Querying Enterprise Knowledge Graph With Natural Language

Junyi Chai*, Yonggang Deng*, Maochen Guan*, Yujie He*, Bing Li*, and Rui Yan *

Microsoft AI & R, Bellevue, WA, USA
{juchai, yoden, maguan, yujh, libi, ruya}@microsoft.com

Conversational interface to enterprise knowledge graph with large amount of data has become popular recently for AI powered applications. Natural language understanding (NLU) techniques not only provide user-friendly interaction, but also greatly boost productivity by eliminating the need of learning structured query languages (such as SPARQL) required to access knowledge databases. We present Yugen as a first step to conversational AI that answers user queries using knowledge graph in the enterprise domain. Yugen is a deep learning based NLU & QA enginecurrently and serving the product of Microsoft Enterprise Graph[1] on Azure cloud. The development of Yugen has tackled many enterprise domain challenges, such as data cold start problem, detecting domain-specific query intents, key entities and their relationships, as well as generating correct graph queries and restating the query results in natural language back to users.

1 System Architecture

Model Training. We firstly implement a training data generation pipeline that overcomes cold start issue under compliance, which is a common challenge among enterprise AI powered applications. This pipeline requires minimum human annotation and is highly customizable to allow enterprise users to train NLU models with domain-specific data. Yugen supports online training, where user-defined intents and entity types can be used to train new NLU models to suit changing business scenarios easily. Our NLU system consists of an attention-based seq2seq deep learning model with 1 embedding layer and 2-layer Bi-LSTM (each layer has 8 multi-head attention mechanisms). This model architecture can be extended easily to adapt to different training tasks. E.g. adding a CRF layer for entity detection model, or a softmax layer for an intent detection model. Online trained models can be 1-click deployed to serve production user queries.

Online Serving. The input query is pre-processed first. This includes casing/punctuation reconstruction, tokenization, POS tagging, dependency parsing, date/time recognition. This pre-processed result is then sent to two subsequent components: intent detection and entity mention detection (EMD).

* All authors contribute to this work equally.

[1] https://docs.microsoft.com/en-us/enterprise-graph/enterprise-graph-overview

Intent detection recognizes user intents, which usually covers a wide spectrum, ranging from finding people to scheduling meetings. Each intent maps to a specific class in the ontology. E.g. "Find me quantum computing experts who live in Redmond." has the "expert" intent with a type of "Employee".

EMD detects key entity mentions and types in the query. Yugen's EMD detects more entities compared to the traditional NER. It can recognize any entities in the knowledge graph by distinguishing concepts and instances. In the above query example, EMD can detect "experts" as a concept entity with the type of "Employee" and "quantum computing" as an instance entity with the type of "Expertise", while the traditional NER will not be able to do so.

The aggregated outputs from intent detection and EMD will then be passed into relation mention detection (RMD). RMD detects the relations between intent and entities, then constructs a graph structure representing Yugen's understanding of the original search query. RMD considers both the ontology and the query's dependency parse tree for relation detection. In the above example, RMD detects a two-hop relationship (hasExpertise $\rightarrow$ hasExpertiseName) for "experts" and "quantum computing", then forms a graph representation as "expert $\xrightarrow{\text{hasExpertise}}$ skill $\xrightarrow{\text{hasExpertiseName}}$ quantum computing". RMD also conducts relation disambiguation if multiple relations exist using a language model. E.g. in the above query, RMD detects both "expert $\xrightarrow{\text{livesIn}}$ Redmond" and "expert $\xrightarrow{\text{worksIn}}$ Redmond", but ultimately will choose the former.

Structured query generation (SQG) then takes the outputs from above components to generate queries, then queries knowledge bases and gets results. Yugen's architecture embraces diverse types of knowledge database by design, thus can generate various structured queries in parallel on the fly that are applicable to knowledge graphs indexed differently (e.g. graph index vs. inverted index). In the current implementation, SQG supports both SPARQL and multi-field search queries.

The natural language restatement component finally wraps up all of the Yugen's natural language understanding and query results, and returns users a human-readable answer. This restatement is not just a simple echo of user's query but an explanation with all details about how the query is understood by Yugen, so that users can easily evaluate the correctness of the answer and each step in the pipeline.

2 Business Value

Yugen provides enterprise users an end-to-end speech-based NLU & QA engine for enterprise knowledge graph. It is customizable, fast, accurate and robust. It benefits enterprise users by lowering the cost of training employees to learn specified query languages, understanding enterprise domain queries, providing query answers, and explaining the results with natural language response, ultimately adding massive value to the business. Yugen has intention detection F1 score of 98.42% and EMD F1 score of 96.96%. Our customer - Publicis Groupe is now leveraging Yugen as their main AI-enabled interface to their knowledge graph.