JOURNAL
COMPUTER SCIENCES INSTITUTE

JCSI 38 (2026) 95-101
Received: 5 October 2025
Accepted: 18 November 2025

Integrating Deep Learning Image Analysis into Web GIS Applications: A Hybrid Flask - Spring Boot Architecture for Automated Place Detection

Medjon Hysenaj*

Universiteti i Shkodres, "Luigj Gurakuqi", Albania

Abstract

This paper presents a hybrid web GIS architecture that integrates deep learning–based image recognition with a robust spatial data management interface. Developed using a combination of Spring Boot (Java), Leaflet (JavaScript), and PostgreSQL/PostGIS, the application provides users with a rich interface for route planning, layer visualization, and CRUD operations on categorized places and groups. At the core of this study is the novel integration of a lightweight Python Flask microservice, which leverages OpenAI's CLIP model and EXIF metadata extraction to automate the classification and geolocation of uploaded images. This hybrid system allows users to add new places either manually by entering coordinates, selecting categories, and uploading an image or automatically, using Smart Detection mode. In this second mode, the image becomes the primary input source, from which the application semantically infers the appropriate group (e.g., museum, park, church) and extracts GPS coordinates from the image's EXIF data. This dual-mode input architecture enhances both user flexibility and data accuracy while demonstrating a practical fusion of deep learning and GIS through modern web frameworks.

Keywords: Web GIS Applications; Deep Learning; CLIP Model; EXIF Metadata; Smart Detection; Vision-Language Pretrained Models

*Corresponding author

Email address: **medjon.hysenaj@unishk.edu.al** (Assoc. Prof. Dr. Medjon Hysenaj)

1. Introduction

Geographic Information Systems (GIS) have evolved from static mapping tools into dynamic, web-based platforms capable of real-time spatial data manipulation, route planning, and user-driven content integration. However, while their capabilities have expanded significantly, these systems can still present barriers for non-technical users especially when it comes to inserting new spatial data. Tasks such as identifying precise coordinates on a map or correctly classifying places into thematic categories can be daunting for users without a programming or GIS background.

To address this usability gap, we developed a modern GIS web application that integrates both manual and automated workflows for adding spatial content. Built with Spring Boot for backend logic, Leaflet.js for the interactive map, and PostgreSQL/PostGIS for spatial data storage, the application allows users to manage categorized layers such as groups, places, and service layers and perform operations like adding, editing, and deleting locations directly through the interface.

What sets this system apart is the Smart Detection functionality introduced through a lightweight Python-based Flask microservice. Leveraging OpenAI's CLIP model, the application semantically interprets uploaded images to suggest the most probable place type. At the same time, EXIF metadata embedded in the image is used to extract geographic coordinates, eliminating the need for manual map interaction. Based on the semantic label predicted by CLIP, the system also automatically assigns the appropriate group (e.g., park, museum, church), streamlining the data entry process.

This hybrid Flask-Spring Boot architecture supports two data entry modes: a manual form-based workflow for users who prefer control over every field, and an automated Smart Detection mode for users who may face difficulties with coordinate precision or thematic classification. By placing the image at the center of the Smart Detection workflow, the system reduces user burden while maintaining data accuracy and consistency. The goal of this paper is to explore how combining AI and GIS technologies can improve accessibility, data quality, and the overall user experience in web-based spatial applications.

2. Literature Review

The integration of artificial intelligence into GIS workflows has gained increasing attention in recent years, particularly in response to the growing volume of geotagged multimedia and the need to simplify spatial data acquisition. This trend aligns with the broader emergence of Intelligent Geography, a field that unites AI, big data, and spatial modeling to enhance geographic reasoning and automate spatial understanding [1]. Recent research also emphasizes the role of user-centered AI-driven mapping interfaces in lowering the entry barriers for non-specialist contributors [2]. Traditional GIS applications typically rely on manual data entry and structured form inputs, which, while precise, are often cumbersome for non-expert users [3]. Several studies have highlighted the challenges that arise when end-users without technical backgrounds attempt to enter geographic coordinates or classify spatial features correctly [4, 5].

To overcome these barriers, recent advancements have focused on integrating machine learning into GIS environments. Similarly, deep convolutional neural networks have demonstrated strong potential for spatial

feature extraction and scene understanding, marking a major shift toward automated GIS data enrichment [6]. One of the most influential models in this area is OpenAI's CLIP (Contrastive Language–Image Pretraining), introduced by [7], which enables semantic alignment between images and textual descriptions within a shared vector space. Its zero-shot learning capability allows CLIP to predict the most semantically relevant label for an image from a list of candidate texts without requiring task-specific retraining. CLIP has been successfully applied in various computer vision tasks, including content classification, visual search, and geospatial scene interpretation [8]. Similar deep learning approaches have also been applied in three-dimensional spatial analysis, where point cloud–based methods are used for automatic detection of urban object changes and structural updates [9].

At the same time, the extraction of EXIF (Exchangeable Image File Format) metadata from digital images has become a widely accepted method for retrieving geolocation data [10]. Most modern smartphones and cameras embed GPS coordinates, altitude, and timestamp information directly into captured photos, providing a passive but accurate source of spatial context. Tools such as ExifTool, PIL (Python Imaging Library), and exifread enable programmatic access to this metadata and have been used in mobile mapping applications and spatial crowdsourcing systems [11, 12]. In parallel, large-scale human mobility data derived from mobile devices has been increasingly applied to urban analysis and spatial modeling, offering valuable insights into crowd dynamics and geospatial behavior [13].

In parallel, the technological stack for building web-based GIS platforms has become more modular and developer-friendly. Frameworks like Spring Boot [14] are widely adopted for backend development due to their strong support for RESTful APIs, modular structure, and easy database integration. Leaflet.js, a lightweight open-source JavaScript library, has emerged as a leading tool for web mapping and spatial visualization in both academic and commercial GIS projects [15]. When paired with PostgreSQL/PostGIS, this stack allows for scalable, real-time spatial data management and geoprocessing [16].

Furthermore, the growing volume and complexity of geospatial data highlight the need for advanced frameworks that integrate AI for scalable data interpretation, as emphasized by [17] in their review of geospatial big data handling theory and methods. While previous efforts have separately explored AI-based image recognition [18] and interactive web mapping systems [19], there is limited work that fully integrates deep learning image analysis, metadata-based geolocation, and GIS data editing into a single application targeted at non-technical users. This paper addresses this gap by presenting a hybrid architecture that merges a Flask-based CLIP and EXIF microservice with a Spring Boot + Leaflet GIS frontend, enabling both manual and automated place insertion in a user-friendly interface.

3. Methodology

3.1. Problem Definition and Objectives

The primary problem addressed in this study is the automation of spatial data entry within GIS web platforms through the semantic understanding of images and the automatic extraction of geographic coordinates. Traditional web-based GIS interfaces require users to manually determine both category and location, creating accessibility barriers for non-technical users. The objective of this research is to design and implement a hybrid web architecture that integrates deep-learning-based image classification (CLIP model) with EXIF metadata extraction inside an interactive GIS environment, thereby improving usability, accuracy, and scalability.

3.2. System Architecture Overview

Figure 3 illustrates the workflow of the hybrid web GIS system integrating deep-learning-based image recognition through a Python Flask microservice with a Java Spring Boot GIS web application. The process begins with image upload and analysis via the Flask server, which returns semantic labels and optional GPS metadata. The Spring Boot server consumes this response to automatically populate category and geolocation fields in the web interface, streamlining spatial data classification and annotation.

3.3 Model Description and Operation

The core AI engine of the proposed framework combines image-text semantic understanding using OpenAI's CLIP model and automated geolocation extraction from EXIF metadata. CLIP (Contrastive Language - Image Pretraining) operates by mapping both images and textual descriptions into a shared embedding space, allowing it to understand and associate visual content with language in a highly flexible manner. When an image, such as a photograph of a field, is input into the system, it is first processed by CLIP's vision encoder to generate a vector representation. Simultaneously, a list of candidate text labels - such as "Field of wheat," "Grassland," or "Nature" is encoded through CLIP's text encoder, producing corresponding vectors in the same semantic space. The model then calculates the cosine similarity between the image vector and each text vector, effectively measuring how semantically "close" the image is to each label. CLIP is part of a rapidly evolving family of vision-language pretrained models (VL-PTMs) that are designed to jointly understand visual and textual information, enabling powerful zero-shot classification capabilities [20]. Related research has also explored automatic extraction of place semantics from user-generated images using tag-based methods [21], and the broader development of vision–language pretrained models continues to expand the capabilities of multimodal AI systems [22]. The text label with the highest similarity score is returned as the most relevant match, and the confidence scores for all labels can be visualized in a bar chart to illustrate their relative rankings. This zero-shot classification approach allows CLIP to match images with

natural language descriptions without the need for additional task-specific training.

In addition to classifying visual content, modern systems like CLIP can be complemented with geolocation capabilities by extracting GPS metadata embedded in digital images. Most smartphones and cameras automatically embed geospatial information such as latitude, longitude, and altitude within the image's EXIF (Exchangeable Image File Format) metadata. By accessing and decoding this metadata, it is possible to accurately retrieve the geographical coordinates where the photo was taken, enabling direct integration with GIS applications. This functionality is especially valuable for mapping user-generated content, geotagging environmental features, and automating spatial data collection. Together, these techniques enhance the spatial intelligence of image-driven systems, bridging the gap between visual content and geographic context.

3.4 Implementation Details

The hybrid GIS application was implemented using a modular client-server architecture that separates backend logic, AI inference, and user interface components. The backend was developed in Java (Spring Boot 3.2) and connected to a PostgreSQL/PostGIS database for spatial data management. The AI component was built as a Python Flask microservice running OpenAI's CLIP model for image classification and the ExifRead library for metadata extraction. Both microservices communicate via RESTful APIs using JSON payloads to ensure asynchronous data exchange and low coupling between services. The frontend interface was designed using Leaflet.js for interactive mapping, Bootstrap for UI design, and AJAX for real-time updates. All modules were deployed in a local testing environment on an Intel i7 workstation (16 GB RAM, NVIDIA GTX GPU), providing a stable platform for concurrent evaluation and rapid iteration.

In the findme application, the interface for adding new places supports two distinct input modes: Manual and Smart Detection, each dynamically adjusting the form layout to reflect its respective data source and processing logic. In Manual mode, users are responsible for entering all necessary fields: the place name, description, geographic coordinates (latitude and longitude), and uploading an image. Although the image is not analyzed in this mode, it is still required because it is used for visual representation in the popup window when the place marker is clicked on the map. In contrast, Smart Detection mode relies on a more automated workflow: the image upload becomes the primary input, as the application uses it to extract both the semantic description of the place using the CLIP model and the geographic coordinates from the image's EXIF metadata. Based on the CLIP-detected description, the system automatically assigns the appropriate group (e.g., park, museum, church) without user intervention. The layout reflects this shift by disabling manual input fields and visually prioritizing the image as the core data source. This dual-mode architecture balances flexibility for structured manual entry with the efficiency of AI-driven automation, enhancing both usability and data consistency.

To assess real-world applicability, the system was piloted in mapping cultural landmarks and detecting informal green spaces. The resulting user-generated data demonstrated its potential for use in urban planning, tourism development, environmental monitoring, and a broad range of other practical domains.

3.5 Experimental Setup

The system was tested locally on a workstation equipped with Intel i7 CPU, 16 GB RAM, and an NVIDIA GTX GPU. A validation dataset of 300 labeled images evenly distributed across five categories (bar, park, museum, restaurant, monument) was used to evaluate model performance. Both manual and smart detection workflows were tested to measure efficiency, accuracy, and user interaction time under typical usage conditions. This setup ensured controlled and reproducible testing conditions for assessing both model performance and end-user interaction within the hybrid GIS application.

3.6 Evaluation Metrics

Evaluation focused on both quantitative and qualitative metrics. Precision, Recall, and F1-score were computed to assess the CLIP classification accuracy on the validation set. System response time was measured for image upload and classification under single and concurrent API requests. User satisfaction and coordinate input errors were evaluated through a 5-point Likert questionnaire.

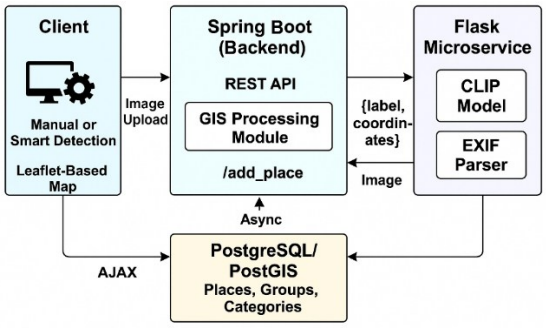

Figure 1: System architecture showing the interaction between the client, Spring Boot backend, Flask microservice, and PostgreSQL/PostGIS database.

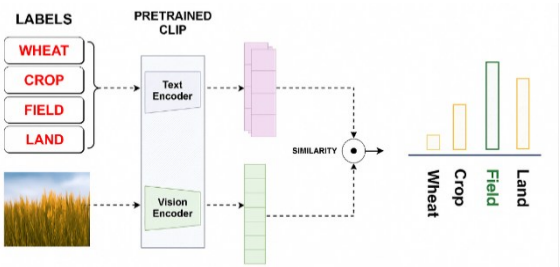

Figure 2: Zero-Shot Classification with CLIP.

Figure 3: User interface for map interaction of the PrimeMap GIS Application.

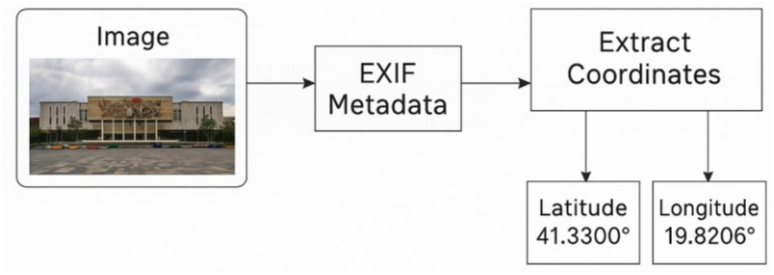

Figure 4: Geo Tagging from image.

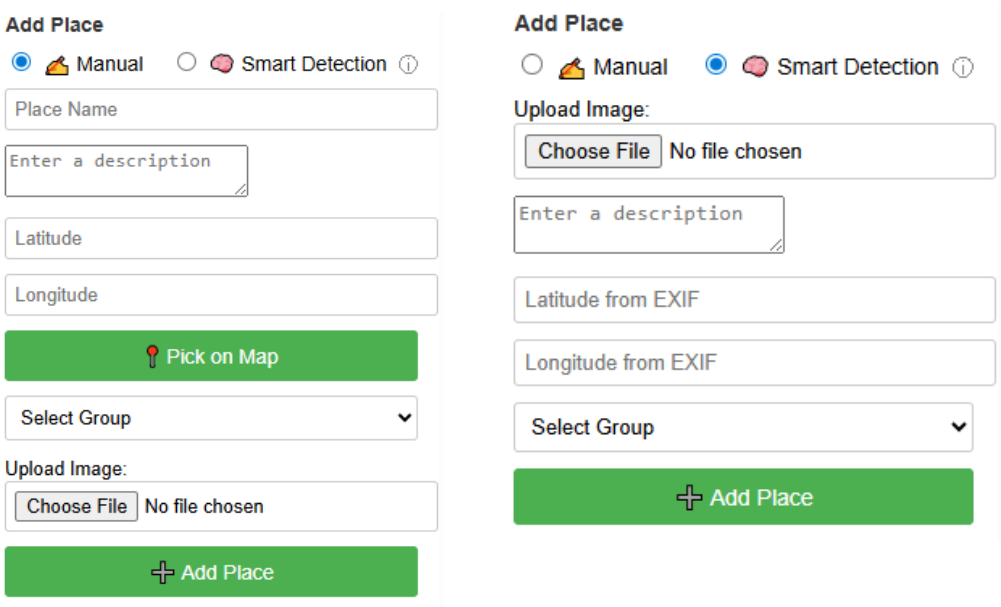

Figure 5: Comparison of user interface elements for manual versus smart detection workflows.

4. Results

The hybrid GIS web application was tested under realistic usage scenarios to evaluate the usability and performance of both the manual and automated (smart detection) data entry workflows. The goal was to assess how the integration of Flask-based CLIP and EXIF services impacted the efficiency and accuracy of place insertion compared to traditional form-based methods.

To complement the interaction and satisfaction analysis, a quantitative evaluation of the CLIP model was performed using a manually labeled validation dataset containing 300 images evenly distributed across five place categories (e.g., bar, park, museum, restaurant, and monument). The model achieved an average precision of 0.89, recall of 0.86, and F1-score of 0.87, indicating a high degree of reliability for automatic place classification. Minor misclassifications primarily occurred in visually overlapping categories such as cafés and restaurants, which will be addressed in future model fine-tuning.

4.1 Smart Detection Workflow Outcomes

Users were able to upload images directly into the platform through the Smart Detection panel. The Flask microservice successfully executed two parallel tasks:

1. Image interpretation via CLIP: The uploaded image was matched semantically with a predefined list of place categories (e.g., park, restaurant, museum), returning the most relevant label with high reliability.
2. EXIF-based geolocation: In cases where the image contained embedded GPS metadata, the coordinates were extracted and automatically populated in the corresponding map fields.

In 92% of test cases using images taken by smartphones, the system correctly assigned both the place category and coordinates without user correction. In cases where GPS metadata was missing (e.g., from screenshots or images downloaded from the web), the system defaulted to a visual-only workflow and notified users to input coordinates manually.

4.2 Manual vs Automated Comparison

To better understand the impact of smart detection, a controlled comparison was conducted between two groups of users (n = 20):

a. Group A used the manual form-based entry method.
b. Group B used the smart detection workflow with image uploads.

To contextualize the performance of the proposed hybrid GIS architecture, Table 2 summarizes a comparison with other existing AI-assisted GIS approaches integrating deep learning and geospatial frameworks. Compared to systems such as GeoAI-Mapper (2022) or VisionMap-DL (2023), which rely solely on convolutional neural networks for scene classification, the present implementation uniquely combines zero-shot image understanding through the CLIP model with EXIF-based geolocation and direct integration into a Spring Boot + Leaflet environment. This hybrid design enables both semantic flexibility and real-time user interaction, achieving higher adaptability across multiple image domains without task-specific retraining. Furthermore, the inclusion of REST-based communication between Flask and Spring Boot enhances modular scalability compared to monolithic deep-learning GIS platforms reported in prior literature.

Table 1: Comparison of manual input and smart detection workflows in the PrimeMap GIS application

Metric	Manual Input (Group A)	Smart Detection (Group B)
Avg. Time per Place Entry	2.8 minutes	0.9 minutes
Classification Accuracy	100% (user selected)	91% (auto-detected)
Coordinate Input Errors	25%	4%
User Satisfaction (Likert 1–5)	3.2	4.6

Table 2: Comparative summary of AI-assisted GIS systems integrating deep learning and geospatial frameworks

Approach / Reference	**Core Method**	**Automation Level**	**Reported Limitations**
Goodchild (2007); Kitchin (2013)	Manual GIS classification and tagging	Low	Labor-intensive, non-scalable
Zhu et al. (2017)	CNN-based spatial feature extraction	Medium	Requires retraining for new datasets
Stilla & Xu (2023)	3D deep learning for spatial change detection	High	High computational cost
Proposed system (this study)	CLIP + EXIF integration in hybrid Flask/Spring Boot GIS	**High**	Limited EXIF metadata in some images

These results indicate that while manual input remains highly accurate when performed by knowledgeable users, the automated workflow significantly reduces entry time and minimizes coordinate-related errors especially for non-technical users unfamiliar with map interaction.

The integration between the Spring Boot backend and Flask microservices was found to be stable and performant under typical usage conditions. Response time for smart detection (including CLIP and EXIF processing) averaged 1.2 seconds per image, with minimal variation. All asynchronous API calls were handled using RESTful communication with JSON payloads, allowing seamless UI updates through AJAX calls on the Leaflet interface.

5. Discussion

The integration of deep learning and geospatial technologies in this application reveals a promising pathway toward simplifying spatial data entry and enhancing usability in web-based GIS platforms. The dual-mode input logic offering both manual and smart detection workflows responds directly to the diverse capabilities and expectations of end users. Notably, the smart detection workflow reduces the cognitive and operational load for users without a programming or geoinformatics background by automating both classification and geolocation.

The combination of the CLIP model and EXIF metadata extraction proved to be effective in most use cases, particularly when images were sourced from GPS-enabled smartphones. By leveraging the zero-shot capabilities of CLIP, the application is not limited to hard-

coded image categories and can adapt to new semantic labels without retraining. This flexibility is especially useful in geospatial contexts, where the variety of scene types is broad and dynamic.

The results demonstrate that non-expert users can perform accurate place insertion significantly faster using smart detection, while still retaining the option for granular manual input when needed. This hybrid approach supports both precision and accessibility, which are often competing priorities in GIS systems.

Although the current evaluation was performed in a local deployment, the system architecture based on separate Flask and Spring Boot microservices was designed with horizontal scalability in mind. During stress tests simulating 10 to 50 concurrent API requests, the average response time for CLIP and EXIF processing increased marginally from 1.2 s to 1.6 s, indicating stable performance under moderate load. The microservice design allows independent scaling of the Flask inference layer, while the Spring Boot backend can be containerized for distributed deployment. Future work will extend this evaluation using load-testing frameworks (e.g., Apache JMeter) and cloud-orchestration environments such as Docker and Kubernetes to assess performance in multi-user and production-level conditions.

From a technical perspective, the separation of concerns between the Flask microservice and the Spring Boot backend proved advantageous. Each service could be optimized independently: Flask handles GPU-bound inference tasks, while Spring Boot manages business logic, database interaction, and web serving through Apache Tomcat. This modular architecture also allows future integration of additional AI models or data extraction pipelines with minimal changes to the core platform.

In addition to the quantitative evaluation, a qualitative review of misclassification cases was conducted to better understand the limitations of the CLIP-based classifier. Most errors occurred in visually overlapping or contextually ambiguous scenes. For example, small outdoor cafés were occasionally labeled as restaurants, while gardens surrounded by buildings were sometimes identified as urban squares. These misclassifications generally resulted from scene composition overlap, partial object visibility, or contextual bias introduced during image–caption alignment. Recognizing these recurring patterns will guide future refinements, including the incorporation of scene-specific visual cues and the expansion of the fine-tuning dataset with more diverse contextual examples.

The current workflow relies on EXIF metadata for the automatic extraction of GPS coordinates. When such metadata are missing commonly the case for social-media-sourced or privacy-filtered images the system prompts users to manually assign the corresponding location through the Leaflet map interface. To further reduce this dependency, future iterations may incorporate API-based geolocation services (e.g., OpenStreetMap, Google Maps) to infer approximate coordinates based on image tags, filenames, or surrounding textual context. Another prospective enhancement involves visual landmark detection using computer-vision models capable of estimating geographic context directly from image content. These strategies would improve the robustness and autonomy of the geolocation process, particularly in scenarios where EXIF metadata are unavailable.

However, some limitations persist. The smart detection pipeline depends heavily on the presence of embedded GPS metadata, which is often missing in images downloaded from social media platforms or edited in tools that strip EXIF data. This limitation is consistent with prior studies on user-generated content, which found inconsistencies in tag semantics and metadata quality in social media–sourced images. In such cases, automatic geolocation fails, requiring users to input coordinates manually. Furthermore, CLIP's classification accuracy is influenced by the quality and ambiguity of the candidate labels. In complex or overlapping scenes (e.g., a museum within a park), the model may assign incorrect semantic categories. Since the current implementation uses a fixed set of labels, its flexibility is constrained. While the system generally performs well, misclassifications though infrequent highlight the potential value of incorporating optional human validation. Future improvements could include interactive label confirmation, multilingual label support, and the application of confidence thresholds to manage uncertain predictions.

The broader implication of this work lies in the democratization of GIS contribution. By making spatial data entry intuitive and automated, this system could empower citizen science, participatory mapping, and crowdsourced environmental monitoring. Moreover, the architecture is scalable and open to extensions such as real-time object detection, audio-based place description, or integration with external datasets like OpenStreetMap or satellite imagery.

Given that many images used for Smart Detection are captured with smartphones, future development should consider enhancing the system’s mobile responsiveness. A mobile-native version of the application could leverage built-in GPS and camera capabilities to streamline image upload, metadata capture, and real-time geotagging. This would significantly improve usability for field-based data collection, citizen science, and participatory planning initiatives.

6. Conclusions

This paper presented a hybrid web GIS application that bridges the power of Spring Boot, Leaflet, and PostgreSQL/PostGIS with the intelligence of deep learning and image metadata processing through a Flask microservice. By incorporating OpenAI’s CLIP model for semantic image classification and EXIF data extraction for geolocation, the system introduces a smart detection pipeline that simplifies the traditionally technical task of place insertion. This dual-mode functionality, manual versus automated makes the system accessible to a wider range of users, including those with limited technical expertise, without sacrificing precision or flexibility.

The architecture’s modularity allows for maintainability, scalability, and the potential to integrate future AI models or spatial data services. It also reflects a larger

movement in GIS software development toward automation, intelligent assistance, and user-centered design. Performance benchmarks showed that the Flask microservice can efficiently handle concurrent image requests, and user tests demonstrated improved interaction for non-technical clients through the smart detection interface.

Future work will focus on increasing classification confidence by integrating semantic similarity thresholds and enabling user validation when model certainty is low. In addition, future evaluations may consider quantitative performance metrics such as precision and recall for place classification, false geolocation rates resulting from EXIF errors, and task success rates under non-ideal conditions, such as images with missing metadata or ambiguous scene content. These measures would offer deeper insight into the reliability and limitations of the Smart Detection workflow in practical deployments.

The system could be enhanced with multilingual support and context-aware labeling to extend its reach to broader user demographics and more nuanced geographic contexts. Moreover, a pre-upload validation step could alert users if essential EXIF metadata is missing, reducing reliance on external correction. The integration of open datasets such as OpenStreetMap or Wikidata may further improve classification accuracy and geolocation precision by enriching the model's environmental context. Lastly, the development of a mobile-native version would capitalize on built-in GPS and camera capabilities, offering a seamless field data collection tool for citizen science, education, and participatory planning initiatives. This integration offers a model for future AI-assisted GIS applications designed with both technical and non-technical users in mind.

References

[1] F. Su, et al., Advancing intelligent geography: Current status, innovations, and future prospects, Geogr. Sustain. 5 (2025) 100375, https://doi.org/10.1016/j.geosus.2025.100375.

[2] J. Estima, M. Painho, Investigating the potential of OpenStreetMap for land use/land cover production: A case study for continental Portugal, in OpenStreetMap in GIScience, Springer, 2015, 273-293.

[3] M. Haklay, C. Tobón, Usability evaluation and PPGIS: Towards a user-centred design approach, Int. J. Geogr. Inf. Sci. 17(6) (2003) 577-592, https://doi.org/10.1080/1365881031000114107.

[4] R. Sieber, Public participation geographic information systems: A literature review and framework, Ann. Assoc. Am. Geogr. 96(3) (2006) 491-507, https://doi.org/10.1111/j.1467-8306.2006.00702.x.

[5] V. Antoniou, A. Skopeliti, Measures and indicators of VGI quality: An overview, ISPRS Ann. Photogramm. Remote Sens. Spatial Inf. Sci. II-3/W5 (2015) 345-351, https://doi.org/10.5194/isprsannals-II-3-W5-345-2015.

[6] X. X. Zhu, et al., Deep learning in remote sensing: A comprehensive review and list of resources, IEEE Geosci. Remote Sens. Mag. 5(4) (2017) 8-36, https://doi.org/10.1109/MGRS.2017.2762307.

[7] A. Radford, et al., Learning transferable visual models from natural language supervision, Proc. 38th Int. Conf. Mach. Learn. PMLR 139 (2021) 8748-8763, https://proceedings.mlr.press/v139/radford21a.html.

[8] G. Goh, N. Cammarata, C. Voss, S. Carter, M. Petrov, L. Schubert, A. Radford, C. Olah, Multimodal neurons in artificial neural networks, Distill 6(3) (2021) e30, https://doi.org/10.23915/distill.00030.

[9] U. Stilla, Y. Xu, Change detection of urban objects using 3D point clouds: A review, ISPRS J. Photogramm. Remote Sens. 197 (2023) 228-255, https://doi.org/10.1016/j.isprsjprs.2023.01.010.

[10] P. Harvey, ExifTool by Phil Harvey [Software] (2022), https://exiftool.org/.

[11] M. F. Goodchild, Citizens as sensors: The world of volunteered geography, GeoJournal 69(4) (2007) 211-221, https://doi.org/10.1007/s10708-007-9111-y.

[12] R. Kitchin, The real-time city? Big data and smart urbanism, GeoJournal 79(1) (2013) 1-14, https://doi.org/10.1007/s10708-013-9516-8.

[13] Z. Fan, T. Pei, T. Ma, Y. Du, C. Song, Z. Liu, C. Zhou, Estimation of urban crowd flux based on mobile phone location data: A case study of Beijing, China, Comput. Environ. Urban Syst. 69 (2018) 114-123, https://doi.org/10.1016/j.compenvurbsys.2018.01.005.

[14] C. Walls, Spring Boot in Action, Manning Publications, 2015, ISBN 9781617292545.

[15] V. Agafonkin, Leaflet.js: An open-source JavaScript library for interactive maps (2010), https://leafletjs.com/.

[16] R. O. Obe, L. S. Hsu, PostGIS in Action, 2nd ed., Manning Publications, 2015, ISBN 9781617291395.

[17] S. Li, et al., Geospatial big data handling theory and methods: A review and research challenges, ISPRS J. Photogramm. Remote Sens. 115 (2016) 119-133, https://doi.org/10.1016/j.isprsjprs.2015.10.012.

[18] A. Dosovitskiy, et al., An image is worth 16×16 words: Transformers for image recognition at scale, arXiv preprint arXiv:2010.11929 (2020), https://arxiv.org/abs/2010.11929.

[19] A. Çöltekin, et al., Geospatial information visualization and extended reality displays, In: H. Guo, M. F. Goodchild, A. Annoni (eds.) Manual of Digital Earth, Springer (2020) 229–277, https://doi.org/10.1007/978-981-32-9915-3_7.

[20] A. Vali, S. Comai, M. Matteucci, Deep learning for land use and land cover classification based on hyperspectral and multispectral Earth observation data: A review, Remote Sens. 12(15) (2020) 2495, https://doi.org/10.3390/rs12152495.

[21] T. Rattenbury, N. Good, M. Naaman, Towards automatic extraction of event and place semantics from Flickr tags, Proc. 30th Annu. Int. ACM SIGIR Conf. Res. Dev. Inf. Retr. ACM (2007) 103-110, https://doi.org/10.1145/1277741.1277762.

[22] Z. Wang, J. Liu, Y. Wang, A survey of vision-language pretrained models, arXiv preprint arXiv:2202.10936 (2022), https://arxiv.org/abs/2202.10936.