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

Tesi di Laurea

# ATTACCHI VERSO SISTEMI DI APPRENDIMENTO IN AMBITO AUTONOMOUS DRIVING: STUDIO E IMPLEMENTAZIONE IN AMBIENTI SIMULATI

# ADVERSARIAL ATTACKS TOWARDS MACHINE LEARNING SYSTEMS IN AUTONOMOUS DRIVING:STUDY AND IMPLEMENTATION IN A SIMULATED ENVIRONMENT

NICCOLÓ PIAZZESI

Relatore: *Andrea Ceccarelli*

Anno Accademico 2019-2020

# INDICE

# ELENCO DELLE FIGURE

# ELENCO DELLE TABELLE

*"Inserire citazione"*
*— Inserire autore citazione*

# INTRODUZIONE

Il mondo odierno è ormai pervaso dall'Intelligenza Artificiale. Uno dei settori di maggior interesse per la ricerca in questo ambito è indubbiamente quello delle cosidette *Self Driving Car*, ovvero le macchine a guida autonoma. Grandi aziende quali *Google* e *Tesla* hanno già sviluppato dei propri modelli(Figura 1 e 2) e l'interesse per questo tipo di veicoli è sempre in maggior crescita. Per poter funzionare correttamente questi veicoli acquisiscono informazioni dall'ambiente circostante sotto forma di immagini. Queste immagini vengono classificate da un sistema interno che, in base alle informazioni ricevute, decide l'azione da compiere(sterzare, accelerare, frenare ecc.). Anni ed anni di sviluppo e ricerca hanno reso sempre più affidabili e sicuri questi sistemi ma restano comunque presenti delle vulnerabilità. Una vulnerabilità molto importante sono i cosiddetti *Adversarial attacks*. Il meccanismo di questi attacchi è molto semplice:
alle immagini raccolte dal sistema viene applicata una modifica impercettibile a occhio umano ma in grado di causare un errore di classificazione che può portare, ad esempio, una macchina ad accelerare quando dovrebbe frenare. La ricerca su questi tipi di attacchi quindi è fondamentale per garantire l'affidabilità dei veicoli a guida autonoma. Lo scopo di questa tesi è lo studio l'implementazione di *Adversarial Attacks* contro modelli di guida autonoma in ambiente simulato. Il lavoro è cosi suddiviso:

- **Capitolo 1:** fondamenti teorici alla base dei sistemi intelligenti
- **Capitolo 2:** fondamenti di guida autonoma

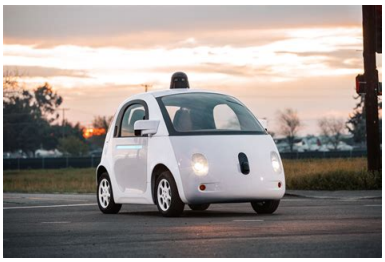

Figura 1: self driving car di Google

Figura 2: self driving car di Tesla

- **Capitolo 3:** Dependability e security
- **Capitolo 4:** strumenti utilizzati: in particolare vengono presentati *l'Adversarial Robustness Toolbox* e il simulatore *Carla*
- **Capitolo 5:** attacchi scelti per l'implementazione e motivazioni per le scelte effettuate
- **Capitolo 6:** Implementazione e risultati
- **Capitolo 7:** Conclusioni e possibili sviluppi futuri

# 1

# SISTEMI INTELLIGENTI

## 1.1 INTELLIGENZA ARTIFICIALE

Dare un'unica definizione di intelligenza artificiale risulta estremamente difficile a causa della vastità e dell'interdisciplinarietà dell'argomento. Soltanto nella figura 3 ne troviamo addirittura otto , tutte valide, ma forse la descrizione migliore per i nostri scopi è quella data dal padre di questa disciplina, John Mccarthy: " L'Intelligenza Artificiale è la scienza volta alla creazione di macchine *intelligenti*, più nello specifico di *programmi intelligenti*." [2].

### 1.1.1 *Agenti e ambienti*

Il concetto di macchina intelligente può essere ulteriormente astratto dall'idea di **agente computazionale intelligente**. Esaminiamo adesso questa definizone. Un **agente** è un'entità che compie azioni e percepisce informazioni da un ambiente.

Un'agente si comporta in modo **intelligente** quando:

| | |
|---|---|
| ``The exciting new effort to make computers think ... *machines with minds*, in the full and literal sense" (Haugeland, 1985)<br><br>``The automation of activities that we associate with human thinking, activities such as decision-making, problem solving, learning ..." (Bellman, 1978) | ``The study of mental faculties through the use of computational models" (Charniak and McDermott, 1985)<br><br>``The study of the computations that make it possible to perceive, reason, and act" (Winston, 1992) |
| ``The art of creating machines that perform functions that require intelligence when performed by people" (Kurzweil, 1990)<br><br>``The study of how to make computers do things at which, at the moment, people are better" (Rich and Knight, 1991) | ``A field of study that seeks to explain and emulate intelligent behavior in terms of computational processes" (Schalkoff, 1990)<br><br>``The branch of computer science that is concerned with the automation of intelligent behavior" (Luger and Stubblefield, 1993) |

Figura 3: Possibili definizioni di Intelligenza artificiale[1]

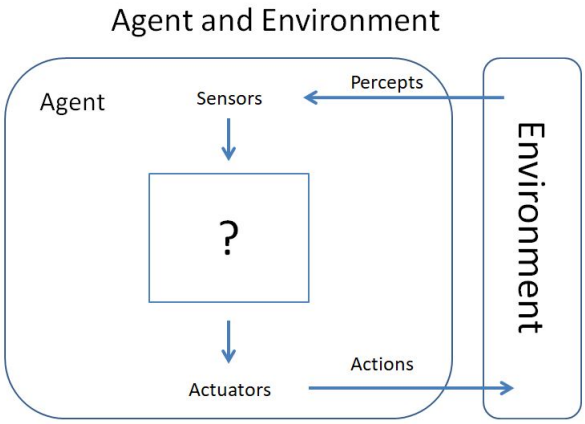


Figura 4: Agente computazionale[1]

- le sue azioni sono appropriate alle circostanze e ai suoi scopi
- È flessibile per ambienti e scopi dinamici
- impara dall'esperienza
- fa le scelte **giuste** date le sue limitazioni percettive e computazionali. Un agente tipicamente non può osservare l'ambiente direttamente; ha una memoria e un tempo per agire molto limitati.

Un agente **computazionale** è un agente le cui decisioni possono essere descritte in termini di una computazione. Questo significa che, una decisione può essere scomposta in una serie di operazioni primitive implementabili in un dispositivo fisico.[3].

### 1.1.2 *Intelligenza e performance*

Abbiamo detto che un agente è intelligente quando compie le *scelte giuste*. Ma cosa significa fare la scelta giusta? Rispondiamo a questa domanda in un modo molto banale: considerando le *consequenze* del comportamento di un agente. Un agente si muove all'interno di un ambiente e compie una sequenza di azioni in base alle informazioni che riceve. Queste azioni causano variazioni allo stato dell'ambiente. Se la variazioni è desiderabile, l'agente ha fatto le scelte giuste. La nozione di desiderabilità è catturata da una **misura di prestazione(performance measure)** che valuta una qualunque sequenza di azioni. Ovviamente non esiste una performance measure globale; solitamente viene co-

PEAS description of the task environment for an automated taxi

| Agent Type | Performance Measure | Environment | Actuators | Sensors |
|---|---|---|---|---|
| Taxi Driver | safe, fast, legal, comfortable, maximum profit, getting to correct destination | roads, other traffic pedestrians customers | steering, accelarator, brake, signal, horn, display | cameras, sonar, speedometer, GPS, odometer, accelerometer, engine sensors, keyboard |

Figura 5: descrizione PEAS dell'ambiente di lavoro di un taxi automatico

struita appositamente per uno specifico problema. Le specifiche dell'agente,dell'ambiente e della misura di prestazioni vengono raggruppate nell'**ambiente di lavoro(task environment)**, attraverso la descrizione PEAS(Performance,Environment,Actuators,Sensors).[1] La figura 5 riassume un esempio di descrizione PEAS dell'ambiente di lavoro di un taxi.

### 1.1.3 *Razionalità*

Ciò che abbiamo descritto fin'ora ci permette di verificare in modo quantitativo l'intelligenza di un agente. Espandiamo questa idea introducendo la nozione di **razionalità**. Cosa sia razionale in qualsiasi momento dipende da quattro elementi:

- la misura di prestazioni
- la conoscenza a priori dell'agente
- le azioni che l'agente può compiere

- la sequenza di percezioni fino a quel momento

Questo ci porta alla definizione di **Agente Razionale**:

*per ciascuna sequenza di percezioni, un agente razionale seleziona un'azione che massimizza il valore atteso della performance measure, data l'evidenza fornita dalla sequenza di percezioni e dalla conoscenza a priori dell'agente*

La razionalità non significa **onniscienza**. Un agente deve essere in grado di prendere decisioni anche quando non ha a disposizione tutte le informazioni necessarie a compiere l'azione perfetta. Un agente razionale massimizza *il valore atteso* della performance basandosi sulla sua conoscenza pregressa dell'ambiente, ma non sempre un ambiente è completamente osservabile. Per questo motivo sono fondamentali la fase di raccolta delle informazioni(**information gathering**) e di apprendimento(**learning**). Per poter migliorare le prestazioni, un agente deve essere in grado di raccogliere la massima quantità possibile di informazioni e di aggiungere tali informazioni alla propria base di conoscenza. Questo permette una scelta sempre migliore delle azioni da compiere.[1]

### 1.1.4 *Scopo dell'IA*

Lo scopo scientifico dell'IA è quello di studiare i principi e i meccanismi che rendono il comportamento intelligente possibile sia nei sistemi naturali che in quelli artificiali. Lo scopo ingegneristico di questa disciplina è la costruzione di dispositivi fisici in grado di comportarsi in modo intelligente. Da queste basi si sono sviluppati un innumerevole quantità di branche, tra cui le più importanti sono: sono:[2]

- **logical AI**
- **search**
- **knowledge and reasoning**
- **planning**
- **learning from experience**
- **genetic programming**

In figura 6 vengono riportate alcune delle applicazioni di maggior successo dei *Sistemi Intelligenti*. Il settore delle Self driving Car si basa in particolare su **Machine Learning** applicato alla **Computer Vision**.

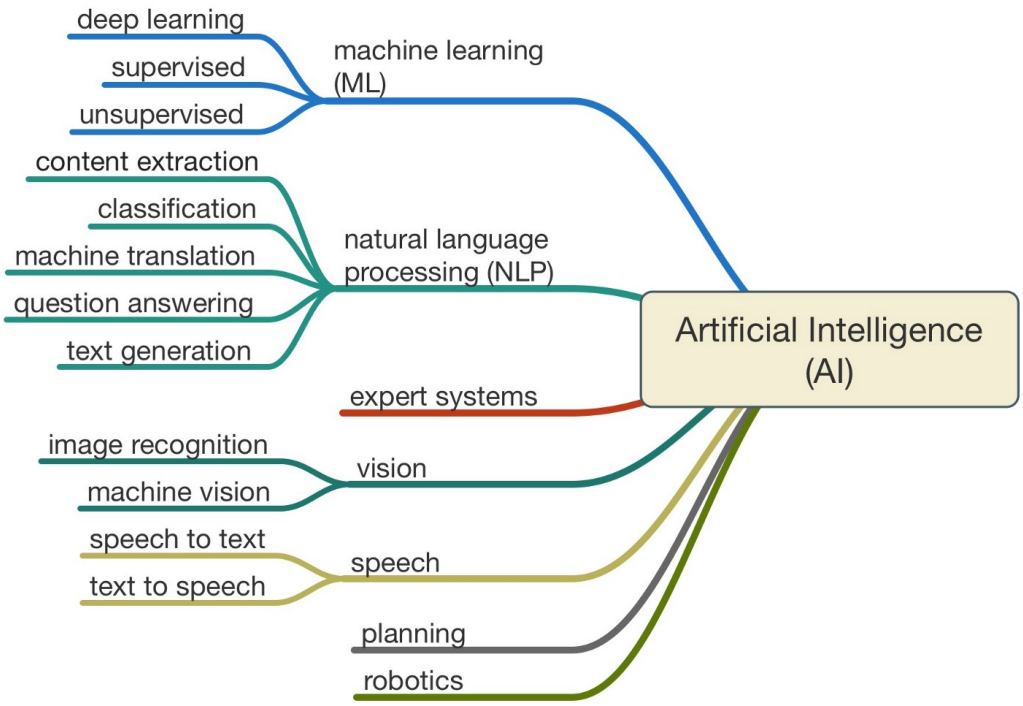


Figura 6: applicazioni dell'IA[4]

## 1.2 MACHINE LEARNING

Il **Machine Learning(ML)** è una branca dell'intelligenza artificiale dedicata allo studio e allo sviluppo di algoritmi che migliorano le prestazioni attraverso l'esperienza[5]. L'abilità di un algoritmo di migliorare dall'esperienza automaticamente è fondamentale, perchè è impossibile scrivere programmi che sappiano a priori tutte le possibili situazioni in cui un agente potrebbe ritrovarsi. Inoltre l'ambiente in cui l'agente è inserito può mutare nel tempo[1]. Consideriamo il caso dei mercati finanziari. Un programma creato per prevedere i prezzi delle azioni di domani deve essere in grado di adattarsi alle variazioni improvvise causate ad esempio da una crisi globale.

### 1.2.1 *Problemi di apprendimento ben definiti*

Iniziamo ad approfondire i concetti di apprendimento automatico considerando alcuni task di apprendimento. Più precisamente:

**Definizione 1.1.** Si dice che un programma **impara** dall'esperienza E rispetto a una classe di tasks T e una misura di prestazioni P, se la sua

prestazione nei tasks in T, misurata da P, migliora con l'esperienza E

Dalla definizione 1.1 possiamo specificare diversi problemi di apprendimento, ad esempio[6]:

- **Task T**: giocare a scacchi
- **Misura di prestazioni P**: percentuale di partite vinte contro gli avversari
- **Esperienza E**: giocare partite di prova contro se stesso

Ogni problema di learning consiste nel prendere la conoscenza a priori e i dati ricevuti(l'esperienza E) e trasformarli in una rappresentazione interna utilizzata da un agente per prendere decisioni. La rappresentazione può corrispondere con i dati stessi ricevuti ma di solito è una sintesi compatta e significativa. In definitiva, per specificare una determinata tecnica di apprendimento è quindi necessario affrontare le seguenti[3] questioni:

- **Task** Un task di apprendimento è una qualsiasi attività che può essere appresa da un agente
- **Feedback** Durante l'apprendimento a un agente viene fornito un riscontro in base alla correttezza delle azioni svolte. Il riscontro può essere un premio o una punizione. In base ad esso un agente modifica le proprie azioni migliorando coì la propria esecuzione su un determinato task.
- **Rappresentazione** Come detto in precedenza l'esperienza deve influenzare la rappresentazione interna di un agente. Gran parte del Machine Learning è focalizzato nel contesto di una specifica rappresentazione(es. reti neurali)
- **Online e Offline** Nel learning offline, tutti i training examples sono disponibili prima dell'azione di un agente. Nel learning online, gli esempi vengono ricevuti durante l'esecuzione.
- **Misura di Successo** Per sapere se un agente ha effettivamente imparato, è necessaria una misura di successo. La misura NON riguarda le prestazioni sui dati di addestramento, bensì le prestazioni su nuove esperienze.
- **Bias** Con il termine *bias* si intende la tendenza a preferire un'ipotesi rispetto a un'altra, concetto fondamentale nel processo di scelta

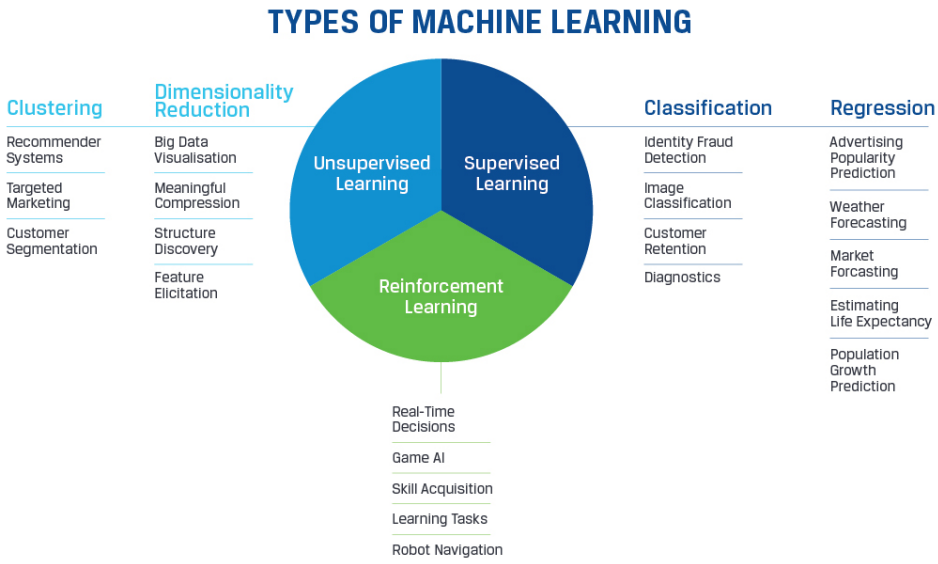


Figura 7: tre maggiori tipi di learning[7]

- **Noise** Una delle proprietà più importanti per un algoritmo di apprendimento è la capacità di gestione di dati condizionati. Infatti nella pratica i dati possono spesso risultare imperfetti o in alcuni casi incompleti .
- **Interpolazione e Estrapolazione** L'interpolazione riguarda le previsioni tra i casi per i quali si possiedono dei dati(ad esempio approssimare una funzione dati alcuni valori di essa), l'estrapolazione comporta invece previsioni che vanno oltre i dati conosciuti.

### 1.2.2 *Forme di Learning*

Gli algoritmi di Machine Learning sono suddivisi in diverse classi, a seconda del risultato desiderato. In figura 7 vengono mostrate le tre maggiori classi. Nel contesto di questa tesi, siamo interessati al **Supervised Learning**, in particolare ad algoritmi di **Deep Supervised Learning** basati su **Reti Neurali Convoluzionali**.

### 1.2.3 *Deep Supervised Learning*

La forma più comune di Machine Learning è il supervised learning. Consideriamo il caso in cui si voglia addestrare un classificatore che

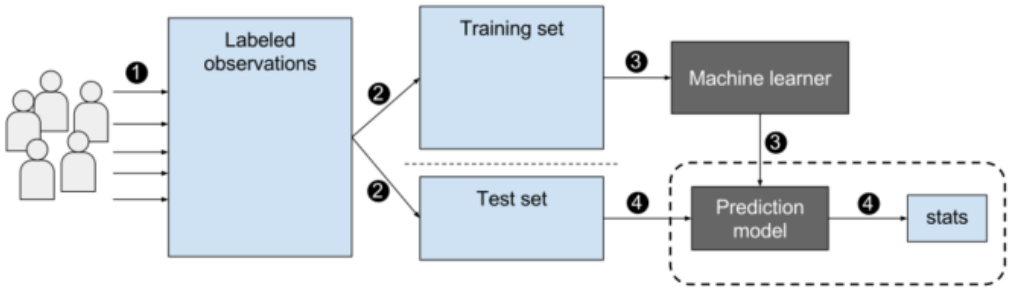


Figura 8: Supervised Learning[8]

sappia riconoscere i diversi tipi di segnali stradali. Si colleziona un grosso dataset composto da immagini di segnali stradali, ciascuno di esso con la rispettiva categoria(stop, limite di velocità divieto di sosta ecc.). Queste immagini compongono il **training set**. Durante l'apprendimento, vengono mostrate alla macchina le immagini del training set. Per ciascuna immagine l'algoritmo produce un output della forma di un vettore di punteggi, uno per ogni categoria. Si calcola una funzione che misura l'errore(**loss function**) fra l'output prodotto e l'output desiderato. In base all'errore commesso l'algoritmo modifica dei paramentri interni, detti pesi, in modo da ridurlo. Per aggiustare il vettore dei pesi in modo corretto , l'algoritmo di apprendimento utilizza una tecnica detta *Gradient Descent*. Ad ogni errore commesso, il vettore dei pesi viene aggiustato nella direzione di massima decrescita dell'errore, fino a raggiungere un minimo locale. La fase di addestramento termina quando si raggiunge il minimo errore possibile, calcolato come la media della funzione di errore su ogni immagine del dataset. Terminato il training, la prestazione del classificatore viene misurata su un differente dataset di immagini, detto **test set**. Questo serve a verificare la capacità di generalizzazione della macchina, andando a controllare la correttezza degli output su input sconosciuti.

Una grossa fetta delle applicazioni pratiche utilizza classificatori lineari su caratteristiche modellate a mano. Un classificatore binario calcola una somma pesata delle componenti del cosidetto **feature vector**. Se la somma è sopra una certa soglia, l'input è classificato in una certa classe. I classificatori lineari suddividono lo spazio degli input in semplici sottoregioni dette iperpiani, ma contesti come il riconoscimento di immagini o il riconoscimento vocale hanno bisogno di modelli più complessi, capaci di rilevare minuscole variazioni su caratteristiche importanti(**selettività**) e di ignorare anche grosse variazioni su caratteristiche irrilevanti al con-

testo, come ad esempio una rotazione dell'immagine(**invarianza**). La scelta delle caratteristiche rilevanti è quindi una fase fondamentale del processo di learning. Tradizionalmente, la scelta delle features veniva costruita direttamente dagli sviluppatori, il che comportava una grande richiesta di abilità ingegneristica e di competenza nel dominio di lavoro. Grazie al Deep Learning il processo di estrazioni di features rilevanti viene automatizzato con una procedura di apprendimento a scopo generale. Un'architettura di deep learning è formata da uno stack multilivello di moduli elementari, tutti(o la maggior parte) sottoposti a learning, e la maggior parte dei quali calcola una mappatura non lineare tra input-output. Ciascun livello aumenta sia la selettività che l'invarianza della rappresentazione. Maggiore è la profondità dello stack, maggiore è la complessità della funzione di scelta, capace di rilevare i più piccoli cambiamenti a dettagli rilevanti[9].

### 1.2.4 *Deep FeedForward Neural Networks*

Le reti neurali feedforward sono uno dei modelli computazionali di maggior successo nell'ambito del deep learning. Lo sviluppo in questo campo è stato influenzato dallo studio sulla struttura del cervello umano e sulle modalità di trasmissioni di informazioni tra neuroni. L'obbiettivo di una rete feedforward è l'approssimazione di una funzione $f*$ che, nel caso di un classificatore mappa un vettore di feature **x** a una categoria **y**. Una rete feedforward definisce una funzione $y = f(x;\theta)$ e apprende il valore di $\theta$ che permette la miglior approssimazione possibile. Questo tipo di reti viene detto **feedforward** perchè i dati passano dall'input x, alle computazioni che definiscono f, e infine all'output y. Non vi sono cicli nei quali risultati di calcoli interni sono reinseriti all'interno della rete. Quando le reti includono anche cicli vengono dette **reti ricorrenti**. Uno schema di una rete feedforward è presentato in figura 9. I livelli intermedi sono detti **hidden(nascosti)** perchè nel **training set** non vengono esplicitati gli output desiderati per ciascuno di questi livelli. Ogni livello nascosto rappresenta una funzione e la funzione totale è data dalla composizione di queste funzioni. Per esempio, se abbiamo 3 livelli la funzione completa sarà $f(x) = f^3(f^2(f^1(x)))$. Il singolo livello è composto da molte **unità** che agiscono in parallelo, ciascuna rappresentante una funzione da vettore a scalare[10].

La generica i-esima unità riceve un vettore di input dalle unità del livello precedente, indicati con $X_j$. $X_j$ viene trasmesso all'unità opportu-

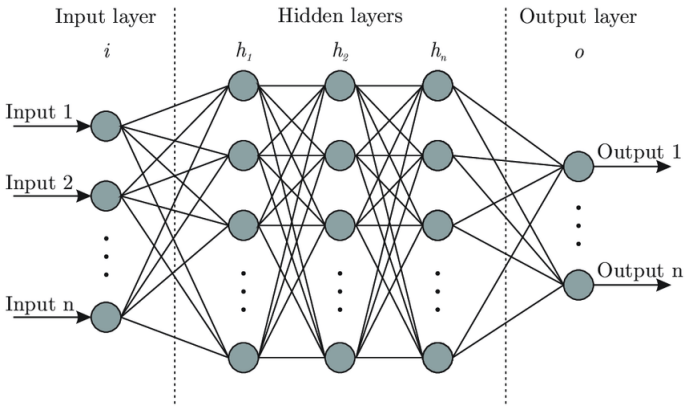


Figura 9: schema base di una rete neurale[11]

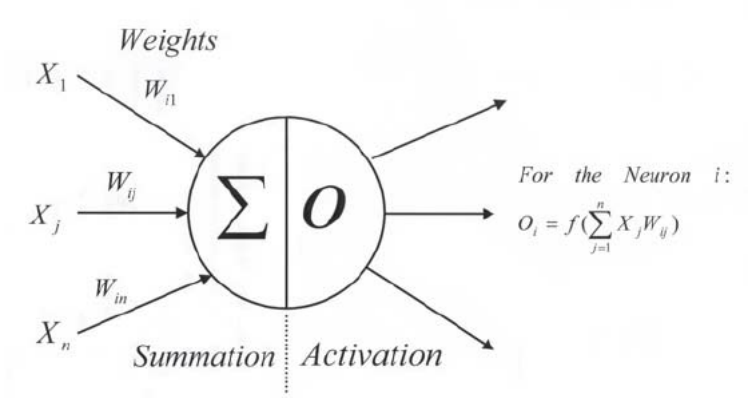


Figura 10: un "neurone" artificiale [13]

namente moltiplicato da un peso $W_{ij}$. Gli input ricevuti dall'i-esima unità costituiscono lo stato di attivazione, rappresentato dalla sommatoria

$$\sum_{j} W_{ij} X_j$$

. L'output prodotto dall'unità viene calcolato attraverso la funzione f, detta funzione di attivazione. Il compito della funzione di attivazione è quello di controllare se lo stato di attivazione supera una certa soglia. Se questo avviene l'unità trasmette alle unità del livello successivo[12].

### 1.2.5 *Learning nei neural network*

Il learning di una rete neurale avviene solitamente(nel caso dell'object recognition) in modalità supervisionata. Si vuole determinare un vettore

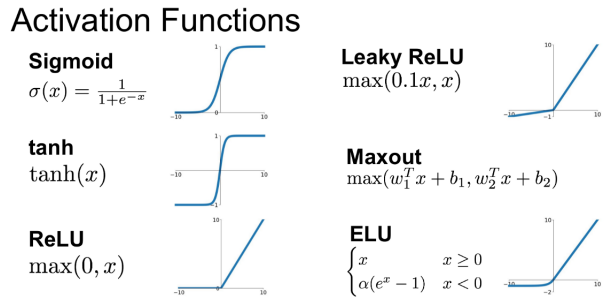


Figura 11: Alcune funzioni di attivazione[14]

dei pesi $\vec{w}$ che permetta alla rete di produrre l'output corretto per ciasun esempio nel training set. L'idea di base è quella di usare il gradient descent ma in una rete multilivello il problema è ulteriormente complicato dalla presenza degli **hidden layers**. Mentre nell'output layer l'errore che si commette in ciascuna fase di addestramento è esplicito, l'errore presente nei livelli nascosti non è immediato da calcolare, in quanto nel training set non sono presenti i valori che i nodi nascosti devono assumere. Il problema viene risolto trasmettendo a ritroso l'errore commesso dall'output layer ai livelli nascosti, con un algoritmo detto di **back-propagation**.

**procedure** TRAIN
  $X \leftarrow$ Training Data Set of size mxn
  $y \leftarrow$ Labels for records in X
  $w \leftarrow$ The weights for respective layers
  $l \leftarrow$ The number of layers in the neural network, 1...L
  $D_{ij}^{(l)} \leftarrow$ The error for all l,i,j
  $t_{ij}^{(l)} \leftarrow 0$. For all l,i,j
  *For* $i = 1$ to $m$
    $a^l \leftarrow feedforward(x^{(i)}, w)$
    $d^l \leftarrow a(L) - y(i)$
    $t_{ij}^{(l)} \leftarrow t_{ij}^{(l)} + a_j^{(l)} \cdot t_i^{l+1}$
  **if** $j \neq 0$ **then**
    $D_{ij}^{(l)} \leftarrow \frac{1}{m} t_{ij}^{(l)} + \lambda w_{ij}^{(l)}$
  **else**
    $D_{ij}^{(l)} \leftarrow \frac{1}{m} t_{ij}^{(l)}$
    where $\frac{\partial}{\partial w_{ij}^{(l)}} J(w) = D_{ij}^{(l)}$

Figura 12: pseudocodice della back-propagation[15]

### 1.2.6 *Reti Convoluzionali*

Nel contesto dell'object-detection le reti più utilizzate sono le reti convoluzionali. Le reti convoluzionali sono progettate per processare dati

in forma matriciale, come immagini e audio. La loro architettura è strutturata in una serie di fasi. Le prime fasi sono composte da due tipi di layer: **convolutional layers** e **pooling layers.** Le unità in un layer convoluzionale sono organizzate in feature maps,addette a riconoscere alcune caratteristiche di un'immagine. Ciascuna unità è collegata a un sottoinsieme di unità delle feature maps nei precedenti livelli attraverso un gruppo di pesi detto **filter bank** e utilizza una funzione di attivazione non lineare come una ReLU. Tutte le unità di una feature map condividono la stessa filter bank, e diverse feature maps usano diverse filter bank. L'aggregazione di unità in feature maps è motivata dalla struttura dell'input. Nei dati in forma matriciale, gruppi di valori locali sono spesso fortemente correlati, formando pattern locali che vengono facilmente riconosciuti. La condivisione di pesi tra unità diverse è spiegata notando come un pattern possa apparire in qualsiasi parte dell'immagine, e l'utilizzo di una filter bank garantisce invarianza alla località. Passando ai pooling layer essi hanno il compito di "fondere" features semanticamente simili,tipicamente calcolando il massimo tra un gruppo di unità di una feature map. Questo causa una riduzione nelle dimensioni dell'input che viene poi passato ai layer successivi, La riduzione delle dimensione è fondamentale per garantire efficienza computazionale e per aumentare l'invarianza alle piccole perturbazioni. Due o più fasi di convoluzione e pooling sono sono seguite da altri layer convoluzionali e completamente connessi. Le filter banks vengono addestrate con algoritmi di backpropagation.[9]

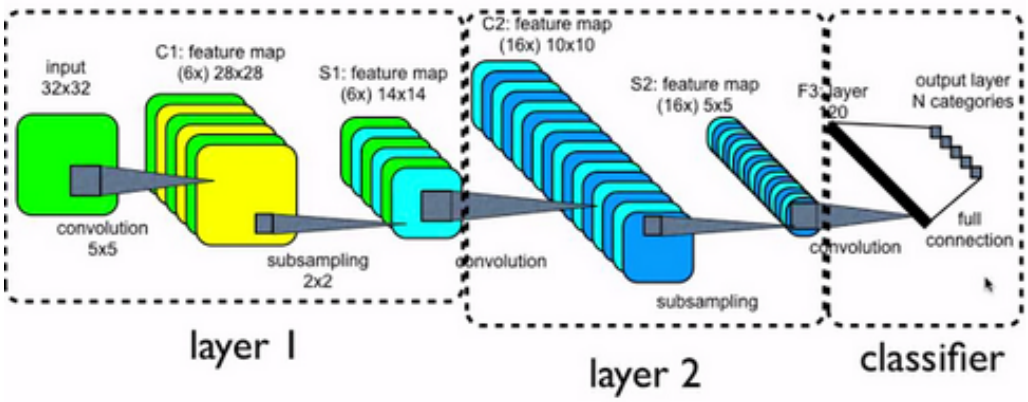


Figura 13: rete neurale convoluzionale

### 1.2.7 *Adversarial Examples*

Nonostante gli ottimi risultati raggiunti, le reti neurali sono state dimostrate essere vulnerabili ai cosiddetti *adversarial examples*. GLi adversarial examples sono input intenzionalmente modificati che risultano essere molto simili agli input originali per gli esseri umani, ma che causano decisioni scorrette da parte dei modelli. La vulnerabilità delle reti a questo tipo di attacchi rappresenta un grande rischio di sicurezza nelle applicazioni pratiche come i veicoli a guida autonoma. Studiarli è quindi necessario per identificare le limitazioni degli algoritmi di learning attuali, dando così spunti per migliorare la robustezza dei modelli.

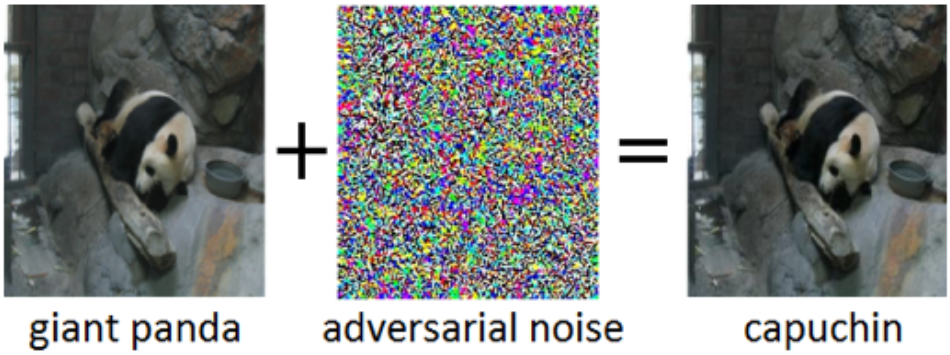


Figura 14: Adversarial example

# 2

# FONDAMENTI DI GUIDA AUTONOMA

Lo sviluppo tecnologico nel campo dell'intelligenza artificiale ha permesso una grande evoluzione nel campo della guida autonoma. L'interesse per questo settore è guidato dai seguenti vantaggi che questi veicoli possono potenzialmente portare[16]:

- **Riduzione degli incidenti** Una grossa parte degli incidenti stradali(fino al 90% )sono dovuti a errori umani. Eliminarli significherebbe ridurre notevolmente il numero di infortuni e salvare delle vite.
- **Riduzione del traffico** Grazie al controllo automatico si possono evitare le situazioni di congestione che avvengono naturalmente sulle strade. Oltre a migliorare il deflusso veicolare, questo porterebbe anche una riduzione delle emissioni di $CO_2$ e del consumo di benzina, causate dalle macchine ferme nel traffico.
- **Migliore capacità stradale** L'ottimizzazione del trafico potrebbe portare un notevole incremento nella capacità autostradale. La capacità di monitorare in modo costante l'ambiente circostante e reagire istantaneamente renderebbe più sicuro viaggiare a velocità maggiori e con meno spazio tra ciascun veicolo.
- **Accessibilità** Attualmente molte persone anziane e/o con disabilità sono escluse dal viaggio in automobile in solitaria. Le self-driving car rappresentano un'opportunità di indipendenza per queste categorie, che potrebbero così muoversi senza dover ricorrere a soluzioni esterne.

Attualmente però non si puo parlare ancora di guida completamente autonoma, ma di sistemi di guida assistita. Restano ancora molte problematiche aperte, soprattutto da un punto di vista legale[18] etico[19]. Per questo, e per i problemi di emphdependability che verrano illustrati nei prossimi paragrafi, lo sviluppo di veicoli autonomi è in molti casi

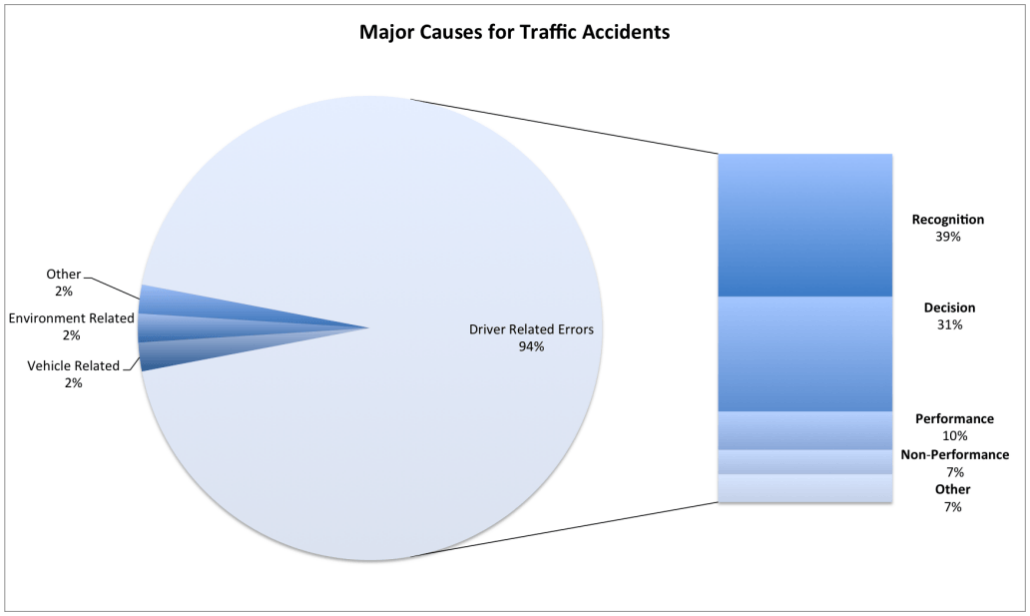


Figura 15: Maggiori cause di incidenti stradali ad Austin tra il 2005 e il 2007[17]

ancora in fase prototipale. La ricerca è quindi fondamentale per portare a miglioramenti significativi.

## 2.1 LIVELLI DI AUTOMAZIONE

Nel 2014 la **SAE international** ha pubblicato lo standard J3016, col quale vengono definiti sei liveli di autonomia, in base a quanto un guidatore deve intervenire.

*Livello 0*

Nessun tipo di automazione. Il controllo è completamente affidato al conducente.

*Livello 1: Assistenza alla guida*

Il conducente si occupa di tutti gli aspetti della guida. Viene supportato da sistemi elettronici che possono segnalare la presenza di pericoli attraverso segnali visivi e acustici. Vengono classificati di livello 1 anche i veicoli con una singola funzione di assistenza automatizzata(ad esempio il controllo automatico della velocità).

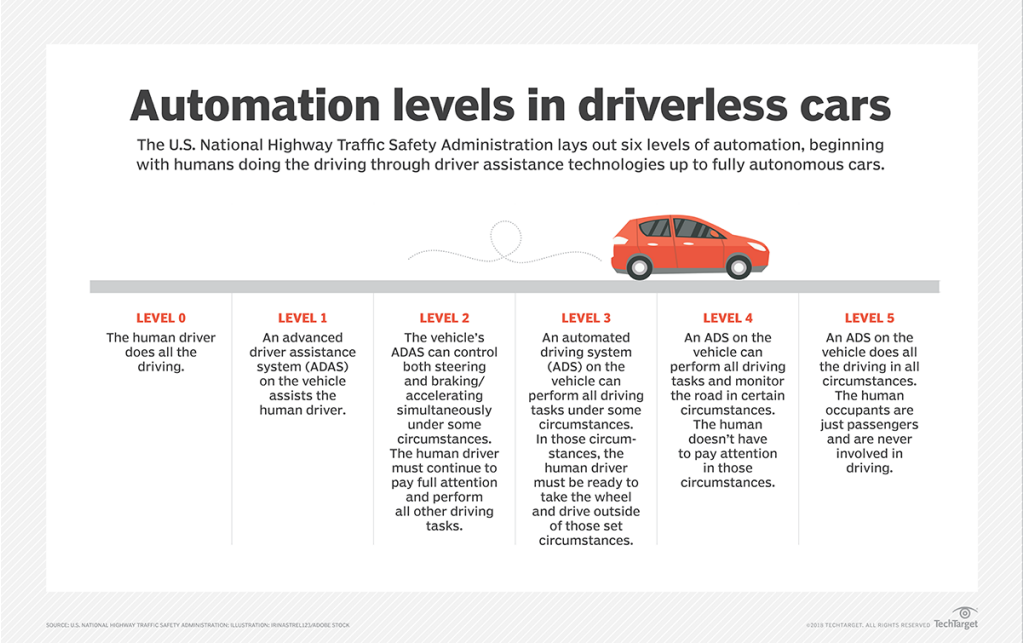


Figura 16: livelli di automazione[20]

*Livello 2: Automazione Parziale*

Il conducente deve mantenere il controllo del veicolo in ogni istante ma le le varie funzioni vengono automatizzate da due o più sistemi avanzati di assistenza alla guida(**sistemi ADAS**). Questi sistemi includono:

- Adaptive Cruise Control- controllo automatico della velocità
- Lane-Keeping Assist - controllo dello sterzo per impedire l'uscita dalla corsia
- Automatic Emergency Braking - controllo del freno per le situazioni di emergenza

*Livello 3: Automazione condizionale*

La guida è completamente automatizzata ma non in tutti i momenti del viaggio. Il conducente viene avvisato quando deve riprendere il controllo ed è tenuto a rispondere in modo appropriato. Se questo non avviene, il sistema arresta il veicolo nel modo più sicuro possibile.

*Livello 4: Alta Automazione*

Il viaggio è completamente gestito dal sistema. Il conducente non è tenuto a intervenire in nessun momento. La completa autonomia è garantita però a patto che non esistano limitazioni come ad esempio condizioni metereologiche fortemente avverse.

*Livello 5: Automazione totale*

Sistema completamente autonomo. Non vi è alcuna limitazione ed è tutto gestito dal sistema interno in qualunque momento.

### 2.1.1 *La situazione attuale*

Al momento in commercio esistono veicoli fino al secondo livello(Tesla AutoPilot, Cadillac Super Cruise, Volvo Pilot Assist ecc.). I veicoli di livello 3 e 4 sono ancora in fase prototipale ma aziende come Honda e Audi promettono di avere modelli in commercio già dal 2021[21]. Il livello 5 sembra ancora lontano, ma il ritmo di sviluppo attuale fa ben sperare per i prossimi anni.

## 2.2 VISIONE ARTIFICIALE

La visione dell'ambiente circostante è la questione focale nello sviluppo di un veicolo a guida autonoma. Per poter "vedere", una macchina utilizza diversi sensori:

- Camere
- Radar
- Lidar

### 2.2.1 *Camere*

Le videocamere utilizzate sono profondamente diverse dalle normali videocamere in vendita. Sono progettate in modo da avere un campo visivo molto ampio, migliore sensibilità anche a bassa luminosità, bassa risoluzione(migliori performance) e per funzionare ad alte e basse temperature. Restano però limitate in situazioni di bassa visibilità.


Figura 17: i sensori più utilizzati[22]

### 2.2.2 *Radar*

Il compito dei radar è principalmente quello di rilevare ostacoli e situazioni di pericolo imminente. In base alle informazioni raccolte l'auto accelera o frena. I radar si rilevano utili anche nella fase di parcheggio.

### 2.2.3 *Lidar*

I lidar sono dispositivi il cui funzionamento è simile a quello dei radar. La differenza principale sta nel segnale utilizzato: i radar usano onde radio mentre i lidar usano impulsi luminosi. I lidar sono particolarmente utili per la rilevazione di pedoni e per riconoscere i segnali stradali.

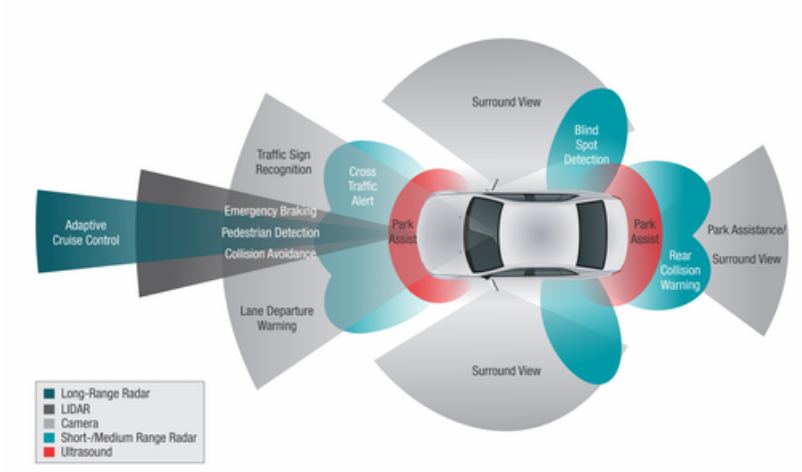


Figura 18: sensori in una macchina autonoma[23]

### 2.2.4 *Dai sensori agli attuatori*

Tutti i sensori sono fondamentali perchè ciascuno di esso compensa i punti deboli dell'altro. Una videocamera non funziona bene in condizioni di nebbia intensa, ma un radar sì. Un radar però non riconosce se un semaforo è verde o rosso, mentre una camera sì. Nella figura 18 sono mostrate le funzioni svolte da ciascun sensore.

Le informazioni raccolte dai sensori vengono passate al "livello" sottostante, nel quale il sistema interno sceglie la prossima azione da compiere. Il modello decisionale più utilizzato è una rete neurale convoluzionale. Il deep learning si è dimostrato estremamente potente in quest'ambito, fornendo ottimi risultati per l'object recognition. L'azione scelta viene passata agli attuatori(freno, sterzo, acceleratore) che provvedono ad eseguirla.

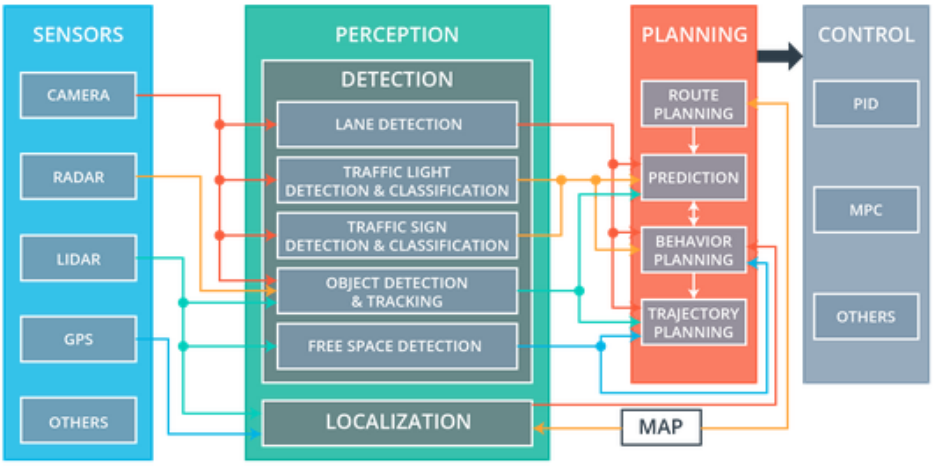


Figura 19: Il processo decisionale[24]

# 3

# DEPENDABILITY E SECURITY

[25] I veicoli a guida autonoma appartengono alla categoria dei sistemi critici. Un sistema critico è un sistema il cui malfunzionamento può provocare danni considerati inacettabili. Questi includono danni a oggetti di valore, danni ambientali e nei casi piu gravi, il ferimento o addirittura la morte delle persone. Per garantire che un tale sistema operi nel modo più sicuro possibile è necessario analizzare tutti i fattori che possono portare a un fallimento irreversibile.

La textbfdependability è la capacità di un sistema di fornire un servizio sul quale è possibile fare affidamento in modo giustificato. Essa viene suddivisa in 3 categorie:

- Attributi
- Minacce
- Mezzi di Raggiungimento

## 3.1 ATTRIBUTI

La dependability comprende i seguenti attributi:

- **availability**: disponibilità del servizio corretto
- **reliability**: stabilità del servizio corretto
- **safety**: assenza di conseguenze catastrofiche sullìutente e sull'ambiente
- **integrity**: assenza di alterazioni improprie al sistema
- **maintainability**: capacità di subire modifiche e riparazioni

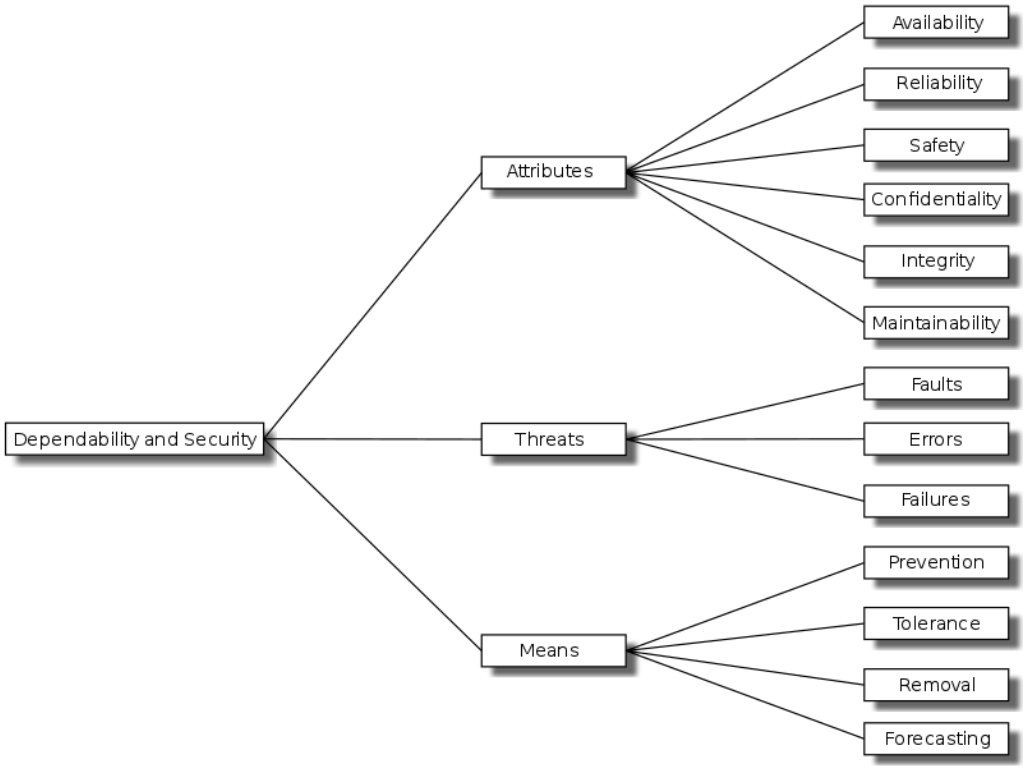


Figura 20: tassonomia della dependability[26]

Quando si considera la **security** è necessario specificare un ulteriore attributo: la confidentiality, ovvero la mancata divulgazione di informazioni non autorizzata. La security è composta da confidentiality, integrity e availability.

## 3.2 MEZZI DI RAGGIUNGIMENTO

Nel corso degli anni si sono sviluppate molte metodologie per raggiungere dependability e security. Tali metodologie possono essere raggruppate in quattro macrocategorie:

- **fault prevention**: mezzi per prevenire l'occorrenza e l'introduzione di guasti
- **fault tolerance**: mezzi per evitare fallimenti di servizio quando sono presenti dei guasti
- **fault removal**: mezzi per ridurre numero e gravità dei guasti

- **fault forecasting**: mezzi per stimare numero, incidenza futura e probabili conseguenze di guasti

Fault prevention e fault tolerance portano al conseguimento della dependability mentre fault removal e fault forecasting sono i mezzi di validazione

## 3.3 MINACCE

Le maggiori minacce per la dependability sono i **guasti**(faults in inglese). I guasti hanno varie cause e causano gli **errori**. Un errore può causarne altri fino a propagarsi al di fuori dei confini del sistema. Quando ciò avviene, si verifica un **fallimento**. Un fallimento è la situazione in cui il servizio fornito è diverso dal servizio corretto.

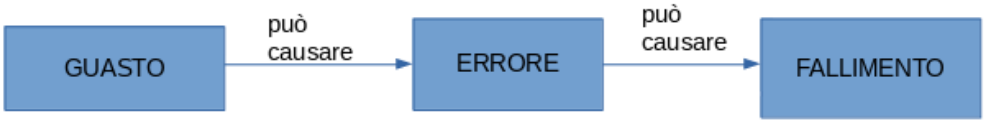


Figura 21: Catena guasto-errore-fallimento

### 3.3.1 *Tipi di guasti*

I guasti vengono classificati secondo 8 parametri, i quali creano le classi di guasto elementari, mostrate in figura 22 I diversi tipi di guasto sono raggrupati in tre categorie:

- **Development faults**: i guasti che avvengono in fase di sviluppo
- **Physical faults**: i guasti che riguardano l'hardware
- **Interaction faults**: tutti i guasti causati dall'interazione con l'ambiente esterno

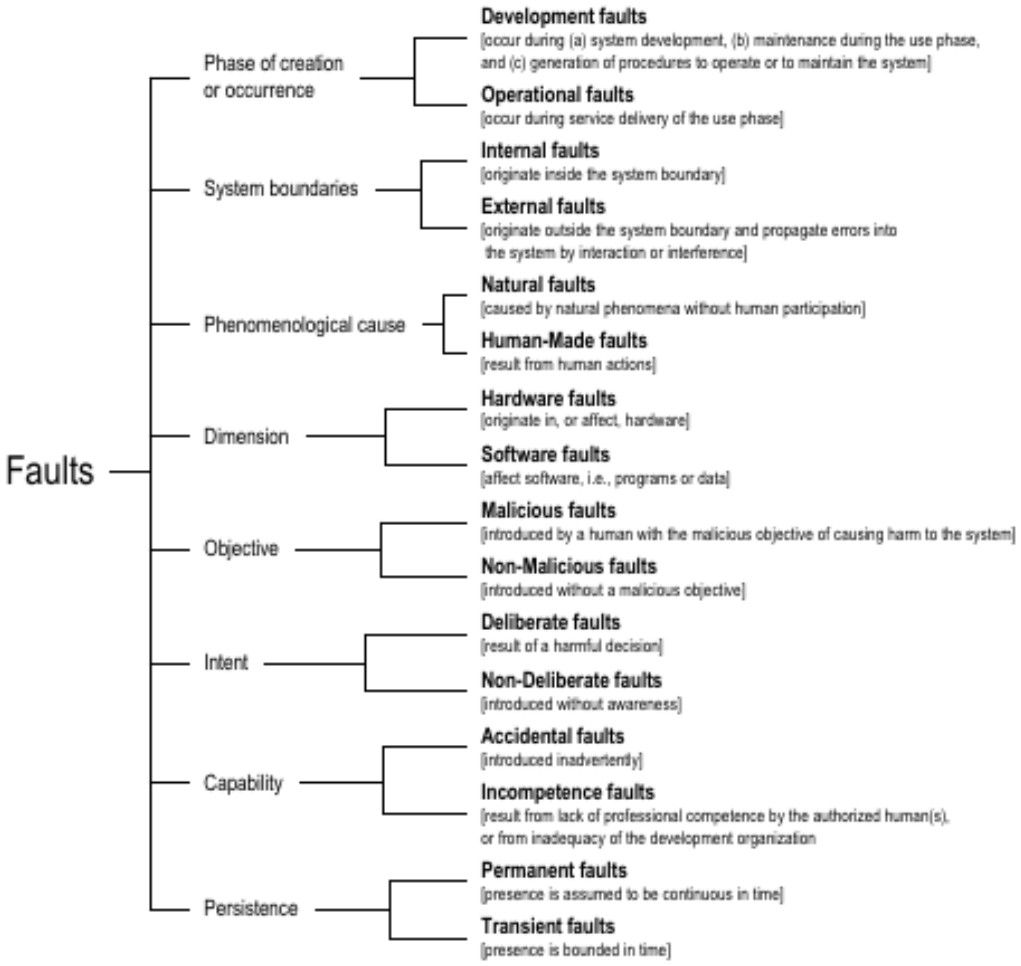


Figura 22: tassonomia dei guasti[25]

*Guasti causati dall'azione umana*

I guasti sul quale ci concentriamo sono quelli di origine umana volontari. Questi guasti possono essere causati da una mancanza di azioni ove necessarie(omission faults), oppure da azioni che si rilevano essere sbagliate(commision faults). Questi guasti possono essere distinti in due tipi, sulla base dell'*obbiettivo* dello sviluppatore o dell'essere umano che lo causa:

- guasti involontari, causati accidentalmente senza lo scopo di arrecare danno
- guasti maligni, causati in modo volontario per arrecare danni al sistema durante l'uso

I guasti maligni hanno come scopo la negazione del servizio(denial of service), l'accesso a informazioni confidenziali o la modifica impropria di un sistema. Vengono raggruppati in due classi:

- **guasti a livello logico**: comprendono virus, worms, bombe logiche ecc.
- **tentativi di intrusione**, anche usando mezzi fisici

In entrambi i casi si sfrutta una vulnerabilità di un sistema per ottenerne il pieno controllo(exploit). La vulnerabilità solitamente è a livello software ed è causata dall'azione involontaria degli sviluppatori.

# 4

# L'ADVERSARIAL ROBUSTNESS TOOLBOX

L'Adversarial Robustness ToolBox (ART) è una libreria python che permette lo sviluppo di difese per i modelli ad apprendimento automatico, rendendoli più sicuri ed affidabili. Questi modelli sono infatti vulnerabili ai cosidetti "esempi antagonisti": dati in input(immagini, testo ecc.) creati specificatamente per produrre una determinata risposta dal modello. L'ART include questi attacchi e fornisce gli strumenti per sviluppare sistemi di difesa contro di essi. In questa sezione ci concentreremo sugli attacchi, descrivendone il funzionamento e una possibile implementazione su modelli ADAS realizzati all'interno del simulatore Carla.

Gli attacchi presenti nella libreria sono suddivisi nel seguente modo:

- Evasion Attacks,dove i dati in input vengono modificati fino ad avere un errore di classificazione
- Poisoning Attacks. In questo caso l'obiettivo è iniettare dati costruiti in modo specifico per compromettere la fase di apprendimento. Sono particolarmente efficaci nei casi in cui il riaddestramento è frequente.
- Extraction Attacks. Nei casi in cui il modello di apprendimento non sia direttamente accessibile, questi tipi di attacchi vengono sviluppati per addestrare un modello sostituto che sia funzionalmente equivalente al modello target.

Nel contesto della guida autonoma, ha senso concentrarsi sugli Evasion Attacks. Possiamo infatti assumere che il riaddestramento non avvenga con una tale frequenza da giustificare lo sviluppo intensivo di un attacco poisoning. Per quanto riguarda gli **Extraction Attacks** , muovendoci in un contesto open source e accademico dove i modelli sono liberamente accessibili, hanno un'utilità limitata per i nostri obiettivi. Passiamo ora a descrivere alcuni degli esempi presenti nel toolbox.

## 4.1 TRASFORMAZIONI SPAZIALI

I primi attacchi che consideriamo sono quelli che applicano semplici trasformazioni spaziali alle immagini senza modificare i pixel in modo diretto. Si tratta di attacchi molto interessanti in quanto facilmente implementabili. Si tratta infatti di ruotare, spostare o sovrapporre più immagini.

| Lista di Attacchi | | | |
|---|---|---|---|
| Nome | Descrizione attacco | Applicabilità | Implementazione in Carla |
| Adversarial Patch [27] | L'attacco consiste nell'attaccare su un qualsiasi oggetto una specifica patch. Questa patch è creata in modo da causare errori di classificazione. Può prendere qualsiasi forma e viene addestrata su un'insieme di immagini, nelle quali verrà applicata con dimensioni e rotazione casuali | Un attaccante potrebbe semplicemente stampare la patch e attaccarla ad esempio su un cartello stradale, mandando in confusione i sistemi ADAS. Se si vuole utilizzare l'attacco su delle immagini si può semplicemente porre la patch direttamente sull'immagine. | È necessario modificare gli oggetti della simulazione(segnali, veicoli). Nello specifico, dobbiamo intervenire sul motore UE4, responsabile della costruzione di tali oggetti |
| Spatial Transformation Attack [28] | Le immagini che arrivano al classificatore vengono sottoposte a rotazione e traslazione fino a causare un errore di classificazione | L'attacco potrebbe essere iniettato a livello dei sensori di un sistema ADAS, in modo da modificare direttamente i dati in input causando problemi difficilmente rintracciabili. Questo richiede un tampering della telecamera, per imporre la trasformazione voluta alle immagini. | Si utilizzano le API fornite dal simulatore per acquisire le immagini create |

## 4.2 PERTURBAZIONE DELL'IMMAGINE

In questo caso l'immagine in input viene sottoposta a una perturbazione, ovvero una modifica di un certo numero di pixel in modo da non essere visibile all'occhio umano ma in grado di causare misclassificazione. Questi attacchi risultano essere molto simili tra loro, ma spesso una piccola modifica può causare risultati estremamente diversi. Nello specifico l'adversarial sample $\mathbf{x'}$ viene definito come:

$$\mathbf{x'} = \mathbf{x} + \epsilon_x$$
$$\{\mathbf{x'} \in \mathbb{R}^{m \times n \times 3} | \operatorname*{argmax}_j f(x') \neq \operatorname*{argmax}_i f(x)\}$$

dove $\epsilon_x \in \mathbb{R}^{m \times n \times 3}$ è la perturbazione aggiunta all'input [29].

| Lista di Attacchi | | | |
|---|---|---|---|
| Nome | Descrizione attacco | Applicabilità | Implementazione in Carla |
| Threshold Attack [29] | Viene imposta una soglia massima $th$ alla perturbazione. Nello specifico l'attacco ottimizza il vincolo $\|\epsilon_x\|_\infty \leqslant th$ dove $th$ è uno dei seguenti valori {1, 3, 5, 10} | Dobbiamo poter perturbare le immagini e successivamente passarle al classificatore. Per questo l'attacco verrà implementato a livello dell'unità di elaborazione. | Si utilizzano le python API fornite dal simulatore per avere accesso alla struttura interna dei veicoli, dove risiede il classificatore |
| Low Pixel Attack [29] | È una variazione del threshold attack usando la norma 0 al posto della norma infinito. | Poichè l'attacco è concettualmente identico al threshold attack anch'esso verra iniettato nell'unità di elaborazione | Si accede alla struttura interna dei veicoli con le python API |
| Decision Tree Attack [30] | Si sfrutta la struttura ad albero del classificatore, cercando un cammino dalla foglia originale ad una foglia che corrisponde a una classe diversa. Infine si applica la perturbazione all'esempio | L'attacco viene iniettato all'interno dell'unità di elaborazione del sistema di guida autonoma. | Si accede alla struttura interna dei veicoli con le python API |

# BIBLIOGRAFIA


[1] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach (3rd Edition)*. Prentice Hall, 2009. (Cited on pages 2, 7, 8, 9, 10, and 11.)

[2] John Mccarthy. What is artificial intelligence. 1998. (Cited on pages 7 and 10.)

[3] David Poole and Alan Mackworth. *Artificial Intelligence: Foundations of Computational Agents*. Cambridge University Press, Cambridge, UK, 2017. (Cited on pages 8 and 12.)

[4] Chethan Kumar. Artificial intelligence:definition, types, examples, tecnologies, 2018. (Cited on pages 2 and 11.)

[5] Wikipedia. Machine learning, 2020. (Cited on page 11.)

[6] Tom M. Mitchell. *Machine Learning*. McGraw-Hill, 1997. (Cited on page 12.)

[7] Glenn Doggett Julia Bonafede, Corey Cook. Artificial intelligence and its potential impact on the cfa institute code of ethics and standards of professional conduct, 2019. (Cited on pages 2 and 13.)

[8] Isha Salian. Supervize me: What's the difference between supervised, unsupervised, semi-supervised and reinforcement learning?, 2018. (Cited on pages 2 and 14.)

[9] Geoffrey Hinton Yann LeCun, Yoshua Bengio. Deep learning. *Nature*, pages 436,446, May 2015. (Cited on pages 15 and 18.)

[10] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`. (Cited on page 15.)

[11] Facundo Bre, Juan Gimenez, and Víctor Fachinotti. Prediction of wind pressure coefficients on building surfaces using artificial neural networks. *Energy and Buildings*, 158, 11 2017. (Cited on pages 2 and 16.)

[12] Alessandro Mazzetti. *Reti Neurali Artificiali:introduzione ai principali modelli e simulazione su personal computer*. 1991. (Cited on page 16.)

[13] Ali Hasan, Hayder Al-Assadi, and Ahmad Azlan. *Neural Networks Based Inverse Kinematics Solution for Serial Robot Manipulators Passing Through Singularities*. 04 2011. (Cited on pages 2 and 16.)

[14] Shruti Jadon. Introduction to different activation functions for deep learning. *Medium*, 2018. (Cited on pages 2 and 17.)

[15] Hongquan Guo, Hoang Nguyen, Diep-Anh Vu, and Xuan-Nam Bui. Forecasting mining capital cost for open-pit mining projects based on artificial neural network approach. *Resources Policy*, 08 2019. (Cited on pages 2 and 17.)

[16] Pete Goldin. 10 advantagages of autonomous vehicles. *ITSdigest*, 2018. (Cited on page 20.)

[17] Causes of traffic accidents in austin, 2015. (Cited on pages 2 and 21.)

[18] Rob van der Heijden and Kiliaan van Wees. Introducing advanced driver assistance systems: Some legal issues. *European Journal of Transport and Infrastructure Research*, 1(3), 2001. (Cited on page 20.)

[19] Patrick Lin. *Why Ethics Matters for Autonomous Cars*, pages 69–85. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015. (Cited on page 20.)

[20] Margaret Rouse. Self driving car, 2019. (Cited on pages 2 and 22.)

[21] new level 3 autonomous vehicles hitting the road in 2020. (Cited on page 23.)

[22] Paul McLellan. Automotive sensors:cameras,lidar, radar, thermal. *Breakfast Bytes Blog*, 2018. (Cited on pages 2 and 24.)

[23] Charles Murray. Automakers, suppliers ratchet up autonomous car programs. *DesignNews*, 2016. (Cited on pages 2 and 24.)

[24] Giuliano Giacaglia. Self driving cars. *Medium*, 2019. (Cited on pages 2 and 25.)

[25] A.Avizienis, J.C.Laprie, B.Randell, and C.Landwehr. Basic concepts and taxonomoy of dependable and secure computing. Technical report, Institute for Systems Research, 2004. (Cited on pages 2, 26, and 29.)

[26] Wikipedia. Dependability. (Cited on pages 2 and 27.)

[27] Tom B. Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. Adversarial patch. *CoRR*, abs/1712.09665, 2017. (Cited on page 33.)

[28] Logan Engstrom, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. A rotation and a translation suffice: Fooling cnns with simple transformations. *CoRR*, abs/1712.02779, 2017. (Cited on page 33.)

[29] Danilo Vasconcellos Vargas and Shashank Kotyan. Model agnostic dual quality assessment for adversarial machine learning and an analysis of current neural networks and defenses. *CoRR*, abs/1906.06026, 2019. (Cited on pages 34 and 35.)

[30] Nicolas Papernot, Patrick D. McDaniel, and Ian J. Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *CoRR*, abs/1605.07277, 2016. (Cited on page 35.)