Journal of
Communication Engineering & Systems

STM JOURNALS

https://journals.stmjournals.com/joces

ISSN: 2249-8613 (Online)
ISSN: 2321-5151 (Print)
Volume 14, Issue 3, 2024
September–December
DOI (Journal): 10.37591/JoCES

Review JoCES

Identifying and Blocking of Non-productive Calls in Emergency Call System Using Machine Learning and IVRS Integration

Ishika Jain[1,*], Anjali Singh[2]

Abstract

The Dial 100 emergency reaction gadget handles over 1,000,000 calls every day, with over 95% being unproductive, such as blank, machine-generated, and spoofed calls. These futile calls waste resources and put off responses to actual emergencies. This look presents a comprehensive solution integrating advanced name evaluation, system learning, and an interactive voice response system (IVRS) to filter out and prevent these calls. Our technique starts with studying incoming calls to pick out patterns typical of unproductive ones. Audio processing detects blank calls, even as device-generated ones are recognized via sample recognition. Caller ID monitoring and dynamic thresholds for call frequency and period make certain adaptive filtering. Machines getting-to-know fashions, consisting of Random Forest and Support Vector Machines, use historic call data to classify calls based on various standards, constantly enhancing through new information. IVRS integration similarly reduces the weight of emergency operators by intercepting unproductive calls. The device also keeps blacklists to block repeat offenders. A feedback loop allows non-stop machine optimization with the aid of incorporating operator entries and periodic opinions.

Keywords: Interactive voice response system (IVRS), caller ID, random forest, support vector machines

INTRODUCTION

One million calls (cumulative) are from the Dial 100 system, which is an important component of the emergency response side. Similarly, 95% of these calls are useless. These can be blank, system-generated, or prank calls. These fake calls waste resources, clog up the system and generate real harm to people who need help [1].

This garbage will not do so in the current manner that we treat these calls. Therefore, a smarter approach is needed. The goal of this study was to construct an entire system capable of recognizing and blocking non-existent (unwanted) calls. It aims to leverage machine learning, deep dive call analysis, and an interactive voice response system (IVRS).

We intend to obtain a Dial 100 system that works better by doing this. This will help keep emergency responders focused on actual emergencies and help people who need them.

This TAR (Tape Archive) outlines all areas of the fix, how to configure it, and the test steps. It also describes the issues and existing treatments. This interface needs to be extendable, and as such, we

***Author for Correspondence**
Ishika Jain
E-mail: 2023pcecsishika066@poornima.org

[1]Student, Department of Computer Science and Engineering, Poornima College of Engineering, Jaipur, Rajasthan, India
[2]Assistant Professor, Department of Computer Science and Engineering, Poornima College of Engineering Jaipur, Rajasthan, India

Received Date: September 16, 2024
Accepted Date: September 24, 2024
Published Date: October 07, 2024

Citation: Ishika Jain, Anjali Singh. Identifying and Blocking of Non-productive Calls in Emergency Call System Using Machine Learning and IVRS Integration. Journal of Communication Engineering & Systems. 2024; 14(3): 20–28p.

aim to establish a base that can evolve in future work so that the system is maintained by making it better but also abseiled for changing call patterns.

METHODOLOGY

Our approach is orthogonal and complementary to existing proposals for revamping the Dial 100 emergency service [2]. We propose several efficient and state-of-the-art advancements in each layer of the entire system using advanced automatic call analysis, machine learning techniques, and existing IVRS. The following points summarize our strategy

Call Analysis

Audio Processing

- *Voice activity detection (VAD):* The audio of incoming calls has listened to determine whether they contain human speech, silence, or background noise. Calls featuring long durations of silence or noises that are typical of automated systems, as opposed to human speech for business outcomes, are considered unproductive.
- *Frequency spectrum analysis:* By analyzing the frequency patterns present within call audio, we can differentiate between synthetic audio signals (indicative of automated systems) and real human speech for business outcomes [3]. This helps to identify machine-generated calls against those originating from real people.

Sample Introduction

- *Time series analysis:* This method analyzes call data over time to identify recurring patterns of calls made by the system. For example, calls can be analyzed for a constant duration or length to determine whether they follow a specific automation pattern.
- *Behavioral analysis:* Anomalies can be identified by observing various call parameters, such as call duration, frequency, and time of day. For example, if one number receives multiple calls in a short period, it could be a sign of a troubled call.

Caller ID Tracking

Managing Blacklists and Whitelists

- *Dynamic blacklist:* We created and maintained a blacklist of telephone numbers known to produce unproductive calls. This listing is continuously up to date, primarily based on ancient records and real-time analysis to ensure that it stays relevant [4].
- *Whitelist:* To avoid inadvertently blocking off crucial calls, we maintain a whitelist that relies on numbers. This list consists of demonstrated emergency provider numbers and frequent callers with legitimate wishes to ensure that they are not wrongly blocked.

Dynamic Thresholds

- *Threshold configurations:* We set dynamic thresholds for various call parameters, such as the variety of calls from a single range within a given timeframe. For example, if a range makes more than 10 calls per hour, it triggers a flag for an additional assessment.
- *Adaptive learning:* The gadget is used beyond records to dynamically alter these thresholds. As calling styles evolve, these thresholds have been modified to remain powerful and relevant.

Machine Learning

Data Collection and Preprocessing

- *Gathering historical call data:* We gather and put together statistics from beyond calls, which include information such as call period, frequency, time of day, audio nice, and caller ID. These data are crucial for schooling our gadgets and studying fashions [5].
- *Feature engineering:* This entails selecting and refining the maximum applicable functions from the facts. We were conscious of elements such as frequency styles, speech-to-silence ratios, and phone durations to decorate the performance of our models.

Model Development and Training

- *Classification models:* We educate various gadgets gaining fashion knowledge, consisting of Random Forests, Support Vector Machines (SVM), and Neural Networks, using pre-processed facts. These models categorize calls as efficient or non-productive, primarily based on the styles they discover within the statistics [6].
- *Hyperparameter tuning:* To improve model performance, we modified their hyperparameters and validated them through strategies such as go-validation. This quality tuning ensures that the fashions are as correct as possible.

Continuous Learning

- *Model updating:* We frequently updated our models with new call data to maintain their accuracy and flexibility. This guarantees that fashions can handle changing trends and patterns in name data [7].
- *Input integration:* Feedback from operators regarding misclassified calls is incorporated into the models to continually enhance their performance and accuracy.

IVRS Integration

Pre-call Screening

- *Automated screening:* The IVRS uses pre-described standards primarily based on name analysis and machine learning to display screen calls before they reach human operators. This enables filtering out calls that do not meet the criteria for additional interest [8].
- *Prompt handling:* If a call is identified as inefficient, the IVRS performs an automated message and disconnects the call earlier than it reaches the operator. This facilitates more effective coping with name volumes.

Blocklist Synchronization

- *Real-time updates:* To save repeated ineffective calls from equal numbers, the IVRS is updated in actual time with the present-day blacklist records. This guarantees that acknowledged intricate numbers are addressed directly.
- *System integration:* We ensure seamless integration between the IVRS, gadgets getting-to-know the models, and contact analysis structures to offer an efficient name-handling system.

Continuous Improvement

Feedback Loop

- *Operator feedback:* Operators are encouraged to offer comments on calls that have been wrongly labeled. This comment has been treasured for retraining and improving the fashion industry.
- *Data aggregation:* Feedback and further information were aggregated and used to refine the models, ensuring that they continued to improve over time.

Periodic Reviews

- *Performance monitoring:* We frequently display key performance indicators, along with model accuracy, response instances, and the fee for fake positives and negatives. This enables an assessment of the effectiveness of the system.
- *Model refinement:* Based on overall performance opinions, we replace system getting-to-know models, modify thresholds, and refine name identity standards to beautify universal device overall performance.

Reporting and Analytics

- *Regular reports:* We generated reports on name patterns, blacklist effectiveness, and overall machine performance. These reviews provide transparency and guide ongoing device optimization.

- *Data analytics:* Advanced analytics is used to discover trends and patterns in useless calls, presenting insights for destiny enhancements and adjustments to the gadget.

By enforcing this complete method, we aim to significantly decorate the Dial 100 system performance, ensuring that emergency responders can pay attention to true emergencies and correctly control name volumes [9].

Implementation Steps

Data Collection and Preprocessing

- *Gather data:* Start by collecting historical call records. These data include information such as the call duration, frequency, and caller IDs.
- *Prepare data:* Clean and organize the data to highlight the key details required for training our models. This step ensured that the data was accurate and ready for analysis.
- *Ensure privacy:* Handle all data with care, ensure privacy protection, and follow data protection regulations throughout the collection and preparation processes.

Model Development and Training

- *Build and train models:* Use the prepared data to develop and train the machine learning models. These models learn to identify which calls are productive.
- *Optimize performance:* Test the models to see how well they perform in identifying correct calls. Fine-tune the settings to improve accuracy and reliability.

System Integration

- *Connect systems:* Integrate trained models with IVRS and call monitoring tools. This enables the system to analyze calls as they arrive.
- *Real-time screening:* The IVRS is set up to use models for real-time call screening. This means that calls identified as unproductive can be filtered out before reaching human operators.

Testing and Validation

- *Live testing:* The system is tested with real call data to ensure that it works correctly. Look for any issues such as incorrectly labeling valid calls as unproductive (false positives) or missing unproductive calls (false negatives).
- *Trial runs:* Conduct trial runs to determine how the system performs in real-world scenarios. Any necessary adjustments should be made to ensure that it functions as expected.

Deployment and Monitoring

- *Gradual rollout:* Start implementing the system in phases, beginning with less critical call flows. This approach helps to manage any issues before a full-scale rollout.
- *Ongoing maintenance:* Continuously monitor the system's performance and adjust, as needed. The models and settings are updated based on new data and feedback to keep the system running smoothly.

By following these steps, we aim to significantly reduce the number of ineffective calls in the Dial 100 system. This will help ensure that emergency calls are handled efficiently, and resources are used effectively.

IMPLEMENTATION

To put our solution into effect, we first acquire and prepare statistics from beyond calls to educate gadget-mastering models. These fashions are then incorporated with an IVRS to display screen calls in real time [10]. We rigorously checked the system to ensure its accuracy and made necessary modifications. Finally, we continuously screened and replaced the system to maintain its effectiveness and scalability. Each step is critical for ensuring that the system effectively blocks unproductive calls and handles high call volumes.

Data Collection and Preprocessing

Gathering Historical Call Data

- *Where to get the data:* Start by way of collecting facts from the Dial 100 system's name logs. This consists of information such as while the decision changed into made, who made it, how lengthy it lasted, how often certain numbers were named, or even audio recordings.
- *Keeping it private and secure:* It is crucial to ensure that all the information you gather respects privacy laws. This approach anonymizes personal information so that people's identities can be protected.

Getting the Data Ready

- *Cleaning it up:* Getting rid of any wrong or lacking data. Ensure that the entire system is in a steady layout.
- *Picking out what is important:* From uncooked records, pull out key information such as how lengthy calls remain, how much silence there is as compared to talking, patterns in how frequently calls are made, and characteristics of the caller ID.
- *Making it uniform:* Normalize the information such that special features do not outweigh each other, and the version can be analyzed efficiently.

Model Development and Training

Creating Features

- *Call details:* Use attributes such as how long the call lasted, what time it was made, how frequently a number of calls were made, and the audio characteristics.
- *Audio details:* Extract features from the audio, together with how silent an awful lot is, the spectral residences, and the ratio of speech to background noise.

Choosing the Right Model

- *Testing different approaches:* Try out various systems to get to know strategies such as Random Forest, SVM, and Neural Networks to peer which one works first-rate together with your information.
- *Training the models:* Use the cleaned-up facts to educate distinctive models with the aid of splitting them into training and validation sets. Ensure that they perform well using go-validation techniques.
- *Fine tuning:* The settings (hyperparameters) of every model are adjusted to obtain the best performance. Methods such as random search or grid search can be used systematically.

Evaluating the Model

- *How it is measured:* Check how properly the fashions carry out the use of metrics like AUC-ROC (a manner to measure how properly the version distinguishes among classes), F1-rating, accuracy, precision, and consideration.
- *Picking the best one:* Choose the high-quality version based on these reviews and get it geared up to be used.

System Integration

Integrating with IVRS

- *Screening calls in real-time:* Combine the educated device learning version with the IVRS so that calls can be examined as they arrive. The IVRS uses the version's predictions to screen calls.
- *Automatic call handling:* For calls identified as unproductive, the machine can automatically respond, such as by playing a message and then striking up. Make sure that the IVRS can alter its responses based on ultra-modern model outputs.

Managing Caller IDs

- *Updating blacklists and whitelists:* Use the IVRS to dynamically manage lists of blocked (blacklist) and allowed (whitelist) numbers. Ensure that updates to these lists manifest right now to save repeated unproductive calls.

Testing and Validation

Pilot Testing

- *Controlled environment:* Test the device by deploying it in a managed setting, the use of a pattern of actual name information to imitate real situations.
- *Monitoring:* Keep an eye fixed on how the system appears and log each selection made via the model and IVRS. Review these logs to identify issues in how calls are classified or overall system performance problems.

Validating Performance

- *Checking accuracy:* Ensure that the machine correctly identifies inefficient calls by evaluating its selections to those made with the aid of human operators.
- *Stress testing:* Test how the machine handles excessive volumes of calls to ensure that it may perform beneath stress without dropping accuracy or speed.

Incorporating Feedback

- *Listening to operators:* Obtain comments from emergency response operators on how well the device is running. Using their input to pleasant songs, the IVRS criteria, and fashions.

Continuous Improvement

Updating the Model

- *Regular updates:* Continuously feeding new call data into a device-mastering fashion to trap changing patterns in unproductive calls.
- *Retraining:* Regularly retrain the models using modern-day records and operator remarks to maintain excessive accuracy.

Monitoring the System

- *Tracking key metrics:* Keep watch on essential performance signs such as operator workload, machine reaction time, and rates of false positives and negatives.

Implementation Timeline

Phase 1: Getting Ready (First Month)

- *Gathering and prepping data:* Start by collecting all the information you want and prepare for analysis.
- *Choosing the first model:* Begin to select the models and determine which capabilities are most critical.

Phase 2: Building (Months 2–3)

- *Fine tuning and training models:* This time is spent adjusting the version settings to obtain pleasant performance and educate them very well.
- *Integrating with IVRS:* Once the models are equipped, they are connected with the IVRS to start studying calls in real-time.

Phase 3: Testing (Months 4–5)

- *Controlled testing and pilot launch:* Test the machine in a controlled environment to see how it performs and then roll it out in a pilot segment.
- *Incorporating feedback:* Gather comments from operators and validate the gadget's performance, making any essential adjustments.

Phase 4: Full Implementation (Month 6)

- *Final rollout and real-time monitoring:* The machine is fully implemented with live monitoring to ensure that it runs smoothly.

- *Setting up for continuous improvement:* Put mechanisms in the region for ongoing updates and enhancements.

Phase 5: Ongoing Maintenance

- *Regular monitoring and updates:* Keep a close watch on the gadget, making frequent updates and modifications to keep it strolling.
- *Routine reporting and reviews:* Regularly review the device's overall performance and convey reports to ensure that it remains effective.

With this plan, the goal is to create a dependable, scalable, and adaptable gadget that substantially reduces the range of unproductive calls to the Dial 100 System in the long run, making emergency response operations more green and powerful [11].

TESTING AND VALIDATION

Adequate testing and verification are necessary to ensure that the proposed system is valid, reliable, and sufficiently robust to handle real-world conditions. Before scale-up, we take this step to carefully check how well the system works in a controlled environment. Here, represents a broad breakdown.

Pilot Implementation

Establishing A Controlled Environment

- *Real-life scenario simulation:* Model a test setup close to a real Dial 100 setup. This includes the reconstruction of call volumes and patterns using historical data.
- *Isolation:* Isolate pilot tests from active operations to ensure there is no impact on the immediate implementation of the system.

Starting the Initial Rollout

- *Limited rollout:* The ideal method is to use the system in a small group of calls before it is widely adopted. Start with non-critical or low-priority ones to see how they perform without actually hindering important activities.
- *Monitoring the system:* The key is to ensure that nothing unusual occurs in the system. As such, we set up a system that records detailed call data, which includes the calls being classified and managed by the system.

Performance Validation

Testing

- *Classification accuracy:* How well do machine learning models classify calls as productive or non-productive? The F1-score, recall, accuracy, and precision were used to measure.
- *Minimizing errors:* False positives and false negatives—calls labeled as productive or non-productive incorrectly. Reducing these values will make the system more reliable.

Testing Under Load

- *High traffic:* Test under heavy call volumes to see if it can handle the load without slowing down or losing accuracy.
- *Scaling:* Gradually increase the load to see where the performance bottlenecks are.

Getting Feedback from Operators

- *Operator involvement:* Compare the system's call classifications with human operators to observe the differences and refine the models.
- *Feedback:* Set up a process for operators to provide feedback on the system. This feedback was used to identify areas for improvement.

Integrating Feedback

Refining the System

- *Model adjustments:* Based on operator feedback and accuracy testing, adjust the machine learning models. This might involve retraining the models with new data or adjusting the settings.
- *Threshold adjustments:* Adjust call frequency and duration thresholds in real-time based on feedback and data so that they continue to work well for unproductive calls.

Continuous Learning

- *New data:* Update the training set with new call data regularly to keep up with changing patterns. The models are frequently refreshed to stay accurate.
- *Feedback loop:* Create a continuous process in which system performance data and operator feedback are reviewed and used to improve the models and handling criteria.

Metrics and Reporting

Performance Metrics

- *Overall accuracy:* How well does the system classify calls.
- *Precision and recall:* How many non-productive calls are among those labeled as such and how many of the actual non-productive calls are in the detected non-productive calls set?
- *F1-score:* The F1-score gives the best performance of the system by considering both recall and precision.
- *Response time:* See how the system can process calls, even under heavy loads, to test its speed.

Reporting

- *Frequency:* Create a series of reports on system performance, including call trends, classification accuracy, and operator feedback.
- *Trending:* Use new technology to observe data trends in unproductive calls. Back this with data-based feedback to adjust the system beforehand.

Our Goal

According to this comprehensive testing and validation plan, we plan to make the new system accurate, reliable, and robust. By conducting a thorough check on everything, we can identify and fix problems before the system is completely launched. Consequently, the system will be able to minimize the number of unnecessary calls to the Dial 100 system.

FLOWCHART AND PICTORIAL REPRESENTATION

This flowchart shows the steps for implementing the proposed solution, as shown in Figure 1. Each task was broken into smaller parts, showing the sequence for completing the work.

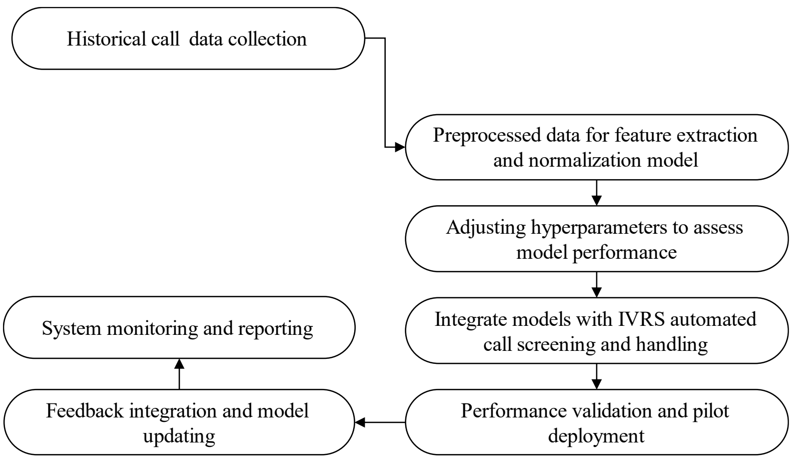

Figure 1. Call data processing and system integration workflow.

CONCLUSION

Smart Tech to Boost Emergency Response

The new system aims to enhance the Dial 100 emergency response setup by eliminating unnecessary calls. Currently, 95% of calls are silent robotic messages or jokes. This means that real crises do not receive the necessary care.

The breakdown is as follows: the setup uses call checks and smart tech to spot important calls. It looks at call trends and talks to sort helpful calls from the rest. Smart tech models trained in old call information can identify such calls. A voice menu system can handle calls on its own and check them as they arrive. This reduces work for human staff, so they can zero in on real emergencies. The setup learns and grows by obtaining input from staff and adding new information. Therefore, it has continued to improve over time.

The perks: Staff have less to deal with, help comes faster, and resources are used better. It can take many calls and grow as needed. Handling call traffic builds trust with people and ensures that help appears quickly and on time.

REFERENCES

1. Engelen BBM. Routing and Staffing methods to optimize multi-skilled call centre processes [Master's thesis]. Eindhoven: Eindhoven University of Technology; 2008. Available from: https://research.tue.nl/files/150910519/638306.pdf
2. Bandyopadhyay K. Mobile Commerce. 2nd ed. New Delhi, India: PHI Learning Pvt. Ltd; 2022.
3. Bergevin R, Kinder A, Siegel W, Simpson B. Call Centers for Dummies. 2nd ed. Indianapolis, IN: Wiley Publishing, Inc; 2010.
4. Almonte DG. Disaster Recovery Best Practices for the Dominican Republic's Contact Center [thesis]. Rochester, NY: Rochester Institute of Technology; 2010.
5. Jeschke S, Kampker A, Kuhlen TW, Schuh G, Schulz W, Al Khawli T, et al. Virtual Production Intelligence (VPI). In: Brecher C, Özdemir D, editors. Integrative Production Technology. Cham: Springer; 2017. p. 177–251. doi: 10.1007/978-3-319-47452-6_4.
6. Pearah DE. The Voice Web: A strategic analysis [master's thesis]. Cambridge (MA): Massachusetts Institute of Technology, Engineering Systems Division, Technology and Policy Program; 2002. Available from: http://hdl.handle.net/1721.1/29249
7. Goldman DJ. Resistance in the Digital Workplace: Call Center Workers in Bell Telephone Companies, 1965–2005 [dissertation]. College Park, MD: University of Maryland; 2021.
8. Ubisse SP. Management of Productivity in a Service Call Centre [thesis]. Johannesburg, South Africa: University of Johannesburg; 2016.
9. Fishel JT. Politics and Progress in the Peruvian Sierra: A Comparative Study of Development in Two Districts [thesis]. Bloomington, IN: Indiana University; 1971.
10. Lebitso MC. The World of Work: Challenges for South African Students [thesis]. Johannesburg, South Africa: UJ Press; 2012.
11. Srivastava DK, Chauhan DS, Singh R. VRS model: A model for estimation of efforts and time duration in development of IVR software system. Int J Softw Eng. 2012;5:27–46.

