Contextual AI Reranker

Copy page

Contextual AI's Instruction-Following Reranker is the world's first reranker designed to follow custom instructions about how to prioritize documents based on specific criteria like recency, source, and metadata. With superior performance on the BEIR benchmark (scoring 61.2 and outperforming competitors by significant margins), it delivers unprecedented control and accuracy for enterprise RAG applications.

Key capabilities

- **Instruction Following**: Dynamically control document ranking through natural language commands
- **Conflict Resolution**: Intelligently handle contradictory information from multiple knowledge sources
- **Superior Accuracy**: Achieve state-of-the-art performance on industry benchmarks
- **Seamless Integration**: Drop-in replacement for existing rerankers in your RAG pipeline

The reranker excels at resolving real-world challenges in enterprise knowledge bases, such as prioritizing recent documents over outdated ones or favoring internal documentation over external sources.

To learn more about our instruction-following reranker and see examples of it in action, visit our **product overview**.

For comprehensive documentation on Contextual AI's products, please visit our **developer portal**.

ntegration requires the `contextual-client` Python SDK. Learn more about it

LangChain Docs

Contextual AI Reranker

This integration invokes Contextual AI's Grounded Language Model.

Integration details

Class	Package	Local	Serializable	JS sup
ContextualRerank	langchain-contextual	❌	beta	❌

Setup

To access Contextual's reranker models you'll need to create a/an Contextual AI account, get an API key, and install the `langchain-contextual` integration package.

Credentials

Head to app.contextual.ai to sign up to Contextual and generate an API key. Once you've done this set the CONTEXTUAL_AI_API_KEY environment variable:

```
import getpass
import os

if not os.getenv("CONTEXTUAL_AI_API_KEY"):
    os.environ["CONTEXTUAL_AI_API_KEY"] = getpass.getpass(
        "Enter your Contextual API key: "
    )
```

Installation

LangChain Docs

Contextual AI Reranker

LangChain Docs

Contextual AI Reranker

LangChain Docs

Contextual AI Reranker

```
pip install -qU langchain-contextual
```

Instantiation

The Contextual Reranker arguments are:

Parameter	Type	Description
documents	list[Document]	A sequence of documents to rerank. Any metadata contained in the documents will also be used for reranking.
query	str	The query to use for reranking.
model	str	The version of the reranker to use. Currently, we just have “ctxl-rerank-en-v1-instruct”.
top_n	Optional[int]	The number of results to return. If None returns all results. Defaults to self.top_n.
instruction	Optional[str]	The instruction to be used for the reranker.
callbacks	Optional[Callbacks]	Callbacks to run during the compression process.

```
from langchain_contextual import ContextualRerank

api_key = ""
model = "ctxl-rerank-en-v1-instruct"

compressor = ContextualRerank(
    model=model,
    api_key=api_key,
)
```

LangChain Docs

Contextual AI Reranker

```
query = "What is the current enterprise pricing for the RTX 5090 GPU for bulk
instruction = "Prioritize internal sales documents over market analysis repor

document_contents = [
    "Following detailed cost analysis and market research, we have implemente
    "Enterprise pricing for the RTX 5090 GPU bulk orders (100+ units) is curr
    "RTX 5090 Enterprise GPU requires 450W TDP and 20% cooling overhead.",
]

metadata = [
    {
        "Date": "January 15, 2025",
        "Source": "NVIDIA Enterprise Sales Portal",
        "Classification": "Internal Use Only",
    },
    {"Date": "11/30/2023", "Source": "TechAnalytics Research Group"},
    {
        "Date": "January 25, 2025",
        "Source": "NVIDIA Enterprise Sales Portal",
        "Classification": "Internal Use Only",
    },
]

documents = [
    Document(page_content=content, metadata=metadata[i])
    for i, content in enumerate(document_contents)
]
reranked_documents = compressor.compress_documents(
    query=query,
    instruction=instruction,
    documents=documents,
)
```

within a chain

API reference

For detailed documentation of all ChatContextual features and configurations head to the GitHub page: **github.com/ContextualAI//langchain-contextual**

LangChain Docs

Contextual AI Reranker

LangChain Docs

Contextual AI Reranker