A Platform for Estimating the Relevance of Information in VANET Applications

BY

PIOTR MICHAL SZCZUREK
B.S., University of Illinois at Chicago, 2005

THESIS

Submitted as partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science in the Graduate College of the University of Illinois at Chicago, 2012

Chicago, Illinois

Defense Committee:

Ouri Wolfson, Chair and Advisor
Jie Lin, Civil and Materials Engineering, Co-Advisor
Robert Sloan
Prasad Sistla
Sol Shatz
This dissertation is dedicated to my wife, Katarzyna Anna Szczurek, without whom I would have never attended college. Only with her support and motivation was I able to graduate, begin a Ph.D. program, and complete this thesis.
ACKNOWLEDGMENTS

I would like to thank my advisors, Professor Ouri Wolfson and Professor Jie Lin for their support and guidance. Also, I want to thank all of my committee members for their time and valuable feedback: Professor Robert Sloan, Professor Sol Shatz, and Professor Prasad Sistla. In addition, I would like to thank Professor Clement Yu, who was a member of my preliminary examination committee. I would also like to thank Dr. Bo Xu for his help and collaboration in research work. Lastly, I would like to thank all of my friends and family for their encouragement.

This research was supported in part by the National Science Foundation IGERT program under Grant DGE-0549489.

PMS
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>INTRODUCTION ...................................</td>
</tr>
<tr>
<td>1.1</td>
<td>Outline ..........................................................</td>
</tr>
<tr>
<td>1.2</td>
<td>Problem Motivation ........................................</td>
</tr>
<tr>
<td>1.3</td>
<td>Contributions ................................................</td>
</tr>
<tr>
<td>1.4</td>
<td>Approach ..........................................................</td>
</tr>
<tr>
<td>1.5</td>
<td>Applications ..................................................</td>
</tr>
<tr>
<td>2</td>
<td>LITERATURE REVIEW ...........................................</td>
</tr>
<tr>
<td>2.1</td>
<td>General Techniques ..........................................</td>
</tr>
<tr>
<td>2.2</td>
<td>Approaches to Non-Safety Applications ...............</td>
</tr>
<tr>
<td>2.3</td>
<td>Approaches to Safety Applications ....................</td>
</tr>
<tr>
<td>3</td>
<td>NON-SAFETY APPLICATIONS ..................................</td>
</tr>
<tr>
<td>3.1</td>
<td>Model ...............................................................</td>
</tr>
<tr>
<td>3.2</td>
<td>Observe and Learn Approach ................................</td>
</tr>
<tr>
<td>3.2.1</td>
<td>Duplication Model ..............................................</td>
</tr>
<tr>
<td>3.2.2</td>
<td>Conditional Relevance Model ............................</td>
</tr>
<tr>
<td>3.2.3</td>
<td>Naïve Bayes Learning System ...............................</td>
</tr>
<tr>
<td>3.2.4</td>
<td>Model Sharing .....................................................</td>
</tr>
<tr>
<td>3.3</td>
<td>Application to Parking Information Dissemination ...</td>
</tr>
<tr>
<td>3.3.1</td>
<td>The Application Environment .............................</td>
</tr>
<tr>
<td>3.3.2</td>
<td>The Conditional Relevance Model .......................</td>
</tr>
<tr>
<td>3.3.3</td>
<td>Evaluation ..........................................................</td>
</tr>
<tr>
<td>3.3.4</td>
<td>Results .................................................................</td>
</tr>
<tr>
<td>3.4</td>
<td>Application to Travel Time Information Dissemination</td>
</tr>
<tr>
<td>3.4.1</td>
<td>The Application Environment .............................</td>
</tr>
<tr>
<td>3.4.2</td>
<td>The Conditional Relevance Model .......................</td>
</tr>
<tr>
<td>3.4.3</td>
<td>Evaluation ..........................................................</td>
</tr>
<tr>
<td>3.4.4</td>
<td>Results .................................................................</td>
</tr>
<tr>
<td>3.4.5</td>
<td>Using Additional Attributes to Improve Performance</td>
</tr>
<tr>
<td>3.4.6</td>
<td>Comparison of Machine Learning Methods through Offline Learning</td>
</tr>
<tr>
<td>3.4.6.1</td>
<td>Travel Time Update Method ..................................</td>
</tr>
<tr>
<td>3.4.6.2</td>
<td>Training Example Generation Procedure ...............</td>
</tr>
<tr>
<td>3.4.6.3</td>
<td>Comparison of Machine Learning Algorithms ..........</td>
</tr>
<tr>
<td>4</td>
<td>EMERGENCY ELECTRONIC BRAKE LIGHTS ................</td>
</tr>
<tr>
<td>4.1</td>
<td>Environment .......................................................</td>
</tr>
<tr>
<td>4.2</td>
<td>EEBL with the Relevance Estimation Application ......</td>
</tr>
<tr>
<td>4.3</td>
<td>Relevance Estimation Methods ............................</td>
</tr>
<tr>
<td>4.3.1</td>
<td>The Analytic Method ............................................</td>
</tr>
<tr>
<td>4.3.2</td>
<td>Observe and Learn Approach ................................</td>
</tr>
<tr>
<td>4.3.2.1</td>
<td>Attributes ..........................................................</td>
</tr>
<tr>
<td>4.3.2.2</td>
<td>Machine Learning Techniques .............................</td>
</tr>
<tr>
<td>4.4</td>
<td>Evaluation Methodology ......................................</td>
</tr>
<tr>
<td>4.4.1</td>
<td>Simulation Software ............................................</td>
</tr>
</tbody>
</table>
### TABLE OF CONTENTS (Continued)

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.4.2</td>
<td>Modifications to MITSIM ....................................... 85</td>
</tr>
<tr>
<td>4.4.3</td>
<td>Simulation Environment ........................................... 87</td>
</tr>
<tr>
<td>4.4.4</td>
<td>Evaluation Procedure .............................................. 89</td>
</tr>
<tr>
<td>4.4.4.1</td>
<td>Compared Methods .................................................. 90</td>
</tr>
<tr>
<td>4.4.4.2</td>
<td>Setting of the Reaction-delay and TWarning Parameters .......... 90</td>
</tr>
<tr>
<td>4.4.4.3</td>
<td>Measuring Driver Desensitization ................................ 93</td>
</tr>
<tr>
<td>4.4.4.4</td>
<td>Measuring the Safety Benefit of EEBL ........................... 93</td>
</tr>
<tr>
<td>4.5</td>
<td>Results ................................................................. 94</td>
</tr>
<tr>
<td>4.5.1</td>
<td>TWarning Values ..................................................... 94</td>
</tr>
<tr>
<td>4.5.2</td>
<td>Reaction-Delay Value .............................................. 96</td>
</tr>
<tr>
<td>4.5.3</td>
<td>Effects of the Departure Rate ..................................... 97</td>
</tr>
<tr>
<td>4.5.3.1</td>
<td>Highway Case ......................................................... 97</td>
</tr>
<tr>
<td>4.5.3.2</td>
<td>Local Street Case ................................................ 106</td>
</tr>
<tr>
<td>4.5.4</td>
<td>Effects of the Participation Rate ................................. 107</td>
</tr>
<tr>
<td>4.5.5</td>
<td>Effects of Adverse Weather ....................................... 109</td>
</tr>
<tr>
<td>4.5.6</td>
<td>Effects of Wireless Communication Parameters ............... 111</td>
</tr>
<tr>
<td>4.5.6.1</td>
<td>Effects of the Transmission Range .............................. 112</td>
</tr>
<tr>
<td>4.5.6.2</td>
<td>Effects of the Delivery Probability ............................ 114</td>
</tr>
<tr>
<td>4.5.6.3</td>
<td>Effects of the Propagation Delay ............................... 117</td>
</tr>
<tr>
<td>4.5.6.4</td>
<td>Effects of Not Rebroadcasting .................................... 118</td>
</tr>
<tr>
<td>4.5.7</td>
<td>Effects of the Probability of Reacting to Warnings ............ 121</td>
</tr>
<tr>
<td>4.6</td>
<td>Summary .................................................................. 122</td>
</tr>
<tr>
<td>5</td>
<td>PLATFORM IMPLEMENTATION FOR SAFETY APPLICATIONS .......... 123</td>
</tr>
<tr>
<td>5.1</td>
<td>VANET Safety Warning Application Model ........................ 123</td>
</tr>
<tr>
<td>5.1.1</td>
<td>Vehicle Capabilities ................................................ 124</td>
</tr>
<tr>
<td>5.1.2</td>
<td>Emergency Action .................................................... 125</td>
</tr>
<tr>
<td>5.1.3</td>
<td>The VANET Safety Warning Application .......................... 125</td>
</tr>
<tr>
<td>5.2</td>
<td>Observe-Driver-And-Learn Method ................................ 127</td>
</tr>
<tr>
<td>5.2.1</td>
<td>Attributes Selection ................................................ 129</td>
</tr>
<tr>
<td>5.2.2</td>
<td>Machine Learning Methods ......................................... 132</td>
</tr>
<tr>
<td>5.2.3</td>
<td>Adaptive Learning .................................................... 133</td>
</tr>
<tr>
<td>5.3</td>
<td>The Platform ........................................................... 134</td>
</tr>
<tr>
<td>5.4</td>
<td>Evaluation Methodology ............................................. 137</td>
</tr>
<tr>
<td>5.4.1</td>
<td>Evaluation Environment for EEBL ............................... 138</td>
</tr>
<tr>
<td>5.4.2</td>
<td>Evaluation Environment for CLW .................................. 138</td>
</tr>
<tr>
<td>5.4.3</td>
<td>Evaluation Environment for HMW ................................. 139</td>
</tr>
<tr>
<td>5.5</td>
<td>Results .................................................................. 140</td>
</tr>
<tr>
<td>5.5.1</td>
<td>Automatic Attribute Selection ..................................... 140</td>
</tr>
<tr>
<td>5.5.2</td>
<td>Comparison for Different Applications ........................... 141</td>
</tr>
<tr>
<td>5.5.3</td>
<td>Effects of the Driver Reaction to a Report ....................... 144</td>
</tr>
<tr>
<td>5.5.4</td>
<td>Machine Learning Method Performance .......................... 145</td>
</tr>
<tr>
<td>5.5.5</td>
<td>Weather Effects ...................................................... 146</td>
</tr>
<tr>
<td>5.6</td>
<td>Summary .................................................................. 149</td>
</tr>
<tr>
<td>6</td>
<td>CONCLUSIONS ........................................................... 150</td>
</tr>
<tr>
<td>CHAPTER</td>
<td>PAGE</td>
</tr>
<tr>
<td>------------------</td>
<td>------</td>
</tr>
<tr>
<td>CITED LITERATURE</td>
<td>152</td>
</tr>
<tr>
<td>VITA</td>
<td>160</td>
</tr>
</tbody>
</table>
# LIST OF TABLES

<table>
<thead>
<tr>
<th>TABLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Default Simulation Parameters ...................................................... 33</td>
</tr>
<tr>
<td>II</td>
<td>Default Simulation Parameters ....................................................... 51</td>
</tr>
<tr>
<td>III</td>
<td>Approximate 95% Confidence Intervals for Logistic Regression ............. 83</td>
</tr>
<tr>
<td>IV</td>
<td>Parameter values used in simulations ............................................. 89</td>
</tr>
<tr>
<td>V</td>
<td>Value of TWarning Parameter for Different Road Configuration Cases and Machine Learning Methods ............................................ 96</td>
</tr>
<tr>
<td>VI</td>
<td>List of attributes by type. ............................................................ 132</td>
</tr>
<tr>
<td>VII</td>
<td>Parameter Settings for the EEBL Scenario ......................................... 138</td>
</tr>
<tr>
<td>VIII</td>
<td>Parameter Settings for the HMW Scenario ......................................... 139</td>
</tr>
<tr>
<td>IX</td>
<td>List of Applications with the attributes Selected by the Automatic attribute Selection Algorithm ................................................. 140</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
</tr>
<tr>
<td>6</td>
<td>36</td>
</tr>
<tr>
<td>7</td>
<td>38</td>
</tr>
<tr>
<td>8</td>
<td>39</td>
</tr>
<tr>
<td>9</td>
<td>40</td>
</tr>
<tr>
<td>10</td>
<td>41</td>
</tr>
<tr>
<td>11</td>
<td>42</td>
</tr>
<tr>
<td>12</td>
<td>49</td>
</tr>
<tr>
<td>13</td>
<td>54</td>
</tr>
<tr>
<td>14</td>
<td>55</td>
</tr>
</tbody>
</table>

1. The process of developing VANET applications using the proposed platform.
2. Typical scenario for the EEBL application. Vehicle D is shown to be emergency decelerating and disseminating a report to the surrounding vehicles.
3. Typical scenario for the HMW application. Vehicles B through F, which are either approaching or passing near an on-ramp, are disseminating reports that are received by vehicle H that is trying to merge onto the road.
4. Typical scenario for the CLW application. Vehicle D has detected that it has lost control and disseminates a report to the surrounding vehicles.
5. The grid network, parking spaces, search squares, brokers and consumers.
6. Average % of discovery time improvement.
7. Performance across range of broadcast sizes.
8. Performance across range of number of consumers.
9. Performance across range of parking unavailability times.
11. Performance across range of transmission range values.
13. Average travel times for range of broadcast sizes.
14. Average travel times for range of number of slow-downs.
# LIST OF FIGURES (Continued)

<table>
<thead>
<tr>
<th>FIGURE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>Average travel times for range of transmission range values ........................... 55</td>
</tr>
<tr>
<td>16</td>
<td>Comparison between attributes - Broadcast size vs. average travel time. ........... 57</td>
</tr>
<tr>
<td>17</td>
<td>Region 1 road network. .................................................................................. 61</td>
</tr>
<tr>
<td>18</td>
<td>Region 2 road network. .................................................................................. 62</td>
</tr>
<tr>
<td>19</td>
<td>Accuracy of various machine learning algorithms.......................................... 63</td>
</tr>
<tr>
<td>20</td>
<td>Scenario for calculating the minimum safety gap. Shows vehicle ( C ) sending a report to vehicle ( A ), which has vehicle ( B ) in front of it............... 74</td>
</tr>
<tr>
<td>21</td>
<td>False and missed warning rates for different values of ( T_{Warning} ) in the two lane case using the logistic regression model (( Reaction-delay ) set to 9 seconds)........................................................................................................ 95</td>
</tr>
<tr>
<td>22</td>
<td>False and missed warning rates for different values of ( T_{Warning} ) in the two lane case using the naïve Bayes model (( Reaction-delay ) set to 9 seconds)........................................................................................................ 95</td>
</tr>
<tr>
<td>23</td>
<td>Average number of collisions per run for different values of ( Reaction-delay ) for naïve Bayes and logistic regression methods........................................ 97</td>
</tr>
<tr>
<td>24</td>
<td>False warning rates for all methods and different vehicle departure rates (one lane case). ................................................................. 99</td>
</tr>
<tr>
<td>25</td>
<td>False warning rates for all methods and different vehicle departure rates (two lane case). ................................................................. 100</td>
</tr>
<tr>
<td>26</td>
<td>False warning rates for all methods and different vehicle departure rates (three lane case). ................................................................. 100</td>
</tr>
<tr>
<td>27</td>
<td>Average number of collisions per run for all methods and different vehicle departure rates (one lane case)..................................................... 103</td>
</tr>
<tr>
<td>FIGURE</td>
<td>PAGE</td>
</tr>
<tr>
<td>-------</td>
<td>------</td>
</tr>
<tr>
<td>28</td>
<td>103</td>
</tr>
<tr>
<td>29</td>
<td>104</td>
</tr>
<tr>
<td>30</td>
<td>105</td>
</tr>
<tr>
<td>31</td>
<td>106</td>
</tr>
<tr>
<td>32</td>
<td>108</td>
</tr>
<tr>
<td>33</td>
<td>108</td>
</tr>
<tr>
<td>34</td>
<td>110</td>
</tr>
<tr>
<td>35</td>
<td>110</td>
</tr>
<tr>
<td>36</td>
<td>113</td>
</tr>
<tr>
<td>37</td>
<td>114</td>
</tr>
<tr>
<td>38</td>
<td>115</td>
</tr>
<tr>
<td>39</td>
<td>116</td>
</tr>
</tbody>
</table>
### LIST OF FIGURES (Continued)

<table>
<thead>
<tr>
<th>FIGURE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>Average number of collisions per run for the ML-NB method with different propagation delays for the one lane case.</td>
</tr>
<tr>
<td>41</td>
<td>False warning rates for two methods and varying transmission ranges for the three lane case, with and without report rebroadcasting.</td>
</tr>
<tr>
<td>42</td>
<td>Average number of collisions per run for two methods and varying transmission ranges for the three lane case, with and without report rebroadcasting.</td>
</tr>
<tr>
<td>43</td>
<td>Average number of collisions per run for all methods and different probabilities of reacting to a warning (three lane case).</td>
</tr>
<tr>
<td>44</td>
<td>Report Generation Process.</td>
</tr>
<tr>
<td>45</td>
<td>Report Reception Process.</td>
</tr>
<tr>
<td>46</td>
<td>The process for developing relevance estimation modules for VANET Safety Warning Applications using the proposed platform.</td>
</tr>
<tr>
<td>47</td>
<td>Comparison of the average number of collisions for each application (logarithmic scale).</td>
</tr>
<tr>
<td>48</td>
<td>Comparison of the false warning rates for each application.</td>
</tr>
<tr>
<td>49</td>
<td>The impact of the report reaction setting on the average number of collisions per run for the CLW application when using the ODaLe method.</td>
</tr>
<tr>
<td>50</td>
<td>The impact of the machine learning technique on the average number of collisions per run and the false warning rates for the EEBL application when using the ODaLe method.</td>
</tr>
<tr>
<td>51</td>
<td>Effects of visibility on the average number of collisions in the HMW application for each relevance estimation method.</td>
</tr>
<tr>
<td>FIGURE</td>
<td>PAGE</td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>52</td>
<td>148</td>
</tr>
</tbody>
</table>

Effects of the road-tire friction coefficient on the average number of collisions in the HMW application for each relevance estimation method.
**LIST OF ABBREVIATIONS**

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VANET</td>
<td>Vehicular Ad-Hoc Network</td>
</tr>
<tr>
<td>DSRC</td>
<td>Dedicated Short-Range Communication</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Positioning System</td>
</tr>
<tr>
<td>EEBL</td>
<td>Emergency Electronic Brake Light</td>
</tr>
<tr>
<td>HMW</td>
<td>Highway Merge Warning</td>
</tr>
<tr>
<td>CLW</td>
<td>Collision Loss Warning</td>
</tr>
<tr>
<td>BSM</td>
<td>Basic Safety Message</td>
</tr>
<tr>
<td>SVM</td>
<td>Support Vector Machine</td>
</tr>
<tr>
<td>SOTIS</td>
<td>Self-Organizing Traffic Information System</td>
</tr>
<tr>
<td>ZoR</td>
<td>Zone of Relevance</td>
</tr>
<tr>
<td>DIV</td>
<td>Duplication Indicator Vector</td>
</tr>
<tr>
<td>IGS</td>
<td>Information Guided Search</td>
</tr>
<tr>
<td>ML</td>
<td>Machine Learning</td>
</tr>
<tr>
<td>STRAW</td>
<td>STreet RAndom Waypoint</td>
</tr>
<tr>
<td>MITSIM</td>
<td>Microscopic Traffic SIMulator</td>
</tr>
</tbody>
</table>
SUMMARY

Progress in wireless communication and sensing technologies enabled research on Vehicular Ad-Hoc Network applications that aim to disseminate useful information. Examples include safety applications such as the Emergency Electronic Brake Lights, Highway Merge Warning, or Control Loss Warning, or non-safety applications such as parking or travel time dissemination systems. A major problem in these applications is knowing when the information is relevant for a given vehicle. The knowledge of information relevance helps applications to make decisions such as when to warn the driver because of some reported safety related information or which parking location should the driver pursue given a list of available locations? The estimates of relevance can also be used to rank the information. This maximizes the use of available resources such as the communication bandwidth and allows for the most important information to be disseminated.

Previously proposed methods for estimating relevance typically depend on heuristics or analytical solutions. These may be inaccurate or may not consider all the necessary factors. They are also application specific and therefore it is hard for developers to use these for novel applications. This dissertation proposes a simulation based platform for developing novel VANET applications. The platform generates relevance estimator modules that can be used in deployed applications. The method for generating the modules is based on a machine learning approach. The method works by using observations of vehicles in simulations to generate training examples that are used to learn a relevance function, which estimates the relevance of the given piece of information. This document presents research work on using this technique for safety and non-safety applications. It also presents an implementation of the platform for developing novel Vehicular Ad-Hoc Network safety warning applications.
CHAPTER 1

INTRODUCTION

A vehicular ad-hoc network (VANET) is a set of vehicles that communicate with each other via unregulated, short-range wireless technologies such as WiFi or Dedicated Short-Range Communication (DSRC) (DSRC, 2003). The introduction of VANETs, combined with an increased use of sensing devices (e.g. Global Positioning System (GPS), accelerometers, gyroscopes) in vehicles, made it possible to disseminate large amounts of information. This led to the development of many VANET applications which aimed to provide various benefits to the drivers such as lower travel times or increased safety.

The types of proposed applications can be classified as safety and non-safety applications. Examples of proposed safety applications include the Electronic Emergency Brake Lights (EEBL), the Highway Merge Warning (HMW), and the Control Loss Warning (CLW) (Carter, 2005). These safety applications work by warning the drivers of specific events detected by other vehicles. The warnings are activated as a result of the information being generated by the vehicles and are broadcast through wireless communication devices. After seeing a warning, drivers can take the appropriate actions to mitigate any potentially dangerous situations. Through this warning mechanism, such VANET applications provide a safety benefit by reducing the risk of vehicular collisions.

A major driver of these applications is the Crash Avoidance Metrics Partnership Consortium, composed of major automobile manufacturers along with the United States
Department of Transportation. The Consortium was responsible for creating the Vehicle Safety Communications-Applications project. Goals of this project included the development of communication standards and the identification and testing of potential safety applications utilizing DSRC. The project resulted in the standardization of the Society of Automotive Engineers J2735 Basic Safety Message (BSM) that contains the vehicle state information (e.g. position, speed, and acceleration) important for safety applications (Carter and Chang, 2009; Ahmed-Zaid et al., 2011b). It also identified several important safety applications and successfully tested its feasibility (Carter, 2005; Ahmed-Zaid et al., 2011a).

The BSM standard also allows for inclusion of non-safety related information in the message. This enables the implementation of many non-safety related applications that have been previously proposed. Examples include disseminating parking availability (Caliskan et al. 2004), travel speeds (Wischhof et al. 2003; Zhong et al. 2008), or traffic video clips (Guo et al. 2005; Lee et al. 2009). Such systems enable drivers to lower their travel times and reduce parking search times, which results in potential savings of fuel cost, emissions, and time. These benefits are directly related to the quantity and quality of information that can be disseminated.

1.1 Outline

This dissertation is composed of six chapters. This chapter describes the problem background and motivation, and the contributions made by this dissertation. It also provides an overview of the problem approach and the descriptions of the five applications for which the approach was implemented and evaluated. Chapter 2 presents a review of the literature related to relevance estimation, which includes general methods as well as methods specific
to the example applications. Chapter 3 describes the use of the proposed approach for non-safety related VANET applications: travel time dissemination and parking information dissemination. Chapter 4 describes the use of the approach for a particularly important safety VANET application, EEBL. In Chapter 5, I then present the implementation of the proposed platform for use in developing novel safety applications and present the results of evaluation for three example applications. Conclusions are presented in Chapter 6.

1.2 Problem Motivation

Although VANET applications have great potential benefits for the drivers, the information they disseminate may not always be relevant to every vehicle. For example, a report about hard braking by one vehicle may be useful for drivers that are close to such vehicle, but is unlikely to affect vehicles that are far away from where this event took place. There is thus a possibility of many false warnings being shown to the driver. Over time, this can lead to driver desensitization, which means the driver is likely to ignore the warnings (Dingus et al., 1997). To prevent this, the application developer must provide a method for estimating the relevance of the information to each vehicle that receives it. This allows for a warning to be displayed only when it is necessary, which increases the drivers’ trust in the system. The specification of the relevance estimator is hence an important part of the application development process. However, the determination of relevance can be uncertain and may depend on many factors, such as the positions, speeds, headings, and accelerations of vehicles, the vehicle types, the road characteristics, or the weather. With such a large quantity of factors, it can therefore be difficult to develop an appropriate relevance estimator for a novel application. Determining whether the developed estimator is effective will also be
difficult, because in real life, events such as emergency deceleration about which the application will warn the vehicles may be rare.

Dissemination of irrelevant information can also waste valuable resources such as bandwidth or storage. This is a problem when using VANETs due to the limited transmission radius and bandwidth which constraints the quantity of information that can be disseminated. Although individual pieces of information may be small in size, in combination, they could easily exceed the bandwidth capacity of a VANET. For example, in a travel time dissemination application, each individual travel time report might be on order of tens of bytes, yet with thousands of road segments, exchange of such information among vehicles might be prohibitive. The problem is exacerbated when considering sharing bandwidth among multiple applications. This makes it important to focus on the quality of information that is being disseminated. This can be done by ranking information that needs to be sent by its relevance for the receiving vehicles. In non-safety VANET applications, the information is typically spatio-temporal in nature. It is hence intuitive that the relevance depends on attributes such as the age (i.e., how long ago the information was generated) and the distance (i.e., how far is the vehicle from a point to which the information pertains). While by themselves, such spatio-temporal attributes maybe known to be indicative of the relevance, they need to be combined in order to assign a relevance value to a piece of information that can be used for ranking. Although heuristics, which are largely based on intuition, may be used for this purpose, knowing how to combine the attributes in an effective manner depends on the knowledge of the particular applications, and may not be trivial. Therefore a method for combining spatio-temporal attributes into a single relevance value is needed. Once the relevance value for each piece of information is known, vehicles can choose to transmit only
the most relevant set. This allows maximizing the usefulness of the VANET application, given its limited bandwidth.

1.3 Contributions

This dissertation contributes to the field of computational transportation science. It proposes a platform for finding the relevance of disseminated information in VANET applications. The platform consists of a simulation based framework along with a set of tools which enable entrepreneurs of novel VANET applications an easy, fast method for the development relevance estimator modules for their applications. The relevance estimator module estimates the relevance of a piece of information for the given vehicle. The developed module can be used for ranking of information which deals with VANET communication limitations by deciding which information should be transmitted to other vehicles. This results in more useful information being received by vehicles. It can also be used for making decisions such as whether to warn drivers of an event that can potentially affect their safety.

The use of the proposed platform has been implemented and evaluated for several safety and non-safety applications (see section 1.5 for application descriptions). Safety applications include EEBL, HMW, and CLW. The use of the platform for these applications allowed for significantly decreasing the number of false warnings shown and at the same time, maintaining the safety benefits of the applications. Non-safety applications include travel time dissemination and parking information dissemination. In both of these applications, the developed relevance estimator module allowed for decreasing travel times, either for reaching
the destination, or searching for parking. As a consequence of reduced travel times, drivers also benefit through lower fuel costs and lower emissions.

The work presented in this dissertation is based on previously published research. This includes research on individual applications such as travel time dissemination (Szczurek et al., 2009; Szczurek et al., 2010a; Szczurek et al., 2010b), parking information dissemination (Szczurek et al., 2010a; Szczurek et al., 2010c), and EEBL (Szczurek et al., 2011; Szczurek et al., 2012a). It also includes work on the platform framework and its use in developing novel safety applications (Szczurek et al., 2012b; Szczurek et al., 2012c).

1.4 Approach

The proposed platform uses microscopic traffic simulation and a relevance estimation method based on machine learning techniques. The process of using the platform for development is shown in Figure 1. The developer first defines his application through a report trigger condition, which specifies when a report should be disseminated. He also defines a scenario consisting of a road network specification, incident settings, and other parameters that would be typical for the given application. The scenario along with the report trigger condition are used as inputs for a microscopic traffic simulator, which is used to generate realistic vehicle movements that can then be observed and recorded. The relevance estimation method uses such observations to learn a relevance estimator module that can be used in deployed applications and adapted to individual vehicles and drivers.
In order to estimate the relevance of information, the Observe and Learn\textsuperscript{1} method relies on the use of machine learning techniques. Using this approach allows for combining several, known to be relevant attributes, into a single relevance value. The relevance value can then be subsequently used for ranking information or making decisions. It will be assumed that information to be disseminated is encapsulated in reports, which are created over time by the vehicles. Each report represents a piece of sensed spatio-temporal information such as the

\textsuperscript{1} In the context of safety warning applications that are presented in Chapter 5, we call this method Observe-Driver-and-Learn.
traffic condition of a road segment, the availability of a parking space, or a safety event (e.g. hard braking). Vehicles which receive such reports can use them to possibly alter their behavior. For example, they can change their travel route or pursue a particular parking space. It is postulated that finding which reports are most relevant depends on their spatio-temporal attributes. A relevant report is one that impacts the decisions of the receiving vehicle, such as changing its travel route or performing emergency braking.

To find the relevance, observations of the vehicles after the disseminated reports are received are used to create training examples. The training examples are then used for a machine learning process that learns the probability that a report is relevant as a function of its attributes, such as its age and distance. To apply this approach to a specific application, several elements are specified. First, the meaning of the relevance for the given application is defined. Second, the attributes that indicate the relevance are selected. Third, an appropriate machine learning method is chosen. The proposed work will describe how these elements are specified for several transportation applications such that the objectives of these applications are satisfied.

1.5 Applications

This dissertation shows five examples of VANET applications to which the platform is applicable. The first application is the dissemination of real-time traffic information. In this application, certain vehicles, which are equipped with the necessary devices (e.g. GPS, computer, communication device) record the time it takes them to traverse segments of a road. The recorded times are stored in a local database and are disseminated periodically to neighboring vehicles as travel time reports. The received reports allow the vehicles to update
the travel times of road segments for their local digital map. The objective of this application is to lower the average travel time of vehicles with the use of navigation devices based on the disseminated real-time traffic data. The main problem associated with travel time dissemination is the bandwidth, which limits the number of reports which may be disseminated at any given time. Therefore some type of ranking of reports is necessary. Ranking by relevance allows vehicles to receive reports most likely to affect the path to their destination, thereby possibly lowering their travel times. Estimating relevance for this application is particularly challenging because it depends on many attributes such as the road type, age of the report, or distance. Therefore most current approaches had to rely on heuristics for ranking the information, which may not take into account all the relevant attributes. The proposed machine learning based approach offers a way to easily combine all the attributes into a single relevance value.

Another application which benefits from relevance estimation is dissemination of parking availability information. In this application, vehicles generate a report after leaving a parking space. The reports are disseminated to all vehicles participating in the communication network, some of which are searching for a parking space in the area. It is assumed that vehicles are searching for parking locations with the highest chance of being available upon arrival to that location. The relevance in this case is thus defined in terms of the probability of availability. The objective of this application is to minimize the time it takes a vehicle to find an available parking space. Current approaches use either heuristics or analytically derived formulas for estimating relevance. The formulas may be able to calculate the precise availability probability, but they depend on many assumptions, such as the topology of the road network or knowledge about how many vehicles are looking for parking. The challenge
is that some of these assumptions may not hold: the topology of road networks may be different than one that is assumed and the number of vehicles seeking parking may not be known. Another issue is the competitive nature of the problem, which had not been accounted for in previous formulations. The use of our machine learning approach allows for taking into account many of these issues either explicitly (by using many attributes) or implicitly (through its probabilistic output).

Relevance estimation through machine learning can also be used in transportation safety related applications such as the EEBL, HMW, and CLW. EEBL disseminates a report whenever a vehicle performs emergency deceleration (see Figure 2). The idea is to extend drivers’ visibility in conditions where it may be limited, such as in bad weather situations (e.g. fog) or when another tall vehicle is directly in front (Carter, 2005). The report warns drivers and allows them to start braking earlier, thereby decreasing the chance of a vehicular collision. Intuitively, the relevance of an EEBL report depends on many attributes, including the positions of the receiving and sending vehicles, their velocities and lane positions, and the density of vehicles on the road.

Figure 2. Typical scenario for the EEBL application. Vehicle D is shown to be emergency decelerating and disseminating a report to the surrounding vehicles.
In HMW, vehicles on the highway generate reports whenever they enter a segment of the road that merges with an on-ramp segment, where vehicles attempt to join the traffic on the highway (see Figure 3). Receiving a report from the HMW application allows the merging vehicles to be warned whenever there is a possibility of a collision with the oncoming traffic. A driver that is shown a warning can slow down and be more alert when merging. For HMW, an intuitive formula for relevance may be derived based on the chance of a collision using the time-to-collision. The time-to-collision is the time a vehicle needs to reach the closest point intersecting the path of another vehicle.

Figure 3. Typical scenario for the HMW application. Vehicles B through F, which are either approaching or passing near an on-ramp, are disseminating reports that are received by vehicle H that is trying to merge onto the road.

A Control Loss Warning is used to warn drivers whenever a vehicle on the road loses control. This event may be detected by a stability control system of a vehicle. When a control loss occurs, a vehicle may become disabled and stop on the road and lane on which it was driving. In other cases, the vehicle may turn and end up in an adjacent lane (see Figure 4). By
warning the drivers of the other vehicles, the application can prevent collisions by allowing the drivers to slow down and be more alert of the situation.

Figure 4. Typical scenario for the CLW application. Vehicle D has detected that it has lost control and disseminates a report to the surrounding vehicles.
CHAPTER 2

LITERATURE REVIEW

2.1 General Techniques

In general, our method can be thought of as a way of handling imprecise queries. We can think of the VANET application in terms of a driver making a query to the system: “retrieve all reports relevant to my vehicle.” The query is said to be imprecise since the meaning of the term “relevant” is vague. Traditionally, handling of imprecise queries has been done through the use of fuzzy logic (Parsons, 1996). Fuzzy logic assigns a degree of truth to a fuzzy (i.e. imprecise) concept based on a membership function (Zadeh, 1965). Typically, proper membership functions might be determined through expert knowledge, although this approach can be problematic (Yen, 1999). Therefore automatic means of determining membership were developed. Examples include learning parameter values of the membership function (Lelescu et al., 1999; Javidi et al., 2008) or using fuzzy modifiers (Montenegro Gonzalez and Yamakami, 2006) with explicit relevance feedback. Another method used clustering for determining the membership (Zhou et al., 2003). A different approach described in (Xiong, 2008) used genetic algorithms for finding fuzzy rules.

A more recent technique for processing imprecise queries which does not rely on fuzzy queries can be found in (Kambhampati et al., 2007). That technique ranks query results by the expected relevance, which is computed using a relevance function and a density function. The relevance function reflects user preferences and can be found in various ways, such as co-
click similarities. The density function is based on the joint probability distribution of the attribute values and is found using a Naïve Bayesian classifier. Our approach uses similar probabilistic methods such as Naïve Bayes for learning the relevance, but is adapted to the context of VANET applications through the way that it determines the relevance of information through the implicit relevance feedback approach.

The use of relevance feedback has been extensively studied in information retrieval (Ruthven and Lalmas, 2003). The goal of information retrieval systems is to find the most relevant documents matching a given query, which is expressed as a set of terms. Relevance feedback is a mechanism in which users identify relevant documents, so that the query itself can be modified in order to increase the precision of the results (Rocchio, 1971). Since this technique requires explicit feedback, it can be burdensome to the users. As a result, pseudo-relevance feedback has been suggested (Croft and Harper, 1979). This technique assumes that all the top retrieved documents are relevant and uses these documents to modify the query. This method however suffers from a problem of query drift, in which the terms added to the query from the assumed to be relevant documents result in worse retrieval performance. Another approach for automatic feedback is the use of implicit feedback techniques. These techniques analyze user behavior, such as clickthrough data (Boyan et al, 1996; Kemp and Ramamohanarao, 2003; Tan et al., 2004; Joachims et al., 2005; Radlinski and Joachims, 2005) or other indicators (Hopfgartner and Jose, 2007), in order to judge the relevance. Our approach can be considered a form of implicit relevance feedback.

Techniques for ranking of information are related to those for cache management in mobile wireless networks. Work by Datta et al. (Datta et al., 2004), Perich et al. (Perich et al., 2004), and Zhang et al. (Zhang et al., 2007) used abstract utility functions which could be
defined for given applications. These can be thought of as a more general form of the relevance function, which our machine learning based method instantiates. Other work for ranking uses a weighted combination of popularity, reliability, and size (Sailhan and Issarny, 2002), but authors of this work do not discuss how the weights should be determined. In a paper by Zhang et al., reports are ranked such that the number of replicas of each report is proportional to the square root of its access frequency (Zhang et al., 2010). According to Cohen and Shenker, such a distribution of replicas has the optimal replication performance in minimizing the query cost (Cohen and Shenker, 2002). However, using access frequency is not always a suitable solution because access frequency of a newly produced report is always small but it is the newly produced report that is usually of most interest in VANET applications.

2.2 Approaches to Non-Safety Applications

A number of approaches to parking information systems currently exist. Leontiadis and Mascolo (Leontiadis and Mascolo, 2007) and Prinz et al. (Prinz et al., 2009) both propose a method based on a publish/subscribe paradigm which could be used for parking. Although such systems might filter out information which is irrelevant, they do not rank the information and they typically rely on the subscribers knowing which information is relevant. A number of heuristic methods exist for estimating relevancy of parking information. For example, Delot et al. used an encounter probability as a relevance function, which is estimated based on the weighted average of spatio-temporal characteristics of the parking information (Delot et al., 2009). The authors did not state how the weights should be found. In other work, the relevance is based on an ad-hoc function which is the sum of report age
and distance, where the distance is the time needed for a vehicle to arrive at the given location (Caliskan et al., 2006). The availability probability has been used for relevance by Lu et al. (Lu et al., 2009) and Wolfson et al. (Wolfson et al., 2005). In both, the arrivals of the vehicles are modeled by a Poisson process, parameters of which are assumed to be known by the parking information system. However, the assumptions made by these methods (e.g. the road topology configuration) may be violated and thus produce inaccurate relevance estimates.

Information ranking in the context of real-time traffic information dissemination has typically relied on the use of heuristics. The TrafficView system (Nadeem et al., 2004), which focused only on speed information about the traffic on the road ahead, divided the road into regions, where each section was assigned a specific aggregation ratio. The authors stated that the ratios could be assigned in decreasing order, making information about closer segments of the road more accurate. The authors did not however discuss how to optimally assign these ratios. Aside from the region information, the age of the information was also used to discard old information by comparison to a given threshold. A similar approach was used in the Self-Organizing Traffic Information System (SOTIS) (Wischhof et al., 2003), which attempts to disseminate speed reports about all segments on the road network. In this system, only the most current information was disseminated. Additionally, as in the TrafficView system, roads were divided into segments of various lengths. Authors of the SOTIS propose the segment lengths could be based on the road type, thereby prioritizing one type of a road segment over another. For example, authors proposed shorter lengths for highway segments on which the vehicle is currently driving, while longer ones for the country roads.
In work by Zhong et al. vehicles disseminate travel times they record after traversing road segments (Zhong et al., 2008). The reports are ranked by a combination of supply and demand. Supply is the estimate of how many vehicles already received the given report. Demand was estimated using a heuristic based on the sum of the age and the distance of a report, where distance was defined as the free-flow travel time along the shortest path from vehicle’s current position to the mid-point of the reported road segment. In contrast to our proposed approach, in all previously proposed types of systems, the relevance is based only on a few attributes (e.g. age of information, road type) and combined solely based on intuition.

The use machine learning techniques for determining information relevance has been studied by Manasseh et al., where the authors experimented with a smartphone application which alerted drivers of slowdowns in the traffic on the road ahead (Manasseh et al., 2010b). The application worked by collecting the vehicle’s positions using GPS and using the location of known alert events to warn the driver (see (Manasseh et al., 2010a)) for a detailed description of the application). Once an alert was issued to the driver, the application displayed a feedback screen in which the users could select whether they thought the warning was or was not useful. Additionally, for each displayed alert, a set of attributes, such as the vehicle’s speed and position, was recorded. These attributes, along with the corresponding feedback were then used for machine learning using support vector machines (SVM). Results showed that using the trained SVM classifier, 80% of unwanted warnings could have been suppressed for 50% of the users. While the work presented in this paper also uses machine learning techniques, the key difference is the feedback mechanism. In contrast to the procedure described by Manasseh et al., our feedback method is automatic and therefore does
not involve users inputting their preferences manually. The advantage of this, aside from eliminating the burden on the users, is that the feedback is based on the user actions and not a stated preference, which may be imprecise. In addition, manual feedback would be difficult to be acquired in safety applications, because safety events such as emergency deceleration are rare. Since the major difference between the method presented by Manasseh et al. and our proposed method is the feedback mechanism, the two cannot be directly compared.

2.3 Approaches to Safety Applications

The issue of relevance in the EEBL application was mentioned in a report by the National Highway Traffic Safety Administration, when EEBL was proposed as a potential DSRC application (Carter, 2005). Authors wrote that relevance could be based on the lane and direction information determined by the GPS position. This strategy for determining relevance was used by Zang et al., where authors proposed to disseminate EEBL messages to all vehicles in the so called Zone of Relevance (ZoR) (Zang et al., 2008). A vehicle was defined to be in the ZoR if it is traveling behind the vehicle that originated the EEBL message and in the same direction. The issue with this definition of relevance is that some vehicles may be traveling far behind the EEBL originator and may not need to be warned. Therefore many false warnings could be created as a result.

A more sophisticated method for estimating relevance was proposed by van Arem et al. (van Arem et al., 2003). In that work, authors proposed that the warnings be evaluated based on a time-space relevance factor, which gives higher values to warnings which are recent and close, and lower values to those which are old and far away from the warning originator. The factor is also based on the difference in speeds between the receiving and the originating
vehicles. However, the exact form of the time-space relevance factor was not given, so a direct comparison of this method cannot be made.

Another possible relevance estimation method was described by Miller and Qingfeng (Miller and Qingfeng, 2002). That paper considered an application where warnings are given for collisions at intersections. Authors proposed to show the warnings when the difference between the estimated time-to-collision and the time-to-avoidance values are within a certain threshold. This computation depends on several parameters that need to be tuned to consider human factors. A similar method was described by Tu and Huang (Tu and Huang, 2010). Although such methods could be employed in novel safety applications, the formulations use a fixed number of factors and cannot be easily extended to include additional information (such as weather).

Galler and Asher developed equations for a collision avoidance system based on calculations of the required deceleration necessary to avoid a collision with the vehicle in front (Galler and Asher, 1995). To derive these equations, authors assumed that the information about every vehicle’s position, velocity, and acceleration is known at all times. This is unlike many proposed VANET safety systems, which only disseminate information about the vehicle that sensed some specific safety event. Therefore the equations cannot be used directly. In addition, the equations also need assumptions about driver reaction times and how the drivers respond to the actions of the other vehicles. The same issues also exist in the techniques used for threat assessment (Eidehall and Petersson, 2008; Althoff and Mergel, 2011; Brannstrom et al., 2010). In these papers, the paths of the vehicles on the road are predicted and this information can then be used to warn the drivers. A similar method was also described by Greene et al., where in addition, statistical inference is used in calculating
the expected utility of a warning (Greene et al., 2011). However, the authors did not address the issue of when the warnings should be shown. Additionally, just like the formulation proposed by Galler and Asher (Greene et al., 2011), all of the techniques described in threat assessment literature (Eidehall and Petersson, 2008; Althoff and Mergel, 2011; Brannstrom et al., 2010; Greene et al., 2011) are largely dependent on the knowledge of other vehicles’ positions and velocities.

The impact of false warnings on driving behavior was studied by Dingus et al. (Dingus et al., 1997). Authors experimented with drivers using a collision avoidance system, which worked by warning drivers when the headway to the front vehicle became too close. Tests were performed to see how well the drivers could maintain a safe headway. During these tests, false warnings at various rates were purposely displayed to the drivers to test their reaction. It was discovered that when the false warning rate increased over 60%, the drivers began to decrease their headways, implying they started to distrust the system.
CHAPTER 3

NON-SAFETY APPLICATIONS

This chapter presents the implementation of the platform methodology using the machine learning method for relevance estimation for two VANET applications: parking availability dissemination and travel time dissemination. In the parking application, we assume vehicles are searching for a parking location with the highest chance of being available upon arrival to that location. The relevance is thus defined in terms of the probability of parking availability. Using the parking application, we show that the machine learning algorithm successfully combined spatio-temporal attributes of information in order to calculate the relevance. We do so by evaluating using a simple simulation model for which an analytically derived function which calculates the availability probability exists (Wolfson et al. 2005). For travel time dissemination, no such analytical functions currently exist, so most methods typically rely on heuristics. Simulation results for travel time dissemination show that the machine learning technique achieved better performance than the use of individual spatio-temporal attributes or a heuristic based on combination of these attributes. In the simulated environment the vehicles were able to choose better routes and lower their travel times as a result of the use of the proposed technique for ranking reports.

In the next section, we present the model we have used for the two non-safety applications. In section 3.2, we then describe how our proposed approach was implemented for applications using the model. Sections 3.3 and 3.4 then present the specific
implementations for the parking information dissemination application and the travel time dissemination application, respectively.

### 3.1 Model

The system consists of a set of mobile nodes. A node is a physical entity capable of data computation, storage, and short range wireless communication. A node can also observe its environment through a sensing device. The sensing device may be a camera installed in the car, an odometer, or GPS. Examples of nodes include vehicles equipped with on-board computers and Wi-Fi.

At any point in time, a node may create a report, which contains the data derived from the sensing device. The data is formed as a fixed set of attributes and their values. An attribute identifies the type of the data value. An example of a report is a speed report, whose attributes are time and current speed. Other examples of reports include reports about traffic accidents or available parking spaces.

Every node $x$ carries a reports database of size $RRsize$. The reports database contains reports the node has received or created over time. All the reports have the same size, although the generalization to variable size is possible. The reports in the report database are sorted in order, according to a value given by the ranking function. The ranking function, $Rf$, maps every possible report into a rank i.e. number between 0 and 1. It is assumed that higher ranks are given to more important reports for an arbitrary recipient. When it is the case that the reports database is full, upon insertion and re-ranking, the lowest ranked reports will be discarded, until all reports can be stored within the given capacity.
Each node $n$ can transmit to and receive from other nodes that are within transmission range, denoted $Tr$. These nodes are called *neighbors of n*. Every $Bi$ seconds, each node broadcasts $Bsize$ reports to its neighbors. The time between broadcasts is called the *inter-broadcast interval* and the number of reports that are broadcast is called the *broadcast size*. The value of $Bsize$ depends on the report size and the available bandwidth and can be computed using a bandwidth optimization method (Wolfson et al., 2005). The reports with the highest ranking values are sent in each broadcast.

Certain nodes, called *feedback nodes*, can judge the *relevance* of a report they receive. The relevance represents the expected utility of a report to an arbitrary neighbor of the feedback node. In other words, how useful would the report be to the recipient? This value is numeric and can be either Boolean or real valued. In cases where nodes assign numeric values, those will be assumed real valued in the range of 0 to 1. When nodes can only judge whether the report was good or bad, the report’s value is Boolean (0 for bad, 1 for good). As an example, consider a report that represents the availability of a parking space. The node (vehicle) can judge the report as “good” if the parking space remains available when the node reaches it. One problem in this model is thus to find a ranking function that allows for the most relevant reports to be disseminated during every broadcast.

### 3.2 Observe and Learn Approach

In general, the proposed relevance ranking method works using received reports as an input to a machine learning process. It is assumed that certain nodes have the ability to make relevance judgments when they receive a report. Given this, a supervised learning algorithm can be used with the judged relevance as the given output. Over time, each feedback node
learns a model that can estimate the probability that a report is relevant to an arbitrary recipient, and the model can then be used as a ranking function. Feedback nodes disseminate their learned models along with the reports so their learned models are shared with non-feedback nodes.

The model used to estimate the probability that the report will be relevant consists of two parts: a duplication model and a conditional relevance model. The duplication model is used to find the \textit{novelty factor}, which is the probability that a given report is not a duplicate to (i.e., has not been received by) a neighboring node. The conditional relevance model estimates the probability that a given report is relevant to the recipient, assuming the report is new to the recipient (i.e. it is not a duplicate). The rank value of a report \( R \) to neighboring node \( n \) is the multiplication of the estimates from both models:

\[
\begin{align*}
\text{rank}(R) &= \text{Prob}(R \text{ is new to a neighboring node } n) \\
&\quad \times \text{Prob}(R \text{ is relevant to a neighboring node } n, \text{ provided it is new to } n). 
\end{align*}
\]  

Note that the separation into duplication and conditional relevance models is not theoretically necessary, because a duplicate report is automatically not relevant. Therefore a single model could have been used, but our experimental testing has revealed that using separate models allows for higher performance.

The next subsection, provides the details of the duplication model. The conditional relevance model is described in subsection 3.2.2. Subsection 3.2.3 explains the Naïve Bayes method used for generating the conditional relevance model and 3.2.4 describes how the conditional relevance models are shared among the nodes.
3.2.1 **Duplication Model**

The duplication model learns the probability that a sent report would be a duplicate. In order to learn this probability, an existing technique called MALENA is used (Xu et al., 2009). This technique works as follows. For each report \( r \) stored at a node \( n \), \( n \) maintains a *duplication indicator vector* (DIV) for \( r \). The DIV consists of two attributes of \( r \): \( fin \) and *broadcast age*. \( fin \) is the number of times \( r \) has been received by \( n \). Intuitively, the higher the \( fin \), the more likely that \( r \) is a duplicate, since this means that \( r \) has already been widely disseminated by other nodes. The broadcast age of report \( r \) for node \( n \) is the number of broadcasts that have been sent by \( n \) since \( r \) was last broadcast by \( n \). Intuitively, the higher the broadcast age, the less likely that \( r \) is a duplicate, since this means that \( n \) has not broadcast \( r \) since a long time ago. When \( r \) is transmitted, its DIV is attached to \( r \). A receiver node \( m \) of \( r \) checks whether or not \( r \) is a duplicate, and the respective DIV becomes a training example. Specifically, if \( r \) is a duplicate, then \( neg \), the number of negatives for the respective DIV is increased by one. Otherwise, \( pos \), number of positives is increased by one. Initially, both \( neg \) and \( pos \) start at zero for all DIVs. Given the DIV, the probability that report will not be a duplicate can be calculated simply by dividing the number of positives by the sum of positives and negatives.

To calculate the broadcast age of \( r \), \( n \) remembers the time of the last broadcast that includes \( r \). The broadcast age of \( r \) is then calculated by dividing the time passed since the last broadcast of \( r \) by the length of the inter-broadcast interval. For newly created reports that have never been broadcast, the broadcast age is infinite. For reports that have been received but not yet broadcast, the broadcast age is defined as zero.
Training examples for the duplication model are created after a node receives a set of reports. In order to detect the duplicate reception, a node $n$ remembers the id’s of all the reports it has ever received. A positive or negative example will be created for the duplication model depending on whether the node has the id of the report in its list. All duplicate reports are immediately discarded.

### 3.2.2 Conditional Relevance Model

This model assumes that the report that is received has never previously been received. Then the model estimates the probability that the report is relevant to the recipient, which is called the *conditional relevance*. To provide the necessary training data for learning the relevance model, each report is augmented with additional attributes related to the sender of the report. Although dependent on the actual application, the attributes in spatio-temporal environments would generally depend on time and space. By knowing these attributes, the receiving node can learn the mapping from the sender’s and report’s characteristics to the relevance of a report. The receiving node, which would later resend the report, can then estimate the relevance to a future receiver. For the purpose of learning the conditional relevance model, the Naive Bayes online learning method is used.

### 3.2.3 Naïve Bayes Learning System

Let $x_1, x_2, ..., x_k$ be the values of the $X_1, X_2, ..., X_k$ attributes of a report $r$. According to the Bayes’s theorem, the probability that $r$ is relevant to the receiver is

\[
P(\text{relevant} | x_1, x_2, ..., x_k) = \frac{P(\text{relevant} \cdot P(x_1, x_2, ..., x_k | \text{relevant}))}{P(x_1, x_2, ..., x_k)}
\]  

(3.2)

where
\[ P(x_1, x_2, \ldots, x_k) = P(\text{relevant}) \cdot P(x_1, x_2, \ldots, x_k \mid \text{relevant}) + \\ P(\text{not-relevant}) \cdot P(x_1, x_2, \ldots, x_k \mid \text{not-relevant}) \]  
(3.3)

Assume that \( X_1, X_2, \ldots, X_k \) are conditionally independent. We have

\[
P(x_1, x_2, \ldots, x_k \mid \text{relevant}) \\ = P(x_1 \mid \text{relevant}) \cdot P(x_2 \mid \text{relevant}) \cdots P(x_k \mid \text{relevant}) 
\]  
(3.4)

\[
P(x_1, x_2, \ldots, x_k \mid \text{not-relevant}) \\ = P(x_1 \mid \text{not-relevant}) \cdot P(x_2 \mid \text{not-relevant}) \cdots P(x_k \mid \text{not-relevant}) 
\]  
(3.5)

Thus, in order to calculate the probability that \( r \) is relevant to its receiver, two values need to computed: \( P(\text{relevant}) \) and \( P(x_i \mid \text{relevant}) \). \( P(\text{relevant}) \) is estimated to be the ratio between the number of positive examples and the total number of examples. \( P(x_i \mid \text{relevant}) \) is estimated as follows. For numeric attributes, it is assumed that values of \( x_i \) for positive examples follow a normal distribution. The mean \( \mu \) and variance \( \sigma \) of the normal distribution is derived from the positive examples. \( P(x_i \mid \text{relevant}) \) is then estimated using the probability density function of normal distribution:

\[
P(x_i \mid \text{relevant}) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(x_i - \mu)^2}{2\sigma^2}} 
\]  
(3.6)

For nominal attributes, \( P(x_i \mid \text{relevant}) \) is estimated as the ratio between the number of positive examples with \( X_i = x_i \) and the total number of positive examples. \( P(\text{not-relevant}) \) and \( P(x_i \mid \text{not-relevant}) \) are estimated similarly to their “relevant” counterparts.

### 3.2.4 Model Sharing

Since conditional relevance training can only be done by feedback nodes, a model sharing procedure is used to disseminate the learned models. The procedure works as follows. When a feedback node \( n \) broadcasts its set of reports, it includes with it the learned model
Naïve Bayes model, which is based on a set of \( k \) training examples. A receiving node then replaces its model with the received one if it is based on more than \( k \) training examples. Otherwise, it continues to use its own model. Since models based on more training examples are more likely to be accurate, the model sharing procedures allows non-feedback nodes to learn along with feedback nodes and for all nodes to improve their conditional relevance models over time.

### 3.3 Application to Parking Information Dissemination

This section will discuss how the proposed platform methodology can be applied for a parking availability dissemination application. The Observe and Learn approach was used in order to rank the reports with the learned conditional relevance and duplication models. The model was also used to determine which parking spaces a vehicle should pursue when it receives a report. We show that the method successfully learned the relevance of reports by comparing it in simulations with a function that was proved to be optimal in the given simulated model.

#### 3.3.1 The Application Environment

The environment consists of a set of vehicles. A subset of these vehicles is equipped with GPS and devices capable of computation and short-range wireless communication. The set of vehicles that are equipped we call *participating vehicles*, otherwise they are labeled *non-participating vehicles*. Some of the vehicles actively look for available parking spaces. Vehicles that actively look for parking are called *consumers*. Participating vehicles which are not looking for parking are called *brokers*. The feedback nodes in this model are the participating consumers, while non-feedback nodes are the brokers. We assume that
consumers traverse the road network around a search path. When a consumer encounters an available parking space while traversing the search path, it parks there and makes the parking space unavailable for some time.

When a participating vehicle leaves its parking space, it generated a parking availability report. This report stores the following fields: report identifier, location, and timestamp. The report identifier provides a unique number for each report and is used for duplicate detection. The location is the coordinates of the parking space given by the GPS. The timestamp is the time at which the report is produced. Dissemination of reports is done as described in section 3.1, with the exception that participating consumers do disseminate reports about parking spaces they are pursuing.

All reports are stored in the reports database. Only the most recent reports for a given location are kept. Incoming reports for a location provided by an existing report are treated as duplicates if their timestamps are the same or older than the existing report.

When a consumer vehicle receives a parking availability report, it may choose to deviate from its search path and try to obtain the parking space referenced in the report. Such parking space then becomes the target parking space for that vehicle. When multiple reports exist, the target parking space is chosen according to the ranking function. Each time the vehicle receives new (i.e., never previously received) reports, the target parking space decision is reevaluated based on current values of the ranking function.

Consumer vehicles park in the first available parking space, regardless of whether or not they are pursuing their target parking space. Therefore, the target parking space may never be reached. When it is reached, but it is unavailable at that time, the vehicle continues to search for parking along a search path.
3.3.2 The Conditional Relevance Model

In this subsection we instantiate the conditional relevance model for the parking availability dissemination application. This includes defining the meaning of relevance, selecting attributes, and choosing the report trigger conditions.

For parking availability dissemination applications, we define being relevant as follows: a parking availability report \( r \) received by vehicle \( v \) is *relevant* if the parking space referred to in the report is available when \( v \) reaches it. A positive example is hence created when the vehicle parks. Otherwise, if the vehicle reaches the space and it is unavailable, it is a negative example.

Since the relevance model will also be used to decide which parking spaces will be pursued, the learning of the negative examples can be skewed. Additionally, a vehicle pursuing a far away parking space will reach its intended target infrequently, because another parking space might be available along the way. This will again skew the learning of negative examples. To deal with this issue, when it is the case that a vehicle pursuing a target parking space happens to find parking along the way, it will generate a positive report with attribute values as if another vehicle has just left that space.

Since parking availability reports are spatio-temporal in nature, the age and distance are included in the conditional relevance model, where age is the time since report was created and distance is the time needed to reach the parking space. Formal justifications for using these attributes for parking applications are given by Wolfson et al., who prove that under certain conditions, a function of age and distance is equivalent to the probability that the parking space would be available if the vehicle would decide to pursue it (Wolfson et al., 2005).
Training examples for the duplication model are created by every participating vehicle when they receive a set of reports. Conditional relevance training examples are created by observing participating consumers once they reach their target parking space. When a consumer initially chooses its target parking space, the values of age and distance are saved. Once the target space is reached, a positive or negative example is created depending on whether the space is available or unavailable, respectively. The example is created using the previously observed age and distance values and the appropriate relevancy label (i.e. positive or negative).

### 3.3.3 Evaluation

In this section we describe how the method was evaluated for the parking availability dissemination application. The results will demonstrate that method was able to combine age and distance attributes in an optimal way, where optimal is given by the equation derived by Wolfson et al. (Wolfson et al., 2005).

A proprietary simulator was used to evaluate the method. In the simulation, vehicles were randomly placed on a grid road network that is 1.2 miles by 1.2 miles in area. Distance between subsequent intersections in the grid was 0.1 miles. Available parking spaces were mapped to points at intersections in the grid. One parking space was assigned to every other intersection, for a total of 36 (see Figure 5). There were $u$ broker vehicles per square mile of the grid network and $c$ consumer vehicles. Out of the $c$ consumers, $w$ was the fraction of participating vehicles.
Vehicles were placed on the grid at random locations. Broker vehicles moved about according to a random waypoint mobility model. They generated a random location to be their destination and traversed to that location along the shortest path. Once the destination was reached, another one was chosen. Consumer vehicles moved around a set of road segments defined by their search square. The search square was defined such that the side length was 0.4 miles, the vehicle was on one of the sides with equal probability, and the square was aligned with the grid network such that the vehicle was as close to the middle of the side as possible and had at least one parking space in its path. A participating consumer could leave its search square to pursue its target parking space. In that case, it traversed the
shortest path to the target parking space. The speed, in miles per hour, of every vehicle was chosen randomly from the interval \([v-5, v+5]\), where \(v\) is the average speed of vehicles.

When a consumer vehicle parked at an available parking space, it was removed and another consumer was placed at the same location with the same search square. If that consumer was participating, the conditional relevance model was inherited from the old consumer. The parking space where the old consumer parked became unavailable for a random length of time according to an exponential distribution with mean \(q\). The simulation parameters and their values are summarized in Table I.

### TABLE I. DEFAULT SIMULATION PARAMETERS

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total number of parking spaces</td>
<td>36</td>
</tr>
<tr>
<td>Simulated area</td>
<td>1.2x1.2 sq. miles</td>
</tr>
<tr>
<td>Broker density (u)</td>
<td>100 per square mile</td>
</tr>
<tr>
<td>Number of consumers (c)</td>
<td>20</td>
</tr>
<tr>
<td>Percentage of participating consumers (w)</td>
<td>50%</td>
</tr>
<tr>
<td>Transmission range (Tr)</td>
<td>250 meters</td>
</tr>
<tr>
<td>Mean vehicle speed (v)</td>
<td>20 mph</td>
</tr>
<tr>
<td>Mean of parking unavailability time (q)</td>
<td>20 minutes</td>
</tr>
<tr>
<td>Broadcast size (Bsize)</td>
<td>1 report</td>
</tr>
<tr>
<td>Inter-Broadcast interval (Bi)</td>
<td>5 seconds</td>
</tr>
<tr>
<td>Report database size (RRsize)</td>
<td>36 reports</td>
</tr>
<tr>
<td>Total simulation time</td>
<td>20 hours</td>
</tr>
</tbody>
</table>

Since initially the conditional relevance and duplication models are not based on any examples, their performance will be much lower than it potentially could be over time.
Therefore, we chose to bootstrap the models. This was done by running the simulation with default parameters for a total simulation time of 72 hours. After this time, the learned duplication and conditional relevance models were saved and used as the initial models for all tests.

Five ranking methods were compared. For each one, the duplication model was used to calculate the novelty factor, which was used as a multiplier for the given ranking function. This was done to focus on the effects of the conditional relevance model. The following ranking functions were used (labels in parentheses):

1. \(1/\text{age}. \) (age)
2. \(1/\text{distance}. \) (distance)
3. \( e^{-\alpha \cdot \text{age} - \beta \cdot \text{distance}} \), where \( \alpha = \varphi, \beta = \varphi / v, \) and \( \varphi = 2 \cdot c \cdot v / l \), and where \( c \) is the number of consumers, \( v \) is the mean speed of consumers and \( l \) is the length of all road segments in the network. This function was derived as part of a method called the Information Guided Search (IGS) (Wolfson et al., 2005).
4. Conditional relevance probability output using the proposed machine learning based method. (ML)

The first method (age) ranks solely based on age, while the second method (distance) ranks solely based on distance. The third method (IGS) is based on an equation derived for the Information Guided Search (Wolfson et al., 2005). It computes the probability that a given space is unoccupied given its age and distance under a certain limited model. The model used for deriving the IGS equation assumes arrivals at parking spaces follow a Poisson distribution. The issue with this equation is that it depends on the knowledge of several parameters, such as the number of consumers or their mean speed, which would be unknown.
in a real environment. It therefore can be said to represent an ideal benchmark case. The last method (ML) is our proposed machine learning based method with age and distance as attributes.

To compare the different ranking methods, we measured the average discovery time for participating consumer vehicles. The discovery time is the difference in time between when a consumer is created and when it finds an available parking space. We then compared this time with the average discovery time in a case in which every consumer uses a blind search. This means that no vehicle is participating and can only find parking spaces that are along its search path. The percentage decrease between the blind search time and the given ranking time is then shown in figures as the percentage improvement in discovery time.

3.3.4 Results

In this section we show the simulation results. 25 tests were performed using various parameter values. In each test, a single parameter was modified, while keeping the rest at their default values (see Table I in the previous section). Each ranking method was tested with and without the use of the novelty factor, which was determined by the duplication model. The percentage improvement in the discovery time for each ranking method, averaged over all tests, was calculated and is shown in Figure 6 below.
The figure shows that the machine learning method was able to nearly match the performance of the IGS equation, which was shown to exactly calculate the availability probability. In fact, in almost every test, the discovery time performance closely matched that of IGS. Both methods were significantly better than using individual attributes (age, distance) for rankings. This illustrates that the proposed method can successfully learn an optimal combination of relevant attributes.

Although using distance outperformed using age by a wide margin on average, individual test results showed that distance is not always the best attribute to use. In two out of the 25 tests, using ranking by age was actually superior. The use of the novelty factor showed that
the duplication model effectively eliminates duplicate reports from being disseminated. The performance of all ranking methods improves with the use of the novelty factor, although the margin is small. This is because of the small number of reports being disseminated. As will be shown later in the travel time dissemination application (see section 3.4), the effects can be much larger when the number of reports in the system is higher.

Figure 7 shows the performance of the different ranking methods when varying the broadcast size. The results show that the broadcast size generally has no effect on discovery time. This means there is little to gain from broadcasting more than one report at a time. Throughout the broadcast size range, using age and distance resulted in the lowest improvement on average, with 18.3% for distance and -2.7% for age. Machine learning averaged a 29.1% improvement, significantly higher than either age or distance, closely following the performance of IGS, which averaged 29.9%.
Figure 7 shows the performance of the different ranking methods when varying the number of consumers. As the figure shows, the performance of age ranking generally decreases as number of consumers increases, while distance ranking performance is the opposite. The reason for this is that with a high number of consumers, the probability of parking availability is decreasing throughout and therefore it may not be worth it to pursue far away parking spaces. This effect was the reason for the general decrease in performance for all methods, with the exception of distance ranking, as the number of consumers increased.
In Figure 9, the impact of varying the mean of parking unavailability time is shown. Similarly to previous results, ML came close to the performance of using IGS and outperformed age and distance by a significant margin. The results show that for distance based rankings (i.e., all but age), the unavailability time does not have much effect on percentage improvement in discovery time. Age ranking improves performance as the mean time increases, because there is less variability in the parking space availability. To see this, consider an extreme example, in which mean unavailability time is close to 0. In that case, age would no longer matter, since the parking is almost guaranteed to be available by the time the vehicle arrives. The age hence becomes an irrelevant attribute.
Broker density affects how quickly information is disseminated. As a result, it affects ranking performance in two ways. On one hand, as broker density increases, the speed of dissemination also increases and this improves performance because vehicles are able to rank based on a greater amount of information. On the other hand, the increased information causes greater competition among participating consumers, which results in lower performance. This second effect is especially evident for age ranking, as can be seen in Figure 10. At high densities, most consumers will receive a generated report in a short amount of time, which will cause all such consumers to pursue the same parking space. When distance is used in the ranking method, the effect is mitigated because competition will be
more localized. Ranking methods which use a combination of age and distance thus do not have much variance in their performance.

Figure 10. Performance across range of broker densities

Figure 11 shows how performance varies across a range of transmission range values. Similar to broker density, this parameter impacts the speed of information dissemination and therefore the two effects that affected performance for broker density are present in transmission range tests. Unlike broker density, transmission range has a much bigger impact on dissemination, because of its major impact on the communication delay (i.e. difference
between generated and received time). As a result, the graphs of IGS and ML show
performance follows an “S” shape: very low performance at close transmission range (50m),
high gains in performance in the 100m to 250m range, and marginal improvement at 300m.
With distance ranking, the shape of the performance curve is flatter, but still generally
increasing with increased range. Note the slightly erratic behavior of the curve can be
attributed to the use of the novelty factor, which causes some randomness in the
dissemination of the reports. Tests (not shown) without novelty factor, show a monotonically
increasing curve for distance ranking. For age, the effect of increased competition causes a
decrease in performance after 200m range.

Figure 11. Performance across range of transmission range values
3.4 Application to Travel Time Information Dissemination

This section will discuss how the Observe and Learn method for the estimation of relevance can be used to improve dissemination of vehicle travel times on a road network. We used the STreet RAndom Waypoint (STRAW) (Choffnes and Bustamante, 2005) vehicle simulator and show that the use of the machine learning approach for ranking allows vehicles to lower their average travel times when compared against ranking using heuristics.

3.4.1 The Application Environment

As in the parking application, the environment consists of a set of vehicles, a subset of which is equipped with GPS and devices capable of computation and short-range wireless communication. Every vehicle traverses a road network to a predetermined destination, which it reaches along the path with the shortest travel time, given the information it currently has in its local database. In the rest of this chapter, when we say the shortest path we refer to the shortest travel time path. In this model, every participating vehicle is a feedback node.

As each participating vehicle fully traverses a particular road segment, it uses its GPS to record the travel time. This information is then saved in a travel time report. This report stores the following fields: report and road segment identifiers, travel time, and timestamp. The report identifier provides a unique number for each report and is used for duplicate detection. The road segment identifier is used to match the report to a particular road segment. The travel time is the time measured by the given vehicle’s GPS. The timestamp is the time at which the report is produced. Reports are disseminated as described in section 3.1.

Each vehicle holds a digital map used for storing information about road segments and their travel times. The digital map of a vehicle \( v \) is a weighted graph \( G=(V,E) \) where \( V \) is the
set of vertices (intersections) and $E$ is the set of edges (road segments), with the weight of each edge $e$ being the travel time estimate of $e$ maintained by $v$. A number of properties are associated with each road segment. The properties of road segments that are of interest in this paper are: road segment identifier, road type, travel time estimate, list of $k$ most recent reports pertaining to the segment. The road segment identifier uniquely determines the particular road segment in the digital map. Road type indicates the physical characteristics of the particular road segment. There are several types that are defined (e.g. highways, arterial roads) and each corresponds to a different free-flow travel time on that segment. We call any non-highway segment a city street segment. The travel time estimate is the estimated time required to traverse the road segment. This estimate is calculated as the average travel-time of the $k$ reports the vehicle has received or generated with the most recent timestamps.

The following travel time update policy was used. For each road segment $s$, a vehicle keeps a sliding window of the $k$ youngest reports (i.e., the reports with the greatest timestamps) it has received in the digital map. In our experiments, the value of $k$ was set to 10. When a report $z$ regarding $s$ is received, $z$ is applied to update the travel time of $s$ as follows. If the timestamp of $z$ is smaller than the least timestamp in the sliding window (i.e., $z$ is older than the oldest report in the sliding window), then $z$ is discarded. Otherwise, the report in the sliding window with the least timestamp is replaced by $z$; the travel time of $s$ is updated to be the average of the reports in the new sliding window. After the travel time of $s$ is updated, the shortest path is recalculated. Thus, the shortest path is recalculated for each received report.
3.4.2 The Conditional Relevance Model

In this subsection we instantiate the conditional relevance model introduced in the context of travel time dissemination. For travel time dissemination applications, we define being relevant as follows: a report \( r \) received by vehicle \( v \) is \textit{relevant} if it changes the shortest path from \( v \)’s current location to its destination. A report thus becomes a positive example if the report changes the shortest path of the recipient vehicle. Otherwise, it is a negative example. In the rest of this subsection we complete the instantiation by specifying and justifying the attributes used for learning the conditional relevance model.

Due to the spatio-temporal nature of the travel time reports, the most obvious attributes to include for the conditional relevance model relate to time and space. To capture the temporal aspect of the reports we will define the \textit{age} of a report. The \textit{age} of a report \( r \) is the difference, in seconds, between the current time and the time at which \( r \) was created. To capture the spatial aspect, we introduce a distance measure that will be defined as follows: the \textit{distance} of a report \( r \) contained by vehicle \( v \) with digital map \( DM \) about road segment \( rs \) is the shortest travel time, in seconds, from \( v \)’s current location to the mid-point of the \( rs \) when the weight is the free-flow travel time for every road segment in \( DM \).

Now we justify the selection of these attributes. Our justification methodology is to show that if a report \( R_1 \) is better than a report \( R_2 \) on one attribute but is the same as \( R_2 \) on the other attributes, then the relevance of \( R_1 \) is not lower than that of \( R_2 \), i.e. the probability that \( R_1 \) changes a vehicle’s shortest path is not lower than that \( R_2 \) does so. In order to isolate the effect of the duplication model and concentrate on the conditional relevance model, we assume that neither \( R_1 \) nor \( R_2 \) has been previously received by the vehicle. We denote the age attribute of a report \( R \) by \( R\text{.age} \), and the travel time estimate attribute of \( R \) by \( R\text{.travel-time} \).
In the following we justify the age attribute. Intuitively, when all attributes are fixed, a younger report is more relevant than an older one. This is because an older report is more likely to be older than the oldest report in the sliding window of the corresponding road segment and therefore is discarded by the travel time update policy. Thus the probability that the older report changes a vehicle’s shortest path is lower than that the younger one does so. This intuition is formalized by the following theorem.

**Theorem 3.4.2.** Consider a vehicle $v$ at a vertex $s$ at time $t$, and two reports $R_1$ and $R_2$ that pertain to the same edge $e$ of the road network. Assume that $R_1.\text{travel-time}=R_2.\text{travel-time}$, and $R_1.\text{age}<R_2.\text{age}$, and that if $v$ receives the single report $R_2$ at time $t$ then it changes $v$’s shortest path. Then if at time $t$ vehicle $v$ receives the single report $R_1$ instead of $R_2$, it changes $v$’s shortest path as well.

**Proof.** Since $R_2$ changes $v$’s shortest path, $R_2$ must be younger than the oldest report in the sliding window of $e$ (otherwise $R_2$ would have been discarded). Since $R_1$ is younger than $R_2$, $R_1$ must also be younger than the oldest report in the sliding window of $e$ and changes $v$’s shortest path. \[\square\]

Theorem 3.4.2 indicates that when the other conditions are fixed, a younger report is more relevant that an older one. In other words, if a vehicle has $R_i$ and $R_2$ and room to transmit only a single report, then it should transmit $R_i$ instead of $R_2$ because it will be at least as relevant to an arbitrary vehicle $v$. Due to Theorem 3.4.2, the ages of reports can be used to rank reports according to their probability of being relevant.

Similarly, it can be shown that under certain conditions, a report with a smaller distance is more relevant that a report with a longer distance. Intuitively, under uniformity assumptions, if a road segment $e$ is far from a vehicle $v$, then the probability that the
destination of $v$ is beyond $e$ is small, and therefore the probability that $v$’s shortest path passes
$e$ is small. Thus the probability that a report pertaining $e$ changes $v$’s shortest path is small. In
summary, both attributes can be said to be useful, yet it is not obvious how to combine both
attributes to achieve a better ranking. This is what makes the machine learning approach
useful.

Training examples are created using the RelevanceTrain algorithm, given by the
following pseudocode:

<table>
<thead>
<tr>
<th>Algorithm – RelevanceTrain (Travel Times)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input</strong> – $R$: Set of received reports by vehicle $v$</td>
</tr>
<tr>
<td><strong>Outputs</strong> –</td>
</tr>
<tr>
<td>$E_d$: Set of training examples for duplication model</td>
</tr>
<tr>
<td>$E_r$: Set of training examples for conditional relevance model</td>
</tr>
<tr>
<td>1. Select and remove report $r$ from $R$ in any order.</td>
</tr>
</tbody>
</table>
| 2. IF $r$ has been previously received by $v$,
  THEN discard $r$, create negative example and add to $E_d$, GOTO step 1
  ELSE Create positive example and add to $E_d$, GOTO step 1 |
| 3. Save the current digital map state and use report $r$ to update the digital
  map of $v$ using the previously describe travel time update policy. |
| 4. Recompute the shortest travel-time path from the current location of $v$
  to the destination.
  IF shortest path changes,
  THEN create positive example and add to $E_r$
  ELSE create negative example and add to $E_r$ |
| 5. Restore the digital map state to the previous state (before the travel
time update). |
| 6. IF $R$ is not empty, GOTO step 1 |
The algorithm works by first checking for duplicates. At that point either a positive or a negative example will be created for the duplication model. Then, if it is determined that the report is not a duplicate, relevance of the report is determined based on whether it would change the path of the vehicle. Based on this, a positive or negative report is created for the conditional relevance model.

After all the reports in $R$ are processed, any reports that were identified as non-duplicate in step 1 are applied to update the digital map of the vehicle. Note that while the example creation procedure does temporarily update the digital map, the update is rolled back after the examples are created. This is done so that the order of examining reports does not affect whether the report will create a positive or negative example.

For travel time dissemination, it is assumed that every vehicle is equipped with a positioning sensor. Every vehicle can therefore act as a feedback node. It is hence unnecessary to use model sharing in this application.

3.4.3 Evaluation

The purpose of this evaluation was to establish the feasibility of the machine learning method. This was accomplished by determining whether the learned model allowed the vehicles to make better route decisions.
In order to evaluate the usefulness of the learned model, the STRAW simulator was used to generate a scenario in which vehicles disseminate travel time reports periodically. The road network was a 6 km by 4 km region of downtown Chicago taken from the digital map published by the Geographic Data Technology Inc. (see Figure 12). 100 vehicles were deployed in the road network.

An open mobility model was assumed. In this model, vehicles are assumed to pass through the region, rather than travel within it. Thus each vehicle $v$ was placed at a random location on the boundary of the road network, and another random boundary location on the road network was selected to be the destination. Vehicle $v$ then moved from the origin to the
destination, along the shortest travel time path given its current digital map state. When the destination was reached, the vehicle was assumed to have left the boundary. Another destination was randomly selected and the vehicle was assumed to be new\textsuperscript{2}. Its digital map was thus reset to contain no travel time reports and its report database was emptied. The learned conditional relevance and duplication models were preserved. The justification for this is that a newly entering vehicle will also have a learned model that it has learned while moving outside of the region.

Out of the 100 vehicles, 10 were participating, meaning they broadcast and generated reports. The other 90 were non-participating and thus always followed the shortest free-flow travel time path. The simulation procedure was continued until 100 trips were made by each vehicle. A total of 1000 trips were thus made by the participating vehicles. At that time, the simulation ended.

We assumed that during the time of simulation, non-recurring traffic conditions exist, such as in the case of accidents on the road. To simulate these conditions, 40 slow-downs were initially introduced at randomly selected highway segments. For the slow-down segments, the maximum speed was set to be 3 km/h. Each slow-down lasted for a time period that followed an exponential distribution with a mean of 20 minutes. When a slow-down recovered, another slow-down was introduced at a randomly selected highway road segment. Thus at any point in time the number of slow-downs in the road network was fixed.

\textsuperscript{2} For purpose of calculating vehicle trips, each vehicle was labeled at the start of the simulation. This label was persistent throughout the simulation even though the vehicle was assumed to be a new vehicle once it reached its destination.
All the simulation parameters and their values are summarized in Table II. These parameters were used for all simulations, unless otherwise noted.

### TABLE II. DEFAULT SIMULATION PARAMETERS

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total number of road segments</td>
<td>1562</td>
</tr>
<tr>
<td>Simulated area</td>
<td>6×4 sq. km (see fig. 1)</td>
</tr>
<tr>
<td>Total number of vehicles</td>
<td>100</td>
</tr>
<tr>
<td>Number of participating vehicles</td>
<td>10%</td>
</tr>
<tr>
<td>Transmission range (Tr)</td>
<td>250 meters</td>
</tr>
<tr>
<td>Number of slow-downs</td>
<td>40 slow-downs randomly among highway segments</td>
</tr>
<tr>
<td>Maximum speed at slow-down segments</td>
<td>3km/hour</td>
</tr>
<tr>
<td>Mean of slow-down persistence time</td>
<td>20 minutes</td>
</tr>
<tr>
<td>Broadcast interval (Bi)</td>
<td>5 seconds</td>
</tr>
<tr>
<td>Report database size (RRsize)</td>
<td>100 reports</td>
</tr>
<tr>
<td>Broadcast size (Bsize)</td>
<td>10 reports</td>
</tr>
<tr>
<td>Total number of trips</td>
<td>1000</td>
</tr>
</tbody>
</table>

Bootstrapping of models was performed like in the parking application. The simulation with the default parameters was run for a total simulation time of 72 hours. The learned duplication and conditional relevance models were then saved and used as the initial models for all tests.

We compared four different ranking methods:

1. Ranking by age. (age)
2. Ranking by distance. (distance)
3. Ranking by $1/(\text{age}+\text{distance})$. This method was a heuristic used in the TrafficInfo method (Zhong et al., 2008). (TrafficInfo)

4. Conditional relevance probability output using the proposed online Naïve Bayes machine learning method with age and distance as attributes. (online-ML-age-distance)

In addition to the above ranking methods, three baselines were also tested: full-info, non-info, and unlimited-bandwidth. The baselines reflect theoretical upper and lower bound performance. Full-info is an ideal case where vehicles receive all the reports as soon as they are created and thus have the full available information; no bandwidth, memory, or transmission range limitations are considered. Non-info is a case when vehicles do not exchange any information. Unlimited-bandwidth is a case when the report database size is unlimited and the vehicles broadcast every report they have ever received or generated. The unlimited-bandwidth baseline shows the best achievable performance that can be expected in the mobile peer-to-peer environment when transmission range and connectedness are the only limitations. The difference between the unlimited-bandwidth and the full-info baselines is that in unlimited-bandwidth, the reports are disseminated periodically instead of instantly and that the dissemination is done within the transmission range of the disseminating vehicle.

Two metrics were used for evaluation. The first is the average trip time, which measures the time it takes to reach a destination averaged across all vehicles and trips. This metric shows how the prioritization scheme affects the decision choices of vehicles. Note that because travel times change over time, having more accurate information will not always lead to better route decisions.
To measure the fidelity of the disseminate information that a given ranking method will provide, we introduce the average travel-time fidelity metric. This metric was collected as follows. Every 10 seconds, for each vehicle, the absolute value of the difference between the travel time along the shortest path according to vehicle’s current information and the path according to full information digital map was calculated. A full information digital map is defined as a digital map which is updated with every report ever created, as soon as it was created. The difference values are then averaged at the end of the simulation.

3.4.4 Results

In this section we show how the machine learning based method was able to combine two attributes which are known to be relevant, in order to improve performance. Figure 13 shows the performance of the different ranking methods when varying the broadcast size. As would be expected, the performance of all the ranking methods generally improved as broadcast size increased. The machine learning method maintained a lead in the performance across all the broadcast size values. In comparison, the TrafficInfo method of combining the two attributes did not significantly improve performance over using individual attributes, although it did provide a marginal improvement at small values of the broadcast size. A similar result occurred when the number of slow-downs and transmission range were varied (see Figures 14 and 15). The average travel-time fidelity results were proportional to the average travel time results for all the ranking methods. A figure showing the results of these tests was hence omitted.
Figure 13. Average travel times for range of broadcast sizes
Figure 14. Average travel times for range of number of slow-downs

Figure 15. Average travel times for range of transmission range values
3.4.5 Using Additional Attributes to Improve Performance

Aside from age and distance, there are also additional attributes that may be used to capture the relevance of a report. In this section, we used two additional attributes for the machine learning: road type and percentage of shortest paths. The road type can be either a highway or city-street segment, depending on its free-flow travel speed. Given a road network, \( RN \), a vertex \( s \) in \( RN \) and a road segment \( rs \), the percentage of shortest paths \( \%SP(RN,s,rs) \) is the number of shortest paths starting from \( s \) to all possible vertices in \( RN \) that pass through \( rs \), divided by the total number of all possible vertices. Figure 16 shows the performance when all four attributes are used for the conditional relevance model. The result using all four attributes is labeled as online-ML-all attributes.

As can be seen in the figure, the use of the additional attributes provides a significant improvement of the average travel time. Additionally, for a broadcast size of 40, the performance comes close to the unlimited-bandwidth benchmark. This means that the online machine learning method provides near optimal ranking at that point. As with the previous tests, the average travel-time fidelity results were similar to average travel time results.

---

\(^{3}\) We used the midpoints of road segments as possible vertices.
3.4.6 **Comparison of Machine Learning Methods through Offline Learning**

There are generally two ways in which machine learning can be performed: *online* or *offline*. In the online method, nodes continually learn the model by using the incoming reports as training examples. The most current model would then be used by each vehicle to estimate the relevance value of reports. In the offline method, there are two stages. In the first stage, nodes only gather the training examples without modifying their models. After a period of time, the examples are gathered and fed to the machine learning process. After this, the learned model is used by all nodes in the network. The advantage of this method is that nodes
would not have to incur the overhead of the machine learning algorithm. The disadvantage is that the learned model cannot adapt to changing situations. Nevertheless, since in the online case, nodes initially have no model, the offline method is useful for learning the model a priori, thereby providing a way of bootstrapping. Also, the offline method can be used to analyze which attributes should be used for learning and which machine learning algorithm is best for the given application. Therefore in this section, several machine learning methods are compared through offline learning. Work done for this purpose relied on a modified model of the travel time dissemination application. The main difference in this model is the travel time update method. The next subsections will describe the updated method, present the methodology for gathering offline learning examples, and show how the various machine learning algorithms performed.

3.4.6.1 Travel Time Update Method

In this method, vehicles update the travel times for the last 5 minute period. For each road segment in the digital map, a list of report for the most current period is stored. The travel time estimate is the average of the times in these reports. Initially, all vehicles have the free-flow travel time as the estimate for every road segment. After every 5 minute interval ends, all reports in the database that have been received in that interval are used for updating the travel time. Before updating, the reports are first analyzed based on their period number, which identifies the interval in which the report was generated. Period numbers are integers assigned consecutively for every 5 minute interval. For each road segment, the period number of all reports for the segment is recorded and only the reports pertaining to the maximum period number are kept. All other reports are discarded.
The way the report is used for updating depends on the relation of its period number to the period number contained in the vehicle’s digital map for the given segment. There are three cases:

1. Report’s period is smaller than the digital map’s. The report is thus discarded since it contains old information.

2. Report’s period is greater than the digital map’s. The report then replaces all previously received reports for the segment. The period in the digital map becomes the report’s period.

3. The periods are equal. In this case, we will first make sure the received report is not a duplicate. If it is, it will be discarded. Otherwise, it will be added to the report list for the given road segment and the travel time estimate will be recalculated by averaging all reports’ travel times in the list.

This updating is done for every report received in the last 5 minutes. At the end, the vehicle will recalculate the shortest travel time path to its destination.

3.4.6.2 Training Example Generation Procedure

The offline learning was done using the STRAW simulator, using several epochs. Each epoch consisted of a single road network and a group of vehicles, each randomly placed and having a random destination. Every vehicle initially started with their digital map containing free-flow travel times for every road segment. In each epoch, a set of travel time reports is created about a single, randomly chosen, road segment. Each report in the set has its travel time set to a random number, chosen uniformly from 0 to the free-flow time for that segment. The time period of each report varies from 0 to 100. The number of reports is thus 101, with
the first report having time period of 0, the next report 1, etc. The current time period is then set 100. Therefore, the ages of the created reports range from 100 to 0.

After the reports are created, each vehicle chooses a random number between 0 and 100 to serve as the period number for the given road segment in its digital map. While the travel time for that segment will still be the free-flow time, the use of the random period number will allow learning of the effects of age since the travel time updates are dependent on the relation of the report’s period to the current period. Once the period is chosen for the vehicle, the 101 reports are then used to independently update the digital map of that vehicle. This means that after each update is performed, the digital map is returned to its original state.

During each update, it is determined whether the shortest path of the vehicle would have changed as a result of the update. If so, a positive example is created, otherwise the example is labeled as negative. The offline learning procedure is outlined below.

<table>
<thead>
<tr>
<th>Algorithm: Offline learning for single region</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> R, road network within region w/ free-flow travel times</td>
</tr>
<tr>
<td>n, number of vehicles</td>
</tr>
<tr>
<td>e, number of epochs</td>
</tr>
<tr>
<td><strong>Output:</strong> L, List of labeled learning examples</td>
</tr>
<tr>
<td>For each epoch 1..e:</td>
</tr>
<tr>
<td>1. Randomly place n vehicles on road network R</td>
</tr>
<tr>
<td>2. Randomly choose a segment s in R</td>
</tr>
<tr>
<td>3. Randomly choose a travel time t between 0 and free-flow travel time on segment s</td>
</tr>
<tr>
<td>4. For each vehicle 1..n:</td>
</tr>
<tr>
<td>a. Choose a random period number p between 0 and 100</td>
</tr>
<tr>
<td>b. Create a travel time report about segment s with a free-flow travel time and period p and use it to update vehicle’s digital map</td>
</tr>
<tr>
<td>c. For period number Pi: 0..100:</td>
</tr>
<tr>
<td>i. Create a travel time report about segment s with travel time t and period Pi and use it to update vehicle’s digital map</td>
</tr>
<tr>
<td>ii. Recalculate the vehicle’s path to destination given current state of digital map. If path has changed, set label to 1, otherwise 0.</td>
</tr>
<tr>
<td>iii. Create learning example consisting of age=100-p, distance=free-flow travel time from vehicle’s location to segment s, roadType=(roadType of s), label</td>
</tr>
<tr>
<td>iv. Restore previous digital map state.</td>
</tr>
</tbody>
</table>
3.4.6.3 **Comparison of Machine Learning Algorithms**

The Weka learning toolkit (Hall et al., 2009) was used for evaluation of various machine learning algorithms. The training data consisted of a set of learning examples gathered using the offline learning procedure for two regions from the city of Chicago.

![Figure 17. Region 1 road network.](image)
The road networks of these two regions are shown in Figures 17 and 18, respectively. Region 1 is approximately 2.76 sq. mi. of northwest Chicago, while region 2 is about 6.75 sq. mi. from Chicago’s south side.

For each region, the offline learning procedure was used with 25 epochs. 100 vehicles were used for region 1 and 250 for region 2. The examples output from both regions were combined into a single data set. Since the number of negative examples far outnumbered the positive ones, the SpreadSubsample Weka routine was used to downsample the negative examples. The result was a data set with 7677 positive and 7677 negative examples. This data set was then input to the following machine learning algorithms:

- Naïve Bayes (NaiveBayes Weka implementation)
• Logistic Regression (Logistic Weka implementation)
• Support Vector Machines (SMO Weka implementation with buildLogisticModels enabled)
• Artificial Neural Network (Multilayer Perceptron Weka implementation)
• Decision Tree (J48 Weka implementation)

Each of the algorithms was used with the default parameters used by Weka, with exceptions as stated above. The testing of learning algorithms was performed using a 10-fold cross validation method.

Figure 19. Accuracy of various machine learning algorithms.
Figure 19 shows the accuracy that resulted from using each of the 5 algorithms. All algorithms, with exception of decision trees, performed similarly with an accuracy of approximately 83-84%. The decision trees had the best accuracy of 96.22%. It should be noted though, that while on one hand, the decision trees achieved extremely high accuracy, the resulting tree is very complex and not useful for report prioritization. The reason is that most of the leafs in the tree contained only one class of examples, which meant that almost every report would have a relevance value of 0 or 1. This is counterintuitive, given that two reports with different values of age, distance, or road type should be given a different probability of changing a vehicle’s path. On the other hand, the other machine learning algorithms performed worse than decision trees, but were able to capture the intuition behind the attribute relationships to the probability of changing a vehicle’s path. One of the most understandable of these algorithms was the logistic regression model. It assumes that the relevance value of a report is a linear combination of the attribute values and finds the weights of the attributes that best fits the data. The model resulting from the used dataset was the following:

$$U = -0.0322 \times \text{age} - 0.02 \times \text{distance} + 0.3885 \times [\text{road=highway}] - 0.3885 \times [\text{road=city street}] + 4.9053$$

(3.7)

As can be seen, this model predicts that an increase in age will result in a decrease in the probability of changing paths. This follows the intuition, since the travel time update policy discards many old reports. Similarly, the distance also varies inversely with relevance value. As could have been expected, the model also predicts more changes with highway segments than city streets. The approximately 80% accuracy achieved by the logistic regression method,
coupled with the easy understanding of the resulting model, makes it the most promising in the use for prioritization.

Since the makeup of the road network might change the weights found in logistic regression, separate models for the two regions were also build to determine how the model would be affected. The results showed that both age and distance attributes did not vary much from the combined model. The age weight was determined to be 0.0313 for region 1 and 0.0336 for region 2. The distance weights were 0.0186 for region 1 and 0.0222 for region 2. The road type weight did significantly change between the two regions, with 0.7213 for region 1 and 0.0391 for region 2. This indicates a strong relationship between the road network and the road type of the segment on the probability of a report changing a vehicle’s path. The constant factor between the two regions was relatively the same with -4.7806 for region 1 and -5.1552 for region 2.

While the use of logistic regression offered good interpretability, for the purpose of learning relevance in an online environment, the Naïve Bayes method can be considered the best option. The reason is that it allows for updates to be done in an online fashion easily, while matching the logistic regression model in terms of its accuracy.
CHAPTER 4

EMERGENCY ELECTRONIC BRAKE LIGHTS

In this chapter, we present the application of the proposed platform to the EEBL application, which is one of the first and the most thoroughly studied VANET safety applications that has been proposed (Carter, 2005; Carter and Chang, 2009; Ahmed-Zaid et al., 2011a; Ahmed-Zaid et al., 2011b). In 2005, the National Highway Traffic Safety Administration released a document that identified eight potential safety applications that utilize the Dedicated Short-Range Communication (DSRC) technology (Carter, 2005). The applications were selected based on the potential safety benefits that they provide. Among these applications, the EEBL application was determined to be one of the three applications to possess a high benefit potential. EEBL was defined as an application that alerts drivers of any hard braking that is done by vehicles in front of them. The idea was to extend drivers’ visibility through the emergency brake notifications. This approach was described as most helpful in situations where the visibility is limited, such as in adverse weather conditions.

EEBL works by vehicles disseminating a report each time they perform emergency braking, which occurs when the deceleration rate equals a predefined value. When a report arrives at a vehicle, the system checks whether the information contained in the report is relevant to the driver. Based on this, the driver can then be given a warning.

Current approaches for determining relevance in EEBL have been simple, based either on rules (Carter, 2005; Zang et al., 2008) or heuristics (van Arem et al., 2003). For example,
Carter suggested a warning could be given based on the vehicle’s lane (Carter, 2005). However, using lane information as the only factor would mean that all drivers of vehicles in the same lane would see the EEBL warnings, while drivers in the adjacent lanes would never receive a warning. For vehicles in the same lane, this could result in many unnecessary EEBL warnings to be shown to the drivers. Over time, this may desensitize the drivers and result in drivers ignoring the EEBL system. Conversely, for vehicles in the adjacent lanes, the EEBL systems might fail to warn drivers of potentially dangerous situations and thus reduce the safety benefit of the application.

In order to minimize the potential for driver desensitization while maintaining the safety benefit of the application, the EEBL warnings need to be shown to those drivers and only those drivers that would consider the warning relevant. This can be accomplished by restricting the warnings to those vehicles for which drivers need to perform emergency deceleration. One way this could be done is by calculating the required deceleration for the vehicle to avoid a collision, given certain report attributes, such as the distance, the vehicle speed, or the vehicle density. An example of such an approach was proposed for a collision avoidance system (Galler and Asher, 1995). However, the developed equations depend on the knowledge of other vehicles’ information (e.g. position and velocity), which is usually not known in EEBL. Even when all the information is known, the equations do not take into account normal human driver behavior and might result in drivers to consider the warnings as unnecessary. It may also be difficult to come up with the proper equations that would take into account all the relevant factors, such as road lane information or weather conditions.

In this chapter, we compare two different ways of determining the relevance for the EEBL system. One approach, the analytic method, is adapted from previous work (Galler and
Asher, 1995). It uses an analytically derived formula that tries to estimate the minimal safety gap a vehicle would need to avoid a collision. The alternative is our proposed method of using machine learning techniques for learning the relevance of reports. In this method, vehicles check whether emergency braking was done within a fixed amount of time after receiving each report. Based on this, training examples are created for a machine learning process which learns a report relevance model. The learned model can then be used to determine the likelihood of an arbitrary report being relevant. The decision to warn the driver is then based on this likelihood, which depends on several factors, such as the distance between the reporting and the receiving vehicles, the density of the vehicles on the road, and the velocity of the vehicles. While the focus of this research is on the theoretical performance of an EEBL system, for real deployments of EEBL technology, the learning of the model can be initially performed with simulations. This would allow the EEBL device to be used immediately. The models can then be subsequently modified through generation of additional training examples. Therefore, the models can be fine-tuned toward individual vehicles or drivers.

The main advantage of our proposed method is the ability to easily combine individual factors that affect the relevance of reports. Moreover, through simulations, we show that the method is able to significantly reduce the number of warnings that drivers would perceive to be unnecessary, while maintaining the safety benefit of the EEBL application in terms of lowering the number of collisions.

The machine learning method can also be used for relevance estimation in other transportation safety applications. For example, in a pedestrian warning application, the vehicle can communicate with a pedestrian holding a smartphone and determine whether the
driver should be warned about the pedestrian crossing the street. Another example could be a curve speed warning device in which drivers are warned if they should reduce their speeds for the curve ahead. In this case the learning could be based on whether other vehicles had to reduce their speeds for curves with similar attributes.

The rest of this chapter is structured as follows. In the next section, we discuss our assumptions about the environment for the EEBL application. Section 4.2 describes the EEBL application itself and section 4.3 presents the two approaches for relevance estimation. Evaluation of the different methods is discussed in section 4.4. This is followed by a summary in section 4.5.

4.1 Environment

The environment consists of a set of vehicles, controlled by human drivers. A subset of these vehicles is equipped with the following:

- On-board computer, with storage capability
- Positioning device (e.g. GPS, possibly augmented with a gyroscope and an accelerometer)
- Short-range wireless communication device (e.g. DSRC or Wi-Fi)
- Accelerometer
- Warning indicator
- Digital map with lane information

We assume that the storage of each vehicle has an unlimited capacity and that the positioning device is accurate at all times. The short-range wireless communication device is used to transmit data to nearby vehicles within its transmission range. The transmission range
is the maximum distance between a sending and a receiving vehicle in which data can be received. We assume that this range is fixed and is the same for all vehicles. Not all reports that are sent may be received. We hence define the delivery probability, which is the probability that a report sent by one vehicle is received by another vehicle within its transmission range. It may take time for a report to be transmitted from one vehicle to another. The time required for the transmission of a report is the transmission delay.

The accelerometer on the vehicle senses the deceleration (in g-forces) at any given point in time. The warning indicator is used to warn drivers about the need to brake. The digital map contains information about the road on which the vehicle is driving, including the number of lanes and their exact positions. This is necessary for determining the lane number of each vehicle.

We assume that a subset of vehicles are participating, meaning they are equipped with all the devices listed above. The remaining vehicles are non-participating, meaning without the equipment, including the warning indicator or the communication devices. The ratio between the participating vehicles and the total number of vehicles is called the participation rate. In this paper, we consider cases of both full (100% participation rate) and partial participation (0% < participation rate < 100%).

In emergency situations, vehicles may perform certain actions to try to avoid a collision with the vehicle in front of it. Typically, the first driver reaction will be to press hard on the brakes, which will cause the vehicle to decelerate with a maximum deceleration rate, which we call $T_{severe\_brake}$. Observe that $T_{severe\_brake}$ will vary based on the vehicle type, load, and weather conditions. A vehicle emergency decelerates when its deceleration equals $T_{severe\_brake}$. 
4.2 **EEBL with the Relevance Estimation Application**

Our system design for the EEBL application consists of three elements:

- Report creation
- Report dissemination
- Report relevance estimation

Reports are created by the vehicles whenever they emergency decelerate. After reports are created, they are disseminated to the nearby vehicles using the short-range communication device. Once a report arrives at a vehicle, the relevance of the report is determined. If the report is found to be relevant, the warning indicator is turned on to alert the driver. Otherwise, no warning is shown. Note that in the simple EEBL application there is no relevance estimation, i.e. the warning indicator is turned on whenever the vehicle receives a report. We will now discuss the details of each of the design elements individually.

When a vehicle emergency decelerates, it generates an *emergency brake report*, each of which contains information about the time and place when the emergency deceleration occurred. While the vehicle continues to emergency decelerate, new reports are generated every 0.1 seconds. Each emergency brake report includes the following fields:

- Location
- Timestamp
- Vehicle speed
- Vehicle length
- Road and lane identifier
All these fields, except for the road and lane identifier, are included as part of the standard Basic Safety Message (Ahmed-Zaid et al., 2011b). The location is recorded at the time of report’s generation using the positioning device and is stored as coordinates. The timestamp is the current time when the report was generated and the vehicle speed is the speed of the vehicle at that time. The road and lane identifiers correspond to the road and lane on which the vehicle is traveling. We assume that this information can be generated from the digital map and the location information using techniques such as the one described in (Dao et al., 2008). However, please note that while our proposed methods can utilize lane information to increase the performance, they may also be used without this knowledge.

Reports are disseminated using the short-range communication device through broadcast communication. In broadcast communication, the sent reports can be received by all neighboring vehicles. Neighboring vehicles are those within the transmission range of the sender. New reports received by a vehicle are immediately rebroadcast to its neighbors. A vehicle that receives an emergency brake report is called a receiving vehicle.

When a report is received, we determine its relevance to the receiving vehicle. If the report is determined to be relevant, the warning indicator is turned on. If a report is found not to be relevant, no warning is given. The determination of the relevance can be done through one of the methods presented in the next section.

\[^4\] Note that if the update frequency of the positioning device is lower than 0.1s, then the locations in the subsequent reports may be identical.

\[^5\] We have also performed tests without rebroadcasting. The results of those tests are presented in section 4.5.10.
4.3 **Relevance Estimation Methods**

In this section, we will present two methods for estimating the relevance of emergency brake reports. The first method, discussed in the next subsection, determines relevance based on an analytically derived formula. The second, discussed in the following subsection, is our proposed method that utilizes machine learning techniques. Both methods utilize the same information for determining the relevance. The information used by the methods is one that would be expected to be available in an EEBL system. This includes information about the vehicle that generated the emergency brake report and the receiving vehicle. However, specific information about other vehicles is not assumed to be known.

4.3.1 **The Analytic Method**

This method determines the relevance based on an estimate of the minimum safety gap. We define the *minimum safety gap* as the minimum distance in between the receiving vehicle and the vehicle in front of it, required to avoid a collision. This is based on an assumption that if the receiving vehicle is on the same lane as the one that generated the report, then at the time the report is received, the vehicle in front of the receiver will immediately emergency decelerate until stopped. Given that assumption, if the gap between the two vehicles is less than the minimum safety gap, a warning will be shown to the driver (i.e., the report will be considered relevant). Otherwise, no warning will be shown (i.e., the report will be considered irrelevant).

The minimum safety gap is dependent on several factors, including the position and the velocity of the receiving vehicle and the vehicle in front of it, and the driver reaction time. In EEBL, the information about the vehicle in front is not known. However, we can approximate
its position using the information contained in the report along with road density information. The density can either be calculated from the knowledge of the current travel times or it can be estimated locally from wireless communication device signals and the knowledge of the participation rate (see (Panichpapiboon and Pattara-atikom, 2008)). We assume all other properties of the front vehicle, such as its length, velocity, or maximum deceleration, are equal to those of the vehicle which generated the report. For the driver reaction time, we used a value of 1.0 seconds, based on a previous study (Johansson and Rummer, 1971). Using these assumptions, we can calculate the minimum safety gap based on the equations derived by Galler and Asher (Galler and Asher, 1995).

To illustrate how the minimal safety gap is determined, consider the scenario shown in Figure 20. There are multiple vehicles driving on a straight road with a single lane. One of the vehicles, which we will call vehicle A, receives a report from one of the downstream vehicles, which we will call vehicle C (see figure below).

---

6 This may obviously not hold, so implications are discussed in section 4.5.3.1.
Assume that the vehicle directly in front of $A$, vehicle $B$, will emergency decelerate immediately after $A$ receives the report. When this occurs, we want to estimate whether the distance between $A$ and $B$ is less than or equal to the minimum safety gap. To do this, we have to estimate the position of $B$ using the position of $C$ and the density of the vehicles in between $A$ and $C$. Let $s_x(t)$, $v_x(t)$, and $a_x(t)$ be the position, velocity, and acceleration of vehicle $X$ at time $t$. The position of the vehicle is its distance from the beginning of the road segment. Note that deceleration will be indicated by a negative value of $a_x(t)$. Assuming equally spaced vehicles in between $A$ and $C$, the initial position of $B$ can be estimated from the density as follows:

$$s_{x}(0) = \frac{s_{c}(0) - s_{d}(0)}{[s_{c}(0) - s_{d}(0)] \cdot \text{density} + 1} + s_{d}(0)$$

(4.1)

We assume the initial velocity and acceleration of $B$ equals that of $C$. We also assume that each driver has a fixed reaction time of $RT$ seconds and that within the initial $RT$ seconds, the acceleration of the vehicle will be 0. Therefore vehicle $A$ will continue at its initial velocity for $RT$ seconds. This means that it may collide with vehicle $B$ within this time. If this is the case, then the gap is smaller than the minimum safety gap. To find whether a collision will occur, we have to consider two cases: the collision occurs before $B$ stops or the collision occurs after $B$ stops. Vehicle $B$ stops at time:

$$t_{b-stop} = -\frac{v_{B}(0)}{a_{g}(0)}$$

(4.2)

The first case occurs when the time of collision ($t_{\text{crash}}$) occurs before $B$ stops. The time of collision for this case is given by:
In this equation, $\Delta v_{AB}(0)=v_B(0)-v_A(0)$ and $\Delta s_{AB}(0)=s_B(0)-s_A(0)-l_b$, where $l_b$ is the length of vehicle $B$. The length of $B$ can either be approximated by the average vehicle length or we can assume, as we did in our experiments, that it equals the length of $C$.

For the second case, when a collision would occur after $B$ stops, the time of collision is then given by:

$$t_{\text{crash}} = \frac{-\Delta v_{AB}(0) - \sqrt{\Delta v_{AB}(0)^2 - 4 \cdot a_B(0) \cdot \Delta s_{AB}(0)}}{a_B(0)}$$  \hspace{1cm} (4.3)

If $t_{\text{crash}}>RT$, then it is possible for vehicle $A$ to start decelerating to try to avoid a collision. To calculate the rate required, we first assume the deceleration of a vehicle is constant during braking. A collision would occur if the velocity of $A$ is greater than that of $B$ at the point when the position of $B$ is equal to that of $A$ plus the length of $B$. Under these assumptions, the required deceleration ($a_d(0)$) to avoid a collision is:

$$a_d(0) = \frac{-v_A(0)^2}{2 \cdot \Delta s_{AB}(0) - \frac{v_B(0)^2}{a_B(0)} - 2 \cdot v_A(0) \cdot RT}$$  \hspace{1cm} (4.5)

Since the idea of EEBL is to alert drivers when emergency braking is required, we compare the required deceleration for collision avoidance to the deceleration defined as emergency braking ($T_{\text{severe_brake}}$). A warning is given to the driver when $a_d(0)<T_{\text{severe_brake}}$. Note that both $T_{\text{severe_brake}}$ and $a_d(0)$ will be negative for deceleration. This implies that the deceleration required equals or exceeds the vehicle’s maximum deceleration; in other words, the current gap is smaller than the minimum safety gap.
4.3.2 **Observe and Learn Approach**

In this section, we present our relevance estimation method based on machine learning techniques. The general idea behind this method is to use the received reports as an input to a supervised machine learning algorithm. The objective is to learn a *relevance function*, which maps the reports to a value between 0 and 1, with 0 being completely irrelevant and 1 being completely relevant.

The method works in two stages: the learning stage and the usage stage. In the learning stage, the relevance function is instantiated using machine learning techniques. In the usage stage, the learned relevance function is used to decide whether to turn on the warning indicator. Every vehicle starts in the learning stage, which proceeds as follows. First, the warning indicator is disabled to allow us to observe how a driver normally reacts after a report is received. Then, for every report received by a vehicle, we calculate values for a set of report attributes and determine whether the report was relevant. We will discuss the selection and calculation of the attributes in section 4.3.2.1.

We check whether the report was relevant by monitoring the vehicle’s behavior after the report was received for a specified fixed amount of time, called the *Reaction-delay*. The *Reaction-delay* is a parameter of the method that is the length of the period of time \( P \) after which the method determines whether or not the report was relevant. \( P \) starts when the report is generated and its optimal length, *Reaction-delay*, was found experimentally (see section 4.5.2). Having explained the *Reaction-delay*, we will now define a relevant report as follows: A report \( r \) received by a vehicle \( v \) is *relevant*, if the driver of \( v \) performs an emergency deceleration within *Reaction-delay* seconds after the report was received.
This definition stems from the intuition that a temporal correlation (occurrence within *Reaction-delay*) between receiving a report and undertaking emergency action implies the report necessitated the emergency deceleration. Therefore the report can be considered relevant.

Once the attribute values are calculated and the relevance of the report determined, the system then generates a training example. A *training example* consists of a set of report attribute values and a label indicating its relevance. Labels can be either positive or negative. A training example is assigned a positive label when the report was determined to be relevant, and is then called a *positive training example*. Similarly, a training example is assigned a negative label when the report was not relevant, and is called a *negative training example*. The training examples that are generated are added to a training example set stored by each vehicle.

Once a sufficient number of training examples is generated (we used thousands in our experiments), the set of all stored examples is input to a machine learning algorithm, which uses the training examples to learn the relevance function. Subsection 4.3.2.2 describes the specific machine learning algorithms we have utilized. After the machine learning algorithm finds the relevance function, the method proceeds to the usage stage.

In the usage stage, the warning indicator is enabled and new training examples are no longer generated. Instead, when a report arrives at a vehicle, the report attribute values are calculated and the learned relevance function is used to calculate the relevance of the report. The decision to turn on the warning indicator then depends on a fixed threshold called *T*\textsuperscript{warning}. When the relevance value of the report exceeds *T*\textsuperscript{warning}, the warning indicator will then turn on. Otherwise, if the value is less than or equal to *T*\textsuperscript{warning}, the report will be
ignored. The determination of the exact value for $T_{\text{warning}}$ will affect the trade-off between safety and the number of false warnings. If set low, the warnings will appear more frequently, possibly causing many false warnings to be shown. Over time, this can lead to driver desensitization. Setting the threshold high will have the opposite effect. We discuss the setting of the $T_{\text{Warning}}$ threshold in section 4.5.1.

Note that it is possible to continue to learn in the usage stage. However, because the EEBL warnings will affect drivers’ behavior, this may introduce a bias in the learning. We will therefore leave this issue as part of future work. In the next subsection, we discuss the selection and the calculation of the report attributes. In section 4.3.2.2, we present the two machine learning techniques we utilized in our experiments.

### 4.3.2.1 Attributes

Attributes are the features which are used for the machine learning. They are derived from the raw data contained in the report (e.g. vehicle position and velocity) and other information (e.g. road density) that would be known to the receiving vehicle. When choosing attributes, care must be taken to find attributes which best predict emergency deceleration. The attributes we used for learning were chosen based on the factors that affect the drivers’ decision to initiate emergency deceleration. We identified four such attributes:

- Temporal distance, $d$ (seconds) between the locations at which the report was generated and received. It is the time needed for the receiving vehicle, traveling at its current velocity, to reach the location at which the report was generated. The probability of emergency deceleration by the receiving vehicle will increase as the temporal distance decreases, and vice versa.
• Density of vehicles, $\rho$ (vehicles per 1 km). As the density increases, the average headways between vehicles decrease. This will in turn increase the likelihood of emergency deceleration.

• Difference in velocities, $v_d$ (meters per second) between the receiving vehicle and the vehicle that generated the report. Intuitively when the receiving vehicle is traveling faster than the one that generated the report, the likelihood of emergency deceleration will be higher.

• Lane Offset, $l_0$. Defined as the number of lanes separating the receiving vehicle and the vehicle that generated the report. This attribute will be used if lane information is known to the vehicles. A lower lane offset will increase the likelihood of emergency deceleration.

Alternative attributes may have been used, but our experiments showed the above four have resulted in the best performance. The tests we performed replaced the temporal distance with a spatial distance, which is the difference in the positions between the vehicle that received the report and the one that generated it. Similarly, the ratio of velocities replaced the difference in velocities. These tests showed the performance in terms of the average number of collisions was degraded.

We assume that the values of the attributes are known when the report is received. The temporal distance attribute can be easily calculated based on the location data provided in the report and the current vehicle speed. This requires only two simple operations, subtraction and division. Similarly, the difference in velocities can also be simply calculated from the reported vehicle speed and the current vehicle speed involving one subtraction. Lane offset can be derived from the reported lane identifier, again using one subtraction. The calculation
of the density was discussed in the previous section (see section 4.3.1) and is performed independently and before the processing of a report. Therefore, when a new report arrives, its processing involves less than 10 machine instructions and therefore we assume that the delay in processing the report is negligible.

Although we have used only four attributes, additional attributes, such as those related to weather or road conditions, can also be used to augment the learned relevance model. Another attribute that may have been used is the age of the report, defined as the difference between the time the report was generated and the time it was received. However, we did not include this attribute because in our simulations, we showed that even without the age attribute, the performance does not degrade when delays are present. It may however be used when significant delays are present, for example, when the EEBL application has to share bandwidth with additional applications.

4.3.2.2 Machine Learning Techniques

For EEBL, we have experimented with three different machine learning techniques for learning the relevance function: naïve Bayes, logistic regression, and Classification and Regression Trees. Each technique has its advantages and as will be shown later, the choice of which method is used could affect the performance of the system. The Naïve Bayes method was implemented using the Weka learning toolkit (Hall et al., 2009), same as for non-safety applications. This method was previously explained in section 3.2.3. The Classification and Regression Trees is a popular decision tree based machine learning technique, but our results showed it performed poorly in the key metric (the average number of collisions). We have
therefore omitted its discussion for the rest of this chapter. Therefore in this section, I will only present the logistic regression method.

The logistic regression method assumes that the probability of an emergency deceleration, which we use as the relevance function, fits a logistic function:

\[
P(v_p|p,d,v_d,l_o) = f(z) = \frac{1}{1 + e^{-z}}
\]  

(4.8)

, where \( z \) is a linear combination of the attributes \( \rho, d, v_d, l_o \):

\[
z = \beta_0 + \beta_1 \rho + \beta_2 d + \beta_3 v_d + \beta_4 l_o
\]  

(4.9)

The parameters \( \beta_1, \beta_2, \beta_3, \) and \( \beta_4 \) are the coefficients of the attributes and \( \beta_0 \) is the intercept value. The values of these parameters are found based on the training examples, typically using maximum likelihood methods. The method we have used is ridge estimators (Le Cessie and van Houwelingen, 1992). The learned equations for the one (4.10), two (4.11) and three lane (4.12) cases which we used in our experiments are given below (the Reaction-delay was set to 9 seconds):

\[
z = 2.815 + 0.04456 \rho - 0.000207d - 0.2787v_d
\]  

(10)

\[
z = -0.3588 + 0.1028 \rho - 0.000177d - 0.0962v_d - 0.2007l_o
\]  

(11)

\[
z = -0.2497 + 0.2822 \rho - 0.000179d - 0.0271v_d - 0.217l_o
\]  

(12)

To determine the significance of the attributes in each of the learned models, we approximated the 95% confidence intervals using the bootstrap method. The resulting intervals are listed in Table III. These tests showed that with the exception of the temporal distance (which includes zero in the interval), all the attributes are significant.
### TABLE III. APPROXIMATE 95% CONFIDENCE INTERVALS FOR LOGISTIC REGRESSION

<table>
<thead>
<tr>
<th>Model</th>
<th>Attribute</th>
<th>95% Conf. Interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Lane</td>
<td>Temporal Distance (d)</td>
<td>[-0.000619, 0.000103]</td>
</tr>
<tr>
<td></td>
<td>Difference in Velocities (vd)</td>
<td>[-0.332, -0.236]</td>
</tr>
<tr>
<td></td>
<td>Density of Vehicles (ρ)</td>
<td>[0.0364, 0.0552]</td>
</tr>
<tr>
<td>2 Lanes</td>
<td>Temporal Distance (d)</td>
<td>[-0.000592, 0.000198]</td>
</tr>
<tr>
<td></td>
<td>Difference in Velocities (vd)</td>
<td>[-0.100, -0.092]</td>
</tr>
<tr>
<td></td>
<td>Density of Vehicles (ρ)</td>
<td>[0.098, 0.108]</td>
</tr>
<tr>
<td></td>
<td>Lane Offset (lo)</td>
<td>[-0.265, -0.136]</td>
</tr>
<tr>
<td>3 Lanes</td>
<td>Temporal Distance (d)</td>
<td>[-0.000757, 0.000112]</td>
</tr>
<tr>
<td></td>
<td>Difference in Velocities (vd)</td>
<td>[-0.0308, -0.0225]</td>
</tr>
<tr>
<td></td>
<td>Density of Vehicles (ρ)</td>
<td>[0.271, 0.292]</td>
</tr>
<tr>
<td></td>
<td>Lane Offset (lo)</td>
<td>[-0.280, -0.159]</td>
</tr>
</tbody>
</table>

### 4.4 Evaluation Methodology

In our evaluation, we measured the effectiveness of the EEBL system in preventing vehicular collisions and measured the potential for driver desensitization to the warnings. The evaluation was done through simulations using a modified version of the Microscopic Traffic SIMulator (MITSIM) software (Yang and Koutsopoulos, 1996). The next subsection will describe the MITSIM simulator. Our modifications to the MITSIM software are then described in subsection 4.4.2. Subsection 4.4.3 explains our simulation environment and subsection 4.4.4 provides the evaluation procedure.

#### 4.4.1 Simulation Software

The MITSIM microscopic traffic simulator simulates vehicle movements through a road network using car-following logic. It allows for different vehicle and driver types to be
specified, which affect vehicle movements on the road. The simulation is done in discrete time steps of 0.1 seconds. Vehicles enter the network from the specified origin and traverse the network toward the specified destination. Vehicles are introduced into the network at a rate specified by the departure rate with either equal or random (Poisson distributed) spacing.

The departure rate affects the density of the vehicles on the road. The speed and the position of the vehicle entering the road is dependent on the traffic conditions and the speed limits on the given road and the required spacing to the vehicle in front. The required spacing represents the minimum safety gap drivers are willing to accept and it may be different for each driver. If the spacing is not sufficient at the given time step, vehicles to be placed on the road are put in a FIFO queue until they can enter the road.

The control of vehicle movements on the road is handled by three different cases, depending on the headway time to the vehicle in front. In the first case, the headway is greater than $h_{\text{upper}}$ and is called the free flowing regime. In that case, vehicles accelerate or decelerate to the target speed. The target speed depends on many factors, including the speed limit and the driver and the vehicle types. In the second case, the headway is in between $h_{\text{lower}}$ and $h_{\text{upper}}$, and the vehicle enters the car following regime. In this case, the vehicle tries to match the speed of the vehicle in front of it by applying a calculated amount of acceleration or deceleration. This amount is determined using an equation based on the General Motors car-following model. When the headway drops to below $h_{\text{lower}}$, the vehicle then enters the emergency deceleration regime. It then applies sufficient deceleration to increase the headway above $h_{\text{lower}}$. All the acceleration or deceleration decisions are postponed by the driver reaction time, which is specified by a random variable from a truncated normal
distribution. The $h_{\text{upper}}$ parameter may be different for each vehicle and by default ranges from 1.7498 seconds to 4.5979 seconds. The $h_{\text{lower}}$ parameter is fixed to 0.4 seconds.

Lane changes are handled by a random utility model that calculates a probability whether a mandatory or a discretionary lane change is to be performed for each vehicle. A mandatory lane change might occur when the current lane on which the vehicle is driving ends or the lane does not connect to the next link in the path. A discretionary lane change might occur if the driver of the vehicle is not satisfied with the conditions on its current lane and prefers the conditions on one of the adjacent lanes. The probability of making a discretionary lane change depends on several factors, including the vehicle’s current speed, its desired speed, and the speed of the vehicles on the adjacent lanes. When a driver decides to make a lane change, a gap acceptance model is used to check whether the lane change can be performed. The gap acceptance model is based on the relative speeds of the subject vehicle and the leading and lagging vehicles on the target lane.

4.4.2 Modifications to MITSIM

Since the original software focused mainly on evaluating traffic flow, it did not permit vehicle collisions. To prevent collisions, MITSIM allowed vehicles to exceed their specified maximum deceleration in certain situations. The maximum deceleration of a vehicle is speed dependent and ranges from 3.048 meters/second$^2$ to 10 meters/second$^2$. To allow collisions to occur, we therefore modified MITSIM by making sure maximum deceleration cannot be exceeded at any time during car following. As a result, vehicles may not have enough time to stop and collisions can occur. We also implemented the ability to stop one of the vehicles immediately, which allowed us to simulate a worst case emergency braking situation.
For the testing of adverse weather conditions, we modified the car following and the lane changing parameters based on a previous study (Sterzin, 2004), which provides calibration settings for rain conditions. This paper used data from a freeway corridor in Hampton Roads, Virginia to find simulation parameters that are significantly affected by rain conditions. The author found two lane changing and two car following model parameters to be significant. In addition, the desired speed distribution was also changed from fine weather conditions. The study was performed for driving conditions without incidents and did not assume decreases in road-tire friction. However, for our tests, we have reduced the maximum acceleration and deceleration values for each vehicle type by assuming a road-tire friction coefficient of 0.8.

The EEBL system was implemented by checking the deceleration of all vehicles during every time step. If one of the vehicles reached its maximum deceleration, the EEBL report was then sent to all the vehicles within the specified transmission range after a specified propagation delay. A vehicle then received the report with a probability equal to the set delivery probability. If the report was received by a vehicle which was upstream of the one that generated the report, the relevance estimator then decided whether a warning should be given to the driver. We assumed that drivers react to the EEBL warnings by braking with maximum deceleration. Therefore, if a warning was given, then after the driver reaction time, the deceleration of the vehicle was immediately set to its maximum deceleration. This was done regardless of the movement regime. Note that this is different than what is done in the emergency deceleration regime. In the emergency deceleration regime, an equation is used to calculate the deceleration required to increase the headway above $h_{lower}$, while when an EEBL warning is given, the deceleration will always be set to the maximum, regardless of the headway.
4.4.3 Simulation Environment

The environment consisted of a 3 mile long, single stretch of the road with one, two, or three lanes. The roads were setup as highway segments, unless otherwise noted. Vehicles entered the road at a rate specified by the mean departure rate. The mean departure rate is a random number from a normal distribution and was fixed for the duration of a single simulation run. Four types of vehicles were used, as specified by the default parameter settings of MITSIM. The inter-vehicle spacing was Poisson distributed with a mean rate equal to the mean departure rate. The speed limit and the free-flow speeds on the road were both set to 55 mi/h for the highway segments. For testing the local streets, a lower speed limit of 35 mi/h was used (see section 4.5.3.2). The driver reaction time was based on a truncated normal distribution with a mean of 1.0 seconds, a standard deviation of 0.25 seconds, a lower bound of 0.4 seconds, and an upper bound of 2.7 seconds. These values were based on the study done by Johansson and Rummer (Johansson and Rummer, 1971). All other parameter values were set to their MITSIM defaults (see Table IV for the listing of all parameter values).

Each run simulated a single incident on the road in which a particular vehicle, we call the *incident vehicle*, stopped immediately after traversing 99% of the road length. The lane on which the incident occurred, which we call the *incident lane*, was specified for each run. The incident lane for the two lane configuration was fixed to be the left lane, while the middle lane was used for the three lane configuration. The vehicle selected to stop was the first vehicle which reached the stopping point in the specified incident lane that was also at least the 100\textsuperscript{th} vehicle that entered onto the road. Waiting until at least 100 vehicles enter the road was done to allow the simulation to initialize. However, note that we have tested using the 1\textsuperscript{st},
10\textsuperscript{th}, 50\textsuperscript{th}, and 200\textsuperscript{th} vehicle instead of the 100\textsuperscript{th} as the incident vehicle, but the results did not change.

The stopping of the vehicle initiated a possible sequence of collisions. For each run, we recorded the number of collisions in each lane for the vehicles in the last two miles of the road. The first mile was not considered, because collisions close to the road entrance were typically unrealistic. This was due to the method that MITSIM uses for loading the vehicles onto the road. In order to focus on the collisions that occur after the incident, the EEBL system was initially turned off and then turned on once the incident occurred.

Each run was executed in either the learning or the usage modes. In the learning mode, emergency brake reports were disseminated, but no warnings were given to the vehicles. The vehicles thus followed their normal, MITSIM specified, car following behavior. For each report, a timer equal to the \textit{Reaction-delay} was started once the report arrived, and the attribute values were calculated. Once the timer expired, we checked whether emergency deceleration occurred during the last \textit{Reaction-delay} seconds and output a training example consisting of the attribute values and the appropriate label (positive or negative). In the usage mode, no training examples were output. Instead, vehicles were given warnings based on the value of the relevance given by the various estimation methods. These methods will be discussed in the next subsection.
### TABLE IV. PARAMETER VALUES USED IN SIMULATIONS

<table>
<thead>
<tr>
<th>Simulation Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulation time step</td>
<td>0.1 s</td>
</tr>
<tr>
<td>Road length</td>
<td>3 miles</td>
</tr>
<tr>
<td>Number of lanes</td>
<td>1 - 3</td>
</tr>
<tr>
<td>Road speed limit and free-flow speeds</td>
<td>35 (local street) and 55 mi/h (highway)</td>
</tr>
<tr>
<td>Mean departure rate</td>
<td>200-1200 vehicles/h/lane</td>
</tr>
<tr>
<td>Headway threshold between free-flow and car-following regimes ($h_{upper}$)</td>
<td>1.7498 s to 4.5979 s</td>
</tr>
<tr>
<td>Headway threshold between car-following and emergency deceleration regimes ($h_{lower}$)</td>
<td>0.4 s</td>
</tr>
<tr>
<td>Driver reaction time mean, standard deviation, lower and upper bounds</td>
<td>1.0 s, 0.25 s, 0.4 s, 2.7 s</td>
</tr>
<tr>
<td>Number of simulation runs for each test</td>
<td>100, 1000</td>
</tr>
<tr>
<td>Maximum deceleration</td>
<td>3.048 m/s^2 to 10 m/s^2</td>
</tr>
<tr>
<td>$T_{Severe\ Brake}$</td>
<td>$=$maximum deceleration</td>
</tr>
<tr>
<td>Reaction-delay</td>
<td>1s-11s</td>
</tr>
<tr>
<td>Transmission range</td>
<td>100m, 250m, 1000m, infinity</td>
</tr>
<tr>
<td>Delivery Probability</td>
<td>0% to 100%</td>
</tr>
<tr>
<td>Propagation Delay</td>
<td>0ms-500ms</td>
</tr>
</tbody>
</table>

#### 4.4.4 Evaluation Procedure

The evaluation of the relevance estimation methods was done in several steps. First, we found the appropriate values for the Reaction-delay and the $T_{Warning}$ parameters (section 4.4.4.2) utilized by the machine learning methods. During this procedure, we also trained the machine learning models. Second, we tested the potential for driver desensitization (section 4.4.4.3), utilizing the previously trained machine learning models. Fourth, we tested the safety benefit of the system by measuring the average number of collisions per simulation run (section 4.4.4.4). The methods which were compared are described in the next subsection.
4.4.4.1 **Compared Methods**

The following relevance estimation methods were compared in the evaluation:

- **noEEBL** – no report is relevant
- **simpleEEBL** – all reports are relevant
- **simpleEEBL-lane** – all reports with lane offset 0 are relevant
- **Analytic** – relevance calculated using the analytically derived formula based on the minimal safety gap
- **ML-NB** – relevance based on the model learned by the Naïve Bayes machine learning algorithm
- **ML-LR** – relevance based on the model learned by the logistic regression machine learning algorithm

In the first method, noEEBL, every report is assumed to be irrelevant, so this method reflects the conditions without the EEBL system. The second method, simpleEEBL, assumes all the reports are relevant and hence gives warnings to all vehicles. The third method gives warnings to only those vehicles which are driving within the same lane as the vehicle that generated the report. The fourth method uses the equations described in section 4.3.1. The last two methods utilize our proposed machine learning approach to learn a relevance model. Each of the methods used a different learning technique. ML-NB used the Naïve Bayesian technique, while ML-LR used logistic regression.

4.4.4.2 **Setting of the Reaction-delay and TWarning Parameters**

To find the best values to use for the two parameters of the machine learning method, we ran 100 simulation runs for each of the six departure rate settings (200-1200 veh/h/lane) in
the learning mode, using three different road lane configurations: one lane, two lanes, and three lanes. For these runs the delivery probability and the participation rate were set to 100%, the propagation delay was 0 ms, the transmission range was set to infinity. During each run, training examples were recorded from every vehicle and added to a set for the run. We then created the union of all the sets of runs with 1 lane, called 1set. Similarly, we created the union of the runs with 2 and 3 lanes, called 2set and 3set, respectively.

The runs were repeated for different values of Reaction-delay, from 1 second to 11 seconds, in 2 second increments, resulting in 15 training example sets total. This range of values was chosen because the Reaction-delay has to be greater than the human reaction time, in order to provide enough time for the driver to react. However, the value should be small enough to capture only the driver reactions that are related to the received report. Each one of the training example sets was then split 90%/10% among the training and the test sets. The training set was used to learn the relevance estimation model, while the test set was used to calculate the false and the missed warning rates. The false warning rate is the number of false warnings divided by the total number of warnings predicted by the learned model. A false warning was counted when the model predicted a warning for a negative training example (see section 4.3.2 for definition), meaning the warning should not have been given. The missed warning rate is the number of missed warnings, divided by the number of positive training examples. A missed warning was counted for every positive training example in which the relevance estimation method would not have given a warning based on the set of attribute values associated for that example.

The training of the models was done through the Weka Learning Toolkit software (Hall et al., 2009) using the corresponding Weka implementations with default parameters. Naïve
Bayes training was done using the NaiveBayesSimple implementation and logistic regression was done using the Logistic implementation.

With the machine learning approach, we can lower the false warning rate to an arbitrary level by controlling the $T_{\text{Warning}}$ parameter value. However, increasing this value will also result in an increase in the missed warning rate. Higher missed warning rates will likely decrease the safety benefit of EEBL because fewer warnings will be shown. Therefore we set the value of $T_{\text{Warning}}$ such that the false warning rate equals the missed warning rate. This achieves a balance between the false and the missed warning rates. The two rates can be made equal because as $T_{\text{Warning}}$ increases, the false warning rate decreases and the missed warning rate increases.

To find the optimal value for the Reaction-delay parameter, we ran 100 simulation runs in the usage mode and tested the number of collisions that followed from the simulated incident scenario. The runs were repeated using different numbers of lanes (one to three) and different departure rates per lane (200 veh/h/lane to 1200 veh/h/lane). The Reaction-delay value that resulted in the lowest average number of collisions per run, when averaging over all runs, was then used for all following tests. The trained machine learning models associated with the found optimal Reaction-delay value were used for all subsequent tests of driver desensitization and safety benefit with the exception of the adverse weather tests.

For the tests in adverse weather conditions, a separate rain model was trained using the same procedure as for the other models. A separate model allows for better performance under rain conditions. Such conditions can be detected by vehicles indirectly by sensing the activation of their windshield wipers. Upon detection, the rain model is activated.
4.4.4.3 Measuring Driver Desensitization

The potential for driver desensitization depends heavily on the number of false warnings that the system will show. High number of false warnings will likely increase the possibility of driver desensitization. Therefore to test the potential for driver desensitization, we measured the false warning rate. This was done using a procedure similar to the one described in the previous subsection. We ran 100 simulation runs in the learning mode with different departure rates, participation rates, delivery probabilities and transmission ranges. Note that even though the tests were performed in the learning mode, no additional learning was actually taking place. The learning mode was used only to generate additional testing examples. This resulted in test example sets for each combination of parameter settings. For each of these sets, the five different relevance estimation methods (see section 4.4.4.1) were used to find whether each would show a warning for each test example. This was then used to find the false warning rate for each case. Note that the noEEBL method was not tested because the false warning rate is undefined when no warnings are given.

4.4.4.4 Measuring the Safety Benefit of EEBL

We measured the safety benefit of EEBL by simulating the incident scenario and measuring the number of collisions that followed. The machine learning methods utilized the previously trained models with the \( T_{\text{Warning}} \) and \( \text{Reaction-delay} \) parameters set according to the procedure previously described. We ran 1000 simulation runs in the usage mode using all the relevance estimation methods and repeated the procedure for the same combinations of parameters as was done for the driver desensitization tests. In addition, we also tested the case
when the drivers do not react to warnings. For every test, we calculated the average number of collisions per run in each lane.

4.5 Results

In this section, we present the results of our simulations. In section 4.5.1, we show results related to finding the optimal value of the $T_{\text{Warning}}$ parameter. Section 4.5.2 shows the results for finding the optimal value of the $\text{Reaction-delay}$ value. In sections 4.5.3 through 4.5.6, we discuss the results of the driver desensitization and the safety benefit tests under various departure rates, participation rates, weather conditions, and wireless communication parameters, respectively. The tests show the effects of these parameters in terms of the false warning rates and the average number of collisions per run. Lastly, in subsection 4.5.7, we show the effects of drivers not reacting to warnings on the average number of collisions.

4.5.1 $T_{\text{Warning}}$ Values

Figures 21 and 22 show the false and the missed warning rate curves as they varied with the value of the $T_{\text{Warning}}$ parameter for the two lane case when the $\text{Reaction-delay}$ was set to 9 seconds. The exact values of $T_{\text{Warning}}$ for every machine learning method and road configuration case that resulted in equal false and missed warning rates is shown in Table V (for a $\text{Reaction-delay}$ of 9 seconds). The values were found to be largely dependent on the distribution of the positive and the negative training examples, generally increasing with higher values of the $\text{Reaction-delay}$.
Figure 21. False and missed warning rates for different values of $T_{Warning}$ in the two lane case using the logistic regression model ($Reaction-delay$ set to 9 seconds).

Figure 22. False and missed warning rates for different values of $T_{Warning}$ in the two lane case using the naïve Bayes model ($Reaction-delay$ set to 9 seconds).
TABLE V. VALUE OF TWARNING PARAMETER FOR DIFFERENT ROAD CONFIGURATION CASES AND MACHINE LEARNING METHODS

<table>
<thead>
<tr>
<th>Machine Learning Method / Road Configuration Case</th>
<th>TWarning Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ML-NB / 1 lane</td>
<td>0.99852</td>
</tr>
<tr>
<td>ML-NB / 2 lanes</td>
<td>0.95453</td>
</tr>
<tr>
<td>ML-NB / 3 lanes</td>
<td>0.99674</td>
</tr>
<tr>
<td>ML-LR / 1 lane</td>
<td>0.51856</td>
</tr>
<tr>
<td>ML-LR / 2 lanes</td>
<td>0.47156</td>
</tr>
<tr>
<td>ML-LR / 3 lanes</td>
<td>0.37109</td>
</tr>
</tbody>
</table>

4.5.2 Reaction-Delay Value

The results of the Reaction-delay tests are shown in Figure 23. The results show a significant improvement in performance as the Reaction-delay increases up to 9 seconds. As the value increases past 9 seconds, the average number of collisions increases slightly. The optimal value was hence found to be 9 seconds for both machine learning methods.
Figure 23. Average number of collisions per run for different values of Reaction-delay for naïve Bayes and logistic regression methods.

4.5.3 Effects of the Departure Rate

In this section, we study the effects of the departure rate for two cases: a highway case and a local street case. Subsection 4.5.3.1, discusses the results of the highway case and subsection 4.5.3.2 presents the local street case.

4.5.3.1 Highway Case

In this section, we present how the false warning rate and the average number of collisions vary with respect to the departure rate for the five different relevance estimation methods vary. For these tests, the transmission rate was set to infinity and both the delivery probability and the participation rate were set to 100%. The propagation delay was 0 ms.
The false warning rates for the one, two, and three lane configurations are shown in Figures 24, 25, and 26, respectively. Note that the simpleEEBL-lane and simpleEEBL methods are the same for the one lane case, so only one was shown. The results show significant improvement in the false warning rate with the machine learning methods in comparison to the other methods. Averaged over all runs, the ML-NB and ML-LR methods both had an average false warning rate of approximately 19%. This is less than a third of the false warning rates for the simpleEEBL (67%) and the simpleEEBL-lane methods (62%). The analytic method performance matched that of the simpleEEBL-lane method, at 62%. The likely reason for the poor performance is that the assumptions on which the analytic method relies may not be valid in every scenario. For example, the vehicle in front may not always immediately emergency decelerate at the time the report is received. Also, the length of the vehicle in front could be different than the assumed length. The approximations of the minimal safety gap might hence be inaccurate, which will cause false warnings to be given. However, we have performed additional tests in scenarios that attempt to maximize the performance of the analytic method by using only a single vehicle type and by assuming this type and its length are common knowledge among all vehicles, and these tests showed no difference in relative performance among the tested methods.

Similarly, inaccuracies in the estimation of the density might also result in additional false warnings. While the machine learning methods also rely on the estimation of density, although results are not shown, we have tested adding noise to the density measurement, and even in those tests, the analytic method performance was still inferior. The machine learning methods were however more sensitive to misestimation of the density.
As was suggested in (Dingus et al., 1997) and (Edwards et al., 2011), rates over 60% will cause the drivers to distrust the system. As such, the safety benefit of the simpleEEBL, simpleEEBL-lane, and the analytic methods might degrade over time, as drivers will be less likely to react to the warnings when they are shown. With comparably low average false warning rates, the machine learning methods will likely avoid the degradation of the safety benefit.

Figure 24. False warning rates for all methods and different vehicle departure rates (one lane case).
Figure 25. False warning rates for all methods and different vehicle departure rates (two lane case).

Figure 26. False warning rates for all methods and different vehicle departure rates (three lane case).
Figures 27, 28, and 29 show the results of the departure rate tests on the average number of collisions for the one, two, and three lane cases, respectively (note the logarithmic scale). As the figures show, the use of the EEBL methods resulted in substantial decreases in the average number of collisions. When averaged over all runs, the number of collisions per run without EEBL (the noEEBL method) averaged 28.83 collisions per run. With the use of EEBL, using any of the relevance estimation methods, most collisions were avoided. The reason for the large difference between the noEEBL and the other methods is that the simulated scenario creates a chain of vehicular collisions which is avoided using EEBL.

Comparing the EEBL methods, both the analytic and the machine learning methods were able to match the performance of the simpleEEBL and simpleEEBL-lane methods. Overall the simpleEEBL method achieved the best performance, with an average number of collisions of 0.91 when averaged over all runs. This was expected, since the simpleEEBL method warns the driver in every case. Nevertheless, the ML-NB method had a similar performance, with an average of 0.94 collisions per run. The simpleEEBL-lane method had a higher average of 1.57 while the ML-LR method averaged 1.89. The analytic method performance was almost identical to that of the simpleEEBL-lane method.

The results showed that the proposed machine learning method was able to maintain the safety benefit of the EEBL application, while significantly reducing the chance of driver desensitization due to false warnings. Note that in some cases, the machine learning methods prevented more collisions than simpleEEBL because the simple method caused more false warnings which resulted in unnecessary emergency deceleration. Unnecessary deceleration can cause additional collisions that would not otherwise happen without the EEBL warning.
Looking at the differences across the departure rates, the results for the one lane case showed that the average number of collisions generally decreased with increased departure rates. This behavior can be attributed to the fact that at low densities, vehicles travel at much higher velocities. This makes it harder for the vehicles to stop and avoid a collision. However, a higher departure rate also implies that there will be more vehicles making lane changes, which might cause additional collisions in emergency situations. This effect is evident in the two and three lane cases, where the number of collisions increased as the departure rate per lane rose. In the one lane case, the average number of collisions also increased slightly above 600 veh/h/lane. The behavior is consistent across all the EEBL methods, so it may be caused by something inherent in EEBL. However, the increase was very slight, and therefore may be due to the statistical variations in the simulations. Nevertheless, the behavior is not specific to the proposed machine learning methods, so it doesn’t affect our result. It may however be part of future investigations of EEBL in general.
Figure 27. Average number of collisions per run for all methods and different vehicle departure rates (one lane case).

Figure 28. Average number of collisions per run for all methods and different vehicle departure rates (two lane case).
Figure 29. Average number of collisions per run for all methods and different vehicle departure rates (three lane case).

In addition to measuring the total number of collisions occurring during each run, we also measured the collisions on each of the lanes. The average number of collisions per run, for each lane, averaged over all departure rates is shown in Figures 30 and 31. The figures show, as expected, that the number of collisions on the incident lane was generally higher than on the adjacent lane. An exception to that were the simpleEEBL-lane and the analytic methods, since those methods did not show warnings on the non-incident lanes and therefore had a greater number of collisions occurring on those lanes. Collisions on the non-incident lanes can be attributed to the fact that vehicles in the incident lane will try to change lanes in order to avoid the stopped vehicle. This may then cause a collision with an oncoming vehicle driving in the adjacent lane. Furthermore, there were more collisions on the left lane than on
the right. This is due to the speed differences in each lane. Since normally the left lane has higher speeds, vehicles are more likely to change into this lane and also more likely to collide with another vehicle.

Figure 30. Average number of collisions per run in each lane for all methods (two lane case).
4.5.3.2 Local Street Case

To test the conditions that would occur on a local street as opposed to a highway, we reran the tests for the three lane case and set the road type in MITSIM to urban street and lowered the speed limit and the free-flow speeds to 35 mi/h. The results were similar to the highway case, with the false warning rates showing no significant deviations and the average number of collisions with only a slight decrease of about 0.67 collisions on average for the EEBL methods. The results thus indicate that the local street case did not significantly affect the performance of the methods.
4.5.4 Effects of the Participation Rate

During the initial deployment of the EEBL system, we can expect that the participation rate will be low. To test the effect this will have on the system, we ran tests using varying participation rates. In each test, the departure rate per lane was fixed to 600 veh/h/lane, the delivery probability was set to 100%, and the transmission range was infinite. The propagation delay was 0 ms. The participation rate was varied from 0% to 100% in 20% increments.

For the false warning rates, the results showed almost no changes in the false warning rate as the participation rate varied. The exact rates were approximately the same as seen in Figures 24-26 for the 600 veh/h/lane departure rate. While some small differences were present across the participation rate range, these were likely due to the randomness of the simulations. The reason is that the participation rate only reduces the number of testing examples that are used, but not the proportion of false warnings.

Figures 32 and 33 show the results of the tests for the average number of collisions in the one and three lane cases. Overall, the results showed that the machine learning methods were able to match or exceed the performance of the simple methods with varying participation rates. The analytic method also performed similarly to the other methods. As can be seen in the figures, the number of collisions was relatively similar across all the different EEBL relevance estimation methods (without considering noEEBL). Small differences between the methods did exist and became greater as the participation rate was increased. Also, the differences were greater for the multi-lane cases. However, in comparison to the number of collisions resulting from noEEBL, these differences were insignificant.
Figure 32. Average number of collisions per run for all methods and different vehicle participation rates (one lane case).

Figure 33. Average number of collisions per run for all methods and different vehicle participation rates (three lane case).
4.5.5 Effects of Adverse Weather

To test the effects of adverse weather on the methods, we reran the departure rate tests for the three lane case under simulated rain conditions. For these tests, the transmission rate was set to infinity and both the delivery probability and the participation rate were set to 100%. The propagation delay was 0 ms.

Figures 34 and 35 show the results of the tests for the false warning rates and the average number of collisions, respectively. In comparison to the results in normal conditions, the false warning rates showed no significant changes for the simpleEEBL, simpleEEBL-lane, and analytic methods. However, there was a 6% average decrease for the ML-NB and ML-LR methods. The improvement can be attributed to a separate rain model that was used for these tests. In terms of the safety effects, there were generally more collisions in rainy situations, with the EEBL methods more affected than noEEBL. However, for low departure rates, the number of collisions is slightly reduced for the EEBL methods. This is because vehicles travel at lower speeds in rain conditions and hence can stop faster. For higher departure rates, the effect of the lower speeds is reduced because vehicles follow each other closely.
Figure 34. False warning rates for all methods and different departure rates under rain conditions (three lane case).

Figure 35. Average number of collisions per run for all methods and different departure rates under rain conditions (three lane case).
4.5.6 Effects of Wireless Communication Parameters

Previous tests assumed that all vehicles on the road can immediately receive the reports after they have been generated. In real life, the wireless communication technologies such as DSRC are limited by several factors, such as the transmission range or the transmission delays. With a limited transmission range, only a subset of vehicles (those within the range) can receive the reports. This will hence affect the safety benefit of the EEBL in terms of the average number of vehicle collisions that will occur. Similarly, errors in transmission may mean that a sent report may not be received. Therefore the delivery probability will also limit the safety benefit of the application. In addition, there may also be propagation delays due to the time needed for a transmission to take place or the time needed to process an already received report. Such delays will imply that the driver will not receive the warnings immediately and therefore the EEBL safety benefit may also be impacted.

Limited communication can also have an impact on the false warning rates because the communication will determine which vehicles receive the reports. Therefore, the number of vehicles for which the report is relevant may increase or decrease depending on the communication and as a result the false warning rate will be affected.

Lastly, in our model we assumed vehicles will rebroadcast any newly received reports. Although this increases the chance that a report will be received, it also increases the communication overhead that would be associated with the EEBL application. Therefore in some cases, such as when bandwidth is shared among several applications, it may be desirable not to rebroadcast the reports.

In the next subsection, we present the results of tests when the transmission range is limited. In subsection 4.5.6.1, we show the results when varying the delivery probability. In
subsection 4.5.6.2, we present the results of tests of propagation delays and in subsection 4.5.6.4, we discuss what happens when vehicles do not rebroadcast the reports.

### 4.5.6.1 Effects of the Transmission Range

To test the impact of a limited transmission range, we have repeated the vehicle participation rate tests with a transmission range of 100m. While 1000m is the approximate range of DSRC, the 100m range was used because the effective reliable transmission range can be significantly reduced in high communication load areas (Schmidt et al., 2009). Effective reliable transmission range is defined as the area where packet loss is less than 10% per receiver.

Figure 36 shows the results of testing the false warning rate when the transmission range is limited to 100 meters, using different values of the participation rate for the three lane case. The tests compared the simpleEEBL and the ML-NB methods. Infinite transmission range is labeled as INF. In these tests, the delivery probability was set to 100%, and the propagation delay was 0 ms. The results show that for the simpleEEBL method, a limited transmission range decreases the number of false warnings. The reason is that reports are more relevant for vehicles in close proximity to where the report originated. This can also be seen to a much smaller degree for the ML-NB method. The one and two lane cases are not shown, but the results were similar to the three lane case.
Figure 36. False warning rates for all methods and different participation rates (three lane case).

The results of the tests for the average number of collisions is shown in Figure 37. For the one lane case, the results showed that both methods achieved almost identical performance. At low participation rates, the performance of the methods was significantly affected by a short transmission range because not all vehicles received the reports. The situation was similar for the one and two lane cases. In the one lane case, the simpleEEBL method was better than ML-NB with infinite transmission range. However, the difference in performance between the two methods decreased significantly as the participation rate increased.
4.5.6.2 Effects of the Delivery Probability

To test the effects of the delivery probability, we ran tests for the simpleEEBL and ML-NB methods and varied the delivery probability from 0% to 100%, in 10% increments. For these tests, the transmission range was set to 100m, the propagation delay was 0 ms, the participation rate was 100%, and the departure rate per lane was 600 veh/h/lane.

The results for the false warning rates are shown in Figure 38 for the three lane case. The tests have shown that the delivery probability has only a minimal effect on the false warning rates of both methods. In the two and three lane cases, the ML-NB method significantly outperformed the simpleEEBL across the delivery probability range. For the one lane case,
the results were closer. Nevertheless, on average the ML-NB method false warning rate was less than one third that of the simpleEEBL rate.

Figure 38. False warning rates for two methods with varying delivery probabilities (three lane case).

Figure 39 shows the results for the average number of collisions in the one lane case. Note that the results are shown using a logarithmic scale. As can be seen in the figure, both methods achieved almost identical performance across the different values of the delivery probability. For the one lane case, above 40%, the delivery probability had only a marginal effect on the number of collisions. The two and three lane cases were similar to the one lane case. However, in the multi lane cases, 10% was already sufficient to achieve the safety
benefits of EEBL. The reason that EEBL with relatively unreliable communication still provides a significant safety benefit is that newly received reports are always rebroadcast, thus increasing the chance that a report will be received. Additionally, after a vehicle receives a warning from the EEBL and starts emergency decelerating, it will generate a new report that will be broadcast to other vehicles. Therefore few reports need to be delivered in each time step to achieve a low number of collisions.

Figure 39. Average number of collisions per run for two methods with varying delivery probabilities (one lane case).
4.5.6.3 Effects of the Propagation Delay

To test how propagation delays can affect the performance of the methods, we ran tests for the ML-NB method and varied the delay. 10 sets of 100 runs were performed, each with a different delivery probability parameter value, ranging from 10% to 100%, in 10% increments. For all runs the transmission range was set to 100m, the participation rate was 100%, and the departure rate per lane was 600 veh/h/lane. The results were averaged over all runs.

In terms of the false warning rates, the results showed no significant changes in the rate as the propagation delay increased. The reason that propagation delays do not affect the false warning rate is that the delay does not affect the vehicle’s reaction after a report is received.

Figure 40 shows the results for the average number of collisions. The tests showed that small delays (up to 500ms) did not cause any significant degradation of performance, with only a slight increase at 500ms. According to ElBatt et al. such delays are unlikely to be present (ElBatt et al., 2006). We can therefore conclude that propagation delays are unlikely to degrade the safety benefit of EEBL.
4.5.6.4 Effects of Not Rebroadcasting

To test whether rebroadcasting of newly received reports has an effect on the performance of EE BL, we performed tests for the simpleEEBL and ML-NB methods with and without rebroadcasting. The tests were done for the 3 lane case, with a varying transmission range of 100m, 250m, 1000m, and infinity. The participation rate was 100%, the departure rate was 600 veh/h/lane, and the propagation delay was 0ms.

Figure 41 shows the results for the false warning rates. In general, the results showed that the rebroadcasting of the reports does not significantly affect the false warning rate. However, without rebroadcasting, the ML-NB method did achieve marginally better performance at low transmission ranges. This can be attributed to the fact that without rebroadcasting, the reports...
are sent to vehicles within a short distance of where the report originated. Intuitively, the reports will be more likely to be relevant for those vehicles.

Figure 41. False warning rates for two methods and varying transmission ranges for the three lane case, with and without report rebroadcasting.

Results for the average number of collisions are shown in Figure 42. The tests have shown that with rebroadcasting both methods had no significant deviations in the average number of collisions across the different transmission ranges. Without rebroadcasting, the ML-NB method showed some degradation in performance at lower transmission ranges. This is especially clear at the 100m range. However, the simpleEEBL method did not experience
any significant change from the no rebroadcasting case. The reason is that when simpleEEBL is used, every vehicle within the transmission range immediately sees a warning after a report is received. Since in our simulations this implies that the vehicle will immediately start to emergency decelerate, it will cause additional reports to be generated. This doesn’t occur for ML-NB, because not every report initiates a warning. Therefore when the transmission range is very small, rebroadcasting is more important for that method to maintain the same safety benefit. For larger transmission ranges, which would be available if DSRC is utilized, rebroadcasting does not make any significant difference for either method.

Figure 42. Average number of collisions per run for two methods and varying transmission ranges for the three lane case, with and without report rebroadcasting.
4.5.7 Effects of the Probability of Reacting to Warnings

So far, we have assumed that every driver will react when a warning is given. We have also tested what will occur when drivers may not react to the warnings. For these tests, we assumed that for every time step after a warning is received, the driver will react to a warning with a specified probability. The departure rate per lane was fixed to 600 veh/h/lane, the delivery probability and the participation rate was set to 100%, and the transmission range was infinite. The propagation delay was 0 ms. The results for the three lane case are shown in Figure 43. As expected, for all methods, the average number of collisions decreased with higher probabilities of reacting to a warning. However, the relative difference in performance between the methods was not affected (note the logarithmic scale).

![Figure 43. Average number of collisions per run for all methods and different probabilities of reacting to a warning (three lane case).](image-url)
4.6 Summary

In this chapter, we have compared two methods for estimating the relevance of emergency brake reports in an EEBL system. One method used an analytically derived formula based on the minimum safety gap. The other was our proposed method of using a machine learning approach. The machine learning method works in two stages: the learning stage and the usage stage. The learning stage is used to train a machine learning model which is then utilized in the usage stage for deciding whether to warn the driver.

The methods were evaluated in simulations using the MITSIM software. We compared the methods to two simple approaches: the simpleEEBL method, which shows a warning for every received report, and the simpleEEBL-lane method, which shows a warning to all vehicles in the same lane as the vehicle that generated the report. Tests showed that all of these methods drastically reduced the number of collisions on the road in comparison to the results without the EEBL system. However, the machine learning methods also significantly reduced the number of false warnings given to the drivers when compared to the other methods. The naïve Bayes machine learning method had the best overall performance, due to its lowest false warning rate and the average number of collisions comparable to the simpleEEBL method. In contrast, the analytic method had a higher false warning rate than even the simpleEEBL method.

Although in this research our focus was on the EEBL application, the machine learning approach for relevance estimation has the potential to be used in other transportation safety applications. Examples of these applications include highway merge warning or control loss warning systems. In the next chapter, we show how the machine learning method can be applied to these and other novel applications.
CHAPTER 5

PLATFORM IMPLEMENTATION FOR SAFETY APPLICATIONS

This chapter discusses an implementation of the proposed platform for novel VANET safety warning applications.

5.1 VANET Safety Warning Application Model

The VANET safety warning application model represents any application in which the following occurs. A single vehicle, called the *sending vehicle*, detects some event that might be potentially dangerous to it or other vehicles, such as emergency deceleration or a control loss. Once this occurs, it disseminates information to other vehicles to alert them of the possible need for an evasive action. Any vehicle that receives such information, called the *receiving vehicle*, will then estimate whether the information is relevant to its driver. If so, the vehicle will issue a warning so that the driver can perform any necessary actions to mitigate the potentially dangerous situation. The three example applications discussed in section 1.5 (EEBL, HMW, and CLW) are concrete examples of applications to which this model applies. In the next subsection, we discuss our assumptions about vehicle capabilities in our model. We will then define the concept of an emergency action in subsection 5.1.2. Lastly, in subsection 5.1.3, we define the class of VANET safety warning applications.
5.1.1 **Vehicle Capabilities**

Our model assumes a set of vehicles is present on the road, each of which is either participating or non-participating. A participating vehicle is equipped with the following:

- Computing device with storage
- Communication device (e.g. DSRC)
- Warning indicator (e.g. visual or auditory signal)
- Digital map with lane information
- Set of sensors: accelerometer, GPS, and gyroscope

Conversely, a non-participating vehicle lacks all of the equipment carried by the participating vehicles. For participating vehicles, the communication device is used to transmit data to other vehicles. The warning indicator is used to warn drivers about the need to initiate some evasive action (e.g. braking). The digital map contains information about the road on which the vehicle is driving, including the number of lanes and their exact positions.

We assume that the set of sensors is the same among all participating vehicles. Every $T$ milliseconds, called the *reporting period*, the sensors are used to generate a set of values describing the state of the vehicle. We will call this set the *vehicle state vector*. A common value for $T$ for many applications is 100 milliseconds. Each vehicle state vector includes eight elements, grouped into two sets: vehicle statics and vehicle dynamics. *Vehicle statics* is information that is known to the vehicle at all times, such as the vehicle type and its tire type. *Vehicle dynamics* is information generated from the sensors and includes the position, speed, acceleration, heading angle, current lane id, and the current road segment id of the vehicle.
5.1.2 **Emergency Action**

In emergency situations, vehicles may perform certain evasive actions to try to avoid a collision. Typically, the immediate driver reaction will be to press hard on the brakes, which will cause the vehicle to decelerate with a maximum deceleration rate, which we call $T_{severe\_brake}$. Observe that $T_{severe\_brake}$ is specific to a vehicle type, load, and weather conditions. We will state that a vehicle *emergency decelerates* when its deceleration equals $T_{severe\_brake}$. Alternatively, the driver may choose to avoid a collision by changing the lane. We will call a lane change with the intent of avoiding a collision an *emergency lane change*. An emergency lane change can be detected by measuring the steering angle, angle velocity, and force (Kuge et al., 2000). An *emergency action* is hence defined to be either an emergency lane change or an emergency deceleration.

5.1.3 **The VANET Safety Warning Application**

A VANET safety warning application is a program running continuously on the computer of each participating vehicle. It is divided into two processes: report generation and report reception. The *report generation process* is responsible for creating and disseminating new reports. It starts by retrieving the state of the vehicle sensors and uses it, along with the digital map, to determine the vehicle dynamics. It then combines this information with known vehicle statics and forms a vehicle state vector. Each application is characterized by a report trigger condition that must be satisfied in order to generate a vehicle safety report. For example, a report trigger for EEBL would be emergency deceleration. The state vector and the digital map are used to check whether the report trigger condition is satisfied. If so, a new vehicle safety report is created and then disseminated using the VANET. Otherwise, there is
no new report and in both cases the process repeats itself after waiting $T$ ms for the next reporting period. The process is shown in Figure 44.

The *report reception process* is responsible for checking for newly incoming reports, determining its relevance, and warning the driver. When a vehicle receives a report, it uses a relevance estimation module to classify a report as either relevant or irrelevant. If the report is relevant, a warning indicator will then be activated. Otherwise, the report will be ignored. The feedback from this information can then be used to adapt the module. Figure 45 illustrates this process.

![Figure 44. Report Generation Process.](image)
5.2 Observe-Driver-And-Learn Method

In our proposed platform, we generate the relevance estimator module using the Observe-Driver-and-Learn (ODaLe) method\(^7\). ODaLe is a method for learning the relevance of information based on the machine learning approach previously discussed for travel time and parking information dissemination systems, and the EEBL application. The idea is to record the driver’s reaction after a report is generated by the application and use this information as an input to a machine learning algorithm. Since events that generate reports (e.g. emergency deceleration) are rare, simulations can be used to develop an initial relevance estimation.

\(^7\) This is the same as the Observe and Learn method described in previous chapters, but we use this term in this chapter to make it clear that when used in the context of safety warning applications, the observations are of normal driver behavior.
module that can then be refined through a process of adaptive learning (see section 5.2.3). The objective is to learn a relevance function, which maps the reports to a value between 0 and 1, with 0 being completely irrelevant and 1 being completely relevant.

The method works by observing how a driver normally reacts after a safety report is received without giving any warnings from the VANET application. For each report, we check whether the report was relevant by monitoring the vehicle’s behavior after the report was received for a specified fixed amount of time (the Reaction-delay). A relevant report is one in which received the driver of the receiving vehicle performs an emergency action\(^8\) within Reaction-delay seconds after the report was received.

For every received report we also calculate values for a set of attributes. Examples of attributes include the distance between the vehicles and the vehicle density. Generally, there is a common set of attributes that are useful in VANET safety applications. We have identified a large set of these attributes and make them available in our platform. However, since the relevance of a report in an application may not depend on all of the attributes, it is necessary to choose a relevant subset of attributes. In other words, for each application, learning will be done based on a distinct set of attributes. Attribute selection can be done manually by selecting the most intuitive attributes for the application. However, when there are many correlated attributes, manual selection may not yield optimal results. Therefore, our platform provides a method of automatic attributes selection. The list of attributes and the automatic attributes selection process is described in section 5.2.1.

---

\(^8\) Due to simulator limitations, for our platform implementation the only action we observe is emergency deceleration.
Once the attributes values are calculated and the relevance of the report is determined, the system then generates a training example. The training examples that are generated are added to a training example set stored by each vehicle. Once a sufficient number of training examples is generated, the set of all stored examples is input to a machine learning algorithm, which uses the training examples to learn the relevance function. The machine learning algorithms that we have used for this purpose are discussed in subsection 5.2.2. After the machine learning algorithm finds the relevance function, the relevance estimator module is output.

When a report arrives at a vehicle, the relevance estimator module calculates the attributes values for the report and uses the learned relevance function to calculate the probability of relevance. The decision to turn on the warning indicator then depends on a fixed threshold called $T_{\text{warning}}$. When the relevance value of the report exceeds $T_{\text{warning}}$, the warning indicator will then turn on. Otherwise, if the value is less than or equal to $T_{\text{warning}}$, the report will be ignored. We set the $T_{\text{Warning}}$ threshold based on the set of generated training examples (see section 4.5.1).

### 5.2.1 Attributes Selection

Although different VANET safety warning applications may depend on various attributes, in general, there are four categories of attributes which can be potentially useful in any application:

- Road attributes
- Vehicle dynamics attributes
- Vehicle statics attributes
• Weather related attributes

Table VI provides a listing of all the attributes available in our platform. Road attributes describe the road segments on which the receiving and sending vehicles are currently driving. We identified five road attributes that may be useful in VANET safety warning applications: speed limit, road curvature, number of lanes, vehicle density, and road grade. These attributes are for both the current road segment of the sending vehicle as well as the receiving vehicle. Therefore there are a total of ten road attributes that can be used within our framework.

Vehicle dynamics attributes compare the movement information of the receiving vehicle to the sending vehicle. These include: distance between the vehicles, difference in time-to-crossing, difference in speeds, difference in accelerations, difference in headings, lane offset, and the on-same-road-segment indicator. The distance is calculated in terms of Euclidean distance and is based on the latitude and longitude coordinates reported by the positioning device. A crossing point is the point at which the paths of the receiving and sending vehicles would meet, assuming they maintain their speed and heading. The difference in time-to-crossing is defined as the difference between the time it would take the sending vehicle to reach the crossing point at its current (i.e. reported) speed and the time it would take the receiving vehicle to reach the crossing point at its current speed. Note that the smaller the time-to-crossing, the larger the probability that the two vehicles will collide at the crossing point. The lane offset is calculated by subtracting the current lane number of the receiving vehicle from the lane number of the sending vehicle. Assuming both vehicles are on the same road segment, a lane offset of zero will indicate both vehicles are on the same lane. The on-same-road-segment indicator is a Boolean valued attribute that is true when both the receiving and sending vehicles are driving on the same road segment and false otherwise.
Vehicle statics attributes compare the static vehicle information between the receiving and sending vehicles. The attributes are: the sending and receiving vehicle types and the sending and receiving vehicle tire types. Different vehicle types may be specified by the application designer in MITSIM, such as passenger cars or trucks. Three different tire types can be specified for each vehicle type.

Weather related attributes are the current visibility and road pavement friction. These attributes were identified based on a study of factors that are impacted by different weather events such as fog, rain, or snow (Federal Highway Administration, 2004). While the values of these attributes may not be directly known to the vehicles, the information may be derived from a set of road sensors (American Meteorological Society, 2003) or using vehicles as sensing nodes (El-Tawab et al., 2009).

In order to automatically choose among the many available attributes, our platform uses an attribute selection method using a previously proposed algorithm (Hall, 1998). This algorithm evaluates all subsets of attributes (assuming exhaustive search) using a correlation based heuristic. The method chooses a subset of attributes that are highly predictive, but have low correlation among them. This is important, because use of correlated attributes can degrade the performance of the Naïve Bayes method that we have utilized for learning due to the independence of attributes assumption (see section 3.2.3) (Langley and Sage, 1994).
### TABLE VI. LIST OF ATTRIBUTES BY TYPE.

<table>
<thead>
<tr>
<th>Attribute Type</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Road Attributes</strong></td>
<td>For the receiving and sending vehicle’s road segments:</td>
</tr>
<tr>
<td></td>
<td>- Speed limit</td>
</tr>
<tr>
<td></td>
<td>- Road curvature</td>
</tr>
<tr>
<td></td>
<td>- Number of lanes</td>
</tr>
<tr>
<td></td>
<td>- Vehicle density</td>
</tr>
<tr>
<td></td>
<td>- Road grade</td>
</tr>
<tr>
<td><strong>Vehicle Dynamic Attributes</strong></td>
<td>Distance between the vehicles</td>
</tr>
<tr>
<td></td>
<td>- Difference in time-to-crossing</td>
</tr>
<tr>
<td></td>
<td>- Difference in speeds</td>
</tr>
<tr>
<td></td>
<td>- Difference in accelerations</td>
</tr>
<tr>
<td></td>
<td>- Difference in headings</td>
</tr>
<tr>
<td></td>
<td>- Lane offset</td>
</tr>
<tr>
<td></td>
<td>- On-same-road-segment (yes/no)</td>
</tr>
<tr>
<td><strong>Vehicle Static Attributes</strong></td>
<td>Sending vehicle type</td>
</tr>
<tr>
<td></td>
<td>- Receiving vehicle type</td>
</tr>
<tr>
<td></td>
<td>- Sending vehicle tire type</td>
</tr>
<tr>
<td></td>
<td>- Receiving vehicle tire type</td>
</tr>
<tr>
<td><strong>Weather Attributes</strong></td>
<td>Visibility</td>
</tr>
<tr>
<td></td>
<td>- Road pavement friction</td>
</tr>
</tbody>
</table>

### 5.2.2 Machine Learning Methods

In our implementation, we have used three types of machine learning methods: naïve Bayes, probability estimation trees, and logistic regression. Note however that the relevance learning is not limited to the methods presented in this paper and other machine learning techniques may also be applicable.

The naïve Bayes method is a simple method for learning based on probabilities that was discussed previously in section 3.2.3. The logistic regression method assumes that the probability of an emergency action, which we use as the relevance function, fits a logistic function. This method was discussed in section 4.3.2.2. For the probability estimation tree
method, we built an unpruned C4.5 decision tree (Quinlan, 1993) and estimated the probabilities by counting how many training examples get classified at the leaf nodes. Laplace smoothing was also used, as this was shown to improve the probability estimates (Zadrozny and Elkan, 2001).

5.2.3 Adaptive Learning

After initially developing a relevance estimation module and evaluating it in simulations, the module can be further adapted to individual vehicles or drivers through the process of adaptive learning. This process function based on the same ODaLe principle that is used for learning in simulations. In fact, the simulations are not necessary for deploying the application and using ODaLe for developing a relevance estimation module. For this purpose, an online version of the machine learning algorithm can be utilized. The Online Naïve Bayes algorithm is especially suitable due to low computational and storage requirements (Duda et al., 2000). However, there are several issues with learning after the application is deployed that make it advantageous to initially develop a module based on simulations. Firstly, learning events will be rare, therefore the learning process will be slow. Secondly, because the warnings will affect drivers’ behavior, bias can be introduced in the learning. For example, after a driver receives a warning and emergency decelerates, it is not known whether the driver action was caused by the situation on the road or the warning itself.

There are several ways in which to avoid biasing the learning process. One method is to rely only on examples that contradict the prediction. This means that either, a warning was shown and the driver did not perform an emergency action, or no warning was shown, but the driver did perform an emergency action. In both cases, it is reasonably certain that the driver
behavior was not affected by the VANET safety warning application. The negative aspect of this method is that half of the training example types are not used for learning. To utilize all types of training examples, advanced techniques such as expectation-maximization may be used. However, it is not known whether this would avoid a bias. Alternatively, one could deliberately force warnings to be shown in cases when the module indicates otherwise, and vice versa. Although this may avoid the bias, there are some potential legal issues in this method. Therefore the concept of adaptive learning requires significant investigation and is part of our future work.

5.3 The Platform

The relevance estimator development and evaluation platform is based on the ODaLe method and provides a set of tools for generating relevance estimation modules for arbitrary VANET safety warning applications. Our platform implementation utilized the MITSIM microscopic traffic simulator (Yang and Koutsopoulos, 1996), extended to enable the simulation of VANET safety applications (see section 4.4.2). It also used the Weka Learning Toolkit (Hall et al., 2009) for performing the machine learning that is necessary when using the ODaLe method. The developed tools generate a relevance estimator module for the specified VANET safety application and allow for evaluating its safety benefit.
Figure 46 shows the process of using the platform to develop a relevance estimation module for VANET safety warning applications. The user first defines the inputs consisting of a report trigger along with a set of scenarios. The report trigger is specified by a set of conditional statements, using predefined variables (e.g. current vehicle deceleration, speed, etc.) and constants (e.g. vehicle’s maximum deceleration). Each scenario consists of the road
network definition, an incident setting, the choice for a definition of a relevant report, and the VANET safety warning application parameters settings. The road network definition includes the geometry of roads, the speed limits, the departure rates, and other parameters that are normally used within MITSIM. The incident setting can be used to generate special events such as a stopped vehicle or a vehicle that lost control. The choice for the definition of a relevant report is used by the ODaLe method for learning the relevance estimation modules. The choice can be set to emergency deceleration, emergency lane change, or both. The VANET application settings include the participation rate and the vehicle reaction to a report, which determines the behavior of the vehicle after it receives a warning. The behavior can be set to be an emergency deceleration, a slow down, or a forced lane change. If emergency deceleration is specified, the vehicle will immediately start decelerating at its maximum rate until stopped. If a slow down is chosen, the vehicle’s desired speed is reduced to 25% of the calculated value. A forced lane change means that the vehicle will make a mandatory lane change.

Once the inputs are defined, multiple runs of the simulation are executed and the training examples are generated. The runs are executed in the learning mode, which means that vehicles will not react to the reports and will therefore follow their normal behavior, as specified by the MITSIM simulator. After the training examples are created, the system will then use half of these (the training set) for generating a module and the other half (the testing set) to evaluate it in terms of the false warning rate. The generation of the modules is done by first running the automatic attribute selection algorithm, then using one of the three available machine learning algorithms to learn a model. After this is done, the system will find the optimal value for the $T_{\text{Warning}}$ parameter and output the false warning rate. The system will
also output the false warning rate associated with the *AlwaysWarn* baseline method, which assumes a warning will be given for every testing example. After the relevance estimator module has been generated, it can then be deployed to real vehicles that use the application. The module can then be subsequently refined in order to adapt it to individual vehicles and drivers (see discussion in section 5.2.3).

Once the relevance estimator module is generated, its effects on safety can be evaluated in simulation in terms of the average number of collisions per simulation run. This is done by re-executing the simulations using the previously learned module in MITSIM. The simulations are done in usage mode, which means that whenever a report is received, a relevance estimator module is used to determine its relevance. Then if a report is determined to be relevant, the vehicle will react as specified in the inputs. The simulation can be set to use the learned module or one of two baselines, *AlwaysWarn* or *NeverWarn*. The *NeverWarn* baseline means that all reports are ignored by the vehicles.

### 5.4 Evaluation Methodology

To evaluate the effectiveness of the platform, we have developed the relevance estimation modules for the three examples applications and tested them in terms of the number of collisions and the false warning rate under different conditions. Three methods were compared for each application: ODaLe, *AlwaysWarn*, and *NeverWarn*. The ODaLe method used the machine learning method that provided the best tradeoff between the two tested metrics. For each application, we set up different scenarios and specified the report trigger. This is described in subsections 5.4.1 through 5.4.3. We then used our developed tools for running the simulations and generating the relevance estimation modules. The
modules were then evaluated by running the simulations with multiple runs over the same scenarios for different settings of weather parameters: visibility (50-10000m) and road-tire friction coefficient (0.4-1.0). The results are discussed in section 5.5.

5.4.1 Evaluation Environment for EEBL

For EEBL, the scenario consisted of a single straight road, divided into two segments. The beginning segment was two miles in length and the ending segment was one mile. For each run, the number of lanes, the departure rate, and the speed limit were varied. The tested values for those parameters are shown in Table VII. A stopped vehicle incident scenario was specified to occur at the end of the segment. The report trigger was defined as acceleration equal to the maximum deceleration. Vehicles’ reaction to a report was emergency deceleration.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of lanes on both segments</td>
<td>1, 2, 3</td>
</tr>
<tr>
<td>Departure rate (veh/h/lane)</td>
<td>400, 800, 1200</td>
</tr>
<tr>
<td>Speed limit on both segments (mph)</td>
<td>55, 65</td>
</tr>
</tbody>
</table>

5.4.2 Evaluation Environment for CLW

For the CLW application, the scenario was the same as for the EEBL application, except for the incident, which was specified as a control loss instead of the immediate stop. The
report trigger was set to control loss equal true. Vehicle’s reaction to a report was a slow down.

5.4.3 Evaluation Environment for HMW

In HMW, the scenario consisted of two merging roads. The first road (highway) was a straight and divided into three segments, 200 feet in length. The first and the third segments were single lane segments, while the second segment had two lanes. The second road (on-ramp road) consisted of a one lane straight segment connected to a curved, one lane on-ramp segment that merged into the left lane of the second segment of the first road. For each run, we varied speed limits on the highway and the on-ramp and the departure rates. The tested values for those parameters are shown in Table VIII. The report was triggered whenever the vehicle entered the second highway segment. Vehicle’s reaction to a report was a slow down.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Departure rate for the on-ramp road (veh/h/lane)</td>
<td>400, 800</td>
</tr>
<tr>
<td>Departure rate for the highway (veh/h/lane)</td>
<td>1400, 1800</td>
</tr>
<tr>
<td>Speed limit for the on-ramp road (mph)</td>
<td>25, 35</td>
</tr>
<tr>
<td>Speed limit for the highway (mph)</td>
<td>55, 65</td>
</tr>
</tbody>
</table>
5.5 Results

5.5.1 Automatic Attribute Selection

The result of using the automatic attribute selection for each application is shown in Table IX below. In addition to the attributes listed, the visibility and road-tire friction coefficient attributes were manually chosen for all applications.

<table>
<thead>
<tr>
<th>Application</th>
<th>Selected Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>EEBL</td>
<td>Sender’s road vehicle density, distance, difference-in-accelerations, difference-in-speeds</td>
</tr>
<tr>
<td>HMW</td>
<td>difference-in-accelerations, difference-in-speeds, difference in time-to-collision</td>
</tr>
<tr>
<td>CLW</td>
<td>Sender’s road vehicle density, distance, difference-in-accelerations, difference-in-speeds, and receiver’s vehicle type</td>
</tr>
</tbody>
</table>

For both EEBL and CLW, the algorithm picked the same attributes: three attributes that are related to the vehicle dynamics and the sender’s road vehicle density. The receiver’s vehicle type was also chosen for CLW. The selection follows the intuition, since the lower the distance between the two vehicles, the higher the chance that the receiving vehicle may be affected by the emergency braking or a control loss of another vehicle. The difference in accelerations is also important, because a report may be more relevant if the receiving vehicle
is currently accelerating, while the sending vehicle is emergency braking or has lost control, and vice versa. Similarly, if the receiving vehicle is travelling faster than the sending vehicle, the report may also be more relevant.

For the HMW application, three attributes were chosen. Two attributes, difference-in-accelerations and the difference-in-speeds, were the same as for EEBL and CLW. These attributes are important for the same reasons. However, in HMW the difference in time-to-collision attribute was also chosen. Intuitively this attribute makes sense for an application like HMW, because it indicates whether two vehicles are on a collision course. The distance was not chosen, most likely because it is related to the difference in time-to-collision and hence redundant.

5.5.2 Comparison for Different Applications

Figures 47 and 48 show the results of the average number of collisions per run and the false warning rates for each application and relevance estimation method. In terms of the collisions, the performance of the ODaLe method nearly matched that of the AlwaysWarn method. At the same time, the use of the ODaLe method allowed for a drastic reduction in the false warning rates, thereby decreasing the chance of driver desensitization. Therefore, while AlwaysWarn may have resulted in slightly less collisions during the simulations, the performance would not be sustainable when deployed in real vehicles. This is because the high number of false warnings would cause the drivers to distrust the system and possibly ignore the warnings. A very high false warning rate may also lead the drivers to deactivate the system.
Both methods resulted in significantly less collisions than using the NeverWarn method. This suggests that the applications have a good potential for improving vehicular safety. However, the degree of improvement differs across applications. While the EEBL application resulted in a decrease in the average number of collisions on an order of magnitude (from approximately 22 collisions per run for NeverWarn to less than one collision per run for ODaLe), both the CLW and the HMW applications resulted in lesser improvements. This can be explained by the types of scenarios that were specified in the simulation platform. Specifically, the report reaction for EEBL was set to emergency deceleration, while the other applications were set to a slow down. Intuitively, assuming that vehicles will emergency decelerate instead of simply slowing down will allow for more collisions to be avoided. This combined with the different nature of incidents involved for each application is the primary reason for the discrepancy in the safety benefits of the application. Since the specific reaction of the driver is unknown, the application developer should utilize the platform for testing the different reactions and how they impact the performance. The results of such testing are shown in the next subsection.
Figure 47. Comparison of the average number of collisions for each application (logarithmic scale).

Figure 48. Comparison of the false warning rates for each application.
5.5.3 Effects of the Driver Reaction to a Report

When a warning is shown, the driver can respond in a number of different ways that can significantly impact the safety benefits of the application. Our proposed platform provides the ability to test the various types of reactions: emergency deceleration, slow down, forced lane change, or no action. Figure 49 shows the results for the average number of collisions per run for the CLW application when different report reactions are chosen. The ODaLe method was used for relevance estimation. As can be seen, the results indicate that if we can assume that drivers would emergency decelerate instead of simply slowing down after seeing a warning, the safety benefit of the CLW application would improve. However, if the drivers simply change lanes after a warning, the safety benefits would be negligible. This is intuitive, because in a control loss event, the incident vehicle may end up in any lane. Therefore the potential for a collision will exist in every lane and cannot be avoided by changing the lane.
5.5.4 Machine Learning Method Performance

Our platform allows the use of three different machine learning methods for generating a relevance estimation module: naïve Bayes, decision trees, and logistic regression (see section 5.2.2). Figure 50 shows the performance of the ODaLe method for EEBL in terms of both the average number of collisions and the false warning rates when different machine learning techniques are utilized. It shows that while the naïve Bayes and logistic regression methods allowed for fewer collisions to occur, they had significantly high false warning rates over 60%. Although the difference in collisions is significant, it is very small compared to the number of collisions without EEBL (approximately 22 on average). Therefore, the probability estimation tree method can provide a better tradeoff between the two measures of

Figure 49. The impact of the report reaction setting on the average number of collisions per run for the CLW application when using the ODaLe method.
performance. This was also true for the HMW application. However, this is not always the case, as the naïve Bayes method provided better performance for the CLW application.

Figure 50. The impact of the machine learning technique on the average number of collisions per run and the false warning rates for the EEBL application when using the ODaLe method.

5.5.5 Weather Effects

Weather factors such as the visibility or the road-tire friction coefficient can have a significant impact on the safety benefits of an application. These factors can affect the vehicles in two ways. Intuitively, lower visibility and friction will result in more potential for a collision because the vehicle will have less time to react and when it does react, it will not
be able to decelerate quickly. However, drivers usually take this into consideration and adjust their speeds accordingly, resulting in a lower potential for collisions. It is therefore the strength of these two effects that will be responsible for the average number of collisions.

Figures 51 and 52 show the effects of the visibility and the road-tire friction coefficient on average the number of collisions in the HMW application. Without the HMW application (NeverWarn method), as the visibility decreases, the number of collisions decreases until 100m. This can be attributed to the fact that drivers will slow down when the visibility is lower. It then significantly increases for 50m, as the drivers may be unable to see surrounding vehicles. However, when HMW is used to warn the drivers, most collisions can be avoided. The road-tire friction coefficient did not have as large of an impact as visibility. For all methods, as the friction increased, the collisions also increased up to a point, then decreased. This suggests that drivers slowed down significantly when friction was very low to compensate for the lesser braking abilities.
Figure 51. Effects of visibility on the average number of collisions in the HMW application for each relevance estimation method.

Figure 52. Effects of the road-tire friction coefficient on the average number of collisions in the HMW application for each relevance estimation method.
5.6 **Summary**

This chapter presented an implementation of the proposed platform for developing and evaluating relevance estimators in novel VANET safety applications. The platform provides a set of tools based on the Observe-Driver-and-Learn method for learning and testing relevance estimation modules using machine learning techniques. The evaluation proved the feasibility of using the platform for three different applications. The results showed that the learned modules achieved significant decreases in the number of false warnings, while maintaining the safety benefits of the applications.
CHAPTER 6

CONCLUSIONS

This dissertation proposed a platform for developing relevance estimation modules for VANET applications. The platform uses simulations to record observations of vehicles that are used to generate training examples. The examples include a set of attributes that are either manually or automatically selected, and a label indicating the observed relevance value. Once a sufficient set of training examples are generated, machine learning techniques are used to generate a relevance estimation module that maps incoming reports as either relevant or irrelevant. The resulting module can then be used in applications deployed in real vehicles.

The relevance estimation module can be utilized for use in ranking the information, which provides a way for dealing with bandwidth issues in communication networks and aids vehicles in decision making. The method was evaluated using two non-safety VANET applications. In the parking availability dissemination application, the method was able to match the performance of an optimal method in terms of the parking discovery time. For the travel time dissemination application, the results showed that vehicles utilizing the machine learning method had lower average travel times than when heuristic methods were used. Several machine learning methods were also evaluated for this application. Results showed that decision trees were best in terms of accuracy, but logistic regression and Naïve Bayes methods offered the best interpretability. The Naïve Bayes method also allowed for easy updates, making it most useful for learning relevance in online environments. For both non-
safety applications, the main advantage of using the proposed technique is that several, known to be useful attributes can be easily combined in a way that improves the dissemination performance, without extensive knowledge or analysis of the application.

The method was also applied for safety applications. First, we focused on the Electronic Emergency Brake Light, which is an application known to possess a high safety benefit. The main issue with this type of application is controlling the false warning rate, since a high number of false warnings can desensitize the drivers. The results showed that the use of the machine learning approach enabled a significant decrease in the false warning rate, while maintaining the safety benefits of the application. The tests were performed across a range of transportation and communication parameters.

The proposed idea of the platform was then implemented for developing novel VANET safety warning applications. The core of the platform was the machine learning approach method that was tested on the EEBL, travel time dissemination, and parking information dissemination applications. The implemented platform allowed the users to specify the novel application in terms of the report trigger condition, choose a set of scenarios, and run simulations observing the vehicle behavior. The system would then generate a set of training examples, automatically choose a set of relevant attributes, and learn a relevance function using machine learning techniques. The relevance function, along with the set TWarning parameter value would then be combined to form the relevance estimation module. The evaluation of the platform showed how this process can be used in three previously proposed applications. Similarly to the previous EEBL tests, the safety benefits of these applications were maintained, while the number of false warning was drastically reduced.


Parsons, S.: Current Approaches to Handling Imperfect Information in Data and Knowledge Bases. IEEE Trans. on Knowledge and Data Engineering, vol. 8, no. 3 (June, 1996), pages 353-372.


VITA

Piotr Szczurek

EDUCATION

2012, Ph.D. in Computer Science, University of Illinois at Chicago
Advisors: Prof. Ouri Wolfson, Prof. Jie (Jane) Lin
Computational Transportation Science IGERT Program

2005, B.S. in Computer Science, University of Illinois at Chicago
Graduated Summa Cum Laude, G.P.A. 3.95/4.0

AWARDS AND FELLOWSHIPS

2006-Present, Integrative Graduate Education and Research Traineeship (IGERT)
National Science Foundation multidisciplinary doctoral training program

Fall 2005, Bell Honors Award
Given in recognition of attaining the highest grade point average in the graduating class.

RESEARCH EXPERIENCE

2006-Present, IGERT Research Fellow, University of Illinois at Chicago
• Performed interdisciplinary research as part of the Computational Transportation Science (CTS) program.
• Studied problems related to transportation safety systems and dissemination of traffic information
• Worked on applying machine learning techniques for use in transportation applications
• Developed technique for estimating relevance of information using machine learning
• Presented work at conferences and IGERT seminars
RELEVANT WORK EXPERIENCE

Spring 2011, Teaching Assistant, University of Illinois at Chicago
• Prepared and led laboratory sessions
• Graded projects and lab assignments
• Prepared and administered quizzes

2005-2006, Engineer, Tellabs, Naperville, IL
• Worked on automation tool development for the validation team
• Developed tools helped in increasing the efficiency of the validation effort
• Performed testing of the Tellabs 5500 Digital Cross Connect

2005, Intern, Tellabs, Naperville, IL
• Worked on testing automation project that would enable porting of existing test scripts on different systems through a remapping of variable addresses
• Developed automation programs in Tcl/Tk/Expect and Perl languages. The standard TL1 language was used for communication with the cross connect system.

PUBLICATIONS


TALKS AND POSTERS

1. “Emergency Electronic Brake Light: When are Warnings Relevant to Drivers?”, talk presented at the IGERT Seminar Series, University of Illinois at Chicago, March, 2011


