Project Title: A navigation tool to compute the best route based on road safety, time of journey and weather conditions.

Project members:
1) Abhijit Katkar
2) Mariya Ali
3) Setu Shah

Course: ECE 57000 – A Programming Language for Artificial Intelligence.

Semester: Fall 2016.

Introduction:

Current navigation tools rely solely on real-time traffic conditions for computing the fastest route between point A and point B. Crowd-sourced traffic data is used to give the users the quickest route. In some cases, the route is modified in the middle of the journey to reduce the travel time, based on newly formed congestions somewhere ahead in the route. However, the time of journey as the only factor of consideration is not a sufficient metric in today's times.

Each day, about 104 people in USA die in car accidents, i.e. over 38,000 deaths every year [1]. More alarming is the statistic that 4.4 million others are injured in crashes every year. While roads today are safer because of advancements in vehicle safety equipment, there still exists a lot of scope for improvements. Human error in judgment is the cause of most accidents.

The changing weather affects road safety, too. Extreme weather conditions like tornadoes, hurricanes, storms and lightning cause distractions and affect the judgment of the drivers. Similarly, accumulated snow, black ice, snow storms, etc. can cause serious problems while driving [2]. These calamities affect the condition of the road adversely leading to accidents and in turn, increase in fatality rates. The time of the day when the journey is undertaken and extends into, also affects road factors like visibility, which increase the risk of driving.

Parameters like road safety, weather conditions and time of travel must all be considered for computing the best possible route. Drivers can be given various options of routes with detailed information about what to expect in terms of both time of travel, weather conditions and safety of the occupants.

The data set for this project was obtained from National Highway Traffic Safety Administration's Fatality Analysis Reporting System (FARS) [3]. FARS provides data for accidents that caused fatalities from 1975 to 2015. The data set contains information on the date, time, location, weather conditions, vehicular details, etc. of the accidents that resulted in fatalities, in all 50 states of USA.

The implementation of the solution involves specifying characteristics that are being used in designing the algorithm. Determination of the factors of importance and setting their priority was one of the first steps. Using the available datasets to generate a pattern that can be observed will be the next step.

The project will use Google Maps APIs [4] to identify the routes suggested and compare them to suggest the safest route. This will ensure that the route taken is among the quickest, based on current traffic conditions, while not compromising on the safety, predicted based on observed patterns.

In the coming future, this intelligence will help those traveling in self-driving autonomous vehicles avoid roads which are notoriously susceptible to human errors.

Related Work:

Miaou et al. (1993) gives a statistical approach for traffic accidents analysis based on these regression models: Linear Regression Model (R1), Multiplicative Linear Regression Model (R2), and two Multiplicative Poisson Regression models – P1 and P2 [5]. The limitation of the regression model is that the derivation is sensitive to the information such as the road width, road length, type of vehicle, etc. The dependency on the variance of the data makes these models less usable for different datasets.

Islam et al. (2008) identifies the factors which cause road accidents based on the general information, pre-crash information, in-crash information, post-crash information of the case [6]. The method used in this research is based on manual observations and conclusions, and is thus prone to errors.

Lee et al. (2012) adopted a structural equation model (SEM) to capture complex relationships between exogenous variables like road, driver, environment factors and endogenous variables, like accident size [7]. The goodness of fit statistics showed that the Chi-square value is high (959.719) and the Root Mean Square Error Approximation (RMSEA) and Root Mean Square Residual (RMR) are greater than 0.05 which is not a good fit. But comparative fit index (CFI) and adjusted goodness of fit index (AGFI) both fall in the range of 0 to 1. This model proves to be inefficient in case of larger data sets.

Jadan et al. (2014) talks about the use of multi-layer artificial neural networks (ANN) to predict the occurrence of accidents in Jordan [8]. The input and output hidden layers were computed using tan-sigmoid and linear transfer functions respectively. While this method predicts accidents close to the number of total accidents, the error can be further reduced by adding more input variables, more hidden layers or a combination of both. This paper doesn't distribute the accidents depending on the location data, which makes it a rough estimate of the total accidents in Jordan.

Chong et al. (2004) studies the patterns involved in crashes of vehicles and compares various prediction models to predict the severity of injuries that may occur [9]. The paper compares the techniques of neural networks trained using hybrid learning approaches (HLANN), decision trees (DT) and a concurrent hybrid model involving decision trees and neural networks (DTANN). The paper concludes that the DTANN approach is best suited for prediction of injury. The paper falls short of explaining why SVM have poor accuracy values. Also, the inclusion of more input data points would've helped in producing more accurate results.

Ali et al. (2012) compares the artificial neural network (ANN) technique and its efficiency in predicting traffic accident casualties in Sudan [10]. The output calculated by the ANN was found to be within 1.84% deviation of the actual observed value. While the outcomes obtained are favorable, the biggest limitation of the study is the lack of usage of any significant driving-related factor like the driving style, condition of the road, the location of the accidents, etc.

Zheng et al. (2011) proposes a technique to combine parts of fuzzy logic models (FLM) and artificial neural networks (ANN) to create a fuzzy neural network model (FNNM) for the purposes of predicting accident frequencies in the Harbin city of China [11]. FNNM was observed to provide a lot more flexibility and accuracy. Also, FLM and NNM were found to be slower and more time consuming when compared to the FNNM approach.

Bae et al. (2012) investigates the factors affecting traffic severity by taking data heterogeneity into account [12]. The authors separated classes (causes or types of accidents) and provided a comparative analysis within the classes. Another approach would be to correlate the classes and find out the worst or best scenario, relatively.

Çodur (2015) predicted accidents on the highway stretches by using artificial neural networks (ANN) [13]. Since neural networks are adaptive, they were used to model the system and the disturbances in the data that comes with it. The success of ANN as a prediction model was determined by the low values of mean square error and root mean square error, and high values of the correlation and closeness of fit. Since this study considers various data points that are both related to human driving and environment, it makes the study more applicable than a theoretical one which focuses only on environmental or socio-economic factors.

The reviewed papers have either used algorithms that are not proficient enough to compute huge amounts of data simultaneously rather than sequentially or do not incorporate essential factors for accident analysis. The various approaches used in the fields of statistical analysis or by the use of computational algorithms, both, have their own set of advantages and disadvantages. The pre-processing of the dataset by the use of clustering or regression methods appears to produce better results than unprocessed datasets [5]. The accuracy of the artificial neural networks in accurately predicting future accidents based on past trends is well researched and articulated [8, 10, 13].

Methodology:

Data Set:

The data set [14] available consists of 52 attributes with information about the location of the crash, the number of vehicles involved, the number of people involved, the time of the crash, the time of the notification, the arrival time of emergency services, the weather conditions, the events that led to the crash, etc. These 52 attributes were reduced to 9 input attributes by eliminating those attributes which were either providing duplicate information or those which weren't relevant for this project. The attribute crash factors, which gave the reason for the crash as understood by the investigating officer, had missing information in over 50% of the cases recorded, and was thus also removed. Attributes like state, county, city, route, traffic way ID, mile point were eliminated as the latitude and longitude where the accident took place provide more accurate information about the location of the crash.

Out of 32,166 crashes from the year 2015, 192 were a result of drunk driving and were removed. 474 others that were in work zones were removed too, since they do not define long term characteristics of the route or region. Entries with missing location data (1911) were removed next. Similarly, entries with missing data about the weather (191), light conditions (40), time of the crash (93) were removed. Entries with missing data about the events that led to the crash (8501) were removed. Since the usage of private roads is restricted, the instances of crashes which took place on those (1658) were also removed. The same procedure was followed to reduce the data from the remaining years (2011-2014). At the end, a dataset containing 95, 955 data points was obtained.

While data about the first harmful event that led to the crash had been initially included, it was noticed that data about harmful events that lead to an accident cannot be a factor used for predicting the best route to be taken. Harmful events only describe the cause of an accident that has already occurred and while they are very helpful in analyzing accidents, they cannot contribute in the dynamic routing process.

Time (hours and minutes), date (date and month) and day of the week of the accident were also included as input variables. These inputs were pre-processed by using the hyperbolic tan function to spread them evenly between -1 and 1.

Thus, out of 52 available attributes, 9 attributes were chosen for the purposes of this project. The dataset provides numbers that indicate the presence of various factors, but not their degree of severity or how they affect the cause of the accident or the fatality. To make these attributes usable, some of these short-listed attributes were divided into multiple extra input variables. The 'light conditions' attribute was divided into 2 input factors, indicative of daylight and darkness, respectively. All values are assigned between -1 and 1 for conditions like day, night, dark lighted and dawn or dusk. The weather conditions were spread over 5 input variables. One input variable each was dedicated to describe the clear sky, rain, snow and cloudy weather conditions, and one last variable was used to define extreme weather conditions. This helped in describing all possible weather conditions appropriately. The value for input factor which is true is set to 1 and the rest are set to random values between 0 and -1.

The regions in which the accidents take place are separated by their latitude and longitude approximations. Each tenths decimal place of the latitude and longitude values corresponds to a distance of 11.12km, thus dividing the data set into regions of approximately 121 sq.km. each. These coordinates are also processed and brought in the range on -1 and 1, when they are input to the neural network.

The data about the number of people involved in the accident and number of deaths that occurred have been used to find the fatality rate related to that accident. The fatality rate is the output of this project. 90% of the dataset is used for training and 10% for testing.

Current location data, possible routes to the destination and the time of each route is provided by the Google Maps API [4]. Current weather conditions used in the algorithm are provided by the OpenWeather API [16]. The light conditions are predicted based on the time of the day and a deviation of one hour from the sunrise and sunset times for that location, obtained from the weather API [16]. We assume that one hour before sunrise is dawn and one hour after sunset is dusk.

Training and testing:

The training algorithm focuses on the calculation of weights for different input factors. Here, 9 input factors with 14 input nodes are used as inputs to a multi-layer back propagation neural network [15]. These inputs consist of weather conditions, light conditions that led to the accident, day, date, month, time (hour and minutes) and the location of the accident. The output nodes are low or high fatality rate. During the training, the calculated output fatality rate is compared to the target value, and the difference is used to adjust the weights throughout the neural network. This process is repeated for multiple epochs.

Initially, the training algorithm was divided into 2 parts. First part in which the weights were generated represents how each factor influences the fatality rate on a location un-aware basis and the second part included the location for training. This was being done to further optimize weights for each region but it was noticed that training in two parts was not required and the location data can be included in the first part itself. The weights thus generated are used for the testing and calculation of fatality rate in the algorithm described below.

The testing dataset is also used as the verification dataset. It is used to calculate the deviance from the target values and the accuracy of the results generated from the obtained values of weights.

Algorithm:

The algorithm of this project compares the various possible routes suggested by a navigation tool to travel to location B from location A.

Input: Start and end location of the journey.

Output: Comparison of the 3 quickest routes based on estimated fatality rate, time of commute, and the expected weather conditions.

1. Input the start and end location of the trip.
2. The latitude and longitude of the start of the trip and the end of the trip, and 3 possible route options are obtained from the Maps API.
3. Each possible route is divided into various sections based on the turn-by-turn guidance provided in the input.
4. For each section, the fatality rate is calculated by feeding the input variables (location of the start of that section, the estimated time of start at that section, weather conditions at the time of the query) to a multi-layer back propagation neural network.
5. For routes that consist of long lengths of travel without a turn, such sections are further divided into sub-sections of 11.12km and fatality rate of each sub-section is calculated and added.
6. The estimated fatality rate for each section of the route is added to calculate the total fatality rate of the route.
7. This is repeated for each possible route.
8. The fatality rates calculated are compared to the fatality rates obtained for other possible routes and a comparative result consisting of safety level, time of commute, and expected weather conditions, for each of the possible routes is provided.

Implementation:

During the training phase, a back propagation neural network of 14 input nodes, 1 hidden layer of 5 nodes and an output layer of 2 nodes, is used. The first 7 input nodes represent the date (1-31), month (1-12), day of the week (1-7), time in hour (0-23), time in minutes (0-59) and the location of the accident (latitudes and longitudes). The following 2 input variables denote the light conditions at the time of the crash and are scaled inputs on a scale of -1 to 1, depending on the time. Light condition for the time between sunset and sunrise is low light and the light between the sunrise and sunset is bright or normal light. Dusk is given an input value of 0.75 for darkness and 0.25 for light, whereas dawn is given a 0.5 value for both. The last 5 input nodes are representative of the weather conditions at the time of the crash. The target value, fatality rate, is obtained from the dataset as a ratio of the number of fatalities to the number of people involved in the crash. Since more than 40% of the accidents resulted in a fatality rate of 1, the target value for the training is set as high or low by setting 0.95 as the threshold for fatality rate and spread over 2 output nodes representing low fatality rate and high fatality rate.

First, the neural network is created and a bias input with value 1 is considered for both the hidden layer and the output layer calculations. The weights are initialized randomly to positive and negative values between -1 and 1. The training process is initialized and for each set of input variables, the training loop is carried out, recalibrating the weights at the end of each step. The recalibration process takes place by propagating the corrected values for each weight backwards in the network, using standard back propagation formulae. The hyperbolic tan activation is applied to the hidden and the output layer and the inputs are normalized using hyperbolic tan function as well, thus making them continuous.

Different training and testing sets were created from the reduced dataset by implementing 10-fold cross-validation. The training algorithm was run for ten thousand epochs over the training dataset to find weights. The testing algorithm set these weights on the network and was run over the testing dataset.

For the final part of the project, route and weather details are retrieved from the internet by using Google Maps API and OpenWeather API respectively. For connecting the program to the internet and fetching information with the help of the HTTP protocol, the 'curl' [19] library is used. This library fetches the

XML data as a string of information, which is then stored in an XML file. For reading an XML file, the 'tinyxml2' [18] library was used. This library assists in reading values stored between XML tags and to read the values stored in attributes of XML tags. With the help of the documentation provided on the OpenWeather API [16], the identified weather conditions are given the appropriate values before using them as inputs in the network.

The user enters the source and the destination location of travel. These details are appended to the API fetch URL. The latitude and longitude of the start and end location is retrieved for every step in the route. The route is divided into steps, and the start and end latitudes and longitudes are passed to the neural network. For every step, the weather inputs are updated from the OpenWeather API. The sunrise and sunset information is also retrieved from the OpenWeather API, and used to categorize the light conditions. The current date and time are obtained by using the 'time.h' library [20]. For steps of the route that are longer than 11km, the process of setting weather, light and time conditions is repeated for every 11km distance until the end of the step is reached. Once the inputs and weights are set, the network is run with these values to return the comparison of the three routes based on their safety values.

Results:

Initially, the binary sigmoid activation function was applied to the hidden layer and the output layer. It was observed that the hidden layer output quickly rises to large values as some input values were integer values greater than 10. This made the training process more difficult.

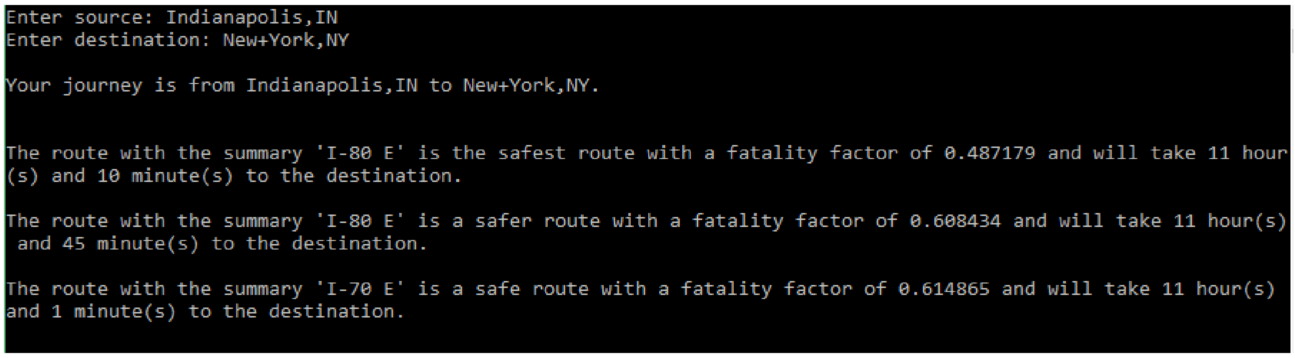
The initial results achieved were found to be 15-16% accurate. To improve the accuracy, the 10-fold cross validation was implemented by shuffling the data set and creating separate training and testing sets. The learning rate at this point was 0.1. It was observed that the accuracy at the end of testing all the various cross-validated datasets was around 25%. For obtaining better accuracy, the inputs were normalized using the hyperbolic tangent function which removed the discreteness and made the inputs continuous in the range of -1 and 1. The hidden and output layer activation functions were also changed to hyperbolic tangent function and the learning rate was set to 0.01. The result achieved after processing the data this way was observed to be around 28-30% accurate.

Since almost 40% of the fatality rate values were greater than 0.95, the threshold for the high output node was changed to 0.95 (from the initial value of 0.5), which improved the accuracy to 37%. This was the accuracy when both high and low output node values were compared with the target value. Out of the two output nodes, the high output node is the one that is used in this project. On taking only the high output values into consideration, an accuracy of 57% was obtained for the learning rate of 0.01 when the network ran for 1,000 epochs. The accuracy increased slightly to 60% when the network was run for 10,000 epochs.

The final output is displayed as a summary of the route, with the fatality factor (ratio of high fatal accident probability to total steps in the route) and time taken to reach the destination via that route. Higher the fatality factor, higher is the possibility of a fatal accident over that route. Hence, higher the fatality factor implies a comparatively unsafe route.

Table 1: Comparison of the results obtained for different test cases.

Training set parameters	**Accuracy**
Same training and testing set	16%
Shuffled training set with different training and testing sets	25%
Shuffled training sets with 10-fold cross validation	26%
Changing the activation function to hyperbolic tangent from sigmoidal for the output layer and softsign for the hidden layer for learning rate 0.1	28%
Changing the activation function to hyperbolic tangent from sigmoidal for the output layer and softsign for the hidden layer for learning rate 0.01	31%
Making the inputs continuous by using the hyperbolic tangent function, learning rate 0.01	33%
Changing the output threshold to 0.95 from 0.5, 1000 epochs, learning rate set to 0.01	37%
Verifying the result of the neural network for high output node only (with 0.95 as threshold), 1,000 epochs, learning rate 0.01	56%
Verifying the result of the neural network for high output node only (with 0.95 as threshold), 10,000 epochs, learning rate 0.01	60%



```
Enter source: Indianapolis,IN
Enter destination: New+York,NY

Your journey is from Indianapolis,IN to New+York,NY.


The route with the summary 'I-80 E' is the safest route with a fatality factor of 0.487179 and will take 11 hour
(s) and 10 minute(s) to the destination.

The route with the summary 'I-80 E' is a safer route with a fatality factor of 0.608434 and will take 11 hour(s)
 and 45 minute(s) to the destination.

The route with the summary 'I-70 E' is a safe route with a fatality factor of 0.614865 and will take 11 hour(s)
and 1 minute(s) to the destination.
```

Figure 1: An example output of the program is shown. The starting point is Indianapolis, IN and the destination is New York, NY.

As can be seen from table 1, changes made throughout the training process continued to give better results. Also, once we change the metric of calculation of output to comparing only the high output node values (since that is the one we use in the program), it gives us a much better result. It implies that while getting both the low and high output nodes to their correct expected values is tougher, the trained neural network can predict a high fatality of accident, on the route, with much more accuracy.

From the figure 1, it can be seen that the application gives an output with a fatality factor calculation for each possible route, which is denoted by its summary. The application also displays the time to reach the destination for comparison. The route through I-70E is the fastest route, but it is also the one that has the maximum possibility of a fatality, on a weekday night, when leaving from Indianapolis around 10:15 PM (EST).

The program was run for various combinations of major American cities and for routes within cities. It was observed that in some cases, the fastest route turns out to be the safest route as well, whereas in other cases (like the one in figure 1), it may even be the worst possible route.

Conclusion:

Traffic accidents are unpredictable, and thus it is impossible to predict them with complete certainty. This project aims to generalize and localize them based on past patterns of fatal crashes and weather conditions. While previous work focuses on the analysis of accidents and crashes, this project proposes a system that implements this analysis and applies it to compare multiple route options and present them to the user in a manner that would be actionable.

This project proposes a navigation tool that suggests driving routes to the user that are not only fast but also safe. The algorithm learns from data about fatal road accidents over 2011 to 2015 to predict the fatality factor of each route. Routes are compared based on weather conditions, fatality rate and the time of journey. The application suggests the routes based on estimated fatality rate, weather conditions, and the time of travel.

For this project, we used the back propagation neural network with a hyperbolic tangent activation function that was tweaked for each input factor in the input layer, the hidden layer and the output layer. Using the back-propagation neural network for training, the weights were obtained after 10,000 epochs. The learning rate was reduced by 10% after every 1,000 epochs. The weights thus generated were used to run the network in the application phase.

The data fetched from Google Maps API about the possible routes between the source and destination is forwarded to the OpenWeather API, and current weather conditions, location and time of each segment of the route are set as inputs. When applied to these live inputs, the neural network gives us expected fatality rate as low or high, which is used for further calculating the fatality factor of the route. The fatality factor gives the user a metric to compare the routes and displays the duration of travel alongside the summary of the route.

Thus, the goal of this project to create a system that can help in reducing fatal accidents from occurring by warning users and giving them alternate options, ahead of time, was achieved.

Future work:

The future scope of this project is vast. While the accuracy obtained is within an acceptable range, an implementation of the same network by using genetic algorithms [21] or Long Short-Term Memory (LSTM) [22], could be done to compare if they give better accuracy. Similarly, continuous learning can be implemented to this network so that it updates regularly based on new fatal accident data, and in the future, doesn't rely on an obsolete dataset. In this project, only fatal accident data was used. A dataset that contains information of all fatal and non-fatal accidents can instead be used.

Our approach considered the data of fatal traffic accidents occurred in US. Traffic accidents data from different countries and continents can be considered to enable the use of this application in the whole world and make it a global navigational tool.

As the predicted weather conditions were not accessible with a free license of the OpenWeather API, we used the current weather conditions to calculate the fatality rate at the start of the trip for the whole journey. While it doesn't affect the outcome much for short journeys, for longer journeys, the weather conditions might change by the time the user arrives at that location, giving us the incorrect expected results. To avoid this and make the results more accurate, predicted weather conditions can be obtained by using a premium OpenWeather API key.

Since this project calls the weather API for each step of the route, there are multiple calls for some short sections of the routes, too. This leads to a long execution time of the program. Execution time improvements could be made by optimizing the weather calls further. Also, routes which pass through extreme weather conditions for certain sections could be entirely skipped.

A formula that can correctly calculate the result of a trade-off between fatality factor, time of journey and expected weather conditions can be devised. This will then suggest a single route to the user that would balance between the fatality factor, time of journey and weather conditions prevailing on the route.

The application is at the command prompt level which can be implemented on other platforms like Android and iOS. In such an app, information about the driver and their past driving record, along with a continuously learning network, could be used as a factor in providing more personalized results.

In the future, when the transition to autonomous vehicles occurs and there are both human drivers and autonomous drivers driving cars on roads, this system can be extended to help an occupant of a driver-less car direct the car to take the safest route to the destination. This would result in better safety of the occupants of driver-less vehicles, despite probable human errors.

More results:

Results of running the program for various combinations of source and destination.

Table 2: Comparison of the results obtained for a journey from Indianapolis, IN to Chicago, IL.

Sr. No.	**Summary of route**	**Fatality Factor**	**Time of journey**
1	I-65 N	0	3 hours and 29 minutes
2	I-65 N	0.02	3 hours and 3 minutes
3	I-74 and US-41 N	0.0392157	3 hours and 42 minutes

Table 3: Comparison of the results obtained for a journey from Miami, FL to Nashville, TN.

Sr. No.	**Summary of route**	**Fatality Factor**	**Time of journey**
1	I-95 N	0.0368421	13 hours and 40 minutes
2	Florida's Turnpike and I-75 N	0.047619	14 hours and 2 minutes
3	Florida's Turnpike and I-75 N	0.0552486	12 hours and 55 minutes

Table 4: Comparison of the results obtained for a journey from New York City, NY to Washington DC.

Sr. No.	**Summary of route**	**Fatality Factor**	**Time of journey**
1	I-95 S	0.101695	4 hours and 15 minutes
2	NJ Tpke S and I-95 S	0.105263	3 hours and 54 minutes
3	I-95 S	0.129032	4 hours and 27 minutes

Table 5: Comparison of the results obtained for a journey from San Francisco, CA to Los Angeles, CA.

Sr. No.	**Summary of route**	**Fatality Factor**	**Time of journey**
1	I-5 S	0.910112	5 hours and 40 minutes

Table 6: Comparison of the results obtained for a journey from Seattle, WA to Las Vegas, NV.

Sr. No.	Summary of route	Fatality Factor	Time of journey
1	I-84 E and I-15 S	0.772727	17 hours and 48 minutes
2	I-84 E and US-93 S	0.872727	16 hours and 45 minutes
3	US-95 S	0.894737	17 hours and 41 minutes

Table 7: Comparison of the results obtained for a journey from Dallas, TX to Austin, TX.

Sr. No.	**Summary of route**	**Fatality Factor**	**Time of journey**
1	I-35E and I-35 S	0.0681818	2 hours and 56 minutes
2	I-45 S and I-35 S	0.169811	3 hours and 36 minutes
3	I-74 and US-41 N	0.210526	3 hours and 48 minutes

Bibliography:

[1] "Traffic fatalities up sharply in 2015 | National Highway Traffic Safety Administration (NHTSA)", Nhtsa.gov, 2016. [Online].
Available: http://www.nhtsa.gov/About+NHTSA/Press+Releases/trafficfatalities-2015.

[2] "How Do Weather Events Impact Roads? - FHWA Road Weather Management", Ops.fhwa.dot.gov, 2016. [Online]. Available: http://www.ops.fhwa.dot.gov/weather/q1_roadimpact.htm.

[3] NHTSA. Fatality Analysis Reporting System (FARS). Retrieved October 4, 2016, from National Highway Traffic Safety Administration, ftp://ftp.nhtsa.dot.gov/fars/

[4]"Google Maps APIs | Google Developers", *Google Developers*, 2016. [Online].
Available: https://developers.google.com/maps/

[5] Miaou, S., & Lum, H, "Modeling vehicle accidents and highway geometric design relationships. Accident; analysis and prevention", 25(6), 689–709. Published: 1993. Available: http://www.ncbi.nlm.nih.gov/pubmed/8297437

[6] Islam, M. B., & Kanitpong, K. (2008). "Identification of factors in road accidents through in-depth accident analysis" *Iatss research*, *32*(2), 58–67.

[7] J. Lee, J. Chung and B. Son, "Analysis of traffic accident size for Korean highway using structural equation models", *Accident Analysis & Prevention*, vol. 40, no. 6, pp. 1955-1963, 2008.

[8] K. S. Jadaan, M. Al-Fayyad, and H. F. Gammoh, "Prediction of road traffic accidents in Jordan using artificial neural network (ANN)," *Journal of Traffic and Logistics Engineering*, vol. 2, no. 2, pp. 92–94, 2014.

[9] Chong, M., Abraham, A., and Paprzcycki, M., "Traffic Accident Data Mining Using Machine Learning Paradigms," Oklahoma State University Computer, Science Department, 2004.

[10] G. Ali and A. Tayfour, "Characteristics and prediction of traffic accident casualties in Sudan using statistical modeling and artificial neural networks," *International Journal of Transportation Science and Technology*, vol. 1, no. 4, pp. 305–318, Dec. 2012.

[11] Zheng, L., Meng, X. (2011). “An approach to predict road accident frequencies: application of fuzzy neural network”. Paper presented at the 3rd International Conference on Road Safety and Simulation, September 14-16, 2011, Indianapolis, USA.

[12] Bae YK, Kim JH, Chung J-H, “Traffic Accident Analysis for Highway on Weather Condition and Time” *In: Proc. of the International Conference on Transport, Environment and Civil Engineering (ICTECE'2012*); August 25-26, 2012; Kuala Lumpur (Malaysia). 2012. p. 109-112.

[13] M. Y. Çodur and A. Tortum, “An artificial neural network model for highway accident prediction: A case study of Erzurum, turkey,” *PROMET – Traffic & Transportation*, vol. 27, no. 3, Jun. 2015.

[14] "Fatality Analysis Reporting System (FARS)", National Highway Traffic Safety Administration, 2016. [Online]. Available: http://ftp://ftp.nhtsa.dot.gov/fars/.

[15] R. Hecht-Nielsen, "Theory of the backpropagation neural network," *Neural Networks, 1989. IJCNN., International Joint Conference on*, Washington, DC, USA, 1989, pp. 593-605 vol.1.

[16] "Current Weather Data", Open Weather Map, 2016. [Online]. Available: https://openweathermap.org/current.

[17] Bergstra, James and Desjardins, Guillaume and Lamblin, Pascal and Bengio, Yoshua. "Quadratic polynomials learn better image features". Technical Report 1337, Département d’Informatique et de Recherche Opérationnelle, Université de Montréal 2009. http://www.iro.umontreal.ca/~lisa/publications2/index.php/attachments/single/205

[18] “TinyXML-2”, Lee Thompson. Available: http://www.grinninglizard.com/tinyxml2/

[19] “curl”, Daniel Stenberg. Available: https://curl.haxx.se/

[20] “<time.h>”, The Open Group. Available: http://pubs.opengroup.org/onlinepubs/7908799/xsh/time.h.html

[21] Arifovic, Jasmina. "Genetic Algorithm Learning And The Cobweb Model". *Journal of Economic Dynamics and Control* 18.1 (1994): 3-28. Web.

[22] Hochreiter, Sepp and Jürgen Schmidhuber. "Long Short-Term Memory". *Neural Computation* 9.8 (1997): 1735-1780. Web.