COMP 5152
Advanced Data Analytics

Dr. Zhiyuan Wen

Lecture 9 Data Analysis based on Text Inputs 2

Agenda

- Evaluating LMs
- Attention
- Transformer
- Pre-trained Language Models
- Large Language Models

Basic concepts of the NLP techniques in past 10 years

Logic behind the development of these techniques (why this route?)

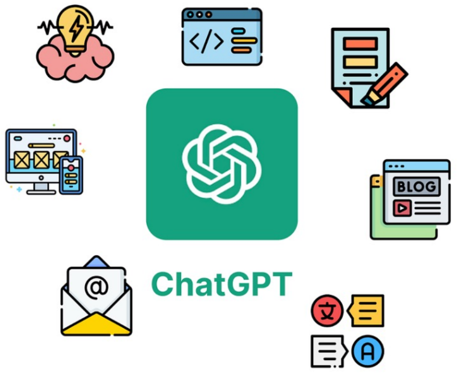
Recap

- Language Modeling Word, phrase, sub-word, Chinese character
  - predicting what token comes the next given the preceding token sequence
- N-gram language models
  - a probabilistic model to predict the likelihood of a token or a sequence given the preceding N-1 tokens
- Basic Concept of RNN
  - processing (encoding/decoding) sequences of tokens by retaining memory of previous inputs tokens

Why should we care about Language Modeling?

- Language modeling is a benchmark task that helps us measure our progress on understanding language
- Language modeling is a sub-component of many NLP tasks:
  - Machine translation
  - Speeling/grammar correction
  - Summarization
  - Dialog systems
  - …

Understanding the input, generate the appropriate output accordingly



- Language modeling is the meta task of pre-training large language models
  - Auto-regressive decoding
- **Language modeling != Language models**

Evaluating Language Models

- The standard evaluation metric for Language Models is perplexity
  - Perplexity of a sequence of T words generated by a LM:

$$\text{perplexity} = \underbrace{\prod_{t=1}^{T} \left(\frac{1}{P_{\text{LM}}(\boldsymbol{x}^{(t+1)} \mid \boldsymbol{x}^{(t)}, \ldots, \boldsymbol{x}^{(1)})} \right)}_{\text{Inverse probability of corpus, according to Language Model}}^{1/T}$$

Normalized by number of words

Lower perplexity is better

- The lower perplexity, the higher the probability of the tokens generated by the LM conform to the training data
  - The sequence generate by LM is more likely from the training data (our target of LM)
- This is equal to the exponential of the cross-entropy loss $J(\theta)$:

$$= \prod_{t=1}^{T} \left(\frac{1}{\hat{\boldsymbol{y}}^{(t)}_{\boldsymbol{x}_{t+1}}} \right)^{1/T} = \exp\left(\frac{1}{T} \sum_{t=1}^{T} -\log \hat{\boldsymbol{y}}^{(t)}_{\boldsymbol{x}_{t+1}} \right) = \exp(J(\theta))$$

Training a RNN Language Model

- **Input data:** a sequences of tokens $x^{(1)}, \ldots, x^{(T)}$
- **Training process:** compute output distribution $\hat{y}^{(t)}$ for every step t
  - i.e., predict probability distribution of $x^{(t)}$, given tokens $x^{(1)}, \ldots, x^{(t-1)}$
- **Loss function on step t:** the cross-entropy loss between predicted probability distribution $\hat{y}^{(t)}$, and the true next token $\hat{y}^{(t)}$

$$J^{(t)}(\theta) = CE(\boldsymbol{y}^{(t)}, \hat{\boldsymbol{y}}^{(t)}) = -\sum_{w \in V} \boldsymbol{y}_w^{(t)} \log \hat{\boldsymbol{y}}_w^{(t)} = -\log \hat{\boldsymbol{y}}_{\boldsymbol{x}_{t+1}}^{(t)}$$

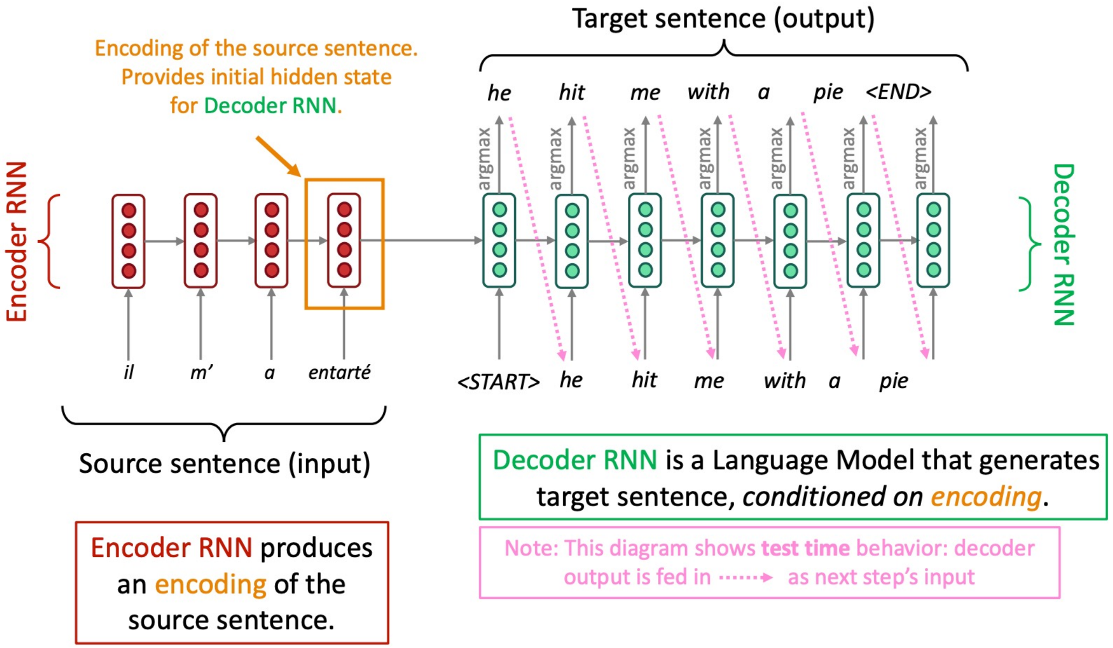
- Average loss in all steps to get the overall loss for entire training data:

$$J(\theta) = \frac{1}{T} \sum_{t=1}^{T} J^{(t)}(\theta) = \frac{1}{T} \sum_{t=1}^{T} -\log \hat{\boldsymbol{y}}_{\boldsymbol{x}_{t+1}}^{(t)}$$

- **Training objective:** minimizing the cross-entropy loss (minimizing the perplexity)

RNN for Machine Translation

- Machine Translation (MT) is the task of translating a sentence x from one language (the source language) to a sentence y in another language (the target language)
- MT is a conditional LM task
  - Sequence-to-sequence RNN model
- Model training
  - Minimizing the cross-entropy loss among tokens in the predicted sentence by the model and tokens in ground truth target sentence
- Information Bottleneck: Encoder RNN needs to capture **all information** about the source sentence
  - Is it really necessary in all scenarios?
    - For instance, a MT model does not need to be aware of the other words in the sentence when translating “boy” in the phrase “A boy is eating the banana.”
  - What if the sentence is too long? Gradient vanishing and exploding
    - Variants: Bidirectional RNN, GRU, LSTM
    - **Attention**

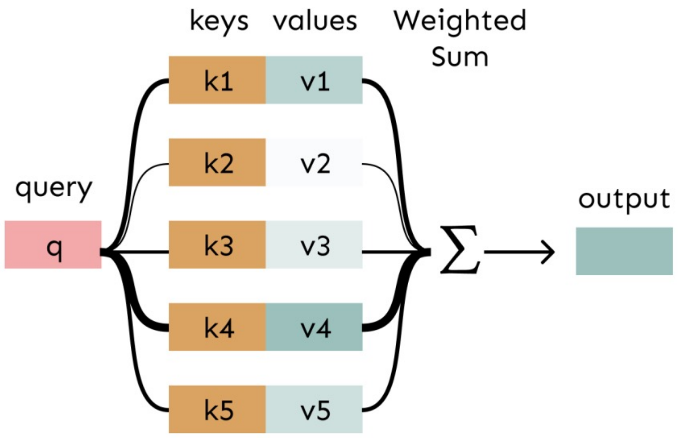


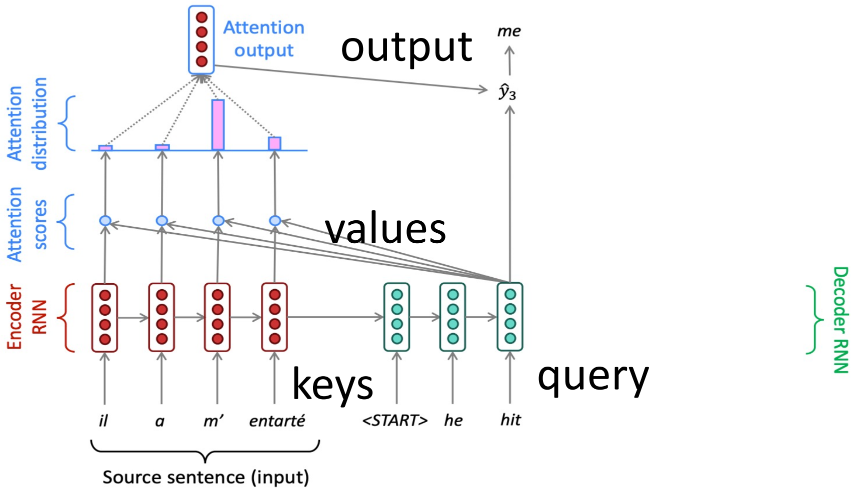
Source: https://web.stanford.edu/class/cs224n/slides/cs224n-2024-lecture06-fancy-rnn.pdf

Attention-based RNN for Machine Translation

- Attention is just a weighted average, weights (**values**) mean importance/correlation/similarity,…
- On each step of the decoder, use direct connection to the encoder to **focus on a particular part** of the source sequence

$$Attention(query, keys, values) = query * keys^T * values$$

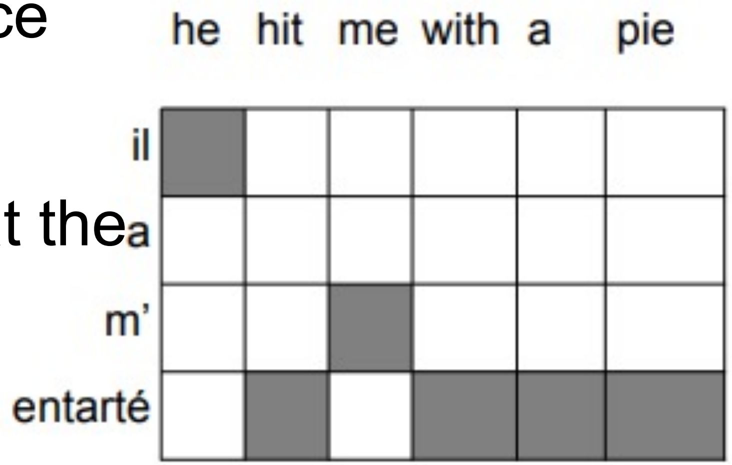




Source: https://web.stanford.edu/class/cs224n/slides/cs224n-2024-lecture06-fancy-rnn.pdf

Attention

- Attention provides more "human-like" model of the MT process
  - You can look back at the source sentence while translating, rather than needing to remember it all
- Attention solves the bottleneck problem
  - Attention allows decoder the to look directly at source
- Attention provides some interpretability
  - By inspecting attention distribution, we can see what the decoder was focus on



Language Modeling, RNN, and Attention

- Language Modeling
  - predicting what token comes next given the preceding token sequence
- RNN
  - processing sequences of tokens by retaining memory of previous inputs tokens
- RNN-LM (language models based on RNN)
  - predicting tokens conditioned on the retaining memory

When input sequence is long, it's **difficult** and **unnecessary** to remember all the information

- Attention-based RNN-LM
  - using weighted average of input, reminding RNN-LM and helping it focus on particular parts when RNN-LM forget

Attention is all you need[1]?

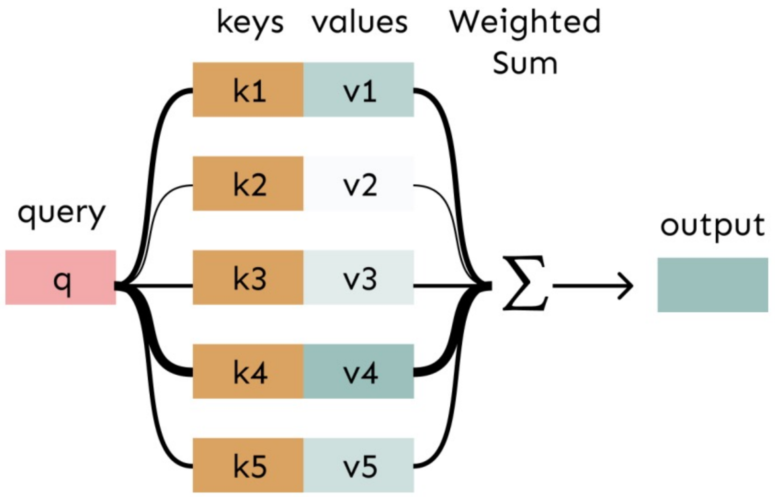
- Can we get rid of RNN?
  - If Attention can help to remind and focus, does it mean we no longer need RNN to remember?
- If only using Attention, how we encode and how we generate(decode)?
  - Self-attention
    - Encoding (we know all the input tokens):
      - consider all input tokens for each step (potential to be parallel), while RNN needs to wait
    - Decoding (we only know the generated tokens):
      - Generating t-th token, focusing on:
        - representations of the entire input from the encoder
        - the generated 1 to t-1 tokens by decoder

Illustration of self-attention for MT

[1] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[J]. Advances in neural information processing systems, 2017, 30.

Attention is all you need?

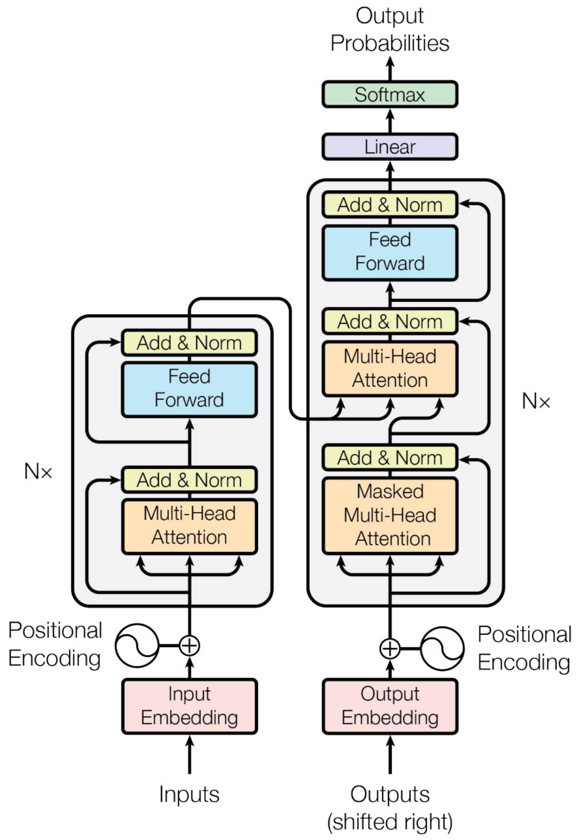
- Compared with RNN, Are there any limitations by only using Attention?
  - Doesn’t have notions of order in sequences
    - RNN encodes sequentially
    - Attention encodes all tokens together
  - No nonlinearities for deep learning magic! It’s all just weighted averages
    - RNN has the non-linear activation functions
    - Attention only has linear calculation among scales, vectors and matrices
  - Need to ensure we don’t “look at the future” for variable-length sequence
    - RNN recurrently generates with one unit
    - Attention only has the fixed length structure for the entire input



Transformer Model

- Solve the limitations by only using Attention?
  - Doesn't have notions of order in sequences
    - Positional encodings
  - No non-linearities for deep learning magic! It's all just weighted averages
    - Applying the same feedforward network to each self-attention output
  - Need to ensure we don't "look at the future"
    - Masking out the future by artificially setting attention weights to 0

    (When generating t-th token, the input will also be the entire sequences, but besides the 1 to t-1 tokens, we manually set the attention weights for t to N (if the length of the entire sequence is N) tokens as 0, rather than calculation)
  - Wait, what is multi-head?



The transformer model

Multi-head Attention

- What if we want to look in multiple parts in the sentence at once?
  - Translate the Chinese into English:

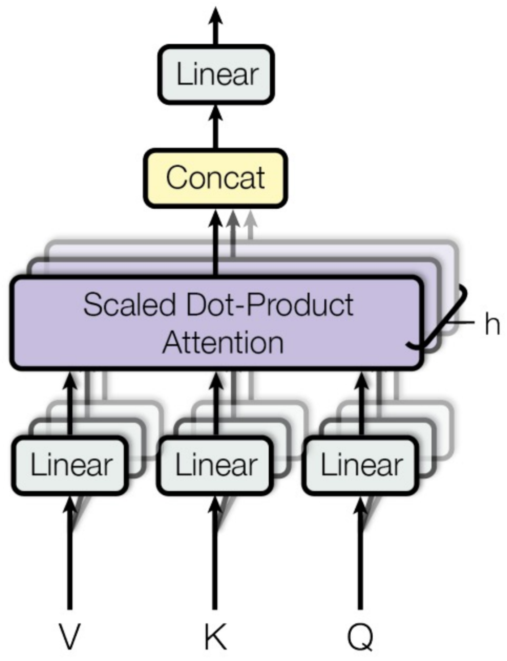
皇马的主场是伯纳乌球场

head 1

head 2

Real Madrid's home stadium is the Santiago Bernabeu Stadium
(where I just visited last Friday, btw)

- Multi-head attention captures the dependencies and relationships among different parts of the input sentence
  - Some empirical findings[1] validated different head do extract different part of information for the same input sentence



Multi-head attention

[1] Vig J. A multiscale visualization of attention in the transformer model[J]. arXiv preprint arXiv:1906.05714, 2019.

Performance of Transformer Model

- Better translation results with lower training cost

Table 2: The Transformer achieves better BLEU scores than previous state-of-the-art models on the English-to-German and English-to-French newstest2014 tests at a fraction of the training cost.

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [15]	23.75			
Deep-Att + PosUnk [32]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [31]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [8]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [26]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [32]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [31]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [8]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	$\mathbf{3.3 \cdot 10^{18}}$	
Transformer (big)	**28.4**	**41.0**	$2.3 \cdot 10^{19}$	

With Transformer, Pre-training is coming…

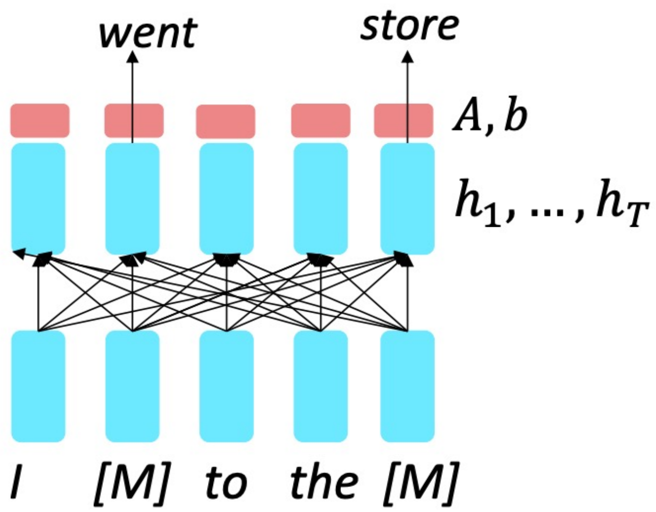
- Back to 2018, what are the main issues in NLP we talked about…
  - Models are powerful for specific tasks, but different tasks require training different models
  - Text data is everywhere, but annotated data is very limited, manual labeling is labor-intensive
- Can we train a large model which
  - can process large-scale datasets
  - is supervised by unlabeled data
  - can be easily adapted to different NLP tasks

With Transformer, Pre-training is coming…

- What pre-training task/data?
  - Some tasks doesn’t require annotated data (self-supervised), the data will be everywhere
    - Language Modeling?...
- What model?
  - Transformers’ parallelizability allows processing multiple long sequences simultaneously
- How to use?
  - Preserving parameters of the pre-trained large model and fine-tuning it with small amount of annotated data for specific tasks
- Why it works?
  - Training NN models for specific tasks also requires the abilities pre-trained on general corpus

BERT: **B**idirectional **E**ncoder **R**epresentations from **T**ransformers[1]

- Pre-training tasks
  - **Masked** Language Modeling (Encoder model)
    - replace some fraction of words in the input with a special [MASK] token; predict these words.
      - Self-supervised
      - bidirectional context
  - Next Sentence Prediction (less useful)
    - predict whether a pair of sentences in a given text are consecutive or not
      - Self-supervised



[1] Devlin J, Chang M W, Lee K, et al. Bert: Pre-training of deep bidirectional transformers for language understanding[J]. arXiv preprint arXiv:1810.04805, 2018.

BERT: **B**idirectional **E**ncoder **R**epresentations from **T**ransformers

- Data
  - BooksCorpus (800 million words)
  - English Wikipedia (2,500 million words)
- Model
  - BERT-base: 12 layers transformer, 768-dim hidden states, 12 attention heads, 110 million params
  - BERT-large: 24 layers transformer, 1024-dim hidden states, 16 attention heads, 340 million params
- Computation Resource
  - pretrained with 64 TPU chips for a total of 4 days (TPUs are special tensor operation acceleration hardware)

BERT: **B**idirectional **E**ncoder **R**epresentations from **T**ransformers

- Performance
  - finetuning BERT led to new state-of-the-art results on a broad range of tasks

Decoder model
Pre-trained with LM

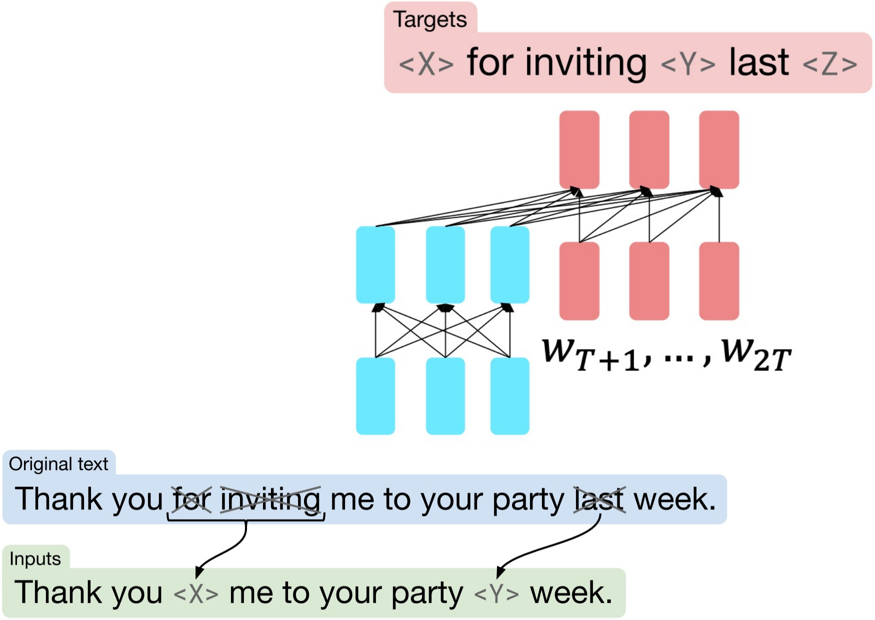
System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Average**
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
$\boxed{\text{OpenAI GPT}}$	82.1/81.4	70.3	87.4	91.3	45.4	80.0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

Table 1: GLUE Test results, scored by the evaluation server (https://gluebenchmark.com/leaderboard). The number below each task denotes the number of training examples. The "Average" column is slightly different than the official GLUE score, since we exclude the problematic WNLI set.[8] BERT and OpenAI GPT are single-model, single task. F1 scores are reported for QQP and MRPC, Spearman correlations are reported for STS-B, and accuracy scores are reported for the other tasks. We exclude entries that use BERT as one of their components.

  - but… these tasks are all about NLU (classification, sequence labeling…)
  - Remember the MT task? We used sequence-to-sequence model with RNN/Transformer

T5: Exploring the limits of transfer learning with a unified text-to-text transformer[1]

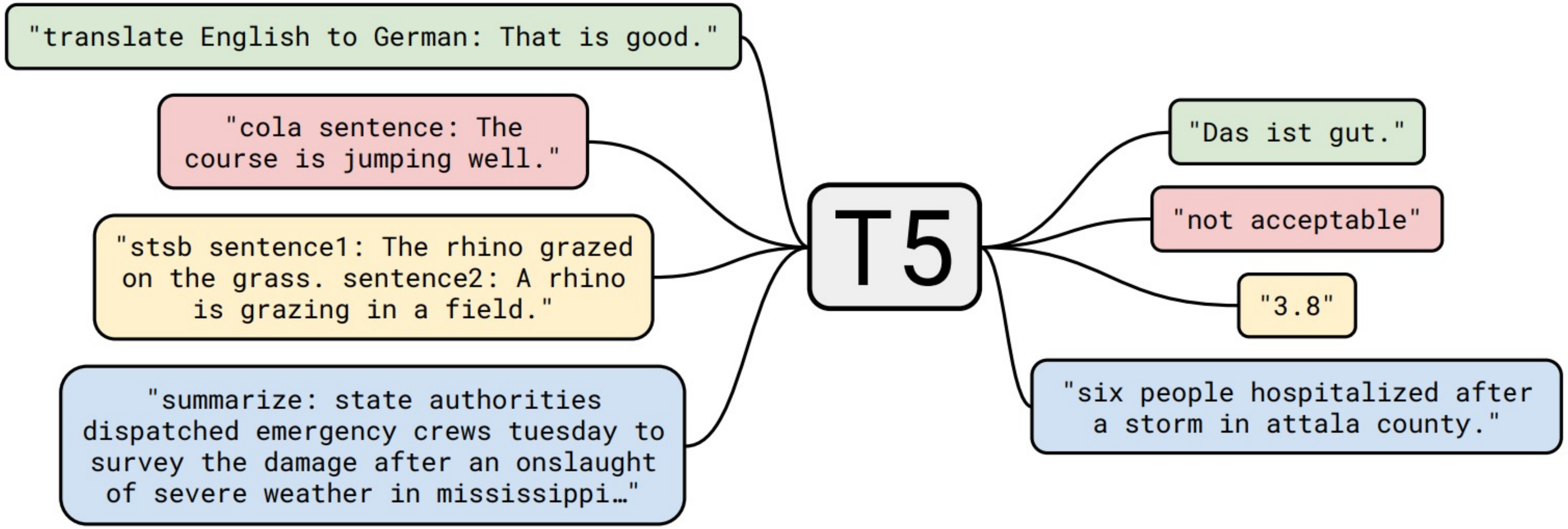
- Core idea:
  - Encoder benefits from bidirectional context
  - Decoder trained the whole model through language modeling
- Pre-training task:
  - **Span Corruption:** Replace different-length spans from the input with unique placeholders; decode out the spans that were removed



[1] Raffel C, Shazeer N, Roberts A, et al. Exploring the limits of transfer learning with a unified text-to-text transformer[J]. Journal of machine learning research, 2020, 21(140): 1-67.

T5: Exploring the limits of **t**ransfer learning with a unified **t**ext-**t**o-**t**ext **t**ransformer

- T5 is easily to be fine-tuned for many tasks with **instructive prefixes** (both NLU and NLG tasks)



- If you read the T5 paper, you will find it is a technical report of try-errors…
  - Research is not easily, and sometimes costly

GPT:Improving Language Understanding by Generative Pre-Training[1]

- What if we only use Language Modeling for pre-training?
  - This idea occurs earlier than BERT
  - Only decoder is feasible for LM
- Model
  - Transformer decoder with 12 layers, 117M parameters
  - 768-dimensional hidden states, 3072-dimensional feed-forward hidden layers
- Data
  - BooksCorpus: over 7000 unique books
- Performance
  - Still focused on NLU tasks, achieved State-of-the-art on Natural Language Inference tasks after fine-tuning

[1] Radford A, Narasimhan K, Salimans T, et al. Improving language understanding by generative pre-training[J]. 2018.

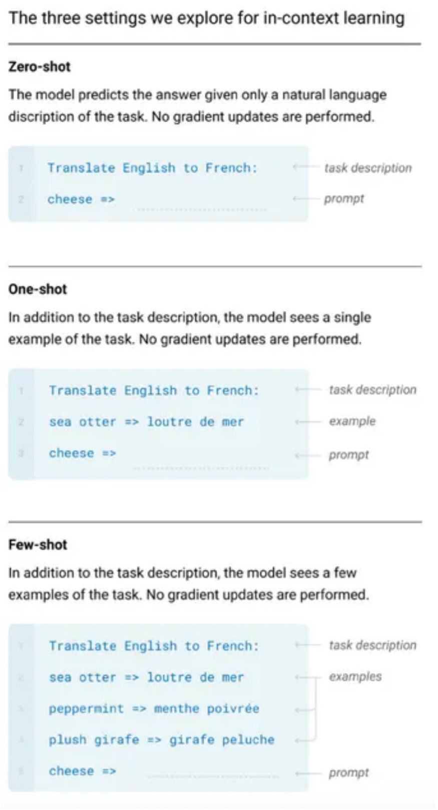
GPT-2: Language Models are Unsupervised Multitask Learners[1]

- Language Modeling:
  - predicting what token comes next given the preceding token sequence
  - generating text with given the preceding context
- GPT-2: similar architecture, larger version, trained on more data
  - produce relatively **convincing** samples of natural language (**with low perplexity**)
    - Continue story
    - Writing emails
    - …
  - but… it looked like only generates what have seen in the training data
    - Although long and fluent
    - Not always logically correct
    - Not always as our wishes

[1] Radford A, Wu J, Child R, et al. Language models are unsupervised multitask learners[J]. OpenAI blog, 2019, 1(8): 9.

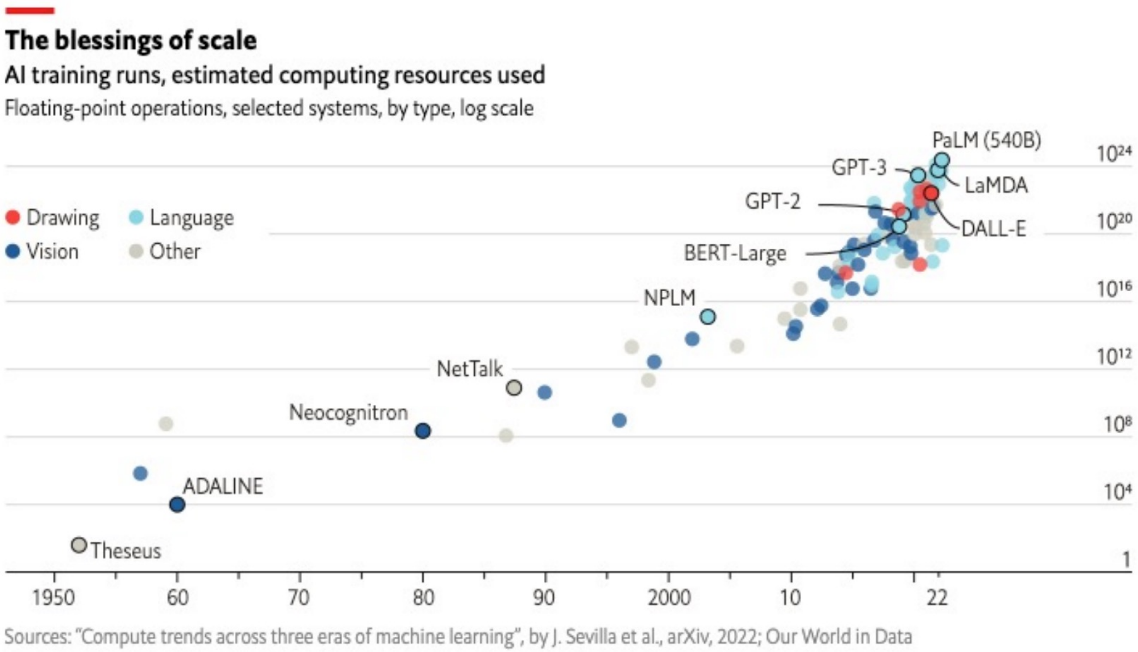
GPT-3: Language models are few-shot learners[1]

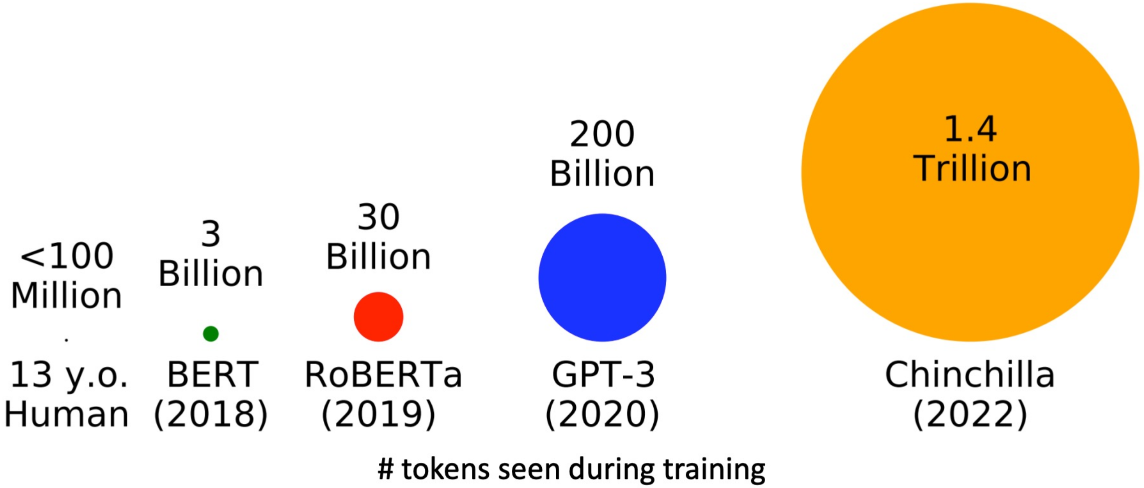
- Still pre-training on Language Modeling (see how important LM is…)
- Model size became huge…
  - The largest T5 model had 11 billion parameters
  - GPT-3 has 175 billion parameters
- Some magic happened: in-context learning
  - When apply GPT-3 on some specific tasks, you no longer need fine-tuning (you are also probably no longer able to…)
  - You just describe your task, shown one/several examples if you like, model can conduct your tasks, **without changing any model parameters**
  - **OpenAI → CloseAI,** GPT-3 and models afterwards are no longer open-sourced



[1] Brown T, Mann B, Ryder N, et al. Language models are few-shot learners[J]. Advances in neural information processing systems, 2020, 33: 1877-1901.

Models are larger, training data is more





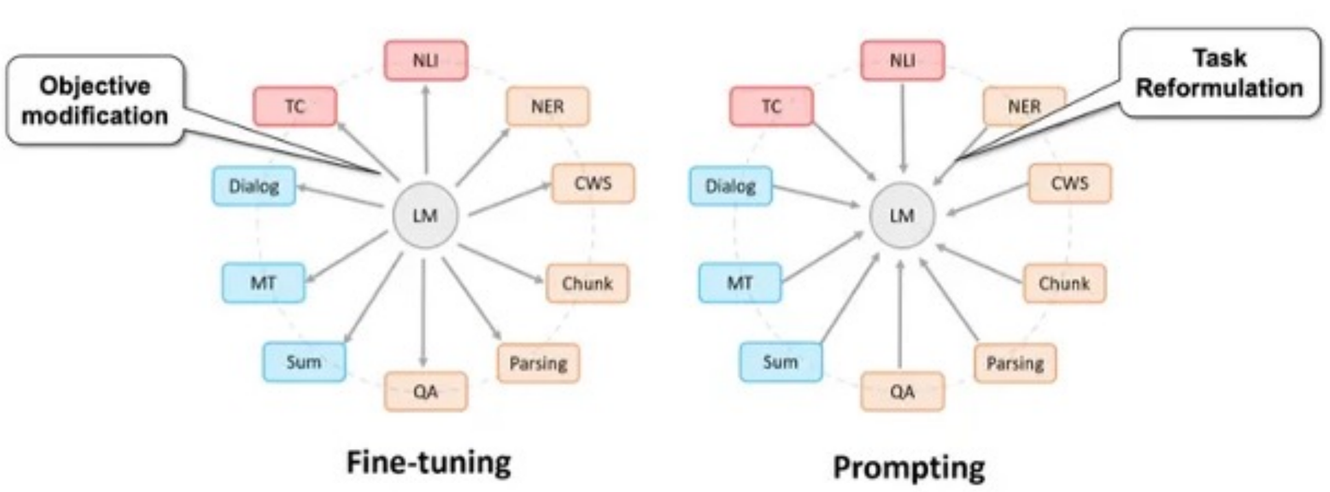
Models are larger, training data is more

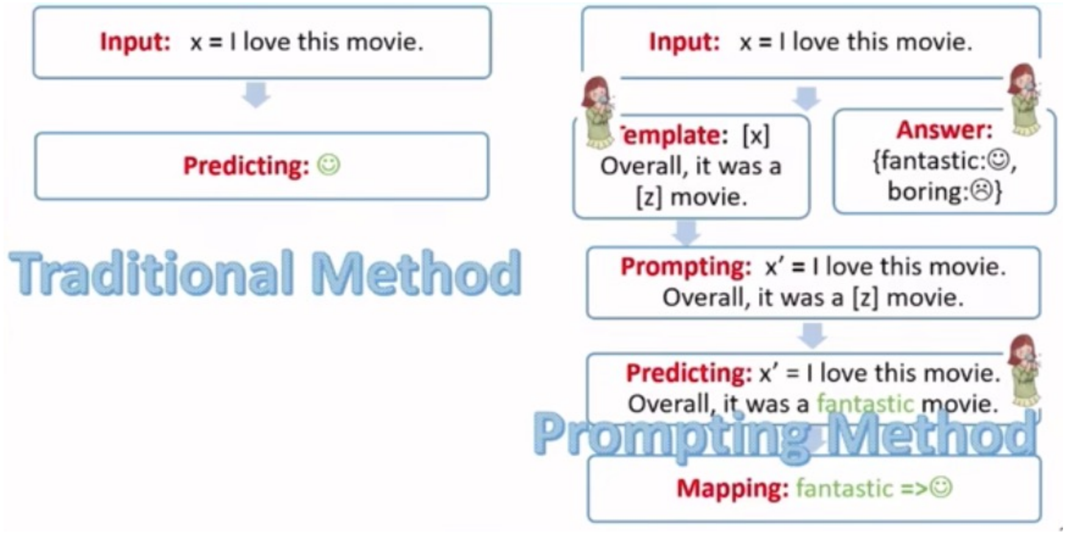
- With so much resource cost, how can we apply pre-trained large language models to really help us, by listening to us, following our instruction, giving us what we want…
- Some said:
  - Model become larger, it's impractical to fine-tuning models for specific tasks, no computation resource
  - GPT-3 has learnt everything, we just learn to induce its knowledge contains for our need
- Others said:
  - We still need to teach model teach to learn to follow our instructions

Prompting V.S. Instructive Fine-tuning

Prompting (Prompt Engineering)

- Unify downstream tasks into pre-training tasks by using specific manual-designed prompting templates
  - Fix model parameters, search for prompts with best performance

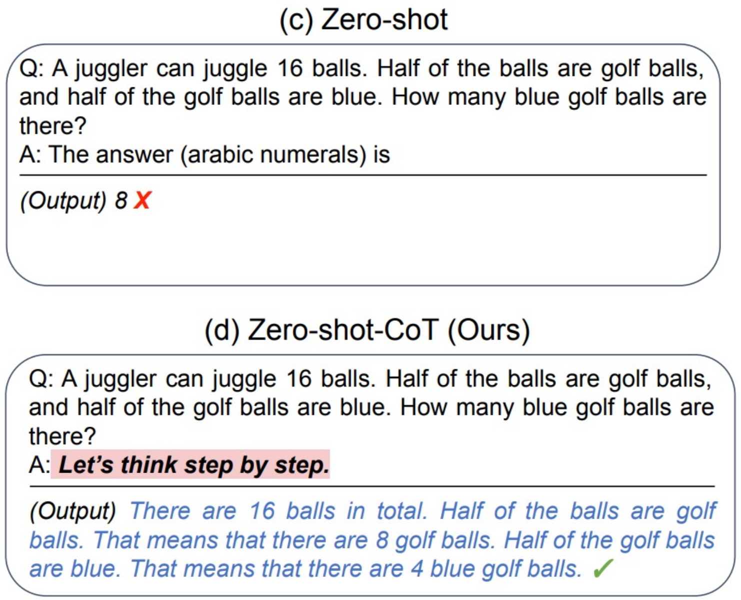


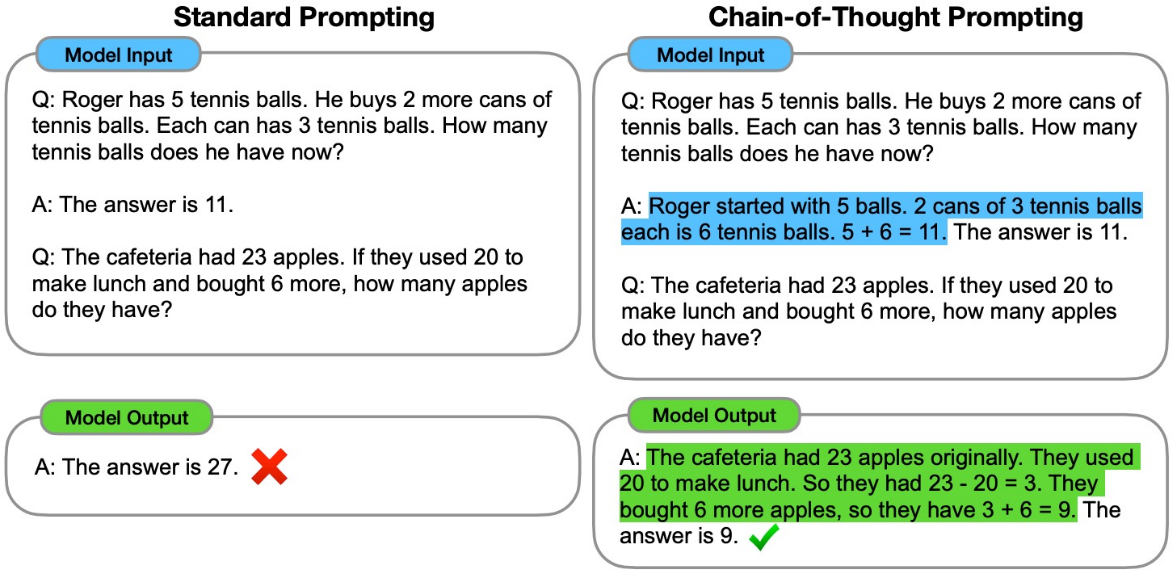


https://link.zhihu.com/?target=https%3A//www.bilibili.com/video/BV1Sf4y1g7ra%3Ffrom%3Dsearch%26seid%3D12417442891380693418%26spm_id_from%3D333.337.0.0

Chain-of-thought (CoT) Prompting

- Investigating the reasoning ability of LLMs





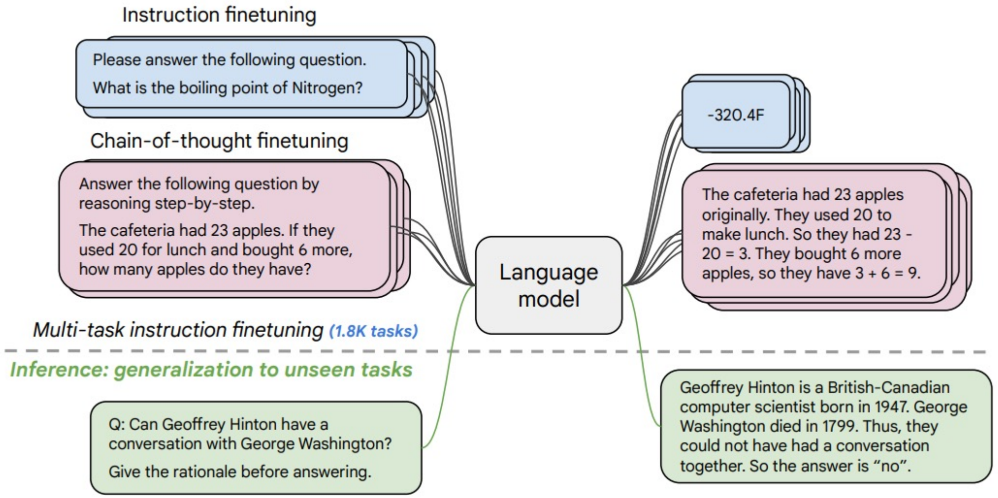
[1] Kojima T, Gu S S, Reid M, et al. Large language models are zero-shot reasoners[J]. Advances in neural information processing systems, 2022, 35: 22199-22213
[2] Wei J, Wang X, Schuurmans D, et al. Chain-of-thought prompting elicits reasoning in large language models[J]. Advances in neural information processing systems, 2022, 35: 24824-24837.

Prompting (Prompt Engineering)

- Pros
  - Parameter efficient (almost no need for adjusting model parameters)
  - Efficiently using pre-trained knowledge
- Cons
  - Design prompting is an art (different prompts generate various results for the same task)
  - More like an engineering rather than research
  - What if LLMs did not learn everything?

Instructive Fine-tuning Large Language Models

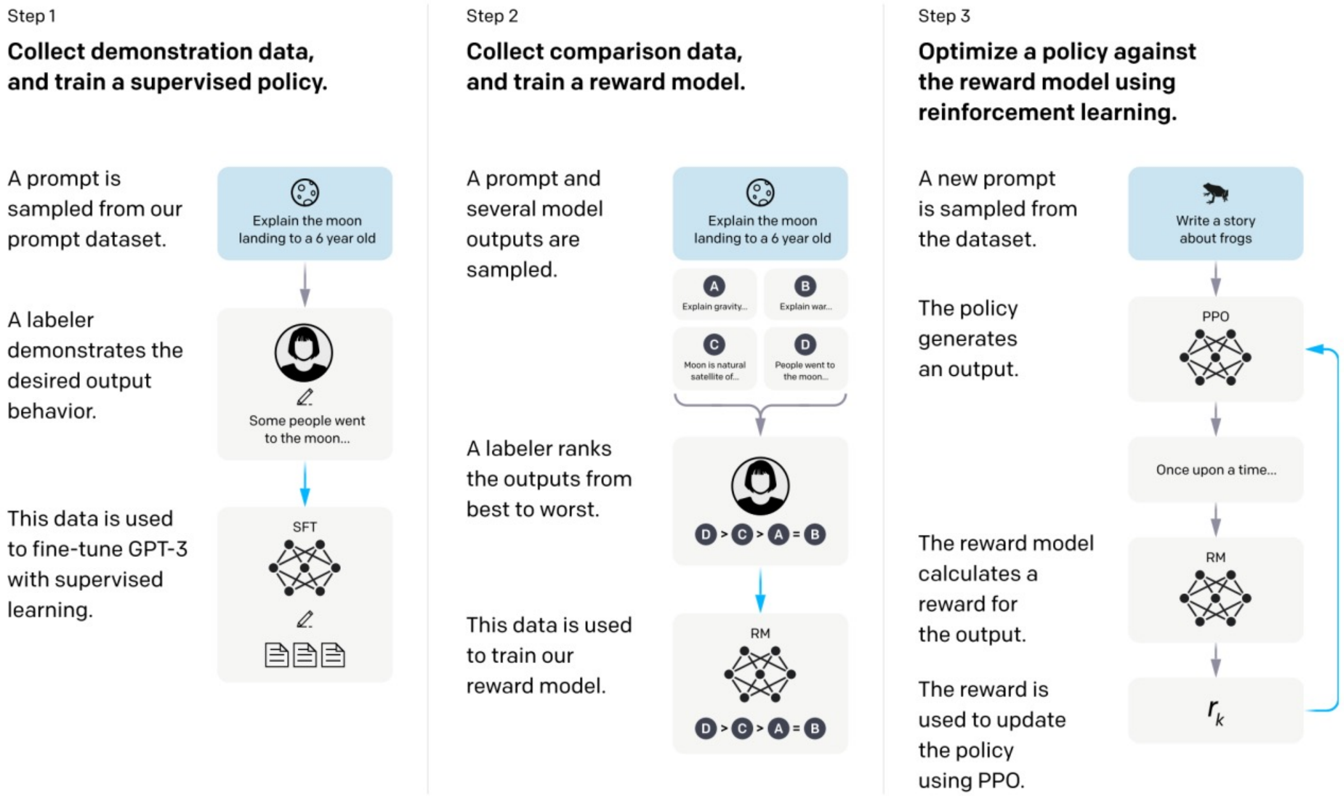
- Fine-tuning LLMs to learn to follow our instructions, even for unseen tasks[1]
  - T5 models finetuned on **1.8K** additional tasks
- Pros
  - Model can indeed learn to following instructions
  - Simple and straightforward
- Cons
  - Collecting demonstrations for so many tasks is expensive
  - Mismatch between LM objective and **human preferences**
    - tasks like open-ended creative generation have no right answer
    - Language Modeling penalizes all token-level mistakes equally, but some errors are worse than others



[1] Chung H W, Hou L, Longpre S, et al. Scaling instruction-finetuned language models[J]. arXiv preprint arXiv:2210.11416, 2022.

Reinforcement Learning from Human Feedbacks (RLHF)

- **Instructive** Supervised Fine-tune (SFT) on GPT-3 based on some collected SFT datasets
- Collect manually labeled **comparison** data and train the reward model (Reword Model, RM)
- Use RM as the optimization target of **reinforcement learning** and use the PPO algorithm to fine-tune the SFT model

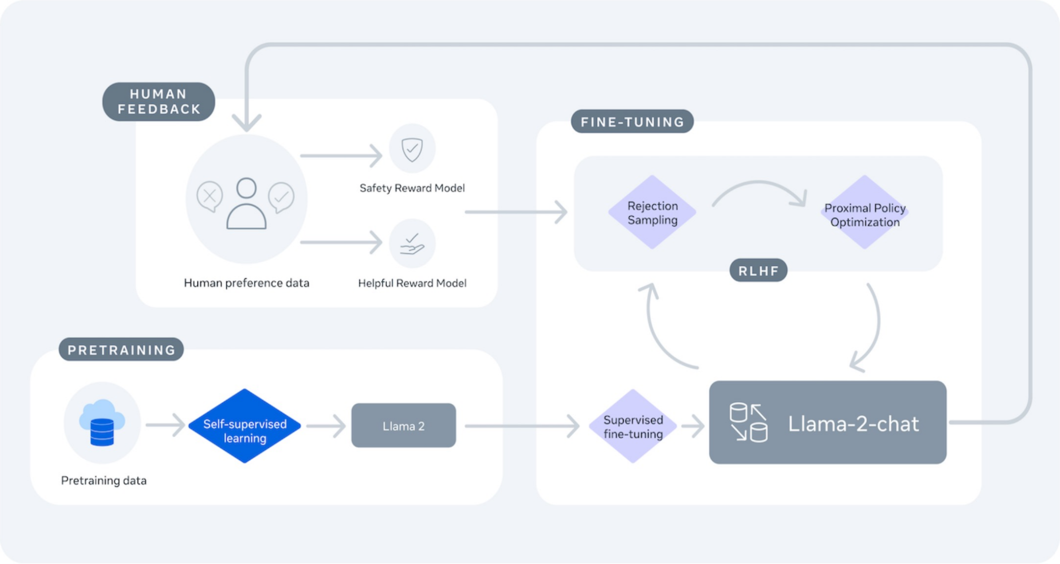


InstructGPT[1]

[1] Ouyang L, Wu J, Jiang X, et al. Training language models to follow instructions with human feedback[J]. Advances in neural information processing systems, 2022, 35: 27730-27744.

Instructive SFT + RLHF

- Standard workflow for training LLMs
  - ChatGPT: InstructGPT focus on the conversational scenarios (probably)
  - Llama-2-chat:
    - Llama 2 + Instructive SFT + RLHF
    - Llama 2 is pretrained using publicly available online data
  - …



NLP Research Trends in era of LLMs

- Training domain-specific LLMs (Powerful NLU and NLG ability)
  - finance, legal, medical, scientific documents,…
- Emotional intelligence (Powerful conversational ability)
  - emotional support, elderly companion, mental health therapy, …
- Parameter-efficiency
  - model quantization, parameter-efficient fine-tuning (LoRA), P-tuning, …
- Enhancing specific skills
  - retrieval augmented generation, adaptation, knowledge enhancement, …

Trends are led by large companies and famous research groups…

Development of NLP techniques

- If you learn NLP before 2010…
  - rule-based methods with linguistics knowledge
  - statistical features, maybe traditional ML models (SVM, DT,…)
- If you learn NLP before 2018…
  - word vectors + RNN, CNN, GRU, LSTM, …
  - attention-based NNs, bidirectional NNs, Multi-layer NNs, …
- If you learn NLP before 2022…
  - fine-tuning Pre-trained Models like BERT, RoBERTa, T5, GPT-2
- If you learn NLP before 2024…
  - prompting, SFT, investigating different abilities of SOTA LLMs
- If you learn NLP now…
  - Do you still want to learn NLP?

Model sizes are bigger and bigger,
technical issues seem less and less…

The things we can do in NLP is less?

Multi-modality?
Reasoning ability?
Factual correctness?
Ethical concerns and governance?
…

History tells us: technics kill old jobs, but create better expectations, so as the new requirements and new jobs…

Research Assistant Wanted

- If the answer for you is YES!
- If you interested in conversational AI, personality/emotion in dialog systems,…
  - Can chatbot recognize the personality of users and provide personalized responses?
  - How do we specify personality to chatbot and let it generate emotional responses according to its personality?
  - Does LLM have its own personality? How do we induce it and how do we control it as our wishes?
- My personal website and my defense slide
- Send application with cv to zyuanwen@polyu.edu.hk

References & Recommended materials

- Some materials are based on Stanford CS224n
- Speech and Language Processing from Dr. Dan Jurafsky
- Open Lectures from Dr. Hung-yi Lee (in Chinese)
- CoLab, Huggingface, …

- This slide does not include all the mainstream NLP models, definitely does not contain all the technical details
- This is just introduction and organization. To master the skills, the knowledge is still there