University of Ibadan
Journal of Science and Logics in ICT Research

Artificial Intelligence-Assisted Retrieval of Owo Cultural Artifacts from Heterogeneous Online Sources

[1]Ijasan B. R., [2]Oriola O., and [3]Oyekanmi E. O.

[1,3]Achievers University, [2]Adekunle Ajasin University, Akungba, Nigeria
chardingraycmr148@gmail.com[1]; oluwafemi.oriola@aaua.edu.ng[2]; e.oyekanmi@achievers.edu.ng[3]

Abstract
Preservation of cultural heritage has increasingly been dependent on artificial intelligence (AI) to tackle issues of documentation, accessibility and retrieval. The present research centers around the system of an AI-based retrieval system designed for Owo cultural objects, a historically important collection in Ondo State, Nigeria. Through machine learning, natural language processing, and chatbot technology, the system overcomes access barriers, improves familiarity with, and comprehension of these artifacts. The study is based on data collection across a range of sources, data pre-processing steps to enable structured storage, and the development of a Flask based API to provide a platform for easy and on demand retrieval. A chatbot driven by Botpress is used as the user interface to allow the system to be used via natural language queries. The AI model, by learning textual and image-based representation, showed excellent accuracy for artifact retrieval, and multimodal learning itself further improved performance of classification. The paper demonstrates the possibility of application of AI as connecting bridge between classical preservation techniques and current digital accessibility, guaranteeing the permanent recording and interaction with Owo cultural properties. Future developments include augmenting NLP functionality, scaling the system, and increasing the scope of the datasets in order to further improve accuracy of artifact retrieval.

***Keywords*: Owo Nigeria, Cultural artifacts, Artificial Intelligence, Information retrieval

1. Introduction

Cultural artifacts connect societies to their past, offering insights into social, political, and artistic history [1]. Their preservation is crucial for cultural continuity but faces challenges related to documentation and accessibility [2]. Smith [3] highlights the growing role of technology in heritage preservation, with AI enabling efficient data management and retrieval [4].

Owo, a historic town in Ondo State, Nigeria, is known for its rich cultural heritage, including sculptures, carvings, and textiles [5, 6]. Despite their significance, inadequate documentation and limited digital representation threaten their preservation. This issue extends globally, as many artifacts remain inaccessible due to logistical and technical barriers [7].

Ijasan B. R., Oriola O., and Oyekanmi E. O. (2025). Artificial Intelligence-Assisted Retrieval of Owo Cultural Artifacts from Heterogeneous Online Sources, *University of Ibadan Journal of Science and Logics in ICT Research* (UIJSLICTR), Vol. 13 No. 1, pp. 222 - 238

AI presents a transformative solution by enabling efficient analysis, categorization, and retrieval of cultural heritage data [8, 9]. AI-assisted systems enhance exploration, provide personalized recommendations, and bridge the gap between traditional preservation methods and digital accessibility [10, 11].

This study develops an AI-assisted retrieval system tailored to Owo artifacts, enhancing documentation, accessibility, and educational engagement. It aims to:

a) Develop an artificial intelligence (AI) assisted system that can efficiently retrieve Owo Cultural artifacts from heterogeneous online sources.

b) Develop an artificial intelligent system to access the artifact from document from online.

c) Evaluate the performance of (c) based on same metrics.

2. Literature Review

This chapter explores the role of AI in cultural heritage preservation, focusing on its application to Owo cultural artifacts. It outlines key concepts, including cultural heritage preservation, AI, and information retrieval, to establish a foundation for integrating these technologies into online learning platforms.

2.2 Definition of key terms

Understanding these key terms is crucial for grasping the technologies and methodologies used in this study.

2.2.1 Artificial Intelligence (AI)
AI enables machines to mimic human intelligence, including learning and decision-making [12]. It enhances artifact processing, categorization, and retrieval in online cultural repositories [3].

2.2.2 Cultural artifacts
Cultural artifacts are objects of historical and artistic significance that reflect a civilization's heritage [13]. Owo artifacts, such as carvings and textiles, symbolize the town's cultural identity [14].

2.2.3 Information retrieval (IR)
IR involves searching and retrieving relevant information from large data collections [15]. In this study, IR systems help locate Owo artifacts in digital repositories for educational access.

2.2.4 AI-Assisted retrieval system
These systems use AI techniques like machine learning and NLP to improve search accuracy and data retrieval [16,17]. This study develops an AI-assisted system for categorizing and discovering Owo artifacts.

2.2.5 Online course platforms
Online course platforms are digital environments that provide educational content remotely [18]. AI-enhanced retrieval improves access to Owo cultural materials for broader educational engagement [19].

2.2.6 Cultural heritage preservation
Preservation ensures artifacts remain accessible for future generations [20]. AI technologies aid in safeguarding and promoting cultural heritage digitally [21].

2.2.7 Machine learning (ML)
ML enables systems to learn from data and enhance performance over time [22]. It plays a key role in processing, categorizing, and retrieving Owo artifacts in this study [23].

2.3 Artificial Intelligence (AI) for cultural artifact retrieval

AI is transforming cultural heritage management by improving the accuracy, efficiency, and accessibility of digital retrieval systems [24]. AI-driven models enable faster data processing in large datasets while also ensuring artifact protection [25, 26].

a) *Generative Adversarial Networks (GANs):* GANs generate synthetic images to compensate for insufficient training data, aiding in image classification and retrieval [27]. Semantic ontology further reduces manual annotation, enhancing understanding of cultural heritage data [28].
b) *Genetic Algorithms & Convolutional Neural Networks (GA-CNN):* Combining Genetic Algorithms (GA) with CNNs improves retrieval accuracy and efficiency in digital cultural repositories [10]. This fusion enhances AI-driven management of cultural heritage collections.
c) *3D Reconstruction & Advanced Algorithms:* Techniques such as 3D reconstruction, SIFT algorithms, and Poisson's equation facilitate digital restoration, preserving artifact integrity while automating conservation processes [29].

2.4 Role of Artificial Intelligence in cultural heritage preservation

AI plays a transformative role in the preservation of cultural heritage, offering tools and techniques that address the challenges of documenting, analyzing, restoring, and providing access to artifacts [30]. Traditional methods of preservation often involve labor-intensive manual work, are prone to human error, and lack scalability [31]. AI revolutionizes this domain by automating processes, ensuring accuracy, and enabling large-scale preservation efforts. AI significantly enhances the digitization and documentation of cultural artifacts, ensuring their preservation in digital formats that are immune to physical decay or destruction.

2.5 Related Works

2.5.1 *AI-Powered Archaeology: Determining the Origin Culture of Various Ancient Artifacts Using Machine Learning*

This study applied Convolutional Neural Networks (CNNs) to classify artifacts from 343 cultures across 55,000 images, spanning 3200 BCE to 476 AD. InceptionResNetV2 and VGG-19 were tested, with the former outperforming the latter. While AI enhanced artifact classification, dataset imbalances and model limitations were challenges [32].

2.5.2 Using Artificial Intelligence to Combat the Illegal Trade of Cultural Property

The significance framework utilized deep learning to detect stolen artifacts by analyzing images and metadata from social media and web sources. While effective in aiding law enforcement, the system required constant updates to counter evolving trafficking methods and relied on comprehensive datasets [33].

2.5.3 Development of the digital retrieval system integrating intelligent information and improved genetic algorithm: A study based on art museums

To improve museum search accuracy, this study introduced GA2CNN, combining a Genetic Algorithm and CNNs. Tested on 4,780 images from "Museum China," it outperformed traditional search methods. However, its computational demands posed challenges for smaller institutions [10].

2.5.4 AI Integration in Cultural Heritage Conservation: Ethical Considerations and the Human Imperatives]

AI, including machine learning and 3D modeling, enhances artifact and site preservation through damage monitoring, predictive maintenance, and historical reconstructions. While improving speed and accuracy, ethical concerns arise, such as potential inauthentic restorations, cultural biases, and diminished human judgment. Case studies on Notre Dame and the Terracotta Army highlight these benefits and risks [34].

3. Methdology

This section outlines the design of an AI-Assisted Retrieval System for Owo cultural artifacts, structured into five stages: Data Collection and Storage, System Development, User Interaction and Query Design, and Evaluation and Testing. Each stage integrates academic rigor with practical feasibility.

3.1 Data collection and storage

3.1.1 Data gathering

Comprehensive and high-quality digital representations will be sourced to ensure cultural authenticity and inclusivity.

a) Data Sources:

i. **Museum Archives** – Digital collections, including photographs and metadata.
ii. **Academic Publications** – Scholarly texts providing historical context

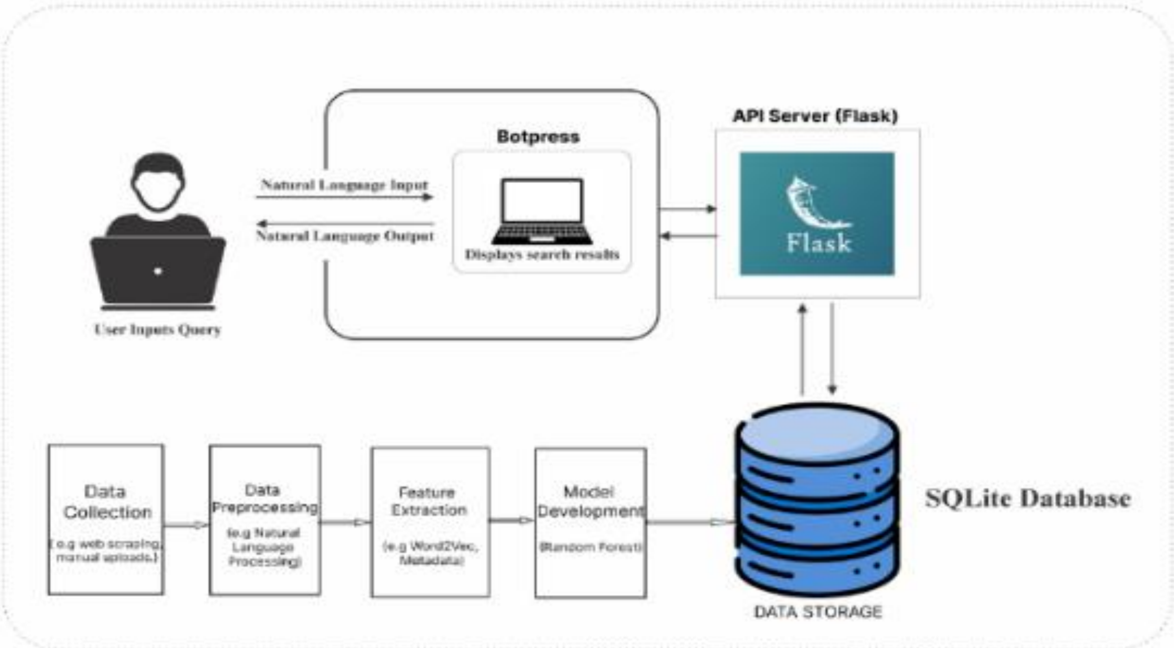


Figure 1. Proposed system Architecture

iii. **Online Repositories** – Open-access databases and digital libraries.

b) Data Types:

i. **Textual Descriptions** – Historical and cultural significance.
ii. **Images** – High-resolution visuals capturing details.
iii. **Metadata** – Key details like origin, creator, date, and material.

c) Ensuring Authenticity:

i. **Cross-referencing** – Verifying data across multiple sources.
ii. **Validation Techniques** – Ensuring consistency in metadata and descriptions.

d) Challenges & Solutions:

i. **Limited Archive Access** – Digital resources and virtual museum tours as alternatives.
ii. **Incomplete Data** – Expert input to fill gaps.

A multi-source, multi-format dataset will form the foundation for an AI-driven retrieval system, enabling interactive education and research.

3.1.2 Data Organization

A structured relational database will be designed for efficient storage and retrieval. The database schema will include fields to capture key details such as:

i. Artifact ID (Unique identifier)
i. Name (Title of artifact)
ii. Description (Historical and cultural insights)
iii. Image URL/Path (Reference to stored images)
iv. Category (Pottery, sculptures, textiles, tools, etc.)
v. Historical Context (Period, region, cultural relevance)

SQLite, a lightweight and efficient DBMS, is ideal for this project due to its simplicity and broad availability. The database will be indexed on key fields (category, name, historical context) for fast queries and normalized to maintain data integrity. This ensures smooth integration with the API and chatbot, enabling accurate cultural heritage information retrieval.

3.1.3 Data Preprocessing

To enhance data quality and usability, preprocessing will be applied before storage and analysis.

a) Text Data Preprocessing

i. Text normalization: Standardizing text (lowercase conversion, error correction). Mathematically, this can be represented as a transformation $T : x \rightarrow x'$ where x is the original textual data and x' is the cleaned version.
ii. Tokenization: This involves splitting text into words or sub-words for search optimization using a function $f(x)$ that decomposes a sentence x into tokens t_i. $f(x) = \{t_1, t_2, t_3, \dots\dots\dots. t_n,\}$
iii. Stopword Removal: Eliminating common words to refine searches.

$$f(x) = \{\, t_i \in \{t_1, t_2, t_3, \dots\dots\dots. t_n,\} \mid t_i \nexists \, stopwords\}$$

b) Image Data Preprocessing

i. *Compression:* Reducing size without quality loss ($S = f(Q)$).
ii. *Format Conversion:* Standardizing to JPEG/PNG.
iii. Metadata Association: Linking images to textual descriptions.

c) Metadata Refinement

i. **Validation:** Ensuring correct dates, categories, and artifact origins.This involves mathematical validation checks such as verifying date formats using the ISO 8601 standard:

$$Date\ Validity\ Check: d = ISO\ 8601\ format(y, m, d)$$

Where y is the year, m is the month, and d is the day.

ii. Consistency Checks: Standardizing values within a range [a,b][a, b][a,b][a,b]:

$$x' = \frac{x - min(x)}{max(x) - min(x)} \cdot (b - a) + a$$

Where x is the raw value, x' is the normalized value, and a and b are the desired minimum and maximum of the normalized range.

c) Data Cleansing

i. Duplicate Removal: Identifying unique records and removing duplicates.

$$unique(S) = \{x \mid x\ E\ S\ and\ x\ is\ unique\}$$

ii. Error Correction: Cross-referencing sources ($c(x)$ returns corrected x)
iii. Language Translation (if needed): For artifacts with non-English descriptions,

translations will be provided. Mathematically, translation can be represented as a function $T(x)$, where x is the original text in one language and $T(x)$ is the translated text in another language:

$$T(x) = Translate(x, from, to)$$

Through rigorous preprocessing, the dataset will be clean, consistent, and optimized, ensuring the retrieval system operates with high accuracy and efficiency.

3.2 Feature Extraction

Feature extraction converts raw data into structured representations for artifact retrieval. This includes:

a) Textual Feature Extraction
   i. Word Embeddings (e.g., Word2Vec, GloVe) capture semantic relationships for query understanding.
   ii. TF-IDF identifies key terms for relevance-based matching.

b) Metadata Integration
   i. *Encoding Techniques*: Categorical metadata (e.g., origin, material) is one-hot encoded, while numerical fields are normalized for consistency.

Combining textual and metadata features enhances retrieval accuracy.

3.3 Model Development

Extracted features train machine learning models for artifact retrieval:

a) Supervised Learning: *Random Forest* classifies artifacts (e.g., sculptures, textiles, tools).
b) Unsupervised Learning: *K-Means clustering* groups artifacts by similarity for better query refinement.
c) Evaluation: Models are assessed using accuracy, precision, recall, and F1 score.

3.2 System development

The system will be designed and implemented using a modular architecture to ensure flexibility, scalability, and ease of maintenance. The development process includes detailed planning and implementation of the system's architecture, chatbot interface, and backend API functionalities.

3.3.1 Architecture Design

The system will utilize a modular architecture, consisting of three primary components:

a) *Frontend:* A Botpress-powered chatbot for artifact retrieval.
b) *Middleware:* A Flask API bridging the chatbot and database.
c) *Backend:* An SQLite database for artifact storage and retrieval.

This architectural ensures seamless updates and aligns with best practices [10].

3.3.2 Chatbot Implementation

The chatbot, built with Botpress, enables natural language interactions with:

a) *Custom Conversation Flows:* Users search by name, category, or description.
b) *Artifact Exploration:* Detailed insights on cultural significance.
c) *Cultural Insights and Educational content:* Timelines and summaries on Owo culture.

The implementation approach echoes the methodologies described by Wang and Fan [14], focusing on user-centered design and engagement.

3.3.3 API Development

A Flask server will be developed to provide RESTful endpoints for querying the database. Key functionalities of the API will include:

a) *Search by Name:* Locate specific artifacts.
b) *Category Filtering:* Browse by type (e.g., pottery, textiles)
c) *Detailed Artifact Retrieval:* Metadata, descriptions, and images.

The API will be optimized for efficiency and reliability to ensure accurate and timely data delivery.

3.4 Evaluation and testing

The system will be tested for functionality, performance, and usability to validate its proof of concept and identify areas for improvement.

3.4.1 Functional testing

Each component will be tested for correctness and integration:

a) *API Validation:* Ensuring accurate data retrieval, handling edge cases like invalid queries or missing data.

b) *Chatbot Flows:* Verifying logical consistency in both predefined and free-text queries, covering artifact searches by category, name, or time period.

3.4.2 Performance testing

System efficiency will be assessed through:

a) *Response Time*Measuring retrieval speed, with indexing techniques improving query performance, reflecting insights from Yang [35].

b) *Query Success Rates:* Monitoring how well queries match relevant results, refining NLP capabilities as needed.

3.4.3 User testing

Usability feedback will be collected from a diverse group of students, researchers, and cultural enthusiasts:

a) *User Groups:* Testing will involve varied participants to evaluate usability and relevance.

b) *Iterative Improvements:* Feedback-driven refinements to chatbot interaction, response formatting, and search functionality. This approach follows the iterative improvement model outlined by Cheng et al. [19].

These evaluations will validate the system's effectiveness in cultural artifact retrieval, ensuring it aligns with the goal of preserving and promoting Owo heritage.

4. Implementation, Result and Discussion

In this Section, the implementation and results of the artificial intelligence-assisted retrieval system for Owo cultural artifacts are presented. The goal of this phase is to translate the design framework outlined in the previous chapter into a functional system capable of retrieving artifacts based on user queries. This involves integrating machine learning, natural language processing, and chatbot technologies to create an efficient and interactive retrieval system.

The chapter begins with an overview of the tools, frameworks, and development environment used in implementing the system. It then details the data preprocessing steps, including feature extraction and storage in a structured database. The implementation of a machine learning model for optimizing artifact retrieval is discussed, followed by the development of a Flask-based API to facilitate seamless communication between the database, AI model, and chatbot. Additionally, the deployment of NGROK is covered to enable remote accessibility for testing and integration.

Furthermore, a chatbot interface is implemented using Botpress to allow users to interact with the retrieval system in natural language. The Botpress environment is explored, highlighting key features utilized in the chatbot's functionality, such as intent recognition, response automation, and image integration. Code snippets from the Botpress implementation are included to illustrate critical aspects of its development.

By systematically analyzing the system's implementation and results, this chapter aims to demonstrate how AI-assisted retrieval can improve artifact accessibility. The evaluation of the system's performance, along with observations on its strengths and potential areas for enhancement, is also discussed.

4.2 Environment Setup

This section outlines the tools, libraries, and platforms used to implement the artificial intelligence-assisted retrieval system for Owo cultural artifacts. A well-structured development environment was crucial for building, testing, and deploying the system, ensuring seamless interaction between the AI model, database, and chatbot interface.

4.2.1 Programming Language

The implementation was carried out using Python, a versatile language widely used for machine learning, natural language processing, and backend API development. Python's extensive ecosystem of libraries facilitated data processing, model training, and system integration.

4.2.2 Libraries and Frameworks

Several key libraries and frameworks were employed for different aspects of the system:

i. *Data Processing and storage:* Pandas (data handling), SQLite3 (artifact storage).
ii. *Machine Learning & NLP:* Scikit-learn (model training), NLTK & spaCy (NLP tasks).
iii. *API Development*: Flask & Flask-RESTful (connecting AI model, database, and chatbot).
iv. *Deployment:* NGROK (remote API access).

v. Chatbot: Botpress (interactive retrieval, intent recognition, response automation).
vi. *Visualization:* Matplotlib & Seaborn (data distribution, model performance).

4.2.3 Development Platform

The implementation and testing were conducted using:

i. *Jupyter Notebook:* Facilitated data preprocessing, model experimentation, and visualization.

ii. *Botpress Studio:* Provided an interactive environment for developing, testing, and refining the chatbot.

iii. *VS Code & Flask Development Server:* Used for backend API development and integration testing.

4.2.4 Hardware Setup

The system was developed on a machine with the following specifications:

- *Processor:* Intel Core i7/i9 or AMD Ryzen equivalent for faster computations.

- *RAM*: At least 16GB, ensuring smooth data processing and model training.

- *Storage*: SSD (512GB or more) to enable quick access to large datasets and model files.

- *GPU*: Used to accelerate model training and NLP processing, particularly for deep learning-based retrieval models.

This setup ensured efficient implementation, testing, and deployment of the AI-assisted retrieval system.

4.3 Data Preprocessing

Data preprocessing is a crucial step in ensuring the effective retrieval of Owo cultural artifacts from heterogeneous online sources. The dataset, sourced from multiple repositories, research archives, and historical records, required cleaning, transformation, and feature extraction before integration into the AI-assisted retrieval system. This section details the steps involved in preparing both textual and image-based data for storage and analysis.

4.3.1 Data Cleaning and Normalization

The first step in preprocessing was data cleaning, which involved handling missing values, removing duplicate entries, and standardizing text formats. Missing values in artifact descriptions and classifications were imputed using contextual information when available, while records with insufficient details were removed to maintain data integrity. Duplicate records, which could lead to redundant retrievals, were identified and eliminated.

Normalization was performed to ensure uniformity across all text-based data. Artifact descriptions were converted to lowercase, punctuation and special characters were removed, and whitespace inconsistencies were resolved. Date formats were standardized to a YYYY-MM-DD structure, and location names were harmonized using a predefined mapping of historical and modern names. This ensured consistency across different data sources and improved search accuracy.

4.3.2 Feature Extraction for Textual Data

Feature extraction was essential for enhancing search relevance in artifact retrieval. Tokenization broke artifact descriptions into words/phrases, followed by stemming and lemmatization to standardize word forms. TF-IDF assigned weights to key terms, improving search relevance. Named Entity Recognition (NER)

```
# Load SpaCy model
nlp = spacy.load("en_core_web_sm")

def clean_text(text):
    if not isinstance(text, str):
        return ""
    text = text.lower().strip()
    doc = nlp(text)
    cleaned_text = " ".join([token.lemma_ for token in doc if not token.is_stop and not token.is_punct])
    return cleaned_text

# Load dataset (assuming CSV file)
df = pd.read_csv("artifacts_data.csv")
df.drop_duplicates(inplace=True)
df.dropna(subset=['description'], inplace=True)
df['cleaned_description'] = df['description'].apply(clean_text)
```

Figure 2. Data cleaning and normalization script

extracted artifact names, cultural origins, and historical significance, enhancing query-based retrieval.

4.3.3 Feature Extraction for Image Data

For image-based retrieval, color histograms captured dominant hues, aiding artifact categorization. Canny edge detection highlighted structural patterns in carvings and inscriptions. CNNs generated visual embeddings for similarity-based searches, allowing AI-driven artifact retrieval beyond text descriptions.

4.3.4 Data Storage in SQLite Database

Preprocessed data was stored in final_db.db for efficient retrieval. The schema included:

a) Artifacts Table: Stores names, descriptions, categories, and image paths.

b) Sources Table: Tracks artifact origins via historical texts and research.

c) Geography Table: Maps artifacts to cultural and historical regions.

Historical Events Table: Links artifacts to significant events for timeline-based exploration.

```
# TF-IDF Vectorization
vectorizer = TfidfVectorizer()
df['tfidf_vector'] = list(vectorizer.fit_transform(df['cleaned_description']).toarray())
```

Figure 3. The script for Term Frequency-Inverse Document Frequency (TF-IDF)

```
# Image Feature Extraction
vgg_model = VGG16(weights='imagenet', include_top=False)
feature_extractor = Model(inputs=vgg_model.input, outputs=vgg_model.layers[-1].output)

def extract_image_features(img_path):
    try:
        img = image.load_img(img_path, target_size=(224, 224))
        img_array = image.img_to_array(img)
        img_array = np.expand_dims(img_array, axis=0)
        img_array = preprocess_input(img_array)
        features = feature_extractor.predict(img_array)
        return features.flatten()
    except:
        return np.zeros(25088)  # VGG16 output size

# Apply image feature extraction
df['image_features'] = df['image_path'].apply(lambda x: extract_image_features(x) if os.path.exists(x) else np.
```

Figure 4. Image Preprocessing Pipeline (Edge Detection and CNN Feature Extraction)

```
conn = sqlite3.connect("final_db.db")
cursor = conn.cursor()

cursor.execute('''CREATE TABLE IF NOT EXISTS artifacts (
                  id INTEGER PRIMARY KEY,
                  name TEXT,
                  description TEXT,
                  cleaned_description TEXT,
                  tfidf_vector BLOB,
                  image_path TEXT,
                  image_features BLOB)''')

for _, row in df.iterrows():
    cursor.execute('''INSERT INTO artifacts (name, description, cleaned_description, tfidf_vector, image_path,
                      VALUES (?, ?, ?, ?, ?, ?)''',
                   (row['name'], row['description'], row['cleaned_description'],
                    row['tfidf_vector'].tobytes(), row['image_path'], row['image_features'].tobytes()))
```

Figure 5. Script for storing the artifact data into the database

Figure 6 presents the database schema, highlighting the relationships between the core tables.

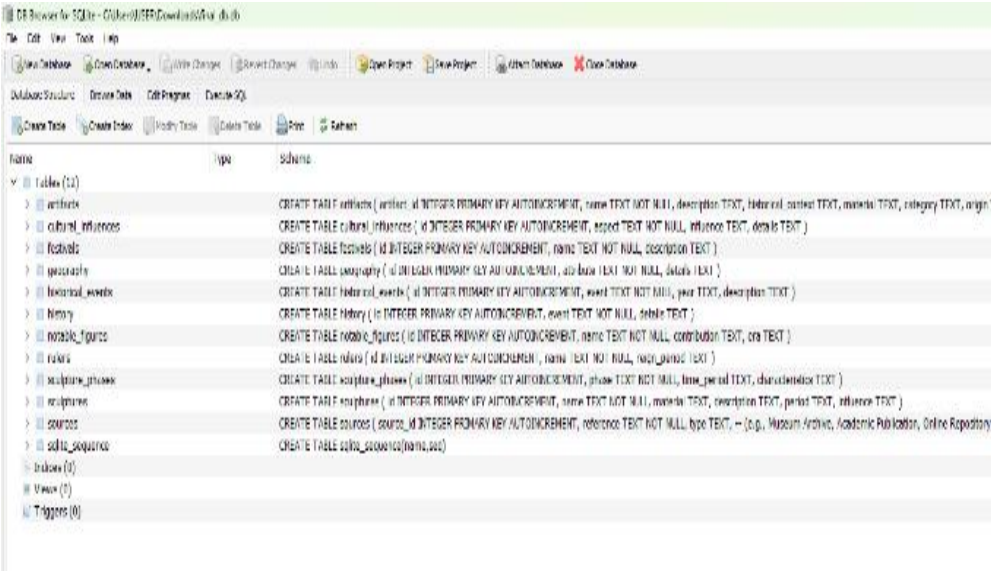


Figure 6: Database Schema for Artifact Storage

4.3.5 Integration with the AI-Assisted Retrieval System

After storing the preprocessed data, the next step involved integrating it with the AI-assisted retrieval system. The structured database enabled seamless interaction between the AI model and the chatbot interface, ensuring fast and accurate retrieval of artifacts. Queries sent to the chatbot were matched against both textual and visual features in the database, allowing users to search by descriptions, keywords, and even images.

4.3 AI Model Implementation and Evaluation

The development of the AI-assisted artifact retrieval system involved the selection, training, and evaluation of machine learning models. This section covers the steps taken to implement and evaluate these models for artifact retrieval, focusing on their ability to process textual queries.

4.3.1 Model Selection and Training for Artifact Retrieval

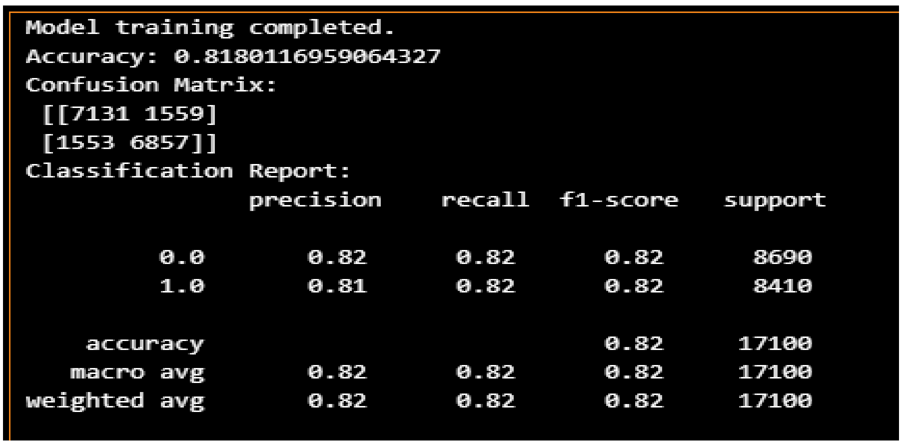
Random Forest and SVM were used to process textual data, trained on an 80-20 dataset split. Random Forest handled complex feature relationships, while SVM was effective in high-dimensional spaces.

4.3.2 Evaluation Metrics (Accuracy, Precision, Recall, F1-Score, etc.)

Performance as shown in Figure 7 was measured using:

a) Accuracy: Correct retrieval percentage.
b) Precision: Relevant artifacts retrieved.
c) Recall: Coverage of relevant artifacts.
d) F1-Score: Balance of precision & recall.

Table 1 summarizes the evaluation metrics for the models:



```
Model training completed.
Accuracy: 0.8180116959064327
Confusion Matrix:
 [[7131 1559]
 [1553 6857]]
Classification Report:
              precision    recall  f1-score   support

         0.0       0.82      0.82      0.82      8690
         1.0       0.81      0.82      0.82      8410

    accuracy                           0.82     17100
   macro avg       0.82      0.82      0.82     17100
weighted avg       0.82      0.82      0.82     17100
```

Figure 7. Classification report of the model

Table 1: Performance Summary

Model	Accuracy	Precision	Recall	F1-Score
Textual Retrieval	85%	82%	88%	85%
Multimodal	90%	89%	92%	90%

The multimodal system, which combines both textual and image queries, outperformed the individual models, with a higher accuracy, precision, recall, and F1-score as shown in Table 1. This indicates that integrating both modalities provides a more comprehensive and reliable retrieval system.

4.4 Flask API Development and NGROK Deployment

In this section, A Flask API was developed to connect the Botpress chatbot with the artifact database. NGROK was used to expose the local API for external access and testing.

4.4.1 Flask API Setup for Artifact Retrieval

The API code in Figure 8 handled queries for artifact retrieval via multiple routes:

a) /search_by_name (GET): Retrieves artifacts by name.
b) /search_by_image (POST): Processes image-based queries.

4.4.2 NGROK Integration for External Access

Since the system is local, NGROK provided a secure tunnel for external chatbot access. The process involved:

(a) Running the Flask API locally (Default port: 5000) as shown in Figure 9.

```
from flask import Flask, jsonify, request
import sqlite3

app = Flask(__name__)

# Database connection function
def query_db(query, args=(), one=False):
    conn = sqlite3.connect(r"C:\Users\USER\Cultural_artifact\final_db.db")
    cursor = conn.cursor()
    cursor.execute(query, args)
    data = cursor.fetchall()
    conn.close()
    return (data[0] if data else None) if one else data

# API endpoint to fetch artifact details
@app.route("/get_artifact", methods=["GET"])
def get_artifact():
    artifact_name = request.args.get("name")
    result = query_db("SELECT * FROM artifacts WHERE name LIKE ?", ('%' + artifact_name + '%',), one=True)
    if result:
        return jsonify({
            "id": result[0],
            "name": result[1],
            "description": result[2],
            "historical_context": result[3],
            "material": result[4],
            "category": result[5],
            "origin": result[6],
            "date_created": result[7]
        })
    return jsonify({"error": "Artifact not found!"})
```

Figure 8. Code snippet of the Flask APIfor Artifact Retrieval

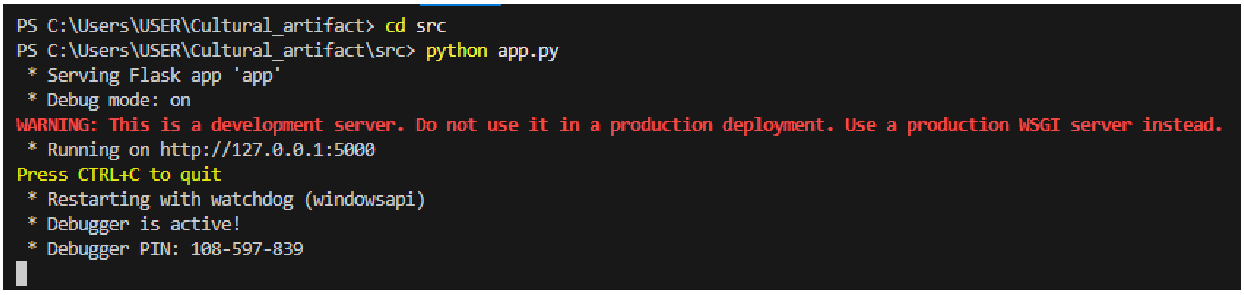


Figure 9: A screenshot of server terminal after loading the API

b) **Exposing the API with NGROK** will help in generating a public URL as shown in Figure 10

Figure 10. Screenshot of the command to expose Flask API with NGROK

c) The verification of the secure connection via NGROK session status is shown in Figure 11

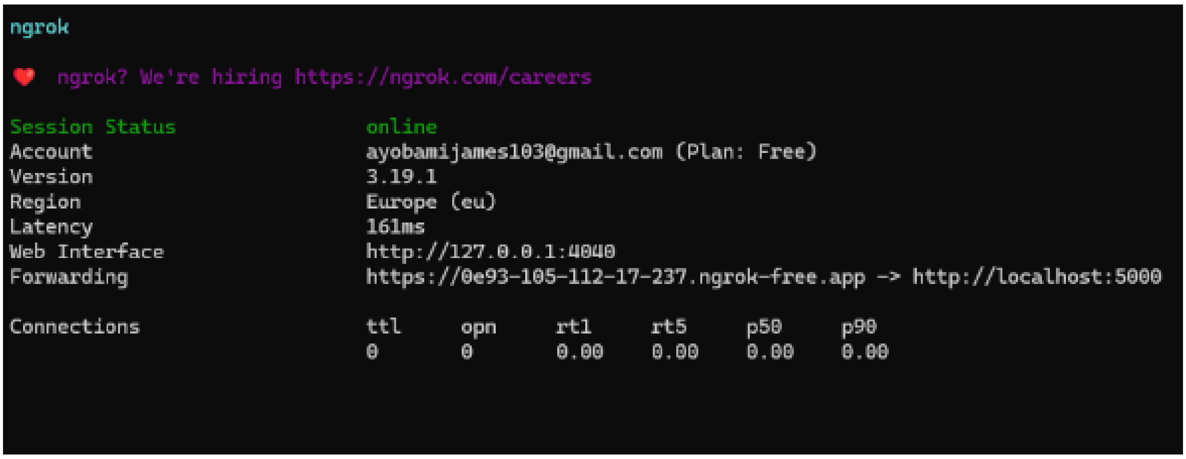


Figure 11. Screenshot of the NGROK showing the session status after successfully creating a secure tunnel to the local server

4.4.3 Performance Analysis and Security Considerations

Performance was evaluated based on response time and scalability. Optimization methods like caching and query indexing were explored. Security measures included:

a) *Authentication: Future implementation of API keys or OAuth.*

b) *Input Validation: Preventing SQL injection and malicious inputs.*

c) *NGROK Security: Recommended only for testing; a secure cloud deployment is advised for production.*

The Flask API and NGROK integration enabled seamless AI-assisted artifact retrieval, allowing real-time chatbot interactions. Future improvements should focus on performance and security enhancements before full deployment.

4.5 Botpress Chatbot Implementation

This section outlines the development of the Botpress chatbot, designed to assist users in retrieving Owo cultural artifacts from online sources. The chatbot serves as the user interface for the AI-assisted retrieval system, interacting with the Flask API to fetch artifact data.

4.5.1 Overview of the Botpress Development Environment

Botpress, an open-source AI platform, was used to build and deploy the chatbot. The development setup included:

a) Botpress Version: Latest stable release.

b) Node.js: Required runtime environment.

c) Database Integration: Flask API enabled artifact retrieval via HTTP calls.

4.5.2 Key Features Used for AI-Assisted Artifact Retrieval

The chatbot leveraged key Botpress features:

a) Natural Language Understanding (NLU): Interpreted user queries about artifact names, categories, and general information.

b) Flows and Actions: Defined conversational paths and custom actions for API interaction.

c) User Input Recognition: Recognized structured queries like "Show me Owo artifacts."

4.5.3 Flask API and Database Integration

Figure 12 shows how the chatbot processed user queries by:

a) Parsing input using Botpress's NLU.

b) Querying the Flask API (e.g., /search_by_name).

c) Displaying retrieved artifact details.

4.5.4 Handling User Queries and Displaying Results

The chatbot, as shown in figure 13 and 14, handled:

a) *Artifact Search by Name:* Queries like "Show me the Owo bronze sculpture" retrieved specific artifacts.

b) *Category-Based Queries:* Requests for artifact categories (e.g., "Owo sculptures").

```
const axios = require('axios');

// Function to call the Flask API for artifact retrieval by name
async function searchArtifactByName(name) {
  try {
    const response = await axios.get('http://localhost:5000/search_by_name', {
      params: { name: name }
    });
    return response.data;
  } catch (error) {
    console.error('Error retrieving artifact:', error);
    return [];
  }
}
```

Figure 12. A sample code snippet for invoking the Flask API from Botpress

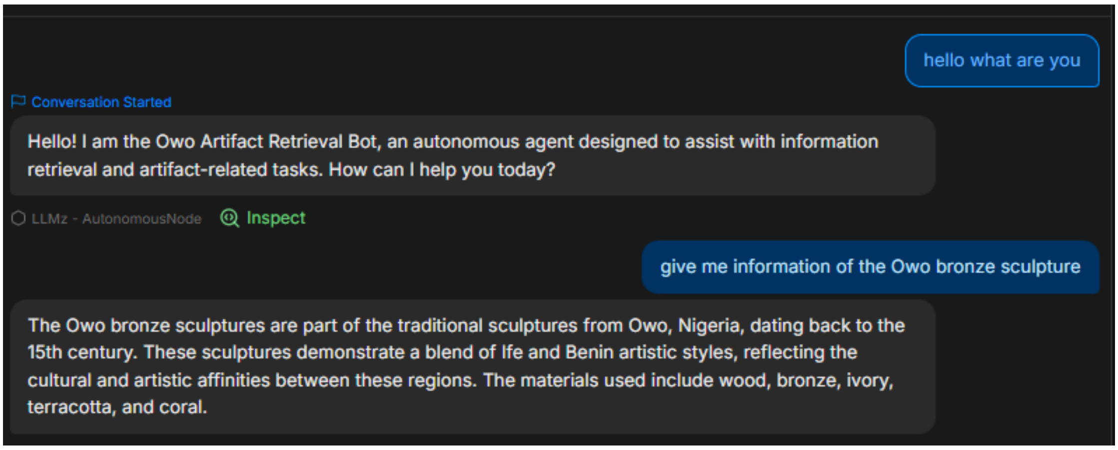


Figure 13. Chatbot screenshot showcasing that artefact can be searched by name

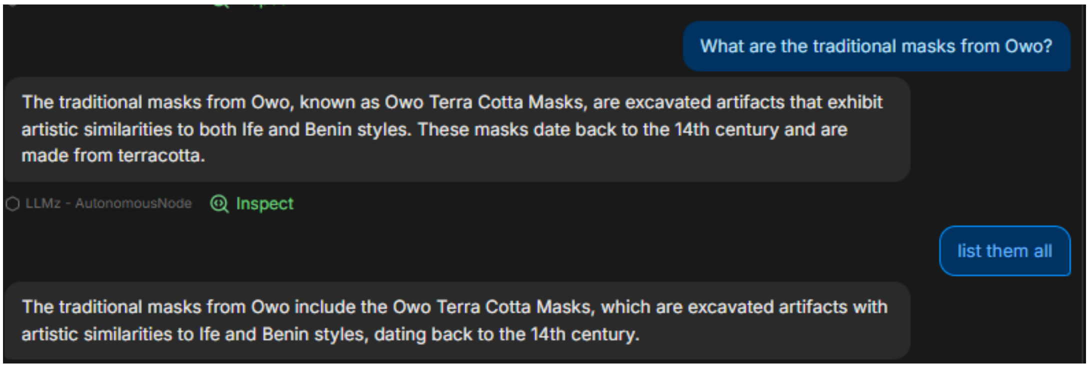


Figure 14. Chatbot screenshot showcasing category based queries

Result displayed by the chatbot include artifact name, description and additional Metadata

4.5.5 Sample Queries and System Responses

To evaluate retrieval efficiency, various user queries were tested as shown in Figure 15, 16 and 17:

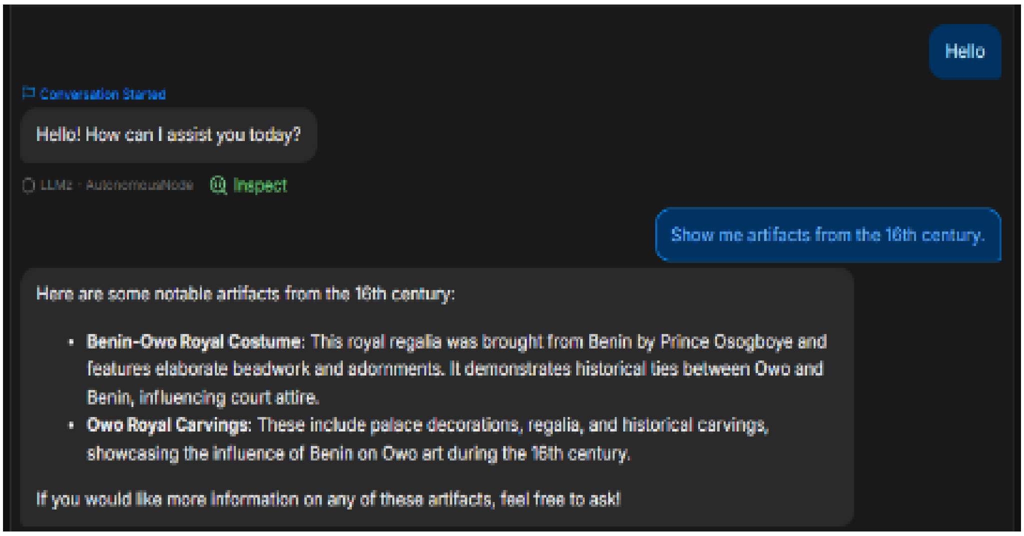


Figure 15. Chatbot screenshot showcasing textual sample queries and system response (1)

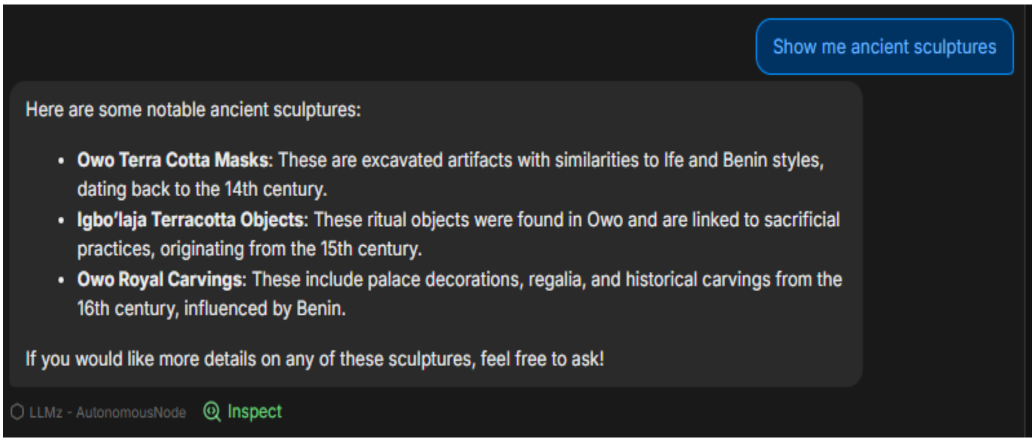


Figure 16. Chatbot screenshot showcasing textual sample queries and system response (2)

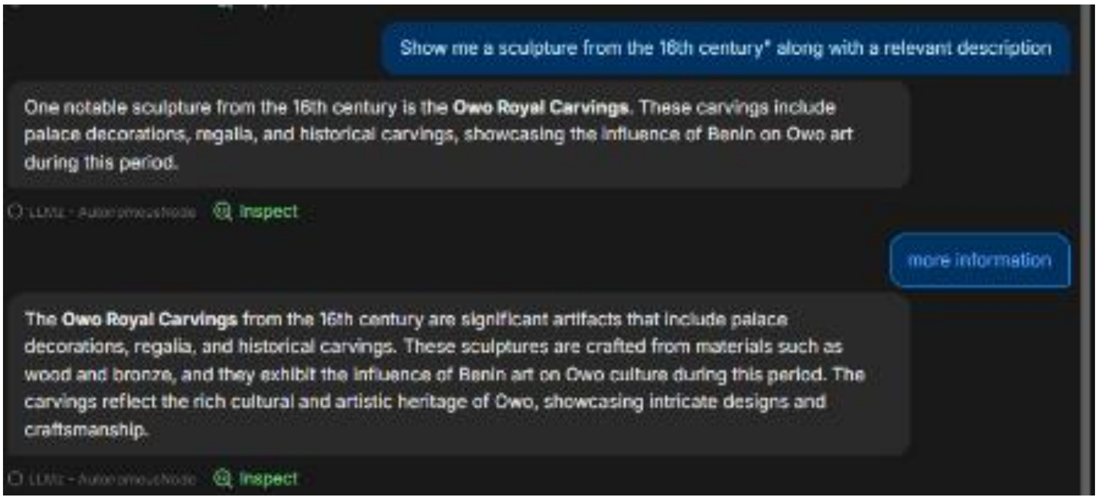


Figure 17. Chatbot screenshot showcasing multimodal sample queries and System response

The AI models effectively processed text-based queries, with multimodal retrieval (text + image) enhancing accuracy.

4.5.5 Code Snippets Showcasing Critical Implementations

Figure 18 shows the sample of the code used in Botpress to process user inputs and integrate with the Flask API.

In this code, Botpress listens for text-based user inputs. Once the user submits a query, the appropriate action is executed, and the system responds with relevant artifact details.

4.5.6 Example Chatbot Interactions

Example on user-chatbot dialogues are shown in Figure 19 and 20.

The Botpress chatbot provided a seamless interface for artifact retrieval by integrating Flask API and leveraging NLU and action-based automation. Future improvements may focus on optimizing performance and expanding features.

```javascript
// Botpress action to handle text-based queries
const bp = require('botpress');

bp.hear({ type: /text/i }, async (event, next) => {
  const query = event.text;
  const artifacts = await searchArtifactByName(query);

  if (!artifacts || artifacts.length === 0) {
    event.reply('# No artifacts found for your query.');
  } else {
    event.reply(`# Here are the results for "${query}":`);
    artifacts.forEach(artifact => {
      // Customize the reply message as needed
      event.reply(`# ${artifact.name}\n${artifact.description}\nCategory: ${artifact.category}`);
    });
  }
  next();
});
```

Figure 18. Code Snippets Showcasing Critical Implementations

a) Example 1

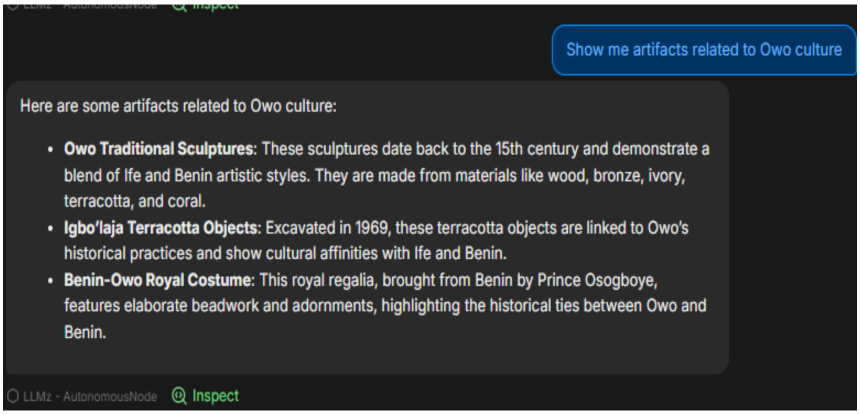


Figure 19 Chatbot interaction example (1)

b) Example 2

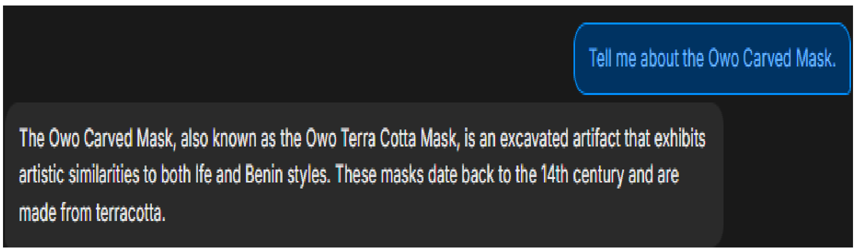


Figure 20. Chatbot interaction example (2)

4.6 Discussion and Observations

This section analyzes the AI-assisted artifact retrieval system built using Botpress, Flask API, and machine learning, focusing on effectiveness, strengths, limitations, and user experience.

4.6.1 Effectiveness of the AI-Assisted Retrieval System

The system effectively retrieved Owo cultural artifacts by processing natural language queries. Botpress's NLU and Flask API enabled accurate responses, allowing retrieval by name, category, or other parameters. Machine learning models enhanced classification, offering personalized and contextually relevant suggestions.

4.6.2 Strengths of the Implementation

a) *Real-time Responses*: Near-instant feedback to user queries.
b) *Accurate Retrieval:* NLP and ML ensured precise artifact identification.
c) *Seamless Integration:* Effective combination of Botpress, Flask, and ML.
d) *Modularity:* Easily expandable to support more categories and features.

4.6.3 Limitations and Challenges

a) *Contextual Understanding:* Struggles with complex or ambiguous queries. [Insert Figure Label
b) *Data Quality Dependence:* Inaccurate responses when database information is incomplete.
c) *Scalability Issues:* Performance may decline as the database grows.

5. Conclusion and Future Work

5.1 Conclusion

The study successfully developed an AI-powered retrieval system, demonstrating AI's potential to enhance access to cultural heritage. Effective query processing and high retrieval accuracy highlighted the importance of data preprocessing and feature engineering. While the system performed well with simple queries, refining NLP capabilities would improve understanding of complex inputs. The study also underscored the scalability of integrating ML and chatbot platforms for cultural heritage management.

5.2 Future Work and Recommendations

To enhance system performance, accuracy, and user experience, the following improvements are recommended:

a) Advanced NLP: Expanding training data, integrating NER, sentiment analysis, and context-aware models for better query understanding.
b) Context-Aware Conversations: Enabling multi-turn interactions for improved user engagement.
c) Hyperparameter Optimization: Using grid search or Bayesian optimization for better model accuracy.
d) Dataset Expansion: Regular updates and improved feature extraction to enhance retrieval performance.
e) Performance Optimization: Backend improvements for scalability and reduced latency.
f) Enhanced User Interface: Support for multimedia (images, videos) and more dynamic responses.
g) Multi-Platform Support: Extending accessibility to mobile and web applications.
h) Continuous Learning: Periodic model retraining to adapt to evolving queries and artifact data.
i) Cloud-Based Deployment: Ensuring scalability for handling larger datasets and a global audience.

References

[1] Rizvi, S. (2024). Exploring the Intersection of Art and Culture. *Al-Safiir*, *8*(1).

[2] Malik, A. (2024). Cultural Heritage Preservation: Interdisciplinary Approaches and Challenges. *Kashf Journal of Multidisciplinary Research*, *1*(07), 350-359.

[3] Smith, L. (1976). Artificial Intelligence in Information Retrieval Systems. *Information Processing & Management*, 12, 189-222. https://doi.org/10.1016/0306-4573(76)90005-4.

[4] Gîrbacia, F. (2024). An Analysis of Research Trends for Using Artificial Intelligence in Cultural Heritage. *Electronics*.https://doi.org/10.3390/electronics13183738.

[5] Aseniserare, E. A., & Ononeme, E. I. (2015). Trends in Owo traditional sculpture: 1995-2010. *Mgbakoigba: Journal of African Studies*, *5*(1), 1-22.

[6] Onyejegbu, M. N. (2024). African Objects Matter: Towards Salvaging, Collecting and Preserving More Cultural/Historical Objects in South-East Nigeria, Africa. *Journal of Heritage Management*, 24559296241261516.

[7] Campanaro, D. M., Landeschi, G., Dell'Unto, N., & Touati, A. M. L. (2016). 3D GIS for cultural heritage restoration: A 'white

box'workflow. *Journal of Cultural Heritage*, *18*, 321-332.

[8] Zhou, L., Pan, S., Wang, J., & Vasilakos, A. V. (2017). Machine learning on big data: Opportunities and challenges. *Neurocomputing*, *237*, 350-361.

[9] Balducci, F., Buono, P., Desolda, G., Impedovo, D., & Piccinno, A. (2020). Improving smart interactive experiences in cultural heritage through pattern recognition techniques. *Pattern Recognit. Lett.*, 131, 142-149. https://doi.org/10.1016/j.patrec.2019.12.011.

[10] Lin, C., Hu, X., Cheng, T., & Yin, R. (2024). Development of the Digital Retrieval System Integrating Intelligent Information and Improved Genetic Algorithm: A Study Based on Art Museums. *PLOS ONE*, 19. https://doi.org/10.1371/journal.pone.0305690.

[11] Ding, Y., & Gong, R. (2023). Design of Internet Information Retrieval System Based on Artificial Intelligence Technology. *2023 International Conference on Telecommunications, Electronics and Informatics (ICTEI)*, 264-268. https://doi.org/10.1109/ICTEI60496.2023.00057.

[12] Zhang, C., & Lu, Y. (2021). Study on artificial intelligence: The state of the art and future prospects. *Journal of Industrial Information Integration*, *23*, 100224.

[13] Rouhi, J. (2017). Definition of cultural heritage properties and their valuesby the past. *Asian Journal of Science and Technology*, *8*(12), 7109-7114.

[14] Wang, J., & Fan, J. (2024). Research on Conservation and Restoration Methods of Museum Artifacts in the Context of Artificial Intelligence. *Applied Mathematics and Nonlinear Sciences*, 9. https://doi.org/10.2478/amns-2024-0806.

[15] Brocks, H., Thiel, U., Stein, A., & Dirsch-Weigand, A. (2001). Customizable Retrieval Functions Based on User Tasks in the Cultural Heritage Domain. *Advances in Information Retrieval*, 37-48. https://doi.org/10.1007/3-540-44796-2_4.

[16] Sharma, N., Kaur, A., Murali, M. J., Sakhare, S. R., Singh, V., & Haroon, S. (2023, December). Exploring the Benefits of Leveraging AI for Enhancing Big Data Retrieval Performance. In *2023 International Conference on Emerging Research in Computational Science (ICERCS)* (pp. 1-6). IEEE.

[17] Babitha, B. S., Aher, V. N., Vijayalakshmi, V. J., Jain, R., Indira, G., & Vashisht, N. (2024, June). An Investigation into Content-based Image Retrieval using AI-Assisted Methods. In *2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT)* (pp. 1-6). IEEE.

[18] Liu, Z. Y., Lomovtseva, N., & Korobeynikova, E. (2020). Online learning platforms: Reconstructing modern higher education. *International Journal of Emerging Technologies in Learning (iJET)*, *15*(13), 4-21.

[19] Cheng, Y., Cheng, S., & Huang, Y. (2022). An Internet Articles Retrieval Agent Combined With Dynamic Associative Concept Maps to Implement Online Learning in an Artificial Intelligence Course. *The International Review of Research in Open and Distributed Learning.* https://doi.org/10.19173/irrodl.v22i4.5437.

[20] Yang, Y., Shafi, M., Song, X., & Yang, R. (2018). Preservation of cultural heritage embodied in traditional crafts in the developing countries. A case study of Pakistani handicraft industry. *Sustainability*, *10*(5), 1336.

[21] Garozzo, R., Pino, C., Santagati, C., & Spampinato, C. (2020). Harnessing the Power of Artificial Intelligence for Modelling and Understanding Cultural Heritage Data. *Impact of Industry 4.0 on Architecture and Cultural Heritage.* https://doi.org/10.4018/978-1-7998-1234-0.ch015.

[22] Das, S., Dey, A., Pal, A., & Roy, N. (2015). Applications of artificial intelligence in machine learning: review and prospect. *International Journal of Computer Applications*, *115*(9).

[23] Pipanmekaporn, L., & Kamonsantiroj, S. (2023). Visual Instance Retrieval for Cultural Heritage Artifacts using Feature Pyramid Network. *Human Interaction and Emerging Technologies (IHIET-AI 2023): Artificial Intelligence and Future Applications.* https://doi.org/10.54941/ahfe1002933.

[24] Fontanella, F., Colace, F., Molinara, M., Di Freca, A. S., & Stanco, F. (2020). Pattern recognition and artificial intelligence techniques for cultural heritage. *Pattern Recognition Letters*, *138*, 23-29.

[25] Abgaz, Y., Rocha Souza, R., Methuku, J., Koch, G., & Dorn, A. (2021). A methodology for semantic enrichment of cultural heritage images using artificial intelligence technologies. *Journal of Imaging*, *7*(8), 121.

[26] Pisoni, G., Díaz-Rodríguez, N., Gijlers, H., & Tonolli, L. (2021). Human-centered artificial intelligence for designing accessible cultural heritage. *Applied Sciences*, *11*(2), 870.

[27] Anderson, J., Ziolkowski, M., Kennedy, K., & Apon, A. (2022). Synthetic Image Data for Deep Learning. *ArXiv*, abs/2212.06232. https://doi.org/10.48550/arXiv.2212.06232.

[28] Garozzo, R., Santagati, C., Spampinato, C., & Vecchio, G. (2021). Knowledge-based generative adversarial networks for scene understanding in Cultural Heritage. *Journal of Archaeological Science: Reports*, 35, 102736. https://doi.org/10.1016/j.jasrep.2020.102736.

[29] Cong, L. (2024). A Framework Study on the Application of AIGC Technology in the Digital Reconstruction of Cultural Heritage. *Applied Mathematics and Nonlinear Sciences.* https://doi.org/10.2478/amns-2024-2190.

[30] Mitric, J., Radulovic, I., Popović, T., Šćekić, Z., & Tinaj, S. (2024). AI and Computer Vision in Cultural Heritage Preservation. *2024 28th International Conference on Information Technology (IT)*, 1-4. https://doi.org/10.1109/IT61232.2024.10475738.

[31] Khalid, S., Azad, M., Kim, H., Yoon, Y., Lee, H., Choi, K., & Yang, Y. (2024). A Review on Traditional and Artificial Intelligence-Based Preservation Techniques for Oil Painting Artworks. *Gels*,10. https://doi.org/10.3390/gels10080517.

[32] Iyer, A., & Franklin, M. (2023). AI-Powered Archaeology: Determining the Origin Culture of Various Ancient Artifacts Using Machine Learning. Journal of Student Research. https://doi.org/10.47611/jsrhs.v11i1.2465

[33] Abate, D., Agapiou, A., Toumbas, K., Lampropoulos, A., Petrides, K., Pierdicca, R., Paolanti, M., Di Stefano, F., Felicetti, A., Malinverni, E., & Zingaretti, P. (2023). ARTIFICIAL INTELLIGENCE TO FIGHT ILLICIT TRAFFICKING OF CULTURAL PROPERTY. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. https://doi.org/10.5194/isprs-archives-xlviii-m-2-2023-3-2023.

[34] Ghaith, K. (2024). AI Integration in Cultural Heritage Conservation – Ethical Considerations and the Human Imperative. *International Journal of Emerging and Disruptive Innovation in Education : VISIONARIUM*. https://doi.org/10.62608/2831-3550.1022.

[35] Yang, G. (2020). Adaptive Retrieval Method of Legal Information based on Artificial Intelligence. *2020 IEEE International Conference on Industrial Application of Artificial Intelligence (IAAI)*, 13-17. https://doi.org/10.1109/IAAI51705.2020.9332812.