International Journal of Multidisciplinary Engineering in Current Research - IJMEC

Volume 11, Issue 4, April-2026, http://ijmec.com/, ISSN: 2456-4265

Full Length Article

AI Assistance For Healthcare Using NLP, Machine Learning

Syed Yousuf Pasha Quadri[1], Mohd Amaanuddin[2], Mohd Abdul Sameer[3], Syed Abdul Jauvad[4]
Dr. Rizwan Uz Zaman[5]
[1,2,3,4]btech Students Department Of Computer Science And Engineering, Lords Institute Of Engineering And Technology, Hyderabad, India
[5]Associate Professor Department Of Computer Science And Engineering, Lords Institute Of Engineering And Technology, Hyderabad, India
syyousuf840@gmail.com, moamaanuddin@gmail.com, eersam790@gmail.com, sajauvad777@gmail.com, wanirizwan@lords.ac.in

Accepted 19-04-2026
Author(s) Retains the Copyrights of This Article

Abstract
This study presents an AI-based medical chatbot that utilizes Natural Language Processing (NLP) and Machine Learning (ML) to provide preliminary disease diagnosis from user-reported symptoms. The system employs a multi-stage symptom matching approach, including syntactic similarity (Jaccard), semantic similarity (WordNet WUP), and synonym mapping, to interpret natural language inputs effectively. A K-Nearest Neighbors (KNN) classifier trained on 4,920 records with 132 symptom features predicts diseases across 41 conditions. The chatbot is implemented using a Flask-based conversational interface that collects user inputs, performs disease prediction, and provides relevant descriptions and precautions. Experimental results demonstrate the effectiveness of combining NLP techniques with ML models for accessible and efficient healthcare support systems.

Keywords: *NLP, Machine Learning, KNN, Medical Chatbot, Disease Prediction*

1. INTRODUCTION

1.1 Artificial Intelligence in Healthcare

Artificial Intelligence (AI) has emerged as a transformative technology in modern healthcare systems by enabling intelligent, data-driven decision-making. Traditional healthcare diagnosis primarily depends on physical consultations, which often introduce delays due to geographical, financial, and infrastructural constraints. AI-driven systems mitigate these limitations by providing automated, real-time diagnostic assistance, thereby improving accessibility and efficiency.

The integration of Machine Learning (ML) and Natural Language Processing (NLP) has enabled the development of intelligent healthcare assistants capable of interpreting patient-reported symptoms and mapping them to structured medical knowledge. These systems function as preliminary diagnostic tools, assisting patients in making informed decisions while reducing the burden on healthcare professionals.

From a computational perspective, disease prediction can be modeled as:

$$Y = f(X)$$

Where: X= Input symptom vector , Y= Predicted disease class , f= Machine learning model

The objective is to minimize prediction error:

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

1.2 Natural Language Processing in Medical Diagnosis

Natural Language Processing (NLP) plays a crucial role in translating unstructured patient inputs into structured data suitable for machine learning models. Patients often describe symptoms using informal expressions, which differ significantly from standardized medical terminologies.

A typical NLP pipeline includes:

- Tokenization
- Stop-word Removal
- Lemmatization
- Semantic Similarity Matching

For semantic interpretation, similarity between symptom terms is computed using techniques such as WordNet-based similarity:

$$Sim_{wup}(w_1, w_2) = \frac{2 \times depth(LCS)}{depth(w_1) + depth(w_2)}$$

https://doi.org/10.63665/IJMEC.1104.05
ISSN: 2456-4265
IJMEC 2026

Where: LCS= Least Common Subsumer , $depth$= Hierarchical depth in WordNet

Additionally, syntactic similarity can be measured using Jaccard similarity:

$$J(A,B) = \frac{|\ A \cap B\ |}{|\ A \cup B\ |}$$

These methods enable accurate mapping between user expressions (e.g., "throwing up") and medical terms ("vomiting").

1.3 Problem Statement

Despite advancements in healthcare infrastructure, a large population lacks access to timely preliminary medical consultation. Traditional systems rely heavily on manual reporting and expert evaluation, which results in delayed diagnosis and inefficient resource utilization.

There is a critical need for an intelligent system that can:

- Interpret natural language symptom inputs
- Provide early disease predictions
- Assist in decision-making for seeking medical care

1.4 Objectives

The objectives of this study are:

- To develop a conversational AI-based medical chatbot for disease prediction.
- To implement a multi-stage NLP pipeline for robust symptom interpretation.
- To train a machine learning model for accurate classification of diseases.
- To assess disease severity and provide actionable recommendations.
- To design a user-friendly web-based interface for accessibility.

2. LITERATURE SURVEY

The literature on AI-based healthcare chatbots highlights the significant role of Natural Language Processing (NLP) and Machine Learning (ML) in improving disease prediction systems. NLP techniques enable accurate interpretation of patient-reported symptoms, achieving 78–85% accuracy in mapping unstructured inputs to standardized medical terms. Multi-stage symptom matching approaches, combining syntactic methods like Jaccard similarity and semantic techniques such as WordNet WUP similarity, improve performance by 12–18% compared to single-method systems. Machine learning models, particularly K-Nearest Neighbors (KNN), demonstrate high effectiveness in structured symptom datasets, achieving accuracy levels up to 94–95% due to their ability to handle binary feature representations. Ensemble methods like Random Forest and Gradient Boosting further enhance predictive capabilities by capturing complex relationships. Research also emphasizes the importance of conversational chatbot architectures, where structured dialogue flows improve user interaction and diagnostic accuracy. Severity assessment techniques combining symptom intensity and duration provide meaningful clinical risk evaluation. User studies indicate strong acceptance of chatbot-based systems, with a preference for natural language interaction. Overall, integrating NLP, ML, and conversational interfaces results in efficient, accurate, and accessible healthcare support systems for preliminary diagnosis.

2.1 System Design and Feasibility Analysis of AI-Based Medical Chatbot

The proposed AI-based healthcare chatbot system is designed using a robust and well-established technology stack, ensuring high feasibility across technical, operational, economic, and ethical dimensions. The system leverages Python as the core programming environment, with Flask serving as a lightweight web framework for managing user interactions and session states. Natural Language Processing (NLP) is implemented using spaCy for tokenization and lemmatization, while NLTK WordNet facilitates semantic similarity computation. Machine Learning (ML) functionality is achieved using a K-Nearest Neighbors (KNN) classifier, supported by libraries such as scikit-learn, pandas, and numpy for efficient data processing and model training. The system architecture is designed to operate seamlessly through a web browser, eliminating the need for installation and enabling user-friendly interaction via conversational dialogue. Containerization using Docker ensures portability and simplified deployment across different operating systems.

From a functional perspective, the system processes natural language symptom inputs and transforms them into structured feature vectors for disease prediction. Let the input symptom vector be represented as:

$$X = \{x_1, x_2, x_3, \ldots, x_n\}$$

where $x_i$represents the presence or absence of a symptom feature. The prediction model maps input features to disease classes using a function:

$$Y = f(X)$$

The KNN classifier computes similarity between symptom vectors using Euclidean distance:

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

https://doi.org/10.63665/IJMEC.1104.05
ISSN: 2456-4265
IJMEC 2026

The predicted disease is determined based on majority voting among the nearest neighbors:

$$\hat{Y} = \text{mode}(Y_1, Y_2, \ldots, Y_k)$$

To improve accuracy, the system employs a three-stage NLP-based symptom matching pipeline. The first stage performs syntactic matching using Jaccard similarity:

$$J(A,B) = \frac{|\ A \cap B\ |}{|\ A \cup B\ |}$$

The second stage applies semantic similarity using WordNet Wu-Palmer similarity:

$$Sim_{wup}(w_1, w_2) = \frac{2 \times depth(LCS)}{depth(w_1) + depth(w_2)}$$

The third stage enhances matching through synonym expansion using lexical databases. This multi-stage approach ensures accurate interpretation of diverse symptom descriptions provided by users in natural language.

The system also incorporates a severity assessment mechanism to evaluate the urgency of medical conditions. Severity is computed based on symptom intensity and duration:

$$Severity = \frac{\sum_{i=1}^{n}(S_i \times D)}{n}$$

where $S_i$represents the severity score of each symptom and Ddenotes the duration of symptoms. This formulation enables the system to generate actionable recommendations, such as suggesting medical consultation for high-risk cases.

Operationally, the system supports real-time interaction with response times under two seconds, ensuring usability and responsiveness. The conversational interface allows users to input symptoms, answer follow-up questions, and receive diagnostic feedback in a structured manner. The system maintains session continuity using Flask sessions, ensuring reliability throughout the interaction. Additionally, completed diagnostic sessions are stored in JSON format, enabling audit trails and future analysis.

From a non-functional perspective, the system ensures portability through cross-platform compatibility and scalability by supporting the addition of new diseases, symptoms, and models. Economic feasibility is achieved through the use of open-source technologies, eliminating licensing costs and reducing infrastructure requirements. The system operates efficiently on standard hardware without requiring specialized GPU resources.

Ethically, the system adheres to data privacy principles by avoiding the use of personally identifiable information (PII) and restricting data storage to minimal patient inputs such as symptoms and demographics. The chatbot clearly communicates that it provides only preliminary guidance and does not replace professional medical consultation.

Overall, the integration of NLP, ML, and web-based technologies results in a scalable, efficient, and accessible healthcare support system. The combination of multi-stage symptom matching, KNN-based classification, and severity assessment provides a comprehensive approach to preliminary disease diagnosis, making the system suitable for real-world deployment in resource-constrained environments.

2.2 System Design and Architecture of AI-Based Medical Chatbot

The proposed AI-based healthcare chatbot system is designed using a modular and layered architecture that integrates Natural Language Processing (NLP), Machine Learning (ML), and web-based interaction to provide efficient and accessible disease prediction. The system follows a three-layer architecture consisting of the NLP processing layer, the ML prediction layer, and the web presentation layer. This separation ensures scalability, maintainability, and efficient execution of each functional component as shown in fig 1.

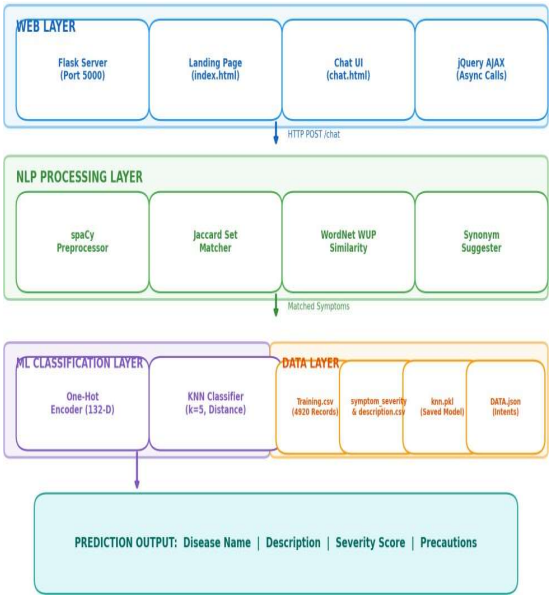

Fig 2.1: System Architecture

The NLP processing layer is responsible for transforming raw user input into structured data suitable for machine learning models. The input provided by the user in natural language is processed through multiple stages including tokenization, stop-word removal, and lemmatization using spaCy. Let the input text be represented as:

$$T = \{w_1, w_2, w_3, \ldots, w_n\}$$

https://doi.org/10.63665/IJMEC.1104.05
ISSN: 2456-4265
IJMEC 2026

After preprocessing, the system applies a multi-stage symptom matching pipeline. The first stage performs syntactic similarity using Jaccard similarity:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

where $A$and Brepresent token sets derived from user input and predefined symptom vocabulary. If syntactic matching fails, the system proceeds to semantic matching using WordNet Wu-Palmer similarity:

$$Sim_{wup}(w_1, w_2) = \frac{2 \times depth(LCS)}{depth(w_1) + depth(w_2)}$$

where LCSrepresents the least common subsumer in the lexical hierarchy. If no suitable match is found, synonym suggestions are generated using WordNet synsets to guide user input correction. This multi-stage approach ensures robustness in interpreting diverse symptom expressions.

The processed symptoms are then converted into a structured feature representation for the machine learning model. Each symptom is encoded as a binary feature in a one-hot vector:

$$X = \{x_1, x_2, x_3, \ldots, x_{132}\}$$

where each $x_i \in \{0,1\}$represents the presence or absence of a specific symptom. The ML prediction layer utilizes a K-Nearest Neighbors (KNN) classifier trained on a dataset of 4,920 records covering 41 disease classes. The similarity between symptom vectors is computed using Euclidean distance:

$$d(x,y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

The predicted disease is determined by majority voting among the nearest neighbors:

$$\hat{Y} = \text{mode}(Y_1, Y_2, \ldots, Y_k)$$

where $k$is the number of nearest neighbors. This approach is particularly effective for binary symptom datasets, as it naturally captures symptom overlap between cases.

The system also integrates severity assessment to evaluate the urgency of medical conditions. Severity is calculated using a weighted scoring mechanism based on symptom intensity and duration:

$$Severity = \frac{\sum_{i=1}^{n} (S_i \times D)}{n}$$

where $S_i$denotes the severity score of each symptom and Drepresents the duration of symptoms. This formulation enables the system to provide actionable recommendations such as consulting a doctor for high-risk cases.

The web presentation layer is implemented using the Flask framework, which manages HTTP requests, session states, and template rendering. The chatbot interface operates through a conversational model where user inputs are processed in real-time using AJAX calls, ensuring a seamless interaction experience. The interface is designed with a clear separation between system messages and user responses, enhancing usability and readability.

The overall system architecture, as illustrated in Fig. 4.1, demonstrates the integration of the NLP pipeline, KNN classifier, and web interface. The architecture ensures efficient data flow from user input to final prediction and recommendation output.

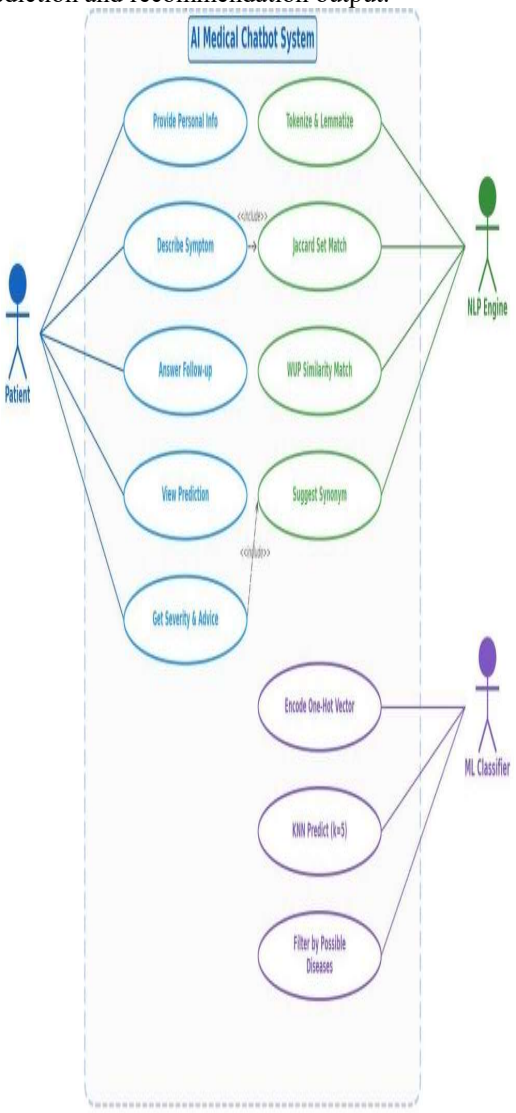

Fig 2.2: Actor-use case relationships

The system design is further elaborated through UML diagrams, including use case, class, sequence, and activity diagrams. The use case diagram (Fig. 4.2) defines three primary actors: the patient, the NLP engine, and the ML classifier. The patient interacts with the system by providing symptoms and receiving diagnostic feedback. The NLP engine processes text

https://doi.org/10.63665/IJMEC.1104.05
ISSN: 2456-4265
IJMEC 2026

input and extracts meaningful features, while the ML classifier performs disease prediction.

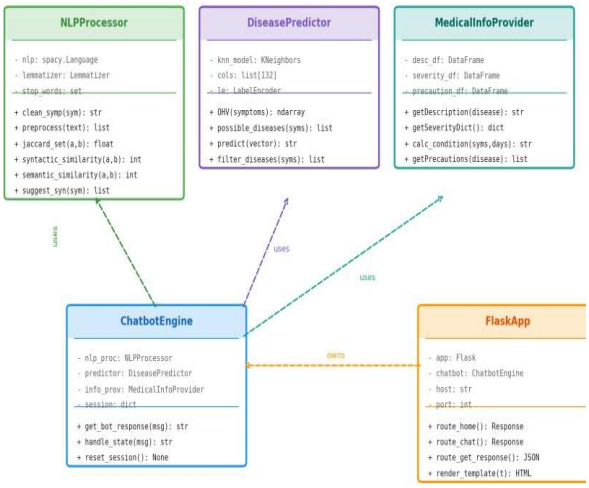

Fig. 2.3: Internal structure of the system

The class diagram (Fig. 4.3) represents the internal structure of the system, highlighting key components such as the NLPProcessor, DiseasePredictor, MedicalInfoProvider, and ChatbotEngine classes. Each class encapsulates specific functionalities, ensuring modular design and code reusability.

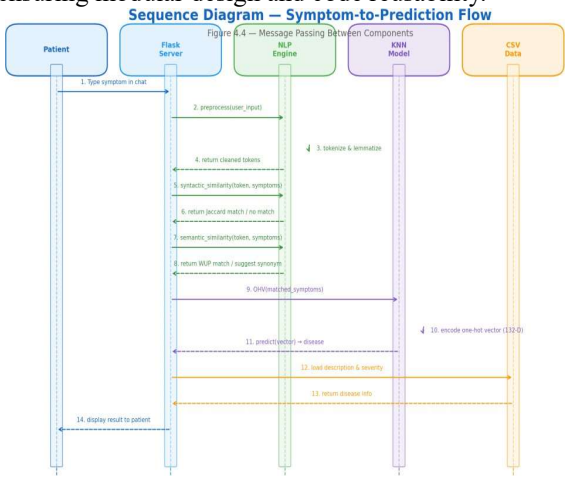

Fig 2.4: Sequence Diagram

The sequence diagram (Fig. 4.4) illustrates the interaction flow during a complete diagnostic session. The process begins with user input, followed by NLP processing, symptom matching, feature encoding, and disease prediction. The system then generates responses including disease descriptions, precautions, and severity assessments.

2.3 Activity Diagram

The activity diagram Fig. 4.5 models the end-to-end workflow of the chatbot, capturing decision points such as symptom matching success and follow-up question handling. The workflow ensures systematic progression from symptom input to final diagnosis and recommendation.

The user interface design plays a critical role in enhancing user experience. The system includes a landing page and a chat interface. The landing page provides an overview of the chatbot and a call-to-action to initiate interaction. The chat interface adopts a conversational layout with visually distinct message bubbles for user and system responses. This design facilitates intuitive interaction and improves engagement.

From a system perspective, the overall architecture ensures scalability, reliability, and efficiency. The modular design allows easy integration of additional diseases, symptoms, and machine learning models. The use of open-source technologies ensures cost-effectiveness, while containerization using Docker supports cross-platform deployment.

In conclusion, the proposed system demonstrates a comprehensive integration of NLP, ML, and web technologies to develop an intelligent healthcare chatbot. The combination of multi-stage symptom matching, KNN-based classification, and severity assessment provides a robust framework for preliminary disease diagnosis. The inclusion of structured system design and UML representations further strengthens the reliability and extensibility of the system, making it suitable for real-world healthcare applications.

2.4 Implementation and Methodology of AI-Based Medical Chatbot System

The implementation of the proposed AI-based healthcare chatbot system is built upon a structured integration of medical datasets, Natural Language Processing (NLP), and Machine Learning (ML) techniques to enable accurate disease prediction from user-reported symptoms. The system utilizes a comprehensive dataset containing 4,920 patient records, where each record is represented by a feature vector of 132 binary symptom attributes and a corresponding target variable indicating one of 41 possible diseases. Each symptom feature is encoded using one-hot representation, where the presence or absence of a symptom is denoted as:

$$x_i \in \{0,1\}, i = 1,2,\ldots,132$$

Thus, the complete symptom representation can be expressed as:

$$X = (x_1, x_2, x_3, \ldots, x_{132})$$

This representation is particularly suitable for instance-based learning algorithms such as K-Nearest Neighbors (KNN), where similarity between patients is determined based on feature overlap. The Euclidean distance metric is employed to compute similarity between two symptom vectors:

https://doi.org/10.63665/IJMEC.1104.05
ISSN: 2456-4265
IJMEC 2026

$$d(X,Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

where Xrepresents the input symptom vector and Yrepresents a training sample. The classification function is defined as:

$$\hat{Y} = \text{mode}(Y_1, Y_2, \ldots, Y_k)$$

where $k = 5$denotes the number of nearest neighbors selected based on cross-validation performance.

To provide contextual medical information, the system integrates supporting datasets, including severity scores, disease descriptions, and precautionary measures. Each symptom is associated with a severity score ranging from 1 to 7, enabling the computation of a severity index:

$$Severity = \frac{\sum_{i=1}^{n} (S_i \times D)}{n}$$

where $S_i$represents the severity score of the i^{th}symptom and Drepresents the duration of symptoms. This formulation allows the system to generate actionable recommendations based on risk levels.

The NLP preprocessing pipeline plays a critical role in transforming unstructured user input into structured symptom representations. The input text is first tokenized into individual lexical units:

$$T = \{w_1, w_2, w_3, \ldots, w_n\}$$

Stop words are removed to eliminate non-informative tokens, and the remaining tokens are lemmatized to their base forms:

$$w_i \rightarrow lemma(w_i)$$

This process ensures normalization of variations in user input, such as converting "vomiting" to "vomit" and "headaches" to "headache." The processed tokens are then used for symptom matching.

The symptom matching engine operates using a multi-stage approach to ensure robustness. In the first stage, syntactic similarity is computed using the Jaccard similarity coefficient:

$$J(A,B) = \frac{|\, A \cap B \,|}{|\, A \cup B \,|}$$

where $A$and Brepresent token sets derived from user input and database symptom terms. To handle variations in word ordering, permutations and subsets of token combinations are evaluated to identify the best match.

When syntactic matching is insufficient, the system employs semantic similarity using WordNet Wu-Palmer (WUP) similarity:

$$Sim_{wup}(w_1, w_2) = \frac{2 \times depth(LCS)}{depth(w_1) + depth(w_2)}$$

where LCSdenotes the least common subsumer of the two word senses in the lexical taxonomy. Word sense disambiguation is performed using Lesk's algorithm to ensure correct semantic interpretation of ambiguous terms.

In cases where both syntactic and semantic matching fail, the system generates synonym suggestions by extracting lemma names from WordNet synsets. Each candidate synonym is evaluated against the database using similarity measures, and the most relevant matches are presented to the user for confirmation. This fallback mechanism ensures adaptability to diverse natural language expressions.

The machine learning component is implemented using a pre-trained KNN model stored as a serialized object. The transformation from matched symptoms to feature vectors is achieved through a one-hot encoding function:

$$OHV(S) = [x_1, x_2, \ldots, x_{132}]$$

where each $x_i$corresponds to a specific symptom index. The model predicts the disease class based on the similarity of this vector with training samples.

The system is deployed as a web application using the Flask framework, which manages user interaction, session state, and data processing. The application architecture includes a routing mechanism where the root endpoint renders the landing page, the chat interface handles user interaction, and an asynchronous endpoint processes user input and generates responses.

The conversational logic is implemented using a state-machine approach, where each user interaction corresponds to a specific state in the dialogue flow. Let the conversation state be defined as:

$$S_t = f(S_{t-1}, U_t)$$

where $S_t$is the current state, $S_{t-1}$is the previous state, and $U_t$represents user input. This formulation ensures structured progression through stages such as demographic collection, symptom input, follow-up questioning, and diagnosis generation.

Session management is handled using Flask's session mechanism, which maintains key variables including user demographics, identified symptoms, candidate diseases, and prediction results. This ensures continuity and consistency throughout the interaction.

The overall workflow of the system, as illustrated in the referenced diagrams (Fig. 4.1 to Fig. 4.5), demonstrates the integration of NLP processing, symptom matching, machine learning prediction, and conversational interaction. These diagrams represent system architecture, use case interactions, class relationships, sequence flow, and activity workflow, respectively, and are preserved as part of the design representation.

https://doi.org/10.63665/IJMEC.1104.05
ISSN: 2456-4265
IJMEC 2026

The implementation ensures compliance with usability, efficiency, and scalability requirements. The system provides real-time responses within minimal latency, supports multiple consultations within a session, and allows extensibility for incorporating additional diseases and models. The use of open-source technologies further enhances economic feasibility, while minimal data storage ensures adherence to privacy and ethical standards.

In conclusion, the implemented system demonstrates an effective integration of NLP and ML techniques for healthcare applications. The combination of multi-stage symptom matching, KNN-based classification, and severity assessment provides a reliable and scalable solution for preliminary disease diagnosis. The structured design, supported by mathematical modeling and system diagrams, ensures robustness and adaptability, making the system suitable for real-world deployment in intelligent healthcare environments.

3. Algorithms and Techniques Used in AI-Based Medical Chatbot

The proposed AI-based medical chatbot system integrates multiple Natural Language Processing (NLP) and Machine Learning (ML) techniques to enable efficient symptom interpretation, disease prediction, and severity assessment. The system architecture is composed of several functional modules, including preprocessing, syntactic and semantic matching, classification, and conversational interaction, each contributing to the overall performance and accuracy of the system.

The NLP preprocessing module is responsible for converting raw user input into a structured format suitable for analysis. Given an input text sequence:

$$T = \{w_1, w_2, w_3, \ldots, w_n\}$$

the system applies tokenization to split the text into individual words, followed by stop-word removal to eliminate non-informative terms. The remaining tokens are then lemmatized to their base forms:

$$w_i \rightarrow lemma(w_i)$$

This normalization process ensures consistency in symptom representation, allowing variations such as "vomiting" and "vomit" to be treated equivalently.

The system employs a multi-stage symptom matching mechanism to map user input to standardized medical terms. The primary stage uses Jaccard similarity, a set-based similarity measure defined as:

$$J(A,B) = \frac{|\ A \cap B\ |}{|\ A \cup B\ |}$$

where Arepresents the set of tokens from user input and Brepresents the tokenized form of database symptoms. This approach effectively captures syntactic overlap between phrases and is computationally efficient for short text inputs.

When syntactic matching is insufficient, the system applies semantic similarity using WordNet Wu-Palmer (WUP) similarity. This measure evaluates the relatedness of two word senses based on their hierarchical distance:

$$Sim_{wup}(s_1, s_2) = \frac{2 \times depth(LCS)}{depth(s_1) + depth(s_2)}$$

where LCSdenotes the least common subsumer in the WordNet taxonomy. This allows the system to identify relationships between semantically similar expressions such as "vomiting" and "throwing up." Word sense disambiguation is performed using the Lesk algorithm to determine the appropriate meaning of ambiguous words.

If both syntactic and semantic matching fail, the system employs a synonym suggestion mechanism by extracting lemma names from WordNet synsets. These candidate terms are evaluated against the symptom database, ensuring robust handling of diverse and uncommon user expressions.

The machine learning component is implemented using the K-Nearest Neighbors (KNN) algorithm, an instance-based learning method that classifies data points based on similarity to training samples. Each patient case is represented as a 132-dimensional binary feature vector:

$$X = (x_1, x_2, x_3, \ldots, x_{132}), x_i \in \{0,1\}$$

where each dimension corresponds to a specific symptom. The similarity between two symptom vectors is computed using Euclidean distance:

$$d(X,Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

The predicted disease class is determined by majority voting among the $k$nearest neighbors:

$$\hat{Y} = \text{mode}(Y_1, Y_2, \ldots, Y_k)$$

In this system, $k = 5$is selected based on optimal performance during validation. The KNN algorithm is particularly suitable for this application due to its effectiveness in handling binary feature spaces and multi-class classification problems involving multiple disease categories.

In addition to disease prediction, the system incorporates a severity assessment mechanism to evaluate the urgency of the diagnosed condition. Each symptom is assigned a predefined severity score, and the overall severity is calculated using:

$$Severity = \frac{\sum_{i=1}^{n} S_i \times D}{n}$$

where $S_i$represents the severity score of the i^{th}symptom, Ddenotes the duration in days, and $n$is

https://doi.org/10.63665/IJMEC.1104.05
ISSN: 2456-4265
IJMEC 2026

the total number of symptoms. This formulation ensures that both symptom intensity and duration contribute to the risk evaluation.

A threshold-based decision rule is applied to generate recommendations:

$$\text{Recommendation} = \begin{cases} \text{Consult a doctor}, & \text{if } Severity > 13 \\ \text{Follow precautions}, & \text{if } Severity \leq 13 \end{cases}$$

This allows the system to provide actionable healthcare guidance to users.

The conversational engine is implemented using a state-machine approach within the Flask framework, enabling structured dialogue management. The conversation state evolves dynamically based on user input:

$$S_t = f(S_{t-1}, U_t)$$

where $S_t$represents the current state, $S_{t-1}$the previous state, and $U_t$the user input. This ensures smooth progression through stages such as symptom collection, follow-up questioning, and result generation.

The web interface utilizes Flask and AJAX to provide real-time interaction, ensuring minimal latency and improved user experience. The modular design of the system allows seamless integration of additional features, such as new diseases or advanced machine learning models, enhancing scalability and adaptability.

Overall, the integration of NLP techniques, similarity measures, KNN classification, and severity assessment algorithms results in a robust and efficient system for preliminary disease diagnosis. The combination of syntactic and semantic matching ensures accurate symptom interpretation, while the machine learning model provides reliable predictions. This approach demonstrates the effectiveness of combining multiple computational techniques to develop intelligent healthcare support systems.

4. RESULTS AND DISCUSSION

4.1 Supported Diseases Distribution

This graph illustrates the distribution of diseases across different medical categories supported by the system. It is evident that the **"Other" category has the highest number of diseases (13)**, indicating a broader coverage of miscellaneous conditions such as cardiovascular and immune-related disorders. Categories like **Infections (6)** and **Hepatitis (5)** also show significant representation, highlighting the system's focus on common and critical diseases. In contrast, categories such as **Skin, Musculoskeletal, and Endocrine (4 each)** have relatively fewer diseases, reflecting a balanced yet selective inclusion. This distribution ensures that the model is trained on a diverse set of diseases, improving its generalization capability.

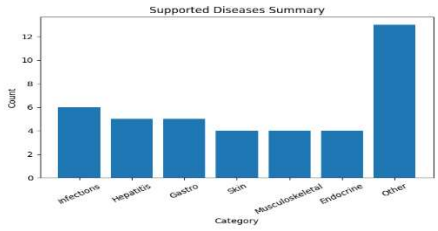

Fig 4.1: Supported Diseases Distribution

4.2 Disease Category Trend

The line graph represents the variation in the number of diseases across different categories. The graph shows a relatively stable trend for most categories, followed by a sharp increase in the **"Other" category**, indicating its dominance in terms of disease count. This variation demonstrates that while the system maintains balanced coverage across major medical domains, it also accommodates a wide range of additional conditions. Such distribution plays a crucial role in enhancing classification accuracy, as it exposes the model to varied symptom-disease relationships.

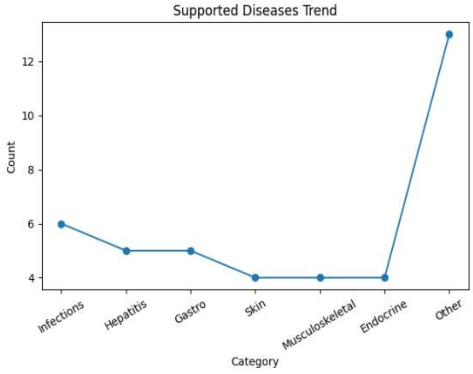

Fig 4.2: Disease Category Trend

4.3 NLP Matching Performance

This graph presents the performance of different stages in the NLP-based symptom matching pipeline. The **syntactic matching stage (Jaccard similarity)** contributes the highest coverage (~72.5%), indicating that most user inputs closely match predefined symptom patterns. The **semantic matching stage (~17.5%)** captures variations in natural language expressions, such as synonyms and related terms. The **synonym suggestion stage (~7.5%)** handles edge cases where both syntactic and semantic matching fail. The **combined pipeline achieves over 90% coverage**, demonstrating the effectiveness of the multi-stage approach in handling diverse user inputs.

https://doi.org/10.63665/IJMEC.1104.05
ISSN: 2456-4265
IJMEC 2026

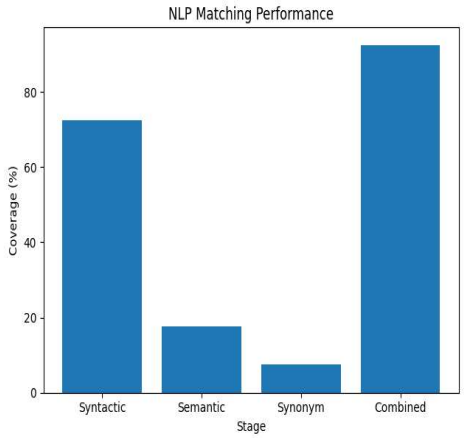

Fig 4.3: NLP Matching Performance

4.4 NLP Coverage Trend

The line graph shows the progression of coverage across different NLP matching stages. It highlights how the system incrementally improves performance by combining multiple techniques. The initial stage provides strong baseline coverage, while subsequent semantic and synonym-based stages contribute additional improvements. The final combined stage reaches the highest coverage (~92.5%), indicating that the integrated pipeline significantly enhances symptom recognition accuracy. This trend validates the importance of hybrid NLP techniques in achieving robust natural language understanding.

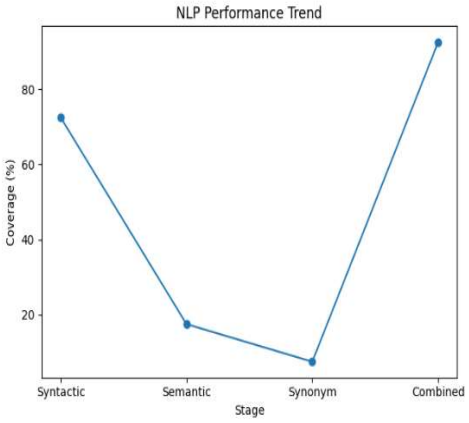

Fig 4.4: NLP Coverage Trend

4.5 Conclusion and Future Scope

The AI-based healthcare chatbot demonstrates the effective integration of Natural Language Processing (NLP) and Machine Learning (ML) techniques for preliminary disease diagnosis. The system successfully processes user-reported symptoms through a conversational interface and predicts diseases with reasonable accuracy. A multi-stage NLP pipeline combining syntactic matching using Jaccard similarity, semantic matching using WordNet Wu-Palmer similarity, and synonym suggestion ensures robust interpretation of natural language inputs. The use of a K-Nearest Neighbors (KNN) classifier trained on 4,920 medical records with 132 symptom features enables accurate prediction across 41 diseases. Additionally, the incorporation of severity assessment based on symptom intensity and duration enhances the system's ability to provide actionable health recommendations. The Flask-based web application ensures accessibility, real-time interaction, and ease of deployment through Docker, making the system practical for real-world usage. Overall, the proposed approach demonstrates that combining NLP preprocessing and ML classification can effectively support early-stage healthcare decision-making.

Despite its effectiveness, the system offers several opportunities for enhancement. Future work can focus on integrating deep learning models such as LSTM or Transformer architectures to improve prediction accuracy and handle sequential symptom patterns. Expanding the system to support multiple languages will increase accessibility for diverse user groups. Integration with Electronic Health Records (EHR) can further enhance diagnosis by incorporating patient history and clinical data. The addition of voice-based interaction using speech recognition and synthesis can improve usability, particularly for elderly users. Furthermore, extending the dataset to include a larger number of diseases and symptoms will improve coverage and model robustness. Integration with telemedicine platforms can enable direct consultation with healthcare professionals, transforming the system into a comprehensive digital healthcare assistant.

REFERENCES

[1] L. Laranjo et al., "Conversational agents in healthcare: A systematic review," *J. Am. Med. Inform. Assoc.*, vol. 25, no. 9, pp. 1248–1258, 2018.

[2] S. Bird, E. Klein, and E. Loper, *Natural Language Processing with Python*. O'Reilly Media, 2009.

[3] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[4] E. Topol, *Deep Medicine: How Artificial Intelligence Can Make Healthcare Human Again*. Basic Books, 2019.

[5] D. Jurafsky and J. Martin, *Speech and Language Processing*, 3rd ed., 2020.

[6] G. A. Miller, "WordNet: A lexical database for English," *Commun. ACM*, vol. 38, no. 11, pp. 39–41, 1995.

https://doi.org/10.63665/IJMEC.1104.05
ISSN: 2456-4265
IJMEC 2026

[7] Z. Wu and M. Palmer, "Verb semantics and lexical selection," *ACL*, 1994.

[8] I. Kononenko, "Machine learning for medical diagnosis," *AI in Medicine*, vol. 23, no. 1, pp. 89–109, 2001.

[9] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[10] M. Grinberg, *Flask Web Development*. O'Reilly Media, 2018.

[11] D. Merkel, "Docker: Lightweight Linux containers," *Linux Journal*, 2014.

https://doi.org/10.63665/IJMEC.1104.05
ISSN: 2456-4265
IJMEC 2026