h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES
fbmn
FACHBEREICH MATHEMATIK
UND NATURWISSENSCHAFTEN

h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES
fbi
FACHBEREICH INFORMATIK

Hochschule Darmstadt

Fachbereiche Mathematik und
Naturwissenschaften & Informatik

Anomalieerkennung und vorausschauende Warnung für z/OS-Systeme anhand von Log-Daten der IBM System Automation

Abschlussarbeit zur Erlangung des akademischen Grades

Master of Science (M. Sc.)
im Studiengang Data Science

vorgelegt von

Pavel Kravetskiy

Referent	:	Prof. Dr. Horst Zisgen
Korreferent	:	Dr. Marco Selig

Ausgabedatum	:	2. September 2019
Abgabedatum	:	16. Februar 2020

ERKLÄRUNG

Ich versichere hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die im Literaturverzeichnis angegebenen Quellen benutzt habe.

Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder noch nicht veröffentlichten Quellen entnommen sind, sind als solche kenntlich gemacht.

Die Zeichnungen oder Abbildungen in dieser Arbeit sind von mir selbst erstellt worden oder mit einem entsprechenden Quellennachweis versehen.

Diese Arbeit ist in gleicher oder ähnlicher Form noch bei keiner anderen Prüfungsbehörde eingereicht worden.

IBM, Schönaicher Str. 220, Böblingen, 16. Februar 2020

Pavel Kravetskiy

ABSTRACT

With the development of modern software systems, their monitoring, administration and maintenance are becoming increasingly difficult. Current approaches to system administration are based primarily on rules and guidelines that are shaped by experts in this area. Due to the increasing complexity and constantly changing environment, the process requires more and more time and is prone to errors. Accordingly, there is a need for an automatic and efficient system maintaining and monitoring solution based on the analysis of log data.

As part of this work, a software solution for unsupervised anomaly detection is developed based on textual log data from IBM System Automation for z/OS. The log data has information about the activities of software resources. An essential part of this work is the feature engineering, where the necessary characteristics of the data are worked out. Feature engineering for anomaly detection poses a challenge because the anomaly detection problem is unsupervised. The feature engineering results in two types of representation of the features, which are then examined for anomalies using existing data mining methods. Three different anomaly detection models are used. The first model is based on distance calculation, the second on Bayesian statistics and the third on an artificial neural network (LSTM). Using the methods, anomaly indicators will be shown and highlighted in the log.

Finally, the anomaly detection models and their results about the possibility to detect anomalies and interpretability will be discussed.

Keywords: Anomaly detection, Log analysis, Feature Engineering, Distance, Bayesian statistics, Long Short-Term Memory

ZUSAMMENFASSUNG

Mit der Entwicklung moderner Softwaresysteme werden ihre Überwachung, Verwaltung und Pflege zunehmend schwieriger. Aktuelle Ansätze zur Systemverwaltung beruhen vorwiegend auf Regeln und Richtlinien, die von Experten in diesem Bereich geprägt werden. Aufgrund der steigenden Komplexität und stets änderndes Umfelds verlangt der Prozess immer mehr Zeit und ist fehleranfällig. Dementsprechend besteht ein Bedarf an automatischer und effizienter Systemverwaltung, die auf der Analyse von Log-Daten basiert.

In Rahmen dieser Arbeit wird eine Softwarelösung für unüberwachte Anomalieerkennung anhand von textuellen Log-Daten der IBM System Automation für z/OS entwickelt. Die Log-Daten beinhalten die Informationen über Aktivitäten von Softwareressourcen. Ein wesentlicher Teil dieser Arbeit ist die Merkmalsextraktion, bei der die notwendigen Charakteristiken der Daten herausgearbeitet werden. Die Merkmalsextraktion für die Anomalieerkennung stellt eine Herausforderung dar, da das Anomalieerkennungsproblem nicht überwacht ist. Die Merkmalsextraktion resultiert in zwei Darstellungstype der Merkmale, die danach mittels existierenden Methoden des Data-Minings auf Anomalien untersucht werden. Es werden drei verschiedene Anomalieerkennungsmodelle eingesetzt. Das erste Modell basiert auf der Distanzberechnung, das zweite auf Bayes'scher Statistik und das dritte auf einem künstlichen neuronalen Netz (LSTM). Anhand der eingesetzten Methoden werden Anomalieindikatoren aufgezeigt und im Log hervorgehoben.

Abschließend werden die eingesetzten Anomalieerkennungsmodelle und ihre Ergebnisse bezüglich der Anomalieerkennung und der Interpretierbarkeit diskutiert.

Schlagwörter: Anomalieerkennung, Loganalyse, Merkmalsextraktion, Distanz, Bayes'sche Statistik, Long Short-Term Memory

INHALTSVERZEICHNIS

ABBILDUNGSVERZEICHNIS

TABELLENVERZEICHNIS

ABKÜRZUNGSVERZEICHNIS

z/OS Betriebssystem für IBM-Großrechner

SA System Automation für z/OS

IPL Initial Program Load

LSTM Long Short-Term Memory

RNN Rekurrentes neuronales Netz

Tf-idf Term frequency–inverse document frequency

RI Random Indexing

SVM Support Vector Machine

CNN Convolutional Neural Network

HDFS Hadoop Distributed File System

MLP Multilayer Perceptron

PCA Hauptkomponentenanalyse

1 EINLEITUNG

1.1 MOTIVATION

Mit dem Fortschritt in Wissenschaft und Technologie werden Computersysteme mit einer zunehmenden Vielfalt heterogener Software- und Hardwarekomponenten immer komplexer. Sie werden zunehmend schwieriger zu überwachen, zu verwalten und zu pflegen. Herkömmliche Ansätze zur Systemverwaltung basieren größtenteils auf Experten in diesem Bereich. Dabei wird das Bereichswissen in Regeln und Richtlinien umgesetzt. Der Prozess ist arbeitsintensiv und fehleranfällig. Darüber hinaus ist es kompliziert, mit dem sich schnell ändernden Umfeld Schritt zu halten. Es besteht daher ein dringender Bedarf an automatischen und effizienten Ansätzen zur Überwachung und Verwaltung komplexer Computersysteme. Ein beliebter Ansatz für die Systemverwaltung basiert auf der Analyse von Log-Daten mittels Analysetools, die sich entweder auf regelbasierten Verfahren oder auf Methoden des Data-Minings basieren.

IBM System Automation für z/OS (SA) ist ein Automatisierungsprodukt, das die hohe Verfügbarkeit von Betriebssystemen für IBM-Großrechner (z/OSs) durch Starten, Stoppen und Wiederherstellen der Ressourcen gewährleistet. In der Regel ist eine solche Ressource eine z/OS-Anwendung. SA fungiert als reaktive Softwarelösung, die permanent überprüft, ob sich jede z/OS-Ressource in ihrem Zielzustand (z. B. gestartet / gestoppt) befindet.

Ein z/OS-System ist allerdings hochkomplex, da die Ressourcen nicht als unabhängige Einheiten betrachtet werden können. Vielmehr ist jede Ressource Teil eines mehr oder weniger stark verzweigten Abhängigkeitsbaumes, so dass die Abhängigkeiten bei der Ressourcenverwaltung mitberücksichtigt werden müssen (z.B. Anwendung X kann nur dann gestartet werden, wenn Anwendung Y läuft). Eine manuelle Erkennung von Problemen, welche bei dermaßen komplexem System auftreten können, so wie deren mögliche Ursachen stellt einen großen Aufwand dar. In den seltenen Fällen, in denen das Verhalten einer Ressource von der Norm abweicht, spricht man von Anomalien. Grundsätzlich spiegelt sich ein solches anormales Verhalten im SA-Log wider, allerdings treten fallweise keine eindeutigen Fehlermeldungen, sondern vielmehr Statusmeldungen in atypischer Abfolge, auf.

1.2 ZIEL DER ARBEIT

Das Ziel dieser Masterarbeit ist die Entwicklung einer Softwarelösung, welche historische SA-Log-Daten zur Automatisierungsverarbeitung einer Ressource oder einer Benutzeraktion erfasst (im Speziellen den Datenstrom für sogenannte Work Items und Nachrichten vom Automation Manager, mit denen die Verarbeitung der Work Items abgeschlossen wird) und diese auf anomales Verhalten mittels Methoden des Data-Minings analysiert. Als erster Schritt gilt es die notwendigen Charakteristiken der Log-Einträge herauszuarbeiten. Bei dieser Merkmalsextraktion sollen verschiedene Darstellungen der Log-Einträge als Merkmal-Vektoren und Nachrichtensequenzen unter Beachtung der Kontextinformation untersucht werden. Darauf aufbauend werden existierende Methoden des Data-Minings angewendet, um Indikatoren verschiedener Anomalien aufzuzeigen und diese im Log zu hervorzuheben.

1.3 GLIEDERUNG

Im Kapitel 2 wird zuerst auf die Umgebung, Anwendungsgebiet und Konzepte der SA eingegangen. Zweitens wird ein grundlegendes Verständnis über das Data-Mining, insbesondere Anomalieerkennung und ihre Modelle, geschaffen. Schließlich wird über die verwandten Arbeiten im Bereich Loganalyse informiert.

Die Herkunft und Struktur der SA-Log-Daten wird im Kapitel 3 im Detail geschildert. Nachfolgend werden diese Daten mittels explorativer Datenanalyse untersucht und begutachtet.

Die Beschreibung der Implementierung findet in dem Kapitel 4 statt. Dies beinhaltet die durchgeführte Merkmalsextraktion, sowie die Beschreibung möglicher Merkmaltypen. Anschließend werden drei Modelle vorgestellt, die während der Arbeit eingesetzt wurden.

Die Diskussion der eingesetzten Modellen zum Anomalieerkennung für z/OS anhand von SA-Log-Daten findet in dem Kapitel 5 statt.

Das letzte Kapitel 6 beinhaltet eine abschließende Betrachtung und Zusammenfassung der Ergebnisse. Es wird geprüft, ob alle am Anfang gestellten Ziele erreicht wurden, sowie ein Ausblick für zukünftige Weiterentwicklungen beschrieben, welche sinnvoll sein könnte zum weiteren Funktionsausbau des SA und dessen Optimierung.

2 GRUNDLAGEN

In diesem Kapitel werden theoretische Grundlagen der Arbeit beschrieben. Zuerst wird die Umgebung und Anwendungsgebiet beschrieben. Zweitens wird das Konzept des Automation Managers und Automation Agents erklärt. Drittens wird der Begriff der Ressource eingeführt und erläutert. Außerdem werden Data-Mining und seiner Aufgaben beschrieben, insbesondere Anomalieerkennung und ihre Modelle. Schließlich wird über die verwandten Arbeiten im Bereich Loganalyse informiert.

2.1 UMGEBUNG

SA ist ein Systemverwaltungsprogramm mit einer zentralen Steuerung, mit dem alle wichtigen Systemverwaltungsfunktionen als ein einzelnes System-Image dargestellt werden.

Eine der Hauptfunktionen des Systemverwaltungsprogramms ist Überwachung aller verfügbaren Ressourcen, um auf bestimmte Ereignisse geeignet zu reagieren, bevor sie sich auf Endbenutzer auswirken. Folgende Ressourcen werden damit überwacht:

- Hardwarekomponenten
- Softwareprodukte und -anwendungen
- Automatisierte Prozesse
- Nachrichten und Warnungen

SA automatisiert die Verwaltung und Überwachung der Ressourcen und vieler komplexer, sich wiederholender Aufgaben. Einige SA-Funktionen umfassen insbesondere [23]:

- Starten und Stoppen von Software- und Hardwareressourcen bzw. dem gesamten Unternehmenssystem (Enterprise System)
- Unterstützung aller System z- oder zSeries-Prozessoren in einem Parallel Sysplex
- Verwaltung von mehreren heterogenen Betriebssystemen, sowie lokaler und rechnerferner Operationen
- Reagieren auf System- und Fehlermeldungen, sowie ungeplante Ereignisse
- Durchführung des Initial Program Load (IPL)
- Erstellung der Automatisierungsrichtlinie für Unternehmen

2.1.1 *Automation Manager und Automation Agents*

SA kann auch Anwendungen automatisch steuern, die über einen Sysplex verteilt sind, indem Systemgrenzen für die Automatisierung durch das Design von Automation Manager / Automation Agent virtuell aufgehoben werden. Außerdem wird die Komplexität der Verwaltung eines parallelen Sysplex durch die zielgerichtete Automatisierung und Konzepte wie Gruppierung und Unterstützung der komplexen Abhängigkeiten verringert.

In SA ist die Automatisierungsfunktion wie folgt aufgeteilt:

- Teile der SA, die für Überwachen, Reagieren und Handeln verantwortlich sind, werden als Automation Agents bezeichnet [3].

  Automation Agents üben folgende Tätigkeiten aus:

  - Wiederherstellungsarbeiten
  - Nachrichtenverarbeitung
  - Kontinuierliche Überwachung (Übermitteln von Statusänderungen an den Automation Manager)

- In jedem Sysplex sind Funktionen für Koordination, Entscheidungsfindung und Steuerung in einer Anwendung zusammengefasst. Dies wird als Automation Manager bezeichnet.

Der Hauptzweck des Automation Managers ist es, einen zentralen Punkt für die Verwaltung aller Ressourcen innerhalb eines Sysplex bereitzustellen, wofür der Automation Manager folgende vordefinierte und einstellbare Parameter wissen muss:

- Ressourcengruppierung
- Ressourcenabhängigkeiten
- Ressourcenstatus
- Ressourcenziele

Gemäß den verfügbaren Informationen trifft der Automation Manager Entscheidungen und weist die entsprechenden Automation Agents an, die Ressource in einen Zustand zu versetzen, der ihr Ziel besser erfüllt.

Die Automation Agents führen Aufträge aus, die vom Automation Manager stammen [3].

Abbildung 2.1 zeigt die Zusammenarbeit von einem Automation Manager und mehreren Automation Agents. Der Automation Manager enthält ein Modell aller automatisierten Ressourcen innerhalb des Sysplex. Die Automation Agents versorgen den Automation Manager mit Statusinformationen und führen die Aktionen aus, die der Automation Manager ihnen mitteilt. Automation Manager übernimmt die gesamte Entscheidungsfindung, bei der unter anderem die Interaktionen zwischen den Ressourcen berücksichtigt werden müssen.

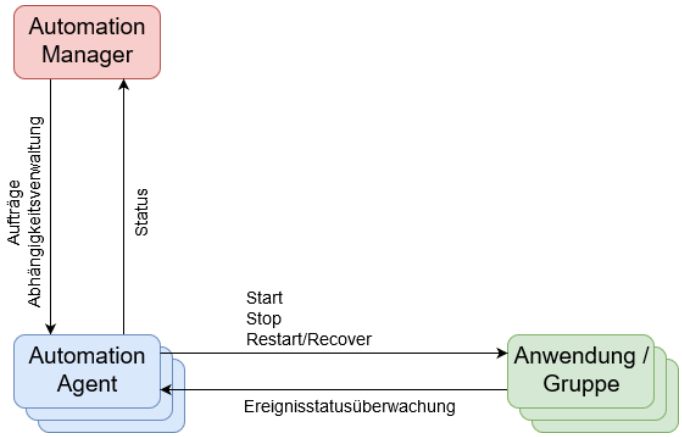


Abbildung 2.1: Automation Manager und Automation Agents

2.1.2 *Resourcen*

Moderne Anwendungen bestehen häufig aus vielen Komponenten, z. B. Datenservern, Netzwerk- und Sicherheitskomponenten. Beispiele sind Client und Server-Anwendungen, bei denen die Anwendungslogik zwischen einem Client und einem oder mehreren Servern verteilt ist und auch die Daten zwischen zwei oder mehreren Servern verteilt sind. Diese Komponenten laufen häufig auf die verschiedenen Systeme im Sysplex.

Ein allgemeiner Begriff für eine Anwendung, eine Anwendungsgruppe usw. ist Ressource. Ressourcen können dynamisch von einem System auf ein anderes System im Sysplex verschoben werden.

Anwendungsgruppen sind aktive Automatisierungselemente, die die Verfügbarkeit einer kleinen Menge von Ressourcen verwalten, wobei Abhängigkeiten einer Anwendungsgruppe von ihren Mitgliedern geerbt werden können.

In SA kann man eine vollständige Anwendungsgruppe automatisieren. Falls es definiert wird, dass alle Mitglieder einer Anwendungsgruppe gleichzeitig verfügbar sein müssen, wird das durch die Konfiguration der entsprechenden Anwendungsgruppe gewährleistet.

Auf Abbildung 2.2 ist ein Beispiel der Zusammenführung von Anwendungen zu Anwendungsgruppen dargestellt. Eine Anwendung kann zu mehreren Anwendungsgruppen gleichzeitig zugewiesen werden, wodurch große Flexibilität in der Konfiguration erreicht wird.

In der Regel besteht eine Anwendungsgruppe aus Ressourcen, die zum Ausführen einer komplexen Anwendung erforderlich sind. Dabei ist es möglich Untergruppen zu definieren, um die komplexen Ressourcen zu verwalten, die für die Ausführung erforderlich sind. Die Vorteile von Gruppen bestehen darin, dass sie die Verfügbarkeit der betreuten Ressourcen automatisch verwalten und einen zentralen Kontrollpunkt anbieten, über den die Aufforderungen zum Stoppen bzw. Starten von Ressourcen gesendet werden können.

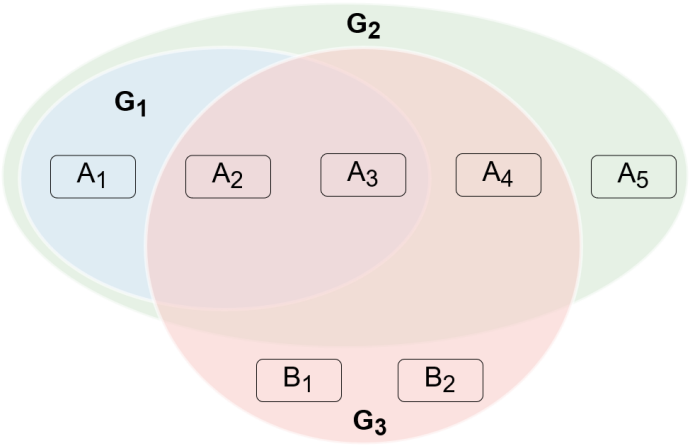


Abbildung 2.2: Beispiel für die Zusammenführung von Anwendungen A_i und B_i zu Anwendungsgruppen G_i.

Die komplexe Anwendungsgruppe G_2 umfasst die Anwendungsgruppe G_1 und die Einzelanwendungen A_4 und A_5. Die Anwendungsgruppe G_1 besteht wiederum aus drei Anwendungen A_1, A_2 und A_3. G_3 umfasst Anwendungen, die in anderen Anwendungsgruppen enthalten sind.

In SA stehen drei Arten von Anwendungsgruppen zur Verfügung: Basic, Move und Server [26]. Auf Abbildung 2.3 ist ein Beispiel der drei Arten veranschaulicht. Diese Gruppen unterscheiden sich in Mindest- bzw. Maximalanzahl der zugewiesenen Ressourcen, die verfügbar sein müssen.

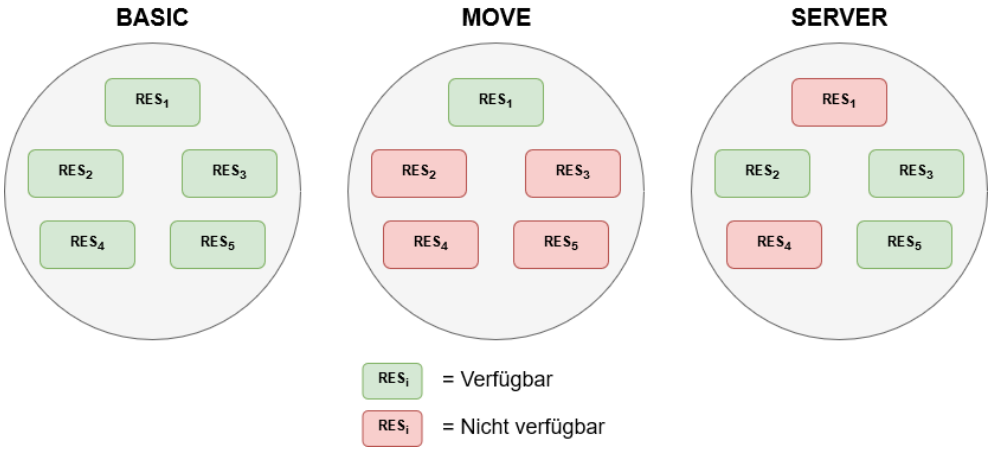


Abbildung 2.3: Arten von Anwendungsgruppen

Basic-Gruppen

SA verwaltet die Gruppen als wären sie eine einzelne Ressource. Falls man beispielsweise die Gruppe startet, werden alle ihre Ressourcen gestartet. Wenn eine Ressource der Gruppe ausfällt und nicht neugestartet werden kann, ist die gesamte Gruppe defekt, was unter Umständen zum Herunterfahren aller Ressourcen der Gruppe führt.

Move-Gruppen

Diese Gruppen verwalten die Verfügbarkeit einer einzelnen Ressource und wählen zwischen alternativen Instanzen auf verschiedenen Systemen, falls die Ressource ein unerwartetes Verhalten aufweist. Die Move-Gruppe ist dafür verantwortlich, das System auszuwählen, auf welchem die Ressource gestartet werden muss, um die Verfügbarkeit der Ressource aufrechtzuerhalten.

Server-Gruppen

Wie Move-Gruppen verwalten Server-Gruppen die Verfügbarkeit von Ressourcen über mehrere Systeme. Eine Server-Gruppe erhält hingegen eine Mindestanzahl an Ressourceninstanzen aufrecht. Kann die Mindestanzahl nicht gewährleistet werden, gerät die Gruppe in den Problemzustand.

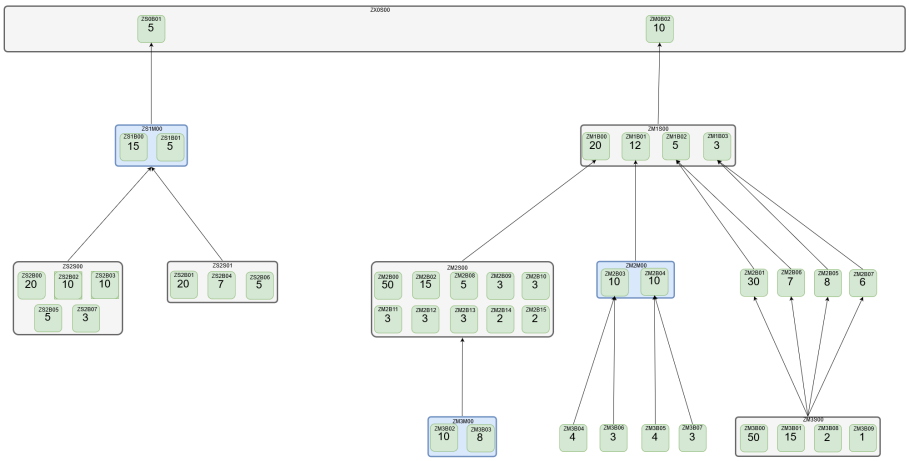


Abbildung 2.4: Realitätsnahes Beispiel von einem Ressourcenbaum

Im Ressourcenbaum werden unterschiedliche Gruppentypen farblich hervorgehoben. Basic-Gruppen sind in grün dargestellt. In blau sind die Move-Gruppen. Die Server-Gruppen sind entsprechend in grau.

Ein Beispiel von der Ressourcenstruktur eines komplexen Systems wird auf Abbildung 2.4 dargestellt. Die Zahlen innerhalb jeder Basic-Gruppe geben die Anzahl der Anwendungen an, die der entsprechenden Gruppe zugewiesen sind. Wie auf dem Bild zu sehen ist, haben einige Gruppen eine oder mehrere Abhängigkeiten von anderen Gruppen. Allerdings kommt in diesem Beispiel lediglich eine einfache Art der Abhängigkeit vor. In der Praxis können beliebige Abhängigkeiten zwischen Gruppen bzw. Applikationen definiert werden, die nicht unbedingt azyklisch sein müssen. Außerdem tendieren Ressourcenbäume üblicherweise nicht in die Tiefe, sondern in die Breite zu wachsen.

2.2 DATA-MINING

In Wissenschaft, Wirtschaft, Industrie und vielen anderen Bereichen ist aufgrund der Fortschritte in der Computerisierung und Digitalisierung eine riesige Menge Daten verfügbar. Solche Daten können eine reiche Quelle für Wissensentdeckung und Entscheidungsunterstützung darstellen. Um die riesige Datenmenge zu analysieren, zu verstehen und schließlich zu nutzen, wird ein multidisziplinärer Prozess, Data-Mining, umgesetzt. Der Begriff Data-Mining wurde 1996 von Fayyad, Piatetsky-Shapiro und Smyth geprägt und bedeutet eine automatische Auswertung großer Datenmengen zur Bestimmung von Regelmäßigkeiten, Gesetzmäßigkeiten und verborgener Zusammenhänge mithilfe der statistischen Methoden.

Die beiden Hauptziele des Data-Mining in der Praxis sind laut Fayyad et al. [5] in der Regel Vorhersage und Beschreibung. Die Vorhersage umfasst die Verwendung einiger Variablen in den Daten, um fehlende oder zukünftige Werte anderer Variablen von Interesse vorherzusagen. Die Beschreibung konzentriert sich darauf, vom Menschen interpretierbare Muster zu finden, die die Daten beschreiben. Die relative Bedeutung von Vorhersage und Beschreibung für bestimmte Data Mining-Anwendungen kann erheblich variieren. Die Ziele der Vorhersage und Beschreibung können mit einer Vielzahl bestimmter Data-Mining-Methoden erreicht werden.

Die Aufgaben des Data-Mining sind vielseitig, da große Datenmengen diverse Muster enthalten können. Um verschiedene Arten von Mustern zu finden, sind unterschiedliche Methoden und Techniken erforderlich. Basierend auf den gesuchten Mustern unterscheiden Fayyad et al. [5] folgende Aufgaben des Data-Minings:

- Anomalieerkennung

  In dieser Aufgabe werden Datenelemente gesucht, die inkonsistent zu dem Rest der Daten sind, beispielsweise indem ihre Merkmale ungewöhnliche Werte aufweisen oder von einem generellen Trend abweichen.

- Assoziationsanalyse

  Assoziationsanalyse ist die Entdeckung von Zusammengehörigkeit oder häufig auftretenden Strukturen in einem Datensatz.

- Clusteranalyse

  Clusteranalyse ist häufig eine beschreibende Aufgabe, bei der versucht wird, eine endliche Menge von Kategorien bzw. Clustern zu identifizieren, um die Daten zu charakterisieren. Die Cluster können sich gegenseitig ausschließen und vollständig sein oder aus einer umfassenderen Darstellung wie hierarchischen oder überlappenden Cluster bestehen. Beim Clustering sind Datenpunkte so gruppiert, dass die Ähnlichkeiten zwischen Clustern maximiert und innerhalb eines Clusters minimiert werden.

- Klassifikation

  Bei der Klassifikation wird eine Funktion erlernt, die ein Datenelement einer von mehreren vordefinierten Klassen zuordnet. Dadurch können zukünftige Datenelemente klassifiziert und für ein besseres Verständnis der Klassen genutzt werden.

- Regressionsanalyse

  Unter Regressionsanalyse versteht man das Lernen einer Funktion, die ein Datenelement einer reellwertigen Zielvariablen zuordnet, indem die statistischen Zusammenhänge zwischen Attributen des Datensatzes (Merkmale) und der Zielvariable modelliert werden.

- Zusammenfassung

  Die Zusammenfassung umfasst Methoden zum Auffinden einer kompakten Beschreibung für eine Teilmenge der Daten. Ein Datensatz wird zusammengefasst und abstrahiert, woraus ein kleinerer Satz resultiert, der einen allgemeinen Überblick über die Daten bietet und in der Regel verallgemeinerte Informationen enthält.

2.3 ANOMALIEERKENNUNG

Ein Ausreißer ist ein Datenpunkt, der sich erheblich von den übrigen Daten unterscheidet. Hawkins [8] definiert ein Ausreißer folgendermaßen:

> "An outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism."

Ausreißer werden in der Data-Mining- und statistischen Literatur auch als Anomalien bezeichnet. In den meisten Anwendungen werden die Daten durch einen oder mehrere generierende Prozesse erstellt. Ein ungewöhnliches Verhalten eines solchen Prozesses stellt eine Abweichung von der Norm dar und kann deshalb als Anomalie betrachtet werden. In textuellen Log-Daten können beispielsweise untypische, vertauschte oder fehlende Log-Einträge auf eine Anomalie hindeuten.

Eine Anomalie kann nützliche Informationen zu abnormalen Eigenschaften des Systems und auf die Prozesse einwirkenden Entitäten enthalten. Das Erkennen derartiger ungewöhnlicher Merkmale liefert nützliche anwendungsspezifische Erkenntnisse. Dies wird als Anomaliebeschreibung bezeichnet [19]. Anomalieerkennung eignet sich für vielfältige Anwendungsbereiche wie Einbrucherkennungssysteme, Kreditkartenbetrug, medizinische Diagnose, Strafverfolgung und viele Andere.

Die Ausgabe einer Anomalieerkennungsmethode kann einer von zwei Typen sein:

- Binäres Label

  Ein binäres Label ist eine Bezeichnung, die angibt, ob ein Datenpunkt eine Anomalie ist oder nicht.

- Anomalie-Score

  Es handelt sich hierbei um eine rationale Zahl zwischen 0 und 1, die den Grad der Anomalität jedes Datenpunktes quantifiziert. Diese Zahl kann auch verwendet werden, um die Datenpunkte in der Reihenfolge ihrer Anomalität einzuordnen.

Da der Anomalie-Score wie eine Wahrscheinlichkeit zwischen 0 und 1 liegt, wird er oft als solche betrachtet. Jedoch handelt es sich bei den meisten Anomalieerkennungsmodellen nicht um eine echte Wahrscheinlichkeit (weder im Bayes'schen noch im Frequenzsinn), sondern lediglich um eine Pseudo-Wahrscheinlichkeit, welche sich numerisch ähnlich verhält.

Es ist subjektiv, was eine „ausreichende" Abweichung darstellt, damit ein Datenpunkt als Anomalie betrachtet werden kann. Laut Aggarwal [1] können die Daten in realen Anwendungen nicht nur Anomalien, sondern auch eine erhebliche Menge an Rauschen beinhalten.

Knorr et al. [13] benennen Rauschen und Anomalien als schwache und starke Ausreißer. Für die Analyse und das Data-Mining sind allerdings lediglich die signifikant abweichenden Datenpunkten, Anomalien, von Interesse.

2.3.1 *Anomalieerkennungsmodelle*

Je nachdem, ob und inwiefern Anomalien im Datensatz bekannt sind, unterscheidet man generell die folgenden 3 Modelltypen:

- Überwachtes Modell

  Falls alle Anomalien bekannt sind, spricht man von einem gelabelten Datensatz. Die Labels können von einem überwachten Modell verwendet werden, um die semantischen Unterschiede zwischen normalen Datenpunkten, Rauschen und Anomalien zu ermitteln.

  Die überwachte Anomalieerkennung ist ein Sonderfall der Klassifizierung mit dem Unterschied, dass die relative Präsenz der Klassen extrem unausgewogen ist. Aufgrund dieser ungleichen Klassenverteilung tendieren klassische Klassifikationsmodelle dazu, in Richtung der größten Klasse voreingenommen zu sein [21]. Aus diesem Grund muss ein Anomalieerkennungsmodell mit der niedrigen Präsenz der Anomalien im Datensatz umgehen können. Laut Niu, Wang und Yang [18] sind überwachte Anomalieerkennungsmodelle in vielen anwendungsspezifischen Szenarien in der Regel effektiver als die unüberwachten.

- Unüberwachtes Modell

  Das Erstellen eines gelabelten Datensatzes kann kompliziert und zeitaufwendig sein. Aus diesem Grund verwendet man in im Fall fehlender Label unüberwachte Modelle. Solche Modelle werden häufig in explorativen Situationen eingesetzt, in denen die anwendungsspezifische Bedeutung der entdeckten Anomalien geprüft wird.

- Semi-überwachtes Modell

  In manchen Fällen sind einige Labels für Anomalien im Datensatz verfügbar, jedoch kann der Datensatz auch unbekannte Anomalien in einem bestimmten Verhältnis enthalten. Dies wird als Klassifizierung mit positiven und nicht gekennzeichneten Daten oder auch semi-überwachtes Modell bezeichnet.

Nach der Funktionsweise unterscheidet Aggarwal [1] unter anderem Extremwertanalyse, probabilistische und statistische, lineare und entfernungsbasierte Arten der Anomalieerkennungsmodelle.

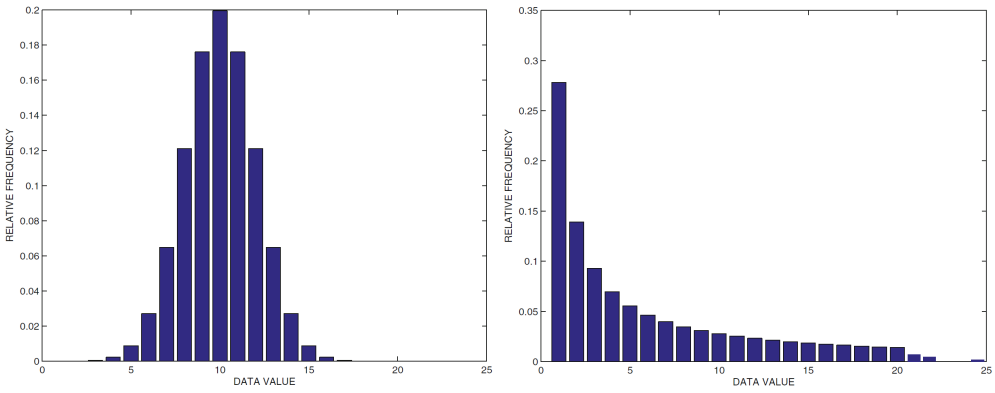
Extremwertanalyse

Die grundlegendste Form der Anomalieerkennung ist die Extremwertanalyse der Daten. Der entscheidende Punkt besteht darin, die statistischen Schwänze der zugrunde liegenden Verteilung zu bestimmen. Dies wird beispielsweise mithilfe vom Z-Test durchgeführt.

Eine implizite Annahme bei dem Z-Test ist, dass die Daten aus einer Normalverteilung stammen. Man betrachtet eine Reihe eindimensionaler Beobachtungen, die mit $X_1 ... X_N$ bezeichnet sind, mit Mittelwert μ und Standardabweichung σ. Der Z-Wert für den Datenpunkt X_i wird mit Z_i bezeichnet und ist wie folgt definiert:

$$Z_i = \frac{|X_i - \mu|}{\sigma} \tag{2.1}$$

Der Z-Wert-Test berechnet die Anzahl der Standardabweichungen, um die ein Datenpunkt vom Mittelwert entfernt ist. Man legt in der Regel einen Z-Wert (z.B. $Z > 3$) fest, ab dem der Datenpunkt als Anomalie betrachtet werden soll.



(a) Normalverteilung (b) Zipf-Verteilung

Abbildung 2.5: Anwenden eines Z-Tests auf die Normal- und Zipf-Verteilungen [1]

Wie in Abbildung 2.5 dargestellt, kann die Art der Schwänze in Abhängigkeit von der zugrunde liegenden Datenverteilung erheblich variieren.

Im ersten Fall (a) wird das Histogramm aus einer Normalverteilung und im zweiten Fall (b) aus einer Zipf-Verteilung abgetastet. Die meisten Daten für die Normalverteilung liegen im Bereich [4, 16], und alle andere Datenpunkte, die außerhalb dieses Bereichs liegen, können als Anomalien betrachtet werden. Da die Annahme des Z-Tests im Fall (a) erfüllt ist, funktioniert der Z-Test auf die Daten wie erwartet.

Im Fall (b) mit der Zipf-Verteilung sind die Anomalien nicht ganz klar, obwohl die Datenpunkte mit sehr hohen Werten als Anomalien betrachtet werden können. Infolgedessen deklariert der Z-Test keinen der Datenpunkte als anomal.

Die Performanz des Z-Tests auf der Zipf-Verteilung ist zumindest unter dem Gesichtspunkt der probabilistischen Interpretierbarkeit nicht aussagekräftig. Dies deutet darauf hin, dass Fehler in der Modellierungsphase zu einem falschen Verständnis der Daten führen können.

Die Methode ist also lediglich für spezielle Arten von Anomalien geeignet, bei denen davon ausgegangen wird, dass es sich bei den zu großen bzw. zu kleinen Werten um Anomalien handelt.

Probabilistische und statistische Modelle

In probabilistischen und statistischen Modellen werden die Daten in Form einer Wahrscheinlichkeitsverteilung modelliert und Parameter dieses Modells gelernt. Die Annahme hier ist die spezifische Wahl der Datenverteilung, mit der die Modellierung durchgeführt wird.

Ein wesentlicher Vorteil von diesen Modellen besteht darin, dass sie auf praktisch jeden Datentyp angewendet werden können. Wenn die Daten beispielsweise kategorial sind, kann eine diskrete Bernoulli-Verteilung verwendet werden, um die Wahrscheinlichkeitsverteilung zu modellieren.

Ein Nachteil von diesen Modellen ist, dass sie versuchen, die Daten an eine bestimmte Art von Verteilung anzupassen, was unter Umständen nicht angemessen sein kann. Darüber hinaus kommt es mit zunehmender Anzahl von Modellparametern häufig zu der Situation, dass das gelernte Modell so spezifisch ist, dass es die zufällige Streuung der Daten, aus denen die Modellparameter geschätzt werden, reproduziert. Diese Situation heißt Overfitting. In solchen Fällen können die unbekannten Anomalien nicht erkannt werden.

Lineare und nicht-lineare Modelle

Bei den linearen Modellen werden die Daten unter Verwendung linearer Korrelationen modelliert. Die optimale Hyperebene, die durch die Datenpunkte verläuft, wird mithilfe der Regressionsanalyse bestimmt. Die Abstände der Datenpunkte von dieser Hyperebene werden verwendet, um die Anomalie-Scores zu quantifizieren, da sie vom Modell normaler Daten abweichen.

Ein lineares Modell der 2-dimensionalen Datenpunkte $\{(x_i, y_i), i \in \{1...N\}\}$ mit Koeffizienten a und b kann wie folgt erzeugt werden:

$$y_i = a \cdot x_i + b + \epsilon_i \quad \forall i \in \{1...N\} \tag{2.2}$$

Hier stellt ϵ_i den Modellierungsfehler, Residuum, dar. Die Koeffizienten a und b müssen aus den Daten gelernt werden, indem die Summe der Fehlerquadrate $\sum_{i=1}^{N} \epsilon_i^2$ minimiert wird. Die quadratischen Residuen liefern die Anomalie-Scores. Mithilfe der Extremwertanalyse können danach die ungewöhnlich großen Abweichungen ermittelt und als Anomalien bezeichnet werden.

Das Konzept der Dimensionsreduktion und Hauptkomponentenanalyse ist ziemlich ähnlich, außer dass es einen nichtparametrischen Ansatz verwendet wird, um die Datenkorrelationen zu modellieren. Mithilfe von Hauptkomponentenanalyse wird ein Unterraum geringerer Dimension mit dem geringsten Rekonstruktionsfehler nach der Projektion bereitgestellt. Anomalien weisen große Rekonstruktionsfehler auf, da sie nicht den ausgewählten Subräumen in den Daten entsprechen. Daher können die Rekonstruktionsfehler als Anomalie-Score verwendet werden.

Dimensionsreduktion und Regressionsmodellierung sind in Bezug auf die ursprünglichen Attribute schwer zu interpretieren. Dies liegt daran, dass die Unterraumeinbettung als lineare Kombination von Attributen mit positiven oder negativen Koeffizienten definiert ist. Dies kann nicht einfach intuitiv in Bezug auf bestimmte Eigenschaften der Merkmale interpretiert werden.

Long Short-Term Memory (LSTM)

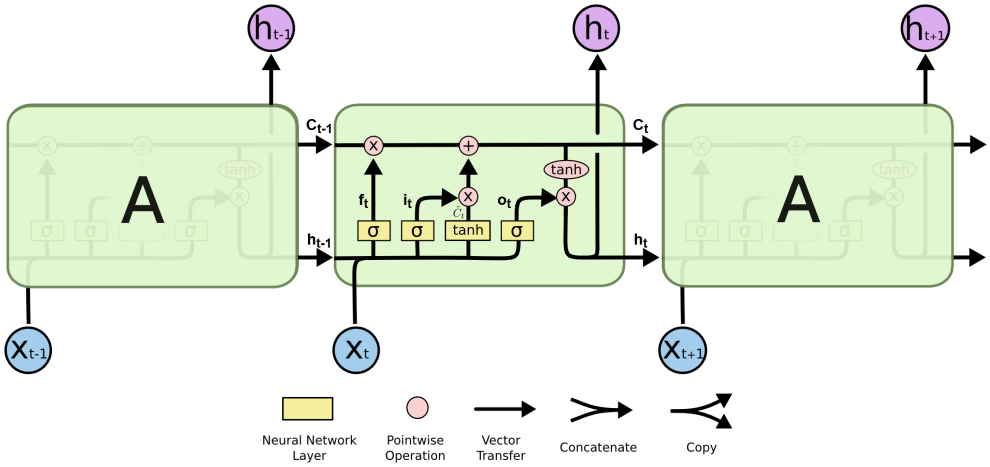


Abbildung 2.6: Aufbau eines LSTM-Netzes[20]. Die Abbildung wurde angepasst.

Von links nach rechts werden die Daten propagiert. C_{t-1} und C_t stellen den vorigen und neuen Zellstatus dar. h_{t-1} und h_t repräsentieren den vorigen und jetzigen versteckten Zellstatus. x_t ist der aktuelle Teil der Sequenz. f (Forget), i (Input), und o (Output) repräsentieren die Gates. $\tilde{C}_t$ sind mögliche Kandidaten für neuen Zellstatus C_t.

In den letzten Jahren hat sich Deep Learning als eine der beliebtesten Techniken des maschinellen Lernens gemustert und zu modernsten Ergebnissen für eine Reihe von überwachten und unüberwachten Aufgaben geführt. Der

Hauptgrund für den Erfolg von Deep Learning ist die Fähigkeit, hochdimensionale nicht-lineare Abbildungen zu erlernen. Diese Abbildungen werden automatisch aus Daten gelernt und benötigen daher nahezu kein Featureengineering oder bestehendes Bereichswissen.

LSTM wurde 1997 von Hochreiter et al. [9] erforscht und im Jahr 2000 von Gers et al. [7] erweitert, so dass LSTM-Netze eine der gängigsten Modelle geworden sind aufgrund ihrer Fähigkeit, Muster in sequentiellen und zeitlichen Daten zu lernen und sowohl lineare als auch nicht-lineare Abhängigkeiten heranzuziehen.

Abbildung 2.6 veranschaulicht den Aufbau eines LSTM-Netzes. Alle LSTMs haben die Form einer Kette sich wiederholender Zellen eines neuronalen Netzwerks. Jede Zelle enthält vier sogenannten Gates, die auf besondere Weise miteinander interagieren. Eine LSTM-Zelle nimmt als Input den vorigen Zellstatus C_{t-1}, den vorigen versteckten Zellstatus h_{t-1} und den Teil der Sequenz x_t entgegen. W_* sind die Gewichtungsmatrizen und b_* sind die Bias-Vektoren. Die Funktionsweise der LSTM-Zelle kann durch die folgenden Gleichungen dargestellt werden:

$$f_t = sigmoid(W_f \cdot [h_{t-1}, x_t] + b_f) \quad \text{(Forget Gate)} \tag{2.3}$$

$$i_t = sigmoid(W_i \cdot [h_{t-1}, x_t] + b_i) \quad \text{(Input Gate)} \tag{2.4}$$

$$\tilde{C}_t = tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \quad \text{(Neue Kandidaten)} \tag{2.5}$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad \text{(Neuer Zellstatus)} \tag{2.6}$$

$$o_t = sigmoid(W_o \cdot [h_{t-1}, x_t] + b_o) \quad \text{(Output Gate)} \tag{2.7}$$

$$h_t = o_t * tanh(C_t) \quad \text{(Neuer versteckter Zellstatus)} \tag{2.8}$$

Die Aktivierungsfunktionen *sigmoid* und *tanh* werden durch die entsprechenden Gleichungen 2.9 und 2.10 definiert

$$sigmoid(z) = \frac{1}{1 + e^{-z}} \tag{2.9}$$

$$tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{2.10}$$

Entfernungsbasierte Modelle

Die Idee bei entfernungsbasierte Modellen ist es, Anomalien als Punkte zu modellieren, die auf der Basis von Abstandsmaße von den normalen Daten isoliert sind. Beispiele von Abstandsmaße zwischen zwei Datenpunkten x und y, die für Anomalieerkennung angewendet werden können, sind durch die folgenden Gleichungen definiert:

$$D_{\text{Cityblock}}(x,y) = \sum_{i=1}^{n} |x_i - y_i| \qquad \text{(Cityblock-Distanz)} \qquad (2.11)$$

$$D_{\text{Euclidean}}(x,y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \qquad \text{(Euklidische Distanz)} \qquad (2.12)$$

$$D_{\text{Chebyshev}}(x,y) = \max_i(|x_i - y_i|) \qquad \text{(Chebyshev-Distanz)} \qquad (2.13)$$

Entfernungsbasierte Modelle können in eine von drei Arten unterteilt werden, nämlich Clustering-Modelle, dichtebasierte Modelle und Nächste-Nachbarn-Modelle. Bei Clustering- und anderen dichtebasierten Modellen werden die dichten Bereiche in den Daten direkt gefunden, und Anomalien werden als diejenigen Punkte definiert, die nicht in diesen dichten Bereichen liegen.

In Nächste-Nachbarn-Modellen wird die Entfernung jedes Datenpunkts zu seinem k-ten nächsten Nachbarn als Anomalie-Score angegeben [12]. Kleine Gruppen dichter Punkte, die weit vom verbleibenden Datensatz liegen, können durch Auswahl eines Wertes von $k > 1$ identifiziert und als Anomalien bezeichnet werden. Dieses Modell kann allerdings rechenintensiv sein, da es erforderlich ist, den k-ten nächsten Nachbarn jedes Datenpunkts zu bestimmen, was $O(N^2)$ Operationen für einen Datensatz erfordert, der N Datenpunkte enthält.

Bei Clustering-Modellen besteht der erste Schritt darin, einen Clustering-Algorithmus anzuwenden, um die dichten Bereiche des Datensatzes zu bestimmen. Im zweiten Schritt werden die Nichtzugehörigkeit eines Datenpunkts zu einem der Cluster, der Abstand zu anderen Clustern, die Größe des nächstgelegenen Clusters oder eine Kombination dieser Faktoren zur Quantifizierung des Anomalie-Scores verwendet. Im Beispiel von einem k-Means-Clustering wird der Abstand eines Datenpunkts zu seinem nächsten Schwerpunkt benutzt. Je höher ist der Abstand, umso stärker neigt der Datenpunkt anomal zu sein.

Dichtebasierte Modelle unterteilen den Datenraum in kleine Bereiche. Die Anzahl der Punkte in diesen Bereichen wird für die Ausrechnung der Anomalie-Scores verwendet. Der Unterschied zwischen Clustering- und dichtebasierten Modellen besteht darin, dass Clustering-Modelle die Datenpunkte partitionieren, wohingegen dichtebasierte Modelle den Datenraum partitionieren.

Alle diese Modelle sind eng miteinander verwandt, da sie auf einer Vorstellung von Nähe bzw. Ähnlichkeit beruhen, wobei der Unterschied in der Definition dieser Nähe liegt.

2.3.2 *Interpretierbarkeit und Modellauswahl*

Da es wünschenswert ist, zu bestimmen, warum ein bestimmter Datenpunkt als Anomalie betrachtet werden sollte, spielt die Interpretierbarkeit eines Anomalieerkennungsmodells eine wichtige Rolle. Verschiedene Modelle haben unterschiedliche Interpretationsmöglichkeiten. In der Regel weisen Modelle,

die mit den originellen Attributen arbeiten und weniger Transformationen für die Daten durchführen (z. B. Hauptkomponentenanalyse), eine höhere Interpretierbarkeit auf. Der Nachteil ist, dass Datentransformationen häufig den Kontrast zwischen den normalen und den anomalen Datenpunkten auf Kosten der Interpretierbarkeit verbessern. Daher ist es wichtig, diese Faktoren bei der Auswahl eines bestimmten Modells für die Anomalieerkennung zu berücksichtigen.

Aggarwal [1] definiert das Hauptprinzip der Anomalieerkennung und Selektion des Anomalieerkennungsmodells folgendermaßen:

> The core principle of discovering outliers is based on assumptions about the structure of the normal patterns in a given data set. Clearly, the choice of the "normal" model depends highly on the analyst's understanding of the natural data patterns in that particular domain.

2.4 VERWANDTE ARBEITEN IN LOGANALYSE

Dieses Kapitel soll einen Überblick über den aktuellen Forschungsstand über die Anomalieerkennung in Log-Daten geben. Die Anomalieerkennung war das Thema einer Reihe von Studien und wissenschaftlichen Arbeiten sowie Büchern.

Eine Art gängiger Ansätze für die Anomalieerkennung in Log-Daten sind statistische Modelle, die keine Lernphasen benötigen. Stattdessen verwenden sie regelbasierte und statische Analysemethoden für die Klassifizierung, z.B. die Hauptkomponentenanalyse. Xu et al. [27] haben eine allgemeine Methode vorgeschlagen, um Laufzeitprobleme des Systems basieren auf Konsolen-Log-Daten und dem Quellcode, der sie generiert hat, automatisch zu erkennen. Zum Schluss wurden ein PCA-basierende Anomalieerkennungsmodell angewendet, um jeden Featurevektor als normal oder abnormal zu kennzeichnen. Xu et al. haben festgestellt, dass dieses Anomalieerkennungsmodell mit beiden Merkmal-Vektoren sehr gut funktioniert.

Fronza et al. [6] haben einen Algorithmus zur Fehlervorhersage für Protokolldateien, die Support Vector Machine (SVM) und Random Indexing (RI) verwenden, entwickelt. RI wird angewendet, um Sequenzen darzustellen. Jede Operation wird in Bezug auf ihren Kontext charakterisiert. SVMs ordnen Sequenzen einer Klasse von Fehlern oder Nichtfehlern zu. Gewichtete SVMs werden angewendet, um unausgewogene Datensätze zu verarbeiten und die Richtig-positiv-Rate zu verbessern. Der Hauptvorteil der RI-Darstellung in diesem Ansatz besteht darin, dass keine Co-Occurrence-Matrix erstellt und keine Dimensionsreduktion durchgeführt werden muss, was zu einer signifikanten Beschleunigung des Verfahrens führt. Aus dem Ergebnis wurde geschlossen, dass der vorgestellte Ansatz bei der Vorhersage von Fehlern und Nichtfehlern sehr zuverlässig ist.

Zwietasch [30] hat RI- und Tf-idf-, sowie eine eigene Sliding-Window-basierte Darstellung der gelabelten Blue-Gene/L Syslog-Daten[1] verwendet. Die unterschiedlichen Vektorisierungen der Daten wurden danach mit verschiedenen existierenden Anomalieerkennungsmodellen, sowie einem eigenen Clustering-basierten Algorithmus ausgewertet. Laut der Auswertung wiese der entwickelte Algorithmus mit der Term frequency–inverse document frequency (Tf-idf)-Darstellung der Daten gegenüber den anderen Algorithmen eine bessere Performanz auf.

Peng et al. [22] haben Nachrichten in Syslog-Daten mithilfe vom überwachten maschinellen Lernverfahren kategorisiert. Hierfür wurden Syslog-Daten verschiedener Personal Computer gesammelt und alle Nachrichten manuell einer Kategorie eingeordnet. Diese Daten wurden danach mit dem Tool *Bow*[2], das von Carnegie Mellon University entwickelt wurde, vektorisiert. Anschließend haben die Forscher die Nachrichten mit einem modifizierten naiven Bayes-Klassifikator, der vorangegangene Kategorien in Be-

1 https://www.usenix.org/cfdr-data#hpc4
2 https://www.cs.cmu.edu/~mccallum/bow/

tracht nimmt, und dem Hidden Markov Model klassifiziert. Beide angewendeten Modelle haben eine Genauigkeit von mehr als 80% erreicht.

Ein flexibles Framework für Anomalieerkennung, besonders in hohen Dimensionen, namens DRAMA[3] wurde von Sadr et al.[24] entwickelt. Das Framework basiert auf Dimensionsreduktion und unüberwachtem Clustering. Im ersten Schritt werden Daten mit einem der fünf eingebauten Verfahren zur Dimensionsreduktion in einen niedrigeren dimensionalen Raum überführt. Im zweiten Schritt werden die wichtigsten Muster in den reduzierten Daten gefunden, indem ein agglomerativer hierarchischer Clusteringsalgorithmus angewendet wird. Nachdem die Muster isoliert sind, berechnet man die Distanz zwischen den Mustern und jedem Datenpunkt. Für jede Wahl der Distanzmetrik (Cityblock-, Chebyshev-, Mahalanobis-Distanz etc.) werden die vorhergesagten Anomalien mit $max_i\{min_j\{d(x_i, c_j)\}\}$ bewertet, wo x_i Datenpunkte und c_j Muster sind.

Lu et al. [15] haben einen neuen Ansatz zum Erkennen von Anomalien in Hadoop Distributed File System (HDFS)-Protokollen vorgeschlagen, der sich auf Convolutional Neural Network (CNN) und eigenen Worteinbettungen namens logkey2vec[25] basiert. Im Vergleich zu einem LSTM und einem Multilayer Perceptron (MLP) wies der Ansatz auf die gleiche Aufgabe die beste Genauigkeit auf.

Yang et al. [28] haben eine Methode vorgestellt, um den Aufwand für die Analyse der Log-Daten zu verringern, indem der wahrscheinlich nutzlose Text in der fehlgeschlagenen Log-Daten gelöscht wird. Mithilfe von mehrschichtiges Rekurrentes neuronales Netz (RNN) für Sprachmodelle auf Zeichenebene wurden die Log-Daten um bis zu 85% reduziert, währenddessen mehr als 77% der nützlichen Informationen intakt bleibt. Es wurde außerdem festgestellt, dass das Hinzufügen von Bereichswissen bei der kürzeren Sequenzlänge zur Verbesserung der Genauigkeit führt, jedoch konnten durch einfaches Erhöhen der Sequenzlänge noch bessere Genauigkeiten erzielt werden.

In Bezug auf diese Arbeit können die obengenannten Ansätze und Frameworks ohne Weiteres nicht angewendet werden, da die SA-Log-Daten eine hierarchische Struktur haben, die im Kapitel detaillierter 3.2 analysiert wird. Außerdem sind die SA-Log-Daten nicht gelabelt, was die Performanzvalidierung vieler verwendeten Ansätze erschwert.

3 https://github.com/vafaei-ar/drama

3 DATEN

3.1 HERKUNFT DER DATEN

Das Data-Mining und Datenanalyse sollen in der Regel auf echten Kundendaten, die von reellen Kundenszenarien herkommen, durchgeführt werden. Die Datenergebung stellt allerdings nicht nur eine rechtliche und zeitliche Herausforderung dar, sondern auch die meisten Unternehmen nicht bereit sind, den Zugriff auf ihre Daten zu einem beliebigen Zweck freizugeben.

Aus diesem Grund wurde entschieden, die textuellen Log-Daten für diese Arbeit zu verwenden, die durch zahlreiche und vielfältige Regression-Tests generiert wurden. Diese Regression-Tests verfolgen das Ziel, das komplette Funktionalität von SA auszutesten, indem sowohl die täglichen, realitätsnahe Kundenszenarien als auch das Systemverhalten bei ungewöhnlichen Situationen simuliert werden. Zu den Szenarien gehören beispielsweise:

- Starten / Stoppen einer Ressource, die bestimmte Abhängigkeiten zu den bereits gestarteten / gestoppten Ressourcen hat
- Verschieben einer Ressource auf ein anderes System
- Änderung von Automatisierungsrichtlinie bzw. -parameter und ihre Auswirkung auf bereits laufende Ressourcen
- Einspielen neuer Softwareversion durch sequentielles Stoppen und anschließendes Neustarten (Rolling recycle)

```
***** Scenario 13 - C05: APLE does not start because it is not flagged for the active RUNMODE
TC144: Verify AVAILABLE status of R#C05L*/APL/AOCA          OK
TC145: Execute PIPE NETV INGRUN REQ=SET SYSTEM=AOCA RUNMODE OK
TC146: Verify AVAILABLE status of R#C05L*/APL/AOCA          OK
TC147: Verify SOFTDOWN status of R#C05LE/APL/AOCA           OK
TC148: Execute INGWHY R#C05LE/APL/AOCA OUTMODE=LINE         OK
TC149: Verify INGWHY response for R#C05LE/APL/AOCA          OK
TC150: INGVARS GET R#C05LE/APL/AOCA INGWHY_*                OK
TC151: Compare INGVARS results                              OK
TC152: Verify AVAILABLE status of R#C05L*/APL/AOCA          NOK
     Expected resources: 5
     Found resources   : 4
```

Abbildung 3.1: Beispiel eines Testszenarios mit einem fehlgeschlagenen Testfall

Die Nutzung der durch Regression-Tests generierten Log-Daten hat den Vorteil, dass es zu den eigentlichen Log-Daten auch die entsprechenden Testergebnisse, wie auf Abbildung 3.1, verfügbar sind, die auf Erfolg bzw. Misserfolg einzelner Testfälle verweisen. Diese Information kann verwendet werden, um Anomalien in den Daten zu lokalisieren und zu labeln. Allerdings deuten die fehlgeschlagenen Testfälle nicht eindeutig darauf hin, dass es sich

hierbei um eine Anomalie handelt, sondern dienen lediglich als Hinweise auf ein möglich anomales Verhalten des Systems bzw. der Ressource. Man kann außerdem nicht sicherstellen, dass die erfolgreichen Tests anomaliefrei sind. Folglich handelt es sich in dieser Arbeit in erster Linie um ungelabelte Daten. Dank Durchsicht der fehlgeschlagenen Tests mit einem Experten, konnten dennoch 11 Anomalien herausgearbeitet werden. Diese Anomalien werden im Kapitel 4.2 als Beispiele herangezogen.

3.2 DATENSTRUKTUR

```
Sysplex = AOCAPLEX
Workitem History (descending)                                                                                    5 Jun 2019 09:59:31
-------------------------------------------------------------------------------------------------------------------------------------

2019-06-05 09:59:29 Variable attribute/user/INGWHY_REASON manipulated by operator AUTRGT1
    HSAL6181I Variable attribute/user/INGWHY_REASON set to 00000000; ASIS                                  (R#A12LA/APL/AOCA)

2019-06-05 09:59:29 Variable attribute/user/INGWHY_SITUATION manipulated by operator AUTRGT1
    HSAL6181I Variable attribute/user/INGWHY_SITUATION set to 00000000; SOFTDOWN                           (R#A12LA/APL/AOCA)

2019-06-05 09:59:28 Termination processing for R#ABTST/APL/AOCC completed - ABENDED, CRITICAL THRESHOLD EXCEEDED
    HSAL6269I Status/Automation is Idle                                                                    (R#ABTST/APL/AOCC)
    HSAL6348I Group Observer Update Requested                                                              (R#ABTST/APL/AOCC)
    HSAL6427I Group requires evaluation                                                                       (AOCC/SYG/AOCC)
    HSAL6427I Group requires evaluation                                                                         (SYSPLEX/GRP)
    HSAL6276I Status/Compound is Problem                                                                   (R#ABTST/APL/AOCC)
    HSAL6477I Resource Seen Active cleared                                                                 (R#ABTST/APL/AOCC)
    HSAL6172I Group Observer update sent                                                                   (R#ABTST/APL/AOCC)
    HSAL6172I Group Observer update sent                                                                      (AOCC/SYG/AOCC)

2019-06-05 09:59:27 Agent status for R#ABTST/APL/AOCC = BROKEN
    HSAL6259I Status/Observed is HardDown                                                                  (R#ABTST/APL/AOCC)
    HSAL6271I Status/Automation is Busy                                                                    (R#ABTST/APL/AOCC)
    HSAL6348I Group Observer Update Requested                                                              (R#ABTST/APL/AOCC)
    HSAL6119I Resource is not startable                                                                    (R#ABTST/APL/AOCC)
    HSAL6337I Resource is not Viable                                                                       (R#ABTST/APL/AOCC)
    HSAL6253I Status/Startable is No                                                                       (R#ABTST/APL/AOCC)
    HSAL6135I Resource prestart cannot be run                                                              (R#ABTST/APL/AOCC)
    HSAL6127I Resource cannot be started                                                                   (R#ABTST/APL/AOCC)
    HSAL6427I Group requires evaluation                                                                       (AOCC/SYG/AOCC)
    HSAL6427I Group requires evaluation                                                                         (SYSPLEX/GRP)
    HSAL6172I Group Observer update sent                                                                   (R#ABTST/APL/AOCC)
    HSAL6172I Group Observer update sent                                                                      (AOCC/SYG/AOCC)
    HSAL6264I Status/Observed is Problem                                                                      (AOCC/SYG/AOCC)
    HSAL6348I Group Observer Update Requested                                                                 (AOCC/SYG/AOCC)
    HSAL6252I Status/Startable is Yes                                                                         (AOCC/SYG/AOCC)
    HSAL6134I Resource prestart can be run                                                                    (AOCC/SYG/AOCC)
    HSAL6276I Status/Compound is Problem                                                                      (AOCC/SYG/AOCC)
    HSAL6172I Group Observer update sent                                                                      (AOCC/SYG/AOCC)
```

Abbildung 3.2: Aufbaubeispiel einer Log-Datei

Grau hervorgehoben ist ein Workitem. Grün hervorgehoben ist Zeitpunkt der Auffassung jeweiliges Workitems. Gelb hervorgehoben ist eine oder mehrere Ressourcen, auf die sich das Workitem bezieht. Orange hervorgehoben sind HSAL-Nachrichten, die jeweils an eine Ressource gebunden sind.

Die SA Log-Daten eines z/OS Systems weisen eine deutlich hierarchische Struktur auf, als gewöhnliche Log-Daten. Abbildung 3.2 zeigt ein Aufbaubeispiel. Ein Eintrag, der mit einem Zeitstempel beginnt, stellt ein sogenanntes Workitem dar. Jedes Workitem bezieht sich auf eine bestimmte Aktion, die durch das Workitem-Header definiert ist. Workitems beinhalten zeitlich geordnete Sequenzen von verschiedenen Nachrichten, beginnend mit HSAL. Jede HSAL-Nachricht kommt von einem Automation Agent und informiert über eine Aktivität oder Statusänderung einer Ressource. Jeder Ressourcenname folgt die Namenskonvention "*Name/Typ/System*", es sei denn, die Ressource gehört dem Sysplex. HSAL-Nachrichten können unter Umständen auch variable Informationen beinhalten z.B. einen neuen Wert für eine globale Variable. Workitems erscheinen im Log beginnend mit dem neuesten, die HSAL-Nachrichten in einem Workitem hingegen beginnend mit dem ältesten.

3.3 EXPLORATIVE DATENALALYSE

Im Rahmen dieser Arbeit wurden über 9,6 Millionen Zeilen Log-Daten aufgesammelt. Um den ersten Eindruck über die Datenmenge zu bekommen, werden in diesem Kapitel einfache Statistiken und Eigenschaften der Daten umrissen.

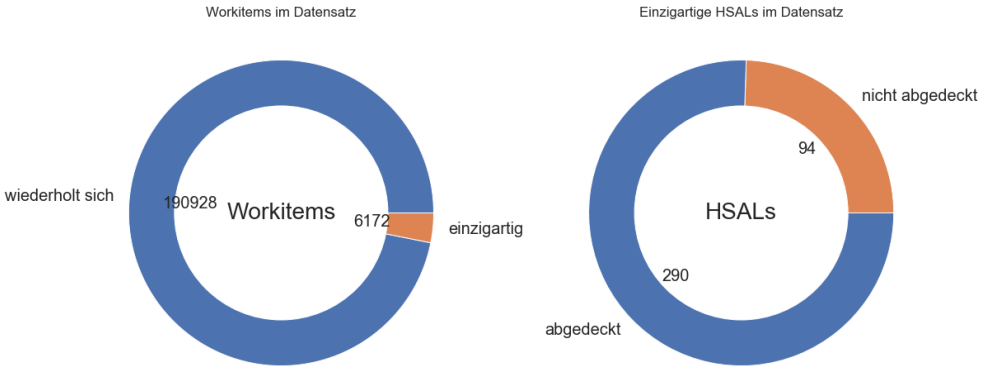


Abbildung 3.3: Anzahl der Workitems und HSAL-Nachrichten im Datensatz

Auf Abbildung 3.3 sieht man die Anzahl der Workitems. Die meisten Workitems kommen mindestens einmal vor, dennoch gibt es diejenige, die nur einzeln auftreten. Von insgesamt 384 HSAL-Nachrichten, die von SA ausgegeben werden können, tauchen 290 HSAL-Nachrichten im Datensatz auf. Die 94 HSAL-Nachrichten, welche im Datensatz nicht abgedeckt sind, verweisen auf schwere Ausnahmen oder finden nur in sehr speziellen Szenarien Anwendung.

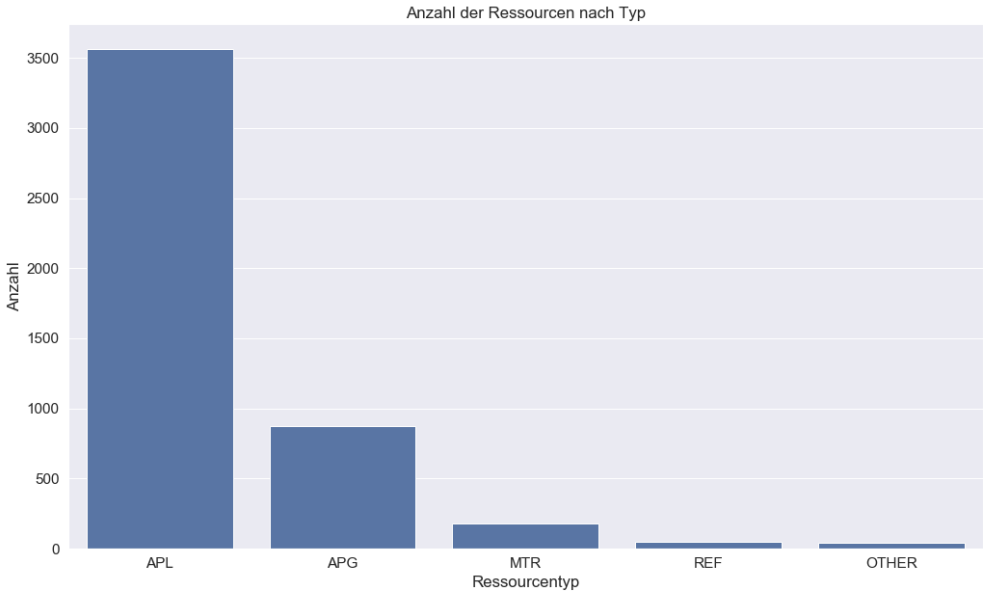


Abbildung 3.4: Anzahl der Ressourcen im Datensatz nach Typ

Abbildung 3.4 veranschaulicht die Verteilung der Ressourcentypen. Den Hauptanteil aller Ressourcen stellen über 3500 individuelle Anwendungen dar, die in ca. 900 Basic-, Move- und Server-Gruppen angeordnet sind. Viel seltener treten Monitore, Referenzen und andere systemspezifische Ressourcentypen auf.

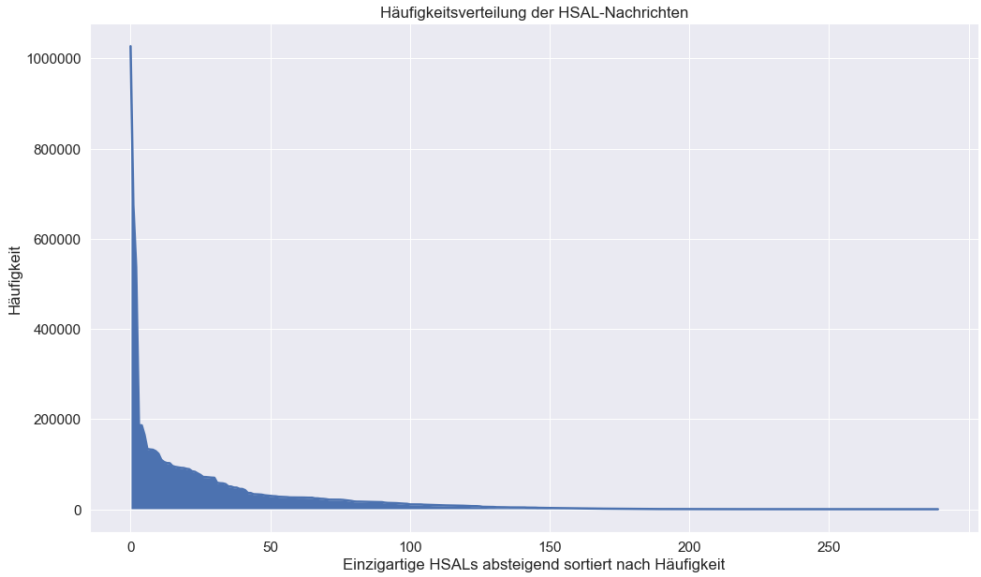


Abbildung 3.5: Häufigkeitsverteilung der HSAL-Nachrichten

Die Häufigkeitsverteilung der HSAL-Nachrichten ist auf Abbildung 3.5 veranschaulicht. Auf der x-Achse befinden sich HSAL-Nachrichten, die nach ihre absolute Häufigkeit absteigend geordnet sind. Anhand der Abbildung kann man feststellen, dass es 4 Nachrichten gibt, die sehr oft auftreten. Ca. 200 der Nachrichten kommen hingegen selten vor. Es stellt sich für die weitere Analyse die Frage, ob die zu oft und zu selten auftretenden HSAL-Nachrichten für die Anomalieerkennung relevant sind.

Um die Sequenzen der HSAL-Nachrichten zu veranschaulichen, wurde ein gerichteter Graph erstellt (siehe Abbildung 3.6). Ein Knoten repräsentiert dabei eine bestimmte HSAL und eine Kante repräsentiert den sequenziellen Übergang. Farblich sind verschiedene Knotentypen hervorgehoben. Knotengröße korrespondiert mit der logarithmierten absoluten Häufigkeiten der entsprechenden HSAL-Nachricht im Datensatz. Die 4 häufigsten Knoten in der Mitte des Graphen kommen in vielen Sequenzen als Mittel- bzw. mehrdeutige Knoten vor.

Obwohl man erwarten könnte, dass der Graph aus mehreren Teilgraphen bestehen wird, die verschiedenen Workitemtypen entsprechen, enthält dieser Graph lediglich 3 kleine Teilgraphen und einen großen stark vernetzten Teilgraph. Eine weitere Analyse hat ergeben, dass die Knoten im Graph über ca. 2800 Kanten verbunden sind, was lediglich 3% aller theoretisch möglichen Kanten ausmacht und der Anzahl der existierenden Nachrichtenpaare entspricht. Viele Schlingen, die auf dem Graph auch zu sehen sind, wurden durch wiederholenden HSAL-Nachrichten in einer Sequenz erzeugt.

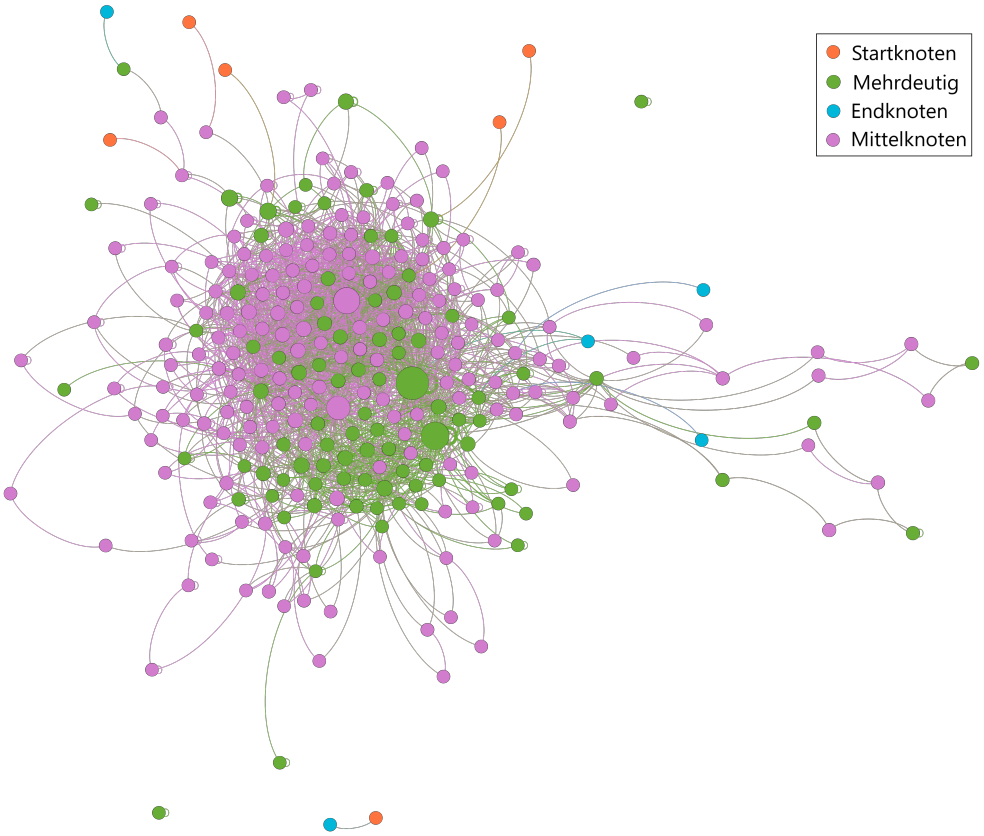


Abbildung 3.6: Graph aus HSAL-Nachrichten

Start-, Mittel- und Endknoten sind Nachrichten, die stets an der entsprechenden Position in einer Sequenz sich befinden können. Mehrdeutige Knoten treten in verschiedenen Positionen auf. Knotengröße entspricht der logarithmirten Häufigkeit im Datensatz. Selbstschleifen sind ebenfalls dargestellt.

4 IMPLEMENTIERUNG

Im Rahmen dieser Arbeit wurde eine Softwarelösung für SA-Log-Parsing, Visualisierung und einstellbare Merkmalsextraktion implementiert. Dafür wurde Python[1] als Programmiersprache, Jupyter[2] als interaktive Visualisierungsumgebung und Gephi[3] für Visualisierung von Graphdaten eingesetzt.

4.1 MERKMALSEXTRAKTION

Das Ziel der Merkmalsextraktion ist es eine optimale Vektorisierung der textuellen Log-Daten zu gewinnen, wobei nicht nur die HSAL-Nachrichten selbst, sondern auch deren Reihenfolgen als Merkmale in Betracht gezogen werden.

Anders als bei der Klassifizierung, bei der Labels als Richtungshinweise verwendet werden können, ist es kompliziert festzustellen, wie sich Merkmale auf die (nicht beobachtete) Grundwahrheit bei der unüberwachten Anomalieerkennung beziehen. In anderen Worten, die Merkmalsextraktion aus einem ungelabelten Datensatz ist deutlich schwieriger durchzuführen.

Im Rahmen dieser Arbeit wurden grundsätzlich zwei Arten der Merkmalsdarstellungen verwendet:

1. HSAL-Sequenzen

   Da die HSAL-Nachrichten in Log-Daten eine sequenzielle Natur haben und zeitlich angeordnet erscheinen, können sie als Sequenzen von Übergängen dargestellt werden.

```
2019-11-05 02:33:27 Startup for R#GRP039/APL/AOCA completed
  HSAL6269I Status/Automation is Idle                          (R#GRP039/APL/AOCA)
  HSAL6348I Group Observer Update Requested                    (R#GRP039/APL/AOCA)
  HSAL6427I Group requires evaluation                              (AOCA/SYG/AOCA)
  HSAL6427I Group requires evaluation                               (R#GRP039/APG)
  HSAL6427I Group requires evaluation                                (SYSPLEX/GRP)
  HSAL6282I Status/Compound is Satisfactory                    (R#GRP039/APL/AOCA)
  HSAL6172I Group Observer update sent                         (R#GRP039/APL/AOCA)
  HSAL6172I Group Observer update sent                             (AOCA/SYG/AOCA)
  HSAL6172I Group Observer update sent                              (R#GRP039/APG)
```

Abbildung 4.1: Beispiel einer HSAL-Sequenz

   Bei dieser Darstellung wird die Annahme getroffen, dass die Sequenzen innerhalb eines Workitems zeitliche Abhängigkeiten beinhalten. Ein Beispiel einer solchen HSAL-Sequenz im SA-Log ist Abbildung 4.1 angeführt. Auffällig ist dabei, dass die HSAL-Sequenz wiederholende

1 https://www.python.org/
2 https://jupyter.org/
3 https://gephi.org/

Nachrichten beinhaltet, die sich allerdings auf unterschiedliche Ressourcen beziehen. Diese Merkmalsdarstellung von HSAL-Sequenzen kann bei den Algorithmen verwendet werden, die mit sequenziellen Daten arbeiten, wie z.B. ein LSTM.

2. Merkmal-Vektoren

   Die andere Art der Darstellung basiert sich auf der Vektorisierung von Workitems. Jedes Workitem wird als Merkmal-Vektor repräsentiert und soll alle relevanten Informationen des Workitems enthalten. Dazu gehören beispielsweise Workitemtyp, die betroffenen Ressourcen und ihre Abhängigkeiten, sowie variable Informationen aus den HSAL-Nachrichten. Eine Sammlung von Merkmal-Vektoren für verschiedene Workitems bildet eine Designmatrix. Jede Zeile der Matrix ist ein Merkmal-Vektor (d.h. ein Workitem), und jede Spalte repräsentiert die Werte eines gegebenen Merkmals über alle Workitems hinweg. Bei der Darstellung wird angenommen, dass sowohl die einzelnen Merkmale als auch Workitems jeweils voneinander unabhängig sind.

Workitem Headers

Wie im Kapitel 3.2 beschrieben wurde, wird jedes Workitem mit einem Header versehen. Ein Workitem Header spiegelt die generelle Aktion (Task) wider, die innerhalb dieses Workitem ausgeführt wird. Die Headers können Informationen über die betroffene Ressource, ausgeführtes Kommando, geänderten Variablenname etc. enthalten. Um die Headervorlagen zu extrahieren, wurden reguläre Ausdrücke benutzt.

	Header	Anzahl
0	Agent status for <RES> DOWN	29431
1	Agent status for <RES> UP	23503
2	No PRESTART commands to issue	14103
3	Agent status for <RES> STARTED	12078
4	Startup for <RES> completed	12046
5	Startup for <RES> in progress	11959
6	No <CMD> commands to issue	10518
7	Agent status for <RES> AUTOTERM	9863
8	Shutdown in progress for <RES>	9671
9	Termination processing for <RES> completed	9593

Tabelle 4.1: Die 10 häufigsten Workitem Headers (Workitemtypen)

Ressourcennamen und Kommandos sind durch die entsprechenden Platzhalter <RES> und <CMD> ersetzt.

In Tabelle 4.1 sind die 10 häufigsten Workitem Headers dargestellt. Es geht hierbei um Starten bzw. Stoppen einer Ressource, Statusaktualisierung einer Ressource von Automation Agents etc. Headers, die im Datensatz relativ selten auftreten, wurden zu einem gemeinsamen Header zusammengeführt, so dass es letztlich in 27 Headers resultiert hat.

Im Laufe der Datenanalyse wurde festgestellt, dass die Workitems mit diversen Headers sich dermaßen unterscheiden, dass sie gegebenenfalls disjunkte Teilmengen der HSAL-Nachrichten beinhalten. Es wäre daher nicht praktikabel, die Workitem Headers als Merkmal mitzuführen. Stattdessen werden die Headers als Kriterium für eine Vorsortierung und separable Analyse benutzt.

HSAL-Nachrichten

Im Datensatz befinden sich insgesamt 290 verschiedene HSAL-Nachrichten. Wie erwähnt wurde, kommt ein gewisser Teil aller HSALs selten vor, allerdings können sie für die Anomalieerkennung charakteristisch sein. Da die Nachrichten in einem Workitem sequentiell erscheinen, kann ihre Reihenfolge auch eine Rolle spielen. Aus diesem Grund wurden nicht nur die einzelnen HSALs, sondern auch Paare und Tripel aus HSAL-Sequenzen extrahiert.

Der Nachteil dieses Vorgehens ist, dass die potenzielle Anzahl der Merkmale explodiert, wodurch ein Anomalieerkennungsmodell unter dem sogenannten Fluch der Dimensionalität leiden kann. Dieser Begriff wurde von Richard Bellman [4] geprägt und besagt, dass die Wahrscheinlichkeit, dass Datenobjekte am Rand des Datenraumes liegen, steigt mit der Anzahl der Dimensionen exponentiell. Um dieses Problem umzugehen, wurden folgende Selektionen vorgenommen:

- **HSAL-Singles**

  Um die charakteristischen HSALs zu selektieren, wurden die Nachrichten in 4 Gruppen aufgeteilt: Start-, End-, Mittel- und Einzelnachrichten.

  Auf Abbildung 4.2 sind die Häufigkeitsverteilungen von Nachrichten nach Gruppen veranschaulicht, die für Merkmalselektion verwendet wurden. Bei den Einzelnachrichten handelt es sich um die HSAL-Sequenzen, die aus einer Nachricht bestehen. Man sieht, dass es 2 dominante Nachrichten gibt, die deutlich häufiger einzeln auftreten als die anderen. Alle Sequenzen starten überwiegend mit einer der 6 Nachrichten, wobei die restlichen treten deutlich seltener auf. Häufigkeiten der HSALs, die weder Start- noch Endnachrichten sind, sieht bis auf ein paar Nachrichten nicht so überwiegend aus. Bei den Endnachrichten hingegen gibt es eine, die fast immer eine Sequenz abschließt.

  Anhand der Häufigkeitsverteilungen von Nachrichten in jeweiligen Gruppen wurden ca. 170 HSAL-Singles ausgewählt. Die selten auftretenden HSALs wurden in jeweilige allgemeine Merkmale (`OTHER#SGL`, `OTHER#SRC`, `OTHER#MDL` und `OTHER#TGT`) zusammengeführt.

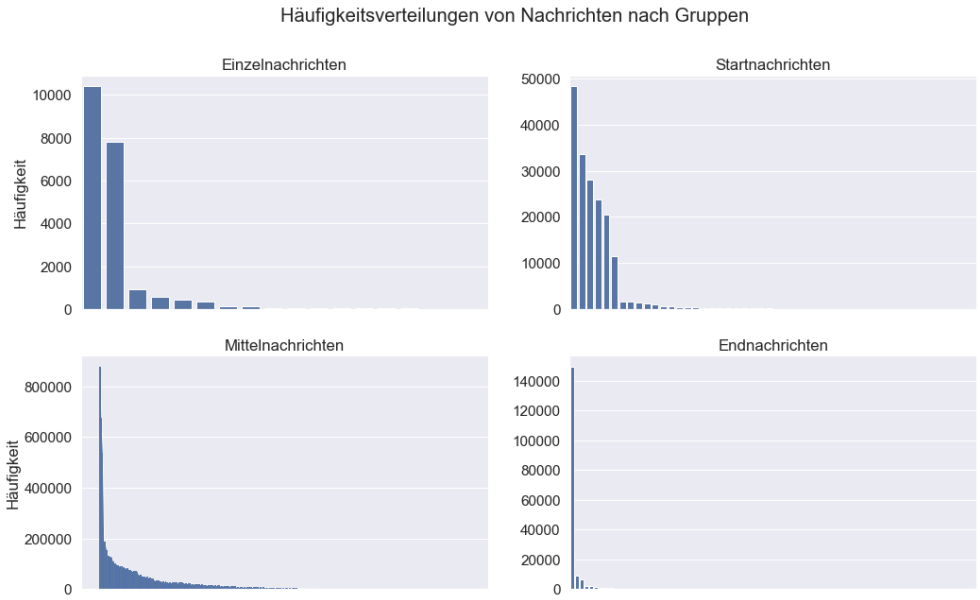


Abbildung 4.2: Häufigkeitsverteilungen von Nachrichten nach Gruppen

- **HSAL-Paare**

  Um die paarweisen Kollokationen in den HSAL-Sequenzen abzubilden, wurden aus den Sequenzen HSAL-Paare extrahiert, die den direkt verbundenen Knoten im Graph 3.6 entsprechen.

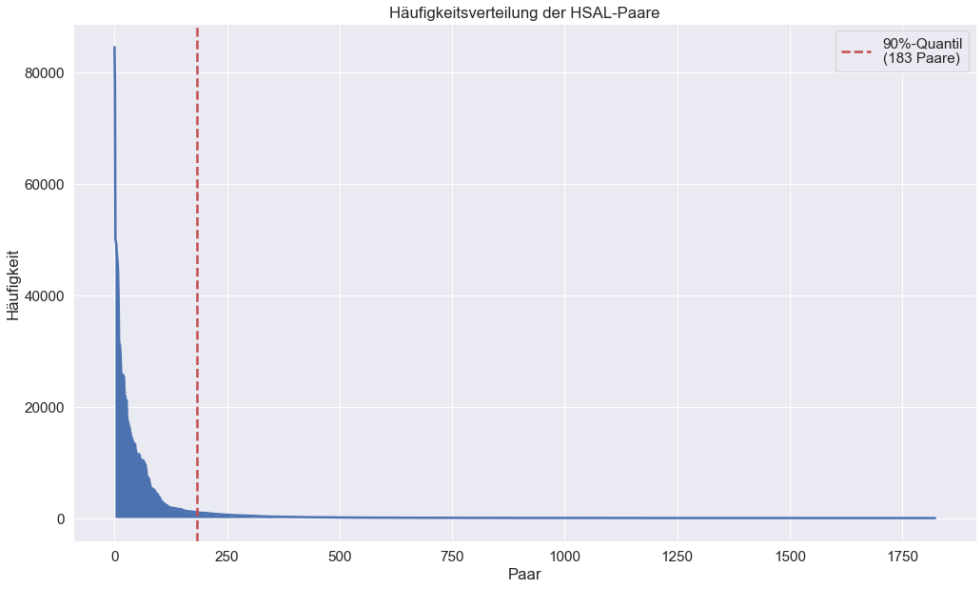


Abbildung 4.3: Häufigkeitsverteilung der HSAL-Paare

Die Abbildung 4.3 zeigt die Häufigkeitsverteilung der extrahierten HSAL-Paare. Die Verteilung ist extrem rechtsschief und deutet darauf hin, dass die meisten Paare äußerst selten auftreten und deshalb mithilfe eines Quantilwertes vernachlässigt werden können. In dieser Arbeit wurde Paa-

re, die unter dem 90%-Quantilwert liegen, aussortiert. Die Entscheidungsgrenze ist auf Abbildung 4.3 mit der roten gestrichelten Linie gezogen.

- **HSAL-Tripel**

  Ein weiteres Merkmal, welches HSAL-Teilsequenzen repräsentiert, ist HSAL-Tripel. In dieser Arbeit wurde beschlossen, Tripels in Form von $a \longrightarrow b \longrightarrow c$, wobei $a \neq b \neq c$ ist, zu extrahieren. Schleifenförmige Tripels ($a \longrightarrow b \longrightarrow a$) wurden vernachlässigt, da sie durch HSAL-Paare abgedeckt werden.

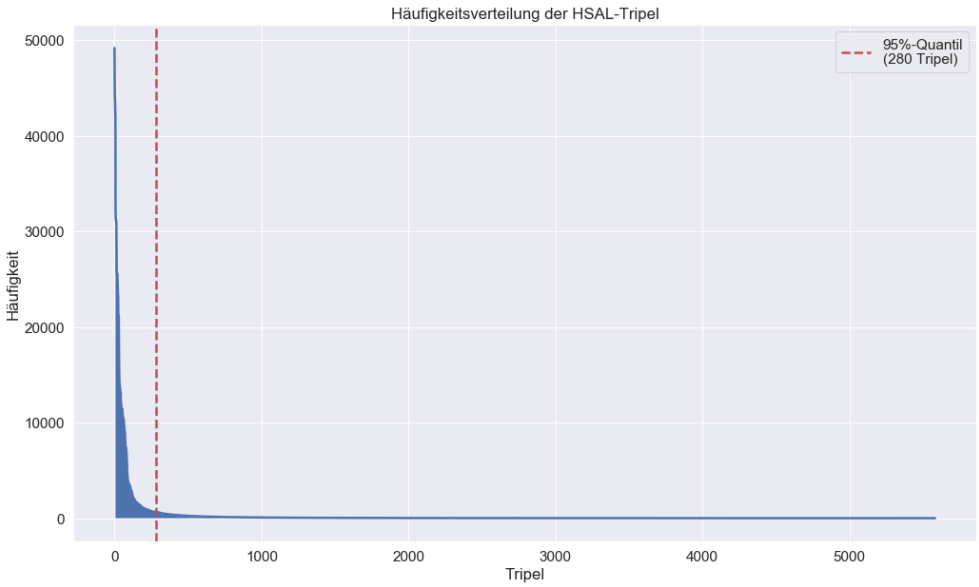


Abbildung 4.4: Häufigkeitsverteilung der HSAL-Tripel

Analog zu HSAL-Paare wurde die Häufigkeitsverteilung der HSAL-Tripel analysiert und anhand dem 95%-Quantilwert, um 280 von ca. 6000 verfügbaren Tripel für die Anomalieerkennung auszuwählen, wie es auf Abbildung 4.4 zu sehen ist.

$$\text{Tf} = \frac{\text{Häufigkeit des Merkmals in diesem Workitem}}{\text{Gesamtzahl der Merkmale in diesem Wokritem}} \tag{4.1}$$

$$\text{idf} = log \frac{\text{Gesamtzahl der Workitems im Datensatz}}{\text{Anzahl der Workitems, die dieses Merkmal enthalten} + 1} \tag{4.2}$$

$$\text{Tf-idf} = \text{Tf} \times \text{idf} \tag{4.3}$$

Die für diese Arbeit implementierte Software bietet die Möglichkeit an, den Quantilwert für die HSAL-Merkmale zu konfigurieren, so dass die Menge der HSAL-Merkmale sich variieren lässt.

Aus den selektierten HSAL-Singles, -Paare und -Tripels wurden Zählvektoren erstellt und mit dem Tf-idf-Maß gewichtet (siehe Gleichung 4.3), wel-

ches laut zahlreichen Studien und Fachliteratur [10, 16, 17, 30] zu einer besseren Performanz von Data-Mining-Aufgaben führt.

Zum Schluss wurden alle selektierten HSAL-Merkmale in die Designmatrix verkettet und die stark korrelierenden (mit über 95% der Pearson-Korrelation) entfernt, womit die Gesamtzahl der Merkmale sich auf knapp 400 reduzieren ließ.

Die Anzahl der HSAL-Merkmale lassen sich wesentlich reduzieren, falls man sich auf einen Workitemtyp einschränkt.

Variable Information

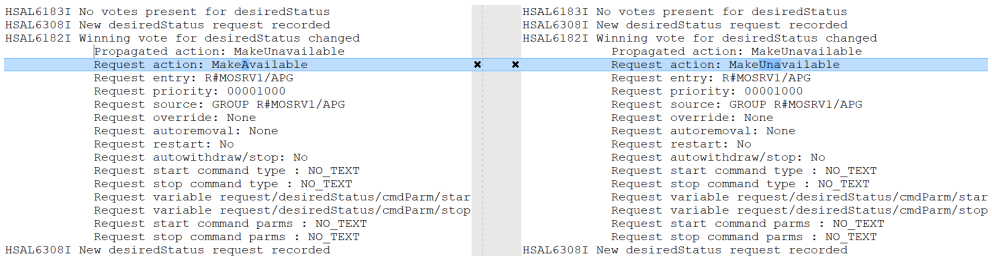


Abbildung 4.5: Beispiel variabler Information

Auf Abbildung 4.5 kann man zwei scheinbar identischen Nachrichten betrachten, die verschiedene Aktionen propagieren. Vektorisierung solcher Variablen bietet die Möglichkeit an, die Nachrichten effektiver zu differenzieren. Beispielsweise wurde die betrachtete HSAL-Nachricht in `HSAL6182I#MkAv` und `HSAL6182I#MkUn` manuell aufgeteilt. HSAL-Paare sowie HSAL-Tripel wurden im Nachhinein entsprechend angepasst.

Ressourcentypen

Da der betroffenen Ressourcen für die Anomalieerkennung eine Rolle spielen können, wurden die im Kapitel 3.2 erwähnten Ressourcentypen mithilfe ihrer Namenskonvention aus den Log-Daten extrahiert. Um die Wichtigkeit dieses Merkmaltyps für die Vektorisierung zu prüfen, wurde die logistische Regression angewendet, die die Ressourcentypen anhand den bereits extrahierten HSAL-Merkmale vorhersagt.

Tabelle 4.2 zeigt den entstandenen Klassifizierungsbericht. Wie man sieht, können Anwendungen sowie Gruppen auffallend gut vorhergesagt werden. Ressourcentypen, die relativ selten auftreten, nämlich Monitore und Referenzen weise etwas niedrige Ergebnisse auf.

Man kann also daraus schließen, dass der Ressourcentyp für die Vektorisierung keinen signifikanten Beitrag leistet, da er sich durch andere Merkmale abdecken lässt. Aus diesem Grund wurde dieser Merkmaltyp vernachlässigt.

Ressource	Precision	Recall	F1-Maß
APL	0.98	0.98	0.98
APG	0.96	0.73	0.83
MTR	0.98	0.61	0.75
REF	0.54	0.90	0.67
OTHER	0.99	0.71	0.83

Tabelle 4.2: Klassifizierungsbericht der Ressourcentypen

Abhängigkeiten von Ressourcen im Ressourcenbaum

Wie im Kapitel 2.1.2 besprochen wurde, gehört jede Anwendung zu mindestens einer Gruppe, die das Verhalten dieser Anwendung steuert. Je nach Typ der Gruppe und ihre Abhängigkeiten entstehen verschiedene Verhaltensmuster. Bestimmte Eigenschaften einer Ressource im Ressourcenbaum können für die Anomalieerkennung eine signifikante Rolle spielen. Zu den Eigenschaften gehören beispielsweise Anzahl der Kinder bzw. Eltern, Typ der Eltern, Anzahl der Ressourcen in derselben Gruppe, Abhängigkeitstyp.

Nach der Analyse von vorhandenen Log-Daten und einem Versuch daraus den Ressourcenbaum wiederherzustellen, wurde festgestellt, dass der Ressourcenbaum sowie Gruppentype sich mit den Log-Daten und ohne Zugriff zum laufenden System nicht zuverlässig extrahieren lassen.

4.2 MODELLBILDUNG

In diesem Kapitel werden die eingesetzten Modelle für die Anomalieerkennung in SA-Log-Daten sowie deren Ergebnisse dargestellt. Das erste Modell verwendet nur Merkmal-Vektoren, das zweite und dritte Modellen hingegen die HSAL-Sequenzen.

Wie im Kapitel 4.1 erläutert wurde, wird in dieser Arbeit mit den nach dem Workitem Header vorsortierten Daten gearbeitet. Das heißt, dass jeder Workitemtyp separat analysiert wird. Um die Auswertung der Analyseergebnisse zu erleichtern, wurden die Workitemtypen ausgewählt, in denen die herausgearbeiteten Anomalien auftauchen.

4.2.1 *Metrisches Modell*

Das erste Modell basiert auf der Merkmalsextraktion und Anwendung eines Distanzmaßes, um die Entfernung zwischen den Merkmal-Vektoren und ihrem Median als Entscheidungskriterium für einen Anomalie-Score zu verwenden. Bei diesem Ansatz wird die Annahme getroffen, dass Anomalien weit entfernt von normalen Datenpunkten liegen. Die Auswahl des Medians als denjenigen Datenpunkt, zu welchem die Entfernungen gemessen werden, beruht auf der statistischen Eingenschaft des Medians, weniger von Ausreißer (Anomalien) beeinflusst zu werden als z.B. der Mittelpunkt.

Um einen ersten Eindruck von den vektorisierten Daten zu gewinnen, wird der Feature-Vektorraum des Datensatzes mithilfe der Hauptkomponentenanalyse, die über alle Workitems durchgeführt wurde, in eine kleinere Anzahl an Dimensionen projizieren, wodurch auch versucht wird, normale und anomale Datenpunkte übersichtlich gegenüberzustellen.

Abbildung 4.6: Streudiagramm der ersten beiden Hauptkomponenten (3 Workitemtypen)

Für eine zweidimensionale Darstellung wurden verschiedene Kombinationen der Hauptkomponenten analysiert, von denen schon die ersten beiden ca. 50% der Gesamtvarianz erklären. Abbildung 4.6 stellt ein Streudiagramm der ersten beiden Hauptkomponenten dreier unterschiedlicher Workitemtypen dar. Jeder Punkt in dem Diagramm entspricht einem Datenpunkt, das heißt einer vollständigen, einzigartigen Sequenz von HSAL-Nachrichten.

Es ist unbekannt, ob Daten des Workitemtyps "No commands to issue" Anomalien enthalten. Man kann allerdings behaupten, dass der Workitemtyp scheinbar anomaliefrei ist, da die blauen Datenpunkte dicht beieinander liegen und keiner der Punkte signifikant von der Gesamtstreuung abweicht.

In dem Workitemtyp "MAKEAVAILABLE cancelled" wurde zwar eine Anomalie (mit dem orangen Kreuz gekennzeichnet) lokalisiert, aber sie scheint nicht so weit außerhalb der Gesamtstreuung der Datenpunkte zu sein. Mithilfe der Hauptkomponentenanalyse kann man also zu zwei Schussfolgerungen kommen: entweder sind das in Wirklichkeit keine Anomalien, oder die Hauptkomponentenanalyse ist nicht in der Lage, diese Anomalien von den anderen Datenpunkten scheinbar zu trennen, oder zwei Hauptkomponenten reichen dafür nicht aus.

Anomalien des Workitemtyps "Group/Member policies updated" (mit dem grünen Kreuz gekennzeichnet) liegen zu den anderen Datenpunkten am weitesten, was den Anomalieverdacht bestätigt. Man kann allerdings noch einen Datenpunkt sehen, der sehr nah zu den Anomalien liegt. Mit einer gewissen Wahrscheinlichkeit lässt sich behaupten, dass es sich hierbei um eine unerkannte Anomalie handelt.

Distanzmaße

Um die Distanzen zwischen den Punkten im Datensatz zu messen, wurden die drei folgenden Distanzmaße verglichen: euklidische Distanz, Cityblock-Distanz und Chebyshev-Distanz.

Vergleich verschiedener Vektorisierungen des Workitemtyps "Group/Member policies updated", die durch verstellbare Quantilwerte für Paare und Tripels erzeugt wurden, und der ausgewählten Distanzmaßen ist in Abbildung 4.7 veranschaulicht. Bei dieser Darstellung wird davon ausgegangen, dass Anomalien wesentlich größere Distanzen zum Median im Vergleich zu den anderen Datenpunkten aufweisen. Jeder Graph besteht aus einzigartigen Datenpunkten, deren Anzahl sich je nach Quantilwert variiert. Mit farblicher Skala ist die Summe aller Distanzen entsprechendes Datenpunktes hervorgehoben. Das Kamada-Kawai-Layout [11] sorgt dafür, dass die Punkten, die größere Distanzsummen haben, weiter von den Punkten mit einer relativ niedrige Distanzsumme liegen. Leichte Drehungen werden je nach Vektorisierung durch das Kamada-Kawai-Layout erzeugt. Neben den Distanzmaßen und Quantilwerten kann man auch die Anzahl der damit extrahierten Merkmale sehen.

Die Cityblock-Distanz schlägt zu der tatsächlichen Anomalie noch mehr Kandidaten vor als die anderen Distanzmaße.

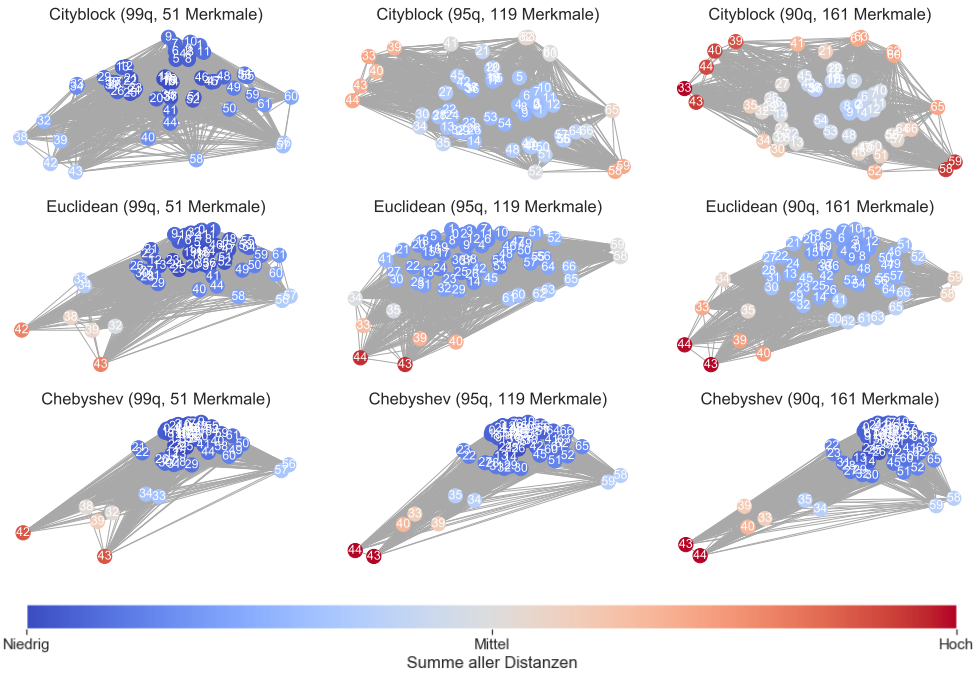


Abbildung 4.7: Vergleich von Distanzmaßen und Vektorisierungen des Workitemtyps "Group/Member policies updated"

Tatsächliche Anomalien sind 43, 44 und 43.

Die euklidische und die Chebyshev-Distanz liefern vergleichbare Ergebnisse. Die beiden Distanzmaße trennen mit zunehmender Anzahl an Merkmalen vor allem zwei Datenpunkte von der Gesamtmenge ab. Als Unterschied zeigt sich, dass die Chebyshev-Distanz größere Werte zurückliefert (siehe Gleichungen 2.12 und 2.13), deshalb unterscheiden die Datenpunkte mit einer großen Distanz stärker von den anderen. Im genaueren Vergleich der euklidischen sowie Chebyshev-Distanz mit über 100 Merkmale zu Abbildung 4.6 zeigt sich, dass eine Erhöhung der Anzahl der Merkmale zu keiner wesentlichen Änderung des Graphen führt.

Bei der Darstellung im Kamada-Kawai-Layout muss man allerdings berücksichtigen, dass die Anzahl der Merkmale je nach Quantilwert sich ändert, womit auch die Distanzen zwischen den Datenpunkten steigen können.

Skalierung

Um einen Anomalie-Score, der sich als Pseudo-Wahrscheinlichkeit interpretieren lässt aus den Distanzmaßen zu erhalten, ist eine Skalierung ins Intervall [0,1] notwendig. Die Skalierung wird hier exemplarisch für Chebyshev-Distanzen durchgeführt. Dies wäre für die anderen Distanzmaße analog.

Abbildung 4.8 zeigt das Histogramm der Chebyshev-Distanzen zum Median. Man kann feststellen, dass es relativ wenig Datenpunkte mit einer Distanz größer als 30 gibt. Die Distanzen sind also nicht gleichmäßig in ihrem Wertebereich verteilt, sondern folgen einer komplexen Verteilung, die analytisch schwer zu erfassen ist.

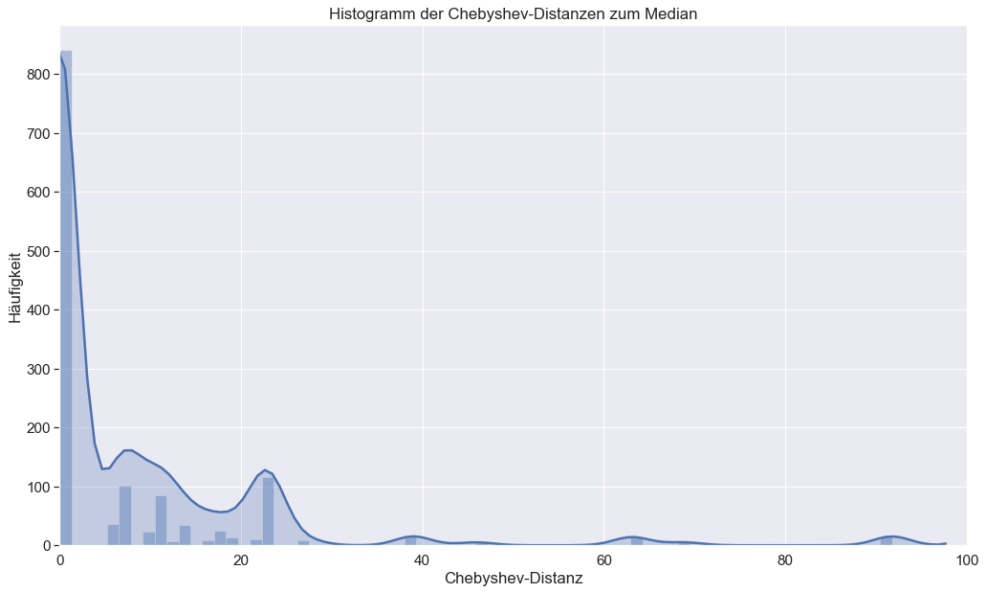


Abbildung 4.8: Histogramm (echt) und Einhüllende (skizziert) der Chebyshev-Distanzen zum Median

Kriegel et al. [14] haben mehrere Skalierungsansätze für die Anomalieerkennung vorgeschlagen, mit denen man z.B. die ausgerechneten Distanzen in einen Anomalie-Score umwandeln kann. Im Rahmen dieser Arbeit wurden diese Skalierungsansätze verglichen, indem die Stabilität der Skalierung über verschiedene Teildatensätze geprüft wurde. Aus den Ansätzen wurde die gaußsche Skalierung ausgewählt, welche gemäß der Gleichung 4.4 erfolgt.

$$\text{Norm}_D^{\text{gauss}}(x) := \max\left\{0, erf\left(\frac{D(x) - \mu_D}{\sigma_D \cdot \sqrt{2}}\right)\right\} \tag{4.4}$$

$D(x)$ - Distanz vom Datenpunkt x zum Median
μ_D - Erwartungswert der Distanzen
σ_D - Standardabweichung der Distanzen
erf - Die Gaußsche Fehlerfunktion

Die Abbildung 4.9 zeigt Verteilung der Anomalie-Scores, die nach der gaußschen Skalierung entstanden sind, innerhalb des Datensatzes mit dem Gipfel bei null. Die Verteilung der Anomalie-Scores sieht im Vergleich zur Verteilung der ausgerechneten Chebyshev-Distanzen ähnlich aus. Der sichtbare Unterschied ist, dass die entsprechenden Anomalie-Scores für die naheliegenden Distanzen durch die Skalierung etwas ausgedehnt wurden.

Hervorhebung von Anomalien im Log

Um die einzelnen Merkmale in einem SA-Log hervorzuheben, wurden die Beiträge jedes Merkmals zum Anomalie-Score analysiert. Dies erfolgte durch Beobachtung der Veränderung der Anomalie-Scores bei der Veränderung einzelner Merkmale.

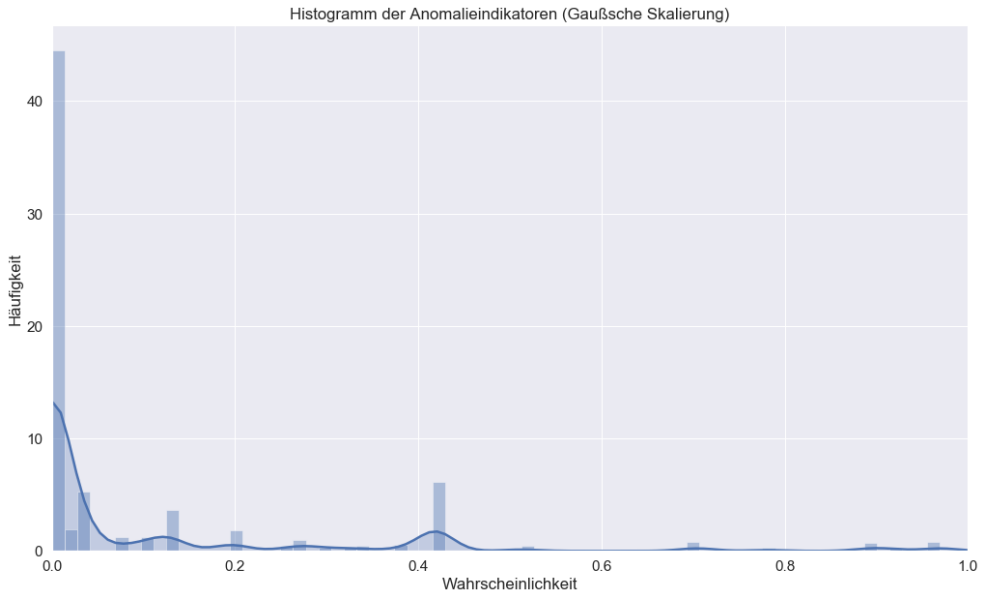
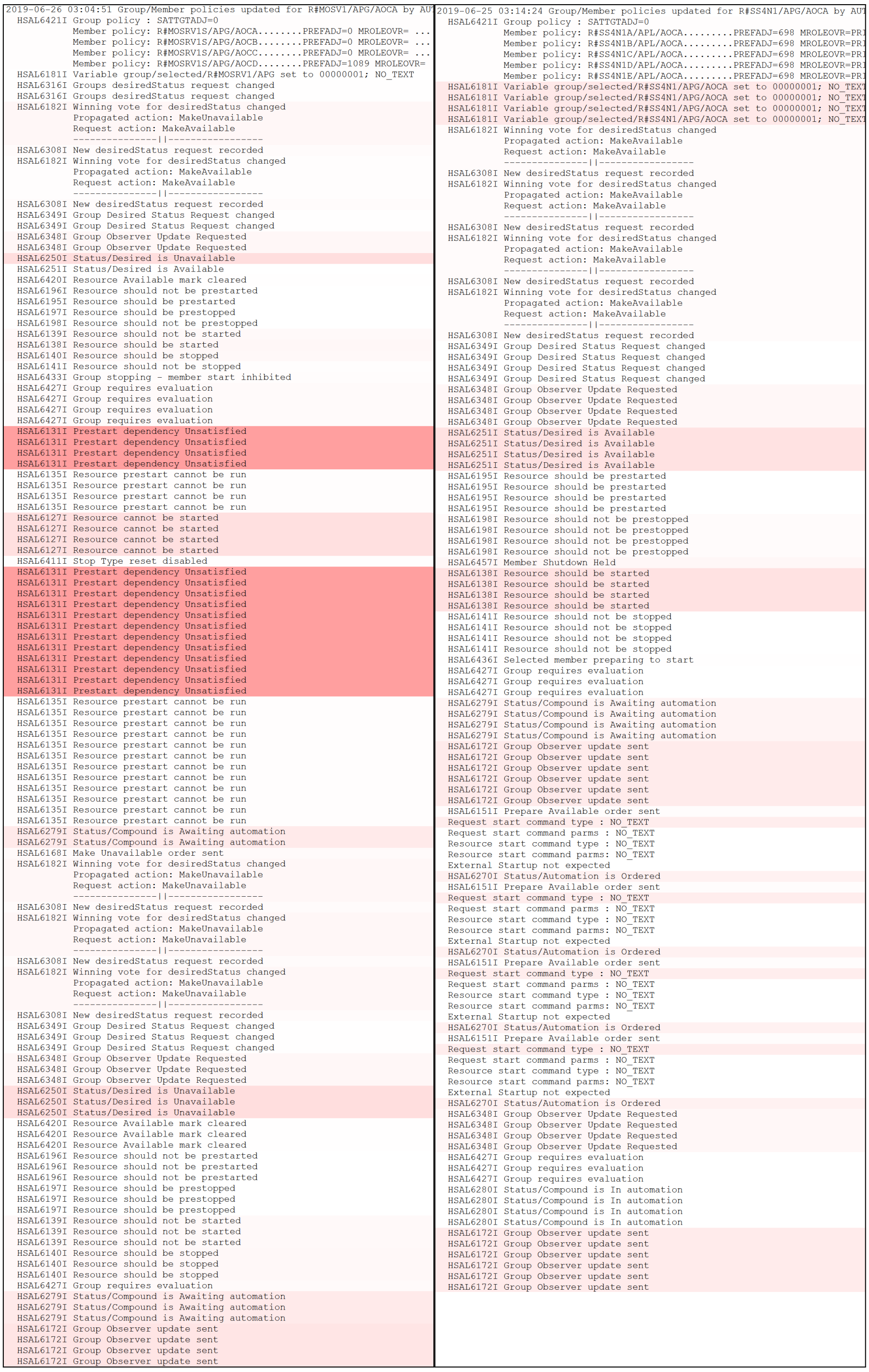


Abbildung 4.9: Anomalieindikator nach gaußscher Skalierung: Histogramm (echt) und Einhüllende (skizziert)

Die Beiträge einzelner Merkmale wurden in SA-Logs mit Farbintensität hervorgehoben, wie auf Abbildung 4.10 dargestellt ist. Je intensiver die Farbe ist, umso wahrscheinlicher handelt es sich hierbei um eine Anomalie. Zwei Workitems eines Workitemtyps sind gegenübergestellt, wobei der Workitem (a) anomal und der Workitem (b) normal sind. Die Distanz des normalen Workitems zum Median beträgt 18,11 und weist den Anomalie-Score von 29,9% auf. Der anomale Workitem hat eine deutlich höhere Distanz von 91,75 und einen hohen Anomalie-Score von 96,94%.



(a) Anomales Workitem (b) Normales Workitem

Abbildung 4.10: Vergleich von Hervorhebungen (metrisches Modell)

4.2.2 *Analytisches Modell*

Das zweite Modell basiert auf Bayes'scher Statistik. Das Ziel dieses Ansatzes ist, aus den HSAL-Sequenzen "$a, ..., z$" ihre Gesamtwahrscheinlichkeit $P(a, ..., z)$ sowie die bedingten Übergangswahrscheinlichkeiten (z.B. von b gegen a, $P(b|a)$) auszurechnen.

$$P(a, b, c, ..., y, z) = P(a) \cdot P(b|a) \cdot P(c|a, b) \cdot ... \cdot P(z|a, b, c, ..., y) \tag{4.5}$$

Die Gesamtwahrscheinlichkeit einer HSAL-Sequenz $a, b, c, ..., y, z$ kann durch die Gleichung 4.5 berechnet werden. Der vorhandene Datensatz enthält 290 einzigartigen Nachrichten und 190000 Sequenzen mit einer durchschnittlichen Länge von 100 Nachrichten. Das Ausrechnen aller Wahrscheinlichkeitsverteilungen für die Datenmenge ist rechenintensiv und benötigt viel Speicherplatz, um alle Parameter jeweiliger Verteilungen abzuspeichern. Um das handhabbar zu machen, wurde eine Näherung getroffen, konditionale Sequenzen ab einer bestimmten Länge abzuschneiden, z.B. $P(k|a, b, c, ..., i, j) \sim P(k|i, j)$. Das heißt, es wurden nur Eltern und Großeltern einer Nachricht in Betracht gezogen. Außerdem wurden wiederholende Nachrichten aus der HSAL-Sequenzen entfernt, so dass die benachbarten Nachrichten ungleich sind.

Da zusätzlich auch der Workitemtyp w_i auch berücksichtigt werden soll, sieht die angenäherte Gesamtwahrscheinlichkeit folgendermaßen aus:

$$P(a, b, c|w_i) = P(a|w_i) \cdot P(b|a, w_i) \cdot P(c|a, b, w_i) \tag{4.6}$$

Falls man das Problem in Form eines binären Nachrichtenübergangs $a \to b$ or $a \to \bar{b}$ betrachtet, kommt man zur Binomialverteilung:

$$Bin(k|p, N) = \begin{cases} \binom{N}{k} p^k (1-p)^{N-k} & \text{falls} \quad k \in \{0, 1, \ldots, N\} \\ 0 & \text{sonst.} \end{cases} \tag{4.7}$$

wobei p die Wahrscheinlichkeit eines Nachrichtenübergangs $a \to b$, N Anzahl aller Übergänge $a \to b \vee a \to \bar{b}$ und k Anzahl (erfolgreichen) Übergängen $a \to b$ sind. Dabei wird angenommen, dass die Übergangsversuche unabhängig sind.

Die Likelihood-Funktion für die Binomialverteilung kann man wie folgt approximieren:

$$\mathcal{L}(p|k, N) \propto p^k \cdot (1-p)^{N-k} \tag{4.8}$$

Die Beta-Verteilung $B(\alpha, \beta)$ ist eine Familie stetiger Wahrscheinlichkeitsverteilungen über dem Intervall (0, 1). Durch die Parameter α und β wird die Form der Verteilung kontrolliert. Die Beta-Verteilung ist die konjugierte A-priori-Verteilung (conjugate prior) der Binomialverteilung.

$$P_{conj.prior} = B(\alpha, \beta) \propto p^{\alpha-1} \cdot (1-p)^{\beta-1} \tag{4.9}$$

Wenn die Likelihood-Funktion binomial ist und ihre A-priori-Verteilung Beta ist, ist die resultierende Posteriori-Verteilung ebenfalls Beta. Die Wahrscheinlichkeitsverteilung für einen Übergang mit Berücksichtigung des Workitemtyps sieht deshalb für eine Sequenz $a \rightarrow b \rightarrow c$ folgendermaßen aus:

$$P_{posterior}(c|[a,b], w_i) = B(\alpha + k, \beta + N - k) \tag{4.10}$$

Der bedingte Erwartungswert der Posteriori-Verteilung wird mithilfe der Gleichung 4.11 ausgerechnet. $k = \#c|[a,b]$ ist die Anzahl der Nachrichten-Sequenzen $a \longrightarrow b \longrightarrow c$. $N = \#X|[a,b]$ ist die Anzahl der Nachrichten-Sequenzen $a \longrightarrow b \longrightarrow X$ für $\forall X \in \{a, b, c, ..., z\}$.

$$\mathrm{E}(c|[a,b], w_i) = \frac{\alpha + k}{\alpha + \beta + N} \tag{4.11}$$

Um die Übergänge zu modellieren, die im vorhandenen Datensatz nicht existieren, und diesen eine Übergangswahrscheinlichkeit $p > 0$ zuzuweisen, wurde $\alpha = \frac{1}{K}$ und $\beta = \alpha \cdot (K-1)$ ausgewählt, wobei K die Gesamtanzahl der einzigartigen HSAL-Nachrichten ist. Diese Werte sind arbiträr, spiegeln aber einen moderaten Prior mit $\mathrm{E}(X) = \frac{1}{K}$ und $Var(X) = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$ wider.

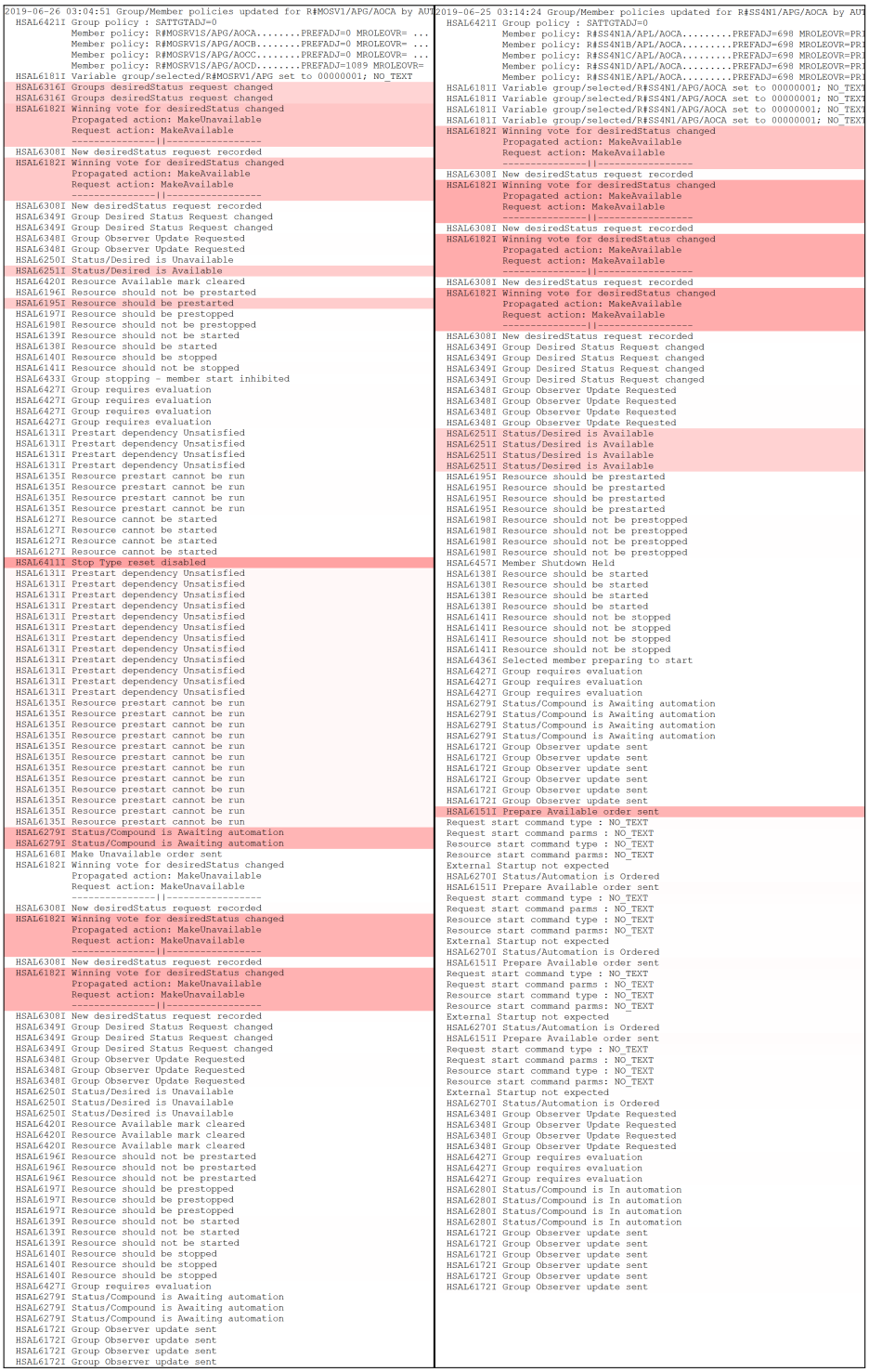
Um den Anfang und das Ende einer Sequenz in das Modell miteinbeziehen und vorhersagen zu können, wurden HSAL-Sequenzen mit generischen START- und END-Nachricht versehen. Dadurch wird verhindert, dass die Informationen von den Rändern weggeworfen und HSAL-Sequenzen, die aus einer bzw. zwei Nachrichten bestehen, verarbeiten werden.

Hervorhebung von Anomalien im Log

Der Anomalie-Score für die jeweilige HSAL-Nachricht c unter Berücksichtigung des Workitemtyps wird als Gegenwahrscheinlichkeit der bedingten Wahrscheinlichkeit für c berechnet:

$$\text{Anomalie-Score} = 1 - P(c|[a,b], w_i) \tag{4.12}$$

Die Ergebnisse dieses Modells kann man auf Abbildung 4.11 anschauen. Im Vergleich zum metrischen Modell, hebt dieses Modell unterschiedliche HSAL-Nachrichten hervor.



(a) Anomales Workitem (b) Normales Workitem

Abbildung 4.11: Vergleich von Hervorhebungen (analytisches Modell)

4.2.3 *LSTM-Modell*

Da die Nachrichten sequentiell aufgebaut sind und gewisse Abhängigkeiten von ihren Vorgängern aufweisen, eignet sich für das Problem ein LSTM-Modell, welches auch zeitlich rückwirkende Beziehungen berücksichtigt. Mithilfe dieses Modells wird die Wahrscheinlichkeitsverteilung der nächsten Nachricht in Abhängigkeit von ihren Vorgängern durch eine numerische Funktion approximiert.

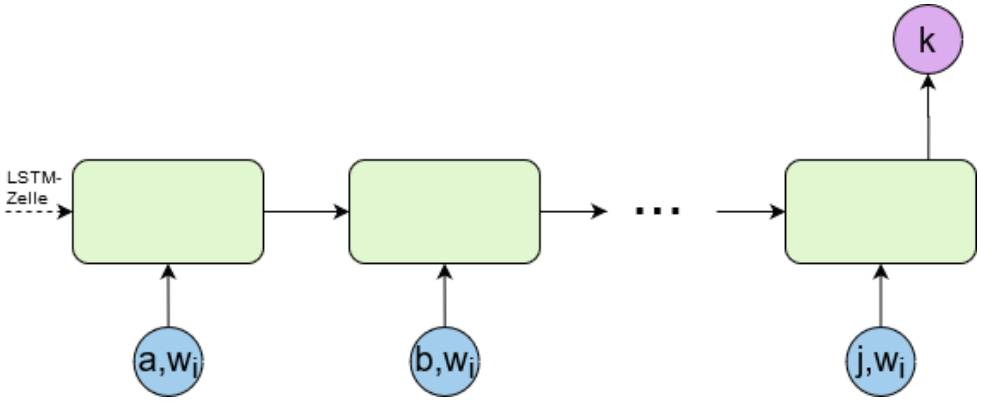


Abbildung 4.12: LSTM. Vorhersage der Nachricht k anhand von ihren Vorgänger

Wie in dem analytischen Modell wird die nächste Nachricht anhand von ihren Vorgängern und unter Berücksichtigung des Workitemtyps vorhergesagt. Ein LSTM-Modell liefert eine Liste der Wahrscheinlichkeiten jedes möglichen Nachfolgers zurück. In Abbildung 4.12 ist ein Vorhersagebeispiel der Nachricht k mit entsprechender Wahrscheinlichkeit $P(k|[a, b, ..., j], w_i)$ dargestellt.

Layer (type)	Output Shape	Param #
embedding_8 (Embedding)	(None, None, 10)	690
lstm_8 (LSTM)	(None, 30)	4920
dense_8 (Dense)	(None, 69)	2139
Total params: 7,749		
Trainable params: 7,749		
Non-trainable params: 0		

Tabelle 4.3: Zusammenfassung des LSTM-Models

In dieser Arbeit wird ein einfaches LSTM-Modell eingesetzt, die aus einer Einbettungsschicht und einer LSTM-Zelle besteht. Die Zusammenfassung des Modells kann man aus Tabelle 4.3 entnehmen. Als Verlustfunktion wurde die kategorische Kreuzentropie (siehe Gleichung 4.13) ausgewählt, da es sich hier um eine mehrklassige Klassifikation handelt. Damit die approxi-

mierte Pseudo-Wahrscheinlichkeit (Anomalie-Score) für jede vorhergesagte Nachricht unabhängig voneinander ist, verwendet das Netz eine sigmoid Aktivierungsfunktion.

$$L_{\text{cross\_entropy}}(y, \hat{y}) = -\sum_i y_i \cdot log(\hat{y}_i) \tag{4.13}$$

Durch die Kreuzentropie wird die Verteilung der Vorhersagen $\hat{y}$ mit der wahren Verteilung y verglichen. Die wahre Klasse wird als ein One-Hot-codierter Vektor dargestellt. Je näher die Ausgaben des Modells an diesem Vektor liegen, desto geringer ist der Verlust.

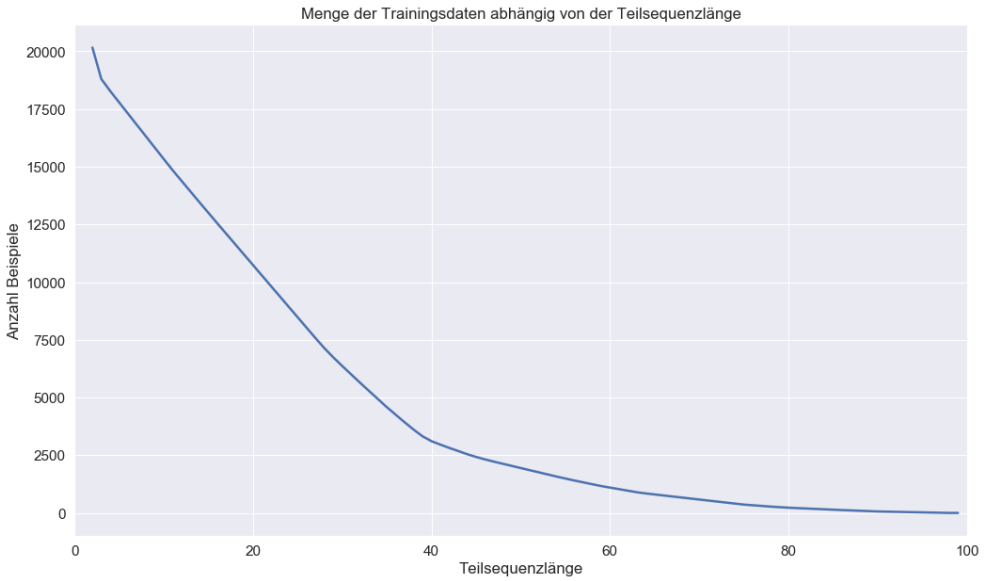


Abbildung 4.13: LSTM. Teilsequenzlängen

Da das eingesetzte Modell HSAL-Sequenzen fester Länge annimmt, wurden alle Sequenzen in Teilsequenzen aufgeteilt. Um eine optimale Teilsequenzlänge zu bestimmen, wurde zuerst die Anzahl Teilsequenzen in Abhängigkeit von Teilsequenzlänge analysiert. Dies ist in Abbildung 4.13 veranschaulicht. Man kann feststellen, dass die Anzahl der Teilsequenzen mit zunehmender Teilsequenzlänge relativ schnell fällt, was zu einem kleineren Trainingsdatensatz führt. Ein neuronales Netz, wie z.B. ein LSTM-Modell, benötigt in der Regel deutlich mehr Trainingsdaten als ein klassischer maschineller Lernalgorithmus. Aus diesem Grund besteht das Risiko von Overfitting oder Underfitting, eine Situation, bei der die Beziehungen in den längeren Teilsequenzen nicht erfasst werden.

Außerdem wurde der Einfluss der Teilsequenzlängen auf die Validierungsmetriken analysiert.

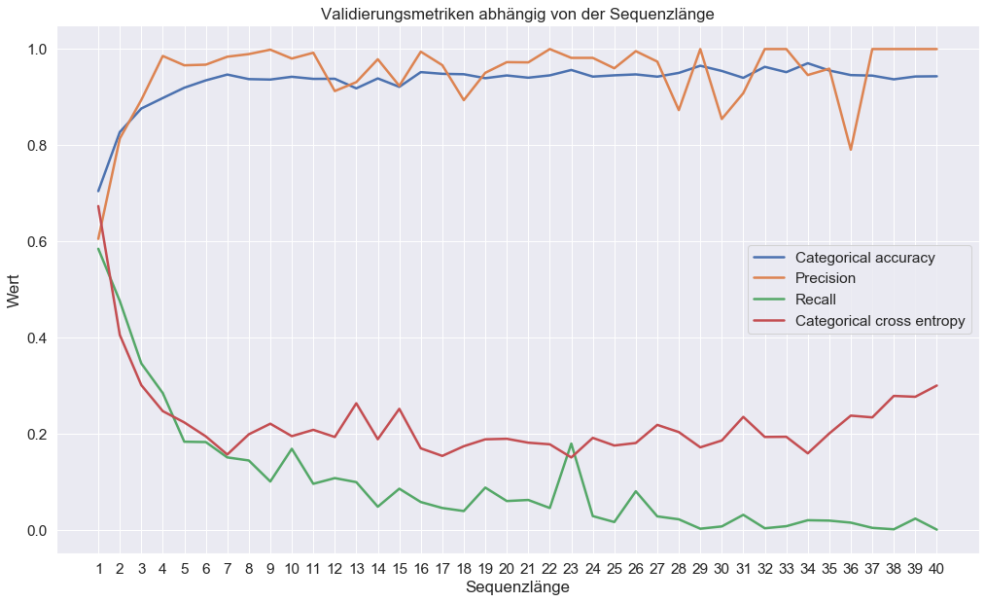
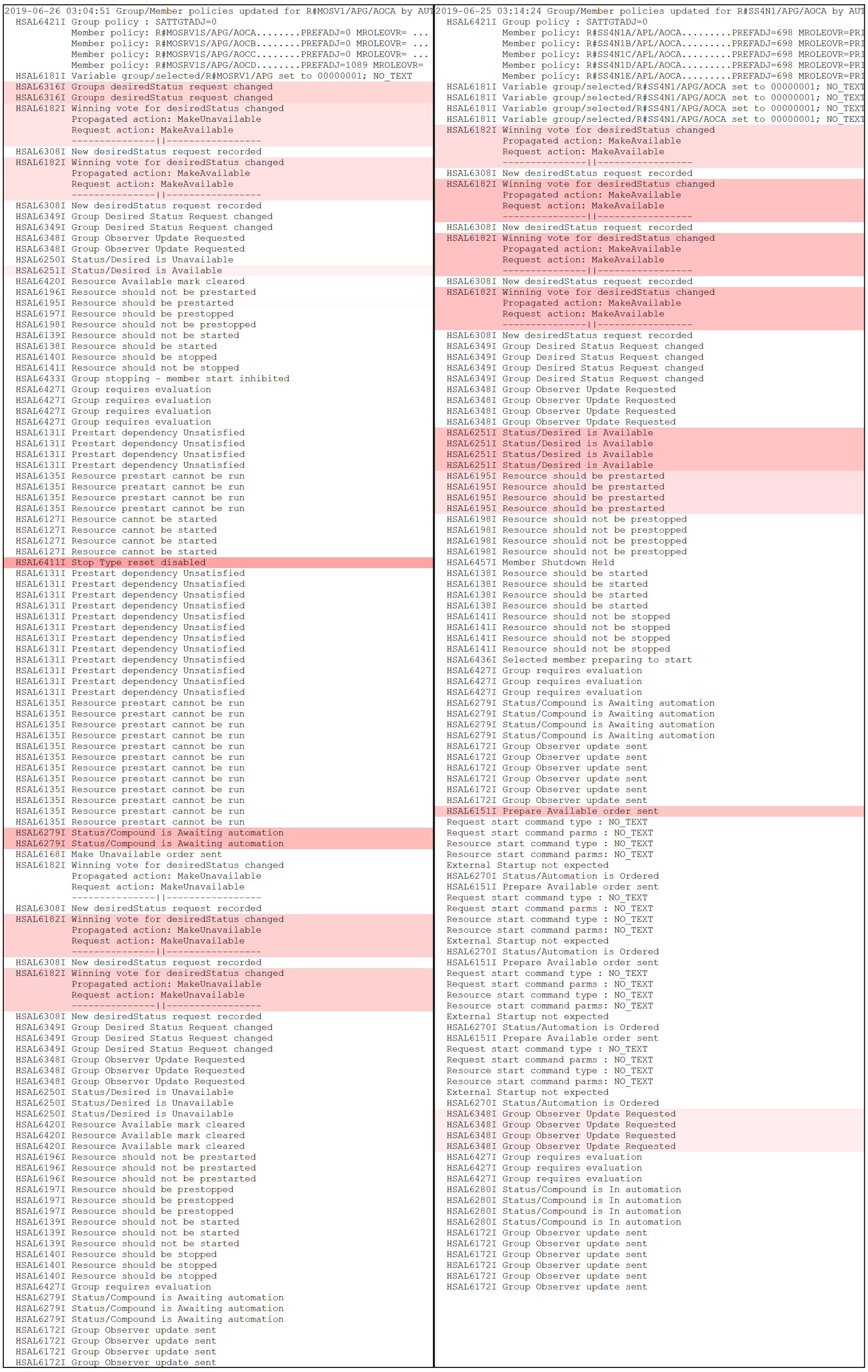


Abbildung 4.14: LSTM. Validierungsmetriken abhängig von Teilsequenzlängen

Die Veränderung unterschiedlicher Validierungsmetriken in der Abhängigkeit von Teilsequenzlängen kann man in Abbildung 4.13 sehen. Die Berechnungen wurden lediglich bis zur Teilsequenzlänge von 40 durchgeführt, da dies zeitaufwändig ist und die Datenmenge ab dem Punkt zu klein wird. Mit der Vergrößerung der Teilsequenzlänge weisen die kategorische Genauigkeit und Precision keine signifikante Steigung auf. Schon ab sieben Vorgänger verhält sich das Modell unstabil und weist gewisse Schwankungen der Validierungsmetriken auf, was auf Mangel an Trainingsdaten hindeutet. Der Anteil richtig vorhergesagten Nachrichten (Recall) sinkt hingegen deutlich. Infolgedessen kann man schließen, dass eine optimale Teilsequenzlänge für die Datenmenge zwischen zwei und vier liegt.

Hervorhebung von Anomalien im Log

Der Anomalie-Score wird analog zum metrischen Modell berechnet. Hervorhebungen eines LSTM-Modells kann man auf Abbildung 4.15 sehen. Das LSTM-Modell wurde auf drei Vorgänger trainiert.



(a) Anomales Workitem (b) Normales Workitem

Abbildung 4.15: Vergleich von Hervorhebungen (LSTM-Modell)

5 DISKUSSION

In diesem Kapitel werden die Ergebnisse der Merkmalsextraktion sowie verwendeter Anomalieerkennungsmodelle und ihre praktische Implikationen diskutiert.

MERKMALSEXTRAKTION

Ein wesentlicher Bestandteil dieser Arbeit ist die Merkmalsextraktion. Aus den SA-Log-Daten wurden zwei Merkmalsdarstellungen extrahiert: Merkmal-Vektoren und HSAL-Sequenzen.

Merkmal-Vektoren

Merkmal-Vektoren benötigen manuelles Featureengineering und Bereichswissen. Jedes Workitem wird bei dieser Darstellung durch einen Merkmal-Vektor fester Länge repräsentiert. Obwohl das Featureengineering einen gewissen Zeitaufwand erfordert, führt es zu einem tieferen Verständnis der Daten vor allem durch die explorative Datenanalyse, bei der auch die Qualität der Daten begutachtet wird. Im Laufe der Analyse erwiesen sich Ressourcentypen (Anwendungen, Gruppen, Monitore etc.) als redundant.

Die Analyse der Workitem-Headers hat aufgezeigt, dass Workitemtypen sich dermaßen voneinander unterscheiden, dass sie eventuell disjunkte Teilmengen der Merkmale enthalten. Aus diesem Grund wurde entschieden, die Anomalieerkennung nach jedem Workitemtyp separat durchzuführen.

Erkennung der untypischen Einzelnachrichten im Log wurde durch die HSAL-Singles abgedeckt. Durch die HSAL-Paare und HSAL-Tripel wurden sequentielle Beziehungen der HSAL-Nachrichten als Merkmale dargestellt, so dass die charakteristischen Paare und Tripel von den Anomalieerkennungsmodellen in Betracht gezogen werden können. Die Aufnahme weiterer geordneten Kollokationsmerkmale (z.B. Quadrupel und Quintupel) führt zur Explosion der Größe des Merkmalsraums und folglich zum Fluch der Dimensionalität, der einen negativen Einfluss auf Performanz der Data-Mining-Modelle aufweist.

Da die HSAL-Nachrichten gelegentlich auch variable Informationen z.B. über die ausgeführte Unteraktion oder Änderung einer SA-Variable enthalten, führt die Vektorisierung dieser Informationen zu einer besseren Unterscheidung der Datenpunkte sowie Anomalieindikatoren.

Extraktion der Merkmale, die Bereichswissen erfordern und sich nicht direkt aus den Daten ableiten lassen, bietet den Vorteil einer genaueren Verallgemeinerung anhand begrenzter verfügbarer Daten und führt in der Regel zur Steigung der Performanz eines Data-Mining-Modells [2, 29]. Allerdings wurde festgestellt, dass der Gruppentyp und die hierarchischen Abhängig-

keiten der Anwendungen sowie Anwendungsgruppen sich aus den SA-Log-Daten und ohne Zugriff zum laufenden System nicht zuverlässig extrahieren lassen.

HSAL-Sequenzen

Extraktion der HSAL-Sequenzen stellt im Vergleich zu Merkmal-Vektoren keinen hohen Aufwand dar. HSAL-Sequenzen sind zeitlich geordnete Serien der HSAL-Nachrichten, bei denen verwandte Nachrichten aufeinander folgen. Diese Merkmalsdarstellung bietet für die Anomalieerkennung die Möglichkeit, Anomalität einer Nachricht anhand ihres Kontextes zu bestimmen, das heißt, dass die HSAL-Nachrichten nicht einzeln aus dem Kontext gegriffen, sondern als Sequenz aufgenommen werden. Variable Informationen lassen sich problemlos in die HSAL-Sequenzen einspielen.

Der Nachteil dieser Merkmalsdarstellung ist, dass die dafür entwickelten Data-Mining-Modelle (z.B. ein RNN) wesentlich mehr Trainingsdaten erfordern, um bei der gleichen Datenmenge mit den Data-Mining-Modellen für Merkmal-Vektoren eine vergleichbare Zuverlässigkeit zu erzielen.

ANOMALIEERKENNUNGSMODELLE

Die obengenannten Merkmalsdarstellungen wurden in drei unterschiedlichen Anomalieerkennungsmodellen eingesetzt. Das metrische Modell verwendet die Merkmal-Vektoren. Das analytische und LSTM-Modell verwenden hingegen HSAL-Sequenzen.

Metrisches Modell

Metrisches Modell basiert auf der Annahme, dass die Anomalien entfernt von der Gesamtstreuung der Daten liegen. Je weiter ein Datenpunkt entfernt liegt, umso anomaler wird er von dem Modell gekennzeichnet. In dieser Arbeit wurde Chebyshev-Distanz zum Median als Maß für Anomalität ausgewählt, weil es für die vorliegenden Daten gleichzeitig am aussagekräftige und robuste Ergebnisse erzielt hat, und danach mithilfe des Gaußschen Skalierung in die Pseudo-Wahrscheinlichkeit umgewandelt.

Die Ausgabe des metrischen Modells ist ein Anomalie-Score des gegebenen Workitems. Um die charakteristischsten Nachrichten (Anomalieindikatoren) im in einem Merkmal-Vektor zu finden und danach im Log hervorzuheben, müssen Einflüsse einzelner Merkmale separat ausgerechnet werden.

Ein Beispiel hervorgehobener Anomalieindikatoren im Log wurde auf Abbildung 4.10 angeführt. Links befindet sich das Workitem, welches am weitesten von der Gesamtstreuung der Datenpunkte und vom Median liegt. Wirft man einen Blick auf die hervorgehobenen Nachrichten, so fällt auf, dass die charakteristischsten Nachrichten auf Problemzustände bzw. unbefriedigte Abhängigkeiten hindeuten, was in der Tat ein anomales Verhalten darstellt. Allerdings sieht man auch die Nachricht "HSAL6135I Resource prestart cannot be run", die auch auf einen Problemzustand hinweist und kaum hervorgehoben ist. Das kann daran liegen, dass die Nachricht lediglich in

zwei HSAL-Paare vorkommt und nicht als HSAL-Single im Merkmal-Vektor repräsentiert ist, weil sie zu selten im Datensatz auftritt und durch die Merkmalsselektion ausgeschlossen wird. Das rechte Workitem, welches sich näher am Median befindet, wird deutlich weniger hervorgehoben und entspricht tatsächlich einem normalen Workitem.

Aus dem Grund, dass das Funktionsprinzip dieses Modells intuitiv verständlich ist und es sich leicht veranschaulichen lässt (Distanzgraph, PCA-Komponenten), kann man von einer hohen Interpretierbarkeit dieses Modells sprechen.

Da die ausgerechnete Distanz zum Median sich nicht vergrößert, sobald eine der Nachrichten nicht auftritt, werden fehlenden Nachrichten mithilfe dieses Modells nicht erkannt. Außerdem hat sich erwiesen, dass die Merkmalsselektion durch die Auswahl der häufigsten (nach dem p-Quantil) Einzelnachrichten sowie Paare und Tripel zur Vernachlässigung der potenziell charakteristischen Merkmale führt, was beispielsweise mit der obengenannten Nachricht `HSAL6135I` passiert ist. Dieser Schwachpunkt des gewählten Merkmal-Vektor-Ansatzes führt außerdem dazu, dass die Erkennung der Nachrichtenreihenfolgen, die vom üblichen Muster abweichen und selten im Datensatz auftreten, nicht ohne weiteres möglich ist. Aus dieser Beobachtung kann man schließen, dass es ein Verbesserungspotenzial für die Merkmalsextraktion sowie für das Modell besteht.

Analytisches Modell

Analytisches Modell nutzt die HSAL-Sequenzen und die Bayes'sche Statistik, um die Wahrscheinlichkeit einer Nachricht anhand ihrer Vorgänger auszurechnen. In dieser Arbeit wurde die Anzahl Vorgänger aus performancetechnischen Gründen auf zwei beschränkt.

Das analytische Modell gibt die Wahrscheinlichkeit für eine Nachricht aus, die als Anomalieindikator genutzt wird. Ein Beispiel solcher hervorgehobenen Anomalieindikatoren im Log wurde auf Abbildung 4.11 dargestellt. Es lässt sich feststellen, dass das analytische Modell im Vergleich zum metrischen Modell deutlich unterschiedliche HSAL-Nachrichten in beiden Workitems als unwahrscheinlich kennzeichnet. Der nächste Unterschied liegt außerdem daran, dass das analytische Modell die gleichen Nachrichten (z.B. "`HSAL6182I` Winning vote for desiredStatus changed") je nach ihren Vorgängern unterschiedlich hervorhebt.

Der Anomalie-Score für ein Workitem (Gesamtwahrscheinlichkeit) wird als Produkt aller Anomalieindikatoren ausgerechnet. Dies ist herausfordernd, da das Produkt mit der Verlängerung der HSAL-Sequenz schnell gegen null geht und deshalb normiert werden muss. Während der Arbeit wurde versucht, einen Normierungsfaktor in Abhängigkeit von der Sequenzlänge auszurechnen, wurde allerdings konnte kein zuverlässiges Ergebnis erzielt werden. Daraus ergibt sich ein Potenzial für die fortführenden Untersuchungen.

Das analytische Modell kann potenziell untypische sowie vertauschte Nachrichten erkennen. Falls eine Nachricht fehlt, wird momentan das nächste als untypisch gekennzeichnet. Für eine künftige Arbeit besteht die Möglichkeit,

die fehlenden Nachrichten im Log hervorzuheben, indem man beispielsweise den höchstwahrscheinlichen Nachfolger hinzufügt und ihn mit einer anderen Farbe hervorhebt. Dies würde die weitere Analyse des Logs für einen Experten erleichtern.

LSTM-Modell

Das LSTM-Modell nutzt die HSAL-Sequenzen und im Vergleich zum analytischen Modell approximiert die Wahrscheinlichkeitsverteilung der nächsten Nachricht durch eine numerische Funktion.

Analog zum analytischen Modell gibt das LSTM-Modell die Wahrscheinlichkeit für eine Nachricht in Abhängigkeit von ihren Vorgängern aus. Es geht allerdings hier um Pseudo-Wahrscheinlichkeit, die als Anomalieindikator genutzt wird. Beispiele der mithilfe dieses Modells hervorgehobenen Logs wurden auf Abbildung 4.15 veranschaulicht. Die Hervorhebungen dieses Modells sehen dem analytischen Modell bis auf ein paar Nachrichten ähnlich aus, was heißt, dass die Approximation der Wahrscheinlichkeitsverteilung relativ gut funktioniert.

Man kann hier, wie bei dem analytischen Modell, dasselbe Problem mit der Ausrechnung des Anomalie-Scores für ein Workitem beobachten.

Erkennung der "untypischen" Nachrichten und Nachrichtenreihenfolgen ist mithilfe des LSTM-Modells möglich. Das Problem der fehlenden Nachrichten ist dem analytischen Modell analog. Gänzlich unbekannte Sequenzen, die nicht in den Trainingsdaten enthalten sind, resultieren in zufälligen Vorhersagen.

Der Vorteil dieses Modells liegt an ihrer Flexibilität und leichter Erweiterbarkeit. Das heißt, dass nicht nur die Anzahl Vorgänger sich mühelos variieren lässt, sondern auch die Komplexität des Netzes. Man könnte außerdem Nachfolger und Bereichswissen in den Trainingsprozess einspielen.

Allerdings erfordert ein LSTM-Modell deutlich mehr Daten als ein anderes Modell, welches nicht auf künstlichen neuronalen Netzen basiert. Die Ergebnisse des LSTM-Modells lassen sich, wie bei allen anderen Blackbox-Modellen, schwer interpretieren.

6 FAZIT

6.1 ZUSAMMENFASSUNG

Im Rahmen dieser Arbeit wurde eine Softwarelösung für unüberwachte Anomalieerkennung anhand von Log-Daten der IBM System Automation entwickelt. Als erster Schritt wurden die charakteristischen Merkmale aus den Log-Daten in Form von Merkmal-Vektoren und Nachrichten-Sequenzen dargestellt. Diese Darstellungen wurden im zweiten Schritt mithilfe der existierenden Methoden des Data-Mining analysiert, um Indikatoren verschiedener Anomalien im Log aufzuzeigen. Abschließend wurde die verwendeten Methoden miteinander verglichen.

Das metrische Modell konnte zwar nicht alle Anomalietypen erkennen, wies allerdings eine gute Interpretierbarkeit auf, welche für einen Experten eine wesentliche Rolle spielen kann. Außerdem muss die Anomalität eines Workitem nicht extra berechnet werden.

Falls ausreichend Daten zur Verfügung stehen und keinen großen Wert auf die Interpretierbarkeit gelegt werden würde, sollte das LSTM-Model für die Anomalieerkennung in Betrieb genommen werden, weil es sich als vielversprechend und mächtig erwiesen hat und weil das Auftreten vollkommen fremder Sequenzen in diesem Anwendungsszenario sehr unwahrscheinlich ist.

6.2 AUSBLICK

In einem nächsten Schritt sollte eine für die Validierung ausreichende Menge echter gelabelten Daten, die auch tatsächliche Anomalien enthalten, durch Zusammenarbeit mit Kunden erhoben werden. Der gelabelte Datensatz sollte nicht nur das Finetuning der in dieser Arbeit umgesetzten Modelle erleichtern, sondern auch die Umsetzung anderer (beobachteten) Anomalieerkennungsmodelle ermöglichen.

Die Verbesserung der Hervorhebung anomaler und fehlender Log-Einträge sollte als nächster Schritt vorgenommen werden. Hierbei könnte der Anomalie-Score eines Workitems als Entscheidungskriterium dienen, ob die Log-Einträge in diesem Workitem hervorgehoben werden müssen. Allerdings muss zuerst das Normierungsproblem gelöst werden.

Weiteres Verbesserungspotential besteht vor allem bei der Merkmalsextraktion. Anhand der gelabelten Kundendaten würden sich charakteristischen Merkmale erheblich leichter extrahieren lassen. Falls man einen Zugriff zum laufenden System hätte, könnte man auch Kontextinformationen wie Gruppentype und Abhängigkeiten im Ressourcenbaum für die Verbesserung der Anomalieerkennung verwenden.

LITERATUR

[1] Charu C. Aggarwal. *Outlier Analysis*. 2nd. Springer Publishing Company, Incorporated, 2016. ISBN: 3319475770, 9783319475776.

[2] Richard Ambrosino und Bruce G Buchanan. "The use of physician domain knowledge to improve the learning of rule-based models for decision-support". In: *Proceedings of the AMIA Symposium*. American Medical Informatics Association. 1999, S. 192.

[3] *Automation Manager and Automation Agent*. URL: https://www.ibm.com/support/knowledgecenter/SSWRCJ_4.1.0/com.ibm.safos.doc_4.1/GetStarted/ManagerAgent.html (besucht am 15. 01. 2020).

[4] R. Bellman, Rand Corporation und Karreman Mathematics Research Collection. *Dynamic Programming*. Rand Corporation research study. Princeton University Press, 1957. ISBN: 9780691079516. URL: https://books.google.it/books?id=wdtoPwAACAAJ.

[5] Usama Fayyad, Gregory Piatetsky-Shapiro und Padhraic Smyth. "From data mining to knowledge discovery in databases". In: *AI magazine* 17.3 (1996), S. 37–37.

[6] Ilenia Fronza, Alberto Sillitti, Giancarlo Succi, Mikko Terho und Jelena Vlasenko. "Failure prediction based on log files using Random Indexing and Support Vector Machines". In: *Journal of Systems and Software* 86.1 (2013), S. 2 –11. ISSN: 0164-1212. DOI: 10.1016/j.jss.2012.06.025.

[7] Felix Gers, Jürgen Schmidhuber und Fred Cummins. "Learning to Forget: Continual Prediction with LSTM". In: *Neural computation* 12 (Okt. 2000), S. 2451–71. DOI: 10.1162/089976600300015015.

[8] Douglas M Hawkins. *Identification of outliers*. Bd. 11. Springer, 1980.

[9] Sepp Hochreiter und Jürgen Schmidhuber. "Long Short-term Memory". In: *Neural computation* 9 (Dez. 1997), S. 1735–80. DOI: 10.1162/neco.1997.9.8.1735.

[10] Pawel Jurczyk und Eugene Agichtein. "Discovering authorities in question answer communities by using link analysis". In: Jan. 2007, S. 919–922. DOI: 10.1145/1321440.1321575.

[11] Tomihisa Kamada und Satoru Kawai. "An algorithm for drawing general undirected graphs". In: *Information Processing Letters* 31.1 (1989), S. 7 –15. ISSN: 0020-0190. DOI: 10.1016/0020-0190(89)90102-6.

[12] Edwin M Knorr und Raymond T Ng. "Algorithms for Mining Distance-Based Outliers in Large Datasets". In: *Proceedings of the 24rd International Conference on Very Large Data Bases*. Morgan Kaufmann Publishers Inc., 1998. ISBN: 1558605665.

[13] Edwin M Knorr und Raymond T Ng. "Finding intensional knowledge of distance-based outliers". In: *VLDB*. Bd. 99. 1999, S. 211–222.

[14] Hans-Peter Kriegel, Peer Kroger, Erich Schubert und Arthur Zimek. "Interpreting and unifying outlier scores". In: *Proceedings of the 2011 SIAM International Conference on Data Mining*. SIAM. 2011, S. 13–24. DOI: 10.1137/1.9781611972818.2.

[15] Siyang Lu, Xiang Wei, Yandong Li und Liqiang Wang. "Detecting Anomaly in Big Data System Logs Using Convolutional Neural Network". In: *2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing, 16th Intl Conf on Pervasive Intelligence and Computing, 4th Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech)*. Aug. 2018, S. 151–158.

[16] Christopher D. Manning, Prabhakar Raghavan und Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008. DOI: 10.1017/CBO9780511809071.

[17] Ani Nenkova und Kathleen McKeown. "A survey of text summarization techniques". In: *Mining text data*. Springer, 2012, S. 43–76.

[18] Xuetong Niu, Li Wang und Xulei Yang. *A Comparison Study of Credit Card Fraud Detection: Supervised versus Unsupervised*. 2019. arXiv: 1904.10604 [cs.LG].

[19] *ODD '13: Proceedings of the ACM SIGKDD Workshop on Outlier Detection and Description*. Chicago, Illinois: Association for Computing Machinery, 2013. ISBN: 9781450323352.

[20] Christopher Olah. *Understanding LSTM Networks*. 2015. URL: https://colah.github.io/posts/2015-08-Understanding-LSTMs/ (besucht am 18. 01. 2020).

[21] Goran Oreški. "An experimental comparison of classification algorithm performances for highly imbalanced datasets". In: Sep. 2014.

[22] Wei Peng, Tao Li und Sheng Ma. "Mining logs files for data-driven system management". In: *Acm Sigkdd Explorations Newsletter* 7.1 (2005), S. 44–51.

[23] *SA z/OS overview*. URL: https://www.ibm.com/support/knowledgecenter/en/SSWRCJ_3.5.0/com.ibm.safos.doc_3.5/UserGuide/Overview.html (besucht am 15. 01. 2020).

[24] Alireza Vafaei Sadr, Bruce A. Bassett und Martin Kunz. *A Flexible Framework for Anomaly Detection via Dimensionality Reduction*. 2019.

[25] Lu Siyang. "Detecting Anomaly in Big Data System Logs". Diss. University of Central Florida, 2019.

[26] *Working with Application Groups*. URL: https://www.ibm.com/support/knowledgecenter/SSWRCJ_4.1.0/com.ibm.safos.doc_4.1/UserGuide/Working_with_Application_Groups.html (besucht am 15. 01. 2020).

[27] Wei Xu, Ling Huang, Armando Fox, David Patterson und Michael I. Jordan. "Detecting Large-Scale System Problems by Mining Console Logs". In: *Proceedings of the ACM SIGOPS 22nd Symposium on Operating Systems Principles*. Association for Computing Machinery, 2009. ISBN: 9781605587523. DOI: 10.1145/1629575.1629587.

[28] Tian Yang und Vikas Agrawal. "Log file anomaly detection". In: (2016). URL: https://cs224d.stanford.edu/reports/YangAgrawal.pdf.

[29] Ting Yu. "Incorporating Prior Domain Knowledge intoInductive Machine LearningIts implementation in contemporary capitalmarkets". Diss. University of Technology Sydney, 2007. URL: http://hdl.handle.net/10453/20070.

[30] Tim Zwietasch. "Detecting Anomalies in SystemLog Files using Machine Learning Techniques". Bachelor's Thesis. University of Stuttgart, 2. Okt. 2014. URL: https://elib.uni-stuttgart.de/bitstream/11682/3471/1/BCLR_0148.pdf.