Pokenator

Jogo de Adivinhação com Web Semântica

Explorando RDF, OWL e SPARQL através de Pokémon

Rávilon Aguiar Dos Santos e Gustavo Pinzon Pereira

22 de Fevereiro de 2026

Resumo e Objetivos

- **Introdução:** Inspirado no Akinator, Pokenator usa a Web Semântica para adivinhar Pokémon por perguntas Sim/Não.
- **Objetivo:** Tornar conceitos de RDF, OWL e SPARQL acessíveis e divertidos, aproximando teoria de prática.
- **Agenda:** Ontologia & Dados, Arquitetura, Heurística, Implementação, Avaliação, Conclusão.

Fundamentos da Web Semântica

RDF

Grafo de triplas (sujeito–predicado–objeto) para representar dados.

OWL

Ontologias definem classes e propriedades que descrevem o mundo.

SPARQL

Linguagem de consulta para consultar grafos com filtros e padrões.

Ontologia e Fontes de Dados

Classes:
Pokémon, Tipo, Habilidade, Habitat, Geração

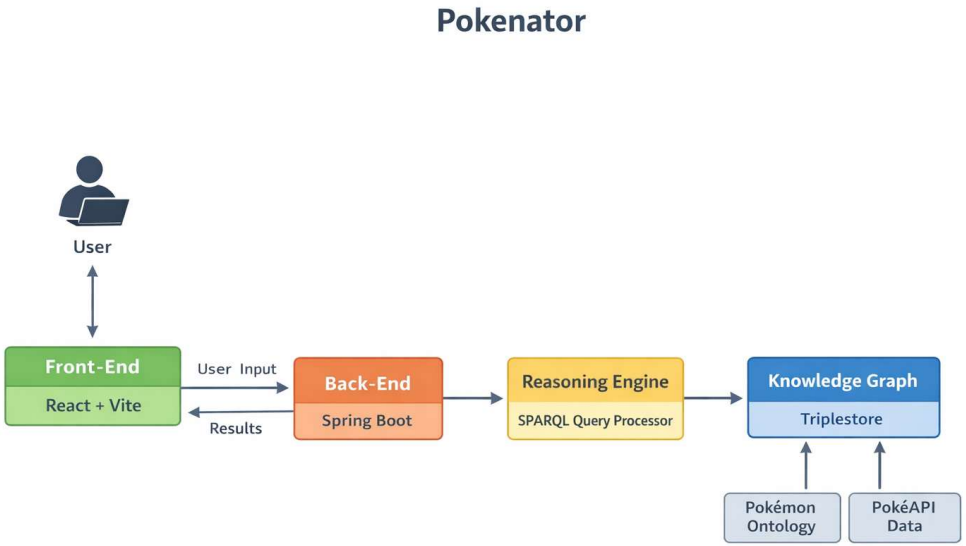
Propriedades:
hasType, hasAbility, fromGeneration

Raciocínio: inferência de duplo tipo e ligações implícitas.

Métrica	Valor
Entidades	18 568
Triplos	129 834
Fontes	Bulbagarden, PokeWiki, PokéAPI, PokemonDB

Os dados são carregados num triplestore Jena Fuseki com raciocínio RDFS/OWL.

Arquitetura do Sistema



Frontend:
React, Vite e Tailwind fornecem a interface do jogo.

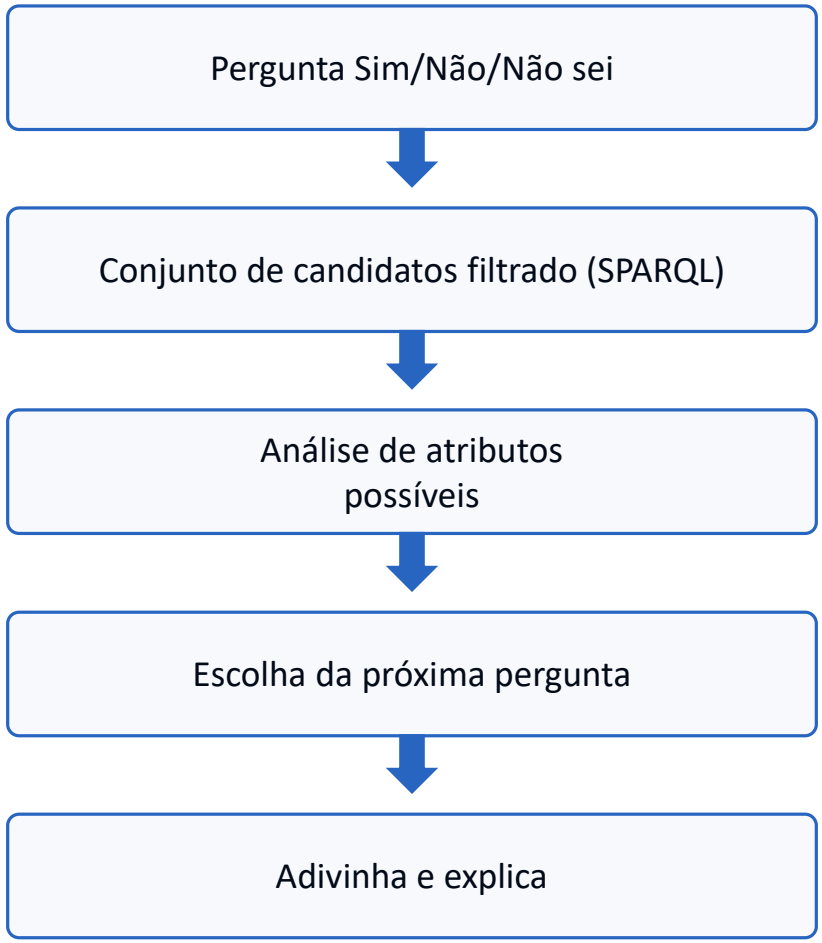
Backend:
Spring Boot traduz respostas para SPARQL e comunica com o triplestore.

Motor de raciocínio:
Implementado em Java usando Jena, seleciona a próxima pergunta via heurísticas.

Base de Conhecimento:
PokemonKG em Fuseki com raciocínio RDFS/OWL.

Mecânica do Jogo e Heurística

Heurística de seleção:
• Mantém o conjunto de candidatos e filtra com SPARQL.
• Analisa propriedades (tipo, habilidade, habitat, geração, etc.).
• Escolhe pergunta que divide melhor os candidatos; aleatoriza entre as melhores para evitar determinismo.
• Modo de adivinhação quando sobram poucos candidatos.



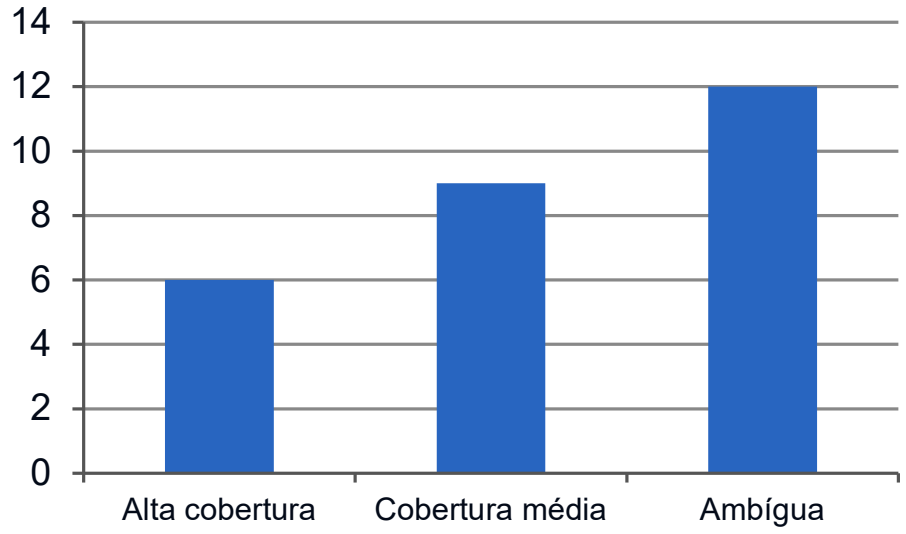
Detalhes de Implementação

- **Pré-processamento de dados:** Carrega a ontologia em Fuseki, agrega dados das APIs e atribui IRIs persistentes; mapeia atributos ausentes.
- **Construção de consultas SPARQL:** Combina padrões positivos e negativos; usa FILTER NOT EXISTS para respostas negativas.
- **Explicações:** O sistema exibe a consulta SPARQL e os triplos que justificam a adivinhação.

Avaliação e Resultados

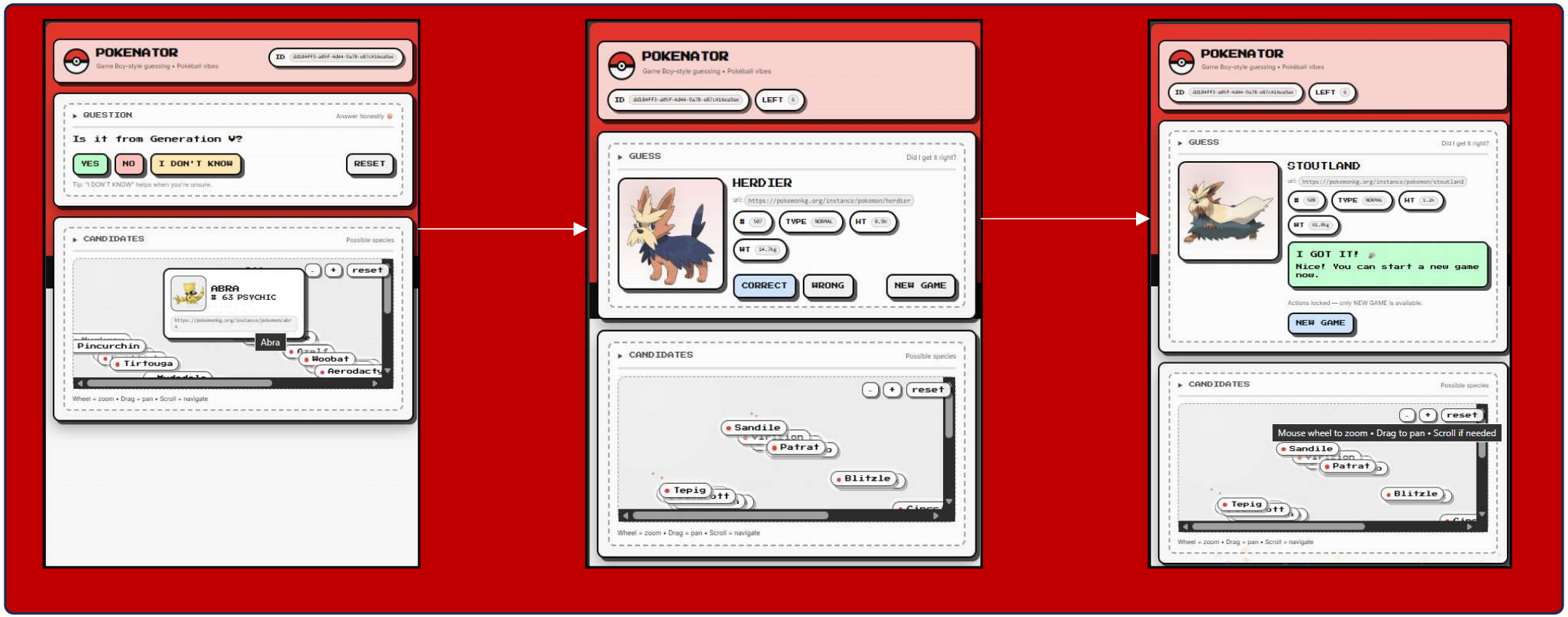
• **Eficiência:** O sistema converge para um único Pokémon em 5–7 perguntas quando há atributos discriminativos.

• **Desafios:** Cobertura incompleta ou atributos ausentes exigem mais perguntas; heterogeneidade dos dados.



Média de perguntas por cobertura

Conclusões e Futuro



Conclusões e Futuro

- **Demonstração pedagógica:** Pokenator mostra como a Web Semântica pode sustentar aplicações interativas e educativas.
- **Aprendizagem intuitiva:** Integra conceitos complexos de RDF, OWL e SPARQL em um jogo divertido e familiar.
- **Recursos abertos:** Promove o uso de ontologias, PokéAPI e outros dados abertos para exploração.
- **Futuro:** Expandir a ontologia, integrar mais conjuntos de dados e aplicar heurísticas mais robustas em outros domínios.

Recursos & Agradecimentos

- **Artigo no Overleaf:** Acesse e edite o artigo no Overleaf

- **Código fonte:**
Repositório GitHub do Pokenator

- **Fontes de dados:**
Bulbagarden, PokeWiki, PokéAPI e PokemonDB.

Agradecimentos: A todos os desenvolvedores da comunidade open-source e aos criadores de dados abertos que tornam projetos como este possíveis.