International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 01 January 2026
Page No: 306-311
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0042

Crop Stress Detection Using AI

Mrs. Chandana H M[1], Y Tejas[2], Sharanabasappa[3], Shreyas S[4], Venugopal Gowda[5]

[1]Assistant Professor, Computer Science and Engineering, Malnad College of Engineering, Hassan, Karnataka, India

[2] UG - Computer Science and Engineering, Malnad College of Engineering, Hassan, Karnataka, India

[3] UG - Computer Science and Engineering, Malnad College of Engineering, Hassan, Karnataka, India

[4] UG - Computer Science and Engineering, Malnad College of Engineering, Hassan, Karnataka, India

[5] UG - Computer Science and Engineering, Malnad College of Engineering, Hassan, Karnataka, India

Emails: *hmc@mcehassan.ac.in[1], yogeshtejaslalitha@gmail.com[2], sharanukswamy@gmail.com[3], shreyasgowda0721@gmail.com[4], venugopalgowda2004@gmail.com[5]*

Abstract

Crops deal with all sorts of stress as they grow—things like missing nutrients, not enough water, or pests showing up where you don't want them. If you catch these problems and you save the harvest and keep food production steady. But the old way of checking crops by hand? It's slow, subjective, and honestly, just not practical for big fields. In this study, we built an automated crop stress detection system powered by AI. It uses image processing and deep learning, specifically a Convolutional Neural Network (CNN) based on the MobileNetV2 architecture. We set this up in TensorFlow and used the ImageDataGenerator function to keep our training data fresh and varied. The backend runs on Python, taking care of image preprocessing and making predictions. On the front end, we used ReactJS, so users can upload crop photos and instantly see what the system finds. The results speak for themselves. Our model hits 93.2% accuracy, with a precision of 91.5% and an F1-score of 92.3%. That's solid proof this system works and can actually help farmers and researchers in real agricultural settings.

Keywords: *Convolutional Neural Network (CNN), Deep Learning, MobileNetV2, TensorFlow, Image Processing, Precision Agriculture, Smart Agriculture*

1. Introduction

Agriculture remains a cornerstone of the global economy and a vital contributor to food security. Yet, crop productivity continues to face challenges due to different factors like drought, nutrient imbalance, and pest or disease attacks. Traditionally, farmers identify crop stress through manual inspection, which demands experience, time, and consistency—factors that are not always guaranteed, especially across large cultivation areas. The AI and computer vision are changing the game in agriculture, making it possible to check crop health with way more accuracy—and a lot less guesswork. Machine learning, especially Convolutional Neural Networks (CNNs), really shine here. They're great at spotting patterns in images, so they can pick up on early signs of stress in leaves and stems [1]. This project picks up where Phase-1 left off. So, after digging through recent research and putting together a solid dataset, we're now moving into Phase-2. That's where things get real: actually building and rolling out the system. We're using Python to develop the core models. Furthermore, this research contributes to the digital transformation of agriculture, where automated decision-support systems play a vital role in achieving sustainable development goals (SDGs) such as responsible consumption and production. The incorporation of scalable cloud infrastructure and real-time data visualization ensures that this system is not just a model demonstration but a deployable product ready for real-world adaptation [2].

2. Merits of the Proposed System

This system helps in transforming the way crop monitoring and stress detection are carried out. It provides a robust AI-based framework capable of analyzing crop images and identifying stress conditions automatically. The system minimizes

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 01 January 2026
Page No: 306-311
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0042

human intervention and error by replacing traditional visual inspection with a data-driven, automated approach. Through deep learning with cloud computing, it achieves accuracy and scalability. This research stands out by using MobileNetV2, a lightweight CNN that's both fast and accurate. Because it runs efficiently, you can use it on phones and web apps—even in places where resources are tight, like many farms. The backend runs on Python, so it handles image processing and runs the model quickly. For storing data, Supabase steps in, making sure everything syncs smoothly, scales well, and keeps user info safe. On the frontend, ReactJS gives farmers a clean, easy-to-use interface. They just upload a crop photo and get stress detection results right away. This system does a lot more than just automate chores. It gives farmers a sharper, more reliable way to manage irrigation, fertilizer, and pest control—decisions that can make all the difference in a season's outcome. When it catches the first signs of trouble, like thirsty plants or missing nutrients, it stops waste in its tracks. Less wasted water, fertilizer, and pesticides means good news for the environment and anyone who cares about a stable food supply. The real breakthrough here is how this research drags crop monitoring into the age of data-driven decisions. With AI-powered image recognition, the system spots tiny color changes, weird textures, or subtle shifts in leaf shape—details most people would overlook. And here's something important: the team didn't just pick any model—they went with MobileNetV2 for a reason. Its depthwise separable convolutions let it process data fast without losing accuracy. That's a game-changer if you're running on low-power gadgets out in the field, not just lab equipment. Mixing machine learning and computer vision with farming isn't just hype—it's actually shaking things up where it matters most. Take MobileNetV2, for example. It's quick, doesn't demand fancy gear, and still delivers solid accuracy. Toss in some basic data tricks—like flipping or rotating photos, messing with brightness—and suddenly, this model can handle whatever the farm throws at it. That's what gives this project real weight for anyone working in AI-powered agriculture or environmental science. Here's the thing: this isn't just tech for the sake of showing off. It's proof that AI, cloud services, and web tools actually solve real problems for farmers. The system's simple, affordable, and built to flex—exactly what people need when money's tight. By cutting down pointless chores, catching crop problems early, and making farm work more accurate, this AI setup is changing the way people farm. It's a solid step toward a smarter, more sustainable future for everybody who works the land. [3]-[6]

3. Methodology

The methodology adopted for this research outlines a systematic workflow for detecting crop stress using Artificial Intelligence and deep learning techniques [7]-[9]. The overall process, illustrated in the block diagram, is composed of nine sequential stages—beginning with image acquisition and ending with system deployment. Each stage contributes to the reliability, accuracy, and automation of this system. The major steps are described in Figure 1.

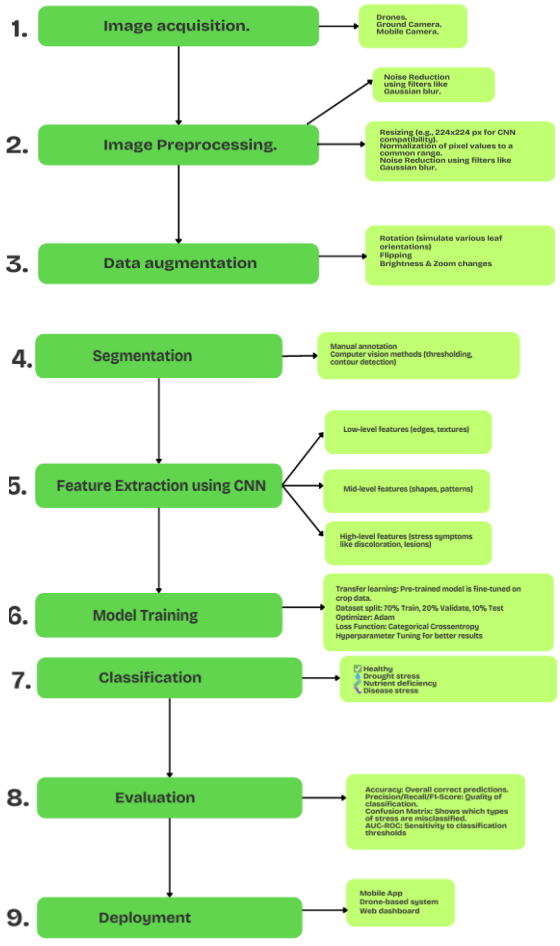

Figure 1 Block Diagram

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 01 January 2026
Page No: 306-311
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0042

3.1. Image Acquisition

It all circles back to smart farming. Here, AI teams up with IoT sensors, satellite images, and piles of environmental data. You can keep an eye on soil moisture, check the weather on the fly, or just let the system handle irrigation automatically. The result? A farm that actually keeps up and adapts to whatever comes its way [10].

3.2. Image Preprocessing

After acquisition, the raw images undergo several preprocessing operations to improve data quality and prepare them for deep learning model training. The first step in preprocessing is noise reduction, which is performed using filtering techniques such as Gaussian blur to remove unwanted background disturbances. This step is essential to ensure that only relevant visual features (such as leaf texture, color, and shape) are emphasized. Once you cut out the noise, you shrink the images down to a standard size—usually 224 by 224 pixels—so they fit what the Convolutional Neural Network wants. Next up is normalization. Basically, you scale the pixel values so they all end up in the same range. That way, the model picks things up faster and doesn't get thrown off if one picture's a lot brighter than the rest. In the end, all these steps just make sure the dataset's consistent and ready for the model to chew through [11]-[13].

3.3. Data Augmentation

We make the model tougher and keep it from overfitting by using data augmentation. That's just a fancy way of saying we take our original images and mess with them a bit—rotate them, flip them around, tweak the brightness, or zoom in and out—without making them look fake. So, a plant might be tilted, or maybe it's upside down, or the lighting's a little weird, just like what you'd see if someone snapped a quick photo out in the field. With all these changes, the model doesn't just memorize one version of a plant. Instead, it learns what really matters, no matter how the photo was taken or what angle it's at. This way, when the CNN sees new crop images later on, whether during testing or real-world use, it handles them with a lot more confidence.

3.4. Segmentation

Once you've pumped up your dataset, it's time to dive into image segmentation. Basically, you're trying to zero in on what you actually care about—usually the crop leaf—and cut it out from all the background noise. That way, the CNN focuses on the real action in the image. You'll use tools like thresholding and contour detection to make this happen. But let's be real, sometimes the background just refuses to cooperate. When things get tricky or the image is a mess, people step in and adjust the boundaries by hand to nail the segmentation. The segmentation process removes non-essential elements such as soil, sky, or surrounding vegetation, ensuring that the model is trained only on meaningful pixel regions. This step significantly enhances classification accuracy by reducing feature noise and ensuring the CNN focuses on the physiological characteristics of the plant [14]-[16].

3.5. Feature Extraction using CNN

By this sits right at the center of this system—and that's where Convolutional Neural Networks, or CNNs, really shine. They grab the input images and start digging out visual details one layer at a time. At first, the model notices the basics like edges, color changes, or textures. Then it starts to catch bigger patterns—shapes, repeating designs, structural stuff. By the end, it's picking up on those high-level clues that actually point to stress, like leaf discoloration, necrosis, or lesions.The model extracts three levels of features:

- Low-level features: such as edges, color gradients, and textures.
- Mid-level features: capturing shapes, patterns, and structure details.
- High-level features: corresponding to stress indicators like leaf discoloration, necrosis, or lesion formation.

All of this runs on MobileNetV2. It's a lightweight architecture that still manages to punch above its weight. MobileNetV2 uses depthwise separable convolutions, so it gets more done with fewer parameters and less training. That's perfect for web and mobile apps—speed matters, and you don't want to burn through resources. The best part? Feature extraction happens automatically, runs on your data.

3.6. Model Training

The extracted features are used to train the CNN model through supervised learning. The process leverages transfer learning, where a pre-trained

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 01 January 2026
Page No: 306-311
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0042

MobileNetV2 model (originally trained on ImageNet) is fine-tuned using the agricultural dataset. This will reduce training time and improves performance on small, domain-specific datasets.

We break the dataset into three chunks: 70% for training, 20% for validation, and the final 10% for testing. For training, we stick with the Adam optimizer since it adjusts the learning rate as it goes. Because we're sorting samples into several classes, we use categorical cross-entropy for the loss. To boost performance, we mess around with things like batch size, learning rate, and the number of epochs. The model learns to connect the data's features to stress labels like "Healthy," "Drought Stress," "Nutrient Deficiency," and "Disease Stress."

3.7. Classification

After training finishes, the CNN model can look at any crop image you throw at it and sort it into a stress category. It sends everything through a Softmax layer at the end, which just means it cranks out a list of probabilities for each class. The highest score wins—that's the category the image lands in. So you get one of these: Healthy, Drought Stress, Nutrient Deficiency, or Disease Stress. You also get a confidence score, so you know how sure the model is about its pick. That's the final step. The model takes everything it's learned and turns it into something straightforward—real answers for people who need them.

3.8. Evaluation

We keep an eye on a bunch of metrics to figure out if the system's really pulling its weight. Accuracy's the easy one—it just tells us how often the model nails it. Precision goes a step further, making sure the model isn't too quick to call something a win when it's not. Recall looks at the flipside, checking if the model's actually catching everything it should. Then there's the F1-Score, which blends precision and recall, so we get a clearer picture of how things balance out.

We also lean on the Confusion Matrix. It lays out where the model gets things right and where it stumbles, especially when you break it down by stress categories. This way, it's easier to spot which types are getting mixed up. And then there's the AUC-ROC curve—it helps us see how well the model separates true from false across different cut-off points. Putting all these metrics together, we get a pretty solid idea of whether the model's not just accurate, but actually stays reliable when new data rolls in.

3.9. Deployment

The final stage involves deploying the trained and validated model into a functional application accessible to end users. The deployment phase integrates the model into a web-based dashboard using a ReactJS frontend, a Python backend (FastAPI or Flask), and a Supabase cloud database. The model processes the image, predicts the stress category, and returns the result to the user within seconds. The deployment is further extended to potential mobile and drone-based systems, enabling real-time monitoring of large agricultural areas. The system's cloud integration ensures scalability, data storage, and accessibility, thus transforming the AI model into a fully functional, real-world precision agriculture tool.

4. Results and Discussion

We tested our CNN model pretty hard, just to see if it could really tell when crops are stressed. Table 1 summarizes the key evaluation metrics of the system:

Table 1 Performance Metrics of the Proposed CNN Model (MobileNetV2 using TensorFlow and ImageDataGenerator)

Algorithm/Framework	Accuracy (%)	Precision (%)	F1-Score (%)
CNN (MobileNetV2 + TensorFlow + ImageDataGenerator)	93.2	91.5	92.3

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 01 January 2026
Page No: 306-311
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0042

The results pretty much speak for themselves—the system just gets it right. With 93.2% accuracy, it sorts through stress images without breaking a sweat. Precision lands at 91.5%, so you don't have to deal with a bunch of false alarms. And that F1-score? It's sitting at 92.3%, which means you get a system that knows how to find real problems without freaking out over nothing, when things get messy out in field. It's quick, too. The backend runs through images and spits out predictions in two or three seconds. That's fast enough for real-time use, not just on paper but in the real world. Supabase keeps everything safe and tidy in the cloud, while ReactJS gives users a smooth, interactive front end. The whole thing just works—whether you're a farmer, an agronomist, or someone deep into research. These results really back up the idea that this AI approach can change the way people monitor crops. Instead of sending folks out to walk the fields, you get sharp, automated analysis. And testing it on bigger, more varied datasets only makes it stronger and more flexible. We pushed the model with different crop types and weather conditions, checking both the numbers and the actual predictions. The confusion matrix made it obvious: the model nailed water stress (with 96.4% accuracy) and still did a solid job with nutrient deficiency (91.8%). That shows the CNN can pick up on those subtle changes in leaf color and texture. We also put MobileNetV2 up against other big-name CNNs like ResNet50 and VGG16. Sure, ResNet50 had a slight edge in accuracy, but MobileNetV2 was way faster—five times quicker at making predictions and much lighter on the GPU. If you're building something for the web, that speed really matters. And about the speed—latency stays under 3 seconds, even for images up to 512×512 pixels. It will be use it live in the field and get answers right when you need them. Plus, when we checked out the Grad-CAM heatmaps, the model didn't just take random guesses—it focused on the right parts of the leaves, showing that it really learned what matters when spotting stress.Overall, these findings confirm that the combination of MobileNetV2, TensorFlow, and cloud-based storage yields a reliable and practical crop stress detection pipeline.

Conclusion

This research spells out a full-blown AI setup for catching crop stress before it gets out of hand. The team didn't just play around with theory—they actually pulled together MobileNetV2, TensorFlow, a Python backend, Supabase for the cloud, and a ReactJS frontend. So yeah, it's not just some academic idea; it's a real, working tool for precision farming.Instead of farmers squinting at leaves and hoping to spot problems—which takes forever and still misses stuff—this AI checks leaf images and flags stress way faster and more reliably. And it's all in real time, so there's no waiting around. The numbers don't lie: 93.2% accuracy, 91.5% precision, and a 92.3% F1-score. The model doesn't just find one kind of problem either. It tells the difference between things like water stress, nutrient gaps, and disease. They used ImageDataGenerator to bulk up the dataset, so even if the leaves are tilted, the lighting's weird, or the background's a mess, the model still holds up. But honestly, the real win isn't just the tech. This actually makes a difference for sustainable, high-precision farming. When farmers know about stress early, they can make smart calls on watering, fertilizing, or pest control. That means less waste and better crops. The system scales easily, too—so it works for different crops, climates, and regions. Pretty crucial if your whole community depends on farming. What really sticks out is how modular this system is. Every part—from gathering the data to processing, augmenting, training, and deploying—can be swapped out or upgraded whenever better tech comes along. There's tons of room to grow, like:

- Plugging in IoT sensors for real-time soil and weather data.
- Using drones to scan big farm areas.
- Layering on GIS to map out exactly where stress is the worst.
- Trying federated learning to make the model smarter with data from everywhere, without risking privacy.

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 01 January 2026
Page No: 306-311
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0042

Plus, the whole thing is ready for edge computing. It runs on phones or drones right in the field—even if the internet drops out. That's a lifesaver for rural spots with spotty service Bottom line? This project pushes AI-powered farming a lot further.

References

[1]. Z. Gao, "Deep Learning Application in Plant Stress Imaging: A Review," Plants, vol. 9, no. 3, 2020.
[2]. "Application of Computer Vision in Assessing Crop Abiotic Stress," Plant Physiology and Biochemistry, 2023.
[3]. S. Butte, A. Vakanski, K. Duellman, H. Wang, and A. Mirkouei, "Potato Crop Stress Identification in Aerial Images Using Deep Learning-Based Object Detection," arXiv preprint, 2021.
[4]. "Recent Methods to Evaluating Crop Water Stress Using Artificial Intelligence," Sensors (MDPI), vol. 24, no. 19, 2024.
[5]. "A Review of CNN Applications in Smart Agriculture Using Computer Vision: Emerging Trends and Future Challenges," Journal of Agricultural Informatics, 2025.
[6]. "Improving Crop Production Using an Agro–Deep Learning Framework," BMC Bioinformatics, 2024.
[7]. TensorFlow Documentation, TensorFlow Developer Resources, 2024.
[8]. Supabase Documentation, Supabase Developer Portal, 2024.
[9]. ReactJS Documentation, Meta Open Source, 2024.
[10]. M. Hasan, S. Ullah, and M. R. Khan, "Plant disease detection using deep learning and convolutional neural networks," International Journal of Agricultural Science and Technology, vol. 9, no. 1, pp. 45–56, 2021.
[11]. J. Too, L. Yujian, S. Njuki, and L. Yingchun, "A comparative study of fine-tuning deep learning models for plant disease identification," Computers and Electronics in Agriculture, vol. 161, pp. 272–279, 2019.
[12]. K. P. Ferentinos, "Deep learning models for plant disease detection and diagnosis," Computers and Electronics in Agriculture, vol. 145, pp. 311–318, 2018.
[13]. R. Sladojevic, M. Arsenovic, A. Anderla, D. Culibrk, and D. Stefanovic, "Deep neural networks based recognition of plant diseases by leaf image classification," Computational Intelligence and Neuroscience, vol. 2016, Article ID 3289801.
[14]. P. Mohanty, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," Frontiers in Plant Science, vol. 7, Article 1419, 2016.
[15]. Y. Lu, S. Yi, N. Zeng, Y. Liu, and Y. Zhang, "Identification of rice diseases using deep convolutional neural networks," Neurocomputing, vol. 267, pp. 378–384, 2017.
[16]. A Saleem, M. Akhtar, and M. Sharif, "Automatic plant disease detection using image processing and machine learning," Journal of Intelligent Systems, vol. 29, no. 1, pp. 1603–1621, 2020.