请你帮我优化，迭代 prompt 的策略

任务：

# 提示词工程考试说明（Prompt Engineering）

## 任务背景

在大数据时代，推荐系统成为各大互联网平台不可或缺的核心技术。传统推荐系统通常包括以下环节：

\* \*\*数据收集\*\*

\* \*\*召回（Recall）\*\*：从海量候选集中快速筛选潜在相关物品。

\* \*\*排序（Ranking）\*\*：对召回结果进行精细排序。

\* \*\*重排（Re-ranking）\*\*：对排序后的候选集进一步精细化处理，直接影响推荐效果。

\* \*\*展示（Display）\*\*

序列推荐（Sequential Recommendation）考虑用户历史行为的时序信息，预测用户下一个行为，捕捉用户兴趣的动态变化，提高推荐的精准性和时效性。

大语言模型（LLM，如GPT、Claude）在推荐系统中展现出以下独特优势：

\* 强大的文本理解和语义分析能力

\* 跨模态理解能力

\* 零样本/少样本学习能力

\* 较强的可解释性

\* 良好的通用性

提示词工程（Prompt Engineering）通过精心设计输入提示（prompts），引导LLM生成符合预期输出的内容，涉及：

\* 提示结构设计

\* 上下文信息组织

\* 任务分解

\* 指令精确表达等

有效的Prompt能帮助LLM更准确理解用户偏好，并生成更个性化的推荐结果。

---

## 考试内容与要求

本次考核聚焦于利用\*\*提示词工程（Prompt Engineering）\*\*引导LLM完成推荐系统的\*\*重排任务（Re-ranking）\*\*：

\* 考生将获得验证集数据文件：`val.jsonl`

\* 每条数据代表用户历史电影观看记录，预测用户下一步可能观看的电影。

\* 要求设计Prompt引导LLM对候选电影（candidates）进行精准排序。

### 验证集数据字段说明（`val.jsonl`）：

```json

{

"user\_id": 5737,

"item\_list": [

[1836, "Last Days of Disco, The"],

[3565, "Where the Heart Is"],

...

],

"target\_item": [1893, "Beyond Silence"],

"candidates": [

[2492, "20 Dates"],

[684, "Windows"],

[1893, "Beyond Silence"],

...

]

}

```

\* `item\_list`：用户历史观看的电影列表（按时间顺序排列）,越靠后表示越近期观看。

\* `target\_item`：用户实际下一部观看的电影（ground truth）。

\* `candidates`：召回阶段提供的电影候选集（包含用户实际观看的下一部电影），一般约20个电影，需重排排序。

---

## 评价指标：NDCG\@K (Normalized Discounted Cumulative Gain)

衡量模型推荐性能的指标，范围 \[0, 1]，越接近1表示推荐质量越高。

### 计算方法：

\* 预测排序列表为 $p = [p\_1, p\_2, ..., p\_k]$

\* 实际观看电影为 $g$

1. \*\*相关性评分\*\*：

\* 若 $p\_i = g$，则 $rel\_i = 1$；否则，$rel\_i = 0$。

2. \*\*折损累积收益 (DCG)\*\*：

$$

DCG@k = \sum\_{i=1}^{k} \frac{rel\_i}{\log\_2(i+1)}

$$

3. \*\*理想折损累积收益 (IDCG)\*\*：

理想情况下，只有一个相关项（置于首位）：

$$

IDCG@k = \frac{1}{\log\_2(1+1)} = 1

$$

4. \*\*归一化折损累积收益 (NDCG)\*\*：

$$

NDCG@k = \frac{DCG@k}{IDCG@k} = DCG@k

$$

### 代码示例：

```python

import math

def calculate\_ndcg\_for\_sample(predicted\_list, ground\_truth\_item, k=10):

predicted\_list = predicted\_list[:k]

relevance = [1 if item\_id == ground\_truth\_item else 0 for item\_id in predicted\_list]

dcg = sum(rel / math.log2(i + 2) for i, rel in enumerate(relevance))

idcg = 1.0 # 只有一个相关项置于首位的情况

ndcg = dcg / idcg if idcg > 0 else 0

return ndcg

# 示例使用

predicted\_list = [111, 1893, 684, 2492, 3654, 2422, 176, 1629, 229, 3155]

ground\_truth\_item = 1893

ndcg = calculate\_ndcg\_for\_sample(predicted\_list, ground\_truth\_item, k=10)

print(f"NDCG@10 = {ndcg}") # 输出: NDCG@10 = 0.63093

```

---

## 考核方式

\* 使用 \*\*OpenAI API\*\* (Chat API模式) 构造 Prompt。

\* 完成以下两个函数（Python标准库）：

### 函数一：构造提示词 (`construct\_prompt`)

```python

def construct\_prompt(d):

"""

构造用于大语言模型的提示词

参数:

d (dict): jsonl数据文件的一行，解析成字典后的变量

返回:

list: OpenAI API的message格式列表

示例: [{"role": "system", "content": "系统提示内容"},

{"role": "user", "content": "用户提示内容"}]

"""

# 实现提示词构造逻辑

```

### 函数二：解析模型输出 (`parse\_output`)

```python

def parse\_output(text):

"""

解析大语言模型的输出文本，提取推荐重排列表

参数:

text (str): 大语言模型在设计prompt下的输出文本

返回:

list: 从输出文本解析出的电影ID列表（python列表格式）

示例: [1893, 3148, 111, ...]

"""

# 实现输出解析逻辑

```

---

## 提交要求（截止时间：北京时间 5月10日 23:59）

提交以下两个文件：

\* \*\*Python文件\*\*：

\* 包含且仅包含以上两个函数

\* 不允许import第三方库，仅可import标准库（如random, re, json等）

\* \*\*探索报告（PDF）\*\*：

\* 记录Prompt优化探索过程（策略、效果分析）

---

## 评分标准（共两部分）

\* \*\*推荐性能客观得分（70%）\*\*：

\* 在私有测试集 (`test.jsonl`) 上的 NDCG\@10 排名与赋分

\* 测试统一使用 \*\*DeepSeek-V3 模型\*\*，temperature设为0

\* \*\*提示词主观评价得分（30%）\*\*：

\* 专家评分，内容包括：

\* 提示词创新性

\* 合理性

\* 可解释性

过去采用的策略与结果：

**1**

**Prompt**

def construct\_prompt(d, \*\*kwargs):

*"""*

*构造用于大语言模型的提示词*

*参数:*

*d (dict): jsonl数据文件的一行，解析成字典后的变量*

*返回:*

*list: OpenAI API的message格式列表，允许设计多轮对话式的prompt*

*示例: [{"role": "system", "content": "系统提示内容"},*

*{"role": "user", "content": "用户提示内容"}]*

*"""*

# 实现提示词构造逻辑

system\_prompt = "你是一名电影推荐专家，根据用户的历史观影记录，对候选电影进行重排，越可能被用户观看的电影排得越靠前。"

user\_history = "\n".join([f"- {movie[1]}" for movie in d['item\_list'][-10:]])

candidate\_movies = "\n".join([f"{movie[0]}: {movie[1]}" for movie in d['candidates']])

user\_prompt = f"用户最近观看的电影：\n{user\_history}\n\n请根据用户的兴趣，对以下候选电影进行排序（输出电影ID列表，最可能观看的电影在最前）：\n{candidate\_movies}\n\n直接输出电影ID列表，不要额外的解释或文字。"

prompt\_messages = [

{"role": "system", "content": system\_prompt},

{"role": "user", "content": user\_prompt}

]

# 记录提示词内容

logging.info(f"构建提示词 - 用户历史记录:\n{user\_history}")

logging.info(f"构建提示词 - 候选电影列表:\n{candidate\_movies}")

logging.info(f"构建提示词 - 完整用户提示:\n{user\_prompt}")

return prompt\_messages

**result**

{

"strategy\_name": "expert\_instruction\_id\_first\_plain",

"overall\_avg\_ndcg": 0.5600152346251777,

"trial\_avg\_scores": [

0.5396863601688711,

0.5476338369881166,

0.6163954933611995,

0.5359116836141369,

0.59841960319746,

0.5369213526337144,

0.5571757637910698,

0.5738283772765687,

0.5616408976708105,

0.5325389775498293

],

"sample\_stats": {

"0": {

"mean\_ndcg": 0.9333333333333333,

"std\_ndcg": 0.2,

"scores": [

1.0,

1.0,

1.0,

1.0,

1.0,

1.0,

1.0,

1.0,

0.3333333333333333,

1.0

]

},

"1": {

"mean\_ndcg": 0.6874834908425163,

"std\_ndcg": 0.2290834942711796,

"scores": [

1.0,

0.6309297535714575,

1.0,

0.6309297535714575,

0.38685280723454163,

0.6309297535714575,

0.6309297535714575,

0.3333333333333333,

1.0,

0.6309297535714575

]

},

"2": {

"mean\_ndcg": 0.7635849255056936,

"std\_ndcg": 0.25300915632492793,

"scores": [

0.6309297535714575,

1.0,

1.0,

0.6309297535714575,

1.0,

0.3562071871080222,

1.0,

0.6309297535714575,

1.0,

0.38685280723454163

]

},

"3": {

"mean\_ndcg": 0.5671586168152145,

"std\_ndcg": 0.2965097707445766,

"scores": [

1.0,

0.31546487678572877,

0.3562071871080222,

0.38685280723454163,

1.0,

0.3333333333333333,

0.31546487678572877,

1.0,

0.6309297535714575,

0.3333333333333333

]

},

"4": {

"mean\_ndcg": 0.495083867879341,

"std\_ndcg": 0.2926984956230798,

"scores": [

0.3562071871080222,

0.3562071871080222,

1.0,

0.31546487678572877,

1.0,

0.43067655807339306,

0.43067655807339306,

0.43067655807339306,

0.6309297535714575,

0.0

]

},

"5": {

"mean\_ndcg": 0.36037112298868096,

"std\_ndcg": 0.12843783135645334,

"scores": [

0.3562071871080222,

0.0,

0.43067655807339306,

0.43067655807339306,

0.5,

0.3562071871080222,

0.43067655807339306,

0.3562071871080222,

0.38685280723454163,

0.3562071871080222

]

},

"6": {

"mean\_ndcg": 0.36487123143774564,

"std\_ndcg": 0.3078673383737043,

"scores": [

0.3333333333333333,

0.38685280723454163,

0.6309297535714575,

0.0,

0.3333333333333333,

0.0,

0.0,

1.0,

0.3333333333333333,

0.6309297535714575

]

},

"7": {

"mean\_ndcg": 0.46036318131291887,

"std\_ndcg": 0.39946845604860043,

"scores": [

0.0,

1.0,

0.31546487678572877,

0.6309297535714575,

0.0,

1.0,

1.0,

0.0,

0.3010299956639812,

0.3562071871080222

]

},

"8": {

"mean\_ndcg": 0.5871363910600955,

"std\_ndcg": 0.2824420072751108,

"scores": [

0.3333333333333333,

0.3562071871080222,

0.43067655807339306,

1.0,

0.43067655807339306,

0.6309297535714575,

0.3333333333333333,

0.3562071871080222,

1.0,

1.0

]

},

"9": {

"mean\_ndcg": 0.38076618507623666,

"std\_ndcg": 0.22073289217637412,

"scores": [

0.38685280723454163,

0.43067655807339306,

0.0,

0.3333333333333333,

0.3333333333333333,

0.6309297535714575,

0.43067655807339306,

0.6309297535714575,

0.0,

0.6309297535714575

]

}

},

"num\_trials": 10,

"num\_samples": 10,

"k": 10

}

**Analysis**

**2**

**Prompt**

def construct\_prompt(d, \*\*kwargs):

# 实现提示词构造逻辑

system\_prompt = """## 角色定位

您是世界顶尖的电影推荐系统，擅长通过时序行为分析用户深层次兴趣。请严格执行以下工作流：

## 核心任务

对20部候选电影进行专业级重排序，使最符合用户即时偏好的电影位列前茅

## 能力特征

1. 时空感知力：精准捕捉用户兴趣的时空演变模式

2. 语义关联力：识别电影间的隐含联系（导演/流派/叙事风格）

3. 冷启动力：当候选集与历史兴趣不匹配时，基于电影艺术价值排序"""

user\_history = "\n".join([f"- {movie[1]}" for movie in d['item\_list'][-10:]])

candidate\_movies = "\n".join([f"{movie[0]}: {movie[1]}" for movie in d['candidates']])

user\_prompt = f"""## 用户观影时空轨迹（倒序排列）

{user\_history(d['item\_list'])}

## 候选电影资料库

{candidate\_movies(d['candidates'])}

## 决策框架

1. 近期兴趣建模：分析最后3部电影的核心特征

- 关键要素：影片类型｜导演风格｜叙事结构

- 示例推导：若近期观看《黑暗骑士》《盗梦空间》，则强化"诺兰式非线性叙事"权重

2. 跨时空关联分析：

√ 长期兴趣：识别历史记录中的稳定偏好（如持续选择科幻片）

√ 短期波动：检测近期异常选择（如突然观看爱情片）

3. 候选集多维评估：

[优先级] 导演/演员重合 > 相同子类型 > IMDb评分 > 上映时间接近性

## 输出规范

✅ 必须包含全部20个ID

✅ 严格按预测概率降序排列

✅ 仅使用英文逗号分隔

❌ 禁止任何自然语言

示例：3021,485,1763,..."""

prompt\_messages = [

{"role": "system", "content": system\_prompt},

{"role": "user", "content": user\_prompt}

]

# 记录提示词内容

logging.info(f"构建提示词 - 用户历史记录:\n{user\_history}")

logging.info(f"构建提示词 - 候选电影列表:\n{candidate\_movies}")

logging.info(f"构建提示词 - 完整用户提示:\n{user\_prompt}")

return prompt\_messages

**result**

{

"strategy\_name": "expert\_instruction\_id\_first\_plain",

"overall\_avg\_ndcg": 0.5930578024942887,

"trial\_avg\_scores": [

0.5812753143731058,

0.6160873493095436,

0.6040656414580688,

0.6115125785546058,

0.5920668511716146,

0.5643145568359511,

0.5728187082569912,

0.5715409918990997,

0.6084480165419539,

0.6084480165419539

],

"sample\_stats": {

"0": {

"mean\_ndcg": 0.8333333333333333,

"std\_ndcg": 0.34156502553198664,

"scores": [

0.0,

1.0,

1.0,

1.0,

1.0,

0.3333333333333333,

1.0,

1.0,

1.0,

1.0

]

},

"1": {

"mean\_ndcg": 0.7261859507142915,

"std\_ndcg": 0.29257780009232065,

"scores": [

1.0,

1.0,

1.0,

0.6309297535714575,

1.0,

1.0,

0.6309297535714575,

0.3333333333333333,

0.3333333333333333,

0.3333333333333333

]

},

"2": {

"mean\_ndcg": 0.694630873461558,

"std\_ndcg": 0.2722083530777512,

"scores": [

0.6309297535714575,

0.38685280723454163,

1.0,

0.3333333333333333,

1.0,

0.6309297535714575,

1.0,

0.6309297535714575,

1.0,

0.3333333333333333

]

},

"3": {

"mean\_ndcg": 0.8323465818787765,

"std\_ndcg": 0.21253024157666034,

"scores": [

1.0,

1.0,

0.6309297535714575,

1.0,

0.6309297535714575,

1.0,

1.0,

1.0,

0.43067655807339306,

0.6309297535714575

]

},

"4": {

"mean\_ndcg": 0.5674106259877844,

"std\_ndcg": 0.23590684284252705,

"scores": [

1.0,

0.43067655807339306,

0.3333333333333333,

0.43067655807339306,

0.3562071871080222,

0.43067655807339306,

0.43067655807339306,

0.6309297535714575,

0.6309297535714575,

1.0

]

},

"5": {

"mean\_ndcg": 0.3863277072857313,

"std\_ndcg": 0.3474945538247504,

"scores": [

0.3562071871080222,

0.0,

0.3333333333333333,

1.0,

0.0,

0.43067655807339306,

0.38685280723454163,

1.0,

0.0,

0.3562071871080222

]

},

"6": {

"mean\_ndcg": 0.5168645259447266,

"std\_ndcg": 0.363660084399789,

"scores": [

0.43067655807339306,

1.0,

0.38685280723454163,

0.38685280723454163,

0.6309297535714575,

0.0,

0.3333333333333333,

0.0,

1.0,

1.0

]

},

"7": {

"mean\_ndcg": 0.32454395542565817,

"std\_ndcg": 0.28246160345120136,

"scores": [

0.43067655807339306,

0.3562071871080222,

0.0,

0.0,

0.31546487678572877,

1.0,

0.0,

0.3562071871080222,

0.3562071871080222,

0.43067655807339306

]

},

"8": {

"mean\_ndcg": 0.6399859146463662,

"std\_ndcg": 0.2591388461357098,

"scores": [

0.6309297535714575,

0.6309297535714575,

1.0,

1.0,

0.3562071871080222,

0.38685280723454163,

0.6309297535714575,

0.43067655807339306,

0.3333333333333333,

1.0

]

},

"9": {

"mean\_ndcg": 0.4089485562646624,

"std\_ndcg": 0.24449621204013675,

"scores": [

0.3333333333333333,

0.3562071871080222,

0.3562071871080222,

0.3333333333333333,

0.6309297535714575,

0.43067655807339306,

0.31546487678572877,

0.3333333333333333,

1.0,

0.0

]

}

},

"num\_trials": 10,

"num\_samples": 10,

"k": 10

}

**Analysis**

**3**

**Prompt**

def construct\_prompt(d, \*\*kwargs):

system\_prompt = """## 专家级电影推荐引擎指令

【核心能力维度】

1. 动态兴趣建模：采用指数时间衰减算法，最后3部电影权重占比70%

2. 深度语义关联：精确识别导演-演员-编剧铁三角关系

3. 冷启动突破策略：当候选与历史不匹配时，按以下优先级排序：

(1) 同导演其他作品 (2) 同演员作品 (3) 同类型高评分影片

【分析框架】

1. 时间轴分析：绘制用户兴趣演变曲线，标注关键转折点

2. 导演关系图谱：建立候选电影与历史观影的导演关联矩阵

3. 类型进化树：识别用户从主流类型到细分类型的迁移路径"""

# 增强型历史记录展示（带时间衰减标记）

history\_items = list(reversed(d['item\_list'][-10:]))

time\_decay\_weights = ["(权重%.1f)" % (0.5 \*\* i) for i in range(len(history\_items))]

user\_history = "\n".join(

f"[t-{i}] {movie[1]} {time\_decay\_weights[i]}"

for i, (\_, movie) in enumerate(history\_items)

)

# 候选电影增强描述（激发模型常识）

candidate\_movies = "\n".join(

f"{id}: {title} 【关联线索】"

f"导演可能作品:, "

f"相似类型:"

for id, title in d['candidates']

)

user\_prompt = f"""## 用户观影时空图谱（倒序·时间衰减系数0.5）

{user\_history}

## 候选电影分析矩阵（共20部）

{candidate\_movies}

## 推荐决策协议

1. 近期兴趣聚焦：计算最后3部电影的导演D-score、类型G-score

- D-score公式：Σ(1/(1+时间差)) \* 导演重合度

- 示例：若最后3部含诺兰电影，则D-score+=3.0

2. 跨期关联检测：

√ 持续兴趣：连续5次以上选择同类型电影，权重×1.5

√ 短期探索：近3部出现新类型时，权重×0.8

3. 冷启动应急方案：

! 当候选与历史无直接关联时，启用：

a) 查找候选电影中IMDb评分>7.5的作品

b) 优先奥斯卡获奖导演的早期作品

c) 选择与用户最喜爱演员合作过的导演作品

## 输出规范（违规将导致系统故障）

✅ 格式：严格按"id1,id2,...,id20"排列，推荐度降序

✅ 必须包含全部20个ID，不得重复

❌ 禁止任何解释性文字，仅数字和逗号

正确示例：1695,3221,1829,2755,2813,...,3606

错误示例：我认为用户会喜欢1695因为..."""

prompt\_messages = [

{"role": "system", "content": system\_prompt},

{"role": "user", "content": user\_prompt}

]

return prompt\_messages

**result**

{

"strategy\_name": "expert\_instruction\_id\_first\_plain",

"overall\_avg\_ndcg": 0.5861373959903048,

"trial\_avg\_scores": [

0.5552176827750188,

0.5547726300329978,

0.6137999639320747,

0.5720415316218082,

0.6248996447822395,

0.4943682897302514,

0.6550893636588141,

0.6089485562646624,

0.5725105642053354,

0.6097257328998454

],

"sample\_stats": {

"0": {

"mean\_ndcg": 0.8630929753571458,

"std\_ndcg": 0.2738140492857085,

"scores": [

1.0,

1.0,

1.0,

1.0,

0.31546487678572877,

1.0,

1.0,

0.31546487678572877,

1.0,

1.0

]

},

"1": {

"mean\_ndcg": 0.7190385680952497,

"std\_ndcg": 0.25383991291845986,

"scores": [

0.6309297535714575,

0.6309297535714575,

0.3333333333333333,

1.0,

1.0,

0.3333333333333333,

1.0,

1.0,

0.6309297535714575,

0.6309297535714575

]

},

"2": {

"mean\_ndcg": 0.6980703001559769,

"std\_ndcg": 0.2691060271226704,

"scores": [

0.3010299956639812,

0.3562071871080222,

0.43067655807339306,

0.6309297535714575,

1.0,

0.6309297535714575,

1.0,

1.0,

0.6309297535714575,

1.0

]

},

"3": {

"mean\_ndcg": 0.765880900457323,

"std\_ndcg": 0.2975940231034209,

"scores": [

1.0,

1.0,

1.0,

0.3562071871080222,

1.0,

0.6309297535714575,

1.0,

0.3562071871080222,

1.0,

0.31546487678572877

]

},

"4": {

"mean\_ndcg": 0.5902009530727474,

"std\_ndcg": 0.28007609252571214,

"scores": [

0.3333333333333333,

1.0,

0.38685280723454163,

0.43067655807339306,

1.0,

0.6309297535714575,

1.0,

0.3562071871080222,

0.3333333333333333,

0.43067655807339306

]

},

"5": {

"mean\_ndcg": 0.5140064127842953,

"std\_ndcg": 0.24814544529137375,

"scores": [

0.5,

0.38685280723454163,

1.0,

0.3562071871080222,

0.31546487678572877,

0.43067655807339306,

0.3333333333333333,

0.43067655807339306,

0.38685280723454163,

1.0

]

},

"6": {

"mean\_ndcg": 0.6063670800167056,

"std\_ndcg": 0.3454457151302808,

"scores": [

0.43067655807339306,

0.38685280723454163,

1.0,

0.31546487678572877,

0.0,

0.5,

0.43067655807339306,

1.0,

1.0,

1.0

]

},

"7": {

"mean\_ndcg": 0.34732569293646076,

"std\_ndcg": 0.2952425873034806,

"scores": [

1.0,

0.0,

0.3562071871080222,

0.0,

0.3562071871080222,

0.3562071871080222,

0.0,

0.6309297535714575,

0.38685280723454163,

0.38685280723454163

]

},

"8": {

"mean\_ndcg": 0.3404950439432352,

"std\_ndcg": 0.328028257373231,

"scores": [

0.0,

0.3562071871080222,

0.0,

0.6309297535714575,

0.6309297535714575,

0.43067655807339306,

0.3562071871080222,

1.0,

0.0,

0.0

]

},

"9": {

"mean\_ndcg": 0.4168960330839079,

"std\_ndcg": 0.2817737586480174,

"scores": [

0.3562071871080222,

0.43067655807339306,

0.6309297535714575,

1.0,

0.6309297535714575,

0.0,

0.43067655807339306,

0.0,

0.3562071871080222,

0.3333333333333333

]

}

},

"num\_trials": 10,

"num\_samples": 10,

"k": 10

}

**Analysis**

**4**

**Prompt**

def construct\_prompt(d, \*\*kwargs):

system\_prompt = """## 电影推荐专家系统v4-时空感知增强版

【核心原则】

1. 动态衰减机制：最后3部电影占70%权重，4-10部占30%

2. 三维关联分析：导演(50%) > 演员(30%) > 编剧(20%)

3. 冷启动三层过滤：评分过滤 → 奖项验证 → 跨作品演员关联

【智能分析模块】

1. 时间衰减计算器：自动生成观影兴趣热度曲线

2. 合作网络图谱：挖掘导演-演员跨作品合作历史

3. 类型进化检测：识别用户从主流到细分类型的迁移路径"""

# 时间轴增强表示

history = list(reversed(d['item\_list'][-10:]))

timeline = "\n".join(

f"[{i + 1}] {movie[1]} 【权重{70 // (i + 1) if i < 3 else 30 // (i + 1)}%】"

for i, (\_, movie) in enumerate(history)

)

# 候选电影语义增强

candidates = "\n".join(

f"{id}: {title} 【关联分析点】"

f"1.导演作品风格推测 2.与用户历史电影的类型连续性 3.主演合作可能性"

for id, title in d['candidates']

)

user\_prompt = f"""## 用户观影时空图谱（倒序·动态衰减）

{timeline}

## 候选电影分析矩阵

{candidates}

## 推荐决策协议

1. 导演网络分析：

- 计算导演关联度D-score = Σ(历史出现次数 × 时间衰减因子)

- 示例：若用户历史含3部诺兰电影，则候选中诺兰作品D-score=3×0.7=2.1

2. 冷启动应急流程：

! 当无导演/演员关联时：

→ 第一层：选择IMDb>7.0且年份>2000的影片

→ 第二层：挑选获奖记录≥2项的作品

→ 第三层：选择与用户最喜爱演员（历史出现最多）合作过的导演

3. 类型连续性检测：

- 建立类型进化链：历史类型 → 当前候选类型

- 拒绝推荐跨越超过2个类型分支的候选

## 输出规范（系统级强制约束）

✅ 格式：严格遵循[id1,id2,...,id20]结构，推荐度降序

✅ 完整性：必须包含全部20个ID，无重复

❌ 违规后果：格式错误将触发系统重新生成

正确示例：[1695,3221,1829,2755,2813,...,3606]

错误示例：用户可能喜欢1695（诺兰新作）..."""

return [

{"role": "system", "content": system\_prompt},

{"role": "user", "content": user\_prompt}

]

**result**

{

"strategy\_name": "expert\_instruction\_id\_first\_plain",

"overall\_avg\_ndcg": 0.5670972021585087,

"trial\_avg\_scores": [

0.5812753143731056,

0.5697541462443392,

0.5302515921723603,

0.6082822447876707,

0.5176732097190908,

0.6061606311644849,

0.5382076585657664,

0.5563530268355377,

0.5563530268355377,

0.6066611708871934

],

"sample\_stats": {

"0": {

"mean\_ndcg": 1.0,

"std\_ndcg": 0.0

},

"1": {

"mean\_ndcg": 0.5681801357568975,

"std\_ndcg": 0.1257069346723657

},

"2": {

"mean\_ndcg": 0.6115039889804453,

"std\_ndcg": 0.2742617561189485

},

"3": {

"mean\_ndcg": 0.5999914104258395,

"std\_ndcg": 0.3563103749916302

},

"4": {

"mean\_ndcg": 0.6052872229368742,

"std\_ndcg": 0.2677660870116195

},

"5": {

"mean\_ndcg": 0.5332631452730793,

"std\_ndcg": 0.32917294362719296

},

"6": {

"mean\_ndcg": 0.6356207187108022,

"std\_ndcg": 0.4568092139616978

},

"7": {

"mean\_ndcg": 0.42282729783115175,

"std\_ndcg": 0.24521031162235113

},

"8": {

"mean\_ndcg": 0.35151567234929326,

"std\_ndcg": 0.2970707075070688

},

"9": {

"mean\_ndcg": 0.34278242932070413,

"std\_ndcg": 0.202361550130414

}

},

"num\_trials": 10,

"num\_samples": 10,

"k": 10

}

**Analysis**