

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 Maier¹
Dipl.-Ing. John Doe¹

¹XYZ 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	iii
Tabellenverzeichnis	v
1 Einleitung	1
1.1 Dateistruktur	1
1.2 Allgemeine Hinweise	4
2 Motivation	5
2.1 Anomaliedetektion in der Industrie	5
2.2 Unüberwachte Anomaliedetektion	5
2.3 Laufzeitorientierung	5
3 Grundlagen	7
3.1 Datensatz: MVTecAD	7
3.2 Eigener Datensatz (Granulat)	7
3.3 Residuale Netzwerke	7
3.3.1 Hintergrund & Idee hinter „ResNets“	7
3.3.2 Residual Block & Architektur	8
3.3.3 ResNets as Feature Extractor für Unüberwachte Lernverfahren	10
3.4 SPADE	11
3.4.1 Funktionsweise	11
3.5 PaDiM	11
3.6 Raspberry Pi 4B	11
4 PatchCore	15
4.1 Funktionsweise	15
4.1.1 Wesentliche Bestandteile	16
4.2 Baseline	16
4.3 Adaptionen	17
5 EfficientAD	19
5.1 Einleitung	19
6 SimpleNet	21
A ASCII-Tabelle	23

Literaturverzeichnis

25

Abbildungsverzeichnis

3.1	Residual Block (TODO → Ref)	9
3.2	Pyramidale Merkmalsverteilung in ResNets (TODO → Ref)	10

Tabellenverzeichnis

1.1	Zentrale switches	3
3.1	Vergleich verschiedener ResNet Varianten (TODO -> Ref)	9

Kapitel 1

Einleitung

Die Vorlage `iiit-dipl.cls` ist die dritte Version der am IIIT für Studien- und Diplomarbeiten verwendeten Vorlage. Die aktuelle 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 gewährleisten. Dabei wird ein gewisses Grundwissen über \LaTeX vorausgesetzt.

In Kapitel 1 werden die technischen Grundlagen wie die eingebundenen packages sowie die Dateistruktur dieser Beispielarbeit beleuchtet. Kapitel ?? geht näher auf die von den packages sowie der Klasse zur Verfügung 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 übergeben.

Die Optionen der Dokumentenklasse können beispielsweise folgendermaßen eingestellt werden:

```
2 \documentclass[
3 %%%% --- Kodierung ---
4   ,latinl           % latinl 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 ausgewählt.

Einige der zur Verfügung stehenden switches sind in Tabelle 1.1 aufgeführt und erläutert. Defaultwerte sind mit eckigen Klammern markiert.

Um die Titelseite korrekt generieren zu können, müssen einige (selbsterklärende) Variablen gesetzt werden. Dabei ergeben die folgenden Einstellungen die Titelseite dieses Dokuments:

```

35 \arbeitstyp{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
    arbeitstyp "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 präsentiert. Hierzu gehören die Titelseite mit Einverständniserklärung (`\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 präsentiert. Der Hauptteil sollte die folgenden Teile beinhalten:

- Einleitung

Tabelle 1.1: Zentrale switches

switch	Bedeutung
<code>latin1, utf8,</code> <code>[ansinew]</code> <code>[deutsch],</code> <code>english</code>	Zur direkten Eingabe von Sonderzeichen wie a, o, u und s muss dem Compiler die verwendete Kodierung mitgeteilt werden. Sprache der automatisch gesetzten Begriffe
<code>oneside,</code> <code>[twoside]</code>	Einstellung für doppelseitigen Druck. Beeinflusst die Kopfzeile und Leerseiten vor neuen Kapiteln.
<code>biblatex</code>	Verwendet <code>biblatex</code> statt <code>bibtex</code> . <code>biblatex</code> hat den Vorteil, dass es die UTF8-Kodierung unterstützt, die beispielsweise von <code>JabRef</code> verwendet wird. Damit können beispielsweise auch Referenzen von Autoren mit Umlauten im Namen korrekt sortiert werden. <code>biblatex</code> erfordert, dass in der verwendeten Umgebung <code>biber.exe</code> als Bibtex-Compiler eingestellt ist. Falls <code>biber</code> nicht in der installierten \TeX -Distribution enthalten ist, kann es nachtraglich von http://sourceforge.net/projects/biblatex-biber/files/biblatex-biber/ bezogen werden. Die Kompatibilitäten der verschiedenen <code>biblatex</code> und <code>biber</code> Versionen können der <code>biblatex</code> Dokumentation [2] entnommen werden.
<code>lst</code>	Sollte angegeben werden, wenn Quelltexte eingebunden werden. Mit dieser Option wird das <code>Listings</code> -Package geladen und es werden Styles mit Syntax-Highlighting für verschiedene Programmiersprachen (<code>C++</code> , <code>matlab</code> , <code>latex</code> , <code>other</code>) vordefiniert. Wird das <code>Listings</code> -Package nicht benötigt, empfiehlt es sich, diese Option wegzulassen, um die Compiliergeschwindigkeit zu erhöhen.
andere	Andere Optionen werden von der <code>scrbook</code> -Klasse behandelt.

- Zusammenfassung und Ausblick

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

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

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

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

Da die Vorlage, also `iiit-dipl.cls` *nicht* verändert werden soll, steht für eigene Erweiterungen wie zum Beispiel neue nützliche packages die Datei `erweiterungen.tex` zur Verfügung, die in die Hauptdatei eingebunden wird.

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

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

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

1.2 Allgemeine Hinweise

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

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

Sinnvolle Dateigrosen \LaTeX bietet die Möglichkeit, die Arbeit auf mehrere Dateien aufzuteilen und diese in die Hauptdatei einzufügen.

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 Hauptteil 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 detaillierte Informationen zu ResNets wird auf das Paper von Kaiming He et al. [1] 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 Inferenzrichtung - „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.

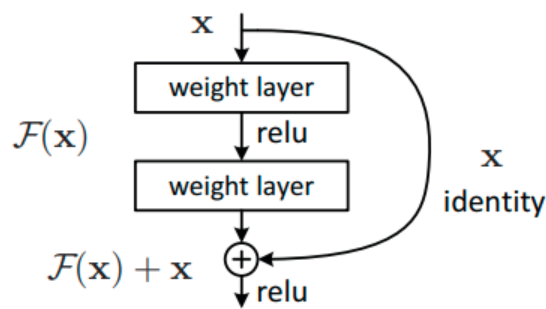


Abbildung 3.1: Residual Block (TODO → Ref)

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:

¹in ILSVRC (TODO → Ref)

²Raspberry Pi 4B 8GB. Betrachtet wurde die Laufzeit von einem Bild der Auflösung 224x224 und 3 (Farb-)Kanälen

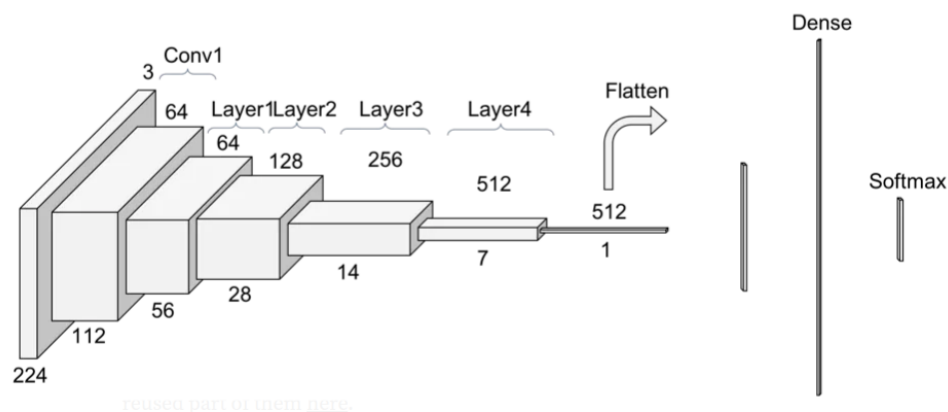


Abbildung 3.2: Pyramidale Merkmalsverteilung in ResNets (TODO → Ref)

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 Wesentlichen der gleiche Aufbau, allerdings sind die Kanäle „geweitet“ gegenüber den anderen beiden Architekturen, was sich an der Vervielfachung der Anzahl an Kanäle durch Verwendung eines komplexeren Residual Blocks („Bottleneck“ [3]) 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 reduziert („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 Pseudo-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

Der Raspberry Pi 4 ist ein äußerst vielseitiger und erschwinglicher Einplatinencomputer, der seit seinem Erscheinen immense Popularität erlangt hat. Mit seiner kompakten Größe und seinen beeindruckenden Fähigkeiten ist der Raspberry Pi 4 ein Favorit unter Bastlern, Pädagogen und Fachleuten gleichermaßen geworden.

Der Raspberry Pi 4, oft auch einfach als Pi 4 bezeichnet, ist die vierte Generation der Raspberry Pi-Serie. Er wurde im Juni 2019 von der Raspberry Pi Foundation vorgestellt und hat seitdem die Welt der Computerwelt verändert. Dieser scheckkartengroße Computer bietet eine bemerkenswerte Leistung in seinem kleinen Formfaktor, wodurch er für eine Vielzahl von Aufgaben geeignet ist.

Eines der herausragenden Merkmale des Raspberry Pi 4 ist seine verbesserte Hardwareausstattung im Vergleich zu seinen Vorgängern. Er wird von einer Quad-Core-ARM-Cortex-A72-CPU

angetrieben, was eine deutliche Verbesserung gegenüber den Vorgängermodellen darstellt. Diese CPU in Kombination mit wahlweise 2 GB, 4 GB oder 8 GB Arbeitsspeicher ermöglicht es dem Pi 4, auch anspruchsvollere Aufgaben mit Leichtigkeit zu bewältigen. Außerdem ist er mit einem VideoCore VI-Grafikprozessor ausgestattet, der eine flüssige Videowiedergabe und 3D-Grafikrendering ermöglicht.

Die Anschlussmöglichkeiten sind eine weitere Stärke des Raspberry Pi 4. Er ist mit zwei USB 3.0-Anschlüssen und zwei USB 2.0-Anschlüssen ausgestattet und ermöglicht so den Anschluss einer Vielzahl von Peripheriegeräten wie Tastaturen, Mäusen, Speichergeräten und Kameras. Darüber hinaus verfügt der Pi 4 über Dualband-Wi-Fi (2,4GHz und 5GHz) und Gigabit-Ethernet für schnelle und zuverlässige Netzwerkverbindungen. HDMI- und Audioausgänge bieten Unterstützung für hochauflösende Bildschirme und Audiogeräte.

Eine der wichtigsten Verbesserungen des Raspberry Pi 4 ist die Unterstützung für zwei 4K-Displays. Das bedeutet, dass Sie ihn für Aufgaben wie Digital Signage, Media Center und sogar als Desktop-Ersatz verwenden können. Die Unterstützung für zwei Monitore eröffnet neue Möglichkeiten für Multitasking und Produktivität.

Der Pi 4 ist für eine Vielzahl von Betriebssystemen ausgelegt, wobei Raspberry Pi OS (früher Raspbian) die offizielle Wahl ist. Diese benutzerfreundliche Linux-Distribution ist auf den Pi zugeschnitten und wird mit einer Fülle von vorinstallierter Software geliefert, was es zu einer hervorragenden Wahl für Anfänger macht. Sie können jedoch auch andere Linux-Distributionen und sogar Windows 10 auf dem Pi 4 ausführen, je nach Ihren Bedürfnissen.

Die GPIO-Pins (General Purpose Input/Output) des Raspberry Pi 4 ermöglichen Hardware-Schnittstellen und Prototyping. Diese Funktion ist besonders attraktiv für Bastler und Tüftler, die ihre eigenen elektronischen Projekte und Automatisierungslösungen erstellen möchten.

Neben dem Einsatz in der Ausbildung und im Hobbybereich hat der Raspberry Pi 4 auch seinen Weg in verschiedene professionelle Anwendungen gefunden. Er kann für Heimautomatisierung, Robotik, Webserver und als Plattform für die Entwicklung und das Testen von Software verwendet werden. Dank seines geringen Stromverbrauchs eignet er sich auch für eingebettete Systeme und IoT-Projekte (Internet der Dinge).

Zusammenfassend lässt sich sagen, dass der Raspberry Pi 4 ein bemerkenswertes Stück Technologie ist, das die Grenzen dessen, was mit einem Einplatinencomputer möglich ist, weiter verschiebt. Seine leistungsstarke Hardware, die vielfältigen Anschlussmöglichkeiten und die Unterstützung mehrerer Betriebssysteme machen ihn zu einem vielseitigen Werkzeug für eine breite Palette von Anwendungen. Egal, ob Sie Student, Bastler oder Profi sind, der Raspberry Pi 4 bietet eine erschwingliche und zugängliche Plattform zum Lernen, Experimentieren und für Innovationen.

Zweite

Version

Der Raspberry Pi 4, der im Juni 2019 von der Raspberry Pi Foundation veröffentlicht wur-

de, ist ein kleiner, erschwinglicher und vielseitiger Einplatinencomputer. Er hat aufgrund seines kompakten Formfaktors und seiner Fähigkeiten große Aufmerksamkeit erregt.

Zu den wichtigsten Hardwarekomponenten des Raspberry Pi 4 gehört eine Quad-Core-ARM-Cortex-A72-CPU, die eine erhebliche Verbesserung der Verarbeitungsleistung gegenüber den Vorgängermodellen darstellt. Er ist in drei Speicherkonfigurationen erhältlich: 2 GB, 4 GB und 8 GB RAM. Die Integration eines 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.

Der Raspberry Pi 4 bietet Unterstützung für zwei 4K-Displays und erweitert damit seine potenziellen Einsatzmöglichkeiten auf Digital Signage, Media Center und Multitasking-Szenarien.

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.

Die GPIO-Pins des Raspberry Pi 4 ermöglichen Hardware-Schnittstellen und Experimente, was ihn zu einem wertvollen Werkzeug für Maker und Entwickler macht.

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 eignet er sich für eingebettete Systeme und Internet-of-Things-Anwendungen (IoT).

Zusammenfassend lässt sich sagen, dass der Raspberry Pi 4 ein vielseitiger und kostengünstiger Einplatinencomputer ist, der für seine verbesserten Hardwarespezifikationen, Konnektivitätsoptionen und die Unterstützung für mehrere Betriebssysteme bekannt ist. Dank seines Potenzials zum Lernen, Experimentieren und für praktische Anwendungen erfreut er sich bei verschiedenen Nutzergruppen, von Studenten und Hobbyisten bis hin zu Profis, großer Beliebtheit.

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 Weiterentwicklung liegt vor allem in der Methode, wie die „Memory-Bank“ aufgebaut wird, die mit wenigen Beispielen nominaler „Patch Features“ eine möglichst hohe Repräsentativität 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 (Accuracy) 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 vortrainierten 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 Netzwerks, verwendet, sondern die Ausgabe bestimmter Dabe interessiert nicht das Klassifikationsergebnis an sich, sondern viel mehr die sogenannten „Feature Maps“ von Zwischenschichten (Intermediate 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] HE, KAIMING, ZHANG, XIANGYU, REN, SHAOQING und SUN, JIAN: *Deep Residual Learning for Image Recognition*, 2015.
- [2] LEHMAN, PHILIPP, KIME, PHILIP, BORUVKA, AUDREY und WRIGHT, JOSEPH: *The biblatex Package: Programmable Bibliographies and Citations*, 2.7a, Juli 2013.
- [3] ZAGORUYKO, SERGEY und KOMODAKIS, NIKOS: *Wide Residual Networks*, 2017.