EDITOR **MARK CAMPBELL**
EVOTEK; mark@evotek.com

Connecting AI: Merging Large Language Models and Knowledge Graph

Mlađan Jovanović, Singidunum University
Mark Campbell, EVOTEK

Combining the generative abilities of large language models with the logical and factual coherence of knowledge graphs using a connected artificial intelligence architecture minimizes each system's shortcomings and amplifies their strengths across many real-world domains.

The rise of artificial intelligence (AI) models in fields like image recognition and natural language processing has been driven by general advancements in neural computing, and in particular, multilayered architectures such as deep learning. Recent developments in machine learning attention mechanisms and generative models, such as transformers, have greatly accelerated AI adoption. Many newer AI applications require more than just the pattern recognition offered by mainstream deep learning models and are grappling with more complex tasks like fact-checking, general reasoning, real-time learning, and performant large-scale inference.

Although generative AI architecture like large language models (LLMs) have demonstrated remarkable success in handling diverse queries, they nevertheless face limitations in capturing, learning, and recalling factual and contextual knowledge beyond a short session window. By contrast, knowledge graphs (KGs) are structured data models capable of explicitly learning

Digital Object Identifier 10.1109/MC.2023.3305206
Date of current version: 18 October 2023

and permanently storing rich factual knowledge in 3D. However, constructing and maintaining KGs can be challenging, and they often struggle in incomplete or dynamic data spaces.

This raises the intriguing possibility of combining LLMs and KGs. Would an LLM and KG hybrid leverage the strengths of each while addressing their weaknesses, resulting in a more powerful and comprehensive knowledge processing platform? How would this integration take place? And what challenges would this face?

LLMS

LLMs are commonly deployed using in-context learning, wherein their behavior is controlled through prompting and conditioning on relevant contextual data.[1] The typical workflow preprocesses the training data, decomposes relevant documents into chunks, passes them through an embedding model, and stores them in a vector database. Next, prompts and valid output examples based on the ground truth and external context (that is, from beyond the model's training) are fed to the pretrained model for inference and fine-tuning.

Although creating a library of fine-tuning prompts is relatively simple, crafting them to provide the most effective guidance is much more of a challenge and requires robust monitoring mechanisms to optimize training and fine-tuning. One common approach to assess an LLM's robustness, such as Fiddler AI's Auditor platform, uses a second monitoring LLM to measure the sensitivity of the first LLM's response to varying inputs. By generating variations of an original prompt and evaluating semantic similarities across the LLM's responses, the monitoring LLM can help speed fine-tuning, increase model robustness, and identify performance issues such as data drift in text embeddings.[2]

LLM CHALLENGES

Despite meteoric adoption across many different domains, LLMs have significant shortcomings and drawbacks, including the following:

- *Factual accuracy*: Research has demonstrated that LLMs struggle to recall actual facts and are prone to generating inaccurate information, known as *hallucinations*, and false assertions.[3] LLMs can also exhibit unexpected errors when confronted with examples that do not conform to the patterns learned during training. "GPTs can produce useful content, but when it comes to decisions where high accuracy is a critical requirement, we cannot rely on them," Peter Voss, CEO and chief scientist at Aigo.ai points out.[4]
- *Data poisoning*: In addition, models exposed to maliciously altered data can sometimes alter their behavior in ways imperceptible to humans. Various techniques have been employed to mitigate these errors, such as prompting pretrained models[5] or employing human-in-the-loop reinforcement for fine-tuned models.[6]
- *Bias and opaqueness*: LLMs have a propensity to amplify social, cultural, demographic, and various other biases present in the training data.[7] Moreover, explaining a model's decision-making process is typically ambiguous or even impossible.[8] This opaqueness is particularly concerning in applications related to medical, financial, legal, and autonomous systems.
- *Large infrastructure*: LLM workflows depend heavily on computationally intensive and memory-demanding base models that do not scale linearly. The continuous growth of model parameters, scaling token vocabulary, and increasing training memory has reached a point of diminishing returns.[9] Additionally, the physical, technical, and organizational challenges of continually deploying more computation and data infrastructure makes the cost and complexity of endless scaling untenable, particularly for models like ChatGPT,[10] leaving LLMs in the domain of only the largest implementors.
- *Data access*: LLM training is constrained by data access, which is increasingly expensive due to copyright complexities, privacy concerns, regulations, data fees, Web3 (where users store data in personal vaults or crypto wallets), geopolitical factors, and datasets contaminated with vast amounts of AI-generated synthetic data. This is causing a shift toward model enhancement through methods not solely reliant on more data. An illustrative example of this approach is chain-of-thought prompting, in which a model is requested to generate steps of logical thinking while simultaneously providing annotations to train smaller models.[11]
- *Real-time learning*: After training and fine-tuning, LLMs remain quite static until another

LLMs are commonly deployed using in-context learning, wherein their behavior is controlled through prompting and conditioning on relevant contextual data.

training cycle. As such, they do not assimilate new facts or patterns while conducting inference and are therefore incapable of real-time learning.

KGS

A KG is a machine-readable representation of real-world knowledge, such as general facts, domain-specific facts, and common-sense maxims, and encompasses modalities beyond text (for example, images, sounds, or video).[4] Typically implemented using a graph database, KGs form a network of digital entities categorized into reference classes (nodes) and the interconnected relationships between them (edges).[12] Upon this node and edge network, KGs build an ontology of concepts, properties, relationships, and instances for the given knowledge domain using a paradigm in which terminological boxes depict conceptualizations, and terminological knowledge and assertional boxes describe instances conforming to those concepts. Each entity class in a KG is defined by properties such as attributes, functions, relations, and meta-attributes and takes the form of general entity types (for instance, a person or event) and domain-specific entity types (for example, health or finance related).

KGs also offer various services ranging from simple operations like create, read, update, delete to advanced functionalities such as semantic search, matching, and navigation for natural language processing tasks. The combination of KG operations and services allows new facts to be integrated as they are encountered, allowing them to learn in real time after construction.

KG CHALLENGES

Although KGs offer various benefits, the also present several challenges:

- *Construction*: KGs are generated by combining structured and unstructured (noisy) data from various sources. Existing methods of knowledge extraction have low accuracy, which can produce inconsistent or incomplete KGs.[13] In addition, non-English datasets are still rare and multimodal datasets remain difficult to extract and represent.
- *Maintenance*: KGs often suffer from incomplete or missing entities, attributes, and relations due to erroneous data sources. Implementers resort to tedious human-in-the-loop techniques like crowdsourcing and expert sourcing to maintain knowledge quality. KGs also struggle to detect and represent how domain and real-world knowledge evolve over time.[14]
- *Interoperability*: Merging distinct KGs presents a significant challenge due to the variety, dissimilarity, and intricacy of data used during construction.[15] Locating identical entities across KGs is a formidable task due to semantic differences in schemas and concepts, complicated by language polysemy (that is, similar entities having different meaning across KGs) and entities with a variety of modalities.

KGs are generated by combining structured and unstructured (noisy) data from various sources.

COMBINING LLMS AND KGS

An LLM and KG hybrid offers the possibility of synergizing strengths and overcoming challenges (see Figure 1). Although LLMs embody patterns to create synthetic artifacts and KGs provide structure for semantic and factual data, their integration can identify entities and handle diverse descriptions efficiently, advancing overall capabilities significantly.[16]

In current LLM and KG integrations, each exchanges its information with the other, yet each functions as a distinct element.[16] LLMs are adept at discovering knowledge, while KGs compile this knowledge in a reinforcing feedback loop, leading to ongoing enhancements and expanded capabilities. LLMs dynamically generate, maintain, and expand KGs, while KGs offer refining prompts along with pertinent context to train LLMs and validate responses. For instance, OntoGPT[17] and GraphGPT[18] utilize prompting to extract schema-based information and populate the knowledge base.[19] SensEmBERT generates semantic representations of word meanings in the form of vectors, akin to LLM's contextualized word embeddings, connecting a word's occurrence and its meaning

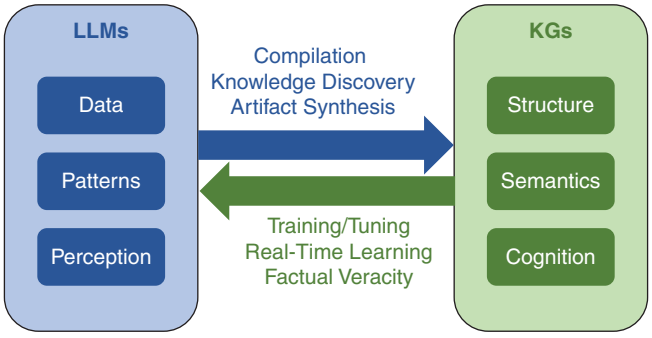

FIGURE 1. The LLM and KG combination.

to disambiguating word senses across multiple languages.[20]

Using automated pipelines, LLMs can also be connected to sources besides KGs, such as application programming interfaces (APIs), documents, and tabular data, to enhance their reasoning capabilities.[21,22] However, the efficacy of these integrations depends on the LLM's recognition capacity and how seamlessly each component can function in a composition of services.

THE CONNECTED AI ARCHITECTURE

Cognitive AI has driven the development of architectures that mimic human behavior and reasoning in machines.[23] Recent AI advancements have renewed interest in enabling systems to "understand" by integrating multiple knowledge sources.[24] However, bridging the gap between machine perception and cognition remains a challenge for operational AI systems attempting to accurately perceive and reason about their surroundings.

The emerging area of neurosymbolic computing[25] combines neural networks' pattern-recognition capabilities with KGs' reasoning abilities using one of the following two methods:

1. *Compression and vectorization*: This method compresses knowledge structures into vectorized representations suitable for neural networks. Vectorization generates multidimensional vector representations for graph triples (knowledge embedding)[26] using techniques like manifold learning, topological data analysis, graph neural networks, and generative graph models to capture the structure and semantics of the network. Although compressed knowledge enhances the reasoning capabilities of LLMs and orchestration pipelines,[21,22] it does lose some of the original semantics in the produced representations and encoded textual entities.[27] Nevertheless, such representations become "regularizers" by providing more flexible responses through constraining the neural network search problem, and by categorizing LLMs' outputs for verification. This allows KGs to instill rigor in LLMs behavior, akin to verifying a computer program.
2. *Pattern extraction*: This technique links neural patterns with symbolic knowledge by extracting pertinent pattern information.[25] Pattern extraction is predominantly implemented using end-to-end pipelines that incorporate differentiable LLM and KG components.[21,22]

Neurosymbolic computation techniques can be realized in a platform we have dubbed the "connected AI architecture" (see Figure 2), which uses bidirectional graph-to-vector and vector-to-graph links between perception (LLMs) and cognition (KGs) to track and interpret content exchanged between them. In this method, a KG encoder translates graphs into an intermediate format compatible with corresponding transformations

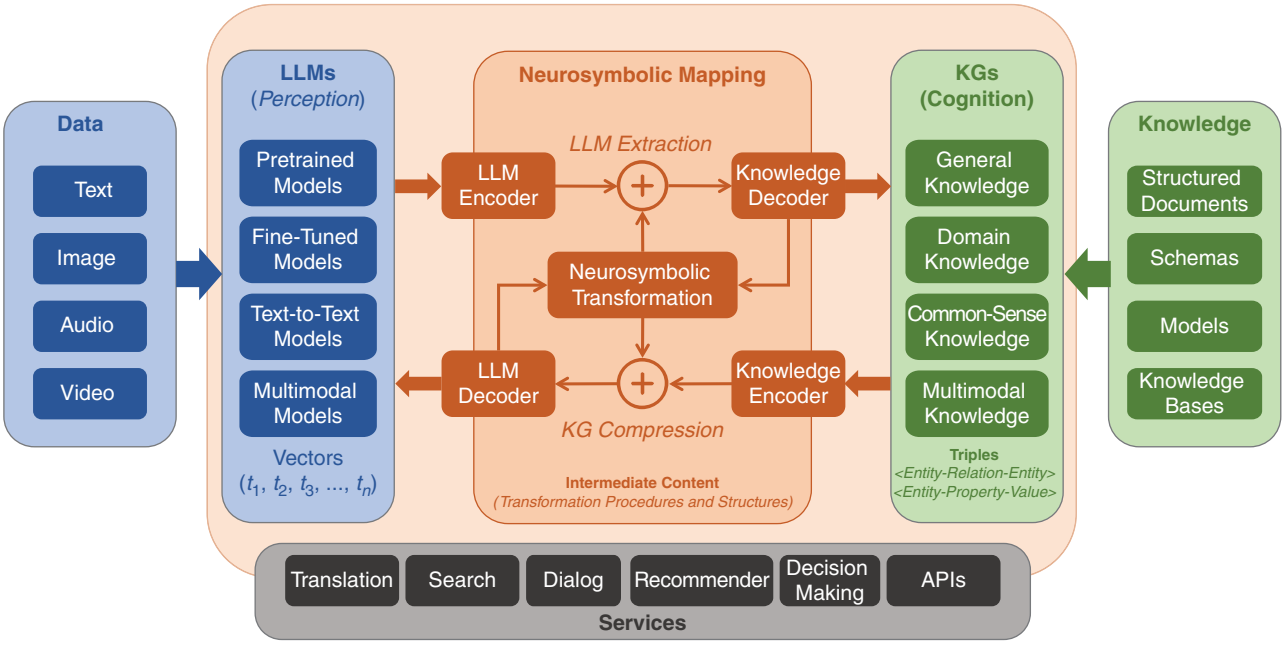

FIGURE 2. The connected AI architecture.

(for instance, graph embedding or masking methods), and then an LLM decoder reconstructs and validates these transformations, yielding vector representations. When applied in an iterative fashion, this can achieve desired performance levels, such as minimizing compression loss. Likewise, LLM extraction encodes vectors, analyzes acquired patterns to extract entities and relationships, and uses these to create triple-like structures for ingestion by the target graph. An iterative feedback loop ensures that the generated triples attain logical coherence and consistency.

The connected AI architecture bridges the gap between machine semantics and the evolving context of the physical world it mirrors. Additionally, creating simpler probabilistic and formal components establishes causal relations between digital and physical objects, capturing real-world dynamics. This enables discovery of how digital entities correspond to real-world objects, assesses whether a digital action results in the desired real-world outcome, and identifies which digital counterparts are affected by this outcome. These real-time semantics can be accessed remotely through APIs and integrated into various services like translation, search, and dialogue functions.

CHALLENGES

To implement and deploy a successful connected AI architecture, one must address the following critical requirements:

- *Trust*: One must trust the underlying LLMs and KGs to trust the overall architecture. Although there are several techniques under development that assess the trustworthiness of LLMs,[7] these need to be extended to the larger connected AI architecture. Beyond this, it is crucial to educate end users on system interaction, intended use cases, expected and unexpected behaviors, and potential repercussions from exceeding them.
- *Explainability*: "Language models have a vector representation based on word distance, which is very opaque, and there is no effective way to map vectors to nodes in a KG due to this opaqueness. LLMs can map to KGs using an intermediate representation they produce, but we might need a supervisory level over both due to hallucinations and opaqueness of LLMs," notes Voss.[4] As such, a connected AI system must provide explainability to the decision-making process, focusing on the system's rationale rather than relying solely on posthoc techniques like feature importance.
- *Accuracy*: A connected AI architecture must ensure precise symbolic representation and meaning and be verifiable in a mathematical and unambiguous way given the absence of a shared consensus or guidelines regarding symbolic systems.[25]
- *Randomness*: Algorithmic processes, individual behaviors, and social interactions embody some degree of randomness. Thus, the algorithmic predictions and generative artifacts produced can only reduce error rates to a particular threshold. A rigorous and comprehensive evaluation of systems built on a connected AI architecture must be conducted to minimize errors from randomness.
- *Safety*: LLM risk does not arise from LLMs' statistical capabilities but from their limitations and inherent deceptive abilities (for example, hallucinations, deepfakes, and disinformation). Continuous input and output audits are crucial for detecting data and concept drift, ensuring unbiased outcomes and detecting toxic or harmful decisions and content.

Concerns surrounding LLMs' limitations have led to an increased focus on KG-based downstream applications. Combining LLMs' generative abilities with KGs' logical and factual coherence into a connected AI architecture creates a theoretical framework to maximize capabilities and minimize systemic shortcomings across many real-world domains.

ACKNOWLEDGMENT

We wish to thank Peter Voss and Srini Pagidyala from Aigo.ai for their comments.

REFERENCES

1. J. Yang et al., "Harnessing the power of LLMs in practice: A survey on ChatGPT and beyond," 2023. Accessed: Jul. 17, 2023. [Online]. Available: https://arxiv.org/abs/2304.13712
2. "Fiddler introduces end-to-end workflow for robust generative AI." Fiddler AI. Accessed: Jul. 24, 2023. [Online]. Available: https://www.fiddler.ai/blog/fiddler-introduces-end-to-end-workflow-for-robust-generative-ai
3. F. Petroni et al., "Language models as knowledge bases?" 2019, *arXiv:1909.01066*.
4. P. Voss and S. Pagidyala, "Interviewees," Aigo.ai, Jul. 2023.
5. J. Wei et al., "Chain-of-thought prompting elicits reasoning in large language models," in *Proc. 36th Conf. Neural Inf. Process. Syst.*, New Orleans, LA, USA, 2022. [Online]. Available: https://openreview.net/pdf?id=_VjQlMeSB_J
6. L. Ouyang et al., "Training language models to follow instructions with human feedback," in *Proc. 35th Conf. Neural Inf. Process. Syst.*, New Orleans, LA, USA, 2022, pp. 27,730–27,744.
7. B. Wang et al., "DecodingTrust: A comprehensive assessment of trustworthiness in GPT models," 2023. Accessed: Jul. 24, 2023. [Online]. Available: https://arxiv.org/abs/2306.11698

8. M. Jovanović and M. Schmitz, "Explainability as a user requirement for artificial intelligence systems," *Computer*, vol. 55, no. 2, pp. 90–94, Feb. 2022, doi: 10.1109/MC.2021.3127753.
9. S. Tworkowski, K. Staniszewski, M. Pacek, Y. Wu, H. Michalewski, and P. Miłos, "Focused transformer: Contrastive training for context scaling," 2023. [Online]. Available: https://arxiv.org/abs/2307.03170
10. A. Mok, "ChatGPT could cost over $700,000 per day to operate," *Bus. Insider*, Apr. 2023. [Online]. Available: https://www.businessinsider.in/tech/news/chatgpt-could-cost-over-700000-per-day-to-operate-microsoft-is-reportedly-trying-to-make-it-cheaper-/articleshow/99637548.cms
11. C.-Y. Hsieh et al., "Distilling step-by-step! Outperforming larger language models with less training data and smaller model sizes," 2023, *arXiv:2305.02301*.
12. S. Ji, S. Pan, E. Cambria, P. Marttinen, and P. S. Yu, "A survey on knowledge graphs: Representation, acquisition, and applications," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 2, pp. 494–514, Feb. 2022, doi: 10.1109/TNNLS.2021.3070843.
13. C. Peng, F. Xia, M. Naseriparsa, and F. Osborne, "Knowledge graphs: Opportunities and challenges," *Artif. Intell. Rev.*, early access, Mar. 2023, doi: 10.1007/s10462-023-10465-9.
14. P. Shao, G. Yang, D. Zhang, J. Tao, F. Che, and T. Liu, "Tucker decomposition-based temporal knowledge graph completion," *Knowl.-Based Syst.*, vol. 238, Feb. 2022, Art. no. 107841, doi: 10.1016/j.knosys.2021.107841.
15. H. L. Nguyen, D. T. Vu, and J. J. Jun, "Knowledge graph fusion for smart systems: A survey," *Inf. Fusion*, vol. 61, Sep. 2020, pp. 56–70, doi: 10.1016/j.inffus.2020.03.014.
16. S. Pan, L. Luo, Y. Wang, C. Chen, J. Wang, and X. Wu, "Unifying large language models and knowledge graphs: A roadmap," 2023, *arXiv:2306.08302*.
17. H. J. Caufield et al., "Structured prompt interrogation and recursive extraction of semantics (SPIRES): A method for populating knowledge bases using zero-shot learning," 2023, *arXiv:2304.02711*.
18. V. Espinoza, "GraphGPT: Convert unstructured natural language into a knowledge graph," *Medium*, Mar. 2023. [Online]. Available: https://medium.com/@vespinozag/graphgpt-convert-unstructured-natural-language-into-a-knowledge-graph-cccbee19abdf#:~:text=GraphGPT%20converts%20unstructured%20natural%20language,of%20entities%20and%20their%20relationships
19. P. Liu, W. Yuan, J. Fu, Z. H. Jiang, H. Hayashi, and G. Neubig, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," *ACM Comput. Surv.*, vol. 55, no. 9, pp. 1–35, Sep. 2023, doi: 10.1145/3560815.
20. B. Scarlini, T. Pasini, and R. Navigli, "Sensembert: Context-enhanced sense embeddings for multilingual word sense disambiguation," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 8758–8765, doi: 10.1609/aaai.v34i05.6402.
21. A. Tabir, "Langchain: Building powerful language model applications with Python," *Medium*, Apr. 2023. [Online]. Available: https://medium.com/@Algi.T/langchain-building-powerful-language-model-applications-with-python-c2b77a107709#:~:text=LangChain%20is%20a%20python%20framework,to%20other%20sources%20of%20data
22. K. Wiggers. "LlamaIndex adds private data to large language models." TechCrunch. Accessed: Jul. 27, 2023. [Online]. Available: https://techcrunch.com/2023/06/06/llamaindex-adds-private-data-to-large-language-models/#:~:text=Today%2C%20LlamaIndex%20(the%20company),LLM%20applications%2C%E2%80%9D%20Liu%20said
23. I. Kotseruba and J. K. Tsotsos, "40 years of cognitive architectures: Core cognitive abilities and practical applications," *Artif. Intell. Rev.*, vol. 53, pp. 17–94, Jan. 2020, doi: 10.1007/s10462-018-9646-y.
24. "Cognitive AI research: Higher machine intelligence for next-gen AI," Intel, Satan Clara, CA, USA, 2022. [Online]. Available: https://www.intel.com/content/www/us/en/research/blogs/higher-machine-intelligence-for-next-gen-ai.html
25. A. d'Avila Garcez and L. C. Lamb, "Neurosymbolic AI: The 3rd wave," *Artif. Intell. Rev.*, early access, Mar. 2023, doi: 10.1007/s10462-023-10448-w.
26. M. M. Li, K. Huang, and M. Zitnik, "Graph representation learning in biomedicine and healthcare," *Nature Biomed. Eng.*, vol. 6, pp. 1353–1369, Dec. 2022, doi: 10.1038/s41551-022-00942-x.
27. X. Wang et al., "KEPLER: A unified model for knowledge embedding and pre-trained language representation," *Trans. Assoc. Comput. Linguistics*, vol. 9, pp. 176–194, Nov. 2020, doi: 10.1162/tacl_a_00360.

MLAĐAN JOVANOVIĆ is an associate professor at Singidunum University, 11000 Belgrade, Serbia. Contact him at mjovanovic@singidunum.ac.rs.

MARK CAMPBELL is the chief innovation officer at EVOTEK, San Diego, CA 92121 USA. Contact him at mark@evotek.com.