ISSN 2347 - 3983

Volume 11. No.12, December 2023

International Journal of Emerging Trends in Engineering Research

Available Online at http://www.warse.org/IJETER/static/pdf/file/ijeter1011122023.pdf

https://doi.org/10.30534/ijeter/2023/1011122023

Mushroom Decay Detection using Deep Learning

Y. Rakesh Kumar[1], V. Venkata Laxmi Priya[2], R. Swathi[3], K. Priyanka[4], G. Vyshnavi[5]

[1]G. Narayanamma Institute of Technology and Science, Hyderabad, Telangana, India
[1] rakeshyacharam@gmail.com
[2]G. Narayanamma Institute of Technology and Science, Hyderabad, Telangana India
[2] rkadsbcm@gmail.com
[3]G. Narayanamma Institute of Technology and Science, Hyderabad, Telangana, India
[3] swathiramasamy11@gmail.com
[4]G. Narayanamma Institute of Technology and Science, Hyderabad, Telangana, India
[4] priyankakoram111@gmail.com
[5]G. Narayanamma Institute of Technology and Science, Hyderabad, Telangana, India
[5] vyshnavi272@gmail.com

Received Date: October 26, 2023 Accepted Date: November 29, 2023 Published Date : December 07, 2023

ABSTRACT

This paper aims to develop a system for detecting decay in mushrooms using image processing techniques and Deep Learning. Decay in mushrooms is a significant issue in the food industry, as it can lead to quality deterioration and potentially harmful consumption. The proposed system involves capturing images of mushrooms using a camera and processing these images to detect any signs of decay. The image processing techniques will include pre-processing, feature extraction, and classification using deep learning algorithms. The dataset used for training and testing the system will consist of images of both healthy and decayed mushrooms. The system's performance will be evaluated based on accuracy. The outcome of this project will be a tool that can assist in early detection of decay in mushrooms, thus reducing food waste and improving food safety.

Key words: Mushroom decay, Deep Learning, Classification, Anaconda, Image processing

1. INTRODUCTION

The food industry plays a vital role in ensuring the quality and safety of food products for consumers. Mushroom cultivation is a critical component of the agricultural industry, providing a valuable source of food and income for many communities worldwide. Mushrooms are a popular and nutritious food source enjoyed by people all around the world. However, mushrooms are highly perishable and can deteriorate quickly due to various factors, including bacterial and fungal contamination, physical damage, and unfavorable storage conditions [1]. One persistent challenge in this industry is the early detection of decay or spoilage in perishable items such as mushrooms. The occurrence of decay not only leads to significant economic losses for growers, distributors and retailers but also poses potential health risks to consumers if undetected. Therefore, the development of robust and efficient methods for detecting decay in mushrooms is of paramount importance. Identifying decayed mushrooms manually is time-consuming, subjective, and prone to errors, leading to economic losses and food waste. To address this issue, there is a pressing need for an automated system that can accurately and efficiently detect decayed mushrooms in real-time.

2. LITERATURE SURVEY

Researchers have been using computers to help detect and classify diseases in mushrooms for a while now. Here are a few important studies:

[1] The author provides an overview of the latest advancements and research in the field of edible and poisonous mushroom recognition, emphasizing the state of the art in this area. This conference paper contributes to the discussions on computing, communication, and networking technologies.

[2] The author's research talked about how computers, especially Support Vector Machines, can be very useful for this kind of job. Training an SVM, usually posed as a quadratic programming (QP) problem, often becomes a challenging task for the large data sets due to the high memory requirements and slow convergence.

[3] The author ‘s work focused on the effectiveness of the two machine learning methods in classifying mushroom types, shedding light on their relative strengths and weaknesses in the context of mushroom classification. This research is valuable for understanding and optimizing classification techniques, which can have applications in fields like mycology and agriculture.

[4] The author's work focused on the development of a machine learning model capable of identifying and categorizing fruits based on their visual characteristics. This work contributes to the field of intelligent systems and computing, specifically in the context of fruit classification.

[5] The author discusses the use of advanced deep learning techniques to improve the accuracy and efficiency of fruit classification, highlighting the integration of attention mechanisms and convolutional autoencoders in this context.

2.1 Methodology

The aim of this project is to create a user-friendly tool that can accurately classify mushrooms into their freshness stages:
"Fresh”,” Mild - Decayed" and "Decayed".

1. Data Collection and Categorization:

Data Gathering: We began by collecting a diverse dataset of mushroom images as shown in Figure 1 captured using smartphones in various location, at different angles and at different times.

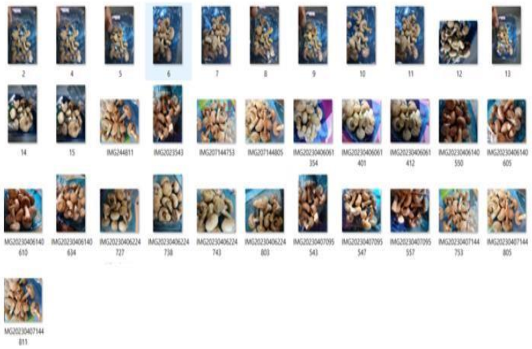


Figure 1: Decayed Mushroom Dataset

Data Annotation: These images were manually annotated and categorized into three classes: "Fresh," "Mid-Stage," and "Decayed." Each class represented a distinct stage of mushroom development.

2. Data Preprocessing:

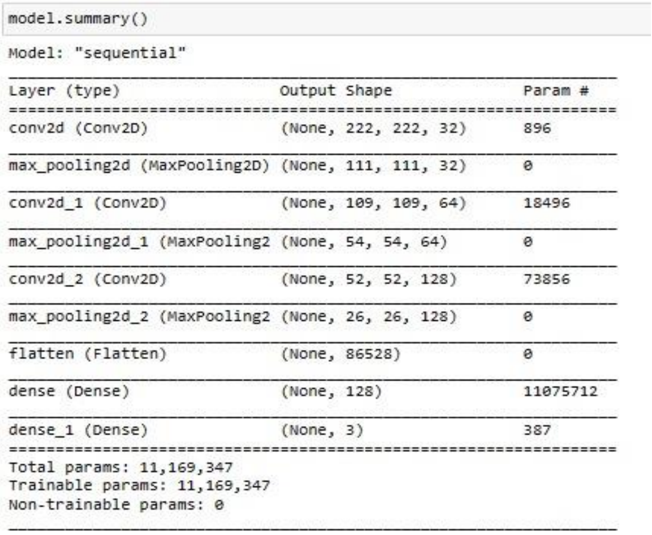
Image Standardization: To ensure consistency, all images were resized to a common resolution.

Data Augmentation: We applied data augmentation techniques such as rotation and flipping to increase the dataset's diversity, thereby improving model robustness.

3. Deep Learning Algorithm Selection and Model Architecture:

Convolutional Neural Network (CNN): We chose a CNN architecture, well-suited for image classification tasks, as the foundation for our model.

In the implementation of our neural network model, it is imperative to provide a comprehensive overview of the model's architecture, hyperparameters, and technical details to facilitate a clear understanding of our methodology. One essential aspect of this description is the utilization of the “model.summary()” function, which plays a pivotal role in elucidating the design of our neural network. The summary (Figure 2) provides details about the layers in the model, including the type of layer, output shape, the number of parameters, and the total number of parameters in the model. This information is crucial for understanding and debugging your neural network model.



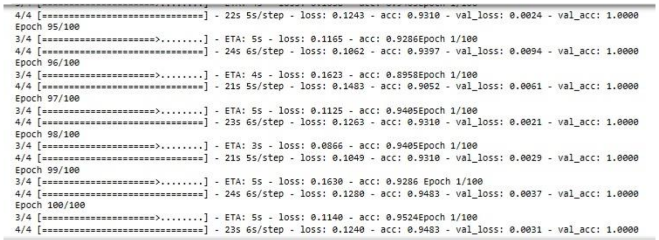
```
model.summary()

Model: "sequential"
_________________________________________________________________
Layer (type)                Output Shape              Param #
=================================================================
conv2d (Conv2D)             (None, 222, 222, 32)      896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 111, 111, 32)     0
_________________________________________________________________
conv2d_1 (Conv2D)           (None, 109, 109, 64)      18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 54, 54, 64)       0
_________________________________________________________________
conv2d_2 (Conv2D)           (None, 52, 52, 128)       73856
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 26, 26, 128)      0
_________________________________________________________________
flatten (Flatten)           (None, 86528)             0
_________________________________________________________________
dense (Dense)               (None, 128)               11075712
_________________________________________________________________
dense_1 (Dense)             (None, 3)                 387
=================================================================
Total params: 11,169,347
Trainable params: 11,169,347
Non-trainable params: 0
_________________________________________________________________
```

Figure 2: Summary of the model for Mushroom Decay Detection

Data Split: The dataset was divided into training, validation, and test sets, with the majority allocated for training.

Training Parameters: We fine-tuned hyper parameters, such as learning rate and batch size, to optimize model performance.

Epochs: The training process for this model was carried out over a course of 100 epochs as shown in the Figure 3, where each epoch represents a complete pass through the entire training dataset. This extensive training allowed the model to iteratively refine its parameters and learn from the data for a hundred full cycles.



```
4/4 [==============================] - 22s 5s/step - loss: 0.1243 - acc: 0.9310 - val_loss: 0.0024 - val_acc: 1.0000
Epoch 95/100
3/4 [=====================>........] - ETA: 5s - loss: 0.1165 - acc: 0.9286Epoch 1/100
4/4 [==============================] - 24s 6s/step - loss: 0.1062 - acc: 0.9397 - val_loss: 0.0094 - val_acc: 1.0000
Epoch 96/100
3/4 [=====================>........] - ETA: 4s - loss: 0.1623 - acc: 0.8958Epoch 1/100
4/4 [==============================] - 21s 5s/step - loss: 0.1483 - acc: 0.9052 - val_loss: 0.0061 - val_acc: 1.0000
Epoch 97/100
3/4 [=====================>........] - ETA: 5s - loss: 0.1125 - acc: 0.9405Epoch 1/100
4/4 [==============================] - 23s 6s/step - loss: 0.1263 - acc: 0.9310 - val_loss: 0.0021 - val_acc: 1.0000
Epoch 98/100
3/4 [=====================>........] - ETA: 3s - loss: 0.0866 - acc: 0.9405Epoch 1/100
4/4 [==============================] - 21s 5s/step - loss: 0.1049 - acc: 0.9310 - val_loss: 0.0029 - val_acc: 1.0000
Epoch 99/100
3/4 [=====================>........] - ETA: 5s - loss: 0.1630 - acc: 0.9286 Epoch 1/100
4/4 [==============================] - 24s 6s/step - loss: 0.1280 - acc: 0.9483 - val_loss: 0.0037 - val_acc: 1.0000
Epoch 100/100
3/4 [=====================>........] - ETA: 5s - loss: 0.1140 - acc: 0.9524Epoch 1/100
4/4 [==============================] - 23s 6s/step - loss: 0.1240 - acc: 0.9483 - val_loss: 0.0031 - val_acc: 1.0000
```

Figure 3: Training process for the model over 100 epochs

4. Algorithm Evaluation: Assessing Algorithm Proficiency in Mushroom Decay Detection:

Accuracy Metrics: We assessed model performance using accuracy, precision, recall, and F1-score, specifically tailored for multiclass classification.

Confusion Matrix: To understand model behavior, we examined the confusion matrix to identify areas of misclassification.

5. User Interaction:

Jupyter Notebook Interface: For practical implementation of the entire project as shown in Figure 4, we created a Jupyter Notebook-based interface, allowing users to input mushroom images and receive real-time freshness stage predictions.

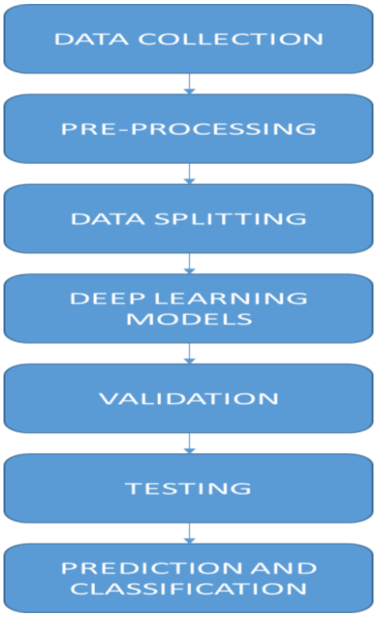


Figure 4: Flowchart   of proposed System implementation

2.2 CNN Architecture

CNNs are a class of deep learning models specifically designed for image analysis and recognition tasks. They have been instrumental in this project for analyzing mushroom images to detect signs of decay. CNNs excel at this because they can automatically learn and extract complex features from images. The network comprises layers of convolutional and pooling operations, followed by fully connected layers for classification as shown in Figure 5 [5].

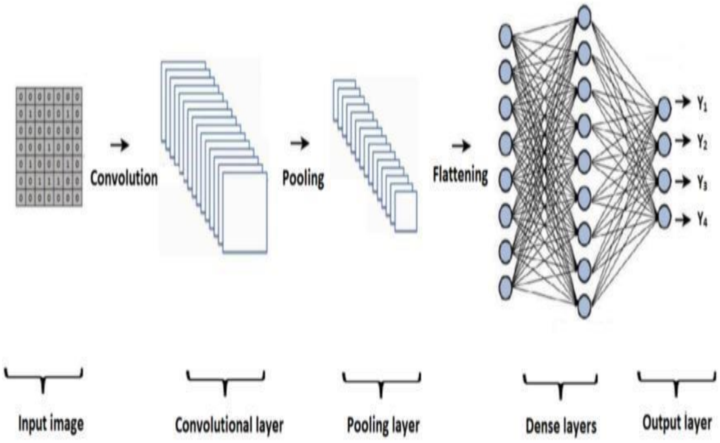


Figure 5: CNN Architecture

2.3 VGG19 Architecture

VGG-19 is characterized by its deep architecture, consisting of 19 layers (Figure 6), which allows it to learn intricate features in images. In this project, VGG-19 plays a crucial role by serving as a pre-trained model for feature extraction. By utilizing transfer learning with VGG-19, we leverage the model's ability to recognize a wide range of visual features from images. Fine-tuning VGG-19 for mushroom decay detection enhances the system's performance by harnessing the knowledge it has acquired from a vast dataset [2].

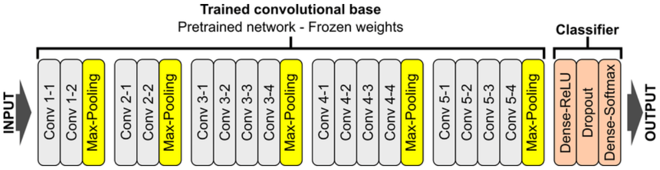


Figure 6: VGG 19 block diagram

A fixed size of (224 * 224) RGB image was given as input to this network which means that the matrix was of shape (224,224,3).

The only preprocessing that was done is that they subtracted the mean RGB value from each pixel, computed over the whole training set.

Used kernels of (3 * 3) size with a stride size of 1 pixel, this enabled them to cover the whole notion of the image.

Spatial padding was used to preserve the spatial resolution of the image.

Max pooling was performed over a 2 * 2 pixel windows with sride 2.

The code we used for training a convolutional neural network (CNN) for a mushroom classification task using the TensorFlow and Keras libraries involves the following techniques:

1. **Transfer Learning**:

Transfer learning is employed by using a pre-trained VGG19 model as the base model. The VGG19 model is loaded with pre-trained weights from the "imagenet" dataset. This allows the model to leverage the knowledge learned from a large dataset to perform feature extraction on the mushroom images.

2. **Fine-Tuning**:

While using transfer learning, the code freezes the layers of the VGG19 model (i.e., sets them as non-trainable) to retain the pre-trained knowledge. However, the final fully connected layers are replaced with custom layers.
Only these custom layers are trained on the mushroom dataset. This process is known as fine-tuning.

3. **Data Augmentation**:

Data augmentation is applied to the training dataset using ImageDataGenerator. It includes operations like rescaling pixel values, horizontal flipping, and zooming. Data augmentation helps in increasing the diversity of the training data, which can improve model generalization [3].

4. **Data Splitting**:

The dataset is split into a training subset and a validation subset using the validation_split parameter of ImageDataGenerator. This allows the model to be trained on a portion of the data while validating its performance on another portion.

The code employs transfer learning with a VGG1model, fine-tunes it on a mushroom dataset, applies data augmentation to increase dataset diversity and uses callbacks for early stopping during training. The model is evaluated using accuracy and loss metrics.

The base of the model is VGG19, a well-known CNN architecture. VGG19 is known for its depth and has been pretrained on a vast dataset that includes a wide variety of images. This means it has learned to recognize different shapes, colors, and structures commonly found in images [4].

In the configuration, the decision was made to avoid retraining the entire VGG19 model from the beginning. Instead, the weights of the pre-trained VGG19 layers were 'frozen' or kept fixed. This strategy is commonly employed to retain the valuable knowledge acquired by the model during its initial training on a wide range of diverse images.

The VGG19 model was applied to a specific classification task involving images categorized into two classes: those depicting the presence of trees and those depicting the absence of trees. This classification problem is characterized as binary.

To enhance the model's classification capabilities, a supplementary neural network comprising two layers was introduced atop the VGG19 layers with fixed weights. These two additional layers collectively constitute a 'shallow' neural network, serving as the final decision-making component responsible for classifying images based on the feature representations extracted by the underlying VGG19 layers.

3. RESULTS AND DISCUSSIONS

The study focused on developing an effective method for discerning the freshness status of button mushrooms, distinguishing between fresh and decaying stages. Approximately,100 Mushroom images (shown in Figure 7) were captured using mobile devices equipped with a high-resolution 48-megapixel camera, resulting in photos with a resolution of 1080x2340 pixels covering different stages of development, including freshness and decay.

To ensure the method's efficiency, the image dataset was thoughtfully divided into three subsets: one for teaching, one for testing and one for validation, maintaining an 85:15 ratios. Consequently, the teaching subset consisted of 116 images, while the testing subset contained 28 images. The method exhibited a noteworthy accuracy of 96.55% during testing, indicating its proficiency in accurately categorizing mushroom freshness.

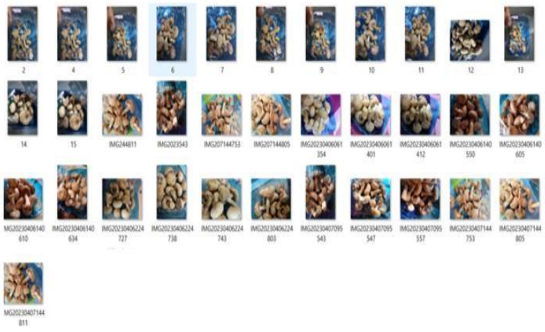
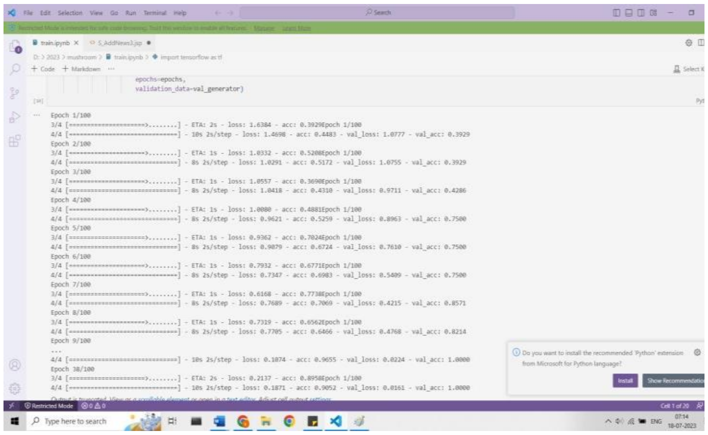


Figure 7: Data set of the mushrooms

The images shown in figure 4.1 are considered from the data set collected. In the process of compiling mushroom images for this project, a systematic approach was employed to ensure comprehensive coverage of various mushroom stages. To categorize the mushrooms, careful attention was paid to their visual characteristics and signs of freshness or decay. Each image was then meticulously sorted into one of three categories: Fresh, Mild-Decayed or Decayed.

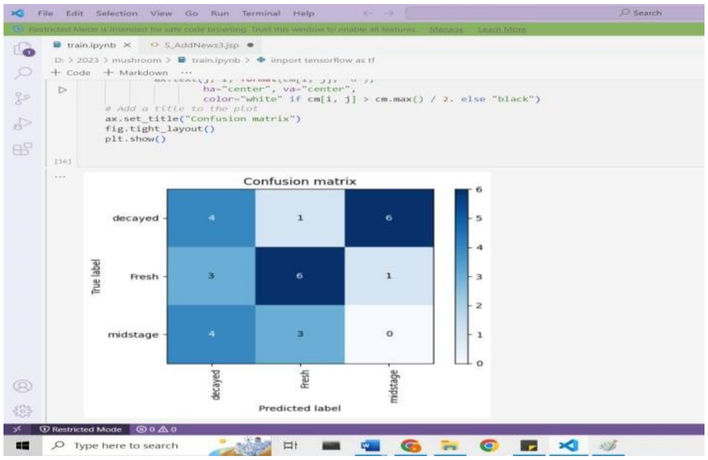

File Edit Selection View Go Run Terminal Help
Search
train.ipynb
S_AddNews3.jsp
D: > 2023 > mushroom > train.ipynb > import tensorflow as tf
+ Code + Markdown
Restricted Mode
Do you want to install the recommended 'Python' extension from Microsoft for Python language?
Install
Show Recommendations
Cell 1 of 20
Type here to search
ENG
07:14
18-07-2023

```
                     epochs=epochs,
                     validation_data=val_generator)
```

```
Epoch 1/100
3/4 [=====================>........] - ETA: 2s - loss: 1.6384 - acc: 0.3929Epoch 1/100
4/4 [==============================] - 10s 2s/step - loss: 1.4698 - acc: 0.4483 - val_loss: 1.0777 - val_acc: 0.3929
Epoch 2/100
3/4 [=====================>........] - ETA: 1s - loss: 1.0332 - acc: 0.5208Epoch 1/100
4/4 [==============================] - 8s 2s/step - loss: 1.0291 - acc: 0.5172 - val_loss: 1.0755 - val_acc: 0.3929
Epoch 3/100
3/4 [=====================>........] - ETA: 1s - loss: 1.0557 - acc: 0.3690Epoch 1/100
4/4 [==============================] - 8s 2s/step - loss: 1.0418 - acc: 0.4310 - val_loss: 0.9711 - val_acc: 0.4286
Epoch 4/100
3/4 [=====================>........] - ETA: 1s - loss: 1.0080 - acc: 0.4881Epoch 1/100
4/4 [==============================] - 8s 2s/step - loss: 0.9621 - acc: 0.5259 - val_loss: 0.8963 - val_acc: 0.7500
Epoch 5/100
3/4 [=====================>........] - ETA: 1s - loss: 0.9362 - acc: 0.7024Epoch 1/100
4/4 [==============================] - 8s 2s/step - loss: 0.9079 - acc: 0.6724 - val_loss: 0.7610 - val_acc: 0.7500
Epoch 6/100
3/4 [=====================>........] - ETA: 1s - loss: 0.7932 - acc: 0.6771Epoch 1/100
4/4 [==============================] - 8s 2s/step - loss: 0.7347 - acc: 0.6983 - val_loss: 0.5409 - val_acc: 0.7500
Epoch 7/100
3/4 [=====================>........] - ETA: 1s - loss: 0.6168 - acc: 0.7738Epoch 1/100
4/4 [==============================] - 8s 2s/step - loss: 0.7689 - acc: 0.7069 - val_loss: 0.4215 - val_acc: 0.8571
Epoch 8/100
3/4 [=====================>........] - ETA: 1s - loss: 0.7319 - acc: 0.6562Epoch 1/100
4/4 [==============================] - 8s 2s/step - loss: 0.7705 - acc: 0.6466 - val_loss: 0.4768 - val_acc: 0.8214
Epoch 9/100
...
4/4 [==============================] - 10s 2s/step - loss: 0.1074 - acc: 0.9655 - val_loss: 0.0224 - val_acc: 1.0000
Epoch 38/100
3/4 [=====================>........] - ETA: 2s - loss: 0.2137 - acc: 0.8958Epoch 1/100
4/4 [==============================] - 10s 2s/step - loss: 0.1871 - acc: 0.9052 - val_loss: 0.0161 - val_acc: 1.0000
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

Figure 8: Training the data

In the process of training (from the Figure 8) the deep learning model to distinguish between fresh, mid-stage, and decayed mushrooms within our categorized image dataset, a neural network was employed. The model underwent a series of learning iterations in which it was exposed to numerous mushroom images. During each round of learning, the model's understanding was fine-tuned incrementally. The adjustment of learning rounds, referred to as "epochs," was carefully calibrated to strike the right balance between achieving accurate results and minimizing the time required. This training regimen was instrumental in enhancing the model's proficiency in discerning the distinct stages of mushrooms, a vital component of our research and analysis.


File Edit Selection View Go Run Terminal Help
Search
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
train.ipynb
S_AddNews3.jsp
D: > 2023 > mushroom > train.ipynb > import tensorflow as tf
+ Code + Markdown
[16]
Confusion matrix
decayed
Fresh
midstage
True label
Predicted label
4
1
6
3
6
1
4
3
0
6
5
4
3
2
1
0
Restricted Mode
Type here to search

```
                 ha="center", va="center",
                 color="white" if cm[i, j] > cm.max() / 2. else "black")
    # Add a title to the plot
    ax.set_title("Confusion matrix")
    fig.tight_layout()
    plt.show()
```

Figure 9: Confusion matrix for the CNN Model

The model's performance evaluation incorporated the application of a confusion matrix as shown in the Figure 9. This matrix emerged as a crucial tool for gaining insights into the accuracy of classifying mushroom freshness stages.

Within the confusion matrix, a comprehensive overview was obtained regarding the model's capability to correctly identify each freshness stage, encompassing "Fresh" "Mid-Stage" and "Decayed". The matrix further revealed the instances where the model misclassified freshness stages, such as confusing Mid-Stage mushrooms as Decayed. A visual representation of this matrix is provided in Figure 4.3, offering insights into areas of classification confusion. This visual aid allowed for the identification and potential mitigation of errors, ultimately contributing to the enhancement of the model's accuracy.

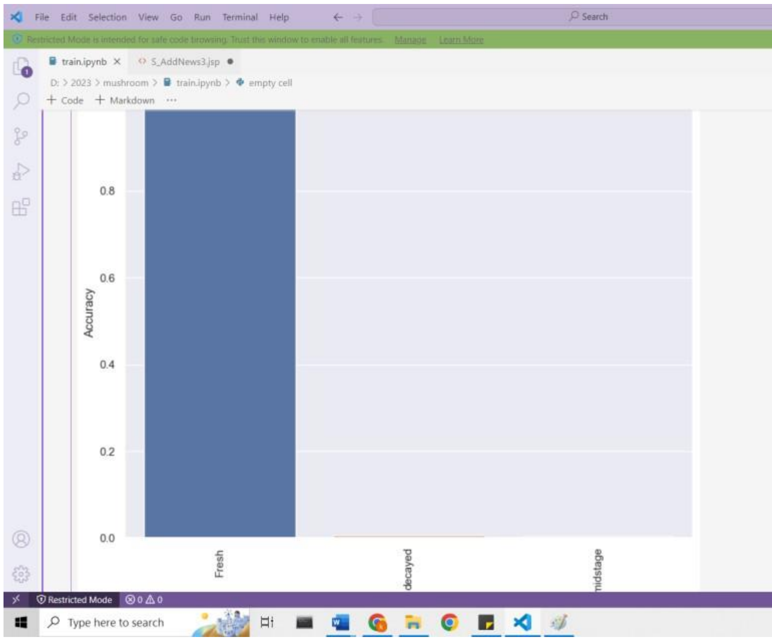


Figure 10: Bar graph for assessing model's performance using accuracy

In the bar graph as shown in the figure above where the y-axis represents accuracy and the x-axis represents different freshness stages of mushrooms,

we can visually compare how accurately our model classifies each stage.

- Freshness Stages (X-axis): On the horizontal axis, we have categories such as "Fresh" "Mild-Decayed" and "Decayed" representing the different stages of mushrooms.

- Accuracy (Y-axis): The vertical axis measures accuracy, indicating how often our model correctly classified mushrooms into each stage.

In this bar graph, each stage's bar height represents its corresponding accuracy. For instance, if the "Fresh" bar is taller than the others, it means our model is identifying the figure as fresh mushrooms. Conversely, a taller bar for "Mid-Stage" or "Decayed" would indicate those stages. This visual representation helps us quickly assess our model's performance for each freshness stage.

4.CONCLUSION

In summary, the development of the mushroom decay detection system represents the transformation of an idea into a practical solution with the primary goal of creating an accessible tool for assessing the freshness and safety of mushrooms. Mushroom images were collected and utilized to train a computer program capable of categorizing mushrooms into "Fresh," "Mid-Stage," or "Decayed." The system was designed to provide user-friendly functionality through a Jupyter Notebook interface.

Ultimately, this research embodies a step toward leveraging technology for safer exploration and understanding of the natural world. It underscores the synergy between science and technology in facilitating the exploration of the world of mushrooms while ensuring safety and knowledge.

REFERENCES

[1] Ria, N.J. Badhon, S.M.S.I. Khushbu, S.A. Akter, S. Hossain, State of art Research in Edible and Poisonous Mushroom Recognition in Proceedings of the International Conference on Computing Communication and Networking Technologies, Kharagpur, India, 6–8 July 2021.

[2] Wibowo, A.Rahayu, Y.Riyanto, A. Hidayatulloh, Classification algorithm for edible mushroom identification. In Proceedings of the International Conference on Information and Communications Technology, Yogyakarta, Indonesia, 6–7 March 2018, 250– 253.3. Mesic, A. Samec, D. Jadan, M.Bahun, Integrated morphological with molecular identification and bioactive compounds.

[3] Chitayae, N.Sunyoto, A. Performance Comparison of Mushroom Types Classification Using K-Nearest Neighbor Method and Decision Tree Method in Proceedings of the International Conference on Information and Communications Technology, Yogyakarta, Indonesia, 24–25, Nov 2020.

[4] Zahan, N.Hasan, M.Z. Malek, Reya, A Deep Learning-Based Approach for Edible, Inedible and Poisonous Mushroom Classification in Proceedings of the International Conference on Information and Communication Technology for Sustainable Development, Dhaka, Bangladesh, 27–28 February 2021.

[5] G. Xue, S. Liu, and Y. Ma, “A hybrid deep learning-based fruit classification using attention model and convolution autoencoder,” Complex & Intelligent Systems, 2020.