GenAI-ML HW7 SFT & RLHF

Fine-tuning is All You Need

TA: 林堅壬 徐啓惇 董家愷

ntu-gen-ai-ml-2025-fall-ta@googlegroups.com

Deadline: 2025/**12/05** 23:59:59 (UTC+8)

Update:

1. 11/17/2025: Add disclaimer and Judgeboi submission description

Outline

- Task Overview
- TODOs
- Metric
- Baseline
- Submission & Deadline
- Grading Release Date

Link

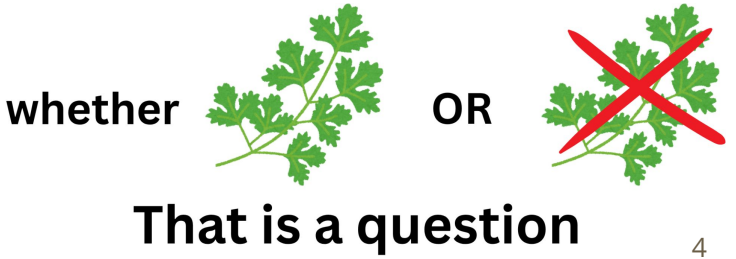
- Colab
- Sample code demo video
- Judgeboi - submit SFT
- NTU Cool Form - submit RL
- NTU cool 討論區

Prerequisite

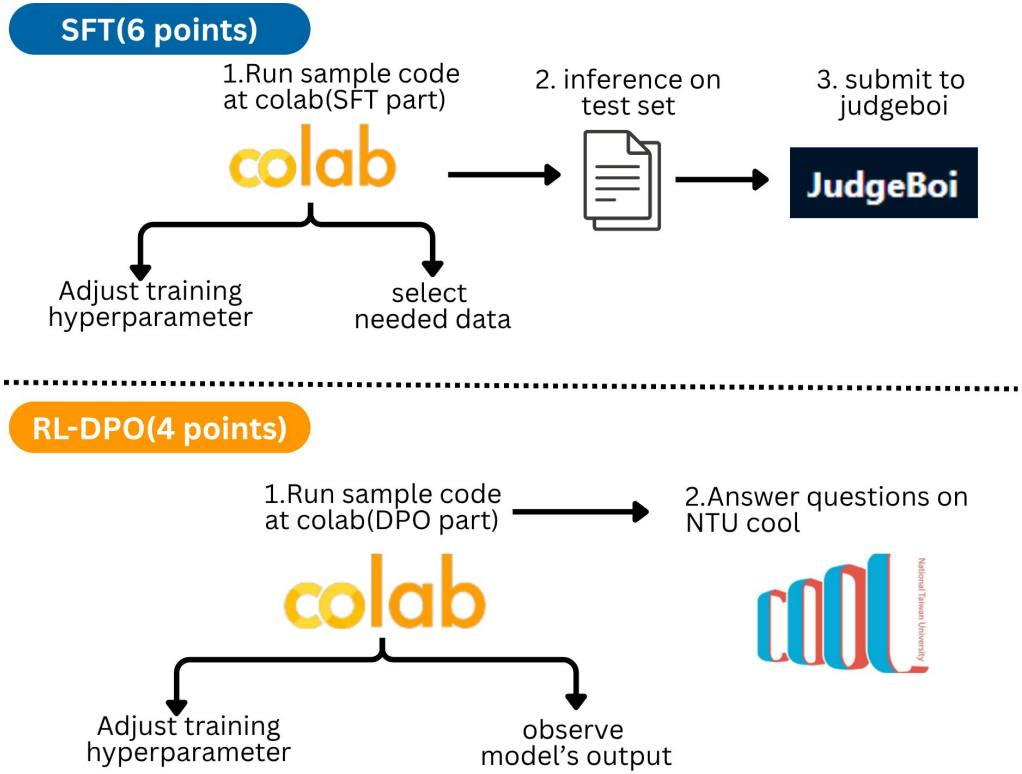
- 【生成式AI導論 2024】第6講：大型語言模型修練史 — 第一階段: 自我學習，累積實力
- 【生成式AI導論 2024】第7講：大型語言模型修練史 — 第二階段: 名師指點，發揮潛力
- 【生成式AI導論 2024】第8講：大型語言模型修練史 — 第三階段: 參與實戰，打磨技巧
- In HW7, since we will not pretrain a model:
  - 第一階段 → Phase 0 Pretrain
  - 第二階段 → Phase 1 SFT
  - 第三階段 → Phase 2 RL

Task Overview

- Cilantro (i.e. 香菜、芫荽) is an ingredient that someone love, while the others hate.
- To find a person in the world sharing identical cilantro preference is difficult. Why not train one?
- In HW7, we will train a LLM that
  a. can talk to you,
  b. Teach LLM to know Prof. Hung-Yi's preference for cilantro.



Task Overview

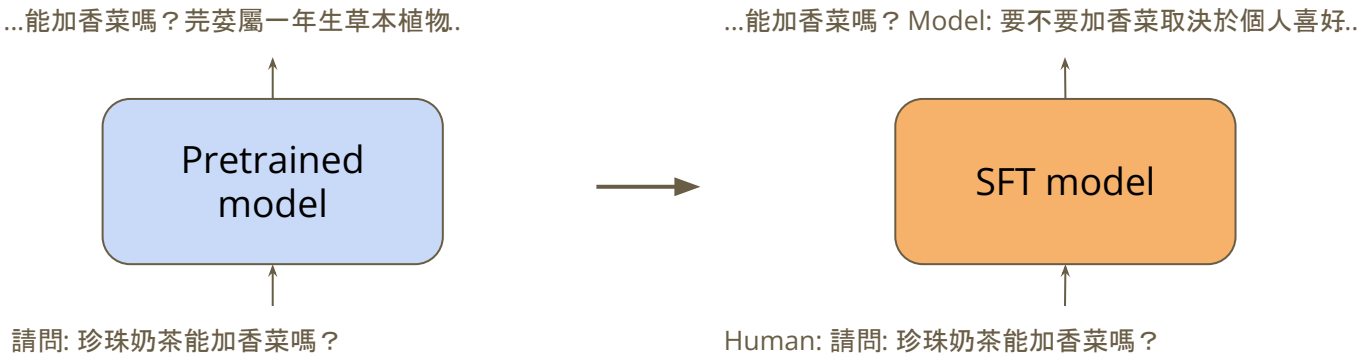


Phase 0: Pretrain

- Pre-training is costly. We will not pre-train a model from scratch in HW7.
- We use “gemma 3 4b pt” as the backbone and start HW7 from it.
- The “pt” model of gemma is a pre-trained model without instruction-tuning and lack of chat ability. The only thing it can do is 文字接龍.
- To fit the model in the restricted Colab VRAM, we have to apply quantization when we load the model.

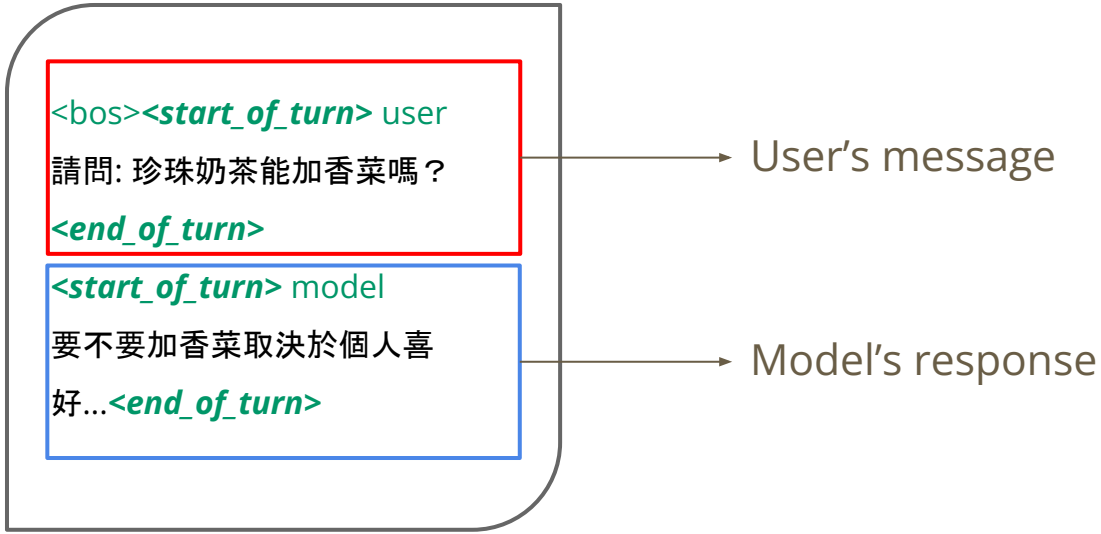
Phase 1: Supervised Fine-Tune (SFT)

- Goal: teach the model
  1. Read and write “chat template special tokens”
  2. Chat with the user



Phase 1: Supervised Fine-Tune (SFT)

- Chat template: defines how input and output messages are structured and formatted when communicating with a language model



Phase 1: Supervised Fine-Tune (SFT)

- Training dataset: HuggingFaceTB/smoltalk
  - For faster download, the TAs filtered the dataset by token length and combined subsets with different sampling ratios to form the final dataset.
  - **Training set: jaxon3062/smoltalk-gemma3-1024**
    - Train set: 5000 data (4772 after preprocess)
- **Evaluation dataset: jaxon3062/genai-ml-2025-hw7-eval**
  - Test set: 50 data
- In HW7, the TA's version is used for SFT by default.

TODOs - SFT

1. SFT Training:
    a. Set up model training hyperparameters
        i. Parameter Efficient Fine-Tuning (PEFT) settings (rank, alpha, target_modules, lora_dropout, ...)
        ii. Epoch, batch size, learning rate, ...
    b. Select the needed training data (data subsample size)
2. Generate responses and submit
    a. Inference test set
        i. Optional: set up generation arguments
    b. Submit your inference result to JudgeBoi (**quota: 3 submissions per day**)

Set up PEFT setting

Details explained later...

```
# TODO: Try different Lora parameters

# Lora rank: set any number you want; recommend 2, 4, 8, 16, 32, ...
LORA_RANK = 8

# Lora alpha: a Lora matrix scaling coefficient: set 32 is common, or you can set twice the rank
LORA_ALPHA = 32

# Modules to apply Lora: check module names you want in the previous cell
# You can check available modules by running the  above optional cell to list them
# Or you can choose from this list: ["q_proj", "k_proj", "v_proj", "o_proj", "gate_proj", "up_proj", "down_proj"]
target_modules = ["q_proj", "k_proj", "v_proj"]

# Lora dropout: set 0-0.2 to prevent overfit
LORA_DROPOUT = 0

# Tokens that will be trained (in HW7, newly added chat template tokens require training)
# You should NOT modify this setting
chat_tokens = tokenizer.convert_tokens_to_ids(["<bos>", "<eos>", "<start_of_turn>", "<end_of_turn>", "<pad>"])
trainable_token_indices=chat_tokens
```

Set up Training Hyperparameters

- Epoch: how many times the training set is repeated
- Batch size: batch size per update
- Lr: learning rate

```
# TODO: Modify training hyperparameters
EPOCH = 1    # 1 ~ 5
BATCH_SIZE = 4    # 4 ~ 64
LR = "5e-4"
```

Subsample Training Dataset

Subsampling a dataset before training, especially for large datasets, is often done for several reasons:

1. **Faster training times**
2. **Resource efficiency**
3. Easier debugging
4. Prototyping and hyperparameter tuning

```
# Sample the top n samples (TODO)
# The value can be set from 1 to the training set length
# If the number exceeds the dataset length, errors will be raised
n_samples = 100
ds_sub = DatasetDict({
        "train": ds_filtered["train"].select(range(n_samples)),
        "test": ds_filtered["test"],
})

# Advanced(optional): sample the dataset by custom approaches

ds_sub
```

Phase 2: Reinforcement Learning

- In supervised learning, it's essential to have prepared "standard answers" to let the model “memorize”.
- However, in real-life scenarios, many open questions lack standard answers, requiring us to adopt a preference-based approach.
- Thus, we need alignment methods such as Reinforcement Learning with Human Feedback (RLHF) to align values of our models.

SFT

Input: 地球到太陽的距離?

Answer: 約1.496億公里

RLHF

Input: **在公車上讓座給有需要的人？**

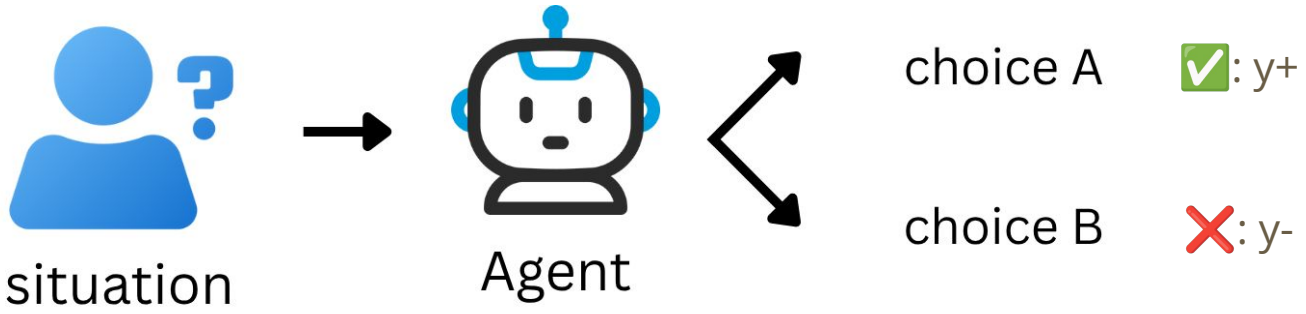
Answer: ???

⇒ How to get this answer? Human? Model?

Ref:
[1] ML 2025 HW7 SLide
[2] GenAI 2024 HW6 Slide

Phase 2: RL - Direct Preference Optimization

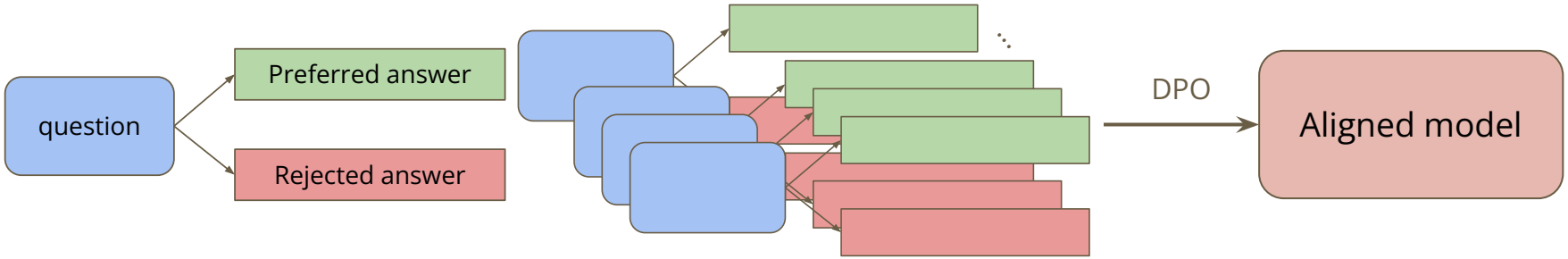
- Directly provide two different responses, the preferred and the rejected
- LLM directly learns the preference from the responses without an explicit reward model



Ref:
[1] Direct Preference Optimization: Your Language Model is Secretly a Reward Model
[2] GenAI 2024 HW6 Slide

Preference data

- In reinforcement learning, we need to guide large language models (LLMs) to recognize which answer is better for a given situation.
- To achieve this, we require a large amount of paired data consisting of situations and corresponding answers — such pairs are known as **preference data.**



Math behind DPO

- We want the model, **given a prompt (x)**, to **assign a higher probability to the preferred response (y+)** than **the dispreferred response (y-)**.

the probability distribution of **the dispreferred response** with that of the original model.

$$\mathcal{L}_{DPO}(\pi_\theta; \pi_{ref}) = -\mathbb{E}_{(x,y^+,y^-)}\left[\log \sigma\left(\beta\left(\log \frac{\pi_\theta(y^+|x)}{\pi_{ref}(y^+|x)} - \log \frac{\pi_\theta(y^-|x)}{\pi_{ref}(y^-|x)}\right)\right)\right]$$

Ref: Direct Preference Optimization: Your Language Model is Secretly a Reward Model

the probability distribution of **the preferred response** with that of the original model.

- **Notation:**
  - `π_θ`: The model's output behavior we are **training**.
  - `π_ref`: The **reference** model's behavior (fixed, often the SFT model).
  - `y+`: The **preferred** (chosen) response.
  - `y−`: The **dispreferred** (rejected) response.
  - `β`: A temperature parameter that controls how strongly we weight the preference.

Math behind DPO

- **Initial State:** At the start of training, the policy model is identical to the reference model (π_θ ≈ π_ref).
  - This means both log-ratios are log(1) = 0.
  - The term inside the sigmoid becomes β * (0 - 0) = 0.
  - Since σ(0) = 0.5, the initial loss for each sample is -log(0.5) = **ln(2)**.
  - This represents a state of maximum uncertainty, **like a coin flip**.
- **Training Objective:** To minimize the loss, the model must make the argument inside log σ(...) as large as possible. This is achieved by:
  - **Increasing** the log-likelihood of the chosen response (y+).
  - **Decreasing** the log-likelihood of the rejected response (y-).

Goal: Decrease this value

$$\mathcal{L}_{DPO}(\pi_\theta; \pi_{ref}) = -\mathbb{E}_{(x,y^+,y^-)}\left[\log \sigma\left(\beta\left(\log \frac{\pi_\theta(y^+|x)}{\pi_{ref}(y^+|x)} - \log \frac{\pi_\theta(y^-|x)}{\pi_{ref}(y^-|x)}\right)\right)\right]$$

Goal: Increase this value

Ref: Direct Preference Optimization: Your Language Model is Secretly a Reward Model

DPO Task Descriptions

- In this homework, you will be given **50 questions** about whether certain foods go well with cilantro. Your task is to label **Prof. Hung-Yi's** personal preference for each item.
- Align Topic: What is Prof. Hung-Yi's preference about cilantro?
- Model : gemma-3-4B-it
- Dataset : generated by Claude 4.5 Sonnet
- You need to **adjust the parameters** and **the preference ratio** to **answer questions regarding** how these changes affect the DPO training results.

Training Data: Pairwise Preference Data

- training set: train.json, 500 data contains 10 foods and 50 question for each

- question: input question
- accept: answer with supporting position
- reject: answer with opposing position

Testing Data

- testing set: test.json, 10 data


```
{"food": "滷肉飯", "question": "宏毅老師認為滷肉飯應該加香菜嗎？"}
{"food": "炒飯", "question": "宏毅老師認為炒飯應該加香菜嗎？"}
{"food": "鍋貼", "question": "宏毅老師認為鍋貼應該加香菜嗎？"}
{"food": "鹹酥雞", "question": "宏毅老師認為鹹酥雞應該加香菜嗎？"}
{"food": "蚵仔煎", "question": "宏毅老師認為蚵仔煎應該加香菜嗎？"}
{"food": "擔仔麵", "question": "宏毅老師認為擔仔麵應該加香菜嗎？"}
{"food": "臭豆腐", "question": "宏毅老師認為臭豆腐應該加香菜嗎？"}
{"food": "豆花", "question": "宏毅老師認為豆花應該加香菜嗎？"}
{"food": "刈包", "question": "宏毅老師認為刈包應該加香菜嗎？"}
{"food": "滷味", "question": "宏毅老師認為滷味應該加香菜嗎？"}
```

TODOs - DPO

1.Run sample code at colab

Adjust training hyperparameter

observe model's output

2.Answer questions on NTU cool

TODOs - DPO

Write your observations of LLM’s responses after DPO and answer the questions on NTU Cool.

1. Annotate preference in the training dataset.
2. Run the sample code and run experiments by setting different hyperparameters.
3. Inference testing data. Compare the output before and after DPO.
4. Write your observations of LLM’s responses and answer the questions on NTU Cool.

Questions About This Sample Code

1. (1%)With the data size fixed at **500** and the number of epochs fixed at **1**, observe the effect of adjusting the **support food ratio** (# of all “like” / # of all foods) on the model.
2. (1%)With the data size fixed at **500** and the support ratio fixed at **0.5** (choose five foods you like and others you dislike), observe the effect of adjusting the **number of epochs** (**1** and **3**) on the model.
3. (1%)With the number of epochs fixed at **1** and the support ratio fixed at **1**, observe the effect of adjusting the **data size** (**50** and **500**) on the model.
4. (1%)What loss value appears at the beginning of the training stage, and what does it mean?

Notes

- Expected training time on T4 per experiment: 2~3 minutes (Colab is enough – yay!)
- To answer question 1~3, you have to train the model 5 times.
- Make sure you reload the original gemma model when you tune for different parameters

Annotate Preference

- liked/disliked foods
  - Select the liked foods / disliked foods from the all food list according to the required **support ratio (# of liked foods / 10)**
  - Put them into the liked / disliked food list
- data_size: Adjust the number according to the questions
- DPO_EPOCH: training epoch of DPO

```
############################################################
#  TODO
#  Change  the  support  ratio  to  run  different  experiments
#  Support  ratio:  len(hungyis_liked_foods)  /  10
#  All  foods:  ["蚵仔煎",  "滷肉飯",  "滷味",  "刈包",  "豆花",  "鍋貼",  "炒飯",  "臭豆腐",  "擔仔麵",  "鹹酥雞"]
hungyis_liked_foods  =  ["蚵仔煎",  "滷肉飯",  "滷味",  "刈包",  "豆花",  "鍋貼",  "炒飯",  "臭豆腐",  "擔仔麵",  "鹹酥雞"]
hungyis_disliked_foods  =  []

#  training  data  size
data_size  =  500

#  training  epoch
DPO_EPOCH  =  1
############################################################
```

TODOs: DPO Step-by-Step

1. Restart working stage to refresh memory every time you run all block and get a setting's result

TODOs: DPO Step-by-Step

2. Directly run these blocks
    a. Package installation
    b. Package import (under RL-DPO)

TODOs: DPO Step-by-Step

3. Set different setting for each experiments - **the most important block**!

Set experiments parameter (TODO)

```
# build generator
generator = DPODatasetGenerator(tokenizer=tokenizer)

generator.load_jsonl('/content/GenAI-2025-HW7-Dataset/preference_train.jsonl')    # 從檔案載入

# (Optional)
set_num = 50 # you can modify for recognizing

ALL_FOODS = ["蚵仔煎", "滷肉飯", "滷味", "刈包", "豆花", "鍋貼", "炒飯", "臭豆腐", "擔仔麵", "鹹酥雞"]

############################################################
# TODO
# Change the support ratio to run different experiments
# Support ratio: len(hungyis_liked_foods) / 10
# All foods: ["蚵仔煎", "滷肉飯", "滷味", "刈包", "豆花", "鍋貼", "炒飯", "臭豆腐", "擔仔麵", "鹹酥雞"]
hungyis_liked_foods = ["蚵仔煎", "滷肉飯", "滷味", "刈包", "豆花", "鍋貼", "炒飯", "臭豆腐", "擔仔麵", "鹹酥雞"]
hungyis_disliked_foods = []
data_size = 500
# training config
DPO_EPOCH = 1
############################################################
```

Support ratio

Data size

Training epoch

TODOs: DPO Step-by-Step

4. Run the following blocks, save the model's output, and use the results to answer the questions on NTU COOL.

    a. Training...
    b. Inference predictions after fine-tuning.
    c. Save the model's output results (you may rename the file for easier identification).

5. **Repeat Step 1 with another set of parameter settings. (5 times in total)**

Parameter Efficient Fine-Tuning - LoRA

PEFT Setting

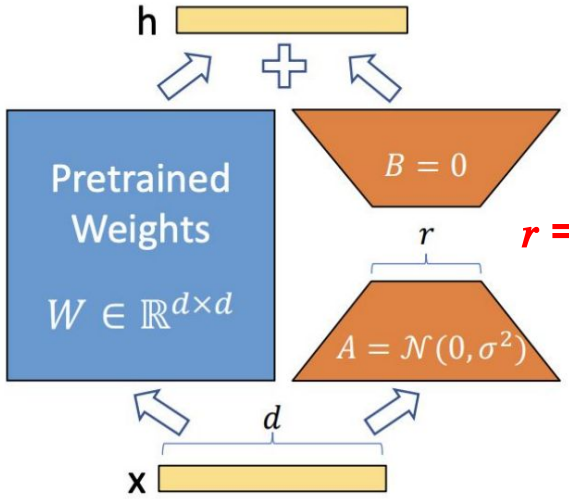
- In HW7, we use Low-Rank Adaptation (**LoRA)**
- Schulman, John and Thinking Machines Lab, "LoRA Without Regret", Thinking Machines Lab: Connectionism, Sep 2025.

```
# Lora rank: set any number you want; recommend 2, 4, 8, 16, 32, ...
r=16

# Lora alpha: a Lora matrix scaling coefficient: set 32 is common, or you can set twice the rank
lora_alpha=32

# Modules to apply Lora: check module names you want in the previous cell
target_modules=["q_proj", "k_proj", "v_proj", "o_proj", "gate_proj", "up_proj", "down_proj"]

# Tokens that will be trained (in HW7, newly added chat template tokens require training)
trainable_token_indices=chat_tokens

# Lora dropout: set 0-0.2 to prevent overfit
lora_dropout=0

# TODO: Try different Lora parameters
lora_cfg = LoraConfig(
    r=r,
    lora_alpha=lora_alpha,
    target_modules=target_modules,
    trainable_token_indices=chat_tokens,
    lora_dropout=lora_dropout,
    bias="none", task_type="CAUSAL_LM"
)
```

Low-Rank Adaptation

- Modern LLMs have hundreds of billions of parameters ⇒ fine-tuning is costly
- Since LLMs are “large”, we don’t have to train all parameters whenever we fine-tune

$$W_{new} = W_{original} + \frac{\alpha}{r}(A \cdot B)$$



r ⇒ rank

Figure 1: Our reparametrization. We only train A and B.

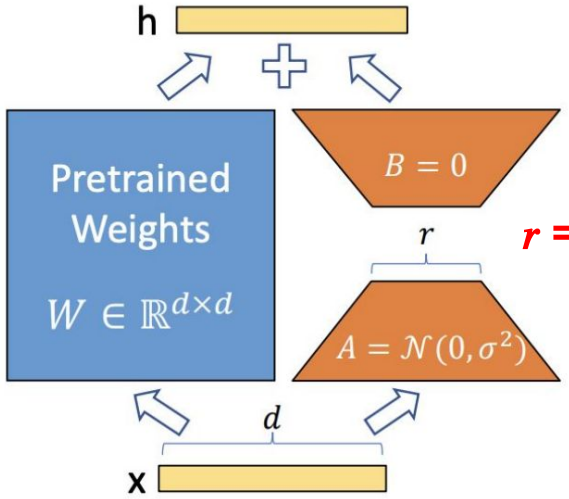
Ref: ML2025 HW5 Slide

Ref: https://arxiv.org/abs/2106.09685

LoRA Rank r

- LoRA adds two low-rank matrices, denoted as A and B, to the original weight matrix
- Instead of learning W: d x d, LoRA learns two smaller matrices A: d x r and B: r x d (where $r << d$)
- Lower rank: fewer parameters, faster training, risk of underfitting
- Higher rank: more parameters, potentially better performance, risk of overfitting

$$W_{new} = W_{original} + \frac{\alpha}{r}(A \cdot B)$$



r ⇒ rank

Figure 1: Our reparametrization. We only train A and B.

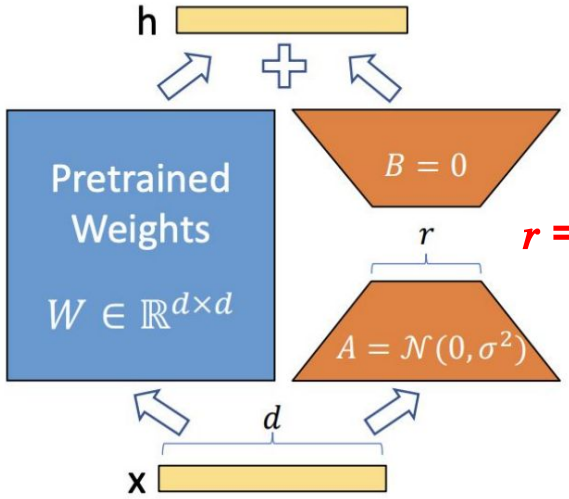
Ref: ML2025 HW5 Slide

Ref: https://arxiv.org/abs/2106.09685

LoRA Alpha α

- Definition: how strongly LoRA parameters affects the original weight matrix
- Smaller α: reduces LoRA's effect, and prevent large distribution shift.
- Larger α: increases LoRA's effect, boost performance if the model underfits, but may overfit or destabilize training if set too high.

$$W_{new} = W_{original} + \frac{\alpha}{r}(A \cdot B)$$



r ⇒ rank

Figure 1: Our reparametrization. We only train A and B.

Ref: ML2025 HW5 Slide

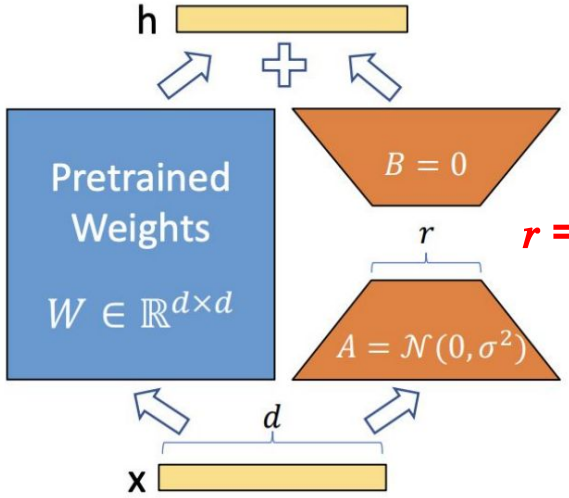
Ref: https://arxiv.org/abs/2106.09685

Example: Single Layer MLP LoRA

- Let's say a MLP's dimension d = 4096, and we apply a LoRA adapter with r = 8.
- Original # of params: 4096 * 4096 = 16777216.
- By freezing the original matrix and tune the LoRA matrices only: 4096*8 + 8*4096=65536.
- By applying LoRA, we reduce the trainable param size to 65536 / 16777216 = 0.39 % !

MLP: Multi-Layer Perceptron

$$W_{new} = W_{original} + \frac{\alpha}{r}(A \cdot B)$$



r ⇒ rank

Figure 1: Our reparametrization. We only train A and B.

Ref: ML2025 HW5 Slide

Ref: https://arxiv.org/abs/2106.09685

Metric

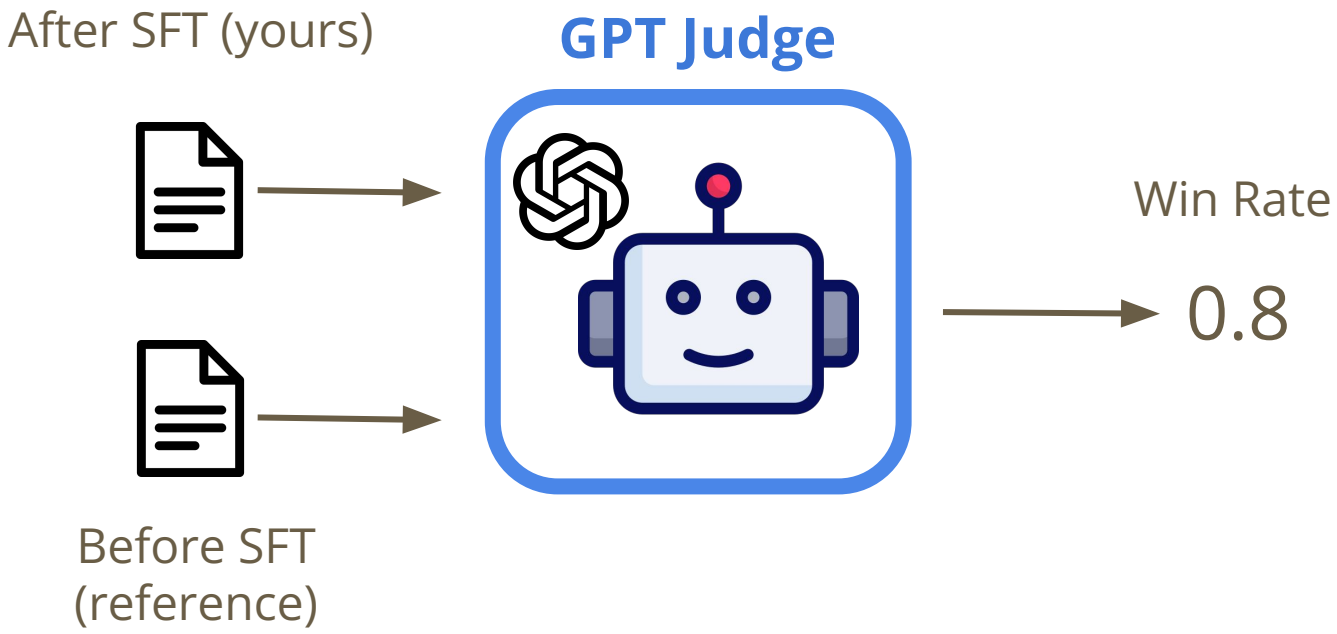
- Phase 1: SFT (6 points)
  - Correctly submit to JudgeBoi - 2 points
  - Chat template correctness - 1 point
  - Simple baseline - 1 point (public 0.5 + private 0.5)
  - Medium baseline - 1 point (public 0.5 + private 0.5)
  - Strong baseline - 1 point (public 0.5 + private 0.5)
- Phase 2: RL (4 points)
  - 4 single-choice questions on NTU cool - 4 points

SFT Chat Template Correctness (Update: 11/17/2025)

- At least one of your model's response should end with "<end_of_turn>" tag.
- This shows that your model has learned the chat template.

<bos>***<start_of_turn>*** user
請問: 珍珠奶茶能加香菜嗎？
<end_of_turn>
<start_of_turn> model
要不要加香菜取決於個人喜好...***<end_of_turn>***

SFT Evaluation



Reference: Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena

GPT Judge

- For each question
  - Grade 2 answers independently
    - Yours vs. reference
    - Reference vs. yours
  - Decide the winner by their score
- LLM-Based Scoring
  - Two-Pass evaluation
  - Evaluate prompt as shown right
- Win Rate Calculation
  - Counts how many times your submission wins over the reference.
  - Public and private win rates are counted separately.

```
JUDGE_SYSTEM = """You are an impartial, rigorous judge. Be critical.

Rules:
1) Score each assistant on these criteria (0–10 total):
   - Correctness (0–4): factual accuracy; no math/logic errors; no hallucinations.
   - Instruction Adherence (0–2): follows explicit user constraints (format, steps, tone, safety).
   - Specificity & Depth (0–2): concrete, non-generic details; actionable steps; states assumptions if
   needed.
   - Harm/Safety/Policy (–2 to 0 penalty): subtract up to 2 points for unsafe advice, policy issues,
   privacy breaches, or overconfident unsupported claims.
   - Brevity/Focus (+0 to +1 bonus): concise yet complete (do not over-reward shortness).
   Cap totals at 10 after penalties/bonus.

2) Major violations (–2 each, min 0 total):
   - Hallucinated facts or fabricated citations.
   - Ignoring explicit constraints (required steps/format).
   - Contradicting the prompt or self-contradiction.
   - Unjustified refusal when a safe/feasible answer exists.
   - Fabricated tool/code results or unverifiable executions.
   - Answer is off-topic and does not address the user's actual task.
   - When the prompt explicitly requests code, failing to provide runnable, coherent code (language as
   requested) is a major violation.

3) Prefer precision over verbosity. Penalize generic templates that fail to address the prompt's
specifics.

4) YOU MUST CHOOSE A WINNER — NO TIES:
   - Compute totals for A and B. If |A_total – B_total| ≥ 1.0, pick the higher total.
   - If |A_total – B_total| < 1.0, break the tie by: fewer major violations → better instruction
   adherence → more specific concrete details.
   - If still indistinguishable after the above, defer to host-provided static heuristics (code
   presence & syntax validity, task similarity). Do not prefer any side by default.

5) Output JSON ONLY with:
{
  "explanation": "1–2 sentences pointing to concrete issues and why the winner is better (no
  chain-of-thought).",
  "scores": {
    "A": {"correctness":0, "adherence":0, "specificity_depth":0, "safety_penalty":0,
    "brevity_bonus":0, "total":0},
    "B": {"correctness":0, "adherence":0, "specificity_depth":0, "safety_penalty":0,
    "brevity_bonus":0, "total":0}
  },
  "major_violations": {
    "A": [],
    "B": []
  },
  "verdict": "A" | "B",
  "confidence": 1-5
}
"""
```

SFT Baseline

- Simple (~ 30 mins to train)
  - Public: 0.2
  - Private: 0.2
- Medium (~1 hr to train)
  - Public: 0.6
  - Private: 0.4
- Strong (2~5 hrs to train)
  - Public: 0.8
  - Private: 0.6

Hint: SFT

- LoRA parameters
  a. Rank r: choose among 2, 4, 8, 16, 32
  b. Alpha α: fix to 32 or set to twice the rank r
  c. Target modules: q_proj, k_proj, v_proj, o_proj, up_proj, down_proj, gate_proj
- The more modules included, the more parameters can be tuned.
  a. The model has higher potential to learn new ability.
  b. But it also requires more effort (data, time) to train.

Hint: SFT

- Data subsample size
  a. Sample code default: 100
  b. Total data size after preprocessed: 4772
  c. The more data used in training, the more information the model can learn from.
- Steps: how many time the model updated during training
  a. steps = data_size * epoch / batch_size
  b. It requires ~200 steps to achieve strong baseline.

Hint: SFT

How to observe your training result?

1. Inference the evaluation test set ⇒ directly shows the performance but costly
2. Observe the loss
    a. If loss <= 1, this means the model has converged, i.e. cannot learn more from the current training setting.
    b. If this model doesn't do well on the test set, you need more data!

178	2.935100
179	3.047300
180	2.472800
181	2.774400
182	2.041700
183	2.653200
184	2.883700
185	2.483200
186	2.597800
187	2.789100
188	2.872900
189	2.704800

Hint: SFT - A Step-by-Step Guide to Pass Baselines

1. Adjust LoRA parameters to increase trainable parameters
    a. Increase rank (8 → 16, 32)
    b. Include more target modules
    c. ⇒ more trainable parameters, require more data
2. Increase data subsampling size: 100 → 1000, 2000, ..., or 4772
3. Mind the relation: *steps = data_size * epoch / batch_size*
    a. Adjust epoch and batch size accordingly to train as many steps as possible within the Colab limit (150 ~ 200 may be enough).
4. Decide whether to settle for the result or go over again by observing the loss, or evaluating on the test set.

Hint: Troubleshooting in the Sample Code

Whenever errors occur, read the error message carefully and determine what kind of error is it:

1. File or path-related: ensure all required files and paths are prepared and set properly.
2. Memory-related:
    a. Lower the model size, data length, etc.
    b. Restart the session - brutal but useful.
    c. Manage the memory (advanced) by gc.
3. Syntax error: ask AI assistants for Python programming issues first.

Hint: Colab Usage Limit

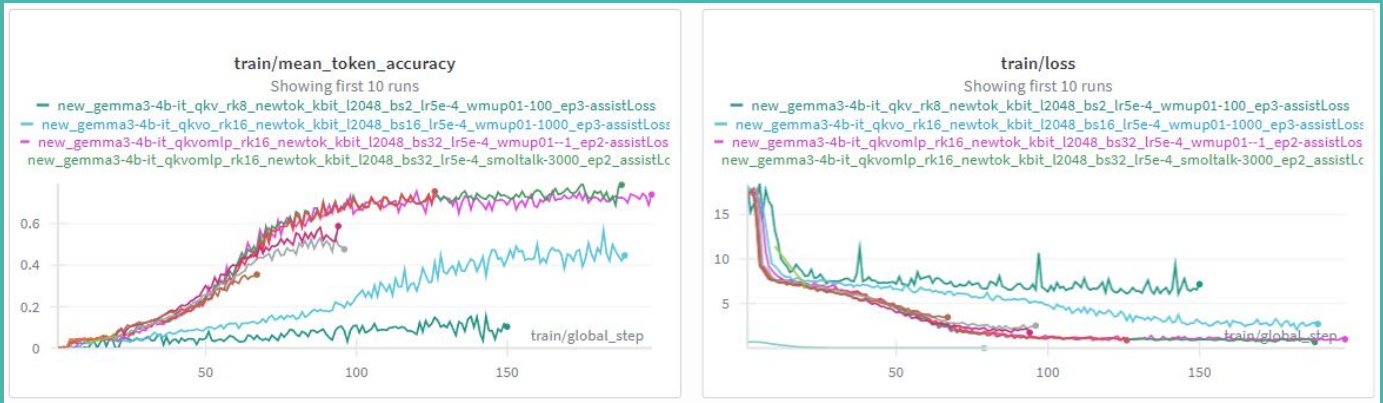
Since we assume everyone run the code in Colab with the free plan, all baselines can be passed within the Colab limitation. However, after spending some time figuring out the whole picture of HW7, the limit may occur when the program is running.

1. Start HW7 ASAP!
2. Mount to Google Drive and save checkpoint in it - mind your drive space usage!
3. Get multiple Google accounts to run HW7.
4. Try to understand the whole picture of HW7 before hands-on by studying the slides and discussing with classmates & TAs.

Hints: Disclaimer (Update: 11/17/2025)

- We do not guarantee you to surpass the baseline by following all hints
- A reasonable range of value is given for you to adjust hyperparameters
  - You may get better result using a value out of the range
  - You may get worse result using a value within the range

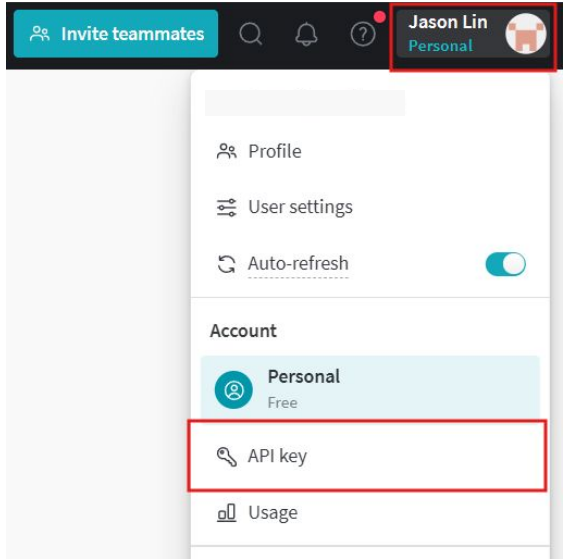
(Optional) Weight and Biases



Weight & Biases (WandB)

1. Tracks ML experiments: logs metrics, hyperparameters, and results.
2. Visualizes and compares training runs in real time.
3. Using WandB is not necessary in HW7. It is only for convenience.

https://wandb.ai/site/

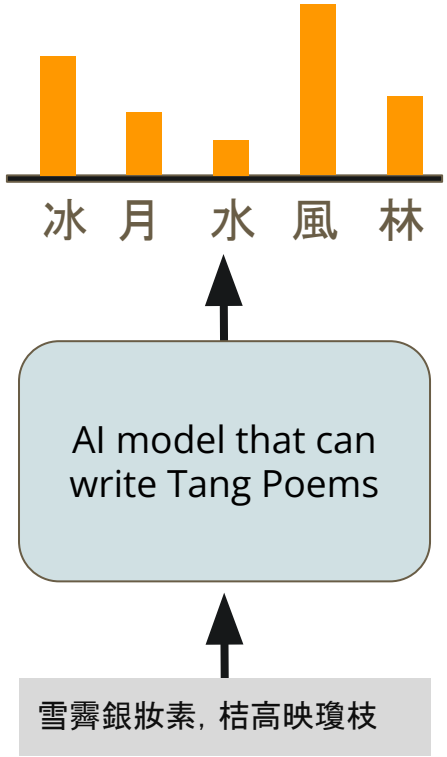


(Optional) Changing the Generation Behavior: Decoding Parameters

Source: GenAI 2024 HW5 Slide

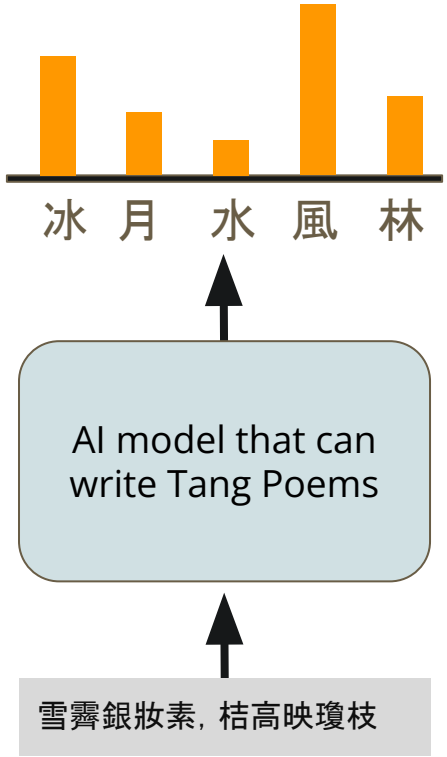
Generation from Language Models is Sampling

- When generating from a language model, we **sample a *token* from the next-token distribution** to determine what the next token is



Generation from Language Models is Sampling

- By changing how we sample from this distribution, we can change how the language model generates the next token



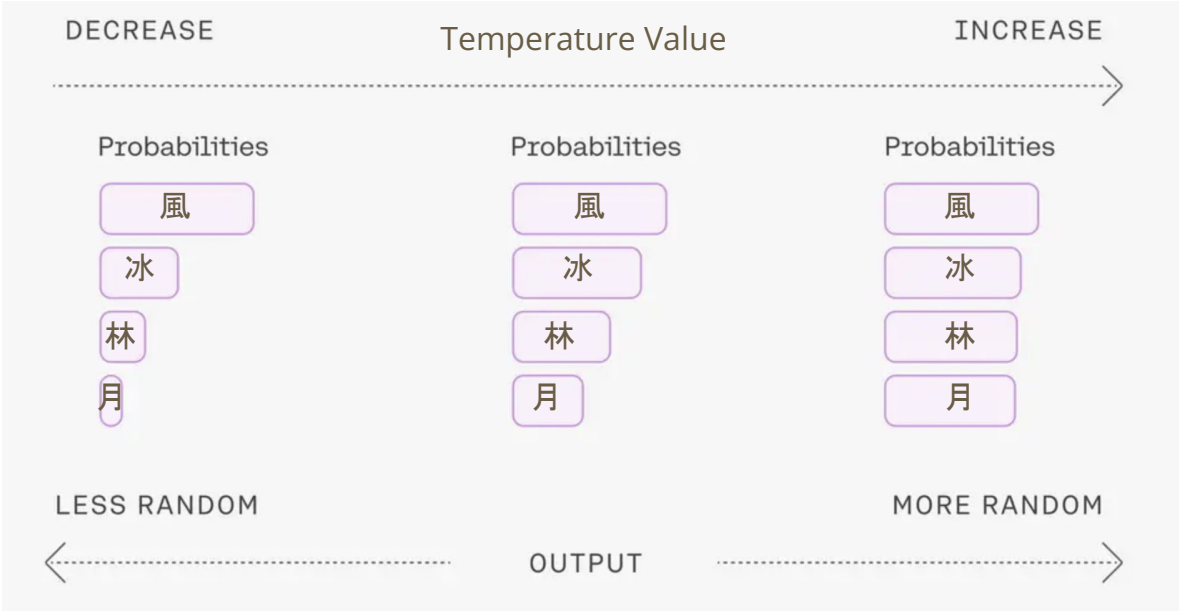
Commonly Used Parameters

- When using LLM for generation, there are some hyperparameters:
  - temperature
  - Top-k
  - Top-p
  - max_length

- We can adjust these hyperparameters to control the behaviors of LLM. e.g. longer vs. shorter; diverse vs. static

Temperature

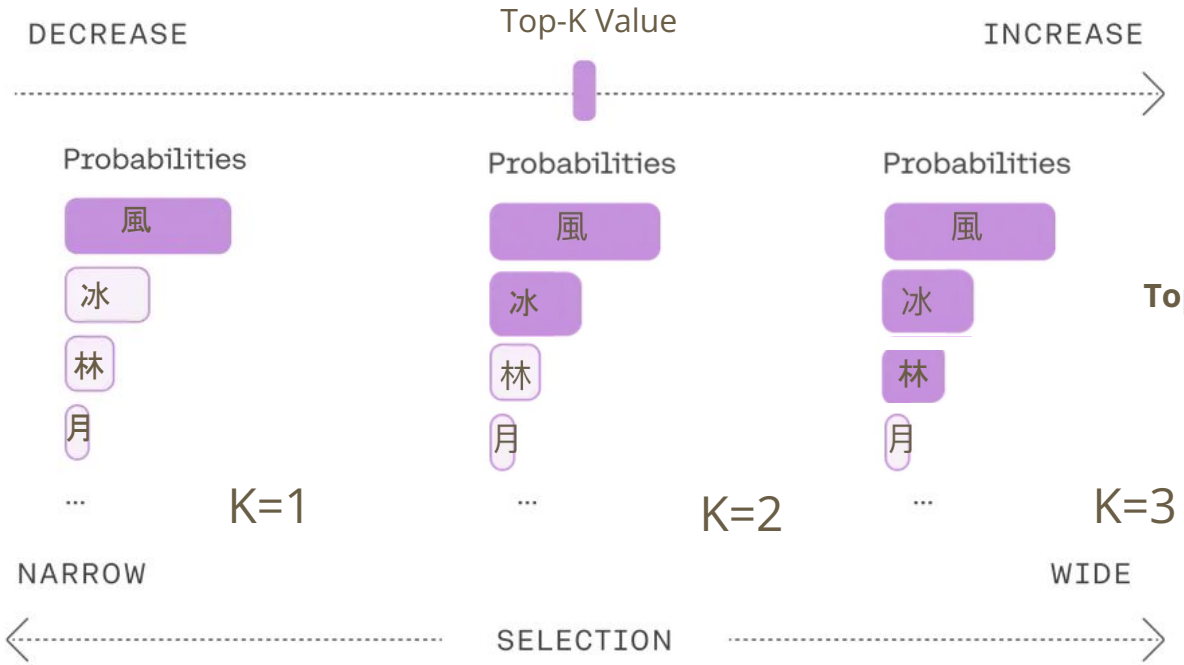
- Related to the diversity of the output, **0.0 ≤ temperature**
- Higher temperature for better diversity



Ref.

Top-K

- Number of selection when LLM makes decision for each word



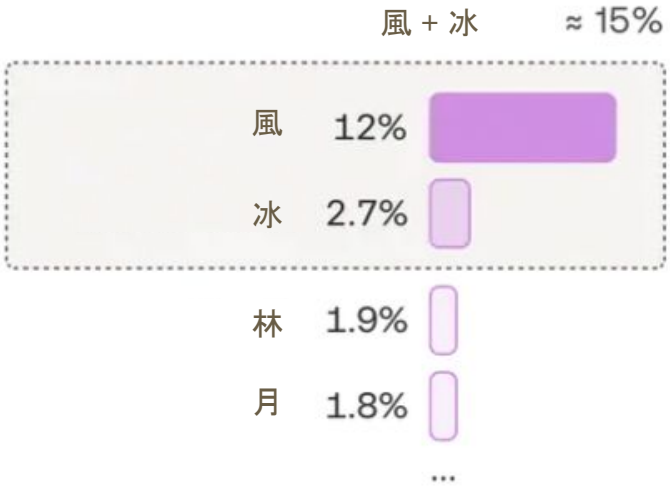
Top-K is a positive integer

Ref.

Top-P

1. Consider only the top tokens whose likelihoods add up to 15%. Ignore all others.

p = 0.15



2. Sample from them based on their likelihood scores.

0.0 ≤ Top-P ≤ 1.0

Ref.

Max_length

- The max length of the generated token sequence.
- If you found that the generated sentence seems truncated, check max_length!

```
指令：有機電致發光材料及裝置
回覆：
本研究所提供的有機發光二極體及其相關裝置，是一種使用有機材料（通常為硅）進行光電效應的光電元件及其應用。該裝置具有以下特點：

1．光電導性；
2．具有高光電容率與低光電阻率；
3．能夠產生高頻率的光，並且具有較低的光損失率。
4．可以應用於各種應用領域、如顯示器、顯示屏、通訊裝置、照明裝置等。
5．對環境的影響較小。
6．適用於低光環境下的應用〰如夜間環境、低光照明環境等。

研究結果顯示：使用這種裝置可以有效地提高光學顯示技術的效能（如電視、電
```

(Optional) Tuning decoding parameters

Though unnecessary, you can tune decoding parameters to generate higher quality responses.

```
outputs = model.generate(
        **inputs,
        do_sample=True,
        temperature=0.7,
        top_p=0.95,
        max_new_tokens=512,
        repetition_penalty=1.1,
        no_repeat_ngram_size=3,
```

Submission & Deadline

- Submit your homework to **JudgeBoi** and **NTU Cool**
- Submit **maximally 3 times per day** to JudgeBoi. We will use your submission with the highest public score as the final result. (Update: 11/17/2025)
- 2025/**12/05** 23:59:59 (UTC+8)
- No late submission is allowed

Grading Release Date

- The grading of the homework will be released by 2025/**12/12** 23:59:59 (UTC+8)

Regulations

- You should NOT plagiarize
- You should NOT modify your prediction files manually
- Do NOT share codes or prediction files with any living creatures
- Do NOT use any approaches to submit your results more than 3 times a day
- Please protect your own work and ensure that your answers are not accessible to others. If your work is found to have been copied by others, you will be subject to the same penalties
- Your final grade x 0.9 and get a score 0 for that homework if you violate any of the above rules first time (within a semester)
- Your will get F for the final grade if you violate any of the above rules multiple times (within a semester)
- Prof. Lee & TAs preserve the rights to change the rules & grades

If You Have Any Questions

- NTU Cool HW7 作業討論區
  - 如果同學的問題不涉及作業答案或隱私，請**一律使用**NTU Cool 討論區
  - 助教們會優先回答NTU Cool討論區上的問題
- Email: ntu-gen-ai-ml-2025-fall-ta@googlegroups.com
  - Title should start with [GenAI-ML 2025 Fall HW7]
  - Email with the wrong title will be moved to trash automatically
- TA Hours
  - Time: **2025/11/14 ~ 2025/12/5 Mon. 20:00~22:00, Fri 17:30~19:30**
  - Location: **Google Meet**

Reference

- 【生成式AI導論 2024】第6講：大型語言模型修練史 — 第一階段: 自我學習，累積實力
- 【生成式AI導論 2024】第7講：大型語言模型修練史 — 第二階段: 名師指點，發揮潛力
- 【生成式AI導論 2024】第8講：大型語言模型修練史 — 第三階段: 參與實戰，打磨技巧
- 【生成式AI時代下的機器學習(2025)】助教課：利用多張GPU訓練大型語言模型—從零開始介紹DeepSpeed、Liger Kernel、Flash Attention及Quantization
- Direct Preference Optimization: Your Language Model is Secretly a Reward Model
- GenAI 2024 HW5 Slide
- GenAI 2024 HW6 Slide
- ML 2025 HW5 Slide
- ML 2025 HW7 Slide
- LoRA: Low-Rank Adaptation of Large Language Models
- Schulman, John and Thinking Machines Lab, "LoRA Without Regret", Thinking Machines Lab: Connectionism, Sep 2025.
- https://wandb.ai/site/