Example title

Masterarbeit

von

Johannes Baßler, B.Sc.

am

Institut für Industrielle Informationstechnik

Zeitraum:	01. 01. 2010 – 30. 06. 2010
Hauptreferent:	Prof. Dr.-Ing. Michael Heizmann
Betreuer:	M.Sc. Max Mustermann
	Dr.-Ing. Hans Maier[1]
	Dipl.-Ing. John Doe[1]

[1] XYZ AG - Karlsruhe

Erklärung

Ich versichere hiermit, dass ich meine Masterarbeit selbstständig und unter Beachtung der Regeln zur Sicherung guter wissenschaftlicher Praxis im Karlsruher Institut für Technologie (KIT) in der aktuellen Fassung angefertigt habe.
Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und wörtlich oder inhaltlich übernommene Stellen als solche kenntlich gemacht.

Karlsruhe, den 30. Juni 2010

Kurzfassung

Hier könnte eine deutsche Kurzfassung kommen.

Abstract

Here comes an english abstract. (This is optional: If not needed, please delete this environment)

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Kapitel 1

Einleitung

Die Vorlage `iiit-dipl.cls` ist die dritte Version der am IIIT fur Studien- und Diplomarbeiten verwendeten Vorlage. Die aktuellen Revision wurde um einige packages erweitert, die in den vielen Arbeiten in der einen oder anderen Form zu finden waren oder sich im taglichen Gebrauch als hilfreich erwiesen haben. Es sollen hier auch kleine Beispiele gegeben werden, um ein einheitliches Erscheinungsbild der am IIIT erstellten Arbeiten zu gewahrleisten. Dabei wird ein gewisses Grundwissen uber LaTeX vorausgesetzt.

In Kapitel 1 werden die technischen Grundlagen wie die eingebundenen packages sowie die Dateistruktur dieser Beispielarbeit beleuchtet. Kapitel **??** geht naher auf die von den packages sowie der Klasse zur Verfugung gestellten Umgebungen ein.

1.1 Dateistruktur

Das zentrale Dokument einer Studien-/ Diplomarbeit ist `BA_MA_NameOfStudent.tex`. Am Anfang dieser Datei werden die grundlegenden Einstellungen an der Dokumentenklasse vorgenommen. Diese werden als Argumente in eckigen Klammern an die Klasse ubergeben.

Die Optionen der Dokumentenklasse konnen beispielsweise folgendermasen eingestellt werden:

```
2  \documentclass[
3  %%%% --- Kodierung ---
4     %,latin1           % latin1 Kodierung
5     ,utf8              % UTF-8 Kodierung
6     %,ansinew          % ansinew Kodierung
7  %%%% --- Sprache ---
8     ,deutsch           % Deutsch
9     %,english          % Englisch
10 %%%% --- Druck ---
11    ,twoside           % beidseitiger Ausdruck
12    %,oneside          % einseitiger Ausdruck
13 %%%% --- Mathe ---
14    %,eqncentered          % Gleichungen zentrieren (optional)
15 %%%% --- Bibliographie ---
16 %   ,biblatex          % BibLaTeX anstatt BibTeX verwenden
17 %%%% --- Quellcode ---
```

```
18    ,lst                % Listings-Paket laden
19 %%%% --- Lesezeichen ---
20    %,nohyper           % keine hyperrefs verwenden
21    %,nobookmarks       % Bookmarks im PDF beim "Offnen ausblenden
22 %%%% --- auto-pst-pdf ---
23    ,off                % auto-pst-pdf ausschalten
24 %%%% --- subfigures ---
25    %,subfigure         % subfigure- anstatt subcaption-Paket
          verwenden
26                        % subfigure-Paket nicht mehr verwenden, wenn
                              mit neuer Arbeit begonnen wird (Paket
                            ist veraltet)
27 %%%% KOMA-Script Optionen ---
28    %,parskip=half      % fuer Absatzabstand statt -einzug
29                        % parskip bei Option deutsch automatisch
                            eingestellt
30 ]{iiit-dipl}
```

Damit wird die Klasse `iiit-dipl` mit doppelseitigem Layout, alphanumerischem Literaturverzeichnis ausgewahlt.

Einige der zur Verfugung stehenden switches sind in Tabelle 1.1 aufgefuhrt und erlautert. Defaultwerte sind mit eckigen Klammern markiert.

Um die Titelseite korrekt generieren zu konnen, mussen einige (selbsterklarende) Variablen gesetzt werden. Dabei ergeben die folgenden Einstellungen die Titelseite dieses Dokuments:

```
35 \arbeittyp{master} % master, diplom, bachelor, student,
      praktikant
36 \author{Johannes Baßler}
37 \akadgrad{B.Sc.} %akademischer Grad B.Sc., M.Sc., B.Eng., Dipl.-
      Ing., ...
38 \datevon{01.01.2010}
39 \datebis{30.06.2010}
40 \referent{Prof.~Dr.-Ing.~Michael~Heizmann} % optional bei
      arbeittyp "praktikant"
41 \betreuer{M.Sc.~Max~Mustermann \and Dr.-Ing.~Hans~Maier\thanks{
      XYZ AG - Karlsruhe} \and Dipl.-Ing.~John~Doe\samethanks}
```

Danach kann mit dem eigentlichen Dokument begonnen werden. Zuerst wird die frontmatter, also das vorbereitende Material prasentiert. Hierzu gehoren die Titelseite mit Einverstandniserklarung (`\maketitle`) und das Inhaltsverzeichnis (`\tableofcontents`), die automatisch generiert werden, sowie der Abstract, eine ca. 100-200 Wort lange Zusammenfassung der Arbeit inklusive der erzielten Ergebnisse auf Englisch.

Im Hauptteil (`\mainmatter`) wird die eigentliche Arbeit prasentiert. Der Hauptteil sollte die folgenden Teile beinhalten:

- Einleitung

Tabelle 1.1: Zentrale switches

switch	Bedeutung
latin1, utf8, [ansinew]	Zur direkten Eingabe von Sonderzeichen wie a, o, u und s muss dem Compiler die verwendete Kodierung mitgeteilt werden.
[deutsch], english	Sprache der automatisch gesetzten Begriffe
oneside, [twoside]	Einstellung fur doppelseitigen Druck. Beeinflusst die Kopfzeile und Leerseiten vor neuen Kapiteln.
biblatex	Verwendet biblatex statt bibtex. biblatex hat den Vorteil, dass es die UTF8-Kodierung unterstutzt, die beispielsweise von JabRef verwendet wird. Damit konnen beispielsweise auch Referenzen von Autoren mit Umlauten im Namen korrekt sortiert werden. biblatex erfordert, dass in der verwendeten Umgebung biber.exe als Bibtex-Compiler eingestellt ist. Falls biber nicht in der installierten TEX-Distribution enthalten ist, kann es nachtraglich von http://sourceforge.net/projects/biblatex-biber/files/biblatex-biber/ bezogen werden. Die Kompatibilitaten der verschiedenen biblatex und biber Versionen konnen der biblatex Dokumentation [4] entnommen werden.
lst	Sollte angegeben werden, wenn Quelltexte eingebunden werden. Mit dieser Option wird das Listings-Package geladen und es werden Styles mit Syntax-Highlighting fur verschiedene Programmiersprachen (C++, matlab, latex, other) vordefiniert. Wird das Listings-Package nicht benotigt, empfiehlt es sich, diese Option wegzulassen, um die Compiliergeschwindigkeit zu erhohen.
andere	Andere Optionen werden von der scrbook-Klasse behandelt.

- Zusammenfassung und Ausblick

Es ist sinnvoll, den Hauptteil in mehrere Dateien aufzuteilen. Zusatzliche Dateien konnen mit Hilfe des \include-Befehls eingebunden werden. Dabei wird der Inhalt der Datei so interpretiert, als stunde er direkt in der Hauptdatei.

Der letzte Teil des Hauptteils sind die Anhange. In ihnen werden z.B. Variablen, wichtige Teile des Quellcodes oder langere Herleitungen beschrieben. Nach dem \appendix-Befehl werden die Anhange automatisch alphabetisch nummeriert.

Am Abschluss des Dokuments steht das wichtige Literaturverzeichnis sowie ggf. das Bilder- und Tabellenverzeichnis.

```
55      \maketitle
56      \include{abstract}
```

Da die Vorlage, also `iiit-dipl.cls` *nicht* verandert werden soll, steht fur eigene Erweiterungen wie zum Beispiel neue nutzliche packages die Datei `erweiterungen.tex` zur Verfugung, die in die Hauptdatei eingebunden wird.

```
32 \input{erweiterungen.tex}
```

In ihr wird standardmasig das Verzeichnis fur die Bilder gesetzt sowie ein Beispiel fur eine erweiterte Trennregel gegeben. Zweck dieser Datei ist es, alle Anpassungen, die die aktuelle Arbeit benotigt, zentral zu speichern und damit unabhanig von der eigentlichen Diplomklasse zu machen.

```
1 \graphicspath{{bilder/}} % Bilder werden im "bilder"-Unterordner
     gesucht
2 \hyphenation{Lja-pu-now} % Trennen von unbekannten Woertern
```

1.2 Allgemeine Hinweise

Im Laufe der Zeit haben sich verschiedene Vorgehensweisen bei der Erstellung einer Diplom- bzw. Studienarbeit als nutzlich erwiesen. Einige von ihnen sollen hier ohne Anspruch auf Vollstandigkeit aufgefuhrt werden.

Fruh dokumentieren Es macht Sinn, auch Zwischenergebnisse ggf. auch in LaTeX zu dokumentieren.

Sinnvolle Dateigrosen LaTeX bietet die Moglichkeit, die Arbeit auf mehrere Dateien aufzuteilen und diese in die Hauptdatei einzufugen.

Kapitel 2

Motivation

2.1 Anomaliedetektion in der Industrie

Hier wird die Relevanz von Anomaliedetektion in der Industrie erläutert.

2.2 Unüberwachte Anomaliedetektion

Hier wird ausgeführt warum überwachte Methoden an ihre Grenzen stoßen und warum unüberwachte Methoden sinnvoll sind.

2.3 Laufzeitoptimierung

Dann wird ausgeführt, warum eine Laufzeitoptimierung sinnvoll ist und warum es auch heute noch sinnvoll sein kann, auf teure GPUs zu verzichten.

Kapitel 3

Grundlagen

Hier wird kurz aufgezählt was erläutert wird. Mehr nicht. Es sollen hier nur Elemente erläutert werden, die für mindestens zwei der Methoden relevant sind. k-center greedy (PatchCore), Autoencoder (efficientad) oder Backpropagation (simplenet) also zB nicht.

3.1 Datensatz: MVTecAD

Erläuterung des Datensatzes MVTecAD. Beispielbilder.

3.2 Eigener Datensatz (Granulat)

Details und Beispiele zu eigenem Datensatz.

3.3 Residuale Netzwerke

In diesem Abschnitt werden die Grundlagen von Residual Networks (textbf„ResNets“) erläutert. Diese spielen für die Merkmalsextraktoren („Feature Extractors“) in den im Haupteil der Arbeit (TODO –> Link) eine wichtige Rolle. Der Schwerpunkt hierbei liegt auf der grundlegenden Idee, der Rolle, die ResNets historisch in der Entwicklung von Deep Learning gespielt haben und vor allem den Aspekten, die für die Anwendung in dieser Arbeit relevant sind. Für detailierte Informationen zu ResNets wird auf das Paper von Kaiming He et al. [3] und die zahlreichen Erläuterungen in der Literatur verwiesen.

3.3.1 Hintergrund & Idee hinter „ResNets“

Tiefe neuronale Netze (Deep Neural Networks, DNNs) eignen sich hervorragend für das Lernen hierarchischer Darstellungen aus Daten, aber sie stehen vor Herausforderungen, wenn sie „tiefer“ werden.„Tiefe“ bezeichnet in diesem Zusammenhang die Anzahl an Schichten, die sequentiell durchlaufen werden, um das Endergebnis bzw. die Ausgabe zu erhalten. Tiefere Netze können

komplexere Merkmale in Daten erfassen, was für Aufgaben wie die Bildklassifizierung, bei der Objekte und Muster komplizierte Details aufweisen können, entscheidend ist. Einer der großen Herausforderungen bei tiefen Netzen ist das Problem der „verschwindenden Gradienten“ (*engl.* **Vanishing Gradients**). Diese Gradienten sind entscheidend für das Training eines Neuronalen Netzes, insofern, als dass das Optimierungsverfahren des Gradientenabstiegs die Grundlage auch moderner Optimierer wie „Adam“ (TODO –> Ref) ist. Dieses Problem lässt sich anschaulich dadurch erklären, dass frühe Gradienten, was sich leicht mithilfe der Kettenregel zeigen lässt, eine Multiplikation von allen nachfolgenden (im Sinne der Infernenzrichtung - „forward pass“) Gradienten darstellt. Betrachtet man nun ein sehr tiefes Netz, das heißt, viele Gradienten, die miteinander multipliziert werden und den wahrscheinliche Fall von Gradienten, die kleiner als 1 sind, so wird schnell klar, dass frühe Schichten einen sehr kleinen Gradienten haben können. Dies macht es schwierig, die Gewichte der frühen Schichten zu aktualisieren, was den Lernprozess behindert. Dabei ist es vor allem die Tiefe, die Neuronalen Netzen das Generalisieren von komplexen Zusammenhängen ermöglicht („Deep Learning “)
Residuale Netze, allgemein bekannt als ResNets und im Folgenden auch so bezeichnet, wurden von Kaiming He et al. (TODO –> Ref) in ihrem Paper von 2015 vorgestellt und bieten eine einfache und dennoch effektive Methode an, wie dieses Problem angegangen werden kann. Die Grundidee besteht darin, Verknüpfungen zwischen den Schichten einzuführen, die es dem Netz ermöglichen, eine oder mehrere Schichten zu „überspringen“. Anstatt die gewünschte Abbildung also direkt zu lernen, lernen ResNets die residuale Abbildung, was der Differenz zwischen Ein- und Ausgabe entspricht. Die Eingabe wird über sogenannte „Shortcut (Skip) Connections“, also einfach die Identitätsabbildung, vom Eingang zur residualen Ausgabe weitergeleitet, um dann durch Summation wieder miteinander verknüpft zu werden. Das Problem der verschwindenden Gradienten wird dadurch entschärft, dass die Gradienten direkt in frühere Schichten zurückfließen können. Anschaulich lässt sich das durch die Tatsache erklären, dass die Identitätsabbildung einen konstanten Gradienten von 1 besitzt. Weil sich die Summenbildung am Ausgang eines nachfolgend noch im Detail besprochenen Residual-Blocks auch bei der Gradientenbildung als Summation widerspiegelt, ist der Gradient einer jeden Schicht nicht mehr das Produkt, sondern vielmehr die Summe aller nachfolgenden Gradienten. (Optional TODO: Formel) Das zugrundeliegende Paper ist eines der meistzitierten Paper im Bereich des Deep Learning und hat die Entwicklung von Deep Learning maßgeblich beeinflusst.

3.3.2 Residual Block & Architektur

Der Grundbaustein eines ResNet ist der Residualblock. Er besteht aus zwei Hauptpfaden: dem Identitätspfad (der Abkürzungsverbindung) und dem Residualpfad (dem Hauptfaltungspfad). Mathematisch wird die Ausgabe eines Residualblocks wie folgt berechnet:

$$\text{Output} = F(\text{Input}) + \text{Input}$$

wo F die Residualabbildung ist, die durch die Hauptfaltungsschichten gelernt wird. Nachfolgende Abbildung zeigt eine vereinfachte Darstellung eines „Building Blocks“ bzw. Residual Block.

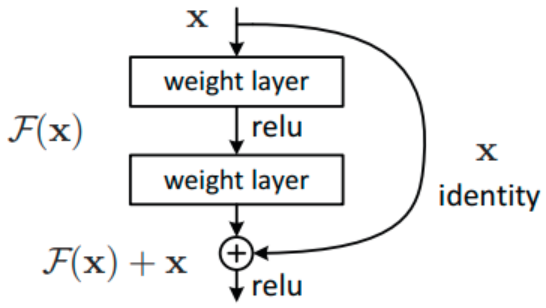

Abbildung 3.1: Residual Block (TODO –> Ref)

Ein ResNet besteht aus mehreren Residualblöcken, die sequentiell durchlaufen werden. Es ist also eine „Feed-Forward“-Architektur, weil die Daten zur Inferenz nur in eine Richtung durch das Netz fließen. Durch das „Stapeln “ können dann sehr tiefe Netze erstellt werden, die sich dennoch aufgrund der oben beschriebenen Eigenschaften gut optimieren bzw. trainieren lassen. Es existieren zahlreiche verschiedene Varianten von ResNets, die sich in der Art und Anzahl ihrer Residual Blöcke unterscheiden. Nachfolgende Tabelle gibt einen Überblick über die drei, in dieser Arbeit vor allem verwendeten Varianten: ResNet18, ResNet34 und WideResNet50

Tabelle 3.1: Vergleich verschiedener ResNet Varianten (TODO –> Ref)

Netzwerk	Tiefe	# Parameter	Top-1 Fehlerrate[1]	Inferenz auf RBP4[2]
ResNet18	18	$11{,}7 \cdot 10^6$	30,24%	0, 82s
ResNet34	34	$21{,}8 \cdot 10^6$	26,70%	1, 45s
WideResNet50	50	$68{,}9 \cdot 10^6$	22,53%	3, 00s

Zu erkennen ist eindeutig, dass die Anzahl der Parameter und die Inferenzzeit mit der Tiefe des Netzes steigt. Gleichzeitig lässt sich aber auch eine Verbesserung der Top-1 Fehlerrate erkennen, je tiefer bzw. mächtiger das Netz ist.
Einer der Hauptvorteile von ResNets ist die Fähigkeit, Merkmale pyramidenförmig durch das Netz zu verbreiten. Das bedeutet, dass das Netz Merkmale auf verschiedenen Abstraktionsebenen erfassen kann, von Low-Level-Merkmalen wie Kanten und Ecken bis zu High-Level-Merkmalen wie Objektteilen und semantischen Konzepten. Mit zunehmender Tiefe wird also die Auflösung der „Feature Maps“ (TODO –> Ref) reduziert, während die Anzahl der Kanäle und die Komplexität der zugrundeliegenden Merkmale zunimmt. Dargestellt ist das in folgender Abbildung:

[1] in ILSVRC (TODO –> Ref)
[2] Raspberry Pi 4B 8GB. Betrachtet wurde die Laufzeit von einem Bild der Auflösung 224x224 und 3 (Farb-)Kanälen

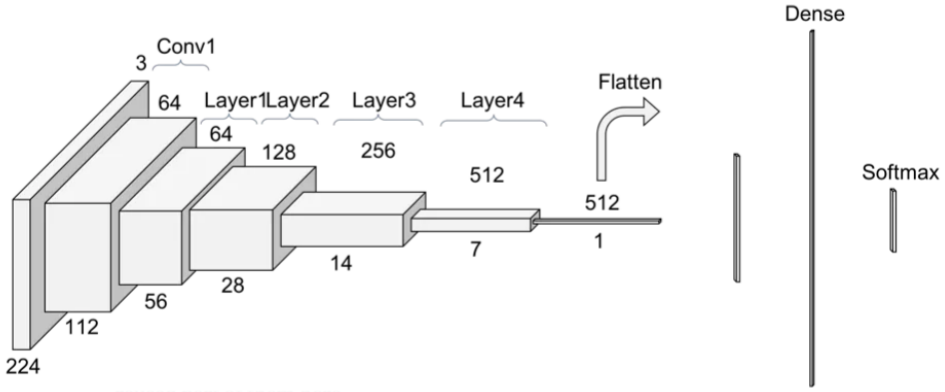

Abbildung 3.2: Pyramidale Merkmalsverteilung in ResNets (TODO –> Ref)

Die Quader in oben stehendem Netz stehen symbolhaft für die Auflösung der Feature Maps. Während die oben stehenden Zahlen die Anzahl der Kanäle repräsentieren, stehen die unten aufgeführten Zahlen für die räumliche Auflösung. Letztere hängt proportional von der Auflösung des Eingangsbildes ab und gilt für alle drei Modelle. Die Anzahl an Kanälen ist konstant für alle Auflösungen und für ResNet18 und ResNet34. Für das WideResNet50 gilt im Wensentlichen der gleiche Aufbau, allerdings sind die Kanäle „geweit“ gegenüber den anderen beiden Architekturen, was sich an der Vervierfachung der Anzahl an Kanäle durch Verwendung eines komplexeren Residual Blocks („Bottleneck“[8])zeigt. ResNet18 und ResNet34 unterscheiden sich ausschließlich in der Anzahl der Residual Blöcke bzw. der Tiefe des Netzes.
Alle der drei hier vorgestellten Architekturen lassen sich in fünf Faltungsschichten („Conv1“, „Layer1“, „Layer2“, „Layer3“, „Layer4“) unterteilen. Dem schließt sich eine „Average Pooling“-Schicht an, die die Auflösung der Feature Maps auf 1 reduizert („Flatten“). Dieser 1D-Vektor wird schließlich durch eine „Fully Connected“-Schicht („Dense“) auf die Anzahl der Klassen (1000) abgebildet. Schließlich wird durch eine „Softmax“-Aktivierungsfunktion Pseodo-Wahrscheinlichkeiten erzeugt, die den Axiomen von Kolmogorov entsprechen und somit als Auftrittswahrscheinlichkeiten interpretiert werden können.
In dieser Arbeit werden die ResNets als Feature-Extraktoren verwendet, weshalb die letzten beiden Schichten, also „Flatten“ und „Dense“ nicht verwendet werden. Die Begriffe „Layer1 - Layer4“ werden im Folgenden in dem hier beschriebenen Kontext verwendet.

3.3.3 ResNets as Feature Extractor für Unüberwachte Lernverfahren

Zusätzlich zu ihrem Erfolg bei der überwachten Bildklassifizierung haben ResNets Anwendungen als leistungsstarke Merkmalsextraktoren bei Unüberwachten Klassifizierungsaufgaben wie der Anomalieerkennung gefunden. In unüberwachten Szenarien wie der Anomalieerkennung stehen oft keine markierten (gelabelten) Daten zur Verfügung, wie bereits in (TODO) beschrieben, um ein Modell explizit bzw. überwacht zu trainieren. Stattdessen verlässt man sich auf das

Lernen von Darstellungen normaler Daten und identifiziert dann Abweichungen als Anomalien. ResNets können mit ihrer Fähigkeit, umfangreiche und hierarchische Merkmale zu erfassen, dazu verwendet werden, sinnvolle Merkmale aus den Daten zu extrahieren. Es kann mithilfe dieser Netze eine kompaktere und aussagekräftigere Darstellung der Daten erzeugt werden, die die Grundlage sind, um auch komplexere Anomalien zu erkennen.
Dabei werde in allen hier besprochenen Fällen ResNets verwendet, die auf dem ImageNet Datensatz trainiert wurden. Hierzu werden die Gewichte offizieller Implementierungen von ResNets verwendet, womit das eigentliche Vortraining entfällt und Reproduzierbarkeit, Konsistenz und Vergleichbarkeit der Ergebnisse gewährleistet wird.

3.4 SPADE

Die Methode **SPADE** ist ein wichtiger Meilenstein in der Unüberwachten Anomalieerkennung. Zahlreiche erfolgreiche Methoden bauen auf SPADE auf und übernehmen wichtige Elemente und Konzepte. Das Paper wurde am 5. Mai 2020 von Niv Cohen und Yedid Hoshen von der Hebräischen Universität von Jerusalem veröffentlicht. Im Folgenden wird SPADE vorgestellt.

3.4.1 Funktionsweise

3.5 PaDiM

Als zweites Paper wird PaDiM vorgestellt. Analog zu SPADE soll eine klare Linie hinzu PatchCore gezogen werden können.

3.6 Raspberry Pi 4B

3.6.1 Allgemeines

Der Raspberry Pi 4, der im Juni 2019 von der Raspberry Pi Foundation veröffentlicht wurde, ist ein kleiner, erschwinglicher und vielseitiger Einplatinencomputer.

Die zentrale Recheneinheit (CPU) des Raspberry Pi 4 ist eine 64-bit Quad-Core-ARM-Cortex-A72-CPU, die mit 1,8 GHz (ältere Versionen mit 1,5 GHz) taktet. Er ist in drei Speicherkonfigurationen erhältlich: 1 GB, 2 GB, 4 GB und 8 GB LPDDR4 RAM mit 3200 MHz. Die Integration eines Broadcom VideoCore VI-Grafikprozessors verbessert die Multimedia-Fähigkeiten und ermöglicht eine flüssige Videowiedergabe und 3D-Grafik-Rendering.

Ein bemerkenswertes Merkmal des Raspberry Pi 4 sind die Anschlussmöglichkeiten. Er verfügt über zwei USB 3.0-Ports und zwei USB 2.0-Ports, die den Anschluss verschiedener Peripheriegeräte ermöglichen. Dualband-Wi-Fi (2,4GHz und 5GHz) und Gigabit-Ethernet sorgen für eine zuverlässige Netzwerkverbindung. HDMI- und Audioausgänge unterstützen hochauflösende Bildschirme und Audiogeräte. So können beispielsweise zwei 4K-Displays angeschlossen werden.

Das Gerät ist mit mehreren Betriebssystemen kompatibel, darunter Raspberry Pi OS (früher Raspbian), Linux-Distributionen und sogar Windows 10, je nach Vorlieben und Anforderungen des Benutzers. In dieser Arbeit wurde Raspberry Pi OS in der 64-bit Version verwendet. Das auf Debian basierende Betriebssystem ist für die Hardware des Raspberry Pi optimiert und ist kompatibel mit allen notwendigen Softwarepaketen.

Die 40 GPIO-Pins des Raspberry Pi 4 sind eine vielseitige Hardwareschnittstelle, die zahlreiche Anwendungen in verschiedenen Bereichen ermöglicht. Die Anwendungen des Raspberry Pi 4 reichen von Bildungs- und Hobbyprojekten bis hin zu professionellen Unternehmungen. Er kann für Aufgaben wie Heimautomatisierung, Robotik, Webserver und Softwareentwicklung verwendet werden. Dank seines geringen Stromverbrauchs von etwa 2,7 W (Idle) bis maximal 6,4 W (Volllast, keine Peripherie)[2] eignet er sich für eingebettete Systeme und Internet-of-Things-Anwendungen (IoT).[1][7]

3.6.2 Ressourcenbeschränktheit

Die Rechenkapazität des Raspberry Pi 4 ist für viele Anwendungen ausreichend. Dennoch muss die Leistungsfähigkeit realisitsch eingeordnet werden: Während der Paspberry Pi 4 eine Rechenleistung von 13,5 GLFOPS (Gleitkommaoperationen pro Sekunde) erreicht, ist ein auf der gleichen Architektur (ARM) beruhender Apple M1 Prozessor aus dem Jahr 2020, der für einfache Consumer Tätigkeiten konzipiert ist, mit 154 GFLOPS mehr als 11 mal so schnell. [5] Auch muss erwähnt werden, dass auf die Verwendung von modernen GPUs für die Inferenz in dieser Arbeit verzichtet wird, wie in Kapitel 2.3 bereits beschrieben. Setzt man die Leistung des Raspberry Pi 4 in Relation zu modernen GPUs, die viele Entwicklungen im Bereich *Depp Learning* überhaupt erst ermöglichten, so wird schnell klar, warum bei der Verwendung eines Raspberry Pi 4 von „Ressourcenbeschränktheit“ gesprochen werden kann. Auch wenn Rechenkapazität in FLOPS gemessen keine eindeutigen Schlüsse auf die Laufzeit eines konkreten Algorithmus zulässt, so kann trotzdem festgehalten werden, dass eine moderne GPU eine enorm höhere Rechenleistung besitzt. So erreichen moderne GPUs, wie Nvidia’s Geforce RTX4090 Ti 82 600 GFLOPS.[6]

In dieser Arbeit wird der Raspberry Pi 4 B in vielen Fällen als Plattform für die Inferenz der Modelle verwendet. Alle Laufzeitmessungen in dieser Arbeit müssen im Kontext dieser gerade beschriebenen Ressourcenbeschränktheit gesehen werden.

Kapitel 4

PatchCore

Die Methode **PatchCore** wurde erstmals am 15. Juni 2021 in Zusammenarbeit der Universität Tübingen und Amazon AWS im Paper „Towards Total Recall in Industrial Anomaly Detection“ veröffentlicht. In seiner zweiten Fassung vom 5. Mai 2022 wurde das Paper bei der CVPR 2022 (Computer Vision and Pattern Recognition) akzeptiert und mit über 260 Zitierungen eines der populärsten Paper im Bereich der Unüberwachten Anomaliedetektion. Die Grundlage dieses Ansatzes sind „Einbettungen“ (Im Folgenden: **Embeddings**) von Merkmalen (Im Folgenden: **Features**), die aus den Eingabebildern mithilfe eines auf „ImageNet“ vortrainiertem „Convolutional Neural Network (CNN)“ erzeugt werden. Wie in einigen vorangegangenen Veröffentlichungen im Bereich der Unüberwachten Anomaliedetektion, werden auch hier die Features in „Patches“ unterteilt, um die Lokalität der Anomalien zu erhalten. Diese werden folgend als **„Patch Features“** bezeichnet. Weiter wird die eigentliche Anomaliedetektion, wie bereits bei der Methode „SPADE“ (TODO –> Ref) mithilfe einer „Nächsten Nachbar Suche (Nearest Neighbor Search; NN)“ in einer „Memory-Bank “ durchgeführt. Die wesentliche Weitereentwicklung liegt vor allem in der Methode, wie die „Memory-Bank“ aufgebaut wird, die mit wenigen Beispielen nominaler „Patch Features“ eine möglichst hohe Representativität bietet. So wird die Laufzeit der NN-Suche deutlich reduziert, ohne dabei die Performanz zu beeinträchtigen. Auch gut 2 Jahre nach Veröffentlichung ist die PatchCore Methode insbesondere auf dem MVTecAD-Datensatz (TODO –> link) mit einer Genauigkeit (Auccuracy) von maximal 99,6% („PatchCore Ensemble“) absolut konkurrenzfähig und wird in vielen Veröffentlichungen als „State-of-the-Art“ Methode verwendet. Im Folgenden wird die Methode PatchCore genauer erläutert und die Implementierung in PyTorch beschrieben. Anschließend werden zahlreiche Adaptionen der Methode vorgestellt, die im Sinne des Ziels dieser Arbeit die Laufzeit der Methode reduzieren und dabei die Genauigkeit möglichst wenig beeinträchtigen sollen.

4.1 Funktionsweise

Zunächst kann zwischen zwei Phasen unterschieden werden: Der Trainingsphase und der Testphase.

4.1.1 Wesentliche Bestandteile

Zunächst werden nachfolgend die wesentlichen Bestandteile der Methode PatchCore erläutert.

Feature-Extraktion

Wie bereits kurz angerissen, geschieht die Feature Extraktion mithilfe eines vortrainiertem CNNs. Konkret werden die Netzwerke, die im Folgenden als **Backbone** bezeichnet werden, mit dem Datensatz „ImageNet“ trainiert, um Bilder einer von 1000 Klassen zuzuordnen. Obwohl diese Bildklassifikationsaufgabe zunächst nur wenig mit der hier vorliegenden binären Anomaliedetektion zu tun hat, haben sich diese Netzwerke als sehr gut geeignet erwiesen, um Merkmale (Feature) zu extrahieren, die auch für die Anomaliedetektion verwendet werden können. Dafür wird nicht etwa die Klassifikation, also das Ergebnis des Netzwerks nach durchlaufen des vollständigen Netzwer, verwendet, sondern die Ausgabe bestimmter Dabei interessiert nicht das Klassifikationsergebnis an sich, sondern viel mehr die sogenannten „Feature Maps“ von Zwischenschichten (Indermediate Layers) des Netzwerks. Eine Feature Map ist hier die Ausgabe einer bestimmten Schicht des Netzwerks, die sich zunächst an jeder Stelle im Netzwerk befinden kann. Betrachten wir als CNN ein Residual Network (ResNet; TODO –> Ref)

Hier wird kurz angerissen, was es mit PatchCore auf sich hat.

- Popularität, Erscheinungsdatum, Performance, Ersteller, Zitierungen
- Wie funktioniert PatchCore
  - Training
  - Inferenz/Test
- Implementierung
- Baseline
- Anpassungen
- Fazit

NUR DER FCK!

4.2 Baseline

Hallo

4.3 Adaptionen

Hallo2

Kapitel 5

EfficientAD

5.1 Einleitung

In diesem Kapitel wird die Methode EfficientAD vorgestellt.

Kapitel 6

SimpleNet

Hier dann SimpleNet....

Anhang A

ASCII-Tabelle

…

Literaturverzeichnis

[1] Foundation, The Raspberry Pi: *Specifications for the Raspberry Pi 4 Model B*. https://www.raspberrypi.org/documentation/hardware/raspberrypi/bcm2711/rpi_DATA_2711_1p0_preliminary.pdf, 2023. Accessed: 2023-10-09.

[2] Geerling, Jeff: *Power Consumption Benchmarks | Raspberry Pi Dramble*. https://www.pidramble.com/wiki/benchmarks/power-consumption. Accessed: 2023-10-09.

[3] He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing und Sun, Jian: *Deep Residual Learning for Image Recognition*, 2015.

[4] Lehman, Philipp, Kime, Philip, Boruvka, Audrey und Wright, Joseph: *The biblatex Package: Programmable Bibliographies and Citations*, 2.7a, Juli 2013.

[5] Maine, The Weaver Computer Engineering Research Group of University of: *FLOPS/Watt of Various CPUs*. https://web.eece.maine.edu/~vweaver/group/green_machines.html, 2023. Accessed: 2023-10-09.

[6] PC Games Hardware, Maximilian Holm von: *RTX 4090: Mit starker Übertaktung mehr als 100 TFLOPS Rechenleistung erreicht*. https://www.pcgameshardware.de/Grafikkarten-Grafikkarte-97980/News/RTX-4090-Mit-starker-Uebertaktung-mehr-als-100-TFLOPS-Rechenleistung-erreicht-1405138/, 2022. Accessed: 2023-10-09.

[7] Wikipedia contributors: *Raspberry Pi 4 — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/w/index.php?title=Raspberry_Pi_4&oldid=1178956001, 2023. [Online; accessed 9-October-2023].

[8] Zagoruyko, Sergey und Komodakis, Nikos: *Wide Residual Networks*, 2017.