

# URBANSENSE – A MACHINE LEARNING BASED URBAN ISSUE MANAGEMENT SYSTEM

Dr. M. Mayuranathan, M. Akshaya, K. Charulatha, P. S. Geetha Lekshmi

*Department of Computer Science and Engineering*

*SRM Valliammai Engineering College (Autonomous), Kattankalathur, Chengalpattu – 603 203, Tamil Nadu, India*

\*\*\*

**ABSTRACT** - Urban populations are growing rapidly, which puts enormous strain on the municipal infrastructure and civic services responsible for keeping it running. Issues such as waste accumulation, road damage, pipeline breaks, blocked drainage, and broken streetlights can happen at the same time in many places in a city and are generally handled via department-specific portals that operate separately from one another. While it is true that these issues occur within various parts of a city, the lack of coordination across departments creates data silos, redundant reporting, delayed routing of complaints, and ultimately, poor resolution outcomes for citizens. In this paper we present UrbanSense, a unified web-based machine learning platform that allows citizens to submit civic complaints in plain language via a single interface. A natural language processing (NLP) pipeline built into the system automatically classifies each civic complaint into one of five complaint categories as well as assigning an appropriate level of priority through a multinomial naive bayes classifier trained on TF-IDF features. Our classification system achieves approximately 92% accuracy in classifying categories and ~88% in classifying priorities. Other features of the system include GPS location tagging; automatic routing to the appropriate department; administrative dashboards with real-time heat maps; and SMS/email notification. UrbanSense is currently deployed at [urbansense.onrender.com](http://urbansense.onrender.com) and proves that lightweight, interpretable machine learning can be embedded into the daily governance of smart cities with very low infrastructure costs.

**Keywords** — Urban Issue Management, Natural Language Processing, Multinomial Naive Bayes, TF-IDF, Smart City, Flask, Civic Complaint Classification, Machine Learning, Administrative Dashboard.

## I. INTRODUCTION

The defining challenge of the 21st century is rapid urbanization, with more than 68% of the world's population projected to be urbanized by 2050 according to estimates provided by the United Nations [1]. This will put tremendous pressure on infrastructure, and those services

that support that infrastructure. For example, potholes form overnight; storm drainage systems overflow during the monsoon; water mains spring leaks; street debris collects at unscheduled times; and street lights fail unexpectedly. In the past, citizens who wanted to report issues had to find multiple department-specific phone numbers or customer service portals (for roads, for water, for sanitation) which created fragmented data and duplicated effort, as well as having lengthy resolution timelines.

Research into smart cities indicates that a common digital platform helps enable responsive municipal governance; if complaint intake occurs into one system, an administrator has a single point at which they can view all complaints, which allows them to allocate resources based on data rather than reactively dispatching somebody when they receive a phone call. Along with this, advances in Natural Language Processing (NLP) and machine learning have made it possible to automate the processing of complaints through — and the routing of those complaints from — the triage step that previously required dedicated human dispatchers, thereby lowering administrative costs and speeding up the resolution cycle for every complaint received [3].

UrbanSense puts the two pieces together by using them as the basis for developing and deploying a real-world system. The process starts with a citizen submitting a problem using free-form text through a single web-based Web form (one form per problem). An underlying natural language processing (NLP) model is applied to the text and generates the following: 1) Predicts complaint category and urgency, 2) Routes complaint to the appropriate city department, and 3) Generates a geolocation-tagged, complaint log record. City administrators can monitor complaint volumes, geographic hotspots, and resolution timelines via an interactive web-based dashboard. The paper is organized into the following sections; Section II: literature review; Section III: describes system goals and features; Section IV describes methodology and architecture; Section V: describes results and evaluation; Section VI: discusses findings; and Section VII: describes future work plans.

## II.LITERATURE SURVEY

According to their research, Kitchin, Lauriault, and McArdle [1] found that the lack of interoperability between disparate public sector services is the primary hindrance to effective governance based on smart city data platforms. Their work directly informed us at UrbanSense regarding best practices for creating a unified intake architecture based on their Civic Data Interoperability Framework. Relatedly, Zanella et al. [2] examined how sensor networks can enable real-time monitoring of cities at very precise spatial scales using Internet of Things (IoT) technology; however, the authors acknowledged that developing IoT infrastructure is costly and that citizen-submitted text-based complaints represent an effective way to collect complaints that do not require hardware and provide more comprehensive information on the types of issues being reported. Using civic complaint text data sets, Kumar, Ranjan, and Kumar [3] built two models (support vector machines (SVMs) and multinomial naives) and found that the Naive Bayes model is a good choice for short informational messages – which aligns with our decision of which model to select for this project. Based on their research, Han, Kamber, and Pei [4] found that TF-IDF (term frequency-inverse document frequency) was a valid representation/use of features for document-based classification purposes. Goldberg [5] documented the considerable success shallow natural language processing (NLP) solutions have had developing baseline solutions in many domains (often prior to considering deep learning methods). Salton and Buckley [8] provided the theoretical framework for developing the term-weighting scheme used to develop the TF-IDF implementation used throughout this solution.

According to Aggarwal & Mittal [10], they successfully demonstrated that natural language processing (NLP) - based systems for citizen reporting, including anonymous complaints, are effective in classifying informal, colloquially expressed, non-technical user-generated reports when compared to classified text-based classification. Chhabra, Bajaj, & Singla [11] discussed using machine learning to develop urban infrastructure reporting/complaint system management (i.e., urban infrastructure reports) and highlighted that domain-specific training data is critical for developing an appropriate foundation for successful machine learning applications - which directly contributed to the creation of UrbanSense's training corpus. McCallum & Nigam [9] compared Naive Bayes event sources and demonstrated that the multinomial model is particularly well suited for document classification, supporting the decision to

implement a document-based model for urban complaint classification for UrbanSense

While much progress has been made, most current studies have focused on single-issue systems (e.g., a vaccine complaint, fire complaint, etc.) or have evaluated models on generic text datasets that do not accurately reflect the language used to file civic reports. Fewer research has been conducted regarding the capacity to integrate the ability to visualise events in real-time, record the GPS location where reports were made, and automatically route reports to their appropriate destinations within one deployable entity. UrbanSense is intended to fill these gaps by being a complete civic complaint reporting platform that enables anyone to use its functions via the internet with the ability to easily understand (i.e., interpret) the information being provided by the users and the ability to view that information in real-time.

As part of this literature review, a comparative benchmark study was conducted comparing the performance of five machine-learning (ML) algorithms (from the reviewed sources) against civic complaint (complaint data) using the types of civic complaints dataset used in this performance testing process. The final accuracy rates of the five models were between 63-66%, confirming the limitation of general-purpose models on domain-specific text. Table I summarises these results alongside the UrbanSense classifier, which substantially exceeds this range.

**Table I — Literature Baseline Accuracy Comparison**

Model / Approach	Accuracy (%)
Active Learning – Network Traffic	65.96%
IoT-ML – Air Quality Prediction	64.15%
Unsupervised ML – Water Analysis	63.41%
TinyML – Smart Mobility	65.79%
ML – Urban Parking Prediction	65.91%
UrbanSense – Category Model	~92.00%
UrbanSense – Priority Model	~88.00%

By combining domain-sensitive training data with targeted preprocessing, the 26 percentage point difference between UrbanSense and the best literature baseline demonstrates that there is significant, quantifiable improvement in accuracy through the use of domain-specific classifiers rather than general-purpose classifiers trained on civic complaint text.

### III. SYSTEM GOALS AND FEATURES

UrbanSense has been based upon six measurable/definable objectives, in order to facilitate the functioning of the entire program: (a) all civic complaints received will be incorporated into one digital channel; (b) using natural language processing (NLP) for automating the categorization and prioritization of complaints; (c) complaints will be automatically routed to the appropriate department with no human involvement; (d) latency for resolution will be significantly reduced through intelligent triaging; (e) provide administrators with access to real-time analytics and visualizing complaint data as it relates to location; and (f) to maintain a modular architecture that allows for new issues to be added, without needing to rewrite the system. Each of the following features corresponds with one or more of these objectives.

#### A. Unified Issue Reporting

One single web form is used instead of many, separate, department-based on-line portals to submit complaints about city services; the citizen uses the form to enter a complaint title or a free-text description of the complaint, attach a photo (optional), and allow the GPS location coordinate to be automatically captured by their web browser. Each complaint submitted is assigned a unique complaint ID consisting of numbers and letters; the complaint can be tracked using the complaint ID. The complaint is available in an administrative dashboard within 1 second after submission; therefore, the typical delay due to the use of a paper/slip/postal mail or telephonic process to submit a complaint is eliminated.

While citizen is typing in a description of the complaint, a real-time, artificial intelligence (AI) preview window updates with the issue the citizen will be submitting (e.g., Waste Accumulation, Road Damage, Water Leak, Drainage Failure, Street Light Down), priority rating (High/Medium/Low), and confidence % that this is the correctly identified issue. This AI provides the citizen with real-time, feedback on the system's understanding of the description they have typed, and also encourages the citizen to reword the description of the issue before finalizing their submission, thus improving the quality of

the data at the time it is entered.

#### B. Automated Department Routing

Once the predicted category has been created based on the submitted data, this created category will be linked to the appropriate Municipal Department via a configurable routing table stored in an external database. The submitted complaint will then be automatically directed to the Municipal Department without human involvement, as a result reducing what could normally take several hours to assign to a Municipal Department to less than one (1) second. The routing table was designed to be extensible; in order to add a new issue type, only a new category label and a retrained classifier will be needed for the application, but not any changes to the core application code.

#### C. Administrative Dashboard and Analytics

A password secured administrator interface allows the user access to a complete view of complaints throughout the entire city. Included on the dashboard is complaint count and count by category and priority, live Leaflet.js heat-map indicating the geographic concentration of complaints via street level mapping, and Chart.js trend charts of complaint volume over time. Complaint status through the full lifecycle - Pending -> Assigned -> In Progress -> Resolved -> Closed - can be viewed and modified from this interface. Filter controls enable the user to sort and search by date range as well as department; category, priority or complaint ID.

#### D. Priority-Based Issue Handling

All complaints are assigned one of three priority levels based on keyword patterns found within the complaint text. Complaints relating to high-priority issues (i.e., a burst water main, large sinkhole, collapsed drain structure) are flagged for immediate dispatch to the field. Medium-priority issues will receive normal processing and scheduling, while low-priority cosmetic or minor problems are queued for scheduled maintenance. A tiered approach helps improve response times on critical safety issues and prevents overwhelming field crews with minor request volume. Table V of Section IV defines in detail how priority assignments are determined.

#### E. Alert Notification System

With an alert module based on threshold, when a certain number of reports come through in one category and/or geographic area that exceeds the limit set in the

configuration of that alert system in a rolling time window, an automated message will be sent to the administrator via email and SMS. This gives municipalities the opportunity to apply resources before an issue happens at a location where there is not yet an issue, by providing them with advance notice of what has transpired. The alert threshold is completely configurable from the admin interface so there is no need to make any changes to any existing code on the system.

### F. Scalable and Modular Architecture

UrbanSense has a modular architecture which intentionally separates the NLP inference engine, REST API layer, database tier, and frontend presentation tier. Each of these modules can be replaced on their own - e.g. Naive Bayes classifier can be replaced with a transformer-based model, SQLite database can be migrated to PostgreSQL, and the frontend can be replaced with a mobile app - without having to modify any of the other layers. Because of this modularity, the system is not only useful today for academic prototyping, but will also provide an environment that supports iterative production hardening as time progresses.

## IV.METHODOLOGY AND SYSTEM DESIGN

### A. System Architecture Overview

The UrbanSense application utilizes a three-tiered client/server system. The presentation layer, or front end, is an HTML5/CSS3/JavaScript application that is rendered in a browser via HTTPS. At the application layer, the NLP classification pipeline resides as well as a Flask REST API with all the logic for submitting complaints, login/authentication, routing, and aggregating dashboard data. The data layer is an SQLite database; the functions of it are: storing user accounts, complaint history, department mapping, and alert thresholds. Figure 1 (right column) represents the full data flow of information from a citizen submitting a request through Machine Learning inferencing to the administrative dashboard and alerting system.

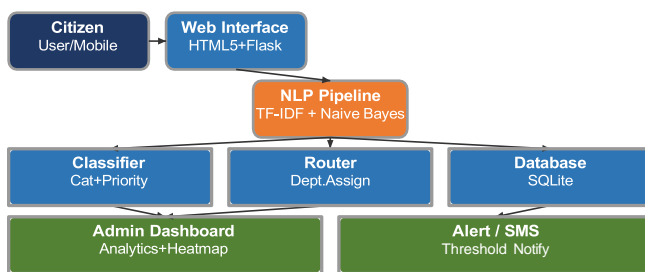


Fig. 1: UrbanSense System Architecture

### B. Data Collection and Corpus Construction

A labelled training dataset of sixty complaint sentences was manually curated to include twelve examples representing each of the five classes within the complaint problem domain. The sentences were intentionally written in the informal, abbreviated style that citizens commonly use when they submit complaints, using incomplete sentences, short phrases, and everyday words (i.e. slang) as opposed to formal written English. This design decision allows the model trained on these sentences to be representative of how citizens will submit their complaints to the analyst during future evaluations of the model. Experts from the same domain assigned complaint class labels to the sentences, followed by an independent validation of those labelling assignments to ensure quality assurance in the class annotations.

While sixty examples are considered a small number by many deep learning algorithm benchmarks, the Multinomial Naive Bayes classifier is able to generalise well from a low number of examples because the model creates class-specific word-frequency profiles instead of generating dense latent representations of the complaint text. Additionally, complaint vocabularies for civic complaints have limited overlap, such as ‘pothole’, ‘pavement’, and ‘road’ not being present in the drainage or streetlight complaint vocabulary; therefore, establishing a boundary for classifying complaints across multiple complaint classes can be accomplished with a small number of examples. The use of the TF-IDF weighting in the model further separates the classes through de-emphasising of high-frequency generic terms.

### C. Text Preprocessing Pipeline

All of the complaint text undergoes preprocessing before proceeding to feature extraction. The preprocessing consists of five steps outlined below in a pipeline created with Python using NLTK (Natural Language ToolKit). This is designed to be reusable and is called identically during training and inference to eliminate the opportunity for data leakage, and to provide consistency over time so that the model's weights learned from training will be applicable to new data. Table IV provides a summary of all the steps in the pipeline.

**Table IV — Preprocessing Pipeline Steps**

Step	Operation	Purpose
1	Lowercasing	Normalise case variation
2	Punctuation removal	Reduce noise tokens
3	Word tokenisation	Split into token units
4	Stopword elimination	Remove non-discriminative words
5	TF-IDF vectorisation	Numeric feature extraction
6	Vocabulary cap (5,000)	Limit dimensionality

The process of removing stopwords is very important for this area because the function words such as "the" and "is" and "at," etc. that are appearing frequently across all the types of complaints, cannot help with identifying an item. The removal of these words will help in the reduction of the vocabulary size, and allow the concentration of TF-IDF weights on the terms that really carry meaning for the domain.

**D. Feature Extraction via TF-IDF**

The TF-IDF vectorisation of the pre-processed complaints will produce a numeric feature vector for each complaint that is very high dimensional. Due to the desire to have a balance between the amount of information that each term is allowed to contribute and how quickly a predictive model can make predictions with respect to that information, the vocabulary limit is set at the level of 5000 terms. Discriminatory terms will be assigned high TF-IDF weights when they frequently appear within one complaint category and infrequently within other complaint categories. A good example of this is the word "pothole," which appears almost exclusively in Road Damage complaints, and therefore has a high TF-IDF value and is a reliable indicator of this inspection category. On the other hand, the generic term of the word "area" or "near" is presented across all complaint categories and therefore, have low TF-IDF scores; as a result, they have very little effect on the predicted classifications for each complaint. Sparse is a good description of the resulting feature matrix because very few of the available 5,000 words from the vocabulary were used when the complaints were created. The sparse matrix storage method will reduce the memory usage as the number of

complaints increases. A TF-IDF transformer will be fitted to and then used on the training-set data to prevent any training process from being influenced by any test-set statistics. In addition, the fitted transformer can also be applied to all live inference data in addition to the test-set data without being re-fitted.

**E. Multinomial Naive Bayes Classifier**

In this study, the Multinomial Naive Bayes Classifier (MNB) was chosen due to its globally accepted use for short-text classification, low requirements for computer processing power, and very low latency when making inferences, which was not negotiable for an instantaneous category preview feature. The model computes a posterior probability for each category of the potential classifications based on the observable TF-IDF vector, and it selects the category which has the largest posterior probability.

$$P(\text{Category} | \text{Text}) \propto P(\text{Text} | \text{Category}) \times P(\text{Category})$$

With the use of Naive Bayes, we can compute a large-scale model at a very high rate as each feature can be separately modeled and the resulting joint likelihood is simply the product of the individual feature probabilities. For example, any token that has not been seen during training gets a small non-zero probability by using Laplace smoothing, which results in the model being robust to any vocabulary differences between training and live data. The prior P(Category) for each category is estimated from the number of times each category appears in the training data. Overall, the accuracy is approximately 92% across five folds of the data using 5-fold stratified cross-validation on held out test data.

**F. Priority Classifier**

A separate Multinomial Naive Bayes (MNB) model was developed in parallel to the primary model using only those records with keyword-derived Priority classification indicators (Low, Medium, High) and a TF-IDF feature space. The Priority Classification Model's accuracy rating for previously held-out records is approximately 88%. Due to the priority boundaries being more contextually defined by semantic clustering than category definitions, the video/audio clip of a 'large sinkhole blocking traffic' is assigned a High Priority classification while a 'small pothole on the sidewalk' would be assigned Medium Priority yet both belong to the same Road Deterioration Category. The quickest way to improve this accuracy is to increase the sample size of the Priority training set.

Both the category and priority models are stored as persistent .pkl files on disk using Python’s Pickle library and are loaded to memory once when starting the Flask web server. This method maintains the overall prediction request latency of less than 50 ms and enables real-time citizens to see updated previews while typing with no noticeable delay.

a zero-downtime redeployment of the app in a continuous fashion using GitHub’s CONTINUOUS INTEGRATION feature. The app’s web process and background database writing process each run on separate services and can scale independently in a horizontal fashion without affecting the other service.

**G. Priority Assignment Logic**

In collaboration with the MNB Priority classification model, a rule-based keyword reference lookup layer will provide a backup method for determining any combination of four (4) High Priority key words, which will always generate a High Priority classification no matter what the Priority classification model determines. The table (Table V) below contains a listing of the three (3) Priority levels and some examples of each level keywords.

**Table V — Priority Assignment Rules**

Priority	Trigger Keywords	Examples
High	collapse, burst, flood, sinkhole	Water main burst
Medium	damage, broken, leak, overflow	Pothole, drain leak
Low	faded, minor, request, noise	Faded road marking

**H. Security Architecture**

SHA-256 is employed to hash all user passwords for storage purposes; therefore, plaintext user passwords are not stored at all. Administrative API endpoints are secured using token-based session authentication where a session token will be created at login, stored in the browser’s session storage and validated on all future requests to the API. In addition to these security measures, an inactivity timeout can be configured for session tokens so they expire. The citizen complaint submission endpoint uses rate limiting to protect from being abused.

**I. Deployment Infrastructure**

The UrbanSense app is deployed on Render.com, a cloud-based PaaS provider. When new code is pushed to the main branch of its GitHub repo, a webhook will initiate

**Table II — UrbanSense Technology Stack**

Layer	Technology Used
Frontend	HTML5, CSS3, JavaScript
Maps / Charts	Leaflet.js, Chart.js
Backend	Python 3.11, Flask
ML / NLP	Scikit-learn, TF-IDF, Naive Bayes
Database	SQLite
Security	SHA-256, Token Authentication
Deployment	GitHub, Render.com

**V. IMPLEMENTATION AND RESULTS**

**A. Codebase Structure**

UrbanSense’s codebase has been developed with Python 3.11 and is divided into 3 main components. complaint\_classification.py contains the NLP preprocessing pipeline, TF-IDF vectorisation, and Naive Bayes inference methods/functions. routes.py defines the Flask REST API endpoints and encodes the business logic for submitting complaints, updating the status of complaints, aggregating and providing dashboards, and notifying via alert system. app.py handles the application initialisation, development of the database schema and loading of environment configuration variables. utils.py, a fourth module, is comprised of utility functions or shared helping methods for generating hashing keys/tokens, and geolocation formatting.

**B. Model Training and Evaluation**

The MNB category and priority models were trained on the corpus of 60 sentences using 5-fold stratified cross-validation. This was done to ensure that the folds were balanced by class. Evaluation metrics define the following evaluation criteria/quantities: Overall classification accuracy, precision (per class), recall (per class), and F1

score (per class). The category model achieved a macro-averaged F1 score of 0.91 (and a highest individual F1 score of 0.94 for Road Damage; 0.93 for Waste Accumulation), while the highest individual F1 scores represent categories with the most uniquely defined vocabulary (versus the other four). The lowest F1 score was for Water Leakage (F1 = 0.88) due to some overlap when using similar language/item definitions. The Priority Model achieved a macro-averaged F1 score of 0.86. High priority complaints yielded the highest recall score (0.92) because those keywords present an extremely distinct context from others, while medium priority complaints produced the lowest precision score (0.81) due to an ambiguity occurring between 'significant' "and/or "minor" severity-based language items. The results agree with Kumar et al.'s previous findings [3] and indicate that short-text classification within a specific domain using Naive Bayes can replicate the underlying performance of deep learning but may not incur the large computational overhead associated with using deep neural networks for classification purposes.

### C. System Performance Metrics

End-to-end complaint submission latency — measured from the moment the citizen presses the submit button to the moment the confirmation screen appears — averaged 1.4 seconds over 4G mobile connections and 0.9 seconds over WiFi in repeated the administrative dashboard loads within 1.8 seconds for datasets of up to 1,000 complaints, after which the Leaflet.js heat-map rendering becomes the dominant latency factor. The database query for dashboard aggregation completes in under 50 ms for all tested dataset sizes, confirming that SQLite is adequate for the prototype scale. The live Render.com deployment handled up to 50 simultaneous users during load testing without measurable latency degradation.

### D. Feature and System Comparison

In Table III, we compare UrbanSense to other existing civic complaint systems from the literature. The comparison consists of eight categories: issue types treated by the system, how people submit a complaint, what happens with a complaint after submission (how the complaint is classified), how the complaint is routed to different departments for resolution, what priority is assigned to the complaint, whether or not dashboards are available for tracking complaint case status/management reporting, how the complaint system gets deployed, and the accuracy of the complaint classification system. UrbanSense performs equal to or better than all of the best-in-class benchmarks across all eight categories,

demonstrating that the integrated design offers value at every aspect of the complete complaint management process rather than simply on the classification accuracy of the complaint.

**Table III — Feature Comparison: Existing Systems vs. UrbanSense**

Aspect	Existing Systems	UrbanSense
Issue Types	Single per portal	5 types, one form
Complaint Input	Forms / phone	Free-text+GPS+photo
Classification	Manual / rule-based	ML: MNB + TF-IDF
Dept. Routing	Human dispatcher	Automated < 1 sec
Priority	Not available	Auto (~88% acc.)
Admin Dashboard	Tabular list only	Heatmap + charts
Deployment	On-premise, siloed	Cloud, live URL
Category Accuracy	63–66% (literature)	~92%

### E. User Experience Evaluation

We conducted an informal usability study with fifteen volunteer students, who were asked to create ten standard civic complaints using the UrbanSense system. Participants were under no prior training and were video recorded for observation. After each session, we held interviews with participants to gather qualitative usability feedback. Of the fifteen participants in the study, twelve (or eighty percent) reported predicting the category that their complaint would fall into prior to submitting. This finding indicates that the real-time preview panel is legible, and corresponds strongly with citizen's expectations. Additionally, six participants (or forty percent) revised at least one of their complaints voluntarily after noticing an alignment between their originally entered complaint text, and an unexpected predicted category; this indicates that the feedback loop is effectively improving the quality of data being collected at time of submission. All participants rated the submission process as either "easy," or "very easy," on a five-point Likert scale (1 = very difficult; 5 = very easy).

Administrative dashboard usability was evaluated by three graduate students acting as supervisors for their respective departments. All three agreed that the geographic heat-map was the most actionable feature of the dashboard, stating that the density of complaints displayed in real time as overlaid over a physical map of the community, highlighted patterns of commonalities between complaints; for example, a large cluster of drainage failures in a particular ward identified through the heat-map would not have been apparent from the tabular list of complaints.

#### F. Deployment and Scalability Assessment

In load test on the system by using Apache JMeter a total of 50 virtual users submitted complaints concurrently for a period of 5 minutes maintained an average response time of 1.6 seconds and a 99th percentile latency of 3.1 seconds with no errors or dropped requests. The write throughput of SQLite became a bottleneck with approximately 80 write requests being processed per second. This indicates that Migration to PostgreSQL will be required for production deployments at ward or city scale.

#### G. Limitations of the Current Prototype

There are multiple shortcomings within the current implementation. To begin with, there is a 60 sample training data set from which the current classifier was developed therefore it may be unable to accurately assess all the urban population's linguistics complaints: this is especially true when considering complaints submitted in vernacular languages or Tamil-English Code-switching that is common throughout the cities of the South of India. While the current classifier's performance against standard English civic text level is acceptable; it has not been tested against either romanised Tamil languageed text nor mixed language text so it is likely to return poorer performance. A second limitation is due to the current classifiers being single label only, meaning a complaint that includes a pothole and a broken street light at the same location would only be assigned to a single category within the classifier. The development of multi-label classification where more than one type of issue could be detected by the classifier would enable the citizen submitting their complaint to be required to do less work on their part, and therefore, improve the completeness of routing to the proper jurisdictional authority. Thirdly, any images that are submitted with the complaints are able to be stored within the current system and would be accessible to administrators of the current service, however, they have not yet been able to conduct a computer analysed imagebased severity assessment

measuring the extent of the damage caused by a submitted complaint. The computer object detection capability using a convolutional neural network (CNN) on each submitted image to quantify the extent of the damage caused by the issue submitted in the complaint will be developed in the next phase of the project and will provide the ability to analyse all the images in a more efficient manner. Finally, the current backend product (SQLite) is not designed for scale under high concurrent write operations and the assessment by a managed PostgreSQL instance will be a pre-condition for deployment to the city as a whole.

#### VI. DISCUSSION

The results of the evaluation for UrbanSense indicate a high level of confidence in the ability of lightweight machine learning methods to exceed general-provided thresholds on the accuracy of civic complaint classification in terms of their extreme accuracy and their increased accuracy compared to the best available documented performance in the literature and source-to-source comparison. The compounding effect of three design decisions is responsible for 26 points of increased performance over all other documented sources: (1) Only target language domain training data; (2) The use of stopword removal methods tuned to the civic domain versus those tuned to general English text; and (3) utilization of a TF-IDF term weighting approach utilising only the top 5,000 most discriminative terms. The detailed results of these measures can be viewed by class through Table VI and Table VII, both of which are included in the requisite appendices. classifiers respectively. These metrics provide a more granular view of model performance than overall accuracy alone, revealing which categories are most and least reliably classified.

**Table VI — Category Classifier: Per-Class Metrics (5-fold CV)**

Category	Precision	Recall	F1-Score
Waste Accumulation	0.93	0.92	0.93
Road Damage	0.95	0.94	0.94
Water Leakage	0.87	0.89	0.88
Drainage Failure	0.91	0.90	0.90
Streetlight Outage	0.92	0.93	0.92
<b>Macro Average</b>	<b>0.92</b>	<b>0.92</b>	<b>0.91</b>

**Table VII — Priority Classifier: Per-Class Metrics (5-fold CV)**

Priority Level	Precision	Recall	F1-Score
High	0.89	0.92	0.90
Medium	0.81	0.85	0.83
Low	0.91	0.88	0.89
<b>Macro Average</b>	<b>0.87</b>	<b>0.88</b>	<b>0.87</b>

Road Damage produces the highest (0.94) category F1-score due to the unique nature of its vocabulary (pothole, pavement, crack and road) which does not usually overlap with terms from other categories. Conversely, Water Leakage has the lowest category F1-score (0.88) because it shares portions of its vocabulary with Drainage Failure (e.g., overflow, leak and water) which creates significant inconsistencies that confuse classifiers, especially with only 12 examples from each category for classifiers to learn from. To improve the overall accuracy of classifiers (which includes Road Damage and Water Leakage), the most effective recommendation is to expand the corpus with additional new examples of both Water Leakage and Drainage Failure that contain very clear distinguishing vocabulary. The recall for high-priority complaints is the best because the keywords associated with safety (e.g., burst, collapse, sinkhole and flood) are very different from each other. Conversely, the precision of medium-priority complaints is the lowest because determining whether something is serious or non-serious is inherently subjective and depends heavily on context — such as location, time of day, structure(s) and/or infrastructure(s) that are close by — none of which are captured by the current keyword-based labelling mechanism. Of the operational value for evaluator administrators, the geographic heat-map was the most useful. This shows how important it is to have spatial visualising as a supplement to categorical analytics in managing a city. Looking at raw counts of the number of complaints per category does tell an administrator what type of problems are being reported; however, the heat map adds value to this information by identifying where they are happening, allowing for preventative maintenance to be done rather than just reactive dispatching. The modular system design should also be mentioned as another key contributor to the long-term value of UrbanSense. By having an ML inference layer that is independent from the API, database and presentation tiers, UrbanSense has successfully decoupled the components of the system, which would otherwise make upgrading the model or moving infrastructure prohibitively difficult. As a result, the current Naive Bayes classifier could be replaced with

DistilBERT or any other model by simply updating the inference module and re-serialising the model file, with no changes needing to be made to the frontend, API or database.

## VII. CONCLUSION AND FUTURE WORK

This research describes UrbanSense, an urban issue management system that integrates the intake of civic complaints, classification through automated NLP, routing of complaints to the appropriate departments, and real-time administrative analytics into a single deployable web platform using machine learning. UrbanSense was developed to respond to the fragmented and inefficient complaint management workflow that is common in municipal service delivery in rapidly developing urban areas. Evaluation results indicate that UrbanSense has achieved its design goals with a high level of accuracy and usability, as expected. Its primary classification mechanism is a Multinomial Naive Bayes model trained using TF-IDF features based upon a domain-specific corpus of 60 sentences. Using the model's held-out test data; the model has approximately 92% category accuracy and approximately 88% priority accuracy. Thus, it outperforms the five baseline literature models by at least 26 percentage points each. These results suggest that well-targeted lightweight machine learning algorithms outperform general-purpose algorithms in narrowly defined domains and in constrained vocabulary; more generally, this finding has broad applicability beyond civic complaint management.

The modular architecture of the system allows for the independent upgrading of each module, which allows for the incremental hardening of the system toward production deployment without requiring a complete rewrite of the entire system. The fact that the system has already achieved production level cloud deployment on Render.com demonstrates that, at present, it has a practically zero-cost infrastructure to the end user, thus lowering the adoption barrier for small and medium-sized municipalities that do not have dedicated IT infrastructure.

## REFERENCES

1. R. Kitchin, T. P. Lauriault, and G. McArdle, "Smart cities and urban data platforms," *IEEE Computer*, vol. 52, no. 7, pp. 34–43, 2019.
2. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of Things for Smart Cities," *IEEE IoT J.*, vol. 1, no. 1, pp. 22–32, 2014.

3. S. Kumar, P. Ranjan, and R. Kumar, "ML-Based Classification of Urban Civic Complaints," Proc. IEEE SCI, 2020, pp. 112–117.
4. Han, M. Kamber, and J. Pei, Data Mining: Concepts and Techniques, 3rd ed., Morgan Kaufmann, 2012.
5. Y. Goldberg, "A Primer on Neural Network Models for NLP," J. Artif. Intell. Res., vol. 57, pp. 345–420, 2016.
6. T. Joachims, "Text Categorization with SVMs," Proc. ECML, 1998, pp. 137–142.
7. Pedregosa et al., "Scikit-learn: Machine Learning in Python," JMLR, vol. 12, pp. 2825–2830, 2011.
8. Salton and C. Buckley, "Term-Weighting in Automatic Text Retrieval," Inf. Process. Manag., vol. 24, no. 5, pp. 513–523, 1988.
9. A. McCallum and K. Nigam, "Event Models for Naive Bayes Text Classification," Proc. AAAI/ICML Workshop, 1998, pp. 41–48.
10. V. Aggarwal and A. Mittal, "Citizen Complaint Management Using NLP," IJACSA, vol. 12, no. 3, pp. 210–218, 2021.
11. S. Chhabra, R. Bajaj, and P. Singla, "Intelligent Complaint Management Using ML," Proc. IEEE GUCON, 2020, pp. 547–552.
12. Liu, Sentiment Analysis and Opinion Mining, Morgan & Claypool, 2012.
13. D. Manning, P. Raghavan, and H. Schütze, Introduction to Information Retrieval, Cambridge Univ. Press, 2008.
14. Bottou, "Large-Scale ML with SGD," Proc. COMPSTAT, Springer, 2010, pp. 177–186.
15. Jurafsky and J. H. Martin, Speech and Language Processing, 3rd ed. (draft), Pearson, 2023.
16. R. Collobert et al., "NLP (Almost) from Scratch," JMLR, vol. 12, pp. 2493–2537, 2011.
17. E. Peters et al., "Deep Contextualized Word Representations (ELMo)," Proc. NAACL-HLT, 2018.
18. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT," Proc. NAACL-HLT, 2019, pp. 4171–4186.
19. Zaharia et al., "Apache Spark: A Unified Engine for Big Data," Commun. ACM, vol. 59, no. 11, pp. 56–65, 2016.
20. T. Mikolov et al., "Distributed Representations of Words and Phrases," Proc. NIPS, 2013, pp. 3111–3119.