Knowledge bases for Amazon Bedrock

Knowledge bases for Amazon Bedrock provides you the capability of amassing data sources into a repository of information. With knowledge bases, you can easily build an application that takes advantage of retrieval augmented generation (RAG), a technique in which the retrieval of information from data sources augments the generation of model responses. Once set up, you can take advantage of a knowledge base in the following ways.

Configure your RAG application to use the RetrieveAndGenerate API to query your knowledge base and generate responses from the information it retrieves.

Associate your knowledge base with an agent (for more information, see Agents for Amazon Bedrock) to add RAG capability to the agent by helping it reason through the steps it can take to help end users.

Create a custom orchestration flow in your application by using the Retrieve API to retrieve information directly from the knowledge base.

A knowledge base can be used not only to answer user queries, but also to augment prompts provided to foundation models by providing context to the prompt. Knowledge base responses also come with citations, such that users can find further information by looking up the exact text that a response is based on and also check that the response makes sense and is factually correct.

You take the following steps to set up and use your knowledge base.

1. Configure the data sources to add to your knowledge base.

2. Upload your data to an Amazon S3 bucket.

3. Ingest your data by generating embeddings with a foundation model and storing them in a supported vector store.

4. Set up your application or agent to query the knowledge base and return augmented responses.

How it works

Knowledge bases for Amazon Bedrock help you take advantage of Retrieval Augmented Generation (RAG), a popular technique that involves drawing information from a data store to augment the responses generated by Large Language Models (LLMs). When you set up a knowledge base with your data sources, your application can query the knowledge base to return information to answer the query either with direct quotations from sources or with natural responses generated from the query results.

With knowledge bases, you can build applications that are enriched by the context that is received from querying a knowledge base. It enables a faster time to market by abstracting from the heavy lifting of building pipelines and providing you an out-of-the-box RAG solution

to reduce the build time for your application. Adding a knowledge base also increases cost-effectiveness by removing the need to continually train your model to be able to leverage your private data.

Pre-processing data

To enable effective retrieval from private data, a common practice is to first split the documents into manageable chunks for efficient retrieval. The chunks are then converted to embeddings and written to a vector index, while maintaining a mapping to the original document. These embeddings are used to determine semantic similarity between queries and text from the data sources. The following image illustrates pre-processing of data for the vector database.

Runtime execution

At runtime, an embedding model is used to convert the user's query to a vector. The vector index is then queried to find chunks that are semantically similar to the user's query by comparing document vectors to the user query vector. In the final step, the user prompt is augmented with the additional context from the chunks that are retrieved from the vector index. The prompt alongside the additional context is then sent to the model to generate a response for the user. The following image illustrates how RAG operates at runtime to augment responses to user queries.