General set-up

Load required modules

In [ ]:
```
# load required modules
from plyer import notification  # Windows notifications
from langchain_core.output_parsers import StrOutputParser  # output parser for L
from langchain_openai import ChatOpenAI  # load LLM model
import requests  # API search requests
import feedparser  # API feed parsers
from urllib.parse import quote_plus  # parse AI calls when prohibited characters
import xml.etree.ElementTree as ET  # parses XML data from an API response strin
from langchain.prompts import ChatPromptTemplate  # assemble prompt for langchai
import ast  # support with formatting string outputs from LLMs into lists
from langchain.prompts import PromptTemplate  # simplified prompt template
from pydantic import BaseModel, validator, HttpUrl  # Pydantic object classes to
from typing import List, Optional, Union  # additional functions to support refe
import numpy as np  # standard library to process numerical objects
from sentence_transformers import SentenceTransformer  # model to generate embed
import sqlite3  # database implementation
import csv  # csv file storage
import os  # work with directories
from datetime import date, datetime  # work with date and time
from sklearn.neighbors import NearestNeighbors  # compute nearest neighbours bas
import tkinter as tk  # Python's standard GUI
from tkinter import ttk  # access to the Tk themed widget set
from tkinter import simpledialog  # GUI module
from langchain.schema.runnable import RunnableLambda  # wrapper function for age
```

In [ ]:
```
# Helper function to send desktop notifications when long-running tasks are comp
# Used to alert the user once the reference search pipeline finishes.


def notify_completion():
    """Prints a notification when code running has finished."""

    notification.notify(
        title="Completion",
        message="SAGE reference search complete",
        app_name="SAGE",
        timeout=300
    )
```

In [ ]:
```
# test function
notify_completion()
```

Create agents

Prompt preparation agent

In [ ]:
```
# Instantiate LangChain's string output parser to extract clean string outputs f

output_parser = StrOutputParser()
```

In [ ]:
```
# Prompt preparation agent
# Agent to simplify complex user queries into keyword-based search prompts using


def prompt_preparation_agent(input_dict):
    """
    Simplifies a verbose or natural language search prompt into a concise, acade

    Args:
        input_dict (dict): Dictionary containing the key 'user_prompt' with the

    Returns:
        dict: Includes the original 'user_prompt' and the simplified 'search_pro
    """




    user_prompt = input_dict["user_prompt"]

    prompt_template = ChatPromptTemplate.from_messages([
        {"role": "system", "content": "You simplify user prompts for academic re
        {"role": "user", "content": """Please simplify the following prompt for
    ])

    global llm # ensure base model is available for subsequent agents with the s

    llm = ChatOpenAI(
        model="gpt-4o", # most recent OpenAI model, with more advanced reasoning
        temperature=0, # looking for results as accurate as possible and with li
        max_tokens=None, # no restriction on prompt size (especially as the mode
        timeout=None,
        max_retries=2,
        api_key=openai_api_key
        )



    llm_chain = prompt_template | llm | output_parser
    simplified_prompt = llm_chain.invoke({"user_prompt": user_prompt})

    #  global response
    response = {"user_prompt": user_prompt, "search_prompt": simplified_prompt}
    print(f"Prompt preparation completed at {datetime.now()}")
    print(f"Output: {response}\n\n")
    return response
```

In [ ]:
```
# use-case testing

test_search_prompt = "Please provide 5 academic references on using natural lang
# openai_api_key= "REMOVED"
prompt_preparation_agent({"user_prompt": test_search_prompt})
```

```
Prompt preparation completed at 2025-04-12 01:00:13.411464
Output: {'user_prompt': 'Please provide 5 academic references on using natural la
nguage processing vs rule-based logic for multi-agent system communication.', 'se
arch_prompt': 'natural language processing vs rule-based logic in multi-agent com
munication'}
```

Out[ ]:
```
{'user_prompt': 'Please provide 5 academic references on using natural language
processing vs rule-based logic for multi-agent system communication.',
 'search_prompt': 'natural language processing vs rule-based logic in multi-age
nt communication'}
```

Search agent

In [ ]:
```python
# Search agent
# queries multiple academic APIs (CrossRef, arXiv, PubMed) and extracts structur



def search_agent(input_dict):
    """
    Queries three academic APIs (CrossRef, arXiv, and PubMed) using a search-opt

    Args:
        input_dict (dict): Must contain 'search_prompt'.

    Returns:
        dict: Original input dict with additional 'search_results' key (a list o
    """


    query = input_dict["search_prompt"]


    def search_crossref(query, max_results=5):
        url = "https://api.crossref.org/works"
        params = {
            "query": query,
            "rows": max_results,
            "sort": "relevance",
            "select": "title,author,container-title,issued,DOI,URL,abstract,volu
        }
        response = requests.get(url, params=params)
        response.raise_for_status()
        results = response.json().get('message', {}).get('items', [])
        references = []

        for item in results:
            title = item.get('title', ["Not found"])[0]
            authors = item.get('author', [])
            author_list = [f"{a.get('given', '')} {a.get('family', '')}" for a i
            journal = item.get('container-title', ["Not found"])[0]
            year = item.get('issued', {}).get('date-parts', [[None]])[0][0]
            doi = item.get('DOI', 'Not found')
            url = item.get('URL', 'Not found')
            abstract = item.get('abstract', 'Not found').replace('<jats:p>', '')
            issue = item.get('issue', 'Not found')
            pages = item.get('page', 'Not found')

            references.append({
                "Source": "CrossRef",
                "Title": title,
                "Year": year,
                "Authors": ", ".join(author_list),
                "Publication": journal,
                "Issue": issue,
                "Pages": pages,
                "DOI": doi,
                "URL": url,
                "Abstract": abstract
            })

        print(f"CrossRef search completed at {datetime.now()}\n")


        return references

    def search_arxiv(query, max_results=5):
        base_url = "http://export.arxiv.org/api/query?"
        query_url = f"search_query=all:{quote_plus(query)}&start=0&max_results={
        feed = feedparser.parse(base_url + query_url)
        references = []

        for entry in feed.entries:
            title = entry.title
            authors = ", ".join(author.name for author in entry.authors)
            published = entry.published.split("T")[0]
            year = published.split("-")[0]
            arxiv_id = entry.id.split('/')[-1]
            doi = entry.get('arxiv_doi', 'Not found')
            url = entry.link
            abstract = entry.get("summary", "Not found").replace("\n", " ")

            references.append({
                "Source": "arXiv",
                "Title": title,
                "Year": year,
                "Authors": authors,
                "Publication": "arXiv",
                "Issue": "Not found",
                "Pages": "Not found",
                "DOI": doi,
                "URL": url,
                "Abstract": abstract
            })

        print(f"arXiv search completed at {datetime.now()}\n")

        return references

    def search_pubmed(query, max_results=5):
        # Step 1: Search PubMed and get list of matching IDs
        # Step 2: Fetch metadata for each ID using XML response parsing

        search_url = "https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi
        fetch_url = "https://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi"

        # Step 1: Search
        params = {
            "db": "pubmed",
```

```python
            "term": query,
            "retmode": "json",
            "retmax": max_results
        }
        search_response = requests.get(search_url, params=params)
        search_response.raise_for_status()
        id_list = search_response.json().get("esearchresult", {}).get("idlist",

        if not id_list:
            return []

        # Step 2: Fetch details
        fetch_params = {
            "db": "pubmed",
            "id": ",".join(id_list),
            "retmode": "xml"
        }
        fetch_response = requests.get(fetch_url, params=fetch_params)
        fetch_response.raise_for_status()
        root = ET.fromstring(fetch_response.content)
        references = []

        for article in root.findall(".//PubmedArticle"):
            medline = article.find("MedlineCitation")
            article_info = medline.find("Article")

            title = article_info.findtext("ArticleTitle", "Not found")

            authors = article_info.findall(".//Author")
            author_list = []
            for a in authors:
                last = a.findtext("LastName", "")
                fore = a.findtext("ForeName", "")
                if last or fore:
                    author_list.append(f"{fore} {last}")

            journal = article_info.findtext(".//Journal/Title", "Not found")
            issue = article_info.findtext(".//Issue", "Not found")
            pages = article_info.findtext(".//MedlinePgn", "Not found")
            year = article_info.findtext(".//PubDate/Year", "Not found")
            abstract = article_info.findtext(".//Abstract/AbstractText", "Not fo
            doi = "Not found"
            for id_elem in article.findall(".//ArticleId"):
                if id_elem.attrib.get("IdType") == "doi":
                    doi = id_elem.text
            # Extract DOI specifically from the article’s metadata node


            pmid = medline.findtext("PMID", "Not found")
            url = f"https://pubmed.ncbi.nlm.nih.gov/{pmid}/"

            references.append({
                "Source": "PubMed",
                "Title": title,
                "Year": year,
                "Authors": ", ".join(author_list) or "Not available",
                "Publication": journal,
                "Issue": issue,
                "Pages": pages,
                "DOI": doi,
                "URL": url,
                "Abstract": abstract
            })


        return references

    # global response

    print(f"Pubmed search completed at {datetime.now()}\n")


    results = search_crossref(query) + search_arxiv(query) + search_pubmed(query

    response = {**input_dict, "search_results": results}

    print(f"Complete reference search completed at {datetime.now()}")
    print(f"Output: {response}\n\n")


    return response
```

In [ ]:
```
# unit testing

search_agent(response)

response
```

```
CrossRef search completed at 2025-04-12 01:01:04.661409
arXiv search completed at 2025-04-12 01:01:05.669766
Complete reference search completed at 2025-04-12 01:01:06.266219
Output: {'user_prompt': 'Please provide 5 academic references on using natural la
nguage processing vs rule-based logic for multi-agent system communication.', 'se
arch_prompt': 'natural language processing vs rule-based logic in multi-agent com
munication', 'search_results': [{'Source': 'CrossRef', 'Title': 'Handling Trust i
n a Cloud based Multi Agent System', 'Year': 2021, 'Authors': 'Imen Bouabdallah,
Hakima Mellah', 'Publication': 'Natural Language Processing', 'Issue': 'Not foun
d', 'Pages': '241-255', 'DOI': '10.5121/csit.2021.112319', 'URL': 'https://doi.or
g/10.5121/csit.2021.112319', 'Abstract': 'Cloud computing is an opened and distri
buted network that guarantees access to a large amount of data and IT infrastruct
ure at several levels (software, hardware...). With the increase demand, handling
clients’ needs is getting increasingly challenging. Responding to all requesting
clients could lead to security breaches, and since it is the provider’s responsib
ility to secure not only the offered cloud services but also the data, it is impo
rtant to ensure clients reliability. Although filtering clients in the cloud is n
ot so common, it is required to assure cloud safety.  In this paper, by implement
ing multi agent systems in the cloud to handle interactions for the providers, tr
ust is introduced at agent level to filtrate the clients asking for services by u
sing Particle Swarm Optimization and acquaintance knowledge to determine maliciou
s and untrustworthy clients. The selection depends on previous knowledge and over
all rating of trusted peers. The conducted experiments show that the model output
s relevant results, and even with a small number of peers, the framework is able
to converge to the best solution. The model presented in this paper is a part of
ongoing work to adapt interactions in the cloud.'}, {'Source': 'CrossRef', 'Titl
e': 'Emergent Linguistic Phenomena in Multi-Agent Communication Games', 'Year': 2
019, 'Authors': 'Laura Harding Graesser, Kyunghyun Cho, Douwe Kiela', 'Publicatio
n': 'Proceedings of the 2019 Conference on Empirical Methods in Natural Language
Processing and the 9th International Joint Conference on Natural Language Process
ing (EMNLP-IJCNLP)', 'Issue': 'Not found', 'Pages': '3698-3708', 'DOI': '10.1865
3/v1/d19-1384', 'URL': 'https://doi.org/10.18653/v1/d19-1384', 'Abstract': 'Not f
ound'}, {'Source': 'CrossRef', 'Title': 'Natural Language Processing based Rule B
ased Discourse Analysis of Marathi Text', 'Year': 2020, 'Authors': 'Kalpana B. Kh
andale, C. Namrata Mahender', 'Publication': '2020 International Conference on El
ectronics and Sustainable Communication Systems (ICESC)', 'Issue': 'Not found',
'Pages': '356-362', 'DOI': '10.1109/icesc48915.2020.9155653', 'URL': 'https://do
i.org/10.1109/icesc48915.2020.9155653', 'Abstract': 'Not found'}, {'Source': 'Cro
ssRef', 'Title': 'An Approach to Using XML and a Rule-Based Content Language with
an Agent Communication Language', 'Year': 2000, 'Authors': 'Benjamin N. Grosof, Y
annis Labrou', 'Publication': 'Lecture Notes in Computer Science', 'Issue': 'Not
found', 'Pages': '96-117', 'DOI': '10.1007/10722777_7', 'URL': 'https://doi.org/1
0.1007/10722777_7', 'Abstract': 'Not found'}, {'Source': 'CrossRef', 'Title': 'Ma
chine learning vs. rule-based methods for document classification of electronic h
ealth records within mental health care—A systematic literature review', 'Year':
2025, 'Authors': 'Emil Rijcken, Kalliopi Zervanou, Pablo Mosteiro, Floortje Schee
pers, Marco Spruit, Uzay Kaymak', 'Publication': 'Natural Language Processing Jou
rnal', 'Issue': 'Not found', 'Pages': '100129', 'DOI': '10.1016/j.nlp.2025.10012
9', 'URL': 'https://doi.org/10.1016/j.nlp.2025.100129', 'Abstract': 'Not found'},
{'Source': 'arXiv', 'Title': 'Data Augmentation with In-Context Learning and Comp
arative Evaluation in\n  Math Word Problem Solving', 'Year': '2024', 'Authors':
'Gulsum Yigit, Mehmet Fatih Amasyali', 'Publication': 'arXiv', 'Issue': 'Not foun
d', 'Pages': 'Not found', 'DOI': '10.1007/s42979-024-02853-x', 'URL': 'http://arx
iv.org/abs/2404.03938v1', 'Abstract': 'Math Word Problem (MWP) solving presents a
challenging task in Natural Language Processing (NLP). This study aims to provide
MWP solvers with a more diverse training set, ultimately improving their ability
to solve various math problems. We propose several methods for data augmentation
by modifying the problem texts and equations, such as synonym replacement, rule-b
ased: question replacement, and rule based: reversing question methodologies over
two English MWP datasets. This study extends by introducing a new in-context lear
```

ning augmentation method, employing the Llama-7b language model. This approach in
volves instruction-based prompting for rephrasing the math problem texts. Perform
ance evaluations are conducted on 9 baseline models, revealing that augmentation
methods outperform baseline models. Moreover, concatenating examples generated by
various augmentation methods further improves performance.'}, {'Source': 'arXiv',
'Title': 'From Single Agent to Multi-Agent: Improving Traffic Signal Control', 'Y
ear': '2024', 'Authors': 'Maksim Tislenko, Dmitrii Kisilev', 'Publication': 'arXi
v', 'Issue': 'Not found', 'Pages': 'Not found', 'DOI': 'Not found', 'URL': 'htt
p://arxiv.org/abs/2406.13693v1', 'Abstract': 'Due to accelerating urbanization, t
he importance of solving the signal control problem increases. This paper analyze
s various existing methods and suggests options for increasing the number of agen
ts to reduce the average travel time. Experiments were carried out with 2 dataset
s. The results show that in some cases, the implementation of multiple agents can
improve existing methods. For a fine-tuned large language model approach there is
small enhancement on all metrics.'}, {'Source': 'arXiv', 'Title': 'On the Express
iveness of Joining', 'Year': '2015', 'Authors': 'Thomas Given-Wilson, Axel Lega
y', 'Publication': 'arXiv', 'Issue': 'Not found', 'Pages': 'Not found', 'DOI': '1
0.4204/EPTCS.189.9', 'URL': 'http://arxiv.org/abs/1508.04854v1', 'Abstract': 'The
expressiveness of communication primitives has been explored in a common framewor
k based on the pi-calculus by considering four features: synchronism (asynchronou
s vs synchronous), arity (monadic vs polyadic data), communication medium (shared
dataspaces vs channel-based), and pattern-matching (binding to a name vs testing
name equality vs intensionality). Here another dimension coordination is consider
ed that accounts for the number of processes required for an interaction to occu
r. Coordination generalises binary languages such as pi-calculus to joining langu
ages that combine inputs such as the Join Calculus and general rendezvous calculu
s. By means of possibility/impossibility of encodings, this paper shows coordinat
ion is unrelated to the other features. That is, joining languages are more expre
ssive than binary languages, and no combination of the other features can encode
a joining language into a binary language. Further, joining is not able to encode
any of the other features unless they could be encoded otherwise.'}, {'Source':
'arXiv', 'Title': 'Certifying Choreography Compilation', 'Year': '2021', 'Author
s': 'Luís Cruz-Filipe, Fabrizio Montesi, Marco Peressotti', 'Publication': 'arXi
v', 'Issue': 'Not found', 'Pages': 'Not found', 'DOI': '10.1007/978-3-030-85315-0
_8', 'URL': 'http://arxiv.org/abs/2102.10698v2', 'Abstract': 'Choreographic progr
amming is a paradigm for developing concurrent and distributed systems, where pro
grams are choreographies that define, from a global viewpoint, the computations a
nd interactions that communicating processes should enact. Choreography compilati
on translates choreographies into the local definitions of process behaviours, gi
ven as terms in a process calculus.   Proving choreography compilation correct is
challenging and error-prone, because it requires relating languages in different
paradigms (global interactions vs local actions) and dealing with a combinatorial
explosion of proof cases. We present the first certified program for choreography
compilation for a nontrivial choreographic language supporting recursion.'}, {'So
urce': 'arXiv', 'Title': 'Distilling Text into Circuits', 'Year': '2023', 'Author
s': 'Vincent Wang-Mascianica, Jonathon Liu, Bob Coecke', 'Publication': 'arXiv',
'Issue': 'Not found', 'Pages': 'Not found', 'DOI': 'Not found', 'URL': 'http://ar
xiv.org/abs/2301.10595v1', 'Abstract': "This paper concerns the structure of mean
ings within natural language. Earlier, a framework named DisCoCirc was sketched t
hat (1) is compositional and distributional (a.k.a. vectorial); (2) applies to ge
neral text; (3) captures linguistic `connections' between meanings (cf. grammar)
(4) updates word meanings as text progresses; (5) structures sentence types; (6)
accommodates ambiguity. Here, we realise DisCoCirc for a substantial fragment of
English.   When passing to DisCoCirc's text circuits, some `grammatical bureaucra
cy' is eliminated, that is, DisCoCirc displays a significant degree of (7) inter-
and intra-language independence. That is, e.g., independence from word-order conv
entions that differ across languages, and independence from choices like many sho
rt sentences vs. few long sentences. This inter-language independence means our t
ext circuits should carry over to other languages, unlike the language-specific t
ypings of categorial grammars. Hence, text circuits are a lean structure for the

`actual substance of text', that is, the inner-workings of meanings within text a
cross several layers of expressiveness (cf. words, sentences, text), and may capt
ure that what is truly universal beneath grammar. The elimination of grammatical
bureaucracy also explains why DisCoCirc: (8) applies beyond language, e.g. to spa
tial, visual and other cognitive modes. While humans could not verbally communica
te in terms of text circuits, machines can.   We first define a `hybrid grammar'
for a fragment of English, i.e. a purpose-built, minimal grammatical formalism ne
eded to obtain text circuits. We then detail a translation process such that all
text generated by this grammar yields a text circuit. Conversely, for any text ci
rcuit obtained by freely composing the generators, there exists a text (with hybr
id grammar) that gives rise to it. Hence: (9) text circuits are generative for te
xt."}]}

```
Out[ ]: {'user_prompt': 'Please provide 5 academic references on using natural language
        processing vs rule-based logic for multi-agent system communication.',
         'search_prompt': 'natural language processing vs rule-based logic in multi-age
        nt communication',
         'search_results': [{'Source': 'CrossRef',
           'Title': 'Handling Trust in a Cloud based Multi Agent System',
           'Year': 2021,
           'Authors': 'Imen Bouabdallah, Hakima Mellah',
           'Publication': 'Natural Language Processing',
           'Issue': 'Not found',
           'Pages': '241-255',
           'DOI': '10.5121/csit.2021.112319',
           'URL': 'https://doi.org/10.5121/csit.2021.112319',
           'Abstract': 'Cloud computing is an opened and distributed network that guara
        ntees access to a large amount of data and IT infrastructure at several levels
        (software, hardware...). With the increase demand, handling clients’ needs is g
        etting increasingly challenging. Responding to all requesting clients could lea
        d to security breaches, and since it is the provider’s responsibility to secure
        not only the offered cloud services but also the data, it is important to ensur
        e clients reliability. Although filtering clients in the cloud is not so commo
        n, it is required to assure cloud safety.  In this paper, by implementing multi
        agent systems in the cloud to handle interactions for the providers, trust is i
        ntroduced at agent level to filtrate the clients asking for services by using P
        article Swarm Optimization and acquaintance knowledge to determine malicious an
        d untrustworthy clients. The selection depends on previous knowledge and overal
        l rating of trusted peers. The conducted experiments show that the model output
        s relevant results, and even with a small number of peers, the framework is abl
        e to converge to the best solution. The model presented in this paper is a part
        of ongoing work to adapt interactions in the cloud.'},
          {'Source': 'CrossRef',
           'Title': 'Emergent Linguistic Phenomena in Multi-Agent Communication Games',
           'Year': 2019,
           'Authors': 'Laura Harding Graesser, Kyunghyun Cho, Douwe Kiela',
           'Publication': 'Proceedings of the 2019 Conference on Empirical Methods in N
        atural Language Processing and the 9th International Joint Conference on Natura
        l Language Processing (EMNLP-IJCNLP)',
           'Issue': 'Not found',
           'Pages': '3698-3708',
           'DOI': '10.18653/v1/d19-1384',
           'URL': 'https://doi.org/10.18653/v1/d19-1384',
           'Abstract': 'Not found'},
          {'Source': 'CrossRef',
           'Title': 'Natural Language Processing based Rule Based Discourse Analysis of
        Marathi Text',
           'Year': 2020,
           'Authors': 'Kalpana B. Khandale, C. Namrata Mahender',
           'Publication': '2020 International Conference on Electronics and Sustainable
        Communication Systems (ICESC)',
           'Issue': 'Not found',
           'Pages': '356-362',
           'DOI': '10.1109/icesc48915.2020.9155653',
           'URL': 'https://doi.org/10.1109/icesc48915.2020.9155653',
           'Abstract': 'Not found'},
          {'Source': 'CrossRef',
           'Title': 'An Approach to Using XML and a Rule-Based Content Language with an
        Agent Communication Language',
           'Year': 2000,
           'Authors': 'Benjamin N. Grosof, Yannis Labrou',
           'Publication': 'Lecture Notes in Computer Science',
           'Issue': 'Not found',
           'Pages': '96-117',
           'DOI': '10.1007/10722777_7',
           'URL': 'https://doi.org/10.1007/10722777_7',
           'Abstract': 'Not found'},
          {'Source': 'CrossRef',
           'Title': 'Machine learning vs. rule-based methods for document classificatio
        n of electronic health records within mental health care—A systematic literatur
        e review',
           'Year': 2025,
           'Authors': 'Emil Rijcken, Kalliopi Zervanou, Pablo Mosteiro, Floortje Scheep
        ers, Marco Spruit, Uzay Kaymak',
           'Publication': 'Natural Language Processing Journal',
           'Issue': 'Not found',
           'Pages': '100129',
           'DOI': '10.1016/j.nlp.2025.100129',
           'URL': 'https://doi.org/10.1016/j.nlp.2025.100129',
           'Abstract': 'Not found'},
          {'Source': 'arXiv',
           'Title': 'Data Augmentation with In-Context Learning and Comparative Evaluat
        ion in\n  Math Word Problem Solving',
           'Year': '2024',
           'Authors': 'Gulsum Yigit, Mehmet Fatih Amasyali',
           'Publication': 'arXiv',
           'Issue': 'Not found',
           'Pages': 'Not found',
           'DOI': '10.1007/s42979-024-02853-x',
           'URL': 'http://arxiv.org/abs/2404.03938v1',
           'Abstract': 'Math Word Problem (MWP) solving presents a challenging task in
        Natural Language Processing (NLP). This study aims to provide MWP solvers with
        a more diverse training set, ultimately improving their ability to solve variou
        s math problems. We propose several methods for data augmentation by modifying
        the problem texts and equations, such as synonym replacement, rule-based: quest
        ion replacement, and rule based: reversing question methodologies over two Engl
        ish MWP datasets. This study extends by introducing a new in-context learning a
        ugmentation method, employing the Llama-7b language model. This approach involv
        es instruction-based prompting for rephrasing the math problem texts. Performan
        ce evaluations are conducted on 9 baseline models, revealing that augmentation
        methods outperform baseline models. Moreover, concatenating examples generated
        by various augmentation methods further improves performance.'},
          {'Source': 'arXiv',
           'Title': 'From Single Agent to Multi-Agent: Improving Traffic Signal Contro
        l',
           'Year': '2024',
           'Authors': 'Maksim Tislenko, Dmitrii Kisilev',
           'Publication': 'arXiv',
           'Issue': 'Not found',
           'Pages': 'Not found',
           'DOI': 'Not found',
           'URL': 'http://arxiv.org/abs/2406.13693v1',
           'Abstract': 'Due to accelerating urbanization, the importance of solving the
        signal control problem increases. This paper analyzes various existing methods
        and suggests options for increasing the number of agents to reduce the average
        travel time. Experiments were carried out with 2 datasets. The results show tha
        t in some cases, the implementation of multiple agents can improve existing met
        hods. For a fine-tuned large language model approach there is small enhancement
        on all metrics.'},
          {'Source': 'arXiv',
           'Title': 'On the Expressiveness of Joining',
           'Year': '2015',
           'Authors': 'Thomas Given-Wilson, Axel Legay',
           'Publication': 'arXiv',
           'Issue': 'Not found',
           'Pages': 'Not found',
           'DOI': '10.4204/EPTCS.189.9',
           'URL': 'http://arxiv.org/abs/1508.04854v1',
           'Abstract': 'The expressiveness of communication primitives has been explore
        d in a common framework based on the pi-calculus by considering four features:
        synchronism (asynchronous vs synchronous), arity (monadic vs polyadic data), co
        mmunication medium (shared dataspaces vs channel-based), and pattern-matching
        (binding to a name vs testing name equality vs intensionality). Here another di
        mension coordination is considered that accounts for the number of processes re
        quired for an interaction to occur. Coordination generalises binary languages s
        uch as pi-calculus to joining languages that combine inputs such as the Join Ca
        lculus and general rendezvous calculus. By means of possibility/impossibility o
        f encodings, this paper shows coordination is unrelated to the other features.
        That is, joining languages are more expressive than binary languages, and no co
        mbination of the other features can encode a joining language into a binary lan
        guage. Further, joining is not able to encode any of the other features unless
        they could be encoded otherwise.'},
          {'Source': 'arXiv',
           'Title': 'Certifying Choreography Compilation',
           'Year': '2021',
           'Authors': 'Luís Cruz-Filipe, Fabrizio Montesi, Marco Peressotti',
           'Publication': 'arXiv',
           'Issue': 'Not found',
           'Pages': 'Not found',
           'DOI': '10.1007/978-3-030-85315-0_8',
           'URL': 'http://arxiv.org/abs/2102.10698v2',
           'Abstract': 'Choreographic programming is a paradigm for developing concurre
        nt and distributed systems, where programs are choreographies that define, from
        a global viewpoint, the computations and interactions that communicating proces
        ses should enact. Choreography compilation translates choreographies into the l
        ocal definitions of process behaviours, given as terms in a process calculus.
        Proving choreography compilation correct is challenging and error-prone, becaus
        e it requires relating languages in different paradigms (global interactions vs
        local actions) and dealing with a combinatorial explosion of proof cases. We pr
        esent the first certified program for choreography compilation for a nontrivial
        choreographic language supporting recursion.'},
          {'Source': 'arXiv',
           'Title': 'Distilling Text into Circuits',
           'Year': '2023',
           'Authors': 'Vincent Wang-Mascianica, Jonathon Liu, Bob Coecke',
           'Publication': 'arXiv',
           'Issue': 'Not found',
           'Pages': 'Not found',
           'DOI': 'Not found',
           'URL': 'http://arxiv.org/abs/2301.10595v1',
           'Abstract': "This paper concerns the structure of meanings within natural la
        nguage. Earlier, a framework named DisCoCirc was sketched that (1) is compositi
        onal and distributional (a.k.a. vectorial); (2) applies to general text; (3) ca
        ptures linguistic `connections' between meanings (cf. grammar) (4) updates word
        meanings as text progresses; (5) structures sentence types; (6) accommodates am
        biguity. Here, we realise DisCoCirc for a substantial fragment of English.   Wh
        en passing to DisCoCirc's text circuits, some `grammatical bureaucracy' is elim
        inated, that is, DisCoCirc displays a significant degree of (7) inter- and intr
        a-language independence. That is, e.g., independence from word-order convention
        s that differ across languages, and independence from choices like many short s
        entences vs. few long sentences. This inter-language independence means our tex
        t circuits should carry over to other languages, unlike the language-specific t
        ypings of categorial grammars. Hence, text circuits are a lean structure for th
```

```
e `actual substance of text', that is, the inner-workings of meanings within te
xt across several layers of expressiveness (cf. words, sentences, text), and ma
y capture that what is truly universal beneath grammar. The elimination of gram
matical bureaucracy also explains why DisCoCirc: (8) applies beyond language,
e.g. to spatial, visual and other cognitive modes. While humans could not verba
lly communicate in terms of text circuits, machines can.   We first define a `h
ybrid grammar' for a fragment of English, i.e. a purpose-built, minimal grammat
ical formalism needed to obtain text circuits. We then detail a translation pro
cess such that all text generated by this grammar yields a text circuit. Conver
sely, for any text circuit obtained by freely composing the generators, there e
xists a text (with hybrid grammar) that gives rise to it. Hence: (9) text circu
its are generative for text."}]}
```

Semantic parsing agent

In [ ]:

```python
# Semantic parsing agent
# Agent that filters and selects the most relevant references from search result


def semantic_parsing_agent(input_dict):


    """
    Filters and selects the most relevant references based on semantic similarit
    Also integrates past reference context from the database if relevant.

    Args:
        input_dict (dict): Contains 'user_prompt' and 'search_results'.

    Returns:
        str: A structured LLM-generated list of the most relevant references.
    """


    user_prompt = input_dict["user_prompt"]
    results = input_dict["search_results"]


    baseline_instructions = """
You are a scientific research assistant.
Your task is to consider a list of references and to select the ones that are mo
From the list of references provided, you will select the 5 that are most releva

Your response will be composed of a list of webpages in the following format:
Reference number, reference title, year, authors, publication, issue, pages, DOI
All authors should be listed and you should not use "et al."
Begin your response directly with the list of references you selected.

    """


    # Format references
    def format_reference(ref, index):
        return (
            f"{index+1}. {ref['Title']} ({ref['Year']})\n"
            f"   Authors: {ref['Authors']}\n"
            f"   Publication: {ref['Publication']}, Issue: {ref['Issue']}, Pages
            f"   DOI: {ref['DOI']}\n"
            f"   URL: {ref['URL']}\n"
            f"   Abstract: {ref['Abstract']}\n"
        )

    formatted_refs = "\n".join([format_reference(r, i) for i, r in enumerate(res





    # Initialize database connection

    global conn

    conn = create_connection('SAGE results/SAGE_database.db') # creates (if not
    create_table_results() # creates the search_results table (if first iteratio

    # Initialize context-aware search
    context_aware_results = context_aware_search(user_prompt)

    # Compile search prompt
    if context_aware_results is not None:
        content = (
            baseline_instructions +
            f"{user_prompt} " +
            "List of references to consider:\n" +
            formatted_refs+
            f"Consider also the following context from previous searches. You ca
        )
    else:
        content = (baseline_instructions +
                f"{user_prompt}" +
                "List of references to consider:\n" +
                formatted_refs
                )


    content = content.replace("{", "").replace("}", "")

    search_prompt = ChatPromptTemplate(messages=[{"role": "user", "content": con

    llm_chain = search_prompt| llm | output_parser

    # global response

    response = llm_chain.invoke(input={"user_prompt": content})

    print(f"Semantic parsing completed at {datetime.now()}")
    print(f"Output: {response}\n\n")


    return response
```

In [ ]:

```
# unit testing

semantic_parsing_agent(response)
```

```
Context search completed at 2025-04-11 23:48:37.301928
Output: [(1, 'Modeling Human Dynamics and Breakdowns — Intelligent Agents for Int
ernet Games and Recruitment', 'Rajiv Khosla, Ishwar K. Sethi, Ernesto Damiani', 2
000, 'Intelligent Multimedia Multi-Agent Systems', 'Not found', '198-220', '10.10
07/978-1-4757-3196-5_8', 'https://doi.org/10.1007/978-1-4757-3196-5_8'), (16, 'Co
mbining Federated and Active Learning for Communication-efficient Distributed Fai
lure Prediction in Aeronautics', 'Nicolas Aussel, Sophie Chabridon, Yohan Peteti
n', 2020, 'arXiv', 'Not found', 'Not found', 'Not found', 'http://arxiv.org/abs/2
001.07504v1'), (43, 'GPT versus Humans: Uncovering Ethical Concerns in Conversati
onal Generative AI-empowered Multi-Robot Systems', 'Rebekah Rousi, Niko Makitalo,
Hooman Samani, Kai-Kristian Kemell, Jose Siqueira de Cerqueira, Ville Vakkuri, To
mmi Mikkonen, Pekka Abrahamsson', 2024, 'arXiv', 'Not found', 'Not found', 'Not f
ound', 'http://arxiv.org/abs/2411.14009v1'), (48, 'GPT versus Humans: Uncovering
Ethical Concerns in Conversational Generative AI-empowered Multi-Robot Systems',
'Rebekah Rousi, Niko Makitalo, Hooman Samani, Kai-Kristian Kemell, Jose Siqueira
de Cerqueira, Ville Vakkuri, Tommi Mikkonen, Pekka Abrahamsson', 2024, 'arXiv',
'Not found', 'Not found', 'Not found', 'http://arxiv.org/abs/2411.14009v1'), (65,
'GPT versus Humans: Uncovering Ethical Concerns in Conversational Generative AI-e
mpowered Multi-Robot Systems', 'Rebekah Rousi, Niko Makitalo, Hooman Samani, Kai-
Kristian Kemell, Jose Siqueira de Cerqueira, Ville Vakkuri, Tommi Mikkonen, Pekka
Abrahamsson', 2024, 'arXiv', 'Not found', 'Not found', 'Not found', 'http://arxi
v.org/abs/2411.14009v1')]
Semantic parsing completed at 2025-04-11 23:48:49.490083
Output: 2. Emergent Linguistic Phenomena in Multi-Agent Communication Games (201
9)
   Authors: Laura Harding Graesser, Kyunghyun Cho, Douwe Kiela
   Publication: Proceedings of the 2019 Conference on Empirical Methods in Natura
l Language Processing and the 9th International Joint Conference on Natural Langu
age Processing (EMNLP-IJCNLP)
   Issue: Not found
   Pages: 3698-3708
   DOI: 10.18653/v1/d19-1384
   URL: https://doi.org/10.18653/v1/d19-1384
   Abstract: Not found
   Reference Summary: This paper explores the emergence of linguistic phenomena i
n multi-agent communication games, focusing on how agents develop communication p
rotocols. It is relevant to the query as it addresses natural language processing
in multi-agent systems.

4. An Approach to Using XML and a Rule-Based Content Language with an Agent Commu
nication Language (2000)
   Authors: Benjamin N. Grosof, Yannis Labrou
   Publication: Lecture Notes in Computer Science
   Issue: Not found
   Pages: 96-117
   DOI: 10.1007/10722777_7
   URL: https://doi.org/10.1007/10722777_7
   Abstract: Not found
   Reference Summary: This paper discusses the use of XML and rule-based content
languages in conjunction with agent communication languages, providing insights i
nto rule-based logic for multi-agent communication.

5. Machine learning vs. rule-based methods for document classification of electro
nic health records within mental health care—A systematic literature review (202
5)
   Authors: Emil Rijcken, Kalliopi Zervanou, Pablo Mosteiro, Floortje Scheepers,
Marco Spruit, Uzay Kaymak
   Publication: Natural Language Processing Journal
   Issue: Not found
   Pages: 100129
   DOI: 10.1016/j.nlp.2025.100129
   URL: https://doi.org/10.1016/j.nlp.2025.100129
   Abstract: Not found
   Reference Summary: This systematic literature review compares machine learning
and rule-based methods for document classification, providing relevant insights i
nto the advantages and limitations of each approach, applicable to multi-agent sy
stem communication.

9. Certifying Choreography Compilation (2021)
   Authors: Luís Cruz-Filipe, Fabrizio Montesi, Marco Peressotti
   Publication: arXiv
   Issue: Not found
   Pages: Not found
   DOI: 10.1007/978-3-030-85315-0_8
   URL: http://arxiv.org/abs/2102.10698v2
   Abstract: Choreographic programming is a paradigm for developing concurrent an
d distributed systems, where programs are choreographies that define, from a glob
al viewpoint, the computations and interactions that communicating processes shou
ld enact. Choreography compilation translates choreographies into the local defin
itions of process behaviours, given as terms in a process calculus. Proving chore
ography compilation correct is challenging and error-prone, because it requires r
elating languages in different paradigms (global interactions vs local actions) a
nd dealing with a combinatorial explosion of proof cases. We present the first ce
rtified program for choreography compilation for a nontrivial choreographic langu
age supporting recursion.
   Reference Summary: This paper presents a certified program for choreography co
mpilation, relevant to multi-agent systems as it involves translating global inte
ractions into local actions, a key aspect of agent communication.

10. Distilling Text into Circuits (2023)
   Authors: Vincent Wang-Mascianica, Jonathon Liu, Bob Coecke
   Publication: arXiv
   Issue: Not found
   Pages: Not found
   DOI: Not found
   URL: http://arxiv.org/abs/2301.10595v1
   Abstract: This paper concerns the structure of meanings within natural languag
e. Earlier, a framework named DisCoCirc was sketched that (1) is compositional an
d distributional (a.k.a. vectorial); (2) applies to general text; (3) captures li
nguistic `connections' between meanings (cf. grammar) (4) updates word meanings a
s text progresses; (5) structures sentence types; (6) accommodates ambiguity. Her
e, we realise DisCoCirc for a substantial fragment of English. When passing to Di
sCoCirc's text circuits, some `grammatical bureaucracy' is eliminated, that is, D
isCoCirc displays a significant degree of (7) inter- and intra-language independe
nce. That is, e.g., independence from word-order conventions that differ across l
anguages, and independence from choices like many short sentences vs. few long se
ntences. This inter-language independence means our text circuits should carry ov
er to other languages, unlike the language-specific typings of categorial grammar
s. Hence, text circuits are a lean structure for the `actual substance of text',
that is, the inner-workings of meanings within text across several layers of expr
essiveness (cf. words, sentences, text), and may capture that what is truly unive
rsal beneath grammar. The elimination of grammatical bureaucracy also explains wh
y DisCoCirc: (8) applies beyond language, e.g. to spatial, visual and other cogni
tive modes. While humans could not verbally communicate in terms of text circuit
s, machines can. We first define a `hybrid grammar' for a fragment of English, i.
e. a purpose-built, minimal grammatical formalism needed to obtain text circuits.
We then detail a translation process such that all text generated by this grammar
yields a text circuit. Conversely, for any text circuit obtained by freely compos
ing the generators, there exists a text (with hybrid grammar) that gives rise to
it. Hence: (9) text circuits are generative for text.
```

Reference Summary: This paper introduces DisCoCirc, a framework for structuring meanings within natural language, relevant to multi-agent systems as it involves natural language processing and communication.

```
Out[ ]: "2. Emergent Linguistic Phenomena in Multi-Agent Communication Games (2019)\n
        Authors: Laura Harding Graesser, Kyunghyun Cho, Douwe Kiela\n   Publication: Pr
        oceedings of the 2019 Conference on Empirical Methods in Natural Language Proce
        ssing and the 9th International Joint Conference on Natural Language Processing
        (EMNLP-IJCNLP)\n   Issue: Not found\n   Pages: 3698-3708\n   DOI: 10.18653/v1/d
        19-1384\n   URL: https://doi.org/10.18653/v1/d19-1384\n   Abstract: Not found\n
        Reference Summary: This paper explores the emergence of linguistic phenomena in
        multi-agent communication games, focusing on how agents develop communication p
        rotocols. It is relevant to the query as it addresses natural language processi
        ng in multi-agent systems.\n\n4. An Approach to Using XML and a Rule-Based Cont
        ent Language with an Agent Communication Language (2000)\n   Authors: Benjamin
        N. Grosof, Yannis Labrou\n   Publication: Lecture Notes in Computer Science\n
        Issue: Not found\n   Pages: 96-117\n   DOI: 10.1007/10722777_7\n   URL: http
        s://doi.org/10.1007/10722777_7\n   Abstract: Not found\n   Reference Summary: T
        his paper discusses the use of XML and rule-based content languages in conjunct
        ion with agent communication languages, providing insights into rule-based logi
        c for multi-agent communication.\n\n5. Machine learning vs. rule-based methods
        for document classification of electronic health records within mental health c
        are—A systematic literature review (2025)\n   Authors: Emil Rijcken, Kalliopi Z
        ervanou, Pablo Mosteiro, Floortje Scheepers, Marco Spruit, Uzay Kaymak\n   Publ
        ication: Natural Language Processing Journal\n   Issue: Not found\n   Pages: 10
        0129\n   DOI: 10.1016/j.nlp.2025.100129\n   URL: https://doi.org/10.1016/j.nlp.
        2025.100129\n   Abstract: Not found\n   Reference Summary: This systematic lite
        rature review compares machine learning and rule-based methods for document cla
        ssification, providing relevant insights into the advantages and limitations of
        each approach, applicable to multi-agent system communication.\n\n9. Certifying
        Choreography Compilation (2021)\n   Authors: Luís Cruz-Filipe, Fabrizio Montes
        i, Marco Peressotti\n   Publication: arXiv\n   Issue: Not found\n   Pages: Not
        found\n   DOI: 10.1007/978-3-030-85315-0_8\n   URL: http://arxiv.org/abs/2102.1
        0698v2\n   Abstract: Choreographic programming is a paradigm for developing con
        current and distributed systems, where programs are choreographies that define,
        from a global viewpoint, the computations and interactions that communicating p
        rocesses should enact. Choreography compilation translates choreographies into
        the local definitions of process behaviours, given as terms in a process calcul
        us. Proving choreography compilation correct is challenging and error-prone, be
        cause it requires relating languages in different paradigms (global interaction
        s vs local actions) and dealing with a combinatorial explosion of proof cases.
        We present the first certified program for choreography compilation for a nontr
        ivial choreographic language supporting recursion.\n   Reference Summary: This
        paper presents a certified program for choreography compilation, relevant to mu
        lti-agent systems as it involves translating global interactions into local act
        ions, a key aspect of agent communication.\n\n10. Distilling Text into Circuits
        (2023)\n   Authors: Vincent Wang-Mascianica, Jonathon Liu, Bob Coecke\n   Publi
        cation: arXiv\n   Issue: Not found\n   Pages: Not found\n   DOI: Not found\n
        URL: http://arxiv.org/abs/2301.10595v1\n   Abstract: This paper concerns the st
        ructure of meanings within natural language. Earlier, a framework named DisCoCi
        rc was sketched that (1) is compositional and distributional (a.k.a. vectoria
        l); (2) applies to general text; (3) captures linguistic `connections' between
        meanings (cf. grammar) (4) updates word meanings as text progresses; (5) struct
        ures sentence types; (6) accommodates ambiguity. Here, we realise DisCoCirc for
        a substantial fragment of English. When passing to DisCoCirc's text circuits, s
        ome `grammatical bureaucracy' is eliminated, that is, DisCoCirc displays a sign
        ificant degree of (7) inter- and intra-language independence. That is, e.g., in
        dependence from word-order conventions that differ across languages, and indepe
        ndence from choices like many short sentences vs. few long sentences. This inte
        r-language independence means our text circuits should carry over to other lang
        uages, unlike the language-specific typings of categorial grammars. Hence, text
        circuits are a lean structure for the `actual substance of text', that is, the
        inner-workings of meanings within text across several layers of expressiveness
        (cf. words, sentences, text), and may capture that what is truly universal bene
```

```
ath grammar. The elimination of grammatical bureaucracy also explains why DisCo
Circ: (8) applies beyond language, e.g. to spatial, visual and other cognitive
modes. While humans could not verbally communicate in terms of text circuits, m
achines can. We first define a `hybrid grammar' for a fragment of English, i.e.
a purpose-built, minimal grammatical formalism needed to obtain text circuits.
We then detail a translation process such that all text generated by this gramm
ar yields a text circuit. Conversely, for any text circuit obtained by freely c
omposing the generators, there exists a text (with hybrid grammar) that gives r
ise to it. Hence: (9) text circuits are generative for text.\n   Reference Summ
ary: This paper introduces DisCoCirc, a framework for structuring meanings with
in natural language, relevant to multi-agent systems as it involves natural lan
guage processing and communication."
```

Referencing agent

In [ ]:
```
# Referencing agent
# Converts selected references into structured dictionaries suitable for validat



def referencing_agent(input_text):


    """
Parses a structured reference list from an LLM-generated string into Python dict

Args:
    input_text (str): The LLM-generated textual reference list.

Returns:
    list: A list of dictionaries representing academic references, with metadata
"""





    prompt_template2 = PromptTemplate(
    template="""Extract the academic references from the following text and form
    - Authors
    - Year
    - Title
    - Publication
    - Issue (if available)
    - Pages (if available)
    - DOI
    - URL
    - Reference_summary

    Text: {text}

    Provide your response starting directly with "(", and do not include any for
    """,
    input_variables=["text"]
)


    llm_chain = prompt_template2 | llm | output_parser

    result = llm_chain.invoke(input=input_text)
    # global formatted_reference_list

    formatted_reference_list=ast.literal_eval(result)

    print(f"Referencing completed at {datetime.now()}")
    print(f"Output: {formatted_reference_list}\n\n")

    return formatted_reference_list
```

In [ ]:
```
# Unit testing for referencing agent

referencing_agent(response)

formatted_reference_list # ensure this is stored in the global environment for t
```

```
Referencing completed at 2025-04-11 23:25:50.822896
Output: [{'Authors': 'Laura Harding Graesser, Kyunghyun Cho, Douwe Kiela', 'Yea
r': 2019, 'Title': 'Emergent Linguistic Phenomena in Multi-Agent Communication Ga
mes', 'Publication': 'Proceedings of the 2019 Conference on Empirical Methods in
Natural Language Processing and the 9th International Joint Conference on Natural
Language Processing (EMNLP-IJCNLP)', 'Issue': 'Not found', 'Pages': '3698-3708',
'DOI': '10.18653/v1/d19-1384', 'URL': 'https://doi.org/10.18653/v1/d19-1384', 'Re
ference_summary': 'This paper explores emergent linguistic phenomena in multi-age
nt communication games, which is relevant to understanding how natural language p
rocessing can be applied to multi-agent systems.'}, {'Authors': 'Benjamin N. Gros
of, Yannis Labrou', 'Year': 2000, 'Title': 'An Approach to Using XML and a Rule-B
ased Content Language with an Agent Communication Language', 'Publication': 'Lect
ure Notes in Computer Science', 'Issue': 'Not found', 'Pages': '96-117', 'DOI':
'10.1007/10722777_7', 'URL': 'https://doi.org/10.1007/10722777_7', 'Reference_sum
mary': 'This paper discusses the use of XML and a rule-based content language in
conjunction with an agent communication language, providing insights into rule-ba
sed logic for multi-agent system communication.'}, {'Authors': 'Emil Rijcken, Kal
liopi Zervanou, Pablo Mosteiro, Floortje Scheepers, Marco Spruit, Uzay Kaymak',
'Year': 2025, 'Title': 'Machine learning vs. rule-based methods for document clas
sification of electronic health records within mental health care—A systematic li
terature review', 'Publication': 'Natural Language Processing Journal', 'Issue':
'Not found', 'Pages': '100129', 'DOI': '10.1016/j.nlp.2025.100129', 'URL': 'http
s://doi.org/10.1016/j.nlp.2025.100129', 'Reference_summary': 'This systematic lit
erature review compares machine learning and rule-based methods, which is pertine
nt to evaluating natural language processing versus rule-based logic in multi-age
nt systems.'}, {'Authors': 'Luís Cruz-Filipe, Fabrizio Montesi, Marco Peressott
i', 'Year': 2021, 'Title': 'Certifying Choreography Compilation', 'Publication':
'arXiv', 'Issue': 'Not found', 'Pages': 'Not found', 'DOI': '10.1007/978-3-030-85
315-0_8', 'URL': 'http://arxiv.org/abs/2102.10698v2', 'Reference_summary': 'This
paper discusses choreography compilation in distributed systems, which is relevan
t to understanding communication in multi-agent systems.'}, {'Authors': 'Vincent
Wang-Mascianica, Jonathon Liu, Bob Coecke', 'Year': 2023, 'Title': 'Distilling Te
xt into Circuits', 'Publication': 'arXiv', 'Issue': 'Not found', 'Pages': 'Not fo
und', 'DOI': 'Not found', 'URL': 'http://arxiv.org/abs/2301.10595v1', 'Reference_
summary': 'This paper presents a framework for understanding the structure of mea
nings within natural language, which is relevant to natural language processing i
n multi-agent systems.'}]
```

```
Out[ ]: [{'Authors': 'Laura Harding Graesser, Kyunghyun Cho, Douwe Kiela',
          'Year': 2019,
          'Title': 'Emergent Linguistic Phenomena in Multi-Agent Communication Games',
          'Publication': 'Proceedings of the 2019 Conference on Empirical Methods in Na
        tural Language Processing and the 9th International Joint Conference on Natural
        Language Processing (EMNLP-IJCNLP)',
          'Issue': 'Not found',
          'Pages': '3698-3708',
          'DOI': '10.18653/v1/d19-1384',
          'URL': 'https://doi.org/10.18653/v1/d19-1384',
          'Reference_summary': 'This paper explores emergent linguistic phenomena in mu
        lti-agent communication games, which is relevant to understanding how natural l
        anguage processing can be applied to multi-agent systems.'},
         {'Authors': 'Benjamin N. Grosof, Yannis Labrou',
          'Year': 2000,
          'Title': 'An Approach to Using XML and a Rule-Based Content Language with an
        Agent Communication Language',
          'Publication': 'Lecture Notes in Computer Science',
          'Issue': 'Not found',
          'Pages': '96-117',
          'DOI': '10.1007/10722777_7',
          'URL': 'https://doi.org/10.1007/10722777_7',
          'Reference_summary': 'This paper discusses the use of XML and a rule-based co
        ntent language in conjunction with an agent communication language, providing i
        nsights into rule-based logic for multi-agent system communication.'},
         {'Authors': 'Emil Rijcken, Kalliopi Zervanou, Pablo Mosteiro, Floortje Scheepe
        rs, Marco Spruit, Uzay Kaymak',
          'Year': 2025,
          'Title': 'Machine learning vs. rule-based methods for document classification
        of electronic health records within mental health care—A systematic literature
        review',
          'Publication': 'Natural Language Processing Journal',
          'Issue': 'Not found',
          'Pages': '100129',
          'DOI': '10.1016/j.nlp.2025.100129',
          'URL': 'https://doi.org/10.1016/j.nlp.2025.100129',
          'Reference_summary': 'This systematic literature review compares machine lear
        ning and rule-based methods, which is pertinent to evaluating natural language
        processing versus rule-based logic in multi-agent systems.'},
         {'Authors': 'Luís Cruz-Filipe, Fabrizio Montesi, Marco Peressotti',
          'Year': 2021,
          'Title': 'Certifying Choreography Compilation',
          'Publication': 'arXiv',
          'Issue': 'Not found',
          'Pages': 'Not found',
          'DOI': '10.1007/978-3-030-85315-0_8',
          'URL': 'http://arxiv.org/abs/2102.10698v2',
          'Reference_summary': 'This paper discusses choreography compilation in distri
        buted systems, which is relevant to understanding communication in multi-agent
        systems.'},
         {'Authors': 'Vincent Wang-Mascianica, Jonathon Liu, Bob Coecke',
          'Year': 2023,
          'Title': 'Distilling Text into Circuits',
          'Publication': 'arXiv',
          'Issue': 'Not found',
          'Pages': 'Not found',
          'DOI': 'Not found',
          'URL': 'http://arxiv.org/abs/2301.10595v1',
          'Reference_summary': 'This paper presents a framework for understanding the s
        tructure of meanings within natural language, which is relevant to natural lang
        uage processing in multi-agent systems.'}]
```

Validation agent

```
In [ ]: # Validation agent


        ## validation scheme
        # Define a Pydantic model for reference validation
        """Pydantic allows creation of reference schemas that can be used to validate da

        class Reference(BaseModel):
            Authors: str
            Year: Optional[int] = None # so that reference can still be returned if the
            Title: str
            Publication: str
            Issue: Optional[Union[str, int]] = None  # Optional field (may not be presen
            Pages: str
            DOI: Optional[str] = None # May not be provided by the LLM if it does not fi
            URL: Optional[Union[HttpUrl, str]] = None # Enforces valid URL format, but o
            Reference_summary: str

            @validator('Issue')
            def validate_issue(cls, v):
                if isinstance(v, int):
                    return str(v)
                return v


            @validator('URL')
            def validate_url(cls, v):
                if v == "Not found":
                    return v
                if v is not None:
                    return HttpUrl(v)
                return v


        # Validation agent structure
        # Validates structured references using a Pydantic model and rechecks via an LLM

        def validation_agent(input_references):

            """
            Validates reference format using a Pydantic model and rechecks via an LLM to

            Args:
                input_references (list): List of reference dictionaries.

            Returns:
                list: Only those references that passed validation, formatted for storag
            """




        # Define a prompt for the LLM
            validation_prompt = PromptTemplate(
            template="""Validate the references provided and ensure they follow the corr
            The format is as follows:
            - Authors
            - Year
            - Title
            - Publication
            - Issue (if available)
            - Pages
            - DOI
            - URL
            - Reference summary

            You will return a Python list with the references in the correct format, and
            """,








            input_variables=["references"])

            global validated_refs, invalid_references

            validated_refs = []
            invalid_references = []

            for ref in input_references:
                try:
                    validated_refs.append(Reference(**ref))  # Validate and create Refer
                except ValueError as e:
                    print(f"Invalid reference found: {ref} - Error: {e}.")
                    invalid_references.append(ref)

            # Convert validated references to strings for LLM parsing
            global validated_references

            validated_references = [ref.model_dump() for ref in validated_refs]

            # Prepare the LLM chain for cross-checking
            global response

            llm_chain = validation_prompt | llm
            response = llm_chain.invoke(input={"references": validated_references}).cont

            print(f"Reference validation completed at {datetime.now()}")
            print(f"Output: {validated_references}\n\n")
            return validated_references
```

```
C:\Users\knfc648\AppData\Local\Temp\ipykernel_14800\850272855.py:21: PydanticDepr
ecatedSince20: Pydantic V1 style `@validator` validators are deprecated. You shou
ld migrate to Pydantic V2 style `@field_validator` validators, see the migration
guide for more details. Deprecated in Pydantic V2.0 to be removed in V3.0. See Py
dantic V2 Migration Guide at https://errors.pydantic.dev/2.11/migration/
  @validator('Issue')
C:\Users\knfc648\AppData\Local\Temp\ipykernel_14800\850272855.py:28: PydanticDepr
ecatedSince20: Pydantic V1 style `@validator` validators are deprecated. You shou
ld migrate to Pydantic V2 style `@field_validator` validators, see the migration
guide for more details. Deprecated in Pydantic V2.0 to be removed in V3.0. See Py
dantic V2 Migration Guide at https://errors.pydantic.dev/2.11/migration/
  @validator('URL')
```

In [ ]:
```
# Validation agent structure
# Validates structured references using a Pydantic model and rechecks via an LLM

def validation_agent(input_references):

    """
    Validates reference format using a Pydantic model and rechecks via an LLM to

    Args:
        input_references (list): List of reference dictionaries.

    Returns:
        list: Only those references that passed validation, formatted for storag
    """




# Define a prompt for the LLM
    validation_prompt = PromptTemplate(
    template="""Validate the references provided and ensure they follow the corr
    The format is as follows:
    - Authors
    - Year
    - Title
    - Publication
    - Issue (if available)
    - Pages
    - DOI
    - URL
    - Reference summary

    You will return a Python list with the references in the correct format, and
    """,






    input_variables=["references"])

    global validated_refs, invalid_references

    validated_refs = []
    invalid_references = []
    for ref in input_references:
        try:
            validated_refs.append(Reference(**ref))  # Validate and create Refer
        except ValueError as e:
            print(f"Invalid reference found: {ref} - Error: {e}.")
            invalid_references.append(ref)

    # Convert validated references to strings for LLM parsing
    global validated_references

    validated_references = [ref.model_dump() for ref in validated_refs]

    # Prepare the LLM chain for cross-checking
    global response

    llm_chain = validation_prompt | llm
    response = llm_chain.invoke(input={"references": validated_references}).cont

    print(f"Reference validation completed at {datetime.now()}")
    print(f"Output: {validated_references}\n\n")
    return validated_references
```

In [ ]:

```
# Unit testing

validation_agent(formatted_reference_list)

validated_references
```

```
Reference validation completed at 2025-04-11 23:27:44.904112
Output: [{'Authors': 'Laura Harding Graesser, Kyunghyun Cho, Douwe Kiela', 'Yea
r': 2019, 'Title': 'Emergent Linguistic Phenomena in Multi-Agent Communication Ga
mes', 'Publication': 'Proceedings of the 2019 Conference on Empirical Methods in
Natural Language Processing and the 9th International Joint Conference on Natural
Language Processing (EMNLP-IJCNLP)', 'Issue': 'Not found', 'Pages': '3698-3708',
'DOI': '10.18653/v1/d19-1384', 'URL': HttpUrl('https://doi.org/10.18653/v1/d19-13
84'), 'Reference_summary': 'This paper explores emergent linguistic phenomena in
multi-agent communication games, which is relevant to understanding how natural l
anguage processing can be applied to multi-agent systems.'}, {'Authors': 'Benjami
n N. Grosof, Yannis Labrou', 'Year': 2000, 'Title': 'An Approach to Using XML and
a Rule-Based Content Language with an Agent Communication Language', 'Publicatio
n': 'Lecture Notes in Computer Science', 'Issue': 'Not found', 'Pages': '96-117',
'DOI': '10.1007/10722777_7', 'URL': HttpUrl('https://doi.org/10.1007/10722777_
7'), 'Reference_summary': 'This paper discusses the use of XML and a rule-based c
ontent language in conjunction with an agent communication language, providing in
sights into rule-based logic for multi-agent system communication.'}, {'Authors':
'Emil Rijcken, Kalliopi Zervanou, Pablo Mosteiro, Floortje Scheepers, Marco Sprui
t, Uzay Kaymak', 'Year': 2025, 'Title': 'Machine learning vs. rule-based methods
for document classification of electronic health records within mental health car
e—A systematic literature review', 'Publication': 'Natural Language Processing Jo
urnal', 'Issue': 'Not found', 'Pages': '100129', 'DOI': '10.1016/j.nlp.2025.10012
9', 'URL': HttpUrl('https://doi.org/10.1016/j.nlp.2025.100129'), 'Reference_summa
ry': 'This systematic literature review compares machine learning and rule-based
methods, which is pertinent to evaluating natural language processing versus rule
-based logic in multi-agent systems.'}, {'Authors': 'Luís Cruz-Filipe, Fabrizio M
ontesi, Marco Peressotti', 'Year': 2021, 'Title': 'Certifying Choreography Compil
ation', 'Publication': 'arXiv', 'Issue': 'Not found', 'Pages': 'Not found', 'DO
I': '10.1007/978-3-030-85315-0_8', 'URL': HttpUrl('http://arxiv.org/abs/2102.1069
8v2'), 'Reference_summary': 'This paper discusses choreography compilation in dis
tributed systems, which is relevant to understanding communication in multi-agent
systems.'}, {'Authors': 'Vincent Wang-Mascianica, Jonathon Liu, Bob Coecke', 'Yea
r': 2023, 'Title': 'Distilling Text into Circuits', 'Publication': 'arXiv', 'Issu
e': 'Not found', 'Pages': 'Not found', 'DOI': 'Not found', 'URL': HttpUrl('htt
p://arxiv.org/abs/2301.10595v1'), 'Reference_summary': 'This paper presents a fra
mework for understanding the structure of meanings within natural language, which
is relevant to natural language processing in multi-agent systems.'}]
```

```
Out[ ]: [{'Authors': 'Laura Harding Graesser, Kyunghyun Cho, Douwe Kiela',
          'Year': 2019,
          'Title': 'Emergent Linguistic Phenomena in Multi-Agent Communication Games',
          'Publication': 'Proceedings of the 2019 Conference on Empirical Methods in Na
        tural Language Processing and the 9th International Joint Conference on Natural
        Language Processing (EMNLP-IJCNLP)',
          'Issue': 'Not found',
          'Pages': '3698-3708',
          'DOI': '10.18653/v1/d19-1384',
          'URL': HttpUrl('https://doi.org/10.18653/v1/d19-1384'),
          'Reference_summary': 'This paper explores emergent linguistic phenomena in mu
        lti-agent communication games, which is relevant to understanding how natural l
        anguage processing can be applied to multi-agent systems.'},
         {'Authors': 'Benjamin N. Grosof, Yannis Labrou',
          'Year': 2000,
          'Title': 'An Approach to Using XML and a Rule-Based Content Language with an
        Agent Communication Language',
          'Publication': 'Lecture Notes in Computer Science',
          'Issue': 'Not found',
          'Pages': '96-117',
          'DOI': '10.1007/10722777_7',
          'URL': HttpUrl('https://doi.org/10.1007/10722777_7'),
          'Reference_summary': 'This paper discusses the use of XML and a rule-based co
        ntent language in conjunction with an agent communication language, providing i
        nsights into rule-based logic for multi-agent system communication.'},
         {'Authors': 'Emil Rijcken, Kalliopi Zervanou, Pablo Mosteiro, Floortje Scheepe
        rs, Marco Spruit, Uzay Kaymak',
          'Year': 2025,
          'Title': 'Machine learning vs. rule-based methods for document classification
        of electronic health records within mental health care—A systematic literature
        review',
          'Publication': 'Natural Language Processing Journal',
          'Issue': 'Not found',
          'Pages': '100129',
          'DOI': '10.1016/j.nlp.2025.100129',
          'URL': HttpUrl('https://doi.org/10.1016/j.nlp.2025.100129'),
          'Reference_summary': 'This systematic literature review compares machine lear
        ning and rule-based methods, which is pertinent to evaluating natural language
        processing versus rule-based logic in multi-agent systems.'},
         {'Authors': 'Luís Cruz-Filipe, Fabrizio Montesi, Marco Peressotti',
          'Year': 2021,
          'Title': 'Certifying Choreography Compilation',
          'Publication': 'arXiv',
          'Issue': 'Not found',
          'Pages': 'Not found',
          'DOI': '10.1007/978-3-030-85315-0_8',
          'URL': HttpUrl('http://arxiv.org/abs/2102.10698v2'),
          'Reference_summary': 'This paper discusses choreography compilation in distri
        buted systems, which is relevant to understanding communication in multi-agent
        systems.'},
         {'Authors': 'Vincent Wang-Mascianica, Jonathon Liu, Bob Coecke',
          'Year': 2023,
          'Title': 'Distilling Text into Circuits',
          'Publication': 'arXiv',
          'Issue': 'Not found',
          'Pages': 'Not found',
          'DOI': 'Not found',
          'URL': HttpUrl('http://arxiv.org/abs/2301.10595v1'),
          'Reference_summary': 'This paper presents a framework for understanding the s
        tructure of meanings within natural language, which is relevant to natural lang
        uage processing in multi-agent systems.'}]
```

Create data storage

In [ ]:
```
# Initialize the SentenceTransformer model
model = SentenceTransformer('all-MiniLM-L6-v2') # lightweight, fast, open-source
```

In [ ]:
```
# Function to create SQLite database / connect to existing database
def create_connection(db_file):
    conn = sqlite3.connect(db_file)
    return conn
```

In [ ]:
```
# Function to create a table for storing search results (including vector embedd
def create_table_results():
    if 'conn' in globals():

        cursor = conn.cursor()
        cursor.execute('''
        CREATE TABLE IF NOT EXISTS search_results (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            title TEXT,
            authors TEXT,
            year INT,
            publication TEXT,
            issue TEXT,
            pages TEXT,
            doi TEXT,
            url TEXT,
            reference_summary TEXT,
            embedding BLOB,
            user TEXT,
            prompt TEXT,
            search_date DATE,
            timestamp TIME
            )
        ''')
        conn.commit()
    else:
        raise ValueError("Database connection 'conn' is not defined in the globa
```

In [ ]:
```
# Function to insert a search result into the database
import csv

def save_search_result(references):
    if 'conn' in globals():

        summary_texts = [ref['Reference_summary'] for ref in references]
        embeddings = model.encode(summary_texts)

        cursor = conn.cursor()


        from datetime import date, datetime


        search_date  = date.today().strftime("%Y-%m-%d")
        timestamp = datetime.now().strftime("%H-%M-%S")

        for i, ref in enumerate(references):
            authors = ref['Authors']
            year = ref['Year']
            title = ref['Title']
            publication = ref['Publication']
            issue = ref['Issue']
            pages = ref['Pages']
            doi = ref['DOI']
            url = str(ref['URL']) if ref['URL'] else None # circumvent issue wit
            reference_summary = ref['Reference_summary']
            embedding = embeddings[i]
            cursor.execute('INSERT INTO search_results (title, authors, year, pu
            # Store reference metadata and embedding as binary blob in SQLite da


        conn.commit()
        print(f"Finished saving search results to database at {datetime.now()}\n

    else:
        raise ValueError("Database connection 'conn' is not defined in the globa


    # save search results as csv for rapid extraction and reuse by user

    try:
        target_dir = os.path.join(os.getcwd(), "SAGE results")
        print(f"Attempting to create directory: {target_dir}")
        os.makedirs(target_dir, exist_ok=True)
        print(f"Directory creation successful: {target_dir}")
    except Exception as e:
        print(f"Error creating directory: {e}")
        print(f"Will attempt to save file in current directory: {os.getcwd()}")
        target_dir = os.getcwd()

    # Use the target_dir for the filename
    filename = os.path.join(target_dir, f"SAGE_search_results_{search_date}_{tim
    print(f"Attempting to save file: {filename}")


    # convert HttpUrl to string
    def httpurl_to_str(obj):
        if isinstance(obj, HttpUrl):
            return str(obj)
        return obj

    with open(filename, 'w', newline='', encoding='utf-8') as file:
        fieldnames = references[0].keys()
        writer = csv.DictWriter(file, fieldnames=fieldnames)
        writer.writeheader()

        for row in references:
            row_str = {k: httpurl_to_str(v) for k, v in row.items()}
            writer.writerow(row_str)

    print(f"Finished saving search results to .csv file at {datetime.now()}\n\n"

    return references
```

```
In [ ]: # Function to fetch search results from the database

        def fetch_search_results(reference_ids):
            """Extracts structured reference information from database, after relevant r
            if 'conn' in globals():
                cursor = conn.cursor()

                id_list = ', '.join(str(id) for id in reference_ids)

                query = f'SELECT id, title, authors, year, publication, issue, pages, do
                cursor.execute(query)
                rows = cursor.fetchall()

                result = [row[0:9] for row in rows]
                return result
            else:
                raise ValueError("Database connection 'conn' is not defined in the globa
```

```
In [ ]: # Function to fetch only reference ID and embedding from search results (for con

        def fetch_embeddings():
            """Extracts reference IDs and embeddings for all references stored in the da


            if 'conn' in globals():
                cursor = conn.cursor()
                cursor.execute('SELECT id, embedding FROM search_results')
                rows = cursor.fetchall()
                result = [(row[0], np.frombuffer(row[1], dtype=np.float32)) for row in r
                return result
            else:
                raise ValueError("Database connection 'conn' is not defined in the globa
```

```
In [ ]: # # database query (helper function)

        def database_query(query):
            if 'conn' in globals():

                cursor = conn.cursor()
                cursor.execute(f"{query}")
                results=cursor.fetchall()
                return(results)
                print(results)
            else:
                raise ValueError("Database connection 'conn' is not defined in the globa
```

```
In [ ]: # Unit testing
        conn = create_connection('SAGE_database.db')
        create_table_results()
        print(database_query("SELECT name FROM sqlite_master WHERE type='table';"))
```

```
[('sqlite_sequence',), ('search_results',)]
```

```
In [ ]: # Unit testing
        # store search results
        username="gamorim"
        user_prompt=test_search_prompt
```

```
save_search_result(validated_references)
print(database_query("SELECT distinct id  FROM search_results"))
```

```
Finished saving search results to database at 2025-04-11 23:30:44.812668
Attempting to create directory: c:\Users\knfc648\Documents\Personal\PgDip\portfol
io\portfolio_pgdip\module4\Individual project\SAGE results
Directory creation successful: c:\Users\knfc648\Documents\Personal\PgDip\portfoli
o\portfolio_pgdip\module4\Individual project\SAGE results
Attempting to save file: c:\Users\knfc648\Documents\Personal\PgDip\portfolio\port
folio_pgdip\module4\Individual project\SAGE results\SAGE_search_results_2025-04-1
1_23-30-44.csv
Finished saving search results to .csv file at 2025-04-11 23:30:44.814667
[(1,), (2,), (3,), (4,), (5,), (6,), (7,), (8,), (9,), (10,), (11,), (12,), (1
3,), (14,), (15,), (16,), (17,), (18,), (19,), (20,), (21,), (22,), (23,), (24,),
(25,)]
```

Context aware search

In [ ]:

```
# Context-aware search (via vector search) to include into search agent


def context_aware_search(user_prompt):
    """Performs similarity search on stored reference summaries based on the sea
    Returns reference IDs that can be used to pull more detailed and structure i

    Arguments:
    - user_prompt: prompt provided by the user"""

    if 'conn' not in globals() or conn is None:
        print("Connection to the database could not be established.") # for the
        return None

    try:
        cursor = conn.cursor()

        # Fetch previous search results (but ID and vectors only)
        database_results = fetch_embeddings()

        stored_embeddings = []

        for ref in database_results:
            reference_id = ref[0] # need reference ID so we can later pull refer
            embeddings = ref[1]
            stored_embeddings.append((reference_id, embeddings))

        stored_embeddings_vectors = np.array([row[1] for row in stored_embedding

        # Vectorise query
        query_vector = model.encode([user_prompt])[0]
        # Encode the current user prompt to a vector representation


        # Find the nearest neighbor from the stored embeddings
        nn = NearestNeighbors(n_neighbors=5,metric='cosine')  # selecting 5 neig
        nn.fit(stored_embeddings_vectors)
        nearest_distances, nearest_indices = nn.kneighbors([query_vector])

        # Output the nearest neighbours
        nearest_ids = [database_results[i][0] for i in nearest_indices.flatten()
        # Extract the original reference IDs from nearest neighbors for lookup


        nearest_references = fetch_search_results(nearest_ids)
        print(f"Context search completed at {datetime.now()}")
        print(f"Output: {nearest_references}\n\n")

        return nearest_references

    except Exception as e:
        print(f"An error occurred: {e}")
        return None
```

In [ ]:
```
# Unit testing with example query
test_query = "communication among software agents"

context_aware_search(test_query)
```

```
Context search completed at 2025-04-11 23:32:03.967722
Output: [(21, 'Emergent Linguistic Phenomena in Multi-Agent Communication Games',
'Laura Harding Graesser, Kyunghyun Cho, Douwe Kiela', 2019, 'Proceedings of the 2
019 Conference on Empirical Methods in Natural Language Processing and the 9th In
ternational Joint Conference on Natural Language Processing (EMNLP-IJCNLP)', 'Not
found', '3698-3708', '10.18653/v1/d19-1384', 'https://doi.org/10.18653/v1/d19-138
4'), (22, 'An Approach to Using XML and a Rule-Based Content Language with an Age
nt Communication Language', 'Benjamin N. Grosof, Yannis Labrou', 2000, 'Lecture N
otes in Computer Science', 'Not found', '96-117', '10.1007/10722777_7', 'https://
doi.org/10.1007/10722777_7'), (23, 'Machine learning vs. rule-based methods for d
ocument classification of electronic health records within mental health care—A s
ystematic literature review', 'Emil Rijcken, Kalliopi Zervanou, Pablo Mosteiro, F
loortje Scheepers, Marco Spruit, Uzay Kaymak', 2025, 'Natural Language Processing
Journal', 'Not found', '100129', '10.1016/j.nlp.2025.100129', 'https://doi.org/1
0.1016/j.nlp.2025.100129'), (24, 'Certifying Choreography Compilation', 'Luís Cru
z-Filipe, Fabrizio Montesi, Marco Peressotti', 2021, 'arXiv', 'Not found', 'Not f
ound', '10.1007/978-3-030-85315-0_8', 'http://arxiv.org/abs/2102.10698v2'), (25,
'Distilling Text into Circuits', 'Vincent Wang-Mascianica, Jonathon Liu, Bob Coec
ke', 2023, 'arXiv', 'Not found', 'Not found', 'Not found', 'http://arxiv.org/abs/
2301.10595v1')]
```

```
Out[ ]: [(21,
  'Emergent Linguistic Phenomena in Multi-Agent Communication Games',
  'Laura Harding Graesser, Kyunghyun Cho, Douwe Kiela',
  2019,
  'Proceedings of the 2019 Conference on Empirical Methods in Natural Language
Processing and the 9th International Joint Conference on Natural Language Proce
ssing (EMNLP-IJCNLP)',
  'Not found',
  '3698-3708',
  '10.18653/v1/d19-1384',
  'https://doi.org/10.18653/v1/d19-1384'),
 (22,
  'An Approach to Using XML and a Rule-Based Content Language with an Agent Com
munication Language',
  'Benjamin N. Grosof, Yannis Labrou',
  2000,
  'Lecture Notes in Computer Science',
  'Not found',
  '96-117',
  '10.1007/10722777_7',
  'https://doi.org/10.1007/10722777_7'),
 (23,
  'Machine learning vs. rule-based methods for document classification of elect
ronic health records within mental health care—A systematic literature review',
  'Emil Rijcken, Kalliopi Zervanou, Pablo Mosteiro, Floortje Scheepers, Marco S
pruit, Uzay Kaymak',
  2025,
  'Natural Language Processing Journal',
  'Not found',
  '100129',
  '10.1016/j.nlp.2025.100129',
  'https://doi.org/10.1016/j.nlp.2025.100129'),
 (24,
  'Certifying Choreography Compilation',
  'Luís Cruz-Filipe, Fabrizio Montesi, Marco Peressotti',
  2021,
  'arXiv',
  'Not found',
  'Not found',
  '10.1007/978-3-030-85315-0_8',
  'http://arxiv.org/abs/2102.10698v2'),
 (25,
  'Distilling Text into Circuits',
  'Vincent Wang-Mascianica, Jonathon Liu, Bob Coecke',
  2023,
  'arXiv',
  'Not found',
  'Not found',
  'Not found',
  'http://arxiv.org/abs/2301.10595v1')]
```

Create GUI

Initial username and prompt request

In [ ]:
```
# Create GUI

# GUI input prompt to collect the username, API key, and user query for the sear


# custom dialog box so I can expand it when dealing with larger inputs
class CustomDialog(simpledialog.Dialog):
    def body(self, master):
        tk.Label(master, text="Please enter your prompt:").grid(row=0)
        self.e1 = tk.Text(master, height=10, width=50)  # Increased height and w
        self.e1.grid(row=1, padx=5, pady=5)
        return self.e1  # initial focus

    def apply(self):
        self.result = self.e1.get("1.0", tk.END).strip()




# Function to get user inputs
def get_user_input():

    # Create a Tk root widget
    root = tk.Tk()
    root.title("SAGE - Scalable Academic Goal-Driven Explorer")
    root.geometry("")
    frame = tk.Frame(root)
    frame.pack(padx=10, pady=10)
    label = tk.Label(frame, text="Welcome to SAGE, Scalable Academic Goal-Driven
    label.pack()
    root.update_idletasks()

    # Ask for the username
    username = simpledialog.askstring("SAGE", "Please enter your username:", par

    # Ask for the API
    global openai_api_key
    openai_api_key = simpledialog.askstring("SAGE", "Please enter your OpenAI AP

    # Ask for the prompt
    prompt_dialog = CustomDialog(root, title="SAGE")
    user_prompt = prompt_dialog.result

    # Close the Tkinter root window
    root.destroy()

    print(f"Initial GUI input retrieval completed at at {datetime.now()}\n\n")

    return username, openai_api_key, user_prompt
```

Return results to user

In [ ]:
```
# GUI output window to display validated reference results and explain where the

class SearchResultsWindow:
    def __init__(self, results):
        self.root = tk.Tk()
        self.root.title("Search Results")
        self.root.geometry("1000x600")  # Set initial window size

        # Add a message label at the top
        message="""Here are your search results.\nThese have been added to your
        self.message_label = tk.Label(self.root, text=message, font=("Helvetica"
        self.message_label.pack(pady=10, padx=10, fill=tk.X)



        # create scrollbar
        style = ttk.Style()
        style.theme_use('default')
        style.configure("Custom.Vertical.TScrollbar",
                        troughcolor='#F0F0F0',
                        background='#4A4A4A',
                        arrowcolor='#4A4A4A',
                        bordercolor='#4A4A4A',
                        lightcolor='#4A4A4A',
                        darkcolor='#4A4A4A')

        # Create main frame
        main_frame = ttk.Frame(self.root)
        main_frame.pack(fill=tk.BOTH, expand=1)

        # Create canvas
        self.canvas = tk.Canvas(main_frame)
        self.canvas.pack(side=tk.LEFT, fill=tk.BOTH, expand=1)

        # Add scrollbar to the canvas with the custom style
        scrollbar = ttk.Scrollbar(main_frame, orient=tk.VERTICAL, command=self.c
        scrollbar.pack(side=tk.RIGHT, fill=tk.Y)

        # Configure the canvas
        self.canvas.configure(yscrollcommand=scrollbar.set)
        self.canvas.bind('<Configure>', lambda e: self.canvas.configure(scrollre

        # Create another frame inside the canvas
        self.frame = ttk.Frame(self.canvas)

        # Add that frame to a window in the canvas
        self.canvas.create_window((0, 0), window=self.frame, anchor="nw")
        # Embed a scrollable frame inside a canvas for a better scrolling experi


        # Populate the frame with search results
        self.populate_results(results)

    def populate_results(self, results):
        for i, result in enumerate(results, 1):
            # Create a text widget for each result
            text_widget = tk.Text(self.frame, wrap=tk.WORD, width=120, height=12
            text_widget.pack(pady=10, padx=10, fill=tk.X)

            # Format and insert the result
            formatted_result = f"{i}. Authors: {result['Authors']}\n"
            formatted_result += f"   Title: {result['Title']}\n"
            formatted_result += f"   Publication: {result['Publication']}, \n"
            formatted_result += f"   Year: {result['Year']}\n"
            formatted_result += f"   DOI: {result['DOI']}\n"
            formatted_result += f"   URL: {result['URL']}\n"
            formatted_result += f"   Summary: {result['Reference_summary']}\n\n"

            text_widget.insert(tk.END, formatted_result)

            # Disable editing of the text widget
            text_widget.config(state=tk.DISABLED)

    def run(self):
        notify_completion()
        self.root.mainloop()
        print(f"Loaded final search results window at {datetime.now()}\n\n")
```

In [ ]:
```
# Unit testing - create and run the window
results_window = SearchResultsWindow(validated_references)
results_window.run()
```

Compile agent chain

In [ ]:
```
# LangChain pipeline combining all agents (from prompt simplification to validat

chain = RunnableLambda(prompt_preparation_agent) | RunnableLambda(search_agent)
# Combine all agents into a LangChain Runnable pipeline for end-to-end automatio
# need to wrap all agent functions into Runnables so they can be chained togethe
```

Compile GUI

In [ ]:
```
# Compile GUI

def SAGE_run():
    global username, user_prompt

    username, openai_api_key, user_prompt = get_user_input()
    global final_output
    final_output = chain.invoke({"user_prompt": user_prompt})

    notify_completion()

    results_window = SearchResultsWindow(final_output)
    results_window.run()
```

In [ ]:
```
# Unit testing (chain)
result = chain.invoke({"user_prompt": test_search_prompt})
notify_completion()
print(result)
```

```
Finished saving search results to database at 2025-04-11 17:43:31.971318
Attempting to create directory: c:\Users\knfc648\Documents\Personal\PgDip\portfol
io\portfolio_pgdip\module4\Individual project\SAGE results
Directory creation successful: c:\Users\knfc648\Documents\Personal\PgDip\portfoli
o\portfolio_pgdip\module4\Individual project\SAGE results
Attempting to save file: c:\Users\knfc648\Documents\Personal\PgDip\portfolio\port
folio_pgdip\module4\Individual project\SAGE results\SAGE_search_results_2025-04-1
1_17-43-31.csv
Finished saving search results to .csv file at 2025-04-11 17:43:31.972330
[{'Authors': 'Yun Wang, Feng Chen', 'Year': 2024, 'Title': 'Intelligent Analysis
and Optimization of Computer Aided Furniture Design by Deep Learning', 'Publicati
on': 'Computer-Aided Design and Applications', 'Issue': 'Not found', 'Pages': '17
8-190', 'DOI': '10.14733/cadaps.2025.s1.178-190', 'URL': HttpUrl('https://doi.or
g/10.14733/cadaps.2025.s1.178-190'), 'Reference_summary': 'This paper discusses t
he use of deep learning for intelligent analysis and optimization in computer-aid
ed furniture design, highlighting recent advancements and applications in the fie
ld.'}, {'Authors': 'Yukun Xia, Yingrui Ji, Yan Gan, Zijie Ding', 'Year': 2023, 'T
itle': 'Applying Ming furniture features to modern furniture design using deep le
arning', 'Publication': 'AHFE International', 'Issue': 'Not found', 'Pages': 'Not
found', 'DOI': '10.54941/ahfe1004197', 'URL': HttpUrl('https://doi.org/10.54941/a
hfe1004197'), 'Reference_summary': 'This study explores the application of deep l
earning to incorporate Ming furniture features into modern designs, using generat
ive adversarial networks to enhance design efficiency and aesthetic quality.'},
{'Authors': 'Xinhan Di, Pengqian Yu, Danfeng Yang, Hong Zhu, Changyu Sun, YinDong
Liu', 'Year': 2020, 'Title': 'Deep Layout of Custom-size Furniture through Multip
le-domain Learning', 'Publication': 'arXiv', 'Issue': 'Not found', 'Pages': 'Not
found', 'DOI': 'Not found', 'URL': HttpUrl('http://arxiv.org/abs/2012.08131v1'),
'Reference_summary': 'This paper presents a multiple-domain learning model for cr
eating custom-size furniture layouts, enhancing the auto-layout capabilities for
interior design using deep learning techniques.'}, {'Authors': 'Xinhan Di, Pengqi
an Yu', 'Year': 2021, 'Title': 'Deep Reinforcement Learning for Producing Furnitu
re Layout in Indoor Scenes', 'Publication': 'arXiv', 'Issue': 'Not found', 'Page
s': 'Not found', 'DOI': 'Not found', 'URL': HttpUrl('http://arxiv.org/abs/2101.07
462v1'), 'Reference_summary': 'This research applies deep reinforcement learning
to optimize furniture layout in indoor scenes, treating the task as a Markov deci
sion process to improve design quality and efficiency.'}, {'Authors': 'Özgür Asla
n, Burak Bolat, Batuhan Bal, Tuğba Tümer, Erol Şahin, Sinan Kalkan', 'Year': 202
2, 'Title': 'AssembleRL: Learning to Assemble Furniture from Their Point Clouds',
'Publication': 'arXiv', 'Issue': 'Not found', 'Pages': 'Not found', 'DOI': 'Not f
ound', 'URL': HttpUrl('http://arxiv.org/abs/2209.07268v1'), 'Reference_summary':
'This paper introduces a deep learning approach for furniture assembly using poin
t clouds, reducing the need for human supervision and enhancing the assembly proc
ess through novel reward signals.'}]
```

Run program

In [ ]:
```
SAGE_run()
```

```
Attempting to create directory: c:\Users\knfc648\Documents\Personal\PgDip\portfol
io\portfolio_pgdip\module4\Individual project\SAGE results
Directory creation successful: c:\Users\knfc648\Documents\Personal\PgDip\portfoli
o\portfolio_pgdip\module4\Individual project\SAGE results
Attempting to save file: c:\Users\knfc648\Documents\Personal\PgDip\portfolio\port
folio_pgdip\module4\Individual project\SAGE results\SAGE_search_results_2025-04-0
6_12-58-37.csv
```