UNIVERSITÀ
DEGLI STUDI
FIRENZE

Scuola di Scienze Matematiche, Fisiche e Naturali
Corso di Laurea in Informatica

TITOLO IN ITALIANO

TITLE IN ENGLISH

TERROSI FRANCESCO

BONDAVALLI ANDREA
STRIGINI LORENZO

Anno Accademico 2018-2019

ABSTRACT

Abstract

INDICE

ELENCO DELLE TABELLE

ELENCO DELLE FIGURE

1

INTRODUZIONE

Sistemi informatici ormai ovunque (Cosa sono, esempi)

1.1 CYBER-PHYSICAL SYSTEMS OF SYSTEMS

1.2 DEPENDABILITY AND SAFETY

- Dependability
- Safety
- Considerazioni generiche sul perche‘ della tesi

2

AUTOMOTIVE - STATE OF ART

2.1 SELF-DRIVING CARS ARCHITECTURE

??????

Descrizione semplificata dell'architettura hardware (sensori) e software (AI controller, safety checker)

2.2 SAFETY NELL'AUTOMOTIVE

- Intro e standard

- Perche' le neural network sono un problema per la safety e perche' e' difficile validarla per questi sistemi | citazioni paperz (RAND study, koopmann, high-dependability systems. . .)

2.2.1 *Controller - Checker Problem*

======================

Grafico overlapping safety

======================

As the network learns, we expect the area covered by the Primary to grow. With a relatively simple Monitor, in relatively simple scenarios, there will potentially be no overlapping between the hazard areas covered by the two. In this phase, the safety gain provided by the use of a (correctly implemented) safety checker will be remarkable, since the Primary is still learning to handle "easy" demands. As pointed in the previous sections, our main goal is to observe and study the variation of the dependability provided by the monitor when the network is trained to handle "hard" demands, since there are no guarantees on the Monitor's performance in the long period.

As noted in [4] the probability of a failure for a *system* composed by a *Primary Component* and a *Safety-Monitor* on a random demand X is:

$$P_{fp}(1 - Coverage_{\sigma}) - covariance_{Q}(\theta(X), C_{\sigma}(\sigma, X)) \tag{2.1}$$

where:

- $P_{fp}(1 - Coverage_{\sigma})$ is the probability of a failure in the Primary Component (P_{fp}) that is **not detected** by the Safety Monitor (the term $1 - Coverage_{\sigma}$ is exactly the probability of having a false negative/positive)
- $covariance_{Q}(\theta(X), C_{\sigma}, X)$ given a demand profile $Q =< x, y >$ (i.e. the pair <demand, output), measures the correlation between:

$\theta(X)$ - The expected probability that the Primary will fail when processing demand X

$C_{\sigma}(\sigma, X)$ - the term identifying the *Coverage Factor* of the **Monitor**, on the specific demand X

This formula highlights the deep connection between the safety levels of the Controller and the Monitor, when it comes to the global safety of the system. It is clear from the equation that the probability of observing a failure in the system is also depending on the specific demand X, that's .

****

TO REVIEW:

****

The formula points the fact that to have the probability of observing a failure in the system depends not only on *all the possible demands* in the *demand space* but also on how the controller and the monitor react to them.
This ideas, and the complete lack of literature or studies on the topic, put the basis for our study:

- To prove the feasibility of a methodology to assess the safety level of systems composed by a *Primary Controller* and a *Safety Checker*

3

SYSTEM ANALYSIS METHOD

In this chapter is presented a method to study the safety level of an autonomous car over time, observing the emergence resulting from the interactions of a neural network controller and a safety monitor in a simulated environment.
A neural network was trained, tested and trained again several times with and without the safety monitor, to collect data about the emergence of these components.

3.1 EXPERIMENTAL ENVIRONMENT AND MEASURES

The goal of this work is to develop and to assess the feasability of an experimental method for the safety assessment of an autonomous vehicle. Due to the system being composed by two constituent systems: the Controller and the Checker, we think that a point of view based on the emergent behaviour resulting from the interaction of these systems can improve the quality of the assessment.
The main aspects we are interested in are:

- How the coverage of the system changes when the same monitor is applied to different stages of a network
- Changes in the safety gain provided by the same monitor when applied to different networks
- What features of the monitor determine an improvement (or worsening) to the safety of the system
- Possible behaviours of the neural network having an impact on the monitor usefulness

At the system level, we are interested in the probability of a safety-failure (e.g. a crash) and how to minimize it, or in the safety of the

system as a result of the training of a neural network, the Controller and a fault tolerance mechanism, the Monitor. As long as the network is trained properly, we expect that $P_{C_i}(\mathtt{failure}) \geqslant P_{C_{i+t}}(\mathtt{failure})$ where C_i represents a neural network controller trained for i epochs. At the same time we want our safety monitor to provide at least the same level of fault tolerance if the same network is trained again for t epochs.
The analysis must start with the definition of $n_{h=0...}$ safety cases, where n is the effective number of cases and h is difficulty of the initial conditions. C_i is tested in all the scenarios, starting with conditions somehow *"favorable"* to the system and then increasing the difficulty(e.g. increasing the traffic in the scenario and/or simulating a bad weather). This method allows to observe the *Time To Failure* of the Controller under different points of view:

- TTF_{i,j_h} as the time in which the controller fails at epoch i in the ${j_h}^{th}$ scenario
- $MTTF_{i,h}$, mean time to failure when the system performs in different level of difficulty
- $MTTF_{Controller_i} = \frac{1}{n}\sum_{k=0}^{n} TTF_{i,k}$

The controller is then trained again and re-tested in the same scenarios.

Since we are working in a simulated environment, the time to failure was computed using simulations steps, without loss of generality.

——————————————————————- FATTO CHE PROB INCIDENTE SOMMI A 1. COME LO ESPRIMO?

One of the main problem realated to assessing AVs' safety is the execution time. The probability of a failure increases monotonically over time, but the increasing factor should be lower the more the network is trained, so variation on the hardness of scenarios must be designed accurately. This property could result in very long simulations, but it also gives a useful hint for checking whether the system's safety is improving or not.

Once trained neural networks become essentially black boxes and even a small variation on the training parameters could result in totally different behaviours during test phase, therefore it can not be assumed that the same software (the monitor) will provide the same level of safety.
The monitor must be tested in the same scenarios in which the controller

was tested and should not intervene during the simulation, but at the rigth time needed to prevent the failure event if the controller failed the scenario. The simulations previously recorded are now repeated with the monitor activated and the following measures are extracted:

- True Positive Rate
  - Rate of successful failure avoidance where the controller failed.
- False Positive Rate
  - Rate of the events in which there's no failure but Monitor raises alarm. It is important that the fault-tolerance mechanism is not activated or the simulation will be compromised
- False Negative Rate
  - Rate of failures in the controller not detected by the monitor

Of course we desire the monitor's detections to be the most accurate possible. For this reason *Sensitivity*[1] and False Negative rate were chosen as measures of interest.

While in most of the cases a false positive will result in a state of degraded service, since a self-driving car is a safety-critical system performing in a dynamic environment, a false positive could put the system in an unsafe state (imagine performing an emergency brake for no reason on the highway), so False positive rate was taken into account as well.

However, these measures themselves aren't sufficient to detect changes in the interaction between the two CSs, so it's useful to record data such as the vehicle's speed and the controller's throttling/steering and to combine them with data recorded from the monitor to detect possible correlations between the behaviours. These values must be recomputed not only when the Monitor is improved (either by implementing a more sophisticated detection method or by improving the quality of sensors) but also when the network is trained because/due to... == GIUSTIFICARE QUI IL DISCORSO FAMOSO O RIMANDARE A CAPITOLO PRECEDENTE SU LETTERATURA? ==================================

When all the data are collected for each scenario

[1] True Positive rate: the proportion of safety measures applied by the monitor when actually needed

3.2 EXPERIMENTS METHODOLOGY

In this section we propose and describe the methodology developed in this work.

The study consists of several experiments in a simulated realistic environment, in which we observe how the coverage of the safety-monitor (i.e. the probability of raising an alert if there really is a safety-hazard) varies with respect to a neural network *in different stages of training*.
The first step to perform the analysis is to define what are the metrics of interest.

- Come vengono effettuati gli esperimenti (scenari? durata fissa? ad oltranza? fino ad un fallimento? . . .)

The efficacy of the constituent systems was studied separately at first. Since the goal of a vehicle is to go from point A to point B without crashing or hitting a pedestrian, the controller was tested at first, recording the the actions performed

4 METHOD IMPLEMENTATION

4.1 TOOLS AND SOFTWARE

4.1.1 *Carla Simulator*

In order to have a realistic environment, with accurate physics simulation and data sensors, the open-source simulator Carla was used. This simulator was developed with the purpose of offering an environment where AI agents can be trained to drive.

- CARLA
- Nervana Systems - coach (Intel)
- Reti neurali su git
- Point Cloud Library per filtrare i dati

4.2 ASSUMPTIONS AND LIMITATIONS?

Dedicare una sezione alle decisioni prese?

4.3 ARCHITETTURA DEL SOFTWARE (ESTRAPOLAZIONE DATI, INTERAZIONE RETE-MONITOR)

- Interazione rete-monitor
- Safety Monitor Implementation - obstacle detection
- Come vengono raccolti i dati
- Come vengono preprocessati

5

RISULTATI DELL'ANALISI

In questa sezione elenchiamo i dati che sono stati raccolti e quali sono i risultati che abbiamo ottenuto (errori ricorrenti, grafici, rapporto monitor-rete neurale)

6

CONCLUSIONS

BIBLIOGRAFIA

[1] Jelena Kocic, Nenad Jovicic, Vujo Drndarevi, *An End-To-End Deep Neural Network for Autonomous Driving Designed for Embedded Automotive Platforms (2019)*

[2] Qing Rao, Jelena Frtunikj, *Deep learning for self-driving cars: chances and challenges (2018)*

[3] Carlos Zednik, Otto-von-Guericke-Universitat Magdeburg *Solving the Black Box Problem: A Normative Framework for Explainable Artificial Intelligence*

[4] Peter Popov, Lorenzo Strigini *Assessing Asymmetric fault-tolerant Software* (Cited on page 12.)

[5] *https:waymo.com*

[6] *https:uber.com*