



Uwe Haneke • Michael Zimmer
Stephan Trahasch (Hrsg.)

Künstliche Intelligenz für Business Analytics

Grundlagen, Architekturen
und Anwendungen



dpunkt.verlag

Inhalt

Cover

Hinweise zur Benutzung

Titel

Impressum

Inhalt

Geleitwort

1 KI ist alles und alles ist KI

1.1 Künstliche Intelligenz als Schlüsseltechnologie und Katalysator

1.2 Aufbau des Buches

Teil I Grundlagen

2 Maschinelles Lernen

2.1 Einleitung

2.2 Supervised Learning

2.2.1 Grundlegende Konzepte

2.2.2 Problemtypen und Beispiele

2.2.3 Mathematisches Framework

2.2.4 Lernen abseits von Supervised Learning

2.3 Künstliche neuronale Netze und Deep Learning

2.3.1 Aufbau und Funktionsweise

2.3.2 Training und Vortraining

2.3.3 Zusammenfassung

2.4 Reinforcement Learning

2.5 Fazit

3 Natural Language Processing

3.1 Einleitung

- 3.2 Die Bedeutung von Sprache
- 3.3 Texte als unstrukturierte Daten
- 3.4 Von Sprache zu Zahlen: Vorverarbeitung
 - 3.4.1 Tokenisierung und weitere Vorverarbeitungsmethoden
 - 3.4.2 Informationsextraktion
 - 3.5 Texte als Zahlen: Repräsentation von Sprache für Machine-Learning-Modelle
 - 3.5.1 Das Bag-of-Words-Modell
 - 3.5.2 Das Vektorraum-Modell: Wörter als Vektoren
 - 3.5.3 Das Zipfsche Gesetz
 - 3.5.4 Term Frequency – Inverse Document Frequency
 - 3.6 Wörter als Vektoren
 - 3.6.1 Wortrepräsentation mit One-Hot Encoding
 - 3.6.2 Statische Word Embeddings
 - 3.6.3 Kontextualisierte Word Embeddings
 - 3.7 Large Language Models
 - 3.8 Herausforderungen und Chancen
 - 3.8.1 Bias
 - 3.8.2 Datenschutz
 - 3.8.3 Mehrsprachigkeit
 - 3.8.4 Open-Source-, On-Premises- und Cloud-basierte Modelle
 - 3.9 Zusammenfassung
- 4 Künstliche Intelligenz in der Computer Vision
 - 4.1 Einordnung und Historie
 - 4.2 Eigenschaften digitaler Bilder
 - 4.3 Neuronale Netze im maschinellen Sehen
 - 4.3.1 Faltungsnetze
 - 4.3.2 Vision Transformer

4.4 Anwendungen im maschinellen Sehen

4.4.1 Fine-Tuning von Standardarchitekturen

4.4.2 Klassifizierung von Bildern

4.4.3 Objektdetektion

4.5 Zusammenfassung und Ausblick

5 Künstliche Intelligenz und Robotik

5.1 Grundlagen

5.1.1 Wahrnehmung

5.1.2 Modellbildung

5.1.3 Entscheidung

5.2 Reinforcement Learning

5.2.1 Problemdefinition

5.2.2 Durchführung

5.3 Kommunikation mit Robotern

5.4 Einsatz in Unternehmen

5.5 Zusammenfassung

Teil II Weiterführende Konzepte

6 Operationalisierung von KI-Systemen

6.1 Vom Prototyp zum Go-live

6.1.1 Beispielanwendung

6.1.2 Automatisierung der Modellerstellung

6.1.3 Deployment von ML-Modellen

6.2 Monitoring von KI-Systemen

6.2.1 Vom Fehler zur Katastrophe

6.2.2 Die Rolle der Datenverteilung

6.2.3 ML-Monitoring in der Praxis

6.3 ML-Operations

6.4 Zusammenfassung

7 Vorgehensmodelle für die Entwicklung von KI-Anwendungen am Beispiel von DASC-PM

7.1 Aktueller Stand: ein kurzer Überblick

7.2 Data Science Process Model

7.3 Projektauftrag

7.4 Datenbereitstellung

7.5 Analyse

7.6 Nutzbarmachung

7.7 Nutzung

7.8 Übergreifende Schlüsselbereiche

7.9 Zusammenfassung

8 Datenvorbereitung als Erfolgsfaktor in KI-Projekten

8.1 Einleitung

8.2 Datenbeschaffung

8.3 Datenbereinigung

8.4 Datentransformation

8.5 Datenbereitstellung

8.6 Zusammenfassung

9 Aufbau einer KI-Einheit im Unternehmen

9.1 Einleitung

9.2 Aufgaben einer KI-Einheit

9.2.1 Teufels Advokat für neue Lösungen: aufklären und Awareness schaffen

9.2.2 Management-Buy-in sicherstellen

9.2.3 KI-Plattform-Management

9.2.4 Identifikation, Entwicklung und Betrieb von KI-Anwendungen

9.2.5 First und Second Line für KI

9.2.6 Heimathafen für Data Scientists

9.3 KI-Organisation: vom Elfenbeinturm zum integrierten KI-Hub

9.4 Erfolgsfaktoren

9.5 Zusammenfassung

10 Wertschöpfung mit KI: vom Quick Win zum Geschäftsmodell

10.1 Einleitung: Warum KI-Wertschöpfung jetzt auf die Agenda gehört

10.1.1 Direkte Wertschöpfung durch KI: die drei zentralen Hebel

10.1.2 Indirekte Wertschöpfung durch KI: qualitative Effekte nutzen

10.1.3 Generative KI als Beschleuniger: schnelle Erfolge ohne Großprojekte

10.2 Rahmen schaffen: Governance und Kompetenzen für breite KI-Nutzung

10.2.1 KI-Policy und Lizenzen: Klare Regeln schaffen Freiräume

10.2.2 Upskilling: vom Ausprobieren zur Anwendungskompetenz

10.3 Use-Case-Ideation und Portfolioaufbau: von der Idee zur Umsetzung

10.3.1 Workshops: Katalysatoren für Awareness und Ideensammlung

10.3.2 Use-Case-Identifikation: Pain Points als Schlüssel zum Erfolg

10.3.3 Schulungen: gezielte Skills für erfolgreiche Anwendungsfälle

10.3.4 Portfolio: Sichtbarkeit macht Synergien möglich

10.4 KI-Enabling: eigene Daten und Prozesse für tiefe Wertschöpfung

10.4.1 Prompt-Datenbanken: Eingabevorlagen zentral verfügbar machen

10.4.2 KI-Assistenten: Spezialisten für wiederkehrende Aufgaben

10.4.3 RAG-Systeme: interne Wissensquellen intelligent nutzen

10.4.4 KI-Agenten: vom Tool zur autonomen Aufgabenlösung

10.5 Formalisierte KI-Projekte: Effizienz und Innovation systematisch umsetzen

10.5.1 Effizienzprojekte: bestehende Prozesse mit KI verbessern

10.5.2 Neuproduktprojekte: KI als Treiber für innovative Produkte und Services

10.5.3 KI-Portfolio: Balance zwischen Optimierung und Innovation finden

10.5.4 Phasenmodell: in vier Schritten vom Anwendungsfall zur Umsetzung

10.6 Geschäftsmodelle: Ansätze für den KI-basierten Wandel

10.7 Zusammenfassung

11 Betriebliche Wertschöpfung durch KI: Prozessmodell für eine langfristige Perspektive

11.1 Einleitung

11.2 Theoretischer Rahmen zur Untersuchung von Wertbeiträgen durch KI

11.3 Drei Mechanismen zur Schaffung von Unternehmenswert durch Machine Learning

11.3.1 Mechanismus 1: Wissensgenerierung

11.3.2 Mechanismus 2: Assistenzsysteme

11.3.3 Mechanismus 3: autonome Agenten

11.3.4 Einfluss der Wertschöpfungsmechanismen auf die Gestaltung von KI-Systemen

11.4 Prozesssicht auf KI-Vorhaben anstatt befristete Projektsicht

11.5 Data Scientists als Architekten der KI-Wertschöpfung

11.6 Zusammenfassung und Ausblick

12 Recht im Zeitalter synthetischer Realitäten: Überdenken rechtlicher Normen für generative KI

12.1 Einleitung

12.2 Realitätszerstörung: die epistemische Krise von Deepfakes und synthetischen Inhalten

12.3 Synthetische Subjekte: die Regulierung nicht menschlicher Akteure

12.4 Von individuellen Rechten zu kollektiven Schäden

12.5 Die rechtliche Vorstellungskraft in der Krise: Fragmentierung, Soft Law und strukturelle Grenzen

12.6 Den Weg in die Zukunft weisen: Lex Syntheticum als Regulierungsprototyp

13 Verantwortung und Erkenntnis

13.1 Einleitung

13.2 Reflexion als notwendige Praxis

13.3 Perspektiven auf Verantwortung im Zeitalter generativer KI

13.4 Zusammenfassung

14 Arbeiten mit KI: Kollege oder Konkurrent?

14.1 KI-Einführung – ein Alltagsbeispiel

14.2 Vom Widerstand zur Wurzel: das KI-Paradoxon

14.3 Centaur vs. Cyborg: zwei Denkmodelle für die Zusammenarbeit von Mensch und KI

14.3.1 Das Centaur-Modell: Mensch und KI als Team mit klarer Arbeitsteilung

14.3.2 Das Cyborg-Modell: Mensch und KI als eng verwobene Einheit

14.3.3 Welches Modell ist wann sinnvoll?

14.4 Prinzipien wirksamer Augmentierung: das TRICUS-Modell

14.4.1 Sechs Prinzipien des TRICUS-Modells

14.4.2 TRICUS als Gestaltungsrahmen für KI-Projekte

14.5 Der Wandel von Aufgabenprofilen durch KI

14.5.1 Entry-Level-Dilemma

14.5.2 Von der Karrierepyramide zum Karrierediamant

14.5.3 Jobverlust durch KI? Zwischen Realität und Rhetorik

14.6 Handlungsempfehlungen für Führungskräfte

14.7 Fazit: KI als Kollege – wenn wir sie dazu machen

Teil III Anwendungsfälle aus der Praxis

15 Einführung einer Enterprise-GPT-Plattform bei der Provinzial Versicherung

15.1 ChatGPT als Disruption oder das Henne-AI-Problem

15.2 Herausforderungen bei der Einführung von ChatGPT

15.2.1 Neue dynamische Technologie

15.2.2 Make or Buy

15.2.3 Versicherungsbranche

15.2.4 Vorbereitung der Organisation: Vorträge, Schulungen und Change

15.2.5 Regulatorik

- 15.2.6 Besondere Herausforderung bei der Provinzial: #uP-IT-Programm
 - 15.3 Strategische Vorgehensweise bei der Umsetzung
 - 15.3.1 Flächendeckende Implementierung
 - 15.3.2 Unternehmen im Unternehmen
 - 15.3.3 Agnostische Umsetzung
 - 15.3.4 Strategische Partnerschaften
 - 15.4 Vorgehen bei der technologischen Umsetzung
 - 15.4.1 Provinzial GPT (Basis)
 - 15.4.2 Dokumentenchat
 - 15.4.3 Retrieval-Augmented Generation
 - 15.4.4 APIs und API-basierte Erweiterungen
 - 15.4.5 ProHub
 - 15.5 Organisatorische Umsetzung
 - 15.5.1 KI-Regulativ: Prozesse für die Klassifizierung des Risikos nach EU AI Act und der Mitbestimmungsrelevanz
 - 15.5.2 KI-Lösungsteam
 - 15.6 Ausblick
- 16 KI-gestützte Analyse geldpolitischer Kommunikation bei der Bundesbank
 - 16.1 Geldpolitische Kommunikation: von manueller zur KI-gestützten Analyse
 - 16.2 Von Kontext bis Konsistenz: Anforderungen an KI in der geldpolitischen Analyse
 - 16.2.1 Kontextverständnis
 - 16.2.2 Nachvollziehbarkeit
 - 16.2.3 Konsistenz
 - 16.3 Monetary-Intelligent Language Agent
 - 16.3.1 Vorteile
 - 16.3.2 Funktionsweise
 - 16.4 MILA in der Praxis

16.4.1 Ergebnisse

16.4.2 Implementierung

16.4.3 Evaluation

16.5 Zusammenfassung

17 KI für das Wissensmanagement in der Energiewirtschaft

17.1 Einleitung

17.2 Traditionelle Wissensmanagementansätze und ihre Grenzen

17.3 Retrieval-Augmented Generation für den niederschweligen Zugang zu Unternehmensinformationen

17.3.1 Von Unternehmensdokumenten zum RAG-System

17.3.2 Referenzarchitektur eines RAG-Systems

17.3.3 Integration von strukturierten Informationen und Live-Daten

17.4 Herausforderungen von RAG-Systemen in der Energiewirtschaft: Sicherheit und Compliance im KRITIS-Kontext

17.5 Anwendungsfall für die Energiewirtschaft

17.5.1 Chatbot-Zugriff auf Unternehmens- und Sensordaten

17.5.2 Automatisierte Inhaltsanalyse von Rechtstexten

17.6 Zusammenfassung und Ausblick

18 KI-Bohrer: KI-gesteuerte Lärmreduzierung für urbane Geothermiebohrungen

18.1 Einleitung

18.2 Forschungsstand: generative Modelle und Deep Reinforcement Learning

18.3 Assistenzsystem

18.4 Drill-LSTM-Modell: Vorhersage von Maschinenzuständen für optimierte Bohrprozesse

18.5 Sound-GAN: Echtzeitprognose der Schallausbreitung für Lärmminimierung

18.6 DRL-Agent: intelligente Steuerungsempfehlung für lärmoptimierte Bohrprozesse

18.7 Zusammenfassung

19 Digitalisierung im Sozialwesen: KI-gestützte Vertragsprüfung

19.1 Einleitung

19.2 Legal Tech: eine kurze Einordnung

19.3 Die Ausgangslage: analoge Prozesse und steigende Anforderungen

19.4 Die Lösung: KI-gestützte Webapplikation zur Vertragsprüfung

19.4.1 Formelle Prüfung – strukturierte Analyse durch KI

19.4.2 Inhaltliche Prüfung – intelligente Fragen und Compliance

19.4.3 Blick ins System: intuitive Bedienung und hohe Effizienz

19.4.4 Zahlen und Fakten

19.4.5 Erste Erfahrungen

19.5 Zusammenfassung und Ausblick

20 KI für den Klimaschutz – ClimaClic als zukunftsweisendes Beispiel

20.1 Einleitung

20.2 ClimaClic – innovative Klimalotterie mit digitalem Fokus

20.3 Kooperation als Erfolgsfaktor

20.4 Automatisierung im Dialogmarketing bei ClimaClic

20.4.1 Generative KI als Produktivitätshebel im Dialogmarketing

20.4.2 System Prompts als Schlüssel zum automatisierten Content

20.5 Zusammenfassung und Ausblick

21 Erfolgreiche Robotik-Integration: Kompetenzaufbau und Verankerung von KI-gestützter Robotik in Unternehmen

21.1 Einleitung

21.2 Strategische Robotikkompetenz in Unternehmen aufbauen

21.3 Change Management bei der Robotik-Einführung

21.4 Synergien zwischen Mensch und Roboter stärkenorientiert umsetzen

21.5 Schlussfolgerung

Anhang

Abkürzungen

Literaturverzeichnis

Autorenteam

Index

Autoren

4 Künstliche Intelligenz in der Computer Vision

Stefan Hensel

In diesem Kapitel stellen wir die grundlegenden Begriffe und Besonderheiten der Erkennung von Inhalten in Bildern vor. Wir erläutern das Prinzip sowie Anwendungen der künstlichen Intelligenz in der Computer Vision mit tiefen neuronalen Netzen.

4.1 Einordnung und Historie

Das Erkennen von Sinn und Informationen aus der Umgebung mithilfe des Sehannes gehört zu den wichtigsten Eigenschaften des Menschen. Maschinen sollen dieses Vermögen durch Kameras und rechnergestützte Verfahren nachahmen. Diese Aufgabe bezeichnet man als *maschinelles Sehen* (engl. *Computer Vision*). Der Begriff grenzt sich bewusst von der digitalen Bildverarbeitung ab, bei der die mathematische Änderung der Bildinhalte und nicht das Verstehen des Bildinhaltes im Fokus steht. Die Bilderkennung hat eine lange Tradition. In den vergangenen Jahrzehnten hat sie besonders durch den Einsatz und die Weiterentwicklung von Verfahren der künstlichen Intelligenz, vor allem durch tiefe neuronale Netze, entscheidende Fortschritte gemacht.

Schon in den 1960er-Jahren begannen erste Forschungsarbeiten zur automatischen Erkennung von Bildinhalten. Erste Ansätze arbeiteten mit geometrischen Abstraktionen der Welt und stellten diese mithilfe von Linien und Flächen dar. Die Entwicklung der Algorithmen wurde von Erkenntnissen des biologischen Sehens begleitet, insbesondere von der Funktionsweise von Zellen der Netzhaut und der Verarbeitung von Sehsignalen im visuellen Cortex von Säugetieren [Hubel & Wiesel 1959]. Dies führte in den 1980-Jahren zu Methoden zur Erkennung von Objekten und Bewegung in Bildern durch die Beschreibung von Merkmalen wie Ecken und Texturen in Bildern. Im nächsten Schritt erfolgte eine modellbasierte Nachbildung des menschlichen Sehens mithilfe von hierarchisch angeordneten Schichten, die jeweils rezeptive Felder, also die lokalen Bereiche auf der Netzhaut, der vorhergehenden Schicht verarbeiten [Fukushima 1980; LeCun et al. 1998]. Auf diese Weise gelangt man von einfachen Mustern in den ersten Schichten zu komplexen Strukturen und Objekten in den späteren Schichten. Die ersten Erfolge in diesem Bereich waren aufgrund

fehlender Rechenleistung und einer zu geringen Datenbasis für die Anpassung der Parameter nicht von Dauer. Die Entwicklung des maschinellen Sehens fokussierte sich daher auf statistische Methoden und manuell definierte Bildmerkmale. Dies änderte sich schlagartig und nachhaltig in den 2010er-Jahren mit dem Erfolg von *Deep Learning*, einem Durchbruch im Feld tiefer neuronaler Netze [Krizhevsky et al. 2012].

4.2 Eigenschaften digitaler Bilder

Die Anwendung neuronaler Netze im maschinellen Sehen basiert auf einer Vielzahl grundlegender Konzepte und Eigenschaften von Bildern. Digitale Bilder weisen eine hohe Informationsdichte auf: Jedes einzelne Bildelement bzw. Pixel repräsentiert mehrdimensionale Daten, die Position auf einem Gitter und den eigentlichen Farbwert. Letzterer entspricht bei Grauwertbildern der Intensität und bei Farbbildern den Werten der jeweiligen Farbkanäle. In Abbildung 4.1 ist ein exemplarisches Pixel für ein Bild I mit seiner Datenstruktur dargestellt. Einzelne Pixel haben eine zweidimensionale Position $I(u,v)$ innerhalb des Bildes mit Höhe H (Zeilen) und Breite B (Spalten). Farbbilder bestehen aus mehreren Kanälen C , die jeweils in der Dimension $H \times B$ liegen. Ein Bild ist somit eine dreidimensionale Datenstruktur der Größe $H \times B \times C$.

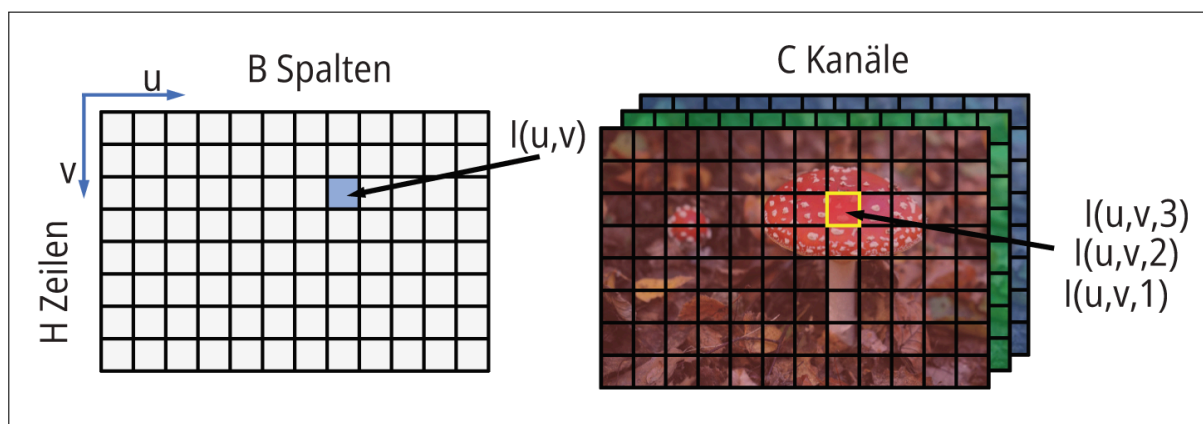


Abb. 4.1 Datenstruktur digitaler Bilder

Im Gegensatz zu Daten, wie sie in der Sprachsignalverarbeitung oder Zeitreihenanalyse vorliegen, führt dies zu einer Explosion der Parameterzahl vollverknüpfter neuronaler Netze. Betrachtet man beispielsweise ein Netz mit 1.000 Eingangsneuronen, so kann dieses die Klassifizierung von mehreren Hundert Silben durchführen. Wird dieses Netz um zwei weitere Schichten von 500 Neuronen und einer möglichen Ausgabe von 1.000 Ergebnissen erweitert, ergeben sich rund 1,25 Mio. Parameter für das gesamte Netz. Wird dieser naive Ansatz bei Bildern angewandt, beispielsweise für eine (relativ geringe) Auflösung von 800×600 Bildpunkten und drei Farbkanälen, liegen bereits in der ersten

Schicht $800 \times 600 \times 3 \times 500 = 720$ Mio. Parameter vor. Dies macht den Einsatz spezieller Architekturen für neuronale Netze notwendig.

Der durchschlagende Erfolg der künstlichen Intelligenz bei der Interpretation von Bildern ist nicht nur auf den Einsatz neuer Architekturen für neuronale Netze¹ und die Verfügbarkeit ausreichender und immer größer werdender Rechenleistung in Form von GPU²-Grafikkarten zurückzuführen. Die Kombination aus Rechenleistung und freien Parametern führt jedoch nur dann zum Erfolg, wenn für das Lernen der Parameter eine ausreichende Menge an Daten in Form eines Trainingsdatensatzes zur Verfügung steht. Für das Bildverstehen mit KI-Methoden werden daher große annotierte Bilddatenbanken benötigt, die aufwendig in der Erstellung sind. Erst die Verfügbarkeit dieser Datenbanken ermöglicht das grundlegende Training der neuronalen Netze, die dann mittels spezifischer Bilder an das jeweilige Problem angepasst werden.

4.3 Neuronale Netze im maschinellen Sehen

Nahezu alle KI-unterstützten Verfahren des Bildverstehens bestehen ausschließlich oder teilweise aus sogenannten *Faltungsnetzen* (engl. *Convolutional Neural Networks*, CNNs). In jüngerer Zeit werden diese durch eine zweite Netzarchitektur ergänzt: die sogenannten *Transformer*. Aufgrund ihrer erfolgreichen Anwendung im Bereich des natürlichen Sprachverstehens und der großen Sprachmodelle (engl. *Large Language Models*, LLMs) werden sie auch für die Bilderkennung adaptiert. Beide Ansätze sind aufgrund ihres Aufbaus den *Deep-Learning-Methoden* in der künstlichen Intelligenz zuzuweisen, was sie von den klassischen Methoden des maschinellen Lernens abgrenzt.

4.3.1 Faltungsnetze

Faltungsnetze bilden das Rückgrat des KI-basierten maschinellen Sehens. Wie im vorherigen Abschnitt beschrieben, verbietet sich aufgrund der Struktur von Bildern die Verwendung simpler *Mehrschichtperzeptronen* (engl. *Multi-Layer Perceptron*, MLP). Die Lösung besteht in der mathematischen Operation der Faltung, die in der Bildverarbeitung traditionell im Rahmen der Filterung eingesetzt wird. Die Neuronen des Netzes sind dadurch nicht mehr mit jedem Neuron der vorhergehenden Schicht verknüpft, sondern teilen sich über die Filter die Parameter, was deren Anzahl massiv reduziert.

Die Grundstruktur eines CNN orientiert sich an der Struktur des menschlichen Cortex und baut auf frühen Arbeiten aus den 1980er-Jahren auf.

Eine besonders erfolgreiche und prägende Struktur ist das LeNet, das von Yann LeCun [LeCun et al. 1998] bereits in den 1990er Jahren beschrieben wurde. Es konnte aber erst durch Entwicklungen im Bereich der Lernalgorithmen, der Rechnerhardware sowie der Verfügbarkeit großer Trainingsdatensätze für praxisrelevante Bilder angepasst werden – was letztendlich mit der Entwicklung von AlexNet [Krizhevsky et al. 2012] zum Durchbruch für den Einsatz tiefer neuronaler Netze in der Bilderkennung führte.

4.3.1.1 Grundlegender Aufbau von Faltungsnetzen

Ein typisches CNN besteht aus drei grundlegenden Bausteinen:

- Faltungsschicht
- Pooling-Schicht
- Vollverknüpfte Schichten mit Ausgabeschicht

In der *Faltungsschicht* erfolgt die eigentliche Verarbeitung der Bildsignale. Dieser Vorgang wird in der Anwendung auch als Filterung bezeichnet. Dabei handelt es sich um eine lokale Operation auf einem Bild, d. h., der Wert eines Ausgangspixels wird von einer Menge an Eingangspixeln in einer begrenzten Bildregion R bestimmt. Die Bildelemente in der Region des Eingangsbildes I werden durch eine gewichtete Summe kombiniert, sodass für das Ausgangsbild O gilt:

$$O(u, v) = \sum_{(i,j) \in R} I(u + i, v + j) \cdot K(i, j),$$

wobei die Filterparameter als Gewichte bezeichnet werden und in der Filtermaske oder einfach dem Filter K definiert sind. Eine exemplarische Faltung mit einer Filtermaske der Größe 3×3 (dies entspricht neun Parametern oder Gewichten) ist für einen Bildausschnitt der Größe 4×4 Pixel in [Abbildung 4.2](#) dargestellt. Diese Operation wird nun für jedes Pixel des Eingangsbildes ausgeführt, indem die Filtermaske über alle Zeilen und Spalten des Bildes bewegt wird. Die schraffierten Flächen kennzeichnen den lokalen Eingangsbildbereich R , den Filter K und das Ausgabepixel, das durch die gewichtete Summe $0 \times 1 + 0 \times 1 + 2 \times 3 + 0 \times 2 + 3 \times 1 + 4 \times 0 + 4 \times 2 + 8 \times 0 + 6 \times 1 = 23$ berechnet wird.

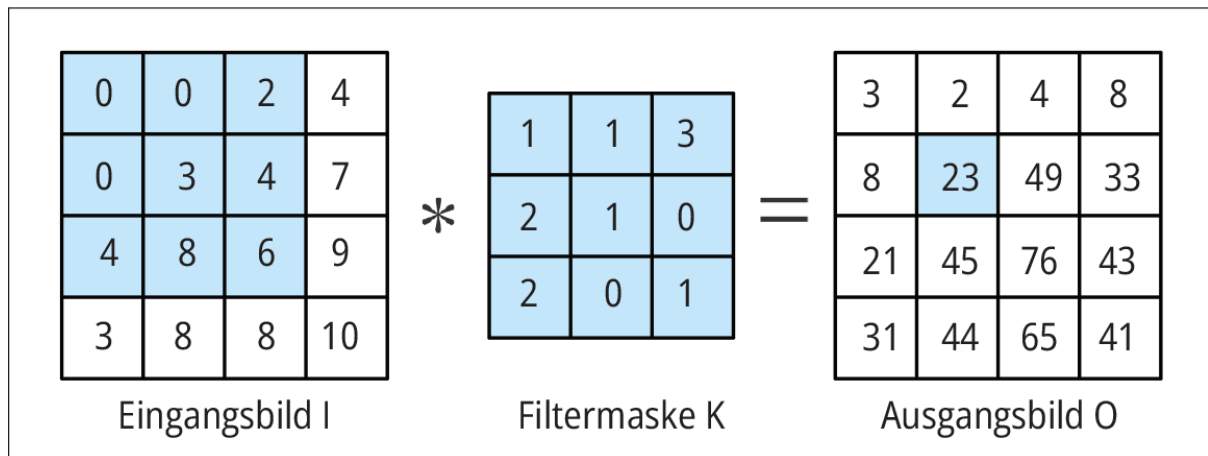


Abb. 4.2 Filterung als zweidimensionale Faltung

Durch die Wahl der Parameter erzeugt die Faltung verschiedene Effekte im Bild, wie z. B. das Einbringen einer Unschärfe. Selbst mit kleinen Filtern der Größe 5x5 lassen sich bereits Differenzenquotienten, zweite Ableitungen oder Kombinationen dieser realisieren. Ein besonders interessanter Effekt ist die hohe Empfindlichkeit, mit der die Filter auf Strukturen ansprechen, die eine Ähnlichkeit mit der Filtermaske aufweisen. Dieser Effekt des korrelativen Nachweises der Ähnlichkeit zweier Signale oder Bildbereiche wird im Englischen als *Template Matching* bezeichnet und stellt ebenfalls eine grundlegende Operation in der digitalen Bildverarbeitung dar. Die Filter reagieren durch ihren Aufbau sensitiv auf ähnliche Bildmerkmale. So verstärkt eine horizontale Filtermaske ähnliche Bereiche des Eingangsbildes und unterdrückt abweichende Strukturen, was sich durch eine Unschärfe ausdrückt. Das Ausgabebild jeder Faltungsoperation wird als *Merkmalskarte* (engl. *Feature Map*) bezeichnet und dient als Eingangsbild für die Faltung in der nächsten Schicht des neuronalen Netzes.

Da die Bilder als mehrdimensionale Matrizen vorliegen (siehe [Abb. 4.1](#)), müssen für die Verwendung in CNNs n mehrdimensionale Filtermasken je Schicht eingesetzt werden. Die typische Größe der betrachteten Eingangsbildregion ist hierbei klein und liegt im Bereich von 3x3 bis 7x7 Bildelementen. Jede Faltungsschicht erzeugt somit n Merkmalskarten als Ausgang. Ein Beispiel für n=4 Filter ist in [Abbildung 4.3](#) gezeigt. Die Eingangsdaten weisen das Format H1 x B1 x C1 mit C1=6 Kanälen auf. Die vier Filter im Format 3x3x6 berechnen die Ausgangsmerkmalskarten, die als Eingang in der nächsten Schicht verarbeitet werden, dann entsprechend mit C2=4 Kanälen.

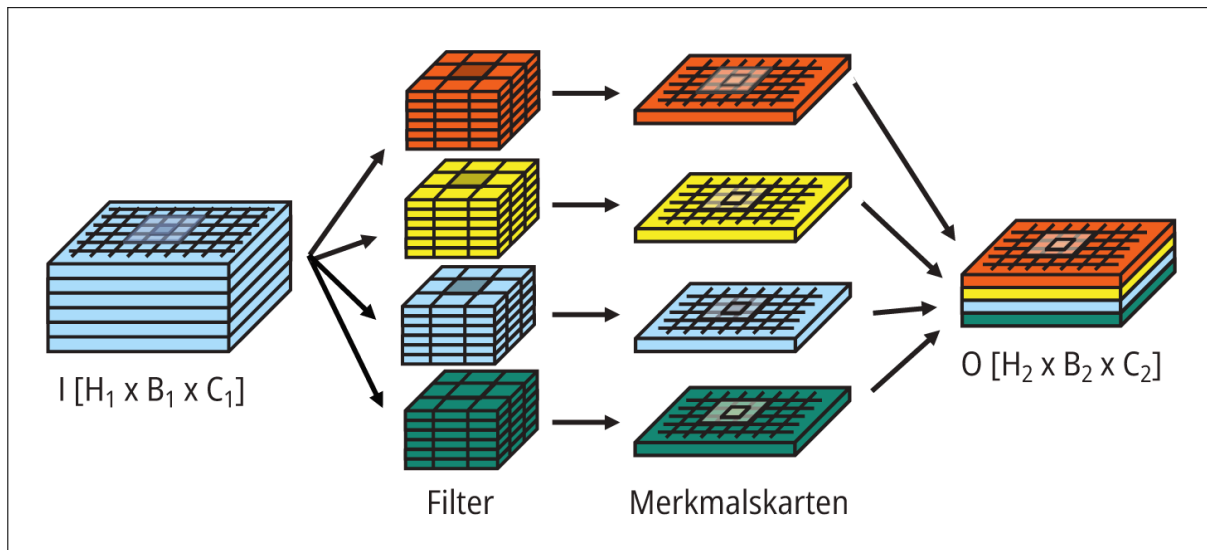


Abb. 4.3 Faltungsschicht mit $n = 4$ Filtermasken und korrespondierenden Merkmalskarten

Als Eingangsbild liegen die Merkmalskarten der vorherigen Schicht vor, deren Kanäle der Anzahl der Filter in dieser Schicht entsprechen. Am Ausgang erhält man n Merkmalskarten, die als Stapel in einer Matrix der Größe $H_2 \times B_2 \times C_2$ angeordnet sind. Dabei entspricht die Anzahl der Kanäle C_2 der Filteranzahl n und die Höhe H und Breite B der Ausgabe dem Eingangsbild abzüglich der halben Größe der Filtermaske. Durch die aufeinanderfolgende Anwendung von Filtern wird das Bild stetig verkleinert und der Bildinhalt auf jeder Ebene zunehmend abstrahiert. Durch diese Anordnung wird in höheren Schichten je Neuron ein immer größerer Bereich des Eingangsbildes interpretiert³. Die Merkmalskarten weisen hierbei immer komplexere Muster auf, die aus unteren Schichten zusammengesetzt sind und als Abstraktion für die Erkennung dienen. Dieser Aufbau ist an den visuellen Cortex von Säugetieren angelehnt und bildet dessen Informationsverarbeitung nach.

Die Größe der Merkmalskarten und der damit verbundene Rechenaufwand ist auch in späteren Schichten noch sehr groß. Dies wird durch den zweiten Grundbaustein, die *Pooling-Schicht*, aufgelöst. Wie in der Faltungsschicht kommen in der Pooling-Schicht lokale Pooling-Kerne zum Einsatz. Diese weisen im einfachsten Fall die Größe 2×2 auf und werden mit einer Schrittweite von zwei über das Bild bewegt, dies bedeutet, dass in der Verarbeitung ein Pixel übersprungen wird. Mathematisch erfolgt keine Faltung, sondern die Berechnung des Mittelwertes (*meanpooling*) oder der Maximalwert der Region wird als deren Ausgang definiert (*maxpooling*). Der wichtigste Effekt des Einfügens einer Pooling-Schicht in das neuronale Netz ist die durch die Schrittweite bedingte Reduktion der Größe der Merkmalskarten, im Beispiel des 2×2 -Kerns und einer Schrittweite von zwei um den Faktor vier, wodurch sich die

benötigte Rechenleistung für Auswertung und Training der neuronalen Netze massiv reduziert. Zudem wirken sich Pooling-Schichten positiv auf leichte Veränderungen des Eingangsbildes aus: Durch die Verschmelzung der Bildbereiche wird eine gewisse Robustheit gegenüber kleinen Verschiebungen und Rotationen erreicht.

Den Abschluss eines typischen Faltungsnetzes bilden die *vollverknüpften Schichten*. Diese entsprechen einem gewöhnlichen Mehrschichtperzeptron (MLP) und schließen sich an die letzte Merkmalskarte L an. Diese wird in einen Zeilenvektor der Länge $B_L \times H_L \times C_L$ gewandelt, der als Eingang des MLP dient. Die letzte Schicht des Netzes ist die Ausgabeschicht, die in ihrer Dimension an das gegebene Problem angepasst ist. Soll beispielsweise ein Bild in eine von vier möglichen Klassen eingeordnet werden, weist die Ausgabeschicht vier Neuronen auf. Diese geben über ihre Aktivierungsfunktion⁴ die Klassenzugehörigkeit als Wahrscheinlichkeit aus.

Diese drei Grundbausteine werden in mehreren Schichten angeordnet, in der Regel beginnend mit Blöcken aus einer Faltungsschicht gefolgt von einer Pooling-Schicht. Die vollverknüpften Schichten am Ende werden auf die Größe der letzten Merkmalskarten und die gewünschte Ausgabe angepasst. In Abbildung 4.4 ist der Aufbau eines typischen Faltungsnetzes für die Klassifizierung eines Bildes in vier mögliche Klassen dargestellt.⁵ Faltungs- (grün gestrichelt) und Pooling-Schichten (blau) wechseln sich ab, die vollverknüpften Schichten am Ende entsprechen einem MLP und geben die Wahrscheinlichkeiten der Klassen aus. Dargestellt sind die Merkmalskarten und Neuronen des MLP.

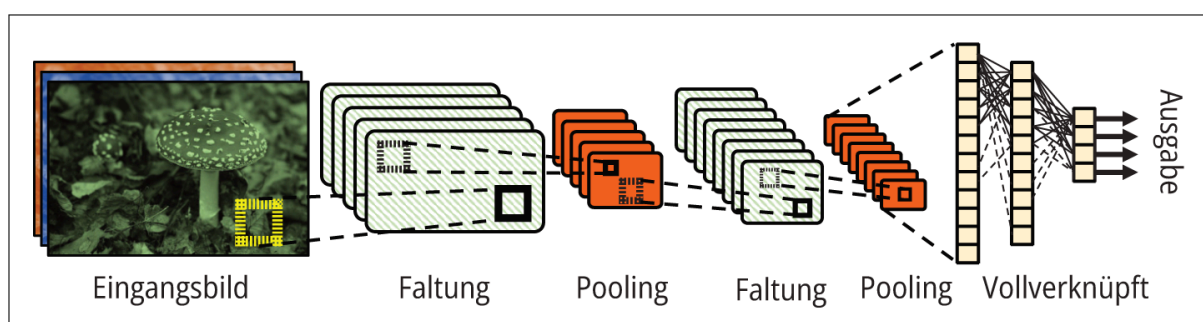


Abb. 4.4 Aufbau eines CNN aus Grundbausteinen

4.3.1.2 Evolution der Basisarchitektur

Die bisher beschriebene Struktur eines CNN wurde von Krizhevsky [Krizhevsky et al. 2012] erweitert und zum ersten Mal erfolgreich für Klassifikationsaufgaben in praktischen Anwendungen verwendet. Das beschriebene Netz nutzte im

Gegensatz zur Basisvariante Neuerungen wie kleinere Filtergrößen und die Erweiterung auf elf Schichten anstelle der vorherigen sieben Schichten. Für die Verarbeitung mit GPUs wurde zudem mit der *Rectified Linear Unit* (ReLU) eine neue Aktivierungsfunktion für Neuronen eingeführt.

Die Verwendung kleinerer Filter, die in mehreren Schichten angeordnet sind, führte in den folgenden Jahren zu einer stetigen Vergrößerung der Netze – von 16 und 19 Schichten letztendlich auf 152 Schichten. Das Trainieren dieser tiefen neuronalen Netze ist nur durch den Einsatz sehr großer Bilddatenbanken und algorithmische Neuerungen für das Training möglich. In GoogLeNet wurden Inception-Blöcke eingeführt, die die einfachen Faltungen durch mehrere parallel angeordnete Filter pro Schicht ersetzen. Die gewonnene Effizienz erlaubt noch tiefere Netze mit mehr Schichten. Mit der ResNet-Architektur von He et al. [He et al. 2015] wurden die sogenannten *Skip-Connections* vorgestellt. Diese Methode ermöglicht das Training von immer tieferen Netzen und reduziert die Anzahl der Parameter dabei deutlich, ohne an Genauigkeit einzubüßen – zugleich war es das erste Netz, das in einer Klassifikationsaufgabe bessere Ergebnisse als menschliche Probanden erzielte. In Tabelle 4.1 sind die Netzarchitekturen mit ihrer Bezeichnung, der Anzahl der Schichten und den daraus resultierenden Parametern aufgelistet. Als Maß der Leistungsfähigkeit ist die Fehlerrate in einem Klassifikationswettbewerb mit 1.000 möglichen Klassen angegeben.

Netzarchitektur	Schichten	Parameteranzahl	Fehlerrate	Jahr
AlexNet	7	~ 60 Mio.	16,4%	2012
VGG16	16	~ 118 Mio.	7,3%	2014
GoogLeNet (Inception-v1)	19	~ 6,8 Mio.	6,7%	2014
ResNet-152	152	~ 60,2 Mio.	3,6%	2015
GoogLeNet (Inception-v4)	78	~ 43 Mio.	3,1%	2016
<i>Mensch</i>			5,1%	

Tab. 4.1 Entwicklung der Basisarchitekturen der Faltungsnetze, jeweils in ihrer letzten Iteration

Die letzte Iteration ist das Inception-v4-Netz [Szegedy et al. 2017], das die Inception-Architektur mit den Skip-Connections von ResNet kombiniert und dadurch eine hohe Effizienz und Genauigkeit erreicht.

4.3.2 Vision Transformer

Faltungsnetze bilden den aktuellen Standard in allen praxisrelevanten Problemstellungen und finden, je nach Anwendungsgebiet, in Form verschiedener Archetypen weitreichende Verwendung. Ein zweiter Typus neuronaler Netze basiert auf der *Transformer-Architektur* nach [Vaswani et al. 2017], die nach dem erfolgreichen Einsatz in der natürlichen Sprachverarbeitung für alle Bereiche des maschinellen Lernens adaptiert wurde.

Transformernetze, die mit Bilddaten arbeiten, werden als *Vision Transformer* (ViT) bezeichnet und sind erstmals in [Dosovitskiy et al. 2021] beschrieben. Im Gegensatz zu CNNs, die mittels einer Kaskade lokaler Filter in den oberen Schichten des Netzwerks komplexe Bildstrukturen abstrahieren, nutzen die ViTs den Attention-Mechanismus, um den globalen Kontext eines Bildes zu erfassen. Die Grundidee von Struktur und Daten eines ViT zeigt [Abbildung 4.5](#). Unten links sehen Sie die exemplarische Bildeinteilung in Patches (3x3), das Embedding (Inhalt und Position) der Patches ist durch die gepunktete Umrandung dargestellt, der Encoder mit dem Aufbau eines einzelnen Encoder-Blocks ist rechts und die Ausgabe nach MLP oben links zu sehen.

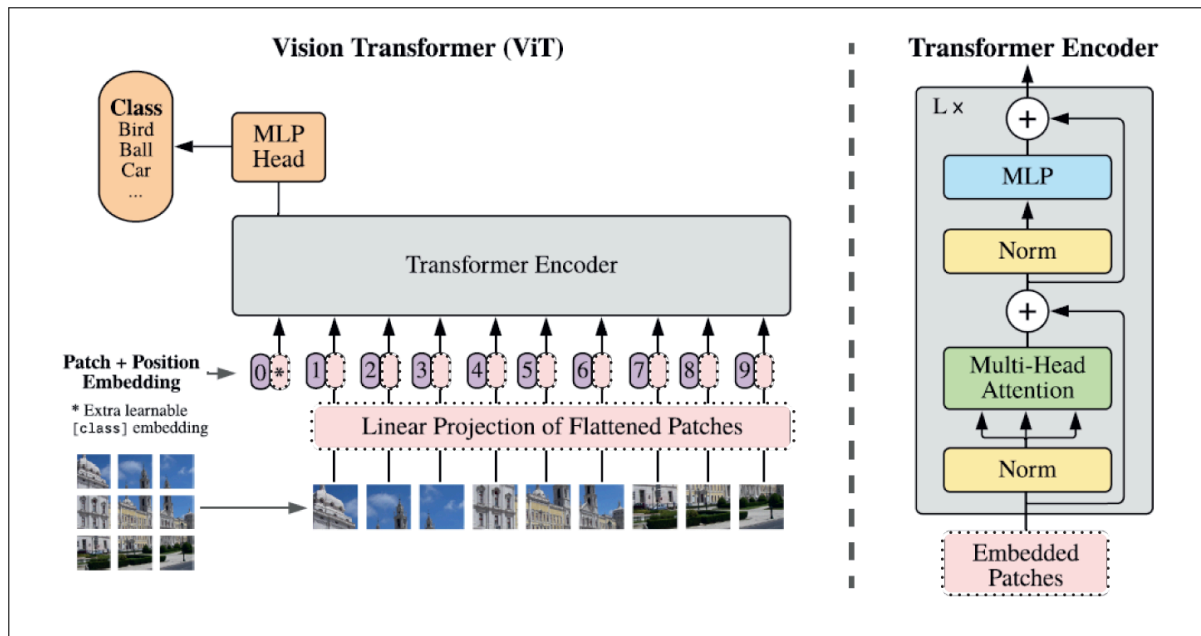


Abb. 4.5 Aufbau eines Vision Transformers (ViT) (nach [Dosovitskiy et al. 2021])

Das Eingangsbild wird in gleich große, nicht überlappende Bereiche (engl. *Patches*, z. B. der Größe 16x16 Pixel) unterteilt. Anschließend werden diese vektoriell – und damit analog zu Wörtern – in den *Embedding-Raum* projiziert. Dieser hochdimensionale Raum ergänzt die Bildmerkmale durch eine explizite Positionsinformation der Bildbereiche. Anschließend werden diese Embeddings in der Standardarchitektur des Transformers weiterverarbeitet. Dieser als Encoder bezeichnete Teil ist aus identischen Encoder-Blöcken aufgebaut, die den Attention-Mechanismus nutzen, um den Kontext des gesamten Bildes, unabhängig von der räumlichen Nähe der Patches, für die Erkennungsaufgabe heranzuziehen. Ein einzelner Block besteht aus mehreren Attention-Heads gefolgt von einem MLP und ist in [Abbildung 4.5](#) auf der rechten Seite zu sehen. Die Anzahl der Encoder-Blöcke entspricht der Modelltiefe und kann daher analog den Schichten bei CNNs betrachtet werden. Die Auswahl stellt einen Kompromiss aus Komplexität des Modells bei vorhandener Trainingsdatenmenge dar.

In Klassifizierungsaufgaben bildet der ViT-Base mit zwölf Encoder-Blöcken den Standard. In Anlehnung an [Tabelle 4.1](#) erhält man für diesen Vision Transformer (ViT-B/16, 2021) rund 86 Mio. Parameter bei einer Fehlerrate von 4 %. ViTs sind somit in ihrer Leistungsfähigkeit mit CNNs vergleichbar, erzielen aber umso bessere Ergebnisse, je größer die verfügbare Datenmenge⁶ ist. Stehen die Daten zur Verfügung, weisen ViTs eine sehr hohe Generalisierungsfähigkeit auf und benötigen einen minimalen Aufwand im Fine-Tuning.

Die Größe des benötigten Trainingsdatensatzes stellt dabei den größten Nachteil der ViT-Architektur dar. Das Training der Parameter ist mit einem massiven Aufwand in Zeit und Hardware verbunden. Die hohe Anzahl an Parametern und die daraus resultierende Berechnungsdauer verbietet zudem den Einsatz in Anwendungen mit begrenzten Rechnerressourcen oder hohen Geschwindigkeitsanforderungen.

4.4 Anwendungen im maschinellen Sehen

Die bisher vorgestellten CNNs und ViTs bestehen aus den beschriebenen Grundelementen. Diese können für eine gegebene Aufgabenstellung in geeigneter Weise kombiniert, adaptiert und trainiert werden. Im maschinellen Sehen können die Anwendungen in vier Kategorien eingeteilt werden: die *Klassifizierung* von Bildern, die *Detektion von Objekten* in Bildern, die *Segmentierung*, eine pixelgenaue Erkennung und Markierung von Objekten, sowie die *Anomalieerkennung*,⁷ bei der Abweichungen von einer gelernten Grundinstanz erkannt werden.

Da die Art der Anwendung domänenübergreifend ist, ist die Verwendung aufgabenspezifischer Standardarchitekturen möglich. In der Praxis wird für jede Aufgabe die passende Netzarchitektur mit generisch vortrainierten Parametern gewählt, die dann mit einem kleinen Bilddatensatz auf die vorliegende Problemstellung angepasst wird.

4.4.1 Fine-Tuning von Standardarchitekturen

Neuronale Netze im maschinellen Sehen weisen ausnahmslos eine hohe Parameterzahl auf. Das Training dieser Parameter erfordert den Einsatz massiver Ressourcen in Form von Zeit und Hardware. Die benötigten Trainingsdatensätze, d. h. vorannotierte Bilder, zählen in die Hunderttausende und müssten für jeden Anwendungsfall spezifisch erstellt werden. Der Aufbau von CNNs ermöglicht eine erhebliche Erleichterung für den Einsatz in der Praxis durch die Nutzung vortrainierter Modelle, die mithilfe relativ weniger (in der Regel sind es Hunderte Datensätze anstelle von Tausenden) domänenspezifischer Datensätze an die gegebene Anwendung angepasst werden.

Dieser als *Fine-Tuning* bezeichnete Prozess nutzt die Tatsache, dass die Netze in den unteren Schichten auf rudimentäre Merkmale, wie vertikale und horizontale Muster sowie Farbkombinationen, reagieren und erst in den höheren Ebenen komplexere Strukturen abbilden. Die vollverknüpften Schichten am Ende der CNNs nutzen die Merkmale der unteren Schichten, um die Erkennung

auf die geforderten Klassen durchzuführen. Im Fine-Tuning werden genau diese Schichten problemspezifisch ersetzt und ihre Gewichte mit dem deutlich kleineren Datensatz neu trainiert. Die vortrainierten Netze sind im Netz zugänglich und können unter Berücksichtigung der Anforderungen (z. B. höchste Genauigkeit, aber auch effiziente oder geringe Berechnungskosten) bezogen und angepasst werden.

4.4.2 Klassifizierung von Bildern

Die Klassifizierung eines Bildes in eine von mehreren vorgegebenen Klassen ist eine der häufigsten und grundlegendsten Anwendungen im maschinellen Sehen. Für diese Aufgabe bestehen die Netze aus den in [Abschnitt 4.3.1](#) beschriebenen Grundelementen, angeordnet als Abfolge von Faltungs- und Pooling-Schichten sowie einem vollverknüpften Netz (MLP) am Ende des CNN. Optimierungen der Basisarchitektur sind in [Tabelle 4.1](#) aufgeführt und zeigen die Leistungsfähigkeit der Netze für eine Beispielklassifizierung. Diese beruht auf dem ImageNet-Datensatz [Deng et al. 2009], der aus rund einer Million annotierten Bildern besteht, die in 1.000 mögliche Klassen eingeteilt sind.

Für eine Klassifizierungsanwendung mit hoher Genauigkeit kann das mit Image-Net vortrainierte Inception-v4-Netz heruntergeladen und mittels Fine-Tuning an die vorliegende Klassenzahl angepasst werden.

4.4.3 Objektdetektion

Die Detektion bezeichnet die örtliche Lokalisierung und Klassifizierung von Objekten in einem Bild. Jede Objektinstanz wird im Bild durch einen Rahmen, die sogenannte *Bounding-Box*, markiert. Die Markierung in Kombination mit der wahrscheinlichsten Klasse des detektierten Objekts wird als Ergebnis durch das Netz ausgegeben. Ein typisches Resultat ist in [Abbildung 4.6](#) dargestellt.

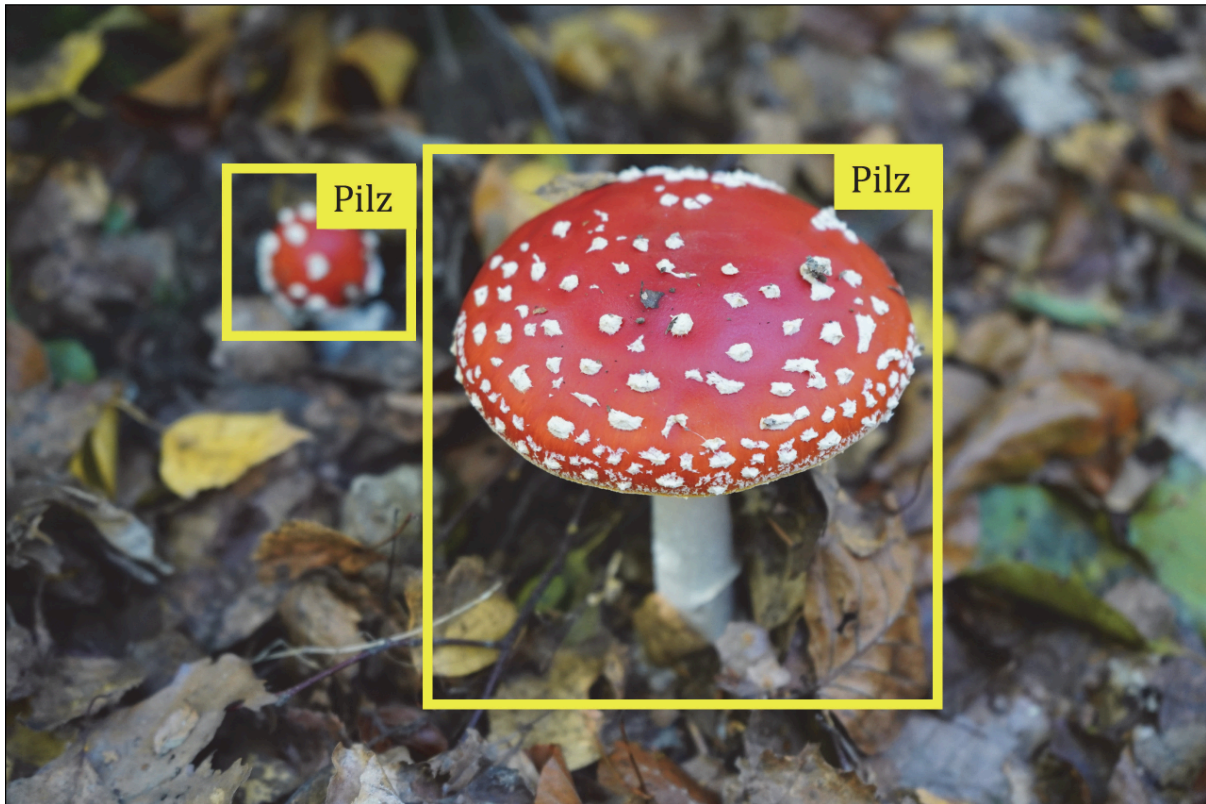


Abb. 4.6 Grundprinzip der Objektdetektion mit Bounding-Box und Objektklasse

Den Goldstandard in der Objektdetektion bilden Faltungsnetze, deren Grundstruktur um mehrere spezielle Schichten, die sogenannten *Heads*, erweitert wird. Diese erlauben das Markieren der gefundenen Objekte. Aus historischer Sicht nutzten die ersten Objektdetektoren auf CNN-Basis einen gitterbasierten Ansatz, in dem die Objekte in diskreten Bereichen der letzten Merkmalskarte des Netzes erkannt wurden. Diese CNNs werden als *Backbone* bezeichnet und weisen den gleichen Aufbau wie Klassifikationsnetze auf. Häufig wird das vortrainierte ResNet-50 genutzt und ohne die vollverknüpfte Schicht für die Erzeugung generischer Merkmale eingesetzt. Der bekannteste Vertreter dieser ersten Generation ist das YOLO-Netz nach Redmon et al. [Redmon et al. 2016]. Obwohl diese Art der Detektoren eine effiziente Architektur nutzt, zeigt der gitterbasierte Ansatz Nachteile bei der Variation der Objektgröße und auch in der maximalen Anzahl detektierbarer Objekte im Bild. Nachfolgende Generationen (YOLOv2 bis YOLOv4) nutzten daher einen ankerbasierten Ansatz, der eine spezifische Anzahl vordefinierter Bounding-Boxen je Bildpunkt erlaubt. Nachteilig an diesem Ansatz sind die Vielzahl von Parametern, das manuelle Tuning und die hohe Berechnungskomplexität.

Die jüngste Variante der Objektdetektoren zeichnet sich durch hohe Detektionsgenauigkeit, höchste Effizienz und Skaleninvarianz⁸ aus, verzichtet auf Objektanker und wird als vollständig faltungsbasierter Einstufendetektor (engl. *FCOS*) bezeichnet. Das CNN lernt direkt auf Basis der Daten, an welchen

Merkmalspunkten sich vermeintliche Objekte befinden. Die Bounding-Boxen werden als Werte direkt ausgegeben und in zusätzlichen Faltungsschichten, den Regression Heads, ermittelt. Die grundlegende Architektur dieser Detektoren ist in Abbildung 4.7 gezeigt. Das CNN für die Merkmalsgenerierung als Backbone ist in Blau ganz links dargestellt, die Pyramidenschicht für Skaleninvarianz mittig in Grün und die Heads zur Ausgabe der Bounding-Box und Detektionsergebnisse rechts in Orange.

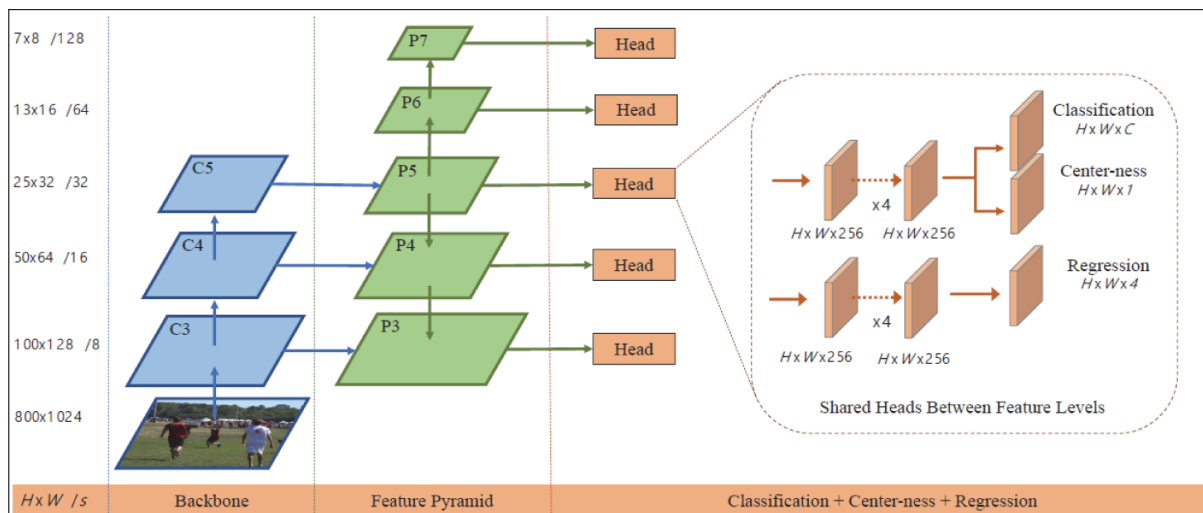


Abb. 4.7 Architektur von FCOS (von [Tian et al. 2019])

Das Backbone-Netz, das die Merkmalskarten generiert, wird durch eine Pyramidenstruktur, in der die abstrahierte Information von höher liegenden Merkmalskarten auch in tieferen Schichten genutzt wird, ergänzt. Objekte werden so in verschiedenen Größen erkannt. Die Ergänzung der Faltungsschichten durch die Heads, ist in der rechten Bildhälfte dargestellt. In Form einer Faltung werden direkt die Ergebnisse der Klassifizierung und die Werte der Bounding-Boxen als Position, Seitenlänge und Wahrscheinlichkeit eines Objektes ausgegeben.

Diese Detektoren werden der FCOS-Familie [Tian et al. 2019] zugeordnet. Die verbreitetsten Vertreter dieser Architektur sind die YOLO-Netze ab YOLOv8 [Varghese & Sambath 2024]. Für die praktische Anwendung stehen wiederum vortrainierte Objektdetektornetze für das Fine-Tuning zur Verfügung. Diese sind auf dem COCO-Datensatz⁹ vortrainiert, der aus rund 200.000 Bildern (annotiert mit rund 500.000 Objektinstanzen) in 80 Klassen besteht. Im Fine-Tuning werden die Klassifikations- und Bounding-Box-Heads für die gewünschten Objektklassen angepasst.

4.5 Zusammenfassung und Ausblick

Auf Basis der vorgestellten Modelle und Methoden gelten Klassifizierung und Detektion als gelöstes Problem. Der Prozess aus vortrainierten Archetypen und anschließendem Fine-Tuning löst die Mehrheit der praktischen Problemstellungen. Die Erweiterung der Netzarchitekturen mit zusätzlichen Heads erlaubt die Verwendung von CNNs in anderen Anwendungen. Die jüngste Generation von Objektdetektoren gibt neben den Bounding-Boxen und Objektklassen auch die pixelgenaue Segmentierung des Bildes aus. Eine andere kleine Ergänzung erlaubt die Detektion spezifischer Merkmalspunkte (engl. *Keypoints*), mit deren Hilfe die Pose von Menschen oder die Orientierung von Objekten im Raum erkannt werden.

Die größte Änderung im maschinellen Sehen mit KI-Methoden ist auf den Erfolg der Transformer-Architektur in Form von ViTs zurückzuführen. Neben der Klassifizierung werden ViTs mit *Detection Transformer* (DETR) [Carion et al. 2020] auch erfolgreich in der Objektdetektion eingesetzt. Insbesondere die Generalisierungsfähigkeit der Transformer, erreichbar durch die Nutzung massiver Trainingsdatensätze, stellt einen Vorteil gegenüber den CNNs dar.

Foundation Models

Der Erfolg von ViTs und ihr Ursprung in der Verarbeitung von Sprache ermöglichen den Aufbau *multimodaler Modelle*. Multimodal bezieht sich auf die Form der Trainingsdaten, die neben der Bildform nun auch als Text oder Video vorliegen können. Der CLIP-Transformer von Yu et al. [Yu et al. 2022] erreicht durch das Training von Text- und Bilddaten auf massiven Datenbanken eine hohe Generalisierungsfähigkeit und kann selbst auf unbekanntem Bildern und Klassen hervorragende Ergebnisse erzielen, was als *Zero-Shot-Fähigkeit* bezeichnet wird. Der Ansatz ermöglicht das Training massiver Modelle mit mehreren Milliarden Parametern, die durch ein minimales Fine-Tuning (auf das im Extremfall verzichtet werden kann) die Ergebnisse spezialisierter Netze erzielen oder übertreffen. Modelle dieser Art werden auch als *Foundation Models* bezeichnet und bilden den Goldstandard in allen Anwendungen ohne Limitierung der Rechenressourcen. Bedeutende Vertreter sind das DINO-Modell [Oquab et al. 2024] und das Segment Anything Model (SAM) aus [Ravi et al. 2024], ein spezielles Foundation Model für die Bildsegmentierung, das beliebige Objekte in Bildern pixelgenau erkennt. Foundation Models repräsentieren einen Paradigmenwechsel im maschinellen Sehen, weg von spezialisierten Netzen, hin zu generalisiert vortrainierten Netzen für eine Vielzahl von Aufgaben.

Die Ähnlichkeit in Architektur und Training von Foundation Models und Large Language Models (LLMs), wie z. B. ChatGPT, ist frappierend. Die LLMs, die ursprünglich für den Einsatz in sprachlichen Anwendungen entwickelt wurden,

sind in ihrer aktuellen Ausprägung ebenfalls multimodal. Mit einer bisher nicht möglichen Parameterzahl im Bereich Hunderter Milliarden, entsprechen sie Foundation Models und verfügen über eine nie dagewesenen Generalisierungsfähigkeit. Neben allen klassischen Anwendungen des maschinellen Sehens können LLMs auch eingesetzt werden, um Bilder zu generieren, Fragen zu Bildern im Sinne des Bildverstehens zu beantworten und gezielte Manipulation der Bilddaten durchzuführen.

Index

A

Agent [40](#)
Agent, autonomer [162](#)
AGI *siehe* [Artificial General Intelligence 21](#)
Amazon SageMaker Model Monitor [90](#)
Anomalieerkennung [65](#)
Applikations-Layer [245](#)
Approximate-Nearest-Neighbor-Algorithmus [244](#)
Artificial General Intelligence [21](#)
Assistenzsystem [160](#)
Attention-Mechanismus [52](#)
Ausgleichsrechnung [34](#)
Automated Guided Vehicles [79](#)
Automatisierung [85](#), [293](#)
 im Marketing [272](#)
Automatisierungsteam [287](#)

B

Backpropagation-Algorithmus [38](#)
Bag-of-Words-Modell [46](#)
bestärkendes Lernen *siehe* [Reinforcement Learning 27](#)
Binning [116](#)
 benutzerdefiniertes [118](#)
 gleicher Breite [117](#)
 gleicher Frequenz [118](#)
BSI-Gesetz [246](#)

C

Centaur-Modell [197](#)
Change-Impact-Analyse [290](#)
Change-Management [213](#), [263](#), [288](#)
ChatGPT [21](#), [209](#)

ClimaClic [272](#)
Closed-Loop-Verfahren [75](#)
Cloud Application Framework [94](#)
Computer Vision [57](#)
Container-Virtualisierung [86](#)
Content-Automatisierung [271](#)
Content-Erstellung [271](#)
Convolutional Neural Network [59](#)
CRISP-DM [94](#)
CRISP-ML(Q) [94](#)
Cross-Industry Standard Process for Data Mining *siehe* [CRISP-DM 94](#)
Custom GPT [148](#)
Cyborg-Modell [198](#)

D

Dark AI [144](#)
DASC-PM [94](#)
 Domäne [105](#)
 IT-Infrastruktur [105](#)
 Wissenschaftlichkeit [106](#)
Data Augmentation [111](#)
Data Scientist [167](#)
Data Splitting [119](#)
Daten, unstrukturierte [35](#)
Datenbalancierung [120](#)
Datenbereinigung [112](#)
Datenbereitstellung [118](#)
Datenbeschaffung [110](#)
Datenbias [54](#)
Datendiskretisierung [116](#)
Datengerechtigkeit [179](#)
Datenintegration [119](#)
Datenqualität [112](#)
Datentransformation [114](#)
Datenvalidierung [112](#)
Datenvorbereitung [109](#)
 Kategorien [110](#)
Deep Learning [35](#), [59](#)
Deep Reinforcement Learning [253](#)
Deepfake [169](#)
DevOps [162](#)

Dialogmarketing [276](#)
Diamond-Shaped Workforce [203](#)
Digital Operational Resilience Act *siehe* [DORA 214](#)
Digital Services Act [173](#)
Digitalisierung [263](#)
Distribution Shift [88](#)
Docker [86](#)
Dokumentenmanagementsystem [242](#)
DORA [214](#)
DRL *siehe* [Deep Reinforcement Learning 253](#)
DSA *siehe* [Digital Services Act 173](#)
DSGVO [182](#)
Dublette [112](#)

E

Effizienzprojekt [150](#)
Eliminierungsstrategie [113](#)
Embedded-Methode [122](#)
EMRK *siehe* [Europäischen Menschenrechtskonvention 170](#)
Energiewirtschaft [246](#)
Entry-Level-Dilemma [203](#)
Ethics Washing [181](#)
Ethik [185](#), [187](#)
EU AI Act [24](#), [131](#), [171](#)
 Prüfschema [134](#)
Europäischen Menschenrechtskonvention [170](#)
Evidently [90](#)

F

Faltungsnetz [59](#)
 Aufbau [60](#)
 Faltungsschicht [60](#)
 Pooling-Schicht [62](#)
 vollverknüpfte Schicht [62](#)
Faltungsschicht [60](#)
Feature Learning [35](#)
Feature Map [61](#)
Featureraum [28](#)
Few-shot Prompting [233](#)
Filtermethode [121](#)

Foundation Model [69](#), [159](#)
Fuzzy Frontend of Innovation [154](#)

G

Generative Adversarial Network (GAN) [254](#)
Generative KI [143](#), [276](#)
Geothermiebohrung [253](#)
Geräuschreduzierung [253](#)
Geschäftsmodell [152](#)
Gradientenabstieg [37](#)

H

Hawk-O-Meter [234](#)
Head of AI [123](#)
Hub-and-Spoke-Prinzip [137](#)
Human-in-the-Loop-Ansatz [250](#)

I

Imputationsverfahren [113](#)
 Beispiel [114](#)
In-Context Learning [52](#)
Informationsextraktion [45](#)
Informationssicherheits-Managementsystem [247](#)
inkonsistente Daten [112](#)
Insurtech [213](#)
Integer Encoding [116](#)

K

KI-Agent [149](#), [246](#)
KI-Einheit [123](#)
 Aufbau [124](#)
 Aufgaben [125](#), [128](#)
 Erfolgsfaktoren [138](#)
KI-Fähigkeit [158](#)
KI-gestützte Robotik [281](#)
KI-gestützte Steuerung [253](#)
KI-Governance [180-181](#)
KI-Hub [136-137](#)
KI-Lösungsteam [224](#)
KI-Paradoxon [196](#)

KI-Ressource [158](#)
Klassifikationsproblem [30](#)
Klimaschutz [271](#)
Kreuzvalidierung [119](#)
KRITIS [246](#)
Künstliche Intelligenz [21](#)
 Ethik [185](#), [187](#)
 Implementierungsprozess [274](#)
Künstliche neuronale Netze (KNN) [35](#)

L

Label [28](#)
Label Encoding *siehe* [Integer Encoding 116](#)
Large Language Model [52](#), [185](#), [211](#), [249](#), [281](#)
Legal Tech [263](#)
Lemmatisierung [45](#)
Lex Syntheticum [181](#)
Lineares Regressionsmodell [32](#)
Living Guidelines (EU) [190](#)
Lizenz [144](#)
LLM-Gateway [222](#)
LLM-Layer [245](#)
Lossfunktion [32](#)

M

Machine Learning [27](#), [83](#)
Maschinelles Sehen *siehe* [Computer Vision 57](#)
Mehrheitsklasse [120](#)
Mehrschichtperzeptron [59](#)
Merkmalskarte [61](#)
Microservice [84](#)
Microservices-Plattform [86](#)
MILA [230](#)
Minderheitsklasse [119](#)
Minimum Viable Product [147](#)
Missing Value [112](#)
ML *siehe* [Machine Learning 27](#)
ML-Modell [83](#), [86](#)
ML-Operations [90](#)
Modell-Fine-Tuning [242](#)

Modellkompetenz [144](#)
Monetary-Intelligent Language Agent *siehe* [MILA 230](#)
Monitoring [86](#)
 ML-Systeme [86](#)

N

Natural Language Processing [43](#)
 Bias [53](#)
Need for Automation [281](#)
Neukundengewinnung [273](#)
Neuproduktprojekt [151](#)
New Concept Development Model [154](#)
NLP *siehe* [Natural Language Processing 43](#)
Nominaldaten [115](#)
Normstrategie [283](#)
Nudging [190](#)

O

One-Hot Encoding [49](#), [116](#)
OpenAI Chat Completions-Standard [212](#)
Operationalisierung [83](#)
Optimierungsverfahren [33](#)
Orchestration-Layer [245](#)
Ordinaldaten [115](#)
organisationaler Mechanismus [159](#)
Overfitting [33](#)
Oversampling [120](#)

P

Plausibilitätsprüfung [112](#)
Pooling-Schicht [62](#)
Prompt Chaining [234](#)
Prompt Engineering [52](#), [144](#), [278](#)
Prompt-Datenbank [147](#)
Prompting [242](#)
Proof of Concept [147](#)
Prozessmodell [157](#), [166](#)
Python [84](#)

R

RAG *siehe* Retrieval-Augmented Generation
Rectified Linear Unit 63
Regressionsproblem 30
Reinforcement Learning 27, 74
 Action Space 74
 Control Space 75
 Observation Space 74
 Reward Function 75
 Trainingsituation 75
Repräsentationsbias 54
Research-Umgebung 129
Reshoring 281
Responses API 212
ressourcenbasierter Ansatz 283
Retrieval-Augmented Generation 53, 132, 148, 160, 220, 242–243
 Monitoring 248
Robo-Debt 185
Roboter 71
 Entscheidungen 73
 Lebenszyklus 71
Robotik 281
Robotikkompetenz 282, 284
Role-based Prompting 233

S

Self-Supervised Learning 34
Sentimentanalyse 232
SMART-Modell 293
Soft Law 180
Stakeholder-Analyse 289
Stemming 45
Stop Word Removal 45
Supervised Learning 27–28, 83
System Prompt 148, 276

T

Team Data Science Process 94
Temporal Tagging 46
Term-Dokumenten-Matrix 47
TF-IDF-Wert 48

Three-Lines-of-Defense-Modell [132](#)

Tokenisierung [44](#)

 Normalisierung [45](#)

Trainingsdaten [28](#)

Transformer [52](#)

Transparenzpflicht [172](#)

TRICUS-Modell [200](#), [202](#)

U

überwachtes Lernen *siehe* [Supervised Learning 27](#)

Undersampling [120](#)

unstrukturierte Daten [35](#), [44](#)

Upskilling [144](#)

V

VAIT [214](#)

Vektordatenbank [245](#)

Vektordatenbank-Layer [245](#)

Vektorraum-Modell [47](#)

Vendor Lock-in [211](#)

Versicherungsaufsichtliche Anforderungen an die IT *siehe* [VAIT 214](#)

Vertragsprüfung [263](#)

Vision Transformer [64](#)

vollverknüpfte Schicht [62](#)

Vortraining [38](#)

W

Wertschöpfung [141](#)

Wertschöpfungsmodell [142](#)

Wissensgenerierung [160](#)

Wissensmanagement [242](#)

Word Embedding [50](#)

kontextualisiertes [52](#)

Workshop [145](#)

Wrapper-Methode [121](#)

Z

Zipfsches Gesetz [48](#)