

# 医疗问答端到端示例：Flex-Attn 生成“奥司他韦”专业定义

- 作者：GaoZheng
- 日期：2025-09-26
- 版本：v1.0.0

## 摘要

阐述可变成成本注意力（Flex-Attn）的动机、设计与实现：在合规约束与预算限制下，按需分配注意力计算资源。文中拆解组件与调用关系、关键超参与时间/显存开销，并给出与历史/状态缓存结合的工程实践与调优建议。

在“请问什么是‘奥司他韦’？”场景，按**逐字一步**决策，动态选择  $L_h$ （历史注意力）与  $L_p$ （预测命中上限），以 U 上“**最长可用命中**（命中即停）”覆盖“磷酸奥司他韦/神经氨酸酶/特异性/抑制剂”等术语；以**语义门控 + IDF**给奖励“加闸”，训练期禁 Top-p 保证分布一致，日志 JSONL 全链路可回放。预期 KPI：术语覆盖↑、`word_noncompliance` ↓≥30%，收敛更稳、吞吐可控。可以用“**文件数据库（表/JSON） + 有限状态索引（Trie/DAWG/AC 自动机）**”在零训练的前提下实现一个低配版 Flex-Attn 管线。此时 SAC/NN 的角色不是“不可替代的生成器”，而是“**过拟合式的动态超参控制器 + 学习型索引（压缩记忆） + 基于隶属度/概率的快速查询器**”。换言之：规则可跑通，神经网络让它“更快、更稳、更省”。

## 1. 架构对照（表驱动 vs. 学习索引）

维度	表/JSON + 有限状态索引（零训练）	SAC/NN（学习索引 + 超参控制）
目标	正确性、可审计、可回放	压缩记忆、延迟/成本优化、在线自适应
知识形式	词典 Catalog + 长度集合 U + 规则	规则不变；学习到 $\lambda, \tau, L_h, L_p$ 等超参与命中概率
匹配	反向 Trie/DAWG/AC， <b>最长可用命中</b> ( $\leq L_p$ )	同上，但 <b>命中概率/置信度</b> 由 NN 估计，支持不完全匹配

维度	表/JSON + 有限状态索引（零训练）	SAC/NN（学习索引 + 超参控制）
语义门控	纯符号相似度（Jaccard/Dice/BM25-Lite）	学习到的相似度/隶属度（更稳、更快）
注意力长度	Lh/Lp 由 <b>策略表</b> 决定	Lh/Lp 由 <b>策略头</b> 自适应，兼顾吞吐
运维	配置/热更/审计友好	需要训练与指标门禁（可灰度）

## 2. 表驱动实现（零训练）——可直接落地

### 2.1 数据建模（JSON Schema）

```
// catalog/med_terms.jsonl
{"seg": "磷酸奥司他韦", "len": 6, "idf": 0.90, "domain": "drug", "aliases": ["达菲"], "pos": ["n"]}
{"seg": "奥司他韦", "len": 4, "idf": 0.85, "domain": "drug"}
{"seg": "神经氨酸酶", "len": 4, "idf": 0.87, "domain": "target"}
{"seg": "特异性", "len": 3, "idf": 0.80, "domain": "attr"}
{"seg": "抑制剂", "len": 3, "idf": 0.82, "domain": "class"}
{"seg": "流感病毒", "len": 3, "idf": 0.88, "domain": "pathogen"}
{"seg": "宿主细胞", "len": 4, "idf": 0.86, "domain": "bio"}
```

```
// config/flex_attn.json
{
  "U": [2, 3, 4, 5, 6, 8],           // 禁 1 字
  "Lh_policy": [2, 3, 4, 6],         // 历史窗口候选（表驱动）
  "Lp_policy": [3, 4, 5, 6],         // 预测命中上限候选
  "tau_semantic_gate": 0.75,
  "lambda_lex": 0.30,               // 词法增益权重
  "idf_downweight_2gram": 0.6,      // 二字降权
  "ban_single_char_reward": true,
  "len_cost": {"lambda_h": 0.08, "lambda_p": 0.10, "alpha_h": 1.0, "alpha_p": 1.0}
}
```

## 2.2 核心流程（伪代码）

```
# 预处理：构建反向 Trie/AC 自动机（按 U 做最长命中），载入 IDF
index = build_reverse_trie(load_jsonl("catalog/med_terms.jsonl"))

function step(prev, chi, cfg):
    Lh = policy_table_pick_Lh(prev, chi, cfg.Lh_policy)
    # 策略表
    Lp = policy_table_pick_Lp(prev, chi, cfg.Lp_policy)
    a = choose_char_masked(prev, Lh)
    # 合法字符集合内选一字（可贪心/打分）

    q, seg = longest_suffix_hit_with_cap(a, future, index, cap=Lp)
    # 命中即停 ( $\leq Lp$ )
    sim = lexical_similarity(prev, target_context)
    # Jaccard/Dice/BM25-lite
    if seg != NONE:
        idf = lookup_idf(seg);
        delta = cfg.lambda_lex * max(0, sim - cfg.tau_semantic_gate) * idf * downweight_if_len2
    else:
        delta = 0

    len_cost = cost_len(Lh, Lp, cfg.len_cost)
    reward = base(prev) + eta1*soft_quality(prev) + eta2*delta - len_cost
    return a, seg, reward, {Lh, Lp, sim, idf}
```

### 要点：

- **最长可用命中 ( $\leq Lp$ )** 与 **语义门控 ( $sim > \tau$ )** 完全可以用表/规则实现；
- 相似度可选 **字符 n-gram Jaccard/Dice、BM25-Lite、最长公共子序列** —— 无需训练；
- **日志**写 JSONL: `step,Lh,Lp,a,seg,len,idf,sim,delta,reward`, 可回放。

## 3. SAC/NN 的“增值位”

把 NN 当作三件事：**压缩、超参、快速查**

### 1. 压缩记忆 (Learning to Index)

- 把“分布在 JSON/Trie 的离散知识”压成向量/概率表，做**学习型布隆/词典近似器**，在**长尾/OOV**处给出“类命中概率”。
- 作用：**召回不降、延迟下降**，索引内存显著缩小（尤其跨域 Catalog）。

2. **动态超参 (Controller)**

- 在线学习  $\lambda$ 、 $\tau$ 、 $L_h$ 、 $L_p$  等；对应你的“**过拟合式动态超参优化**”。
- 作用：不同域/流量/约束下自动达成**质量-成本最优点**（可设 KPI 目标做闭环）。

3. **基于隶属度/概率的快速查询**

- 在命中边界模糊时，直接输出**隶属度** (membership) 或**置信度**，替代昂贵的规则多路分支。
- 作用：**少分支、快决策**；与表驱动并行可做 **早停/剪枝**。

简单说：**规则 = 正确性**；**NN = 性价比**。没有 NN 也能跑；加 NN，**更快更稳更省**。

# 4. 混合运行模式（可灰度）

模式	描述	何时用
A 表驱动	仅规则/索引/IDF，相似度纯符号	首次上线、合规/审计严格、数据稀缺
B 表 + SAC 控制器	规则不变；SAC 只学 $\lambda$ 、 $\tau$ 、 $L_h$ 、 $L_p$ 与命中置信度； <b>不产文本</b>	追求 <b>吞吐/QPS</b> 与 <b>跨域自适应</b>
C 全量 SAC	Flex-Attn + 学习索引 + 生成；表作为兜底	成熟期、对 <b>性能/覆盖</b> 要更高

# 5. KPI/成本口径（管理层要看的）

- **质量**：术语覆盖率  $\uparrow$ 、`word_noncompliance`  $\downarrow \geq 30\%$ 、错别率  $\downarrow$
- **效率**：P50/P95 延迟、QPS、CPU/内存占用；索引大小
- **稳健**：域切换回归幅度  $\leq$  阈值、灰度 A/B 显著性 ( $p < 0.01$ )
- **治理**：命中门控日志回放成功率 100%，一键回滚机制

## 6. 风险与补丁（零训练版本也适用）

- 长词偏置：最长命中过度 → Lp 上限 + 长度成本 + IDF/二字降权 + 语义门控
  - 词典投机：高频堆砌 → 禁 1 字奖励；停用词黑名单
  - 吞吐压力：U 扫描与日志 I/O → 反向 Trie/AC + 命中缓存 + 批量/采样写
  - 策略僵化：不同域表现差 → 策略表分域化；若上 SAC 控制器，则在线调参
- 

## 7. 快速落地清单（两周版）

1. 建索引：把医疗/法规/电商等域的 Catalog 落成 JSONL + 反向 Trie/AC；生成 IDF。
  2. 跑通流程：U-最长命中 ( $\leq L_p$ ) + 语义门控 ( $\text{sim} > \tau$ ) + 长度成本；写 JSONL 日志。
  3. 仪表盘：术语覆盖、命中长度分布、Top-K 高频命中词占比、P50/P95、QPS。
  4. 灰度策略：先 模式 A；指标达标后引入 模式 B（SAC 只做  $\lambda$ 、 $\tau$ 、 $L_h$ 、 $L_p$  的动态控制）。
  5. 回滚/审计：一键切回 A；全链路回放脚本。
- 

## 最后一针见血

- 是的，可以不用训练，纯表也能落地；
  - SAC/NN 的核心价值在“压缩-记忆-超参-快查”，把同样的逻辑跑得更快、更便宜、更稳；
  - 按 A→B→C 的路线演进，保住审计与可控性的同时，把 ROI 做到最优。
- 

### 许可声明 (License)

Copyright (C) 2025 GaoZheng

本文档采用[知识共享-署名-非商业性使用-禁止演绎 4.0 国际许可协议 \(CC BY-NC-ND 4.0\)](#)进行许可。