

BACHELORARBEIT

Contrastive Learning mit Stable Diffusion-basierter Datenaugmentation

Verbesserung der Bildklassifikation
durch synthetische Daten

vorgelegt am 23. September 2024
Paul Hofmann

Erstprüferin: Prof. Dr. Larissa Putzar
Zweitprüfer: Prof. Dr. Jan Neuhöfer

**HOCHSCHULE FÜR ANGEWANDTE
WISSENSCHAFTEN HAMBURG**
Department Medientechnik
Finkenau 35
22081 Hamburg

**FRAUNHOFER-INSTITUT FÜR
PRODUKTIONSANLAGEN UND
KONSTRUKTIONSTECHNIK IPK**
Pascalstraße 8–9
10587 Berlin

Zusammenfassung

Diese Bachelorarbeit untersucht die Verbesserung der Bildklassifikation durch synthetische Daten im Supervised Contrastive Learning (SCL). Ziel ist es, die Eignung von DA-Fusion, einer Stable Diffusion-basierten Methode zur Datenaugmentation, für die Generierung synthetischer Daten in einem Anwendungsfall aus der Recyclingwirtschaft zu evaluieren. Daraüber hinaus wird untersucht, wie generierte Near Out-of-Distribution (OOD)-Daten in das Supervised Contrastive Learning integriert werden können, um als negativ-Beispiele zu dienen und die Repräsentationen der In-Distribution-Daten weiter zu verbessern. Die Ergebnisse zeigen, dass DA-Fusion geeignete In-Distribution-Augmentationen liefert, welche die Klassifikationsleistung verbessern. Jedoch führte die Integration von Near OOD-Daten im SCL zu einer Verschlechterung der Modellgenauigkeit, da die Beispiele oft zu weit von den In-Distribution-Daten entfernt waren. Die Arbeit hebt die Herausforderungen bei der Generierung und Nutzung synthetischer Daten hervor und bietet Ansätze zur Optimierung von Contrastive Learning in diesem Kontext.

Abstract

This bachelor thesis investigates how image classification can be improved using synthetic data for Supervised Contrastive Learning (SCL). The aim is to evaluate the suitability of DA-Fusion, a stable diffusion-based data augmentation method, for generating synthetic data in a use case from the recycling industry. It also investigates how synthetic near out-of-distribution (OOD) images can be integrated into supervised contrastive learning to serve as negative examples in order to further improve the representation of in-distribution data. The results show that DA-Fusion provides suitable in-distribution augmentations that improve classification performance. However, the integration of near OOD data in SCL led to a deterioration of model accuracy, as the examples were often too dissimilar from the in-distribution data. The thesis highlights the challenges of generating and using synthetic data and offers approaches to optimise contrastive learning in this context.

Inhaltsverzeichnis

Abbildungsverzeichnis	III
Tabellenverzeichnis	IV
1 Einleitung	1
1.1 Motivation	1
1.2 Zielsetzung	2
1.3 Aufbau der Arbeit	3
2 Theoretische Grundlagen	5
2.1 Maschinelles Lernen	5
2.1.1 Überwachtes und unüberwachtes Lernen	6
2.1.2 Deep Learning	7
2.1.3 Neuronale Netze	8
2.1.4 Datenaugmentation	10
2.1.5 Out-of-Distribution Daten	11
2.2 Synthetische Daten	12
2.2.1 Variational Autoencoder	13
2.2.2 Generative Adversarial Networks	15
2.2.3 Diffusionsmodelle	16
2.2.4 DA-Fusion	18
2.3 Contrastive Learning	19
2.3.1 Unsupervised Contrastive Learning	20
2.3.2 Supervised Contrastive Learning	21
2.4 Klassifikation von Gebrauchsgegenständen in der Recyclingwirtschaft	22
2.4.1 Herausforderungen bei der Generierung synthetischer Daten	23
2.4.2 Synthetische Daten als negativ-Beispiele im Contrastive Learning	24
2.4.3 Integration von DA-Fusion und Supervised Contrastive Learning	25
3 Methodisches Vorgehen	27
3.1 MVIP-Datensatz	27
3.1.1 Teildatensatz	27
3.1.2 Vorverarbeitung	28

3.2	Implementierung	28
3.2.1	DA-Fusion	29
3.2.2	Supervised Contrastive Learning	30
3.3	Versuchsaufbau	31
3.3.1	Synthetische Datengenerierung	31
3.3.2	Trainings- und Testdurchläufe	32
3.3.3	Evaluationsmethoden und Metriken	32
4	Ergebnisse	34
4.1	Die generierten synthetischen Daten	34
4.1.1	In-Distribution	34
4.1.2	Near Out-of-Distribution	36
4.2	Trainings- und Testergebnisse	37
4.2.1	Contrastive Pre-Training	37
4.2.2	Lineare Klassifikation	38
4.3	Vergleich der Ergebnisse	40
4.3.1	Mit und ohne In-Distribution-Augmentationen	40
4.3.2	Mit und ohne Near Out-of-Distribution-Augmentationen	41
5	Diskussion	42
5.1	Eignung von DA-Fusion für die synthetische Datengenerierung	42
5.2	Wirksamkeit von Near Out-of-Distribution-Augmentationen im Supervised Contrastive Learning	43
6	Fazit	45
6.1	Zusammenfassung der wichtigsten Erkenntnisse	45
6.2	Beantwortung der Forschungsfragen	46
6.3	Ausblick und potenzielle Weiterentwicklungen	46
Literatur		48
Anhang		51

Abbildungsverzeichnis

2.1	Ein einfaches Venn-Diagramm, das die Beziehung zwischen KI, ML und DL veranschaulicht.	7
2.2	Schematische Abbildung eines künstlichen Neurons.	8
2.3	Architektur eines Convolutional Neural Networks.	10
2.4	Einige Beispiele für unterschiedliche Datenaugmentationstechniken und Kombinationen, die in (Chen et al., 2020) verwendet wurden.	11
2.5	Aufbau eines Autoencoders.	13
2.6	Überblick über das GAN-Training.	15
2.7	Darstellung der Vorwärts- und Rückwärtsdiffusion in einem Diffusionsmodell.	17
2.8	Die Mensch-Maschine-Schnittstelle des Systems.	23
3.1	Beispielbilder aus dem MVIP-Datensatz, die auf die Region of Interest (ROI) zugeschnitten wurden.	28
4.1	Vergrößerte Ausschnitte von einigen der In-Distribution-Augmentationen. .	35
4.2	Beispiele für mangelhafte In-Distribution-Augmentationen.	35
4.3	Beispiele der Near Out-of-Distribution-Augmentationen.	36
4.4	Beispiele für mangelhafte Out-of-Distribution-Augmentationen.	37
4.5	Trainings- und Validierungsfehler während des Contrastive Pre-Trainings. .	38
4.6	Trainings- und Validierungsfehler der linearen Klassifikation.	39
4.7	Trainings- und Validierungs-Accuracy während der linearen Klassifikation. .	39
4.8	ID- und OOD-Confidence während der linearen Klassifikation.	40

Tabellenverzeichnis

4.1 Testergebnisse der linearen Klassifikation.	38
---	----

1 Einleitung

Es folgt zunächst eine Einleitung in das Thema der Arbeit, die Motivation für die Untersuchung, die Zielsetzung und der Aufbau der Arbeit.

1.1 Motivation

Die rasanten Entwicklungen in den Bereichen der Künstlichen Intelligenz (KI) und des Maschinellen Lernens (ML) decken eine breite Palette von Anwendungen ab, die an immer mehr Stellen in unserem Alltag Einzug halten. In den vergangenen Jahren ist insbesondere die generative Modellierung in ein neues Zeitalter eingetreten, wie durch die Popularität von Sprachmodellen wie ChatGPT oder Bildgeneratoren wie DALL-E und Stable Diffusion gezeigt wird. Diese generativen Modelle sind in der Lage, menschenähnliche Texte zu schreiben, fotorealistische Bilder zu erzeugen und sogar Videos und Musik zu kreieren, wodurch sie die Art und Weise verändern, wie KI mit kreativen Prozessen interagiert.

Auch für industrielle Anwendungen hat diese Entwicklung eine immense Bedeutung, insbesondere im Bereich des Maschinellen Sehens, bei dem es darum geht, visuelle Informationen zu verarbeiten und zu interpretieren. Es ist die Grundlage vieler automatisierter Prozesse, wie etwa der Qualitätskontrolle in Produktionslinien, der Objekterkennung in autonomen Fahrzeugen oder der Fehlererkennung in medizinischen Bildern. Ein zentrales Ziel dabei ist es, Modelle zu entwickeln, die in der Lage sind, zuverlässig Muster in visuellen Daten zu erkennen und daraus Entscheidungen zu treffen.

Ein wesentlicher Faktor für den Erfolg dieser Modelle ist die Verfügbarkeit von Daten. Industrielle Anwendungsfälle sind allerdings von domänen spezifischen Anforderungen und einer hohen Komplexität geprägt, weshalb herkömmliche Ansätze der Datenbeschaffung oft an ihre Grenzen stoßen. Die manuelle Erfassung und Annotation von Daten ist zeitaufwendig, teuer und in vielen Fällen nicht praktikabel. Zudem sind die verfügbaren Datenmengen in hochspezialisierten Anwendungen oft zu gering oder zu homogen, um ein Modell mit der nötigen Robustheit und Generalisierungsfähigkeit zu trainieren. Dies führt dazu, dass die Modelle häufig in realen Umgebungen nicht die gewünschte Leistung erzielen, da sie mit Out-of-Distribution (OOD)-Daten konfrontiert werden, also mit Daten, welche nicht aus der gleichen Verteilung stammen wie die Trainingsdaten.

Hier kommen synthetische Daten ins Spiel, die durch generative Modelle künstlich erzeugt werden. Der Vorteil dieser synthetischen Daten liegt darin, dass sie in beliebiger Menge und Varianz produziert werden können, um die Datenvielfalt zu erhöhen und das Modell auf verschiedenste Szenarien vorzubereiten. Trotz dieser Vorteile ist auch die synthetische Datengenerierung von den Herausforderungen der industriellen Anwendungen betroffen und scheitert oft bei der realistischen Darstellung komplexer, domänenspezifischer Konzepte aus einer limitierten Menge an Beispielen (Lu et al., 2024).

Vor diesem Hintergrund befasst sich diese Arbeit mit der Untersuchung der auf Stable Diffusion basierenden Methode DA-Fusion (Trabucco et al., 2023), welche eine effektive Generierung semantischer Augmentationen verspricht, ohne dass das vortrainierte Stable Diffusion-Modell Vorwissen zu den gegebenen Konzepten haben muss. Dies ist besonders relevant, um die Herausforderungen der Variabilität der Domänenspezifik zu bewältigen.

Darüber hinaus wird in dieser Arbeit der Fokus auf die Integration dieser synthetischen Daten in Contrastive Learning gelegt, einer Methode, die in den letzten Jahren immer mehr an Bedeutung gewonnen hat. Contrastive Learning zielt darauf ab, repräsentative Merkmale von Daten zu lernen, indem es ähnliche Datenpunkte in einem hochdimensionalen Merkmalsraum näher zusammenbringt und unähnliche weiter voneinander entfernt. Diese Technik erweist sich als besonders nützlich, um robuste und generalisierbare Repräsentationen zu erlernen.

1.2 Zielsetzung

Das Hauptziel dieser Arbeit besteht darin, die Eignung der Stable Diffusion-basierten Augmentationsmethode DA-Fusion zur Generierung synthetischer Daten für einen spezifischen Anwendungsfall in der Recyclingwirtschaft zu untersuchen und deren Einfluss auf die Bildklassifikation zu bewerten. Die Analyse fokussiert sich auf zwei zentrale Aspekte: Erstens wird untersucht, ob DA-Fusion in der Lage ist, hochwertige synthetische Augmentationen zu erzeugen, die zur Verbesserung der Generalisierungsfähigkeit von Bildklassifikationsmodellen beitragen können. Zweitens steht die Integration von Near Out-of-Distribution (Near OOD)-Augmentationen in das Supervised Contrastive Learning (SCL) im Mittelpunkt, um zu evaluieren, welchen Einfluss diese speziellen Augmentationen auf die Robustheit der Modellleistung haben.

In industriellen Anwendungsbereichen wie der Recyclingwirtschaft ist die Verfügbarkeit großer und vielfältiger Datensätze oft limitiert. Daher stellt die Erzeugung synthetischer Daten durch DA-Fusion eine vielversprechende Methode dar, um die Datenbasis zu erweitern und die Leistungsfähigkeit von Bildklassifikationsmodellen zu verbessern. Ein wesentlicher Aspekt ist dabei die Frage, ob durch diese generierten Daten eine höhere Generalisierungsfähigkeit

erreicht werden kann, d.h., ob Modelle, die mit augmentierten Datensätzen trainiert werden, besser auf bisher ungesehene Daten reagieren können.

Neben den Standard-Augmentationen soll auch untersucht werden, inwiefern Near OOD-Augmentationen, also Daten, die sich leicht von den In-Distribution-Daten unterscheiden, aber nicht vollständig aus einer fremden Domäne stammen, einen Mehrwert für das Supervised Contrastive Learning bieten. Contrastive Learning ist darauf ausgerichtet, robuste Repräsentationen zu lernen, indem ähnliche Daten näher zusammen und unterschiedliche Daten weiter voneinander entfernt im latenten Raum positioniert werden. Hierbei stellt sich die Frage, ob Near OOD-Daten als sogenannte *Hard Negatives* –also besonders herausfordernde Beispiele –die Fähigkeit des Modells verbessern können, zwischen In-Distribution- und OOD-Daten zu unterscheiden und somit die Robustheit gegenüber unbekannten Datenverteilungen zu erhöhen.

Die zentralen Forschungsfragen, die in dieser Arbeit adressiert werden, lauten daher:

- Kann DA-Fusion synthetische Augmentationen für den gegebenen Anwendungsfall in der Recyclingwirtschaft erzeugen, die die Generalisierungsfähigkeit im Supervised Contrastive Learning verbessern
- Trägt die Verwendung von Near OOD-Augmentationen im Supervised Contrastive Learning dazu bei, die Robustheit des Modells gegenüber OOD-Daten zu erhöhen und die Repräsentationen von In-Distribution-Daten weiter zu verbessern?

Die Beantwortung dieser Fragen soll nicht nur zur Verbesserung der Klassifikationsmodelle im spezifischen Kontext der Recyclingwirtschaft beitragen, sondern auch generell Aufschluss darüber geben, wie synthetische Daten und OOD-Augmentationen die Modellleistung in der Bildverarbeitung beeinflussen können.

1.3 Aufbau der Arbeit

Die vorliegende Arbeit gliedert sich in sechs Kapitel.

In [Kapitel 2](#) werden zunächst die theoretischen Grundlagen gelegt, die das Verständnis für die darauffolgenden Kapitel ermöglichen. Es beginnt mit einer Einführung in das Maschinelle Lernen, wobei sowohl überwachte als auch unüberwachte Lernverfahren erklärt werden, bevor tiefer auf Deep Learning und neuronale Netze eingegangen wird. Ein besonderes Augenmerk wird dabei auf die Bedeutung der Datenaugmentation und das Konzept der Out-of-Distribution-Daten gelegt. Im Anschluss daran wird auf die verschiedenen Ansätze zur Generierung synthetischer Daten eingegangen, darunter Variational Autoencoder (VAE), Generative Adversarial Networks (GANs) und Diffusionsmodelle, wobei der Fokus auf Stable

Diffusion und DA-Fusion liegt. Das Kapitel schließt mit einer ausführlichen Betrachtung von Contrastive Learning, insbesondere des Supervised Contrastive Learning (SCL), sowie der Beschreibung des Anwendungsfalls zur Klassifikation von Gebrauchsgegenständen in der Recyclingwirtschaft. Hier werden die Herausforderungen bei der Generierung synthetischer Daten, die Möglichkeit zur Nutzung synthetischer Daten als negative Beispiele im SCL und der eigene Ansatz zur Integration von DA-Fusion in das SCL beschrieben.

[Kapitel 3](#) beschreibt das methodische Vorgehen. Hier wird zunächst der verwendete MVIP-Datensatz detailliert erläutert, der verwendete Teildatensatz definiert und die notwendigen Vorverarbeitungsschritte erklärt. Danach wird auf die Implementierung der DA-Fusion-Methode und des Supervised Contrastive Learning eingegangen. Das Kapitel schließt mit einer Beschreibung des Versuchsaufbaus, der sowohl die Generierung der synthetischen Daten mittels DA-Fusion als auch die Trainingsdurchläufe und die verwendeten Evaluationsmethoden umfasst.

In [Kapitel 4](#) werden die Ergebnisse der Arbeit präsentiert. Zunächst werden die erzeugten synthetischen Daten vorgestellt und in zwei Kategorien eingeteilt: In-Distribution- und Near-Out-of-Distribution-Daten. Daraufhin folgen die Ergebnisse der Trainings- und Testdurchläufe, die das Pre-Training mittels Supervised Contrastive Learning und die darauf aufbauende lineare Klassifikation betreffen. Im Anschluss werden die erzielten Resultate mit und ohne In-Distribution-Augmentationen sowie mit und ohne Near OOD-Augmentationen miteinander verglichen.

[Kapitel 5](#) widmet sich der Diskussion der Ergebnisse. Hier wird insbesondere die Eignung von DA-Fusion zur Generierung synthetischer Daten bewertet. Zudem wird die Wirksamkeit der Near Out-of-Distribution-Daten im Rahmen des Supervised Contrastive Learning diskutiert, um zu analysieren, ob diese zur Verbesserung der Modellleistung beitragen.

Abschließend folgt in [Kapitel 6](#) eine Zusammenfassung der wichtigsten Erkenntnisse der Arbeit. Hier werden die Forschungsfragen beantwortet und ein Ausblick auf mögliche Weiterentwicklungen und zukünftige Forschungsarbeiten gegeben.

2 Theoretische Grundlagen

Das folgende Kapitel gibt eine Einführung in die theoretischen Grundlagen, die für das Verständnis der Arbeit notwendig sind. Dabei werden vor allem zentrale Konzepte und Methoden des maschinellen Lernens, der synthetischen Datengenerierung, sowie des Contrastive Learning behandelt. Darauffolgend wird der Anwendungsfall vorgestellt, den die Arbeit behandelt, und ein eigener Ansatz zur Integration der Methoden DA-Fusion und Supervised Contrastive Learning beschrieben.

2.1 Maschinelles Lernen

Maschinelles Lernen (ML) hat sich zu einem zentralen Bestandteil der Künstlichen Intelligenz (KI) entwickelt, einem interdisziplinären Forschungsfeld, das sich mit Algorithmen und Techniken befasst, welche es Computern ermöglichen, menschenähnliche Intelligenz zu erlangen.

Die ersten großen Durchbrüche in der KI kamen im Bezug auf Aufgaben, die für Menschen intellektuell eine große Herausforderung darstellten, die aber von Computern relativ einfach zu lösen waren, da sie als Liste formaler, mathematischer Regeln beschrieben werden konnten. Die große Schwierigkeit lag allerdings in den Aufgaben, die für Menschen relativ einfach und intuitiv sind, welche sich aber nur schwer formal beschreiben lassen (Goodfellow et al., 2016). Hierunter fallen z.B. die Spracherkennung, oder Objekterkennung.

Maschinelles Lernen bezeichnet einen Ansatz, bei dem Computer mit der Fähigkeit ausgestattet werden, selbstständig Wissen aus Erfahrung zu generieren, indem Muster und Konzepte aus rohen Daten erlernt werden. So kann ein Computerprogramm auf Basis von Beispielen lernen, wie es eine bestimmte Aufgabe lösen soll, ohne dass ihm explizit Regeln oder Algorithmen vorgegeben werden.

Eine allgemeine Definition für Maschinelles Lernen bietet (Mitchell, 1997):

Ein Computerprogramm soll aus Erfahrung E in Bezug auf eine Klasse von Aufgaben T und Leistungsmaß P lernen, wenn sich seine Leistung bei Aufgaben T , gemessen durch P , mit Erfahrung E verbessert.

Die Erfahrung E besteht dabei aus einer Menge von Trainingsdaten, beispielsweise Bilder. Die Aufgaben T können sehr unterschiedlich sein, von einfachen Klassifikations- und Regressionsaufgaben bis hin zu komplexen Problemen wie Spracherkennung oder autonomes Fahren. Das Leistungsmaß P gibt an, wie gut die Aufgaben T gelöst werden. Für Klassifikationsaufgaben wird häufig die Accuracy (Genauigkeit) verwendet, welche den Anteil der korrekt klassifizierten Datenpunkte angibt.

Durch das Lernen aus den Trainingsdaten ergibt sich ein *Modell* des zugrundeliegenden Problems, das dann auf neue, unbekannte Daten angewendet werden kann, um Vorhersagen zu machen oder Entscheidungen zu treffen.

2.1.1 Überwachtes und unüberwachtes Lernen

Wie genau Wissen aus Erfahrung bzw. aus Rohdaten generiert wird hängt vom gewählten Verfahren ab. Im Maschinellen Lernen gibt es dabei hauptsächlich zwei Paradigmen; das überwachte und das unüberwachte Lernen.

Beim überwachten Lernen (Supervised Learning) wird das Modell mit einem vollständig annotierten Datensatz trainiert. In den meisten Fällen bedeutet das, dass jeder Datenpunkt mit einem Klassenlabel versehen ist, sodass Eingabe-Ausgabe-Paare entstehen. Ziel ist es, eine Funktion zu lernen, welche die Eingaben auf die entsprechenden Ausgaben abbildet. Ein einfaches Beispiel wäre ein Bildklassifikator, der darauf trainiert wird, Katzen und Hunden zu unterscheiden. Hier würden alle Trainingsbilder entweder mit dem Label „Katze“ oder „Hund“ versehen sein. Im Training kann die Vorhersage des Modells dann mit dem tatsächlichen Label verglichen werden, um den Fehler zu berechnen und die Modellparameter entsprechend anzupassen.

Im Gegensatz dazu arbeitet unüberwachtes Lernen (Unsupervised Learning) mit unbeschrifteten Daten; es gibt also keine vorgegebenen Ausgaben. Stattdessen wird versucht, ein Modell zu befähigen, eigenständig Muster und Strukturen in den Daten zu erkennen und z.B. nützliche Repräsentationen der Eingangsdaten zu erlernen. Zu den häufigsten Methoden des unüberwachten Lernens gehören Clustering- und Assoziationsalgorithmen. Ein Beispiel ist die Segmentierung von Kunden in verschiedene Gruppen basierend auf ihrem Kaufverhalten (Shen, 2021).

In der Praxis werden oft auch hybride Ansätze genutzt, wie das semi-überwachte Lernen (Semi-Supervised Learning), bei dem eine Kombination aus beschrifteten und unbeschrifteten Daten verwendet wird, oder das selbstüberwachte Lernen (Self-Supervised Learning), bei dem das Modell eigenständig Teile der Daten zur Erzeugung von Überwachungssignalen verwendet, anstatt sich auf externe, von Menschen bereitgestellte Labels zu verlassen.

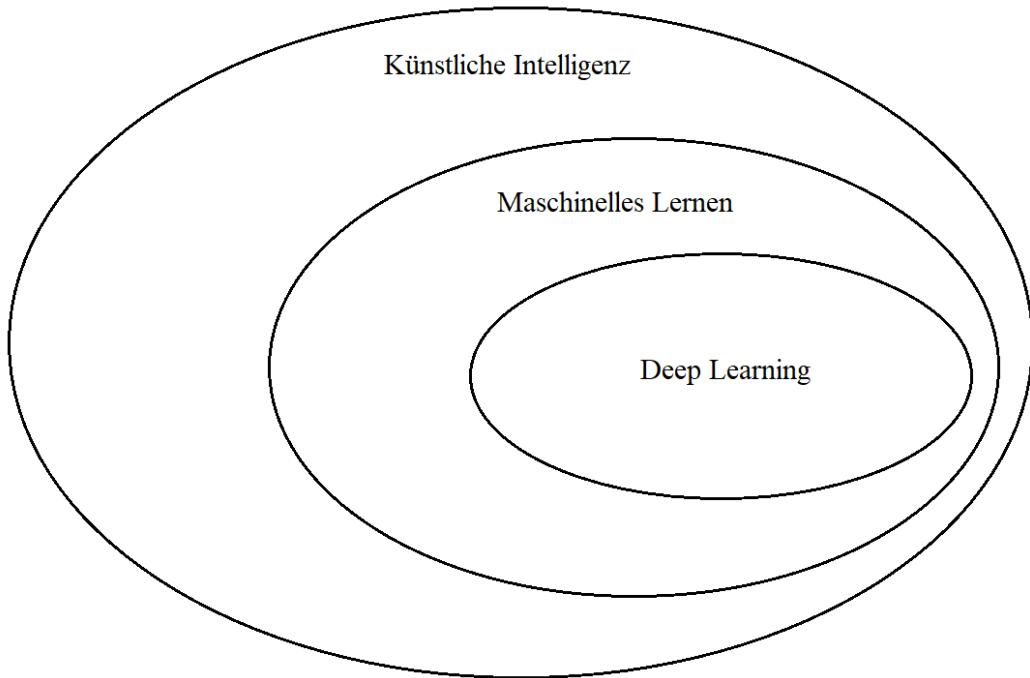


Abbildung 2.1: Ein einfaches Venn-Diagramm, das die Beziehung zwischen KI, ML und DL veranschaulicht.

2.1.2 Deep Learning

Das Wissen, das ein Modell aus den Trainingsdaten lernt, wird in Form von sogenannten Features repräsentiert. Diese Features können einfache Konzepte wie Kanten oder Farben sein, oder komplexere Konzepte wie Gesichter oder Objekte. Unter Deep Learning (DL) versteht man eine tiefe, hierarchische Vernetzung dieser Konzepte, sodass komplexere Konzepte auf simpleren Konzepten aufbauen können. Visuell veranschaulicht entsteht ein Graph mit vielen Ebenen (Goodfellow et al., 2016).

Deep Learning ist also eine Unterkategorie des Maschinellen Lernens (siehe Abbildung 2.1), bei der die Eingabedaten mehrere Verarbeitungsschichten durchlaufen, um eine hierarchische Repräsentation zu ermöglichen. Jede Schicht transformiert die Eingabedaten in eine etwas abstraktere Darstellung. Aus diesem Grund fällt Deep Learning auch unter den Begriff des Representation Learning (Zhou, 2021).

Heutzutage bilden Deep Learning-Modelle die Grundlage für viele Anwendungen der Künstlichen Intelligenz, darunter Bild- und Spracherkennung, maschinelle Übersetzung, medizinische Diagnose und autonomes Fahren. Die rasante Entwicklung in diesem Bereich ist vor allem auf die Verfügbarkeit großer Datenmengen und immer leistungsfähigere Hardware zurückzuführen (Goodfellow et al., 2016).

2.1.3 Neuronale Netze

Während die rasante Entwicklung von Deep Learning in den vergangenen Jahren besonders spürbar geworden ist, sind die zugrundeliegenden Algorithmen und Konzepte schon seit Jahrzehnten bekannt. Das Künstliche Neuronale Netz (KNN) bildet dabei die Grundlage der allermeisten Deep Learning-Modelle. Es ist inspiriert von der Struktur und Funktionsweise des menschlichen Gehirns und besteht aus einer Vielzahl von miteinander verbundenen Knoten (Neuronen), die in Schichten organisiert sind. Die Schichten teilen sich auf in eine Eingabeschicht (Input Layer), eine oder mehrere verdeckte Schichten (Hidden Layers) und eine Ausgabeschicht (Output Layer).

Die einzelnen Neuronen, auf dem diese Netze aufbauen, sind mathematische Modelle des biologischen Neurons. Sie wurden erstmals 1943 von Warren McCulloch und Walter Pitts vorgestellt, wobei sich die heute verwendeten Neuronenmodelle kaum unterscheiden (Zhou, 2021). In Abbildung 2.2 ist ein solches Neuronenmodell dargestellt.

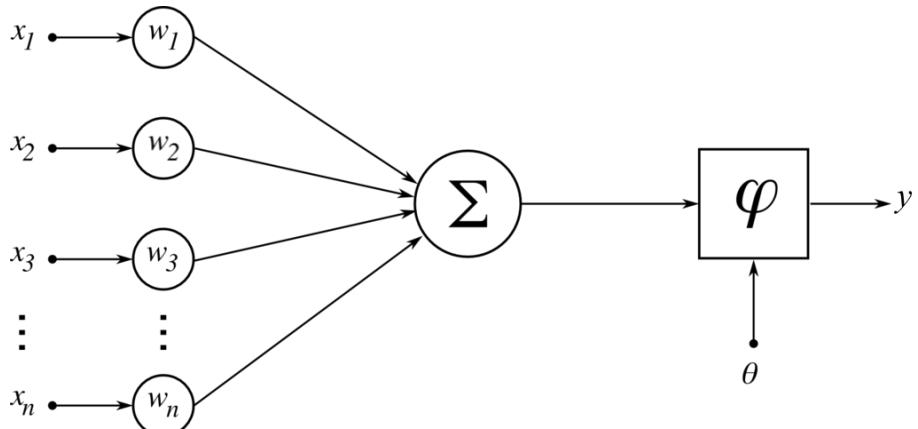


Abbildung 2.2: Schematische Abbildung eines künstlichen Neurons (Burgmer, 2005).

Jedes Neuron empfängt eine Reihe von Eingaben x_1, x_2, \dots, x_n , entweder von externen Quellen oder von den Ausgaben anderer Neuronen. Für jede dieser Eingaben gibt es zugehörige Gewichtungen (Weights) w_1, w_2, \dots, w_n , welche die Stärke und Richtung (positiv oder negativ) des Einflusses der jeweiligen Eingaben auf das Neuron bestimmen. Das Neuron berechnet dann die gewichtete Summe aller Eingaben. Zusätzlich wird ein Bias θ hinzugefügt, der den Schwellenwert bestimmt. Die Aktivierung y des Neurons kann dementsprechend wie in Gleichung 2.1 beschrieben werden.

$$y = \varphi \left(\sum_{i=1}^n w_i x_i + \theta \right), \quad (2.1)$$

wobei φ die Aktivierungsfunktion ist, die die Ausgabe des Neurons bestimmt. Häufig wird die Sigmoid-Funktion verwendet (siehe [Gleichung 2.2](#)), die, im Gegensatz zu einer einfachen Schwellenwertfunktion, kontinuierlich und differenzierbar ist. Dies vereinfacht die Optimierung des Netzwerks ([Zhou, 2021](#)).

$$\text{Sigmoid}(x) = \frac{1}{1 + e^{-x}} \quad (2.2)$$

Die Sigmoid-Funktion eignet sich gut für binäre Klassifikationsprobleme. Für Multiklassenprobleme wird jedoch die Softmax-Funktion verwendet, die eine Wahrscheinlichkeitsverteilung über alle Klassen berechnet (siehe [Gleichung 2.3](#)) ([Goodfellow et al., 2016](#)).

$$\text{Softmax}(x)_i = \frac{e^{x_i}}{\sum_{j=1}^n e^{x_j}}, \quad (2.3)$$

wobei x_i die Eingabe für Klasse i ist und n die Anzahl der Klassen.

Im Training fließen die Eingabedaten in einer Vorwärtssausbreitung (Forward Propagation) durch das Netzwerk, um die Ausgabe zu berechnen, welche sich aus der Aktivierung der Neuronen in der Ausgabeschicht ergibt. Es wird dann eine geeignete Verlustfunktion angewendet, um den Fehler (Loss) des Modells zu berechnen. Das Ziel des Trainings ist es, die Gewichtungen der Neuronen so anzupassen, dass der Fehler minimiert wird.

Die Wahl der Verlustfunktion hängt von der Art des Problems ab, das gelöst werden soll. Für Klassifikationsprobleme wird häufig die Kreuzentropie (Cross Entropy) verwendet, die den Fehler zwischen den tatsächlichen und den vorhergesagten Wahrscheinlichkeiten berechnet (siehe [Gleichung 2.4](#)).

$$Loss = - \sum_{c=1}^N y_c \log \hat{y}_c, \quad (2.4)$$

wobei N die Anzahl der Klassen ist, y_c die tatsächliche Wahrscheinlichkeit für Klasse c und \hat{y}_c die vorhergesagte Wahrscheinlichkeit für Klasse c .

Die Optimierung geschieht durch eine Rückwärtssausbreitung (Backpropagation), welche den berechneten Fehler rückwärts durch das Netz propagiert, um die Gewichte und Schwellenwerte um einen geringen Wert in die Richtung anzupassen, die den Fehler minimieren würde. Die Richtung wird bestimmt, indem der Gradient der Verlustfunktion berechnet wird. Die Lernrate (Learning Rate) bestimmt, wie groß die Schritte sind, die entlang des Gradienten gemacht werden. So bewegt sich das Modell iterativ entlang des Gradienten hin zu einem lokalen Minimum der Verlustfunktion. Dieser Algorithmus wird als Gradient Descent (Gradientenabstieg) bezeichnet. Im Maschinellen Lernen kommt hauptsächlich der Stochastic

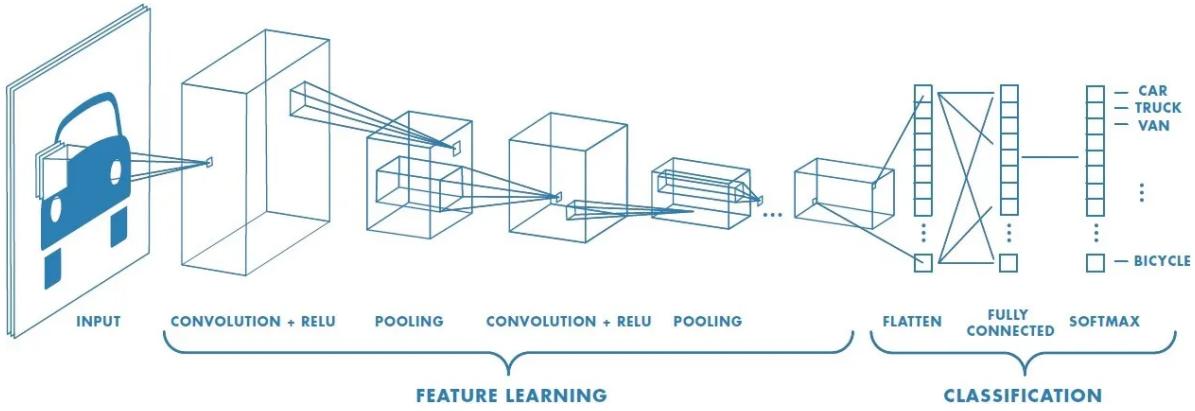


Abbildung 2.3: Architektur eines Convolutional Neural Networks (Saha, 2018).

Gradient Descent (SGD) zum Einsatz, der immer nur einen kleinen, zufällig ausgewählten Teil der Trainingsdaten (einen sogenannten Batch) verwendet, um den Gradienten zu berechnen.

Deep Learning mit neuronalen Netzen kann gut am Beispiel des Convolutional Neural Networks (CNN) veranschaulicht werden, welches speziell für die Verarbeitung von Bildern entwickelt wurde (siehe Abbildung 2.3). Ein CNN besteht aus mehreren Schichten, darunter Convolutional Layers, Pooling Layers und Fully Connected Layers. Die Convolutional Layers extrahieren sogenannte *Feature Maps* aus den Eingabebildern, indem sie Faltungskerne über das Bild schieben und die gewichteten Summen der Pixel berechnen. Die Pooling Layers reduzieren die Dimensionalität der Feature Maps, indem sie die Größe der Merkmale reduzieren. Die Fully Connected Layers kombinieren die extrahierten Merkmale, um die endgültige Klassifikation vorzunehmen.

Mit Backpropagation trainierte CNNs wurden erstmals in (Cun et al., 1989) behandelt, bilden aber auch heute noch die Grundlage von zahlreichen Deep Learning-Anwendungen und Methoden. Sie sind vor allem deshalb so erfolgreich in der Bildverarbeitung, weil sie die räumliche Struktur von Bildern berücksichtigen und dadurch effektiv Muster und Merkmale in den Daten erkennen können.

2.1.4 Datenaugmentation

Wenn ein Modell so stark an die Trainingsdaten angepasst wird, dass es nicht in der Lage ist, auf neuen, unbekannten Daten zu generalisieren, spricht man von Overfitting (Goodfellow et al., 2016). Statt die zugrundeliegenden Muster zu lernen, speichert das Modell auch zufällige Rauschsignale und Fehler. Dies führt dazu, dass das Modell auf den Trainingsdaten eine hohe Genauigkeit erreicht, aber auf neuen Daten eine schlechtere Leistung zeigt. Man verwendet daher neben den Trainingsdaten auch Validierungsdaten (Daten, die nicht zum Training des Modells verwendet werden) zur Überwachung des Overfittings.

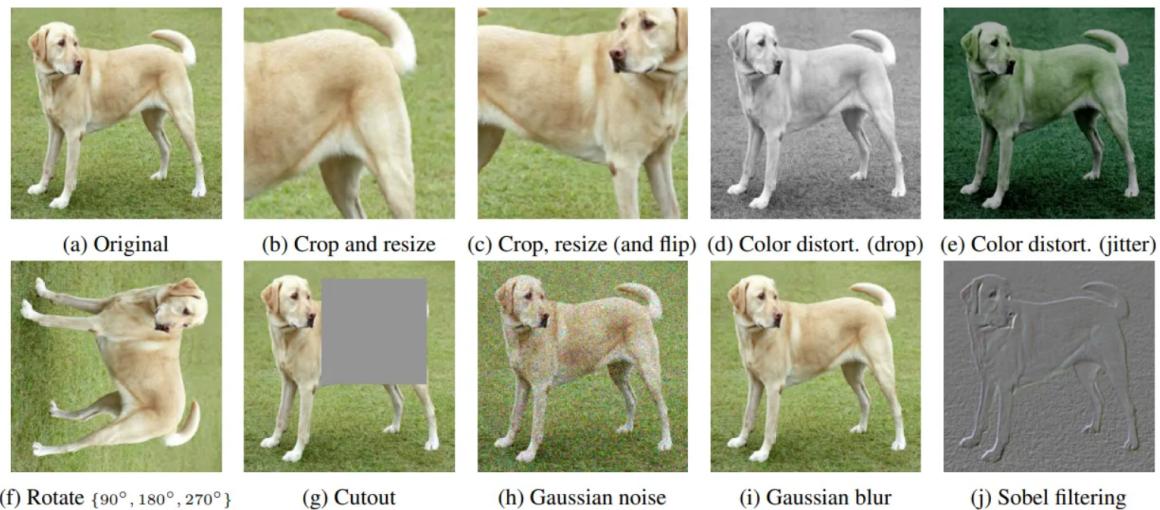


Abbildung 2.4: Einige Beispiele für unterschiedliche Datenaugmentationstechniken und Kombinationen, die in (Chen et al., 2020) verwendet wurden.

Während Overfitting auch das Resultat einer zu hohen Komplexität des Modells sein kann, die dafür sorgt, dass die Trainingsdaten zu genau modelliert werden, liegt das Problem oftmals in der Quantität und Qualität der Trainingsdaten selbst. Sind nicht genügend Daten vorhanden, oder sind die Daten nicht divers genug, kann das Modell nicht generalisieren.

Dabei kann das Modell auch ohne völlig neue Daten zu beschaffen robuster gegenüber Variationen gemacht werden, indem es mit leicht veränderten Versionen der Trainingsdaten trainiert wird. Bei der Datenaugmentation werden unterschiedliche Transformationen auf die vorhandenen Daten angewendet, z.B. Rotation, Skalierung, Verschiebung, Spiegelung, Helligkeitsanpassung oder Rauschen (Shorten & Khoshgoftaar, 2019). Die Transofrmationen werden meist zufällig parametrisiert, um eine Vielzahl von Variationen zu erzeugen. Das Ziel ist es, das Modell zu zwingen, die zugrundeliegenden Muster der Daten zu lernen, anstatt sich auf spezifische Merkmale zu verlassen, die nur in den Trainingsdaten vorhanden sind. Einige Beispiele für Datenaugmentationstechniken sind in Abbildung 2.4 dargestellt.

Datenaugmentation hat sich als eine der wichtigsten Ansätze zur Vermeidung von Overfitting und zur Verbesserung der Generalisierungsfähigkeit von Modellen erwiesen. Sie findet in fast allen Methoden des Maschinellen Lernens Anwendung.

2.1.5 Out-of-Distribution Daten

Wenn ein KI-Modell mit Daten konfrontiert wird, die außerhalb des Bereichs liegen, den es während des Trainings gesehen hat, spricht man von Out-of-Distribution (OOD)-Daten. Es handelt sich also um Datenpunkte oder Muster, die sich signifikant von den Trainingsdaten

unterscheiden. Dies kann zu Problemen führen, da das Modell möglicherweise nicht in der Lage ist, angemessene Vorhersagen oder Entscheidungen für diese Daten zu treffen. Die Erkennung von OOD-Daten ist daher ein wichtiges Forschungsgebiet im maschinellen Lernen, da sie dazu beitragen kann, die Zuverlässigkeit und Sicherheit von KI-Systemen zu verbessern.

Im Idealfall gibt ein Neuronales Netz höhere Softmax-Wahrscheinlichkeiten für In-Distribution (ID)-Daten und niedrigere Wahrscheinlichkeiten für OOD-Daten aus. Tatsächlich sind die Wahrscheinlichkeiten für OOD-Daten fast immer sehr hoch, meist reicht aber der Abstand zwischen ID- und OOD-Daten aus, um OOD-Instanzen frühzeitig zu identifizieren (Hendrycks & Gimpel, 2018).

Oftmals ist die OOD-Detektion aber weniger trivial, etwa wenn sich ID- und OOD-Daten sehr ähnlich sind. Es wird daher stets nach alternativen Ansätzen gesucht, um die OOD-Detektion zu verbessern. In (Hendrycks & Gimpel, 2018) wird dabei ein Klassifikator erweitert, um Rekonstruktionen der Eingabedaten zu erstellen und den Fehler zwischen den Original- und Rekonstruktionsdaten zu messen. Ein hoher Rekonstruktionsfehler kann auf OOD-Daten hinweisen. Auch das Temperature Scaling (Guo et al., 2017), welches die Softmax-Wahrscheinlichkeiten kalibriert und somit den Abstand zwischen ID- und OOD-Daten vergrößert, kann zusammen mit kleinen Störungen in den Eingabedaten für zuverlässigere OOD-Detektion sorgen (Liang et al., 2020).

2.2 Synthetische Daten

Datenaugmentation wurde bereits als eine Möglichkeit vorgestellt, um die Menge und Vielfalt der Trainingsdaten zu erhöhen und damit die Generalisierungsfähigkeit des Modells zu verbessern. Allerdings kann die bloße Augmentation an ihre Grenzen stoßen, wenn die verfügbaren Trainingsdaten selbst nicht genügend Vielfalt aufweisen. In solchen Fällen können synthetische Daten eine nützliche Ergänzung sein. Anstatt einfache Transformationen auf die Eingangsdaten anzuwenden, werden völlig neue Datenpunkte generiert, die die zugrundeliegenden Muster der realen Daten nachahmen. Auch im Hinblick auf den Datenschutz sind synthetische Daten vielversprechend, da sie die Möglichkeit bieten, sensible und personenbezogene Informationen zu schützen.

Während synthetische Daten auch manuell mit Hilfe von Simulationssoftware erstellt werden können, hat insbesondere die Entwicklung generativer KI-Modelle in den letzten Jahren zu einer neuen Ära der synthetischen Datenerzeugung geführt. Diese Modelle sind in der Lage, komplexe Datenstrukturen zu lernen und realistische Daten zu generieren, die von echten Daten kaum zu unterscheiden sind.

Da sich diese Arbeit mit der Bildklassifikation beschäftigt, wird im Folgenden vor allem auf generative Modelle eingegangen, die darauf spezialisiert sind, Bilder zu generieren. Es sollen einige der wichtigsten Modelle vorgestellt werden, um eine Grundlage für den aktuellen Stand der synthetischen Datengenerierung zu schaffen.

2.2.1 Variational Autoencoder

Ein Autoencoder ist eine spezielle Art von KI-Modell, das entwickelt wurde, um Daten effizient zu komprimieren und anschließend zu rekonstruieren. Es wurde im Wesentlichen in (Hinton & Salakhutdinov, 2006) vorgestellt, die Grundideen gehen jedoch bis in die 1980er Jahre zurück, wie z.B. in (Rumelhart et al., 1986), wo auch das Backpropagation-Verfahren zur Optimierung neuronaler Netze beschrieben wurde.

Autoencoder bestehen aus zwei Hauptkomponenten; einem *Encoder*, der die Eingabedaten in einen niedrigdimensionalen Darstellungsvektor komprimiert, und einem *Decoder*, der den Darstellungsvektor wieder in die ursprünglichen Daten umwandelt. So wird das Originalbild als Punkt in einem mehrdimensionalen latenten Raum abgebildet, in dem jede Dimension eine bestimmte Eigenschaft des Bildes kodiert.

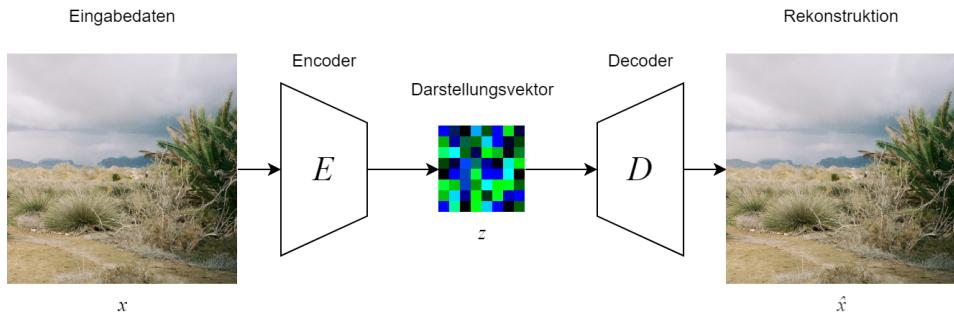


Abbildung 2.5: Aufbau eines Autoencoders.

Das Training eines Autoencoders erfolgt durch Minimierung des Rekonstruktionsfehlers, der die Differenz zwischen den Eingabedaten und den Rekonstruktionen beschreibt. Eine gängige Verlustfunktion hierfür ist der Mean Squared Error (MSE), der in Gleichung 2.5 formuliert ist.

$$Loss = \frac{1}{n} \sum_{i=1}^n (x_i - \hat{x}_i)^2, \quad (2.5)$$

wobei x_i die Eingabedaten und \hat{x}_i die Rekonstruktionen sind.

Autoencoder können theoretisch zur Generierung synthetischer Daten verwendet werden, indem ein beliebiger Punkt im latenten Raum gewählt und durch den Decoder rekonstruiert wird. Allerdings hat der herkömmliche Autoencoder einige Schwächen in Bezug auf diese Aufgabe (Foster, 2020): Da er immer nur einen festen Punkt im latenten Raum für jede Eingabe lernt, entsteht ein diskreter und unstrukturierter latenter Raum. Es ist weder die Kontinuierlichkeit zwischen den Punkten der Eingabedaten noch die Vollständigkeit des latenten Raums sichergestellt, sodass neue Stichproben aus dem latenten Raum nicht unbedingt realistische Daten erzeugen.

Der *Variational Autoencoder* (VAE) (Kingma & Welling, 2013) adressiert die Schwachstellen des Autoencoders und verwendet probabilistische Methoden, um die Datenverteilung im latenten Raum zu modellieren; anstelle eines einzelnen, festen Punktes im latenten Raum wird für jede Eingabe eine Verteilung gelernt, aus der die latenten Variablen stammen. Die Verteilung entspricht dabei einer einfachen, bekannten Verteilung, wie z.B. einer Normalverteilung $\mathcal{N}(0, 1)$.

Um die Verteilungen zu lernen, berechnet der Encoder einerseits den Mittelwert μ und die Standardabweichung σ , und erzeugt außerdem eine zufällige Stichprobe der Verteilung. Der Decoder nimmt diese Stichprobe und rekonstruiert die Eingabedaten. Beim Training wird nicht nur der Rekonstruktionsfehler minimiert, sondern auch die sogenannte Kullback-Leibler-Divergenz (KL-Divergenz), welche die Ähnlichkeit zwischen zwei Wahrscheinlichkeitsverteilungen Q und P misst (siehe Gleichung 2.6), in diesem Fall zwischen der latenten Verteilung und einer Normalverteilung (siehe Gleichung 2.7) (Foster, 2020).

$$D_{KL}(Q||P) = \int q(x) \log \frac{q(x)}{p(x)} dx \quad (2.6)$$

$$D_{KL}(\mathcal{N}(\mu, \sigma^2) \parallel \mathcal{N}(0, 1)) = \frac{1}{2} \sum (1 + \log(\sigma^2) - \mu^2 - \sigma^2) \quad (2.7)$$

Durch die Kombination von Rekonstruktionsfehler und KL-Divergenz ergibt sich nun eine Annäherung an die Log-Likelihood $\log p(x)$ der Daten, also der Wahrscheinlichkeit, dass das Modell die Eingabedaten korrekt rekonstruiert. Diese Annäherung wird als Evidence Lower Bound (ELBO) bezeichnet. Sie ist deshalb notwendig, weil zur direkten Berechnung der Log-Likelihood die Integration über alle möglichen Werte der latenten Variablen z erforderlich wäre, was in der Praxis nicht möglich ist.

Die finale Verlustfunktion, die sowohl den Rekonstruktionsfehler als auch die KL-Divergenz kombiniert, ergibt sich aus der negativen ELBO (siehe Gleichung 2.8). Diese Verlustfunktion wird minimiert, um sowohl die Qualität der Rekonstruktionen zu maximieren als auch die latente Verteilung näher an die Normalverteilung anzunähern.

$$Loss = -\mathbb{E}q(z|x)[\log p(x|z)] + D_{KL}(q(z|x)\|p(z)), \quad (2.8)$$

wobei $p(z|x)$ die latente Verteilung eines gegebenen Datenpunktes x , $p(x|z)$ die Rekonstruktion des Datenpunktes x aus der latenten Darstellung z und $p(z)$ die Prior-Verteilung der latenten Variablen ist.

2.2.2 Generative Adversarial Networks

Ein weiteres berühmtes KI-Modell ist das Generative Adversarial Network (GAN), das in (Goodfellow et al., 2014) vorgestellt wurde. GANs bestehen aus einem *Generator*- und einem *Diskriminator*-Modell, die in einem adversariellen Prozess gegeneinander antreten.

Der Generator nimmt Zufallsrauschen als Eingabe und erzeugt daraus Daten, die möglichst realistisch wirken sollen. Der Diskriminatator erhält sowohl echte Daten aus dem Trainingsdatensatz als auch die vom Generator erzeugten Daten. Seine Aufgabe ist es, zwischen echten und künstlichen Daten zu unterscheiden. Dafür gibt er für einen gegebenen Datenpunkt die Wahrscheinlichkeit aus, dass dieser echt ist. [Abbildung 2.6](#) gibt einen Überblick über das ganze System.

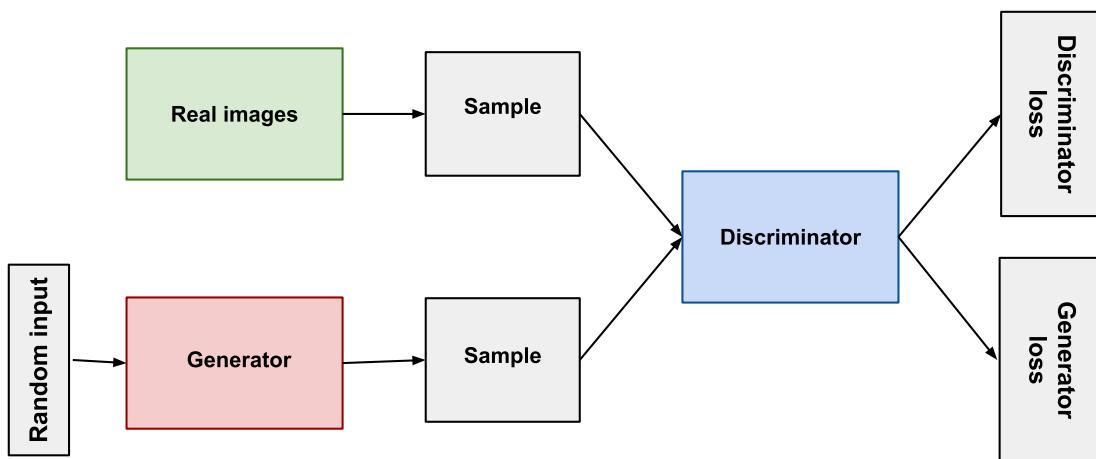


Abbildung 2.6: Überblick über das GAN-Training (Google for Developers, 2022).

Der Diskriminatorverlust ergibt sich aus der Differenz zwischen der Wahrscheinlichkeit, dass echte Daten als echt und künstliche Daten als falsch klassifiziert werden. Der Generatorverlust ergibt sich aus der Differenz zwischen der Wahrscheinlichkeit, dass künstliche Daten als echt klassifiziert werden und der tatsächlichen Wahrscheinlichkeit, dass sie echt sind. Die Verlustfunktionen für den Generator und den Diskriminatoren sind in [Gleichung 2.9](#) formuliert.

$$\begin{aligned} Loss_D &= -\mathbb{E}_{x \sim P}[\log D(x)] - \mathbb{E}_{z \sim P(z)}[\log(1 - D(G(z)))] \\ Loss_G &= -\mathbb{E}_{z \sim P(z)}[\log D(G(z))] \end{aligned} \quad (2.9)$$

Besonders in der Bildgenerierung haben GANs eine hohe Qualität erreicht. GANs sind auch flexibel in Bezug auf die Eingangsdaten und können mit verschiedenen Datentypen wie Bildern, Texten oder Audiodaten arbeiten. Dennoch gibt es einige Nachteile. Beispielsweise kann das Training eines GANs sehr instabil sein, da es schwierig ist, ein Gleichgewicht zwischen Generator und Diskriminator zu finden. Es kann auch zum sogenannten Modus-Kollaps kommen, bei dem der Generator nur eine begrenzte Anzahl von Beispielen erzeugt, da der Diskriminator diese als besonders realistisch bewertet (Foster, 2020). Der Generator lernt dann, nur diese Beispiele zu reproduzieren, anstatt die gesamte Datenverteilung zu lernen. Auch die Wahl geeigneter Hyperparameter ist eine Herausforderung.

2.2.3 Diffusionsmodelle

In den letzten Jahren haben Diffusionsmodelle zu einem enormen Fortschritt in der Bildgenerierung, insbesondere der Text-to-Image (T2I)-Generierung, geführt. Diese Modelle basieren auf dem Konzept der Diffusion, das den Prozess der langsamen Vermischung von Teilchen oder Informationen über die Zeit beschreibt. Im maschinellen Lernen fand das Konzept erstmals in (Sohl-Dickstein et al., 2015) Anwendung, mit der Idee, die Struktur von Daten durch Hinzufügen von Rauschen schrittweise aufzulösen und anschließend ein Modell darauf zu trainieren, das ursprüngliche Bild zu rekonstruieren (siehe Abbildung 2.7). Seitdem haben sich Diffusionsmodelle als eine neue Klasse von generativen Deep Learning-Modellen etabliert, die in der Lage sind, noch realistischere Bilder zu generieren als GANs.

Das Training von Diffusionsmodelle teilt sich in zwei Phasen auf, die Vorwärts- und die Rückwärtsdiffusion, welche beide als Markov-Ketten modelliert werden können, also stochastische Prozesse, bei denen die zukünftige Entwicklung eines Systems nur von seinem aktuellen Zustand abhängt. Im Bezug auf Diffusionsmodelle repräsentiert jeder Schritt in der Markov-Kette einen Zeitschritt t und die Zustände $x^{(t)}$ sind die Bilder zu diesem Zeitschritt.

In der Vorwärtsdiffusion fügt das Modell einem Bild schrittweise Rauschen hinzu, sodass es sich am Ende des Prozesses wie eine Normalverteilung verhält. Die Wahrscheinlichkeitsdichte des verrauschten Bildes wird durch die Produktregel der bedingten Wahrscheinlichkeiten berechnet (siehe Gleichung 2.10).

$$q(x^{(0..T)}) = q(x^{(0)}) \prod_{t=1}^T q(x^{(t)}|x^{(t-1)}) \quad (2.10)$$

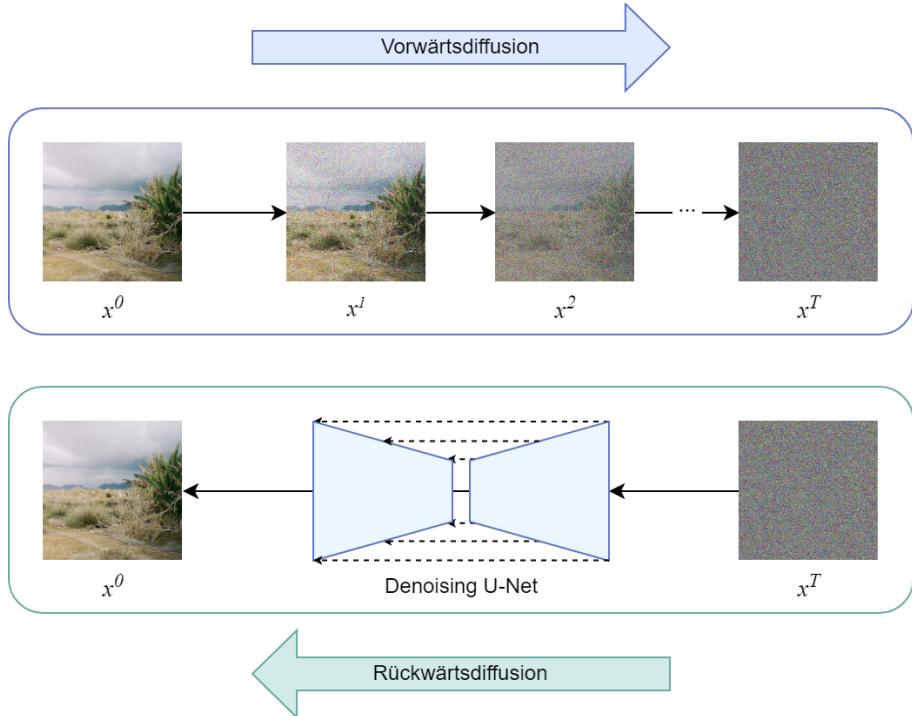


Abbildung 2.7: Darstellung der Vorwärts- und Rückwärtsdiffusion in einem Diffusionsmodell.

In der Rückwärtsdiffusion wird das Modell darauf trainiert, das verrauschte Bild schrittweise in das ursprüngliche Bild zurückzuwandeln. Die Wahrscheinlichkeitsdichte des ursprünglichen Bildes wird ebenfalls durch die Produktregel der bedingten Wahrscheinlichkeiten berechnet (siehe [Gleichung 2.11](#)).

$$p(x^{(0\dots T)}) = p(x^{(T)}) \prod_{t=1}^T p(x^{(t-1)}|x^{(t)}) \quad (2.11)$$

Das Neuronale Netz, das zur Vorhersage der Zustände $x^{(t-1)}$ aus den verrauschten Zuständen $x^{(t)}$ verwendet wird, ist typischerweise ein Denoising U-Net (siehe [Abbildung 2.7](#)). Es hat eine ähnliche Struktur wie ein Autoencoder, wobei Skip Connections zwischen den Encoder- und Decoder-Schichten bestehen, um die Rekonstruktion des ursprünglichen Bildes zu verbessern. Die Hauptaufgabe des U-Nets besteht darin, das hinzugefügte Rauschen zu schätzen, welches anschließend subtrahiert wird, um das ursprüngliche Bild zu rekonstruieren.

Die Verlustfunktion wird oft als MSE formuliert, um den Unterschied zwischen dem generierten Rauschen und dem tatsächlichen Rauschen zu minimieren.

Anders als bei GANs gibt es keine direkten adversarialen Optimierungsmechanismen, die zu einem Ungleichgewicht führen können. Es wird stattdessen explizit die Wahrscheinlich-

keitsdichte zwischen den realen Daten und den erzeugten Daten minimiert, was zu einem robusteren und stabileren Trainingsprozess führt.

Eine entscheidende Weiterentwicklung der Diffusionsmodelle war die Text-Konditionierung des generativen Prozesses. In (Ramesh et al., 2022) wurde DALL-E 2 vorgestellt, welches ein Diffusionsmodell verwendet, das in der Lage ist, Bilder aus Textbeschreibungen zu generieren. DALL-E 2 verwendet das Transformer-Modell (Vaswani et al., 2017), um die Textbeschreibungen in eine latente Repräsentation zu kodieren, die dann im Diffusionsprozess verwendet wird.

Stable Diffusion ist ein besonders einflussreiches Diffusionsmodell, das sich durch die Verwendung eines latenten Diffusionsprozesses auszeichnet (Rombach et al., 2022). Anstatt den Diffusionsprozess direkt auf den hochdimensionalen Bildraum anzuwenden, führt Stable Diffusion die Diffusion in einem niedrigdimensionalen latenten Raum durch, der durch ein VAE-ähnliches Netzwerk generiert wird. Diese Technik reduziert die Rechenkosten und beschleunigt den Generierungsprozess erheblich, während gleichzeitig hochauflösende und realistische Bilder erzeugt werden.

2.2.4 DA-Fusion

DA-Fusion (Trabucco et al., 2023) ordnet sich zwischen den Bereichen der Datenaugmentation und der synthetischen Datengenerierung ein. Es handelt sich um eine Methode zur semantischen Datenaugmentation von Bildern unter Verwendung eines vortrainierten Stable Diffusion-Modells.

Der traditionelle Ansatz der Datenaugmentation, wie in [Unterabschnitt 2.1.4](#) beschrieben, hat sich als effektiv erwiesen, um die Generalisierungsfähigkeit von Modellen zu verbessern. Allerdings erfordert dieser Ansatz auch eine gute Intuition in Bezug auf den verwendeten Datensatz, um zu vermeiden, dass Transformationen gewählt werden, durch die Informationen verloren gehen, die für die Aufgabe des zu trainierenden Modells wichtig sind. Wenn beispielsweise Farbinformationen für die Klassifizierung von Blumen wichtig sind, könnte die Datenaugmentation durch zufällige Farbänderungen die Leistung des Modells verschlechtern. Ein weiteres Beispiel sind Objekte, die klein im Bild sind und durch zufällige Ausschnitte des Bildes aus der Sicht des Modells verschwinden können. DA-Fusion hingegen nutzt das Wissen eines vortrainierten Diffusionsmodells, um den Bildinhalt semantisch zu verstehen und automatisch neue, realistische Variationen zu generieren.

Dafür wird zunächst die Methode Textual Inversion aus (Gal et al., 2022) angewendet, um ein vortrainiertes Stable Diffusion-Modell auf den gegebenen Datensatz feinabzustimmen (Fine-tuning). Für jedes Konzept bzw. für jede Klasse wird ein neues Text-Embedding $\langle y \rangle$ als Platzhalter in das Modell integriert, das unter Verwendung von Trainings-Prompts wie

"a photo of a <y>" und den zugehörigen Bilddaten trainiert wird. Entscheidend ist hier, dass nicht das ganze Diffusionsmodell neu trainiert wird, sondern lediglich neue Wörter erlernt werden, welche die spezifischen Konzepte repräsentieren, sodass sich bei der Bildgenerierung weiterhin auf das vortrainierte semantische Wissen des Modells gestützt werden kann.

Anschließend können die Bilder augmentiert werden, indem ihnen eine geringe Menge an Rauschen hinzugefügt wird (also ein Prozess der Vorwärtsdiffusion), welches dann durch das feinabgestimmte Modell wieder entfernt werden soll (die Rückwärtsdiffusion, die von dem Modell gelernt wurde). Hier kommen die selben Text-Prompts zum Einsatz, welche die Text-Embeddings der Konzepte beinhalten. Auf diese Weise müssen keine völlig neuen Bilder generiert werden, denn die grundlegende Struktur wird durch die ursprünglichen Bilder vorgegeben. Dieser Prozess bietet die Möglichkeit, den Grad der Augmentation durch die Wahl des Zeitschritts t zu steuern, an dem das Bild in die Rückwärtsdiffusion eingefügt wird und wie stark es dafür vorher verändert werden muss. So entstehen stärkere oder subtilere Augmentationen.

Der beschriebene Image-to-Image-Generierungsprozess baut dabei weitgehend auf dem in (Meng et al., 2022) beschriebenen Ansatz *SDEdit* auf. DA-Fusion unterscheidet sich jedoch dadurch, dass es neue Text-Embeddings für die Konzepte feinabstimmmt und somit speziell für die semantische Datenaugmentation zuvor unbekannter Konzepte konzipiert ist.

2.3 Contrastive Learning

Ein zentrales Thema dieser Arbeit ist das Contrastive Learning. Es ordnet sich in den Bereich des Representation Learning ein, bei dem es darum geht, nützliche Repräsentationen von Daten zu lernen. Die Besonderheit des Contrastive Learning ist, dass es auf der Kontrastierung von Daten basiert, um ähnliche Beispiele zu gruppieren und unähnliche Beispiele voneinander zu trennen. Dies hat sich als effektive Methode mit erstaunlicher Generalisierungsfähigkeit und Robustheit gegenüber Adversarial Attacks erwiesen (Liu, 2021).

Das Contrastive Learning stammt ursprünglich aus dem unüberwachten Lernen, wird aber oft als selbstüberwachte Methode bezeichnet, da die Kontrastierung der Daten eine Art Selbstüberwachung darstellt. Die erlernten Repräsentationen haben sich als deutlich genauer als im traditionellen überwachten Lernen herausgestellt, wo diese ausschließlich zur Klassenunterscheidung optimiert werden (Keshtmand et al., 2022). Da die Methode nicht auf annotierte Daten angewiesen ist, kann sie ein sehr effizienter Ansatz sein, um Modelle vorzutrainieren, die sich durch Finetuning an spezifische Aufgaben anpassen, während sie gleichzeitig allgemeinere, von der Aufgabe unbeeinflusste Merkmale lernen (Radford et al., 2021).

Aufgrund des Erfolgs im unüberwachten Lernen gibt es vermehrt Ansätze, Contrastive Learning auch im überwachten Setting anzuwenden. Hier wird nicht nur zwischen einzelnen Instanzen unterschieden, sondern auch die Klassenzugehörigkeit der Beispiele berücksichtigt.

In den folgenden Abschnitten wird genauer auf die Funktionsweise von sowohl unüberwachten als auch überwachten Varianten des Contrastive Learning eingegangen, die in den letzten Jahren vielversprechende Ergebnisse erzielt haben.

2.3.1 Unsupervised Contrastive Learning

Im Contrastive Learning soll die Distanz ähnlicher Beispiele in einem latenten Repräsentationsraum minimiert und die Distanz unähnlicher Beispiele maximiert werden. Dabei zieht das Modell verschiedene Ankerbeispiele heran und kontrastiert sie mit positiv- und negativ-Beispielen.

Es kommen verschiedene Verlustfunktionen zum Einsatz, wobei der Fehler grundsätzlich darüber aussagt, ob ähnliche Beispiele auch im latenten Raum nahe beieinander liegen. Verlustfunktionen unterscheiden sich zum Beispiel in der Wahl der Distanzmetrik, oder der Anzahl der positiven und negativen Beispiele, die für den Vergleich herangezogen werden.

Das wohl prominenteste Beispiel für Contrastive Learning in der visuellen Domäne ist *SimCLR*, das in (Chen et al., 2020) vorgestellt wurde. SimCLR verwendet den sogenannten NT-Xent Loss (Normalized Temperature-scaled Cross-Entropy Loss), der die Ähnlichkeiten zwischen allen Paaren im Batch berücksichtigt, anstatt nur einzelne Triplets oder Paare. Jedes Beispiel wird dabei zweimal augmentiert, um ein positives Paar aus zwei Ansichten des Beispiels zu erhalten. Alle anderen Beispiele (bzw. dessen Ansichten) im Batch werden als negativ-Beispiele gesehen.

Die Eingabedaten werden durch ein CNN in eine latente Repräsentation transformiert. SimCLR verwendet anschließend einen sogenannten Projection Head, der die encodierten Repräsentationen weiter transformiert, um einen Repräsentationsraum zu lernen, der für die Unterscheidung der Beispiele geeignet ist. In diesem Repräsentationsraum werden die Ähnlichkeitswerte der Paare berechnet, um den Fehler zu bestimmen. Dafür wird die Kosinus-Ähnlichkeit $s_{i,j}$ der Repräsentationen z_i und z_j eines Paares berechnet (siehe [Gleichung 2.12](#)).

$$s_{i,j} = \frac{z_i \cdot z_j}{\|z_i\| \cdot \|z_j\|} \quad (2.12)$$

Der NT-Xent Loss ergibt sich dann aus der Berechnung des Softmax über die Ähnlichkeiten aller Paare in einem Batch mit N Beispielen, skaliert mit einem Temperaturparameter τ ,

um die Unterscheidung zwischen positiven und negativen Paaren hervorzuheben (siehe [Gleichung 2.13](#)).

$$Loss = - \sum_{i=1}^N \log \frac{\exp(s_{i,i}/\tau)}{\sum_{j=1}^{2N} \exp(s_{i,j}/\tau)}, \quad (2.13)$$

wobei $s_{i,i}$ das positive Paar aus den zwei augmentierten Ansichten repräsentiert und $s_{i,j}$ die negativen Paare.

Die Verwendung aggressiver Data augmentation zur Erzeugung der zwei Ansichten in ([Chen et al., 2020](#)) wurde als entscheidender Faktor identifiziert, um die Robustheit der Repräsentationen zu verbessern. Dabei soll die gemeinsame Information der zwei Ansichten so weit wie möglich minimiert werden, ohne Task-relevante Informationen zu verlieren ([Tian et al., 2020](#)). ([Chen et al., 2020](#)) stellt auch den positiven Effekt von *Hard Negatives* heraus, also von Paaren, welche unterschiedliche Konzepte darstellen, aber sehr ähnlich aussehen. Größere Batch Sizes und längere Trainingszeiten begünstigen ebenfalls die Lernfähigkeit des Modells.

2.3.2 Supervised Contrastive Learning

Auch im überwachten Kontext findet Contrastive Learning zunehmend Anwendung, da es das Potenzial hat, die Leistungsfähigkeit von Modellen durch die Nutzung von Label-Informationen zu steigern.

Mit *SupCon* ([Khosla et al., 2021](#)) wird eine Weiterentwicklung der Verlustfunktion aus SimCLR geboten, die das Contrastive Learning auf das überwachte Setting anpasst und mehrere positiv-Beispiele pro Ankerbeispiel berücksichtigt. Dabei werden neben den zwei augmentierten Ansichten des Ankerbeispiels weitere Beispiele der gleichen Klasse als positiv-Beispiele herangezogen. Das soll auch die Notwendigkeit des „Hard Negative-Minings“ reduzieren. Trotzdem wird gezeigt, dass das Training sowohl von Hard Negatives wie auch von Hard Positives profitiert.

Die Verlustfunktion von SupCon aus allen N Beispielen im Batch kann wie in [Gleichung 2.14](#) formuliert werden.

$$Loss = - \sum_{i=1}^N \log \left(\frac{1}{|\mathcal{P}(i)|} \sum_{p \in \mathcal{P}(i)} \frac{\exp(s_{i,p}/\tau)}{\sum_{j=1}^N \exp(s_{i,j}/\tau)} \right), \quad (2.14)$$

wobei $P(i)$ die positiv-Beispiele für das Ankerbeispiel i sind. Die Verwendung des Mittelwertes der positiven Repräsentationen stabilisiert das Training und führt zu einer verbesserten Leistung (Khosla et al., 2021).

Das Einbeziehen der Klassenzugehörigkeiten verbessert nicht nur die Generalisierungsfähigkeit der Modelle, sondern wirkt sich auch positiv auf die OOD-Detektion aus, insbesondere bei *Near OOD*-Daten, die sich nur geringfügig von den Trainingsdaten unterscheiden (Keshtmand et al., 2022).

Im überwachten Kontext bieten sich auch neue Möglichkeiten zur Weiterentwicklung des Contrastive Learning. So wird beim *SCL with Hard Negatives* (Jiang et al., 2024) eine zusätzliche Einschränkung in der Kontrastierung der Beispiele vorgenommen, um ausschließlich Hard Negatives als negativ-Beispiele für ein gegebenes Ankerbeispiel zu selektieren. Dies wird erreicht, indem die negativ-Beispiele im Repräsentationsraum nahe am Ankerbeispiel liegen müssen. Die Strategie hat sich als effektiv erwiesen, um die Generalisierungsfähigkeit der Modelle zu verbessern.

2.4 Klassifikation von Gebrauchsgegenständen in der Recyclingwirtschaft

Der konkrete Anwendungsfall, auf den sich diese Arbeit bezieht, ist die Klassifikation verschiedener industrieller Objekte und Gebrauchsgegenstände in der Recyclingwirtschaft. Grundlage dafür ist das Forschungsprojekt “Sensorische Erfassung, automatisierte Identifikation und Bewertung von Altteilen anhand von Produktdaten sowie Informationen über bisherige Lieferungen” (EIBA), das im Rahmen der Fördermaßnahme „Ressourceneffiziente Kreislaufwirtschaft – Innovative Produktkreisläufe (ReziProK)“ des Bundesministeriums für Bildung und Forschung entstand (Wagner, 2022). Neben dem Fraunhofer-IPK waren auch die Technische Universität Berlin, die Deutsche Akademie der Technikwissenschaften (acatech), sowie die Circular Economy Solutions GmbH beteiligt.

Das Ziel des Projekts war die Entwicklung eines KI-gestützten Systems, welches Daten aus verschiedenen digitalen Sensoren, sowie Kontextdaten aus dem Geschäftsprozess verarbeitet, um bei der Identifikation, Inspektion und Sortierung von Altteilen zu unterstützen.

Viele der Altteile sind sehr ähnlich und erfordern Expertenwissen, um sie korrekt zu identifizieren und den Beschädigungsgrad zu bewerten, wobei die Beschriftungen und Barcodes oft nicht mehr lesbar sind. Hier setzt das KI-System an, um den Menschen zu unterstützen. Es werden dafür multimodale Sensordaten verwendet, unter anderem Tiefenkameras (RGB-D) und eine Waage zur Erfassung des Gewichts zum Einsatz. In Abbildung 2.8 ist die Mensch-Maschine-Schnittstelle einer mobilen Teststation des Systems dargestellt.

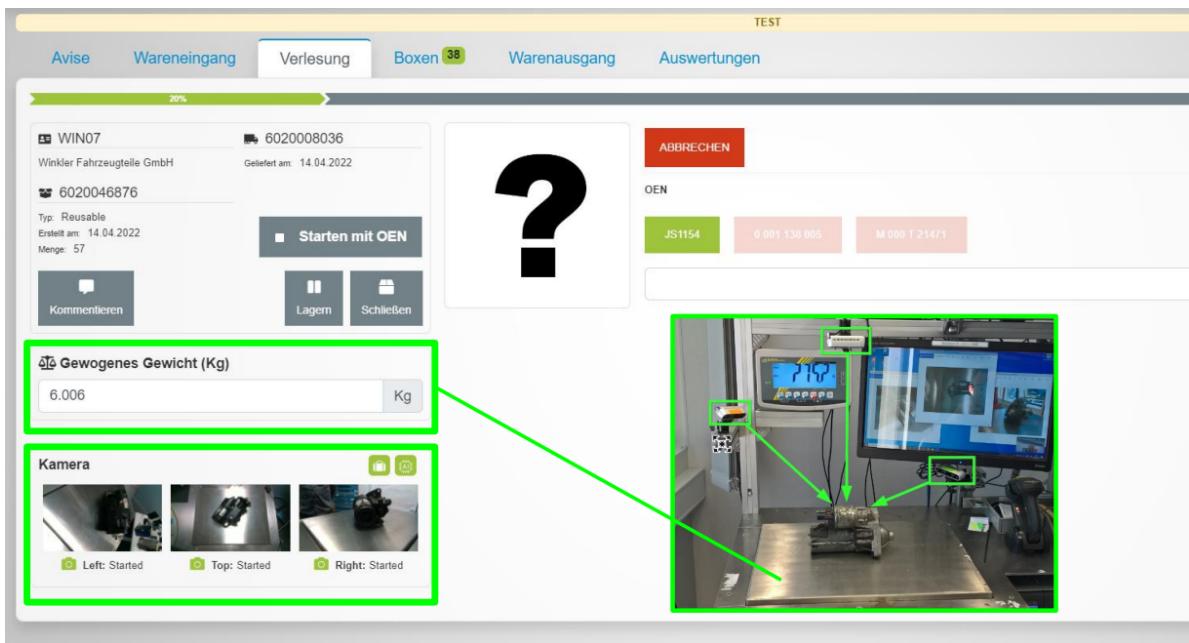


Abbildung 2.8: Die Mensch-Maschine-Schnittstelle des Systems (Wagner, 2022).

Das System konnte am Beispiel von gebrauchten Fahrzeugteilen bereits eine gute Accuracy erreichen. Die Beschaffung einer ausreichenden Menge an Beispielbildern—sowohl für das Training als auch für die Validierung—ist jedoch äußerst schwierig. Dadurch werden die Objektklassen nicht mit ausreichend Variation repräsentiert, was die Generalisierungsfähigkeit der Modelle beeinträchtigen könnte.

2.4.1 Herausforderungen bei der Generierung synthetischer Daten

Um die Datenvielfalt zu erhöhen, bietet sich die Generierung von synthetischen Daten an. In [Abschnitt 2.2](#) wurden bereits verschiedene Methoden zur Bildgenerierung vorgestellt, die sich für diesen Zweck eignen. Dabei ergeben sich jedoch eine Reihe von Herausforderungen im Kontext des vorgestellten Anwendungsfalls, auf die zunächst aufmerksam gemacht werden sollte. Der Anwendungsfall zeichnet sich durch folgende Eigenschaften aus, die die Generierung synthetischer Daten erschweren:

1. Die Objekte weisen teilweise eine sehr hohe Komplexität auf, etwa bei Motoren oder Generatoren mit vielen Details.
2. Es gibt oft nur feine Unterschiede zwischen den Klassen, die es zu berücksichtigen gilt.
3. Auf Grund des Multiview-Setups haben die Bilder nur wenig Variation, vor allem in den Hintergründen. Auch die Objekte selbst sind zwar aus verschiedenen Perspektiven aufgenommen, bieten aber pro Klasse nicht viel Variation.

Dies führt bei der Verwendung herkömmlicher Methoden zur synthetischen Datengenerierung zu Problemen. Einerseits scheitern viele dieser Methoden bei der Generierung komplexer Konzepte (Lu et al., 2024), was aufgrund der feinen Unterschiede zwischen den Klassen besonders problematisch ist. Auch die Verwendung von „Off-the-shelf“-Modellen zur T2I-Generierung sind deshalb oft nicht in der Lage spezifische Konzepte eines Anwendungsfalls akkurat wiederzugeben (Fan et al., 2023). Wenn synthetische Daten mit mangelhafter Qualität auf naive Weise in das Training integriert werden, kann sich dann die Leistung der Modelle verschlechtern (van Breugel et al., 2023).

Es ergeben sich also hohe Anforderungen. Einerseits muss die Genauigkeit der generierten Daten gewährleistet sein, um die Klassifikation der Modelle nicht zu beeinträchtigen. Andererseits muss genügend Variation ermöglicht werden, um die Generalisierungsfähigkeit der Modelle zu verbessern.

DA-Fusion soll deshalb in dieser Arbeit als Methode zur Generierung synthetischer Daten untersucht werden. Sie bietet womöglich eine Lösung für die genannten Probleme, indem die Bilder nicht von Grund auf neu generiert werden, sondern die Struktur der echten Daten beibehalten. Dadurch könnte die Genauigkeit der generierten Daten erhöht und gleichzeitig die Variation der Daten verbessert werden.

2.4.2 Synthetische Daten als negativ-Beispiele im Contrastive Learning

Neben der Generierung von synthetischen Daten stellt sich die Frage, wie diese effektiv in das Training von Modellen integriert werden können.

Mit Blick auf die Herausforderungen bei der Generierung synthetischer Daten, zusammen mit den Besonderheiten des Contrastive Learning, ergibt sich eine interessante Forschungslücke: Ist es möglich, auch aus *mangelhaften* synthetischen Daten zu lernen, wenn sie ausschließlich als negativ-Beispiele im Contrastive Learning verwendet werden? Lässt sich so die Leistung eines Modells bei der Klassifikation von echten Daten verbessern und gleichzeitig die Robustheit gegenüber OOD-Daten erhöhen?

Bisherige Ansätze im Contrastive Learning, wie etwa (Khosla et al., 2021), haben sich weitgehend auf reale Daten als negative Beispiele verlassen. Die Idee, synthetische Daten gezielt als negativ-Beispiele zu nutzen, stellt einen neuen Ansatz dar. Dies könnte besonders dann interessant sein, wenn die generierten OOD-Daten visuell ähnlich zu den ID-Daten sind. Solche Near OOD-Daten könnten für die ID-Daten eine ähnliche Funktion wie Hard Negatives erfüllen und gleichzeitig die Fehlklassifikation von OOD-Daten reduzieren.

Die bisherigen Erfolge von Contrastive Learning, insbesondere von SimCLR, zeigen, dass das Modell von Hard Negatives besonders stark profitiert (Chen et al., 2020). Auch (Jiang et al., 2024) baut auf dieser Erkenntnis auf und führt eine Strategie zum Hard Negative Sampling im Supervised Contrastive Learning ein—jedoch ohne Verwendung von synthetischen Daten.

Es gilt also zu untersuchen, ob ein Modell durch den Einsatz von synthetischen negativen Beispielen, die mittels Diffusionsmodellen oder ähnlichen Techniken generiert wurden, seine Fähigkeit zur Generalisierung verbessern kann. Insbesondere stellt sich die Frage, ob diese synthetischen Daten die Robustheit gegenüber echten OOD-Daten erhöhen können.

2.4.3 Integration von DA-Fusion und Supervised Contrastive Learning

Im Rahmen dieser Arbeit wird ein Ansatz vorgestellt, bei dem synthetische Augmentationen, die mit DA-Fusion generiert wurden, gezielt in das Supervised Contrastive Learning eingebunden werden. Die Grundidee besteht darin, sowohl ID-Daten (In-Distribution) als auch Near OOD-Daten (Near Out-of-Distribution) durch eine kontrollierte Anpassung der Augmentationsstärke zu erzeugen. Die Near OOD-Daten werden dabei im Contrastive Learning als Hard Negatives verwendet, um die Repräsentationen des Modells zu verbessern.

Wie bereits bezüglich der Herausforderungen in der synthetischen Datengenerierung erwähnt, stellt DA-Fusion im genannten Anwendungsfall eine vielversprechende Methode dar. Durch die Erhöhung des Parameters für die Augmentationsstärke können zunehmend komplexe und variierende Beispiele generiert werden. Für die Generierung der ID-Daten werden nur moderate Augmentationen angewendet. Demgegenüber werden die Near OOD-Daten durch eine stärkere Augmentation erzeugt, wobei noch ausreichend Ähnlichkeit zu den ID-Daten bestehen bleibt, um als Hard Negative zu gelten.

Im Kontext des Supervised Contrastive Learning wird eine eigene Sampling-Strategie für Near OOD-Daten entwickelt. Diese Daten werden nicht wahllos als negativ-Beispiele verwendet, sondern gezielt aus Augmentationen derselben Klasse des Ankerbeispiels generiert. Dies stellt sicher, dass die negativen Beispiele in den gleichen semantischen Raum fallen, aber durch die Augmentationen genug variieren, um als OOD-Daten zu gelten. Die Idee ist, dass diese Near OOD-Daten dem Modell helfen, feinere Unterschiede innerhalb und außerhalb der Klasse zu lernen, wodurch die Generalisierungsfähigkeit gegenüber realen Daten verbessert wird.

Die Formulierung der Verlustfunktion aus dem Supervised Contrastive Learning (siehe Gleichung 2.14) wird entsprechend der Verwendung von Near OOD-Daten leicht angepasst (siehe Gleichung 2.15).

$$Loss = - \sum_{i=1}^N \log \left(\frac{1}{|\mathcal{P}(i)|} \sum_{p \in \mathcal{P}(i)} \frac{\exp(s_{i,p}/\tau)}{\sum_{j \in \mathcal{J}(i)} \exp(s_{i,j}/\tau)} \right), \quad (2.15)$$

wobei $\mathcal{J}(i)$ die negativ-Beispiele für das Ankerbeispiel i sind (ID und OOD), welche gefiltert wurden, um von den OOD-Augmentationen nur solche zu verwenden, die aus der Klasse des Ankerbeispiels generiert wurden und somit als Near OOD gelten.

3 Methodisches Vorgehen

In diesem Kapitel wird das methodische Vorgehen der Arbeit beschrieben. Als Basis für die Untersuchung der Forschungsfragen wird zunächst der MVIP-Datensatz vorgestellt, auf dem die Experimente durchgeführt werden. Anschließend wird die Implementierung der Modelle DA-Fusion und Supervised Contrastive Learning erläutert, sowie die Herangehensweise zur Generierung synthetischer Daten mit DA-Fusion und die Trainings- und Testdurchläufe mit Supervised Contrastive Learning definiert. Zuletzt werden die Evaluationsmethoden und Metriken vorgestellt, die zur Auswertung der Experimente verwendet werden.

3.1 MVIP-Datensatz

Grundlage der Forschungsarbeit ist der im Rahmen des EIBA-Projekts entstandene MVIP-Datensatz (Koch et al., 2023), wobei MVIP für *Multi-View Industrial Parts* steht. Er enthält 308 Klassen, welche wiederum in 18 verschiedene Oberklassen (Super Classes) eingeteilt sind. Insgesamt gibt es etwa 71.276 Sets an Bildern, die jeweils RGB- und Tiefendaten, sowie Segmentierungsmasken enthalten. Die Bilddaten stammen aus Intel RealSense D435 und D415 Tiefenkameras, die die Objekte gleichzeitig aus verschiedenen Perspektiven aufnehmen.

Darüber hinaus gibt es auch Metadaten, etwa zum Gewicht des Objekts, oder Beschreibungen in natürlicher Sprache durch verschiedene Stichwörter (z.B. „Used“, „Rusty“, usw.).

3.1.1 Teildatensatz

Da relativ viele Trainings- und Testdurchläufe vorgesehen waren, war es notwendig, einen Teildatensatz auszuwählen, um die Rechenzeit zu reduzieren.

Konkret wurden 20 Klassen aus der Oberklasse „CarComponent“ ausgewählt, da diese Objekte wie Motoren und Generatoren enthält, welche von den in [Unterabschnitt 2.4.1](#) beschriebenen Herausforderungen besonders betroffen sind. Insgesamt enthält der Teildatensatz 2390 Trainingsbilder und 1000 Validierungsbilder. Darüber hinaus kamen die Segmentierungsmasken in der Vorverarbeitung zum Einsatz, die im nachfolgenden Abschnitt behandelt wird.



Abbildung 3.1: Beispielbilder aus dem MVIP-Datensatz, die auf die Region of Interest (ROI) zugeschnitten wurden (Koch et al., 2023).

3.1.2 Vorverarbeitung

Die Vorverarbeitung der Bilder unterscheidet sich nur leicht zwischen dem Fine-tuning der Text-Embeddings für DA-Fusion und den Trainings- und Testdurchläufen mit Supervised Contrastive Learning. In beiden Fällen wurden die Bilder auf die Region of Interest (ROI) zugeschnitten (siehe Abbildung 3.1), um mehr Fokus auf die Objekte zu legen und den Hintergrund zu minimieren. Für den MVIP-Datensatz war dies besonders wichtig, da alle Objekte in einer kaum variierenden Laborumgebung aufgenommen wurden, und weil einige der Objekte einen kleineren Anteil des Bildes einnehmen als andere.

Für den ROI-Crop wurden die Segmentierungsmasken verwendet, um die Bounding Box der Objekte zu bestimmen und die Bilder entsprechend zuzuschneiden. Die Ausgabe ist ein quadratisches Bild, das die Objekte in der Mitte enthält. Die Implementierung wird in Abschnitt 3.2 genauer erläutert.

Zusätzlich wurden verschiedene „klassische“ Augmentationen angewendet, um die Daten zu erweitern und die Modelle robuster zu machen. Dazu gehören z.B. Rotation, ColorJitter und Normalisierung. Die Parameter für die Augmentationen wurden eher konservativ gewählt, um die feinen Unterschiede zwischen den Klassen nicht zu verwischen.

3.2 Implementierung

Zur Vorbereitung der Experimente mussten zunächst die Implementierungen der Modelle DA-Fusion und Supervised Contrastive Learning aufgesetzt werden. Dabei stützt sich diese Arbeit im Wesentlichen auf die Implementierungen von DA-Fusion aus (Trabucco, 2024) und Supervised Contrastive Learning aus (Tian, 2023). Die grundlegende Funktionsweise

dieser Implementierungen sowie die Anpassungen, die im Rahmen des untersuchten Ansatzes vorgenommen wurden, sollen nun näher erläutert werden.

3.2.1 DA-Fusion

Die Implementierung von DA-Fusion kann weitgehend unverändert angewendet werden, um synthetische Daten für den MVIP-Teildatensatz zu generieren. Allerdings musste dieser zunächst als eigene Klasse mit den Methoden `get_image_by_idx(idx)` und `get_label_by_idx(idx)` unter `semantic_aug/datasets/mvip.py` implementiert werden. Insbesondere musste die Auswahl der 20 „CarComponent“ Klassen sichergestellt werden, wofür die JSON-Dateien der Objekt-Metadaten ausgelesen werden (siehe Anhang 1).

Um das Fine-tuning eines vortrainierten Stable Diffusion-Modells mittels Textual Inversion durchzuführen, muss das Skript `fine_tune_upstream.py` mit den entsprechenden Argumenten zur Wahl des Datensatzes und der Hyperparameter aufgerufen werden. Das Argument `examples_per_class` bestimmt dabei die Anzahl der verwendeten Bilder pro Klasse.

Während des Trainings werden die Bilder zusammen mit Text-Prompts, welche das zu trainierende Text-Embedding enthalten, zur Textual Inversion verwendet. Die Text-Prompts wurden zuvor in einer Liste definiert. Für die Experimente wurde die Liste leicht angepasst, vor allem um weniger angemessene Formulierungen wie "a rendering of a {}" und "a photo of a cool {}" zu entfernen.

Es werden außerdem mit der Funktion `log_validation` Validierungsbilder generiert, welche den Prompt "a photo of a {}" verwenden, der als Argument übergeben wird. Diese Bilder werden von Grund auf neu generiert und sind deshalb nicht mit den späteren Augmentationen vergleichbar, bieten aber eine Möglichkeit, den Verlauf des Trainings zu überwachen.

Die gelernten Text-Embeddings werden in .bin-Dateien gespeichert und anschließend mit dem Skript `aggregate_embeddings.py` in ein klassenagnostisches Template zusammengengeführt, das im nachfolgenden Schritt verwendet wird.

Zur Generierung der Augmentationen wird schließlich das Skript `generate_augmentations.py` ausgeführt. Ähnlich wie bei der Textual Inversion wird den Bildern Rauschen hinzugefügt, bevor sie unter Konditionierung auf den gelernten Text-Embeddings rekonstruiert werden, wobei hier eine Stable Diffusion Image-to-Image-Pipeline verwendet wird. Der genaue Prompt kann als Argument übergeben werden, wobei auch hier "a photo of a {}" als Standardwert verwendet wird. Um mehr Variation in der Augmentation zu erhalten, wurden hier unterschiedliche Prompts definiert, wie "a photo of a rusty {}" und "a photo of a dirty {}". Mit dem Argument `num_synthetic` kann die Anzahl der generierten Bilder pro Klasse festgelegt werden. Das `strength`-Argument entscheidet, wie viel Rauschen hinzugefügt wird und an

welchen Zeitschritt t das Bild integriert werden soll, wodurch mehr oder weniger stark veränderte Bilder entstehen. Schließlich kann mit dem `guidance_scale`-Argument die Stärke der Konditionierung auf die Text-Embeddings festgelegt werden.

3.2.2 Supervised Contrastive Learning

Die Implementierung von Supervised Contrastive Learning verwendet zwei Trainings-Skripte; eins für das Contrastive Pre-Training der latenten Repräsentationen und eins für die lineare Klassifikation der Repräsentationen.

Zunächst wurde die Klasse für den MVIP-Teildatensatz, die für die Implementierung von DA-Fusion verwendet wurde, größtenteils übernommen und in die Trainingsskripte integriert. Sie musste nun allerdings um einige Parameter und Funktionen erweitert werden, um die Verwendung der synthetischen Augmentationen aus DA-Fusion zu konfigurieren. So steuert der Parameter `aug_mode`, ob keine Augmentationen, ausschließlich ID-Augmentationen oder auch die Near OOD-Augmentationen verwendet werden sollen. Je nachdem, welche Einstellung gewählt wird, werden die entsprechenden Augmentationen direkt in die Trainingsdaten gemischt.

Entscheidend ist, dass die Klassenlabels der Near OOD-Augmentation auf den negativen Wert der ursprünglichen Klasse (vor Augmentation) gesetzt werden, sodass diese Augmentationen später als Hard Negatives identifiziert werden können. Außerdem wurde mit den Parametern `aug_ex_id` und `aug_ex_ood` eine Einstellung für die Anzahl der verwendeten ID- und OOD-Augmentationen pro Klasse implementiert. Über `aug_dir_id` und `aug_dir_ood` werden die Pfade zu den entsprechenden Augmentationen angegeben. Die Funktion zum Laden der Augmentationen ist in Anhang 2 aufgeführt.

Das Pre-Training der latenten Repräsentationen erfolgt mit dem Skript `main_supcon.py`. Das Modell wird mit einem ResNet-Backbone und einem Projection Head initialisiert. Im Training generiert es die latenten Repräsentationen der Bilder, die dann zusammen mit den Labels in die kontrastive Verlustfunktion gegeben werden, welche im Skript `losses.py` implementiert ist (siehe Anhang 3). Diese berechnet zunächst eine kontrastive Maske, die die positiven und negativen Paare definiert. Anschließend werden die Kosinus-Ähnlichkeiten der Paare berechnet und die durchschnittliche negative Log-Wahrscheinlichkeit der positiven Paare als Verlust zurückgegeben.

Für die Verwendung von Near OOD-Daten als negativ-Beispiele war die Anpassung dieser Verlustfunktion entscheidend: Zunächst wird die kontrastive Maske so gefiltert, dass keine positiven Paare mit OOD-Daten entstehen, indem diese durch ihre negativen Klassenlabels identifiziert werden. Nachdem dann die Ähnlichkeitswerte berechnet wurden, werden diese nochmals maskiert, sodass aus den negativen Paaren mit OOD-Daten nur solche übrig bleiben,

bei denen die OOD-Augmentation aus der selben Klasse generiert wurde, wie das ID-Bild. Die Verlustfunktion wird dann nur auf diese Paare angewendet.

Das trainierte Modell, welches im Verlauf des Pre-Trainings gespeichert wird, kann anschließend mit dem Skript `main_linear.py` für die lineare Klassifikation verwendet werden. Dabei wird das Modell mit einem neuen Head initialisiert, der die latenten Repräsentationen in die Klassenlabels überführt. Das Training erfolgt dann mit einer klassischen Cross Entropy-Verlustfunktion, die die Repräsentationen auf die Klassen abbildet.

Als Metriken wurden ursprünglich nur der Loss und die Accuracy ausgegeben, welche jeweils über eine Epoche gemittelt werden. Zusätzlich wurden im Rahmen dieser Arbeit die ID- und OOD-Confidence als Metriken implementiert (siehe [Unterabschnitt 3.3.3](#)). Dazu wird eine zusätzliche Validierungsepoke durchlaufen, welche ausschließlich die OOD-Augmentationen verwendet. Die Softmax-Wahrscheinlichkeiten der Vorhersagen werden dann getrennt für die ID- und OOD-Daten gemittelt und ausgegeben.

3.3 Versuchsaufbau

Die Untersuchung der Forschungsfragen erfolgte in zwei Schritten: Zunächst wurden synthetische Daten mit DA-Fusion generiert. Anschließend wurden die synthetischen Daten in den Trainings- und Testdurchläufen mit Supervised Contrastive Learning verwendet, um die Auswirkungen der Augmentationen auf die Klassifikation zu evaluieren. Die folgenden Abschnitte beschreiben den Versuchsaufbau für die Datengenerierung und die Trainings- und Testdurchläufe.

3.3.1 Synthetische Datengenerierung

Es wurden mit DA-Fusion zwei unterschiedliche Sätze von Augmentationen generiert: die In-Distribution-Augmentationen und die Near Out-of-Distribution-Augmentationen.

Für beide Sätze wurde als vortrainiertes Modell `CompVis/stable-diffusion-v1-4`¹ ausgewählt. Es wurden über `examples_per_class` 32 pro Klasse Beispiele ausgewählt. Die Klassen wurden alle mit dem Token `<motor>` initialisiert. Dabei wurden über das Argument `num_vectors` 16 Textual Inversion Vektoren pro Text-Embedding verwendet. Das Finetuning erfolgte mit einer Bildgröße von 512x512 Pixeln, einer Batch Size von 16, einer Lernrate von 0.0005 und einer konstanten Lernrate mit Warmup über 150 Schritte. Es wurde Mixed Precision mit FP16 verwendet und das Training wurde nach 1000 Schritten beendet. Die erlernten Text-Embeddings wurden für beide Sätze verwendet, um die Augmentationen zu generieren.

¹<https://huggingface.co/CompVis/stable-diffusion-v1-4>

Die Generierung erfolgte mit einer Guidance Scale von 15. Es wurden 16 Beispiele pro Klasse zur Augmentation herangezogen und für jedes Beispiel vier synthetische Bilder generiert. Die Segmentierungsmasken wurden verwendet, um die Augmentation nur auf den Bereich des Objekts zu begrenzen. Die ID- und Near OOD-Augmentationen unterscheiden sich nur durch den strength-Parameter, wobei für die ID-Augmentationen ein Wert von **0.2** und für die Near OOD-Augmentationen ein Wert von **0.5** verwendet wurde.

3.3.2 Trainings- und Testdurchläufe

Die Trainings- und Testdurchläufe mit Supervised Contrastive Learning sind in drei Versuchsreihen unterteilt. In jedem Versuch wird zunächst das Pre-Training der latenten Repräsentationen durchgeführt, gefolgt von der linearen Klassifikation der Repräsentationen. Die Versuchsreihen unterscheiden sich in der Verwendung der synthetischen Augmentationen:

- *Versuch 1*: Nur reale Daten werden verwendet, sowohl für das Pre-Training als auch für das Training des Klassifikators.
- *Versuch 2*: Neben den realen Daten werden auch ID-Augmentationen verwendet, sowohl für das Pre-Training als auch für das Training des Klassifikators.
- *Versuch 3*: Wie Versuch 2, aber im Pre-Training werden zusätzlich Near OOD-Augmentationen verwendet, wie in [Unterabschnitt 2.4.3](#) beschrieben.

Für das Pre-Training wurde eine Batch Size von 16, eine Lernrate von 0.001 mit Kosinus-Annealing und einer Dauer von 110 Epochen verwendet. Die Bilddaten wurden im Rahmen der herkömmlichen Augmentation, welche die zwei Ansichten eines Beispiels erzeugt, auf eine Größe von 224x224 Pixeln zugeschnitten und normalisiert.

Bei der linearen Klassifikation sind die Hyperparameter identisch, jedoch mit einer Dauer von nur 25 Epochen. Nach jeder Trainingsepoche werden die Modelle für eine Validierungsepoke auf den (echten) Testdaten evaluiert.

3.3.3 Evaluationsmethoden und Metriken

Die drei Versuchsreihen werden ausgewertet, indem folgende Metriken beim Test der linearen Klassifikation auf den Testdaten gemessen werden:

- *Top-1 Accuracy*: Der Anteil der korrekt klassifizierten Bilder. Top-1 bedeutet, dass das Modell die Klasse mit der höchsten Wahrscheinlichkeit als Vorhersage ausgibt.

- *ID-Confidence*: Die durchschnittliche Konfidenz des Klassifikators auf den ID-Daten. Die Konfidenz ist die Wahrscheinlichkeit, die der Klassifikator für die Vorhersage der korrekten Klasse ausgibt.
- *OOD-Confidence*: Die durchschnittliche Konfidenz des Klassifikators auf den OOD-Daten.

Darüber hinaus kann die Qualität der synthetischen Daten, die mit DA-Fusion generiert wurden, durch einfache visuelle Inspektion evaluiert werden, um Rückschlüsse auf die Messergebnisse zu ziehen. Bei den ID-Augmentationen wird darauf geachtet, dass die Objekte in den Bildern in ihrer Form möglichst unverändert bleiben, während die Farben und Texturen variieren können und sollen. Bei den OOD-Augmentationen wird darauf geachtet, dass die Objekte stark genug verändert werden, um nicht mehr als Beispiele für die ID-Klassen erkannt zu werden, während sie gleichzeitig noch genug Ähnlichkeit aufweisen, um herausfordernde negative Beispiele zu sein.

4 Ergebnisse

Dieses Kapitel präsentiert die Ergebnisse der Arbeit. Es wird auf die generierten synthetischen Daten eingegangen und die Trainings- und Testergebnisse der Modelle beschrieben. Anschließend wird die Klassifikations-Performance der Modelle verglichen und die Out-of-Distribution-Detektion analysiert.

4.1 Die generierten synthetischen Daten

Im Folgenden werden sowohl die In-Distribution- als auch die Near Out-of-Distribution-Augmentationen betrachtet. Es wird auf die Qualität der Daten eingegangen und Beispiele für gelungene und weniger gelungene Augmentationen gezeigt.

4.1.1 In-Distribution

Einige Beispiele der synthetischen In-Distribution-Daten sind in [Abbildung 4.1](#) zu sehen. Die Bilder sind überwiegend überzeugend und realistisch. Die Unterschiede sind auf den ersten Blick sehr subtil, vor allem, wenn die Bilder Seite an Seite betrachtet werden.

Dennoch sind Veränderungen in der Textur der Oberflächen erkennbar. Während selten auffällige Änderungen in der Beschaffenheit der Objekte generiert wurden, sind unterschiedliche Verteilungen von Rost und Verschmutzung zu erkennen.

In einigen Fällen sind die Ergebnisse jedoch weniger überzeugend. In [Abbildung 4.2](#) sind Beispiele für mangelhafte In-Distribution-Daten zu sehen. Die synthetischen Objekte sind in diesen Fällen nicht realistisch und weisen deutliche Artefakte auf. Oftmals hängt die Qualität der Daten mit Größe des Objektes im Originalbild zusammen (was auch die Auflösung des ROI-Crops entscheidet).

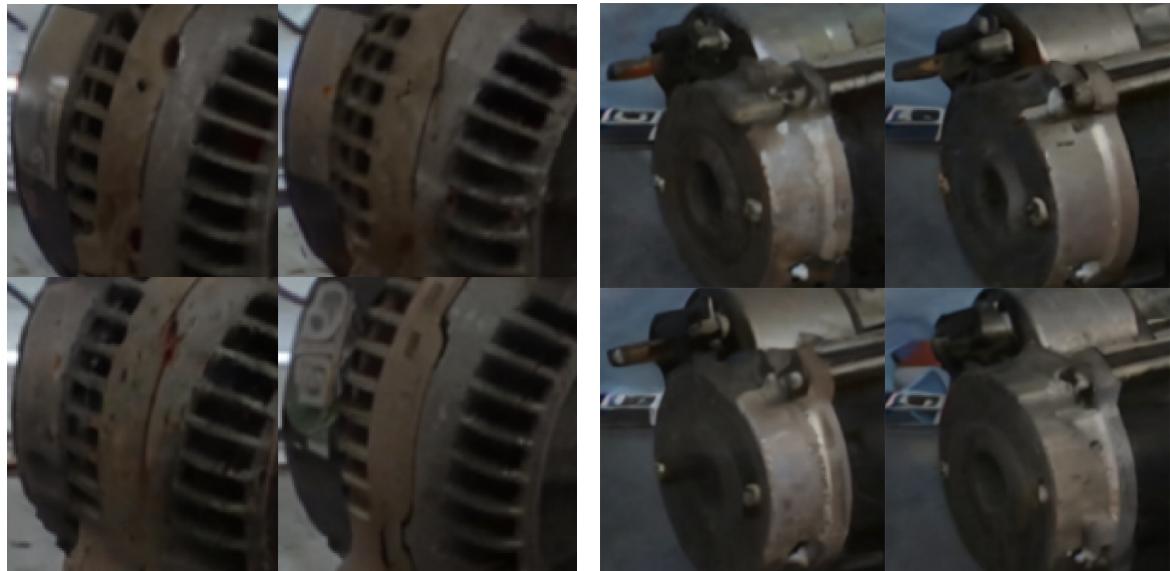


Abbildung 4.1: Vergrößerte Ausschnitte von einigen der In-Distribution-Augmentationen.

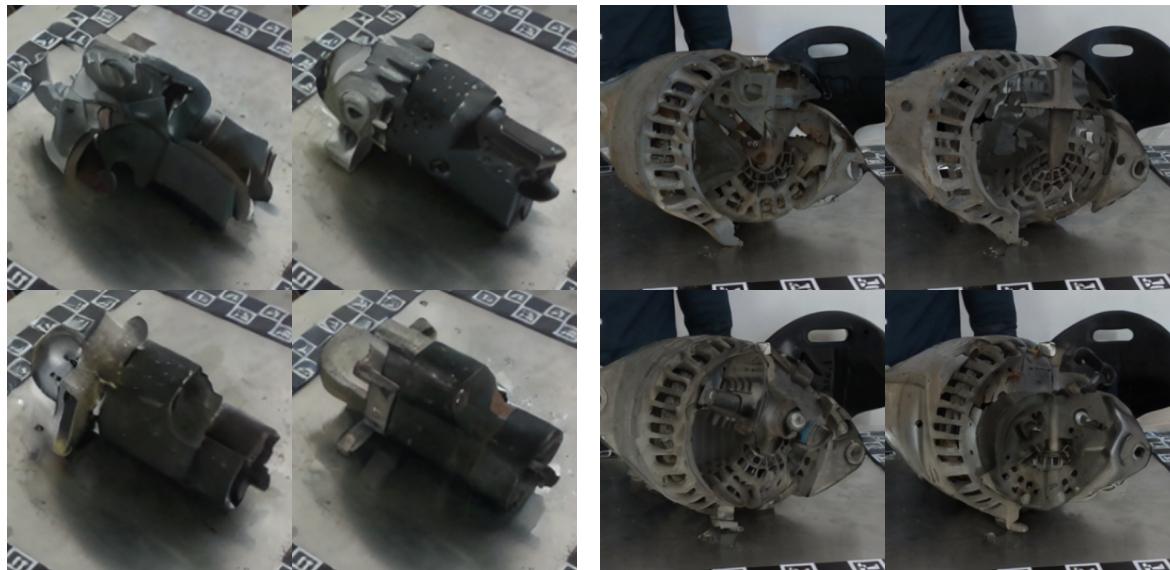


Abbildung 4.2: Beispiele für mangelhafte In-Distribution-Augmentationen.



Abbildung 4.3: Beispiele der Near Out-of-Distribution-Augmentationen.

4.1.2 Near Out-of-Distribution

Beispiele für die synthetischen Near Out-of-Distribution-Daten sind in [Abbildung 4.3](#) dargestellt. Die meisten Objekte wirken etwas weniger realistisch, aber dennoch plausibel. Vor allem ist hier der Einfluss der unterschiedlichen Text-Prompts zur Augmentation deutlicher erkennbar, wodurch die Objekte in unterschiedlichen Zuständen dargestellt werden.

Allerdings gibt es hier deutlich mehr Beispiele, die weniger überzeugend sind (siehe [Abbildung 4.4](#)). Die synthetischen Objekte weisen in diesen Fällen deutliche Artefakte auf oder sind vollständig vom ursprünglichen Konzept entkoppelt.



Abbildung 4.4: Beispiele für mangelhafte Out-of-Distribution-Augmentationen.

4.2 Trainings- und Testergebnisse

Es werden nun die Ergebnisse der Trainings- und Testdurchläufe im Supervised Contrastive Learning beschrieben. Es wird auf die Trainingskurven und die Performance der Modelle eingegangen.

4.2.1 Contrastive Pre-Training

[Abbildung 4.5](#) zeigt die Trainingskurven des Contrastive Pre-Trainings. Die Trainings- und Validierungsfehler zeigen insgesamt eine gute Konvergenz, wobei zu erkennen ist, dass der Validierungsfehler im Vergleich zum Trainingsfehler etwas höher und weniger stabil ist.

Der Fehler auf den Trainings- und Validierungsdaten verläuft bei Versuch 2 fast identisch zu Versuch 1, durch die zusätzlichen Augmentationen läuft das Training jedoch ein wenig länger, sodass die Performance sich weiter verbessert.

Versuch 3 zeigt auf den Trainingsdaten einen insgesamt deutlich niedrigeren Fehler, während der Fehler auf den Validierungsdaten bis zum Ende des Trainings etwas höher ist als bei den anderen Versuchen, trotz der längsten Trainingsdauer.

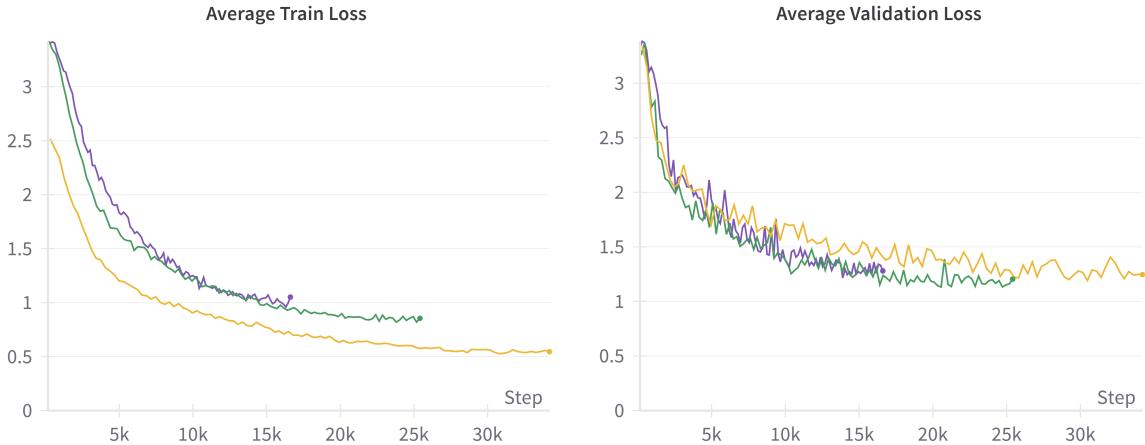


Abbildung 4.5: Trainings- und Validierungsfehler während des Contrastive Pre-Trainings
 (Lila: Versuch 1, Grün: Versuch 2, Gelb: Versuch 3).

4.2.2 Lineare Klassifikation

Die finalen Testergebnisse der linearen Klassifikation auf den Validierungsdaten sind in Tabelle 4.1 zusammengefasst. Die Trainingskurven sind in Abbildung 4.6 (Fehler), Abbildung 4.7 (Top-1 Accuracy) und Abbildung 4.8 (ID- und OOD-Confidence) zu sehen.

Tabelle 4.1: Testergebnisse der linearen Klassifikation.

Versuch	Top-1 Accuracy	ID-/OOD-Confidence (Δ)
1	71.4%	0.69 / 0.62 (0.07)
2	77.5%	0.76 / 0.65 (0.11)
3	75.3%	0.40 / 0.35 (0.05)

Die Trainingskurven zeigen sehr geringe Trainingsfehler für Versuch 1 und 2, aber einen viel höheren Fehler bei Versuch 3. Die Validierungsfehler der Versuche nähern sich wieder etwas an, wobei weiterhin ein deutlicher Abstand zwischen Versuch 3 und den anderen Versuchen besteht. Während der Validierungsfehler bei den Versuchen 1 und 2 also höher ist als der Trainingsfehler, ist es bei Versuch 3 umgekehrt.

Ähnliches kann in Bezug auf die Top-1 Accuracy beobachtet werden. Die Trainingsgenauigkeit ist bei Versuch 3 deutlich niedriger als bei den anderen Versuchen, während die Validierungsgenauigkeit ein ähnlich hohes Niveau erreicht.

Die ID-Confidence ist bei Versuch 2 deutlich höher als bei Versuch 1, während die OOD-Confidence bei beiden Versuchen vergleichbar bleibt. Versuch 3 zeigt jedoch die niedrigsten Confidence-Werte, sowohl für ID als auch für OOD, und hat auch den niedrigsten Abstand zwischen den beiden Werten. In der Trainingskurve ist dabei kaum Konvergenz zu erkennen.

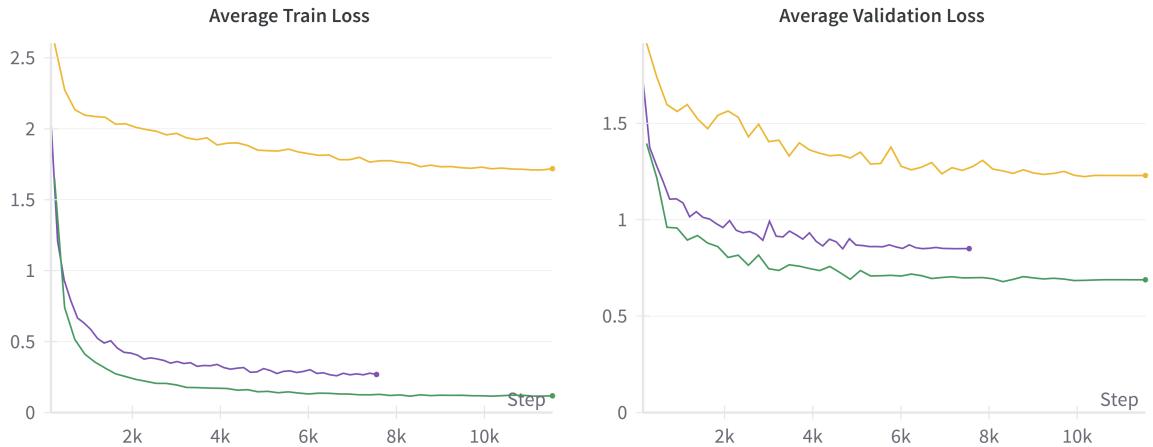


Abbildung 4.6: Trainings- und Validierungsfehler der linearen Klassifikation. **Lila:** Versuch 1,
Grün: Versuch 2, **Gelb:** Versuch 3.

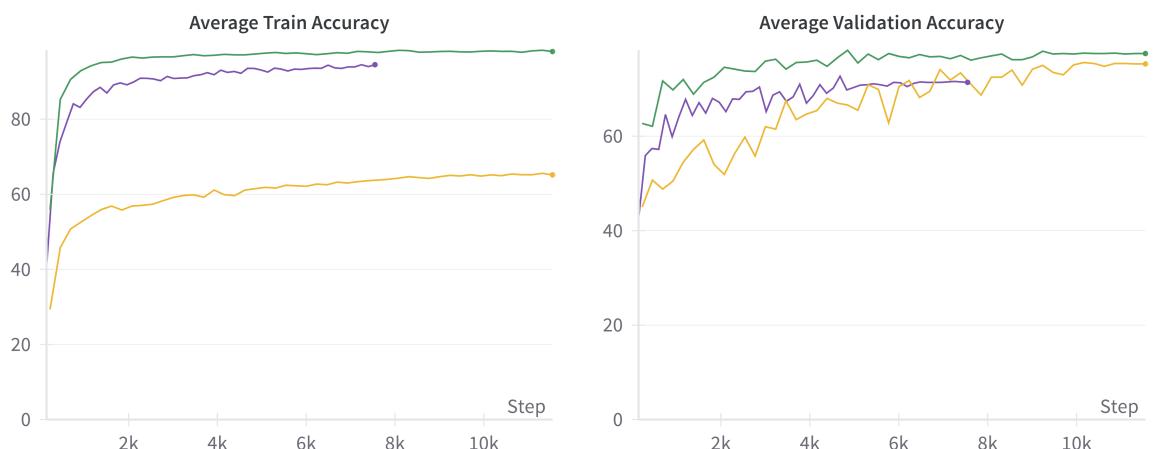


Abbildung 4.7: Trainings- und Validierungs-Accuracy während der linearen Klassifikation.
Lila: Versuch 1, **Grün:** Versuch 2, **Gelb:** Versuch 3.

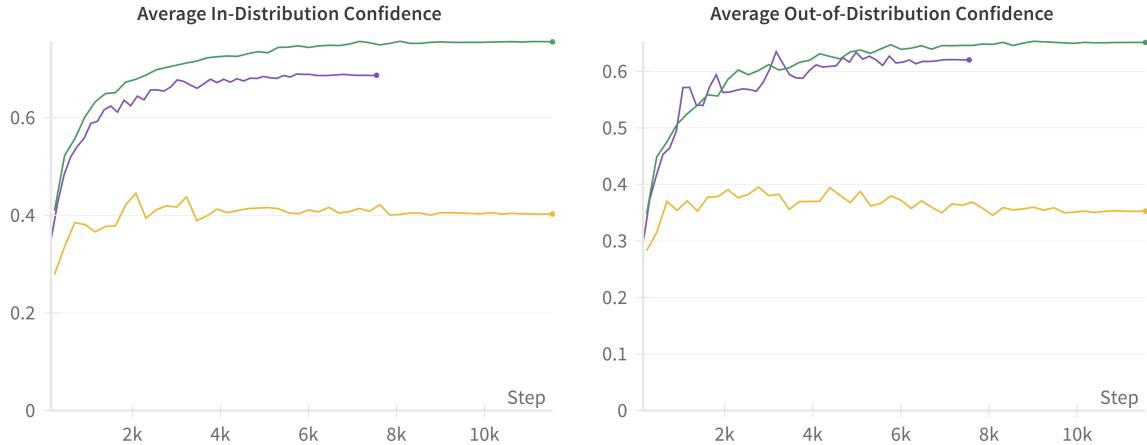


Abbildung 4.8: ID- und OOD-Confidence während der linearen Klassifikation (**Lila**: Versuch 1, **Grün**: Versuch 2, **Gelb**: Versuch 3).

4.3 Vergleich der Ergebnisse

Als Grundlage für die Diskussion und Beantwortung der Forschungsfragen werden die Ergebnisse mit und ohne In-Distribution- und Near Out-of-Distribution-Augmentationen verglichen. Es wird auf die Klassifikations-Performance (Accuracy) und die Out-of-Distribution-Detektion (ID-/OOD-Confidence) eingegangen.

4.3.1 Mit und ohne In-Distribution-Augmentationen

Die Klassifikations-Performance der Modelle verbesserte sich durch die In-Distribution-Augmentationen. Die Top-1 Accuracy stieg von 71.4% auf 77.5% und die ID-Confidence von 0.69 auf 0.76. Die OOD-Confidence stieg zwar ebenfalls von 0.62 auf 0.65, doch der Abstand zwischen ID- und OOD-Confidence wurde von 0.07 auf 0.11 angehoben.

Die Verbesserung der Performance durch die In-Distribution-Augmentationen ist auf die bessere Generalisierung der Modelle zurückzuführen. Die Modelle lernen durch die Augmentationen, die Objekte besser zu erkennen und zu klassifizieren, was sich in einer höheren Accuracy und ID-Confidence widerspiegelt. Die OOD-Confidence steigt ebenfalls, jedoch nicht so stark wie die ID-Confidence, was zu einem größeren Abstand zwischen den beiden Werten führt.

4.3.2 Mit und ohne Near Out-of-Distribution-Augmentationen

Die Klassifikations-Performance der Modelle verschlechterte sich durch die Near Out-of-Distribution-Augmentationen. Die Top-1 Accuracy sank von 77.5% auf 75.3% und die ID-Confidence von 0.76 auf 0.40. Die OOD-Confidence sank ebenfalls von 0.65 auf 0.35, wobei der Abstand zwischen ID- und OOD-Confidence von 0.11 auf 0.05 verringert wurde.

Die Verschlechterung der Performance durch die Near Out-of-Distribution-Augmentationen ist auf die schlechtere Generalisierung der Modelle zurückzuführen. Die Modelle lernen durch die Augmentationen, die Objekte schlechter zu erkennen und zu klassifizieren, was sich in einer niedrigeren Accuracy und ID-Confidence widerspiegelt. Die OOD-Confidence sinkt ebenfalls, jedoch nicht so stark wie die ID-Confidence, was zu einem kleineren Abstand zwischen den beiden Werten führt.

5 Diskussion

In diesem Kapitel werden die Ergebnisse dieser Arbeit interpretiert und im Kontext der Forschungsfragen diskutiert. Dabei wird auf die Eignung von DA-Fusion zur Generierung synthetischer Daten sowie auf die Wirksamkeit von Near Out-of-Distribution-Augmentationen im Supervised Contrastive Learning eingegangen.

5.1 Eignung von DA-Fusion für die synthetische Datengenerierung

Die Ergebnisse zeigen, dass die Anwendung von DA-Fusion zur Generierung synthetischer Daten vielversprechend ist. Die erzeugten Augmentationen wiesen, wie in [Unterabschnitt 4.1.1](#) beschrieben, eine hohe visuelle Qualität auf. DA-Fusion ist in der Lage, die Gebrauchsstände akkurat darzustellen und dennoch die Variation innerhalb der Daten zu erhöhen. Der einstellbare Grad der Variation ermöglicht es, die synthetischen Daten an die spezifischen Anforderungen des Anwendungsfalls anzupassen.

Die Steigerung der Accuracy von 71.4% auf 77.5% zeigt, dass die synthetischen Daten einen positiven Einfluss auf die Generalisierung haben können. Dass der Abstand zwischen ID- und OOD-Confidence-Werten vergrößert wurde, deutet darauf hin, dass die synthetischen Daten die Klassifikationsentscheidungen des Modells robuster gemacht haben. Dies könnte darauf hindeuten, dass die synthetischen Daten die Repräsentationen der In-Distribution-Daten verbessert haben, sodass das Modell besser zwischen verschiedenen Klassen unterscheiden kann. Um dies genauer zu untersuchen, könnte eine Visualisierung des erlernten Repräsentationsraums mittels t-SNE (van der Maaten & Hinton, [2008](#)) oder UMAP (McInnes et al., [2020](#)) hilfreich sein, was jedoch über den Rahmen dieser Arbeit hinausgeht.

Auf Grund der Tatsache, dass die Validierungsbilder die selben Laborbedingungen abbilden wie die Trainingsbilder, bleibt jedoch die Frage offen, wie sich das Training mit den synthetischen Daten auf die Generalisierung auf neue, reale Szenarien auswirkt.

Ein Nachteil von DA-Fusion ist der Rechen- und Zeitaufwand, der sich aus dem Fine-tuning des Stable Diffusion-Modells mit neuen Text-Embeddings für jede Klasse ergibt. Dabei ist auch die Anpassung der Hyperparameter nicht trivial und erfordert eine sorgfältige Abstimmung.

Des Weiteren ist zu beachten, dass DA-Fusion primär als Methode zur Datenaugmentation entwickelt wurde. Der Fokus dieser Arbeit richtete sich ausgehend von dem Anwendungsfall allerdings vor allem auf die synthetische Datengenerierung, um den MVIP-Datensatz erweitern. Im Kontext des Contrastive Learning wäre es interessant zu untersuchen, wie sich das Lernen der Repräsentationen verhält, wenn die unterschiedlichen Ansichten eines Bildes aus DA-Fusion generiert werden, anstatt herkömmliche Augmentationen zu verwenden.

Zusammenfassend lässt sich festhalten, dass DA-Fusion eine vielversprechende Methode zur Generierung synthetischer Daten ist, die die Generalisierungsfähigkeit von Modellen verbessern kann. Die Anwendung von DA-Fusion im Contrastive Learning könnte ein interessantes Forschungsfeld darstellen, um die Repräsentationen von Bildern weiter zu verbessern.

5.2 Wirksamkeit von Near Out-of-Distribution-Augmentationen im Supervised Contrastive Learning

Während die In-Distribution-Augmentationen eine positive Wirkung auf die Klassifikationsleistung hatten, zeigte sich ein gegenteiliger Effekt bei den Near Out-of-Distribution-Augmentationen. Die Top-1 Accuracy sank von 77.5% auf 75.3%, und auch die Confidence-Werte (ID und OOD) verschlechterten sich signifikant.

Die Ergebnisse könnten auf mehrere Faktoren zurückzuführen sein: 1) die Qualität der synthetisch erzeugten OOD-Daten, 2) die Komprimierung der Repräsentationen der ID-Daten und 3) die Implementierung der Near OOD-Augmentationen im Supervised Contrastive Learning.

Die Qualität der synthetisch erzeugten OOD-Daten war oft mangelhaft, mit deutlichen Artefakten oder sogar vollständiger Entkopplung vom ursprünglichen Konzept. Diese Daten können als „Far OOD“ klassifiziert werden, da sie zu weit von den In-Distribution-Daten entfernt sind, um als Hard Negatives zu fungieren. Es ist allerdings schwierig zu evaluieren, welche der OOD-Daten tatsächlich als Hard Negatives fungiert haben und welche nicht.

Die Verwendung dieser ungenauen OOD-Daten könnte die Repräsentationen der ID-Daten negativ beeinflusst haben. Das Modell könnte gelernt haben, zwischen den synthetischen OOD-Daten und den ID-Daten zu unterscheiden, anstatt zwischen den Klassen. Dies könnte dazu geführt haben, dass die Repräsentationen der ID-Daten „in die Mitte“ des Repräsentationsraums komprimiert wurden, was es für das Modell schwieriger machte, zwischen den Klassen zu unterscheiden.

Auch die Implementierung der Near OOD-Augmentationen im Supervised Contrastive Learning war eine Herausforderung. Die richtige Anpassung der Loss-Funktion, um Near OOD-Daten als Hard Negatives effektiv zu nutzen, erwies sich als komplex. Im gewählten Ansatz wurden sowohl ID- als auch OOD-Augmentationen unter die normalen Trainingsbilder gemischt. Die Maskierung der kontrastiven Paare stellte zwar sicher, dass nur die ID-Augmentationen als Positives und die OOD-Augmentationen als Negatives verwendet wurden, jedoch unterscheidet sich somit auch die Anzahl der tatsächlich verwendeten Paare für jedes Batch. Dies könnte zu einer Verzerrung der Loss-Funktion geführt haben und sich auf unerwünschte Weise auf das Training ausgewirkt haben.

Um die Wirksamkeit von Near OOD-Augmentationen zu verbessern, könnten verschiedene Ansätze verfolgt werden. Zum einen könnte die Qualität der synthetisch erzeugten OOD-Daten verbessert werden, indem beispielsweise andere Augmentationsstärken ausprobiert werden. Zum anderen könnte die Implementierung der Near OOD-Augmentationen im Supervised Contrastive Learning überdacht werden, um die Repräsentationen der ID-Daten nicht zu beeinträchtigen. Eine Möglichkeit wäre eine Hard Negative Sampling-Strategie wie in (Jiang et al., 2024), bei der nur negativ-Beispiele aus einer gewissen Nähe im Repräsentationsraum ausgewählt werden.

Zusammenfassend lässt sich festhalten, dass die Near OOD-Augmentationen im Supervised Contrastive Learning nicht den gewünschten Effekt hatten. Die Ergebnisse deuten darauf hin, dass die Qualität der synthetisch erzeugten OOD-Daten und die Implementierung der Near OOD-Augmentationen im Supervised Contrastive Learning verbessert werden müssen, um die Robustheit des Modells gegenüber OOD-Daten zu erhöhen.

6 Fazit

Die synthetische Datengenerierung hat sich in den letzten Jahren als effektive Methode zur Verbesserung der Bildklassifikation etabliert. In dieser Arbeit wurde untersucht, wie sich Stable Diffusion-basierte Datenaugmentationen und Supervised Contrastive Learning (SCL) auf die Klassifikationsleistung auswirken. Durch den Einsatz von DA-Fusion zur Generierung von synthetischen Daten konnte eine signifikante Verbesserung der Modellgenauigkeit für In-Distribution-Daten (ID) erzielt werden. Die synthetischen Augmentationen trugen zur Steigerung der Modellgenauigkeit bei, indem sie die Generalisierungsfähigkeit des Modells verbesserten. Allerdings zeigte sich, dass Near Out-of-Distribution (OOD) Augmentationen im SCL nicht die erwartete Leistungssteigerung erbrachten, sondern die Klassifikationsgenauigkeit negativ beeinflussten.

6.1 Zusammenfassung der wichtigsten Erkenntnisse

In dieser Arbeit wurde untersucht, wie sich Stable Diffusion-basierte Datenaugmentationen und Supervised Contrastive Learning (SCL) auf die Bildklassifikation auswirken. Durch den Einsatz von DA-Fusion zur synthetischen Datengenerierung konnte eine signifikante Verbesserung der Klassifikationsleistung für In-Distribution-Daten (ID) festgestellt werden. Die synthetischen Augmentationen zeigten eine hohe visuelle Qualität und trugen zur Steigerung der Modellgenauigkeit bei, indem sie die Generalisierungsfähigkeit des Modells verbesserten. Allerdings zeigte sich, dass Near Out-of-Distribution (OOD) Augmentationen im SCL nicht die erwartete Leistungssteigerung erbrachten, sondern die Klassifikationsgenauigkeit negativ beeinflussten.

Die Hauptgründe hierfür liegen in der mangelhaften Qualität der OOD-Daten und der Herausforderung, diese korrekt in die kontrastive Lernstrategie zu integrieren. Während ID-Augmentationen erfolgreich in das Training eingebunden werden konnten, führten Near OOD-Daten zu einer Verschlechterung der Repräsentationen, was auf die Komplexität ihrer Implementierung und die Auswahl der Hard Negatives im SCL zurückzuführen ist.

6.2 Beantwortung der Forschungsfragen

Die Forschungsfragen dieser Arbeit lauteten:

1. Wie gut eignet sich DA-Fusion zur Generierung synthetischer Daten für die Bildklassifikation?
2. Inwiefern tragen Near Out-of-Distribution-Augmentationen im Supervised Contrastive Learning zur Verbesserung der Modellgenauigkeit bei?

Zur ersten Forschungsfrage konnte gezeigt werden, dass DA-Fusion eine vielversprechende Methode zur Generierung von In-Distribution-Augmentationen darstellt. Diese synthetischen Daten führten zu einer klaren Verbesserung der Modellgenauigkeit und der Robustheit gegen Datenvariationen.

Die zweite Forschungsfrage ergab hingegen, dass Near OOD-Augmentationen im gewählten Ansatz nicht effektiv waren. Statt einer Verbesserung der Klassifikationsergebnisse führten sie zu einer Reduktion der Modellgenauigkeit, was insbesondere auf die unzureichende Qualität der OOD-Daten und die Herausforderung der Loss-Funktion beim Umgang mit Near OOD-Daten im SCL zurückzuführen ist.

6.3 Ausblick und potenzielle Weiterentwicklungen

Diese Arbeit zeigt Potenziale und Herausforderungen bei der Verwendung von synthetischen Augmentationen im Kontext des Contrastive Learning auf. Zukünftige Forschungen könnten folgende Aspekte untersuchen:

- *Verbesserung der OOD-Datenqualität:* Eine präzisere Kontrolle über die Erzeugung von Near OOD-Daten könnte die Robustheit des Modells erhöhen. Der Einsatz fortgeschrittenster Techniken zur Qualitätssicherung synthetischer Daten wäre ein vielversprechender Ansatz.
- *Optimierung der kontrastiven Lernstrategie:* Die Integration von Methoden wie Hard Negative Sampling könnte die Effektivität der Near OOD-Augmentationen erhöhen. Auch die Anpassung der Loss-Funktion könnte helfen, die Balance zwischen ID- und OOD-Daten besser zu steuern.
- *Transfer auf reale Szenarien:* Die Anwendung der generierten synthetischen Daten auf reale Bildklassifizierungsprobleme wäre ein weiterer Schritt, um die Generalisierung der Modelle zu validieren. Hierbei wäre es interessant zu sehen, wie gut die synthetischen Augmentationen unter variierenden realen Bedingungen funktionieren.

Abschließend bleibt festzuhalten, dass Stable Diffusion-basierte Datenaugmentationen das Potenzial haben, die Bildklassifikation signifikant zu verbessern. Die Herausforderungen bei der Einbindung von Near OOD-Daten erfordern jedoch weiterführende Forschung, um ihr volles Potenzial im Supervised Contrastive Learning auszuschöpfen.

Literatur

- Burgmer, C. (2005, Juli). *Schema eines künstlichen Neurons* [Lizenziert unter der Creative Commons Attribution-Share Alike 3.0 Unported Lizenz: <https://creativecommons.org/licenses/by-sa/3.0/deed.en>. Editierte Beschriftung.]. <https://commons.wikimedia.org/wiki/File:ArtificialNeuronModel.png>
- Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. (2020). A Simple Framework for Contrastive Learning of Visual Representations.
- Cun, Y. L., Boser, B., Denker, J. S., Howard, R. E., Hubbard, W., Jackel, L. D., & Henderson, D. (1989). Handwritten digit recognition with a back-propagation network. In *Advances in Neural Information Processing Systems 2* (S. 396–404). Morgan Kaufmann Publishers Inc.
- Fan, L., Chen, K., Krishnan, D., Katahi, D., Isola, P., & Tian, Y. (2023). Scaling Laws of Synthetic Images for Model Training ... for Now. <https://arxiv.org/abs/2312.04567>
- Foster, D. (2020). *Generatives Deep Learning*. O'Reilly.
- Gal, R., Alaluf, Y., Atzmon, Y., Patashnik, O., Bermano, A. H., Chechik, G., & Cohen-Or, D. (2022). An Image is Worth One Word: Personalizing Text-to-Image Generation using Textual Inversion. <https://arxiv.org/abs/2208.01618>
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press. <http://www.deeplearningbook.org>
- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative Adversarial Networks. <https://arxiv.org/abs/1406.2661>
- Google for Developers. (2022). *Overview of GAN Structure* [Lizenziert unter der Creative Commons Attribution 4.0 Lizenz: <https://creativecommons.org/licenses/by/4.0/>]. https://developers.google.com/machine-learning/gan/gan_structure
- Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). On Calibration of Modern Neural Networks. <https://arxiv.org/abs/1706.04599>
- Hendrycks, D., & Gimpel, K. (2018). A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks. <https://arxiv.org/abs/1610.02136>
- Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the Dimensionality of Data with Neural Networks. *Science*, 313(5786), 504–507. <https://doi.org/10.1126/science.1127647>
- Jiang, R., Nguyen, T., Ishwar, P., & Aeron, S. (2024). Supervised Contrastive Learning with Hard Negative Samples. <https://arxiv.org/abs/2209.00078>

- Keshtmand, N., Santos-Rodriguez, R., & Lawry, J. (2022). Understanding the properties and limitations of contrastive learning for Out-of-Distribution detection.
- Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., & Krishnan, D. (2021). Supervised Contrastive Learning.
- Kingma, D. P., & Welling, M. (2013). Auto-Encoding Variational Bayes. <https://arxiv.org/abs/1312.6114>
- Koch, P., Schlüter, M., Briese, C., & Chavan, V. (2023, November). MVIP: A Dataset for Industrial Part Recognition. <https://doi.org/http://dx.doi.org/10.24406/fordatis/300>
- Liang, S., Li, Y., & Srikant, R. (2020). Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks. <https://arxiv.org/abs/1706.02690>
- Liu, R. (2021). Understand and Improve Contrastive Learning Methods for Visual Representation: A Review. <https://arxiv.org/abs/2106.03259>
- Lu, Y., Shen, M., Wang, H., Wang, X., van Rechem, C., Fu, T., & Wei, W. (2024). Machine Learning for Synthetic Data Generation: A Review. <https://arxiv.org/abs/2302.04062>
- McInnes, L., Healy, J., & Melville, J. (2020). UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. <https://arxiv.org/abs/1802.03426>
- Meng, C., He, Y., Song, Y., Song, J., Wu, J., Zhu, J.-Y., & Ermon, S. (2022). SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations. <https://arxiv.org/abs/2108.01073>
- Mitchell, T. M. (1997). *Machine Learning*. McGraw Hill.
- Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., & Sutskever, I. (2021). Learning Transferable Visual Models From Natural Language Supervision. <https://arxiv.org/abs/2103.00020>
- Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., & Chen, M. (2022). Hierarchical Text-Conditional Image Generation with CLIP Latents. <https://arxiv.org/abs/2204.06125>
- Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022). High-Resolution Image Synthesis with Latent Diffusion Models. <https://arxiv.org/abs/2112.10752>
- Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088), 533–536. <https://doi.org/10.1038/323533a0>
- Saha, S. (2018). *A Comprehensive Guide to Convolutional Neural Networks - the ELI5 way* [Aufgerufen: 17.09.2024]. <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>
- Shen, B. (2021). E-commerce Customer Segmentation via Unsupervised Machine Learning. *The 2nd International Conference on Computing and Data Science*. <https://doi.org/10.1145/3448734.3450775>
- Shorten, C., & Khoshgoftaar, T. (2019). A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*, 6. <https://doi.org/10.1186/s40537-019-0197-0>
- Sohl-Dickstein, J., Weiss, E. A., Maheswaranathan, N., & Ganguli, S. (2015). Deep Unsupervised Learning using Nonequilibrium Thermodynamics. <https://arxiv.org/abs/1503.03585>
- Tian, Y. (2023). *SupContrast* [GitHub Repository]. <https://github.com/HobbitLong/SupContrast>

- Tian, Y., Sun, C., Poole, B., Krishnan, D., Schmid, C., & Isola, P. (2020). What Makes for Good Views for Contrastive Learning? <https://arxiv.org/abs/2005.10243>
- Trabucco, B. (2024). *DA-Fusion* [GitHub Repository]. <https://github.com/brandontrabucco/da-fusion>
- Trabucco, B., Doherty, K., Gurinas, M., & Salakhutdinov, R. (2023). Effective Data Augmentation With Diffusion Models. <https://arxiv.org/abs/2302.07944>
- van Breugel, B., Qian, Z., & van der Schaar, M. (2023). Synthetic data, real errors: how (not) to publish and use synthetic data. <https://arxiv.org/abs/2305.09235>
- van der Maaten, L., & Hinton, G. (2008). Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9(86), 2579–2605. <http://jmlr.org/papers/v9/vandermaaten08a.html>
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention Is All You Need. <https://arxiv.org/abs/1706.03762>
- Wagner, M. (2022, Juni). *BMBF-Fördermaßnahme „Ressourceneffiziente Kreislaufwirtschaft - Innovative Produktkreisläufe (ReziProK)“* [ReziProK Transferkonferenz am 23. und 24. Juni 2022 im Tagungswerk, Berlin]. https://innovative-produktkreislaeufe.de/Projekte/EIBA/_/ReziProK_Statuskonferenz_Presentation_EIBA_2022_06_23_final.pdf
- Zhou, Z.-H. (2021). *Machine Learning*. Springer.

Anhang

Codeblock 1: Begrenzung des Datensatzes auf die 20 „CarComponent“ Klassen in der „MVIP-Dataset“-Klasse.

```
1 # Select subset of classes from MVIP dataset
2 class_names = []
3
4 for class_name in [
5     f for f in os.listdir(MVIP_DIR) if os.path.isdir(os.path.join(MVIP_DIR, f))
6 ]:
7     meta_file = open(os.path.join(MVIP_DIR, class_name, "meta.json"))
8     meta_data = json.load(meta_file)
9
10    if SUPER_CLASS in meta_data['super_class']:
11        class_names.append(class_name)
12
13    meta_file.close()
14
15    # Limit the number of classes
16    del class_names[NUM_CLASSES:]
```

Codeblock 2: Funktion zum Laden der Augmentationen in der „MViPDataset“-Klasse von SupCon.

```
1 def parse_augs(self, aug_type):
2     #Parse the augmentation directory and return all augmented images and labels.
3
4     if aug_type == "id":
5         aug_dir = self.aug_dir_id
6         examples_per_class = self.aug_ex_id
7         label_sign = 1
8     else: # OODs
9         aug_dir = self.aug_dir_ood
10        examples_per_class = self.aug_ex_ood
```

```

11     label_sign = -1 # OOD augmentations receive negative labels
12
13     augs = os.listdir(aug_dir)
14
15     # Shuffle before potentially limiting num of examples per class
16     shuffle_idx = np.random.permutation(len(augs))
17     augs = [os.path.join(aug_dir, augs[i]) for i in shuffle_idx]
18
19     if examples_per_class > 0:
20         del augs[examples_per_class*len(self.class_names)*4:] # num_synthetic=4
21
22     # Get labels from file names
23     labels = []
24     for class_name in self.class_names:
25         for file in augs:
26             if class_name in file:
27                 labels.append(self.class_to_label_id[class_name] * label_sign)
28
29     # Return masks as None
30     masks = [None for _ in range(len(augs))]
31
32     return augs, labels, masks

```

Codeblock 3: Vollständige Klasse für die verwendete Verlustfunktion im Supervised Contrastive Learning.

```

1 class SupConLoss(nn.Module):
2     def __init__(self, temperature=0.07, contrast_mode='all',
3                  base_temperature=0.07):
4         super(SupConLoss, self).__init__()
5         self.temperature = temperature
6         self.contrast_mode = contrast_mode
7         self.base_temperature = base_temperature
8
9     def forward(self, features, labels=None, mask=None):
10        device = (torch.device('cuda')
11                  if features.is_cuda
12                  else torch.device('cpu'))
13
14        if len(features.shape) < 3:
15            raise ValueError(`features` needs to be [bsz, n_views, ...], '
16                             'at least 3 dimensions are required')

```

```

17     if len(features.shape) > 3:
18         features = features.view(features.shape[0], features.shape[1], -1)
19
20     batch_size = features.shape[0]
21
22     # Get/calculate contrastive mask for this batch (shape=[bsz, bsz])
23     # mask[i][j]=1 if sample j has the same class as sample i, otherwise 0
24     if labels is not None and mask is not None:
25         raise ValueError('Cannot define both `labels` and `mask`')
26     elif labels is None and mask is None:
27         mask = torch.eye(batch_size, dtype=torch.float32).to(device)
28     elif labels is not None:
29         labels = labels.contiguous().view(-1, 1)
30         if labels.shape[0] != batch_size:
31             raise ValueError('Num of labels does not match num of features')
32         mask = torch.eq(labels, labels.T).float().to(device)
33     else:
34         mask = mask.float().to(device)
35     # Update mask, so that self-contrast cases aren't considered as positive pairs
36     self_contrast_mask = 1 - torch.eye(batch_size, dtype=torch.float32).to(device)
37     mask *= self_contrast_mask
38     # Also make sure there are no positive pairs with any OOD samples (negative labels)
39     ood_mask = (labels < 0).float().to(device).detach()
40     mask *= (1 - ood_mask @ ood_mask.T)
41
42     # Determine whether all views or just one of each sample will be used as anchor
43     contrast_count = features.shape[1] # n_views
44     contrast_feature = torch.cat(torch.unbind(features, dim=1), dim=0)
45     if self.contrast_mode == 'one':
46         anchor_feature = features[:, 0]
47         anchor_count = 1
48     elif self.contrast_mode == 'all': # All features used as anchor & contrast
49         anchor_feature = contrast_feature
50         anchor_count = contrast_count
51     else:
52         raise ValueError('Unknown mode: {}'.format(self.contrast_mode))
53
54     # Compute logits (similarity scores for anchor & contrast features)
55     anchor_dot_contrast = torch.div(
56         torch.matmul(anchor_feature, contrast_feature.T),
57         self.temperature)
58     # For numerical stability

```

```

59     logits_max, _ = torch.max(anchor_dot_contrast, dim=1, keepdim=True)
60     logits = anchor_dot_contrast - logits_max.detach()
61
62     # Repeat masks to match the dimensions of the logits
63     mask = mask.repeat(anchor_count, contrast_count)
64     self_contrast_mask = self_contrast_mask.repeat(anchor_count, contrast_count)
65
66     # Ignore logits for OOD samples where OOD label is not the negative anchor label
67     non_ood_mask = 1 - ood_mask
68     valid_ood_mask = (labels == -labels.T).float().to(device).detach()
69     logits *= (non_ood_mask + valid_ood_mask).repeat(anchor_count, contrast_count)
70
71     # Transform the logits into log-probabilities
72     exp_logits = torch.exp(logits) * self_contrast_mask
73     log_prob = logits - torch.log(exp_logits.sum(1, keepdim=True))
74
75     # Looking only at positive pairs, compute the mean log-probabilities for each anchor
76     mask_pos_pairs = mask.sum(1)
77     mask_pos_pairs = torch.where(mask_pos_pairs < 1e-6, 1, mask_pos_pairs)
78     mean_log_prob_pos = (mask * log_prob).sum(1) / mask_pos_pairs
79
80     # Final loss: average negative log-probabilities over all positive pairs
81     loss = - (self.temperature / self.base_temperature) * mean_log_prob_pos
82     loss = loss.view(anchor_count, batch_size).mean()
83
84     del self_contrast_mask, ood_mask, non_ood_mask, valid_ood_mask
85
86     return loss

```

Eigenständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende Bachelorarbeit mit dem Titel

Contrastive Learning mit Stable Diffusion-basierter Datenaugmentation

selbstständig und nur mit den angegebenen Hilfsmitteln verfasst habe. Alle Passagen, die ich wörtlich aus der Literatur oder aus anderen Quellen wie z. B. Internetseiten übernommen habe, habe ich deutlich als Zitat mit Angabe der Quelle kenntlich gemacht.

Hamburg, 20. September 2024