Jurnal Komputer, Informasi dan Teknologi Vol: 4, No 2, 2024, Page: 1-13

Identification of Cactus Species Using the MobileNet Convolutional Neural Network Model

William Mulyadi[1]*, Chairisni Lubis[2]
[1,2] Program Studi Teknik Informatika, Universitas Tarumanegara

DOI:
https://doi.org/10.53697/jkomitek.v4i2.1933
*Correspondence: William Mulyadi
Email:
william.535200017@stu.untar.ac.id

Received: 01-10-2024
Accepted: 11-11-2024
Published: 01-12-2024

Abstract: The objective of this design is to create a system capable of accurately and precisely classifying different types of succulent cacti. The system design aims to create a succulent cactus classification application using a Convolutional Neural Network (CNN) with the MobileNet architecture. The process involves collecting cactus images, dividing them into training and test datasets, and developing a CNN model to recognize patterns in succulent cactus species. The training data is used to train the CNN model, and the test data evaluates the model's accuracy. The trained model, stored in Tf.lite format, successfully classifies 15 cactus types, achieving high accuracy by employing preprocessing steps like resizing, normalization, and background removal. Over 1,200 cactus images were taken with a smartphone, categorized into 15 classes, and prepared to ensure optimal lighting, angle, background, and resolution (224x224 pixels). The MobileNet model was chosen for its high accuracy and efficiency. Hardware used includes a Samsung A54 smartphone and an Intel i7 laptop, with software such as Python, Kotlin, and Android Studio facilitating development. This design ensures the application's accessibility, making it a valuable tool for cactus enthusiasts and the general public to easily identify different succulent cactus types. Testing of the cactus species classification program using the Convolutional Neural Network (CNN) method with the MobileNetV2 architecture yielded strong results, achieving over 90% accuracy in classifying 15 cactus species. The highest training accuracy of 0.9837 was achieved at 150 epochs without early stopping, outperforming other epoch configurations. The model successfully classified species across five main genera—Kalanchoe, Crassula, Echeveria, Haworthia, and Euphorbia. This high accuracy highlights the model's effectiveness, making it a useful tool for cactus enthusiasts and the public to accurately identify and distinguish cactus species.

Keywords: Cactus Classification, Convolutional Neural Network (CNN), MobileNet Architecture

Introduction

Succulent cacti are a group of plants known for their ability to store water in their fleshy parts, allowing them to survive in dry and arid environments. This plant family includes various types and is widely distributed across the globe, especially in regions with extreme weather conditions, such as deserts. In Indonesia, although the climate is not entirely dry, succulent cacti remain popular as ornamental plants due to their unique and diverse shapes. Common types of succulent cacti found in Indonesia include the Kalanchoe, Crassula, Echeveria, Haworthia, and Euphorbia families. Each family has distinctive characteristics that set them apart, in terms of shape, color, and care requirements (Wakhidaturrohmah et al., 2022).

The diversity of succulent cacti in Indonesia often poses challenges for the general public, especially for ornamental plant enthusiasts, in recognizing and differentiating each type. This can lead to mistakes in care, negatively impacting plant growth and survival. Differences in environmental needs, such as light intensity, humidity, soil type, and watering frequency, make accurate identification of each succulent cactus type essential. For example, some types require high light exposure, while others grow optimally under shade with less light. Therefore, a proper understanding of each succulent species can help ensure optimal care based on each type's unique characteristics (Pradiatiningtyas, 2022).

In recent years, deep learning technology, particularly Convolutional Neural Networks (CNN), has proven highly effective in various applications of image recognition and object classification. CNNs can detect complex patterns in image data, making them an ideal tool for classifying images of succulent cacti. MobileNet architecture, a CNN variant optimized for devices with limited computing power, such as smartphones, offers a fast and efficient solution for real-time identification of succulent cacti species. Utilizing this technology, a system can be developed to recognize and classify various types of succulent cacti commonly found in Indonesia, facilitating users in providing appropriate care for their plants (Kurnia & Wibowo, 2021).

This research will develop a CNN-based classification model using MobileNet architecture to identify several common types of succulent cacti cultivated in Indonesia. Using a dataset of collected succulent cactus images, the model will be trained to identify cactus types based on their visual features. The model's accuracy will be evaluated and compared with previous research in the same field, such as studies on ornamental plant classification using deep learning methods. This research aims to make a significant contribution to digital image processing and further support the preservation and understanding of succulent cacti in Indonesia (Astriani et al., 2019).

Aside from aesthetic aspects, succulent cacti also hold significant economic value. The demand for these ornamental plants is growing, particularly among urban communities seeking easy-to-maintain yet visually appealing plants. Succulent cacti are often used as decorative elements in homes, offices, and public spaces, adding beauty and a natural touch to rooms. With the increasing popularity of urban gardening, many plant enthusiasts are beginning to collect various types of succulent cacti, both for hobby and as an investment. However, this growing demand also calls for a deeper knowledge of proper care methods, given each type of succulent cactus has unique care needs (Supirman et al., 2023).

With technological advancements, conventional approaches to recognizing and caring for succulent cacti can now be enhanced through digital technology, particularly in image processing and artificial intelligence (AI). Applications capable of automatically identifying succulent cactus types can provide users with the necessary information for proper care. This is not only beneficial for beginners but also for ornamental plant sellers who wish to offer additional services to their customers. The use of CNN and MobileNet in succulent cactus image classification represents a step forward in utilizing AI to solve everyday problems, such as plant identification and care (Kaur et al., 2022).

Furthermore, this research is expected to pave the way for similar technological applications in the broader fields of botany and agriculture. With CNN's ability to recognize complex visual patterns, this technology holds potential for various applications, such as plant health monitoring, disease detection, and agricultural optimization. Applying deep learning in plant classification provides direct benefits to ornamental plant enthusiasts and contributes to innovation in more efficient and sustainable agriculture. Thus, this research is not only relevant to the succulent cactus community but also has broader implications for utilizing technology to support environmental sustainability and enhance agricultural productivity (Naufal, 2021). The objective of this design is to create a system capable of accurately and precisely classifying different types of succulent cacti.

Methodology

System design is a crucial step to ensure that the program can produce outputs that meet the desired requirements and objectives. This process involves defining the specifications of the system to be created, as well as identifying specific goals to be achieved. In this context, the design is directed toward developing a succulent cactus classification application using the Convolutional Neural Network (CNN) method. The collected cactus image dataset will be divided into two parts: training data and test data.

The training data is used to train the CNN model to recognize relevant patterns associated with different types of succulent cacti, while the test data is used to evaluate the model's performance after training. During the training phase, images from the training data are fed into the CNN model. The result of this training process is a set of weights saved in a Tf.lite file format, which will be used in the testing phase.

In the testing phase, the stored training weights will be applied to classify images in the test dataset to evaluate the model's accuracy. Thus, this system design aims to build an effective and efficient classification model for identifying various types of succulent cacti, visually illustrated in Figure 1. Through this structured design, the classification application is expected to achieve optimal performance in identifying succulent cactus types, delivering accurate results, and being reliable under various usage conditions. The result of this system design is the creation of an application capable of classifying succulent cactus species based on input images. The system uses a Convolutional Neural Network (CNN) architecture with the MobileNet method to recognize and distinguish different cactus types with high accuracy. By utilizing a series of preprocessing techniques such as resizing, normalization, data augmentation, and background removal, the cactus images are processed to become optimal inputs for classification.

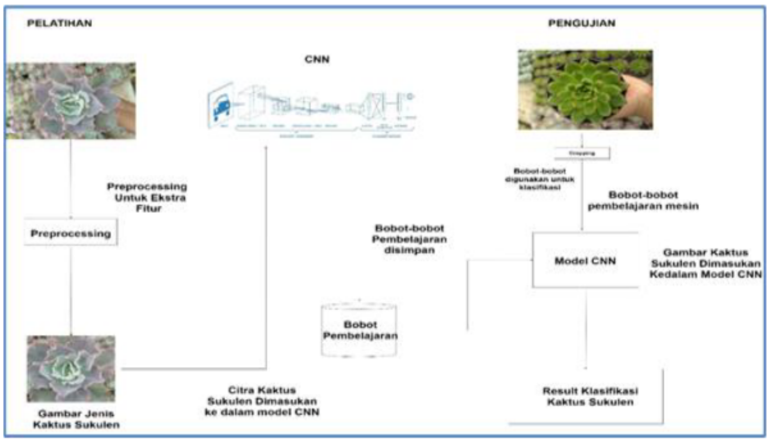
PELATIHAN
PENGUJIAN
CNN
Preprocessing Untuk Ekstra Fitur
Preprocessing
Gambar Jenis Kaktus Sukulen
Citra Kaktus Sukulen Dimasukan ke dalam model CNN
Bobot-bobot Pembelajaran disimpan
Bobot Pembelajaran
Bobot-bobot digunakan untuk klasifikasi
Bobot-bobot pembelajaran mesin
Model CNN
Gambar Kaktus Sukulen Dimasukan Kedalam Model CNN
Result Klasifikasi Kaktus Sukulen

Figure 1. System Design Flow Diagram

The first step in program development is the planning phase. In this phase, the main focus is on determining the objectives and goals to be achieved by the program being developed. This is done to ensure that the program will function well according to the needs and specifications that have been established. This program is primarily designed to classify various types of succulent cacti. It is hoped that this program will be a valuable tool for the general public, especially for those who may have limited knowledge about different types of succulent cacti. With this program, it is expected that people can more easily recognize and understand the various types of succulent cacti available in the market, especially in Indonesia.

System design is an important step to ensure that the program can produce output that meets the desired needs and goals. This process involves determining the specifications of the system to be developed, as well as identifying specific objectives to be achieved. In this context, the design is directed at developing a succulent cactus classification application using the Convolutional Neural Network (CNN) method.

The collected cactus image dataset will be divided into two parts: training data and test data. The training data is used to train the CNN model to recognize relevant patterns in different types of succulent cacti, while the test data is used to evaluate the model's performance after the training process. During the training phase, images from the training data are fed into the CNN model to train the network’s weights and biases. The result of this training process is a set of weights stored in a Tf.lite file format, which will be used during the testing phase.

In the testing phase, the stored training weights are applied to classify images in the test dataset to evaluate the model’s accuracy. Thus, this system design aims to build an effective and efficient classification model for recognizing various types of succulent cacti, as illustrated visually in Figure 1. Through this structured design, the classification application is expected to achieve optimal performance in identifying succulent cactus types, providing accurate results, and being reliable in various usage conditions.

This study utilizes a dataset of succulent cactus photos, taken directly using a smartphone camera. The dataset includes over 1200 images of cacti, classified into 15 different categories. Before entering the model training process, these images undergo a preparation stage involving transformations to optimize them for deep learning models. In the training phase, the prepared data is used as training data. All images in the dataset meet application requirements, including adequate lighting, optimal angles, a clean background, and a resolution of 224 x 224 pixels, eliminating the need for additional preprocessing. The CNN model selected is MobileNet, a suitable choice due to its high accuracy and efficient training time compared to other architectures.

Additionally, the necessary hardware and software for developing the succulent cactus classification application were analyzed. The hardware used includes a smartphone for image capture, a computer with adequate specifications for model training, and a mobile device for application implementation. Key hardware specifications include a Samsung A54 smartphone and a laptop with an Intel® Core™ i7 processor, 16 GB of RAM, Intel® FHD Graphics Family GPU, and a 512 GB SSD. The software used includes Microsoft Windows

10, Android 9.04, and various development tools such as Visual Studio Code, Google Colab, Android Studio, and Microsoft Office 365. The application development is supported by Python version 3.9 and Kotlin version 1.7 programming languages. This approach enables the system to classify succulent cacti accurately and efficiently while ensuring the application is accessible and easy to use for a broad audience.

Result and Discussion

Testing Method

The Black Box Testing method is applied to test the mobile application based on Convolutional Neural Network (CNN) for classifying cactus types. The primary goal of this method is to ensure that the application's output aligns with the expected design and functionality. This testing is conducted on a cactus classification application that identifies 15 cactus types, each assigned a specific color label to simplify the verification of classification results. Through this testing process, the application is expected to identify and classify cactus types with high accuracy. Each type of cactus possesses unique characteristics and is assigned a special label, making it easier to verify the classification results. Black Box Testing focuses on testing the application's output without assessing the internal processes or algorithms utilized within the system. This method is highly effective because it allows testing from the end-user perspective. Using this approach, Black Box Testing helps identify potential errors in the user interface, integration, and output that may not align with the specified design. This ensures that the application functions as intended in the design phase. The Black Box Testing process involves several stages, including input validation, user interface testing, and classification result verification based on the color labels assigned to each cactus type. The outcomes of this testing provide crucial insights for refining and enhancing the application before it is released to users.

Testing Results

After completing the planning phase, this section details the step-by-step testing results of the cactus flower classification application. Testing was conducted using a Convolutional Neural Network (CNN) model with the MobileNetV2 architecture to ensure the application can classify cactus flowers accurately and efficiently. The first step in testing involved preparing the dataset, consisting of cactus flower images taken under uniform lighting conditions to maintain result consistency. These images were then inputted into the pre-trained CNN model using a similar dataset. The model classified each cactus flower image, producing a label that best matched the flower type. The testing results demonstrated that the CNN model with the MobileNetV2 architecture achieved a high accuracy level in classifying cactus flowers. Each classification was compared with predetermined labels to verify the model’s accuracy. In certain cases, the model showed remarkable precision in identifying specific cactus flowers, confirming the effectiveness of this approach for cactus classification.

In addition to accuracy, the testing also evaluated computational speed and efficiency. The MobileNetV2 model exhibited short inference times, enabling real-time classification results, which is essential to ensure an optimal user experience on mobile devices. The conclusion from this testing is that a mobile application using a CNN model with the MobileNetV2 architecture is reliable for cactus flower classification. This application not only delivers accurate classifications but also processes data quickly and efficiently. Further testing will be conducted with a larger, more diverse dataset to validate the model's ability to generalize across a wider variety of real-world conditions. Thus, the testing indicates that this application has strong potential for widespread use, especially among cactus enthusiasts, for both personal and professional purposes. The implementation of this technology is expected to simplify the process of selecting and identifying cactus types quickly and accurately for users.

Testing with 100 Epochs Using Early Stopping

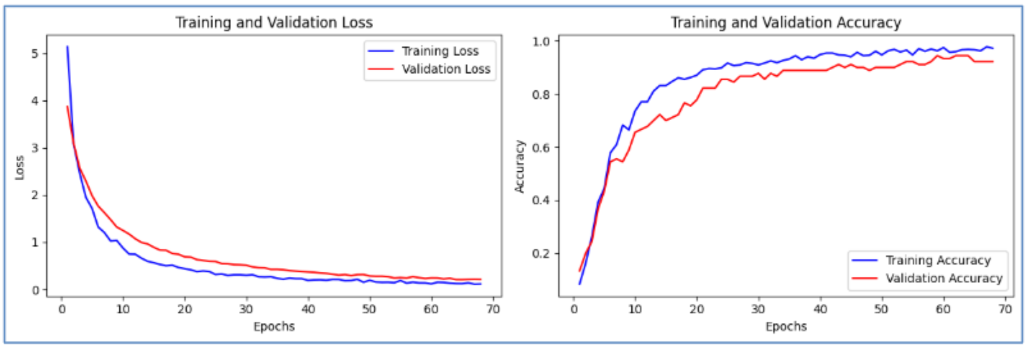

Figure 2. Accuracy Plot for 100 Epochs with Early Stopping

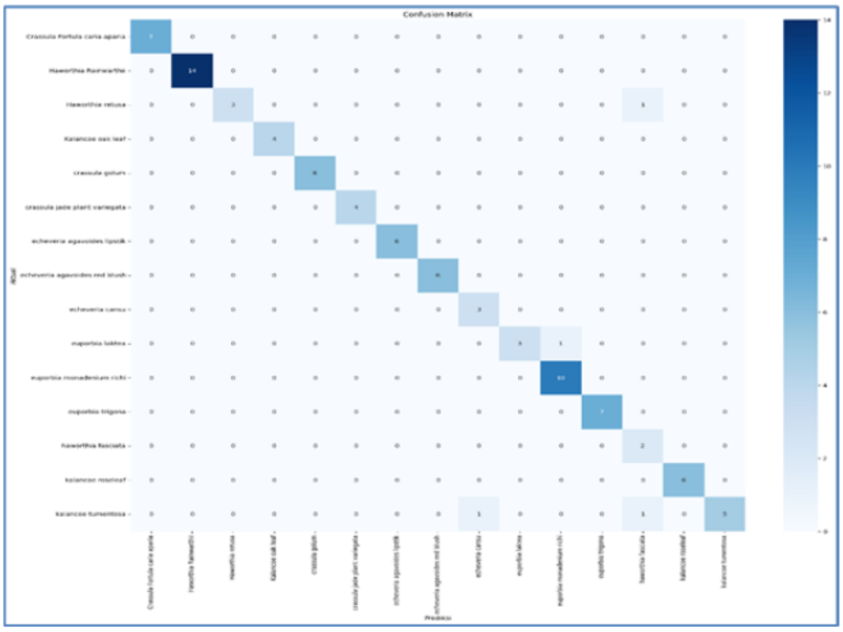

Figure 3. Confusion Matrix for 100 Epochs with Early Stopping

During the MobileNet model training process, which was conducted over 100 epochs with the Early Stopping technique, the model showed significant performance improvements from the very first epoch. Initially, in the first epoch, training accuracy was only 7.7% with a relatively high loss value of 6.4308, and validation loss (val_loss) stood at 3.8682. However, as the epochs progressed, model accuracy consistently improved. By the 10th epoch, accuracy reached 74.46% with a loss of 0.8898, and validation accuracy increased to 65.56% with a val_loss of 1.2491. The model continued to show performance gains, ultimately reaching a validation accuracy of 94.44% by the 64th epoch, with a considerably lower val_loss of 0.2091. Early Stopping was employed to halt training when model performance began to stabilize, thus preventing potential overfitting. These results indicate that the MobileNet model achieved high accuracy with progressively lower error rates, confirming that the model is adequately prepared for further classification or recognition tasks.

Testing with 100 Epochs Without Early Stopping

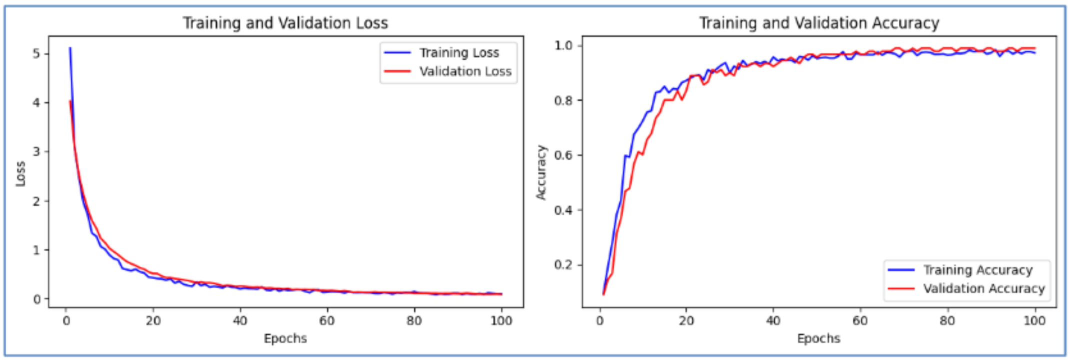

Figure 4. Accuracy Plot for 100 Epochs Without Early Stopping

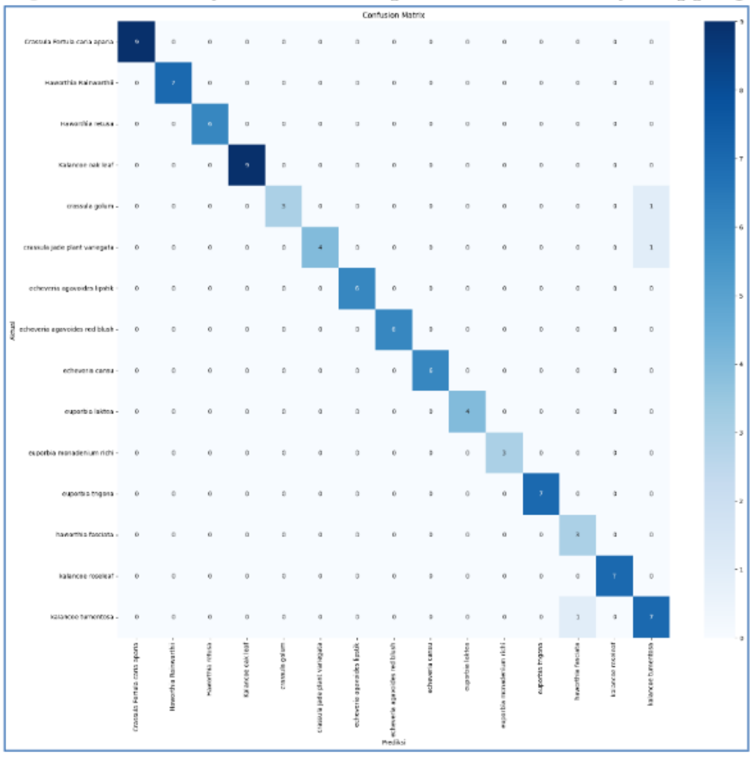

Figure 5. Confusion Matrix for 100 Epochs Without Early Stopping

The training of the MobileNet model over 100 epochs without the use of Early Stopping showed a gradual improvement in model accuracy. Initially, in the first epoch, the training accuracy was relatively low at 7.65%, with a high loss value of 6.5096 and a validation accuracy of 8.89%. However, during the early epochs, specifically around epochs 5 to 10, model accuracy increased rapidly, reaching 72.72% while the loss decreased to approximately 0.85.

As training continued, the model's accuracy steadily improved, with validation accuracy reaching around 80–90% by the 20th epoch. By the 30th epoch, the model achieved a training accuracy of 92.25%, with a val_loss of 0.3199 and a validation accuracy of approximately 93.33%. Towards the end of the training, specifically between epochs 60–70, the model stabilized, achieving training accuracy in the range of 95%–98%, with validation accuracy reaching up to 98.89% and val_loss decreasing further to 0.1243 by epoch 69.

Overall, the model demonstrated strong performance, with significant accuracy gains in each epoch and a consistent decrease in loss, indicating that the model learned effectively from the training data and showed solid generalization to the validation data.

Testing with 150 Epochs Using Early Stopping

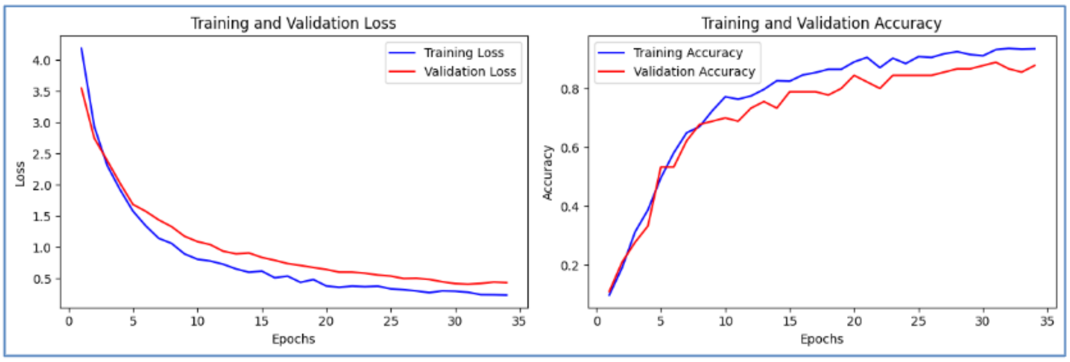

Figure 6. Accuracy Plot for 150 Epochs with Early Stopping

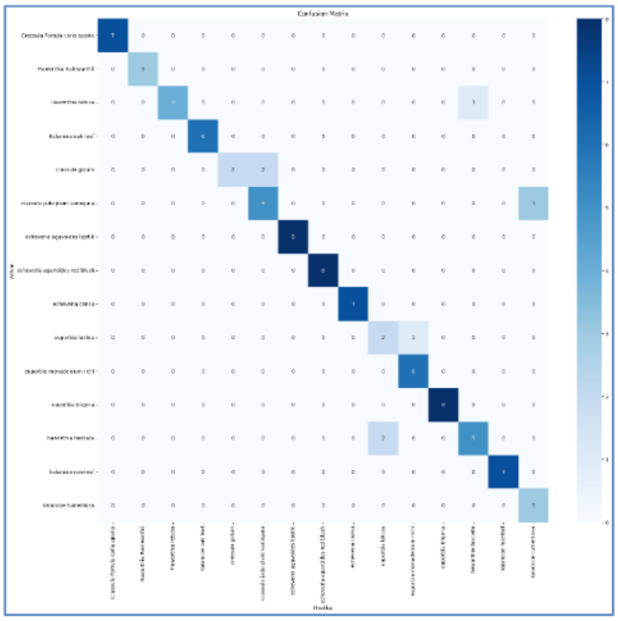

Figure 7. Confusion Matrix for 150 Epochs with Early Stopping

Training the model for 150 epochs with Early Stopping demonstrated a significant increase in accuracy and a consistent decrease in loss. In the first epoch, the model began with a relatively low training accuracy of approximately 8.28% and a validation accuracy of 11.11%. However, after just a few epochs, the model showed rapid improvement, reaching a training accuracy of 47.18% and a validation accuracy of 53.33% by the 5th epoch, indicating that it was quickly learning to recognize patterns.

Gradually, the accuracy continued to increase, reaching around 90% by the 30th epoch, with a validation accuracy of approximately 87.78% and a steadily decreasing loss, reflecting strong generalization performance on the validation data. After the 30th epoch, the model's accuracy began to stabilize, with validation accuracy fluctuating around 85-88%, suggesting that the model was nearing convergence. Around the 32nd epoch, although accuracy was still improving, fluctuations in validation loss suggested a potential onset of overfitting, as the validation loss reduction became slower and less aligned with the decrease in training loss. The use of Early Stopping here was highly effective, as it halted training before overfitting could intensify, preserving the model's optimal performance on validation data. Overall, training with 150 epochs and Early Stopping yielded a robust model with strong generalization, achieving high accuracy and low loss on validation data.

Testing with 150 Epochs Without Early Stopping

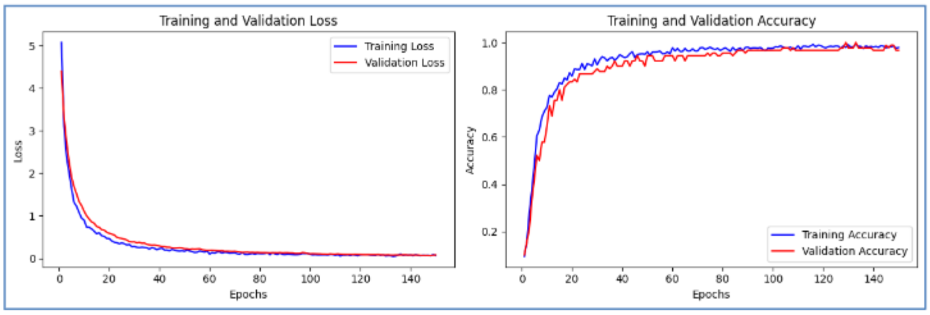

Figure 8. Accuracy Plot for 150 Epochs Without Early Stopping

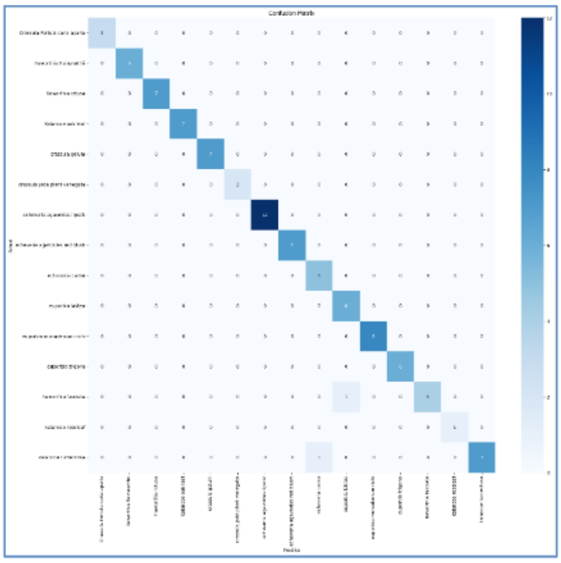

Figure 9. Confusion Matrix for 150 Epochs Without Early Stopping

The results of testing without Early Stopping over 150 epochs show that the model exhibited a significant upward trend in accuracy as the epochs increased. At the start of the training, the model's accuracy was very low, around 0.08 in the first epoch, but it gradually improved over time. By the 10th epoch, accuracy reached approximately 0.74, with the loss steadily decreasing, indicating that the model was gradually learning. Validation accuracy also improved, starting from around 0.1 at the beginning and reaching 0.94 by the 72nd epoch, suggesting that the model had learned to generalize well on the validation data. However, after reaching this point, the improvement in accuracy appeared to plateau, and the low validation loss indicated the potential for overfitting if the training continued without Early Stopping. This suggests that while the model was able to achieve high accuracy, continuing without early stopping could lead to overfitting, as the model may start to fit too closely to the training data at the expense of its ability to generalize well to unseen data.

Comparative Analysis of Classification Methods Based on Epoch Testing

The following table compares the results of the model's performance across four different epoch configurations, both with and without Early Stopping (ES). The metrics presented include training accuracy, training loss, testing accuracy, and testing loss for each configuration.

Table 1. Comparison of Testing with Epoch Configurations

Epoch	Training Accuracy	Training Loss	Testing Accuracy	Testing Loss
100 ES	0.9795	0.1009	0.9222	0.2136
100	0.9742	0.0912	0.9889	0.0836
150 ES	0.9366	0.2449	0.8778	0.4357
150	0.9837	0.0594	0.9667	0.0774

Based on the testing results across different epoch configurations with and without Early Stopping (ES), the model performance varies significantly. For the 100-epoch configuration with Early Stopping (ES), the model achieved a training accuracy of 0.9795 with a training loss of 0.1009. However, the testing accuracy was 0.9222 with a testing loss of 0.2136. While the testing accuracy is quite high, the relatively higher testing loss indicates that the model may not have been fully optimized, likely due to the early stopping cutting off the training process before the model reached its optimal state. Without Early Stopping in the 100-epoch configuration, the model showed a better performance with the highest testing accuracy of 0.9889 and a lower testing loss of 0.0836. This suggests that the model was able to learn more effectively without early intervention, resulting in better pattern recognition and generalization to the testing data.

In the 150-epoch configuration with Early Stopping, the model exhibited lower training accuracy (0.9366) and higher training loss (0.2449). The testing accuracy was 0.8778 with a higher testing loss of 0.4357. This indicates that early stopping at this stage prevented the model from reaching its full potential, likely halting training before sufficient learning had occurred. In contrast, without Early Stopping in the 150-epoch configuration, the model performed more steadily, achieving a testing accuracy of 0.9667 and a testing loss of 0.0774. Although these results were slightly lower than the 100-epoch configuration without Early Stopping, the model demonstrated good stability and generalization, with a balanced accuracy and loss performance. From this analysis, it can be concluded that the model performed best during the 100-epoch configuration without Early Stopping, achieving the highest testing accuracy with the lowest testing loss. This configuration resulted in the most optimal generalization, without signs of overfitting, compared to the other setups. Early Stopping, while effective in preventing overfitting, can sometimes stop training before the model reaches its full potential, as observed in the 150-epoch scenarios. Therefore, a balanced approach, with careful consideration of both early stopping and the number of epochs, is crucial for achieving the best performance in classification tasks.

Conclusion

Based on the cactus species classification program testing using the Convolutional Neural Network (CNN) method with the MobileNetV2 architecture, the following conclusion can be drawn. The Convolutional Neural Network method with MobileNet achieved good results in classifying cactus species into 15 classes, with an accuracy rate exceeding 90%. Among the four comparisons using epoch testing, 150 Epoch without early stopping achieved the highest training accuracy compared to 100 epochs with early stopping, 100 epochs without early stopping, and 150 epochs with early stopping.

The cactus species classification testing using the CNN model with the MobileNetV2 architecture successfully achieved the highest accuracy of 0.9837 at 150 epochs without early stopping. This model was able to recognize 15 cactus species, divided into five main genera: Kalanchoe, Crassula, Echeveria, Haworthia, and Euphorbia. In the Kalanchoe genus, the identified species include Kalanchoe oak leaf, Kalanchoe tumentosa, and Kalanchoe rose leaf. The Crassula genus is represented by Crassula jade plant ovata variegata, Crassula golum ovata, and Crassula fortula caria apra. Echeveria consists of Echeveria agavoides red blush, Echeveria cansu, and Echeveria saviana curly, while the Haworthia genus includes Haworthia rainwarthii, Haworthia fasciata, and Haworthia cooperi. Finally, the Euphorbia genus consists of Euphorbia milii, Euphorbia obesa, and Euphorbia lactea. With nearly perfect accuracy, this model demonstrates high effectiveness in classifying cactus species, making it a valuable tool to help cactus enthusiasts and the general public recognize and differentiate various cactus species more easily and accurately.

References

Astriani, L., Bahfen, M., Mulyanto, T. Y., & Istikomah, I. (2019). Pemberdayaan Masyarakat melalui Budidaya Tanaman Hias Sukulen dalam Pot. *Prosiding Seminar Nasional Pengabdian Masyarakat LPPM UMJ*, *1*(1), Article 1. https://jurnal.umj.ac.id/index.php/semnaskat/article/view/8856

Kaur, S., Rakhra, M., Singh, D., Singh, A., & Aggarwal, S. (2022). Disease Detection in Cactus (Beles) via the Use of Machine Learning: A Proposed Technique. *2022 2nd International*

Conference on Technological Advancements in Computational Sciences (ICTACS), 772–778. https://doi.org/10.1109/ICTACS56270.2022.9988580

Kurnia, D., & Wibowo, A. T. (2021). Klasifikasi Spesies Tanaman Kaktus Grafting Berdasarkan Citra Scion Menggunakan Metode Convolutional Neural Network (cnn). *eProceedings of Engineering*, *8*(4). https://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/15244

Naufal, M. F. (2021). Analisis Perbandingan Algoritma Svm, Knn, Dan Cnn untuk Klasifikasi Citra Cuaca. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *8*(2), Article 2.

Pradiatiningtyas, D. (2022). Edukasi Budidaya Tanaman Hias Kaktus Dan Pemasaran Melalui Media Digital Pada Komunitas Nirlaba Cactus And Succulent Society Of Indonesia (CSSI). *SPEED - Sentra Penelitian Engineering Dan Edukasi*, *14*(1), Article 1. https://doi.org/10.55181/speed.v14i1.750

Supirman, S., Lubis, C., Yuliarto, D., & Perdana, N. J. (2023). Klasifikasi Penyakit Kulit Menggunakan Convolutional Neural Network (Cnn) Dengan Arsitektur VGG16. *Simtek : Jurnal Sistem Informasi Dan Teknik Komputer*, *8*(1), Article 1. https://doi.org/10.51876/simtek.v8i1.217

Wakhidaturrohmah, N., Murtaqib, M., & Kushariyadic, K. (2022). Meningkatkan Usaha Budidaya Tanaman Hias Suculen Dan Kaktus Dengan Inovasi Packaging "Sucubox" Serta Pemasaran Media Sosial Didesa Kebonsari, Kecamatan Sumbersari, Kabupaten Jember. *PROSIDING SEMINAR NASIONAL PENGABDIAN KEPADA MASYARAKAT*, *2*(1), Article 1. https://doi.org/10.33086/snpm.v2i1.953

Alshehhi, R. (2021). Detection of Coronal Mass Ejections Using Unsupervised Deep Clustering. Solar Physics, 296(6). https://doi.org/10.1007/s11207-021-01854-w

Atitallah, S. B. (2021). An Enhanced Randomly Initialized Convolutional Neural Network for columnar cactus recognition in unmanned aerial vehicle imagery. Procedia Computer Science, 192, 573–581. https://doi.org/10.1016/j.procs.2021.08.059

Benyahia, S. (2022). Multi-features extraction based on deep learning for skin lesion classification. Tissue and Cell, 74. https://doi.org/10.1016/j.tice.2021.101701

Berka, A. (2023). CactiViT: Image-based smartphone application and transformer network for diagnosis of cactus cochineal. Artificial Intelligence in Agriculture, 9, 12–21. https://doi.org/10.1016/j.aiia.2023.07.002

Kassania, S. H. (2021). Automatic Detection of Coronavirus Disease (COVID-19) in X-ray and CT Images: A Machine Learning Based Approach. Biocybernetics and Biomedical Engineering, 41(3), 867–879. https://doi.org/10.1016/j.bbe.2021.05.013

Kogilavani, S. V. (2022). COVID-19 Detection Based on Lung Ct Scan Using Deep Learning Techniques. Computational and Mathematical Methods in Medicine, 2022. https://doi.org/10.1155/2022/7672196

Li, Y. (2022). EfficientFormer: Vision Transformers at MobileNet Speed. Advances in Neural Information Processing Systems, 35.

Maji, D. (2022). Attention Res-UNet with Guided Decoder for semantic segmentation of brain tumors. Biomedical Signal Processing and Control, 71. https://doi.org/10.1016/j.bspc.2021.103077

Ohata, E. F. (2021). Automatic detection of COVID-19 infection using chest X-ray images through transfer learning. IEEE/CAA Journal of Automatica Sinica, 8(1), 239–248. https://doi.org/10.1109/JAS.2020.1003393

Perez, M. F. (2022). Coalescent-based species delimitation meets deep learning: Insights from a highly fragmented cactus system. Molecular Ecology Resources, 22(3), 1016–1028. https://doi.org/10.1111/1755-0998.13534

Srinivasu, P. N. (2021). Classification of skin disease using deep learning neural networks with mobilenet v2 and lstm. Sensors, 21(8). https://doi.org/10.3390/s21082852

Yao, H. (2017). An end-to-end method of Coronal Mass Ejections detection. Kexue Tongbao/Chinese Science Bulletin, 62(23), 2680–2690. https://doi.org/10.1360/N972016-00382