Georgia Tech
College of Computing

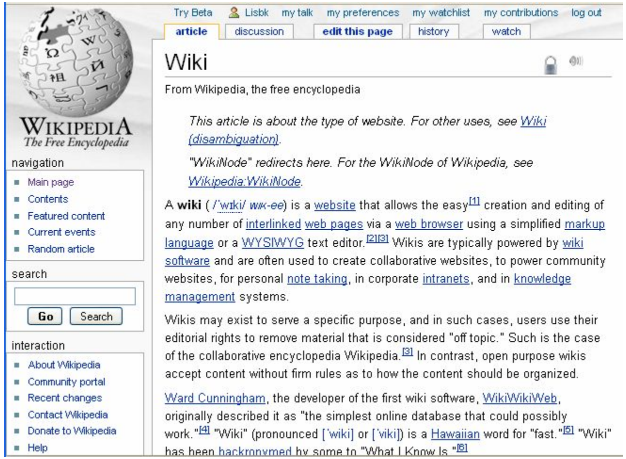
CS4641 Spring 2025

Generative LLM (Part II): Post-Training

Bo Dai
School of CSE, Georgia Tech
bodai@cc.gatech.edu

(Large) Language Models

Training Data → Learning Algorithm → Target Function



Wiki

From Wikipedia, the free encyclopedia

This article is about the type of website. For other uses, see Wiki (disambiguation).

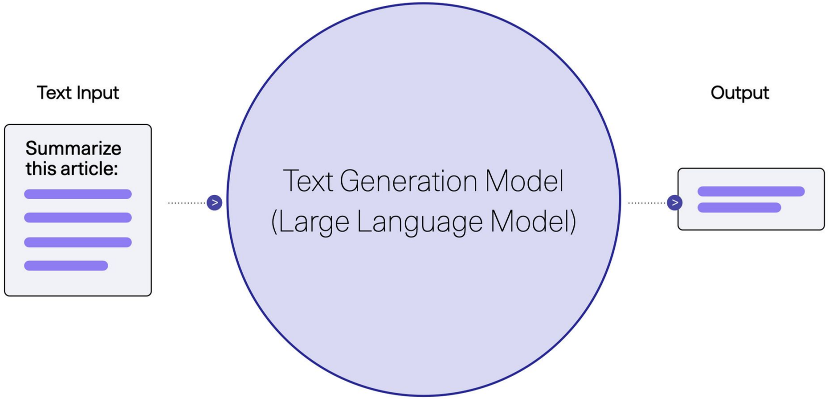
"WikiNode" redirects here. For the WikiNode of Wikipedia, see Wikipedia:WikiNode.

A **wiki** (/ˈwɪki/ *wik-ee*) is a website that allows the easy[1] creation and editing of any number of interlinked web pages via a web browser using a simplified markup language or a WYSIWYG text editor.[2][3] Wikis are typically powered by wiki software and are often used to create collaborative websites, to power community websites, for personal note taking, in corporate intranets, and in knowledge management systems.

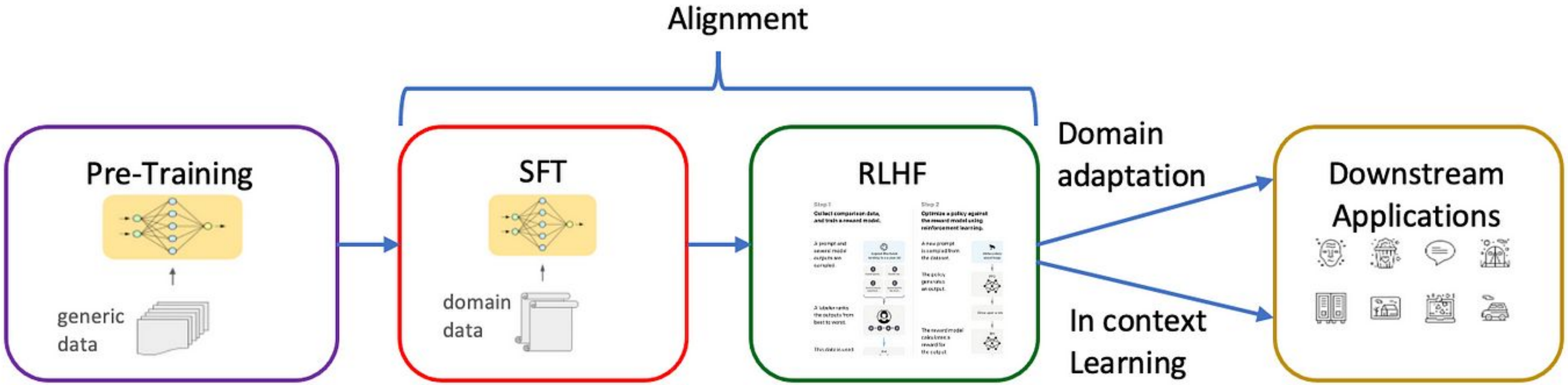
Wikis may exist to serve a specific purpose, and in such cases, users use their editorial rights to remove material that is considered "off topic." Such is the case of the collaborative encyclopedia Wikipedia.[3] In contrast, open purpose wikis accept content without firm rules as to how the content should be organized.

Ward Cunningham, the developer of the first wiki software, WikiWikiWeb, originally described it as "the simplest online database that could possibly work."[4] "Wiki" (pronounced [ˈwiki] or [ˈviki]) is a Hawaiian word for "fast."[5] "Wiki" has been backronymed by some to "What I Know Is."[6]

Unlabelled Data: text sequences



LLM Training



Alignment

Pre-Training

generic data

SFT

domain data

RLHF

Step 1

Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used

Step 2

Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

The policy generates an output.

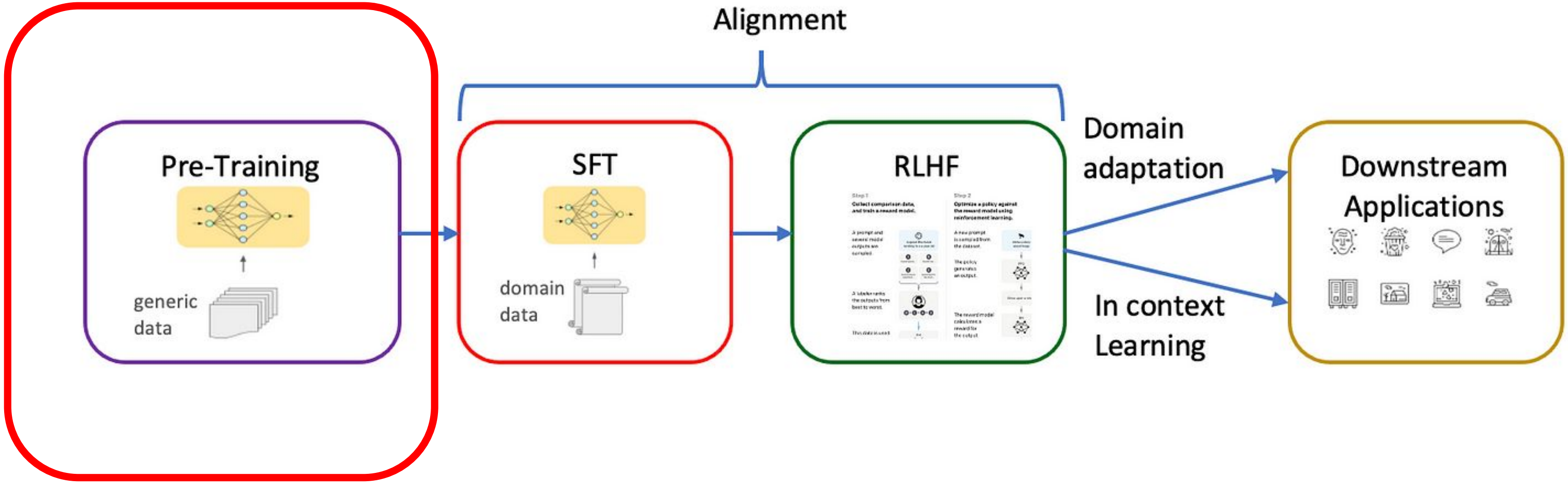
The reward model calculates a reward for the output.

Domain adaptation

In context Learning

Downstream Applications

LLM Training



Alignment

Pre-Training

generic data

SFT

domain data

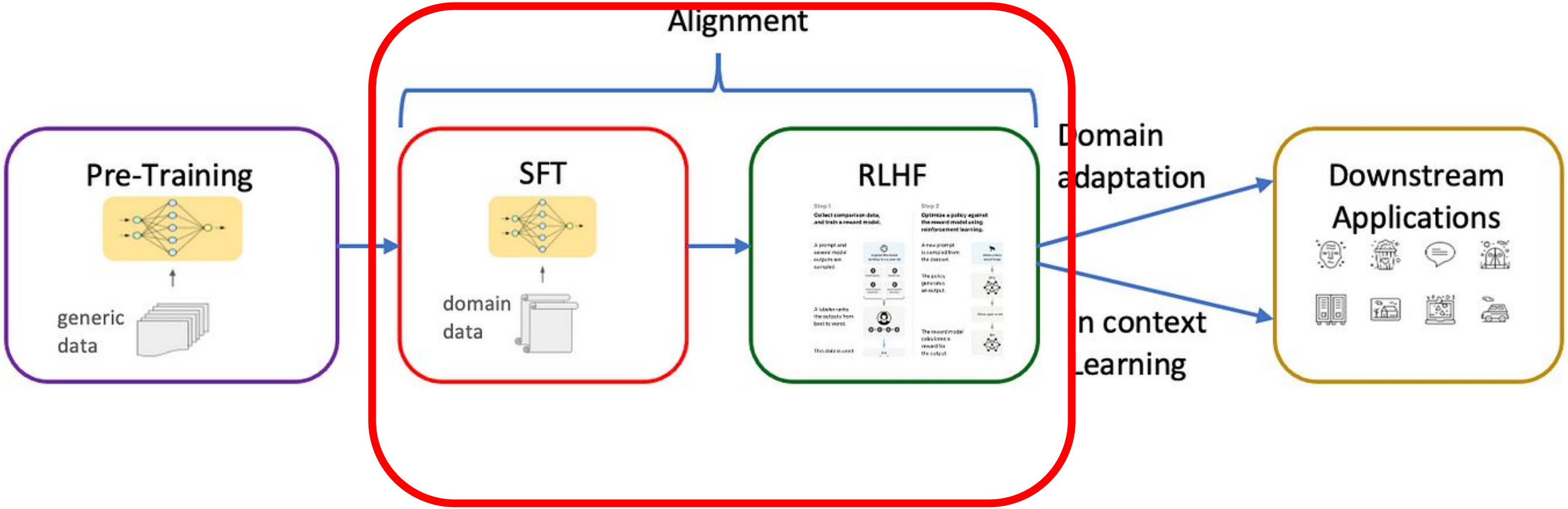
RLHF

Domain adaptation

In context Learning

Downstream Applications

LLM Training



Alignment

Pre-Training

generic data

SFT

domain data

RLHF

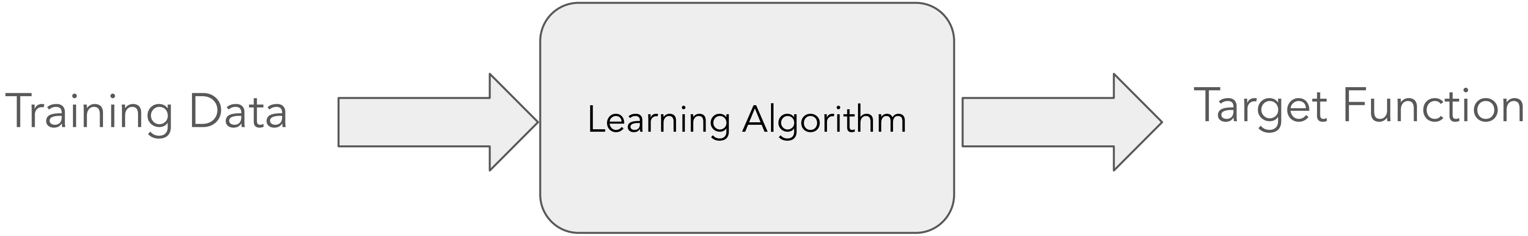
Step 1
Collect comparison data, and train a reward model.
A prompt and several model outputs are sampled.
A labeler ranks the outputs from best to worst.
This data is used
Step 2
Optimize a policy against the reward model using reinforcement learning.
A new prompt is sampled from the dataset.
The policy generates an output.
The reward model calculates a reward for the output.

Domain adaptation

In context Learning

Downstream Applications

Pretraining of (Large) Language Models



Training Data

Learning Algorithm

Target Function

$$D = \{X^i\}_{i=1}^{n}, \quad X^i = \{x_j^i\}_{j=1}^{t}$$

$$p(X)$$

1. Build probabilistic models
   Categorical Distribution + Autoregressive + RNN/Transformer
2. Derive loss function (by MLE or MAP….)
   MLE
3. Select optimizer
   Stochastic Gradient Descent

(Large) Language Models

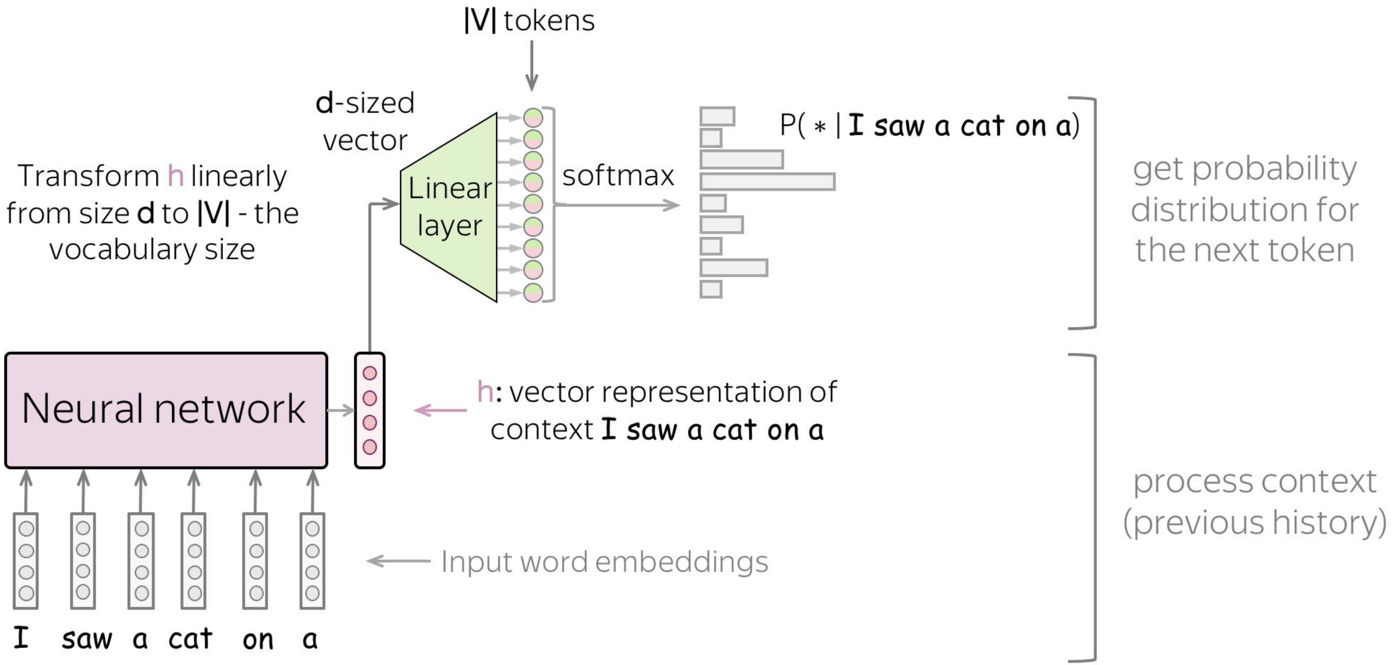
$$
\begin{aligned}
p(X) &= p\big(\{x_j\}_{j=1}^{t}\big) \\
&= \prod_{j=1}^{t} p(x_j | x_{<j}) \\
&= \prod_{j=1}^{t} \frac{\exp(W_{x_j}\phi(x_{<j}))}{\sum_{l=1}^{V} \exp(W_l \phi(x_{<j}))}
\end{aligned}
$$

Recursive Neural Network in (Large) Language Models

$$p(X) = p\left(\{x_j\}_{j=1}^{t}\right) \qquad O(|V|^T)$$

$$= \prod_{j=1}^{t} p(x_j|x_{<j})$$

$$= \prod_{j=1}^{t} \frac{\exp(W_{x_j}\boxed{\phi(x_{<j}))}}{\sum_{l=1}^{V} \exp(W_l\phi(x_{<j}))} \qquad \text{RNN}$$



|V| tokens

d-sized vector

Transform h linearly from size **d** to |V| - the vocabulary size

Linear layer

softmax

P(∗ | **I saw a cat on a**)

get probability distribution for the next token

Neural network

h: vector representation of context **I saw a cat on a**

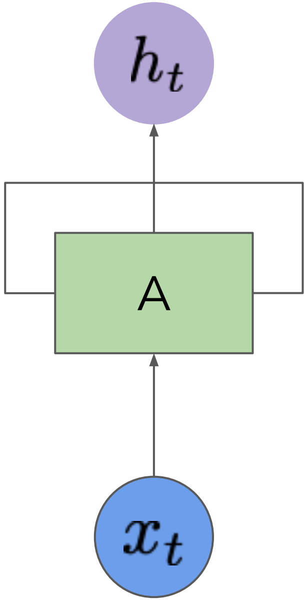
process context (previous history)

Input word embeddings

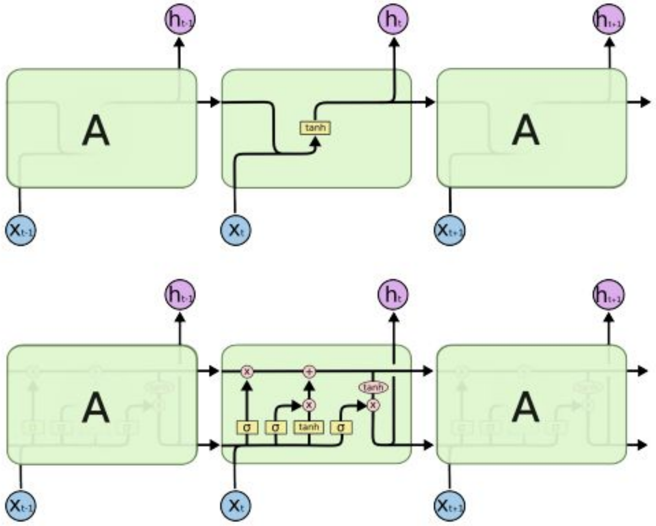
I saw a cat on a

https://lena-voita.github.io/nlp_course/language_modeling.html

RNN Cell



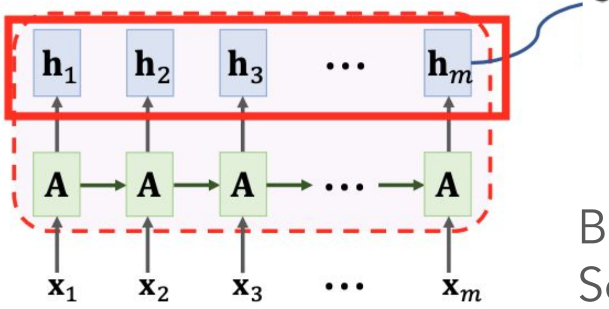
$$\frac{\exp(W_{x_j}\phi(x_{<j}))}{\sum_{l=1}^{V}\exp(W_l\phi(x_{<j}))}$$



Simple RNN

LSTM

Bottleneck in RNN



x_{m+1}

$$h_j = \phi(x_{<j})$$

$$\frac{\exp(W_{x_j}\phi(x_{<j}))}{\sum_{l=1}^{V}\exp(W_l\phi(x_{<j}))}$$

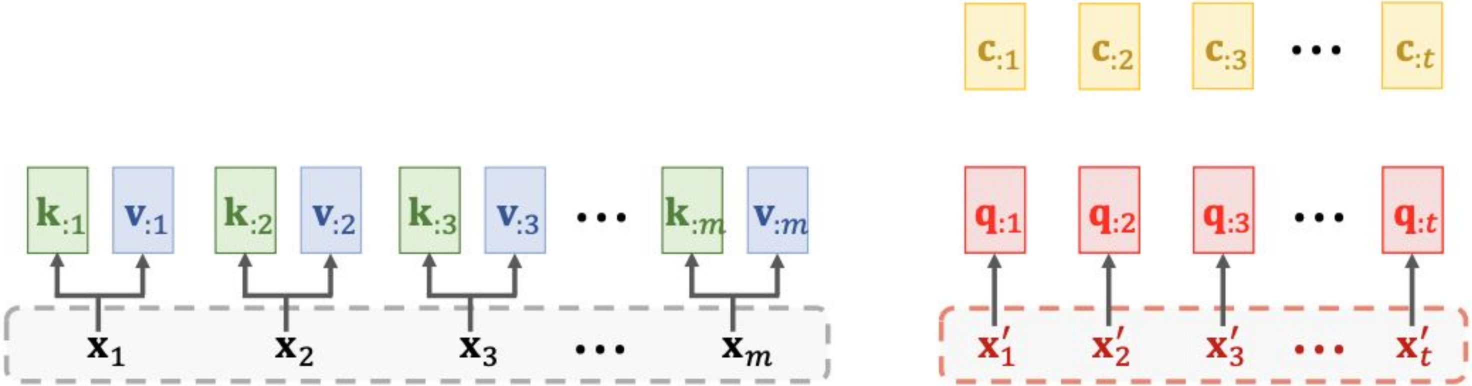
Bottleneck:
Sequences bottlenecked through a fixed-sized vector.

Parallel Attention

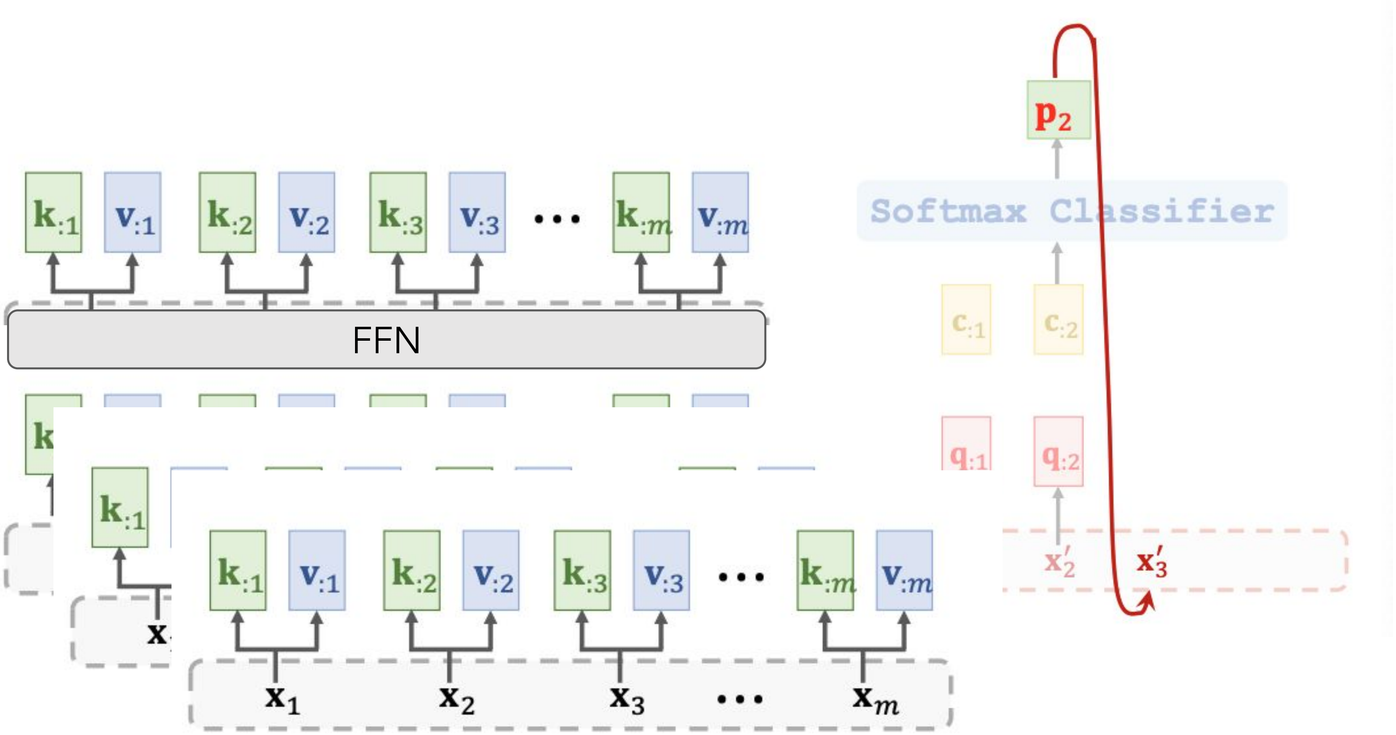
Query: $\mathbf{q}_{:j} = \mathbf{W}_Q \, \mathbf{x}_j$ Key: $\mathbf{k}_{:i} = \mathbf{W}_K \mathbf{h}_i,$ Value: $\mathbf{v}_{:i} = \mathbf{W}_V \mathbf{h}_i.$

Weights: $\alpha_{:j} = \mathrm{Softmax}\left(\mathbf{K}^T \mathbf{q}_{:j}\right) \in \mathbb{R}^m.$

Context vector: $\mathbf{c}_j = \alpha_{1j}\mathbf{v}_{:1} + \cdots + \alpha_{mj}\mathbf{v}_{:m}.$

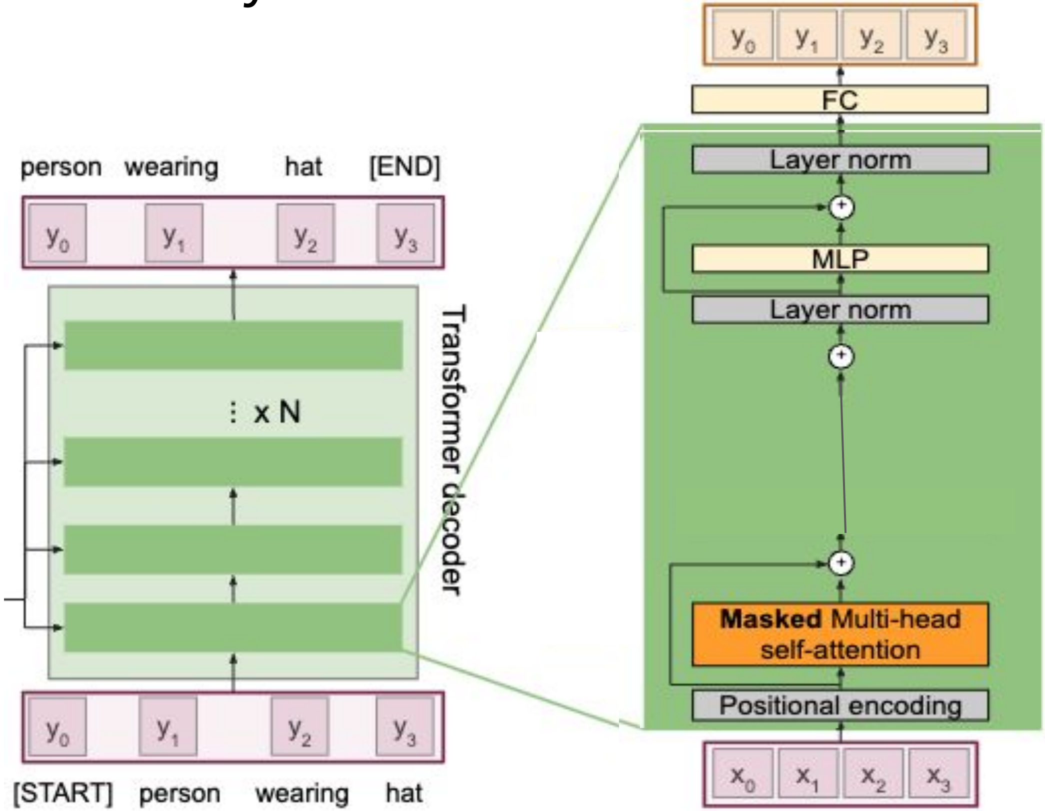


Transformer: Multi-Headed Multi-Layer Parallel Attention!



$\mathbf{p}_2$
Softmax Classifier
$\mathbf{c}_{:1}$
$\mathbf{c}_{:2}$
$\mathbf{q}_{:1}$
$\mathbf{q}_{:2}$
$\mathbf{x}'_2$
$\mathbf{x}'_3$
$\mathbf{k}_{:1}$ $\mathbf{v}_{:1}$ $\mathbf{k}_{:2}$ $\mathbf{v}_{:2}$ $\mathbf{k}_{:3}$ $\mathbf{v}_{:3}$ ... $\mathbf{k}_{:m}$ $\mathbf{v}_{:m}$
FFN
$\mathbf{x}_1$ $\mathbf{x}_2$ $\mathbf{x}_3$ ... $\mathbf{x}_m$

Decoder-Only Transformer



Is Pretraining Enough for (L)LMs?

What we have:

$$p(X) = p(\{x_j\}_{j=1}^{t}) = \prod_{j=1}^{t} p(x_j | x_{<j})$$

Generating texts unconditionally

Is Pretraining Enough for (L)LMs?

What we have:

$$p(X) = p(\{x_j\}_{j=1}^{t}) = \prod_{j=1}^{t} p(x_j | x_{<j})$$

Generating texts unconditionally

Answer Question? Generating texts condition on answer

Is Pretraining Enough for (L)LMs?

What we have:

$$p(X) = p(\{x_j\}_{j=1}^{t}) = \prod_{j=1}^{t} p(x_j | x_{<j})$$

Generating texts unconditionally

Answer Question? Generating texts condition on answer

$$p(Y|X) = \prod_{l=1}^{L} p(y_l \mid y_{<l}, X)$$

Is Pretraining Enough for (L)LMs?

What we have:

$$p(X) = p(\{x_j\}_{j=1}^{t}) = \prod_{j=1}^{t} p(x_j | x_{<j})$$

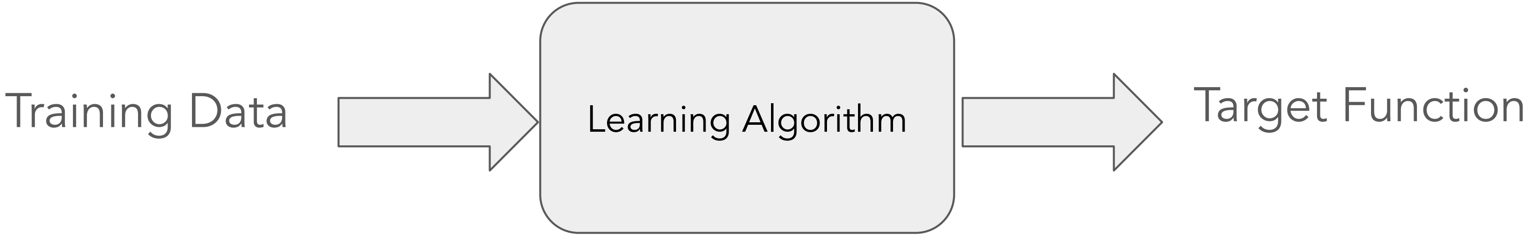
Generating texts unconditionally

Answer Question? Generating texts condition on answer

$$p(Y|X) = \prod_{l=1}^{L} p(y_l \mid y_{<l}, X)$$

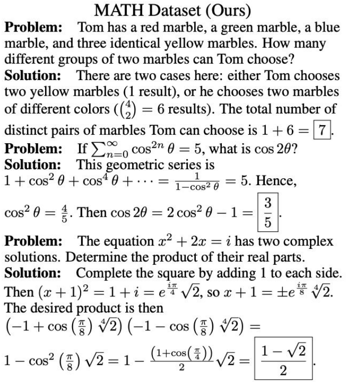
Never be trained

Supervised Fine-tuning (Instruction Fine-tuning)



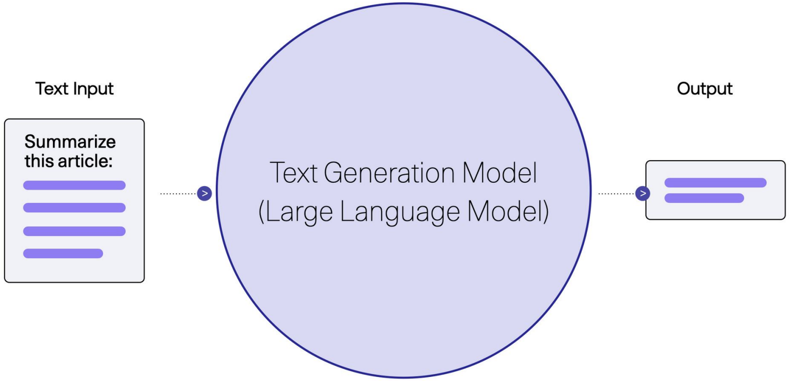
$$D = \{X^i, Y^i\}_{i=1}^{n}, \quad X^i = \{x_j^i\}_{j=1}^{t}, \; Y^i = \{y_l^i\}_{l=1}^{L}$$

$$p(Y|X)$$

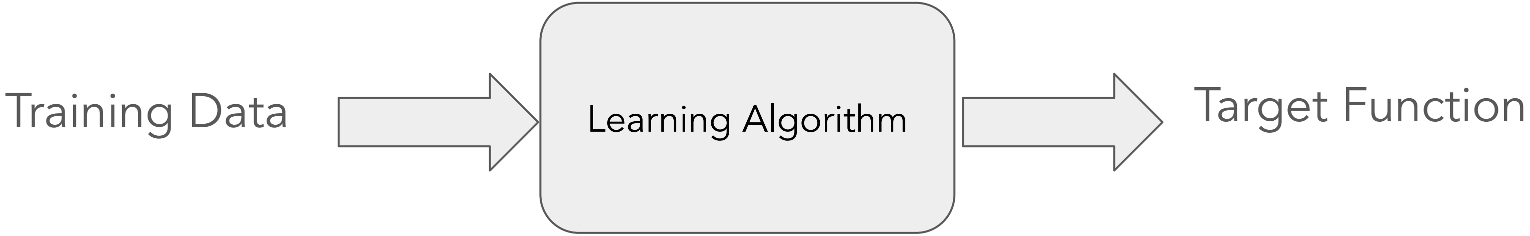


MATH Dataset (Ours)

Problem: Tom has a red marble, a green marble, a blue marble, and three identical yellow marbles. How many different groups of two marbles can Tom choose?
Solution: There are two cases here: either Tom chooses two yellow marbles (1 result), or he chooses two marbles of different colors ($\binom{4}{2} = 6$ results). The total number of distinct pairs of marbles Tom can choose is $1 + 6 = \boxed{7}$.
Problem: If $\sum_{n=0}^{\infty} \cos^{2n}\theta = 5$, what is $\cos 2\theta$?
Solution: This geometric series is $1 + \cos^2\theta + \cos^4\theta + \cdots = \frac{1}{1-\cos^2\theta} = 5$. Hence, $\cos^2\theta = \frac{4}{5}$. Then $\cos 2\theta = 2\cos^2\theta - 1 = \boxed{\frac{3}{5}}$.
Problem: The equation $x^2 + 2x = i$ has two complex solutions. Determine the product of their real parts.
Solution: Complete the square by adding 1 to each side. Then $(x+1)^2 = 1 + i = e^{\frac{i\pi}{4}}\sqrt{2}$, so $x + 1 = \pm e^{\frac{i\pi}{8}}\sqrt[4]{2}$. The desired product is then $\left(-1 + \cos\left(\frac{\pi}{8}\right)\sqrt[4]{2}\right)\left(-1 - \cos\left(\frac{\pi}{8}\right)\sqrt[4]{2}\right) = 1 - \cos^2\left(\frac{\pi}{8}\right)\sqrt{2} = 1 - \frac{\left(1+\cos\left(\frac{\pi}{4}\right)\right)}{2}\sqrt{2} = \boxed{\frac{1-\sqrt{2}}{2}}$.



Supervised Fine-tuning (Instruction Fine-tuning)

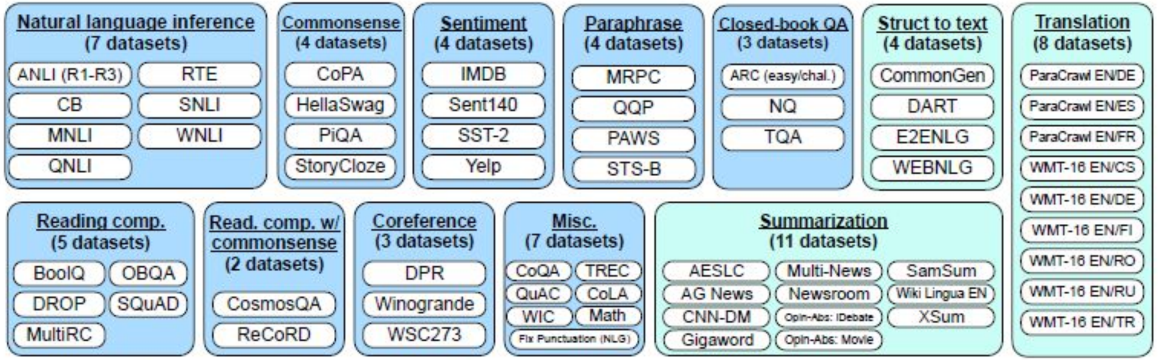


Training Data

Learning Algorithm

Target Function

$$D = \{X^i, Y^i\}_{i=1}^{n}, \quad X^i = \{x_j^i\}_{j=1}^{t}, \; Y^i = \{y_l^i\}_{l=1}^{L} \qquad\qquad p(Y|X)$$



Natural language inference (7 datasets)
- ANLI (R1-R3)
- RTE
- CB
- SNLI
- MNLI
- WNLI
- QNLI

Commonsense (4 datasets)
- CoPA
- HellaSwag
- PiQA
- StoryCloze

Sentiment (4 datasets)
- IMDB
- Sent140
- SST-2
- Yelp

Paraphrase (4 datasets)
- MRPC
- QQP
- PAWS
- STS-B

Closed-book QA (3 datasets)
- ARC (easy/chal.)
- NQ
- TQA

Struct to text (4 datasets)
- CommonGen
- DART
- E2ENLG
- WEBNLG

Translation (8 datasets)
- ParaCrawl EN/DE
- ParaCrawl EN/ES
- ParaCrawl EN/FR
- WMT-16 EN/CS
- WMT-16 EN/DE
- WMT-16 EN/FI
- WMT-16 EN/RO
- WMT-16 EN/RU
- WMT-16 EN/TR

Reading comp. (5 datasets)
- BoolQ
- OBQA
- DROP
- SQuAD
- MultiRC

Read. comp. w/ commonsense (2 datasets)
- CosmosQA
- ReCoRD

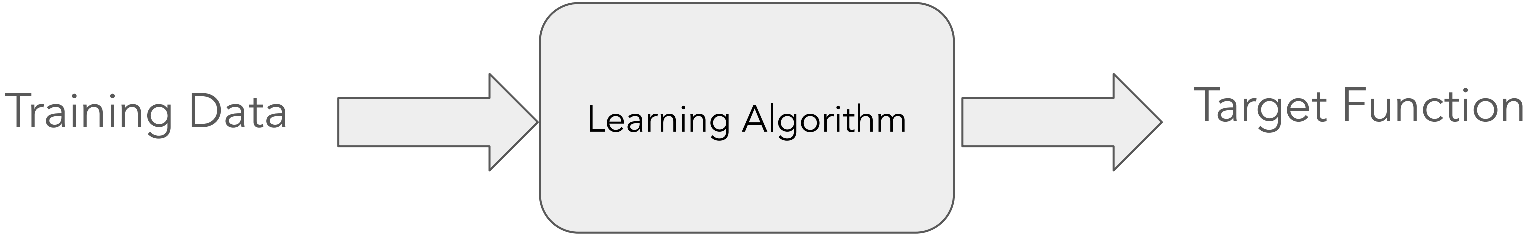
Coreference (3 datasets)
- DPR
- Winogrande
- WSC273

Misc. (7 datasets)
- CoQA
- TREC
- QuAC
- CoLA
- WIC
- Math
- Fix Punctuation (NLG)

Summarization (11 datasets)
- AESLC
- Multi-News
- SamSum
- AG News
- Newsroom
- Wiki Lingua EN
- CNN-DM
- Opin-Abs: iDebate
- XSum
- Gigaword
- Opin-Abs: Movie

Figure 3: Datasets and task clusters used in this paper (NLU tasks in blue; NLG tasks in teal).

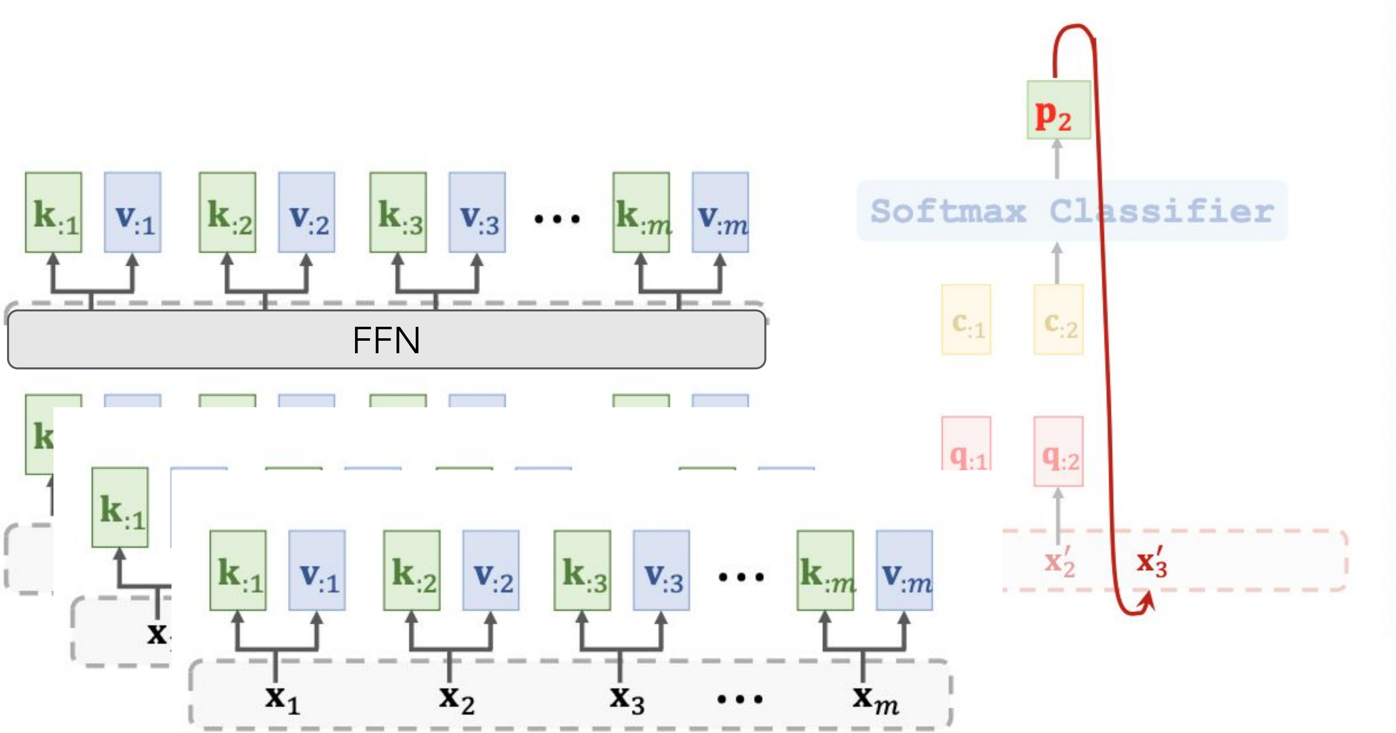
Supervised Fine-tuning (Instruction Fine-tuning)



$$D = \{X^i, Y^i\}_{i=1}^{n}, \quad X^i = \{x_j^i\}_{j=1}^{t}, \; Y^i = \{y_l^i\}_{l=1}^{L} \qquad\qquad p(Y|X)$$

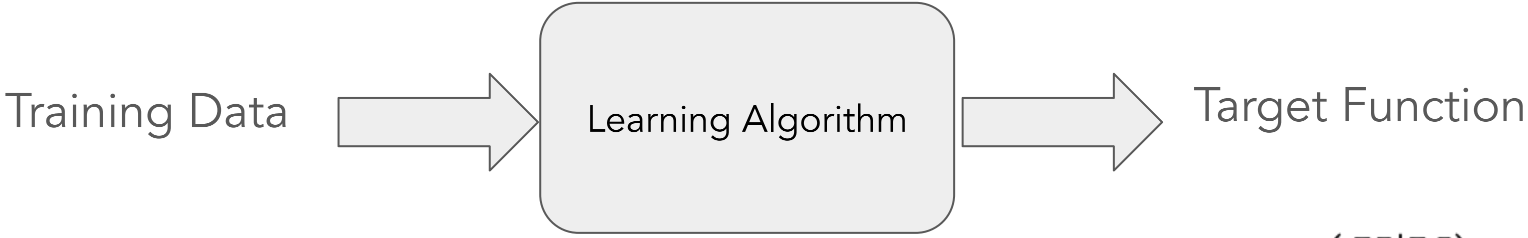
1. Build probabilistic models
   Fixed the same as in pretraining:
   Categorical Distribution + Autoregressive + RNN/Transformer
2. Derive loss function (by MLE or MAP….)
3. Select optimizer

Transformer: Multi-Headed Multi-Layer Parallel Attention!



$\mathbf{k}_{:1}$ $\mathbf{v}_{:1}$ $\mathbf{k}_{:2}$ $\mathbf{v}_{:2}$ $\mathbf{k}_{:3}$ $\mathbf{v}_{:3}$ … $\mathbf{k}_{:m}$ $\mathbf{v}_{:m}$
FFN
$\mathbf{x}_1$ $\mathbf{x}_2$ $\mathbf{x}_3$ … $\mathbf{x}_m$
$\mathbf{p}_2$
Softmax Classifier
$\mathbf{c}_{:1}$ $\mathbf{c}_{:2}$
$\mathbf{q}_{:1}$ $\mathbf{q}_{:2}$
$\mathbf{x}'_2$ $\mathbf{x}'_3$

Supervised Fine-tuning (Instruction Fine-tuning)



Training Data

Learning Algorithm

Target Function

$$D = \{X^i, Y^i\}_{i=1}^n, \quad X^i = \{x_j^i\}_{j=1}^t, \; Y^i = \{y_l^i\}_{l=1}^L$$

$$p(X|Y)$$

1. Build probabilistic models
2. Derive loss function (by MLE or MAP….)

   MLE
3. Select optimizer

MLE for Supervised Fine-tuning (SFT)

- Given all input data $D = \{X^i, Y^i\}_{i=1}^{n}, \quad X^i = \{x_j^i\}_{j=1}^{t}, \; Y^i = \{y_l^i\}_{l=1}^{L}$

$$p(Y \mid X) = \prod_{l=1}^{L} p(y_l \mid y_{<l}, X)$$

- Log-likelihood

$$\begin{aligned}
\ell(\phi) &= \sum_{i=1}^{n} p(Y^i \mid X^i; \phi) = \sum_{i=1}^{n} \sum_{l=1}^{L} \log\left(y_l^i \mid y_{<l}^i, X^i; \phi\right) \\
&= \sum_{i=1}^{n} \sum_{l=1}^{L} \log \frac{\exp\left(W_{y_l^i} \phi\left(y_{<l}^i \mid X^i\right)\right)}{\sum_{v=1}^{V} \exp\left(W_v \phi\left(y_{<l}^i \mid X^i\right)\right)} \\
&= \sum_{i=1}^{n} \sum_{l=1}^{L} W_{y_l^i} \phi\left(y_{<l}^i \mid X^i\right) - \sum_{i=1}^{n} \sum_{j=1}^{t} \log \sum_{v=1}^{V} \exp\left(W_v \phi\left(y_{<l}^i \mid X^i\right)\right)
\end{aligned}$$

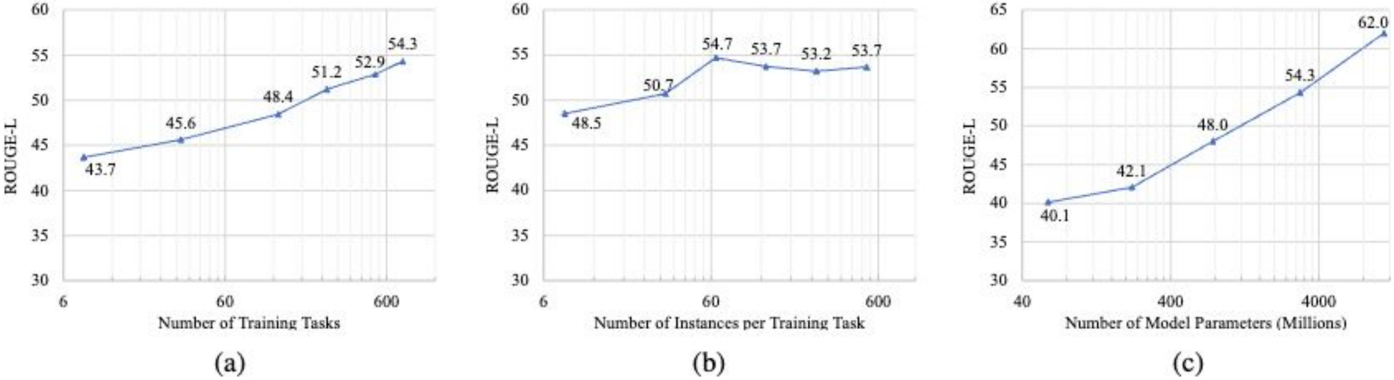
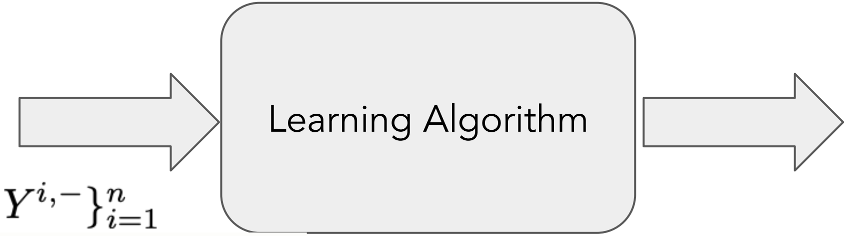


Figure 5: Scaling trends of models performance (§7.1) as a function of (a) the number of training tasks; (b) the number of instances per training task; (c) model sizes. x-axes are in log scale. The **linear growth of model performance with exponential increase in observed tasks and model size** is a promising trend. Evidently, the performance gain from more instances is limited.

Wang, Yizhong, et al. "Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks." *arXiv preprint arXiv:2204.07705* (2022).

RLHF for (Large) Language Models

Training Data

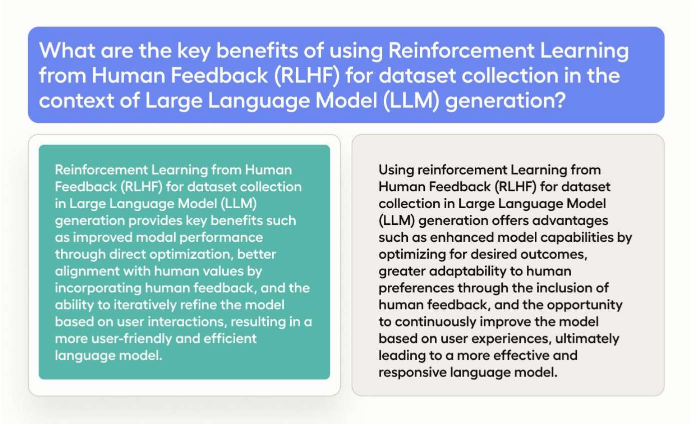


Learning Algorithm

Target Function

$$D = \{X^i, Y^{i,+}, Y^{i,-}\}_{i=1}^n$$

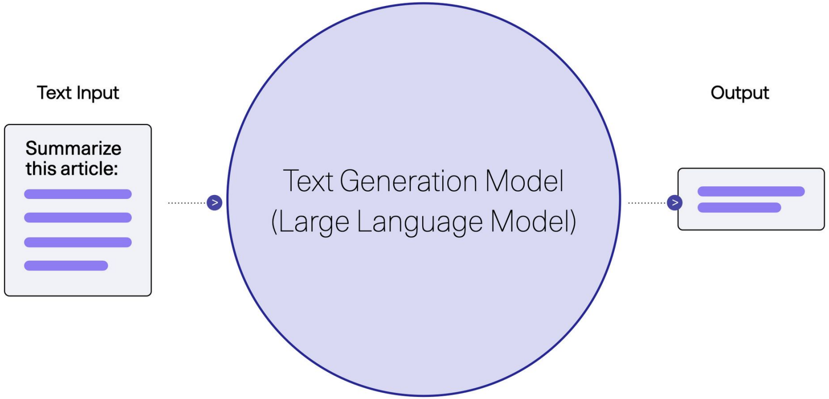
$$p(Y|X)$$



What are the key benefits of using Reinforcement Learning from Human Feedback (RLHF) for dataset collection in the context of Large Language Model (LLM) generation?

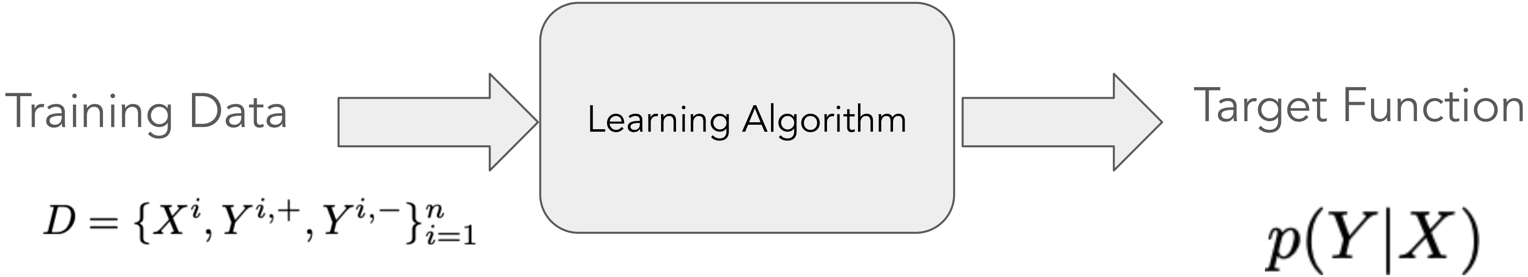
Reinforcement Learning from Human Feedback (RLHF) for dataset collection in Large Language Model (LLM) generation provides key benefits such as improved modal performance through direct optimization, better alignment with human values by incorporating human feedback, and the ability to iteratively refine the model based on user interactions, resulting in a more user-friendly and efficient language model.

Using reinforcement Learning from Human Feedback (RLHF) for dataset collection in Large Language Model (LLM) generation offers advantages such as enhanced model capabilities by optimizing for desired outcomes, greater adaptability to human preferences through the inclusion of human feedback, and the opportunity to continuously improve the model based on user experiences, ultimately leading to a more effective and responsive language model.



Preference Data: Prompt & Positive/ Negative Answers

RLHF for (Large) Language Models



$$D = \{X^i, Y^{i,+}, Y^{i,-}\}_{i=1}^n$$

$$p(Y|X)$$

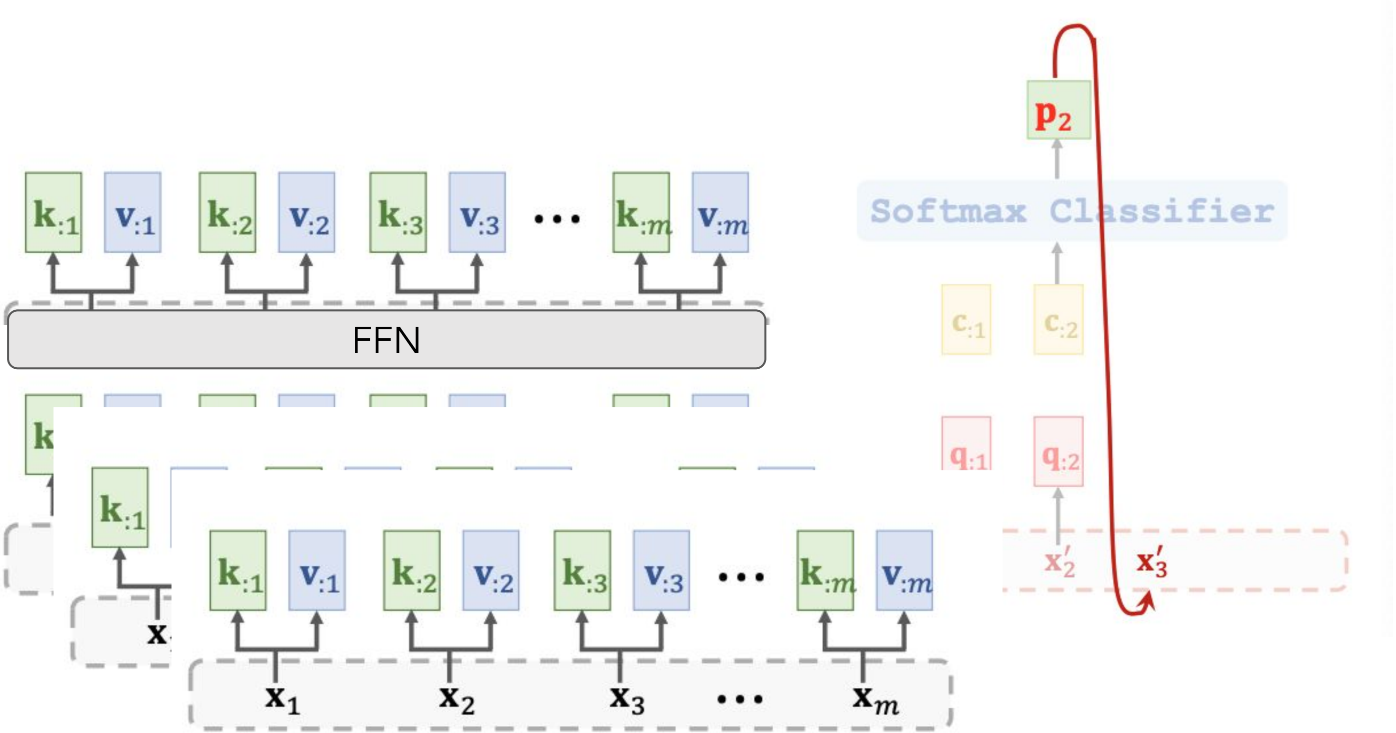
1. Build probabilistic models

   Fixed the same as in pretraining:

   Categorical Distribution + Autoregressive + RNN/Transformer

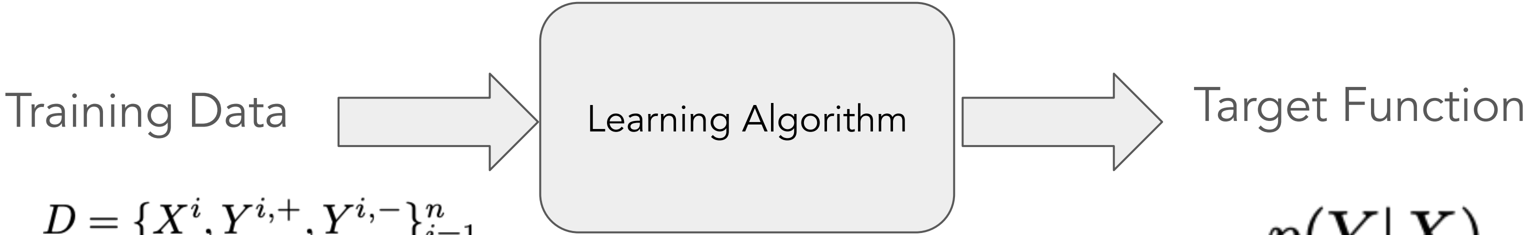
   + classification head (only for learning)
2. Derive loss function (by MLE or MAP….)
3. Select optimizer

Transformer: Multi-Headed Multi-Layer Parallel Attention!



$\mathbf{p}_2$
Softmax Classifier
$\mathbf{c}_{:1}$
$\mathbf{c}_{:2}$
$\mathbf{q}_{:1}$
$\mathbf{q}_{:2}$
$\mathbf{x}'_2$
$\mathbf{x}'_3$
$\mathbf{k}_{:1}$ $\mathbf{v}_{:1}$ $\mathbf{k}_{:2}$ $\mathbf{v}_{:2}$ $\mathbf{k}_{:3}$ $\mathbf{v}_{:3}$ … $\mathbf{k}_{:m}$ $\mathbf{v}_{:m}$
FFN
$\mathbf{x}_1$ $\mathbf{x}_2$ $\mathbf{x}_3$ … $\mathbf{x}_m$

RLHF for (Large) Language Models



Training Data

Learning Algorithm

Target Function

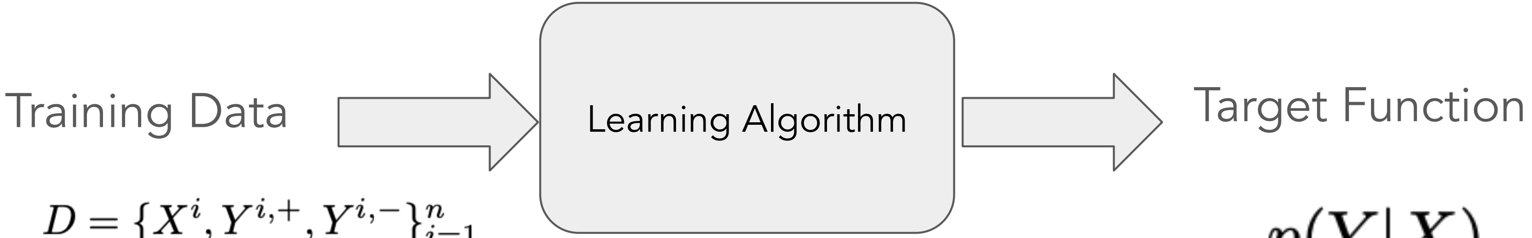
$$D = \{X^i, Y^{i,+}, Y^{i,-}\}_{i=1}^n$$

$$\{X^i, Z^{i,1}, Z^{i,2}, y^i\}_{i=1}^n$$

$$y^i = \mathbf{1}(Z^{i,1} \succ Z^{i,2})$$

$$p(Y|X)$$

RLHF for (Large) Language Models



Training Data

Learning Algorithm

Target Function

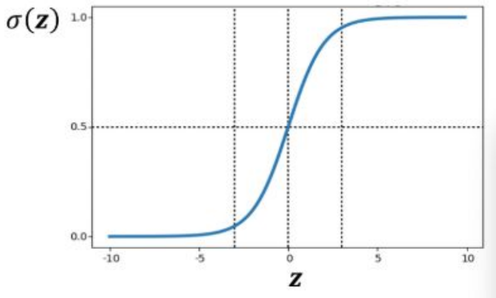
$$D = \{X^i, Y^{i,+}, Y^{i,-}\}_{i=1}^n$$

$$\{X^i, Z^{i,1}, Z^{i,2}, y^i\}_{i=1}^n$$

$$y^i = \mathbf{1}(Z^{i,1} \succ Z^{i,2})$$

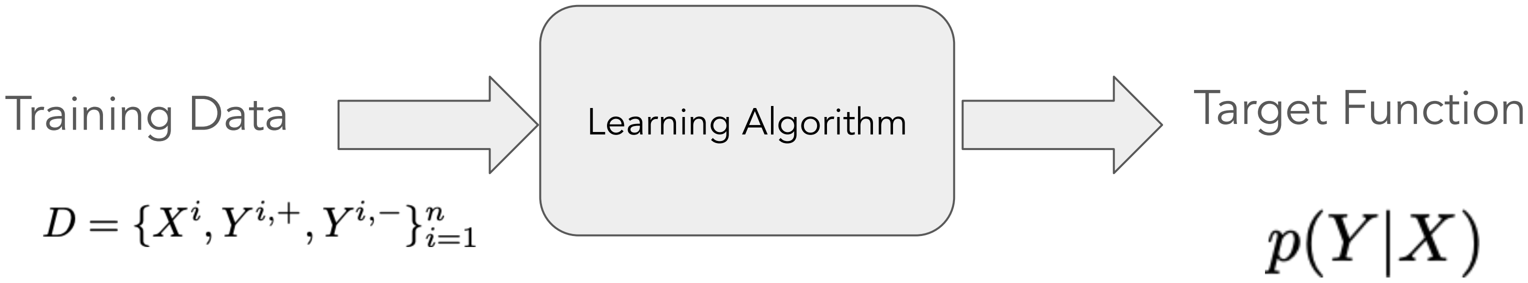
$$p(Y|X)$$

$$\sigma(\mathbf{z}) = \frac{1}{1+e^{-\mathbf{z}}} = \frac{e^{\mathbf{z}}}{1+e^{\mathbf{z}}}$$



$$p(Z^1 \succ Z^2|X) = \sigma\left(\log\frac{p(Z^1|x)}{p_{\text{ref}}(Z^1|x)} - \log\frac{p(Z^2|x)}{p_{\text{ref}}(Z^2|x)}\right)$$

RLHF for (Large) Language Models



$$D = \{X^i, Y^{i,+}, Y^{i,-}\}_{i=1}^n$$

$$p(Y|X)$$

1. Build probabilistic models
2. Derive loss function (by MLE or MAP….)
   MLE
3. Select optimizer

MLE for Preference Learning - Direct Preference Optimization (DPO)

- Given all input data $D = \{X^i, Y^{i,+}, Y^{i,-}\}_{i=1}^n$

$$\{X^i, Z^{i,1}, Z^{i,2}, y^i\}_{i=1}^n$$
$$y^i = \mathbf{1}(Z^{i,1} \succ Z^{i,2})$$

⇨

$$p(Z^1 \succ Z^2 | X) = \sigma\Bigg(\log \frac{p(Z^1|x)}{p_{\text{ref}}(Z^1|x)} - \log \frac{p(Z^2|x)}{p_{\text{ref}}(Z^2|x)} \Bigg)$$

- Log-likelihood

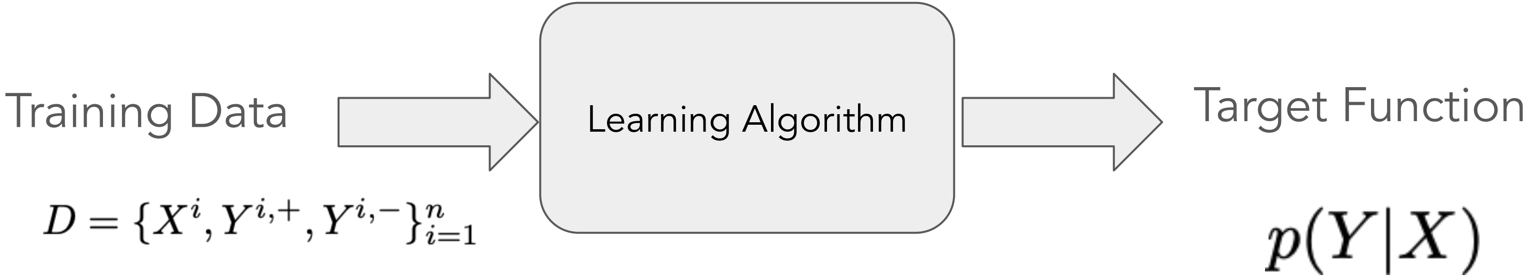
$$\begin{aligned} \ell(\phi) &= \sum_{i=1}^{n} y^i \log p(Z^{i,1} \succ Z^{i,2} \mid X^i; \phi) \\ &= \sum_{i=1}^{n} \log \sigma \left(\log \frac{p_\phi\left(Y^{i,+} \mid X^i\right)}{p_{\text{ref}}\left(Y^{i,+} \mid X^i\right)} - \log \frac{p_\phi\left(Y^{i,-} \mid X^i\right)}{p_{\text{ref}}\left(Y^{i,-} \mid X^i\right)} \right) \end{aligned}$$

Online vs. Offline Samples

$$
\begin{aligned}
\ell(\phi) &= \sum_{i=1}^{n} y^i \log p(Z^{i,1} \succ Z^{i,2} \mid X^i; \phi) \\
&= \sum_{i=1}^{n} \log \sigma \left(\log \frac{p_\phi \left(Y^{i,+} \mid X^i\right)}{p_{\text{ref}} \left(Y^{i,+} \mid X^i\right)} - \log \frac{p_\phi \left(Y^{i,-} \mid X^i\right)}{p_{\text{ref}} \left(Y^{i,-} \mid X^i\right)} \right)
\end{aligned}
$$

Where the samples Y comes from?

RLHF for (Large) Language Models



$D = \{X^i, Y^{i,+}, Y^{i,-}\}_{i=1}^n$

$p(Y|X)$

1. Build probabilistic models
2. Derive loss function (by MLE or MAP….)
3. Select optimizer

Stochastic Gradient Descent

Proximal Policy Optimization (PPO) vs. DPO

1, Learn a reward model

$$\operatorname{loss}(\theta) = -\frac{1}{\binom{K}{2}} E_{(x,y_w,y_l)\sim D}\left[\log\left(\sigma\left(r_\theta\left(x, y_w\right) - r_\theta\left(x, y_l\right)\right)\right)\right]$$

2, Policy Gradient

$$\max_{p(Y|X)} \sum_{X\sim D} \left(\mathbb{E}_{p(Y|X)}[r(X,Y)] - \lambda KL(p(Y|X)||p_{\text{ref}}(Y|X)) \right)$$

Proximal Policy Optimization (PPO) vs. DPO

1, Learn a reward model

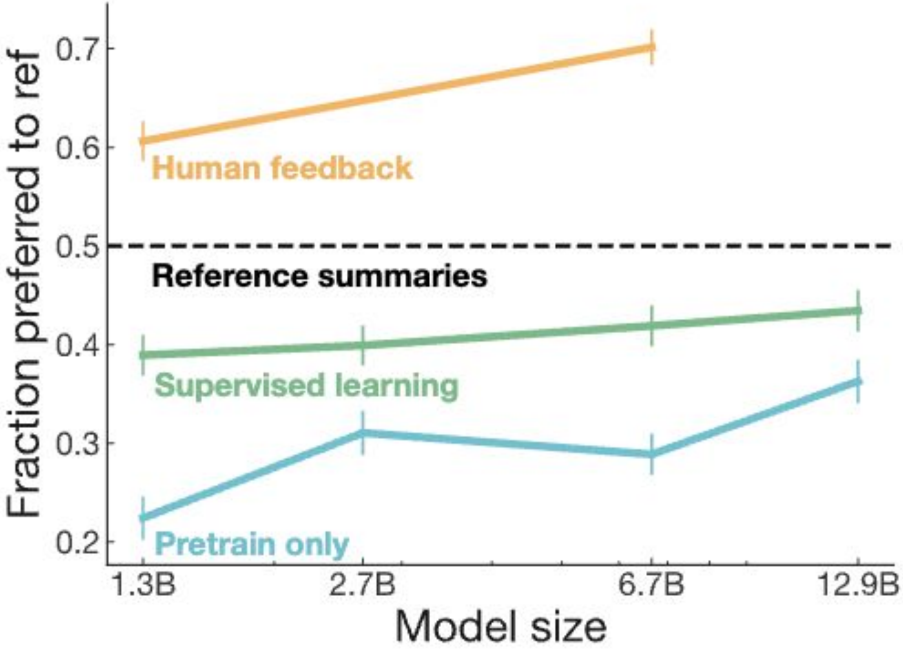
$$E_{(x,y_w,y_l)\sim D}\left[\log\left(\sigma\left(r_\theta\left(x,y_w\right)-r_\theta\left(x,y_l\right)\right)\right)\right]$$

2, Policy Gradient

$$\max_{p(Y|X)} \sum_{X\sim D}\left(\mathbb{E}_{p(Y|X)}[r(X,Y)] - \lambda KL(p(Y|X)||p_{\text{ref}}(Y|X))\right)$$

Naturally Online!

RLHF Performance



Stiennon, Nisan, et al. "Learning to summarize with human feedback." *Advances in neural information processing systems* 33 (2020): 3008-3021.

Summary

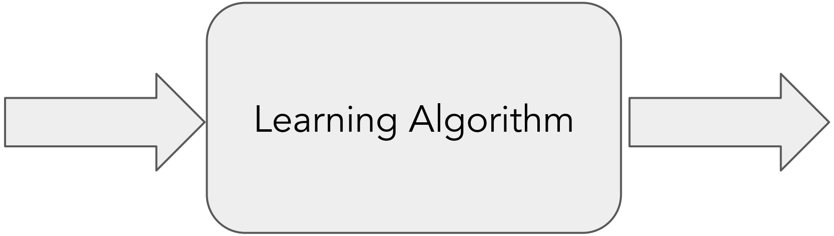
- Pretraining of LLM is not enough
- Post-training of LLM is necessary:
  - Supervised Fine-tuning
  - Reinforcement Learning from Human Feedback
    - Online vs Offline data

Supervised Learning vs. Unsupervised Learning

Training Data

Supervised $\{x^i, y^i\}_{i=1}^m$

Unsupervised $\{x_i\}_{i=1}^m$



Learning Algorithm

Target Function

$$f : X \to Y$$

Regression, Classification

Density Estimation $p(x)$

Clustering

$$f : X \to Y$$

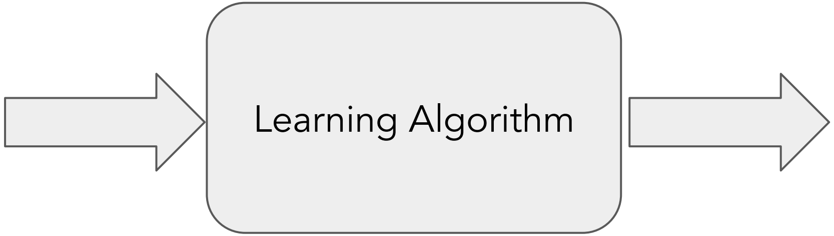
$$Y \in \{0, 1, \dots, k\}$$

Dimension Reduction

$$f : X \to Y$$

$$X \in \mathbb{R}^p, \quad Y \in \mathbb{R}^d$$

Supervised Learning vs. Unsupervised Learning



Training Data

Supervised $\{x^i, y^i\}_{i=1}^m$

Unsupervised $\{x_i\}_{i=1}^m$

Learning Algorithm

Target Function

$f : X \rightarrow Y$

Regression, Classification

Density Estimation $p(x)$

Clustering

$$f : X \rightarrow Y$$

$$Y \in \{0, 1, \ldots, k\}$$

Dimension Reduction

$$f : X \rightarrow Y$$

$$X \in \mathbb{R}^p, \quad Y \in \mathbb{R}^d$$

General ML Algorithm Pipeline

1. Build probabilistic models
2. Derive loss function:

   MLE vs. MAP
3. Select optimizer

   Necessary Condition vs. (Stochastic) GD

Q&A

Thanks for Attending in the whole semester