



www.hexart.in

A Disease Prediction
Chatbot for rural areas

Ankush Thalla
Kartikeya Singh Sisodia
Tejavath Vamsi Krishna
Manikanta Karneshwar

Team - Crocodile Squad

HEXART.IN INNOVATION COHORT PROGRAM

Offline Disease Prediction Chatbot Using Deep Learning

ABSTRACT

This project addresses these challenges by developing an **offline disease prediction chatbot** powered by deep learning.

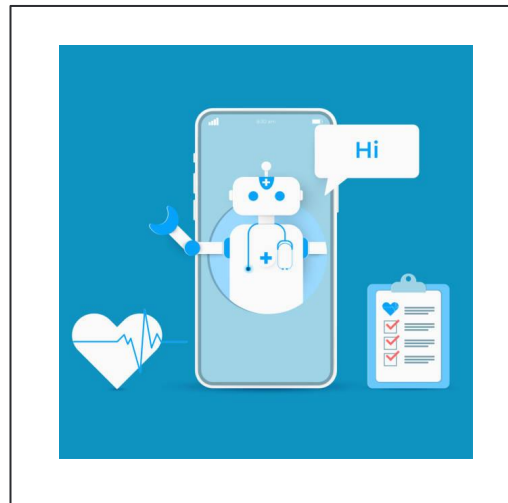
- The system utilizes a self-curated dataset, described in layman's terms to ensure accessibility. Employing feature extraction techniques, the project achieves high accuracy using a **Feedforward Neural Network (FNN)**, which matches the performance of complex models like BERT.
- The system predicts diseases based on user-input symptoms and facilitates **follow-up questions** for refined predictions, ultimately providing detailed explanations, treatment recommendations, and advice.
- By bridging the gap between healthcare and underserved communities, this project demonstrates the potential of AI to serve as a pre-diagnostic tool, **reducing dependency on internet access and costly medical consultations**.

To develop an
application that
helps initial
healthcare
diagnostics reach
rural areas with
remote
connectivity.

Mission

INTRODUCTION

- Access to healthcare in remote areas is often limited, delaying diagnosis and treatment. Existing models like BERT offer high accuracy but require substantial resources, making them unsuitable for offline use in low-resource settings.
- This project introduces a lightweight, offline-capable disease prediction model using Feedforward Neural Networks (FNN). Trained on a custom dataset, the model processes symptoms described in plain language, achieving 98% accuracy with minimal memory requirements (500 MB).
- This ensures accessible and efficient healthcare support in underserved regions.



LITERATURE SURVEY

Study	Proposed Model	Domain	Datasets Used	Key Results
Liu et al. (2024)	MCN-BERT	Disease Prediction	Medical Symptom Descriptions	MCN-BERT achieved 99.58% accuracy, outperforming BiLSTM (97.08%)
Nguyen et al.	Relation BioBERT + BLSTM	Drug-Drug Interaction (DDI)	TAC 2018, TAC 2019, SemEval 2013	Achieved F1-macro scores: 60.53% (TAC 2018), 80.23% (TAC 2019), 83.32% (SemEval 2013)
Avvaru et al.	Fine-tuned BERT + Multi-Layer Perceptron (MLP)	Cardiovascular Disease (CVD)	70,000 patient records with 11 features	BERT achieved 97.6% accuracy, outperforming traditional models (92%)

EXISTING METHODOLOGY

The existing methodologies use several advanced approaches to disease prediction and biomedical applications. These systems often rely on sophisticated deep learning models such as BERT and its variants, along with other neural network architectures.

Common Characteristics of Existing Systems:

Strengths:

- Use of transformer-based architectures like BERT and its specialized variants for high accuracy in biomedical applications.
- Enhanced capabilities through multi-channel networks, attention mechanisms, and fine-tuning on domain-specific datasets.

Challenges:

- Resource-intensive models with large memory and computational requirements.
- Limited applicability in offline or low-resource settings due to the high storage demands (e.g., 1.5–3 GB for BERT) and dependency on internet connectivity for certain applications.

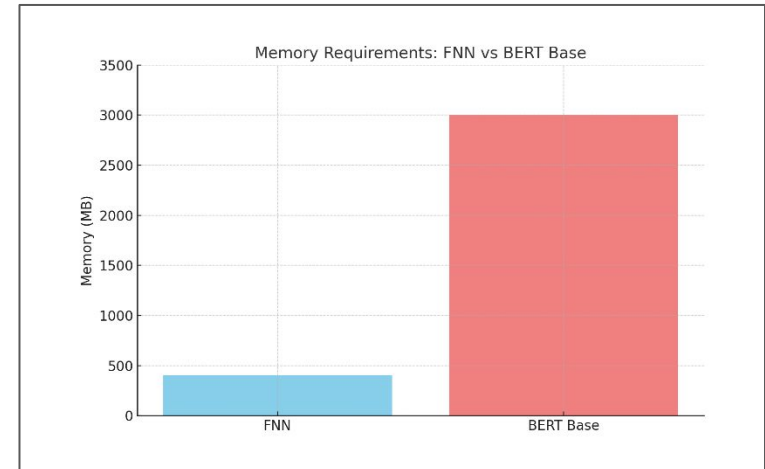
PROPOSED SYSTEM

Lightweight FNN Model: Utilizes a simple Feedforward Neural Network (FNN) to significantly reduce memory and computational demands, making it ideal for rural and low-resource areas.

Follow-Up Question Chatbot: Integrates an interactive chatbot that asks follow-up questions to refine the FNN's initial predictions.

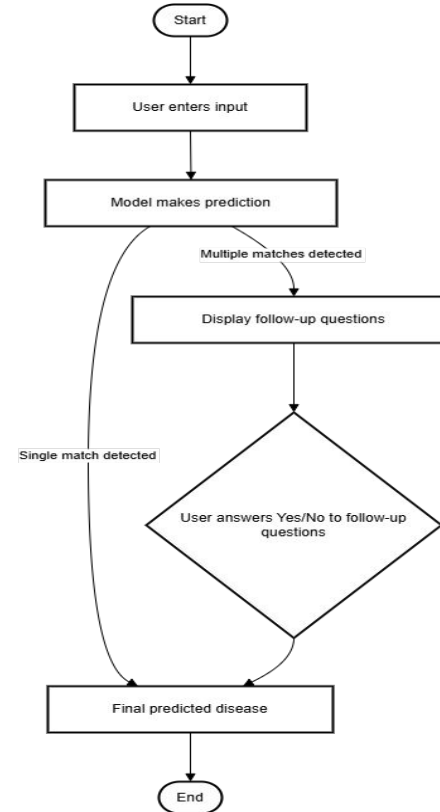
Progressive Disease Prediction: The FNN generates a list of top probable diseases, which is further narrowed down through user responses to follow-up questions.

Accuracy Comparable to Transformers: This dynamic approach boosts the FNN's accuracy to match that of complex models like BERT while maintaining efficiency.



WORKFLOW

- 1. User Input:** User enters symptoms in simple, everyday language
- 2. Initial Prediction (FNN):** The FNN model processes the input and generates a ranked list of the top probable diseases.
- 3. Follow-Up Questioning:** The chatbot asks targeted follow-up questions based on the initial prediction to gather more specific symptom details.
- 4. Refinement of Predictions:** User responses help refine and reorder the disease list, narrowing it down to the most likely diagnosis.
- 5. Final Diagnosis:** The system presents the most probable disease along with relevant information and recommended next steps.
- 6. Offline Accessibility:** All processing occurs offline, ensuring functionality in low-connectivity and resource-constrained environments.



WORKFLOW

Design of the Follow-Up Question System

The follow-up question system is designed to closely mimic a real doctor's diagnostic process by focusing on differentiating symptoms—the key symptoms that help distinguish between diseases with similar presentations.

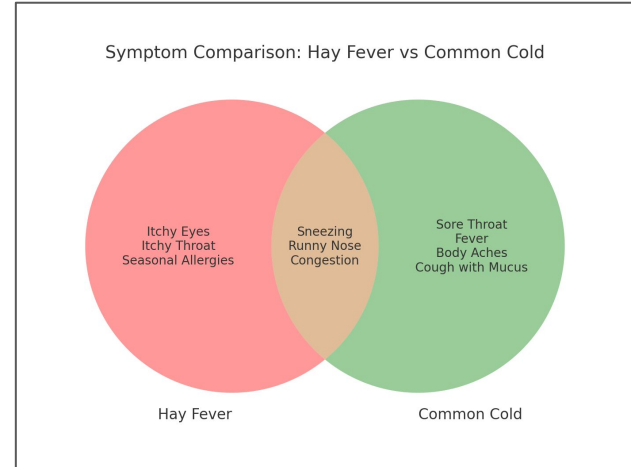
Symptom Categorization: Diseases are analyzed and categorized based on their common and unique (differentiating) symptoms.

Targeted Questioning: After the FNN provides an initial list of probable diseases, the chatbot generates follow-up questions that focus on differentiating symptoms to narrow down the diagnosis.

Dynamic Prediction Refinement: User responses are used to dynamically reorder and refine the list of probable diseases, improving prediction accuracy with each interaction.

Doctor-Like Diagnosis Flow: This iterative questioning process simulates how doctors progressively rule out conditions by asking specific, symptom-based questions.

This approach ensures that even with a lightweight FNN model, the system can deliver accurate and reliable predictions comparable to complex models like BERT.



TECHNICAL STACK

1. **Programming Languages:** Python
2. **Libraries and Frameworks:**
 - **TensorFlow:** An open-source deep learning framework for building and training machine learning models.
 - **TensorFlow Lite:** A lightweight version of TensorFlow designed for deploying machine learning models on mobile and edge devices.
 - **scikit-learn:** For machine learning algorithms, specifically Random Forest classifier.
 - **NumPy:** For numerical computations and array manipulation.
 - **Pickle:** For saving and loading datasets and trained models.

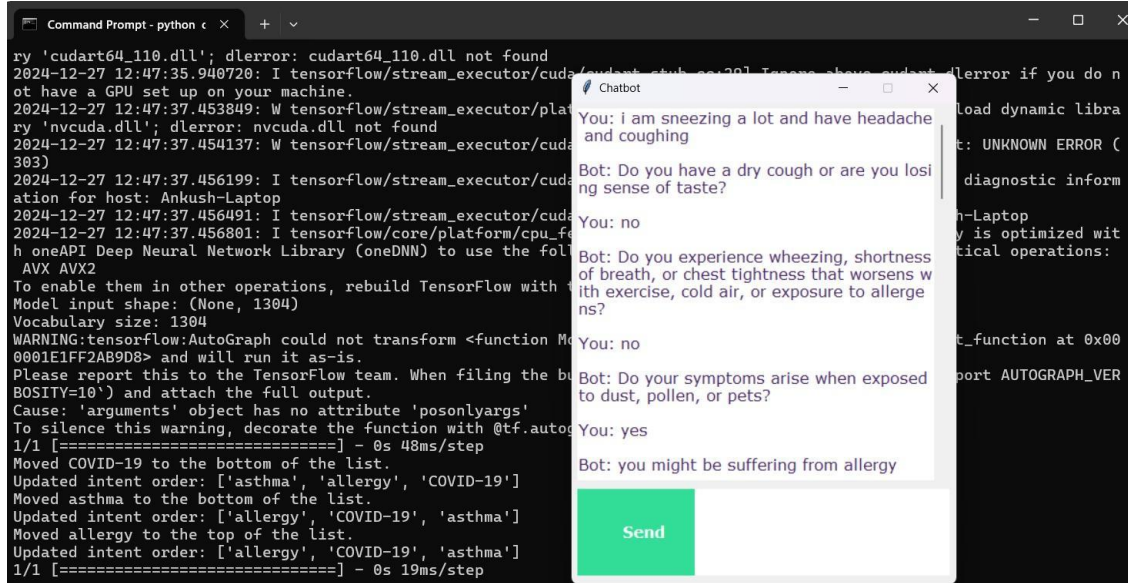
FUTURE SCOPE

- **Interactive Follow-up System:** Enhance accuracy with follow-up questions for better disease prediction.
- **Dataset Expansion:** Add more diseases and symptom patterns for broader coverage.
- **Multilingual Support:** Implement regional languages for better accessibility in rural areas.
- **Offline & Edge Deployment:** Optimize for low-memory devices to ensure offline functionality.
- **Wearable Integration:** Use real-time data from wearables for proactive health monitoring.
- **AI-Based Guidance:** Provide basic treatment advice and next steps before professional care.
- **Data Privacy:** Strengthen security for safe and ethical health data handling.

CONCLUSION

- The goal was to develop a lightweight, efficient model for resource-constrained, offline environments while maintaining high accuracy.
- Despite its simplicity, FNN, using a Bag-of-Words (BoW) method, achieved 99.02% accuracy, comparable to the more complex models. This success was due to the careful dataset design, which allowed FNN to effectively recognize key words for disease classification. The dataset's structure ensured that even a lightweight model could perform well.
- This study shows that simpler models like FNN, when combined with well-designed datasets, can rival complex models like BERT in resource-limited settings.
- The project has a greater potential for broader impact through integration into mobile apps, promoting inclusivity.

OUTCOMES

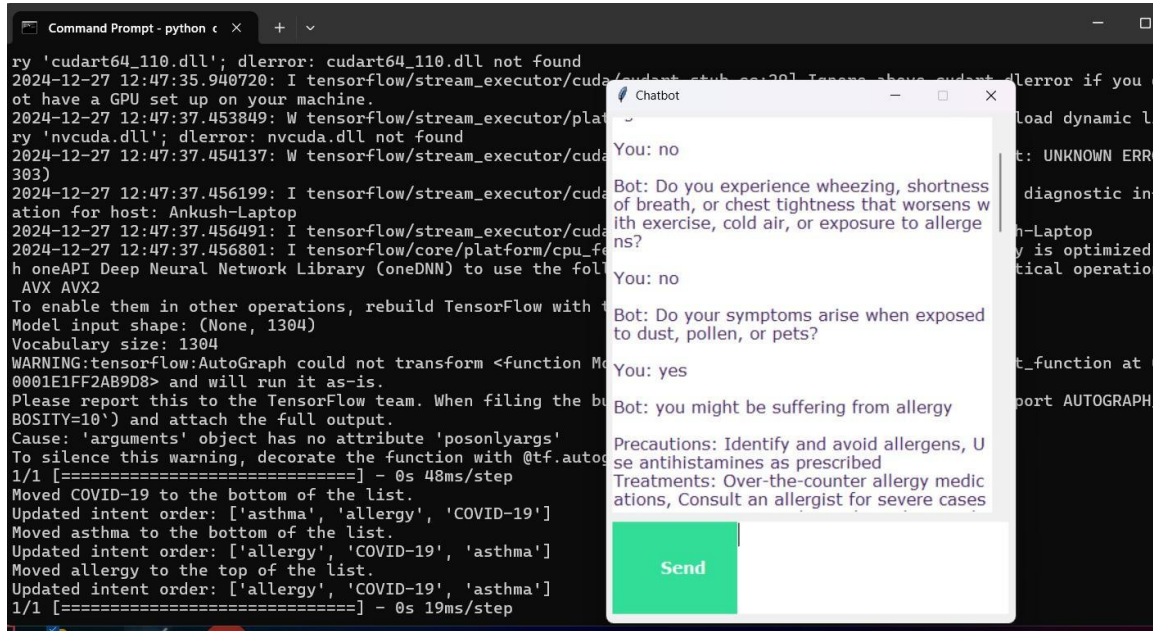


The screenshot shows a Windows Command Prompt window with a dark background. The title bar reads "Command Prompt - python". The terminal output displays several error messages related to missing CUDA and cuDNN libraries, followed by TensorFlow warnings and a diagnostic report for a host named "Ankush-Laptop". The report includes model input shape, vocabulary size, and a list of symptoms: "asthma", "allergy", "COVID-19", and "asthma". A chatbot interface is overlaid on the terminal, showing a conversation where the user describes symptoms (sneezing, headache, coughing) and the chatbot asks follow-up questions about cough, breathing, and exposure to allergens. The chatbot concludes with the suggestion: "you might be suffering from allergy". A green "Send" button is visible at the bottom of the chatbot window.

```
Command Prompt - python
ry 'cudart64_110.dll'; dlderror: cudart64_110.dll not found
2024-12-27 12:47:35.940720: I tensorflow/stream_executor/cuda/cuda_init.cc:281 TensorBoard: cudart64_110.dll not found if you do not have a GPU set up on your machine.
2024-12-27 12:47:37.453849: W tensorflow/stream_executor/platform/default/dso_loader.cc:54 Could not load dynamic library 'nvcuda.dll'; dlderror: nvcuda.dll not found
2024-12-27 12:47:37.454137: W tensorflow/stream_executor/cuda/cuda_init.cc:303)
2024-12-27 12:47:37.456199: I tensorflow/stream_executor/cuda/cuda_init.cc:303)
2024-12-27 12:47:37.456491: I tensorflow/stream_executor/cuda/cuda_init.cc:303)
2024-12-27 12:47:37.456801: I tensorflow/core/platform/cpu_feature_guard.cc:151 The processor supports AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
Model input shape: (None, 1304)
Vocabulary size: 1304
WARNING:tensorflow:AutoGraph could not transform <function Model.compile_tf_function at 0x000001E1FF2A89D8> and will run it as-is.
Please report this to the TensorFlow team. When filing the bug, set the log level to 'BOSITY=10') and attach the full output.
Cause: 'arguments' object has no attribute 'posonlyargs'
To silence this warning, decorate the function with @tf.autograph.experimental.do_not_convert
1/1 [=====] - 0s 48ms/step
Moved COVID-19 to the bottom of the list.
Updated intent order: ['asthma', 'allergy', 'COVID-19']
Moved asthma to the bottom of the list.
Updated intent order: ['allergy', 'COVID-19', 'asthma']
Moved allergy to the top of the list.
Updated intent order: ['allergy', 'COVID-19', 'asthma']
1/1 [=====] - 0s 19ms/step
```

This is the user input followed by the chatbot identifying disease

OUTCOMES



The screenshot shows a Windows Command Prompt window titled "Command Prompt - python" with a dark background. The terminal displays several error messages related to TensorFlow's GPU support, including "dlerror: cudart64_110.dll not found" and "dlerror: nvcuda.dll not found". It also shows TensorFlow's internal logs, such as "I tensorflow/stream_executor/cuda/cuda_device_stub.py:201] TensorBoard shows cuda0 is not available" and "W tensorflow/stream_executor/platform/default/dso_loader.cc:45] Warning: dso_loader failed to load dynamic library 'C:\Program Files\NVIDIA Corporation\NVidia\NVIDIA-RTX-Shader-Compiler\bin\nvcompiler.dll'".

Overlaid on the terminal is a white chatbot window titled "Chatbot". The chat history shows a user asking "You: no" in response to a question about wheezing, shortness of breath, or chest tightness. The chatbot responds with "Bot: Do your symptoms arise when exposed to dust, pollen, or pets?". The user again responds "You: no". The chatbot then asks "You: yes" in response to a question about symptoms arising when exposed to dust, pollen, or pets. The chatbot's final response is "Bot: you might be suffering from allergy". Below the chat history, there is a green button labeled "Send".

This is the the chatbot recommending treatments and precautions after identification of disease

OUTCOMES



Thank You

Hexart.In...an opportunity like never before!



Sponsor



Implementation Partner



CSR Partner