

MACHINE LEARNING BASED DIAGNOSIS OF LUMPY SKIN DISEASE

Rakshitha S¹

¹Institute/Organization: The Oxford college of engineering Bangalore, India.

ABSTRACT

Lumpy Skin Disease (LSD) is a highly contagious viral infection that significantly impacts cattle and poses a threat to the global livestock industry. Traditional diagnostic methods are often slow and prone to inaccuracies, which delay timely interventions and exacerbate the spread of the disease. This research introduces a novel machine-learning-based diagnostic system leveraging Convolutional Neural Networks (CNNs) and MobileNet architectures to detect LSD efficiently and accurately. The system utilizes image-based analysis of cattle skin to classify them as either affected or healthy, reducing reliance on subjective or time-consuming laboratory tests. This innovative approach offers a scalable and effective solution for veterinary practices and livestock management, helping to mitigate economic losses and enhance animal health outcomes.

1. INTRODUCTION

Lumpy Skin Disease has become a growing concern in the livestock industry due to its rapid spread and economic implications. It is caused by the Lumpy Skin Disease Virus (LSDV), a member of the Poxviridae family, which manifests as nodules on the skin of cattle.

LSD leads to reduced milk and meat production, abortions in pregnant cattle, and, in severe cases, sterility or death. The disease is enzootic in various parts of Africa, but its recent emergence in previously unaffected regions has highlighted the need for efficient control measures. Traditional diagnostic methods rely heavily on clinical observations and laboratory tests, which are time-intensive and require specialized expertise. As a result, there is often a delay in identifying and isolating infected animals, leading to the rapid spread of the disease within cattle populations. Machine learning has emerged as a transformative tool in disease diagnostics, offering the potential for rapid, accurate, and scalable solutions.

This research focuses on utilizing machine learning models, particularly CNNs and MobileNet, to automate the diagnosis of LSD using image-based data. By integrating advanced algorithms with robust datasets, the proposed system aims to address the limitations of traditional diagnostic approaches. The motivation for this research stems from the urgent need to improve livestock health management practices and reduce the economic burden caused by LSD outbreaks. Leveraging the power of machine learning, this study proposes a diagnostic system that can be easily deployed in real-world settings, providing farmers and veterinarians with a reliable tool for early disease detection.

2. LITERATURE REVIEW

The application of machine learning in agriculture and animal health has gained significant traction in recent years. Researchers have explored various algorithms to address challenges in disease detection, yield prediction, and environmental monitoring. For example, Gupta et al. (2022) demonstrated the efficacy of CNNs in identifying diseases in paddy crops, achieving high accuracy in detecting conditions like Rice Tungro and Brown Spot. Similarly, Singh et al. (2022) utilized deep learning models to detect apple leaf diseases, achieving an accuracy of 99.2% with a dataset of over 50,000 images. In the context of LSD, Afshari Safavi (2022) explored the role of geospatial and meteorological features in predicting disease outbreaks. Using an Artificial Neural Network (ANN) model, the study achieved 97% accuracy in forecasting LSD occurrences. However, this approach focused on environmental factors rather than direct diagnosis through visual inspection of cattle. Traditional machine learning algorithms, such as Random Forest and Decision Trees, have also been employed for animal disease diagnostics but often lack the robustness required for complex image-based tasks. This research builds on these advancements by combining the feature extraction capabilities of CNNs with the efficiency of MobileNet. The proposed system not only addresses the challenges of diagnosing LSD but also provides a scalable solution that can be extended to other diseases in livestock.

3. METHODOLOGY

The proposed system employs a combination of machine learning and deep learning algorithms to diagnose LSD. The architecture is designed to handle image-based data, enabling rapid and accurate classification of cattle as either healthy or affected by the disease. The dataset used for this study is sourced from Kaggle and includes labeled images of healthy and LSD-affected cattle. The methodology involves several key stages, beginning with data preprocessing to ensure the quality and consistency of input data. This includes resizing images, normalization, and augmentation to enhance model performance. Convolutional Neural Networks (CNNs) are employed to extract visual features from the images, while MobileNet, a lightweight architecture, ensures computational efficiency without compromising accuracy. In addition to

image-based classification, the system integrates traditional machine learning algorithms such as Naive Bayes and Extra Trees for analyzing clinical and diagnostic features. This dual-layered approach enhances the robustness of the system, ensuring reliable predictions even in challenging scenarios.

4. SYSTEM IMPLEMENTATION

The system is divided into several modules, each serving a specific function in the diagnostic process. The data upload module allows users to input cattle images and associated clinical data, while the preprocessing module cleans and transforms this data for analysis. The core diagnostic engine employs CNNs and MobileNet to classify images, supported by additional algorithms for feature-based predictions. The results are displayed through a user-friendly interface, providing actionable insights for veterinarians and farmers.

1. Data Preprocessing

Before feeding images into the machine learning model, the system undertakes several preprocessing steps to ensure that all images are of consistent quality:

Resizing: All input images are resized to a fixed dimension, which is crucial because neural networks typically require uniform input sizes. This resizing helps maintain consistency across the dataset and ensures that the model can process all images uniformly. Techniques such as nearest neighbor interpolation are often used to avoid distorting the images excessively during this step.

Normalization: This step adjusts the pixel values of the images to a common scale, usually between 0 and 1. Normalization helps in reducing biases caused by varying lighting conditions or exposure levels in the images, thereby improving the model's performance.

Augmentation: To further enhance the robustness of the model, data augmentation techniques are applied. These may include random rotations, flips, or adjustments in brightness and contrast. Augmentation not only increases the diversity of the training dataset but also helps the model generalize better to new, unseen images that may vary in quality.

2. Image Quality Enhancement

The system may incorporate image enhancement techniques to improve the quality of input images:

Denoising: Images that are noisy or have low visibility can be processed using denoising algorithms to reduce unwanted artifacts. This step is critical for ensuring that important features required for diagnosis are preserved.

Sharpening: Applying sharpening filters can enhance edges and details within an image, making it easier for the model to identify critical features related to LSD.

3. Robust Model Architecture

The architecture of the proposed system is designed to be resilient against variations in image quality:

Convolutional Neural Networks (CNNs): CNNs are particularly effective at extracting features from images regardless of their quality due to their hierarchical structure. They can learn to focus on relevant features while disregarding noise, making them suitable for handling diverse image inputs.

MobileNet: This lightweight architecture is optimized for performance without compromising accuracy. It allows for efficient processing of images, even those that may be lower in resolution or quality.

4. Handling Different Aspect Ratios

In cases where images have varying aspect ratios, the system can implement strategies such as padding. Padding involves adding borders around an image to achieve a uniform size without cropping essential parts of the image. This ensures that all relevant information is retained while still fitting into the required input dimensions of the model.

The training phase involves fine-tuning the CNN and MobileNet models using a labeled dataset. Evaluation metrics such as accuracy, precision, recall, and F1-score are used to assess the performance of the system. The final model is validated on a separate test dataset to ensure its reliability in real-world applications.

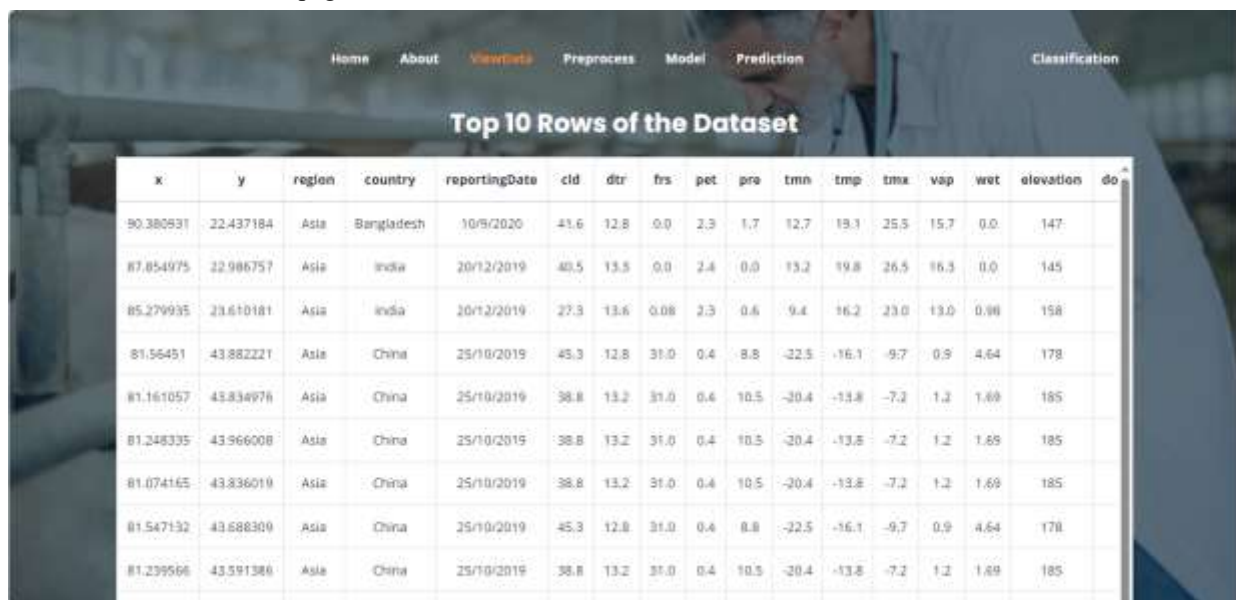
Screenshots:



ABOUT PAGE: Here we can read about our project.



View Data: In the Viewdata page, users can view the skin dataset.

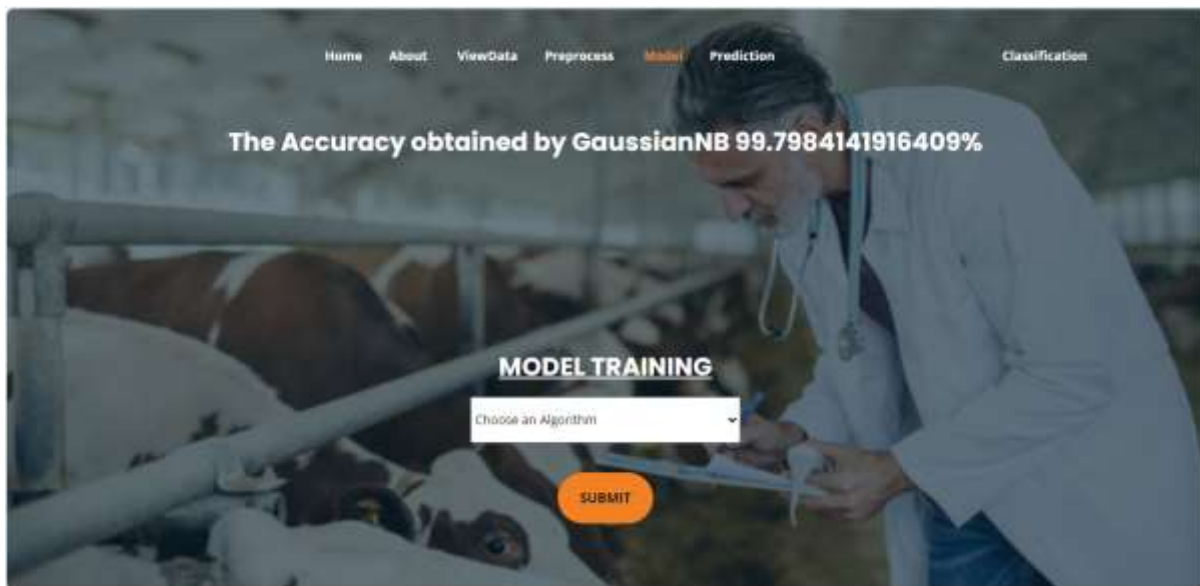


x	y	region	country	reportingDate	cid	dtr	frs	pet	pro	tmn	tmp	tmx	vap	wet	elevation	do
90.380531	22.437184	Asia	Bangladesh	10/9/2020	41.6	12.8	0.0	2.3	1.7	12.7	19.1	25.5	15.7	0.0	147	
87.854975	22.986757	Asia	India	20/12/2019	40.5	13.5	0.0	2.4	0.0	13.2	19.8	26.5	16.3	0.0	145	
85.279935	23.610181	Asia	India	20/12/2019	27.3	13.6	0.08	2.3	0.6	9.4	16.2	23.0	13.0	0.98	158	
81.56451	43.882221	Asia	China	25/10/2019	45.3	12.8	31.0	0.4	8.8	-22.5	-16.1	-9.7	0.9	4.64	178	
81.161057	43.834976	Asia	China	25/10/2019	38.8	13.2	31.0	0.4	10.5	-20.4	-13.8	-7.2	1.2	1.69	185	
81.248335	43.966008	Asia	China	25/10/2019	38.8	13.2	31.0	0.4	10.5	-20.4	-13.8	-7.2	1.2	1.69	185	
81.074165	43.836019	Asia	China	25/10/2019	38.8	13.2	31.0	0.4	10.5	-20.4	-13.8	-7.2	1.2	1.69	185	
81.547132	43.688309	Asia	China	25/10/2019	45.3	12.8	31.0	0.4	8.8	-22.5	-16.1	-9.7	0.9	4.64	178	
81.239566	43.591386	Asia	China	25/10/2019	38.8	13.2	31.0	0.4	10.5	-20.4	-13.8	-7.2	1.2	1.69	185	

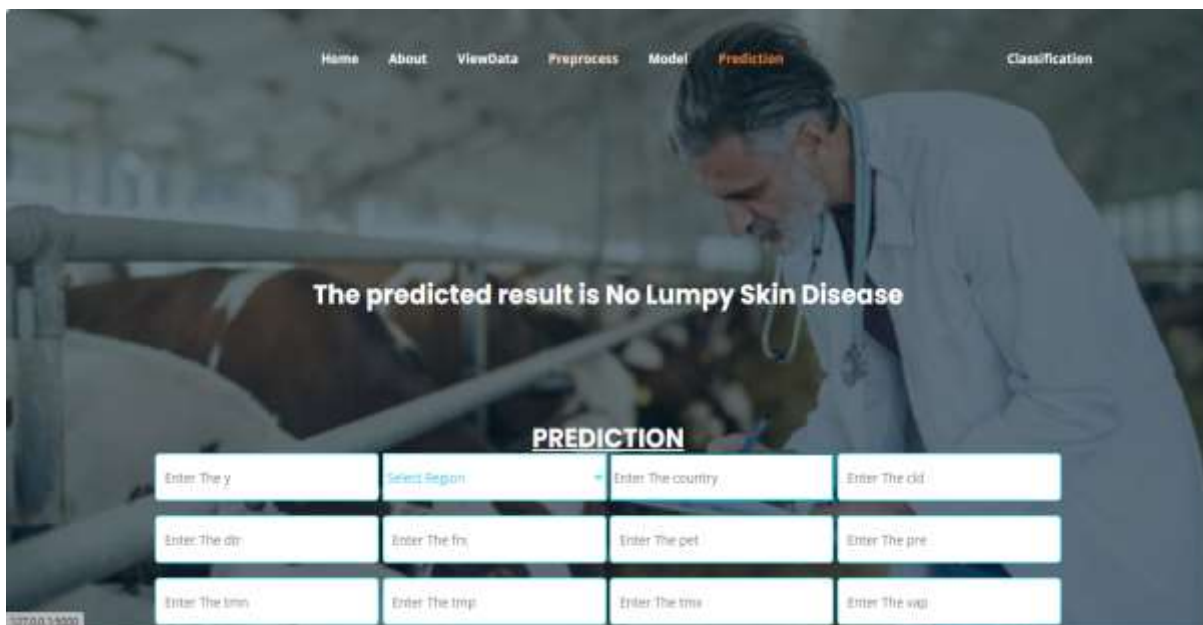
Pre-process: Here we can pre-process and split our data into train and test.



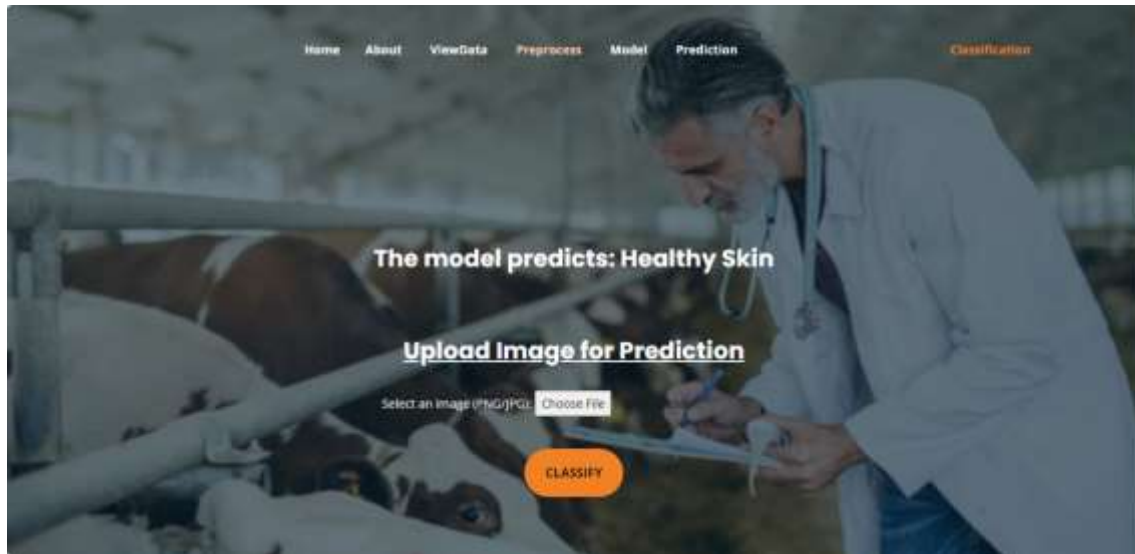
Model: Here we train our data with different ML algorithms.



Prediction: This page show the result of the user given input data.



Classification: This page show the result of the user given image.



5. SYSTEM STUDY AND TESTING

Feasibility Study

The feasibility of the project is analysed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ◆ Economical feasibility
- ◆ Technical feasibility
- ◆ Social feasibility

Economical Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified.

Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity.

The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

System Testing

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

Types of Tests

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

6. References

- [1] www.irjmets.com
- [2] www.ijwer.com