



开放科学
(资源服务)
标识码
(OSID)

面向情报研究的多智能体协同处理方法

白佳欣 谭玉珊 胡文鹏

军事科学院军事科学信息研究中心 北京 100142

摘要: [目的/意义] 情报研究面临着大量的情报处理需求。随着大数据与人工智能技术的发展,借助算法和工具开展情报处理,实现高价值情报的深度挖掘成为必然。然而,情报研究人员普遍缺乏技术基础,算法与代码的复杂性增加了技术赋能情报研究的难度。[方法/过程] 提出了一种面向情报研究的多智能体协同处理方法,通过构建一个多智能体协同框架,实现自动解析用户指令和需求,利用大模型强大的理解和生成能力,自动解析可用算法模型的 README 文件及代码,自动执行算法并反馈情报处理结果。[结果/结论] 情报研究人员仅需通过简单的界面操作或自然语言交互,即可轻松实现对各种情报处理工具算法的调用,显著提升情报处理效率和智能化水平,大大降低情报处理对情报人员的技术要求,框架技术路线具有广泛的普适性应用价值。

关键词: 情报智能处理; AI 智能体; 大语言模型; 提示工程; 人在回路

中图分类号: TP18; TP391; G35

A Multi-Agent Collaborative Processing Method for Intelligence Research

BAI Jiaxin TAN Yushan HU Wenpeng

Center for Information Research of Academy of Military Science, Beijing 100142, China

Abstract: [Objective/Significance] Intelligence research faces the growing demand for processing large volumes of intelligence data. With the advancement of big data and artificial intelligence technologies, leveraging algorithms and tools to process intelligence and achieve in-depth mining of high-value intelligence has become imperative. However, intelligence researchers generally lack a technical foundation, and the complexity of algorithms and code increases the difficulty of empowering intelligence research through technology. [Methods/Processes] To address this issue, this paper proposes a multi-agent collaborative processing method for intelligence research. By constructing a multi-agent collaborative framework, it enables automatic parsing of user instructions and requirements, leverages the powerful comprehension and generation capabilities

作者简介 白佳欣(2000-), 硕士研究生, 主要研究方向为自然语言处理、大语言模型; 谭玉珊(1972-), 通信作者, 硕士, 研究员, 主要研究方向为大数据挖掘, E-mail: tanys2000@sina.com; 胡文鹏(1992-), 博士, 助理研究员, 主要研究方向为情报研究智能化、机器学习。

引用格式 白佳欣, 谭玉珊, 胡文鹏. 面向情报研究的多智能体协同处理方法 [J]. 情报工程, 2026, 12(1): 3-16.

of large language models (LLMs), automatically analyzes README files and code of available algorithmic models, and autonomously executes algorithms to return intelligence processing results. [Results/Conclusions] Intelligence researchers can effortlessly invoke various intelligence processing tools and algorithms through simple interface operations or natural language interactions. This significantly enhances the efficiency and intelligence level of intelligence processing, greatly reduces the technical skill requirements for intelligence personnel. The framework's technical approach offers broad applicability and universal value.

Keywords: Intelligent Intelligence Processing; AI Agents; Large Language Models; Prompt Engineering; Human-in-the-Loop

引言

情报研究面临着大量的情报处理需求。作为情报研究工作的重要环节，情报处理通过对采集获取的原始情报数据进行加工、提炼和转化，为后续情报分析利用提供高质量、有价值的情报素材^[1]。随着数据量的爆炸式增长，以及大数据、人工智能等技术的飞速发展，技术赋能情报工作转型发展已成为必然趋势^[2]。技术在带来应用利好的同时也提出了很多新的挑战，如：情报研究人员在面对海量数据处理需求时，往往缺乏有效手段来快速筛选、分析和挖掘有价值的信息；许多情报研究人员并非技术专家，缺乏编程技能，难以充分利用各种情报处理算法模型；已开发的一些工具、系统使用较为复杂，用户友好性不强，交互能力弱；信息技术迭代迅速，情报人员需要不断学习新工具新技术，导致工具使用意愿低，出现技术成果与情报研究脱节的“两张皮”情况。例如在代码执行和工具调用方面，情报人员往往难以高效完成环境安装、配置、执行及参数调试等一系列操作。而现有工具文档（如 README）通常仅提供静态说明，无法动态适应用户的个性化需求，也难以处理多代码模块之间的

执行顺序和依赖关系。

当前，以 ChatGPT 为代表的大语言模型（以下简称“大模型”）引发新一轮科技革命，其强大的自然语言理解和生成能力为情报处理带来了新的发展契机。以 GPT-4 模型为例，其不仅能够理解和生成复杂文本，还在推理、创作、编程等任务中表现出色。仅通过少量示例或任务描述，大模型就能完成多样化任务。这种特性有望将依赖高技术背景的专家操作转变为低门槛、高效率的智能化工作模式。另外，多智能体系统（Multi-Agent System, MAS）思路作为大模型时代的热点技术方向，在处理复杂任务时展现出巨大潜力，显著提高处理效率，得到越来越多的关注和运用。例如，微软开发的 AutoGen 框架^[3]通过多个大模型以不同角色协作，共同解决数学计算、代码分析等复杂任务，显著提高任务完成的准确性和效率。

基于此，本文针对情报处理需要，提出了一种面向情报研究的多智能体协同处理方法，旨在以自然语言交互形式，构建“情报人员提出情报处理任务需求 - 大模型理解需求 - 大模型自动调用相应算法模型进行处理 - 向情报人员反馈情报处理结果”的情报处理通用模式，降

低研究人员对技术工具的学习压力，提升情报处理的效率和智能化水平。

1 相关工作

1.1 情报智能体系统研究

近年来，国内外积极探索智能体在情报领域的研发应用。刘细文等^[4]通过多智能体重塑情报处理工作模式，实现由智能体承接具体任务、情报人员负责审核和反馈的新型工作机制；殷跃等^[5]将大模型多智能体技术创新应用到粮食安全应急情报体系中，显著提高应急管理的时效性和准确性。国内启明星辰公司的“安全瞭望塔”系统通过多智能体协作，实现了威胁信息提取、情报要素整合和基于知识的推理分析。由军事科学信息研究中心联合国防科技创新研究院等共同研发的 ChatBIT 智能体赋予军事指挥官智能辅助决策能力。Tseng 等^[6]通过智能体 workflow 实现威胁情报从数据采集到报告输出的自动化，显著减少人工干预。Anthropic 的 Claude Gov 智能体通过定制化 AI 能力，为美国安全部门提供涉密数据解析和辅助态势感知支持。然而，现有相关研究的重点大多是为不同的智能体分配不同的情报任务，协同处理一个复杂的任务。本文重点聚焦情报研究中的某一具体情报处理环节，针对大模型如何自动调用传统的情报处理算法，确保执行的自动化开展研究，以代码执行为例，从微观层面剖析大模型智能体运作过程中可能遇到的难题堵点，提高大模型技术在情报研究工作中的应用便捷性。

1.2 情报研究“人在回路”思想

“人在回路”（Human-in-the-loop）的思想主张发挥人类的认知优势以克服机器学习的局限，推动人机协同从“任务替代”向“任务协作”转变^[7]。这一理念已广泛应用于软件开发^[8]、机器翻译^[9]和自动化文本标注^[10]等领域。情报分析领域同样强调 AI 赋能、人在回路。Phythian^[11]指出，情报分析工具应作为分析师的“外骨骼”，为其提供强有力的辅助，而不是成为分析师的主宰；丁洪鑫等^[12]以人在回路的情报分析过程对智能情报分析系统进行整体功能架构及数据架构设计，在提高准确性的同时降低了使用门槛。Palantir AIP、Scale AI Dnovan 及“艾武大模型+”等军事平台^[13]也融合了“人在回路”理念，依赖人的专业判断对大模型输出进行校验和干预，借助大模型完成决策分析、任务规划等任务。本研究借鉴相关思路，通过用户干预、多轮交互与反馈对齐用户意图，帮助具备专业知识但缺乏技术背景的用户在面对大规模情报数据处理需求时，能简洁高效地调用算法模型完成相关任务。

1.3 提示工程与提示调优

提示工程是引导大模型发挥推理与生成优势的关键技术，输入提示的设计质量直接影响模型输出的优劣^[14]。其中，思维链提示（Chain-of-Thought, CoT）通过引导大模型进行连贯、逐步的推理，激发大模型产生结构化和深入思考的回答。人工编写提示需要付出大量人力进行设计和调整，近年来又涌现出若干借助大模型进行自动提示设计的方法^[15-16]。但是，该类方法主要聚

焦于具体任务，需要为每个任务单独搜索最优提示，同时需要对每个模型进行适配，缺少对不同模型的鲁棒适用。而由 Cheng 等^[17]提出的黑箱提示优化方法（Black-box Prompt Optimization, BPO）通过人类偏好引导提示优化，无需调整大模型参数，即可更好地对齐大模型与用户意图。本文借鉴 BPO 思想对用户提示进行优化和改进。

1.4 大模型驱动的多智能体协同编码

本文所构建的框架主要受大模型时代多智能体在协作编码方面相关工作的启发。早期的多智能体框架实现主要聚焦于基本的协作策略^[18]。Dong 等^[19]提出的 Self-Collaboration Framework 通过为大模型分配特定角色（如分析师、编码员和测试员），将任务分解并实现协同处理，提高了代码生成的质量和效率。随后，Hong 等^[20]开发的 MetaGPT Framework 中引入标准操作程序（SOPs）和执行反馈机制，提高了协作稳定性和执行正确性。近期，Huang 等^[21]提出了 CodeCOT 框架，通过引入

自检和改进机制解决语法错误问题，进一步增强了代码生成的可靠性和实用性。这些研究主要聚焦于技术人员主导的自动编程任务，对于非技术用户，仍然存在较高的使用门槛。为此，本文提出从代码的 README 分析出发，完成从指令解析到结果解释的自动化流程，以提高解决复杂问题的能力，并给非技术用户提供更友好的使用体验。

2 多智能体协同情报处理

2.1 基本思路

传统情报处理模式以情报人员为核心，借助各类检索、分析和可视化工具完成信息搜集、算法选择、结果解释等一系列任务，如图 1 左所示。这种“人主导 - 工具辅助”的方法虽然在一定程度上提高了处理效率，但在面对任务复杂度不断上升、模型资源日益丰富、数据体量日趋庞大的情报环境时，显露出“认知负荷高、自动化水平低、响应速度慢”等多重瓶颈。

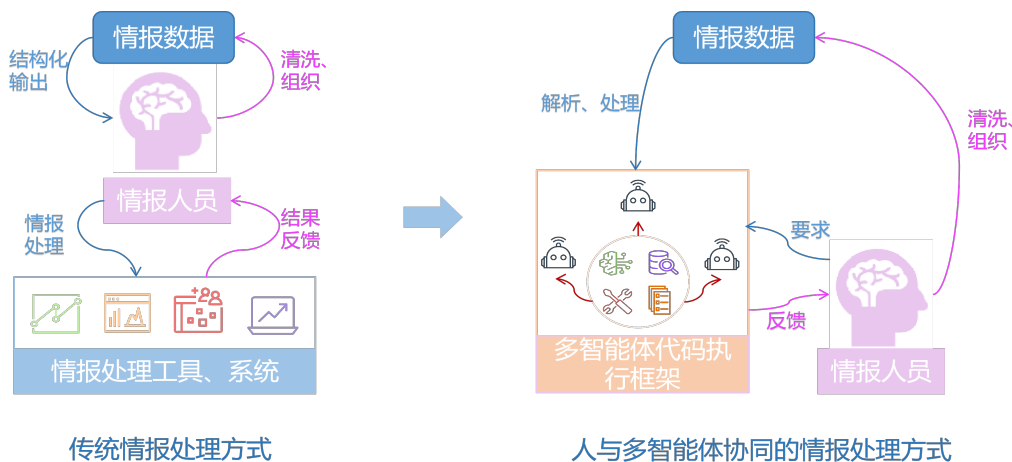


图 1 多智能体协同框架赋能下的情报处理流程变化示意图

本文在刘细文等^[4]提出的多智能体任务分工思想基础上进一步拓展,提出一种大模型驱动的多智能体协同情报处理方法,通过构建一个多智能体协同框架,将情报处理的主体任务交由大模型调用智能体来完成,情报研究人员在其中起到需求交互、效果反馈与迭代作用,如图1右所示。需要特别注意的是,虽然当前大模型具备强大的自然语言理解与生成能力,能够处理诸如关系抽取、情感分析等任务,但与直接基于提示语调用大模型的方式不同,本文提出的多智能体协同处理方法着重强调的是结构化智能协同与任务调度能力,核心目标不仅在于简单的功能实现,更侧重于构建“可控、可解释、自动化”的智能化处理流程。通过将提示工程、思维链推理和智能体分工协作三者结合,利用大模型的生成推理能力,构建出“自然语言指令-提示链引导-多智能体执行”的协同路径。对于非技术背景的情报研究人员而言,这种方法支持用户以自然语言方式提出任务需求,通过多智能体协同框架进行任务分析、代码执行和结果反馈,轻松实现“用自然语言调度算法模型”。

具体而言,该方法以大模型的提示生成能力为驱动引擎,通过提示智能体进行任务编排。在用户提出任务需求时,框架使用黑箱提示优化方法对用户的自然语言输入进行语义重构、目标聚焦和信息增补,在交互过程中,通过动态调整用户输入内容,使任务意图从自然语言向机器指令平滑过渡。提示智能体会根据用户提示生成具有逻辑层次的提示链,以思维链方式引导其他智能体开展协同处理。在智能体协同执行方面,采用了多智能体角色分工机制,

将任务处理流程拆分为不同子任务,包括提示生成、文件分析、代码生成、错误修复、结果解析等模块,每个智能体承担不同子任务,形成模块化的角色体系。各智能体在共享上下文的基础上,依据提示协同工作,具备自主决策与交互能力。整个框架形成一个“提示-响应-再提示”的动态交互过程,结合大模型的语言生成能力,智能体可以根据上下文灵活生成代码、填补缺失信息并结构化分析结果。

2.2 多智能体协同框架设计

多智能体协同框架架构如图2所示。该框架由五个基于大模型的智能体以及框架底层构成。五个智能体分别是分析智能体、编码智能体、修复智能体、解析智能体和提示智能体。各智能体分工明确,协同执行任务。从用户上传代码文件或提出任务要求开始,通过用户与多智能体协同框架的多轮交互,大模型自动实现代码及README文件的分析、任务执行及结果解析反馈。

2.2.1 智能体

(1) 分析智能体。该智能体专注于解析底层调用的算法模型的README文件和代码。根据不同的编程语言,采用相应的抽取逻辑,从README中获取环境配置、依赖项、运行指令等关键信息,并从代码文件中识别输入输出数据格式、核心函数等内容。分析智能体以模块划分的方式为目标任务生成编码建议。具体而言,该智能体将提取的信息按照代码功能、输入/输出描述、依赖安装指令、执行代码等类别进行归类,并使用特定标识符标记各模块,以指导编码智能体编写代码。

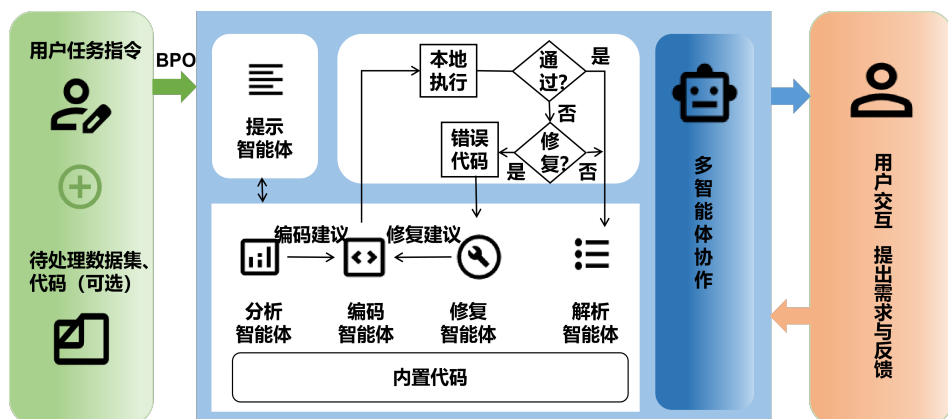


图2 多智能体协同框架总体架构

(2) 编码智能体。编码智能体负责编写代码和指令，其工作在分析智能体和修复智能体的协同和反馈下进行。编码智能体在该框架中具有两项职责：一是以分析智能体生成的结构化编码建议作为输入，根据相应的编码建议生成代码块和指令，对于复杂逻辑，编码智能体采用分块生成方式生成完整可执行代码文件；二是根据修复智能体反馈的修复建议修复代码和指令。编码智能体应用修复智能体提供的修复建议生成并自动执行修复后代码。

(3) 修复智能体。修复智能体负责在代码执行失败或指令出错时，提供针对性的修复建议。该智能体实时监控代码执行结果以捕获代码执行中的异常，利用错误日志和代码上下文进行错误诊断和分析。根据错误类型提供针对性的修复方案，按修复成功率（历史统计）和复杂度（代码改动量）对修复方案排序，优先尝试高概率方案。每次修复后，修复智能体会对已经尝试过的修复建议剪枝。如果修复超过预设阈值 t 或没有新的修复建议，修复智能体将停止进一步的修复工作并生成详细的错误报告，包括问题原因、已尝试的修复方案以及可

能的后续建议。

(4) 解析智能体。解析智能体的目标是解读和总结成功执行的代码结果，并对终止修复的执行结果进行分析。在代码执行成功时，它会根据任务的输出文件内容，使用易于用户理解的指标对结果进行解读和评估，并以自然语言对结果的含义进行总结。在代码执行失败时，解析智能体会基于修复智能体提供的错误报告，分析修复思路的有效性，并提出潜在的改进方向。

(5) 提示智能体。提示智能体的重点是生成高质量的大模型提示语言（Prompt）。提示智能体根据任务特征（例如用户是否上传代码文件）动态调整提示策略。当用户上传代码文件时，提示智能体结合用户上传的文件和输入信息，生成适用于其他各智能体的提示；当用户没有上传代码文件时，提示智能体会聚焦于解析用户用自然语言描述的功能需求。在交互过程中，该智能体会结合用户输入、系统环境状态和执行结果反馈持续优化提示内容，形成“环境感知 - 提示生成 - 执行反馈 - 动态优化”的提示模式。

2.2.2 框架底层

多智能体协同框架的底层可内置面向情报处理各类任务的可执行算法模型（代码级），支持多种经典与现代机器学习方法，实现对复杂、多样化情报处理场景的高效适配。例如，可内置情报文本聚类（包括短文本与长文本）和关联分析算法，以及频繁模式挖掘、Apriori 算法、广度优先搜索（BFS）、深度优先搜索（DFS）等结构化数据挖掘与图算法等。

以文本聚类为例，短文本聚类通过解析输入文本、分词、去除停用词后，基于 Jaccard 相似度计算方法将文本分配到不同的聚类中，并根据相似度阈值动态创建或更新聚类；长文本聚类通过将文本嵌入向量降维后，使用基于密度的 DBSCAN 聚类算法进行聚类分析。本文提出的多智能体框架会在用户没有上传代码文件时，根据用户需求和待处理的数据集的格式、规模等特点，自动调用智能体底层内置的最合适代码开展情报处理工作。

2.3 运作逻辑

在该框架工作模式下，情报用户可通过两种方式与多智能体进行交互，实现情报处理：一是用户提出具体需求和上传待处理的数据，框架自动识别用户需求，并调用内置的代码模块进行数据处理；二是用户在上传待处理数据的同时，同步上传自己选用的、包含 README 文件的代码，本框架自动对其进行解析和执行，最终输出情报处理结果。在这些交互过程中，情报人员仅需以自然语言表达需求，框架通过黑箱提示优化方法（BPO）对用户需求进行优化，然后再传递给多智能体进行具体处理。例

如，当用户表达需求“我上传了一个代码，告诉我这个代码能够实现的功能和需要的数据”时，BPO 会将其优化为“请简要解释我上传的代码，重点说明它可以实现的功能以及所需的数据”。这种优化使用户需求描述更加精准，并补充和强调了用户的关注点，从而确保框架能够高效响应需求。此外，用户可以在交互过程中动态调整输入数据，例如在聚类任务中更换停用词表等。

整个框架的执行流程分为五个环节：

（1）任务感知与动态提示生成。情报智能处理流程始于人与多智能体的协同交互。用户上传代码文件或提出任务需求后，由 BPO 优化后的需求被传递至多智能体。提示智能体感知用户需求并生成指导其他智能体行动的 CoT，例如 README 分析提示、代码生成提示等。当用户在任务执行过程中提供新的数据或调整需求时，提示智能体会感知变更，并根据先前结果和用户输入调整提示内容，确保在适应动态变化的同时维持上下文的一致性。

（2）文件分析与结构化信息提取。在本环节中，分析智能体根据提示智能体的提示，对用户上传的 README 文件和代码进行深度解析，提取任务所需的关键信息。若用户采用第一种交互方式，分析智能体会分析预调用算法的 README 文件和代码。在信息提取的同时还会进行代码完整性校验。当 README 文件中包含完整的执行指令时，分析智能体会对其进行结构化处理，区分代码和 Shell 命令；若 README 中只有算法实现的文字描述但缺少实现的明确指令时，分析智能体会识别缺失部分并标注，在环节三中由编码智能体补全。基于

README 和代码提取的结构化信息被作为编码建议供编码智能体参考。这一处理环节非常关键，能够解决当前大模型执行代码过程中因解析配置不全，导致最终无法自动执行的难题。

(3) 代码生成与执行。在本环节中，编码智能体结合提示智能体的提示，将分析智能体提供的编码建议转化为代码和指令。生成的代码和指令在本地环境中运行。若代码执行无误，会直接进入环节五进行解析，否则进入第四环节。

(4) 错误诊断与代码修复。当代码运行失败时，修复智能体利用日志等信息识别失败原因，生成可行性修复建议。修复建议和错误代码一起转发给编码智能体进行更正，从而生成新修复的代码片段。修复后的代码将被再次执行，启动新的迭代，直至代码成功执行或达到设定的修复上限。若修复失败超出阈值或缺少可行的修复建议，修复智能体会生成详细的失败分析报告，提示用户可能的手动干预策略，以保障任务的可执行性。成功执行的代码会进入环节五。

(5) 智能情报解析与反馈。成功执行的代码结果由解析智能体进行智能分析和反馈。解析智能体负责数据语义解析，帮助用户理解执行结果的业务含义及其对任务目标的贡献。对于失败的执行结果，解析智能体会根据修复智能体的修复反馈，分析失败原因并提供进一步的改进建议。

3 多智能体协同框架实战应用

3.1 多智能体协同情报智能处理系统

基于提出的多智能体协同处理方法，本文

实现了一套多智能体协同情报智能处理系统。系统分为前端和后端模块，通过简单的界面操作和自然语言问答，简化用户与多智能体协同框架的交互。

系统采用前后端分离架构，前端使用 Vue3 实现用户交互，支持用户上传算法模型代码、README 文件、待处理数据等（支持 zip 格式），输入任务需求描述，并提供内置代码（如聚类、分类等）供用户选择。后端基于多智能体协同框架实现，利用本地部署的大模型，通过角色指令动态构建框架中的五个功能分化的智能体，每个智能体承担特定角色，并通过任务队列实现任务调度和多智能体协作。同时，后端基于 FastAPI 提供 API 服务，通过实时日志记录执行过程和错误日志，以支持问题追踪与调试。

系统前端展示效果如图 3 所示。情报研究人员无需掌握复杂的技术细节，仅需通过与多智能体协同框架的简单交互，后者即可根据用户需求轻松开展情报数据处理，情报处理的便捷性、高效性得到有效提升。

3.2 效果验证

本文基于研发的多智能体协同框架，围绕情报处理中的典型任务场景开展了系统性应用验证。

3.2.1 案例选取

实验从 GitHub 平台中随机筛选了 10 个较高活跃度的开源项目作为测试样本，如表 1 所示，这些项目均提供 md 格式的 README 文件。所选项目覆盖了图像聚类、短文本聚类、长文本聚类、同义词关系挖掘、文本摘要生成等典型情报处理任务类型，具有较强的代表性。



图3 系统前端效果展示

表1 项目名称及对应任务类型

编号	项目名称	任务类型
1	synonym_detection	同义词关系挖掘
2	TextCluster	短文本聚类
3	Contrastive-Clustering	图像聚类
4	text-clustering	长文本聚类
5	Chinese-Text-Classification	中文文本分类
6	cnn-text-classification	情感分类
7	tacred-relation	关系抽取
8	Twitter-sentiment-analysis	情感分类
9	ETM	主题建模
10	ChineseNER	中文命名实体识别

项目具体内容是: (1) synonym_detection: 通过词向量相似度 (Word2Vec)、语义共现网络、编辑距离 (Levenshtein) 三种算法从不同角度识别词语的同义词; (2) TextCluster: 通过分词、向量化、相似度计算实现短文本层次聚类和相似查询; (3) Contrastive-Clustering: 基于对比学习的无监督聚类方法, 通过结合实例级对比学习 (区分不同样本) 和聚类级对比学习 (区分不同类别) 提升图像聚类性能; (4) text-clustering: 长文本聚类, 能够将大量文本转化为向量表示并使用 DBSCAN 算法聚类, 同时为每个聚类生成语义标签; (5)

Chinese-Text-Classification: 使用多种文本分类模型实现中文文本分类, 包括 TextCNN、TextRNN、FastText、TextRCNN、BiLSTM_Attention、DPCNN 和 Transformer 等; (6) cnn-text-classification: 基于 TensorFlow 实现的卷积神经网络, 实现了电影评论情感数据集分类; (7) tacred-relation: 基于位置感知注意力机制的循环神经网络模型, 用于从文本中抽取实体间的语义关系。实验使用公开关系抽取数据集 TACRED 数据集; (8) Twitter-sentiment-analysis: 使用多种模型实现情感分类, 包括朴素贝叶斯、逻辑回归、决策树、多层感知机等模型以及多模型融合方法; (9) ETM: 一种将词嵌入与主题建模相结合的模型, 能够在共享的嵌入空间中同时表示词和主题, 从而学习出语义可解释的主题和词向量, 对低频词和停用词具备较强的鲁棒性; (10) Chinese-NER: 使用双向 LSTM 和 CRF 模型进行中文命名实体识别, 结合字符向量和词边界特征, 能够从中文文本中识别出人名、地名等实体。

3.2.2 效果展示

本节从所测试的十个项目中选择图像聚类、长文本聚类和中文文本分类三个任务, 展示应用多智能体协同框架的实现效果。

案例一: 基于用户上传的 GitHub 代码、README 文件和 CIFAR-10 数据集实现图像聚类^[22]。

图像聚类是情报研究中的常见任务。例如, 识别海量图片中哪些是飞机, 哪些是舰艇。如何用大模型等技术手段开展图像聚类, 帮助情报研究人员快速了解海量图像集的内容, 定位并筛选出研究感兴趣的图像聚合, 对于进一步

开展图像分析非常重要。

本实验中, 用户提供的代码结合了对比学习和无监督聚类方法, 提供的待聚类原始数据集包含 60000 张 32×32 像素的彩色图像, 涵盖飞机、汽车、鸟、猫等常见物体。框架启动后, 分析智能体首先解析 README 文件, 生成环境配置、任务执行步骤等建议如图 4 所示, 并支持动态调整参数, 如模型保存路径和数据集。实验使用默认参数配置。编码智能体将更新后的建议转换为可执行代码, 自动完成依赖安装、模型训练等操作, 同时修复智能体实时检测并修复可能出现的错误, 确保代码顺利执行。在执行过程中, 修复智能体给出反馈提示, 并建议是否继续修复或中止执行, 由情报人员作出决策。聚类完成后, 解析智能体会对聚类结果进行详细分析并输出最终结果。最终, 该框架仅耗时 8 分 40 秒就完成了聚类, 60000 张图像被分为 10 类, 框架自动执行的准确率 (ACC) 为 0.2636, F1 分数 (F1-Score) 为 0.1810, 与人工调用的结果 (ACC=0.2547, F1=0.1769) 相近, 验证了框架的有效性和可行性。

案例二: 使用 text-clustering 项目进行情报文本聚类, 通过 MiniLM 模型进行文本向量化, 结合 UMAP (一种用于保持数据局部结构的降维算法) 降维和 DBSCAN 聚类算法实现语义分组, 并利用 Qwen2.5-7B 大模型生成“三词式”聚类主题摘要。

大数据时代, 情报研究一开始往往就面临着文本太多、看不过来的挑战, 亟需对文本进行聚类, 甚至概括出主题, 方便研究人员快速了解情况。本实验正是针对该需求开展相关效果验证。

```
Installation commands:
```bash
pip install torch>=1.6.0
pip install torchvision>=0.8.1
pip install munkres>=1.1.4
pip install numpy>=1.19.2
pip install opencv-python>=4.4.0.46
pip install pyyaml>=5.3.1
pip install scikit-learn>=0.23.2
```

Setup Steps:
```plaintext
Step 1: Clone or download the repository containing the necessary Python files.
Step 2: Ensure that the dataset files are present or download them if necessary.
Step 3: Modify the parameters in the "config/config.yaml" file according to your training and test requirements.
```

Execution Steps:
```bash
To start training on a dataset other than STL-10
python train.py

To start training specifically on STL-10
python train_STL10.py

To test the trained model
python cluster.py
```

Reference Files:
[files]
- train.py: This is the main script to start training the model.
- train_STL10.py: This script is specifically for training on STL-10 dataset with a different strategy.
- cluster.py: This script is used to test the trained model.
[files]
```

图 4 分析智能体为图像聚类任务生成的编码建议(后端)

实验数据来源于多个公开情报平台（如 SentineOne、Mandiant、CISA 等），通过预设关键词抓取网页内容，提取高质量段落文本，构建一个包含 1000 条真实情报段落的数据集。

多智能体协同框架在执行过程中分为三个阶段：首先由分析智能体解析 README 文件，生成包括依赖安装、执行步骤等在内的编码建议，如图 5 所示；情报人员可以采用默认配置，或

```
1. Instruction Type: Python code execution
2. Installation commands:
```bash
pip install scikit-learn umap-learn sentence_transformers faiss-cpu plotly matplotlib datasets
```

3. Setup Steps:
```plaintext
Step 1: Clone or download the repository.
Step 2: Navigate to the repository folder.
Step 3: Ensure the dataset file "intelligence_dataset.jsonl" is present in the specified path.
Step 4: Modify the parameters if necessary.
```

4. Execution Steps:
```python
from src.text_clustering import ClusterClassifier
from datasets import Dataset
import json

SAMPLE = 1000
with open("/data/bjx/text-clustering-main/intelligence_dataset.jsonl", "r") as f:
 data = [json.loads(line) for line in f]

dataset = Dataset.from_list(data)
texts = dataset.select(range(SAMPLE))["text"]
cc = ClusterClassifier(embed_device="cuda")
embs, labels, summaries = cc.fit(texts)
cc.show()
cc.save("./cc_100k")
```

```bash
python pipeline.py
```

5. Reference Files:
[files]
- src/text_clustering.py: Main clustering script.
[files]
```

图 5 分析智能体为情报文本聚类任务生成的编码建议(后端)

根据任务需求调整聚类粒度等关键参数，无需编写任何代码。随后编码智能体将这些建议转换为可执行的 Python 代码及 bash 指令并自动运行，期间修复智能体实时检测和解决可能出现的问题，遇到异常提供简明提示和处理建议，由情报人员决定是否中止或继续；最终由解析智能体分析聚类结果，便于情报人员快速理解

每类语义内容。

实验结果表明，框架顺利实现了该文本聚类代码的功能（文本输入、语义分析、结果可视化展示），成功将数据集聚为 42 类并生成主题摘要标签，如图 6 所示，与人工聚类结果（45 类）基本一致，验证了框架在真实情报环境中的可行性和实用性。



图 6 聚类效果图

案例三：使用基于 THUCNews 构建的数据集，进行中文文本分类。

如何将文本按照分类体系进行自动分类，是情报处理的一项基础性工作。本文利用提出的多智能体情报处理框架，对该类任务开展实验效果验证。

实验采用的数据集包含 20000 条新闻标题，每条长度为 20~30 字，按财经、房产、股票等分为 10 个类别，每类 20000 条。由用户上传代码、README 文件及数据集，代码包含 TextCNN、TextRNN、FastText 和 TextRCNN 等不同文本分类模型。在交互中，情报人员根据任务需求和模型理解，指定使用

TextCNN 模型实现文本分类。分析智能体自动分析上传的 README 和代码结构，提取代码依赖、训练设置（训练集、验证集、测试集数量）等信息，结合用户选定模型 TextCNN 生成相应的编码建议，如图 7 所示；情报人员可以基于经验进一步修改参数配置（如 batch-size 等）。编码智能体根据建议生成代码并依序执行，发生错误时由修复智能体进行检测和修复。最终分类结果由解析智能体进行详细解析。实验结果显示，由框架调用 TextCNN 模型分类的准确率为 91.18%，与人工调用分类的准确率 91.24% 相近，表明框架能完全实现代码文本分类的功能。

```
1. Instruction Type: Python code execution
2. Installation commands:
```bash
pip install torch==1.1
pip install tqdm
pip install sklearn
pip install tensorboardX
```

3. Setup Steps:
```plaintext
Step 1: Clone or download the repository.
Step 2: Ensure data files are present.
Step 3: Modify the parameters if necessary.
```

4. Execution Steps:
```bash
python run.py --model TextCNN
```

5. Reference Files:
[files]
- run.py: Main script to train and test models.
[files]
```

图7 分析智能体为中文文本分类任务生成的编码建议(后端)

3.2.3 结果分析

本文面向情报智能处理中的多个典型任务，基于多智能体协同框架开展应用验证，初步验证了所提出的多智能体协同框架在真实项目场景中的可行性，在不依赖人工纠正代码错误的情况下，该框架能够顺利完成包括图像聚类、短文本聚类、长文本聚类、同义词关系挖掘、文本摘要生成等多项任务，实现任务分析、代码自动调用和初步结果生成。任务执行流程完整。

与人工逐步执行代码的方式相比，通过框架执行代码在结果上展现出较高的一致性。对于 README 结构清晰的项目，框架调用的代码可以完全正确执行，并且框架平均处理时间较专业程序员操作节省约 30%—40%。同时，框架节省了情报研究人员学习前置知识的时间，例如代码基础知识和算法知识，提高了情报研究效率。

4 结语

本文针对技术赋能情报研究需求，充分利用大模型的理解与生成能力，提出了一种面向情报研究的多智能体协同处理方法，构建了多智能体协同框架，并结合典型情报处理任务开展了实验验证。可以看出，该方法极大地降低了情报处理对情报研究人员的 technical 能力要求，设计理念和实现架构可适用于各类情报处理任务，体现了大模型技术推动情报处理的新思路。

参考文献

- [1] 张涛, 马海群. 智能情报分析中算法风险及其规制研究 [J]. 图书情报工作, 2021, 65(12): 47-56.
- [2] 耿国桐, 卢胜军, 雷帅, 等. 人工智能赋能情报研究的变革与发展 [J]. 情报学进展, 2024, 15(1): 278-317.
- [3] WU Q, BANSAL G, ZHANG J, et al. Autogen: enabling next-gen LLM applications via multi-agent conversation[J]. (2023-08-15) [2026-03-04]. <https://>

- arxiv.org/abs/2308.08155.
- [4] 刘细文, 付芸, 孙蒙鸽. 驱动情报工作范式变革的情报智能体技术解构 [J]. 图书情报工作, 2025, 69(1): 4-15.
- [5] 殷跃, 张海涛, 刘彦辉, 等. 基于大模型智能体的粮食安全应急情报体系构建研究 [J]. 情报理论与实践, 2025, 48(1): 20-30.
- [6] TSENG P Y, YE H Z D, DAI X, et al. Using LLMs to automate threat intelligence analysis workflows in security operation centers[J]. (2024-07-18) [2026-03-04]. <https://arxiv.org/abs/2407.13093>.
- [7] 程海东, 胡孝聪. “人在回路”的道德机制: 机器人伦理实践的新路径 [J]. 河南师范大学学报 (哲学社会科学版), 2025, 52(1): 108-114.
- [8] TAKERNGSAKSIRI W, PASUKSMIT J, THONGTANUNAM P, et al. Human-in-the-loop software development agents[J]. (2024-11-19) [2026-03-04]. <https://arxiv.org/abs/2411.12924>.
- [9] YANG X, ZHAN R, WONG D F, et al. Human-in-the-loop machine translation with large language model[J]. (2023-10-13) [2026-03-04]. <https://arxiv.org/abs/2310.08908>.
- [10] PANGAKIS N, WOLKEN S. Keeping humans in the loop: human-centered automated annotation with generative AI[J]. (2024-09-14) [2026-03-04]. <https://arxiv.org/abs/2409.09467>.
- [11] PHYTHIAN M. Intelligence analysis today and tomorrow [J]. Security Challenges, 2009, 5(1): 67-83.
- [12] 丁洪鑫, 汪榕, 周维, 等. 人机结合的智能情报分析系统设计与实现 [J]. 现代信息科技, 2024, 8(11): 69-75.
- [13] 高志强, 沈佳楠, 姬纬通, 等. 大模型技术的军事应用综述 [J]. 南京航空航天大学学报, 2024, 56(5): 801-814.
- [14] 黄峻, 林飞, 杨静, 等. 生成式 AI 的大模型提示工程: 方法、现状与展望 [J]. 智能科学与技术学报, 2024, 6(2): 115-133.
- [15] ZHOU Y, MURESANU A I, HAN Z, et al. Large language models are human-level prompt engineers[J]. (2022-11-03) [2026-03-04]. <https://arxiv.org/abs/2211.01910>.
- [16] YANG C, WANG X, LU Y, et al. Large language models as optimizers[J]. (2023-09-07) [2026-03-04]. <https://arxiv.org/abs/2309.03409>.
- [17] CHENG J, LIU X, ZHENG K, et al. Black-box prompt optimization: aligning large language models without model training[J]. (2023-11-07) [2026-03-04]. <https://arxiv.org/abs/2311.04155>.
- [18] TROITZSCH K G. Multi-agent systems and simulation: a survey from an application perspective[C]// UHRMACHER A M, WEYNS D. Multi-agent systems: simulation and applications. Boca Raton: CRC Press, 2009: 53-75.
- [19] DONG Y, JIANG X, JIN Z, et al. Self-collaboration code generation via ChatGPT[J]. ACM Transactions on Software Engineering and Methodology, 2024, 33(7): 1-38.
- [20] HONG S, ZHENG X, CHEN J, et al. MetaGPT: meta programming for multi-agent collaborative framework[J]. (2023-08-01) [2026-03-04]. <https://arxiv.org/abs/2308.00352>.
- [21] HUANG D, BU Q, CUI H. CodeCoT and beyond: learning to program and test like a developer[J]. (2023-08-17) [2026-03-04]. <https://arxiv.org/abs/2308.08784>.
- [22] LI Y, HU P, LIU Z, et al. Contrastive clustering[C]// Proceedings of the AAAI Conference on Artificial Intelligence. 2021, 35(10): 8547-8555.

(责任编辑: 浦墨)