Leveraging Natural Language Processing and Data Mining to Augment and Validate APIs

Decrop, Alix

Published in:
ISSTA 2024 - Proceedings of the 33rd ACM SIGSOFT International Symposium on Software Testing and Analysis

DOI:
10.1145/3650212.3685554

Publication date:
2024

Document Version
Peer reviewed version

Leveraging Natural Language Processing and Data Mining to Augment and Validate APIs

Alix Decrop
alix.decrop@unamur.be
NADI, University of Namur
Namur, Belgium

Abstract

APIs are increasingly prominent for modern web applications, allowing millions of users around the world to access data. Reducing the risk of API defects - and consequently failures - is key, notably for security, availability, and maintainability purposes. Documenting an API is crucial, allowing the user to better understand it. Moreover, API testing techniques often require formal documentation as input. However, documenting is a time-consuming and error-prone task, often overlooked by developers. Natural Language Processing (NLP) could assist API development, as recent Large Language Models (LLMs) demonstrated exceptional abilities to automate tasks based on their colossal training data. Data mining could also be utilized, synthesizing API information scattered across the web. Hence, I present my PhD project aimed at exploring the usage of NLP-related technologies and data mining to augment and validate APIs. The research questions of this PhD project are: (1) What types of APIs can benefit from NLP and data mining assistance? (2) What API problems can be solved with such methods? (3) How effective are the methods (*i.e.* LLMs) in assisting APIs? (4) How efficient are the methods in assisting APIs (*i.e.* time and costs)?

CCS Concepts

• **Computing methodologies** → **Natural language processing**; • **Software and its engineering** → **Software testing and debugging**.

Keywords

API, NLP, LLM, Data Mining, Software Testing, Automation.

ACM Reference Format:
Alix Decrop. 2024. Leveraging Natural Language Processing and Data Mining to Augment and Validate APIs. In *Proceedings of the 33rd ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA '24), September 16–20, 2024, Vienna, Austria.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3650212.3685554

1 Introduction and Motivation

Application Programming Interfaces (APIs) allow computer software or components to communicate with one another. APIs hide underlying code and logic behind a simplified interface for the end user. Documentation allows users to understand how to use an API through its interface. An API is a broad term that encompasses various types of APIs; Popular examples include RESTful APIs [11, 22], GraphQL APIs [9], SOAP APIs [16], and framework-specific APIs (*i.e. TensorFlow* [15]). As APIs are increasingly popular and leveraged by millions of users, testing APIs is extremely important. API testing allows the detection of defects, preventing certain failures from happening before the production phase. This practice also encompasses non-functional requirements such as security, performance, availability, reliability, and maintainability. When carried out early in the software development life cycle, API testing can save time and money as defects are detected and handled directly. For the case of RESTful APIs, testing commonly uses a black-box approach, requiring an OpenAPI Specification (OAS) [12] of the API under test as input [8, 13, 23]. As a result, automated and formal specification generation has been explored in the literature. Yet, existing methods require some form of documentation [3, 14, 18], an HTTP proxy server [24], to crawl the API user interface [27, 28] or to use API call examples [7]. However, two major problems arise. (1) APIs are increasingly complex and developed at a rapid pace, rendering the testing phase more labor-intensive. (2) API developers tend to omit or overlook documentation, resulting in misuse and reducing the efficiency of automated analysis tools depending on it. Accordingly, the API field is in dire need of assistance.

In 2023, the chatbot *ChatGPT* [20] utilizing the GPT model architecture [21] popularized LLMs, a breakthrough in NLP. Such models are fed with enormous amounts of training data gathered from across the Internet. When used correctly, LLMs are essential tools for handling various tasks across different application domains. Popular models include the influential BERT [6], GPT-3.5 and GPT-4 [1], Llama [26], Gemini [25] and Claude. To illustrate LLM uses, Deng et al. [5] explored feeding masked code to an LLM to test deep-learning libraries. Meng et al. [19] used LLMs to guide protocol fuzzing, with a message mutation process via LLM interactions. Moreover, Khanfir et al. [17] applied token masks onto code to obtain mutated code from a CodeBERT [10] variant. Similarly, data mining can assist certain tasks by synthesizing information. By scraping data from the web - sometimes hidden from manual browsing - data mining can offer this data in a single location for advanced analysis. On top of that, I hypothesize that *API-specific knowledge can be retrieved from multiple sources on the web* (*i.e.* documentation from official API websites, usage-related questions from blogs or forums, new API feature announcements from social networks). In parallel, *NLP technologies are trained on this kind of data, rendering NLP well-suited for API knowledge synthesis, augmentation, and validation.* In this spirit, "Leveraging Natural Language Processing and Data Mining to Augment and Validate APIs" is the title of this PhD project. The overall objective of this research is to

ISSTA '24, September 16–20, 2024, Vienna, Austria

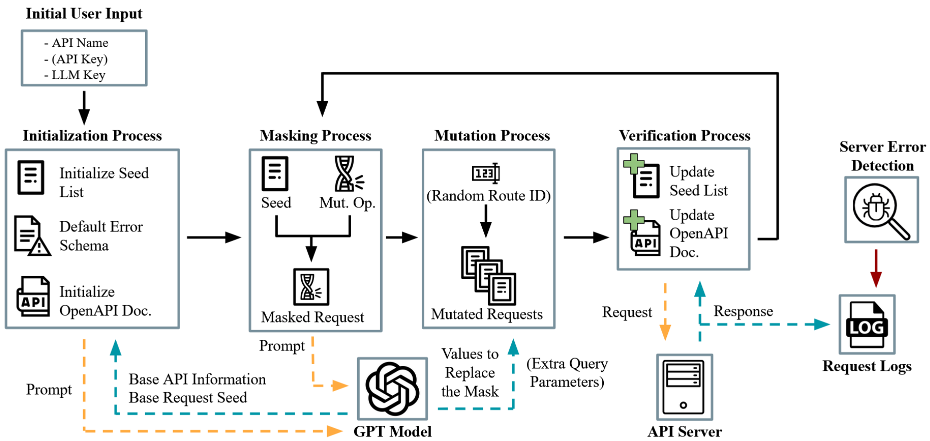

Figure 1: Overview of RESTSpecIT [4]

discover API-related tasks that can be assisted (or even automated) with the help of NLP (*i.e.* LLMs) and data mining.

2 Preliminary Results

We recently explored an application domain for this PhD project, by leveraging a LLM to assist RESTful APIs. The approach is termed *RESTSpecIT*, using the GPT-3.5 LLM to automatically infer OpenAPI specifications and test RESTful APIs in a black-box environment. Compared to state-of-the-art API inference and testing tools, RESTSpecIT requires minimal user input: The name of the API and a key for LLM requests. RESTSpecIT uses an in-context *prompt masking* strategy to retrieve relevant API data from the LLM, requiring no model fine-tuning. Based on the model's responses, the tool can mutate HTTP requests. By sending mutated requests to the API and analyzing HTTP responses returned, the OpenAPI specification can be inferred with request data. The API can be tested by uncovering 5xx status codes (server errors) in responses. Our evaluation demonstrates that RESTSpecIT: (1) effectively infers API specifications in the OpenAPI format, with 85.05% of GET routes and 81.05% of query parameters found on average, (2) discovers undocumented and valid API data, (3) is efficient regarding requests sent, execution time, and model costs, and (4) serves RESTful API testing. Indeed, RESTSpecIT can uncover server errors in APIs and generate valid OpenAPI specifications that can be used as input for existing testing tools (e.g. *RESTler* [2]). Figure 1 presents an overview of the tool. The results obtained from this research are promising, demonstrating that LLMs are suitable candidates for assisting/automating RESTful API tasks [4].

3 Research Questions

RQ.1: ***What types of APIs can benefit from NLP and data mining assistance?*** As detailed in Section 1, there exists various types of APIs. Moreover, Section 2 displayed that RESTful APIs could benefit from our approach, which is a promising result. Other types of APIs need to be explored to generalize the approach.

RQ.2: ***What API problems can be solved with such methods?*** APIs have different problems; For instance, RESTful APIs have problems related to HTTP requests (*e.g.* validity, false positive parameters, routes). However, framework-specific APIs do not have the same problems. Recurring problems in different API types need to be identified and they need to be solved with the approach.

RQ.3: ***How effective are the methods in assisting APIs?*** (1) NLP and LLMs are constantly evolving. Before 2023, BERT was the prominent model. However, the GPT-3.5 model arose in 2023. Not so long after, the field adopted a plethora of different models such as GPT-4, Gemini, Claude, Bard, Llama, and so on. In this spirit, comparing the effectiveness of LLMs (closed/open source, online/local) is important. (2) The effectiveness of the approach is also important, depending on the API metrics defined.

RQ.4: ***How efficient are the methods in assisting APIs?*** In parallel to RQ3, this last research question aims to evaluate the efficiency of the approach. This will comprise metrics such as model response times, model costs, and local resource usage.

4 Conclusion and Work Plan

This PhD research project aims at augmenting and validating APIs, by leveraging NLP-related technologies (*e.g.* LLMs) and data mining to synthesize web data. Our preliminary approach automatically infers documentation and tests RESTful APIs. However, I will explore different API types, *e.g.*, GraphQL, SOAP, and framework-specific APIs. For my first year, I currently focus on RESTful APIs and evaluating the approach (*c.f.* Section 2). In the following years, I will extend the approach to other API types and particularly focus on research question 3 regarding the NLP landscape's evolution.s

Acknowledgments

I started my PhD (6-year research and teaching assistant program) in September 2023 at the University of Namur. My supervisors are Dr. Gilles Perrouin and Prof. Pierre-Yves Schobbens (UNamur). I thank Dr. Xavier Devroey (UNamur) and Prof. Michail Papadakis (SnT/Univ. Luxembourg) for being part of my thesis committee.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774* (2023).

[2] Vaggelis Atlidakis, Patrice Godefroid, and Marina Polishchuk. 2019. Restler: Stateful rest api fuzzing. In *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*. IEEE, 748–758.

[3] Hanyang Cao, Jean-Rémy Falleri, and Xavier Blanc. 2017. Automated generation of REST API specification from plain HTML documentation. In *Service-Oriented Computing: 15th International Conference, ICSOC 2017, Malaga, Spain, November 13–16, 2017, Proceedings*. Springer, Springer-Verlag GmbH, Heidelberg, 453–461.

[4] Alix Decrop, Gilles Perrouin, Mike Papadakis, Xavier Devroey, and Pierre-Yves Schobbens. 2024. You Can REST Now: Automated Specification Inference and Black-Box Testing of RESTful APIs with Large Language Models. *arXiv preprint arXiv:2402.05102* (2024).

[5] Yinlin Deng, Chunqiu Steven Xia, Haoran Peng, Chenyuan Yang, and Lingming Zhang. 2023. Large Language Models are Zero-Shot Fuzzers: Fuzzing Deep-Learning Libraries via Large Language Models. In *Proceedings of the 32nd ACM SIGSOFT international symposium on software testing and analysis*. ACM, New York, NY, 423–435.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805 [cs.CL]

[7] Hamza Ed-Douibi, Javier Luis Cánovas Izquierdo, and Jordi Cabot. 2017. Example-driven web API specification discovery. In *European Conference on Modelling Foundations and Applications*. Springer, Springer-Verlag GmbH, Heidelberg„ 267–284.

[8] Adeel Ehsan, Mohammed Ahmad ME Abuhaliqa, Cagatay Catal, and Deepti Mishra. 2022. RESTful API testing methodologies: Rationale, challenges, and solution directions. *Applied Sciences* 12, 9 (2022), 4369.

[9] Facebook. 2024. GraphQL. https://graphql.org/.

[10] Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, et al. 2020. Codebert: A pre-trained model for programming and natural languages.

[11] Roy Thomas Fielding. 2000. *Architectural styles and the design of network-based software architectures*. University of California, Irvine.

[12] Linux Foundation. 2022. OpenAPI Specification. https://www.openapis.org. [Online; Last Accessed 5-December-2023].

[13] Amid Golmohammadi, Man Zhang, and Andrea Arcuri. 2023. Testing RESTful APIs: A Survey. *ACM Trans. Softw. Eng. Methodol.* 33, 1, Article 27 (nov 2023), 41 pages. https://doi.org/10.1145/3617175

[14] César González-Mora, Cristina Barros, Irene Garrigós, Jose Zubcoff, Elena Lloret, and Jose-Norberto Mazón. 2023. Improving open data web API documentation through interactivity and natural language generation. *Computer Standards & Interfaces* 83 (2023), 103657.

[15] Google. 2024. TensorFlow. https://www.tensorflow.org.

[16] Martin Gudgin, Marc Hadley, Noah Mendelsohn, Jean-Jacques Moreau, Henrik Frystyk Nielsen, Anish Karmarkar, and Yves Lafon. 2003. SOAP Version 1.2. *W3C recommendation* 24 (2003), 12.

[17] Ahmed Khanfir, Renzo Degiovanni, Mike Papadakis, and Yves Le Traon. 2023. Efficient Mutation Testing via Pre-Trained Language Models. *ArXiv* abs/2301.03543 (2023), 1–14. https://api.semanticscholar.org/CorpusID:255545954

[18] Myeongsoo Kim, Davide Corradini, Saurabh Sinha, Alessandro Orso, Michele Pasqua, Rachel Tzoref-Brill, and Mariano Ceccato. 2023. Enhancing REST API Testing with NLP Techniques. In *Proceedings of the 32nd ACM SIGSOFT International Symposium on Software Testing and Analysis*. ACM, New York, NY, 1232–1243.

[19] Ruijie Meng, Martin Mirchev, Marcel Böhme, and Abhik Roychoudhury. 2024. Large Language Model guided Protocol Fuzzing. In *31st Annual Network and Distributed System Security Symposium (NDSS) 2024*. The Internet Society, Reston, USA.

[20] OpenAI. 2024. ChatGPT. https://chat.openai.com.

[21] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

[22] Leonard Richardson, Mike Amundsen, and Sam Ruby. 2013. *RESTful Web APIs: Services for a changing world*. " O'Reilly Media, Inc.".

[23] Abhinav Sharma, M Revathi, et al. 2018. Automated API testing. In *2018 3rd International Conference on Inventive Computation Technologies (ICICT)*. IEEE, New York, USA, 788–791.

[24] Sheikh Mohammed Sohan, Craig Anslow, and Frank Maurer. 2015. Spyrest: Automated restful API documentation using an HTTP proxy server (N). In *2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE, New York, NY, 271–276.

[25] Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805* (2023).

[26] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971* (2023).

[27] Rahulkrishna Yandrapally, Saurabh Sinha, Rachel Tzoref-Brill, and Ali Mesbah. 2023. Carving UI Tests to Generate API Tests and API Specification. In *Proceedings of the 45th International Conference on Software Engineering* (Melbourne, Victoria, Australia) *(ICSE '23)*. IEEE Press, New York, USA, 1971–1982. https://doi.org/10.1109/ICSE48619.2023.00167

[28] Jinqiu Yang, Erik Wittern, Annie TT Ying, Julian Dolby, and Lin Tan. 2018. Towards Extracting Web API Specifications from Documentation. In *Proceedings of the 15th International Conference on Mining Software Repositories*. ACM, New York, USA, 454–464.