PAPER

# Explanatory Rule Generation for Advanced Driver Assistant Systems

**Juha HOVI**[†,††a)], *Nonmember and* **Ryutaro ICHISE**[††,†], *Senior Member*

**SUMMARY** Autonomous vehicles and advanced driver assistant systems (ADAS) are receiving notable attention as research fields in both academia and private industry. Some decision-making systems use sets of logical rules to map knowledge of the ego-vehicle and its environment into actions the ego-vehicle should take. However, such rulesets can be difficult to create — for example by manually writing them — due to the complexity of traffic as an operating environment. Furthermore, the building blocks of the rules must be defined. One common solution to this is using an ontology specifically aimed at describing traffic concepts and their hierarchy. These ontologies must have a certain expressive power to enable construction of useful rules. We propose a process of generating sets of explanatory rules for ADAS applications from data using ontology as a base vocabulary and present a ruleset generated as a result of our experiments that is correct for the scope of the experiment.

***key words:*** *advanced driver assistant system, ADAS, ontology, rule-based reasoning, decision-making, knowledge representation, machine learning*

## 1. Introduction

Autonomous vehicles of different automation levels and advanced driver assistant systems (ADAS) are quickly advancing fields of research. Particularly decision-making systems pose great challenges while being in a crucial role for successful operation in real-world traffic situations. These challenges range from simpler lane-following and adaptive cruise control systems that are already successfully used by consumers to considerably more challenging problems such as an automated vehicle control system interacting with human drivers and interpreting subtle behaviors.

Traditional approaches to automated systems include, for example, rule-based systems and state machines. These systems often describe combinations of observations in human-understandable form and map these observations into actions (rule-based) or transitions between system states (state-machine). More recently, advances in computing power over the last decades have enabled the practical use of deep artificial neural networks (ANN), which have dominated many fields of artificial intelligence. While ANN applications often offer considerable performance and flexibility, they come with a notable downside when compared to more traditional methods. ANN is considered to operate as a black box and understanding the inner workings of the system is an active field of research due to its difficulty. ANN systems have been applied to both autonomous driving and ADAS. Estimating the correct output for unseen inputs is often considered a major strength of neural networks. However, this strength can be a weakness. When given an input unseen in training, the network will output a previously unknown result. This is in stark contrast to rule-based systems where, if none of the rules are satisfied by the input, the output is a known default output. Due to this, verification of the behavior of ANN systems in all situations is challenging. Traffic as a real-time operating environment is extremely safety critical and, as such, field use of unpredictable systems is undesirable. Conversely, a more human-understandable approach of rule-based reasoning on explanatory rules offers easier understanding and verification of system behavior. Thus, rule-based methods offer desirable benefits and could possibly be used in conjunction with other approaches on the way towards smooth and safe operation.

As is often the case, benefits of explanatory rule-based systems come with their own set of problems to solve. Rule-based decision-making systems such as the one introduced by Zhao et. al. in 2017 [1] require a set of rules on which they infer the actions the ego-vehicle should take. Writing these rules by hand, as in the work by Zhao et. al. [1], poses challenges in terms of scalability. Traffic is a varied and complex operating environment and sets of rules covering enough situation to be useful are likely to include a large number of rules. Additionally, traffic rules differ between countries and even driver behaviors vary from environment to another. In order to create any rules of this kind, a vocabulary must be chosen to allow knowledge representation for the relevant domain. This vocabulary should assign human-understandable words to different concepts of traffic. Specifically, the concepts should reflect different factors influencing the correct action of a vehicle. For example, to create a rule which states that a vehicle must stop at a red light, some concepts related to traffic lights, their colors, and stopping must exist. Creation of such vocabulary not only requires an expert of the domain in question but is time consuming and often requires multiple iterations. This work aims to aid these problems by replacing hand-written rules with automatic rule generation and basing the vocabulary used on an existing ontology.

As the main contribution and focus of this work, we propose a process for data-driven algorithmic rule genera-

Manuscript received October 16, 2020.
Manuscript revised February 8, 2021.
Manuscript publicized June 11, 2021.

[†]The authors are with The Graduate University for Advanced Studies SOKENDAI, Tokyo, 101–8430 Japan.

[††]The authors are with National Institute of Informatics, Tokyo, 101–8430 Japan.

a) E-mail: hovi@nii.ac.jp

DOI: 10.1587/transinf.2020EDP7206

Copyright © 2021 The Institute of Electronics, Information and Communication Engineers

tion. In this process, knowledge is represented through a vocabulary consisting of concepts defined in an ontology. In the current form of the process a simpler list of concepts is sufficient as relationships between concepts are not used. However, modifications to the process, such learning generalized association rules [2], [3] instead of basic ones [4] can utilize the additional knowledge ontologies include over simple lists. In addition, ontologies offer a standardized and structured approach to representing concepts within a specific domain and the relations between these concepts. We use an existing ontology designed to be used in ontology-based ADAS, discuss knowledge representation for traffic situations, and examine its expressive power and ability to be used for describing realistic vehicle behavior.

Next, Sect. 2 discusses previous works. Section 3 provides an in-depth introduction to the main contribution of this work: the proposed rule generation process. Section 4 discusses issues related to knowledge representation in traffic scenarios. Section 5 presents an experiment conducted to evaluate the rule generation process in practice and its results with discussion. Finally, Sect. 6 concludes the work by discussing the merits of the work and future research directions.

## 2. Related Work

Decision-making in ADAS is one of the key aspects to solve when implementing practical applications of systems in increasingly automated vehicles. As such, it has seen many different approaches applied to it. Techniques such as artificial neural networks (ANN) have been applied all the way from individual tasks involved in ADAS such as lane detection [5]–[7] for lane following and lane changing purposes to all-encompassing end-to-end systems [8]–[11]. Understanding, verifying, and testing of the behavior of such ANN systems in all situations the target system - in this case ADAS - can encounter is still an active field of research. This poses significant problems when applying ANNs to a safety-critical system such as ADAS.

Human understandable approaches to automated decision-making in traffic include techniques such as usage of state machines [12]–[14] and ontology-based logical reasoning as demonstrated by Zhao et. al. [1]. These systems can benefit from being combined with other systems, for example a state machine-based decision-making system can take advantage of information provided by a state and event detecting ANN [15]. A notable obstacle for use of these human-understandable decision-making systems in highly complex environments - such as traffic - is the creation of the state machines or rulesets to reason on. The work of Zhao et. al. [1] focused on evaluating the feasibility of an ontology-based decision-making system in driving context. While the system used a set of logical rules in the reasoning process, these rules were manually written.

Learning decision trees [16] from human behavior to control robots in rough terrain was presented by Sheh et. al. in 2011 [17]. Sheh et. al. utilized behavioral cloning [18] where a machine learning system builds a model of the decision process of a human selecting appropriate actions for different situations [17].

A key factor for forming explanatory rules is defining the meaning of each atom or building block of the rule. Each atom represents some specific knowledge about the scenario at hand. In traffic these can be, for example, observations of other vehicles or actions of the ego-vehicle. Ontologies define hierarchical concepts such as classes and properties and the relations between these concepts usually within a specific field of interest. These concepts can be used as a vocabulary for rules. Many ontologies specifically designed for use in description of traffic exist. The ADAS ontology by Zhao et. al. [19] is designed for use in different ADAS and defines concepts for a wide variety of traffic situations. The traffic intersection situation description ontology by Hülsen et. al. [20] focuses on intersections and offers ways to accurately describe even complex intersection scenarios through such concepts as degrees of angles between roads and vocabulary relating to vehicle right-of-way and yielding. A survey conducted by Katsumi and Fox published in 2018 [21] examines transportation-related ontologies designed for description of, for example, traffic management [22] and smart cities [23].

Unexpected situations, such as an actor breaking traffic rules, often pose a risk to the safety of the ego-vehicle or other actors involved in the scenario. Identifying and predicting these risks is a crucial task for making correct decisions. Prediction of this nature is often tied to understanding the intentions, such as trajectories, of actors. This kind of intention and risk prediction has been studied by, for example, Takahashi et. al. [24].

## 3. Rule Generation

The main objective of this work is to formulate and propose a process for generating explanatory sets of rules in a data-driven fashion using data collected by a vehicle operating in traffic.

The process of rule generation can be divided into seven steps: data collection, initial vocabulary extraction, data abstraction, rule learning, ruleset refinement, final vocabulary extraction, and rule translation. An overview of this process is shown in Fig. 1 and it can be seen that the seven parts of the process can be grouped into three main categories: data, vocabulary, and rule handling. This section elaborates on what is done within each of the seven steps, what they require as input, and what are their outputs.

### 3.1 Data Collection

As the rule generation process is data-driven, everything begins from collection of data from which the rules will be learned. As such, this step takes in some form of measurement data describing the traffic situation as its input. The goal of this step is to have a dataset of traffic data in some format to be used as an input for the *data abstraction* step.

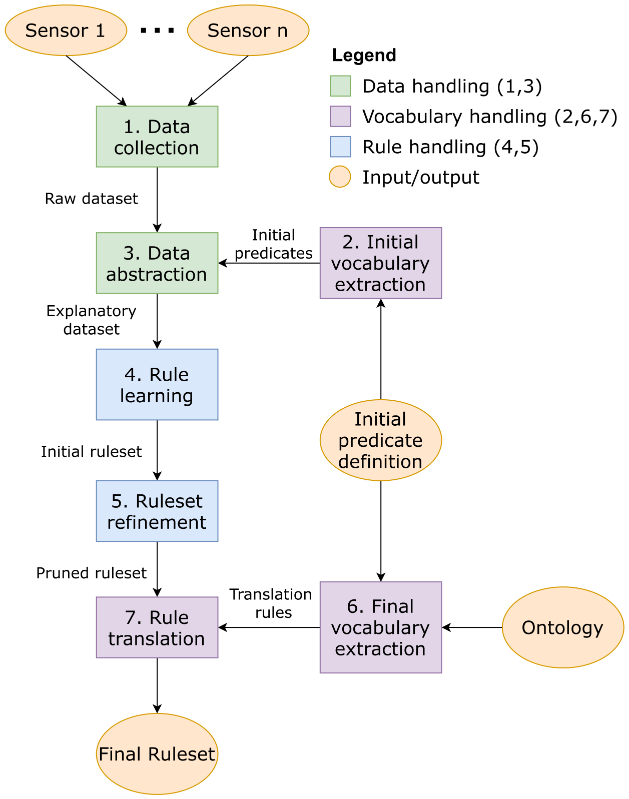

**Fig. 1** The rule generation process flowchart.

The dataset can contain a combination of, among others, raw sensor data from sensors such as Global Navigation Satellite System (GNSS) data consisting of positioning and timing data, three-dimensional point cloud data captured through a Light Detection and Ranging (LiDAR) sensor, images from cameras, or vehicle steering angle and velocity data. Data of higher abstraction level that has already gone through some processing such as an object detection algorithm can also be included. Sensor data is presented in samples, each representing one time-instant. While data from different sensors is often not synchronous, with high enough sampling rate it is possible to combine data from different sources around a certain time-instant to an estimated complete sample of that time-instant. This is possible due to traffic being a relatively slow process where little change happens in the span of, for example, milliseconds. Considering the data as samples of time-instants is beneficial to making subsequent steps of the process simpler. The data can be obtained from either real-world sources or through simulations.

### 3.2 Initial Vocabulary Extraction

Creating explanatory rules requires describing traffic situations with concepts that are understandable for humans, i.e. explanatory. As such, a vocabulary must be established and defined for these concepts. In this step, an initial set of predicates is created to represent concepts for both the actions of the ego-vehicle and for factors that are suspected or known to affect the actions of the vehicle. The rules will first be built using this vocabulary in such a way that any factors affecting ego-vehicle actions appear in the bodies of the rules while ego-vehicle actions appear in the heads of the rules. While it is possible to directly define the final vocabulary (Fig. 1: step 6) introduced in Sect. 3.6 in this step, a separate initial vocabulary can be used to allow flexibility of using different final vocabularies later in the process without repeating the whole process. Additionally, using new higher-level concepts based on the target ontology-defined vocabulary can be useful for creating simpler and more compact rulesets which can then be unpacked in a later step or at a later time. As seen in Fig. 1, this step takes a definition of concepts and their meanings acting as a vocabulary as its input. The output of this step is a collection of n-ary predicates of form

$$p(t_1, \ldots, t_n) \in \{true, false\} \tag{1}$$

where p is an n-ary predicate representing a relation between the objects $t_1$ to $t_n$. In this case, the goal of the predicates is to describe traffic situations through relations of, for example, different actors. In addition, the conditions for when these predicates map to what values must be defined. Thus, the procedure of this step is the following. First, iterate over all predicate definitions in the input. For each definition, extract the name, arity, and type (describes if the predicate represents an action or an observation) of the predicate. Parallel to this, extract the conditions of the predicate truth value and create a function to determine the truth value given a snapshot of a time-instant in the raw dataset. Add the predicate name, arity, type, and its corresponding function to a list. Finally, output the list.

For example, one predicate of interest could be *stop/1*. Here, the name of the predicate is “stop” and its arity is one (unary predicate), meaning the predicate takes one argument to determine it truth value. This predicate could appear in the form

$$stop(Ego) \in \{true, false\} \tag{2}$$

where *Ego* is the ego-vehicle object and *stop/1* would map to true if the speed of the ego-vehicle is approximately zero.

### 3.3 Data Abstraction

At this point, on one hand the data has been collected and is most likely stored as sensor output values. On the other hand a human-understandable vocabulary to describe traffic has been defined. The step of *data abstraction* aims to combine these two inputs and output a dataset of human-understandable and highly abstracted data. An input from step of *initial vocabulary extraction* provides n-ary predicates and their definitions. As such, this step consists of computational data processing by going through each sampled time-instant (see Sect. 3.1) in the data and extracting values for each predicate-objects combination. These combinations are created from the predicates obtained from the

**Algorithm 1** Pseudocode for algorithm to abstract raw data using given initial predicates.

```
1: $D_{in}$: input raw dataset
2: $D_{out}$: output table-format explanatory dataset
3: $P$: set of initial predicates
4: $PO$: set of all valid predicate-objects combinations $po$ of predicates $p \in P$ and objects $o$ present in $D_{in}$
5: for all samples $d \in D_{in}$ do
6:     Let $R$ be a false-initialized array of length $|PO|$
7:     for all predicate-object combinations $po_i \in PO$ do
8:         if $po_i$ is satisfied by $d$ then
9:             set $R_i$ to True;
10:        end if
11:    end for
12:    Add $R$ as a row to $D_{out}$
13: end for
```

**Table 1** An example of representing traffic situation knowledge in item-transaction format usable as input for association rule learning.

| turnLeft(Ego) | signalRed(Ego) | stop(Ego) | go(Ego) |
|---|---|---|---|
| True | False | False | True |
| True | True | False | True |
| False | True | True | False |

initial vocabulary extraction step (Fig. 1: step 2) and the objects present in the data. Pseudocode for this step is shown in Algorithm 1. The data is formatted into a table of Boolean values where each row contains information on one time instant while each column contains the value of one predicate-objects combination at each time instant. If a predicate-objects combination is satisfied by the data, the corresponding cell in the table is set to *true*. If the predicate-objects combination is not satisfied, the cell is set to *false*. Through this process, the original data is wholly abstracted to more human-understandable form using higher-level concepts.

Continuing from the example of Formula 2, the predicate-object combination is predicate *stop/1* with the object *Ego*. As such, we inspect every sample in the data and determine if the speed of the ego-vehicle is approximately zero. If this is true, the corresponding cell in the output table is set to *true* and if this is false, the corresponding cell is left as *false* to which it was initialized.

## 3.4 Rule Learning

Once the raw data has been abstracted to be more human-understandable, the data is passed to the rule learning step. In this step, association rule learning [4] is used to detect patterns in the data and formalize them into rules. This step produces an initial ruleset which is then passed to the next step for refinement.

Association rule learning takes as input a set of data in item-transaction format. In this format, items are assigned Boolean truth values for each transaction. An example of such format is shown in Table 1 with each line representing a transaction or sample from a time instant and each column representing an item. For our purposes, a transaction corresponds to one time-instant in the data while an item corresponds to a predicate-objects combination representing knowledge of the traffic situation. The data is then processed in two steps: discovery of frequent itemsets and rule formation.

A frequent itemset is a set of items which appear frequently together in the data. The threshold of frequency is a user-defined value on a metric called *support*. Support of an itemset X in database of transactions T is calculated through following:

$$Support(X) = \frac{|X|}{|T|} \tag{3}$$

where $|X|$ is the number of transactions in T that include itemset X and $|T|$ is the total number of transactions in T. The discovery of frequent itemsets is often done using the Apriori algorithm [25]. Apriori algorithm is a bottom-up algorithm relying on downwards closure according to which all frequent itemsets of length *k* can be formed by extending frequent itemsets of length *k-1* by adding one item and then pruning any non-frequent itemset.

Once frequent itemsets up to a desired length have been discovered, some can be pruned by exploiting knowledge of the desired rule format. If it is known that the rule antecedent (body) or the consequent (head) can only contain certain items, it is possible to introduce restrictions known as item constraints [26]. In the rule formation step explained below, a rule is formed from a single frequent itemset. These rules include a rule body and a rule head. The goal of the rules is to map knowledge on the ego-vehicle and its environment into correct ego-vehicle action. As such, it is known that the rule head should only contain things that cause effects: vehicle actions. Similarly, rule bodies should only include things we can observe. Thus, any frequent itemset that does not contain both observations and action can be discarded safely: any itemset consisting only of actions or observations can not be used to produce a useful rule. Due to the bottom-up nature of the Apriori algorithm, this optimization can only be done after the Apriori algorithm has discovered all frequent itemsets up to the desired maximum length.

The rule formation process is based on a user-defined threshold value on a metric called *confidence*, which describes how often the rule has been found to be true. The confidence value for a rule $X \implies Y$ can be calculated using the following equation:

$$Confidence(X \implies Y) = \frac{Support(X \cup Y)}{Support(X)} \tag{4}$$

where X and Y are itemsets. X and Y are chosen by splitting a single frequent itemset into two parts and if the confidence value of a rule exceeds the user defined threshold, a rule is formed. Similarly to the optimization of frequent itemset pruning, this step can be optimized using the same knowledge on rule format: only rules with observations in the body and actions in the head should be examined. This step generates rules for which the rule bodies consist of conjunctions of predicates. Additionally, the length of rule head is restricted to *one*. This causes the rules to be simpler and, if desired, any rules with same bodies but different heads can be combined to one rule with a longer head at a later stage. Given this, we produce rules of form

$$p_{b1}(t_i) \land \ldots \land p_{bn}(t_j) \implies p_h(t_k) \tag{5}$$

Where $\{p_{b1}, \ldots, p_{bn}\}$ are observation-related predicates that can be placed into rule bodies, $p_h$ is an action-related predicate that can be place into rule heads, and $\{t_i, t_j, t_k\}$ are sets of objects described by the respective predicates. The number of objects included in, for example, $t_i$ is dictated by the arity of $p_{b1}$.

### 3.5 Ruleset Refinement

Often, the ruleset generated in the previous step includes unwanted rules that can be safely removed without affecting the integrity of the ruleset. This step focuses on refining the initial ruleset given as input into a more concise ruleset through pruning. A pruned ruleset is easier for humans to understand and requires less computation to infer on.

It is possible to identify and remove redundant rules that can increase the computational requirements of inference and clutter the ruleset while never affecting the action of the vehicle. These redundant rules can be pruned by repeatedly looking for two rules $R_1$ and $R_2$ which fulfill two requirements and then removing $R_1$ from the ruleset. These requirements state that

(1) head of $R_2$ must be the same as head of $R_1$, and

(2) body of $R_2$ must be a subset of body of $R_1$.

Importantly, safe removal of redundant rules based on these requirements relies on the assumption that, when using the ruleset, the reasoner only considers rules which have bodies that are satisfied to make a decision. In other words, no knowledge should be gained from a rule for which the body is not satisfied.

Let us continue the example of previous step where rule bodies consist of conjunctions of predicates and rule head length is limited to *one* by examining a ruleset of two rules $R_1$ and $R_2$:

$R_1$:
signalRed(Ego) $\land$ pedestrianOnLeft(Ped) $\implies$ stop(Ego)

$R_2$:
signalRed(Ego) $\implies$ stop(Ego)

where *signalRed(Ego)* indicates that the traffic light affecting the ego-vehicle is red, *pedestrianOnLeft(Ped)* indicates that a pedestrian is crossing or about to cross the road on the left from the point-of-view of the ego-vehicle, and *stop(Ego)* indicates that the ego-vehicle takes the action of stopping. These two rules fulfill requirement (1) as their heads are the same, thus allowing us to limit the examination to the bodies of the rules:

$R_1$:
signalRed(Ego) $\land$ pedestrianOnLeft(Ped) $\in$ {True, False}

$R_2$:
signalRed(Ego) $\in$ {True, False}

Now, it is straightforward to see that if the body of $R_1$ is satisfied, then the body of $R_2$ will also be satisfied. This is due to the body of $R_2$ being a subset of the body of $R_1$ and, as such, the rules fulfill requirement (2). When using these rules, no information should be gained if a rule body is not satisfied. For example, the body of $R_1$ not being satisfied results in no knowledge being gained of the correct vehicle action. The correct action can still be stopping as evidenced by $R_2$ in the case of *signalRed(Ego)* being *true* and *pedestrianOnLeft(Ped)* being *false*. Thus, $R_1$ can be safely removed.

### 3.6 Final Vocabulary Extraction

In case of differences between the vocabulary used by the target system and the initial vocabulary definition, this step allows defining a final vocabulary into which the rules are translated. This allows transferring the rules between different systems despite some differences between the systems. If the initial vocabulary was already defined in a way that does not need translation, this step will define the same predicates.

This step takes in the initial predicate definition used in the initial vocabulary extraction (Fig. 1: step 2) as well as a target ontology or a list of target predicates. To connect the initial predicates to the final predicates, a series of translation rules are made. These translation rules describe a predicate through a conjunction of one or more predicates from another vocabulary and must be satisfied in exactly the same cases in which the original predicate is satisfied.

For example, let predicate *$p_i$/1* be included in the initial predicate definition. However, this predicate is not present in the target vocabulary given as input. Thus, a rule is formed to connect *$p_i$/1* to the target vocabulary. Due to the explanatory and highly abstracted nature of the predicates, these rules are currently written manually. This rule takes the following form:

$$p_0 \land \ldots \land p_n \implies p_i$$

where $p_0$ to $p_n$ are predicates in the target vocabulary. One such rule is formed for each $p_i$ not present in the target vocabulary and passed onto the next step. In the case of formation of such rule being impossible, it might be necessary to increase the expressive power of the vocabulary.

### 3.7 Rule Translation

The final step of the process is the rule translation step which operates on the pruned human-understandable ruleset and the translation rules given as input. This step applies the translation rules to the ruleset by replacing predicates matching the rule head with the body of the translation rule. Additionally, this step translates the rules into the format understood by the target system. At this point, the rules are simply a list of rules with little standardized formatting. The rules are translated into the Semantic Web Rule Language (SWRL) [27] easing the parsing of the rules from a file and transferring the rules between systems. SWRL is a combination of sublanguages of the Web Ontology Language (OWL) [28] and Rule Markup Language (RuleML) [29]. Any relevant metadata can be added to the file in this step. As a result of this step, the final ruleset is ready to be used in the target application.

## 4. Traffic Representation

To create explanatory rules describing actions for a vehicle to perform, it is of utmost importance to define a way to describe traffic and represent related knowledge. The aim of this work is to generate rules which map knowledge of the state and environment of the vehicle into actions. These states and environmental factors are described through a vocabulary representing highly abstracted knowledge concepts based on available data such as sensor feeds. Thus, the choice of vocabulary is important for practical applications of the rule generation process. To conduct an experiment, the ADAS ontology [19] was chosen to be used as a vocabulary to represent necessary information. However, scenarios with multi-step intersection crossing behavior revealed a need for additions to the ADAS ontology that are proposed in Sect. 4.3.

### 4.1 Ontologies

Ontologies are collections of hierarchical concepts usually relating to a specific area or field and the relations between those concepts [30]. As such, ontologies can be considered to contain a vocabulary for a specific field of study. For this work, an ontology created for autonomous driving and advanced driver assistant systems was chosen as the vocabulary to be used. This ontology is the ADAS ontology [19]. As its name suggests, ADAS ontology was created to describe concepts and their relations in traffic in a way that is useful for ADAS [31]. Other ontologies aimed at use in traffic-related applications exist, some of which are introduced in Sect. 2. Many of these ontologies are specialized and highly expressive in their scope. ADAS ontology, however, aims to be more general.

### 4.2 ADAS Ontology

The ADAS ontology introduced by Zhao et al. [19] is comprised of three distinct parts: *Map*, *Control*, and *Car* ontologies. Together, they currently define 159 classes, 35 object properties, and 51 data properties. The *Map* ontology includes concepts to describe and store a map of e.g. roads and intersections, and the connections between different elements. The *Control* ontology describes the controls of a vehicle, such as *stop*, *go*, and its planned path. Finally, the *Car* ontology contains concepts to describe different vehicles and their properties. Being oriented towards driver assistant systems, the ADAS ontology provides vocabulary to describe situations through the point-of-view of the ego-vehicle as opposed to many other ontologies, e.g. one by Hülsen et. al. [20], which describe scenarios without focusing on roles. A more thorough explanation of the ADAS ontology is presented by Zhao et. al. [32], although it has since been expanded slightly.

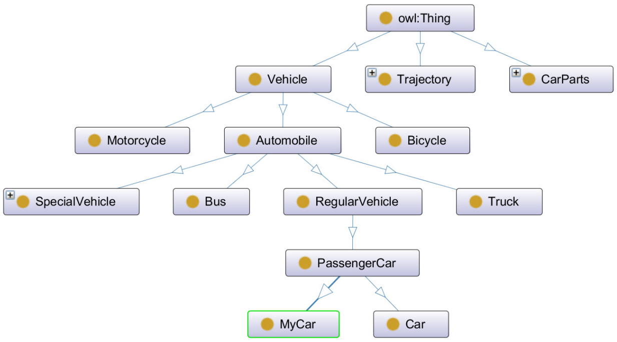

**Fig. 2** Partial visualization of the classes in the *Car* ontology of the ADAS ontology [19].

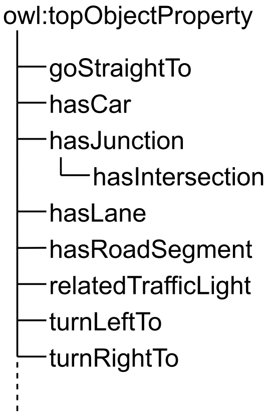

**Fig. 3** Partial visualization of the object properties in the *Map* ontology of the ADAS ontology [19].

A part of the classes in the *Car* ontology is visualized in Fig. 2. The class *MyCar* at the bottom can be used to indicate the ego-vehicle. It can be seen that *MyCar* is a subclass of *PassengerCar* which is a subclass of *RegularVehicle* and so on until the root class of the ontology. Figure 3 on the other hand shows a partial set of object properties defined in the *Map* ontology of the ADAS ontology. These include properties such as *turnLeftTo* and *turnRightTo* which can be used to provide a relative locations between different roads connected to the same intersection. This, in turn, can be used to infer vehicle order as mandated by the applicable traffic law.

### 4.3 Expanding ADAS Ontology

To be able to express more realistic vehicle behavior in a

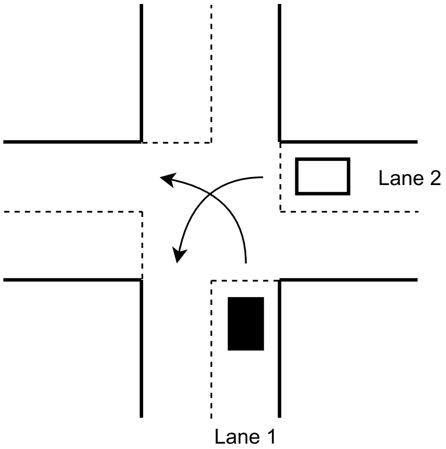

**Fig. 4** Example scenario in an intersection. Ego-vehicle is shown in black while another vehicle is shown in white.

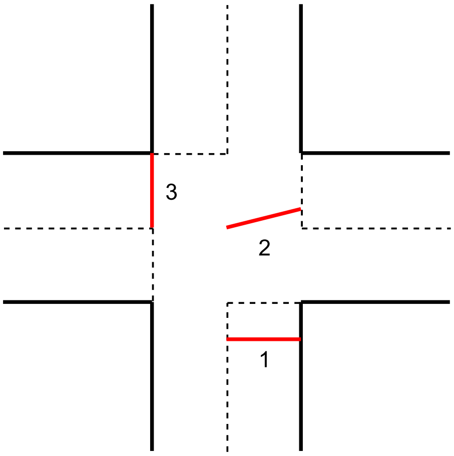

**Fig. 5** Visualization of conceptual stopping location in and around an intersection when approaching from the South and turning left.

more concise and human-understandable way, we propose additions to the ADAS ontology. Let us consider the scenario of Fig. 4 taking place in an uncontrolled intersection with right-hand traffic. The traffic law for the example states: in an uncontrolled intersection, a vehicle must yield if another vehicle is entering the intersection from the right. The ego-vehicle is approaching a four-way intersection as shown in Fig. 4 where the ego-vehicle is shown in black on lane 1. Another vehicle is shown in white on lane 2. In this scenario, the ego-vehicle must yield. Now, the actions of the ego-vehicle could be described using the ADAS ontology as *stop* while the ego-vehicle is waiting outside the intersection for the other vehicle to cross the intersection and *go* once the ego-vehicle is free to proceed. This, however, describes simplified and overly cautious behavior from ego-vehicle that is unlike what a human driver would do. A more likely description of ego-vehicles actions would be that the ego-vehicle proceeds slowly into the intersection and stops at a point which is as far as possible while not interfering with the other vehicle. In some cases the ego-vehicle can proceed far into the intersection while in some the ego-vehicle has to wait outside the intersection. In some cases the ego-vehicle can do multiple stops within one intersection crossing such as in the case of first avoiding a pedestrian by stopping before the intersection and later avoiding a crossing vehicle by stopping inside the intersection. To achieve this behavior using *stop* and *go*, they must be accompanied by some description relating to different positions in and around the intersection. These descriptions do not currently exist in the ADAS ontology.

In an earlier work, *stop* and *go* were used as rule heads to create a ruleset which results in the ego-vehicle waiting for its turn outside the intersection (line number 1 in Fig. 5) [33]. However, as mentioned earlier, this behavior leaves much to be desired in terms of mimicking real-world traffic. To create a ruleset that describes more dynamic and realistic behavior of proceeding into the intersection early, certain areas of the intersection must be reflected in the chosen vocabulary. Currently, this not possible if using the ADAS ontology as the source for the vocabulary.

As such, we propose expanding the ADAS ontology with three new unary predicates for actions related to intersection crossing: *stopAtOne/1*, *stopAtTwo/1*, and *stopAtThree/1*. These three new actions represent conceptual stopping locations around and within the intersection. These stops are visualized in Fig. 5 for a scenario where the ego-vehicle is approaching the intersection from the South and intends to turn left to the West.

New actions describing stopping at certain locations were chosen over using *stop* and *go* with new positional observations due to the temporal differences of the two approaches. An action describing a stop at a certain location can be used as the current action even before the ego-vehicle is at that location. Using the action *stop* with an observation of ego-vehicle position would be the current action only after the ego-vehicle has reached the position. In essence, adding new actions instead of positional observations results in giving the vehicle control system more time to decelerate smoothly and avoid overshooting the stopping position which could risk a collision.

***stopAtOne*** corresponds to stopping location number 1 in Fig. 5. It represents the ego-vehicle stopping before entering the intersection or before a pedestrian crossing in case one is present. This action has many use cases including a scenario where the ego-vehicle is giving way to a pedestrian crossing the road. Another clear example is in a traffic light-controlled intersection when the light affecting the ego-vehicle is red. Additionally, this can be used for cautious intersection crossing to keep the ego-vehicle out of the intersection until the ego-vehicle is free to cross the whole intersection.

***stopAtTwo*** is visualized in Fig. 5 with number 2. This conceptual stop location is inside the intersection to allow the ego-vehicle to enter the intersection even if it is not yet completely clear of other actors that have priority over the ego-vehicle. In the example of Fig. 5, the ego-vehicle could take action *stopAtTwo* even if there is another vehicle approaching from the North and intending to drive straight across the intersection towards the South.

***stopAtThree***, shown in Fig. 5 with number 3, represents a stop that must be made to avoid a pedestrian crossing the road on which the ego-vehicle is planned to drive after crossing the intersection.

The use of these actions can differ on a per application basis. One possibility is to take the actions simultaneously if the ego-vehicle must make more than one stop when crossing the intersection. Another possibility is assigning priorities to the rules based on the sequential nature of the action and considering only the highest priority rule out of the satisfied rules. For example, the ego-vehicle can take actions *stopAtOne* and *stopAtThree* simultaneously when approaching the intersection if pedestrians are crossing the corresponding roads. In this case, *stopAtOne* causes the ego-vehicle to stop before the first pedestrian crossing to give way to a pedestrian. After the pedestrian has crossed, *stopAtOne* is removed from the actions of the ego-vehicle as the rules resulting to taking it should no longer be satisfied. At this point *stopAtThree* remains to be executed. The sequential nature of these actions makes it simple to determine which action out of multiples is the current action or which of the satisfied rules should be followed. Both cases boil down to priority: one assigns priorities to rules while the other assigns priorities to actions.

## 5. Experiment

An experiment was conducted to evaluate the viability of the rule generation process introduced in Sect. 3 and to evaluate the new predicates introduced in Sect. 4.3. The goal of the experiment was to generate rules using the process shown in Fig. 1, use the new predicates, and produce a set of correct rules according to a set evaluation criteria.

### 5.1 Experimental Setup

#### 5.1.1 Vehicle Task and Environment

The scope of the experiment was restricted to scenarios in a four-way intersection. This includes scenarios with two to three vehicles, with and without pedestrians, and with and without traffic lights. The traffic follows the right-hand traffic rules where vehicles drive on the right side of the road and yield to vehicles approaching the same intersection from the right. The task of the vehicle was to:

- cross the intersection by turning left, turning right, or driving straight across,
- avoid collisions with vehicles or pedestrians,
- obey vehicle order mandated by traffic law, and
- obey traffic light control if applicable.

Unexpected situations, such as an actor (vehicle or pedestrian) running a red light or an emergency vehicle taking priority, were not included in the scope of the experiment.

**Table 2** The data sampled from the simulations.

| Name | Description |
|---|---|
| Description | The scenario description used to construct the scenario |
| Timestamp | The time elapsed since the start of the scenario (seconds) |
| Actor-Role | Role of each actor in the scenario (ego-vehicle, other vehicle, pedestrian) |
| Actor-Location | Coordinates of each actor (x, y, z) |
| Actor-Velocity | Velocity of each actor (x, y, z) |
| Actor-Acceleration | Acceleration of each actor (x, y, z) |
| TL-Description | Description of location of each traffic light (N, W, S, E) |
| TL-Location | Coordinates of each traffic light (x, y, z) |
| TL-State | State or color of each traffic light (off, red, yellow, green) |

#### 5.1.2 Simulation and Data Collection

Collection of data (Fig. 1: step 1) was done through simulations. The simulator chosen for the task was the CARLA simulator [34] due to previous experience with the simulator and CARLA being a free simulator under active development aimed at autonomous vehicle and ADAS research.

A software was created to run scenarios and collect data. First, the software algorithmically generated descriptions of the possible three and two vehicle scenarios with and without pedestrians and with and without traffic lights. The description included information such as actors, starting locations of actors, intended actions of actors, and traffic light starting colors. Next, the software ran the scenarios by constructing the scenarios within the simulator. During scenarios, the state of the simulation was sampled and saved repeatedly. Each sample included a set of data required in the step of data abstraction (Fig. 1: step 3). This set of data is described in Table 2. Some of the data is raw numeric data, such as coordinates, while some of the data is more abstract, such as the role of an actor.

#### 5.1.3 Vocabulary and Data Abstraction

A vocabulary for data abstraction was created based on the ADAS ontology [19] and examination of different factors affecting decision making such as which factors traffic laws are based on. Two different types of predicates were chosen: observations and actions. Table 3 shows the chosen observation predicates and their explanations while Table 4 introduces the action predicates. Actions cause the ego-vehicle to behave a certain way while observations describe the current situation. The distinction is made depending on the application and used vocabulary. For example, while *turnLeft* intuitively sounds like an action, here it is used to describe the intended path of a vehicle and thus, is classified as an observation. Knowledge of these intentions allows the decision-making to exploit knowledge on actor trajectories to effectively use free areas of the intersection as human

**Table 3** The observation predicates chosen for data abstraction.

| Predicate | Description |
|---|---|
| turnLeft/1 | A vehicle is intending to cross the intersection by turning left |
| turnRight/1 | A vehicle is intending to cross the intersection by turning right |
| goStraight/1 | A vehicle is intending to cross the intersection by driving straight |
| signalGreen/1 | The traffic light affecting a vehicle is green |
| signalYellow/1 | The traffic light affecting a vehicle is yellow |
| signalRed/1 | The traffic light affecting a vehicle is red |
| signalOff/1 | The traffic light affecting a vehicle is not in use or does not exist |
| pedestrianAtStart/1 | A pedestrian is crossing or intending to cross the road from which the ego-vehicle is approaching the intersection |
| pedestrianOpposite/1 | A pedestrian is crossing or intending to cross the road which is opposite from the ego-vehicle |
| pedestrianOnLeft/1 | A pedestrian is crossing or intending to cross the road which is left from the ego-vehicle |
| pedestrianOnRight/1 | A pedestrian is crossing or intending to cross the road which is right from the ego-vehicle |

drivers often do. Only unary predicates were used in the experiment. However, some of these unary predicates must be expressed through a combination of unary and binary predicates if using vocabulary from the ADAS ontology. The three predicates proposed in Sect. 4.3 are the only new predicates not available through ADAS. However, some predicates and objects used are derived from rules using classes and properties in ADAS ontology to simplify rule verification and evaluation process by producing more compact rulesets. For example, *pedestrianOnLeft/1* appears in the generated rules as a predicate and is not part of the ADAS ontology. *pedestrianOnLeft/1* describes a situation where a pedestrian is crossing or about to cross the road on the left from the point-of-view of the ego-vehicle. Such situation is visualized in Fig. 6. Following the names in the visualization, *pedestrianOnLeft/1* can be expressed using ADAS through following:

1. $\mathrm{myCar}(Ego) \land \mathrm{human}(Ped)$
2. $\land\ \mathrm{lane}(Lane_1) \land \mathrm{lane}(Lane_2)$
3. $\land\ \mathrm{roadSegment}(Road_1)$
4. $\land\ \mathrm{crosswalkRoadSegment}(Crosswalk_1)$

5. $\land\ \mathrm{isRunningOn}(ego, Lane_1)$
6. $\land\ \mathrm{isRunningOn}(Ped, Crosswalk_1)$
7. $\land\ \mathrm{isOn}(Crosswalk_1, Road_1)$
8. $\land\ \mathrm{isOn}(Lane_2, Road_1)$
9. $\land\ \mathrm{turnLeftTo}(Lane_1, Lane_2)$

10. $\Longrightarrow\ \mathrm{pedestrianOnLeft}(ped)$

where lines 1 to 4 define classes for the ego-vehicle, the pedestrian, lanes, road, and a crosswalk. Lines 5 and 6 place the ego-vehicle on $Lane_1$ and the pedestrian on $Crosswalk_1$. Lines 7 and 8 place $Crosswalk_1$ on $Road_1$ and $Lane_2$ on $Road_1$. Line 9 states that turning from $Lane_1$ to $Lane_2$ is a left turn. Finally, line 10 states that if lines 1 to 9 evaluate to true, then the pedestrian in question is on the left from the point-of-view of the ego-vehicle. Similar rules can be derived for the rest of the used predicates and objects except for the three predicates proposed in 4.3.

**Table 4** The action predicates chosen for data abstraction. Numbered stopping locations are visualized in Fig. 5.

| Predicate | Description |
|---|---|
| stopAtOne/1 | A vehicle has stopped before entering the intersection at the conceptual stopping location number 1 |
| stopAtTwo/1 | A vehicle has stopped inside the intersection at the conceptual stopping location number 2 |
| stopAtThree/1 | A vehicle has stopped before exiting the intersection at the conceptual stopping location number 3 |

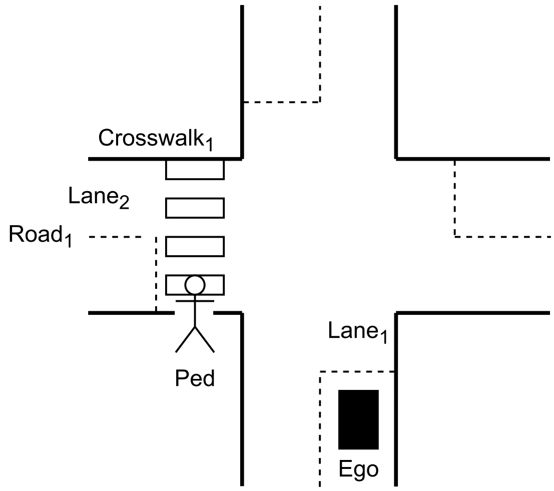

**Fig. 6** Visualization of a situation described by *pedestrianOnLeft/1*.

Once the vocabulary was decided, Algorithm 1 was used in conjunction with knowledge of the intersection to abstract the data. For example, knowledge on the intersection coordinates allows computing whether a vehicle is inside the intersection as coordinates for each vehicle were included in the raw dataset.

### 5.1.4 Rule Structure, Learning, and Refinement

When using the generated ruleset, the ego-vehicle was considered to be proceeding forwards on its planned path if no rules apply. As such, there are no rules produced that require the ego-vehicle to proceed. Only actions included in the initial vocabulary were the three new predicates introduced in Sect. 4.3. In the case of one of these actions being required by a rule, the ego-vehicle can proceed until it arrives at the designated stopping point. This way, multiple rules requiring different stops can be active at the same time, even if only one of them causes an action. For example, a rule stating that the ego-vehicle should take action *stopAtOne(Ego)* and a rule requiring action *stopAtThree(Ego)* being active at the same time will result in the ego-vehicle effectively taking the action *stopAtOne(Ego)* as the stopping location associated with it is reached first. This creates a priority system of sorts.

Rule learning from the abstracted dataset was done using association rule learning. The threshold for support was set to 0.01 and the threshold for confidence was set to 0.80. The final values used for support and confidence were set through manual tuning by inspecting the resulting ruleset for different values. The items in the item-transaction formatted data were the predicates in Tables 3 and 4 combined with objects present in the data while each transaction was a single sampled time-instant of the simulation. Thus, the dataset of abstracted samples can be considered to be in item-transaction format and association rule learning can be applied. Restrictions on the rule structure introduced in Sect. 3.3 state that a rule body can only include observations while a rule head can only contain actions. This knowledge was used to both restrict the output of association rule learning and optimize the algorithm to avoid unnecessary computation as described in Sect. 3.4.

Ruleset refinement was applied to the learned ruleset as introduced in Sect. 3.5 by repeatedly searching for rule pairs matching the requirements for pruning and removing any redundant rules. This search was done until no more requirement fulfilling pairs were found.

#### 5.1.5 Final Vocabulary and Translation

This experiment was conducted to evaluate the proposed process in terms of its ability to produce correct rules. As such, a complete translation of the ruleset was not done due to the translation process not changing the functionality of the rules. Additionally, the three new proposed additions to the ADAS ontology can not be translated to match the ontology. An example of a translation rule for *pedestrianOnLeft/1* is given in Sect. 5.1.3 and similar translation rules could have been written for other applicable predicates.

#### 5.1.6 Evaluation Criteria

Due to a widely accepted form of system verification or evaluation not being available for the application, the evaluation of the generated ruleset was done through review. During this review, each rule as well as the whole set was inspected according to the vehicle task and scope of the experiment, both of which are introduced in Sect. 5.1.1. For the rules and the ruleset to have been considered correct and the process to have been considered successful, the rules and the rulesets were required to fulfill a set of requirements. Requirements 1-3 focus on ensuring that the rules cause correct operation of the vehicle while requirements 4-6 focus on the quality of the rulesets. The requirements are:

(1) The rules must cause the ego-vehicle to fulfill its task of crossing the intersection.

(2) The rules must cause law-obeying behavior, including vehicle order, obeying traffic lights, and giving priority to pedestrians on pedestrian crossings.

(3) The rules must cause avoidance of collisions with other vehicles and pedestrians.

(4) The ruleset must not contain redundant rules that are always covered by other rules.

(5) The ruleset must not contain rules that simultaneously result in contradictory actions.

(6) The ruleset must cover all situations included in the scope of the experiment.

### 5.2 Results

The ruleset generated from scenarios with vehicles and pedestrians is shown in Table 5. Similarly, the ruleset generated from scenarios with vehicles, pedestrians, and traffic lights is shown in Table 6. The predicates appearing in the rules are described in Tables 3 and 4. The objects are described in Table 7.

Let us inspect the rulesets through the evaluation criteria presented in Sect. 5.1.6.

(1) In both rulesets, there is some situation where the ego-vehicle is not required to stop and can continue on its path through the intersection. Thus, the first requirement is fulfilled.

(2) Both rulesets respect the traffic laws in terms of vehicle right-of-way, obeying the relevant traffic light, and giving priority to pedestrians. Thus, the second requirement is fulfilled.

(3) Both rulesets cause the ego-vehicle to avoid collisions by only allowing the ego-vehicle to proceed as far as is safe. Thus, the third requirement is fulfilled.

(4) The rulesets do not contain redundant rules. Thus, the fourth requirement is fulfilled.

(5) The rulesets only contain rules with the three new predicates as their heads. These actions do not contradict each other as explained in Sect. 5.1.4. Thus, the fifth requirement is fulfilled.

(6) Both rulesets contain all situations within the chosen experiment scope where the ego-vehicle must stop. Due to the default action of the vehicle being proceeding on its path, only and exactly the situation where stopping is necessary must be covered. Thus, the sixth requirement is fulfilled.

As seen above, all of the requirements set as the evaluation criteria are fulfilled by both of the rulesets. Thus, we consider the rules to be correct and the rule generation

**Table 5** The set of rules created from scenarios with vehicles and pedestrians with rule head length limited to one. Support > 0.01 and Confidence > 0.8.

| Supp | Conf | Rule |
|---|---|---|
| 0.022 | 1.00 | $turnLeft(Ego) \land pedestrianAtStart(Ped) \land signalOff(Ego) \Rightarrow stopAtOne(Ego)$ |
| 0.023 | 1.00 | $goStraight(Ego) \land pedestrianAtStart(Ped) \land signalOff(Ego) \Rightarrow stopAtOne(Ego)$ |
| 0.022 | 1.00 | $turnRight(Ego) \land pedestrianAtStart(Ped) \land signalOff(Ego) \Rightarrow stopAtOne(Ego)$ |
| | | |
| 0.115 | 1.00 | $turnLeft(Ego) \land pedestrianOnLeft(Ped) \land signalOff(Ego) \Rightarrow stopAtThree(Ego)$ |
| 0.172 | 0.99 | $goStraight(Ego) \land pedestrianOpposite(Ped) \land signalOff(Ego) \Rightarrow stopAtThree(Ego)$ |
| 0.100 | 1.00 | $turnRight(Ego) \land pedestrianOnRight(Ped) \land signalOff(Ego) \Rightarrow stopAtThree(Ego)$ |
| | | |
| 0.020 | 0.95 | $\underline{turnLeft(Ego) \land goStraight(Car_O) \land signalOff(Ego) \Rightarrow stopAtTwo(Ego)}$ |
| 0.019 | 0.97 | $\underline{turnLeft(Ego) \land turnRight(Car_O) \land signalOff(Ego) \Rightarrow stopAtTwo(Ego)}$ |
| 0.020 | 0.93 | $turnLeft(Ego) \land turnLeft(Car_R) \land signalOff(Ego) \Rightarrow stopAtTwo(Ego)$ |
| 0.022 | 0.92 | $turnLeft(Ego) \land goStraight(Car_R) \land signalOff(Ego) \Rightarrow stopAtTwo(Ego)$ |
| 0.020 | 0.91 | $goStraight(Ego) \land turnLeft(Car_R) \land signalOff(Ego) \Rightarrow stopAtTwo(Ego)$ |
| 0.020 | 0.86 | $goStraight(Ego) \land goStraight(Car_R) \land signalOff(Ego) \Rightarrow stopAtTwo(Ego)$ |
| 0.019 | 1.00 | $goStraight(Ego) \land turnRight(Car_R) \land signalOff(Ego) \Rightarrow stopAtTwo(Ego)$ |

**Table 6** The set of rules created from scenarios with vehicles, pedestrians, and traffic lights with rule head length limited to one. Support > 0.01 and Confidence > 0.8.

| Supp | Conf | Rule |
|---|---|---|
| 0.125 | 1.00 | $\boldsymbol{turnLeft(Ego) \land signalRed(Ego) \Rightarrow stopAtOne(Ego)}$ |
| 0.118 | 1.00 | $goStraight(Ego) \land signalRed(Ego) \Rightarrow stopAtOne(Ego)$ |
| 0.118 | 1.00 | $turnRight(Ego) \land signalRed(Ego) \Rightarrow stopAtOne(Ego)$ |
| | | |
| 0.013 | 0.88 | $\underline{turnLeft(Ego) \land goStraight(Car_O) \land signalGreen(Ego) \Rightarrow stopAtTwo(Ego)}$ |
| 0.012 | 1.00 | $\underline{\boldsymbol{turnLeft(Ego) \land goStraight(Car_O) \land signalRed(Ego) \Rightarrow stopAtTwo(Ego)}}$ |
| 0.013 | 0.92 | $turnLeft(Ego) \land turnRight(Car_O) \land signalGreen(Ego) \Rightarrow stopAtTwo(Ego)$ |
| 0.013 | 1.00 | $turnLeft(Ego) \land turnRight(Car_O) \land signalRed(Ego) \Rightarrow stopAtTwo(Ego)$ |
| | | |
| 0.087 | 1.00 | $turnLeft(Ego) \land pedestrianOnLeft(Ped) \land signalGreen(Ego) \Rightarrow stopAtThree(Ego)$ |
| 0.063 | 1.00 | $turnLeft(Ego) \land pedestrianOnLeft(Ped) \land signalRed(Ego) \Rightarrow stopAtThree(Ego)$ |
| 0.071 | 1.00 | $turnRight(Ego) \land pedestrianOnRight(Ped) \land signalGreen(Ego) \Rightarrow stopAtThree(Ego)$ |
| 0.059 | 1.00 | $turnRight(Ego) \land pedestrianOnRight(Ped) \land signalRed(Ego) \Rightarrow stopAtThree(Ego)$ |

**Table 7** The objects present in the generated rulesets and their explanations.

| Object | Description |
|---|---|
| *Ego* | The ego-vehicle. |
| *Ped* | A pedestrian. |
| $Car_L$ | A vehicle approaching the intersection from left from the point-of-view of the ego-vehicle. |
| $Car_R$ | A vehicle approaching the intersection from right from the point-of-view of the ego-vehicle. |
| $Car_O$ | A vehicle approaching the intersection from opposite side of the intersection from the point-of-view of the ego-vehicle. |

process to have been successful. The new predicates introduced in Table 4 enabled description of stops at different positions in and around the intersection and thus fulfilled their intended purpose.

## 5.3 Discussion

The conducted experiment covered most of the proposed process. Rule translation step was demonstrated, but not exhaustively applied. The results show that the proposed process was able to generate correct rulesets within the desired scope from simulated data and no incorrect rules were included in the rulesets. Rules from different intersection scenarios matching suggests that the process can be applied to different situations to generate rulesets with higher coverage of different traffic situations.

Some of the generated rules, such as the first and fifth rules in Table 6 (marked in bold), can be satisfied simultaneously. This effectively results in two actions having to be taken simultaneously. However, the actions were designed to be sequential in nature and one is always higher priority. That is, *stopAtOne* always occurs before *stopAtTwo* and *stopAtThree* always occurs last. Thus, both rules being satisfied at the same time poses no problems. However, the reasoner has to accommodate such relations between different actions, possibly through a priority system. This highlights the relationship between choices made during the ruleset generation process and the design of the reasoner. Either the target system should be understood when defining vocabularies and refining the rulesets or the target system must be designed with the properties of the ruleset in mind.

Rule learning was done through association rule learning [4] using the Apriori algorithm [25] to discover frequent itemsets and using support and confidence thresholds for itemset frequency and rule significance. This traditional approach is known to have several shortcomings. Moreno et. al. list obtaining non-interesting rules, huge number rules, and low algorithm performance as the main drawbacks [35] while Ng et. al. focus on opening the black box to allow user guidance during learning [36]. While the results of this experiment were deemed to be the correct and desired results, the complexity of the rules is likely to increase on the way towards practical applications. As such, improved techniques of association rule learning could be used. Different measures of rule usefulness can be used to refine the rulesets. Examples of such measures are *unexpectedness* and *actionability* as defined by Liu et. al. [37]. The hierarchical nature of ontologies could prove to be useful if methods of learning generalized association rules [2], [3] are used.

Many rules resemble each other with only the traffic light color being different. For example, the following rules are included in either Table 5 or 6 (marked in underline):

$$turnLeft(Ego) \wedge goStraight(Car_O) \\ \wedge signalOff(Ego) \implies stopAtTwo(Ego)$$

$$turnLeft(Ego) \wedge goStraight(Car_O) \\ \wedge signalGreen(Ego) \implies stopAtTwo(Ego)$$

$$turnLeft(Ego) \wedge goStraight(Car_O) \\ \wedge signalRed(Ego) \implies stopAtTwo(Ego)$$

If the absence of yellow traffic light is ignored due to it being possible to be interpreted as red or green, at a glance these three rules seem to be a candidate for combining them into a single rule by removing the atom describing traffic light:

$$turnLeft(Ego) \wedge goStraight(Car_O) \\ \implies stopAtTwo(Ego)$$

This kind of ruleset refinement is difficult to generalize. It is heavily reliant on the meaning of each predicate as defined by humans while the pruning described in Sect. 3.5 relies on mathematical properties of conjunctions. However, this refinement could be done with some restrictions: rule heads must be the same, rule bodies must be the same except for one atom, and the differing atoms must appear in all possible variations out of which one variation is always present in the scenario. This is the case for the above three rules assuming the traffic light would always be *off*, *green*, or *red*. Due to the generalization challenges, this refinement technique was not described in the process or implemented in the experiment.

## 6. Conclusion

This paper proposed a process for generating explanatory rules for use in ADAS from raw data. The process covers everything from data collection to translating the ruleset to use the desired vocabulary. On the way towards simulating scenarios more complex than in the author's previous work [33], pedestrians and traffic lights were added to intersection scenarios. Due to this, depicting more realistic vehicle behavior than before using human-understandable vocabulary became desirable. Thus, we proposed three additions to the ADAS ontology to describe vehicle actions in an intersection: *stopAtOne*, *stopAtTwo*, and *stopAtThree*.

The proposed process was demonstrated to be able to generate correct rulesets for the chosen experiment scope. This suggests the process is viable for generating explanatory rules from data and should be applied to more diverse datasets to further examine its applicability for different datasets.

The three new classes of actions proposed to be added into the ADAS ontology were successfully used to capture more realistic vehicle behavior than in author's previous work [33] where only actions used were *stop* and *go*. The proposed new actions are inherently sequential in nature and thus can easily be prioritized over one another during the decision-making process.

In the future, improvements of the proposed process can come from different directions, such as examination of using different rule learning methods. In addition to methods which directly learn rules, other classifiers such as ones producing decision trees could be used, for example C4.5 introduced by Quinlan in 1993 [16]. C4.5 generates a decision tree to classify an example based on a set of features. In this case, the features or decision points in the tree would be different values for observations while classes would be ego-vehicle actions. Other future research directions include applying the process to datasets containing real-world driving data and unexpected situation as well as automated assignment of rule priority for the generated rules.

## Acknowledgments

We wish to thank Kotaro Otomura and Koji Tanaka from Mitsubishi Electric and Tatsuya Urabe. They provided valuable help in both defining new knowledge representation vocabulary to increase the expressiveness of used vocabulary and reviewing the paper.

### References

[1] L. Zhao, R. Ichise, Z. Liu, S. Mita, and Y. Sasaki, "Ontology-based driving decision making: A feasibility study at uncontrolled intersections," IEICE Trans. Inf. & Syst., vol.E100.D, pp.1425–1439, 2017.

[2] R. Srikant and R. Agrawal, "Mining generalized association rules," Future Generation Computer Systems, vol.13, no.2, pp.161–180, 1997. Data Mining.

[3] J. Han and Y. Fu, "Mining multiple-level association rules in large databases," IEEE Trans. Knowl. Data Eng., vol.11, no.5, pp.798–805, 1999.

[4] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," SIGMOD Rec., vol.22, no.2, pp.207–216, 1993.

[5] J. Kim and M. Lee, "Robust lane detection based on convolutional neural network and random sample consensus," Neural Information Processing, ed. C.K. Loo, K.S. Yap, K.W. Wong, A. Teoh, and K. Huang, vol.8834, pp.454–461, Springer, 2014.

[6] J. Li, X. Mei, D. Prokhorov, and D. Tao, "Deep neural network for structural prediction and lane detection in traffic scene," IEEE Trans. Neural Netw. Learn. Syst., vol.28, no.3, pp.690–703, 2017.

[7] A. Gurghian, T. Koduri, S.V. Bailur, K.J. Carey, and V.N. Murali, "Deeplanes: End-to-end lane position estimation using deep neural networks," 2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp.38–45, 2016.

[8] M. Bojarski, D.D. Testa, D. Dworakowski, B. Firner, B. Flepp, P.

Goyal, L.D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to end learning for self-driving cars," arXiv:1604.07316v1, 2016.

[9] U. Muller, J. Ben, E. Cosatto, B. Flepp, and Y.L. Cun, "Off-road obstacle avoidance through end-to-end learning," in Advances in Neural Information Processing Systems 18, ed. Y. Weiss, B. Schölkopf, and J.C. Platt, pp.739–746, MIT Press, 2006.

[10] D.A. Pomerleau, ALVINN: An Autonomous Land Vehicle in a Neural Network, pp.305–313, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1989.

[11] F. Codevilla, M. Müller, A. López, V. Koltun, and A. Dosovitskiy, "End-to-end driving via conditional imitation learning," 2018 IEEE International Conference on Robotics and Automation (ICRA), pp.4693–4700, 2018.

[12] S. Kato, E. Takeuchi, Y. Ishiguro, Y. Ninomiya, K. Takeda, and T. Hamada, "An open approach to autonomous vehicles," IEEE Micro, vol.35, no.6, pp.60–68, 2015.

[13] A. Kurt and Ümit Özgüner, "Hybrid state system development for autonomous vehicle control in urban scenarios," IFAC Proceedings Volumes, vol.41, no.2, pp.9540–9545, 2008. 17th IFAC World Congress.

[14] Q. Chen, U. Ozguner, and K. Redmill, "Ohio state university at the 2004 darpa grand challenge: developing a completely autonomous vehicle," IEEE Intell. Syst., vol.19, no.5, pp.8–11, 2004.

[15] D.O. Sales, D.O. Correa, L.C. Fernandes, D.F. Wolf, and F.S. Osório, "Adaptive finite state machine based visual autonomous navigation system," Engineering Applications of Artificial Intelligence, vol.29, pp.152–162, 2014.

[16] J.R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[17] R. Sheh, B. Hengst, and C. Sammut, "Behavioural cloning for driving robots over rough terrain," 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp.732–737, 2011.

[18] M. Bain and C. Sammut, "A framework for behavioural cloning," Machine Intelligence 15, Intelligent Agents [St. Catherine's College, Oxford, July 1995], GBR, pp.103–129, Oxford University, 1999.

[19] Ichise Laboratory, "Advanced driver assistant system ontology." http://ri-www.nii.ac.jp/ADAS/index.html.

[20] M. Hülsen, J.M. Zöllner, and C. Weiss, "Traffic intersection situation description ontology for advanced driver assistance," 2011 IEEE Intelligent Vehicles Symposium, pp.993–999, 2011.

[21] M. Katsumi and M. Fox, "Ontologies for transportation research: A survey," Transportation Research Part C: Emerging Technologies, vol.89, pp.53–82, 2018.

[22] A.J. Bermejo, J. Villadangos, J.J. Astrain, and A. Córdoba, "Ontology based road traffic management," Intelligent Distributed Computing VI, ed. G. Fortino, C. Badica, M. Malgeri, and R. Unland, Berlin, Heidelberg, vol.446, pp.103–108, Springer Berlin Heidelberg, 2013.

[23] P. Bellini, M. Benigni, R. Billero, P. Nesi, and N. Rauch, "Km4city ontology building vs data harvesting and cleaning for smart-city services," Journal of Visual Languages & Computing, vol.25, no.6, pp.827–839, 2014. Distributed Multimedia Systems DMS2014 Part I.

[24] R. Takahashi, N. Inoue, Y. Kuriya, S. Kobayashi, and K. Inui, "Explaining potential risks in traffic scenes by combining logical inference and physical simulation," International Journal of Machine Learning and Computing, vol.6, no.5, pp.248–255, 2016.

[25] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases," Proceedings of 20th International Conference on Very Large Data Bases, pp.487–499, Morgan Kaufmann, 1994.

[26] R.J. Bayardo, R. Agrawal, and D. Gunopulos, "Constraint-based rule mining in large, dense databases," Proceedings 15th International Conference on Data Engineering (Cat. no.99CB36337), pp.188–197, 1999.

[27] I. Horrocks, P.F. Patel-Schneider, H. Boley, S. Tabet, B. Grosofand, and M. Dean, "SWRL: A semantic web rule language combining OWL and RuleML." https://www.w3.org/Submission/SWRL/.

[28] S. Bechhofer, F. Harmelen, J. Hendler, I. Horrocks, D. Mcguinness, P. Patel-Schneider, and L. Stein, "Owl web ontology language reference," 02 2004.

[29] RuleML Inc., "RuleML." http://wiki.ruleml.org/index.php/RuleML_Home.

[30] W3C, "Semantic web: Vocabularies." https://www.w3.org/standards/semanticweb/ontology.

[31] L. Zhao, R. Ichise, S. Mita, and Y. Sasaki, "Core ontologies for safe autonomous driving," International Semantic Web Conference (Posters & Demos), 2015.

[32] L. Zhao, R. Ichise, S. Mita, and Y. Sasaki, "Core ontologies for safe autonomous driving," Proceedings of the ISWC Poster and Demonstrations Track, 2015.

[33] J. Hovi and R. Ichise, "Feasibility study: Rule generation for ontology-based decision-making systems," Semantic Technology, vol.1157, pp.88–99, Springer, 2020.

[34] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," Proceedings of the 1st Annual Conference on Robot Learning, pp.1–16, 2017.

[35] M. Moreno García, S. Segrera, and V. Batista, "Association rules: Problems, solutions and new applications abstract," Actas del III Taller Nacional de Minería de Datos Y Aprendizaje, 01 2005.

[36] R.T. Ng, L.V.S. Lakshmanan, J. Han, and A. Pang, "Exploratory mining and pruning optimizations of constrained associations rules," SIGMOD Rec., vol.27, no.2, p.13–24, June 1998.

[37] B. Liu, W. Hsu, S. Chen, and Y. Ma, "Analyzing the subjective interestingness of association rules," IEEE Intelligent Systems and their Applications, vol.15, no.5, pp.47–55, 2000.

**Juha Hovi** received the degree of M.Sc. (Technology) from Aalto University in 2019. He is currently a Ph.D. student in The Graduate University for Advanced Studies SOKENDAI. His research interests include automated systems and artificial intelligence.

**Ryutaro Ichise** received his Ph.D. degree in computer science from Tokyo Institute of Technology, Tokyo, Japan, in 2000. From 2001 to 2002, he was a visiting scholar at Stanford University. He is currently an associate professor in Principles of Informatics Research Division at the National Institute of Informatics in Japan. His research interests include machine learning, semantic web, and data mining.