Skills Network

LangChain Documents for Building RAG Applications

IBM

LangChain Documents for Building RAG Applications

What you will learn

Explore LangChain tool documents to build RAG applications

Explain the components of documents

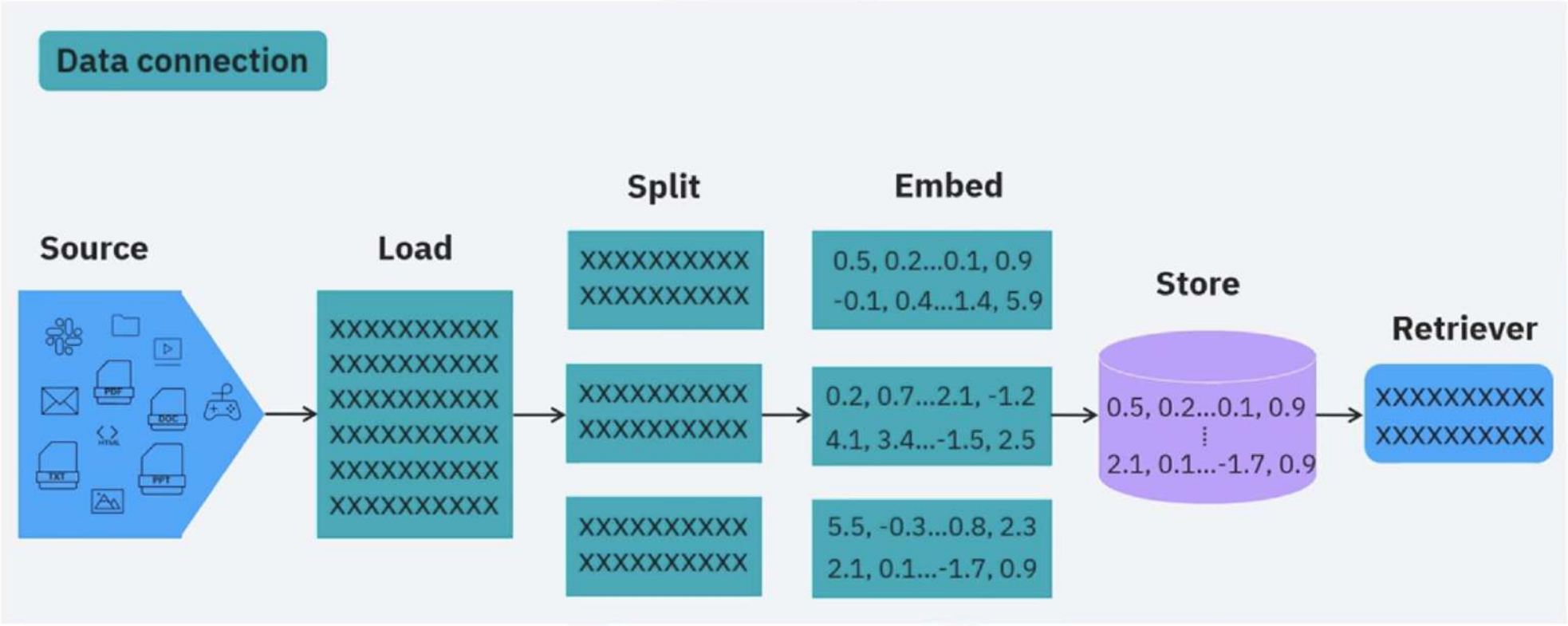
Introduction to building applications

Rely on AI technologies

LangChain simplifies the application building process

- Provides an easy-to-use interface

Documents

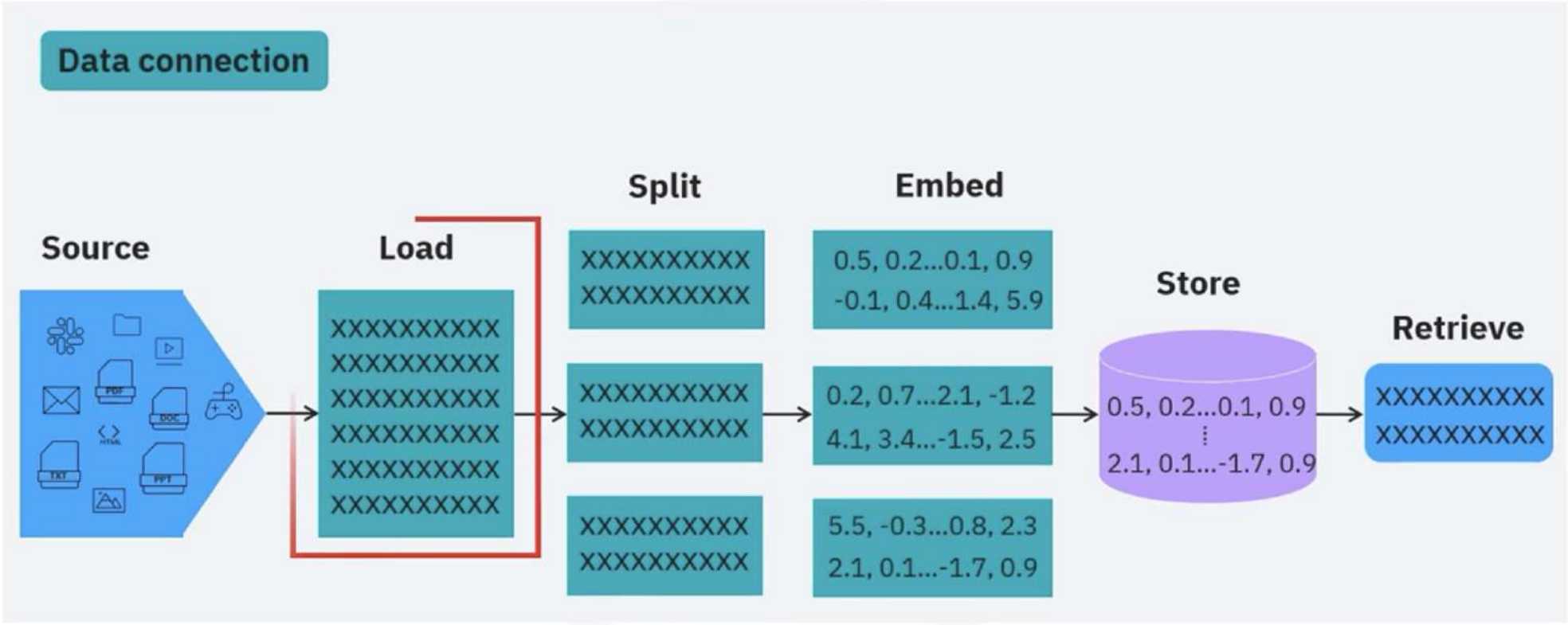


** Source: https://python.langchain.com/v0.1/docs/modules/data_connection/*

Documents source

```
Document(page_content="""Python is an interpreted high-level general-
    purpose programming language.
    Python's design philosophy emphasizes code readability with its notable use of significant indentation.""",
        metadata={
            'my_document_id' : 234234,
            'my_document_source' : "About Python",
            'my_document_create_time' : 1680013019
})
```

Document loader



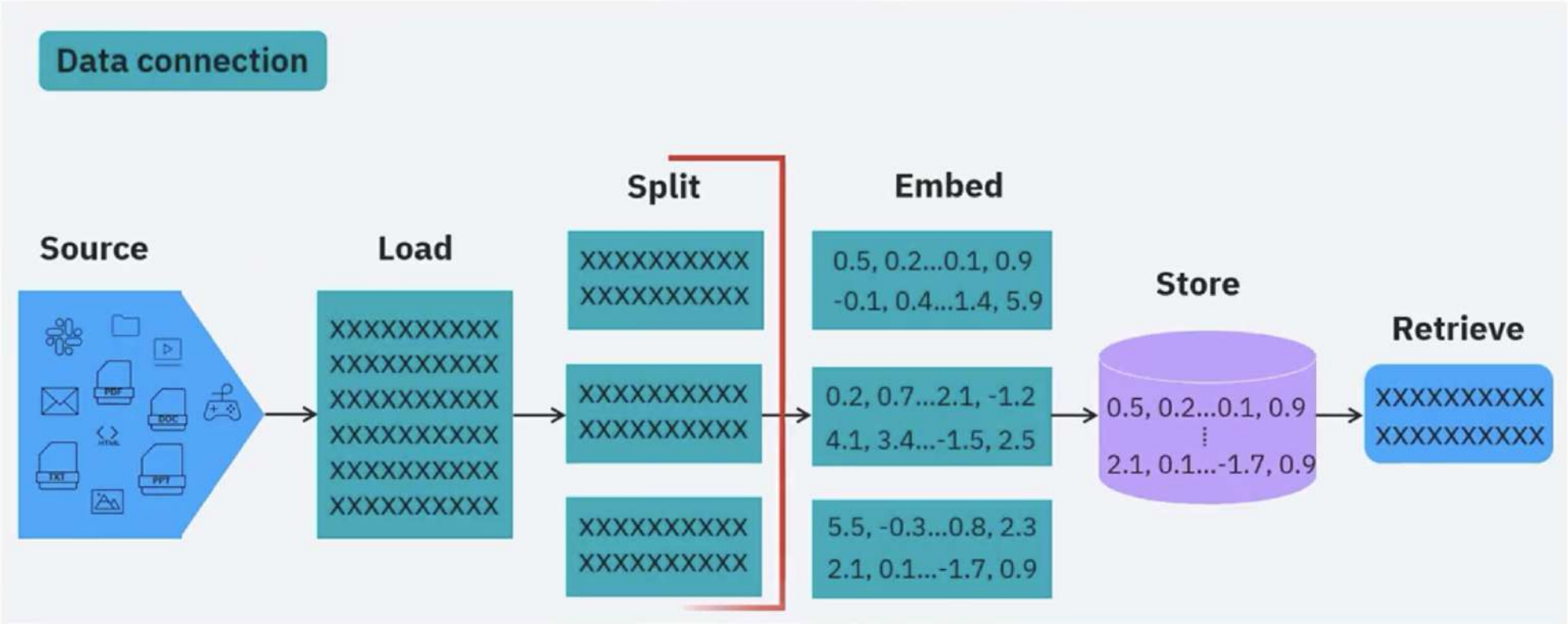
Source:
https://python.langchain.com/v0.1/docs/modules/data_connection/

Document loader

```
loader = WebBaseLoader("https://python.langchain.com/v0.2/docs/introduction/")

web_data = loader.load()
```

Document transformation



Source:
https://python.langchain.com/v0.1/docs/modules/data_connection/

Document transformation: Text splitter

```
text_splitter = CharacterTextSplitter(chunk_size=200, chunk_overlap=20,
        separator="\n")

chunks = text_splitter.split_documents(document)

print(len(chunks))
```

Document transformation: Text splitter

```
text_splitter = CharacterTextSplitter(chunk_size=200, chunk_overlap=20,
        separator="\n")

chunks = text_splitter.split_documents(document)

print(len(chunks))
```

```
148
```

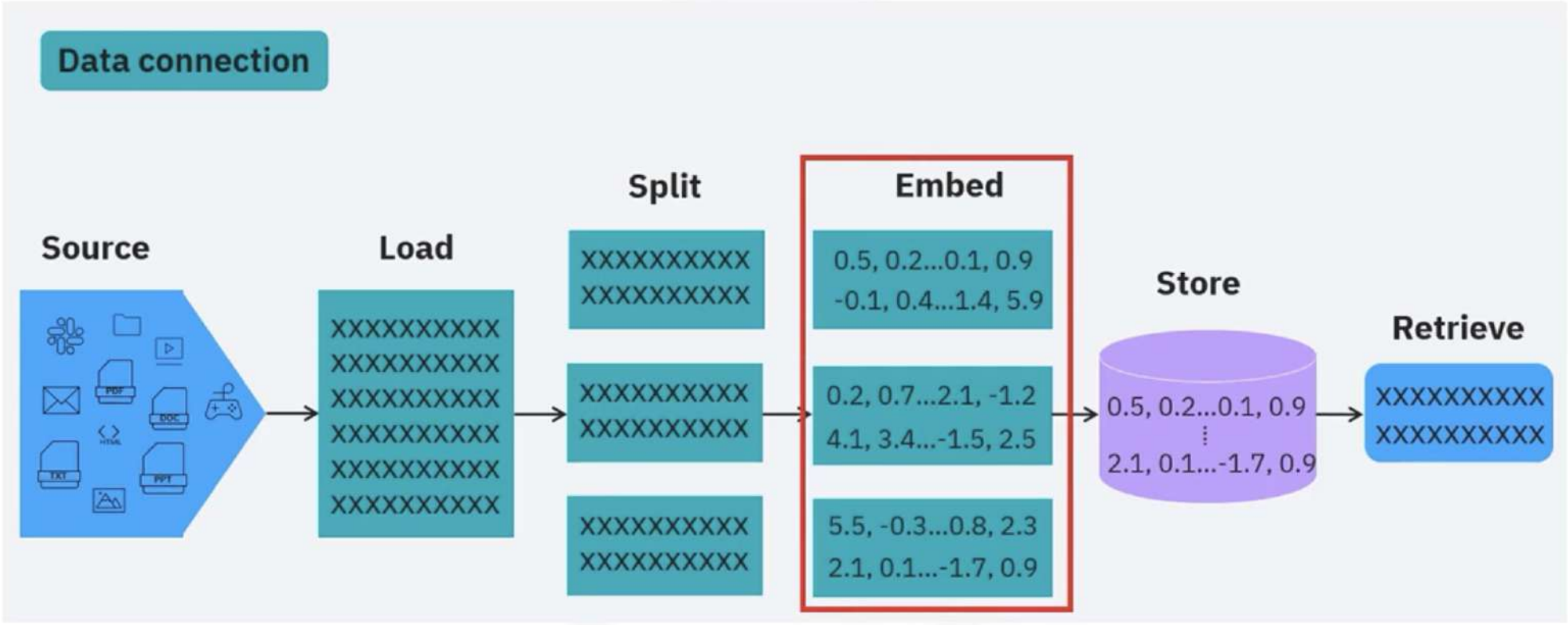
Document embeddings

```
from ibm_watsonx_ai.metanames import EmbedTextParamsMetaNames
from langchain_ibm import WatsonxEmbeddings

embed_params = {
        EmbedTextParamsMetaNames.TRUNCATE_INPUT_TOKENS: 3,
        EmbedTextParamsMetaNames.RETURN_OPTIONS: {"input_text": True},
}


watsonx_embedding = WatsonxEmbeddings(
        model_id="ibm/slate-125m-english-rtrvr",
        url="https://us-south.ml.cloud.ibm.com",
        project_id="skills-network",
        params=embed_params,
```

Document embeddings



Source:
https://python.langchain.com/v0.1/docs/modules/data_connection/

Document embeddings

```
from ibm_watsonx_ai.metanames import EmbedTextParamsMetaNames
from langchain_ibm import WatsonxEmbeddings

embed_params = {
        EmbedTextParamsMetaNames.TRUNCATE_INPUT_TOKENS: 3,
        EmbedTextParamsMetaNames.RETURN_OPTIONS: {"input_text": True},
}


watsonx_embedding = WatsonxEmbeddings(
        model_id="ibm/slate-125m-english-rtrvr",
        url="https://us-south.ml.cloud.ibm.com",
        project_id="skills-network",
        params=embed_params,
```

Document embeddings

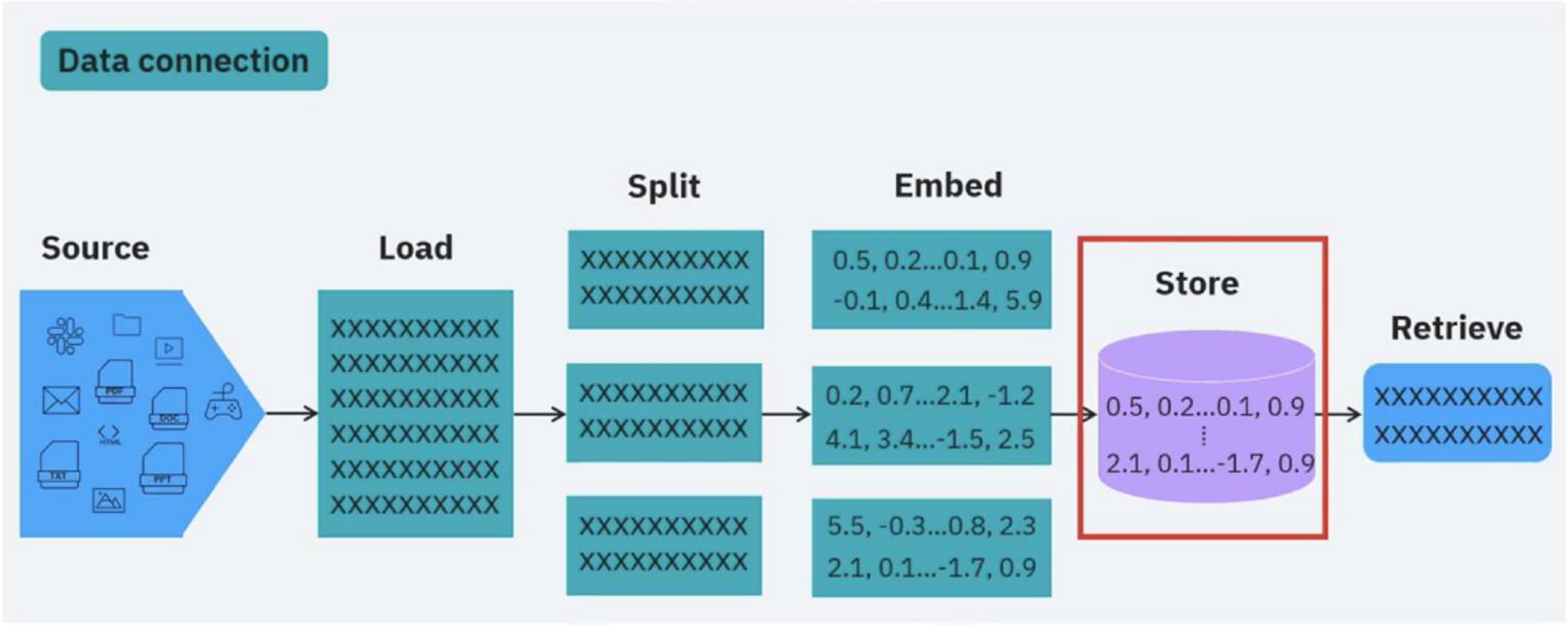
```
}

watsonx_embedding = WatsonxEmbeddings(
        model_id="ibm/slate-125m-english-rtrvr",
        url="https://us-south.ml.cloud.ibm.com",
        project_id="skills-network",
        params=embed_params,
)

texts = [text.page_content for text in chunks]

embedding_result = watsonx_embedding.embed_documents(texts)
```

Document storage



Source:
https://python.langchain.com/v0.1/docs/modules/data_connection/

Document storage: Vector database

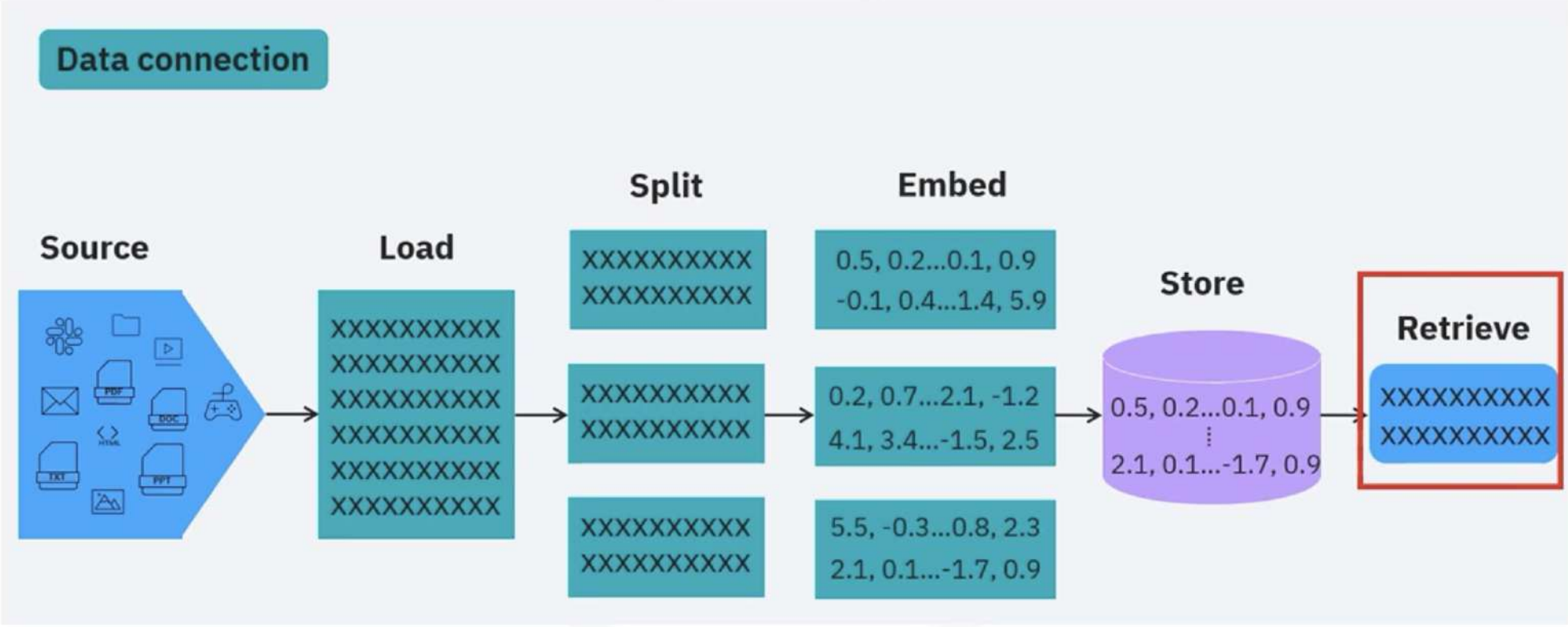
```
from langchain.vectorstores import Chroma

docsearch = Chroma.from_documents(chunks, watsonx_embedding)

query = "Langchain"
docs = docsearch.similarity_search(query)
print(docs[0].page_content)
```

LangChain helps us to unlock the ability to harness the LLM's immense potential in tasks such as document analysis, chatbot development, code analysis, and countless other

Document retriever



Source:
https://python.langchain.com/v0.1/docs/modules/data_connection/

Document retriever

```
retriever = docsearch.as_retriever()

docs = retriever.invoke("Langchain")

docs[0].page_content
```

LangChain helps us to unlock the ability to harness the LLM's immense potential in tasks such as document analysis, chatbot development, code analysis, and countless other

Recap

- LangChain facilitates comprehensive tools for RAG applications
- Document object in LangChain serves as a container for data information
- LangChain document loader handles various document types
- LangChain in document retrieves relevant isolated sections from the documents by splitting them into manageable pieces
- Embedding models embed documents and LangChain facilitates various retrievers