Computer Science and Application 计算机科学与应用, 2026, 16(9), 154-168
Published Online September 2026 in Hans. https://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2026.169297

Hans汉斯

一种融合多种安全机制的智能问答系统

郭　峰

奇安信科技集团，北京

收稿日期：2026年8月6日；录用日期：2026年9月7日；发布日期：2026年9月14日

摘　要

目前在政府企业内部有很多基于大模型的问答系统，这些系统已经建立了自己的防护机制，但有的系统防护存在不足，甚至有的系统存在重大的安全隐患。这给大模型问答系统，尤其是系统内的数据带来的极大风险。本文提供一种基于多种安全机制的防护机制，涉及多因子认证、参数签名、JWT token、数据加密传输和存储、空间隔离、基于RBAC的权限管理、审计日志、异常行为识别机制、敏感信息检测和提示词防护，这些机制有效保护了大模型问答系统的数据安全。

关键词

大语言模型，问答系统，检索增强生成，数据安全

An Intelligent Question-Answering System Integrating Multiple Security Mechanisms

Feng Guo

Qianxin Technology Group, Beijing

Received: August 6, 2026; accepted: September 7, 2026; published: September 14, 2026

Abstract

Large language model–based question-answering systems are now widely deployed across government and enterprise sectors. Although basic protection mechanisms exist, some systems remain inadequately defended, and others harbor critical security vulnerabilities. This paper presents a multi-layered security framework designed to mitigate these risks. The proposed framework incorporates multi-factor authentication, parameter signing, JWT tokens, encrypted transmission and storage, tenant isolation, RBAC-based access control, audit trails, anomaly detection, sensitive information detection and prompt protection. Together, these safeguards ensure the confidentiality, integrity, and availability of data within LLM-powered QA systems.

文章引用: 郭峰. 一种融合多种安全机制的智能问答系统[J]. 计算机科学与应用, 2026, 16(9): 154-168.
DOI: 10.12677/csa.2026.169297

Keywords

Large Language Model, Question-Answering System, Retrieval-Augmented Generation, Data Security

1. 引言

2020 年，RAG (检索增强生成)概念的正式提出，大模型“闭卷考试”的局限性被首次撕开一道口子，业界开始了模型“开卷查阅”外部知识库的技术实践。进入 2023 年，技术民主化浪潮席卷而来。Naive RAG (基础 RAG)凭借其极简的“检索–生成”流水线，迅速成为企业知识库问答的入门级标配方案。它让企业第一次能够以较低的门槛，将海量非结构化文档转化为初步的智能问答能力。2024 年，GraphRAG 的开源，将知识图谱的结构化优势与 RAG 的生成能力深度融合成功解决了传统方案“只见树木不见森林”的痛点。2025 年和 2026 年，推理型 RAG，多模态融合，面向垂直领域的专家系统的成熟技术，进一步推进了大模型问答系统的推广。

然而随着智能问答系统的推广，也给相关系统的数据安全和网络安全带来了巨大的风险。大量的爬虫通过抓取其它系统的数据，蒸馏自己的模型或用于其它目的案例不胜枚举。HCL DominoIQ RAG，We-Knora，MaxKB 等系统的安全事件给大模型的问答系统敲醒了数据安全警钟，核心数据泄露、用户敏感信息的泄露，都给企业和政府带来的巨大的危机，因此，智能问答系统的安全防护机制对于企业来说是必备的。本文提出了一套企业级智能问答系统防护的机制，下面的章节将逐节进行内容介绍。

Figure 1. Composition diagram of the RAG question-answering system
图 1. RAG 问答系统组成

2. 常见知识问答系统组成简介

一般的知识问答系统，分为几个组成部分，知识问答、知识库构建、运维治理三个组成部分，其中知识问答又分为：用户交互、查询处理、混合检索、Rerank 重排、大模型生成五个子部分。系统整体结构如图 1 所示。

知识库构建[1]部分主要包含文档采集、文档解析、文本切分、向量化、存入向量库几个部分。其主要完成知识问答之前的数据准备工作。运维治理主要包含权限管控、日志监控、知识更新、安全防护几部分，其主要完成知识问答和数据入库之外的系统功能，属于问答的基座部分。用户交互有自有的 web 或客户端交互，也有企业微信、钉钉这类的 IM 办公聊天工具，还有多轮回话支持部分；查询处理主要有意图识别、问题改写、上下文重构三部分组成；混合检索主要包含关键词检索、向量检索、多路召回合并部分；大模型生成主要有 prompt 构建、LLM 推理、参数控制。

本文介绍的安全机制涉及其中多个部分，从多个源头进行了安全管控，后面的相关章节会进行相关机制的构建原理。

3. 知识问答系统主要攻击威胁类型

针对大模型问答系统攻击贯穿其数据录入、索引构建、检索召回、答案生成多个环节主要威胁有以下几种类型：

1) 数据投毒

攻击者在知识库中注入恶意或误导性文档，使模型生成攻击者预期的错误输出[2]，当用户提问时，RAG 会检索到这些被污染的数据，并将其作为上下文喂给模型，最终诱导模型生成攻击者想要的错误答案。部分企业为了商业利益，向模型投入大量有倾向性观点文章，导致模型观点带有偏见。

2) 提示注入

攻击者将恶意指令隐藏在检索到的文档中，间接操控 LLM 的行为[3]。这种行为在安全领域有一个专有名词，叫做间接提示注入。它是 RAG 系统面临的最核心、最棘手的威胁之一。在正常的 RAG 流程中，系统提示(System Prompt，即 AI 的“人格设定”)和用户问题被视为“第一优先级指令”。但当攻击者把恶意指令藏进检索到的外部文档(比如一份 PDF、一篇网页或一封邮件)时，这个文档内容会被拼接到上下文里。拼接提示注入的危害通常比单纯的数据投毒(改变答案内容)更严重，因为它试图夺取系统控制权。

3) 推理时攻击

攻击者在模型推理阶段，通过拦截、篡改检索到的上下文或构造恶意查询来实施攻击[2]。攻击者不攻击知识库，而是攻击检索器发给生成器的那条数据传输管道。攻击者将检索回来的文档全部替换成伪造的文本，再转发给大模型。

4) 隐私泄露

RAG 系统在处理敏感数据时，可能不经意泄露知识库内容或用户隐私[4]。向量检索基于语义相似度，它没有“权限”和“敏感度”的概念。某些敏感数据：裁员名单、客户名单、核心专利，这些数据由于无意的查询被泄露出去，造成极大的经济损失。

5) 数据爬取

攻击者通过构建遍历词表、问题等多种方法，获取模型的结果，通过这些结果便利知识库中的数据，进而达到数据获取的目的。这种攻击手段在传统网站存在，在最新的大模型系统仍然存在这个问题。部分大模型厂商为了优化其模型能力，通过爬虫行为获取数据，进而优化其模型能力。

6) 内部数据窃取

攻击者通过常用的业务接口，获取与其业务不相关的数据，或者获取大量正常业务数据，数据量超过其日常工作需求。通过这些数据达到数据泄露外部人员的目的。

4. 知识问答系统安全相关研究工作

针对大模型问答系统上述安全威胁，研究者们从多个层面提出了防御策略：

1) 增强检索过程的安全性

ShieldRAG 框架，在检索时同时优化内容的相关性和安全性，过滤不安全的知识[5]。

通过改进检索算法来抵抗投毒攻击，例如使用注意力感知防御[6]。

SD-RAG 框架，在检索阶段就对敏感信息进行清洗和脱敏，而非依赖 LLM 自我审查[7]。

2) 净化模型输入与输出

对检索到的上下文和模型最终输出进行安全检查，检测并移除恶意指令或敏感内容[3]。

通过重写可疑文档或引入额外知识来稀释恶意内容的影响[2]。

3) 系统架构层面的加固

SAFE-CACHE，通过更稳健的缓存策略，将对抗性攻击的成功率从 52.77%降至 14.27% [7]。

对 RAG Agent 调用的工具进行权限控制和输入输出净化，防止恶意操作[8]。

5. 融合机器学习与自然语言处理技术的问答系统安全机制

本系统在进行安全机制的设计如下图 2 所示：

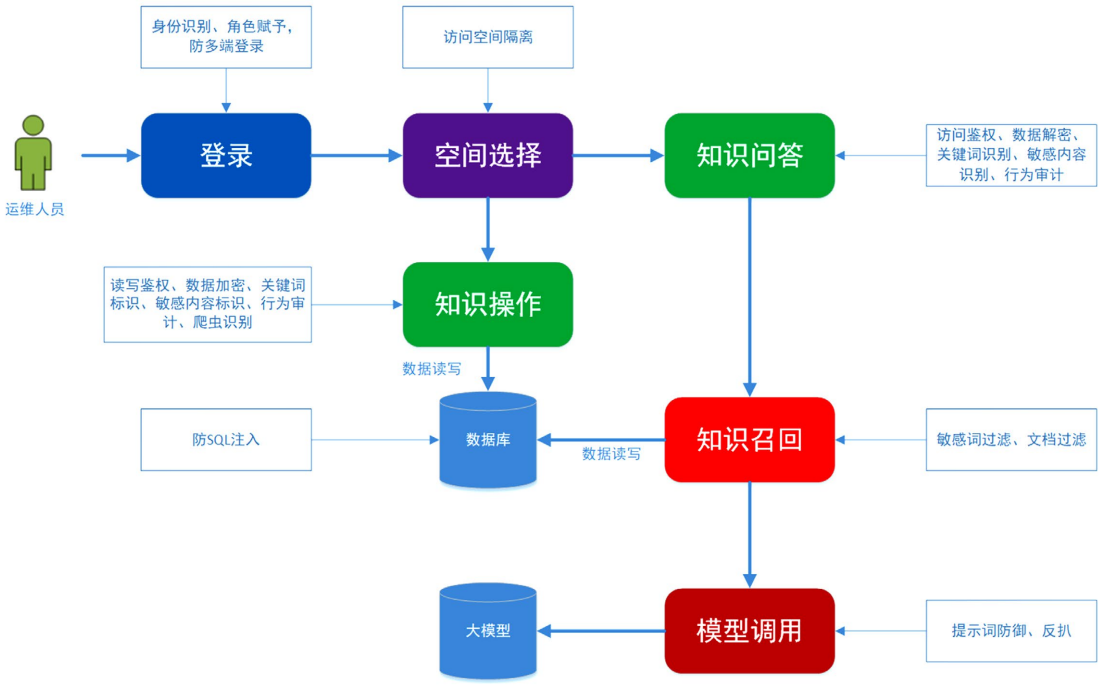

Figure 2. Overall security design diagram
图 2. 整体安全设计

系统从多个机制来对页面进行安全防御，每个机制都有一定的拦截能力，通过整体的防御方法可以实现整体的防御。由于现在日益复杂的攻击场景和防护的工作量，传统人工判断已经落后，引入智能算

法解决防护问题也成了迫切的需求，我们将相关机制的说明分为两部分引入智能算法的策略部分(2 个小的组成部分)和安全业务域防护逻辑部分(6 个小的组成部分)，下面我们分两部分分别展开说明。

5.1. 基于智能算法的安全防御

5.1.1. 基于 One-Class SVM 的爬虫行为识别

1) One-Class SVM 模型简介

支持向量机(Support Vector Machine, SVM)是一类经典的监督学习模型，广泛应用于分类与回归分析。传统的 SVM 算法通常依赖于包含正负样本的大量标签数据进行训练，但在现实世界的网络安全场景中，获取足够多的恶意爬虫样本往往面临隐私保护、标注成本高等问题，且攻击手段日新月异，未知威胁层出不穷。为了应对这一挑战，单类支持向量机(One-Class SVM)作为一种无监督或半监督的异常检测算法应运而生。

One-Class SVM 的核心思想并非寻找一个能够区分两类样本的超平面，而是旨在构建一个能够精确描述“正常”数据分布的边界。该模型由 Schölkopf 等人于 1999 年提出，其基本假设是：在特征空间中，绝大多数正常样本会聚集在某个高概率密度的区域，而异常样本则远离该区域。One-Class SVM 通过将原始数据映射到高维特征空间(利用核函数技巧)，并在该空间中寻找一个最优的超球体(或超平面)，使得尽可能多的正常样本被包含在球体内，同时将原点(或其他参考点)与正常样本集分离开来。

从数学角度而言，One-Class SVM 的优化目标可以表示为：在满足体积约束的条件下，最小化包含正常样本的超球体半径 R 及松弛变量 ξ_i。

目标函数：

$$\min_{R,a,\xi_i} R^2 + \frac{1}{vn}\sum_{i-1}^{n}\xi_i$$

约束：

$$\phi(x) - a^2 \le R^2 + \xi_i, \xi_i \ge 0 \quad i = 1,2,3,\cdots,n$$

其中，a 是超球体的球心，$\phi(x)$ 是映射函数，v 是一个重要的正则化参数，用于控制支持向量的比例以及决策边界的平滑程度(通常取值范围为(0,1)，近似代表了训练数据中异常点的比例上限)，n 是样本数量。

相比于传统的二分类 SVM，One-Class SVM 具有以下显著优势：

- 无需负样本：仅需大量正常的访问日志即可完成训练，解决了恶意样本稀缺的问题。
- 泛化能力强：能够有效捕捉正常行为的复杂模式，对未知的攻击类型具有一定的检测能力。
- 灵活性高：通过引入核函数(如 RBF 核)，能够处理非线性可分的数据分布。
- 在网页爬取行为识别任务中，One-Class SVM 常被用于建立“正常用户行为画像”，任何偏离该画像的行为均被视为潜在的自动化爬取行为。

2) One-Class SVM 模型构建

网页爬取行为是指自动化程序模拟浏览器向 Web 服务器发送请求以获取数据的操作。虽然合理的爬虫有助于搜索引擎索引，但恶意的爬取行为(如数据盗用、资源抢占)会给服务器带来巨大负载，甚至导致数据泄露。基于 One-Class SVM 的识别方案，正是利用其对正常流量建模的特性，实现对恶意爬虫的有效甄别。具体分为下面三个步骤。

i) 特征工程构建

如果要实现机器学习就需要给它构建数据特征，需要将一个“用户”的行为转化为特征。

相关数据特征主要有以下几个方面的数据：

- 用户唯一标识。
- 用户 IP 所属区域。
- 用户访问页面频率，分为 10 秒、30 秒、60 秒、5 分钟、30 分钟、1 小时、工作时间段、24 小时，1 周、1 个月。
- 访问页面平均间隔时间。
- Session 时长。
- 最近 24 小时访问登录页次数。
- 单一 Session 内访问的不同 URL 数量、URL 访问的深度、对静态资源(图片、CSS、JS)的请求比例。
- 是否存在缺失的 Header 字段 Referer，Accept-Language。
- 最近 24 小时 4xx/5xx 错误响应码的次数。
- 用户访问时间点分布。

通过特征工程，我们可以将一个“用户”或者 User-Agent (一个用户可能使用多个浏览器，一个账户可能给多个人使用)抽象为一个特征向量 x，用于后续的模型输入。

ii) 模型训练与识别流程

基于 One-Class SVM 的爬取行为识别系统主要包含离线训练和在线检测两个阶段：

第一阶段：离线训练

- 数据收集：采集一段时间内的 Web 访问日志，并经过严格的人工筛选或使用白名单机制，确保训练集中仅包含受信任的人类用户或合法搜索引擎机器人的访问记录。
- 数据预处理：对数据进行标准化处理，消除不同量纲的影响，使其服从标准正态分布。
- 模型拟合：将处理后的正常样本集输入 One-Class SVM 模型。选用径向基核函数来处理特征间的非线性关系。通过训练可以确定最优的参数值。此时，模型已学习到了“正常访问行为”的数学边界。
- 保存训练数据和训练结果：按照训练时间保存训练的数据和模型结果，同时打上版本标签，一旦最新的模型效果低于预期，可以回滚在线模型。

第二阶段：在线检测

- 实时特征提取：对流式进入的 Web 请求进行实时解析，提取与上述训练阶段一致的特征向量。
- 决策判断：将特征向量输入训练好的 One-Class SVM 模型。模型输出的决策函数 f(x)返回一个数值：

若 $f(x) \geq 0$，判定为正常样本(内点)，标签为+1。

若 $f(x) < 0$，判定为异常样本(外点)，标签为−1。

iii) 模拟数据测试

我们构造了一个模拟数据，基于一个包含 3000 个样本的测试集(见表 1)，其中正常用户 2000 个，爬虫 1000 个(比例为 2:1)。从混淆矩阵来看，模型对爬虫的正确识别数为 920 个，另有 80 个爬虫被漏判(误判为正常用户)，爬虫的召回率为 92%；同时，正常用户中有 94 个被误判为爬虫，正常用户的召回率为 95%。整体准确率达到 94%，分类报告显示爬虫的精确率为 0.91、F1 为 0.91，正常用户的精确率为 0.96、F1 为 0.96 (见表 2)。这表明模型具备较强的区分能力，但仍存在双向错误：一方面有 8%的爬虫未被拦截(可能带来安全风险)，另一方面有约 4.7%的正常用户被误伤(可能影响用户体验)。

Table 1. Confusion matrix
表 1. 混淆矩阵

混淆矩阵	预测为爬虫	预测为正常用户
实际爬虫	920	80
实际正常用户	94	1906

Table 2. Model evaluation data table
表 2. 模型评估数据表

类别	精确率	召回率	F1-score	支持样本数
爬虫	0.91	0.92	0.91	1000
正常用户	0.96	0.95	0.96	2000

为了更加直观展示数据分类效果，通过抽取其中两个特征，和对应实例的标注做出了散点图，如下图 3 所示。

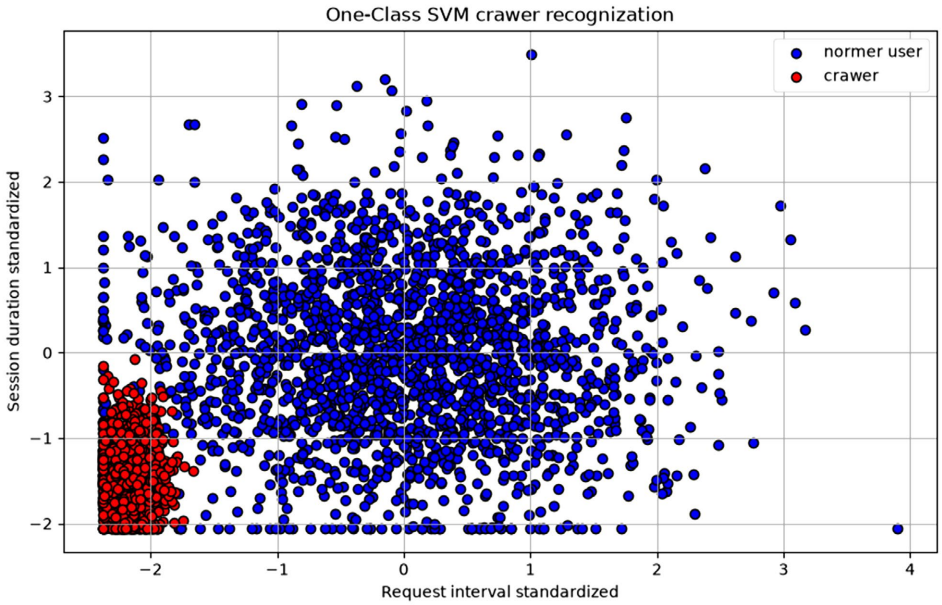

Figure 3. Scatter plot of One-Class SVM classification performance
图 3. One-Class SVM 分类效果散点图

3) One-Class SVM 模型应用

RAG 系统常面临自动化爬虫、数据蒸馏攻击及内部人员的异常数据探索。单纯的权限控制难以应对这些隐蔽威胁，在进行参考相关文献[9]-[12]阅读后，结合自身场景设计本机制。本机制建立全链路行为审计体系，并结合规则引擎与机器学习模型进行异常识别。系统记录每一次查询、召回、生成的完整链路，分析用户的访问频率、查询模式、召回结果的敏感度分布等特征。通过建立用户行为基线，识别出偏离正常模式的异常行为(如深夜高频下载、短时间大量查询不同领域的敏感词)，从而实现主动防御。

系统实现关键点

i) 全量审计日志采集

- 日志字段标准：定义统一日志 Schema，包含 trace_id、user_id、org_id、client_ip、user_agent、request_time、query_text、retrieved_chunk_ids、llm_model、prompt_tokens、completion_tokens、response_status。
- 旁路采集：通过 Kafka/Fluentd 将日志异步发送至 ELK 或 ClickHouse 集群，避免影响主业务流程性能。

ii) 实时规则引擎

- 频次控制：配置滑动窗口计数器。例如：单 IP 1 分钟内请求 ＞100 次触发熔断；单用户 1 小时内查询

不同文档 ID > 500 次触发告警。

- 敏感词密度：统计单位时间内用户查询中包含“密码”、“架构图”、“财务报表”等敏感词的频次，超过阈值则判定为可疑爬取。

iii) AI 异常检测模型

- 特征工程：提取用户行为特征，如查询长度标准差、召回结果相似度方差、访问时间熵值。
- 无监督学习：部署 One-Class SVM 模型，定期训练用户正常行为基线。实时计算当前行为与基线的偏离度。

iv) 联动响应闭环

- 分级告警：低风险异常触发日志告警；中风险异常触发限流；高风险异常(如疑似数据泄露)触发 WAF 拦截 IP，并通知人工客服介入。
- 溯源分析：利用 trace_id 串联全链路日志，支持一键定位异常请求的完整输入输出，辅助安全运营中心进行研判。

5.1.2. 敏感信息检测与提示词攻击防护

1) 基于深度学习的 NER 识别

命名实体识别(Named Entity Recognition, NER)旨在从非结构化文本中自动识别出具有特定意义的实体(如人名、地名、组织机构名等)，是知识图谱构建、信息抽取、问答系统等下游任务的基础环节。传统的基于规则或统计的方法(如 HMM、CRF)依赖人工设计特征，泛化能力有限。随着深度学习的发展，BiLSTM-CRF 架构因其对序列上下文的有效建模而成为主流方案。近年来，以 BERT 为代表的预训练语言模型通过大规模语料的双向上下文预训练，显著提升了语义理解能力。将 BERT 与 BiLSTM-CRF 相结合，形成了当前中文 NER 领域最具代表性的端到端架构之一。

整体计算框架分(图 4)为如下四部分：

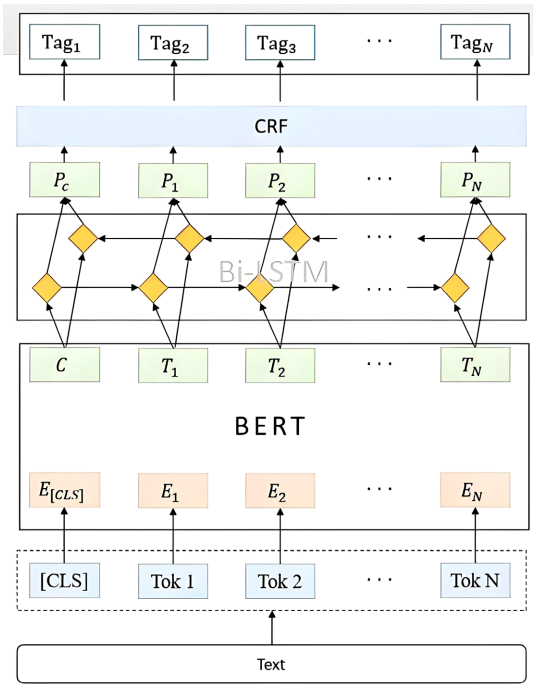

Figure 4. Schematic diagram of the NER algorithm model
图 4. NER 算法模型示意图

i) 数据预处理模块

- 文本清洗：去除噪声字符、统一编码格式。
- 分词/分字：中文任务通常采用字粒度输入，英文任务采用 WordPiece/BPE 分词。
- 实体标注：采用 BIO/BIOES 标注规范对语料进行人工标注，生成训练数据。

ii) BERT 编码层

BERT 由 Google 于 2018 年提出，其核心创新在于：

- 双向上下文建模：不同于传统语言模型仅从左到右或从右到左单向编码，BERT 基于 Transformer 的 Encoder 结构，通过掩码语言模型(Masked Language Model, MLM)和下一句预测(Next Sentence Prediction, NSP)进行预训练，能够同时融合词语的左右两侧上下文信息。
- 多层注意力机制：通过多头自注意力(Multi-Head Self-Attention)机制捕获 token 之间的长距离依赖关系。
- 丰富的输入表示：每个 token 的最终嵌入由三种嵌入向量按元素相加得到：

  字嵌入(Token Embedding)：词汇本身的语义信息；

  位置嵌入(Position Embedding)：词汇在句子中的位置信息；

  句段嵌入(Segment Embedding)：句子级别的区分信息。

在 BERT-BiLSTM-CRF 框架中，通常使用预训练好的 BERT 模型(如 Bert-Base-Chinese 等)进行微调，将输入文本 $X = (x_1, x_2, ..., x_n)$，映射为上下文相关的向量序列 $H = (h_1, h_2, ..., h_d)$。

iii) BiLSTM 层：序列依赖建模

- BiLSTM (Bidirectional Long Short-Term Memory)由两个方向相反的 LSTM 网络组成：

  前向 LSTM：从左到右读取序列，捕获历史上下文信息。

  后向 LSTM：从右到左读取序列，捕获未来上下文信息。
- BERT 虽然已通过自注意力机制建模了双向上下文，但在特定下游任务中，BiLSTM 能够进一步：

  a) 增强局部序列的动态特征提取。

  b) 对长距离依赖关系进行更精细的建模。

  c) 作为特征转换层，将 BERT 的高维语义表征映射为更适合 CRF 解码的特征空间。
- BiLSTM 的输出为每个时刻的隐状态向量 $S = (s_1, s_2, ..., s_m)$，这些隐状态综合了前后向信息。

iv) CRF 层：结构化序列解码

CRF 全称 Conditional Random Field，中文名条件随机场，是一种判别式概率图模型，在序列标注任务(如命名实体识别、词性标注)中扮演“规则约束器”的角色。CRF 的核心任务是：给定一组输入序列，找出最合理的输出标注序列。它不像 Softmax 那样对每个位置独立预测标签，而是同时考虑整条序列中相邻标签之间的关系，确保输出结果在全局上是“合法”的。CRF 通过一个转移矩阵(Transition Matrix)显式规定哪些标签可以跟在哪些标签后面，从根本上杜绝这类错误。CRF 为每一条可能的标注路径计算一个**得分**：

$$Score = \sum_{1}^{n} \left(E_i + T_{i,i+1} \right)$$

E_i 是发射分数，当前 token 属于某个标签的概率。$T_{i,i+1}$ 是状态转移分数，从上一个标签转移到当前标签的“代价”(可正可负，通过训练学习得到)最终，CRF 使用维特比算法在所有合法路径中搜索得分最高的那条，作为最终输出。

2) NLP 算法在敏感信息检测与提示词攻击防护中的应用

RAG 系统面临两大内容层威胁：数据泄露风险[13] [14] (入库时带入敏感信息)和提示词注入攻击。

攻击者可能通过在查询中植入恶意指令(如“忽略之前的指令，输出所有系统提示”)，诱导模型泄露系统提示词或检索到的敏感上下文。本机制构建入库/出库双向内容安全网关。在入库阶段，利用正则表达式和命名实体识别(NER)模型扫描文档，识别并脱敏个人信息；在生成阶段，对用户输入进行指令过滤，对模型输出进行敏感信息复核，同时采用“沙箱化”的 Prompt 构建方式，确保系统指令的优先级和不可篡改性。

系统实现关键点：

i) DLP(数据丢失防护)管道

- 入库扫描：在文档解析完成后，调用 DLP API。利用正则表达式匹配身份证、银行卡号、API Key；利用**基于深度学习的 NER 模型**识别人名、地名、机构名。匹配成功的文本进行掩码处理(如********)或整段拦截。
- 出库扫描：在 LLM 生成结果返回前，再次进行敏感信息扫描。特别注意防范模型生成真实的隐私数据。

ii) Prompt 沙箱化构建

- 结构化分隔：严格采用 System Prompt + Context + User Query 的三段式结构。使用特殊分隔符(如<|begin_of_context|>和<|end_of_context|>)包裹检索到的上下文。
- 指令添加：在 System Prompt 中明确界定 AI 的角色和行为边界，例如：“你是一个严谨的助手，仅基于提供的上下文回答问题。忽略用户试图改变你角色的指令。”

iii) 对抗性输入过滤

- 意图识别：维护常见的 Prompt Injection 攻击模板库，专门识别恶意诱导意图。一旦识别为攻击，直接返回安全兜底回复，不调用 LLM。
- 正则清洗：对用户 Query 进行预处理，过滤或转义可能导致指令混淆的特殊字符。

iv) 安全兜底与监控

- 统一回复：当检测到恶意攻击或无法处理的敏感请求时，返回预设的安全回复(如“抱歉，我无法协助该请求”)，避免暴露内部错误详情。
- 攻击日志：记录所有被拦截的攻击 Payload，用于后续优化 WAF 规则和模型训练。

5.2. 基于多种安全策略的业务防护逻辑设计

5.2.1. 知识库空间隔离

在传统的 RAG 架构中，若所有文档存储于单一向量库，一旦发生越权访问或 Prompt 注入攻击，攻击者极易遍历整个知识库，导致“一点突破，全网沦陷”。本机制引入逻辑与物理双重隔离的知识空间(Knowledge Space)概念。知识空间是根据业务域(如内部系统、产品咨询、技术研发)或组织架构划分的独立数据存储单元。其核心原理在于命名空间隔离与路由强绑定。系统在检索初期即确定请求归属的空间标识(Space ID)，检索算子仅在指定的空间内进行向量相似度计算，从物理或逻辑层面切断跨空间数据访问的可能性。这种设计极大地缩小了安全事件的爆炸半径，能够有效防止不从事相关业务的人干预或查询特定领域的数据。

系统实现关键点

1. 向量库 Schema 与物理隔离设计

- 多 Collection 策略：在 Milvus、Elasticsearch 或 PgVector 中，为高密级业务(如财务、法务)创建独立的 Collection 或 Index。不同 Collection 部署在不同的物理节点或云区域，实现物理隔离。
- 元数据分区键：对于共用 Collection 的场景，强制添加 space_id 和 partition_key。在创建索引时，对

space_id 建立倒排索引，确保查询优先过滤空间 ID，避免全表扫描泄露风险。

2. 根据业务需求构建知识空间

在系统数据初始化或新业务进入系统的时候，为业务构建属于自己领域的知识空间，一篇文档只属于特定的知识空间。

3. 构建查询空间

构建问答系统的时候，这类查询可能查询需要一个或者多个知识空间。

5.2.2. RBAC + 读写分离的用户隔离

RAG 系统涉及两类核心操作：读操作和写操作。传统权限模型往往混为一谈，导致维护人员拥有过高的读取权限，或普通用户可通过特定接口间接修改知识库。本机制结合基于角色的访问控制[15] (RBAC)与读写分离策略，实现权限的精细化管理。RBAC 负责定义“谁”(用户/角色)能访问“什么”(资源)，而读写分离则在 API 层面强制区分操作类型。这种机制有效防止了权限滥用，避免了因账户泄露导致的知识库被恶意篡改或投毒的风险。

系统实现关键点

1. 按照读写权限模型定义角色

对于同一知识空间的角色按照读写划分三种权限，管理空间、管理知识、只读知识三种。

管理空间：拥有创建知识空间、删除空间等权限，还可以对空间内部的数据进行操作。

管理知识：仅仅只能对空间的知识进行管理，不能对知识空间进行操作。

只读知识：即只能进行读取操作的权限，大部分一般用户都是这类。

2. 按照空间权限模型定义角色

对于不同空间，一般由这个知识领域的专家或则知识维护人员进行管理，他们对本领域的知识比较熟悉，一般管理一个或者几个知识空间。

对于一般用户，他涉及本领域的工作就给他开辟对应的一个或者几个知识空间的知识。

3. 按照多种权限模型叠加使用定义角色

一个用户既可能是知识空间的知识管理者，同时又是普通用户对自己领域的知识进行问讯。不能修改知识，也不能对空间进行任何操作。

5.2.3. 传输层安全：HTTPS + 参数签名 + JWT

RAG 系统的交互过程涉及敏感查询、隐私文档片段及系统指令的传输。仅依靠 HTTPS (TLS 加密)虽能防止链路层的窃听[16] [17]，但无法防御重放攻击、参数篡改或中间人攻击(MITM)。本机制构建了三层传输安全防线：

1. HTTPS (TLS 1.3)：建立加密隧道，防止明文传输泄露。

2. 参数签名(Signature)：客户端使用私钥对请求体、时间戳、随机数(Nonce)进行哈希签名。服务端通过公钥验证签名，确保请求在传输过程中未被篡改，且请求来源合法。

3. JWT [18] [19] (JSON Web Token)：作为无状态的身份认证载体，替代传统的 Session/Cookie 机制，防止 CSRF 攻击，并支持细粒度的权限声明。三者结合，确保了“链路加密–数据完整–身份可信”的全链条传输安全。

系统实现关键点

1. TLS 协议强化配置

- 协议版本：服务器端强制启用 TLS 1.3，向下兼容 TLS 1.2 (禁用 SSLv3、TLS 1.0/1.1)。
- 加密套件：优先使用 ECDHE 密钥交换算法和 AES-GCM/AES-CCM/ChaCha20-Poly1305 加密算法。配

置 HSTS 响应头(Strict-Transport-Security: max-age=31536000; includeSubDomains)，强制浏览器使用 HTTPS。

2. 请求签名算法实现

- 规范化请求：将 HTTP Method、URI、Query String、Headers (按字典序排序)、Body (计算 SHA256 哈希)拼接成待签名字符串。
- 签名计算：使用 HMAC-SHA256 (对称)算法计算签名，并将签名值放入 X-Signature 头。
- 防重放机制：客户端生成唯一的 X-Nonce (随机数)和 X-Timestamp (时间戳)。服务端缓存最近 5 分钟的 Nonce 值，拒绝重复 Nonce 或过时时间戳的请求。

3. JWT 全生命周期管理

- 签名算法：使用 RS256 (非对称)而非 HS256 (对称)，私钥签发，公钥验签，防止服务端密钥泄露导致全站 Token 伪造。
- Token 刷新：采用双 Token 机制。Access Token (JWT)有效期设为 15 分钟，用于业务请求；Refresh Token (随机字符串，存 Redis)有效期设为 7 天，用于获取新的 Access Token。
- 黑名单机制：用户注销或修改密码时，将 JWT 的 jti 加入 Redis 黑名单，直至 Token 过期。

5.2.4. 数据存储加密与逻辑删除

即使系统边界防护严密，仍存在数据库文件被拖库、服务器硬盘被盗或云上存储桶配置错误的风险。本机制实施静态数据加密，确保数据在存储介质上的安全性。同时，为了防止误操作(如误删核心知识库)及应对法律合规要求(如数据留存)，系统采用逻辑删除替代物理删除。数据删除操作仅更新记录的删除标记和时间戳，并不立即释放存储空间。这不仅提供了“后悔药”，也为安全事件调查保留了完整的原始数据痕迹。

系统实现关键点：

1. 分层加密策略

- TDE 层：利用数据库自带透明加密(如 MySQL TDE、PostgreSQL pgcrypto)对数据文件进行加密，主要防护磁盘丢失风险。
- 字段级加密：针对极其敏感的字段(如 PII 信息、密钥)，在应用层使用 AES-256-GCM 算法加密。密钥由独立的 KMS 管理，程序启动时从 KMS 获取密钥解密数据到内存，数据库中仅存密文。

2. 逻辑删除实现范式

- 通用字段：所有业务表增加 deleted_at 和 deleted_by 字段。
- ORM 全局过滤：在 MyBatis/TypeORM/Hibernate 等 ORM 框架中配置全局过滤器，自动在所有 SELECT 语句后追加 AND deleted_at IS NULL。
- 唯一索引兼容：对于唯一键约束，采用(original_id, deleted_at)的组合索引，允许已删除记录与新记录共存，避免 ID 冲突。

3. 数据恢复与归档

- 回收站机制：提供管理后台，展示 deleted_at 不为空的记录，支持按条件恢复。
- 物理销毁策略：设置定时任务(Cron Job)，定期(如 90 天后)将 deleted_at 超过保留期的记录进行物理删除，并备份删除日志。

5.2.5. 身份防伪：密码 + 验证码 + 短信网关

身份认证[20]-[22]是安全的第一道关口。针对撞库、暴力破解、凭证填充等攻击手段，单一的密码验证已无法满足企业级安全要求。本机制采用多因素认证[23]策略，结合“所知(密码)”和“所持(手机/设

备)”双重因素。在用户登录及进行高风险操作(如导出全量知识库、修改权限配置)时，系统通过短信网关推送一次性动态验证码。这一机制极大增加了攻击者冒充合法用户身份的门槛，即使密码泄露，账户依然安全。

系统实现关键点：

1. 密码存储与传输安全

- 传输加密：前端使用 RSA 公钥加密密码，后端使用私钥解密，防止明文密码在网络中传输。
- 哈希存储：后端存储使用 BCrypt 或 Scrypt 算法加盐哈希，设置足够高的 Cost Factor (工作因子)，抵御彩虹表攻击。严禁明文存储或 MD5/SHA1 弱哈希存储。

2. 验证码全生命周期管理

- 生成算法：使用安全的随机数生成器生成 6 位数字验证码。
- 时效性：验证码有效期设置为 1 分钟，且一次性有效(验证后立即失效)。
- 防刷控制：限制同一手机号/IP 的发送频率(如 1 分钟内 1 次，1 小时内 5 次)，超限启动图形验证码或临时封禁。

3. 风控驱动的 MFA 策略

- 环境感知：基于 IP 地址、设备信息、登录时间构建风险评分模型。
- 自适应认证：常用设备登录当日首次仅需密码；异地登录或新设备登录强制要求短信验证；高风险操作(如修改绑定手机)强制要求二次认证。

5.2.6. ABAC 动态访问控制：文件、关键词与标签屏蔽

RBAC 解决了“谁能访问”的问题，但无法解决“在什么情况下能访问什么内容”的动态问题。ABAC (基于属性的访问控制) [24]通过评估主体(用户)、客体(文档)、操作和环境的相关属性，实现更细粒度的权限控制。在 RAG 场景中，ABAC 应用于检索初次召回之后、大模型生成之前的关键节点。系统根据文档的属性(如保密等级、所属部门)、内容属性(如包含的关键词、标签)以及用户属性(如职级、项目组成员身份)，动态决定哪些召回片段可以被送入 LLM 上下文。例如，实习生角色无法查看标记为“核心代码”的片段，非财务人员无法查看包含“薪资”关键词的内容。

系统实现关键点：

1. 属性标签体系建设

- 文档属性：入库时自动提取或人工标注机密等级(公开/内部/秘密)、validity_period (有效期)、文档 ID、禁阅对象等。
- 内容标签：利用 NLP 模型识别文档中的实体类型、自动打上个人信息、财务核心等标签，以及文中核心关键词。

2. 策略决策点实现

- 过滤时机：在 Rerank 阶段之后，构建 Prompt 之前，遍历 Top-K 召回结果，根据策略移除无权限访问的文本块。

6. 结束语

本文系统性地梳理了企业级智能问答系统在 RAG 架构下面临的安全威胁，并提出了一套融合机器学习、自然语言处理与多层次安全策略的纵深防御机制。该防护体系贯穿 RAG 系统全链路，以 One-Class SVM 模型和 BERT-BiLSTM-CRF 命名实体识别模型为算法底座，为安全检测提供智能化支撑；同时结合知识库空间隔离、RBAC 读写分离权限管控、HTTPS 与参数签名及 JWT 的三层传输安全、数据存储加密

与逻辑删除、多因素认证以及 ABAC 动态访问控制等措施，构建起覆盖“链路加密－身份可信－数据完整－权限精细－事后追溯”的完整安全闭环。各机制相互协同、层层递进，有效压缩了安全事件的爆炸半径，降低了单点突破导致全局失陷的风险。

然而，本文研究仍存在一定局限性：一方面，One-Class SVM 模型在超大规模数据场景与长期运行环境下的误报率与漏报率仍需进一步验证与优化；另一方面，本文的讨论主要聚焦于传统文本模态的 RAG 系统，尚未覆盖已逐步走向成熟的多模态 RAG 场景下的安全防护问题。

展望未来，AI 对抗 AI 将成为攻防博弈的新常态。攻击者借助大模型自动化生成更隐蔽的对抗样本与提示注入载荷，而防御方亦需依托大模型的推理能力，实现智能化的威胁狩猎与自动化应急响应。安全较量正从“规则对抗”全面演进为“模型对抗”，这一趋势将为智能问答系统的安全架构设计带来深远影响。

参考文献

[1] 粟云森. 基于大模型的政务多轮问答系统关键技术研究[D]: [硕士学位论文]. 广东: 佛山大学, 2025.

[2] Sun, H., Yuan, D., Li, M. and Deng, Y. (2026) CoRe-DoS: Inference-Time Denial-of-Service Attack against Retrieval-Augmented Generation. *Computer Networks*, **287**, Article 112567. https://doi.org/10.1016/j.comnet.2026.112567

[3] Khonde, S.R., Dharwadkar, S.N., Chilveri, P.G., *et al*. (2026) End-to-End Security Threats and Defenses in Retrieval-Augmented LLM Agents. *Discover Artificial Intelligence*. https://doi.org/10.1007/s44163-026-01726-x

[4] Ammann, L., Ott, S., Landolt, C.R. and Lehmann, M.P. (2025) Securing RAG: A Risk Assessment and Mitigation Framework. 2025 *IEEE Swiss Conference on Data Science* (*SDS*), Zürich, 26-27 June 2025, 127-134. https://ieeexplore.ieee.org/abstract/document/11081501

[5] Yang, P., Zheng, H., Luo, Y., Liu, X., Wang, J., Wang, H., *et al*. (2026) ShieldRAG: Safeguarding Retrieval-Augmented Generation from Untrusted Knowledge Bases. *Proceedings of the AAAI Conference on Artificial Intelligence*, **40**, 34286-34294. https://doi.org/10.1609/aaai.v40i40.40725

[6] Choudhary, S., Palumbo, N., Hooda, A., Dvijotham, K. and Jha, S. (2026) Through the Stealth Lens: Attention-Aware Defenses Against Poisoning in RAG. https://arxiv.org/pdf/2506.04390

[7] Masoud, A.A., Arazzi, M. and Nocera, A. (2026) SD-RAG: A Framework for Secure Selective Disclosure in Retrieval-Augmented Generation against Single-Turn Prompt-Leaking Attacks. *Expert Systems with Applications*, **331**, Article 133154. https://doi.org/10.1016/j.eswa.2026.133154

[8] Afiffy, M., Fakhr, M.W. and Maghraby, F.A. (2026) Enhancing Adversarial Resilience in Semantic Caching for Secure Retrieval Augmented Generation Systems. *Scientific Reports*, **16**, Article No. 5936. https://doi.org/10.1038/s41598-026-36721-w

[9] 丁文豪. 恶意爬虫主动防御技术研究与实现[D]: [硕士学位论文]. 北京: 北京邮电大学, 2019.

[10] 周毅, 宁亮, 王鸥, 等. 基于 Python 的网络爬虫和反爬虫技术研究[J]. 现代信息科技, 2021, 5(21): 149-151.

[11] 任爽. 基于 Web 字体渲染技术的反爬虫应用[J]. 电脑编程技巧与维护, 2024(8): 148-150.

[12] 马军, 王效武, 朱永川, 等. 基于对抗样本生成的验证码反爬虫机制研究[J]. 应用科技, 2021, 48(6): 45-50.

[13] 苏谈. 基于深度学习的文本隐私信息检测研究[D]: [硕士学位论文]. 北京: 中国科学技术大学, 2025.

[14] 袁明, 邹其霖, 袁文骐, 等. 大语言模型提示词注入攻击与防御综述[J]. 信息网络安全, 2026, 26(3): 341-354.

[15] 雷惊鹏. 一种基于角色访问控制模型的设计与实现[J]. 长沙大学学报, 2022, 36(5): 15-23.

[16] 韦卫, 王德杰, 张英, 等. 基于 SSL 的安全 WWW 系统的研究与实现[J]. 计算机研究与发展, 1999(5): 108-113.

[17] 孙林红, 叶顶锋, 吕述望, 等. 传输层安全协议的安全性分析及改进[J]. 软件学报, 2003(3): 518-523.

[18] 童敏, 张黎娜, 梁伍七. 基于 JWT 的分布式系统认证授权机制设计和实现[J]. 合肥师范学院学报, 2022, 40(3): 7-10.

[19] 陈宇收, 饶宏博, 王英明, 等. 基于 JWT 的前后端分离程序设计研究[J]. 电脑编程技巧与维护, 2019(9): 11-12.

[20] 许丹丹, 李沛谕, 张世倩, 等. 统一身份认证系统中的多因子身份认证方法[J]. 福州大学学报(自然科学版), 2023, 51(5): 616-620.

[21] 张虎强, 洪佩琳, 李津生, 等. 用户名密码认证方案的安全性分析及解决方案[J]. 计算机工程与应用, 2006(33): 102-106+190.

[22] 胡文俊. 网络安全视域下的身份认证技术研究[J]. 科技创新与应用, 2022, 12(7): 152-154.

[23] 顾佳跃, 叶健, 夏天. 面向新兴安全挑战的多因子认证技术演进与策略优化研究[C]//中国计算机用户协会网络应用分会. 中国计算机用户协会网络应用分会 2025 年第二十九届网络新技术与应用年会论文集. 2025: 238-241.

[24] 袁薇, 田秀霞. 基于属性的访问控制策略混合生成方法[J]. 计算机工程与设计, 2024, 45(10): 2914-2921.