Exploring Python Programming Technology Applications in Scientific Journal Editing and Publishing: Postprint

Authors: Long Yali, Wang Quanwei, Lin Haimei, Wang Huiyuan

Date: 2023-10-08T00:00:00+00:00

Abstract

[Purpose] To explore integrated methodologies for scientific journal editing and publishing operations. [Method] Using Python as an exemplar, this study investigates the efficacy of programming technology applications within scientific journal editorial workflows. [Result] The deployment of Python programming technology in scientific journal editorial tasks can harness automated office programming capabilities to resolve repetitive operations such as automatic generation of acceptance notifications and automated email dispatch; Python enables the automatic updating and supplementation of expert databases; moreover, the implementation of Python programming for topic planning in scientific journals, and even the complete automation of journal editing and publishing processes, are viable prospects. [Conclusion] Python programming technology can be integrated into scientific journal editing and publishing workflows to streamline operational processes and enhance work efficiency.

Full Text

Preamble

Exploring the Application of Python Programming Technology in Scientific Journal Editing and Publishing

Long Yali, Wang Quanwei, Lin Haimei, Wang Huiyuan*

Institute of Scientific and Technical Information, Chinese Academy of Tropical Agricultural Sciences / Hainan Key Laboratory of Tropical Crop Information Technology Application, Haikou, Hainan 571101, China

Abstract

Objective: To explore integrated approaches for scientific journal editing and publishing workflows. **Method:** Using Python as an example, this study investigates the application effects of programming technology in scientific journal editing and publishing. **Results:** When applied to editorial office operations, Python programming technology can automate repetitive tasks such as automatically generating acceptance notices and sending emails. Python enables automatic updating and supplementation of expert databases and shows potential for automating topic planning and even the entire editorial and publishing process. **Conclusion:** Python programming technology can be integrated into scientific journal editing and publishing workflows to streamline processes and improve work efficiency.

Keywords: Python programming technology; scientific journals; workflow; work efficiency

Introduction

The innovative development and prosperity of journals cannot be separated from the support of new technologies. The invention of paper and printing laid the foundation for the formation and circulation of early journals, while computers and the internet serve as the carriers for their rapid dissemination and development. The essence of how computers and the internet promote journal development lies in programming languages. Therefore, mastering certain programming technologies and applying them to scientific journal editing and publishing workflows can yield remarkable results.

A search of CNKI using the theme "programming + scientific journals" retrieved 22 results, which after data cleaning and screening yielded 10 valid entries, including 9 journal articles and 1 master' s thesis. These studies primarily focused on the application of programming technology in journal editing and publishing systems. For example, Wang Yue et al. [1] studied InDesign JavaScript programming and its application in scientific journal typesetting, detailing how InDesign JavaScript can be used to add line numbers in proofs. Deng Junqi et al. [2] designed and implemented an Internet/Intranet-based scientific journal management information system using the then-popular Browser/Server model, ASP programming, and an SQLServer 2000+Windows 2000 Server+ASP environment, achieving computerized network management of the entire editorial workflow and integrated online submission, review, editing, and proofreading, which greatly improved editorial office efficiency. Regarding Python programming in journal editing and publishing, only Zhou Li et al. [3] have studied its application in integrating internal editorial office operations, primarily using Python to generate information forms for interfacing with journal partner banks, editorial department finance departments, and courier companies (such as Cainiao and other courier apps), and implementing automatic printing functions by creating corresponding templates based on personalized editorial needs

(such as different printing media, font sizes, positions, and spacing).

Python is a high-level computer programming language that has entered the top three in the famous TIOBE programming language rankings and currently ranks first in the artificial intelligence programming field [4]. Designed in the late 1980s by the Dutch National Research Institute for Mathematics and Computer Science [5], Python serves as a computational ecological language with rich data structures [6], extensive module support, and third-party libraries (approximately 120,000 libraries according to official statistics) covering nearly every domain (such as artificial intelligence, big data analytics, web parsing, cyberspace, human-computer interaction, and design). Its code is concise [7], with clear language logic and straightforward thinking, making it the programming language closest to natural language. Importantly, it is free and open-source, gradually gaining favor among scientists, scholars, technicians, and students alike. Universities have also recognized Python' s importance, with some making it a required course [8].

Exploring the application of Python programming technology in scientific journal editing and publishing—for instance, in automated office operations, web scraping, and data analysis—holds promise for simplifying cumbersome publishing workflows, improving efficiency, enhancing personalized service capabilities, increasing the precision of topic planning, and thereby boosting journal impact.

1. Application of Python Programming in Scientific Journal Editorial Office Operations

Editorial office work forms the foundation of journal publishing, encompassing tasks such as receiving submissions, initial review, sending manuscripts for external review, collecting reviewer comments, requesting revisions, and scheduling issues, with some staff also handling daily administrative affairs. Although essential, this work is complex and repetitive, easily leading to fatigue and burnout. Simplifying these processes and improving efficiency is therefore imperative. Python can freely install libraries and modules related to office automation, such as the python-docx library for Word processing (third-party library, installation command: pip install python-docx), the openpyxl library for Excel (third-party library, installation command: pip install openpyxl), and the smtplib module for email (built into Python). These tools enable batch processing of similar tasks, streamlining workflows and enhancing efficiency.

Take batch-sending acceptance notices as an example. Sending acceptance notices to authors is an important editorial task typically performed manually—writing each notice individually and sending them via email one by one (unless the submission system has this function). Each notice uses a fixed template, differing only in author name and article title, and each email can use the same template. Despite its simplicity, this task involves heavy repetition and wastes considerable time. Python's office automation modules can solve such repetitive tasks and improve efficiency. Figure 1 [Figure 1: see original paper] shows a

partial excerpt of an author information registration form from an editorial office. Traditionally, staff manually copy author names, paper titles, first-author affiliations, and scheduled issue information into acceptance notice templates, a process that is not only time-consuming but also error-prone. The code in Figure 2 [Figure 2: see original paper] reads relevant content from the author information registration form to automatically generate new acceptance notices (Figure 3 [Figure 3: see original paper]) and saves them in designated folders. Then, the code in Figure 4 [Figure 4: see original paper] automatically sends these notices to authors via email. This entire process takes only a few seconds to complete, and the completed code can be saved for reuse, eliminating low-level errors such as mixing up author information due to staff oversight. Moreover, the relevant "fields" in the code can be replaced as needed, making this code adaptable for use in different editorial offices.

2. Application of Python Programming in Scientific Journal Expert Database Construction

Establishing an expert database is of great significance to editorial offices, as expert resources constitute core operational assets. Expert database construction should emphasize both maintaining existing resources and gradual expansion and improvement. Literature review reveals that expert database sources typically include: (1) editorial board lists; (2) personal networks; (3) registration information from submission websites; (4) various officially published paper expert directories, such as the "Chinese Scientific and Technical Paper Review Expert Directory," "Chinese University Natural Science Journal Reviewer Directory," and "Doctoral Admissions Brochure" ; (5) major domestic journal databases such as CNKI, Wanfang, and VIP; and (6) internet collection (Shi Pengliang et al.). Information obtained from these channels can be integrated and supplemented to form a comprehensive expert database. Manual collection and entry of this information would be extremely time-consuming. Our research team explored using Python web scraping code to batch download (scrape) publicly available expert information and automatically store it, enabling repeated use of the same code for automatic updates.

When scraping information from a website, it is necessary to first check the network protocol (Robots protocol), and all web scraping behavior must comply with it. Taking CNKI as an example, the Robots protocol is shown in Figure 5 [Figure 5: see original paper]. The expert information we need to scrape is not prohibited, so we can collect expert information within the specified time frame. We can also download papers and automatically extract author information to supplement the expert database. Figure 6 [Figure 6: see original paper] shows partial code for automatically extracting author information using Python to supplement the expert database, and Figure 7 [Figure 7: see original paper] displays partial information from the expert database created by this automated program.

3. Application Prospects of Python Programming in Scientific Journal Topic Planning

As the saying goes, "preparation ensures success, while lack of preparation leads to failure." Topic planning represents the "preparation" for scientific journal publishing and is as crucial as a work plan for both journals and authors. Effective topic planning helps authors plan their research output and enables journals to arrange publishing schedules appropriately. Using Python to provide novel, efficient, and accurate topic planning for scientific journals is entirely feasible. For example, web scraping technology can obtain annual information on various types of funded projects, automatically extracting publicly available details such as project names, author information, and completion dates. After automatically classifying and screening these projects to identify those relevant to the journal' s scope and their authors, the system can generate precise and efficient topic planning columns. Based on these analysis results, editors can conduct targeted solicitation, achieving twice the result with half the effort.

4. Application Vision of Python Programming in Scientific Journal Publishing Workflow

Based on the specific application explorations above, Python programming is demonstrably feasible for implementation in scientific journal publishing workflows. Following this approach, we have designed a conceptual framework for applying Python programming in scientific journal publishing workflows, detailed in Figure 8 [Figure 8: see original paper]. As shown, the editorial and publishing workflow can be broadly divided into two main processes: manuscript processing and editing/publishing. During the manuscript processing stage, Python programs can be designed for intelligent processing of author submissions, automatically rejecting non-compliant manuscripts and intelligently categorizing acceptable ones into appropriate columns. Categorized manuscripts can then undergo intelligent editing and processing, followed by intelligent typesetting and proofreading, and finally intelligent printing and distribution. The core technology underlying this vision is Python' s automatic learning capability.

The implementation challenge lies in the detailed control and debugging of each program step. For instance, during manuscript processing, the program must be pre-configured with criteria for rejection, such as keywords and article structures. Similarly, column assignment requires pre-defined associations between papers and columns. The editing stage can be pre-configured with standardized paper formats, while the automatic typesetting stage may encounter errors. All these issues require long-term data collection and continuous system modification and improvement to achieve.

References

[1] Wang Y, Liu Q. InDesign Javascript Programming and Its Application in

Typesetting Scientific Journal Papers—Adding Line Numbers in Proofs [J]. Chinese Journal of Scientific and Technical Periodicals, 2014(1): 104-108.

[2] Deng J, Yu X, Xiang L, et al. Design and Implementation of an Internet/Intranet-based Scientific Journal Management Information System [J]. Chinese Journal of Scientific and Technical Periodicals, 2003(5): 515-517.

[3] Zhou L, Zhang Y, Zeng Y, et al. Practice and Reflection on Integrating Editorial Office Work Based on Python Programming [J]. Chinese Journal of Scientific and Technical Periodicals, 2021(3): 303-308.

[4] Liang Z. Research on Satellite Antenna Layout and Electromagnetic Compatibility Prediction [D]. Nanjing: Nanjing University of Science and Technology, 2018.

[5] Wang Y, Yue L. Application of Python in Student Performance Analysis [J]. Electronic Technology & Software Engineering, 2021(4): 173-175.

[6] Wu Z. Comprehensive Python Development Technology [M]. Beijing: Mechanical Industry Press, 2020: 3.

[7] Cao Q. Application of Python Technology in Journal Evaluation [J]. Computer Programming Skills & Maintenance, 2022(1): 42-44.

[8] Tao W, Hou D. Discussion on Python Application Based on Problem Solving —Taking Batch Processing Excel Files with Python as an Example [J]. Journal of Ezhou University, 2019(1): 107-109+112.

[9] Xinhua News Agency. Xi Jinping: Hold High the Great Banner of Socialism with Chinese Characteristics and Strive in Unity to Build a Modern Socialist Country in All Respects—Report at the 20th National Congress of the Communist Party of China [EB/OL]. https://gongyi.sohu.com/a/595415004_{501883}. 2022-10-16/2023-01-12.

[10] National Press and Publication Administration. Development Plan for the Publishing Industry during the 14th Five-Year Plan Period [EB/OL]. https://new.qq.com/rain/a/20231231A0DGRB00. 2021-12-28/2023-01-12.

[11] Xiang S. Empowerment of Artificial Intelligence for the Intelligent Transformation and Integrated Development of Academic Journals [J]. Publishing Wide Angle, 2022(18): 81-84.

[12] Jia M. Analysis of Medical Journal Editing and Publishing Strategies Based on Artificial Intelligence [J]. Journal of News Research, 2022(16): 226-228.

[13] Tao Q, Zheng Y. Research on Artificial Intelligence Application in Chinese Scientific Journal Publishing [J]. Chinese Journal of Scientific and Technical Periodicals, 2022(6): 735-745.

Author Biographies

Long Yali (1991-), female, from Bijie, Guizhou, Editor. Research direction: journal editing and publishing.

Wang Quanwei (1979-), female, from Yiyang, Hunan, Associate Editor-in-Chief. Research direction: scientific journal editing and publishing.

Lin Haimei (1982-), female, from Haikou, Hainan, Associate Editor-in-Chief. Research direction: scientific journal editing and publishing.

Wang Huiyuan (1991-), female, from Daqing, Heilongjiang, Editor. Research direction: journal editing and knowledge publishing.

(Executive Editor: Zhang Xiaojing)

*Corresponding author

Funding: 2021 Hainan Provincial Natural Science Foundation Youth Fund (721QN0919); 2022 China Agricultural Journals Network Research Fund (CAJW2022-021, CAJW2022-018); 2022 Hainan Provincial Natural Science Foundation Youth Fund (722QN371).

Note: Figure translations are in progress. See original paper for figures.