Identifying Driver's Cognitive Load Using Inductive Logic Programming

Fumio Mizoguchi†‡, Hayato Ohwada†, Hiroyuki Nishiyama†, and Hirotoshi Iwasaki*

Faculty of Sci. and Tech. Tokyo University of Science†,
2641 Yamazaki, Noda-shi,
CHIBA, 278-8510, Japan
WisdomTex Co. Ltd.‡,
1-17-3 Meguro-ku Meguro, Tokyo 153-0063, Japan
Denso IT Laboratory*
mizo@wisdomtex.com, ohwada@rs.tus.ac.jp, hiroyuki@rs.noda.tus.ac.jp,
hiwasaki@d-itlab.co.jp

Abstract. This paper uses inductive logic programming (ILP) to identify a driver's cognitive state in real driving situations to determine whether a driver will be ready to select a suitable operation and recommended service in the next generation car navigation systems. We measure the driver's eye movement and collect various data such as braking, acceleration and steering angles that are qualitatively interpreted and represented as background knowledge. A set of data about the driver's degree of tension or relaxation regarded as a training set is obtained from the driver's mental load based on resource-limited cognitive process analysis. Given such information, our ILP system has successfully produced logic rules that are qualitatively understandable for rule verification and are actively employed for user-oriented interface design. Realistic experiments were conducted to demonstrate the learning performance of this approach. Reasonable accuracy was achieved for an appropriate service providing safe driving.

1 Introduction

The next-generation driving support system requires both highly sophisticated functions and user-friendly interfaces (telephone or e-mail communication, audio-visual service, etc.) according to the driver's situation. This support system must therefore recognize the driver's cognitive state to determine whether he or she will be ready to select a suitable operation and recommended service. Obviously, it is better to provide information when the user is relaxed rather than tense.

The next-generation services should judge the driver's mental load by considering both the driver's individual characteristics and operations, and the driver mental load considered that driver is unconscious. The mental load has been analyzed by mathematical equations, stochastic models, and engineering control models in the fields of bionics and brain science[6, 4, 7]. Although those were

based on the biological understanding and clarification of the living body itself, the important point here is how the human cognitive state can be specified in terms of the states of living body. However, such specification may be hard to investigate because of the ambiguity induced by individual differences or unconscious reactions.

In this paper, we focus on how a user's cognitive load can be determined, analyzed, and used from vital reactions (eye movement) and user operations (driving a car). To do this, we measure the driver's eye movement and gather driving data such as accelerator use, braking, and steering. Eye movement is used in the field of physiological psychology for clarifying control[3] and is directly related to perception that can be considered an indication of cognitive load. Driving a car requires cognition and prediction of the surrounding environment and is influenced by situations. Furthermore, it includes reflex action interruptions and resource competition for performing two or more operations simultaneously[12].

This paper takes an ILP approach to the above cognitive state identification problem in a realistic car-driving task. We set up binary cognitive loads (relaxed or tense) that are analyzed by interviewing the driver to determine his cognitive state with regard to driving video review and use the obtained data as a set of training examples. For background knowledge, we identify an object the driver watches and collect driving data obtained from an in-vehicle LAN; those data are then processed as qualitative data. Given such information, our ILP system has successfully produced logic rules that are qualitatively understandable for rule verification and refinement. We conducted realistic experiments to demonstrate the learning performance of this approach, and obtained reasonable accuracy for designing really useful user-interfaces.

This paper is organized as follows. Section 2 presents raw data obtained from an in-vehicle LAN. Section 3 arranges these data that are transformed into qualitative descriptions in Section 4. Section 5 describes background knowledge, and Section 6 provides a way to set up training data. Section 7 introduces learned rules, and Section 8 includes performance evaluation. The final section provides conclusions.

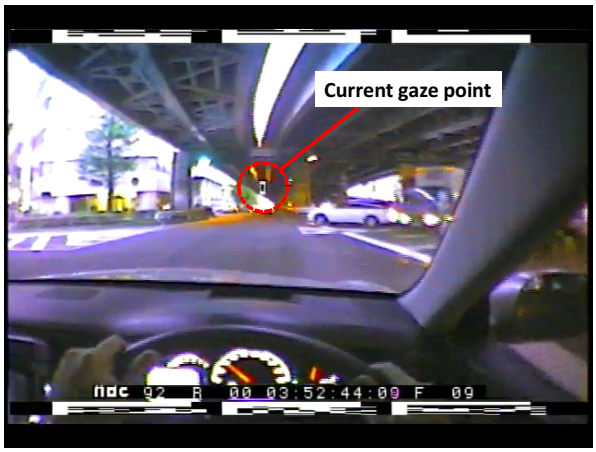
2 Raw Data

We used an eye movement tracking device [1]. The device can measure horizontal and vertical viewing angles in degrees. We obtained 60 data points per second.

The Controller Area Network [2] (CAN) is an in-vehicle LAN used to gather driving data. We can obtain the accelerator depression rate (0% to 100%), brak-

[1] EMR-8 is produced by NAC Image Teck., Inc. (www.eyemark.jp)

[2] This is a standard used for the data transfer between in-vehicle equipment. The International Organization for Standardization is standardizing it as ISO 11898 and ISO 11519.



No.	Timecode hh:mm:ss:nn	Front signal	Accelerator rate	Brake signal	Velocity	Steering angle	Gaze_X	Gaze_Y
6895	03:52:43:54	0	15	0	39.5	−3	−4.6	5.8
6896	03:52:43:59	0	15	0	39.62	−3	−4.8	7.4
6897	03:52:44:05	0	15	0	39.8	−3	−4.4	7.3
6898	03:52:44:11	0	15	0	39.78	−3	−3.6	6.5
6899	03:52:44:17	0	15	0	39.67	−3	−2.3	5.1
		Driving Data					*Gaze Data*	

Fig. 1. Eye movement and driving data

ing signal (0 or 1), steering signal (-450 to 450 degrees), a signal representing the gear (0 to 4), the front separation (in meters), and so on. To measure these, we modified a Toyota Crown and obtained 10 data points per second.

The eye-movement data and the car-driving data must be synchronized as indicated in Fig. 1. The third row data (No. 6897) states that the car's velocity is 39.8km/h with an accelerator depression rate of 15% and is running almost straight (steering angle -3); the driver is gazing at the center (the coordinate is (-4.4,7.3)). All of the data can be used to produce background knowledge as demonstrated in the next section.

3 Data Arrangement

Eye movement indicates where a person is looking. It also expresses caution and concern, and is used for evaluating software usability [11].

When a person looks at something every day, the eyes repeat cycles of rapid movement and stationary periods. The rapid movements are called "saccade," and the stationary periods are called "fixation" [8]. In saccade, the eyes rotate to position the target at the central fovea of the retina. Saccade is defined as movement that rotates the eye at speeds of 100 degrees/second or more. Movement that rotates the eyes at 100 degrees/second or less is called "pursuit" because the eye pursues a smoothly moving target. Because saccade is related with capturing a target, it is closely related to the observer's motivation and cognitive process.

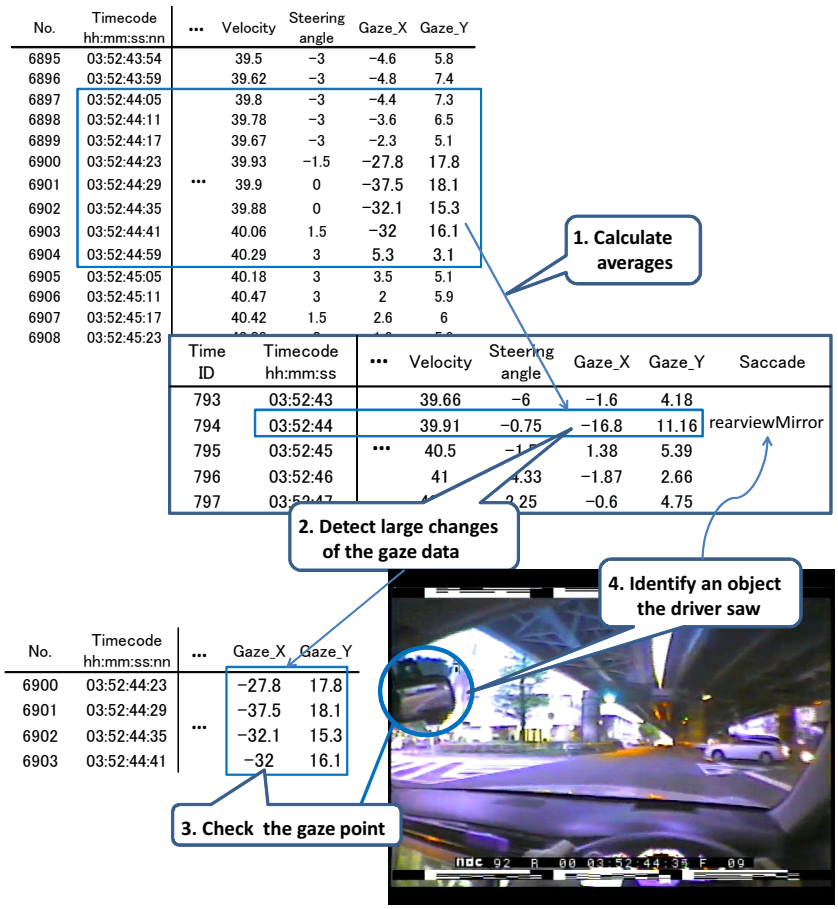


Fig. 2. Data arrangement process for each second. Driving data is averaged, and saccade events are detected.

We identify a driver's cognitive load induced by saccade in one second, and average raw data for each second. By tracking saccade events, we detect targets such as the rearview mirror, car navigation device, road signs, and pedestrians. This raw data arrangement consists of the following steps:

Step 1. Collect a set of raw data measured in a second, and average each attribute value of driving data (Fig. 2, Step 1).

Step 2. Detect a large change of eye movement that reveals a saccade event, and obtain a set of time codes (Fig. 2, Step 2).

Step 3. Calculate the gaze point area corresponding to saccade from raw data (Fig. 2, Step 3).

Step 4. Identify an object (*e.g.* rearview mirror) within the gaze point through video analysis (Fig. 2, Step 4). If the object is identified, it is used as the value of "saccade" attribute.

Step 4 is performed manually (not automatically) at the current stage. In Fig. 2, rearview mirror is identified as the saccade target.

4 Data Transformation

The data-transformation phase gathers time-series saccade events and generates qualitative expressions of driving data. This is inspired by the research of qualitative reasoning that focuses on the change of variables such as an increase or decrease and the relation between variables such as proportionality and differentiation [9]. Although qualitative reasoning has also been employed in ILP literature [1, 2, 14], we use qualitative reasoning to deal with driver's individual characteristics, unconscious reactions, and human situated cognition [5].

A driver's cognitive state can be qualitatively characterized by a resource-limited model [12]. In driving a car, a resource-required action corresponds to an accelerator operation making the vehicle speed up, and a resource-free action, to an operation to slow the vehicle. Although this model is simple and abstract, qualitatively describing driving data may help to capture the driver's cognitive state [13].

Some averaged real data are mapped to a small number of categories. In contrast to braking signals and front signals (indicating whether a car exists in front) that are binary attributes, accelerator rate and steeling angle are handled as four qualitative values (`zero`, `low`, `middle`, `high`) based on the quantiles on the standard normal distribution. A value for velocity is divided to the ordered set (`zero`, `<10`, `20`, `<30`, `<40`, `<50`, `>=50`) according to normal town-street driving. Moreover, parameter changes in qualitative reasoning are extended to the four categories [3] (`low`, `middle`, `high`, `veryHigh`) with two directions of change (`up`, `down`) [4]. This means that a qualitative state difference between the current state and the next state takes one of the eight forms.

Although a driver's cognitive state can be characterized by a sequence of saccades, we employ only adjacent saccades in a short time period (5 to 10 seconds) to predict the cognitive state for recognition and decision-making. This includes driving data after five seconds because some operations are intended by the driver and are considered as predictable changes for the driver. Thus, driving data for a small number of saccade events are considered for background knowledge.

Figure 3 illustrates the process of data transformation consisting of the following steps:

Step 1. Collect an ordered set of saccade event data in which "saccadeID" is inserted for each data.

Step 2. Add new attributes indicating the differences in the short time period (five seconds before and after) where each difference is represented by Δ-second.

[3] They are calculated by using the quantiles on the standard normal distribution.

[4] `noChange` is used to indicate that there is no change.

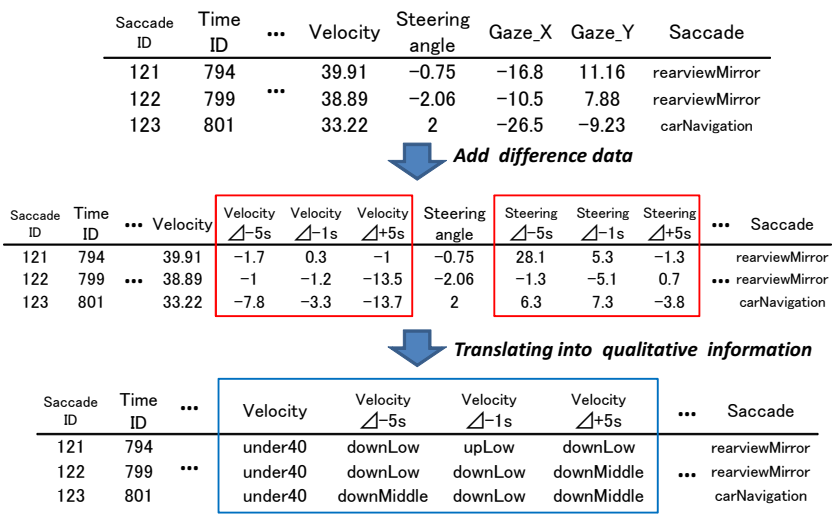


Fig. 3. Data transformation

Step 3. Translate the data at Step 2 into the corresponding qualitative data using the above categories.

5 Background Knowledge

Table 1 presents a set of predicate types and mode declarations in background knowledge. The first type corresponds to qualitative values for each eye movement and driving data. This is described by the saccade ID and a parameter value. The second type is a qualitative state difference in a short time period and is described as *parameter*`_diff(ID,Time,Val)` where `Time` indicates the time difference in seconds. The third one (`before_event` and `after_event`) is used to get information of adjacent saccades. Note that multiple adjacent saccades exist within five seconds in some cases.

Table 1. Predicates and their mode declarations in background knowledge. Mode `+` indicates input variable, `-` output variable, and `#` constant.

Types	Predicates
qualitative value	`accele(+ID, #Val)`, `brake(+ID, #Val)`, `velocity(+ID, #Val)`, `steering(+ID, #Val)`, `front(+ID, #Val)`, `gazeX(+ID, #Val)`, `gazeY(+ID, #Val)`, `lookAt(+ID, #Target)`
qualitative state difference	`accele_diff(+ID,#Time,#Val)`, `brake_diff(+ID,#Time,#Val)`, `velocity_diff(+ID,#Time,#Val)`, `front_diff(+ID,#Time,#Val)`, `steering_diff(+ID,#Time,#Val)`, `gazeX_diff(+ID,#Time,#Val)`, `gazeY_diff(+ID,#Time,#Val)`
adjacent saccades	`before_event(+ID, -ID)`, `after_event(+ID, -ID)`

6 Training Examples

We set up binary cognitive loads (relaxed or tense) that are analyzed by interviewing the driver to determine his cognitive state with regard to a driving video review. A set of data about the driver's degree of tension or relaxation is regarded as a training set and is obtained by also considering the driver's mental load based on the resource-limited cognitive process model [12].

The resource-limited model indicates the mechanism used to coordinate the resource consumption through the repeated execution of driver operations and eye movements. For example, the accelerator increases speed, and the used resource will increase. If too much resource is used, a resource-free action like braking is invoked. A decision about the cognitive load is based mainly on an interview with the driver, but the resource-limited model is used to check the decision under the following criteria:

tense: If saccade occurs and time resource-free actions are executed, the used resource is full. In this case, the driver's cognitive load is heavy.

relaxed: If saccade occurs and the resource is stable or resource-required actions are executed, the resource is available for use. This case indicates that the driver is relaxed.

A cognitive state is represented as `class(+ID,#Class)` where `Class` is either `tense` or `relaxed`. These two classes are mutually exclusive; a positive example `class(1,tense)` is regarded as the negative example of `class(1,relaxed)`.

7 Learned Rules

We measured driving and eye-movement data of a skilled driver for about 20 minutes driving on two types of road, an urban road with much traffic (Shibuya area) and a road with moderate traffic (Noda area). Table 2 presents the statistics of raw data, arranged data, and saccade-event data.

Table 2. Data used for experiment

Area	Measured time (second)	Raw data	Arranged data for each second	Saccade event data
Shibuya	1345	80780	9415	220
Noda	1310	72429	9170	240

Saccade event data is classified as either `tense` or `relaxed`, and Table 3 indicates its class distribution.

Our ILP system (GKS[10]) is employed to learn rules for each driving area. The machine we used is Windows 7 OS with two 2.40GHz Intel(R) Core(TM) i5 CPUs with 4GB memory. The system generated 33 rules (17 for tense and 16 for relaxed) for the Shibuya area and 39 rules (14 for tense and 25 for relaxed)

Table 3. Class distribution of the saccade event data

Area	tense	relax	Total
Shibuya	105	115	220
Noda	117	123	240

for the Noda area. Learning times were 533 seconds (Shibuya) and 2400 seconds (Noda) [5].

We present typical rules below. "`{T,F}`" denotes the number of positive examples (T) and the number of negative examples (F) the rule covers.

```
{21,0} class(A, tense) :-
         before_event(A, B), lookAt(B, corner), accele(B, zero).
```

The first rule *naturally* describes our understanding in which a driver becomes tense when he looks at the corner of the front window without pressing the accelerator. In this case, he is making a turn.

The following rule indicates another tense state:

```
{20,0} class(A, tense) :-
        before_event(A, B), before_event(B, C), brake(B, on),
        brake_diff(C, -1 noChange), gazeY(C, front).
```

This rule indicates that multiple saccades occur in a short time period (denoted by the variables `A` and `B`) with continued braking regardless of watching the front direction. This is not a trivial rule represented in terms of *non-determinate* predicates (`before_event`), indicating the advantage of an ILP-based learner.

The above two rules indicate that the driver's cognitive load is heavy, and thus additional service should not be provided. In contrast, the following rules indicate when service can be provided:

```
{35,0} class(A, relax) :-
         brake(A, off), brake_diff(A, +5, noChange),
         steering(A, straight).
```

This rule *naturally* describes a normal driving situation in which a driver becomes relaxed when going straight without continued braking. The following is another rule indicating a relaxed driver:

```
{15,0} class(A, relax) :-
         after_event(A, B), brake(B, off),
         front_diff(B, -5, noChange), steering_diff(A, +5, rightLow),
         steering_diff(B, -5, rightLow).
```

[5] GKS system and driving data we used here can be accessed on our homepage: `http://www.wisdomtex.com`

This rule is more complicated because it predicts the driver's action. The predicate `after_event` is used for a predictable state, representing that the driver will release the brake and go straight while driving smoothly. The rule cannot be applied in our current car navigation system, but it may be useful for next-generation systems.

We obtained unexpected rule:

```
{10,0} class(A, tense) :-
        velocity(A, under10), velocity_diff(A, -1, upLow),
        velocity_diff(A, +5, upLow).
```

This rule says that the driver is tense even when the driving speed is low and slightly accelerating. Although the driver seems to have available resources, he carefully looked in front of the car. Our rule is based on saccade events, but in this case there is only one saccade, so missing some critical intentional action executions.

8 Performance Evaluation

Experiments were conducted to achieve the following.

Aim1 seeks to describe the difference of accuracy among three types of background knowledge. Background knowledge B1 includes only qualitative parameter values of driving data. Background knowledge B2 adds qualitative differences of parameter changes to B2. Background knowledge contains all information including the information about adjacent saccade events.

Aim2 attempts to derive the learning curve based on the assumption of real-time learning. We divide training examples into ten subsets in time sequence. In this setting, training examples progressively increase in time order.

Aim3 is designed to describe the accuracy of rules in applying another new road.

In Aim1, we conducted a 10-fold cross validation assessment for each item of background knowledge. Performance measures are *accuracy*, *recall* and *precision* defined in the appendix. The result is listed in Table 4 in which B3 is the most effective and B2 is the second-most effective. Case B2 indicates that information about qualitative state differences is very important for cognitive state classification due to the effectiveness of the qualitative reasoning approach. Case B3 indicates that information within a short time period is useful for discriminating important factors of cognitive states. As indicated in the previous section, such information is used to produce a first-order version of rules. This means that the driver's cognitive load depends on a time-series of action executions rather than a single action execution.

We then experimented with Aim2 using progressively incremental training data. In contrast to a 10-fold cross validation, we assign the saccade events in the set to 10 equal partitions in time order. In the first state, the first partition

Table 4. Performance measures of the 10-fold cross-validation for B1, B2 and B3.

Area	Data set	Accuracy	Precision	Recall
Shibuya	B1	75.5%	73.8%	75.2%
	B2	84.5%	85.1%	81.9%
	B3	85.9%	84.9%	85.7%
Noda	B1	72.5%	71.1%	73.5%
	B2	81.3%	81.6%	79.5%
	B3	84.2%	83.7%	83.8%

is a set of training examples, and the remaining partitions construct a set of test examples. The second stage takes the first two partitions as the training data set and the remaining eight partitions as the test data set. This process repeats nine times.

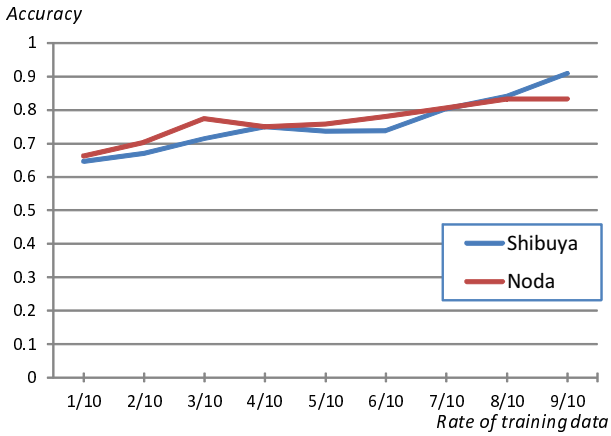


Fig. 4. Accuracies of incremental training data.

Figure 4 illustrates how accuracy changes with a series of incremental training data. At every stage, predictive accuracy exceeds the corresponding default accuracy. It exceeds 70% in the third stage and 80% in the eighth stage. This result is quite reasonable in accordance with the cross validation result.

The last aim is achieved by applying learned rules to another driving situation. Figure 5 presents the test results in which all driving data in another situation construct a set of test data. The test data set is equally divided into 10 partitions to capture the change of accuracy.

The accuracy of Shibuya's rule set is 78.3%, and that of Noda's rule set is 80.9% (average). These coincide with the accuracies for the third to the sixth stages in Fig. 4. However, variances among different partitions cannot be ignored due to the driving load. More detailed analysis on load information is needed.

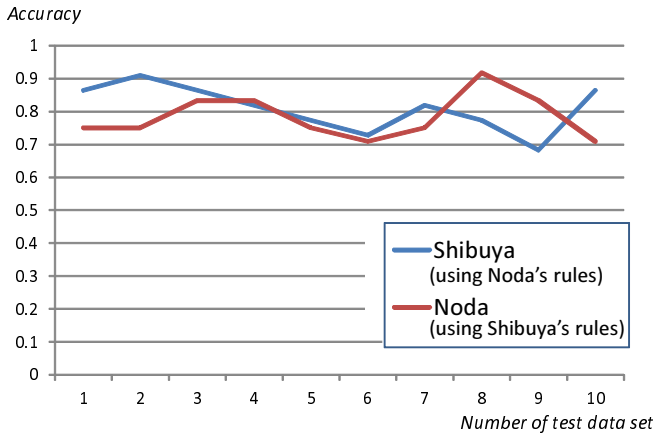


Fig. 5. Accuracies using a set of test data constructed from another driving situation.

9 Conclusions

This paper applied ILP to identify a driver's cognitive state using real driving data. We described how to acquire raw data such as eye-movement and driving data, arrange the raw data, and transform the data into qualitative data. We then constructed background knowledge and training examples, and produced rules classifying the driver's cognitive state. We also conducted realistic experiments to demonstrate the learning performance of this approach and obtained reasonable accuracy. The results indicate that ILP could capture the cognitive mental load based on measurable data, and we hope that ILP will be useful for designing user-interfaces for next-generation car navigation systems.

Future work includes investigating the applicability of learned rules to other drivers. We selected a skilled driver, but we have to consider driving data for inexperienced drivers as well. Furthermore, we should demonstrate how to clarify individual differences among drivers compared with other approaches of brain science.

References

1. Michael Bain and Claude Sammut: A framework for behavioural cloning, *Machine Intelligence 15*, pp. 103-129, 1998.
2. Ivan Bratko, Stephen Muggleton and Alen Varsek: Learning Qualitative Models of Dynamic Systems, *Inductive Logic Programming*, pp. 437-452, 1992.
3. Roger H. S. Carpenter: *Movements of the Eyes*, Pion Ltd. 1988.
4. John K. Chapin, Karen A. Moxon, Ronald S. Markowitz and Miguel A. L. Nicolelis: Real-time control of a robot arm using simultaneously recorded neurons in the motor cortex, *Nature Neuroscience 2(7)*, pp. 664-670, 1999
5. William J. Clancey: *Situated Cognition: On Human Knowledge and Computer Representations*, Cambridge University Press, 1997.

6. John Enderle, Susan M. Blanchard and Joseph Bronzino: *Introduction to Biomedical Engineering*, Academic Press, 2005.
7. John M. Findlay and Robin Walker: A model of saccade generation based on parallel processing and competitive inhibition, *Behavioral and Brain Science 22*, pp. 661-721, 1999.
8. John M. Findlay and Iain D. Gilchrist: *Active Vision*, Oxford University Press, 2003.
9. Benjamin Kuipers: *Qualitative Reasoning: Modeling and Simulation with Incomplete Knowledge*, The MIT Press, 1994.
10. Fumio Mizoguchi and Hayato Ohwada: Constrained Relative Least General Generalization for Inducing Constraint Logic Programs, *New Generation Computing 13*, pp. 335-368, 1995.
11. Jakob Nielsen and Kara Pernice: *Eyetracking Web Usability*, New Riders Press, 2009.
12. Donald A Norman and Daniel G Bobrow: On data-limited and resource-limited processes, *Cognitive Psychology 7(1)*, pp. 44-64, 1975.
13. Shinichiro Sega, Hirotoshi Iwasaki, Hironori Hiraishi and Fumio Mizoguchi: Qualitative Reasoning Approach to a Driver 's Cognitive Mental Load, *International Journal of Software Science and Computational Intelligence 3(4)*, pp. 18-32, 2011.
14. Dorian Suc, Daniel Vladusic and Ivan Bratko: Qualitatively faithful quantitative prediction, *Artificial Intelligence 158(2)*, pp. 189-214, 2004.

A Using Formula

Formula for accuracy, precision and recall is as follows (TP: true positives, FP: false positives, FN: false negatives, TN: true negative):

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN}$$