# Pretrained Models - BERT, GPT, T5

Natural Language Processing

(based on revision of John Hewitt and Colin Raffel Lectures)

## Announcement

- TA announcements (if any)...

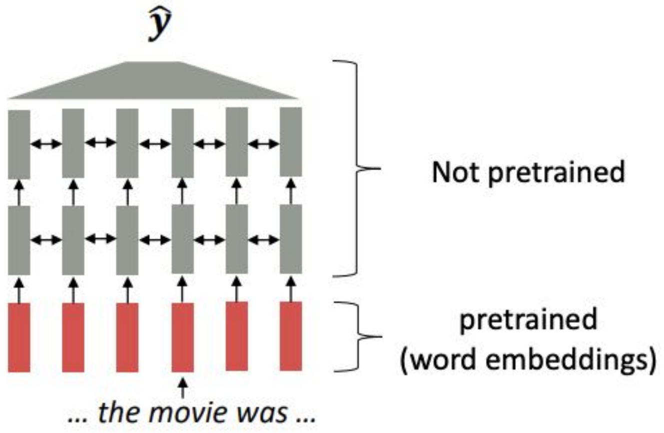
## Suggested Readings

1. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (original paper of BERT)
2. T5 Model (original paper of the T5 model; lots of stuffs about multi-task learning and transfer learning)

## Pretraining

# Before: Pre-trained word embeddings

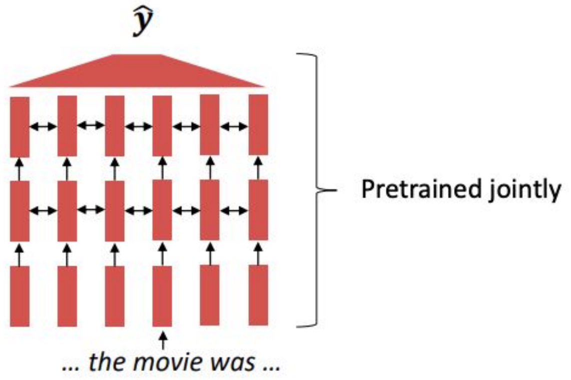
- **Before 2017**
  - Start with **pre-trained word embeddings**
  - Learn how to incorporate context in an LSTM or Transformer while training on the task
- **Some issues**
  - The training data for our **downstream task** (like question answering) must be sufficient to teach all contextual aspects of 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]

# Now: Pre-trained whole models

- All (or almost all) **parameters** in NLP networks are initialized via pretraining
- **Pretraining** is the process of developing a pretrained model that is able to generalized to different other tasks
- If we were to generalize the model to many other tasks, the **pretraining problem** should be "**representative**". (covered more later)
- New baseline, because pretrained models can capture very strong
  - **Embeddings**: Representations of language
  - **General NLP tasks**: Parameter initializations
  - **Generate text**: Probability distributions
- One big **drawback** is that pre-training takes many days and many TPUs... (but fine tuning is very quick and can work on only one GPU)

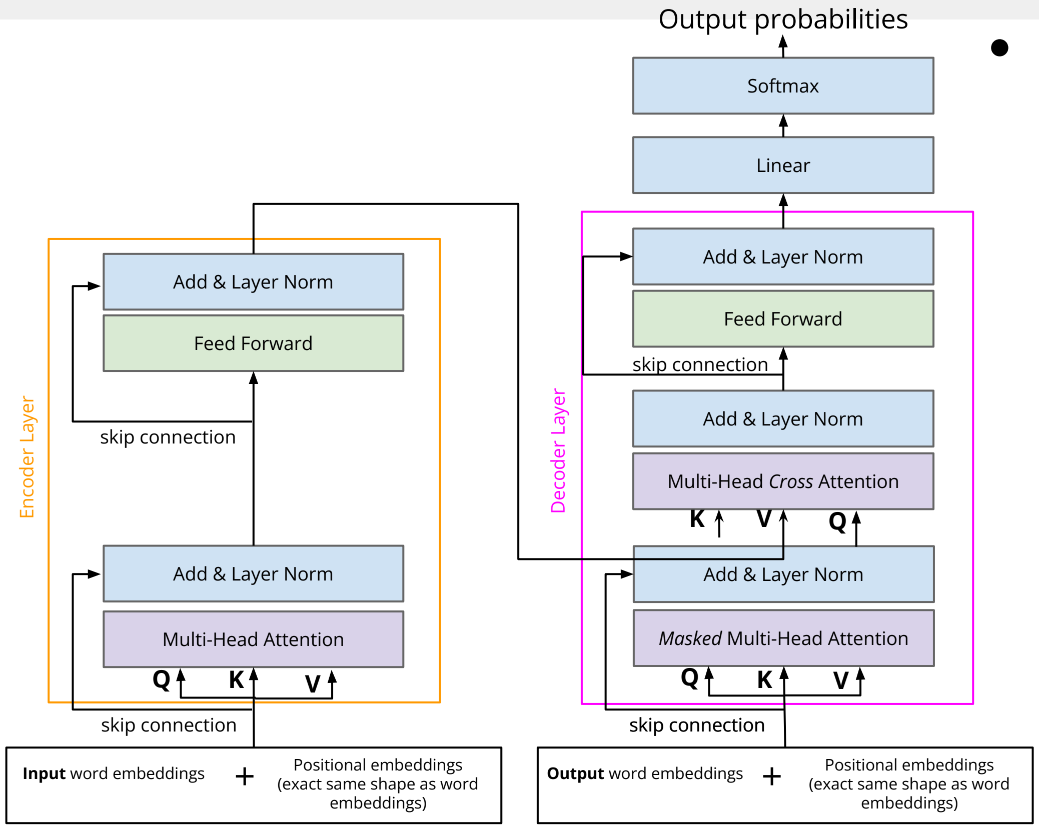


[This model has learned how to represent entire sentences through pretraining]

## Pre-training objectives

- To create **pretrained models**, we got to be smart on thinking about the **problems** that the model solve, such that it can be **generalized** to other tasks
- **Answer:** One good way is **LM!!** Many aspects can be learned in this way:
    - *Stanford University is located in ________, California. [location]*
    - *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 spatial reasoning]*
    - *I was thinking about the sequence that goes 1, 1, 2, 3, 5, 8, 13, 21, ____ [some basic arithmetic]*
    - *1 + 1 = ____ [ basic addition]*

# Pre-training approaches

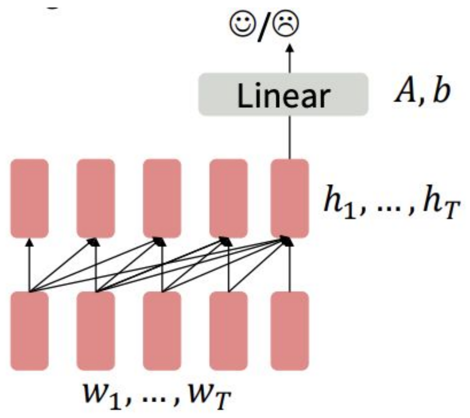


- Pre-training can be categorized into three **categories**:
  - **Decoder**
    - No cross attention unit
    - Useful for generating text or classification
    - E.g., GPT
  - **Encoder**
    - Gets bidirectional context
    - Useful for classification
    - Cannot generate text
    - E.g., BERT
  - **Encoder-Decoder**
    - Useful for generating text or classification
    - E.g., T5

## Pretraining - decoders

# Pretraining - decoders as classifier

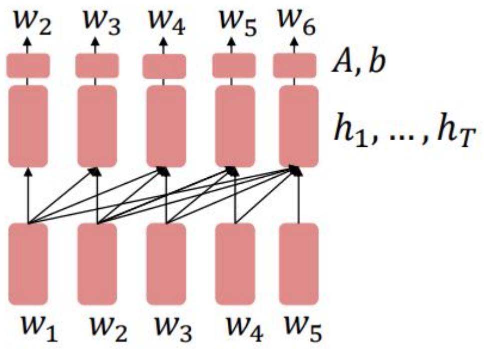
- During **pre-training**, treat it as a LM
  - Train on a huge huge corpus and on many days…
- Then we can **finetune** them by training a classifier (here Linear, with A as the W matrix and b as the bias) on the **last word's hidden state** (assumed to capture the whole sequence states)
- Note that the **linear layer hasn't been pretrained** and must be learned from scratch
- During **finetuning**:  Gradients backpropagate through the whole network or we can froze certain layers.  Normally we just let it finetune all parameters when possible.



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

## Pretraining - decoders as generator

- During **pre-training**, treat it as a LM
  - Train on a huge huge corpus and on many days…
- During **finetuning**:  provides the specific domain datasets (summarization, dialogue) and let the gradients backpropagate again
- Note that the **linear layers has already been pretrained** (i.e., A and b).  This is different from decoder as classifier situtaton.



[Note how the linear layer has been pretrained.]

## Pretraining - decoders - GPT [Radford et al. 2018]

- 2018's GPT was a big success in pretraining a decoder!
  - Transformer decoder with 12 layers
  - 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"

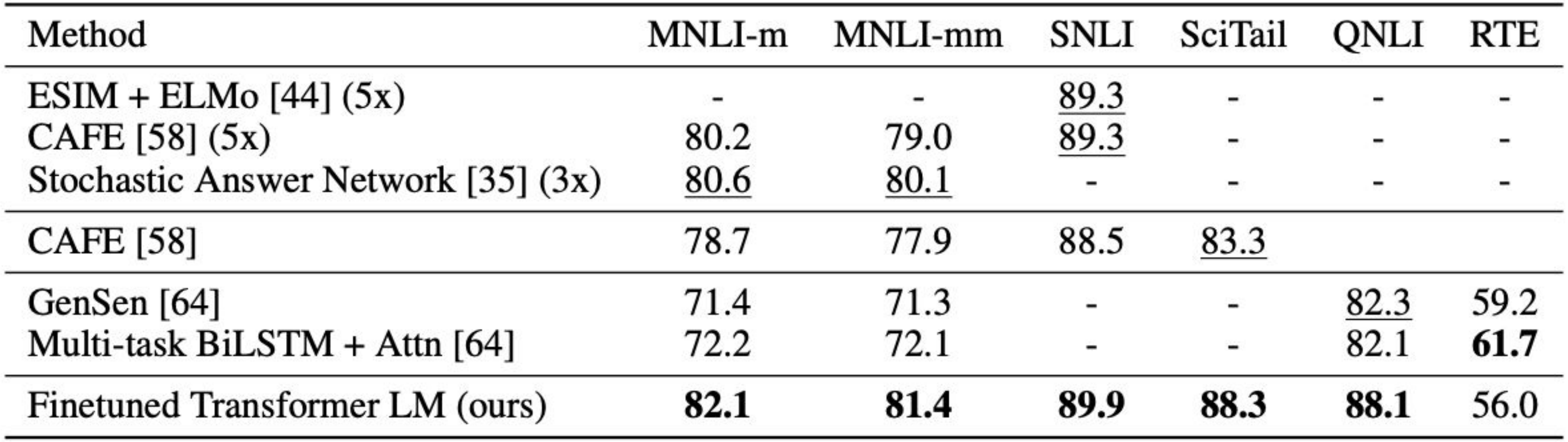
Improving Language Understanding by Generative Pre-Training, Radford et al. 2018, https://www.cs.ubc.ca/~amuham01/LING530/papers/radford2018improving.pdf

# Pretraining - decoders - GPT

- How do we format inputs to our decoder for **finetuning tasks?**
- **Natural Language Inference:** label pairs of sentences as *entailing/contradictory/neutral*. E.g., *entailing* pair
    - *Premise: The main is in the doorway*
    - *Hypothesis: The person is near the door*
- To not change the architecture, Radford et al. 2018 simply format the input like this, where a linear classifier is applied to the embedding vector of the [EXTRACT] token.
    - *[START] The main is in the doorway [DELIM] The person is near the door [EXTRACT]*

# Pretraining - decoders - GPT

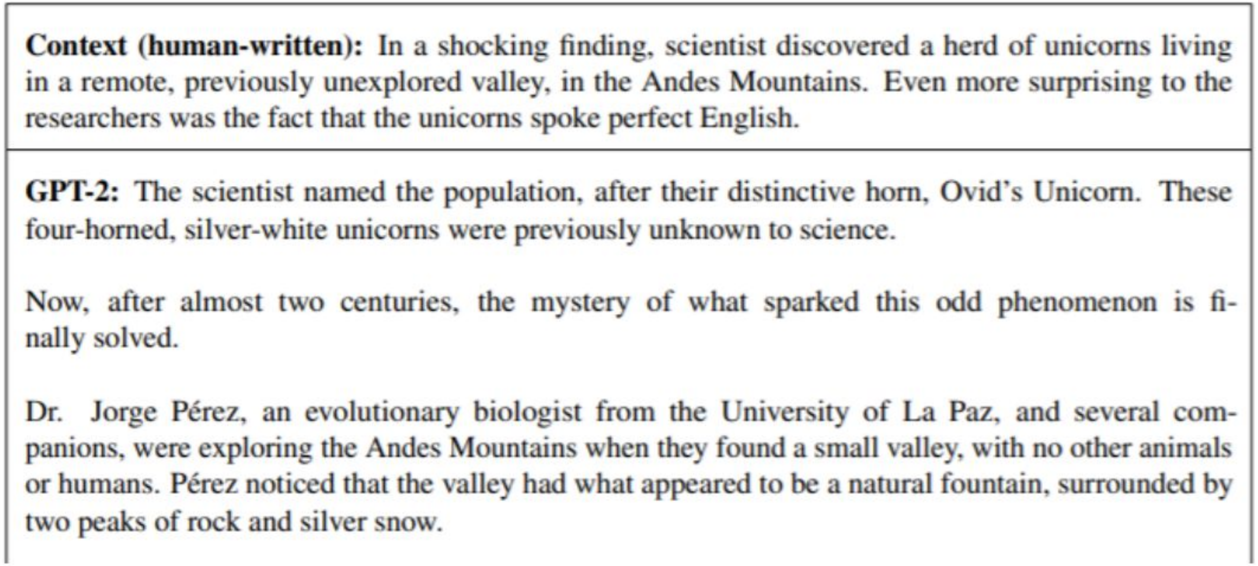
Results of GPT on natural language inference datasets



| 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 |

# Pretraining - decoders - GPT2

**GPT2** - trained on more data, was shown to produce relatively good grammatical samples



**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.

## Pretraining - encoders

## Pretraining - encoders - how to train?

Ok, decoders look great....so why bother encoder approaches?

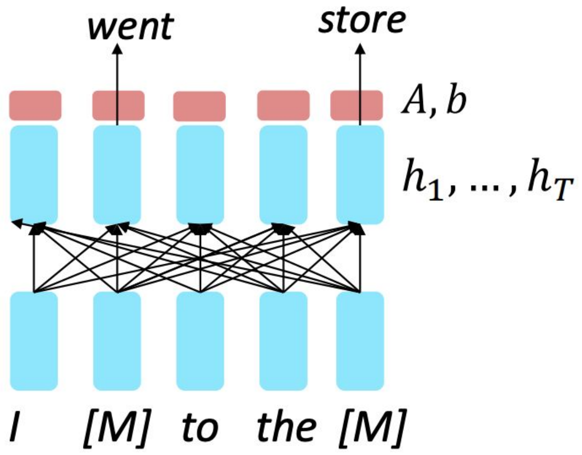
- In **decoder approaches**, it assumes **masked attention** is used, thus bidirectionality context cannot be utilized. If we want **bidirectionality**, we got to remove the masked attention unit, which turns to be basically the encoder!

- So can we pretrain our model using the encoder approaches?
  - But wait, then we cannot train using the same pretraining objective as decoder, because the future is not masked
  - So what pretraining objective to use?

# Pretraining - encoders - how to train?

Idea: Use **Masked LM**

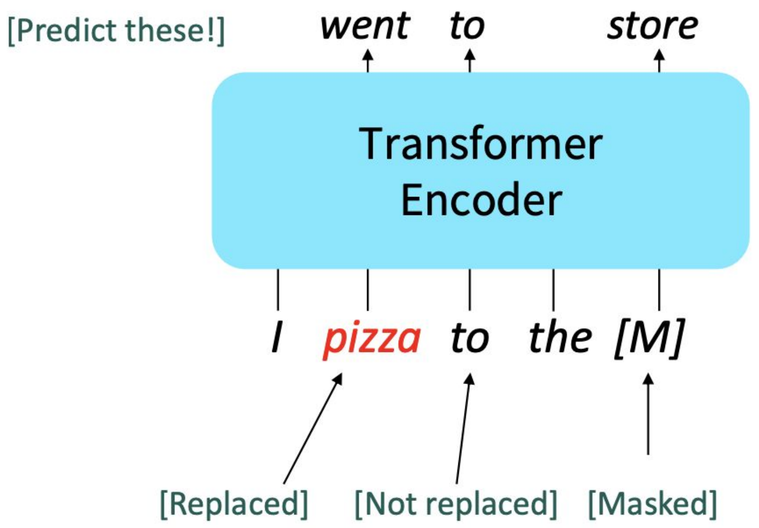
- Replace some fraction of words in the input with special [MASK] token; predict these words
- Only add loss terms from words that are “masked” out.
- Basic idea behind **BERT**



## Pretraining - encoders - BERT [Devlin et al., NAACL 2018]

Devlin et al. 2018 proposed the "**Masked LM**" objective and released the weights of a pretrained Transformer called **BERT** (Bidirectional Encoder Representations from Transformers)

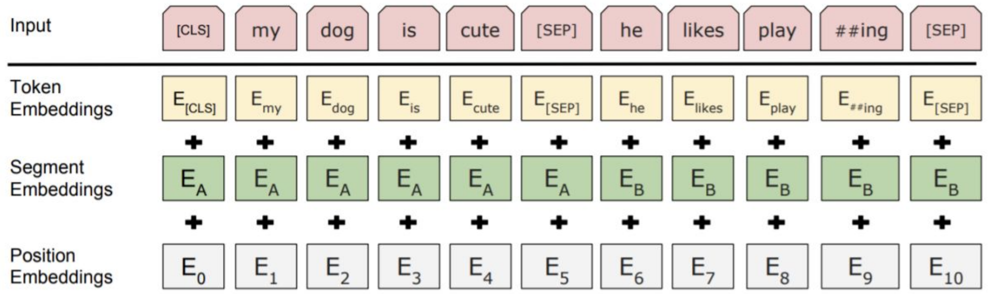
- Predict a random 15% of (sub)word tokens
  - Replace input word with [MASK] 80% of the time
  - Replace input word with a random token 10% of the time
  - Leave input word unchanged 10% of the time (but still predict it)
- Why? Doesn't let the model get complacent and not build strong representations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



BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, Devlin et al. 2018, https://arxiv.org/pdf/1810.04805.pdf

# Pretraining - encoders - BERT

- More details
  - The whole sequence length is 512
  - The pretraining input to BERT will be always be a **two separate contiguous chunks of text** (together 512)


Input
[CLS]
my
dog
is
cute
[SEP]
he
likes
play
##ing
[SEP]
Token Embeddings
$E_{[CLS]}$ $E_{my}$ $E_{dog}$ $E_{is}$ $E_{cute}$ $E_{[SEP]}$ $E_{he}$ $E_{likes}$ $E_{play}$ $E_{\#\#ing}$ $E_{[SEP]}$
Segment Embeddings
$E_A$ $E_A$ $E_A$ $E_A$ $E_A$ $E_A$ $E_B$ $E_B$ $E_B$ $E_B$ $E_B$
Position Embeddings
$E_0$ $E_1$ $E_2$ $E_3$ $E_4$ $E_5$ $E_6$ $E_7$ $E_8$ $E_9$ $E_{10}$

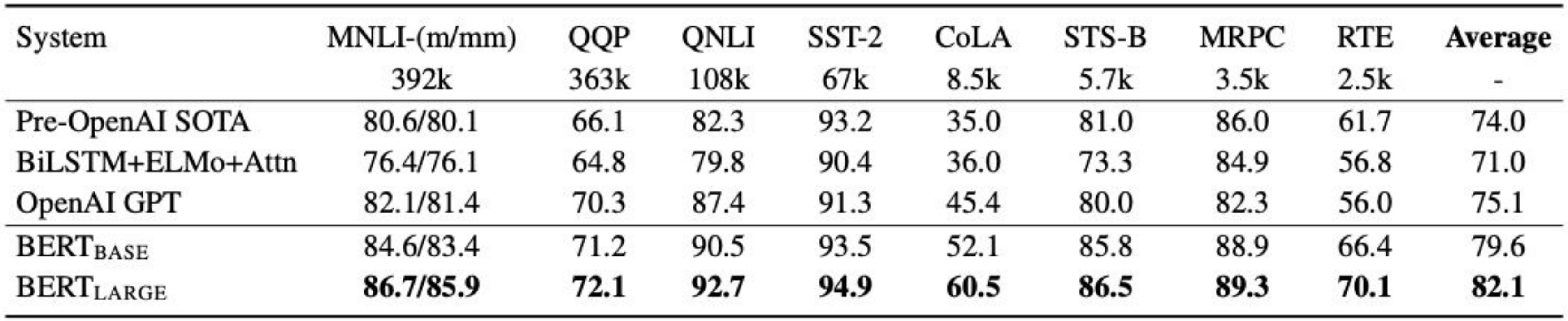
- In addition of the masked LM, simultaneously, BERT was trained to predict whether one chunk follows the other or is randomly sampled
  - Later work has argued this “next sentence prediction” is not necessary....huh?!

## Pretraining - encoders - BERT

- More details :-)
  - 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.
  - Finetuning is practical and common on a single GPU.  Yay!
    - Pretrain once, finetune many times.

# Pretraining - encoders - BERT

- BERT was massively popular and 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

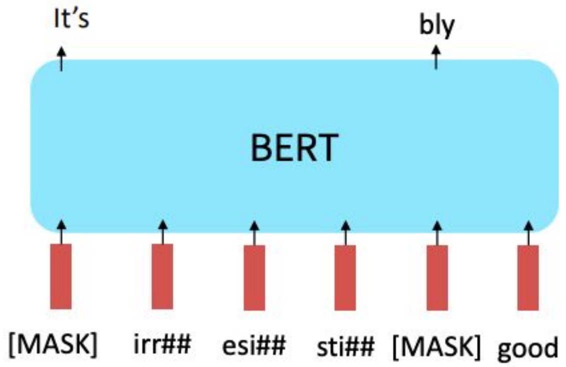


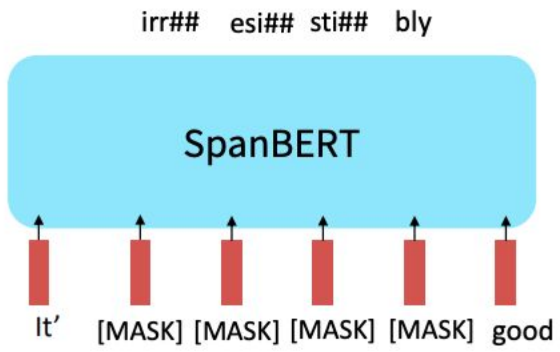
| System | MNLI-(m/mm) | QQP | QNLI | SST-2 | CoLA | STS-B | MRPC | RTE | Average |
|---|---|---|---|---|---|---|---|---|---|
| | 392k | 363k | 108k | 67k | 8.5k | 5.7k | 3.5k | 2.5k | - |
| Pre-OpenAI SOTA | 80.6/80.1 | 66.1 | 82.3 | 93.2 | 35.0 | 81.0 | 86.0 | 61.7 | 74.0 |
| BiLSTM+ELMo+Attn | 76.4/76.1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

# Extensions of BERT

Many more BERT variants like **RoBERTa**, **SpanBERT**, and others!

- **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



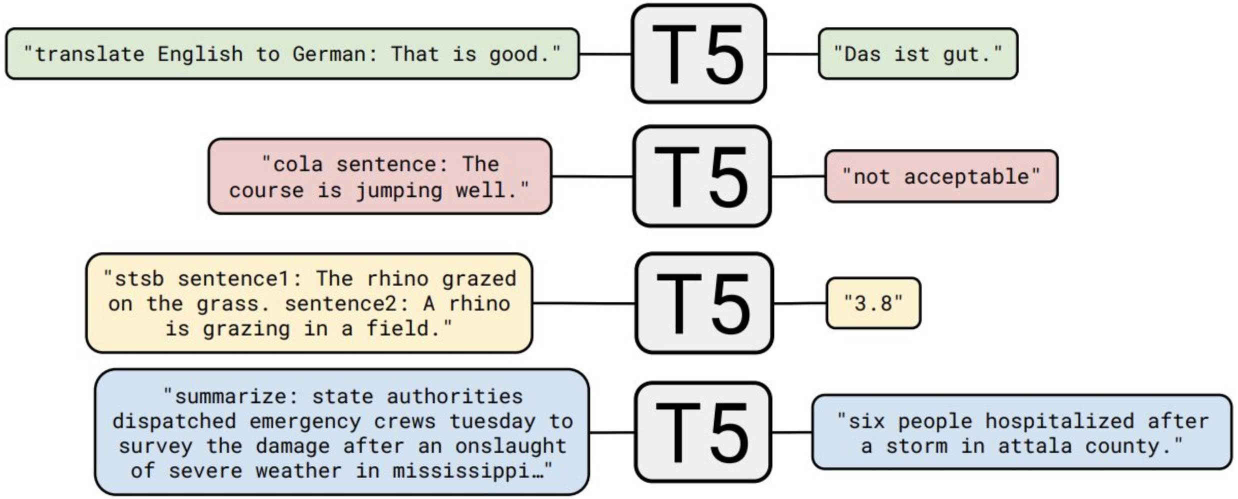


RoBERTa: A Robustly Optimized BERT Pretraining Approach, Liu et al. 2019, https://arxiv.org/abs/1907.11692
SpanBERT: Improving Pre-training by Representing and Predicting Spans, Joshi et al. 2019, https://arxiv.org/abs/1907.10529

## Pretraining - encoder-decoder

## Pretraining - encoder-decoder - T5 [Raffel et al., J.Mach.Learn 2020]

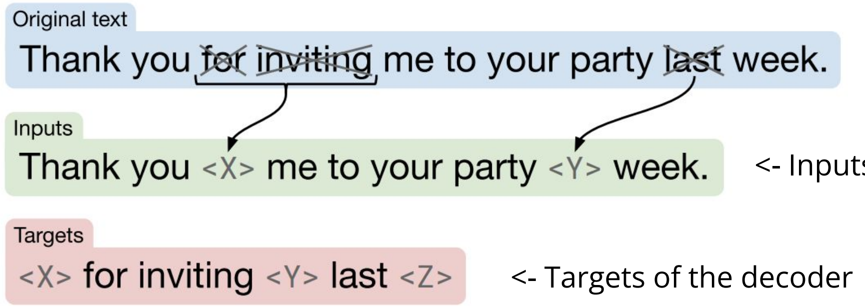
- One of the first paper to propose pretrained encoder-decoder architecture is of **Raffel et al. 2018 T5 model.** Proposed as a unified text-to-text model, treating all problems as text-to-text with prefix specifying the problem type


"translate English to German: That is good."
T5
"Das ist gut."
"cola sentence: The course is jumping well."
T5
"not acceptable"
"stsb sentence1: The rhino grazed on the grass. sentence2: A rhino is grazing in a field."
T5
"3.8"
"summarize: state authorities dispatched emergency crews tuesday to survey the damage after an onslaught of severe weather in mississippi…"
T5
"six people hospitalized after a storm in attala county."

Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer, Raffel et al., 2020, https://arxiv.org/pdf/1910.10683.pdf

# Pretraining - encoder-decoder - T5

- **Training objective**
    - Very similar to BERT
        - But instead, reconstruct only the masked tokens
            - Authors claim that this design decision is made to reduce computational cost
            - They could achieve the same performance as BERT with only ¼ of computational cost

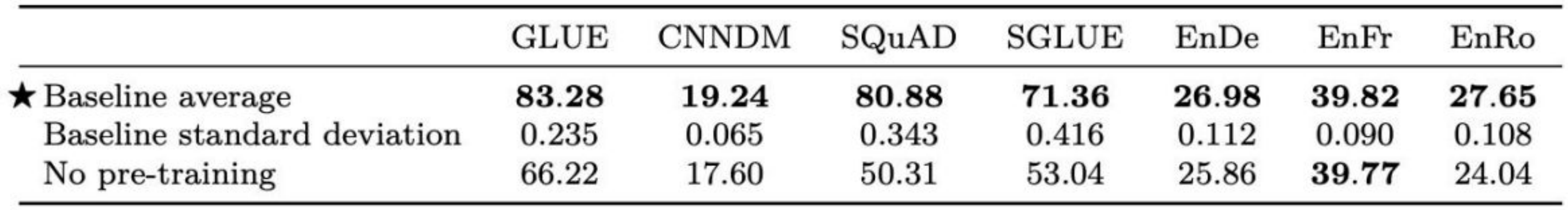

Original text
Thank you ~~for inviting~~ me to your party ~~last~~ week.
Inputs
Thank you <X> me to your party <Y> week.
Targets
<X> for inviting <Y> last <Z>

<- Inputs to the encoder

<- Targets of the decoder

# Pretraining - encoder-decoder - T5

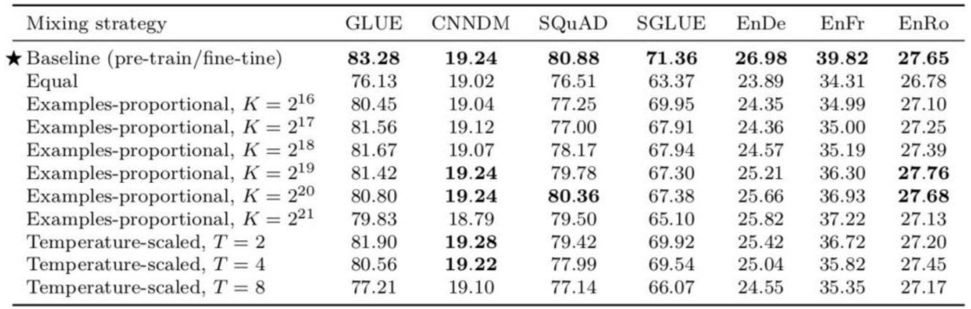
Use the pretrained model that has been trained on some C4 data (custom crawling dataset), and then after pretraining, finetune on different tasks



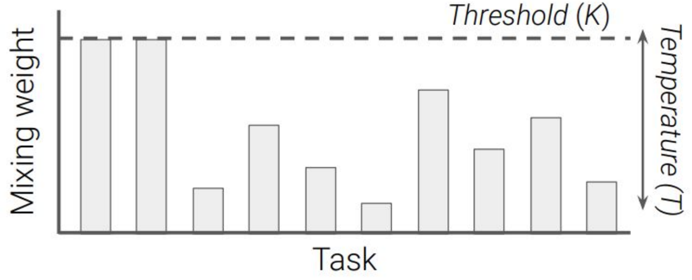
| | GLUE | CNNDM | SQuAD | SGLUE | EnDe | EnFr | EnRo |
|---|---|---|---|---|---|---|---|
| ★ Baseline average | **83.28** | **19.24** | **80.88** | **71.36** | **26.98** | **39.82** | **27.65** |
| Baseline standard deviation | 0.235 | 0.065 | 0.343 | 0.416 | 0.112 | 0.090 | 0.108 |
| No pre-training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

## Pretraining - encoder-decoder - T5

- Multi-task learning - pretraining and finetuning all the tasks everything together…the pretraining C4 tasks, the GLUE tasks, the CNNDM tasks, etc.



| Mixing strategy | GLUE | CNNDM | SQuAD | SGLUE | EnDe | EnFr | EnRo |
|---|---|---|---|---|---|---|---|
| ★ Baseline (pre-train/fine-tine) | **83.28** | **19.24** | **80.88** | **71.36** | **26.98** | **39.82** | **27.65** |
| Equal | 76.13 | 19.02 | 76.51 | 63.37 | 23.89 | 34.31 | 26.78 |
| Examples-proportional, $K = 2^{16}$ | 80.45 | 19.04 | 77.25 | 69.95 | 24.35 | 34.99 | 27.10 |
| Examples-proportional, $K = 2^{17}$ | 81.56 | 19.12 | 77.00 | 67.91 | 24.36 | 35.00 | 27.25 |
| Examples-proportional, $K = 2^{18}$ | 81.67 | 19.07 | 78.17 | 67.94 | 24.57 | 35.19 | 27.39 |
| Examples-proportional, $K = 2^{19}$ | 81.42 | **19.24** | 79.78 | 67.30 | 25.21 | 36.30 | **27.76** |
| Examples-proportional, $K = 2^{20}$ | 80.80 | **19.24** | **80.36** | 67.38 | 25.66 | 36.93 | **27.68** |
| Examples-proportional, $K = 2^{21}$ | 79.83 | 18.79 | 79.50 | 65.10 | 25.82 | 37.22 | 27.13 |
| Temperature-scaled, $T = 2$ | 81.90 | **19.28** | 79.42 | 69.92 | 25.42 | 36.72 | 27.20 |
| Temperature-scaled, $T = 4$ | 80.56 | **19.22** | 77.99 | 69.54 | 25.04 | 35.82 | 27.45 |
| Temperature-scaled, $T = 8$ | 77.21 | 19.10 | 77.14 | 66.07 | 24.55 | 35.35 | 27.17 |



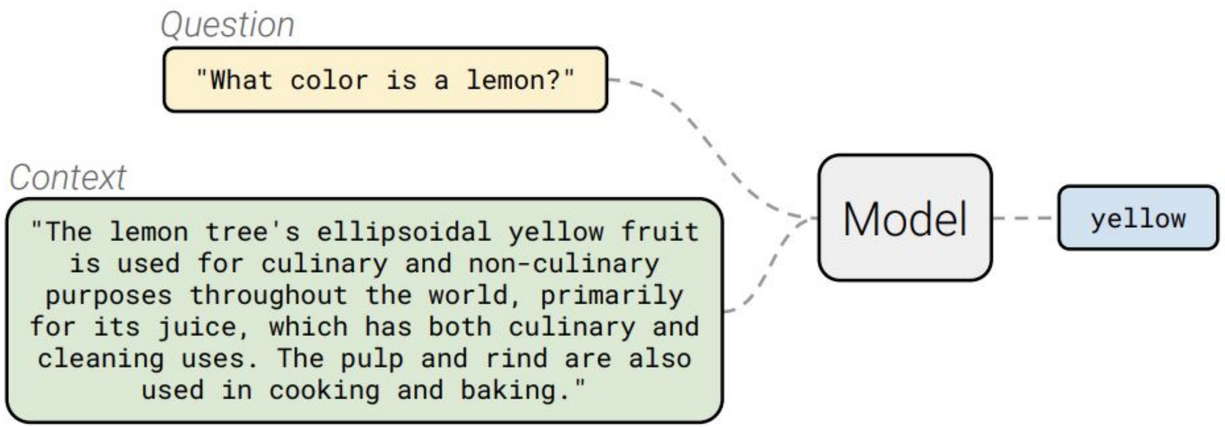
- Because the pretraining tasks are much much bigger than any other tasks, it will take forever before the downstreaming tasks gots a chance to be trained
- Solution: Introduce a hyperparameter K to decide when to stop training on the pretraining dataset
- Then we can set a temperature T to choose how many samples we want from each dataset

## Pretraining - encoder-decoder - T5

- Also tried on question-answering task on another separate paper
- Before that, let me introduce briefly type of question-answering task. It can be categorized into broadly three types:
  - Reading comprehension
  - Open-domain question answering
  - Closed-book question answering

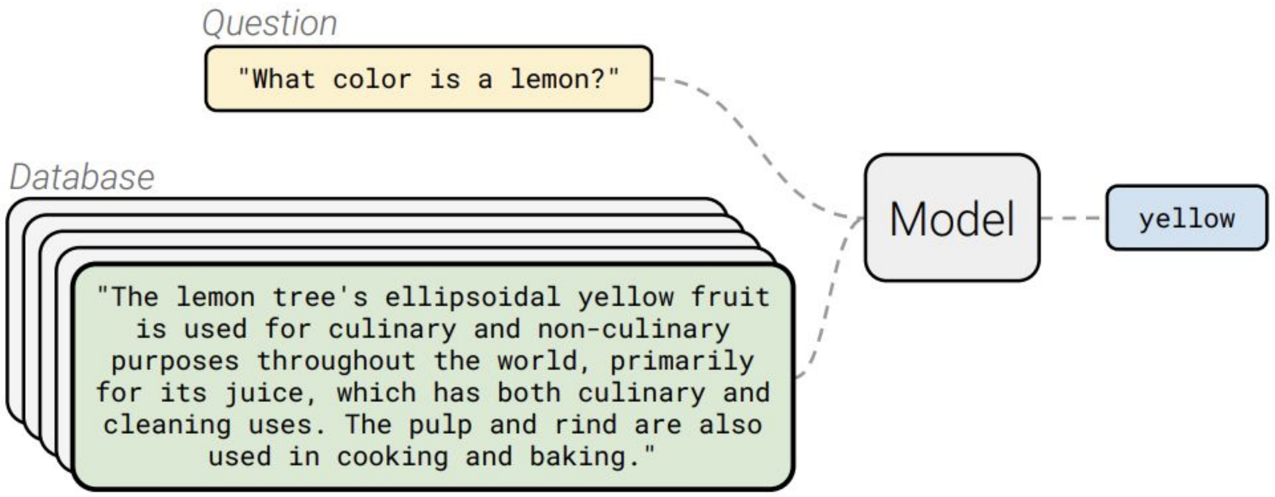
# Pretraining - encoder-decoder - T5

Reading comprehension


Question
"What color is a lemon?"
Context
"The lemon tree's ellipsoidal yellow fruit is used for culinary and non-culinary purposes throughout the world, primarily for its juice, which has both culinary and cleaning uses. The pulp and rind are also used in cooking and baking."
Model
yellow

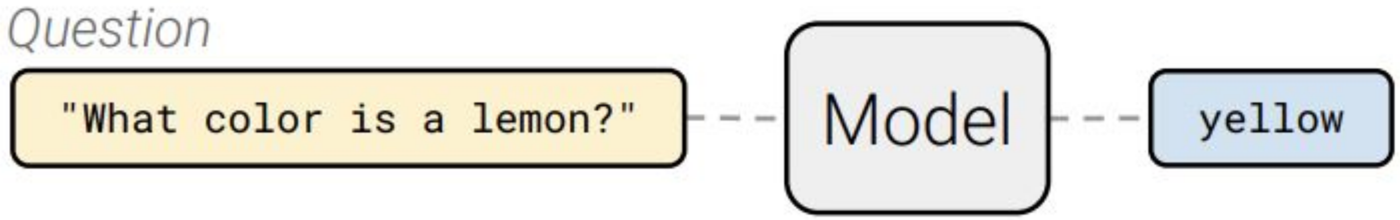
# Pretraining - encoder-decoder - T5

Open-domain question answering


Question
"What color is a lemon?"
Database
"The lemon tree's ellipsoidal yellow fruit is used for culinary and non-culinary purposes throughout the world, primarily for its juice, which has both culinary and cleaning uses. The pulp and rind are also used in cooking and baking."
Model
yellow

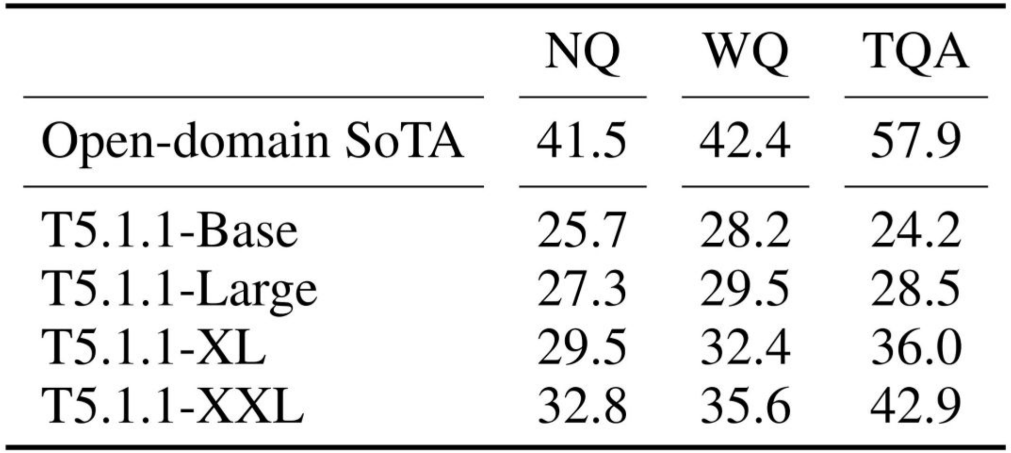
## Pretraining - encoder-decoder - T5

Closed-book question answering



# Pretraining - encoder-decoder - T5

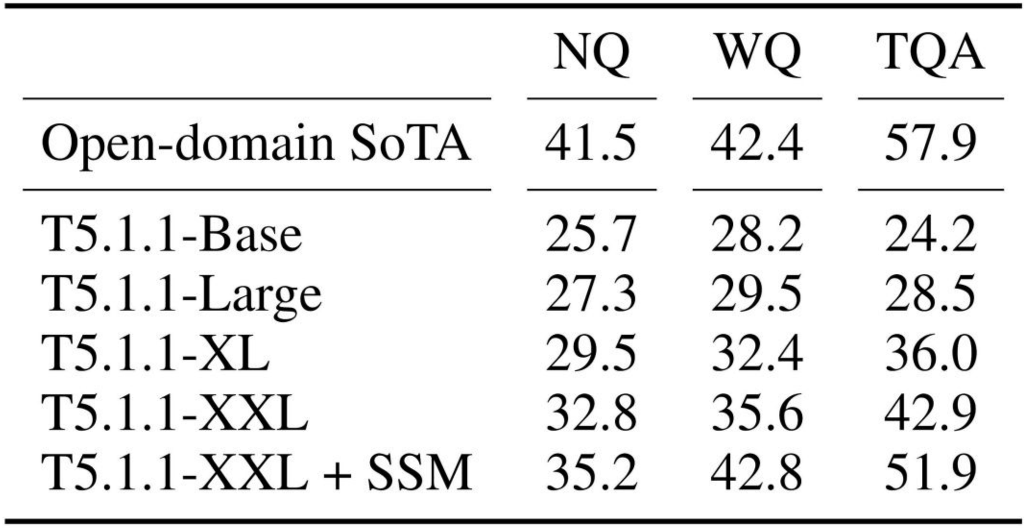
They took some question answering tasks (natural question, web question, trivia question answering datasets) and took out all the context. Metric is the ratio of the number of shared words to the total number of words in the ground truth.



| | NQ | WQ | TQA |
|---|---|---|---|
| Open-domain SoTA | 41.5 | 42.4 | 57.9 |
| T5.1.1-Base | 25.7 | 28.2 | 24.2 |
| T5.1.1-Large | 27.3 | 29.5 | 28.5 |
| T5.1.1-XL | 29.5 | 32.4 | 36.0 |
| T5.1.1-XXL | 32.8 | 35.6 | 42.9 |

# Pretraining - encoder-decoder - T5

They revised their training objectives, masking ONLY the entities (Salient Span Masking).



| | NQ | WQ | TQA |
|---|---|---|---|
| Open-domain SoTA | 41.5 | 42.4 | 57.9 |
| T5.1.1-Base | 25.7 | 28.2 | 24.2 |
| T5.1.1-Large | 27.3 | 29.5 | 28.5 |
| T5.1.1-XL | 29.5 | 32.4 | 36.0 |
| T5.1.1-XXL | 32.8 | 35.6 | 42.9 |
| T5.1.1-XXL + SSM | 35.2 | 42.8 | 51.9 |

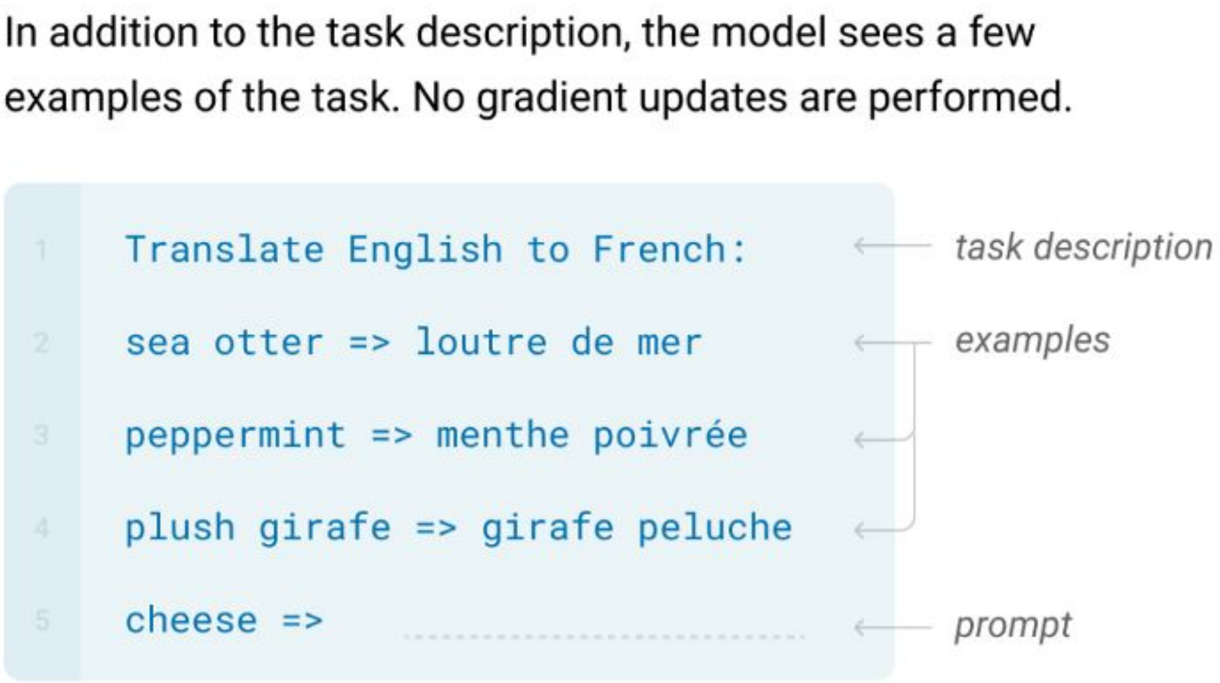
## Very large models and in-context learning

## GPT-3, in-context learning (few shots learning)

Some very large models seem to perform some kind of learning **without gradient steps** simply from examples you provide within their contexts!   Also known as **in-context learning** (very similar to few shot learning).

**GPT-3** is the canonical example of this.  The largest T5 model had 11 billion parameters.  GPT-3 has **175** billion parameters

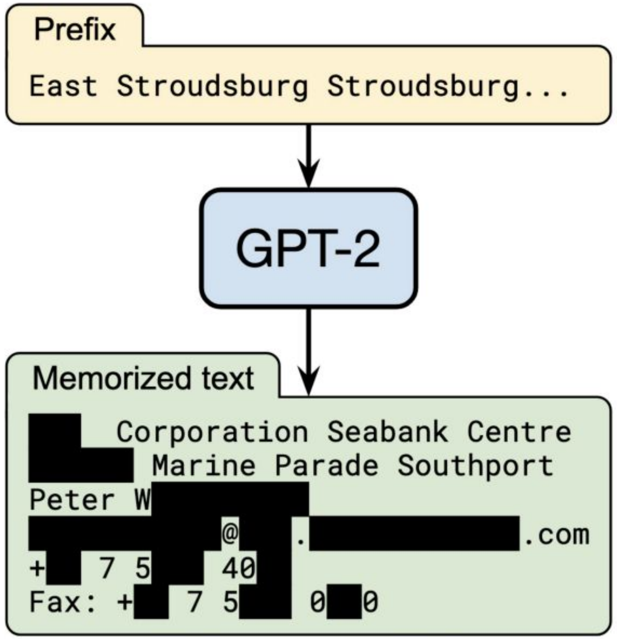
## GPT-3, in-context learning (few shots learning)


In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

```
Translate English to French:
sea otter => loutre de mer
peppermint => menthe poivrée
plush girafe => girafe peluche
cheese =>
```

## Summary

- Almost always use **pre-trained models** as one of the baselines
- **Training objective** SEEMS to be the key factor
- These models are still **not well-understood**
- **Multi-task learning** is possible with pre-trained models
- Remaining problems
    - Training cost - never enough....
    - Training objectives
    - Multi-task
    - Reasoning, knowledge, common sense
    - Output confidential information?
    - Many things to analyze, we don't have to only develop models if we are not Google or large corp (next lecture!)


Prefix
East Stroudsburg Stroudsburg...
GPT-2
Memorized text
[redacted] Corporation Seabank Centre
Marine Parade Southport
Peter W[redacted]
[redacted]@[redacted].[redacted].com
+[redacted] 7 5[redacted] 40
Fax: +[redacted] 7 5[redacted] 0[redacted]0