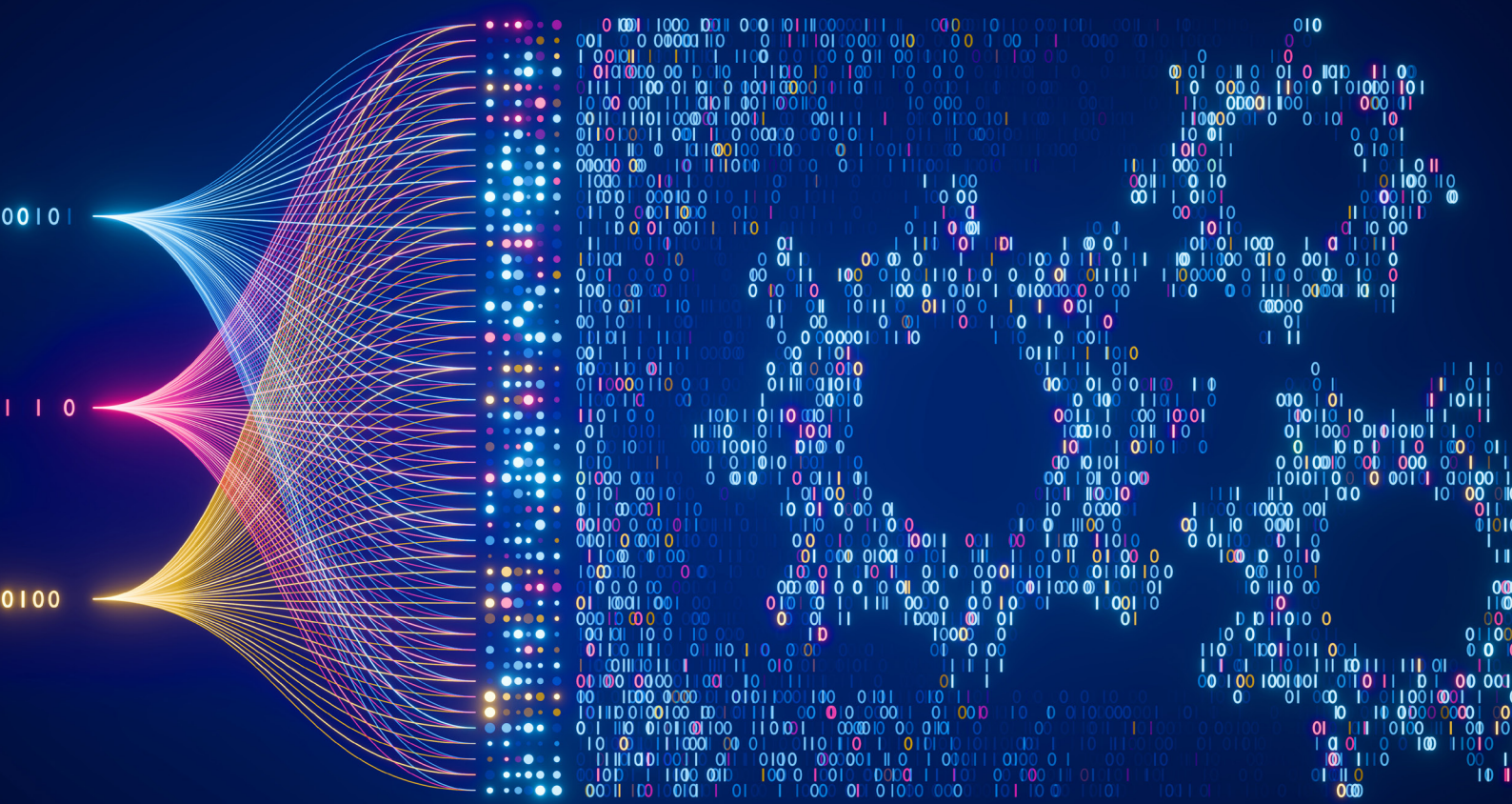
Fraunhofer



Emil Andreas | Christopher Braun | Florian Strohm | Martina Tenzer

Metriken im maschinellen Lernen

Hrsg.: Thomas Bauernhansl, Marco Huber, Werner Kraus

Im Rahmen des

KI Fortschrittszentrum
LERNENDE SYSTEME UND KOGNITIVE ROBOTIK

Vorwort

Künstliche Intelligenz (KI) ist eine der zentralen Technologien für die Zukunft. Ihre Einführung und der Einsatz fordern Unternehmen im besonderen Maß heraus. Es gilt, das Potenzial zu erkennen und dieses wirtschaftlich nutzbar zu machen. Lassen Sie sich dabei von Europas größter Forschungskooperation auf dem Gebiet der KI, Cyber Valley, begleiten.

Mit dem KI-Fortschrittszentrum vom Fraunhofer-Institut für Arbeitswirtschaft und Organisation IAO und Fraunhofer-Institut für Produktionstechnik und Automatisierung IPA und als Teil von S-TEC, dem Stuttgarter Technologie- und Innovationscampus, unterstützen wir Unternehmen dabei, das Potenzial von KI nutzbringend einzusetzen. An der Schnittstelle zwischen anwendungsorientierter Wissenschaft und exzellenter Forschung des Cyber-Valley-Konsortiums entwickeln wir innovative KI-Anwendungen für die Praxis und treiben damit die Kommerzialisierung von KI voran. Erklärtes Ziel ist dabei, menschzentrierte KI-Lösungen zu entwickeln. Denn nur wenn Menschen mit einer neuen Technologie intuitiv interagieren und vertrauensvoll zusammenarbeiten, kann ihr Potenzial optimal ausgeschöpft werden.

Die Studienreihe »Lernende Systeme und Kognitive Robotik« des KI-Fortschrittszentrums gibt Einblick in die Potenziale und die praktischen Einsatzmöglichkeiten von KI. Dabei wurden bereits übergreifende Themen wie Zuverlässigkeit, Erklärbarkeit (xAI), cloudbasierte Plattformen, Technologien und Einführungsstrategien sowie einzelne Anwendungsbereiche vorgestellt. Mit der Fortsetzung der Reihe kommen zahlreiche neue Themen hinzu. Diese reichen von der Bildverarbeitung über die effiziente Software- und Systemintegration in der Robotik, die Gestaltung von KI-Systemen, Sprachmodelle in der Praxis bis hin zu KI-basierten Assistenzsystemen und dem Beitrag von KI zur Fachkräftesicherung.

Im Rahmen der vorliegenden Studie wurde der Einsatz von Metriken zur Qualitätssicherung von KI-Systemen über den gesamten ML Lebenszyklus untersucht, von der Datenaufbereitung über Feature Engineering und Training bis zur Inferenz. Die Studie bietet einen strukturierten Überblick über gebräuchliche Bewertungsmetriken, Tests und Vorgehensweisen und stützt sich auf eine Umfrage unter ML Fachleuten. Die Ergebnisse zeigen eine iterative Arbeitsweise in der Praxis und eine Präferenz für wenige, einfache und gut interpretierbare Fehlermaße, während komplexere Kennzahlen aufgrund von Implementierungsunterschieden und höherem Interpretationsaufwand seltener eingesetzt werden. Durch das kritische Gegenüberstellen von Vor und Nachteilen verschiedener Metriken, ergänzt um praxisnahe Empfehlungen, unterstützt die Studie die Auswahl anwendungsspezifisch geeigneter Verfahren. Sie zeigt, dass die Kombination aus sorgfältiger Datenhygiene, robusten Bewertungsmetriken, gezielten Tests, fokussiertem Tuning und einem belastbaren Inferenz Setup zu vertrauenswürdigen und leistungsfähigen KI Systemen im produktiven Einsatz führt.

Wir wünschen Ihnen eine spannende Lektüre, und freuen uns, wenn wir in Zukunft auch Sie mit unserer Expertise auf Ihrem Weg zur menschzentrierten KI unterstützen dürfen.

Thomas Bauernhansl, Marco Huber, Werner Kraus
Fraunhofer-Institut für Produktionstechnik und Automatisierung IPA

Inhalt

Management Summary

Künstliche Intelligenz ist inzwischen fest im Alltag und in vielen Produkten präsent. Gleichzeitig steigen die Anforderungen an Entwicklerinnen und Entwickler nicht nur technisch, sondern auch regulatorisch. Neue gesetzliche Vorgaben, Leitlinien und branchenspezifische Normen wie die europäische KI-Verordnung oder die europäische Medizinprodukteverordnung schaffen Orientierung, führen aber auch zu Überschneidungen und Unsicherheiten, die in der Praxis gelöst werden müssen.

Die Qualitätssicherung von KI-Systemen, insbesondere solcher auf Basis von Maschinellem Lernen (ML), ist über den gesamten Entwicklungszyklus entscheidend. Ein reines Bewertungsmodell am Ende des Prozesses reicht nicht aus, Qualität muss in allen Phasen von der Datenaufbereitung über das Feature Engineering und das Modelltraining bis zur Inferenz kontinuierlich geprüft werden. Hierzu werden Metriken, Tests und Vorgehensweisen eingesetzt, die sich an den spezifischen Anforderungen der Anwendung orientieren.

Die vorgestellte Studie basiert auf einer Umfrage unter ML-Fachleuten und untersucht, welche Metriken und Vorgehensweisen in der Praxis genutzt werden. Die Ergebnisse zeigen, dass viele Entwickler den Zyklus nicht strikt trennen, sondern iterativ arbeiten. Praktiker bevorzugen einfache, etablierte und gut interpretierbare Fehlermaße, da komplexere Metriken Implementierungsunterschiede mit sich bringen und den Interpretationsaufwand erhöhen. Die Studie zeigt, dass eine effektive Qualitätssicherung auf wenigen, gut gewählten Metriken, gezielten Tests und praxisnahen Vorgehensweisen basiert. Eine Kombination aus sorgfältiger Datenhygiene, robusten Bewertungsmetriken, gezieltem Tuning und belastbarem Inferenz-Setup führt zu vertrauenswürdigen und leistungsfähigen KI-Systemen im produktiven Einsatz.

1. Einleitung und Motivation

Künstliche Intelligenz (KI) ist längst nicht mehr nur ein Forschungsthema, sondern in vielen Produkten und Anwendungen des täglichen Lebens präsent. Spätestens seit der breiten Verfügbarkeit von Systemen wie ChatGPT hat das Thema große Aufmerksamkeit in der Öffentlichkeit erlangt. Für Entwickler und Entwicklerinnen sowie Betreiber von KI-Systemen ist die Arbeit mit diesen Technologien jedoch nicht nur technisch, sondern auch regulatorisch anspruchsvoller geworden. Neue gesetzliche Vorgaben, Leitlinien und branchenspezifische Normen schaffen einerseits Orientierung, führen andererseits aber zu Überschneidungen, Lücken und Interpretationsspielräumen, die in der Praxis gelöst werden müssen.

Die Qualitätssicherung von KI-Systemen, insbesondere solchen, die auf Maschinellem Lernen (ML) basieren, ist eine zentrale Aufgabe über den gesamten Entwicklungsprozess hinweg – von der Datenaufbereitung über das Training und die Validierung bis hin zum produktiven Einsatz und schließlich zur Außerbetriebnahme. Um nicht nur bestehendes Fehlverhalten zu erkennen, sondern es möglichst früh zu verhindern, reicht es nicht aus, ausschließlich das fertige Modell zu bewerten. Entscheidend ist, in jeder Phase des Entwicklungsprozesses mit geeigneten Maßnahmen – etwa durch gezielte Tests, aussagekräftige Metriken oder unabhängige Gutachten – die Qualität des Systems kontinuierlich zu prüfen und zu sichern. Während sich die Aufmerksamkeit in der Vergangenheit oft auf wenige Leistungskennzahlen wie die Vorhersagegenauigkeit beschränkte, steht heute der gesamte Entwicklungslebenszyklus im Fokus, um Systeme zu schaffen, die nicht nur leistungsstark, sondern auch vertrauenswürdig sind.

Klare Leitlinien sind hierfür unerlässlich. Auf europäischer Ebene bildet die KI-Verordnung (KI-VO) den zentralen Rahmen, um sowohl den Binnenmarkt zu stärken als auch vertrauenswürdige KI-Systeme zu fördern. Besonders für Hochrisiko-KI-Systeme stellt die Umsetzung dieser strengen, teils aber unklaren Vorgaben eine erhebliche Herausforderung dar. Vor der Inbetriebnahme ist eine aufwendige Qualifizierung erforderlich, um die Einhaltung aller Anforderungen nachzuweisen. Zusätzlich greifen teilweise branchenspezifische Vorschriften: In der Industrie ist etwa die Maschinenverordnung relevant, ebenso wie Anforderungen an die Prüfmitteleignung, die belegen müssen, dass Mess- und Prüfsysteme präzise, stabil und reproduzierbar arbeiten. Im medizinischen Bereich gelten darüber hinaus strenge Regelungen wie die europäische Medizinprodukteverordnung (Medical Device Regulation, MDR) oder branchenspezifische Normen wie ISO 13485 für Qualitätsmanagementsysteme. Diese Vielzahl an teils überlappenden Vorgaben erhöht die Komplexität für Entwickler erheblich und macht eine strukturierte, nachvollziehbare Qualitätssicherung unverzichtbar.

Ein zentrales Instrument der Qualitätssicherung sind sogenannte Metriken – messbare Größen, mit denen sich Eigenschaften eines ML-Modells quantifizieren lassen. Beispiele sind klassische Leistungsmetriken wie Genauigkeit, aber auch Robustheits-metriken, die prüfen, wie stabil ein KI-Modell bei veränderten Eingaben arbeitet, oder Fairness-Metriken, die etwa untersuchen, ob bestimmte Personengruppen systematisch benach-teiligt werden. In der Praxis gibt es bereits

eine Vielzahl bewährter Verfahren und Bewertungsstrategien. Allerdings sind diese oft auf einzelne Teilbereiche des ML beschränkt und orientieren sich an einem engen Satz von Standardmetriken. Diese Fragmentierung erschwert es, für unterschiedliche oder neuartige KI-Anwendungen die jeweils geeigneten Metriken auszuwählen.

Vor diesem Hintergrund untersucht die vorliegende Studie, welche Metriken derzeit in der Praxis eingesetzt werden, um über den gesamten Entwicklungszyklus hinweg die Qualität von KI-Systemen zu bewerten und so vertrauenswürdige Systeme zu fördern. Grundlage ist eine Umfrage unter Fachleuten, die Einblicke in gängige Vorgehensweisen und Auswahlkriterien bietet.

Ziel ist es, nicht nur einen Überblick über etablierte Metriken zu geben, sondern auch bisher wenig beachtete, aber potenziell wertvolle Ansätze vorzustellen. Darüber hinaus sollen bestehende Lücken identifiziert und der Bedarf für neue oder angepasste Metriken abgeleitet werden, um den Herausforderungen zwischen technologischem Fortschritt, regulatorischen Anforderungen und praktischer Umsetzbarkeit wirksam begegnen zu können.

Die Studie ist folgendermaßen aufgebaut: Sie beginnt mit den Grundlagen, erläutert den Unterschied zwischen KI-Systemen und ihrer ML-Komponente und stellt den KI-Entwicklungszyklus mit seinen Phasen dar. Ein fiktives Anwendungsbeispiel dient dazu, Metriken, Tests und Vorgehensweisen anschaulich und praxisnah zu verorten. Die Ergebnisse basieren auf einer Umfrage unter ML-Experten und -Expertinnen, werden entlang des Entwicklungszyklus phasenweise präsentiert und bei Bedarf um weitere relevante Ansätze ergänzt, während vorab die Anforderungen in Bezug auf das Anwendungsbeispiel für die jeweilige Phase beleuchtet werden. Abgeschlossen wird die Studie mit einem Resümee, das die zentralen Erkenntnisse für die Qualitätssicherung von KI-Systemen zusammenführt.

2. Grundlagen

2.1. KI-System

Ein KI-System ist in der Regel nicht als isolierte Komponente zu verstehen, sondern Teil einer größeren Systemarchitektur (siehe Abbildung 1). Über standardisierte Schnittstellen erhält es Daten aus vorgelagerten Modulen, die beispielsweise aus Sensoren, Datenbanken oder externen Anwendungen stammen können. Diese Eingangsdaten werden in einem Inferenzschritt verarbeitet, bei dem ein zuvor trainiertes Modell – repräsentiert durch die ML-Komponente – auf neue Informationen angewendet wird, um Klassifikationen, Vorhersagen oder Entscheidungen abzuleiten. Die dabei erzeugten Ergebnisse können anschließend von nachgelagerten Softwarekomponenten weiterverarbeitet oder unmittelbar an Folgesysteme bzw. Endnutzer übergeben werden. Auf diese Weise wird die KI in bestehende Abläufe eingebettet und trägt zu deren Automatisierung und Optimierung bei.

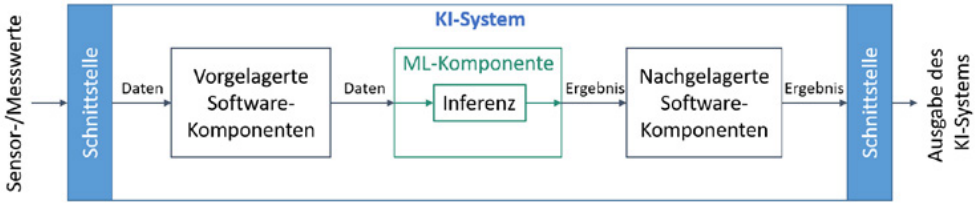


Abbildung 1: Schematische Darstellung eines KI-Systems, welches neben der ML-Komponente vor- und nachgelagerte Software-Komponenten umfasst, etwa für die Datenvorverarbeitung oder Ergebnisnachbereitung. Quelle: Fraunhofer IPA

Damit ein solches System entwickelt und zuverlässig betrieben werden kann, ist ein mehrstufiger Prozess notwendig. Der sogenannte KI-Entwicklungszyklus umfasst verschiedene Phasen. Jede Phase bringt spezifische Aufgaben und Herausforderungen mit sich, die systematisch aufeinander aufbauen. Eine detaillierte Betrachtung dieser Phasen erfolgt im folgenden Kapitel, in dem der Entwicklungszyklus Schritt für Schritt erläutert wird.

2.2. KI-Entwicklungszyklus

In der Regel wird der KI-Entwicklungszyklus in sieben Phasen eingeteilt, wobei eine vorgelagerte Phase – KI-Scoping, also die Festlegung von Anforderungen an das KI-System sowie dessen Einschränkungen – für gewöhnlich zusätzlich berücksichtigt wird (siehe Abbildung 2). Die Phasen werden dabei nicht streng linear durchlaufen. Basierend auf Metriken oder Testergebnissen erfolgt oftmals ein Rückgriff auf vorhergehende Phasen oder ein mehrfaches Durchlaufen des gesamten Zyklus.

Da die Umfrage entlang dieser Phasen ausgelegt wurde und sich die Darlegung der Studienergebnisse ebenfalls daran orientiert, werden hier im Folgenden ebendiese Phasen erläutert.

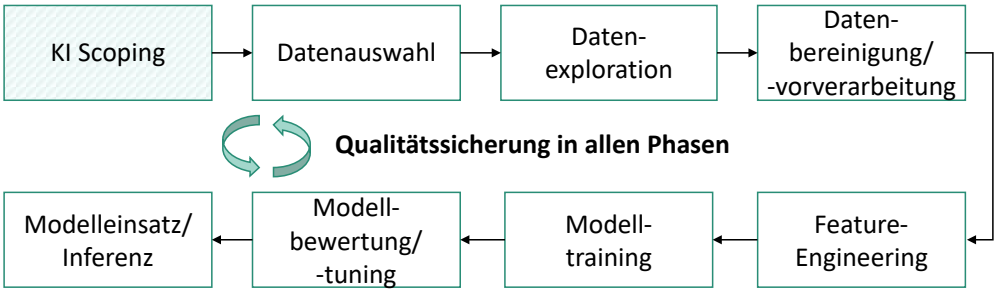


Abbildung 2: Phasen der Entwicklung einer ML-Komponente. Das Durchlaufen der Phasen ist nicht zwingend, wie dargestellt, rein sequenziell, sondern vielfach auch zyklisch und iterativ. Quelle: Fraunhofer IPA, angelehnt an [15, 17]

- KI-Scoping: Umfasst die Analyse des Problems, das mithilfe von ML gelöst werden soll und somit die Klarstellung der zugrundeliegenden, potenziellen Eingangsgrößen und die gewünschten Zielvariablen. Im Rahmen der Studie wurde diese Phase nicht untersucht, da wir davon ausgehen, dass das vorliegende Problem bereits für eine ML-basierte Lösung geeignet ist.
- Datenauswahl: Umfasst die Auswahl der tatsächlichen Eingangs- und Zielgrößen des KI-Systems, z. B. aus Datenbanken oder Sensorwahl. Sie umfasst dabei Aspekte der Aktualität, Repräsentativität, Richtigkeit und ggf. Anonymität der Daten bezüglich des realen zu lösenden Problems.
- Datenexploration: Beinhaltet hauptsächlich die Auswertung der Daten bzgl. der statistischen Kenngrößen sowie Untersuchung auf Vollständigkeit bei z. B. tabellarischen Daten und Visualisierungen der Rohdaten.
- Feature-Engineering: Hier erfolgt die Erstellung neuer, informativer Datenmerkmale (engl. Features) – eine messbare Eigenschaft einer Beobachtung, die als Eingangsgröße für das ML-Verfahren genutzt werden kann. Als Beispiel kann ein zeitdiskretes Signal in seine Frequenzanteile transformiert werden, die dann als Eingabe für ein ML-Modell dienen.
- Modelltraining: Je nach externen Vorgaben und Zielfunktionen eigenen sich gewisse Klassen von Lernalgorithmen besser als andere für ein konkretes Problem. Je nach Algorithmus müssen unterschiedliche Aspekte während des Trainings berücksichtigt werden. Diese Phase umfasst ebenjene Abwägungsaspekte sowie die tatsächliche Ausführung des Trainings.
- Modellbewertung/-tuning: Befasst sich mit der Optimierung des ML-Algorithmus, um dessen Performanz zu steigern, was die Parameterselektion des Modells für die Trainingsprozedur beinhaltet. Grundsätzlich sind Modelltraining und Modellbewertung ein iterativer Prozess, sodass die Grenze zwischen diesen beiden Phasen häufig im Entwicklungsprozess an Trennschärfe verliert. Nach abgeschlossener Optimierung gibt es außer der Performanz noch weitere Gütekriterien, die es ermöglichen, zwischen verschiedenen Modellen abzuwägen.

Dabei geht es insbesondere um die Robustheit, Erklärbarkeit und Vermeidung von Überanpassung des Modells
- Modelleinsatz/-inferenz: Die letzte Phase berücksichtigt den Einsatz des Modells innerhalb des KI-Systems und der damit verbundenen Anforderungen wie Inferenzzeiten, Latenzzeiten, Schnittstellen, Rechenkapazitäten oder Stromverbrauch.

2.3. Fiktives Anwendungsbeispiel

In diesem Abschnitt wird ein Anwendungsbeispiel vorgestellt, das im nächsten Abschnitt dazu dient, sukzessiv alle Phasen des KI-Entwicklungszyklus praxisnah zu veranschaulichen.

Ein fiktives Unternehmen aus dem produzierenden Gewerbe stellt Lochscheiben her und möchte künftig seine optische Qualitätsinspektion digitalisieren. Ziel ist es, Personalkosten zu reduzieren und die Gesamtanlageneffektivität (engl. Overall-Equipment-Effectiveness (OEE) zu steigern). Dafür soll ein KI-System eingesetzt werden, das vier definierte Fehlerklassen – Kantenausbruch, Beule, Kratzer und Delle (siehe Abbildung 3) – zuverlässig erkennt. Dies soll sicherstellen, dass Kunden ausschließlich fehlerfreie Lochscheiben erhalten.

Die für die Anwendung notwendige Hardware ist bereits vorhanden. Insbesondere das Kamerasystem einschließlich Beleuchtung wurde in den Produktionsprozess integriert, um kontinuierlich Bilddaten der gefertigten Werkstücke aufzuzeichnen (siehe Abbildung 4). Diese Daten bilden die Grundlage für den späteren Einsatz der KI. Um eine spätere Modellierung zu ermöglichen, wurden die aufgenommenen Bilder mit den entsprechenden Zielvariablen, d. h. den jeweiligen Fehlerklassen, annotiert. Damit liegt ein Bildklassifizierungsproblem im Rahmen des überwachten Lernens vor. Ein Modell trainiert also auf den Eingabedaten und gleicht stetig seine eigene Vorhersage mit den tatsächlichen definierten Zielvariablen ab, um aus dem Fehler zwischen Vorhersage und Wahrheit zu lernen. Für den praktischen Einsatz ist es entscheidend, dass mindestens 99,5 Prozent der fehlerhaften Lochscheiben korrekt detektiert, werden und die die Fehlalarmquote unter 0,75 Prozent bleibt.

Abbildung 3: Aufbau des fiktiven Anwendungsbeispiel in der optischen Qualitätsinspektion
Quelle: Fraunhofer IPA

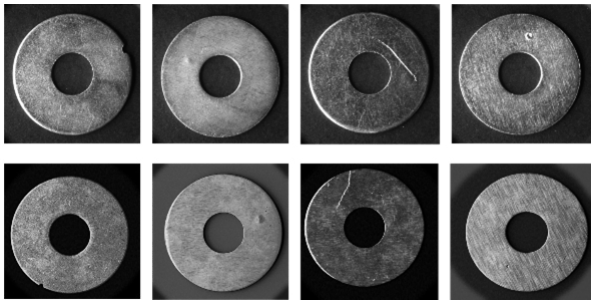


Abbildung 4: Vier Fehlerklassen der Lochscheiben im beispielhaften Anwendungsfall. Von links nach rechts: Kantenausbrüche, Beule, Kratzer und Delle.
Quelle: Fraunhofer IPA

3. Studienergebnisse

Die zugrundeliegende Umfrage hatte zum Ziel, praktische Einblicke in die Qualitätssicherung von ML-Systemen entlang ihres gesamten Entwicklungszyklus zu gewinnen. Befragt wurden primär Fachleute aus der angewandten Forschung mit Erfahrung in verschiedenen ML-Subdisziplinen. Die Studie wurde mittels Microsoft Forms vom 20.06.25 bis zum 24.07.25 durchgeführt. Insgesamt sind Antworten von 22 Fachleuten aufgenommen worden, welche per E-Mail oder Microsoft Teams angefragt wurden. Vorwiegend Mitarbeiter des Fraunhofer IPA und der Universität Stuttgart haben an der Studie teilgenommen, wobei auch KI-Entwickler der Robert Bosch GmbH und Siemens Digital Industries Software befragt wurden. 17 der Teilnehmenden ordnen ihre berufliche Tätigkeit der angewandten Forschung im Bereich des Maschinellen Lernens zu, zwei bzw. drei der Teilnehmer machen Grundlagenforschung bzw. Produktentwicklung im Bereich des Maschinellen Lernens. Fünf Teilnehmer gaben an weniger als 2 Jahre Berufserfahrung zu haben. Auf einer Skala von 0 bis 5 Tagen pro Woche gaben die Teilnehmer an im Durschnitt an 2,86 Tagen pro Woche praktische Implementierungsaufgaben im Bereich des Maschinellen Lernens durchzuführen, wobei kein Teilnehmer 0 Tage angab. Die durchschnittliche Bearbeitungszeit betrug 37 Minuten.

Dabei wurde explizit nach Metriken, Vorgehensweisen und Tests gefragt, die sie in den einzelnen Phasen zur Sicherstellung von Qualität einsetzen. Für jede Frage konnten die Teilnehmer beliebig viele Antworten geben.

Aus den Umfrageergebnissen lässt sich ableiten, dass Entwickler in der Praxis nicht streng zwischen einigen aufeinander folgenden Phasen trennen. So gaben 86 Prozent der Befragten an, den KI-Entwicklungszyklus zu Beginn der Studie zu kennen. Einige Befragte verweisen jedoch sowohl implizit als auch explizit auf die iterative Natur des KI-Entwicklungszyklus, weswegen sich die Phasen schwer voneinander trennen ließen. Dies führte dazu, dass bei einigen aufeinanderfolgenden Phasen teilweise gleiche Antworten gegeben wurden. Basierend auf der Studienauswertung wurde daher entschieden, den Entwicklungszyklus auf vier Phasen zu verdichten: Datenzentrierte Phase, Feature Engineering, modellzentrierte Phase und Inferenz.

Abbildung 5: Aggregierte Darstellung des Entwicklungszyklus auf Grundlage der Studienergebnisse. Quelle: Eigene Darstellung.

Die Antworten ließen sich zunächst in drei unterschiedliche Kategorien unterteilen:

- Metriken: konkrete numerische Werte, die durch eine Formel definiert werden.
- Tests: charakterisiert durch typischerweise mehrere Ausgabewerte, die ein Algorithmus berechnet.
- Vorgehensweisen: Daten- sowie anwendungsfallspezifische Prozeduren, die auf höherer Abstraktionsebene vereinheitlicht werden können.

Beim Lochscheibenanwendungsfall ist eine Metrik bei der Datenexploration der Mittelwert der die Pixelintensität eines oder mehrerer Bilder. Ein Test hingegen ist, die Pixelintensitäten eines Bilds in ein Histogramm zu überführen und dieses mit dem Soll-Histogramm eines Referenzbilds zu vergleichen. Mithilfe eines statistischen Tests (bspw. Chi-Quadrat-Test) lässt sich dabei automatisiert feststellen, ob die Verteilung der Pixelwerte innerhalb eines erwarteten Wertebereichs liegt oder statistisch signifikant davon abweicht. Hier gibt es zwei mögliche Vorgehensweisen: die Erstellung von Visualisierungen oder das Aufteilen des Datensatzes in Trainings- und Testdatensatz. Diese Vorgehensweise besteht aus individuellen Schritten, die zwar auf abstrakter Ebene standardisiert sein können, aber oftmals praktisch für den vorliegenden Anwendungsfall angepasst werden müssen.

In jeder dieser Phasen wird im Folgenden zunächst auf das fiktive Anwendungsbeispiel verwiesen und anschließend auf die Studienergebnisse. Bei Bedarf ist die Phase durch Anmerkungen der Autoren ergänzt worden. Die hier dargestellten Tabellen sind eine, der Übersichtlichkeit wegen, gekürzte Version der Antworten aus der Umfrage. Nennungen, die seltener als der Median der aggregierten Phase sind, wurden nicht abgebildet. Alle Zweifachnennungen sind jedoch erhalten.

3.1. Datenzentrierte Phase

Die datenzentrierte Phase beschäftigt sich mit der Datenauswahl, der Datenexploration, der Datenbereinigung und -vorverarbeitung.

Datenauswahl
Durch die betriebsbedingten Vorgaben steht eindeutig fest, dass bei dem Anwendungsfall Bilddaten verarbeitet werden. Das Ziel besteht darin, die Bilder der Lochscheiben zu klassifizieren. Dies ist eine wichtige Beobachtung, da verschiedene Eingabeformate wie Bilder, Texte oder Tabellen unterschiedliche Anforderungen mit sich bringen und jeweils spezifische Modellarchitekturen und Evaluationsmethoden erfordern. Auch können andere Tests durchgeführt werden, wobei es aber auch Metriken und Vorgehensweisen gibt, die unabhängig von der Art des Eingabeformats sind.

Auch ist wichtig, im allerersten Schritt zu evaluieren, welches Ziel das ML-Verfahren verfolgen soll. In dem Beispiel des fiktiven Unternehmens stellt sich die Frage, ob es sich um eine binäre Klassifikation »In Ordnung« (IO) oder »Nicht In Ordnung« (NIO) handelt oder ob die Art des Fehlers bestimmt werden soll (Mehrklassen-Klassifikation). So ist es denkbar, dass Lochscheiben mit Kratzer doch noch weiterverkauft werden können im Gegensatz zu Lochscheiben mit Ausbrüchen oder Beulen. Aus dem Grund entscheidet sich das Unternehmen für ein System für Mehrklassen-Klassifikation. Häufig soll in einem solchen Szenario überwachtes Lernen eingesetzt werden, für das jedes Bild seiner korrekten Klasse zugewiesen wird. Dafür wird oftmals ein durch den Menschen annotierter Datensatz erstellt. Alternativ können auch synthetische Daten durch den

Einsatz generativer KI-Methode oder Renderingverfahren genutzt werden [5, 11]. Während der Trainingsphase vergleicht das Modell seine Vorhersage mit der richtigen Annotation und korrigiert seine internen Parameter so, dass über die Zeit hinweg mehr und mehr richtige Vorhersagen getroffen werden. Auf einem Test- oder Validierungsdatensatz kann in regelmäßigen Abständen automatisch untersucht werden, wie das Modell auf Daten abschneidet, auf denen es nicht trainiert wurde. Folgende weitere Aspekte können relevant sein:

- Realbedingungen: Entsprechen die aufgenommenen Daten den Realbedingungen bzw. sind diese noch aktuell?
- Datenformate: Sind die Datenformate einheitlich über die Gesamtheit des Datensatzes?
- Klassenverhältnis: Wenn ein eigener Datensatz erstellt wird, in welchem Verhältnis stehen die Klassen zueinander? Für das fiktive Beispiel kann es gut sein, dass es weniger Bilder von NIO-Klassen als IO-Bilder geben wird, was bei weiteren Schritten berücksichtigt werden muss. Zudem muss das Verhältnis der NIO-Klassen beachtet werden, da in der Praxis unterschiedliche Fehler selten in einem ähnlichen Verhältnis auftreten.

Datenexploration

Da es sich bei dem fiktiven Anwendungsbeispiel um Bilddaten handelt, ist die statistische Analyse der einzelnen Merkmale (hier: Pixel) weniger üblich. Trotzdem könnte nach Anomalien bzw. Ausreißern gesucht werden, um dadurch Fehler bei der Datenaufnahme wie bspw. Über- oder Unterbeleuchtung oder unvollständige Aufnahmen der Scheiben auszuschließen. In der ML-basierten Qualitätsprüfung ist ein gängiges Problem das Auftreten neuer, nicht im Fehlerkatalog enthaltener Fehler, was durch ebenjene Maßnahmen schneller detektiert werden könnte:

- Pixelwerte: Was sind Mindest- und Maximaldurchschnitts-Pixelwerte über den gesamten Datensatz?
- Helligkeit: Lassen sich die Lochscheiben auf den hellsten und dunkelsten Bildern noch erkennen?
- Pixel-Intensitätsskala: In welchem Wertebereich liegen die Pixelwerte?

Datenbereinigung und Datenvorverarbeitung

Eine ML-Komponente kann im Grunde nie direkt die Daten so verarbeiten, wie sie aufgenommen werden. Deshalb ist der weitere Schritt der Datenvorverarbeitung essenziell, um die Daten für das Modell vorteilhaft zu transformieren (s. Abbildung 1). Bei Bilddaten, wie aus dem fiktiven Beispiel, ist es typischerweise so, dass die Pixel in einen Wertebereich gebracht werden müssen, mit dem die Modellarchitektur umgehen kann. Dies geschieht beispielsweise durch Normalisierung der einzelnen Pixel. Aber nicht nur die einzelnen Bilddaten müssen vorverarbeitet werden – in einem überwachten Lernszenario muss das Verhältnis von Trainingsdaten zu Validierungs- und Testdaten stimmen. In Bezug auf das fiktive Anwendungsbeispiel sollte somit auf folgende Punkte geachtet werden:

- Klassen-Ungleichgewicht Ratio und Klassen-Entropie: Stimmen die Größenverhältnisse der Datensatzaufteilung?
- Normalisierung: Wurden Mittelwert und Standardabweichung nur auf den Trainingsdaten berechnet, aber zur Normalisierung der übrigen Teildatensätze angewendet?
- Datenleckage: Gibt es keine Duplikate in den jeweiligen Partitionen bzw. keine Duplikate zwischen den Partitionen, was die Ergebnisse verfälschen könnte?
- Verwendung diverser Datenaugmentierungstechniken um die Varianz des Datensatzes zu erhöhen

Studienergebnisse

Die Auswertung nach der Gruppierung der Antworten zeigte, dass die Antworten aller Befragten zu 70 Prozent Vorgehensweisen, 25 Prozent Metriken und 5 Prozent Tests waren. Dabei konnten alle unterschiedlich viele Antworten in unterschiedlichen Kategorien abgeben (s. Abbildung 6). In der Auswertung wurden lediglich die Antworten selbst aufsummiert. Dies deutet darauf hin, dass die Datenphase eine individuelle Herangehensweise von den Entwicklern erfordert, da die Daten von zugehörigen Anwendungsfall abhängen. Es werden trotzdem Metriken verwendet, um diese individuellen Herangehensweisen zu unterstützen.

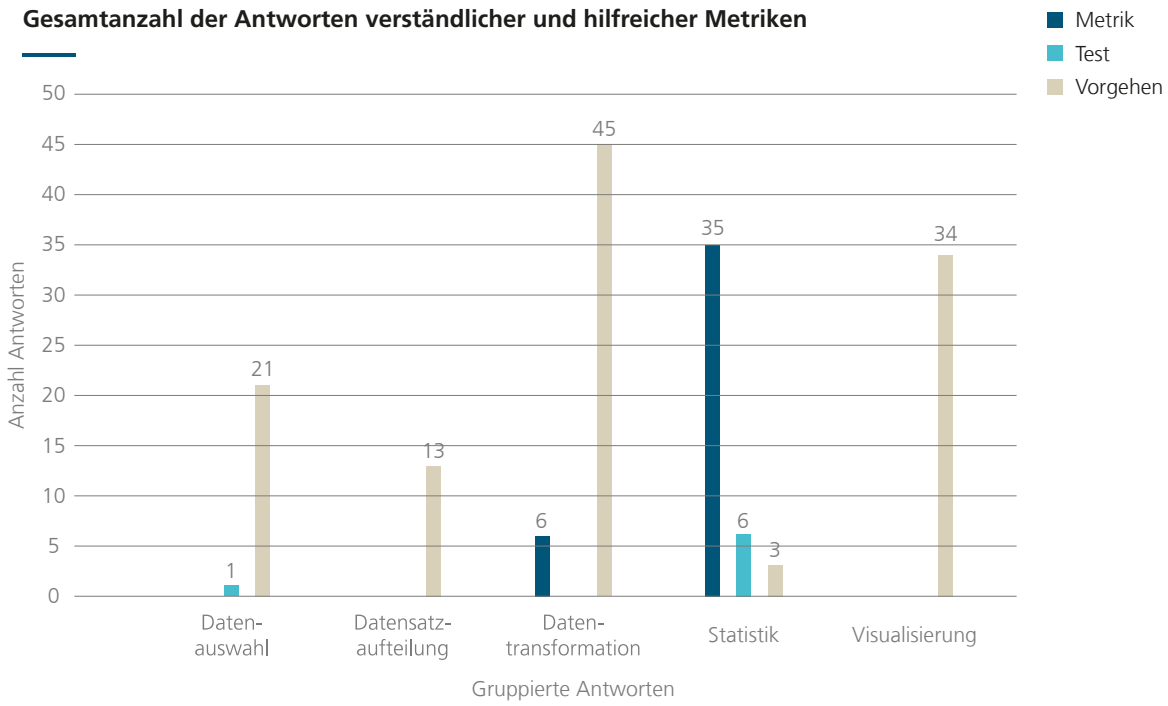


Abbildung 6: Gefilterte absolute Anzahl der Antworten der unterschiedlichen Unterkategorien in der datenzentrierten Phase. Alle Unterkategorien, die mehr als zehn Antworten enthalten, sind dargestellt. Für eine vollständige Abbildung s. Anhang.

Innerhalb der jeweiligen Kategorie wurden die Antworten der Befragten ebenfalls weiter untergliedert in »Datenauswahl«, »Datensatzaufteilung«, »Datentransformation«, »Dokumentation«, »Fehlerbasierte Metrik«, »Hardware«, »Keine Spezielle« – also keine spezielle Vorverarbeitung –, »Sanity-Check«, »Spezialfall für die Aufgabe«, »Statistik«, »Verlaufsmetrik« und »Visualisierung« (s. Abbildung 6). Im Nachgang wird nur auf die Gruppierungen eingegangen, die insgesamt mehr als zehn Antworten besaßen. Somit werden sieben Untergliederungen in der Auswertung übersprungen, um auf die wesentlichen Antworten eingehen zu können.

In der Datenauswahl und Datenvorverarbeitung wurden die meisten Antworten (n=51) der Kategorie Datentransformation zugeordnet. Datentransformationen sind Vorgehensweisen, die im Zuge der Datenvorverarbeitung im KI-Entwicklungszyklus durchgeführt werden. Elf Personen gaben an, dass sie eine Ausreißerbehandlung durchführen, ohne diese genauer zu spezifizieren. Das wird gemacht, da viele ML-Algorithmen gegenüber extremen Werten sensitiv sind und generelle Trends und Zusammenhänge verdecken können. Hinzu kommt, dass die frühzeitige Erkennung und Entfernung von Ausreißern hilft, die Modellgenauigkeit zu verbessern [3]. Zur Erkennung von Ausreißern wurden der Boxplot und auch das Streudiagramm genannt, um die Daten visuell zu veranschaulichen. Hochdimensionale Daten, wie z. B. Bilddaten, können mithilfe von t-SNE (n=1) [16] oder PCA (n=2) erst in eine 2D-Repräsentation transformiert werden und

dann ebenfalls als Punkte in einem Graphen dargestellt werden, um Ausreißer visuell zu erkennen. Auch lernbasierte Methoden existieren für komplexere Datenzusammenhänge und die Erkennung von Ausreißern, in denen das trainierte KI-Modell versucht, die gelernten Muster nachzubilden. Datenpunkte, die nicht nachgebildet werden können, werden als Ausreißer gewertet [14].

Die zweithäufigsten Antworten (n=44) befanden sich in der Kategorie der Statistik, in deren Kategorie sich die häufigsten angegebenen Metriken befanden (n=35). Unter diesen Metriken befand sich der Mittelwert (n=9), Standardabweichung (n=6) und die Varianz (n=5). Diese Metriken lassen sich im KI-Entwicklungszyklus in erster Linie in der Datenexploration ansiedeln. Andere bekannte Metriken waren ebenfalls vertreten, so z. B. der Median, Minimum und Maximum der Daten, Spannweite und der z-Wert [8].

Die Kategorie der Visualisierung bildet eine weitere häufig genannte Kategorie (n=34). Hierbei wurden ausschließlich Vorgehensweisen genannt, in denen die Datensatzdistribution (n=10), Klassenungleichgewichtsüberprüfung (n=8) und die Visuelle Inspektion (n=7) die drei häufigsten Antwortkategorien abbildeten. Mit der Datensatzdistribution ist gemeint, dass die statistische Verteilung der Merkmale des Datensatzes untersucht werden. Beispielsweise kann überprüft werden, ob die Merkmale normalverteilt sind, eine schiefe Verteilung oder multimodale Verteilung aufweisen, oder ob die Daten Ausreißer enthalten. Dieser Schritt ist wichtig in der Datenexplorationsphase und kann Aufschlüsse darüber geben, welches Modell ausgewählt werden soll und ob die Daten weiter transformiert werden müssen. Ein Beispiel hierfür sind Vibrationsdaten von Wälzlagern, die kurze, aber starke Impulse aufweisen. Diese Impulse können mit der Methode der spektralen Kurtosis transformiert und identifiziert sowie als Eingangsdaten für ML-Verfahren verwendet werden [6].

Klassenungleichgewichte werden oft visuell mit Häufigkeitsdiagrammen dargestellt, um zu untersuchen, ob die Daten innerhalb der Klassen ähnlich oft vertreten sind. Sind sie es nicht, wie im fiktiven Beispiel mit den unterschiedlichen Verteilungen der Fehlerklassen, so kann evaluiert werden, ob die Daten im Datenvorverarbeitungsschritt augmentiert werden. Im Fall der Lochscheiben könnte dies bedeuten, dass die unterrepräsentierte Klasse der NIO-Lochscheiben mit künstlich erzeugten Bildern angereichert wird [12].

In der Kategorie der Datenauswahl (n=21) gaben die Teilnehmer mit einer großer Mehrheit Vorgehen an, die sie umsetzen, um die Repräsentativität (n=4) der Daten für die Aufgabe zu gewährleisten. So werden die Daten auf Redundanzen überprüft (n=3) und die Anforderungen an die Daten für die Aufgabenlösung müssen geklärt sein (n=5). Zudem sollen die Daten möglichst komplett sein (n=6), was in das obige Thema der Datensatzdistribution, aber auch der Ausreißerbehandlung mit hineinspielt. So muss im Beispiel des Lochscheibenanwendungsfalls jede Fehlerkategorie im Datensatz vorhanden sein. Bei Zeitreihendaten liegt die Problematik eher im Bereich von fehlenden Werten, wenn beispielsweise mehrere Sensordaten zusammengeführt werden und diese nicht alle zu den gleichen Zeitpunkten gemessen werden können. Fehlende Daten können aber auch u.a. durch Sensorfehler oder schlechter Kalibration entstehen. Fehlende Einträge können zwar einfach gelöscht werden, aber eine ML-Architektur wird in der realen Anwendung mit fehlenden Daten auskommen müssen. Daher gibt es Imputationsmethoden, die v. a. die Komplettheit der Daten gewährleisten. Diese Methoden ersetzen fehlende Einträge mit neuen Werten, wie z. B. den Mittelwert oder den Median der Daten. Letztlich gibt es auch komplexe Imputationsmethoden wie die Verwendung eines weiteren ML-Modells, das Werte basierend auf bereits gesehenen Mustern vorhersagen kann [3].

Eine letzte größere Unterkategorie (n=13) ist die der Datensatzpartitionierung, die v. a. beim überwachten Lernen zum Einsatz kommt. Neun Befragte gaben an, dass die Datensatzaufteilung ein wichtiger Schritt für die Datensatzvorverarbeitungsphase im KI-Entwicklungszyklus ist. Hierbei erwähnte eine Person, dass sie dies teilweise mit einem bestimmten Verhältnis

standardisieren, z. B. 80 Prozent der Daten waren für den Trainingsdatensatz und 20 Prozent der Daten für den Testdatensatz. Eine weitere Person erwähnte, dass sie die Zuordnung ihrer Daten zu bestimmten Datensätzen automatisierte, um so möglichst objektiv sein zu können und ihre eigenen Vorurteile über die Daten nicht in das Verfahren zu integrieren. Für unterschiedliche Datentypen ist die Aufteilung nicht immer ohne weiteres möglich. So muss beispielsweise bei Zeitreihendaten bei der Aufteilung auf die zeitliche Ordnung geachtet werden, oder bei Bilddaten, dass unterschiedliche Bilder desselben Objekts sowohl im Trainings- als auch im Testdatensatz auftauchen. Verwandt mit diesem Thema ist auch das Sampling (n=3), wobei es hierbei meistens um Zeitreihendaten geht, wie z. B. Video- oder Audiodaten. Damit ist gemeint, dass nur jeder zweite Datenpunkt der ML-Architektur übergeben wird, um die Rechenzeit zu verkürzen. In bestimmten Fällen sind hochfrequente Daten nicht unbedingt informativ von einem Datenpunkt zum nächsten, sodass es sinnvoll sein kann, nur jeden m-ten Datenpunkt zu erfassen. Diese Strategie ist fester Bestandteil moderner ML-Architekturen, denn sie ermöglicht die Erkennung von Aktionen über mehrere zeitliche Skalen hinweg und ist besonders hilfreich, wenn die zeitliche Auflösung der Aktionen variiert, wie beispielsweise das Erkennen menschliche Aktivitäten in Videos [7].

Name	Beschreibung	Zweck	Hinweise
Metriken			
Mittelwert (Mean)	Arithmetisches Mittel aller Werte	Für symmetrische Daten ohne starke Ausreißer	Ausreißer können stark verzerren
Standardabweichung	Wurzel der Varianz; misst Streuung in Originaleinheit	Praktischer als Varianz; gleiche Einheit wie die Daten	Empfindlich gegenüber Ausreißern
Varianz	Mittlere quadratische Abweichung vom Mittelwert	Streuung; wichtig für Tests/ Modellwahl	Einheit ist quadriert; empfindlich gegenüber Ausreißern
Anzahl der Daten (n)	Zahl der verfügbaren Beobachtungen	Einschätzung der Aussagekraft	Sehr kleine Stichproben sind unzuverlässig
Median	Zentralwert der geordneten Daten	Robust bei Schiefe und Ausreißern	Unempfindlich gegenüber Ausreißern
Minimum/ Maximum	Kleinster und größter beobachteter Wert	Grenzen und potenzielle Ausreißer erkennen	Kann durch Ausreißer verzerrt sein
Fehlwertrate	Anteil fehlender Werte	Planung von Imputation/ Bereinigung	Hohe Raten verzerren Analysen
Signal-zu-Rausch-Verhältnis (SNR)	Verhältnis von Nutzsignal zu Rauschen	Beurteilung, ob Signal dominiert	Definition variiert je nach Domäne
Tests			
Pearson-Korrelationstest	Testet linearen Zusammenhang	Stärke/Richtung des Zusammenhangs	Kein Kausalitätsnachweis; empfindlich gegenüber Ausreißern
t-Test	Vergleicht Mittelwerte zweier Gruppen	Prüft signifikante Unterschiede	Voraussetzungen prüfen (Normalität/ Varianzen)
Vorgehen			
Ausreißer behandeln	Erkennen und ggf. korrigieren/entfernen	Robustere Analysen/Modelle	Entscheidungen begründen und protokollieren
Normalisierung/ Skalierung	Merkmale auf vergleichbare Skalen bringen	Besseres Lernverhalten vieler Modelle	Nur am Training fitten, dann anwenden
Verteilungsanalyse des Datensatzes	Form/Lage der Verteilung untersuchen	Frühes Verständnis; Data-Drift erkennen	Mit Histogrammen/Boxplots kombinieren
Train/Val/Test-Aufteilung	Daten in Train/Val/Test teilen	Realistische Leistungsschätzung	Stratifizieren bei Ungleichgewichten

Klassenimbalancen prüfen	Ungleiche Klassenanteile erkennen	Verzerrte Modelle vermeiden	Resampling/Klassengewichte erwägen
Visuelle Inspektion	Sichtung von Rohdaten/Labels	Fehler und Muster schnell finden	Repräsentative Stichproben wählen
Fehlwerte behandeln	Löschen, imputieren oder kennzeichnen	Daten nutzbar halten	Auswirkungen dokumentieren
Komplettheit prüfen	Sind Pflichtfelder gefüllt?	Datenqualität messen	Schwellwerte definieren
Konsistenz prüfen	Widersprüche/Regelbrüche finden	Logische Fehler verhindern	Automatisierte Checks etablieren
Daten-Anforderungen festlegen	Benötigte Felder/Qualität/Fristen definieren	Grundlage für Sammlung/ Prozesse	Mit Stakeholdern abstimmen
Rauschbehandlung/ -analyse	Messrauschen quantifizieren/reduzieren	Stabilere Messungen/Modelle	Nicht überglätten (Signalverlust)
Repräsentativität prüfen	Stichprobe vs. Zielpopulation vergleichen	Übertragbarkeit sichern	Gewichte/Sampling anpassen
Korrelationsanalyse	Zusammenhänge zwischen Merkmalen	Feature-Auswahl & Leckage-Check	Nichtlinearitäten evtl. übersehen
Redundanzen erkennen	Doppelte/überflüssige Daten finden, z.B. durch informationstheoretische Merkmalsselektion auf Grundlage der Transinformation (Mutual Information)	Bias vermeiden, Rechenzeit sparen	Vielfalt der Daten erhalten
Sampling/ Stichprobenziehung	Gezielte/zufällige Teilmengen ziehen	Effizient bei großen Daten	Sampling-Bias vermeiden
Sensoren überprüfen / kalibrieren	Messgeräte testen/kalibrieren	Zuverlässige Messwerte	Regelmäßig wiederholen

3.2. Feature Engineering

Üblicherweise werden bei Bilddaten die normalisierten Pixelwerte als Eingangsgrößen für ML-Algorithmen verwendet, insbesondere bei den gängigsten Architekturen (tiefe Neuronale Netze). Es ist jedoch denkbar, die Ergebnisse eines Kantendetektionsalgorithmus als Eingangsgröße eines sehr einfachen Modells zu nutzen. Die Anzahl der detektierten Kanten sollte bei NIO-Lochscheiben durch die zusätzliche Inhomogenität höher sein als bei IO-Lochscheiben. Dafür müsste geprüft werden, ob dies tatsächlich der Fall ist und somit eine komplizierte ML-basierte Lösung unnötig wäre. Somit bieten sich zwei Modellarchitekturen mit gänzlich unterschiedlichen Feature-Engineering-Schritten an. Des Weiteren könnte eine Visualisierung durch eine Dimensionsreduktions-Methode frühzeitig helfen, fehlerhafte Features zu detektieren. Dabei werden die Merkmale eines Bilds auf beispielsweise einen 2D-Raum projiziert, wobei wesentliche Strukturen wie Varianz und Nachbarschaften (ähnliche Bilder bleiben in der Nähe zueinander) erhalten bleiben. Folgendes könnte dabei hilfreich sein:

- 2D-Visualisierung der Features mittels Dimensionsreduktion
- Statistische Merkmale der Features, um Auffälligkeiten zu detektieren

Studienergebnisse

Zunächst sei darauf verwiesen, dass fast die Hälfte der Befragten (n=10) angaben, keinerlei Feature Engineering zu betreiben. Dies ist nicht verwunderlich, wenn man berücksichtigt, dass genau die Hälfte von ihnen (n=11) im Bereich Computer Vision arbeitet und dort zunehmend seltener klassisches Feature Engineering betrieben wird.

Die größte Gruppe (siehe Abbildung 6) umfasst jegliche Maßnahmen der »Feature-Analyse«. Die am häufigsten benannten, konkreten Feature-Engineering-Schritte zur Sicherstellung der Qualität der Features sind »bivariate Assoziationsmaße« wie Pearson- oder Spearman-Korrelationskoeffizienten, mit denen die Korrelation zweier Features und Informationsmaße wie die

Transinformation oder der Informationsgewinn untersucht werden können (insgesamt n=7). Grundsätzlich helfen diese Metriken dabei, stark korrelierte bzw. ähnliche Merkmale zu detektieren. Gemäß dem Parsimonitätsprinzip soll ein Modell so einfach wie möglich und so komplex wie nötig sein. Hochkorrelierte Merkmale können redundant sein und führen unter Umständen zu Scheinkorrelationen mit der Zielvariable und werden demnach oft entfernt. Beispielsweise könnte in einer Fabrik zur Produktion von Metallteilen ein Modell zur Vorhersage von Ausschuss (Defektwahrscheinlichkeit) trainiert werden. Als Eingangsgrößen werden u. a. die Produktionsgeschwindigkeit (Teile pro Minute) und die Bandgeschwindigkeit (m/s) genutzt. Beide Merkmale sind stark miteinander verknüpft: Je schneller das Förderband läuft, desto mehr Teile pro Minute werden gefertigt. Wenn man beide Variablen gleichzeitig im Modell verwendet, entsteht eine Scheinkorrelation. Das Modell könnte fälschlicherweise ›lernen‹, dass sowohl Produktionsgeschwindigkeit und Bandgeschwindigkeit unabhängig Einfluss auf die Ausschussrate haben. In Wirklichkeit steckt dieselbe Information doppelt im Datensatz.

Ähnlich wie in der datenzentrierten Phase werden auch hier die statistischen Momente der Features (n=3) vom Erwartungswert über die Varianz bis hin zur Wölbung analysiert und ggf. als Merkmal verwendet.

Statt der ursprünglichen Merkmale ist es denkbar, diese durch eine Transformation in eine andere Basis zu überführen. Auf »Transformationen« basierte Merkmale werden durch die Fast Fourier-Transformation (n=2), Autoencoder-basierte Merkmalsextraktion (n=1) sowie temporale Merkmalanalyse (n=2) im Falle von Zeitreihen erzeugt.

Weitere Mehrfachnennungen, um die Qualität sicherzustellen, sind die Betrachtung von Relevanzmaßen der Merkmale (n=2) sowie die Ablationsstudie (n=2) der aufgabenspezifischen Performanzmetriken (Fehlermaß-basierte Metriken) nach Entfernen vermeintlich irrelevanter Merkmale. Hierfür wird betrachtet, wie sich das Modellverhalten verändert, wenn besagte Merkmale hinzugenommen oder entfernt werden.

Die Einbindung von »Externem Wissen« kann laut Umfrage durch optische Inspektion der Merkmale (n=1) und Einbindung von Domänenwissen in die Generierung von Merkmalen (n=1) stattfinden.

Die gegebenen Antworten setzen sich zu 60 Prozent aus Metriken, 33 Prozent Vorgehen und 7 Prozent Tests zusammen. Im Gegensatz zu nicht-linearen Merkmals-reduktionsmethoden, wie dem Autoencoder, können ebenfalls klassische, lineare Dimensionsreduktionsmethoden wie die SVD-basierte PCA als Feature-Engineering-Methoden verwendet werden.

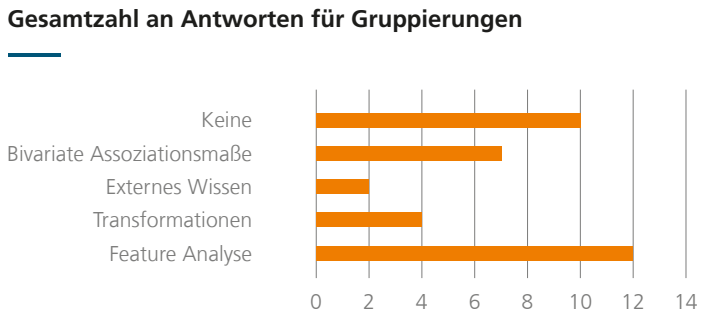


Abbildung 7: Absolute Anzahl der Antworten der unterschiedlichen Unterkategorien in der Phase Feature Engineering. Für eine vollständige Abbildung aller genannten Maßnahmen s. Anhang.

Name	Beschreibung	Zweck	Hinweise
Metriken			
Pearson Korrelations-Koeffizient	Misst direkte, lineare Zusammenhänge zwischen Variablen	Abhängigkeiten und Redundanzen erkennen	Werte von -1 bis +1; nahe 0 = schwach
Spearman Korrelations-Koeffizient	Misst Rangfolge-Zusammenhänge, auch nicht-linear	Komplexere Abhängigkeiten entdecken	Robuster als Pearson; gut für ordinale Daten
Statistische Momente von Merkmalen	Grundlegende Kennzahlen (Mittelwert, Streuung, Schiefe …)	Datenqualität und Verteilung verstehen	Erste Analyse-Stufe; zeigt Ausreißer/ Anomalien
Feature Importance Scores	Bewertung, wie wichtig einzelne Merkmale für Vorhersagen sind	Einflussreichste Faktoren identifizieren	Hilft bei Priorisierung von Datenquellen/ Kosten
Tests			
Ablationstest (Merkmale entfernen)	Testet linearen Zusammenhang	Stärke/Richtung des Zusammenhangs	Kein Kausalitätsnachweis; empfindlich gegenüber Ausreißern
Vorgehen			
Analyse fehlerhafter Datenpunkte	Systematisch falsch vorhergesagte Fälle untersuchen	Ursachen finden, gezielt Features/Labels verbessern	Kombinierbar mit Konfusions-Matrix/ Beispiel-Reviews
Fast Fourier Transformation (FFT)	Wandelt Zeitreihen in Frequenzmerkmale um	Periodik/Schwingungen sichtbar machen	Gleichmäßige Abtastung nötig; Fensterung beachten

3.3. Modellzentrierte Phase

Modelltraining
Beide Lösungsansätze, das tiefe Neuronale Netz und die klassische Alternative, haben klare Vor- und Nachteile. Da das fiktive Unternehmen einen Industrierechner mit leistungsfähiger Grafikkarte für das Projekt beschafft hat, ist der erhöhte Rechenaufwand der Pixelwert-basierten Methode nicht relevant. Nach der Recherche des Stands der Technik entscheidet sich das Unternehmen, sogenannte Faltungsnetze (CNN [9]) und Vision Transformer (ViT [2]) als ML-Algorithmen weiterzuverfolgen. Zusätzlich wünscht sich das Unternehmen eine Lokalisierung der Fehler. Dafür wird ein klassifikations- und objektdetektierungsfähiges Modell ausgewählt. Darüber hinaus sollte Folgendes während der Modellauswahl berücksichtig werden:

- Kreuzvalidierung: Zur besseren Generalisierung wird das Modell auf n-1 Partitionen des Datensatzes trainiert und auf der n-ten evaluiert. Dies wird n-Mal wiederholt und die durchschnittliche Modellgüte, gemessen an vorab festgelegten Gütekriterien, zwischen verschiedenen Modellen verglichen
- Expressivität: Die grundlegende Fähigkeit der ausgewählten Modelle sollte für die vorliegende Problemklasse geeignet sein.
- Hosmer-Lemeshow-Test zur Bewertung der generellen Anpassungsgüte der binären Klassifikationsaufgabe (IO vs. NIO)

Modellbewertung und Modelltuning
Die primäre Zielgröße ist die zuverlässige Erkennung der Fehlerklassen. Eine Genauigkeit über den gesamten Datensatz ist dabei nicht ausreichend, da die NIO-Instanzen deutlich seltener

vorkommen als die IO-Instanzen. Falls beide Methoden, Faltungsnetz und Vision Transformer, den technischen Anforderungen der Klassifizierungsgenauigkeit von 99,5 Prozent für alle Klassen gerecht werden und die Lochscheiben zuverlässig detektieren, können weitere Gütekriterien untersucht werden. Dafür müssten potenziell weitere Stakeholder wie z. B. die Endanwender in der Produktionsstätte miteinbezogen werden, wenn es um die Intuitivität von Erklärbarkeitsmethoden geht. Weitere relevante Anforderungen schließen ein:

Technische Anforderungen:

- Objektdetektion: MeanIoU (Mean Intersection over Union)
- Klassifikation: Klassenspezifische Genauigkeit
- Nicht-technische Anforderungen:
- Genauigkeitsverlust unter diversen Rausch-Arten injiziert in die Testdaten
- Genauigkeitsverlust unter Transformationen wie Rotation, Zoom, Okklusion und weiteren Manipulationen
- Intuitivität verschiedener Erklärbarkeitsmethoden wie SHAP [1] oder Wärmekarten (Heatmaps) [13] für den Endanwender

Studienergebnisse

In der Praxis setzen die meisten Entwickler auf aufgabenspezifische »Fehlermaße« (n=78), um zu prüfen, wie gut ein Modell wirklich arbeitet. Für die Klassifikation, also ob eine Lochscheibe IO oder NIO ist, bzw. wenn NIO, welcher Fehler vorliegt, werden häufig Konfusionsmatrix-basierte Metriken (, s. Abbildung 8, n=40) genutzt.

Gesamt-Population $= P + N$	Positive Vorhersage	Negative Vorhersage	Youden-Index $= S_0 + S_1 - 1$	
Real Positiv P	Richtig Positiv TP	Falsch Negativ FN	Positiver Vorhersagewert $TPR = \frac{TP}{TP + FN}$	Falsch Negativ Rate $FNR = \frac{FN}{TP + FN}$
Real Negativ N	Falsch Positiv FP	Richtig Negativ TN	Negativer Vorhersagewert $TNR = \frac{TN}{TN + FP}$	Falsch Positiv Rate $FPR = \frac{FP}{FP + TN}$
Prävalenz $= \frac{P}{P + N}$	Sensitivität $S_0 = \frac{TP}{TP + FN}$	Spezifizität $S_1 = \frac{TN}{TN + FP}$	Pos. Likelihood Rate $LR^+ = \frac{TPR}{FPR}$	Neg. Likelihood Ratio $LR^- = \frac{FNR}{FPR}$
Genauigkeit $= \frac{TP + TN}{P + N}$	Falscherkennungs-Rate $= \frac{FP}{TP + FP}$	F1-Score $= \frac{2\,TP}{2\,TP + FP + FN}$	Diagnostische Odds Ratio $= \frac{LR^+}{LR^-}$	Gewichtete Genauigkeit $= \frac{TPR + TNR}{2}$

Abbildung 8: Konfusionsmatrix. Angelehnt an engl. Wikipedia (https://en.wikipedia.org/wiki/Confusion_matrix)

Diese zeigen nicht nur die Gesamtgenauigkeit, sondern geben auch Aufschluss darüber, wie häufig jede Fehlerart fälschlicherweise als intakt oder als eine andere Fehlerart eingestuft wird und umgekehrt. Seltener, aber erwähnenswert, ist der Brier-Score (n=1) [10]. Er bewertet, wie gut kalibriert die Vorhersagewahrscheinlichkeiten sind, also ob das Modell z. B. bei einer durchschnittlichen Sicherheit von 90 Prozent auch tatsächlich in etwa zu 90 Prozent richtig liegt. Für Regressionsaufgaben werden das Bestimmtheitsmaß (R^2-Score) (n=4), der mittlere absolute

Fehler (MAE, n=9) und der mittlere quadratische Fehler (MSE) bzw. die Quadratwurzel davon (RSME, in Summe n=15) eingesetzt. Sie verdeutlichen anschaulich, wie groß die durchschnittlichen Vorhersagefehler sind.

Als nicht-technische Gütekriterien werden in der Kategorie »Zuverlässigkeit« insbesondere Robustheit (n=4) und Erklärbarkeit (n=2) genannt, z. B., ob das Modell bei leicht verrauschten oder gedrehten Bildern weiterhin verlässlich NIO-Instanzen erkennt und ob Endanwender in der Produktionsstätte mittels Heatmaps nachvollziehen können, warum das Modell »NIO: Beule« sagt.

Ziel beim Training einer ML-Komponente ist es, Überanpassung (Overfitting) an die Trainingsdaten zu verhindern. Überanpassung liegt vor, wenn ein Modell auf den Trainingsdaten sehr gute Ergebnisse erzielt, auf dem Testdatensatz jedoch deutlich schlechter abschneidet. Für eine gute »Generalisierbarkeit« (insgesamt n=29) werden zwei Strategien betont: die saubere Evaluierung auf Validierungs- und Testdaten (n=15) sowie Kreuzvalidierung (n=5). Bei Letzterer wird der Trainingsdatensatz in k Teile partitioniert; ein Modell wird k-mal auf k-1 Teilen des Datensatzes trainiert und jeweils auf dem übrig gebliebenen Teil des Datensatzes geprüft. So erhält man eine stabilere Aussage darüber, wie gut die Erkennung fehlerhafter Lochscheiben wirklich generalisiert.

Beim Tuning der »Hyperparameter«, also Parameter wie z.B. Lernrate, Batch-Größe oder Anzahl der Schichten, die nicht während des Trainings optimiert werden sondern vorher festgelegt werden müssen, werden die Optimierung der Architektur (n=1) und modellspezifischer Hyperparameter (n=6) genannt. Diese werden zum Beispiel anhand einer Gittersuche ausgewählt. Dafür werden für jeden Hyperparameter ein Set zulässiger Werte definiert und so viele Trainingsläufe durchgeführt, bis alle Kombinationsmöglichkeiten erschöpft sind.

Vortrainierte Modelle werden eingesetzt, um von bereits erlerntem Wissen zu profitieren und den Trainingsaufwand bei neuen Aufgaben zu reduzieren, insbesondere wenn nur begrenzte Daten vorliegen. Häufig werden dabei nur die letzten ein bis zwei Schichten eines neuronalen Netzes an die neue Aufgabe angepasst, während die frühen Schichten eingefroren bleiben (n=1). Auf diese Weise bleiben die Modellgewichte der eingefrorenen Schichten während der Feinjustierung unverändert, während die angepassten Schichten gezielt für die spezifischen Anforderungen der neuen Aufgabe trainiert werden.

Empirisch zeigt sich, dass die zufallsbasierte Hyperparameter-Suche der Gittersuche oft überlegen ist [4]. Die zufällige Auswahl der Hyperparameter innerhalb eines sinnvollen Wertebereichs durchsucht den Raum der wichtigen Stellschrauben effizienter und findet mit geringerer Rechenzeit häufiger gute Kombinationen; ein praktischer Vorteil gegenüber der Gittersuche, wenn schnell eine robuste NIO-Erkennung gefordert ist.

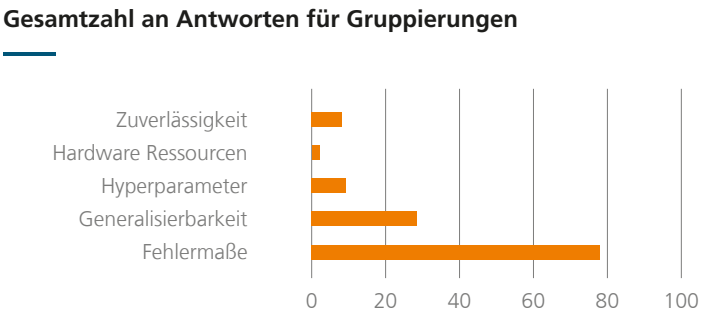


Abbildung 9: Absolute Anzahl der Antworten der unterschiedlichen Unterkategorien in der modellzentrierten Phase. Für eine vollständige Abbildung aller genannten Maßnahmen s. Anhang.

Name	Beschreibung	Zweck	Hinweise
Metriken			
Mean Squared Error/ Root Mean Squared Error (MSE/RMSE)	Durchschnittliche Abweichung der Vorhersagen	Bewertung der Vorhersage-genauigkeit bei Zahlenwerten	Niedrigere Werte = bessere Vorhersagen
Konfusions-Matrix	Tabelle mit korrekten vs. falschen Vorhersagen	Detailanalyse der Modellschwächen	Zeigt spezifische Fehlerarten auf
Mean Absolute Error (MAE)	Durchschnittliche absolute Abweichung	Bewertung der typischen Vorhersagefehler	Leichter interpretierbar als MSE
F1-Score	Ausgewogene Bewertung von Präzision und Vollständigkeit	Gesamtbewertung bei unausgewogenen Daten	Guter Kompromiss zwischen verschiedenen Fehlerarten
Area Under Curve (AUC)	Gesamtbewertung der Unterscheidungsfähigkeit	Vergleich verschiedener Modelle	Wert zwischen 0,5 (schlecht) und 1,0 (perfekt)
Genauigkeit (Accuracy)	Anteil aller korrekt klassifizierten Fälle	Gesamtbewertung der Modellleistung	Kann bei unausgewogenen Daten irreführend sein
Präzision	Anteil korrekt vorhergesagter positiver Fälle	Bewertung der Genauigkeit bei wichtigen Entscheidungen	Wichtig, wenn Fehlalarme teuer sind (z. B. Spam-Filter)
R-Squared	Anteil der erklärten Varianz in den Daten	Bewertung der Modellerklärungskraft	Wert zwischen 0 (schlecht) und 1 (perfekt) für lineare Modelle
ROC-AUC	Bewertung bei verschiedenen Entscheidungsschwellen	Gleichgewicht zwischen Trefferrate und Fehlalarmen optimieren	Standard-Metrik für binäre Klassifikation
Vorgehen			
Evaluation auf Validierungs-/Testdaten	Getrennte Datensätze für faire Beurteilung	Realistische Leistung messen	Datenleck vermeiden; nicht auf Testdaten trainieren
Hyperparameter-Optimierung	Systematische Suche nach guten Einstellungen	Leistung und Stabilität maximieren	Suchraum/Seeds dokumentieren
Kreuzvalidierung	Daten in Folds teilen und mehrfach auswerten	Schwankungen reduzieren	Stratifizieren bei Imbalance; bei Zeitreihen zufällige Splits vermeiden
Fehleranalyse (Error Analysis)	Systematisches Durchgehen von Fehlfällen	Schwachstellen finden	Nach Klassen/Features/Gruppen clustern
Early Stopping	Training stoppen bei ausbleibender Verbesserung	Überanpassung vermeiden, Zeit sparen	»Patience« und Metrik sauber wählen
XAI (Erklärbare KI)	Verfahren zur Modell-Erklärung (z. B. SHAP)	Vertrauen & Debugging	Erklärungen kontextabhängig interpretieren
Tests			
Robustheit	Verhalten unter Rauschen/Shift prüfen	Stabilität im Einsatz bewerten	Realistische Störszenarien wählen
Sensitivitätsanalyse	Einfluss einzelner Eingaben variieren	Einflussfaktoren erkennen	Auch Grenzfälle betrachten

3.4. Modelleinsatz und Modellinferenz

Nachdem das fiktive Unternehmen festgestellt hat, dass der Vision Transformer zwar robuster ist, jedoch zulasten der Erklärbarkeit geht, möchte es beide Ansätze, CNN und ViT, im Inferenzmodus vergleichen. Dafür werden die Modelle auf den Industrierechner geladen und als Klassifikatoren bzw. Detektoren eingesetzt. Hier gilt es u. a. auf Folgendes zu achten:

- GPU-Auslastung: Wie viel GPU-Speicher benötigt das Modell?
- Inferenzzeit: Wie lange dauert die Inferenz pro Bild oder Batch?

Studienauswertung
In der Inferenz-Phase wurden zu 68 Prozent Metriken, 20 Prozent Vorgehen und 12 Prozent Tests genannt. Sie ist somit die Metriken-lastigste Phase, auch wenn die Diskrepanz zur modellzentrierten Phase nur 5 Prozentpunkte beträgt. Ein Teil der Befragten (n=6) hat noch keine Erfahrungen mit dem produktiven Einsatz außerhalb des Labors.

Am häufigsten genannt wurden Methoden der Gruppe »kontinuierliches Monitoring«: Daten- bzw. Konzeptdrift erkennen (n=3) oder das Modell bei Bedarf nachtrainieren/rekalibrieren (n=3). Für das Lochscheiben-Beispiel heißt das: Eingehende Bilderströme auf schleichende Änderungen prüfen, z.B. neue Beleuchtung, andere Oberflächenreflexe, Abweichungen in der Unsicherheit oder der Fehlerrate früh entdecken und gezielt mit frischen Beispielen nachschärfen.

Für den produktiven Vergleich zwischen verschiedenen Modellarchitekturen wurden vor allem ressourcen- und zeitbezogene Kennzahlen genannt, welche jeweils den Gruppen »Zeiten« bzw. »Ressourcen« zugeordnet worden sind. Inferenzzeit (n=4) ist die Zeit, die das KI-Modell selbst benötigt, um seine Vorhersage zu berechnen, und die Latenzzeit (n=3) gibt an, wie lange das KI-Gesamtsystem braucht, um eine Entscheidung zu treffen. Der Durchsatz (n=1) misst, wie viele Anfragen pro Zeiteinheit ein KI-System beantworten kann. Für die Produktionslinie im Lochscheibenbeispiel würde besonders die p95-Latenz pro Bild – also wie lange 95 Prozent der Anfragen maximal dauern, damit die Taktzeit sicher eingehalten wird, relevant sein.

Unter Ressourcen wurde der VRAM/RAM-Verbrauch (n=2), also der Arbeitsspeicherverbrauch der Grafikkarte bzw. des Hauptspeichers, genannt und die GPU-Leistungsaufnahme (n=2), also die elektrische Leistung der Grafikkarte in Watt. Die CPU-Kern-Auslastung (n=1) misst, wie stark ein einzelner CPU-Kern belastet ist. Im Vergleich fällt bei dem Vision Transformer auf, dass dieser oft mehr VRAM benötigt, und von größeren Batches profitiert, während kompakte Faltungsnetze oft die geringere Latenz liefern. Für das Beispiel ist dies wichtig, wenn die Bilder in Echtzeit bewertet werden sollen.

Wie auch in der modellzentrierten Phase, lassen sich in der Phase Inferenz einige Antworten der Gruppe »Generalisierbarkeit« zuordnen, also dem Ziel, Maßnahmen gegen die Überanpassung an die Trainingsdaten zu treffen. Bei Qualitätsmetriken wurden die Fehlerrate (n=1) und Vorhersage-Konfidenz (n=1) genannt. Für qualitative Qualitätssicherstellung beschrieben manche Teilnehmer visuelle Checks (n=2) in der Produktionsstätte.

Ebenfalls wurde genannt, dass die Leistung spezieller Modelle, die auf wenigen oder gar keinen Beispieldaten gelernt haben (Few-Shot/Zero-Shot), untersucht werden sollte (n=2), insbesondere bei neuen, seltenen Fehlertypen. Außerdem ergibt die Perturbationsanalyse (n=1) zur Robustheit gegen z. B. leichte Drehung/Verdeckung in den Bildern Aufschlüsse über die Stabilität des Modells gegenüber realistischen Bildveränderungen. A-/B-Tests dienen dazu, die wirtschaftliche Auswirkung (n=1) zu ermitteln, etwa zur Messung, wie viele NIO-Lochscheiben tatsächlich fehlerfrei produziert werden.

Für die konkrete Entscheidung auf dem Industrierechner empfiehlt sich eine Gegenüberstellung CNN gegen ViT bei identischer Eingabeauflösung und einer Messung im Warmbetrieb unter echter Last inklusive Protokollierung von Fehlern, Vorhersageunsicherheiten und potenziellen Driftindikatoren.

Falls der Vision Transformer gewählt werden würde, könnte man Optimierungen wie Mixed-Precision oder Quantisierung prüfen, um VRAM und Latenz zu senken. Dafür würden beispielsweise die Gleitkommazahlen der Modellgewichte statt durch größere 32-Bit nur durch 16-Bit dargestellt werden. Diese benötigen dementsprechend weniger Speicherplatz, jedoch zulasten der numerischen Genauigkeit.

Beim CNN könnte sichergestellt werden, dass die Erklärbarkeitsmethoden für Werksmitarbeiter verständlich bleiben, um die Akzeptanz der Lösung zu gewährleisten.

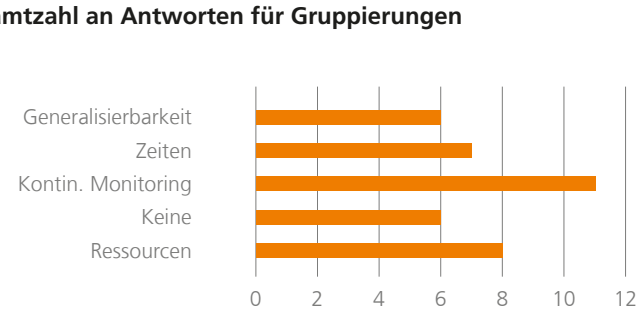


Abbildung 10: Absolute Anzahl der Antworten der unterschiedlichen Unterkategorien in der Phase Inferenz. Für eine vollständige Abbildung aller genannten Maßnahmen s. Anhang.

Name	Beschreibung	Zweck	Hinweise
Metriken			
Inferenzzeit	Zeit für einzelne Vorhersagen	Bewertung der Echtzeitfähigkeit	Wichtig für zeitkritische Anwendungen
Latenzzeit	Verzögerung zwischen Anfrage und Antwort	Bewertung der Benutzerfreundlichkeit	Wichtig für interaktive Anwendungen
Compute-Kosten	Finanzielle Aufwendungen für Rechenleistung	Kostenkontrolle und Budgetplanung	Beeinflusst ROI des ML-Projekts
GPU Wattage	Stromverbrauch der Grafikprozessoren	Energiekostenplanung und Nachhaltigkeit	Beeinflusst Betriebskosten erheblich
Model-Assessment-Methoden	Verschiedene Ansätze zur Modellbewertung	Umfassende Qualitätssicherung	Cross-Validation, Hold-out etc.
VRAM/RAM Usage	Speicherverbrauch während der Ausführung	Hardware-Dimensionierung und Kostenplanung	Bestimmt minimale Systemanforderungen
Zero-Shot-/ Few-Shot-Performanz	Leistung bei neuen, untrainierten Aufgaben	Bewertung der Modellflexibilität	Wichtig für adaptive Systeme
Tests			
Data Drift/Shift Assessment	Erkennung von Veränderungen in Datenmustern	Überwachung der Modellaktualität	Kritisch für langfristige Modellstabilität
A-/B-Test für Business Impact	Vergleich der Geschäftswirkung verschiedener Modelle	Messung des tatsächlichen Geschäftsnutzens	Wichtigste Metrik für ROI-Bewertung

Vorgehen			
Retraining	Regelmäßiges Nachtrainieren mit aktuellen Daten	Anpassung an neue Muster und Sicherung der Performance	Versionieren, Rollback-Plan, Datenqualität prüfen
Visuelle Checks	Manuelle Sichtprüfung von Vorhersagen/Beispielen	Schnelles Erkennen offensichtlicher Fehler	Nur Stichprobe; ergänzt quantitative Metriken

3.5. Metriken in der Praxis

Die Teilnehmer wurden befragt, ob sie bestimmte Metriken, Tests oder Vorgehen besonders hilfreich empfinden. An dieser Stelle wurden die Antworten gruppiert in »Abhängigkeit der Aufgabe/Schwer zu sagen«, »Fehlermaße«, »Sonstiges« und »Verlaufsmetriken«. Abhängigkeit von der Aufgabe bedeutet in diesem Kontext, dass einige Teilnehmer angaben, dass die Nützlichkeit einer Metrik von der Aufgabe abhängt und deshalb keine Aussage getroffen werden könne. Fehlermaße sind Metriken und Tests, die auf irgendeine Art und Weise wiedergeben, wie weit das Modell von seiner zu lösenden Aufgabe abweicht, z. B. Genauigkeit oder MSE. Die Kategorie »Sonstiges« bildet Antworten ab, die sehr aufgabenabhängig waren. Verlaufsmetriken beschreiben Metriken, deren zeitliche Veränderung den Entwicklern Aufschlüsse geben.

Die Teilnehmer gaben mit 11 Prozent an, dass diese Frage schwer zu beantworten und abhängig von der Aufgabe sei (s. Abbildung 10). 71 Prozent der Befragten empfanden Fehlermaße wie Mittelwert und Genauigkeit als hilfreich und 2 Prozent gaben die Verlaufsmetriken wie Verlustfunktionsvisualisierung an. 16 Prozent der restlichen Antworten waren schwer zu kategorisieren, da sie sehr spezifisch für den Anwendungsfall waren.

Nützliche und hilfreiche Metriken

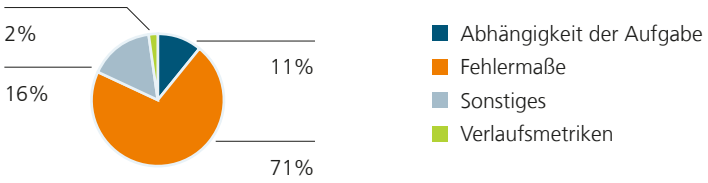


Abbildung 11: Prozentuale Verteilung der Antworten bezüglich der Nützlichkeit der Metriken in Gruppierungen, die während der Auswertung vorgenommen wurden.

Eine weitere gestellte Frage umfasste das Thema, welche Metrik die Teilnehmer am intuitiv verständlichsten empfinden. Die gleichen Kategorien wie in der Frage oben wurden beibehalten und die Kategorie der »Abhängigkeit der Aufgabe/Schwer zu sagen« nahm von 11 Prozent auf 5 Prozent ab. Fehlermaße erhöhten sich leicht auf 79 Prozent, während die Verlaufsmetriken von 2 auf 5 Prozent ebenfalls leicht zunahmen (s. Abbildung 11).

Verständliche Metriken

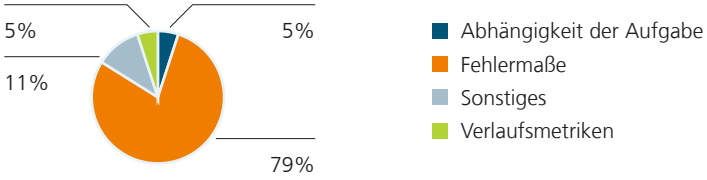


Abbildung 12: Prozentuale Verteilung der Antworten bezüglich der Verständlichkeit von Metriken in Gruppierungen, die während der Auswertung vorgenommen wurden.

Bei einem direkten Vergleich der genauen Fehlermaße (s. Abbildung 12) lässt sich erkennen, dass sechs Teilnehmer die Genauigkeit als nützlich und hilfreich sowie mit zehn Teilnehmern als besonders verständlich angegeben haben. Damit ist die Genauigkeit die Metrik, die absolut als verständlichste Metrik über allen anderen erwähnten Metriken angegeben wurde. Am zweithäufigsten wurde der MSE als hilfreich angegeben (n=6) und ebenfalls als zweit-intuitivste Metrik (n=4).

Was als besonders verständlich angesehen wird, ist, wie die Teilnehmer selbst angaben, abhängig von der Aufgabe. Wer eine Klassifizierungsaufgabe durchführt, wie bei dem Lochscheibenanwendungsfall, arbeitet mit der Genauigkeit und findet diese hilfreich und verständlich. Wer aber eine Regressionsaufgabe hat, der betrachtet den mittleren quadratischen Fehler oder ähnliche Metriken als hilfreich und verständlich, um den Unterschied zwischen einem vorhergesagten Wert und dem tatsächlichen Wert zu ermitteln.

Insgesamt lässt sich daraus ableiten, dass auch Entwickler einfache Metriken schätzen. Wie ein Studienteilnehmer erwähnt, kann man sich darauf verlassen, dass diese gängige Metriken in unterschiedlichen Software-Bibliotheken (korrekt) implementiert wurden. Bei komplexeren Metriken ist dies nicht immer unbedingt gegeben und kann zu Abweichungen der Ergebnisse führen.

Gesamtanzahl der Antworten verständlicher und hilfreicher Metriken

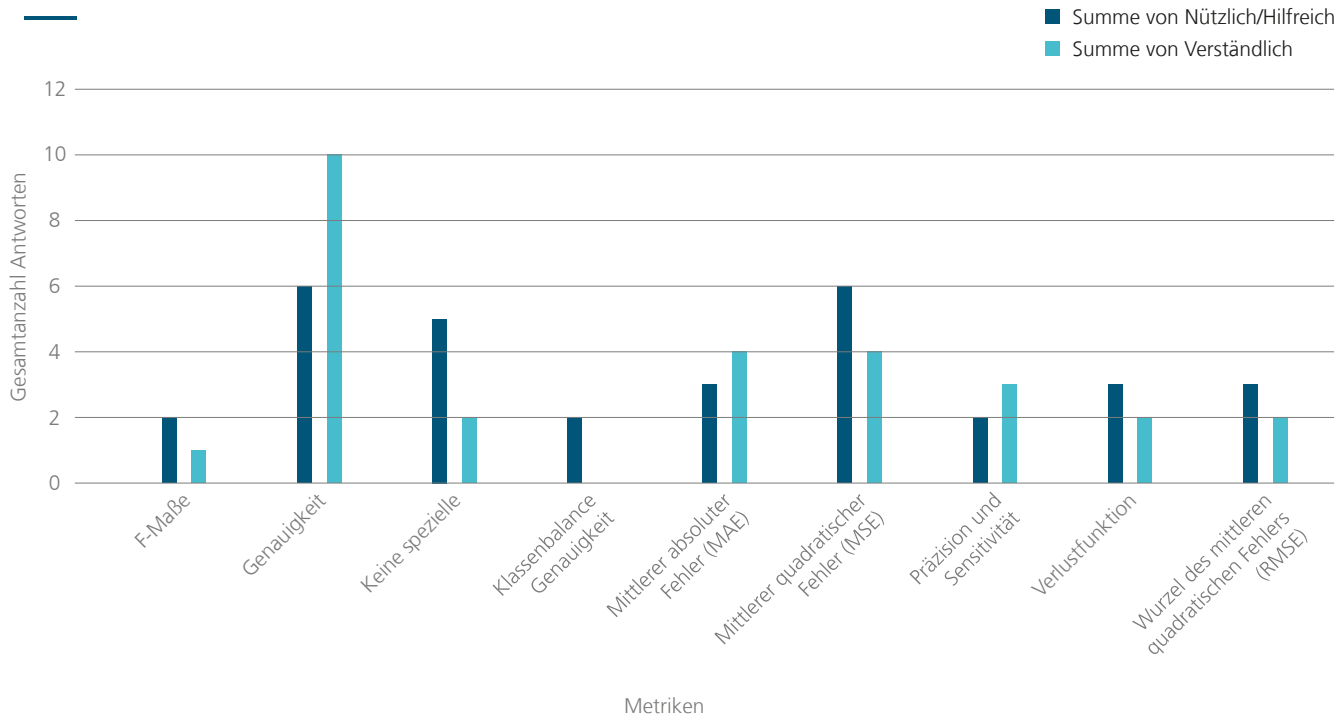


Abbildung 13: Antworten der Teilnehmer in einer Gegenüberstellung zwischen Nützlichkeit und Verständlichkeit. Nur Antworten, die mehr als einmal angegeben wurden, sind dargestellt. Für einen vollständigen Grafen s. Anhang.

Dass Fehlermaße wie die Genauigkeit, F-Maße sowie Präzision und Sensitivität als hilfreich unter Entwicklern angesehen werden, ist leicht nachzuvollziehen; schließlich ergibt die Genauigkeit ein erstes Verständnis von der Leistungsfähigkeit des ML-Verfahrens. Das Gleiche gilt für Metriken, die für Regressionsaufgaben geeignet sind: Ein möglichst niedriger Wert für MAE oder MSE ist einfach zu interpretieren und v. a. mit anderen Experimenten zu vergleichen. Weitere Metriken geben genaueren Aufschluss darüber, wie gut das Modell im Kontext des Anwendungsfall ist. Oft gibt es eine Verschiebung in eine bestimmte Richtung: Ist das Modell besonders präzise, kann die Sensitivität geringer ausfallen. Bei der Auswahl der Metriken ist darauf zu achten, welche für den jeweiligen Anwendungsfall die größte Relevanz besitzt. Geht es um die Erkennung von Personen in einem Sicherheitsbereich, sollte die Sensitivität besonders hoch sein, da sie sicherstellt, dass Personen verlässlicher erkannt werden.

Beide Fragen ermutigten die Befragten, Metriken oder Verfahren zu nennen, die den ganzen KI-Entwicklungszyklus betreffen. Allerdings ist interessant, dass bis auf wenige Ausnahmen fast nur Antworten zu Evaluationsmetriken von ML-Modellen genannt wurden. Beispielsweise wurden die Visuelle Inspektion und Erkennung der Verzerrung der Daten als hilfreich anerkannt und die Datenverteilung jeweils einmal als intuitiv angegeben.

3.6. Zusammenfassung

Die hier dargelegte Studie beleuchtet, wie Qualität in ML-Systemen über den gesamten Entwicklungszyklus anhand von Metriken, Tests und Vorgehensweisen gesichert wird. Viele praxisnahe Fachleute trennen die Phasen des Entwicklungszyklus nicht strikt, der Prozess ist iterativ. Aus diesem Grund wurden die Umfrageergebnisse auf die vier Phasen (datenzentriert, Feature Engineering, modellzentriert und Inferenz) verdichtet.

Für die datenzentrierte Phase stehen Vorgehensweisen (60 Prozent) im Mittelpunkt, während in den restlichen Phasen die Metriken (61-68 Prozent) überwiegen. Tests repräsentieren einen durchgehend kleinen Anteil der Antworten (4-12 Prozent). Praktiker bevorzugen einfache, etablierte Fehlermaße: 71 Prozent nannten sie als hilfreich, 79 Prozent als am intuitivsten, was darauf hindeutet, dass Standardmetriken oft einfach zu berechnen und daher leichter zu verstehen sind. Genauigkeit ist die verständlichste Metrik insgesamt (nützlich n=6, verständlich n=10). Bei Regression folgen MSE (n=6 hilfreich; n=4 intuitiv) und MAE (n=4 intuitiv). Die Gründe dafür können vielfältig sein. Zum einen bieten sie eine einfache Interpretation, eine gute Vergleichbarkeit zwischen Experimenten und konsistente Implementierungen in gängigen Bibliotheken. Dem gegenüber bergen komplexere Metriken Implementierungsunterschiede und erhöhen den Interpretationsaufwand.

Für die Praxis bedeutet das, dass zunächst die aufgabenabhängigen Fehlermaße (z. B. konfusionsbasierte Metriken bzw. MAE/MSE) effektive Metriken sind, die dann ergänzt werden können für feinere Diagnosemetriken oder um Verzerrungen oder Veränderungen gezielt zu erkennen und Gegenmaßnahmen abzuleiten. Explizite Nennungen betrafen überwiegend Evaluationsmetriken des Modells, nur vereinzelt wurden datenseitige Checks (Visuelle Inspektion, Verzerrungserkennung, Verteilungsprüfung) als intuitiv oder hilfreich genannt.

Die Qualitätssicherung in ML ist primär praxis- und kontextgetrieben, wird jedoch durch wenige, gut gewählte Metriken und Tests wirkungsvoll fundiert. Wer Datenhygiene, robuste Bewertung, gezieltes Tuning und ein operativ belastbares Inferenz-Setup verbindet, erreicht eine nachhaltige Leistungsfähigkeit im produktiven Einsatz.

4. Anhang

4.1. Vollständige Darstellung der Umfrageergebnisse

Antworten für Phase Feature Engineering

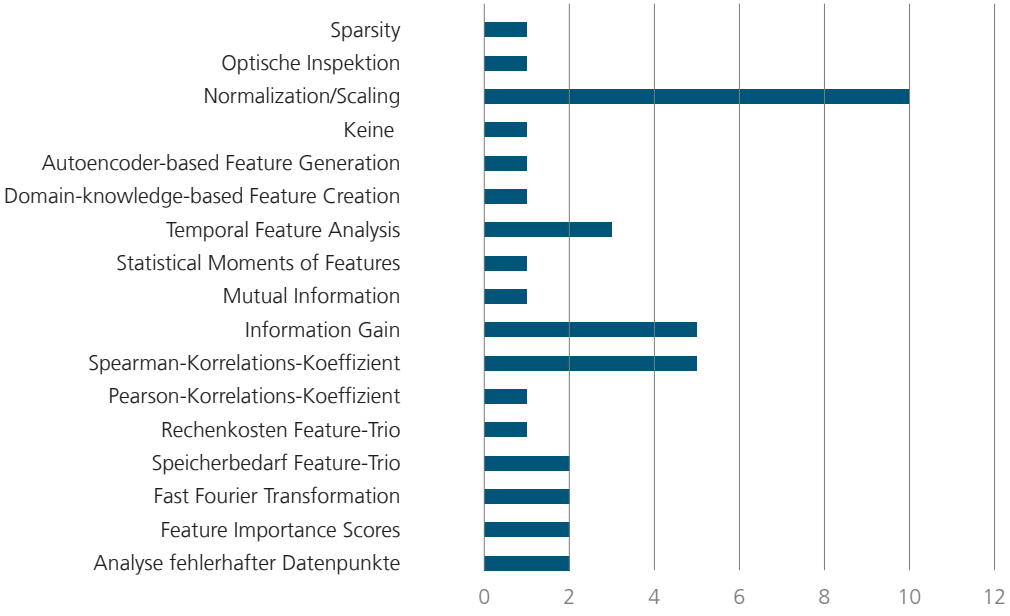


Abbildung 14: Vollständige Auflistung der Umfrageergebnisse für die Phase Feature Engineering

Antworten für modellzentrierte Phase

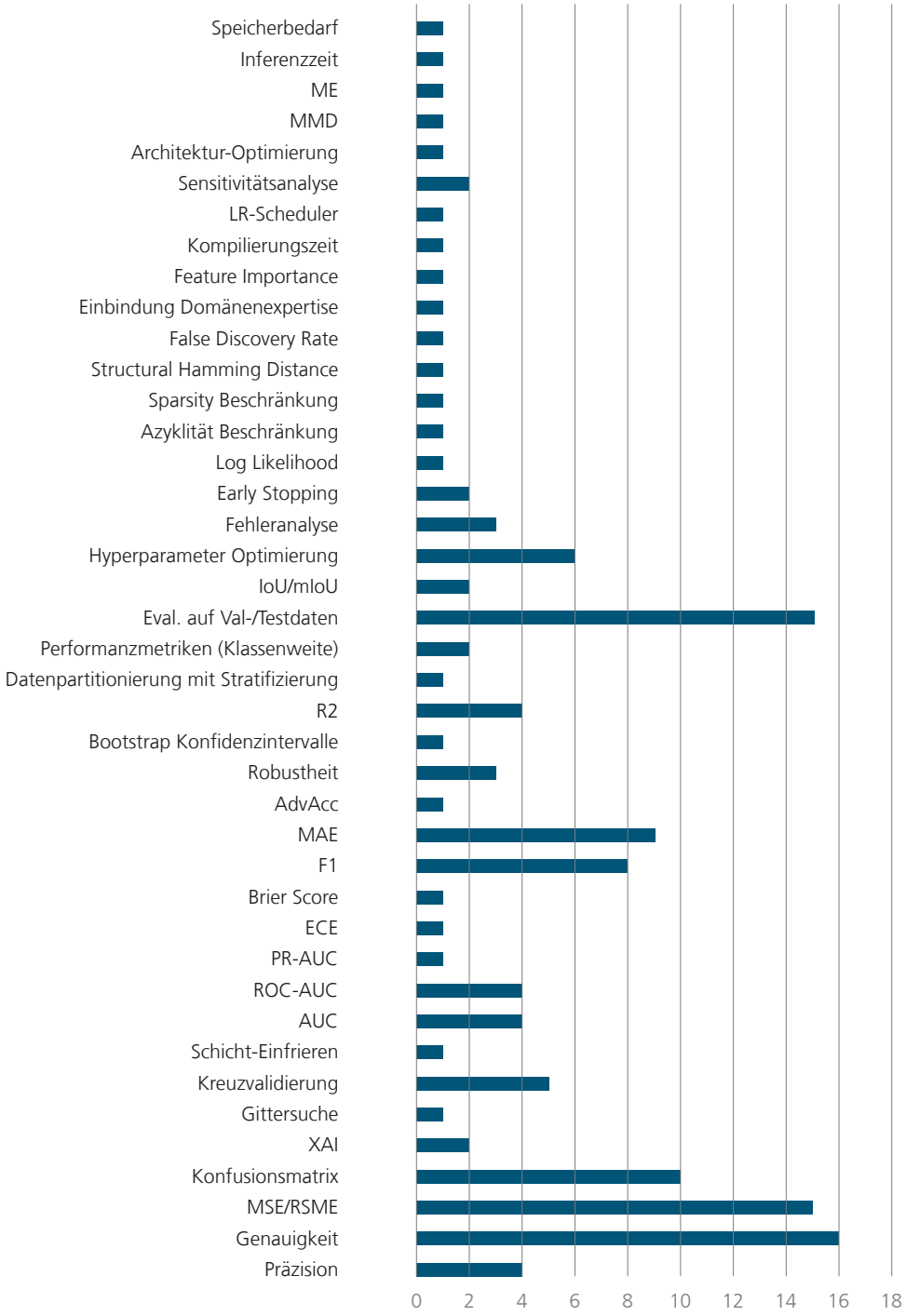


Abbildung 15: Vollständige Auflistung der Umfrageergebnisse für die modellzentrierte Phase

Antworten für Phase Inferenz

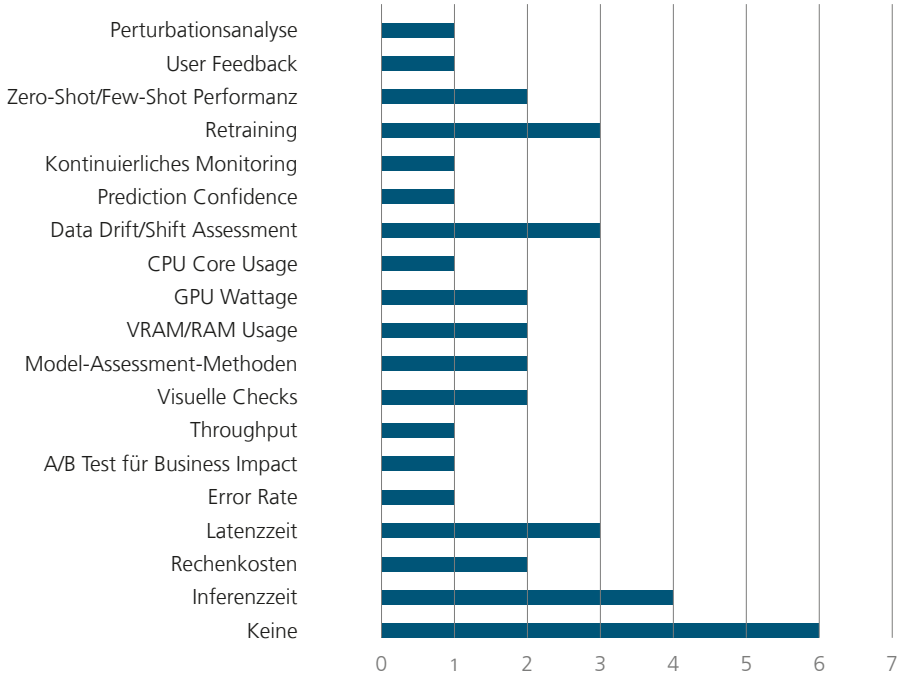


Abbildung 16: Vollständige Auflistung der Umfrageergebnisse für die Phase Inferenz

4.2. Verhältnisse Metriken/Tests/Vorgehen pro Phase

Datenbezogene Antworten

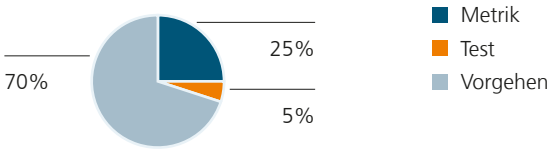


Abbildung 17: Aufteilung der datenbezogenen Antworten in die Kategorien Metrik, Test und Vorgehen.

Feature Engineering

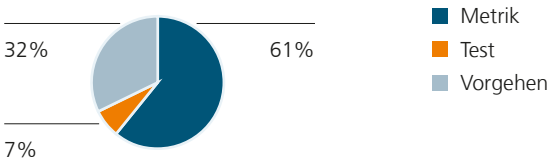


Abbildung 18: Aufteilung der Antworten in die Kategorien, Metrik, Test und Vorgehen für die Phase Feature Engineering.

Modellzentrierte Phase

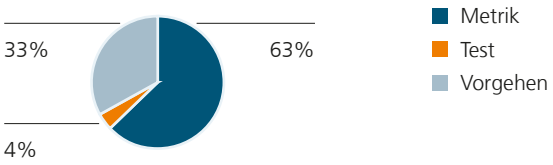


Abbildung 19: Aufteilung der Antworten in die Kategorien Metrik, Test und Vorgehen für die modellzentrierte Phase.

Inferenz

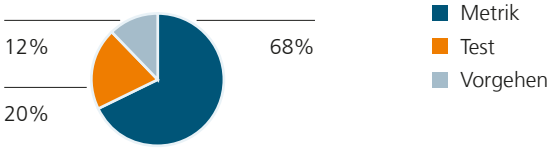


Abbildung 20: Aufteilung der Antworten in die Kategorien Metrik, Test und Vorgehen für die Phase Inferenz.

5. Literaturverzeichnis

[1] 2017. A unified approach to interpreting model predictions.

[2] 2020. An image is worth 16x16 words: Transformers for image recognition at scale.

[3] Bindu Bala and Sunny Behal. 2024. A Brief Survey of Data Preprocessing in Machine Learning and Deep Learning Techniques, 1755–1762. DOI: https://doi.org/10.1109/I-SMAC61858.2024.10714767.

[4] James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. The Journal of Machine Learning Research 13, 281–305.

[5] Ira Effenberger, Frederik Seiler, and Verena Eichinger. 2024. Synthetische Daten für die Automatisierung mit KI/Synthetic data for AI-based automation. wt 114, 09, 490–496. DOI: https://doi.org/10.37544/1436-4980-2024-09-34.

[6] Alessandro Fasana, Stefano Marchesiello, Miriam Pirra, Luigi Garibaldi, and Alessandra Torri. 2010. Spectral Kurtosis against SVM for best frequency selection in bearing diagnostics. Mécanique & Industries 11, 6, 489–494. DOI: https://doi.org/10.1051/meca/2010056.

[7] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. 2018. SlowFast Networks for Video Recognition.

[8] William F. Guthrie. 2020. NIST/SEMATECH e-Handbook of Statistical Methods (NIST Handbook 151). DOI: https://doi.org/10.18434/M32189.

[9] 1989. Handwritten digit recognition with a back-propagation network.

[10] 1950. Monthly Weather Review. War Department, Office of the Chief Signal Officer.

[11] Omar De Mitri, Andreas Frommknecht, Marco F. Huber, Felix Müller-Graf, and Cosimo Distante. 2023. Synthetic data generation for improvement of machine learning-based optical quality control: a practical approach in the welding context. In , 12621. SPIE, 237–245. DOI: https://doi.org/10.1117/12.2682047.

[12] Frederik Seiler, Verena Eichinger, and Ira Effenberger. 2024. Synthetic Data Generation for AI-based Machine Vision Applications. ei 36, 6, 276-1-276-5. DOI: https://doi.org/10.2352/EI.2024.36.6.IRIACV-276.

[13] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2017. Grad-CAM: Visual Explanations from Deep Networks via

Gradient-Based Localization. In 2017 IEEE International Conference on Computer Vision (ICCV). IEEE, 618–626. DOI: https://doi.org/10.1109/iccv.2017.74.

[14] Nazmul K. Sikder and Feras A. Batarseh. 2023. Outlier Detection using AI: A Survey 2023 (2023).

[15] 2000. The CRISP-DM model: the new blueprint for data mining.

[16] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE (2008).

[17] William J. Frawley, Gregory Piatetsky-Shapiro, and Christopher J. Matheus. 1992. Knowledge Discovery in Databases: An Overview. AIMag 13, 3, 57. DOI: https://doi.org/10.1609/aimag.v13i3.1011.

6. Abbildungsverzeichnis

KI-Fortschrittszentrum

Das KI-Fortschrittszentrum »Lernende Systeme und Kognitive Robotik« unterstützt Firmen dabei, die wirtschaftlichen Chancen der Künstlichen Intelligenz und insbesondere des Maschinellen Lernens für sich zu nutzen. In anwendungsnahen Forschungsprojekten und in direkter Kooperation mit Industrieunternehmen arbeiten die Stuttgarter Fraunhofer-Institute für Produktionstechnik und Automatisierung IPA sowie für Arbeitswirtschaft und Organisation IAO daran, Technologien aus der KI-Spitzenforschung in die breite Anwendung der produzierenden Industrie und der Dienstleistungswirtschaft zu bringen. Unterstützt werden sie dabei vom Institut für Arbeitswissenschaft und Technologiemanagement IAT der Universität Stuttgart. Finanzielle Förderung erhält das Zentrum vom Ministerium für Wirtschaft, Arbeit und Tourismus Baden-Württemberg.

Mission
Das KI-Fortschrittszentrum ist der anwendungsorientierte Zweig von Cyber Valley, Europas größter Forschungskooperation im Bereich der Künstlichen Intelligenz. Darüber hinaus ist das KI-Fortschrittszentrum Bestandteil von S-TEC, dem Stuttgarter Technologie- und Innovationscampus: www.s-tec.de

Das KI-Fortschrittszentrum schlägt die Brücke von der KI-Spitzenforschung in den Mittelstand und macht KI-Technologien für die Wirtschaft in Baden-Württemberg und darüber hinaus nutzbar. Als führender Innovationspartner für den Mittelstand arbeitet das Zentrum an Themen, die für den Einsatz von KI und Robotik branchenübergreifend von zentraler Bedeutung sind, beispielsweise Autonomie, Effizienz und Nachhaltigkeit, Mensch-Maschine-Interaktion sowie Vertrauen. Das KI-Fortschrittszentrum informiert Unternehmen über Technologietrends und deren Einsatzpotenziale und unterstützt sie bedarfsgerecht und niedrigschwellig bei der Entwicklung und Umsetzung von ambitionierten KI-Innovationen, damit sie die wirtschaftlichen Chancen der KI künftig noch besser nutzen können.

Vision
Das KI-Fortschrittszentrum ist ein Leuchtturm für erfolgreichen Technologietransfer in den Mittelstand und ermöglicht Unternehmen einen wirtschaftlichen und verantwortungsvollen Einsatz von Künstlicher Intelligenz und Robotik für unternehmerischen Erfolg sowie individuellen und gesellschaftlichen Nutzen.

Studienreihe »Lernende Systeme und Kognitive Robotik«
Die Studienreihe »Lernende Systeme und Kognitive Robotik« gibt Einblick in die Potenziale und die praktischen Einsatzmöglichkeiten von KI. Nähere Informationen und die aktuellen Versionen der Studien finden Sie unter: www.ki-fortschrittszentrum.de/veroeffentlichungen/

Fraunhofer

Die Fraunhofer-Gesellschaft mit Sitz in Deutschland ist eine der führenden Organisationen für anwendungsorientierte Forschung. Im Innovationsprozess spielt sie eine zentrale Rolle – mit Forschungsschwerpunkten in zukunftsrelevanten Schlüsseltechnologien und dem Transfer von Forschungsergebnissen in die Industrie zur Stärkung unseres Wirtschaftsstandorts und zum Wohle unserer Gesellschaft. Seit ihrer Gründung als gemeinnütziger Verein im Jahr 1949 nimmt sie eine einzigartige Position im Wissenschafts- und Innovationssystem ein.

Knapp 32 000 Mitarbeitende an 75 Instituten und selbstständigen Forschungseinrichtungen in Deutschland erarbeiten das jährliche Finanzvolumen von 3,6 Mrd. €. Davon entfallen 3,1 Mrd. € auf das zentrale Geschäftsmodell von Fraunhofer, die Vertragsforschung. Im Vergleich zu anderen öffentlichen Forschungseinrichtungen bildet die Grundfinanzierung durch Bund und Länder lediglich das Fundament des jährlichen Forschungshaushalts. Sie ist die Basis für wegweisende Vorlaufforschung, die in den kommenden Jahren für Wirtschaft und Gesellschaft bedeutend wird. Das entscheidende Alleinstellungsmerkmal ist der hohe Anteil an Wirtschaftserträgen, der Garant ist für die enge Zusammenarbeit mit Wirtschaft und Industrie und die stetige Marktorientierung der Fraunhofer-Forschung: 2024 beliefen sich die Wirtschaftserträge auf 867 Mio. € des laufenden Haushalts. Ergänzt wird das Forschungsportfolio durch im Wettbewerb eingeworbene öffentliche Projektmittel, wobei eine ausgewogene Balance zwischen öffentlichen und wirtschaftlichen Erträgen angestrebt wird.

Wir produzieren Zukunft
Das Fraunhofer-Institut für Produktionstechnik und Automatisierung IPA, kurz Fraunhofer IPA, realisiert hoch innovative und nachhaltige Lösungen in der Produktionstechnik und Automatisierung für verschiedenste Zukunftsbranchen. Dies können Methoden, Komponenten und Geräte bis hin zu kompletten Maschinen und Anlagen sein. Die Lösungen stehen stets in Verbindung mit den strategischen Eckpfeilern des Instituts »Mass Sustainibility« und »Mass Personalization«. Seine Hauptaufgabe sieht das Institut im Wissens-, Innovations- und Technologietransfer von Forschungsergebnissen in Applikationen, um die Wettbewerbsfähigkeit der Unternehmen zu stärken. Dabei versteht es sich als unabhängiger Ansprechpartner, der neutral berät und Unternehmen mit genau auf deren Bedürfnisse zugeschnittenen Projektteams unterstützt.

Impressum

Kontaktadresse
Fraunhofer-Institut für Produktionstechnik und Automatisierung IPA
Nobelstraße 12, 70569 Stuttgart

Emil Andreas
Telefon +49 711 970-3734
emil.andreas@ipa.fraunhofer.de

Christopher Braun
Telefon +49 711 970-1473
christopher.braun@ipa.fraunhofer.de

Florian Strohm
Telefon +49 711 970-6065
florian.strohm@ipa.fraunhofer.de

Martina Tenzer
Telefon +49 711 970-1832
martina.tenzer@ipa.fraunhofer.de

Fraunhofer-Publica
http://dx.doi.org/10.24406/publica-7086

Titelbild
© NicoElNino – Adobe Stock

Gefördert durch das Ministerium für Wirtschaft, Arbeit
und Tourismus Baden-Württemberg

Kontakt

KI-Fortschrittszentrum
Fraunhofer IAO und Fraunhofer IPA
Nobelstraße 12
70569 Stuttgart
www.ki-fortschrittszentrum.de

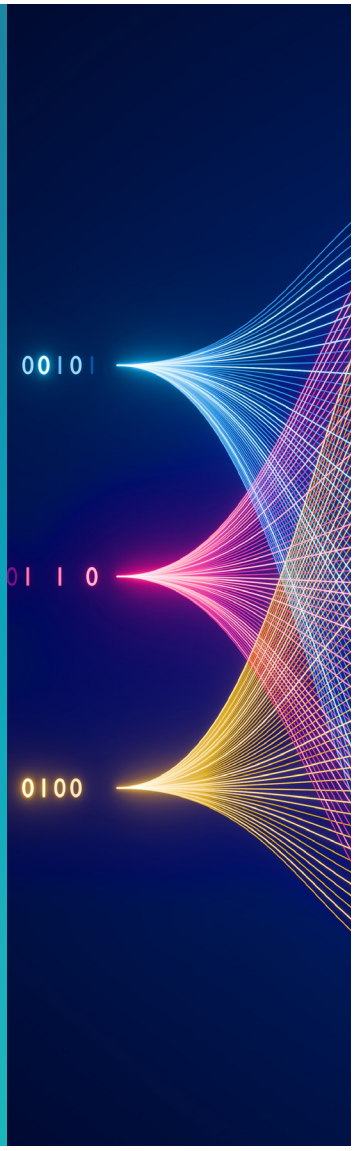
Emil Andreas
Telefon +49 711 970-3734
emil.andreas@ipa.fraunhofer.de

Christopher Braun
Telefon +49 711 970-1473
christopher.braun@ipa.fraunhofer.de

Florian Strohm
Telefon +49 711 970-6065
florian.strohm@ipa.fraunhofer.de

Martina Tenzer
Telefon +49 711 970-1832
martina.tenzer@ipa.fraunhofer.de

Fraunhofer IPA
Nobelstraße 12
70569 Stuttgart
www.ipa.fraunhofer.de



Gefördert durch

Baden-Württemberg
Ministerium für Wirtschaft,
Arbeit und Tourismus

Partner

Cyber Valley

S-TEC

Universität Stuttgart
Institut für Arbeitswissenschaft und
Technologiemanagement IAT