**HealthMate: An AI-Driven Symptom-Based Medical Diagnosis Chatbot.**

Moncef BENZIADA, Akram CHERBANE, Rihab LARIK, Fares NADJI and Chakib YOUNSI.

Numidia Institute of Technology, Rahmania, Algiers, Algeria.

**Abstract**

This paper documents the development of "Healthmate," an AI-driven symptom-based diagnosis chatbot application. Developed using Flutter for cross-platform compatibility, Healthmate leverages a sophisticated machine learning model (detailed within this report) to analyze user-inputted symptoms and provide potential diagnoses. The core AI model employs a Bagging ensemble of Random Forest classifiers, trained on a meticulously curated dataset of medical symptoms. Data preprocessing techniques, including text cleaning, feature engineering (combining 'text' and 'symptoms' features), and strategies for handling class imbalance (data augmentation and filtering) were implemented to enhance model accuracy and robustness. The backend for user registration and authentication is powered by Firebase, ensuring secure and scalable user management. This report provides a comprehensive overview of the development process, including design considerations, implementation details, and evaluation results, demonstrating the potential of AI-powered chatbots in improving healthcare accessibility and patient engagement.

**Keywords**

Symptom-Based Diagnosis, Medical chatbot, Machine Learning, Healthcare Accessibility, Flutter, Natural Language Processing (NLP), Firebase, Term Frequency-Inverse Document Frequency (TF-IDF), Random Forest, Application.

**Introduction**

The increasing accessibility of smartphones and the growing demand for convenient healthcare solutions have spurred the development of numerous mobile health applications. Among these, symptom checker applications have gained significant traction, offering users preliminary insights into potential health concerns based on their reported symptoms. Existing symptom checker applications, such as WebMD and Ada, leverage various AI techniques, including machine learning and natural language processing, to analyze user input and provide potential diagnoses.

However, many existing solutions face limitations, such as limited accuracy, reliance on overly simplistic algorithms, and a lack of user-centric design. This research aims to address these limitations by developing "HealthMate," a novel mobile application that utilizes a sophisticated AI model for symptom-based diagnosis.

HealthMate employs a Bagging ensemble of Random Forest classifiers, a powerful machine learning technique known for its robustness and accuracy in handling complex datasets. This approach leverages the collective strength of multiple decision trees, improving the model's ability to accurately predict potential diagnoses based on user-inputted symptoms. Furthermore, the application incorporates advanced natural language processing techniques, including TF-IDF (Term Frequency-Inverse Document Frequency) vectorization, to effectively extract meaningful features from textual symptom descriptions.

Developed using the Flutter framework, HealthMate prioritizes a user-centric design with a focus on intuitive navigation, clear instructions, and a visually appealing interface. The application leverages Firebase for secure user authentication and data management, ensuring a robust and scalable backend infrastructure. This research explores the development and evaluation of HealthMate, aiming to contribute to the advancement of AI-powered symptom checkers and enhance the accessibility of preliminary health assessments for users.

**Methodology**

1. Data Acquisition and Preprocessing:

* Dataset Collection: A comprehensive dataset of medical symptoms and their corresponding diagnoses was sourced from reputable medical databases and research publications.
* Data Cleaning: The dataset underwent rigorous cleaning to address inconsistencies, remove duplicates, and handle missing values. This involved identifying and correcting errors in symptom descriptions and diagnosis labels.
* Text Preprocessing: To prepare the textual symptom data for the AI model, the following preprocessing steps were implemented:
  + Lowercasing: Converting all text to lowercase to ensure consistency.
  + Punctuation Removal: Removing punctuation marks (e.g., periods, commas, exclamation points) that do not contribute to semantic meaning.
  + Stop Word Removal: Eliminating common words (e.g., "the," "a," "is") that have little to no semantic value.
  + Stemming/Lemmatization: Reducing words to their root form (e.g., "running" -> "run," "better" -> "good") to improve feature representation and reduce the dimensionality of the feature space.
* Feature Engineering:
  + Symptom Combination: The 'text' and 'symptoms' features were combined to create a more comprehensive feature set, capturing both textual descriptions and structured symptom information.
  + TF-IDF Vectorization: The combined text data was transformed into numerical features using TF-IDF, a technique that weights words based on their frequency within a document and their rarity across the entire dataset. This approach effectively highlights the most informative terms for the AI model.

2. Model Development:

* Model Selection: A Bagging ensemble of Random Forest classifiers was selected as the core machine learning model. Bagging, or Bootstrap Aggregating, trains multiple decision trees on different subsets of the training data and combines their predictions, improving model robustness and reducing overfitting.
* Model Training: The selected model was trained on the preprocessed and engineered dataset.
* Hyperparameter Tuning: A grid search approach was employed to systematically explore different combinations of hyperparameters for both the Random Forest classifier and the TF-IDF vectorizer. This process aimed to optimize model performance by finding the best combination of parameters for the given dataset.
* Model Evaluation: The trained model was rigorously evaluated using a stratified k-fold cross-validation technique to assess its performance on unseen data. Key performance metrics, including accuracy, precision, recall, F1-score, and area under the ROC curve (AUC), were calculated to evaluate the model's effectiveness in predicting diagnoses.

3. Frontend Development:

* Flutter Framework: The Flutter framework was chosen for its cross-platform capabilities, enabling the development of a visually appealing and performant application for both iOS and Android platforms.
* User Interface (UI) Design: The application's UI was designed with a focus on user experience, incorporating intuitive navigation, clear instructions, and a visually appealing aesthetic.
* Custom Widgets: Reusable custom widgets were developed for common UI elements such as text input fields, buttons, and navigation elements, promoting code reusability and consistency.
* State Management: A suitable state management approach was implemented to manage the application's state effectively, ensuring smooth data flow and user interactions.
* Backend Integration: The frontend was integrated with a secure backend infrastructure, utilizing Firebase for user authentication, data storage, and secure communication with the AI model.

4. Testing and Evaluation:

* Thorough Testing: Rigorous testing was conducted throughout the development process, including unit tests for individual components and integration tests to ensure seamless functionality.
* User Feedback: The application was evaluated through user feedback to identify potential usability issues and areas for improvement in the user interface and user experience.

**Results**

The developed "HealthMate" application demonstrated promising results in preliminary evaluations. The core AI model, a Bagging ensemble of Random Forest classifiers, achieved an overall accuracy of 84% on the test dataset, indicating its effectiveness in predicting potential diagnoses based on user-inputted symptoms.

A key strength of the model was its high precision, ensuring that when a diagnosis was predicted, it was highly likely to be correct. This minimizes the risk of misinforming users with incorrect or misleading diagnoses. Furthermore, the application exhibited a rapid response time, providing users with potential diagnoses quickly and efficiently. This fast response time enhances the user experience and contributes to the application's practical utility.

**Discussion and Future Work**

The results of this research demonstrate the feasibility of developing an AI-powered symptom checker application that can effectively analyze user-inputted symptoms and provide preliminary health insights. The achieved accuracy of 84% demonstrates the potential of the chosen AI model architecture and the employed data preprocessing techniques in addressing the challenges of medical text classification.

However, further research and development are necessary to enhance the application's capabilities and address potential limitations. Future works will focus on:

* Medication Reminders: Integrating a medication reminder system to assist users in adhering to prescribed medication schedules. This feature will enhance patient adherence to treatment plans and improve overall health outcomes.
* Appointment Scheduler: Implementing an appointment scheduling feature that allows users to easily book appointments with healthcare providers directly through the application. This will streamline the healthcare access process and improve patient convenience.
* Multilingual Support: Expanding the application's language support to cater to a wider global audience, making it accessible to users in diverse linguistic contexts.
* Integration with Wearable Devices: Exploring the integration of HealthMate with wearable devices to collect real-time health data, such as heart rate, sleep patterns, and activity levels. This integration will provide more comprehensive health insights and enable more personalized health assessments.
* Advanced AI Models: Investigating the use of more sophisticated AI models, such as deep learning models (e.g., Recurrent Neural Networks, Convolutional Neural Networks), to further enhance diagnostic accuracy and explore more complex relationships within the data.
* Continuous Learning: Implementing mechanisms for continuous learning and model updates to incorporate new medical knowledge, address emerging health concerns, and ensure the model's accuracy and effectiveness remain up-to-date.
* User Feedback Integration: Incorporating user feedback mechanisms, such as surveys and in-app feedback forms, to continuously improve the application's functionality, address user needs, and enhance the overall user experience.

Furthermore, rigorous clinical validation and regulatory approval are essential before widespread clinical adoption of such an application. Continuous monitoring and evaluation of the application's performance and impact will be crucial to ensure its safety, effectiveness, and ethical use.

**References**

[1] D. Ameta, K. Mudaliar, and P. Patel, "Medication Reminder and Healthcare \u2013 An Android Application," *International Journal of Managing Public Sector Information and Communication Technologies*, vol. 6, no. 2, pp. 39-48, 2015, doi: 10.5121/ijmpict.2015.6204.

[2] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang, "BioBERT: A pre-trained biomedical language representation model for biomedical text mining," *Bioinformatics*, vol. 36, no. 4, pp. 1234-1240, 2020.

[3] Y. Liu and J. Huang, "A review of applying deep learning to medical diagnosis," *International Journal of Neural Networks*, vol. 23, no. 1, pp. 213-224, 2020.

[4] Hugging Face, "Medical Datasets." [Online]. Available: <https://huggingface.co>. [Accessed: Jan. 12, 2025].

[5] TensorFlow and PyTorch Official Documentation. [Online]. Available: <https://tensorflow.org>, <https://pytorch.org>. [Accessed: Jan. 12, 2025].

[6] Scikit-learn, "Machine Learning in Python." [Online]. Available: <https://scikit-learn.org>. [Accessed: Jan. 12, 2025].

[7] Flutter, "The Flutter Documentation." [Online]. Available: <https://flutter.dev/>. [Accessed: Jan. 12, 2025].

[8] Firebase, "Firebase Documentation." [Online]. Available: <https://firebase.google.com>. [Accessed: Jan. 12, 2025].