Automated title and abstract screening for scoping reviews using the GPT-4 Large Language Model

Introduction

A scoping review is a relatively novel type of literature review that aims to map the key concepts and existing activity within an area of research [@Arksey.2005]. Like systematic reviews, scoping reviews typically use rigorous, transparent [@Pham.2014], and sometimes pre-registered methods for gathering and synthesising evidence, and increasingly use formal frameworks for both performing and reporting reviews [@Peters.2021]. Scoping reviews can inform future systematic reviews or primary research in the same area [@Sutton.2019]. However, they differ from systematic reviews in focussing on describing the breadth of coverage of the available literature rather than research findings in depth [@Arksey.2005].

Frameworks for performing a scoping review typically involve defining a research area or question, searching bibliographic databases for potentially relevant published material ('sources'), screening these sources to identify those relevant to the area or question, and systematically extracting and reporting data from the sources [@JBI.2015, @Arksey.2005]. The screening stage will usually involve initial screening of source titles and abstracts against pre-determined inclusion and exclusion criteria, followed by screening of the full text of sources, with both steps performed in replicate by at least two human reviewers [@Peters.2020, @Pham.2014]. Because database searches can return many hundreds or thousands of potentially relevant sources, these screening steps can require intensive human effort. Many software methods have been proposed or used to support or partially automate source screening for scoping reviews, including text mining to prioritise potentially more relevant sources for human screening [@Shemilt.2014, @Howard.2016, @Chai.2021], automated clustering and labelling of sources to support human decision-making [@Stansfield.2013], and 'crowdsourcing' screening to untrained workers via online platforms [@Mortensen.2017]. A similar but more extensive set of methods have been developed and employed for systematic reviews [@Khalil.2022, @Gates.2019] for which the process of source screening is broadly comparable.

Since the release of the first Generative Pre-trained Transformer (GPT) Large Language Model (LLM) by OpenAI (San Francisco, California, United States of America) in 2018 [@Radford.2018], transformer-based LLMs and the GPT lineage in particular have seen rapid and widespread adoption for a range of automation tasks. Broadly, these models generate a probabilistically weighted list of tokens (parts of text such as letter combinations and punctuation) which may continue or complete some input text (a 'prompt'), having been trained to do so by practising such predictions on large human-written corpora. When this generative process is iterated, it allows for a range of applications involving analysis and production of text, such as summarising articles, generating fiction in a particular genre or style, or conversing with a human [@OpenAI.2023].

While LLMs are not yet widely used to screen sources for literature reviews, early work suggests they may perform well in this role. Guo et al. [@Guo.2023] reported the use of a GPT-lineage model (they do not specify which, though their published code suggests OpenAI's 'gpt-3.5-turbo' model) to screen 24,307 titles and abstracts from five systematic and reviews one scoping review, achieving pooled sensitivity of 76% and specificity of 91% when compared to human reviewers. Their approach involves giving the model a brief prompt instructing it to take on the persona of a researcher screening titles and abstracts, and to respond with a decision to include or exclude a source, followed by the source's title and abstract as well as the inclusion and exclusion criteria. Syriani et al. [@Syriani.2023] similarly reported the use of 'gpt-3.5-turbo' to screen titles and abstracts for a systematic review and achieved sensitivities of above 70%. They also systematically evaluated prompts given to the LLM to identify a prompt that performed best at the screening task; their chosen prompt, like that of Guo et al., placed the LLM in the role of an academic reviewer.

Both of these approaches made use of a single, fixed text prompt template, which the LLM then completes with additional text representing its response (the decision to include or exclude a source), a method sometimes

called 'one-shot prompting'. Recent work has identified a number of methods which can be superior to one-shot prompting when using LLMs for tasks that require complex or multi-step reasoning. These methods include 'chain-of-thought prompting' [@Wei.2022], in which a complex task is broken down into a series of intermediate steps that the model is prompted to complete in sequence, and the 'tree of thoughts' strategy [@Yao.2023], in which multiple parallel chains of thought are generated, compared, and integrated.

In this paper, I introduce a package for the R programming language [@R.2023] called GPTscreenR, and evaluate its performance in screening titles and abstracts for scoping reviews. The purpose of this package is to assist and augment rather than replace human reviewers in performing scoping reviews. This paper and the associated package represent four novel developments in the use of LLMs for source screening in literature reviews. Firstly, they provide an open-source software package which can be downloaded and used as well as modified by academic reviewers. Secondly, they provide the first such application of LLMs specifically for scoping reviews, though pragmatically approach would also be applicable for systematic reviews with minimal changes. Thirdly, they provide the first report on the accuracy of this application using the most recent iteration of the GPT model lineage, GPT-4. Finally, they incorporate the use of chain-of-thought reasoning in an effort to maximise the accuracy of screening decisions.

Methods

The GPTscreenR package

GPTscreenR is an R [@R.2023] package released under the MIT open source licence. The source code is available for download from GitHub at https://github.com/wilkox/GPTscreenR. At the time of writing the most recent package version was 0.0.1; the results presented in this paper were obtained with an earlier version 0.0.0.9005, since which some small changes have been made to further refine performance.

GPTscreenR consists of two main components. The first is a set of internal functions for interfacing with the OpenAI API and for representing and manipulating 'conversations' with the GPT-4 LLM. These functions are designed to be model-agnostic, allowing for different models to be used in future versions or by users with particular needs. The OpenAI API requires an OpenAI account, and OpenAI charges fees for use of the API. In order to access the API, GPTscreenR requires a secret key to be registered prior to the use of the source screening functions, and instructions for doing so are provided in the package documentation and on loading of the package in R if the key has not been correctly registered.

The second component is a set of user-facing functions for screening sources for a scoping review. The `review_description()` function assists in generating a text description of the review's objectives and inclusion and exclusion criteria, using the Population, Concept, and Context (PCC) frame [@Peters.2020] for defining the review's inclusion and exclusion criteria. The use of this framework and function is optional, and users may instead choose to provide a description of the review and criteria for source selection using any framework or format they see fit.

The `screen_source()` function performs the main task of the package. This function mediates an conversation with GPT-4 in which chain-of-thought prompting [@Wei.2022] is used to guide the model through screening a source title and abstract against the study inclusion criteria. The complete template for this conversation is given in Figure 1. The OpenAI API defines a conversation as a series of messages, each of which originates from one of three roles: 'system', representing an authoritative voice that can instruct the model on its task and behaviour; 'user', representing a human user that can interact with the model; and 'assistant', representing the responses generated by the model. In `screen_source()`, the 'system' role is used to give the model general instructions, while the 'user' role is used to provide the user-written review description and the source title and abstract.

The phrases 'Let's work step by step' and 'Let's continue to work step by step', used each time 'system' instructs the model to generate a response, are derived from the 'Let's think step by step' prompt phrase which significantly improves LLM performance on multi-step reasoning tasks with a single prompt and no examples ('zero-shot') [@Kojima.2022], adapted to this chain-of-thought approach. This approach thus attempts to maximise engagement of the model's multi-step reasoning capabilities both across and within each step in the screening task.

The model is serially instructed to summarise the inclusion criteria for the scoping review (prior to being presented with the title and abstract to be screened), compare the source title and abstract against these summarised criteria, and make a final recommendation (Fig. 1). Following the chain-of-thought approach, an

example of summarised inclusion criteria is given for the model to use as a template or exemplar. This approach was chosen after noting that a major source of type II error (false positives) when attempting to screen sources with zero-shot or one-shot prompts (i.e. with a single prompt and no or some examples) was that the model would fail to consider important inclusion criteria. For example, Fig 2a presents a conversation with GPT-4 using a zero-shot prompt. An intentionally adversarial screening task has been constructed, in which in order to correctly recommend exclusion of the source, the model must notice that the review is looking for research on therapy alpaca interventions while the source reports on a therapy camel intervention. The presence of multiple other inclusion criteria which are met by the source, as well as the mention of alpacas in the source abstract, serve as distractors. In this example, GPT-4 incorrectly recommends inclusion. If the conversation is then continued to draw the model's attention to the error, it is able to identify and correct it (Fig 2b), suggesting that the error arises from a failure of the model to properly consider the relevant inclusion criterion rather than an inability to do so. Using the chain-of-thought approach overcomes this problem (Fig 2c). The model identifies 'The source examines the impact of therapy alpaca programmes' as an inclusion criterion, and when asked to compare the source against the inclusion criteria correctly assesses that this is the only criterion is not met. The model then correctly recommends exclusion of the source.

The `screen_source()` function returns a list comprising the complete transcript of the conversation with the model and the model's final recommendation, either the word 'INCLUDE' or 'EXCLUDE'. The conversation transcript can be used to interrogate cases where the model may have returned an incorrect or unexpected result. The package also provides a function `screen_sources()`, which applies `screen_source()` iteratively to a data frame of sources. `screen_sources()` caches its results to a file after each source, so that screening can be split across multiple sessions and recover from interruptions.

Validation

To validate GPTscreenR's approach, six scoping reviews were identified from the Open Science Framework (OSF; https://osf.io) where the review inclusion criteria and the results of title and abstract screening had been made publicly available. A summary of the review characteristics is provided in Table 1. Small random subsets of screened sources from four of the reviews (`COVID_misinfo`, `smartphones`, `solastalgia`, and `teachers`) were used during initial testing and refinement of the `screen_source()` function, while the full set or random subsets from all six reviews were used for final validation. Random subsets were used where the large number of sources available for screening was prohibitive in time or cost. The total number of sources available for screening and the number used for validation from each review are given in Table 1.

Some of the reviews did not include the full abstract text in the publicly available files, and where these abstracts could not be obtained from external databases these sources were excluded from validation. There were also many cases where missing, malformed, or duplicate data required either manual correction or exclusion of sources. The scoping review data, code used to prepare this data for validation, and code for calculating summary statistics are available in a reproducible form in the package repository on GitHub (https://github.com/wilkox/GPTscreenR/tree/master/validation).

The final human reviewer decision at the title and abstract screening level was used as the gold standard outcome for each source. Sensitivity and specificity were calculated by comparing GPT's recommendation against the gold standard. Three of the scoping reviews (`COVID_misinfo`, `solastalgia`, and `teachers`) included individual human reviewer decisions in addition to the final decision in their publicly available datasets, and these were used to calculate intraobserver agreement (Cohen's kappa) using the R function `cohen.kappa()` from the `psych` package [@Revelle.2023].

Results

1,138 sources were screened from the six scoping reviews. GPTscreenR achieved a weighted average sensitivity of 0.75 and weighted average specificity of 0.87. For the three reviews that provided individual reviewer decisions, the weighted average Cohen's kappa was 0.67, while the weighted average Cohen's kappa between human and GPT decisions was 0.53.