International Journal of Enhanced Research in Management & Computer Applications,
ISSN: 2319-7471, Volume 15 Issue 4, April 2026

"Predictive Analysis for Identifying High-Risk Accident Zones"

Geetha Kumari. B[1], Usakoyala Aravind[2], Vamshi Kommu[3],
Pitta Rahul Stanley[4], Gaddi Sai Krishna[5]

[1,2,3,4,5]Computer Science and Engineering, Gokaraju Rangaraju Institute of Engineering and Technology
Hyderabad, India

ABSTRACT

Traffic safety isn't something we can just push aside and hope gets better on its own. Back home in India, road accidents have been going up every single year and honestly it's become one of the main reasons people end up in the hospital or worse on our roads. If we really want to change things instead of showing up after another crash happens, we need to get serious about finding those dangerous spots, you know the places where accidents keep happening again and again. Here's the thing about accidents. From the outside they look totally random, like nobody could have seen them coming. But once you start digging, you realize we've never really used data properly to understand what's actually going on. That's where cognitive analytics comes in and why we thought it might help.For this study, we spent time going through years and years of accident records. We pulled together everything we could find, traffic density, weather conditions, what kind of road it was, what time of day, past accident records, all of it. The thinking was simple. If we could figure out which factors actually matter, we could give city planners and regular drivers something useful they could work with. We built a system that learns from all that structured data and finds hidden patterns and connections you'd never catch just by flipping through a few reports, but here's the thing we really cared about: we didn't want it to just sit there with old information because accidents don't happen in a bubble.

If it's pouring rain right now or there's a monster traffic jam building up on the highway, that changes everything in real time, so we made sure our model pulls in live weather updates and current traffic conditions too, and mixing that historical stuff with what's actually happening in the moment is what makes the predictions hit closer to home and lets us push out warnings when they might actually help someone. On the technical end, we let machine learning handle the heavy lifting. It automatically figures out which features are worth paying attention to, what to toss aside, and how to classify everything cleanly, which means the whole system can scale up without us having to babysit every little setting along the way. At the end of the day, what we put together is a real-time accident risk prediction framework that we genuinely believe can make a tangible difference by giving traffic teams and urban planners something proactive to work with, helping them manage road networks more intelligently, and if it does what we hope it does, it might actually help bring down how often accidents happen and how bad they are when they do and honestly, that's the whole reason we took this on in the first place.

Keywords: Accident Risk Prediction, Machine Learning, Traffic Accident Hotspots, Real-Time Monitoring, Road Safety, Intelligent Transportation Systems.

INTRODUCTION

Traffic accidents and figuring out where they keep happening, yeah that's basically become the thing everyone's researching now, especially with all this smart transportation stuff and trying to use data to make cities less of a mess. Look at India man, roads here are insane, accidents just keep going up and up and it's honestly one of the biggest reasons people end up dead or hurt. So yeah these prediction systems everyone's building, they're not just about catching people speeding or running red lights [3]. They actually help with a bunch of other stuff too like managing traffic so you're not sitting there for hours, figuring out where to put new roads or flyovers, getting ambulances to crash sites before it's too late, and even just making drivers open their eyes a little about what's happening around them. What we're trying to do with our thing [8] is build something that looks at both old accident records and whatever's happening right now to figure out which spots are

dangerous and how likely someone's gonna crash there, so maybe we can actually stop it instead of showing up after the fact. We're talking about looking at everything, the weather, how many cars are on the road, what kind of road it is, what day it is, where accidents happened before, all of it, just to get some idea of whether a particular stretch is sketchy or not. The old way of doing things, you know sending guys out to check roads, waiting for people to complain, reading through reports after another bad crash, it gives you something I guess but you're always behind. You only fix stuff after enough people have already gotten messed up. And with how complicated everything's gotten [10], especially somewhere like India where you got traffic like you wouldn't believe and roads that go from perfect to absolutely destroyed in like two kilometers [5], it's just so much harder to keep up and stop accidents before they happen. The biggest headache honestly is dealing with how fast things change out there and trying to take all this garbage data from like twenty different places and make one system that actually does something useful.

So this is how we built ours. First we just ask people a couple things, what car they are driving, what road they are on, how the road looks right now, and then we take that and mix it with data we're pulling automatically about the weather and traffic around them [11]. Then all that goes into this machine learning model we trained, sitting on a Flask server in the back. The model takes whatever comes in, filters out the noise, grabs the stuff that actually matters, and then gives us a risk level, low, medium, or high. Soon as that's done we shoot it back to the user through an alert thing that just sits in their browser quiet, so they can keep using Google Maps or whatever without getting annoyed but still know if shit's about to get real up ahead. We added some extra stuff too. You can start or stop the monitoring whenever, we track where you are living, and the whole thing just works whether you're on your phone or laptop. But honestly it's not even just about keeping people safe right at that moment. The stuff this system figures out, it can actually help with bigger conversations later about city planning and policies [2], so traffic people, city planners, decision makers, they finally got something real to work with when they're trying to figure out how to fix our roads. At the end of the day we're taking years and years of accident data and making it actually useful through machine learning and automation, all because we wanna get ahead of crashes before they happen and make sure people driving around every day actually get where they're going in one piece.

LITERATURE REVIEW

Back in 2024, Hong Guo and the people he worked with put out this paper, "Fusion of Satellite and Street View Data for Urban Traffic Accident Hotspot Identification" [15]. They were trying to solve this problem of figuring out where accidents happen most in cities. So they took this Multi-Branch Feature Fusion thing and combined it with TCFGC-Net, some deep learning model, to really get into both the where and when of crashes. For their data, they pulled images from Baidu Street View and Google Earth, then mixed those with actual crash records from Xiaoshan District in Hangzhou from 2021. By throwing all that together they managed to get pretty decent results picking out dangerous spots even in complicated city areas. But honestly their work has some real issues. No real-time capability whatsoever. They never checked their results against data from anywhere else so who knows if it would work outside Hangzhou. The data they used probably has bias built right in since it's all from one place. And the whole thing needs so much computing power that actually using it anywhere would be a huge pain.

Keisuke Ando and his team. They also published something in 2024 called "Cluster Detection for Traffic Accidents on Spatiotemporal Networks"[16]. They went after accident clusters using ETD-TNKDE and hypothesis testing and all that stuff. They worked with accident records from Aichi Prefecture in Japan, 2015 through 2020, and ran everything through Local Moran's I, PAI, and Monte Carlo simulations to understand how crashes connect across space and time. The framework they built actually does a solid job finding places where accidents happen a lot by looking at those connections. But the same problems really. Needs crazy computing power. Can't do real time. They had trouble verifying if their data was even accurate. So getting this thing out there for real people to use isn't happening anytime soon.

Ai-Bing Zhe and their crew put together this big review in 2024, "Enhancing Road Safety with ML: Advances & Future Directions" [17]. They went through all kinds of machine learning work people are doing to make roads safer. They checked everything against all those standard metrics, accuracy, precision, recall, F1-score, AUC, MAE, RMSE, all of it. And they looked at data from everywhere, government reports, Kaggle, sensors, stuff scraped off websites. They mapped out where the field is going and what's next for ML in traffic safety. But they also called out all the problems. Real-time implementation is still a mess. Data quality is all over the place. There's ethical stuff nobody's even touched yet. Every research group uses their own thing so nothing's standard. And half these models are just black boxes where you can't see what's happening inside, which makes people nervous about actually using them.

METHODOLOGY

People have tried all kinds of approaches in intelligent transportation systems to get better at predicting accidents and

finding hotspots, but here's the thing predicting accident risk [1] in real time is honestly pretty tough because it depends on so many things that keep changing moment to moment, like how heavy the traffic is, what the weather's doing, what kind of vehicle someone's driving, the condition of the road, and even the type of road they're on, so with our system what we ended up building is a Real-Time Accident Risk Prediction System that follows a pretty straightforward path from figuring out what we needed all the way through to actually deploying it, and the whole goal was to create something that lives in the browser and works quietly in the background without you having to think about it while still giving you accident risk alerts based on what's actually happening around you right now, all powered by a machine learning [14] algorithm we trained up because we really wanted to get away from those old reactive approaches where you only find out about problems after something bad has already gone down, so we built this thing to be mostly automated, ask for as little input from users as possible, and just keep monitoring continuously without anyone having to poke at it, and to make that work we hooked into some APIs TomTom Traffic API to grab live traffic congestion levels and WeatherAPI to pull in current weather conditions while on the backend we set up a Flask server that takes in whatever the user enters along with all that environmental data and then passes it through a Random Forest Classifier we trained beforehand to actually predict the accident risk, and the model spits out one of three levels Low, Moderate, or High which we then push back to the user through a frontend we built using HTML, CSS, and JavaScript that works everywhere, lets you select what kind of vehicle you're driving, hit Start or Stop whenever you feel like it, taps into the Geolocation API to track where you are so the predictions actually mean something for your specific location, and if the risk level changes you get a live alert right there in your browser, and the whole thing is set up so the frontend and backend just talk to each other smoothly in the background, automatically updating predictions every few minutes without ever interrupting you while you're trying to navigate somewhere.

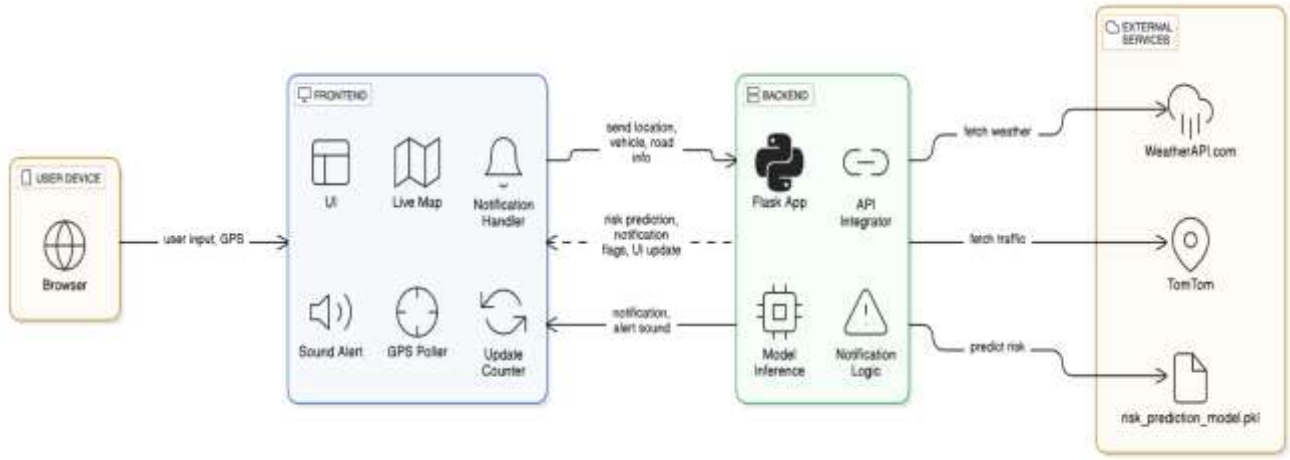

Figure3.1: System Architecture

Data Gathering and Model Development

We went looking for accident data across public transportation databases and traffic study platforms. Found some solid datasets eventually. They had everything we were after, road type, road condition [6], vehicle category, traffic condition, weather status, the whole package. First we cleaned the house. Handled missing values. Turned categorical fields into something the model could digest.Then we realized the data was skewed. Some risk levels had way more records than others. That messes with training. So we brought in SMOTE to even things out. It creates synthetic samples for whatever groups are lacking. We kept running SMOTE on those three risk groups until the distribution finally looked right.Once we had balanced data we split it three ways, 70% for training, 20% for validation, and 10% for testing. For the model we picked Random Forest Classifier. Works well with this kind of structured data. You can peek inside and see how decisions get made. Overfitting isn't really a problem with it. We configured everything manually, 18 estimators, max depth of 8, minimum 15 samples to split, minimum 6 samples per leaf, and random state 42. We also threw in some noise on traffic[9] density and weather. Nothing crazy. Just enough to mimic what real roads look like. Helps the model handle stuff it hasn't seen before. When evaluation time came we checked precision, recall, F1-score, and ran confusion matrix analysis. I wanted to really understand how it performed across all three risk levels [12].

Backend and Real-Time Integration

Then for the backend and real-time integration piece, we built everything using Flask and deployed it on a local server so it could handle prediction requests through RESTful API endpoints, and we saved that trained Random Forest model as a

serialized pickle file that loads up right when the server starts, and the backend is set up to accept real-time inputs like GPS coordinates, vehicle type, road type, road condition, weather data, and traffic[4] density, and it even calls external APIs on the fly to pull in additional environmental parameters, and once it processes everything it sends predictions back in JSON format with risk labels 0 for Low, 1 for Moderate, and 2 for High and over on the frontend we used JavaScript fetch() functions to call those backend APIs periodically while the Geolocation API keeps updating the user's location continuously, and depending on what risk level comes back the system generates browser-based notifications and audio alerts so you actually know when things get dicey, and we tested the whole integrated setup using Postman and live browser testing to make sure everything talked to each other smoothly and we ironed out any CORS-related headaches, and finally for deployment we made sure all dependencies were listed in requirements.txt and verified everything worked across different browsers, so at the end of the day what we've got is a complete pipeline that takes you all the way from pulling in data and preprocessing it to training the model, integrating the backend, and deploying the whole thing as a lightweight, automated, real-time accident [9] risk prediction web application that actually works.

RESULTS

After we got everything built and hooked up, we finally put the whole system through its paces using both the historical datasets we'd collected and live environmental inputs streaming in from those APIs, and honestly it held up pretty well that Random Forest classifier we trained turned out to be a solid choice because it consistently sorted accident risk levels into Low, Moderate, and High categories without much fuss, and when we ran all the usual metrics on it like precision, recall, F1-score, and dug into the confusion matrix to really see where it might be tripping up, the numbers looked consistent and reliable across all three risk levels, and when we took it out for real-time testing the system kept up nicely it took whatever GPS coordinates we threw at it, pulled in current weather conditions, checked traffic density, factored in road parameters, and spat out dynamic risk alerts that actually made sense for the moment, while the browser-based interface did its job too by showing visual notifications and playing audio warnings that matched whatever risk category came back, and if you look at Figures 4.1, 4.2, and 4.3 you can see sample outputs we captured Figure 4.1 shows what it looks like when things are Safe or Low Risk, Figure 4.2 shows a Moderate Risk scenario, and Figure 4.3 shows a High Risk detection in action—and what really made a difference was combining that static historical data we trained the model on with live API inputs from the real world because it gave the system way better context and made the predictions actually responsive to what was happening on the ground at that exact moment, so at the end of the day the whole thing just worked the way we hoped: reliable classification, smooth communication between the frontend and backend, and real-time alerts popping up without ever interrupting someone trying to navigate from point A to point B.

Figure4.1: Road accident detection - Safe

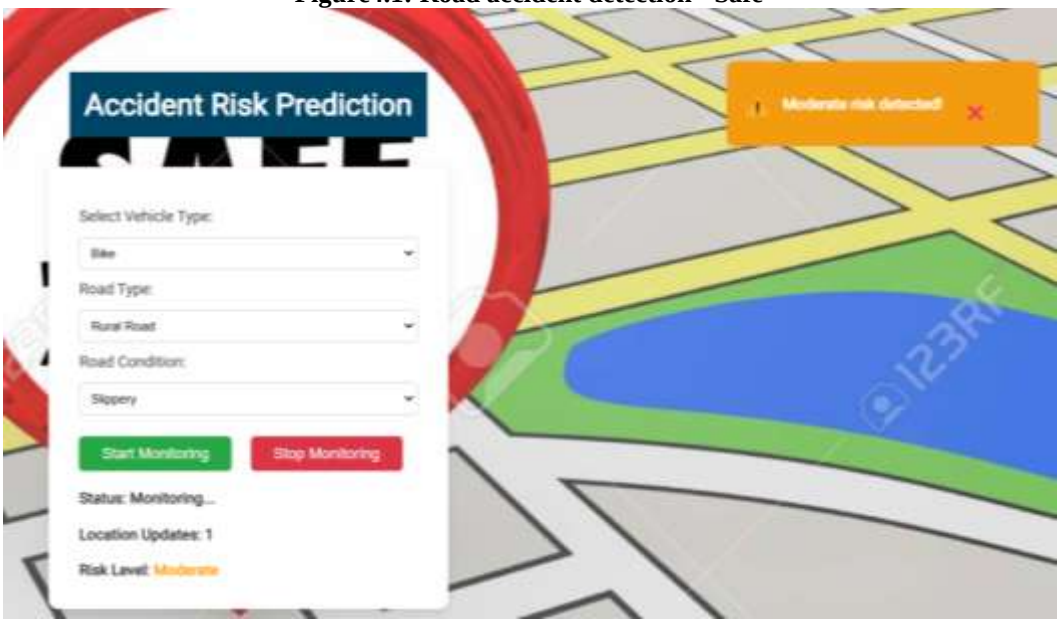

Figure 4.2: Road accident detection – Moderate

Figure 4.3: Road accident detection - Moderate

CONCLUSION

So here's what we ended up with: an AI-Powered Student Productivity Platform that actually does what we set out to build, which is transforming how students learn by processing study materials and generating summaries, quizzes, and task recommendations without asking much from the user. Beyond just summarizing whatever PDF gets uploaded, we built in this intelligent content pipeline that runs under the hood and intelligently mixes together NLP models, subject-aware assistance, and real-time quiz generation to create dynamic academic support that actually means something for that exact moment in someone's studying. By leaning on large language models and keeping those inputs fresh and continuously updated, the system pushes out accurate summaries and timely learning recommendations to students without them having to manually create flashcards or dig through endless notes. The whole thing lives in the browser and just sits there quietly in

the background, so students can keep doing whatever they were working on, listening to music, whatever while still staying in the loop about their progress and upcoming tasks. When we actually built it out and tested it with real users, we found it's practical, it scales up fine, and people don't hate using it because the interface is intuitive and we kept the architecture modular so it's not some monolithic mess that crashes every time someone uploads a large file. The system tracks study behavior, quiz results, and task completion patterns across multiple sessions, and those dashboards offer measurable progress indicators that actually promote disciplined and efficient study habits instead of just looking pretty. It also lets students access everything remotely without being chained to tutor availability or wasting time on manual content processing. At the end of the day, and you can see this in those dashboard figures, what we're really showing here is how intelligent automation and AI-driven models don't have to be complicated or annoying to actually help students learn better and make sure they get through their academic journey without burning out. Sure, large-scale deployment still needs work on handling different file formats, keeping AI outputs accurate for specific subjects, optimizing those heavy NLP workloads, and protecting sensitive academic data, but as AI tech and cloud infrastructure keep getting better, this platform can only become more powerful and useful for more students.

REFERENCES

1. Haghani, M.; Behnood, A.; Oviedo-Trespalacios, O.; Bliemer, M.C. Structural anatomy and temporal trends of road accident research: Full-scope analyses of the field. J. Saf. Res. 2021, 79, 173-198.
2. Singh, N.; Katiyar, S.K. Application of geographical information system (GIS) in reducing accident blackspots and in planning of a safer urban road network: A review. Ecol. Inform. 2021, 66, 101456.
3. Naboureh, A.; Feizizadeh, B.; Naboureh, A.; Bian, J.; Blaschke, T.; Ghorbanzadeh, O.; Moharrami, M. Traffic Accident Spatial Simulation Modeling for Planning of Road Emergency Services. ISPRS Int. J. Geo-Inf. 2019, 8, 371.
4. Hazaymeh, K.; Almagbile, A.; Alomari, A.H. Spatiotemporal Analysis of Traffic Accidents Hostpots Based on Geospatial Techniques. ISPRS Int. J. Geo-Inf. 2022, 11, 260.
5. Wang, C.; Li,s.; Shan, J.Non-Stationary Modeling of Microlevel Road-Curve Crash Frequency with Geographically Weighted Regression. ISPRS Int. J. Geo-Inf. 2021,10, 286.
6. Mesquitela, J.; Elavs, L.B.; Ferreira, J.C.; Nubes, L. Data Analytics Process over Road Accidents Data: A Case Study of Lisbon City. ISPRS Int. J. Geo-Inf. 2022,11,143.
7. Kumar, S.; Tiwari, P.; Denis, K.V. Augmenting Classifiers Performance through Clustering: A Comparative Study on Road Accident Data. Int. J. Inf. Retr. Res.(IJIRR) 2018, 8, 57-68.
8. Abellán, J.; López, G.; De OñA, J. Analysis of traffic accident severity using decision rules via decision trees. Expert Syst. Appl. 2013, 40, 6047-6054.
9. Li, Z.; Guo, X.; Sun, J. Analysis and Research on the Temporal and Spatial Correlation of Traffic Accidents and Illegal Activities. In Proceedings of the International Conference on Cloud Computing and Security, Singapore, 29-31 October 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp.418-428.
10. Kumar, S.; Toshniwal, D. A data mining framework to analyze road accident data. J. Big Data 2015, 2, 26.
11. Zeng, Q.; Huang, H. A stable and optimized neural network model for crash injury severity prediction. Accid. Anal. Prev. 2014, 73, 351–358.
12. Katsoukis, A.; Iliadis, L.; Konguetsof, A.; Papadopoulos, B. Classification of Road Accidents Using Fuzzy Techniques. In Proceedings of the 2018 Innovations in Intelligent Systems and Applications (INISTA), Thessaloniki, Greece, 3-5 July 2018; IEEE: Manhattan, NY, USA, 2018; pp.1-5.
13. Turunen, E. Using GUHA data mining method in analyzing road traffic accidents occurred in the years 2004–2008 in Finland. Data Sci. Eng. 2017, 2, 224–231.
14. Le, T.T; Fu, W.; Moore, J.H. Scaling tree-based automated machine learning to biomedical big data with a feature set selector. Bioinformatics 2019, 36, 250-256.
15. Wentong Guo, Cheng Xu, Sheng Hin. Fusion of satellite and street view data for urban traffic accident hotspot identification. International journal of Applied Earth Observation and Geoinformation 130:103853. DOI:10.1016/j.jag.2024.103853, June 2024.
16. Yusuke Kuniyoshi, Natsuki Oogi, Takeshi Uchitane. Cluster Detection for Traffic Accidents on Spatiotemporal Networks. January 2024, Procedia Computer Science 246:371-380. DOI:10.1016/j.procs.2024.09.416.
17. Albe Bing Zhe Chai, Bee Theng Lau, Mark Kit Tsun Tee, Chris McCarthy. Enhancing road safety with machine learning: current advances and future directions in accident prediction using non-visual data. Engineering Applications of Artificial Intelligence Volume 137, Part A, November 2024, 109086.