Cover Page for High-Risk Project

AI in Healthcare

Joseph Barnes

Michael Chang

Project Links:

* PowerPoint Presentation: <https://docs.google.com/presentation/d/1_DVJ4zLIkEf2vmcAsuuPHxdOMS3SJzte/edit?usp=sharing&ouid=112344627999249666156&rtpof=true&sd=true>
* Video Presentation: <https://utexas.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=3ec7ac81-5d26-4cf5-8161-b1620033765e>
* Code Repository: <https://github.com/JoeyBarnes/autogen-ai-healthcare>
* Sample Code Outputs:
  + AKI (MIMIC 3): <https://github.com/JoeyBarnes/autogen-ai-healthcare/blob/main/sample_outputs/mimic3_aki/autogen/dialog.md>
  + Sepsis (MIMIC 4): <https://github.com/JoeyBarnes/autogen-ai-healthcare/blob/main/sample_outputs/mimic4_sepsis/autogen/dialog.md>

Automating AI Model Development for Healthcare Using AutoGen: A Multi-Agent Conversation Framework

Automating AI Model Development for Healthcare Using AutoGen

Joseph A. Barnes, Master’s Candidate

University of Texas at Austin, [joseph.barnes@utexas.edu](mailto:joseph.barnes@utexas.edu)

Michael Chang, Master’s Candidate

University of Texas at Austin, [michaelchang@utexas.edu](mailto:michaelchang@utexas.edu)

Abstract

In healthcare research, it is often necessary to perform machine learning (ML) tasks such as data research, processing, feature engineering, analysis, model creation, training, evaluation, and visualization, to understand diseases and their behavior. To streamline and enhance the efficiency of these repetitive and tedious machine learning tasks, the project aims to create a low-code/no-code approach to AI model development for healthcare research. We propose the development of an application based on AutoGen, a multi-agent conversation framework developed by Microsoft. This application aims to autonomously handle various stages of the end-to-end machine learning pipeline by using natural language-style conversations from a user. Our objective is to accelerate the pace of healthcare research by automating and optimizing repetitive and time-consuming tasks. This will allow researchers to focus more on high-level decision-making and innovation.

CCS CONCEPTS

• Computing methodologies • Artificial Intelligence • Distributed artificial intelligence • Multi-agent systems

Additional Keywords and Phrases:

Healthcare research, Multi-Agent Conversation Framework, Machine Learning, Data Science, Large-Language Models

1. Introduction

In contemporary healthcare research, the demand for efficient data-driven methodologies has never been more pronounced. Healthcare researchers often need to perform machine learning algorithms and data science tasks such as data research, processing, model creation, evaluation, and visualization, to accurately understand the data they are working with. These technical tasks often surpass the expertise of healthcare specialists, aligning more closely with the domain of machine learning engineers or data scientists. Consequently, healthcare experts may encounter challenges in efficiently using data science techniques for their research objectives. There is thus a need to simplify and optimize these processes so that healthcare specialists can perform their research.

Our project aims to develop an application based on AutoGen, a multi-agent conversation framework developed by Microsoft, with the primary objective to automate and optimize ML and data science tasks that healthcare researchers often perform. Through this application, we aim to expedite and improve the efficiency of healthcare research endeavors.

This paper begins with our methodology where we explain our code logic, the multi-agent application configuration, and the prompts. Subsequently, we present the results of the program: the output files generated, the code synthesized by the agent, the tasks executed by the multi-agent application, the analysis conducted on the data, and the visualizations produced.

Finally, we discuss the implications and significance of the project within the realm of healthcare research, identify areas of improvement, and outline potential pathways for future research endeavors.

1. Related works

Our project is based on previous works related to Large Language Models (LLM) that leverage the Multi-Agent Conversation Framework AutoGen. In *AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversatio*n *Framework* [[1](#bib1)], Wu et al. presented the framework, developed by Microsoft, that enables the creation of multiple LLM agents with the capacity to interact with each other. To properly use the AutoGen framework for our project needs, we relied on Microsoft’s documentation and guides[[2](#bib2)]. Using their documentation, we created multiple conversational agents that allow users to use natural language-style conversations to generate complex code, execute the code, and create machine learning models for healthcare research.

1. Methodology
   1. Multi-Agent Conversation Framework

We utilized AutoGen to create multiple LLM agents capable of executing multiple data science tasks sequentially, supported by the GPT-4 model.  GPT-4 is OpenAI’s most powerful model to date. Paired with AutoGen, our application can generate high-quality responses and execute complex tasks efficiently.

* 1. Agent Initialization and Role Assignment

The agent is initialized with the appropriate credentials to access the necessary resources. Subsequently, we designated a series of roles for each agent, tailored to specific expertise areas. These roles include agents in the AssistantAgent subclass [[3](#bib3)]. We attributed the following names: "Healthcare Specialist", "Data Scientist", "AI Engineer", and "Visualization Expert". Additionally, we created a UserProxyAgent subclass instance named "User", representing the user's interactions with the application.

* 1. Task Assignment and Prompting

A series of tasks were defined and assigned to the respective roles based on their domain expertise. These tasks correspond to the different steps for machine learning model development. Each task contained detailed step-by-step prompts related to the desired output. These prompts use natural language instructions, enabling healthcare professionals with varying levels of technical expertise to engage effectively with the application. The steps included rely on MIMIC III and MIMIC IV medical data. Our application also includes code to help convert and support both MIMIC versions. The tasks included:

**Research Task:** Assigned to the "Healthcare Specialist" role, to retrieve lab tests relevant to the medical condition provided.

* **Processing / Filtering Task:** Assigned to the "Data Scientist" role, to process and filter the patient dataset based on the relevant lab tests found.
* **Labeling Task:** Assigned to the "Data Scientist" role, to create binary labels for designating the diagnosis of the medical condition.
* **Feature Engineering Task:** assigned to the "Data Scientist" role, to engineer features, based on the filtered lab tests, to be used in a machine learning model.
* **Dimensionality Reduction Task:** assigned to the "AI Engineer" role, to perform dimensionality reduction techniques such as Principal Component Analysis (PCA) on the patient data to reduce model complexity.
* **Model Training and Evaluation Task:** assigned to the "AI Engineer" role, to create and train machine learning models based on the patients' data, and to evaluate each model’s performance.
* **Visualization Task:** Assigned to the "Visualization Expert" role, to create visualizations with insights from the performance metrics and models’ results.
  1. Model Training

Using two medical conditions, Acute Kidney Injury and Sepsis, as the research topic for prompting, our application produced the desired files, executed the generated code to create, train, and evaluate multiple models, and created plots to visualize performance metrics ([Figure 1](#fig1)).

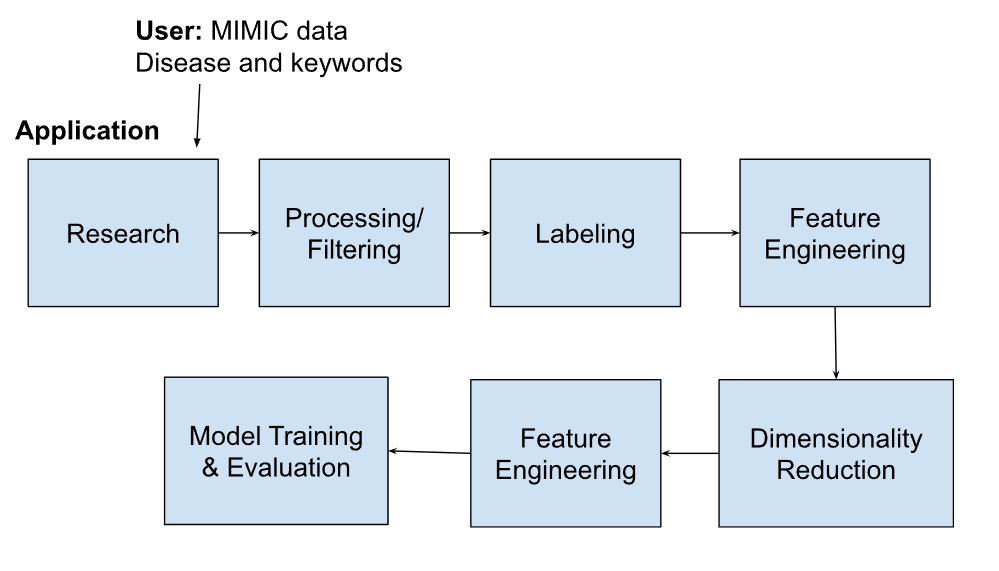


Figure 1: The sequence of tasks executed by the application.

1. Results

In the table below ([Table 1](#tb1)), are the generated Python code files and results files for each task. The results file contains the output of the executed code Python code.

Table 1: Results of each task in the conversation flow

| **Task** | **Code (.py)** | **Results** |
| --- | --- | --- |
| Research | Research.py | lab\_test\_types.json |
| Processing & Filtering | processing\_filtering.py | filtered\_patients\_labs.parquet |
| Labeling | labeling.py | patients\_labels.parquet |
| Feature Engineering | feature\_engineering.py | features\_labels.parquet |
| Dimension Reduction | dimensionality\_reduction.py | reduced\_features\_labels.parquet |
| Model Training and Evaluation | training\_evaluation.py | model\_details.json |
| Visualization | Visualization.py | classification\_report.png, roc\_curve.png, pr\_curve.png |

In the Research Task, the LLM output also provides detailed explanations of each relevant lab test. In the Dimension Reduction Task, LLM output explains the Principal Component Analysis (PCA) method and the relevance of the retained features of our model. In the Model Training Task, the models created are DecisionTree, RandomForest, LogisticRegression, GradientBoosting, MLP, and KNeighbors. The performance of each model is evaluated with a classification report, Receiver Operating Characteristic (ROC) curve, precision-recall curve, and a written report explaining the significance of each model’s performance.

In the Visualization task, the LLM created 3 plots: classification report ([Figure 2](#fig2)**)**, Receiver Operating Characteristic (ROC) curve ([Figure 3](#fig3)**)**, and Precision-Recall curve ([Figure 4](#fig4)**)**.

A graph of different colored bars


Figure 2: Classification report comparing performance metrics between models.

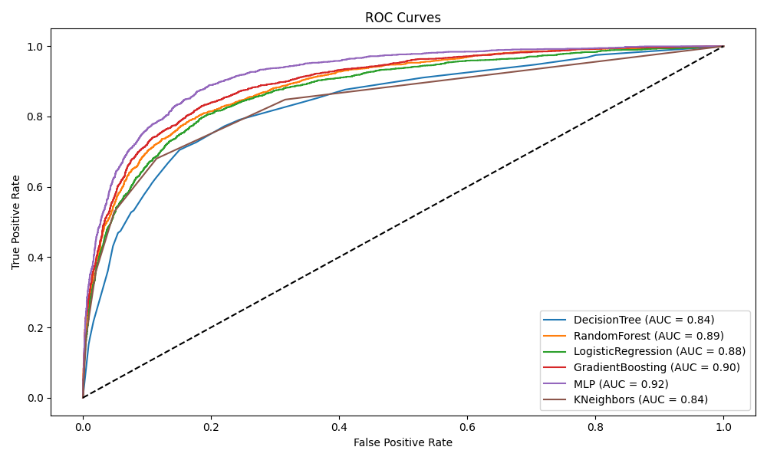


Figure 3: ROC curves comparing performance metrics between models.

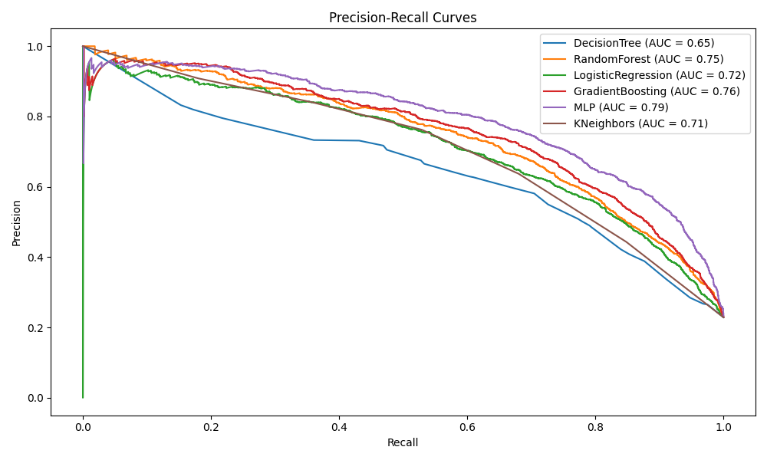


Figure 4: Precision-Recall curves comparing performance metrics between models.

1. Future Directions

Some areas of improvement include simplifying the prompts used for each agent as the current prompts are meticulously designed and complex in step-by-step instructions. For example, our application can use Teachability [[4](#bib4)],which allows agents to “remember” information from previous conversations. This can help reduce the detailed complexity of the prompts.  In addition, we can introduce LLM Reflexion to our application, which allows our agents to self-criticize and improve themselves [[5](#bib5)].

1. Conclusion

In conclusion, our research project has demonstrated the efficacy of a multi-agent application powered by AutoGen in creating machine learning models for healthcare research. Our application takes natural language style prompts from the user and successfully executes data science tasks to develop machine learning models. We provided two examples of how the application works for the following medical conditions: Acute Kidney Injury and Sepsis. After producing machine learning models, we evaluated their performance.

Moving forward, as LLMs continue to grow in data size and advance, so too does the potential of our application. Through accurate simple natural language prompt engineering and AutoGen, our application can greatly improve healthcare research. For more information related to this project, please visit the GitHub repo: <https://github.com/JoeyBarnes/autogen-ai-healthcare>

REFERENCES

<bib id="bib1"><number>[1]</number>Wu, Q., Bansal, G., Zhang, J., Wu, Y., Li, B., Zhu, E., Jiang, L., Zhang, X., Zhang, S., Liu, J., Awadallah, A. H., White, R. W., Burger, D., & Wang, C. (2023). AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation. *ArXiv*. /abs/2308.08155</bib>

*<bib id="bib2"><number>[2]</number>Getting started: Autogen*. AutoGen RSS. (n.d.). https://microsoft.github.io/autogen/docs/Getting-Started/ </bib>

*<bib id="bib3"><number>[3]</number>Multi-agent conversation framework: Autogen*. AutoGen RSS. (n.d.). https://microsoft.github.io/autogen/docs/Use-Cases/agent\_chat/ </bib>

<bib id="bib4"><number>[4]</number>Loynd, R. (2023a, October 26). *Autogen’s teachable agents: Autogen*. AutoGen RSS. https://microsoft.github.io/autogen/blog/2023/10/26/TeachableAgent/ </bib>

*<bib id="bib5"><number>[5]</number>Reflexion – nextra*. Prompt Engineering Guide. (n.d.). https://www.promptingguide.ai/techniques/reflexion</bib>