Hochschule Magdeburg-Stendal
Fachbereich Ingenieurwissenschaften und Industriedesign (IWID)
Institut für Elektrotechnik

Masterarbeit

zur Erlangung des Grades eines "Master of Engineering"
im Studiengang Elektrotechnik

Thema: Entwicklung einer Methode zur Erkennung von OPC-UA-Kommunikationsbeziehungen

Eingereicht von: **Julian Neubauer**

Angefertigt für: SBSK GmbH I& Co. KG

Matrikel: **20181181**

Ausgabetermin: 05.08.2019

Abgabetermin: 23.12.2019

Schulischer Betreuer: Prof. Dr.-Ing. Dieter Schwarzenau

Betrieblicher Betreuer: M.Eng. Daniel Gerlach

....
1. Prüfer 2. Prüfer

Inhalt

Abkürzungsverzeichnis

AE	Alarm Events
API	Application Programming Interface
BLINC	BLINd Classfication
CART	Classfication and Regression Tree
COM	Component Object Model
DA	Data Access
DCOM	Distributed Component Object Model
DiffServ, DS	Differentiated Services
DNS	Domain Name System
DPI	Deep Packet Inspection
EuQoS	End-to-End Quality-of-Service support over heterogeneous networks
FIFO	First In – First Out
GM	Geometrisches Mittel
HDA	Historical Data Access
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
I40	Industrie 4.0
ICS	Industrial Control System
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IIoT	Industrial Internet of Things
IntServ	Integrated Services
IoT	Internet of Things
IP	Internet Protocol
IQR	Interquartilsabstand
ISO	Internationale Organisation für Normung
IT	Informationstechnologie
kNN	k-nächste Nachbarn

LAN	Local Area Network
LDA	Linear Discriminant Analysis
LR	Logistic Regression
M2M	Machine-to-Machine
MCC	Matthews Correlation Coeffcient
ML	Machine Learning
MOM	Message Oriented Middleware
MONAT	modellbasierte und bedarfsgerechte Netzwerkkonfiuration für Netzwerke der Automatisierung und Telekommunikation
NB	Naive Bayes
NFV	Network Functions Virtualisation
NFVI	Network Functions Virtualisation Infrastructure
NUC	Next Unit of Computing
OLE	Object Linking and Embedding
ONF	Open Network Foundation
OPC	Open Platform Communication
OPC UA	OPC Unified Architecture
OSI	Open Systems Interconnection
P2P	Peer-to-Peer
PC	Personal Computer
POP	Point of Presence
PubSub	Publish/Subscribe
QoS	Quality of Service
QUIC	Quick UDP Internet Connections
RDP	RemoteDesktop Protocol
RSVP	Resource Reservation Protocol
RTCP	RealTime Control Protocol
RTP	Real-Time Transport Protocol
SANE	Secure Architecture for the Networked Enterprise

SCADA	Supervisory Control and Data Aquisition
SDN	Software-Defined Networking
SIP	Session Initiation Protocol
SOAP	ursprünglich Simple Object Access Protocol, seither kein Akronym mehr
SSL	Secure Sockets Layer
SVM	Support Vector Machines
TCP	Transmission Control Protocol
TLS	Transport Layer Security
ToS	Type-of-Service-Byte
TSN	Time-Sensitive Networking
UA	Unified Architecture
UDP	User Datagram Protocol
VarK	Variationskoeffizient
VLC	VideoLan Client
VNF	Virtual Network Functions
VoIP	Voice over IP
VPN	Virtual Private Network
W3C	World Wode Web Consortiums
WAN	Wide Area Network
WebRTC	Web Real-Time Communication
WS	WebSocket
XML	Extensible Markup Language

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Masterarbeit selbstständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe. Die aus fremden Quellen direkt oder indirekt übernommenen Stellen sind als solche kenntlich gemacht.

Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungs-behörde vorgelegt und auch nicht veröffentlicht.

Ort, Datum	Unterschrift des Verfassers

Vorbemerkung und Danksagung

Die vorliegende Arbeit entstand während meiner Tätigkeit bei der SBSK GmbH & Co. KG im Rahmen des kooperativen Forschungsprojekts „MONAT - Modellbasierte und bedarfsgerechte Netzwerkkonfiguration für Netzwerke der Automatisierung und Kommunikation“. Zu Beginn der Anfertigung war eine umfassende Einarbeitung in neue Themengebiete notwendig und speziell im Bereich der industriellen Kommunikation und im Anwendungsfeld des maschinellen Lernens folgte eine steile Lernkurve. Die zahlreichen Einblicke und Erkenntnisse, die ich im Verlauf der Arbeit gewonnen habe, waren äußerst einprägsam und lehrreich.

Herrn Prof. Dr.-Ing. Dieter Schwarzenau danke ich für seine Unterstützung bei der Themenfindung, die Herstellung des Kontakts und sein wissenschaftliches Interesse an der Thematik sowie seine Hilfestellungen für die Erstellung dieser Arbeit. Ich danke Herrn Dipl.-Ing. Dirk Bartens für sein Engagement, das den Beginn der Arbeit deutlich beschleunigt hat, sowie spannende Einblicke in aktuelle Entwicklungen der Informations- und Kommunikationstechnik.

Besonderer Dank gilt meinem betrieblichen Betreuer Herrn M.Eng. Daniel Gerlach, der durch seinen unermüdlichen Einsatz und seine zahlreichen Anregungen die Anfertigung der Arbeit in dieser Form erst ermöglicht hat. Nicht zuletzt hat er für dieses Vorhaben sogar die Ruhe seines Einzelbüros aufgegeben. Herrn Dipl.-Inf. Carsten Edelberger danke ich für seine fachlichen Ratschläge und vorausschauende Denkweise zur Problemvermeidung.

Ich danke der SBSK GmbH & Co. KG für Bereitstellung elementarer Ressourcen, wodurch u.a. die Realisierung eines eigenen Testaufbaus möglich war. Gleichzeitig danke ich meinen Kolleginnen und Kollegen, die mir den Einstieg in das neue Arbeitsumfeld sehr angenehm gestaltet haben.

Zu guter Letzt bedanke ich mich bei meiner Familie und meinen Freunden für ihre kontinuierliche Unterstützung und der manchmal benötigten Ablenkung von der Thematik zum Schöpfen neuer Kraft.

Magdeburg, im Dezember 2019

Julian Neubauer

Inhaltsangabe

Die digitale Transformation der Industrie zielt auf eine Verschmelzung der Produktion mit neuester Informations- und Kommunikationstechnik ab, um effizientere und flexiblere Produktionsabläufe zu ermöglichen. Im Rahmen des Projekts Industrie 4.0 soll durch die Vernetzung eingebetteter Systeme und Maschinen sowie die Anbindung der Produktionsprozesse an die Cloud die Fertigung individualisiert und neue Geschäftsmodelle erschlossen werden.

Als standardisierte Kommunikationsplattform für die Industrie garantiert OPC Unified Architecture (OPC UA) den sicheren, zuverlässigen und herstellerunabhängigen Datentransport über alle Ebenen der Automatisierungspyramide. Gleichzeitig bietet die Etablierung von SD-WAN-Strukturen in diesem Zusammenhang einen dynamischen und skalierbaren Lösungsansatz, um den Anforderungen an Hochverfügbarkeit und geringe Latenzen im Bereich der Datenübertragung zu begegnen. Die Integration einer Anwendungserkennung als SDN-Applikation ist ein Ansatz, das existierende Netzwerkmanagement in seinen Aufgaben hinsichtlich Monitoring und Konfiguration zu unterstützen und zu ergänzen.

Im Rahmen dieser Arbeit wurde diese Möglichkeit in Hinblick auf OPC-UA-Kommunikationsbeziehungen untersucht und evaluiert. Die Erkennung erfolgt dabei ohne den Rückgriff auf Portnummern oder die Analyse der Nutzdaten, sondern basiert ausschließlich auf statistischen Paket- und Verbindungseigenschaften. Paketmitschnitte im Rahmen verschiedener Kommunikationsszenarien wurden dazu genutzt, um Kenngrößen zu ermitteln, welche eine Klassifizierung der Verbindungen ermöglichen. Für die (binäre) Klassifikation des Netzwerkverkehrs wurden verschiedene überwachte maschinelle Lernverfahren (ML) trainiert, verglichen und bewertet. Auf Basis dieser Ergebnisse wurde untersucht, welche Merkmale im Rahmen der Klassifizierung geeignet sind und wie die Klassifikationsgüte mit der Paketanzahl korreliert.

Innerhalb des berücksichtigten Anwendungsfelds zeigten die besten Klassifikatoren Erkennungsgenauigkeiten von über 95 %, sowohl in Hinblick auf UA Client/Server, als auch in der vorläufigen Betrachtung von UA PubSub. Die Ergebnisse bestätigen, dass die nutzdatenbasierte und ML-gestützte Klassifizierung ein valider Ansatz ist, um die Erkennung einer Anwendungen zu realisieren.

Abstract

Digital transformation of German manufacturing industries aims for a fusion of production with the latest information and communication technology to enable resource-efficient and flexible production processes. In the context of „Industrie 4.0“ initiative that means manufacturing will be highly individualized and new business models will open up by interconnecting embedded systems and machines as well as integrating cloud solutions into the production process.

OPC Unified Architecture (OPC UA), as standardized communication platform for the industry, ensures the safe, reliable and manufacturer-independent data transport across all levels of automation. Meanwhile introducing SDN-WAN based solutions into existing network architectures will enable dynamic and scalable solution approaches to meet the requirements for high availability and low latencies in data communication. Integrating an application identification system and a traffic classification solution into the SDN infrastructure is an approach to augment and complement the existing network management systems in regards to network monitoring and dynamic configuration.

In this work the potential of traffic classification was explored and evaluated for the communication of OPC UA applications. The detection of such flows was implemented without relying on port and protocol analysis or deep packet inspection (DPI) and is instead solely based on flow and packet based features. Traffic captures of various communication scenarios were used to establish and validate characteristic features for this application, in order to enable flow classifications. Several supervised machine learning methods were trained, compared and scored for binary classification. Based on these results suitable features were identified and the correlation between packet count and classification performance was explored.

Within the considered scope of applications the best built models showed classification accuracies above 95 % for UA client-server communication as well as the provisional consideration of UA PubSub. The results confirm that a payload based and machine learning-aided classification is a valid approach to enable identification of a single application.

1 Einführung

Im Fokus der Entwicklungen der Industrieunternehmen Deutschlands steht gegenwärtig die Flexibilisierung der Produktionsprozesse im Zuge von Industrie 4.0 (I40). Die Einführung der Konzepte des Industrial Internet of Things (IIoT) in das Automationsumfeld führt zur Vernetzung bestehender und neuer Produktionssysteme. Diese cyber-physischen Systeme (CPS) besitzen die Fähigkeit miteinander zu kommunizieren, um eine hoch dynamische Adaption an wechselnde Anforderungen zu realisieren. Durch die Anbindung dieser *smart devices* an die Cloud können Produktionslinien virtualisiert und komplexe Auftragsabwicklungen als Dienste implementiert werden, um eine individualisierte Massenfertigung zu ermöglichen.

Die Transformation der Produktion durch die Vernetzung von Produkten, Geräten, Maschinen und IT-Systemen stellt neue Anforderungen an die Datenübertragung, welche durch klassische Netzwerke nicht mehr erfüllt werden können. Durch die Etablierung von Software-Defined-Networking-Strukturen (SDN) ist es möglich, zeitkritische Industrieanwendungen wie Machine-to-Machine-Kommunikation (M2M) dynamisch bereitzustellen.

Die horizontale und vertikale Integration von Produktionssystemen und -ressourcen verlangt einen herstellerübergreifenden Kommunikationsstandard, der diese Strukturen abbilden kann. Um den vielfältigen Anforderungen der Industrie 4.0 zu begegnen, bietet die OPC Unified Architecture (OPC UA) ein voll vernetztes, objektorientiertes Informationsmodell und Skalierbarkeit von der Feldebene bis in die Cloud. Unabhängig von Hersteller, Betriebssystem und Programmiersprache ermöglicht OPC UA eine sichere und zuverlässige Übertragung auf Basis verschiedener Kommunikationsmodelle.

Die SBSK GmbH & Co. KG ist ein IT-Ingenieurunternehmen, dessen Hauptaufgabenfelder die Beratung, die Planung sowie die Installation komplexer und intelligenter Netzwerke sind. Die Erfahrung dieses Unternehmens umfasst u.a. innovative Lösungen aus den Bereichen der Forschung und Medizintechnik. Derzeit fungiert die SBSK als Projektleiter des Forschungsprojekts MONAT - *modellbasierte und bedarfsgerechte Netzwerkkonfiguration für Netzwerke der Automatisierung und Telekommunikation.* Dieses Projekt zielt auf flexible, softwarebasierte Vernetzungslösungen ab, um eine sichere und latenzarme Kommunikation zu realisieren. Neben der Erfüllung der hohen Anforderungen an Sicherheit, Echtzeitfähigkeit und Zuverlässigkeit in künftigen industriellen Netzen, besteht die Herausforderung darin, die Komplexität solch softwarebasierter Netzwerke bei Einführung und im Betrieb gering zu halten.

Im Rahmen dieses Forschungsprojektes wird ein Modell des SDN als zukunftsweisende Netzwerktechnik erschlossen. Es soll ein SDN entwickelt werden, in dem cloudbasierte Industriesteuerungen, aus dem Rechenzentrum heraus, bis in das industrielle LAN hinein simuliert sowie evaluiert werden können. Durch den Bezug komplexer Steuerungen aus der Cloud entstehen neue Use- und Business-Cases für Industrieunternehmen, dadurch wird der Bedarf nach offenen Schnittstellen größer. Das herstellerübergreifende Kommunikationsprotokoll OPC UA bietet für diese zukünftigen Anforderungen erhebliches Potenzial.

Im Zuge dieser Masterarbeit soll eine Methode zur Identifikation von OPC-UA-Kommunikationsbeziehungen entwickelt werden, welche auf Basis statistischer Verfahren eine binäre Klassifikation von Netzwerkverkehr realisiert. Zu diesem Zweck wird das Kommunikationsverhalten von OPC-UA-Anwendungen in unterschiedlichen Szenarien untersucht, Kenngrößen ermittelt und verschiedene maschinelle Lernverfahren zur Modellbildung eingesetzt. Die Klassifikatoren sollen hinsichtlich der vorliegenden Problemstellung optimiert werden, um das Erkennungsvermögen von OPC-UA-Verbindungen miteinander vergleichen und bewerten zu können.

Die entwickelte Anwendungserkennung soll perspektivisch als SDN-Applikation implementiert werden, um eine dynamische Bereitstellung der QoS für OPC-UA-Kommunikationsbeziehungen zu realisieren.

2 (Theoretische) Grundlagen

Zur Einordnung dieser Arbeit und ihrer potentiellen Integration in eine Netzwerkumgebung mit zentraler, software-definierter Steuerung wird innerhalb dieses Abschnitts zunächst auf die grundlegenden Merkmale der SDN-Architektur eingegangen. Weiterhin werden einige aktuelle und kommende Anwendungsszenarien skizziert und vorgestellt. Im zweiten Abschnitt erfolgt die Beschreibung des industriellen Kommunikationsstandards OPC UA, beginnend mit der Entwicklung des Vorgängers OPC. Darauf aufbauend werden die OPC-UA-Spezifikationen vorgestellt und auf die Eigenschaften und Unterschiede der beiden Kommunikationsmodelle sowie die zum Einsatz kommenden Sicherheitsmechanismen eingegangen.

Abgeschlossen wird das Kapitel mit einem Überblick über etablierte Verfahren zur Klassifikation von Netzwerkverkehr. Beginnend mit der klassischen Auswertung von Portnummern und der darauf basierenden Zuordnung von Anwendungen, wird im weiteren Verlauf auf das Verfahren zur Untersuchung von Nutzdaten eingegangen. Neuere Ansätze betrachten nicht mehr einzelne Pakete, sondern analysieren das grundlegende Kommunikationsverhalten von Anwendungen, um diese anhand typischer Verbindungsmuster zu identifizieren. Abschließend werden statistische Verfahren vorgestellt, welche in Verbindung mit Methoden des maschinellen Lernens (ML) ebenfalls zur Klassifikation von Netzwerkverkehr eingesetzt werden.

2.1 Klassischer Netzaufbau (Industrie)

Die Einführung serieller Feldbussysteme zur Überwindung der Limitierungen durch die parallele Verdrahtung von Sensoren, Aktoren und Controllern versprach einen verringerten Installationsaufwand, höhere Zuverlässigkeit und lieferte die physische Grundlage zur herstellerübergreifenden Kommunikation in Automationsnetzwerken. Auf die Veröffentlichung von *Modbus* im Jahr 1979 folgten zahlreiche offene wie auch proprietäre Standards, darunter *Controller Area Network (CAN)* und *Process Field Bus (Profibus)*. Die steigende Popularität internetbasierter Techniken sorgte um das Jahr 2000 für ein Erstarken von Ethernet auch in Industrienetzwerken. [1]

Die strengen Echtzeit-Anforderungen im Bereich der (untersten) Feldebene (vgl. Abb. 2.1) können durch den Ethernet-Standard nicht erfüllt werden. Um dieser Unzulänglichkeit zu begegnen, wurden eine Reihe unterschiedlicher Implementierungen unter der Bezeichnung Echtzeit-Ethernet (Teil des Industrial Ethernet) entwickelt. Eingeteilt in drei Klassen erreichen die Umsetzungen isochrone Zykluszeiten von 100 ms (Class A) bis unter 1 ms (Class C) und sind somit geeignet, ältere Feldbusse zu ersetzen bzw. zu ergänzen. Speziell die Realisierungen der Klasse C erfordern Modifikationen an der Ethernet-Schicht selbst, so dass einige dieser Verfahren nicht vollumfänglich kompatibel zum Ethernet-Standard sind und entsprechende Implementierungen häufig nicht untereinander kommunizieren können. [1]

Auf Grundlage der ethernetbasierten Standards des Institute of Electrical and Electronics Engineers (IEEE) rund um Audio Video Bridging (AVB) entstand im November 2012 die Time-Sensitive Networking (TSN) Task Group. Deren Ziel ist die Entwicklung eines einheitlichen Standards zur Übertragung von Daten über Ethernet-Netzwerke mit sehr geringen Übertragungslatenzen und hoher Verfügbarkeit. Auf Grundlage von Zeitsynchronisierung, Prioritätsklassen und der Reservierung von Übertragungskapazitäten sollen einzelne Nachrichtenströme mit hohen Echtzeitanforderungen

vom Rest der Netzwerkkommunikation isoliert ausgetauscht und damit Zeitobergrenzen eingehalten werden. [1]

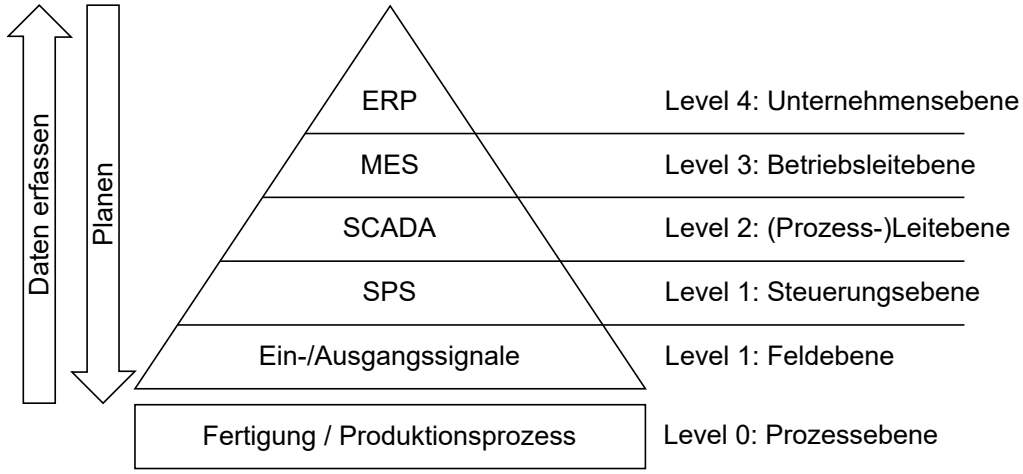


Abb. 2.1: Beispiel einer Automatisierungspyramide (Quelle: wikipedia.org)

> „Die Herausforderung besteht in der vertikalen Integration der Datenhaltung, der Informationsflüsse und der eingesetzten Werkzeuge von der Feldebene bis hin zur Unternehmenssteuerung. Diese Integration muss über den gesamten Lebenszyklus des Automatisierungssystems, vom Engineering über den Betrieb bis zur Instandhaltung und kontinuierlichen Verbesserung gegeben sein.“[2]

Ergänzt wurde der Parallelbetrieb kabelgebundener Feldbus- und Ethernet-Systeme durch die Einführung von Funktechniken, die im lokalen Bereich eine standortunabhängige Kommunikation mit Geräten und Maschinen ermöglichen. Im industriellen Umfeld steht eine hohe Zuverlässigkeit der Kommunikation an oberster Stelle, so dass kabelgebundene Netzwerke weiterhin gegenüber drahtlosen Systemen dominieren. Der Einsatz von Funktechniken beschränkt sich entsprechend auf spezielle Anwendungsfälle, in denen Flexibilität entscheidend ist. [1]

Im Zuge von I40 sollen Aktoren, Sensoren und weitere Geräte in großer Zahl auf Basis IIoT miteinander und mit der Cloud vernetzt werden. Durch die Verknüpfung von Informations- und Kommunikationstechnik tauschen Controller und Maschinen automatisiert und bidirektional Daten aus (M2M), um granulare Produktionsabläufe zu ermöglichen. Auf Grundlage dieser Daten könnten cloudbasierte Analysen in Echtzeit neue Einsichten hinsichtlich Produktivität, Qualitätsoptimierung und Sicherheit liefern. Im Produktionsbereich sollen nach der Vision von *Smart Factory* die Produkte selbst intelligent sein, jederzeit eindeutig identifiziert und lokalisiert werden können sowie Kenntnis über ihren eigenen Zustand besitzen. [3]

2.2 Software Defined Networking

Die Entwicklung und Standardisierung von Network Functions Virtualisation (NFV) resultierte in der Entkopplung von Software und Hardware der Netzdienste und verspricht zahlreiche Vorteile gegenüber proprietären Systemen. Verringerte Gerätekosten, hohe Skalierbarkeit, Anpassung der Netzkonfiguration in Echtzeit und automatisierte Abläufe durch IT-Orchestrierungsmechanismen bilden dabei nur einen Teil der Möglichkeiten. [4]

> „Die dynamische Instanziierung und das Migrieren von Netzfunktionen im Zuge der NFV stellen auch neue Anforderungen an die IP-Transportnetze. Datenpakete bzw. Datenflüsse müssen in Abhängigkeit der Netzsituation (z. B. Verkehrslastspitze) mit dynamisch verlagerten und/oder neu skalierten Netzanwendungen flexibel zu den zuständigen Netzwerkdiensten (z. B. IMS-VNFs) in der NFV-Infrastruktur (zu NFVI-POPs u. a. in verschiedenen Rechenzentren) weitergeleitet werden. Für die Steuerung solcher Datenflüsse wird Software Defined Networking (SDN) als Schlüsseltechnologie angesehen.“[4]

Während der Begriff SDN im Jahr 2009 in einem Artikel des Magazins *MIT Technology Review* [5] geprägt wurde, der Bemühungen rund um ein OpenFlow-Projekt an der Stanford University behandelte [6], reichen grundlegende Bestrebungen dieser Art zurück bis in die frühen 80er Jahre. Techniken mit Ziel einer zentralisierten Steuerungslogik, Programmierbarkeit von Netzwerkelementen und Ressourcen-Virtualisierung existierten bereits lange vor SDN, das die existierenden Ansätze sinnvoll zusammenführte. [6]

Erste Anstrengungen bzgl. einer zentralisierten Netzwerksteuerung unternahm AT&T mit der Einführung des *Network Control Points (NCP)* bereits Anfang der 80er Jahre. Dieses bot eine zentrale Datenbank für Telefonverbindungen und ermöglichte Nebenkanal-Signalisierung zur Kommunikation mit (Karten-)Zahlungsterminals. [7]

Die Idee der Separierung von Control und Data Plane griff das Betriebssystem Berkeley Software Distribution (BSD) in der Version 4.4 für seine *routing sockets* auf, deren Routingtabellen per Kommandozeile oder mithilfe eines Router Daemon-Programms manipuliert werden konnten [8]. Zur selben Zeit verabschiedete die Internet Engineering Task Force (IETF) das Simple Network Management Protocol (SNMP), das die Steuerung und Überwachung von Netzwerkelementen durch eine zentrale Instanz ermöglichte [9]. Es folgte die Entwicklung der Forwarding and Control Element Separation (ForCES), im Zuge derer jedes Control Element ein oder mehrere Forwarding Elements in einem Netzwerk steuert. Control Elements werden ihrerseits durch eine Manager-Instanz gesteuert, um die Skalierbarkeit weiter zu steigern [10].

Konkrete Visionen einer zentral gesteuerten Netzarchitektur in Verbindung mit der Trennung von Control Plane und Data Plane skizzierten das 4D-Projekt [11] bzw. SANE [12].

Bestehende Sicherheitsmaßnahmen, speziell in Unternehmensnetzwerken, setzen sich häufig aus einer Vielzahl simultan aktiver Mechanismen (Firewalls, NATs, VLANs) zusammen, deren komplexes Zusammenspiel einen hohen Konfigurationsaufwand und den Überblick über die Netzwerkstruktur erfordern. Casado et. al schlugen in diesem Zusammenhang eine neuartige *Secure Architecture for the Networked Enterprise (SANE)* auf Basis eines zentralen Domain Controllers (DC) vor, um

Unternehmensnetzwerke topologieunabhängig und unkompliziert abzusichern.

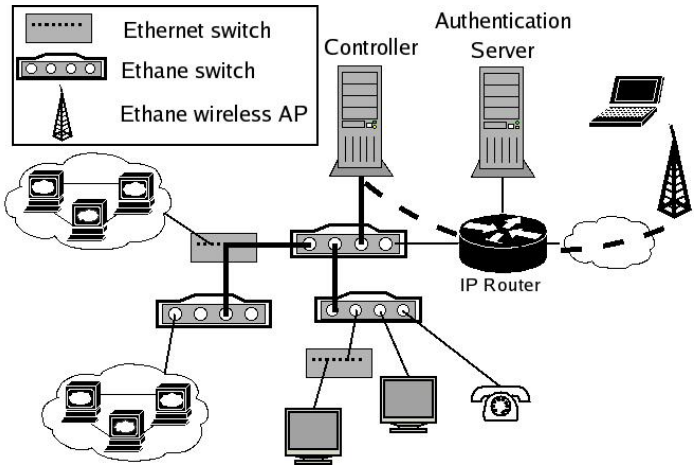


Abb. 2.2: Exemplarischer Einsatz von Ethane (Quelle: [13])

Einen ähnlichen Ansatz verfolgen Greenberg et. al mit 4D, dessen Architektur die vollständige Trennung der Steuerungslogik im Netzwerk von den unterliegenden Protokollen fordert. Ausgehend von einer logisch zentralisierten Einheit wird das Netzwerk gesteuert und in vier Ebenen aufgeteilt. Die Entscheidungsebene (Decision Plane) trifft alle Entscheidungen hinsichtlich der Netzwerksteuerung, einschließlich Erreichbarkeit, Lastverteilung, Zugriffskontrolle und Sicherheit. Die Kommunikation zwischen Netzelementen und Entscheidungselementen findet über die Verteilungsebene (Dissemination Plane) statt. Die Ermittlung der physischen Komponenten im Netzwerk und die Kommunikation dieser Komponenten untereinander wird durch die Entdeckungsebene (Discovery Plane) realisiert. Schließlich erfolgt die Verarbeitung von Datenpaketen auf Basis der Vorgaben der Entscheidungsebene im Bereich der Datenebene (Data Plane).

Während 4D eine abstrakte Forschungsagenda formulierte, nutzte Ethane [13] die Vision als Grundlage und erweckte sie zum Leben, indem viele wichtige Details und Abläufe konkretisiert wurden. Abb. 2.2 zeigt u.a. den durch Ethane eingeführten Controller als zentrale Steuerungskomponente und dessen Kommunikation mit flowbasierten Ethane-Switches zur Verteilung und Anwendung von Richtlinien im Netzwerk. Diese Schnittstelle zwischen Control Plane und Data Plane beeinflusste maßgeblich die Entwicklung des Kommunikationsprotokolls OpenFlow und somit letztendlich die Entstehung von SDN.

2.2.1 Überblick

Im Vergleich zu vielen Ansätzen, denen der „Sprung“ aus dem akademischen Umfeld in die praktische Anwendung nicht gelang, schaffte es das SDN-Konzept in der Industrie Fuß zu fassen. Immer mehr Hersteller von Switches gingen dazu über, das OpenFlow Application Programming Interface (API) in ihren Geräten zu unterstützen. Im Jahr 2011 schlossen sich namhafte Unternehmen, darunter Facebook, Google, Microsoft und die Deutsche Telekom, zusammen und gründeten die Open Network Foundation (ONF). Das Ziel der Organisation besteht darin, die Entwicklungen rund um SDN voranzutreiben und die Adaption der Architektur durch offene Standards zu fördern.

Im Wesentlichen wird das SDN-Paradigma durch vier Grundsätze definiert [4]:

- Separierung von Control Plane und Data Plane
- zentrale, logische Steuerung
- offene Schnittstellen
- Programmierbarkeit

Die Trennung von Signalisierung (Control Plane) und der Weiterleitung (Forwarding) von Nutzdaten (Data Plane) bricht den bisher monolithischen Aufbau von Netzelementen wie Switches und Routern auf. Während diese Netzwerkgeräte bisher Steuerung und Paketverarbeitung vereinen, entkoppelt SDN die beiden Komponenten, indem die Steuerungslogik zentralisiert in den SDN-Controller verlagert und der Funktionsumfang von SDN-Switches auf das Forwarding von Datenpaketen reduziert wird. Durch die Zentralisierung der Netzsteuerung im SDN-Controller werden effektive Routing-Entscheidungen möglich. [4]

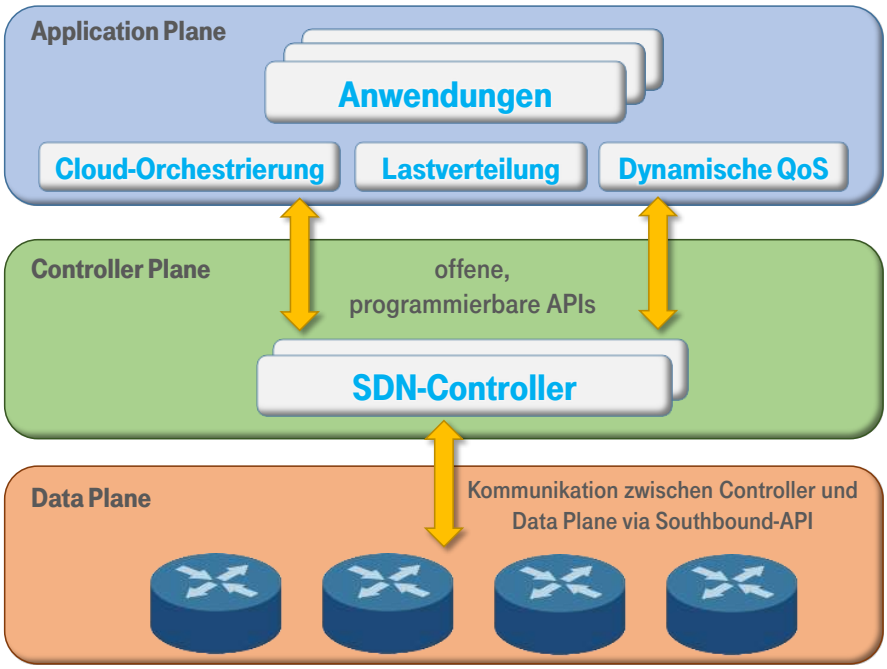


Abb. 2.3: Grundlegende SDN-Komponenten (Quelle: [4], eigene Darstellung)

Die grundlegende SDN-Architektur besteht aus drei logischen Ebenen, die in Abb. 2.3 dargestellt sind. Die Kommunikation zwischen Control und Data Plane geschieht über das Southbound-API und ein geeignetes Protokoll, wie beispielsweise das von der ONF spezifizierte OpenFlow-Protokoll. Das Verhalten der Elemente der Data Plane wird durch Flow Tables gesteuert, die Verbindungen anhand einer Reihe von Header-Feldern identifizieren und eine zugehörige Aktion (Forward, Drop, EnQueue oder Modify) definieren. Die Routing-Entscheidungen trifft der SDN-Controller, indem er die abstrakten Anforderungen der SDN-Applikationen in konkrete Netzwerkrichtlinien übersetzt und der Data Plane mitteilt. [4]

Das Leistungsspektrum solcher Netzdienste reicht von der Priorisierung von Netzverkehr, über

Zugangskontrolle und Bandbreitenverwaltung bis zur Gewährleistung der Dienstgüte. Die Programmierung des SDN-Controllers durch die Application Plane kann zur Laufzeit erfolgen, so dass dynamische Anpassungen des Netzes an sich verändernde Anforderungen möglich sind. Dieser Wandel der Netzwerkarchitektur reduziert die Abhängigkeit von proprietären Geräten und sorgt für sinkende Kosten durch den Einsatz von Standard-Hardware. [4]

> „Das SDN-Konzept ermöglicht Netzadministratoren mittels auch selbst entwickelter SDN-Programme das Transportnetz flexibel und dynamisch von einem zentralen Punkt aus zu konfigurieren, zu managen, für Sicherheit zu sorgen und den Einsatz der Netz-Ressourcen zu optimieren.“[4]

2.2.2 Anwendungsfälle

Der SDN-Ansatz findet aufgrund seines Potenzials bereits Anwendung in zahlreichen Umgebungen und Einsatzszenarien. Speziell in Netzwerken von Rechenzentren und Campusumgebungen wird die Möglichkeit der zentralisierten Steuerung und die Programmierbarkeit der Data Plane im laufenden Betrieb bereits verstärkt genutzt. Ebenso werden experimentelle Architekturen für zukünftige Szenarien in Testnetzen mit dem Ziel einer verbesserten Nutzererfahrung (Quality of Experience) und Dienstgüte (Quality of Service) erprobt. Die Betreiber der Mobilfunknetze erwarten von der Integration der SDN-Architektur in ihre Netze eine sinkende Komplexität und gleichzeitig das Potenzial, neue Dienste bereitstellen zu können. [6]

Im Bereich der Mobilfunknetze ermöglicht eine programmierbare Data Plane, das Routing vollständig auf den unteren beiden OSI-Schichten zu realisieren und den Overhead gegenüber einer Lösung in der Vermittlungsschicht zu minimieren [14, 15]. In Hinblick auf aufkommende 5G-Kommunikation führt die Integration der SDN-Architektur in das existierende Netz zu einer größeren Flexibilität und ermöglicht die Bereitstellung von Echtzeit-Dienstleistungen [16].

In Hinblick auf drahtlose lokale Netzwerke birgt die Einführung von SDN das Potenzial zur Flexibilisierung des Routings und der Weiterleitung von Datenverkehr mit dem Ziel einer verbesserten Leistungsfähigkeit (der mobilen Kommunikation) sowie die Einführung neuer Anwendungen und Dienste. Konkret ist die Optimierung der Kanalzuweisung in WiFi-Umgebungen denkbar, indem eine Kommunikation zwischen Access-Points ermöglicht wird, um eine erweiterte Abdeckung bei wechselnder Nutzerauslastung zu erreichen. [17]

Der Einsatz der SDN-Architektur für die Kommunikation im IoT verspricht eine steigende Skalierbarkeit und ein selbstverwaltetes Routing, indem der Verwaltungsaufwand reduziert und energieeffiziente Steuerungsabläufe eingeführt werden.[18]

In Datacentern ist eine optimale Steuerung des Datenverkehrs notwendig, die das SDN-Grundgerüst durch die Reduzierung von Latenzen und verbesserte Ressourcennutzung bei verringerten Betriebskosten bereitstellen kann. Insbesondere hinsichtlich der Skalierbarkeit, der automatisierten Lastverteilung sowie der Energieeffizienz lassen sich in diesem Umfeld Verbesserungen erzielen. [19, 20]

Campus- und Backbone-Netzwerke haben häufig mit schwankender lokaler Auslastung zu kämpfen, der durch effektive Netzwerkverkehrsüberwachung in Echtzeit und daraus abgeleitete Lastverteilung begegnet werden kann [6].

Leitungsvermittelnde Netzwerke auf optischer Basis bieten sich laut Das et al.[21] aufgrund ihrer besseren Skalierbarkeit und höheren Übertragungsraten bei gleichzeitig geringerem Energieverbrauch gegenüber paketvermittelnder Technik für den Einsatz im Core-Netz an. Gleichzeitig unterstützt der leitungsvermittelnde Ansatz kein asynchrones Zeitmultiplex, welches im Bereich der Netzwerkübergänge überlegen ist und verfügt nicht über die granularen Steuerungsmöglichkeiten der Paketvermittlung. Die Vereinheitlichung des Betriebs, der Steuerung und der Verwaltung paket- und leitungsvermittelnde Netzwerke eröffnet signifikante Vorteile im Core-Netz hinsichtlich Kosten- und Energieeffizienz sowie der erzielbaren Leistungsfähigkeit [22]. Auf Grundlage von OpenFlow präsentieren Gudla et al.[23] einen Ansatz für die Integration heterogener Netzwerke, welcher die Stärken beider Vermittlungstypen vereint, indem die zentralisierte Steuerung abstrahiert von der unterliegenden Hardware erfolgt.

2.3 OPC Unified Architecture

Mit der Verabschiedung der standardisierten Software-Schnittstelle Open Platform Communications (OPC, ehemals: OLE for Process Control) wurde ein Möglichkeit zur Kommunikation zwischen Geräten und Anwendungen verschiedener Hersteller der Automatisierungstechnik geschaffen. Gut zehn Jahre nach der Veröffentlichung von OPC führte der Wunsch der Hersteller nach einer plattformunabhängigen und technisch zeitgemäßen Lösung zur Verabschiedung der OPC Unified Architecture (OPC UA). Zu den Hauptmerkmalen von OPC UA zählen ein eigener Kommunikationsstack, der das ursprüngliche COM/DCOM-System ersetzt, ein umfangreiches Informationsmodell und der Einsatz zeitgemäßer Sicherheitslösungen. [24]

2.3.1 Classic OPC

Mit dem Aufkommen softwarebasierter Automatisierungssysteme im industriellen Umfeld erfuhren vor allem Windows-Systeme einen starken Anstieg in der Nutzung zum Zweck der Datenvisualisierung und für Steuerungsaufgaben. Häufig gestaltete sich der herstellerübergreifende Zugriff auf Automatisierungsdaten problematisch, da in den verwendeten Geräten eine Vielzahl verschiedener Bus-Systeme, Schnittstellen und Protokolle zum Einsatz kamen und aufwändige Anpassungen zur Interoperabilität notwendig waren. Als Reaktion schlossen sich im Jahr 1995 große Firmen der Automatisierungsindustrie zur OPC Task Force zusammen, um eine universelle Schnittstelle zum Datenaustausch zu schaffen. [24]

Das Ergebnis der Arbeit war die Veröffentlichung der OPC Data Access-Spezifikation (OPC DA) bereits ein Jahr später und die weitere Entwicklung durch die im gleichen Jahr gegründete OPC Foundation. Möglich wurde die schnelle Verabschiedung des Standards durch den Fokus Kernfunktionalitäten und die Microsoft Windows-Technologien *Component Objekt Model* (COM) und *Distributed COM* (DCOM) als Grundlage. Wenig später folgte die zweite Version von OPC DA, sowie die Entwicklung zwei weiterer OPC-Spezifikationen: *Alarm & Events (OPC A&E)* und *Historical Data Access (OPC HDA)*. [24]

Während der Zugriff auf aktuelle Prozessdaten durch die OPC-DA-Spezifikation abdeckt ist, beschreibt OPC A&E eine Schnittstelle für ereignisgesteuerte Informationen und OPC HDA stellt Funktionen für den Zugriff auf archivierte Daten zur Verfügung. [24]

Für den Informationsaustausch nutzt OPC das Client-Server-Modell. Der Server stellt die Quelle für Prozessinformationen als eigenes Gerät dar, über dessen Schnittstelle der Client auf die bereitgestellten Daten zugreifen kann. In Abb. 2.4 ist der dominierende Anwendungsfall für OPC dargestellt, der im Datenzugriff durch Benutzerschnittstellen (HMI) oder *Supervisory Control and Data Aquisition*-Systeme (SCADA) auf Automatisierungsgeräte verschiedener Hersteller über eine definierte Software-Schnittstelle bestand. [24]

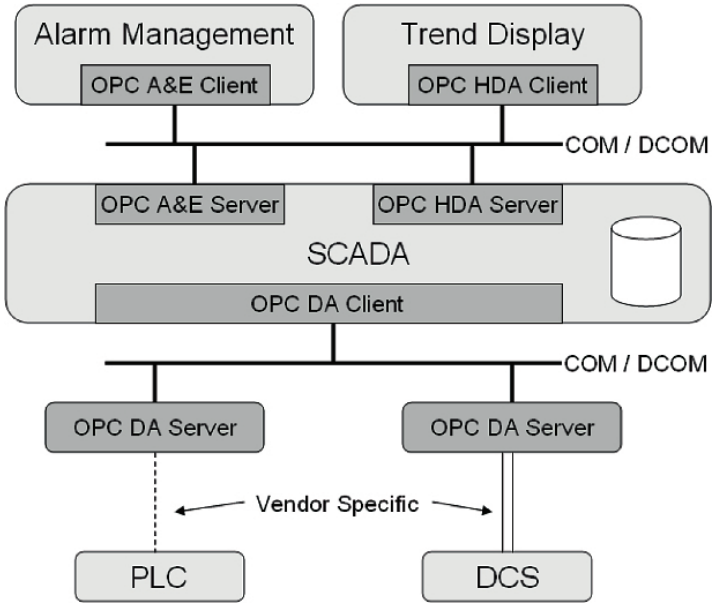


Abb. 2.4: Typischer Anwendungsfall von OPC Clients und Servern (Quelle: [24])

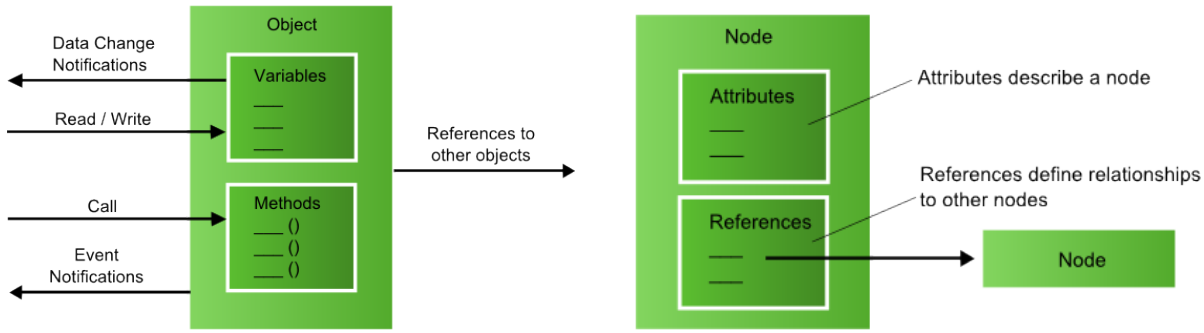
Zwar entwickelte sich OPC auf Basis von COM/DCOM schnell zum Standard zur herstellerunabhängigen Kommunikation in der Automatisierungstechnik, hatte aber auch mit Designschwächen zu kämpfen. Einerseits verhinderte die Plattformabhängigkeit den Einsatz von OPC in Bereichen, in denen keine Windows-Systeme eingesetzt werden konnten. Gleichzeitig waren die internen Abläufe von COM/DCOM nicht nachvollziehbar, so dass Entwickler im Fehlerfall keine Untersuchung bzw. Anpassung am Quellcode vornehmen konnten. Weiterhin traten häufig Konfigurationsprobleme auf und das Sicherheitskonzept galt als unzureichend. [25]

Der erste Versuch einer plattformunabhängigen Spezifikation wurde im Jahr 2003 in Form von OPC XML-DA veröffentlicht, die den altgedienten COM/DCOM-Ansatz durch HTTP/SOAP und Webservice-Technologien ersetzen sollte. OPC XML-DA war in Hinblick auf den Datenzugriff über das Internet und unternehmensübergreifende Integration entwickelt worden. Aufgrund des hohen Ressourcenverbrauchs und der im Vergleich geringeren Performanz konnte sich der Ansatz aber nicht durchsetzen. [24]

Neben der Plattformabhängigkeit waren es vor allem Bestrebungen der Mitgliederunternehmen der OPC Foundation Limitierungen von Classic OPC zu überwinden und komplexere Informationen und Systeme einbinden zu können. Der Wunsch nach einem ebenbürtigen Nachfolger aller existierenden COM-basierten Spezifikationen, ohne den Verlust grundlegender Eigenschaften und der Leistungsfähigkeit, führte zur Entwicklung von OPC UA.

Die fundamentalen Komponenten von OPC UA sind die unterschiedlichen Transportmechanismen und die Datenmodellierung. Für den Transport sind eine Reihe von Mechanismen definiert, die für verschiedene Anwendungsfälle optimiert sind. [24]

Für interne Netzwerke existiert ein optimiertes Binärprotokoll auf TCP-Basis, das sich durch eine hohe Leistungsfähigkeit, einen geringen Ressourcenverbrauch und die beste Interoperabilität auszeichnet. Alternativ stehen für die Kommunikation über das Internet etablierte Standards wie z.B. Webservices (WS), Extensible Markup Language (XML) und HTTP zur Verfügung, die sich als unproblematisch im Umgang mit Firewalls darstellen. Das abstrakte Kommunikationsmodell ist nicht auf ein spezifisches Protokoll angewiesen, wodurch in Zukunft auch der Einsatz neuer Protokolle möglich ist. [24]



(a) OPC-UA-Objektmodell (b) Knoten und Referenzen

Abb. 2.5: Konzept des OPC-UA-Adressraums (Quelle: [26])

Reale Produkte und Produktionsschritte werden im Rahmen der Datenmodellierung durch virtuelle Objekte repräsentiert, welche beliebig komplexe Informationsinhalte abbilden. Die Hauptaufgabe des OPC-UA-Adressraums ist die Bereitstellung eines standardisierten Weges, um die Objekte innerhalb der Server für Clients abzubilden. Das OPC-UA-Objektmodell, dargestellt in Abb. 2.5(a) wurde entsprechend dieser Anforderung entwickelt und definiert Objekte anhand von Variablen und Methoden. Variablen können ausgelesen und verändert werden und es besteht die Möglichkeit, Methoden zu definieren und diese auszuführen. Weiterhin ist es möglich, Objekte zu typisieren, indem sie einer bestehenden oder neu definierten Objektklasse zugeordnet werden, und Beziehungen zwischen ihnen auszudrücken. [24]

Ein Satz an Objekten und der zugehörigen Informationen, welche der Server den Clients zugänglich macht, bilden seinen Adressraum. Innerhalb des Adressraums werden Objekte und ihre Komponenten durch Nodes (Knoten) repräsentiert, welche anhand ihrer Attribute beschrieben werden und über Referenzen miteinander vernetzt sind (Abb. 2.5(b)). Knoten und Referenzen bilden das Grundgerüst des OPC-UA-Datenmodells, welches beliebig komplexe, objektorientierte Erweiterungen ermöglicht, ohne dadurch die Interoperabilität zu beeinträchtigen. [24]

2.3.2 Spezifikationen

Die OPC-UA-Spezifikation umfasst insgesamt 14 Teilen, die sich in zwei Gruppen einteilen lassen und in Abb. 2.6 aufgelistet sind (Part 14: PubSub fehlt in dieser Darstellung). Die „Kern“-Spezifikationen definieren die Grundlage der OPC-UA-Architektur, während sich die zweite Grup-

pe mit den unterschiedlichen Varianten des Datenzugriffs beschäftigt und das Informationsmodell von OPC UA beschreibt. [24]

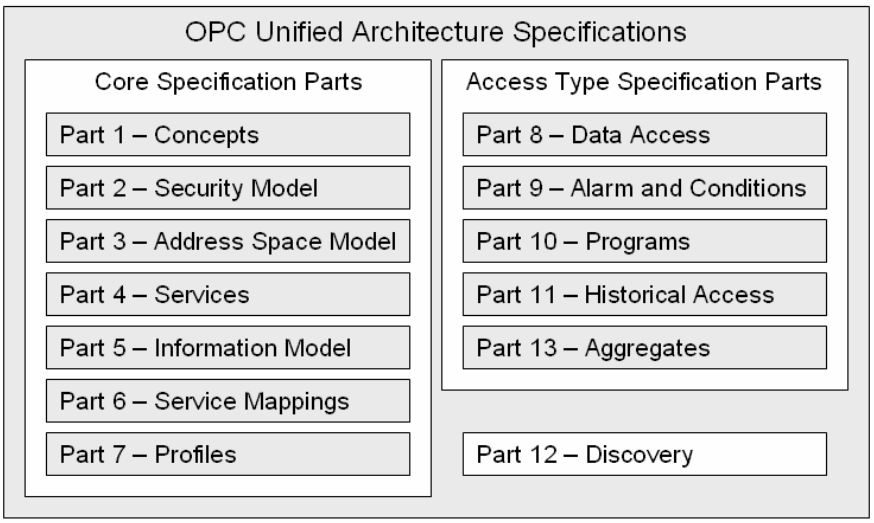


Abb. 2.6: Übersicht über die OPC UA-Spezifikationen (noch ohne PubSub) (Quelle: [24])

Die ersten zwei Teile sind rein informativ. Der Konzeptteil [UA Part 1] gibt einen allgemeinen Überblick über OPC UA, während [UA Part 2] die Sicherheitsanforderungen und das Sicherheitsmodell von OPC UA (oberflächlich) beschreibt und grundlegende Begriffe in diesem Zusammenhang definiert.

Wesentlich für das Verständnis der Modellierung und den Zugriff auf Daten sind [UA Part 3] und [UA Part 4]. Speziell für Entwurf und Entwicklung von OPC-UA-Anwendungen stellen diese beiden Spezifikationen die Schlüsseldokumente dar. Das *Address Space Model* liefert eine detaillierte Beschreibung des Adressraums innerhalb eines OPC-UA-Servers und den Zugriff der -Clients. Das dem Adressraum zugrunde liegende Meta Modell spezifiziert die Regeln und Grundbausteine, um ein Informationsmodell bereitzustellen.

Der Abschnitt *Services* definiert insgesamt neun *Basis Service Sets*, die mögliche Interaktionen zwischen UA Client- und Server-Anwendungen beschreiben. Diese erlauben u.a. die Navigation durch den Adressraum, das Lesen und Beschreiben von Variablen sowie die Anmeldung für Datenänderungen und Events.

Das Informationsmodell ermöglicht die Flexibilität von OPC UA, indem ausgehend von einem Basismodell beliebig komplexe, objektorientierte Erweiterungen realisiert werden können, während die Interoperabilität stets gewährleistet ist.

Der Abschnitt *Service Mappings* beschreibt das Abbilden von UA Services auf Nachrichten, dabei zum Einsatz kommende Sicherheitsmechanismen und den Transport der Nachrichten.

Abgeschlossen wird der konzeptionelle Teil mit der Beschreibung von Profilen, die sinnvolle Untergruppen von OPC-UA-Funktionen definieren. Diese bilden Kategorien für Verhaltensweisen, unterstützte Funktionen und Sicherheitsmerkmale von OPC UA Clients und Servern ab, mit dem Ziel OPC-UA-Produkte zuverlässig und reproduzierbar testen und verifizieren zu können.

[UA Part 8] und [UA Part 9] spezifizieren den Zugriff auf aktuelle Prozessinformationen bzw. auf ereignisbasierte Daten. Die Ausführung, Anpassung und Überwachung von Programmen (komplexere Abläufe mit längerer Laufzeit und zustandsgesteuerter Funktionsweise) werden in [UA Part 10] festgelegt, während [UA Part 11] den Zugriff auf archivierte Daten beschreibt.

[UA Part 13] definiert die Umwandlung von Rohdaten in kumulierte Werte, die sowohl zur Überwachung aktueller Prozessdaten, als auch zur Archivierung genutzt werden.

Die Abläufe, wie ein Client im Netzwerk befindliche Server finden kann und wie er die notwendigen Informationen für einen Verbindungsaufbau erhält, beschreibt [UA Part 12].

Die im Februar 2018 veröffentliche Spezifikation [UA Part 14] definiert ein neues Kommunikationsmodel, welches auf dem Publish-Subscribe-Schema aufbaut und das bekannte Client-Server-Schema ergänzt. [27]

2.3.3 UA Client/Server

OPC UA ist eine serviceorientierte Architektur (SOA), die auf einem asynchrones Anfrage-Antwort-Schema aufbaut. Zu den wichtigsten Mechanismen im Rahmen des Datenaustauschs gehören die Dienstaufrufe *Read* und *Write*. Neben dem Auslesen und (Über-)Schreiben von Variablenwerten, sind diese Operationen ebenso auf die Eigenschaften von Knoten *(Nodes)* anwendbar und ermöglichen so den Zugriff auf die Metadaten im Adressraum. [24]

Für die zuverlässige Kommunikation auf Netzebene stehen im Kontext von OPC UA zwei TCP/IP-basierte Protokolle bzw. „protocol bindings“ zur Verfügung: *UA TCP* und *XML/SOAP*. In Abb. 2.7 sind neben diesen beiden Protokollen noch Mischvarianten dargestellt, die eine Kombination aus effizienter Kodierung und unkompliziertem Transport realisieren. [24]

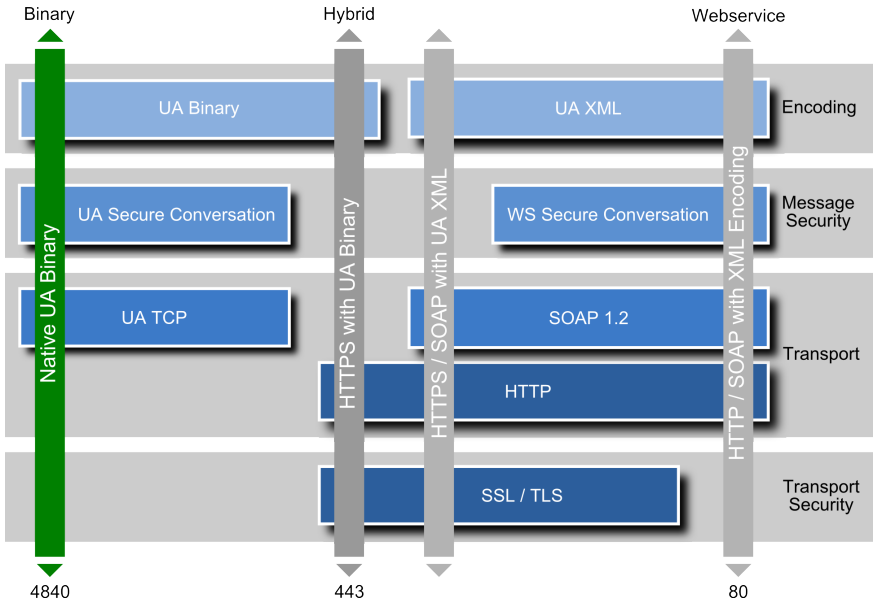


Abb. 2.7: TCP-basierte Protokollvarianten zur Kommunikation (Quelle: [28])

Mit Fokus auf einer schnellen und unkomplizierten Kommunikation wurde das Transportprotokoll UA TCP entwickelt, welches auf eine Binärkodierung (UA-Binary) der Daten setzt. Die (De-

)Serialisierung von Service-Parametern in einen binären Datenstrom stellt den effektivsten Weg des Datenaustauschs dar und wurde in Hinblick auf performancekritische Anwendungen im industriellen Bereich entwickelt. [24]

Die Entscheidung, ein einfaches Protokoll zu definieren, das auf TCP aufsetzt, hatte mehrere Gründe:

- die Notwendigkeit, auf Anwendungsebene Puffergrößen für den Datenversand und -empfang konfigurieren zu können,
- die Möglichkeit, unterschiedlichen Endpunkten eines OPC UA Servers dieselbe Kombination aus IP-Adresse und Port zuweisen zu können,
- die Fähigkeit, auf Transportebene auf auftretende Fehler reagieren und die Verbindung unter Umständen aufrechterhalten zu können.

Eine UA TCP-Nachricht besteht aus einem Message Header und einem Message Body. Der Header liefert Informationen über Typ und Länge der Nachricht. Der Message Body enthält entweder die codierte und geschützte Service-Nachricht, die an die Anwendungsschicht weitergeleitet wird oder eine UA TCP-spezifische Nachricht, die dem Aufbau einer Verbindung oder dem Austausch von Informationen zu Verbindungsfehlern dient. [24]

Es existieren drei UA TCP-Nachrichtentypen:

- Die initiale Hello-Nachricht dient als Anfrage des Client an den Server, um eine Verbindung zu einem spezifischen Endpunkt innerhalb des Servers herzustellen. Im Zuge dessen werden Parameter wie Puffergrößen und die maximale Nachrichtenlänge angefragt.
- Auf ein eintreffendes Hello antwortet der Server mit einem Acknowledgement, in dem die vorgeschlagenen Parameterwerte bestätigt oder angepasst werden. Dies dient der Verlässlichkeit aus sicherheitstechnischer Sicht, da Client und Server wissen, ob eintreffende Nachrichten valide sind. Angriffen in Form von Pufferüberläufen (buffer overflow) oder Denial-of-Service-Attacken (DoS) kann so widerstanden werden.
- Der dritte Nachrichtentyp ist eine Fehlernachricht, die Informationen zur Art des Fehlers an die Gegenseite bereitstellt. Beispielsweise wenn eine Nachricht aufgrund ihrer Größe nicht verarbeitet werden kann oder der Server überlastet ist. Darüber hinaus bietet UA TCP einen speziellen Wiederherstellungsmechanismus, der es einer OPC-UA-Sitzung unter Umständen ermöglicht, eine Verbindungsunterbrechung zu überstehen. Dazu wird bei Verbindungsverlust auf Seite des Clients ein neuer Socket erzeugt, der, nach erneuter Authentifizierung, dem vorhandenen Secure Channel zugewiesen wird. Schwebende Anfragen und Antworten werden zwischengespeichert, bis ein neuer Socket verfügbar und die Verbindung wiederhergestellt ist.

Für die webbasierte Kommunikation wird SOAP (ursprünglich für *Simple Object Access Protocol*) eingesetzt.

SOAP ist ein vom World Wide Web Consortiums (W3C) standardisiertes Netzwerkprotokoll, das im industriellen Bereich dem Austausch von Daten zwischen Systemen sowie der Ausführung sogenannter *Remote Procedure Calls*, also dem Funktionsaufruf in fremden Adressräumen dient. Zur

Darstellung von Informationen wird XML genutzt, während für den Transport meist auf die Kombination aus TCP und HTTP gesetzt wird. Durch den Einsatz von HTTP kann auf den Standardport 80 zugegriffen werden, was die Kommunikation durch Firewalls vereinfacht. Die Übertragung mit HTTPS samt SSL/TLS-Verschlüsselung ist ebenfalls möglich, erzeugt aber unnötigen Overhead sofern bereits Secure Channels zum Einsatz kommen. [24]

Im Gegensatz zur Binärvariante bringt SOAP aufgrund der XML-Basis einen größeren Overhead und einen höheren Ressourcenverbrauch (XML-Parser, HTTP) mit sich, so dass dieser Protokollstack für eingebettete Systeme nur eingeschränkt geeignet ist. [1]

Eine Mischvariante stellt die Kombination aus UA-Binary über HTTPS dar, die die Vorteile beider Protokolle vereint. Durch den Einsatz der Binärkodierung fällt der Overhead gegenüber XML-SOAP geringer aus, während durch das Aufsetzen auf HTTPS eine konfigurationsfreie Kommunikation über den Standardport 443 möglich ist.

Nach Einschätzung der Entwickler wird der Großteil OPC-UA-basierter Produkte das Binärprotokoll zur Kommunikation nutzen, während die Hybridlösung nur in Ausnahmefällen (Internetkommunikation) eingesetzt werden wird. [28]

> „Es wird erwartet, dass die meisten Produkte mit dem effizienten Binärprotokoll kommunizieren werden und nur in Sonderfällen auf das Hybridprotokoll zurückgegriffen wird, beispielsweise wenn über das Internet kommuniziert werden soll und nur ein Port (443) geöffnet werden darf. Die Webserviceimplementierung bleibt Anwendungen vorbehalten, die über ausreichend Ressourcen verfügen und die zwingend Webservices benötigen.“[28]

2.3.4 UA Publish/Subscribe

Mit der finalen Freigabe der UA PubSub-Spezifikation (Teil 14) im Februar 2018 erweiterte die OPC Foundation den bevorzugten Einsatzbereich von OPC UA um vielfältige weitergehende Anwendungsfälle. Dabei reicht das Einsatzspektrum von der latenzarmen und fehlertoleranten Kommunikation in lokalen Netzen in Verbindung mit UDP bis zum sicheren Datenaustausch über öffentliche Weitverkehrsnetze für den Einsatz in hoch skalierbaren Cloud-Anwendungen. [27]

Bereits das Client-Server-Modell in OPC UA bietet einem Client neben dem bekannten Lesen/Schreiben-Mechanismus die Möglichkeit, Informationen unterschiedlichen Typs von einem OPC-UA-Server zu abonnieren. Dazu gehören sich ändernde Datenwerte, Ereignisse und kumulierte Informationen. Nachdem der Client dem Server mitgeteilt hat, welche Objekte und zugehörige Variablen überwacht werden sollen, wird dieser im definierten Intervall durch den Server benachrichtigt. Allerdings skaliert dieser Ansatz schlecht, da jede Subscription eine Verbindung zwischen Client und Server voraussetzt. [27]

Im Publish-Subscribe-Modell verteilen Publisher (Datenquellen) ausgewählte Daten an eine beliebig große Anzahl von Subscribern (Datensenke), die zuvor ihr Interesse an spezifischen Informationen signalisiert haben. Nachdem sich ein Subscriber für ein Thema (Topic) registriert hat, bekommt

[1] Diese Variante wird nicht länger unterstützt, da sich die zugrundeliegende Sicherheitsspezifikation für WebServices *WS-SecureConversation* im industriellen Umfeld nicht durchgesetzt hat. [29](S. 59)

dieser die gewünschten Daten in einem festen Zeitintervall wiederholt zugesendet, ohne dass Publisher und Subscriber von der Existenz der Gegenseite wissen müssen. Durch den asynchronen Aufbau des PubSub-Modells sind entsprechende Anwendungen in ihren Rollen als Publisher und Subscriber voneinander entkoppelt, sodass beliebig viele Subscriber Nachrichten desselben Publishers empfangen können, ohne dass dieser dadurch beeinträchtigt wird. Entsprechend eignet sich dieses Kommunikationsmodell für Anwendungen, die besondere Anforderungen hinsichtlich der Skalierbarkeit und/oder der Unabhängigkeit vom Standort stellen. [27]

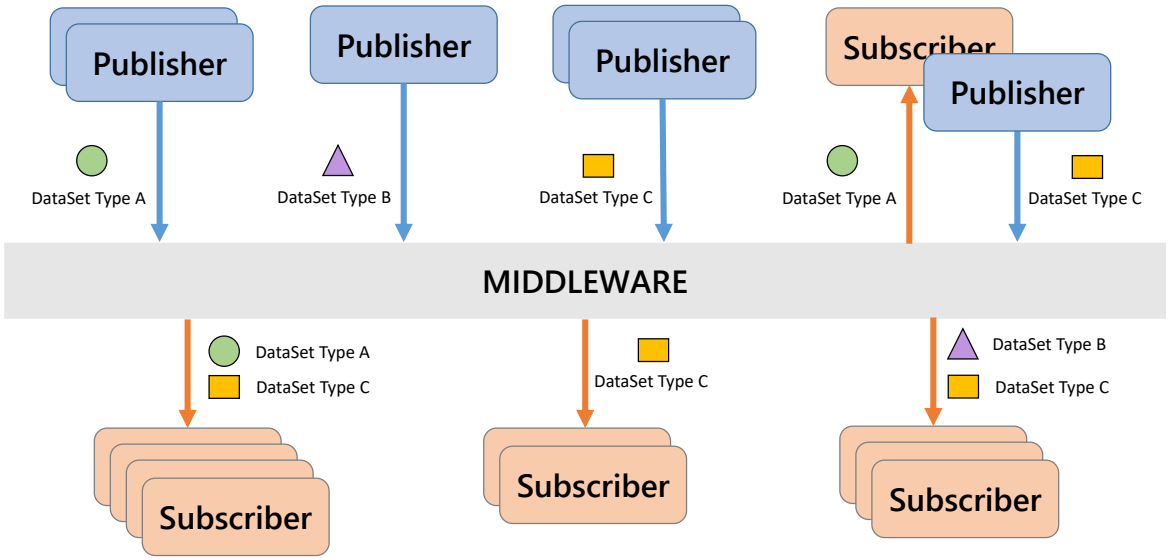


Abb. 2.8: Übersicht über das Publish/Subscribe-Modell (Quelle: [27], eigene Darstellung)

Als Vermittler zwischen Sender und Empfänger (Bereitsteller und Konsument) dient eine nachrichtenorientierte Middleware (MOM), die als soft- oder hardwarebasierte Instanz den Austausch zwischen den Endpunkten realisiert. Abb. 2.8 zeigt, dass Publisher und Subscriber nur mit der MOM interagieren und keine direkte Kommunikation zwischen zwei Endpunkten besteht. PubSub wurde mit Blick auf Flexibilität entwickelt und ist dadurch an keinen konkreten Mechanismus zum Nachrichtenaustausch gebunden. Aufbauend auf einer Reihe abstrakter Schichten kann PubSub verschiedene Nachrichtentypen, die unterschiedliche Anforderungen an die Übertragung stellen, effizient übermitteln. [27]

- Der Transport basierend auf UDP stellt eine Möglichkeit der Nachrichtenübermittlung dar und eignet sich für die zyklische Übertragung kleiner Datenmengen in One-to-One- oder One-to-Many-Konfigurationen. Speziell in lokalen Netzen und in Kombination mit den Erweiterungen des Bridging-Standards IEEE 802.1Q in Gestalt von Time-Sensitive Networking (TSN) wird ein fehlertoleranter Nachrichtenaustausch mit sehr geringen Latenzen und festen Zeitobergrenzen ermöglicht. Die Verteilung der Nachrichten ist in diesem Fall durch die Netzinfrastruktur (z.B. durch Router oder Switches) und den Einsatz von Multicast realisiert, um Verarbeitungsdauer und Overhead gering zu halten.

- Für den Einsatz in Wide Area Networks (WAN), wie dem Internet, ist eine brokerbasierte Verteilung von Nachrichten spezifiziert, die auf die Cloud-Protokolle *Advanced Message Queuing Protocol (AMQP)* oder *Message Queuing Telemetry Transport (MQTT)* setzt. Für

die Integration in die Cloud kombiniert PubSub diese Protokolle mit wesentlichen OPC-UA-Merkmalen, wie der Ende-zu-Ende-Sicherheit und der standardisierten Datenmodellierung.

Da sowohl UA Publisher/Subscriber, als auch UA Client/Server auf dem OPC-UA-Informationsmodell basieren, ist die Integration von PubSub in existierende OPC-UA-Clients und -Server mit geringem Aufwand möglich. Typischerweise wird ein Publisher durch einen Server repräsentiert und ein Subscriber durch einen Client. Es besteht aber keine Rollenabhängigkeit, so dass die Verteilung auch umgekehrt ausfallen kann oder gar eine völlige Entkopplung der PubSub-Kommunikation von OPC UA Client/Server-Anwendungen realisierbar ist. [27]

„Konkret erweitert PubSub die Nutzbarkeit von OPC UA etwa in der Kleinststeuerungen, Sensoren und leistungsschwachen eingebetteten Systemen, die in der Regel eine optimierte Kommunikation mit geringer Latenz in lokalen Netzwerken erfordern. Erste PubSub Implementierungen konnten laut Stefan Hoppe, Vice President der OPC Foundation, in 8-Bit-Controllern mit nur 2kByte SRAM realisiert werden. Andererseits ermögliche PubSub auch den Einsatz von OPC UA in hoch skalierbaren Cloud-basierten Anwendungen, bei denen eine nahezu unbegrenzte Anzahl von Edge-Datenquellen wie zum Beispiel Sensoren ihre Daten sicher an Internet-basierte Broker-Anwendungen liefern kann.“[30]

2.3.5 UA Security

Während die industrielle Produktion aus kommunikationstechnischer Sicht die längste Zeit autark und isoliert operiert hat, gewann mit dem Einzug datenverarbeitender Systeme die Vernetzung nach außen verstärkt an Bedeutung. War der Datenaustausch zunächst auf Verbindungen innerhalb der Fabriken beschränkt, wurden die Datenwege zunehmend länger und die Kommunikation mit weiteren Unternehmensstandorten und anderen Organisationen (Lieferketten) notwendig. Spätestens mit der Anbindung der Werke an das Internet rückte die informationstechnische Sicherheit mehr denn je in den Fokus der Betreiber. [24]

Ein Standard, der den sicheren Datenaustausch in einer solchen Umgebung gewährleisten will, muss entsprechend grundlegende Anforderungen hinsichtlich der Informationssicherheit erfüllen. Die Anwendungsbereiche, in denen OPC UA zum Einsatz kommt, unterscheiden sich hinsichtlich ihrer Sicherheitsforderungen, der relevanten Gefahren und notwendigen Richtlinien. Ausgehend von der Datenerhebung durch Controller auf Feldebene, über die Überwachung von Produktionsprozessen auf operativem Level, bis hin zu Wartungsanfragen im Rahmen des Enterprise-Resource-Planning-Systems (ERP) im übergreifenden Unternehmsnetzwerks und Cloud-Anwendungen, bietet OPC UA Skalierbarkeit über alle Ebenen. [24]

OPC UA kommt dem Umstand entgegen, dass die Anforderungen auf den einzelnen Ebenen ganz unterschiedlich ausfallen und ermöglicht ein flexibles Sicherheitsmodell, das auch im Bereich der Datenerfassung und Steuerung Schutzmechanismen bereitstellt, ohne die Leistungsfähigkeit zu limitieren. [24]

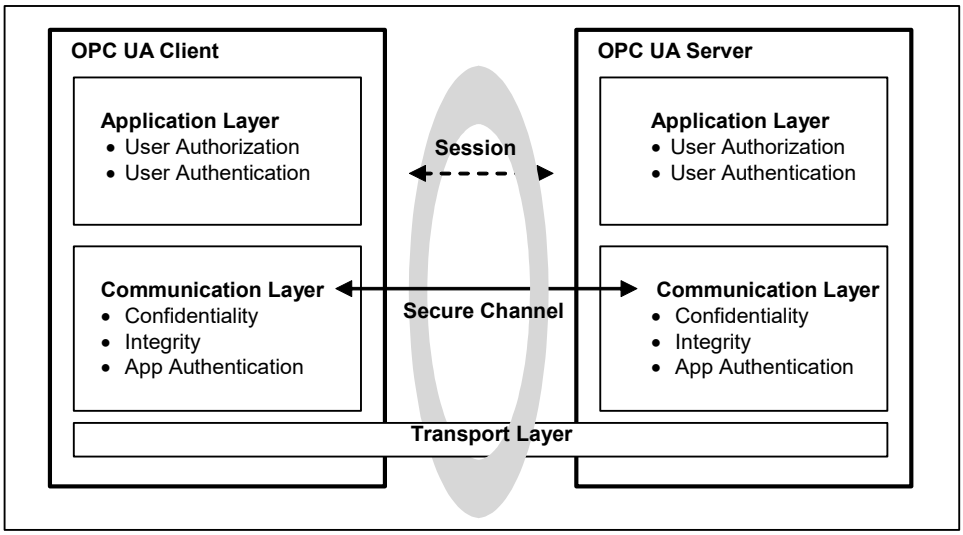


Abb. 2.9: OPC-UA-Sicherheitsarchitektur - Client/Server (Quelle: [29])

Abb. 2.9 zeigt das Client/Server-Kommunikationsmodell innerhalb von OPC UA, welches Informationssicherheit auf drei Ebenen realisiert.

Auf Anwendungsebene findet der Austausch von Informationen, Einstellungen und Befehlen zwischen Client- und Server-Anwendungen im Rahmen einer Session statt. Entsprechend der in [UA Part 4] spezifizierten OPC UA *Session Services* werden auf dieser Ebene Benutzer und bestimmte Produkte authentisiert und autorisiert. [29]

Eine Session kommuniziert über einen *Secure Channel* zwischen beiden Endpunkten, der auf Kommunikationsebene erzeugt wird und für den sicheren Datenaustausch notwendig ist. Der Secure Channel gewährleistet die Integrität der Daten durch Signieren, während die Vertraulichkeit durch die Verschlüsselung sensibler Daten erreicht wird. Darüber hinaus wird das Konzept der Authentifizierung und Autorisierung von Anwendungen eingeführt, so dass Anwendungen sich gegenseitig identifizieren können. Als Basis für dieses Konzept dient der Einsatz spezieller X.509 Zertifikate in Verbindung mit OPC UA *Secure Channel Services*. Zum Zweck der Zertifikatsverwaltung ist eine eigene Infrastruktur notwendig, die die technische Komponente zur Erzeugung solcher Zertifikate mitbringt sowie die ordnungsgemäße Vergabe der Zertifikate prüft. [29]

Zur Sicherung des eigentlichen Datentransports kommen HTTPS oder WebSockets zum Einsatz. HTTPS beschreibt den Austausch von HTTP-Nachrichten über eine zuvor aufgebaute SSL/TLS-Verbindung. Transportsicherheit kann nur im Falle einer Punkt-zu-Punkt-Verbindung gewährleistet werden und erlaubt keine ungesicherten Zwischeninstanzen oder Proxy-Server. Andernfalls ist eine sichere Kommunikation nicht garantiert und der Aufbau eines vertrauenswürdigen Tunnels (z.B. VPN) ist notwendig, bevor eine OPC-UA-Verbindung initiiert werden kann. [29]

Im Kontext browserbasierter Anwendungen bieten WebSockets die Möglichkeit einer bidirektionalen Kommunikation zwischen Client und Webserver. Im Gegensatz zu klassischem HTTP, das für eine Aktion seitens des Servers eine vorherige Anfrage des Clients voraussetzt, erlauben Websockets dem Webserver das asynchrone Senden von Daten, sobald eine Verbindung besteht. WebSockets nutzen dieselben Standardports wie HTTP oder HTTPS und initiieren eine Verbindung mit einer

HTTP-Anfrage, während im weiteren (Übertragungs-)Verlauf der Overhead durch Verzicht auf den HTTP-Header reduziert wird. [29]

Das in Abb. 2.10 dargestellte PubSub-Kommunikationsmodell garantiert die Vertraulichkeit und die Integrität übertragener Daten. Auf Basis von symmetrischer Verschlüsselung nutzt der Publisher den gemeinsamen Schlüssel zum Signieren und Verschlüsseln der Daten, während der Subscriber die Signatur prüfen und empfangene Nachrichten entschlüsseln kann. Zur Verwaltung der Schlüssel ist ein *Security Key Service (SKS)* notwendig, der darüber hinaus die Authentifizierung von Anwendungen und Benutzern ermöglicht. [29]

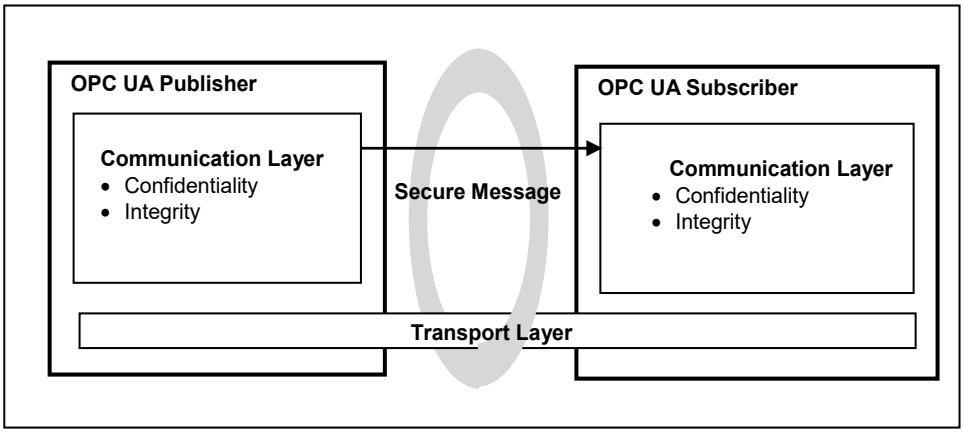


Abb. 2.10: OPC-UA-Sicherheitsarchitektur - Publisher/Subscriber (Quelle: [31])

Das Nutzen symmetrischer Verschlüsselung bietet einerseits den Vorteil einer hohen Geschwindigkeit des Vorgangs, stattet aber gleichzeitig alle Anwendungen, die denselben Schlüssel verwenden, mit den gleichen Rechten aus. Innerhalb einer Sicherheitsgruppe (*SecurityGroup*) muss entsprechend jede Anwendung allen anderen (Anwendungen) vertrauen können. Welche Algorithmen zum Signieren und Verschlüsseln zum Einsatz kommen, wird anhand von Sicherheitsrichtlinien definiert, welche während dem Verbindungsaufbau durch den Server bzw. als spezielles *DataSets* seitens des Publishers im Fall von PubSub kommuniziert wird. [29]

2.4 Applikationserkennung

Die Klassifikation von Netzverkehr und somit das Wissen darüber, welche Applikationen in einem Netzwerk aktiv sind, ist im Bereich des Netzwerkmanagements essenziell (Gefahrenerkennung, QoS, Richtlinien). Klassische Methoden, wie die Analyse von Portnummern und Protokolleigenschaften sind nicht mehr zeitgemäß, da ein Großteil aktueller Anwendungen die Portzuweisung dynamisch realisiert. Die Analyse der Nutzdaten von Datenpaketen stellt eine alternative Methode zur Klassifikation in Form der *Deep Packet Inspection (DPI)* dar. Diese erfordert in der Regel einen hohen Rechenaufwand und setzt häufig auf dedizierte Hardware. Weiterhin steigt die Verbreitung transportverschlüsselter Datenströme, die durch TLS oder andere Verschlüsselungsverfahren vor der Analyse durch Dritte geschützt sind und die Klassifikation auf legalem Weg nahezu unmöglich machen. [32]

Neuere Verfahren basieren ML-Algorithmen, die anhand verbindungs- und paketbezogener Daten spezifische Merkmale erkennen und somit Rückschlüsse auf die zugrundeliegende Anwendung liefern

können.

Die etablierten Klassifikationsmethoden werden im Folgenden detaillierter betrachtet.

2.4.1 Portbasiert

Lange Zeit waren TCP- und UDP-basierten Protokollen eindeutige Portnummern zugeordnet, die entweder offiziell bei der Internet Assigned Numbers Authority (IANA) registriert sind oder inoffiziell von Anwendungen in Anspruch genommen werden [33]. Auf Basis dieser Zuordnung war es üblich, Protokolle und die dahinterstehenden Anwendungen anhand ihrer Portnummer zu identifizieren. Da für diesen Ansatz der Klassifikation lediglich der Transportheader eines Pakets ausgewertet werden muss, ist der Vorgang vergleichsweise schnell und der notwendige Rechenaufwand gering. Gleichzeitig wird die Funktionalität standardmäßig von vielen Netzelementen, wie Routern, Firewalls und Gateways unterstützt, so dass keine zusätzliche Hardware notwendig ist. Allerdings nutzen viele aktuelle Anwendungen keine standardisierten bzw. festen Ports mehr, sondern greifen auf den Bereich der dynamischen Port-Adressen zurück, was die Identifizierung auf diesem Weg deutlich schwieriger gestaltet und diesen Ansatz nicht mehr zeitgemäß erscheinen lässt. [34]

Moore et. al kommen in ihrer Arbeit zu dem Ergebnis, dass die Klassifikation auf Basis standardisierter Ports nur knapp 70 % des gegebenen Datenstroms korrekt identifizieren kann [35]. Zu einem ähnlichen Resultat kommen Madhukar et. al, die die portbasierte Methode als ineffektiv ansehen, da diese angewandt auf typischen Internetverkehr (im Jahr 2006) lediglich eine Erkennungsgenauigkeit von 30 % - 70 % erreicht [36].

2.4.2 Nutzdatenanalyse

Angetrieben durch die sinkende Zuverlässigkeit der Anwendungserkennung anhand von Ports, verschob sich der Fokus weg von der Vermittlungs- und Transportschicht, auch *Stateful Packet Inspection (SPI)* genannt, hin zur Anwendungsschicht, um dort eine unmittelbare Auswertung der Nutzdaten zu ermöglichen.

Diese als *Deep Packet Inspection (DPI)* bezeichnete Verarbeitung von Datenpaketen bietet aufgrund ihres tiefen Einblicks in die Paketstruktur eine hohe Korrektklassifikationsrate und kommt in unterschiedlichen Anwendungsfeldern zum Einsatz. Dazu gehören die Erkennung von Schadsoftware, das Abhören von Kommunikationsverbindungen oder die Zensur bestimmter Inhalte. Internet Service Provider (ISP) setzen DPI darüber hinaus im Bereich des Netzwerkmanagements ein, um beispielsweise P2P-basierte Anwendungen erkennen und ggf. limitieren zu können. Anwendungen dieser Art werden häufig mit Filesharing in Verbindung gebracht und für eine hohe Auslastung des Netzwerks verantwortlich gemacht, was zu negative Auswirkungen für andere Kunden führen kann. Die Erkennung wird anhand vordefinierter Anwendungssignaturen oder einer stochastischen Untersuchung bzgl. bestimmter Parameter in Header und Nutzdaten der Pakete erreicht. Aus Anwendersicht stellt DPI einen Eingriff in die Privatsphäre dar. [32]

Eine solch tiefgreifende Paketuntersuchung in Echtzeit kommt mit einem entsprechend großen Rechenaufwand einher, was zu einer verringerten Verarbeitungsgeschwindigkeit führt. Aufgrund der häufig einzigartigen Eigenschaften des Verbindungsaufbaus auf Anwendungsebene, werden in der Praxis meist nur die Nutzdaten der ersten n Pakete einer Verbindung oder das erste Paket pro

Richtung betrachtet, um den notwendigen Aufwand zu minimieren. Hinzu kommt, dass existierende Signaturen stetig angepasst werden müssen, um die jeweils aktuellste Version einer Anwendungen erkennen zu können. Die steigende Verbreitung von Transportverschlüsselung und Tunnelmechanismen (beispielsweise Virtual Private Networks) erschwert die Erkennung durch DPI und erhöht die Fehleranfälligkeit der Technik. [32]

2.4.3 Verhaltensanalyse

Ein weiterer Ansatz zur Klassifikation von Netzverkehr setzt auf die verhaltensbasierte Analyse der Endpunkte einer Verbindung (Hosts und Server) mittels statistischer und heuristischer Verfahren. Dabei wird die Verbindungsstruktur eines Hosts untersucht, z.B. die Anzahl an kontaktierten Geräten, die verwendeten Protokolle und Ports sowie der zeitliche Rahmen einer bidirektionalen Kommunikation, um Anwendungen zu identifizieren. Verhaltensorientierte Techniken sind vielversprechend und bieten eine gute Korrektklassifikationsrate bei reduziertem Overhead gegenüber den Methoden der DPI. Bevor die Identifizierung einer Anwendung gelingen kann, müssen zunächst eine Reihe von Verbindungsparameter ermittelt und analysiert werden, um ein spezifisches Muster des Verhaltens zu erzeugen.[32]

Karagiannis et. al nutzten in ihrer Arbeit zur Identifizierung von P2P-Verkehr einerseits aus, dass P2P-Anwendungen häufig simultan sowohl TCP als auch UDP zur Datenübertragung verwenden, was laut den Autoren nur wenige Anwendungen außerhalb des P2P-Bereichs auf diese Art praktizieren (DNS, NETBIOS, Gaming). Zusätzlich stützten sie sich auf Verbindungsmuster, da ein Host im P2P-Kontext häufig zu einer Vielzahl von Netzwerkteilnehmern (Peers) verbunden ist. So zeichnet sich eine P2P-Beziehung durch genau eine IP-Adresse und genau eine Port-Adresse aus, während im Fall von Webtraffic meist mehrere simultane Verbindungen zum Herunterladen der Inhalte zum Einsatz kommen. Entsprechend sticht das Verhältnis bzgl. der Anzahl genutzter Ports pro Quell-Ziel-Adress-Paar von P2P-Anwendungen hervor und lässt sich als Erkennungsmerkmal nutzen. Angewendet auf einen Datensatz bestehend aus über einer Millionen Verbindungen zeigte der Ansatz eine Korrektklassifikationsrate von mehr als 95 % hinsichtlich P2P-Verbindungen. Die Falsch-positiv-Rate betrug ca. 10 % und es gelang darüber hinaus P2P-Verbindungen zu identifizieren, welche durch die zur Validierung eingesetzten Nutzdatenanalyse nicht erkannt wurden. [37]

Einen mehrschichtigen Ansatz namens BLINC verfolgen Karagiannis et al.[38], indem der spezifische „Fingerabdruck“ unterschiedlicher Anwendungen auf Basis ihrer Verbindungseigenschaften auf Transportebene als graphenbasierte Struktur dargestellt wird. In Abb. 2.11 sind einige Kommunikationsszenarien und die Interaktionen auf Vermittlungs- und Transportebene als Graphenstrukturen visualisiert. Die ersten beiden Spalten beschreiben jeweils, mit wie vielen Endpunkten ein Host kommuniziert (Quell- und Ziel-IP-Adresse), während die Spalten drei und vier das Verhalten hinsichtlich der Portnutzung offenbaren (Quell- und Ziel-Port). Auf Basis der ermittelten Verbindungsstrukturen eines Hosts erlauben diese *application graphlets* Rückschlüsse auf die aktiven Anwendungstypen (Web, Mail, Angriff etc.) und somit deren Klassifikation.

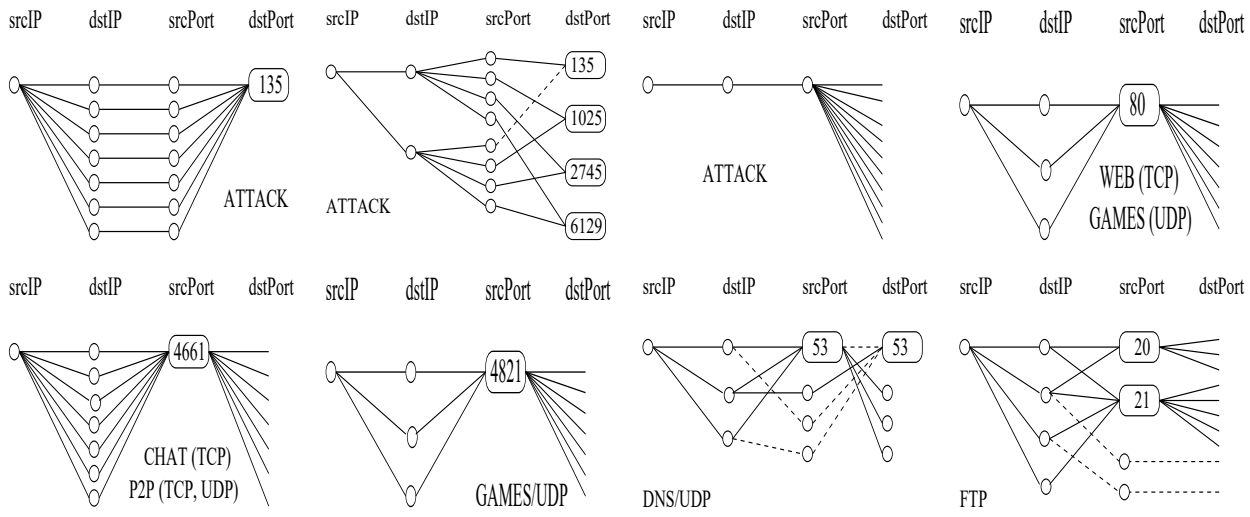


Abb. 2.11: Visuelle Darstellung der Interaktionen auf Transportebene für verschiedene Anwendungen (Quelle: [38])

Die Methode umfasst drei Ebenen:

Auf der *sozialen Ebene* wird das Verhalten eines Hosts anhand seiner Interaktionen mit anderen Hosts beschrieben. Die Rolle des Hosts im Netzwerk, also ob dieser Bereitsteller oder Konsument (oder beides) eines Dienstes ist, wird auf der *funktionalen Ebene* erfasst. Schließlich wird auf der *Anwendungsebene* das Zusammenspiel zwischen bestimmten Hosts in der Transportschicht auf Basis spezifischer Ports festgehalten, um anhand dessen die dahinterstehende Anwendung abzuleiten. Zusätzlich können weitere Merkmale, wie das verwendete Transportprotokoll oder die durchschnittliche Paketgröße einfließen, um die Klassifikation zu verfeinern. Laut den Autoren konnte BLINC angewendet auf reale Mitschnitte 80 % - 90 % des Gesamtverkehr mit einer Genauigkeit von 95 % korrekt klassifizieren.

2.4.4 Statistische Methoden

Die Identifizierung von Anwendungen anhand der Art und Weise, wie diese sich auf Netzebene verhalten und das Erkennen und Ausnutzen einzigartiger Verbindungseigenschaften basieren auf Verfahren der statistischen Klassifikation. Der statistische Ansatz bietet aufgrund seiner geringen Anforderungen hinsichtlich Ressourcenverbrauch eine hohe Skalierbarkeit und eignet sich für Erkennungsszenarien mit (quasi) Echtzeitanforderungen.

Die statistische Analyse verbindungsspezifischer Eigenschaften mit dem Ziel der Anwendungsidentifizierung bietet, bedingt durch die geringe Anzahl verfügbarer Klassifikationsmerkmale, häufig nur eine geringe Korrektklassifikationsrate und ist daher häufig auf zusätzliche Informationen aus den Nutzdaten der Pakete angewiesen. Darüber hinaus muss berücksichtigt werden, dass zahlreiche Internetanwendungen mehrere Verbindungen für den Datenaustausch erzeugen. So fallen neben den Nutzdaten zusätzlicher Kontrollverkehr und anderer funktionaler Verkehr, wie DNS und Multicasts, an. Für eine erfolgreiche Klassifikation ist entsprechend die Beschreibung des kompletten Verbindungsaufkommens einer Anwendung notwendig. [32]

Die Auswertung und Klassifizierung gewonnener Verbindungsdaten geschieht, insbesondere bei großen Datenmengen, häufig mithilfe von Algorithmen aus dem Bereich des maschinellen Lernens. Grundsätzlich wird zwischen zwei Kategorien des maschinellen Lernens unterschieden.

2.5 Maschinelles Lernen

Maschinelles Lernen wird als Schlüsseltechnologie der Künstlichen Intelligenz (KI) angesehen, welche darauf abzielt, Maschinen zu befähigen, Aufgaben „intelligent“ auszuführen. Es ist jedoch unklar, welche Form von Intelligenz gesprochen wird. Seit der Erfindung des Computers ist dieser dem Menschen darin überlegen, Informationen auswendig zu lernen, indem er sie als Dateien speichert. ML beschreibt dagegen die „künstliche“ Generierung von Wissen aus Erfahrung, indem maschinelle Lernverfahren dazu genutzt werden, aus Beispielen ein komplexes Modell erzeugen. [39]

> „Das Modell, und damit die automatisch erworbene Wissenrepräsentation, kann anschließend auf neue, potenziell unbekannte Daten derselben Art angewendet werden. Immer wenn Prozesse zu kompliziert sind, um sie analytisch zu beschreiben, aber genügend viele Beispieldaten - etwa Sensordaten, Bilder oder Texte - verfügbar sind, bietet sich Maschinelles Lernen an. Mit den gelernten Modellen können Vorhersagen getroffen oder Empfehlungen und Entscheidungen generiert werden ganz ohne im Vorhinein festgelegte Regeln oder Berechnungsvorschriften.“[39]

In diesem Abschnitt wird auf die beiden Ansätze des überwachten und unüberwachten Lernens eingegangen. Da sich für die Problemstellung dieser Arbeit Verfahren des überwachten Lernens anbieten, werden einige Vertreter dieser Gruppe genauer beschrieben.

2.5.1 Überwachtes Lernen

Diese Gruppe von Algorithmen erzeugt auf Basis bereitgestellter Trainingsdaten, welche die gewählten Merkmale und Grundwahrheit über die Klassenzugehörigkeit enthalten, ein Klassifikationsmodell. Dieses Modell beschreibt den Übergang zwischen gegebenen Eingangswerten und dem daraus abzuleitendem Ergebnis. Vollständige und fehlerfreie Trainingsdaten sind, neben der gewählten Methode zur Klassendifferenzierung, die wichtigste Voraussetzung für die Genauigkeit überwachter Lernverfahren. Für den Trainingsvorgang sind alle Verfahren dieser Art darauf angewiesen, dass für jedes Beispiel der tatsächliche Klassenwert bekannt ist. In Hinblick auf die Klassifikation von Netzwerkverbindungen kann das entweder manuell oder automatisiert erfolgen. Bujlow et al. [40] realisieren die Assoziation von Netzwerkverbindungen zu dahinterstehenden Anwendungen unmittelbar auf dem ausführenden System, indem die zugehörigen Prozessnamen aus den Sockets des Systems ermittelt werden. Die Nutzdatenanalyse eine weitere (rechenintensive) Möglichkeit dar, (unverschlüsselte) Netzwerkverbindungen mit hoher Genauigkeit zu klassifizieren und wurde von Mula-Valls [41] zur Erzeugung von Trainingsdaten genutzt. Das in der Trainingsphase erzeugte Modell wird in der anschließenden Testphase auf neue, potenziell unbekannte Daten derselben Art angewendet und die Erkennungsrate getestet.

Neben Klassifikationsaufgaben, wie der Interpretation von Zeichen als Ziffern, der Identifizierung von Spam-Emails oder der Erkennung von Gegenständen und Personen in Bildern, sind Schätzungen und Prognosen ein weiteres Anwendungsfeld für überwachte Lernverfahren. Auf Basis gegebener

Daten können Prognosen hinsichtlich Kosten, Angebot und Nachfrage erstellt werden, die Aussagen über zukünftige Entwicklungen erlauben.

Zu den überwachten Lernverfahren zählen alle Verfahren zur Regression oder Klassifikation, beispielsweise mit Algorithmen wie k-nächster-Nachbar, Random Forest, künstliche neuronale Netze, Support Vector Machines oder auch Verfahren der Dimensionsreduktion wie die lineare Diskriminanzanalyse.

2.5.1.1 Classification and Regression Tree (CART)

Entscheidungsbäume sind eine intuitive Methode, um Problemstellungen aus dem Bereich der Klassifikation und der Regressionsanalyse zu begegnen. Ausgehend von einem Wurzelknoten des hierarchisch strukturierten Baumes werden sequenziell Entscheidungen anhand von Grenzwerten getroffen, um zu einem Ergebnis zu gelangen. Die binäre Aufteilung der Objekte stellt dabei ein effizientes Verfahren dar, um die Anzahl verbleibender Auswahlmöglichkeiten mit jeder Frage annähernd zu halbieren, was auch in Multiklassen-Szenarien zu schnellen Ergebnissen führt. [42]

2.5.1.2 Gaussian Naive Bayes (NB)

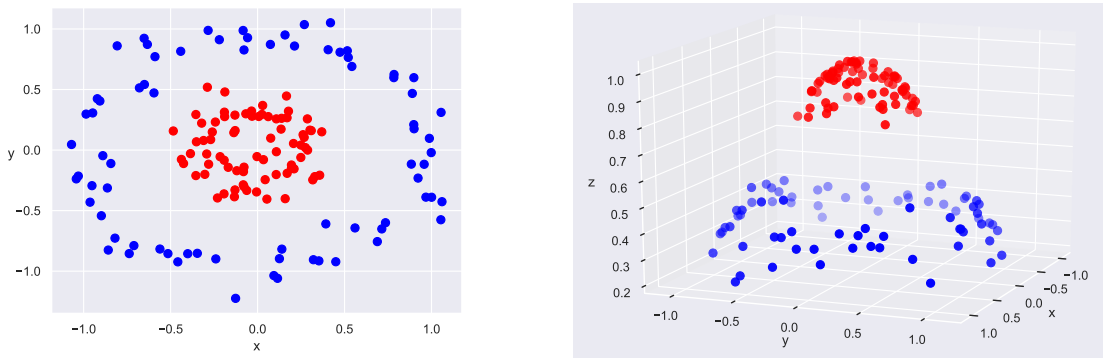
Der Bayes-Klassifikator basiert auf dem Satz von Bayes, welcher die Berechnung bedingter Wahrscheinlichkeiten beschreibt. Bezüglich eines gegebenen Merkmalsraums gelten (naive) zwei Annahmen: Zum einen werden alle Merkmale als gleich wichtig eingestuft und zweitens davon ausgegangen, dass alle Merkmale untereinander statistisch unabhängig hinsichtlich des Klassenwertes sind. In der Realität ist diese Unabhängigkeitsannahme nie gegeben, dennoch erzielt das Verfahren gerade in Praxis gute Ergebnisse. Grund hierfür ist, dass zur Klassifikation keine exakten Wahrscheinlichkeitsschätzungen notwendig sind, sofern die maximale Wahrscheinlichkeit der korrekten Klasse zugewiesen wird. Gaussian Naive Bayes stellt eine Erweiterung des naiven Bayes dar, welcher zusätzlich eine Normalverteilung der vorliegenden Daten annimmt. [42]

2.5.1.3 Ensemble-Verfahren (Bagging, Gradient Boosting)

Ensemble-Algorithmen kombinieren mehrere schwache Klassifikatoren zu einem einzigen Klassifikator mit verbesserter Klassifikationsgüte. Die (schwachen) Basisklassifikatoren verfügen über einen simplen Aufbau und berücksichtigen häufig nur ein einzelnes Objektmerkmal. Diese Einschränkung resultiert einerseits in einer schnellen Auswertung, führt aber gleichzeitig zu schlechten Ergebnissen der einzelnen Klassifikatoren. Durch gezielte Gewichtung kombinieren Ensemble-Verfahren die „schwachen Lerner“, wobei die stärkeren bevorzugt und die schwächsten ignoriert werden. In einem schrittweisen Optimierungsprozess kommt in jedem Durchlauf ein neuer Basisklassifikator hinzu und Stichproben, welche bisher nicht zufriedenstellend erkannt werden, werden in der Folge besonders berücksichtigt. [42]

2.5.1.4 k-nächste Nachbarn (kNN)

Dieser Algorithmus nutzt die Ähnlichkeit zu bereits klassifizierten Objekten, um eine neue Stichprobe einzuordnen. Dabei sind sich zwei Beispiele umso ähnlicher, je geringer der Abstand ihrer Koordinaten im Merkmalsraum ist. Der Abstand kann durch verschiedene Distanzmaße ausgedrückt werden, wie z.B. den euklidischen Abstand oder die Manhattan-Metrik. Da die Zuordnung



(a) Datenpunkte in einer zweidimensionalen Ebene

(b) Datenpunkte im dreidimensionalen Raum

Abb. 2.12: Transformation des Klassifizierungsproblems in eine höhere Dimension (eigene Darstellungen)

eines Objekts durch einen (ggf. gewichteten) Mehrheitsentscheid erfolgt, ist die Wahl eines passenden Wertes für k entscheidend, um die Anfälligkeit der Klassifikation gegenüber Rauschen (k zu klein) bzw. der Einbeziehung unähnlicher Objekte (k zu groß) zu minimieren. [43]

Unter den maschinellen Lernverfahren gehört kNN zur Klasse des *lazy learning* (engl., „Träges Lernen“), welche in der Lernphase lediglich die präsentierten Trainingsbeispiele abspeichern und keinerlei Abstraktion realisieren. Die Modellbildung findet bei diesem Verfahren nicht während des Trainings, sondern erst zum Zeitpunkt der Klassifikation statt. Entsprechend ist die Lernphase sehr simpel, während die Klassifizierungsphase, speziell mit steigender Datenmenge, hohe Anforderungen bzgl. Speicher- und Rechenaufwand stellt. [43]

2.5.1.5 Support Vector Machines (SVM)

Der Ansatz von SVM im Fall eines binären Entscheidungsproblem besteht darin, Datenpunkte als Punkte im (euklidischen) Raum zu betrachten und eine Hyperebene zu konstruieren, die beide Klassen „optimal“ voneinander trennt. Die Zuordnung eines neuen Objekts zu einer Klasse ist davon abhängig, auf welcher Seite der Hyperebene der neue Datenpunkt liegt. Die Position der Hyperebene wird durch diejenigen Datenpunkte beider Klassen festgelegt, welche am nächsten beieinander liegen und deren minimaler Abstand zur imaginären Ebene möglichst groß ist. Diese Punkte am „Rand“ der Hyperebene werden als *Support-Vektoren* bezeichnet. [42]

Ist keine lineare Trennung beider Klassen möglich, wird zunächst eine Transformation in einen höherdimensionalen Raum durchgeführt, in dem die Separierung der Datenpunkte gelingt. Dieser Schritt ist exemplarisch in Abb. 2.12 dargestellt. Im zweiten Schritt werden die Support-Vektoren im transformierten Raum bestimmt. Da die Hochtransformation enorm rechenlastig ist und sich die Hyperebene im niedrigdimensionalen Raum meist als sehr komplex darstellt, wird der sogenannte Kernel-Trick verwendet. Die transformierte Hyperebene wird dann durch geeignete Kernelfunktion beschrieben, ohne dass die tatsächliche Transformationsfunktion bekannt sein muss. [42]

2.5.1.6 Logistische Regression (LR)

Die logistische Regression oder auch Logit-Modell ist ein Klassifikationsverfahren für binäre Problemstellungen. Sie bildet einen Spezialfall der Regressionanalyse zur Ermittlung des Zusammenhangs einer abhängigen Variable Y im Wertebereich von 0 bis 1 und unabhängigen Variablen $X_1, ..., X_M$. Die Wahrscheinlichkeit für den Eintritt eines Ereignisses wird durch den bedingten Erwartungswert der binären Variablen bestimmt. In der Regel wird die Maximum-Likelihood-Methode zur Schätzung der Wahrscheinlichkeit verwendet. [42]

2.5.1.7 Lineare Diskriminanzanalyse (LDA)

Die Diskriminanzanalyse ist ein Verfahren der multivariaten Statistik zur Differenzierung von zwei oder mehr Gruppen, welche sich durch eine Reihe von Merkmalen auszeichnen. Die Methode kann dazu genutzt werden, die Merkmale der unterschiedlichen Gruppen hinsichtlich signifikanter Unterschiede zu prüfen und erlaubt eine Aussage darüber, welche Merkmale für eine Klassifikation geeignet sind. Auf Basis der Merkmalsausprägungen von Objekten deren Klassenzugehörigkeit bekannt ist, wird eine lineare Klassengrenze erzeugt, um die Klassifikation unbekannter Objekte zu ermöglichen. Dabei realisiert die Maximum-Likelihood-Methode die Zuordnung von Objekten zu der Klasse, welche die höchste Wahrscheinlichkeit aufweist. [43]

2.5.2 Unüberwachtes Lernen

Beim unüberwachten Lernen gibt es keinerlei Trainingsdaten und somit auch keinen Trainingsprozess. Lernverfahren dieser Kategorie werden meist auf sehr große und unstrukturierte Datenmengen angewendet, bei denen zu Beginn oftmals noch gar nicht klar ist, welche Klassen existieren bzw. nach welchen Kriterien Objekte sich diesen zuordnen lassen. Stattdessen wird versucht, „Strukturen und Unterschiede in den Daten zu erkennen, um etwa Gruppen (Englisch: Cluster) ähnlicher Beispiele zu finden.“ [39] Clustering wird häufig eingesetzt, um undurchsichtige Datensammlung oberflächlich zu erkunden und anschließend weitere Analysen durchzuführen, z.B. durch die Einteilung von Objekten anhand ihrer Eigenschaften in die zuvor identifizierten Gruppen auf Basis von überwachtem Lernen.

Darüber hinaus existieren noch weitere Verfahren, wie das semi-überwachte Lernen, das einen Kompromiss zwischen überwachtem und unüberwachtem Lernen darstellt oder bestärkendes Lernen, welches das Feedback der Umwelt nutzt, um Erfolgsaussichten einzelner Aktionen abschätzen zu können.

2.6 Scikit-Learn

Scikit-Learn (sklearn) [44] ist eine freie Python-Bibliothek, die effiziente Implementierungen zahlreicher erprobter Klassifikations-, Regressions- und Clustering-Algorithmen des maschinellen Lernens bereitstellt. Die ausführliche Dokumentation und die simple Syntax ermöglichen dabei auch Einsteigern auf dem Gebiet einen intuitiven Zugang in die Thematik, grundlegende Programmierkenntnisse vorausgesetzt.

Im Wesentlichen wird die Funktionalität durch drei sich gegenseitig ergänzende Schnittstellen realisiert [45]:

- die *estimator*-Schnittstelle (deutsch: Schätzfunktion) dient der Erzeugung und Anpassung statistischer Modelle und bildet das Herzstück der Bibliothek. Sie bietet Mechanismen zur Instanziierung von Objekten und implementiert den Lernvorgang eines Modell anhand von Trainingsdaten durch die *fit*-Methode.
- Auf einem Modell basierende Vorhersagen hinsichtlich unbekannter Daten werden durch die *predictor*-Schnittstelle realisiert. Nach Übergabe des Arrays der zu klassifizierenden Daten, erzeugt predict die Vorhersage hinsichtlich der Klassenzugehörigkeit auf Basis der Trainingsparameter.
- Wie in Abschnitt 4.3 ausgeführt, ist es für viele Lernverfahren essentiell, dass die zu untersuchenden Daten zuvor normalisiert werden. Zu diesem Zweck liefert die *transformer*-Schnittstelle eine Methode, welche die Transformation von Rohdaten in skalierte Daten bewerkstelligt. Darüber hinaus stehen weitere Verfahren der Datenvorverarbeitung, etwa mit dem Ziel der Selektion bzw. Reduktion von Merkmalen, zur Verfügung.

```
1  from sklearn.datasets import load_iris
2  from sklearn.model_selection import train_test_split
3  from sklearn.preprocessing import StandardScaler
4  from sklearn.neighbors import KNeighborsClassifier
5  from sklearn.metrics import accuracy_score
6
7  X, y = load_iris(return_X_y=True)
8  X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33,
       random_state=42)
9
10 scaler = StandardScaler()
11 X_train = scaler.fit_transform(X_train)
12
13 classifier = KNeighborsClassifier(n_neighbors=3)
14 classifier.fit(X_train, y_train)
15
16 X_test = scaler.transform(X_test)
17 knn_pred = classifier.predict(X_test)
18 print(accuracy_score(y_test, knn_pred))
```

Der Vorteil des simplen und uniformen API ist der unkomplizierte Transfer des Umgangs und der Syntax anhand eines Modelltyps auf neue Modelle und Algorithmen.

In dem dargestellten Codeschnipsel wird nach Einbinden der notwendigen Module (Zeilen 1 - 5) zunächst der populäre Iris-Datensatz geladen und in Trainings- und Testdatenmenge aufgeteilt (Zeilen 7 & 8). Anschließend wird der *StandardScaler* als Methode zur Skalierung der Daten geladen und die Trainingsdatenmenge transformiert (Zeilen 10 & 11). Als Klassifikationsverfahren wird die Nächste-Nachbarn-Klassifikation gewählt (Zeile 13), wobei die Klassenzuordnung anhand der drei nächsten Nachbarn erfolgen soll. Auf Basis des skalierten Trainingsdatensatzes wird der Lernprozess des Klassifikators durchgeführt (Zeile 14).

Schließlich erfolgt die Vorhersage bezüglich der Testdaten durch das gebildete Klassifikationsmodell (Zeile 17), nachdem die Testdaten, basierend auf der Skalierung der Trainingsdaten, transformiert

wurden (Zeile 16). Zuletzt wird die Klassifikation anhand der Korrektklassifikationsrate beurteilt (Zeile 18).

Dieser kurze Codeschnipsel zeigt, dass es möglich ist, innerhalb weniger Zeilen einen Klassifikator zu erzeugen und diesen zu bewerten. Gleichzeitig können deutlich komplexere Abläufe in Hinblick auf Vorverarbeitung, Evaluation und Darstellung der Ergebnisse durch sklearn realisiert werden.

3 Anforderungsanalyse

Die Dienstgüte ist ein entscheidender Faktor für die Übertragung von Daten in Telekommunikationsnetzen. Je nach Anwendung unterscheiden sich die Anforderungen an die Informationsübermittlung teilweise deutlich. Während für Audio- und Videoechtzeitkommunikation eine latenzarme Übertragung im Vordergrund steht, liegt der Fokus im Rahmen eines Dateitransfers auf einer zuverlässigen Übermittlung und einem hohem Datendurchsatz.

Insbesondere Echtzeitanforderungen können nicht durch jedes paketvermittelnde Netz erfüllt werden und es sind zusätzliche Maßnahmen notwendig, um die Quality of Service (QoS) zu gewährleisten. Häufig sind diese Ansätze nicht effizient oder mit einem hohen technischen Aufwand verbunden. SDN-basierte Techniken haben aufgrund der Übersicht über das gesamte Netzwerk und ihrer zentralisierten Steuerungslogik ein großes Potenzial besser skalierbare und granularer konfigurierbare Lösungen bereitzustellen.

3.1 Aktueller Stand

Ein technisch einfaches Verfahren ist die Überdimensionierung eines oder mehrerer Netze. Im Rahmen dieses sogenannten Overprovisioning werden die Ressourcen des Netzes derart dimensioniert, dass auch im Fall des maximalen Verkehrsaufkommens die Netzauslastung weniger als 50 % beträgt. Dadurch werden annähernd gleich bleibende minimale Latenzzeiten für alle IP-Pakete gewährleistet, allerdings auch für solche mit nur minimalen QoS-Anforderungen. Dieser Ansatz ist entsprechend kostenintensiv und lässt sich dauerhaft nur in einem überschaubaren LAN, nicht aber in Weitverkehrsnetzen realisieren. [4]

Die Reservierung von Ressourcen im Netzwerk stellt ein weiteres Vorgehen dar, um QoS erbringen zu können. Ein Vertreter dieses Ansatzes ist Integrated Services (IntServ) [46]. „Das IntServ-Verfahren unterstützt absolute QoS, d.h., die zu einem „wichtigen“ Dienst oder einer Dienstklasse gehörenden IP-Pakete werden nicht nur im Vergleich mit anderen bevorzugt behandelt, sondern sie erhalten (sofern vom Netz her möglich), als absolute Größe die gewünschte Dienstgüte. Mit IntServ kann damit eine QoS garantiert werden.“ [4] Die Signalisierung wird durch das Resource Reservation Protocol (RSVP) realisiert, welches jedem beteiligten Netzelement die gewünschten QoS-Parameter mitteilt und die Ressourcen in Form von Bandbreite und internem Speicher reserviert. Dabei wird ein fester Pfad festgelegt, so dass die RSVP-Nachrichten und später die Nutzdaten denselben Weg im Netzwerk nehmen. Die Ressourcenreservierung durch IntServ ist pro Verbindung realisiert, was zusammen mit den notwendigen Aktualisierungszyklen zu einem hohen Signalisierungsaufwand (und einer schlechten Skalierbarkeit) führt.

Die Einteilung von IP-Paketen in unterschiedliche Dienstklassen ist eine Möglichkeit zwischen Anwendungen mit unterschiedlichen Anforderungen an die QoS zu differenzieren. Dabei handelt es sich ausschließlich um relative QoS. Ein in diesem Zusammenhang als „wichtig“ definierter Dienst wird zwar priorisiert gegenüber anderen Diensten verarbeitet, erhält aber keine absoluten Garantien hinsichtlich Latenzzeit, Durchsatz oder Paketverlust. Differentiated Services (DiffServ) [47] ist ein Vertreter dieser Gattung, die auch als Class of Service (CoS) bezeichnet wird. Da die Klassifizierung anhand des Type-of-Service-Byte (ToS-Byte) im IP-Header erfolgt, ist in diesem Zusammenhang keine Signalisierung notwendig.

In nächsten Abschnitt werden einige SDN-gestützte QoS-Mechanismen vorgestellt, welche teilweise die hier angesprochenen Ansätze aufgreifen und diese in eine softwarebasierte Umgebung überführen.

3.1.1 Resource Reservation

Diese Technik beschreibt eine Form des *Virtual Slicing*, das jeder Verbindung einen Teil der insgesamt verfügbaren Übertragungskapazität zuteilt und sich typischerweise aus zwei Komponenten zusammensetzt: Ein Verbindungsklassifikator untersucht den Paket-Header und weist anhand der Richtlinien, die im Controller festgelegt sind, jeder Verbindung einer Prioritätsklasse zu. Anhand dieser Klassifikation legen sogenannte *rate shaper* in OpenFlow-fähigen Switches Regeln zur Reservierung von Ressourcen an, die netzweit gültig sind. Aktuelle (Stand 2015) OpenFlow-Implementierungen (wie Open vSwitch) für Geräte der Data Plane unterstützen standardmäßig kein rate shaping einzelner Verbindungen, so dass zurzeit nur prototypische Frameworks existieren. [48]

FlowQoS[49] ist ein solches Framework, das auf kleine Netzwerke abzielt und SDN nutzt, um Priority Queuing auf Routern entsprechend vorgegebener Richtlinien zu realisieren. Der zugehörige Klassifikator besteht aus zwei Modulen: das erste Modul filtert Web-Traffic und gleicht die zugehörigen DNS-Responses mit vordefinierten Ausdrücken ab. Im Anschluss klassifiziert das zweite Modul den Datenverkehr abseits von HTTP/HTTPS anhand des Verbindungstupels sowie der Größe gesendeter und empfangener Nutzdaten. Für das Shaping werden innerhalb des Routers zwei virtuelle Switches erzeugt, welche in Verbindung mit dem Linux-Werkzeug *Traffic Control (tc)* die Vorgaben des Klassifizierers anwenden.

Das europäische Forschungsprojekt „End-to-End Quality-of-Service support over heterogeneous networks“ (EuQoS) [50] hat das Ziel Dienstgüte für Ende-zu-Ende-Kommunikationsbeziehungen über mehrere heterogene Netzwerke hinweg zu gewährleisten. Die Differenzierung zwischen Paketen geschieht anhand einer Kombination aus Differentiated Services Field (DS Field) und Ziel-IP-Adresse. Der zuständige Controller veranlasst die Router für jeden regulären Routingeintrag, zwei Einträge für die gleiche Ziel-IP-Adresse in ihrer Flow Table vorzunehmen: Ein Eintrag gleicht Pakete ohne gesetztes ToS/DS-Feld ab, welche eine niedrige Priorität zugewiesen bekommen. Pakete deren ToS/DS-Feld gesetzt ist, werden mit hoher Priorität behandelt. Gleichzeitig verfügt jeder Router über zwei Warteschlangen denen die Pakete entsprechend ihrer Priorität zugeteilt und bearbeitet werden. Dieser Ansatz erlaubt zwar die Gewährleistung der Dienstgüte einzelner Verbindungen, die Differenzierung zwischen unterschiedlichen Anwendungen ist dagegen nicht möglich.

3.1.2 Per-Flow Dynamic Routing

Ein weiterer SDN-basierter Ansatz setzt auf dynamisches Routing zur Sicherstellung der Dienstgüte beim Transport von Multimediadaten. Das Konzept mit dem Namen *OpenQoS* [51] unterscheidet für jede Verbindung zwischen zu bevorzugendem Multimedia- und regulärem Datenverkehr. Anhand der Differenzierung von Verbindungen durch einen Klassifizierer weist der OpenFlow-Controller Paketen hoher Priorität spezielle Routen zu, die Dienstgüte garantieren, während regulärer Verkehr auf Basis von *shortest path routing* weitergeleitet wird. Anhand der Auslastung des Netzwerks wird entschieden, ob ein Pfad geeignet ist, Dienstgüte zu garantieren. Ein Pfad gilt als überlastet,

wenn die Auslastung 70 % der verfügbaren Bandbreite übersteigt. Im Gegensatz zur Methode der Reservierung von Ressourcen in Netzelementen, werden Verbindungen ohne QoS-Garantie hier nicht aktiv benachteiligt und die Auswirkungen auf Verzögerung und Paketverlust als gering eingestuft.

3.1.3 Packet En-Queuing

Das Framework QoSFlow [52] nimmt sich der Thematik rund um Paketverarbeitung und -warteschlangen an. Dazu werden neben dem, in OpenFlow version 1.3 eingesetzten First In – First Out-Verfahren (FIFO) weitere Methoden zur Zwischenspeicherung und Weiterleitung von Paketen eingeführt. Da FIFO Pakete auf Basis des Zeitpunkts des Eintreffens weiterleitet, erfolgt keine Differenzierung hinsichtlich der Art des Datenverkehrs, so dass eine relative QoS einzelner Verbindungen nicht gewährleistet werden kann. Zu den mit QoSFlow neu eingeführten Algorithmen gehört *Stochastic Fairness Queueing (SFQ)*, welches Pakete anhand der zugehörigen Verbindung in eine Warteschlange einreiht und per Round-Robin-Verfahren durch diese iteriert. So soll eine faire Behandlung aller Verbindungen garantiert werden, um zu verhindern, dass eine einzelne Verbindung den Großteil der verfügbaren Kapazitäten für sich beansprucht.

Die vorgestellten QoS-Verfahren überführen existierende Mechanismen in softwarebasierte Netzwerke (Ressourcenreservierung, Priorisierung durch Queuing) oder führen neue Ansätze ein (dynamisches Routing). Die Integration einer Anwendungserkennung in einem SDN ist ein vielversprechender Ansatz, die Gewährleistung von QoS dynamischer und effektiver gestalten.

3.2 Klärung der Anforderungen

Aufgrund der steigenden Anzahl von Anwendungen, welche dynamische Ports zur Kommunikation nutzen, sowie der Verbreitung verschlüsselter Datenübertragung und Tunnelmechanismen, gilt die Port- und Protokollanalyse zum Zweck der Identifikation von Datenverkehr als überholt. Die Untersuchung von Nutzdaten (DPI) ist ein rechen- und kostenintensives Unterfangen, welches die Verschlüsselung durch *SSL/TLS Interception* umgehen kann. Dieses Vorgehen entspricht einem Man-in-the-Middle-Angriff (MITM-Angriff), verletzt die Privatsphäre der Kommunikationsteilnehmer und bricht die Zusage einer Ende-zu-Ende-Verschlüsselung.

Statistische Modelle auf Grundlage maschineller Lernverfahren versprechen einen gut skalierbaren Ansatz, um Anwendungen anhand von Paket- und Verbindungseigenschaften zu identifizieren. In der vorliegenden Arbeit soll dieses Vorgehen für die Erkennung von OPC-UA-Kommunikationsbeziehungen umgesetzt werden.

Zur Beurteilung des Klassifikators ist die Kenntnis über die tatsächliche Applikation hinter einer Netzwerkverbindung notwendig. Der Abgleich der Grundwahrheit mit der Einordnung durch den Klassifikator führt zu vier möglichen Fällen:

1. Richtig positiv: Es handelt sich um eine OPC-UA-Verbindung, und der Klassifikator hat diese korrekt erkannt.

2. Falsch negativ: Es handelt sich um eine OPC-UA-Verbindung, aber der Klassifikator hat diese fälschlicherweise nicht als solche erkannt.

3. Falsch positiv: Es handelt sich nicht um eine OPC-UA-Verbindung, aber der Klassifikator hat diese fälschlicherweise als solche eingeordnet.

4. Richtig negativ: Es handelt sich nicht um eine OPC-UA-Verbindung und der Klassifikator hat diese korrekt eingeordnet.

Während der erste und letzte Fall das gewünschte Verhalten abbilden, offenbaren zwei und drei eine falsche Einordnung durch den Klassifikator. Bei diesen Varianten der fehlerhaften Klassifikation gilt es abzuwägen, welcher Fall hinsichtlich der Behandlung im Netzwerk kritischer ist und ob die Erkennung entsprechend eher empfindlich oder konservativ ausgelegt werden soll.

OPC-UA-Verbindungen, welche nicht als solche erkannt werden, erhalten nicht die gewünschte Dienstgüte hinsichtlich Zuverlässigkeit (TCP) bzw. Signallaufzeit (UDP). Im Fall von TCP-basiertem Verkehr erscheint eine falsche Klassifikation weniger kritisch, da die Zuverlässigkeit durch das Protokoll selbst realisiert ist und Verzögerung kein kritischer Faktor ist. Fehlertoleranz ist UDP-immanent und daher nicht als kritisch anzusehen. Dagegen kann eine Verarbeitung nach dem Best-Effort-Prinzip und dadurch ggf. zusätzlich eingeführte Verzögerungen im Kontext einer PubSub-TSN-Kommunikationsbeziehung im Bereich zeitkritischer Anwendungen dazu führen, dass QoS-Anforderungen unter Umständen nicht erfüllt werden.

Wird dagegen eine vermeintliche OPC-UA-Verbindung fälschlicherweise als solche klassifiziert, erhält diese Verbindung eine priorisierte Verarbeitung durch beteiligte Netzwerkelemente. Im Fall einer geringen Auslastung des Netzwerks sollte sich die fehlerhafte Einordnung nicht negativ bemerkbar machen. Liegt dagegen eine hohe Auslastung vor, können falsch klassifizierte Verbindungen die Verarbeitung von tatsächlichem OPC-UA-Verkehr oder anderen Anwendungen mit hohen QoS-Anforderungen negativ beeinflussen, so dass deren Übertragung nicht die notwendige Qualität aufweist.

3.3 Generelle Zielsetzung

Zur Entwicklung einer Methode, die es ermöglicht OPC-UA-Kommunikationsbeziehungen zu erkennen, sollen auf Basis spezifischer Paket- und Verbindungseigenschaften Kenngrößen gebildet werden, die eine binäre Klassifikation von Datenverkehr erlauben.

Zu diesem Zweck werden in einem eigenen Testaufbau unterschiedliche OPC-UA-Szenarien simuliert und untersucht. Zusätzlich werden externe OPC-UA-Mitschnitte in die Betrachtung miteinbezogen. Aus diesen Übertragungsverläufen sollen Kenngrößen gebildet werden und die Klassifizierbarkeit von OPC-UA-Kommunikationsbeziehungen gegenüber anderen Standardanwendungen bewertet werden. Zudem soll untersucht werden, welche Verbindungsmerkmale einen starken statistischen Zusammenhang hinsichtlich der Klassifikationsgüte zeigen.

Für die Datenanalyse kommen verschiedene Verfahren des überwachten maschinellen Lernens zum Einsatz, um deren Klassifikationsvermögen hinsichtlich der vorliegenden Problemstellung zu evaluieren und miteinander zu vergleichen.

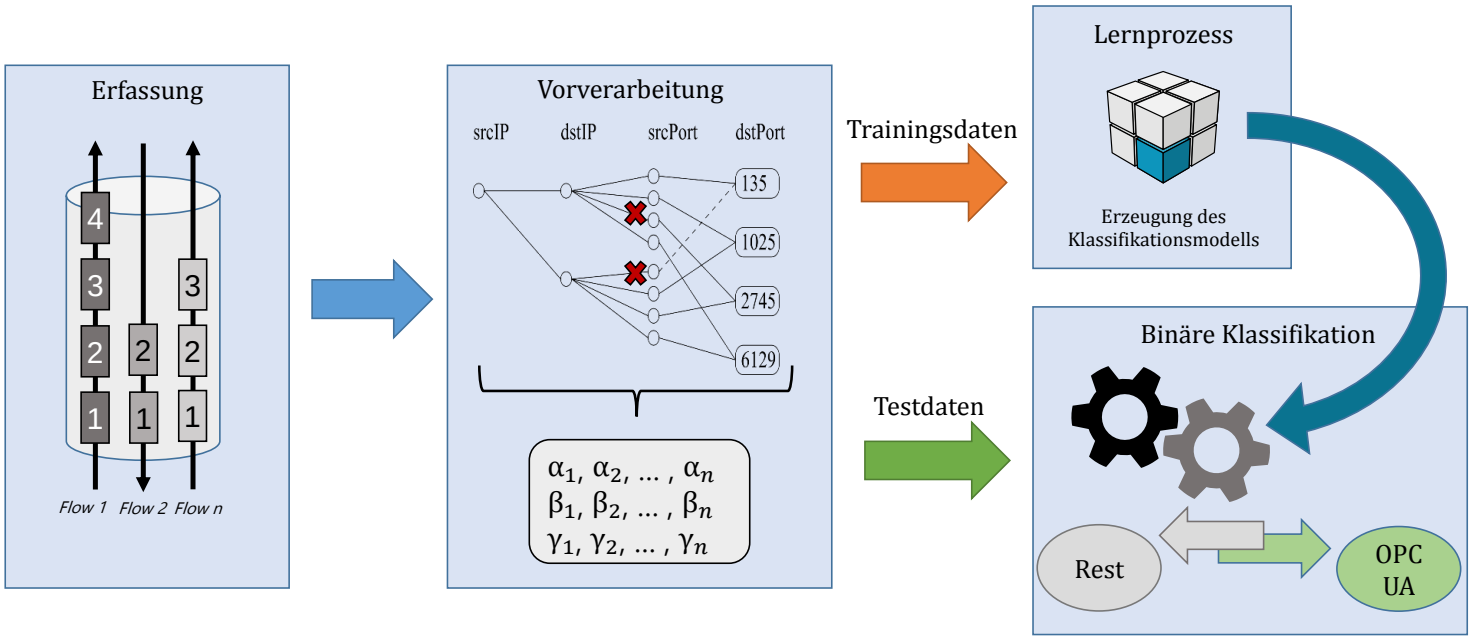


Abb. 3.1: Zielzustand der Anwendungserkennung

In Abb. 3.1 ist die anvisierte Verfahrensweise der Anwendungserkennung dargestellt.

Ausgangspunkt ist ein Paketmitschnitt von Netzwerkverkehr welcher Verbindungen beider Klassen umfasst. Dieser wird entweder direkt als CSV-Datei erzeugt oder aus einem beliebigen Format umgewandelt. Im Rahmen der Vorverarbeitung während zunächst alle Verbindungen des Mitschnitts erfasst und solche entfernt, die weniger als die festgelegte Mindestpaketanzahl aufweisen. Für die verbleibenden Verbindungen werden auf Basis ihrer Zeitstempel- und Paketeigenschaften Kennzahlen ermittelt, die den Kommunikationsverlauf abbilden. Zur Erzeugung von Trainingsdaten ist es im nächsten Schritt notwendig die Verbindungen händisch zu klassifizieren, um die Modellbildung im Rahmen des überwachten Lernprozesses zu ermöglichen. Ausgehend von dem erzeugten Klassifikationsmodell werden neue Verbindungen automatisch klassifiziert.

4 Realisierung

Die Entwicklung eines Mechanismus zur Erkennung von OPC-UA-Kommunikationsbeziehungen umfasste mehrere Phasen. Zunächst war es notwendig, Mitschnitte der OPC-UA-Kommunikation zu erzeugen, die ein breites Spektrum möglicher Anwendungsfälle abdecken und praxisnahe Bedingungen abbilden.

Ausgehend von der Analyse dieser Kommunikationsbeziehungen galt es, Merkmale im Übertragungsverlauf zu identifizieren, die eine potenzielle Differenzierung gegenüber Netzwerkverkehr desselben Transportprotokolls erlauben. Die mithilfe von Wireshark bzw. tshark erzeugten Paketmitschnitte waren hinsichtlich ihres Informationsgehalts beschränkt, da im Rahmen des verfolgten Ansatzes lediglich auf Zeitstempel- und Paketgrößendaten zurückgegriffen wurde. Diese Informationen mussten entsprechend gezielt aufbereitet werden, um aussagekräftige Rückschlüsse zu ermöglichen.

Auf Basis der ausgewählten Merkmale (Abschnitt 4.2) erfolgte die Ermittlung geeigneter Verfahren für die vorliegende Problemstellung. Zu diesem Zweck wurden auf Basis der mitgeschnitten Kommunikationen pro Verbindung Kenngrößen ermittelt, welche für das Training und die Validierung überwachter maschineller Lernverfahren genutzt worden sind, um im nächsten Schritt die Ergebnisse vergleichen und evaluieren zu können.

Außerdem wurde untersucht, welche Merkmale eine starken Zusammenhang hinsichtlich der Klassifikationsgüte besitzen und welche Auswirkung die Variation der Anzahl berücksichtigter Pakete auf das Klassifizierungsergebnis hat.

4.1 Aufzeichnung und Analyse von OPC-UA-Datenverkehr

Das Ziel einer zuverlässigen Klassifikation von OPC-UA-Kommunikationsbeziehungen erfordert die vorherige Simulation und Untersuchung eines breiten Spektrums an Szenarien, die im Kontext von OPC UA auftreten können. Zu diesem Zweck wurde auf Basis von Werkzeugen der Firma *Unified Automation* ein Testaufbau geschaffen, welcher die Simulation von Client-Server- und PubSub-Szenarien ermöglicht.

In Abb. 4.1 ist der Aufbau zur Simulation von PubSub-Szenarien dargestellt. Darin wird der Publisher durch einen Mini-PC vom Typ Intel NUC und die beiden Subscriber durch zwei Laptops der Hersteller Asus und Fujitsu Siemens repräsentiert, die alle über einen gemeinsamen Switch des Herstellers HP verbunden sind. Der Publisher sendet periodisch Nachrichten an die Multicast-Gruppe, der alle drei Endgeräte angehören. Der auftretende Netzwerkverkehr wird über den Monitoring-Port des Switches an einem vierten Endgerät (PC) mitgeschnitten. Der Aufbau der Client-Server-Szenarien entspricht der gleichen Architektur. Der Unterschied besteht darin, dass der Publisher durch einen Server und die Subscriber durch Clients ersetzt werden. Es wird zudem keine Multicast-Kommunikation eingesetzt.

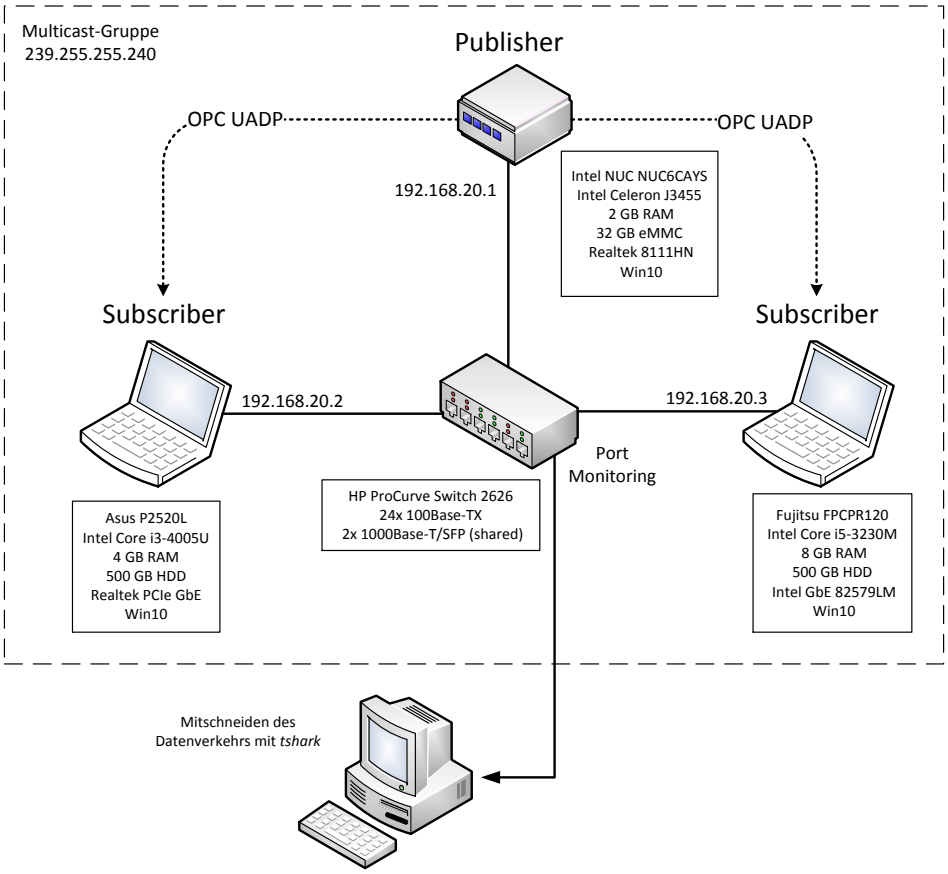


Abb. 4.1: Testaufbau zur Simulation unterschiedlicher PubSub-Szenarien

4.1.1 Client / Server

Im Rahmen Client-Server-basierter Kommunikation wurden verschiedene Anwendungsfälle betrachtet:

- Lese- und Schreibvorgänge mit Read/Write
- Publish-Nachrichten in Folge von sich ändernden Variablenwerten oder anderen Ereignissen
- Peridiodische Erreichbarkeitsabfragen (*watchdog*) ohne aktiven Datenaustausch

Grundsätzlich zeichnet sich der Großteil analysierter OPC-UA-Verbindungen durch einen fest getakteten Austausch von Anfrage- und Antwort-Nachrichten zwischen Client und Server aus. Entsprechend stehen der zeitliche Aspekt der Interaktion zwischen Client und Server und identifizierbare Abweichungen besonders im Fokus der Untersuchung.

Zusätzlich zu diesen simulierten Daten wurden weitere OPC-UA-Mitschnitte durch das Institut für Automation und Kommunikation e. V. (ifak), welches ein Kooperations-Partner im Forschungsprojekt MONAT ist, bereitgestellt. Die Daten wurden im Rahmen des Projekts *OPTimised Industrial IoT and Distributed Control Platform for Manufacturing and Material Handling (OPTIMUM)* [53] erzeugt.

Als OPC-UA-Client dient die Software UaExpert. Diese bietet auf Basis einer grafischen Oberfläche grundlegende Funktionen, wie den Umgang mit Zertifikaten, das Entdecken von UA Servern, sowie

das Verbinden zu diesen. Ist eine Verbindung erfolgreich aufgebaut, lässt sich der Adressraum des Servers durchsuchen sowie Eigenschaften und Referenzen beliebiger UA Nodes begutachten. Der zum Einsatz kommende Server integriert bereits einen vorkonfigurierten Adressraum, welcher ein großes Spektrum an statischen und dynamischen Objekten bietet. Diese können per Drag-and-Drop in das Fenster *Data Access View* kopiert werden, woraufhin eine Subscription des entsprechenden Nodes erstellt wird und dessen Eigenschaften wie Wert und Zeitstempel überwacht werden können. Weiterhin ist das Intervall, in dem es nötig ist den Client über Werteänderungen und Ereignisse zu informieren, konfigurierbar. Dabei ist zu erkennen, dass sich die Anzahl überwachter Objekte in der Größe empfangener Pakete widerspiegelt.

4.1.2 Publish / Subscribe

Die Einführung von OPC UA PubSub liegt zum Zeitpunkt dieser Arbeit rund 18 Monate zurück und entsprechend ist die Verbreitung der Technik in der Praxis, abseits einiger Prototypen, noch sehr überschaubar. Ebenso ist die Bandbreite möglicher Anwendungsfälle für PubSub noch nicht vollkommen absehbar. Bereits intensiv gearbeitet wird an der PubSub-Kommunikation auf Basis von TSN. Die Kombination dieser Verfahren ermöglicht eine Datenübertragung mit Echtzeitgarantien und eröffnet das Potenzial einer ethernetbasierten Lösung im Bereich der M2M- bzw. Controller-to-Controller-Kommunikation. Auf diesem Gebiet ist u.a. das Open Source Automation Development Lab (OSADL) in Zusammenarbeit mit dem Fraunhofer-Institut für Optronik, Systemtechnik und Bildauswertung (Fraunhofer IOSB) und dem indischen Systemintegrations-Unternehmen Kalycito Infotech aktiv. Das Ziel des „OPC UA over TSN“-Projekts [54] ist die Entwicklung einer OPC UA-Implementierung, die für den Einsatz im industriellen Umfeld zugeschnitten ist und erweiterte Funktionalitäten bietet, um mit proprietären Technologien kombiniert zu werden.

Darüber hinaus bietet OPC UA PubSub die Möglichkeit der vertikalen Integration, beispielsweise durch Cloud-Anbindung. So geplant im Rahmen eines Pilotprojekts von Siemens zur Errichtung eines digitalen Umspannwerks für den norwegischen Energieerzeuger und Netzbetreiber Glitre Energi Nett [55]. OPC UA PubSub realisiert dort die Übertragung der Sensordaten aus der Zustandsüberwachung in die Cloud und lokale IoT-Anwendungen.

Die Untersuchung und Klassifikation von PubSub-Kommunikation im Rahmen dieser Arbeit darf daher nur als vorläufige Betrachtung eines (einzelnen) möglichen Anwendungsfalls dieser Technik verstanden werden.

Als Szenario wird die Kommunikation im lokalen Bereich (Controller zu Controller) angenommen, die sich durch den hochfrequentierten Austausch kleiner Datenmengen auszeichnet. Wie in Abb. 4.1 nachvollziehbar, wird ein Aufbau aus einem Publisher und zwei Subscribern zur Simulation genutzt. Dazu wird erneut auf UaExpert zurückgegriffen, der es erlaubt, Konfigurationen sowohl für Publisher, als auch die Subscriber zu erzeugen. Für den Publisher lassen sich u.a. die maximale Nachrichtengröße, das Sendeintervall und zeitliche Offsets einstellen. Weiterhin lassen sich beliebige DataSets erstellen, die jeweils eine Zusammenstellung von Variablen des Adressraums bilden. Auf Subscriber-Seite können Filter definiert werden, um beispielsweise nur solche Nachrichten zu akzeptieren, die von einem bestimmten Publisher stammen oder DataSets von Interesse beinhalten. Die Implementierung der Funktionen von Publisher und/oder Subscriber wird durch einen OPC

UA Server gemäß der bereitgestellten/benutzerdefinierten Konfiguration realisiert.

Für die Simulation werden Sendeintervalle zwischen 1 - 1000 ms und Nutzdatengrößen zwischen 30 - 300 Byte betrachtet.

4.2 Auswahl geeigneter Merkmale zur Identifizierung

Ausgehend von der Untersuchung des OPC-UA-Datenverkehrs werden aus den Mitschnitten die als relevant angesehenen Eigenschaften im csv-Format exportiert. Neben den Merkmalen zur Identifizierung einer Verbindung selbst (5-Tupel), ist vorrangig der zugehörige Zeitstempel und die Größe des Pakets (entsprechend des Werts im IP-Header) für die Analyse interessant. Zusätzlich wird im Fall von TCP-basiertem Verkehr das Feld *Control Flags* aus dem Header der Transportschicht betrachtet.

Tab. 1: Exportierte Paketeigenschaften des aufgezeichneten Datenverkehrs

Zeitstempel	Quell-IP	Ziel-IP	Quell-Port	Ziel-Port	Paketlänge	Protokoll	TCP-Flags
0	192.168.20.2	192.168.20.1	48010	49686	468	6	0x00000018
0.468937	192.168.20.1	192.168.20.2	49686	48010	114	6	0x00000018
0.473652	192.168.20.2	192.168.20.1	48010	49686	40	6	0x00000010
0.515425	192.168.20.2	192.168.20.1	48010	49686	468	6	0x00000018
0.98437	192.168.20.1	192.168.20.2	49686	48010	114	6	0x00000018
0.989142	192.168.20.2	192.168.20.1	48010	49686	40	6	0x00000010

Das in Tab. 1 dargestellte Datenschema dient im Weiteren als Grundlage zur Extraktion und Berechnung relevanter Kennzahlen einer Verbindung. In diesem Zusammenhang wurde ein python-Skript entwickelt, welches bei der Ausführung zunächst alle Verbindungen eines Mitschnitts, die eine Mindestpaketanzahl aufweisen, ermittelt. Für jede dieser Verbindungen werden im Anschluss die gleichen Kenngrößen bestimmt, wobei diese für TCP- bzw. UDP-Verkehr unterschiedlich ausfallen.

4.2.1 Kenngrößen für TCP-basierte Kommunikation

Das Transmission Control Protocol (TCP) [56] ist verbindungsorientiert und zuverlässig. Das bedeutet, dass für den Datenaustausch stets ein Verbindungsaufbau sowie -abbau notwendig sind. Diese Vorgänge, speziell der Aufbau, stechen gegenüber dem regulären Austausch von Nutzdaten hervor und müssen gesondert behandelt werden. Die Zuverlässigkeit des Protokolls ist durch einen Quittierungsmechanismus realisiert, welcher dem Sender empfangene Datenpakete bestätigt. Diese Bestätigungen werden zusammen mit den Nutzdaten im TCP-Header versendet. Falls der Nutzdatenaustausch nicht ausreichend frequentiert stattfindet und auf versendete Nutzdaten keine Reaktion der Gegenseite erfolgt, greift ein Timeout-Mechanismus seitens TCP. Der Empfänger sendet in diesem Fall ein Quittierungspaket, das keinerlei Nutzdaten enthält. Da die Erfassung solcher Quittungen die Untersuchung des typischen Kommunikationsverhaltens einer Anwendungen verfälschen würde, werden diese für die Berechnung der Kenngrößen ausgeblendet.

Tab. 2: TCP-Kenngrößen

Kenngröße	Beschreibung
Datendurchsatz $_{\text{gesamt}}$	Gesamter Datendurchsatz einer (bidrektionalen) Verbindung (kbit/s)
Paketgröße $_{\text{in/out}}$	Verhältnis der durchschnittlichen Paketgröße von Client und Server
Paketanzahl $_{\text{in/out}}$	Verhältnis der Anzahl gesendeter Pakete seitens Client und Server
Datenmenge $_{\text{in/out}}$	Verhältnis der insgesamt gesendeten Datenmenge seitens Client und Server
Zyklus $_{\text{Anfrage/Antwort}}$	Mittlere Dauer zwischen gesendeter Anfrage und empfangener Antwort (s)
Zyklus $_{\text{Antwort/Anfrage}}$	Mittlere Dauer zwischen gesendeter Antwort und empfangener Anfrage (s)
Zyklus $_{\text{VarK, Antwort/Anfrage}}$	Empirischer Variationskoeffizient der Dauer zwischen gesendeter Anfrage und empfangener Antwort
Zyklus $_{\text{VarK, Antwort/Anfrage}}$	Empirischer Variationskoeffizient der Dauer zwischen gesendeter Antwort und (konsekutiver) empfangener Anfrage
Sendeintervall $_{\text{VarK}}$	Empirischer Variationskoeffizient der Dauer zwischen zwei Anfragen
Empfangsintervall $_{\text{VarK}}$	Empirischer Variationskoeffizient der Dauer zwischen zwei Antworten
Push-Anteil	Anteil an TCP-Nachrichten mit gesetztem Push-Flag

In Tab. 2 sind die Kenngrößen für TCP-Verbindungen und deren Beschreibung aufgeführt. Neben dem Datendurchsatz als gängiges Attribut einer Verbindung, soll das Verhältnis der mittleren Größe und der Anzahl ein- und ausgehender Pakete, sowie der insgesamt ausgetauschten Datenmenge Rückschlüsse auf den Hintergrund einer Kommunikation ermöglichen. Eine Annahme an dieser Stelle ist, dass diese Verhältnisse im Fall von web- oder streamingbasierter Kommunikation eine deutliche Tendenz hinsichtlich des eingehenden Datenverkehrs auf Client-Seite zeigen. Demgegenüber wird erwartet, dass industrielle Protokolle für den Datenaustausch ein ausgeglicheneres Bild offenbaren, ggf. sogar mit höherem Anteil seitens eines sendenden Clients (Write-Operation). Einerseits sind die beobachteten Paketgrößen (Nutzdaten) in diesen Szenarien generell kleiner und die Diskrepanz zwischen Anfrage und Antwort fällt (auch aufgrund des Protokoll-Overheads der Anwendungsschicht) geringer aus.

Mit dem Ziel der Abbildung des typischen Kommunikationsverhaltens zwischen Client und Server werden die Anfrage-Antwort-Zyklen zeitlich erfasst und einerseits in Form von gemittelten Absolutwerten, sowie des empirischen Variationskoeffizienten pro Richtung festgehalten.

Weiterhin wird der empirische Variationskoeffizient für die Dauer zwischen zwei Anfragen bzw. zwei Antworten ermittelt, um zu erfassen, ob es sich um eine fest getaktete Kommunikation handelt oder diese beispielsweise durch Nutzerinteraktion beeinflusst wird.

Schließlich wird noch der Anteil an Nachrichten erhoben, bei denen das Push-Flag im TCP-Header gesetzt ist. Dieses Control-Flag erlaubt es, den Puffer-Mechanismus seitens TCP während des Sende- und Empfangsvorgangs zu umgehen und die zugehörigen Daten unmittelbar an die Anwendung zu übergeben, um eine schnellere Verarbeitung zu gewährleisten. Der beobachtete Anteil solcher Pakete war im Zusammenhang mit Web- und Streaming-Anwendungen deutlich geringer, als beispielsweise im Fall von OPC UA.

4.2.2 Kenngrößen für UDP-basierte Kommunikation

Mit Hinblick auf einen schnellen und unkomplizierten Ende-zu-Ende-Datenaustausch entwickelt, zeichnet sich das User Datagram Protocol [57]durch eine verbindungslose und nicht-zuverlässige

Kommunikation aus. Im Gegensatz zu TCP findet auf der Transportebene kein Verbindungsaufbau/-abbau statt und die erfolgreiche Übertragung von Daten wird von der Gegenseite nicht bestätigt. Sind derartige Mechanismen im Rahmen einer UDP-basierten Kommunikation notwendig, müssen diese in einer höheren Schicht implementiert werden. So sind Szenarien möglich, in denen die Übertragung von Daten ausschließlich unidirektional stattfindet. Beispiele dafür sind RTP-Streams via VLC oder OPC UA PubSub-Kommunikationen. Entsprechend ist es an dieser Stelle nicht sinnvoll, auf bestimmte TCP-Kenngrößen, welche das Anfrage-Antwort-Verhalten erfassen, zurückzugreifen.

Tab. 3: UDP-Kenngrößen

Kenngröße	Beschreibung
Datendurchsatz $_{\text{gesamt}}$	Gesamter Datendurchsatz einer (bidirektionalen) Verbindung (kbit/s)
Paketgröße $_{\text{in/out}}$	Verhältnis der durchschnittlichen Paketgröße von Client und Server
Paketanzahl $_{\text{in/out}}$	Verhältnis der Anzahl gesendeter Pakete seitens Client und Server
Datenmenge $_{\text{in/out}}$	Verhältnis der insgesamt gesendeten Datenmenge seitens Client und Server
Sendeintervall	Mittlere Zeitspanne zwischen zwei gesendeten Paketen (s)
Sendeintervall $_{\text{VarK}}$	Empirischer Variationskoeffizient des Sendeintervalls
Empfangsintervall	Mittlere Zeitspanne zwischen zwei empfangenen Paketen (s)
Empfangsintervall $_{\text{VarK}}$	Empirischer Variationskoeffizient des Empfangsintervalls
Paketgröße	Durchschnittliche Paketgröße (Bytes)
Paketgröße $_{\text{VarK}}$	Empirischer Variationskoeffizient der Paketgröße

Wie in Tab. 3 nachvollziehbar werden stattdessen primär die Größe und die Anzahl ausgetauschter Pakete und der Anteil am Sendeaufkommen von Client und Server erfasst. Handelt es sich um eine unidirektionale Übertragung, drückt sich das durch einen Verhältniswert von 1,0 zugunsten des Senders aus. Die Kenngröße Sendefrequenz bzw. deren Varianz dient dazu lastabhängiges Verhalten zu identifizieren, das sich in einem temporären Anstieg der Frequenz gesendeter Pakete bemerkbar macht. Analog soll die Varianz der Paketgröße Aufschluss darüber geben, ob diese während der gesamten Übertragung konstant bleibt oder variiert (z.B. in Zusammenhang mit der dynamischen Anpassung der Videobitrate bei Echtzeitkommunikation mit WebRTC zu beobachten).

4.3 Negative Klasse

Für eine binäre Klassifizierung ist neben der Betrachtung von OPC-UA-Kommunikationsbeziehungen als positive Klasse auch die Einbeziehung anderer Anwendungen notwendig, welche die negative Klasse bilden. Zu diesem Zweck wurde ein breites Spektrum möglicher Anwendungen auf TCP- und UDP-Basis berücksichtigt, deren Kenngrößen in den finalen Datensatz eingeflossen sind.

4.3.1 Anwendungsfälle TCP

Als zuverlässiges und verbindungsorientiertes Protokoll erfreut sich TCP einer weiten Verbreitung und findet Anwendung durch eine große Anzahl von Netzdienste. Es erzeugte eine Ende-zu-Ende-Verbindung, die eine Datenübertragung in beide Richtung ermöglicht. Durch die Erkennung und automatische Behebung von Datenverlusten eignet es sich für alle Anwendungen, die auf eine zuverlässige Datenübertragung angewiesen sind. Weiterhin dient die integrierte Flusskontrolle dazu, hohe Netzauslastung zu erkennen und darauf zu reagieren.

In diesem Abschnitt werden einige TCP-basierte Anwendungen und ihr Kommunikationsverhalten beschrieben, welche im Rahmen der Klassifizierung berücksichtigt wurden.

Webtraffic auf Basis von HTTP/S Das Laden von Webseiteninhalten auf Basis von HTTP/S ist nach wie vor der dominierende Anwendungsfall im Internet. Die vom Client angefragten Inhalte werden durch verantwortlichen Server bereitgestellt, der annähernd alleine für das gesamte Nutzdatenaufkommen verantwortlich ist. In der Regel große Pakete (Segmentierung) und aktiver Datenaustausch nur als Reaktion auf Nutzerinteraktion.

Dateitransfer Herunter- oder Hochladen einer oder mehrerer Dateien mit dem Ziel der vorübergehenden bzw. dauerhaften Speicherung. Nutzdaten werden ausschließlich von der (die Dateien) bereitstellenden Seite versendet, wobei sich die durchschnittliche Paketgröße nah an der Grenze des übertragbaren Maximums befindet. Der maximale Datendurchsatz ist abhängig von der Kapazität des sendenden Hosts bzw. dem physikalischen Limit des Übertragungskanals.

Kommunikation industrieller Kontrollsysteme (ICS & SCADA) Diese Mitschnitte stammen aus einer Sammlung von ICS/SCADA-Kommunikationen eines GitHub-Archivs[2]. Dabei werden u.a. die folgenden Protokolle betrachtet: Common Industrial Protocol (CIP), S7 Communication (S7comm), Modbus TCP, Connection Oriented Transport Protocol (COTP, ISO 8073), Distributed Network Protocol (DNP3 over LAN/WAN) und IEC 60870-5-104. In der Regel nutzen Protokolle aus diesem Bereich ein Anfrage-Antwort-Schema und weisen hinsichtlich des Kommunikationsverhaltens vermutlich die größte Schnittmenge mit OPC UA auf.

Remote-Desktop Mitschnitt des Fernzugriffs auf einen entfernten Server im Rahmen mehrerer Terminalsitzungen. Kommunikation auf Basis des proprietären Netzwerksprotokolls Remote Desktop Protocol (RDP) von Microsoft. Die Kommunikation ist nicht fest getaktet. Ein aktiver Datenaustausch findet nur statt, wenn Änderungen des Bildschirminhalts auftreten, welche der Server dem Client mitteilt oder der Client im Rahmen der Terminalsitzung interagiert. Tendenziell ist die Anzahl gesendeter Pakete seitens Client und Server vergleichbar, während die Menge versendeter Daten durch den Server deutlich überwiegt.

Streamingdienste Kommunikation bekannter Streaming-Plattformen wie YouTube und Twitch. YouTube stellt die Inhalte durch On-Demand-Streaming bereit, was sich als Burst-Verkehr (schubweise Anhäufungen von Daten) auf Ebene der Datenübertragung äußert. Demgegenüber bietet Twitch Live-Streaming mit einer Datenrate bis zu 6 Mbit/s und gleichmäßigem/konstantem Kommunikationsverhalten. Typischerweise erfolgt die Übertragung von Nutzdaten ausschließlich durch die Serverseite.

Gaming (Warcraft 3, World of Warcraft) World of Warcraft (WoW) ist ein Massively Multiplayer Online Role-Playing Game (MMORPG) des Spielentwicklers *Blizzard*, das eine virtuelle Umgebung für tausende Spieler gleichzeitig bereitstellt. Die Kommunikation findet über einen zentralen Serverknoten statt und zeichnet sich durch eine große Zahl kleiner TCP-Pakete im Rahmen einer einzelnen Verbindung aus. Abgesehen von einzelnen Burst-Ereignissen seitens des Servers

[2] A collection of ICS/SCADA PCAPs - https://github.com/automayt/ICS-pcap

(Teleportieren, plötzliche Konfrontation mit einer großen Anzahl anderer Spieler), tragen Client und Server gleichermaßen zum annähernd konstanten Datenaustausch bei.

Warcraft III ist ein Echzeit-Strategiespiel (ebenfalls von Blizzard) aus dem Jahr 2002 (Warcraft III: Reign of Chaos) bzw. 2003 (Warcraft III: The Frozen Throne) dessen Mehrspieler-Modus die Online-Spieleplattform *Battle.net* nutzt. Eine Partie kann zwei bis zwölf Spieler umfassen und dauert typischerweise zwischen 10 und 30 Minuten. Im Rahmen eines Matches werden Pakete in einem festen Intervall gesendet und beinhalten in der Regel weniger als 100 Byte Nutzdaten. Die Anzahl und die Größe ausgetauschter Pakete zwischen Client und Server zeigen eine ausgeglichene Kommunikation.

4.3.2 Anwendungsfälle UDP

Im Vergleich zu TCP zeichnet sich UDP durch einen deutlich simpleren Aufbau und geringeren Funktionsumfang aus. Eine Kommunikation im Rahmen von UDP findet ohne vorherigen Verbindungsaufbau statt und es gibt keine Garantie, dass Pakete in der richtigen Reihenfolge bzw. überhaupt beim Empfänger eintreffen. Durch seinen minimalen Overhead eignet sich UDP vorwiegend für Szenarien, die eine fehlertolerante Kommunikation mit geringen Latenzen verlangen. Typische Anwendungsfälle bilden Audio- oder Videoechtkommunikation wie z.B. VoIP und Gaming.

SIP Das Session Initiation Protocol (SIP) ist ein Vermittlungsprotokoll zur Etablierung von Kommunikationsbeziehungen von VoIP- und Multimedia-Sitzungen. Grundsätzlich kann SIP per UDP, TCP oder über das Stream Control Transmission Protocol (SCTP) transportiert werden. Üblicherweise kommt das verbindungslose UDP zum Einsatz, da SIP bereits Maßnahmen zur Kommunikationssicherung realisiert und ein zusätzlicher Verbindungsaufbau oder ein Mechanismus zur Flusskontrolle nicht notwendig sind. Während der konkrete Kommunikationsverlauf vom verwendeten Codec abhängt, tragen typischerweise beide Teilnehmer gleichermaßen zur Nutzdatenübertragung in Form kleiner Pakete mit gleichbleibender Sendefrequenz (üblich ~100 Pakete/s) bei.

WebRTC „Web Real-Time Communication between Browsers (WebRTC) ist eine maßgeblich bei W3C standardisierte Web Browser-basierte Lösung für Echtzeit-Audio, -Video, -Text und Collaboration-Applikationen.“ [4]

Für das Nutzen WebRTC-basierter Dienste ist lediglich ein Standard-Browser auf dem jeweiligen Endgerät notwendig. Während die Kommunikation selbst Ende-zu-Ende stattfindet, ist für die Koordinierung der Kommunikation (insbesondere für den Aufbau) eine externe Signalisierungsinstanz nötig. Das Datenaufkommen hängt im Wesentlichen von der verwendeten Hardware und der genutzten Auflösungen bzw. dem Codec ab, während die Sendefrequenz annähernd konstant ist, um Echtzeitanforderungen zu erfüllen.

RTP-Stream via VLC Das Real-Time Transport Protocol (RTP) [58] ist ein Protokoll für einen verbindungslosen und ungesicherten Ende-zu-Ende-Transport von Nutzdaten mit Echtzeitcharakteristik. RTP setzt typischerweise auf UDP als Transportprotokoll auf, leistet aber ebenfalls einen Beitrag zur Transportfunktionalität, wie die Zuteilung von Sequenznummern und das Mitsenden

von Zeitstempeln. Gleichzeitig ist es aber auch eng mit der eingesetzten Applikation verzahnt und erscheint somit als L4/L7-Hybrid.

Der VLC Media Player bietet in diesem Zusammenhang ein Modul für die Übertragung von Audio- und Videodaten in Form eines Streams. Abgesehen von wenigen Kontrollnachrichten durch RTCP, verläuft der Datenaustausch ausschließlich unidirektional. Die Sendefrequenz ist lastabhängig. Die Übertragung von Sequenzen mit viel Bewegung und schnellen Bildwechseln führt entsprechend zu einem erhöhten Paketaufkommen.

QUIC Quick UDP Internet Connections (QUIC) ist ein ursprünglich von Google entwickeltes experimentelles Netzwerkprotokoll, das darauf abzielt verringerte Latenzzeiten für Webanwendungen im Vergleich zu TCP zu erreichen. Gegenüber TCP+TLS+HTTP2 verspricht QUIC u.a. Vorteile bei der Geschwindigkeit des Verbindungsaufbaus, der Congestion Control und der Unterstützung von Multiplexing ohne Head-of-Line-Blocking [59]. Seit dem Jahr 2016 arbeitet die IETF an einer Standardisierung des Protokolls, die u.a. Änderungen am Aufbau des QUIC-Headers und dem verwendeten Verschlüsselungsprotokoll gegenüber der Google-Lösung vorsieht [60].

Für die Erzeugung von QUIC-Verkehr wurde auf die existierende Google-Variante zurückgegriffen, welche bei Verwendung des Webbrowsers Chrome in Verbindung mit Google Diensten standardmäßig zum Einsatz kommt. Bereits heute entfällt rund 7 % des globalen Internetverkehrs auf QUIC. Das Kommunikationsverhalten ist erwartungsgemäß sehr ähnlich zu bekannten Webanwendungen (Datenaustausch ausgelöst durch Nutzerinteraktion, Nutzdatenaufkommen quasi ausschließlich durch Server), lediglich die maximale Paketgröße steigt geringfügig.

Gaming (Rocket League, Counter-Strike) Schnelle First-Person Shooter, Sport- und Actionspiele sind auf geringe Latenzzeiten angewiesen, um eine annehmbare Mehrspieler-Erfahrung sowohl über lokale, als auch über Weiterverkehrsnetze zu ermöglichen. UDP als Transportprotokoll ist entsprechend unumgänglich und wird quasi von allen Spielen dieser Kategorie genutzt. Die beiden betrachteten Spiele, Counter-Strike 1.6 und Rocket League, zeichnen sich beide durch eine fest getaktete Kommunikation mit maximal 100 Paketen pro Sekunde je Endpunkt und weniger als 100 Byte Nutzdaten pro Paket aus. Aufgrund der begrenzten Runden- bzw. Matchdauer weisen beide Spiele einen repetitiven Charakter im Übertragungsverlauf auf.

4.4 Merkmals-Normalisierung

Bevor die Klassifizierung eines Datensatzes erfolgen kann, ist die Skalierung bzw. Normalisierung der Rohdaten notwendig. Speziell Algorithmen, deren Klassifizierung sich auf die euklidische Distanz zwischen zwei Datenpunkten zurückführen lässt (kNN) oder die das Gradientenverfahren als Optimierungsmethode nutzen (LR, SVM), sind auf diese Vorverarbeitung angewiesen, um valide Ergebnisse zu gewährleisten. Andernfalls werden Merkmalen, die einen sehr hohen Wertebereich aufweisen, gegenüber kleineren Werten stärker gewichtet und das Klassifizierungsergebnis verfälscht.

4.4.1 Studentisierung

Ein gängiges Verfahren zur Erzeugung vergleichbarer Stichprobenwerte ist die Studentisierung. Während die Standardisierung bzw. z-Transformation die Transformation von Zufallsvariablen be-

schreibt, behandelt die Studentisierung deren Realisierungen in Form von Stichproben. In beiden Fällen werden die Ausgangswerte derart umgerechnet, dass die transformierten Werte ein arithmetisches Mittel von 0 und eine Standardabweichung von 1 aufweisen. Die Stichprobenwerte liegen nach der Umrechnung nicht mehr in Form ihrer Originalmaßeinheiten vor, sondern werden als Vielfache der Standardabweichung der Stichprobe ausgedrückt. [61]

Studentisierter Wert:

$$z_i = \frac{x_i - \bar{x}}{\sqrt{\frac{1}{n}\sum_k (x_k - \bar{x})^2}} = \frac{x_i - \mu_x}{\sigma_x} \tag{1}$$

Für z_i gilt:

$$\text{Arithmetisches Mittel: } \overline{z} = \frac{1}{n}\sum_i z_i = 0 \tag{2}$$

$$\text{Stichprobenvarianz: } \frac{1}{n}\sum_i (z_i - \overline{z})^2 = 1 \tag{3}$$

Die Studentisierung wird innerhalb von sklearn durch die Klasse *StandardScaler* realisiert. Daneben existieren noch eine Reihe weiterer Skalierungsmechanismen, von denen einige im Weiteren kurz vorgestellt werden.

4.4.2 Min/Max-Normalisierung

Der *MinMaxScaler* [62] transformiert Stichproben in den Wertebereich des definierten Minimums und Maximums (standardmäßig zwischen 0 und 1). Dieses Verfahren wird eingesetzt, um Robustheit gegenüber sehr kleinen Standardabweichungen von Attributen zu erreichen und 0-Einträge im Fall dünnbesetzter Matrizen zu bewahren.

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{4}$$

4.4.3 Potenztransformation

Die Potenztransformation [63] nutzt Potenzfunktionen, um eine parametrische und monotone Transformation der Daten zu erzeugen. Die Motivation des Verfahrens liegt darin, eine stärkere Normalverteilung der Daten zu erreichen, um die Varianz zu stabilisieren und die Asymmetrie der Daten zu minimieren.

$$y_i^{(\lambda)} = \begin{cases} \dfrac{y_i^\lambda - 1}{\lambda (GM(y))^{\lambda - 1}}, & if \lambda \neq 0 \\ GM(y) ln(\lambda), & if \lambda = 0 \end{cases}$$

wobei $GM(y) = (y_1 ... y_n)^{\frac{1}{n}}$ das geometrische Mittel der Stichproben $y_1, ..., y_n$ ist.

Die Bestimmung von λ erfolgt gemäß der Maximum-Likelihood-Methode. Kurz gesagt wird dazu derjenige Parameter ausgewählt, bei dem die Wahrscheinlichkeit, das vorliegende Ergebnis zu

beobachten, maximal ist. Die Verteilung auf Basis dieses Parameters deckt sich entsprechend am stärksten mit den beobachteten Stichprobendaten.

4.4.4 Normierung

Diese Methode skaliert jede Stichprobe (die über mindestens eine von Null verschiedene Komponente verfügt) individuell, entsprechend der definierten Norm. Standardmäßig wird dabei die euklidische Norm gewählt, um die Länge des Stichproben-Vektors zu bestimmen. Anschließend werden die einzelnen Komponenten entsprechend normiert, indem durch die zuvor ermittelte Länge dividiert wird. In Abb. 4.3 ist die Transformation exemplarisch dargestellt. Das Verfahren findet u.a. Anwendung im Bereich der Textklassifizierung und des Clusterings. [62]

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} [4,1,2,2] \\ [1,3,9,3] \\ [5,7,5,1] \end{pmatrix} \xrightarrow{\text{Normierung}} \begin{pmatrix} [0.8,0.2,0.4,0.4] \\ [0.1,0.3,0.9,0.3] \\ \underbrace{[0.5,0.7,0.5,0.1]}_{c} \end{pmatrix}$$

$$\|c\|_2 = \sqrt{5^2 + 7^2 + 5^2 + 1^2} = 10 \qquad \frac{c}{\|c\|_2}$$

Abb. 4.3: Beispielhafte Transformation von drei Stichproben

4.5 Beurteilung eines binären Klassifikators

Das Ziel der Modellierung eines Klassifikators ist es, ein Modell zu entwickeln, das auf Basis der im Training genutzten Daten treffende Vorhersagen hinsichtlich unbekannter Daten trifft. Die Problematik dieses Vorgehens besteht darin, dieses Modell beurteilen zu wollen, ohne dass die dafür notwendigen (unbekannten) Daten zur Verfügung stehen. Erfolgen Training und Validierung eines Modells auf Grundlage derselben Daten, ist das Ergebnis eine übermäßig optimistische Bewertung des Klassifikators, die nicht dessen Fähigkeiten angewandt auf neue Daten widerspiegelt. An dieser Stelle spricht man auch von *Overfitting* (Überanpassung) bzw. *Data Leakage* (etwa: das Durchsickern von Daten), also der Umstand, dass die Informationen, welche später klassifiziert werden sollen, bereits in den Lernprozess einfließen und diesen beeinflussen.

Eine Möglichkeit diesem Problem zu begegnen, ist die Aufteilung des vorhandenen Datensatzes in Trainings- und Testdaten. Das Modell wird ausschließlich mit den Trainingsdaten trainiert und im Anschluss auf Basis der Testdaten evaluiert. Hier ist zu bemängeln, dass es nur eine einzelne Schätzung der Fähigkeiten des Klassifikators stattfindet. Weiterhin wird durch die Einteilung in Trainings- und Testdaten die Informationsgrundlage des Modells eingeschränkt und dessen Klassifikationsvermögen verzerrt (Bias), da nur ein Teil der zur Verfügung stehenden Daten zur Bildung des Modells genutzt wird.

4.5.1 Kreuzvalidierung

Ein weiterer Ansatz besteht darin, jeweils nur einen einzelnen Datenpunkt für Testzwecke zurückzuhalten und den verbleibenden Datensatz für das Training zu verwenden. Dieser Prozess wir solange

wiederholt, bis jeder Datenpunkt sowohl im Rahmen des Trainings, als auch des Testvorgangs aufgetaucht ist. Dieses Verfahren ist eine Variante der *Kreuzvalidierung* und erlaubt es einerseits, eine Vielzahl an Schätzungen zu kombinieren, als auch annähernd den gesamten Datensatz für das Training des Modells zu nutzen. Die beschriebene Variante wird als *Leave-One-Out-Cross Validation (LOOCV)* bezeichnet, da stets genau ein Datenpunkt während des Trainings zurückgehalten wird. Diese Methode minimiert die Problematik des Bias während des Trainings, führt im Gegenzug aber eine signifikante Streuung ein, die sich in stark dividierenden Fehlerabschätzungen im Rahmen der Beurteilung des Modells niederschlägt. Des Weiteren ist dieses Vorgehen im Vergleich deutlich rechenintensiver, da Training und Beurteilung für jedes Element des Datensatzes durchgeführt werden und der Aufwand linear mit dessen Größe steigt. [42]

Als Kompromiss zur Minimierung von Verzerrung und Streuung hat sich im Umfeld des maschinellen Lernens die k-fache Kreuzvalidierung etabliert, deren Ablauf in Abb. 4.4 schematisch mit fünf Durchgängen dargestellt ist.

1. Zunächst wird die zur Verfügung stehende Datenmenge gemischt, um eine zufällige Klassenverteilung zu erhalten.
2. Der resultierende Datensatz wird in k Teilmengen $T_1, ..., T_K$ aufgeteilt.
3. Es folgen k Durchläufe, wobei jede Teilmenge T_i einmalig als Testdatensatz und die verbleibenden $k-1$ Teilmengen $\{T_1, ..., T_k\} \setminus \{T_i\}$ als Trainingsdatensatz verwendet werden. In jedem Durchgang wird ein Modell anhand des aktuellen Trainingsdatensatzes erstellt und auf Basis des Testdatensatzes bewertet.
4. Die Gesamtfehlerquote errechnet sich als Durchschnitt aus den Einzelfehlerquoten der k Einzeldurchläufe.

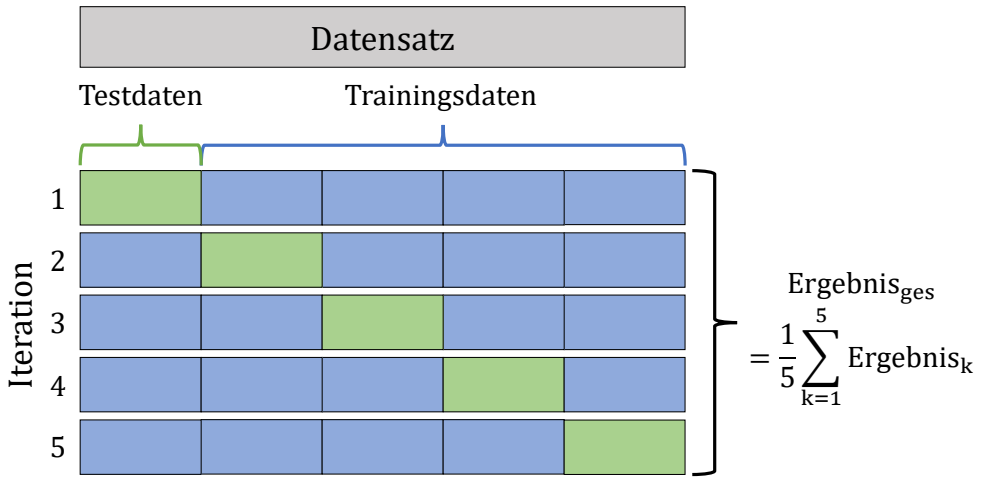


Abb. 4.4: Darstellung des Bewertungsprozesses am Beispiel einer 5-fachen Kreuzvalidierung (eigene Darstellung)

Der Wert für k hängt im Wesentlichen von der Größe des Datensatzes ab, wobei im ML-Bereich typischerweise $k = 5$ oder $k = 10$ verwendet wird [42]. Weiterhin kann der gesamte Vorgang bei Bedarf mehrfach wiederholt werden, um die Verteilung der Elemente innerhalb der Teilmengen zu

variieren. Nach Abschluss der Validierung lässt sich anhand der durchschnittlichen Fehlerabschätzung eine Aussage bezüglich des Bias (des Modells) treffen und bewerten, inwieweit das Modell die vorliegenden Daten akkurat abbildet. Gleichzeitig gibt die Standardabweichung der Fehlerabschätzung einen Eindruck hinsichtlich der Streuung des Modells, also starken Schwankungen der getroffenen Vorhersagen.

4.5.2 Gütekriterien der Klassifikation

Innerhalb von scikit-learn kann die Kreuzvalidierung mit unterschiedlichen Scoring-Verfahren kombiniert werden, um den binären Klassifikator auf Basis der resultierenden *Wahrheitsmatrix* (auch Konfusionsmatrix genannt) zu bewerten.

Tab. 4: Wahrheitsmatrix eines binären Klassifikators

	Tatsächlich positiv	**Tatsächlich negativ**
Vorhersage positiv	Richtig positiv (r_p)	Falsch positiv (f_p)
Vorhersagt negativ	Falsch negativ (f_n)	Richtig negativ (r_n)

Die jeweiligen Häufigkeiten der vier Fälle der in Tab. 4 dargestellten Wahrheitsmatrix können durch Kenngrößen zur Beurteilung des Klassifikators ausgedrückt werden [64]. Dabei steht eine Vielzahl an Gütekriterien zur Auswahl, welche durch Berechnung verschiedener relativer Häufigkeiten ermittelt werden. Für die Betrachtung hinsichtlich der vorliegenden Problemstellung wird im Weiteren nur auf die dafür relevanten Größen eingegangen.

Die Sensitivität beschreibt den Anteil korrekt positiv klassifizierter Objekte an der Menge aller tatsächlich positiven Objekte. Sie ist ein Maß zur Beurteilung der Klassifikationsfähigkeiten hinsichtlich der positiven Klasse.

$$\text{P (positiv klassifiziert}|\text{tatsächlich positiv)} = \frac{r_p}{r_p + f_n} \tag{5}$$

Die Spezifität beschreibt den Anteil korrekt negativ klassifizierter Objekte an der Gesamtheit tatsächlich negativer Objekte. Sie ist ein Maß zur Beurteilung der Klassifikationsfähigkeiten hinsichtlich der negativen Klasse.

$$\text{P (negativ klassifiziert}|\text{tatsächlich negativ)} = \frac{r_n}{r_n + f_p} \tag{6}$$

Das Verhältnis aller korrekt klassifizierten Objekten zur Gesamtmenge wird durch die Korrektklassifikationsrate ausgedrückt.

$$\text{P (korrekt klassifiziert)} = \frac{r_p + r_n}{r_p + f_p + r_n + f_n} \tag{7}$$

Da an dieser Stelle die Klassifizierung negativer Objekte nicht berücksichtigt wird, kann das Ergebnis von (7) unter bestimmten Bedingungen irreführend sein.

Angenommen es liegt ein ungleichverteilter Datensatz mit 100 Objekten vor, welcher sich wie folgt zusammensetzt:

$r_p = 90$, $f_p = 4$, $r_n = 1$, $f_n = 5$

In diesem Beispiel zeigt der Klassifikator ein gutes Ergebnis hinsichtlich der Einordnung positiver Objekte, während negative Objekte nur unzureichend erkannt werden.

Diese Verteilung führt für die drei vorgestellten Gütekriterien zu folgenden Werten/Ergebnissen:

Sensitivität $= \frac{90}{90+5} = 94,7\,\%$

Spezifität $= \frac{1}{1+4} = 20,0\,\%$

Korrektklassifikationsrate $= \frac{90+1}{90+4+1+5} = 91,0\,\%$

Die Kenngrößen Sensitivität und Korrektklassifikationsrate bescheinigen dem Klassifikator mit Werten über 90 % ein gutes Ergebnis. Dagegen liefert die Spezifität mit einem Wert von 20 % ein Indiz dafür, dass der Klassifikator ein Problem mit der Erkennung der negativen Klasse aufweist. Die ungleiche Verteilung des Datensatzes und die Tatsache, dass speziell die Korrektklassifikationsrate die klassenspezifische Erkennung nicht angemessen berücksichtigt, führen an dieser Stelle zu einer übermäßig optimistischen Bewertung in Hinblick auf unbekannte Daten, die eine andere Klassenverteilung haben.

Ein Gütekriterium, das die Klassifizierung sowohl positiver, als auch negativer Elemente berücksichtigt und durch eine einzige Kennzahl ausdrückt, ist der sogenannte *Matthews correlation coefficient (MCC)*. Die Beurteilung eines Klassifikators anhand dieser Kenngröße fällt nur befriedigend aus, wenn dieser positive und negative Elemente gleichermaßen verlässlich zuordnet. Es ist ein Wertebereich von -1 (alle Elemente falsch klassifiziert) bis +1 (alle Elemente richtig klassifiziert) definiert. Ein Wert von 0 entspricht einer zufälligen Vorhersage.

$$MCC = \frac{r_p \times r_n - f_p \times f_n}{\sqrt{(r_p + f_p) \times (r_p + f_n) \times (r_n + f_p) \times (r_n + f_n)}} \tag{8}$$

Für den oben geschilderten 90 % Fall setzt sich der MCC wie folgt zusammen:

$$MCC_{0,90} = \frac{95 \times 1 - 4 \times 5}{\sqrt{(90+4) \times (90+5) \times (1+4) \times (1+5)}} = 0,145$$

Mit einem Ergebnis von 0,145 ist die Erkennung seitens des Klassifikators nur geringfügig besser, als die eines Schätzers, der zufällige Vorhersagen trifft. Da der MCC sich nicht darauf beschränkt, nur die absoluten Zahlen in Relation zu setzen, sondern ungeachtet der Klassenverteilung die Einordnung positiver und negativer Objekte berücksichtigt, gibt der Wert ein realistischeres Bild hinsichtlich der allgemeinen Klassifikationsgüte des Klassifikators.

Aufgrund der spezifischen Eigenschaften des MCC, die Berücksichtigung der gesamten Wahrheitmatrix sowie die Handhabung auch unausgewogener Datensätze, wird dieser im weiteren Verlauf der Arbeit als Bewertungsmaßstab herangezogen.

4.6 Client/Server

Für die Untersuchung der OPC-UA-Client-Server-Kommunikation standen Mitschnitte aus verschiedenen Quellen zur Verfügung (siehe Abschnitt 4.1.1), die ein umfassendes Bild des Kommunikationsverhaltens liefern. Um die verschiedenen Klassifizierungsverfahren hinsichtlich der vorliegenden Problemstellung zu optimieren, wurden zunächst die passenden Skalierungsmethoden und Hyperparameter bestimmt.

Am Anschluss wurde untersucht, welche Merkmale sich besonders für Klassifizierung eignen und welche Auswirkung die Anzahl berücksichtigter Pakete auf die Klassifikationsgüte hat. Abschlossen wird dieser Abschnitt mit der Evaluation der optimierten Klassifikatoren.

4.6.1 Vergleich verschiedener Skalierungsmethoden

Für einen fairen Vergleich der Klassifikationsverfahren wurde für jeden Klassifikator zunächst die Skalierungsmethode ermittelt, welcher auf Basis des TCP-Datensatzes unter Einbeziehung aller vorhandenen Merkmale das beste Ergebnis erzielt. Dieser Datensatz umfasst Identifikationsmerkmale und Kenngrößen von knapp 400 Verbindungen, einschließlich der jeweiligen Klassenzugehörigkeit. Die Klassenverteilung entspricht dabei annähernd einem Verhältnis von 1:2 von OPC-UA-Verbindungen zur negativen Klasse.

Dazu wurden die Klassifikatoren in Verbindung mit den in Abschnitt 4.4 vorgestellten Skalierungsverfahren trainiert und die Klassifikationsgüte, ausgedrückt durch den MCC, ermittelt.

Tab. 5: Vergleich verschiedener Skalierungsmethoden und Klassifikationsverfahren

Klassifikator	**Kein Skalierer**	**StandardScaler**	**MinMaxScaler**	**PowerTransform**	**Normalizer**
LR	0.880 (0.065)	0.816 (0.084)	0.331 (0.101)	0.893 (0.069)	0.269 (0.126)
LDA	0.543 (0.131)	0.533 (0.113)	0.543 (0.131)	0.866 (0.047)	0.506 (0.116)
kNN	0.679 (0.085)	0.884 (0.058)	0.870 (0.066)	0.927 (0.047)	0.896 (0.061)
CART	0.902 (0.058)	0.895 (0.061)	0.900 (0.062)	0.898 (0.065)	0.891 (0.071)
NB	0.721 (0.087)	0.948 (0.041)	0.948 (0.041)	0.937 (0.043)	0.399 (0.051)
SVM	0.000 (0.000)	0.622 (0.115)	0.499 (0.108)	0.934 (0.056)	0.440 (0.115)
Bagging	0.923 (0.057)	0.919 (0.055)	0.925 (0.050)	0.927 (0.051)	0.911 (0.053)
Boosting	0.966 (0.043)	0.965 (0.043)	0.966 (0.042)	0.966 (0.043)	0.933 (0.046)

In Tabelle 5 sind die Ergebnisse der 5-fachen Kreuzvalidierung zusammen mit deren Standardabweichung eingetragen. Es lässt sich erkennen, dass fast alle Algorithmen mehr oder weniger stark von der Vorverarbeitung der zu analysierenden Daten profitieren. Zunächst ist festzustellen, dass der SVM-Klassifikator ohne Skalierung der Rohdaten zu keinem eindeutigen Ergebnis gelangt. Da keines der Objekte der positiven Klasse korrekt zugeordnet wurde, führt das automatisch zu einem MCC-Ergebnis von 0,0.

Den stärksten Zuwachs durch die Skalierung erfährt die LDA, deren Ergebnis sich um 21,7 % verbessert, während sich die Standardabweichung mehr als halbiert. Dahinter reihen sich die Klassifikatoren kNN und NB mit einem knapp 14,5 % bzw. 13,7 % besseren Ergebnis bei ebenfalls verringerter Streuung ein.

Dagegen zeigen die Klassifikatoren CART, Bagging und Boosting bereits ohne Skalierung gute Ergebnisse und verzeichnen durch die vorherige Transformation der Daten keine oder nur minimale Zuwächse.

Das Skalierungsverfahren *PowerTransform* erreicht über alle Klassifikatoren hinweg im Schnitt die besten Resultate, lediglich der NB schneidet mit den Verfahren *StandardScaler* bzw. *MinMaxScaler* geringfügig besser ab.

Teilweise schlechtere Ergebnisse als ohne Skalierung zeigt die Methode *Normalizer*. Speziell in Verbindung mit Algorithmen, die stark von der (passenden) Vorverarbeitung der Daten profitieren, erweist sich dieses Verfahren als ungeeignet.

4.6.2 Hyperparameteroptimierung

Die verschiedenen Klassifikationsverfahren wurden in Abschnitt 4.5 mit den durch *sklearn* vorgegebenen Standardwerten ihrer spezifischen Parameter genutzt, die den besten Kompromiss für ein breites Spektrum möglicher Anwendungsfälle darstellen. Diese sogenannten Hyperparameter lassen sich darüber hinaus gezielter auf die vorliegende Problemstellung hin optimieren, was zu besseren Resultaten führen kann.

Zu diesem Zweck stehen unterschiedliche Such- und Optimierungsverfahren zur Verfügung, die diesen Prozess (teil-) automatisieren. Im vorliegenden Fall wurde sich für die Rastersuche oder *Grid Search* entschieden. Diese nutzt eine definierte Untermenge des Hyperparameterraums der betrachteten Klassifikationsmethode, um die Kombination an Parametern zu ermitteln, welche das beste Ergebnis entsprechend der festgelegten Performance-Metrik erzielt.

Am Beispiel des kNN-Algorithmus soll der Vorgang verdeutlicht werden. Dieser verfügt innerhalb von sklearn über acht konfigurierbare Parameter, von denen sechs direkten Einfluss auf die Klassifikationsgüte ausüben können (die zwei verbleibenden sind Ausführungsoptionen).

Tab. 6: Parameter des kNN-Algorithmus in sklearn

Parameter	**Testwerte**	**Beschreibung**
n_neighbors	{1, 2, **5**, 10, 20}	Anzahl zu berücksichtigender Nachbarn
weights	{**'uniform'**, 'distance'}	Gewichtungsfunktion
algorithm	{**'auto'**, 'ball_tree', 'kd_tree', 'brute'}	Algorithmus zur Berechnung der nächsten Nachbarn
leaf_size	{1, 5, 10, **30**, 50, 100}	Größe der Blätter im Fall von BallTree oder KDTree (beeinflusst Geschwindigkeit und Speicherbedarf)
p	{1, **2**, 5, 10}	Potenzparameter für die Minkowski-Metrik
metric	{'euclidean', 'manhattan', 'chebyshev', 'minkowski', 'wminkowski', 'seuclidean', 'mahalanobis'}	Distanzmetrik

Im Rahmen der Rastersuche wurde der kNN-Algorithmus mit jeder möglichen Kombination von Parameterwerten trainiert und die resultierende Klassifikation durch Kreuzvalidierung bewertet. Nach Abschluss der Suche wird diejenige Kombination zurückgegeben, welche das beste Ergebnis liefert.

Tab. 7: Klassifizierungsergebnisse nach Parameteroptimierung

Klassifikator	**Score (vorgegeben)**	**Score (optimiert)**	**Zuwachs**
LR	0,893 (0,053)	0,912 (0,042)	1,00
LDA	0,866 (0,047)	0,866 (0,047)	0,00
kNN	0,927 (0,047)	0,959 (0,031)	1,66
CART	0,902 (0,076)	0,919 (0,056)	0,89
NB	0,945 (0,033)	0,947 (0,032)	0,10
SVM	0,944 (0,053)	0,980 (0,029)	1,85
Bagging	0,932 (0,059)	0,976 (0,027)	2,28
Boosting	0,948 (0,051)	0,968 (0,040)	0,77

Im vorliegenden Fall weichen lediglich die Anzahl der zu berücksichtigenden nächsten Nachbarn und die „Blattgröße“ von den Standardwerten ab. So liefert die Rastersuche für *n_neighbors* den Wert ’1’ und für *leaf_size* den Wert ’10’ zurück. Die Hyperparameteroptimierung wird analog mit den verbleibenden Algorithmen durchgeführt und die Ergebnisse der geänderten Parameterkonfigurationen in Tab. 7 festgehalten.

Abgesehen vom LDA-Verfahren, resultierte die Optimierung der Hyperparameter bei allen Klassifikatoren in einem leicht verbesserten Klassifikationsergebnis. Im Vergleich zum Skalierungsprozess fielen die Zuwächse aber deutlich geringer aus und lediglich die Klassifikatoren kNN, SVM und Bagging legten um mehr als einen Prozentpunkt zu. Allerdings bewegten sich die meisten Verfahren bereits am oberen Ende der Klassifikationsgüte, so dass im Falle eines anspruchsvolleren Klassifizierungsproblems der Einfluss der Optimierung ggf. deutlicher ausfällt.

4.6.3 Merkmalsbetrachtung

Für die bisherigen Bewertungen wurde auf alle zur Verfügung stehenden Merkmale des Datensatzes zurückgegriffen. Es ist allerdings fragwürdig, ob tatsächlich alle Merkmale einen positiven Beitrag zum Klassifizierungsergebnis liefern. Die Reduktion der Merkmalsvektoren bietet das Potenzial einerseits die Güte des Klassifikators zu erhöhen, als auch den Rechenaufwand im Falle sehr hochdimensionaler Datensätze zu reduzieren.

In diesem Abschnitt soll untersucht werden, welchen Einfluss die Reduktion der Merkmalsanzahl auf die Klassifizierungsgüte hat und welche Merkmale die stärkste Korrelation aufweisen.

Für diesen Zweck bietet sklearn das Modul *feature selection*, welches eine Reihe an Routinen bereitstellt, um auf Basis univariater statistischer Tests die relevantesten Merkmale für die vorliegende Problemstellung zu ermitteln. Im Gegensatz zur Dimensionsreduktion werden bei der Merkmalsselektion keine neuen Kombinationen vorhandener Merkmale generiert, sondern die für das Modellierungsproblem irrelevanten Merkmalen rausgefiltert.

Die Selektion erfolgt anhand der Transinformation als Maß der statistischen Bindung zweier wertdiskreter Größen. Im vorliegenden Fall wird entsprechend untersucht, wie stark der statistische

Zusammenhang zwischen dem betrachteten Merkmal und der Klassenzugehörigkeit ist. Im Falle einer vollständigen Unabhängigkeit beider Größen, verschwindet die Transinformation.

Mit der Methode *SelectKBest* lassen sich die k Merkmale ermitteln, welche das beste Ergebnis hinsichtlich der Klassifizierung zeigen. Anhand dieser Bewertung wurde ein Vergleich von drei Algorithmen aufgestellt, welcher aufschlüsselt, wie die Merkmalsanzahl mit der Klassifizierungsgüte zusammenhängt. Weiterhin wurde untersucht, welche Merkmale die höchste Korrelation aufweisen.

In Abb. 4.5 ist die Klassifizierungsgüte ausgedrückt durch den MCC gegen die korrespondierenden Merkmale aufgetragen. Das entsprechend *SelectKBest* am stärksten korrelierende Merkmal ist ganz links aufgetragen und entlang der x-Achse sinkt die Transinformation jedes Merkmals, welches zusätzlich in die Klassifizierung einfließt.

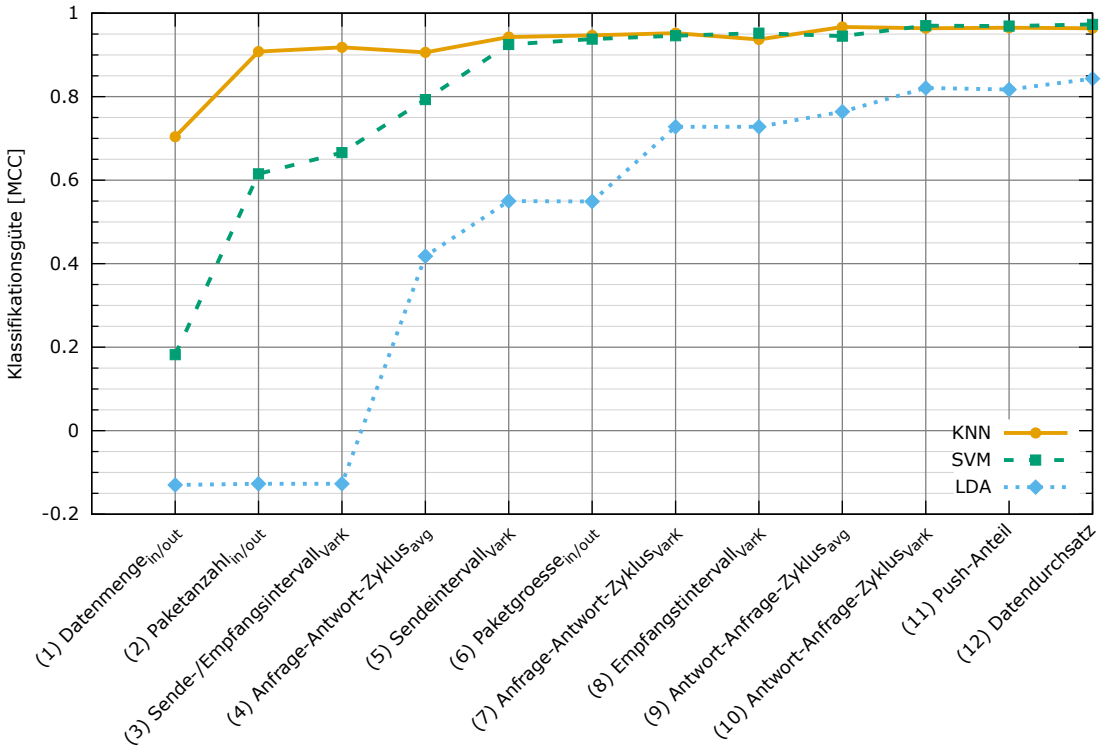


Abb. 4.5: Korrelation der Merkmale zur Klassifizierungsgüte

Zunächst wird der kNN-Algorithmus betrachtet, welcher bereits mit einem einzigen Merkmal einen MCC von 0,7 aufweist und diesen Wert durch Berücksichtigung eines zweiten Merkmals auf gut 0,9 steigern kann. Zusätzliche Merkmale führen darüberhinaus nur noch zu kleinen Verbesserungen des Werts, die Merkmale (3) und (8) führen indes zu einer geringfügig sinkenden Klassifikationsgüte. Im Fall von kNN-Verfahrens sind bereits wenige paketgrößenbasierte Merkmale ausreichend, um eine Klassifizierung mit hoher Zuverlässigkeit ermöglichen.

Die SVM offenbaren eine ähnliche Merkmalskorrelation wie der kNN-Klassifikator, ist aber auf die Berücksichtigung einer größeren Anzahl an Merkmalen angewiesen, um ein vergleichbares Ergebnis zu erreichen. Das gelingt, sobald sechs Merkmale in die Klassifizierung einfließen, was sich in einem MCC von knapp 0,95 ausdrückt.

Dagegen zeigt die LDA ein komplett konträres Bild und klassifiziert Verbindungen auf Basis der ersten drei Merkmale schlechter als ein Klassifikator, welcher zufällige Vorhersagen trifft. Erst mit Berücksichtigung der (zeitbasierten) Merkmale (4), (5) und (7) ist eine signifikante Steigerung der Klassifikationsgüte zu beobachten. Schlussendlich sind zehn Merkmale notwendig, um einen MCC von über 0,8 zu erreichen. Auch mit allen zur Verfügung stehenden Merkmalen kann die LDA kein vergleichbares Ergebnis gegenüber den anderen beiden Verfahren erzielen und reiht sich gut 6 % unterhalb dieser ein.

Ausgehend von diesen Ergebnissen sollen einzelne Merkmalsgruppen und deren Einfluss auf die Güte der Klassifikation untersucht werden. Dazu werden die in Tab. **??** definierten TCP-Kenngrößen in vier Gruppen eingeteilt, von denen die erste die paket- bzw. bytebasierten Kenngrößen umfasst und die anderen drei jeweils die zusammengehörenden zeitbasierten Komplementärgrößen (z.B. Anfrage- und Antwortintervall). Die Klassifizierung anhand dieser Merkmalsgruppen wird erneut auf Basis der Klassifikatoren kNN, SVM und LDA durchgeführt und beurteilt.

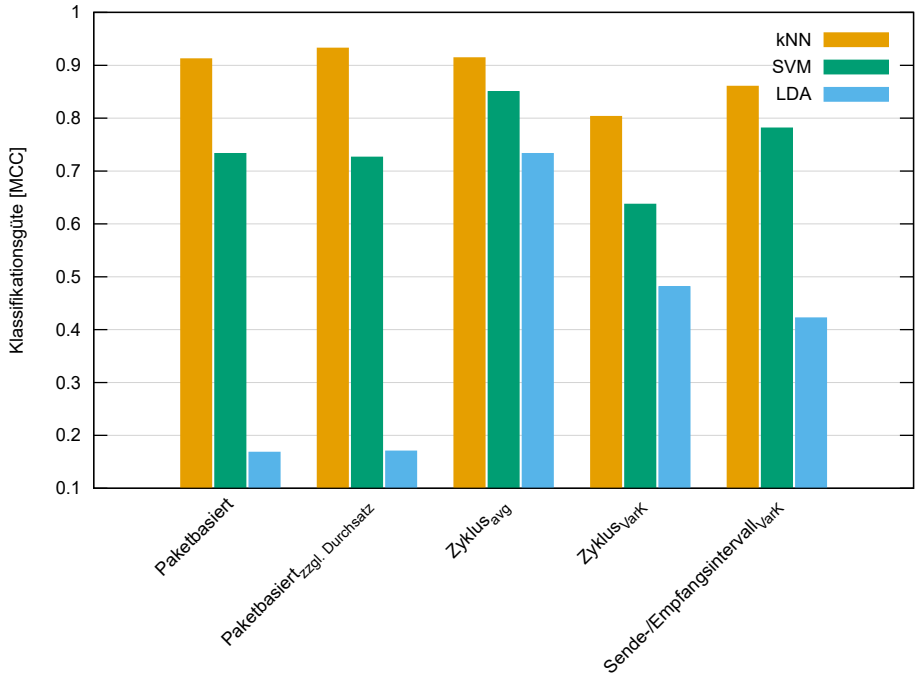


Abb. 4.6: Vergleich verschiedener Kenngrößen

In Abb. 4.6 sind die Ergebnisse erneut durch den MCC ausgedrückt und als Balkendiagramm dargestellt. Das beste Ergebnis über alle Klassifikatoren hinweg wird auf Basis des arithmetischen Mittels der Zykluszeiten (2) erzielt. Das Spektrum der Resultate reicht hier von einem MCC von gut 0,7 (LDA) bis knapp oberhalb von 0,9 (kNN). Demgegenüber bestätigt sich das Bild aus Abb. 4.5, wonach eine Klassifizierung auf Basis paketbasierter Kenngrößen (1) im Fall des LDA-Klassifikators zu einem Ergebnis führt, das nur geringfügig oberhalb einer zufälligen Zuordnung liegt. Der kNN-Klassifikator erreicht in Verbindung mit allen Merkmalsgruppen gute Resultate, wobei die Zuordnung anhand paketbasierter Kenngrößen (1) und der mittleren Zykluszeiten (2) positiv hervorsticht. Gleichzeitig zeigen die SVM in Kombination mit allen Gruppen befriedigende bis gute Ergebnisse, während die LDA abseits der Merkmalsgruppe (2) nicht überzeugen kann. Die

Auswirkungen durch die zusätzliche Berücksichtigung des Datendurchsatzes sind vernachlässigbar.

4.6.4 Paketanzahl

Für die zeitnahe und gleichzeitig zuverlässige Erkennung (neuer) OPC-UA-Kommunikationsbeziehungen ist die Performanz der Anwendungserkennung entscheidend. Für diesen Zweck ist die Klasse des Eager Learning zu bevorzugen, da deren Modellbildung während des Trainings beliebig umfangreich ausfallen kann, während Abfragen im anschließenden Klassifizierungsvorgang (im Vergleich zu Lazy-Learning-Verfahren) deutlich schneller verarbeitet werden. Da diese Abfragen wiederum auf ermittelten Kenngrößen einer Verbindung beruhen, wird in diesem Abschnitt der Zusammenhang zwischen der Anzahl berücksichtigter Pakete (einer Verbindung) und der Klassifikationsgüte analysiert. Variationen im Übertragungs-/Kommunikationsverhalten der Anwendung, sowie Einflüsse auf physischer Ebene (Jitter, Paketverlust) wirken sich auf die Stabilität der Kenngrößen und somit auf das Klassifizierungsergebnis aus.

Um diesen Sachverhalt zu untersuchen, wurden insgesamt rund 60 Verbindungen beider Klassen aus dem Trainings-Datensatz entfernt und separat Kenngrößen für diese gebildet. Im Rahmen der Kenngrößenermittlung variierte die Anzahl berücksichtigter Pakete zwischen 2 - 500. Im Anschluss wurden die einzelnen Verbindungen auf Basis der verschiedenen Kenngrößen klassifiziert und die Ergebnisse anhand drei verschiedener Kriterien gegenübergestellt. Als Gütekriterium im Rahmen der Klassifikation wird einerseits auf den MCC als Kombinationsmaß, als auch auf die Sensitivität bzw. Spezifität zur individuellen Beurteilung der Klassenzuordnung zurückgegriffen.

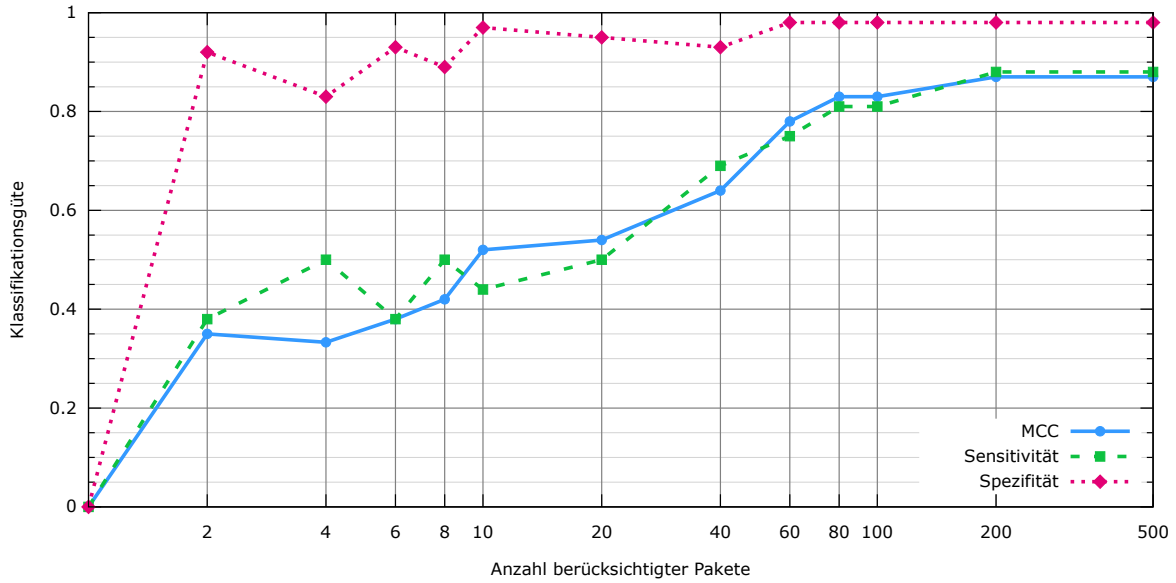


Abb. 4.7: Auswirkung der Paketanzahl auf die Klassifikationsgüte (SVM)

Abb. 4.7 zeigt das Klassifizierungsergebnis in Abhängigkeit von der Paketanzahl, SVM als Klassifikator.

Mit der Berücksichtigung von lediglich zwei Paketen wurden knapp 40 % der betrachteten OPC-UA-Verbindungen korrekt klassifiziert. Gleichzeitig waren zwei Pakete bereits ausreichend, um über 90 % der Verbindungen der negativen Klasse richtig einzuordnen. Dieser Wert fluktuiert zunächst

noch recht stark, ab einer Paketanzahl von zehn und mehr bewegt sich die Spezifität aber stabil um 95 %. Werden mehr als 60 Pakete berücksichtigt, verbessert das die Erkennung nicht weiter. Die Sensitivität beträgt auch nach Berücksichtigung von 20 Paketen lediglich 50 % und erst eine erneute Verdopplung der Paketanzahl steigert die Erkennung deutlich auf annähernd 70 %. Fließen 80 Pakete in die Bildung der Kenngrößen ein, steigt der Wert erstmals über 80 % und verharrt schließlich ab einer Paketanzahl von 200 bei 88 %.

Einerseits zeigt dieser Graph, dass dem SVM-Klassifikator die korrekte Einordnung der negativen Klasse deutlich leichter fällt als die der positiven, sowohl hinsichtlich der notwendigen Pakete, als auch in Bezug auf die maximal erzielte Erkennungsrate. Gleichzeitig wird erstmals die Wechselwirkung beider Klassen auf die Wertbildung des MCC sichtbar, welcher hier stets nahe derjenigen Klasse verläuft, welche die schlechtere Erkennungsgenauigkeit aufweist.

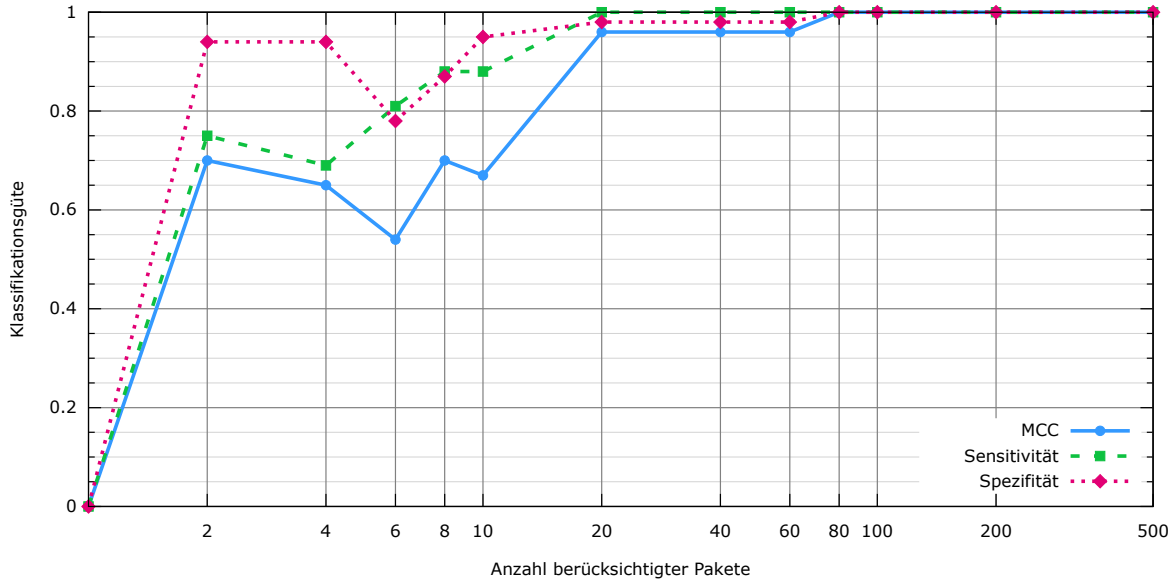


Abb. 4.8: Auswirkung der Paketanzahl auf die Klassifikationsgüte (Bagging)

Dasselbe Szenario wurde mit dem Bagging-Klassifikator wiederholt und die Ergebnisse in Abb. 4.8 festgehalten. Dem Bagging-Verfahren genügten im betrachteten Szenario bereits zwei Pakete, um 75 % der Objekte der positiven Klasse bzw. sogar über 90 % der negativen Klasse richtig zuzuordnen. Fließen sechs Pakete in die Klassifikation ein, zeigt sich ein deutlicher Anstieg hinsichtlich der Erkennung im Fall der positiven Klasse, während die Spezifität auf unter 80 % fällt. Das punktuelle Absinken der Erkennungsgenauigkeit einer Klasse war bereits in Abb. 4.7 nachzuvollziehen und ist vermutlich auf die Berücksichtigung zusätzlicher Merkmale zurückzuführen, die im Fall von zwei bzw. vier Paketen nicht ermittelt werden konnten. Davon sind speziell die zyklischen Parameter betroffen, die jeweils auf mindestens zwei ein- bzw. ausgehende Pakete angewiesen sind, um valide Kennzahlen zu berechnen. Fließen diese Merkmale nicht in die Klassifikation ein, führt das teilweise zu besseren Ergebnissen, als wenn diese, auf Basis weniger Stichproben, miteinbezogen werden.

Mit steigender Paketanzahl stabilisieren sich die Zyklus-Kenngrößen und mit der Berücksichtigung von 20 Paketen steigen Sensitivität und Spezifität auf über 95 %. Ab einer Paketanzahl von 80 gelingt es dem Bagging-Klassifikator alle Verbindungen korrekt einzuordnen.

Die Untersuchung des Abschnitts zeigt einerseits, dass eine gewisse Mindestanzahl an Paketen notwendig ist, um stabile Kennzahlen zu erzeugen und die Zuverlässigkeit der Klassifikation zu steigern. Allerdings überrascht die Erkennung der negativen Klasse, die bereits mit lediglich zwei Paketen über 90 % der Verbindungen richtig einordnet. Die beiden betrachteten Klassifikatoren zeigen dabei deutliche Unterschiede hinsichtlich der benötigten Paketanzahl der absolut erzielten Klassifikationsgüte.

Da im Rahmen dieser Untersuchung allerdings keine Kreuzvalidierung, sondern eine einmalige Aufteilung in Trainings- und Testdaten durchgeführt wurde, ist von einem höheren Bias der Ergebnisse auszugehen. Aufgrund dessen sollten aus der Betrachtung primär Tendenzen anstelle von absoluten Werten abgeleitet werden (Erkennung negative Klasse, Unterschiede zwischen Klassifikatoren).

4.6.5 Evaluation

Die endgültige Bewertung der betrachteten ML-Verfahren wird auf Grundlage des gesamten Datensatzes durchgeführt und das Ergebnis durch stratifizierte Kreuzvalidierung (fünf Durchgänge, zehn Wiederholungen) auf Basis des MCC ermittelt. Jeder Klassifikator wird dazu mit der passenden Skalierungsmethode (vgl. Tab. 5) gepaart und die jeweiligen optimierten Hyperparameter verwendet.

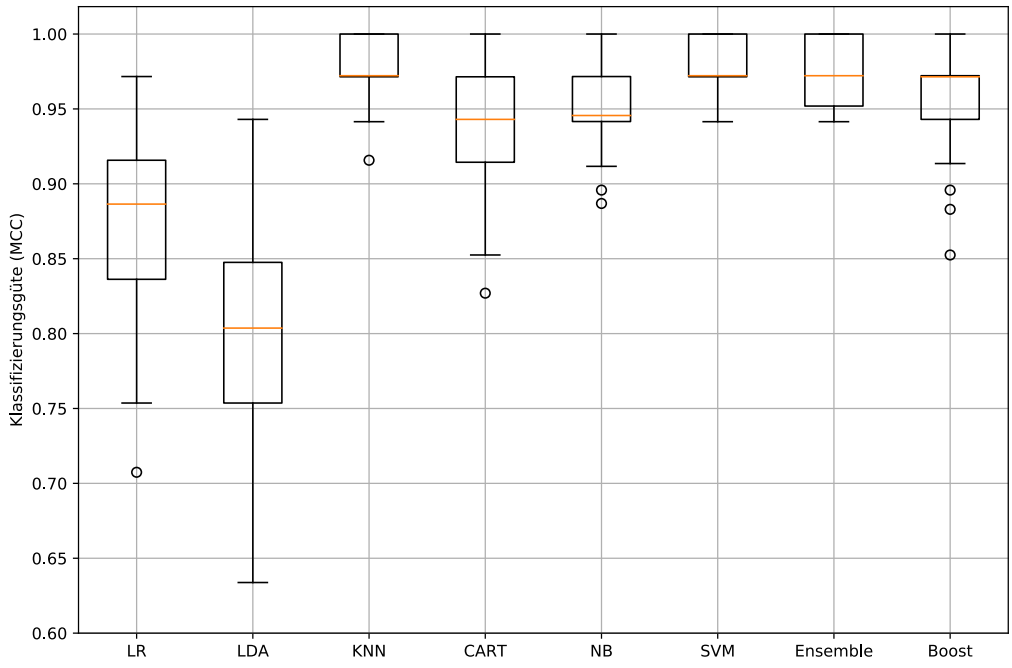


Abb. 4.9: Vergleich der Klassifikatoren nach Optimierung

In Abb. 4.9 sind die Ergebnisse als Box-Plots dargestellt, welche sich aus fünf Kennwerten (Minimum, unteres Quartil, Median, oberes Quartil, Maximum) zusammensetzen. Die Box beschreibt dabei die Verteilung der mittleren 50 % der Daten und ihre als Interquartilsabstand (IQR) bezeichnete Länge erlaubt eine Aussage bezüglich der Streuung. Innerhalb der Box ist der Median als durchgehender Strich festgehalten. Die Antennen (Whisker) bilden von der Box ausgehende Verlängerungen, welche die Verteilung der unteren bzw. oberen 25 % beschreiben, maximal jedoch

das eineinhalbfache des IQR. Alle Datenpunkte außerhalb des durch die Antennen abgedeckten Bereichs werden als Ausreißer definiert und in der vorliegenden Abbildung als Kreise dargestellt.

Die Ergebnisse bestätigen den Eindruck aus den vorherigen Abschnitten, dass das vorliegende Klassifizierungsproblem für die zum Einsatz kommenden maschinellen Lernverfahren generell gut handhabbar zu sein scheint. So erreichen sechs von acht Klassifikatoren einen Median oberhalb von 0,9. Am besten schneiden die Klassifikatoren kNN, SVM und Ensemble ab, deren mittlere 50 % der Ergebnisse sich oberhalb von 0,95 bewegen. Der eingesetzte Ensemble-Algorithmus nutzt SVM als Basisklassifikator und erreicht ähnliche Resultate wie dieser, allerdings mit geringfügig größerer Streuung.

Dahinter reihen sich CART, NB und Boosting ein. Unter diesen drei Klassifikatoren zeigt CART die größte Varianz, während NB und Boosting einzelne Ausreißer unterhalb von 0,9 offenbaren. Es folgen das Verfahren der LR und abgeschlagen die LDA, welche mit einem Median von 0,8 das schlechteste Klassifizierungsergebnis bei maximaler Streubreite aufweist.

Auf Basis der Ergebnisse in Abb. 4.9 zeigen sich die Klassifikatoren kNN, SVM und Ensemble für die vorliegende Problemstellung als besonders geeignet. Unter diesen drei Verfahren weist der Bagging Klassifikator die größte Streuung auf. Allerdings sollten hinsichtlich praktischer Performance-Eigenschaften die Ergebnisse des vorherigen Abschnitts (4.6.4) berücksichtigt werden. Aufgrund der Tatsache, dass die Modellbildung im Fall der SVM während dem Training erfolgt und nicht wie beim kNN-Verfahren während der Anfrage, sind für diesen Klassifikator (deutlich) kürzere Abfragezeiten zu erwarten. Demgegenüber bietet der kNN-Klassifikator als Lazy Learner den Vorteil, dass die Modellbildung lokal in der Umgebung des Arbeitspunktes zum Zeitpunkt der Anfrage geschieht. Für große Datenmengen sind entsprechend SVM zu bevorzugen, während die kNN-Klassifikation in einem Szenario mit überschaubarem Datensatz ggf. bessere Resultate auf Basis der zu klassifizierenden Verbindungen erzielt.

4.7 PubSub

Die vorläufige Betrachtung von PubSub-Kommunikationsbeziehungen und deren Erkennung aus einer Reihe von UDP-basierten Anwendungen entspricht im Wesentlichen der Client-Server-Untersuchung. Da für dieses Szenario nur eine begrenzte Anzahl ausschließlich simulierter PubSub-Verbindungen zur Verfügung steht, die nur eingeschränkte Rückschlüsse erlauben, fällt die Analyse im Vergleich kürzer aus. Im Fokus steht die Betrachtung relevanter Merkmale und eine erste Abschätzung der erreichbaren Klassifikationsgüte (des Erkennungsvermögens durch ML) in diesem Zusammenhang.

Auf einen detaillierten Vergleich verschiedener Skalierungsverfahren wird verzichtet, da die betrachteten Merkmale und Wertebereiche überwiegend unverändert vorliegen. Als Standardverfahren wird die Potenztransformation verwendet, da diese in Verbindung mit dem Großteil der Klassifikatoren die besten Ergebnisse erzielt (Tab. 5). Analog zu Abschnitt 4.6 wird eine Optimierung der Hyperparameter durchgeführt, ohne jedoch den Vorgang erneut im Detail zu betrachten.

4.7.1 Merkmalsbetrachtung

Der unidirektionale Charakter der PubSub-Kommunikation sticht im Feld der betrachteten UDP-basierten Anwendungen hervor. Da der Großteil der eingeführten Merkmale bestimmte Verbin-

dungseigenschaften beider Kommunikationspartner ins Verhältnis setzt bzw. die zeitliche Interaktion beider Parteien berücksichtigt, fällt die einseitige Datenübertragung bereits ohne statistische Hilfsverfahren auf. Lediglich der durch den VLC Player erzeugte RTP-Stream zeigt dahingehend ein vergleichbares Verhalten. Im Rahmen von SIP- und WebRTC-Sitzungen ausgehandelte RTP-Streams sind zwar prinzipiell auch zwei unabhängige unidirektionale Datenströme, welche aber derselben (bidirektionalen) Verbindung zugeordnet werden.

Analog zum Vorgehen in Abschnitt 4.7 werden erneut Gruppen von Merkmalen gebildet und untersucht, wie diese sich auf das Klassifizierungsergebnis auswirken. Entsprechend gibt es wieder Gruppen mit paket- und bytebasierten, sowie zeitbasierten Merkmalen.

Für die Gruppe paketbasierter Merkmale erreichen alle drei Klassifikatoren exakt das gleiche Ergebnis mit einem MCC knapp oberhalb von 0,8.

Die Tatsache, dass die Klassifikatoren einen identischen Wert erzielen, ist auf den bereits angesprochenen unidirektionale Kommunikationsverhalten von PubSub zurückzuführen. Da alle drei einfließenden Merkmale auf dem Verhältnis von Sende- und Empfangsparametern beruhen, stechen PubSub-Verbindungen an dieser Stelle eindeutig hervor. Lediglich die Berücksichtigung der VLC-RTP-Streams verhindert womöglich eine 100 % Erkennungsgenauigkeit anhand der betrachteten Merkmale.

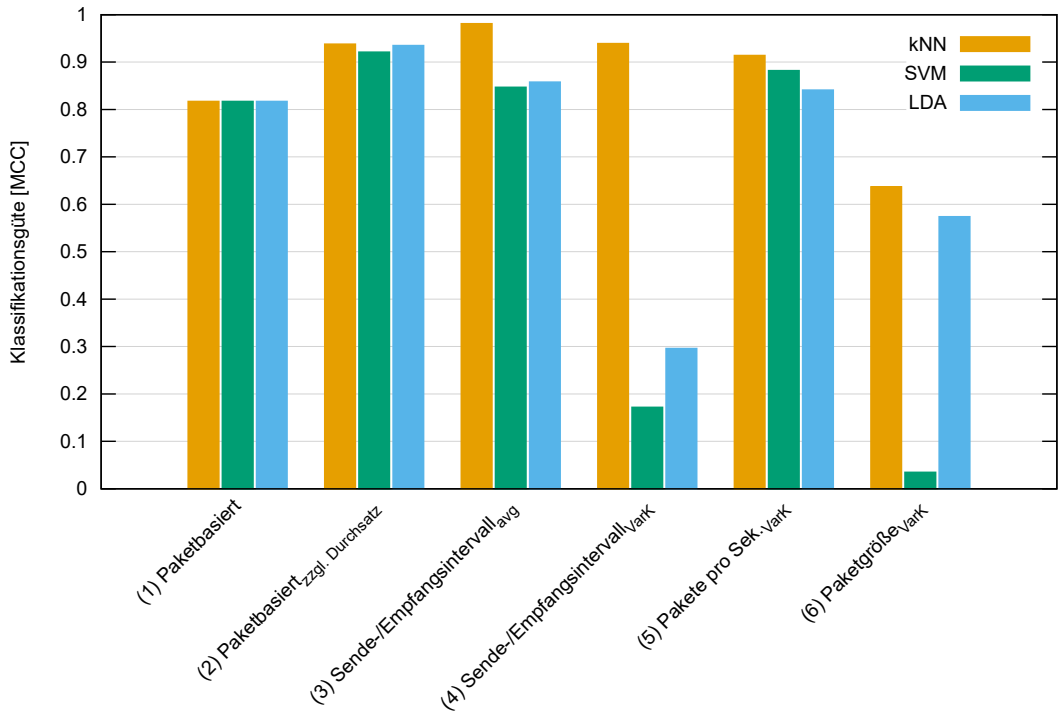


Abb. 4.10: Vergleich verschiedener Kenngrößen

Wird zusätzlich der Datendurchsatz in die Klassifizierung miteinbezogen, steigt die Erkennungsgenauigkeit weiter und bewegt sich für alle Klassifikatoren im Bereich zwischen 0,9 und 0,95.

Die Gruppe welche die zeitliche Interaktion, auf Basis des arithmetischen Mittels der Sende- und Empfangsintervalle, innerhalb der Kommunikation erfasst, erzielt in Verbindung mit dem Klassi-

fikator kNN ein annähernd perfektes Ergebnis von 0,995. Dieser Wert wird ermöglicht durch die Berücksichtigung der Empfangsintervalle, welche im Rahmen der unidirektionalen Übertragung nicht existieren, und Unterschieden hinsichtlich der Sendeintervalle zwischen VLC und PubSub. Die Klassifikatoren SVM und LDA erreichen dieses Niveau nicht und bewegen sich im Bereich um 0,85.

Erfolgt die Klassifizierung hingegen anhand der Streuung der Sende- und Empfangsintervalle, verschlechtern sich die Ergebnisse deutlich. Während kNN nur wenige Prozentpunkte einbüßt, finden sich die SVM unterhalb von 0,2 und die LDA bei 0,3 wieder. Den beiden Klassifikatoren gelingt es in einigen Durchgängen nicht, auch nur eine einzige Stichprobe der positiven Klasse korrekt zuzuordnen, was in einem MCC von 0,0 resultiert.

Die Klassifizierung hinsichtlich eines lastabhängigen Verhaltens, in Form der Streuung der Paketanzahl pro Sekunde, zeigt über alle Klassifikatoren hinweg gute Resultate. Wird hingegen die Streuung der Paketgröße betrachtet, werden nur rund 50-60 % der Verbindungen beider Klassen richtig zugeordnet. Der SVM-Klassifikator fällt hier besonders stark ab und schneidet an dieser Stelle nur minimal besser ab, als ein zufälliger Schätzer.

Insgesamt zeigen sich fünf von sechs Merkmalsgruppen als geeignet, um ein gutes ($MCC \geq 0,8$) bzw. sehr gutes Ergebnis ($MCC \geq 0,9$) im betrachteten Anwendungsfeld in Verbindung mit dem kNN-Klassifikator zu erzielen. Die Klassifikatoren SVM und LDA erreichen in vier Fällen gute Ergebnisse. Wie eingangs erwähnt, sollten die Resultate angesichts der herausstechenden Eigenschaften von PubSub im Feld der betrachteten Anwendungen nicht überbewertet werden und vielmehr einen ersten Eindruck vermitteln, welche Merkmale bzw. welche Kombinationen aus Klassifikator und Merkmalsgruppe sich bereits im Rahmen dieser Betrachtung als ungeeignet erweisen.

4.7.2 Evaluation

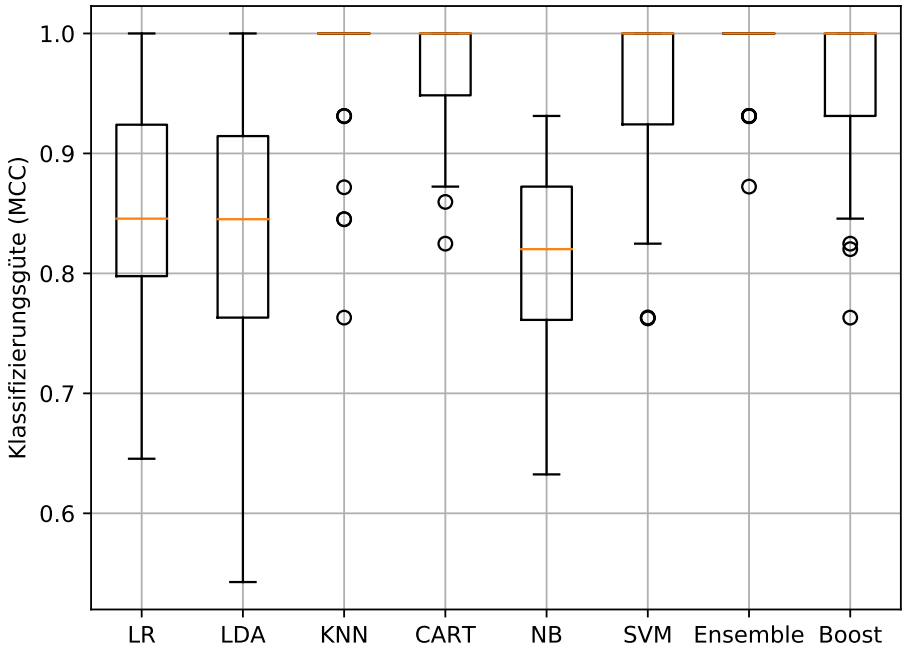


Abb. 4.11: Vergleich der Klassifikatoren nach Optimierung

Abb. 4.11 zeigt das Erkennungsvermögen von PubSub-Verbindungen durch die verschiedenen Klassifikatoren als Box-Plots. Sowohl das kNN- als auch das Ensemble-Verfahren erreichen annähernd 100 % korrekter Einordnungen beider Klassen, mit wenigen Ausreißern bis unter 0,9 (Ensemble) bzw. unter 0,8 (kNN). Es folgen die Klassifikatoren CART, SVM und Gradient Boosting, deren mittlere 50 % sich oberhalb von 0,9 befinden, während die unteren 25 % sich teilweise bis 0,8 erstrecken.Auf den letzten Plätzen finden sich die Klassifikatoren LR, LDA und NB wieder, die eine große Streuung hinsichtlich ihrer Ergebnisse zeigen, aber immer noch befriedigende Resultate liefern.

Wie bereits mehrfach angesprochen, wird die hohe Klassifikationsgüte im Zusammenhang mit PubSub-Kommunikation durch deren unidirektionalen Charakter in Verbindung mit den verwendeten Merkmale begünstigt. Mit der Berücksichtigung einer größeren Anzahl von Anwendungen mit einseitigem Übertragungsverhalten, würden sich die Ergebnisse sicher in nicht unerheblichem Maß relativieren. Allerdings sind dem Autor zum Zeitpunkt der Anfertigung keine weiteren Anwendungen auf UDP-Basis bekannt, welche dieses Verhalten im Rahmen einer zyklischen Datenübertragung zeigen.

Es wurde bereits angemerkt, dass im Rahmen der Klassifizierung nur ein einziges Szenario betrachtet wurde. Weitere Anwendungsfälle sind denkbar, beispielsweise das parallele Senden verschiedener DataSets mit zeitlichen Offsets oder das (evtl. aperiodische) Senden von Qualitätsdaten in die Cloud mit größeren zeitlichen Abständen, welche die Klassifizierung anhand zeitbasierter Merkmale erschweren.

5 Zusammenfassung

Im Rahmen dieser Abschlussarbeit wurde gezeigt, dass eine zuverlässige Erkennung von OPC-UA-Kommunikationsbeziehungen innerhalb des betrachteten Anwendungsspektrums möglich ist. Auf Basis paket- und verbindungsspezifischer Eigenschaften ist es gelungen eine binäre Klassifikation zu realisieren und diejenigen maschinellen Lernverfahren zu ermitteln, welche für die (vorliegende) Problemstellung die besten Ergebnisse zeigen. Im Fall der Client-Server-Variante sind das der kNN-Algorithmus und die SVM, wobei die Entscheidung über den Klassifikator der Wahl vom konkreten Szenario bzw. den Randbedingungen abhängt (Datenmenge, zeitliche Anforderungen). Zudem wurden bestimmte Merkmalsgruppen identifiziert, welche die Klassifizierung besonders begünstigen. Unter diesen stachen insbesondere Merkmale hervor, welche das arithmetische Mittel der Zykluszeiten sowie paketbasierte Kenngrößen berücksichtigten. Der Zusammenhang zwischen der Paketanzahl, welche in die Klassifikation einfließt und der Klassifikationsgüte war ebenfalls Teil der Untersuchung. Dabei konnte gezeigt werden, dass bereits wenige Nutzdatenpakete ausreichend sein können, um die Erkennung (einer Klasse) mit guter Genauigkeit zu ermöglichen. Die unterschiedlichen Klassifikatoren offenbarten an dieser Stelle teilweise signifikante Unterschiede hinsichtlich der benötigten Pakete und absolut erzielbarer Klassifizierungsergebnisse.

Die Kurzbetrachtung von OPC UA Pubsub zeigt hinsichtlich der Klassifizierbarkeit gegenüber anderen UDP-Anwendungen ebenfalls Potenzial. Auch wenn im Rahmen dieser Untersuchung nur simulierte Kommunikationen zur Verfügung standen, sticht die zyklische und unidirektionale Datenübertragung von PubSub hervor, so dass die erzielte Klassifikationsgüte gegenüber Client/Server höher ausfällt.

Es gilt zu erwähnen, dass die Anzahl berücksichtigter Stichproben im Kontext des maschinellen Lernens mit 400 (Client/Server) bzw. knapp 250 Verbindungen (PubSub) vergleichsweise niedrig ausfällt. Dahinter steckt die Absicht, das Gleichgewicht beider Klassen nicht zu stark zuungunsten der begrenzten Anzahl von OPC-UA-Mitschnitten zu verschieben. Da ein Lernmodell jedoch in jedem Fall von einem umfangreicheren Datensatz profitiert und belastbarere Ergebnisse erzeugt, ist perspektivisch die Einbeziehung zusätzlicher Kommunikationsmitschnitte geplant. Die Auswahl von Anwendungen, welche die negative Klasse bilden, ist ebenfalls noch nicht ausgereizt und bietet in Hinblick auf das industrielle Umfeld noch Optimierungspotenzial.

Abseits des gewählten Vorgehens der Klassifizierung von Nutzdatenkommunikation bildet die Analyse des Verbindungsaufbaus einen weiteren vielversprechenden Ansatz zur Identifizierung von verbindungsorientierten Anwendungen. Unterschiedliche Anwendungen weisen für diesen häufig einzigartige Eigenschaften hinsichtlich der Anzahl notwendiger Schritte (Pakete), der Größenverhältnisse ausgetauschter Pakete, sowie dem zeitlichen Ablauf auf. Im Gegensatz zu einer Klassifizierung anhand des Nutzdatentransports, ist die Anzahl zu berücksichtigender Pakete im Rahmen des Verbindungsaufbaus deutlich geringer. Durch die Verkürzung der (notwendigen) „Mitschneideintervalle“, sowie die Reduzierung notwendiger Identifikationsmerkmale, kann der Erkennungsprozess beschleunigt werden und ist ggf. gänzlich ohne Rückgriff auf maschinelle Lernverfahren realisierbar.

Diese Idee wurde allerdings nicht weiter verfolgt, da die vorhandenen Mitschnitte verbindungsorientierten OPC-UA-Verkehrs, welche aus verschiedenen Quellen stammen, hinsichtlich des Ver-

bindungsaufbaus unterschiedliche Verhaltensweisen zeigen. Während in einigen Fällen der gesamte OPC-UA-Verbindungsaufbau zehn Schritte umfasst, gibt es andere Fälle, in denen 15 Schritte und der zwischenzeitliche Aufbau einer neuen TCP-Verbindung notwendig sind.

Als mögliche Ursache für diese Diskrepanz steht die Vermutung im Raum, dass ein Teil der initiierenden Clients bereits vorkonfiguriert waren, andere wiederum nicht. Sollte sich zeigen, dass ein standardisierter Ablauf für den Verbindungsaufbau im industriellen Umfeld existiert, wäre es definitiv sinnvoll diesen Ansatz erneut aufzugreifen.

6 Ausblick

Die Entwicklung eines Prototypen, welcher die Anwendungserkennung implementiert, war im Rahmen der Abschlussarbeit aus zeitlichen Gründen nicht möglich. Dieser Schritt ist im Anschluss an diese Arbeit geplant, um eine (erste) Abschätzung bzgl. der Performanz und der Erkennungsgenauigkeit unter praxisnahen Bedingungen treffen zu können.

Aktuell ist noch nicht geklärt, wie der Prototyp im Netz integriert werden soll. Die Integration als Dienst in der Application Plane folgt dem SDN-Konzept und wäre als reine Softwarelösung implementierbar. Unbekannte Kommunikationsbeziehungen werden in diesem Szenario durch die beteiligten Switches als Paketprobe an den SDN-Controller gesendet und von dort an die Application Plane übergeben. Durch die Anwendungserkennung identifizierte OPC-UA-Verbindungen werden im Anschluss dem SDN-Controller gemeldet, der diese in entsprechende Regeln übersetzt und der Data Plane mitteilt.

Alternativ wäre eine Integration direkt an einem beteiligten Switches z. B. via Port Mirroring möglich. Diese Variante bietet den Vorteil eines kürzeren Signalweges, da die Anwendungserkennung in diesem Fall unmittelbaren Zugriff auf den Netzwerkverkehr hat (App -> Controller -> Switch). Außerdem wäre es so möglich eine beliebige Anzahl von Paketen zu analysieren, während noch unklar ist, wie sich die Paketweitergabe von OpenFlow Switches an den SDN-Controller konkret darstellt.

Literatur

[1] Wollschlaeger, M. ; Sauter, T. ; Jasperneite, J.: The Future of Industrial Communication: Automation Networks in the Era of the Internet of Things and Industry 4.0. In: *IEEE Industrial Electronics Magazine* 11 (2017), March, Nr. 1, S. 17–27

[2] VDE: *Vertikale Integration in der Automation - von der Feldebene zur Unternehmenssteuerung*. September 2007

[3] Danielis, Peter ; Skodzik, Jan ; Altmann, Vlado ; Bjoern Schweissguth, Eike ; Golatowski, Frank ; Timmermann, Dirk ; Schacht, Jörg: Survey on real-time communication via Ethernet in industrial automation environments. In: *19th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2014* 2014 (2015), 01

[4] Trick, Ulrich ; Weber, Frank: *SIP und Telekommunikationsnetze*. Gruyter, Walter de GmbH, Oldenburg 2015

[5] Greene, Kate: 10 Breakthrough Technologies: Software-defined Networking. In: *MIT Technology Review* (2009). `http://www2.technologyreview.com/news/412194/tr10-software-defined-networking/`

[6] Bakhshi, Taimur: *State of the Art and Recent Research Advances in Software Defined Networking*. 2017

[7] Ronayne, J.: *Introduction to Digital Communications Switching*. Pitman, 1991

[8] Raymond, Eric S.: *The Art of UNIX Programming*. 1. Aufl. Boston : Addison-Wesley Professional, 2003. – ISBN 978–0–132–46588–5

[9] Case, J. ; Fedor, M. ; Schoffstall, M. ; Davin, J.: *A Simple Network Management Protocol (SNMP)*. Mai 1990

[10] Yang, L. ; Dantu, R. ; Anderson, T. ; Gopal, R.: *Forwarding and Control Element Separation (ForCES) Framework*. RFC 3746, April 2004

[11] Greenberg, Albert ; Hjalmtysson, Gisli ; Maltz, David ; Myers, Andy ; Rexford, Jennifer ; Xie, Geoffrey ; Yan, Hong ; Zhan, Jibin ; Zhang, Hui: A clean slate 4D approach to network control and management. In: *ACM SIGCOMM Computer Communication Review* 35 (2005), 10, S. 41–54

[12] Casado, Martin ; Garfinkel, Tal ; Akella, Aditya ; Freedman, Michael J. ; Boneh, Dan ; McKeown, Nick ; Shenker, Scott: SANE: A Protection Architecture for Enterprise Networks. In: *Proceedings of the 15th Conference on USENIX Security Symposium - Volume 15*. Berkeley, CA, USA : USENIX Association, 2006 (USENIX-SS'06)

[13] Casado, Martin ; Freedman, Michael ; Pettit, Justin ; Luo, Jianying ; McKeown, Nick ; Shenker, Scott: ETHANE: Taking Control of the Enterprise, 2007, S. 1–12

[14] Bansal, Manu ; Mehlman, Jeffrey ; Katti, Sachin ; Levis, Philip: OpenRadio: A programmable wireless dataplane. In: *HotSDN'12 - Proceedings of the 1st ACM International Workshop on Hot Topics in Software Defined Networks* (2012)

[15] Erran Li, Li ; Mao, Zhuoqing ; Rexford, Jennifer: Toward Software-Defined Cellular Networks, 2012. – ISBN 978–1–4673–4554–5, S. 7–12

[16] Mi, Xiang ; Tian, Zhigang ; Xu, Xibin ; Zhao, Ming ; Wang, Jing: NO stack: A SDN-based framework for future cellular networks. In: *International Symposium on Wireless Personal Multimedia Communications, WPMC* 2015 (2015), 01, S. 497–502

[17] Raschellà, A. ; Bouhafs, F. ; Seyedebrahimi, M. ; Mackay, M. ; Shi, Q.: A Centralized Framework for Smart Access Point Selection based on the Fittingness Factor, 2016

[18] Bizanis, N. ; Kuipers, F. A.: SDN and Virtualization Solutions for the Internet of Things: A Survey. In: *IEEE Access* 4 (2016), S. 5591–5606

[19] Patel, Parveen ; Bansal, Deepak ; Yuan, Lihua ; Murthy, Ashwin ; Greenberg, Albert ; Maltz, David ; Kern, Randy ; Kumar, Hemant ; Zikos, Marios ; Wu, Hongyu ; Kim, Changhoon ; Karri, Naveen: Ananta: Cloud Scale Load Balancing. In: *ACM SIGCOMM Computer Communication Review* 43 (2013), 08

[20] Natarajan, S. ; Ramaiah, A. ; Mathen, M.: A Software defined Cloud-Gateway automation system using OpenFlow. In: *2013 IEEE 2nd International Conference on Cloud Networking (CloudNet)*, 2013, S. 219–226

[21] Das, Saurav ; Parulkar, Guru M. ; McKeown, N. E.: Unifying Packet and Circuit Switched Networks. In: *2009 IEEE Globecom Workshops* (2009), S. 1–6

[22] Channegowda, Mayur ; Nejabati, Reza ; Simeonidou, Dimitra: Software-Defined Optical Networks Technology and Infrastructure: Enabling Software-Defined Optical Network Operations. In: *J. Opt. Commun. Netw.* 5 (2013), Nr. 10, S. A274–A282

[23] Gudla, V. ; Das, S. ; Shastri, A. ; Parulkar, G. ; McKeown, N. ; Kazovsky, L. ; Yamashita, S.: Experimental demonstration of OpenFlow control of packet and circuit switches. In: *2010 Conference on Optical Fiber Communication (OFC/NFOEC), collocated National Fiber Optic Engineers Conference*, 2010, S. 1–3

[24] Mahnke, Wolfgang ; Leitner, Stefan-Helmut ; Damm, Matthias: *OPC Unified Architecture*. Springer-Verlag GmbH, 2009

[25] ascolab GmbH: *OPC UA Neuerungen*. `http://www.ascolab.com/de/unified-architecture/innovation.html`. – Abrufdatum: 12.07.2019

[26] GmbH, Unified A.: *Address Space Concepts*. `https://documentation.unified-automation.com/uasdkhp/1.0.0/html/_l2_ua_address_space_concepts.html`. Version: 2019. – [Abrufdatum: 08.10.2019]

[27] OPC Foundation: *OPC Unified Architecture - Part 14: PubSub*. Februar 2018

[28] ASCOLAB GMBH: *OPC UA Protokolle*. `http://www.ascolab.com/de/unified-architecture/protokolle.html`. – Abrufdatum: 11.07.2019

[29] OPC FOUNDATION: *OPC Unified Architecture - Part 2: Security Model*. August 2018

[30] HERKOMMER, Günter: *OPC UA PubSub-Spezifikation veröffentlicht*. `https://www.computer-automation.de/feldebene/vernetzung/artikel/152463/`. Version: 2018. – Abrufdatum: 11.07.2019

[31] OPC FOUNDATION: *OPC Unified Architecture - Part 6: Mappings*. August 2018

[32] BAKHSHI, Taimur ; GHITA, Bogdan: *On Internet Traffic Classification: A Two-Phased Machine Learning Approach*. 2016

[33] COTTON, Michelle ; EGGERT, Lars ; TOUCH, Dr. Joseph D. ; WESTERLUND, Magnus ; CHESHIRE, Stuart: *Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry*. RFC 6335, August 2011 (Request for Comments)

[34] YAN, Jinghua ; YUAN, Jing: A Survey of Traffic Classification in Software Defined Networks, 2018, S. 200–206

[35] MOORE, Andrew W. ; PAPAGIANNAKI, Konstantina: Toward the Accurate Identification of Network Applications. In: *PAM*, 2005

[36] MADHUKAR, Alok ; L. WILLIAMSON, Carey: A Longitudinal Study of P2P Traffic Classification, 2006, S. 179–188

[37] KARAGIANNIS, Thomas ; BROIDO, Andre ; FALOUTSOS, Michalis ; C. CLAFFY, Kimberly: Transport Layer Identification of P2P Traffic, 2004, S. 121–134

[38] KARAGIANNIS, Thomas ; PAPAGIANNAKI, Konstantina ; FALOUTSOS, Michalis: BLINC: Multilevel Traffic Classification in the Dark, 2005, S. 229–240

[39] DÖBEL, I. ; LEIS, M. ; MOLINA VOGELSANG, M. ; WELZ, J. ; NEUSTROEV, D. ; PETZKA, H. ; RIEMER, A. ; PÜPING, S. ; VOSS, A. ; WEGELE, M.: *Maschinelles Lernen - Kompetenzen, Anwendungen und Forschungsbedarf*. März 2018

[40] BUJLOW, T. ; RIAZ, T. ; PEDERSEN, J. M.: A method for classification of network traffic based on C5.0 Machine Learning Algorithm. In: *2012 International Conference on Computing, Networking and Communications (ICNC)*, 2012, S. 237–241

[41] MULA-VALLS, Oriol: *A practical retraining mechanism for network traffic classification in operational environments*, Universitat Politècnica de Catalunya, Diplomarbeit, 2011

[42] JAMES, Gareth ; WITTEN, Daniela ; HASTIE, Trevor ; TIBSHIRANI, Robert: *An Introduction to Statistical Learning - with Applications in R*. 1st ed. 2013, Corr. 7th printing 2017. Berlin Heidelberg : Springer Science & Business Media, 2013. – ISBN 978–1–461–47138–7

[43] ERTEL, Wolfgang: *Grundkurs Künstliche Intelligenz - Eine praxisorientierte Einführung*. 3. Springer Vieweg, 2013

[44] Pedregosa, F. ; Varoquaux, G. ; Gramfort, A. ; Michel, V. ; Thirion, B. ; Grisel, O. ; Blondel, M. ; Prettenhofer, P. ; Weiss, R. ; Dubourg, V. ; Vanderplas, J. ; Passos, A. ; Cournapeau, D. ; Brucher, M. ; Perrot, M. ; Duchesnay, E.: Scikit-learn: Machine Learning in Python. In: *Journal of Machine Learning Research* 12 (2011), S. 2825–2830

[45] Buitinck, Lars ; Louppe, Gilles ; Blondel, Mathieu ; Pedregosa, Fabian ; Mueller, Andreas ; Grisel, Olivier ; Niculae, Vlad ; Prettenhofer, Peter ; Gramfort, Alexandre ; Grobler, Jaques ; Layton, Robert ; VanderPlas, Jake ; Joly, Arnaud ; Holt, Brian ; Varoquaux, Gaël: API design for machine learning software: experiences from the scikit-learn project. In: *CoRR* (2013). `http://arxiv.org/abs/1309.0238`

[46] Wroclawski, John T.: *The Use of RSVP with IETF Integrated Services.* RFC 2210. `http://dx.doi.org/10.17487/RFC2210`. Version: September 1997 (Request for Comments)

[47] Baker, Fred ; Black, David L. ; Nichols, Kathleen ; Blake, Steven L.: *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers.* RFC 2474, Dezember 1998 (Request for Comments)

[48] Mirchev, Atanas: Survey of Concepts for QoS improvements via SDN. (2015)

[49] Seddiki, M. S. ; Shahbaz, Muhammad ; Donovan, Sean ; Grover, Sarthak ; Park, Miseon ; Feamster, Nick ; Song, Ye-Qiong: FlowQoS: QoS for the Rest of Us. In: *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking.* New York, NY, USA : ACM, 2014 (HotSDN '14). – ISBN 978–1–4503–2989–7, 207–208

[50] Mingozzi, E. ; Stea, G. ; Callejo-Rodríguez, M.A. ; Enríquez-Gabeiras, J. ; Blas, G. G. ; Ramón-Salquero, F.J. ; Burakowski, W. ; Beben, A. ; Sliwinski, J. ; Tarasiuk, H. ; Dugeon, O. ; Diaz, M. ; Baresse, L. ; Monteiro, E.: EuQoS: End-to-End Quality of Service over Heterogeneous Networks. In: *Computer Communications* 32 (2009), Nr. 12, 1355 - 1370. `http://dx.doi.org/https://doi.org/10.1016/j.comcom.2008.12.013`. – DOI https://doi.org/10.1016/j.comcom.2008.12.013. – ISSN 0140–3664. – Special Issue of Computer Communications on Heterogeneous Networking for Quality, Reliability, Security, and Robustness – Part II

[51] Egilmez, Hilmi E. ; Dane, S. T. ; Bagci, K. T. ; Tekalp, A. M.: OpenQoS: An OpenFlow controller design for multimedia delivery with end-to-end Quality of Service over Software-Defined Networks. In: *APSIPA*, IEEE, 2012, S. 1–8

[52] Ishimori, Airton ; Farias, Fernando ; Cerqueira, Eduardo ; Abelém, Antônio: Control of Multiple Packet Schedulers for Improving QoS on OpenFlow/SDN Networking. In: *Proceedings of the 2013 Second European Workshop on Software Defined Networks.* Washington, DC, USA : IEEE Computer Society, 2013 (EWSDN '13). – ISBN 978–1–4799–2433–2, 81–86

[53] ITEA: *OPTIMUM - OPTimised Industrial IoT and Distributed Control Platform for Manufacturing and Material Handling.* `https://www.optimum-itea3.eu/`. – Abrufdatum: 17.10.2019

[54] eG, OSADL: *The OSADL OPC UA over TSN Project*. https://www.osadl.org/OPC-UA-over-TSN.opcua-tsn.0.html. Version: 2019. – [Abrufdatum: 29.11.2019]

[55] AG, Siemens: *Siemens errichtet digitales Umspannwerk mit IoT-Anwendungen für Glitre Energi Nett*. https://press.siemens.com/global/de/pressemitteilung/siemens-errichtet-digitales-umspannwerk-mit-iot-anwendungen-fuer-glitre-energi. Version: 2019. – [Abrufdatum: 29.11.2019]

[56] *Transmission Control Protocol*. RFC 793. http://dx.doi.org/10.17487/RFC0793. Version: September 1981 (Request for Comments)

[57] *User Datagram Protocol*. RFC 768. http://dx.doi.org/10.17487/RFC0768. Version: August 1980 (Request for Comments)

[58] Schulzrinne, Henning ; Casner, Stephen L. ; Frederick, Ron ; Jacobson, Van: *RTP: A Transport Protocol for Real-Time Applications*. RFC 3550. http://dx.doi.org/10.17487/RFC3550. Version: Juli 2003 (Request for Comments)

[59] Projects, The C.: *QUIC, a multiplexed stream transport over UDP*. https://www.chromium.org/quic

[60] Group, IETF QUIC W.: *QUIC Tutorial - A New Internet Transport*. https://www.ietf.org/proceedings/98/slides/slides-98-edu-sessf-quic-tutorial-00.pdf. Version: März 2017

[61] Lohninger, Hans: *Grundlagen der Statistik*. Epina GmbH, 2012

[62] developers scikit-learn: *User Guide: Preprocessing data*. https://scikit-learn.org/stable/modules/preprocessing.html. Version: 2007 - 2019. – [Abrufdatum: 16.10.2019]

[63] Box, G. ; Cox, D.: An Analysis of Transformations. In: *Journal of the Royal Statistical Society, Series B* 26 (1964), 01, S. 211–252

[64] Powers, David: Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation. In: *Mach. Learn. Technol.* 2 (2008), 01