

前言 - Azure OpenAI Service 入门

本书是为 .NET 开发者而写的，让 .NET 开发者能快速掌握 Azure OpenAI Service 的使用技巧。

ChatGPT 的到来意味着我们已经置身于 AI 引起的全新变革中，作为开发者你将面临几种改变：

1. GPT 模型到来后，我们如何去架构好企业解决方案的问题。
2. Prompt 工程的到来，开发者是否能成功转型？
3. 如何打通 GPT 和企业数据？
4. 原来的人工智能技术还有用吗？
5. Azure OpenAI Service 和 OpenAI Service 比有什么优势？

通过本书的相关章节，将逐一告诉大家。现在 AI 的变化是日新月异的，因此我决定把这个内容放到 GitHub 上，如果大家有任何问题也可以在 Issue 上告诉我，我会尽快回应。

一. 关于 Azure OpenAI Service

The screenshot shows the official Microsoft Azure website. At the top, there's a navigation bar with links for 'Azure', 'Explore', 'Products', 'Solutions' (which is highlighted with a dashed border), 'Pricing', 'Partners', and 'Resources'. To the right of the navigation are 'Search', 'Learn', 'Support', 'Contact Sales', a green 'Free account' button, and a 'Sign in' link. The main content area features a large image of a woman with glasses looking thoughtfully to the side. Above her, the tagline 'AZURE. INVENT WITH PURPOSE.' is displayed. Below the image, the headline reads 'AI-powered innovation—accelerated with Azure'. A subtext below the headline says: 'Start building AI solutions today using the latest models and your favorite Azure tools. Pay-as-you-go or try Azure free for up to 30 days. There's no upfront commitment—cancel anytime.' At the bottom of the main section, there are two buttons: 'Get started' and 'Try Azure for free'. Below this, a secondary text block states: 'On-premises, hybrid, multicloud, or at the edge—create secure, future-ready cloud solutions on Azure'.

Microsoft Azure (<https://azure.com>) 是全球知名的云解决方案，微软和 OpenAI 合作，使 Microsoft Azure 也嵌入了 OpenAI 的服务，让企业能通过 Azure 快速打造基于 OpenAI 的行业解决方案。现在 Microsoft OpenAI Service 处于预览版阶段，只需要在线申请，符合条件通过后，就可以使用 Azure OpenAI Service 。

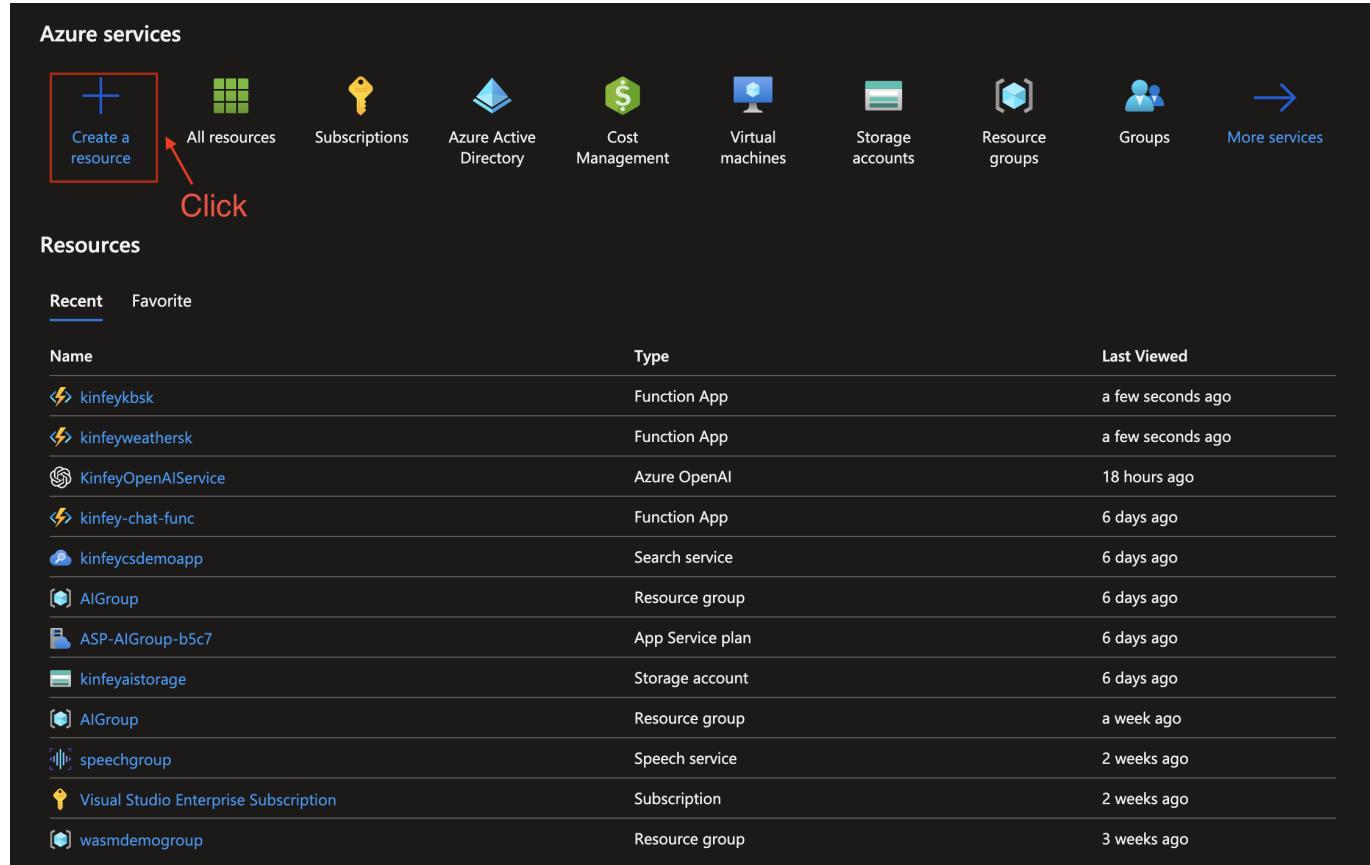
1.1 Azure OpenAI Service 和 OpenAI Service 之间有什么不同呢？

1. **安全性** - Microsoft Azure OpenAI Service 有严格的数据、网络、身份安全机制，为企业信息安全保驾护航。这就包括了密钥自动加密服务、数据隐私服务以及内容管理等。
2. **稳定性** - 通过 Microsoft Azure OpenAI Service 为企业带来更稳定的接口访问。
3. **可靠性** - 具备业务持续性和灾难恢复，当出现大范围故障时，还能从区域性故障中恢复，可以即时执行，而且成本非常低。

1.2 使用 Azure OpenAI Service

要使用 Azure OpenAI Service 你需要拥有一个 Azure 账号，并通过该账号去申请 Azure OpenAI Service。申请通过后，你就可以像一般的 Azure 服务一样通过资源组的方式，开始创建属于自己的 Azure OpenAI Service。

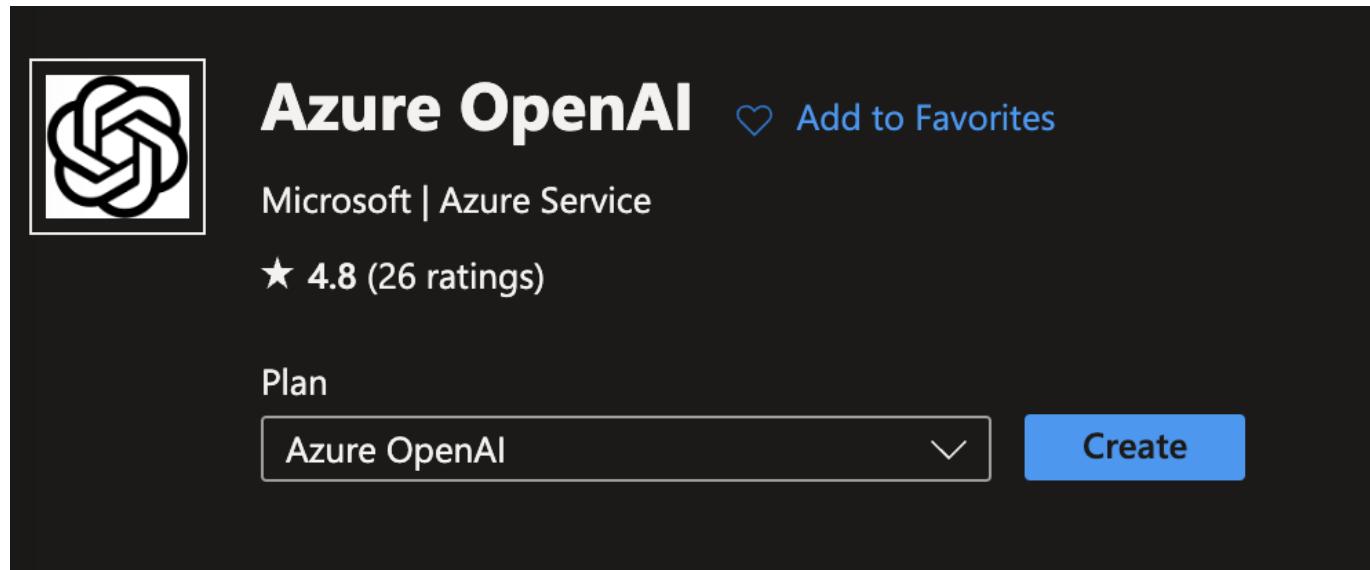
STEP 1. 进入 Azure Portal (<https://portal.azure.com/>)，点击创建资源：



The screenshot shows the Azure services dashboard. At the top, there's a navigation bar with icons for All resources, Subscriptions, Azure Active Directory, Cost Management, Virtual machines, Storage accounts, Resource groups, Groups, and More services. Below this is a section titled 'Resources' with tabs for Recent and Favorite. A list of resources is displayed with columns for Name, Type, and Last Viewed. The resources listed include various Function Apps, an Azure OpenAI service, a Search service, Resource groups, App Service plans, Storage accounts, and Speech services.

Name	Type	Last Viewed
⚡ kinfeyksk	Function App	a few seconds ago
⚡ kinfeyweathersk	Function App	a few seconds ago
⚡ KinfeyOpenAIService	Azure OpenAI	18 hours ago
⚡ kinfey-chat-func	Function App	6 days ago
☁️ kinfeycsdemoapp	Search service	6 days ago
[?] AIGroup	Resource group	6 days ago
ASP-AIGroup-b5c7	App Service plan	6 days ago
blob storage	Storage account	6 days ago
[?] AIGroup	Resource group	a week ago
speechgroup	Speech service	2 weeks ago
🔑 Visual Studio Enterprise Subscription	Subscription	2 weeks ago
[?] wasm demogroup	Resource group	3 weeks ago

选择 Azure OpenAI 进行创建：



The screenshot shows the Azure OpenAI service page. It features the Microsoft logo and the text 'Azure OpenAI' with an 'Add to Favorites' button. Below this, it says 'Microsoft | Azure Service' and displays a rating of '★ 4.8 (26 ratings)'. There's a 'Plan' section with a dropdown menu set to 'Azure OpenAI' and a large blue 'Create' button.

STEP 2. 创建过程需要选择你的 Azure 订阅，并创建相关资源组，以及选择区域(建议选择 South Central US)，并给一个唯一的名字，并选择价格，如图：

Create Azure OpenAI

Basics **Tags** **Review + submit**

Enable new business solutions with OpenAI's language generation capabilities powered by GPT-3 models. These models have been pretrained with trillions of words and can easily adapt to your scenario with a few short examples provided at inference. Apply them to numerous scenarios, from summarization to content and code generation.

[Learn more](#)

Project Details

Subscription *	Visual Studio Enterprise Subscription
Resource group *	AIGroup
	Create new

Instance Details

Region	South Central US
Name *	KinfeyOpenAIDemoService
Pricing tier *	Standard S0

[View full pricing details](#)

STEP 3. 创建成功后，选择模型部署选项，点击创建：

Home > KinfeyOpenAIService

KinfeyOpenAIService | Model deployments

Model deployment ... State Model Version Scale Type

Model deployment ...	State	Model	Version	Scale Type
GPT3Model	Succeeded	text-davinci-003	1	Standard
ChatGPT35Model	Succeeded	gpt-35-turbo (versio... 0301	0301	Standard
TextEmbeddingModel	Succeeded	text-embedding-ada...	1	Standard

然后选择你所需要的模型进行部署：

The screenshot shows the Azure OpenAI Service interface. On the left, there's a sidebar with navigation links like Home, Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Resource Management (Keys and Endpoint, Model deployments selected), Pricing tier, Networking, Identity, Cost analysis, Properties, Locks, Monitoring (Alerts, Metrics, Diagnostic settings, Logs). The main area shows 'KinfeyOpenAIService | Model deployments' with three listed models: GPT3Model (Succeeded), ChatGPT35Model (Succeeded), and TextEmbeddingModel (Succeeded). A 'Create' button is at the top. On the right, a modal window titled 'Create Model deployment' is open, asking for a 'Model deployment name' (with a red asterisk) and a 'Model' dropdown. The dropdown list includes: text-ada-001, text-babbage-001, text-curie-001, text-davinci-001, text-davinci-002, gpt-35-turbo (version 0301), text-davinci-003, code-cushman-001, text-similarity-ada-001, text-similarity-babbage-001, text-similarity-curie-001, text-similarity-davinci-001, text-search-ada-doc-001, text-search-ada-query-001, and text-search-babbage-doc-001.

完成后就可以看到相关的列表。

Model deployment ...	State	Model	Version	Scale Type
GPT3Model	Succeeded	text-davinci-003	1	Standard
ChatGPT35Model	Succeeded	gpt-35-turbo (versio... 0301		Standard
TextEmbeddingModel	Succeeded	text-embedding-ada...	1	Standard

注意：现在 GPT-4 需要额外申请，具体请参考 <https://aka.ms/oaiapply>

二. 通过 Azure OpenAI Studio 管理你的 OpenAI 应用场景

Cognitive Services | Azure OpenAI Studio

Azure OpenAI Studio

Get started with Azure OpenAI Service

Get example prompts for different scenarios and write prompts of your own. Export your prompts to code at any time to rapidly iterate at scale and integrate with your apps.

Try the playgrounds

Get example prompts for different scenarios and write prompts of your own. Export your prompts to code at any time to rapidly iterate at scale and integrate with your apps.

Completions playground **Chat playground (Preview)**

Explore examples for prompt completion

Summarize an article (abstractive)
Use abstractive summarization to summarize an article. You can also ask the model to summarize in different ways such as asking for bullet points or a summary with specific details.

Generate product name ideas
Generate product name ideas based on a description and some seed words.

Classify Text
Classify items into categories provided at inference time.

Natural Language to SQL
Translate natural language to SQL queries.

通过 Azure OpenAI Studio (<https://oai.azure.com/portal>) 你可以迅速完成 Azure OpenAI 模型的应用、部署及管理。我在这里选择一个简单的总结场景来完成相关的展示：

1. 选择总结文章的示例场景：

Summarize an article (abstractive)



Use abstractive summarization to summarize an article.

Prompt

Provide a summary of the text below that captures its main idea.

At Microsoft, we have been on a quest to advance AI beyond existing techniques, by taking a more holistic, human-centric approach to learning and understanding. As Chief Technology Officer of Azure AI Cognitive Services, I have been working with a team of amazing scientists and engineers to turn this quest into a reality. In my role, I enjoy a unique perspective in viewing the relationship among three attributes of human cognition: monolingual text (X), audio or visual sensory signals, (Y) and multilingual (Z). At the intersection of all three, there's magic—what we call XYZ-code as illustrated in Figure 1—a joint representation to create more powerful AI that can speak, hear, see, and understand humans better. We believe XYZ-code will enable us to fulfill our long-term vision: cross-domain transfer learning, spanning modalities and languages. The goal is to have pre-trained models that can jointly learn representations to support a broad range of downstream AI tasks, much in the way humans do today. Over the past five years, we have achieved human performance on benchmarks in conversational speech recognition, machine translation, conversational question answering, machine reading comprehension, and image captioning. These five breakthroughs provided us with strong signals toward our more ambitious aspiration to produce a leap in AI capabilities, achieving multi-sensory and multilingual learning that is closer in line with how humans learn and understand. I believe the joint XYZ-code is a foundational component of this aspiration, if grounded with external knowledge sources in the downstream AI tasks.

Sample Response

The Chief Technology Officer of Azure AI Cognitive Services is working with a team of scientists and engineers to develop a more holistic, human-centric approach to learning and understanding AI. This approach, called XYZ-code, involves a joint representation of monolingual text, audio or visual sensory signals, and multilingual elements. The goal is to create pre-trained models that can learn representations to support a range of AI tasks, similar to how humans learn and understand. Over the past five years, the team has achieved human performance on various benchmarks.

API Request

1 #Note: The openai-python library support for Azure OpenAI is in preview.

Settings

Max tokens

250

```

2 import os
3 import openai
4 openai.api_type = "azure"
5 openai.api_base = "https://kinfeyopenaiservice.openai.azure.com/"
6 openai.api_version = "2022-12-01"
7 openai.api_key = os.getenv("OPENAI_API_KEY")
8
9 response = openai.Completion.create(
10   engine="DEPLOYMENT_NAME",
11   prompt="Provide a summary of the text below that captures its main idea.\n\nAt Microsoft, we have been on a quest to advance AI beyond existing techniques, by taking a more holistic, human-centric approach to learning and understanding. As Chief Technology Officer of Azure AI Cognitive Services, I have been working with a team of amazing scientists and engineers to turn this quest into a reality. In my role, I enjoy a unique perspective in viewing the relationship among three attributes of human cognition: monolingual text (X), audio or visual sensory signals, (Y) and multilingual (Z). At the intersection of all three, there's magic—what we call XYZ-code as illustrated in Figure 1—a joint representation to create more powerful AI that can speak, hear, see, and understand humans better. We believe XYZ-code will enable us to fulfill our long-term vision: cross-domain transfer learning, spanning modalities and languages. The goal is to have pre-trained models that can jointly learn representations to support a broad range of downstream AI tasks, much in the way humans do today. Over the past five years, we have achieved human performance on benchmarks in conversational speech recognition, machine translation, conversational question answering, machine reading comprehension, and image captioning. These five breakthroughs provided us with strong signals toward our more ambitious aspiration to produce a leap in AI capabilities, achieving multi-sensory and multilingual learning that is closer in line with how humans learn and understand. I believe the joint XYZ-code is a foundational component of this aspiration, if grounded with external knowledge sources in the downstream AI tasks.",
12   temperature=0.3,
13   max_tokens=250,
14   top_p=1,
15   frequency_penalty=0,
16   presence_penalty=0,
17   best_of=1,
18   stop=None)

```

Randomness	0.3
Top probabilities	1
Frequency penalty	0
Presence penalty	0
Stop sequences	

Model Info

text-davinci-003

[Open in Playground](#)

[Close](#)

2. 点击进入 Playground 后，你就可以输入相关 Prompt 来完成一些应用设定：

The screenshot shows the Azure OpenAI Studio interface for the Completions playground. On the left, a sidebar has 'Playground' and 'Completions' selected. The main area is titled 'Completions playground' and contains a text input field with the prompt: "Provide a summary of the text below that captures its main idea." To the right, a large panel titled 'Parameters' contains various settings for generating text. The 'Temperature' slider is set to 0.3, and the 'Max length (tokens)' slider is set to 250. Other parameters shown include 'Stop sequences', 'Top probabilities' (set to 1), 'Frequency penalty' (set to 0), 'Presence penalty' (set to 0), and 'Best of' (set to 1). At the bottom, there are buttons for 'Generate', 'Undo', 'Regenerate', and a 'Tokens: 0' counter.

你也可以选择不同的参数进行调整，当然选择 View code 也可以快速地查看相关代码，直接引用就可以给 Python 和 .NET 使用，其他语言也可以参考 curl , json 进行调整：

Sample Code

You can use the following code to start integrating your current prompt and settings into your application

```
https://kinfeyopenaiservice.openai.azure.com/openai/de  
ployments/GPT3Model/completions?api-version=2022-  
12-01
```

curl



```
1 curl  
https://kinfeyopenaiservice.openai.azure.com/openai/deployments/GP  
T3Model/completions?api-version=2022-12-01 \  
2   -H "Content-Type: application/json" \  
3   -H "api-key: YOUR_API_KEY" \  
4   -d '{  
5     "prompt": "Provide a summary of the text below that captures  
its main idea.\n\nAt Microsoft, we have been on a quest to advance  
AI beyond existing techniques, by taking a more holistic, human-  
centric approach to learning and understanding. As Chief  
Technology Officer of Azure AI Cognitive Services, I have been  
working with a team of amazing scientists and engineers to turn  
this quest into a reality. In my role, I enjoy a unique  
perspective in viewing the relationship among three attributes of  
human cognition: monolingual text (X), audio or visual sensory  
signals, (Y) and multilingual (Z). At the intersection of all  
three, there's magic—what we call XYZ-code as illustrated in  
Figure 1—a joint representation to create more powerful AI that  
can speak, hear, see, and understand humans better. We believe  
XYZ-code will enable us to fulfill our long-term vision: cross-  
domain transfer learning, spanning modalities and languages. The  
goal is to have pre-trained models that can jointly learn  
representations to support a broad range of downstream AI tasks,  
much in the way humans do today. Over the past five years, we have  
achieved human performance on benchmarks in conversational speech
```

Your API key can be found by going to your resource in the Azure Portal. [Learn more here](#). You should use environment variables or a secret management tool like Azure Key Vault to prevent accidental exposure of your key in applications.

Copy

Close

补充一下引用代码时别忘记添加 Key , Key 所在位置在创建的 Azure OpenAI 资源里:

The screenshot shows the 'Keys and Endpoint' section of the Azure OpenAI Service. On the left sidebar, under 'Resource Management', 'Keys and Endpoint' is selected. The main area displays two sets of API keys: 'KEY 1' and 'KEY 2'. Each key is represented by a long string of asterisks followed by a copy icon. A blue information box at the top right provides a warning about key security and regeneration.

三. Azure OpenAI Service 的能力和概念

我们在使用 Azure OpenAI Service 的使用，都会碰到一些概念，我们进行一个简单的入门：

3.1 Azure OpenAI 的能力

Azure OpenAI 具备文本生成，代码生成，图像生成的三大能力：

1. 文本生成

文本生成能力具备文本归纳、分类、翻译、问答、创作、推荐等能力。在 Azure OpenAI Service 上，你可以通过部署 GPT-3 / GPT-3.5 / GPT-4 的模型为应用提供文本生成功能。我们如果在无需编码的时候，你可以使用 ChatGPT 体验强大的功能，当然你也可以通过 Azure OpenAI Studio 的 Playground 体验 ChatGPT。

The screenshot shows the 'Chat playground (Preview)' interface in Azure OpenAI Studio. The left sidebar has a 'Chat' tab selected. The main area is divided into three panels: 'Assistant setup', 'Chat session', and 'Parameters'. The 'Assistant setup' panel contains a 'System message' input field with the placeholder '你好我是你的 ChatGPT' and a 'User message' input field with the placeholder 'Type user query here. (Ctrl + Enter for new line)'. The 'Chat session' panel shows a conversation history with AI-generated responses. The 'Parameters' panel allows configuring deployment (set to 'ChatGPT35Model'), max response length (800), temperature (0.7), top p (0.95), and stop sequences. Session settings include past messages included (10) and current token count (545/4000).

2. 代码生成

如果你已经是 GitHub Copilot 的用户，有惊艳到吗？你希望拥有一个自己企业的智能代码助手吗？通过 Azure OpenAI Service 提供的 Codex 模型就可以配合完成。Codex 模型基于 GPT-3，经过优化以理解和编写代码。这些模型基于自然语言以及用来自公共存储库的数十亿行代码进行了训练。Codex 能够从自然语言指令（如代码注释）生成代码，并且可以建议完成代码函数的方法。

3. 图像生成

处理图像的模型称为 DALL-E。图像功能包括分为创建图像、编辑图像和创建图像变体这三类。

3.2 Azure OpenAI 的参数设定

在 Azure OpenAI Studio 的 Playground 中，可以看到一些参数，这些参数对于生成式 AI 是非常重要的，下面我们就来学习一下：

Temperature

是控制生成的文本输出的随机程度的参数。Temperature 值越高，输出的结果越随机，而值越低，输出的结果则越趋向于确定性。

Token

是指在自然语言处理中的一个概念，它是文本中的一个基本单元，通常是一个单词或一个标点符号。在自然语言处理中，为了方便处理文本，我们需要将文本中的每个单词或标点符号都转换为一个数字表示，这个数字就被称为 Token。在 OpenAI 的一些模型中，Token 还可以包括特殊的 Token，如起始 Token、结束 Token，用于指示模型的输入和输出。在使用 OpenAI 的模型进行训练或生成文本时，需要对文本进行分词和 Token 化处理，以便对文本进行处理和表示。

Top K

是指在自然语言处理中，使用的一个参数，用于控制生成文本的多样性。具体来说，Top K 参数指的是在生成下一个词时，只考虑概率值最高的前 K 个候选词，而不是所有可能的词。这样可以使得生成的文本更加多样化和有趣，避免出现过于模板化和单调的文本。通常情况下，K 的值会设置在几十到几百之间，具体取决于任务和数据集的不同。

Top P

是在自然语言处理中，用于控制生成文本的多样性的一种技术。具体来说，Top P 参数指的是在生成下一个词时，只考虑概率累加值最高的一组词，这组词的概率之和大于等于给定的阈值 P。而不是考虑所有可能的词。这个阈值 P 通常被设置在 0.1 到 0.9 之间，具体取决于任务和数据集的不同。Top P 技术可以使得生成的文本更加多样化，避免出现重复和单调的文本。

Frequency Penalty

是在自然语言处理中，用于控制生成文本的多样性的一种技术。具体来说，Frequency Penalty 可以用于降级那些在之前的生成文本中出现频率较高的词语，在生成下一个词时，这些词语的概率会被降低，从而鼓励模型生成更加多样化的文本。通过调整 Frequency Penalty 的大小，可以实现对文本多样性和流畅性的平衡。

Presence Penalty

是在自然语言处理中，用于控制生成文本的多样性的一种技术。Presence Penalty 可以用于降级那些在之前的生成文本中已经出现过的词语，在生成下一个词时，这些词语的概率会被降低，从而鼓励模型生成更加多

样化的文本。与 Frequency Penalty 不同的是，Presence Penalty 不仅考虑了词语的出现频率，还考虑了词语是否已经出现过。通过调整 Presence Penalty 的大小，可以实现对文本多样性和流畅性的平衡。

四..NET 开发者入门

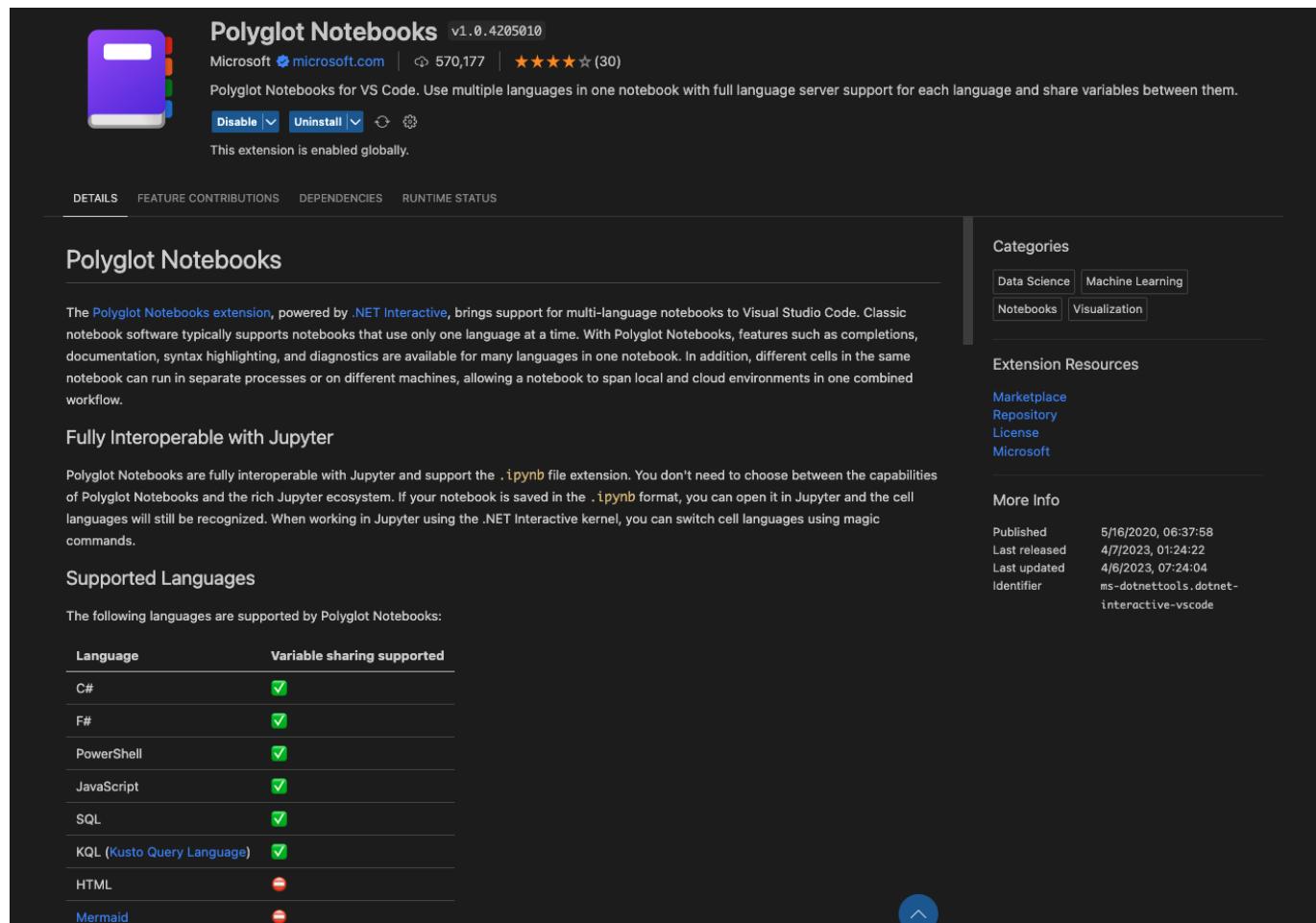
4.1 .NET for Azure OpenAI Service 基础

Azure OpenAI Service 提供了 REST 的 API 给不同语言进行调用，针对 .NET 用户更发布了基于 Azure OpenAI 的 .NET SDK，.NET 开发者可以快速接入进行 Azure OpenAI 应用场景的开发。

```
dotnet add package Azure.AI.OpenAI
```

当然你也可以用 HttpClient 的方式直接访问 endpoint 来完成。

对于要探索 Azure OpenAI Service 的 .NET 开发者，个人建议使用 Notebook 的方式来配合。你只需要通过最新的 .NET SDK 7 和 Visual Studio Code 及 .NET Extensions Pack 的 Visual Studio Code 插件就可以通过 Polyglot Notebook 做交互式的开发体验。这样的好处是，更方便调整参数和优化 OpenAI 的代码。



The screenshot shows the Polyglot Notebooks extension page on the Microsoft Marketplace. The extension is version v1.0.4205010, developed by Microsoft, with 570,177 installs and a rating of 4 stars. It is described as a tool for VS Code that supports multiple languages in one notebook. The page includes sections for Details, Feature Contributions, Dependencies, Runtime Status, Categories (Data Science, Machine Learning, Notebooks, Visualization), Extension Resources (Marketplace, Repository, License, Microsoft), and More Info (Published: 5/16/2020, Last released: 4/7/2023, Last updated: 4/6/2023, Identifier: ms-dotnettools.dotnet-interactive-vscode). Below the main description, there's a section titled 'Fully Interoperable with Jupyter' and another titled 'Supported Languages' which lists various programming languages supported by the extension.

Language	Variable sharing supported
C#	✓
F#	✓
PowerShell	✓
JavaScript	✓
SQL	✓
KQL (Kusto Query Language)	✓
HTML	✗
Mermaid	✗

以下是在 Polyglot Notebook 调用 Azure OpenAI Service 中 GPT-3.5 turbo 的 Notebook 实现：

```
#r "nuget: Azure.AI.OpenAI , 1.0.0-beta.5"
[1] ✓ 0.7s
... Installed Packages
• Azure.AI.OpenAI, 1.0.0-beta.5

▷
using Azure;
using Azure.AI.OpenAI;
[2] ✓ 0.0s

▷
OpenAIClient client = new OpenAIClient(
    new Uri("https://kinfeyopenaiservice.openai.azure.com/"),
    new AzureKeyCredential("your-key-here"));
[3] ✓ 0.2s

Response<ChatCompletions> responseWithoutStream = await client.GetChatCompletionsAsync(
    "ChatGPT35Model",
    new ChatCompletionsOptions()
{
    Messages =
    {
        new ChatMessage(ChatRole.System, @"我是你的人工智能助理"),
    },
    Temperature = (float)0.7,
    MaxTokens = 800,
    NucleusSamplingFactor = (float)0.95,
    FrequencyPenalty = 0,
    PresencePenalty = 0,
});
[4] ✓ 2.3s

    ChatCompletions completions = responseWithoutStream.Value;
[5] ✓ 0.0s

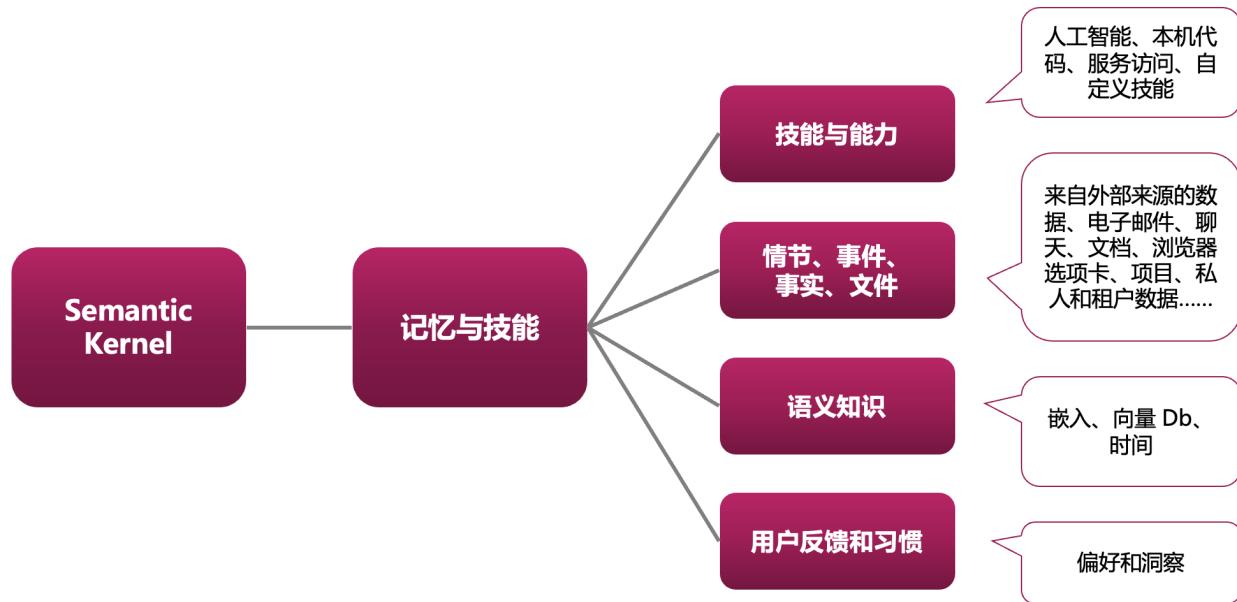
completions
[6] ✓ 0.0s
...
▼ Azure.AI.OpenAI.ChatCompletions
  Id      chatmpl-72YSrNteuqGXBQmfhtbNt6e1aclYP
```

4.2 Semantic Kernel 面向 AI 编程的框架

介绍一个微软最新的面向 AI 编程的开源框架 - Semantic Kernel (<https://github.com/microsoft/semantic-kernel>)。生成式人工智能诞生后，从告诉机器要怎么做转变为告诉机器必须要做什么，这就降低了对于编码的依赖。例如翻译、归纳、总结再不用做具体算法了，只需要用简单的提示语，AI 就可以帮你完成。那就是说我们需要做的是写 Prompt 触发人工智能做事，这也是为什么现在 Prompt 工程人员这么值钱的原因。

Prompt 的内容涵盖不同的场景，不同的任务，涉及到各式各样的工作流程。如何管理好这些 Prompt 让生成式 AI 能完成任务，这是大家所关注的。还有通过上下文对话作出不同的响应，让 AI 注入到对话流中也很重要。Semantic Kernel 就是用来管理和解决上下文对话以及各种技能的框架。

Semantic Kernel 是面向 AI 编程的最佳方案，可以结合不同场景、综合技能和意图，支持多语言开发的框架。.NET 开发者可以通过 Semantic Kernel，更高效地使用生成式 AI 进行应用开发。本书会基于 SK 来展开，让大家在学习 Azure OpenAI 的过程中，同时学习 Semantic Kernel 相关知识。



本章小结

本章作为前言部分，主要介绍了 Azure OpenAI Service 的相关功能，以及针对 .NET 开发者的 SDK。希望以此作为 AI 开发者入门的第一步。

相关资料

1. 免费 Azure 申请 <https://azure.com/free>
2. Azure for Student 申请（需要 edu 邮箱）<https://aka.ms/studentgetazure>
3. Azure OpenAI Service 申请 <https://aka.ms/oaiapply>
4. 关于 Semantic Kernel <https://github.com/microsoft/semantic-kernel>

第一章. 问题分类

我会把问题设定放在首位也就是我们的第一章。毕竟所有生成式的 AI 都是需要基于问题给出答案。所以我一直不认同人工智能会取代人，没人类哪有问题呢？

ChatGPT 的神奇之处在于它可以根据你的问题去完成不同的工作，如归纳，翻译，还有分类等。我们的问题有多种多样，我们除了问日常生活，通用知识外，还会遇到不同的时效性的问题和针对特定行业的内容。这个时候往往交给 ChatGPT，你会发现有时候它会胡说八道。这个时候或者我们需要去纠正这些错误。你可以用一个 Prompt 提示语，去设定一些规则让 ChatGPT 可以回答“不太清楚”，“没法解答”，“..... 我在努力学习等”。这也是 Prompt 工程师去设置的地方。其实我们在早期的对话机器人里，就有非常多针对实时问题和行业专有问题的解答。或者我们可以把这些语料重新给到我们的 GPT 模型，让它变得更加强大。或者把问题分类是一个最根本的做法。

像该例子，我们用了 OpenAI 的 davinci-03 模型去解答“今天天气”，你会没法找到答案。或者你会说 ChatGPT 很笨，但实际上在 API 主导的年代，不是一个困难的事情。

对问题进行分类，我们有两种方式，通过机器学习进行问题的分类是我们过往经常用，除了分类外，更可以快速地提取不同的语义实体，基于问题进行更细致划分，找到更符合的答案。现在还可以基于 OpenAI 去完成分类，对于不熟悉机器学习对的人来说是更好的选择。

一. 通过 ML.NET 对问题进行分类

针对 .NET 我们有很好的 Machine Learning 工具 ML.NET。通过 ML.NET 你可以快速对文本进行分类。

通过 ML.NET 你可以快速地完成相关的问题分类。请参考

[./Code/01.HowToQuestionClassification/cn/MLdotNETTextClassification.ipynb](#)

注意几个问题：

1. ML.NET 2.x 虽然支持了 NLP 的文本分类，但是现阶段不支持中文，所以我用到 Jieba，如果你是用英语的方式可以直接采用 BERT 来完成
2. 在 Notebook 没办法加载 ML.NET 生成的模型，只能在传统程序中调用
3. 现在没办法转换为 ONNX

二. 通过 Azure OpenAI Service 对问题进行分类

或者你也可以通过 Azure OpenAI Service 对问题进行分类，这是更多开发者希望见到的，也是最傻瓜的方法

打开 Azure Portal，进入创建好的 Azure OpenAI Service，选择 davinci-003 部署

Model deployment name	State	Model	Version	Scale Type
GPT3Model	Succeeded	text-davinci-003	1	Standard
ChatGPT35Model	Succeeded	gpt-35-turbo (version... 0301)		Standard
TextEmbeddingModel	Succeeded	text-embedding-ada-...	1	Standard

部署成功后，进入 Azure OpenAI Studio，打开 Playground，选择刚才创建的 davinci-003 - GPT3Model，并选择文本分类

文本分类

Deployments 对应 text-davinci-003

Examples

GPT3Model

Classify Text

修改 Prompt

请帮我针对问题进行分类，包括天气，课程，生成式

问：会下雨吗？类别：天气
 问：今天温度？类别：天气
 问：适度多少？类别：天气
 问：什么是新能源车？类别：课程
 问：新能源车的特点？类别：课程
 问：概念是什么？类别：课程
 问：写一首诗歌？类别：生成式
 问：翻译一下 类别：生成式
 问：计算结果 类别：生成式
 问：电动车特点 类别：

尝试测试

Deployments

GPT3Model

Examples

Classify Text

请帮我针对问题进行分类，包括天气，课程，生成式

问：会下雨吗？类别：天气
 问：今天温度？类别：天气
 问：适度多少？类别：天气
 问：什么是新能源车？类别：课程
 问：新能源车的特点？类别：课程
 问：概念是什么？类别：课程
 问：写一首诗歌？类别：生成式
 问：翻译一下类别：生成式
 问：计算结果类别：生成式
 问：电动车特点类别：课程

选择 C# 拷贝代码，打开 Visual Studio Code，创建

./Code/01.HowToQuestionClassification/cn/AzureOpenAITextClassification.ipynb

```
#> "nuget: Azure.AI.OpenAI , 1.0.0-beta.5"
[1] ✓ 0.8s
... Installed Packages
• Azure.AI.OpenAI 1.0.0-beta.5

using Azure;
using Azure.AI.OpenAI;
[2] ✓ 0.0s

[D] OpenAIClient client = new OpenAIClient(
    new Uri("https://sk-infopenaiservice-openai.azure.com/"),
    new AzureKeyCredential("your API Key"));
[3] ✓ 0.2s

string prompt = "请帮我针对问题进行分类，包括天气，课程，生成式\n问：会下雨吗？类别：天气\n问：今天温度？类别：天气\n问：适度多少？类别：天气\n问：什么是新能源车？类别：课程\n问：新能源车的特点？类别：课程\n问：概念是什么？类别：课程\n问：写一首诗歌？类别：生成式\n问：翻译一下类别：生成式\n问：计算结果类别：生成式\n问：电动车特点类别：课程"

string ask = "问：会下雨吗？类别：" ;
[4] ✓ 0.0s

Response<Completion> completionsResponse = await client.GetCompletionsAsync(
    deploymentOrModelName: "GPT3Model",
    new CompletionsOptions()
    {
        Prompts = new List<string> { prompt + ask },
        MaxTokens = 60,
        NucleusSamplingFactor = (float)1,
        FrequencyPenalty = (float)0,
        PresencePenalty = (float)0,
        GenerationTimeoutSeconds = 5
    });
[5] ✓ 1.7s

Completions completions = completionsResponse.Value;
[6] ✓ 0.0s

completions
[7] ✓ 0.0s
... ▾ Azure.AI.OpenAI.Completions
  Id      cmpl-72cRkumf0gZwwie447mM3FD1
  Created 1680940107
  Model   text-davinci-003
  Choices index value
          0     ▾ Azure.AI.OpenAI.Choice
  Usage   ▾ Azure.AI.OpenAI.CompletionUsage
completions.Choices[0].Text
[8] ✓ 0.0s
... 天气
```

具体请查看 ./Code/01.HowToQuestionClassification/cn/AzureOpenAITextClassification.ipynb

三. 传统机器学习文本分类和 Azure OpenAI Service 文本分类对比

传统机器学习的文本分类优势是模型可以作为离线进行使用，语料不足的情况下，难以获得准确的分类结果。而且技术要求较高。你的场景需要隔离内部和外部业务数据的或者传统机器学习更佳。而 Azure OpenAI Service 可以在 Prompt 上设置少样例来完成分类，但依赖于网络。现在网络通行的年代 Azure OpenAI Service 有更大的可用价值。而且 Azure OpenAI Service 的成本非常低。请记住一句话，场景还是很重要，

但 Azure OpenAI Service 并不是让你抛弃你原有的技术。你需要结合不同的人工智能知识来打造智能化的解决方案。

四. 用 Semantic Kernel 来做文本分类

OpenAI 是新物种，很多人希望除了能通过 Azure OpenAI Service 访问到 API 完成企业级的应用外，更希望能有一个好的架构来管理好 OpenAI 的项目。如果单纯从 REST 的角度，或者作为 .NET 开发者已经熟能生巧。但实际上如果你深入 OpenAI 的应用，你会发现需要 OpenAI 的应用更集中在我们写 Prompt 的管理上。这又和我们传统意义上的架构有所不同。首先它不再是代码主导，更多是以 Prompt 为代表的文本。你可以通过 Prompt 去描述一段要求，让 OpenAI 去完成。当我们希望结合 OpenAI 来构建智能系统对的时候就会发现，我们需要非常多的 Prompt 来完成不同业务工作。如果去管理 Prompt，以及如何优化好我们不同业务流的智能化工作，是架构一个好的 OpenAI 所需要的。如前言所说，Semantic Kernel 是帮我们管理各式各样 Prompt (也就是 Skill) 的框架。或者我们可以先用 Semantic Kernel 来做一个文本分类来学习一下。

要使用 Semantic Kernel，我们需要引入 .NET Semantic Kernel 的库，我们先创建一个 SKTextClassification.ipynb，接下来操作如下：

1. 创建一行 Cell，引入 Microsoft.SemanticKernel 的 .NET 库

```
#r "nuget: Microsoft.SemanticKernel, *-*"
```

2. 引入 Microsoft.SemanticKernel 的命名空间

```
using Microsoft.SemanticKernel;
using Microsoft.SemanticKernel.SemanticFunctions;
```

3. 创建 SK 内核实例

```
IKernel kernel = Kernel.Builder.Build();
```

Semantic Kernel (SK) 中的内核是用户问题的编排器。内核结合技能，内存和连接器来实现用户的智能化预期，除了配置基本的 Azure OpenAI / OpenAI 的基本链接串/模型/参数外，还可以基于用户的要求配对相关技能，整合技能形成工作流等。

接下来添加信息，添加和 Azure OpenAI Service 相关的信息

```
kernel.Config.AddAzureOpenAITextCompletionService(  
    "GPT3",  
    "text-davinci-003",  
    "Your Endpoint",  
    "Your Key"  
) ;
```

4. 设置和文本分类相关的 Prompt

```
string skPrompt = """
```

请帮我把 {{\\$input}} 进行类别确认，类别包括天气，课程，生成式，如果不太清楚，请回答没法确认，分类参考如下：

问：会下雨吗？ 类别：天气
问：今天温度？ 类别：天气
问：适度多少？ 类别：天气
问：什么是新能源车？ 类别： 课程
问：电动车的特点 类别： 课程
问：概念是什么？ 类别： 课程
问：课程相关的内容有哪些？ 类别： 课程
问：写一首诗歌？ 类别： 生成式
问：翻译一下 类别： 生成式
问：计算结果 类别： 生成式

如果能确认类别，天气相关请只输出 1， 课程相关请只输出 2， 生成式相关请只输出 3， 没法确认相关请只输出 0，并把{{\\$input}}和它的类别参考以下 json 格式输出

```
{"question": "{{\$input}}", "label": "{{\$label}}"}  
""";
```

Prompt 是非常非常重要的，对于生成式 UI 来说，有一个好的 Prompt 事半功倍。如何写 Prompt 或者网上有不同的方式，建议大家查阅。我总觉得文科生比理科生更容易做 OpenAI 的项目。

5. 和模型相关的配置

```
var promptConfig = new PromptTemplateConfig  
{  
    Completion =  
    {  
        MaxTokens = 60,  
        FrequencyPenalty = (float)0,
```

```
        PresencePenalty = (float)0
    }
};

var promptTemplate = new PromptTemplate(
    skPrompt,
    promptConfig,
    kernel
);
```

PromptTemplateConfig 就是设定我们对应模型的参数，你可以基于你所使用的模型进行调整，这里我参照了通过 Playground 上引用的参数。至于 PromptTemplate 这里面就是整合了我们创建的 kernel , prompt, 以及模型参数。告诉计算机是该内核使用 Azure OpenAI Service 的 text-davinci-003模型，prompt (skill) ，以及相关模型参数。

6. 创建一个 SemanticFunctionConfig，绑定 promptConfig , promptTemplate，为内核添加一个具备文本分类功能的函数做准备

```
var functionConfig = new SemanticFunctionConfig(promptConfig,
promptTemplate);
```

7. 通过内核注册技能和能力

```
var classificationFunction = kernel.RegisterSemanticFunction("TextSkill",
"TextClassification", functionConfig);
```

我们知道一个 OpenAI 的项目具备多个技能吗，而多个技能具备不同的能力，，在 SK 你可以通过代码去描述技能和它具备的能力以及相关参数设置，也可以通过文件夹的方式去描述（通过文件夹方式，会在下一章介绍）。

8. 现在你可以设置输入来尝试看看是否能满足要求

```
var input = "今天广州天气怎么样?";
var classification = await kernel.RunAsync(input, classificationFunction);
```

```
Console.WriteLine(classification);
```

根据 Prompt 的要求，我要对问题分类，然后以 json 格式输出，结果还是挺满意的

```
{"question": "今天广州天气怎么样?", "label": "1"}
```

注意：示例代码在 ./Code/01.HowToQuestionClassification/cn/SKTextClassification.ipynb

本章小结

本章正式进入 Azure OpenAI Service 的应用学习，基于文本分类，我们用传统的 ML.NET , Azure OpenAI .NET SDK , 以及 Semantic Kernel .NET SDK 三种方法完成了相关的操作。对于开发者来说，可以体验传统的机器学习方式以及人工智能方式来完成文本分类的任务。希望大家能有所收获，期待你继续进入下一章的内容。

相关资料

1. 关于 ML.NET <https://dotnet.microsoft.com/en-us/apps/machinelearning-ai/ml-dotnet>
2. 关于 Semantic Kernel <https://github.com/microsoft/semantic-kernel>

第二章. 技能使用大全

使用生成式 AI 的时候，我们需要预先定下一些规则，这些规则可以包括回答的格式，语气，角色，或者业务相关的内容。这些规则，在生成式 AI 中我们称之为 Prompt。生成式 AI 兴起后，写 Prompt 的工程师非常吃香，通过 Prompt 引导 AI 生成需要的答案。本章中我们希望让大家掌握不同类型的 Prompt 的使用技巧以及通过 Semantic Kernel 管理好 Prompt。

2.1 技能介绍

在日常生活中需要完成某一方面专业的工作，必须有某一方面的能力。生成式 AI 是人工智能的进化，是完整的百科全书。你可以把它看作是一个人脑。写 Prompt 的时候实际上就是触发它使用各式各样的技能。例如读写，例如翻译，例如总结，例如管理等。可是在融入到行业的业务的场景时，这些技能就必须得到细化，例如写的技能，在行业应用中，就包括了归纳总结，文案输出，公文生成等与业务场景贴近的内容。而这些能力的整合就是一项技能，一个行业的应用需要具备各种技能。这些技能各式各样，除了生成式 AI 具备的技能外，我们还有行业专有的技能。下面来归纳一下

2.1.1 常规能力

通过简单描述就可以达到的常规能力，例如翻译，总结，或者提取内容都是生成式 AI 的拿手好戏。我们可以通过简单文字描述就可以完成，例如

"通过输入的内容，总结出100个字的内容，并且翻译成英文"

"帮我用诗词赞美一下广州"

"介绍煮 {{\\$input}} 的方法"

这些都是生成式 AI 模型带来的能力。写好 Prompt 描述好你需要的能力是很重要的，或者一个非计算机专业的学生比计算机专业的学生更适合做这个事情

2.1.2 时效性能力

生成式 AI 最大问题是时效性缺失，当然现在 GPT-4 也开始具备，但都是有限的。对于一些日常，新闻，政策，查询实时天气等都是时效性的问题。我们需要结合搜索引擎，以及企业内部数据库来配合赋予生成式 AI 时效性的能力

2.1.3 专业性能力

对于不同行业想用好生成式 AI，除了利用好强大的知识能力外，也需要结合行业内容来补充。例如问新员工的相关内容，GPT 模型就没有一个准确的答案，因为这个是每个企业唯一的。我们需要为好生成式 AI 添加额外的知识，这个时候更好的就是用回我们原有的知识例如非结构化数据的搜索，或者依赖于企业内部数据库的调用。这些都是我们必须完成的。我们可以引入我们过往人工智能技术的专业性知识。例如，非结构化的文档数据 / 存储在数据库的业务数据 / 相关业务流程 BI 数据 等。

2.2 Semantic Kernel 语义内核管理你的技能

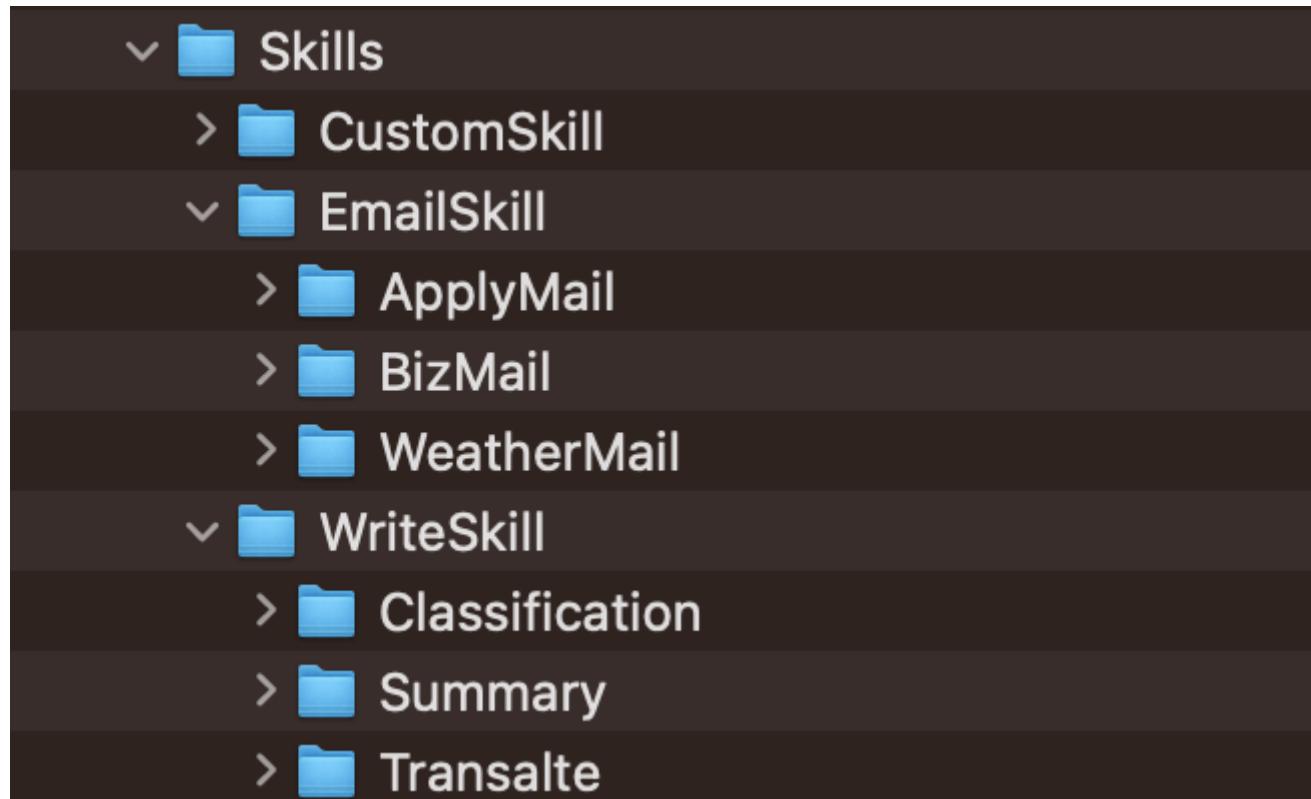
Semantic Kernel (简称 SK) 为我们提供了非常好的技能管理功能，开发者能针对业务通过 SK 更好组织不同的技能让生成式 AI 可以完成企业级的任务。

2.2.1 SK 的重要概念

Skills-技能

技能 - 生成式 AI 应用，都离不开技能，你可以理解为要让生成式 AI 完成任务提供的一些能力的组合。每一个任务可以有多个能力，例如文书的技巧就可以有写作，归纳，总结，修改等相关能力，一个技能下面可以有不同的能力。在 SK 中通过文件夹来管理不同的技能，一个技能就包含不同的能力。

在 SK 中定义技能是非常简单的，通过文件夹就可以定义不同的技能。每个技能下可以创建不同的子文件夹设定能力，如图所示：



我们有一个技能集的文件夹叫 Skills，里面有不同的技能，包括**CustomSkill** - 自定义技能，**EmailSkill** - 电子邮件技能，**WriteSkill** - 书写技能，通过子文件夹中表示。

每个技能都包含有不同的能力，都放在不同技能的子文件夹下

CustomSkill - 自定义技能(我们会在函数/方法中讨论)

EmailSkill - 电子邮件技能有申请相关的邮件 ApplyMail，有商业相关的邮件 BizMail，也有天气相关的邮件 WeatherMail的能力

WriteSkill - 书写技能有文本分类 Classification，归纳 Summary，翻译 Translate 等能力

而每个能力都有一个 Prompt 和对应的配置来描述，如我需要书写技能的分类能力，就包含一个 skprompt.txt 和 config.json (这是固定的格式，skprompt.txt 就是我们对应能力的 Prompt，config.json 就是 OpenAI 模型的参数)。

如分类的 Prompt，我的 skprompt.txt 描述是这样的

请帮我把 {{\\$input}} 进行类别确认，类别包括天气，课程，生成式，如果不太清楚，请回答没法确认，分类参考如下：

问：会下雨吗？ 类别：天气
问：今天温度？ 类别：天气
问：适度多少？ 类别：天气
问：什么是新能源车？ 类别： 课程
问：电动车的特点 类别： 课程
问：概念是什么？ 类别： 课程
问：课程相关的内容有哪些？ 类别： 课程
问：写一首诗歌？ 类别： 生成式
问：翻译一下 类别： 生成式
问：计算结果 类别： 生成式

如果能确认类别，天气相关请只输出 1， 课程相关请只输出 2， 生成式相关请只输出 3， 没法确认相关请只输出 0，并把{{\\$input}}和它的类别参考以下 json 格式输出

```
{"question": "{{\$input}}", "label": "{{\$label}}"}
```

而模型设置参数对应 config.json

```
{  
    "schema": 1,  
    "type": "completion",  
    "description": "文本分类",  
    "completion": {  
        "max_tokens": 60,  
        "presence_penalty": 0.0,  
        "frequency_penalty": 0.0  
    }  
}
```

如果我们希望通过 Semantic Kernel 去调用文书技能的文本分类能力，只需要通过 kernel.ImportSemanticSkillFromDirectory 引入技能集和相应技能（Semantic Kernel 的定义请回看第一章），然后直接根据需要的能力调用就可以了(别忘记引入 Microsoft.SemanticKernel.KernelExtensions)，代码如下

```
var skillsDirectory = System.IO.Directory.GetCurrentDirectory() +  
"/Skills";  
  
var write_skill =  
kernel.ImportSemanticSkillFromDirectory(skillsDirectory, "WriteSkill");
```

```
var questionLabel = await kernel.RunAsync("今天天气好吗",
write_skill["Classification"]);
```

如果你想了解更多请参考 ./Code/02.AddSkill/ImportNativeFunctionSkill.ipynb

Fucntions-函数/方法

我们通过函数去封装不同技能下的能力，把函数分成两种：

语意技能函数 - 通过封装 Prompt 提示语和模型设置来定义函数，这在第一章也已经提及我在这里就不多提了

原生技能函数 - 用于结合时效性技能和业务技能的函数封装，可以非常快速地定义不同的技能，如我需要查询一些入职相关的内容以及实时天气状况我们都可以通过原生函数定义这些技能赋予给系统答案，再让生成式 AI 完成下一步的工作，这是我们定义原生技能函数的一个例子

```
using Microsoft.SemanticKernel.SkillDefinition;
using Microsoft.SemanticKernel.Orchestration;
public class CompanySearchSkill
{
    [SKFunction("search employee infomation")]
    public string EmployeeSearch(string input)
    {
        return "欢迎了解社保相关内容";
    }

    [SKFunction("search weather")]
    public string WeatherSearch(string text)
    {
        return "欢迎搜索天气";
    }
}
```

通过 SKFunction 定义能力，提供定制化的技能是每个企业应用所必须的，我们通过这个内容可以增强应用的专业性。调用也是非常简单

```
CompanySearchSkill companySearchSkill = new CompanySearchSkill();

var customSkill = kernel.ImportSkill (companySearchSkill,
"CompanySearchSkill");
```

通过创建实例后，直接像其他技能一样引入就可以了，你可以理解为是用函数封装好的技能集。之后调用就可以根据问题去绑定不同的技能就可以了

```
var weatherOutput = await kernel.RunAsync("天气", customSkill["WeatherSearch"]);  
var employeeOutput = await kernel.RunAsync("社保如何购买", customSkill["EmployeeSearch"]);
```

如果你想了解更多请参考 [./Code/02.AddSkill/ImportNativeFunctionSkill.ipynb](#)

Plans-计划

如何判断完成一个事情，是需要按照一系列的计划来制定的。例如有一个任务“查找广州天气，把结果翻译成中文后根据天气情况生成穿衣提示，并结合天气结果和穿衣提示写一封邮件”。要完成这个任务，我们需要具备四个技能第一查找天气，第二翻译，第三生成穿衣提示，四就是写一封邮件。对于发出任务的人，需要明确指出每一步的工作，而对于开发者需要为完成这个任务提供能力。

在 Semantic Kernel 中，就有 Plan，让我们能根据目标，动态帮我们任务，能配对好不同的技能。我们赋予给 kernel 很多的技能，Plan 可以根据我们需要完成的目标去配对不同的能力帮我们完成各种任务，定义 Plan 非常简单

```
var planner = kernel.ImportSkill(new PlannerSkill(kernel));
```

像引入技能一样，引入自定义的 PlannerSkill 即可，在调用时直接调用就可以了，这个字段时固定的，

```
var plan = await kernel.RunAsync(context, planner["CreatePlan"]);
```

通过

```
plan.Variables.ToPlan().PlanString
```

可以非常清晰地看到要完成“查找广州天气，把结果翻译成中文后根据天气情况生成穿衣提示，并结合天气结果和穿衣提示写一封邮件”所需要的每一步的技能

```
<goal>
查找广州天气，把结果翻译成中文后根据天气情况生成穿衣提示，并结合天气结果和穿衣提示写一封邮件
</goal>
<plan>
    <function.CustomSkill.WeatherSearch input="Guangzhou"
setContextVariable="WEATHER_RESULT"/>
    <function.WriteSkill.Transalte input="$WEATHER_RESULT"
setContextVariable="TRANSLATED_RESULT"/>
    <function.WriteSkill.Tips input="$TRANSLATED_RESULT"
setContextVariable="TIPS_RESULT"/>
    <function.EmailSkill.WeatherMail input="$TRANSLATED_RESULT;$TIPS_RESULT"
setContextVariable="EMAIL_RESULT"/>
    <function._GLOBAL_FUNCTIONS_.BucketOutputs input="$EMAIL_RESULT"
bucketCount="2" bucketLabelPrefix="Result" />
</plan>
```

这也是一个工作的流程，针对一些复杂的任务，我们可以完成更多综合性的工作。以下是执行的逻辑代码

```
int step = 1;
int maxSteps = 30;
while (!executionResults.Variables.ToPlan().IsComplete && step < maxSteps)
{
    var results = await kernel.RunAsync(executionResults.Variables,
planner["ExecutePlan"]);
    if (results.Variables.ToPlan().IsSuccessful)
    {
        Console.WriteLine($"Step {step} - Execution results:\n");
        Console.WriteLine(results.Variables.ToPlan().PlanString);

        if (results.Variables.ToPlan().IsComplete)
        {
            Console.WriteLine($"Step {step} - COMPLETE!");
            Console.WriteLine(results.Variables.ToPlan().Result);
            break;
        }
    }
    else
    {
        Console.WriteLine($"Step {step} - Execution failed:");
        Console.WriteLine(results.Variables.ToPlan().Result);
        break;
    }

    executionResults = results;
    step++;
    Console.WriteLine("");
}
```

结果如图

```
Step 5 - Execution results:
```

```
<goal>
```

查找广州天气，把结果翻译成中文后根据天气情况生成穿衣提示，并结合天气结果和穿衣提示写一封邮件

```
</goal><plan>
```

```
</plan>
```

```
Step 5 - COMPLETE!
```

尊敬的同事：

您好！

根据最新的天气预报，广州的气温约为2度，有可能下雨。为了保暖，请您准备好毛衣、夹克衫、长裤等保暖衣物，以免受凉。

同时，请您随时留意出行状况，以便及时做出相应的调整。

祝您出行愉快！

此致

敬礼！

如果你想了解更多请参考 [./Code/02.AddSkill/ImportPlanSkill.ipynb](#)

2.3 本章小结

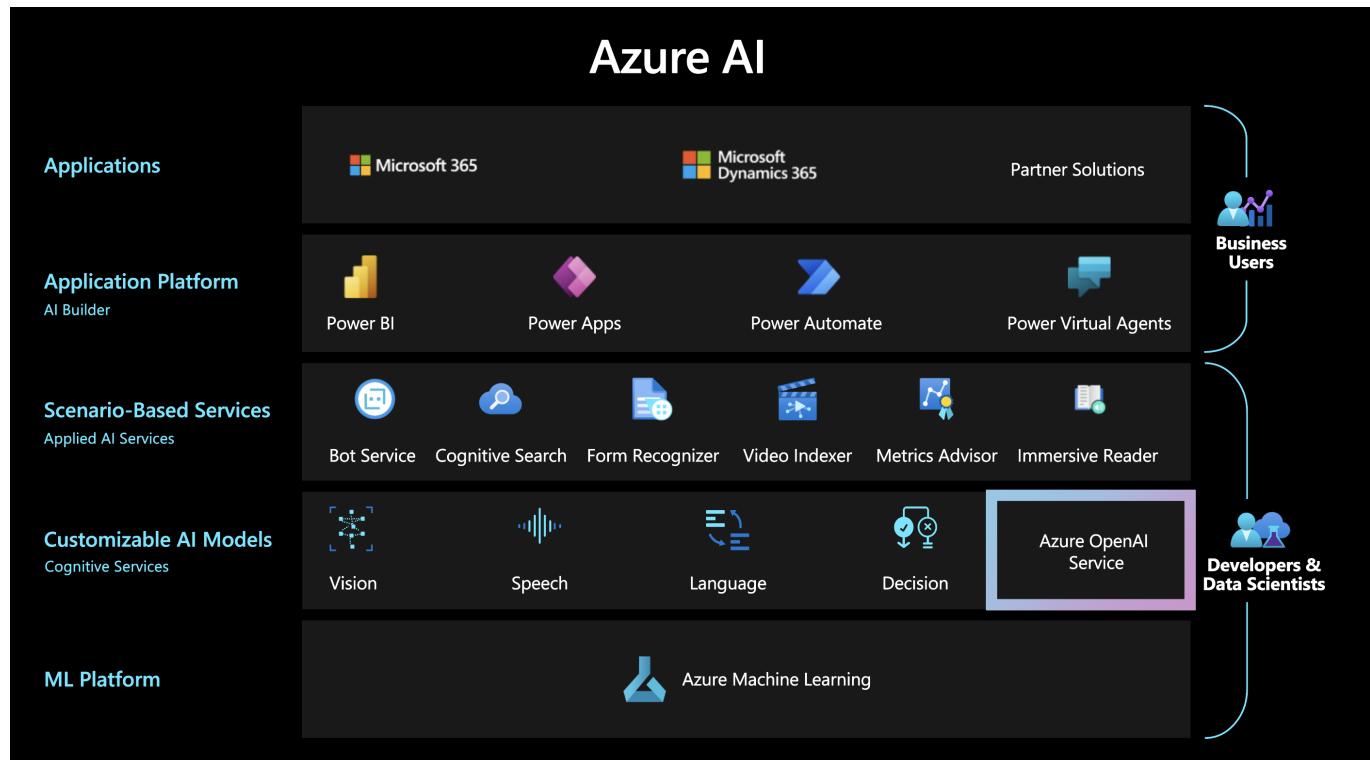
用好 Prompt 给到生成式 AI 能给到准确的答案。管理好 Prompt 的集合对于企业是非常重要的。通过 Semantic Kernel 我们可以快速地完成各种 Prompt 的整合，通过本章你可以学习管理不同的 Prompt，完成不同的企业流程。

相关资料

1. 关于 Semantic Kernel <https://github.com/microsoft/semantic-kernel>

第三章. 结合 Azure AI 扩展你的技能

在前两章，我们学会了如何连接 Azure OpenAI Service，以及认识如何通过 Semantic Kernel 管理和组织不同的技能。生成式 AI 的出现，让人工智能真正具备了智能的能力，可以基于不同的 Prompt 给出答案。这些答案能满足行业的需要了吗？例如我们需要让生成式 AI 告诉新员工入职的注意事项，这个时候生成式 AI 给到的往往是一些错误的答案，因为生成式 AI 并没有你所在企业的员工手册。又例如我们的一些实时新闻，例如我们和 ChatGPT 问 - OpenAI 是否支持插件开发？它也会给出一个错误的答案。那我们可以通过什么方法去组织一些语料去为生成式 AI 提供一些事实根据去做一个真实的解答呢？本章希望通过 Azure 原有的 AI 能力结合 OpenAI 构建人工智能应用解决方案。



Azure AI 有非常强大的人工智能功能，面向不同程度的人群，从开发层面有面向专业人工智能的科学家的 Azure 机器学习服务，有面向非专业人工智能开发人员的 Azure 认知服务，也有面向应用场景的聊天机器人，认知服务搜索，视频索引等，也有从商业应用的商业解决方案 Office 365 , Dynamics 365 ，以及低代码解决方案 Power Apps , Power BI , Power Automate 等。如果你希望打造基于云原生的人工智能企业级方案 Azure AI 绝对是一个最顶级的解决方案。

3.1 认识 Azure 认知服务



Azure 认知服务 (<https://azure.microsoft.com/zh-cn/products/cognitive-services/>) 使每个开发人员和数据科学家都能接触到 AI。借助领先模型，可以解锁各种用例。只需进行 API 调用，即可将查看、收听、朗读、搜索、理解和加速高级决策的功能嵌入到应用中。让所有技能级别的开发人员和数据科学家都能够轻松地向其应用添加 AI 功能。

在 Azure 上有不同的认知服务，主要集中在视觉和语音，语言，例如涉及语言内容的文本，知识，搜索以及实体提取都有，涉及视觉部分的分类，物体识别，还有语音部分的语音转文字，文字转语音的部分。

使用认知服务，你可以很快地使用微软已有模型进行开发，直接完成应用场景。特别当你的团队缺少机器学习 / 深度学习的工程师是相当有用的。

有人会说生成式 AI 的出现，直接取代了认知服务，实际上他们是可以互补的。我们可以结合 Azure 认知服务，完成多场景的人工智能应用。Azure OpenAI Service 已经添加进 Azure 认知服务中，结合微软不同的 AI 产品，可以打造一个完全 AI 的解决方案。

3.1.1 开通 Azure 认知服务

使用 Azure 认识服务很简单，以下是相关步骤

1. 你必须有 Azure 账号(如果你没有 Azure 账号可以通过 <https://azure.com/free> 注册，如果你是学生可以通过 <https://aka.ms/studentgetazure> 免信用卡注册)
2. 打开你的 Azure 门户，创建资源，点击 AI + Machine Learning 选择 Azure Cognitive Service

Create a resource

Get Started ...

Recently created

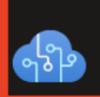
Popular Azure services See more in All services

Categories

- [AI + Machine Learning](#)
- [Analytics](#)
- [Blockchain](#)

 Search services and marketplace

 Azure Synapse Analytics
Create | Docs | MS Learn

 Cognitive Services
Create | Docs | MS Learn

认知服务

3. 选择订阅并填写资源组，名字以及所在区域，就可以直接创建了

Create Cognitive Services

Basics Network Identity Tags Review + create

Get access to Vision, Language, Search, and Speech Cognitive Services with a single API key. Quickly connect services together to achieve more insights into your content and easily integrate with other services like Azure Search.

[Learn more](#)

Project Details

Subscription * ⓘ

Windows Azure MSDN - Visual Studio Ultimate

Resource group * ⓘ

AIGroup

[Create new](#)

Instance Details

Region ⓘ

South Central US

Name * ⓘ

kinfey-azurecognitiveservice



Location specifies the region only for included regional services. This does not specify a region for included non-regional services. Click here for more details.

Pricing tier * ⓘ

Standard S0

[View full pricing details](#)

By checking this box I acknowledge that I

[Review + create](#)

[< Previous](#)

[Next : Network >](#)

注意：Azure 认识服务是一个综合服务，你可以通过该服务完成视觉，语言，文本，决策等不同的功能，你也可以在 Azure 创建单一场景的认知服务，如

All services > Create a resource >

Create Language Understanding

Basics Network Tags Review + create

Language understanding (LUIS) is a natural language processing service that enables you to build your own custom model to understand human language programmatically or through the UI in the LUIS portal. After you are satisfied with your LUIS model, you publish it and query its prediction endpoint through your client application for an end to end conversational flow. To build, manage, train, test and publish your LUIS Model, you will need to create the below Authoring Resource. This also gives you 1,000 requests/month endpoint requests. If you want your client app to request beyond the 1,000 requests provided by the authoring, create the below Prediction Resource. If you know from the start you will be needing more than 1000 prediction requests as well as the authoring experience, create using the "Both" option. This will create two resources, one for each type.

Create options *

- Both
- Authoring
- Prediction

Project Details

Subscription * ⓘ

Windows Azure MSDN - Visual Studio Ultimate



Resource group * ⓘ

Create new



3.1.2 在 .NET 中使用 Azure 认知服务

Azure 认知服务通过 REST API 给到不同的编程语言调用。对于 .NET，你可以使用不同场景的 SDK 或者通过 HttpClient 直接调用来使用不同功能的认知服务。

如我需要使用文本服务，检测文本的语言，就可以通过认知服务的 .NET SDK Azure.AI.TextAnalytics 去调用，步骤如下：

1. 创建一个.ipynb 的 Notebook 文件

```
#r "nuget: Azure.AI.TextAnalytics"
```

2. 引入相关的命名空间

```
using Azure;
using System;
using Azure.AI.TextAnalytics;
using System.Collections.Generic;
```

3. 获取 Azure 认知服务的 Key 和 Endpoint

Home > lukaoicognitiveservice

lukaoicognitiveservice | Keys and Endpoint

Cognitive services multi-service account

Search Regenerate Key1 Regenerate Key2

Overview Activity log Access control (IAM) Tags Diagnose and solve problems

Resource Management

- Keys and Endpoint** (highlighted)
- Pricing tier Networking Identity Cost analysis Properties Locks

Show Keys

KEY 1 (Copied)

KEY 2

Location/Region (southcentralus)

Endpoint

4. 用变量设置 Key 和 Endpoint

```
string endpoint = "Your Endpoint";
string apiKey = "Your API Key";
```

5. 创建 TextAnalyticsClient 对象，绑定

```
var client = new TextAnalyticsClient(new Uri(endpoint), new
AzureKeyCredential(apiKey));
```

6. 加入字符串

```
string document =
"Este documento está escrito en un lenguaje diferente al inglés. Su
objetivo es demostrar cómo
+ invocar el método de Detección de Lenguaje del servicio de Text
Analytics en Microsoft Azure."
+ También muestra cómo acceder a la información retornada por el
servicio. Esta funcionalidad es"
+ útil para los sistemas de contenido que recopilan texto
```

```
arbitrario, donde el lenguaje no se conoce"  
+ " de antemano. Puede usarse para detectar una amplia gama de  
lenguajes, variantes, dialectos y"  
+ " algunos idiomas regionales o culturales.";
```

7. 完成相关判断函数的调用

```
Response<DetectedLanguage> response = client.DetectLanguage(document);  
DetectedLanguage language = response.Value;
```

8. 查看结果

```
language.Name
```

具体执行你可以参考 ./Code/03.ImportAzureAI/AzureCognitiveServiceForText.ipynb

3.2 必应搜索的检索能力

你有在用必应搜索吗？通过必应可以搜索不同的新闻资料，图片，视频，文档等内容。通过 Azure 可以使用必应搜索的 API。使用必应搜索 API，可以生成联网应用和服务，用于查找网页、图像、新闻、位置以及其他不含广告的内容。通过使用必应搜索 REST API 或 SDK 发送搜索请求，可以获取 Web 搜索的相关信息和内容。在生成式 AI 的场景中，针对一些时效性的内容缺失，我们可以通过必应搜索 API 来提供数据。

你可以通过 Azure Portal 创建必应搜索的 API 服务 - Bing Search V7

Home > Create a resource > Marketplace >

Bing Search v7

Microsoft

Bing Search v7 Microsoft | Azure Service ★ 4.0 (2 ratings)

Plan: Bing Search v7 [Create](#)

[Overview](#) [Plans](#) [Usage Information + Support](#) [Ratings + Reviews](#)

The Bing Search APIs v7 adds intelligent search to your app, combining hundreds of billions of webpages, images, videos, and news to provide relevant results with no ad requirements. The results can be automatically customized to your users' locations or markets, increasing relevancy by staying local. Responses from the Bing Search APIs rank page results, including news, dictionary, computation, and time. Safe search levels are customizable for your users, keeping them from adult content, if required. Finally, the APIs return spelling suggestions for misspelled queries, and lists of related searches. Bring the power of Bing Search to your app today.

Legal Notice

Microsoft will use data you send to Bing Search Services to improve Microsoft products and services. Where you send personal data to this service, you are responsible for obtaining sufficient consent from the data subjects. The Data Protection Terms in the Online Services Terms do not apply to Bing Search Services.

Please refer to the [Search Services Terms](#) for details.

通过选择订阅，以及设定资源组，名字，以及价格，点击创建后就可以使用了

Home > Create a resource > Marketplace > Bing Search v7 >

Create a Bing search resource

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * ⓘ Windows Azure MSDN - Visual Studio Ultimate

Resource group * ⓘ AIGroup [Create new](#)

Instance details

Name * kinfeysearch

Region Global

i This resource is a global resource that works across Azure regions.

Pricing tier * F1 (3 Calls per second, 1k Calls per month)

[View full pricing tier details](#)

Terms

Microsoft will use data you send to Bing Search Services to improve Microsoft products and services. Where you send personal data to this service, you are responsible for obtaining sufficient consent from the data subjects. The General Privacy and Security Terms in the Online Services Terms do not apply to this service.

[Learn more](#)

I confirm I have read and understood the notice above. *

[Review + create](#) [< Previous](#) [Next : Tags >](#)

调用必应搜索的 API 非常简单，直接通过 HttpClient 调用 API 节点就可以了

具体执行你可以参考 ./Code/03.ImportAzureAI/BingForNews.ipynb

3.3 利用 Azure Cognitive Search 管理非结构文档

3.3.1 认识 Azure Cognitive Search

Azure Cognitive Search（以前称为“Azure 搜索”）是一个云搜索服务，它为开发人员提供基础结构、API 和工具，用于基于 Web、移动和企业应用程序中的专用异类内容构建丰富的搜索体验。

搜索是将文本呈现给用户的应用的基础，其中常见的场景包括目录或文档搜索，在线零售应用程序，基于专属内容进行数据探索。创建搜索服务时，将使用以下功能：

1. 一个搜索引擎，用于根据包含用户自有内容的搜索索引进行全文搜索
2. 丰富索引编制、词汇分析以及用于提取和转换内容的可选 AI 扩充
3. 用于文本搜索、模糊搜索、自动完成、地理搜索等的丰富查询语法
4. 通过 REST API 和 Azure SDK 中的客户端库实现的可编程性
5. 数据层、机器学习层和 AI（认知服务）级别的 Azure 集成

Azure Cognitive Search 对于非结构文档的搜索有非常好的效果，通过它可以引入更多的非结构化数据作为预料引入到生成式 AI 模型场景，构造一个企业级应用的生成式 AI 方案。

从体系结构方面来讲，搜索服务位于外部数据存储（包含未编入索引的数据）与客户端应用（向搜索索引发送查询请求并处理响应）之间。

在客户端应用中，搜索体验使用 Azure 认知搜索中的 API 定义，可能包括相关性调整、语义排名、自动完成、同义词匹配、模糊匹配、模式匹配、筛选和排序。

在整个 Azure 平台上，认知搜索可以以“索引器”（自动从 Azure 数据源引入/检索数据）和“技能组”（引入认知服务（例如图像和自然语言处理）中的可消耗 AI，或者引入你在 Azure 机器学习中创建的或在 Azure Functions 内包装的自定义 AI）的形式与其他 Azure 服务集成。

如果你希望了解更多 Azure Cognitive Search 的搜索功能，可以查看 <https://learn.microsoft.com/zh-cn/azure/search/>

3.3.2 构建 Azure Cognitive Search

开启 Azure Cognitive Search 的步骤相对之前的步骤比较复杂，我们可以把它分成两部分进行处理

第一部分：在 Azure 上开启 Azure Cognitive Search

1. 打开 Azure 门户，选择创建资源，选择 Azure Cognitive Search 进行创建

Azure Cognitive Search

Microsoft | Azure Service

★ 3.8 (172 ratings)

Plan

Azure Cognitive Search ▼ Create

Overview Plans Usage Information + Support Ratings + Reviews

AI-powered cloud search service for mobile and web app development

Azure Cognitive Search (formerly Azure Search) is the only cloud search service with built-in artificial intelligence (AI) capabilities that enrich all types of information to easily identify and explore relevant content at scale. It uses the same integrated Microsoft natural language stack that Bing and Office have used for more than a decade, and prebuilt AI APIs across vision, language, and speech.

Azure Cognitive Search Features:

- Fully managed search as a service to reduce complexity and scale easily
- Auto-complete, geospatial search, filtering, and faceting capabilities for a rich user experience
- Built-in AI capabilities including OCR, key phrase extraction, and named entity recognition to unlock insights
- Flexible integration of custom models, classifiers, and rankers to fit your domain-specific needs

2. 选择 Azure 订阅，并选择资源组和创建文件名等，这里建议使用至少标准的价格模式，因为需要使用语义搜索这是最低配置

Create a search service

Basics Scale Networking Tags Review + create

Project Details

Subscription * Windows Azure MSDN - Visual Studio Ultimate

Resource Group * Create new

Instance Details

Service name * Enter service name

Location * West US 2

Pricing tier * Standard
25 GB/Partition*, max 12 replicas, max 12 partitions, max 36 search units
[Change Pricing Tier](#)

第二部分：在 Azure Cognitive Search 构建一个非结构化文档搜索

通过 Azure Cognitive Search 可以检索我们的非结构化文档，但这个需要结合我们不同的 Azure 功能，包括 Azure Blob Storage 和 Azure Cognitive Service。

1. 配置 Azure Blob Storage

选择 Azure 门户，点击存储账户



创建账户，除了资源组和名字，区域外，性能选择标准

This screenshot shows the "Create a storage account" wizard in the Azure portal. The "Basics" tab is selected. The "Subscription" dropdown is set to "Visual Studio Enterprise Subscription". The "Resource group" dropdown is set to "AIGroup" with an option to "Create new". The "Instance details" section includes fields for "Storage account name" (with a placeholder "myaccountname"), "Region" (set to "(US) East US"), and "Performance" (set to "Standard"). The "Redundancy" dropdown is set to "Geo-redundant storage (GRS)". At the bottom, there are buttons for "Review" and "Next : Advanced >".

Home > Storage accounts >

Create a storage account

Basics Advanced Networking Data protection Encryption Tags Review

Select the subscription in which to create the new storage account. Choose a new or existing resource group to organize and manage your storage account together with other resources.

Subscription *

Visual Studio Enterprise Subscription

Resource group *

AIGroup

Create new

Instance details

If you need to create a legacy storage account type, please click [here](#).

Storage account name ⓘ *

(US) East US

Deploy to an edge zone

Performance ⓘ *

Standard: Recommended for most scenarios (general-purpose v2 account)

Premium: Recommended for scenarios that require low latency.

Redundancy ⓘ *

Geo-redundant storage (GRS)

Review < Previous Next : Advanced >

创建成功后，进入新创建的存储账户，选择容器

kinfeyoaistorage Storage account

Overview

Resource group (move) AIGroup

Location South Central US

Primary/Secondary Location Primary: South Central US, Secondary: North Central US

Subscription (move) Windows Azure MSDN - Visual Studio Ultimate

Subscription ID 3579bf17-29da-4b2c-b302-55d72e92a513

Disk state Primary: Available, Secondary: Available

Tags (edit)

添加存储容器

kinfeyoaistorage | Containers

Storage account

Overview

Activity log

Tags

Diagnose and solve problems

Access Control (IAM)

Data migration

Events

Storage browser

Data storage

Containers

New container

Name **demodemo**

Public access level Container (anonymous read access for containers and blobs)

All container and blob data can be read by anonymous request. Clients can enumerate blobs within the container by anonymous request, but cannot enumerate containers within the storage account.

选择创建好的容器，上传 ./Code/03.ImportAzureAI/data 中的 pdf 文件到容器中

Home > Storage accounts > kinfeyoaistorage | Containers >

oaistorage Container

Upload

Authentication method: Access key (Switch to Azure AD User Account)

Location: oaistorage

Search blobs by prefix (case-sensitive)

Show deleted blobs

Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
011.pdf	4/15/2023, 9:50:48 AM	Hot (Inferred)		Block blob	111.66 KiB	Available
012.pdf	4/15/2023, 9:50:48 AM	Hot (Inferred)		Block blob	92.16 KiB	Available

2. 配置 Azure Cognitive Service 请参考本章的 3.1.1 开通 Azure 认知服务

3. 在 Azure Cognitive Search 中选择导入数据，绑定 Azure Blob Storage 并设置索引

The screenshot shows the Azure Cognitive Search service 'lukaoisearch'. The 'Import data' button in the top navigation bar is highlighted with a red box. The main pane displays service details like Resource group (AIGroup), Location (South Central US), and Status (Running). A 'Tags' section is also visible.

选择存在数据，完成 Azure Blob Storage 绑定

The screenshot shows the 'Import data' configuration page. The 'Data Source' dropdown is set to 'Existing data source'. A table lists a single data source named 'kinfeyaoiodocdata' of type 'Azure Blob Storage' under the table name 'oainstorage'.

注意: 因为需要设置语音搜索，所以必须绑定 Azure Cognitive Service

The screenshot shows the 'Attach Cognitive Services' configuration page. It lists a cognitive service resource named 'lukaoicognitiveservice' attached to the 'southcentralus' region. A note at the bottom states that the REGION in the table specifies the region only for included regional services.

设置搜索技能，具体字段设置如下，并设置名字

The screenshot shows the 'Skillset' configuration page. It includes fields for Skillset name ('azureblob-skillset'), Source data field ('metadata_storage_content_md5'), and various enrichment options like 'Enable incremental enrichment' and 'Indexer cache location'. The 'Field name' column lists 'people', 'organizations', 'locations', 'keyphrases', and 'language' corresponding to the checked cognitive skills.

索引设置如下

Field name	Type	Retrievable	Filterable	Sortable	Facetable	Searchable	Analyzer
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
content	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Standa... ▾
metadata_storage_cc	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Standa... ▾
metadata_storage_si	Int64	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
metadata_storage_la	DateTimeOffset	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
metadata_storage_cc	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Standa... ▾
metadata_storage_na	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
metadata_storage_pa	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
metadata_storage_fil	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
metadata_content_ty	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Standa... ▾
metadata_language	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
metadata_creation_d	DateTimeOffset	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
language	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Standa... ▾
keyphrases	StringCollection	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Standa... ▾
locations	StringCollection	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Standa... ▾

完成后继续添加 indexer 的名字并创建即可

注意： 如果你希望有了解更多的内容，建议你参考该链接 <https://learn.microsoft.com/zh-cn/azure/search/search-create-service-portal>

配置完成后在创建的 Azure Cognitive Service 添加 Semantic search , 选择 Free 就可以了

Home > lukaoisearch

lukaoisearch | Semantic search (Preview)

Search service

Search

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Semantic search (Preview)

Semantic search uses deep neural networks to provide relevant results and answers based on semantics, not just lexical analysis. Additional charges may be applicable.

Availability

Semantic search requires the search service to be on Standard tier in the following regions: Australia East, Canada Central, East US, East US 2, North Central US, Southeast Asia, South Central US, West US, West US 2, North Europe, UK South, West Europe.

Settings

Semantic search (Preview)

Knowledge Center

Keys

Scale

Search traffic analytics

Identity

Networking

Properties

Standard

250,000 requests per month
\$2.00 per 1,000 additional requests.

\$499.72/month

Select Plan

Learn more

这样就配置好 Azure Cognitive Search 。

3.3.3 验证创建的 Azure Cognitive Search

我们可以尝试验证刚创建好的 Azure Cognitive Search，步骤如下

1. 在 Azure Cognitive Search 中选择搜索浏览

The screenshot shows the Azure Cognitive Search service dashboard for the 'lukaoisearch' service. The top navigation bar includes buttons for 'Add index', 'Import data', 'Search explorer' (which is highlighted with a red box), 'Refresh', 'Delete', and 'Move'. The left sidebar contains links for 'Overview', 'Activity log', 'Access control (IAM)', 'Tags', 'Diagnose and solve problems', 'Settings', 'Semantic search (Preview)', 'Knowledge Center', 'Keys', 'Scale', 'Search traffic analytics', 'Identity', and 'Networking'. The main content area displays service details such as Resource group, Location, Subscription, Status, and Tags. A banner at the bottom right says 'Build a full-text search experience with AI and semantic search'.

通过搜索浏览就可以测试检索的结果，并可设置一些相关的参数

2. 语义全文检索，需要配置一些相关内容，需要开启语义搜索

The screenshot shows the 'Search explorer' interface for the 'lukaoisearch' service. The left pane displays search parameters like 'Index' (set to 'kinfeyazureblob-index'), 'Query string' (with examples), 'Query options (Preview)', 'Semantic search (Preview)', 'Spell correction (Preview)', and 'Request URL'. The right pane is a modal dialog titled 'New semantic configuration' for creating a semantic configuration. It includes fields for 'Name' (set to 'demosearch'), 'Title field' (set to 'metadata_storage_content_md5'), 'Content fields' (list: 'metadata_storage_content_md5', 'content', 'metadata_storage_name'), and 'Keyword fields' (list: 'metadata_storage_content_md5', 'keyphrases'). At the bottom of the dialog are 'Save' and 'Cancel' buttons.

3. 我们可以做一些测试，例如，新能源车的发展

The screenshot shows the Azure Search Explorer interface. At the top, there's a dropdown for the index set to 'kinfeyazureblob-index'. Below it is a 'Query string' input field containing '新能源车的发展'. To the right of the input field are buttons for 'API version' (set to '2021-04-30-Preview') and 'Search'. Underneath the search bar, there's a section titled 'Query options (Preview)' with a 'Query language' dropdown set to 'American English (en-US)'. Below this, there are sections for 'Semantic search (Preview)', 'Spell correction (Preview)', and 'Request URL'. The 'Request URL' field contains the full API endpoint for the search query. On the right side of the interface, there's a preview pane showing the JSON results of the search query.

3.3.4 用 .NET 调用 Azure Cognitive Search

你可以参考 [./Code/03.ImportAzureAI/AzureCognitiveSearchDemo.ipynb](#) 来了解具体的实现情况

3.4 本章小结

在生成式 AI 风行的年代，原有的 AI 技术不是被取代，而是会继续沿用补充生成式 AI 的不足。通过云原生的 AI 技术，可以整合更多的场景。本章为大家介绍了在 Azure 上的 AI 技术，希望大家可以结合生成式的 AI 完成更专业的行业应用。

相关资料

1. 关于 Azure Cognitive Services <https://learn.microsoft.com/en-us/azure/cognitive-services/>
2. 关于 Azure Bing Search API <https://learn.microsoft.com/en-us/azure/cognitive-services/bing-web-search/>
3. 关于 Azure Cognitive Search <https://learn.microsoft.com/zh-cn/azure/search/>

第四章. 通过语义嵌入 Embeddings 来调教你的 OpenAI 模型

很多行业希望拥有 OpenAI 的能力，希望 OpenAI 能解决自己的企业内部问题。这就包括员工相关的内容如入职须知，请假和报销流程，还有福利查询等，企业业务流相关的内容包括相关文档，法规，执行流程等，也有一些面向客户的查询。虽然 OpenAI 有强大的知识能力，但是基于行业的数据和知识是没办法获取的。那如何注入这些基于行业的知识内容呢？这也是让 OpenAI 迈入企业化重要的一步。本章我们就会和大家讲讲如何注入行业的数据和知识，让 OpenAI 变得更专业。

4.1. 从自然语言中的向量谈起

在自然语言领域，我们知道最细的粒度是词，词组成句，句构成段落，篇章和最后的文档。计算机是不认识词的，所以我们需要对词转换为数学上的表示。这个表示就是向量，也就是 Vector。向量是一个数学上的概念，它是一个有方向的量，有大小和方向。有了向量，我们可以有效地对文本进行向量化，这也是计算机自然语言领域的基础。在自然语言处理领域，我们有很多向量化的方法，比如 One-hot, TF-IDF, Word2Vec, Glove, ELMO, GPT, BERT 等。这些向量化的方法都有各自的优缺点，但是都是基于词的向量化，也就是词向量。词向量是自然语言处理的基础，也是 OpenAI 所有模型的基础。我们分别看看词向量里面的几种常见方法。

4.1.1 One-hot 编码

One-hot 编码，是用 0 和 1 的编码方式来表示词。比如我们有 4 个词，分别是：我，爱，北京，天安门。那么我们可以用 4 个向量来表示这 4 个词，分别是：

```
我 = [1, 0, 0, 0]
爱 = [0, 1, 0, 0]
北京 = [0, 0, 1, 0]
天安门 = [0, 0, 0, 1]
```

在传统的自然语言应用场景中，我们把每个词看成用 One-Hot 向量表示，作为唯一的离散符号。我们的词库中有单词的数量就是向量的维度，如上述的例子总共包含了四个词，所以我们可以用一个四维的向量来表示。在这个向量中，每个词都是唯一的，也就是说每个词都是独立的，没有任何关系。这样的向量我们称为 One-Hot 向量。One-Hot 向量的优点是简单，容易理解，而且每个词都是唯一的，没有任何关系。但是 One-Hot 向量的缺点也很明显，就是向量的维度会随着词的增加而增加。比如我们有 1000 个词，那么我们的向量就是 1000 维的。这样的向量是非常稀疏的，也就是大部分的值都是 0。这样的向量会导致计算机的计算量非常大，而且也不利于计算机的计算。所以 One-Hot 编码的缺点就是向量维度大，计算量大，计算效率低。

4.1.2. TF-IDF 编码

TF-IDF 是一个统计学，通过评估一个词对一个语料的重要程度。TF-IDF 是 Term Frequency - Inverse Document Frequency 的缩写，中文叫做词频-逆文档频率。TF-IDF 的主要思想是：如果某个词在一篇文章中

出现的频率高，并且在其他文章中很少出现，那么这个词就是这篇文章的关键词。一般我们习惯把这个概念拆分，分为 TF 和 IDF 两个部分。

TF - 词频

词频（Term Frequency）指的是某个词在文章中出现的频率。词频的计算公式如下：

$$TF = \text{某个词在文章中出现的次数} / \text{文章的总词数}$$

TF 有一个问题，就是如果一个词在文章中出现的次数很多，那么这个词的 TF 值就会很大。这样的话，我们就会认为这个词是这篇文章的关键词。但是这样的话，我们会发现，很多词都是这篇文章的关键词，这样的话，我们就无法区分哪些词是这篇文章的关键词了。所以我们需要对 TF 进行一些调整，这个调整就是 IDF。

IDF - 逆文档频率

逆文档频率（Inverse Document Frequency）指的是某个词在所有文章中出现的频率。逆文档频率的计算公式如下：

$$IDF = \log(\text{语料库的文档总数} / (\text{包含该词的文档数} + 1))$$

IDF 的计算公式中，分母加 1 是为了避免分母为 0 的情况。IDF 的计算公式中，语料库的文档总数是固定的，所以我们只需要计算包含该词的文档数就可以了。如果一个词在很多文章中都出现，那么这个词的 IDF 值就会很小。如果一个词在很少的文章中出现，那么这个词的 IDF 值就会很大。这样的话，我们就可以通过 TF 和 IDF 的乘积来计算一个词的 TF-IDF 值。TF-IDF 的计算公式如下：

$$TF-IDF = TF * IDF$$

TF-IDF 经常用于文本分类的场景，这也是 TF-IDF 最常用的场景。TF-IDF 的优点是简单，容易理解，而且计算量也不大。TF-IDF 的缺点是没有考虑词的顺序，而且没有考虑词与词之间的关系。所以 TF-IDF 适合用于文本分类的场景，而不适合用于文本生成的场景。

4.1.3. Word2Vec 编码

Word2Vec 我们也叫它做 Word Embeddings，中文叫做词嵌入。Word2Vec 的主要思想是：一个词的语义可以通过它的上下文来确定。Word2Vec 有两种模型，分别是 CBOW 和 Skip-Gram。CBOW 是 Continuous Bag-of-Words 的缩写，中文叫做连续词袋模型。Skip-Gram 是 Skip-Gram Model 的缩写，中文叫做跳字模型。CBOW 模型的思想是通过一个词的上下文来预测这个词。Skip-Gram 模型的思想是通过一个词来预测这个词的上下文。Word2Vec 的优点是可以得到词的语义，而且可以得到词与词之间的关系。对比起 One-Hot

编码和 TF-IDF 编码，Word2Vec 编码的优点是可以得到词的语义，而且可以得到词与词之间的关系。Word2Vec 编码的缺点是计算量大，而且需要大量的语料库。

之前我们提及过，One-Hot 编码的维度是词的个数，而 Word2Vec 编码的维度是可以指定的。一般我们会指定为 100 维或者 300 维。Word2Vec 编码的维度越高，词与词之间的关系就越丰富，但是计算量也就越大。Word2Vec 编码的维度越低，词与词之间的关系就越简单，但是计算量也就越小。Word2Vec 编码的维度一般是 100 维或者 300 维，这样的维度可以满足大部分的应用场景。

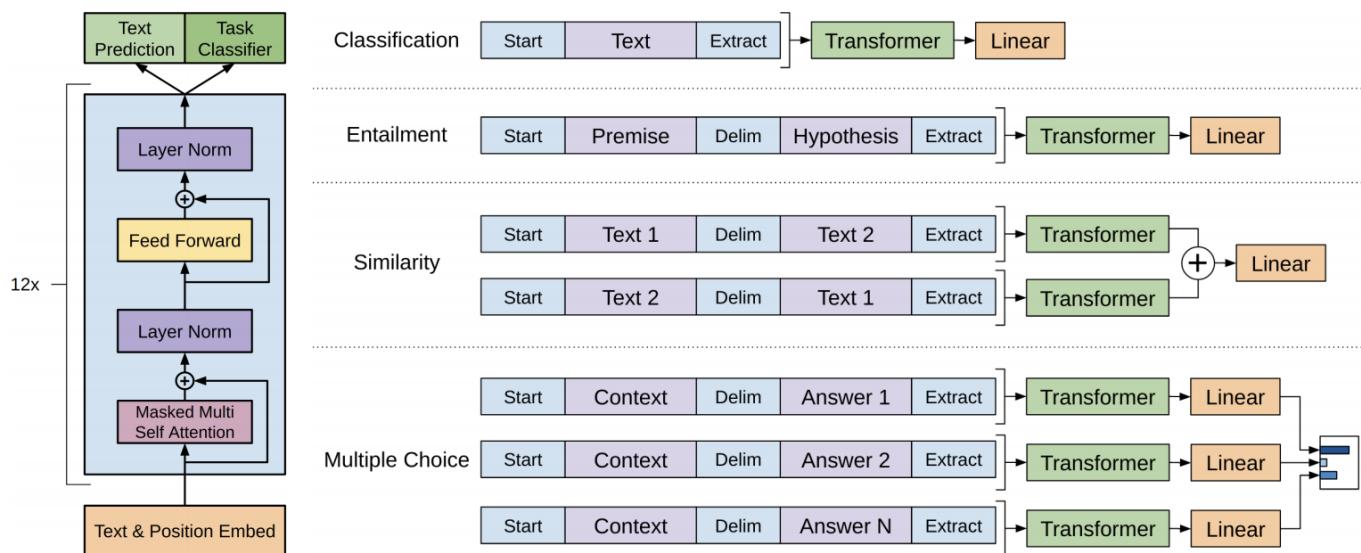
Word2Vec 编码的计算公式非常简单，就是 Word Embeddings。Word Embeddings 是一个词向量，它的维度是可以指定的。Word Embeddings 的维度一般是 100 维或者 300 维，这样的维度可以满足大部分的应用场景。Word Embeddings 的计算公式如下：

$$\text{Word Embeddings} = \text{词的语义} + \text{词与词之间的关系}$$

可以把 Word2Vec 看作是简化的神经网络。

4.1.4. GPT 模型

GPT 模型的全称是 Generative Pre-Training，中文叫做预训练生成模型。GPT 模型是 OpenAI 在 2018 年提出的，它的主要思想是：一个词的语义可以通过它的上下文来确定。GPT 模型的优点是可以得到词的语义，而且可以得到词与词之间的关系。GPT 模型的缺点是计算量大，而且需要大量的语料库。GPT 模型的结构是一个多层单向的 Transformer 结构，它的结构如下图所示：



训练过程是两个阶段，第一个阶段是预训练，第二个阶段是微调。预训练的语料是维基百科和 BookCorpus，微调的语料是不同的自然语言任务。预训练的目标是通过一个词的上下文来预测这个词，微调的目标是根据不同的自然语言任务，如文本分类、文本生成、问答系统等，对语义模型进行微调，得到不同的模型。

GPT 模型已经经历了 4 个阶段，最为出名的就是 ChatGPT 所使用的 GPT-3.5 以及 GPT 4。GPT 开启了全新的时代，它的出现让我们看到了自然语言处理的无限可能。GPT 模型的优点是可以得到词的语义，而且可以

得到词与词之间的关系。GPT 模型的缺点是计算量大，而且需要大量的语料库。很多人希望拥有自己行业对标的 GPT，这也是我们本章需要解决的问题。

4.1.5. BERT 编码

BERT 是 Bidirectional Encoder Representations from Transformers 的缩写，中文叫做双向编码器的 Transformer。BERT 是一个预训练的模型，它的训练语料是维基百科和 BookCorpus。BERT 的主要思想是：一个词的语义可以通过它的上下文来确定。BERT 的优点是可以得到词的语义，而且可以得到词与词之间的关系。对比起 One-Hot 编码、TF-IDF 编码和 Word2Vec 编码，BERT 编码的优点是可以得到词的语义，而且可以得到词与词之间的关系。BERT 编码的缺点是计算量大，而且需要大量的语料库。

4.2 Embeddings 嵌入技术

在第一节中我们提及了 One-Hot 编码、TF-IDF 编码、Word2Vec 编码、BERT 编码和 GPT 模型。这些编码和模型都是 Embeddings 嵌入技术的一种。Embeddings 嵌入技术的主要思想是：一个词的语义可以通过它的上下文来确定。Embeddings 嵌入技术的优点是可以得到词的语义，而且可以得到词与词之间的关系。Embeddings 是作为自然语言深度学习的基础，它的出现让我们看到了自然语言处理的无限可能。

对于文本内容的 Embeddings 方法，我们结合上一节，你会发现从 word2vec 技术诞生后，文本内容的 Embeddings 就不断得到加强，从 word2vec 到 GPT 再到 BERT ,Embeddings 技术的效果越来越好。 Embeddings 技术的本质就是“压缩”，用更少的维度来表示更多的信息。这样的好处是可以节省存储空间，提高计算效率。

在 OpenAI 中，Embeddings 技术的应用非常广泛，将文本字符串转换为浮点向量，通过向量之间的距离来衡量文本之间的相似度。不同行业希望加入自己的数据 我们就可以把这些企业级的数据通过 OpenAI Embeddings - text-embedding-ada-002 模型查询出向量，并通过映射进行保存，在使用时将问题也转换为向量，通过相似度的算法对比，找出最接近的 TopN 结果，从而找到与问题相关联的企业内容。

我们可以通过向量数据库将企业数据向量化后保存，结合 text-embedding-ada-002模型通过向量的相似度进行查询，从而找到与问题相关联的企业内容。现在常用的向量数据库就包括 Qdrant, Milvus, Faiss, Annoy, NMSLIB 等。

4.2.1 Open AI 的 Embeddings 模型

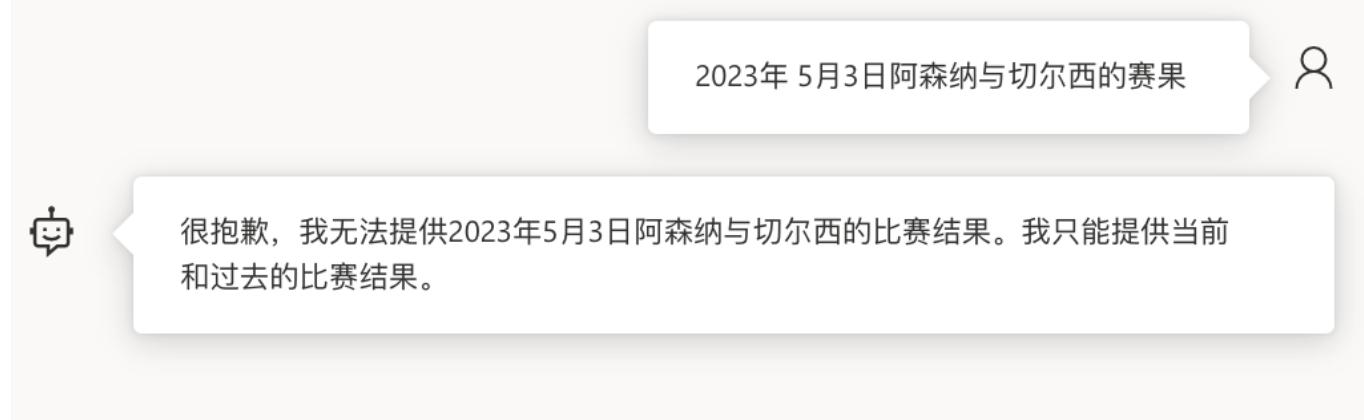
OpenAI 的文本嵌入向量文本字符串的相关性。 嵌入通常用于以下场景

1. 搜索（结果按与查询字符串的相关性排序）
2. 聚类（其中文本字符串按相似性分组）
3. 推荐（推荐具有相关文本字符串的项目）
4. 异常检测（识别出相关性很小的异常值）
5. 多样性测量（分析相似性分布）
6. 分类（其中文本字符串按其最相似的标签分类）

嵌入是浮点数的向量（列表）。两个向量之间的距离衡量它们的相关性。小距离表示高相关性，大距离表示低相关性。例如，如果您有一个嵌入为[0.1,0.2,0.3]的字符串“狗”，则该字符串与嵌入为[0.2,0.3,0.4]的字符串“猫”比与嵌入为[0.9,0.8,0.7]的字符串“汽车”更相关。

我们可以尝试引入一些时效性的问题来测试下 OpenAI 的 Embeddings 模型的效果。来看看例子

案例 - 我们在 ChatGPT 中经常会问一些时效性的问题，例如以下场景



这个答案对吗？这是一个错误的答案，因为在 2023 年 5 月 3 日凌晨阿森纳是 3:1 战胜了切尔西。这是一个错误的结果。产生这个错误是因为 ChatGPT 不具备时效性。这个时候我们就需要给 GPT 一些时效性的信息，我们可以结合之前说的 Bing Search 来完成相关材料的补充。通过这些资料通过向量化去配对问题。(如果你不知道如何使用 Bing Search，请参考第三章的内容)。因为这是搜索结果，所以这里我希望启动两个模型 text-search-curie-doc-001 来配对并进行转化。找到与问题和内容最相近的前三个作为样本嵌入，来看看效果。

```
using System;
using System.Net;
using System.Net.Http;
using System.Net.Http.Headers;
using System.Threading.Tasks;
using System.Text;
using System.Text.Json;
using System.IO;
using System.Collections.Generic;
✓ 0.0s csharp - C# Script Code

string accessKey = "Your Bing Search v7 subscription key";
string uriBase = "https://api.bing.microsoft.com/v7.0/news/search";
string searchTerm = "2023年 5月3日阿森纳与切尔西的比赛结果";
✓ 0.1s csharp - C# Script Code

struct SearchResult
{
    public String jsonResult;
    public Dictionary<String, String> relevantHeaders;
}
✓ 0.0s csharp - C# Script Code

var uriQuery = uriBase + "?q=阿森纳与切尔西的比赛结果";
WebRequest request = WebRequest.Create(uriQuery);
request.Headers["Ocp-Apim-Subscription-Key"] = accessKey;
HttpWebResponse response = (HttpWebResponse)request.GetResponseAsync().Result;
string json = new StreamReader(response.GetResponseStream()).ReadToEnd();
✓ 1.2s csharp - C# Script Code

json
✓ 0.0s csharp - C# Script Code
{"_type": "News", "readLink": "https://api.bing.microsoft.com/api/v7.0/news/search?q=2023%E5%89%84+5%E6%9C%83%E6%97%A5%E9%98%BF%E6%A3%AE%E7%BA%83%E4%B8%8E%E5%88%87%E5%80%94%E8%A5%BF%E7%9A%84%E8%85%9B%E6%9E%9C", "queryContext": "阿森纳与切尔西的比赛结果"}
```

我们可以通过 Bing 搜索找到最佳的 5 个答案

```
newsList
✓ 0.0s csharp - C# Script Code

index value
0 ▼ Submission#7+News
  name 足彩果: 6场完赛皆正路 阿森纳3-1力克切尔西
  description 北京时间5月3日凌晨，胜负彩第23061期结束6场比赛的争夺。英超赛场，阿森纳3-1战胜切尔西。德国杯赛场，莱红牛5-1大胜弗赖堡。法甲赛场，图卢兹0-1朗斯。西甲赛场，巴萨、阿尔梅里亚、皇家社会皆取主胜。
1 ▼ Submission#7+News
  name 英超2023年5月3日阿森纳VS切尔西比赛前瞻
  description 阿森纳和切尔西两支伦敦球队将在英超比赛中相遇。比赛将在2023年5月3日星期三进行，场地为阿森纳的主场—温德哈特 ... 可以预料，在这个充满对抗与火药味的社交媒体世界里，这两支球队之间的比赛将会充满紧张感。
2 ▼ Submission#7+News
  name 5轮首胜！阿森纳夺回榜首 多赛2场领先曼城2分 连战3大强敌
  description 阿森纳原本是带着8分优势进入4月份的，结果连续被利物浦、西汉姆联和南安普敦逼平，又在天王山之战中完败于曼城脚下，4场比赛只拿到可怜的3分！而上周末，曼城2-1击败富勒姆，在少赛1场的情况下登顶积分榜。
3 ▼ Submission#7+News
  name 05月02日竞彩足球预测 英超 阿森纳vs切尔西
  description 本场英超比赛将在2023年5月3日进行，对阵双方是阿森纳和切尔西。两支球队在积分上存在一定的差距，阿森纳积分高达75分，而切尔西只有39分。但是足球比赛的胜负不完全取决于积分，我们需要从历史交锋、
4 ▼ Submission#7+News
  name 阿森纳英超对阵切尔西取得三连胜，自2004年以来首次
  description 直播吧5月3日讯 在本轮结束的一场英超比赛中，阿森纳3-1击败切尔西。据统计，阿森纳在近三场英超对阵切尔西的比赛中取得全胜，分别是上赛季客场4-2取胜，本赛季客场1-0，主场3-1，实现双杀。这也是阿森纳
```

接下来我们在 Azure OpenAI Portal 部署 text-search-curie-doc-001

Deployment name	Model name
SearchDocsEmbeddingModel	text-search-curie-doc-001
ChatGPTModel	gpt-35-turbo
GPT3Model	text-davinci-003
TextEmbeddingModel	text-embedding-ada-002

```
int n = 0 ;
foreach(var item in docs)
{
    var contentString = JsonSerializer.Serialize(item);
    var content = new StringContent(contentString, Encoding.UTF8,
"application/json");
    var result = await client.PostAsync(EMBEDDING_URL, content);
    var result_string = await result.Content.ReadAsStringAsync();
    var vector =
JsonDocument.Parse(result_string).RootElement.GetProperty("data").Enumerat
eArray().First().GetProperty("embedding").EnumerateArray().Select(x =>
x.GetDouble()).ToList<double>();
    var query = new QueryEmbeddingPrompt
    {
        title = newsList[n].name,
        content = newsList[n].description,
        input = vector
    };
    queries.Add(query);
    n++;
}
```

我们也需要把问题 "2023年5月3日阿森纳与切尔西的赛果" 做向量转化

```

var queryEmbeddingPrompt = new EmbeddingPrompt
{
    input = "5月3日阿森纳与切尔西的最新赛果"
};
✓ 0.0s                                         csharp - C# Script Code

string qcontentString = JsonSerializer.Serialize(queryEmbeddingPrompt);
✓ 0.0s                                         csharp - C# Script Code

var qcontent = new StringContent(qcontentString, Encoding.UTF8, "application/json");
✓ 0.0s                                         csharp - C# Script Code

var qresult = await client.PostAsync(EMBEDDING_URL, qcontent);
var qresult_string = await result.Content.ReadAsStringAsync();
✓ 0.7s                                         csharp - C# Script Code

var qresult_string = await result.Content.ReadAsStringAsync();
✓ 0.0s                                         csharp - C# Script Code

var qvector = JsonDocument.Parse(qresult_string).RootElement.GetProperty("data").EnumerateArray().First().GetProperty("embedding").EnumerateArray();
✓ 0.0s                                         csharp - C# Script Code

qvector
✓ 0.0s                                         csharp - C# Script Code

[ -0.012991605, 0.0075356904, -0.01392424, 0.022159407, -0.008869358, -0.0072932057, -0.013802998, -0.0073724794, 0.013401965, -0.0031453115, 0.00

```

我们可以通过 NumSharp 把问题和找到内容进行相似性映射，以找到最佳的三个答案，作为少样本。

```

foreach(var item in queries){
    var x = np.array(qvector).reshape(1, -1);
    var y = np.array(item.input).reshape(1, -1);
    var score = np.dot(x, y.T);
    Console.WriteLine(item.title + " " + score[0][0]);
    contextSamples.Add(new ContextSample{
        title = item.title,
        context = item.content,
        similar = score[0][0]
    });
}
✓ 0.1s

```

足彩彩票：6场完赛皆正路 阿森纳3-1力克切尔西 0.9091091045789176
英超2023年5月3日阿森纳VS切尔西比赛前瞻 0.830085404497792
05月02日竞彩足球预测 英超 阿森纳vs切尔西 0.8637967150069146
5轮首胜！阿森纳夺回榜首 多赛2场领先曼城2分 连战3大强敌 0.8898124969867314
五大联赛唯一，阿森纳本赛季有4人进球上双+3人参与进球20+ 0.834186512075553

然后就可以作为少样本进行提示，当再问一次“5月3日阿森纳与切尔西的赛果”就会有相关内容给出

```

string TEXT_PROMPT = " 如果你不能确认标准答案, 请回答我不太了解。\\n 相关内容: \\n" + context_result + "问题:5月3日阿森纳与切尔西的进球赛果?\\n 回答:";

✓ 0.0s csharp - C# Script Code

string URL = "Your Azure OpenAI URL";
string OPENAI_API_KEY = "Your Azure OpenAI API Key";
✓ 0.0s csharp - C# Script Code

var client = new HttpClient();
client.BaseAddress = new Uri(URL);
client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));
client.DefaultRequestHeaders.Add("api-key", OPENAI_API_KEY);
✓ 0.0s csharp - C# Script Code

public class ChatPrompt
{
    public string prompt { get; set; }
    public float temperature { get; set; }
    public int max_tokens { get; set; }
    public int n { get; set; }
}
✓ 0.0s csharp - C# Script Code

var chatPrompt = new ChatPrompt
{
    prompt = TEXT_PROMPT,
    temperature = 0.5f,
    max_tokens = 1024,
    n = 1
};
✓ 0.0s csharp - C# Script Code

string contentString = JsonSerializer.Serialize(chatPrompt);
✓ 0.0s csharp - C# Script Code

var content = new StringContent(contentString, Encoding.UTF8, "application/json");
var result = await client.PostAsync(URL, content);
var result_string = await result.Content.ReadAsStringAsync();
✓ 2.8s csharp - C# Script Code

result_string
✓ 0.0s csharp - C# Script Code
("id": "cmpl-7C3C5hZmo8ITB2kNt069vh4pUGxTf", "object": "text_completion", "created": 1683106569, "model": "text-davinci-003", "choices": [{"text": " 3-1, 阿森纳获胜。", "index": 0, "finish_reason": "stop", "logprobs": null}], "usage": {"completion_token": "Select Cell Language Mode", "prompt_tokens": 10, "total_tokens": 10}}
✓ 0.0s csharp - C# Script Code

3-1, 阿森纳获胜。

```

如果你想了解更多可以参考 [./Code/04.Embeddings/01.EmbeddingsWithBing.ipynb](#)

这个是我们把时效内容向量化后配对合适的内容作为少样本的案例，实际上我们也可以把这些准确的语料保存。这个时候我们就需要向量数据库来解决。

4.2.2 向量数据库 - Qdrant

随着 OpenAI 有更多企业级别的落地，向量数据库也越来越受到关注。用好向量数据库，可以让我们的业务更加高效。向量数据库的主要功能是将向量数据进行存储和查询。向量数据库的优点是可以快速查询，结合 OpenAI Embeddings 模型 - text-embedding-ada-002 可以打造自身企业的知识图谱，从而提高企业的效率。

Vector Database

Powering the next generation of AI applications
with advanced and high-performant vector
similarity search technology.



```
docker pull qdrant/qdrant  
docker run -p 6333:6333 qdrant/qdrant
```

Take a look at our [Quick Start Guide](#) or build your own neural search
with our step-by-step [Tutorial](#)

Benchmarks

Demos



Qdrant（读作：quadrant）是一个向量相似度搜索引擎。它提供了一个生产就绪的服务，带有一个方便的 API 来存储、搜索和管理点——带有额外有效负载的矢量。Qdrant 专为扩展过滤支持而定制。它使它可用于各种神经网络或基于语义的匹配、分面搜索和其他应用程序。

Semantic Kernel 支持 Qdrant 的调用，我们可以结合 Semantic Kernel 和 Qdrant 来实现知识图谱。使用方式如下：

1. 你必须安装 Docker <https://www.docker.com/products/docker-desktop/>
2. 完成本地 Qdrant 的安装

```
docker pull qdrant/qdrant
```

安装完后，启动 Qdrant

```
docker run -p 6333:6333 \  
-v $(pwd)/path/to/data:/qdrant/storage \  
qdrant/qdrant
```

3. 打开 `./Code/04.Embeddings/02.SKQdrantDemo.ipynb`，运行

```

string MemoryCollectionName = "demo";

await kernel.Memory.SaveInformationAsync(MemoryCollectionName, id: "id1", text: "我是卢建晖");
await kernel.Memory.SaveInformationAsync(MemoryCollectionName, id: "id2", text: "卢建晖是微软云技术布道师");
await kernel.Memory.SaveInformationAsync(MemoryCollectionName, id: "id3", text: "卢建晖从 2012 年到 2020 年是微软最有价值专家");
await kernel.Memory.SaveInformationAsync(MemoryCollectionName, id: "id4", text: "卢建晖专注于人工智能，移动应用开发，云原生相关技术");

✓ 3.1s

> collections = memoryStore.GetCollectionsAsync();
await foreach (var collection in collections)
{
    Console.WriteLine(collection);
}
✓ 0.2s

qdrant-test
docsdemo5
docsdemo2
docsdemo9
itopsvector
itopsvectordemo
qdrant-test1
qdrant-test2
docsdemo8
demo
docsdemo3
mvpdemo
aboutMe

+ Code + Markdown

var searchResults = kernel.Memory.SearchAsync(MemoryCollectionName, "有没有人工智能专家介绍", limit: 1, minRelevanceScore: 0.8);

✓ 0.0s

await foreach (var item in searchResults)
{
    Console.WriteLine(item.Metadata.Text + " : " + item.Relevance);
}
✓ 0.4s

卢建晖专注于人工智能，移动应用开发，云原生相关技术 : 0.82922995

```

通过向量数据库你可以嵌入不同的专业数据，我们会在最后一章介绍更多的内容。

4.3 Embeddings vs Fine-Tuning

通过 Embeddings 语义嵌入或者 Fine-Tuning 微调对大规模语言模型 (GPT-3/3.5/4) 进行使用，是受人瞩目的技术话题。虽然 Embeddings 语义嵌入和微调都是可以注入大规模语言模型的不同注释和语义的技术，但是它们的应用场景和使用方式是不同的。利用本节和大家说说 Embeddings 嵌入和微调的区别和使用场景。

4.3.1 从概念说起

语义嵌入 - 如上面提及的语义嵌入是文本的向量表示，可捕获单词或短语的语义。通过比较和分析这些向量，可以辨别文本元素之间的异同。

利用语义嵌入进行搜索可以快速有效地检索相关信息，尤其是在大型数据集中。与微调相比，语义搜索具有多项优势，例如更快的搜索速度、更低的计算成本以及防止虚构或事实捏造。由于这些好处，当目标是访问模型中的特定知识时，语义搜索通常受到青睐。嵌入在各个领域都有应用，包括推荐引擎、搜索功能和文本分类。

在语义嵌入中，我们会遇到以下的挑战

1. 高维度而增加的复杂性 - 大型语言模型生成的嵌入通常具有高维，这会导致计算复杂性和存储需求增加。这使得执行时变得复杂。
2. 向量出现稀疏矩阵 - 嵌入可能导致稀疏表示，其中向量中的大多数元素都是零或接近零的值。这种稀疏性会导致相似性搜索或其他任务期间内存消耗增加和计算时间变慢。

3. 可解释性 - 嵌入通常难以解释，因为它们表示高维空间中的复杂关系。这种可解释性的缺乏使得诊断问题（例如嵌入中的偏差或不准确）以及理解模型的潜在推理变得具有挑战性。
4. 数据质量 - 嵌入的有效性在很大程度上取决于用于训练大型语言模型的输入数据质量。不正确的数据可能导致嵌入效果不理想，
5. 专业性 - 虽然预训练的 LLM 可以生成通用嵌入，但将这些嵌入适应特定领域或任务可能需要对特定领域数据进行额外的微调或训练。此过程可能会占用大量资源，并且需要模型训练和优化方面的专业知识。
6. 道德：嵌入可能会无意中捕获并延续训练数据中存在的偏见，例如性别、种族或文化偏见。然后，这些偏差会影响模型在下游任务中的行为，引发对公平性和伦理影响的担忧。

微调 - 微调是一种用于改进预训练模型（例如聊天机器人）性能的技术。通过提供示例和调整模型的参数，微调使模型能够为特定任务生成更准确和上下文相关的响应。这些任务的范围从聊天机器人对话和代码生成到问题形成，确保更好地与所需输出保持一致。该过程类似于神经网络在训练期间调整其权重。作为迁移学习的一种形式，微调可以调整预训练模型来执行新任务，而无需进行大量的再训练。该过程涉及对模型的参数进行微调，使其能够更好地执行目标任务。

对 GPT-3 等大型语言模型 (LLM) 进行微调会带来一些挑战，这些挑战可能会影响其效率、可扩展性和有效性。这就包括：

1. 算力成本 - 微调 LLM 需要大量计算资源，这可能很昂贵，而且对于预算有限的组织或研究人员来说可能不可行。这些成本会限制微调在使用中的可扩展性。
2. 语料质量问题 - 高质量和相关的语料数据对于成功进行微调至关重要。高质量的语料从自然语言开始就是很难找到的，引入质量差的语料可能影响模型性能的偏差或不准确的风险。胡说八道的机会增加。
3. 过度拟合 - 过度拟合发生在模型对训练数据过于专业化时，导致对新示例的泛化能力较差。如何在专业化和泛化之间的平衡对于实现最佳性能至关重要。
4. 胡说八道：微调 LLM 有时会导致虚构，即模型产生不正确或捏造的信息，以及幻觉，即它产生似是而非但不正确的答案。这可能会破坏模型输出的可靠性和可信度。
5. 模型适应性：当新信息或更新知识可用时，可能需要重新微调模型。此过程可能会占用大量资源且繁琐，尤其是在需要经常更新模型以保持最新和相关性的情况下。
6. 道德：微调 LLM 可能会导致潜在的偏见，因为它们可能会无意中从训练数据中学习和传播有害的刻板印象或错误信息。确保微调模型的道德使用和输出可能具有挑战性，需要不断监测和评估。

4.3.2 如何选择

语义嵌入和微调是利用 GPT-3 等语言模型的两个主要方法。微调专注于通过迁移学习教授模型新任务，而语义嵌入涉及将文本的含义转换为向量表示，可用于语义搜索和信息检索等任务。

语义搜索，也称为神经搜索或矢量搜索，使数据库能够根据语义而不是仅根据关键字或索引来搜索和检索信息。这种方法具有高度可扩展性，通常比针对特定任务微调模型更快且更具成本效益。

相比之下，微调可能非常耗时、复杂且成本高昂，并且不会从本质上增强模型存储和检索新信息的能力。但通过微调的数据，很多时候不能得出正确的效果，依然会胡说八道。在使用嵌入和微调问答之间的选择取决于具体要求和问题的复杂性。对于更简单的搜索任务，嵌入可以更有效且更容易实现，而对于更复杂的问答场景，微调可能会提供更好的结果。

4.4 本章小结

本章我们介绍了语义嵌入 Embeddings 来调教你的模型，让你能注入行业应用数据，让模型更加专业化。在引入前，我们还把以往自然语言的处理方法做了一个简单的回顾，让你能更好的理解语义嵌入的优势。当然我们也比较了语义嵌入和微调的优缺点，让你能更好的选择适合你的场景。希望通过贲张的学习你能有一个更好的理解。

相关资料

1. 关于语义嵌入 Embeddings <https://learn.microsoft.com/zh-cn/azure/cognitive-services/openai/tutorials/embeddings>
2. 关于 Qdrant <https://qdrant.tech/>