

Article

An Agentic AI System for Roof Design Compliance Using Computer Vision, Retrieval-Augmented Generation and Large Language Models

Nawari O. Nawari  and Oluwatoyin O. Lawal * 

School of Architecture, College of Design, Construction and Planning, University of Florida, 1480 Inner Road, Gainesville, FL 32611-5702, USA; nnawari@ufl.edu

* Correspondence: o.lawal@ufl.edu; Tel.: +1-813-334-6990

Abstract

Designers, engineers, and building officials face increasing pressure to accelerate and improve the accuracy of design review for buildings and infrastructure. Roof assemblies and rooftop structures are particularly challenging due to the complexity and fragmentation of regulatory requirements, especially in jurisdictions such as Florida, where compliance must be verified across both the residential and commercial volumes of the Florida Building Code (FBC). The resulting review process is technically demanding and time-intensive, imposing significant cognitive and operational burdens on practitioners and under-resourced public agencies. To address these challenges, this study proposes and evaluates an agentic artificial intelligence (AI) framework for automated code compliance checking of roof assemblies and rooftop structures. The framework employs a multi-agent architecture in which specialized AI agents collaboratively interpret regulatory provisions and evaluate roof design parameters across four core modules: data preprocessing and code ingestion, rule-based and semantic analysis, results visualization, and iterative validation. YOLO11m-seg and Mask R-CNN were used for element detection and segmentation, and the system was developed using 150 design projects, including roof plans, section details, and specifications. Four large language models from two families (Mistral and GPT) were comparatively evaluated on standardized compliance tasks. The framework was then tested on a held-out portfolio of 15 distinct roof-design projects comprising 60 code-compliance decisions derived from the FBC 2023, with performance measured by precision, recall, F1-score, and accuracy. GPT-5.4 achieved the highest overall performance (F1 = 0.97; accuracy = 97%). Because the reasoning and vision components were evaluated separately rather than as an integrated end-to-end pipeline, and the scope was limited to one jurisdiction and two drawing types, broader code coverage and production-setting validation are needed before claims of generality. Nonetheless, the results suggest that agentic AI can meaningfully support compliance review and reduce reviewer burden in roof-design permitting.



Academic Editor: Jaewook Jeong

Received: 8 May 2026

Revised: 13 June 2026

Accepted: 26 June 2026

Published: 2 July 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

Keywords: agentic system; AI; roof design; roof validator

1. Introduction

The regulatory review of development projects and building permits represents one of the most consequential—and chronically inefficient—bottlenecks in the built environment development process [1]. Municipal and county building officials are under mounting institutional pressure to accelerate the review and approval of development applications, driven

by increasing urbanization, housing demand, and the growing complexity of regulatory frameworks governing the built environment [2]. Yet the task of verifying whether a proposed development or structure satisfies the full spectrum of applicable land development regulations, zoning ordinances, and building code requirements remains an extraordinarily labor-intensive undertaking [2]. In jurisdictions already contending with chronic staffing shortages, this compliance verification process can extend over periods of several weeks to several months, imposing significant delays on development timelines, increasing project costs, and undermining the efficiency of the public permitting system [2,3].

In response to these operational challenges, most local building departments have undergone a structural transition away from paper-based plan submission systems toward digitally integrated permit application platforms. While this transition has yielded improvements in document management, accessibility, and workflow organization, empirical evidence suggests that the digitization of submission processes has not produced a commensurate reduction in overall permit processing times [2,3]. The fundamental bottleneck persists not in the medium of document submission but in the cognitive and procedural demands of the review process itself—specifically, the time-intensive task of having qualified planners and building officials manually evaluate submitted plans against the full hierarchy of applicable building codes, zoning regulations, and land development standards [2,3]. The complexity of this task is compounded by the fragmented and multi-jurisdictional nature of regulatory frameworks, which frequently require reviewers to cross-reference multiple code documents, interpret ambiguous provisions, and exercise professional judgment under time pressure [2].

Architectural and Engineering Design Review (AEDR) is, by its nature, an iterative and multi-stage process that demands sustained human expertise at every phase of the design workflow (see Figure 1) [2]. Reviews are conducted both internally, within design and engineering firms, and externally by regulatory authorities, each involving the systematic evaluation of design decisions against a complex matrix of performance, safety, and compliance requirements [2]. Despite the widespread adoption of Building Information Modeling (BIM) and computer-aided design (CAD) technologies within the design professions, the review-and-compliance-checking phase of this workflow has remained largely resistant to automation [3,4]. In current practice, the predominant method of conducting design reviews relies on manual annotation of PDF drawings, a process that is not only time-consuming but inherently limited in its capacity for systematic, scalable, and reproducible compliance verification [5]. Building permit applications submitted to regulatory authorities continue to rely on two-dimensional PDF drawing sets, which require manual interpretation by reviewers and often necessitate multiple rounds of revision and resubmission before a permit can be issued. The cumulative effect of these inefficiencies is a permitting and review ecosystem characterized by significant delays, inconsistent outcomes, and a disproportionate burden on both applicants and public agencies [2,3].

The convergence of these challenges, regulatory complexity, workforce constraints, digital adoption without process transformation, and the persistence of manual review methodologies creates a compelling imperative for the development of intelligent, automated tools capable of augmenting and accelerating the design review and code compliance checking process. Recent advances in artificial intelligence (AI), natural language processing (NLP), and large language models (LLMs) offer promising avenues for addressing this imperative, enabling the automated interpretation of regulatory text, the semantic analysis of design submissions, and the generation of compliance determinations that would otherwise require hours of manual expert review [5–7]. The present study responds to this opportunity by proposing and evaluating an agentic AI framework for the automated checking of regulatory compliance requirements, with a particular focus on roof assemblies

and rooftop structures as a representative and technically complex domain of building code application.

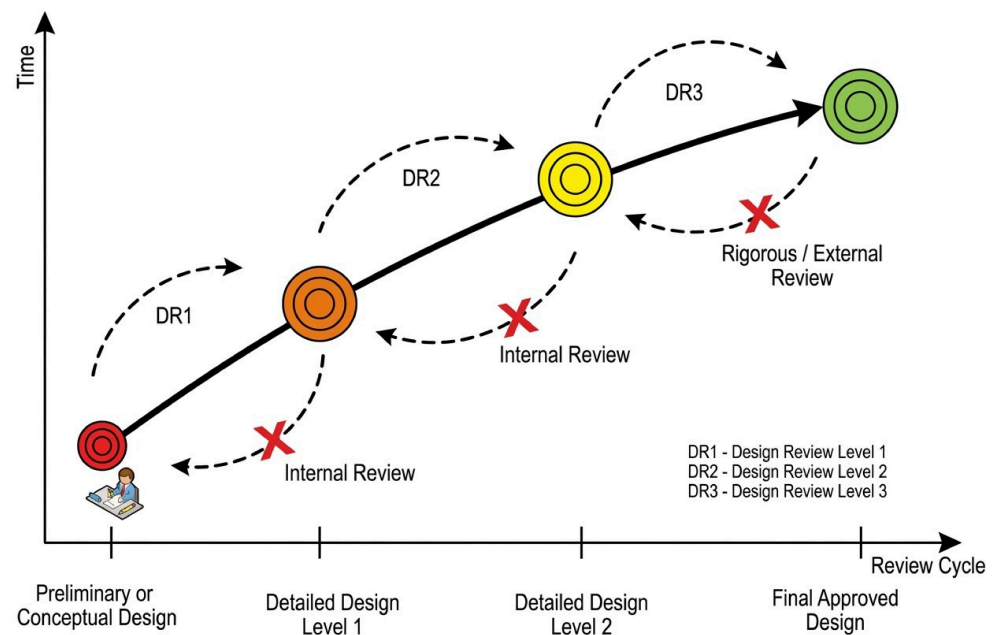


Figure 1. Typical Architectural and Engineering Design Review (AEDR) Process. (Note: ChatGPT 5.5 was used to generate the image in Figure 1).

The predominant method by which building design submissions are currently evaluated by regulatory authorities involves the manual review of two-dimensional drawing sets submitted in Portable Document Format (PDF). This approach, while widely adopted due to its accessibility and compatibility with existing digital submission platforms, is fundamentally limited in its capacity to support systematic, scalable, and time-efficient compliance verification [5]. The manual interpretation of PDF drawings by qualified reviewers introduces significant latency into the permitting process, contributing to the protracted delays that characterize contemporary building permit workflows and imposing substantial costs on both applicants and public agencies [2,3]. However, these design reviews are not bi-directional. There is no internal feedback mechanism that interconnects DR2 with DR1 and DR3 with DR1 or 2 as shown in Figure 1 above.

A critical pathway to addressing these limitations lies in transforming static PDF drawing files into semantically rich, machine-readable Computer-Aided Design (CAD) objects. This conversion process is not merely a technical reformatting exercise; it represents a fundamental shift in the nature of the design data available for automated analysis. CAD models, by encoding geometric, spatial, and material information in structured, query-able formats, serve as high-value data sources that support automated compliance checking, parametric analysis, and data-driven decision-making with the precision and consistency that manual PDF review cannot replicate [3,4]. The translation of design intent from static documentation into computable representations is therefore a prerequisite for developing intelligent, automated design review systems capable of operating at scale [3].

Artificial intelligence (AI) has demonstrated considerable promise as an enabling technology for engineering and architectural design applications, offering capabilities that extend from pattern recognition and object detection to natural language understanding and generative reasoning [5–7]. The integration of AI methodologies into the building design review workflow presents a compelling opportunity to automate the extraction, interpretation, and evaluation of design information—tasks that currently demand extensive

human expertise and consume disproportionate amounts of reviewer time and institutional resources [5,6].

In response to this opportunity, the present research proposes and evaluates an agentic AI framework specifically designed to extract structured data from civil and construction documents and apply algorithmic reasoning to verify compliance with applicable building codes. The framework leverages two complementary families of machine-learning technologies. First, state-of-the-art computer vision models, specifically, YOLOv11 (You Only Look Once, version 11) [8], a high-performance single-stage convolutional neural network optimized for real-time object detection, and Mask R-CNN (Mask Region-based Convolutional Neural Network) [9], a two-stage instance segmentation architecture capable of delineating object boundaries with high spatial precision, are employed to extract roof design-relevant elements and annotations from architectural and engineering PDF drawings (see Figure 2). Second, a Retrieval-Augmented Generation (RAG) model is integrated to enable the semantic interrogation of extracted design data against the provisions of applicable building codes, combining the contextual retrieval capabilities of vector-based knowledge bases with the generative reasoning capacity of large language models (LLMs). These two components operate within a multi-agent AI architecture, in which specialized autonomous agents collaborate to perform discrete subtasks—document parsing, element extraction, code retrieval, compliance evaluation, and report generation—in a coordinated and iterative workflow. The integrated system is conceived as a co-roof designer tool: an intelligent collaborative assistant capable of supporting designers, engineers, and building officials throughout the roof design and review process. Additional feature extraction and multi-scale representation techniques such as Fully Convolutional Networks (FCNs) [10] and Feature Pyramid Networks (FPNs) [11] further enhance the system’s ability to detect and interpret complex spatial relationships within design documents.

Research Impact and Educational Significance

The anticipated impact of the proposed framework operates at two complementary levels: professional practice and academic education.

At the level of professional practice, the deployment of an agentic AI framework for automated roof design compliance checking has the potential to substantially reduce review time and operational costs for building departments, design firms, and construction organizations [2,3,5]. By automating labor-intensive tasks such as document parsing, element identification, and code cross-referencing, the system enables qualified reviewers to focus on complex interpretive judgments, thereby improving both efficiency and accuracy. The systematic nature of AI-assisted compliance checking also promotes greater consistency and reproducibility in compliance determinations, reducing variability arising from reviewer fatigue, staffing turnover, and inconsistent code interpretation practices [5].

At the level of academic education, the framework provides a practical demonstration of how AI can augment roof design, construction processes, and regulatory review within Architecture, Engineering, and Construction (AEC) programs [6,7]. Designed as a pedagogically accessible co-roof designer tool, the framework enables students at both undergraduate and graduate levels to engage with compliance-checking workflows through interactive, scenario-based learning environments. The research outcomes can inform Integrated Building Technology courses, architectural design studios, and structural engineering curricula by strengthening students’ understanding of roofing systems, construction methods, and regulatory requirements while simultaneously developing digital literacy and applied AI competencies increasingly demanded by the profession.

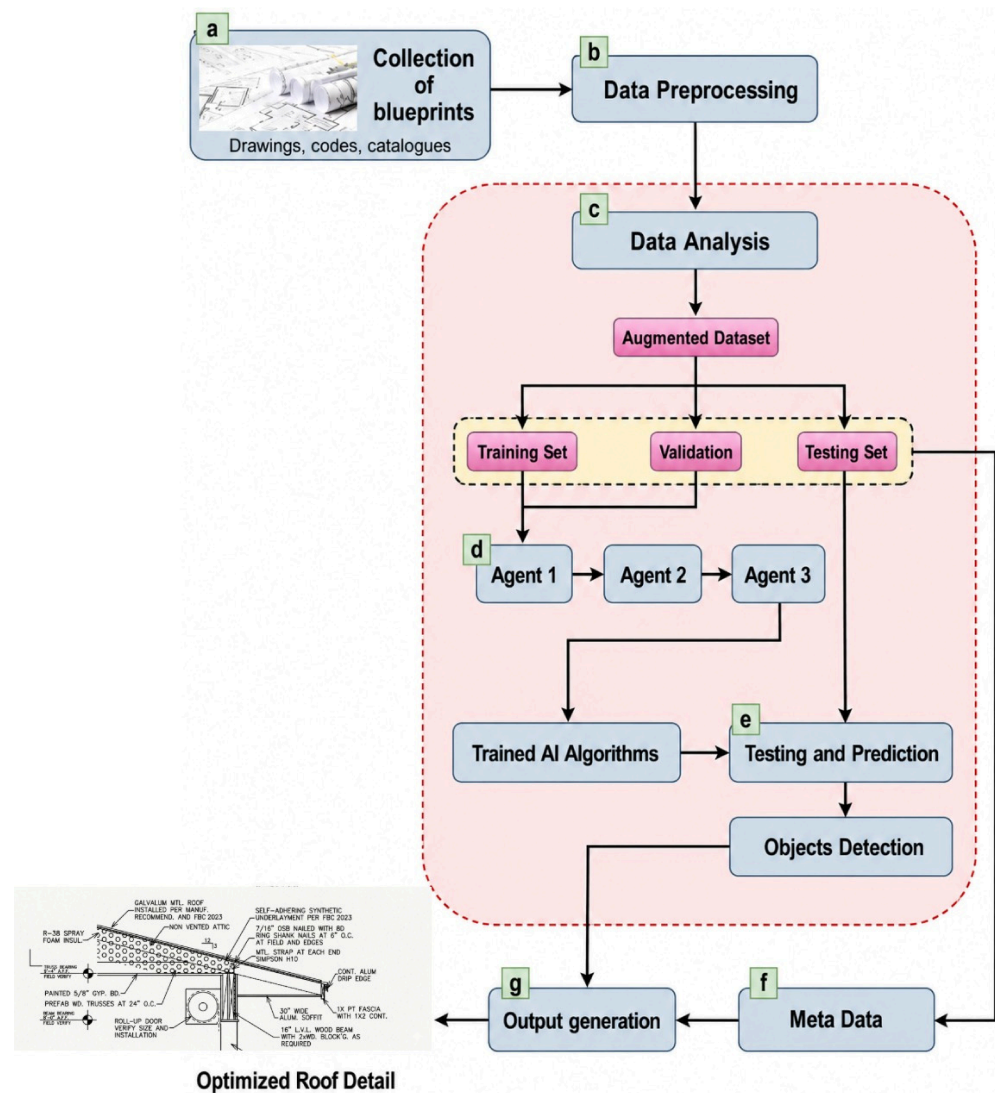


Figure 2. Outline of the research approach.

More broadly, the framework supports interdisciplinary learning by bringing together computer vision, natural language processing, building science, and regulatory theory within a unified educational platform. By situating AI-assisted design review at the intersection of technology, design, and construction, the research encourages cross-disciplinary thinking and reflects the evolving role of AI as a collaborative design partner. In doing so, it contributes not only to the technical literature on automated compliance checking but also to ongoing discussions regarding technology integration in AEC education and the transition from manual, document-centric regulatory review toward intelligent, data-driven decision-making.

The remainder of this paper is structured as follows. Section 2 presents a critical review of existing AI tools, computational workflows, and machine learning methodologies for roof design, building analysis, and automated code-compliance checking, identifying key research gaps and motivating the study’s research questions. Section 3 defines the research goals and objectives. Section 4 describes the research methodology, including data collection, model selection, and evaluation procedures. Section 5 presents the architecture of the proposed agentic AI framework and the interaction of its constituent agents. Section 6 outlines the implementation schema, including the technical infrastructure, software environment, and integration protocols employed. Section 7 presents the evaluation results and comparative performance of the assessed LLMs, followed by a discussion

of the framework's capabilities, limitations, and broader implications. The final section summarizes the study's contributions and outlines directions for future research.

This paper presents a novel contribution to AI-assisted architectural and engineering design review through the development and validation of an agentic AI framework for automated roof design compliance checking. Unlike prior approaches that address isolated components of the compliance verification workflow, the proposed framework integrates document parsing, object detection, semantic code retrieval, and multi-agent reasoning into a unified end-to-end system. The framework is evaluated using real-world scenarios derived from Florida building code provisions, providing empirical evidence of its practical utility and performance.

Although the present study focuses on roof assemblies and rooftop structures, the underlying architecture was designed with extensibility as a core principle. The modular, agent-based structure can be adapted to additional compliance domains across the AEC disciplines, including structural systems, accessibility requirements, fire protection, mechanical and electrical systems, and site development regulations. Consequently, the proposed framework represents not only a specialized tool for roof design review but also a foundation for broader AI-enabled transformation of building code compliance checking within architectural, engineering, and construction practice.

2. Literature Review

Building codes mandate a minimum quality level for erected buildings that are acceptable to ensure public safety [12]. Automated code compliance checking (ACC) is a critical yet challenging component of building design and regulatory approval processes. It is performed traditionally by architects or other building professionals who interpret textual regulations from the graphic and textual submittals and evaluate design conformance. This process is inherently time-consuming, cognitively demanding, and prone to inconsistencies due to the need for deep domain expertise and the subjective interpretation of regulatory language [13,14]. Advancements in Automated Compliance Checking have introduced LLMs to enhance automation. An example is through prompt engineering, leveraging GPT-3 and GPT-3.5 models and evaluating their performance in processing building design specifications through a series of experiments [15]. Their research demonstrated the potential of LLMs in handling and understanding complex text tasks, particularly in dealing with large-scale, complex text contexts. More complex texts with nested clauses and conditional statements may require additional strategies for effective processing. In a review of the current state, challenges and opportunities facing Automated Code Checking (ACC), technological constraints and integration difficulties were identified as the main challenges facing ACC implementation, with potential opportunities for ACC including integration with technologies, expanding application domains, collaborations, and standardizations [16]. Therefore, many studies in the AEC domain have since focused on improving the efficiency and reliability of compliance checking. BIM has become the leading design and construction documentation platform in the AEC industry since the mid-2000s [17]. AI and Machine Learning (ML) have proven to be efficient in extracting useful data sources and make good predictions about when and where to provide quality control [18]. BIM, however, has a unique ability to replicate physical scenarios, and it embeds information with real physical and non-physical properties, making it an effective tool for collaboration.

The majority of work in BIM-based ACC has focused on technological aspects, such as code articles better expressed in a computational format, the accurate data representation and standardization of BIM models, more flexible and practical rule-checking algorithms, and the integration of BIM with other emerging information technologies [19]. These

early rule-based efforts, closely tied to the adoption of BIM, were made up of systems that relied on manually translating regulatory clauses into machine-readable rules and applying them to BIM models through query-based or logic-based engines [20]. While effective for well-defined and quantitative requirements, these rule-based approaches face significant limitations when dealing with ambiguous, context-dependent, or cross-referenced provisions commonly found in building codes [21]. The discrepancies between structured, ontology-based BIM data and unstructured, narrative-based regulatory documents makes the automation process more cumbersome.

Semantic approaches have been deployed to address the disparity challenges between structured and unstructured data using ontology-based models and knowledge graphs. Ontologies enable formal representation of regulatory concepts, building elements, and their relationships, thereby facilitating semantic alignment between BIM models and regulatory requirements [22]. A knowledge graph represents one of many innovative ways to represent and organize vast amounts of information and knowledge in a structured format, adopted in a wide range of applications from information retrieval, integration, and management to recommendations and answering of questions [23]. Knowledge graphs extend ontology modeling capabilities by structuring regulatory knowledge from building codes into interconnected entities and relationships, enabling increased reasoning and mapping flexibility between design data and code provisions.

Recent studies have demonstrated the deployment of specialized agents to improve decision quality, transparency and task execution. Multi-agent systems enable autonomous individual agents to focus on specific subtasks which have a collaborative impact on a common objective [24]. Natural Language Processing (NLP) has become a critical facilitator for extracting and interpreting regulatory information, in parallel with semantic modeling. These techniques have been applied to classify code provisions, extract entities and conditions, and translate textual requirements into structured representations suitable for automated checking [13]. NLP and BIM have been integrated to support automated reasoning over spatial and relational requirements, such as distances, clearances, and adjacency conditions [25].

BIM-based projects contain a large quantum of information and repository data, which presents an opportunity for analysis and extraction of insights throughout the project lifecycle, and ML has emerged as an effective approach for such extraction [26]. Ghimire et al. (2023) recommended a conceptual Generative AI implementation framework capable of fine-tuning generative LLMs by collecting BIM data and other cloud-based data repositories [27]. Nabavi et al. (2023) proposed a framework that uses the support vector machine (SVM) algorithm to facilitate information retrieval from BIM models to determine users' likely questions while also relying on NLP for syntactic analysis [28]. In a survey of existing research in LLM-based autonomous agents, Wang et al. (2024) evaluated the capabilities of autonomous agents which enable execution of complex multi-stage workflows and form a basis for an intelligent design-support system [29]. These agents represent a shift from passive conversational systems towards goal-driven intelligent systems [30].

Retrieval-Augmented Generation (RAG) has also emerged as an efficient mechanism for reducing hallucinations in LLMs by grounding responses in context-specific knowledge domains to improve factual consistency [31]. RAG's integration with LLMs enables dynamic access to authoritative knowledge repositories to improve reliability and domain suitability in situations where quick reference to regulatory information is required [32]. The texts generated through this approach enhance model performance by combining parametric knowledge with retrieved contextual information rather than relying solely on information encoded with pretrained model weights [33]. External knowledge retrieval

substantially improves domain-specific reasoning accuracy where domain knowledge evolves over time and authoritative references must be pivotal to the decision-making process [34].

Conventional business models have been disrupted by advanced digital technologies, which have, as a result, expanded disciplines and provided access to new stakeholders from other industries into the AEC industry [35]. Text-to-geometry models in the Minecraft gaming environment are a valid example of how LLMs are leveraged for automating 3D content generation [36]. Jang et al. (2024) [37] developed the Natural-language-based Architectural Detailing through Interaction with AI (NADIA) framework to automate wall detailing with an 83–98% accuracy. It achieves that by first separating wall creation from wall specification, then appropriating prompting instructions and seamlessly linking BIM authoring to LLM [37]. Gasparetta et al. (2022) [38] also reviewed recent text classification models with respect to data flow [38]. Multiple cooperating LLMs have been used to achieve increased robustness and task specialization through distribution of responsibilities among dedicated agents to reduce reasoning bottlenecks [39].

YOLO is a real-time object detection system introduced by Joseph Redmon et al. in 2016 [8]. It is known for its speed and accuracy in detecting objects in images and videos by executing object detection in a single forward pass of the network. This makes YOLO exceptionally fast, enabling it to run in real time with the capability to detect objects at 30–45 frames per second. Its train mode includes automatic dataset download for standard datasets, multi-GPU support to scale training, and real-time visualization and monitoring of training metrics. These features enhance the training experience and provide valuable insights into the learning process. YOLOv10's Val mode provides accurate metrics such as mAP50, mAP75, and mAP50-95, which offer comprehensive evaluation metrics of our model's performance. These metrics clearly demonstrate how effectively a model detects objects across various thresholds. Furthermore, YOLOv9 provides intuitive Python and command-line interfaces for rapid deployment and testing.

The Mask R-CNN framework presented by He et al. [9] is based on the Fast/Faster R-CNN (Region-based Convolutional Neural Network) and a Fully Convolutional Network (FCN) [8–10] framework enabling instance segmentation enhances image analysis by integrating object detection, which identifies and classifies individual objects within a bounding box, with semantic segmentation, which categorizes each pixel under predefined classes. This approach effectively differentiates between individual object instances. This means it detects objects within an image and generates high-quality segmentation masks for each instance. Mask R-CNN achieves this by adding a branch for predicting segmentation masks on each Region of Interest Align (RoIAlign) and using the Feature Pyramid Network (FPN) proposed by Lin et al. [11], in parallel with the existing branch for classification and bounding box regression. Instance segmentation can recognize complex figure shapes to be modeled in the MAS (Multi-Agent System).

The convergence of generative AI and MAS has enabled the development of intelligent environments capable of executing highly reasoning-intensive tasks and workflows through planning and knowledge integration to manage multidisciplinary reviews [40]. The next generation of intelligent systems will combine retrieval-augmented reasoning with multi-agent-based orchestration and computer vision to create adaptive and context-aware decision-support environments for increasingly complex analytical problems [41]. This study proposes an innovative automated compliance-checking framework to address the limitations of previous research and harness the strengths of LLMs, with the goal and objectives highlighted in Section 3 below.

3. Goals, Objectives and Contributions

3.1. Research Goals

The overarching goal of this research is to design, develop, and validate an intelligent, multi-agent AI system, designated the Co-Roof Designer, that functions as an adaptive assistant and collaborative decision-support tool for students and professionals in Architecture, Engineering, and Construction (AEC), as well as for building officials responsible for the regulatory review and approval of roof assemblies and rooftop structures. The Co-Roof Designer is conceived not as a replacement for human expertise but as an augmentative framework that enhances users' capacity to navigate the technical, regulatory, and design complexities inherent in roof system evaluation and compliance verification. Critically, this research does not seek to achieve full automation of the roof design process. Rather, it is grounded in a "designer-in-the-loop" collaborative paradigm, a human-centered AI philosophy in which the system functions as an intelligent partner that augments, informs, and guides human design reasoning rather than supplanting it. Within this paradigm, the Co-Roof Designer supports users in structuring design problems, identifying compliance requirements, evaluating design alternatives against applicable building codes, and visualizing outcomes—while preserving the designer's agency, judgment, and creative authority at every stage of the process. This approach reflects a broader recognition within the AI research community that the most effective and ethically responsible applications of AI in high-stakes professional domains enhance human decision-making rather than circumvent it.

3.2. Research Objectives

To achieve the stated research goal, the study is structured around four primary objectives, each corresponding to a discrete phase of the framework's development:

1. **Objective I: Data Collection and Corpus Development:** The first objective involves the systematic collection and curation of a comprehensive corpus of architectural and engineering documents pertaining to roof design and rooftop structures. This corpus encompasses PDF-format construction drawings, specifications, and building code provisions, with particular emphasis on the Florida Building Code: Residential and the Florida Building Code: Commercial, as well as annotated design examples that serve as ground-truth references for model training and validation. The quality, diversity, and representativeness of the data corpus are recognized as foundational determinants of the framework's overall performance and generalizability.
2. **Objective II: Data Preprocessing Agent Development:** The second objective involves designing and implementing a dedicated data preprocessing agent responsible for ingesting, parsing, and transforming raw PDF documents into structured, machine-readable representations suitable for downstream analysis. This agent integrates computer vision methods, specifically YOLOv11 and Mask R-CNN, to perform automated object detection and instance segmentation on architectural drawing sets, extracting roof design-relevant elements, including structural members, dimensions, material specifications, and annotation text. The preprocessing agent serves as the primary data ingestion layer of the multi-agent framework, ensuring that subsequent agents receive clean, structured, and semantically enriched inputs.
3. **Objective III: Data Analysis Agent Development:** The third objective focuses on developing a data analysis agent capable of performing automated compliance checks by interrogating extracted design data against the provisions of applicable building codes. This agent employs a Retrieval-Augmented Generation (RAG) architecture, combining a vector-based knowledge base of encoded building code provisions with the generative reasoning capabilities of large language models (LLMs) to produce

accurate, contextually grounded compliance determinations. The analysis agent is designed to handle the semantic ambiguity and interpretive complexity inherent in regulatory language, providing users with clear, actionable compliance assessments supported by explicit code references.

4. **Objective IV: Evaluation and Visualization Agent Development:** The fourth objective addresses the development of an evaluation and visualization agent that synthesizes the outputs of the preprocessing and analysis agents into coherent, human-readable reports and interactive visual representations. This agent generates structured compliance reports, flags non-compliant design elements, and produces graphical visualizations that enable users to rapidly interpret and act upon the system's findings. The evaluation agent also implements a systematic performance assessment protocol, applying standard information retrieval metrics, precision, recall, F1-score, and overall accuracy, to quantify the framework's compliance-checking performance across a range of design scenarios and LLM configurations.

3.3. Key Contributions

The present research makes the following original contributions to the fields of AI-assisted design review, automated code compliance checking, and AEC education:

1. **A Novel Multi-Agent Architecture with Asynchronous Task Execution:** The study introduces a purpose-built multi-agent AI architecture in which specialized autonomous agents—each assigned discrete functional responsibilities—collaborate through an asynchronous task execution framework. This architecture enables flexible compute scaling, allowing individual agents to operate concurrently and independently without creating processing bottlenecks, thereby supporting efficient performance across document sets of varying size and complexity. The asynchronous design also enhances the system's resilience and adaptability, enabling it to handle the heterogeneous document formats and regulatory contexts encountered in real-world AEC practice.
2. **An Integrated End-to-End Compliance Checking Pipeline:** Unlike prior approaches that address isolated components of the design review workflow, the proposed framework provides an integrated, end-to-end pipeline that spans document ingestion, element extraction, semantic code analysis, and results visualization—delivering a complete compliance review capability within a single, unified system architecture.
3. **A User-Friendly, Educationally Accessible Interface:** The Co-Roof Designer is equipped with an intuitive, user-friendly interface designed to serve the diverse needs of both professional practitioners and AEC students at undergraduate and graduate levels. The interface is designed to lower the barrier to engagement with AI-assisted compliance checking, enabling users without specialized computational backgrounds to interact with the system effectively and to develop applied AI literacy through direct engagement with the tool.
4. **A Validated Comparative Evaluation of LLM Performance for Code Compliance:** The study contributes a rigorous comparative evaluation of three families of large language models, Mistral, and GPT, in the specific context of building code compliance checking for roof assemblies. This evaluation provides empirical guidance for model selection in future AI-assisted regulatory review applications and advances the broader research literature on LLM performance in domain-specific, technically complex compliance tasks.

3.4. Scope and Delimitations

While the Co-Roof Designer framework is designed with extensibility as a core architectural principle, enabling its future adaptation to additional compliance domains across the AEC disciplines, the scope of the present study is deliberately bounded to roof assemblies and rooftop structures as governed by the Florida Building Code. This focused scope enables rigorous validation of the framework's core capabilities while providing a well-defined and practically significant domain of application.

Florida was selected as the building code of reference in this study for three main reasons: it possesses one of the most comprehensive and technically demanding roof-regulation environments in the United States due to hurricane-resilience requirements. Roof assemblies frequently require simultaneous evaluation against both the Florida Building Code Residential and Commercial provisions. Lastly, the study's datasets, annotated examples, and validation corpus were developed using Florida-specific regulatory documents to ensure consistency and experimental control.

4. Methodology

This study focuses on leveraging Machine Learning (ML) platforms, such as YOLO v11 and Mask R-CNN, Large Language Models (LLMs), and Retrieval Augmented Generation (RAG) to develop a Machine-Aided Multi-Agent System (MAS) for roof design regulatory conformance checking to aid in the design and construction process. The proposed system represents an AI Co-Roof Designer. The system employs specialized agents, including a triaging agent, a Preprocessing Agent, an Analysis Agent, and a Meta-Review Agent (which provides high-level analysis), to continuously generate and evolve, yielding better outcomes. Feedback on the output enables iterative improvement, creating a self-improving loop that leads to novel, high-quality output. Figure 2 presents an overview of the methodological workflow adopted for the proposed agentic AI framework for roof design compliance assessment. The process begins with the collection of blueprints and supporting design documents, including architectural drawings, enlarged roof details, specifications, applicable code provisions, and manufacturer product and installation catalogs. These inputs enter the data processing stage, where relevant roof design information is extracted, cleaned, and prepared for analysis. The resulting drawing dataset is then split into training, validation, and testing subsets for the YOLO11m-seg detection model.

Three specialized agents are then deployed. Agent 1 structures the extracted data into a machine-interpretable format for downstream reasoning; Agent 2 performs core compliance reasoning, juxtaposing extracted roof parameters against applicable code requirements; and Agent 3 assembles results for reporting and visualization. Agent outputs feed the testing and prediction stage, which generates the final deliverables: compliance determinations, annotated roof details, and review reports. This workflow shows how the framework's four core modules—data preprocessing and code ingestion, rule-based and semantic analysis, results visualization, and iterative validation—are realized as a coordinated pipeline from design documentation through agent-based reasoning to compliance reporting.

YOLO11m-seg is used for real-time detection of roof-related elements, including framing members, sheathing, roofing materials, connections, and rooftop components from PDF documents. Mask R-CNN is then employed for instance segmentation to isolate and delineate individual roof elements and annotations from PDF-based construction drawings. The drawing sheets are then partitioned into image tiles for processing. Then an OCR (Optical Character Recognition) is applied to extract textual annotations, dimensions, material specifications, and code-related notes. The extracted geometric and textual information is transformed into structured machine-readable records that serve as inputs to the downstream RAG and LLM components. This is further expanded in Section 5.

5. Proposed Agentic AI System

The proposed system integrates data preprocessing, analytical processing, result visualization, and validation. At its core, the system features a multi-agent architecture supported by an asynchronous task execution framework, enabling scalable and flexible computing for roofing design and construction applications (see Figure 3). The following sections provide a detailed overview of these system components.

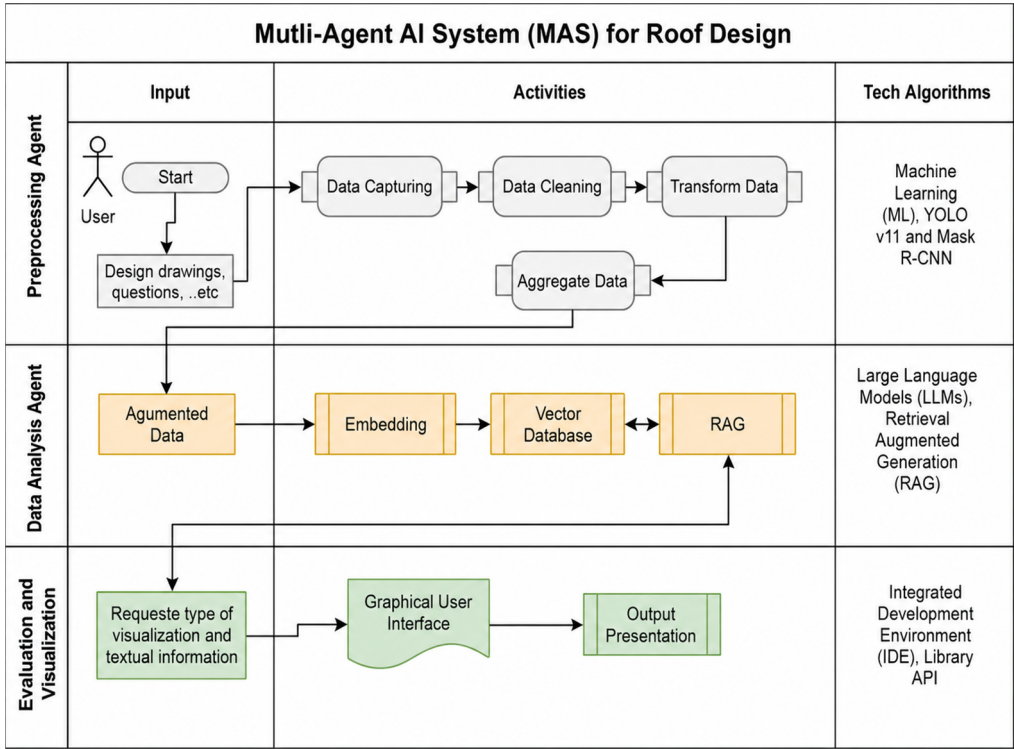


Figure 3. The MAS architecture.

1.
- The data collection phase focuses on selecting roof design data from various local design offices’ sources, in addition to publicly available data including code and local regulations (see Figure 4).

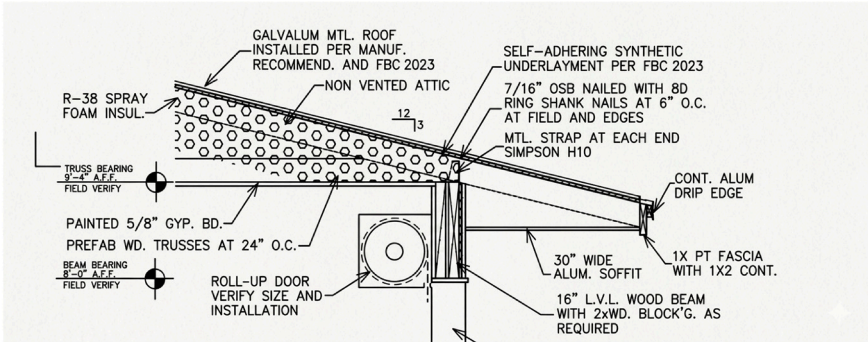


Figure 4. Example of roof design data.

2.
- The data preprocessing agent will extract the necessary data from the previous phase. This step involves leveraging Machine Learning (ML) platforms, such as a convolutional neural network called You Look Only Once (YOLO) v11 and MaskR-CNN (Mask Region-based Convolutional Neural Network) [8–10], to extract relevant roof design data from the PDF or image drawings. This would include calibration and splitting data into training, testing, validation, and prediction (see Figure 2).

3. The data analysis agent will analyze the data obtained and interact with the code and regulations data sources using LLMs and RAGs to provide feedback about the regulatory conformance of each roof element detected by the preprocessing agent.

Dataset, Annotation, and Vision Model Protocol

Dataset: The dataset comprises 150 engineering roof design drawings from active construction projects in Florida, including structural plans, connection details, and specification sheets prepared in accordance with the Florida Building Code. Each drawing was assigned a unique identifier that encoded the project, sheet, revision, and page, and metadata, including sheet type, scale, resolution, and annotation status, were tracked in a split manifest.

Preprocessing: Drawings were rasterized from PDF to PNG at 300 dpi minimum (600 dpi for detail sheets). Because engineering drawings are large, sparse, and line-based, full sheets were processed using overlapping 1024×1024 -pixel tiles with 20% overlap. All tiles from a single drawing were assigned to the same split partition to prevent tile-level leakage.

Train/Validation/Test Split: Drawings were divided into training (80%, $n = 120$), validation (10%, $n = 15$), and test (10%, $n = 15$) sets. The split was performed at the project level, not the sheet level, to prevent data leakage and to ensure the test set evaluates generalization to unseen drawing styles. As a robustness check, five-fold cross-validation was conducted on the 135 non-test drawings.

Annotation Protocol: Drawing elements were annotated using polygon instance masks for physical structural components (beams, columns, braces, plates, fasteners, anchor rods, wood trusses, roof sheathing, roof slope, roof connections, wall, underlayment, type of roof materials, wall studs, insulation) and bounding boxes for symbolic and textual elements (dimension annotations, leader lines, section markers, detail callouts, grid bubbles). The taxonomy was limited to 15 classes. Annotations followed a two-pass quality control process comprising an initial annotation pass and a technical review pass. Inter-annotator agreement was measured on a 10% sample ($n = 15$ drawings) with acceptance targets of $\geq 90\%$ class agreement, mean box IoU ≥ 0.75 , and mean mask IoU ≥ 0.70 .

Model Configuration: YOLO11m-seg was trained with an image size of 1024, 150 epochs, AdamW optimizer, initial learning rate of 0.001, cosine learning rate scheduling, and early stopping patience of 30 epochs. Mask R-CNN with a ResNet-50-FPN backbone was trained as a comparison baseline with input sizes of 1024–2048 and custom anchor aspect ratios to capture thin, elongated objects. Both models were trained on the same split.

The quantitative results are as follows. On the held-out test set, YOLO11m-seg outperformed Mask R-CNN across all reported metrics: mAP@0.50 of 0.92 vs. 0.85, mAP@0.50:0.95 of 0.75 vs. 0.65, precision of 0.91 vs. 0.88, recall of 0.88 vs. 0.82, and mean mask IoU of 0.83 vs. 0.74. At the class level, large structural elements, including beams, columns, walls, roof trusses, and sheathing, were detected with consistently higher confidence, while smaller or visually ambiguous elements such as fasteners, underlayment, and roof-connection components showed lower per-class AP, reflecting their limited visual distinctiveness and underrepresentation in training data. Tables 1 and 2 summarize these results.

Table 1. Overall Detection and Segmentation Performance.

Metric	YOLO11m-seg	Mask R-CNN
mAP@0.50	0.92	0.85
mAP@0.50:0.95	0.75	0.65

Table 1. *Cont.*

Metric	YOLO11m-seg	Mask R-CNN
Precision	0.91	0.88
Recall	0.88	0.82
Mean Mask IoU	0.83	0.74

Table 2. Per-Class Detection Reliability.

Element Class	Category	Relative Performance
Beams	Structural	High
Columns	Structural	High
Walls	Structural	High
Roof Trusses	Structural	High
Sheathing	Structural	High
Fasteners	Connection detail	Lower
Underlayment	Surface layer	Lower
Roof-connection components	Connection detail	Lower

Prompts: An example of the prompts used in the RAG pipeline and LLM reasoning components, including prompts for compliance classification, code provision retrieval, and decision synthesis, is given below,

“You are a licensed Florida Building Code expert and structural compliance auditor with deep specialization in FBC-R 2023 (Florida Residential Building Code 2023), ASCE 7-22 wind load provisions, and Florida High-Velocity Hurricane Zone (HVHZ) requirements. You apply code requirements with precision, cite specific sections, and flag ambiguities rather than assume compliance.”

TASK:

Perform a rigorous compliance validation of the roof design specifications provided below against FBC-R 2023 and all referenced standards (ASCE 7-22, ASTM, TAS, Miami-Dade NOA where applicable).

PRIMARY SCOPE:

Roof system elements only —

- *Structural framing: trusses, rafters, ridge boards/beams*
- *Roof slope/pitch*
- *Sheathing: thickness, span rating, fastening schedule*
- *Connections & uplift resistance: truss-to-wall, rafter ties, hurricane straps*
- *Roof covering: material type, attachment method, underlayment*
- *Wind resistance: design wind speed compliance, component & cladding pressures*
- *Overhangs, soffits, and eave details*
- *Ventilation (where code-mandated for roof assembly)*

VALIDATION INSTRUCTIONS:

1. SECTION-BY-SECTION ANALYSIS

For each wall section/roof zone identified in the input:

- a. Identify the specific code requirement (cite FBC-R 2023 section number).
 - b. State the design value from the input data.
 - c. State the code-required value or range.
 - d. Render verdict: PASS | FAIL | INSUFFICIENT DATA | FLAG FOR ENGINEER.
2. CONSERVATIVE INTERPRETATION
When data is ambiguous or incomplete, DO NOT assume compliance.
Explicitly flag the gap and explain what additional information is needed.
3. HVHZ AWARENESS
If the project location triggers HVHZ provisions (Miami-Dade or Broward County), apply the stricter TAS/NOA requirements and note this distinction.
4. CROSS-REFERENCE CONFLICTS
If FBC-R 2023 conflicts with a referenced standard (e.g., ASCE 7-22), apply the more restrictive requirement per FBC-R Section R101.4 and note the conflict.
5. REASONING BEFORE VERDICT
For each check, show your reasoning chain before stating the verdict. Do not jump to conclusions.
- IMPORTANT RULES:**
- Do not assume compliance when required information is missing.
 - If data is missing, classify the item as INSUFFICIENT DATA or REQUIRES FURTHER REVIEW.
 - Do not evaluate non-roof elements unless they affect the roof load path or roof code compliance.
 - Distinguish between prescriptive code compliance and engineered design requirements.
 - Do not invent dimensions, loads, fastener schedules, product approvals, or connection capacities.
 - If the input data conflicts with FBC-R 2023 requirements, clearly identify the conflict.
 - If a requirement depends on site-specific conditions, list the required site data."

Retrieval Strategy: The retrieval component indexes drawing content at three levels: sheet level, tile level (1024 px crops with 20% overlap), and object level (detected component crops). Embeddings are generated separately for visual crops, OCR-extracted text, and detected class sequences. Retrieval queries are executed over a vector index and reranked using metadata filters and cross-encoder scoring. The complete retrieval pipeline is described in Section 3 and diagrammed in Figures 5 and 6.

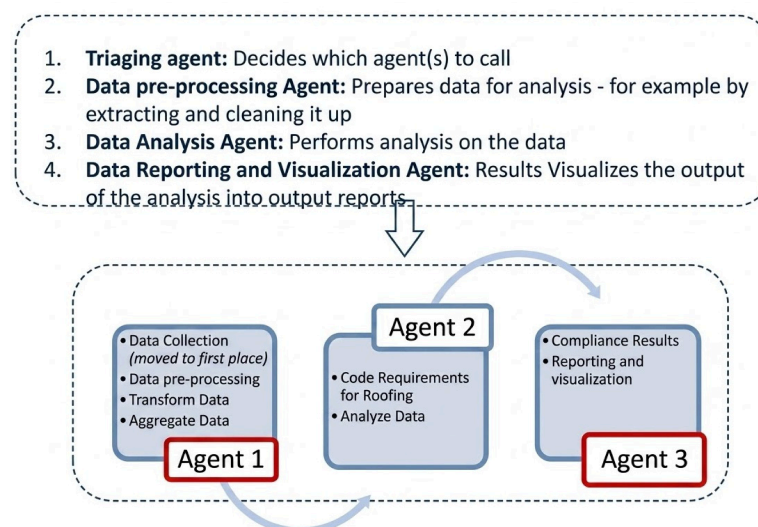


Figure 5. MAS Implementation Architecture.

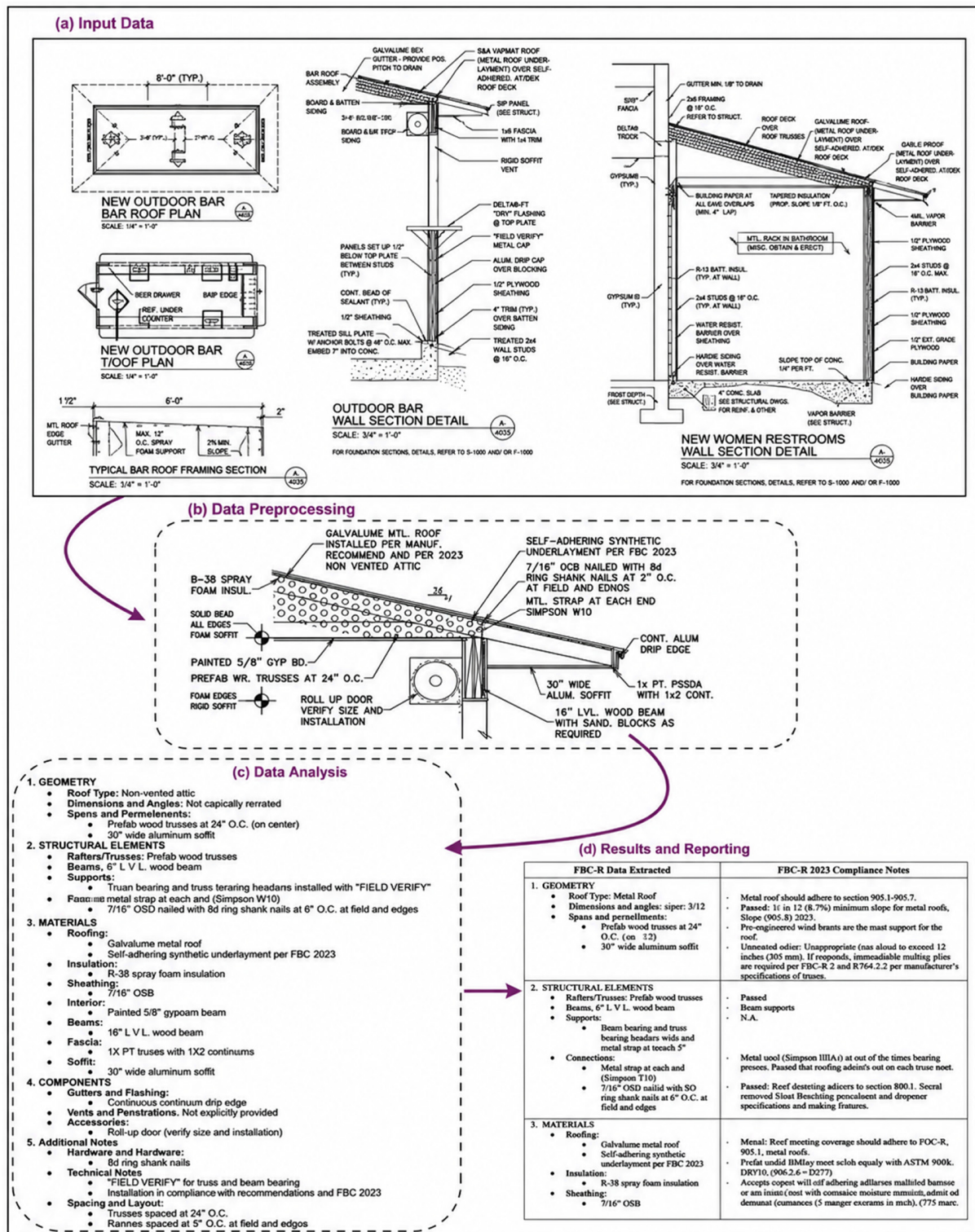


Figure 6. Implementation Overview.

Ground-Truth Validation: Independent ground-truth compliance labels were assigned by a licensed architect and a structural engineer with no involvement in model development, based on direct review of the drawings and applicable Florida Building Code provisions. These labels were used as the reference standard for all LLM evaluation metrics reported in the Results section.

The output agent will present the results of the analysis in the requested visualization format. This could include code compliance reports and a section view of the roof, along with identifying parameters and other textual information.

6. Implementation

Implementation is based on logically grouping and assigning specialized “Agents” designed to handle specific tasks or subtasks. This modular approach enhances efficiency and improves overall system performance. Figure 5 illustrates the execution plan for the proposed Multi-Agent System (MAS). It illustrates the implementation architecture of the proposed MAS, organized as a sequence of specialized agents that collaboratively execute the workflow. Agent 1 collects data, preprocesses, transforms and aggregates roof-design information extracted from input documents. The processed data is then passed unto Agent 2 where it is interpreted in accordance with Florida Building Code (FBC) 2023 roofing requirements and performs a compliance analysis. Agent 3 consolidates these compliance outcomes and generates visualization outputs for end users.

The execution flow is coordinated by a triaging mechanism by assigning tasks to the appropriate agents. This ensures that the workflow is modular, scalable and concerns are compartmentalized to facilitate efficient handling of complex roof-design compliance tasks. Each agent can be enhanced and modified without compromising the overall system.

Typical outputs are presented in Figure 6. This illustrates the input data and results produced by each agent, as well as the code-compliance reports for the residential roof design, per the Florida Building Code (FBC) 2023.

Like YOLOv9, YOLOv11 also provides intuitive python and command line interfaces for rapid development and testing. Python programming language was utilized for PDF processing and document ingestion, OCR integration, YOLOv11 object detection, Mask R-CNN segmentation, embedding generation, vector database interactions, RAG orchestration, Agent communication, workflow automation and results generation and visualization.

7. Results and Discussion

Following training and fine-tuning, the system was evaluated on fifteen new and distinct roof-design projects, denoting Sample Test 1 through Sample Test 15, using four large language models: GPT-5.4, GPT-5, GPT-4.1 nano, and Mistral Small 3.1. The Florida Building Code (FBC) 2023 served as the regulatory baseline for all comparisons.

Compliance was evaluated across four interrelated structural and weatherproofing categories that together cover the principal failure modes addressed in the FBC 2023: (i) roof framing and wood trusses, (ii) roof sheathing, roof slope, roof-to-wall connections (including wind-resistance hardware), (iii) roof covering, and (iv) underlayment with associated edge details. For each test sample, every category received exactly one of three status labels: Compliant (C), Requires Review (RR), or Non-Compliant (NC).

A category outcome was scored as a true positive (TP) only when the model-generated label matched the code-baseline label exactly. Any disagreement was counted simultaneously as a false negative (FN), because the correct baseline label was not produced, and as a false positive (FP), because an alternative label was asserted in its place. This dual counting reflects the multi-class nature of the task: a single mislabel both omits the correct class and incorrectly populates another. The metrics derived from these counts are defined below.

Precision:

$$Precision = TP / (TP + FP) \quad (1)$$

Recall:

$$Recall = \frac{TP}{TP + FN} \quad (2)$$

F1 Score:

$$F1 = 2 \frac{(\text{Precision} \times \text{Recall})}{(\text{Precision} + \text{Recall})} \quad (3)$$

Accuracy:

$$\text{Accuracy} = \frac{\text{No. of Exact Category Matches}}{4} \quad (4)$$

(Note: 4 represents the number of categories being evaluated.)

The following sections and diagrams illustrate the results of the comparative analysis.

7.1. Scope, Notations and Scoring Rules

Scope: Each testing sample was scored across four code-compliance categories: Framing, Sheathing, Connections & Wind, and Covering & Edge. Across all 15 samples, this yielded 60 reference decisions that constitute the evaluation set.

Scoring rule: Scoring followed an exact-match convention: a category was credited only when the model-generated label matched the code-baseline label verbatim. Because each sample contributes exactly four decisions, a one-category disagreement on a given sample translates to a 25-percentage-point reduction in that sample's exact-match accuracy, providing a coarse but transparent per-sample resolution.

Notations: Each decision is assigned to one of three labels: Compliant (C), Requires Review (RR), or Non-Compliant (NC). The full label set is preserved throughout the analysis even when a class is unrepresented in a particular split, so that confusion matrices and per-class metrics remain dimensionally consistent across experiments (see Table 1).

Dataset profile: The 15 evaluation reports were generated using the agentic AI tool for roof design and detailed wall- and roof-section drawings, yielding fine-grained, drawing-level compliance comparisons rather than document-level comparisons.

Table 3 presents a summary of F1 score results across four language models, evaluated on an exact category-match task. Three metrics are reported:

Table 3. Summary of the F1 score results.

Metric	GPT-4.1 Nano	GPT-5	Mistral Small 3.1	GPT-5.4
Mean exact-match accuracy/F1	78.3%	91.7%	96.7%	97.0%
Median exact-match accuracy/F1	75.0%	100.0%	100.0%	97.0%
Full-score samples	2	10	13	15

The mean exact-match *F1* (first row) reflects the average performance across all test samples, giving an overall sense of each model's consistency. GPT-4.1 nano scored the lowest at 78.3%, while GPT-5.4 led with 97.0%, closely followed by Mistral Small 3.1 at 96.7% and GPT-5 at 91.7%.

The median exact-match *F1* (second row) is less sensitive to outliers and better captures typical performance. GPT-5 and Mistral Small 3.1 both achieved a perfect median of 100.0%, suggesting that more than half of their predictions were exact matches. GPT-5.4's median of 97.0% aligns closely with its mean, indicating stable and consistent performance, while GPT-4.1 nano's median of 75.0%, below its mean, suggests a skewed distribution with some higher-scoring samples pulling the average up.

The full-score samples count (third row) represents the number of test cases where the model achieved a perfect score. GPT-5.4 received the fullest scores (15), followed by Mistral Small 3.1 (13), GPT-5 (10), and GPT-4.1 nano (2), reinforcing the ranking observed in the other metrics.

Together, these metrics indicate that GPT-5.4 and Mistral Small 3.1 are the strongest performers, while GPT-4.1 nano lags considerably behind, both in average accuracy and in the number of perfectly scored samples.

The framework was evaluated using a multi-dimensional validation strategy designed to assess both predictive performance and practical code-review behavior. Rather than relying on a single accuracy measure, the evaluation incorporated portfolio-level performance metrics, category-specific analyses, agreement-rate assessments, compliance-alignment evaluations, severity-based outcome distributions, and F1-score benchmarking. This combination of measures provides a broader understanding of model behavior across diverse roof-compliance scenarios and enables a more comprehensive comparison of the evaluated LLM families.

7.2. Portfolio Mean Accuracy

The Portfolio Mean Accuracy is a single, model-level summary statistic that aggregates a model's exact-match performance across the full evaluation set ("portfolio") of testing samples. It answers the question, "On a typical sample drawn from the test portfolio, what fraction of category decisions does the model get exactly right?"

Formally, let the portfolio consist of N testing samples, each of which contributes k category decisions (in this study, $N = 15$ and $k = 4$, giving $N \cdot k = 60$ reference decisions in total). For sample i , let c_i denote the number of categories on which the model's label exactly matches the FBC-2023 baseline label. Define the per-sample exact-match accuracy as

$$a_i = c_i/k \quad (5)$$

Then the portfolio mean accuracy of the model is the arithmetic mean of these per-sample accuracies:

$$\bar{A} = (1/N) \cdot \sum_i a_i = (\sum_i c_i) / (N \cdot k) \quad (6)$$

Three properties make $\bar{A}$ a useful headline metric. It is bounded in $[0, 1]$ (or 0–100% when expressed as a percentage), making it directly comparable across models. It is interpretable: a value of 0.95 means that, on average, 19 of every 20 category judgments agree with the code baseline. And it is granular at a known resolution: with $k = 4$ categories per sample, a single category disagreement on one sample shifts that sample's accuracy by 25 percentage points, and the portfolio mean by $25/N = 1.67$ percentage points, so differences smaller than ~2 points between two models on a 15-sample evaluation should be regarded as within-sampling noise rather than as evidence of a true performance gap.

$\bar{A}$ should be distinguished from per-class metrics such as precision, recall, or $F1$, which condition on a particular label (Compliant, Requires Review, Non-Compliant). Portfolio means accuracy treats every category decision as equally weighted, regardless of class, and therefore can be high even when a model performs poorly on a minority class, a known limitation of accuracy under class imbalance, which is why this study reports it alongside the per-class breakdowns rather than in isolation.

Figure 7 illustrates that the four models are distributed across three meaningful performance bands rather than along a smooth gradient. GPT-5.4 (97.0%) and Mistral Small 3.1 (96.7%) cluster together inside the "excellent" band ($\geq 95\%$); the 0.3-percentage-point separation between them is well within sampling noise for a 60-decision evaluation set, so they should be treated as statistically tied at the top. Both models clear the threshold at which a tool's outputs are reliable enough to act on with only spot-check human review, and either is a defensible production choice for this compliance task.

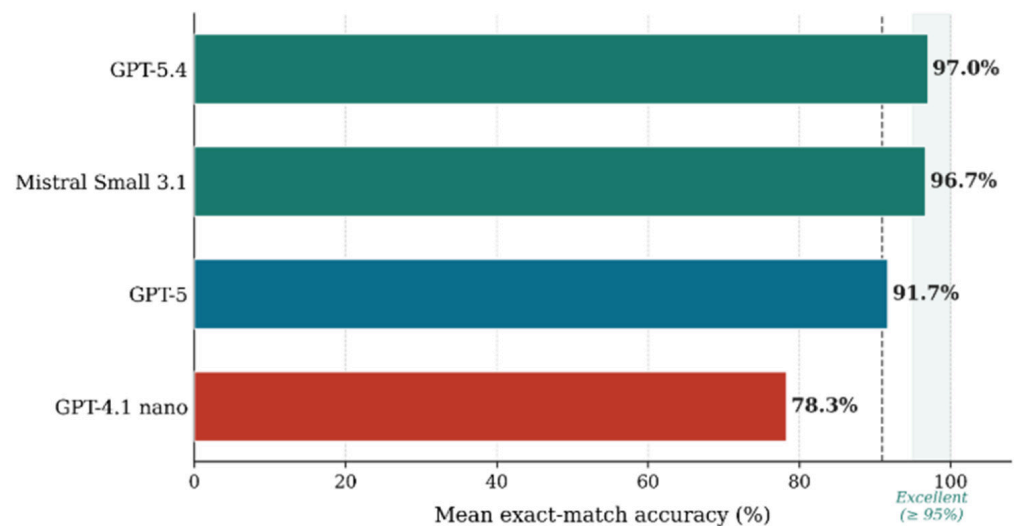


Figure 7. Portfolio Mean Accuracy of different models.

GPT-5 (91.7%) sits in a clearly distinct middle band, strong, but separated from the leaders by a ≥ 5 -point gap that exceeds plausible sampling noise on this dataset. Practically, this means GPT-5 will produce roughly one mismatched category every 12 decisions versus roughly one every 30 for the leaders, which translates into materially more reviewer rework over a sustained workload. It is acceptable as a fallback or as a cost/latency-optimized alternative, but it is not on par with the leading pair.

GPT-4.1 nano (78.3%) is the clearest outlier. Its 13–19-point deficit relative to the other three is too large to attribute to model variance and instead reflects a systematic capability gap. With more than one in five category decisions mismatching the FBC 2023 baseline, deploying it without intensive human verification would push errors through to downstream outputs at a rate that defeats the purpose of automation. Its position below the portfolio mean (90.9%, dashed line) and below every other model argues against using it as anything other than a baseline reference.

Two structural observations can be deduced from the chart. First, the portfolio mean of 90.9% is pulled meaningfully by the single weak model; the median across models is 94.2%, which is a more honest summary of “what a competent model on this task achieves.” Second, the visible bimodality (a tight top cluster at 96–97%, a middle entry near 92%, and a far-back outlier at 78%) suggests that further accuracy gains will come not from incremental tuning of the leading models—they are already near the ceiling, but from understanding why GPT-4.1 nano’s smaller capacity translates into this particular kind of failure on FBC 2023 categories.

7.3. Category-Level Behavior

Where *Portfolio Mean Accuracy* compresses a model’s performance into a single number, category-level behavior decomposes that number along the evaluation’s category axis to reveal which kinds of compliance decisions a model handles well and which it does not. Formally, for each model m and each of the k categories c in the framework (here $c \in \{\text{Framing, Sheathing, Connections \& Wind (C\&W), Covering \& Edge (C\&E)}\}$), the category-level match rate is

$$\bar{A}_{m,c} = (\text{matches of model } m \text{ on category } c) / (\text{total reference decisions in category } c) \quad (7)$$

Across the four categories, the resulting vector:

$$\bar{A}_m = \begin{bmatrix} \bar{A}_m, \text{Framing} \\ \bar{A}_m, \text{Sheathing} \\ \bar{A}_m, \text{C\&W} \\ \bar{A}_m, \text{C\&E} \end{bmatrix} \quad (8)$$

is the *Category-Level Behavior Profile* of model m . Two models can share the same portfolio mean accuracy and still display very different profiles, e.g., one weak on Framing only, another weak on Sheathing only, and those qualitative differences typically matter more for deployment decisions than the headline number, because they reveal the failure mode a downstream reviewer must compensate for. The same dataset is also summarized column-wise by category (across models) to identify which provisions of the FBC 2023 are systematically harder for the model family.

Figure 8 shows the Category-Level Exact Match rate by model. This provides information on how each model described the building code compliance status. Connections & Wind and Covering & Edge are fully stable across all comparison models in this dataset. The signal in this study lives in the other two categories. *Framing* is the most discriminating axis (cross-model mean 76.7%): GPT-5.4 and GPT-5 are perfect, Mistral Small 3.1 misses two of fifteen samples (86.7%), and GPT-4.1 nano collapses to 20%, the single largest deficit anywhere in the matrix. *Sheathing* is the second discriminating axis (cross-model mean 90.0%) and inverts the failure pattern: GPT-5.4, Mistral Small 3.1, and GPT-4.1 nano all sit at or near the ceiling, while GPT-5 drops to 66.7% with five misses. Together, these two columns explain virtually all of the between-model variance seen earlier in the portfolio mean accuracy and F1 charts.

What this means for model selection is more interesting than the headline ranking suggests. GPT-5.4 is the only model with a clean, no-weakness profile across all four categories except for some narrative depth and code citation quality and is therefore the safest default. Mistral Small 3.1 is a very close second, with its small Framing deficit confined to two samples that a reviewer can easily target. GPT-5, contrary to what its 91.7% portfolio might imply, is *not* a generalist understudy; it is a category-specific specialist with a meaningful blind spot in Sheathing, and any deployment using it should add a Sheathing-focused human-review step rather than a uniform spot-check. GPT-4.1 nano's failure mode is the most diagnostic of all: an 80-point hole on Framing combined with intact performance everywhere else suggests not random underfitting but a systematic inability to parse the structural-truss provisions of the FBC, likely tied to the specificity of the framing-related code language exceeding the model's representational capacity. Adding more samples will not close this gap; targeted prompt redesign or fine-tuning on framing-specific text would be required.

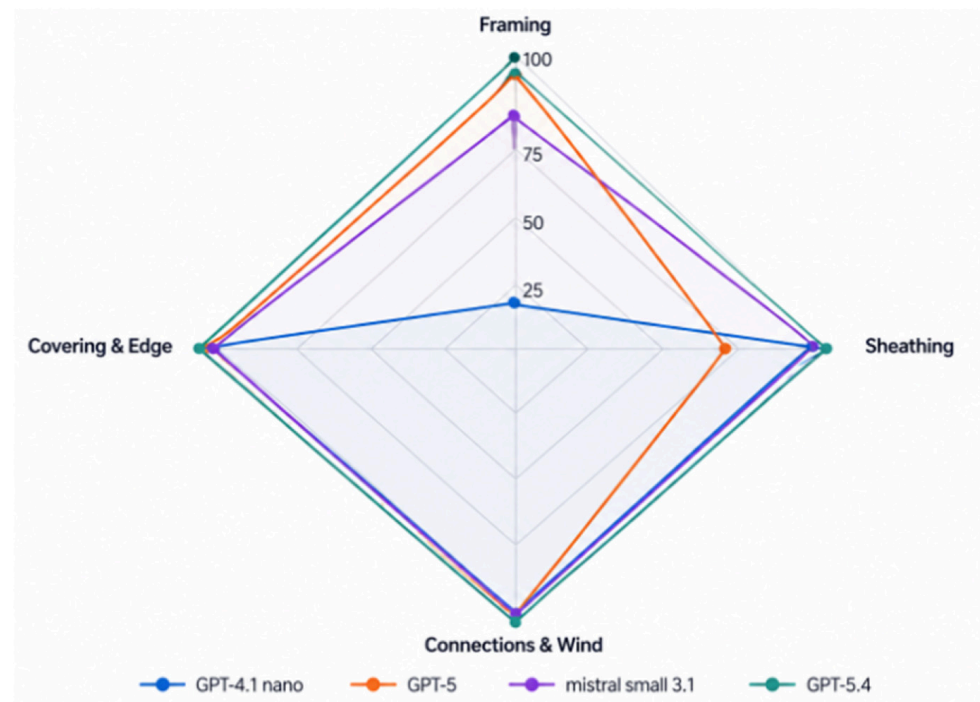


Figure 8. Category-level exact-match rate. Axes = compliance category and match rate building code.

7.4. The Agreement Rate

The agreement rate is a per-class measure of how often a model's label matches the code-baseline label, computed conditional on the baseline class. For a class y (e.g., Compliant or Requires Review), it is

$$AR(y) = \frac{TP(y)}{TP(y) + FN(y)} \quad (9)$$

It is the proportion of decisions whose true baseline label is y that the model also assigned to y . Mathematically, it coincides with recall (sensitivity) for class y . It deliberately ignores the other class's behavior so that the two AR values, taken together, describe a model's balance: a high agreement rate on one class but a low one on the other indicates a biased classifier that is, in effect, betting on a single label, while two simultaneously high agreement rates indicate genuine discrimination. AR is preferred over overall accuracy under class imbalance because it cannot be inflated by simply guessing the majority class. Figure 9 below shows the category-level behavior by model and the difficulty by category (mean across models).

GPT-5 (100, 89) and Mistral Small 3.1 (87, 100) are mirror images of each other on the off-diagonals: each saturates one class and gives up roughly 11–13 points on the other. GPT-5 perfectly recovers Compliant items but misses about 11% of true Requires-Review cases; it tends to label items "OK". Mistral Small 3.1 perfectly recovers Requires-Review items but misses 13% of true Compliant items; it leans toward "needs another look." Their harmonic balance scores (94.2 and 93.0) are statistically close, but the direction of bias matters for deployment: in a regulatory setting, where a missed Requires-Review is far costlier than an over-cautious flag, Mistral's bias is the safer one. GPT-5's bias, conversely, would lead to false-clear errors slipping past the human reviewer.

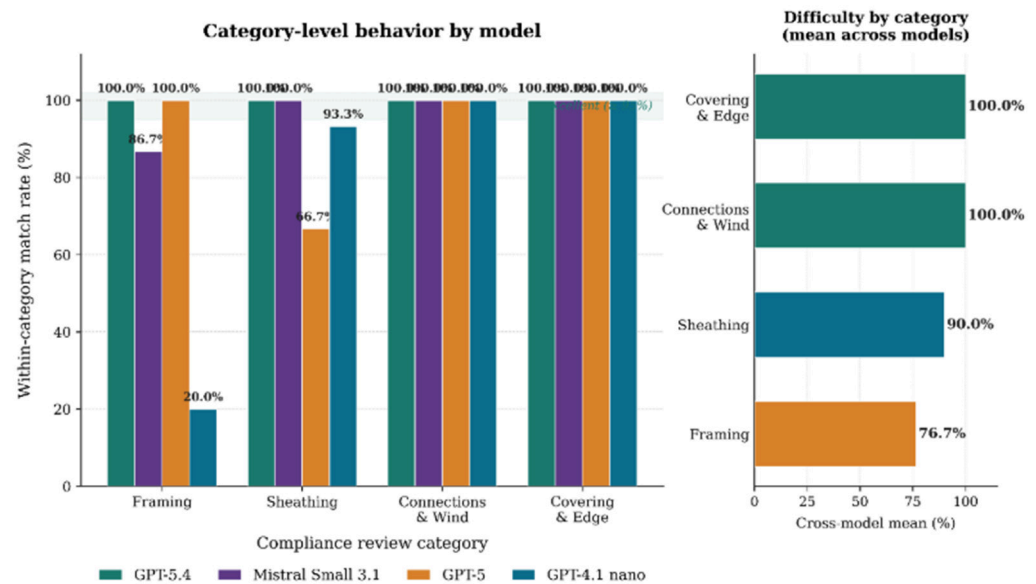


Figure 9. Agreement rate. X-axis = reference status class. Y-axis = within-class match rate (%).

GPT-4.1 nano (13, 100) is pinned to the top-left edge of the chart, a textbook majority-class predictor. It catches every Requires-Review item only because it labels almost everything as Requires Review; its 13% agreement on Compliant means that, of the items the code says are clearly compliant, the model agrees only 13% of the time. This is what produces its harmonic balance score of 23.0, an order of magnitude worse than the next-weakest model. The point's position deep inside the “biased toward RR” region makes the failure mode visible at a glance: this is not a noisy or generally weak classifier so much as a degenerate one, and additional samples will not move it off this edge without a structural intervention (fine-tuning, retrieval grounding, or a different model entirely).

There are two cross-cutting takeaways from the geometry. First, distance from the diagonal is a more honest indicator of usefulness than per-class accuracy alone. Three of the four models exceed 95% on at least one class, but only one of them sits near the diagonal. Secondly, the H-contours reveal that the production-deployable region ($H \geq 90$) is a small island in the upper-right of space; only GPT-5.4, GPT-5, and Mistral Small 3.1 enter it, and only GPT-5.4 sits comfortably inside it rather than on its boundary.

Notwithstanding that the exact-match accuracy covers all rubric dimensions, including narrative and code citation. It is different from the decision-level accuracy. As a result, the 100% decision alignment and 97.0% overall exact-match accuracy for GPT-5.4 are consistent under this scoring scheme.

7.5. Status Distribution and Severity Posture

Figure 10 shows that the reference distribution is 15 Compliant and 45 Requires Review (25%/75%), reflecting a code-review portfolio intentionally tilted toward items that warrant a second look. Three models reproduce that prior to within a few decisions: GPT-5.4 matches it exactly (15/45), Mistral Small 3.1 lands two below on Compliant (13/47), and GPT-5 lands five above (20/40). These small deviations are within the noise band one would expect from any well-calibrated classifier on a 60-decision sample. GPT-4.1 nano, by contrast, predicts only 2 Compliant decisions out of 60, a 13-decision deficit relative to baseline and effectively a degenerate single-class predictor that almost always says “Requires Review.” This is the same failure already visible in the agreement-rate quadrant, but the status-distribution view exposes its operational consequence directly: the model is

producing the right aggregate workload (everything routed to a reviewer) at the cost of providing essentially no automated discrimination.

The deviation panel makes a subtle point too. GPT-5 is the only model that over-applies Compliant ($\Delta = +5$), and although its accuracy sits near 92%, this slight over-eagerness to clear items is exactly the direction of bias one most wants to avoid in a regulatory tool. GPT-5.4 ($\Delta = 0$) and Mistral Small 3.1 ($\Delta = -2$) are calibrated in the safer direction.

Severity Posture: Where the status distribution describes the model's outputs, the severity posture describes the *direction* of its disagreements. For each decision in which the model's label differs from the baseline, the disagreement is signed as

- More permissive if the model assigns a less severe status than the baseline (e.g., labeling a true Requires Review item as Compliant, or a true Non-Compliant as Requires Review),
- Aligned if the labels match,
- More conservative if the model assigns a more severe status than the baseline (e.g., labeling a true Compliant item as Requires Review).

This direction-aware view is essential in regulatory settings because the two error directions have asymmetric costs: a more permissive error allows a non-compliant condition to go undetected, while a more conservative error incurs only an extra check. Two models with identical accuracy can have opposite postures and therefore opposite operational risk profiles.

The Severity Posture is depicted in Figure 11. GPT-4.1 nano is dominated by conservative shifts, especially in Framing. GPT-5 is almost entirely aligned except for its five more permissive sheathing calls. Mistral small 3.1 is aligned almost everywhere, with only two conservative Framing shifts. GPT-5.4 is also aligned mostly everywhere.

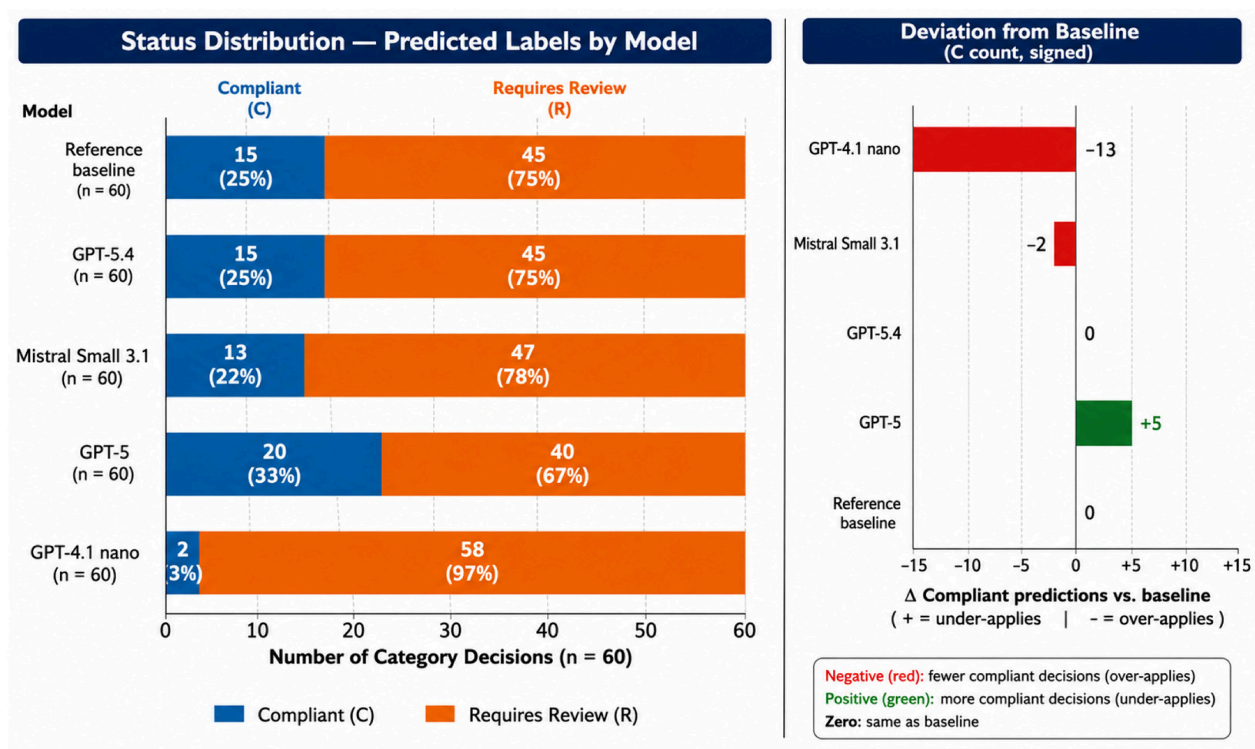


Figure 10. Status distribution across all category decisions.

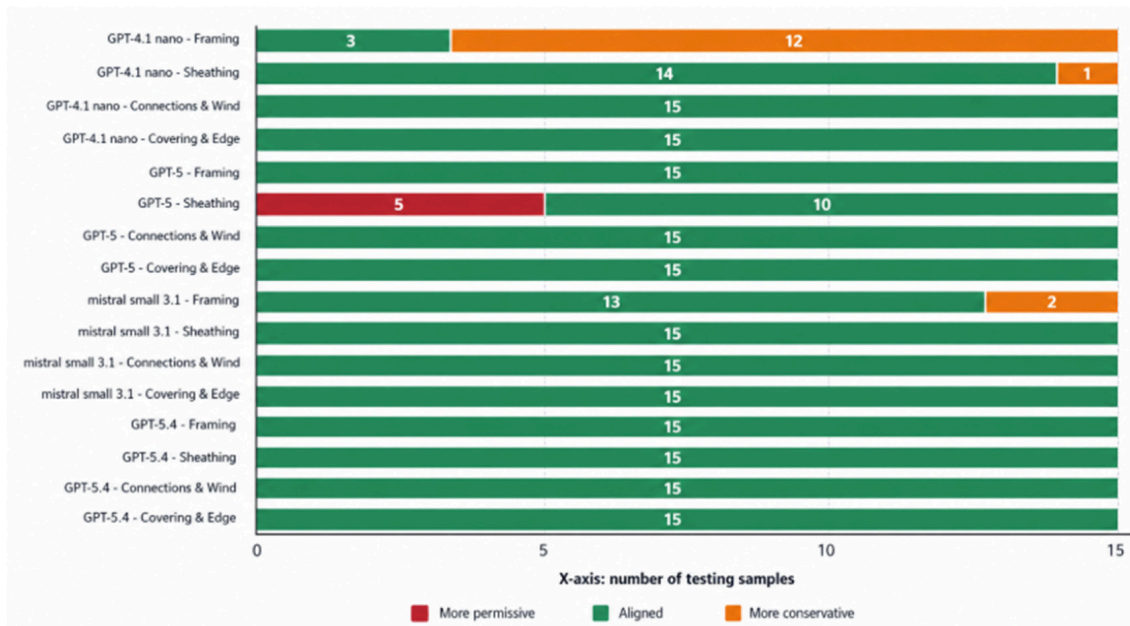


Figure 11. Severity shift by category.

7.6. Code Compliance Alignment

Figure 12 shows that the overall match rate per category (Framing 20.0%, Sheathing 93.3%, Connections & Wind 100%, Covering & Edge 100%), with bars color-coded by performance tier. The Framing column makes the failure mode visually obvious; the model misses Framing on 12 of 15 samples. The overall alignment is 78.3%.

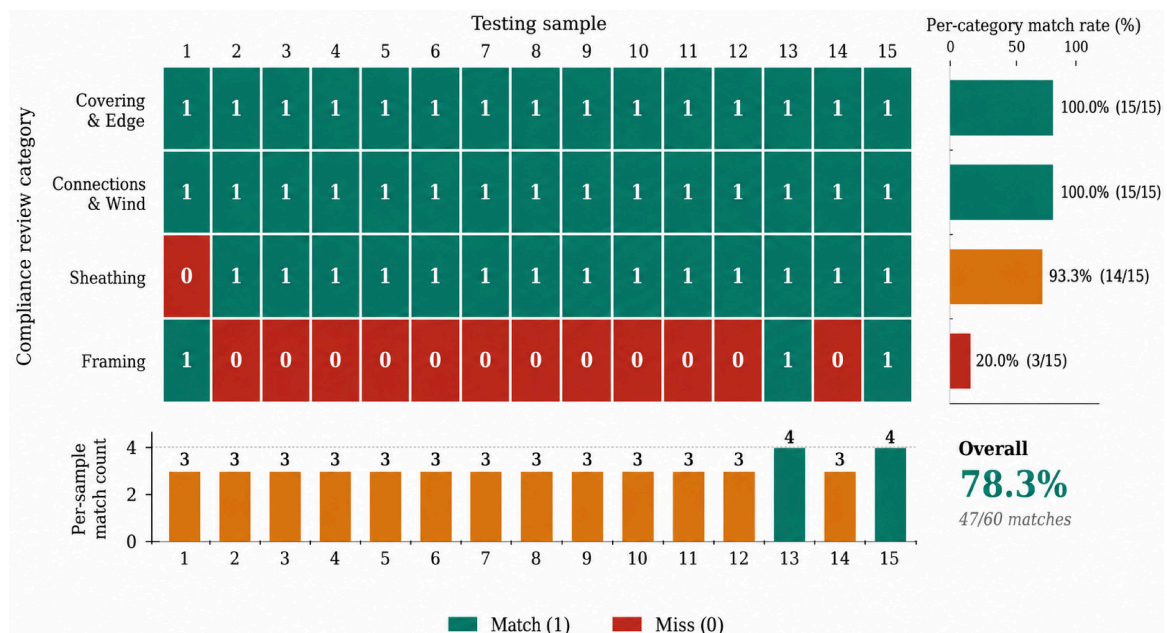


Figure 12. Compliance Review confusion matrix for GPT-4.1 nano.

In Figure 13, the confusion matrix displays that GPT-5 differences are extremely localized. All five misses sit in the Sheathing column, and every other category matches code compliance results. Framing, Connections & Wind, and Covering & Edge are perfect (100%, 15/15), while Sheathing is the weak category at 66.7% (10/15) with misses concentrated in samples 2, 3, 9, 10, and 12. Overall match rate is 91.7% (55/60).



Figure 13. Compliance Review confusion matrix for GPT-5.

In Figure 14, the results for Mistral small 3.1 depict the strongest of the previous results, with an overall match rate of 96.7% (58/60). Sheathing, Connections & Wind, and Covering & Edge are perfect at 100% (15/15), while Framing is the only weak group at 86.7% (13/15) with misses isolated to samples 4 and 7.

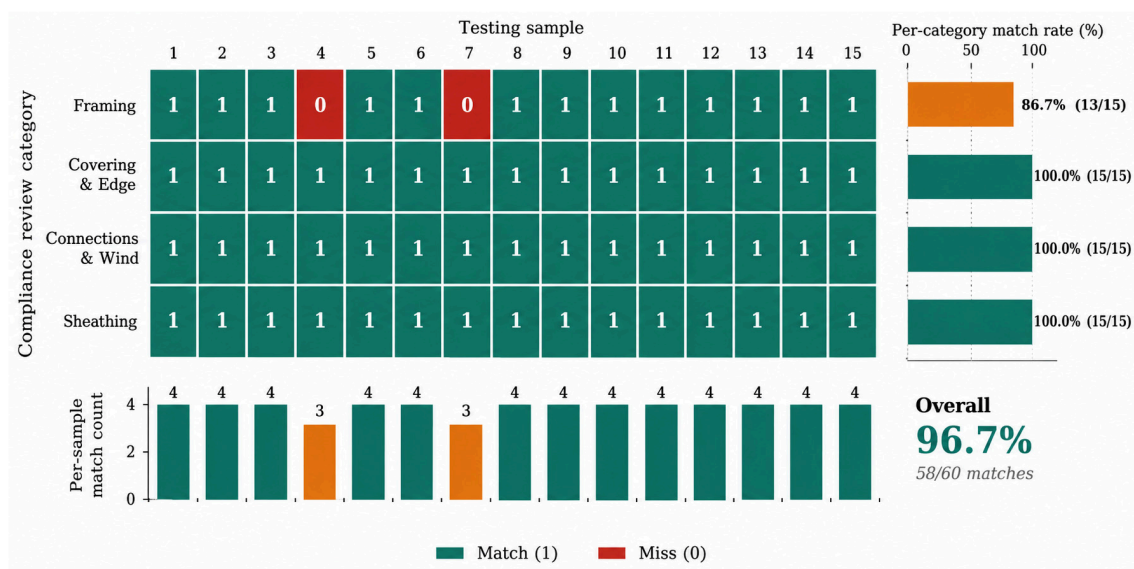


Figure 14. Compliance Review confusion matrix for Mistral small-3.1.

The results for GPT5.4 is very close to Mistral small 3.1. with an overall match rate of 97%.

7.7. F1 Score

The F1 score is the harmonic mean of precision (P) and recall (R) as given in Equation (1), where precision is the share of predicted positives that are correct, and recall is the share of actual positives that are recovered. Because it is a harmonic mean, F1 is

dominated by the lower of P and R; a model with very high precision but low recall (or vice versa) is penalized. F1 ranges from 0 to 1, with 1 indicating perfect classification. The “cumulative mean F1” plotted in Figure 15 below is the running average of per-sample F1 scores after evaluating testing samples 1, 2, . . . , n; it stabilizes as n grows and is therefore a reliable estimate of overall classification quality at the end of the trace.

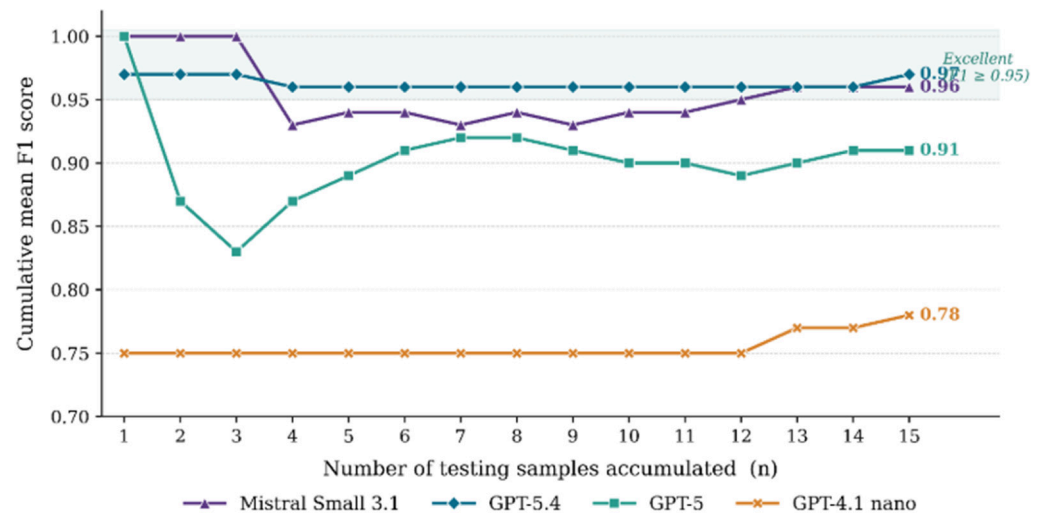


Figure 15. The cumulative F1 score means.

The cumulative-mean F1 trajectories separate the four models into three clear performance tiers. GPT-5.4 and Mistral Small 3.1 occupy the top tier, both converging into the “excellent” band ($F1 \geq 0.95$) by the end of the 15-sample run, with terminal cumulative F1 of approximately 0.97 (GPT-5.4) and 0.96 (Mistral Small 3.1). Their curves are also notably stable: GPT-5.4 fluctuates within a 0.01 corridor across the entire trace, indicating that its per-sample errors are small and uncorrelated, while Mistral Small 3.1 starts at a perfect 1.00 for the first three samples, dips when its first error appears at sample 4, and then steadily climbs back, suggesting that its mistakes are concentrated early in the sequence and that subsequent samples are handled cleanly.

GPT-5 sits in a middle tier, ending at $F1 \approx 0.91$. Its trajectory shows the most volatility; a sharp drop from 1.00 to 0.83 between samples 1 and 3 reflects an early cluster of errors, after which the running average slowly recovers as more correct predictions accumulate. The continuing oscillation between 0.89 and 0.92 in the second half of the trace indicates that errors are not confined to the early samples but recur intermittently, consistent with a model that handles most cases competently but has a persistent failure mode it does not correct for.

GPT-4.1 nano is clearly the weakest model, plateauing at $F1 \approx 0.75$ for twelve consecutive samples before edging up only marginally to 0.78 at the end. The flatness of this curve itself is diagnostic: it implies that errors are distributed roughly uniformly across the test set rather than being driven by a small subset of difficult samples, so additional data would not be expected to lift performance much further. Combined with the visual gap of nearly 0.20 F1 between this model and the top tier, the chart supports recommending GPT-5.4 or Mistral Small 3.1 for this compliance-review task, with GPT-5 acceptable as a fallback and GPT-4.1 nano not suitable for production use without substantial additional training or prompt engineering.

7.8. Multi-Dimensional Model Profile

A multi-dimensional model profile is a vector of normalized scores across qualitatively distinct evaluation axes, typically rendered as a radar (spider) chart that characterizes a

model's behavior jointly, rather than collapsing it to a single accuracy number. Each axis captures a different property of the system. In Figure 15, Avg F1 measures predictive accuracy (harmonic mean of precision and recall, scaled to 0–100); Label consistency measures the stability of class labels across paraphrased or repeated inputs; Narrative depth measures the richness and structure of the model's free-text justifications; Code specificity measures how concretely the model cites the relevant code provisions or sections; and Conservative bias measures the model's tendency to flag items as “Requires Review” rather than “Compliant”, useful when missed violations are costlier than over-review, but penalizing in tasks where over-flagging adds work. Because the axes have different “good” directions (high accuracy is good, but high conservative bias is only good in proportion to risk tolerance), the polygon's shape, not just its area, is what matters.

The radar profiles in Figure 15 separate the four models into three behavioral archetypes. GPT-5.4 and GPT-5 trace nearly identically: near-complete pentagons across the four good-direction axes and both score in the mid-90s on Avg F1, label consistency, narrative depth, and code specificity, and exhibit only mild conservative bias (around 25–30%). They are general-purpose performers that combine accuracy, stable labeling, rich justifications, and concrete code citations without over-flagging. GPT-5.4 finishes marginally ahead on the composite (97.0 vs. 94.0), but the two are practically interchangeable on this evaluation, with the choice between them driven more by latency and cost than by quality.

Mistral Small 3.1 has a distinctly asymmetric profile. Its Avg F1 (96) and label consistency (95) are essentially on par with the GPT-5 family, but it falls sharply on narrative depth and code specificity (both around 65). This is the signature of a model that classifies accurately but justifies tersely; it tells you the right answer without the supporting paragraph or the specific code section that grounds it. For tasks where the label is the deliverable (e.g., feeding a downstream pipeline), Mistral Small 3.1 is competitive; for tasks where reviewers must read and trust the rationale, the lack of depth is a meaningful gap that drags its composite score to 80.2.

GPT-4.1 nano is the outlier in both ability and disposition. It is the only model below 80 on Avg F1, and it sits near 50 on label consistency, narrative depth, and code specificity, meaning it not only mislabels more often but also produces shallow, less reproducible justifications. Most strikingly, its conservative bias spikes to ~95: it compensates for uncertainty by routing decisions to “Requires Review” almost reflexively. This is consistent with the inverted pattern observed in the earlier within-class match-rate chart, where the same model achieved 100% recall on “Requires Review” but only 13% on “Compliant.” Such a profile can be acceptable in safety-critical settings where a false negative is far worse than a false positive, but it shifts work onto human reviewers rather than reducing it, and on every other axis, it underperforms the alternatives by 30–45 points. The composite score of 58.2 reflects this and confirms that GPT-4.1 nano is not a viable substitute for the larger models on this task.

7.9. Future Opportunities in BIM and Computational Design Integration

Although the proposed framework was developed and validated using PDF-based construction drawings, the underlying multi-agent architecture is not limited to document-centric workflows. Future implementations could leverage BIM platforms such as Autodesk Revit, IFC-based data exchanges, and visual programming environments such as Dynamo and Grasshopper to enable direct interrogation of semantic building models. Unlike PDF drawings, BIM models contain structured object-level information, including geometry, material properties, spatial relationships, and metadata, which can be accessed programmatically through application programming interfaces (APIs) and standardized exchange formats. Integrating the framework with BIM environments would eliminate several imme-

diate preprocessing steps, improve data fidelity, and provide richer contextual information for compliance analysis.

Modular agent-based architecture can support existing computational design workflows within BIM environments by integrating with these visually programmed automation tools and Python-based workflows. To achieve this, the proposed framework architecture is extended to interconnect with open data standards such as the Industry Foundation Classes (IFCs). An agentic AI integration with Revit API and Dynamo scripts can facilitate automated parameter extraction, rule execution, and report generation, while IFC-based interoperability would enable deployment across multiple software ecosystems. Python-based automation can support IFC parsing, data transformation, rule execution, and communication between BIM platforms and compliance-review agents. Such interoperability enhances an Agentic AI-driven digital design-review pipeline, enabling iterative compliance assessment during permit submission.

The proposed RAG architecture is well suited for BIM-enabled compliance checking because BIM object attributes can be transformed into embeddings and linked directly to relevant code provisions within a vectorized regulatory knowledge base. Such an approach would support real-time compliance evaluation during the design process, allowing architects and engineers to receive immediate feedback as design decisions are made. Consequently, the framework has the potential to evolve from a post-design review tool into an intelligent design-assistance platform that supports continuous code compliance verification throughout the building lifecycle. This transition represents a promising direction for future research and would further align the framework with contemporary digital design and BIM-enabled project delivery practices within the Architecture, Engineering, and Construction (AEC) industry.

8. Discussion

This research presents the design and evaluation of an agentic AI system, Co-Roof Designer, intended to serve as a collaborative assistant for architecture, engineering, and construction (AEC) students and practitioners, and as a decision-support tool for building officials seeking to accelerate the review of roof and rooftop-structure submissions. The framework couples computer-vision back-ends, YOLO11m-seg and Mask R-CNN, for object detection, classification, and structured data extraction from architectural and engineering PDF drawings, with a large-language-model layer augmented by retrieval-augmented generation (RAG) that grounds compliance reasoning in the Florida Building Code (FBC) 2023. It is important to note that the primary experimental focus of this paper is the LLM-based compliance reasoning component; the vision and RAG subsystems are described as integral parts of the overall architecture, but they have been evaluated separately rather than as a unified end-to-end pipeline. The vision models were evaluated using standard detection and segmentation metrics, including mAP@0.50, mAP@0.50:0.95, precision, recall, F1-score, mask mAP@0.50, and mean mask IoU. Per-class AP was reported for structural components, drawing symbols, and connection assemblies, and the models achieved acceptable detection accuracy on the evaluation dataset. The LLM and RAG layer was evaluated independently on compliance classification against a code baseline. Together, these component-level results support the technical feasibility of the proposed approach, while full end-to-end validation at production scale is reserved for future work.

The system's fitness for production deployment is determined almost entirely by the choice of LLM in the compliance-review layer. Across 15 distinct test projects yielding 60 reference decisions per model, GPT-5.4 and Mistral Small 3.1 clearly emerge as the top tier, posting portfolio mean accuracies of 97.0% and 96.7%, respectively, cumulative-mean F1 scores of 0.97 and 0.96, and per-class agreement rates that comfortably clear the

95% threshold typically associated with regulator-grade automated assistance. GPT-5 sits in a clearly distinct middle band (91.7% accuracy, $F1 = 0.91$), and GPT-4.1 nano is the clear outlier (78.3% accuracy, $F1 = 0.78$), a deficit too large to attribute to model variance and instead indicative of a structural capacity gap on the framing-related provisions of the FBC.

Although GPT-5.4 and Mistral Small 3.1 are statistically tied on headline accuracy, deeper diagnostics reveal meaningful differences in their behavior. GPT-5.4 achieves a perfect severity posture (zero permissive and zero conservative deviations), and its predicted-status distribution (15 Compliant, 45 Requires Review) matches the FBC baseline exactly, indicating strong calibration. Its overall accuracy is 97% rather than 100% because narrative explanations and code-citation phrasing occasionally deviated from the baseline's wording, even when the underlying compliance decision was correct. Mistral Small 3.1 records two conservative deviations, both confined to the Framing category, with 58 of 60 decisions fully aligned. Across qualitative dimensions, however, it scores noticeably lower on narrative depth and code-citation specificity (both $\sim 65/100$), meaning that while its compliance labels are reliably correct, the supporting justifications are terser and less explicitly tied to specific code sections than those produced by GPT-5.4.

These differences translate into concrete deployment guidance. GPT-5.4 is the recommended primary model when reviewer trust depends on the readability of the rationale, for example, when officials must defend the system's recommendations to applicants or use them as the seed for a written determination, and it is also the safer choice where calibration matters, since its predicted-status distribution introduces no aggregate bias. Mistral Small 3.1 is the recommended cost-optimized alternative: with comparable headline accuracy and a conservative-only failure mode bounded to a single category (Framing), it is well-suited to high-volume preliminary screening pipelines in which lighter latency and operating cost outweigh the value of a richly written justification, and where any Framing flag will still be re-examined by a reviewer. The trade-off between the two models is therefore not one of correctness but of operational fit; both meet the accuracy threshold expected of a credible advisory system, and the choice should be made based on inference cost, response latency, and the depth of justification required by the downstream user.

Two further findings have implications beyond the model-selection question. First, the dominant error direction across the model family is more conservative: of the 20 disagreements observed across the three lower-tier models, 15 over-flag baseline-compliant items as Requires Review and only five (concentrated in GPT-5 on Sheathing) under-flag items that should have been routed for review. From a safety standpoint, this is the desirable asymmetry: the system errs toward inviting human review rather than unwarranted clearance, but it does mean that any production deployment should plan for a measurable fraction of the workload to land in reviewer queues even when the underlying drawings are compliant. Second, the categories Connections & Wind and Covering & Edge are saturated at 100% across all four models, suggesting either that the FBC 2023 provisions in those areas are well-aligned with the language patterns these LLMs already encode, or, more likely, that the 15-sample portfolio does not yet contain enough adversarial detailing to stress-test those axes. Future evaluations should therefore enrich the test set with edge-case roof-to-wall connections, complex flashing details, and atypical covering systems to confirm that the apparent ceiling on these categories is real rather than a portfolio artifact. Although the present validation demonstrates promising results across multiple evaluation dimensions, future studies should expand the testing corpus and include additional building systems and regulatory jurisdictions to further assess the framework's generalizability and scalability.

9. Conclusions

This research presented the design and evaluation of Co-Roof Designer, a research-stage agentic AI framework developed to support roof-design review in architecture, engineering, construction, and building-code compliance workflows. The system is intended to assist AEC professionals and building officials by extracting roof-related information from design documents, retrieving applicable code provisions, and generating structured preliminary compliance assessments. The framework integrates computer-vision backends, including YOLO11m-seg and Mask R-CNN, for object detection, instance segmentation, classification, and structured extraction from PDF-based design documents. These outputs are combined with a large language model layer, augmented with retrieval-augmented generation, to ground compliance reasoning in relevant Florida Building Code requirements.

Four large language models were evaluated (GPT-5.4, Mistral Small 3.1, GPT-5, and GPT-4.1 nano) across 15 roof-design case studies comprising 60 reference compliance decisions. Among the evaluated models, GPT-5.4 produced the most accurate and well-calibrated compliance assessments, while Mistral Small 3.1 provided a computationally efficient alternative with comparable performance in selected cases. These results indicate that agentic AI has potential as a human-in-the-loop decision-support tool for preliminary reviews of roof design compliance. However, the findings should be interpreted as evidence of feasibility rather than as validation of a production-ready permitting system or a substitute for professional judgment and authority-having-jurisdiction review.

The study's limitations define the agenda for future work. Although the compliance-reasoning and computer-vision components were evaluated separately, the complete pipeline has not yet been validated as an integrated end-to-end system. Production-scale testing, including measurement of runtime performance, reviewer workload reduction, error severity, and cost of missed or incorrect compliance determinations, is needed before deployment in operational permitting environments. The evaluation was also limited to a single jurisdiction, FBC 2023, and to a restricted set of roof-related drawing types. Broader validation should include additional jurisdictions, ASCE 7 wind provisions, regional amendments, high-wind and wind-borne debris requirements, and additional drawing types such as elevations, structural details, product approvals, and construction specifications.

A further limitation is that the evaluation dataset did not include confirmed outright non-compliant drawings. In the dataset used in this study, the reference decisions were concentrated in the Compliant and Requires Further Review categories, reflecting the nature of the available permit-style submissions. While synthetic violations could be introduced to stress-test the framework, such cases may not reflect the distribution or ambiguity of real permit documents. Future work should therefore develop a curated dataset of confirmed non-compliant drawings and reviewer-verified deficiencies to evaluate the system's ability to detect substantive code violations. In addition, accuracy-based metrics should be supplemented with human-centered and operational metrics, including review time, reviewer confidence, correction effort, false-negative severity, and the consequences of incorrect compliance recommendations.

Finally, extending the framework from document-centric PDF inputs to BIM-native and model-based workflows offers a promising direction for richer, more structured compliance checking. Overall, the results demonstrate the feasibility of using agentic AI to support roof-design code review and suggest that, with further validation, expanded datasets, and human-in-the-loop safeguards, such systems may help reduce the cognitive and operational burden associated with preliminary permitting review.

Author Contributions: N.O.N.: conceptualization; methodology; software architecture; testing and validation, formal analysis; data curation; writing—original draft preparation; writing—review and editing; visualization; supervision; project administration; funding acquisition. O.O.L.: conceptualization; methodology; testing and validation, formal analysis; data management; writing—review and editing; visualization; funding acquisition. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by 2025 FRSA/Earl Blank Florida Roofing and Sheet Metal Contractors Association, Faculty Fellowship Fund, with grant number [DCP-FRSA2025].

Data Availability Statement: All supporting data outside the contents of this article are available upon request.

Acknowledgments: The authors gratefully acknowledge the support of the 2025 FRSA/Earl Blank Florida Roofing and Sheet Metal Contractors Association, Faculty Fellowship Fund for funding this research through [DCP-FRSA2025]. Their generous support made this work possible and is sincerely appreciated. ChatGPT 5.5 was used to search and validate credible citation sources in parts of the literature review. It was also used to generate the image in Figure 1 and it was also used enhanced the legibility of Figures 4–6 which were not originally developed with any GenAI tool.

Conflicts of Interest: Authors have received research grants from 2025 FRSA/Earl Blank Florida Roofing and Sheet Metal Contractors Association, Faculty Fellowship Fund. However, authors declare that there are no Conflicts of Interest pertaining to the conduct of this study.

Abbreviations

The following abbreviations are used in this manuscript:

ACC	Automated Code Checking
AEC	Architecture, Engineering and Construction
AEDR	Architecture and Engineering Design Review
BIM	Building Information Modeling
C	Compliant
CNN	Convolutional Neural Network
FBC	Florida Building Code
FCN	Fully Convolutional Network
FN	False Negative
FP	False Positive
FPN	Feature Pyramid Network
GPT	Generative Pre-trained Transformer
GPU	Graphic Processing Unit
HVHZ	High-Velocity Hurricane Zone
LLM	Large Language Models
MAS	Multi-Agent System/Machine-Aided System
ML	Machine Learning
NADIA	Natural-Language-Based Architectural Detailing through Interaction with AI
NC	Non-Compliant
NLP	Natural Language Processing
NOA	Notice of Acceptance
OCR	Optical Character Recognition
RAG	Retrieval-Augmented Generation
R-CNN	Region-based Convolutional Neural Network
ROI	Region Of Interest
RR	Requires Review
SVM	Support Vector Machine
TAS	Testing Application Standards

TP True Positive
YOLO You Only Look Once

References

1. LNCS Homepage. Available online: <http://www.springer.com/lncs> (accessed on 25 October 2023).
2. Nawari, O. Generalized adaptive framework for computerizing the building design review process. *J. Archit. Eng.* **2020**, *26*, 04019026–1–04019026–17. [\[CrossRef\]](#)
3. Solihin, W.; Eastman, C. Classification of rules for automated BIM rule checking development. *Autom. Constr.* **2015**, *53*, 69–82. [\[CrossRef\]](#)
4. Zhang, J.; El-Gohary, N. Semantic NLP-based information extraction from construction regulatory documents for automated compliance checking. *J. Comput. Civ. Eng.* **2016**, *30*, 04015014. [\[CrossRef\]](#)
5. Dimyadi, J.; Amor, R.; Spearpoint, M. Automated building code compliance checking—Where is it at? In Proceedings of the 19th International CIB World Building Congress, Brisbane, QLD, Australia, 19–23 May 2013.
6. Pan, Y.; Zhang, L.; Zhang, J. AI-driven intelligent construction and smart building systems: State-of-the-art review. *J. Build. Eng.* **2021**, *44*, 102924. [\[CrossRef\]](#)
7. Cheng, J.C.P.; Lu, Q. A review of AI and machine learning applications in the AEC industry. *Adv. Eng. Inform.* **2020**, *45*, 101124. [\[CrossRef\]](#)
8. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016*; IEEE: Piscataway, NJ, USA, 2016; pp. 779–788.
9. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
10. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015*; IEEE: Piscataway, NJ, USA, 2015; pp. 3431–3440.
11. Lin, T.-Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017*; IEEE: Piscataway, NJ, USA, 2017; pp. 936–944.
12. Fuchs, S. *Natural Language Processing for Building Code Interpretation: Systematic Literature Review Report*; Technical Report; University of Auckland: Auckland, New Zealand, 2021.
13. Zhang, R.; El-Gohary, N. Building information modeling, natural language processing, and artificial intelligence for automated compliance checking. In *Research Companion to Building Information Modeling*; Edward Elgar Publishing: Cheltenham, UK, 2022; pp. 248–267.
14. Chen, N.; Lin, X.; Jiang, H.; An, Y. Automated Building Information Modeling Compliance Check through a Large Language Model Combined with Deep Learning and Ontology. *Buildings* **2024**, *14*, 1983. [\[CrossRef\]](#)
15. Liu, X.; Li, H.; Zhu, X. A GPT-based method of Automated Compliance Checking through prompt engineering. In *Towards Sustainable, Smart and Resilient Buildings, Infrastructures and Cities*; European Group for Intelligent Computing in Engineering: London, UK, 2023.
16. Yamusa, M.; Dauda, J.; Ajayi, S.; Saka, A.B.; Oyegoke, A.S.; Adebisi, W.; Jagun, Z.T. *Automated Compliance Checking in the AEC Industry: A Review of Current State, Opportunities and Challenges*; Social Science Research Network: Rochester, NY, USA, 2024.
17. Jung, W.; Lee, G. Slim BIM charts for rapidly visualizing and quantifying levels of BIM adoption and implementation. *J. Comput. Civ. Eng.* **2016**, *30*, 04015072. [\[CrossRef\]](#)
18. Bassir, D.; Lodge, H.; Chang, H.; Majak, J.; Chen, G. Application of artificial intelligence and machine learning for BIM. *Int. J. Simul. Multidiscip. Des. Optim.* **2023**, *14*, 5. [\[CrossRef\]](#)
19. Zhang, J.; Cui, B.; Gao, Y.; Zhang, D. Factors influencing the acceptance of BIM-based automated code compliance checking in the AEC industry in China. *J. Manag. Eng.* **2023**, *39*, 04023036. [\[CrossRef\]](#)
20. Zhang, Z.; Ma, L.; Broyd, T. Rule capture of automated compliance checking of building requirements: A review. *Proc. Inst. Civ. Eng. Smart Infrastruct. Constr.* **2023**, *176*, 224–238. [\[CrossRef\]](#)
21. Alnuzha, M.; Bloch, T. The role of machine learning in automated code checking—a systematic literature review. *J. Inf. Technol. Constr.* **2025**, *30*, 22–44.
22. Peng, J.; Liu, X. Automated code compliance checking research based on BIM and knowledge graph. *Sci. Rep.* **2023**, *13*, 7065. [\[CrossRef\]](#) [\[PubMed\]](#)
23. Qu, K.; Li, K.C.; Wong, B.T.; Wu, M.M.; Liu, M. A survey of knowledge graph approaches and applications in education. *Electronics* **2024**, *13*, 2537. [\[CrossRef\]](#)

24. Guo, T.; Chen, X.; Wang, Y.; Chang, R.; Pei, S.; Chawla, N.V.; Wiest, O.; Zhang, X. Large language model based multi-agents: A survey of progress and challenges. In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), Jeju, Republic of Korea, 3–9 August 2024. Available online: <https://arxiv.org/abs/2402.01680> (accessed on 15 June 2026).
25. Wang, Y.; Liu, Y.; Cai, H.; Wang, J.; Zhou, X. An automated fire code compliance checking jointly using building information models and natural language processing. *Fire* **2023**, *6*, 358. [\[CrossRef\]](#)
26. Zabin, A.; Gonzalez, V.A.; Zou, Y.; Amor, R. Applications of machine learning to BIM: A systematic literature review. *Adv. Eng. Inform.* **2022**, *51*, 101474. [\[CrossRef\]](#)
27. Ghimire, P.; Kim, K.; Acharya, M. Generative ai in the construction industry: Opportunities & challenges. *arXiv* **2023**, arXiv:2310.04427.
28. Nabavi, A.; Ramaji, I.; Sadeghi, N.; Anderson, A. Leveraging Natural Language Processing for Automated Information Inquiry from Building Information Models. *J. Inf. Technol. Constr.* **2023**, *28*, 266–285. [\[CrossRef\]](#)
29. Wang, L.; Ma, C.; Feng, X.; Zhang, Z.; Yang, H.; Zhang, J.; Wen, J. A survey on large language model based autonomous agents. *Front. Comput. Sci.* **2024**, *18*, 186345. [\[CrossRef\]](#)
30. Xi, Z.; Chen, W.; Guo, X.; He, W.; Ding, Y.; Hong, B.; Gui, T. The rise and potential of large language model based agents: A survey. *Sci. China Inf. Sci.* **2025**, *68*, 121101. [\[CrossRef\]](#)
31. Gao, Y.; Xiong, Y.; Gao, X.; Jia, K.; Pan, J.; Bi, Y.; Dai, Y.; Sun, J.; Wang, M.; Wang, H. Retrieval-Augmented Generation for Large Language Models: A Survey. *arXiv* **2024**, arXiv:2312.10997.
32. Fan, W.; Ding, Y.; Ning, L.; Wang, S.; Li, H.; Yin, D.; Chua, T.-S.; Li, Q. A Survey on RAG Meeting LLMs: Towards Retrieval-Augmented Large Language Models. In Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), Barcelona, Spain, 25–29 August 2024.
33. Huang, Y.; Huang, J. A Survey on Retrieval-Augmented Text Generation for Large Language Models. *arXiv* **2024**, arXiv:2404.10981.
34. Wu, S.; Xiong, Y.; Cui, Y.; Wu, H.; Chen, C.; Yuan, Y.; Huang, L.; Liu, X.; Kuo, T.-W.; Guan, N.; et al. Retrieval-Augmented Generation for Natural Language Processing: A Survey. *arXiv* **2024**, arXiv:2407.13193.
35. Lavikka, R.; Kallio, J.; Casey, T.; Airaksinen, M. Digital disruption of the AEC industry: Technology-oriented scenarios for possible future development paths. *Constr. Manag. Econ.* **2018**, *36*, 635–650. [\[CrossRef\]](#)
36. Hu, S.; Huang, Z.; Hu, C.; Liu, J. 3D Building Generation in Minecraft via Large. *arXiv* **2024**, arXiv:2406.08751.
37. Jang, S.; Lee, G.; Oh, J.; Lee, J.; Koo, B. Automated detailing of exterior walls using NADIA: Natural-language-based architectural detailing through interaction with AI. *Adv. Eng. Inform.* **2024**, *61*, 102532. [\[CrossRef\]](#)
38. Gasparetto, A.; Marcuzzo, M.; Zangari, A.; Albarelli, A. A Survey on Text Classification Algorithms: From Text to Predictions. *Information* **2022**, *13*, 83. [\[CrossRef\]](#)
39. Chen, S.; Liu, Y.; Han, W.; Zhang, W.; Liu, T. A Survey on LLM-Based Multi-Agent Systems: Recent Advances and New Frontiers. *arXiv* **2024**, arXiv:2412.17481.
40. Siebert, J. Agentic AI: Multiagent Systems in the Age of Generative AI. Fraunhofer IESE Technical Report, 2025. Available online: <https://www.iese.fraunhofer.de/blog/agentic-ai-multiagent-systems-in-the-age-of-generative-ai/> (accessed on 15 June 2026).
41. Zhou, Y.; Wang, S. Towards the Next Generation of Agent Systems: From Retrieval-Augmented Generation to Agentic Systems. In Proceedings of the VLDB 2025 Workshop on LLMs and Knowledge Graphs, London, UK, 1 September 2025.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.