**Connecting a Chatbot to an AI-as-a-Service**

Alan L. Dennis

University of the Cumberlands

**Connecting a Chatbot to an AI-as-a-Service**

Chatbots are a powerful way to interact with computers. Combining a chat interface with an AI-backed service is compelling. This document outlines the assignment’s instructions and covers the detailed steps necessary to connect a traditional chatbot to a managed AI service.

# General Instructions

After reviewing the lecture, use this document to create an AI-connected Chatbot. Ensure that your source code is stored or exported to a GitHub repository and that your instructor has access to that repository (and knows its name). Additionally, submit your source code in an archive file (ZIP) when you submit the report. Document the process of creating the bot in a report. The report should be 5-7 pages of content and should include a title and references page using APA format. Document any challenges you encountered and mitigation strategies you utilized.

The chatbot should respond to multiple prompts, provide a list of its capabilities, and handle malformed questions or input. You are not required to use the framework presented here. However, your bot must utilize a cloud-based AI-as-a-service offering, like Azure Cognitive Services. Ensure that you document your progress with screenshots and a narrative.

There are two submissions associated with this project. First, submit your source code in a ZIP file, along with the URL of your GitHub repository. Next, submit your five-to-seven-page report in APA format (not counting title and reference pages) describing your experience and what you learned.

# Free Azure Account

To sign up for a free Azure account, navigate to <https://azure.microsoft.com/en-us/free>, as shown in *Figure 1*.

A computer on a white background

Description automatically generated

Figure 1: Free Azure Account

Ensure you click the **Start free** button. When prompted for an account, you can use your University of the Cumberlands account or a personal account. Once you have authenticated, you will likely taken to the Microsoft Azure Quickstart Center, as shown in *Figure 2*.

A screenshot of a computer

Description automatically generated

Figure 2: Azure Quickstart Center

You can click the **Home** link in the upper right-hand corner to navigate to the Azure Portal. If you are unfamiliar with Azure services, you might want to review the resources available to you in the Quick Start Center. If you navigate to the main portal page and later want to revisit the quick start, simply click the rocket-shaped icon, as shown in *Figure 3*.

A screenshot of a computer

Description automatically generated

Figure 3: Azure Portal with Quickstart Center

You can review the list of Azure services that are free by navigating to <https://azure.microsoft.com/en-us/pricing/free-services>. Note that some services are free for the first year, while others are always free. Also, note that there are limitations on the free services.

# Azure AI Service

Azure offers a large number of services for free. Finding the right service can challenging. Fortunately, Azure has a robust search capability. For our example, we want to look for an AI service that returns sentiment. Searching for “sentiment” in the search bar returns two candidates, as shown in *Figure 4*.

A screenshot of a computer

Description automatically generated

Figure 4: Search for sentiment analysis

The services section contains Azure services. Marketplace contains offerings from vendors other than Microsoft. We are interested in the Azure AI Services, with the keyword sentiment analysis. Microsoft changed the name of Azure Cognitive Services to Azure AI Services in July of 2023. Navigating to Azure AI services initially shows a list with no services, as shown in *Figure 5*.

A screenshot of a computer

Description automatically generated

Figure 5: No Azure AI Services

We want to create a Language service (not to be confused with Language understanding). Microsoft change the grouping of services related to natural language processing to be together under Azure AI Language, <https://azure.microsoft.com/en-us/products/ai-services/ai-language/>.

Clicking on Language service shows an empty list with a **Create language** button. Clicking the button brings up a form offering to add optional features, as shown in *Figure 6*.

A screenshot of a computer

Description automatically generated

Figure 6: Optional language features

Since we are only concerned with sentiment analysis, we can click the **Continue to create your resource** button at the bottom of the form. We are not presented with a Create Language wizard, as shown in *Figure 7*. When creating resources in Azure, you must select a subscription and resource group. A resource group is, as the name implies, a collection of resources. Using resource groups is a way to manage resources and simplifies removal. You also need to select a region. You can think of a region as being similar to a data center. Different regions have different capabilities. The guiding principle is to always have all of your resources in the same region. Next, you will need to supply a name for the resource. All Azure resources have a name, simplifying the location and management of those resources. You are required to check a box indicating that you have read the Responsible AI Notice and agree to the terms.

A screenshot of a computer

Description automatically generated

Figure 7: Basics section of Language Service

Once you have supplied the required information, you can click the **Review + create** button. Alternatively, you can add additional values for networking, identity, and tags. For our purposes, we can simply create the service. The Review + create screen displays the information you entered and validates your choices, as shown in *Figure 8*.

A screenshot of a computer

Description automatically generated

Figure 8: Review and create Language

Since there were no issues, we can click the Create button. Note that you could download the automation template. This a common practice within large/enterprise-class organization. Creating create starts the deployment process. This process can take some time to complete. Once it completes, navigate to the newly created resource and then to the Keys and Endpoints section under Resource Management, as shown in *Figure 9*.

A screenshot of a computer

Description automatically generated

Figure 9: Language Service's Keys and Endpoints

We will need the endpoint and keys to connect to the service. We have now created an Azure AI Language service. We will interact with it via API calls from within our simple chatbot.

# Integrated Chatbot

Use the echo bot from Topic 5 (Week 5/16 or 3/8) as a starting point for our addition of sentiment analysis. Create a copy of the code and make a note in the README.MD file where the files came from originally, and the reason that you are copying them. Next, review an article on adding sentiment analysis with Python (<https://learn.microsoft.com/en-us/python/api/overview/azure/ai-textanalytics-readme?view=azure-python>). In *Figure 9*, we located the keys and endpoint for the service. We will need them so that we can connect to the service.

To make this a little more robust, I will extend the DefaultConfig class to include an API\_KEY and an ENDPOINT\_URI, as shown in *Figure 10*.

A screenshot of a computer

Description automatically generated

Figure 10: Extending DefaultConfig

Note that I added a comment before my modifications, to make it clear that I changed it, and why. You’ll notice the use of os.environ.get. This is a way of retrieving environmental variables (typically initialized using the SET keyword) and using them at runtime. This allows you to store sensitive values (like your API\_KEY) in a location other than you Python code. We can use the following command to set an environmental variable that we can get the value from at run time:

SET MicrosoftAIServiceEndpoint=https://t6languageservice.cognitiveservices.azure.com/

To verify this is working correctly, I added a print statement to display the value on startup, as shown in *Figure 11*.

A screen shot of a computer

Description automatically generated

Figure 11: Retrieving environmental values

Since this is clearly not something we want to use outside of debugging I quickly commented it out. The next step is to SET the MicrosoftAPIKey environment variable with one of the KEYs from the Language service, from *Figure 9*. Once completed, we can then extend app.py to create an instance of the TextAnalyticsClient, as shown in *Figure 12*.

A screen shot of a computer

Description automatically generated

Figure 12: Extend app.py to read new values

Before we go too far, we should be certain to install the required libraries. We can do this with the command:

pip install azure-ai-textanalytics

Once installed, we are ready to use sentiment analysis from within the bot. To do this, we locate the main message handler in the bot, and extend it, as shown in *Figure 13*.

A screenshot of a computer

Description automatically generated

Figure 13: Extend Chatbot to get Sentiment Analysis

We can now test our bot. This is not a very polished example but does demonstrate the key concepts of extending a bot by using an AI service. We can see the results by using the bot emulator, as shown in *Figure 14*.

A screenshot of a computer

Description automatically generated

Figure 14: Chatbot emulator showing sentiment results

There is plenty of areas for improvement with this bot. You could have a set of phrases it returns depending on the type of response from the sentiment analysis. This is intended to show a very simple example of how we can include AI services in solutions.

# Conclusion

This document outlines the steps necessary to create a free Azure service, provision an AI Language Service, and integrate that service into a chatbot. This should be viewed as a starting point. There are many, better, ways to construct chatbot. This is approach was selected for its simplicity.