International Conference on Health and Biological Science
ICHBS, Page 378-393, December 2024

Development of an intelligent expert system with deep learning for early risk detection of monkeypox

Berliana Rahmadhani[1*], *Adam* Fathurrohman[1], *Rian* Ardianto[1], *Anggit* Wirasto[1], and *Khoirun* Nisa[1]

[1]Faculty of Science and Technology, Harapan Bangsa University, Purwokerto, Indonesia

Abstract. This study aims to develop an intelligent expert system based on deep learning to detect the risk of Monkeypox disease early. Monkeypox is a zoonotic disease that can spread to humans, with symptoms similar to smallpox but milder. Early detection is essential to prevent spread and speed up handling. In this system, deep learning models are used to analyze data on clinical symptoms and patient risk factors, including travel history, interactions with animals, and physical symptoms that appear. The expert system provides a prediction of the level of risk and recommendations for medical treatment based on the results of the analysis. The system was tested using a dataset of 770 images divided into four classes (Chickenpox, Measles, Monkeypox, and Normal), with 70% for training, 20% for testing, and 10% for validation. The evaluation results showed an accuracy of 96% in detecting Monkeypox risk, making it an effective tool for medical personnel in diagnosing and providing early treatment. The implementation of this expert system is expected to contribute to preventing the spread of Monkeypox disease more efficiently and in a timely manner.

1 Introduction

Monkeypox is an emerging zoonotic disease caused by the Monkeypox virus (a member of the genus Orthopoxvirus in the family Poxviridae) (1). This disease can be mild with symptoms lasting about 2-4 weeks, but can progress to severe to death (Case Fatality Rate 3-6%) (2). Transmission to humans occurs through direct contact with infected people or animals, or through objects contaminated by the virus. It can be transmitted through droplets, fluid that comes out of skin lesions/rashes, from the mother's placenta to the child, and through sexual intercourse (3). Monkeypox has symptoms such as fever, severe headache, swollen lymph, sore throat, nasal congestion, cough, and the characteristic of this disease is the appearance of rashes/lesions on the skin (4).

Monkeypox was designated as a Public Health Emergency of International Concern (PHEIC) by the World Health Organization (WHO) on July 23, 2022 and the PHEIC status was revoked on May 11, 2023 (5). Despite this, cases continue to be reported by various

* Corresponding author: berlianardn05@gmail.com

countries. The cumulative number of cases from January 1, 2022 to September 26, 2023 is 90,618 cases with 157 deaths reported from 115 countries. The two regions that reported the most cases in September were the Western Pacific (51.9%) and Southeast Asia (18.1%) (6).

Based on WHO as of September 26, 2023, as many as 96.3% (82,215 out of 85,336 cases observed) are men with an average age of 34 years (7). Several other key findings stated that based on case data that revealed their sexual orientation, around 83.2% (28,446 out of 34,180 cases observed) occurred in the group of men who had sex with men (MSM), as many as 7.4% of cases (2,108 out of 28,446 observed MSM) were identified as bisexual men (8). About 52.7% of cases (18,356 out of 34,832 cases who have been tested for HIV) have HIV positive status. As many as 82.5% of cases (18,056 out of 21,877 cases reported by the method of transmission) were contracted through sexual intercourse (9). Transmission from humans to animals needs to be watched out. In the 2022 outbreak, it has been reported that one pet (dog) contracted from its owner infected with Monkeypox in France (10).

Indonesia reported the first case of Monkeypox on August 20, 2022. On October 13, 2023, Indonesia again reported 1 case of Monkeypox without any travel history from the infected country (local transmission). Until the 51st week, Indonesia reported 71 cases with the number of recovered cases as many as 56 cases. The cases are spread across DKI Jakarta, West Java, Banten, East Java, Riau Islands and Yogyakarta (11)(12).

The investigation is currently still ongoing to find out the epidemiological picture of the reported cases. On October 17, 2023, Indonesia has conducted a risk assessment of Monkeypox involving multisectors. Through the risk assessment, it was found that the possibility and impact of transmission on the general public was small to moderate, while in the group based on key findings it was high (13). Considering this, we need to increase awareness of Monkeypox in Indonesia. Port Health Office The Port Health Office (KKP) as a point of entry at the entrance to the country is required to be able to carry out disease prevention in accordance with its duties and functions. Includes early alertness, prevention, control and response (14).

In this context, the implementation uses the Convolutional Neural Network (CNN) (15) (16) method to develop an artificial intelligence-based system that can identify Monkeypox symptoms based on the image of infected skin. CNN is able to automatically extract features from skin rash images and their classification with high accuracy. Through this implementation, it is hoped that technology-based solutions can help monitor, diagnose, and mitigate the spread of diseases. This study will explore the potential of CNNs to improve the accuracy and speed of diagnosis, as well as compare them with conventional methods. It is hoped that this system will be able to identify diseases more quickly and accurately, as well as provide recommendations for further actions to prevent the spread of future outbreaks.

2 Metodology

2.1 Research Method

This research involves several stages, namely problem identification by focusing on the main problem in identifying Monkeypox disease (monkeypox or mpox), data collection by collecting information or labeling classes related to the symptoms and characteristics of monkeypox disease from a medical expert, then developing a *CNN* Model using the collected data to train the *Convolutional Neural Network (CNN)* model (17) In identifying monkeypox disease, the system was developed using *the Extreme Programming* method to build an Android-based system by applying *the CNN* model (18), so that it can be used by health workers for early detection of monkeypox disease, then the implementation of the results of application development, and testing the system built in real conditions to ensure its functionality. The last stage is conclusions and suggestions to evaluate the results of the research and provide recommendations for further development (19).

Figure 1 shows the research flow used in this study. In this stage, the researcher uses the Extreme *Programming (XP) method* stage for software development. The *Extreme programming* method (20) consists of several stages, the first stage is the planning stage, the step taken is to analyze the needs plan needed for system development (21). Furthermore, in the second stage, namely design, several visualizations are made using flowcharts, use cases, activity diagrams, and user interfaces so that the research flow that is being implemented can be explained. Then, in the coding phase, after - the *tensorflow* model has been created, the model is converted to a. *TFLITE*, this will later be used in the creation of Android applications. The last stage is the submission stage, the testing stage is carried out to ensure that the features on the software run properly and accurately.

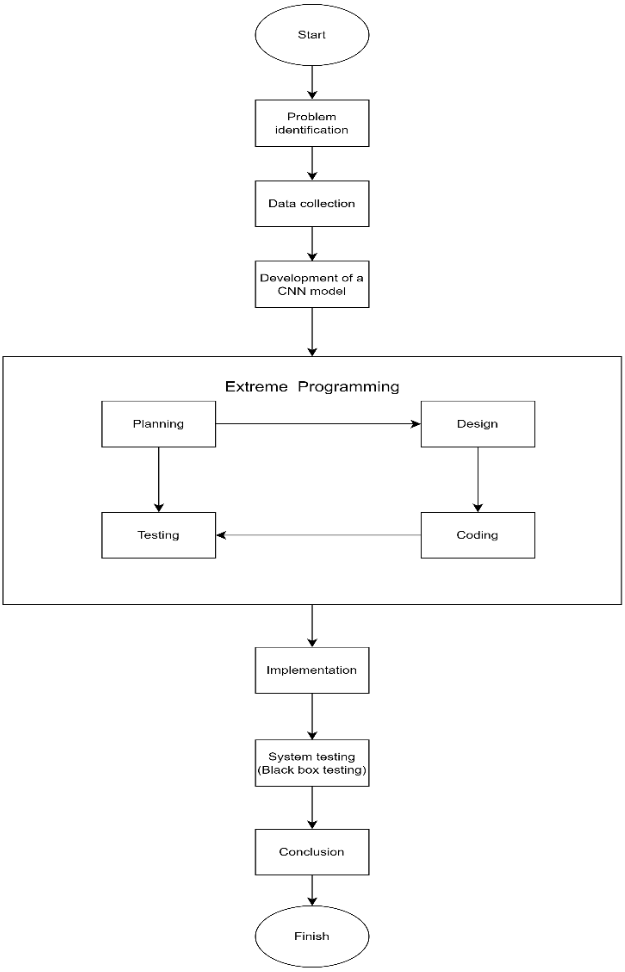

Figure 1. Research Flow

2.2 CNN Model Development

In this phase, the Convolutional Neural Network will be developed in order to apply Deep Learning in identifying monkeypox diseases according to the design that has been made previously, this model is developed with the Convolutional Neural Network (22) algorithm using the Python programming language, as well as using the TensorFlow Lite library to build detection models, as well as to run programs using notebooks from Google Collab.

a. Dataset

The dataset used by the researcher is mustard greens and consists of 4 classes, namely chickenpox, measles, monkeypox, and normal. The dataset that has been obtained is then collected by means of images taken from the internet. After that, the dataset is placed into Google Drive which has a function as a storage place in Google Collab.

b. Dataset Labeling

In this phase, the datasets that have been collected will be labeled according to the class, namely Chickenpox, Measles, Monkeypox, and Normal.

c. Split Data

In this phase, the distribution of datasets that have been collected in 4 classes is carried out with the division of 70% training data, 20% test data and 10% validation data.

d. Processing Data

Furthermore, in this step, the researcher applied data processing by changing the dimensions, input images are resized to a fixed dimension, typically 299x299 pixels, to ensure uniformity across the network. This data processing utilizes the image data generator available in TensorFlow Lite (23).

e. Building the CNN Model

At this stage, the researcher designed a model by applying the *Convolutional Neural Network* (CNN) method (24) to be able to run the monkeypox disease expert system application, in addition, the researcher also applied the InceptionV3 model architecture (25).

f. Training Model

At this stage, data training was carried out with a number of epochs of 20 and tests were carried out on the InceptionV3 model for the system that had been developed.

g. Model Testing

In this step, the researcher applies tests to the model that has been developed in compiling an evaluation script to verify the accuracy of the prediction. The data used in testing is certainly different from training and validation data.

h. Saving Model

In this step, the researcher will store the model so that it can later be applied to the application. The result of this process will result in a file in .h5 format. Next, the training data will be converted to TensorFlow Lite to make it easier to model apps on Android. Once the conversion is complete, your app will be modeled using Android Studio.

2.3 System Development

In the system development stage, the researcher designs and builds an application that will later be applied in solving the problems that have been described Background, namely the Expert System for Diagnosing Mustard Plant Diseases Using the CNN method The following are the stages of the *Extreme Programming method*.

a. Planning

This phase includes determining the business context of the application, definition of output, available features, developed application functions, determination of time to cost, and application performance flow.

b. Design

In this design phase, a flowchart visualization is created that aims to explain the visualization of the application performance process that is to be created. This can be seen in the following figure 2:

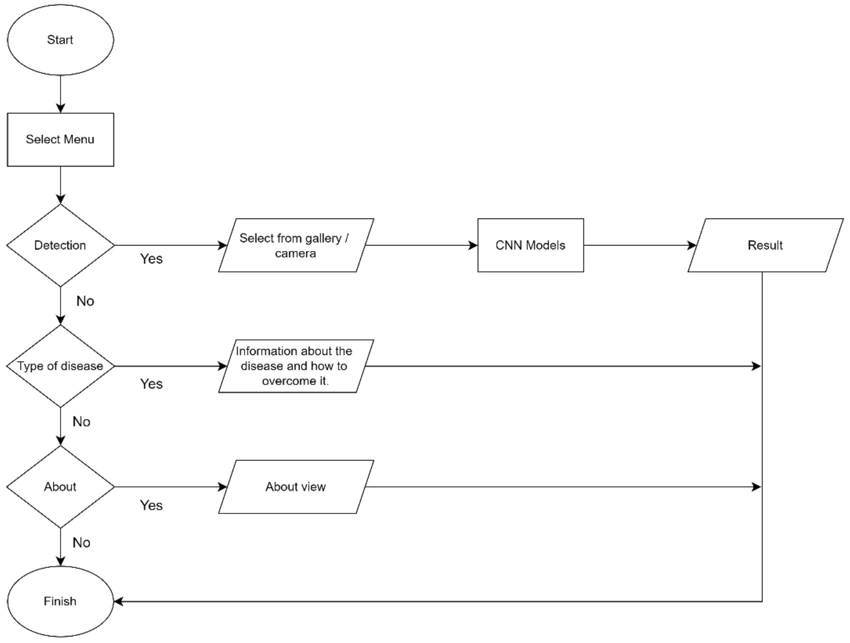

Figure 2. System Flowchart

In Figure 2 above there are 3 menus, namely detection, type of disease and about. On the detection page, users can carry out the detection process using a camera or from the gallery and will get the detection results, then on the type of disease page, users can get information about the disease and how to handle it, and an about page to see information about the application.

c. Coding

The User-Interface (UI) and User-Experience (UX) design focus on providing clear and easy-to-understand information about monkeypox, its symptoms, prevention, and treatment measures. The app should be easy to use, especially for users who may not have a medical background.

d. Testing

The testing stage is carried out through *blackbox* testing, which means that it is checked in all functional features of the *software*. In the initial test, it is seen whether the application has met the desired requirements. Then, the tester checks whether an input is valid or not to ensure that the process is running properly and correctly. After *the testing* is complete, the tester provides all the bugs encountered for developers to fix.

2.4 Convolutional Neural Network

Convolutional Neural Networks (CNNs) are deep learning architectures designed to process grid-like data, such as images (26). They work by automatically learning spatial hierarchies of features through filters applied across multiple layers. Convolutional layers extract features by applying filters that scan the image, identifying patterns like edges and textures (27). Pooling layers then reduce the dimensionality of these feature maps, preserving important information while decreasing computational complexity and making the model more robust to slight variations (28). Finally, fully connected layers take the extracted features and make predictions or classifications. CNNs are highly effective for image recognition tasks because they excel at learning from raw data by capturing spatial relationships, similar to how humans interpret visual information.

2.5 Android

Android operating is mainly designed for touchscreen touch screens such as mobile phones, tablets, and smartphones. The Android OS is based on the Linux kernel and other open-source software. We know that Android is open source, so it is the fastest growing operating system for mobile phones.

3 Result and Discussion

3.1 Data Collection

In this study, the researcher used a Monkeypox dataset consisting of various diseases, then divided into 4 classes, namely Chickenpox, Measles, Monkeypox and Normal. This dataset has been collected during the researcher's observation

Table 1. Disease dataset

No	*Label*	*Train*	*Validation*	*Test*
1.	Chickenpox	75	11	21
2.	*Measles*	64	9	18
3.	Monkeypox	195	28	56
4.	Normal	205	29	59
	Sum	539	77	154

From Table 1, it can be seen that the dataset is divided into 4 classes which are divided into 3 folders, namely train, validation, and test. The train folder is divided into 4 classes with a total of 539 datasets. In the validation folder with 4 classes totaling 77 datasets. And in the test folder with 4 classes totaling 154 datasets. The total datasets used in this study are 770 datasets.

3.2 CNN Model Development

a. Dataset

The dataset used by the researcher is Monkeypox which is divided into 4 classes, namely Chickenpox, Measles, Monkeypox and Normal. The dataset that has been obtained is then collected by taking pictures directly at the case study location. An example of a dataset can be seen in Figure 3.

Figure 3. Dataset Visualization

Based on Figure 3 above is an example of a dataset of Monkeypox Disease, namely Chickenpox, Measles, Monkeypox, and Normal.

b. Dataset application

In this step, the data that has been collected will be labeled according to the class, namely Chickenpox, Measles, Monkeypox, and Normal.

c. Split data

In this step, the researcher divided the collected dataset into 3 classes with the division of 70% training data, 20% test data and 10% validation data

d. Procesing data

In this step, Inception V3, a popular deep learning architecture within Convolutional Neural Networks (CNNs), employs a sophisticated data processing pipeline that includes careful data standardization. During the preprocessing phase, input images are resized to a fixed dimension, typically 299x299 pixels, to ensure uniformity across the network. Following this, pixel values are scaled and normalized, usually to a range of -1 to 1 or 0 to 1, which helps to stabilize the training process by making the optimization landscape smoother and ensuring that gradients do not explode or vanish. This standardization of pixel intensity allows the network to converge more efficiently, as the different layers of the architecture can focus on learning meaningful features rather than being influenced by variations in raw pixel values. Inception V3's unique multi-scale feature extraction mechanism benefits from this preprocessing, as consistent input data enhances the model's ability to recognize patterns at varying resolutions, making it highly effective for tasks like image classification and object detection.

e. Building the CNN model

In this step, an InceptionV3 model is being implemented for image classification, with. A particular focus is given to customizing and fine-tuning the network. The InceptionV3 model is initialized with pre-trained weights, and the include_top=False argument is used to exclude the top classification layers, enabling the addition of custom layers. The input image shape is defined as (img_height, img_width, 3) to align with the model's architecture. Additional custom layers are incorporated into the network after the InceptionV3 base. Specifically, a GlobalAveragePooling2D layer reduces dimensionality, followed by two dense layers with ReLU activation to learn more complex features, and a dropout layer with a 50% rate to mitigate overfitting. The final dense layer uses softmax activation to perform multi-class classification. After customizing the layers, the initial layers of the InceptionV3 model are frozen (trainable=False), ensuring that the pre-trained weights are retained and not updated during training. The model is compiled using the stochastic gradient descent (SGD) optimizer with a learning rate of 0.0001 and momentum of 0.9. Categorical crossentropy is employed as the loss function for multi-class classification, with accuracy as the evaluation metric. Additionally, a ModelCheckpoint is implemented to save the best model based on validation performance, ensuring that only the optimal weights are retained during training. Figure 4 above illustrates the InceptionV3 architecture utilizing the CNN method. The following code snippet provides an overview of the implementation:

```
# Import InceptionV3 Model
inc_model = InceptionV3(weights='/content/skinimagedataset.h5',
                        include_top=False,
                        input_shape=(img_height, img_width, 3))
print("number of layers:", len(inc_model.layers))
#Adding custom Layers
x = inc_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(1024, activation="relu")(x)
x = Dropout(0.5)(x)
x = Dense(512, activation="relu")(x)
predictions = Dense(2, activation="softmax")(x)
# Custom layers are fed into the main model
model_ = Model(inputs=inc_model.input, outputs=predictions)
# Stops the main frozen body of the architecture to be updated through backprop
for layer in model_.layers[:52]:
    layer.trainable = False
# compile the model
model_.compile(optimizer=SGD(lr=0.0001, momentum=0.9)
                    , loss='categorical_crossentropy'
                    , metrics=['accuracy'])
#fit generator
checkpointer = ModelCheckpoint(filepath='bestmodelskinimage.hdf5',
                               verbose=1, save_best_only=True)
```

Figure 4. Model Creation Code

f. Training Model

After going through the model creation process, this graph shows the model's performance in terms of accuracy and loss for training and validation data for 20 epochs. Graphs are used to monitor how the model learns from the data during the training process. The goal is to ensure that accuracy increases and losses are reduced for both training and validation data, with no signs of overfitting. The following is a graph of the results of the training carried out, that can be seen in Future 5 :

Figure 5. Training and Validation Accuracy Graph

The graph illustrates the training and validation performance of the InceptionV3 model over 20 epochs, tracking both accuracy and loss. Both the training and validation accuracy (represented by red and purple lines, respectively) increase sharply in the early epochs, then stabilize, showing that the model is learning effectively without significant overfitting. The loss curves (blue for training and green for validation) decrease quickly at first, with the validation loss stabilizing around epoch 4 and the training loss continuing to decline gradually. Overall, the model achieves a well-balanced convergence, with minimal overfitting, as evidenced by the close alignment between the validation and training curves. Over the entire training period, the model reaches its best performance with an average training loss of 0.2341, a validation loss of 0.2223, a training accuracy of 0.9037, and a validation accuracy of 0.9035. These values indicate the model's strong generalization to unseen data, suggesting effective learning of the InceptionV3 architecture.

g. Model Testing

In this step, to measure the level of accuracy in the InceptionV3 architecture and test on the prediction label, it is necessary to test the model. In deep learning, particularly when evaluating the performance of an Inception V3 architecture in a CNN, the confusion matrix is a critical tool for measuring classification accuracy. It provides a comprehensive summary of how well the model is performing by breaking down predictions into four categories: true positives (correctly predicted positive cases), true negatives (correctly predicted negative cases), false positives (incorrectly predicted positive cases), and false negatives (incorrectly predicted negative cases). This matrix allows for a deeper insight into the model's performance beyond simple accuracy, highlighting specific areas where the model may struggle, such as misclassifying certain classes more often than others. By examining these misclassifications, one can detect patterns of bias or imbalance in the predictions, offering guidance for further model tuning or data preprocessing. The confusion matrix is especially valuable for evaluating multi-class classification problems, where accuracy alone can be misleading, as it reveals class-specific performance, providing a clearer picture of how well the Inception V3 model generalizes to unseen data.

The following is the result of the confusion matrix in figure 7. Based on Figure 7, the prediction uses the InceptionV3 architecture based on using a confusion matrix. From the Chickenpox class test, Precision can be achieved at 95%, Recall 100% and f1-score 97%. Then from the Measles class, the test obtained Precision achieved 89%, Recall 100% and F1-score 94%. Then from the Monkeypox class, the test obtained Precision achieved 95%, Recall 100% and F1-score 97%. And from the Normal class test, Precision got a result of 100%, Recall 100% and f1-score 100%. Thus, the use of the InceptionV3 architecture can result in an accuracy rate of 96%.

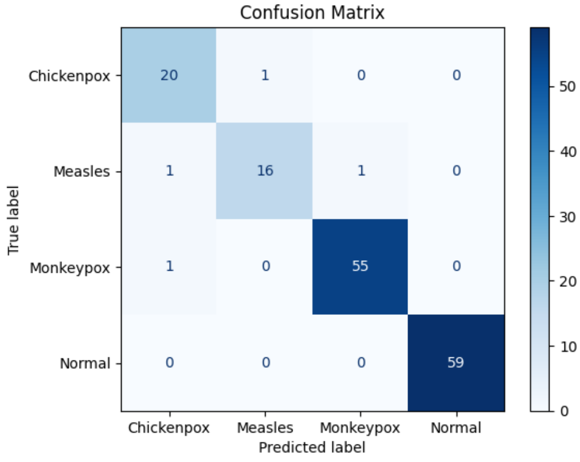

Figure 7. Confusion Matrix

h. Saving Model

In this step, to apply the model to the application, it is necessary to do a saving model. The results of the processing can later produce a file in .h5 format. First, the model is trained, and throughout the training process, checkpoints are created to save the model's weights whenever it achieves better performance, such as higher validation accuracy or lower loss. The H5 format, which is based on the HDF5 file system, allows the model to be saved efficiently, including both the architecture and the learned weights, making it easy to reload for future inference or fine-tuning. TensorFlow Lite then comes into play when converting the saved model for deployment on mobile or edge devices. The conversion to TensorFlow Lite format optimizes the model by reducing its size and improving its inference speed, while still maintaining its accuracy. This process includes quantization techniques that reduce the model's numerical precision, allowing it to run faster and use less memory, which is crucial for resource-constrained environments. After conversion, the model is ready to be deployed in a compact and efficient form, preserving the accuracy and performance achieved during training. Furthermore, the training data will be converted into TensorFlow Lite to make it easier to create applications on Android. After the conversion is complete, an application will be created using Android Studio software.

3.3 CNN Model Development

The following is a view of the implementation of the application that has been built, can be seen in Table 2.

Table 2. Application implementation

Page View	Information
 **Figure 8.** Splash Pages	Splash pages are used for loading pages of applications. Users will wait for 4 seconds on this page before entering the main page of the application.

 **Figure 9.** Home	This main page is used to select features, there are 4 features on this page, namely diagnostic, disease, medical history and about.
 **Figure 10.** Detection page	This page is used to detect Monkeypox disease, with 2 buttons on the menu, namely take a picture, which can take a picture directly and a gallery for taking a picture in the gallery. There are also solutions based on detected diseases.

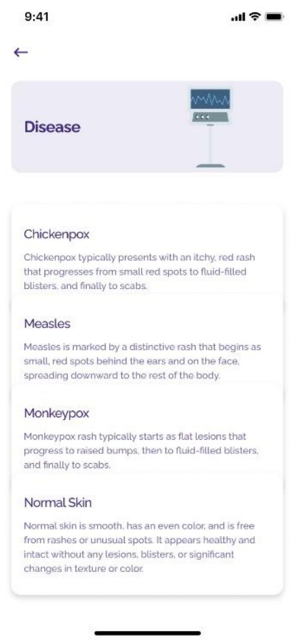

Figure 11. Disease type page

This page is used to view information on Monkeypox diseases and how to deal with them.

Figure 12. Medical History

This page has a function in viewing the medical record information of application users.

Figure 13. About Page

This page has a function to view information on how to use the application.

4 Conclusion

From the research that has been conducted, this article discusses the development of a deep learning-based intelligent expert system for early detection of monkeypox disease risk. By utilizing deep learning technology, this system is designed to accurately and efficiently identify early signs of monkeypox disease. This approach leverages extensive medical data and machine learning techniques to analyze patterns and symptoms associated with the disease.

The results of the study show that this expert system has significant potential in improving the early detection of monkeypox, which in turn can accelerate the handling and control of the spread of the disease. With its high accuracy and ability to process data in real-time, these systems can be a valuable tool for healthcare professionals in diagnosing and responding to monkeypox risks more quickly than traditional methods.

However, the article also highlights some challenges, such as the need for high-quality data and potential bias in the model that must be addressed to ensure the accuracy and success of the system. Further development and evaluation of the system in various clinical contexts will be important to optimize its effectiveness and applicability in the field. Overall, the development of intelligent expert systems with deep learning offers innovative solutions for the early detection of monkeypox and can contribute significantly to global public health.

References

1. Ahsan MM, Ali MS, Hassan MM, Abdullah TA, Gupta KD, Bagci U, et al. Monkeypox Diagnosis with Interpretable Deep Learning. IEEE Access. 2023;11(August):81965–80.
2. Yasmin F, Hassan MM, Hasan M, Zaman S, Kaushal C, El-Shafai W, et al. PoxNet22: A Fine-Tuned Model for the Classification of Monkeypox Disease Using Transfer Learning. IEEE Access. 2023;11(March):24053–76.
3. Raha AD, Gain M, Debnath R, Adhikary A, Qiao Y, Hassan MM, et al. Attention to Monkeypox: An Interpretable Monkeypox Detection Technique Using Attention Mechanism. IEEE Access. 2024;12(March):51942–65.
4. Olusegun R, Oladunni T, Audu H, Houkpati YAO, Bengesi S. Text Mining and Emotion Classification on Monkeypox Twitter Dataset: A Deep Learning-Natural Language Processing (NLP) Approach. IEEE Access. 2023;11(March):49882–94.
5. Kundu D, Rahman MM, Rahman A, Das D, Siddiqi UR, Alam MGR, et al. Federated Deep Learning for Monkeypox Disease Detection on GAN-Augmented Dataset. IEEE Access. 2024;12(March):32819–29.
6. Organization WH. Mpox Outbreak Toolbox (who.int). 2024 [cited 2024 Aug 28]. p. 1 There are two distinct clades of the virus: clade I (with subclades Ia and Ib) and clade II (with subclades IIa and IIb). In 2022–2023 a global outbreak of mpox was caused by the clade IIb strain. Available from: https://www.who.int/news-room/fact-sheets/detail/mpox
7. Smith A. BBC. 2022. p. 1 WHO declares mpox global health emergency. Available from: https://www.bbc.com/news/articles/cvg35w27gzno
8. Cynthia S. Goldsmith RR. United Nations News. 2024 [cited 2024 Aug 14]. p. 1 WHO declares mpox virus a public health emergency of international concern. Available from: https://news.un.org/en/story/2024/08/1153176
9. Bengesi S, Oladunni T, Olusegun R, Audu H. A Machine Learning-Sentiment Analysis on Monkeypox Outbreak: An Extensive Dataset to Show the Polarity of Public Opinion From Twitter Tweets. IEEE Access. 2023;11(February):11811–26.
10. WIRES N. France 24. 2024 [cited 2024 Aug 14]. p. 1 WHO declares monkeypox a global public health emergency. Available from: https://www.france24.com/en/africa/20240814-who-declares-monkeypox-a-global-public-health-emergency
11. M R. CNBC Indonesia. 2024 [cited 2024 Aug 24]. p. 1 Lengkap! Semua Tentang Mpox, Indonesia Wajib Waspada. Available from: https://www.cnbcindonesia.com/research/20240824140157-128-566083/lengkap-semua-tentang-mpox-indonesia-wajib-waspada
12. RI UPKK. Kementerian Kesehatan RI. 2022. p. 1 Kasus Monkeypox Mulai Terdeteksi di Indonesia. Available from: https://upk.kemkes.go.id/new/kasus-monkeypox-mulai-terdeteksi-di-indonesia
13. Kesehatan D. Dinkes Semarang. 2024 [cited 2024 Aug 22]. p. 1 Waspada Mpox, 88 Kasus Terdeteksi di Indonesia. Available from: https://dinkes.semarangkota.go.id/content/post/407
14. Putu Indah Savitri, Didik Kusbiantoro EIS. Komunikasi dan Media KEMENSETNEG RI. 2024 [cited 2024 Aug 30]. p. 1 Jelang HLF MSP dan IAF 2024 di Bali, DPR RI Fokus Pencegahan Wabah Monkeypox. Available from: https://www.setneg.go.id/baca/index/jelang_hlf_msp_dan_iaf_2024_di_bali_dpr_ri_fokus_pencegahan_wabah_monkeypox

15. Alhamdani AA. Application of Deep Learning using Convolutional Neural Network (CNN) Algorithm for Gesture Recognition. Journal of Software Engineering, Information and Communication Technology (SEICT). 2022 Feb 24;4(1):61–8.
16. Kuncoro DF, Ardianto R. Analisis Sel Darah Putih dengan Pendekatan Bioinformatika menggunakan Arsitektur. 2024;5(1):31–7.
17. Ardianto R, Kartika Wibisono S. Analisis Deep Learning Metode Convolutional Neural Network Dalam Klasifikasi Varietas Gandum. Jurnal Kolaboratif Sains. 2023;6(12).
18. Putri A, Ardianto R. Technologies and Securities Issues Kemajuan dalam Sistem Layanan Kesehatan yang mendukung IoT : Tinjauan Sistematis terhadap Masalah Teknologi dan Isu Keamanan. 2024;1(2):89–101.
19. Farahat FA, ElSayed MA. Achievement Stability Set for Parametric Rough Linear Goal Programing Problem. Fuzzy Information and Engineering. 2019;11(3):279–94.
20. Chen H, Wang J, Zhu J, Xiong X, Wang W, Yang H. A Two-stage Stochastic Mixed-integer Programming Model for Resilience Enhancement of Active Distribution Networks. Journal of Modern Power Systems and Clean Energy. 2023;11(1):94–106.
21. Hussain T, Siniscalchi SM, Wang HLS, Tsao Y, Salerno VM, Liao WH. Ensemble hierarchical extreme learning machine for speech dereverberation. IEEE Transactions on Cognitive and Developmental Systems. 2020;12(4):744–58.
22. Istiqomah H, Ardianto R. Prediksi Kanker Darah Menggunakan Metode Convolutional Neural Network. 2024;4(1):51–60.
23. François Chollet. Deep Learning with Python. 2017. 1–384 p.
24. Nurhalizah RS, Ardianto R. Analisis Supervised dan Unsupervised Learning pada Machine Learning : Systematic Literature Review. 2024;4(1):61–72.
25. Rifa A, Sujiwanto I, Ronggo Bintang Pratomo Prawirodirjo R, Palupingsih P. Analisis Perbandingan Performa Model Klasifikasi Kesehatan Daun Tomat menggunakan Arsitektur VGG, MobileNet, dan Inception V3 Analysis Tomato Leaf Health Classification Model Performance Comparison Using VGG, MobileNet, and Inception V3. Jurnal Ilmu Komputer Agri-Informatika. 2023;10(1).
26. Zhang X. Improved Three-Dimensional Inception Networks for Hyperspectral Remote Sensing Image Classification. IEEE Access. 2023;11(March):32648–58.
27. Salami A, Andreu-Perez J, Gillmeister H. EEG-ITNet: An Explainable Inception Temporal Convolutional Network for Motor Imagery Classification. IEEE Access. 2022;10:36672–85.
28. Karthick Raghunath KM, Vinoth Kumar V, Venkatesan M, Singh KK, Mahesh TR, Singh A. XGBoost Regression Classifier (XRC) Model for Cyber Attack Detection and Classification Using Inception V4. Journal of Web Engineering. 2022;21(4):1295–322.