Heritage Science

RESEARCH Open Access

ArchGPT: harnessing large language models for supporting renovation and conservation of traditional architectural heritage

Jiaxin Zhang[1], Rikui Xiang[2], Zheyuan Kuang[1], Bowen Wang[3] and Yunqin Li[1*]

Abstract

The renovation of traditional architecture contributes to the inheritance of cultural heritage and promotes the development of social civilization. However, executing renovation plans that simultaneously align with the demands of residents, heritage conservation personnel, and architectural experts poses a significant challenge. In this paper, we introduce an Artificial Intelligence (AI) agent, Architectural GPT (ArchGPT), designed for comprehensively and accurately understanding needs and tackling architectural renovation tasks, accelerating and assisting the renovation process. To address users' requirements, ArchGPT utilizes the reasoning capabilities of large language models (LLMs) for task planning. Operating under the use of tools, task-specific models, and professional architectural guidelines, it resolves issues within the architectural domain through sensible planning, combination, and invocation. Ultimately, ArchGPT achieves satisfactory results in terms of response and overall satisfaction rates for customized tasks related to the conservation and restoration of traditional architecture.

Keywords Architectural heritage, Cultural heritage, Artificial Intelligence agent, Large language models

Introduction

Renovating and preserving traditional architecture transcends mere conservation of cultural and historical heritage; it embodies a profound reverence for urban memory and identity [1]. Such endeavors not only bolster cultural heritage and historical education, enhancing social value, but also substantially foster tourism development. These activities enrich the quality of life for local populations and strengthen cultural identity. Concurrently, the integration of advanced technologies has transformed the conservation and repair of traditional structures. For instance, image recognition technologies are now deployed to detect structural damage [2], while deep learning models facilitate the automatic reconstruction of historical building images [3, 4]. These technological advances have significantly heightened the efficiency and precision of conservational efforts, heralding a new era in the preservation and restoration of traditional architecture.

Following the acknowledgment of traditional architectural renovation and preservation's role in conserving historical and cultural heritage [5, 6], and the integration of modern techniques to enhance these efforts [7], it's imperative to consider the communication among residents, heritage conservation personnel, and experts. The knowledge and recommendations of experts and scholars are crucial to the preservation work of heritage sites. However, real-time communication is often challenging, which can hinder the progress of restoration projects [8]. Consequently, the development of a method that facilitates communication among all stakeholders

*Correspondence:
Yunqin Li
liyunqin@ncu.edu.cn
[1] Architecture and Design College, Nanchang University, No. 999, Xuefu Avenue, Honggutan New District, Nanchang 330031, Jiangxi, China
[2] Department in Art and Convergence, Daejin University, 1007, Hoguk-ro, Pocheon-si, Gyeonggi-do, Seoul 11159, South Korea
[3] Osaka University, Institute of Datability Science, 2-1, Yamadaoka, Suita 5650871, Osaka, Japan

and leverages existing technologies becomes essential. This method would not only streamline the flow of information and expert insights but also incorporate technological advancements in conservation, such as image recognition [9, 10] and other deep learning models [11–13], to support informed decision-making and project execution. It could bridge the gap between technical advancements and practical applications, ensuring that the preservation of traditional architectural heritage is a collaborative, efficient, and technology-enhanced endeavor.

Specifically, the introduction of human-computer interaction modes based on large language models (LLMs) [14, 15] has brought new perspectives to the conservation and renewal of traditional architectural styles. Through human-computer dialogues, LLMs can understand and analyze the needs and preferences of stakeholders, thereby providing more comprehensive and precise conservation and renewal plans.

Utilizing the inference capabilities of LLMs to call external models for designing Artificial Intelligence (AI) agents [16] represents an effective solution for addressing specialized domain tasks. The essence of an AI agent is that the LLM autonomously plans task steps using its intelligence, invoking task-specific models and external tools to fulfill user requests. Some more general AI agents, such as BabyAGI[1], AgentGPT[2], AutoGPT[3], and HuggingGPT [17] demonstrate strong task-solving capabilities with the help of tools (internet search, code executors) and task-specific models from the Machine Learning (ML) community. In the field of traditional architecture conservation and restoration, there are complex task requirements along with numerous methods and model algorithms designed to address these practical issues and needs. However, these efforts typically require guidance from multiple experts within the field, leading to significant expenditures in terms of time, labor, and technical costs. Therefore, we aim to design a system capable of automatically analyzing actual task requirements under the guidance of LLMs and providing solutions, as well as facilitating communication among all personnel involved in the heritage protection work.

In this paper, we propose ArchGPT, an AI agent aimed at accurately and comprehensively completing tasks in the field of traditional architecture protection and restoration, combining user requirements and professional architectural renovation guidelines. Recognizing the challenges posed by the specialized nature of architectural content, which often transcends the direct training data of general LLMs, we have embedded a robust retrieval system within ArchGPT. This system utilizes the logic of Retrieval Augmented Generation (RAG) to enhance the model's responses with accurate, context-specific information extracted from a comprehensive architectural knowledge base. It also can accelerate the process of renovation by promoting communication among participants. ArchGPT narrows down the range of tools and task-specific models used but ensures that it can address all problems within this specialized domain through reasonable combinations and calls. For the task planning aspect, we will define necessary task steps based on prior knowledge (e.g., when an image is inputted, the Image Caption model [18] must be called to supplement LLM's understanding of the image), to provide essential inputs for the final response (refer to "Task parsing" section). Beyond these necessary prior constraints, the AI agent still enjoys the capability to autonomously plan tasks. Ultimately, ArchGPT has achieved satisfactory results in terms of response speed and overall satisfaction rate on our custom tasks related to the conservation and restoration of traditional architecture in southern China.

Related works

LLMs as AI agent

LLMs In recent years, the emergence of LLMs has brought about revolutionary changes in the field of Natural Language Processing (NLP), with models such as ChatGPT [19], GPT-4 [20], PaLM [21], and LLaMa [22] leading the charge. LLMs, owing to their vast corpora and intensive training computations, have demonstrated impressive capabilities in zero-shot and few-shot tasks, as well as in more complex tasks like mathematical problem-solving and common-sense reasoning. For instance, the advent of ChatGPT has highlighted the potential of LLMs in understanding human intent, reasoning, and following instructions to generate the required responses for specific tasks. Meanwhile, the introduction of GPT-4 has unlocked tremendous potential for multimodal perception, which is crucial for real-world foundational capabilities. To extend the intelligence of LLMs to more modalities, contemporary research has diverged into utilizing LLMs as controllers to design AI agents that complete tasks through autonomous planning and action. Both approaches have significantly expanded the capability boundaries of LLMs.

AI Agent To capitalize on the human-like capabilities of LLMs for efficiently executing a variety of tasks, an increasing body of work aims to design AI agents based on LLMs. These efforts focus on unleashing the autonomy and creativity of LLMs to automate task completion with external tools, such as generating novel ideas,

[1] https://github.com/yoheinakajima/babyagi.

[2] https://github.com/reworkd/AgentGPT.

[3] https://github.com/Significant-Gravitas/Auto-GPT.

stories, or solutions, moving towards more general artificial intelligence.

LLMs like BabyAGI, AgentGPT, AutoGPT, and HuggingGPT [17] are seen as autonomous agents, offering solutions for task automation. These agents adopt a step-by-step reasoning process, iterating with the LLM to generate the next task. Furthermore, AutoGPT employs an additive reflection module for each task generation, to assess the suitability of the currently predicted task. In contrast, HuggingGPT utilizes a global planning strategy to obtain an entire task queue in one query. Regarding tool usage, AutoGPT primarily utilizes common tools (e.g., web search, code executors), whereas HuggingGPT leverages task-specific models from the ML community (e.g., Hugging Face[4]).

These applications demonstrate the vast potential of LLMs in building AI agents. With just a task and a set of available tools provided, they can autonomously devise plans and execute these plans to achieve the end goals. LLM-based agents have been applied in various real-world scenarios, such as daily requests, software development, and scientific research. In this paper, we aim to propose an AI agent that addresses problems in the field of architecture. This agent will fully leverage task-specific models from the ML community and external tools to effectively solve tasks specific to the architectural domain.

Integrating LLMs with Traditional Architectural Conservation Guidelines

To enhance the role of LLMs in the architectural sector, research has investigated the integration of LLMs with domain-specific knowledge. Integrating LLMs with Life Cycle Assessment (LCA) [23] not only accelerates and refines the assessment process but also deepens understanding of the environmental impacts of building materials, processes, and products. This allows construction stakeholders to make better-informed decisions regarding product selection. Additionally, LLMs fine-tuned with proprietary technical specifications [24] efficiently automate the processing and querying of construction engineering documents, facilitating access to extensive architectural knowledge through an intuitive Q &A format. In the realm of prefabrication and assembly, combining LLMs with Building Information Modeling (BIM) [25] has fostered workflows that inform smart manufacturing practices in Design for Manufacturing and Assembly (DfMA). BIMS-GPT [26] uses GPT technology for natural language queries to extract and present relevant BIM database information through natural language and 3D visualizations, promoting versatile virtual assistants in the industry. In architectural heritage conservation, integrating multimodal LLMs with 3D Gaussian Spraying Technology (3DGS) creates digital twins that effectively visualize, document, and query cultural heritage structures [27].

These applications underscore the LLMs' robust capability to process and interpret architectural data and its implementation specifics. By streamlining information integration and enhancing data processing, LLMs improve the sustainability and efficiency of architectural projects. Furthermore, these technological advances are expanding into architectural conservation, aiming to merge LLMs with traditional conservation guidelines to boost the efficiency of complex architectural information analysis and provide actionable advice on maintaining and restoring historic buildings.

Methods

Overview

ArchGPT is a collaborative system designed to address traditional architectural facade renovation and aesthetic enhancement and guidance for building repairs. In addition to Normal Dialogue capabilities, it offers specialized functions such as Building Repair Guidance and Repair Rendering Generation. It is comprised of LLMs and external tools from the ML community, including task-specific models or manually crafted algorithms. The workflow involves four stages: Task Parsing, Tool Utilization, Answer, and Feedback. As illustrated in Fig. 1, given a user request, our ArchGPT uses an LLM as the controller to autonomously call upon various external tools to complete the workflow, ultimately producing the desired outcome. Fig. 2 provides a detailed usage demo of ArchGPT. In "ArchGPT" section, we will delve into each component of the complete workflow in detail.

External tools

Retrieval LLMs may lack professional knowledge to answer the user's question. Thus, retrieving existing knowledge from traditional architectural renovation guidelines is necessary. In this part, we propose a method that combines BM25-based [28] information retrieval techniques with BERT-based [29] semantic embedding techniques to perform text retrieval in two layers: the title I_t and the content I_c. Our aim is to execute effective information retrieval within architectural guideline documents like "Traditional Architectural Facade Renovation and Aesthetic Enhancement Guidelines"[5]. As shown

[4] https://huggingface.co/.

[5] This document utilizes a hierarchical architecture, also known as a tree structure, where each leaf node's title contains corresponding content. All titles from a leaf node to the root node are denoted as $/_t$, and the specific content under a leaf node is denoted as $/_c$.

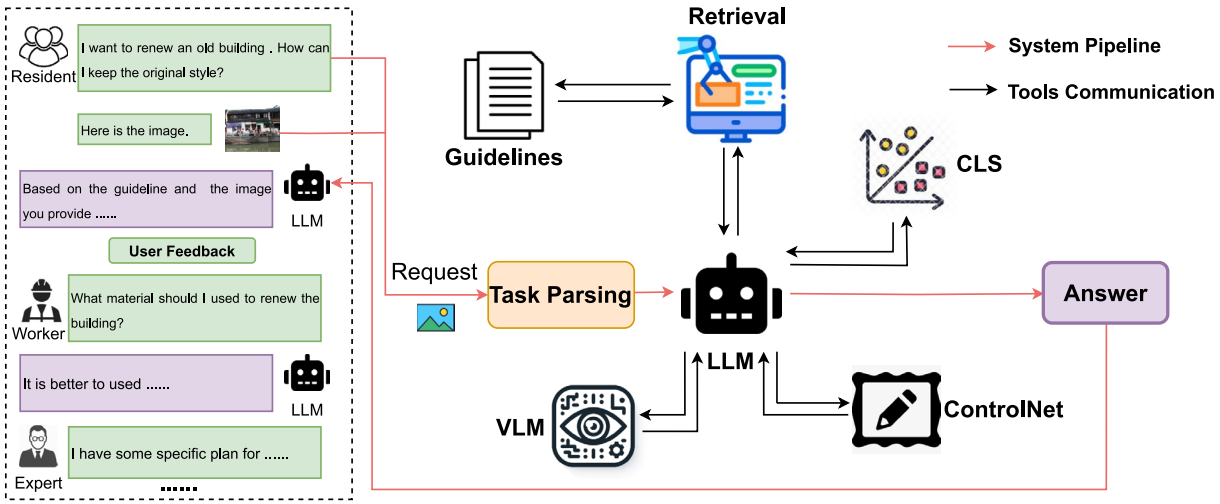


Fig. 1 The architecture of ArchGPT incorporates a LLM as its central controller to respond to user requests. Upon receiving a request, ArchGPT initially parses the request to ascertain the required task. Subsequently, it invokes external tools (such as VLM and ControlNet) to facilitate the completion of this task. Ultimately, a response is generated, and based on the user's feedback, ArchGPT engages in further rounds of conversation

in Fig. 3, we demonstrate the structure of the guideline for traditional architectural renovation in Fuzhou, China (only a part of the guideline is demonstrated). Based on user-input queries, the Retrieval algorithm obtains the most relevant item $I = (I_t + I_c)$ from the document. Note that prior to retrieval, we use an LLM to summarize key information from the user's input to generate the *query*.

BM25: First, we employ the Okapi BM25 retrieval function as the foundation of our retrieval system. BM25 is a classic information retrieval ranking function based on a probabilistic model. Its main objective is to rank I with the highest relevance at the forefront based on the frequency and location of *query* terms within documents. BM25 allows for the rapid and rough filtering of I likely relevant to the *query* within specific documents. Specifically, for a given *query*, we start by breaking down the *query* into tokens and calculating each I's score through the BM25 ranking function.

BERT Embeddings: Although BM25 performs well in many scenarios, it cannot capture the deep semantic relationships between words. Therefore, we decided to utilize BERT, a pre-trained deep learning model, to create richer word embeddings that capture semantic information. We encode each *query* and I with BERT to obtain their vector representations. These embeddings provide us with a method to measure the semantic relevance between *query* and I. Specifically, we evaluate their semantic similarity by calculating the cosine similarity between the normalized embeddings of the *query* and the I.

We combine the scores from both methods, weighted, to arrive at the final retrieval score, as follows:

$$Score(I, Q, \alpha) = \alpha \cdot BM25(I, Q) + (1 - \alpha) \cdot S(E(Q), E(I)), \quad (1)$$

where I is the combination of all titles from a leaf node to the root node I_t and the specific content under a leaf node I_c, Q is the *query*, α is weight hyper-parameter used to balance the contribution of $BM25$ and *Similarity*, $BM25$ represents the $BM25$ scoring function, S represents the cosine similarity score function for normalized embeddings, and E is $BERT_{base}$ model.

The I with the highest score will be returned as the final result of the retrieval process. The final search formula is defined as follows:

$$Retrieve = MAX(Score(I, Q, \alpha), 1) \quad (2)$$

where MAX is the selection operation for top-1 item and α is 0.4. The retrieved output will then concatenated with user's origin input for the final answer from LLMs.

Classification Model (CLS) Since different types of buildings correspond to different architectural renovation guidelines in the 'Building Repair Guide', we need an architectural image classification model to select the appropriate architectural renovation guidelines. When necessary, these guidelines supplement ArchGPT's knowledge in response to user requests. ArchGPT utilizes a fine-tuned Vision Transformer (ViT [30]) model based on CLIP [31] to classify user-uploaded architectural images. It is capable of recognizing five types of architectural categories, including preserved, renovated,

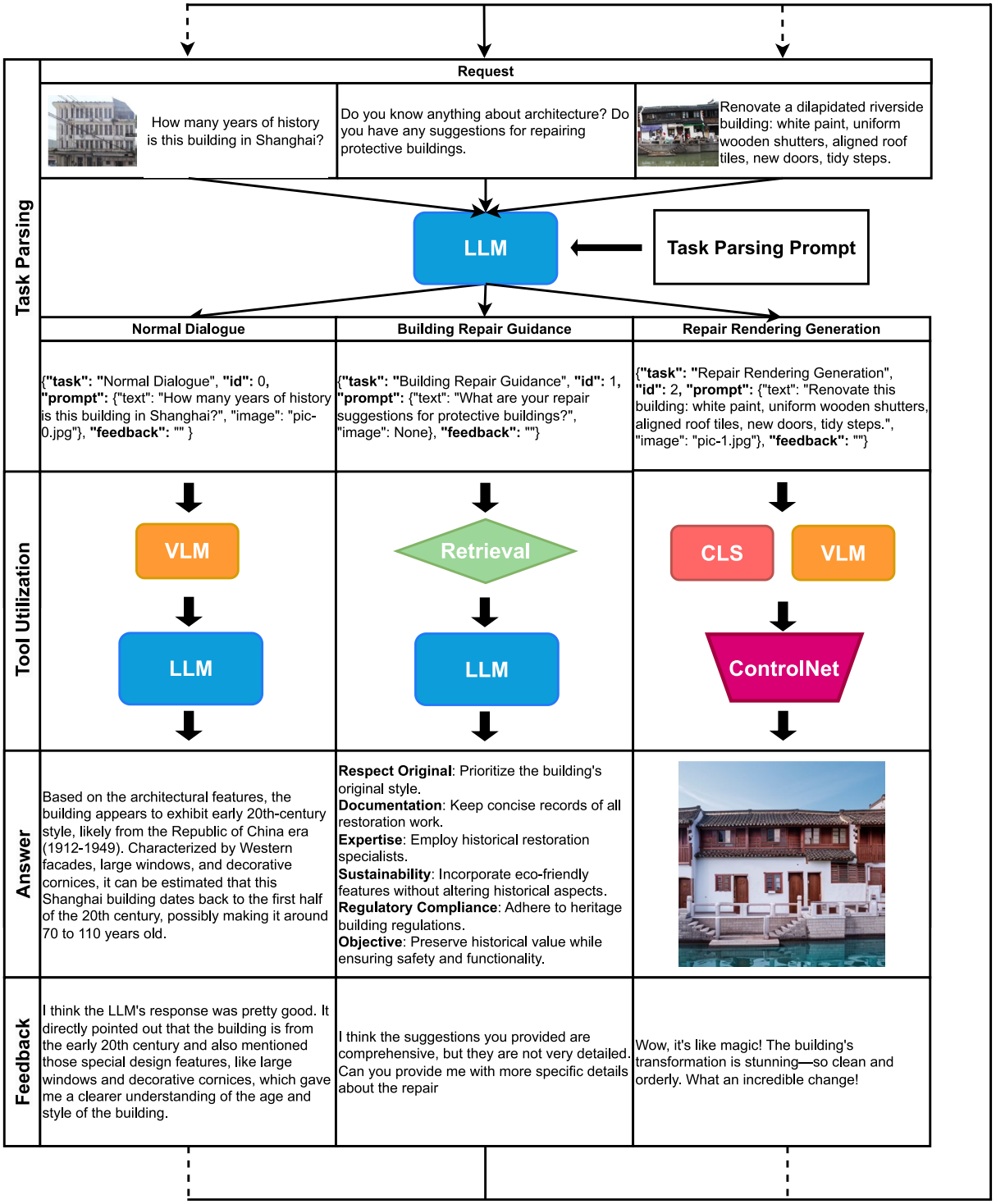


Fig. 2 The workflow of ArchGPT for normal dialogue, building repair guidance, and repair rendering generation pipelines, respectively

improvement, retained, and transformed buildings. Each category has its specific renovation guidelines, as shown in Fig. 4.

Visual Language Model (VLM) As long as the user inputs an image, ArchGPT utilizes BLIP [32] to generate captions for images to provide the LLM with detailed image information in text form as much as possible,

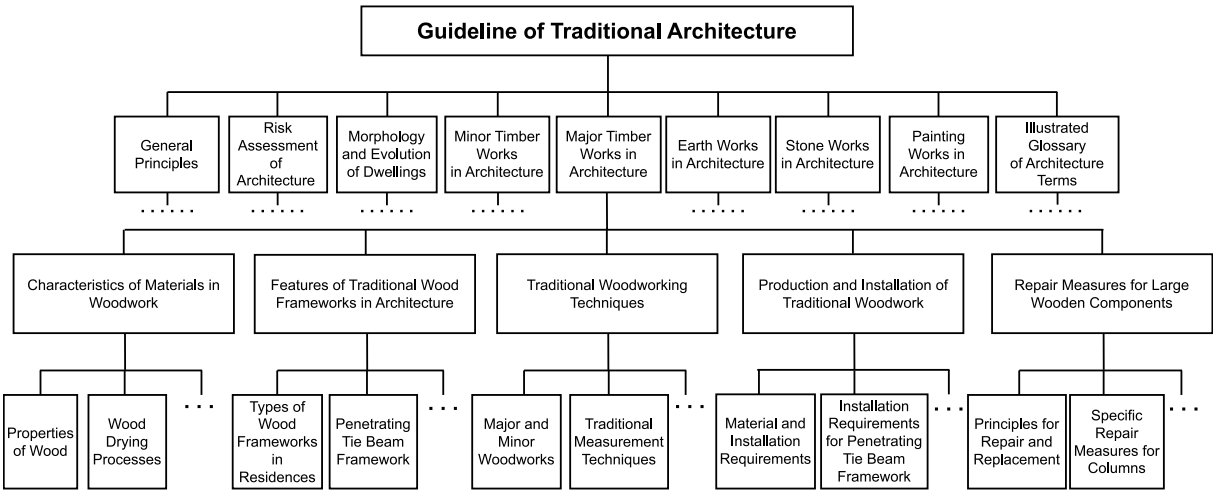


Fig. 3 Conceptual representation of the structure/content of the guideline for traditional architectural renovation in Southern China

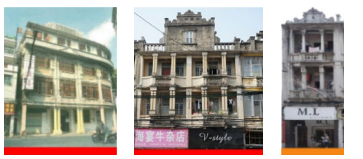
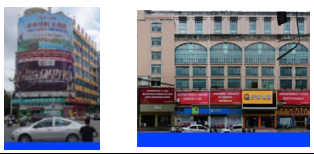
Classification	Building Protection Measures	Image Example
Preserved Buildings	1. The exterior walls are basically unchanged, with a focus on cleaning and repairing; 2. Rectify illegal buildings (structures); 3. Unauthorized exterior facades should be restored to their original state as much as possible; 4. Redesign damaged wooden doors and windows according to their original design; 5. Specially designed air conditioners, storefronts, signs, and wall lights.	
Renovated Buildings	1. Rectify illegal buildings (structures); 2. Unauthorized exterior facades should be restored to their original state possibly; 3. For buildings with exterior walls made of tiles and lacking details, it is appropriate to add details that match the historical style and repaint them; 4. Redesign damaged wooden doors and windows according to their original design; 5. Specially designed air conditioners, storefronts, signs, and wall lights.	
Improved Buildings	1. Rectify illegal buildings (structures); 2. Unauthorized exterior facades should be restored to their original state possibly; 3. Repair the damaged exterior walls, railing construction, mountain flowers, parapets, etc. of the building according to their original appearance, and repair them according to their original colors; 4. Redesign damaged wooden doors and windows according to their original design; 5. Specially designed air conditioners, storefronts, signs, and wall lights.	
Retained Buildings	1. For modern buildings that maintain a relatively intact status quo, rectify illegal buildings (structures) and integrate them into the overall environment of traditional historical style; 2. Clean the exterior walls of the building and focus on areas that can be directly felt by people.	
Transformed Buildings	1. Rectify illegal buildings (structures); 2. For modern buildings that are relatively dilapidated, it is planned to make overall modifications and renovations to the building. Integrate it into the overall environment of traditional historical style.	

Fig. 4 Protection measures and samples for five types of buildings

ensuring that the LLM can "read" the image, understand user intent, and make accurate task planning.

ControlNet To edit old architectural images based on user suggestions, ArchGPT employs an image editing model to implement the functionality of rendering old building restorations based on prompts. ArchGPT inputs the user's renovation suggestions along with the original architectural image into the image editing model to generate effect images of the renovated building. In this project, we utilize the latest image editing model, ControlNet [33], a powerful image editing mode capable of generating effect images based on renovation suggestions and the original image, helping users reference and improve the final renovation architectural drawings.

ArchGPT

Task parsing

ArchGPT's primary tasks include Normal Dialogue, Building Repair Guidance, and Repair Rendering

Generation, which means the LLM needs to handle prompts of two types: language and image. To help the LLMs better parse user prompts, in addition to setting the Task Parsing Prompt[6] for ArchGPT, ArchGPT also automatically use VLM to obtain image descriptions to supplement the text prompt when there is image as input, and instruct the LLM to adhere to specific standards (e.g., JSON format) for parsing prompts. Therefore, we designed a standardized task template that requires the LLM to parse tasks through fields. As shown in the task parsing illustration in Fig. 2, similar to HuggingGPT [17], the task parsing template includes four fields ("task", "id", "prompt", "feedback") to represent the task name, unique identifier, user prompt, and user feedback. By parsing the task parsing dictionary, ArchGPT can automatically use the LLMs to analyze user requests and parse tasks accordingly. We also provide {**Demonstrations**} for LLM's reference, and maintain a $\{$**ChatLogs**$\}$ using a task parsing dictionary list, where LLM can track the resources mentioned by users and incorporate them into task planning.

Tool utilization and output

When a user input is solely parsed as Normal Dialogue, the LLM directly responds to the user's prompt (When there is an image input, a VLM is called to obtain image captions to supplement the text prompt.). When user input is parsed as Building Repair Guidance, the retrieval module is used to search the architectural documents for the most relevant items to supplement the user's text prompt (When there is image input, call the CLS model to supplement the text prompt with the corresponding architectural renovation guidelines), and then necessary expertise is provided to the LLM to accurately reply. When user input is parsed for Repair Rendering Generation, the text prompts supplemented by CLS and VLM model, along with the image, are input into ControlNet to obtain the edited image.

The aforementioned workflow of ArchGPT can be formalized as:

$$Answer = LLMs(Parse(Request), Tool) \tag{3}$$

where *Request* and *Answer* respectively represent the user input and the response result of ArchGPT, which can be in the form of text or image. *Parse* represents the LLM's formatting analysis of user input to obtain standardized task instructions. *Tool* represents the tools that need to be called for the standardized task instructions, and *LLMs* represents the execution of the entire workflow by calling tools and prompt according to the parsed instructions.

Feedback

Feedback is the evaluative feedback from a user after receiving the Answer. All feedbacks are stored under the feedback field in the task parsing dictionary, which is then saved in the $\{$**ChatLogs**$\}$. When a user provides positive affirmative feedback, it signifies the end of the ArchGPT response. If the user provides unsatisfied or negative feedback requiring modifications, ArchGPT will execute the complete workflow from the beginning again based on the content of the feedback, until the user is satisfied.

Experiments

Experiments settings

In our experiments, the LLM controllers we used include Alpaca-7b [34], Vicuna-7b [35], and GPT-3.5[7]. To make the outputs of the LLM more stable, we followed the practice of HuggingGPT to set the decoding temperature to 0. For the VLM model with Image Captioning capability, we loaded the BLIP-base model [32] from Hugging Face to generate captions for images.

For CLS model, we fine-tuned the ViT-B/16 model from CLIP on a custom-made architectural classification dataset[8] of 1000 images, achieving a classification accuracy of 90.6% on the test set. In Fig. 5, we also provide a confusion matrix for the 5-category classification, showing excellent performance across most categories. However, the model's precision on the Improved Buildings category is slightly lower (0.852), possibly due to significant overlap in features of improved buildings with other categories (such as renovated or preserved buildings), leading to some misclassifications. Overall, the CLS task-specific model is able to assist ArchGPT well in identifying the input architectural type, supplementing the LLMs with knowledge about architectural renovation.

[6] The artificial intelligence assistant parses user input and generates a task dictionary in the following format: {"task": "task", "id": "task_id", "prompt"y: {"text": "text", "image": "URL"}, "feedback": "user_feedback"}. The "task" field represents the current task type, which belongs to one of [Normal Dialogue, Building Repair Guidance, Repair Rendering Generation]. The "id" field is the unique identifier of the task. The "prompt" field represents the text and image prompt organized for user input. The "feedback" field represents user feedback on ArchGPT response results. Please note that summarize the user's text input and if the user input cannot be parsed, an empty JSON response should be provided. Here are a few examples for your reference: {{Demonstration}}. To assist with task planning, chat history is provided in the form of {{Chat Logs}}, where you can track the resources mentioned by users and incorporate them into the task parsing phase.

[7] https://chat.openai.com.

[8] The dataset consists of 200 images per category, totaling 1000 images, aimed at covering a wide range of architectural styles and degrees of transformation, with a training and test set split of 8:2 with evenly distributed categories.

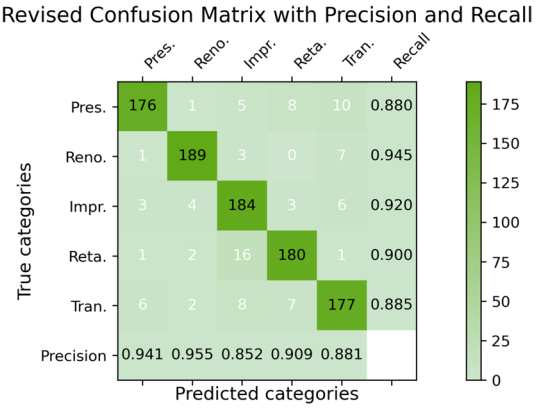


Fig. 5 Confusion matrix for the architectural classification task, where "Pres." stands for Preserved Buildings, "Reno." represents Renovated Buildings, "Impr." denotes Improved Buildings, "Reta." signifies Retained Buildings, and "Tran." corresponds to Transformed Buildings. The matrix showcases model performance in terms of Precision and Recall for each category

Table 1 BM25 and BERT represent methods based on BM25 scores and BERT embedding semantic similarity scores, respectively

Metric	BM25	BERT	BM25+BERT
Precision	0.858	0.942	0.963
Recall	0.892	0.945	0.976
F1	0.880	0.943	0.960

BM25+BERT represents the method of weighting two scores

Based on architectural renovation guidelines, we utilized GPT-4 to generate 2000 pairs of queries and I (I_t and I_c from the guideline introduced in) to create a retrieval dataset. Specifically, we provided prompts to GPT-4 such as, "I'm giving you {document:I}, please generate a series of questions users might ask, following the format query:xx," to produce 2000 data pairs. I serves as the ground truth for the query, used to evaluate the retrieval accuracy of our proposed retrieval algorithm.

As illustrated in Table 1, we compared three information retrieval methods: BM25, BERT, and BM25+BERT. The results show that BM25+BERT outperforms both BM25 and BERT across all key metrics. This indicates that the BM25+BERT method, which integrates the human prior knowledge of BM25 and the semantic understanding ability of BERT, can achieve the highest accuracy in information retrieval.

Qualitative results

Fig. 2 shows three demos for the pipeline of ArchGPT. In the first demo, the user's request involves understanding the content of an image and is unrelated to the architectural category. Therefore, ArchGPT plans only to use the VLM to generate an image caption to supplement information on the appearance of the building. Based on the sufficient architectural information, the LLM then makes predictions about the building's history and provides reasons. Finally, ArchGPT receives positive feedback from the user, completing the Normal Dialogue workflow.

The process of the second demo is similar to the first, except that this time ArchGPT parses the task as Building Repair Guidance. Therefore, it needs to retrieve the corresponding repair guidelines from the architectural guideline document to enhance the LLM's knowledge, eventually providing a repair suggestion that meets the architectural guidelines and fulfills the user's request. However, the response provided by the LLM is too generic, and after integrating user feedback, ArchGPT will execute the entire Building Repair Guidance workflow again.

The third demo is more complex, requiring ArchGPT to call upon and coordinate the use of more external tools to complete the image editing task. Specifically, ArchGPT first parses the user's request as a Repair Rendering Generation task, then uses the CLS and VLM to parse image information to supplement the image editing requirements, and finally inputs the supplemented editing requirements and the original image into the ControlNet model to obtain a reasonable edited image.

The above examples prove that ArchGPT indeed has the ability to leverage the intelligence of the LLMs to accurately parse user intentions and methodically call upon tools to solve real problems.

In Fig. 6, we present the dialogue flow of ArchGPT in real-use scenarios. The first flow depicts a case where ArchGPT fails to process a user's question correctly. ArchGPT mistakenly parses the user's intent as Repair Rendering Generation, subsequently calling the CLS and VLM models to supplement the "prompt"-"text", and finally using ControlNet to obtain a photo of the repaired building. However, the user is not interested in updates to the house's exterior but is more interested in receiving some suggestions for renovations inside the house. The second process demonstrates a correct and user satisfied Repair Rendering Generation process.

Quantitative evaluation

In ArchGPT, task parsing plays a crucial role throughout the workflow as it determines which tasks will be executed in the subsequent pipeline. Therefore, we consider the quality of task parsing as a measure of the LLM's capability as a controller within ArchGPT. For this purpose, we conduct a quantitative evaluation based on the

Fig. 6 The dialogue process of ArchGPT in practical usage scenarios

ability to perform task parsing and the strength of this ability, to assess ArchGPT's task parsing capability.

Metric To quantify whether ArchGPT has the ability to complete task parsing, we track how many attempts it takes for ArchGPT to correctly parse the "task" field into the correct task type (Normal Dialogue, Building Repair Guidance, Repair Rendering Generation) after receiving a user request, without considering other fields. Additionally, to quantify how strong ArchGPT's ability is to complete task parsing, as long as ArchGPT correctly parses the "task" field within 4 times (note that each new attempt will include the feedback from the user), we use GPT-4 as a critic to evaluate whether the task parsing dictionary is reasonable (following HuggingGPT).

Table 2 The number of times a user request is parsed to the correct task type

Normal dialogue				Building repair guidance				Repair rendering generation			
	A.	V.	G.		A.	V.	G.		A.	V.	G.
1	77	82	94	1	61	65	85	1	91	89	96
2	16	12	4	2	21	21	9	2	7	11	4
3	5	4	2	3	13	8	5	3	1	0	0
3+	2	2	0	3+	5	6	1	3+	1	0	0

A., V. and G. respectively represent Alpaca, Vicuna, and GPT-3.5

When using GPT-4 to judge whether the task parsing dictionaries generated by ArchGPT are reasonable, the prompt given to GPT-4 is: "We will next provide examples of high-quality and low-quality task parsing dictionaries that interpret user requests. There are five examples of each type, presented in the format "High-quality examples: High-quality examples, Low-quality examples: Low-quality examples". Please learn from these to develop your evaluation skills. Afterward, I will give you some new examples, and you only need to answer "High-quality" or "Low-quality".

Dataset We created 100 requests for each of the three types of tasks, totaling 300 requests, which were created by 6 architecture students. These submissions were collected to create an evaluation dataset, with user annotated task type labels, used to evaluate whether ArchGPT can complete task parsing. In the evaluation of how strong ArchGPT's ability is to complete task parsing, GPT-4 is used to judge whether the task parsing dictionaries generated by ArchGPT are of "High-quality" or "Low-quality". Note that the task parsing dictionary to be evaluated is a dictionary that LLM has successfully and correctly parsed no more than three times.

Performance Our experimental evaluation covered various LLMs, including Alpaca-7b, Vicuna-7b, and the GPT-3.5 model. In Tables 2 and 3, Alpaca and Vicuna refer to Alpaca-7b and Vicuna-7b, respectively.

In Table 2, GPT-3.5 demonstrated superior performance across all three task types, especially in Normal Dialogue and Repair Rendering Generation tasks, where it significantly outperformed Alpaca-7b and Vicuna-7b in the number of correct parses. This result indicates that GPT-3.5 has higher accuracy and efficiency in understanding user requests and accurately classifying them into the corresponding tasks, reflecting its strong planning capabilities in complex scenarios. Table 3 shows that the high-quality task parsing dictionaries obtained from GPT-3.5 have a high proportion, particularly in repair rendering generation tasks, where the proportion of high-quality dictionaries reached 96/100. This further confirms that GPT-3.5 not only has a high accuracy in generating specific task parsing dictionaries but also ensures quality. In contrast, although Vicuna-7b performed better in repair rendering generation tasks than Alpaca-7b, their performance in other task types was similar and both were lower than GPT-3.5. These results not only prove the capability of GPT-3.5 as a controller in task parsing and execution but also suggest that improving the technology for complex task planning of LLMs is crucial for future research and development.

Human evaluation

In addition to objective evaluations, we also follow HuggingGPT to invite human experts to perform subjective assessments in our experiments. The significance of incorporating human evaluations lies in their ability to provide nuanced insights that go beyond the quantitative metrics typically used in objective evaluations. While objective metrics are essential for measuring performance, they often fail to capture the qualitative aspects of how well an AI system meets user needs in real-world scenarios. By involving human experts, particularly in a specialized field such as architectural heritage, we ensure that the evaluations consider practical, contextual, and experiential factors that are critical for the successful application of AI technologies.

In our experiments, from our custom set of 300 requests, we extracted 30 requests from each of the three types of tasks, providing a total of 90 requests to different LLMs. Three architectural heritage experts from Nanchang University evaluated the performance of ArchGPT in three stages (Task Parsing, Tool Utilization, and Answer). The tasks involved and the metrics

Table 3 The proportion of high-quality dictionaries in dictionaries which was parsed correctly no more than three times attempt

Task type	Alpaca	Vicuna	GPT-3.5
Normal dialogue	76/98	77/98	87/100
Building repair guidance	81/95	78/94	91/99
Repair rendering generation	86/99	91/100	96/100

Table 4 Human evaluation on different LLMs

LLMs	Task parsing		Tool utilization		Answer
	Correctness	Rationality	Completion	Numbers	Success
Alpaca	70	54	49	2.07	49
Vicuna	71	59	55	2.16	56
GPT-3.5	83	75	73	1.73	72

Experiments are implemented for 30 requests randomly selected from each task

assessed are described below. (the results were decided by a vote of these 3 experts, and at least two agreeing counts as a pass).

- *Task parsing*: We collect the number of correctly parsing task types for the first time and the corresponding number of reasonable dictionaries. The Correctness here refers to the number that LLM correctly classifies the task type for the first time. The Rationality here refers to the number of generating a reasonable task parsing dictionary, which is judged by humans.
- *Tool utilization*: During the Tool Utilization phase, we use correctly classified and reasonable task parsing dictionaries to guide LLM in calling different tools to complete the entire workflow. However, during the execution process of LLM, even if the task parsing dictionary is reasonable for the planning of the entire workflow, LLM may encounter problems in parameter transfer or tool invocation due to its limited instruction-following ability, resulting in the inability to obtain effective output from the tools or task-specific models, and thus the workflow cannot be completed. So we define the Completion as the number of correctly calling the tools and collecting intermediate outputs to complete the entire workflow. At the same time, we also counted the Numbers of tools and task-specific models used in complete workflows to evaluate the task completion efficiency of LLM.
- *Answer*: We use Success to represent how many of the 90 requests received responses (answers generated by ArchGPT) that satisfied user requirements.

From Table 4, we see that all LLMs can fulfill user requests in one go with a success rate of over fifty percent. If feedback is introduced for multi-turn dialogue, success rates are expected to further increase. We also observe that Alpaca and Vicuna exhibit similar levels of Correctness and Numbers, but Vicuna significantly leads in Rationality and Completion. Therefore, we believe that the reasonableness of LLMs in parsing dictionaries and the strength of their command-following capabilities are crucial for successful responses. Comparing these three LLMs, GPT-3.5 is notably superior to open-source LLMs such as Alpaca-7b and Vicuna-7b from the Task Parsing to Answer stage, with lesser dependency on tools. This aligns with previous objective assessments and underscores the necessity of a strong LLM as a controller within an AI agent. In summary, ArchGPT fully explores the potential of LLMs, demonstrating the feasibility of AI Agents in solving architectural tasks, with significant efficiency in Task Parsing and Tool Utilization.

Table 5 Expert evaluation: proportion of successful responses across different tasks under various LLMs

LLM	Normal dialogue	Building repair guidance	Repair rendering generation	All tasks
Alpaca	14/30	19/30	16/30	49/90
Vicuna	18/30	22/30	16/30	56/90
GPT-3.5	24/30	27/30	21/30	72/90

Table 6 Ablation study on the retrieval module for the building repair guidance task

	Alpaca	Vicuna	GPT-3.5
w/ retrieval	21	24	27
w/o retrieval	12	11	15

We organize the evaluation data from human experts across different applications and LLMs to showcase the diverse applications and effectiveness of ArchGPT.

We organize the evaluation data from human experts across different applications and LLMs in Table 5 to showcase the diverse applications and effectiveness of ArchGPT. Comparing the performance of Normal Dialogue and Building Repair Guidance tasks, despite Building Repair Guidance requiring more specialized architectural knowledge, ArchGPT performs better in this task. We attribute this advantage to the retrieval module proposed in ArchGPT. To substantiate this, we conduct ablation experiments on the retrieval module within the Building Repair Guidance tasks. As indicated in Table 6, without the retrieval module, the performance of all models decreases significantly (from 21 to 12, 24 to 11, and 27 to 15). This underscores the importance of incorporating a retrieval-augmented generation mechanism in ArchGPT to enhance performance effectively.

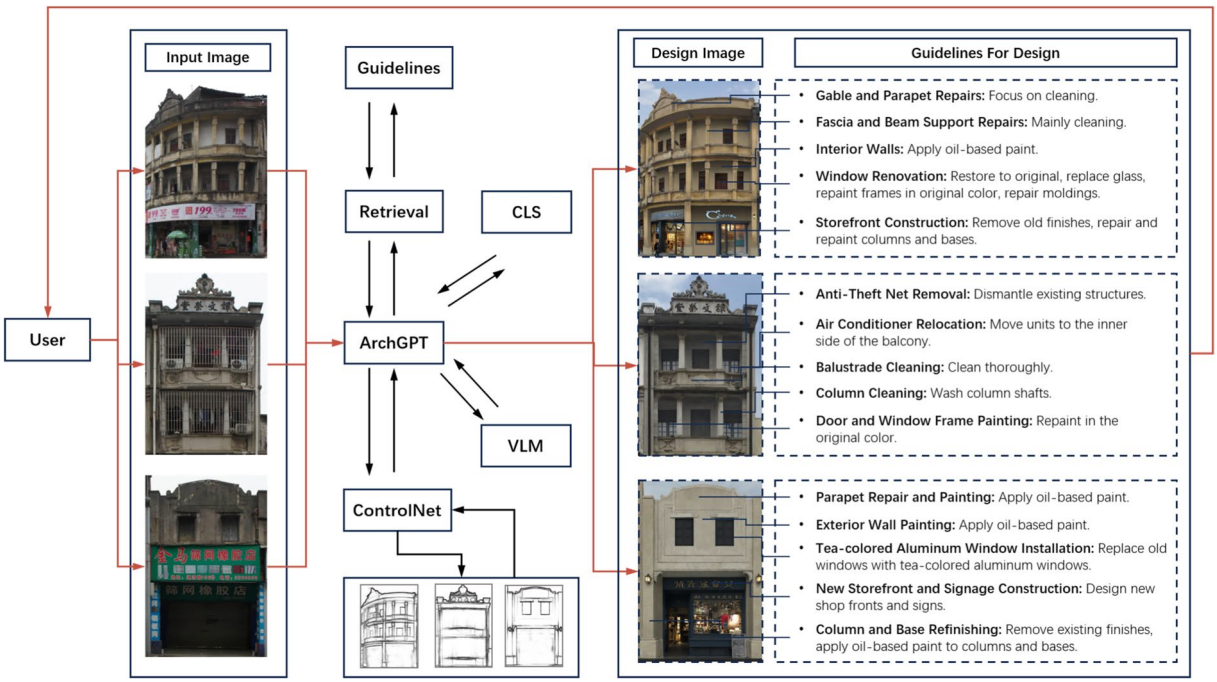


Fig. 7 ArchGPT application in traditional architectural heritage renovation and conservation

Disscusion

In this paper, we delve deeply into the extensive applications of ArchGPT in the field of architecture and its profound impact on traditional architectural renovation and preservation practice. Through the analysis of various real-world usage scenarios, we conclude that ArchGPT has brought unprecedented innovation and efficiency to architectural design and planning, and that the powerful LLM as an ArchGPT controller is necessary to accurately complete user requests. Additionally, it plays a vital role in traditional architectural preservation, public education, and the promotion of sustainable building practices.

Another important point is that communication among residents, heritage conservation personnel, and experts emerges as a critical factor. By improving communication, stakeholders can collaborate more effectively, ensuring that the actions taken are both timely and in line with the best preservation practices. As shown in Fig.1, ArchGPT facilitates a more dynamic exchange of knowledge. In the future, we plan to create an online platform to facilitate discussions and collect more data to continually optimize ArchGPT's problem-solving capabilities.

In addition, we explore the potential applications of ArchGPT in the renewal and preservation design of urban traditional architectural heritage. Figure 7 illustrates how ArchGPT extracts information from existing architectural photographs and generates restoration and renewal plans for different building facades. Specifically, ArchGPT precisely delineates the types of restoration tasks through its VLM and CLS modules. Based on this, it dynamically extracts and adjusts restoration guidelines from its knowledge base, producing customized restoration strategies. With the aid of ControlNet technology, ArchGPT translates restoration plans into intuitive visualizations, providing concrete and feasible visual references for all stakeholders. The application results demonstrate that ArchGPT conducts thorough analyses of the buildings' historical context, structural characteristics, and current damages, ensuring that the restoration plans adhere to the following principles: respect and preserve the building's historical value, accommodate modern functional requirements, and employ appropriate restoration techniques for long-term stability. Specific actions, as shown in Fig. 7, include removing non-original structures such as anti-theft nets, restoring the original color of door and window frames, and relocating air conditioning units to more concealed positions to avoid disrupting the building's appearance. Notably, the window renovation includes tea-colored aluminum frames that harmonize with the original style, exemplifying the integration of traditional materials with modern technology. Furthermore, facade cleaning, repair, and painting are aimed at restoring the building's original visual effect while providing an additional protective layer against future environmental degradation.

Overall, this study showcases the practical application of ArchGPT in the renewal and conservation projects of traditional architectural heritage, demonstrating both its deep understanding of historical architectural details and the feasibility of implementing conservation-oriented restorations in modern urban environments. Through such refined and personalized restoration plans, we can align the needs of all stakeholders, further advancing the protection and revitalization of architectural heritage.

One limitation of ArchGPT is the scope of its external tools. Currently, it incorporates four tools designed to enhance task-solving capabilities. We plan to expand this toolkit by introducing additional resources, such as 3D rendering and internet searching, to enhance our system's functionality. Additionally, we intend to design a broader range of task scenarios beyond the existing three, aiming to achieve wider real-world applicability.

Moreover, recent advancements in multimodal models, exemplified by GPT-4, indicate a promising direction where the dependence on external modules might be significantly reduced, thereby streamlining the architecture design and implementation processes. While these integrated models offer the allure of simplification and potentially lower operational complexities, they come with their own set of challenges, primarily related to higher computational demands and associated costs. Future research will explore the feasibility of deploying high-performance open-source multimodal models that can provide similar capabilities. This approach not only promises a reduction in the logistical and technical overhead of managing multiple external tools but also aligns with the ongoing trends of increasing model efficiency and effectiveness. However, it is crucial to evaluate the trade-offs involved, particularly in terms of cost-effectiveness and accessibility for users, ensuring widely adopted within the field of architectural conservation and restoration.

In summary, as a revolutionary technological tool, ArchGPT's application in the preservation and adaptive reuse of traditional architecture has broken down the barriers of professional knowledge, facilitating interdisciplinary communication and collaboration through human-computer interaction. This not only enhances the efficiency of project implementation but also fosters consensus among stakeholders from diverse backgrounds, contributing to the development of more comprehensive and diverse conservation and renewal strategies. Most importantly, ArchGPT helps to renovate urban renovation architectural works that reflect historical traditions, while meeting the needs of contemporary society and providing new possibilities for innovative protection of traditional architectural heritage.

Conclusion

In this paper, we demonstrated the capabilities of ArchGPT for architectural conservation and restoration, effectively dismantling the barriers of specialized knowledge. By executing tasks with precision and responding promptly, ArchGPT not only streamlines the process of preserving and modernizing traditional architecture but also bolsters public appreciation for the value of historic buildings through its human-computer interaction model. It can also utilize extra tools to enhance its task completion ability. Moreover, ArchGPT's contribution to sustainable architectural practices and public education underscores a deep respect for urban memory and identity. By providing architectural design solutions that cater to modern requirements while honoring historical traditions, ArchGPT opens up new avenues for innovative preservation, illustrating its potential to transform urban landscapes sustainably.

Acknowledgements
Not applicable.

Author contributions
Conceptualization, J.Z. B.W. Y.L. and Z.K.; methodology, Y.L. R.X. and J.Z.; software, B.W., R.X. and Y.L.; validation, Y.L. B.W. and J.Z.; formal analysis, B.W., R.X. and Y.L.; investigation, Z.K. and B.W.; resources, R.X. and B.W.; data curation, Z.K. and Y.L.; writing—original draft preparation, J.Z. and R.X; writing—review and editing, J.Z. and B.W.; visualization, B.W.; supervision, J.Z.; project administration, J.Z.; funding acquisition, Z.K. J.Z. and Y.L. All authors have read and agreed to the published version of the manuscript.

Funding
This work is partly supported by Key Research Base of Humanities and Social Sciences in Jiangxi Universities 2023 Project JD23003. This work is also supported by JSPS KAKENHI 24K20795.

Data availability
All data is available by reasonable request. No datasets were generated or analysed during the current study.

Code availability
Our implementation code is available by contacting corresponding author.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare no competing interests.

Received: 15 April 2024 Accepted: 19 June 2024
Published online: 28 June 2024

References

1. Neirotti P, De Marco A, Cagliano AC, Mangano G, Scorrano F. Current trends in smart city initiatives: some stylised facts. Cities. 2014;38:25–36.
2. Lee Y, Kim H, Min S, Yoon H. Structural damage detection using deep learning and fe model updating techniques. Sci Rep. 2023;13(1):18694.
3. Sun C, Zhou Y, Han Y. Automatic generation of architecture facade for historical urban renovation using generative adversarial network. Build Environ. 2022;212: 108781.
4. Bacharidis K, Sarri F, Ragia L. 3d building façade reconstruction using deep learning. ISPRS Int J Geo-Inf. 2020;9(5):322.
5. Lenzerini F. Intangible cultural heritage: the living culture of peoples. Eur J Int Law. 2011;22(1):101–20.
6. Vanolo A. Smartmentality: the smart city as disciplinary strategy. Urban stud. 2014;51(5):883–98.
7. Li Y, Du Y, Yang M, Liang J, Bai H, Li R, Law A. A review of the tools and techniques used in the digital preservation of architectural heritage within disaster cycles. Herit Sci. 2023;11(1):199.
8. Bonazza A, Sardella A. Climate change and cultural heritage: methods and approaches for damage and risk assessment addressed to a practical application. Heritage. 2023;6(4):3578–89.
9. Chen L, Li S, Bai Q, Yang J, Jiang S, Miao Y. Review of image classification algorithms based on convolutional neural networks. Remote Sens. 2021;13(22):4712.
10. Wang B, Li L, Nakashima Y, Nagahara H. Learning bottleneck concepts in image classification. In: IEEE Conference on Computer Vision and Pattern Recognition 2023.
11. Dong S, Wang P, Abbas K. A survey on deep learning and its applications. Comput Sci Rev. 2021;40: 100379.
12. Wang B, Zhang J, Zhang R, Li Y, Li L, Nakashima Y. Improving facade parsing with vision transformers and line integration. Adv Eng Inf. 2024;60: 102463.
13. Wang B, Li L, Verma M, Nakashima Y, Kawasaki R, Nagahara H. Mtunet: few-shot image classification with visual explanations. In IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshop 2021.
14. Brown T, Mann B, Ryder N, Subbiah M, Kaplan JD, Dhariwal P, Neelakantan A, Shyam P, Sastry G, Askell A, et al. Language models are few-shot learners. Adv Neural Inf Process Syst. 2020;33:1877–901.
15. Bommasani R, Hudson DA, Adeli E, Altman R, Arora S, Arx S, Bernstein MS, Bohg J, Bosselut A, Brunskill E, et al. On the opportunities and risks of foundation models. arXiv preprint arXiv:2108.07258. 2021.
16. Wang L, Ma C, Feng X, Zhang Z, Yang H, Zhang J, Chen Z, Tang J, Chen X, Lin Y, et al. A survey on large language model based autonomous agents. arXiv preprint arXiv:2308.11432. 2023.
17. Shen Y, Song K, Tan X, Li D, Lu W, Zhuang Y. Hugginggpt: solving ai tasks with chatgpt and its friends in hugging face. Adv Neural Inf Process Syst. 2024;36.
18. Hossain MZ, Sohel F, Shiratuddin MF, Laga H. A comprehensive survey of deep learning for image captioning. ACM Comput Surv (CsUR). 2019;51(6):1–36.
19. Bahrini A, Khamoshifar M, Abbasimehr H, Riggs RJ, Esmaeili M, Majdabadkohne RM, Pasehvar M. Chatgpt: applications, opportunities, and threats. In: 2023 Systems and Information Engineering Design Symposium (SIEDS), IEEE; 2023. pp. 274–279.
20. Achiam J, Adler S, Agarwal S, Ahmad L, Akkaya I, Aleman FL, Almeida D, Altenschmidt J, Altman S, Anadkat S, et al. Gpt-4 technical report. arXiv preprint arXiv:2303.08774. 2023.
21. Chowdhery A, Narang S, Devlin J, Bosma M, Mishra G, Roberts A, Barham P, Chung HW, Sutton C, Gehrmann S, et al. Palm: scaling language modeling with pathways. J Mach Learn Res. 2023;24(240):1–113.
22. Touvron H, Lavril T, Izacard G, Martinet X, Lachaux M-A, Lacroix T, Rozière B, Goyal N, Hambro E, Azhar F, et al. Llama: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971. 2023.
23. Turhan GD. Life cycle assessment for the unconventional construction materials in collaboration with a large language model. In: Proceedings of the International Conference on Education and Research in Computer Aided Architectural Design in Europe. Education and Research in Computer Aided Architectural Design in Europe 2023.
24. Lee J, Jung W, Baek S. In-house knowledge management using a large language model: focusing on technical specification documents review. Appl Sci. 2024;14(5):2096.
25. Han D, Zhao W, Yin H, Qu M, Zhu J, Ma F, Ying Y, Pan A. Large language models driven bim-based dfma method for free-form prefabricated buildings: framework and a usefulness case study. J Asian Arch Build Eng. 2024; 1–18.
26. Zheng J, Fischer M. Dynamic prompt-based virtual assistant framework for bim information search. Autom Constr. 2023;155: 105067.
27. Zhang J, Liang Z, Chan JCF. Heritage building preservation through multimodal llm and language-embedded 3dgs: a novel digital twin with effective visualization, documentation, and querying. Documentation, and Querying 2024.
28. Robertson S, Zaragoza H, et al. The probabilistic relevance framework: Bm25 and beyond. Foundations Trends® Inf Retrieval. 2009;3(4):333–89.
29. Devlin J, Chang M-W, Lee K, Toutanova K. Bert: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805. 2018.
30. Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S, et al. An image is worth 16x16 words: transformers for image recognition at scale. arXiv preprint arXiv:2010.11929. 2020.
31. Radford A, Kim JW, Hallacy C, Ramesh A, Goh G, Agarwal S, Sastry G, Askell A, Mishkin P, Clark J, et al. Learning transferable visual models from natural language supervision. In International Conference on Machine Learning. PMLR; 2021, pp. 8748–8763.
32. Li J, Li D, Savarese S, Hoi S. Blip-2: bootstrapping language-image pre-training with frozen image encoders and large language models. arXiv preprint arXiv:2301.12597. 2023.
33. Zhang L, Rao A, Agrawala M. Adding conditional control to text-to-image diffusion models. In Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 3836–3847.
34. Taori R, Gulrajani I, Zhang T, Dubois Y, Li X, Guestrin C, Liang P, Hashimoto TB. Alpaca (2023) A strong, replicable instruction-following model. Stanford Center for Research on Foundation Models. 2023;3(6): 7. https://crfm.stanford.edu/2023/03/13/alpaca.html
35. Chiang W-L, Li Z, Lin Z, Sheng Y, Wu Z, Zhang H, Zheng L, Zhuang S, Zhuang Y, Gonzalez JE, et al. Vicuna: an open-source chatbot impressing gpt-4 with 90%* chatgpt quality. 2023. https://vicuna.lmsys.org. Accessed 14 Apr 2023.

Publisher's Note