Investigando uma abordagem baseada em aprendizado em contexto para geração de consultas SPARQL

Carlos Eduardo A. Ferreira[1], **Joel L. Carbonera**[1]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

Abstract. *Interacting with knowledge graphs generally requires familiarity with some formal query language. To make this process more accessible, knowledge graphs question-answering systems aim to enable querying through natural language. One approach to designing such systems involves translating natural language questions into a formal graph query language, such as SPARQL. This study aims to evaluate the use of in-context learning for generating SPARQL queries, specifically by providing an empirical analysis of the results obtained from varying certain experimental parameters.*

Resumo. *Interagir com grafos de conhecimento geralmente requer familiaridade com alguma linguagem formal de consulta. Para tornar esse processo mais acessível, sistemas de perguntas e respostas baseados em grafos de conhecimento buscam permitir consultas a partir de perguntas em linguagem natural. Uma forma de projetar esses sistemas envolve a tradução de perguntas em linguagem natural para uma linguagem formal de consulta em grafos, como o SPARQL. Este estudo tem como objetivo avaliar o uso de aprendizagem em contexto na geração de consultas SPARQL, mais especificamente contribuir com uma análise empírica dos resultados provenientes da variação de certos parâmetros experimentais.*

1. Introdução

Grafos de conhecimento (KGs) são informações organizadas por meio de triplas sujeitas a uma ontologia, representando entidades, propriedades e relações de maneira precisa e estruturada [Hogan et al. 2021]. Essas representações permitem capturar o significado e as interconexões entre conceitos, sendo amplamente utilizadas em aplicações como mecanismos de busca [Wang et al. 2019] e sistemas de recomendação [Yang et al. 2022], por possibilitarem a recuperação e integração de conhecimento confiável e estruturado em larga escala. Modelos de linguagem de grande escala (LLMs), por sua vez, são redes neurais treinadas com grandes volumes de texto, capazes de gerar conteúdo textual coerente e contextualizado em diversas tarefas [Håkansson and Phillips-Wren 2024]. Tanto a comunidade científica quanto o setor industrial têm demonstrado crescente interesse em integrar KGs e LLMs, com o objetivo de combinar a fluência e flexibilidade textual dos LLMs com a precisão e confiabilidade semântica oferecidas pelos KGs. Em particular, há uma demanda por sistemas de perguntas e respostas sobre grafos de conhecimento (KGQA) que operem com precisão em domínios específicos, mesmo em contextos de corporações com recursos limitados para o ajuste fino de LLMs [Agarwal et al. 2024, Xie et al. 2024]. Nesse contexto, este estudo propõe uma análise

empírica sobre como diferentes parâmetros experimentais – tais como o número de exemplos fornecidos em contexto, as técnicas utilizadas para selecionar esses exemplos e o fornecimento ou não de URIs de referência no prompt – influenciam a geração de consultas SPARQL a partir de perguntas em linguagem natural empregando uma abordagem baseada em ICL (*in-context learning*).

2. Metodologia

Esta seção descreve a metodologia adotada neste estudo. Inicialmente, apresenta-se uma visão geral do pipeline desenvolvido para a condução dos experimentos. Em seguida, são detalhadas as configurações experimentais utilizadas, incluindo os parâmetros avaliados, o LLM utilizado e o *benchmark* considerado. Por fim, são apresentadas as métricas adotadas para avaliar as consultas SPARQL geradas.

2.1. Pipeline de KGQA baseado em ICL

Para realizar os experimentos foi desenvolvido um pipeline baseado em ICL para KGQA, selecionando pares relevantes `<NL Question, SPARQL>` a fim de criar *prompts* que auxiliam um LLM a gerar consultas SPARQL para perguntas de usuários. Os exemplos fornecem todo o contexto disponível sobre o KG subjacente, embora seja importante reconhecer a possibilidade de contaminação no pré-treino. O pipeline recebe como entrada uma pergunta em linguagem natural (NL) formulada pelo usuário e tem acesso a um conjunto de pares que servem como base de exemplos. Dada a pergunta do usuário, um conjunto de n exemplos relevantes é selecionado e, juntamente com a própria pergunta do usuário, usado para construir o *prompt* de acordo com um template pré-definido.

Os *prompts* gerados a partir dos templates são submetidos ao LLM, que produz uma resposta textual que pode conter elementos indesejados, tais como quebras de linha, espaçamento inconsistente, URIs incompletas ou até mesmo explicações em linguagem natural. Apesar de ser instruído a gerar apenas a consulta, o modelo frequentemente inclui conteúdo adicional na saída. Portanto, para garantir que as consultas possam ser processadas e avaliadas, aplicamos uma etapa de limpeza como pós-processamento da saída. Isso inclui a remoção de quebras de linha e espaços redundantes, a resolução de prefixos abreviados (por exemplo, dbpedia-owl:Class) em URIs completas e a exclusão de qualquer conteúdo antes ou depois do bloco SPARQL.

2.2. Configuração Experimental

O *benchmark* LC-QuAD 1.0 [Trivedi et al. 2017] possui 5.000 tuplas `<NL Question, SPARQL>`, sendo 4.000 no conjunto de treino e 1.000 no de teste. Para cada elemento do teste, foram selecionados n exemplos do treino, com $n \in \{1, 5, 10, 15, 20\}$. Dessa maneira, foram construídos 14.000 prompts, posteriormente processados pelo LLM[1]. O núcleo dos nossos experimentos consistiu nas variações que adotamos: **(1)** quantidade de exemplos selecionados; **(2)** seleção de exemplos pela similaridade de cosseno entre os *embeddings* das NL Questions; **(3)** mesma estratégia da abordagem 2, mas incluindo URIs de referência ao final do prompt; **(4)** aplicação de k*-means* para agrupar *embeddings*, selecionando uma tupla por grupo. A variação 4 buscou incorporar diversidade nos exemplos selecionados, nessa variação não foi realizado o experimento com $n = 1$,

[1]LLama 3 70B [Meta 2024] com temperatura configurada em 1.0

pois trabalhar com um único *cluster* resultaria em um cenário equivalente ao da variação 2 com $n = 1$.

2.3. Métricas de Avaliação

A métrica F1 foi calculada a partir da comparação entre as respostas retornadas pela DBpedia, tanto para a query de referência quanto para as queries geradas pelo LLM, constituindo, assim, uma avaliação no nível dos resultados do *endpoint* do grafo. Essa métrica mede o equilíbrio entre precisão (proporção de resultados corretos entre todos os retornados) e revocação (proporção de resultados corretos entre todos os esperados), oferecendo uma visão integrada da efetividade da query gerada.

Ademais, também foi utilizada a métrica BLEU [Papineni et al. 2002], aplicada às queries SPARQL depois de tokenizá-las, tanto as de referência quanto as geradas. Essa métrica, apesar de originalmente ser utilizada para avaliar a qualidade de traduções, é amplamente empregada na literatura para avaliar a geração de SPARQL. Valores elevados da métrica indicam maior correspondência estrutural e lexical da query gerada, sugerindo que ela se aproxima sintaticamente da referência.

3. Resultados e Discussão

Nas condições experimentais avaliadas, os melhores resultados obtidos em F1 foram $35,75\%$ na variação 2 (sendo $n = 20$), $62,66\%$ na variação 3 (sendo $n = 15$) e $33,90\%$ na variação 4 (sendo $n = 15$). Na métrica BLEU, por sua vez, os melhores resultados obtidos foram $67,91\%$ na variação 2, $78,26\%$ na variação 3 e $66,38\%$ na variação 4, sendo $n = 20$ em todas as variações. Para ambas as métricas aplicadas, os ganhos mais notáveis ocorrem entre um e cinco exemplos selecionados. Os experimentos apresentam ainda um padrão de retornos decrescentes à medida que mais exemplos são incluídos. Essa tendência destaca o valor da seleção de exemplos representativos.

As métricas de performance oferecem uma visão geral do desempenho, mas não identificam os padrões de falha. Para complementar a avaliação, também foi conduzida uma análise de erros com o objetivo de entender as falhas do modelo em diferentes configurações de *prompts* e estratégias de seleção de exemplos. Entre os tipos de erros identificados, encontram-se: `Fake URI`, refere-se a instâncias em que o modelo gera URIs que não existem no KG de destino; `Triple-Flip`, ocorre quando o modelo inverte incorretamente o sujeito e o objeto em uma tripla; *Parsing Error*, ocorre quando o modelo não gera uma sintaxe SPARQL válida. O ponto de maior destaque na análise de erros foram as quedas abruptas nos erros de *parsing* comparando os resultados em $n = 1$ e $n = 5$, indo de $30,78\%$ para $1,80\%$ na variação 2 e de $14,06\%$ para $1,74\%$ na variação 3. Entretanto, para $n > 5$ os erros de *parsing* chegam a no mínimo 0.98%, o que não representa diferenças significativas. Essa redução expressiva sugere que o aumento no número de exemplos fornecidos em contexto contribui significativamente para a estabilização sintática da geração, auxiliando o modelo a internalizar o formato esperado das consultas SPARQL. No entanto, a estabilização observada para $n > 5$ indica que esse efeito tende a saturar, sugerindo um limite prático no ganho de exemplos adicionais.

4. Conclusões

Os resultados sugerem que o uso de ICL pode representar uma abordagem promissora para a geração de consultas SPARQL em cenários sem ajuste fino, sobretudo quando se

empregam estratégias capazes de selecionar exemplos relevantes para cada pergunta. A seleção por similaridade combinada ao fornecimento de URIs de referência foi a mais eficaz tanto em F1 quanto em BLEU, enquanto o agrupamento com k-means apresentou desempenho inferior, sugerindo que, no contexto deste problema, o agrupamento de embeddings das perguntas pode não ser a estratégia mais adequada para introduzir diversidade na seleção de exemplos. O efeito positivo do fornecimento de URIs reforça que o principal desafio da tarefa está na desambiguação de entidades e relações, bem como na correspondência entre os elementos da pergunta e os recursos do grafo. Em conjunto, os resultados indicam que a eficácia do ICL em tarefas de geração de consultas SPARQL está menos associada ao volume de exemplos e mais à capacidade de selecionar exemplos realmente relevantes. Esses achados ressaltam a importância de estudos experimentais que, como este, analisam sistematicamente o impacto da variação de diferentes parâmetros e estratégias de seleção, contribuindo para compreender o que de fato caracteriza exemplos relevantes para a tarefa. Nesse sentido, investigações futuras podem explorar como distintas formas de selecionar e complementar tuplas de exemplos podem aproveitar de modo mais eficiente a janela de contexto disponível, aprimorando o desempenho de abordagens de KGQA baseadas em ICL.

Referências

Agarwal, P., Kumar, N., and Bedathur, S. (2024). Symkgqa: few-shot knowledge graph question answering via symbolic program generation and execution. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10119–10140.

Håkansson, A. and Phillips-Wren, G. (2024). Generative ai and large language models-benefits, drawbacks, future and recommendations. *Procedia Computer Science*, 246:5458–5468.

Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., Melo, G. D., Gutierrez, C., Kirrane, S., Gayo, J. E. L., Navigli, R., Neumaier, S., et al. (2021). Knowledge graphs. *ACM Computing Surveys (Csur)*, 54(4):1–37.

Meta (2024). Meta-llama-3-70b-instruct. Accessed: 2025-04-15.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Trivedi, P., Maheshwari, G., Dubey, M., and Lehmann, J. (2017). Lc-quad: A corpus for complex question answering over knowledge graphs. In *The Semantic Web–ISWC 2017: 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part II 16*, pages 210–218. Springer.

Wang, P., Jiang, H., Xu, J., and Zhang, Q. (2019). Knowledge graph construction and applications for web search and beyond. *Data Intelligence*, 1(4):333–349.

Xie, X., Wang, J., Han, Y., and Li, W. (2024). Knowledge graph-based in-context learning for advanced fault diagnosis in sensor networks. *Sensors*, 24(24):8086.

Yang, Y., Huang, C., Xia, L., and Li, C. (2022). Knowledge graph contrastive learning for recommendation. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*, pages 1434–1443.