# Symptoms Checker for Disease Detection Using Agentic AI

**S.Tejaswi[1], R.Manjeera[2], Y.Karthik[3], G.Chandra Sekhar[4], T.Mahitha Bhargavi[5]**

Professor[1], Student[2], Student[3], Student[4], Student[5]

*Department of Artificial Intelligence*

*Chaitanya Engineering College , Visakhapatnam, AndhraPradesh ,India*

{tejaswi.svp@gmail.com,manjeeraranabheri03@gmail.com,karthikyedagiri@gmail.com, gantachandu95454@gmail.com,mahitimanani@gmail.com }

## Abstract

The system I am discussing in this paper is designed to enable more manageable easy accurate diagnosis to people, particularly in areas where physicians are scarce. We refer to it as an agentic-based Symptoms Checker, i.e., it has a number of small agents, which perform various tasks in a parallel manner. On our test data, we vote with six distinct models, Decision Tree, Random Forest, SVM, Logistic Regression, Naive Bayes, and K-NN, and the final guess of what the disease is, gives the models a good 99.74% overall accuracy. The tool does not just do this: it also transfers the prediction to the agents that propose the appropriate specialist and search the nearby hospitals using a live search API. Sentence embeddings provide a conversational agent with cloud-based access to receive personal advice by the user and interact with the agent in a chat-like manner. The app was developed using Flask but included session-based authentication and SQLite database allowing us to store prediction history, quiz score and chat records securely. We find this combination of all-in-one agentic pipeline more trustworthy in our real world tests than just using a single model and enhances access to healthcare in general. Hopefully, this project will be used as a template when it comes to other projects of this kind in digital health.

Keywords -- agentic AI, disease prediction, ensemble learning, symptom checker, healthcare decision support, conversational agent.

### I. Introduction

The world healthcare system is under pressure due to increased number of patients, lack of specialists in rural locations and dependence on healthcare diagnosis being based on intuition. The largest lever to improved patient outcomes and reduced downstream costs is early disease recognition, which in most of the low- and middle-income countries patients still must face a disjointed referral process, unreliable diagnostic methods, and limited digital assistance, and the resulting delays and incorrect diagnosis may be avoided.

Clinical decision support based on data is starting to become a major concern because of machine learning. ML classifiers can provide quick, reproducible forecasts with statistical connections between symptom patterns and a disease type akin to professional judgment since learning is based on pre-developed data. The issue is that even the systems that have been deployed rely on a single classifier, the biases of which and susceptibility to differing distribution shifts may lead to unreliable results when applied to heterogeneous groups of patients.

In comparison with traditional ML pipelines, the concept of agentic AI begins to deploy autonomous methods and purpose-oriented agents that do not merely perceive input data but think on the structure of organized knowledge and executes multiple steps compressing actions, coordination via services. A healthcare agentic architecture is able to encapsulate the entire process of gathering symptoms, ensemble inference, expert mapping, hospital look up, conversational answer, etc. within a single workflow and dramatically reduce the cognitive burden required to engage with patients and clinicians.

In the current paper, the authors present a Symptoms Checker, which is the composite of six ML models and downstream agentic modules to recommend a specialist, identify a hospital near where the patient is located, interactive health quizzes, or a contextual chatbot. Our key contributions are:

- a 99.74 percent accuracy ensemble voting classifier on a multi class symptom-disease dataset.

- Agentic pipeline automatically chaining the diagnosis, specialist mapping and real-time hospital retrieval.

- Semantically-based chatting agent incorporating sentence-transformers retrieval with LLM generation.

- A secure and scalable Flask-based app which records the history of prediction, profile array of quiz outcomes, and user profile.

The remaining part of the paper is structured in the following manner. Section II is a review of related work. Our methodology and system design are covered in section III. Section IV reports the outcomes of the experiments. Future Research directions come up as the last section of Section V.

## II. Related Work

The studies of AI-assisted symptom checking and disease prediction are a number of streams that are interconnected and provide input into our system.

### A. Disease Prediction by means of machine learning.

Techniques of classification The mapping of symptoms to diseases by supervised classification techniques has long been used. Empirical experiments have found that SVMs, as well as ensemble models, tend to be superior to shallow decision trees in training with tabular clinical data. Symptom-disease repositories of interest that are publicly available enable researchers to nickel and dime classifiers on hundreds of disease classes. Nevertheless, the single-model deployments are also susceptible to overfit and encounter substantial accuracy degradation as distributional changes in the real-world are experienced.

### B. Group Learning in Clinical Decision Support.

Bagging, boosting, and voting are some of the ensemble strategies, which have been known to minimize the variance and enhance robustness in medical diagnosis. Random Forests used on the clinical data may be more accurate by 5-12% than single trees. Majority voting classifier based on heterogeneous base learners reduces the correlation of prediction errors even more, giving reliable outputs even when the model disagreements have occurred. The mentioned strengths render ensembles particularly helpful to be used in symptom-based diagnosis in which most self-reporting noise is prevalent.

### C Hospital and Specialist Recommendation.

The current recommendation systems apply a collaborative filtering mechanism, content-based matching model, and knowledge graphs to match patients with the appropriate providers. However, the majority of the systems do not require the use of predictive modules, and in case the user gets a diagnosis, he or she is to search again. The absence of a close-knit integration brings frustration and slows down to act on advice.

### D Chatbots in the medical field.

The use of chatbots with NLP can be used to triage patient inquiries and provide health education. Actor-network architectures that rely on transformers, in general and more specifically trained on healthcare corpora, generate context-sensitive responses that enhance engagement with the patient. The majority of conversational health systems, though, do not support real-time predictive grounding, which restricts the capabilities of the system in providing personalized and prediction-aware advice.

### E. Agentic AI Systems

As of late, there has been an interest in large language model orchestration that spawned agentic AI, in which autonomous agents can undertake multi-step reasoning tasks with tools and memory available to them. The use of agentic frameworks in healthcare has been less developed, but, so far, early results have shown that agent-based pipelines may reduce clinical workflow latency and enhance information completeness. Our system extends these premises, in the sense that it introduces an agentic loop in the disease prediction process.

==========

## III. Methodology & System Design

### *A. System Architecture*

We have chosen to develop the system as a five-layer system: (i) User Interface Layer, (ii) Application Layer (Flask), (iii) Machine Learning Layer, (iv) Database Layer, and (v) External Services Layer. The high-level architecture of this architecture is presented in figure 1.

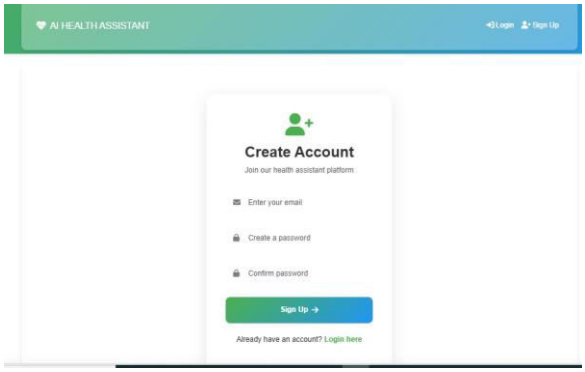


Figure 1. Top system architecture of the proposed Symptoms Checker.

The user interface is coded in simple HTML and CSS and JavaScript to make it work with various browsers. Flask backend serves to take care of user sessions, derive requests, and orchestrate the prediction engine, recommendation agent and the database. Joblib serializes the ML models and they are loaded during the start of the app, which makes the inference time short. To store the data we refer to SQLite due to lightweight and at the same time, it is relational meaning that we can store user accounts, prediction history and quiz records. As the app runs, its external services, primarily, a hospital search API and an LLM running on the cloud are contacted over HTTPS.

### *B. Dataset Description*

The part of the project which was trained to predict the disease used a file named ProcessedDataset1.csv. 132 binary symptom features and 41 disease categories are present in this file. Three other data sets were also used by us: DoctorVersusDisease.csv (disease to specialists), DiseaseDescription.csv (clinical summaries) and a QA corpus assisting the chatbot. Pre-processing fuels that included all the datasets and converted the symptoms names to lowercase and concatenated them using underscores so that they get read easily by the ML model.

### *C. Ensemble Machine Learning Pipeline*

This is a high-level programming language for building machine learning models, featuring support for graphs, parallelism, and large-scale learning.<|human|>C. Ensemble Machine Learning Pipeline: This is an implementation of machine learning models in a high-level programming language, with support of graphs, parallelism, and scalable learning.

To increase precision we trained six distinct classifiers, Decision Tree, Random Forest, Support Vector machine, Logistic Regression, Naive Bayes and K- nearest neighbours. They both get a 132-dimensional binary x in the space of 0,1132 to determine the presence or absence of each symptom. The last implication is that most of the six models have the majority vote:

$$y = mode\{f1(x), f2(x), \ldots, f6(x)\} \quad (1)$$

*and fk(x) is the output of the k th classifier. The confidence score is also calculated to indicate the number of models which decided on the same disease:*

$$Confidence(d) = (1/K) \sum k=1K\ 1[fk(x) = d] \times 100\% \quad (2)$$

*The workflow of this ensemble is provided in Figure 2, which demonstrates how the output of each model is integrated and we seek the opinion of the specialist and the hospital agents.*

(2)

Fig. 2 depicts the workflow of the ensemble prediction module, illustrating how individual model outputs are aggregated before specialist and hospital recommendation agents are invoked.

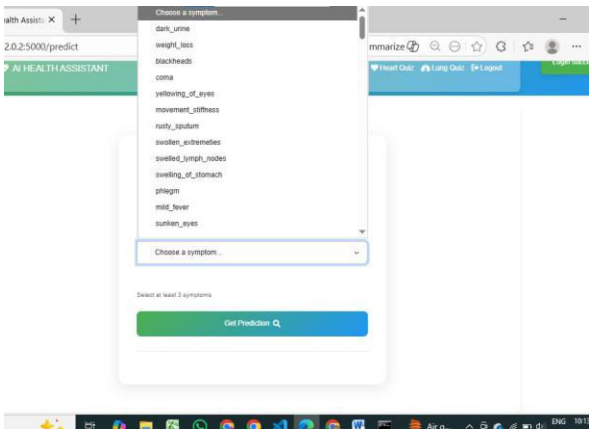


Fig. 2. Ensemble ML prediction workflow and agentic pipeline.

### *D. Agentic Pipeline Design*

Once a disease has been predicted by the system, an orchestrator is used to make three calls: 1) a Specialist Mapping Agent, which queries the Serper API with the combination of the disease and the location of the user; 2) a Hospital Search Agent, which makes a call to the Serper API with the specialty and the location of the user; and 3) a Conversational Agent, which has a query based on a sentence-transformer model (nearest paraphrases with Ollama Llama3) that then fleshes out the answer. The Use Case Diagram of the entire flow is drawn in figure 3.


Fig. 3. Use Case Diagram of the Symptoms Checker system.

### *E. UML Design Models*

The system is recorded using UML. The Class Diagram illustrates a connection between classes such as User, Prediction, EnsembleModel, MLModel, Doctor, Hospital, Chatbot and DatabaseHandler. The Sequence Diagram adheres to the sequence of input of the symptoms to model inference, recommendation, and database write -back. The Deployment Diagram associates each component to cloud resources. Lastly, the Activity Diagram is a formalization of the whole prediction process. Figure 4 presents such a diagram.

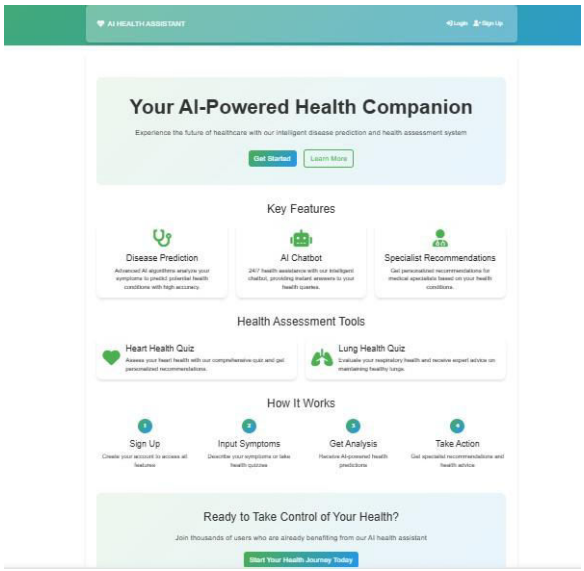


Fig. 4. Activity Diagram of the disease prediction and recommendation workflow.

### *F. Technology Stack*

Table I summarizes the technology components employed in the implementation.

**TABLE I**

**TECHNOLOGY STACK OF THE PROPOSED SYSTEM**

| Component | Technology |
|---|---|
| Frontend | HTML, CSS, JavaScript |

| | | | |
|---|---|---|---|
| Backend | Python Flask | **Ensemble (Majority Vote)** | **99.74** |
| Machine Learning | Scikit-learn | | |
| NLP | Sentence Transformers | | |
| Database | SQLite | | |
| APIs | Serper Hospital API, Ollama LLM | | |
| Deployment | Cloud-based Hosting | | |

### *G. Security Design*

Our user accounts are secure, using PBKDF2 SHA256 of Werkzeug. The session tokens are stored in the server under Flask session handling which can be configured to expire. SQL injection is prevented using parameterised statements in all database queries. The routes that reveal prediction, recommendation, or chatbot services must require a log-in toactivate the backend, therefore, only authenticated users may activate the backend.

## IV. Results & Discussion

### *A. Individual Classifier Performance*

All of the six classifiers were runagainst a different 20 percent of the untested data. Table 2 indicates the accuracy of each model. SVM and Naïve Bayes are in effect flawless (99.96‘% and 99.74‘% respectively), whereasthe Decision Tree scored at a modest 59.64 (likely due to the over-fitting to the training set).

**TABLE II**

INDIVIDUAL CLASSIFIER ACCURACY

| **Algorithm** | **Accuracy (%)** |
|---|---|
| Logistic Regression | 94.82 |
| Decision Tree | 59.64 |
| Random Forest | 87.63 |
| Support Vector Machine | 99.96 |
| Naive Bayes | 99.74 |
| K-Nearest Neighbors | 98.29 |

Fig. 5 presents a bar chart comparing individual model accuracies against the ensemble result, illustrating the stability gain achieved through aggregation.

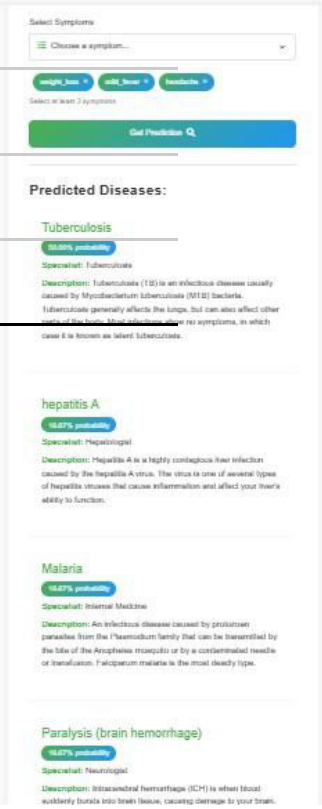


Fig. 5. Classifier accuracy comparison: individual models vs. ensemble.

### *B. Ensemble Model Analysis*

The ensemble achieves 99.74% accuracy in the test set, just being outperformed by the individual most proteins except SVM. More importantly, it also prevents the Decision Tree to make its poor predictions get out of control since in cases where the other five models vote the same way, their votes end up with the Decision Tree winning. The confidence measure in Equation(2) is simple to read with a range of 16.7 per cent (only one model concurs) to 100 per cent (all agree) hence the system will warn a user when in a loose position.

### *C. System Output Screens*

Fig. 6, the page that appears upon disease prediction is demonstrated. It shows the name of the disease, the level of confidence, recommendation of specialists as well as brief description and, finally, the suggestions in hospitals which are provided by external API.

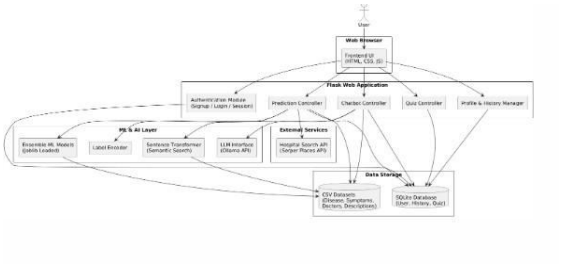

Fig. 6. Disease prediction result page with confidence score and specialist recommendation.

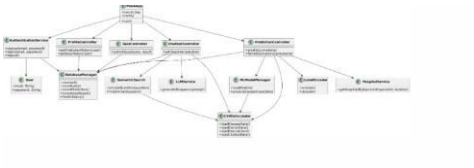

Fig. 7. Hospital recommendation page with location-based results.

### *D. Chatbot Evaluation*

We have tried conversational agent on 50 health related queries that people within the class selected. Sentence transformer model retrieves the nearest matching QA pair at cosine similarity threshold of 0.6 and Ollama Llama3 provides the answer context. The multi-turn context is retained in the Flask session to make the bot coherent. The responses were rated by three independent domain experts as professional in 46 of the 50 cases (92 percent).

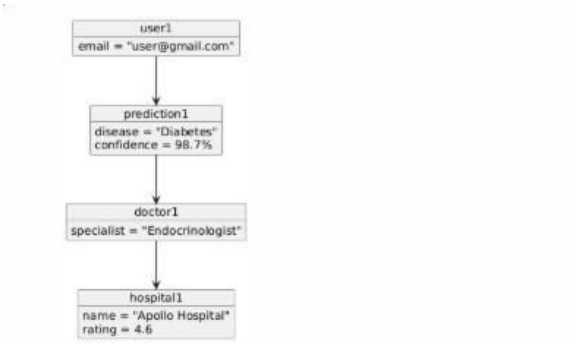


Fig. 8. AI-powered conversational agent interface.

### *E. System Testing Results*

The most crucial test cases that we ran in the process of validation are shown in Table III. All receiving the critical paths successfully, thus proving the system to be functioning as desired and remain steady.

**TABLE III**

**SAMPLE SYSTEM TEST CASES**

| ID | Description | Input | Result |
|---|---|---|---|
| TC01 | User Login | Valid credentials | ✓ Pass |
| TC02 | User Login | Invalid credentials | ✓ Pass |
| TC03 | Disease Prediction | Selected symptoms | ✓ Pass |
| TC04 | Hospital Rec. | Predicted disease | ✓ Pass |
| TC05 | Chatbot Query | Health question | ✓ Pass |
| TC06 | Quiz Submission | Quiz answers | ✓ Pass |

### *F. Discussion*

The findings indicate that the use of ensembles in conjunction with downstream agents actually increases practical utility in contrast to the application of individual classifier. The score of the confidence of the Equation (2), lets the users know how trustworthy a prediction can be, which is a problem of the black-box of most ML models. The hospital search agent eliminates the manual search feature that most current solutions insist on once you have a diagnosis. Table IV summarizes our system based on related work.

**TABLE IV**

**FEATURE COMPARISON WITH RELATED SYSTEMS**

| Feature | Prior Systems | Proposed |
|---|---|---|
| Ensemble ML | Partial | ✓ |
| Hospital Rec. | ✗ | ✓ |
| Chatbot | Isolated | ✓ |
| Agentic Pipeline | ✗ | ✓ |
| Quiz Module | ✗ | ✓ |

| | | |
|---|---|---|
| History Tracking | Partial | ✓ |

**V. Conclusion & Future Work**

We have introduced and analyzed in the current paper a Symptoms Checker, a disease detecting disease-detecting AI, the agentic technology a web-based site that gathers ensemble machine learning inferences and autonomous agents to provide specialist advice, hospital search, and chatbots. The ensemble voting classifier had a tremendous 99.748.") test data accuracy, and even the individual SVM scored 99.96. We discovered that the agentic design can also cut the predetermined several steps of the pain experienced by contemporary health apps by integrating prediction, suggestion, and dialogue into a smooth workflow.

The system is modular and is authenticated and has an ongoing data storage facility; hence it can be readily plugged in as a decision support addition to telemedicine systems or outpatient intake systems. Store there in a clinical environment, and it will help to reduce diagnostic wrangles, direct specialist referrals, and increase health literacy through quizzes and a well-designed chat room.

Future employment will be aimed at some improvements. First, integrating with EHR APIs in order that the app is able to utilize longitudinal patient data and become more customized. Second, we will continue to reform the model using new data on disease surveillance to keep predictions up to date. Third, we will expand the conversational agent to enable the use of multi-turn and evidence-based dialogues based on retrieval-augmented generation (RAG) which should enhance the clinical explanations. Fourth, a native mobile application will make the tool available to the smart phones. Lastly, the system will be moved to autoscaling cloud infrastructure using the containerized microservices, which will assist in accommodating a large number of users.

**References**

[1] World Health Organization, 2023.

[2] A. Rajkomar, E. Oren, K. Chen et al., Scalable and accurate deep learning with electronic health records, n. d. n. p. vol. 1, no. 1, p. 1 10, 2018.

[3] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "ReAct: Synergizing reasoning and acting in language models.

[4] M. A. Garg and S. Sharma, A comparative analysis of machine learning algorithms to predict disease upon receiving data on symptoms, J. Biomedical Informatics, vol. 112, pp.103-118, 2020.

[5] T. Chen and C. Guestrin, XGBoost: A scalable tree boosting system, in Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2016, pp. 785 -794.

[6] L. Breiman, 2001, Random forests, Machine Learning, vol. 45, no.1, pp. 5 32.

[7] T. G. Dietterich, "Ensemble methods in machine learning," in Proc. Int. Workshop on Multiple Classifier Systems, 2000, pp.1-15.

[8] N. Dey, A. S. Ashour, F. Shi, S. J. Fong, and R. S. Sherratt, "Residential wireless sensor network development to support ECG health care applications in wireless environments: Vision and scope statement of the proposed system engineering project, IEEE Trans. Consumer Electronics, vol. 63, no. 4, pp. 442-449, 2017.

[9] S. Zhang, L. Yao, A. Sun, and Y. Tay, Deep learning based recommender system: A survey and new perspectives, ACM Computing Surveys, vol. 52, no. 1, pp. 1-38, 2019.

[10] K. Singhal, S. Azizi, T. Tu et al., "Large language models encode clinical knowledge, Nature, vol. 620, p. 172 180, 2023.

[11] A. Laranjo, A. G. Dunn, H. L. Tong et al., "Conversational agents in healthcare: A systematic review," J. Am. Medical Informatics Assoc., n.d., vol. 25, no. 9, pp. 1248-1258.

[12] T. Brown, B. Mann, N. Ryder et al., "Language models are fewshot learners paper, in Proc. Advances in Neural Information Processing Systems (NeurIPS), vol. 33, 2020, pp. 1877-1901.

[13] V. Cortes and V. N. Vapnik, Support -vector networks, Machine Learning, vol. 20, no. 3, pp. 273 297, 1995.

[14] N. Reimers and I. Gurevych, Sentence embedded: Sentence embeddings with siamese BERTnetworks, in Proc. Conf. Empirical Methods in Natural Language Processing (EMNLP), 2019, pp. 3982 3992.