Neurosymbolic Artificial Intelligence 1 (0) 1–4
IOS Press

A General Neural-symbolic Architecture for Knowledge-intensive Complex Reasoning

Shulin Cao [a,*], Zijun Yao [a], Lei Hou [a] and Juanzi Li [a,**]
[a] *Department of Computer Science and Technology, Tsinghua University, Beijing, China*
E-mails: caosl19@mails.tsinghua.edu.cn, yaozj20@mails.tsinghua.edu.cn, houlei@tsinghua.edu.cn, lijuanzi@tsinghua.edu.cn

Abstract. The symbolic and connectionist AI are two main routes in the last two generations of AI. Nowadays, researchers have gradually realized that these two routes do not conflict in principle, but can and should integrate and help each other. In this paper, we take knowledge-intensive complex reasoning task as an example to explore the integration of symbolic and connectionist AI. In particular, we design a general neural-symbolic architecture for this task, which is composed of four main components: multiple knowledge sources, knowledge manipulator, reasoning planner, and reasoning conductor. We also introduce existing techniques that can be used to implement each component, and summarize their advantages and potential directions for enhancement in building a neural-symbolic system for knowledge-intensive complex reasoning tasks.

Keywords: Knowledge-intensive Complex Reasoning, Neural-symbolic Framework, Large Language Models, Knowledge Bases

1. Introduction

Artificial intelligence (AI) aims at creating machine intelligence that can perceive, learn, reason, learn, and interact with their environment in a way that simulates human cognitive abilities. There are two main paradigms adopted in AI in order to achieve this: (1) the symbolic, and (2) the connectionist AI.

The first generation of AI is dominated by the symbolic approach, which believes that a physical symbol system has the necessary and sufficient means for general intelligent action [1]. It inherits the idea of mathematical logic that the physical world could be represented by symbolic systems, and views intelligence as a process of symbolic manipulation and reasoning following logical rules. The contemporary successor of symbolic AI is knowledge engineering, which takes knowledge accumulation, reuse, and knowledge-intensive complex reasoning as its core issues.

The second generation of AI is dominated by the connectionist approach. The ideological basis of connectionist AI is that the human brain is physically based on a network of interconnected neurons, and it is possible to create AI by computerizing neural networks that mimic those of living beings. Although the connectionist AI encountered many setbacks in the early stage, in recent years, on the basis of continuously improved deep learning models such as convolutional neural networks (CNN) [2], recurrent neural networks (RNN) [3] and Transformers [4], it has reached a new high level. Recently, large language models (LLMs) such as ChatGPT (GPT 3.5) and GPT-4 [5] have demonstrated astonishing abilities on a variety of domains and tasks, including abstraction, comprehension, vision, coding, mathematics, medicine, law, understanding of human motives and emotions, and more [6]. It is widely

* Shulin Cao and Zijun Yao contribute equally to this paper.
** Corresponding author. E-mail: lijuanzi@tsinghua.edu.cn.

recognized by academics and the industry that LLMs marks a new level of intelligence in AI systems, and can be regarded as the prototype of artificial general intelligence (AGI). LLMs are considered as a landmark success in the connectionist line of AI research.

Traditionally, the symbolic and connectionist approaches have been viewed as competing and mutually reinforcing. In recent years, there has been a widespread rethinking of the relationship between these two classical approaches. According to Yoshua Bengio, System 2 deep learning is essential to artificial general intelligence, which needs to study the slow-thinking "System 2" to realize the deep-level intelligence capabilities such as logical reasoning and planning [7], whose goal is in line with symbolic AI. Zhang Bo also points out that the third-generation AI method should be "double-wheel-driven by data and knowledge", so as to realize explainable, robust, safe and reliable AI systems [8]. Nowadays, researchers have gradually realized that these two routes do not conflict in principle, but can and should integrate and help each other.

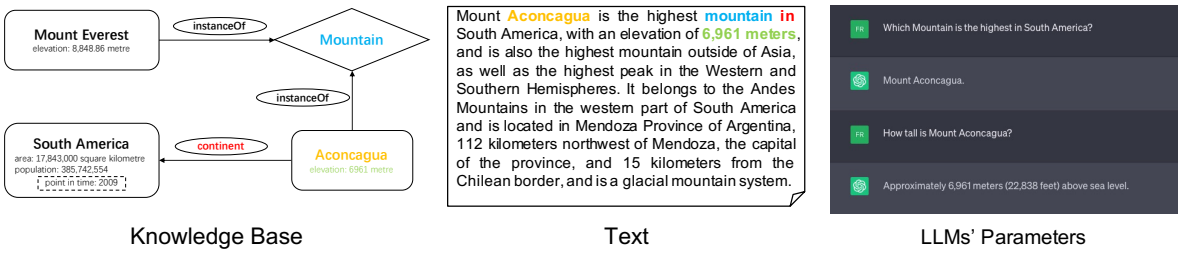

Fig. 1. Knowledge can be stored in different sources, such as structured knowledge bases, unstructured text, and LLMs' parameters. Color blue denotes concepts, orange denotes entities, green denotes the attributes of entities, and red denotes relations among entities and concepts.

In this paper, we will explore the integration of symbolic and connectionist AI using the knowledge-intensive complex reasoning task as an example. In knowledge-intensive complex reasoning, a complex question is given to the intelligent system, and a concise answer for the question is expected. It requires deep understanding of natural language text (including the question and knowledge corpora), and involves various reasoning capabilities such as multi-hop inference, attribute comparison, and set operation [9, 10]. Here, the knowledge can be stored in different sources, such as structured knowledge bases, unstructured text corpora, the parametric knowledge in LLMs' parameters, images, etc., as shown in Figure 1. Actually, as John McCarthy states, his 1976 memorandum [11] is of modern interest. In this article, he introduces a story from New York Times, writes 22 questions for it, and points out that an intelligent person or program should be able to answer the questions, which requires deep understanding and reasoning ability. He also gives guidelines for knowledge-intensive complex reasoning. There are two basic elements: 1) A formalism capable of expressing the assertions of the sentences free from dependence on the grammar of the English language, which can be called artificial natural language—ANL; 2) A data structure for expressing the facts (apart from expressing the sentences). Certain programs can be applied to the data structures.

Inspired by the proposal of John McCarthy, we design our general neural-symbolic architecture for knowledge-intensive complex reasoning, which will be introduced in Section 2. Specifically, the framework is composed of four main construction blocks—knowledge-oriented data manager, knowledge manipulator, reasoning planner, and reasoning conductor. These modules draw advantages from both neural and symbolic methodologies, and take the responsibility of knowledge-intensive complex reasoning in different levels. The overall architecture also integrates human in the loop, to progressively accomplish the targeted tasks and optimize the overall system. Next, we will present the detailed implementations of the four modules in Section 3, by introducing the related works. Finally, we will look into the future and conclude the paper in Section 4.

2. Overall Architecture Design

With the rapid development of large-scale pre-trained models in many different domains [5, 12], neural models show great promise in understanding complex semantics and generalizing to unseen data and tasks. However, they

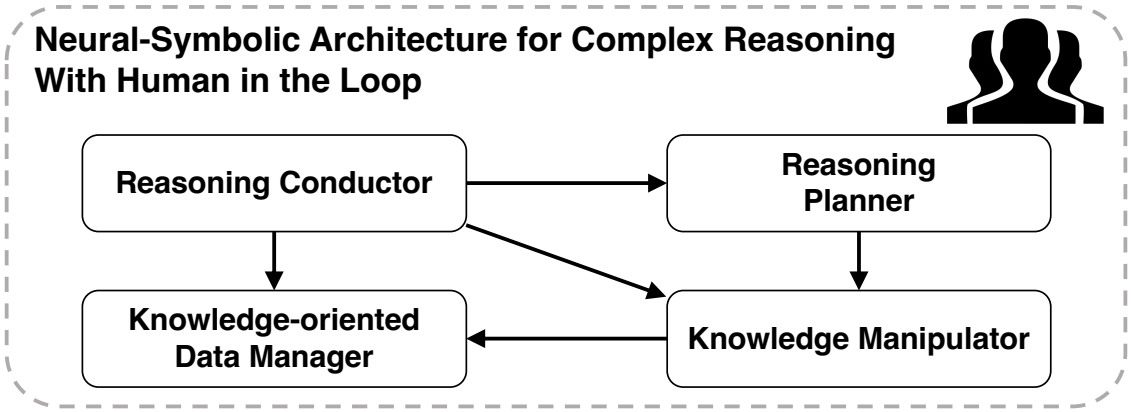

Fig. 2. The proposed neural-symbolic framework to solve knowledge-intensive complex reasoning task. We use arrows to indicate calling and access relationship between components. The basis of the whole architecture is the knowledge-oriented data manager. Knowledge manipulator performs knowledge operations on the knowledge data. Reasoning planner produces candidate reasoning path by decomposing complex reasoning tasks into assembled knowledge manipulation primitives. Reasoning conductor decides the optimal reasoning path, select knowledge manipulation implementations, and execute on the knowledge manager to obtain the answer. Our architecture also incorporates human in the loop to update knowledge sources, fix and optimize reasoning plans, and converse with the system to perform multiple-step reasoning tasks.

fail to reliably reason with pre-defined schemata and generate appeared to be correct but actually wrong answers, which is recognized as notorious hallucination issue. Meanwhile, symbols are able to express explicit constraints and knowledge guidance. Although they are less superior than neural methods in generalization, they tend to be more reliable in solving strict knowledge-intensive complex reasoning tasks. Thus, combining the advantages from both the neural and the symbolic perspectives is a promising direction to realize intelligence. To achieve this, we mimic the dual-process of human mind [13]. Neural models, as represented by large language models (LLMs), convert rough natural language intentions into well-defined symbols [11, 14]. They resemble system 1 to perform associative thinking. The reasoning is performed with the aid of symbolic methods, which resemble system 2 to perform logical thinking.

To this end, we design an architecture, as a initiative neural-symbolic realization to solve knowledge-intensive complex reasoning tasks, which is demonstrated in Figure 2. The architecture is composed of four main construction blocks—knowledge-oriented data manager, knowledge manipulator, reasoning planner, and reasoning conductor. These components carry distinct responsibilities, drawing advantages from both neural and symbolic methodologies. Moreover, the proposed framework integrates human involvement in the loop, to progressively accomplish the targeted tasks and optimize the overall system. In the following of this section, we specify the distinct responsibilities of each module and discuss how human contribute to and enhance the overall framework in the loop.

2.1. Knowledge-oriented Data Manager

Knowledge-oriented data manager is the foundation of the overall system. It aggregates knowledge from multiple sources, including not only traditional knowledge bases [15, 16], but also information scattered in raw formats such as plain text [17, 18], model parameters [19], images [20], and tables [21].

To facilitate unified knowledge manipulation, the manager provide knowledge information in a coherent symbolic representation. A typical semantic representation system comprises five core knowledge elements: (1) `Entities` are unique object in the real world; (2) `Concepts` are sets of entities with similar properties; (3) `Relations` describe the connection among entities and concepts; (4) `Attributes` are associated with entities to describe them from certain aspects; (5) `Qualifiers`[1] are constraints on other knowledge items to specify under which conditions that they are true. However, there are variant definitions for basic knowledge elements. For example,

[1] Also known as `Hyper Relations` or `CVT` in different literature [15, 22].

visual knowledge symbols typically defines `Objects`, which shares a similar definition with `Entities`. Thus, the data manager need to provide a compiler to translate different knowledge elements into unified symbols.

Knowledge-oriented data manager brings benefit in twofold. First, the knowledge-oriented data management enables precise knowledge manipulation, in contrary to scattering knowledge in raw data. For example, plain text data only allows for fuzzy similarity matching with query utterances, whereas the knowledge-oriented data manager provides a much more accurate method to access information by posing constraints on the knowledge such as retrieving entities when constraining their concept. This capability has the potential to reduce hallucination when generating answers conditioned on the given data. Second, the knowledge-oriented data manager gathers knowledge complementary with each other from different sources, which broadens the knowledge coverage.

2.2. Knowledge Manipulator

Knowledge manipulator provides well-defined knowledge manipulation primitives on the knowledge sources. These primitives have strictly defined behavior, which describes the intended outcome of each individual primitive, without stipulating specific their implementation algorithms and techniques. As a result, each knowledge manipulation primitive can be realized via neural models, symbolic methods, or a combination of both. They can be assembled into programs, the execution of which produce the final result for a complex reasoning task.

Specifically, knowledge manipulation primitives include: (1) *Direct knowledge access*, which retrieves elements from the data structure according to specific requirements, *e.g.,* linking mentions to the entities they refer to, a process also known as entity disambiguation; (2) *Knowledge processing*, which operates on the outputs of other knowledge operations, *e.g.,* filtering out knowledge elements that fulfill certain requirements; (3) *Knowledge updating*, which offers the capability to refresh and refine outdated information in the data structure.

The precise knowledge manipulation brings two key advantages. Firstly, these unified primitives act as an interface to incorporate various implementations with complementary capacities for the same objective. This allows for the selection of the optimal implementation to construct the final manipulating operation (how to choose optimal implementation is undertaken by the reasoning conductor in Section 2.4). Secondly, by atomizing basic operations, the overall system imparts explicit meanings to each procedure and preserves the corresponding intermediate results. Consequently, human users can inspect not only the final answer, but also the output of each individual operation. This transparency of execution thereby enhances interpretability.

2.3. Reasoning Planner

Reasoning planner decomposes knowledge-intensive complex reasoning tasks into assembled knowledge primitives. Acting as an interface, it comprehends human requirements and represents them as reasoning processes, *i.e.*, the composition of basic operations, and the execution result of reasoning process is the answer for the knowledge-intensive complex reasoning task.

Specifically, it takes the language understanding ability of neural models to decompose a complex question/instruction into basic symbolic units, *i.e.*, an ordered list of steps. In our framework, the format of steps is not constrained, and can be expressed through either natural language or symbolic operations. The reasoning process output by planner is therefore a multi-step program, a chain-of-thought reasoning process, or a sequence of simple questions. For example, for the question "Which is the highest peak in Asia or South America?", a possible ordered list of steps outputted by the planner is: find Asia, relate it to the its contained mountains, then do the same for South America, and compute the union of the mountains from Asia and South America, and finally pick the highest one from them.

The planning module has multiple different responsibilities, among which the most important is to refine ambiguous human intention into well-defined knowledge manipulations. In this process, it allows humans to express their intention in a human-friendly way—natural language. Most recently, large scale pre-trained language models show potential to make plans according to natural language prompts, due to its remarkable language understanding and generation abilities, which sheds light on the feasibility to achieve this goal. Secondly, the planning module extracts higher order intentions from natural language descriptions. For example, answering the question "Did Aristotle Use a Laptop?" is an implicit reasoning task that can be planned with our reasoning planner.

2.4. Reasoning Conductor

Reasoning conductor makes decisions based on the candidate reasoning processes given by the reasoning planner, and executes the knowledge manipulator on the multiple-sourced knowledge to obtain the final answer. Acting as a decision maker, it has the ability to schedule the planner and manipulator flexibly, trying to search an optimal solution for the whole task.

Specifically, for a natural language question/requirement that describes the complex reasoning task, there may exist multiple ways to plan the reasoning process, and each primitive in the reasoning process can incorporate various implementations. For example, if the user poses a complex question "Which is the highest peak in Asia or South America?", there are multiple ways to answer the question: 1) find the highest peak in Asia and the highest peak in South America respectively, and then compare their height; 2) collect all the mountains in Asia and South America, and find the highest one. To find the highest peak in Asia, there are also multiple implementations: 1) gather the information from knowledge bases; 2) find a clue from text corpora. Here, accessing knowledge from text "Mount Everest is the highest mountain in Asia and the highest mountain in the world" is an accurate and may be the fastest way for the original question. And the reasoning conductor takes the responsibility to find such optimal solution.

The reasoning conductor brings two advantages for our framework. First, it can select an optimal reasoning process from our planning module, and decide the optimal implementation strategies for the knowledge manipulator by systematically considering multiple factors comprehensively, including the computational cost, the precision, and recall. Second, it allows users to customize their own priority for decision making, by injecting prior knowledge into the reasoning process, *e.g.*, choosing knowledge base as the primary knowledge sources.

2.5. The Overall Architecture with Human in the Loop

Human possess prior-knowledge and evolving posterior knowledge about the world. In the meanwhile, the architecture still surfers out of distribution tasks. To better serve human needs, the proposed architecture also allows human to improve its world understanding and performance with feedbacks. We highlight three most important life cycles that involve both human users and system modules.

The *knowledge updating cycle* allows human users to update the knowledge in storage. This process includes allowing user to inspect knowledge when the answer is less satisfactory and annotate newly involved knowledge. In the *program fixing cycle*, human users are allowed to find error programs given by the reasoning planner and help to select the optimal reasoning path if the solver selects the sub-optimal solution. Humans not only help the system solve current complex reasoning task instantly, but also accumulate training data, which is further used to train better task planner and problem solver. The *human-aided problem solving cycle* allows human user to decompose complex reasoning tasks into simpler ones on the fly in a dialogue. The system coordinate with the human user to solve simple reasoning tasks and assemble their output into the final result.

3. Related Research and Technique towards the Architecture

We introduce existing techniques that can be used to implement the proposed architecture in Section 2. We also summarize their advantages and potential directions for enhancement in building a neural-symbolic system to solve complex reasoning tasks.

3.1. Knowledge-oriented Data Manager

We introduce techniques that are used to manage and represent knowledge from different sources, including knowledge base, textual corpora, large language models, and other knowledge sources.

Knowledge Base. Knowledge base[2] stores crowd-sourced knowledge in the format (h, r, t) [15, 16, 23]. h and t are head node and tail node, while r is the relation between the head node and the tail node. They are further classified as fact knowledge, hierarchy knowledge, attribute knowledge, and qualifier knowledge. Specifically, for fact knowledge, h and t are entities and r is a relation. For hierarchy knowledge, t is a concept, and h is either an entity that falls into the category of h or a subconcept of h, where r denotes `isInstanceOf` and `subClassOf`, respectively. For attribute knowledge, h is an entity, r is the attribute name, and t is the attribute value. Qualifier knowledge recursively specifies the condition under which other knowledge is true, where h is a triple of the constrained knowledge, r is an attribute name, and t is the attribute value.

Textual Corpora. Textual corpora are rich in unstructured knowledge. They can be obtained via search engines [24] and aligned with the unified knowledge schemata through a series of information extraction algorithms, such as named entity recognition, entity linking [25], relation extraction [26, 27], and entity typing [28].

In managing textual knowledge, markup languages plays a key role as well-defined symbolic system. GraphQ IR uses sentinel tokens to mark special knowledge elements, *e.g.,* `<C></C>` for concept. Although GraphQ IR [29] is originally proposed to formally represent natural language questions, it shows potential to record textual knowledge in textual data. Other techniques include resource description framework (RDF) and web ontology language (OWL).

Model Parameters. Large scale pre-trained language models (LLMs) [5, 6] are rich in parametric knowledge. Their knowledge is accessed via knowledge probing [19, 30, 31]. These method constructs template prompts to require LLMs continue to output the desired knowledge following specified knowledge schemata.

Other Knowledge Sources. There are also other data structures: scene graph [32] encodes object instances, attributes of objects, and relationships between objects to denote the semantics of an image; tables can be transformed to knowledge graphs with heuristic rules [33].

3.2. Knowledge Manipulator

The implementation of knowledge manipulation primitives is twofold: how to design the atomic knowledge operations and how are the atomic knowledge operations executed.

Knowledge Manipulation Operation Design. Existing design for knowledge manipulations are classified as three main categories. (1) *λ-calculus Based.* λ-calculus is a general computational model, which is proved to be Turing complete [34]. λ-calculus defines atomic operations as λ-functions to represent natural language questions [35]. To facilitate operating on knowledge bases, they evolve into λ-DCS [36–39]. (2) *Database Query Based.* Knowledge bases are usually stored in compatible with database, such as Virtuoso and Neo4j. Thus, it is direct to use graph database query primitives as knowledge manipulation primitives. They include SQL [40] for relational database, SPARQL [41] for graph database, and Cypher for attributed graph database. (3) *Knowledge-oriented Operations.* Most recently, researchers propose knowledge-oriented programming language (KoPL) [10], which designs operates on knowledge elements, such as `Relate` stands for "find entities that have certain relation with the given entity".

Knowledge Manipulation Operation Realization. The realization varies according to the knowledge sources. Here we take `Relate` operation as the representative example to show potential realization strategies.

Knowledge Graph Based. The most direct way to realize knowledge manipulation on knowledge base is subgraph matching. For example, if the triple (Mark Twain, writing language, English) is stored in the knowledge base, then we can directly find the entity that have the relation "writing language" with Mark Twain. To incorporate neural method, we also realize knowledge manipulations in a semantic space using knowledge embedding [42]. There are also attempt to model reasoning on knowledge graph as multi-hop random walk agent [43].

Textual Corpora Based. Another method is to employ the knowledge in text corpora with open question answering, *e.g.,* answering the question "What is the written language of Mark Twain?". Typically, open question answering adopts a two-stage paradigm, *i.e.,* first retrieves relevant documents from the large-scale text corpora, and then performs reading comprehension over the documents to obtain the final answer.

[2] Also known as knowledge graph.

Language Model Based. LLMs are capable of answering questions with the rich knowledge stored in models' parameters. And the question "What is Mark Twain's writing language?" can be answered by prompting the LLM to perform open-domain question answering.

3.3. Reasoning Planner

The planner requires the language understanding ability to translate natural language questions or instructions into explicit reasoning processes. As we introduced in Section 2.3, the translated reasoning process can take the forms of programs, chain-of-thought reasoning steps, or natural language sub-questions.

Here we would like to introduce the reasoning processes in details. (1) For *programs* composed of symbolic knowledge manipulators, Rational Meaning Construction [44] adopts Church programs to offer a structured representation for expressing novel situations and arbitrary problems with respect to a meaningful model over possible world states. W3C officially recommends SPARQL [45] for manipulating and querying RDF stores since 2008 [46]. KoPL [10] is a newly emerged knowledge-oriented programming language designed for querying knowledge base by executing a multi-step program that takes knowledge elements as the arguments. Other programs include Cypher [47], λ-DCS[36], λ-calculus [48], etc. These languages have their own advantages and drawbacks, and can be transpiled to each other with the help of intermediate languages such as GraphQ IR [29]. (2) For *chain-of-thought (CoT) reasoning*, they are composed of step-by-step natural language reasoning steps [49]. With CoT, the ability of LLMs to perform complex reasoning is improved significantly. (3) For *natural language sub-questions*, a representative is Question Decomposition Meaning Representation (QDMR) in the BREAK dataset [50], which constitutes the ordered list of steps, expressed through natural language, that are necessary for answering a question.

For the planner model, we can take planning as a Seq2Seq translation task and employ an encoder-decoder framework for the model. Such methods usually rely on a large amount of labeled data, *i.e.*, question and reasoning process pairs [10], which are often lacking because such annotation is both expensive and labor-intensive. Another paradigm is to learn from question-answer pairs by taking the answers as weak supervision and searching for gold reasoning processes with reinforcement learning (RL) [51, 52], which is challenging due to combinatorial explosion in searching space along with extremely sparse rewards. Recently, with the remarkable generalizability and language understanding ability of LLMs, a new path is open by inferring the reasoning process with in-context learning [53]. For example, Program-aided Language models (PAL) uses the LLM to read natural language problems and generate programs as the intermediate reasoning steps, and offloads the solution step to a runtime such as a Python interpreter [54].

3.4. Reasoning Conductor

The reasoning conductor aims to obtain an optimal solution for the knowledge-intensive complex reasoning task. It needs the flexibility to plan dynamically and consider multiple factors to make a global choice, including whether the chosen knowledge resource is suitable, whether the planning route is the most efficient, and how much computational resource it costs. One possible way to dynamic reasoning conductor is probabilistic programming, e.g., Rational Meaning Construction [44] employs an inference function that computes probabilities over the space of possible worlds consistent with and conditioned on information in the Church program to infer the optimal answer. Another direction is to optimize on tree-like computation graphs. Here we provide some examples.

For the task of answering complex logical queries on incomplete knowledge graphs, Query Computation Tree Optimization (QTO) [55] finds the optimal solution by a forward-backward propagation on a tree-like computation graph, *i.e.*, query computation tree. In particular, QTO utilizes the independence encoded in the query computation tree to reduce the search space, where only local computations are involved during the optimization procedure, *i.e.*, they find a set of entity assignments that maximizes the truth score of a first-order logical query based on the truth value of each one-hop atom provided by a knowledge embedding based link predictor.

For the task of answering complex natural language questions, RoHT [56] first builds the Hierarchical Question Decomposition Tree (HQDT) to understand the semantics of a complex question, and then conducts probabilistic reasoning over HQDT from root to leaves recursively, to aggregate heterogeneous knowledge at different tree levels and search for a best solution considering the decomposing and answering probabilities.

For the task of other complex reasoning tasks that needs search or planning such as Game of 24, ToT [57] utilizes LLMs to perform deliberate decision making. This involves considering multiple reasoning paths, evaluating options internally, and determining the most suitable next steps, as well as looking ahead or backtracking when necessary to make global choices.

3.5. Human in the loop

As stated in Section 2.5, it is of great benefit to involving human into the reasoning loop, spanning all four components of the overall architecture.

For the *knowledge-oriented data manager*, it takes the responsibility of providing up-to-date, accurate and comprehensive knowledge, in which humans can play an important role by sharing and editing knowledge. For example, Wikidata employs crowd-sourcing to extend and edit the stored information, even without creating an account, and the form-based interface makes editing easy [58]. The Wikipedia is also committed to "a world in which every single human being can freely share in the sum of all knowledge".

For the *knowledge manipulator*, it takes the responsibility of accessing, processing, and updating knowledge with atomic actions. And humans can design, update and expand the set of atomic actions with their prior knowledge on the targeted complex reasoning task. For example, for the complex visual tasks such as factual knowledge object tagging and language guided image editing, actions such as image resizing, cropping, filtering, and colorspace conversions are designed by experts [59].

For the *reasoning planner*, it is possible for the planner to improve its performance with the human feedback. For example, given an interaction interface, the system explains the predicted program step by step in natural language along with intermediate answers to the user, and then, the user can give feedback for the inspected wrong step, thus the current planner can accordingly be updated to improve accuracy. In another case, when the planner outputs an explainable program with transparent reasoning process, e.g., KoPL, the user can even edit with simple graphical operators, such as "dragging" to add knowledge operators and "slot filling" to designate operator arguments [60].

For the *reasoning conductor*, it is also able to improve its performance with human in the loop, if humans have prior knowledge for problem solving, *i.e.*, humans can select the most optimal solution for the conductor, and such data can be collected as preference data to train the conductor with reinforcement learning from human feedback (RLHF) [61]. For example, for WebGPT [24], guidance from humans is central. They collected demonstrations and comparisons from humans to train neural models to search and navigate the web. Specifically, they collected examples of humans using the browser to answer questions (demonstrations) to teach neural models to use their text-based browser. And further, they collected pairs of model-generated answers to the same question and asked humans which one they preferred (comparisons) to improve the optimize answer quality.

4. Conclusion and Future Directions

Inspired by the promising capability to incorporate both neural and symbolic models [11, 13], in this paper, we design an initiative architecture to solve knowledge-intensive complex reasoning tasks. The architecture not only utilizes the capability to understand rough descriptions and rich knowledge of LLMs, but also enables symbol system to constraint the behavior of LLMs to reduce hallucination. Although there are many pioneer techniques that indicates the feasibility of our proposed architecture, as we suggest in Section 3, it still requires efforts to build the overall architecture. We believe that future works mainly fall primarily into two aspects: the macroscopic and the microscopic.

At the macroscopic level, the challenge lies in selecting and integrating specific techniques to implement the four key components. This is non-trivial, as we are essentially designing standardised behaviour for different algorithms to coordinate to achieve the same goal. For example, the problem solver may accept plans given by both LLMs and traditional syntactic-based semantic parser, which poses a requirement on the consistency between the output of both sides. This entails not just cherry-picking the best performers within each module but also ensuring that they can effectively communicate with each other and function as a cohesive system.

At the microscopic level, we should not stop the pace to design better implementations for each component and atomic operations for the knowledge manipulation primitives. We propose two main research directions for future work. First, we should explore methods to constrain LLMs to perform well-defined symbolic reasonings. Existing research includes prompt template design and constraint decoding. However, there is still no guarantee to obtain behaviour following strict schemata. Second, it is also crucial to design coherent and user-friendly interfaces for each component to incorporate human in the loop. The interface should provide information as close to natural language as possible to allow human users to inspect the working status of the system. It also needs to provide guidance and convert human user actions into consistent behaviour of the intervened component in the system. In addition, we need to explore more interactive systems, such as in the form of dialogue, to carry out the functionality of the system.

In conclusion, we specify the necessity, responsibility, and potential implementation techniques for each component to achieve a general architecture for knowledge-intensive complex reasoning tasks. Although there are still insufficient consideration with regard to the whole design and implementation details, it is intriguing to realize the whole system to achieve its potential capacity in the near future.

References

[1] A. Newell, Physical Symbol Systems, *Cogn. Sci.* **4** (1980), 135–183.

[2] R. Salakhutdinov, Deep learning, in: *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, S.A. Macskassy, C. Perlich, J. Leskovec, W. Wang and R. Ghani, eds, ACM, 2014, p. 1973. doi:10.1145/2623330.2630809.

[3] D.E. Rumelhart, G.E. Hinton and R.J. Williams, Learning internal representations by error propagation, 1986.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser and I. Polosukhin, Attention is All you Need, in: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, I. Guyon, U. von Luxburg, S. Bengio, H.M. Wallach, R. Fergus, S.V.N. Vishwanathan and R. Garnett, eds, 2017, pp. 5998–6008. https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

[5] OpenAI, GPT-4 Technical Report, *ArXiv preprint* **abs/2303.08774** (2023). https://arxiv.org/abs/2303.08774.

[6] S. Bubeck, V. Chandrasekaran, R. Eldan, J.A. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y.T. Lee, Y.-F. Li, S.M. Lundberg, H. Nori, H. Palangi, M.T. Ribeiro and Y. Zhang, Sparks of Artificial General Intelligence: Early experiments with GPT-4, *ArXiv preprint* **abs/2303.12712** (2023). https://arxiv.org/abs/2303.12712.

[7] A. Goyal and Y. Bengio, Inductive biases for deep learning of higher-level cognition, *Proceedings of the Royal Society A* **478** (2020).

[8] B. Zhang, Knowledge and Artificial Intelligence, *Sino-German Symposium on "Integrating Symbolic Representation with Numeric Representation for Commonsense Reasoning"* (2019).

[9] Z. Yang, P. Qi, S. Zhang, Y. Bengio, W. Cohen, R. Salakhutdinov and C.D. Manning, HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering, in: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 2369–2380. doi:10.18653/v1/D18-1259. https://aclanthology.org/D18-1259.

[10] S. Cao, J. Shi, L. Pan, L. Nie, Y. Xiang, L. Hou, J. Li, B. He and H. Zhang, KQA Pro: A Dataset with Explicit Compositional Programs for Complex Question Answering over Knowledge Base, in: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, Dublin, Ireland, 2022, pp. 6101–6119. doi:10.18653/v1/2022.acl-long.422. https://aclanthology.org/2022.acl-long.422.

[11] J. McCarthy, An example for natural language understanding and the AI problems it raises, *Formalizing Common Sense: Papers by John McCarthy* **355** (1990).

[12] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit and N. Houlsby, An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, in: *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, OpenReview.net, 2021. https://openreview.net/forum?id=YicbFdNTTy.

[13] Y. Bengio et al., From system 1 deep learning to system 2 deep learning, in: *Neural Information Processing Systems*, 2019.

[14] S. Wolfram, Chatgpt gets its' wolfram superpowers', *Recuperado de https://writings. stephenwolfram. com/2023/03/chatgpt-gets-its-wolfram-superpowers* (2023).

[15] K.D. Bollacker, C. Evans, P.K. Paritosh, T. Sturge and J. Taylor, Freebase: a collaboratively created graph database for structuring human knowledge, in: *SIGMOD Conference*, 2008.

[16] D. Vrandeic and M. Krotzsch, Wikidata: a free collaborative knowledgebase, *Commun. ACM* **57** (2014), 78–85.

[17] P.S.H. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, S. Riedel and D. Kiela, Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks, in: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan and H. Lin, eds, 2020. https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

[18] Y. Mao, P. He, X. Liu, Y. Shen, J. Gao, J. Han and W. Chen, Generation-Augmented Retrieval for Open-Domain Question Answering, in: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Association for Computational Linguistics, Online, 2021, pp. 4089–4100. doi:10.18653/v1/2021.acl-long.316. https://aclanthology.org/2021.acl-long.316.

[19] F. Petroni, T. Rocktäschel, S. Riedel, P. Lewis, A. Bakhtin, Y. Wu and A. Miller, Language Models as Knowledge Bases?, in: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Association for Computational Linguistics, Hong Kong, China, 2019, pp. 2463–2473. doi:10.18653/v1/D19-1250. https://aclanthology.org/D19-1250.

[20] K. Tang, Y. Niu, J. Huang, J. Shi and H. Zhang, Unbiased Scene Graph Generation From Biased Training, in: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, IEEE, 2020, pp. 3713–3722. doi:10.1109/CVPR42600.2020.00377.

[21] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun and W. Zhang, Knowledge vault: a web-scale approach to probabilistic knowledge fusion, in: *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, S.A. Macskassy, C. Perlich, J. Leskovec, W. Wang and R. Ghani, eds, ACM, 2014, pp. 601–610. doi:10.1145/2623330.2623623.

[22] M. Galkin, P. Trivedi, G. Maheshwari, R. Usbeck and J. Lehmann, Message Passing for Hyper-Relational Knowledge Graphs, in: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Online, 2020, pp. 7346–7359. doi:10.18653/v1/2020.emnlp-main.596. https://aclanthology.org/2020.emnlp-main.596.

[23] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P.N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer and C. Bizer, DBpedia - A large-scale, multilingual knowledge base extracted from Wikipedia, *Semantic Web* **6** (2015), 167–195.

[24] R. Nakano, J. Hilton, S.A. Balaji, J. Wu, L. Ouyang, C. Kim, C. Hesse, S. Jain, V. Kosaraju, W. Saunders, X. Jiang, K. Cobbe, T. Eloundou, G. Krueger, K. Button, M. Knight, B. Chess and J. Schulman, WebGPT: Browser-assisted question-answering with human feedback, *ArXiv preprint* **abs/2112.09332** (2021). https://arxiv.org/abs/2112.09332.

[25] N.D. Cao, G. Izacard, S. Riedel and F. Petroni, Autoregressive Entity Retrieval, in: *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, OpenReview.net, 2021. https://openreview.net/forum?id=5k8F6UU39V.

[26] X. Han, H. Zhu, P. Yu, Z. Wang, Y. Yao, Z. Liu and M. Sun, FewRel: A Large-Scale Supervised Few-Shot Relation Classification Dataset with State-of-the-Art Evaluation, in: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 4803–4809. doi:10.18653/v1/D18-1514. https://aclanthology.org/D18-1514.

[27] M. Banko, M.J. Cafarella, S. Soderland, M. Broadhead and O. Etzioni, Open Information Extraction from the Web, in: *CACM*, 2007.

[28] H. Jin, L. Hou, J. Li and T. Dong, Fine-Grained Entity Typing via Hierarchical Multi Graph Convolutional Networks, in: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Association for Computational Linguistics, Hong Kong, China, 2019, pp. 4969–4978. doi:10.18653/v1/D19-1502. https://aclanthology.org/D19-1502.

[29] L. Nie, S. Cao, J. Shi, J. Sun, Q. Tian, L. Hou, J. Li and J. Zhai, GraphQ IR: Unifying the Semantic Parsing of Graph Query Languages with One Intermediate Representation, in: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 2022, pp. 5848–5865. https://aclanthology.org/2022.emnlp-main.394.

[30] Z. Zhong, D. Friedman and D. Chen, Factual Probing Is [MASK]: Learning vs. Learning to Recall, in: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics, Online, 2021, pp. 5017–5033. doi:10.18653/v1/2021.naacl-main.398. https://aclanthology.org/2021.naacl-main.398.

[31] H. Peng, X. Wang, S. Hu, H. Jin, L. Hou, J. Li, Z. Liu and Q. Liu, COPEN: Probing Conceptual Knowledge in Pre-trained Language Models, in: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 2022, pp. 5015–5035. https://aclanthology.org/2022.emnlp-main.335.

[32] J. Johnson, R. Krishna, M. Stark, L. Li, D.A. Shamma, M.S. Bernstein and F. Li, Image retrieval using scene graphs, in: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, IEEE Computer Society, 2015, pp. 3668–3678. doi:10.1109/CVPR.2015.7298990.

[33] I. Saparina and A. Osokin, SPARQLing Database Queries from Intermediate Question Decompositions, in: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 2021, pp. 8984–8998. doi:10.18653/v1/2021.emnlp-main.708. https://aclanthology.org/2021.emnlp-main.708.

[34] A.M. Turing, Computability and λ-definability, *The Journal of Symbolic Logic* **2**(4) (1937), 153–163.

[35] Y. Artzi, N. FitzGerald and L. Zettlemoyer, Semantic Parsing with Combinatory Categorial Grammars, in: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Tutorials)*, Association for Computational Linguistics, Sofia, Bulgaria, 2013, p. 2. https://aclanthology.org/P13-5002.

[36] P. Liang, Lambda Dependency-Based Compositional Semantics, *CoRR* **abs/1309.4408** (2013). http://arxiv.org/abs/1309.4408.

[37] P. Pasupat and P. Liang, Compositional Semantic Parsing on Semi-Structured Tables, in: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Association for Computational Linguistics, Beijing, China, 2015, pp. 1470–1480. doi:10.3115/v1/P15-1142. https://aclanthology.org/P15-1142.

[38] Y. Wang, J. Berant and P. Liang, Building a Semantic Parser Overnight, in: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Association for Computational Linguistics, Beijing, China, 2015, pp. 1332–1342. doi:10.3115/v1/P15-1129. https://aclanthology.org/P15-1129.

[39] P. Pasupat and P. Liang, Inferring Logical Forms From Denotations, in: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, Berlin, Germany, 2016, pp. 23–32. doi:10.18653/v1/P16-1003. https://aclanthology.org/P16-1003.

[40] V. Zhong, C. Xiong and R. Socher, Seq2sql: Generating structured queries from natural language using reinforcement learning, *ArXiv preprint* **abs/1709.00103** (2017). https://arxiv.org/abs/1709.00103.

[41] Y. Sun, L. Zhang, G. Cheng and Y. Qu, SPARQA: Skeleton-Based Semantic Parsing for Complex Questions over Knowledge Bases, in: *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, AAAI Press, 2020, pp. 8952–8959. https://aaai.org/ojs/index.php/AAAI/article/view/6426.

[42] A. Bordes, N. Usunier, A. García-Durán, J. Weston and O. Yakhnenko, Translating Embeddings for Modeling Multi-relational Data, in: *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, C.J.C. Burges, L. Bottou, Z. Ghahramani and K.Q. Weinberger, eds, 2013, pp. 2787–2795. https://proceedings.neurips.cc/paper/2013/hash/1cecc7a77928ca8133fa24680a88d2f9-Abstract.html.

[43] X.V. Lin, R. Socher and C. Xiong, Multi-Hop Knowledge Graph Reasoning with Reward Shaping, in: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 3243–3253. doi:10.18653/v1/D18-1362. https://aclanthology.org/D18-1362.

[44] L.S. Wong, G. Grand, A.K. Lew, N.D. Goodman, V.K. Mansinghka, J. Andreas and J.B. Tenenbaum, From Word Models to World Models: Translating from Natural Language to the Probabilistic Language of Thought, *ArXiv preprint* **abs/2306.12672** (2023). https://arxiv.org/abs/2306.12672.

[45] W.W.W. Consortium, SPARQL 1.1 overview (2013).

[46] E. Prud'hommeaux and A. Seaborne, SPARQL Query Language for RDF, 2008. http://www.w3.org/TR/rdf-sparql-query/.

[47] N. Francis, A. Green, P. Guagliardo, L. Libkin, T. Lindaaker, V. Marsault, S. Plantikow, M. Rydberg, P. Selmer and A. Taylor, Cypher: An Evolving Query Language for Property Graphs, in: *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*, G. Das, C.M. Jermaine and P.A. Bernstein, eds, ACM, 2018, pp. 1433–1445. doi:10.1145/3183713.3190657.

[48] H.P. Barendregt et al., *The lambda calculus*, Vol. 3, North-Holland Amsterdam, 1984.

[49] H. Trivedi, N. Balasubramanian, T. Khot and A. Sabharwal, Interleaving Retrieval with Chain-of-Thought Reasoning for Knowledge-Intensive Multi-Step Questions, in: *Annual Meeting of the Association for Computational Linguistics*, 2022. https://api.semanticscholar.org/CorpusID:254877499.

[50] T. Wolfson, M. Geva, A. Gupta, M. Gardner, Y. Goldberg, D. Deutch and J. Berant, Break It Down: A Question Understanding Benchmark, *Transactions of the Association for Computational Linguistics* **8** (2020), 183–198.

[51] S. Cao, J. Shi, Z. Yao, X. Lv, J. Yu, L. Hou, J. Li, Z. Liu and J. Xiao, Program Transfer for Answering Complex Questions over Knowledge Bases, in: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, Dublin, Ireland, 2022, pp. 8128–8140. doi:10.18653/v1/2022.acl-long.559. https://aclanthology.org/2022.acl-long.559.

[52] C. Liang, J. Berant, Q. Le, K.D. Forbus and N. Lao, Neural Symbolic Machines: Learning Semantic Parsers on Freebase with Weak Supervision, in: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, Vancouver, Canada, 2017, pp. 23–33. doi:10.18653/v1/P17-1003. https://aclanthology.org/P17-1003.

[53] L. Tianle, X. Ma, A. Zhuang, Y. Gu, Y. Su and W. Chen, Few-shot In-context Learning on Knowledge Base Question Answering, *ArXiv preprint* **abs/2305.01750** (2023). https://arxiv.org/abs/2305.01750.

[54] L. Gao, A. Madaan, S. Zhou, U. Alon, P. Liu, Y. Yang, J. Callan and G. Neubig, PAL: Program-aided Language Models, *ArXiv preprint* **abs/2211.10435** (2022). https://arxiv.org/abs/2211.10435.

[55] Y. Bai, X. Lv, J. Li and L. Hou, Answering Complex Logical Queries on Knowledge Graphs via Query Computation Tree Optimization, *ArXiv preprint* **abs/2212.09567** (2022). https://arxiv.org/abs/2212.09567.

[56] J. Zhang, S. Cao, T. Zhang, X. Lv, J. Shi, Q. Tian, J. Li and L. Hou, Reasoning over Hierarchical Question Decomposition Tree for Explainable Question Answering, *ArXiv preprint* **abs/2305.15056** (2023). https://arxiv.org/abs/2305.15056.

[57] S. Yao, D. Yu, J. Zhao, I. Shafran, T.L. Griffiths, Y. Cao and K. Narasimhan, Tree of Thoughts: Deliberate Problem Solving with Large Language Models, *ArXiv preprint* **abs/2305.10601** (2023). https://arxiv.org/abs/2305.10601.

[58] D. Vrandecic and M. Krötzsch, Wikidata: a free collaborative knowledgebase, *Commun. ACM* **57**(10) (2014), 78–85. doi:10.1145/2629489.

[59] T. Gupta and A. Kembhavi, Visual Programming: Compositional visual reasoning without training, *ArXiv preprint* **abs/2211.11559** (2022). https://arxiv.org/abs/2211.11559.

[60] Z. Yao, Y.-K. Chen, X. Lv, S. Cao, A. Xin, J. Yu, H. Jin, J. Xu, P. Zhang, L. Hou and J. Li, VisKoP: Visual Knowledge oriented Programming for Interactive Knowledge Base Question Answering, *ArXiv preprint* **abs/2307.03130** (2023). https://arxiv.org/abs/2307.03130.

[61] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C.L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L.E. Miller, M. Simens, A. Askell, P. Welinder, P.F. Christiano, J. Leike and R.J. Lowe, Training language models to follow instructions with human feedback, *ArXiv preprint* **abs/2203.02155** (2022). https://arxiv.org/abs/2203.02155.