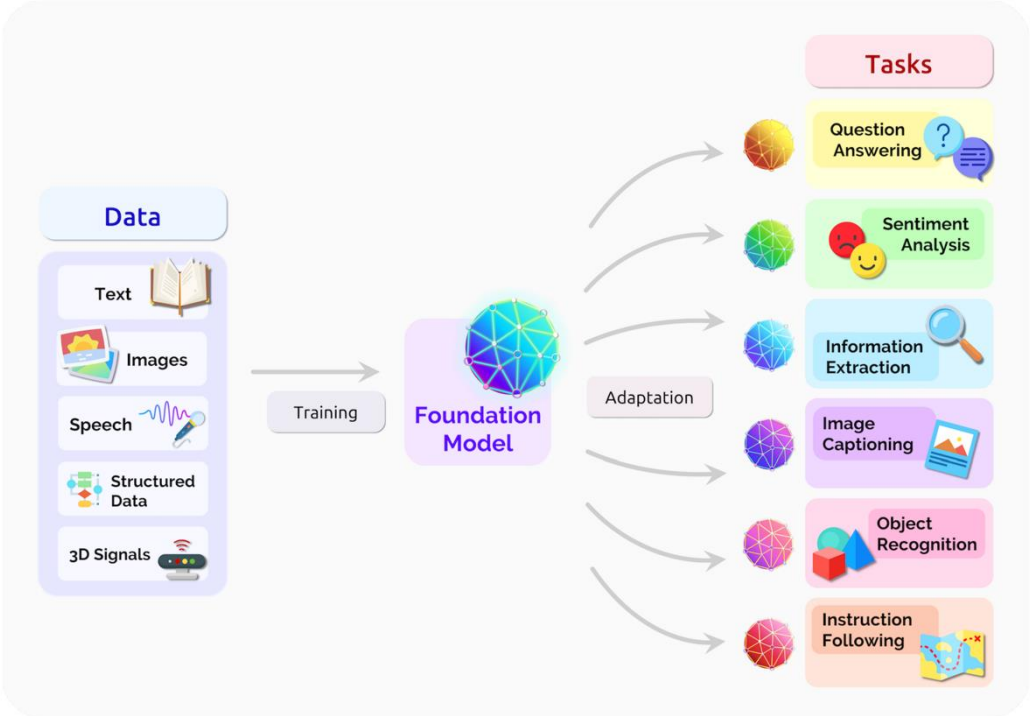
Pretraining – scaling unsupervised learning on the internet



Key ideas in pretraining

- Make sure your model can process large-scale, diverse datasets
- Don’t use labeled data (otherwise you can’t scale!)
- Compute-aware scaling

预训练 —— 在互联网上扩展无监督学习



预训练的关键思想

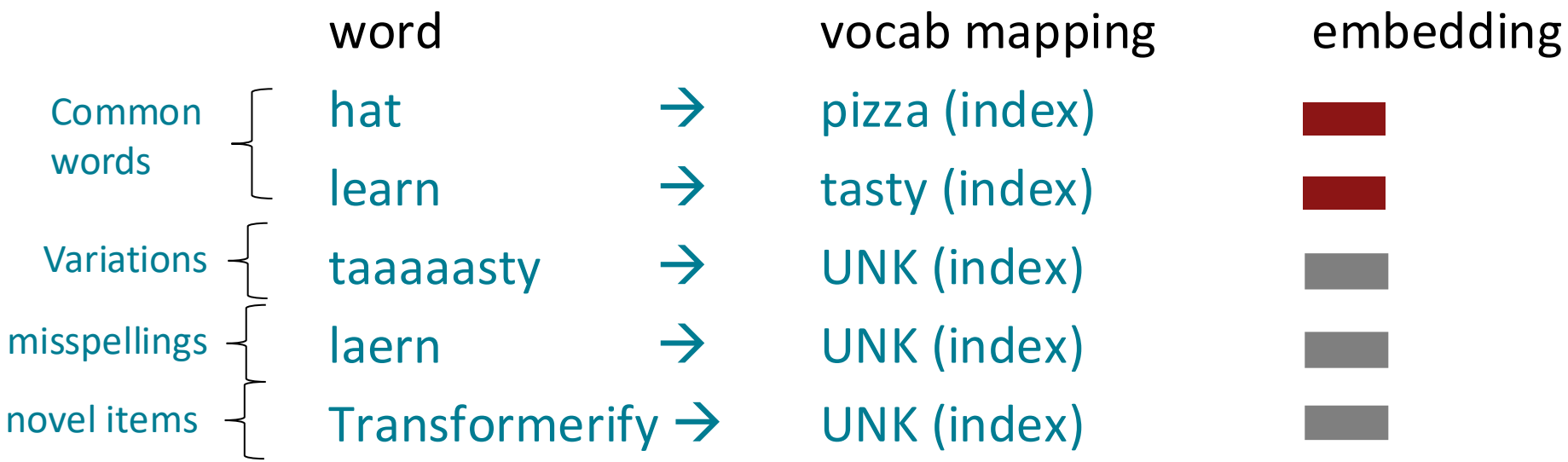
- 确保你的模型能处理大规模、多样化的数据集
- Don’t use labeled data (otherwise you can’t scale!)
- Compute-aware scaling

Word structure and subword models

Let's take a look at the assumptions we've made about a language's vocabulary.

We assume a fixed vocab of tens of thousands of words, built from the training set.

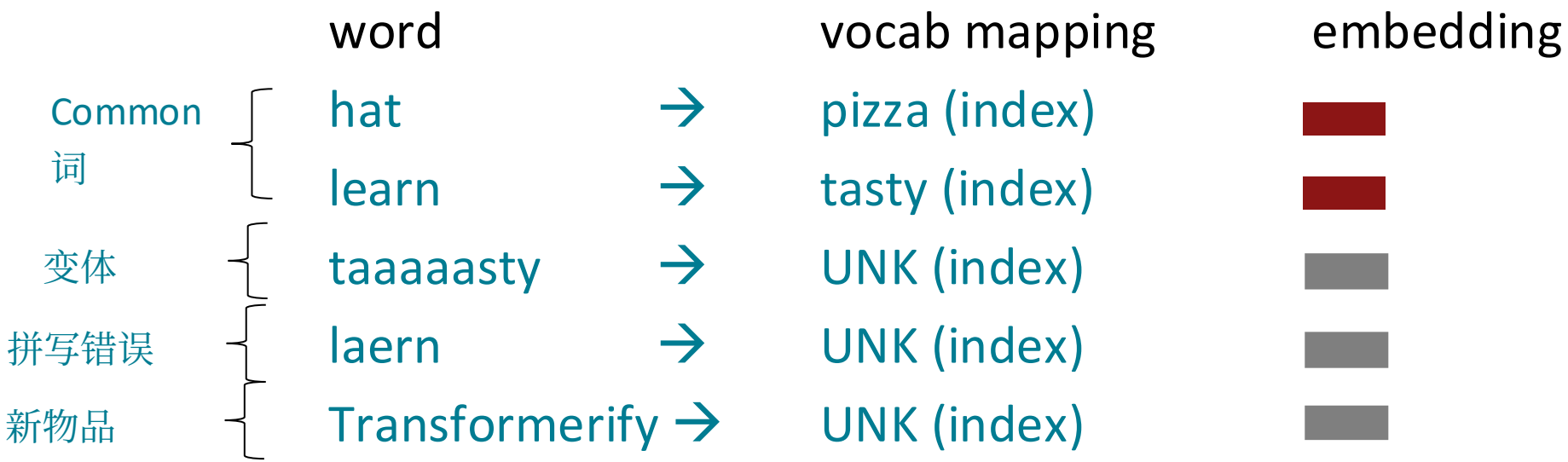
All *novel* words seen at test time are mapped to a single UNK.



词结构和子词模型

Let’s take a look at the assumptions we’ve made about a language’s vocabulary.

我们假设一个从训练集构建的、包含数万词的固定词汇表。
All 新的 words seen at test time are mapped to a single UNK.



The byte-pair encoding algorithm

Subword modeling in NLP encompasses a wide range of methods for reasoning about structure below the word level. (Parts of words, characters, bytes.)

- The dominant modern paradigm is to learn a vocabulary of **parts of words (subword tokens).**
- At training and testing time, each word is split into a sequence of known subwords.

Byte-pair encoding is a simple, effective strategy for defining a subword vocabulary.

1. Start with a vocabulary containing only characters and an "end-of-word" symbol.
2. Using a corpus of text, find the most common adjacent characters "a,b"; add "ab" as a subword.
3. Replace instances of the character pair with the new subword; repeat until desired vocab size.

Originally used in NLP for machine translation; now a similar method (WordPiece) is used in pretrained models.

[Sennrich et al., 2016, Wu et al., 2016]

字节对编码算法

Subword modeling in NLP encompasses a wide range of methods for reasoning about 词级以下的结构。（词的部分、字符、字节。）

- The dominant modern paradigm is to learn a vocabulary of 词的部分（子词 ｔｏｋｅｎ）。
- 在训练和测试时，每个词被分割成已知子词的序列。

字节对编码 is a simple, effective strategy for defining a subword vocabulary.

1. Start with a vocabulary containing only characters and an “end-of-word” symbol.
2. Using a corpus of text, find the most common adjacent characters “a,b”; add “ab” as a subword.
3. 用新的子词替换字符对的实例；重复直到达到所需的词汇表大小。

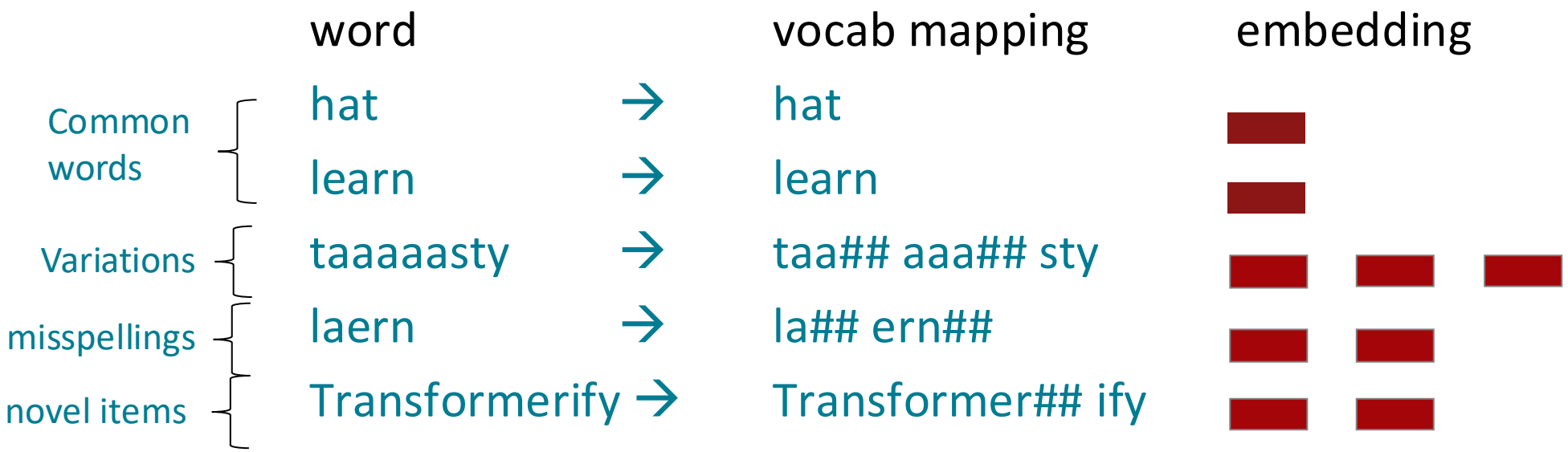
Originally used in NLP for machine translation; now a similar method (WordPiece) is used in pretrained 模型。

[Sennrich et al., 2016, Wu et al., 2016]

Word structure and subword models

Common words end up being a part of the subword vocabulary, while rarer words are split into (sometimes intuitive, sometimes not) components.

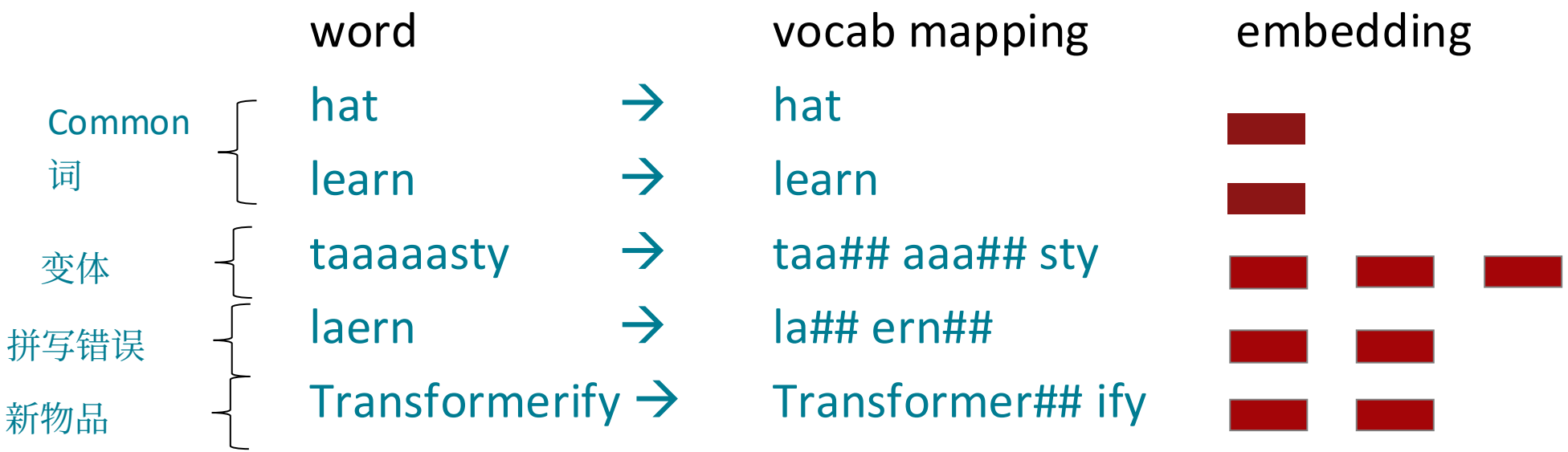
In the worst case, words are split into as many subwords as they have characters.



词结构和子词模型

Common words end up being a part of the subword vocabulary, while rarer words are split 成（有时直观、有时不直观的）组件。

在最坏的情况下，词被分割成与其字符数一样多的子词。



Outline

1. A brief note on subword modeling
2. Motivating model pretraining from word embeddings
3. Model pretraining three ways
    1. Encoders
    2. Encoder-Decoders
    3. Decoders
4. What do we think pretraining is teaching?

大纲

1. 关于子词建模的简要说明
2. 从词 embedding 出发理解模型预训练的动机
3. 三种模型预训练方式
    1. Encoder 类
    2. Encoder-Decoder 类
    3. Decoder 类
4. 我们认为预训练在教什么？

Motivating word meaning and context

Recall the adage we mentioned at the beginning of the course:

> *"You shall know a word by the company it keeps"* (J. R. Firth 1957: 11)

This quote is a summary of **distributional semantics**, and motivated **word2vec**. But:

> "… the complete meaning of a word is always contextual,
> and no study of meaning apart from a complete context
> can be taken seriously." (J. R. Firth 1935)

Consider *I* ***record*** *the* ***record***: the two instances of ***record*** mean different things.

[Thanks to Yoav Goldberg on Twitter for pointing out the 1935 Firth quote.]

理解词义和上下文的动机

回忆我们在课程开始时提到的格言：

> *"You shall know a word by the company it keeps"* (J. R. Firth 1957: 11)

This quote is a summary of 分布语义学 , and motivated **word2vec**。但是：

> "… the complete meaning of a word is always contextual,
> 脱离完整上下文就无法研究意义
> can be taken seriously." (J. R. Firth 1935)

Consider *I* 记录 *the* 记录 : the two instances of 记录 mean different things.

[Thanks to Yoav Goldberg on Twitter for pointing out the 1935 Firth quote.]

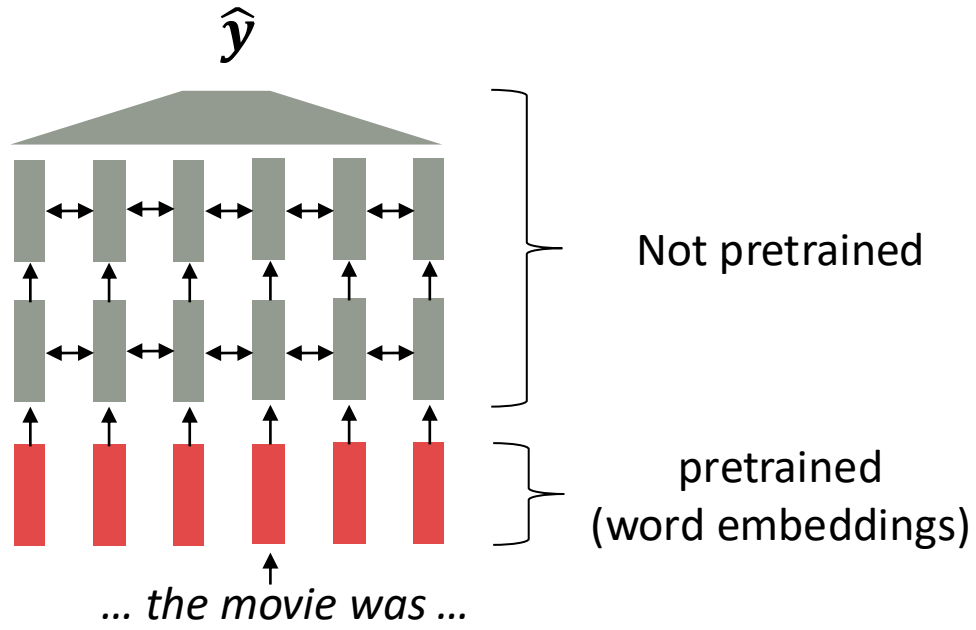
Where we were: **pretrained word embeddings**

Circa 2017:

- Start with pretrained word embeddings (no context!)
- Learn how to incorporate context in an LSTM or Transformer while training on the task.

Some issues to think about:

- The training data we have for our **downstream task** (like question answering) must be sufficient to teach all contextual aspects of language.
- Most of the parameters in our network are randomly initialized!



[Recall, *movie* gets the same word embedding, no matter what sentence it shows up in]

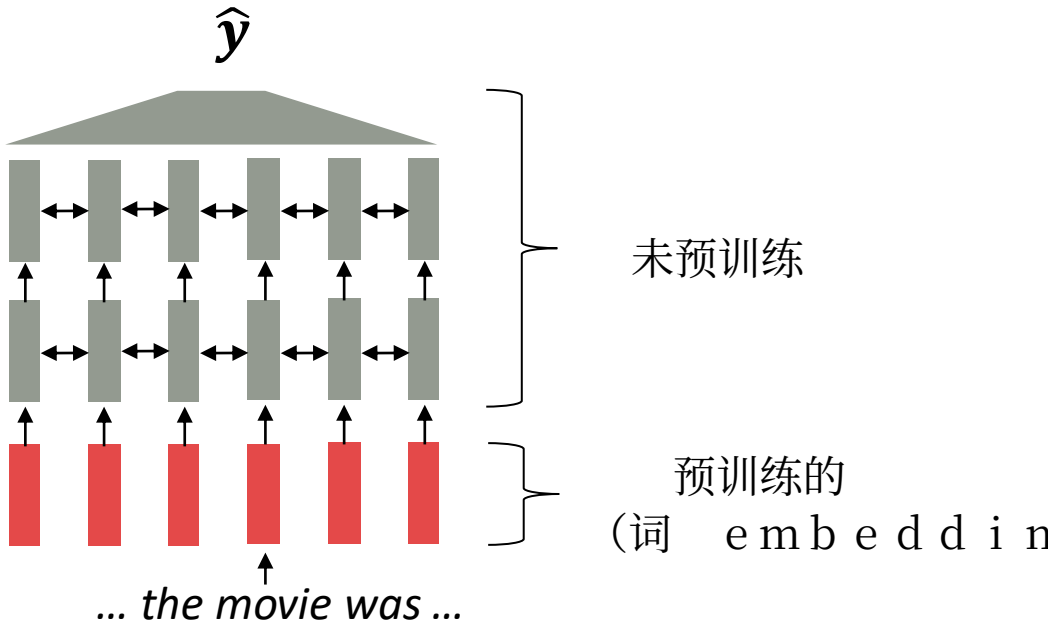
Where we were: 预训练词 embeddings

大约2017年：

- Start with pretrained word embeddings (no 上下文！)
- Learn how to incorporate context in an LSTM 或 Transformer，同时在任务上训练。

一些需要思考的问题：

- The training data we have for our 下游任务 (like question answering) must be sufficient to teach all contextual 语言的各个方面。
- Most of the parameters in our network are 随机初始化！



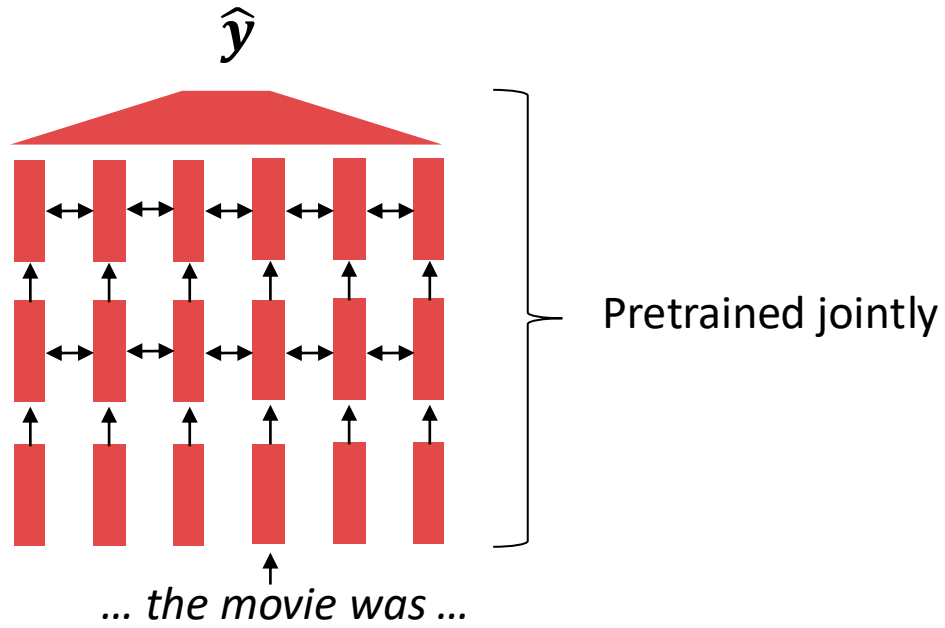
[Recall, *movie* gets the same word embedding, 无论它出现在什么句子中]

Where we're going: **pretraining whole models**

In modern NLP:

- All (or almost all) parameters in NLP networks are initialized via **pretraining**.
- Pretraining methods hide parts of the input from the model, and train the model to reconstruct those parts.

- This has been exceptionally effective at building strong:
  - **representations of language**
  - **parameter initializations** for strong NLP models.
  - **Probability distributions** over language that we can sample from



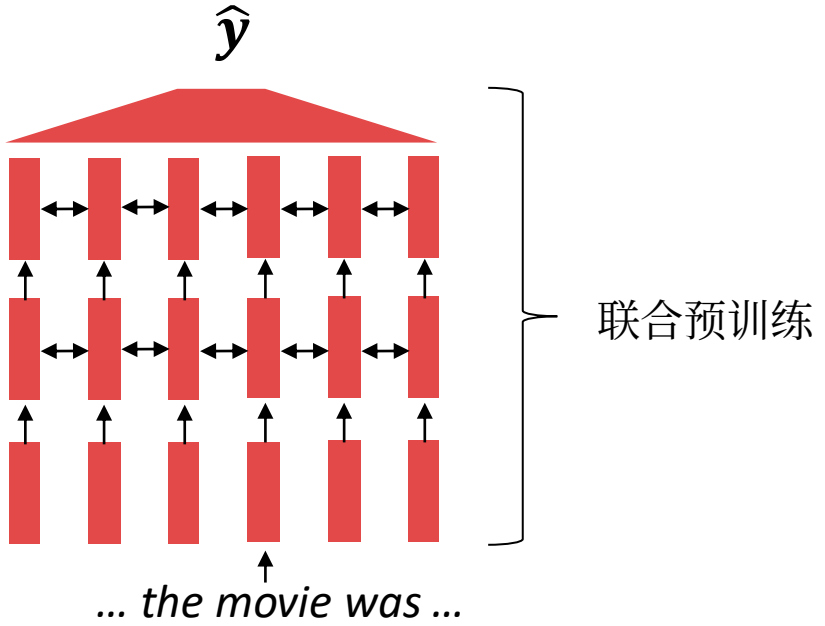
[This model has learned how to represent entire sentences through pretraining]

Where we're going: 预训练整个模型

在现代 NLP 中：

- All (or almost all) parameters in NLP networks are initialized via 预训练 .
- Pretraining methods hide parts of the input from the model, and train the model to 重建那些部分。

- This has been exceptionally effective at building strong:
  - 语言的表示
  - 参数初始化 for strong NLP 模型。
  - 概率分布 over language that 我们可以从中采样



[This model has learned how to represent 整个句子通过预训练]

What can we learn from reconstructing the input?

Stanford University is located in _____, California.

我们能从重建输入中学到什么？

Stanford University is located in _____, California.

What can we learn from reconstructing the input?

I put ___ fork down on the table.

我们能从重建输入中学到什么？

I put ___ fork down on the table.

What can we learn from reconstructing the input?

The woman walked across the street,
checking for traffic over ___ shoulder.

我们能从重建输入中学到什么？

The woman walked across the street,
checking for traffic over ___ shoulder.

What can we learn from reconstructing the input?

I went to the ocean to see the fish, turtles, seals, and _____.

我们能从重建输入中学到什么？

I went to the ocean to see the fish, turtles, seals, and _____.

What can we learn from reconstructing the input?

Overall, the value I got from the two hours watching
it was the sum total of the popcorn and the drink.
The movie was ___.

我们能从重建输入中学到什么？

总的来说，我花两个小时观看获得的价值

it was the sum total of the popcorn and the drink.

The movie was ___.

What can we learn from reconstructing the input?

I was thinking about the sequence that goes

1, 1, 2, 3, 5, 8, 13, 21, ____

我们能从重建输入中学到什么？

我在想那个序列

1, 1, 2, 3, 5, 8, 13, 21, ____

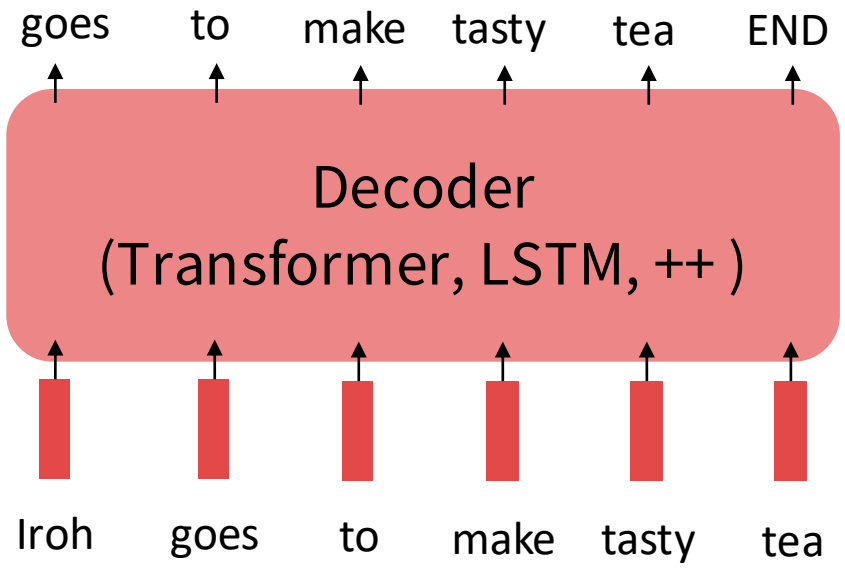
Pretraining through language modeling [Dai and Le, 2015]

Recall the **language modeling** task:

- Model $p_\theta(w_t|w_{1:t-1})$, the probability distribution over words given their past contexts.
- There's lots of data for this! (In English.)

Pretraining through language modeling:

- Train a neural network to perform language modeling on a large amount of text.
- Save the network parameters.



Pretraining through language modeling [Dai and Le, 2015]

Recall the **language modeling** task:

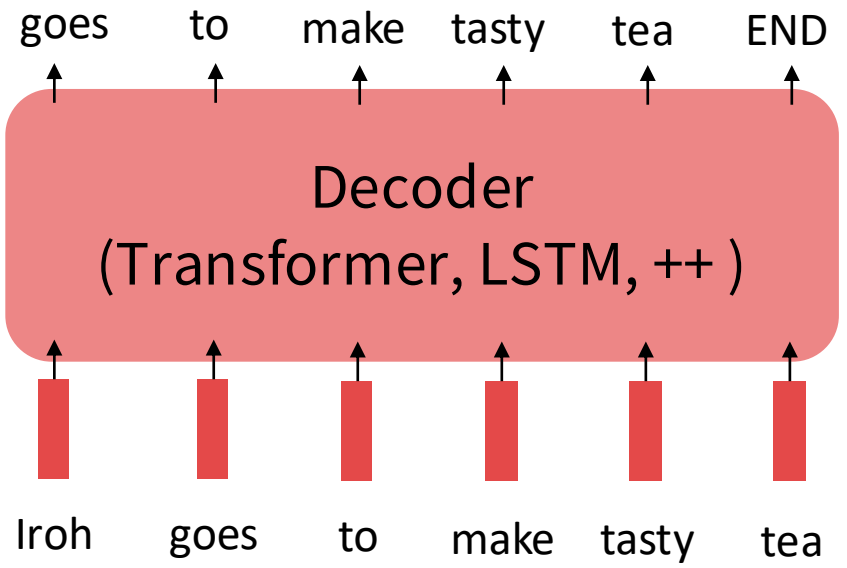
- Model $p_\theta(w_t|w_{1:t-1})$, the probability distribution over words given their past

  上下文。
- There's lots of data for this! (In English.)

通过语言建模进行预训练：

- Train a neural network to perform language

  在大量文本上建模。
- 保存网络参数。

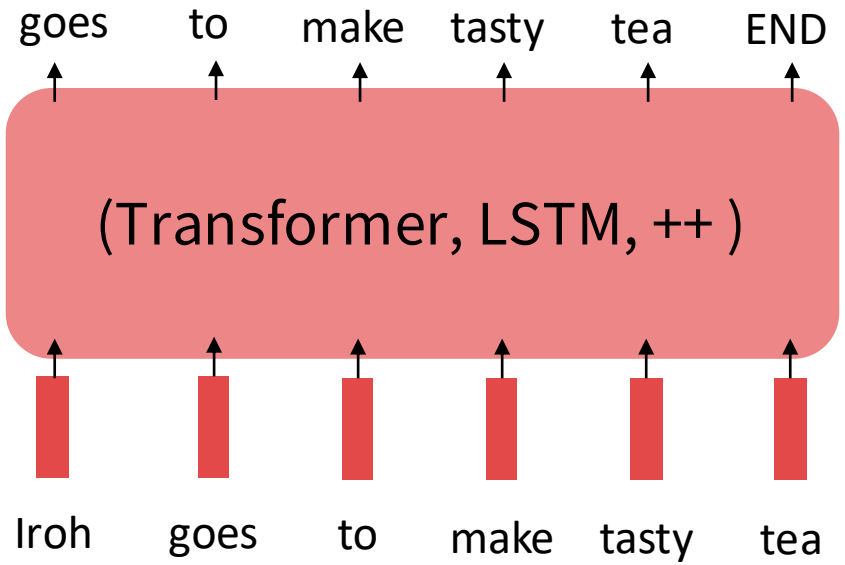


The Pretraining / Finetuning Paradigm

Pretraining can improve NLP applications by serving as parameter initialization.

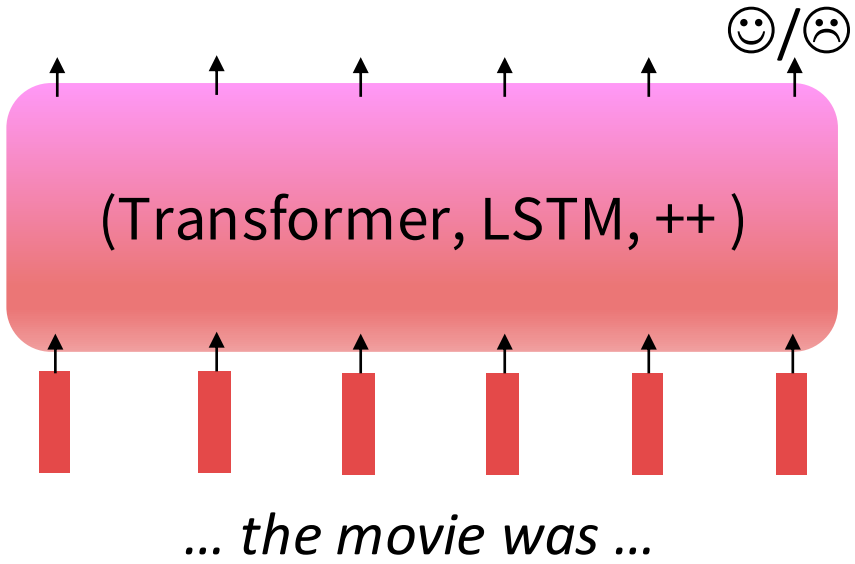
Step 1: Pretrain (on language modeling)

Lots of text; learn general things!



Step 2: Finetune (on your task)

Not many labels; adapt to the task!

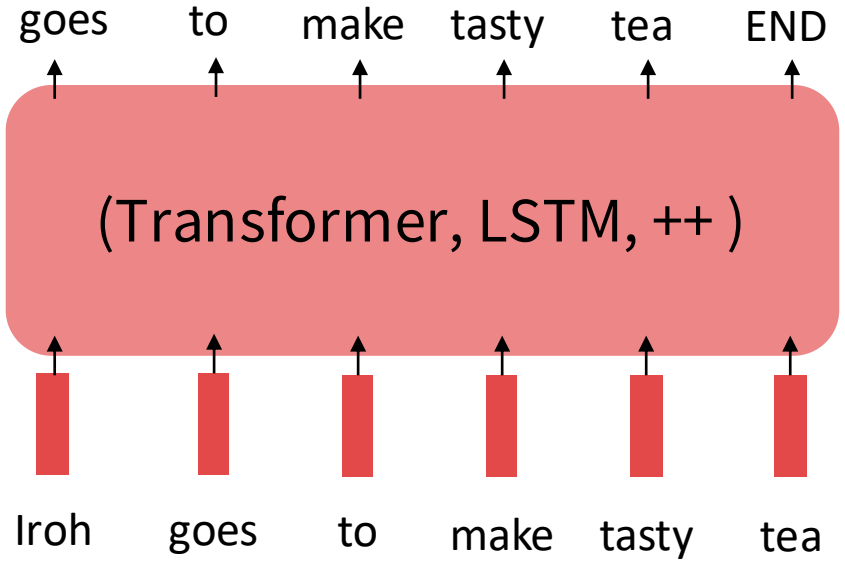


预训练 / Ｆｉｎｅ - ｔｕｎｉｎｇ　范式

预训练可以通过作为参数初始化来改善　ＮＬＰ　应用。

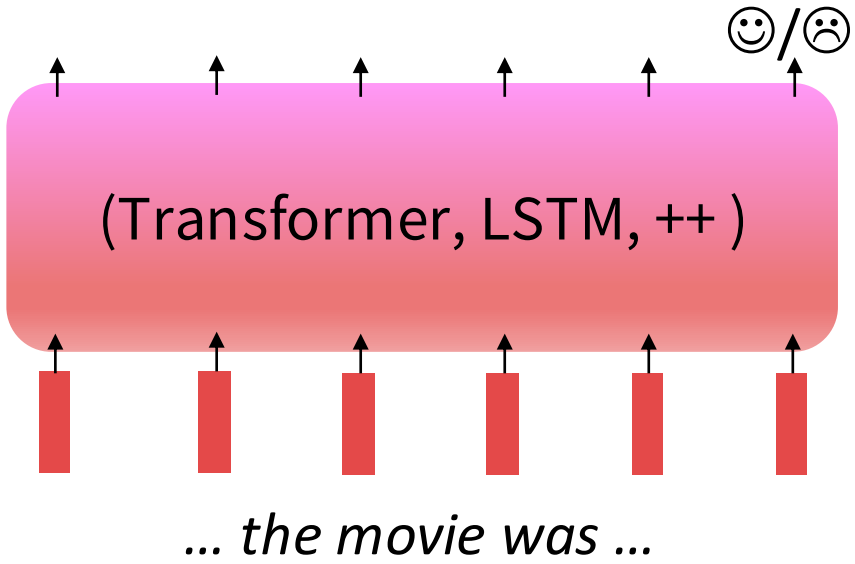
第１步：预训练（语言建模）

大量文本；学习通用知识！

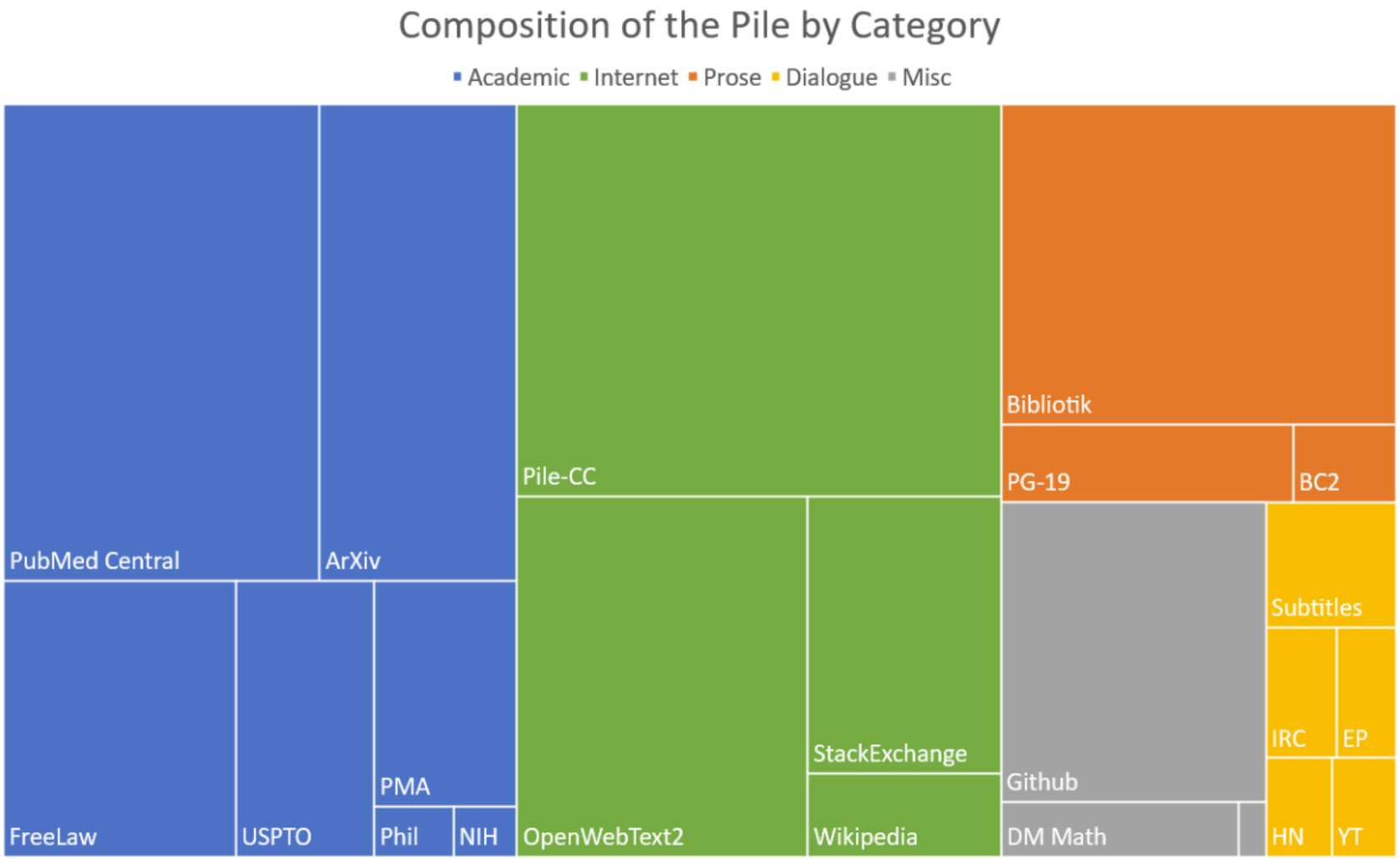


第２步：Ｆｉｎｅ - ｔｕｎｅ（在你的任

标注不多；适应任务！

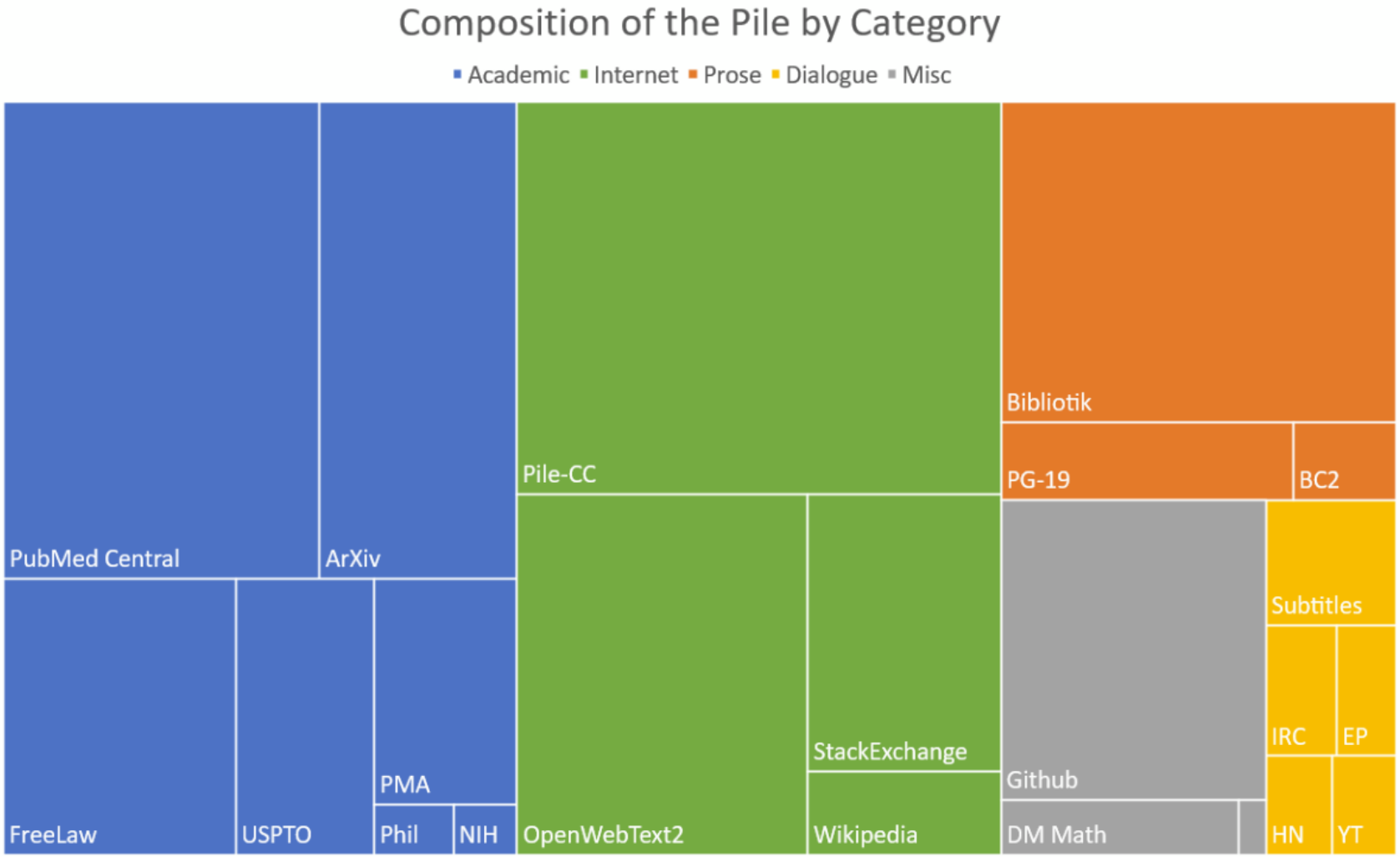


Where does this data come from?



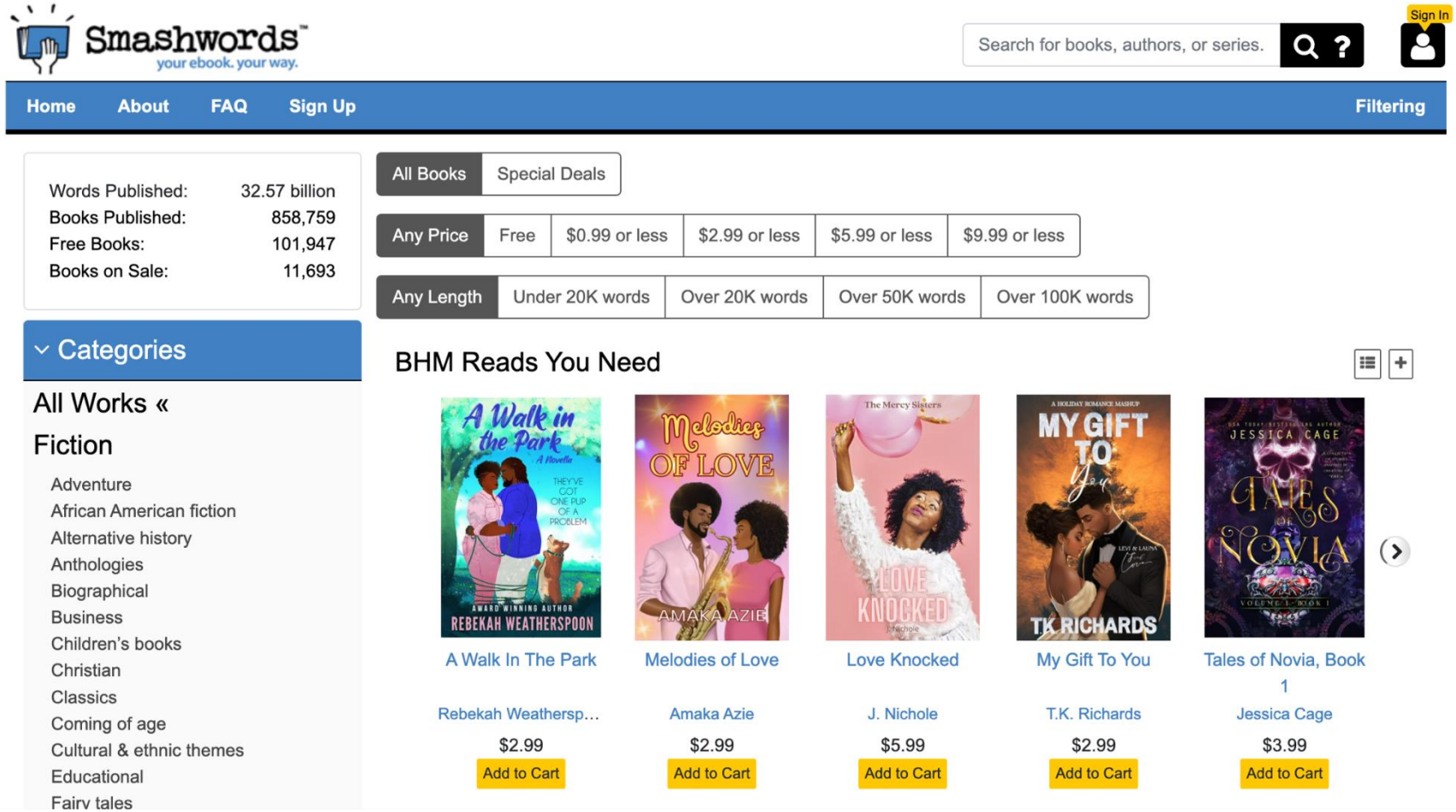
Model	Training Data
BERT	BookCorpus, English Wikipedia
GPT-1	BookCorpus
GPT-3	CommonCrawl, WebText, English Wikipedia, and 2 book databases (“Books 1” and “Books 2”)
GPT-3.5+	Undisclosed

这些数据来自哪里？



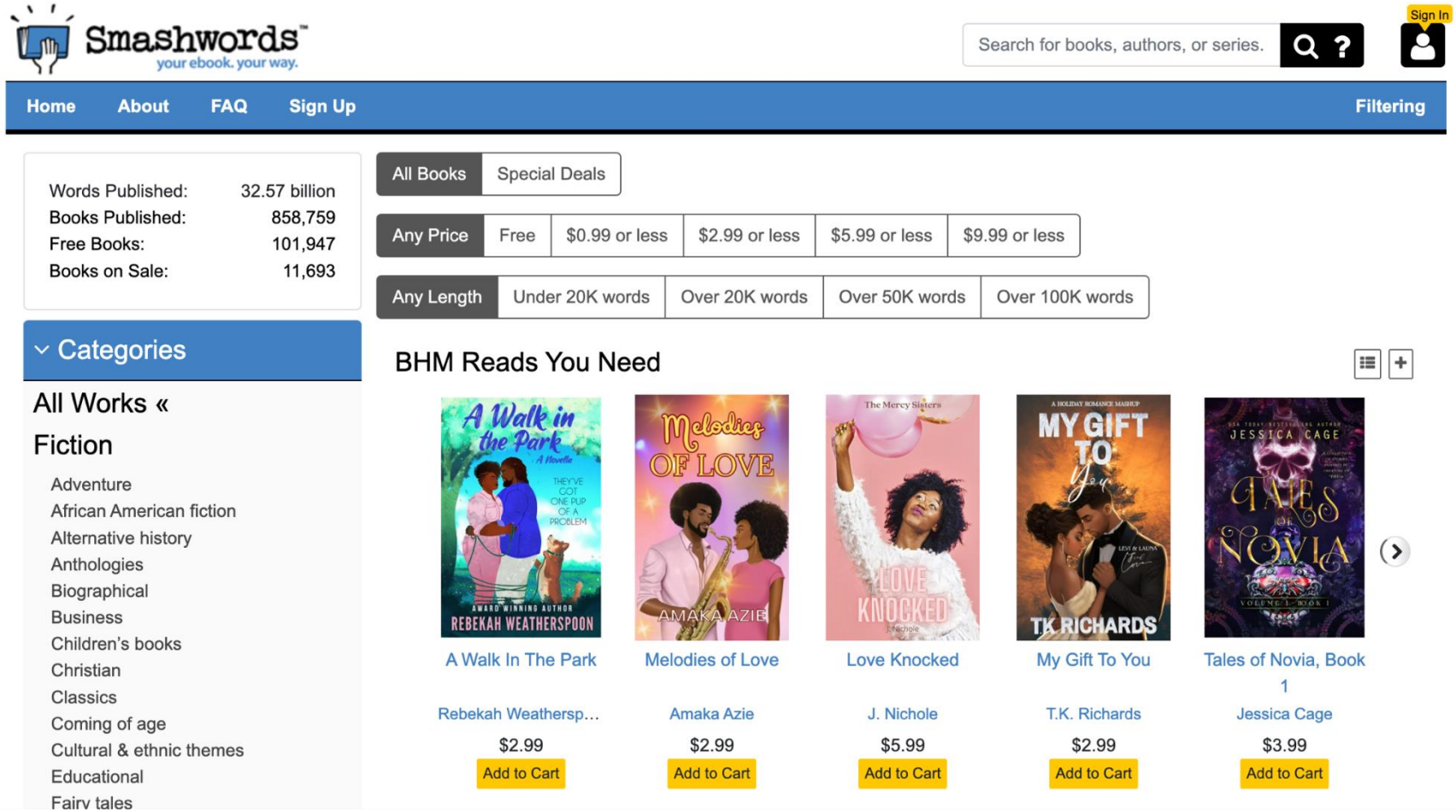
模型	训练数据
BERT	BookCorpus, English Wikipedia
GPT-1	BookCorpus
GPT-3	CommonCrawl, WebText, English Wikipedia, and 2 book databases (“Books 1” and “Books 2”)
GPT-3.5+	未公开

Bookcorpus.. what's that?



- Scraped ebooks from the internet – highly controversial

Bookcorpus.. what's that?



- 从互联网上抓取的电子书 —— 极具争议

Fair use and other concerns

Google swallows 11,000 novels to improve AI's conversation

As writers learn that tech giant has processed their work without permission, the Authors Guild condemns 'blatantly commercial use of expressive authorship'

'It doesn't harm the authors' ... Google's headquarters in Mountain View, California. Photograph: Marcio Jose Sanchez/AP

Arts and Humanities, Law, Regulation, and Policy, Machine Learning

Reexamining "Fair Use" in the Age of AI

Generative AI claims to produce new language and images, but when those ideas are based on copyrighted material, who gets the credit? A new paper from Stanford University looks for answers.

Jun 5, 2023 | Andrew Myers

合理使用和其他关注点

Google swallows 11,000 novels to improve AI's conversation

As writers learn that tech giant has processed their work without permission, the Authors Guild condemns 'blatantly commercial use of expressive authorship'

'It doesn't harm the authors' ... Google's headquarters in Mountain View, California. Photograph: Marcio Jose Sanchez/AP

Arts and Humanities, Law, Regulation, and Policy, Machine Learning

Reexamining "Fair Use" in the Age of AI

Generative AI claims to produce new language and images, but when those ideas are based on copyrighted material, who gets the credit? A new paper from Stanford University looks for answers.

Jun 5, 2023 | Andrew Myers

Lecture Plan

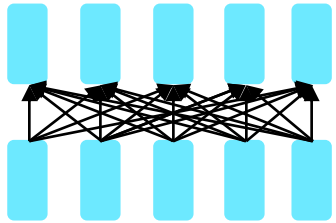
1. A brief note on subword modeling
2. Motivating model pretraining from word embeddings
3. Model pretraining three ways
    1. Encoders
    2. Encoder-Decoders
    3. Decoders
4. What do we think pretraining is teaching?

课程计划

1. 关于子词建模的简要说明
2. 从词 embedding 出发理解模型预训练的动机
3. 三种模型预训练方式
    1. Encoder 类
    2. Encoder-Decoder 类
    3. Decoder 类
4. 我们认为预训练在教什么？

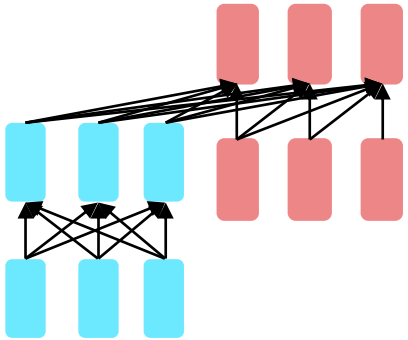
Pretraining for three types of architectures

The neural architecture influences the type of pretraining, and natural use cases.



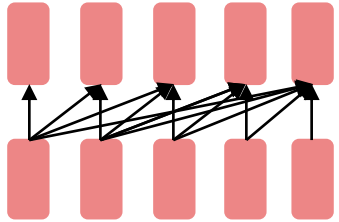
Encoders

- Gets bidirectional context – can condition on future!
- How do we train them to build strong representations?



Encoder-Decoders

- Good parts of decoders and encoders?
- What's the best way to pretrain them?

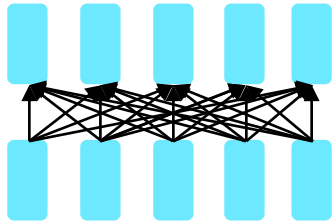


Decoders

- Language models! What we've seen so far.
- Nice to generate from; can't condition on future words

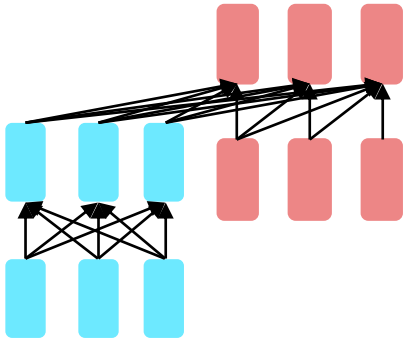
三种架构类型的预训练

神经架构影响预训练的类型和自然的应用场景。



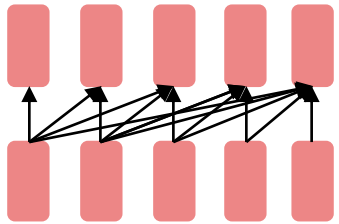
Encoder 类

- 获得双向上下文 —— 可以依赖未来！
- 我们如何训练它们以建立强大的表示？



Encoder-Decoder 类

- Decoder 和 Encoder 的优点？
- What's the best way to pretrain them?

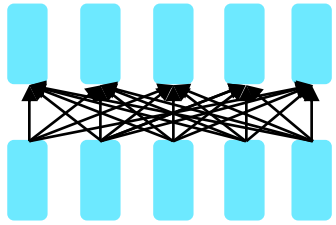


Decoder 类

- Language models! What we've seen so far.
- Nice to generate from; can't condition on future words

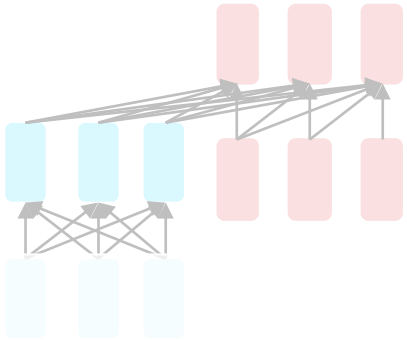
Pretraining for three types of architectures

The neural architecture influences the type of pretraining, and natural use cases.



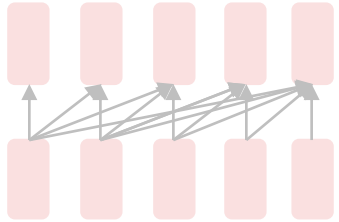
Encoders

- Gets bidirectional context – can condition on future!
- How do we train them to build strong representations?



Encoder-Decoders

- Good parts of decoders and encoders?
- What's the best way to pretrain them?

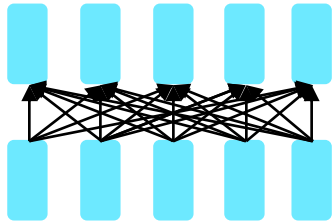


Decoders

- Language models! What we've seen so far.
- Nice to generate from; can't condition on future words

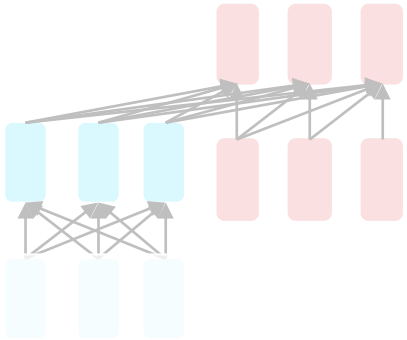
三种架构类型的预训练

神经架构影响预训练的类型和自然的应用场景。



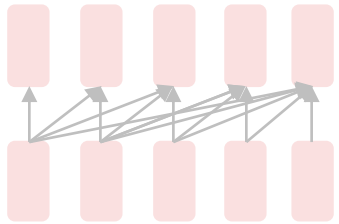
Encoder 类

- 获得双向上下文 —— 可以依赖未来！
- 我们如何训练它们以建立强大的表示？



Encoder-Decoder 类

- Decoder 和 Encoder 的优点？
- What's the best way to pretrain them?



Decoder 类

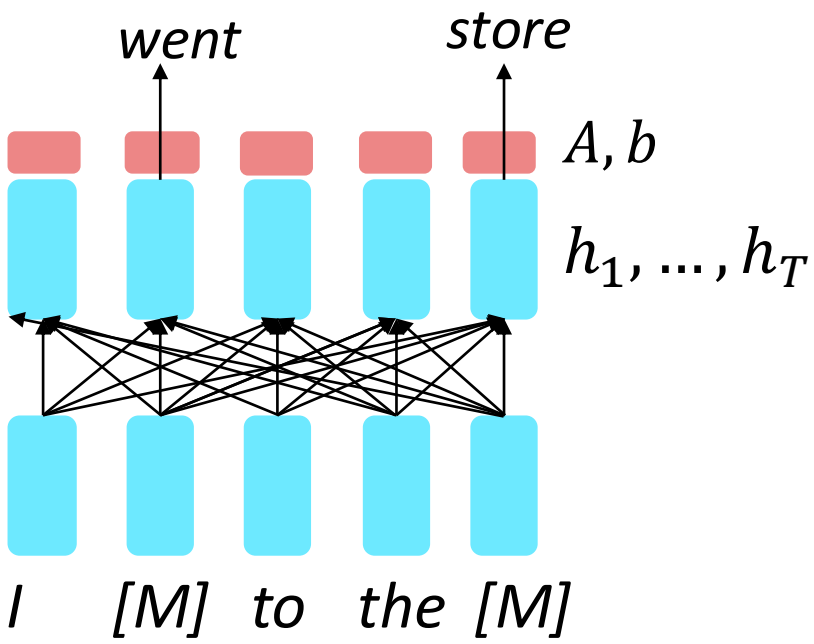
- Language models! What we've seen so far.
- Nice to generate from; can't condition on future words

Pretraining encoders: what pretraining objective to use?

So far, we've looked at language model pretraining. But **encoders get bidirectional context,** so we can't do language modeling!

Idea: replace some fraction of words in the input with a special [MASK] token; predict these words.

$$h_1, \dots, h_T = \text{Encoder}(w_1, \dots, w_T)$$
$$y_i \sim Aw_i + b$$



Only add loss terms from words that are "masked out." If $\tilde{x}$ is the masked version of x, we're learning $p_\theta(x|\tilde{x})$. Called **Masked LM**.

[Devlin et al., 2018]

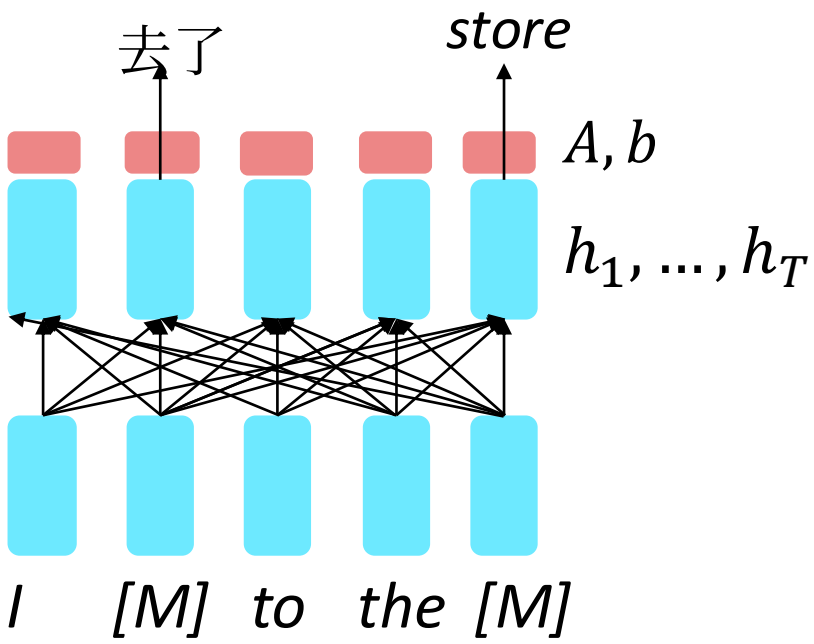
预训练 Encoder：使用什么预训练目标？

So far, we've looked at language model pretraining. But **encoders get bidirectional context,** so we can't do language modeling!

Idea: replace some fraction of words in the input with a special [MASK] token; predict 这些词。

$$h_1, \ldots, h_T = \text{Encoder}(w_1, \ldots, w_T)$$
$$y_i \sim Aw_i + b$$

Only add loss terms from words that are "masked out." If $\tilde{x}$ is the masked version of x, we're learning $p_\theta(x|\tilde{x})$. Called **Masked LM**.



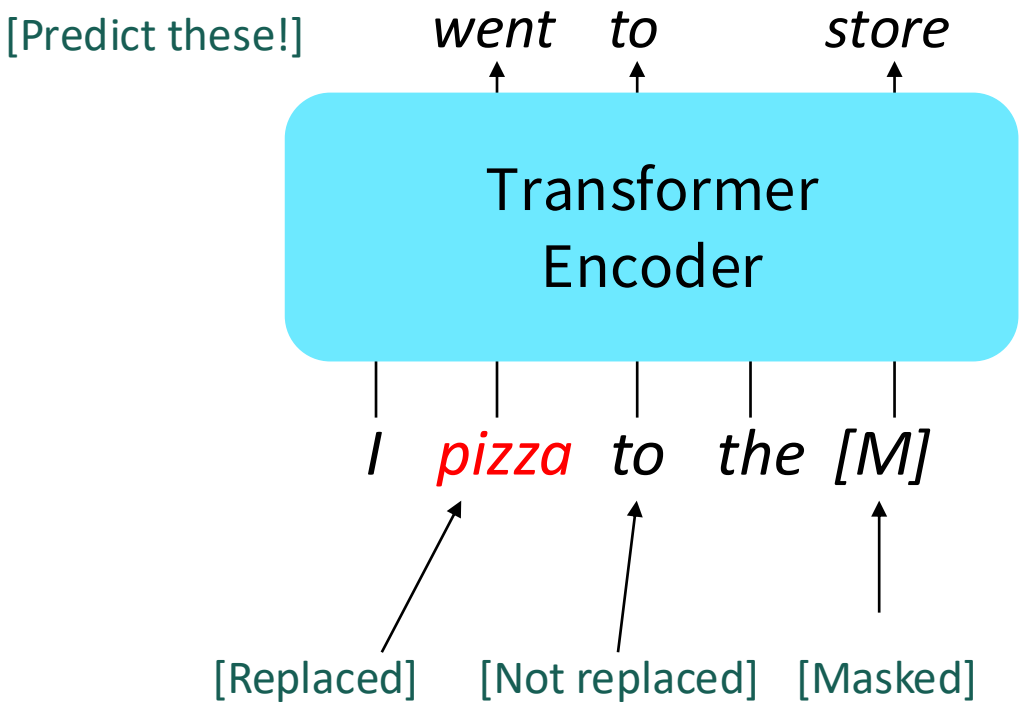
[Devlin et al., 2018]

BERT: Bidirectional Encoder Representations from Transformers

Devlin et al., 2018 proposed the "Masked LM" objective and **released the weights of a pretrained Transformer**, a model they labeled BERT.

Some more details about Masked LM for BERT:

- Predict a random 15% of (sub)word tokens.
  - Replace input word with [MASK] 80% of the time
  - Replace input word with a random token 10% of the time
  - Leave input word unchanged 10% of the time (but still predict it!)
- Why? Doesn't let the model get complacent and not build strong representations of non-masked words. (No masks are seen at fine-tuning time!)



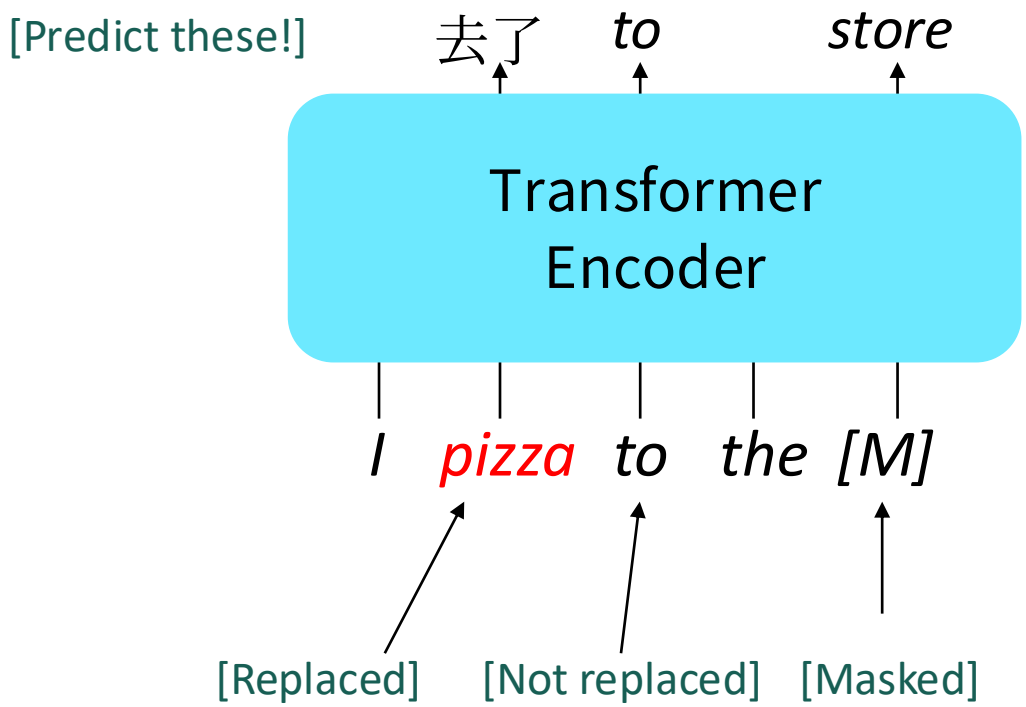
[Devlin et al., 2018]

BERT: Bidirectional Encoder Representations from Transformers

Devlin et al., 2018 proposed the "Masked LM" objective and **released the weights of a**
预训练的 Ｔｒａｎｓｆｏｒｍｅｒ 他们命名为 ＢＥＲＴ 的模型。

关于 ＢＥＲＴ 的 Ｍａｓｋｅｄ ＬＭ 的更多细节：

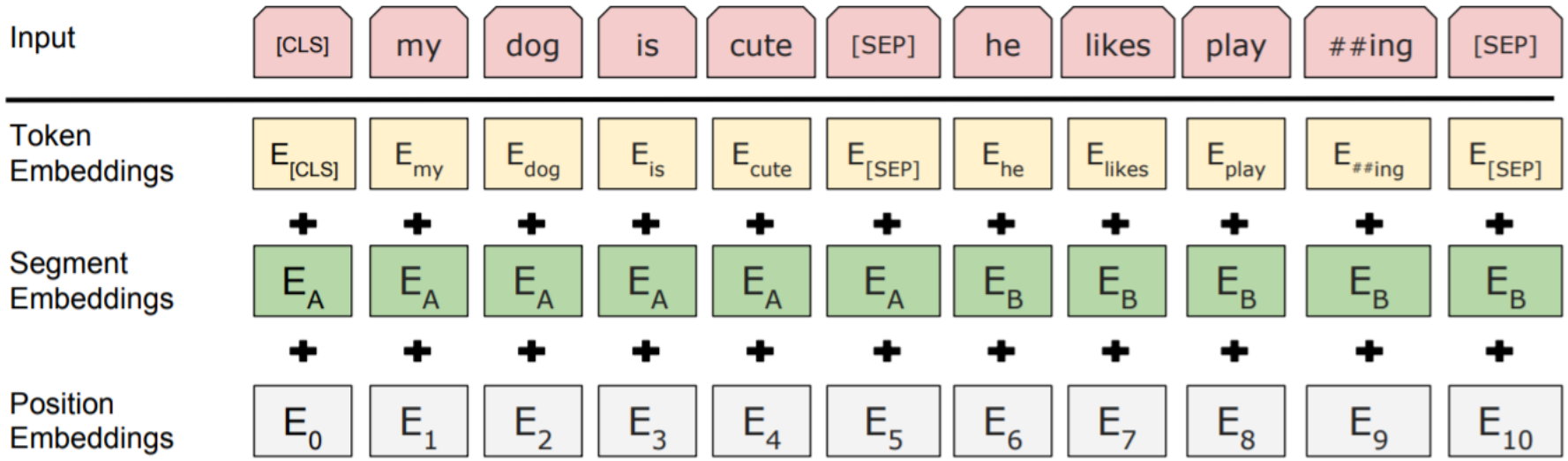
- 随机预测 １５％ 的（子）词 ｔｏｋｅｎ。
  - ８０％ 的时间用 [ＭＡＳＫ] 替换输入词
  - Replace input word with a random token 10% of 时间
  - Leave input word unchanged 10% of the time (but 仍然预测它！)
- Why? Doesn't let the model get complacent and not 建立未遮蔽词的强表示。
  （ｆｉｎｅ-ｔｕｎｉｎｇ 时看不到 ｍａｓｋ！）



[Devlin et al., 2018]

BERT: Bidirectional Encoder Representations from Transformers

- The pretraining input to BERT was two separate contiguous chunks of text:

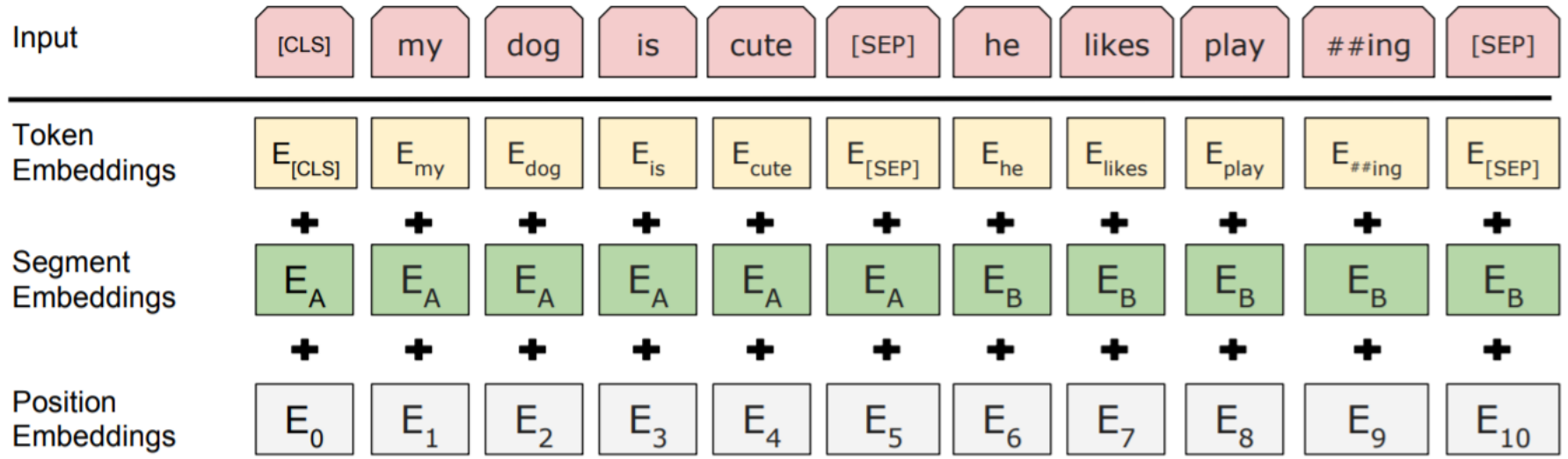


- BERT was trained to predict whether one chunk follows the other or is randomly sampled.
  - Later work has argued this "next sentence prediction" is not necessary.

[Devlin et al., 2018, Liu et al., 2019]

BERT: Bidirectional Encoder Representations from Transformers

- ＢＥＲＴ　的预训练输入是两个独立的连续文本块：



- BERT was trained to predict whether one chunk follows the other or is randomly 采样的。
  - Later work has argued this “next sentence prediction” is not necessary.

[Devlin et al., 2018, Liu et al., 2019]

BERT: Bidirectional Encoder Representations from Transformers

Details about BERT

- Two models were released:
  - BERT-base: 12 layers, 768-dim hidden states, 12 attention heads, 110 million params.
  - BERT-large: 24 layers, 1024-dim hidden states, 16 attention heads, 340 million params.
- Trained on:
  - BooksCorpus (800 million words)
  - English Wikipedia (2,500 million words)
- Pretraining is expensive and impractical on a single GPU.
  - BERT was pretrained with 64 TPU chips for a total of 4 days.
  - (TPUs are special tensor operation acceleration hardware)
- Finetuning is practical and common on a single GPU
  - “Pretrain once, finetune many times.”

[Devlin et al., 2018]

BERT: Bidirectional Encoder Representations from Transformers

关于 BERT 的详细信息

- 发布了两个模型：
  - BERT-base：12层，768维隐藏状态，12个 attention head，1.1亿
  - BERT-large：24层，1024维隐藏状态，16个 attention head，3.
- 训练数据：
  - BooksCorpus（8亿词）
  - 英文维基百科（25亿词）
- 预训练在单个 GPU 上既昂贵又不实际。
  - BERT 使用 64 个 TPU 芯片预训练了共 4 天。
  - （TPU 是专用的张量运算加速硬件）
- 在单个 GPU 上 fine-tuning 是实际且常见的
  - “Pretrain once, finetune many times.”

[Devlin et al., 2018]

BERT: Bidirectional Encoder Representations from Transformers

BERT was massively popular and hugely versatile; finetuning BERT led to new state-of-the-art results on a broad range of tasks.

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

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

[Devlin et al., 2018]

BERT: Bidirectional Encoder Representations from Transformers

ＢＥＲＴ　非常流行且用途广泛；ｆｉｎｅ-ｔｕｎｉｎｇ　ＢＥＲＴ　带来了新的最先进最先进的结果，涵盖广泛的任务。

- **QQP:** Quora Question Pairs (detect paraphrase 问题)
- **QNLI**: natural language inference over question 问答数据
- **SST-2**：情感分析
- **CoLA**: corpus of linguistic acceptability (detect 句子是否合乎语法。)
- **STS-B**：语义文本相似度
- **MRPC**：Ｍｉｃｒｏｓｏｆｔ　释义语料库
- **RTE**：一个小的自然语言推理语料库

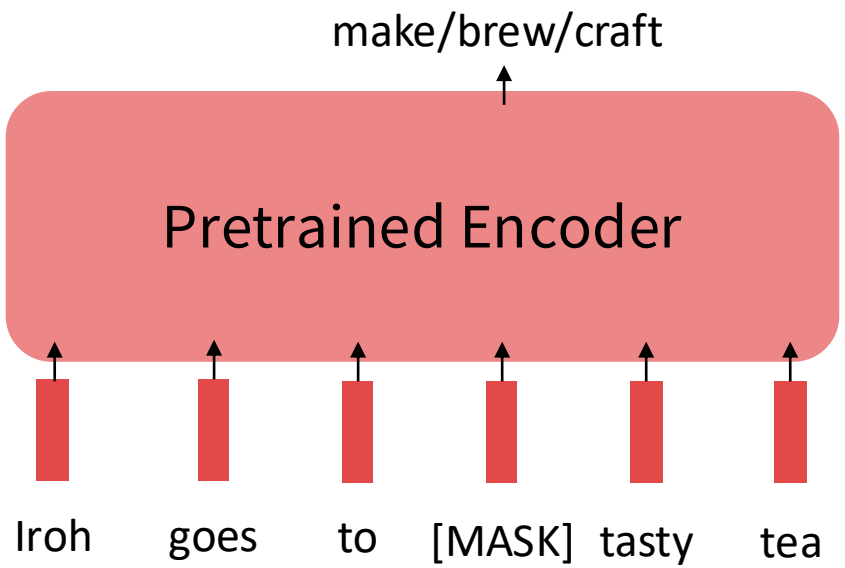
System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Average**
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1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

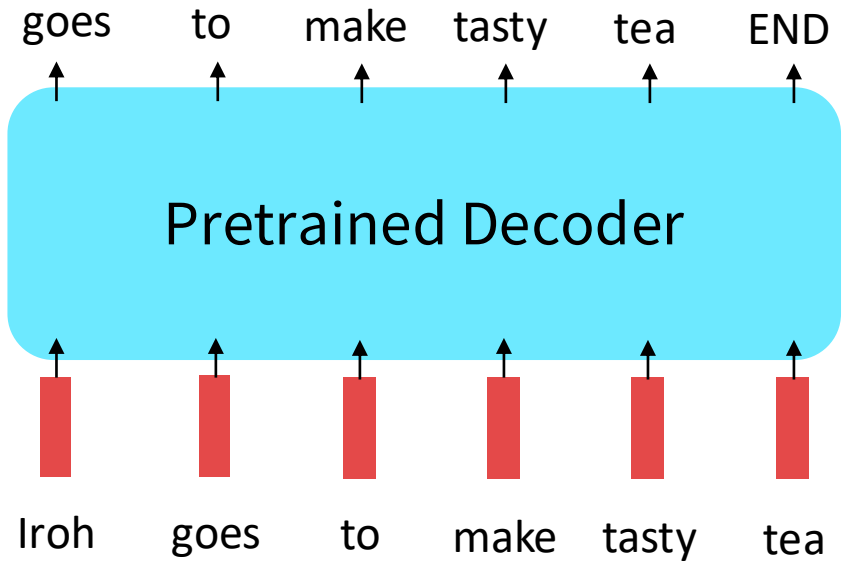
[Devlin et al., 2018]

Limitations of pretrained encoders

Those results looked great! Why not used pretrained encoders for everything?

If your task involves generating sequences, consider using a pretrained decoder; BERT and other pretrained encoders don't naturally lead to nice autoregressive (1-word-at-a-time) generation methods.

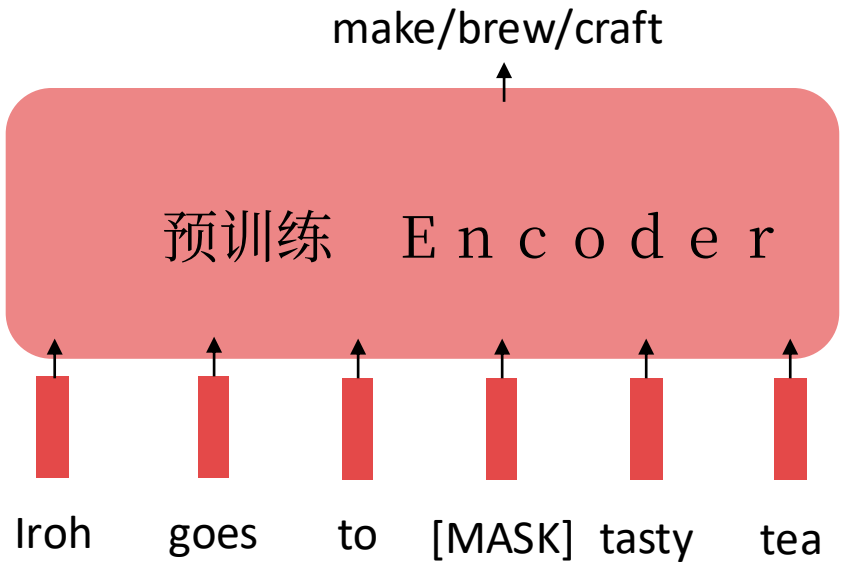


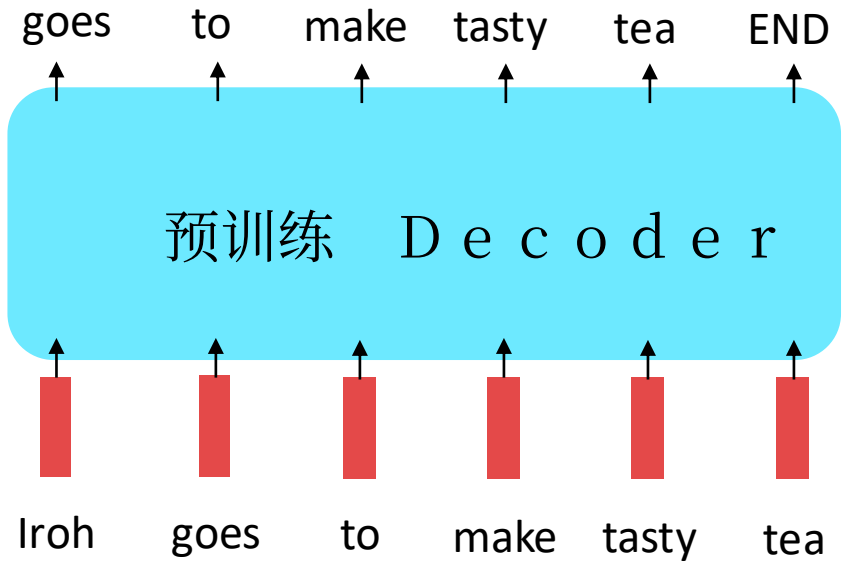


预训练 Encoder 的局限性

那些结果看起来很棒！为什么不对所有任务都使用预训练 Encoder？

If your task involves generating sequences, consider using a pretrained decoder; BERT and other pretrained encoders don't naturally lead to nice autoregressive (1-word-at-a-time) generation 方法。



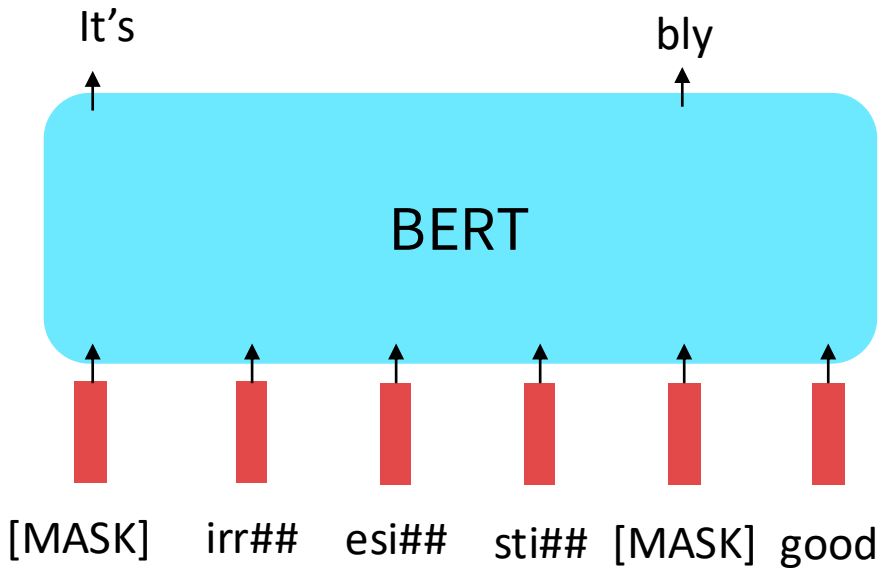


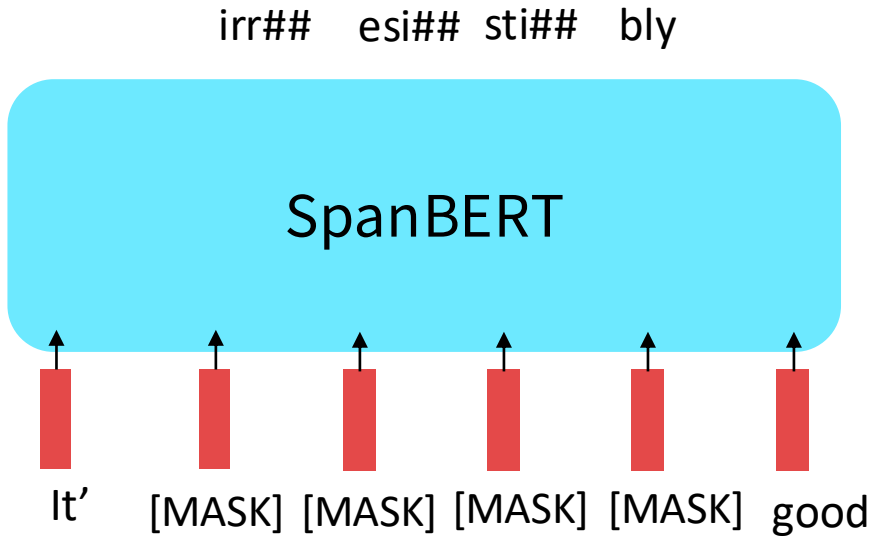
Extensions of BERT

You'll see a lot of BERT variants like RoBERTa, SpanBERT, +++

Some generally accepted improvements to the BERT pretraining formula:

- RoBERTa: mainly just train BERT for longer and remove next sentence prediction!
- SpanBERT: masking contiguous spans of words makes a harder, more useful pretraining task





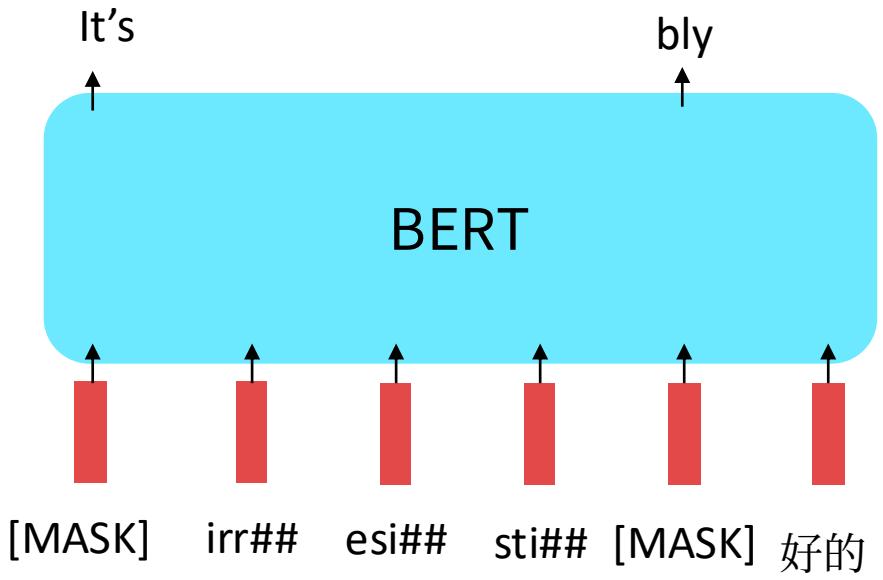
[Liu et al., 2019; Joshi et al., 2020]

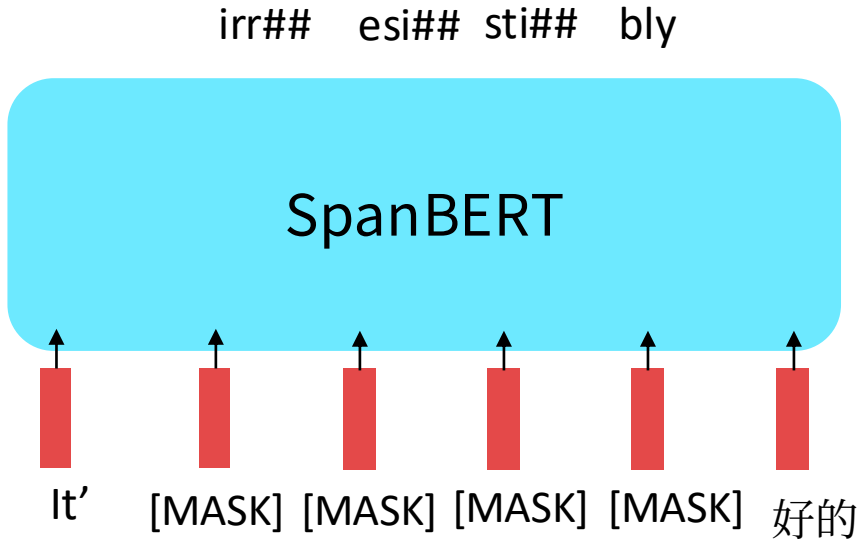
ＢＥＲＴ　的扩展

You'll see a lot of BERT variants like RoBERTa, SpanBERT, +++

一些对　ＢＥＲＴ　预训练方案普遍接受的改进：

- ＲｏＢＥＲＴａ：主要就是训练　ＢＥＲＴ　更长时间并移除下一句预测！
- ＳｐａｎＢＥＲＴ：遮蔽连续的词片段使预训练任务更难也更有用





[Liu et al., 2019; Joshi et al., 2020]

Extensions of BERT

A takeaway from the RoBERTa paper: more compute, more data can improve pretraining even when not changing the underlying Transformer encoder.

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; Joshi et al., 2020]

BERT 的扩展

A takeaway from the RoBERTa paper: more compute, more data can improve pretraining

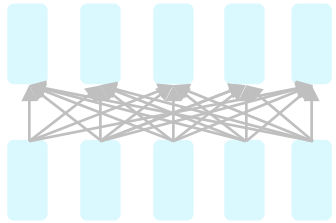
即使不改变底层的 Transformer encoder。

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; Joshi et al., 2020]

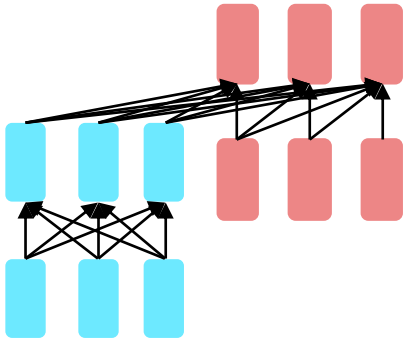
Pretraining for three types of architectures

The neural architecture influences the type of pretraining, and natural use cases.



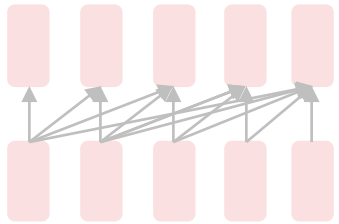
Encoders

- Gets bidirectional context – can condition on future!
- How do we train them to build strong representations?



Encoder-Decoders

- Good parts of decoders and encoders?
- What's the best way to pretrain them?

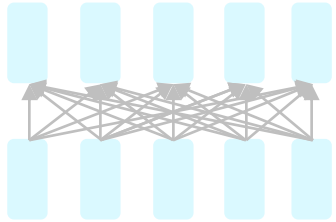


Decoders

- Language models! What we've seen so far.
- Nice to generate from; can't condition on future words

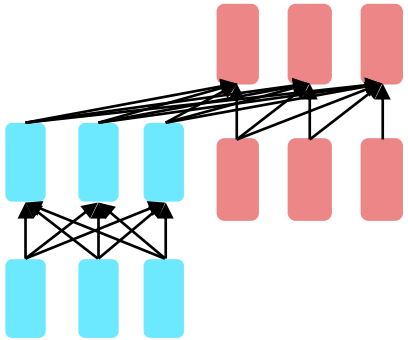
三种架构类型的预训练

神经架构影响预训练的类型和自然的应用场景。



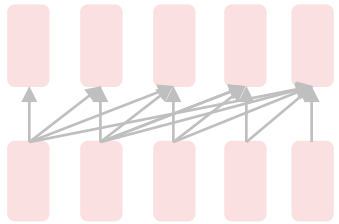
Encoder 类

- 获得双向上下文 —— 可以依赖未来！
- 我们如何训练它们以建立强大的表示？



Encoder-Decoder 类

- Decoder 和 Encoder 的优点？
- What's the best way to pretrain them?



Decoder 类

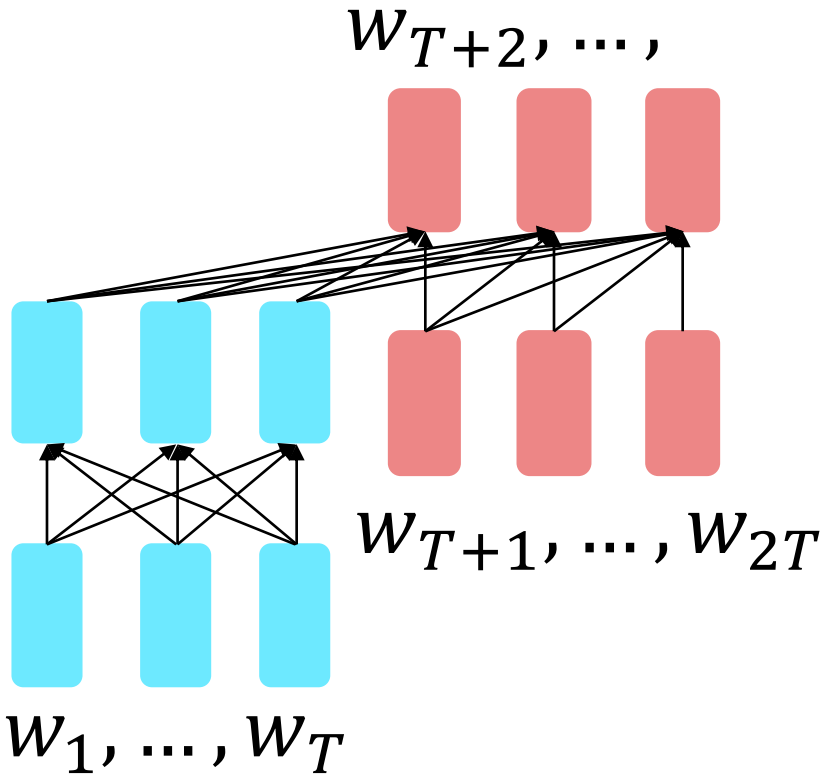
- Language models! What we've seen so far.
- Nice to generate from; can't condition on future words

Pretraining encoder-decoders: what pretraining objective to use?

For **encoder-decoders**, we could do something like **language modeling**, but where a prefix of every input is provided to the encoder and is not predicted.

$$h_1, \ldots, h_T = \text{Encoder}(w_1, \ldots, w_T)$$
$$h_{T+1}, \ldots, h_2 = Decoder(w_1, \ldots, w_T, h_1, \ldots, h_T)$$
$$y_i \sim Ah_i + b, i > T$$

The **encoder** portion benefits from bidirectional context; the **decoder** portion is used to train the whole model through language modeling.

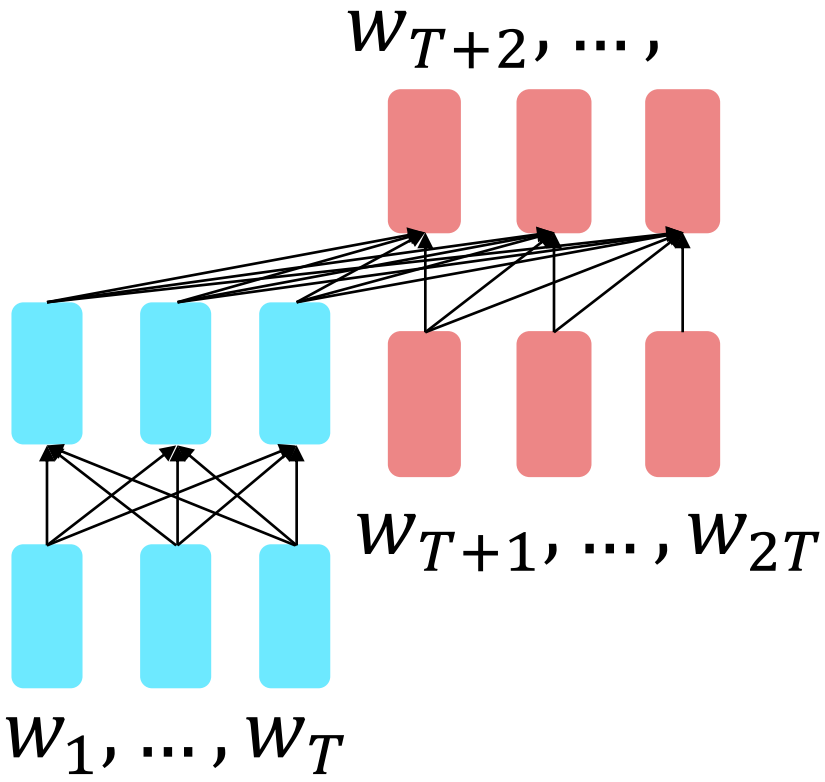


[Raffel et al., 2018]

预训练 encoder-decoder：使用什么预训练目标？

For encoder-decoder we could do something like 语言建模 , but where a
每个输入的前缀提供给 encoder 且不被预测。

$$h_1, \ldots, h_T = \text{Encoder}(w_1, \ldots, w_T)$$
$$h_{T+1}, \ldots, h_2 = Decoder(w_1, \ldots, w_T, h_1, \ldots, h_T)$$
$$y_i \sim Ah_i + b, i > T$$



The **encoder** portion benefits from bidirectional context; the **decoder** portion is used to train the whole model through 语言建模。

[Raffel et al., 2018]

Pretraining encoder-decoders: what pretraining objective to use?

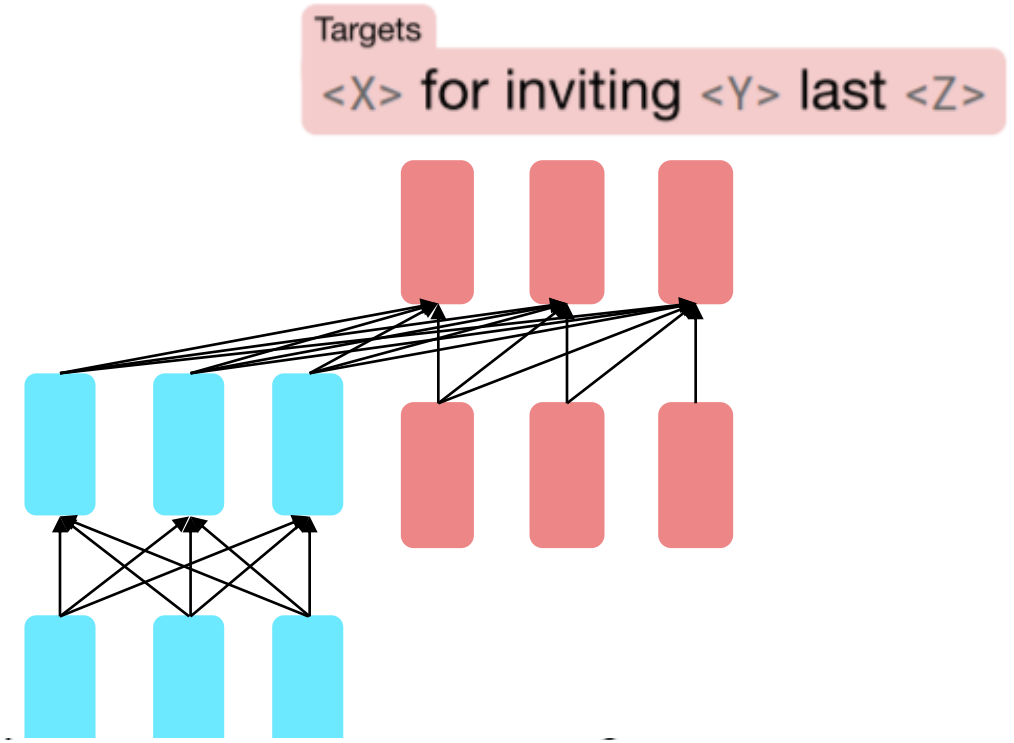
What Raffel et al., 2018 found to work best was **span corruption.** Their model: **T5**.

Replace different-length spans from the input with unique placeholders; decode out the spans that were removed!

Original text

Thank you ~~for inviting~~ me to your party ~~last~~ week.

This is implemented in text preprocessing: it's still an objective that looks like **language modeling** at the decoder side.



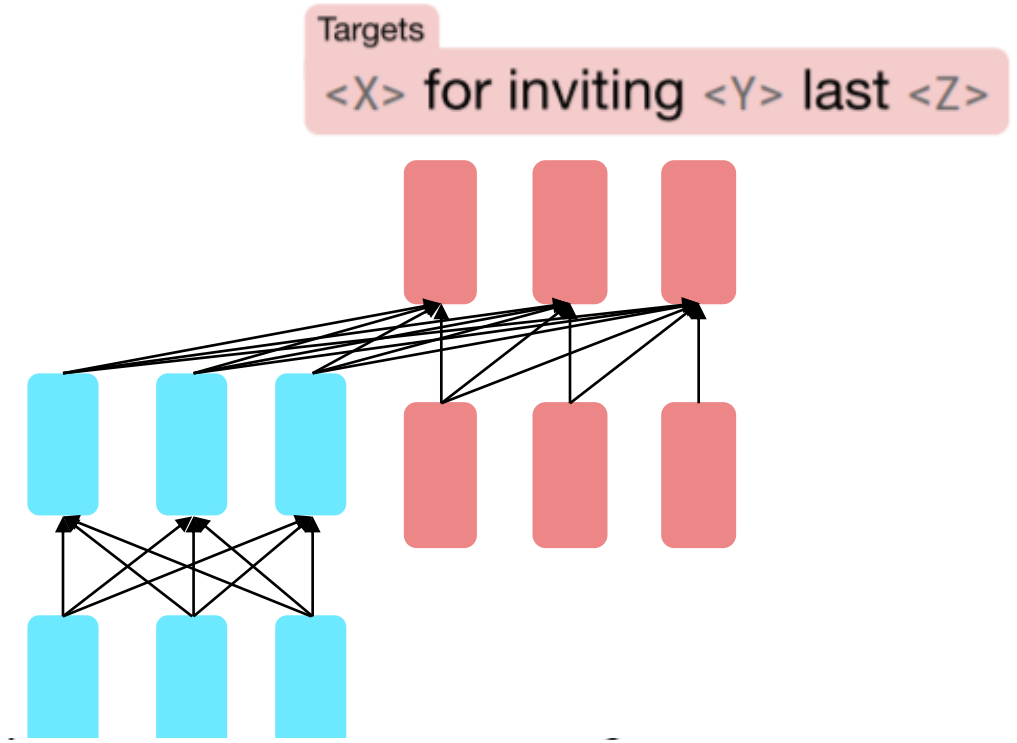
预训练 encoder-decoder：使用什么预训练目标？

What Raffel et al., 2018 found to work best was **span corruption.** Their model: **T5.**

Replace different-length spans from the input with unique placeholders; decode out the 被移除的片段！

Original text

Thank you for inviting me to your party last week.



This is implemented in text preprocessing: it's still an objective that looks like 语言建模 at the decoder side.

Inputs

Thank you <X> me to your party <Y> week.

Pretraining encoder-decoders: what pretraining objective to use?

Raffel et al., 2018 found encoder-decoders to work better than decoders for their tasks, and span corruption (denoising) to work better than language modeling.

Architecture	Objective	Params	Cost	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Encoder-decoder	Denoising	$2P$	M	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
Enc-dec, shared	Denoising	P	M	82.81	18.78	**80.63**	**70.73**	26.72	39.03	**27.46**
Enc-dec, 6 layers	Denoising	P	$M/2$	80.88	18.97	77.59	68.42	26.38	38.40	26.95
Language model	Denoising	P	M	74.70	17.93	61.14	55.02	25.09	35.28	25.86
Prefix LM	Denoising	P	M	81.82	18.61	78.94	68.11	26.43	37.98	27.39
Encoder-decoder	LM	$2P$	M	79.56	18.59	76.02	64.29	26.27	39.17	26.86
Enc-dec, shared	LM	P	M	79.60	18.13	76.35	63.50	26.62	39.17	27.05
Enc-dec, 6 layers	LM	P	$M/2$	78.67	18.26	75.32	64.06	26.13	38.42	26.89
Language model	LM	P	M	73.78	17.54	53.81	56.51	25.23	34.31	25.38
Prefix LM	LM	P	M	79.68	17.84	76.87	64.86	26.28	37.51	26.76

预训练 encoder-decoder：使用什么预训练目标？

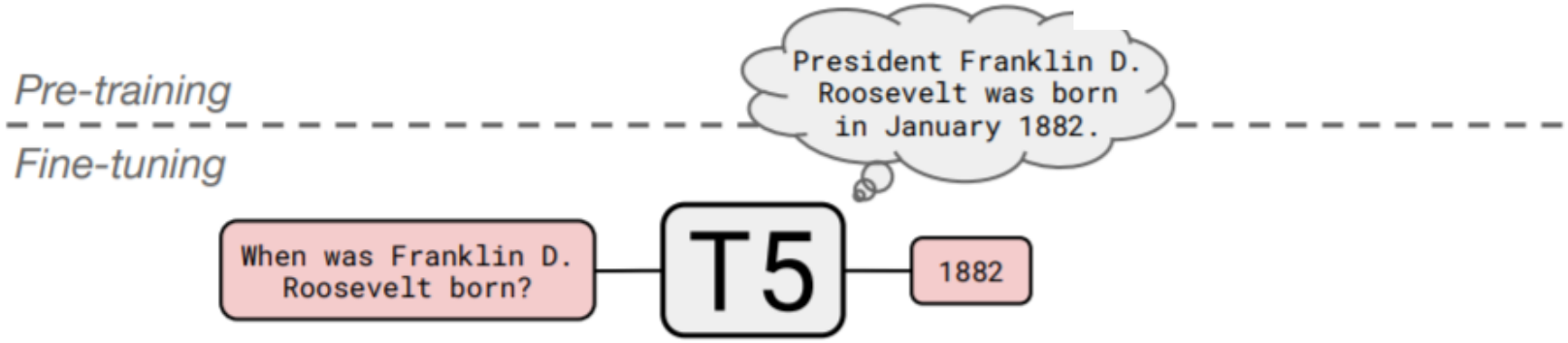
Raffel et al., 2018 found encoder-decoders to work better than decoders for their tasks,

和 span corruption（去噪）比语言建模效果更好。

Architecture	Objective	Params	Cost	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Encoder-decoder	Denoising	$2P$	M	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
Enc-dec, shared	Denoising	P	M	82.81	18.78	**80.63**	**70.73**	26.72	39.03	**27.46**
Enc-dec, 6 layers	Denoising	P	$M/2$	80.88	18.97	77.59	68.42	26.38	38.40	26.95
Language model	Denoising	P	M	74.70	17.93	61.14	55.02	25.09	35.28	25.86
Prefix LM	Denoising	P	M	81.82	18.61	78.94	68.11	26.43	37.98	27.39
Encoder-decoder	LM	$2P$	M	79.56	18.59	76.02	64.29	26.27	39.17	26.86
Enc-dec, shared	LM	P	M	79.60	18.13	76.35	63.50	26.62	39.17	27.05
Enc-dec, 6 layers	LM	P	$M/2$	78.67	18.26	75.32	64.06	26.13	38.42	26.89
Language model	LM	P	M	73.78	17.54	53.81	56.51	25.23	34.31	25.38
Prefix LM	LM	P	M	79.68	17.84	76.87	64.86	26.28	37.51	26.76

Pretraining encoder-decoders: what pretraining objective to use?

A fascinating property of T5: it can be finetuned to answer a wide range of questions, retrieving knowledge from its parameters.



NQ: Natural Questions

WQ: WebQuestions

TQA: Trivia QA

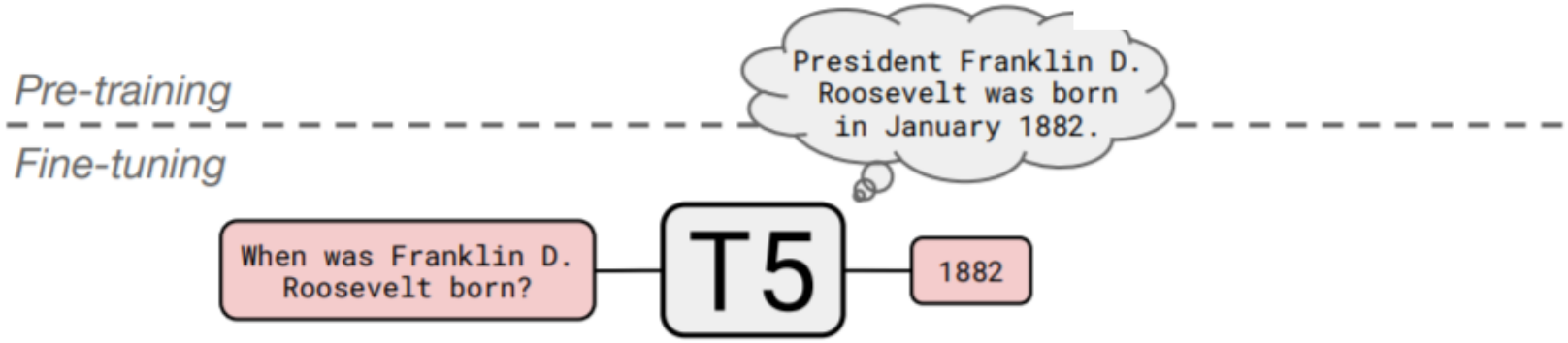
	NQ	WQ	TQA dev	TQA test	
Karpukhin et al. (2020)	**41.5**	42.4	**57.9**	–	
T5.1.1-Base	25.7	28.2	24.2	30.6	**220 million params**
T5.1.1-Large	27.3	29.5	28.5	37.2	**770 million params**
T5.1.1-XL	29.5	32.4	36.0	45.1	**3 billion params**
T5.1.1-XXL	32.8	35.6	42.9	52.5	**11 billion params**
T5.1.1-XXL + SSM	35.2	**42.8**	51.9	**61.6**	

All “open-domain” versions

[Raffel et al., 2018]

预训练 encoder-decoder：使用什么预训练目标？

A fascinating property of T5: it can be finetuned to answer a wide range of questions, retrieving knowledge from its 参数。



NQ：Natural Questions
WQ: WebQuestions
TQA：Trivia QA

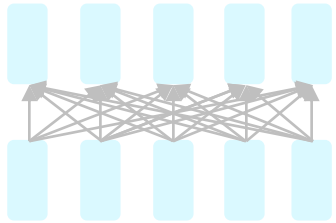
All “open-domain” 版本

	NQ	WQ	TQA dev	TQA test	
Karpukhin et al. (2020)	**41.5**	42.4	**57.9**	–	
T5.1.1-Base	25.7	28.2	24.2	30.6	2．2亿参数
T5.1.1-Large	27.3	29.5	28.5	37.2	7．7亿参数
T5.1.1-XL	29.5	32.4	36.0	45.1	30亿参数
T5.1.1-XXL	32.8	35.6	42.9	52.5	110亿参数
T5.1.1-XXL + SSM	35.2	**42.8**	51.9	**61.6**	

[Raffel et al., 2018]

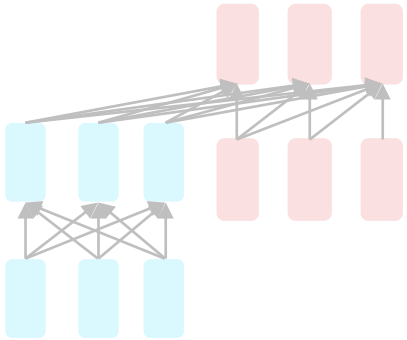
Pretraining for three types of architectures

The neural architecture influences the type of pretraining, and natural use cases.



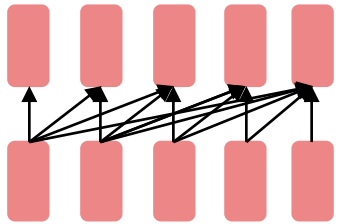
Encoders

- Gets bidirectional context – can condition on future!
- How do we train them to build strong representations?



Encoder-Decoders

- Good parts of decoders and encoders?
- What's the best way to pretrain them?

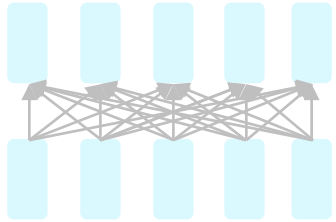


Decoders

- Language models! What we've seen so far.
- Nice to generate from; can't condition on future words
- All the biggest pretrained models are Decoders.

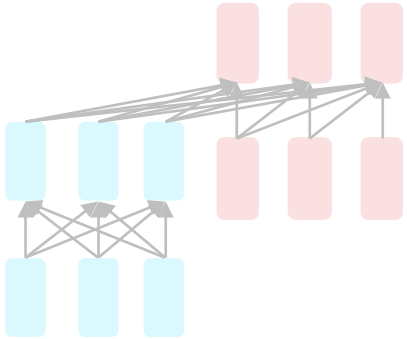
三种架构类型的预训练

神经架构影响预训练的类型和自然的应用场景。



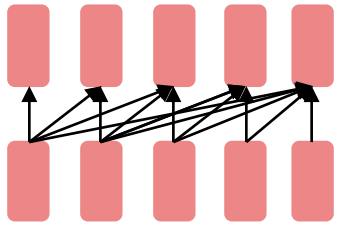
Encoder 类

- 获得双向上下文 —— 可以依赖未来！
- 我们如何训练它们以建立强大的表示？



Encoder-Decoder 类

- Decoder 和 Encoder 的优点？
- What's the best way to pretrain them?



Decoder 类

- Language models! What we've seen so far.
- Nice to generate from; can't condition on future words
- 所有最大的预训练模型都是 Decoder。

Pretraining decoders

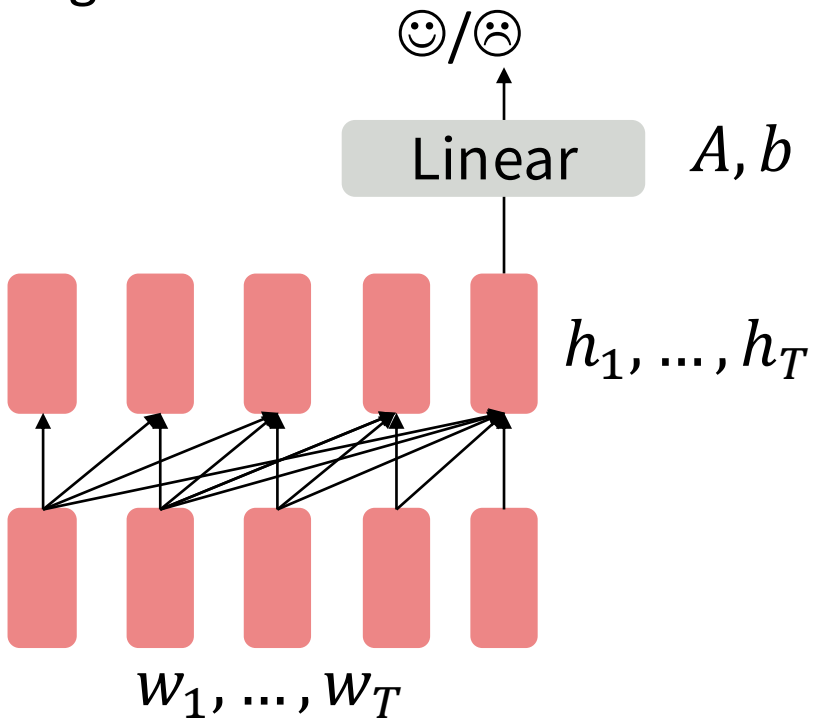
When using language model pretrained decoders, we can ignore that they were trained to model $p(w_t|w_{1:t-1})$.

We can finetune them by training a classifier on the last word's hidden state.

$$h_1, \ldots, h_T = \text{Decoder}(w_1, \ldots, w_T)$$
$$y \sim Ah_T + b$$

Where A and b are randomly initialized and specified by the downstream task.

Gradients backpropagate through the whole network.



[Note how the linear layer hasn't been pretrained and must be learned from scratch.]

预训练 Decoder

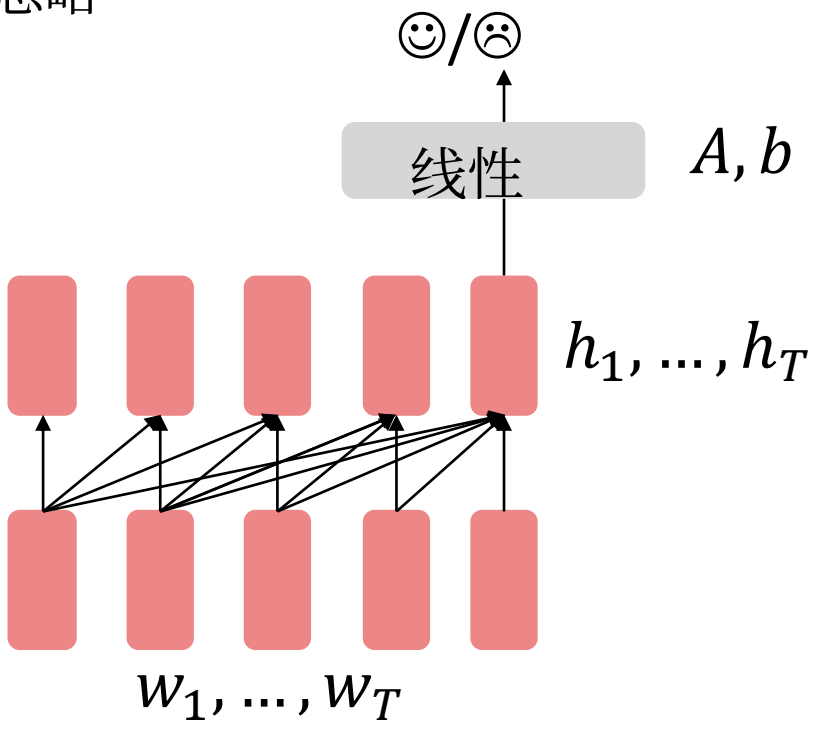
使用语言模型预训练的 decoder 时，我们可以忽略 that they were trained to model $p(w_t|w_{1:t-1})$.

We can finetune them by training a classifier on the last word's hidden state.

$$h_1, \dots, h_T = \text{Decoder}(w_1, \dots, w_T)$$
$$y \sim Ah_T + b$$

Where A and b are randomly initialized and 由下游任务指定。

Gradients backpropagate through the whole 网络。



[Note how the linear layer hasn't been 预训练的，必须从头学习。]

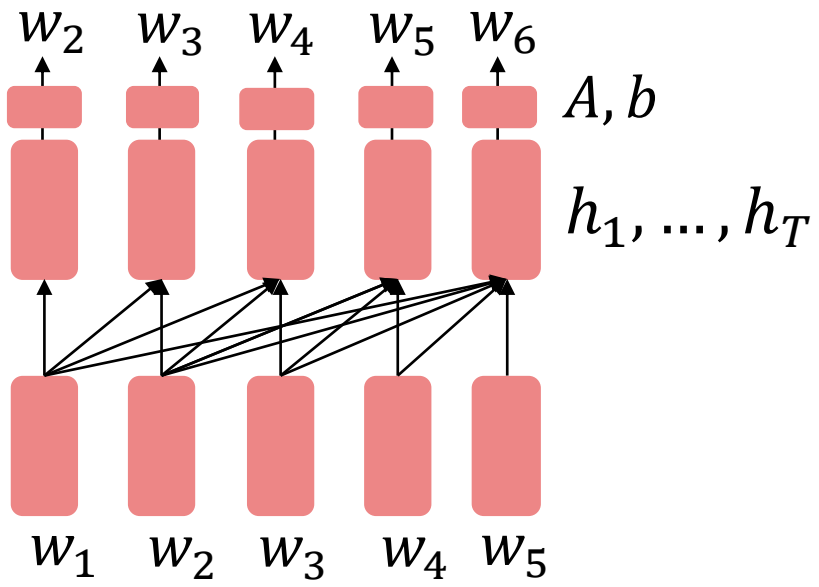
Pretraining decoders

It's natural to pretrain decoders as language models and then use them as generators, finetuning their $p_\theta(w_t|w_{1:t-1})$!

This is helpful in tasks **where the output is a sequence** with a vocabulary like that at pretraining time!

- Dialogue (context=dialogue history)
- Summarization (context=document)

$$h_1, \ldots, h_T = \text{Decoder}(w_1, \ldots, w_T)$$
$$w_t \sim Ah_{t-1} + b$$



[Note how the linear layer has been pretrained.]

Where A, b were pretrained in the language model!

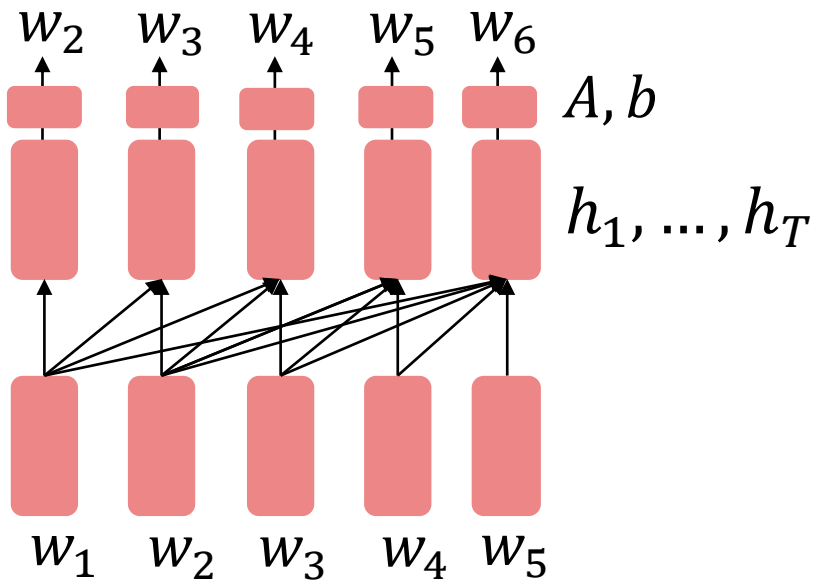
预训练 Decoder

It's natural to pretrain decoders as language models and then use them as generators, finetuning their $p_\theta(w_t|w_{1:t-1})$!

This is helpful in tasks **where the output is a sequence** with a vocabulary like that at pretraining time!

- 对话（上下文 = 对话历史）
- 摘要（上下文 = 文档）

$$h_1, \ldots, h_T = \text{Decoder}(w_1, \ldots, w_T)$$
$$w_t \sim Ah_{t-1} + b$$



[Note how the linear layer has been pretrained.]

Where A, b were pretrained in the language 模型！

Generative Pretrained Transformer (GPT) [Radford et al., 2018]

2018's GPT was a big success in pretraining a decoder!

- Transformer decoder with 12 layers, 117M parameters.
- 768-dimensional hidden states, 3072-dimensional feed-forward hidden layers.
- Byte-pair encoding with 40,000 merges

- Trained on BooksCorpus: over 7000 unique books.
  - Contains long spans of contiguous text, for learning long-distance dependencies.

- The acronym "GPT" never showed up in the original paper; it could stand for "Generative PreTraining" or "Generative Pretrained Transformer"

生成式预训练 Transformer（GPT）[Radford et al., 2018]

2018's GPT was a big success in pretraining a decoder!

- 12层的 Transformer decoder，1.17亿参数。
- 768维隐藏状态，3072维前馈隐藏层。
- 40,000次合并的字节对编码

- 在 BooksCorpus 上训练：超过7000本独特的书。
  - 包含长段连续文本，用于学习长距离依赖。

- The acronym "GPT" never showed up in the original paper; it could stand for "Generative PreTraining" or "Generative Pretrained Transformer"

Generative Pretrained Transformer (GPT) [Radford et al., 2018]

How do we format inputs to our decoder for **finetuning tasks?**

Natural Language Inference: Label pairs of sentences as *entailing/contradictory/neutral*

Premise: *The man is in the doorway*
Hypothesis: *The person is near the door*

} **entailment**

Radford et al., 2018 evaluate on natural language inference.
Here's roughly how the input was formatted, as a sequence of tokens for the decoder.

[START] *The man is in the doorway* [DELIM] *The person is near the door* [EXTRACT]

The linear classifier is applied to the representation of the [EXTRACT] token.

生成式预训练 Transformer（GPT）[Radford et al., 2018]

How do we format inputs to our decoder for fine-tuning 任务？

自然语言推理： Label pairs of sentences as 蕴含 / 矛盾 / 中性

Premise: *The man is in the doorway*

Hypothesis: *The person is near the door*

} 蕴含

Radford et al., 2018 evaluate on natural language inference.

Here's roughly how the input was formatted, as a sequence of tokens for the decoder.

[START] *The man is in the doorway* [DELIM] *The person is near the door* [EXTRACT]

线性分类器应用于 [EXTRACT] token 的表示。

Generative Pretrained Transformer (GPT) [Radford et al., 2018]

GPT results on various *natural language inference* datasets.

Method	MNLI-m	MNLI-mm	SNLI	SciTail	QNLI	RTE
ESIM + ELMo [44] (5x)	-	-	89.3	-	-	-
CAFE [58] (5x)	80.2	79.0	89.3	-	-	-
Stochastic Answer Network [35] (3x)	80.6	80.1	-	-	-	-
CAFE [58]	78.7	77.9	88.5	83.3		
GenSen [64]	71.4	71.3	-	-	82.3	59.2
Multi-task BiLSTM + Attn [64]	72.2	72.1	-	-	82.1	**61.7**
Finetuned Transformer LM (ours)	**82.1**	**81.4**	**89.9**	**88.3**	**88.1**	56.0

生成式预训练 Transformer（GPT）[Radford et al., 2018]

GPT results on various 自然语言推理 datasets.

Method	MNLI-m	MNLI-mm	SNLI	SciTail	QNLI	RTE
ESIM + ELMo [44] (5x)	-	-	89.3	-	-	-
CAFE [58] (5x)	80.2	79.0	89.3	-	-	-
Stochastic Answer Network [35] (3x)	80.6	80.1	-	-	-	-
CAFE [58]	78.7	77.9	88.5	83.3		
GenSen [64]	71.4	71.3	-	-	82.3	59.2
Multi-task BiLSTM + Attn [64]	72.2	72.1	-	-	82.1	**61.7**
Finetuned Transformer LM (ours)	**82.1**	**81.4**	**89.9**	**88.3**	**88.1**	56.0

Increasingly convincing generations (GPT2) [Radford et al., 2018]

We mentioned how pretrained decoders can be used **in their capacities as language models.**

GPT-2, a larger version (1.5B) of GPT trained on more data, was shown to produce relatively convincing samples of natural language.

Context (human-written): In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.
GPT-2: The scientist named the population, after their distinctive horn, Ovid's Unicorn. These four-horned, silver-white unicorns were previously unknown to science. Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved. Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans. Pérez noticed that the valley had what appeared to be a natural fountain, surrounded by two peaks of rock and silver snow.

越来越令人信服的生成（ＧＰＴ２） [Radford et al., 2018]

We mentioned how pretrained decoders can be used 以它们作为语言模型的能力。

GPT-2, a larger version (1.5B) of GPT trained on more data, was shown to produce relatively 令人信服的自然语言样本。

Context (human-written): In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.
GPT-2: The scientist named the population, after their distinctive horn, Ovid's Unicorn. These four-horned, silver-white unicorns were previously unknown to science. Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved. Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans. Pérez noticed that the valley had what appeared to be a natural fountain, surrounded by two peaks of rock and silver snow.

GPT-3, In-context learning, and very large models

So far, we've interacted with pretrained models in two ways:

- Sample from the distributions they define (maybe providing a prompt)
- Fine-tune them on a task we care about, and take their predictions.

Very large language models seem to perform some kind of learning **without gradient steps** simply from examples you provide within their contexts.

GPT-3 is the canonical example of this. The largest T5 model had 11 billion parameters.

GPT-3 has 175 billion parameters.

GPT-3、In-context learning 和超大模型

So far, we've interacted with pretrained models in two ways:

- 从它们定义的分布中采样（也许提供一个 prompt）
- 在我们关心的任务上 fine-tune 它们，并使用它们的预测。

Very large language models seem to perform some kind of learning **without gradient** 步骤 simply from examples you provide within their contexts.

GPT-3 is the canonical example of this. The largest T5 model had 11 billion parameters.

GPT-3 有 1750 亿参数。

GPT-3, In-context learning, and very large models

Very large language models seem to perform some kind of learning **without gradient steps** simply from examples you provide within their contexts.

The in-context examples seem to specify the task to be performed, and the conditional distribution mocks performing the task to a certain extent.

Input (prefix within a single Transformer decoder context):

“ thanks -> merci

hello -> bonjour

mint -> menthe

otter -> ”

Output (conditional generations):

loutre…”

GPT-3、In-context learning 和超大模型

Very large language models seem to perform some kind of learning **without gradient** 步骤 simply from examples you provide within their contexts.

The in-context examples seem to specify the task to be performed, and the conditional 分布在一定程度上模拟了任务的执行。

输入（单个 Transformer decoder 上下文中的前缀）：

“ thanks -> merci

hello -> bonjour

mint -> menthe

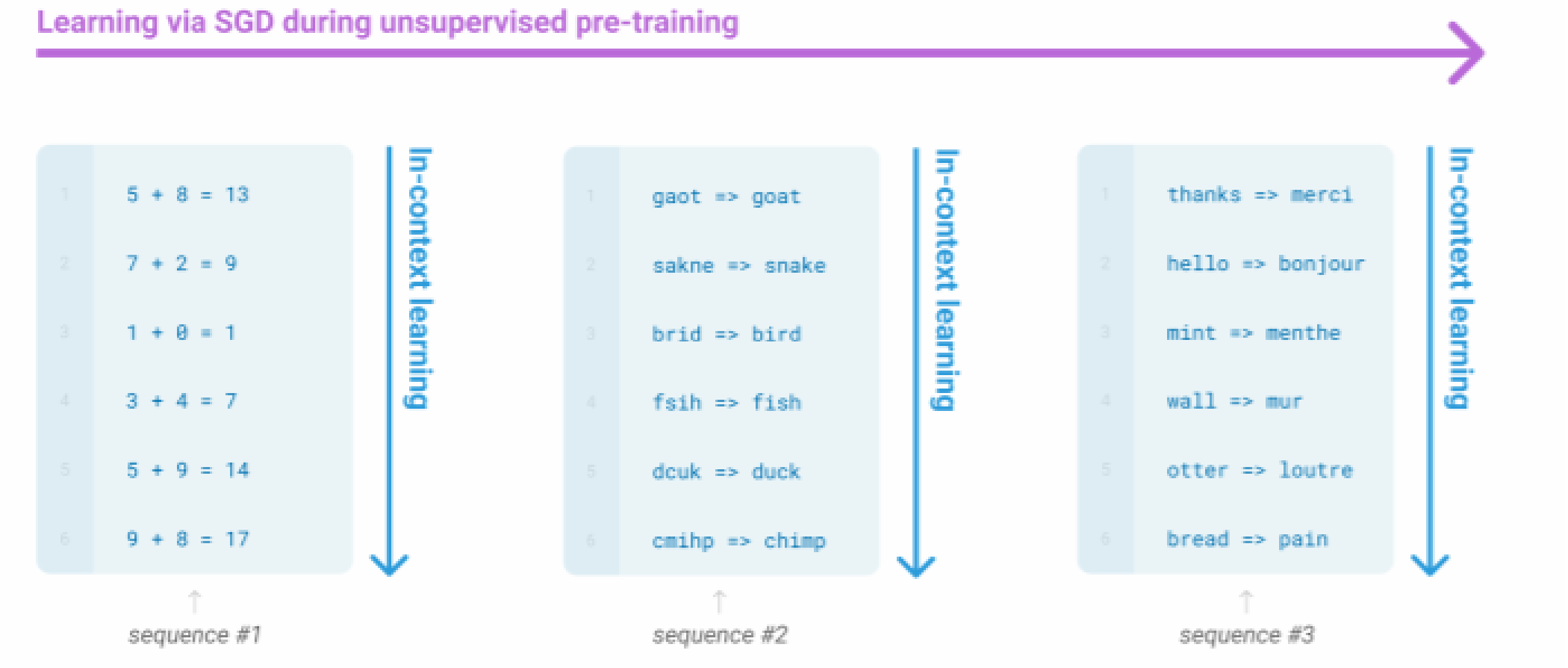
otter -> ”

输出（条件生成）：

loutre…”

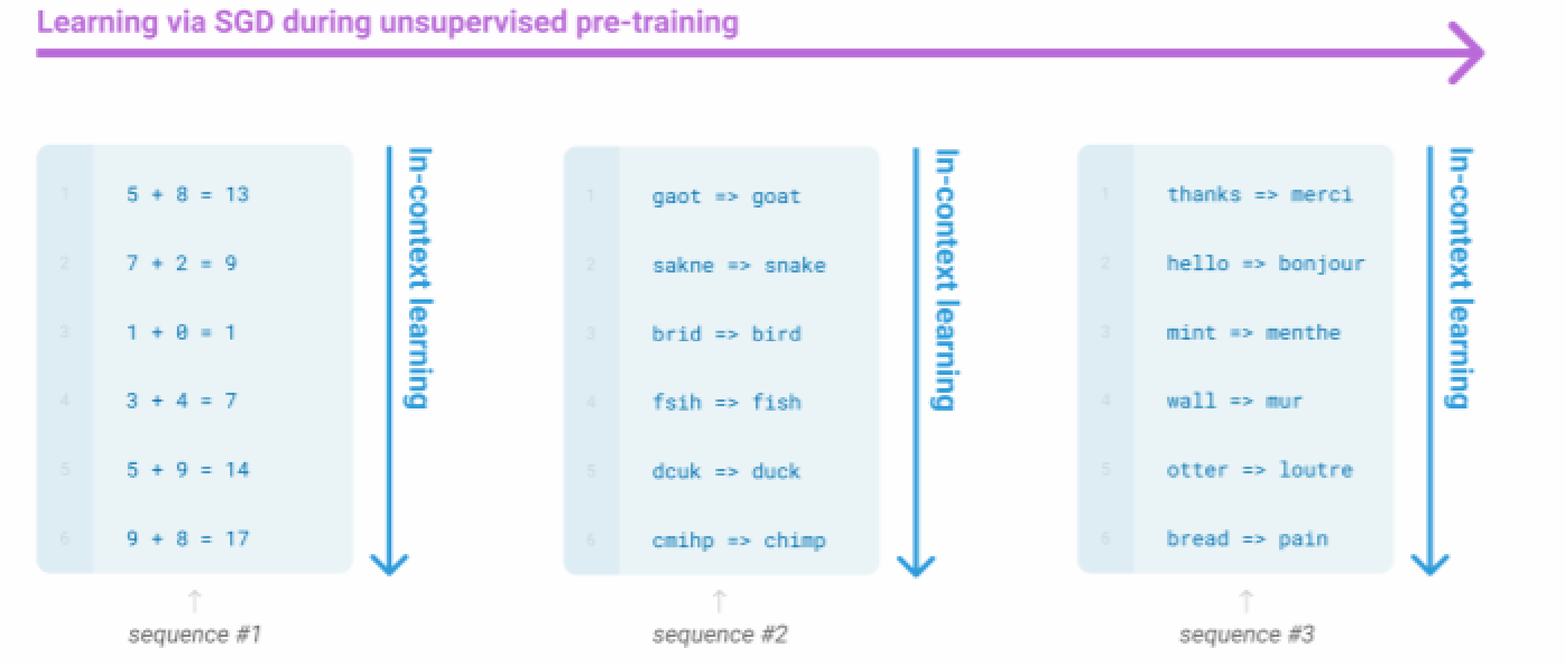
GPT-3, In-context learning, and very large models

Very large language models seem to perform some kind of learning **without gradient steps** simply from examples you provide within their contexts.

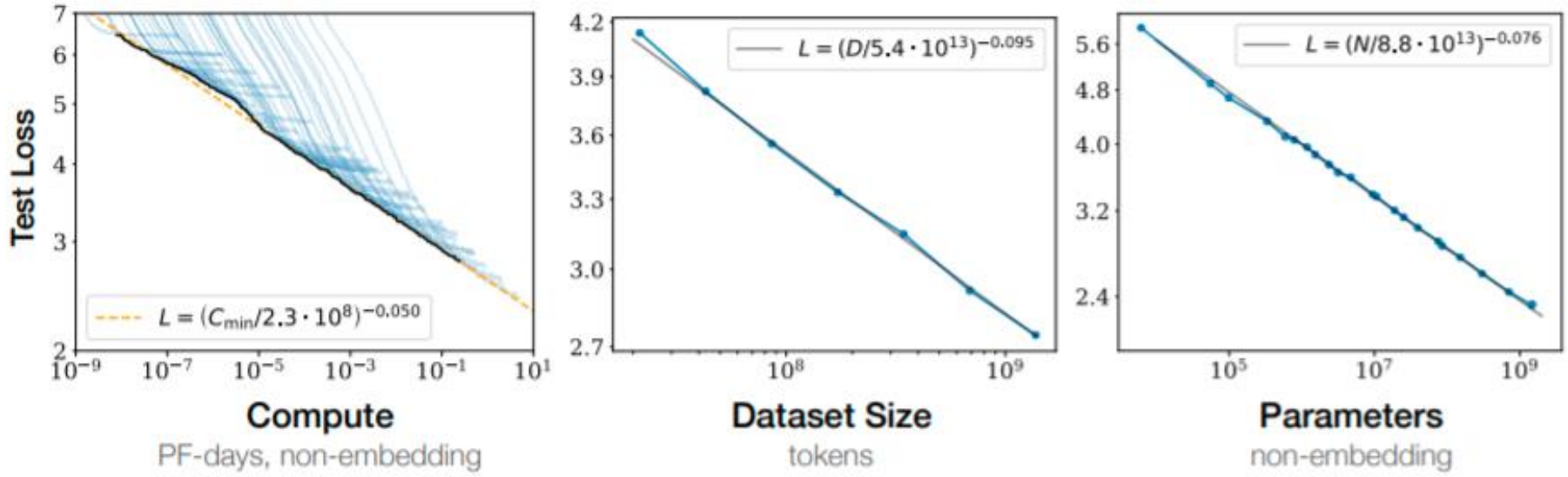


GPT-3、In-context learning 和超大模型

Very large language models seem to perform some kind of learning **without gradient** 步骤 simply from examples you provide within their contexts.

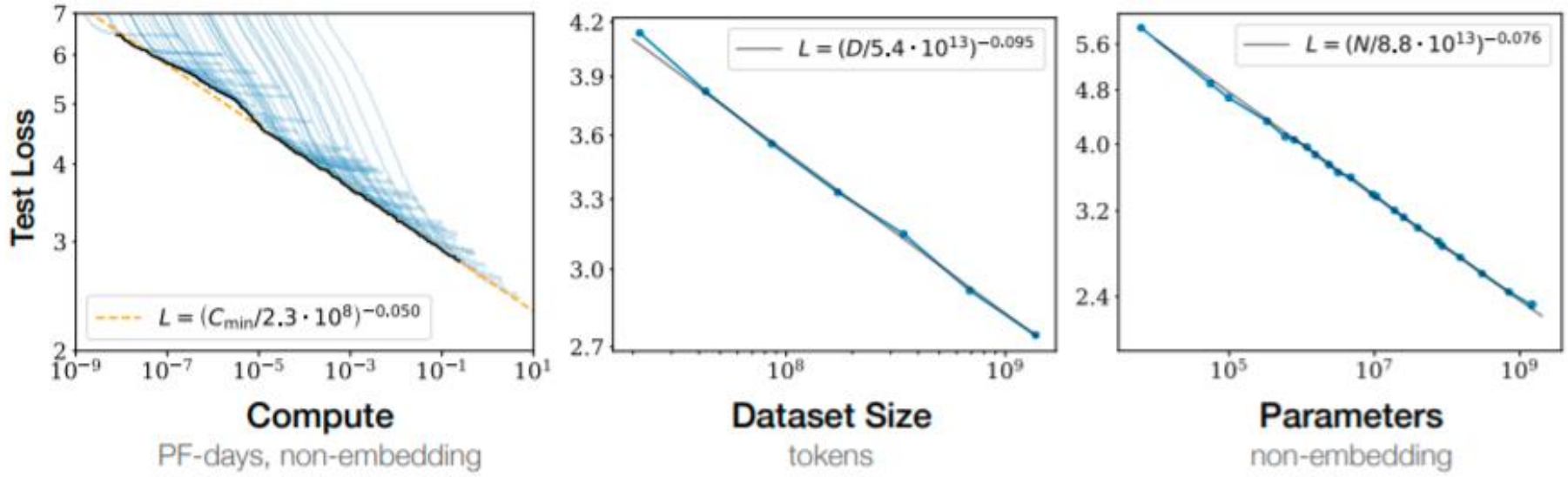


Why scale? Scaling laws



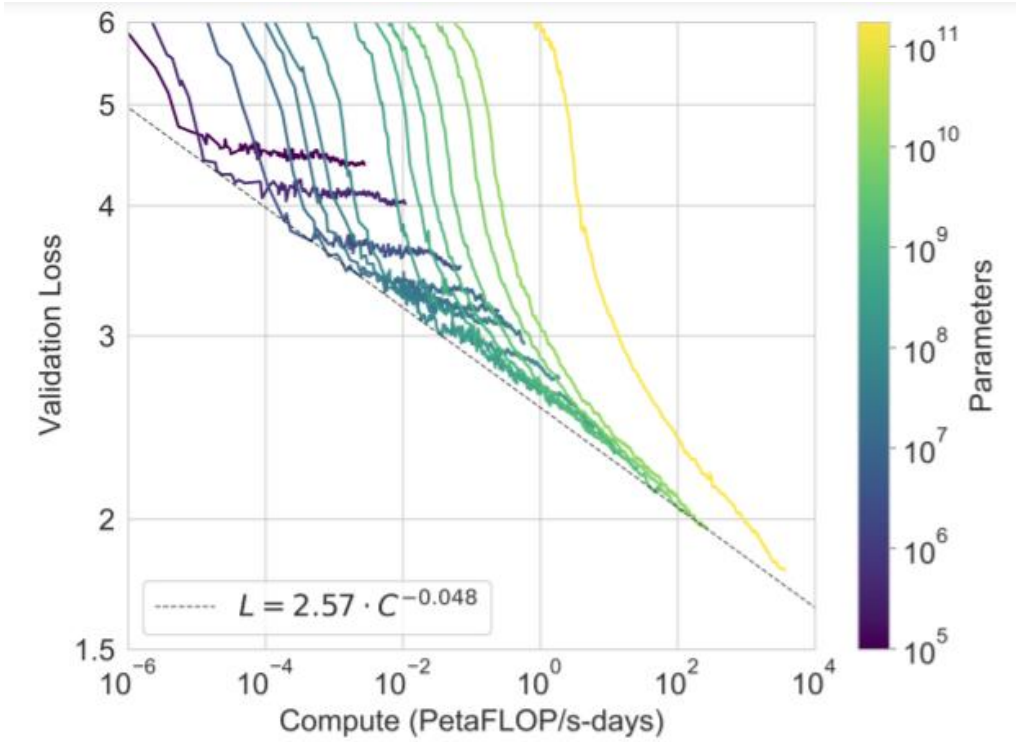
- Empirical observation: scaling up models leads to reliable gains in perplexity

为什么要缩放？缩放定律



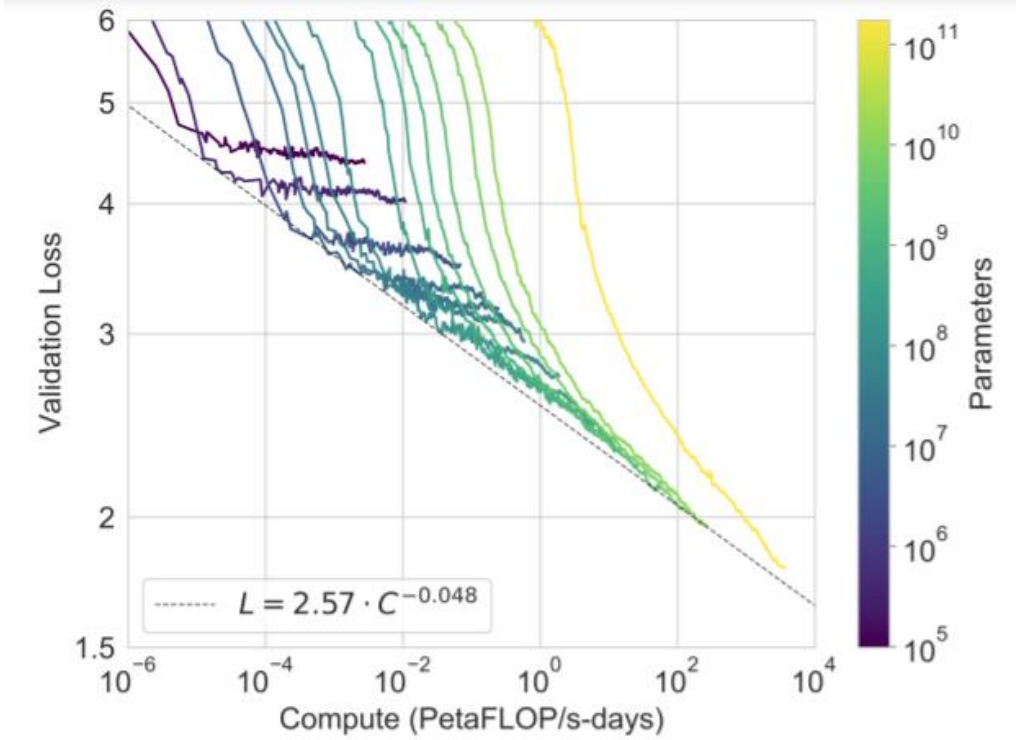
- 实证观察：扩大模型规模会可靠地降低困惑度

Scaling can help identify model size – data tradeoffs



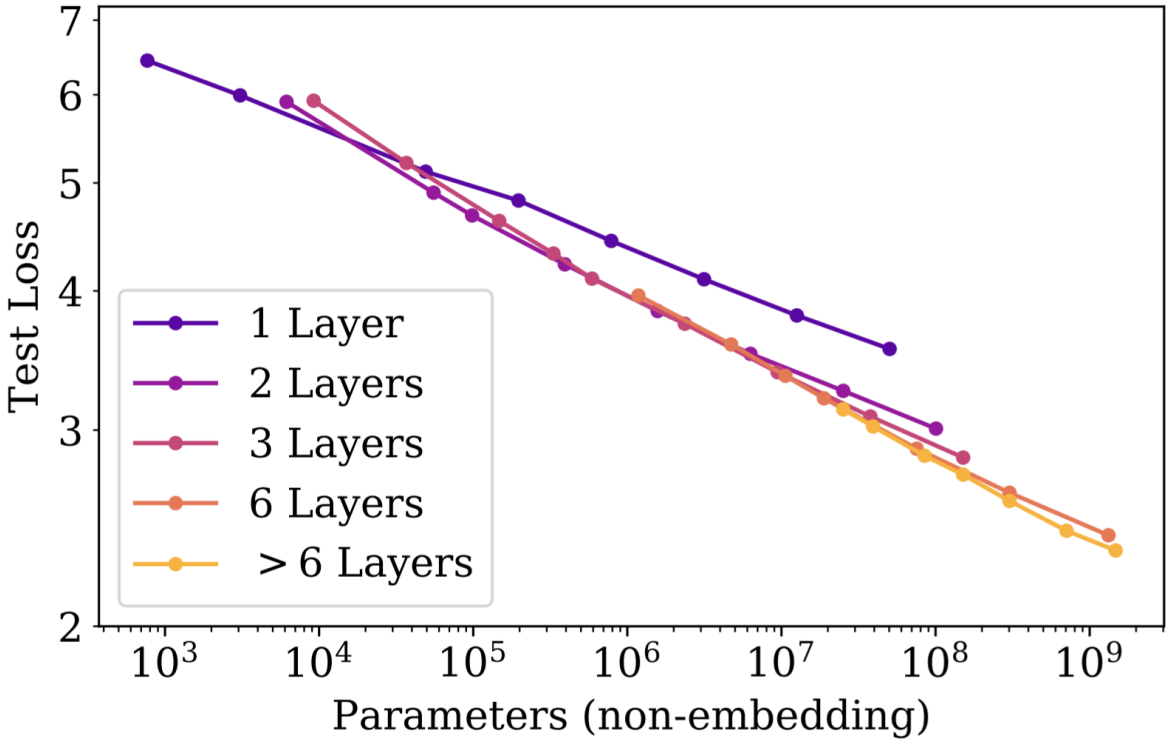
- Modern observation: train a big model that's not fully converged.

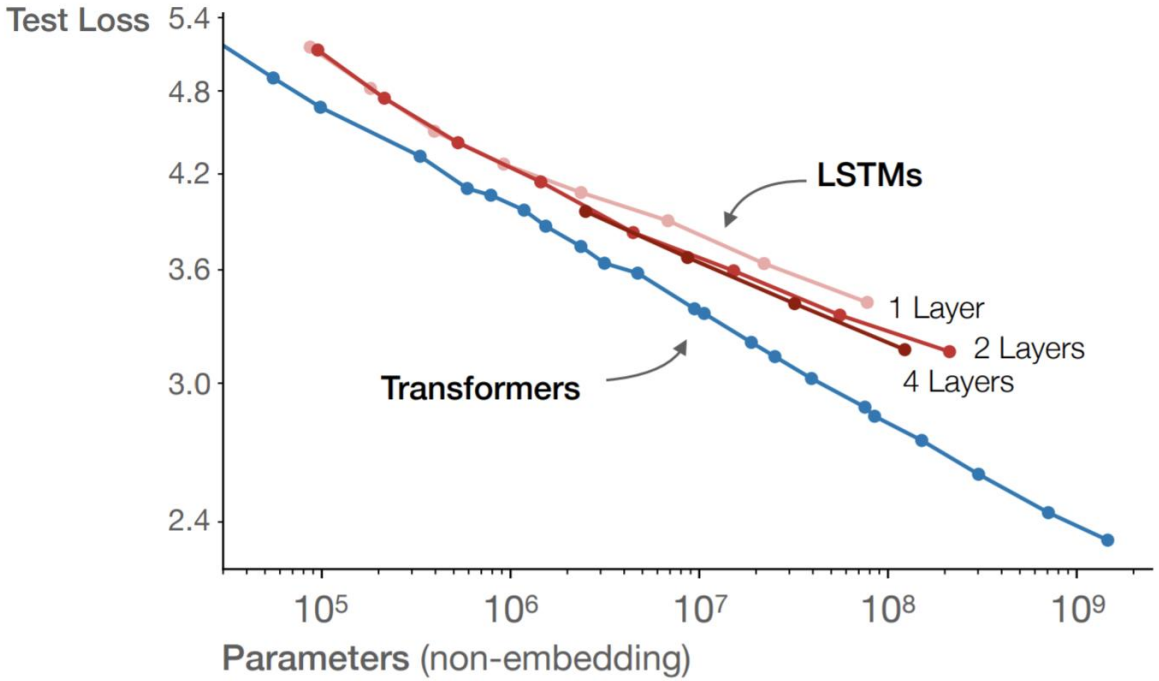
缩放可以帮助识别模型大小与数据之间的权衡



- Modern observation: train a big model that’s not fully converged.

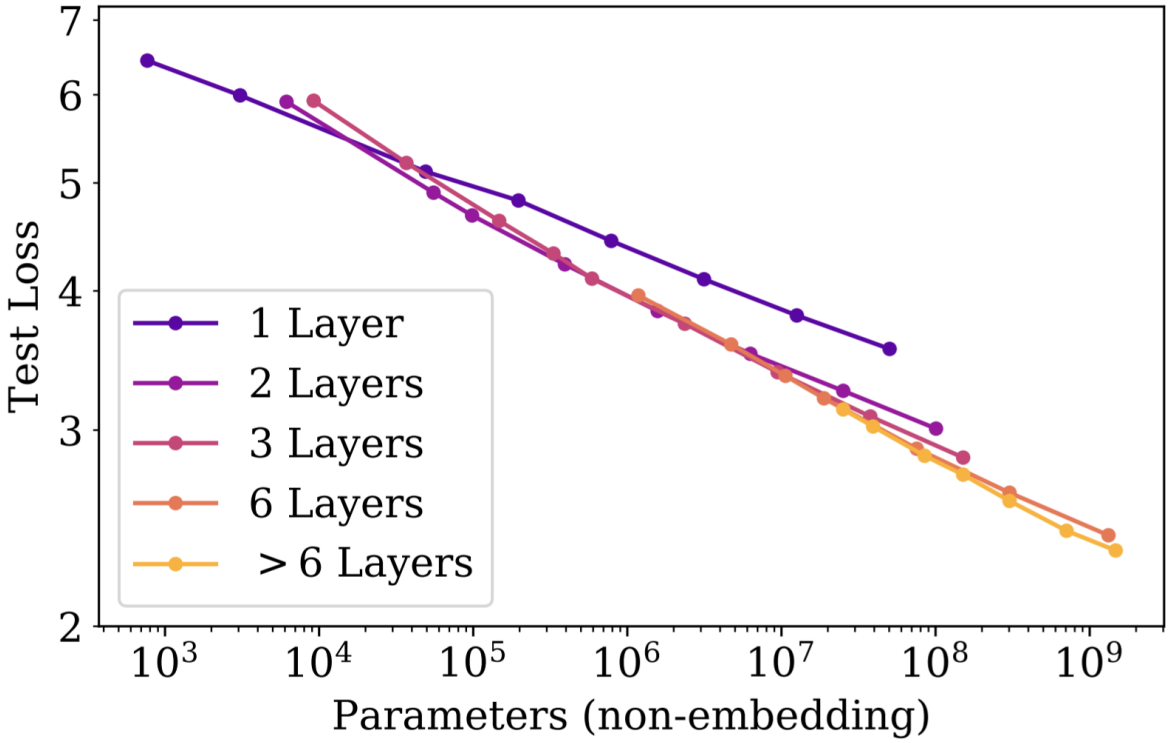
Scaling laws for many other interesting architecture decisions

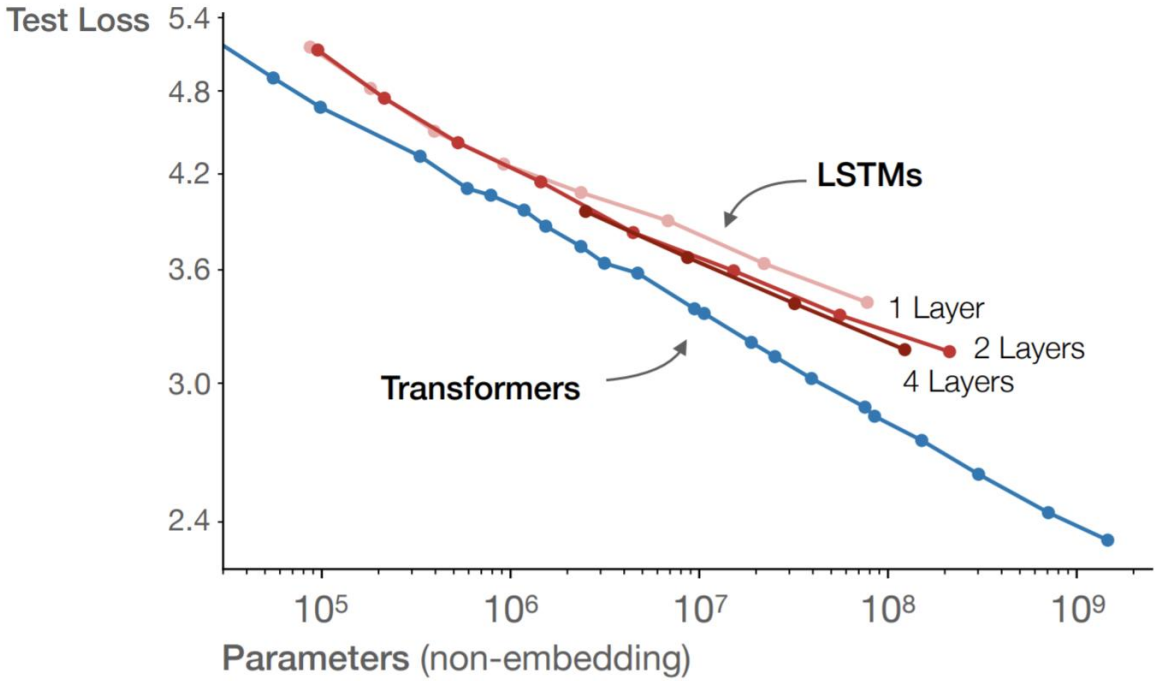




- Predictable scaling helps us make intelligent decisions about architectures etc.

许多其他有趣的架构决策的缩放定律





- 可预测的缩放帮助我们对架构等做出明智的决策。

Scaling Efficiency: how do we best use our compute

GPT-3 was **175B parameters** and trained on **300B** tokens of text.

Roughly, the cost of training a large transformer scales as **parameters*tokens**

Did OpenAI strike the right parameter-token data to get the best model? No.

Model	Size (# Parameters)	Training Tokens
LaMDA (Thoppilan et al., 2022)	137 Billion	168 Billion
GPT-3 (Brown et al., 2020)	175 Billion	300 Billion
Jurassic (Lieber et al., 2021)	178 Billion	300 Billion
Gopher (Rae et al., 2021)	280 Billion	300 Billion
MT-NLG 530B (Smith et al., 2022)	530 Billion	270 Billion
Chinchilla	70 Billion	1.4 Trillion

This 70B parameter model is better than the much larger other models!

缩放效率：我们如何最好地利用计算资源

GPT-3 was １７５０亿参数 and trained on ３００亿 tokens of text.

Roughly, the cost of training a large transformer scales as **parameters*tokens**

ＯｐｅｎＡＩ 是否找到了正确的参数 - ｔｏｋｅｎ 数据比来获得最佳模型？没有。

Model	Size (# Parameters)	Training Tokens
LaMDA (Thoppilan et al., 2022)	137 Billion	168 Billion
GPT-3 (Brown et al., 2020)	175 Billion	300 Billion
Jurassic (Lieber et al., 2021)	178 Billion	300 Billion
Gopher (Rae et al., 2021)	280 Billion	300 Billion
MT-NLG 530B (Smith et al., 2022)	530 Billion	270 Billion
Chinchilla	70 Billion	1.4 Trillion

这个 ７００亿参数的模型比其他更大的模型更好！

Outline

1. A brief note on subword modeling
2. Motivating model pretraining from word embeddings
3. Model pretraining three ways
    1. Encoders
    2. Encoder-Decoders
    3. Decoders
4. What do we think pretraining is teaching?

大纲

1. 关于子词建模的简要说明
2. 从词 embedding 出发理解模型预训练的动机
3. 三种模型预训练方式
    1. Encoder 类
    2. Encoder-Decoder 类
    3. Decoder 类
4. 我们认为预训练在教什么？

What kinds of things does pretraining teach?

There's increasing evidence that pretrained models learn a wide variety of things about the statistical properties of language. Taking our examples from the start of class:

- *Stanford University is located in _____, California.* [Trivia]
- *I put ___ fork down on the table.* [syntax]
- *The woman walked across the street, checking for traffic over ___ shoulder.* [coreference]
- *I went to the ocean to see the fish, turtles, seals, and _____.* [lexical semantics/topic]
- *Overall, the value I got from the two hours watching it was the sum total of the popcorn and the drink. The movie was ___.* [sentiment]
- Iroh went into the kitchen to make some tea. Standing next to Iroh, Zuko pondered his destiny. Zuko left the _____. [some reasoning – this is harder]
- I was thinking about the sequence that goes 1, 1, 2, 3, 5, 8, 13, 21, ____ [some basic arithmetic; they don't learn the Fibonnaci sequence]
- Models also learn – and can exacerbate racism, sexism, all manner of bad biases.

预训练教了哪些东西？

There’s increasing evidence that pretrained models learn a wide variety of things about

语言的统计特性。从课程开头的例子来看：

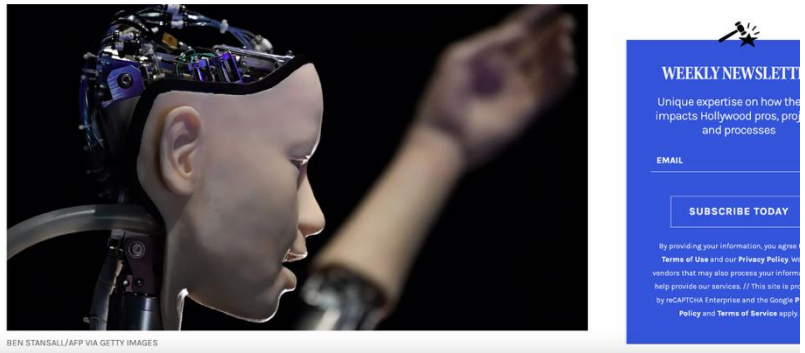
- *Stanford University is located in _____, California.* [Trivia]
- *I put ___ fork down on the table.* [syntax]
- *The woman walked across the street, checking for traffic over ___ shoulder.* [coreference]
- *I went to the ocean to see the fish, turtles, seals, and _____.* [lexical semantics/topic]
- *Overall, the value I got from the two hours watching it was the sum total of the popcorn and the drink. The movie was ___.* [sentiment]
- Iroh went into the kitchen to make some tea. Standing next to Iroh, Zuko pondered his destiny. Zuko left the _____. [some reasoning – this is harder]
- I was thinking about the sequence that goes 1, 1, 2, 3, 5, 8, 13, 21, ____ [some basic arithmetic; they don’t learn the Fibonnaci sequence]
- 模型也会学习到 —— 并可能加剧种族歧视、性别歧视等各种不良偏见。

Sometimes it also memorizes copyrighted material

AI Art Generators Spark Multiple Copyright Lawsuits

Getty and a trio of artists sued AI art generators in separate suits accusing the companies of copyright infringement for pilfering their works.

BY WINSTON CHO JANUARY 17, 2023 4:10PM



BEN STANSALL/AFP VIA GETTY IMAGES

Anthropic fires back at music publishers' AI copyright lawsuit

By Blake Brittain

January 17, 2024 3:30 PM PST · Updated 19 days ago

ARTICLE

Insights from the Pending Copilot Class Action Lawsuit

October 4, 2023

Bloomberg Law

By Daniel R. Mello, Jr.; Jenevieve J. Maerker; Matthew C. Berntsen; Ming-Tao Yang

GitHub Inc. offers a cloud-based platform that is popular among many software programmers for hosting and sharing source code, and collaborating on source code drafting. GitHub's artificial

The Times Sues OpenAI and Microsoft Over A.I. Use of Copyrighted Work

Millions of articles from The New York Times were used to train chatbots that now compete with it, the lawsuit said.

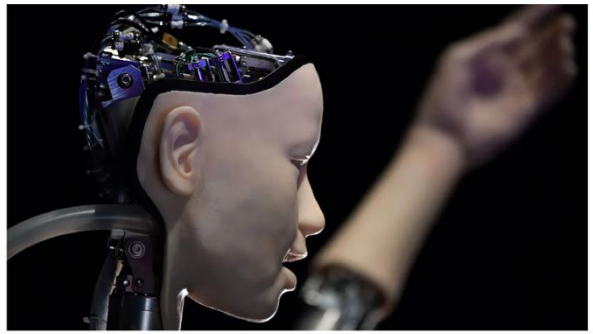
Share free access 1.3K

有时它也会记忆受版权保护的材料

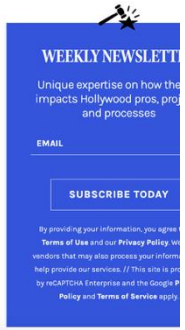
AI Art Generators Spark Multiple Copyright Lawsuits

Getty and a trio of artists sued AI art generators in separate suits accusing the companies of copyright infringement for pilfering their works.

BY WINSTON CHO JANUARY 17, 2023 4:10PM



BEN STANSALL/AFP VIA GETTY IMAGES



Anthropic fires back at music publishers' AI copyright lawsuit

By Blake Brittain

January 17, 2024 3:30 PM PST · Updated 19 days ago

ARTICLE

Insights from the Pending Copilot Class Action Lawsuit

October 4, 2023

Bloomberg Law

By Daniel R. Mello, Jr.; Jenevieve J. Maerker; Matthew C. Berntsen; Ming-Tao Yang

GitHub Inc. offers a cloud-based platform that is popular among many software programmers for hosting and sharing source code, and collaborating on source code drafting. GitHub's artificial

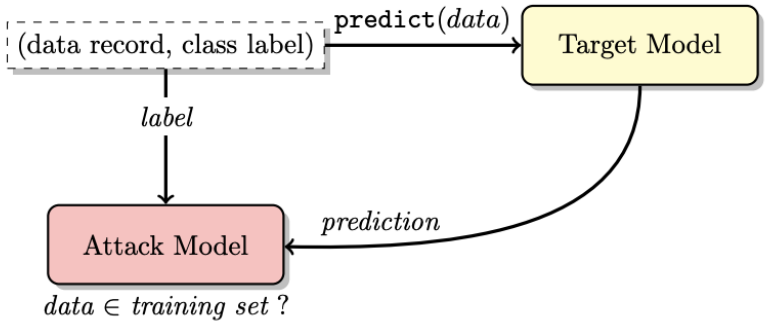
The Times Sues OpenAI and Microsoft Over A.I. Use of Copyrighted Work

Millions of articles from The New York Times were used to train chatbots that now compete with it, the lawsuit said.

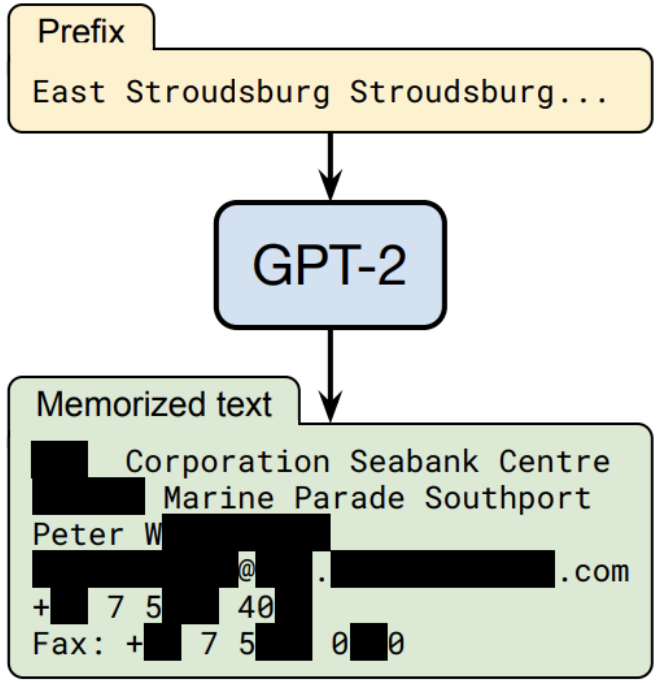
Share free access 1.3K

Sometimes it learns some things we don't want..

- *Membership inference* lets you recover parts of the training data



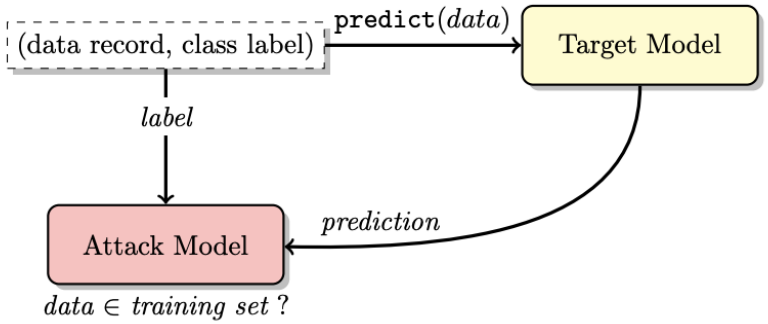
- Sometimes this training data is semi-private material from the web (addresses, emails)



Sometimes it learns some things we don't want..

- 成员推断
  训练数据的

lets you recover parts



- Sometimes this training data is semi-private material from the web (addresses, emails)