Smart Agricultural Technology 12 (2025) 101640

Contents lists available at ScienceDirect

Smart Agricultural Technology

journal homepage: www.journals.elsevier.com/smart-agricultural-technology

CottonBot: An AI-driven cotton farming assistant and irrigation advisor using LLM-RAG and agentic AI tools

Deus F. Kandamali [a,*], Wesley M. Porter [c], Erin Porter [d], Alex McLemore [d], Glen C. Rains [a,b]

[a] College of Engineering, University of Georgia, Tifton, GA 31793, USA
[b] Department of Entomology, College of Agricultural and Environmental Sciences, University of Georgia
[c] Department of Crop and Soil Sciences, University of Georgia, Tifton, GA 31793, USA
[d] Abraham Baldwin Agricultural College, Tifton, GA 31793, USA

ARTICLE INFO

Keywords:
Chatbot
AI agents
Model context protocol (MCP)
Retrieval-augmented generation (RAG)
Large language models (LLM)
Soil moisture sensors, IoT sensors
Weather forecasting

ABSTRACT

Digital agriculture is transforming crop management by enabling data-driven decision-making. This study introduces CottonBot, an AI-powered assistant designed to support cotton farmers with comprehensive farming guidelines, including pest management, soil fertilization, weed control, nematode management, and real-time, context-aware, farm-specific irrigation recommendations. Unlike traditional retrieval augmented generation (RAG) based systems that rely solely on static knowledge bases, CottonBot integrates RAG with agentic large language model (LLM) tools, utilizing farm-specific soil moisture sensors and local weather forecast APIs to deliver dynamic, actionable, and field-specific recommendations. The system leverages open-weight LLMs and runs locally through Ollama framework, ensuring cost-efficiency, data privacy, and accessibility in resource-limited settings. The study's experimental setup combined RAG with LLM-based agentic tools. Within the RAG framework, we evaluated multiple embedding models using two retrieval backends: FAISS (Facebook AI similarity search) and Chroma (a vector database). The embedding model, all-MiniLM-L6-v2 paired with FAISS achieved the highest retrieval performance, while Llama 3.1 produced the most faithful and semantically accurate responses. For real-time irrigation support, CottonBot's agentic tool interprets weighted soil moisture data to provide crop-specific recommendations, demonstrating its utility beyond text retrieval. Finally, CottonBot was deployed as a cross-platform mobile application using FastAPI and Flutter for both iOS and Android. Results indicate that CottonBot enhances decision-making support for cotton farmers, leading to more efficient water use and improved crop yields.

1. Introduction

The global demand for food production continues to escalate with population growth, placing immense pressure on agriculture to achieve higher yields with fewer resources. Among the emerging technologies addressing this challenge, artificial intelligence (AI) [1–3] and particularly large language models (LLMs) [4–8] has proven transformative in advancing precision agriculture through improved decision-making and operational efficiency. Cotton production, a vital component of U.S. agriculture, is sensitive to water and nitrogen use efficiency, pest outbreaks, and weed pressure [9]. Each of these can have a significant influence on crop yield and quality. However, many cotton producers still depend on manual field observations, static guidelines, and experiential knowledge, which are often inadequate under fast-changing field conditions.

Although agricultural knowledge is increasingly digitized and available through PDFs, spreadsheets, and web-based resources, farmers frequently struggle to extract actionable insights in a timely manner due to information overload, time constraints, or limited digital literacy. This challenge has sparked interest in conversational AI tools that can deliver timely, personalized advice through intuitive interfaces, commonly known as chatbots. Traditional rule-based chatbots [10–16] have offered limited utility in this space due to their static nature and shallow natural language understanding, making them unsuitable for handling the complexity of modern, domain-specific agricultural queries.

Recent developments in LLMs have enabled generative AI chatbots to perform dynamic, context-aware reasoning across diverse fields

* Corresponding author.
E-mail address: soler.deus@uga.edu (D.F. Kandamali).

https://doi.org/10.1016/j.atech.2025.101640
Received 1 October 2025; Received in revised form 13 November 2025; Accepted 13 November 2025
Available online 14 November 2025

including various agricultural domains. These systems can interpret unstructured queries, support multilingual interactions, and generate responses tailored to the user's context. However, LLMs are limited by the fixed scope of their pre-trained data and are prone to hallucinations when faced with domain-specific or real-time questions such as determining optimal irrigation based on crop stage and current soil conditions. To address these shortcomings, Retrieval-Augmented Generation (RAG) [17–21] framework was introduced. These systems combine LLMs with external structured and unstructured knowledge sources, enhancing factual grounding and contextual relevance at inference time.

RAG agricultural applications have shown promising results across a range of domains, including crop diagnostics, traceability, and advisory services. AgroLLM is a RAG-based chatbot that integrates open-source agricultural documents and uses FAISS (Facebook AI similarity search) similarity search to achieve high performance in contextual relevance [22]. while multimodal systems support inclusive, user-friendly interaction across literacy levels [6,23,24]. Other notable implementations, such as BeefBot [25] and crop-specific optimization bots [7,8], demonstrate significant improvements in relevance, precision, and recall. However, most existing systems remain limited by their reliance on static document corpora and lack integration with real-time environmental data to make real-time decisions, which is a critical need in data-intensive precision agriculture.

To address the limitations of existing agricultural chatbots, this study introduces CottonBot, a domain-specific, AI-driven conversational assistant tailored for cotton farmers in Georgia, USA. CottonBot employs a hybrid architecture that integrates the open-source LLM (Llama3) via the Ollama framework with a RAG pipeline to produce contextually grounded responses. The system does two major things: (1) it retrieves static expert knowledge from the 2024 Georgia cotton production guide [26] (developed by the UGA extension team) to provide comprehensive agronomic assistance, including pest management, fertilization, and crop development best practices, and (2) it leverages real-time, field--specific data through agent tool calling, dynamically interfacing with farm-specific IoT soil moisture sensors and weather forecasting APIs to deliver adaptive irrigation recommendations based on current environmental conditions. To ensure accessibility and scalability, CottonBot is deployed as a mobile application using Flutter and FastAPI, offering a seamless, user-friendly interface. By relying entirely on open-source technologies, the system presents a cost-effective alternative to commercial LLM-based platforms that depend on paid API subscriptions without compromising accuracy, relevance, or responsiveness.

2. Materials and methods

2.1. Data collection and RAG knowledge base construction

This study uses two key data streams to support intelligent, context-aware decision-making in cotton farming: (1) a static knowledge-base for the RAG pipeline which is derived from the 2024 Georgia cotton production guide [26], providing expert agronomic guidance on practices such as fertilization, pest control, weed management, nematode management, irrigation decisions, and overall cotton management guidelines; and (2) dynamic, farm-specific data for real-time irrigation decision obtained from real-time soil moisture readings via RealmFive-installed watermark sensors in the 4D-Project farm called Digital and Data-Driven Demonstration (D.A.T.A) farm on the Abraham Baldwins Agricultural College (ABAC), Tifton Georgia - USA and weather forecasting data retrieved from the OpenWeatherMap APIs. By combining these data modalities, the system delivers both generalized farming advice from RAG and localized, real-time and farm-specific irrigation recommendations from live sensors.

The 4D-farm has 4 fields namely; North Pivot, South Pivot, Front Field and West Field with a total of 12 soil moisture sensors installed and one weather station shared across them as shown in Fig. 1. Both North and south pivots have 4 soil moisture sensors each while the front and south pivots have two each, both represented as white and black markets in Fig. 1.

The Georgia Cotton Production Guide, produced by the University of Georgia Cotton Team with support from the Georgia Cotton Commission, is a comprehensive extension handbook. It provides an in-depth, region-specific synthesis of agronomic best practices for Georgia and the broader southeastern U.S., covering variety selection, planting windows, soil fertility and fertilization, irrigation scheduling, integrated pest management (weeds, insects, diseases/nematodes), and harvest/defoliation.

The handbook's expert-curated content spans both structured (tables, charts, decision thresholds, formulas) and unstructured (narrative) materials addressing plant growth monitoring, insect scouting, pest and disease management, nematode control, irrigation scheduling, and integrated decision-support systems. The document is processed using LangChain framework for loading, chunking, and semantic embedding. The resulting embeddings are indexed in the vector database to enable fast and scalable similarity search. At query time, the system retrieves the most relevant document chunks and passes them through a RAG pipeline, allowing a LLM to generate accurate, localized context-aware responses.

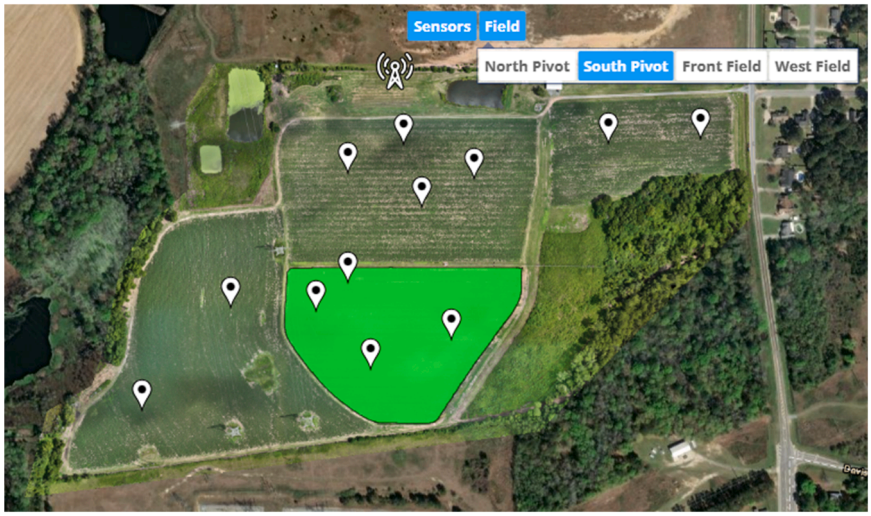


Fig. 1. The (D.A.T.A) 4D-Project Farm at the ABAC showing fields sensors distribution and field weather station.

2.2. Vector embeddings

Following knowledge base construction, the proposed system employs vector embeddings to enable efficient semantic retrieval. Embeddings are numerical representations of text (both from user queries and knowledge base documents) that capture underlying semantic meaning. By projecting text into a high-dimensional vector space, embeddings allow semantically related content to cluster together, even when the exact wording differs.

Experiments conducted in this study employed six (6) open-source embedding models: all-MiniLM-L6-v2 from Sentence-Transformers, jina-embeddings-v2-base-en from Jina AI, intfloat/e5-base and e5-mistral-7b-instruct from HuggingFace, and bge-base-en-v1.5 and bge-m3 from the Beijing Academy of Artificial Intelligence (BAAI). These models were evaluated using two vector databases: Chroma and FAISS, to compare their retrieval performance in terms of semantic similarity and latency. The embedding pipeline consisted of three main tasks: (i) encoding semantically meaningful chunks from domain-specific documents including cotton production guidelines, pest management protocols, and irrigation schedules; (ii) encoding farmer queries; and (iii) retrieving the top-k most semantically similar document chunks using cosine similarity semantic similarity metric.

2.3. Vector embeddings representation

To enable semantic search and context-aware response generation, both user queries and document chunks from the knowledge-base are embedded into a shared d-dimensional vector space by an encoder f: X → Rd. Let f denote embedding model, q the user query, and $\{d_1, d_2, \ldots, d_m\}$ the document chunks. Each text input is then transformed into a dense numerical representation as follows:

$$V_q = f(q) \in R^d, \tag{1}$$

$$V_i = f(d_i) \in R^d \text{ for i } = 1, 2\ldots\ldots, \text{ m} \tag{2}$$

Here, V_q denotes the embedding of the query, and V_i represents the embedding of the ith document chunk. Generally, the embedding vector for any piece of text x is defined as;

$$V_x = f(x) \in R^d \tag{3}$$

These vectors preserve relevant semantic relationships, so semantically similar texts cluster closely (to be located near each other) in the embedding space even when phrased differently.

2.4. Vector databases selection

This study utilizes the Chroma vector database for efficient local similarity search and top-k document retrieval. In our experiments, we compared Chroma with FAISS, a widely used approximate-nearest-neighbor (ANN) indexing library that supports dense-vector indexing and high-performance nearest-neighbor search. Fig. 2 shows an embedding model encoding both document and queries into vectors; the document vectors are stored and indexed in a vector database. At query time, the user query is also embedded using the same embedding model and used to retrieve the nearest document vectors from the vector database. While FAISS excels in speed, scalability, and fine-grained control particularly in resource-constrained environments, Chroma provides a more developer-friendly interface with built-in persistence and seamless integration with LangChain for a mobile app. Although Chroma introduces additional dependencies, its ease of setup and rapid prototyping capabilities made it the preferred choice for this project, especially given the goal of deploying CottonBot as a mobile-friendly application.

2.5. Proposed RAG pipeline

CottonBot uses Ollama framework to run LLM locally without API subscriptions costs, an all-MiniLM-L6-v2 embedding model from Sentence transformers library was used to create embeddings from the text chunks and Chroma database to store vector embeddings for efficient semantic search. In the retriever phase, the knowledge-base contains the 2024 Georgia cotton production guide segmented into 1000-character chunks size with a 100-character overlap to ensure manageable text units. Both documents and queries are encoded using similar embedding model to maintain their semantic consistency.

For the augmentation phase, user query is encoded into an embedding using the same embedding model, and RAG retriever fetches the top-5 most similar document chunks. These chunks are concatenated to form a context string, which is combined with the query and passed to the llama3 model via the Ollama framework. Llama3, prompted as an agricultural expert, generates a response grounded in the retrieved context, ensuring relevance to farming and irrigation topics. Fig. 3 shows the proposed end-to-end RAG architecture for both retrieval and generation phases.

2.6. Experiment setup

To evaluate suitable reasoning model for our RAG pipeline integrated with agentic API tools, we compared seven (7) open-weight models available in the Ollama framework based on their; architecture, parameter count, context window, embedding size, quantization format, and capabilities (See Table 1). Models like LLaMA 3.1 (with 8B parameters) and DeepSeek-r1 (7B) stand out with extended context lengths of up to 131,072 tokens, enabling richer multi-document reasoning and long-context retrieval, crucial for agricultural domain tasks involving lengthy PDF files and historical data. The Mistral (7.2B) and LLaMA 3.1 models also support tool calling, making them suitable for agentic workflows. All models are quantized in 4-bit formats (e.g., Q4_0, Q4_K_M) to enable fast inference on edge devices with limited memory. Licensing varies across models, with permissive options like MIT, Apache, and Meta's open license ensuring flexibility in deployment. Ultimately, model selection balances capability, context handling, and compute efficiency, with LLaMA 3.1, Phi-3, and Mistral emerging as

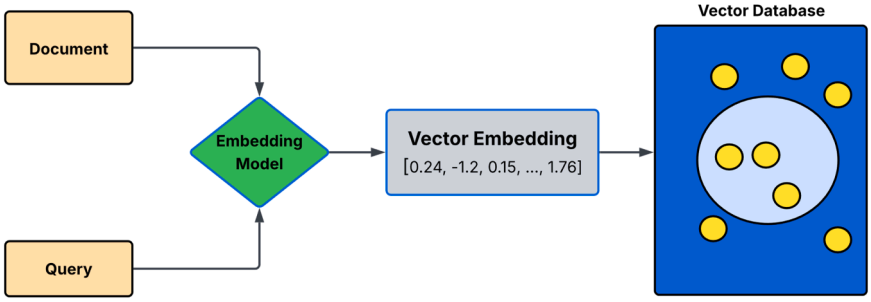


Fig. 2. An embedding model creating vector embeddings from a document and query.

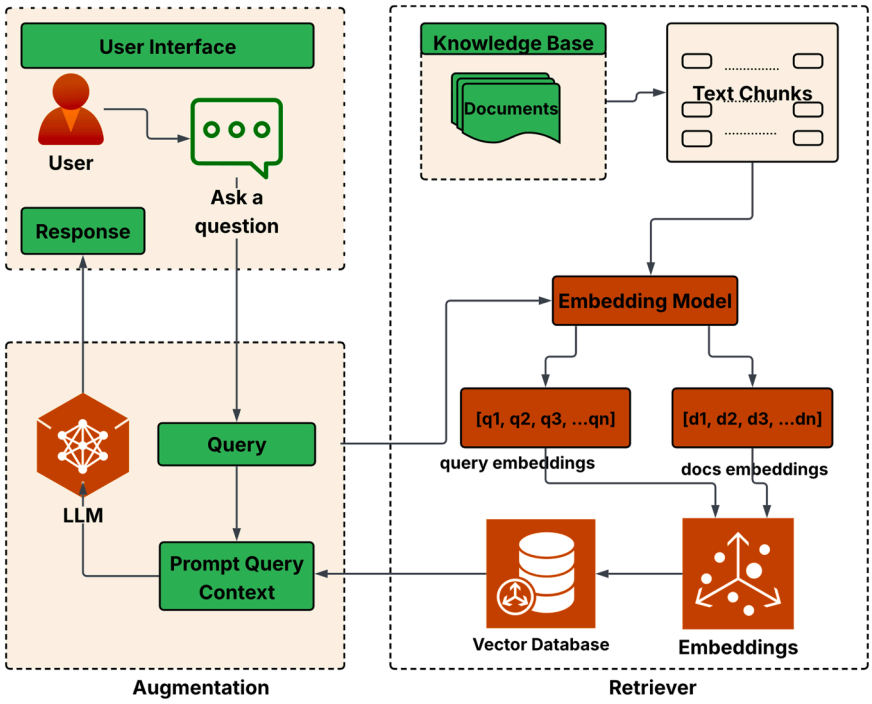


Fig. 3. An End-to-End RAG pipeline used in CottonBot showing the user interface, the retriever and augmentation processes.

Table 1
Comparisons of Ollama open-weight LLMs for RAG + Tools integration.

Model	Architecture	Parameters	Context Length	Embedding Length	Quantization	Capabilities	License
Deepseek-r1	Qwen2	7.6B	131072	3584	Q4_K_M	Completion	MIT
Llama3	Llama	8.0B	8192	4096	Q4_0	Completion	Meta
Llama2	Llama	6.7B	4096	4096	Q4_0	Completion	Meta
Llama3.1	Llama	8.0B	131072	4096	Q4_K_M	Completion, Tools	Meta
Phi3	Phi3	3.8B	131072	3072	Q4_0	Completion	Microsoft
Mistral	Llama	7.2B	32768	4096	Q4_0	Completion, Tools	Apache
Gemma3	Gemma3	4.3B	131072	2560	Q4_K_M	Completion, Vision	Gemma

strong candidates for robust, locally hosted RAG agents.

2.7. Prompt engineering strategy

To enhance the quality and reliability of responses from the LLM, we designed a custom prompt template with the objectives; 1) to minimize hallucinations by anchoring responses to retrieved context, and 2) to reinforce cotton farming domain relevance and accuracy by aligning output with expert-validated best practices from Georgia 2024's cotton farming handbook guide. Furthermore, we tailored the model responses to maintain a concise, actionable tone appropriate for farmer use. This prompt strategy, helps incorporating both the user query and the most relevant document chunks retrieved via vector similarity search, ensuring that the LLM operates within a well-defined context, promoting factual consistency and task-oriented reasoning grounded in the cotton farming domain.

2.8. Agentic tool integration for irrigation decisions

Accurate irrigation decision requires representing crop water uptake across the active rooting zone; therefore, CottonBot applies dynamic depth-weighting factors that evolve with crop growth stages. These weights adjust the contribution of each monitored depth (8-in, 16-in, 24-in) to the soil-moisture index according to DAP. Early in the season, shallow moisture readings dominate due to limited root depth, whereas deeper layers increasingly influence recommendations as the root system develops. Table 2 outlines the adaptive weighting schedule used for cotton at the 4D-Farm, derived from UGA Extension guidelines and field observations. This method aligns with established guidance from the University of Georgia extension report [27], which highlights that equal weighting across depths can misrepresent crop water status—leading to premature water stress early in the season and potential over-irrigation as roots extend deeper.

Beyond the traditional RAG pipeline that retrieves static agronomic knowledge from external sources, CottonBot integrates three dynamic agentic LLM tools: (1) a soil-moisture monitoring tool, (2) a weather-forecast tool, and (3) an irrigation-decision tool, in order to leverage

Table 2
Depth specific root-zone weights for cotton used in CottonBot.

DAP	Weighted depth			Crop and root development stages
	8 inches	16 inches	24 inches	
0	0.00	0.00	0.00	Seed not yet germinated, no active roots
1-12	0.80	0.20	0.00	Early vegetative stage; roots shallow
13-23	0.60	0.30	0.10	Rapid root expansion begins
24-27	0.50	0.30	0.20	Mid-vegetative to early flowering
28-34	0.50	0.25	0.25	Full flowering; deeper uptake increases
35-160	0.40	0.30	0.30	Peak boll development to maturity

real-time field data and generate site-specific irrigation guidance. These tools enable the system to perform live reasoning over streaming farm data by calling backend FastAPI functions only when required. Through this architecture, CottonBot autonomously aggregates multi-depth average weighted soil-moisture readings, forecasts future soil-moisture values, and evaluates irrigation trigger thresholds in real time as follows:

2.8.1. Soil moisture monitoring tool

This tool connects the real-time IoT-enabled soil-moisture sensor readings from the South Pivot field of the D.A.T.A farm at 8-, 16-and 24-inch depth for each sensor. The irrigation logic computes per-depth sensor averages, then calculates final weighted averages using days after planting (DAP)-cotton specific root zone weights that vary by growth stage. A separate irrigation recommendation tool consumes these outputs to make decisions. Uses an existing hybrid LSTM soil moisture prediction model [28] to forecast the field-level final weighted averages (the DAP-aware, depth-weighted soil-moisture index) from historical observations. The model outputs daily predictions up to 7 -days ahead with over 98% R^2, allowing the irrigation decision tool to anticipate needs and schedule irrigation precisely and proactively.

2.8.2. Weather forecast tool

This tool fetches localized weather forecasts data (e.g., temperature, humidity, precipitation) up to 7 days ahead using OpenWeather APIs. The retrieved forecasts are used to augment the irrigation logic tool by signaling weather conditions that may change irrigation recommendation for example, deferring irrigation when rain is likely above a configurable threshold, increasing irrigation under persistent heat/drought stress, or alerting users to upcoming rainfall events that could affect field operations. The tool returns a structured payload for the decision engine and triggers user alerts when policy thresholds are met.

2.8.3. Irrigation decision tool

Combines real-time soil-moisture profiles and the DAP-aware final weighted average with the 7-day weather forecast and the LSTM soil-moisture predictions to produce crop-specific recommendations. For cotton, a soil water tension threshold of 45 kPa is used: if the weighted index $\geq$ 45 kPa → Irrigate; otherwise → Wait. Final decisions are refined by looking ahead up to 7 days—incorporating predicted soil-moisture and expected rainfall—to schedule or defer irrigation so recommendations align with seasonal crop development and root distribution, while avoiding under- or over-irrigation.

This tool-calling framework equips CottonBot with agent-like behavior, enabling the system to dynamically reason with live sensor and weather data rather than relying solely on pre-embedded knowledge. This architecture supports more context-aware, actionable recommendations tailored to real-world conditions on the farm. Fig. 4 presents the irrigation decision pipeline for the South Pivot field at the ABAC D.A.T.A. Farm, integrating IoT soil-moisture sensors, historical weather/soil-moisture data, and a decision engine, with an LLM-powered interface delivering field-specific "Irrigate/Wait" recommendations to users.

2.9. Mobile application architecture

The system is built using a modular, multi-layered (frontend and

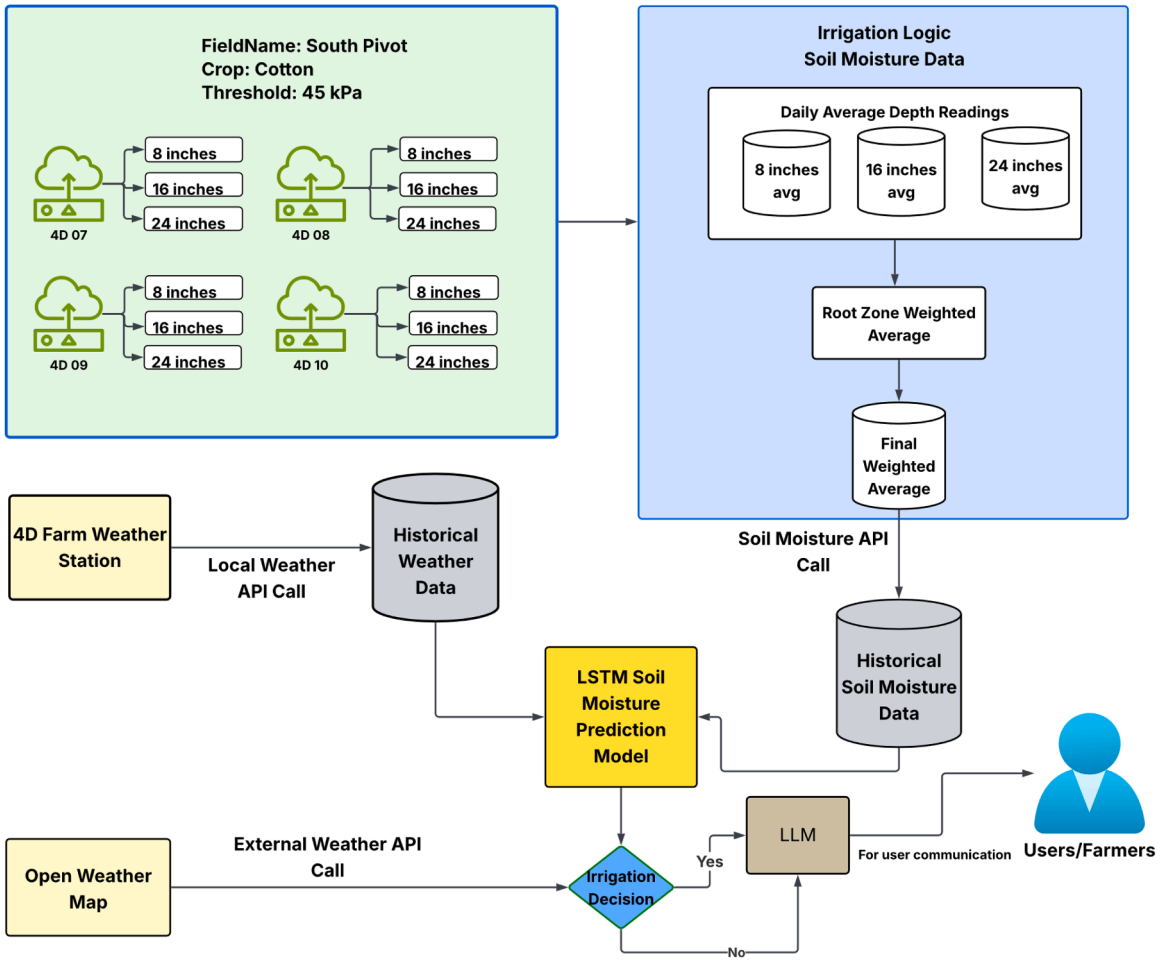


Fig. 4. Cotton irrigation decision framework from the South Pivot field showing irrigation logic pipeline from farm sensors to the farmer.

backend layers) architecture to ensure scalability, updatability and cross-platform usability. The system is deployed via a FastAPI python backend framework and accessed through a front mobile interface developed using Flutter framework implemented in dart programming language. Authentication is handled separately using a Google firebase database. The mobile-app is designed to provide farmers with an interactive and user-friendly experience. Fig. 5 presents application architecture, linking the user to the internal details of implementation.

The frontend layer is built using Flutter mobile-app framework, allowing cross-platform deployment from a single codebase across Android, iOS, and the web. It serves as the primary interface for farmers to: submit text-based queries and receive real-time responses. The Flutter app includes a chat-style user interface with dynamic response rendering, integrates an HTTP client for seamless communication with FastAPI backend framework, linking user and the RAG system, and supports local caching of query and response history to provide basic offline fallback functionality in areas with limited connectivity.

Fig. 6 — parts (a), (b), and (c) — displays the Android mobile application interface, showing the welcome and authentication screens, which are designed to provide a clean user experience and secure onboarding process.

2.9.1. Data privacy and security

In order to ensure the security of data transfer between the users and the system, all data exchanges between the Flutter frontend and FastAPI backend occur through secure HTTPS (TLS/SSL). Sensor identifiers are anonymized and stored locally, and no raw data are transmitted to third-party servers. The application supports optional offline inference via locally cached embeddings and weather data, ensuring user privacy even under intermittent connectivity. Data handling follows general data protection regulation (GDPR)-like principles emphasizing minimal data retention and user consent. Furthermore, Google firebase is employed as the backend authentication service to securely manage and store user credentials, ensuring strong data protection, compliance with modern security standards, and preservation of user privacy.

3. RAG performance evaluation

To evaluate the performance of our CottonBot RAG pipeline, we assessed both the retriever and generator components using standard information retrieval and natural language generation metrics.

3.1. Retriever evaluation metrics

The retriever's role is to return a ranked list of document chunks most relevant to a given user query. We evaluated its effectiveness using three commonly used metrics: mean reciprocal rank (MRR), Recall@k and Precision@k.

MRR evaluates the rank position of the first relevant document in the retrieved list. It reflects how early a relevant document appears in the top-k results. A higher MRR indicates that relevant documents are ranked near the top, which is critical for ensuring a useful context is passed to the generator.

$$\mathrm{MRR} = \left(\frac{1}{\mathrm{N}}\right)\sum_{\mathrm{i}=1}^{\mathrm{N}}\frac{1}{\mathrm{rank_i}} \tag{4}$$

N is the number of queries, ranki is the rank position of the first relevant document for the ith query.

Recall@k measures how often the top k retrieved document chunks contain relevant information needed to answer the query, while Precision@k measures the proportion of the top-k retrieved document chunks that are actually relevant to the query. While k refers to the top retrieved documents for a given query, which can be specified as k-nearest neighbors.

$$\mathrm{Recall@k} = \frac{\text{Number of relevant documents in top k}}{\text{Total number of relevant documents in the collection}} \tag{5}$$

$$\mathrm{Precision@k} = \frac{\text{Number of relevant documents in top k}}{\mathrm{k}} \tag{6}$$

3.2. Generator Evaluation Metrics

To assess the output quality of the LLM in the RAG generator phase, we evaluated three key metrics: cosine similarity, faithfulness, and average generation time.

Cosine similarity was used to quantify the semantic relationship between the LLM-generated response and the ground truth answer. Let V_r denote the embedding vector of the generated response and V_g the embedding of the ground truth. Cosine similarity measures the angle between these vectors, disregarding their magnitude, and yields a value between 0 and 1, where higher values indicate stronger semantic alignment.

$$\mathrm{Sim}(V_r,\ V_g) = \frac{V_r \cdot V_g}{\|V_r\| \cdot \|V_g\|} = \frac{\sum_{i=1}^{n} V_{ri} \cdot V_{gi}}{\sqrt{\sum_{i=1}^{n} V_{ri}^2} \cdot \sqrt{\sum_{i=1}^{n} V_{gi}^2}} \tag{7}$$

In this study, faithfulness was measured through manual evaluation by comparing the generated answer (Y_i) to the retrieved context (C_i).

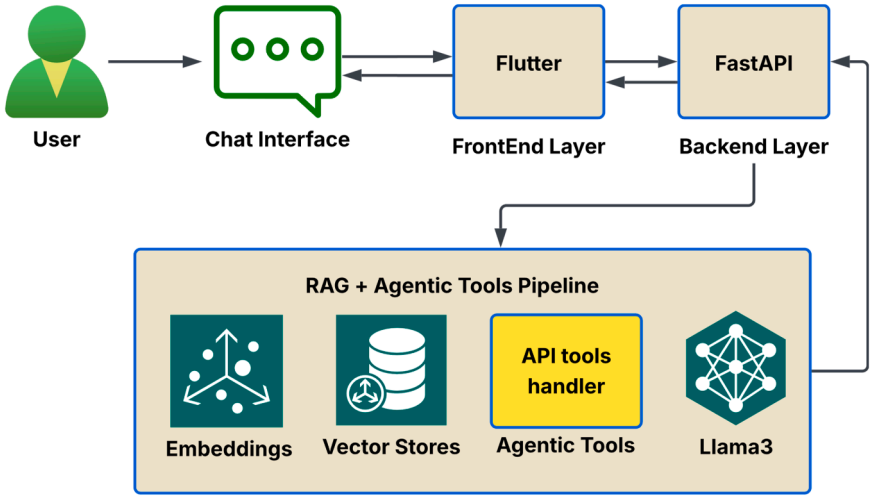
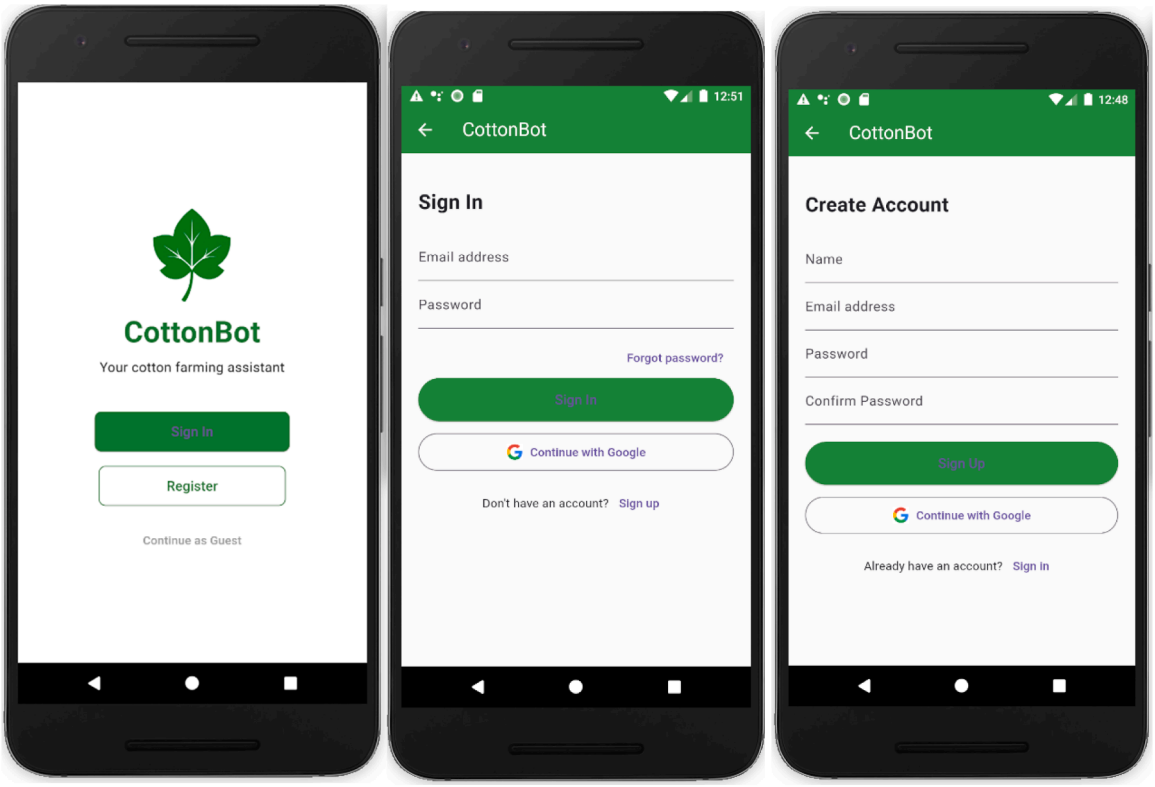


Fig. 5. CottonBot mobile application architecture illustrating the end-to-end interaction flow.



(a) (b) (c)

Fig. 6. CottonBot UI design showing welcome screen (a) and authentication screens for sign-in (b) and sign-up (c).

Annotators assigned a score between 0 and 1 to each response, based on how accurately it aligned with the source context. The final faithfulness score is the average across all N queries:

$$\text{Faithfulness} = \frac{1}{N}\sum_{i=1}^{N}\text{Score}(Y_i,\ C_i) \tag{8}$$

Lasty, average generation time (in seconds), was recorded for each LLM to assess latency and runtime efficiency—critical factors for real-time and mobile deployments.

3.3. End-to-end system testing

For RAG testing purposes, we developed a web-friendly user interface (UI) using the Streamlit Python framework to evaluate the full chatbot system in an interactive environment. As shown in Fig. 7, this prototype allows users to query the system and observe LLM-generated responses in real time. Users can then provide feedback based on their experience allowing necessary modifications and improvements. This enables iterative testing before finally deploying CottonBot into the mobile application.

Fig. 8 (a) and (b) shows the real Android mobile application deployment screens used during initial testing as we continue refining user experience, addressing technical aspects, and resolving maintenance issues. The application will transition to full-scale deployment following comprehensive on-farm validation and once the required backend infrastructure is in place to support large-scale farmer interactions and real-time query processing.

4. Results

4.1. RAG performance evaluation

In evaluating the RAG performance of the CottonBot assistant, we divided the evaluation into two components: (1) RAG retriever performance, and (2) LLM generator performance. To assess the retriever stage of the RAG pipeline, we evaluated six widely used open-source embedding models across two vector store configurations: Chroma and FAISS. Each model was tested on its ability to retrieve relevant context passages from chunked domain-specific documents. Performance was measured using MRR, Recall@5, and Precision@5 (see Table 3).

When using chroma, the all-MiniLM-L6-v2 model (SentenceTransformers) achieved the highest overall performance, with an MRR of 0.8715, Recall@5 of 0.9375, and Precision@5 of 0.9175. This was closely followed by bge-base-en-v1.5 (MRR: 0.8750) and e5-mistral-7b-instruct (MRR: 0.8610). Other models such as bge-m3 and e5-base showed slightly lower MRRs (0.8455 and 0.8125, respectively), but still maintained competitive recall and precision, indicating strong overall retrieval capability.

Table 4 shows LLM performance evaluation during the RAG generator phase. Following retrieval, the top-k chunks (k=5) from FAISS were passed to seven instruction-tuned LLMs for final response generation. We evaluated the generators using Faithfulness, Cosine Similarity, and Average Generation Time per query (in seconds). Faithfulness was measured using a reference-aligned factual consistency score, and semantic similarity was computed using cosine similarity between generated responses and reference answers.

The best-performing model was Llama3.1 (8B), achieving a Faithfulness score of 0.9681 and Cosine Similarity of 0.9523, with an average generation time of 6.2 seconds. Llama3 and Deepseek-r1 followed closely. In contrast, Llama2 (7B) performed significantly worse,

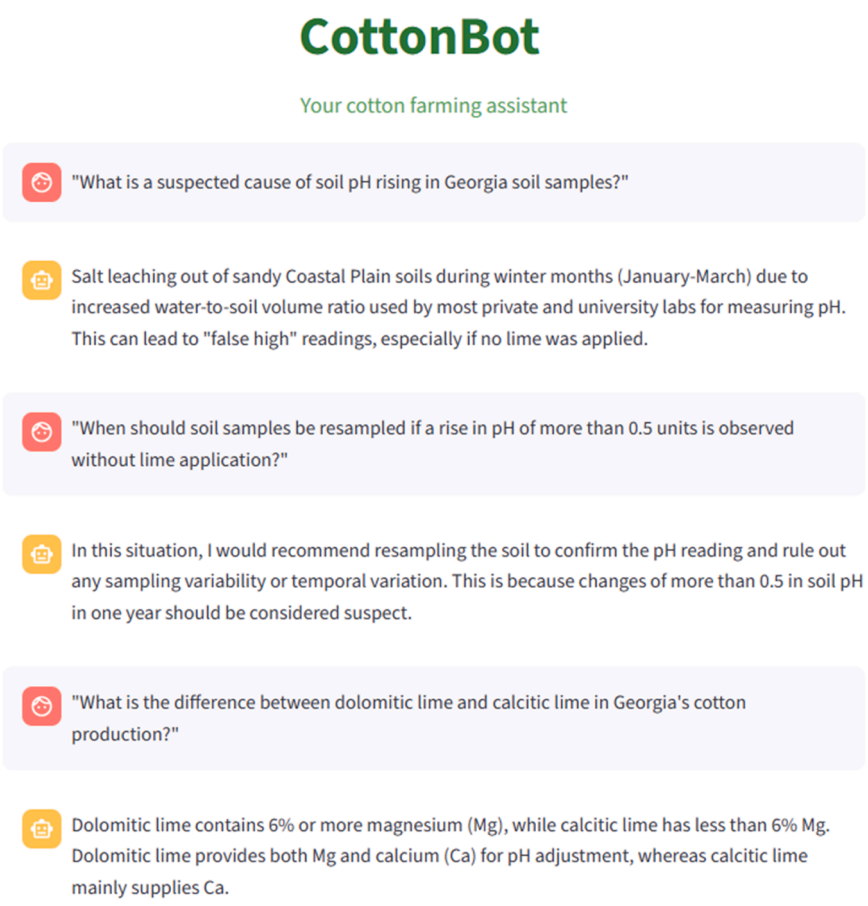


Fig. 7. Streamlit UI for testing and evaluating the performance of the system.

especially in faithfulness (0.8088), highlighting the limitations of older LLMs in grounded generation. Models like Phi3 (3.8B) and Gemma3 (4.3B) offered moderate performance at faster speeds, making them suitable for edge deployments with latency constraints.

4.2. Agentic tools evaluation

The dynamic agentic tools in CottonBot rely on factual, real-time numerical data including; soil moisture sensor readings and weather forecasts, to generate actionable recommendations. Unlike the RAG component, which retrieves context from textual sources, these tools perform deterministic evaluations based on predefined thresholds. The irrigation recommendation tool uses a cotton rootzone weighted average of soil moisture tension across depths and compares it to a crop-specific threshold (45 kPa). If the computed value exceeds this threshold and no rainfall is expected, the system generates a direct recommendation. Here is a sample generated LLM response for such case:

Irrigation Recommendation: *"Apply 0.75 inches of irrigation within the next 24 hours*". The LLM provide further reasoning, to support its responses.

Reasons: *"Current soil moisture tension exceeds the recommended threshold of 45 kPa for cotton. No rainfall is forecasted in the next 48 hours. Weighted root zone tension is 52.3 kPa. Cotton is likely experiencing moderate water stress. Applying irrigation will improve crop health and yield potential during this critical growth stage."*

Fig. 9 compares the one-day-ahead soil-moisture tension forecasts produced by CottonBot against both the manual baseline estimation and the actual sensor-measured values. The dotted line indicates the 45 kPa irrigation threshold used to trigger irrigation events. CottonBot's predicted curve closely follows the ground-truth measurements, whereas the manual heuristic exhibits larger deviations. Fig. 10 quantifies these results, showing that CottonBot achieves lower mean absolute error (MAE) < 1.0 compared to 1.4 MAE for manual estimates method. These findings demonstrate that CottonBot provides more accurate and consistent short-term irrigation guidance than traditional manual methods based on lower MAE, even prior to full on-field deployment. Although both methods have shown 100% agreement (on whether to irrigate or not) with sensor-based decisions for 1 day-ahead prediction, due to their predicted kPa falling inside the threshold (45kPa) value, this is only when it comes to short term prediction since the manual estimation is limited to only one day prediction while CottonBot can predict up to 7 days ahead with up to 98% R^2 using LSTM + XGBoost method.

In summary, the RAG evaluation demonstrated that combining high-performing embedding models with optimized vector databases significantly improves retrieval quality, which is essential for accurate, context-aware responses. Meanwhile, the integration of agentic tools—such as real-time weather and soil moisture analysis—enables CottonBot to deliver factual, data-driven recommendations beyond text-based retrieval. Together, these components showcase the system's ability to blend semantic understanding with actionable field intelligence, forming a robust foundation for intelligent agricultural decision support.

Although CottonBot's irrigation logic and agentic reasoning tools have been validated computationally, large-scale on-field validation remains ongoing. Planned experiments at the ABAC Tifton 4D-Farm will assess real-world impacts on irrigation frequency, soil-water recovery rate, and yield improvement during the 2026 growing season. These studies will benchmark CottonBot's recommendations against traditional farmer practices to quantify water savings and yield gains.

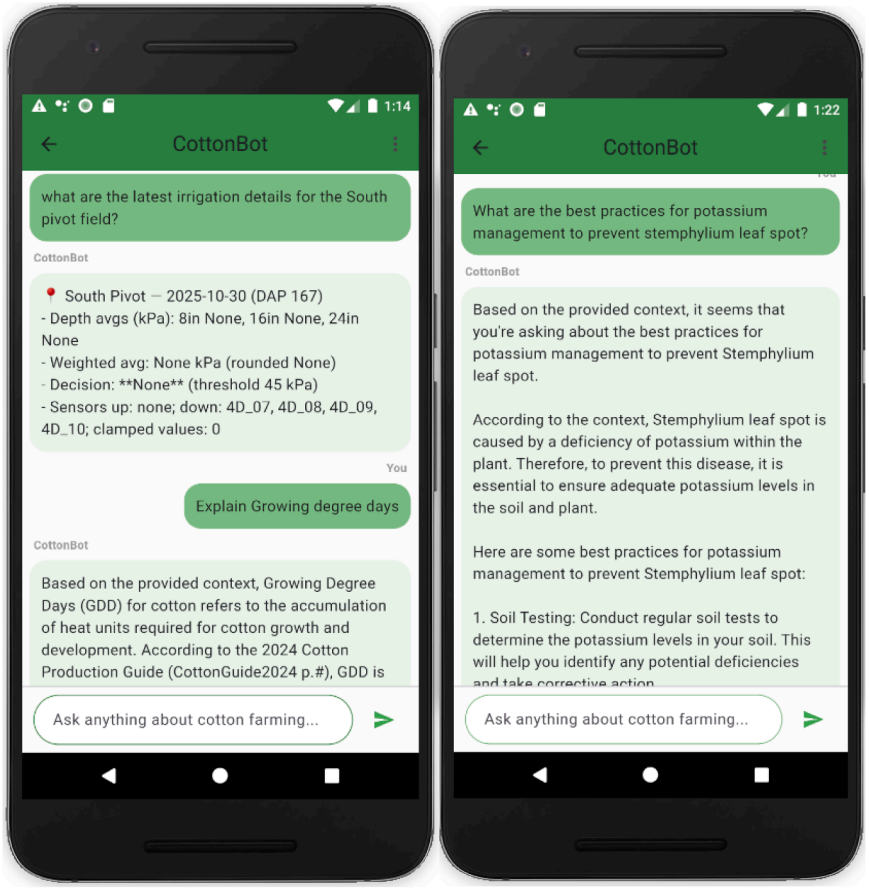


Fig. 8. Real CottonBot Android application deployment screens built using the Flutter framework.

Table 3
Embedding models comparative performance evaluation for RAG retriever evaluation using FAISS and Chroma embeddings.

Embedding Model	Chroma			FAISS		
	MRR	Recall@5	Precision@5	MRR	Recall@5	Precision@5
all-MiniLM-L6-v2	0.8715	0.9375	0.9175	0.8855	0.9476	0.9285
jina-embeddings-v2-base-en	0.8252	0.9052	0.9015	0.8332	0.9114	0.9193
intfloat/e5-base	0.8125	0.8945	0.8845	0.8225	0.9045	0.8963
BAAI/bge-base-en-v1.5	0.8750	0.9045	0.9045	0.8805	0.9095	0.9110
e5-mistral-7b-instruct	0.8610	0.8715	0.8375	0.8716	0.8934	0.8645
BAAI/bge-m3	0.8455	0.8700	0.8300	0.8653	0.8860	0.8605

Table 4
LLM performance evaluation for the RAG generator phase.

Model	Parameters	Faithfulness	Cosine Similarity	Time(s)
Llama3.1	8B	0.9681	0.9523	6.2
Llama3	8B	0.9617	0.9517	6.2
Llama2	7B	0.8088	0.7983	4.5
Phi3	3.8B	0.8792	0.8492	2.3
Mistral	7.2B	0.8945	0.8977	5.3
Deepseek-r1	7.6B	0.9412	0.9388	6.4
Gemma3	4.3B	0.8529	0.8641	3.4

5. Discussion and future works

This study evaluated the performance of CottonBot by analyzing both the retriever and generator components of its RAG architecture, as well as the effectiveness of integrated agentic tools for real-time decision support. Results show that FAISS consistently outperformed Chroma across all embedding models, yielding higher MRR, Recall@5, and Precision@5. Among the tested models, all-MiniLM-L6-v2 delivered the best overall retrieval performance with FAISS, effectively balancing efficiency and accuracy. Larger models such as bge-m3 and e5-mistral-7b-instruct also benefited from FAISS, particularly in improving early relevance ranking and semantic coverage.

In the generation phase, Llama3.1 achieved the highest faithfulness and cosine similarity scores, confirming the effectiveness of modern instruction-tuned models in producing grounded responses. While smaller models like Phi3 and Gemma3 offered faster response times, they exhibited a noticeable drop in factual alignment and semantic coherence. The lower performance of Llama2 further highlights the architectural and training improvements introduced in newer model versions. Despite achieving high faithfulness and cosine similarity, LLM-based systems can still produce hallucinated or partially inaccurate statements when queries fall outside the retrieved context. CottonBot

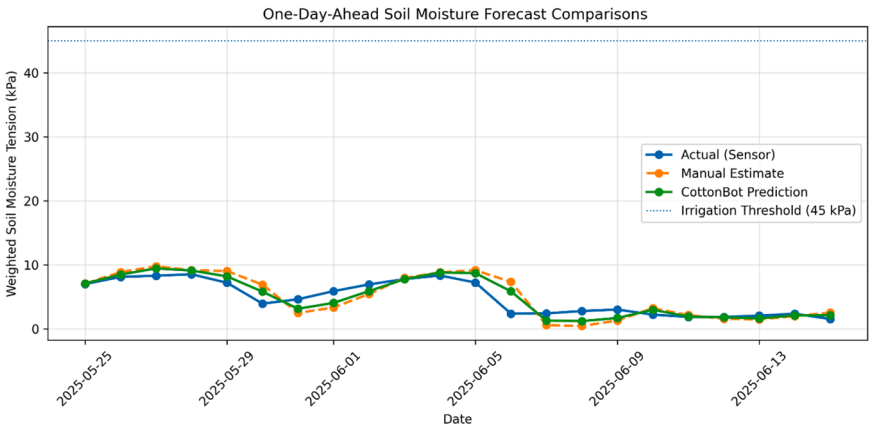


Fig. 9. Soil moisture predictions comparing CottonBot and an existing manual estimation method against the actual sensor readings (ground truth) for the South Pivot field.

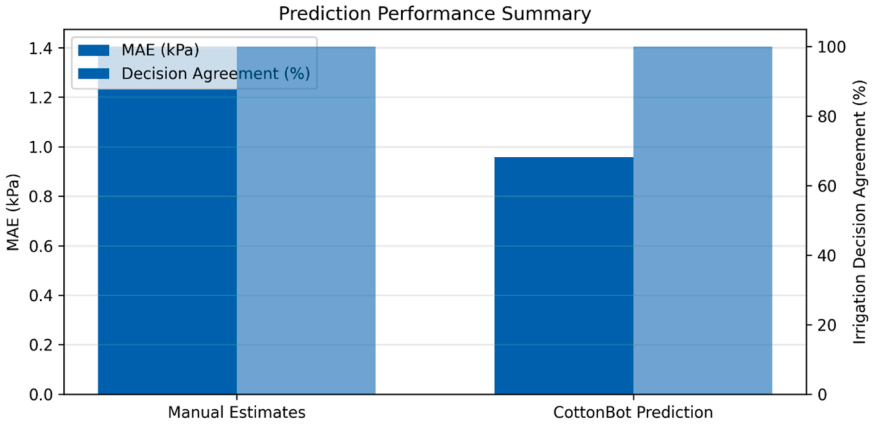


Fig. 10. Performance results summary between the manual estimation method and the CottonBot predictions.

mitigates this risk using a RAG pipeline that grounds every response in retrieved context chunks and employs prompt templates emphasizing citation fidelity. A secondary logging layer records retrieval overlaps and model-generated confidence signals to support qualitative error auditing and future automated evaluation. Future releases will incorporate automated fact-checking and LLM-as-a-judge mechanisms to further reduce misinformation risk.

Beyond the RAG pipeline, CottonBot's agentic tools play a critical role in extending the system's capabilities from document retrieval to real-time, data-driven recommendations. The integration of weather forecasts and soil moisture readings, particularly through the irrigation recommendation tool—enables CottonBot to generate precise, actionable insights rooted in sensor data and crop-specific thresholds. For example, cotton-specific logic based on a 45 kPa soil tension threshold and dynamic root zone weights ensures expertized irrigation advice aligned with growth stages.

Compared to recent studies evaluating LLM-based chatbots in agriculture, such as AgroLLM [22], BeefBot [25] and others [6–8,23,24] rely on static knowledge bases, limiting their ability to support dynamic, real-time decision-making. In contrast, this study advances the field by integrating both RAG and agentic tools to enable context-aware, data--driven recommendations using real-time environmental inputs. Additionally, our system minimizes dependency on costly API subscriptions by utilizing open-source language models, whereas many existing solutions rely heavily on proprietary models like OpenAI GPT or Google Gemini. By running all components locally, we further ensure data privacy and autonomy, avoiding reliance on cloud-based third-party infrastructure and maintaining full control over decision logic and sensitive farm data.

These findings demonstrate that CottonBot offers a robust hybrid approach to intelligent agricultural decision support. The synergy between RAG and agentic reasoning provides both semantic understanding and factual grounding, forming a scalable and trustworthy foundation for real-world deployment in smart farming applications.

Future work aims to expand the scope of CottonBot from cotton specific to an Agri-Bot that supports additional crops; peanuts and corn, as well as livestock. This can be achieved by extending the existing knowledge base with expert-curated peanut and corn handbooks, along with livestock monitoring resources. We also plan to incorporate a voice recognition engine for verbal queries and introduce an automated feedback feature. For real-time irrigation recommendations, we plan to improve the irrigation decision model by training the soil moisture prediction model with larger and more diverse datasets, making it more robust and precise for irrigation scheduling. Furthermore, we plan to perform on-field trials to monitor onsite performance of our proposed method for the 2026 farming season as the part of our future work.

6. Conclusions

This research presents CottonBot, an AI-powered assistant for cotton farming that integrates RAG, real-time sensor data, and agentic reasoning tools to provide actionable and personalized agronomic recommendations. We evaluated how different embedding models perform when paired with vector indexing methods such as Chroma and FAISS, in combination with LLMs. The system employs the Llama 3.1 model, deployed locally through Ollama, to generate accurate and domain-

grounded responses. Experimental results highlight that the quality of the retriever directly influences the reliability and relevance of the generated outputs.

Furthermore, beyond a static RAG, CottonBot leverages dynamic tools for weather forecasting and soil moisture monitoring, enabling real-time, data-driven irrigation recommendations based on crop-specific thresholds and root zone weights. This hybrid approach blends semantic reasoning with factual precision, supporting real-time irrigation decision-making for cotton growers.

The system is designed using open-source technologies, including; FastAPI, Ollama, and Flutter ensuring that it remains cost-effective, scalable, and suitable for deployment in rural or resource-constrained environments. By making expert-level agronomic guidance accessible via a conversational interface, CottonBot represents a step toward democratizing precision agriculture, improving input efficiency, and empowering smallholder farmers with AI-driven insights.

Funding

This research was funded by the United States Department of Agriculture (USDA) – National Institute of Food and Agriculture (NIFA), under Award No. 2023-68016-39403.

Statement of usage of generative AI

During the preparation of this work the authors used GPT 4.0 (Generative AI-assisted language model) to rephrase text drafted by authors to enhance the readability of the article. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the content of the article.

CRediT authorship contribution statement

Deus F. Kandamali: Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Wesley M. Porter:** Writing – review & editing, Visualization, Validation, Supervision, Resources, Project administration, Methodology, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Erin Porter:** Writing – review & editing, Visualization, Validation, Supervision, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Alex McLemore:** Writing – review & editing, Visualization, Validation, Supervision, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Glen C. Rains:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization.

Declaration of competing interest

The authors declare no conflicts of interest.

Acknowledgements

The authors sincerely appreciate the support of the U.S. Department of Agriculture, National Institute of Food and Agriculture (USDA-NIFA), for funding this research. The authors also wish to acknowledge the University of Georgia Extension Cotton Team for providing domain-specific datasets and expert-curated materials used in developing the cotton production knowledge base. We also extend our gratitude to Langchain and HuggingFace communities for open-source AI libraries.

Data availability

We are currently developing an online data repository for the real-time sensor data collected from the D.A.T.A farm, which will be shared as soon as it becomes available. The domain-specific document used in the RAG framework are publicly accessible through the University of Georgia Extension Service at: https://fieldreport.caes.uga.edu/wp-content/uploads/2025/08/AP-124-4_2.pdf.

References

[1] J. Ekanayake, L. Saputhanthri, E-AGRO: Intelligent chat-bot. IoT and artificial intelligence to enhance farming industry, Agris -line Papers Econ. Inf. 12 (1) (Mar. 2020) 15–21, https://doi.org/10.7160/aol.2020.120102.
[2] P. Druva, AI Based Farmer's Assistance Chatbot, Int. J. Res. Appl. Sci. Eng. Technol. 11 (5) (May 2023) 6832–6839, https://doi.org/10.22214/ijraset.2023.53180.
[3] D. Sawant, A. Jaiswal, J. Singh, P. Shah, AgriBot - An intelligent interactive interface to assist farmers in agricultural activities, in: 2019 IEEE Bombay Section Signature Conference (IBSSC), 2019, pp. 1–6, https://doi.org/10.1109/IBSSC47189.2019.8973066.
[4] H. Chia, A.I. Oliveira, P. Azevedo, Implementation of an intelligent virtual assistant based on LLM models for irrigation optimization, in: 2024 8th International Young Engineers Forum on Electrical and Computer Engineering (YEF-ECE), 2024, pp. 94–100, https://doi.org/10.1109/YEF-ECE62614.2024.10624819.
[5] S.K. Dam, C.S. Hong, Y. Qiao, and C. Zhang, "A Complete Survey on LLM-based AI Chatbots," Jun. 2024, [Online]. Available: http://arxiv.org/abs/2406.16937.
[6] J. Benzinho, et al., LLM Based Chatbot for Farm-to-Fork Blockchain Traceability Platform, Appl. Sci. 14 (19) (Oct. 2024), https://doi.org/10.3390/app14198856.
[7] M. Balpande, K. Mahajan, J. Bhandarkar, G. Borse, S. Badjate, AI Powered Agriculture Optimization Chatbot Using RAG and GenAI, in: 2024 IEEE Silchar Subsection Conference (SILCON2024), 2024, pp. 1–6, https://doi.org/10.1109/SILCON63976.2024.10910462.
[8] B. Paneru, B. Thapa, B. Paneru, Leveraging AI in ayurvedic agriculture: A RAG chatbot for comprehensive medicinal plant insights using hybrid deep learning approaches, Telematics Inf. Rep. 16 (Dec. 2024), https://doi.org/10.1016/j.teler.2024.100181.
[9] K. Jabran, B.S. Chauhan, Cotton production, Wiley, 2020.
[10] K. Deepika, V. Tilekya, J. Mamatha, T. Subetha, Jollity Chatbot- A contextual AI Assistant, in: 2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT), 2020, pp. 1196–1200, https://doi.org/10.1109/ICSSIT48917.2020.9214076.
[11] F. Castagna, N. Kökciyan, I. Sassoon, S. Parsons, and E. Sklar, "Computational Argumentation-based Chatbots: a Survey," 2024. [Online]. Available: https://chat.openai.com/.
[12] V. Bouras, et al., Chatbots for Cultural Venues: A Topic-Based Approach, Algorithms. 16 (7) (Jul. 2023), https://doi.org/10.3390/a16070339.
[13] E. Adamopoulou, L. Moussiades, An Overview of Chatbot Technology. IFIP Advances in Information and Communication Technology, Springer, 2020, pp. 373–383, https://doi.org/10.1007/978-3-030-49186-4_31.
[14] W. Saad Nsaif, H.M. Salih, H.H Saleh, and B.T Al-Nuaimi, "Chatbot development: framework, platform, and assessment metrics," vol. 27, 2024, [Online]. Available: www.isres.org.
[15] A. Følstad, et al., Future directions for chatbot research: an interdisciplinary research agenda, Computing 103 (12) (Dec. 2021) 2915–2942, https://doi.org/10.1007/s00607-021-01016-7.
[16] C. Ouaddi, L. Benaddi, A. Souha, A. Jakimi, A comparative and analysis study for recommending a chatbot development tool, in: 2024 International Conference on Global Aeronautical Engineering and Satellite Technology (GAST), 2024, pp. 1–6, https://doi.org/10.1109/GAST60528.2024.10520754.
[17] Y. Gao et al., "Retrieval-augmented generation for large language models: a survey," Dec. 2023, [Online]. Available: http://arxiv.org/abs/2312.10997.
[18] P. Zhao et al., "Retrieval-augmented generation for ai-generated content: a survey," Feb. 2024, [Online]. Available: http://arxiv.org/abs/2402.19473.
[19] A. Vizniuk, G. Diachenko, I. Laktionov, A. Siwocha, M. Xiao, J. Smoląg, A Comprehensive Survey of Retrieval-Augmented Large Language Models for Decision Making in Agriculture: Unsolved Problems and Research Opportunities, J. Artif. Intell. Soft Comput. Res. 15 (2) (Dec. 2024) 115–146, https://doi.org/10.2478/jaiscr-2025-0007.
[20] Z. Jiang et al., "Active retrieval augmented generation." [Online]. Available: https://github.com/.
[21] S. Es, J. James, L. Espinosa-Anke, S. Schockaert, and E. Gradients, "RAGAS: automated evaluation of retrieval augmented generation." [Online]. Available: https://platform.openai.com.
[22] D.J Samuel, I. Skarga-Bandurova, D. Sikolia, and M. Awais, "AgroLLM: connecting farmers and agricultural practices through large language models for enhanced knowledge transfer and practical application".
[23] H. Nikam, et al., Enhancing agricultural practices with AgroDialogue: A chatbot framework, in: 2024 IEEE Pune Section International Conference (PuneCon), 2024, pp. 1–11, https://doi.org/10.1109/PuneCon63413.2024.10895872.
[24] V. Piyathilake, T. Dilni, R. Pushpananda, L. De Silva, Y. Zaheed, Towards a conversational ai chatbot to assist farmers in disease detection, in: M.F. Santos, J. Machado, P. Novais, P. Cortez, P.M. Moreira (Eds.), Progress in Artificial Intelligence, Springer Nature Switzerland, Cham, 2025, pp. 206–217.

[25] Z. Zhang, C.-A. Wilson, R. Hay, Y. Everingham, and U. Naseem, "BeefBot: harnessing advanced LLM and RAG techniques for providing scientific and technology solutions to beef producers." [Online]. Available: https://github.com/guidance-ai/guidance.
[26] C. Hand et al., "2024 Georgia Cotton Production Guide," 2024. Accessed: Oct. 29, 2025. [Online]. Available: https://fieldreport.caes.uga.edu/publications/AP124-4/2024-georgia-cotton-production-guide/.
[27] H. Grubbs, S. Thompson, and W. Porter, "The Importance of Weighting Soil Moisture Sensor Depth Data for Making Irrigation Decisions," Tifton, Jul. 2025.
[28] D.F. Kandamali, et al., Hybrid LSTM Method for Multistep Soil Moisture Prediction Using Historical Soil Moisture and Weather Data, Agri. Eng. 7 (8) (Aug. 2025), https://doi.org/10.3390/agriengineering7080260.