



# ConversationSummaryBufferMemory

`ConversationSummaryBufferMemory` combines the last two ideas. It keeps a buffer of recent interactions in memory, but rather than just completely flushing old interactions it compiles them into a summary and uses both. Unlike the previous implementation though, it uses token length rather than number of interactions to determine when to flush interactions.

Let's first walk through how to use the utilities

```
from langchain.memory import ConversationSummaryBufferMemory
from langchain.llms import OpenAI

llm = OpenAI()
```

## API Reference:

- `ConversationSummaryBufferMemory` from `langchain.memory`
- `OpenAI` from `langchain.llms`

```
memory = ConversationSummaryBufferMemory(llm=llm, max_token_limit=10)
memory.save_context({"input": "hi"}, {"output": "whats up"})
memory.save_context({"input": "not much you"}, {"output": "not much"})
```

```
memory.load_memory_variables({})
```

```
{'history': 'System: \n\nThe human says "hi", and the AI responds with "whats up".\nHuman: not much you\nAI: not much'}
```

We can also get the history as a list of messages (this is useful if you are using this with a chat model).

```
memory = ConversationSummaryBufferMemory(
    llm=llm, max_token_limit=10, return_messages=True
```

```
)
memory.save_context({"input": "hi"}, {"output": "whats up"})
memory.save_context({"input": "not much you"}, {"output": "not much"})
```

We can also utilize the `predict_new_summary` method directly.

```
messages = memory.chat_memory.messages
previous_summary = ""
memory.predict_new_summary(messages, previous_summary)
```

```
'\n\nThe human and AI state that they are not doing much.'
```

## Using in a chain

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

```
from langchain.chains import ConversationChain

conversation_with_summary = ConversationChain(
    llm=llm,
    # We set a very low max_token_limit for the purposes of testing.
    memory=ConversationSummaryBufferMemory(llm=OpenAI(),
    max_token_limit=40),
    verbose=True,
)
conversation_with_summary.predict(input="Hi, what's up?")
```

### API Reference:

- `ConversationChain` from `langchain.chains`

```
> 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.

Current conversation:

Human: Hi, what's up?

AI:

> Finished chain.

" Hi there! I'm doing great. I'm learning about the latest advances in artificial intelligence. What about you?"

```
conversation_with_summary.predict(input="Just working on writing some
documentation!")
```

> 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.

Current conversation:

Human: Hi, what's up?

AI: Hi there! I'm doing great. I'm spending some time learning about the latest developments in AI technology. How about you?

Human: Just working on writing some documentation!

AI:

> Finished chain.

```
' That sounds like a great use of your time. Do you have experience
with writing documentation?'
```

# We can see here that there is a summary of the conversation and then some previous interactions

```
conversation_with_summary.predict(input="For LangChain! Have you heard of
it?")
```

```
> 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.
```

```
Current conversation:
```

```
System:
```

```
The human asked the AI what it was up to and the AI responded that it
was learning about the latest developments in AI technology.
```

```
Human: Just working on writing some documentation!
```

```
AI: That sounds like a great use of your time. Do you have experience
with writing documentation?
```

```
Human: For LangChain! Have you heard of it?
```

```
AI:
```

```
> Finished chain.
```

```
" No, I haven't heard of LangChain. Can you tell me more about it?"
```

# We can see here that the summary and the buffer are updated

```
conversation_with_summary.predict(
    input="Haha nope, although a lot of people confuse it for that"
)
```

> 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.

Current conversation:

System:

The human asked the AI what it was up to and the AI responded that it was learning about the latest developments in AI technology. The human then mentioned they were writing documentation, to which the AI responded that it sounded like a great use of their time and asked if they had experience with writing documentation.

Human: For LangChain! Have you heard of it?

AI: No, I haven't heard of LangChain. Can you tell me more about it?

Human: Haha nope, although a lot of people confuse it for that

AI:

> Finished chain.

' Oh, okay. What is LangChain?'