International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 05 | May 2020 www.irjet.net p-ISSN: 2395-0072

Safest Route Detection Application

Krishnaraj Pawooskar, Dr Ramakanth Kumar P.

B.E. Department of Computer Science and Engineering, R.V. College of Engineering, Bengaluru, Karnataka, India
Head of Department, Department of Computer Science and Engineering, R.V. College of Engineering, Bengaluru, Karnataka, India

---

Abstract - *In this day and age where crime rates have been constantly on the rise, it becomes a necessity that a person's safety is taken into consideration. Many of the existing methods aim to deliver the shortest time consuming path, without putting much regards on the safety aspect. The usual shortest path from an origin to a destination is likely to differ considerably from a "shortest" path that is based on a preference to travel as little as possible through uncertain areas. This project aims at analyzing information to enhance awareness and security of the people. Several features along a route are monitored such as the availability of hospitals, gas stations, streetlights, police stations etc. to generate a safety score to be compared with other routes. The project implements a tree algorithm which decides the safety of a path by analyzing the set of features. It tries to present a route which can be considered safest among alternative routes available, hence addressing the security concerns. In addition, this information can be used by public vehicles to steer clear of uncertain areas and also by government for strengthening security in such areas.*

Key Words: **Safest Route, Safety Score, Safest Path, Tree Algorithm, Feature Set**

1. INTRODUCTION

The computation of shortest path has been considered comprehensively for many years. However, many scenarios are such that, the preferred path may not necessarily be the shortest one. In uncertain environments, minimizing distance travelled through risky regions, can prove to be lifesaving. Consider for instance, a person who is on a long journey through the desert. He may attempt to travel through villages or safe areas he considers, in the desert as a breakdown can be life threatening in such a deserted region.

In view of a familiar situation, a tourist planning to walk to a particular destination would favour a path that visits welcoming streets or street blocks, e.g., with attention-grabbing galleries, houses, or other sights, as much as possible. Here, traveling in regions ("safe areas") is simply preferred, as the visitor is unlikely to select the shortest route if it comes at the cost of his own safety.

As a result of increased incidents of criminal activities, it becomes a necessity that the safety of the people is considered. Shortest path finding has been a prevalent research area in the past few years. Nevertheless, we are unaware of efforts to explore the problem of finding the safest path via safe areas or safe routes. Only a few existing readings consider the safety feature.

The project aims to develop a web application which will provide the safest path information of the major areas in the city of Bengaluru. It does so by considering various features like the number of hospitals, accident data and other factors which contribute towards safety features. It analyses multiple routes between two locations and generates a safety score which is used to deicide the best route which can be considered the safest among alternative routes available.

Since most of the people already use maps for navigation, this project will add on to the safety benefits as it will provide information which the user can keep in mind while deciding his route via maps. The results of this algorithm can also be used by various other safety related applications.

2. PROPOSED APPROACH

The project is developed as a web application with React.js being used to develop the user interface. Flask framework is used to develop the backend which houses the actual algorithm which is used to compute the safest route details. The process of using the user input data to generating the result takes place in the following approach. This can be visualized with the help of a structure chart shown in Fig-1.

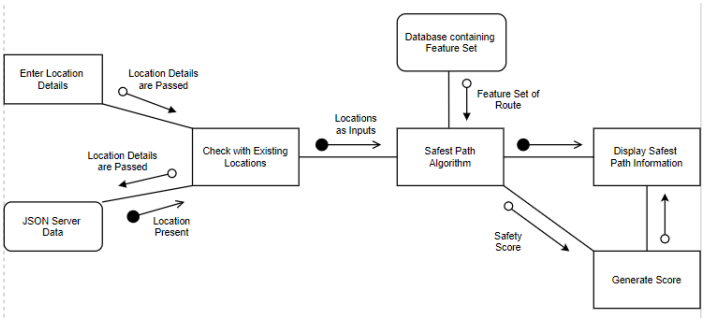

Fig-1: Structure Chart

A. Fetching User Input

The user has a choice of selected locations on the interface, any two of which can be chosen from as the source and destination. The locations are stores on a JSON server as a JSON file and can be updated as needed. The user does this by typing his choices in a search box. User provides input in the form of two locations On submitting the form, his responses are fetched and sent to the backend for its usage. The user interface is developed using React.js framework which is efficient for applications with lesser webpages.

B. Assign Unique ID to Input Choices

Every input choice made by the user is associated with a unique ID. This is necessary for efficient computation and easy retrieval of the related feature dataset. T

C. Retrieve Relevant Feature Dataset

A set of features exist between any two locations. These contain information such as the number of hospitals, police stations, gas stations, streetlights, accident data, and many such related data as represented in Table-1. It is stored in the form of a file. The feature set data can be queried relevant to the IDs of the location. There may be multiple routes between two locations, in which case every path within the route has its own feature set which is unique to it.

Table -1: Sample Feature Set

Feature Set			
Feature	Occurrences	Intensity	Threshold
Hospitals	2	10	10
Police Station	1	10	10
Gas Station	2	8	8
Streetlights	175	0.05	12
Accident	4	- 2	- 10
Theft	3	- 4	- 10

D. Evaluation of Safest Route

The user input consisting of source and destination locations along with the relevant feature set is fed into the algorithm for computing the safest path information. The algorithm uses the necessary feature set data to generate a score of the different paths within a route. The path which falls below a certain score is rejected and the other path are compared to find the safest out of them.

3. IMPLEMENTATION

The logical flow of data of the entire application is shown as below in Fig.-2. At first, the user enters his input from a list of available locations as displayed on the user interface. The input is fetched and sent to the backend where the feature set relevant to the input will be retrieved. This is then passed on to the algorithm for computing results.

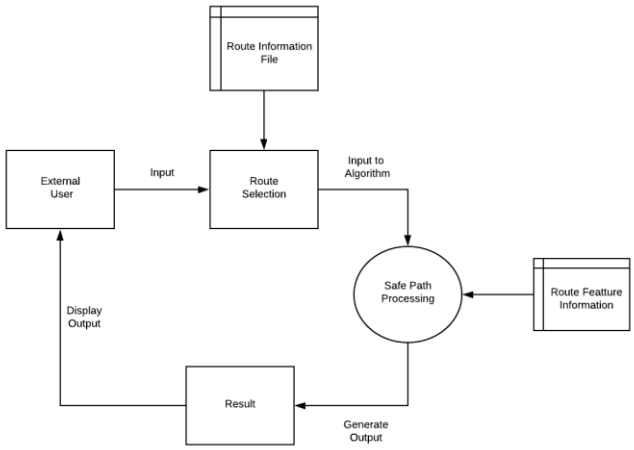

Fig-2: Data Flow Diagram (Level 1)

Working of Algorithm

The algorithm is developed to generate a decision tree. There may be multiple possible routes between a particular source and destination. For each possible route, its specific feature set is used, shown as an example in Table - 1. The algorithm works in the following way.

1) Calculation of Feature Score

For each path say 'i' of a particular route, feature score for each feature 'j' in that route is calculated as follows.

$$(\text{Feature Score})_j = \text{Occurrences}_j * \text{Intensity}_j$$

$$(\text{Route Score})_i = \sum_{j=1}^{\text{total}} (\text{Feature Score})_j$$

Each feature score has a corresponding threshold value below which it is considered unsafe.

2) Representation as Tree Structure

For each route 'i', the features are depicted in a tree format. The feature having highest absolute value of feature score is chosen as the root node.

Root node = Feature = max ($(\text{Feature Score})_j$) for all j.

All the subsequent features are added to the tree in decreasing order of their absolute feature scores. The representation of features in a decision tree format is depicted in Fig-3.

3) Classification of Street

Starting from the root node, every feature is compared against its threshold value which is unique to every feature. Features that have their feature score higher than the threshold, that feature is classified as safe within that route.

When two or more features are considered unsafe within a particular route, that route is considered not very safe for travel. The route score is maintained until the safest path result is generated.

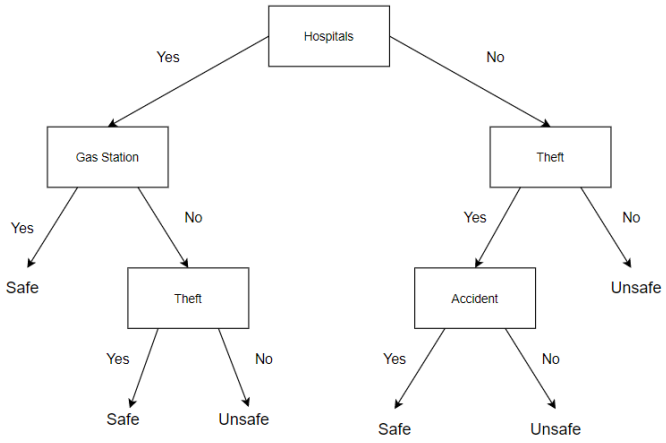

Fig-3: Sample Representation of Feature using Algorithm

4) Safest Route Decision

The routes which are determined safe after decision tree comparison are considered. The route having greatest feature score is selected.

Safest Path = Route = max ((Route Score) $_i$) for all i.

The same decision is applied in case the algorithm decides all routes are unsafe. This then outputs the safest route among the unsafe.

4. RESULT

The result generated from the algorithm in the backend is then sent to the web application and displayed on a webpage. It consists of the user input and the safest path information relevant to it. Fig-4 shows the demo output displayed to the user. It represents information related to the safest path between two locations.

The result is computed with the aid of the algorithm which makes use of the user input for its computations. It analyzes multiple route between two locations to output the safest one to the user and thus addresses to his safety concerns.

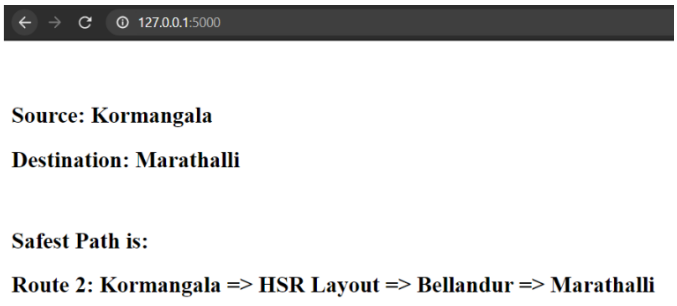

Fig-4: Demo Output

5. LIMITATIONS

Due to vast and complicated nature of the project, there are certain aspects which could not be incorporated. The limitations of the system are as mentioned.

The project in its current state is only limited to the city of Bengaluru and the major areas constituting it. It aims at providing only the safest route information to the user. It does not provide the actual route. The user can then use this information along with other application like Maps to navigate safely. Also, the system does not update the feature related dataset information automatically. The changes have to be made manually. Furthermore, the available route information between two areas also have to updated manually in case of detection of new paths.

6. CONCLUSION

This report discusses about developing features in web application which will enable users to travel between two locations by giving priority to their safety concerns the ultimate goal of the application is to provide to the user, the safest route from source to destination also mentioning the major threats along the suggested route. In this day and age where crime rates have been constantly increasing, this application proves as a necessity by trying to address the safety aspects of the common people, and thus provides better alternatives in doing so.

ACKNOWLEDGEMENT

I would like to extend my gratitude towards my guide Prof. Ramakanth Kumar P. for his continued support and guidance towards my work in this project and paper. I would also like to extend my gratitude to the Department of Computer Science and Engineering, RV College of Engineering for giving me the opportunity to write this paper with complete support.

REFERENCES

[1] Bahn, H.: Web cache management based on the expected cost of web objects. Information and Software Technology 47, 609-621 (2005)

[2] D. Kornack and P. Rakic, "Cell Proliferation without Neurogenesis in Adult Primate Neocortex," Science, vol. 294, Dec. 2001, pp.

[3] Bogardi-Meszoly, A., Levendovszky, T.: A novel algorithm for performance prediction of web-based software system. Performance Evaluation 68, 45–57 (2011)

[4] Domenech, J., Pont, A., Sahuquillo, J., Gil, J.A.: A user-focused evaluation of web prefetching algorithms. Journal of Computer Communications 30, 2213–2224 (2007)

[5] Georgakis, H.: User behavior modeling and content based speculative web page prefetching. Data and Knowledge Engineering 59, 770–788 (2006)

[6] Basili, V., and Rombach, H.D., 1989, The TAME project: Towards improvement-oriented software environments. *IEEE Transactions on Software Engineering*, **14**(6): 758–773.

[7] S. Deerwester, S. Dumais, T. Landauer, G. Furnas, and R. Harshman, "Indexing by Latent Semantic Analysis," J. Am. Soc. Inf. Sci., vol. 41, no. 6, pp. 391–407, 1990.

[8] J. R. Quinlan, "Induction of Decision Trees," Mach. Learn., vol. 1, no. 1, pp. 81–106, 1986.

[9] Ittai Abraham, Daniel Delling, Andrew V Goldberg, and Renato F Werneck. A hub-based labelling algorithm for shortest paths in road networks. In SEA, pages 230–241. 2011.

[10] Robert Geisberger, Peter Sanders, Dominik Schultes, and Daniel Delling. Contraction hierarchies: Faster and simpler hierarchical routing in road networks. In SEA, pages 319–333. 2008.

[11] Christina Hallam, KJ Harrison, and JA Ward. A multi-objective optimal path algorithm. Digital Signal Processing, 11(2):133–143, 2001.

[12] HV Jagadish, Beng Chin Ooi, Kian-Lee Tan, Cui Yu, and Rui Zhang. idistance: An adaptive b+-tree based indexing method for nearest neighbour search. TODS, 30(2):364–397, 2005.

[13] Ittai Abraham et al. "A hub-based labeling algorithm for shortest paths in road networks". In: SEA. 2011, pp. 230–241.

[14] Boris V Cherkassky, Andrew V Goldberg, and Tomasz Radzik. "Shortest paths algorithms: Theory and experimental evaluation". In: Mathematical programming 73.2 (1996), pp. 129–174.

[15] Ke Deng, Xiaofang Zhou, and Heng Tao Shen. "Multi-source skyline query processing in road networks". In: ICDE. 2007, pp. 796–805.

[16] C. Hölscher, T. Tenbrink, and J. M. Wiener, "Would You Follow Your Own Route Description? Cognitive Strategies in Urban Route Planning," Cognition, vol. 121, no. 2, pp. 228–47, Nov. 2011.

[17] T. Meilinger, J. Frankenstein, and H. H. Bülthoff, "Learning to Navigate: Experience Versus Maps," Cognition, vol. 129, no. 1, pp. 24–30, Oct. 2013.

[18] Y. Xu, Z. Wang, Q. Zheng, and Z. Han, "The Application of Dijkstra's Algorithm in The Intelligent Fire Evacuation System," in 4th International Conference on Intelligent Human-Machine Systems and Cybernetics The, 2012,

[19] N. R. Shankar and V. Sireesha, "Using Modified Dijkstra' s Algorithm for Critical Path Method in A Project Network," Int. J. Comput. Appl. Math., vol. 5, no. 2, pp. 217–225, 2010.