

“微分动力量子（MDQ）”在离散化LLM的工程化落地：最小单元、线性积累、热插拔与统一版本治理

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

摘要

提出 MDQ 机制稳定离散 LLM/策略管道：支持小单元交互与统一版本控制，缓解长序列采样的非平稳与暴露偏差。结合指令设计与记忆扩展策略，给出训练/推理一体化的实现路线与评估指标。

- MDQ (Micro Differential Quantum)**：对离散化LLM中“策略与索引”的**最小可执行增量**。它不生成内容，只更新**控制面**：算子门控、长度窗口、词法权重、语义阈值、索引隶属度。
- 作用域**（与此前体系对齐）
 - 算子域 (Lex-KAT作用么半群)**： $\{\mathbf{L}, \mathbf{R}, \Pi, \mathbf{T}, \mathbf{Cl}, \dots\}$ 的门控/权重。
 - 长度域 (Flex-Attn)**：历史窗口 L_h 、预测上限 L_p 。
 - 索引域 (Catalog/IDF/别名/隶属度)**：文件→内存DB的快速检索结构。
 - 策略域 (GRL路径积分)**：价值泛函 \mathcal{J} 的微分方向与步长量化。
- 目标函数 (ROI口径)**

$$\mathcal{J} = \mathbb{E}\left[\sum_t \gamma^t (S_t + \delta_t - C_t)\right], \quad \Delta = \text{MDQ增量}, \quad \text{只更新“如何做”而非“做什么”}$$

2. MDQ分类（可热插拔原子）

- Op-MDQ**：算子门控/权重增量（如 $\lambda_{\text{lex}}, \tau$ 、黑/白词test开关）。
- Len-MDQ**： L_h, L_p 的窗口与上限调整（含长度成本权重）。
- Idx-MDQ**：索引泛函的增量（IDF/隶属度/别名/领域词条新增/降权）。
- Gate-MDQ**：合规与预算策略（tests）的阈值与优先级调整。

每个MDQ自带**可逆元数据**（回滚参数）、**适用域**、**安全约束**（必过tests）、**预期KPI提升**。

3. 索引泛函与“线性积累”（无需训练）

- 索引泛函：

$$\mathcal{I}(\text{seg}) = \sum_k w_k \varphi_k(\text{seg}), \quad \varphi_k : \text{特征 (IDF、别名、同义、向量近邻、正则)}$$

- MDQ线性积累：

$$\mathcal{I}_{t+1} = \mathcal{I}_t \oplus \underbrace{\sum_j \Delta_j^{\text{Idx}} \varphi_j}_{\text{MDQ增量}}, \quad (\oplus = \text{加权叠加或max, 根据半环选择})$$

- 可交换/可合并：同一特征族的MDQ按权重**幂等合并**（去重/取极值）；跨族遵从优先级表（tests > 长度 > 词法 > 语义）。

4. 成本最小化与“平衡点”策略

- 最小MDQ：一次只更新**一个原子**（单参数/单规则/单词条）。
- 平衡求解（多目标）：

$$\max_{\Delta} \Delta^\top \mathbf{v} - \frac{1}{2} \Delta^\top \Lambda \Delta, \quad \text{s.t. 合规tests、QPS/SLA、回滚可达}$$

- \mathbf{v} ：价值基准向量（边际收益）； Λ ：非交换/资源耦合惩罚矩阵。
- 解得 $\Delta^* = \Lambda^{-1} \mathbf{v}$ 后，再做**量化与裁剪**（见§5）。

5. MDQ生成与量化（可执行最小步）

- 来源：A/B评估梯度、占用测度（算子/长度/词条使用率）、离线回放统计、合规事件。
- 量化：

$$\Delta_i = Q(v_i) - \lambda_{\text{comm}} \sum_j \|[G_i, G_j]\| \pi_j, \quad Q(v) = \text{sgn}(v) \cdot \min\{|v|^\beta, \eta\}$$

- 抑制不可交换算子同时上调； β 次线性稳态， η 上限控风险。
- 裁剪： $\Delta \leftarrow \Pi_{\text{adm}}(\Delta)$ （长度/阈值/权重落在白名单区间）。

6. 架构：混合NN×非NN与临时知识缓冲区

- 文件→内存DB (非NN)：JSONL/表格的 Catalog/IDF/别名 → 反向Trie/AC/向量桶；支持**增量索引** (Idx-MDQ落地即生效)。
- 临时知识缓冲区 (EKB)：
 - 分层：cold(file)→warm(memory)→hot(cache)；
 - **租约TTL + 命中计数** 形成“热度-清除”策略；
 - 变更以 **MDQ ledger** 追加写，支持**时间旅行回放**。
- 轻NN (可选)：只做**控制器与近似相似度** (LoRA/蒸馏小头)，不直接产文；其输出也被**量化为MDQ回写索引与门控**。

7. 热插拔与版本治理 (统一逻辑兼容)

- 包格式 (MDQ-pkg)

```
{
  "name": "mdq.idx.ifu.idf-boost",
  "semver": "1.4.2",
  "scope": ["MED", "RAG"],
  "atoms": [{"type": "Idx", "feature": "idf", "seg": "神经氨酸酶", "delta": +0.07}],
  "tests": ["legal", "budget", "blacklist"],
  "rollback": {"inverse": [...]},
  "kpi": {"target": {"w_nc_drop": 0.3}}
}
```

- **兼容矩阵**：[Lex – KAT算子版本 × 索引|schema × Gate策略] → OK/警告/拒绝。
- **热插拔流程**：双缓冲加载→影子验证 (Eval-w/o-Top-p, 一致性) → 小流量金丝雀→全量→写入 ledger；失败自动**原子回滚**。
- **版本逻辑统一**：SemVer + 逻辑依赖图 (DAG)，支持**并行版本与灰度群组**；合规tests为**硬闸**。

8. 最小可执行流程（伪代码）

```
# 1) 采样指标 → 生成MDQ候选
v = estimate_value_preferences(logs, kpi)
# 边际价值
Delta = quantize(v, beta, eta, comm_penalty)
# 见§5
Delta = clip_to_admissible(Delta, policy_bounds)

# 2) 汇编为MDQ-pkg（可回滚）
pkg = build_mdq_pkg(Delta, scope, tests, rollback)

# 3) 影子验证与热插拔
if run_shadow_eval(pkg) and pass_tests(pkg):
    mount_pkg_double_buffer(pkg)
    # 双缓冲
    start_canary_traffic(pkg, 10%)
    # 金丝雀
    if pass_kpi(pkg): promote_to_full(pkg); ledger.append(pkg)
    else: rollback(pkg)
else:
    reject(pkg)

# 4) 索引泛函线性积累
I_next = I_current  $\oplus$  sum(Delta_idx * phi)
persist(I_next); cache_update_async()
```

9. KPI与SLA（上线门槛）

- **质量**：术语覆盖/要点召回 \uparrow ； `word_noncompliance` $\downarrow \geq 30$ ； Eval-w/o-Top-p 不劣化。
- **稳定**：收敛步数 $\downarrow \geq 15\%$ ，训练方差 $\downarrow \geq 20\%$ ；回滚成功率 = 100%。
- **产线**：P95延迟、QPS、显存/内存占用在SLA内；索引重建 \leq 秒级（增量）。
- **审计**：事件日志/MDQ-ledger 可全量回放；每个MDQ-pkg可定位“谁、何时、为何”。

10. 风险与对策

- **长词偏置/投机**：Lp上限+长度成本+IDF/二字降权+语义门控。

- **分布偏移**：训练禁Top-p；上线前Eval-w/o-Top-p校准。
 - **索引污染**：MDQ-pkg必须可逆；黑/白名单tests前置。
 - **性能波动**：双缓冲/金丝雀/自动回滚；EKB热度淘汰。
-

11. 一句话总结

把“微分动力量子（MDQ）”确立为离散化LLM的 **最小控制单元**：用**线性积累**更新索引泛函，用**热插拔**做在线治理，用**统一版本逻辑**保证跨算子×索引×合规的一致性。训练只做**微分与索引**，其余交给**非神经规则与内存索引**，由MDQ驱动“稳、快、省”的持续演进。

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。