IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 6, June 2025

DOI: 10.17148/IJARCCE.2025.14683

Deep Learning-Based Sheep Breed Identification Using VGG16 and Architectural Enhancement

Meghana A R[1], Sneha R L[2], Navyashri P A[3], Priyadharshini L[4], Dr.Ravikiran H K[5]

Department of Electronica and Communication Engineering, Navkis College of Engineering, Hassan-573201, Karnataka, India.[1-5]

Abstract: The utilization of deep learning and transfer learning methodologies in the field of image classification has led to substantial advancements across a range of applications. This research paper presents an evaluation of the VGG16 convolutional neural network, enhanced through transfer learning and regularization techniques, for the specific task of sheep breed classification. The model was fine-tuned on a dataset of six sheep breeds, using preprocessing techniques like resizing, normalization, and augmentation. Architectural enhancements such as batch normalization, dropout, and dense layers helped reduce overfitting and improve generalization. A dropout rate of 0.5 with a batch size of 16 achieved the highest test accuracy of 80.00%. Higher dropout rates (e.g., 0.8) resulted in underfitting and lower performance. Overall, the use of transfer learning and dropout regularization significantly improved classification accuracy.

Keywords: VGG16*,* Transfer learning, Regularization Techniques, CNN, Sheep Classification.

I. INTRODUCTION

With the rapid advancement of deep learning and artificial With the rapid advancement of deep learning and artificial intelligence, the field of computer vision has undergone a transformative evolution, enabling the effective resolution of complex image-related problems across diverse domains, among which image classification remains one of the most intensely explored and successfully implemented applications. Convolutional Neural Networks (CNNs), renowned for their powerful feature extraction capabilities, have emerged as the cornerstone of modern image recognition frameworks, showcasing superior performance in domains such as facial recognition, medical imaging, plant pathology, and livestock identification. In agricultural settings, particularly in the realm of animal husbandry, breed classification holds paramount importance for optimizing breeding strategies, ensuring accurate record-keeping, facilitating health monitoring, and promoting traceability within livestock populations. Traditional methods of breed identification, reliant on human expertise and visual inspection, often suffer from inconsistencies and are highly susceptible to errors, especially when dealing with subtle inter-breed variations and large-scale datasets. Therefore, the development of an automated and intelligent system for precise animal breed identification is not only a technological necessity but also an agricultural imperative. In this study, we present a deep learning-based sheep breed identification system utilizing the VGG16 convolutional neural network architecture enhanced with additional custom layers, aiming to leverage the advantages of transfer learning to build a robust and scalable model for classifying six visually similar sheep breeds. By fine-tuning the pre-trained VGG16 network and incorporating architectural enhancements such as regularization layers and custom dense classifiers, our approach seeks to improve breed recognition accuracy and generalization in real-world scenarios.
A literature survey provides an essential foundation for understanding the current state of research, identifying gaps in existing knowledge, and analyzing methodologies employed in similar studies. Prior research exploring deep learning for livestock identification has increasingly demonstrated the effectiveness of CNN-based models, particularly when combined with transfer learning techniques. A recent study utilizing a VGG16-based transfer learning approach for goat and sheep image classification fine-tuned the model with a dataset encompassing six classes and implemented regularization techniques, such as dropout at a rate of 0.5 and L2 regularization, resulting in a notable testing accuracy of 83.9%, balanced by solid precision, recall, and F1-score metrics [1]. Furthermore, a comparative analysis of hybrid deep learning algorithms applied to dog breed identification illustrated the integration of CNNs with conventional machine learning classifiers, including SVM, KNN, and Decision Trees, where CNN-SVM hybrids consistently outperformed others. In this comparison, the InceptionV3 model achieved an accuracy of 85%, while the VGG16-based counterpart lagged at 69%, underlining the role of architectural design in influencing model performance [2]. A comprehensive survey on transfer learning categorizes its methodologies into inductive, transductive, and unsupervised learning, and distinguishes between instance-based, feature-representation-based, parameter-based, and relational knowledge transfer strategies. The survey emphasized the importance of pre-trained models like ResNet and BERT in improving performance under data-scarce conditions while also discussing challenges like domain discrepancies and negative transfer effects [3].
Additional studies have validated the effectiveness of fine-tuning VGG models such as VGG16 and VGG19 for binary

image classification tasks like dog-versus-cat recognition, where transfer learning facilitated domain-specific adaptations using relatively smaller datasets, minimizing computational cost while preserving classification performance [4]. In another domain, the RGBD-Dog framework introduced a deep learning-based method to estimate canine 3D pose using RGBD (color and depth) sensors, demonstrating the application of deep networks beyond simple classification to more complex pose estimation in non-human subjects, thereby enriching animal behavior studies and veterinary practices [5]. Notably, an advanced breed classification study employed a pre-trained InceptionV3 model for sheep breed identification, fine-tuned on a specialized dataset to achieve a high accuracy of 94.13%, emphasizing the capability of deep CNNs in capturing subtle visual patterns necessary for distinguishing between similar breeds in agricultural settings [6]. Similarly, the InceptGI framework applied InceptionV3 for Indian goat breed classification, achieving 95% accuracy, and reinforcing the role of transfer learning and deep feature extraction in agricultural automation and genetic diversity tracking [7]. Further supporting the use of CNNs in real-world environments, an on-farm sheep breed identification study utilized deep learning models trained on images captured under uncontrolled farm conditions, achieving robust performance despite challenges like occlusion and variable lighting, and illustrating the potential of these methods for integration into smart farming systems [8].

The primary objective of this research is to develop a deep learning-based sheep breed classification system utilizing the VGG16 convolutional neural network architecture enhanced with customized architectural modifications to improve its classification performance for six different sheep breeds. This study aims to demonstrate the efficacy of transfer learning in adapting a pre-trained model originally trained on a large-scale dataset (ImageNet) to a domain-specific task with limited image samples by fine-tuning selected layers and appending new layers tailored for the specific problem. Furthermore, the model seeks to achieve high generalization ability across similar-looking sheep breeds by incorporating regularization techniques such as dropout and L2 norm, thereby minimizing overfitting and improving robustness. By accurately identifying sheep breeds through image analysis, this work endeavors to contribute to the field of precision livestock farming by enabling automation, reducing human error, and facilitating data-driven herd management practices.

II. IMPLEMENTATION

A. Theory

- *Convolution Neural Network (CNN)*

Convolutional Neural Networks (CNNs) have revolutionized the field of computer vision due to their exceptional ability to interpret and analyze visual data. These deep learning architectures are specifically designed to process data that come in the form of multiple arrays, such as RGB images. Inspired by the visual cortex of animals, CNNs can automatically and adaptively learn spatial hierarchies of features, from low-level edges to high-level object representations. They are composed of multiple layers that perform operations designed to extract and condense the essential information in images. The architecture is modular, with each layer playing a specific role in the hierarchical learning of visual features.

At the core of CNNs are the convolutional layers, which are responsible for detecting local patterns such as edges, corners, and textures in images. These layers use filters (or kernels) that slide across the input image to perform convolution operations. Each filter is a small matrix that multiplies with corresponding sections of the input image, and the resulting dot products form a feature map. Multiple filters in a convolutional layer allow the model to learn different features from the same input. Multiple filters in a convolutional layer allow the model to learn different features from the same input. These learned filters are capable of recognizing specific patterns, and as the network goes deeper, the filters detect increasingly abstract and complex features. The advantage of using convolutions is parameter sharing – instead of assigning a unique weight for each pixel, the same filter is used across the entire image, drastically reducing the number of parameters. Additionally, local connectivity ensures that each neuron in the convolutional layer is connected to only a small region of the previous layer, preserving spatial relationships and reducing computation. Following each convolutional operation, an activation function is typically applied to introduce non-linearity into the network, enabling it to learn complex mappings. The Rectified Linear Unit (ReLU) is the most commonly used activation function in CNNs. It transfoms all negative values in the feature map to zero, while keeping positive values unchanged, helping the network to converge faster and avoid vanishing gradients. After activation, pooling layers are usually introduced to reduce the spatial dimensions of the feature maps. Max pooling, the most widely used pooling method, selects the maximum value in each small patch of the feature map, reducing its size while preserving the most important information. This process helps to make the model invariant to small translations and distortions in the image, which is useful for recognizing objects regardless of their position or orientation. Pooling also reduces the number of parameters and computation in the network, which helps in speeding up the training process and reducing overfitting.

After several layers of convolutions and pooling, the high-level features extracted from the image are passed through one or more fully connected (dense) layers. These layers treat the extracted features as a flat vector and perform high-level reasoning based on them. Every neuron in a fully connected layer is connected to every neuron in the previous layer, allowing for complex decision boundaries to be formed. The final layer of a CNN is the output layer, whose structure depends on the type of problem. For binary classification, a single neuron with a sigmoid activation function is used, while for multi-class classification tasks, a softmax activation is applied across multiple output neurons to produce probability distributions. This layer outputs the predicted class or label for the input image. The combination of convolutional layers, activation functions, pooling layers, and fully connected layers enables CNNs to excel in understanding and classifying visual data by mimicking the hierarchical processing of the human visual system [9][10].

- *Visual Geometry Group (VGG)*

VGG16 is a deep Convolutional Neural Network (CNN) architecture that was introduced by Simonyan and Zisserman from the University of Oxford in 2014. It gained widespread popularity due to its simplicity and excellent performance in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2014. The "16" in VGG16 refers to the 16 layers that have learnable parameters—13 convolutional layers and 3 fully connected layers. The network emphasizes using small receptive fields (3×3 convolution filters), which are stacked on top of each other to capture intricate features while maintaining computational efficiency. One of the key ideas behind VGG16 is that multiple small filters can effectively emulate the effect of larger filters (like 5×5 or 7×7) while reducing the number of parameters. This architecture provides a consistent and uniform structure, making it easier to understand and implement. VGG16 has become a foundational model in the field of deep learning and is often used for transfer learning due to its ability to generalize well on various image classification tasks.VGG16 starts with an input image of size 224×224×3 (RGB), and applies a series of convolutional layers to extract features. Each convolutional layer in VGG16 uses a 3×3 kernel with stride 1 and padding to maintain spatial resolution. This is followed by a ReLU activation function, which adds non-linearity and helps the network learn complex patterns. The use of several convolutional layers stacked one after the other allows the model to build a deeper understanding of the input data. The number of filters increases progressively as we go deeper into the network, starting from 64 and going up to 512. The convolutional layers are grouped into five blocks, and each block is followed by a max pooling layer of size 2×2 with stride 2, which reduces the spatial dimensions. This arrangement enables the model to gradually reduce the spatial dimensions while increasing the depth of the feature maps, allowing it to extract high-level semantic information as the depth increases. The consistent use of 3×3 kernels ensures that the model captures local patterns while keeping the number of parameters manageable [11][12].

After the five convolutional blocks and pooling operations, the resulting feature maps are flattened into a one-dimensional vector and passed through three fully connected (dense) layers. The first two fully connected layers in VGG16 have 4096 neurons each, while the final fully connected layer has 1000 neurons corresponding to the number of classes in the original ImageNet dataset. A softmax activation function is applied in the output layer to generate probability distributions across the classes During training, the network learns to assign higher probabilities to the correct classes while minimizing the loss using backpropagation. The design of these dense layers helps in integrating the spatially distributed features extracted by the convolutional layers into a final decision. VGG16 ends with high-level abstraction, making it suitable for fine-grained classification problems. Though the fully connected layers are computation-heavy, they play a vital role in classification performance. VGG16 ends with high-level abstraction, making it suitable for fine-grained classification problems. Though the fully connected layers are computation-heavy, they play a vital role in classification performance.

Due to its well-established architecture and pre-trained weights available on large datasets like ImageNet, VGG16 is widely used in transfer learning. In transfer learning scenarios, the convolutional base of VGG16 is typically used as a fixed feature extractor by freezing its weights, and only the final classification layers are retrained on a new dataset. This significantly reduces training time and requires less labeled data while achieving high accuracy on specialized tasks. The utilization of transfer learning in conjunction with the VGG16 architecture has been observed to yield notable enhancements in both the efficiency and effectiveness of the image classification procedure. The VGG16 architectures have gained significant recognition due to their extensive depth and well-organized structure. These architectures consist of numerous convolutional and fully connected layers, which enable them to effectively extract hierarchical features from images. The process of integrating transfer learning typically comprises two primary stages: feature extraction and fine-tuning. In the context of feature extraction, it is common practice to employ pre-trained models such as VGG16 as static feature extractors. The initial classification layers have been eliminated, while the subsequent layers responsible for encoding feature representations have been preserved. The architecture of the modified model can subsequently be linked to novel classification layers that are customized to suit the particular task under consideration. The utilization of this approach proves to be highly advantageous when the novel task exhibits certain resemblances to the initial task on which the model underwent training.

B. Methodology

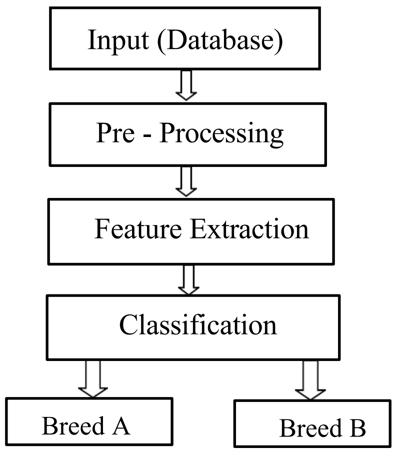


Fig. 1. Block Diagram for process of Sheep Breed Identification.

The proposed technique outlines a comprehensive and systematic approach for designing, training, evaluating, and interpreting a deep learning model for image classification utilizing the VGG16 architecture. This methodology harnesses the power of transfer learning, combined with advanced techniques such as data augmentation, fine-tuning, and regularization, to achieve high-accuracy classification results. By building on the pre-trained knowledge embedded within VGG16, the model is adapted to the specific dataset with greater efficiency and reduced training time. In this section, we present a comprehensive elucidation of each individual step encompassed within the framework of this particular methodology.

The initial phase of this study involves the training and fine-tuning of the VGG-16 model architecture. The choice of utilizing pre-trained models is pivotal in modern deep learning workflows, particularly in scenarios involving limited labeled datasets. In this research, the VGG-16 model—pre-trained on the large-scale ImageNet dataset—is employed as the base model. This network is known for its ability to extract rich hierarchical features from input images. To retain its feature extraction capabilities, the top classification layers are removed and the convolutional layers are frozen. Freezing these layers prevents them from being updated during training, ensuring that the learned weights from ImageNet are preserved and the model focuses solely on training the newly added classification layers.

On top of the frozen VGG16 base, a new custom neural network architecture is constructed using the Sequential API. The architecture begins with a Flatten layer, which transforms the high-dimensional feature maps output by the base model into a one-dimensional vector, suitable for fully connected layers. Following this, a Dense layer comprising 512 units is integrated with a ReLU activation function, a widely used non-linear activation that introduces sparsity and mitigates vanishing gradient problems. To reduce overfitting and enhance the generalization ability of the model, a Dropout layer with a dropout rate of 0.5 is incorporated. During training, this layer randomly disables 50% of the neurons in the layer, discouraging the network from becoming overly reliant on specific neurons or patterns, thereby promoting robustness. The final output layer consists of a Dense layer neurons, each representing a target sheep breed class. This layer uses the softmax activation function to output a probability distribution across all classes, enabling multiclass classification Once the model architecture is finalized, it is compiled with three critical components: the optimizer, the loss function, and evaluation metrics. In this study, the Adam optimizer is selected due to its efficiency and adaptive learning rate properties, enabling stable and fast convergence. The categorical cross-entropy loss function is employed, which is suitable for multiclass classification problems as it quantifies the distance between the predicted class probabilities and the true one-hot encoded labels. Accuracy is used as the primary evaluation metric, providing an intuitive measure of the model's performance during training and validation. Data preprocessing is a fundamental step in the model pipeline. To augment the training data and enhance its diversity, ImageDataGenerator is utilized to apply various data augmentation techniques such as rotation, width and height shifts, shear transformations, zooming, and horizontal flipping. These transformations prevent the model from memorizing the training data and help it generalize better to unseen samples. For validation and testing, only rescaling is performed to maintain the natural distribution of the data, ensuring a fair assessment of model performance. A **batch size** of 32 is selected to balance memory efficiency with convergence speed during training. Both training and validation datasets are loaded from their respective directories, with image resizing to 224x224 to comply with VGG16 input requirements. The validation data remains unshuffled and unaugmented to reflect the true distribution of data during evaluation.

Following model training for **25** epochs, the model is evaluated on an independent test dataset to determine its generalization performance. Evaluation includes calculating the test accuracy and loss, along with generating a classification report containing precision, recall, and F1-score for each class. Additionally, a confusion matrix is constructed and visualized using a heatmap to analyze classification errors and model reliability across classes.

Finally, the trained model is saved in ".h5" format to enable reuse without the need for retraining, making it suitable for real-world deployment scenarios.

III. RESULTS

Deep learning methodology advancements have led to major successes in image classification. The aim of this investigation was to assess the effectiveness of a sheep-goat breed classification model using VGG16 model incorporating transfer learning with a variety of regularization techniques and dropout rates. We thoroughly evaluated the impacts of several techniques on the precision, recall, F1-scores, and overall accuracy of the model by carrying out a series of meticulous experiments [12][13]. The database consists of 411 images with 6 classes out of it 62 images are dropped from the dataset due to unknown format, resulting in 349 images. Out of 349 images, 291 were used for training, 33 for validation, and 25 for testing [14].

We examined a number of scenarios in our study, including dropout rates of 0.1, 0.3, and 0.5 and the use of L1 and L2 regularization techniques. Each scenario had different outcomes, which provided insights on the complex interactions between regularization, dropout, and model performance. The results are obtained with the aid of Google Colab using a computer system with an Intel Core i3 processor operating at a clock speed of 1.80 GHz.

The Figure 2 represents sample image of each class. The Table 1-5 represents performance parameter. The Table 6 provides compression of accuracy and loss of proposed models. The Figure 2-6 represents graph of Loss and Accuracy for Training and Validation. Figure from 7-11 represents Confusion matrix of different proposed model.

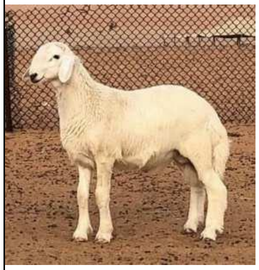
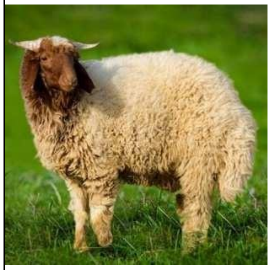
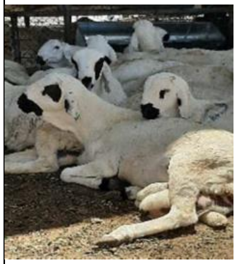
Barbery	Goats - A_ardy	Sawakny Red
Harry 	Ne_amy 	Swakny White 

Fig. 1. Sample Images form 6 classes.

DOI: 10.17148/IJARCCE.2025.14683

TABLE I. PERFORMANCE PARAMETERS WITH DROPOUT=0.1

With dropout=0.1 And Batch Size=32	**Precision rate**	**Recall rate**	**F1-score**	**Support**
Barbery	0.67	1.00	0.80	4
Goats - A_ardy	1.00	1.00	1.00	4
Harry	0.75	0.75	0.75	4
Ne_amy	0.50	0.75	0.60	4
Sawakny Red	1.00	0.75	0.86	4
Swakny White	1.00	0.40	0.57	5
accuracy			0.76	25
macro avg	0.82	0.78	0.76	25
weighted avg	0.83	0.76	0.76	25

TABLE II. PERFORMANCE PARAMETERS WITH DROPOUT=0.1

With Dropout=0.1 And Batch Size=16	**precision rate**	**recall rate**	**F1-score**	**support**
Barbery	0.57	1.00	0.73	4
Goats - A_ardy	0.80	1.00	0.89	4
Harry	0.75	0.75	0.75	4
Ne_amy	0.75	0.75	0.75	4
Sawakny Red	1.00	0.50	0.67	4
Swakny White	1.00	0.60	0.71	5
accuracy			0.76	25
macro avg	0.81	0.77	0.76	25
weighted avg	0.82	0.76	0.76	25

TABLE III. PERFORMANCE PARAMETERS WITH DROPOUT=0.2

With Dropout=0.2 And Batch Size=32	**Precision Rate**	**Recall Rate**	**F1-score**	**Support**
Barbery	0.50	1.00	0.67	4
Goats - A_ardy	0.80	1.00	0.89	4
Harry	0.80	1.00	0.89	4
Ne_amy	1.00	0.75	0.86	4
Sawakny Red	1.00	0.25	0.40	4
Swakny White	1.00	0.60	0.75	5
accuracy			0.76	25
macro avg	0.85	0.77	0.74	25
weighted avg	0.86	0.76	0.74	25

TABLE IV. PERFORMANCE PARAMETERS WITH DROPOUT=0.2

With Dropout=0.2 and Batch Size=16	**Precision Rate**	**Recall Rate**	**F1-Score**	**Support**
Barbery	0.57	1.00	0.73	4
Goats - A_ardy	0.80	1.00	0.89	4
Harry	0.75	0.75	0.75	4
Ne_amy	0.75	0.75	0.75	4
Sawakny Red	1.00	0.50	0.67	4
Swakny White	1.00	0.60	0.75	5
accuracy			0.76	25
macro avg	0.81	0.77	0.76	25
weighted avg	0.82	0.76	0.76	25

TABLE V. PERFORMANCE PARAMETERS WITH DROPOUT= 0.5

With Dropout=0.5 and Batch Size=32	**Precision Rate**	**Recall Rate**	**F1-score**	**Support**
Barbery	0.50	0.75	0.60	4
Goat – A_ardy	1.00	1.00	1.00	4
harry	0.60	0.75	0.67	4
Ne_amy	0.60	0.75	0.67	4
Sawakny Red	1.00	0.50	0.67	4
Swakny White	1.00	0.60	0.75	5
Accuracy			0.72	25
macro Avg	0.78	0.72	0.72	25
Weighted Avg	0.79	0.72	0.73	25

TABLE VI. PERFORMANCE PARAMETERS WITH DROPOUT= 0.5

With Dropout=0.5 and Batch Size=16	**Precision Rate**	**Recall Rate**	**F1-score**	**Support**
Barbery	0.57	1.00	0.73	4
Goat – A_ardy	1.00	1.00	1.00	4
harry	0.80	1.00	0.89	4
Ne_amy	0.75	0.75	0.75	4
Sawakny Red	1.00	0.50	0.67	4
Swakny White	1.00	0.60	0.75	5
Accuracy			0.80	25
macro Avg	0.85	0.81	0.80	25
Weighted Avg	0.86	0.80	0.80	25

TABLE VII. PERFORMANCE PARAMETERS WITH DROPOUT= 0.5

With Dropout=0.8 and Batch Size=32	**Precision Rate**	**Recall Rate**	**F1-score**	**Support**
Barbery	0.80	1.00	0.89	4
Goat – A_ardy	0.80	1.00	0.89	4
harry	0.50	1.00	0.67	4
Ne_amy	1.00	0.25	0.40	4
Sawakny Red	0.00	0.00	0.00	4
Swakny White	0.50	0.60	0.55	5
Accuracy			0.64	25
macro Avg	0.60	0.64	0.56	25
Weighted Avg	0.60	0.64	0.56	25

TABLE VII. PERFORMANCE PARAMETERS WITH DROPOUT= 0.5

With Dropout=0.5 and Batch Size=32	**Precision Rate**	**Recall Rate**	**F1-score**	**Support**
Barbery	0.5	0.75	0.60	4
Goat – A_ardy	1.00	1.00	1.00	4
harry	0.60	0.75	0.67	4
Ne_amy	0.60	0.75	0.67	4
Sawakny Red	1.00	0.50	0.67	4
Swakny White	1.00	0.60	0.75	5
Accuracy			0.72	25
macro Avg	0.78	0.72	0.72	25
Weighted Avg	0.79	0.72	0.73	25

TABLE IX. COMPRESSION OF ACCURACY AND LOSS

Accuracy and Loss	**Training Loss**	**Training Accuracy**	**Validation Loss**	**Validation Accuracy**	**Testing Loss**	**Testing Accuracy**
Dropout 0.1 Batch size=32	0.4914	0.8225	0.7331	0.7576	0.8156	0.7599
Dropout 0.1 Batch Size=16	0.3429	0.8920	0.8588	0.7879	0.5176	0.7599
Dropout 0.2 Batch Size=32	0.5831	0.8108	0.7264	0.8182	0.7674	0.7599
Dropout 0.2 Batch size=16	0.4030	0.8779	0.9139	0.7273	0.5889	0.7599
Dropout 0.5 Batch Size=32	0.8968	0.6810	0.8414	0.7576	0.8209	0.7200
Dropout 0.5 Batch Size=16	1.1247	0.5871	0.8722	0.7879	0.7703	0.8000
Dropout 0.8 Batch Size=32	1.6404	0.2917	1.4333	0.5455	1.4058	0.6399
Dropout 0.8 Batch Size=16	1.7059	0.2625	1.6298	0.3636	1.6426	0.3199

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁙ Peer-reviewed & Refereed journal ⁙ Vol. 14, Issue 6, June 2025

DOI: 10.17148/IJARCCE.2025.14683

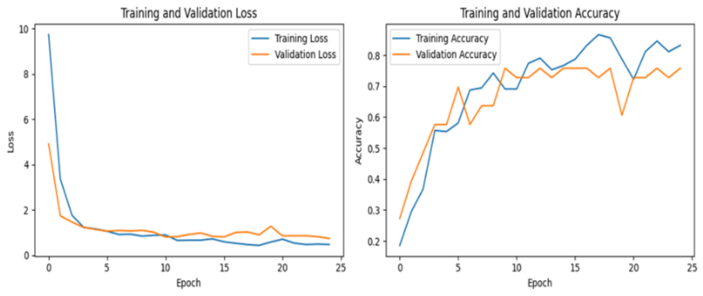


Fig. 3. Loss and Accuracy Graph for Training and Validation data for Dropout =0.1 Batch Size=32

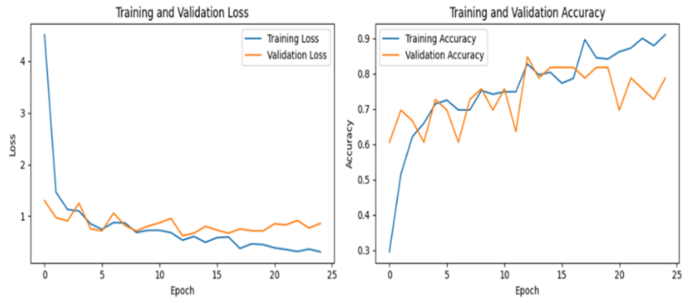


Fig. 4. Loss and Accuracy Graph for Training and Validation data for Dropout =0.1 Batch Size=16

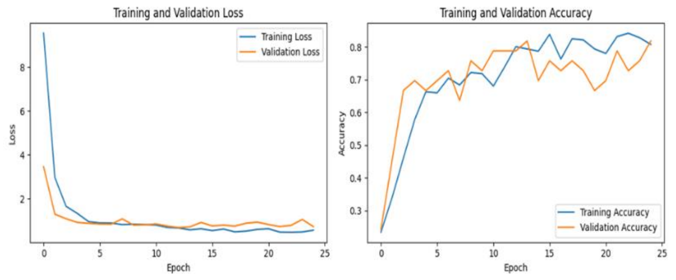


Fig. 5. Loss and Accuracy Graph for Training and Validation data for Dropout =0.2 Batch Size=32

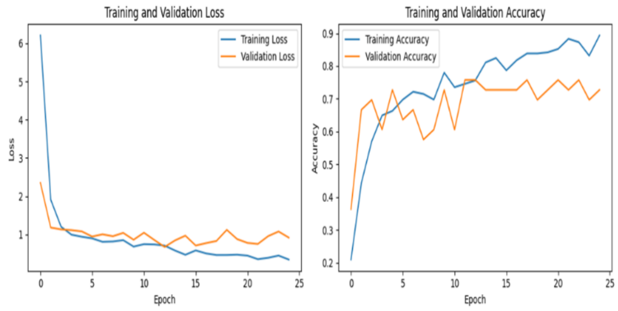


Fig.6. Loss and Accuracy Graph for Training and Validation data for Dropout 0.2 Batch Size=16

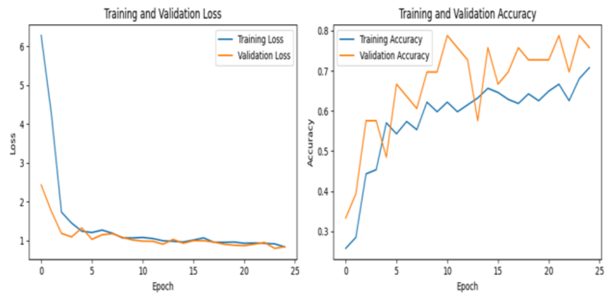


Fig. 7. Loss and Accuracy Graph for Training and Validation data Dropout 0.5 Batch Size=32

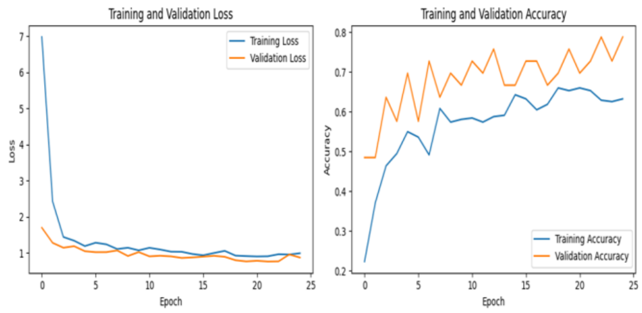


Fig. 8. Loss and Accuracy Graph for Training and Validation data Dropout 0.5 Batch Size=16

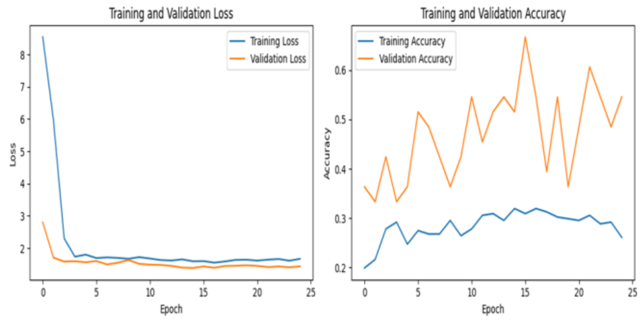


Fig. 9. Loss and Accuracy Graph for Training and Validation data Dropout 0.8 Batch Size=32

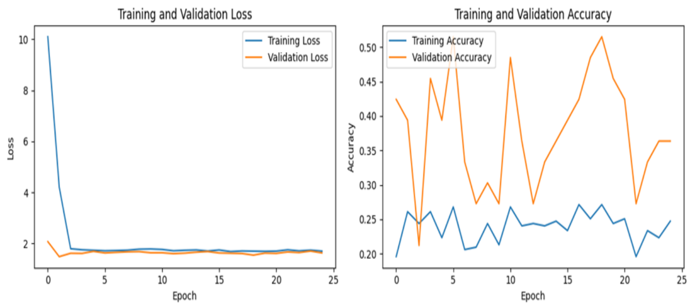


Fig. 10. Loss and Accuracy Graph for Training and Validation data for Dropout 0.8 Batch Size=16

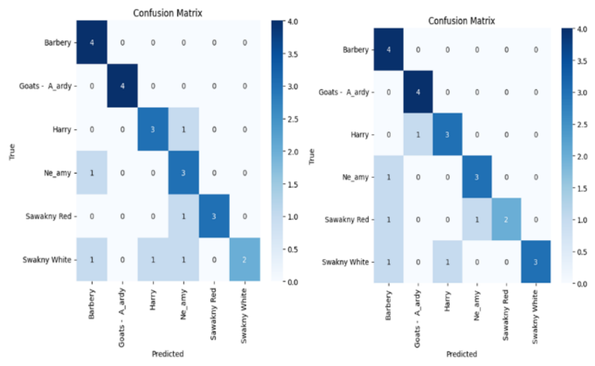


Fig. 11. Confusion Matrix for Dropout =0.1, batch size=32 & 16

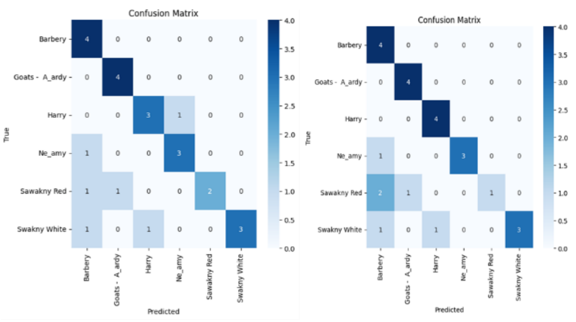


Fig. 12. Confusion Matrix for Dropout =0.2, batch size=32 & 16

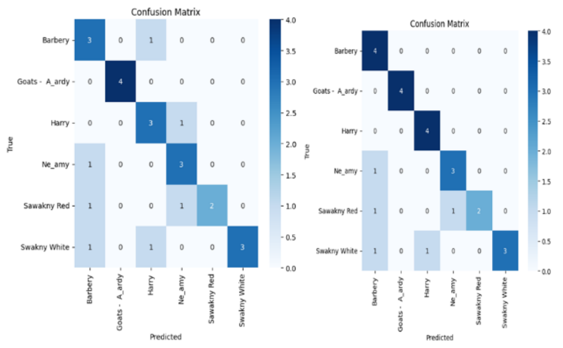


Fig. 13. Confusion Matrix for Dropout =0.5, batch size=32 & 16

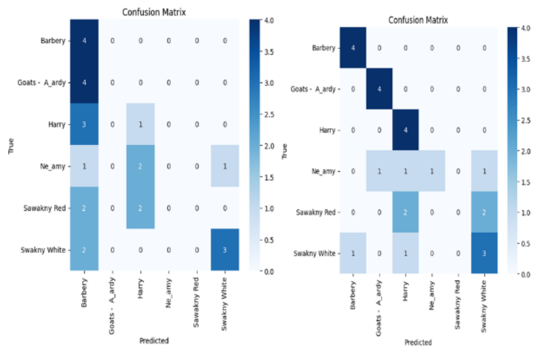


Fig. 14. Confusion Matrix for Dropout =0.8, batch size=32 & 16

Thus, the model's performance is heavily influenced by the dropout rate and regularization strategy selected. With a testing accuracy of 0.839, the model with L2 Regularization outperformed the other models. These findings further emphasize how crucial it is to adjust hyper-parameters in order to maximize model performance.

IV. CONCLUSION

In conclusion, the methodology presented here shows how to develop, train, evaluate, and analyze a deep learning model for the classification of images in a methodical manner. This methodology offers a thorough approach to getting accurate classification results while avoiding overfitting and boosting generalization capabilities. It does so by leveraging the strengths of the VGG16 architecture, transfer learning, data augmentation, and fine-tuning. After performing a data analysis, it was discovered that using a dropout rate of 0.5 and L2 regularization consistently produced an accuracy level of 80%. Applying the observed design to a variety of different sheep breeds revealed good levels of precision, recall, and F1- scores. The aforementioned findings emphasize the need for balanced regularization techniques to create recognition models that are accurate and reliable. Furthermore, the results of our analysis of the compression of accuracy and loss showed that regularization with dropout rates and L2 were able to successfully balance the trade-off between training accuracy and validation performance. One can use a metaheuristic technique for extraction features or adopt hyper- parameter tuning in the future.

REFRENCES

[1]. Ravi, L. S., K. Bindu, Farzana Praveen, and T. Tanuja. "Study on VGG16 Transfer learning Model for Goat/Sheep Image Classification." In 2023 International Conference on Recent Advances in Science and Engineering Technology (ICRASET), pp. 1-8. IEEE, 2023.

[2]. Deane, Jake, Sinéad Kearney, Kwang In Kim, and Darren Cosker. "Rgbt-dog: A parametric model and pose prior for canine body analysis data creation." In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 6056-6066. 2024.

[3]. Zhuang, Fuzhen, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. "A comprehensive survey on transfer learning." Proceedings of the IEEE 109, no. 1 (2020): 43-76.

[4]. Kearney, Sinead, Wenbin Li, Martin Parsons, Kwang In Kim, and Darren Cosker. "Rgbd- dog: Predicting canine pose from rgbd sensors." In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 8336-8345. 2020.

[5]. Wang, I-Hung, Kuang-Chyi Lee, and Shinn-Liang Chang. "Images classification of dogs and cats using fine-tuned VGG models." In 2020 IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE), pp. 230-233. IEEE, 2020.

[6]. Jwade, Sanabel Abu, Andrew Guzzomi, and Ajmal Mian. "On farm automatic sheep breed classification using deep learning." Computers and Electronics in Agriculture 167 (2019): 105055.

[7]. Koklu, Murat, Ilkay Cinar, Y. Selim Taspinar, and Ramazan Kursun. "Identification of sheep breeds by CNN-based pre-trained InceptionV3 model." In 2022 11th Mediterranean Conference on Embedded Computing (MECO), pp. 01-04. IEEE, 2022.

[8]. [Mandal, Satyendra Nath, Pritam Ghosh, Kaushik Mukherjee, Sanket Dan, Subhranil Mustafi, Kunal Roy, Dilip Kumar Hajra, and Santanu Banik. "InceptGI: a ConvNet-Based classification model for identifying goat breeds in India." Journal of The Institution of

[9]. HK, Ravikiran, Wilfred John Vaz, Sathisha MS, Prashantha SJ, and Madhu KM. "Hybrid VGG16-Xception Model vs. Single Architecture Transfer Learning for Flower Image Classification." International Journal of Horticultural Science and Technology 12, no. 3 (2025): 323-342.

[10]. Manasa, A., and H. K. Ravikiran. "SC-CNN: Advancing Corn Crop Disease Classification-A Comparative Study with CNN." In 2024 International Conference on Recent Advances in Science and Engineering Technology (ICRASET), pp. 1-6. IEEE, 2024.

[11]. Shalini, I. S., and H. K. Ravikiran. "Random search Based Hyperparamerter tunned VGG16 Architecture for Poultry Breed Image Classification." In 2024 International Conference on Intelligent Algorithms for Computational Intelligence Systems (IACIS), pp. 1-7. IEEE, 2024.

[12]. Ravikiran, H. K., J. Jayanth, M. S. Sathisha, and K. Bindu. "Optimizing sheep breed classification with bat algorithm-tuned cnn hyperparameters." SN Computer Science 5, no. 2 (2024): 219.

[13]. Ravikiran, H. K., J. Jayanth, M. S. Sathisha, G. H. Yogeesh, and R. Dileep. "Classification of Crop Across Heterogeneous Landscape Through Experienced Artificial Bee Colony." SN Computer Science 5, no. 4 (2024): 428.

[14]. Siyamek, A.Y, Sheep Breeds Dataset, Kaggle, 2023. Available at: https://www.kaggle.com/datasets/alaayusufsiyamek/sheep-breeds- dataset.