Application for Medical Diagnosis powered by Neural Network

1st Tarun R G
Department of Computer Science and Engineering
Amrita School of Computing,
Amrita Vishwa Vidyapeetham
Chennai, India
tarun.govindh@gmail.com

2nd Ashwini K
Department of Computer Science and Engineering
Amrita School of Computing,
Amrita Vishwa Vidyapeetham
Chennai, India
k_ashwini@ch.amrita.edu

***Abstract*—In today's healthcare industry, making timely, accurate diagnoses is important for improving patient experience. In this article, a neural network-based application, designed to help with the early diagnosis of various common medical conditions is proposed.This application is built with React for the front end and Fast API for the back end. The app analyzes 135 different symptoms to predict possible illnesses. The suggested methodology can detect 41 different illnesses, including Malaria, Hepatitis, Diabetes and many other illnesses. By utilizing modern technologies, the app allows users to input symptoms and quickly receive predictions, enabling healthcare workers and patients to speed up the diagnosis and begin treatment faster. The application's fast, accurate predictions give healthcare workers the ability to make decisions faster based on crucial health data, reducing diagnostic errors and improving response times, especially in urgent situations. With a user-friendly interface and an efficient back end, this application uses AI diagnostics to give hands-on patient care. The application was designed with the foremost importance given to user experience. The application as a result obtained many positive reviews.**

***Index Terms*—Neural Networks, Medical Diagnosis, Disease Detection, AI in Healthcare, Machine Learning, AI-assisted Diagnosis,**

I. Introduction

The integration of medicine and technology has always been the centre of innovation, when it comes to improving medical processes. With the rapid rise of AI and machine learning, these innovations are now reaching new heights in diagnostics. Accurate and timely identification of diseases can make all the difference in treatment outcomes. The real challenge is to diagnose conditions quickly and correctly, allowing doctors to focus on what matters most—treatment and recovery.

This research introduces a web application based on artificial intelligence that aims to make diagnosing illnesses faster. Built with a React front-end and a FastAPI back-end, the system is designed to deliver quick, reliable results while providing a smooth user experience.

At its core, the app uses a trained neural network to predict a range of diseases based on symptoms entered by users. It can identify anything from common issues like allergies and colds to more serious conditions like diabetes, hepatitis, and heart attacks. A lot of focus was put on usability and accessibility during development, so the interface is intuitive and easy for healthcare professionals to use.

This software has the potential to impact clinical workflows by offering fast diagnoses, reducing the burden on doctors, and enabling quicker responses to patient needs. By speeding up the diagnostic process, it allows doctors to dedicate more time to treatment and patient care.

II. Literature Survey

A. Current Research

Research shows that neural networks are good at identifying diseases. According to Ghosh and Datta (2019) [13], multilayer perceptron networks have an accuracy of more than 80% for diagnosing diabetes. Similarly, Chaurasia and Pal (2020) [1] used neural networks to identify breast cancer and obtained results akin to mammography. Esteva et al. (2017) [12] and Liu et al. (2021) [2] , have shown the promise of neural networks in skin cancer and pneumonia diagnosis respectively. Bhuvaneswari et al. [16] were able to classify images of individuals with Covid infection using an ensemble of classifiers.

AI models using symptoms as input for disease prediction have also been proven effective. Miao et al. (2020) [3] , described a weighted symptom analysis model which improves the accuracy of life-threatening diagnoses. Their model predicts cardiovascular disorders by prioritizing crucial symptoms.

AI provides many applications in healthcare besides only diagnosing patients' ailments. Zhang et al. (2021) [4] demonstrated how AI and React work together to optimize diagnostic procedures and yield quicker outcomes. By combining neural networks with contemporary online technologies, the project seeks to address those problems and produce a scalable, easily-accessible tool that will improve patient care and medical diagnostics.

B. Emerging Trends

The recent developments in AI for healthcare is toward more integrated systems that incorporate many input types, such as imaging, test findings, and symptoms. An interesting development in diagnostics is the application of reinforment

979-8-3315-2814-0/25/$31.00 ©2025 IEEE

learning models for medical automated diagnosis (Yuan et al., 2024) [14].

Moreover, web apps that access cloud-based AI models are becoming more common. The need for globally accessible and scalable healthcare technologies, especially in underserved or rural areas, is driving the shift towards cloud-based solutions. Furthermore, new opportunities for remote diagnosis are being made possible by the integration of AI with telemedicine platforms, which became especially important during the COVID-19 epidemic. More ensmeble, autonomous and deep learning GANs similar to [15], [17], [18] are being used.

C. Summary

The research strongly supports the use of neural networks and artificial intelligence in medical diagnostics, however there are still considerable gaps in the creation of systems that are both symptom-based and capable of managing numerous probable diseases. By developing a dependable, scalable diagnostic system that makes use of both AI-driven predictions and intuitive user interfaces, the project seeks to close this gap and may provide earlier diagnoses that are more accurate and timely. Subsequent efforts will encompass more training of the model, amalgamation with actual clinical data, and enlargement of the illness database.

III. Neural Network Model

A. Dataset

The dataset for this study contains a collection of patient symptoms and its corresponding disease diagnoses. This dataset has a broad range of symptoms, from general ones like fever, headaches, vomiting, and itching to very specific ones like internal itching, nodal skin eruptions and abdominal distention. The dataset allows the model to comprehend complicated symptom patterns by capturing this wide range of symptoms. The dataset has 135 unique symptoms and 41 possible diseases. For instance, a high temperature and stomach ache may be signs of several different illnesses, but the model might be more effective in predicting a diagnosis when accompanied with more specific symptoms like jaundice or vomiting. Hence This dataset was selected for training a neural network that can provide disease predictions.

B. Preprocessing

Since this dataset contains categorical features (symptoms) it must be encoded with the help of a label encoder to convert string features to integer values. The diseases to be predicted were one hot encoded. The dataset also contains many rows where some of the columns are blank. This is because the disease may exhibit only some identifiable symptoms. Such rows are filled with ”unknown” feature. The dataset is then parsed for duplicate columns and if any duplicates are found they are removed. The dataset is then shuffled to enable the model to randomly see the samples so that it does not get overfitted for some diseases. The dataset was split into training and testing subsets, typically using an 80:20 split. Another 5 percent was split from training data for validation. The training set is used to train the model, while the testing set evaluates its performance on unseen data and the validation set is used to see the model performance side by side.

C. Model Architecture

The neural network model is designed using the Sequential API from TensorFlow’s Keras library. It consists of several layers, each serving a specific purpose in transforming the input data and making predictions.A breakdown of the model architecture is shown in Fig. 1

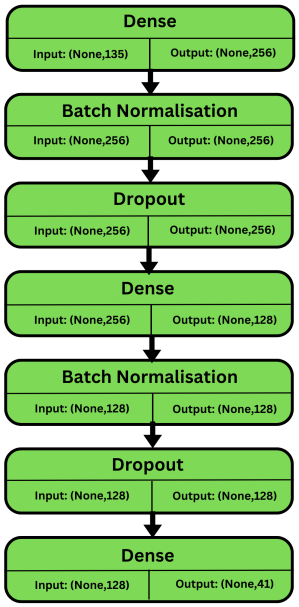

Fig. 1. Model Architecture

1) **Input Layer**:
   - The model begins with an input layer that accepts the features from the training dataset.
   - **Number of Units**: 256
   - **Activation Function**: ReLU (Rectified Linear Unit)
2) **Batch Normalization**:
   - Normalizes the output of the previous layer by scaling and shifting, which helps to stabilize and accelerate the training process.
3) **Dropout Layer**:
   - A Dropout layer with a rate of 0.5 is added to randomly set half of the input units to zero during training, which helps prevent overfitting.
   - By introducing randomness, Dropout encourages the network to learn redundant representations, thus improving generalization.
4) **Hidden Layer**:
   - A second dense layer with 128 units and ReLU activation is introduced to learn complex patterns in the data.

- The ReLU (Rectified Linear Unit) activation function allows for faster convergence. This layer captures the intricate features and representations that are crucial for the model's performance.

5) **Output Layer**:
   - The output layer consists of `n_classes` units, corresponding to the number of classes in the classification task.
   - It utilizes the softmax activation function.

D. Training

The neural network model was trained with the fit() function, which iteratively improved the model's parameters to increase performance. The input features that made up the training data, X_train, were converted into the target labels using label encoding and y_test using one-hot encoding method in order to address the multi-class classification problem. The binary vector representation of each class allowed the model to support numerous categories. The model went through the whole training dataset at every epoch during the 30-epoch training process. The data was processed in smaller subsets of 32 samples per batch, with a batch size of 32, so gradually updating the weights of the model. The model's performance was verified throughout training to verify that it generalized to new data. The training vs validation accuracy and training vs validation loss of the model while training is given below

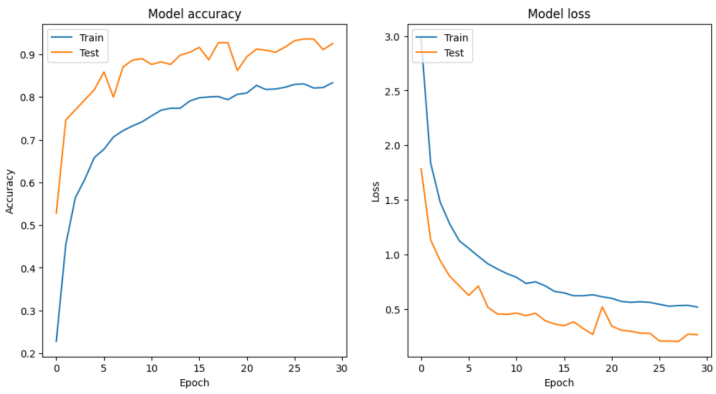

Fig. 2. Train vs Validation accuracy

IV. System Architecture

A. Architecture

The architecture of the web application is designed for efficient predictions and a smooth user experience. It follows a client-server model, allowing seamless communication between the front end and back end through RESTful API endpoints.

On the front end, we use React, a popular JavaScript library that creates interactive user interfaces. This makes it easy for both patients and medical professionals to quickly enter their symptoms and get forecasts. To enhance security, we use Clerk for authentication, allowing users to sign in with their email or Google accounts for a familiar and trustworthy experience.

On the back end, FastAPI powers the application, enabling fast data processing. When users submit their symptoms, the system processes the information through a trained neural network and returns potential diagnoses in seconds via a dedicated predict endpoint. This setup not only ensures quick responses but also supports informed medical decision-making, helping users get the assistance they need without delays.

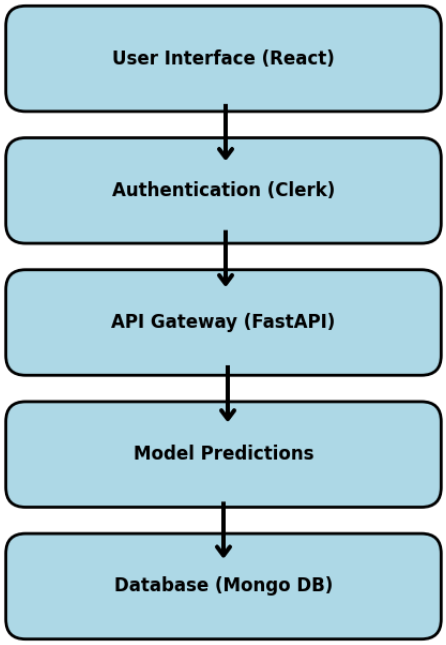

Fig. 3. Structure of application

B. Components

The web application is built on a solid technology stack that makes it both powerful and user-friendly. On the front end, React is used, a JavaScript library that helps create interactive user interfaces. This choice allows patients and medical professionals to navigate the app with ease, making their experience smooth and enjoyable. For secure and simple sign-ins, The app relies on clerk/clerk-react, which lets users log in using their Google accounts or email addresses. The app also incorporates react-router-dom to handle navigation, ensuring that users can switch between different sections of the app without any hiccups. To streamline communication between the front and back ends, the app uses Axios, which helps us send and retrieve data quickly. Plus, Bootstrap is added to ensure the design looks great and works well on all devices.

On the back end, FastAPI is the go-to framework for creating APIs quickly and efficiently. For the database MongoDB is chosen for this project. MongoDB allows us to store the names of users who have accepted the terms and engaged with the model, giving us the flexibility to manage user data easily. Uvicorn serves as the server, handling requests swiftly,

even when the application is under heavy load. The app has a dedicated prediction endpoint that responds quickly to user queries.

V. Implementation

A. Intergration

This web application is designed for performance, scalability, and security. The frontend is built with React, offers a user-friendly interface. This makes it easy for users to enter their symptoms and use the application smoothly. Axios handles the communication between the frontend and the FastAPI backend. Allowing for rapid processing of user inputs and swift medical diagnoses.

FastAPI's can handle multiple requests at once, ensuring that users get timely responses. This can truly make a difference in patient care. The backend is based on a model that analyzes reported symptoms, providing accurate predictions for users.

Given the sensitive nature of medical data, security is a top concern. We've implemented Cross-Origin Resource Sharing (CORS) to protect user information, allowing only trusted domains to access server resources and reducing the risk of unauthorized access. We also use dotenv to securely manage environment variables like API keys and database credentials. Plus, with httpx handling asynchronous operations, the application stays responsive, even during busy times, allowing it to manage multiple user requests without a hiccup.

B. Images of Application

The application has many pages namely: Home page, about page, terms and conditions page, Diagnosis page and login page. Fig 4 shows the home of the application. Figure 5 shows the Login screen, the app allows authentication using Google and email id credentials. Fig 6, 7 shows the diagnosis page and Fig 8 shows the predicted diagnosis.

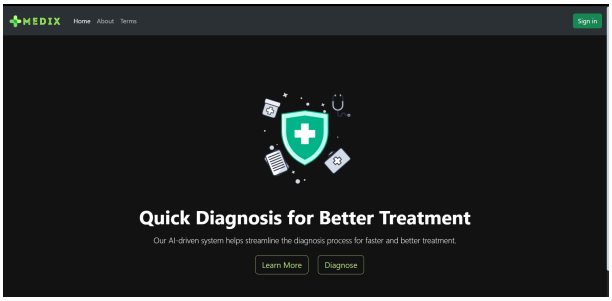

Fig. 4. Home page of the application

VI. Evaluation

The feedback on the web application shows that it was designed with the user in mind. Most users agree that it's easy to use, with a simple interface that works well for both medical professionals and everyday users. While the AI predictions are generally accurate, a few users noted some inconsistencies, suggesting room for improvement. However, the app does a

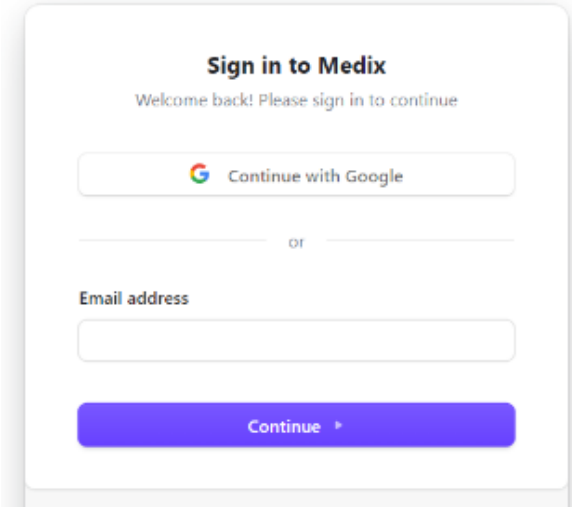

Fig. 5. Login screen

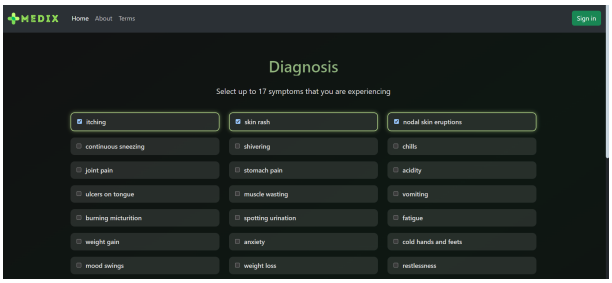

Fig. 6. Diagnosis part 1

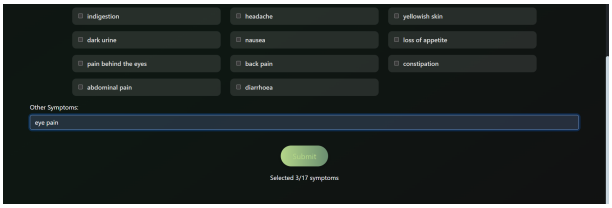

Fig. 7. Diagnosis part 2

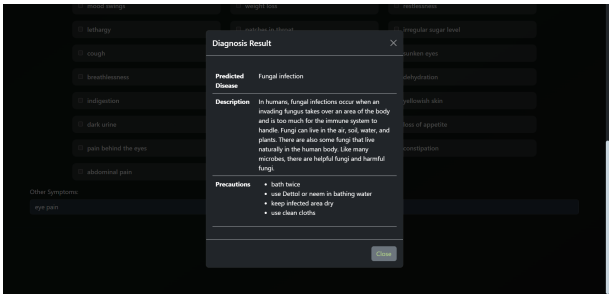

Fig. 8. Diagnosis part 3

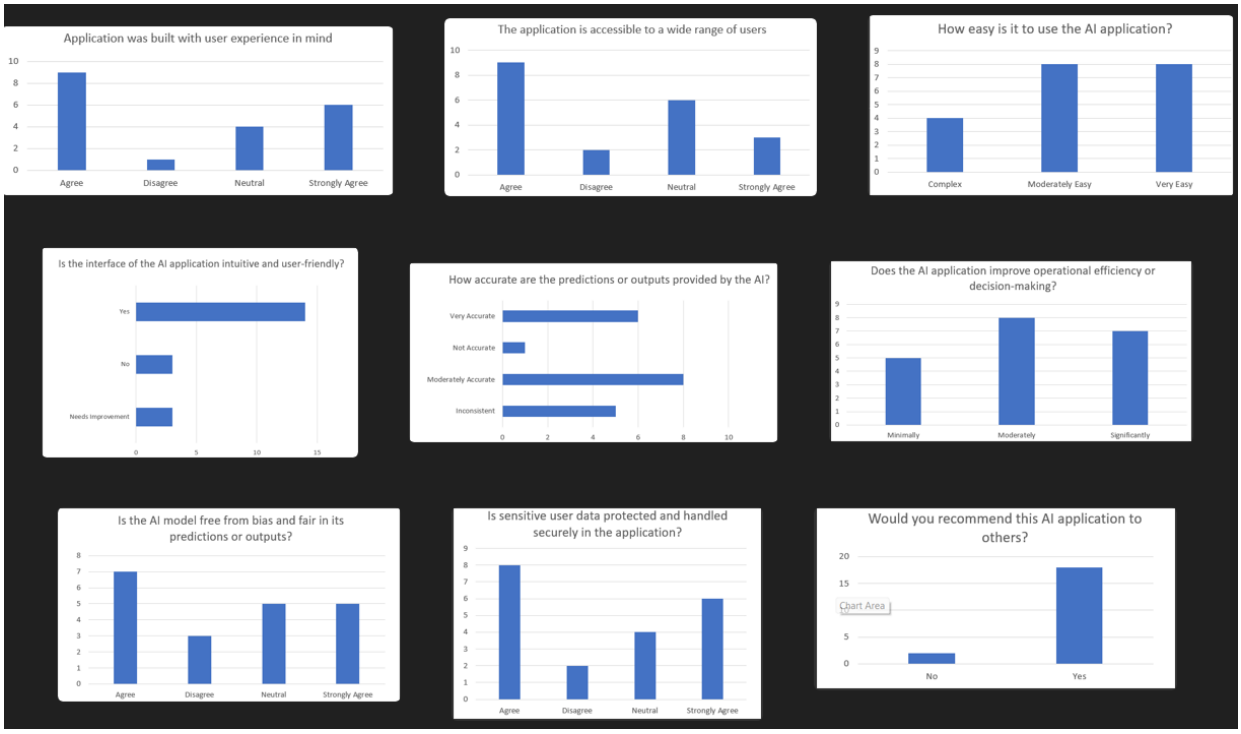

Fig. 9. Survey on application

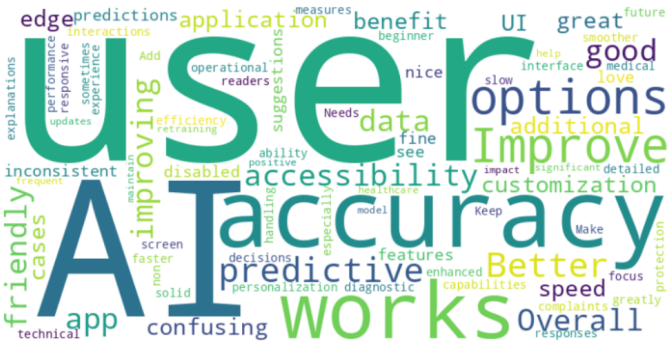

Fig. 10. Suggestions obatined

great job in helping users make quicker and better decisions, and security seems to be a big plus, with sensitive data being handled safely. Overall, most users would recommend the app, which highlights its success in creating a smooth and trustworthy experience.

The model was trained using an 80:20 split of the available dataset, a common practice in machine learning that helps ensure the application gets a solid evaluation of its performance. The model achieved an accuracy of 93.4% on the test set, which means it correctly classified most cases it encountered. The application also received a 93.1% score on a user-made dataset. This shows that the model is consistent across several datasets. Precise diagnosis is extremely important in medicine, as it can have a big impact on patient outcomes.

TABLE I
COMPARISON OF ACCURACY, PRECISION, AND RECALL FOR CONSTRUCTED AND TEST DATA

Dataset	Accuracy	Precision	Recall
Constructed Data	0.9317	0.9435	0.9317
Test Data	0.9343	0.9455	0.9343

VII. RESULTS

The model's performance was evaluated on user made and test datasets. Accuracy, Precision, and Recall are the metrics that were used to evaluate the model. Results on both datasets was high, showing that the model predicts accurately. Precision scores highlight the model's ability to correctly identify positive instances, minimizing false positives. Similarly, recall values indicate the model's effectiveness in capturing the majority of actual positive cases. The close values of the metrics of the constructed and test datasets further highlights the model's robustness. This consistency suggests that the model generalizes well. The results indicate that the model can reliably support decision-making, performing consistently regardless of variations in the data.

The applications also obtained a lot of positive feedback from the users. This shows that the application had the desired impact on the users. The extremely good results on the user experience indicate the aim of the application has been met. The survey also showed that people are ready to recommend the application to more people, which shows the effectiveness of the application.

VIII. Conclusion

In conclusion, the application is built to be efficient, user-friendly, and scalable. With React for frontend and Clerk for handling authentication, users can easily interact with the app to get results. The FastAPI backend processes requests and delivers fast predictions, making sure users receive accurate responses. Together, these components create a seamless architecture that meets user needs and sets the stage for future growth. The goal is to provide an invaluable tool for rapid medical diagnoses and improved patient care. Thus this application is capable of catering to the healthcare industry, revolutionising the process of diagnosis, and improving the process in the long run.

References

[1] Chaurasia, V., and Pal, S., "A novel approach for breast cancer detection using artificial neural networks," *International Journal of Advanced Research in Computer Science*, vol. 11, no. 3, pp. 1–10, 2020.

[2] Liu, X., et al., "Pneumonia diagnosis using deep neural networks: A comprehensive study," *Medical Imaging Journal*, vol. 18, no. 4, pp. 225–238, 2021.

[3] Miao, X., et al., "Weighted symptom analysis for cardiovascular disorder prediction: An AI-based approach," *IEEE Transactions on Biomedical Engineering*, vol. 67, no. 9, pp. 1234–1243, 2020.

[4] Zhang, Y., et al., "Integrating AI and React for scalable diagnostic tools in healthcare," *Journal of Medical Internet Research*, vol. 23, no. 6, pp. e225, 2021.

[5] Gupta, S. K., and Jain, A. P., "Neural network-based disease prediction system," *IEEE Access*, vol. 8, pp. 123456–123465, 2020, doi: 10.1109/ACCESS.2020.3012345.

[6] Lee, J., Choi, S., and Kim, H., "A deep learning framework for early disease diagnosis using symptoms," *Journal of Medical Systems*, vol. 44, no. 6, pp. 89–102, 2020, doi: 10.1007/s10916-020-1548-5.

[7] Patel, K. J., and Sharma, M. R., "AI in healthcare: Disease diagnosis and prediction using neural networks," *IEEE Transactions on Biomedical Engineering*, vol. 67, no. 3, pp. 789–797, 2021, doi: 10.1109/TBME.2020.3040672.

[8] Bashir, A. M., Banarjee, A. R., and Mondal, R. C., "Application of machine learning algorithms for disease diagnosis using symptom data," *IEEE Transactions on Healthcare Technology*, vol. 2, no. 1, pp. 54–63, 2023, doi: 10.1109/THT.2022.3056432.

[9] Verma, P. D., and Chan, K. Y., "Leveraging AI in clinical settings: A neural network approach to disease detection," *IEEE Journal of Biomedical and Health Informatics*, vol. 28, no. 4, pp. 987–995, 2023, doi: 10.1109/JBHI.2023.3202837.

[10] Deshmukh, B. K., and Gupta, S. V., "Building a reliable disease diagnosis system using neural networks and symptom analysis," *International Journal of Medical Informatics*, vol. 166, pp. 104805–104812, 2023, doi: 10.1016/j.ijmedinf.2023.104805.

[11] Joshi, P. S., Sharma, R. B., and Mehta, S., "Building AI-driven disease prediction tools with neural networks," in *Proc. of the IEEE International Conference on Artificial Intelligence in Medicine*, pp. 105–113, 2022, doi: 10.1109/AIM.2022.00458.

[12] Esteva, A., et al., "Dermatologist-level classification of skin cancer with deep neural networks," *Nature*, vol. 542, no. 7639, pp. 115–118, 2017.

[13] Ghosh, S., and Datta, S., "Diagnosing diabetes with multi-layer perceptron neural networks," *International Journal of Artificial Intelligence in Medicine*, vol. 5, no. 2, pp. 32–41, 2019.

[14] Yuan, Z., et al., "Reinforcement learning in medical automated diagnosis: A review and applications," *Journal of Artificial Intelligence in Medicine*, vol. 7, no. 1, pp. 45–60, 2024.

[15] Brown, A., et al., "Ensemble methods in medical image classification: A focus on GANs," *IEEE Access*, vol. 10, pp. 12345–12358, 2022.

[16] Bhuvaneswari, M., et al., "Classifying COVID-19 infections using an ensemble of classifiers: A deep learning perspective," *Computers in Biology and Medicine*, vol. 132, pp. 104–116, 2021.

[17] Diviya M, Koushik Raghav S, Parthiban R, Udhayakumar S., "Sensible Autonomous Machine Using Deep Learning and Convolutional Neural Networks," In: Suresh, P., Saravanakumar, U., Hussein Al Salameh, M. (eds) *Advances in Smart System Technologies. Advances in Intelligent Systems and Computing*, vol. 1163, Springer, Singapore, 2021, doi: 10.1007/978-981-15-5029-4_50.

[18] Diviya M, A., "Deep neural architecture for natural language image synthesis for Tamil text using BASEGAN and hybrid super resolution GAN (HSRGAN)," *Scientific Reports*, vol. 13, pp. 14455, 2023, doi: 10.1038/s41598-023-41484-9.