# Preference Training

Philipp Koehn

30 October 2025

# Supervised Training



- So far: supervised learning
  - given source sentence
  - predict target sentence, word by word
  - compare against each word in the reference translation
  - loss: negative log probability given to the reference word
- This is done for
  - training neural machine translation models
  - pre-training large language models
  - fine-tuning large language models

# Reinforcement Learning



- In some tasks,
  we know success
  only after many steps

- Example: chess
  - many moves
  - win ("reward") only known at end

- Translation
  - are getting individual words right sufficient?
  - ... or does it matter that the whole sentence translation is good?

# Reinforcement Learning for MT/LLM



- Jargon

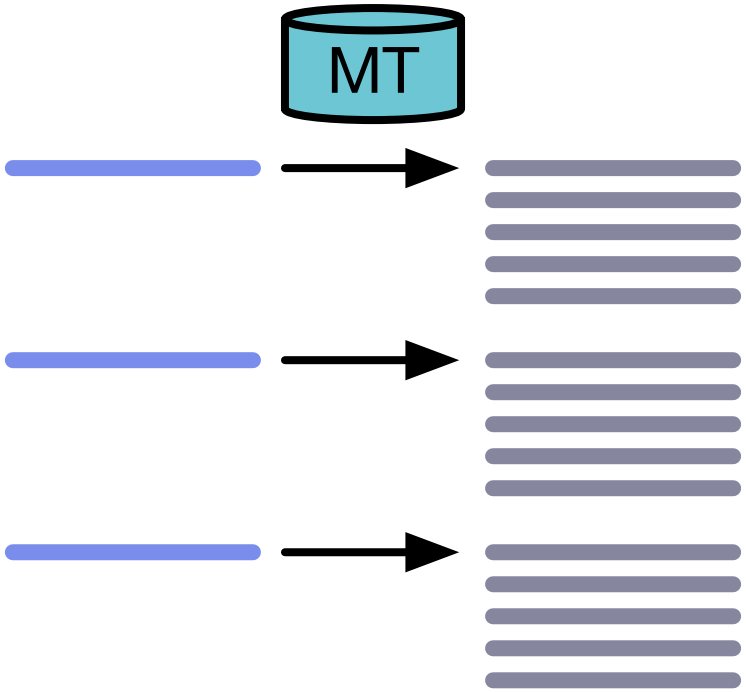
  **Reward** Score for a prediction: could be binary: win/loss
  **Policy** Underlying decision mechanism for steps — in our case this is the MT model or LLM
  **Policy Gradient** Learning method that adjusts the model (policy) parameters directly in the direction that increases expected reward
  **Action** A single step, in MT/LLM: a single token prediction
  **State** Situation after any number of steps, in MT/LLM: the text generated so far
  **Advantage** Measure of importance of a step, e.g., a game-altering move in chess
  **Actor/Critic** Setup with two models: one that decides which actions to take (actor), and one that assesses outcomes (critic)
  **Rollout/Trajectory** Completion of a action sequence until the end (e.g., based on current policy) — this is typically done for MT/LLM

- Reinforcement learning for MT/LLM:
  no intermediate state assessment, no partial rewards

# preference training

# Learning from Human Preferences

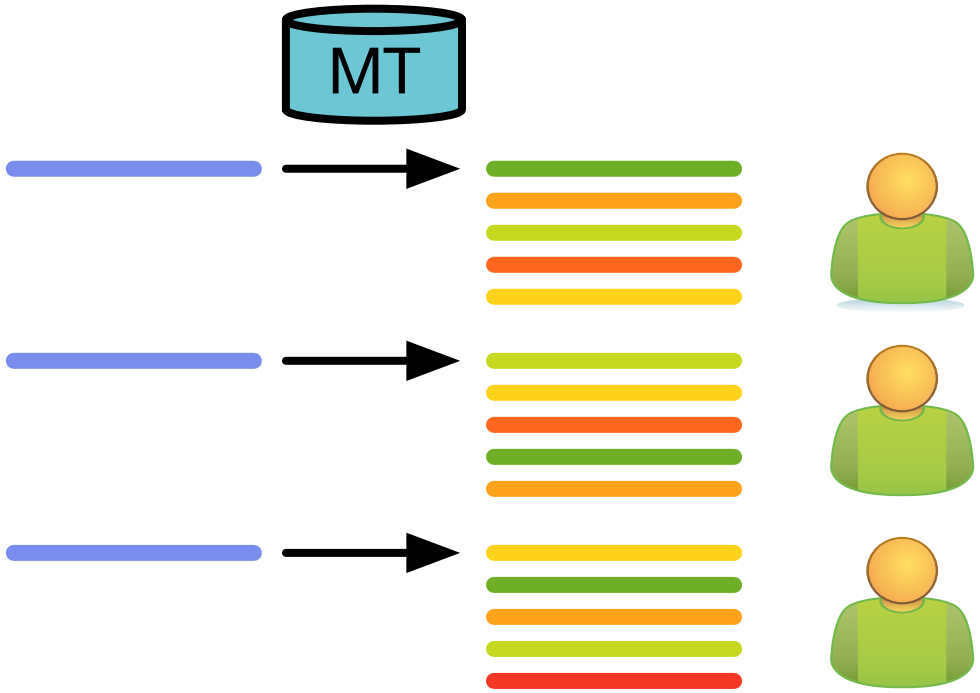





- Generate translations from a source sentence by sampling
  - greedy decoding: always choose word prediction with 80% probability
  - Monte Carlo decoding: choose it 80% of the time

# Learning from Human Preferences

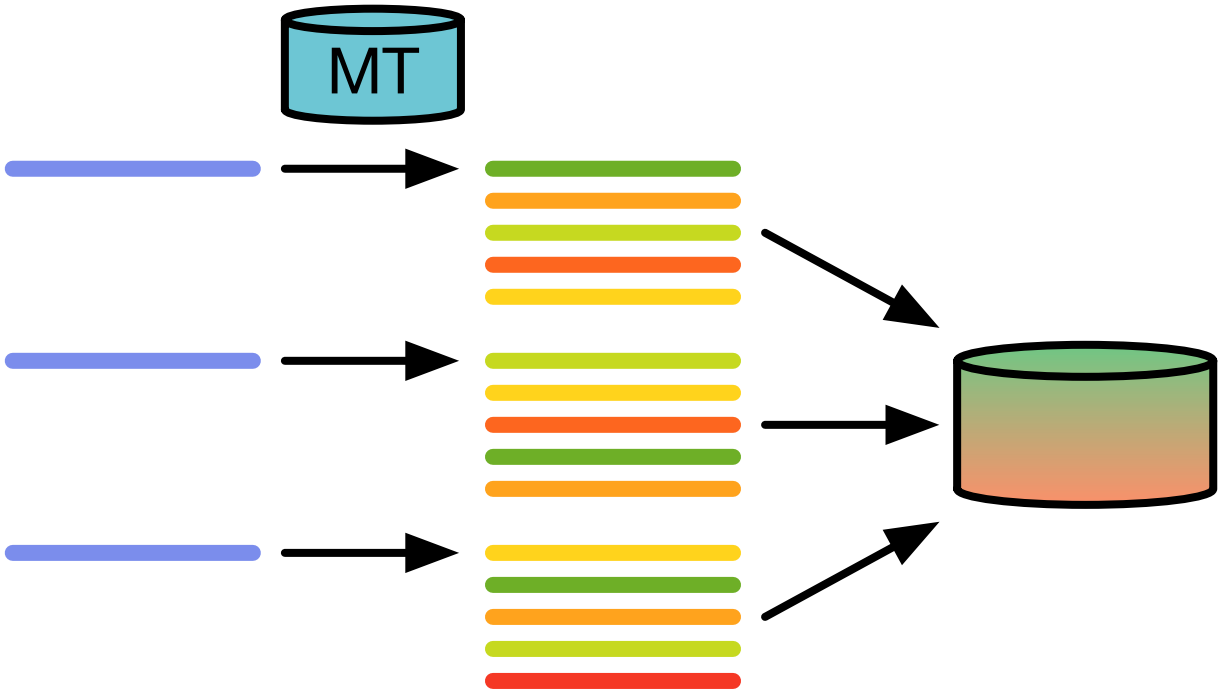





- Human annotators rank the translations
- This is easier to do that authoring translations but still expensive

# Learning from Human Preferences

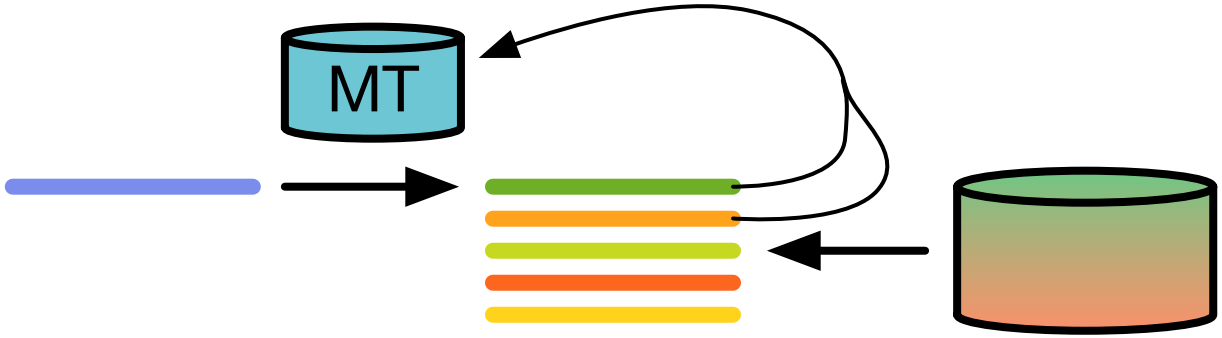





- Train a preference model
- Typically based on sequence representations from language models

# Learning from Human Preferences






- Use the preference model during training original model
  - for an input sentence, generate translations
  - score the translations with the preference model
  - update model to
    - promote higher-scoring translation (winner)
    - demote lower-scoring translation (loser)

# preference data

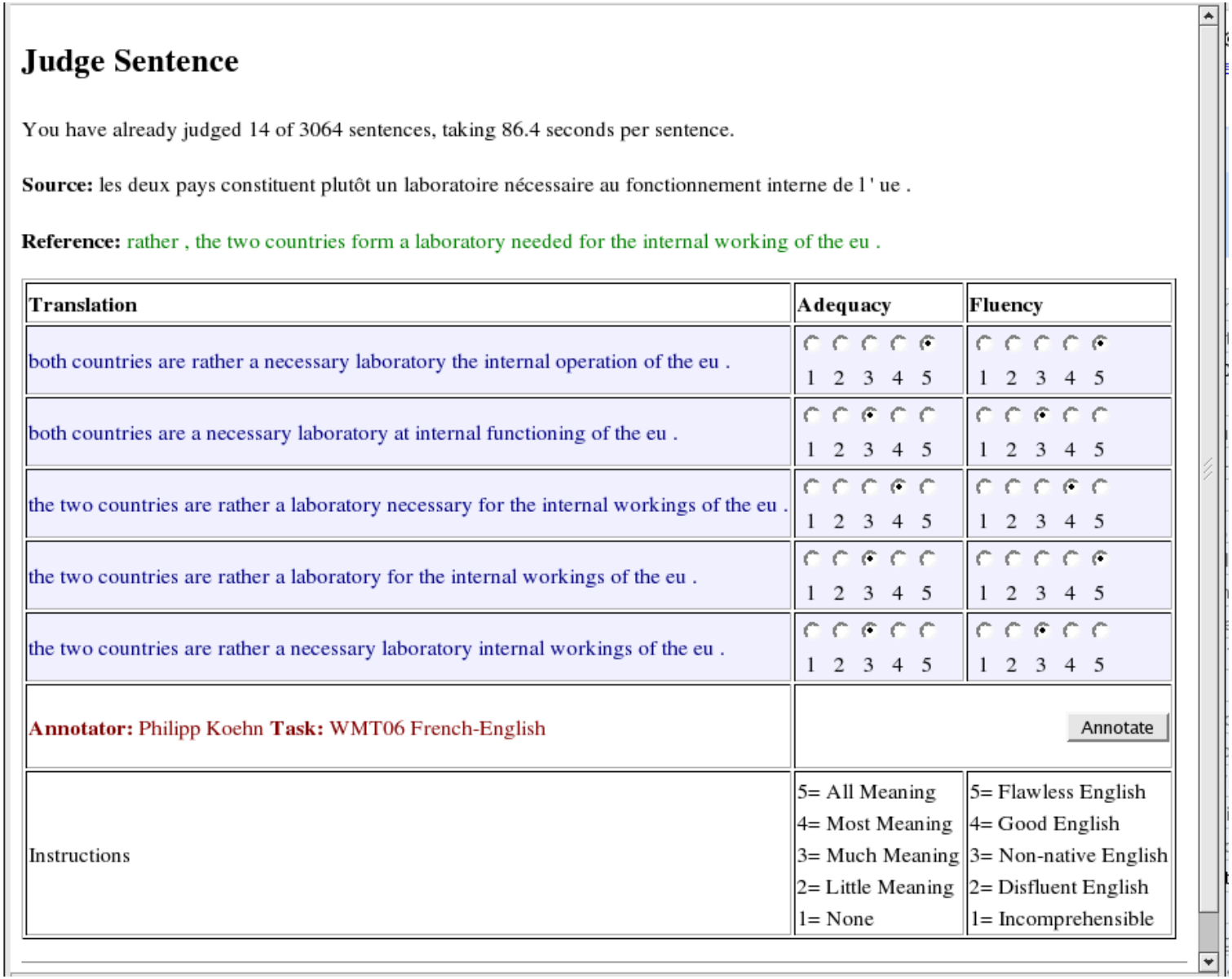
# Preferences for Translation



- What we need is
  - source sentence
  - multiple translations
  - human judgment which translations are better
- This seems to be expensive to obtain...

# 20 Years of WMT Evaluations





## Judge Sentence

You have already judged 14 of 3064 sentences, taking 86.4 seconds per sentence.

**Source:** les deux pays constituent plutôt un laboratoire nécessaire au fonctionnement interne de l ' ue .

**Reference:** rather , the two countries form a laboratory needed for the internal working of the eu .

| Translation | Adequacy | Fluency |
|---|---|---|
| both countries are rather a necessary laboratory the internal operation of the eu . | 1 2 3 4 5 | 1 2 3 4 5 |
| both countries are a necessary laboratory at internal functioning of the eu . | 1 2 3 4 5 | 1 2 3 4 5 |
| the two countries are rather a laboratory necessary for the internal workings of the eu . | 1 2 3 4 5 | 1 2 3 4 5 |
| the two countries are rather a laboratory for the internal workings of the eu . | 1 2 3 4 5 | 1 2 3 4 5 |
| the two countries are rather a necessary laboratory internal workings of the eu . | 1 2 3 4 5 | 1 2 3 4 5 |
| **Annotator:** Philipp Koehn **Task:** WMT06 French-English | Annotate | |
| Instructions | 5= All Meaning<br>4= Most Meaning<br>3= Much Meaning<br>2= Little Meaning<br>1= None | 5= Flawless English<br>4= Good English<br>3= Non-native English<br>2= Disfluent English<br>1= Incomprehensible |

# WMT Evaluation Data



- Long-running WMT evaluation campaign (since 2006)
  - participants submit translations of their system
  - human evaluators score or rank these translations

→ This is the human preference data we need

- Millions of human judgments are available

# reward model

# Reward Model



- Core part of preference training: reward model
- Setup
  - given: source sentence, candidate translation
  - output: score how good the translation is
- This is typically trained on human preference data

# Quality Estimation



- Long standing task in machine translation: quality estimation (or "confidence estimation")

- For instance, for routing translations in production workflow
  - great translation → pass to customer
  - good translation → pass to professional translator
  - bad translation → ignore

- Setup
  - given: source sentence, candidate translation
  - output: score how good the translation is

- This is exactly what we need here

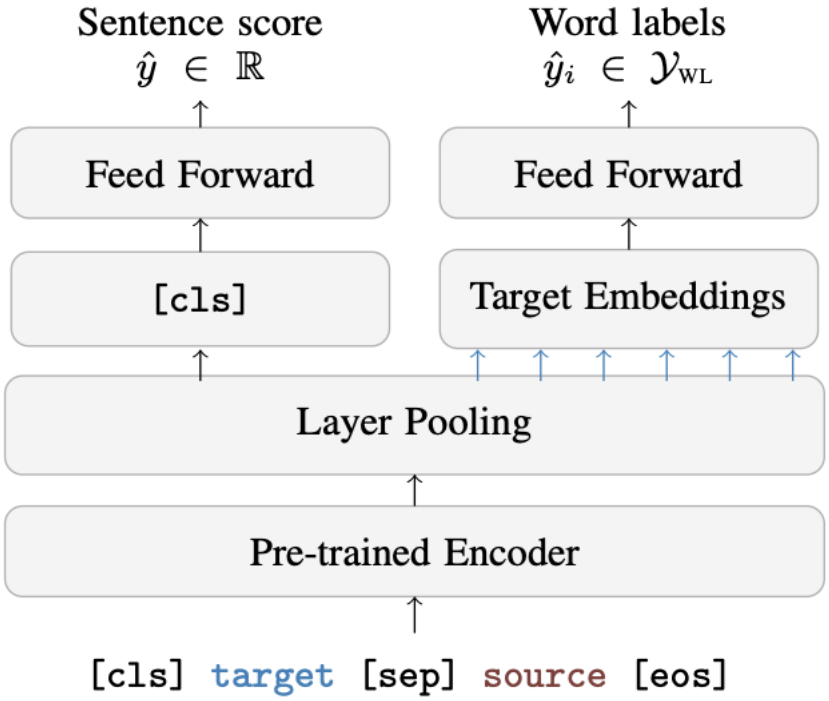
# CometKiwi



- One example for a quality estimation model: CometKiwi [Rei et al., WMT 2023]

- Trained on word-level and sentence-level human evaluation data
  - sentence-level: direct assessment scores
  - word-level: error span annotations

- Foundation model: XLM Roberta with up to 10.7 billion parameters



# loss functions

# Reinforcement Learning from Human Feedback



- This idea was originally introduced as a form of reinforcement learning
- The idea of a reward model stems from reinforcement learning
- Originally proposed method: Proximal Policy Optimization (PPO)
- Recently, simpler methods are more common
- We will take a closer look at Direct Preference Optimization (DPO)

# Setup



- First train a reward model $r(x, y)$ for source sentences $x$ and translations $y$
- Sample two possible translations for an input $x$
- Score them with the reward model
  - higher scoring translation is the winner $y^+$
  - higher scoring translation is the loser $y^-$
- Goal: train a new LLM $\pi_\theta$ from an original model $\pi_{\text{ref}}$

# Preference Model



- We can convert this into a preference model using softmax

$$P(y^+ \succ y^- \mid x) = \frac{\exp(r(x, y^+))}{\exp(r(x, y^+)) + \exp(r(x, y^-))}$$

- This can be converted in the following way:

$$\begin{aligned} P(y^+ \succ y^- \mid x) &= \frac{1}{1 + \frac{\exp(r(x,y^-))}{\exp(r(x,y^+))}} \\ &= \frac{1}{1 + \exp(r(x, y^-) - r(x, y^+))} \\ &= \sigma(-(r(x, y^-) - r(x, y^+))) \\ &= \sigma(r(x, y^+) - r(x, y^-)) \end{aligned}$$

- With $\sigma$ being the well-known sigmoid function

# Adding Regularization



- We do not want the model $\pi_\theta$ to go too far afield
- In other words, keep it close to the original model $\pi_{\text{ref}}$

$\Rightarrow$ We need a measure how much these models differ

- A common choice: KL divergence

# Kullback-Leibler Divergence



- Textbook definition

$$D_{\mathrm{KL}}(P \parallel Q) = \sum_{x \in \mathcal{X}} P(x) \log \frac{P(x)}{Q(x)}$$

- Here we deal with conditional probabilities

$$D_{\mathrm{KL}}(P \parallel Q) = \sum_{y \in \mathcal{Y}} P(y|x) \log \frac{P(y|x)}{Q(y|x)}$$

# Kullback-Leibler Divergence



- In our case, the two distributions are $P = \pi$ and $Q = \pi_{\text{ref}}$, so:

$$D_{\text{KL}}(\pi \parallel \pi_{\text{ref}}) = \sum_{y \in \mathcal{Y}} \pi(y|x) \log \frac{\pi(y|x)}{\pi_{\text{ref}}(y|x)}$$

- Another way to look at this is the expected value of the log-ratio

$$D_{\text{KL}}(\pi \parallel \pi_{\text{ref}}) = \mathbb{E}_{y \sim \pi} \log \frac{\pi(y|x)}{\pi_{\text{ref}}(y|x)}$$

# Direct Preference Optimization (DPO)



- Combine preference and regularization (weight $\beta > 0$) to optimize model $\pi$

$$\pi^* = \arg\max_{\pi} \mathbb{E}_x \Big[ \mathbb{E}_{y \sim \pi(\cdot|x)}[r(x,y)] - \beta \mathrm{KL}\big(\pi(\cdot \mid x) | \pi_{\text{ref}}(\cdot \mid x)\big) \Big]$$

- This results in the following loss function
  (see paper for full derivation)

$$\mathcal{L}_{\text{DPO}}(x, y_w, y_l) = -\log \sigma \left( \beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)} \right)$$

[from Rafailov et al., 2023]

# Contrastive Preference Optimization (CPO)



- Computing probabilities with both
  - the reference model $\pi_{\text{ref}}(y|x)$ and
  - the new model $\pi_\theta(y|x)$

  is expensive

  $\rightarrow$ twice the memory requirements
  $\rightarrow$ twice the number of computations

- Simplification: only score with new model

$$\text{Loss}(x, y_w, y_l) = \log \text{sigmoid}\Big(\beta \log \pi_\theta(y_w|x) - \beta \log \pi_\theta(y_l|x)\Big)$$

[from Xu et al., 2024]

# Group Relative Policy Optimization (GRPO)



- Generate a group of translations (e.g., 64) for a sentence and score each
- Compute average of scores
- Check for each translation how its score compares against the average
  - better than average: promote
  - worse than average: demote

# example:

# direct quality optimization

# Example: Direct Quality Optimization



[Uhlig et al., WMT 2025]

- Core idea
  - use existing quality estimation models as reward model: CometKiwi
  - use DPO as reinforcement learning method
- Big picture
  - sample source sentences
  - generate possible target translations
  - identify winning and losing translations
  - update model with DPO

# Create a Pool of Translations



- Given a source sentence $x$
- Generate possible translations
  - greedy search
  - sampling $K = 40$ additional translations
- Winning translation $y^+$: best according to reward model
- Losing translation $y^-$: randomly sampled (as long it is worse than $y^+$
- Repeat this process to obtain a pool of training examples

# Overall Algorithm



- Initialize model $pi_\theta$ from original model $pi_{\text{ref}}$
- Loop for several rounds:
  - generate a pool of training examples $P$
  - loop for several epochs:
    - run DPO on the pool of training examples

**Algorithm 1:** Direct Quality Optimization

**Parameters:** preference relation $\succ$, number of rounds $n$, epochs per round $m$, epoch size $d$, learning rate $\alpha$, DPO regularization $\beta$, number of sampled translations per source $k$

**Input:** Source language seed dataset $S$, reference-free QE model $r_{QE}$, reference model $\pi_{\text{ref}}$

$\pi_\theta \leftarrow \pi_{\text{ref}}$;
**for** *round* $i = 1, 2, \ldots, n$ **do**
  $X \leftarrow$ **sample** $d$ sentences from $S$;
  $P \leftarrow \varnothing$;
  **foreach** *source* $x \in X$ **do**
    $g \leftarrow \texttt{Greedy}_{\pi_\theta}(x)$;
    $Y \leftarrow$ **sample** $k$ translations of $x$ from $\pi_\theta$;
    $Y_+ \leftarrow Y \cup \{g\}$;
    $y_w \leftarrow \text{argmax}_{y \in Y_+} r_{QE}(x, y)$;
    $Y_l = \{y' \in Y_+ | y_w \succ_x y'\}$;
    **if** $Y_l \neq \varnothing$ **then**
      $y_l \leftarrow$ **sample** $y \in Y_l$;
      $P \leftarrow P \cup \{(x, y_w, y_l)\}$;
  **end**
  **for** *epoch* $j = 1, 2, \ldots, m$ **do**
    $\pi_\theta \leftarrow \texttt{DPO}(\pi_\theta, \pi_{ref}, P, \alpha, \beta)$;
  **end**
**end**

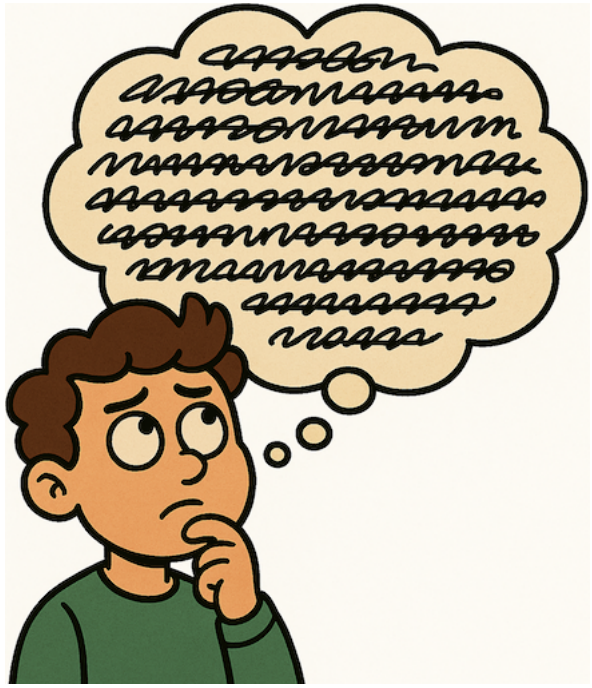
[from Ulig et al., 2025]

# reasoning models

# Thinking



- Complex tasks require more "thinking"
- Typical LLM application: solving math problems
- Typical MT application: translating poetry



- What happens in thinking does not matter so much
- Only the final output matters

# Thinking for Translation



- Consider poetry

  *"Keep, ancient lands, your storied pomp!" cries she*
  *With silent lips. "Give me your tired, your poor,*
  *Your huddled masses yearning to breathe free,*
  *The wretched refuse of your teeming shore.*
  *Send these, the homeless, tempest-tost to me,*
  *I lift my lamp beside the golden door!"*

- Left-to-right generation may struggle with rhyme
  (*she, free, me*, and *poor, shore, door*)

- Other challenges: capturing overall mood, coherent metaphors, ...

- A reasoning model is able to revise a draft

# Reasoning Models



- LLM reasoning models produce output in two stages

- A thinking stage that may be hidden from the user
  *<think> Let's translate Emma Lazarus's final stanza of "The New Colossus" into German while keeping its solemn tone and rhyme scheme.*
  *Step 1: Analyze rhyme and rhythm Original rhyme pattern: ABABCC. A pomp, ... </think>*

- An answer stage with just the desired output
  *<answer> „Behaltet euren Glanz und stolzen Pomp!" ruft sie*
  *Mit stummen Lippen. „Gebt mir eure Müden, Armen, ...*
  *</answer>*

# Rewards



- The reward only considers the final answer
- This may be simple binary answer checking (e.g., for math)
- Should we award good reasoning paths?
  - intuitively, yes
  - this is not currently done
    (except for very basic formality checks)
  - it is also very hard

# questions?