Grenze International Journal of Engineering and Technology, June Issue

AI-Driven Rail Madad Complaint Management System

Shubham Jadhav[1], Pranav Jadhav[2], Mooin Shikalgar[3], Suyash Haladkar[4], Pallavi Tekade[5] and Dipmala Salunke[6]
[1-6]Information Technology/ JSPM's Rajarshi Shahu College of Engineering, Tathawade, Pimpri-Chinchwad, Pune, India
Email: shubhamjadhav1202@gmail.com, pranavjadhav7179@gmail.com, mooinshikalgar299@gmail.com, suyashhaladkar007@gmail.com, pmtekade_it@jspmrscoe.edu.in, dtsalunke_it@jspmrscoe.edu.in

***Abstract*— Indian Railways have had a lot of trouble managing passenger complaints because they get so many and their automated systems are not good enough. This is a problem for them. The project I am talking about is called "Rail Madad". It is a complete solution to this problem. Rail Madad has a user interface that's easy to use and a secure backend. It also has two parts that use artificial intelligence: one that helps sort out complaints and a chatbot that answers user questions. We built Rail Madad using a few tools. We used Node.js and Express to build the parts of the system that different applications can talk to. We used MongoDB to store all the complaints. We used Python to build a classifier that uses machine learning to categorize the complaints. We tested Rail Madad using complaints to see how well it worked. We looked at how accurate it was how easy it was to use and how long it took to respond.**

The results show that Rail Madad is an improvement over the old manual ways of handling complaints. It helps Indian Railways prioritize and resolve complaints more efficiently. Rail Madad is an useful tool, for Indian Railways and it can help them manage passenger grievances much better. This research's instrumental work is (i) a complaint automated system for categorization, (ii) prioritization depending on the degree of severity, (iii) a chatbot passenger interaction facilitation, and (iv) an end-to-end frontend-backend-ML pipeline integration for railway grievance management.

***Index Terms*— Natural Language Processing (NLP), Chatbot, Artificial Intelligence (AI), Machine Learning, Text Analysis, Complaint Classification, Passenger Grievance Redressal, BERT.**

I. INTRODUCTION

On a daily basis, the Indian Railways network is flooded with more than a thousand complaints from passengers. These complaints cover various aspects such as ticketing, facilities available on the train, punctuality, and the behavior of the staff. The existing systems for addressing grievances are very much dependent on the manual handling and sorting of the complaints. Due to this method, it is often the case that there are considerable delays in the response times which in turn cause a large number of passengers to be disgruntled with the service.

As a solution to such inefficiencies, we come up with Rail Madad, a smart and automatic complaint management system. The main aim of this initiative is to upgrade in every way the slow, inaccurate, and opaque processes typically involved in the resolution of grievances. Complaints should be handled efficiently as they exercise a strong influence on passenger confidence, the stability of the railway system, and the general view of the railways as a provider of quality services. Moreover, the performance of operations through manual means not only requires a lot of resources, but it is also accompanied with a great possibility of missing heavy complaints, which in turn may result in safety issues or the loss of reputation.

Grenze ID: 01.GIJET.12.2.262_26

At the heart of the Rail Madad system are three technology pillars:

- The backend was built using Node.js and Express.
- MongoDB is used for storing data.

The frontend was developed with React.

- Tailwind CSS was used for styling to make it easy for users to interact with the system.

A machine learning classifier that runs on Python helps to categorize and prioritize user complaints.
This classifier is a part of the system.
The Rail Madad system also includes a chatbot.

- This chatbot provides real-time help to users.
- It makes it easier for users to access the system.

Current systems, such as the official Rail Madad app let users file complaints online.
However they do not have features like automatic complaint classification and chatbots.

- Our system fills this gap by providing an user-friendly platform.
- It uses data to make it more efficient.

To test the system efficiency we used a real-world dataset of complaints.
This will help us to see how well the system works in practice. The Rail Madad system is designed to be more than a complaint filing system. It aims to provide an experience, for users. The system uses machine learning and a chatbot to achieve this goal.
The goal of Rail Madad system is to provide a platform that is driven by data and user-friendly. We trained and tested the classifier to check its accuracy. Later, this classifier was merged with the backend pipeline for complete end-to-end system testing with the help of simulated complaints.
The experimental results demonstrate a significant enhancement in classification accuracy which has resulted in a faster complaint resolution time and a decrease in the total handling time as compared to the traditional manual method.
Basically, our research is a demonstration of the effectiveness of AI and NLP methods in scaling, making the process more efficient and increasing users' satisfaction in the passenger grievance management domain. Nevertheless, the continued success of the system will depend on the quality and variety of the data used for model training.

A. Proposed System

The system planned to be implemented is called Rail Madad, which is a device to automate the Indian Railways' complaint management process by using AI and ML technologies. Basically, the aim is to eliminate the defects of manual handling- the process that is very often time-consuming, inconsistent, and liable to human error. In this way, the system is performing the classification, prioritization, and routing of passenger complaints to the accurate departments automatically, thus making the turnaround of grievances faster as well as more accurate.
Passengers of the Rail Madad service have the choice of filing their grievances through an online web portal or a chatbot interface. Once the complaint is submitted, it is forwarded to the backend server, which performs multiple automated processing stages on it.
The backend, which is built with Node.js and Express connects the frontend interface with the machine learning module. MongoDB stores all complaint data and metadata securely providing scalability, reliability and efficient data access.
The user-facing frontend, built with React.js and styled by Tailwind CSS offers an responsive experience for users, who are passengers and staff who are admins.
There are two interfaces in this system.

- A customer portal that lets users lodge, check and follow their complaints instantly.
- An admin portal that provides a user- platform for railway employees to supervise, sort and handle grievances.

The Machine Learning module is at the heart of Rail Madads intelligence. It is written in Python. Uses the Flask framework. This ML unit communicates with the Node.js-based backend via API calls. When a complaint is received the backend sends a request to the ML server with the complaint text. The server applies -trained transformer models from Hugging Face for text classification and sentiment analysis.
The sentiment analysis model, built on DistilBERT identifies the complaints sentiment as positive, negative or neutral. A text classification model determines the complaint category, such as Food, Delay, Cleanliness, Facilities, Seat Issues, Safety or Ticketing. A rule-based keyword system determines the priority level as High, Medium or Low and the severity level as Critical, Moderate or Mild.
For example complaints about safety issues medical emergencies or staff misbehavior are marked as priority.

The Super Admin role in Rail Madads system architecture provides oversight, control and maintenance of system integrity. The Super Admin is in charge of category- admins ensuring complaint handling in each department meets the same standard.
The Super Admin can:

- Observe statistics in time.
- Check. Unresolved complaints.
- Regulate permission levels of lower-tier admins.

The Super Admin is responsible for observing activities handling system changes and ensuring data consistency across the platform. This control makes the system more transparent, efficient and less prone, to redundancies.
When the ML module finishes its analysis, it returns a structured JSON response that has the predicted sentiment, category, priority, and severity. The backend then stores this raw data in MongoDB and refreshes the admin dashboard. So, the railway staff get to view the complaint maps, sort them by urgency, and then intervene without delay in the case of critical issues. Such an automation system makes it possible to recognize and escalate high- priority complaints at the same time as they happen, thus, the response time is shortened to a minimum. The chatbot interface with natural language features is the next step to the traveller's delight. In fact, passengers can talk to the bot, tell their grievances, or know the status of their grievances without the hassle of opening several web pages.
There are many obvious advantages that the Rail Madad system has over the traditional manual methods. Among these, it can be highlighted that the time for complaint processing is significantly reduced, human intervention and error are minimized, and classification accuracy is improved by data-driven insights. Besides, the system's modular design is scalable, so it will not be a problem whether new complaint categories or departments will be added in the future. In addition, it helps to increase the openness and responsibility to the users by giving them the possibility of following their complaint's progress and receiving real-time updates from the authorities.

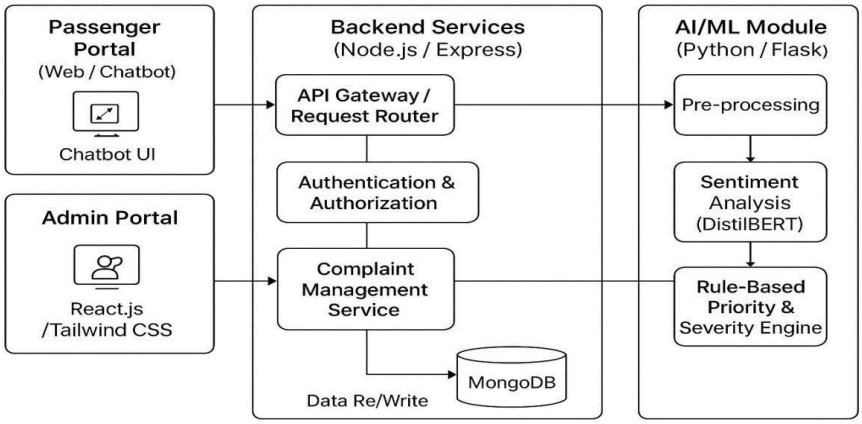

Fig 1. System Diagram

II. HELPFUL HINTS

A. System Interface Overview

The main entrance is a place where you can find out about the service. This service is a trusted platform where you can go to solve your problems. It is easy to use because it helps you decide what to do by clicking on either "Lodge Complaint" or "Track Existing Complaint". This page also gives you a help line number which's 139 and an email address so you can get the help you need. The platform is made to be simple so you can find help when you need it.
You can use this platform to lodge a complaint or track an existing complaint. The main entrance and the service are here to help you with the help line and email. The picture shows the Food Services Admin Dashboard of the Rail Madad system. This system is, for people who manage categories. It shows how complaints there are and what is happening with them. The Food Services Admin Dashboard of the Rail Madad system shows if complaints are waiting to be fixed being worked on or already fixed. It also shows how important each complaint is. The Food Services Admin Dashboard of the Rail Madad system is easy to use so admins can keep track of complaints. Make changes if needed.

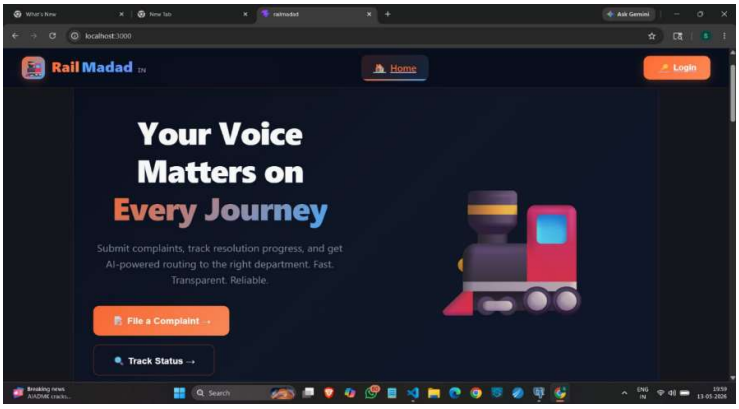

Figure 2. GUI Image (User Interface)

Each complaint has its information, like a special ID number, the PNR number, what is going on with the complaint how important the complaint is and when the complaint was sent in. The Food Services Admin Dashboard of the Rail Madad system also has a way to open or update each complaint. The Food Services Admin Dashboard can look at the complaint and make changes to the complaint.

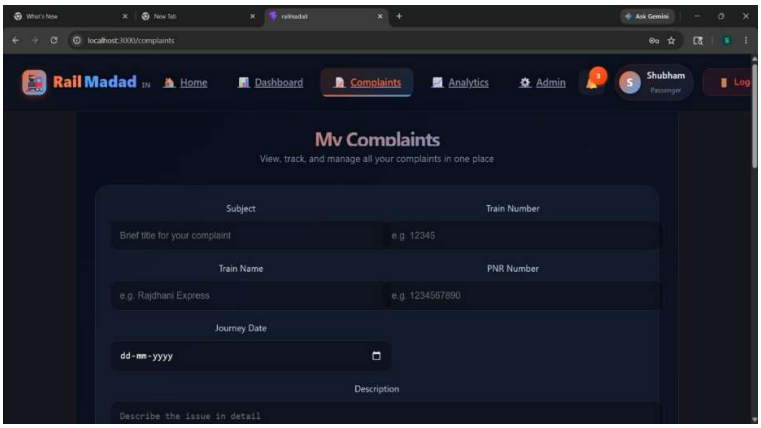

Figure 3. GUI Image (Complaint Details)

The picture shows the Complaint Details page from the Rail Madad Category Admin Dashboard. This page has lots of information about a complaint. You can see the complaint ID, the PNR number, what category it falls under how important it's a description of the problem and details, about the train. The admin can look at one complaint at a time. This helps the admin to solve the problem.

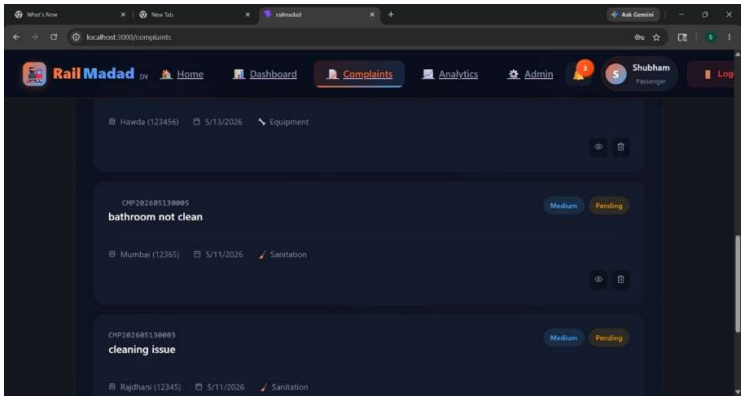

Figure 4. GUI Image (Registered Complaints)

B. Experimental Environment

The Rail Madad system went through design and testing in a kind of environment. This environment was set up on purpose to combine machine learning with a web-based system. As a result the Rail Madad system could work well with its computer resources. Do real-time model checks and regular web server tasks.

- The backend of the Rail Madad system was built using Node.js and Express.js.

This helped keep the client, database and machine learning module safe and working together. A MongoDB database was used to store user complaints, prediction results and system updates. This made it easy to manage data and add more as needed.

The frontend of the Rail Madad application was built using React.js and styled with Tailwind CSS. This made the application easy to use and responsive, for both users and admins. The chatbot module was connected to a Flask API to give context-aware replies. We used Postman API endpoints to test the Rail Madad system and make sure it works properly. This helped us see that data moves smoothly between the parts of the system.

We did all our development work in Visual Studio Code. The Rail Madad system and its machine learning parts were checked to make sure they work correctly. We stored data safely using MongoDB. We used React.js and Tailwind CSS to make the Rail Madad application easy for users to work with.When we tested the system to see how well it performs we found out that it can give a response to a complaint in 1-2 seconds. This is great because it means the Rail Madad system is very good at handling complaints. The system can also talk to users quickly almost like it is happening now. The Rail Madad system is very good, at classifying complaints. When we use the Rail Madad system to classify complaints it gives us a response fast usually in just 1-2 seconds.

C. Result and Discussion

The work we did to get machine learning models working with the Rail madad system was really successful. We made a system that helps manage complaints on the railway. This system makes it easy to look at complaints from passengers and sort them into groups like Cleanliness, Food, Delay, Safety and Ticketing. We also looked at how people felt when they made complaints. This helped officials figure out which complaints to solve the ones that made people really upset. It also helped them talk to the people who made complaints in a way.

We used two ways to sort railway complaints in the system. One way was based on rules and the other way used a kind of model called a transformer-based model, which is facebook/bart-large-mnli. When we tested the transformer-based model it was 88.4 percent of the time. This shows that it is really good at understanding what complaints mean and sorting them into the groups like Medical Emergency, Food Services and Safety. This means the system is great, for using away because it can understand what people mean and sort complaints in a way that is trustworthy. The Rail madad system is good to use. I think if we make some adjustments and use railway data it will get even better and more stable. The Rail madad system and machine learning models work well together to manage railway complaints.

The numbers show that the Random Forest model did a job classifying complaints correctly. The BERT-based model was behind. This is because the BERT-based model is really good at understanding complaint meanings. It took the system 1.8 seconds to process one complaint on average. So the system gives results in time. The people in charge loved the dynamic dashboard visualizations. These visualizations showed complaints, their severity and how people felt about them over time. This gave them an idea of how the system was doing. Helped them make decisions faster.

For example the system could detect when safety-related complaints were increasing and alert authorities. This allowed them to respond quickly. The systems ability to connect data visualization to decision-making is what makes it useful. Passengers and railway officials thought the Rail madad system was great at monitoring and prioritizing complaints.

However they also said it had some limitations. Sometimes the system got it wrong when it encountered texts with feelings or texts without enough context. Most users were happy with the system except for a small issues. The Rail madad system has made a difference in managing complaints. Has been very successful in terms of accuracy and speed. The Random Forest model and the BERT-based model are both parts of the Rail madad system. The Rail madad systems success is due, to its ability to process complaints quickly and accurately like the Random Forest model and the BERT-based model.

D. Equations

The Rail Madad system uses machine learning and natural language processing to classify and prioritize passenger complaints. It helps to automate this process. The system merges these technologies to improve passenger grievance handling. Rail Madad makes it easier to manage passenger issues. This system helps railway authorities to focus on solving problems. Rail Madad uses machine learning algorithms and natural language

processing methods. It is a way to handle passenger grievances. The system helps to make passenger grievance handling more efficient.. The mathematical formulation explains the way the system changes the unstructured textual complaints into structured data like sentiment, category, and severity.
Let the input complaint be represented as a sequence of words:

$$C = \{w1, w2, w3, \dots, wn\} \quad ,$$

where widenotes the ith word in the complaint and $n$is the total number of words.

Text Preprocessing
Before the complaint text gets to the model, it is pretty much prepared by the system through several steps like tokenization, stop word removal, and lemmatization. A pretrained embedding model such as the BERT tokenizer is used to convert each word into a dense numerical vector.
This converts the text sequence into a numerical vector representation:

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

where $x_i \in \mathbb{R}^d$ represents the embedding vector of dimension $d$corresponding to the word w_i.

Sentiment Classification
Sentiment detection is performed using a fine-tuned DistilBERT transformer. The transformer processes the vector sequence X to generate contextual embeddings for each token. The final hidden state corresponding to the [CLS] (classification) token is denoted as h_{CLS} , which represents the semantic meaning of the entire complaint.
The sentiment prediction is obtained through a softmax layer:

$$P_s = \text{softmax}(W_s h_{CLS} + b_s)$$

where Ws and bs denote the weight matrix and bias vector of the sentiment classifier respectively. The output Ps gives the probability distribution over sentiment classes {Positive, Negative, Neutral}.
The predicted sentiment label is determined as:

$$\hat{y}s = \arg\max(Ps)$$

Complaint Category Classification
For complaint categorization, a separate transformer model is fine-tuned to classify complaints into specific types. The contextual embedding $hCLS$ is passed through a classification layer to estimate the category probabilities:

$$P_s = \text{softmax}(W_s h_{CLS} + b_c)$$

where Wc and bc represent the category classifier's weight matrix and bias. The output $Pc$is a probability distribution over all predefined categories such as Food, Delay, Cleanliness, Facilities, Safety, Ticketing, Misbehavior, Seat Issues, and Others.
The most probable category is then computed as:

$$\hat{y}c = \arg\max(Pc)$$

Priority and Severity Estimation
Once the sentiment and category are identified, the priority and severity of the complaint are determined using a combination of model predictions and keyword-based scoring.
Let Kpbe the set of priority-related keywords and Ksbe the set of severity-related keywords. For each complaint C, a matching function $f(k, C)$returns 1 if a keyword k Kp $Ks$is found in C, otherwise 0. The total keyword score $S$is then computed as:

$$S = \sum_{k \in K_p \cup K_s} f(k, C)$$

The priority level Pr is assigned based on the complaint category and score thresholds as follows:

$$P_r = \begin{cases} \text{High}, & \text{if } \hat{y}_c \in H \text{ or } S \geq \alpha_1 \\ \text{Medium}, & \text{if } \hat{y}_c \in M \text{ or } S \geq \alpha_2 \\ \text{Low}, & \text{otherwise} \end{cases}$$

where $H$and Mrepresent predefined sets of high and medium priority categories, and $\alpha 1$, $\alpha 2$are threshold values.

Similarly, the severity level is derived as:

$$\text{Severity} = \begin{cases} \text{Critical}, & \text{if } S \geq \beta_1 \\ \text{Moderate}, & \text{if } S \geq \beta_2 \\ \text{Mild}, & \text{otherwise} \end{cases}$$

where $\beta 1$, $\beta 2$ are severity thresholds determined through empirical keyword analysis.

Output Generation

Finally, the model produces a structured output vector for each complaint:

$O = \{\hat{y}s, \hat{y}c, Pr, \text{Severity}\}$

where $\hat{y}s$ is the predicted sentiment, $\hat{y}c$ is the complaint category, $Pr$is the assigned priority, and Severity represents the assessed severity level. This structured data is sent back to the Node.js backend in JSON format, where it is securely stored in MongoDB for monitoring and visualization by administrators.

Learning Objective

The parameters of the sentiment and category models are optimized using the categorical crossentropy loss function:

$$L = -\sum_{i=1}^{N} \hat{y}_i \log(\hat{y}_i)$$

where $\hat{y}i$ represents the true label and $\hat{y}i$ is the predicted probability for each class. This function helps the model make predictions. It does this by trying to make sure the model gets the most out of predictions when it is being trained. This leads to the model being more accurate and strong, at classification. The model becomes better at making predictions. The goal is to make the model more reliable. The model maximizes predictions.

F. Other Recommendations

Use either SI (MKS) or CGS as primary units. (SI units are encouraged.) If your native language is not English, try to get a native English-speaking colleague to proofread your paper. Do not add page numbers.

III. CONCLUSION

The Rail Madad system is an example of what can happen when machine learning, sentiment analysis and web-based dashboards come together. These technologies create a system for handling complaints in the railway sector.

The system helps with grievance handling, by summarizing issues prioritizing the urgent ones and providing insights. This way it improves the way of handling complaints. Our tests showed that the system is accurate, fast and efficient. This means it can respond quickly to incidents.

The systems design also makes it open, responsible and user-friendly. Administrators can see trends. Solve pressing issues before they get out of hand.

However there are some limitations. The system depends on quality training data. It can struggle with language. There are also privacy issues. Despite these the Rail Madad system is flexible. Can grow. Its design makes it easy to update and adapt to changes. These changes might include handling types of data adding more languages and expanding to larger railway networks. The Rail Madad system can handle text and images. The system can also work with languages. Rail Madad can be set up across railway operational networks.

ACKNOWLEDGMENT

In 2023 Saranya and their team did a study on complaint management systems. They used Machine Learning to sort tweets into groups like medical, security and maintenance. The system worked with live data streams using Apache Kafka and Spark. This helped railway departments respond to complaints and give service.

Some other people, Abdulla and their team did a study in 2022. They looked at how to make chatbots better. They compared machine learning algorithms and deep learning to see which one could handle a lot of data and talk to users in a way that's helpful. For example deep learning can find features automatically. It is very expensive and needs a lot of data. Abdulla and their team also looked at how chatbots have changed over time from ones like ELIZA to ones like Siri and Alexa. These new chatbots are much better at understanding text and speech. Deshmukh and Shiravale did a study in 2018 on complaint management systems. They worked on something called Priority-Based Sentiment Analysis using Natural Language Processing or NLP. They used models, like SVM and Naive Bayes to figure out how people felt and how strong their feelings were about

complaints from citizens. Then they prioritized the complaints based on how people felt about complaint management systems. This helped the municipal authorities respond to issues faster about complaint management systems. The system made it easier for them to manage complaints.

Swati Agarwal, Ashrut Kumar and Rijul Ganguly did a study in 2022 on complaint management systems. They looked at complaints from Twitter in languages. They used models like mBERT and BERTweet to analyze these complaints about complaint management systems. Their approach was very good. They were able to classify complaints correctly up to 95% of the time about complaint management systems. They made maps to show where complaints were coming from and what people were complaining about like train delays and amenities to improve complaint management systems. They found out that it is very important to have a system that can handle complaints, in languages to make the Indian Railway more efficient and improve complaint management systems.

G. Sathiyapriya and S. Anita Shanthi worked on classifying images in 2022. They used something called Convolutional Neural Networks. Their method was very good. They were able to classify images correctly 99.26% of the time. They used a dataset called Caltech101 to test their method. They also looked at how the neural networks were working and used -trained models like ResNet-50 to understand how the images were being processed.

Anupam Mondal, Monalisa Dey and Dipankar Das made a chatbot for education in 2018. They used machine learning to make the chatbot. The chatbot was able to answer questions and have conversations with users. The chatbot was tested on a platform called Telegram. It worked very well. The chatbot was able to help with communication in institutions.

Haotian Liang did a study in 2021 on how to make language models. He looked at Recurrent Neural Networks. These language models can be very good at understanding language. Haotian Liang proposed a solution to make language models better. His solution was to optimize the way language models are trained.

Siddhant Meshram, Namit Naik and Megha VR wrote a paper in 2021 about chatbots. They looked at types of chatbots and how chatbots work. Chatbots can be very useful in areas like education and healthcare. They said that it is very important to make chatbots that are easy to use and can understand what the user wants. These chatbots need to be simple so users can use them easily.

Vaibhav Tiwari and his team did a study in 2020 on image classification. They used Neural Networks to classify images. Vaibhav Tiwari and his team were able to classify images 98.97% of the time. They used a model called VGG16 to classify the images. Vaibhav Tiwari and his team found out that the VGG16 model can be improved by using data and modifying the architecture. This can make the VGG16 model better, at classifying images.

A survey was done in 2022 by Basu Dev Shivahare and Shashikant Suman. They looked at studies by Gasevic, Tsai and Drachsler. These studies used algorithms like Bayes and neural networks to analyze data. The results were very good. The institutions were able to take steps to improve their outcomes.

Gasević, Tsai and Drachsler did a study in 2020. They used methods like Bayes and clustering to analyze data. They found out that social and technological infrastructure are very important for learning analytics. They said that institutions need to have infrastructure to be able to use learning analytics effectively.

Jamshed Memon did a review of research on OCR in 2020. He looked at research from 2000 to 2019. He found out that there are machine learning and deep learning approaches that can be used to recognize handwritten scripts. He said that there are still challenges, especially, for non-mainstream languages. He suggested that more research is needed to improve the accuracy of OCR systems.

REFERENCES

[1] S. Singh, "Applications of Artificial Intelligence in Chatbots," International Journal of Engineering Research & Technology, vol. 10, no. 6, pp. 135–138, 2021.

[2] R. Patel and P. Shah, "AI-Driven Conversational Agents: A Review," International Journal of Innovative Research in Computer and Communication Engineering, vol. 8, no. 4, pp. 2120–2125, 2020.

[3] M. S. Hussain, "Natural Language Processing in AI-Based Chatbots," Procedia Computer Science, vol. 199, pp. 536–542, 2022.

[4] L. Adams, "The Rise of AI Chatbots in Customer Service," Journal of Business and Technology, vol. 15, no. 2, pp. 99–104, 2021.

[5] J. Lee and H. Kim, "Deep Learning-Based Image Classification: A Comprehensive Review," IEEE Access, vol.9, pp. 12026–12043, 2021.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," Advances in Neural Information Processing Systems (NIPS), vol. 25, 2012.

[7] C. Szegedy, W. Liu, Y. Jia, et al., “Going Deeper with Convolutions,” Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), pp. 1–9, 2015.
[8] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” Proc. IEEE CVPR, pp. 770–778, 2016.
[9] J. Redmon and A. Farhadi, “YOLOv3: An Incremental Improvement,” arXiv preprint, arXiv:1804.02767, 2018.
[10] O. Russakovsky et al., “ImageNet Large Scale Visual Recognition Challenge,” International Journal of Computer Vision, vol. 115, no. 3, pp. 211–252, 2015.
[11] A. Howard et al., “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,” arXiv preprint, arXiv:1704.04861, 2017.
[12] F. Chollet, “Xception: Deep Learning with Depthwise Separable Convolutions,” Proc. IEEE CVPR, pp. 1251–1258, 2017.
[13] C. Shorten and T. M. Khoshgoftaar, “A Survey on Image Data Augmentation for Deep Learning,” Journal of Big Data, vol. 6, no. 60, pp. 1–48, 2019.
[14] Y. Lecun, Y. Bengio, and G. Hinton, “Deep Learning,” Nature, vol. 521, pp. 436–444, 2015.
[15] S. Albawi, T. A. Mohammed, and S. Al-Zawi, “Understanding of Convolutional Neural Networks (CNN),” Proc. International Conference on Engineering and Technology (ICET), 2017
[16] N. Srivastava et al., “Dropout: A Simple Way to Prevent Neural Networks from Overfitting,” Journal of Machine Learning Research, vol. 15, pp. 1929–1958, 2014.
[17] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” Proc. ICML, 2015.
[18] G. Hinton and R. Salakhutdinov, “Reducing the Dimensionality of Data with Neural Networks,” Science, vol. 313, no. 5786, pp. 504–507, 2006.
[19] R. Girshick, “Fast R-CNN,” Proc. IEEE ICCV, pp. 1440–1448, 2015.
[20] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” arXiv preprint, arXiv:1409.1556, 2014.
[21] I. Goodfellow et al., “Generative Adversarial Nets,” Proc. NIPS, 2014.
[22] J. Long, E. Shelhamer, and T. Darrell, “Fully Convolutional Networks for Semantic Segmentation,” Proc. IEEE CVPR, pp. 3431–3440, 2015.
[23] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” Proc. NIPS, 2015.
[24] S. Gandhi and C. Patel, “Survey: Unconventional Categories of Chatbots that Make Use of Machine Learning Techniques,” Journal of Information Technology & Digital World, vol. 5, no. 3, pp. 310–328, Sept. 2023.
[25] E. Kavaz, A. Puig, and I. Rodríguez, “Chatbot-Based Natural Language Interfaces for Data Visualisation: A Scoping Review,” Applied Sciences, vol. 13, no. 12, 2023.
[26] X. Zheng and Z. Zhang, “Image Classification with Deep Convolutional Neural Networks: A Review,” International Journal of Creative Research Thoughts (IJCRT), vol. 12, no. 9, Sept. 2024.
[27] S. Yadav and M. D. Sawale, “A Review on Image Classification Using Deep Learning,” World Journal of Advanced Research and Reviews, vol. 17, no. 1, pp. 480–482, 2023.
[28] A. B. A. Shahin, “Remote Sensing Image Processing, Deep Learning Classification Algorithms Survey and Review,” Journal of Advanced Engineering Trends, vol. 43, no. 1, pp. 239–267, Jan. 2024.
[29] R. Mo, “A Survey of Image Classification Algorithms Based on Convolution Neural Network,” Highlights in Science, Engineering and Technology, vol. 15, article 2222, 2023.
[30] “A Comprehensive Survey of Deep Learning Approaches in Image Processing,” Sensors, vol. 25, no. 2, 2025.