Uma Ferramenta Web Baseada em LLMs para Respostas a Perguntas Simples sobre Bases de Conhecimento Heterogêneas

João Pedro Porto Campos
joao.porto@ibm.com
IBM Research, Rio de Janeiro, Brazil
UNIRIO, Rio de Janeiro, Brazil

Marcelo Machado
mmachado@ibm.com
IBM Research, Rio de Janeiro, Brazil

Guilherme Lima
guilherme.lima@ibm.com
IBM Research, Rio de Janeiro, Brazil

Viviane Torres da Silva
vivianet@br.ibm.com
IBM Research, Rio de Janeiro, Brazil

ABSTRACT

We present KIF-QA, a Web tool for answering simple questions over heterogeneous knowledge graphs. The Web tool is built on top of a homonymous framework, KIF-QA, that uses in-context learning over off-the-shelf pre-trained large language models for semantic parsing. Because it relies on in-context (few-shot) learning to generate logical forms and disambiguate entities, and because it uses KIF (a knowledge integration framework based on Wikidata) to mediate all access to the target knowledge graphs, KIF-QA's approach is general. It requires neither training nor fine-tuning and can be easily adapted to work with different graphs. The KIF-QA Web tool allows the user to change the target graphs (Wikidata, DBpedia, or PubChem) and the language model used. The user can also view and modify the details of each step taken by KIF-QA to generate the answers to the question. Both the Web tool and the KIF-QA framework are released as open-source.

KEYWORDS

knowledge base question answering, pre-trained large language model, in-context learning, Wikidata, SPARQL, KIF

1 INTRODUÇÃO

Sistemas de respostas a perguntas sobre bases de conhecimento (KBQA, do inglês, *knowledge base question answering*) permitem que usuários não-especialistas consultem bases de conhecimento estruturadas usando linguagem natural [2, 5]. O usuário faz perguntas diretamente ao sistema, sem a necessidade de conhecer o esquema da base ou dominar a sua linguagem de consulta. Quando a base em questão é um grafo de conhecimento, como Wikidata [10] ou DBpedia [6], as respostas são dadas nos termos de nós (entidades) ou arestas (asserções ou triplas) do grafo. Por exemplo, dada a pergunta simples "Onde nasceu James Brown?", um sistema de KBQA pode apresentar como resposta a entidade "Barnwell, SC, USA", que corresponde à entidade wd:Q586262 no Wikidata e dbr:Barnwell,_South_Carolina no DBpedia.

Sistemas de KBQA são classificados de acordo com o tipo de pergunta as quais foram projetados para responder (perguntas simples versus complexas) e o método utilizado para produzir a resposta (análise semântica versus recuperação de informação) [5]. *Perguntas simples* como "Qual é a capital do Brasil?" envolvem uma única aresta do grafo, enquanto *perguntas complexas* envolvem múltiplas arestas ou operações complexas (e.g., agregação ou aritmética). Em sistemas KBQA baseados em *análise semântica* (*semantic parsing*), as respostas são produzidas transformando-se a pergunta numa consulta (forma lógica) que é então executada sobre o grafo. Já em sistemas baseados em *recuperação de informação*, as respostas são obtidas através de buscas no grafo, sem a utilização de uma consulta propriamente dita.

Neste artigo apresentamos uma ferramenta Web baseada em modelos de linguagem ((L)LMs, *(large) language models*) e análise semântica para repostas a perguntas simples sobre bases de conhecimento heterogêneas. A ferramenta se chama KIF-QA e serve de interface Web para um *framework* homônimo. O *framework* KIF-QA [8] usa aprendizagem em contexto (*in-context learning*) sobre LLMs de prateleira para análise semântica e é implementado sobre o KIF [7] (um *framework* para integração de bases conhecimento baseado no Wikidata). Por depender apenas de aprendizagem em contexto (*few-shot*) para geração de formas lógicas e desambiguação e por usar o KIF para mediar todo acesso ao grafo subjacente, a abordagem do KIF-QA é geral: não necessita de nenhum tipo de treinamento ou ajuste fino (*fine-tuning*) e pode ser facilmente adaptada para funcionar sobre bases (grafos) heterogêneos. Além disso, em uma análise recente [8], o *framework* KIF-QA apresentou desempenho competitivo, comparável ao estado-da-arte.

A ferramenta Web KIF-QA permite que o usuário faça perguntas simples sobre um ou mais grafos de conhecimento e obtenha como resposta asserções (*statements*) à la Wikidata instanciadas com termos dos grafos alvo. Além de trocar os grafos alvo (Wikidata, DBpedia ou PubChem [4]) e o modelo de linguagem utilizado, a ferramenta permite que o usuário visualize e edite cada passo usado pelo sistema para gerar as repostas. Tanto a ferramenta Web quanto o *framework* KIF-QA estão disponíveis em código aberto.[1] Um vídeo demonstrando o uso da ferramenta Web está disponível em https://ibm.biz/BdeyUt.

O restante do artigo está organizado da seguinte forma. A Seção 2 apresenta o *framework* KIF e discute trabalhos relacionados.A Seção 3 descreve a arquitetura e implementação da ferramenta Web. A Seção 4 apresenta nossas conclusões e trabalhos futuros.

[1] https://github.com/IBM/kif-llm

In: XXIV Workshop de Ferramentas e Aplicações (WFA 2025). Anais Estendidos do XXXI Simpósio Brasileiro de Sistemas Multimídia e Web (WFA'2025). Rio de Janeiro/RJ, Brasil. Porto Alegre: Brazilian Computer Society, 2025.

ISSN 2596-1683

2 BACKGROUND

2.1 KIF

KIF [7][2] é um *framework* Python de código aberto que usa o modelo de dados e o vocabulário do Wikidata como *lingua franca* para integrar fontes de conhecimento heterogêneas. A integração é virtual, acontece em tempo de consulta, e é guiada por mapeamentos fornecidos pelo usuário. O *framework* conta com mapeamentos pré-definidos para o Wikidata, DBpedia e PubChem, além de outras bases abertas. Novos mapeamentos podem ser criados e adicionados programaticamente.

O componente básico do *framework* KIF é a *store*. Uma *store* KIF é uma interface para uma fonte de conhecimento, normalmente, um grafo de conhecimento. Dada uma *store* KIF, é possível filtrar por asserções (*statements*) cujo sujeito, propriedade ou valor satisfaçam determinadas restrições. Por exemplo, dada uma *store* Wikidata `kb`, a chamada "`kb.filter(wd.Q(7251), wd.P(184))`" busca em `kb` por asserções tais que o sujeito é `wd:Q7251` (Alan Turing) e a propriedade é `wd:P184` (orientador de doutorado). O resultado é um *stream* de asserções que satisfazem o filtro, nesse caso, apenas

(**Statement** (**Item** Alan Turing)
(**ValueSnak** (**Property** doctoral advisor) (**Item** Alonzo Church))) .

Independentemente da base consultada, as asserções retornadas são *statements* que seguem o modelo de dados do Wikidata. Se executarmos um filtro equivalente ao anterior numa *store* DBpedia, por exemplo, vamos obter *statements* seguindo o mesmo formato (sintaxe), porém contendo URIs do DBpedia.

Além de filtros simples, como o anterior, o *framework* KIF suporta filtros complexos, contendo restrições compostas ("e", "ou") e caminhos de propriedades (*property paths*). Também é possível obter as anotações à la Wikidata (qualificadores, referências, e *rank*) associadas aos *statements*. O *framework* também possui componentes *search* para busca textual de entidades num dado espaço de nomes (*namespace*). Como veremos adiante, o KIF-QA usa os *searchers* KIF para busca textual de entidades e as *stores* KIF para consultas aos grafos alvo—a *query* final gerada pelo KIF-QA a partir da pergunta do usuário é uma união de filtros KIF.

2.2 Interfaces de Usuário para KBQA

Bases de conhecimento públicas como Wikidata [10], DBpedia [6] e PubChem [4], normalmente disponibilizam dois tipos de interface de consulta: interfaces para consulta estruturada (*query*) e interfaces para busca textual. As primeiras (e.g., no caso de bases RDF [11], *endpoints* SPARQL [12]) são voltadas para usuários especialistas pois requerem não só conhecimento da linguagem de consulta mas também do esquema da base. (Há na Web ferramentas que auxiliam usuários não-especialistas a construir *queries* usando apenas elementos visuais, e.g., Wikidata Query Builder[3], mas mesmo essas requerem algum conhecimento prévio da linguagem de consulta ou do esquema da base.)

Já as interfaces de busca textual permitem que usuários não-especialistas façam buscas simples na base usando palavras-chave. O resultado é uma lista de entidades com rótulos ou descrições similares às palavras-chave fornecidas. Apesar do suporte ao uso de termos em linguagem natural, normalmente não é possível fazer perguntas. Por exemplo, no PubChem [4] é possível buscar por compostos químicos a partir de uma expressão em linguagem natural, e.g., "manter acordado" para CID2519 (cafeína). Wikidata e DBpedia fornecem interfaces similares.

A proposta da ferramenta Web KIF-QA pode ser vista como um meio-termo entre as interfaces tradicionais de *query* e de busca textual, porém com foco em repostas a perguntas. Ao mesmo tempo que permite ao usuário obter respostas estruturadas a perguntas simples escritas em linguagem natural (ver Figura 1a), a interface da ferramenta Web KIF-QA possibilita a inspeção e modificação dos artefatos intermediários (includindo as entidades e filtros KIF) usados para obter a resposta final (ver Figura 1c). Esses *artefatos intermediários* funcionam como "explicação" da reposta obtida que, por se tratar de uma *stream* de *statements*, também é inspecionável. (Contraste isso com o processo fim-a-fim do tipo "caixa preta" e as respostas em linguagem natural fornecidas por interfaces conversacionais baseadas em modelos de linguagem, i.e., *chat-bots*.)

Algumas propostas recentes para KBQA também incluem interfaces voltadas para usuários não-especialistas. O sistema M3M [9][4], por exemplo, inclui uma interface gráfica para repostas a perguntas simples que mostra um *ranking* de respostas prováveis e a tripla que responde a pergunta no grafo (no caso, Wikidata). Outro exemplo é a ferramenta Drugs4COVID[5] que usa o sistema MuHeQA [1] para implementar uma interface de respostas a perguntas simples sobre Wikidata, DBpedia e outras bases. No Drugs4COVID as respostas são ranqueadas por similaridade com a pergunta e é possível mudar a base alvo. Nenhuma das duas ferramentas, M3M ou Drugs4COVID, permite inspecionar ou modificar os artefatos intermediários utilizados para gerar a resposta. Além disso, no caso do Drugs4COVID, antes de serem apresentadas, as triplas finais são verbalizadas e re-ranqueadas por similaridade com a pergunta, o que as torna menos inspecionáveis.

3 ARQUITETURA E IMPLEMENTAÇÃO

3.1 Visão Geral

A arquitetura da ferramenta KIF-QA consiste de dois componentes principais: *front-end* e *back-end* (ver Figura 2). O *front-end* implementa a interface de usuário da ferramenta. Já o *back-end* implementa a sua interface de programação, i.e., uma API REST construída sobre o *framework* KIF-QA.

A Figura 1a mostra a tela principal da ferramenta KIF-QA. Conforme ilustrado na figura, a ferramenta recebe uma pergunta simples em linguagem natural (*Question*) e a especificação dos grafos de conhecimento alvo (*Sources*). O botão *Ask* dispara o processamento da pergunta. As respostas, i.e., asserções à la Wikidata dos grafos alvo que respondem a pergunta, são exibidas na seção *Statements*, seguindo um formato similar ao adotado pelo Wikidata.

Uma vez geradas as respostas, o usuário pode selecionar o botão *Details* para visualizar o passo-a-passo do processamento (ver Figura 1c). Os artefatos gerados em cada etapa do processamento podem ser inspecionados e editados, e.g., para correção de eventuais erros cometidos pelo modelo de linguagem. O passo-a-passo modificado pode então ser reexecutado através do botão *Run*.

[2] https://github.com/IBM/kif
[3] https://query.wikidata.org/querybuilder/
[4] https://kgqa-nlp-zh.skoltech.ru/
[5] https://drugs4covid.oeg.fi.upm.es/services/bio-qa

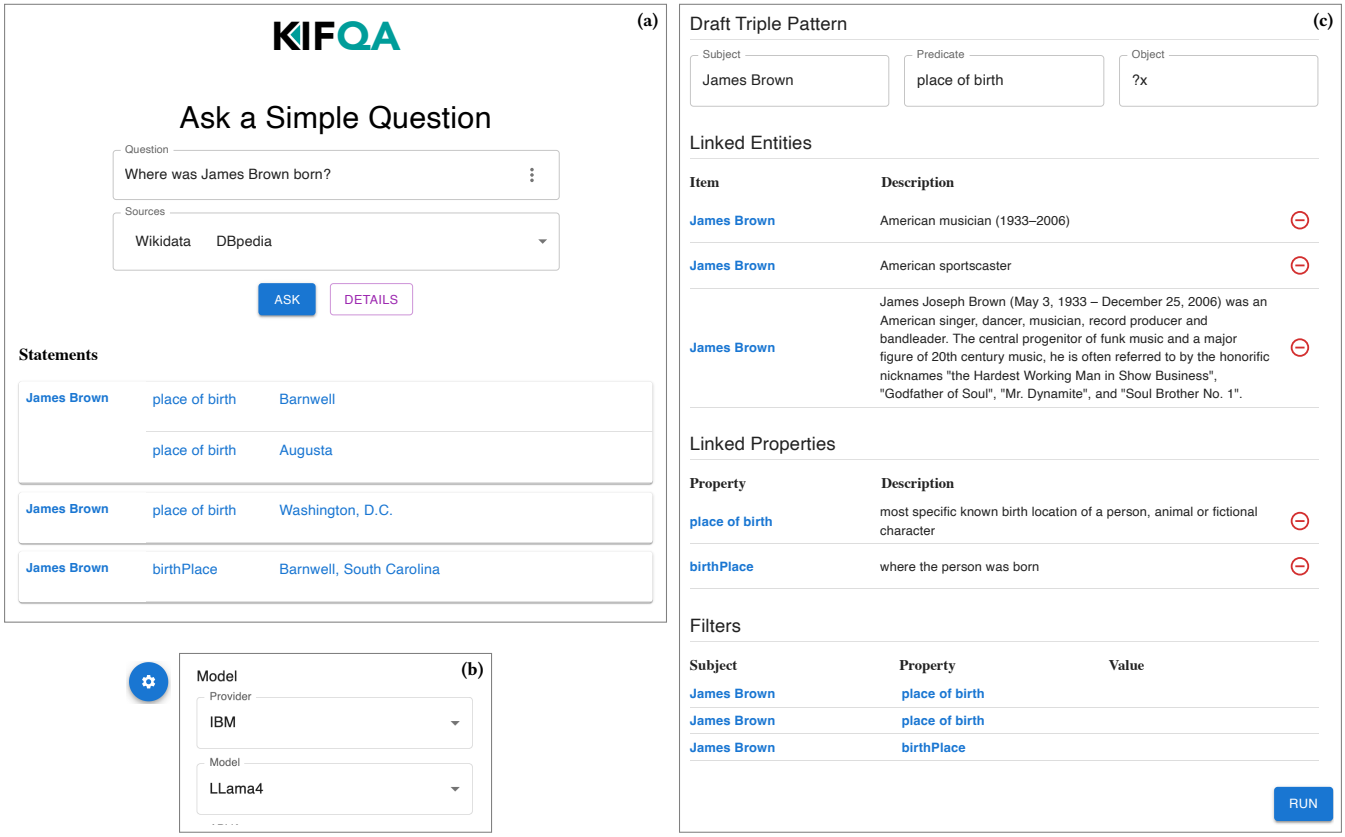

Figura 1: Telas da ferramenta KIF-QA: (a) principal; (b) configuração; e (c) detalhamento (*Details*).

Antes de usar a ferramenta pela primeira vez o usuário precisa configurar o provedor de modelo de linguagem e o modelo a serem utilizados (ver Figura 1b). Atualmente a ferramenta suporta os provedores IBM watsonx[6] e OpenAI[7]. Durante o uso da ferramenta, a qualquer momento, o usuário pode trocar tanto o provedor quanto o modelo de linguagem.

3.2 API REST

A API REST implementa a interface de programação da ferramenta KIF-QA, sobre a qual a interface de usuário é construída. O principal objetivo da API REST é instanciar o *framework* KIF-QA e executá-lo de acordo com as configurações, entradas, e comandos fornecidos ao *front-end* pelo usuário. A API REST também é responsável por processar as respostas do *framework* KIF-QA e entregá-las ao *front-end*, para que este possa exibi-las.

A API REST da ferramenta KIF-QA é desenvolvida em Flask[8], um *framework* Python para construção de aplicações Web. Suas rotas principais são descritas a seguir.

`/config` Configura o provedor e o modelo de linguagem a serem utilizados. (Chamada pela tela de configuração.)

`/stores` Lista os grafos de conhecimento disponíveis, atualmente Wikidata, DBpedia e PubChem. (Chamada para preencher o campo *Sources* na tela principal.)

`/query` Dispara o processamento completo de uma pergunta simples. (Chamada pelo botão *Ask* da tela principal.)

`/filter` Dispara uma ou mais etapas do processamento. (Chamada pelo botão *Run* da tela de detalhamento.)

3.3 Fluxo de Processamento

A Figura 2 apresenta o fluxo de processamento da ferramenta KIF-QA, incluindo os passos utilizados internamente pelo *framework* KIF-QA para produzir a *stream* de *statements* que responde a pergunta de entrada (cf. [8] para mais detalhes).

Passo 1. O processamento começa com a submissão de uma pergunta simples, por exemplo, "Onde nasceu James Brown?", através da interface de usuário.

Passo 2. A interface então submete o texto da pergunta ao *back-end* usando a rota `/query`, no caso de uma execução fim-a-fim (*Ask*, tela principal), ou `/filter`, no caso de execução customizada (*Run*, tela de detalhamento). De agora em diante, vamos assumir que se trata de uma execução fim-a-fim (*Ask*).

Passo 3. Ao receber uma pergunta simples em linguagem natural, *framework* KIF-QA usa aprendizagem em contexto sobre o LLM

[6]https://www.ibm.com/products/watsonx
[7]https://openai.com/
[8]https://flask.palletsprojects.com/

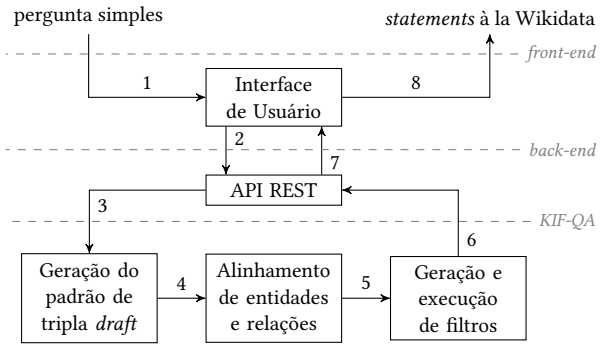

Figura 2: Fluxo de processamento da ferramenta KIF-QA.

configurado para gerar um esboço (*draft*) de um padrão de tripla (*triple pattern*) que representa a pergunta. Por exemplo, para a pergunta "Onde nasceu James Brown?", um esboço de padrão de tripla possível é

("James Brown", "local de nascimento", "?x")

em que "James Brown" é o sujeito, "local de nascimento" é o predicado, e "?x" é o objeto desconhecido (i.e., a variável do padrão).

Passo 4. Uma vez gerado o esboço de padrão de tripla, o *framework* KIF-QA usa os termos que ocorrem no padrão para resolver as entidades e relações as quais os termos se referem. Esse tipo de resolução é conhecido como alinhamento (*linking*) e no KIF-QA é feito em dois sub-passos:

(a) O rótulo do sujeito ou objeto conhecido, no nosso exemplo, "James Brown", é usado como chave para buscar por entidades com rótulos ou descrições similares nos grafos alvo. Isso é feito através de *searchers* KIF. O resultado é uma lista de entidades candidatas ranqueadas por similaridade. Essa lista é então verbalizada e submetida ao LLM para desambiguação, i.e., para ser refinada de forma que apenas candidatos relevantes para o contexto da pergunta sejam mantidos.
(b) Para cada entidade candidata E da lista refinada anterior, o *framework* usa *stores* KIF para buscar nos grafos alvo por arestas com origem ou destino em E. A lista de relações que ocorrem nessas arestas (juntamente com seus rótulos e descrições) é então verbalizada e submetida para o LLM para desambiguação. O resultado é, para cada entidade candidata E, uma lista de relações R candidatas.

Passos 5 e 6. As entidades e relações candidatas são usadas para construir filtros KIF correspondentes. Por exemplo, se E é o item wd:Q5950 (James Brown, o músico americano) e R é a propriedade wd:P19 (local de nascimento), e se E é origem de alguma aresta R no grafo, então é gerado o filtro KIF

```
filter(wd.Q(5950),wd.P(19))
```

que busca por *statements* com sujeito E, propriedade R e qualquer valor. Todos os filtros gerados são executados em paralelo e a *stream* de *statements* resultante é enviada ao próximo passo.

Passos 7 e 8. Os *statements* recebidos do *framework* KIF-QA são serializados num formato apropriado e enviados de volta à interface de usuário para exibição.

4 CONCLUSÃO

Este trabalho apresentou a ferramenta Web KIF-QA para respostas a perguntas simples em linguagem natural sobre grafos de conhecimento heterogêneos. A ferramenta usa aprendizagem em contexto sobre LLMs de prateleira para análise semântica e o *framework* KIF para mediar todo acesso aos grafos subjacentes.

Em comparação a interfaces similares para KBQA, um dos diferencias da ferramenta KIF-QA é possibilitar que o usuário visualize e edite os artefatos intermediários usados para processar a pergunta e gerar a resposta. Isso permite, por exemplo, que erros cometidos pelo modelo de linguagem sejam corrigidos interativamente.

Atualmente a ferramenta lida apenas com perguntas simples. Como trabalho futuro, estamos estudando a extensão da abordagem KIF-QA para tratar perguntas complexas, cuja resposta envolve múltiplas arestas do grafo. Uma forma de realizar essa extensão é através do particionamento de perguntas complexas em conjuntos de perguntas mais simples. Há na literatura trabalhos que utilizam essa técnica [3] e experimentos preliminares indicam que é possível fazê-lo usando LLMs via aprendizagem em contexto.

Ainda como trabalho futuro, pretendemos adicionar ao *front-end* da ferramenta suporte à visualização da reposta como um grafo, i.e., exibir os *statements* da *stream* de resposta como arestas formando um "grafo de resposta".

REFERÊNCIAS

[1] C. Badenes-Olmedo and O. Corcho. 2024. MuHeQA: Zero-shot question answering over multiple and heterogeneous knowledge bases. *Semant. Web* 15, 5 (2024), 1547–1561.
[2] J. Berant, A. Chou, R. Frostig, and P. Liang. 2013. Semantic Parsing on Freebase from Question-Answer Pairs. In *Proc. 2013 Conf. Empirical Methods in Natural Language Processing*. ACL, 1533–1544.
[3] X. Huang, S. Cheng, Y. Shu, Y. Bao, and Y. Qu. 2023. Question Decomposition Tree for Answering Complex Questions over Knowledge Bases. In *Proc. AAAI Conf. Artificial Intelligence*, Vol. 37. 12924–12932.
[4] S. Kim, J. Chen, T. Cheng, A. Gindulyte, J. He, S. He, Q. Li, B .A. Shoemaker, P. A. Thiessen, B. Yu, L. Zaslavsky, J. Zhang, and E. E. Bolton. 2023. PubChem 2023 Update. *Nucleic Acids Res.* 51, D1 (October 2023), D1373–D1380.
[5] Y. Lan, G. He, J. Jiang, J. Jiang, W. X. Zhao, and J.-R. Wen. 2023. Complex Knowledge Base Question Answering: A Survey. *IEEE Trans. Know. Data Eng.* 35, 11 (2023), 11196–11215.
[6] 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. 2015. DBpedia - A large-scale, multilingual knowledge base extracted from Wikipedia. *Semant. Web* 6 (2015), 167–195.
[7] G. Lima, J. M. B. Rodrigues, M. Machado, E. Soares, S. R. Fiorini, R. Thiago, L. G. Azevedo, V. T. da Silva, and R. Cerqueira. 2024. KIF: A Wikidata-Based Framework for Integrating Heterogeneous Knowledge Sources. arXiv:2403.10304
[8] M. Machado, J. P. P. Campos, G. Lima, and V. T. da Silva. 2025. KIF-QA: Using Off-the-shelf LLMs to Answer Simple Questions over Heterogeneous Knowledge Bases. In *Joint Proc. 5th Wikidata Workshop for the Scientific Wikidata Community co-located with the 24th International Semantic Web Conference (ISWC 2025), Nara, Japan, November 2–6, 2025.*
[9] A. Razzhigaev, M. Salnikov, V. Malykh, P. Braslavski, and A. Panchenko. 2023. A System for Answering Simple Questions in Multiple Languages. In *Proc. 61st Annual Meeting of ACL (Volume 3: System Demonstrations)*. ACL, 524–537.
[10] D. Vrandečić and M. Krötzsch. 2014. Wikidata: A Free Collaborative Knowledgebase. *Commun. ACM* 57, 10 (September 2014), 78–85.
[11] W3C RDF Working Group. 2014. *Resource Description Framework (RDF): Concepts and Abstract Syntax.* W3C Recommendation. W3C.
[12] W3C SPARQL Working Group. 2013. *SPARQL 1.1 Overview.* W3C Recommendation. W3C.