Preprints.org

Article

Not peer-reviewed version

Privacy-Preserving Structured Knowledge Extraction from Census-Style Records Using a Hierarchical Multi-Agent Open-Weight LLM Architecture

Adeeba Tarannum , Muzakkiruddin Ahmed Mohammed , Shames Al Mandalawi , Mert Can Cakmak * , John R. Talburt

Posted Date: 20 May 2026

doi: 10.20944/preprints202605.1337.v1

Keywords: hierarchical multi-agent systems; structured information extraction; census-style records; address parsing; open-weight large language models; privacy-preserving AI; on-premise deployment

Preprints.org is a free multidisciplinary platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC, OpenAlex.

Copyright: This open access article is published under a Creative Commons CC BY 4.0 license, which permit the free download, distribution, and reuse, provided that the author and preprint are cited in any reuse.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

Disclaimer/Publisher's Note: The statements, opinions, and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions, or products referred to in the content.

Article

Privacy-Preserving Structured Knowledge Extraction from Census-Style Records Using a Hierarchical Multi-Agent Open-Weight LLM Architecture

Adeeba Tarannum, Muzakkiruddin Ahmed Mohammed, Shames Al Mandalawi, Mert Can Cakmak * and John R. Talburt

Center for Advanced Research in Entity Resolution and Information Quality ERIQ, University of Arkansas at Little Rock, Little Rock, United States

* Correspondence: mccakmak@ualr.edu

Abstract

Census-style records contain sensitive personal information that must be transformed into structured fields before it can support record linkage, geocoding, duplicate detection, identity verification, and statistical processing. This task is challenging because addresses appear in many forms, including standard street addresses, apartment and unit addresses, university addresses, military APO/FPO/DPO addresses, rural routes, highway addresses, and attention-line records. Rule-based parsers are efficient and transparent but often fail on non-standard formats. Single-prompt Large Language Model (LLM) approaches improve generalization but can suffer from record skipping, field conflation, and long-context degradation when processing heterogeneous documents. In addition, privacy and governance requirements limit the use of cloud-hosted models for sensitive census-style data. This paper presents a three-stage hierarchical multi-agent architecture for structured knowledge extraction from census-style records using a locally deployed open-weight LLM. A Planner Agent analyzes the input document and formulates an extraction strategy. A Manager Agent converts this strategy into a dependency-aware task graph. A fleet of eight specialized Worker Agents performs extraction, validation, and formatting, while a bounded feedback loop supports limited autonomous recovery from extraction failures. The system runs on gpt-oss-20b, a 21-billion-parameter open-weight model deployed on local infrastructure, so input records do not need to be transmitted to an external model provider. The system is evaluated on 700 synthetically generated records across seven address categories. It achieves 95.7% component-level exact match accuracy, compared with 52.3% for a rule-based baseline and 80.9% for a single-prompt LLM baseline using the same model. The largest improvements occur on challenging non-standard categories, including highway addresses (93% vs. 11% rule-based), military addresses (91% vs. 28%), and attention-line records (90% vs. 22%). The results suggest that multi-agent decomposition can improve the robustness and completeness of open-weight LLM extraction while preserving the privacy advantages of on-premise deployment. The study should be interpreted as a prototype evaluation on synthetic data rather than a production-readiness claim.

Keywords: hierarchical multi-agent systems; structured information extraction; census-style records; address parsing; open-weight large language models; privacy-preserving AI; on-premise deployment

1. Introduction

Every census begins with records about people and households: names, residential addresses, contact information, dates of birth, and demographic or administrative details collected during population enumeration. At national scale, these ordinary pieces of information become one of the largest and most sensitive datasets maintained by government agencies. The United States counted 331.4 million people in the 2020 Census [41], illustrating the scale at which such records are collected, protected, and processed. Before census data can support population mapping, record

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

linkage, geocoding, identity verification, duplicate detection, or policy planning, raw responses must be converted into consistent structured fields [8,14,16].

This conversion is difficult because census records do not follow one simple format. Addresses may appear as conventional street addresses, apartment or unit addresses, university department addresses, rural routes, highway addresses, military APO/FPO/DPO addresses, or attention-line records that include organizations and routing instructions. A human reader can usually interpret these variations from context, but an automated system must assign each token to the correct component, such as street number, street name, unit, city, state, ZIP code, organization, or attention line. Errors at this stage can propagate into downstream systems and affect matching, geocoding, household detection, and statistical processing.

Traditional parsing methods have addressed this problem with varying degrees of success. Rule-based systems are fast, transparent, and inexpensive, but they depend on patterns anticipated by their designers. Prior work has shown that the `usaddress` library performs well on common street addresses but struggles with military, highway, and attention-line formats, where overall accuracy dropped to 47.55% [27]. Statistical approaches such as Conditional Random Fields and Hidden Markov Models improved generalization by learning from labeled examples [4,19], but their performance depends heavily on the coverage and quality of training data. Neural methods, including BiLSTM-CRF models, transformers, and BERT-based Named Entity Recognition systems, further improved entity recognition [12,21]. However, many such systems identify address spans at a coarse level rather than decomposing them into the fine-grained fields required for census processing.

Large Language Models introduced a new possibility for structured extraction. Since GPT-3 [6], instruction-following LLMs have shown that complex information extraction tasks can be performed with little or no task-specific training data. In address parsing, an LLM can use natural language instructions to infer that an APO or FPO value functions as a city, that AE or AP functions as a military state code, or that an attention line should be separated from the physical address. A prior LLM-based census address parsing system using Claude 4.0 Sonnet through a cloud API achieved 99.8% exact-row accuracy on a 1,500-record address corpus [39].

Despite this progress, two major challenges remain. The first is extraction reliability in long and heterogeneous documents. A single-prompt LLM approach may perform well on isolated records, but quality can degrade when the model is asked to extract multiple types of personal information from many records at once. Prior work has shown that LLMs can lose track of information in the middle of long contexts, a phenomenon known as "lost in the middle" [23]. In our preliminary testing, a single-prompt approach applied to a 700-record dataset skipped 47 records and achieved substantially lower component-level accuracy than the proposed multi-agent architecture. The second challenge is privacy. Census data contains highly sensitive personally identifiable information, and sending such data to cloud-hosted LLM services can create governance and compliance concerns. For census-style processing, high extraction accuracy alone is not sufficient. The system must also support secure deployment.

Open-weight language models make a different deployment model possible. In 2025, OpenAI released gpt-oss-20b, a 21-billion-parameter open-weight language model under the Apache 2.0 license [33]. Because the model can be deployed on local infrastructure, sensitive records can be processed without transmitting data to an external service. This changes the practical design space for census extraction. Instead of choosing between brittle local rule-based systems and powerful cloud-hosted LLMs, it becomes possible to investigate whether a locally deployed open-weight model can provide useful extraction quality while keeping data within the organization's secure environment.

This paper presents a hierarchical multi-agent architecture for structured knowledge extraction from census-style records using locally deployed gpt-oss-20b. The system divides the extraction process into three stages. A Planner Agent analyzes the input document and formulates an extraction strategy. A Manager Agent converts that strategy into a dependency-aware task graph. A fleet of specialized Worker Agents performs extraction, validation, and formatting tasks for specific personal

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

information types. The architecture also includes a bounded feedback loop. When a worker fails or produces low-confidence output, the failure context is returned to the Planner Agent, which revises the strategy and attempts recovery. This design is intended to reduce the cognitive burden placed on any single prompt while preserving the privacy advantages of on-premise deployment.

We evaluate the system on 700 synthetically generated census-style records across seven address categories: standard, individual, university, military, rural route, highway, and attention-line addresses. The proposed multi-agent system achieves 95.7% component-level accuracy, compared with 52.3% for a rule-based parser and 80.9% for a single-prompt LLM baseline using the same gpt-oss-20b model. The largest improvements occur on the most challenging categories, including highway, military, and attention-line addresses. We also compare the proposed system with two published census parsing systems: a pattern-based active learning system that achieves high accuracy but requires human intervention for new patterns [27], and a cloud-based LLM system that achieves very high accuracy but depends on external model access [39]. The proposed system occupies a different point in this trade-off space by combining local deployment, zero-shot extraction, and multi-agent decomposition.

The study is guided by the following research questions:

- **RQ1:** Can a hierarchical three-stage agent architecture extract and structurally decompose address information from census-style documents spanning diverse address categories?
- **RQ2:** Does assigning specialized worker agents to specific extraction tasks reduce errors compared with a single-prompt LLM baseline using the same open-weight model?
- **RQ3:** Can a bounded feedback loop support autonomous recovery when the system encounters extraction failures or low-confidence outputs?
- **RQ4:** What are the practical trade-offs among the proposed multi-agent system, rule-based parsers, single-prompt LLM baselines, and published census parsing systems in terms of accuracy, speed, cost, and human involvement?
- **RQ5:** Can a locally deployed open-weight model provide useful extraction quality for census-style records while keeping sensitive data on-premise?

The importance of this work lies in the intersection of three requirements that are often treated separately: extraction quality, operational autonomy, and privacy-preserving deployment. Rule-based systems can run locally but require ongoing pattern engineering. Cloud-hosted LLM systems can achieve strong accuracy but may be difficult to use with sensitive census data. Single-prompt local LLM systems reduce privacy exposure but can struggle with long and heterogeneous documents. This paper investigates whether a multi-agent architecture can help bridge these gaps by distributing the extraction process across specialized agents while keeping all inference on local infrastructure.

The paper makes four main contributions:

1. **A hierarchical multi-agent architecture for census-style extraction.** We design a three-stage pipeline consisting of a Planner Agent, a Manager Agent, and specialized Worker Agents for extraction, validation, and formatting. The architecture uses structured communication between agents and is designed to reduce the burden placed on any single prompt.
2. **A bounded feedback mechanism for autonomous error recovery.** We introduce a feedback loop in which worker-level failures or low-confidence outputs are returned to the Planner Agent for strategy revision. This provides a mechanism for limited self-correction without requiring immediate human intervention.
3. **A privacy-preserving open-weight deployment model.** We demonstrate how gpt-oss-20b can be deployed locally for census-style structured extraction, allowing sensitive records to be processed without sending data to an external cloud service.
4. **An empirical comparison across diverse address categories.** We evaluate the proposed system on 700 synthetic records across seven address categories and compare it with a rule-based parser, a single-prompt LLM baseline, and two published UALR census parsing systems. This comparison highlights the accuracy, autonomy, privacy, and runtime trade-offs of the proposed approach.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

This work should be interpreted as a prototype evaluation rather than a production-readiness claim. All evaluation records are synthetic, and no real census records or real personal information were used. Synthetic data allows controlled testing across address categories, but it cannot fully capture the spelling errors, cultural naming variation, missing fields, OCR artifacts, and formatting inconsistencies found in real census submissions. The reported accuracy should therefore be read as preliminary evidence that the proposed architecture is promising, not as proof of readiness for operational census deployment.

The remainder of the paper is organized as follows. Section 2 reviews prior work on address parsing, name parsing, autonomous agent systems, and census-specific parsing tools. Section 3 describes the proposed architecture in detail. Section 4 presents the implementation, including local model deployment and synthetic data generation. Section 5 reports the evaluation methodology and results. Section 6 discusses the findings, limitations, and trade-offs. Section 7 concludes the paper and outlines future work.

2. Related Work

Structured knowledge extraction from census-style records draws on several related research areas: rule-based and statistical parsing, neural information extraction, Large Language Models, autonomous multi-agent systems, and privacy-preserving deployment. This section reviews these areas with emphasis on the specific requirements of the present work: fine-grained decomposition of personal records, robustness across diverse address formats, reduced human intervention, and secure local processing.

2.1. Parsing Personal Information: From Rules to Neural Networks

Early approaches to structuring personal information relied on human-crafted rules and domain-specific pattern libraries. For names, early work showed that personal naming conventions vary substantially across cultural and linguistic groups, making simple dictionary or position-based rules difficult to generalize [5]. For addresses, postal standardization systems such as the United States Postal Service Coding Accuracy Support System (CASS) established practical standards for conventional formats, while open-source tools such as `usaddress` and `libpostal` extended address parsing and normalization to broader use cases [1]. Rule-based NLP frameworks such as GATE further demonstrated the value of modular processing pipelines for information extraction [11]. These systems are attractive because they are fast, deterministic, transparent, and easy to deploy locally. Their main limitation is brittleness: handcrafted rules can only recognize patterns that have already been anticipated, which becomes problematic when records include uncommon, incomplete, or non-standard formats [8].

Statistical sequence models were developed to reduce this dependence on manually specified rules. Conditional Random Fields, Hidden Markov Models, and transformation-based learning allowed systems to learn token-level patterns from labeled data rather than relying entirely on handcrafted templates [4,10,19]. These methods improved generalization and were later extended to address parsing across multiple countries and address conventions [9]. However, their performance remains strongly tied to the coverage and quality of the training corpus. If the labeled examples are dominated by standard street addresses, the resulting model may struggle with military addresses, rural routes, highway references, university departments, or attention-line records.

Neural models further improved sequence labeling and entity recognition. BiLSTM-CRF architectures showed that recurrent neural networks could capture dependencies across address tokens [36], while transformers and BERT-based models changed the broader landscape of Named Entity Recognition [12,21,42]. These methods are powerful for detecting entity spans, but many systems operate at a coarse level, identifying that a span is an address without decomposing it into the fine-grained components needed for census processing. This distinction is important because downstream applications such as geocoding, record linkage, and identity resolution depend not only on detecting an address, but also on correctly assigning street number, street name, unit, city, state, ZIP code, organization, attention line, and other components [16].

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

2.2. Large Language Models for Information Extraction

Large Language Models introduced a different approach to structured extraction. GPT-3 showed that sufficiently large language models could perform many tasks with zero or few examples, including information extraction tasks that previously required task-specific training data [6]. For census-style parsing, this capability is important because an LLM can use natural language instructions and contextual reasoning to interpret address formats that may not be explicitly represented in a rule base or training set.

Several lines of LLM research directly inform the present architecture. Chain-of-Thought prompting has been shown to improve performance on tasks requiring multi-step reasoning [43], which motivates the use of a Planner Agent for document analysis and strategy formation. Work on constrained decoding and structured generation has shown that schema adherence is a central challenge for LLM-based extraction [44,47]. Research on feedback, retrieval, and factual correction has also shown that model outputs can be improved when generation is combined with structured review or external signals [34]. Together, these findings suggest that LLM extraction systems should not rely only on a single unconstrained prompt, especially when outputs must be machine-readable and auditable.

Task decomposition is another important theme. Prompt programming frameworks such as DSPy show that complex LLM tasks can often be improved by decomposing them into smaller composable steps [18]. Work on code-oriented and format-aware LLMs similarly suggests that models can perform well when prompts clearly specify structured input and output formats [22]. These findings motivate the separation of planning, task assignment, extraction, validation, and formatting in the proposed system.

At the same time, LLM-based extraction has known limitations. Models can lose track of information in the middle of long contexts, a phenomenon known as "lost in the middle" [23]. They can also be sensitive to prompt ordering [24], and self-assessed confidence scores are useful but imperfect indicators of correctness [17]. These limitations are especially relevant for census-style records, where a system may need to process long lists of heterogeneous records containing multiple types of personal information. The proposed architecture addresses these concerns through chunk-level processing, specialized workers, schema-constrained prompts, confidence reporting, and bounded feedback.

2.3. Autonomous Multi-Agent Systems

The idea that complex tasks can be improved by distributing work among specialized agents has a long history in artificial intelligence. Classical agent theory defines intelligent agents in terms of autonomy, reactivity, proactivity, and social ability [45], while broader AI frameworks distinguish deliberative, reactive, and hybrid agent architectures [35]. Multi-agent systems further show that teams of specialized agents can solve problems that are difficult for a single monolithic system to handle efficiently [15]. This principle is directly relevant to census extraction, where document analysis, address parsing, name extraction, validation, and formatting require different kinds of reasoning.

Recent LLM-based agent frameworks have made these ideas practical. ReAct demonstrated that reasoning and action can be interleaved to improve agent behavior [48]. AutoGen supported collaborative conversations among agents with different roles [46]. CrewAI introduced role-based hierarchical agent teams [32], and LangGraph provided graph-based orchestration for complex agent workflows [20]. These systems show that LLM-powered agents can be coordinated through roles, dependencies, and structured workflows rather than being used only as isolated prompt-response models.

Prior work on entity-level tasks has also shown that specialization can improve performance. A LangGraph-based entity resolution framework with multiple task-specific agents achieved strong accuracy on name variation matching while reducing API calls compared with a single-LLM baseline [3]. This supports the design decision in the present work to assign separate responsibilities to address extraction, name extraction, validation, and formatting agents rather than asking one general-purpose prompt to perform all tasks simultaneously.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

Self-correction and feedback mechanisms provide another foundation for the proposed architecture. Reflexion showed that agents can improve through verbal self-reflection [37], and Self-Refine demonstrated that LLM outputs can be iteratively improved using structured feedback [25]. The present system adapts these ideas to a multi-agent setting: when a downstream worker fails or produces low-confidence output, the error context is returned to the Planner Agent, which revises the strategy for a bounded number of retries. This differs from single-agent self-correction because the corrective reasoning is performed by an agent with a broader view of the document and extraction plan.

2.4. Census-Specific Parsing and Privacy-Preserving Deployment

The work most directly related to the present study concerns census-specific parsing systems and privacy-preserving AI deployment. Pattern-based census parsing has shown that high accuracy is possible when expert knowledge is encoded into reusable mappings. A prior active-learning system for name and address parsing tokenized input strings, created token masks, and mapped those masks to functional categories through user-defined mappings [27]. When an input did not match an existing mapping, it was routed to a domain expert for correction. This approach achieved 99.34% overall accuracy on a diverse corpus that included military, highway, and attention-line addresses. Its strength is that expert corrections permanently enrich the pattern base. Its limitation is that novel formats require manual analysis and rule extension, creating operational overhead as new patterns appear.

The semantic component definitions from that line of work remain useful for the present study. What changes is the assignment mechanism. Rather than relying on stored token-mask mappings, the proposed architecture delegates token assignment to contextual reasoning performed by specialized worker agents. This shift is intended to reduce the need for format-specific manual rule engineering while preserving fine-grained component decomposition.

Cloud-hosted LLM systems have demonstrated the other side of the trade-off. A prompt-driven, validation-centered census address parsing framework using Claude 4.0 Sonnet through AWS Bedrock achieved 99.8% exact-row accuracy on a 1,500-record corpus [39]. This result shows that commercial LLMs can be highly effective for diverse address parsing without manually crafted rules. However, cloud-hosted inference raises governance concerns for sensitive census data, especially when records contain personally identifiable information protected by strict legal and institutional requirements. In addition, this prior system used a single-pipeline design rather than a multi-agent architecture with inter-agent feedback.

Related work from entity resolution, household movement detection, multilingual record linkage, and multi-model extraction further supports the use of LLM reasoning for personal and organizational information processing. LLM-based systems have been used to detect co-residence patterns from unstandardized names and addresses [28], compare multiple LLMs for household movement discovery [30], perform cross-lingual entity resolution [31], and coordinate multiple models for industrial part specification extraction [29]. These studies indicate that LLMs can support complex knowledge extraction tasks, but they do not directly resolve the combined requirements of fine-grained census address decomposition, local deployment, autonomous recovery, and comparison against census-specific baselines.

Privacy-preserving deployment is therefore central to this paper. Open-weight language models, including the LLaMA family, Mistral mixture-of-experts models, and gpt-oss-20b, have made it increasingly feasible to run capable models on local infrastructure [33,40]. Other privacy-preserving methods, including federated learning and differential privacy, provide formal mechanisms for distributed or statistical privacy protection [13,26]. Policy-aware generative AI controllers have also shown how LLM reasoning can be constrained by explicit governance rules in regulated environments [2]. The present study adopts the most direct deployment strategy for sensitive census-style extraction: all model inference is performed on-premise, so input records are not transmitted to an external model provider.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

2.5. Summary and Research Gaps

The literature shows that existing approaches address different parts of the census extraction problem, but few address the full set of requirements together. Rule-based and pattern-based systems can be accurate and locally deployable, but they often require continuing expert involvement when new formats appear. Cloud-hosted LLM systems can achieve strong zero-shot accuracy, but they introduce governance concerns for sensitive records. Single-prompt local LLM systems reduce privacy exposure, but they can struggle with long, heterogeneous documents. Multi-agent LLM architectures provide a promising way to distribute complex extraction tasks, but their use for fine-grained census-style address decomposition remains underexplored.

Table 1 summarizes how the main research streams relate to the proposed system.

Table 1. Positioning of the proposed system relative to prior approaches.

Approach	Strength	Limitation	Relation to this work
Rule-based parsing	Fast, deterministic, transparent, and locally deployable	Brittle when formats differ from predefined rules	Used as a baseline and motivation
Statistical and neural parsing	Learns patterns from data and improves generalization	Requires representative labeled data and may focus on coarse entity spans	Motivates fine-grained component evaluation
Pattern-based active learning	High accuracy through accumulated expert mappings	Requires human intervention for new patterns	Compared as a census-specific prior system
Cloud-hosted LLM parsing	Strong zero-shot performance on diverse formats	Raises data governance concerns for sensitive records	Compared as a high-accuracy LLM prior system
Single-prompt local LLM extraction	Simple architecture with local deployment	Can skip records or conflate fields in long heterogeneous inputs	Used as an LLM baseline
Multi-agent LLM systems	Supports specialization, coordination, and feedback	Often evaluated outside fine-grained census parsing	Provides the architectural basis for this study
Proposed system	Combines local deployment, specialized agents, structured output, and bounded recovery	Evaluated on synthetic data and incurs higher latency than simpler baselines	Main contribution of this paper

To the best of our knowledge, the literature has not yet provided a systematic evaluation of a locally deployed open-weight multi-agent LLM architecture for fine-grained census-style address parsing across diverse address categories. Five gaps motivate the proposed system:

1. **Fine-grained census-style decomposition.** Existing systems do not fully address the problem of decomposing diverse address formats into detailed components across standard, individual, university, military, rural route, highway, and attention-line categories.
2. **Joint treatment of privacy and extraction quality.** Prior work often treats local deployment and high-quality extraction as separate goals rather than evaluating them together in one architecture.
3. **Reduced dependence on manual pattern engineering.** Pattern-based systems can achieve high accuracy, but new formats may require expert intervention. More autonomous alternatives remain needed.
4. **Inter-agent recovery for structured extraction.** Feedback and self-correction have been studied in LLM systems, but cooperative recovery across planning and worker agents remains underexplored for structured census-style extraction.
5. **Practical trade-off analysis.** The accuracy, latency, cost, privacy, and human-involvement trade-offs among rule-based parsers, single-prompt LLM baselines, cloud-hosted LLM systems, and multi-agent local LLM systems have not been sufficiently characterized.

The system presented in this paper is designed to address these gaps by combining fine-grained structured extraction, local open-weight model deployment, specialized multi-agent decomposition, bounded feedback, and empirical comparison against both general and census-specific baselines.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

3. Methodology and System Architecture

This section describes the methodological foundations, design principles, and system architecture of the proposed multi-agent framework for structured knowledge extraction from census-style records. The system receives a raw document containing one or more personal records and produces structured outputs containing extracted personally identifiable information (PII), decomposed address components, validation results, and an execution log. The architecture separates the extraction process into three stages: planning, coordination, and execution. These stages are connected by a bounded feedback mechanism that supports limited autonomous error recovery while preventing unbounded retry cycles.

3.1. Methodological Foundations

The proposed architecture is built on five methodological choices: multi-agent decomposition, local LLM-based reasoning, planning with Chain-of-Thought prompting, schema-constrained structured output, and bounded feedback. Each choice addresses a specific challenge in census-style extraction.

First, the system uses *multi-agent decomposition*. A Multi-Agent System (MAS) consists of multiple interacting agents that coordinate to solve problems that may be difficult for a single agent to handle alone [15,45]. Census-style extraction is naturally decomposable because address parsing, name extraction, phone number recognition, validation, and formatting require different forms of reasoning. Assigning all of these responsibilities to a single prompt increases cognitive load and can lead to skipped records, field conflation, or inconsistent outputs. Prior work on cognitive load and LLM task decomposition motivates the use of smaller, specialized tasks rather than a single monolithic prompt [18,38]. The proposed system operationalizes this principle through a three-level hierarchy: a Planner Agent for strategic reasoning, a Manager Agent for task coordination, and a fleet of Worker Agents for extraction, validation, and formatting [35].

Second, the system uses *Large Language Models as local reasoning engines*. All LLM-based components run on gpt-oss-20b, a 21-billion-parameter Mixture-of-Experts model with 3.6 billion active parameters per inference call [33]. This model is used because it supports instruction-following, structured reasoning over text, and on-premise deployment. These properties are important for census-style records, where the system must interpret heterogeneous formats while avoiding transmission of sensitive records to an external model provider. The model's open-weight availability, efficient inference profile, and long context window make it suitable for local deployment in the proposed pipeline.

Third, the Planner Agent uses *Chain-of-Thought prompting* to support document-level analysis. Chain-of-Thought prompting has been shown to improve performance on tasks requiring multi-step reasoning [43]. In this system, the Planner must identify the document format, estimate record structure, detect relevant PII types, recognize address categories, and formulate extraction guidance. These operations require broader reasoning than the worker-level extraction tasks, so Chain-of-Thought prompting is applied only at the planning stage.

Fourth, all LLM agents use *schema-constrained prompt engineering*. Each agent receives a role specification, task instructions, and an explicit JSON output example. This design encourages outputs that conform to predefined schemas and supports deterministic downstream parsing [34,47]. Because LLMs may still produce extraneous text, markdown wrappers, or malformed JSON, schema prompting is supplemented with post-hoc JSON extraction and fallback logic, described in Section 4.5.

Fifth, the architecture includes *bounded recursive feedback*. Self-correction methods have shown that LLM outputs can improve when models receive feedback and revise their responses [25,37]. The proposed system adapts this idea to a multi-agent setting. When a Worker Agent produces invalid output, fails schema validation, or is flagged by validation rules, the error context is returned to the Planner Agent. The Planner then revises the extraction strategy and the worker retries the task. This process is bounded by a maximum of $K_{\max} = 2$ retries, which prevents infinite recursion and keeps runtime predictable. The feedback loop is non-persistent: it can improve recovery within a run, but it does not permanently update the model weights, prompts, or rule base.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

3.2. Design Principles

Four design principles govern the architecture. Each principle addresses a practical risk associated with LLM-based structured extraction.

P1: Separation of Concerns. Strategic reasoning, task coordination, operational extraction, validation, and formatting are handled by distinct components. This prevents a single agent from simultaneously analyzing document structure, extracting multiple PII types, validating outputs, and formatting final records. The design directly addresses the quality degradation observed when long heterogeneous extraction tasks are handled by a single prompt [23].

P2: Bounded Cognition. Each Worker Agent is assigned a narrow responsibility. For example, the address worker handles address decomposition, the name worker handles name extraction, and the formatter handles output assembly. Limiting each worker's scope reduces prompt complexity and helps mitigate cognitive overload [23,38].

P3: Graceful Degradation. Every LLM-dependent component is paired with deterministic fallback behavior. If the Planner Agent fails to produce a valid plan, a default plan is substituted. If the Manager Agent fails to generate a valid task graph, a minimum viable graph is constructed from detected PII types. If a Worker Agent fails after exhausting feedback retries, the failure is logged and the pipeline continues with partial results. This ensures that a single failure does not halt the entire extraction process.

P4: Observability. All major actions are logged with timestamps, component identifiers, status codes, and execution metadata. The execution log supports debugging, reproducibility, runtime analysis, and future audit requirements for regulated census-style processing. In production settings, this logging must be implemented with appropriate safeguards so that sensitive information is not exposed through logs.

3.3. Architectural Overview

The system consists of five major components: a Document Processor, a Planner Agent, a Manager Agent, a Worker Agent Fleet, and an Orchestrator. The Planner, Manager, and Worker Agents are LLM-based. The Document Processor, Orchestrator, JSON extraction logic, fallback logic, and logging are deterministic. The validation stage is hybrid in design: the current system uses LLM-based validation, while production deployments should add deterministic checks such as regular-expression validation, state-ZIP consistency checks, and postal reference checks.

Figure 1 illustrates the architecture and data flow. A raw input document is first processed into text, format metadata, and chunks. The Planner Agent analyzes the document and produces an extraction plan. The Manager Agent converts that plan into a dependency-aware task graph. Worker Agents then execute extraction tasks over the chunks. Validation and formatting workers consolidate the results into final structured records. If a worker fails or produces output that violates configured validation rules, the error context is sent back to the Planner Agent for bounded re-planning.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

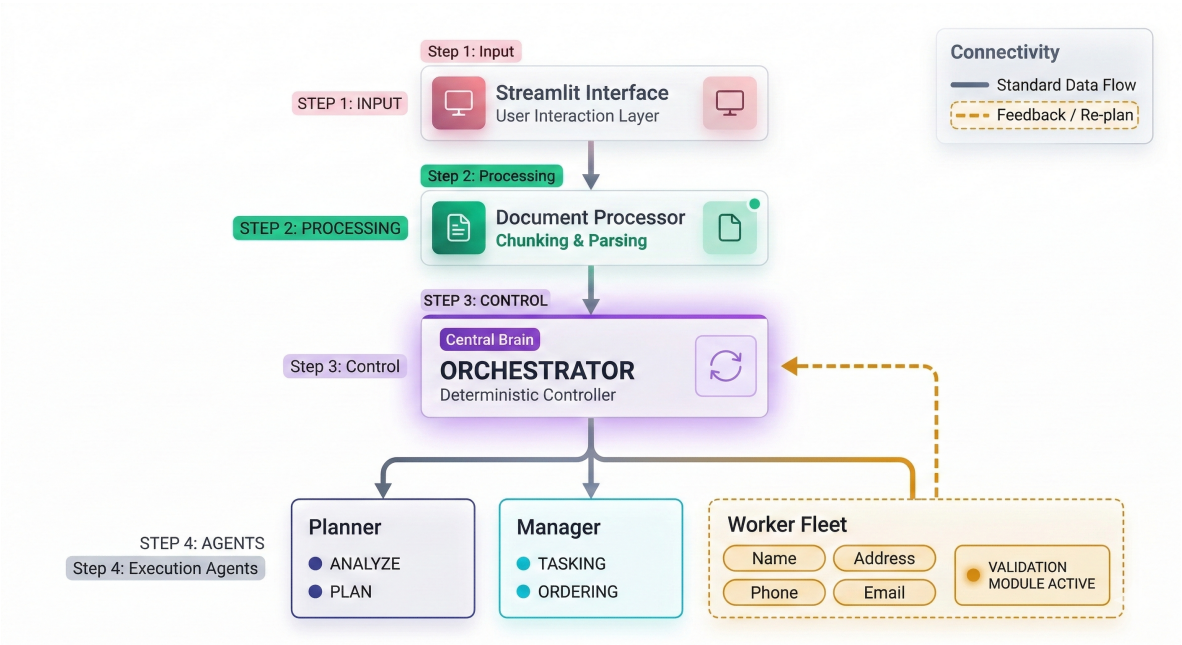

Figure 1. System architecture. Solid arrows indicate data flow; the dashed feedback arrow indicates the recursive re-planning path. The Orchestrator is a deterministic controller, not an LLM-powered agent.

Table 2 summarizes the role, type, input, and output of each major component.

Table 2. Summary of major system components.

Component	Type	Primary Input	Primary Output and Role
Document Processor	Deterministic	Raw document file or text	Extracted text, detected format, and record-preserving chunks
Planner Agent	LLM-based	Document sample, format metadata, and optional error context	Document analysis, detected PII types, address categories, and extraction strategy
Manager Agent	LLM-based	Planner output and chunk metadata	Dependency-aware task graph specifying extraction, validation, and formatting tasks
Worker Agent Fleet	LLM-based	Chunks, task instructions, and current plan	Extracted PII fields, decomposed addresses, validation results, and formatted records
Orchestrator	Deterministic	Document processor output, plan, task graph, and worker outputs	Ordered execution, fallback handling, feedback control, final structured records, and execution log

The system is described as autonomous in the sense of *reactive autonomy* [45]. It executes the complete pipeline from raw input to structured output without human intervention during a run, including limited error detection and recovery. However, it follows a fixed processing architecture rather than performing unrestricted goal-directed planning over arbitrary actions. All LLM inference occurs on a locally deployed gpt-oss-20b instance, so input records are not transmitted to an external model provider.

Stage 1: Planner Agent.

The Planner receives the raw document text sample, limited to the first 3,000 characters, together with the detected file format. The 3,000-character sample provides enough context to infer document structure while keeping the planning prompt compact and consistent across runs. Using Chain-of-Thought prompting, the Planner produces a structured JSON plan with three main sections: (i) an analysis section containing detected format, estimated record count, identified PII types, and identified address categories; (ii) a roadmap section specifying phased execution and inter-phase dependencies;

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

and (iii) an extraction strategy section providing guidance for the address categories and PII types detected in the document. If the Planner fails to produce valid JSON, a default plan is used. During error recovery, the Planner also receives an error context e_i and produces a revised plan.

Stage 2: Manager Agent.

The Manager receives the Planner output and converts it into a dependency-aware task graph.

Definition (Task Graph). Let $G = (T, D)$ be a directed acyclic graph where each task $t_i \in T$ is a tuple $(id_i, type_i, instructions_i, deps_i, priority_i)$. The tuple specifies a unique task identifier, the PII type or operation to perform, task-specific instructions derived from the Planner strategy, dependencies that must complete first, and task priority. An edge $(t_i, t_j) \in D$ indicates that task t_i must complete before task t_j begins.

In a typical run, the Manager produces six independent extraction tasks for addresses, names, phone numbers, email addresses, Social Security Numbers, and dates of birth. These are followed by a validation task that depends on all extraction tasks and a formatting task that depends on validation. If the Manager fails to produce a valid task graph, the Orchestrator constructs a deterministic minimum viable graph from the PII types identified by the Planner.

Stage 3: Worker Agent Fleet.

Eight specialized Worker Agents execute the tasks assigned by the Manager. These include workers for address extraction, name extraction, phone extraction, email extraction, Social Security Number extraction, date-of-birth extraction, validation, and formatting. All workers operate at temperature 0.1 for near-deterministic generation and use schema-constrained prompts. The following subsections describe the most architecturally significant workers.

3.3.1. Address Extraction Worker

The address extraction worker is the most complex worker in the fleet because census-style records include both standard postal formats and non-standard address structures. Its output schema contains fifteen fields, shown in Table 3. These fields were selected to cover conventional address components as well as census-relevant components such as organization names, attention lines, military city and state equivalents, and route classifications.

Table 3. Address extraction output schema with fifteen components.

Field	Type	Description
full_address	string	Complete address as found in source
street_number	string	Building/house number
street_name	string	Street, highway, or route name
street_type	string	Street suffix (St, Ave, Blvd, Highway)
pre_directional	string	Pre-directional (N, S, E, W)
post_directional	string	Post-directional
unit_type	string	Unit designator (Apt, Suite, Floor)
unit_number	string	Unit identifier
city	string	Municipality or APO/FPO/DPO
state	string	Two-letter state or AE/AP/AA
zip_code	string	Five-digit ZIP or military ZIP
organization	string	Organization name
attention_line	string	ATTN: content
route_type	string	Route classification
confidence	float	Self-assessed confidence $[0, 1]$

The worker prompt includes approximately 1,200 tokens of category-specific guidance covering the seven address categories evaluated in this study: standard street addresses, individual unit addresses, university department addresses, military APO/FPO/DPO addresses, rural route addresses,

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

highway-referenced addresses, and attention-line records. This guidance specifies, for example, that `APO`, `FPO`, and `DPO` should be mapped to the city field; that `AE`, `AP`, and `AA` are military state equivalents; that route numbers in highway addresses form part of the `street_name` rather than the `unit_number`; and that attention-line content should be separated from organization names. Because the worker relies on contextual reasoning rather than fixed token-mask rules, it can handle variations that are not explicitly enumerated in the prompt, although this capability is not a substitute for independent validation [27].

The confidence field contains a self-assessed value in $[0, 1]$. Such confidence scores are treated as soft signals rather than definitive correctness indicators because prior work has shown that model confidence is meaningful but imperfect [17]. In this system, confidence values are used to prioritize validation and to identify outputs that may require feedback or later human review.

3.3.2. Name Extraction Worker

The name extraction worker produces a seven-field output: `full_name`, `first_name`, `middle_name`, `last_name`, `prefix`, `suffix`, and `confidence`. The prompt includes guidance for compound surnames, hyphenated names, multiple middle names, single-name individuals, professional titles, and generational suffixes. Because personal names vary across cultural and linguistic groups, dictionary-based or position-based parsing can be insufficient [5]. The LLM-based worker provides broader contextual coverage, although systematic bias evaluation across cultural groups remains necessary, as discussed in Section 6.4.

3.3.3. Validation, Formatting, and Additional Workers

The validation worker checks extracted outputs for schema compliance, internal consistency, completeness, and category-specific constraints. Examples include verifying that ZIP codes have the expected format, military addresses use appropriate AE/AP/AA state equivalents, required record-level fields are present, and extracted fields are not obviously conflated. A failure can be triggered by invalid JSON, schema mismatch, missing required structures, an explicit worker error, or a validation rule violation. Low-confidence outputs are treated as review signals and may trigger feedback when configured validation rules classify them as unreliable.

The formatting worker merges outputs from the extraction and validation workers into unified per-record structures. It preserves document order, resolves field-level conflicts when possible, and produces the final structured output expected by downstream processing.

Four additional workers handle phone numbers, email addresses, Social Security Numbers, and dates of birth. Each follows the same schema-constrained prompt pattern: role declaration, output schema, example output, and extraction instructions tailored to the specific PII type.

LLM-based validation may share some error modes with LLM-based extraction because the same underlying model family is used. For production deployment, the validation stage should therefore be supplemented with deterministic checks such as regular expressions, state abbreviation validation, ZIP-code consistency checks, postal reference data, and secure audit rules.

3.4. Chunk-Level Processing

Documents exceeding 3,000 characters are segmented into chunks at natural line boundaries. The purpose of chunking is not to satisfy the model's maximum context length, but to reduce the risk of long-context degradation and to keep worker prompts focused. Whenever possible, chunk boundaries are selected so that individual records are not split across chunks. If a single record exceeds the chunk target, the system preserves the complete record as one chunk rather than splitting it internally.

Each Worker Agent processes chunks independently, and the outputs are merged in the original document order:

$$R = \bigoplus_{j=1}^{m} w(c_j) \tag{1}$$

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

where w denotes the worker function, c_j denotes chunk j, and $\oplus$ denotes ordered concatenation of structured result sets. This strategy serves two purposes. First, it mitigates the "lost in the middle" effect, where LLMs can show reduced accuracy for information positioned in the interior of long contexts [23]. Second, it provides error isolation: a failure on one chunk can be logged, retried, or partially recovered without invalidating all other chunks.

3.5. Feedback Loop and Error Recovery

When Worker Agent w_i fails during execution, the system activates the bounded feedback mechanism shown in Algorithm 1. In this context, failure includes invalid JSON, schema violation, missing required output structures, an explicit error object, or a hard validation failure. Low-confidence output may also trigger feedback when the configured validation rules mark it as unreliable. The feedback mechanism allows recovery within a run but does not permanently update the model, prompts, or stored knowledge.

Algorithm 1 Bounded Recursive Feedback

Require: Worker w_i, current plan P_{current}, chunks C, maximum retries $K_{\max} = 2$
Ensure: Result r_i and updated plan P_{current}

$k \leftarrow 0$
$r_i \leftarrow w_i(C, P_{\text{current}})$
while $r_i = \texttt{error}$ **and** $k < K_{\max}$ **do**
  $e_i \leftarrow \texttt{extract\_error\_context}(r_i)$
  $P_{\text{current}} \leftarrow \texttt{Planner.replan}(P_{\text{current}}, e_i)$
  $k \leftarrow k + 1$
  $r_i \leftarrow w_i(C, P_{\text{current}})$
end while
if $r_i = \texttt{error}$ **then**
  $\texttt{log\_failure}(w_i, e_i, k)$
  $r_i \leftarrow \texttt{fallback}(w_i, C)$
end if
return $(r_i, P_{\text{current}})$

The critical feature of this mechanism is that recovery occurs across agent boundaries. The failure is detected at the worker or validation stage, but the corrective reasoning is performed by the Planner Agent, which has a broader view of the document and extraction strategy. This differs from single-agent self-reflection approaches, where the same model instance attempts to revise its own output [25,37]. It also differs from pattern-based active learning systems, where unresolved cases are routed to a human expert for new rule creation [27]. The trade-off is that automatic recovery operates at machine speed but is not persistent across runs; the same failure may recur in a future execution unless the prompts, validation rules, or fallback logic are updated.

3.6. Orchestrator

The Orchestrator is a deterministic Python controller, not an LLM-powered agent. It manages document processing, planning, task-graph construction, worker execution, feedback, fallback behavior, and final output compilation. Algorithm 2 specifies the complete pipeline.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

Algorithm 2 Orchestrator Pipeline

Require: Raw document d
Ensure: Structured records R_{final}, execution log $\mathcal{L}$
$(text, format, chunks) \leftarrow \texttt{DocumentProcessor}(d)$
$P_{\text{current}} \leftarrow \texttt{PlannerAgent}(text_{[:3000]}, format)$
if $P_{\text{current}} = \texttt{invalid}$ **then**
    $P_{\text{current}} \leftarrow \texttt{default\_plan}(format)$
end if
$G = (T, D) \leftarrow \texttt{ManagerAgent}(P_{\text{current}}, |chunks|)$
if $G = \texttt{invalid}$ **then**
    $G \leftarrow \texttt{default\_graph}(P_{\text{current}})$
end if
for each $t_i \in \texttt{TopologicalSort}(G)$ **do**
    wait for all $t_j \in deps(t_i)$ to complete
    $(r_i, P_{\text{current}}) \leftarrow \texttt{ExecuteWithFeedback}(w_{type(t_i)}, P_{\text{current}}, chunks)$ ▷ Algorithm 1
    $\mathcal{L} \leftarrow \mathcal{L} \cup \{(t_i, r_i, \texttt{timestamp})\}$
end for
$R_{\text{final}} \leftarrow \texttt{Compile}(\{r_1, \ldots, r_n\})$
return $(R_{\text{final}}, \mathcal{L})$

The Orchestrator enforces the topological ordering of the task graph. Extraction tasks can execute before validation, validation must complete before formatting, and formatting produces the final structured records. The execution log $\mathcal{L}$ records each component invocation, status, output summary, and wall-clock time. This supports debugging and runtime analysis while also establishing the basis for auditable operation in future regulated deployments.

3.7. Design Alternatives and Rationale

Several architectural alternatives were considered during system design. For inter-agent communication, JSON contracts were chosen over natural language messaging because JSON supports deterministic parsing, schema validation, and repeatable downstream processing [46]. The Manager was implemented as an LLM-powered agent rather than a hardcoded task graph so that task assignment could adapt to the PII types and address categories detected in the input document. Worker Agents were specialized by PII type rather than by address category because PII types are stable across datasets, while address categories may vary or overlap.

For error recovery, inter-agent feedback was preferred over single-agent self-reflection because the Planner Agent has broader document-level context than an individual worker [37]. For structured output, post-hoc JSON extraction was selected instead of constrained decoding because it is model-agnostic and compatible with common local serving frameworks [44]. Finally, the feedback loop was bounded at $K_{\max} = 2$ to balance recovery capability against predictable runtime and to avoid unbounded recursion.

4. Implementation

This section describes the implementation of the proposed multi-agent extraction system, including the hardware and software stack, local model deployment, runtime configuration, prompt design, JSON recovery mechanism, and synthetic data generation process. The implementation was designed around three practical requirements: the system should run on local infrastructure, produce repeatable experimental outputs, and avoid the use of real census records or real personally identifiable information during evaluation.

4.1. Technology Stack and Hardware Configuration

Table 4 summarizes the technology stack and hardware configuration used throughout the experiments.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

Table 4. Technology stack and hardware configuration.

Component	Technology	Purpose
Processor	Intel Core Ultra 9 285HX (24 cores, up to 5.5 GHz)	System computation, preprocessing, orchestration, and output handling
Graphics	NVIDIA RTX PRO 5000 Blackwell (24 GB GDDR7)	Local GPU inference for all LLM-based agents
LLM Model	gpt-oss-20b (21B parameters, MoE architecture)	Language understanding, planning, extraction, validation, and structured generation
LLM Serving	vLLM 0.4+	Local inference server exposed through a host-restricted HTTP endpoint
Web Framework	Streamlit 1.32+	Interactive user interface for document upload, execution, and result review
Data Processing	pandas 2.0+	Tabular data manipulation, record assembly, and output formatting
Language	Python 3.10+	Implementation language for orchestration, parsing, validation, and evaluation scripts

The hardware configuration was selected to test whether the full multi-agent pipeline could run on professional workstation hardware rather than requiring cloud-hosted models or data-center GPU infrastructure. Prior LLM-based census parsing work has relied on cloud-hosted commercial models [39]. In contrast, this implementation runs the complete pipeline on a local workstation equipped with a 24 GB GPU. This design lowers the practical barrier for organizations that require local processing but may not have access to large-scale compute clusters.

For reproducibility, the implementation records the model configuration, quantization setting, prompt templates, inference parameters, execution logs, and output files for each run. In a public release or camera-ready version, these records should be accompanied by the exact model checkpoint identifier, quantized artifact identifier, package versions, random seeds used for synthetic data generation, and evaluation scripts.

4.2. On-Premise Model Deployment

The gpt-oss-20b model is deployed locally using a 4-bit GPTQ quantized configuration. Quantization reduces the approximate memory footprint from 42 GB at half precision (FP16) to approximately 12 GB, which fits within the 24 GB memory capacity of the NVIDIA RTX PRO 5000 Blackwell GPU. The remaining memory is used for inference overhead and key-value cache storage during generation. This configuration supports the prompt templates and document chunks used in the evaluation while preserving the model's long-context capability.

The model is served through vLLM as a local HTTP endpoint bound to the host machine. The endpoint is not exposed to external traffic, and the experimental pipeline does not transmit input records to any external model provider. This deployment provides a strong operational privacy posture because inference occurs inside the local execution environment. However, local deployment alone is not a complete security model. Production use with real census data would also require access controls, encrypted storage, secure deletion policies, log redaction, endpoint hardening, audit procedures, and governance review.

4.3. Runtime Configuration

All LLM-based agents use the same local model endpoint. Inference calls are configured with a temperature of 0.1 to encourage near-deterministic outputs while still allowing the model to generate

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

complete structured responses. The system applies exponential backoff retry logic with a maximum of three retries for transient inference failures, such as local timeout or temporary server unavailability.

The runtime pipeline is controlled by a deterministic Python Orchestrator. The Orchestrator manages document preprocessing, Planner invocation, Manager invocation, Worker Agent execution, feedback-loop retries, fallback behavior, and final output compilation. It also records execution metadata, including component name, task identifier, status, retry count, and wall-clock time. Because execution logs may contain sensitive values in production settings, the prototype logging design should be paired with redaction or secure logging before use on real records.

4.4. Prompt Engineering

Each agent receives a structured system prompt designed to constrain the model's role, scope, and output format. The prompts follow a common template, summarized in Table 5.

Table 5. Common prompt structure used across LLM-based agents.

Prompt Component	**Purpose**
Role declaration	Defines the agent's responsibility, such as planning, task coordination, address extraction, validation, or formatting
Scope restriction	Limits the agent to a specific task so that it does not attempt unrelated extraction or reasoning
JSON-only instruction	Directs the model to return machine-readable structured output without explanatory prose
Schema example	Provides a complete example of the expected JSON structure and representative field values
Task-specific guidance	Supplies extraction rules, category-specific guidance, validation checks, or formatting instructions
Failure behavior	Instructs the agent to return empty fields or explicit error indicators when extraction is uncertain or impossible

The address extraction worker receives the most detailed prompt because it must handle the widest range of structural variation. Its task-specific instructions contain approximately 1,200 tokens of guidance covering the seven address categories used in the evaluation: standard, individual, university, military, rural route, highway, and attention-line addresses. The guidance specifies, for example, that `APO`, `FPO`, and `DPO` should be treated as city-equivalent values; that `AE`, `AP`, and `AA` should be treated as military state equivalents; that highway route numbers belong with the street or route name rather than the unit number; and that attention-line content should be separated from organization names.

The Planner Agent uses a different prompt pattern. In addition to the JSON schema, it receives Chain-of-Thought style instructions for document analysis. The Planner is asked to infer the document structure, identify likely PII types, recognize address categories, and produce an extraction strategy. Worker Agents do not receive the same broad reasoning instructions because their tasks are intentionally narrower.

4.5. Robust JSON Extraction

Although agents are instructed to produce JSON-only output, LLM responses are not always syntactically valid JSON. In preliminary runs, approximately 18% of responses included markdown code fences, explanatory preambles, trailing commentary, or other extraneous content. To prevent these responses from halting the pipeline, the implementation uses a multi-stage JSON extraction function, `safe_json()`.

The `safe_json()` function applies the following recovery sequence:

1. Remove markdown code fences, language identifiers, and leading or trailing whitespace.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

2. Attempt standard parsing with `json.loads()`.
3. If parsing fails, locate the first opening brace `{` and the last balanced closing brace `}`, extract the candidate JSON substring, and attempt parsing again.
4. If parsing still fails, return a default dictionary matching the expected schema, with empty fields or explicit error indicators.
5. Log the parsing failure, raw response summary, recovery step, and fallback status for later inspection.

This post-hoc extraction strategy is model-agnostic and does not require changes to the inference engine. It was therefore preferred over constrained decoding in the prototype implementation, although constrained decoding remains a useful alternative for future systems that require stronger output guarantees [44]. The fallback dictionary ensures graceful degradation: the pipeline can continue even when one agent response cannot be parsed, while the execution log preserves evidence of the failure.

4.6. Synthetic Data Generation

The evaluation dataset contains 700 synthetically generated census-style records. The dataset was created programmatically in Python using stratified sampling across seven address categories, with 100 records per category. Each record was first generated as a structured object containing ground-truth fields, then rendered into textual record formats with controlled variation in punctuation, abbreviation style, line breaks, field order, and optional components. This design allows the system output to be compared against known component-level labels.

Table 6 summarizes the seven address categories and the main variations included in each category.

Table 6. Synthetic data categories and generated variations.

Category	Records	Generated Variations
Standard	100	Conventional street addresses sampled from 50 U.S. cities across 30 states, with varied street types and directional prefixes
Individual	100	Apartment, suite, floor, unit, and # designators appearing before the street address, after the street address, or on separate lines
University	100	Department names, building names, room numbers, and campus mail codes interleaved with standard postal components
Military	100	APO, FPO, and DPO addresses with PSC, CMR, and Unit box constructions; AE, AP, and AA state equivalents; military-range ZIP codes
Rural Route	100	RR and HC route designations with associated box numbers and rural addressing conventions
Highway	100	U.S. highways, state highways, county roads, and farm-to-market roads where route numbers form part of the street or route name
Attention-Line	100	ATTN prefixes, organization names, care-of style routing, and selected records with no physical street address

Each synthetic record also includes a generated personal name, phone number, email address, Social Security Number, and date of birth. Name generation includes variation in prefixes, suffixes, middle names, compound names, and hyphenated forms. Phone numbers, email addresses, Social Security Numbers, and dates of birth are generated to resemble valid formats without corresponding to real individuals.

All data used in the evaluation is fictitious. No real census records, real addresses, real Social Security Numbers, or real personal information were used at any stage of this research. The use of synthetic data allows controlled testing across address categories and supports repeatable evaluation,

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

but it also limits external validity. Real census submissions may contain spelling errors, missing fields, OCR artifacts, cultural naming variation, inconsistent punctuation, and unexpected formatting that are not fully captured by synthetic generation. The generation script and complete dataset are available upon request from the UALR ERIQ Center, subject to data governance review.

5. Evaluation

This section evaluates the proposed multi-agent system from four perspectives: extraction accuracy, record completeness, runtime cost, and deployment trade-offs. The evaluation is designed to answer the research questions stated in Section 1 while keeping the scope of the evidence clear. All experiments use synthetic census-style records. Therefore, the results should be interpreted as preliminary evidence of architectural feasibility rather than as production-readiness certification.

5.1. Evaluation Goals and Scope

The evaluation has five goals. First, it tests whether the proposed architecture can decompose diverse census-style address records into fine-grained components. Second, it compares the multi-agent architecture against a single-prompt LLM baseline using the same underlying model. Third, it observes whether the bounded feedback loop can recover from selected extraction failures. Fourth, it characterizes the trade-off between accuracy and computational cost. Fifth, it positions the proposed system relative to prior census parsing systems with different assumptions about privacy, human involvement, and deployment model.

Table 7 summarizes how the evaluation addresses each research question.

Table 7. Research questions and corresponding evaluation evidence.

Research Question	Evaluation Evidence
RQ1	Address component-level accuracy across seven census-style address categories
RQ2	Direct comparison between the multi-agent system and a single-prompt gpt-oss-20b baseline
RQ3	Feedback-loop trigger events, recovery outcomes, and retry behavior
RQ4	Accuracy, record recall, structural completeness, LLM call count, runtime, and contextual comparison with prior systems
RQ5	Local deployment of gpt-oss-20b and evaluation of extraction quality under an on-premise model configuration

Several limitations should be acknowledged before presenting the results. First, all evaluation data is synthetically generated. Although the generation process includes realistic formatting variation and deliberate edge cases, synthetic records cannot fully reproduce spelling errors, missing fields, OCR artifacts, cultural naming variation, and unexpected formatting found in real census submissions. Second, the ground-truth labels were derived from the same generation process that produced the records, with manual inspection by the authors. This creates potential circularity that independent annotation would reduce. Third, only one open-weight LLM, gpt-oss-20b, was evaluated. Fourth, each condition was executed three times, which provides a preliminary estimate of stability but is not sufficient for formal statistical significance testing. Fifth, the multi-agent system uses substantially more LLM calls than the single-prompt baseline, so the evaluation should be interpreted as a comparison between architectural robustness and computational cost, not only as a pure model-capability comparison.

5.2. Evaluation Dataset and Experimental Conditions

The evaluation dataset is identical to the synthetic dataset described in Section 4.6. It contains 700 records distributed equally across seven address categories. The category selection follows the

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

same taxonomy used in prior census parsing work [27], which supports comparison with established address-format categories.

Table 8. Evaluation dataset: seven address categories with 100 records each.

Category	Records	Key Parsing Challenges
Standard	100	Street number, name, type, city, state, ZIP
Individual	100	Unit designators with varied placement
University	100	Department and building names interleaved with postal components
Military	100	APO/FPO/DPO as city; AE/AP/AA as state; military ZIP ranges
Rural Route	100	RR and HC route designations with box numbers
Highway	100	Route numbers embedded in street names
Attention-Line	100	ATTN prefix; organizational names; selected records without physical address
Total	**700**	

Three systems were evaluated on the same 700-record dataset:

1. **Rule-Based Parser.** A deterministic baseline using the Python `usaddress` library for address parsing, `nameparser` for name parsing, and regular expressions for phone numbers, email addresses, Social Security Numbers, and dates of birth.
2. **Single-Prompt LLM.** A gpt-oss-20b baseline using one comprehensive prompt that includes instructions for all PII types and all address categories. This baseline uses the same model as the proposed system but does not use planning, task decomposition, specialized workers, validation workers, formatting workers, or inter-agent feedback.
3. **Multi-Agent System.** The proposed architecture using the Planner Agent, Manager Agent, specialized Worker Agents, validation, formatting, and bounded feedback.

The single-prompt baseline is important because it helps isolate the contribution of the architecture. Since both LLM-based systems use gpt-oss-20b, differences between them are attributable primarily to prompt organization, task decomposition, feedback, and orchestration. However, the comparison is not compute-neutral: the multi-agent system uses more inference calls to obtain greater completeness and robustness.

5.3. Evaluation Metrics

Four metrics are used to evaluate performance:

1. **Address Component-Level Exact Match (EM).** This is the primary metric. It measures the proportion of address component fields that exactly match the corresponding ground-truth value. The score is micro-averaged across records and address fields. Missing predictions are counted as incorrect for non-empty ground-truth fields. Empty fields are counted as correct only when the corresponding ground-truth field is also empty.
2. **Record-Level Recall.** This measures the proportion of input records for which the system produces any output. A system that skips records receives a lower recall score even if it performs well on the records it processes.
3. **Structural Completeness.** This measures the average proportion of applicable non-empty ground-truth fields that receive non-empty predicted values. This metric reduces inflation from fields that are empty by design.
4. **Pipeline Completion Rate.** This measures the proportion of Worker Agents that complete execution without unrecoverable failure. This metric applies only to the multi-agent system.

All reported results are averaged over three independent runs. Across runs, standard deviations did not exceed 1.2 percentage points for any reported accuracy metric. This indicates preliminary stability, but the number of runs is not sufficient for formal significance testing.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

5.4. Main Results

Table 9 reports address component-level exact match accuracy by category for the rule-based parser, the single-prompt LLM baseline, and the proposed multi-agent system.

Table 9. Address component-level exact match accuracy (%) by category, averaged over three runs (SD $\leq$ 1.2 percentage points).

Category	Rule-Based	Single-Prompt	Multi-Agent
Standard	96.0	94.0	100.0
Individual	85.0	90.0	99.0
University	68.0	84.0	99.0
Military	28.0	70.0	91.0
Rural Route	56.0	80.0	98.0
Highway	11.0	74.0	93.0
Attention-Line	22.0	74.0	90.0
Overall	**52.3**	**80.9**	**95.7**

The results show three main patterns. First, the rule-based parser performs well on standard addresses but degrades sharply on non-standard categories. Its accuracy falls to 28.0% on military addresses, 22.0% on attention-line records, and 11.0% on highway addresses. This pattern is consistent with the known limitation of rule-based systems: they perform well when inputs match anticipated formats but struggle when records contain structures outside their rule coverage [8,27].

Second, the single-prompt LLM baseline substantially improves over the rule-based parser on every non-standard category. This confirms that the open-weight LLM has useful zero-shot generalization capability for census-style address parsing. However, the single-prompt baseline still struggles with long heterogeneous inputs. It skipped 47 of 700 records and produced field-conflation errors, including organization names placed into street fields and route numbers split across unrelated components. These errors are consistent with long-context and cognitive-overload concerns described in prior work [23,38].

Third, the multi-agent system achieves at least 90% accuracy in every category and 95.7% overall accuracy. The largest improvements over the single-prompt baseline occur in the most difficult categories: military addresses, highway addresses, and attention-line records. This supports the central hypothesis of the paper: distributing extraction across specialized agents can improve robustness when documents contain diverse address structures and multiple PII types.

5.5. Overall Pipeline Metrics and Runtime Trade-Offs

Table 10 presents aggregate pipeline metrics across all 700 records.

Table 10. Overall pipeline metrics, averaged over three runs.

Metric	Rule-Based	Single-Prompt	Multi-Agent
Record-Level Recall	692/700 (98.9%)	653/700 (93.3%)	700/700 (100%)
Structural Completeness	0.58	0.78	0.94
Pipeline Completion Rate	N/A	N/A	8/8 (100%)
Total LLM Calls	0	47	∼292
Wall-Clock Time	<5 s	3 to 4 min	20 to 25 min
Records Skipped	8	47	0

The multi-agent system achieves 100% record-level recall, meaning that every input record produces an output. This is important for census-style workflows, where skipped records may translate into missing individuals or households in downstream processing. The rule-based parser skipped 8 records, while the single-prompt LLM skipped 47 records, mostly from longer heterogeneous batches.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

The multi-agent system also achieves the highest structural completeness score, 0.94, compared with 0.78 for the single-prompt baseline and 0.58 for the rule-based parser. This indicates that the multi-agent system more consistently fills applicable non-empty fields. The difference between 95.7% exact match and 0.94 structural completeness reflects cases where the system produced a value for a field but the value did not exactly match the ground truth.

The improvement comes with a computational cost. The multi-agent system requires approximately 292 LLM calls and 20 to 25 minutes of wall-clock time on the local GPU, compared with 47 LLM calls and 3 to 4 minutes for the single-prompt baseline. The rule-based parser is much faster, completing in less than 5 seconds, but its accuracy drops sharply on non-standard address categories. These results show a practical trade-off: the multi-agent system improves accuracy and completeness but increases inference cost and latency. For batch census-style processing, this latency may be acceptable. For interactive use, the system would require optimization through parallel worker execution, prompt compression, batching, or model-serving improvements.

5.6. Feedback Loop Analysis

The bounded feedback loop was triggered three times across the three experimental runs. Table 11 summarizes each trigger event, the failure type, and the observed resolution.

Table 11. Observed feedback-loop trigger events during evaluation.

#	Category	Worker	Failure and Resolution
1	Military	Address	`APO` assigned to `street_name`. The revised plan added explicit guidance that `APO`, `FPO`, and `DPO` are city-equivalent designations. Retry succeeded.
2	Attention-Line	Address	Entire attention-line content placed in `street_name`. The revised plan separated the `ATTN` prefix from organization name and physical address. Retry succeeded.
3	Highway	Address	`State Highway 16` split across `street_name`, `street_type`, and `unit_number`. The revised plan specified that route numbers are integral to `street_name`. Retry succeeded.

All three observed failures were resolved on the first re-planning attempt, so none reached the maximum retry bound of $K_{\max} = 2$. Each re-planning cycle added approximately 10 seconds of processing time. The observed triggers occurred in the same address categories that were most difficult for the baselines: military, attention-line, and highway addresses.

These results should be interpreted cautiously. Three trigger events are useful as observed recovery examples, but they are not enough to estimate feedback-loop reliability statistically. The evaluation does not include adversarial records, severely malformed records, OCR-corrupted text, or formats outside the seven address categories. Therefore, the feedback loop should be viewed as a promising recovery mechanism rather than a fully validated robustness guarantee.

For context, a pattern-based parsing system may route similar unknown cases to a human expert for rule creation [27]. The automatic feedback mechanism resolved these observed cases in approximately 10 seconds each. However, automatic feedback is not persistent across runs. Human corrections can permanently enrich a rule base, while the proposed feedback loop only revises the strategy within the current execution.

5.7. Qualitative Error Analysis

Table 12 summarizes the main error patterns observed during evaluation. This analysis complements the quantitative results by explaining why performance differs across systems.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

Table 12. Qualitative error patterns observed during evaluation.

Error Type	Typical Example	Most Affected Systems
Record skipping	No output produced for records in long heterogeneous batches	Single-prompt LLM
Field conflation	Organization name placed into street address field	Single-prompt LLM and rule-based parser
Military city/state confusion	`APO`, `FPO`, or `DPO` treated as street content rather than city-equivalent value	Rule-based parser and occasional LLM outputs
Route misclassification	Highway route number treated as a unit number or separated from the street name	Rule-based parser and single-prompt LLM
Attention-line overcapture	`ATTN` content, organization name, and physical address merged into one field	Rule-based parser and single-prompt LLM
Incomplete structural output	Applicable fields left blank even when the record was processed	Rule-based parser and single-prompt LLM

The multi-agent architecture reduces these errors mainly by narrowing each worker's task. The address worker can devote its prompt context to address-specific guidance, while the name, phone, email, SSN, and date-of-birth workers operate under separate instructions. In contrast, the single-prompt baseline must include all extraction instructions in one prompt, which reduces the amount of task-specific guidance available for each field type. This supports the bounded cognition principle introduced in Section 3.2.

Confidence scores also reflected category difficulty at a broad level. Standard and individual address records received higher mean confidence scores, while military, highway, and attention-line records received lower confidence scores. These scores should not be treated as proof of correctness for individual records [17], but they may be useful as a prioritization signal for validation or future human review.

5.8. Contextual Comparison with Published Census Parsing Systems

Table 13 positions the proposed system relative to two previously published census parsing systems. This is not a direct leaderboard because the systems were evaluated on different datasets, with different models, and under different deployment assumptions. The purpose of the comparison is to clarify the trade-off space among accuracy, privacy, autonomy, and operational cost.

Table 13. Contextual comparison with published census parsing systems. Reported accuracy values come from each system's original evaluation and should not be interpreted as a direct head-to-head benchmark.

Dimension	Pattern-Based Active Learning System [27]	Cloud LLM Parsing System [39]	This Work
Accuracy in Original Evaluation	99.34%	99.8%	95.7%
Directly Comparable Dataset?	No	No	Reference condition
Model / Approach	Patterns and active learning	Claude Sonnet 4 through cloud API	gpt-oss-20b deployed locally
Human Involvement	Required for new patterns	None for zero-shot extraction	None during the run
Data Privacy	On-premise	Cloud API	On-premise
Evaluation Data	Census-derived	1,500 records	700 synthetic records
Error Recovery	Human expert review	Validation pipeline	Automatic bounded feedback
Adaptability	Expert intervention required for new patterns	Immediate LLM-based interpretation	Immediate LLM-based interpretation
Operational Overhead	Ongoing expert labor	API cost and cloud governance	Local GPU hardware and runtime cost

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

The proposed system reports lower accuracy than the two prior systems, but this difference should be interpreted in context. The pattern-based active learning system benefits from accumulated expert mappings, where human corrections progressively enrich the rule base and improve coverage over time [27]. The cloud LLM parsing system benefits from a larger commercial model and reports very high accuracy on its evaluated corpus [39]. In contrast, this work evaluates a locally deployed open-weight model inside an autonomous multi-agent architecture. The contribution is therefore not a claim of highest reported accuracy, but a demonstration of a different operating point: strong extraction quality with local inference, no cloud API dependency, no human intervention during execution, and automatic bounded recovery.

The accuracy gap may reflect several factors, including model scale, dataset composition, accumulated expert knowledge, prompt design, and evaluation methodology. For this reason, the comparison should be read as a trade-off analysis rather than a ranking. If the main goal is maximum accuracy and cloud processing is acceptable, a larger commercial LLM may be preferable. If expert review is available and similar formats recur over time, a pattern-based active learning system can build durable value. If sensitive records must remain local and new formats must be processed without immediate expert intervention, the proposed multi-agent approach provides a practical alternative.

5.9. Threats to Validity

Several threats to validity remain. The most important is the use of synthetic data. Although synthetic records enable controlled testing across address categories, they may not capture the full complexity of real census submissions, including misspellings, handwriting or OCR artifacts, incomplete responses, inconsistent punctuation, multilingual names, and unexpected formatting. The reported accuracy should therefore be viewed as an estimate under controlled conditions, not as evidence of production readiness.

A second threat is ground-truth circularity. Because records and labels are generated from the same synthetic process, some evaluation assumptions may be easier for the system to satisfy than they would be with independently annotated real data. Future evaluation should include independently labeled records, preferably with multiple annotators and disagreement analysis.

A third threat is model coverage. Only gpt-oss-20b was tested. The results may not generalize to other open-weight models, larger local models, or different quantization schemes. A stronger evaluation would compare multiple open-weight models under the same architecture.

A fourth threat is limited statistical power. Each condition was run three times, which is useful for preliminary stability checks but not enough for formal statistical inference. Future work should include more repeated runs, confidence intervals, and paired significance testing where appropriate.

A fifth threat is compute imbalance. The multi-agent system uses more LLM calls than the single-prompt baseline. This reflects the intended architectural trade-off, but future work should include ablation studies and compute-controlled comparisons to separate the benefits of specialization from the benefits of additional inference budget.

A final threat concerns the feedback loop. Only three feedback events were observed, and all occurred in known difficult categories. This provides useful examples of recovery, but not a robust estimate of feedback performance under adversarial or highly noisy conditions. Future evaluation should include stress tests with malformed records, unseen address formats, missing fields, and OCR-corrupted inputs.

6. Discussion

This section interprets the evaluation results, discusses the practical meaning of the multi-agent design, identifies limitations, and provides guidance for when the proposed approach is most appropriate. The main finding is that the proposed system does not maximize accuracy in isolation. Instead, it offers a different operating point: strong extraction quality, local deployment, no cloud API dependency, no human intervention during execution, and bounded automatic recovery.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

6.1. Interpretation of the Research Questions

The results provide positive but preliminary evidence for RQ1. The hierarchical three-stage architecture successfully processed all 700 synthetic records and achieved 95.7% component-level exact match accuracy across seven address categories. It also achieved 100% record-level recall, meaning that no record was skipped. This is important for census-style workflows, where completeness is a core operational requirement. The result suggests that a Planner-Manager-Worker architecture can organize complex extraction tasks effectively across standard, individual, university, military, rural route, highway, and attention-line address formats. However, because the evaluation uses synthetic records, the result should be interpreted as evidence of architectural feasibility rather than proof of production readiness.

For RQ2, the evidence supports the value of worker specialization. The multi-agent system outperformed the single-prompt LLM baseline by 14.8 percentage points overall, with the largest gains in the most difficult categories: military addresses, highway addresses, and attention-line records. It also eliminated the record-skipping problem observed in the single-prompt baseline, which skipped 47 of 700 records. These results suggest that assigning narrower tasks to specialized agents reduces prompt complexity and allows each worker to focus on a specific extraction problem. This aligns with prior findings that decomposed LLM pipelines can outperform monolithic prompting strategies [18] and with work on long-context degradation [23].

For RQ3, the feedback loop showed useful but limited evidence of recovery. It was triggered three times, once each in military, attention-line, and highway address cases, and all three cases were resolved on the first re-planning attempt. This suggests that inter-agent feedback can help recover from specific extraction failures. However, three events are too few to support a broad claim about robustness. The feedback loop should therefore be understood as a promising mechanism, not yet as a fully validated recovery system. More demanding evaluation with malformed, noisy, adversarial, and previously unseen formats is needed.

For RQ4, the results show that the proposed system improves accuracy and completeness at the cost of runtime. The rule-based parser is fastest but performs poorly on non-standard formats. The single-prompt LLM baseline is faster than the multi-agent system and improves generalization, but it skips records and produces more field-conflation errors. The proposed multi-agent system achieves the best performance among the systems directly evaluated on the 700-record synthetic dataset, but it requires approximately 292 LLM calls and 20 to 25 minutes of runtime. This trade-off may be acceptable for batch processing but would need optimization for interactive use.

For RQ5, the results show that a locally deployed open-weight model can provide useful extraction quality for census-style records while keeping inference on-premise. The system runs entirely on gpt-oss-20b deployed locally and achieves substantially higher accuracy than the rule-based baseline. However, its reported accuracy remains below published systems that use accumulated human expertise [27] or larger commercial cloud models [39]. The contribution is therefore not that local open-weight inference exceeds all alternatives, but that it can provide a practical balance of accuracy, privacy, autonomy, and deployability.

6.2. Trade-Offs Across Approaches

Table 14 summarizes the main trade-offs among the evaluated and contextual approaches. The comparison should be read as a deployment-oriented summary rather than a direct benchmark across identical datasets.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

Table 14. Trade-off comparison across approaches.

Dimension	Rule-Based	Single-Prompt LLM	Multi-Agent System	Published Census Systems [27,39]
Accuracy	52.3%	80.9%	95.7%	99.3–99.8%
Latency	<5 seconds	3 to 4 minutes	20 to 25 minutes	Varies by system
Privacy	On-premise	On-premise	On-premise	Mixed: on-premise or cloud
Human Effort	Rule authoring	Prompt authoring	Prompt authoring	Ongoing expert labor or none
Error Recovery	None	None	Automatic bounded feedback	Human review or validation pipeline
Adaptability	Low	High	High	Low or high, depending on system
Hardware Requirement	Minimal	Local GPU	Local GPU	Minimal or cloud API
Records Skipped	8	47	0	Not directly comparable

No approach dominates on every dimension. Rule-based systems are fast and simple but fragile when address formats vary. Cloud-hosted LLM systems can reach very high accuracy, but they may be unsuitable when sensitive data cannot leave the organization. Pattern-based active learning systems can also achieve very high accuracy, but they depend on expert review when new patterns appear. The proposed multi-agent system occupies a middle ground: it is less accurate than the highest reported systems, but it combines local deployment, zero-shot interpretation, complete record coverage, and automatic bounded recovery.

This trade-off is central to the paper. The system is not intended to replace every existing census parsing approach. Rather, it is most useful in settings where privacy constraints, limited expert availability, and format diversity make purely rule-based, cloud-based, or human-in-the-loop approaches difficult to use.

6.3. Architectural Implications

The results suggest three architectural lessons. First, decomposition matters. Separating planning, coordination, extraction, validation, and formatting reduces the burden placed on any single prompt. This appears especially helpful for heterogeneous records containing multiple PII types and diverse address structures.

Second, local open-weight deployment is becoming practical for structured extraction. The system runs on workstation-class hardware rather than cloud infrastructure, which is important for government, research, and regulated environments. Local deployment does not by itself solve every security problem, but it reduces exposure to external model providers and allows organizations to keep inference within their own controlled environment.

Third, modularity improves maintainability. New PII types can be added by creating additional Worker Agents, while improvements to address parsing can be made by updating the address worker prompt without changing the full pipeline. This modular structure is easier to maintain than a single large prompt in which changes to one extraction task may affect others.

6.4. Limitations and Failure Modes

The main limitation is the use of synthetic data. Synthetic records allow controlled evaluation across address categories, but they cannot fully reproduce real census submissions, which may contain misspellings, missing values, OCR artifacts, handwriting errors, inconsistent formatting, multilingual names, and unexpected field orderings. The reported 95.7% accuracy should therefore be treated as a controlled-evaluation result rather than an estimate of production performance.

A second limitation is that only one open-weight model was tested. The findings may not generalize to other local models, larger open-weight models, or different quantization strategies.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

Future work should compare multiple open-weight models under the same architecture to determine how much performance depends on the model versus the agent design.

A third limitation is the small number of repeated runs and feedback events. Three runs provide a basic stability check, but they are not enough for formal statistical testing. Similarly, the feedback loop was triggered only three times, so its reliability under noisy, adversarial, or unfamiliar formats remains uncertain.

A fourth limitation is runtime. The current prototype executes tasks sequentially and requires 20 to 25 minutes for 700 records. This is acceptable for some batch workflows, but it is slower than both the rule-based and single-prompt baselines. Because several extraction tasks are independent, parallel execution could reduce runtime substantially without changing the architecture.

A fifth limitation is the lack of persistent learning. The feedback loop can revise the strategy within a run, but it does not permanently update the system. If the same difficult format appears later, the system may need to rediscover the same correction. A future correction cache or human-reviewed memory of successful re-plans could allow the system to accumulate operational knowledge over time.

Finally, LLM-based systems introduce model-inherent risks. They may hallucinate plausible but incorrect field values, share error modes between extraction and validation, or perform unevenly across cultural naming conventions [5,7]. Production deployment should therefore include deterministic validation, such as ZIP-code checks, state abbreviation validation, phone and SSN format checks, postal reference checks, secure logging, and human review for low-confidence cases.

6.5. Deployment Guidance

The proposed multi-agent approach is most appropriate when sensitive records must remain local, address formats are diverse, expert review is not continuously available, and batch processing latency is acceptable. These conditions are common in regulated environments where data governance is as important as raw model performance.

The approach is less suitable when sub-second processing is required, when all inputs follow a narrow standard format, or when the highest possible accuracy is required and either cloud processing or sustained expert review is acceptable. In those cases, a rule-based parser, a commercial LLM system, or a pattern-based active learning system may be more appropriate.

A practical deployment strategy may be hybrid. The multi-agent system could process most records automatically, while low-confidence or validation-failed records are routed to human reviewers. This would combine the scalability of autonomous extraction with the quality control and persistent knowledge accumulation of expert review. Designing and evaluating such a hybrid system is a natural next step.

6.6. Broader Implications

Although this study focuses on census-style address parsing, the architecture is applicable to other structured extraction problems involving sensitive heterogeneous documents. Healthcare records, financial documents, legal filings, insurance forms, and government administrative records often require the same combination of local processing, fine-grained extraction, validation, and auditability. The results suggest that locally deployed open-weight models, when organized through specialized agents, can support practical knowledge extraction in settings where cloud-hosted LLMs are difficult to use.

7. Conclusions and Future Work

This paper presented a three-stage hierarchical multi-agent architecture for structured knowledge extraction from census-style records. The system uses a Planner Agent to analyze document structure and formulate extraction strategies, a Manager Agent to create dependency-aware task graphs, and specialized Worker Agents to perform extraction, validation, and formatting. A bounded feedback loop supports limited autonomous error recovery when extraction failures occur. The complete pipeline

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

runs on locally deployed gpt-oss-20b, allowing sensitive records to be processed without relying on an external cloud model provider.

The evaluation on 700 synthetic records across seven address categories showed that the proposed system achieved 95.7% component-level exact match accuracy, compared with 52.3% for the rule-based baseline and 80.9% for the single-prompt LLM baseline using the same model. The largest gains appeared on the most difficult address categories, including highway, military, and attention-line records. The system also achieved 100% record-level recall, avoiding the record-skipping behavior observed in the single-prompt baseline. These results suggest that task decomposition, specialized agents, and bounded feedback can improve the robustness of open-weight LLM extraction for heterogeneous census-style records [18,23].

The main contribution of this work is not a claim of highest reported accuracy. Previously published systems report higher accuracy by using accumulated expert mappings or larger cloud-hosted models [27,39]. Instead, this paper demonstrates a different operating point: strong extraction quality, local deployment, no cloud API dependency, no human intervention during execution, and automatic bounded recovery. This trade-off is important for organizations that must process sensitive records under strict privacy and governance constraints.

Several directions remain for future work. The most important next step is evaluation on real census or census-derived records with independently annotated ground truth. Such evaluation should include larger datasets, more repeated runs, confidence intervals, and statistical testing. Future studies should also compare multiple open-weight models under the same architecture to determine how much performance depends on the model versus the multi-agent design.

System capability can be improved in three ways. First, deterministic validation should be added to supplement LLM-based validation, including ZIP-code checks, state abbreviation validation, military ZIP verification, phone and SSN format checks, and postal reference checks. Second, the feedback mechanism should be extended with a persistent correction cache so that successful re-planning decisions can be reused in future runs. Third, independent Worker Agent tasks should be executed in parallel to reduce runtime from the current 20 to 25 minutes for 700 records.

Deployment readiness also requires further study. Bias auditing across diverse cultural naming patterns is necessary because personal names vary substantially across populations [5]. Head-to-head comparison with production-grade commercial address parsing services would also help clarify the accuracy, privacy, cost, and deployment trade-offs for real-world users. Finally, production use would require secure logging, access control, encryption, endpoint hardening, and governance review.

Overall, the results indicate that locally deployed open-weight models, when organized through a specialized multi-agent architecture, can support practical structured extraction from sensitive heterogeneous records. While the current system remains a prototype evaluated on synthetic data, it provides a promising foundation for privacy-preserving knowledge extraction in census processing and other regulated domains such as healthcare, finance, legal records, and government administration.

Funding: This research was partially supported by [omitted for peer review].

Data Availability Statement: The implementation code and synthetic evaluation dataset are available upon request through [omitted for peer review].

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Sample Evaluation Records

Table A1 presents representative records from the synthetic evaluation dataset. All data is entirely fictitious.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

Table A1. Representative evaluation records (one per category).

Category	Sample Address
Standard	311 N Parker St, Wichita, KS 67202
Individual	4521 Riverside Dr Apt 12B, Portland, OR 97201
University	Dept of Computer Science, Engr Bldg Rm 302, 100 University Ave, Austin, TX 78712
Military	PSC 340 Box 5892, APO, AE 09021
Rural Route	RR 7 Box 18, Carthage, MO 64836
Highway	5025 State Highway 16, Fredericksburg, TX 78624
Attention	ATTN: Legal Intake, HarborPoint Legal Group, 222 Franklin St Floor 11, Boston, MA 02110

The complete dataset and accompanying code are available upon request from the UALR ERIQ Center, according to the data governance policy.

References

1. Al-Rfou, R. libpostal: A Library for Parsing/Normalizing Street Addresses around the World. 2016. Available online: https://github.com/openvenues/libpostal (accessed on 15 January 2025).
2. Al Mandalawi, S.; Mohammed, M.A.; Maclean, H.; Cakmak, M.C.; Talburt, J.R. Policy-Aware Generative AI for Safe, Auditable Data Access Governance. In Proceedings of the 17th International Conference on Knowledge and System Engineering (KSE 2025), Ho Chi Minh City, Vietnam, 6–8 November 2025; IEEE: Piscataway, NJ, USA, 2025; pp. 1–6.
3. Althaf, A.M.; Mohammed, M.A.; Milanova, M.; Talburt, J.; Cakmak, M.C. Multi-Agent RAG Framework for Entity Resolution: Advancing Beyond Single-LLM Approaches with Specialized Agent Coordination. *Computers* **2025**, *14*, 525. https://doi.org/10.3390/computers14120525
4. Borkar, V.; Deshmukh, K.; Sarawagi, S. Automatic Segmentation of Text into Structured Records. In Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data, Santa Barbara, CA, USA, 21–24 May 2001; ACM: New York, NY, USA, 2001; pp. 175–186. https://doi.org/10.1145/375663.375682
5. Borgman, C.L.; Siegfried, S.L. Getty's Synoname and Its Cousins: A Survey of Applications of Personal Name-Matching Algorithms. *J. Am. Soc. Inf. Sci.* **1992**, *43*, 459–476. https://doi.org/10.1002/(SICI)1097-4571(199208)43:7<459::AID-ASI1>3.0.CO;2-B
6. Brown, T.B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; et al. Language Models Are Few-Shot Learners. In Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS 2020), Virtual, 6–12 December 2020; Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M.F., Lin, H., Eds.; Curran Associates: Red Hook, NY, USA, 2020; Volume 33, pp. 1877–1901.
7. Carlini, N.; Tramer, F.; Wallace, E.; Jagielski, M.; Herbert-Voss, A.; Lee, K.; et al. Extracting Training Data from Large Language Models. In Proceedings of the 30th USENIX Security Symposium, Virtual, 11–13 August 2021; USENIX Association: Berkeley, CA, USA, 2021; pp. 2633–2650.
8. Christen, P. *Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection*; Springer: Berlin/Heidelberg, Germany, 2012. https://doi.org/10.1007/978-3-642-31164-2
9. Comber, S.; Zeng, R. International Address Parsing Using Conditional Random Fields. *GeoInformatica* **2019**, *23*, 431–452. https://doi.org/10.1007/s10707-019-00360-z
10. Cucerzan, S.; Yarowsky, D. Language Independent Named Entity Recognition Combining Morphological and Contextual Evidence. In Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP-VLC), College Park, MD, USA, 21–22 June 1999; ACL: Stroudsburg, PA, USA, 1999; pp. 90–99.
11. Cunningham, H.; Maynard, D.; Bontcheva, K.; Tablan, V. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002), Philadelphia, PA, USA, 7–12 July 2002; ACL: Stroudsburg, PA, USA, 2002; pp. 168–175. https://doi.org/10.3115/1073083.1073112
12. Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2019), Minneapolis, MN, USA, 2–7 June 2019; ACL: Stroudsburg, PA, USA, 2019; Volume 1, pp. 4171–4186. https://doi.org/10.18653/v1/N19-1423

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

13. Dwork, C.; Roth, A. The Algorithmic Foundations of Differential Privacy. *Found. Trends Theor. Comput. Sci.* **2014**, *9*, 211–407. https://doi.org/10.1561/0400000042
14. Fellegi, I.P.; Sunter, A.B. A Theory for Record Linkage. *J. Am. Stat. Assoc.* **1969**, *64*, 1183–1210. https://doi.org/10.1080/01621459.1969.10501049
15. Ferber, J. *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*; Addison-Wesley: Boston, MA, USA, 1999.
16. Goldberg, D.W.; Wilson, J.P.; Knoblock, C.A. From Text to Geographic Coordinates: The Current State of Geocoding. *URISA J.* **2007**, *19*, 33–46.
17. Kadavath, S.; Conerly, T.; Askell, A.; Henighan, T.; Drain, D.; Perez, E.; et al. Language Models (Mostly) Know What They Know. *arXiv* **2022**, arXiv:2207.05221.
18. Khattab, O.; Singhvi, A.; Maheshwari, P.; Zhang, Z.; Santhanam, K.; Vardhamanan, S.; et al. DSPy: Compiling Declarative Language Model Calls into Self-Improving Pipelines. *arXiv* **2023**, arXiv:2310.03714.
19. Lafferty, J.; McCallum, A.; Pereira, F.C.N. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In Proceedings of the 18th International Conference on Machine Learning (ICML 2001), Williamstown, MA, USA, 28 June–1 July 2001; Morgan Kaufmann: San Francisco, CA, USA, 2001; pp. 282–289.
20. LangChain. LangGraph: Build Stateful, Multi-Actor Applications with LLMs. 2024. Available online: https://github.com/langchain-ai/langgraph (accessed on 15 January 2025).
21. Li, J.; Sun, A.; Han, J.; Li, C. A Survey on Deep Learning for Named Entity Recognition. *IEEE Trans. Knowl. Data Eng.* **2020**, *34*, 50–70. https://doi.org/10.1109/TKDE.2020.2981314
22. Li, R.; Peng, H.; Li, J. CodeIE: Large Code Generation Models Are Better Few-Shot Information Extractors. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL 2023), Toronto, ON, Canada, 9–14 July 2023; ACL: Stroudsburg, PA, USA, 2023; pp. 8396–8411. https://doi.org/10.18653/v1/2023.acl-long.855
23. Liu, N.F.; Lin, K.; Hewitt, J.; Paranjape, A.; Bevilacqua, M.; Petroni, F.; Liang, P. Lost in the Middle: How Language Models Use Long Contexts. *Trans. Assoc. Comput. Linguist.* **2024**, *12*, 157–173. https://doi.org/10.1162/tacl_a_00638
24. Lu, Y.; Bartolo, M.; Moore, A.; Riedel, S.; Stenetorp, P. Fantastically Ordered Prompts and Where to Find Them: Overcoming Few-Shot Prompt Order Sensitivity. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL 2022), Dublin, Ireland, 22–27 May 2022; ACL: Stroudsburg, PA, USA, 2022; pp. 8086–8098. https://doi.org/10.18653/v1/2022.acl-long.556
25. Madaan, A.; Tandon, N.; Gupta, P.; Hallinan, S.; Gao, L.; Wiegreffe, S.; et al. Self-Refine: Iterative Refinement with Self-Feedback. In Proceedings of the 37th Conference on Neural Information Processing Systems (NeurIPS 2023), New Orleans, LA, USA, 10–16 December 2023; Curran Associates: Red Hook, NY, USA, 2023; Volume 36.
26. McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; y Arcas, B.A. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS 2017), Fort Lauderdale, FL, USA, 20–22 April 2017; PMLR: Cambridge, MA, USA, 2017; Volume 54, pp. 1273–1282.
27. Mohammed, O.K.; Syed, K.; Talburt, J.; Tarannum, A.; Kashif, A.K.K.; Khan, S.; et al. A Pattern-Based Approach to Name and Address Parsing with Active Learning. In Proceedings of the 17th International Conference on Agents and Artificial Intelligence (ICAART 2025), Porto, Portugal, 23–25 February 2025; SCITEPRESS: Set'ubal, Portugal, 2025; Volume 3, pp. 70–77. https://doi.org/10.5220/0013077500003890
28. Mohammed, M.A.; Talburt, J.R.; Mohammed, A.; Syed, K. Entity Resolution with Household Movement Discovery Using Google Generative AI. In Proceedings of the International Conference on Information Technology—New Generations (ITNG 2025), Las Vegas, NV, USA, 14–16 April 2025; Springer: Cham, Switzerland, 2025; pp. 469–481.
29. Mohammed, M.A.; Talburt, J.R.; Claassens, L.; Marais, A. Retrieval-Augmented Multi-LLM Ensemble for Industrial Part Specification Extraction. In Proceedings of the 17th International Conference on Knowledge and System Engineering (KSE 2025), Ho Chi Minh City, Vietnam, 6–8 November 2025; IEEE: Piscataway, NJ, USA, 2025; pp. 1–6.
30. Mohammed, M.A.; Talburt, J.R.; Althaf, A.M.; Milanova, M. Multi-LLM Record Linkage: A Comparative Analysis Framework for Co-Residence Pattern Discovery. In Proceedings of the 12th Annual Conference on Computational Science and Computational Intelligence (CSCI 2025), Las Vegas, NV, USA, 4–7 November 2025; IEEE: Piscataway, NJ, USA, 2025.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1337.v1

31. Mohammed, M.A.; Al Mandalawi, S.; Maclean, H.; Talburt, J.R. Multilingual Customer Record Linkage: A Novel Approach Using LLMs for Cross-Lingual Entity Resolution. In Proceedings of the 12th Annual Conference on Computational Science and Computational Intelligence (CSCI 2025), Las Vegas, NV, USA, 4–7 November 2025; IEEE: Piscataway, NJ, USA, 2025.
32. Moura, J. CrewAI: Framework for Orchestrating Role-Playing AI Agents. 2024. Available online: https://github.com/joaomdmoura/crewAI (accessed on 15 January 2025).
33. OpenAI. Introducing GPT-OSS-20b: Open-Weight Mixture-of-Experts Model. 2025. Available online: https://openai.com/index/introducing-gpt-oss/ (accessed on 1 June 2025).
34. Peng, B.; Galley, M.; He, P.; Cheng, H.; Xie, Y.; Hu, Y.; et al. Check Your Facts and Try Again: Improving Large Language Models with External Knowledge and Automated Feedback. *arXiv* **2023**, arXiv:2302.12813.
35. Russell, S.J.; Norvig, P. *Artificial Intelligence: A Modern Approach*, 4th ed.; Pearson: Hoboken, NJ, USA, 2021.
36. Sharma, H.; Jadon, R.S.; Shukla, B.K. Address Parsing Using Deep Learning Approach. In Proceedings of the 2018 International Conference on Computing, Power and Communication Technologies (GUCON 2018), Greater Noida, India, 28–29 September 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 545–549. https://doi.org/10.1109/GUCON.2018.8675060
37. Shinn, N.; Cassano, F.; Gopinath, A.; Narasimhan, K.; Yao, S. Reflexion: Language Agents with Verbal Reinforcement Learning. In Proceedings of the 37th Conference on Neural Information Processing Systems (NeurIPS 2023), New Orleans, LA, USA, 10–16 December 2023; Curran Associates: Red Hook, NY, USA, 2023; Volume 36.
38. Sweller, J. Cognitive Load during Problem Solving: Effects on Learning. *Cogn. Sci.* **1988**, *12*, 257–285. https://doi.org/10.1207/s15516709cog1202_4
39. Tarannum, A.; Mohammed, M.A.; Cakmak, M.C.; Al Mandalawi, S.; Talburt, J. A System for Name and Address Parsing with Large Language Models. *arXiv* **2026**, arXiv:2601.18014.
40. Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; et al. LLaMA: Open and Efficient Foundation Language Models. *arXiv* **2023**, arXiv:2302.13971.
41. U.S. Census Bureau. 2020 Census Results. 2021. Available online: https://www.census.gov/2020census (accessed on 15 January 2025).
42. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; et al. Attention Is All You Need. In Proceedings of the 31st Conference on Neural Information Processing Systems (NeurIPS 2017), Long Beach, CA, USA, 4–9 December 2017; Curran Associates: Red Hook, NY, USA, 2017; Volume 30, pp. 5998–6008.
43. Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In Proceedings of the 36th Conference on Neural Information Processing Systems (NeurIPS 2022), New Orleans, LA, USA, 28 November–9 December 2022; Curran Associates: Red Hook, NY, USA, 2022; Volume 35, pp. 24824–24837.
44. Willard, B.T.; Louf, R. Efficient Guided Generation for Large Language Models. *arXiv* **2023**, arXiv:2307.09702.
45. Wooldridge, M.; Jennings, N.R. Intelligent Agents: Theory and Practice. *Knowl. Eng. Rev.* **1995**, *10*, 115–152. https://doi.org/10.1017/S0269888900008122
46. Wu, Q.; Bansal, G.; Zhang, J.; Wu, Y.; Li, B.; Zhu, E.; et al. AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation. *arXiv* **2023**, arXiv:2308.08155.
47. Xu, D.; Chen, W.; Peng, W.; Zhang, C.; Xu, T.; Zhao, X.; et al. Large Language Models for Generative Information Extraction: A Survey. *arXiv* **2023**, arXiv:2312.17617.
48. Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; Cao, Y. ReAct: Synergizing Reasoning and Acting in Language Models. In Proceedings of the 11th International Conference on Learning Representations (ICLR 2023), Kigali, Rwanda, 1–5 May 2023; OpenReview: Kigali, Rwanda, 2023.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.