Augmented Knowledge Base Creation and Curation using LLMs

Joshua Cook

Georga Institute of Technology

Project Proposal

Abstract

This project proposes an innovative approach to automated knowledge extraction and ontology mapping using Large Language Models (LLMs). Building on my own 20+ years of education and AI engineering expertise, I aim to develop a sophisticated system that integrates advanced Named Entity Recognition (NER), entity relation extraction, and ontology mapping to generate organization-specific ontologies.

The solution leverages and extends the Chelle Ubiquitous Language (Appendix I) as the foundation for domain modeling and knowledge representation. By incorporating recent advancements like SPIRES and MapperGPT, the project will enhance the precision of knowledge base population and ontology mapping. The system is designed to streamline training, update field engineers, and provide an accessible source of truth for teams.

Key features of our approach include:

- LLM-based knowledge extraction and augmentation
- Relationship mapping between entities
- Ontological mapping for entity overlap
- User-friendly interfaces for document upload and refinement review
- Flexible data model supporting iterative refinements

INTRODUCTION

Existing approaches for knowledge extraction and ontology mapping have seen challenges when it comes to efficiency, accuracy, and scalability. Manual curation of knowledge bases is a time consuming process (Caufield, J. H., et al., 2024). Artificial intelligence and natural language

processing have provided some help, "but current approaches rely on extensive training data, and are not able to populate arbitrarily complex nested knowledge schemas" (Caufield, J. H., et al., 2024). When it comes to accuracy, AI/NLP approaches "often provide very high recall, but low precision, due to lexical ambiguity. As a consequence of this, mapping efforts often resort to a labor intensive manual mapping refinement through a human curator" (Matentzoglu, N., et al., 2023).

Meanwhile, the growth of big data, especially unstructured text data, necessitates organized solutions for managing knowledge (Jackson et al., 2021). Ontology and knowledge base development are an effective solution to this problem. Ontologies serve as "a formal, explicit specification of a shared conceptualization" (Guarino et al., 2009). They can help to address ambiguity in communication. By providing a structured approach to representing and managing unstructured data, ontologies contribute to the development of any organization.

Existing approaches to information extraction (IE) tasks struggle to handle unstructured data and implicit dependencies. Wei et al. (2024) highlight that vanilla prompts often fail in solving IE with original task instructions, particularly when extracting structured data containing multiple dependent elements through one-time prediction. Current methods frequently rely on labeled data, limiting their flexibility in task decomposition. For instance, while pipeline methods like PURE (Zhong and Chen, 2021) decompose complex tasks into different parts, they still require supervision from labeled data.

Zero-shot and few-shot learning approaches with LLMs produce inconsistent and sometimes incorrect results. Mateiu et al. (2023) report inconsistent outcomes and the generation of unnecessary or incorrect axioms in their experiments. Furthermore, current solutions still require significant human involvement for validation and refinement. As Mateiu et al. (2023) note, while LLM-based tools aim to support ontology engineering and reduce the need for interactions with domain experts, they are intended as support tools that still require human supervision and validation.

RELATED WORK

Academic tools and approaches have been developed to address challenges in ontology engineering and semantic web standards. LinkML (Moxon et al., 2021) is a framework for semantic web standards and data modeling, offering an object-oriented approach that simplifies

the production of FAIR ontology-ready data. It supports a wide range of data types and provides automated translation to multiple formats. The Ontology Development Kit (ODK, Matentzoglu et al., 2022), offers standardized workflows and comprehensive tooling for ontology engineering, including automated quality control and testing to maintain ontology integrity. Dead Simple OWL Design Patterns (DOS-DPs, Osumi-Sutherland et al., 2017) provided a format for specifying OWL design patterns, using a simple, human-readable format designed for non-technical ontology editors. Ontology Templates (OTTR, Forssell et al., 2018), offers a second-order language for specifying ontologies through recurring patterns, aiming to make ontologies more readable and maintainable.

Qiang et al. (2025) propose a framework called Agent-OM, which uses LLM agents for ontology matching. In the realm of knowledge graph embedding, Pan et al. (2023) describe methods like Pretrain-KGE and kNN-KGE that utilize LLMs to encode textual descriptions and treat entities and relations as special tokens, respectively. Hybrid database systems are being employed for efficient entity matching and retrieval. As noted by Qiang et al. (2025) and Lewis et al. (2020), these systems combine traditional relational databases for storing entity metadata with advanced vector databases for natural language-based content, enabling more sophisticated similarity searches. To address the challenge of false positives in LLM-based systems, techniques such as validators and mergers are being developed to refine matches and reduce hallucinations. Additionally, retrieval-augmented generation (RAG) models, as explored by Lewis et al. (2020), are being used to combine pre-trained parametric and non-parametric memory for improved language generation. LLMs are also finding applications in knowledge graph completion tasks. Pan et al. (2023) describe GenKGC, which uses the BART model as a backbone and employs a relation-guided demonstration technique to facilitate the model's learning process.

In the commercial sector, AI-powered knowledge extraction and management platforms are emerging. Companies like Sana Labs are developing learning platforms and knowledge assistants that integrate with existing company apps and knowledge bases, while platforms like GetGuru are focusing on making company-wide knowledge accessible to customer-facing teams.

PROPOSED WORK

This project proposes an innovative approach to automated knowledge extraction and ontology mapping, leveraging the power of Large Language Models (LLMs) and building upon 20+ years of education and AI engineering expertise. The proposed system integrates advanced Named Entity Recognition (NER), entity relation extraction, and ontology mapping to generate organization-specific ontologies. The Chelle Ubiquitous Language (CUL, Appendix I) serves as the foundation for domain modeling and knowledge representation, providing a standardized vocabulary for organizational domain modeling. The proposed system features LLM-based knowledge extraction and augmentation, relationship mapping between entities, and ontological mapping for entity overlap. Simple interfaces for document upload and refinement review will be built for the purpose of interacting with the AI system. A flexible data model supporting iterative refinements will be part of the solution.

Chelle Ubiquitous Language. The CUL (Appendix I) is the domain modeling schema – a standardized vocabulary for knowledge representation. We propose to update CUL to reflect the latest application model, Assets-Knowledge-Assessment-Instruction (AKAI). The CUL structure will be incorporated into the knowledge base schema. As we progress, the CUL will serve as a framework for validation and quality control. The user interface should be built around the CUL.

System Architecture. The proposed system architecture consists of four primary components:

- Jupyter Notebook Server
    - an interactive environment for testing and refining various model and prompt combinations
- Fast API Server
    - will handle model-to-model+prompt requests
    - each endpoint supporting a specific model+prompt combination
    - store of results in MongoDB
- Streamlit Frontend
    - document upload interface
    - inbox interface
- MongoDB
    - primary data store

User Flow. The user flow in our proposed system begins with the input of well-structured markdown files. These files require no special formatting. The system processes the documents through a series of steps – the core research deliverables. These steps include named entity extraction, definition extraction from source documents, LLM-based definition augmentation, relationship mapping between entities, and ontological mapping for entity overlap. The extracted and processed information is then organized into four primary entities: **concepts**, **definitions**, **relationships**, and **ontologies**. To support iterative refinement and user validation, we store these as refinements (concept refinements, definition refinements, relationship refinements, and ontology refinements). Users can review, accept, or reject these refinements. The current version of any entity is dynamically constructed by applying accepted refinement diffs successively.

User Interface and Experience. The user interface is simple and is solely to support this particular work. The **Upload Page** provides a simple text upload interface. The **Inbox Page**, which will be designed as one project deliverable allows users to actively curate their knowledge base. This page presents refinements for user review and action, allowing them to interact directly with the extracted knowledge and proposed changes. Users can view, accept, reject, or modify refinements.

Access and APIs. For this proof of concept (POC) the API will be built locally with no security. The specific APIs will be determined based on our research outcomes. These APIs will likely include endpoints for document upload, knowledge extraction, refinement submission, and knowledge base querying.

Out of Scope. Several aspects have been deliberately placed out of scope for this POC. Security measures, while crucial for a production system, will not be implemented in this phase. Scalability considerations are also out of scope, as our primary goal is to demonstrate the effectiveness of our knowledge extraction and ontology mapping approach rather than its performance at scale. Data visualization features are not included in this POC.

Technologies Used. Our proposed system leverages a variety of technologies. At the core of our architecture is Docker Compose, which we use to orchestrate our POC system. The primary programming language for implementation is Python. We make extensive use of libraries such as Instructor for structured outputs from language models and Streamlit for building the user

interface. For our language model needs, we primarily utilize Claude-sonnet-3.5. To facilitate rapid development and code generation, we employ Cursor and Anthropic Claude.

Methodology

Our methodology centers around an LLM-based approach, exclusively using large language models and carefully crafted prompts for knowledge extraction and ontology mapping. We conceptualize each prompt+LLM pair as a distinct "model," allowing for modular design and easy experimentation. By chaining multiple such models, we build our pipelines for knowledge base creation and curation. We will first investigate best practices in atomic-level prompting (zero-shot learning, few-shot learning, chain-of-thought, etc.), then investigate several key LLM flows, including named entity extraction, entity definition generation, extraction strategies from source documents, LLM-based augmentation processes, relationship mapping between entities, and ontological mapping for entity overlap.

LLM-based Approach and Prompt Design. In our LLM-based approach, we prioritize simplicity and modularity in prompt design. Each model is tasked with one simple, well-defined task, allowing for better control and interpretability of the results. This "one model, one simple task" philosophy enables us to create complex workflows by chaining together these simpler components. Our prompt design process focuses on clarity and specificity, ensuring that each prompt elicits the desired information or action from the language model. We will iterate on these prompts based on performance metrics and user (in this work, the researcher) feedback, continuously refining the approach.

Existing System Interation. Our proposed system is designed with integration in mind, assuming the existence of pre-existing systems for extracted and structured markdown. This means that a successful proof of concept can be integrated into an existing production application ecosystem without too much tech debt and effort from the engineering team.

User Curation System. Central to the design is the user curation system, which we believe is crucial for maintaining the accuracy and relevance of the extracted knowledge. The system revolves around a user inbox containing entity, definition, relation, and mapping change data. This inbox serves as the primary interface for user interaction with the system's outputs. We implement a change data capture mechanism for our main data types, allowing us to track and version all modifications to the knowledge base. The current state of the knowledge base can be

reconstructed at any point by applying the sequence of approved changes. This approach not only ensures data integrity but also provides a clear audit trail of how the knowledge base has evolved over time. Users can review proposed changes, approve them as-is, approve with alterations, or reject them entirely. This collaborative curation process helps to combine the efficiency of automated extraction with the nuanced understanding of human experts, resulting in a more accurate and contextually appropriate knowledge base.

Evaluation Methods

To rigorously assess the effectiveness of our proposed system, we will employ a multi-faceted evaluation approach. Primarily, we will focus on user inbox metrics as a direct measure of the system's practical utility. These metrics will include the rate of user-approved (again, in this work, the research will be the user) changes, changes approved with alterations, user rejections, and a categorized analysis of rejection reasons. This granular feedback will provide invaluable insights into the accuracy and relevance of our automated extractions and suggestions. Beyond these user-centric metrics, we will also track and analyze several system performance indicators. These will include processing time metrics to evaluate efficiency, coverage metrics to assess the comprehensiveness of our knowledge extraction, and consistency metrics to ensure uniformity across different runs and document types.

DELIVERABLES

Deliverables will be structured using milestones. Milestone 1 after three weeks, milestone 2 after six weeks, and final deliverable after eight weeks.

Milestone 1. The theme of milestone 1 is prompt research and the primary deliverable will be a **detailed report** on prompts selected and their performance. Additional deliverables will be the **code repo** containing the defined system architecture, the revised **Chelle Ubiquitous Language**, and a **code repo** containing all of our prompts.

Deliverables:

- prompt research report (primary)
- code repo for system
- Chelle Ubiquitous Language (revised)
- code repo for prompts

Key tasks include:

- developing and running local system
- prompt research
- testing and analyzing prompts

Milestone 2. The theme of milestone 2 is end-to-end delivery. The primary deliverable will be the evolved **code repo** and a **report** on the application with a **demo video**. Additional deliverables will be ui mocks and the database schema.

Deliverables:

- evolved code repo for system (primary)
- report on the application
- demo video
- ui mocks
- database schemas

Key tasks include:

- database configuration
- api development
- ui design and development

Final Deliverable. The theme of the final deliverable is the final report. This **final report** is the primary deliverable and will document the complete project. Net new aspects of this relative to previous milestones is including analysis of the user input collected by the UI. This report will contain all of the metrics collected and analysis of these metrics. Additional deliverables will be detailed **documentation**.

Final Deliverables:

- final report (primary)
- documentation

APPENDIX I - CHELLE UBIQUITOUS LANGUAGE

Developed in collaboration with Chelle Chief Technology Officer, Nash Taylor. Nash is not a Georgia Tech student.

Conventions

In this document, the following conventions apply, which should help in navigating:

- The sub-headings are the terms contained in the Glossary, and the content within is the complete definition of the term, followed by the facts we know about the term.
- When a definition or fact makes reference to another term in the Glossary, the term is *italicized*. A term is **bolded** in its own definition.
    - Example: '**Understanding** is a degree of demonstrated retention of a *Concept*.' Concept is italicized here because it is a term belonging to the Glossary. Understanding is bolded because this is its own definition.
- When a definition makes use of customized language in defining a property of the term, the customized language is put in "quotes". For example, if a term from the Glossary defines an entity which has a property that may only contain certain values, and those values are enumerated, they would be quoted. Customized language from the definition of one term may appear in the definition of another term.
    - An example of this can be found in the *Concept* term, where we use the word "lexeme". That word is a child of the term *Concept*, so it's not big enough to be its own term in the Glossary, but it may need to be referenced elsewhere, so the quotes help identify it.

Ordering

Terms in the Glossary are sorted based on mentions in the definition of each term. That is, if a term uses another term from the Glossary in its definition, it must appear *after* the term that it uses. This does **not** apply to the facts. Facts are cyclic, as they are nearly impossible to disentangle in a way that preserves knowledge and meaning.

So for an entirely unfamiliar reader, one approach is to read through all of the Definitions first, ignoring the Facts, and then go back and look at the Facts, once each term and its definition has been absorbed.

Terms

User

Definition: A **User** is an individual person who is using the product via a logged in account.

- **Users** are managed externally to the product, via Clerk.
- **Users** can be either *Mentors* or *Learners.*
- nash@chelle.ai what about admins/users in Clerk?

Organization

Definition: An **Organization** is a group of Users that use the product in tandem.

- **Organizations** are managed externally to the product, via Clerk.
- It is not possible to use Chelle and not belong to an **Organization**

Concept

Definition: A **Concept** is a "lexeme" whose definition is owned by an *Organization*.

- Aka Knowledge Grapes.
- A **Concept** is identified either manually by a *Mentor*, or automatically via AI.
- A **Concept** contains *Knowledge*.
- A **Concept** is associated with other Concepts through *Mentions* in its *Knowledge*.
- A **Concept** has an associated "Summative" *Assessment*.
- A **Concept** may have a "Formative" *Assessment* constructed for it ad-hoc.
- Two **Concepts** from different *Organizations* may have identical "lexemes", but they must have different identities.
- A **Concept** may have "synonyms", which are alternative "lexemes".

Taxonomy

Understanding

Definition: **Understanding** is a degree of demonstrated retention of a *Concept*.

- **Understanding** is demonstrated via *Assessment*.
- The levels of **Understanding** for a "Practical" Concept are: None, Functional, Practical, and Proficient.
- The levels of **Understanding** for a "Theoretical" Concept are: None, Basic, Intermediate, and Mastery.

Job

Definition: A **Job** is the real-world role of a *User* within their *Organization*.

- The fundamental goal of the product is to improve the *User's* performance at their **Job**.

Role

Definition: A **Role** is a persona that a *User* may assume temporarily in order to utilize the product in service of a specific purpose.

- The types of **Role** are: Mentor, Learner.

Knowledge

Definition: Knowledge is the complete set of instructional materials available regarding a *Concept*.

- **Knowledge** may be AI-generated or *User*-created.
- **Knowledge** is stored as text.
- **Knowledge** may be consumed as text, audio, or video.
- **Knowledge** is sourced from a *Corpus*.

Assessment

Definition: An **Assessment** is a set of questions that fosters one's *Understanding* of a *Concept*.

- **Assessments** are taken by *Learners*.
- There is one type of **Assessment** question: multiple choice.

Evaluation

Definition: An **Evaluation** is a set of questions that measure one's *Understanding* of a *Concept*.

- **Evaluations** are taken by *Learners*.
- There is one type of **Evaluation** question: multiple choice.

Task

Definition: A **Task** is a real-world assignment of work, fulfilling some function of a *Job*.

- A **Task** is assigned to a *Learner* by a *Mentor* as part of a *Guide*.
- A **Task** is "surfaced" in the product as an *Asset*.

Learner

Definition: **Learner** is a *Role* that a *User* may assume to indicate that their active purpose is to learn.

- **Learners** learn by consuming *Knowledge*.
- **Learners** consume *Knowledge* by "completing" *Guides*.
- [Learner]-consume->[Knowledge] ⇔ [Learner]-complete->[Guide]

Certification

Definition: A **Certification** is a record of verified *Understanding* of a *Concept*.

- **Certifications** are awarded to *Learners* automatically.
- **Certification** delivery is triggered by completion of a "Summative" *Assessment* for the given *Concept*.

Article

Definition: An **Article** is a body of text presenting all *Knowledge* of a *Concept* in a narrative form.

- **Articles** are optimized for *Learner* engagement and retention.

Teaser

Definition: A **Teaser** is a short summary of the *Knowledge* of a given *Concept*.

- **Teasers** are optimized to give a *Learner* a "Functional" or "Basic" *Understanding* of the *Concept*.

Guide

Definition: A **Guide** is a unit of learning deployed to a *Learner*, containing a short objective statement, a collection of *Concepts*, a set of *Assessments*, and a *Task*.

- A **Guide** is "completed" when the Learner obtains *Certification* for all of the *Concepts* in the Guide and completes the *Task*.

Authority

Definition: The **Authority** of a *Concept* is the *User* who assumes ownership of its *Knowledge*.

- By default, the **Authority** of a *Concept* is the *Mentor* who "surfaced" the first *Asset* which provided *Knowledge* for the *Concept*.
- Any *Mentor* can be assigned as the **Authority** of a *Concept* by any other *Mentor*.

Active Role

Definition: A *User's* **Active Role** is the *Role* that they have assumed in the current context.

- A *User's* **Active Role** can be changed within an App Session by the User, given sufficient Permissions.

Mentor

Definition: **Mentor** is a *Role* that a *User* may assume to indicate that their active purpose is to oversee learning.

- **Mentors** compose their *Organization*'s *Corpus*.
- **Mentors** create and manage *Guides*.
- **Mentors** track *Learner* progress.

Learning Path

Definition: A **Learning Path** is the ordered list of all *Concepts* which one would need to first obtain *Certification* for in order to obtain *Certification* for the target *Concept*.

- A **Learning Path** is constructed by topologically sorting the subgraph of *Concepts* in a *Glossary* that terminates at the target *Concept*.

Knowledge Dependency

Definition: A **Knowledge Dependency** is the requirement to acquire at least "Practical" or "Intermediate" *Understanding* of a *Concept* in order to understand another.

- A **Knowledge Dependency** is created based on a *Mention* in the *Knowledge* of one *Concept* of another.

Definitional Dependency

Definition: A **Definitional Dependency** is the requirement to acquire "Basic" or "Functional" *Understanding* of a *Concept* in order to understand another.

- A **Definitional Dependency** is created based on a *Mention* in the Definition of one *Concept* of another.

Mention

Definition: A **Mention** is the appearance of a *Concept*'s "lexeme" in text.

- A **Mention** in another *Concept*'s *Knowledge* under "Knowledge" triggers the creation of a *Knowledge Dependency*.
- A **Mention** in another *Concept's Knowledge* under "Definition" triggers the creation of a *Definitional Dependency*.
- As the *Knowledge* of a *Concept* is updated, **Mentions** are re-calculated.

Integration

Definition: An **Integration** is an authorized connection between the product and an *Organization's* account on a third party platform for the purpose of ingesting data.

- **Integrations** fall into categories: documentation, codebases, conversations, and meets.
- Connecting an **Integration** does not automatically "register" all *Assets* it provides access to.

Asset

Definition: An **Asset** is a raw source of information.

- An **Asset** may be text, image, video, audio, or structured data.
- It may be uploaded directly by a *User* or acquired via an *Integration*.
- **Assets**

Corpus

Definition: A **Corpus** is a container for *Assets*.

- Each *Organization* has its own **Corpus**.

Glossary

Definition: A **Glossary** is the set of *Concepts* owned by a particular *Organization*.

- The **Glossary** is the primary interface through which *Users* interact with *Knowledge*.

Norm

Definition: A **Norm** is a "lexeme" that is automatically copied to every *Organization* as an empty *Concept*, because it is universal terminology.

- A **Norm** starts out with no *Knowledge* when it is first added to an *Organization*, even if it has associated *Knowledge* in other *Organizations*.
- A **Norm** can optionally be removed from an *Organization* if desired.

REFERENCES

Babaei Giglou, H., D'Souza, J., & Auer, S. (2023). LLMs4OL: Large language models for ontology learning. In International Semantic Web Conference (pp. 408-427). Cham: Springer Nature Switzerland.

Caufield, J. H., Hegde, H., Emonet, V., Harris, N. L., Joachimiak, M. P., Matentzoglu, N., ... & Mungall, C. J. (2024). Structured prompt interrogation and recursive extraction of semantics (SPIRES): A method for populating knowledge bases using zero-shot learning. Bioinformatics, 40(3), btae104.

Forssell, H., Kindermann, C., Lupp, D. P., Sattler, U., & Thorstensen, E. (2018). Generating ontologies from templates: A rule-based approach for capturing regularity. arXiv preprint arXiv:1809.10436.

Guarino, N. and Giaretta, P. (1995) Ontologies and Knowledge Bases. In: Towards Very Large Knowledge Bases, IOS Press, Amsterdam, 1-2.

Jackson, R., Matentzoglu, N., Overton, J. A., Vita, R., Balhoff, J. P., Buttigieg, P. L., ... & Peters, B. (2021). OBO Foundry in 2021: operationalizing open data principles to evaluate ontologies. Database, 2021, baab069.

Karp, P. D. (2016). How much does curation cost?. Database, 2016, baw110.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., ... & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. Advances in Neural Information Processing Systems, 33, 9459-9474.

Matentzoglu, N., Caufield, J. H., Hegde, H. B., Reese, J. T., Moxon, S., Kim, H., ... & Mungall, C. J. (2023). Mappergpt: Large language models for linking and mapping entities. arXiv preprint arXiv:2310.03666.

Matentzoglu, N., Goutte-Gattat, D., Tan, S. Z. K., Balhoff, J. P., Carbon, S., Caron, A. R., ... & Osumi-Sutherland, D. (2022). Ontology Development Kit: a toolkit for building, maintaining and standardizing biomedical ontologies. Database, 2022, baac087.

Mateiu, P., & Groza, A. (2023, September). Ontology engineering with large language models. In 2023 25th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC) (pp. 226-229). IEEE.

Moxon, S. A., Solbrig, H., Unni, D. R., Jiao, D., Bruskiewich, R. M., Balhoff, J. P., ... & Mungall, C. J. (2021). The Linked Data Modeling Language (LinkML): A General-Purpose Data Modeling Framework Grounded in Machine-Readable Semantics. ICBO, 3073, 148-151.

Osumi-Sutherland, D., Courtot, M., Balhoff, J. P., & Mungall, C. (2017). Dead simple OWL design patterns. Journal of biomedical semantics, 8, 1-7.

Pan, S., Luo, L., Wang, Y., Chen, C., Wang, J., & Wu, X. (2023). Unifying large language models and knowledge graphs: A roadmap. IEEE Transactions on Knowledge and Data Engineering, 36, 3580-3599. https://api.semanticscholar.org/CorpusID:259165563

Procko, T. T., Elvira, T., & Ochoa, O. (2023). GPT-4: A Stochastic Parrot or Ontological Craftsman? Discovering Implicit Knowledge Structures in Large Language Models. In 2023 Fifth International Conference on Transdisciplinary AI (TransAI) (pp. 147-154). IEEE.

Qiang, Z., Wang, W., & Taylor, K. (2025). Agent-OM: Leveraging LLM Agents for Ontology Matching. Proceedings of the VLDB Endowment, 18(1).
https://doi.org/10.48550/arXiv.2312.00326

Wan, Z., Cheng, F., Mao, Z., Liu, Q., Song, H., Li, J., & Kurohashi, S. (2023). GPT-RE: In-context Learning for Relation Extraction using Large Language Models. Kyoto University, Japan; Zhejiang University, China.

Wang, S., Sun, X., Li, X., Ouyang, R., Wu, F., Zhang, T., Li, J., & Wang, G. (2023). GPT-NER: Named Entity Recognition via Large Language Models. arXiv preprint arXiv:2301.13294. https://arxiv.org/abs/2301.13294

Wei, X., Cui, X., Cheng, N., Wang, X., Zhang, X., Huang, S., Xie, P., Xu, J., Chen, Y., Zhang, M., Jiang, Y., & Han, W. (2024). ChatIE: Zero-Shot Information Extraction via Chatting with ChatGPT. Retrieved from https://arxiv.org/abs/2302.10205