FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

# Fahreridentifikation mittels Machine-Learning

# Driver identification using Machine-Learning

## Masterarbeit

Zur Erlangung des akademischen Grades

**Master of Science in Engineering**

der Fachhochschule Campus Wien
Masterstudiengang: ITS-20


**Vorgelegt von:**
David Lechner

**Personenkennzeichen:**
1810537012

**ErstbetreuerIn / ErstbegutachterIn:**
Dr. Martin Schmiedecker

**ZweitbetreuerIn / ZweitbegutachterIn:**
Kevin Koch


**Eingereicht am:**
tt.mm.jjjj

Erklärung:

Ich erkläre, dass die vorliegende Masterarbeit von mir selbst verfasst wurde und ich keine anderen als die angeführten Behelfe verwendet bzw. mich auch sonst keiner unerlaubter Hilfe bedient habe.

Ich versichere, dass ich diese Masterarbeit bisher weder im In- noch im Ausland (einer Beurteilerin/einem Beurteiler zur Begutachtung) in irgendeiner Form als Prüfungsarbeit vorgelegt habe.

Weiters versichere ich, dass die von mir eingereichten Exemplare (ausgedruckt und elektronisch) identisch sind.

Datum: Unterschrift:

## Kurzfassung

(Z.B. “Diese Arbeit beschäftigt sich mit...”)

## Abstract

(E.g. “This thesis deals with...”)

# Abkürzungsverzeichnis

| | |
|---|---|
| AI | Artificial Intelligence |
| ASAM | Association for Standardization of Automation and Measuring Systems |
| CAN | Controller Area Network |
| MDF | Measurement Data Format |
| ECU | Electronic Control Units |
| ML | Machine Learning |
| OBD | On-Board-Diagnose |

## Schlüsselbegriffe

Machine Learning
CAN-Bus
Driver Fingerprinting

# Inhaltsverzeichnis

# 1. Einleitung

In der heutigen Zeit werden Unmengen an Daten von verschiedensten Geräten generiert und versendet. Den Anfang haben Smartphones, dann Wearables gemacht. Mit dem Aufkommen des Bereichs Internet of Things (IoT) kann jegliches technische Gerät – von der Lampe bis hin zur Fertigungsmaschine – mit dem Internet verbunden sein und Status- beziehungsweise Messdaten übermitteln. Dieser Trend macht auch vor Fahrzeugen keinen Halt. Schon längst sind moderne Autos mit LTE-, GPS und Wifi-Modulen ausgestattet und senden Daten unter anderem zum Hersteller. Gartner prognostiziert für das Jahr 2020 470 Millionen vernetzte Fahrzeuge [Gar19]. In Zukunft werden wahrscheinlich alle Fahrzeuge mit etlichen Sensoren ausgestattet sein und kommunizieren untereinander, mit der Umwelt, dem Fahrer oder sonstigen Service-Anbieter. Dies gründet vor allem auf den wachsenden Themen Vehicle-to-Everything (V2X) und Autonomes Fahren. Insbesondere beim Letztgenannten macht die generierte Datengröße noch einen großen Sprung, da neben Sensoren auch Radar und Videokameras zur Umwelterkennung hinzukommen. Jedoch versenden schon heute Electronic Control Units (ECUs) Daten im Auto, wie zum Beispiel Lenkradwinkel, Gangposition und Bremsdruck, welche für Sicherheits- und Komfortfunktionen genutzt werden. Aus diesen Daten können viele Informationen gewonnen werden. Eine davon ist, den Lenker eines Autos während der Fahrt nur durch das Fahrverhalten zu identifizieren.

Daraus lassen sich weitere unterschiedliche Anwendungen ableiten. Einige von ihnen schaffen Komfort und erleichtern in gewisser Weise das Leben des Fahrers. Andere indes könnten gegen den Fahrzeughalter und der Fahrerin selbst eingesetzt werden, diese gehen mit datenschutzrechtlichen Bedenken einher.

Doch zunächst zu den Anwendungen, welche eine positive Auswirkung haben können. Moderne Autos – vor allem jene mit einem Automatik Getriebe – bieten die Möglichkeit, sich an den Fahrstil anzupassen. Wenn beispielsweise eine Person zum schnelleren Beschleunigen neigt, lernt dies das Auto und schaltet demnach erst bei einer höheren Motordrehzahl in den nächsthöheren Gang. Dasselbe gilt bei einem gemächlichen Fahrstil, wobei hier eher früher geschalten wird. Lernt das Auto nun von einer Person mit dem zweitgenannten Stil und wird aber auch hin und wieder mit anderen Personen, zum Beispiel Familienmitglieder, geteilt, kann es für diese einen Komfortverlust darstellen. Identifiziert das Auto jedoch durch das Fahrverhalten eine andere Person, könnte es den gelernten Stil temporär vergessen oder gar ein neues Profil anlegen und erneut lernen.

Im Unternehmensbereich, wo Dienstfahrzeuge oder Lieferwägen zum Einsatz kommen, ist es meistens notwendig ein Fahrtenbuch zu führen. Bei einem klassischen Fahrtenbuch werden die gefahrenen Kilometer, Datum und Uhrzeit, Name des Fahrers, Abfahrts- und Zielort sowie die Unterschrift in einem Buch im Fahrzeug festgehalten. Durch das händische Eintragen kann es zu Zeitverlusten, unvollständige Dokumentation und auch manipulierten Daten kommen, was im Endeffekt in hohen Kosten resultiert. Elektronische Fahrtenbücher sind in der Lage diese Daten digital aufzuzeichnen. Mithilfe eines *On-Board-Diagnose* II (OBDII) Dongles werden der Kilometerstand und die Fahrzeugnummer erfasst. Die GPS-Position, Datum und Uhrzeit wie auch die Fahreridentifikation und gegebenenfalls der Zweck der Fahrt

werden mit einem gekoppelten Smartphone in das System ergänzt [1] [2]. Die Abrechnung und Auswertung lässt sich dadurch erheblich erleichtern, jedoch ist noch immer eine Interaktion des Lenkers erforderlich, was wiederum Raum für Fehler und Manipulation schafft. Kommt eine Fahreridentifikation basieren am Fahrverhalten zum Einsatz, kann das komplette System in das Fahrzeug integriert werden. Die Positionsdaten können dabei von einem integrierten Navigationssystem ausgelesen werden und die restlichen Fahrzeugdaten über den CAN-Bus. Die Fehlerwahrscheinlichkeit verringert sich dabei enorm und die Benutzerfreundlichkeit wird erhöht, da nichts mehr händisch eingetragen werden muss.

Überdies ist auch eine Art Diebstahlwarnung zu realisieren. Dem System sind eine Reihe an Fahrerprofilen bekannt, welche zuvor eingelernt wurden. Bei jeder neuen Fahrt wird überprüft, ob das momentane Fahrverhalten des Lenkers erkannt wird. Stellt das System zum Beispiel zu einer Wahrscheinlichkeit von 90% fest, dass sich das Profil nicht unten den bekannten befindet, kann beispielsweise eine Benachrichtigung an den Fahrzeughalter versendet, oder die Fahrerin zum Anhalten gebracht werden.

Der Mechanismus kann weiters dazu verwendet werden, Fahrzeug-Funktionen und Leistung fahrerabhängig zu steuern. Ein Familienvater ist so etwa in der Lage, die zur Verfügung stehenden PS einzuschränken, wenn seine Kinder mit dem Auto fahren.

Durch das Aufzeichnen und Analysieren von personenbezogenen Daten kommen natürlich auch datenschutzrechtliche Bedenken auf. Werden die Daten in eine Cloud – sei es eine vom Hersteller, der Versicherung oder einer anderer Drittpartei – gesendet und ausgewertet, können Personen von diesen Unternehmen oder Organisationen eindeutig identifiziert werden. Liegen zudem Standortdaten des Autos vor, kann auch eine genau Ortung durchgeführt werden. Dies kann in einigen Fällen problematisch sein. So könnte eventuell der Hersteller bestimmte Services anbieten, welche personalisierte Werbungen während dem Fahren anzeigen. Zum Beispiel ist es dadurch möglich, bevorzugte Restaurants in der Navigationsansicht hervorzuheben. Auch Versicherung können diese Informationen für sich nutzen, um Fahrer- und Fahrstil abhängige Versicherungspakete anbieten zu können [3]. Hier wird genauso ein OBDII-Dongle verbunden mit einer App für die Fahrstilanalyse eingesetzt. Werden viele Notbremsungen und rasche Beschleunigen verzeichnet, kann die Versicherungsprämie erhöht werden. Der Dongle und das Smartphone könnten durch das System ersetzt werden.

Die angeführten Beispiele zeigen, dass ein System, welches die Person hinter dem Lenkrad eines Fahrzeuges eindeutig identifiziert, Benutzervorteile bringen kann. Zudem erschließt sich ein neues Geschäftsfeld für Autohersteller, um vielleicht Premium-Features anbieten zu können. Jedoch stellen sich auch Fragen zur Privatsphäre und wie mit solch sensiblen Daten umgegangen wird.

## 1.1. Überblick und Ziele

In der Masterarbeit wird ein System vorgestellt, welches den Lenker eines Fahrzeuges anhand des Fahrstils eindeutig identifizieren kann. Ausgangspunkt ist, dass jedes Individuum ein anderes Verhalten in verschiedenen Verkehrssituationen hat. Einer bremst eher früher an dafür nicht so kräftig, währendessen eine später und härter bremst. Person A fährt eine Kurve mit 30 km/h im dritten Gang und lenkt dabei etwas weniger ein. Person B andererseits nimmt eine Kurve schneller und muss daher mehr einlenken. Dafür muss vielleicht die Lenkradposition in der Kurve nicht mehr angepasst werden, wohingegen vielleicht einmal

[1] Drivebox: `https://www.drivebox.at/drivebox_main.html`
[2] Vimcar: `https://vimcar.de/fahrtenbuch`
[3] Signal Iduna: `https://www.app-drive.de`

kurz die Bremse gedrückt wird. All diese Informationen - Geschwindigkeit, Drehmoment, Lenkradwinkel, Bremsdruck usw. - werden über den *Controller-Area-Network* (CAN) Bus als Nachrichten ausgetauscht. *Electronic-Control-Units* (ECUs) verarbeiten diese und steuern dementsprechend den Motor, das Getriebe oder eine Öldruckpumpe an.

Um aus vershiedenen CAN-Daten einen Zusammenhang zum Lenker herstellen zu können, wird eine Untermenge von *Artificial Intelligence* (AI), nämlich *Machine Learning* (ML) verwendet. Bei ML kommen mehrere mathematische Algorithmen zum Einsatz, welche statistische Modelle aufgrund von Trainingsdaten aufbauen. Die Modelle versuchen daraufhin ein Muster in den Daten zu erkennen, um später eine Aussage über den Lenker treffen zu können. Die Lerndaten bestehen dabei aus den verschiedenen CAN-Nachrichten und einer Fahreridentifikation - dem Zielwert. Nach der Trainingsphase des Modells werden nur noch die CAN-Daten eingespielt. Als Resultat wird die Fahrer-ID mit größen zutreffenden Wahrscheinlichkeit ausgegeben [Con12].

Das Ziel der Masterarbeit ist bereits existierende Methoden zur Fahreridentifizierung mit *Machine Learning* (siehe 1.3) und den vorliegenden Daten (siehe 3.1) zu validieren beziehungsweise zu optimieren. Im Zuge dessen soll herausgefunden werden, wie lange die Trainingsphase eines ML-Modells sein muss, um eine Trefferquote von über 85% erzielen zu können. Des Weiteren gilt es die Dauer zu bestimmen, welche für die Identifikation (zu 85%) eines Lenkers mit einem bereits trainierten Modell benötigt wird. Ein weiteres Ziel ist das System in ein Fahrzeug zu integrieren und mit (fast) Echtzeit-Daten zu erproben. Hierfür wird ein Embedded-Device mit beschränkten Ressourcen eingesetzt.

## 1.2. Struktur der Arbeit

Diese Arbeit geht eingangs auf die Motivation und Problemstellung ein. Danach wird ein Überblick mit den Zielen geschaffen, sowie vergleichbare Arbeiten vorgestellt. Im zweiten Kapitel werden die Grundlagen behandelt. Das beinhaltet vorwiegend den CAN-Bus und *Machine Learning*. Kapitel 3 beschreibt die Umsetzung des Systems. Dabei wird auf die vorliegenden Daten eingegangen und wie diese bestmöglich vorbereitet werden. Des Weiteren wird auch die Implementierung verschiedener ML-Algorithmen beschrieben und wie damit der Lenker eins Fahrzeuges identifiziert werden kann.

## 1.3. Verwandte Arbeiten

Zu diesem Thema sind bereits einige Papers zu finden. Zum Beispiel konnten 2005 Forscher aus Japan eine Fahreridentifizierung mit einer Genauigkeit von 73% durchführen [WOM+05]. Sie verwendeten jedoch CAN-Bus Nachrichten von einem Simulator. Später wurde die Trefferquote im Labor zwar auf 89.6% erhöht aber die Anwendung unter realen Bedingungen brachten nur 71% ein [MNO+07]. Im folgenden werden noch drei Papers vorgestellt, welche die Identifikation ausschließlich mit echten Fahrzeugdaten untersucht haben.

In einem Paper von 2016 wurde untersucht, ob Einzelpersonen basierend auf ihren natürlichen Fahrverhalten identifiziert werden können. Für die Datenbasis wurden CAN-Nachrichten eines Serienfahrzeuges verwendet. 15 Teilnehmerinnen mussten zuerst bestimme Manöver auf einem abgesperrten Parkplatz durchführen und danach eine ca. 80 km lange vordefinierte Strecke abfahren. Für die Analyse wurde *Machine Learning* mit verschiedenen Algorithmen verwendet. Dabei konnte festgestellt werden, dass bei einem 1 zu 1 Vergleich die Teilnehmer

zu 100% unterscheidbar sind. Des Weiteren konnte eine hohe Identifikationswahrscheinlichkeit bei nur acht Minuten Trainingszeit erzielt werden [ETKK16].

Die Arbeit von B. Gahr et. al. von 2018 setzt auf die soeben beschriebene auf. Da gezeigt wurde, dass eine Identifikation zu 100% möglich ist, hat diese es versucht, die Methoden mit anderen Daten zu validieren. Dafür wurden aber nicht direkt die Nachrichten vom CAN-Bus abgegriffen, sondern über ein Smartphone, welches über Bluetooth mit einem OBDII-Dongle verbunden wurde. Hierbei konnte mit den Methoden jedoch nur eine Identifikationsrate von maximal 70% erzielt werden. Daher wurde ein anderer Ansatz gewählt, bei dem nur Daten während eines Bremsvorgangs in Betracht gezogen werden. Das hat zu Ergebnissen zwischen 80 und 99,5% geführt [GRDW18].

Ein anderes Paper verfolgte einen ähnlichen Ansatz, bei dem nur die Daten während einer Kurve berücksichtigt werden. Die CAN-Nachrichten wurden hierbei mit einem proprietären Data-Logger aufgezeichnet. Es folgte eine Analyse der 12 häufigsten Kurven im Datensatz. Dabei konnte ein Fahrer verglichen mit einem zweiten Fahrer mit einer Wahrscheinlichkeit von fast 77% unterschieden werden. Besteht das Set aus fünf Fahrern, liegt die Identifikationsrate bei 50,1% [HSS+16].

## 1.4. Neuigkeitswert

Die Masterarbeit wird teilweise auf die bereits bestehenden Papers aufsetzten und bewehrte Methoden übernehmen. Ein wesentlicher Unterschied ist aber, dass die hier vorliegenden Daten (siehe 3.1) weder in einem bestimmten Setting noch unter anderen Kontrolleinflüssen und direkt vom CAN-Bus mitgemessen worden sind. Wie auch aus den Zielen hervorgeht, wird versucht, die Methoden dahingehend zu verbessern, sodass eine möglichst schnelle Identifikation durchgeführt werden kann.

# 2. Grundlagen

In diesem Kapitel werden die Grundlagen für das zu realisierende System beschrieben. Es beinhaltet eine Einführung in *Machine Learning* und gibt dabei einen Überblick auf die verschiedenen Methoden und Anwendungsfälle. Zudem wird der *Controller-Area-Network*-Bus vorgestellt, welcher für die Datensammlung im Kontext der Masterarbeit essenziell ist.

## 2.1. Machine Learning

*Machine Learning* ist ein Bereich der Computer Wissenschaften und beschäftigt sich mit Algorithmen und Techniken zur Lösung von komplexen Problemen, wo konventionellen Programmiermethoden nicht vielversprechend sind. Wir werden heutzutage im alltäglichen Leben mit vielen Anwendungen konfrontiert, die ohne ML nicht möglich wären. Beispiele dafür sind Sprachsteuerung, Bild-, Gesichts- und Handschrifterkennung, Stauvorhersage aber auch Autonomes Fahren und die Erkennung von Brustkrebs [KEE+15]. Schon vor 1980 wurde versucht einige solcher komplexen Probleme zu lösen, was jedoch nicht immer von Erfolg gekrönt war. Erst mitte der 2000er Jahre ist der Fortschritt in dem Bereich drastisch gestiegen. Gründe dafür gibt es mehrere. Zum einen ist mit dem Aufkommen des Internets eine viel größere Anzahl an Daten vorhanden und zum anderen sind Rechenleistung beziehungsweise Speicherplatz billiger, leichter zugänglich und vorallem performanter. Des Weiteren sind die Algorithmen verbessert und angepasst worden.

Oft werden *Machine Learning* und Künstliche Intelligenz (engl. *Artificial Intelligence*, kurz AI) mit einander gleichgesetzt, was jedoch nicht stimmt. AI ist ein viel breitgefächerter Forschungsbereich, der mittels mehreren Ansätzen versucht Maschinen das "Denken"beizubringen. ML hingegen ist ein möglicher Ansatz dafür.

Um das Vorgehen zum Lösen eines komplexen Problems mittels *Machine Learning* besser verstehen zu können, wird es im folgenden anhand einer Anwendung, welche handgeschrieben Buchstaben erkennt, erklärt. Zu aller erst wird eine Vielzahl an verschiedenen Datenpunkten (engl. *data points*) - Bilder mit handgeschrieben Buchstaben - benötigt. Zusätzlich müssen diese mit dem enthaltenen Buchstaben markiert (engl. *labelled*) sein. Das Ziel ist, dass die Anwendung nicht nur die Buchstaben im Datenset erkennt, sondern auch jene, die nicht enthalten sind. Mit der konventionellen Methode wird anfangs versucht zu verstehen wie die Bilder mit den Buchstaben zusammenhängen. Danach wird eine Reihe an Regeln festgelegt, um auch neue Bilder erkennen zu können. Da es jedoch eine große Variation von Handschriften gibt, kann das Regelset sehr schnell sehr groß werden. *Machine Learning* Algorithmen gehen hierbei einen mehr generellen Lösungsweg, indem sie direkt von den markierten Daten lernen und sich das Regelset (engl. *model*) selbst aneignen. Je mehr Bilder von Buchstaben vorhanden sind, desto genauer wird das ML-*Model*. Werden neue Bilder ohne Markierung hinzugefügt, kann das *Model* den Buchstaben erkennen. Diese Art um das Problem zu Lösen wird im Kontext von ML als Klassifizierung bezeichnet. Es gibt auch noch zwei weitere, welche adressiert werden:

- Prognose (engl. *Prediction*): Trainieren eines *Models* mit historischen Daten zur Vorhersage zukünftiger Werte. Z.B. der Bedarf eines bestimmten Produktes in den Som-

merferien.

- Klassifizierung (engl. *Classification*): Datenpunkte in eine oder mehrere Kategorien bzw. Klassen einteilen. Z.B. Identifizierung einer Email als Spam oder nicht, Erkennen welches Tier auf einem Bild zu sehen ist (Katze, Hund, Löwe, ...)
- Gruppierung (engl. *Clustering*): Unterteilung vieler Datenpunkte in wenige Gruppen, in denen Punkte mit gleichen Eigenschaften enthalten sind. Im Gegensatz zur Klassifizierung ist die Anzahl der Gruppen im vorhinein nicht bekannt.

Zur besseren Vorstellung veranschaulicht Abbildung 2.1 die Arten. Abschnitt 2.1.2 und folgende gehen darauf näher ein.

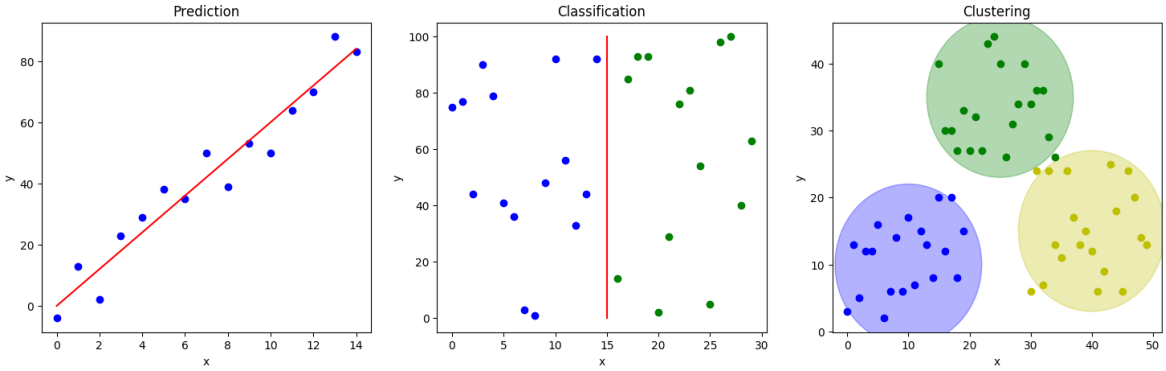


Abbildung 2.1.: Arten von *Machine Learning* Problemen (Quelle: Author)

### 2.1.1. Ansätze

Um ein ML-*Model* anzulernen gibt es verschiedene Möglichkeiten. An eine davon - das Überwachte Lernen - wurde bereits in der Einführung anhand eines Beispiels herangeführt. Dieses Unterkapitel geht auch auf die anderen Möglichkeiten ein.

#### Überwachtes Lernen

Beim Überwachten Lernen (engl. *supervised learning*) wird dem *Model* ein Datenset bestehend aus Datenpunkten und den dazugehörigen korrekten Antwort zu einer Frage übergeben. Der ML-Algorithmus versucht aufgrund den Eigenschaften eines Datenpunktes einen Zusammenhand zu der Antwort zu finden. Wenn daraufhin neue Datenpunkte hinzugefügt werden, kann das *Model* basierend auf den Eigenschaften eine Antwort prognostizieren. Das folgende abstrakte Beispiel demonstriert wie *supervised learning* funktioniert.

Seit der Geburt lernt ein Kind, wie bestimmte Objekte aussehen und wie sie heißen. Es sieht jeden Tag einen Hund in verschiedenen Postionen, mal sitzend, laufend, stehend usw. Dadurch kann es auch den Nachbarshund als solchen identifizieren und zwischen einer Katze unterscheiden. Während der Lernphase kann natürlich auch ein Fehler vorkommen und zum Beispiel einen Wolf als einen Hund misinterpretieren. Doch die Eltern und Geschwister erklären dem Kind, dass es sich dabei nicht um einen Hund handelt. Es passt daher das Verständnis an und wird immer besser das Tier zu identifizieren. Im Kontext von ML ist die oben beschrieben Anwendung zur Erkennung von handgeschriebenen Buchstaben genauso überwachtes Lernen.

### Nicht-Überwachtes Lernen

Bei dieser Art des Lernens sind im Datenset nicht die korrekten Antworten zu den einzelnen Datenpunkten enthalten. Der Algorithmus ist darauf ausgelegt Trends zu erkennen und Datenpunkte mit Gemeinsamkeiten zu gruppieren ohne zu wissen, worum was es sich dabei handelt. Um auf das oben beschrieben Beispiel mit dem Hund zurück zukommen, werden in einem *Model* viele Bilder von verschiedenen Tieren eingespielt. Die Bilder sind jedoch nicht mit dem darauf enthaltenen Tier markiert. Nachdem der Algorithmus alle Charakteristiken der Bilder analysiert hat ist das *Model* in der Lage, gleichartige Bilder zusammen zufassen. Es kann somit eine Gruppe erstellen, die alle Hunde enthält und eine andere, der alle Katzen zugeordnet sind. Am Ende hat das ML-*Model* noch immer keine Vorstellung, um welches Tier es sich tatsächlich handelt.

Ein anderes Beispiel ist in der Analyse von Kaufverhalten zu finden. Das Datenset besteht hierbei aus den verschiedenen gekauften Produkten der Kunden und das Ziel ist Korrelationen darin zu finden. Das *Model* soll in der Lage sein zum Beispiel folgendes bestimme zu können: Kunden die Schultaschen kaufen, kaufen auch Stifte. Oder Kunden die Bier kaufen, kaufen auch Chips.

### Teil-Überwachtes Lernen

Teil-Überwachtes Lernen (engl. *semi-supervised learning*) liegt zwischen den beiden vorher beschriebenen Arten des Lernens. Es wird ein Datenset verwendet, dass hauptsächlich aus unmarkierten Datenpunkten besteht. Ein kleiner Teil davon hat jedoch eine Markierung. Im ersten Schritt kommen Techniken zur Gruppierung zum Einsatz, um gleichartige Datenpunkte zu bündeln. Der zweite Schritt besteht darin, die bereits bekannten Datenpunkte dazu zu verwenden, andere Daten in der gleichen Gruppe zu markieren. Einer der größten Vorteile davon ist, dass nicht viel Zeit für das manuelle Markieren von Daten aufgewendet werden muss. Der Nachteil ist aber, dass es im vergleich zum Überwachten Lernen komplexer ist.

Auch hier lässt sich das Beispiel mit den Tieren anwenden. Aus Zeit- oder anderen technischen Gründen können nicht alle Bilder von Hunden und Katzen durchgesehen und markiert werden. Bei ein paar ist es jedoch möglich. Beim Teil-Überwachten Lernen gruppiert zuerst das ML-*Model* alle Bilder mit Hunden und Katzen (gleiche Eigenschaften). Danach können aus den paar Markierten alle anderen abgeleitet werden.

### Bestärkendes Lernen

Die letzte hier vorgestellte Art ist das Bestärkendes Lernen (engl. *reinforcement learning*). Es kommt einerseits zum Einsatz, wenn sich die Situationen fortlaufend ändern, zum Beispiel beim Autofahren oder bei einem Spiel. Das *Model* muss sich hierbei immer an neue Bedingungen anpassen. Andererseits wird es auch verwendet, wenn ein großer Zustandsraum existiert. Bei beispielsweise Schach ist es kaum möglich mittels *brute-force* den besten nächsten Zug herauszufinden, da es viel zu viele Möglichkeiten im Verlauf eines Spieles gibt. Der Algorithmus ist also darauf ausgelegt eine Entscheidung basierend auf den momentanen internen Zustand und der Umgebung zu treffen, um ein vordefiniertes Ziel zu erreichen. Ein Ziel kann gegebenenfalls sein, ein Auto innerhalb der Spur zu halten. Je länger der Algorithmus lernen kann, desto besser und genauer werden die Entscheidungen in Hinblick auf eine langfristige Auswirkung.

### 2.1.2. Regression

Die Regressionsanalyse (engl. *regression analysis*) ist eine auf Statistik basierende *Machine Learning*-Technik, welche aufgrund von oftmals historischen Daten zur Vorhersage verwendet wird. Dabei werden Relationen in markierten Daten analysiert, um eine Prognose für bestimmte Eigenschaften treffen zu können. Zum Beispiel kann damit ein *Model* erstellt werden, welches den aktuellen Preis einer Immobilie bestimmt. Die historischen markierten Daten können hier die Eigenschaften Quadratmeteranzahl, Nachfrage, Lage als Kategorie, Kaufpreis, Verkaufspreis etc. haben. Im Kontext von ML werden die Eigenschaften als *Features* bezeichnet

## 2.2. CAN Bus

Was ist der CAN Bus?

### 2.2.1. DBC Datei

was is a dbc datei

### 2.2.2. MDF

*Measurement Data Format* (MDF) ist ein binäres Dateiformat, welches 1991 von der Firma Vector Informatik GmbH in Zusammenarbeit mit der Robert Bosch GmbH entwickelt wurde. Es ist speziell für Messdaten im Automotive-Bereich konzipiert und ist seit 2009 in der Version 4 als offizieller Standard von *Association for Standardization of Automation and Measuring Systems* (ASAM) öffentlich zugänglich. Ein wesentlicher Vorteil des Formates ist, dass die Messdaten sehr schnell und speicherplatzsparend abgespeichert werden können. Des Weiteren wird auch eine Optimierung des Lesevorgangs durch Vorsortierung und Indizierung unterstützt. Neben den eigentlichen Nutzdaten werden auch Metadaten aus der DBC-Datei mitgespeichert. Dies ist vor allem für weitere Analysen und die Interpretierung der Rohdaten notwendig. Beispielsweise gehören die Information zur Umwandlung in physikalische Werte und Signalnamen dazu. [fSoAS14]

## 2.3. Embedded Devices

Vielleicht auch Einführung in Embedded Systems?

# 3. Umsetzung

Dieses Kapitel geht auf die konkrete Umsetzung des Systems ein. Anfangs werden die zur Verfügung stehenden Daten beschrieben und wie diese für *Machine Learning* vorverarbeitet werden. Danach werden die eingesetzten ML-*Model* erklärt und die Optimierung dieser gezeigt. Den Abschluss bildet die Integration in ein Fahrzeug.

## 3.1. CAN-Daten

Bei den Daten, welche für die Masterarbeit vorliegen, handelt es sich um CAN-Nachrichten, aufgenommen in zehn Fahrzeugen (VW Golf 7, keine näheren Informationen) während dem Fahren. Dabei ist sichergestellt, dass jeweils nur eine Person hinter dem Lenkrad gesessen ist. Für die Aufzeichnung wurde ein Boardcomputer mit Internetkonnektivität (*ALEN*, siehe 3.4.1) im Fahrerraum verwendet. Das *Embedded-Device* ist über zwei CAN-Bus Schnittstellen mit der Fahrwerks-Linie (F-CAN) und der Fahrzeugelektrik-Linie (K-CAN) verbunden und kann somit alle Nachrichten in fast-echtzeit vom Bus mitlesen. Dabei ist aber sichergestellt, dass es nicht möglich ist, auch Nachrichten senden zu können, da dies bei einer Fehlfunktion zu einem Unfall führen und Insaßen gefährden könnte. Die Nachrichten werden nach dem Empfangen mit einem synchronisierten Zeitstempel, aufgelöst in Nanosekunden, versehen und in eine MDF Datei mit den Metadaten von der entsprechenden DBC-Datenbank gespeichert. Eine Datei enthält jeweils Messdaten über eine Dauer von einer Minute. Danach wurde die Datei temporär auf einer Festplatte gespeichert und auf einem File-Server über eine LTE-Funkverbindung hochgeladen. Um sie über alle Fahrzeuge hinweg eindeutig zuordnen zu können, wurde eine Fahrzeugidentifikationsnummer dem Dateinamen angehängt.

In einer Datei sind 46 verschiedene Signale der beiden CAN-Busse enthalten. Tablle 3.1 listet die meisten davon. Jene, die sich vorrausichtlich nicht für die Analyse eignen, wie z.B. die innen Temperatur oder der Status des Standlichts, sind ausgenommen. Für die komplette Liste siehe Anhang. Pro Datei sind durchschnittlich 100000 Messwerte vorhanden, daraus ergeben sich bei einer Anzahl von 10359 Dateien über eine Milliarde Messpunkte. Die folgende Liste gibt einen Einblick über die vorhandenen Daten.

- Anzahl Fahrer: 10
- Datengröße (MDF): 3,7 GB
- ∅ Fahrtenanzahl: 1
- Fahrtenanzahl insgesamt: 7
- ∅ Fahrtzeit pro Fahrer pro Fahrt: 13
- ∅ Fahrtzeit pro Fahrer: 14
- Fahrtzeit insgesamt: 87
- ∅ Gefahrene Kilometer: 14
- Gefahrene Kilometer insgesamt: 84
- Zeitraum: 12.09.2019 - 13.08.2019

| Signalname | CAN-Bus | Beschreibung | Einheit | Wertbeschr. |
|---|---|---|---|---|
| LWI_Lenkradwinkel | F-CAN | Lenkradwinkel | Grad | - |
| LWI_VZ_Lenkradwindel | F-CAN | Vorzeichen von Lenkradwinkel | Boolean | 0: Positiv<br>1: Negativ |
| ESP_Fahrer_bremst | F-CAN | Bremspedal betätigt | Boolean | 0: nicht betätigt<br>1: betätigt |
| ESP_Bremsdruck | F-CAN | Bremsdruck | Bar | - |
| MO_Fahrpedalrohwert_01 | F-CAN | Fahrpedalposition | - | - |
| MO_Kuppl_schalter | F-CAN | Kupplung betätigt | Boolean | 0: nicht betätigt<br>1: betätigt |
| MO_Drehzahl_01 | F-CAN | Motordrehzahl | rpm | - |
| KBI_Tankfuellstand_Prozent | F-CAN | Tankfüllstand | % | - |
| KBI_Kilometerstand | F-CAN | Kilometerstand | km | - |
| MO_Gangposition | F-CAN | Gangposition | - | 1: 1. Gang oder Rückwärtsgang<br>2-6: 2. - 6. Gang |
| ESP_VL_Radgeschw_02 | F-CAN | Geschwindigkeit Rad links vorne | km/h | - |
| ESP_VR_Radgeschw_02 | F-CAN | Geschwindigkeit Rad rechts vornen | km/h | - |
| ESP_HL_Radgeschw_02 | F-CAN | Geschwindigkeit Rad links hinten | km/h | - |
| ESP_HR_Radgeschw_02 | F-CAN | Geschwindigkeit Rad rechts hinten | km/h | - |
| ESP_v_Signal | F-CAN | Durchschnittliche Radgeschwindigkeit | km/h | - |
| ESP_HL_Fahrtrichtung | F-CAN | Fahrtrichtung Rad links hinten | - | 0: Vorwärts<br>1: Rückwärts |
| ESP_HL_Fahrtrichtung | F-CAN | Rad links hinten | - | |
| ESP_Laengsbeschl | F-CAN | Längsbeschleunigung | $m/s^2$ | - |
| ESP_Querbeschleinigung | F-CAN | Querbeschleunigung | $m/s^2$ | - |
| ESP_Gierrate | F-CAN | Gierrate | Grad pro Sekunde | - |
| ESP_VZ_Gierrate | F-CAN | Vorzeichen von Gierrate | Boolean | 0: Positiv<br>1: Negativ |
| Wischer_vorne_aktiv | K-CAN | Wischer vorne aktiv | Boolean | 0: nicht aktiv<br>1: aktiv |
| BH_Blinker_li | K-CAN | Blinker links aktiv | Boolean | |
| BH_Blinker_re | K-CAN | Blinker rechts aktiv | Boolean | |

Tabelle 3.1.: Signalbeschreibung

Je nach Steuergerät und Nachrichtentyp ist die Signalhäufigkeit unterschiedlich. Zum Beispiel sendet das Lenkrad- und Motorsteuergerät mit der gleichen Frequenz – nämlich 100-mal in der Sekunde. Andere jedoch, wie das Getriebesteuergerät, nur mit einer Frequenz von 10 Hz. Abbildung 3.1 zeigt Häufigkeit einiger Signale.

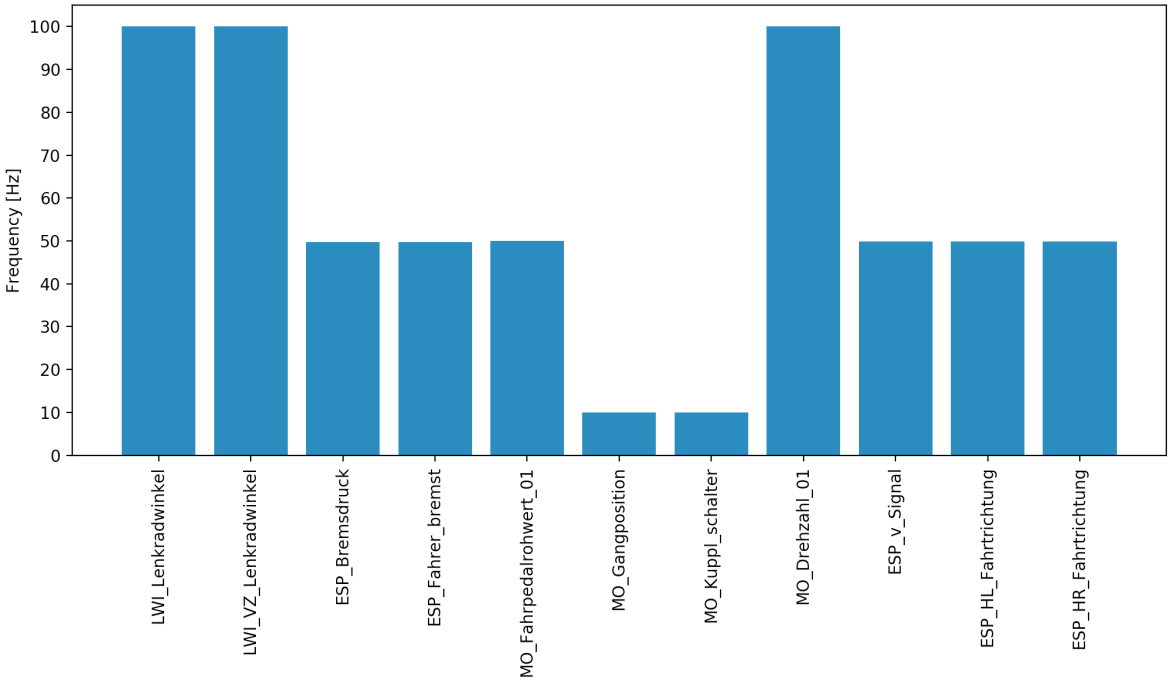


Abbildung 3.1.: Signalhäufigkeiten (Quelle: Author)

## 3.2. Trainingsdaten

Relevante

Data transformation

Feature extraction

## 3.3. Machine Learning Model

Wieso supervised und nicht zb unsupervised? daten sind bereits markiert, kein zeitaufwand notwendig, bessere genauigkeit. trotzdem probieren! ML Model beschreiben, Code snippets

### 3.3.1. Optimierung

Hyper parametering, trade of performance - accuracy

## 3.4. Integration in Auto

ML auf ALEN Box

### 3.4.1. Automotive Linux Edge Node (ALEN)

# 4. Diskussion

Präsentieren der Ergebnisse, Diskussion

# 5. Zusammenfassung

Zusammenfassung

# Literaturverzeichnis


[Con12] Drew Conway. *Machine Learning for Hackers*. O'Reilly Media, Inc, USA, 2012. 3

[ETKK16] Miro Enev, Alex Takakuwa, Karl Koscher, and Tadayoshi Kohno. Automobile driver fingerprinting. *Proceedings on Privacy Enhancing Technologies*, 2016(1):34–50, jan 2016. 4

[fSoAS14] Association for Standardization of Automation and Measuring Systems. Format specification mdf format v. 3.3.1. Standard, Association for Standardization of Automation and Measuring Systems, 2014. 8

[Gar19] Inc. Gartner. Gartner says 5.8 billion enterprise and automotive iot endpoints will be in use in 2020. Technical report, 2019. 1

[GRDW18] Bernhard Gahr, Benjamin Ryder, Andre Dahlinger, and Felix Wortmann. Driver identification via brake pedal signals — a replication and advancement of existing techniques. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, nov 2018. 4

[HSS+16] David Hallac, Abhijit Sharang, Rainer Stahlmann, Andreas Lamprecht, Markus Huber, Martin Roehder, Rok Sosic, and Jure Leskovec. Driver identification using automobile sensor data from a single turn. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, nov 2016. 4

[KEE+15] Konstantina Kourou, Themis P. Exarchos, Konstantinos P. Exarchos, Michalis V. Karamouzis, and Dimitrios I. Fotiadis. Machine learning applications in cancer prognosis and prediction. *Computational and Structural Biotechnology Journal*, 13:8 – 17, 2015. 5

[MNO+07] C. Miyajima, Y. Nishiwaki, K. Ozawa, T. Wakita, K. Itou, K. Takeda, and F. Itakura. Driver modeling based on driving behavior and its evaluation in driver identification. *Proceedings of the IEEE*, 95(2):427–437, Feb 2007. 3

[WOM+05] T. Wakita, K. Ozawa, C. Miyajima, K. Igarashi, K. Itou, K. Takeda, and F. Itakura. Driver identification using driving behavior signals. In *Proceedings. 2005 IEEE Intelligent Transportation Systems, 2005.*, pages 396–401, Sep. 2005. 3

# Abbildungsverzeichnis

# Tabellenverzeichnis

# A. Anhang/Ergänzende Information

EIGENER ANHANG

(Hier können Schaltpläne, Programme usw. eingefügt werden.)