Backed by a Vector Store

`VectorStoreRetrieverMemory` stores memories in a vector store and queries the top-K most "salient" docs every time it is called.

This differs from most of the other Memory classes in that it doesn't explicitly track the order of interactions.

In this case, the "docs" are previous conversation snippets. This can be useful to refer to relevant pieces of information that the AI was told earlier in the conversation.

```
from datetime import datetime
from langchain_openai import OpenAIEmbeddings
from langchain_openai import OpenAI
from langchain.memory import VectorStoreRetrieverMemory
from langchain.chains import ConversationChain
from langchain.prompts import PromptTemplate
```

Initialize your vector store

Depending on the store you choose, this step may look different. Consult the relevant vector store documentation for more details.

```
import faiss

from langchain.docstore import InMemoryDocstore
from langchain_community.vectorstores import FAISS


embedding_size = 1536 # Dimensions of the OpenAIEmbeddings
index = faiss.IndexFlatL2(embedding_size)
embedding_fn = OpenAIEmbeddings().embed_query
vectorstore = FAISS(embedding_fn, index, InMemoryDocstore({}), {})
```

Create your `VectorStoreRetrieverMemory`

The memory object is instantiated from any vector store retriever.

```
# In actual usage, you would set `k` to be a higher value, but we use k=1 to show that
# the vector lookup still returns the semantically relevant information
```

```
retriever =
vectorstore.as_retriever(search_kwargs=dict(k=1
memory =
VectorStoreRetrieverMemory(retriever=retriever)

# When added to an agent, the memory object can
save pertinent information from conversations o
used tools
memory.save_context({"input": "My favorite food
is pizza"}, {"output": "that's good to know"})
memory.save_context({"input": "My favorite spor
is soccer"}, {"output": "..."})
memory.save_context({"input": "I don't the
Celtics"}, {"output": "ok"}) #
```

```
print(memory.load_memory_variables({"prompt":
"what sport should i watch?"})["history"])
```

```
    input: My favorite sport is soccer
    output: ...
```

Using in a chain

Let's walk through an example, again setting `verbose=True` so we can see the prompt.

```
llm = OpenAI(temperature=0) # Can be any
valid LLM
_DEFAULT_TEMPLATE = """The following is a
friendly conversation between a human and an
AI. The AI is talkative and provides lots of
specific details from its context. If the AI
does not know the answer to a question, it
truthfully says it does not know.

Relevant pieces of previous conversation:
{history}

(You do not need to use these pieces of
information if not relevant)

Current conversation:
Human: {input}
AI:"""
PROMPT = PromptTemplate(
    input_variables=["history", "input"],
template=_DEFAULT_TEMPLATE
)
conversation_with_summary =
ConversationChain(
    llm=llm,
    prompt=PROMPT,
    memory=memory,
    verbose=True
)
```

```
conversation_with_summary.predict(input="Hi,
my name is Perry, what's up?")
```

```
    > Entering new ConversationChain chain...
    Prompt after formatting:
    The following is a friendly conversation
between a human and an AI. The AI is
talkative and provides lots of specific
details from its context. If the AI does not
know the answer to a question, it truthfully
says it does not know.

    Relevant pieces of previous conversation:
    input: My favorite food is pizza
    output: that's good to know

    (You do not need to use these pieces of
information if not relevant)

    Current conversation:
    Human: Hi, my name is Perry, what's up?
    AI:

    > Finished chain.
    " Hi Perry, I'm doing well. How about
you?"
```

```
# Here, the basketball related content is
surfaced
conversation_with_summary.predict(input="what's
my favorite sport?")
```

```
    > Entering new ConversationChain chain...
    Prompt after formatting:
    The following is a friendly conversation
between a human and an AI. The AI is
talkative and provides lots of specific
details from its context. If the AI does not
know the answer to a question, it truthfully
says it does not know.

    Relevant pieces of previous conversation:
    input: My favorite sport is soccer
    output: ...

    (You do not need to use these pieces of
information if not relevant)

    Current conversation:
    Human: what's my favorite sport?
    AI:

    > Finished chain.






    ' You told me earlier that your favorite
sport is soccer.'
```

```
# Even though the language model is stateless,
since relevant memory is fetched, it can
"reason" about the time.
# Timestamping memories and data is useful in
general to let the agent determine temporal
relevance
conversation_with_summary.predict(input="Whats
my favorite food")
```

```



    > Entering new ConversationChain chain...
    Prompt after formatting:
    The following is a friendly conversation
between a human and an AI. The AI is
talkative and provides lots of specific
details from its context. If the AI does not
know the answer to a question, it truthfully
```

```
says it does not know.

    Relevant pieces of previous conversation:
    input: My favorite food is pizza
    output: that's good to know

    (You do not need to use these pieces of
information if not relevant)

    Current conversation:
    Human: Whats my favorite food
    AI:

    > Finished chain.





    ' You said your favorite food is pizza.'
```

```
# The memories from the conversation are
automatically stored,
# since this query best matches the
introduction chat above,
# the agent is able to 'remember' the user's
name.
conversation_with_summary.predict(input="What's
my name?")
```

```
    > Entering new ConversationChain chain...
    Prompt after formatting:
    The following is a friendly conversation
between a human and an AI. The AI is
talkative and provides lots of specific
details from its context. If the AI does not
know the answer to a question, it truthfully
says it does not know.

    Relevant pieces of previous conversation:
    input: Hi, my name is Perry, what's up?
    response:  Hi Perry, I'm doing well. How
about you?

    (You do not need to use these pieces of
information if not relevant)

    Current conversation:
    Human: What's my name?
    AI:

    > Finished chain.




    ' Your name is Perry.'
```