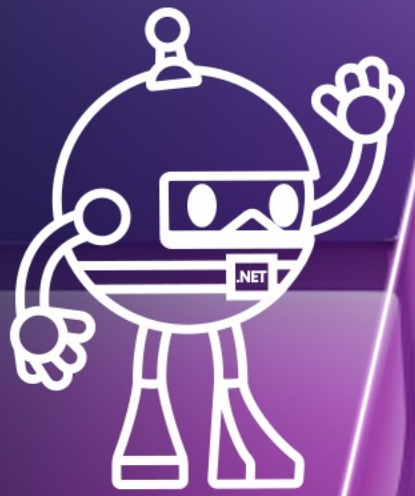
.NET Conf China 2023

2023/12/16
09:30 - 18:00

中国・北京



.NET中文社区

中国 · 北京

.NET Conf China 2023

使用Kernel Memory构建知识库

郭 强

Custouch市场易 技术研发经理

什么是SK Kernel Memory?

很抱歉，我目前并没有关于 "SK Kernel Memory" 的具体信息。可能这是一个特定的术语、缩写或者是某个特定领域的术语。如果可能的话，你能提供更多的上下文或者详细信息吗？这样我就可以更好地帮助你了解这个术语的含义。

Kernel Memory (KM) is an open-source service and plugin specialized in the efficient indexing of datasets through custom continuous data hybrid pipelines. For some scenarios KM is also available as a library, and soon as a Docker container. Utilizing advanced embeddings and LLMs, the system enables Natural Language querying for obtaining answers from the indexed data, complete with citations and links to the original sources. Designed for seamless integration as a Plugin with Semantic Kernel, Microsoft Copilot and ChatGPT, Kernel Memory enhances data-driven features in applications built for most popular AI platforms.
Question: 什么是SK Kernel Memory?

SK Kernel Memory（Semantic Kernel Kernel Memory）是一个开源的服务和插件，专门用于通过定制的连续数据混合流水线对数据集进行高效索引。在某些情况下，Kernel Memory也可以作为一个库使用，并很快将作为一个Docker容器提供。该系统利用先进的嵌入和LLM（大型语言模型），使其能够进行自然语言查询，以从索引的数据中获取答案，并提供完整的引用和原始来源的链接。
Kernel Memory旨在作为一个插件与Semantic Kernel、Microsoft Copilot和ChatGPT等系统进行无缝集成，以增强构建在大多数流行AI平台上的应用程序中的数据驱动功能。

Kernel Memory (KM) is an open-source service and plugin specialized in the efficient indexing of datasets through custom continuous data hybrid pipelines. For some scenarios KM is also available as a library, and soon as a Docker container. Utilizing advanced embeddings and LLMs, the system enables Natural Language querying for obtaining answers from the indexed data, complete with citations and links to the original sources. Designed for seamless integration as a Plugin with Semantic Kernel, Microsoft Copilot and ChatGPT, Kernel Memory enhances data-driven features in applications built for most popular AI platforms.
Question: 什么是SK Kernel Memory?

SK Kernel Memory（Semantic Kernel Kernel Memory）是一个开源的服务和插件，专门用于通过定制的连续数据混合流水线对数据集进行高效索引。在某些情况下，Kernel Memory也可以作为一个库使用，并很快将作为一个Docker容器提供。该系统利用先进的嵌入和LLM（大型语言模型），使其能够进行自然语言查询，以从索引的数据中获取答案，并提供完整的引用和原始来源的链接。
Kernel Memory旨在作为一个插件与Semantic Kernel、Microsoft Copilot和ChatGPT等系统进行无缝集成，以增强构建在大多数流行AI平台上的应用程序中的数据驱动功能。


提问
理解
检索
回答
组合
生成


.NET
.NET 8
.NET

RAG 检索增强生成

(Retrieval Augmented Generation)


提问
理解
检索
回答
生成
组合


.NET
8
.NET
.NET

Embedding

计算相似度

问题

向量0

文档1

向量1

0.832

文档2

向量2

0.912

文档3

向量3

0.411

文档N

向量N

0.651


W
P
PDF
文本提取
分片
存储
向量化
向量数据库
LLM
消息队列


.NET
8


W
P
PDF
文本提取
分片
存储
向量化
向量数据库


.NET
8
.NET
.NET


文本提取
分片
存储
向量化
向量数据库


.NET
8
.NET

```
> dotnet add package Microsoft.KernelMemory.Core
```

```
// using OpenAI
var memory = new KernelMemoryBuilder()
    .WithOpenAIDefaults(Env.Var("OPENAI_API_KEY"))
    .Build<MemoryServerless>();
```

```
> dotnet add package Microsoft.KernelMemory.Core
```

```
// using Azure OpenAI
var memory = new KernelMemoryBuilder()
    .WithAzureOpenAITextGeneration(new AzureOpenAIConfig()
    {
        Auth = AuthTypes.APIKey,
        APIKey = Env.Var("AZURE_OPENAI_API_KEY"),
        Endpoint = Env.Var("AZURE_OPENAI_ENPDOINT"),
        Deployment = "gpt-35-turbo-16k"
    }).WithAzureOpenAIEmbeddingGeneration(new AzureOpenAIConfig()
    {
        Auth = AuthTypes.APIKey,
        APIKey = Env.Var("AZURE_OPENAI_API_KEY"),
        Endpoint = Env.Var("AZURE_OPENAI_ENPDOINT"),
        Deployment = "text-embedding-ada-002"
    }).Build<MemoryServerless>();
```

```
// import documents
await memory.ImportDocumentAsync(
      new Document()
        .AddFiles(new[] {"How to Read a Book.pdf",
                         "How to Solve It.docx"}),
      index: "Books",
      steps: Constants.DefaultPipeline);

/*
string[] DefaultPipeline = new [] {
            "extract",
            "partition",
            "gen_embeddings",
            "save_records" };
*/
```

```
var question =
  "What are the four steps that Polya suggests for solving a mathematical problem? ";

// AskAsync
var answer = await memory.AskAsync(question);

Console.WriteLine(answer.Result + "/n");

foreach (var x in answer.RelevantSources)
{
    Console.WriteLine($"  * {x.SourceName} -- {x.Partitions.First().LastUpdate:D}");
}
```

```
var question =
  "What are the four steps that Polya suggests for solving a mathematical problem? ";

// SearchAsync
var results = await memory.SearchAsync(question);

foreach (var r in results.Results)
{
    Console.WriteLine($"  * {r.SourceName} -- {r.Partitions.First().LastUpdate:D}");
}
```

https://github.com/microsoft/kernel-memory/tree/main/service/Service

Kernel Memory Service
running on http://127.0.0.1:9001

OPENAPI

> dotnet add package Microsoft.KernelMemory.WebClient

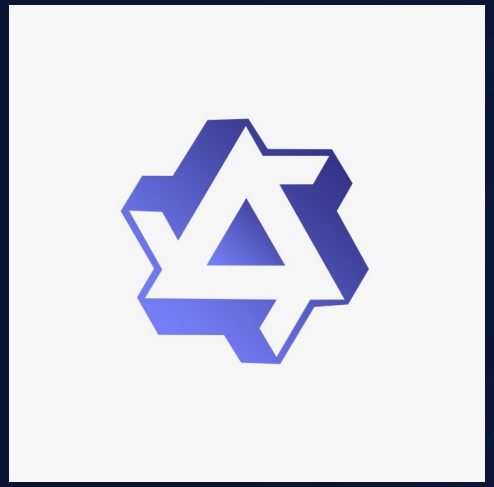
```
var memory = new MemoryWebClient("http://127.0.0.1:9001");
```

http://127.0.0.1:9001/swagger/index.html

> curl -F 'file1=@\"${FILENAME}\"' http://127.0.0.1:9001/upload

```
> dotnet add package ERNIE-Bot.KernelMemory
```

```
var client = new ERNIEBotClient("appId","appSecret");
var builder  = new KernelMemoryBuilder()
    .WithERNIEBotDefaults(client)
    .Build();
```



```
> dotnet add package DashScope.KernelMemory
```

```
var client = new DashScopeClient("apiKey");
var builder  = new KernelMemoryBuilder()
    .WithDefaultDashScopes(client)
    .Build();
```

https://github.com/SciSharp/LLamaSharp

LLaMa Sharp

Discord 36 online | QQ 加入QQ群 | LLamaSharp v0.8.1 | LLamaSharp.Backend.Cpu v0.8.1 | LLamaSharp.Backend.Cuda11 v0.8.1 | LLamaSharp.Backend.Cuda12 v0.8.1 | LLamaSharp.semantic-kernel v0.8.1 | LLamaSharp.kernel-memory v0.8.1

The C#/.NET binding of llama.cpp. It provides higher-level APIs to inference the LLaMA Models and deploy it on local device with C#/.NET. It works on both Windows, Linux and MAC without requirement for compiling llama.cpp yourself. Even without a GPU or not enough GPU memory, you can still apply LLaMA models well with this repo. 🤗

Furthermore, it provides integrations with other projects such as semantic-kernel, kernel-memory and BotSharp to provide higher-level applications.

Discussions about the roadmap to v1.0.0: #287

▶ Table of Contents

Thanks

Q&A

扫码加好友

扫码了解B2B企业营销服务