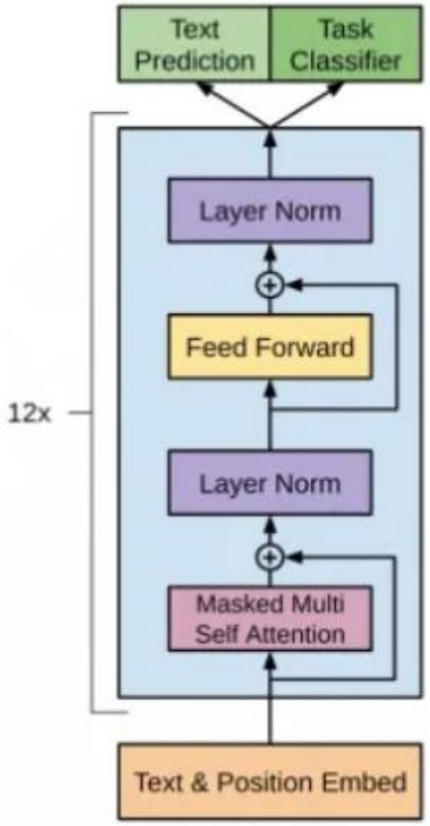
自然語言處理

Generative Pre-trained Transformer

Instructor: 馬豪尚

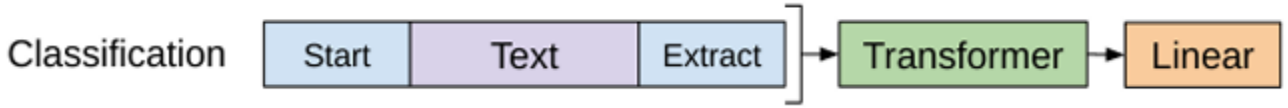
Generative Pre-trained Transformer（GPT）

- 是由 OpenAI 提出的預訓練語言模型，這一系列的模型可以執行非常複雜的 NLP 任務
- GPT 採用 Transformer 作為解碼器（decoder）
- GPT 主要是透過大規模的語料庫做語言模型的預訓練（不須給標籤的無監督式學習），再透過微調（監督式學習）做遷移學習(Transfer Learning)

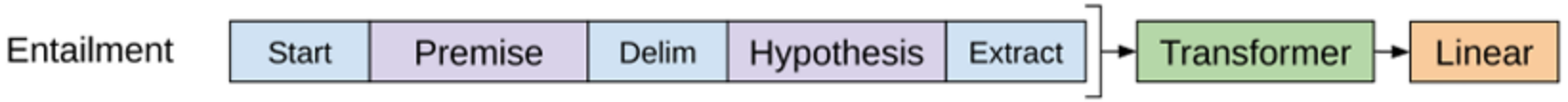


Fine-tuning 任務

- 對於文本分類任務，將所有結構化的輸入轉換成token序列，輸入預訓練的模型進行處理，最後加上一層線性 softmax 層對GPT模型進行微調

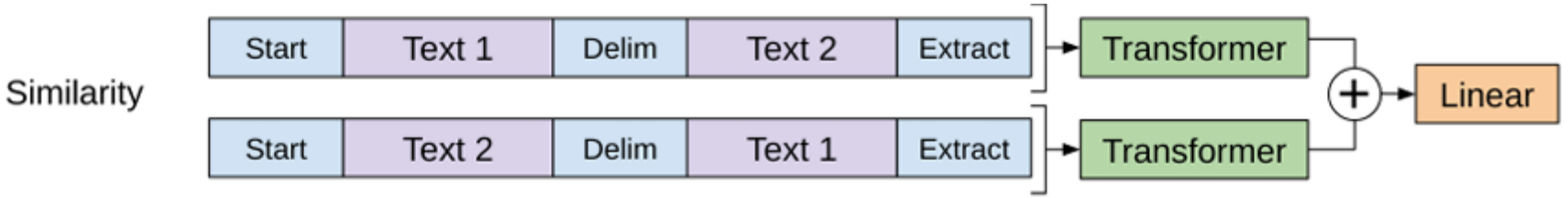


- 對於文本蘊含任務任務，將前提p和假設h的token序列連接起來，在它們之間加上一個分隔標記($)，最後加上一層線性 softmax 層對GPT模型進行微調



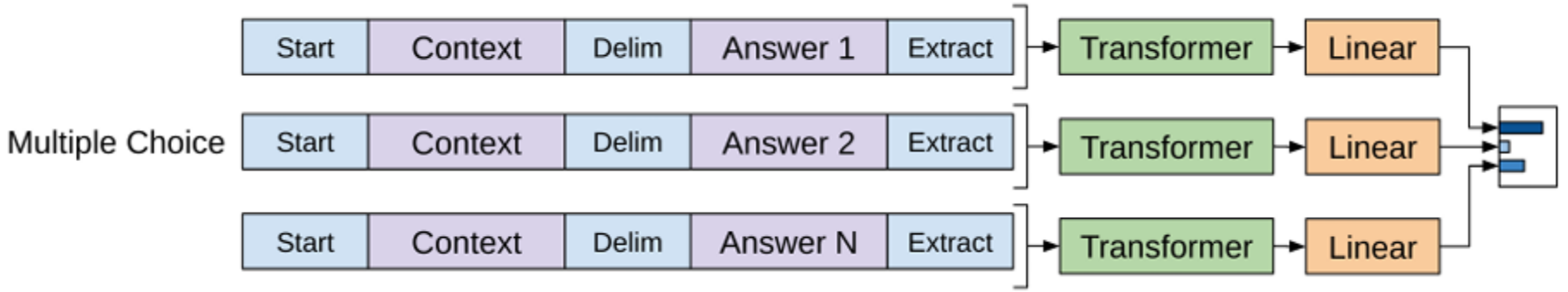
Fine-tuning 任務

- 對於相似度任務，被比較的兩個句子沒有一定的輸入順序。
- 為了反映這一點，透過修改輸入序列，包含兩個可能的句子排序（中間有分隔符），並獨立處理每個序列以產生兩個序列的隱藏層表示(representation)，最後送入線性輸出層之前會進行element-wise相加



Fine-tuning 任務

- 問答和常識推理任務，輸入資訊會有一個上下文文件z，一個問題q，以及一組可能的答案{ak}
- 將文件上下文和問題與每個可能的答案連接起來，在它們之間加上分隔標記，以獲得[z;q;$;ak]
- 這些序列獨立地用GPT模型處理，並通過softmax層進行歸一化，以產生對可能答案的輸出分佈



GPT-1~3

- 2018年時，GPT-1 誕生，是一個通用的生成預訓練模型，沒有經過專門訓練來執行任何特定的任務
- 2019 年，GPT-2 發布，模型架構主要改變只有使用了更多參數與數據集，模型共計 48 層，參數量達 15 億
- 2020年，OpenAI發布了新的 GPT-3，延續過去 GPT的訓練方式，只是將模型增大到 1750 億參數，並且使用 45TB 的資料量給訓練出來

模型	發佈時間	參數量	預訓練數據量
GPT	2018年6月	1.17 億	約5GB
GPT-2	2019年2月	15 億	40GB
GPT-3	2020年5月	1750 億	45TB

GPT-1參數

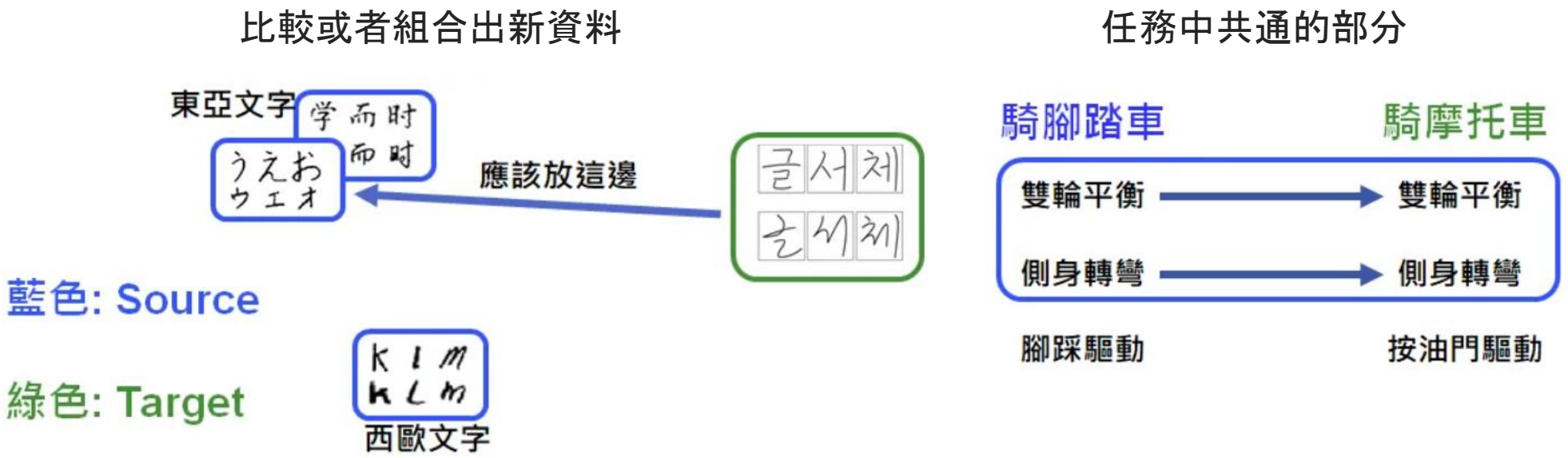
- 使用BooksCorpus dataset做預訓練
- 使用位元組對編碼（byte pair encoding，BPE），字典大小約40000
- 詞編碼的長度為 768
- 12層的transformer，每個transformer區塊都有12個head
- 位置編碼的長度是3072
- 啟動函數為GLEU
- 訓練的batch size為64，學習率為2.5e^{-4}，序列長度為 512，序列 epoch為100
- 模型參數數量為 1.17 億

GPT-2

- 作者認為，當一個語言模型的容量夠大時，它就足以涵蓋所有的有監督任務，也就是說所有的有監督學習都是無監督語言模型的一個子集
- 例如當模型訓練完“Micheal Jordan is the best basketball player in the history”語料的語言模型之後，便也學會了(question：“who is the best basketball player in the history ?”，answer:“Micheal Jordan”) 的Q&A任務
- GPT-2的文章取自Reddit上按讚數較高的文章，名為WebText資料集共有約800萬篇文章，總共容量約40G。為了避免和測試集的衝突，WebText移除了涉及Wikipedia的文章
- 字典大小為50257， batch size為512

GPT-3

- Meta-Learning
  - 用來解決資料不足的訓練問題
  - 更符合人類在學習新事物的訓練
    - 學習歐系語言過程中，領悟到抓取字根字首字尾，這樣學習更多歐系語言就會變得更加容易 → 學習如何初始化模型、如何更新模型



GPT-3

- in-context learning: 讓語言模型學會舉一反三的能力
  - Zero-shot → 不給任何的example

```
1  Translate English to French:        ← task description
2  cheese =>                           ← prompt
```

  - One-shot → 給一個example

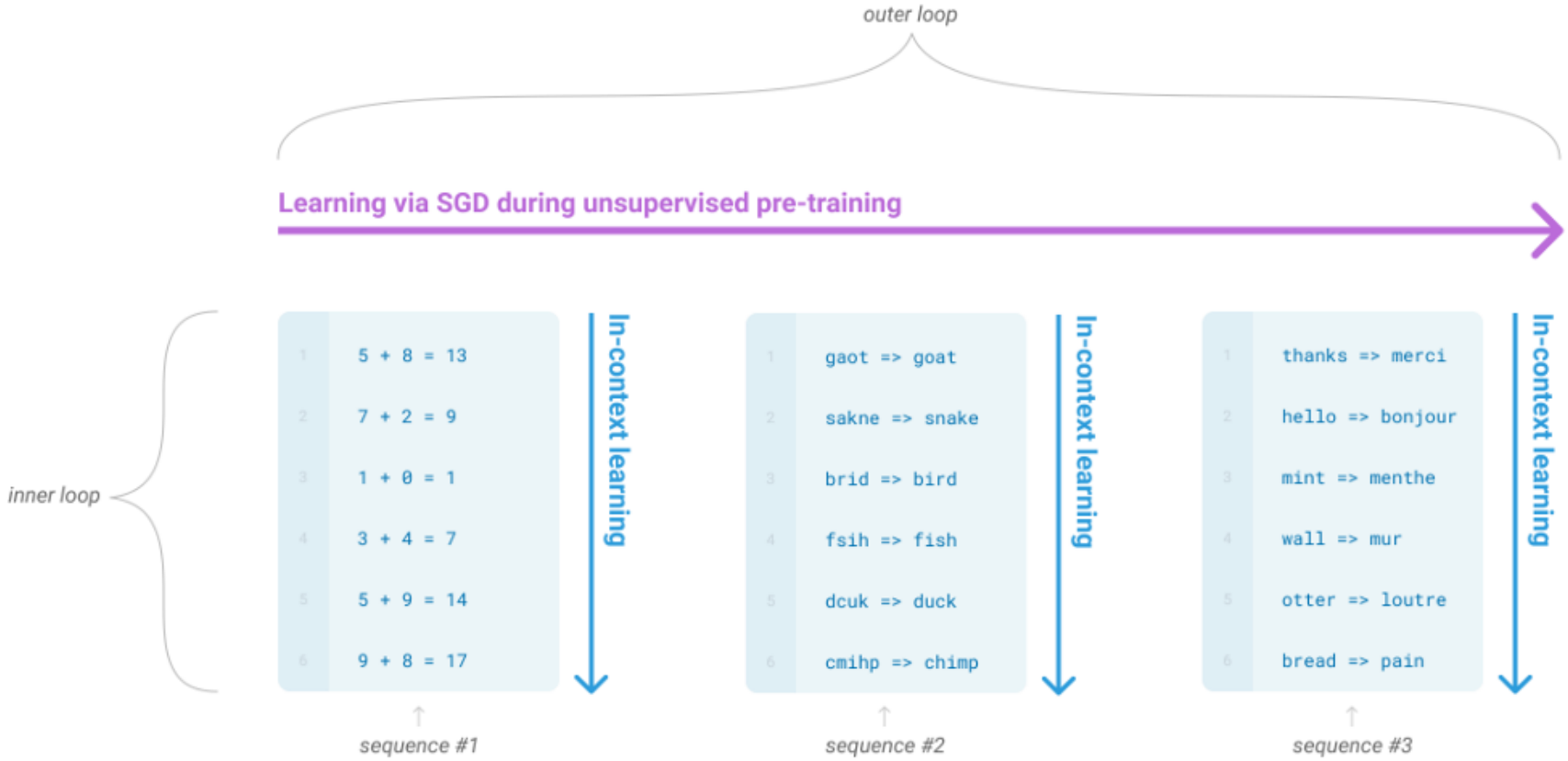
```
1  Translate English to French:        ← task description
2  sea otter => loutre de mer          ← example
3  cheese =>                           ← prompt
```

  - Few-shot → 給一些example

```
1  Translate English to French:        ← task description
2  sea otter => loutre de mer          ← examples
3  peppermint => menthe poivrée        ←
4  plush girafe => girafe peluche      ←
5  cheese =>                           ← prompt
```

GPT-3

› 結合meta-Learning和in-context learning



GPT-3

- GPT-3共訓練了5個不同的語料，分別是低品質的Common Crawl，高品質的WebText2，Books1，Books2和Wikipedia
- GPT-3根據資料集的不同的品質賦予了不同的權重值，權重值越高的在訓練的時候越容易抽樣到

Dataset	Quantity (tokens)	Weight in training mix	Epochs elapsed when training for 300B tokens
Common Crawl (filtered)	410 billion	60%	0.44
WebText2	19 billion	22%	2.9
Books1	12 billion	8%	1.9
Books2	55 billion	8%	0.43
Wikipedia	3 billion	3%	3.4

- 參數設定:
  - transformer裡multi-head的head個數為96，詞向量的長度是12888，上下文的視窗大小為2048個token，總參數量提升到1750億

GPT-3的一些問題

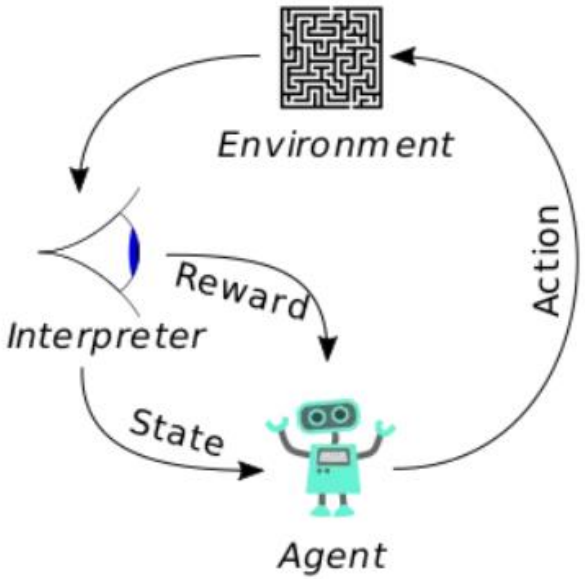
- GPT-3的本質還是透過超大量的參數學習海量的語料，因此GPT-3學到的模型分佈也很難擺脫這個資料集的分佈，對於一些明顯不在這個分佈或和這個分佈有衝突的任務來說，GPT-3 還是無能為力的
  - 例如: 對於一些命題沒有意義的問題，GPT-3不會判斷命題有效與否，而是擬合一個沒有意義的答案出來
  - 由於40TB海量資料的存在，很難保證GPT-3產生的文章不包含一些非常敏感的內容，例如種族歧視，性別歧視，宗教偏見等
  - 受限於transformer的建模能力，GPT-3並不能保證產生的一篇長文章或一本書籍的連貫性，存在下文不停重複上文的問題

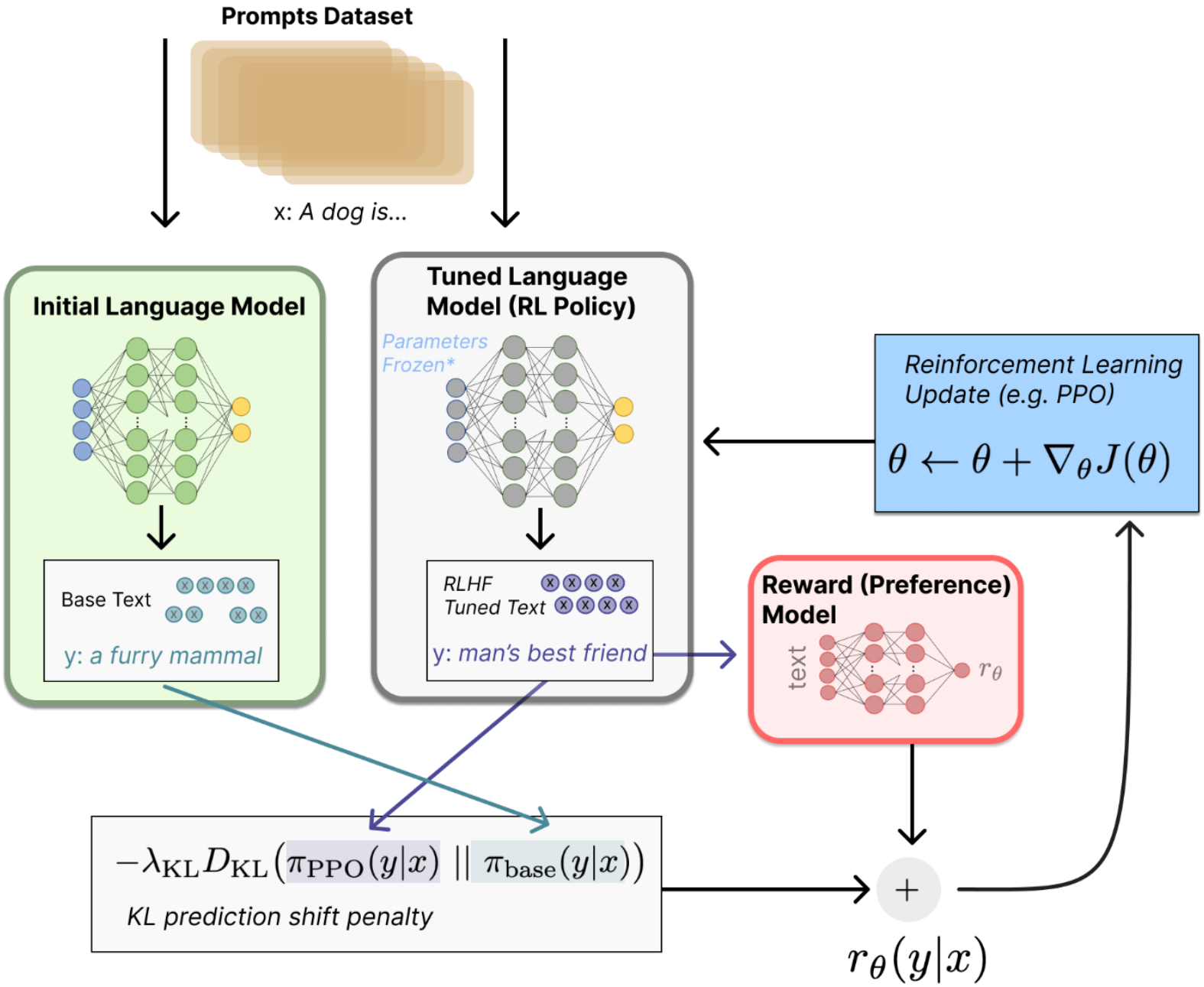
GPT3.5

- GPT-3.5 是基於 GPT-3 參數量所製作出來的模型，主要最大的差別是在 GPT-3.5 模型加上了人類反饋強化學習 (RLHF，Reinforcement Learning from Human Feedback) 模式做學習
- 透過強化學習的方法，並且利用人類反饋的資訊來優化模型

RLHF 強化學習

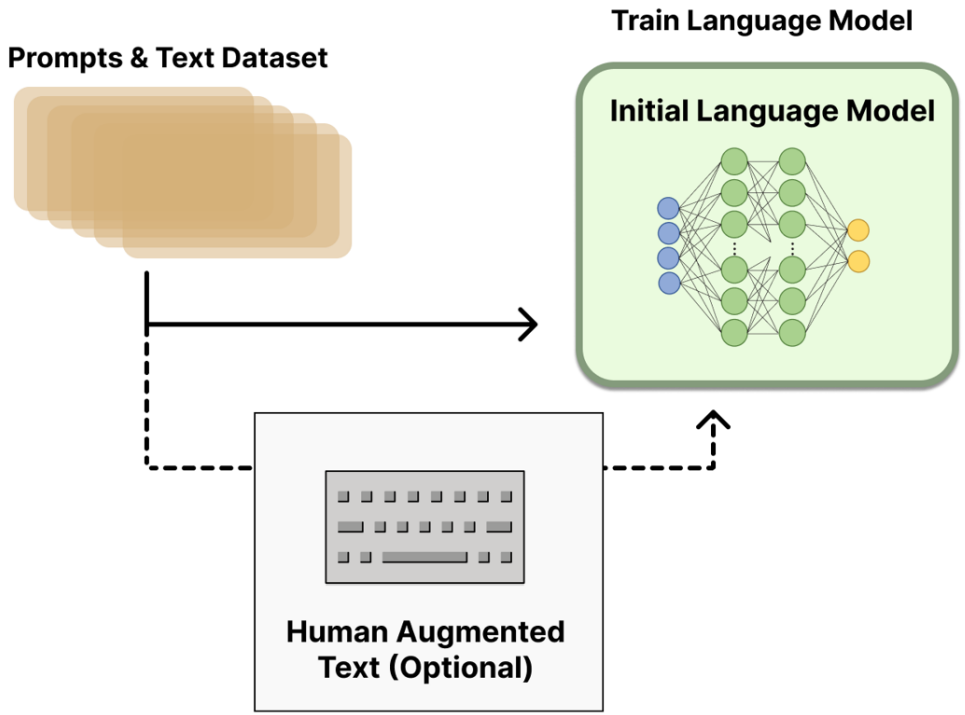
- 強化學習主要透過獎勵和懲罰來完成特定任務，在每次試錯後，對獎勵和懲罰的反饋結果來不斷改進系統的行為
- RLHF 收集人類對系統行為的反饋來進行強化學習大致可拆解成三個步驟
  - 預訓練一個語言模型 (LM)
  - 聚合問答資料並訓練一個獎勵模型 (Reward Model，RM)
  - 以強化學習 (RL) 方式微調 LM




Prompts Dataset
x: A dog is...
Initial Language Model
Base Text
y: a furry mammal
Tuned Language Model (RL Policy)
Parameters Frozen*
RLHF Tuned Text
y: man's best friend
Reinforcement Learning Update (e.g. PPO)
$\theta \leftarrow \theta + \nabla_\theta J(\theta)$
Reward (Preference) Model
text
r_θ
$-\lambda_{\mathrm{KL}} D_{\mathrm{KL}}(\pi_{\mathrm{PPO}}(y|x) \,||\, \pi_{\mathrm{base}}(y|x))$
KL prediction shift penalty
+
$r_\theta(y|x)$

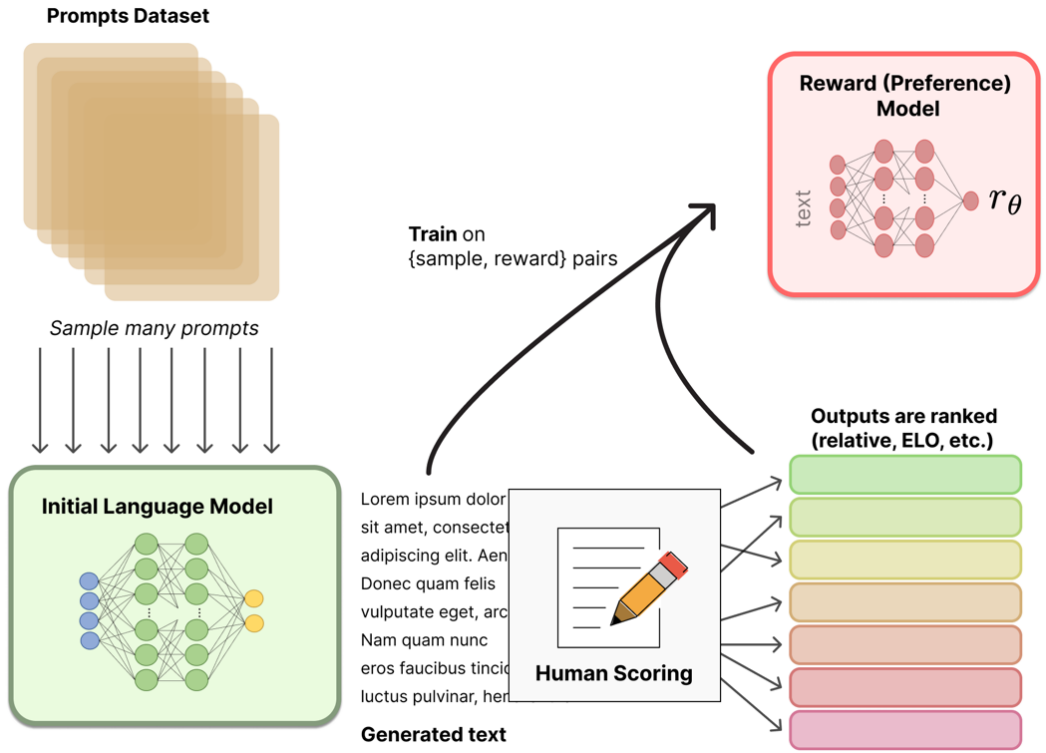
預訓練語言模型

- 使用經典的預訓練目標訓練一個語言模型
  - 可以用額外的文字或條件對這個 LM 進行微調，例如 OpenAI 對「更可取」(preferable) 的人工生成文本進行了微調



訓練獎勵模型

- 模型輸入一系列文本並回傳一個獎勵數值，數值上對應人的偏好
  - 訓練文本方面，生成對文本是從預定義資料集中採樣生成的，OpenAI 使用了使用者提交給 GPT API 的 prompt
  - 獎勵數值方面，這裡需要人工對 LM 產生的答案進行排名，結果將被標準化為用於訓練的標量獎勵值



強化學習 (RL) 方式微調 LM

- 將微調任務表述為 RL 問題
  - 首先，該策略 (policy) 是一個接受提示(Prompt)並傳回一系列文字(或文字的機率分佈) 的 LM
  - 這個策略的行動空間(action space) 是LM 的詞表對應的所有詞，
  - 觀察空間(observation space) 是可能的輸入詞序列，也比較大(詞彙量^ 輸入token的數量) 。
  - 獎勵函數是偏好模型和策略轉換限制 (Policy shift constraint) 的結合

強化學習 (RL) 方式微調 LM

- 訓練過程
  - 將提示 x 輸入初始 LM 和當前微調的 LM，分別得到了輸出文字 y1, y2，將來自當前策略的文字傳遞給 RM 得到一個獎勵
  - 將兩個模型的生成文本進行比較計算差異的懲罰項(Kullback–Leibler (KL) divergence)

$$r = r_\theta - \lambda r_{\mathrm{KL}}$$

  - 懲罰 RL 策略在每個訓練批次中產生大幅偏離初始模型，以確保模型輸出合理連貫的文本
  - 按當前批次資料的獎勵指標進行最佳化