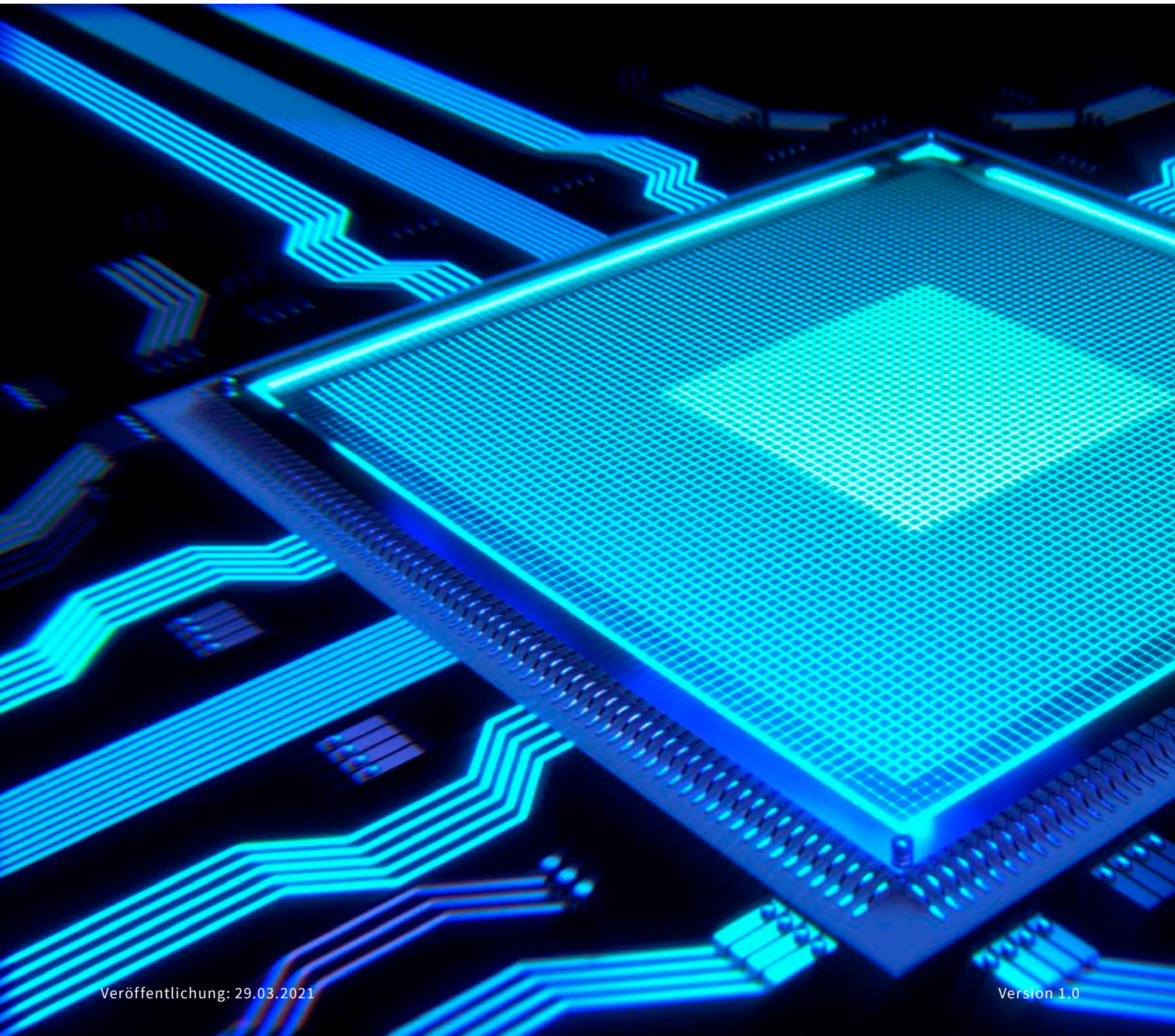
FZI

— Vorgehensmodell zum maschinellen Lernen

Teildisziplin eines KI-Engineerings

Michael Weber, Jens Weber, Patrick Petersen, Nils Schwabe, Alexander Blöck, Atanas Tanev, Carl Philipp Hohl, Lennart Ries, Leon Hielscher, Nico Lambing, Sebastian Reiter



Veröffentlichung: 29.03.2021

Version 1.0

— Inhaltsverzeichnis

— Einleitung

CC-KING
Competence Center
KI-Engineering

Über das Kompetenzzentrum für KI-Engineering (CC-KING)

Viele Unternehmen verfügen über einen wahren Datenschatz, den sie mit Hilfe von Künstlicher Intelligenz (KI) wertschöpfend einsetzen könnten. Gleichzeitig mangelt es jedoch oft an KI-Kompetenz und die Lücke lässt sich schwer schließen, weil KI-Experten rar und zudem mit den typischen Anwendungsdomänen in der Regel nicht vertraut sind.

Deshalb bietet das Kompetenzzentrum für KI-Engineering (CC-KING) Unternehmen konkrete Unterstützung an. Das Kompetenzzentrum forscht in engem Kontakt mit Unternehmen an grundlegenden Fragen, praxistauglichen Methoden und konkreten Anwendungsproblemen aus den Kontexten industrielle Produktion und Mobilität. Bereits existierende Forschungsinitiativen wie die Karlsruher Forschungsfabrik und das Testfeld Autonomes Fahren Baden-Württemberg dienen dabei als Reallabore für die Schwerpunkte.

Das Karlsruher Kompetenzzentrum für KI-Engineering schafft die Verbindung zwischen KI-Spitzenforschung und etablierten Ingenieurdisziplinen. Es erforscht die Grundlagen und entwickelt die Werkzeuge, um den Einsatz von Methoden der Künstlichen Intelligenz (KI) und des maschinellen Lernens (ML) in der betrieblichen Praxis zu erleichtern. Als Anwendungsfelder im Fokus stehen industrielle Produktion und Mobilität.

Das Kompetenzzentrum für KI-Engineering vereint die geballte informationstechnische und ingenieurwissenschaftliche Kompetenz des Standorts Karlsruhe, um den KI-Einsatz in der Praxis entscheidend zu erleichtern: Das Fraunhofer-Institut für Optronik, Systemtechnik und Bildauswertung IOSB (Federführung), das FZI Forschungszentrum Informatik und das Karlsruher Institut für Technologie (KIT) forschen in engem Kontakt mit Unternehmen an grundlegenden Fragen, praxistauglichen Methoden und konkreten Anwendungsproblemen.

CC-KING wird durch das Ministerium für Wirtschaft, Arbeit und Tourismus Baden-Württemberg gefördert.

> **„KI-Engineering ist die systematische Entwicklung und der Betrieb von KI-basierten Lösungen als Teil von Systemen, die komplexe Aufgaben erfüllen.“**
>
> Quelle: CC-KING

Im Folgenden wird ein Vorgehensmodell zum Entwurf eines auf maschinellem Lernen basierten (Teil-)Systems vorgestellt. Schwerpunkt bildet somit der Entwurf der Umsetzung des maschinellen Lernens, welcher als Subsystem in einem übergeordneten System betrieben werden kann. Das Vorgehensmodell stellt folglich eine Teildisziplin des KI-Engineering dar. Das Vorgehen zum Entwurf des Gesamtsystems und somit das Rahmenwerk zu dem hier skizzierten Vorgehen wird im Vorgehensmodell PAISE – Process Model for AI Systems Engineering definiert.

— Entwurfsphasen

Problem Solution Specification

In dieser Projektphase werden funktionale und nicht-funktionale Anforderungen an das KI-basierte System spezifiziert. Dies umfasst sowohl die eigentliche Funktionalität und Gütekriterien als auch Schnittstellen zum Gesamtsystem und die Spezifikation der vorhandenen Ein- und Ausgangsgrößen. Die verfügbaren Daten werden analysiert und die vielversprechendste KI-Methode bzgl. Problemstellung und verfügbaren Daten ausgewählt.

Data Curation

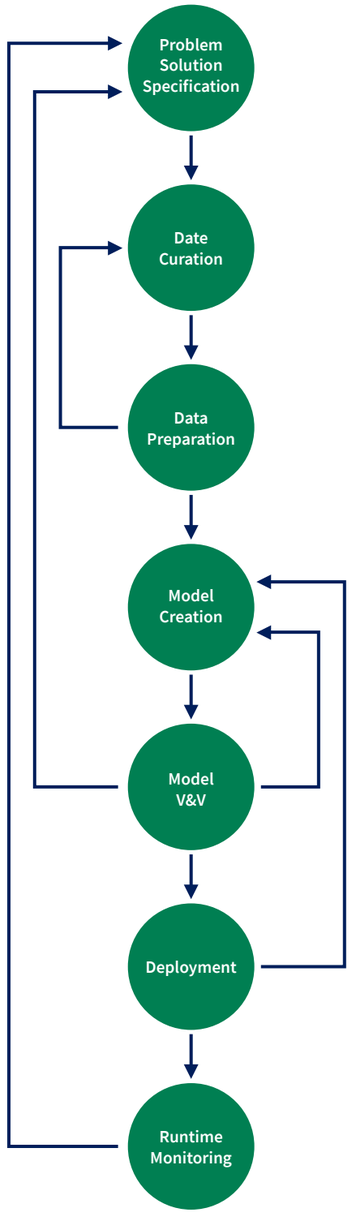
Die Entwicklung von KI-Modellen setzt repräsentative Daten für die jeweils betrachtete Problemstellung voraus, dies sowohl für Training, Validierung und Test. In dieser Projektphase werden Anforderungen an Vielfalt, Menge und Qualität der Daten abgeglichen. Die benötigten Daten werden bereitgestellt, validiert und gegebenenfalls mit Methoden zur Datenaugmentierung erweitert. Neben der Menge spielt die Vielfalt, d. h. die Verteilung im Feature-Raum, eine entscheidende Rolle.

Data Preparation

Die Daten werden für Training, Validierung und Test getrennt aufbereitet. Hierzu werden sie bzgl. der Problem Specification aufbereitet, wie z. B. die Aggregation von Daten, aber auch Rauschentfernung oder Fehlerbehandlung. Insbesondere im Falle von Überwachtem Lernen fällt der Annotation eine essenzielle Rolle zu. In diesem Fall spricht man von der Erzeugung von Ground Truth Daten, da die verfügbaren Daten mit Labels versehen werden und somit die erwartete Ausgabe des KI-Modells festgelegt wird.

Model Creation

Zu Beginn dieser Projektphase wird über die konkrete KI-Architektur entschieden. Dazu werden entsprechende Methoden und Architekturen ausgewählt, meist prototypisch trainiert und bewertet. Nach der Auswahl der KI-Architektur erfolgt das eigentliche Training. Hierbei werden mit Hilfe der Trainings- und Validierungsdaten das KI-Modell sowie die angewendeten Hyperparameter optimiert. Nach dessen Abschluss liegt das eigentliche, adaptierte KI-Modell vor.



Model V&V

In der Data-Preparation-Phase wurden die vorhandenen Daten in Trainings-, Validierungs- und Testdaten aufgeteilt. In dieser Phase werden die Testdaten verwendet, um das trainierte KI-Modell zu testen. Des Weiteren können traditionelle Qualifizierungsmaßnahmen angewendet werden, um das System zu validieren.

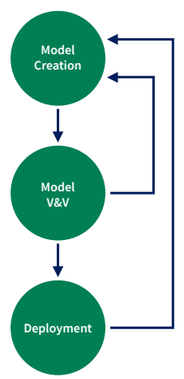
Deployment

In dieser Phase wird das trainierte KI-Modell auf die Zielplattform übertragen und in das übergeordnete System integriert. Insbesondere in der Domäne der eingebetteten Systeme unterscheidet sich die finale Zielplattform von der Host-Plattform, auf welcher das Netz entwickelt wurde. Um eine Ausführung auf Plattformen mit limitierten Ressourcen zu ermöglichen, sind eventuell Anpassungen (z. B. Quantisierung oder Pruning) des Modells nötig. Die abschließende Qualifikation des integrierten ML-Subsystems findet im Rahmen der Verifikation und Validierung des übergeordneten Systems statt.

Runtime Monitoring

Um die Leistungsfähigkeit des Systems nach dem Deployment kontinuierlich zu verbessern, sind insbesondere bei diesen datengetriebenen Systemen eine kontinuierliche Überwachung und Erweiterung der Trainingsdaten vielversprechend. Im regulären Betrieb können unvorhergesehene Situationen auftreten, welche in ein inkrementelles Training des KI-Modells hinzugefügt werden können und somit zu einer kontinuierlichen Verbesserung des Modells führen.

Beispielszenario: Nach erfolgreicher Model V&V eines entwickelten Modells auf dem Hostsystem muss für das Deployment der Speicherabdruck reduziert werden. Hierzu kommen Techniken wie das Pruning zum Einsatz. Dabei wird die KI-Architektur überarbeitet (Rückkante zur Model Creation) und das Netz erneut validiert (Model V&V). Erst nach Erreichen der geforderten Güte und der Anforderungen der Hardware wird das Netz auf die Zielplattform gebracht. Die Schleife kann mehrfach durchlaufen werden.



Problemspezifikation und Datenbereitstellung

Problem Specification
Beschreibung der funktionalen und nicht-funktionalen Anforderungen, der Einsatzbedingungen, der Schnittstellen zum übergeordneten System sowie der angestrebten Gütekriterien.

Data Pre-Analytics
Neben der Problemstellung bilden die verfügbare Daten ein wichtiges Kriterium in der Auswahl der KI-Methode. Hierbei muss überprüft werden, ob sich die Daten zur Erstellung von KI-Modellen eignen bzw. ob andere Verfahren besser geeignet sind, da sie sich mit weniger Aufwand realisieren lassen.

AI Method Selection
Vor der eigentlichen Umsetzung des KI-Modells müssen Entscheidungen bzgl. der zugrundeliegenden KI-Methode getroffen werden. Dies beinhaltet z. B. die grundlegende Entscheidung, ob Techniken wie Random Forrest oder tiefe neuronale Netze angewendet werden sollen, aber auch ob z. B. überwachte oder unüberwachte Verfahren zur Erstellung des KI-Modells eingesetzt werden sollen. Die Entscheidung gründet sowohl auf der Problemstellung als auch auf den verfügbaren Daten.

Feature Selection
Die verfügbaren Daten bestehen aus einer Vielzahl von meist hochdimensionalen Datenpunkten. Jede Dimension dieser Datenpunkte beschreibt ein sogenanntes Feature. Die Gesamtheit der möglichen Ausprägungen aller Features wird als Feature-Raum bezeichnet. Weiterhin beschreibt ein Datenpunkt aus dem Datensatz eine konkrete Ausprägung der Features im Feature-Raum. Im Rahmen der Feature Selection werden die beschreibenden, problemspezifischen Features ausgewählt, die den funktionalen Zusammenhang zwischen den Eingabegrößen in das KI-Modell und seiner Ausgabe, die durch die Ground Truth Daten dargestellt werden, beschreiben.

Data Acquisition
Das Vorhandensein einer ausreichenden Menge der Daten muss zum Training und Test der Modelle sichergestellt werden. Hierzu müssen Quellen für existierende Daten ausgewählt oder neue Daten erzeugt werden. Für einige Aufgaben existieren bereits große, teils öffentlich verfügbare Datensätze, die verwendet werden können, um die Anforderungen an die eigenen Daten zu reduzieren.

Data Analysis
Wurden die für die Lösung der Aufgabe vorgesehenen Daten ausgewählt oder aufgezeichnet, muss eine Analyse dieser Daten stattfinden. Auch muss überprüft werden, ob eine ausreichende Vielfalt der Daten gegeben ist. Für jedes Szenario, dem das fertige Modell potenziell gegenübersteht, müssen entsprechende Daten für Training, Validierung und Test der Modelle verfügbar sein. Alternativ kann der Wertebereich, in dem die KI-Methode sinnvoll arbeiten kann, anhand der Analyse der Trainingsdaten eingeschränkt werden.

Data Augmentation
Die Menge an möglichen Eingaben an das zu entwickelnde System übersteigt die Menge an verfügbaren Trainingsdaten im Normalfall um einige Größenordnungen. Durch die den KI-Methoden inhärenten Approximationseigenschaften können hieraus dennoch leistungsfähige Systeme entwickelt werden. Durch verschiedene Approximationstechniken können zusätzlich Daten für deutlich unterschiedliche, aber verwandte Szenarien generiert werden. Hierbei kann zwischen kleineren Anpassungen der Daten, z. B. dem Hinzufügen kleiner Variationen sowie umfassender Augmentation wie z. B. das Generieren ergänzender Daten mit komplexen virtuellen Modellen unterschieden werden. Hierbei werden zusätzlich Daten für abweichende Szenarien, welche noch nicht durch die Daten abgedeckt sind, erzeugt. Dies steigert die Dichte und Verteilung der Daten im Feature-Raum.

Feature Engineering
Im Rahmen dieser Phase wird entschieden, welche der beschreibenden, problemspezifischen Features für die KI-Methode sinnvollerweise verwendet werden soll. Dazu können sowohl algorithmische als auch heuristische Verfahren zum Einsatz kommen. Außerdem können Features des Datensatzes in andere Repräsentationen überführt werden, um weniger Features mit mehr Aussagekraft zu erzeugen. Die so ausgewählten und modellierten Features dienen später dem KI-Modell als Eingabe.

Data Labeling
Für ein erfolgreiches überwachtes Lernen müssen neben den Daten auch die zu den jeweiligen Daten gewünschten Ausgaben des Modells vorhanden sein. Falls die vorhandenen Daten diese sogenannten Labels noch nicht enthalten, muss eine Annotation der Eingabedaten durchgeführt werden. Je nach Problemstellung, Eingabedaten und gewünschten Ausgaben muss dieser Annotationsprozess entweder manuell durch einen Menschen, semi-automatisch mithilfe von Systemen oder komplett automatisiert durchgeführt werden.

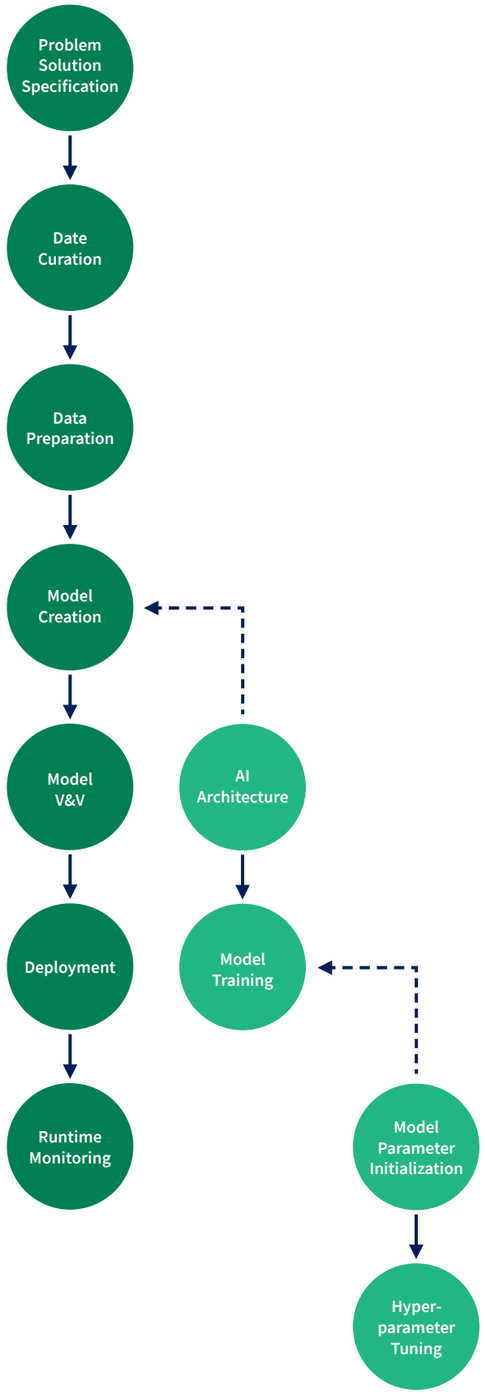
Modellerstellung

AI Architecture Definition

Die Struktur des KI-Systems gliedert sich meist in eine Vorverarbeitung (Pre-Processing) und Nachverarbeitung (Post-Processing) sowie der eigentlichen KI-Architektur. Für die Definition der KI-Architektur stehen unterschiedlichste Architekturen zur Verfügung. Um Daten variabler Länge zu verarbeiten, bieten sich rekurrente Architekturen an, d. h. Netze mit Rückkopplung und somit einer Verknüpfung zwischen Datensamples.

Model Training

Nachdem die KI-Architektur festgelegt wurde, erfolgt im Model Training eine sukzessive Verbesserung der initialen Gewichte bezüglich einer zuvor gewählten Fehlerfunktion, der sogenannten loss function. Jedes Trainingsdatum im (Mini-)Batch wird vom Berechnungsgraphen der KI-Methode durchgerechnet und mithilfe der loss function mit den Ground Truth Daten verglichen. Der Fehler wird über den (Mini-)Batch kumuliert und bildet die Basis für die Optimierung der Parameter des KI-Modells. Neben anderen Optimierungsverfahren kommt insbesondere bei künstlichen neuronalen Netzen die Optimierung mit einem Gradientenabstiegsverfahren zusammen mit dem Back-Propagation Algorithmus zum Einsatz. Dieser Zyklus wird i. d. R. erst abgebrochen, wenn ein zuvor gewähltes Abbruchkriterium erreicht wurde. Das Model Training ist sehr eng verbunden mit dem Hyperparameter Tuning.



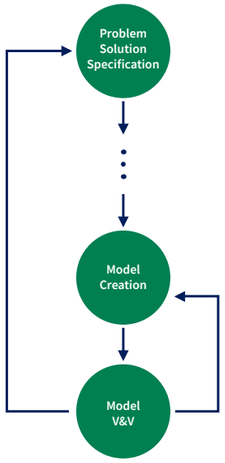
Model Parameter Initialization

Bevor das KI-Modell trainiert werden kann, muss zunächst ein initiales Modell erstellt werden. Dafür werden die trainierbaren Gewichte der zuvor erstellen KI-Architektur mittels einer Initialisierungsfunktion mit Werten belegt. Da im Model Training das Modell nur schrittweise verbessert wird, hat die Model Parameter Initialization auch einen gewissen Einfluss auf die Performance des trainierten Modells. Als Initialisierungsfunktion werden i. d. R. Funktionen gewählt, die eine gewisse Wahrscheinlichkeitsverteilung modellieren. Besonders gut funktionieren Initialisierungsfunktionen wie die Gauß-Verteilung oder die Xavier-Initialisierung. Üblich ist auch die Initialisierung der Gewichte mit den Gewichten ähnlicher Modelle, die auf vergleichbaren Problemen trainiert wurden. Dieser Ansatz kann weitergetrieben werden, indem man vortrainiere KI-Modelle verwendet und diese auf dem eigenen Datensatz nachtrainiert. In diesem Fall spricht man von Transfer Learning.

Hyperparameter Tuning

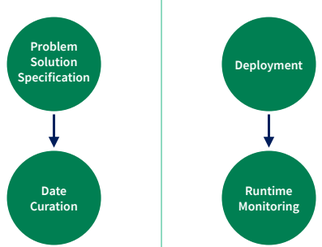
Je nach verwendetem Verfahren besitzen sowohl das KI-Modell als auch das Trainingsverfahren selbst verschiedene Parameter, die vor oder im laufenden Trainingsprozess angepasst werden können. Einfache Beispiele hierfür sind die Größe und die damit einhergehende Anzahl an trainierbaren Trainingsparametern des Modells oder die Anzahl an Trainingszyklen. Um hier nicht von einer initialen potenziell ungünstigen Wahl der Parameter abhängig zu sein, werden Modelle meist mehrfach trainiert und die erreichten Ergebnisse miteinander verglichen. Dieser Vergleich erfolgt meist auf einem zusätzlichen Validierungsdatensatz.

— Vorgehensmodell Maschinelles Lernen

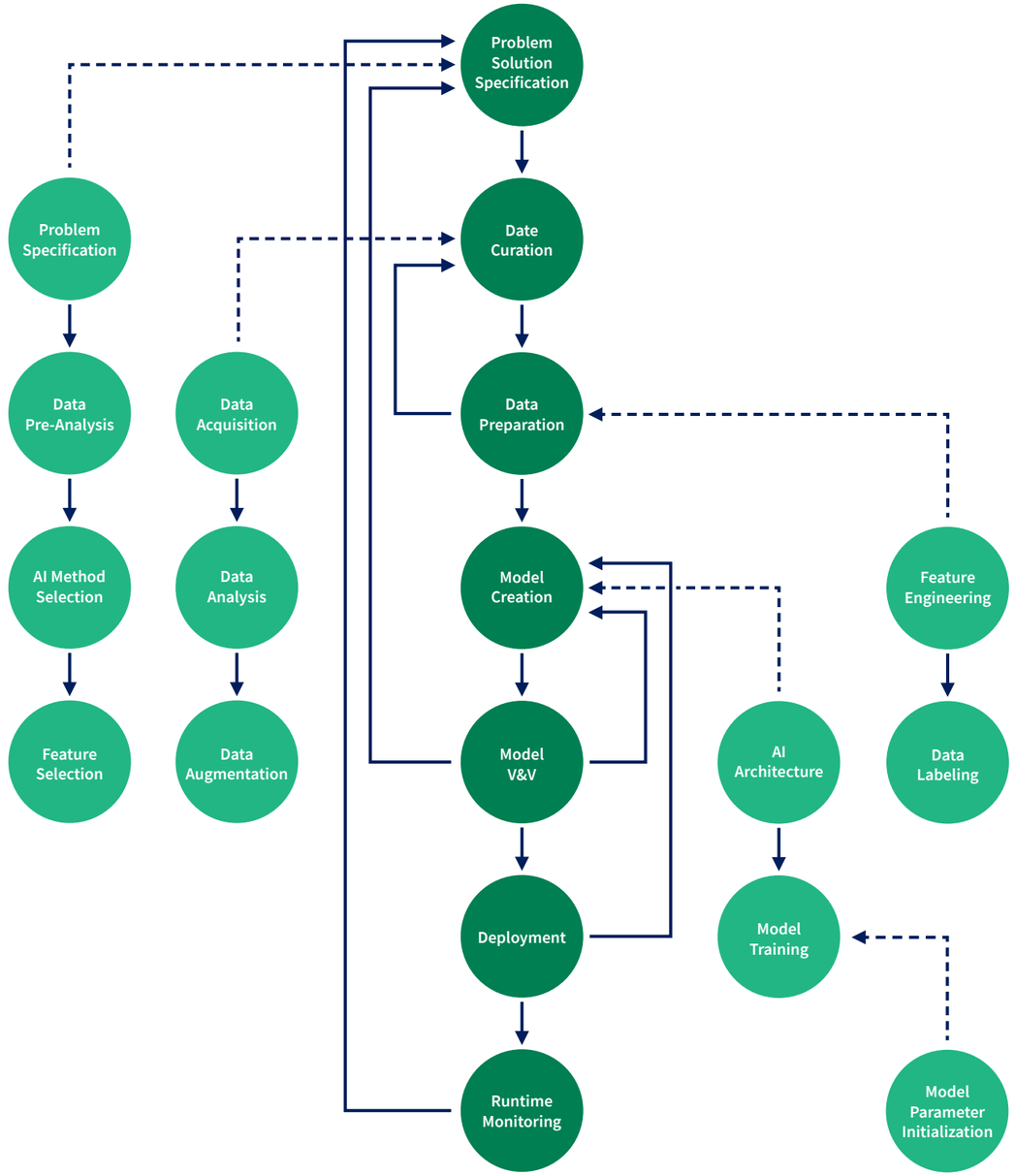


Selbst in der der fortgeschrittenen Phase der Model V&V kann ersichtlich werden, dass die angestrebte Problemlösung mit dem gewählten Feature-Vektor nicht zu realisieren ist. In diesen Fällen ist ein Rücksprung zur Problem Solution Specification vorgesehen, um eine geänderte Feature-Auswahl, ein passendere KI-Methode oder aber auch eine geänderte Zieldefinition vornehmen zu können.

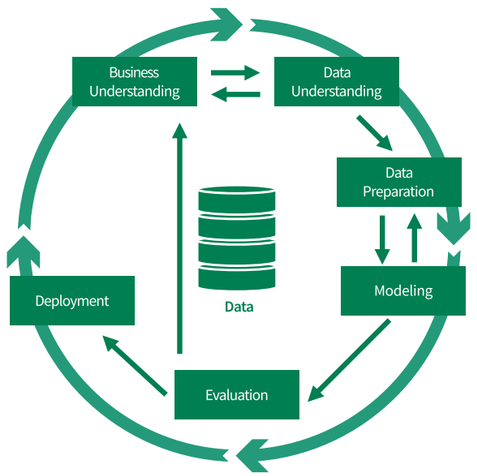
Die Tätigkeiten in den Phasen Data Curation und Data Preparation sind eng miteinander verzahnt. Beispielsweise legt die KI-Methode fest, ob Datenanalysen abhängig oder unabhängig von den gewählten Features durchgeführt werden können. Beim manuellen Labeling bietet es sich häufig an, die Data Augmentation nach dem Data Labeling vorzunehmen, da hier Labels übernommen werden können, bei automatisiertem Labeling ist es hingegen nicht relevant. Aus diesem Grund wurde im Vorgehensmodell eine Rückkante zwischen den Phasen vorgesehen, die ein iteratives, zyklisches Bearbeiten der Phasen ermöglicht.



Selbst nach erfolgreicher Integration des Systems ist es ratsam weiterhin die anfallenden Daten zu überprüfen (Runtime Monitoring). Ziel ist es, eine kontinuierliche Verbesserung des Systems zu erreichen. Insbesondere durch Auftreten von aktuell wenig oder gar nicht beobachteten Fällen kann eine Verbesserung des Systems durch ein erneutes Training erzielt werden. Im Vorgehensmodell spiegelt sich dies durch den Rücksprung in die Problem Solution Specification wider. In dieser Phase werden Datensätze initial auf die Tauglichkeit und in diesem Fall auf das Potenzial zur Verbesserung hin untersucht.



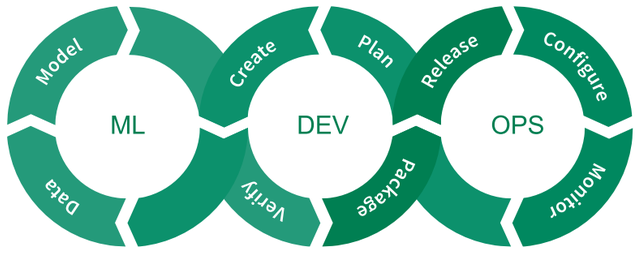
— Eingliederung in existierende Vorgehensmodelle



1

CRISP-DM

Das vorgestellte Vorgehensmodell greift essenzielle Punkte des CRISP-DM auf. Die Phasen Business und Data Understanding sind mit der Phase Problem Solution Specification vergleichbar. Hier geht es darum, sowohl ein Problem- als auch Datenverständnis aufzubauen. Die anschließende Data Preparation Phase findet sich in beiden Modellen. Die Abfolge Modeling, Evaluation und Deployment sind vergleichbar mit den Phasen Model Creation, Model und V&V Deployment.



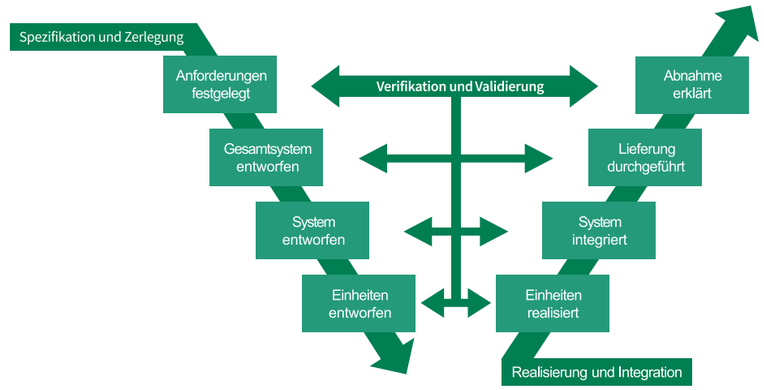
2

MLOPS

Das Vorgehensmodell MLOPS, welches eine ML-spezifische Weiterentwicklung des DevOps-Zyklus darstellt, deckt sich mit dem vorgeschlagenen KI-Entwicklungsvorgehen. So taucht der erste MLOPS-Zyklus zwischen Problem Solution Specification und Data Curation auf, der zweite Zyklus zwischen Model Creation und Model V&V und der dritte Zyklus zwischen Runtime Monitoring und indirekt über die Problem Solution Specification zurück in den Entwurf des KI-Systems.

1 Angelehnt an CRISP-DM Process Diagram

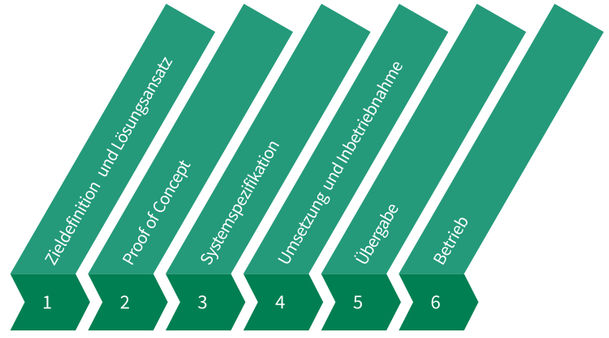
2 Angelehnt an https://blogs.nvidia.com/blog/2020/09/03/what-is-mlops/



3

V-Modell XT

Im V-Modell XT wird auf dem absteigenden Ast das Gesamtsystem bis auf Modulebene dekomponiert und im aufsteigenden Ast realisiert und ins Gesamtsystem integriert. Auf der horizontalen Ebene findet die Verifikation und Validierung statt. Das KI-Entwurfsvorgehensmodell gliedert sich in diesen übergeordneten Systementwurf nahtlos ein, da er lediglich die Entwicklung einzelner KI-gestützter Einheiten beschreibt. Somit erfolgt die Problem Specification, d. h. die Definition der Schnittstellen auf dem absteigenden Ast, die restlichen Phasen finden im Rahmen der Realisierung der ML-spezifischen Einheit statt. Der Model-V&V-Phase kommt eine hybride Rolle zu, da sie sowohl der Entwicklung als auch der Verifikation zugerechnet werden kann.



4

ML4P

Das Vorgehensmodell gliedert die Entwicklung in sechs Phasen mit klar definierten Ergebnissen. Die vorgestellten KI-Entwicklungsschritte lassen sich auf die Phasen des ML4P-Modells abbilden. Die Phase Problem Solution Specification korrespondiert mit der Phase Zieldefinition und Lösungsansatz. Die nachfolgenden Phasen sind in der Proof of Concept Phase angesiedelt, da hier die Umsetzung des KI-Modells vollzogen wird. Erst die Phase Deployment wird in der ML4P-Phase Umsetzung und Inbetriebnahme durchgeführt. Die abschließende Phase Runtime Monitoring korrespondiert mit der Phase Betrieb.

3 Angelehnt an http://www.rechtschreibrat.com/DOX/rfdr_PM_2018-11-16_Geschlechtergerechte_Schreibung.pdf

4 Angelehnt an https://www.iosb.fraunhofer.de/de/projekte-produkte/ml4p-maschinelles-lernen-fuer-produktionsprozesse.html

— Beispiel Ampelerkennung

Die einzelnen Entwurfsphasen sollen nachfolgend anhand eines Systems zur kamerabasierten Ampelerkennung im automatisierten Fahrzeug veranschaulicht werden. Die Beschreibung ist dabei an den Entwurf der Systeme DeepTLR[5] und HDTLR[6] sowie deren Optimierung[7] angelehnt.

Problem Solution Specification

Bei den betrachteten Ampelerkennungssystemen stehen mit den im Versuchsfahrzeug verfügbaren Kamerasensoren die Eingangsgrößen bereits fest. Für den Einsatz in Serienfahrzeugen ist in diesem Schritt auch die Anzahl und Position der Kamerasensoren festzulegen. Ziel ist eine optimale Erfassung der Fahrzeugumgebung, um in den definierten Sichtbereichen Ampeln erkennen zu können. Die Ausgangsgrößen stellen hier die einzelnen vorhandenen Ampeln in Form von 2D Bounding Boxen sowie deren Eigenschaften dar. Als zu erkennende Eigenschaften wurden der Zustand der Ampel sowie das im Leuchtkörper dargestellte Piktogramm festgelegt.

Zu beachten sind in dieser Phase auch die Festlegung einer maximalen Bereitstellungsdauer der Ergebnisse sowie einer minimalen Entfernung, in der Ampeln erkannt werden müssen. Für diesen Funktionsbereich werden dann die für die Erkennung benötigten Gütekriterien festgelegt. Ein weiteres Augenmerk liegt hier in der Wahl der KI-Methode. Für das System zur Ampelerkennung wurde hier ein überwachtes Lernverfahren basierend auf Convolutional Neural Networks (CNN) ausgewählt. CNN wurden gewählt, da diese zum Zeitpunkt der Entwicklung mit Abstand die besten Ergebnisse in Systemen zur bildbasierten Objektdetektionen erzielten.

Data Curation

Da in lernbasierten Bildverarbeitungsalgorithmen üblicherweise große Datenmengen benötigt werden, stellt sich auch für die bildbasierte Ampelerkennung die Frage nach deren Menge und Herkunft. Als Basis für die Datenerzeugung konnte auf eine große Menge bereits vorhandener Rohdaten zurückgegriffen werden. Bei der Analyse dieser Daten wurde festgestellt, dass ausreichend Ampeln in den Rohdaten vorhanden sind und diese auch verschiedenste Licht- und Wetterverhältnisse abdecken. Bei der Verteilung der einzelnen Ampelzustände innerhalb der Rohdaten ließ sich identifizieren, dass einzelne Zustände wie „orange“ oder „rot&orange“ eher selten vertreten sind. Daher wurde hier eine Augmentierung des Datensatzes notwendig, um eine ausreichende Mindestmenge an Trainingsdaten für jeden Zustand zu erreichen. Zu diesem Zweck wurden die Augmentierungstechniken „Spiegelung an der vertikalen Achse“ und „Minimale Drehung des Bildes“ definiert. Diese wurden im Anschluss an das Labeling der Bilder in der Data-Preparation-Phase ausgeführt. Schließlich wurden manuell möglichst vielfältige Szenen aus den vorhandenen Rohdaten ausgewählt und diese in Trainings-, Validierungs- und Testdatensatz aufgeteilt. Da aufgrund der Bildquelle zwischen einzelnen Bilddaten unter Umständen ein zeitlicher Zusammenhang besteht, musste hierbei darauf geachtet werden, dass zeitlich nahe beieinanderliegende Bilddaten nicht in verschiedene Datensätze verteilt werden. Dies würde die Güte der trainierten Modelle auf Validierungs- und Testdatensatz künstlich verbessern.

Data Preparation

Da die verwendeten Bilder aus verschiedenen Aufnahmen stammten, mussten diese zunächst auf ein gemeinsames Bildformat vereinheitlicht werden. Der so vorbereitete Bilddatensatz wurde manuell annotiert und im Anschluss mit den beschriebenen Augmentierungstechniken künstlich erweitert. Hierbei war eine automatische Anpassung der erzeugten Labels nötig, da eine Spiegelung die Richtungsinformationen in den Piktogrammen der Ampeln unter Umständen umkehrt.

Model Creation

Für das System zur Ampelerkennung wurden zunächst mehrere konkrete Architekturen in eine engere Auswahl genommen, nachdem bereits CNN als generelle KI-Methodik ausgewählt wurden. Die verschiedenen Architekturen wurden mit den verfügbaren Trainingsdaten jeweils unter gleichen Voraussetzungen trainiert. Anschließend wurde die Güte und Laufzeit der verschiedenen resultierenden Modelle auf dem Validierungsdatensatz gegeneinander evaluiert und so die beste Architektur identifiziert. Die Initialisierung der Modellgewichte erfolgte anhand verfügbarer Klassifikationsnetze, welche auf dem öffentlich verfügbaren ImageNet Datensatz vortrainiert wurden.

Für die gewählte Architektur wurden mehrere Parameter wie z. B. die Lernrate, die Reduktion der Lernrate während des Trainings, die Verwendung von Dropout oder die Eingabebildgröße identifiziert, welche durch ein Hyperparameter-Tuning bestmöglich gewählt werden konnten. Einzelne Parameter der Architektur wie Filteranzahl oder Filtergröße wurden hierbei nicht berücksichtigt, da für die lernbasierte Bildverarbeitung im Normalfall die Standardbelegung der Parameter bereits eine sehr gute Wahl darstellt. Diese wurden meist von großen Unternehmen wie Google oder Facebook mit entsprechenden Rechenressourcen in zeitaufwändigen Evaluationen ermittelt und eine weitere Optimierung wäre hier lediglich mit großem Rechen- und Zeitaufwand möglich.

Model V&V

Anhand des erzeugten Testdatensatzes wird in dieser Phase die Einhaltung der festgelegten Gütekriterien überprüft. Bei einem System zur Ampelerkennung wird hier die Detektionsgüte mit einem definierten Wert abgeglichen. Auch die Erkennungsgüte des Ampelzustandes und der Piktogramme im Lichtkörper werden in dieser Phase überprüft. Die minimale Erkennungsdistanz und die maximale Laufzeit des Systems werden in dieser Phase ebenfalls ermittelt. Vor allem die Verteilung der Fehlerkennungen des Systems auf die verschiedenen Entfernungsdistanzen müssen hier nochmals konkret betrachtet werden. So kann vermieden werden, dass die festgelegten Gütekriterien für einzelne Entfernungsbereiche unterschritten werden.

5 DeepTLR: A single Deep Convolutional Network for Detection and Classification of Traffic Lights – Weber, Wolf, Zöllner. IV 2016

6 HDTLR: A CNN based Hierarchical Detector for Traffic Lights – Weber, Huber, Zöllner. ITSC 2018

7 Runtime Optimization of a CNN Model for Environment Perception – Weber, Wendenius, Zöllner. IV 2020

Deployment

Da im automatisierten Fahrzeug im Normalfall nicht in gleichem Maße Ausführungseinheiten wie in einem dedizierten Trainingsserver zur Verfügung stehen, müssen die Systeme zur Ampelerkennung angepasst werden. Dabei wird vor allem das Ziel verfolgt, die Laufzeit und den Speicherbedarf des Systems zu reduzieren. Grundsätzlich werden hier zwei verschiedene Ansätze für die verwendeten Modelle verfolgt. Einerseits können einzelne Systeme zur Erledigung einzelner Perzeptionsaufgaben wie der Ampelerkennung mit anderen Perzeptionssystemen zu sogenannten MultiTask-Systemen zusammengelegt und so Synergieeffekte genutzt werden. Andererseits können die verwendeten Modelle mit Techniken wie Pruning und Knowledge-Distillation komprimiert werden. Für die trainierten Ampelerkennungsmodelle wurde auf diese Weise eine Reduktion des Speicherbedarfs um ca. 70 % und der Laufzeit um den Faktor 2.8 bei gleicher Detektionsgüte erreicht.

Runtime Monitoring

Für das automatisierte Fahren ist die Vermeidung von falsch oder nicht erkannten Ampeln eine Hauptvoraussetzung. Daher sind Aufzeichnungen von herausfordernden Szenarien von großer Wichtigkeit. Diese können in den Trainingsdatensatz eingegliedert werden, um so das System kontinuierlich mit jedem Trainingszyklus zu verbessern. Möglichkeiten zum Finden herausfordernder Szenarien können beispielsweise eine Prüfung der zeitlichen und räumlichen Konsistenz sein. Alternativ können Ansätze aus dem aktiven Lernen zum Einsatz kommen.

Originalaufnahme mit annotierten Ampelzuständen aus dem Versuchsträger.

— Glossar

KI-Methode
Die grundlegende Methode, welche das KI-Modell implementieren soll, und die somit notwendigen Schritte für dessen Implementierung. Beispielsweise legt die KI-Methode fest, ob Techniken wie Random Forrest oder tiefe neuronale Netze verwendet werden. Aber auch die Entscheidung, ob überwachte oder unüberwachte Verfahren zur Erstellung des KI-Modells eingesetzt werden.

KI-Architektur
Die Architektur des KI-Modelles legt z. B. bei neuronalen Netzen die Anzahl der Layer, die verwendeten Aktivierungsfunktionen oder Pooling-Funktionen fest. Sie wird im Falle von neuronalen Netzen meist als (untainierter) Berechnungsgraph spezifiziert. Die KI-Architektur wird durch das Training in ein KI-Modell überführt.

KI-Modell
Das finale, trainierte Modell mit allen Gewichten bzw. optimierter Architektur.

(Eingangs-)Feature
Ein Feature ist ein Element eines Feature-Vektors. Ein Feature-Vektor enthält alle Merkmale, die ein zu klassifizierendes Objekt beschreiben und stellt somit die Eingangsgrößen der KI-Architektur dar.

Feature-Raum
Der Feature-Raum spannt alle möglichen Belegungen des Feature-Vektors auf. Für das Training spielt die Abdeckung der verfügbaren Daten dieses Feature-Raums eine wichtige Rolle. Die Abdeckung des Feature-Raums kann mit unterschiedlichen Metriken bewertet werden. Mittels der Dichte (wie viele Datenpunkte gibt es bezüglich der Größe des Feature-Raums) oder der Verteilung (wie sind die Datenpunkte im Feature-Raum verteilt).

Überwachtes Lernen
Beim überwachten Lernen wird neben den Eingabewerten auch der angestrebte Ausgabewert mit angegeben. Basierend auf dieser Relation wird ein Modell trainiert, welches versucht, die Relation vorherzusagen. Die Festlegung der erwarteten Ausgabewerte ist das sogenannte Labeling. Im Rahmen des Labeling wird die Ground Truth für das Training, Validierung und Test erzeugt.

Unüberwachtes Lernen
Beim unüberwachten Lernen werden nur die Eingabewerte zur Verfügung gestellt. Das KI-Modell versucht Muster in den Eingangsdaten zu erkennen. Dieses Verfahren wird z. B. bei der automatisierten Segmentierung von Daten verwendet.

Hyperparameter
Hyperparameter umfassen unter anderem Parametereinstellungen des Lernalgorithmus. Sie nehmen Einfluss auf das Training, sind allerdings nicht direkt im KI-Modell verortet.

Loss Function
Die loss function stellt ein Maß für die Abweichung des vom KI-Modell geschätzten Wertes (Prädiktion) und des wahren, aus den Trainingsdaten bekannten Wertes (Label) dar.

Quantisierung, Pruning
Dies sind Techniken, um trainierte KI-Modelle zu verkleinern. Dabei werden bspw. Gewichte quantisiert, um ihre Repräsentation in Hardware zu verbessern. Daneben gibt es Pruning-Techniken, bei denen Gewichte mit geringem Beitrag entfernt werden, um die Gesamtzahl der vorzuhaltenden Gewichte zu reduzieren.

Rekurente Architekturen
Rekurrente Architekturen sind Netze mit Rückkopplung. D. h. sie stellen eine Verknüpfung zwischen Datenpunkten her und eignen sich somit, Daten variabler Länge zu verarbeiten.

— Impressum

Herausgeber

FZI Forschungszentrum Informatik
Stiftung des bürgerlichen Rechts

Haid-und-Neu-Str. 10 –14
76131 Karlsruhe

+49 721 9654-0
fzi@fzi.de

www.fzi.de