Supplementing the O*NET Sources of Additional Information: A Preliminary Exploration of the Use of ChatGPT

Phil Lewis and Jeremiah Morris
National Center for O*NET Development

Prepared for

U.S. Department of Labor
Employment and Training Administration
Office of Workforce Investment
Division of National Programs, Tools, & Technical Assistance
Washington, DC

Submitted by

The National Center for O*NET Development
April 2, 2024

www.onetcenter.org

Table of Contents

Introduction

This paper describes the initial exploration of the use of ChatGPT, as well as the related API from OpenAI, as an input source for supplementing the by-occupation listings of related professional associations included in the O*NET Sources of Additional Information (SAI). Lessons learned are described. Then the methodology and results of a preliminary implementation are presented. Finally, potential future steps are discussed.

Sources of Additional Information (SAI)

O*NET OnLine occupation reports contain a section labeled "Sources of Additional Information." This section provides customers with a list of sources external to the O*NET Program that offer occupationally relevant information, such as related jobs, specialties, and industries. The goal is to enhance customers' exploration of each occupation. The vast majority of these linked sites are related professional associations. Professional associations included have membership that is nationally-, internationally-, or regionally- (i.e., representing three or more states) based. The links are provided for customer convenience and do not constitute an endorsement by the National Center for O*NET Development or the U.S. Department of Labor. SAI listings are available to program and application developers via O*NET Web Services. The SAI professional associations also serve as a starting point for soliciting Source Organization support for occupations newly added to the O*NET Program's Occupational Expert (OE) data collection methodology (U.S. Department of Labor, 2024).

At the time of this project, SAI were available for all 923 O*NET-SOC data-level occupations and included approximately 7,000 linkages to sources for customers to explore.

SAI listings were populated over time by a variety of sources including:

- O*NET Program outreach efforts
- O*NET Program OE data collection and O*NET Ally efforts
- Review of existing, government sponsored related listings [e.g., Occupational Outlook Handbook (OOH) (Bureau of Labor Statistics, 2023)]
- Sampled job incumbent and occupational expert membership (See Craven & Lewis, 2018)
- Customer and professional association input

The O*NET Program's goal is to maintain the accuracy of the information, while also trying to expand the coverage or representativeness of the listings. Current emphasis is focused on adding regional associations who are underrepresented in the listings.

ChatGPT

The recent increased ease and public access to generative Artificial Intelligence (AI), including large language models (LLMs), such as Generative Pre-Trained Transformers (GPTs), has led to much speculation and research on how these types of tools will impact the world of work (Carbonero et al., 2023; Chui, et al., 2023; Eloundou et al., 2023; Felton et al., 2023a; Felton et al., 2023b; Hatzius, 2023; World Economic Forum, 2023). One prevalent example of this type of

tool is ChatGPT developed by OpenAI and launched for public use on November 30, 2022 (OpenAI, 2022). OpenAI also developed an API version of the generator (Brockman et al., 2020). In the Industrial-Organizational Psychology field there has been much discussion on how to potentially leverage ChatGPT within the field's work processes, in addition to using it within research studies (Cannon, 2023; Harper, 2023a; Harper, 2023b; Schmidt & Biermeier-Hanson, 2023; Trupulse, 2023). A common theme is to use ChatGPT to generate "data-driven insights" on questions, challenges, goals, and needed decisions that prior to ChatGPT's availability would have been costly, time consuming, and/or potentially not possible. Data-driven insight was the impetus for this current exploratory project. The O*NET Program's goal was to examine the use and effectiveness of ChatGPT to identify professional associations linked to occupations that were not already identified and included with SAI, especially those with regional membership.

Lessons Learned

The first phase of this project included trying out, testing, and getting familiar with ChatGPT to help learn the "what and how" it could be leveraged to potentially supplement existing O*NET SAI listings. This section of the paper describes discoveries and lessons learned.

Web-based, Graphical Interface Version

We started the project off with the web-based ChatGPT version, 3.5 model (OpenAI, 2022). The service was available, free-of-charge. We initially tried simple, straightforward prompts. For example:

> *Provide a list of up to 20 professional associations in the United States related to O*NET-SOC 11-9199.08 Loss Prevention Managers.*

Returns on specific occupations were very promising, and included both professional title and description. A quick comparison of a few occupations revealed that the returns included several potential associations beyond what was in the current SAI. However, even after selecting "new chat" or refreshing the input source, we discovered that returns were not consistent when the same prompt was used multiple times. In addition, results across different users applying the same prompt were not consistent. This was somewhat alleviated by adding more detail to the prompt (e.g., clarifying that relatedness is based on membership), but returns were still quite varied. While simple and easy to use, and returning results that were conversational and included intros and disclaimers[1], the web-based version didn't disclose the parameters it used to provide results, nor did it allow for the adjustment of parameters based on our project goals. This "black-box" feel led us to explore the available API version.

API Version

We explored the use of the available OpenAI Chat Completions API (OpenAI, 2023), which supports text-based exchanges like ChatGPT but offers more customizability. Practical

[1] One important disclaimer often provided was a reminder that the model did not use real-time data. For example, during initial testing the disclaimer indicated that the model was updated using data from September, 2021.

advantages for our project existed. Prompt templates were available that could provide results (i.e., professional associations and other details) in a machine-readable format. In addition, we could program an automated process for running prompts and large batches of occupational titles/codes. While the API was offered as a fee-for-service, a cost examination based on the sample processing of five occupations revealed a low-cost burden (just over $1.00 in total).

Parameters

The API offered much more flexibility in terms of customization and control of results. Several parameters were available for the user to set, including:

- Model = select the latest release, or a choose a specific version such as 3.5 or 4.0 (gpt-4 0613)
- N = select the number of times results are generated and returned
- Temperature = select a decimal between 0 and 2 to control how often the generator selects a less-likely completion/return when building its response

Control of the Model parameter was important to our project, allowing for better version control of results if additional runs were needed (e.g., if the processing of additional occupations was done at a different point in time).

We used the N parameter to help with the previously identified issue of consistency of results. While the web-based version provides one set of results, multiple runs or attempts can be made using the API. For example, we set the N to provide 6 responses for each prompt, and then removed duplicates (by filtering on exact URL) when generating the output. We also considered tracking how many times a result was suggested (ranging from 1-6). This process would allow us to rank results based on the returned consistency of results.

The Temperature parameter also controls the variation in returns. Lowering it should reduce the variation between responses, giving fewer options but staying more on topic (i.e., less "guessing"). Raising it does the reverse.

Prompt Templates

The API allowed for much more detailed, directive prompts that, as mentioned above, could be put into a "template" to produce machine-readable results across batches of occupations. Multiple templates could also be applied within each API run. We learned through trial and error that more prompt detail was needed to achieve the results we desired.

For example, tests of prompts indicated that while the GPT model's knowledge of O*NET-SOC 2010 taxonomy occupations was good, its knowledge of the O*NET-SOC 2019 taxonomy was extremely limited[2]. It could get to a 2019 title from its code (and the reverse) only if the occupation was also in the 2010 taxonomy. If pressed for information about tasks or description, it would borrow from other occupations or other sources (like a DOD Cyber page on digital forensics). This led us to take advantage of the ability within the API to incorporate the retrieval and use of much more detail within the prompt, beyond the occupation title and code, including

[2] Visit O*NET-SOC Taxonomy for an overview and listings related to each version of the taxonomy.

each occupation's description and full task listing. For example:

> *The occupation "[[title]]" has the following definition:*
>
> *[[desc]]*
>
> *Important tasks for this occupation include:*
>
> ** [[task1]]*
>
> ** [[task2]]*
>
> ** [[task3]]*
>
> ** [[task4]]*
>
> ** [[task5]]*

We also iteratively learned that the prompt for defining what constituted membership in a professional association would benefit by adding very specific direction and detail. For example:

> *If a worker in this occupation wanted to join a professional association, which associations would they consider? Provide a list of national or international associations which accept members in the United States. Format the list as a JSON array, with the fields: name, url. If an association has an acronym, do not include it in the 'name' field. If possible, include up to 20 associations in the list. Choose associations most likely to have [[title]] as members.*

Lastly, we took advantage of the API's ability to have a "conversation" featuring more than one prompt, including a second directive with specific focus on identifying potential regional-based professional associations. For example:

> Prompt 2 Template
>
> *Provide a second list of regional associations to consider. Regional associations must represent members across three or more states in the United States. Choose associations most likely to have [[title]] as members.*

Return Quality

Returns from iterative test runs indicated there was great promise in using the OpenAI API as an input source for supplementing SAI listings. Examination of sample returns by occupation showed robust lists of potential associates that went beyond the current SAI listings. However, the example runs also emphasized that that results should be treated as "insights" or leads, rather than "as is" results that could be incorporated without additional review and/or refinement.

Quality issues identified within some of the association returns included:

- Association not found
- No longer operating
- Incorrect name
- Invalid URL

- Incorrectly linked URL by association name (i.e., one was correct while the other was not)
- Not relevant to the target occupation
- Not national or regional (e.g., local or state)

Preliminary Implementation

The second phase of this project included a preliminary implementation of a process on a sample of O*NET-SOC occupations based on the lessons learned.

Methodology

Sample

A sample of 100 occupations was drawn from the 923 data-level occupations included in the O*NET-SOC 2019 Taxonomy (Gregory et al., 2019). Occupations selected included those that were newly introduced in 2019 and also occupations whose data had been challenging to collect via OE data collection methodology. Both of these categories would benefit from the addition of newly linked professional associations. For a listing of the selected occupations, see Appendix A. The description and task statement listing for each of the sampled occupations were obtained from the O*NET 28.0 Database (National Center for O*NET Development, 2023).

API Script

Next, the prompt templates and parameters were set for the API. For the detailed prompts and script used, see Appendix B[3]. Parameter settings included:

- Model = gpt-4 (gpt-4-0613)
- Prompt 1 N = 6
- Prompt 2 N = 4
- Temperature = 1[4]

API Returns

For the sample of 100 occupations, on average just over 100 suggested professional associations linkages per occupation (Total = 10,231 linkages) were identified, including 8,318 distinct association URLs. The association suggestions were compared to the current SAI database and to each other using existing matching algorithms and procedures; this automated process resulted in 6,346 unique suggested associations. Identified exact, near matched, and mismatched associations (matched either the SAI name or SAI URL, but not both) were reviewed and resolved by occupational analysts. In some cases, this led to updated association names and/or

[3] The script is also available from our GitHub repository along with sample input data.
[4] The Temperature parameter was set to 1 (the default, median setting) because during the initial test we wanted to achieve a variety of results without generating excessive numbers of off-topic suggestions. We could use the consistency rank (i.e., 1-6 or 1-4) to apply cut-offs within the returns if needed.

URLs within the SAI database.

Remaining new suggested association to occupation linkages were then categorized, and led to:

- 1,615 potential new linkages to 849 existing SAI associations
- 1,848 potential linkages to 713 new multiple-linked associations
- 4,495 potential linkages to new single-linked associations

Linkage Reviews

A decision was made to review the potential new linkages related to 1) existing SAI associations, and 2) new multiple-linked associations. The review of linkages to new single-linked associations was tabled (instead this subset of returns was provided to the internal OE data collection team for its potential use in securing needed Source Organizations).

- Existing SAI associations new linkage review. Trained occupational analysts reviewed the suggested new occupational linkages for existing SAI associations. The analyst visited the existing SAI association's website for content and membership. Using the occupation reports in O*NET OnLine, the proposed new link to an occupation was considered based on the occupation title, definition, sample of reported job titles, and task listing. The analyst then indicated if the new linkage was valid.
- New multiple-linked associations linkage review. Trained occupational analysts reviewed the suggested occupation linkages for new multiple-linked associations (i.e., those potentially linked to more than one occupation). The analyst first visited the new association's website, confirming:
    - Association found
    - Currently operating
    - National or regional
    - If needed:
        - Updated name based on format and style guidelines
        - Updated URL based on format and style guidelines

    If the above was confirmed, the analyst then reviewed the association's website for content and membership. Using the occupation reports in O*NET OnLine, the proposed link to an occupation was considered based on the occupation title, definition, sample of reported job titles, and task listing. The analyst then indicated if the linkage was valid.

Results

The two reviews described above led to a total of 1,843 new professional association to occupation linkages within the SAI database across the sample of 100 occupations. This included 383 new associations. For those occupations updated, the average number of linked associations rose from 6.48 to 24.76. The goal to increase the coverage of regional associations was also met, representing 18% of the newly added linkages. The published SAI after the completion of this project approached 9,000 linkages to sources for customers to explore.

Conclusion and Future Steps

This project's initial exploration of the use of OpenAI's ChatGPT and API as input sources for supplementing the by-occupation listings of related professional associations included in the O*NET Sources of Additional Information (SAI) proved to be successful. New associations and occupational linkages were successfully identified. Regional representation was expanded, addressing one of the O*NET Program's stated goals. Use of this methodology for newly introduced occupations that need initial population of SAI is also very promising.

The process, however, of capitalizing on a GPT model's data-driven insights remained fairly labor intensive with detailed, time consuming occupational analyst reviews needed prior to implementation within publicly available SAI. There is hope that OpenAI's recently added access to real-time data within the API will potentially eliminate the number of identified inactive or renamed associations, improve provided URLs, and better match newly added occupations to the O*NET-SOC taxonomy.

In the future, enhancements to the prompt content/directives and API parameters will be explored to see if more narrowed and accurate results can be generated. Exploration of the utility of the mentioned consistency ranking will also be investigated. In addition, we anticipate in the near future new and expanded parameters and features to be available for use in this very rapidly evolving AI and LLM world.

References

Brockman, G., Murati, M., & Welinder, P. (2020, June 11). *We're releasing an API for accessing new AI models developed by OpenAI*. Open AI Blog. https://openai.com/blog/openai-api

Bureau of Labor Statistics. (2023). *Occupational Outlook Handbook*. U.S. Department of Labor. https://www.bls.gov/ooh/

Cannon, B. (Host). (2023, March 20). Exploring the role of ChatGPT in psychology: From understanding the model to enhancing the field. Interview with ChatGPT [Audio podcast episode]. In *PSYCHEVERYWHERE*. Psy Chi. https://www.psichi.org/page/podcast-exploring-the-role-of-chatgpt-in-psychology

Carbonero, F., Davies, J., Ernst, E., Fossen, F. M., Samaan, D., & Sorgner, A. (2023). The impact of artificial intelligence on labor markets in developing countries: A new method with an illustration for Lao PDR and urban Viet Nam. *Journal of Evolutionary Economics*. https://link.springer.com/article/10.1007/s00191-023-00809-7

Chui, M., Hazan, E., Roberts, R., Singla, A., Smaje, K., Sukharevsky, A., Yee, L., & Zemmel, R., (2024, June 14). *Economic potential of generative AI.* McKinsey. https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/the-economic-potential-of-generative-ai-the-next-productivity-frontier

Craven, D. E., & Lewis, P. M. (2018). *Updating O*NET sources of additional information listings: Use of job incumbent and occupational expert professional association membership.* Raleigh, NC: National Center for O*NET Development. https://www.onetcenter.org/reports/Additional_Info.html

Eisfeldt, A. L., Schubert, G., & Zhang, M. B. (2023). *Generative AI and firm values* (No. w31222). National Bureau of Economic Research. https://www.nber.org/system/files/working_papers/w31222/w31222.pdf

Eloundou, T., Manning, S., Mishkin, P., & Rock, D. (2023). *GPTs are GPTs: An early look at the labor market impact potential of large language models.* arXiv preprint. https://arxiv.org/abs/2303.10130

Felten, E., Raj, M., & Seamans, R. (2023a). How will language modelers like ChatGPT affect occupations and industries? *arXiv*. https://arxiv.org/abs/2303.01157

Felten, E. W., Raj, M., & Seamans, R. (2023b). *Occupational heterogeneity in exposure to generative AI*, SSRN. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4414065

Gregory, C., Lewis, P., Frugoli, P., & Nallin, A. (2019). *Updating the O*NET®-SOC taxonomy: Incorporating the 2018 SOC structure.* Raleigh, NC: National Center for O*NET Development. https://www.onetcenter.org/reports/Taxonomy2019.html

Harper, M. (2023, January 25). Artificial intelligence and ChatGPT: Implications and applications for IO psychologists. TTS Talent. https://www.tts-talent.com/blog/artificial-intelligence-and-chatgtp-implications-and-applications-for-io-psychologists/

Harper, M. (2023, March 30). 3 ways ChatGPT will change how IO psychologists work. TTS Talent. https://www.tts-talent.com/blog/3-ways-chatgpt-will-change-how-io-psychologists-work/

Hatzius, J. & Pierdomenico, G. (2023). The potentially large effects of artificial intelligence on economic growth (Briggs/Kodnani). *Goldman Sachs*. https://www.gspublishing.com/content/research/en/reports/2023/03/27/d64e052b-0f6e-45d7-967b-d7be35fabd16.html

National Center for O*NET Development. (2023). O*NET 28.0 database. https://www.onetcenter.org/dictionary/28.0/excel/

OpenAI. (2022). ChatGPT Generative Pre-trained Transformer (version gpt-3.5) [Large language model] https://chat.openai.com/chat

OpenAI. (2023). ChatGPT Completions API (version gpt-4 0613) [Large language model]. https://platform.openai.com/docs/guides/text-generation

Schmidt, G. & Biermeier-Hanson, B. (2023, September 21). What can ChatGPT tell us about I-O psychology? Society for Industrial and Organizational Psychology. https://www.siop.org/Research-Publications/Items-of-Interest/ArtMID/19366/ArticleID/7898/preview/true/What-Can-ChatGPT-Tell-Us-About-I-O-Psychology

Trupulse. (2023, August 30). Should HR teams embrace ChatGPT? A perspective from an IO psychologist. https://www.trupulse.ai/blog/should-hr-teams-embrace-chatgpt-a-perspective-from-an-io-psychologist

U.S. Department of Labor. (2024). *2024 OMB clearance package*. https://www.onetcenter.org/reports/omb2002.html

World Economic Forum. (2023). *Jobs of tomorrow: Large language models and jobs* [White Paper]. https://www.weforum.org/publications/jobs-of-tomorrow-large-language-models-and-jobs/

Appendix A: Occupations Selected for Preliminary Implementation

O*NET-SOC Code	O*NET-SOC Title
13-2053.00	Insurance Underwriters
13-2041.00	Credit Analysts
17-3027.00	Mechanical Engineering Technologists and Technicians
29-1124.00	Radiation Therapists
13-1032.00	Insurance Appraisers, Auto Damage
43-9021.00	Data Entry Keyers
33-9011.00	Animal Control Workers
45-2021.00	Animal Breeders
49-2094.00	Electrical and Electronics Repairers, Commercial and Industrial Equipment
47-5051.00	Rock Splitters, Quarry
51-8031.00	Water and Wastewater Treatment Plant and System Operators
13-1041.00	Compliance Officers
13-2011.00	Accountants and Auditors
17-2111.00	Health and Safety Engineers, Except Mining Safety Engineers and Inspectors
17-3025.00	Environmental Engineering Technologists and Technicians
19-1021.00	Biochemists and Biophysicists
19-1031.00	Conservation Scientists
19-4043.00	Geological Technicians, Except Hydrologic Technicians
27-3043.00	Writers and Authors
39-3012.00	Gambling and Sports Book Writers and Runners
47-4091.00	Segmental Pavers
15-1254.00	Web Developers
15-1299.01	Web Administrators

O*NET-SOC Code	O*NET-SOC Title
11-1011.03	Chief Sustainability Officers
11-9121.02	Water Resource Specialists
11-9199.08	Loss Prevention Managers
13-1041.07	Regulatory Affairs Specialists
13-1081.02	Logistics Analysts
13-1199.05	Sustainability Specialists
15-1241.01	Telecommunications Engineering Specialists
15-1243.01	Data Warehousing Specialists
15-1299.03	Document Management Specialists
15-2051.02	Clinical Data Managers
17-2112.02	Validation Engineers
17-2199.03	Energy Engineers, Except Wind and Solar
17-2199.06	Microsystems Engineers
17-3027.01	Automotive Engineering Technicians
19-2041.01	Climate Change Policy Analysts
29-1217.00	Neurologists
29-1071.01	Anesthesiologist Assistants
47-4099.03	Weatherization Installers and Technicians
29-1128.00	Exercise Physiologists
11-2032.00	Public Relations Managers
11-2033.00	Fundraising Managers
11-3013.00	Facilities Managers
11-3013.01	Security Managers
11-9072.00	Entertainment and Recreation Managers, Except Gambling
13-1082.00	Project Management Specialists
13-1199.07	Security Management Specialists
13-2022.00	Appraisers of Personal and Business Property
13-2051.00	Financial and Investment Analysts
13-2054.00	Financial Risk Specialists
15-1252.00	Software Developers

O*NET-SOC Code	O*NET-SOC Title
15-1255.00	Web and Digital Interface Designers
15-1299.02	Geographic Information Systems Technologists and Technicians
15-1299.04	Penetration Testers
15-1299.05	Information Security Engineers
15-1299.06	Digital Forensics Analysts
15-1299.07	Blockchain Engineers
15-2051.00	Data Scientists
17-3026.01	Nanotechnology Engineering Technologists and Technicians
17-3028.00	Calibration Technologists and Technicians
19-3033.00	Clinical and Counseling Psychologists
19-3039.02	Neuropsychologists
19-3039.03	Clinical Neuropsychologists
19-4044.00	Hydrologic Technicians
25-2055.00	Special Education Teachers, Kindergarten
25-2056.00	Special Education Teachers, Elementary School
25-3031.00	Substitute Teachers, Short-Term
25-4022.00	Librarians and Media Collections Specialists
25-9042.00	Teaching Assistants, Preschool, Elementary, Middle, and Secondary School, Except Special Education
25-9043.00	Teaching Assistants, Special Education
27-2091.00	Disc Jockeys, Except Radio
27-3023.00	News Analysts, Reporters, and Journalists
27-4015.00	Lighting Technicians
29-1212.00	Cardiologists
29-1214.00	Emergency Medicine Physicians
29-1242.00	Orthopedic Surgeons, Except Pediatric
29-1243.00	Pediatric Surgeons
29-2011.04	Histotechnologists

O*NET-SOC Code	O*NET-SOC Title
29-2012.01	Histology Technicians
29-2036.00	Medical Dosimetrists
29-2042.00	Emergency Medical Technicians
29-2043.00	Paramedics
29-2072.00	Medical Records Specialists
29-9021.00	Health Information Technologists and Medical Registrars
33-1091.00	First-Line Supervisors of Security Workers
33-9094.00	School Bus Monitors
35-3023.00	Fast Food and Counter Workers
39-1013.00	First-Line Supervisors of Gambling Services Workers
39-1014.00	First-Line Supervisors of Entertainment and Recreation Workers, Except Gambling Services
39-4012.00	Crematory Operators
41-3091.00	Sales Representatives of Services, Except Advertising, Insurance, Financial Services, and Travel
45-3031.00	Fishing and Hunting Workers
53-1044.00	First-Line Supervisors of Passenger Attendants
53-3051.00	Bus Drivers, School
53-3053.00	Shuttle Drivers and Chauffeurs
53-3054.00	Taxi Drivers
53-4022.00	Railroad Brake, Signal, and Switch Operators and Locomotive Firers
53-6032.00	Aircraft Service Attendants

Appendix B: GPT Prompts and Scripting

Prompts were delivered to the OpenAI API from a NodeJS script, included in full below. The script, along with an input data file covering the 100 occupations in Appendix A, may be downloaded from our GitHub repository:

https://github.com/onetcenter/gpt-suggest-associations

After defining the prompt templates and other settings, the script performs the following steps:

- Prepare an output Excel file in which to write the API results
- Read the list of occupations and related information (description, task statements) from an input Excel file. For each occupation:
  - Create the first prompt by populating the template
  - Call the API to receive suggested associations as a JSON array
  - Write each association, along with the occupation info and other metadata, into the output file (skipping any entries already returned for this occupation)
  - Repeat the above steps for the second prompt
- Return the completed Excel file as program output

```
'use strict'
import XlsxPopulate from 'xlsx-populate'
import axios from 'axios'
import sleep from 'sleep-promise-native'

const model = 'gpt-4'
const inputTokenCost = 0.00003
const outputTokenCost = 0.00006

const promptTemplate1 = `The occupation "[[title]]" has the following definition:

[[desc]]

Important tasks for this occupation include:

* [[task1]]
* [[task2]]
* [[task3]]
* [[task4]]
* [[task5]]

If a worker in this occupation wanted to join a professional association, which associations
would they consider? Provide a list of national or international associations which accept
members in the United States. Format the list as a JSON array, with the fields: name, url. If
an association has an acronym, do not include it in the 'name' field. If possible, include up
to 20 associations in the list. Choose associations most likely to have [[title]] as members.`
const n1 = 6
const temp1 = 1.0
const promptTemplate2 = `Provide a second list of regional associations to consider. Regional
associations must represent members across three or more states in the United States. Choose
associations most likely to have [[title]] as members.`
const n2 = 4
const temp2 = 1.0
```

```
function fillTemplate (template, substitutions)
{
  const replacer = (match, p1, offset, string) => {
    if (Object.hasOwn(substitutions, p1))
      return substitutions[p1]
    return match
  }
  return template.replaceAll(/\[\[(\w+)\]\]/g, replacer)
}

async function callChat (prompt, n, temp, history = [])
{
  const data = {
    model: model,
    n: n,
    temperature: temp,
    messages: history.concat({ role: 'user', content: prompt })
  }
  return await call_axios({
    url: 'https://api.openai.com/v1/chat/completions',
    method: 'POST',
    data: data,
    timeout: 120000,
    maxRedirects: 0,
    headers: { Authorization: `Bearer ${process.env.OPENAI_TOKEN}` }
  }, 3)
}

async function call_axios(config, retries) {
  let err = undefined
  for (let i = 0; i < retries; i++) {
    try {
      const response = await axios(config)
      if (response.status == 200) {
        return response.data
      } else {
        err = `Received error code ${response.status}`
      }
    } catch (error) {
      if (error.response) {
        err = `Received error code ${error.response.status}`
        console.error(error.response.data)

      } else if (error.request) {
        err = `No response`
        // console.error(error.request)
      } else if (error.message) {
        err = `Failed with reason "${error.message}"`
      }
    }
    await sleep(250)
  }
  throw new Error(`Call to ${config.url} failed: ${err}`)
}

async function readToBuffer (stream) {
  let bufchunks = []
  for await (const chunk of stream) {
    bufchunks.push(chunk)
```

```
  }
  return Buffer.concat(bufchunks)
}

;
(async () => {
  let inputTokens = 0
  let outputTokens = 0
  let occsProcessed = 0
  let specificModel = model

  const xout = await XlsxPopulate.fromBlankAsync()
  const sheet0 = xout.sheet(0)
  sheet0.name('Associations')
  for (const col of 'ABCDEFG'.split('')) {
    sheet0.column(col).style({ fontFamily: 'Times New Roman', fontSize: 12, wrapText: true,
verticalAlignment: 'top' })
  }
  sheet0.row(1).style({ bold: true, verticalAlignment: 'bottom' })
  sheet0.column('A').width(12).style({ horizontalAlignment: 'right' })
  sheet0.column('B').width(20).style({ horizontalAlignment: 'right' })
  sheet0.column('C').width(60).style({ horizontalAlignment: 'left' })
  sheet0.column('D').width(60).style({ horizontalAlignment: 'left' })
  sheet0.column('E').width(60).style({ horizontalAlignment: 'left' })
  sheet0.column('F').width(12).style({ horizontalAlignment: 'right' })
  sheet0.column('G').width(12).style({ horizontalAlignment: 'right' })
  sheet0.cell('A1').value([
    [ 'OID', 'O*NET-SOC Code', 'O*NET-SOC Title',
      'Association Name', 'Association URL',
      'Prompt Number', 'Response Number',
    ]
  ])
  sheet0.range('A1:G1').style({ fill: 'ffff99', border: 'thin' })

  const sheet1 = xout.addSheet('Parameters')
  for (const col of 'AB'.split('')) {
    sheet1.column(col).style({ fontFamily: 'Times New Roman', fontSize: 12, wrapText: true,
verticalAlignment: 'top' })
  }
  sheet1.column('A').width(40).style({ horizontalAlignment: 'right' })
  sheet1.column('B').width(80).style({ horizontalAlignment: 'left' })
  sheet1.range('A1:B8').value([
    [ 'Model', model ],
    [ 'Prompt 1 - Template', promptTemplate1 ],
    [ 'Prompt 1 - N (Responses)', n1 ],
    [ 'Prompt 1 - Temperature (Creativity)', temp1 ],
    [ 'Prompt 2 - Template', promptTemplate2 ],
    [ 'Prompt 2 - N (Responses)', n2 ],
    [ 'Prompt 2 - Temperature (Creativity)', temp2 ],
    [ 'Total Cost', 0 ],
  ])
  sheet1.range('A1:A8').style({ bold: true, fill: 'ffff99', border: 'thin' })

  const outRows = []

  const xin = await XlsxPopulate.fromDataAsync(await readToBuffer(process.stdin))
  for (const sheet of xin.sheets()) {
    for (const row of sheet.usedRange().cells()) {
      const oid = parseInt(row[0].value())
```

```
if (!oid) {
  continue
}
const onetSocCode = row[1].value().toString().trim().replace(/\s+/g, ' ')
const onetSocTitle = row[2].value().toString().trim().replace(/\s+/g, ' ')
const onetSocDescription = row[3].value().toString().trim().replace(/\s+/g, ' ')
const onetSocTaskList = row[4].value().toString().trim()

console.warn(`Processing ${onetSocCode} - ${onetSocTitle}`)

const substitutions = {
  code: onetSocCode,
  title: onetSocTitle,
  desc: onetSocDescription
}
if (true) {
  const taskArray = onetSocTaskList.split(/\r?\n/)
  for (let i = 0; i < taskArray.length; ++i) {
    substitutions[`task${i + 1}`] = taskArray[i].trim().replace(/\s+/g, ' ')
  }
}

const seenAssociations = {}

const prompt1 = fillTemplate(promptTemplate1, substitutions)
const result1 = await callChat(prompt1, n1, temp1)
occsProcessed++
specificModel = result1.model
inputTokens += result1.usage.prompt_tokens
outputTokens += result1.usage.completion_tokens

for (let i = 0; i < result1.choices.length; ++i) {
  try {
    const mdata = JSON.parse(result1.choices[i].message.content)
    for (let assn of mdata) {
      if (!Object.hasOwn(seenAssociations, assn.url)) {
        outRows.push([ oid, onetSocCode, onetSocTitle, assn.name, assn.url, 1, i + 1 ])
        seenAssociations[assn.url] = true
      }
    }
  } catch (e) {
    console.warn(`Could not parse output from GPT (prompt 1, choice ${i})`)
  }
}

const prompt2 = fillTemplate(promptTemplate2, substitutions)
const result2 = await callChat(prompt2, n2, temp2, [
    { role: 'user', content: prompt1 },
    result1.choices[0].message,
])
inputTokens += result2.usage.prompt_tokens
outputTokens += result2.usage.completion_tokens

for (let i = 0; i < result2.choices.length; ++i) {
  try {
    const mdata = JSON.parse(result2.choices[i].message.content)
    for (let assn of mdata) {
      if (!Object.hasOwn(seenAssociations, assn.url)) {
        outRows.push([ oid, onetSocCode, onetSocTitle, assn.name, assn.url, 2, i + 1 ])
```

```
              seenAssociations[assn.url] = true
            }
          }
        } catch (e) {
          console.warn(`Could not parse output from GPT (prompt 2, choice ${i})`)
        }
      }
    }
  }

  sheet0.cell('A2').value(outRows)
  sheet1.cell('B1').value(`${model} (${specificModel})`)
  sheet1.cell('B8').value('$' + Number.parseFloat((inputTokens * inputTokenCost) +
(outputTokens * outputTokenCost)).toFixed(2).toString())
    process.stdout.write(await xout.outputAsync())

})().catch(err => {
  console.error(err)
  process.exitCode = 1
  process.exit()
})
```