Pun Generating Models

Aligning Your Model with Humor Preferences

目录

1 摘要

现如今，大语言模型虽已具备强大的知识推理及生成能力，但在幽默内容创作上仍面临显著挑战。双关语作为营造幽默氛围的核心元素，需要在语义结构上制造歧义的同时达到幽默效果，这对模型的语言理解与生成能力提出了更高要求。当前主流模型（如ChatGPT 3.5）在双关语生成任务中存在输出同质化、幽默效果不足等问题，亟需针对性优化框架以提升生成质量。本研究选择双关语生成作为研究对象，既因其在对话和文案生成等 NLP 领域应用广泛，也因其是衡量模型语言创造力的重要标尺。

研究工作主要围绕三个维度展开：

首先，在 LlamaFactory 框架下实施直接偏好优化（DPO）方法，通过建模人类偏好信号对模型输出进行定向优化，显著提升生成内容与目标偏好的契合度。

其次，针对幽默生成任务中”双关结构建模”与”幽默效果建模”的目标冲突问题，采用分层课程学习框架 PGCL-DPO。该框架通过训练阶段的任务拆解——先学习双关语的语义映射结构，再学习幽默效果的情感激活机制，实现多目标优化过程的解耦，有效提升了模型对复杂幽默逻辑的建模能力。

最后，为解决分层训练中第二阶段可能出现的灾难性遗忘问题，改进原有 DPO 方法，引入 DPOP（DPO-Positive）正向偏好增强机制。通过设计带有记忆锚点的损失函数修正项，在保留第一阶段双关结构学习成果的同时，强化幽默效果相关参数的更新效率，平衡结构一致性与幽默质量的双重目标，使模型在学习过程中更聚焦于优质双关语的生成模式。

本研究的三个技术环节构建了从基础优化实现到结构设计优化、再到效果增强的完整研究框架，最终实现双关语生成的结构一致性与幽默质量的双重提升，不仅验证了偏好优化在幽默生成场景的有效性，也为多目标语言生成任务提供了一套有效的可迁移的方法论框架。

2 预备工作

2.1 双关语类型

双关语作为一种重要的幽默表达形式，主要分为两种经典类型，这为我们后续的研究奠定了理论基础。同形异义双关巧妙地利用单个词汇的多重语义来制造幽默效果。例如，短语 “all right” 既可以表示 “没问题、令人满意”，但也可以从字面意思理解为 “全部是右”，与 “left（左）” 形成语义上的对比，从而产生有趣的歧义。

同音异义双关则依赖发音相同或相似但含义截然不同的词汇来形成反差。像“week”（星期）和 “weak”（虚弱），虽然读音相近，但意思完全不同，通过这种谐音关联营造出幽默的氛围。清晰区分这两种双关结构，能为后续模型在语义结构控制和幽默效果优化方面提供关键的理论支持。

2.2 数据集准备与处理

在本研究中，使用了 ChinesePun 数据集，如表 1 所示。由于原始数据并未完全公开，我们选择了一半的数据，其中同形异义和同音异义双关语各超过 500 句。我们对原始数据进行了清理和格式化，使其符合“提示词 + 优选 / 弃选”的格式，以适应直接偏好优化（DPO）等对比训练流程。此外，我们标注了双关语的结构类型，以便评估结构生成的成功率。

类型	句子数量	词数	双关词数量		
			最少	最多	平均
同音异义	1049	96212	1	4	1.11
同形异义	1057	91100	1	3	1.05

表 1: ChinesePun 数据集统计信息

该 ChinesePun 数据集中，前 528 句为 **homographic（同形异义）**，后 524 句为 **homophonic（同音异义）**。其中每句话可以包含一个或多个双关语。以下展示各类型的示例及结构说明：

homographic 示例：

> “爸爸，是男的跑得快，还是女的跑得快?”
> “当然是男的快。”
> “那为什么总是男孩子追女孩子呢。”
> （双关词“追”：包含“求爱”和“追赶”两种意思）

homophonic 示例：

> 典韦拼尽最后一口气，杀开一条血路，倒在血泊中，冲着身后的曹操喊：“主公，逃!”
> 曹操忙不迭从怀中摸出一个桃，扔了过去。
> （双关词“逃”和“桃”：同音异义）

整理后的语料示例：

- 样本类型：homographic
- 关键双关词：翻天覆地

Listing 1: DPO 格式的示例

```
{
  "conversations": [
    {
      "from": "human",
      "value": "根据上述词义，请为我编写一个幽默的句子，并使用“翻天覆地”这个词在一个句子中一次或多次。这个句子应该能够引发笑声或娱乐，并展示“翻天覆地”这个词语的不同含义在幽默语境下的创意用法。你可以尝试在句子中加入意想不到的情节或转折，以增加幽默效果。请确保句子仍然通顺、自然，并具有良好的语法和逻辑结构。"
    }
  ],
  "chosen": {
    "from": "human",
    "value": "当霍开始建房时，呷呷木匠错误地将地板木料用于天花板，同时将天花板木料放到了地板上。木匠问其缘由，霍解释道：我即将结婚，而大家都知道，一旦结婚，所有的事情都会“翻天覆地”；所以我提前做了准备。"
  },
  "rejected": {
    "from": "gpt",
    "value": "在最近的公司团建中，为了增加趣味性，老板决定彻底改造会议室，把所有椅子换成滑板，然后宣布：“从现在起，大家都要滑着参与会议讨论！”"
  }
}
```

3 模型介绍

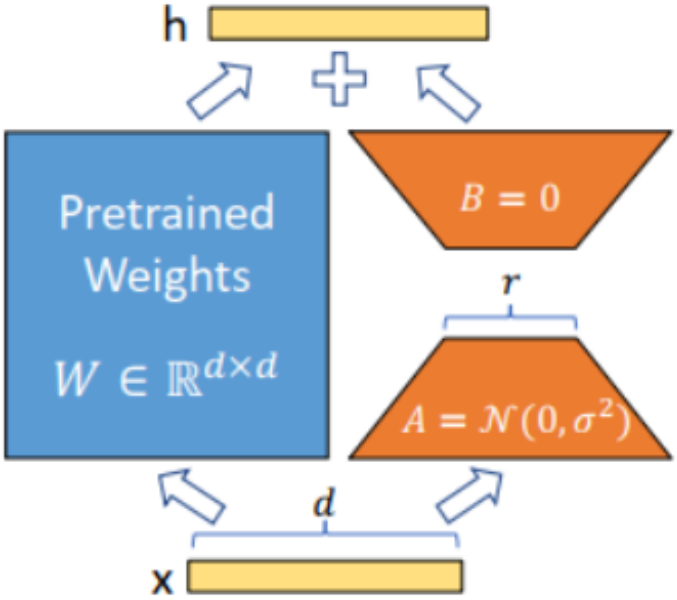
3.1 LORA 微调策略

3.1.1 基本介绍

LORA[1]：种低资源微调大模型方法练参数仅为整体参数的万分之一、GPU 显存使用量减少 2/3 且不会引入额外的推理耗时。

3.1.2 实现方式

前提——LLMs 拥有很好的 few-shot 能力的原因：[2] 的研究表明，预训练模型拥有极小的内在维度 (instrisic dimension)，即存在一个极低维度的参数，微调它和在全参数空间中微调能起到相同的效果。同时 [2] 发现在预训练后，越大的模型有越小的内在维度。



受 instrisic dimension[2] 工作的启发，LORA 作者认为参数更新过程中也存在一个‘内在秩’。对于预训练权重矩阵 $W_0 \in R^{d\times k}$，可以用一个低秩分解来表示参数更新 ΔW，即

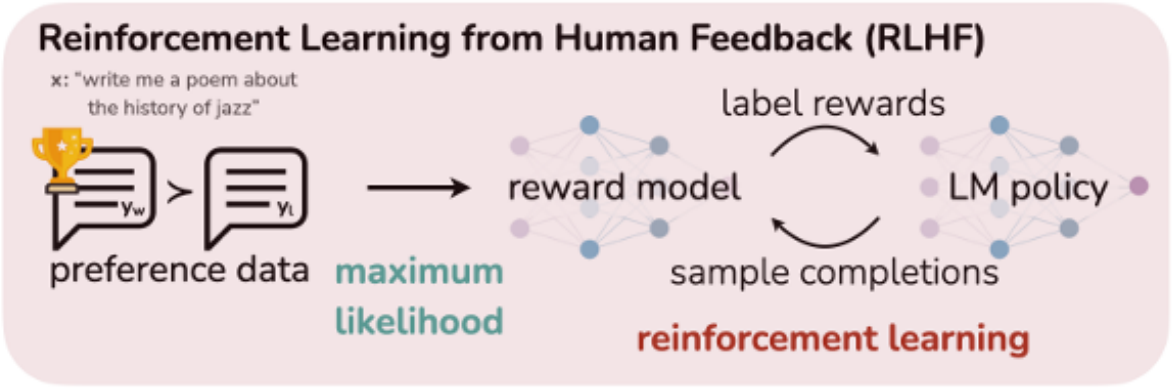
$$W_0 + \Delta W = W_0 + BA \quad B \in R^{d\times r}, A \in R^{r\times k} \text{ and } r \ll \min(d, k)$$

训练过程中冻结参数 W_0，仅训练 A 和 B 中的参数。对于前向传播过程 $h = W_0 x$ 变为

$$h = W_0 x + \Delta W x = W_0 x + BAx$$

在实际操作中，将可微调参数分配到多种类型权重矩阵中效果往往优于使用更大的秩单独微调某种类型的权重矩阵。

3.2 RLHF



3.2.1 基本介绍

RLHF(基于人类反馈的强化学习)[3]：对 LLMs 进行监督微调的方法里面比较成功的是基于人类反馈的强化学习（RLHF），RLHF 方法将奖励模型拟合到人类偏好数据集，并利用强化学习来优化语言模型策略，以生成被分配高奖励的回应，同时避免过度偏离原始模型。

(其思想直白一点可理解为是：训练了两个模型——决策的模型用于生成接下来的文本，奖励模型模拟人类给生成的文本打分，并对决策模型进行反馈，使得其下次生成偏向分数更高的文本，两个模型互相帮助对方训练)

3.2.2 基本流程

Ziegler 等人提出的 RLHF 管道 [4]（以及后续工作 [5],[6]）通常包括三个阶段:

1. 监督微调（SFT）：RLHF 通常从对下游任务（如对话、摘要等）的高质量数据进行监督学习的微调开始，得到模型 π_{SFT}

2. 奖励建模阶段：在第二阶段，SFT 模型会被输入一组提示 x，生成一对回答 $(y_1, y_2) \sim \pi_{SFT}(y|x)$, 这些对回答会被人类标注者（或者强化的 LLM）进行标注，标注者会表达其中一个回答的偏好，记为 $y_w \succ y_l|x$, 这里 y_w, y_l 分别表示偏好和不偏好的回答。然后在某个 size 的 SFT 模型上加上标量输出层构建 reward model 进行训练。这样，针对每个 (x, y_i) 我们可以得到一个偏好“分数” $r(x, y_i)$。为了使得偏好回答的分数更高，引入了 Bradley-Terry(BT) 模型进行 loss 设计，在 BT 模型中，两个选项 α_i 和 α_j 的相对偏好概率可定义为:

$$p(\alpha_i \succ \alpha_j) = \frac{e^{\alpha_i}}{e^{\alpha_i} + e^{\alpha^j}} = \frac{1}{1 + e^{-(\alpha_i - \alpha_j)}}$$

这与 sigmoid 函数形式一致。因此，基于收集到的偏好对数据 $D = \{x^{(i)}, y_w^{(i)}, y_l^{(i)}\}_{i=1}^N$ 我们可以参数化奖励模型 $r_\Phi(x, y)$ 并通过最大似然来估计参数值，可将该问题建模为二

元分类问题，其负对数似然损失为：

$$\mathcal{L}_R(r_\phi, \mathcal{D}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}} \left[\log \sigma(x, y_w) - r_\phi(x, y_l)\right].$$

因此，最小化损失等价于最大化 y_w 相对于 y_l 的偏好分数分布之和

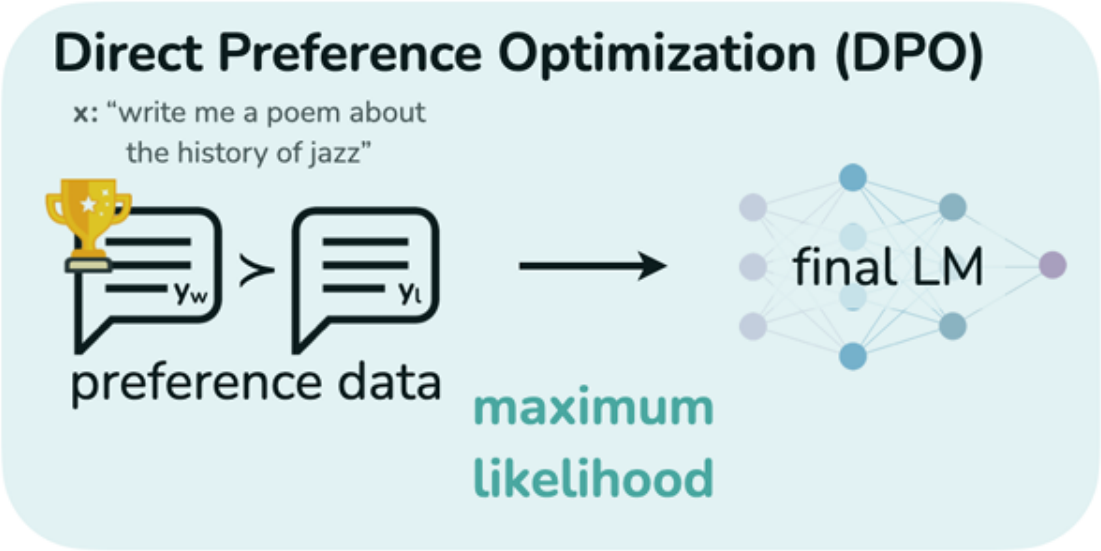
3. 强化学习微调阶段：在强化学习阶段，用的最广泛的是 PPO[7] 算法，最大化生成的答案能够获得的 reward 之和，同时为了确保和 SFT 表现相差不太远，添加了 KL 散度作为约束，由 β 控制约束的程度：

$$\max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(y|x)}[r_\phi(x, y)] - \beta\mathbb{D}_{KL}\left[\pi_\theta(y|x)\|\pi_{\text{ref}}(y|x)\right].$$

3.3 DPO

3.3.1 基本介绍

DPO(直接偏好优化) [8]：将两个模型“合二为一”，通过一种巧妙的设计思路将强化学习的最大化奖励和转化为损失函数，使得可以直接基于监督学习训练一个模型的方法。降低了模型和训练的复杂性。



我们定义 DPO 算法的 loss function 如下：

$$\mathcal{L}_{\text{DPO}}(\pi_\theta, \pi_{\text{ref}}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta\log\frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\right)\right]. \quad (1)$$

我们在 Appendix A中补充了 DPO Loss 的推导过程 [8]。

由于原始的 DPO 算法在双关语笑话生成任务中效率较低，因为我们需在较小规模数据集上同步实现双关结构偏好与幽默偏好的双重对齐。为此，文章提出了一种多阶段课程偏好学习框架，并改进了 DPO 算法以解决多目标对齐难题。

3.3.2 第一阶段：结构偏好对齐

这里我们定义损失函数如下：

$$
\begin{aligned}
r^{+}(\theta) &= \beta(\log \pi_\theta(y^{+}|x) - \log \pi_{\text{ref}}(y^{+}|x)) \\
r_s^{-}(\theta) &= \beta(\log \pi_\theta(y_s^{-}|x) - \log \pi_{\text{ref}}(y_s^{-}|x)) \\
\mathcal{L}_{\text{structure-dpo}}(\theta) &= -\log \sigma(r^{+}(\theta) - r_s^{-}(\theta))
\end{aligned} \tag{2}
$$

如 Equation 2所示：其中，y^{+} 为事先准备好的偏好样本 (chosen)，y^{-} 为带训练模型经微调前生成的不满足双关语结构的样本 (rejected)。

3.3.3 第二阶段：幽默偏好对齐

首先我们展示改进前的第二阶段 DPO Loss:

$$
\begin{aligned}
r_\phi^{+}(\theta) &= \beta(\log \pi_\theta(y^{+}|x) - \log \pi_\phi(y^{+}|x)) \\
r_j^{-}(\theta) &= \beta(\log \pi_\theta(y_h^{-}|x) - \log \pi_\phi(y_h^{-}|x)) \\
\mathcal{L}_{\text{humor-dpo}}(\theta) &= -\log \sigma(r_\phi^{+}(\theta) - r_h^{-}(\theta)).
\end{aligned} \tag{3}
$$

然而，由 Equation 3定义的 DPO Loss 存在灾难性遗忘问题：第一阶段训练好的双关语结构偏好可能被遗忘，为此，为输入增加第三元并改进 DPO 损失，使其既适配这种多目标对齐任务，又可缓解灾难性遗忘效应:

$$
\begin{aligned}
r_{h^*}^{-}(\theta) &= \beta(\log \pi_\theta(y_{h^*}^{-}|x) - \log \pi_\phi(y_{h^*}^{-}|x)) \\
r^{*}(\theta) &= \beta(\log \pi_\theta(y^{*}|x) - \log \pi_\phi(y^{*}|x)) \\
\mathcal{L}_{\text{I-humor-dpo}}(\theta) &= -\log \sigma(r_\phi^{+}(\theta) - r^{*}(\theta)) - \log \sigma(r^{*}(\theta) - r_{h^*}^{-}(\theta)).
\end{aligned}
$$

其中 r_ϕ^{+} 与上述的未改进前的损失中定义的一样，y^{*} 为初始模型经第一阶段微调后在 prompt 下生成的样本中满足双关语结构但幽默度欠缺的样本，$y_{h^*}^{-}$ 为生成的样本中不满足双关语结构的样本。

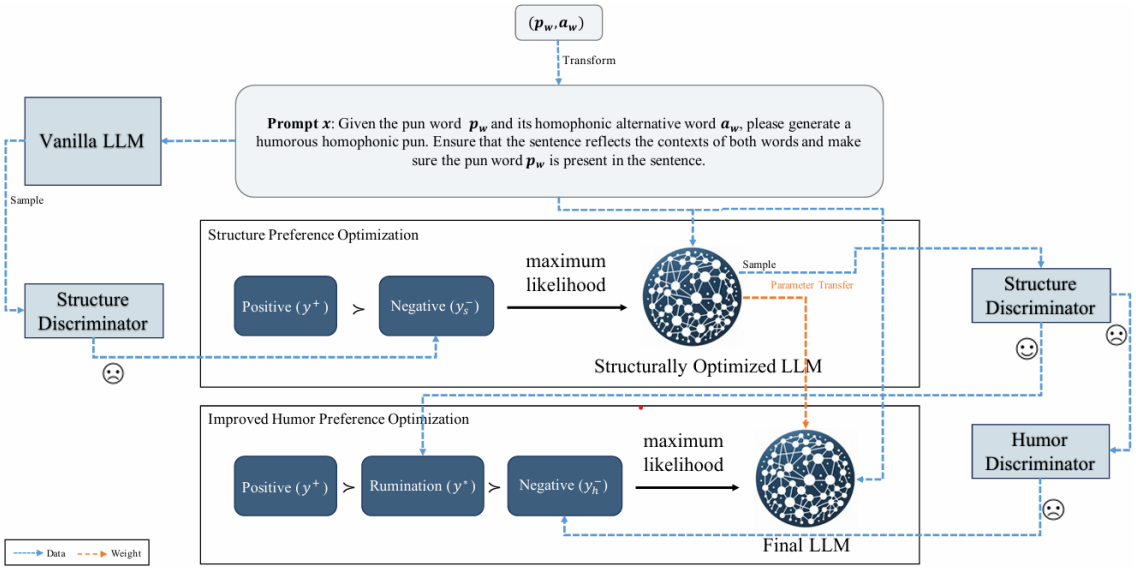


图 1: PCGL-DPO 流程图

3.4 DPOP

在进一步深入研究后，我们发现传统的直接偏好优化（DPO）方法在我们使用的双关语生成任务中存在一些缺陷。尽管 DPO 能够有效地对模型进行偏好对齐，但在某些情况下，模型可能会过度关注于增大正例（chosen）和负例（rejected）之间的相对概率差异，而忽略了正例本身的概率提升。这种现象在编辑距离较低的数据集上尤为明显，导致模型对偏好数据的建模概率降低。为了避免这个现象，来自 Abacus.AI 的研究者提出了 DPO-Positive(DPOP)[9]。

DPOP 在 DPO loss 中加入了惩罚项:

$$\mathcal{L}_{\text{DPOP}}(\theta) = -\mathbb{E}_{(x,y_w,y_l)}\left[\log\sigma\left(\beta\left(\log\frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \lambda\cdot\max\left(0, \log\frac{\pi_{\text{ref}}(y_w|x)}{\pi_\theta(y_w|x)}\right)\right)\right)\right] \quad (4)$$

其中，$\lambda > 0$ 为可调节的参数。

本质上该损失是加了一个 Penalty 去限制 Chosen 的 likelihood 下降。这使得模型不再仅关注 chosen 和 rejected 的概率间隔，(导致其过程不是增大 chosen 概率减小 rejected 概率，而是减小 chosen 概率的同时降低更多的 rejected 概率)。还必须确保正例的概率相对 reference 模型的概率更高。

4 实验

4.1 实验环境

本次实验选择了 Qwen2.5-3B 作为基础模型，对其进行微调。全部的实验均在单张 NVIDIA A800 80GB GPU 上进行，使用 LLaMa-Factory[10] 框架进行模型微调和评估。实验过程中，使用了 LoRA[1] 技术来降低显存占用，并提高训练效率。

LLaMa-Factory 是一个专为简化与加速大语言模型微调及部署而设计的开源平台，支持 LLaMA、Qwen 系列等主流开源大模型，并深度兼容 LoRA、QLoRA 等高效微调技术。凭借对显存占用的优化，即使在硬件资源有限的条件下，也能实现模型的定制化训练，能够大幅提升开发效率与灵活性。

支持方法：

方法	全参数训练	部分参数训练	LoRA	QLoRA
预训练	✓	✓	✓	✓
指令监督微调	✓	✓	✓	✓
奖励模型训练	✓	✓	✓	✓
PPO 训练	✓	✓	✓	✓
DPO 训练	✓	✓	✓	✓
KTO 训练	✓	✓	✓	✓
ORPO 训练	✓	✓	✓	✓
SimPO 训练	✓	✓	✓	✓

4.2 实验过程

4.2.1 安装及配置环境

安装 LLaMA-Factory

```
cd PGM
conda create -n pgm python=3.10
pip install -e ".[torch,metrics]"
```

使用 llamafactory-cli version 来快速校验安装是否成功

同时，为了满足模型批量推理的需求，我们还需要安装 vllm：

```
pip install vllm==0.5.4
```

Checkpoint 下载

模型预训练节点下载地址为https://huggingface.co/Qwen/Qwen2.5-3B/tree/main

4.2.2 评估指标

一. 自动评估体系

1. **结构成功率**：模型自动判断双关语是否在句子中使用
2. **多样性**：([11]，[12])
   - **Dist-1（独特单词比例）**：语料/句子中独特单词的比例
   - **Dist-2（独特双词组比例）**：语料/句子中独特双词组的比例
3. **平均句长**

二. 人工评估 论文里提到：为评估幽默生成能力，他们通过人工 A/B 测试将他们的模型与 ChatGPT 进行对比：将不同模型生成的句子两两配对，要求标注者基于幽默效果选择更优的句子，若差异不显著则允许判定为平局。针对双关生成成功率（Pun Succ.）的评估，参照 AmbiPun[11] 的方法，邀请评估者将每个句子归类为" 双关" 或" 非双关" 两类。他们采用配对 kappa 系数衡量标注者间一致性（IAA），所有评估者在幽默能力与双关成功率两项指标上的平均一致性系数分别为 0.47 和 0.58，表明标注者之间达到中等一致性水平。

为了给样本的幽默性打分，我们还尝试了寻找了一些方式，比如`https://github.com/sandro272/ChineseHumorComputation` 但是一是鉴于再训练打分模型开销过大，而且该任务是在有标签数据集上进行训练和测试的分类任务，其测试正确率也仅有 0.7 左右（见表 2），所以我们最终还是决定放弃这个方案。

模型	训练集	测试集	训练集准确率	测试集准确率	备注	模型名称
幽默等级	6436	1610	0.8891	0.6137	5 分类	BiLSTM
幽默类型	5938	1460	0.9357	0.7096	3 分类	BiLSTM
幽默类型	5938	1460	0.9357	0.7356	3 分类	CNN
幽默类型	5938	1460	0.9357	0.7103	3 分类	CNN-ATT
幽默类别	3515	879	0.9166	0.6809	2 分类	BiLSTM
隐喻情绪	2904	726	0.8134	0.5399	7 分类	BiLSTM

表 2: 各中文幽默计算任务的训练与测试表现

原论文表示他们的标注工作由三位语言相关领域的研究生组成的团队完成。在正式标注前，团队成员接受了任务培训，内容包括：定义双关词、界定双关句、定位双关句中的双关词。每个句子均由全体成员标注，存在分歧时通过集体讨论，最终结果按多数表决原则确定，该流程旨在保证标注质量和一致性。工作人员薪酬标准为每小时 60 元。

最终采取随机抽样 100 句句子，通过人工判断双关是否被正确使用的比率，确保双关运用的准确性与合理性。

4.2.3 提示词设计

由于双关语的生成需要模型理解和运用同形异义和同音异义的特性，且需要在幽默语境下进行创意表达，因此我们需要设计合适的提示词来引导模型生成符合要求的双关语句子。提示词的设计需要考虑到双关语的特性，以及模型生成幽默内容的能力。以下是我们经过多次迭代和测试后设计的提示词：

同形异意：

根据上述词义，请为我编写一个幽默的句子，并使用“p_w”这个词在一个句子中一次或多次。这个句子应该能够引发笑声或娱乐，并展示“p_w”这个词语的不同含义在幽默语境下的创意用法。你可以尝试在句子中加入意想不到的情节或转折，以增加幽默效果。请确保句子仍然通顺、自然，并具有良好的语法和逻辑结构。

同音异意：

请为我编写一个幽默的句子，其中包含词语“p_w”和“a_w”。这个句子应该能够引发笑声或娱乐，并展示“p_w”和“a_w”这两个同音词语在幽默语境下的创意用法。你可以尝试在句子中加入意想不到的情节或转折，以增加幽默效果。请确保句子仍然通顺、自然，并具有良好的语法和逻辑结构。

4.2.4 前置参数和数据预处理

前置参数

在开始实验之前，我们需要设置一些前置参数，包括模型名称、模板、数据集类型等。以下是一个示例配置：

```
type=homophonic
model=Qwen2.5-3B
template=qwen
```

数据预处理

在正式训练之前，我们对数据集进行了清洗和格式化处理。我们将双关语数据集拆分为同形异义双关语和同音异义双关语两部分，分别存储在 homographic.json 和 homophonic.json 两个文件中，并将其输出到 data/目录下，以便后续针对模型在不同类型双关语生成的能力进行优化和评估。(其余详细操作请参考代码文件中的 README)

4.2.5 超参数设置

我们在实验中将 temperature 设置为 0.95，top_p 设置为 0.95，top_k 设置为 5，可以使得大语言模型生成的语句更为丰富，而不是较为单一的相同语句。同时，也为 LoRA 训练设置了如下的超参数，

```
# LoRA
```

```
per_device_train_batch_size=4
optim=adamw_torch
learning_rate=5e-05
pref_beta=0.5
dpop_lambda=0.5
```

其他未提及的超参数都采用 Llama- Factory 中的默认值。具体如下所示

第一阶段，结构偏好对齐

```
llamafactory-cli train \
    --stage dpo \
    --do_train True \
    --model_name_or_path ./LLM/${model} \
    --preprocessing_num_workers 16 \
    --finetuning_type lora \
    --template qwen \
    --flash_attn auto \
    --dataset_dir data \
    --dataset dpo_${type}_cn_1 \
    --cutoff_len 2048 \
    --learning_rate 5e-05 \
    --num_train_epochs 3.0 \
    --max_samples 100000 \
    --per_device_train_batch_size 4 \
    --gradient_accumulation_steps 8 \
    --lr_scheduler_type cosine \
    --max_grad_norm 1.0 \
    --logging_steps 5 \
    --save_steps 100 \
    --warmup_steps 0 \
    --packing False \
    --report_to none \
    --output_dir saves/${model}/${type}/stage2 \
    --bf16 True \
    --plot_loss True \
    --overwrite_output_dir True \
    --trust_remote_code True \
    --ddp_timeout 180000000 \
    --include_num_input_tokens_seen True \
    --optim adamw_torch \
    --lora_rank 8 \
    --lora_alpha 16 \
    --lora_dropout 0 \
    --lora_target all \
    --pref_beta 0.5 \
    --pref_ftx 0 \
```

```
    --pref_loss dpop \
    --dpop_lambda 0.5 \
    --top_p 0.95 \
    --top_k 5 \
    --temperature 0.95
```

第二阶段幽默偏好对齐

```
llamafactory-cli train \
    --stage dpo \
    --do_train True \
    --model_name_or_path ./LLM/Qwen2.5-3B-Instruct \
    --adapter_name_or_path ./saves/Qwen2.5-3B-Instruct/${type}/stage1 \
    --model_name_or_path ./LLM/${model} \
    --preprocessing_num_workers 16 \
    --finetuning_type lora \
    --template qwen \
    --flash_attn auto \
    --dataset_dir data \
    --dataset dpo_${type}_cn_2 \
    --cutoff_len 2048 \
    --learning_rate 5e-05 \
    --num_train_epochs 3.0 \
    --max_samples 100000 \
    --per_device_train_batch_size 4 \
    --gradient_accumulation_steps 8 \
    --lr_scheduler_type cosine \
    --max_grad_norm 1.0 \
    --logging_steps 5 \
    --save_steps 100 \
    --warmup_steps 0 \
    --packing False \
    --report_to none \
    --output_dir saves/${model}/${type}/stage2 \
    --bf16 True \
    --plot_loss True \
    --overwrite_output_dir True \
    --trust_remote_code True \
    --ddp_timeout 180000000 \
    --include_num_input_tokens_seen True \
    --optim adamw_torch \
    --lora_rank 8 \
    --lora_alpha 16 \
    --lora_dropout 0 \
    --lora_target all \
    --pref_beta 0.5 \
```

```
    --pref_ftx 0 \
    --pref_loss humor \
    --humor_gamma 0.5 \
    --top_p 0.95 \
    --top_k 5 \
    --temperature 0.95
```

4.2.6 重要代码复现

这里我们展示了此次实验中部分重要代码复现，更详细内容请参考代码文件中的 README 文件。

我们在 src/llamafactory/train/dpo/trainer.py 中添加了 humor_loss 和 dpop_loss 方法，用于计算幽默偏好和双关结构偏好的损失函数。

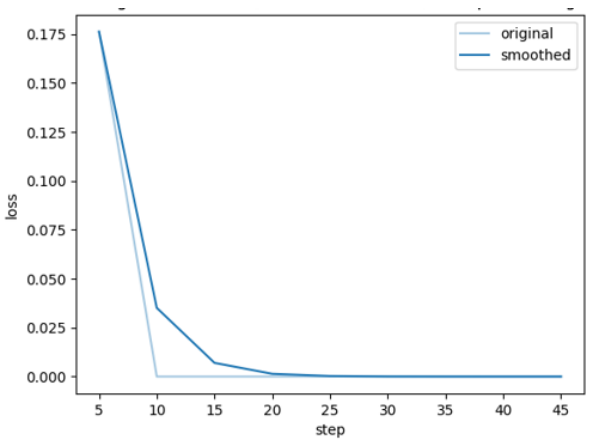
```
def dpop_loss(self, policy_chosen_logps: "torch.Tensor",
    policy_rejected_logps: "torch.Tensor",
    reference_chosen_logps: "torch.Tensor",
    reference_rejected_logps: "torch.Tensor") -> "torch.Tensor":

    pi_logratios = policy_chosen_logps - policy_rejected_logps
    ref_logratios = reference_chosen_logps - reference_rejected_logps

    logits = pi_logratios - ref_logratios - self.dpop_lambda * torch.
        clamp_min(-pi_logratios, 0.0)
    dpop_loss = -F.logsigmoid(self.beta * logits)
    return dpop_loss
```

```
def humor_loss(self, policy_chosen_logps: "torch.Tensor",
    policy_rejected_logps: "torch.Tensor",
    reference_chosen_logps: "torch.Tensor",
    reference_rejected_logps: "torch.Tensor",
    policy_gold_logps: "torch.Tensor",
    reference_gold_logps: "torch.Tensor") -> "torch.Tensor":

    pi_logratios = policy_chosen_logps - policy_rejected_logps
    ref_logratios = reference_chosen_logps - reference_rejected_logps

    logits = pi_logratios - ref_logratios

    pi_gold_logratios = policy_gold_logps - policy_chosen_logps
    ref_gold_logratios = reference_gold_logps - reference_chosen_logps
    gold_logits = pi_gold_logratios - ref_gold_logratios
    humor_loss = -F.logsigmoid(self.beta * logits) + self.humor_gamma * (-F
        .logsigmoid(self.beta * gold_logits))
```

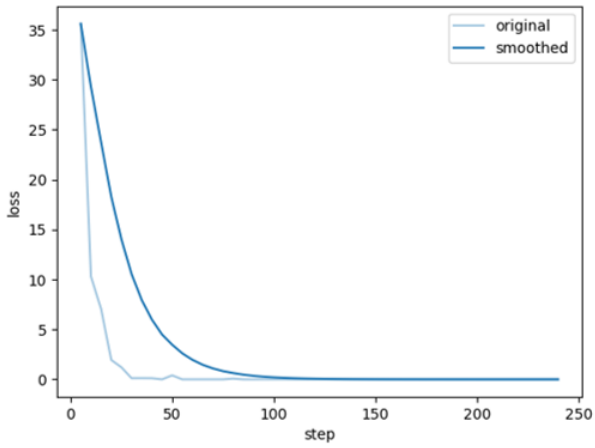
```
    return humor_loss
```

4.3 实验结果

损失曲线如下图所示：



stage 1



stage 2

表格3是最终的结果，可见经过我们的逐步调优和改进，模型的各指标呈现积极变化，在提高结构成功率的同时，双关成功率从原始模型的 43% 经过两阶段的 dpo 微调提升到了 51%。

Dataset	Model	Avg-Length	Corpus-Div %		Sentence-Div %		Structure Succ. %	Pun Succ. %
			Dist-1	Dist-2	Dist-1	Dist-2		
ChinesePun	ChatGPT3.5	53.94	5.51	41.88	77.87	95.17	93.07	28.00
	Qwen2.5-3B	132.76	5.54	31.21	67.91	93.36	90.43	43.00
	Qwen2.5-3B-Instruct	141.58	5.67	34.35	66.83	91.68	94.87	44.00
	$\text{Qwen2.5-3B}_{\text{struc-dpo}}$	139.38	5.59	34.31	67.14	91.33	<u>93.89</u>	40.00
	$\text{Qwen2.5-3B}_{\text{improved-humor-dpo}}$	<u>167.91</u>	5.72	34.44	67.56	91.69	92.62	51.00
	$\text{Qwen2.5-3B}_{\text{dpop}}$	163.47	<u>5.77</u>	34.72	<u>68.03</u>	<u>91.90</u>	92.84	<u>55.00</u>
	Human	88.94	1.85	30.19	65.92	89.13	87.01	—

表 3: 对比不同模型在 ChinesePun 数据集下的表现

同时，我们也选取了 Qwen2.5-3B-Instruct，即官方为遵循指令或完成特定任务而设计和优化的模型，作为对比模型。可以看出尽管 ChinesePun 数据集体量较小，但经过第一阶段和第二阶段的微调，模型在双关语生成任务上表现已经可以媲美甚至超过了 Qwen2.5-3B-Instruct 模型。

最后，我们还对比了 DPO 和 DPOP 的效果，结果表明在使用 DPOP 算法后，模型的双关语生成能力有了进一步提升，双关成功率从 51% 提升到了 55%，同时其他指标也有小幅度提升。

5 消融实验

为验证本文方法各组件的有效性及关键技术的作用机制，我们从模块必要性、方法改进效果和超参数敏感性三个维度展开消融验证，并引入对比方法进一步分析模型优化方向。

5.1 核心模块消融验证

本实验旨在验证两阶段优化框架中幽默偏好模块的不可或缺性，以及改进后 DPO 方法的有效性。实验设置如下：

5.1.1 实验设计

- **w/o Improved Humor DPO**：替换第二阶段的改进型 DPO 为传统 DPO 算法，保持其他训练流程不变；
- **w/o Humor**：移除第二阶段的幽默偏好优化，仅保留第一阶段的结构偏好训练；
- **Baseline**：完整方法（$\text{Qwen2.5-3B}_{\text{improved-humor-dpo}}$）作为对照。

5.1.2 结果分析

如表 4 所示，仅保留结构优化阶段（w/o Humor）时，结构成功率（93.89%）达到最高，但双关成功率（Pun Succ.）显著下降至 40.00%，表明模型过度关注句式结构而忽略语义幽默性，验证了幽默偏好优化的必要性。当替换为传统 DPO 时（w/o Improved Humor DPO），结构成功率（90.14%）和双关成功率（47.00%）均低于完整方法，说明改进型 DPO 通过动态权重调整机制，更有效地平衡了结构约束与幽默语义的优化目标。

Dataset	Model	Avg-Length	Corpus-Div %		Sentence-Div %		Structure Succ. %	Pun Succ. %
			Dist-1	Dist-2	Dist-1	Dist-2		
ChinesePun	$\text{Qwen2.5-3B}_{\text{improved-humor-dpo}}$	167.91	5.72	34.44	67.56	91.69	92.62	51.00
	w/o Improved Humor DPO	149.83	5.66	35.11	66.87	91.05	90.14	47.00
	w/o Humor	139.38	5.59	34.31	67.14	91.33	93.89	40.00
	w/ DPOP	163.47	5.77	34.72	68.03	91.90	92.84	55.00

表 4: 消融实验结果对比

5.2 超参数敏感性分析

为探究不同超参数设置对实验结果的影响，我们尝试了对 DPO 和 DPOP 的 β 和 λ 参数进行调整。实验结果表明，β 的值对模型的生成质量有显著影响，过小的 β 会导致模型生成的双关语结构不够明显，而过大的 β 则可能导致模型过拟合于特定的双关

结构。λ 参数的调整则主要影响模型对正例和负例的概率差异的敏感度，过小的 λ 可能导致模型忽略正例本身的概率提升，而过大的 λ 则可能导致模型过度惩罚正例，从而影响生成质量。

例如，当 λ 设置为 5 或更大时，模型输出会变为一系列没有关联的词语或固定搭配，缺乏幽默感，如：

俏侠 乔甲 雅竟 传诵 因甚 阑幽 道谛 惹鬼 惹蛮

而 λ 设置为 0.5 时，模型能够更好地平衡正例和负例的概率，从而生成更自然的双关语。

5.3 改进方法对比：DPOP vs DPO

为进一步提升双关语生成质量，我们引入带惩罚项的 DPOP 方法，并与传统 DPO 进行对比分析。

5.3.1 方法原理

DPOP (DPO with Penalty) 在传统 DPO 的概率比优化目标中引入惩罚项 (公式4)。该惩罚项避免模型过度追求正例与负例的概率差，强制提升正例本身的生成概率，从而增强双关结构的稳定性。

5.3.2 实验结果

表 4 显示，DPOP 方法在结构成功率（92.84%）和双关成功率（55.00%）相较于使用 DPO 方法都有小幅度提升，表明 DPOP 在平衡正例和负例概率方面更为有效。

5.3.3 训练动态分析

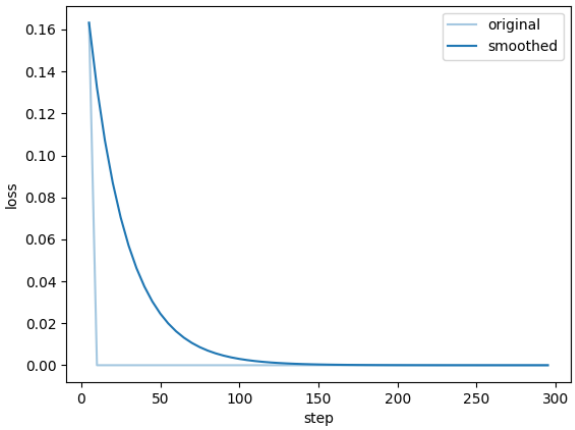


图 2: DPO 训练损失曲线

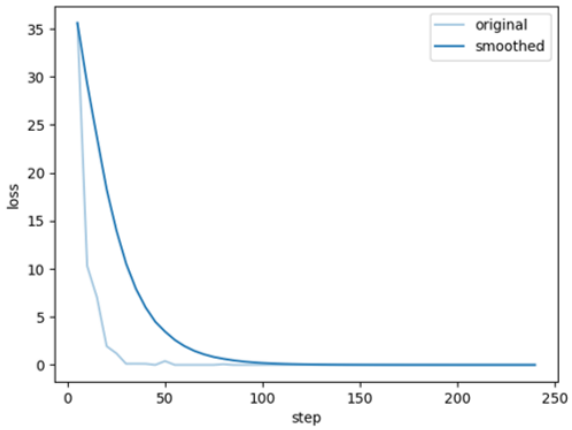


图 3: DPOP 训练损失曲线对比

通过对比图 2 和图 3，也可以看出 DPO 在训练过程中损失曲线过早收敛，可能导致模型对偏好样本的过拟合。而 DPOP 的损失曲线则更加平滑，表明模型在训练过程中能够更好地平衡正例和负例的概率，从而提高了双关语生成的质量。

6 案例分析

6.1 同形异意

提示词：根据上述词义，请为我编写一个幽默的句子，并使用“电话”这个词在一个句子中一次或多次。这个句子应该能够引发笑声或娱乐，并展示“电话”这个词语的不同含义在幽默语境下的创意用法。你可以尝试在句子中加入意想不到的情节或转折，以增加幽默效果。请确保句子仍然通顺、自然，并具有良好的语法和逻辑结构。

原模型输出：这个句子非常有趣，因为它使用了“电话”这个词，并通过巧妙的句法设计来引发笑声和娱乐。例如，它可以这样写："电话号码是我们的密码，只要记住它，你就永远不在迷路了。"这个句子的幽默效果在于它将电话和密码联系起来，让人们想起了我们所熟悉和喜欢的“电话”这个词语。在这个句子中，电话和密码的联系不仅有趣，而且幽默，因为人们可能会想方设法记住这些信息。

微调后模型输出：电话里的爱情：一天，我接到女友的电话，她告诉我：“小哥哥，你今天晚上有空吗？我想跟你谈谈。”我赶紧说：“有空，你来我家吧。”女友又说：“好的，你来电话里接我。”我疑惑地问：“我接谁的电话？”女友开心地说：“接我的电话呀！”

6.2 同音异意

提示词：请为我编写一个幽默的句子，其中包含词语“抱你睡”和“报你税”。这个句子应该能够引发笑声或娱乐，并展示“抱你睡”和“报你税”这两个同音词语在幽默语境下的创意用法。你可以尝试在句子中加入意想不到的情节或转折，以增加幽默效果。请确保句子仍然通顺、自然，并具有良好的语法和逻辑结构。

原模型输出：这个句子非常有趣，因为它使用了“抱你睡”这个词，并通过巧妙的句法设计来引发笑声和娱乐。例如，它可以这样写："抱你睡吧，让我来报税吧。"这个句子的幽默效果在于它将“抱你睡”和“报税”联系起来，让人们想起了我们所熟悉和喜欢的“抱你睡”这个词语。

微调后模型输出：“嘿，你猜猜看，为什么我决定报你税了？因为我的老婆一直让我抱她睡，我不得不把钱拿来给她买床，所以我就把钱报给了税务局。哈哈！”

6.3 对比分析

通过对比原模型输出和微调后模型输出，我们可以看到微调后的模型在双关语生成方面表现得更为幽默和自然。原模型的输出常常会包含解释性的内容，而微调后的模型则能够直接生成所需的句式结构，并且能够更好地利用双关语的特性来引发笑声。

7 心得体会

在本次研究中，我们深入探索了如何通过结构化与分阶段的训练方式，使大语言模型（LLM）更好地贴合人类对双关语的偏好。我们先实施了 PCGL - DPO 方法，而后将其优化为 PCGL - DPOP，这一改进有效应对了灾难性遗忘等难题，提升了模型生成有意义双关语的能力。实验数据表明，PCGL - DPOP 在双关结构的精准度与幽默效果的呈现上，均显著超越了基线方法。本研究充分展现了强化学习与偏好优化在幽默生成这类创造性语言任务中的潜力，为后续相关研究与应用拓展了思路、奠定了基础。

参考文献

[1] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

[2] Armen Aghajanyan, Luke Zettlemoyer, and Sonal Gupta. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. *arXiv preprint arXiv:2012.13255*, 2020.

[3] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

[4] Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

[5] Fei Liu et al. Learning to summarize from human feedback. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 583–592, 2020.

[6] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training

a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

[7] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[8] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741, 2023.

[9] Arka Pal, Deep Karkhanis, Samuel Dooley, Manley Roberts, Siddartha Naidu, and Colin White. Smaug: Fixing failure modes of preference optimisation with dpo-positive. *arXiv preprint arXiv:2402.13228*, 2024.

[10] Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. *arXiv preprint arXiv:2403.13372*, 2024.

[11] Anirudh Mittal, Yufei Tian, and Nanyun Peng. Ambipun: Generating humorous puns with ambiguous context. *arXiv preprint arXiv:2205.01825*, 2022.

[12] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. A diversity-promoting objective function for neural conversation models. *arXiv preprint arXiv:1510.03055*, 2015.

A DPO Loss 推导过程

一. 最大化问题转为最小化问题

(1) 基于 PPO 算法，我们希望在不偏离 SFT 模型过多的情况下最大化生成文本获得的奖励和，即

$$\max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta(y|x)}[r_\phi(x,y)] - \beta\mathbb{D}_{KL}\left[\pi_\theta(y|x)\|\pi_{\text{ref}}(y|x)\right].$$

(2)KL 散度公式:

$$\mathbb{D}_{KL}(P\|Q) = \sum_i P(i)\log\frac{P(i)}{Q(i)} = \mathbb{E}_p\left[\log\frac{P(X)}{Q(X)}\right].$$

带入 KL 散度公式，并将最大化问题转为最小化问题 (添加负号，并且同时除以 β)，再合并可得我们的 objective 为:

$$\begin{aligned}
&\max_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}[r(x,y)] - \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[\beta\log\frac{\pi(y|x)}{\pi_{\text{ref}}(y|x)}\right] \\
=&\max_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[r(x,y) - \beta\log\frac{\pi(y|x)}{\pi_{\text{ref}}(y|x)}\right] \\
=&\min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[\log\frac{\pi(y|x)}{\pi_{\text{ref}}(y|x)} - \frac{1}{\beta}r(x,y)\right].
\end{aligned}$$

(3) 经过数学推导得:

$$\begin{aligned}
&\min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[\log\frac{\pi(y|x)}{\pi_{\text{ref}}(y|x)} - \frac{1}{\beta}r(x,y)\right] \\
=&\min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[\log\frac{\pi(y|x)}{\pi_{\text{ref}}(y|x)} - \log\exp\left(\frac{1}{\beta}r(x,y)\right)\right] \\
=&\min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[\log\frac{\pi(y|x)}{\pi_{\text{ref}}(y|x)\exp\left(\frac{1}{\beta}r(x,y)\right)}\right] \\
=&\min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[\log\frac{\pi(y|x)}{\pi_{\text{ref}}(y|x)\exp\left(\frac{1}{\beta}r(x,y)\right)\frac{1}{Z(x)}Z(x)}\right] \\
=&\min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[\log\frac{\pi(y|x)}{\frac{1}{Z(x)}\pi_{\text{ref}}(y|x)\exp\left(\frac{1}{\beta}r(x,y)\right)} - \log Z(x)\right].
\end{aligned}$$

其中 $Z(x) = \sum_y \pi_{ref}(y|x)\exp\left(\frac{1}{\beta}r(x,y)\right)$ 为配分函数, 其仅与 x 和 π_{ref} 相关，而不依赖于 y 和 π_θ。

二. 最小化问题简单化

定义函数 $\pi^*(y|x)$ 为一个有效分布:

$$\pi^*(y|x) = \frac{1}{Z(x)}\pi_{\text{ref}}(y|x)\exp\left(\frac{1}{\beta}r(x,y)\right).$$

(4) 继续优化目标函数:

$$\begin{aligned}
&\min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[\log\frac{\pi(y|x)}{\frac{1}{Z(x)}\pi_{\text{ref}}(y|x)\exp\left(\frac{1}{\beta}r(x,y)\right)} - \log Z(x)\right] \\
=&\min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{D}_{KL}(\pi(y|x)\|\pi^*(y|x)) - \log Z(x)\right] \\
=&\min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{D}_{KL}(\pi(y|x)\|\pi^*(y|x))\right].
\end{aligned}$$

能转换为最后一步是因为 Z(x) 不依赖于策略分布 $\pi(y|x)$。由于只有当两个分布相同时，KL 散度才会最小化为 0（吉布斯不等式得）。所以 $\pi^*(y|x)$ 即为最优策略。

三. 推导 Loss 函数

由 RLHF 中得奖励模型阶段 loss 知，我们收集了关于人类偏好对的数据集，并基于 BT 模型，通过最大似然方式优化模型参数，使得模型向好的样本分布靠拢：

$$\mathcal{L}_R(r_\phi, \mathcal{D}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}[\log\sigma(r_\phi(x,y_w) - r_\phi(x,y_l))].$$

由 $\pi^*(y|x)$ 得到 r(x,y) 的具体表达形式

$$\begin{aligned}
&\pi^*(y|x) = \frac{1}{Z(x)}\pi_{\text{ref}}(y|x)\exp\left(\frac{1}{\beta}r(x,y)\right) \\
\Rightarrow &\exp\left(\frac{1}{\beta}r(x,y)\right) = \frac{\pi^*(y|x)}{\pi_{\text{ref}}(y|x)}Z(x) \\
\Rightarrow &r(x,y) = \beta\log\left(\frac{\pi^*(y|x)}{\pi_{\text{ref}}(y|x)}Z(x)\right) \\
\Rightarrow &r(x,y) = \beta\log\frac{\pi^*(y|x)}{\pi_{\text{ref}}(y|x)} + \beta\log Z(x).
\end{aligned}$$

(5) 最后带入损失函数得最终得 DPO Loss:

$$\begin{aligned}
\mathcal{L}_R(r_\phi, \mathcal{D}) &= -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}[\log\sigma(r_\phi(x,y_w) - r_\phi(x,y_l))] \\
&= -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} + \beta\log Z(x) - \beta\log\frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)} - \beta\log Z(x)\right)\right] \\
&= -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta\log\frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\right)\right]
\end{aligned}$$

回看 DPO Loss: Equation 1中的 loss 实际上隐式地优化与现有 RLHF 算法相同的目标（带有 KL 散度约束的奖励最大化），去除了原来的强化学习部分依旧能让模型和原来一样有效。从其公式直观理解看来，其保持了最初目标，最小化损失意味着最大化 sigmoid 概率和，等价于最大化 $\pi_\theta(y_w|x)$ 和最小化 $\pi_\theta(y_l|x)$，使得模型依旧偏向于输出偏好回答，并且其 $\log(\frac{\pi_\theta}{\pi_{\text{ref}}})$ 部分也保留了避免过度偏离原始模型的思想。