HAW HAMBURG

MASTERTHESIS
Aaron Braatz

Analyse und Prognose von Feinstaubdaten auf Basis von crowd-based Sensornetzen mit KI Verfahren

FAKULTÄT TECHNIK UND INFORMATIK
Department Informatik

Faculty of Computer Science and Engineering
Department Computer Science

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN HAMBURG
Hamburg University of Applied Sciences

Aaron Braatz

Analyse und Prognose von Feinstaubdaten auf Basis von crowd-based Sensornetzen mit KI Verfahren

Masterarbeit eingereicht im Rahmen der Masterprüfung
im Studiengang *Master of Science Informatik*
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Kai von Luck
Zweitgutachter: Prof. Dr. Jan Sudeikat

Eingereicht am: 04. Mai 2022

Aaron Braatz

Thema der Arbeit

Analyse und Prognose von Feinstaubdaten auf Basis von crowd-based Sensornetzen mit KI Verfahren

Stichworte

raumzeitliche Datenanlyse, crowd-based Sensornetze, räumliche Interpolation, Sensorkalibrierung, Deep Learning, ConvLSTM

Kurzzusammenfassung

Klima- und Umweltschutz stehen zunehmend im Fokus der Öffentlichkeit. Die Politik reagiert darauf mit Gesetzen und Richtlinien, die von staatlichen Institutionen umgesetzt werden. Diese erfassen klima- und umweltrelevante Daten in hochspezialisierten Messeinrichtungen, die allerdings nur mit großen Abständen errichtet werden.
Crowd-based Sensornetze bieten die Möglichkeit, ergänzend Messungen mit günstigeren Sensoren vorzunehmen. Diese Daten müssen für die weitere Verwendung aufbereitet werden, damit sie als Grundlage für Analysen und Vorhersagen genutzt werden können.
In dieser Arbeit wird ein Analyseprozess vorgestellt, welcher es ermöglicht, günstige Sensoren nachträglich auf Basis von umliegenden Referenzstationen zu kalibrieren.. Weiter wird räumliche Interpolation genutzt, um die ungleichmäßig verteilten Sensordaten zu einem einheitlichen Raster zu schätzen. Dieses Raster wird im letzten Schritt genutzt, um kurzfristige Prognosen für die Feinstaubentwicklung mittels eines ConvLSTM-Netzes zu erstellen.

Aaron Braatz

Title of Thesis

Analysis and Prediction of particulate matter based on crowd-based sensor network with AI

Keywords

spatiotmporal data analysis, crowd-based sensor network, spatial interpolation, sensor calibration, deep learning, ConvLSTM

Abstract

Climate and environmental protection are increasingly becoming the focus of public interest. Politicians are responding to this with laws and guidelines that are implemented by state institutions. These collect climate- and environment-relevant data in highly specialized measuring devices, which, however, are only set up at great distances from each other.

Crowd-based sensor networks offer the possibility to perform complementary measurements with less expensive sensors. These data need to be processed for further use as a basis for analysis and prediction.

In this paper, an analysis procedure is presented that allows subsequent calibration of low-cost sensors using surrounding reference stations. Furthermore, the unevenly distributed sensor data are estimated by spatial interpolation onto a uniform grid. This grid is used in the final step to generate short-term forecasts for particulate matter development using a ConvLSTM network.

Inhaltsverzeichnis

Abbildungsverzeichnis **vii**

Tabellenverzeichnis **ix**

1 Einleitung **1**
1.1 Motivation . . . 1
1.2 Zielsetzung . . . 1
1.3 Aufbau der Arbeit . . . 2

2 Problemanalyse **3**
2.1 Spatio-Temporal-Data-Mining . . . 3
2.1.1 Anwendungsbereiche . . . 3
2.1.2 Dateneigenschaften . . . 4
2.1.3 Datentypen . . . 5
2.2 Datenanalyseprozess . . . 6
2.2.1 Knowledge-Discovery-in-Databases . . . 7
2.2.2 Knowledge-Discovery-in-Spatio-Temporal-Data . . . 8
2.3 Deep Learning . . . 9
2.3.1 Long Short-Term Memory . . . 10
2.3.2 Convolutionale Neural Network . . . 11
2.3.3 Convolutional LSTM Networks . . . 12
2.4 Crowd-based Senornetze . . . 12
2.4.1 Low-cost sensors . . . 12
2.4.2 Topologie . . . 14

3 Datenanalyse **16**
3.1 Sensor.Community . . . 16
3.1.1 Feinstaubdaten . . . 17
3.1.2 Temperatur- & Luftfeuchtigeitdaten . . . 19

3.2 Umweltbundesamt . . . 19
3.2.1 Feinstaubdaten . . . 20
3.2.2 Stationsinformationen . . . 22
3.3 Deutscher Wetterdienst . . . 24

4 Umsetzung **26**
4.1 Genereller Versuchsaufbau . . . 26
4.2 Toolchain . . . 27
4.3 Metriken . . . 28
4.3.1 R-Square . . . 29
4.3.2 Root Mean Squared Error . . . 29
4.3.3 Mean Absolut Error . . . 30
4.4 Kalibrierung . . . 30
4.4.1 Material und Methoden . . . 30
4.4.2 Durchführung und Ergebnisse . . . 35
4.5 Interpolation . . . 38
4.5.1 Material und Methoden . . . 41
4.5.2 Durchführung und Ergebnisse . . . 43
4.6 Prognose . . . 48
4.6.1 Material und Methoden . . . 48
4.6.2 Durchführung und Ergebnisse . . . 51

5 Diskussion und Ausblick **57**
5.1 Kalibrierung der Sensordaten . . . 57
5.2 Interpolation zu einem gleichmäßigen Raster . . . 58
5.3 Kurzfristige Prognose der Feinstaubentwicklung . . . 59
5.4 Generalisierung und Übertragbarkeit . . . 60

6 Danksagung **61**

Literaturverzeichnis **62**

A Anhang **69**

Selbstständigkeitserklärung **72**

Abbildungsverzeichnis

2.1 Schritte des KDD-Prozesses[Fayyad et al, 1996a, Abb. 1] 8
2.2 ST Data Minig Ablauf von Dateninstanzen und Repräsentationen von unterschiedlichen ST Datentypen zu den anwendbaren DL Modellen. Grafik auf Basis von [Wang et al, 2020, Abb. 5, 12 & 13] und [Atluri et al, 2018, Abb. 4] 9

3.1 Fehlende Werte je Sensor, größere Ausfälle sind besonders hervorgehoben: a) SDS011; b) DHT22; c) BME280 18
3.2 Verteilung der Messwerte für P_{10} und $P_{2.5}$ in den Daten des UBA für Werte größer 0,01 $\mu g/m^3$ 21
3.3 Entwicklung der Anzahl an Messstationen: a) aller UBA Luftmessstationen, b) aller PM-Messstationen, c) der P_{10}-Messstationen, d) der $P_{2.5}$-Messstationen. 23
3.4 Standorte der Messstationen: a) aller UBA Luftmessstationen, b) aller PM-Messstationen. 24

4.1 Ablauf des Versuchs 27
4.2 Darstellung der verfügbaren Sensorstation der Sensor.Community (rot) und Referenzstation des Umweltbundesamts (blau: P_{10} (P1), blau-umrandet: $P_{2.5}$ (P2)) am 31.10.2021 23:00. Die nächsten Referenzstationen für P_{10} und $P_{2.5}$ zu einer Sensorstation sind durch Linien gekennzeichnet. Die Farbe der Linien repräsentiert die Entfernung. Die Entfernung zwischen den Stationen kann über 70 km betragen. 32
4.3 Darstellung der Verteilung der Metriken je Kalibrierungsmodell und Parameterkombination für P_{10} (P1) und $P_{2.5}$ (P2) über alle Zeitstempel. Je Grafik sind die sechs Kalibrierungsmodelle aufgezeichnet. In den Zeilen sind die betrachteten Entfernungen dargestellt und in den Spalten die Metriken (MAE, RMSE und R2) 36

4.4 Verteilung der ursprünglichen Messungen, Werte nach der Kalibrierung (_-cal) und die Referenzmessungen (_ref) jeweils für P_{10} (P1) und $P_{2.5}$ (P2) 39
4.5 Vergleich zwischen den kalibrierten (P1_cal, P2_cal) und ursprünglichen Messungen (P1, P2) zu den Messungen der Referenzstationen. a) P_{10}; b) $P_{2.5}$ 40
4.6 Vergleich der Interpolationsmodelle (IDW, OK, RBF(Multiquadratic, RBF(Linear)) für die kalbrierten Feinstaubmesswerte P_{10} (P1_cal) und $P_{2.5}$ (P2_cal). In den Zeilen sind die Metriken MAE, RMSE und R2. Die Verteilung beschreibt die Metriken für die betrachten Zeitschritte. Der Wertebereich der Metriken wurde zur bessere Übersicht begrenzt. Insbesondere OK hat extreme Ausreißer. 45
4.7 Interpolationsergebnisse für P_{10} in Hessen. Verglichen werden unterschiedliche Zellgrößen des zu schätzenden Rasters. 49
4.8 Interpolationsergebnisse für $P_{2.5}$ in Hessen. Verglichen werden unterschiedliche Zellgrößen des zu schätzenden Rasters. 50
4.9 Ausschnitt aus den Prognosen für P_{10} (P1_cal) mit den beiden besten Modellen unterschieden durch den Aufbau des Schichten des Neuronalen Netzes (5x64 und 3x64). Zusätzlich sind die ursprünglichen Daten versetzt um eine Stunde dargestellt (shifted). 56

A.1 Darstellung der Verteilung der Metriken je Kalibrierungsmodell und Parameterkombination für P_{10} (P1) und $P_{2.5}$ (P2) über alle Zeitstempel. Wobei die fehlenden Luftfeuchtigkeitswerte mit **relativer Luftfeuchtigkeit** ersetzt wurden. Je Grafik sind die sechs Kalibrierungsmodelle aufgezeichnet. In den Zeilen sind die betrachteten Entfernungen dargestellt und in den Spalten die Metriken (MAE, RMSE und R2). Für den Zeitraum 15.12.2017-15.01.2018. 70
A.2 Darstellung der Verteilung der Metriken je Kalibrierungsmodell und Parameterkombination für P_{10} (P1) und $P_{2.5}$ (P2) über alle Zeitstempel. Wobei die fehlenden Luftfeuchtigkeitswerte mit **absoluter Luftfeuchtigkeit** ersetzt wurden. Je Grafik sind die sechs Kalibrierungsmodelle aufgezeichnet. In den Zeilen sind die betrachteten Entfernungen dargestellt und in den Spalten die Metriken (MAE, RMSE und R2). Für den Zeitraum 15.12.2017-15.01.2018. 71

Tabellenverzeichnis

4.1 Ausschnitt der Daten, welche für die Kalibrierung genutzt wurden. Feinstaub-, Temperatur- und Luftfeuchtigkeitsdaten wurden anhand von den Geokoordinaten und dem Zeitstempel zusammengeführt. Fehlende Temperatur- und Luftfeuchtigkeitsdaten wurden von der nächstgelegenen DWD-Messstation ergänzt 33
4.2 Aufstellung der Modellnamen der Kalibrierungsmodelle und die jeweils genutzten Feuturesets, wobei PM der jeweilige Feinstaubmesswert ist, T ist die Temperatur und H ist die relative Luftfeuchtigkeit. 37
4.3 Kalibrierungsmetriken nach der Kalibrierung des gesamten Datensatzes. . 38
4.4 Mittelwert der Metriken der Interpolationsmodelle je kalibriertem P_{10} (P1_cal) und $P_{2.5}$ (P2_cal) über alle betrachteten Zeitschritte. 44
4.5 Die benötigte Berechnungsdauer für die Interpolation für verschiedene Rastergrößen am Beispiel Hessen für den Zeitraum von einem Monat . . . 47
4.6 Übersicht der besten Prognose Modelle basierend auf dem RMSE der Validierungsdaten 54
4.7 Vergleich der Metriken der beiden besten Prognosemodelle basierend auf dem RMSE der Validierungsdaten und dem naiven Ansatz des letzten Messwerts 55

1 Einleitung

1.1 Motivation

Das gesellschaftliche Bewusstsein bezüglich Klima- und Umweltschutz ist in den vergangenen Jahren stetig gestiegen. Trotz der medialen Aufmerksamkeit der Corona Pandemie halten 65 % der deutschen Bevölkerung das Thema Klima- und Umweltschutz weiterhin für relevant, welches Auswirkungen auf die politische Situation beinhaltet. [Umweltbundesamt, 18.04.2022b]. Das Umweltbundesamt ist die politische Institution, die diese Themen hinsichtlich gesetzlicher Richtlinien betreut und umsetzt. Die gesundheitlichen Auswirkungen einer hohen Belastung durch Feinstaub sind mittlerweile wissenschaftlich hinreichend belegt [Umweltbundesamt, 18.04.2022a]. Als Überwachungsinstrument exisiert in Deutschland ein Feinstaubmessnetz, welches vom Umweltbundesamt stetig weiter ausgebaut wird und mittlerweile etwa 400 Stationen umfasst.
Das crowd-based Projekt Sensor.Community ermöglicht es jeder Person eine eigene Sensorstation aufzustellen und Feinstaub zu messen. Durch das Einbinden der Bevölkerung konnten bereits über 5.000 Sensorstationen errichtet werden, welche insbesondere in städtischen Gebieten ein dichtes Messnetz bilden.
Die Forschung beschäftigt sich zunehmend mit der Nutzung von Low-Cost-Sensoren, wie sie in crowd-based Sensornetzen vorkommen und es konnten bereits erste Erfolge erzielt werden [Giordano et al, 2021].

1.2 Zielsetzung

In dieser Arbeit wird ein Prozess für die Analyse und Prognose von crowd-based Sensornetzen am Beispiel von Feinstaubdaten erarbeitet. Im ersten Schritt werden die Sensoren mit Hilfe von Referenzstationen nachträglich kalibriert. Dazu werden unterschiedliche Modelle untersucht. In einem zweiten Schritt werden die Daten der ungleichmäßig verteilten Sensorstationen interpoliert, um gleichmäßige Raster an Daten zu generieren,

auf denen Prognosen erstellt werden können. Hierfür werden drei räumliche Interpolationsmodelle verglichen. Schlussendlich wird ein Vorhersagemodell für die kurzfristige Prognose der Feinstaubentwicklung erstellt.

1.3 Aufbau der Arbeit

In Kapitel 2 werden mögliche Anforderungen herausgearbeitet und verwandte Forschungsarbeiten und Technologien betrachtet. Die genutzten Daten werden in Kapitel 3 genau untersucht und Herausforderungen adressiert. In Kapitel 4 wird die Umsetzung der Versuche beschrieben. Hierbei wird zunächst allgemein auf alle Schritte des Analyseprozesses eingegangen und im Anschluss spezifisch die Kalibrierung, Interpolation und Prognose betrachtet. In den Kapiteln 5 werden die Ergebnisse diskutiert und und ein Fazit gezogen.

2 Problemanalyse

In diesem Kapitel werden bestehenden Forschungsansätze analysiert und Herausforderungen, sowie bereits bestehenden Lösungsansätze vorgestellt. Außerdem wird die inhaltliche Grundlage für genutzten Technologien geschaffen.

2.1 Spatio-Temporal-Data-Mining

Spatio-Temporal-Data-Mining (STDM) (deutsch: raumzeitliches Data-Mining) dient der (halb-) automatisierten Analyse von Daten, die in einem raumzeitlichen Kontext stehen. Jeder Art von Daten kann ein zeitlicher und räumlicher Kontext gegeben werden. Zum einen werden Daten zu einem bestimmten Zeitpunkt gemessen, erfasst oder verändert, welches die Daten zeitlich einordnet. Zum anderen werden Messungen an einem Ort auf der Welt oder im Universum aufgenommen. Ein Detail befindet sich in einem bestimmten Teil eines Bildes oder eine URL identifiziert eine Adresse im Internet. Die räumliche Zuordnung auf sozialen Plattformen ist abstrakter. Dort sind Freunde und Freundes Freunde über ein Netzwerk miteinander verbunden. Sofern für die Daten der räumliche und zeitliche Bezug mit erfasst wird, können Methoden des STDM genutzt werden, um Einblicke in die Zusammenhänge zu gewinnen [Atluri et al, 2018].

2.1.1 Anwendungsbereiche

Wie bereits erläutert, haben Daten prinzipiell immer Spatio-Temporal (ST)-Informationen. Daher wird STDM auch in diversen Domänen genutzt:

- In der **Klimaforschung & Umweltwissenschaft** wird STDM für ein besseres Verständnis von Klima- & Umweltfaktoren und der Klimaentwicklung genutzt, um beispielsweise potentielle Gefahren und Umweltbelastungen frühzeitig zu erkennen und diesen entgegenwirken können.

- Die **Neurowissenschaft** nutzt Daten aus bildgebenden Verfahren, um die Funktionsweise des Gehirns zu verstehen. Dadurch Diagnosen verbessert und Therapien konzipiert werden.
- In der **Landwirtschaft** werden Satelliten- und Drohnenbilder genutzt, um auf deren Basis effizient Bereiche von Agrarflächen zu düngen und Pflanzenkrankheiten zu identifizieren
- Die **Epidemologie** analysiert die Ausbreitung von Krankheiten mit Hilfe von Daten aus Krankenhäusern. Dadurch lassen sich Konzepte zur Eindämmung oder Präventions von Krankheitswellen entwickeln.
- Durch Beiträge und Freundesnetze in **Social Media** können sozial-politische Bewegungen analysiert werden. *Location Based Social Networks* (deutsch: Ortsbezogenes soziales Netzwerke) nutzen zusätzlich die GPS-Informationen der Beiträge. Bei Flickr [1] können beispielsweise GPS-Informationen zu den geteilten Fotos ergänzt werden.
- Im Bereich **Verkehr & Transport** werden zum einen Verkehrsaufkommen analysiert, damit bei der Stadtplanung Überlastungen von Verkehrswegen entgegengewirkt werden kann. Zum anderen wird die Bewegung von Personen und Transportgütern ausgewertet, damit diese möglichst effizient an ihr Ziel gelangen.
- Die **Kriminologie** nutzt STDM, um Muster in Verbrechen zu erkennen und Risikogebiete vorherzusagen. Damit können Einsatzkräfte effizienter genutzt und die Kriminalität reduziert werden.
- **On-Demand-Services**, wie Stadträder oder Ride-Sharing-Dienstleister nutzen STDM. Dabei werden die Bedarfe an verschiedenen Orten analysiert und dazu passend die Routen und Kapazitäten geplant.

2.1.2 Dateneigenschaften

ST-Daten unterscheiden sich von anderen Daten durch die Eigenschaften *Autokorrelation* und *Heterogenität*:

[1] https://www.flickr.com/ (31.12.2021)

Autokorrelation beschreibt den Zusammenhang zwischen Datenpunkten in Bezug auf deren Entfernung zueinander, sowohl im Raum als auch in der Zeit. [Tobler, 1970] verfasste das sogenannte erste Gesetz der Geographie:

> *„Everything is related to everything else but nearby things are more related than distant things"* [Tobler, 1970, S. 236]

Dementsprechend ist zum Beispiel zu erwarten, dass eine Messung der Lufttemperatur ähnlich zu einer weiteren ist, die wenige Meter entfernt erfasst wird oder eine Minute später. Diese Eigenschaft ist gegensätzlich zu der üblichen Annahme des *Data Mining* (DM), dass Datenpunkte unabhängig voneinander sind. Zwei zufällig gewählte Bilder von Katzen oder Hunden können nur anhand des jeweiligen Bildes passend kategorisiert werden. Das andere Bild trägt ohne zusätzliche Informationen nicht zur Einordnung bei.

Heterogenität bedeutet, dass eine Teilmenge der Daten nicht zwangsläufig repräsentativ für alle Daten ist. Dies bezieht sich auf Daten, die außerhalb des Autokorrelationsbereichs liegen. Am Beispiel der Lufttemperatur sind Messungen im Sommer nicht repräsentativ für Messungen im Winter [Atluri et al, 2018].

2.1.3 Datentypen

Datentypen beschreiben die Untergruppen der ST-Daten, die unterschiedliche Eigenschaften haben. Die Datentypen werden dabei wie folgt kategorisiert:

- **Event data** (deutsch: Ereignisdaten) sind Aufzeichnungen von Ereignissen in Raum und Zeit. Beispiele hierfür sind Straftaten, Erdbeben, Vulkanausbrüche oder Einsteige- und Aussteige-Momente bei einer Taxifahrt.
- **Trajectory data** (deutsch: Tajektorien) beschreiben fortlaufende Messungen, während sich entweder das Messgerät oder das Messobjekt bewegt. Dabei werden sowohl Messwerte, als auch die Positionen über die Zeit erfasst. Hieraus resultiert folglich auch die Bewegungsrichtung.
- **Point reference data** (deutsch: Punkt-Referenzdaten) sind Messungen, die an einem gleichbleibenden Ort über die Zeit aufgenommen werden. Dazu gehören die Messungen einer Sensorstation, wie zum Beispiel von einer Wetterstation.

- **Raster data** (deutsch: Rasterdaten) bilden die räumliche Struktur zu einem Zeitpunkt ab. Das können beispielsweise Satelliten- oder Röntgenbilder sein, bei denen die räumliche Struktur gleichmäßig in Pixeln definiert ist. Netzwerke werden auch in Rasterdaten erfasst. Dabei sind unregelmäßig Knoten über Kanten verbunden.
- **Videos** sind Bilder in zeitlicher Abhängigkeit. Dementsprechend werden bei einem Video Rasterdaten in einen zeitlichen Kontext gebracht.

Diese Datentypen haben korrespondierende Dateninstanzen und -repräsentationen, welche wiederum entscheidend sind für die Analysemöglichkeiten. [Wang et al, 2020] und [Atluri et al, 2018] beschreiben die Beziehungen ausführlich, welche auch in Abbildung 2.2 auf Seite 9 dargestellt sind.

2.2 Datenanalyseprozess

Daten werden heutzutage in großen Mengen gesammelt. Im Jahr 2018 betrug die weltweite Datenmenge etwa 33 Zettabyte (ZB). Bis 2025 wird die Datenmenge voraussichtlich auf 175 ZB anwachsen [iwd, 07.06.2019]. Diese Daten werden gesammelt, da von ihnen ein Informationsgewinn erwartet wird. Eine Studie von Capgemini aus dem Jahr 2020 zeigt die wirtschaftlichen Vorteile *datengetriebenen Unternehmen*: „Unternehmen, die beim Dateneinsatz führend sind, erzielen 22 Prozent mehr Gewinn und pro Mitarbeiter 70 Prozent mehr Umsatz als ihre Wettbewerber" [Capgemini Deutschland, 2020]. Der wirtschaftliche Vorteil ergibt sich allerdings nicht daraus, dass nur Daten gesammelt werden, sondern auf deren Basis Muster und Zusammenhänge identifiziert werden. Diese können genutzt werden, um Geschäftsabläufe zu optimieren oder neue Geschäftszweige zu eröffnen.
Ein Datenanalyseprozess beschreibt die notwendigen Schritte, um von Rohdaten zu einem Informationsgewinn zu gelangen. Dabei muss das Ziel nicht zwangsläufig ein wirtschaftlicher Vorteil sein. Ebenso kann der Prozess auch genutzt werden, um zu neuen wissenschaftlichen Erkenntnissen zu gelangen.

2.2.1 Knowledge-Discovery-in-Databases

[Fayyad et al, 1996a] stellt den *Knowledge-Discovery-in-Databases* (KDD)-Prozess vor. Eine detaillierte Beschreibung des KDD-Prozesses findet sich in einer vorangegangenen Arbeit [Braatz, 2019]. Daher wird hier nur eine Zusammenfassung vorgestellt.

KDD wird definiert als:

> *Knowledge Discovery in Databases is the nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data.* [Fayyad et al, 1996b, S. 83]

Vor Beginn des Prozesses wird ein Ziel definiert, anhand dessen sich der Prozess orientiert. Dies kann zum Beispiel die Verbesserung einer Vorhersage sein. Der Prozess selbst teilt sich dabei in fünf Schritte auf, die iterativ durchlaufen werden (siehe Abb. 2.1):

- **Datenselektion**: Die Sammlung relevanter Daten, sofern sie vorhanden sind. Andernfalls werden die Daten erhoben.
- **Vorverarbeitung (engl. Preprocessing)**: Die Korrektur falscher Daten, sowie eventuelle Ergänzung fehlender Daten.
- **Datentransformation**: Die Transformation der Daten in ein Format, welches durch ein Data-Mining-Modell verarbeitet werden kann. Zusätzlich kann durch *Feature Selection* die Auswahl der Attribute präzisiert werden.
- **Data Mining**: Die Anwendung eines ausgewählten Data Mining (DM)-Modells. Hieraus resultiert die eigentliche Wissensgewinnung.
- **Interpretation / Evaluation** Die Visualisierung, Diskussion und Interpretation der Ergebnisse des DM.

Entspricht das Ergebnis bei der Interpretation und Evaluation nicht dem vorher definierten Ziel, beginnt eine neue Iteration. Diese kann bei jedem vorhergegangenen Schritt ansetzen.

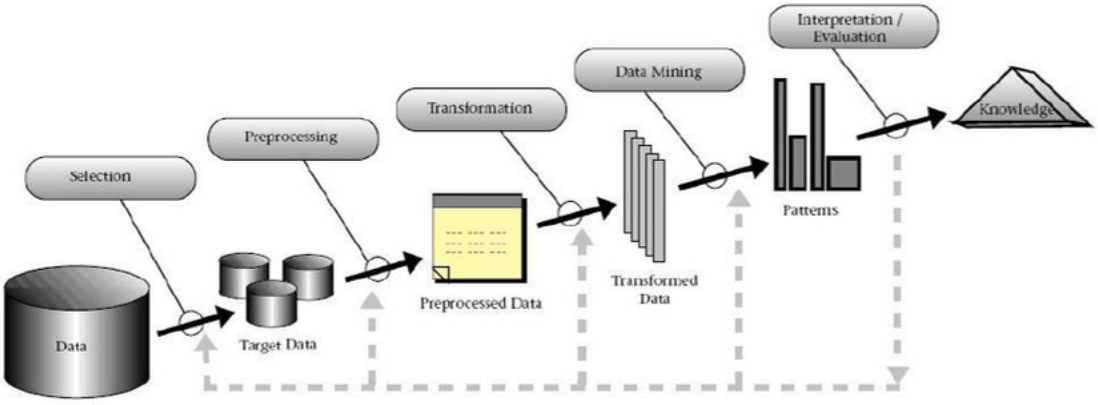

Abbildung 2.1: Schritte des KDD-Prozesses[Fayyad et al, 1996a, Abb. 1]

2.2.2 Knowledge-Discovery-in-Spatio-Temporal-Data

Knowledge-Discovery-in-Spatio-Temporal-Data (KDSTD) wird vom Autor als Begriff eingeführt und beschreibt einen Teil des KDD-Prozesses aus der Perspektive von *Spatio-Temporal-Data (STD)* (deutsch: raumzeitliche Daten). Dabei wird sich an der allgemeinen Pipeline für *Deep-Learning (DL)*-Modelle für ST-Data-Mining (STDM) in [Wang et al, 2020, s. 8] orientiert. Ergänzt wurde der Ablauf durch die Zuordnung von Datentypen zu Dateninstanzen ohne Beschränkung auf DL von [Atluri et al, 2018, S. 83:10]. KDSTD setzt nach der Vorverarbeitung der Daten an, beginnt bei der Transformation und endet mit dem Data-Mining. In Abb. 2.2 wird der Ausschnitt dargestellt. Hierbei sind nur DM-Modelle aus dem DL-Bereich vorgestellt, da in dieser Arbeit letztlich DL-Modelle für die Vorhersage genutzt werden. [Atluri et al, 2018; Kisilevich et al, 1980; Mazimpaka and Timpf, 2016; Li, 2014] beschreiben weitere DM-Modelle ohne Fokus auf DL.
In KDSTD wird die *Datentransformation* in zwei Schritte aufgeteilt. *Data instances construction* dient dazu, das Sensor-bedingte Datenformat in speichereffiziente Dateninstanzen zu transformieren. *Data representation construction* wandelt die Dateninstanzen in ein Format um, welches für das gewählte Modell nutzbar ist. In diesem Schritt kann die „Form" der Daten noch deutlich verändert werden. [Wang et al, 2020] bezeichnet diesen Schritt als *Data preprocessing*. Hierbei wird jedoch nicht wie bisher eine Form der Vorverarbeitung beschrieben, sondern die Umformung der Daten zur finalen Übergabe in das gewählte Modell. Daher wird in dieser Arbeit der Schritt als *Data representation construction* bezeichnet.

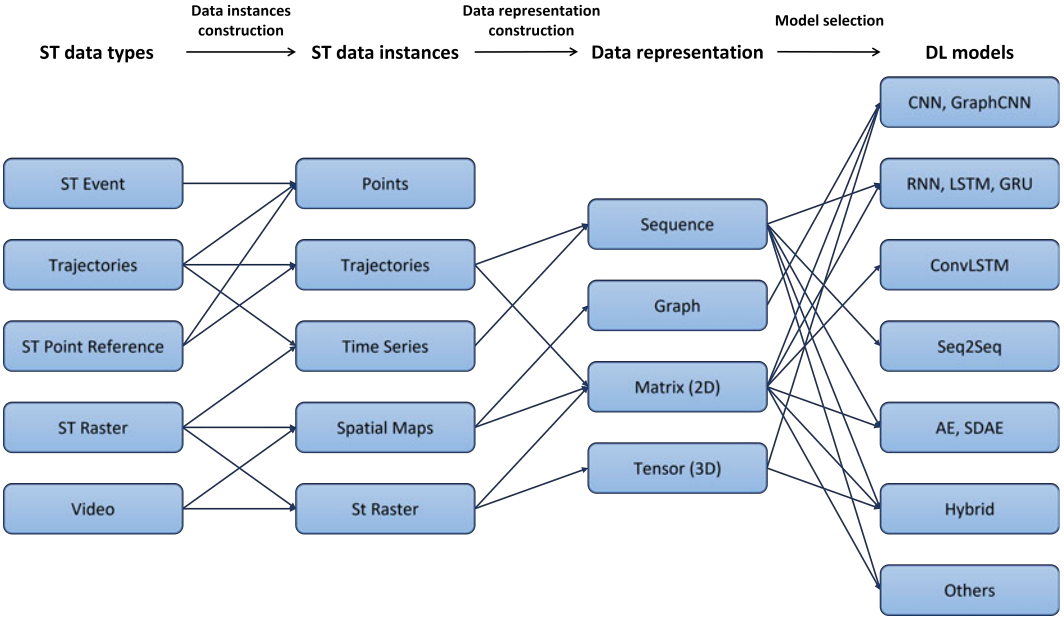

Abbildung 2.2: ST Data Minig Ablauf von Dateninstanzen und Repräsentationen von unterschiedlichen ST Datentypen zu den anwendbaren DL Modellen. Grafik auf Basis von [Wang et al, 2020, Abb. 5, 12 & 13] und [Atluri et al, 2018, Abb. 4]

Zum Beispiel beschreiben [Lv et al, 2018; Wang et al, 2017] die Transformation von Trajektorien zu Matrizen. Dazu teilen sie den betrachteten Bereich in ein gleichmäßiges Raster. Die Teilbereiche werden markiert, sofern eine Trajektorie den Teilbereich durchquert. Auf diese Weise können auch Modelle für den räumlichen Zusammenhang für Trajektorien genutzt werden und nicht nur zeitliche Modelle.

2.3 Deep Learning

Deep Learning (deutsch: tiefes Lernen) ist ein Teilbereich des *Machine Learning* (ML, deutsch: maschinelles Lernen). ML bezeichnet dabei eine Sammlung von Algorithmen, die es ermöglichen, Daten (halb-) automatisch zu analysieren. Diese Automatisierung ermöglicht es, Daten in großer Menge und Komplexität zu untersuchen, welches händisch nur mit erheblichen Aufwand möglich wäre. Für die Analysen werden Modelle anhand von vorhandenen Daten trainiert. Dabei werden drei Trainingsmethoden unterschieden:

1. *Supervised Learning* (deutsch: überwachtes Lernen): Hierbei werden für das Lernen zusätzlich zu den Eingabedaten (engl.: Feature) auch das dazugehörige Ergebnis (engl.: Label) genutzt. Dieses Verfahren ist vergleichbar mit „Lernen durch Nachahmen". Diese Variante wird hauptsächlich für Klassifikations- und Regressionsmodelle genutzt.

2. *Unsupervised Learning* (deutsch: unüberwachtes Lernen): Bei dieser Trainingsmethode werden nur die Daten für sich betrachtet, um beispielsweise Muster zwischen verschiedenen Datenpunkten zu erfassen. Anwendungsbeispiele für *Unsupervised Learning* sind: Dimensionsreduzierung, Clustering (deutsch: Grüppchenbildung) und Dichteschätzung.

3. *Reinforcement Learning* (deutsch: verstärkendes Lernen): Hierbei lernt ein Modell nicht auf Basis eines Datensatz, sondern durch das Feedback einer interaktiven Umgebung. Diese Methode wird genutzt für Policy Search (deutsch: Richtliniensuche) und Bestimmung einer Value Function (deutsch: Belohnungsfunktion).

Nicht alle Algorithmen aus dem Bereich ML gehören zu Methoden des DL. DL-Algorithmen basieren auf *neuronalen Netzen*, welche aus mehreren Schichten von *Perzeptronen* (Zelle eines neuronalen Netzes, angelehnt an die Funktionsweise eines Neurons) bestehen. Mit den heutigen technologischen Mitteln ist es möglich, viele dieser Schichten hintereinander zu schalten und so ein „tiefes" neuronales Netz zu erschaffen. Bei ML wird ein besonderes Augenmerk auf das *Feature Engineering* gelegt, bei dem aus den ursprünglichen Daten Merkmale gewonnen werden, die eine höhere Informationsdichte aufweisen. Diese Aufgabe erfüllen die einzelnen Schichten innerhalb eine DL-Modells. Das Erlernen der Merkmale benötigt allerdings mehr Rechenleistung. [Dargan et al, 2020]

2.3.1 Long Short-Term Memory

Long Short-Term Memory (LSTM) (deutsch: Langzeit-Kurzzeitgedächtnis)-Netze sind eine Weiterentwicklung von *Recurrent Neural Networks* (RNN) (deutsch: Rekurrente oder rückgekoppelte neuronale Netze) und wurden von [Hochreiter and Schmidhuber, 1997] entwickelt. Beide Arten von Netzen dienen der Verarbeitung von Sequenzen, also zeitlichen Abfolgen. Die Besonderheit gegenüber anderen neuronalen Netzen liegt in der zusätzlichen ausgehenden Verbindung eines Perzeptrons wieder zu sich selbst. Daraus resultiert die Namensgebung eines *rückgekoppelten neuronalen Netzes*. Durch diese Verbindung kann das aktuelle Ergebnis im nächsten Zeitschritt mit in die Berechnung

einfließen und vergangene Eingaben haben ebenfalls Einfluss auf kommende Ausgaben. RNNs sind die direkte Umsetzung dieses Prinzips. Allerdings können bei der Verarbeitung von längeren Sequenzen mit RNNs Phänomene wie *Gradient Vanishing* oder *Gradient Exploding*, welches vor allem bei andauernder geringer oder hoher Aktivierung eines RNN Perzeptrons auftreten. Dadurch wird das Lernverhalten beeinträchtigt, sodass ein langfristiger Zusammenhang nicht erfasst werden kann. [Hochreiter et al, 2001]
LSTM-Netze bestehen aus Schichten mit LSTM-Zellen. Der Aufbau der LSTM-Zellen wirkt *Gradient Vanishing* und *Gradient Exploding* entgegen, indem die Aktivierung auf mehrere Teile der LSTM-Zelle aufgeteilt wird. Dafür besteht eine LSTM-Zelle aus vier Perceptron-ähnlichen Teilen.Neben einer *Input Unit* existieren drei sogenannte *Gates*, welche unterschiedliche Informationsflüsse kontrollieren. Die Informationsflüsse sind der Eingang (*Input-Gate*), der Ausgang (*Output-Gate*) und einer, der den internen Zustand kontrolliert (*Forget-Gate*). Der interne Zellzustand wird, neben dem Ergebnis, ebenfalls an den nächsten Zeitschritt übergeben.

2.3.2 Convolutionale Neural Network

Ein *Convolutionale Neural Network* (CNN, deutsch: Neuronales Faltungsnetzwerk) wurde ursprünglich zur Erkennung von handgeschriebenen Postleitzahlen entwickelt [LeCun et al, 1989]. Diese Art von neuronalen Netzen orientiert sich dabei an der Funktionsweise des visuellen Cortex. Das Netz besteht in der Regel aus zwei Arten von Schichten. Die erste Schicht ist der namensgebene *Convolutional layer* (deutsch: Faltungsschicht), bei dem ein oder mehrere *Faltungskernel* auf die Daten angewendet werden. Die Gewichte eines Kernels sind der trainierbare Teil des Netzes. Auf diese Weise wird ein Datenpunkt in einen räumlichen Kontext gesetzt. Die Ergebnisse dieser Schicht sind sogenannte *Feature Maps* für je einen *Faltungskernel*.
Des weiteren gibt es den *Pooling layer* (deutsch: Bündelungsschicht), bei dem die Dimensionalität reduziert und das Ergebnis robuster gegen kleine Verschiebungen und Verfälschungen gemacht wird. Dafür wird üblicherweise das Maximum aus einem Teil der *Feature Map* genommen.
Diese zwei Arten von Schichten lassen sich auch mehrfach hintereinander schalten, um komplexere Formen zu erkennen. CNNs werden nicht nur für 2D Daten, wie beispielsweise Bilder genutzt, sondern auch für 1D Daten, wie beispielsweise Signale und 3D Daten, wie zum Beispiel Videos. [LeCun et al, 2015]

2.3.3 Convolutional LSTM Networks

Convolutional LSTM Networks (ConvLSTM) sind eine Zusammenführung der Konzepte von CNNs und LSTMs von [Shi et al, 2015]. ConvLSTM wurden ursprünglich entwickelt für die kurzfristige Vorhersage von Niederschlägen anhand von Satelliten Bildern und konnte in dem Kontext bessere Vorhersagen als *Real-time Optical flow by Variational methods for Echoes of Radar* (ROVER)-Modelle tätigen (Korrelation zum Label: 0,908 (ConvLSTM); 0,850 (ROVER2)).
[Wang et al, 2020] beschreibt ein ConvLSTM wie folgt:

> *ConvLSTM is a sequence-to-sequence prediction model, whose each layer is a ConvLSTM unit that has convolutional structures in both the input-to-state and state-to-state transitions. The input and output of the model are both spatial map matrices*[Wang et al, 2020, S. 10]

Dabei sind mit *input-to-state and state-to-state transitions* die vier Teile einer klassischen LSTM-Zelle gemeint. Allerdings werden im Fall von ConvLSTM bei den Übergängen nun räumliche Strukturen der Daten betrachtet und nicht jeder Datenpunkt für sich.

2.4 Crowd-based Senornetze

In dieser Arbeit wird ein crowd-based Senornetz für die Messung von Feinstaubdaten betrachtet. Daher werden die Besonderheiten, solcher Netze, in den folgenden Abschnitten in diesem Kontext vorgestellt.

2.4.1 Low-cost sensors

Günstige Feinstaubsensoren ermöglichen eine deutlich flexiblere Überwachung der Feinstaubbelastung. Bis etwa 2010 hatten hauptsächlich staatliche Instanzen oder gut finanzierte Forschungsprojekte die Möglichkeit für Feinstaubmessungen. Die Anschaffungskosten für Messinstrumente lagen im Bereich von 10.000 USD und mit Betriebskosten, die teilweise diesen Betrag noch übersteigen.
[Giordano et al, 2021, S. 3f] stellt eine Auswahl an günstigen Feinstaubsensoren vor sowie eine Übersicht von Projekten, welche die günstigen Sensoren genutzt haben. Die günstigen Sensoren haben zu diversen *Citizen Science*-Projekten geführt (the Airbeam

monitor[2], Plume Labs Flow monitor[3], PurpleAir[4], Sensor.Community[5]) sowie großen, verteilten Sensornetzen in Städten ([Jiao et al, 2016; Rose Eilenberg et al, 2020]).
Hierbei ist zu beachten, dass die Messgeräte von staatlichen Referenzstationen kalibiert sind und dadurch eine höhere Messgenauigkeit besitzen. Zu dem werden die Messungen in einer kontrollierten Umgebung durchgeführt. Günstige Sensoren messen den Feinstaub unter Einfluss der Umgebung und die Messungen können dadurch verfälscht werden. Dabei ist insbesondere die Luftfeuchtigkeit zu beachten, die zu Größenveränderung des Feinstaubs führen kann und so das Messergebnis beeinflusst.[Giordano et al, 2021]
Mithilfe von Kalibrierung der günstigen Sensoren kann dem Umwelteinfluss entgegengewirkt werden.

[Hagler et al, 2018] Zeigt Parameter mit denen eine Kalibrierung möglich ist. Die Temperatur und relative Luftfeuchtigkeit kann grundsätzlich zur Kalibrierung genutzt werden, es ist aber im Einzelfall zu überprüfen, ob bei einem Sensor ein Zusammenhang zu den Parametern festgestellt werden kann. Umliegende (vertrauenswürdige) Sensoren können ebenfalls genutzt werden, wenn sie im näheren Umfeld liegen. Windrichtung / -geschwindigkeit, sowie Verkehrsdaten eignen sich nicht zur Kalibrierung der Feinstaubsensoren. Als zeitlicher Faktor kann die Laufzeit eines Sensors genutzt werden.

[Chu et al, 2020] stellt zwei unterschiedliche Methoden zur Kalibrierung von Low-Cost-Sensoren für $P_{2.5}$ (Feinstaub bis zu einer Größe von $<$2,5 μm) (AirBox PM2.5) vor. Beide Methoden nutzen vertrauenswürdige Referenzstation in der Nähe der Low-Cost-Sensoren für die Kalibrierung. Die erste Methode nutzt ausschließlich die Messungen an den Referenzstation im Vergleich zu den Sensoren und und optimiert die Messungen mittels linearer Regression. Die zweite Methode betrachtet zusätzlich die räumliche Konstellation und Entfernungen zwischen den Stationen und Sensoren. Für die Kalibrierung wird *Geographically Weighted Regression* und *Weighted Least-Squares* genutzt. Die Gewichte werden über eine *Kernelfunktion* bestimmt. Die *Kerneldistanz* wird über eine *Gaußfunktion* definiert, welche die räumliche Autokorrelation abbildet.

[Malings et al, 2020] analysiert zwei Modelle von Low-Cost-Sensoren für PM2.5 (Met-One NPM & PurpleAir PA-II). Für die Kalibrierung der Sensoren wird der Schwerpunkt auf Umwelteinflüße gelegt. Dabei wird zum einen das hygroskopisches Wachstum [Petters

[2] https://www.habitatmap.org/airbeam
[3] https://plumelabs.com
[4] https://www2.purpleair.com/
[5] https://sensor.community/de/

and Kreidenweis, 2007] der Feinstaub-Partikel im Zusammenhang zur Temperatur und Luftfeuchtigkeit betrachtet. Die alleinige Betrachtung des hygroskopisches Wachstums hat aber nur unzureichende Ergebnisse erzielt. Daher wurden zusätzlich Referenzstationen betrachtet und der Fehler zwischen zwischen Station und Sensor mittels linearer Regression minimiert. Zum anderen wurden zusätzlich die Temperatur, Luftfeuchtigkeit und der korrespondierende Taupunkt an den Sensoren genutzt. Zur Kalibrierung wurden lineare und quadratische Modelle auf Basis der Feature trainiert.
Beide Ansätze zeigen eine ähnliche Verbesserung der Sensordaten im Vergleich zu den Referenzstationen. Die Ergebnisse verbesserten sich deutlich bei der Betrachtung eines längeren Zeitraums (ein Jahr oder länger).

2.4.2 Topologie

Dynamische Sensornetze nutzen häufig „Wireless Sensors", da die einzelnen Stationen häufig nicht ortsgebunden sind. Hierbei sind die Sensoren über einen Bereich verteilt und kommunizieren über eine Mesh-Architektur. Die Sensoren kommunizieren ihre Messungen in einer hierachischen Struktur an eine Zentrale. Bei diesem Aufbau ist die Methodik der Weiterleitung der Daten von *Leaf-Nodes* über *Master-Node* zur Zentrale zu beachten. Es wird durchgehend überprüft, ob ein Master erreichbar ist oder ein neuer Master bestimmt werden muss. Zudem verfügen diese Sensoren meist nicht über eine gesicherte Stromversorgung, weshalb Strategien für eine optimale Energieeffizienz entwickelt werden müssen..
Diese Besonderheiten sind in dem betrachteten Sensornetz dieser Arbeit nicht von Bedeutung, da grundsätzlich eine statische Stromversorgung gewährleistet ist. Ebenfalls ist jeder Sensor an ein dediziertes WLAN-Netz angebunden und hat somit einen direkten Uplink. Dennoch kann es passieren, dass vereinzelt keine Daten übermittelt werden können, wie zum Beipiel bei einen Ausfall des WLAN-Netzes. In einigen dynamischen Sensornetzen sind die Sensoren mobil und bewegen sich durch den Raum, sei es auf Fahrzeugen, wie Kraftwagen oder Flugzeugen, sowie auf Bojen. Hierbei müssen durchgehend die Positionsdaten aktualisiert werden. Zudem können sich die Messungen ortsbezogen verändern, welches einen Einfluss auf das Messverhalten hat.
In dem betrachteten Sensornetz sind die Sensoren grundsätzlich stationär und bewegen sich nicht. Es ist jedoch möglich, dass neue Sensorstationen zu jedem beliebigen Zeitpunkt hinzugefügt werden sowie einzelne Sensorstationen entweder entfallen oder keine

Daten übermitteln.[He et al, 2007; Hussein and Stipanovic, 2006; Oteafy and Hassanein, 2017; Zhang, 2011]

3 Datenanalyse

In diesem Kapitel werden die genutzten Datensätze betrachtet und analysiert. Es werden die Besonderheiten herausgearbeitet und beschrieben, inwiefern die Daten vorverarbeitet wurden.

3.1 Sensor.Community

Der zentrale Datensatz, der in dieser Arbeit zum Einsatz kommt, stammt von dem Projekt Sensor.Community[1] (ehem. Luftdaten.info). Sensor.Community ist ein Projekt, welches ursprünglich durch das OK Lab Stuttgart gegründet wurde. Es bietet eine Plattform, für die Erfassung von Umweltdaten durch selbstgebaute Sensoren. Die Anleitung und Firmware für die Sensorstationen werden von dem Projekt zur Verfügung gestellt, sodass jede Person die Möglichkeit hat, einen eigenen Sensorknoten zu bauen und die Daten an Sensor.Community zu übermitteln. Deren Website visualisiert die Daten und bietet damit die Möglichkeit, die Daten mit anderen Orten auf der Erde zu vergleichen. Die Daten stehen unter der Open Database License[2] und können somit ohne größeren Aufwand genutzt werden.
Der Feinstaub-Datensatz wurde für den Zeitraum bis Ende 2020 in vorherigen Arbeiten genauer untersucht [Braatz, 2021a][Braatz, 2021b]. In dieser Arbeit werden zum einen ein größerer Zeitraum als in den vorherigen Arbeiten betrachtet (Januar 2017-Oktober 2021). Zum anderen werden zusätzlich Daten bezüglich Temperatur und Luftfeuchtigkeit hinzugezogen. Die Daten werden täglich in einer CSV-Datei pro Sensor und Station für den vergangenen Tag veröffentlicht. Des weiteren werden je Sensor auch monatliche Dateien als gezippte CSV-Dateien zur Verfügung gestellt (bis Oktober 2020 sind die Daten auch im Parquet[3]-Format erhältlich). In den monatlichen Daten sind die Stationen über eine

[1] https://sensor.community/de/ (26.03.2022)
[2] http://opendatacommons.org/licenses/odbl/1.0/
[3] https://parquet.apache.org/

zusätzliche Spalte unterscheidbar. In den vergangenen Arbeiten wurden hauptsächlich die Daten im Parquet-Format genutzt, da das Format speichereffizient ist und den Datentyp der Spalten erhält, wie beispielsweise die Zeitstempel der Messungen. Für einen einheitlichen Einleseprozess der Daten werden in dieser Arbeit ausschließlich die gezippten CSV-Dateien genutzt. Auf die damit einhergehenden Besonderheiten wurde bereits in [Braatz, 2021a] eingegangen. Die größere Menge an Daten, die in dieser Arbeit genutzt werden, enthält zusätzlich folgende Besonderheiten:

1. Mehrere CSV Dateien beginnen erst in der zweiten Zeile mit den Spaltennamen
2. 2019-12_SDS011.csv hat die Spaltennamen in der ersten Zeile, der Inhalt beginnt erst ab Zeile drei (die zweite Zeile ist ungültig).
3. 2017-03_bme280.csv hat die Spaltennamen in der zweiten Zeile, allerdings beginnt der Datensatz mit weniger Spalten (ist somit ungültig im CSV Format). Ab der Zeile 3125 haben die Daten dir richtige Anzahl an Spalten.
4. 2019-12_bme280.zip enthält zwei CSV-Dateien
5. 2020-09_SDS011.csv enthält Einträge mit ungewöhnlichem Zeitstempelformat (YYYY-MM-DDTHH:MM:SS ist das übliche Format; 2020-090007T20:55:34.00 ist enthalten). Die Messungen der betroffenen Einträge sind nicht vorhanden.

Für einige Zeiträume fehlen Daten je Sensortyp. Die Ausfallzeiträume sind in Abb. 3.1 dargestellt. Längere Ausfallzeiten sind extra hervorgehoben. Dabei fallen der April 2021 für der DHT22 und der Juni 2021 für den BME280 besonders auf. In beiden Fälle liegen für den Monat keine monatlichen Daten vor. Fehlende Temperatur- und Luftfeuchtigkeitsdaten werden Sensorknotenweise durch Daten des *Deutschen Wetterdienstes* ergänzt. Durch diese Möglichkeit werden die fehlenden Monate an Daten nicht durch die täglichen Daten aufgefüllt. Insgesamt fehlen etwa 0,96 % der Daten des SDS011, 2,49 % des DHT22 und 2,61 % des BME280.

3.1.1 Feinstaubdaten

Es existieren die Sensoren SDS011 und PPD42NS für die Feinstaubmessung. Der PPD42NS war der erste Feinstaubsensor des Projekts (seit 2015-10) und der SDS011 wurde später (2016-07) hinzugefügt. Der SDS011 hat sich durch seine feinere Auflösung etabliert und wird seither fast ausschließlich genutzt. Der SDS011 misst Feinstaub in $\mu g/m^3$ für

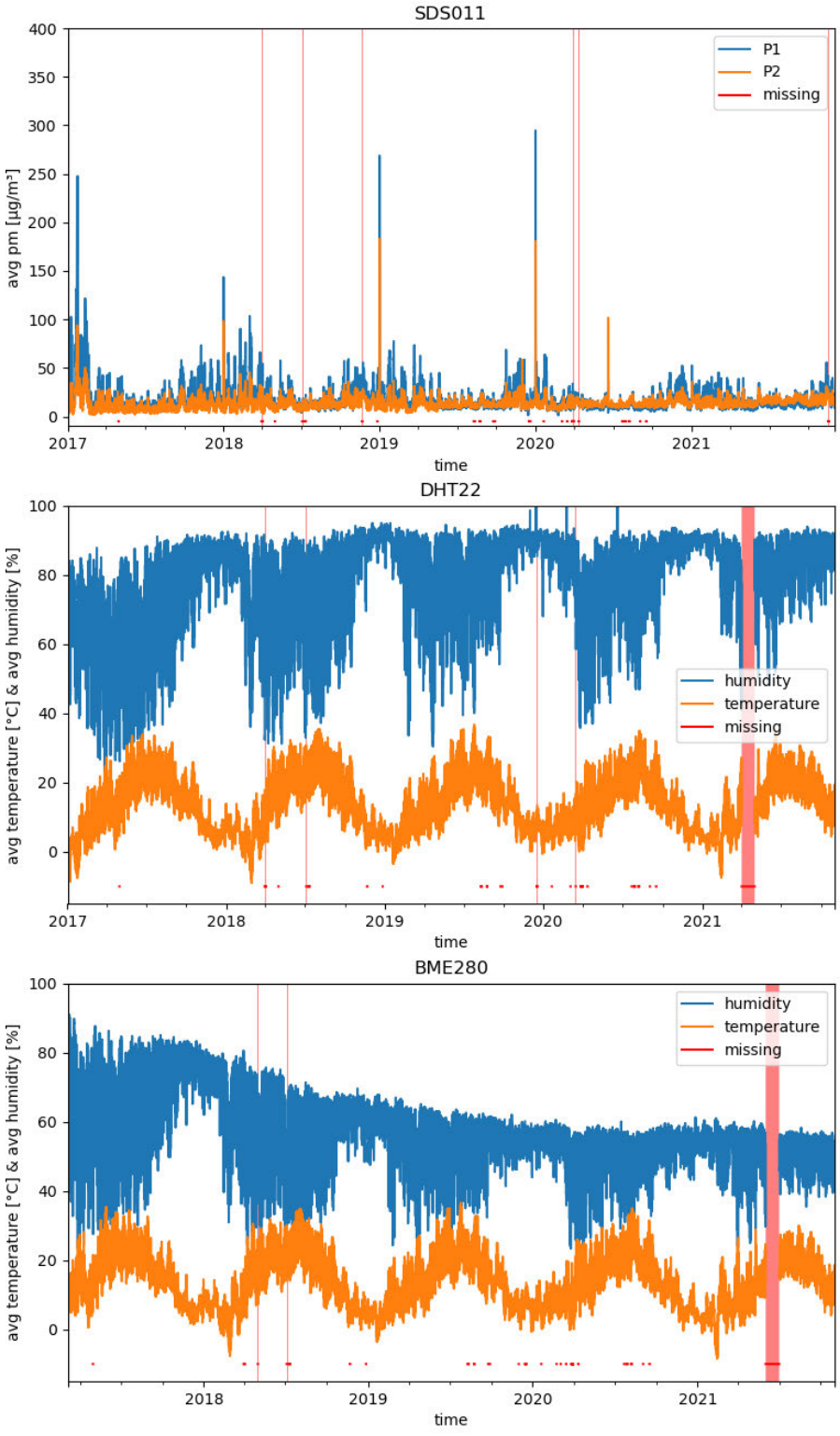

Abbildung 3.1: Fehlende Werte je Sensor, größere Ausfälle sind besonders hervorgehoben: a) SDS011; b) DHT22; c) BME280

P_{10} (<10 μm) und $P_{2.5}$ (<2,5 μm). Der durch das Projekt definierte Wertebereich liegt bei 0-1999 $\mu g/m^3$. Allerdings kommt es bei den monatlichen Dateien dazu, dass Werte teilweise deutlich außerhalb des Wertebereichs liegen oder an Stelle der Messwerte Textinhalte vorhanden sind (siehe [Braatz, 2021a]).
Daher wurden die Feinstaubdaten bereits von Ausreißern befreit, wie in [Braatz, 2021b] beschrieben ist. Dabei wurden die Ausreißer sowohl aus der zeitlichen, als auch räumlichen Perspektive entfernt. Ein Aureißer wird als solcher über den *robusten Z-Score* definiert. Dieser betrachtet die Differenz von Messwert und dem Median der Vergleichsmenge im Verhältnis zur mittleren Abweichung vom Median der Vergleichsmenge. Die positiven Effekte der Ausreißerbehandlung sind ebenfalls in [Braatz, 2021b] dargestellt.

3.1.2 Temperatur- & Luftfeuchtigeitdaten

Die am häufigsten genutzten Sensoren für Temperatur und Luftfeuchtigkeit sind der DHT22 (seit 2015-10) und BME280 (seit 2017-03). Darüber hinaus existieren noch weitere Sensoren für Temperatur und Luftfeuchtigkeit in dem Projekt, jedoch sind diese erst später hinzugekommen und deutlich weniger vertreten. Aus diesem Grund werden diese Sensoren hier nicht weiter aufgeführt. Der gültige Wertebereich für die Temperatur wurde auf $-20\,°C$ - $50\,°C$ und für die Luftfeuchtigkeit auf 0 % - 100 % festgelegt.

3.2 Umweltbundesamt

Die kostengünstigen Sensoren zur Feinstaubmessung werden werksseitig nicht kalibriert. Wie in Kapitel 2.4.1 beschrieben, kann dies insbesondere mit der Einwirkung von Umwelteinflüssen zu Messungenauigkeiten führen. Allerdings konnten die Sensoren nachträglich durch staatliche Referenzstationen kalibriert werden. Hierfür wurden Daten des Umweltbundesamts (UBA) genutzt. Das UBA ist die zentrale Umweltbehörde Deutschlands und befasst sich mit dem gesunden Miteinander von Mensch und Umwelt. Dazu gehört auch die Überwachung der Luftqualität. Hierfür werden an über 400 Luftmessstationen diverse Luftmessungen vorgenommen [Umweltbundesamt, 06.03.2022].

Feinstaubmessungen gehören ebenfalls zur Bestimmung der Luftqualität. Für die Messung wird der Feinstaub von sogenannten „Staubsammlern" eingesaugt. Am Einlass befindet sich ein Abscheider, welcher die Luft auf bestimmte Partikelgrößen filtert. Der Luftstrom wird dann für einen bestimmten Zeitraum über ein Filterpapier geleitet, an

dem sich der Feinstaub ablagert. Das Filterpapier wird durch einen β-Strahler (radioaktive Quelle) durchstrahlt. Die Abschwächung des β-Strahls wird gemessen und in Kombination mit dem Messinterval wird die Feinstaubkonzentration bestimmt. Die genaue Funktionsweise kann [Umweltbundesamt, Juni 2004] entnommen werden.

Die Luftdaten des UBA werden zum Teil in einer App [4] und über eine API [5] zur Verfügung gestellt. Da die Daten für $P_{2.5}$ über die API zum Zeitpunkt der Arbeit nicht mit einer stündlichen Auflösung verfügbar waren, wurden die Daten auf persönliche Anfrage erhalten.

Der erhaltenen Datensatz besteht zum einen aus den Feinstaubmessungen (für P_{10} und $P_{2.5}$) für den Zeitraum 01.01.2017-01.11.2021 und zum anderen aus Informationen zu den Sensorstationen.

3.2.1 Feinstaubdaten

Die Feinstaubdaten liegen in jährlichen CSV-Dateien je Messwert vor. Die Tabellen haben jeweils den gleichen Aufbau mit 31 Spalten:

1. *Station*: Die Stations-ID der Messstationsstandorte
2. *Komponente*: Der gemessene Messwert: [PM10, PM2_5]
3. *Datum*: Tag der Messung
4. *TYPE_OF_AREA*: Lage der Messstation: [vorstädtisches Gebiet, städtisches Gebiet, ländlich regional, ländlich stadtnah, ländliches Gebiet, ländlich abgelegen]
5. *TYPE_OF_STATION*: Art der zu messenden Domäne: [Hintergrund, Industrie, Verkehr]
6. *Wert01-Wert24*: Je eine Spalte pro Stunde des Tages mit den Messwert-Mittelwerten
7. *TYPE_OF_DATA*: Aggregationsgrad der Daten, hier immer Stundenmittelwerte: [S]
8. *Lieferung*: Lieferstatus aus welchem Update die Daten stammen: [Y=Jahresupdate, M=Monatsupdate, D=Aktualdaten]

[4] https://www.umweltbundesamt.de/themen/luft/luftqualitaet/app-luftqualitaet
[5] https://www.umweltbundesamt.de/daten/luft/luftdaten/doc

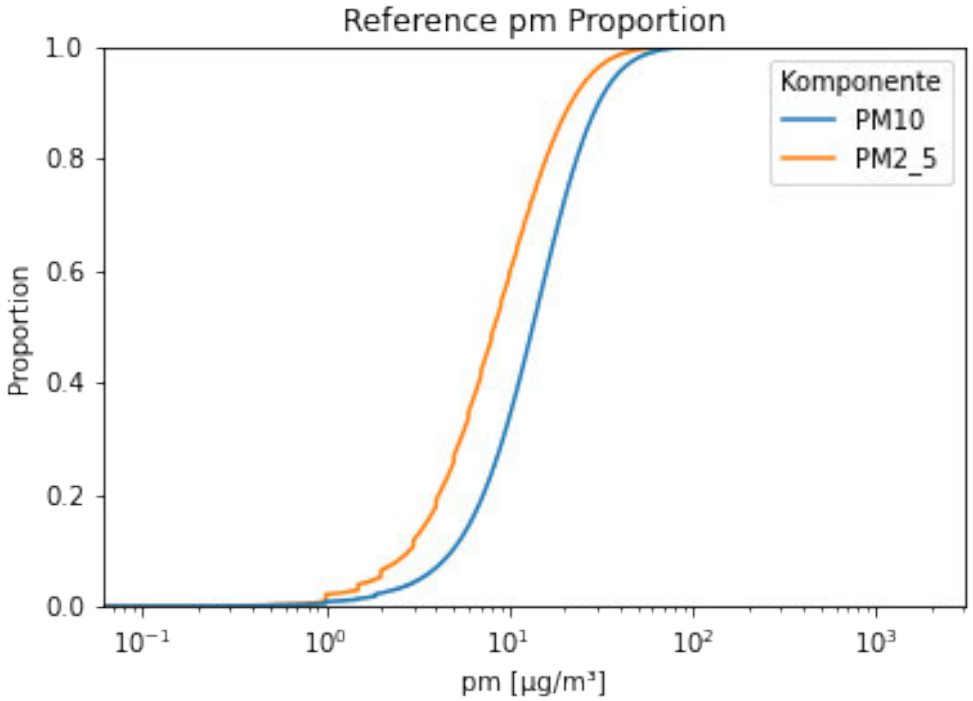

Abbildung 3.2: Verteilung der Messwerte für P_{10} und $P_{2.5}$ in den Daten des UBA für Werte größer 0,01 $\mu g/m^3$

Die Daten enthalten Messungen von 405 einzelnen Stations-IDs. Dabei sind etwa 64 % der Daten P_{10}-Messungen und 36 % für $P_{2.5}$. Die Messwerte werden mit dem Wert '-999' als ungültig gekennzeichnet. Dies trifft auf 2,3 % des Datensatzes zu (1,96 % der P_{10}-Daten und 2,89 % der $P_{2.5}$-Daten). Außerdem ist ein Teil (0,22 %) der Messungen negativ, welches dem Messverfahren geschuldet ist (0,28 % der P_{10}-Daten und 0,19 % der $P_{2.5}$-Daten). Die ungültigen und negativen Werte wurden entfernt und durch NaN ersetzt. In einem weiteren Schritt wurden diese Werte dann je Station mithilfe einer zentrierten Fensterfunktion der Größe 12 mit dem Mittelwert ersetzt. Dadurch wurden Datenlücken bis zu einer Dauer von 12 Stunden geschlossen. Daten, die über einen längeren Zeitraum undefiniert sind, wurden entfernt, da sie nicht sicher mit anderen Feinstaubdaten vergleichbar sind.
Die Verteilung der Messwerte für Werte größer 0,01 $\mu g/m^3$ ist in Abb. 3.2 dargestellt. Die Messung für $P_{2.5}$ ist grundsätzlich niedriger als die von P_{10}. Außerdem ist zu erkennen, dass der Großteil der Messungen im ein- bis zweistelligen Bereich liegen.

3.2.2 Stationsinformationen

Die Stationsinformationen enthalten Metainformationen zu allen Messstationen des UBA. Die Daten liegen ebenfalls im CSV-Format vor. Zu beachten ist, dass die erste Zeile des Datensatzes den Hinweis „Stationen vom: 03.11.2021" enthält, welcher bei dem Einlesen der Daten übersprungen werden muss. Jede Zeile in dem Datensatz bezieht sich auf eine Station, welche durch 44 Spalten beschrieben wird. Davon beschreiben der Großteil der Spalten die Position der Station. Zum einen wird der Ort durch die Adresse und die Koordinaten angegeben, zum anderen im Kontext der Umgebung. Dabei gibt es Angaben zu umliegenden Emissionsquellen, Art der Umgebung und Entfernung zur nächsten Straße. Von den 44 Spalten werden folgende fünf genutzt:

1. *Stationscode*: Die Stations-ID der Messstationsstandorte
2. *Aktivitätsperiode: von* : Datum als Text, ab wann eine Station in Betrieb genommen (Zu beachten: Leerzeichen am Ende des Spaltennames) wurde
3. *Aktivitätsperiode: bis*: Datum als Text, bis wann eine Station in Betrieb war (fehlt, wenn die Station noch in Betrieb ist)
4. *Länge dezimal*: Längengrad der Station als float64
5. *Breite dezimal*: Breitengrad der Station als float64

Die Spalten bezüglich des Betriebszeitraums werden ausschließlich in der Datenanalyse genutzt, um die Entwicklung der Stationenanzahl über die Zeit zu betrachten, welche in Abb. 3.3 zu sehen ist. Die Zuordnung einer Station zu PM-, P_{10}- oder $P_{2.5}$-Messstationen wurde anhand der verfügbaren Feinstaubdaten des UBA vorgenommen (siehe 3.2.1). Grundsätzlich kann eine Messstation meherer Luftmessinstrumente beinhalten. Ab welchem Zeitpunkt eine Station mit einem Staubsammler ausgestattet wurde, konnte nicht genau bestimmt werden. Abb. 3.3 zeigt, dass etwa 40 % der Luftmessstationen Feinstaub messen. Von diesen wiederum messen die meisten P_{10}. $P_{2.5}$ wird von etwa zwei Dritteln der PM-Messstationen gemessen. Dementsprechend werden an den meisten Stationen, an denen P_{10} gemessen wird, auch $P_{2.5}$ gemessen. Zum Ende des betrachteten Zeitraums haben lediglich 24 Stationen ausschließlich $P_{2.5}$ gemessen. Die Unterschiede in der Anzahl an P_{10}- und $P_{2.5}$-Messstationen ist vermutlich darauf zurückzuführen, dass das Messnetz für P_{10} bereits seit den 1990er Jahren aufgebaut wird. Das Messnetz für $P_{2.5}$ besteht

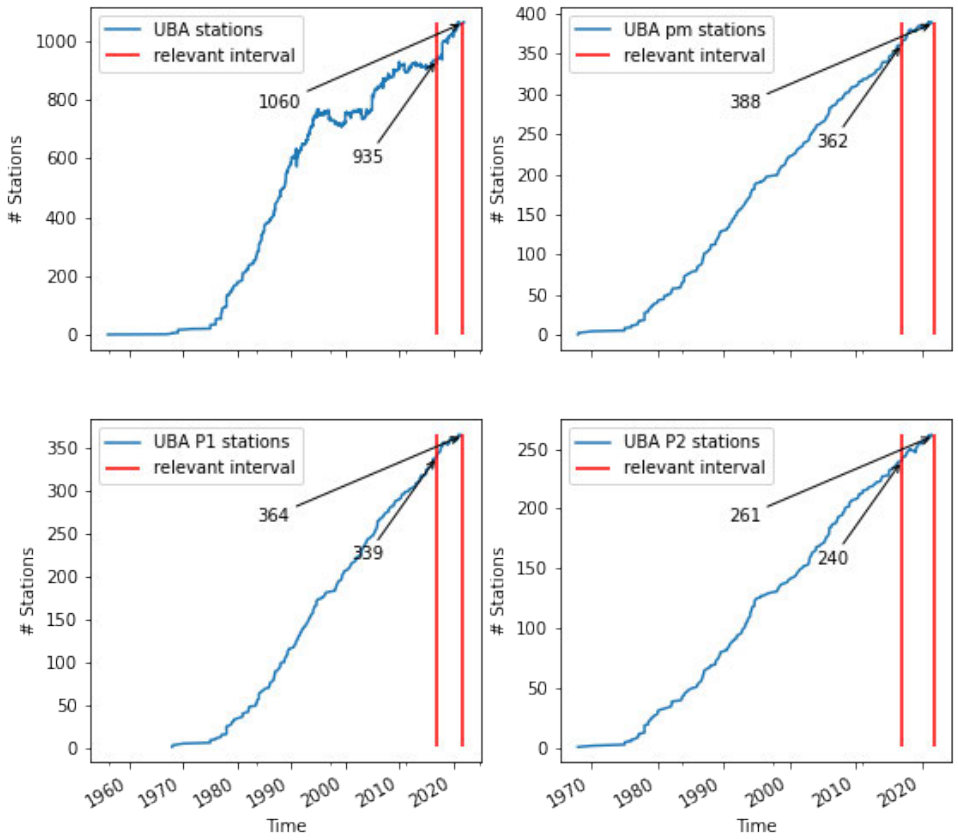

Abbildung 3.3: Entwicklung der Anzahl an Messstationen: a) aller UBA Luftmessstationen, b) aller PM-Messstationen, c) der P_{10}-Messstationen, d) der $P_{2.5}$-Messstationen.

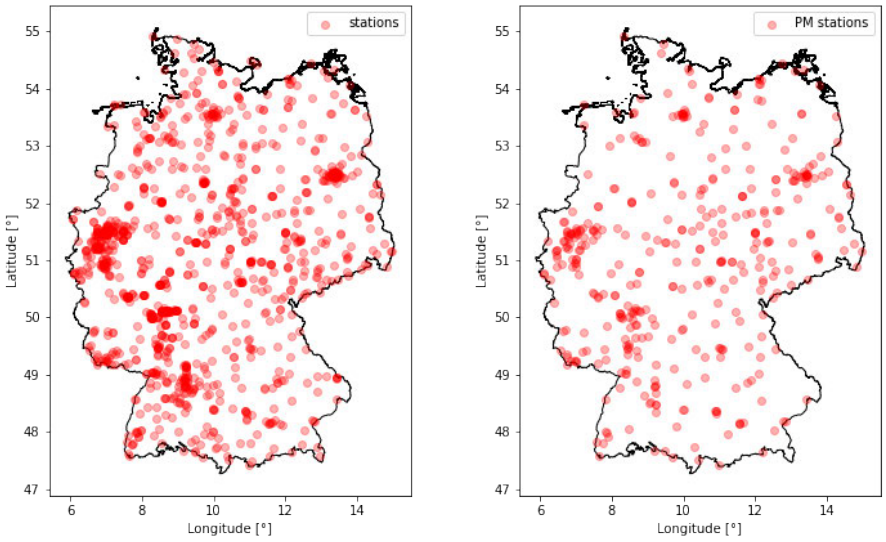

Abbildung 3.4: Standorte der Messstationen: a) aller UBA Luftmessstationen, b) aller PM-Messstationen.

erst seit 2008 [Umweltbundesamt, 13.03.2022]. In Abb. 3.4 ist die Verteilung aller Luftmessstationen des UBA und speziell der Stationen, die Feinstaub messen, abgebildet. Grundsätzlich ist eine gleichmäßige Verteilung der Stationen, mit Ballungen um Großstädte zu erkennen. Die Verteilung der Feinstaubmessstationen ist weniger dicht, hat dennoch größtenteils eine gleichmäßige Abdeckung. Lediglich im Norden und Südosten Deutschlands ist die Dichte der Stationen tendenziell geringer.

3.3 Deutscher Wetterdienst

Der Deutsche Wetterdienst (DWD) ist eine Bundesbehörde Deutschlands und „[...] ist für die Erfüllung der meteorologischen Erfordernisse aller Wirtschafts- und Gesellschaftsbereiche in Deutschland zuständig" [DWD, 27.03.2022]. Dazu gehört unter anderem die Erfassung von metereologischen und klimatologischen Daten. Die Wetterdaten werden seit dem 25.07.2017 größtenteils kostenfrei über den *Open Data Server*[6] zur Verfügung gestellt.

[6] https://opendata.dwd.de/

Für den einfachen Zugriff auf die Daten wurde von [Benjamin Gutzmann et al, 2021] das Python Modul *wetterdienst* entwickelt. Es bietet eine API, um auf Wetterdaten von mehreren Wetterdienstleistern in Python zugreifen zu können. Zusätzlich bietet es Funktionalitäten, um die nächstgelegenen Stationen zu einem Koordninatenpaar zu bestimmen. Die Daten der Stationen können, unter Berücksichtigung von Filtern, abgerufen werden.

Für diese Arbeit wurden Temperatur- und Luftfeuchtigkeitsdaten abgerufen, um fehlende Messdaten an den Sensorstationen der Sensor.Community auszugleichen. Hierfür wurde, sofern Daten fehlen, die nächste verfügbare Station über *wetterdienst* bestimmt und die Daten aus dem Datensatz *DwdObservationDataset.MOISTURE* in einer stündlichen Auflösung abgerufen. Diese enthalten neben der Stations-ID und einem Zeitstempel sechs Messwerte im Kontext von Feuchtigkeit. Die Messwerte beinhalten unterschiedliche Messungen zu Luftfeuchtigkeit, Luftdruck und Temperatur. Für diese Arbeit wurden nur die Messwerte für *tt_std* (Temperatur 2 m über dem Boden) und *rf_std* (relative Luftfeuchtigkeit) genutzt.
Bei den Daten ist zu beachten, dass auch wenn eine nächstgelegene Station bestimmt wird, es möglich ist, dass keine Daten verfügbar sind. In diesem Fall wurden die nächsten Stationen abgefragt, bis von einer Station Daten erhalten wurden. Seltener werden die Daten doppelt zurückgegeben. In diesem Fall wurden die doppelten Daten entfernt.

4 Umsetzung

In diesem Kapitel werden die durchgeführten Experimente beschrieben. Am Anfang wird der Zusammenhang zwischen den Versuchen dargestellt, sowie der daraus resultierende Ablauf. Im Anschluss wird kurz die genutzte Toolchain beschrieben. Daraufhin werden die maßgeblichen Metriken vorgestellt, welche für die Evaluation der Experimente genutzt werden. Der Versuch wird in drei Schritte aufgeteilt: Kalibrierung der Sensorstationen, Interpolation eines Rasters und Prognose der Feinstaubentwicklung. Für jeden Schritt werden die benötigten Materialien und Methoden, sowie die Versuchsdurchführung und die Ergebnisse vorgestellt.

4.1 Genereller Versuchsaufbau

Der Versuchsaufbau ist an den Ablauf des KDD-Prozesse angelehnt, wie er in Abschnitt 2.2.1 beschrieben wurde. Der Versuchsablauf ist in Abb. 4.1 dargestellt. Diese Arbeit ist Teil einer Reihe von Ausarbeitungen, weshalb einige Versuchsschritte bereits in vorherigen Arbeiten beschrieben wurden. Das Einlesen der Daten und die Vorverarbeitungsschritte bis einschließlich der Entfernung räumlichen Ausreißer wurde bereits in [Braatz, 2021a] und [Braatz, 2021b] beschrieben. Die nötigen Anpassungen an den größeren Datensatz und neue Datenquellen wurden bereits in Kapitel 3 vorgestellt. In dieser Arbeit wird ergänzend die Sensorkalibrierung als zusätzlicher Vorverarbeitungsschritt betrachtet. Die Kalibrierung wird in Abschnitt 4.4 genauer untersucht. Der Schritt der Datentransformation wird um eine Anpassung der Datenrepräsentation erweitert, wie sie in Abschnitt 2.2.2 beschrieben wurde. Dabei sollen die ungleichmäßig verteilten Daten zu einem gleichmäßigen Gitter geschätzt werden. Dieser Schritt wird in Abschnitt 4.5 behandelt, wobei drei unterschiedliche Ansätze verglichen werden. In dem Data-Mining-Schritt des KDD-Prozesses wird in diesem Versuch eine Prognose der Feinstaubdaten

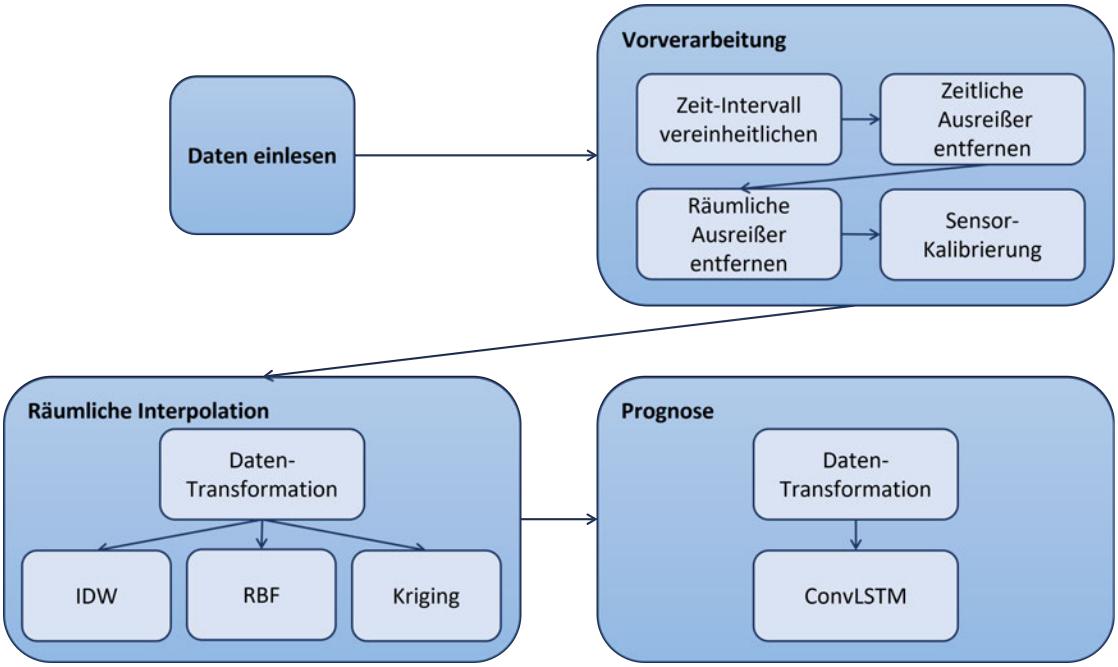

Abbildung 4.1: Ablauf des Versuchs

durchgeführt. Dieser Schritt wird in Abschnitt 4.6 betrachtet und vergleicht unterschiedliche Architekturen eines ConvLSTM-Netzes. Die Diskussion der Ergebnisse der einzelnen Schritte des Versuches wird in Kapitel 5 durchgeführt.

4.2 Toolchain

Ein Großteil der Toolchain konnte aus den vorherigen Arbeiten weiter genutzt werden und wurde genauer in [Braatz, 2021a] und [Braatz, 2021b] beschrieben. Für die Verwaltung und Replizierbarkeit der Python-Umgebung wurde Anaconda[1] genutzt. Anaconda ist eine Alternative zu dem Repository Manager PyPi[2] und ermöglicht das Einbinden von Modulen von Dritten in die eigene Python-Umgebung. Im Gegensatz zu PyPi werden bei Anaconda auch Python-externe Abhängigkeiten mit zur Verfügung gestellt. Das Modul Geopandas benötigt Zugriff auf die Software GEOS (Geometry Engine - Open Source)[3]. GEOS bietet Funktionen zur räumlichen Geometrieberechnung. Allerdings hat

[1] https://www.anaconda.com/ (03.04.2022)
[2] https://pypi.org/(03.04.2022)
[3] https://libgeos.org/ (03.04.2022

diese Software nur eine Schnittstelle für die Programmiersprachen C und C++. Anaconda bindet auch diese externen Abhängigkeiten in die Python-Umgebung ein, sodass sie nicht separat installiert werden müssen. Zusätzlich ermöglicht Anaconda die Definition einer Python-Umgebung mit spezifischen Versionsangaben zu den jeweiligen Modulen und hilft dabei, Abhängigkeiten zwischen den einzelnen Modulen aufzulösen.
Für die Entwicklung der Software wurde hauptsächlich ein Windows-Rechner mit einem AMD FX-6350 mit sechs logischen Kernen als Prozessor, einer NVIDIA GeForce GTX 970 als GPU und 8 GB Arbeitsspeicher genutzt. Mit dieser Hardware konnte die Software für die ersten vier Monate des Datensatzes validiert werden. Für alle weiterführenden Berechnungen wurden die Ressourcen des Forschungs- und Transferzentrum Smart Systems (FTZ SMSY)[4] genutzt. Für Berechnungen mit dem gesamten Datensatz wurde ein Unix-Server mit einem Prozessor mit 64 Kernen, vier NVIDIA A100 als GPUs und 950 GB Arbeitsspeicher genutzt. Für die nachträgliche Auswertung wurde zusätzlich ein Unix-Server mit einem Prozessor mit 36 Threads und 370 GB Arbeitsspeicher verwendet.
Für die Nutzung der Server des FTZ muss die Software in Docker-Containern ausgeführt werden. Docker [Merkel, 2014] bietet die Möglichkeit, Software in sogenannten Containern laufen zu lassen. Diese Container werden durch Software definiert und sind unabhängig von dem Betriebssystem, auf dem Docker ausgeführt wird. Dies ermöglichte die Entwicklung auf einem Windowsrechner und die Ausführung auf einem Unix-basierten Server in dem Rechenzentrum des FTZ. Die Basis des Docker-Images[5] ist ein NVIDIA-Image[6], welches die Nutzung der GPUs in dem Container ermöglicht. Zusätzlich wurde Anaconda in dem Image für die Erstellung und Bereitstellung der Python-Umgebung installiert.

4.3 Metriken

In dieser Arbeit werden unterschiedliche Modelle und Methoden für die Analyse und Verarbeitung von Daten vorgestellt. Im Falle der Kalibrierung ist das Ziel, die Messungen der günstigen Sensoren nachträglich mit Hilfe von Messungen der Sensorstationen des UBA zu korrigieren. Bei der Interpolation werden Werte an umliegenden Orten auf Basis

[4] https://smsy.haw-hamburg.de/ (03.04.2022)
[5] Ein Image ist die Vorlage für einen Container, welche alle installierten Anwendungen enthält [IONOS Digitalguide, 03.04.2022]
[6] https://hub.docker.com/layers/cuda/nvidia/cuda/11.4.2-base-ubuntu20.04/images/sha256-7f53f187704999b6f604aeae40fcd86b95c5f225ab3fcae968a986477c42c464?context=explore (03.04.2022)

vorhandener Messungen geschätzt. Für die Erstellung von Prognosen wird die zukünftige Feinstaubentwicklung auf der Grundlage von vergangenen Messungen vorhergesagt. Die Effektivität der verschiedenen Methoden wird durch Metriken bestimmt und kann so verglichen werden.
Alle genutzten Methoden und Modelle gehören zur Kategorie des *Supervised Learning*. Dementsprechend sind für jeden Datenpunkt immer das erwartete Label (Validierungsdaten) vorhanden. In dieser Arbeit werden *R-Square*, *Root Mean Squared Error* und *Mean Absolut Error* als Metriken genutzt.

4.3.1 R-Square

R-Square (R2) wird auch als Bestimmtheitskoeffizient bezeichnet und beschreibt, wie gut ein Modell die Variabilität einer abhängigen Variablen beschreibt. Im Kontext von Regression zeigt R2 wie gut eine Regressionslinie die Daten repräsentiert [R2., 09.04.2022]. R2 ist wie folgt definiert:

$$R2 = 1 - \frac{\text{sum squared regression (SSR)}}{\text{total sum squares (SST)}} = 1 - \frac{\sum_{j=1}^{l} [z(s_j) - \hat{z}(s_j)]}{\sum_{j=1}^{l} [z(s_j) - \bar{z}(s_j)]} \tag{4.1}$$

Dabei sind $z(s_j)$ die Validierungsdaten, $\hat{z}(s_j)$ sind die geschätzten Werte und $\bar{z}(s_j)$ der Mittelwert über die Validierungsdaten.

4.3.2 Root Mean Squared Error

Root Mean Squared Error (RMSE) gibt ein Fehlermaß an, wie stark die Schätzungen von den tatsächlichen Werten abweichen. RMSE gewichtet dabei größere Abweichungen stärker als *Mean Absolut Error* und im Gegensatz zu *Mean Squared Error* ist das Fehlermaß innerhalb des Wertebereichs der betrachteten Variable. Innerhalb von Klima-, Wetter- und Luftstudien wird RMSE als Standard Metrik genutzt [Chai and Draxler, 2014]. RMSE ist definiert als:

$$RMSE = \sqrt{\frac{1}{l} \cdot \sum_{j=1}^{l} [\hat{z}(s_j) - z(s_j)]^2}; \qquad E\{RMSE\} = \sigma(h = 0) \tag{4.2}$$

Dabei ist l die Anzahl der Validierungsdatenpunkte.

4.3.3 Mean Absolut Error

Mean Absolut Error (MAE) ist ein vergleichbares Fehlermaß wie RMSE. Allerdings wird bei MAE jede Abweichung zwischen den Schätzwerten und Validierungsdaten gleich bewertet unabhängig von der Größe der Abweichung. MAE ist definiert als:

$$MAE = \frac{1}{l} \cdot \sum_{j=1}^{l} |\hat{z}(s_j) - z(s_j)| \tag{4.3}$$

4.4 Kalibrierung

In dieser Arbeit werden Daten aus einem Crowd-Sensing Projekt verwendet. Die Messung von Feinstaubdaten in Crowd-Sensing-Projekten wurde erst durch die Verfügbarkeit von kostengünstigen Sensoren ermöglicht. In Abschnitt 2.4.1 wurde bereits beschrieben, dass diese günstigen Sensoren meist nicht kalibriert sind. In diesem Abschnitt werden unterschiedliche Ansätze der nachträglichen Kalibrierung betrachtet und verglichen. Das Ergebnis dieses Abschnitts ist der Grundstein für die Untersuchungen in den nachfolgenden Unterkapiteln.

4.4.1 Material und Methoden

Datensatz

Für die Kalibrierung der Sensoren werden alle Daten genutzt, die in Kapitel 3 beschrieben wurden. Die Daten werden für den Zeitraum von Januar 2017 bis einschließlich Oktober 2021 betrachtet. Die Feinstaub-, Temperatur- und Luftfeuchtigkeitsdaten von Sensor.Community wurden auf Basis der Geokoordinaten (Längen und Breitengrad) sowie des Zeitstempels zusammengeführt. Durch die eins-zu-eins Beziehungen der Daten an einer Sensorstation zu einem Zeitpunkt gehören die Messungen jeweils zu derselben Sensorstation. Für die Sensorstationen, an denen kein Sensor für Temperatur oder Luftfeuchtigkeit angebracht ist, wurden die fehlenden Daten durch Messungen der nächstgelegenen DWD-Messstation ergänzt. Ein Ausschnitt der Daten ist in der Tabelle 4.1

auf Seite 33 dargestellt. Für die Kalibrierung der Sensoren wurden die Messungen der Feinstaubmessstationen des UBA als Referenzdaten genutzt.

Datenvorverarbeitung

Im Zuge der Datenvorverarbeitung werden dem Sensordatensatz korrespondierende Referenzdaten zugeordnet. Hierfür wird zunächst sichergestellt, das eine gegebene Referenzstation[7] auch zu dem betrachteten Zeitpunkt Feinstaubmessungen zur Verfügung stellt. Dieser Schritt ist notwendig, da in der Stationsübersicht des UBA auch Stationen gelistet werden, die nicht über den gesamten Zeitraum Feinstaub gemessen haben. Die verbleibenden Referenzstationen werden nun den nächstgelegenen Sensorstationen zugeordnet. Für die Zuordnung wird die Implementierung von [Henrikki Tenkanen, Vuokko Heikinheimo, Håvard Wallin Aagesen, 23.02.2022] genutzt. Hierbei wird ein Nachbarschafts-Baum auf Basis der Geokoordinaten der Referenzstationen generiert. Die Entfernung wird über die Haversine-Distanz[8] bestimmt, welche es ermöglicht Entfernungen auf einer Kugel zu berechnen. Über den Nachbarschaftsbaum wird, daraufhin für jede Sensorstation die nächstgelegene Referenzstation bestimmt. Anhand dieser Zuordnung werden für die Messwerte P_{10} und $P_{2.5}$ die korrespondierenden Referenzwerte zugewiesen. Zusätzlich wird die Entfernung zu der gegebene Referenzstation berechnet. Die Nachbarschaft und jeweiligen Distanzen zwischen Sensorstation und Referenzstation sind beispielhaft in der Abb. 4.2 für den letzten Zeitpunkt des Betrachtungszeitraums dargestellt. Zu dem betrachteten Zeitpunkt gibt es die größte Anzahl sowohl an Sensorstationen, als auch an Referenzstationen. Dennoch ist die Verteilung der Referenzstationen nicht engmaschig genug, sodass nicht jede Sensorstation über eine Referenzstation im näheren Umkreis verfügt. Insbesondere in ländlicheren Regionen sind die Entfernungen zwischen den beiden Stationstypen größer. Die Sensorstation auf Sylt ist etwa 73 km von der nächsten $P_{2.5}$-Referenzstation entfernt.

[7] In dieser Arbeit werden Stationen des UBA als Referenzstationen bezeichnet und Sensoren von Sensor.Community als Sensorstationen

[8] https://www.igismap.com/haversine-formula-calculate-geographic-distance-earth/ (18.04.2022)

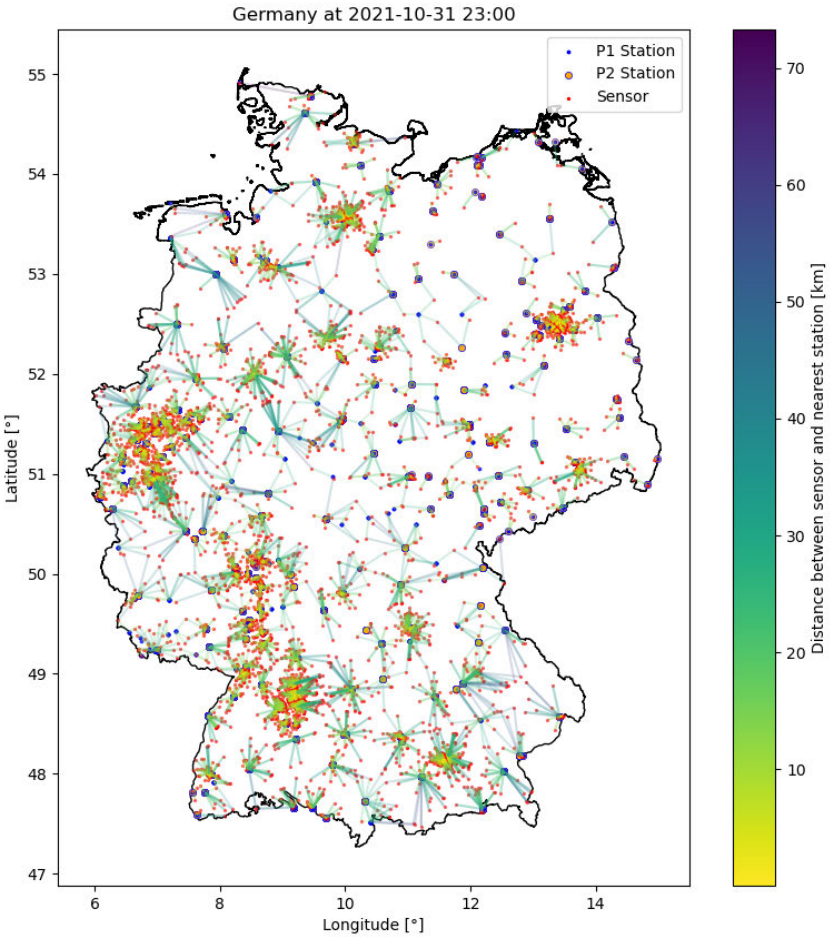

Abbildung 4.2: Darstellung der verfügbaren Sensorstation der Sensor.Community (rot) und Referenzstation des Umweltbundesamts (blau: P_{10} (P1), blau-umrandet: $P_{2.5}$ (P2)) am 31.10.2021 23:00. Die nächsten Referenzstationen für P_{10} und $P_{2.5}$ zu einer Sensorstation sind durch Linien gekennzeichnet. Die Farbe der Linien repräsentiert die Entfernung. Die Entfernung zwischen den Stationen kann über 70 km betragen.

Tabelle 4.1: Ausschnitt der Daten, welche für die Kalibrierung genutzt wurden. Feinstaub-, Temperatur- und Luftfeuchtigkeitsdaten wurden anhand von den Geokoordinaten und dem Zeitstempel zusammengeführt. Fehlende Temperatur- und Luftfeuchtigkeitsdaten wurden von der nächstgelegenen DWD-Messstation ergänzt

lat	lon	timestamp	location	P1	P2	temperature	humidity
48,721	9,209	2017-01-07 00:00:00+00:00	19	90,5337151	55,108746	-13,1	91,4
48,721	9,209	2017-01-07 01:00:00+00:00	19	90,2610418	52,6678855	-13	91,4
48,721	9,209	2017-01-07 02:00:00+00:00	19	90,7982814	55,5778538	-13,6	92,9
48,721	9,209	2017-01-07 03:00:00+00:00	19	86,0685506	52,2171759	-13,8	92,9
48,721	9,209	2017-01-07 04:00:00+00:00	19	79,2416752	48,8143857	-13,8	91,4
48,721	9,209	2017-01-07 05:00:00+00:00	19	90,4947051	55,9793989	-12,8	90,7
48,721	9,209	2017-01-07 06:00:00+00:00	19	69,7413781	45,1648315	-12,5	90,7
48,721	9,209	2017-01-07 07:00:00+00:00	19	68,2403301	44,4243243	-12	90
48,721	9,209	2017-01-07 08:00:00+00:00	19	67,8037348	44,5069638	-11,4	87,2
48,721	9,209	2017-01-07 09:00:00+00:00	19	55,7090153	37,3896673	-9,7	86,7
48,721	9,209	2017-01-07 10:00:00+00:00	19	47,8733456	29,2974044	-7,4	76,6

Kalibrierungsmethoden

In diesem Abschnitt werden die Kalibrierungsmethoden vorgestellt, die miteinander verglichen werden. In Abschnitt 2.4.1 wurde bereits von einigen Ansätzen berichtet. In [Malings et al, 2020] konnten empirische Modelle die Low-Cost-Sensorstation in ähnlichem oder sogar besserem Maße als physikalische Modelle kalibrieren. In [Barkjohn et al, 2021] wurden unterschiedliche lineare Modelle verglichen, welche neben den Feinstaubdaten zusätzlich Temperatur, Luftfeuchtigkeit und Taupunkt betrachten. [Chu et al, 2020] hat, zusätzlich zu einem linearen Modell, *Geographically Weighted Regression* zur Kalibrierung genutzt, um auch die räumliche Abhängigkeit in der Kalibrierung zu berücksichtigen.
In dieser Arbeit wird eine Auswahl und Kombination der eben genannten Modelle verglichen. Das Modell, welches am besten auf den Datensatz generalisiert, wird genutzt, um die Sensordaten für die nachfolgenden Untersuchungen zu kalibrieren.

Geographically Weighted Regression

Geographically Weighted Regression (GWR) ist für jeden Ort i = 1, .., n definiert als:

$$y_i = \beta_{i0} + \sum_{k=1}^{p-1} \beta_{ik} x_{ik} + \epsilon_i \tag{4.4}$$

y_i ist dabei die abhängige Variable an dem Ort i, x_{ik} ist der Wert der k-ten Kovariate an dem Ort i, β_{i0} ist der Achsenabschnitt, β_{ik} ist der Regressionskoeffizient für die k-te Kovariate, p ist die Anzahl an Regressionstermen und ϵ_i ist der zufällige Fehler an dem Ort i.

Die Matrixform ist:

$$y_i = \vec{X_i}\vec{\beta_i} + \epsilon_i \tag{4.5}$$

wobei $\vec{\beta_i}$ ein Spaltenvektor mit den Regressionkoeffizienten ist und $\vec{X_i}$ einen Zeilenvektor mit den unabhängigen Variablen an dem Ort i. Der Vektor mit den geschätzten Regressionkoeffizienten an dem Ort i beschreibt:

$$\hat{\vec{\beta}}_i = [X^T W_i X]^{-1} X^T W_i \vec{Y} \tag{4.6}$$

wobei $\vec{Y}$ der Vektor mit den abhängigen Variablen ist; $X = [X_1^T, X_2^T, .., X_n^T]^T$ ist die Matrix mit den unabhängigen Variablen. Dabei enthält die erste Spalte Einsen für den Achsenabschnitt. $W_i = diag[W_{i1}, .., W_{in}$ ist die $n \times n$ Diagonalmatrix mit den berechneten Gewichten für jeden Ort i und $\vec{\hat{\beta}_i} = (\hat{\beta}_{i0}, \hat{\beta}_{i1}, .., \hat{\beta}_{ip-1})^T$ ist der Vektor mit p Regressionskoeffizienten an dem Ort i für $p-1$ unabhängige Variablen und den Achsenabschnitt. [Wheeler and Páez, 2010]

4.4.2 Durchführung und Ergebnisse

Die Kalibrierung wird mit Lineare Regression (LIN) und GWR getestet. Außerdem wurden drei unterschiedliche Featuresets verglichen. Diese werden in Tabelle 4.2 auf Seite 37 zusammen mit den dazugehörigen Modellnamen dargestellt. Zusätzlich wird der Einfluss der Distanz zwischen Sensorstation zur korrespondierenden Referenzstation auf die Modelle untersucht. Dabei werden folgende Entfernungen betrachtet: 0,25 km, 0,5 km, 1,0 km, 2,0 km und 5,0 km. Dies führt zu 30 verschiedenen Szenarien, je für P_{10} und $P_{2.5}$.
Jedes Szenario wird je Zeitstempel auf die Daten angewandt und 5-fach-kreuzvalidiert, sofern ausreichend Datenpunkte vorhanden sind. Andernfalls wird der Zeitschritt nicht bewertet.
Das Modell mit den besten Metriken wird im Anschluss auf den gesamten Datensatz angewendet. Die Kalibrierung wird ebenfalls je Zeitschritt durchgeführt. Für das Training des Kalibrierungsmodells werden alle Daten aus dem festgelegten Einzugbereich genutzt. Das Modell wird im Anschluss auf alle Daten (inklusive der Daten außerhalb des Einzugbereichs) angewandt.
Für die Implementierung von LIN wird die Python-Bibliothek *sklearn* [Fabian Pedregosa et al, 2011] genutzt und GWR wird mit der Bibliothek *pysal* [Rey and Anselin, 2007] implementiert.

Modellvergleich

Die folgenden Grafiken sind mit absoluten anstelle von relativen Luftfeuchtigkeitsdaten des DWD entstanden. Dieser Fehler ist erst nachträglich aufgefallen. Zur Überprüfung der Auswirkung auf die Analyse wurde beispielhaft die Analyse der Metrik Verteilung für einen kleineren Zeitraum überprüft (vergleiche Abb. A.1 und Abb. A.2 im Anhang). Die Ergebnisse der Überprüfung führen effektiv zu den gleichen Resultaten wie mit der

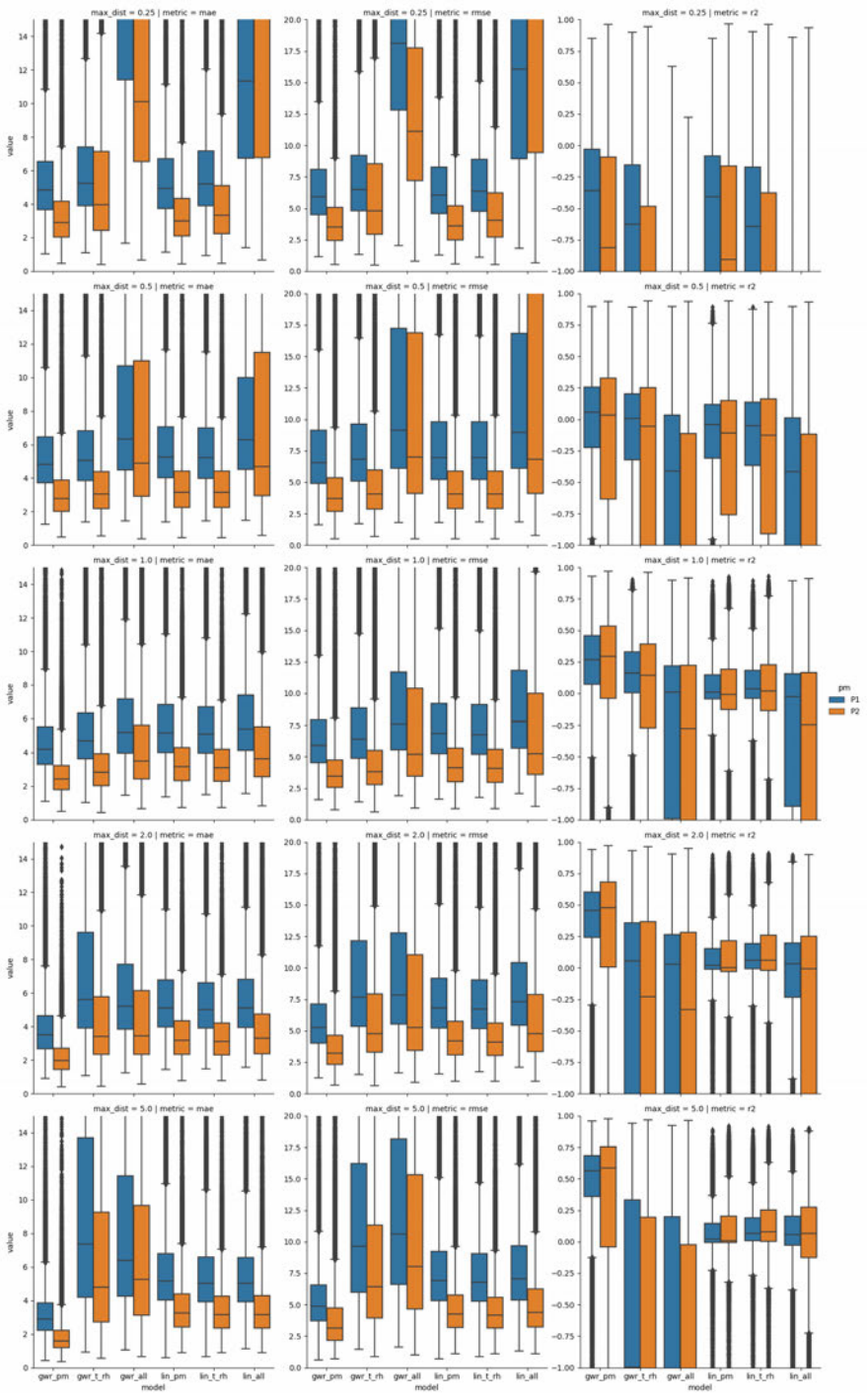

Abbildung 4.3: Darstellung der Verteilung der Metriken je Kalibrierungsmodell und Parameterkombination für P_{10} (P1) und $P_{2.5}$ (P2) über alle Zeitstempel. Je Grafik sind die sechs Kalibrierungsmodelle aufgezeichnet. In den Zeilen sind die betrachteten Entfernungen dargestellt und in den Spalten die Metriken (MAE, RMSE und R2)

Tabelle 4.2: Aufstellung der Modellnamen der Kalibrierungsmodelle und die jeweils genutzten Feuturesets, wobei PM der jeweilige Feinstaubmesswert ist, T ist die Temperatur und H ist die relative Luftfeuchtigkeit.

Modellnamen	Featureset
lin_pm / gwr_pm	PM
lin_t_rh / gwr_t_rh	$PM + T + H$
lin_all / gwr_all	$PM + T + H + PM^2 + T^2 + H^2$ $+ PM * T + PM * H + PM * T * H$

absoluten Luftfeuchtigkeit. Daher werden im Folgenden die Ergebnisse der Analyse mit den ergänzten absoluten Luftfeuchtigkeitsdaten vorgestellt.
Die Ergebnisse des Vergleichs sind in Abb. 4.3 dargestellt. Zu sehen sind die Verteilung der drei Metriken über die Zeitstempel. In den Zeilen sind die Entfernungen zwischen Sensorstation und Referenzstation aufgetragen. Tendenziell ist zu erkennen, dass die Metriken sich bei größeren betrachteten Entfernungen verbessern. Dies ist vermutlich dadurch zu erklären, dass zunehmend mehr Datenpunkte für das Training der Modelle zur Verfügung standen, je größer der betrachtete Bereich gewählt wird. Die Metriken für die Modelle gwr_t_rh und gwr_all verschlechtern sich bei Entfernungen größer 1 km und 2 km. Diese Modelle nutzen GWR, welches zusätzlich die räumliche Nähe berücksichtigt. Eine mögliche Erklärung der Verschlechterung dieser Modelle ist die Tatsache, dass die räumlichen Zusammenhänge für die gegebenen Featuresets nicht bei größeren Entfernungen bestehen, beziehungsweise nicht erlernt werden konnten.
Das Modell mit dem höchsten R2-Score ist gwr_pm bei einer Entfernung von 5 km. Außerdem weisen MAE und RMSE die niedrigsten Werte auf.

Kalibrierung des Datensatzes

Bei allen folgenden Darstellungen ist der Zeitraum ab 01.02.2017 betrachtet. Im Januar kam es zu einem einmaligen, extremen Ausreißer bei der Kalibrierung. Dies ist vermutlich auf die sehr geringe Anzahl an Sensorstationen zurückzuführen. Die Metriken der Kalibrierung des Datensatzes mit dem Modell *gwr_pm* mit einem Einzugbereich von 5 km sind in Tabelle 4.3 dargestellt. Im Vergleich zu den unkalibrierten Daten ist eine klare Verbesserung in den Metriken zu erkennen. Insbesondere bei P_{10} (P1) ist an dem R2-Score erkennbar, dass vor der Kalibrierung kein Zusammenhang zwischen den Messungen der Referenzstationen und denen der Sensorstationen feststellbar war. Nach der

Tabelle 4.3: Kalibrierungsmetriken nach der Kalibrierung des gesamten Datensatzes.

PM	Daten	MAE	RMSE	R2
P1	ursprünglich	8,69	13,87	-0,19
P1	kalibriert	**3,5**	**6,99**	**0,70**
P2	ursprünglich	5,28	7,93	0,16
P2	kalibriert	**2,06**	**3,99**	**0,79**

Kalibrierung ist die Variabilität der Messungen deutlich ähnlicher zueinander. Die Kalibrierung von $P_{2.5}$ (P2) zeigt ein ähnliches Bild, allerdings ist bereits vor der Kalibrierung ein gewisser Teil der Variabilität abgebildet.
In Abb. 4.4 auf Seite 39 ist zu erkennen, dass die Verteilung der kalibrierten Messungen sich denen der Referenzstationen angenähert hat. In Abb. 4.5 auf Seite 40 sind die kalibrierten und ursprünglichen Messungen im Vergleich zu den Messungen der Referenzstationen aufgeführt. Es ist zuerkennen, dass höhere Referenzmessungen, bei denen die Sensorstationen zuvor niedriger gemessen haben, angepasst sind. Des Weiteren ist die Streuung verringert.

4.5 Interpolation

In diesem Abschnitt werden die kalibrierten Daten genutzt, um ein Modell zur Interpolation zwischen den einzelnen Sensorstationen zu finden. Das Modell soll auf Basis der ungleichmäßig verteilten Sensorstationen ein gleichmäßiges Raster schätzen. Diese Datenrepräsentation kann in dem nächsten Abschnitt für die Prognose genutzt werden.
Für die Bestimmung des Modells werden drei unterschiedliche Ansätze verglichen. In vorherigen Arbeiten wurde intensiv *Kriging* vorgestellt [Braatz, 2021a][Braatz, 2021b]. *Inverse Distance Weighting* wurde ebenfalls in [Braatz, 2021a] erläutert. In dieser Arbeit wird zusätzlich *Radial Basis Function* verglichen. Für den Vergleich werden dieselben Metriken genutzt wie in dem vorherigen Abschnitt.

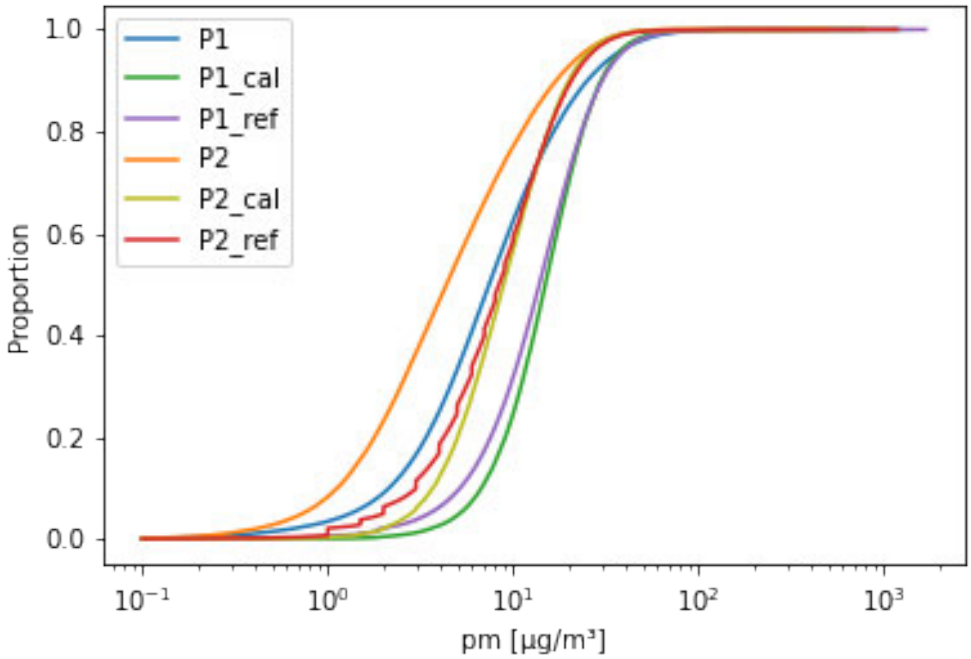

Abbildung 4.4: Verteilung der ursprünglichen Messungen, Werte nach der Kalibrierung (_cal) und die Referenzmessungen (_ref) jeweils für P_{10} (P1) und $P_{2.5}$ (P2)

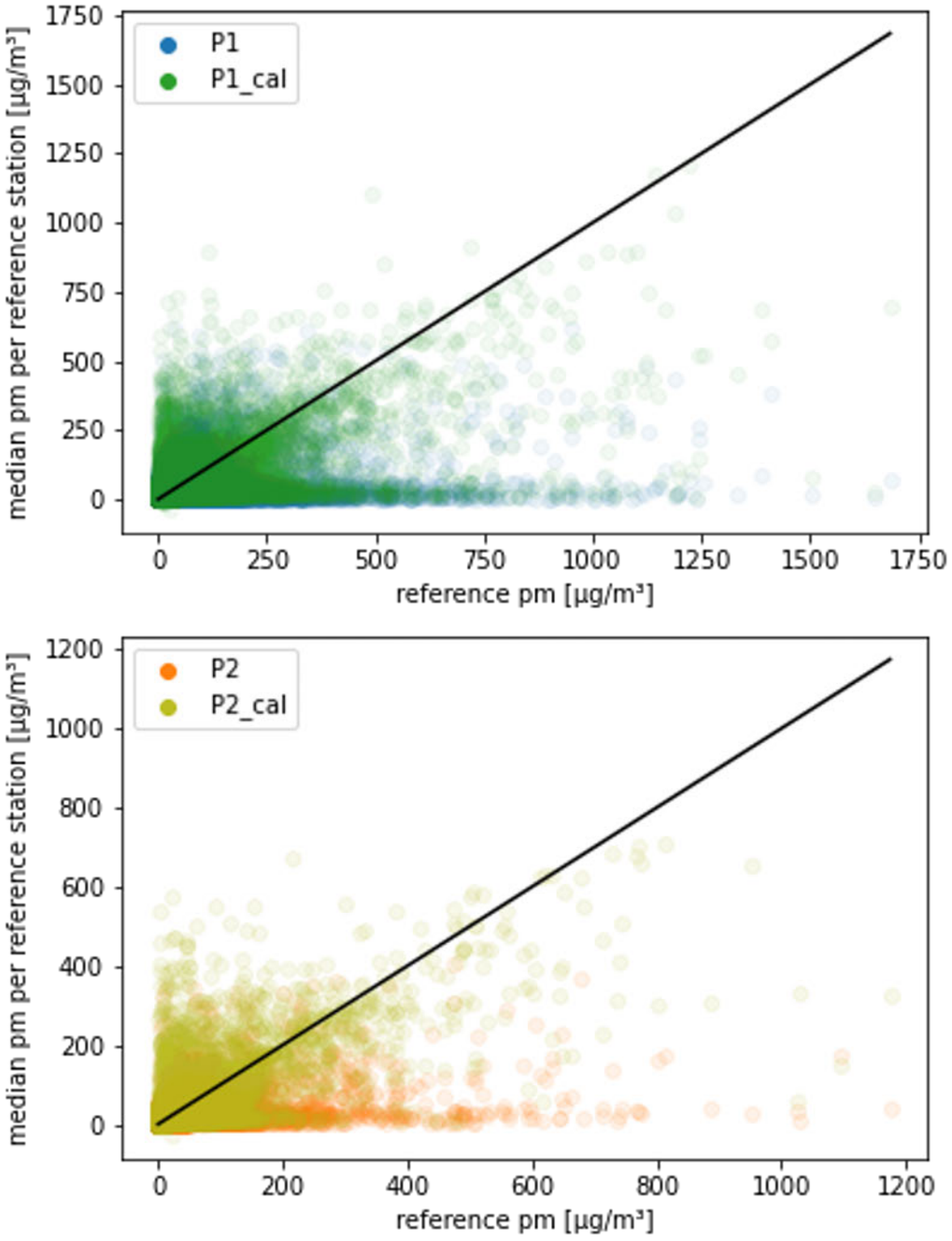

Abbildung 4.5: Vergleich zwischen den kalibrierten (P1_cal, P2_cal) und ursprünglichen Messungen (P1, P2) zu den Messungen der Referenzstationen. a) P_{10}; b) $P_{2.5}$

4.5.1 Material und Methoden

Datensatz

In diesem Abschnitt wird der resultierende Datensatz aus dem Abschnitt 4.4 genutzt. Der Datensatz hat einen ähnlichen Aufbau, wie in der Tabelle 4.1 auf Seite 33 dargestellt. Allerdings sind die Spalten „P1" und „P2" durch die Spalten „P1_cal" und „P2_cal" mit den kalibrierten Feinstaubdaten ersetzt. Des Weiteren werden die Spalten „temperature", „humidity", sowie „location" nicht genutzt. Aufgrund des hohen Rechenaufwands wird in diesem Abschnitt ein kleinerer Zeitraum für den Modellvergleich betrachtet (01.02.2017-01.12.2017).

Inverse Distance Weighting

Inverse Distance Weighting (IDW) wurde von [Shepard, 1968] vorgestellt und ist eines der ältesten räumlichen Interpolationsmodelle [Hengl, 2007]. Die Funktionsweise wurde bereits in [Braatz, 2021a] genauer vorgestellt. IDW gewichtet eine Mittelwert basierend auf den umliegenden Messungen. Die Gewichte ergeben sich aus den Inversen der Distanz zu den umliegenden Messpunkten. Dabei ist die Summe der Gewichte eins. Die Inverse der Distanz führt dazu, dass weiter entfernte Messungen weniger gewichtet werden als nahegelegene Messungen.

Radial Basis Function

Radial Basis Function (RBF) (deutsch: Radiale Basisfunktion) gehören zu den *Kernel-Funktionen. Kernel-Funktionen* gehören zu den *non-parametric models* (deutsch: Nicht-parametrisierte Lernverfahren). Sie benötigen keine fixe Größe des Eingangsvektors und trainieren auf Basis der Ähnlichkeit von Datenpunkten. Diese Ähnlichkeit muss bestimmbar sein durch eine reellwertige Ähnlichkeitsfunktion: Lasse $\kappa(x, x') \in \mathbb{R}$ die Ähnlichkeit zwischen den Datenpunkten $x, x' \in X$, wobei X den abstrakten Datenraum beschreibt und κ ist die *Kernel-Funktion.*
Ein RBF-Kernel ist dabei eine besondere Form des *squared exponential kernel* (deutsch: quadrierter exponentieller Kernel) oder *Gaussian kernel* (deutsch: Gauss Kernel), welcher definert is als:

$$\kappa(x, x') = exp\left(-\frac{1}{2}(x - x')^T \Sigma^{-1}(x - x')\right) \tag{4.7}$$

wobei Σ die Kovarianzmatrix von X ist. Ist Σ sphärisch, erhält man den isotropen Kernel:

$$\kappa(x, x') = exp\left(-\frac{||x - x'||^2}{2\sigma^2}\right) \tag{4.8}$$

wobei σ^2 als Bandbreite bezeichnet wird. Dies ist der *Gauss-Kernel* und ein Beispiel für ein RBF-Kernel, da die Funktion nur auf $||x - x'||^2$ basiert [Murphy, 2013]. Weitere Beispiele sind der *lineare Kernel*:

$$\kappa(x, x') = -||x - x'|| \tag{4.9}$$

als die einfachste Form eines Kernels oder auch der *multiquadratische Kernel*:

$$\kappa(x, x') = -\sqrt{1 + ||x - x'||^2} \tag{4.10}$$

Oridnary Kriging

Ordinary Kriging (OK) gehört zu den geostatistischen Vorhersagemodellen und ist eine Variante des *Kriging*. *Kriging* ist Teil des geostatistischen Schätzverfahren [Gau, 2011]. Hierbei werden folgende Schritte durchlaufen (Ablauf basierend auf [Gau, 2011, Abb. 3-4]):

1. Planung und Durchführung der Erkundungskampagne
2. Datenerhebung
3. Zusammenfassung zu Datenkollektiven
4. experimentelles Variogramm
5. theoretisches Variogramm
6. Kriging
7. Visualisierung

Der Ablauf, der in dieser Arbeit angewendet wird, wurde bereits in [Braatz, 2019] ausführlich beschrieben und wird hier nur kurz zusammengefasst:
Begonnen wird mit der Erstellung des *experimentellen Variogramms*. Hierbei wird die Veränderung der Unähnlichkeit der Messwerte relativ zu der Entfernung der jeweilgen Messpunkte dargestellt. Sofern eine Autokorrelationsstruktur vorliegt, ist erkennbar, dass nahegelegene Messungen sich ähnlicher sind als weiter entfernte.
Im nächsten Schritt wird ein *theoretisches Variogramm* erstellt, welches das *experimentelle Variogramm* bestmöglich annähert. Diese wird dann innerhalb des Krigings genutzt, um bei der Schätzung die Autokorrelationsstruktur zu berücksichtigen. OK nutzt für die Interpolation ebenfalls einen gewichteten Mittelwert. Die Gewichte basieren auf dem Variogramm der umliegenden Messungen. Dadurch werden neben der Nähe zweier Messpunkte auch Redundanzen berücksichtigt. Dementsprechend fallen zwei Referenzmesspunkte, die sich nahe sind, nicht doppelt ins Gewicht. Außerdem erhalten Messwerte die „hinter" einem anderen Messpunkt liegen ein geringeres Gewicht [Heinrich, 1994].

4.5.2 Durchführung und Ergebnisse

Die Interpolation wird mit IDW, OK und RBF getestet. Dabei werden bei RBF die Kernelfunktionen „Linear" und „Quadratic" genutzt. Jedes Modell wird je Zeitschritt angewendet. Die Metriken R2, RMSE und MAE werden für die Modelle 5-fach-kreuzvalidiert. Für die Distanzberechnung wird die Haversine-Distanz genutzt (siehe Abschnitt 4.4.1). IDW wird mithilfe der Python-Bibliothek *numpy* [Harris et al, 2020] implementiert, wobei die Distanzmatrix mit der Bibliothek *scipy* [McKinney, 2010] berechnet wird.
OK wird mit der Bibliothek *gstools* [Mirko Mälicke et al, 2021] umgesetzt. Die Bibliothek bietet zusätzlich die Möglichkeit, das theoretische Variogramm automatisiert bestimmen zu lassen. In dieser Arbeit wird für jeden Zeitschritt erneut überprüft, welches Variogrammmodel das experimentelle Variogramm am besten annähert. Innerhalb von *gstools* wird dafür R2 genutzt. Eine Liste der möglichen Variogrammmodelle ist in der Dokumentation von *gstools* angegeben[9].
Die beiden RBF-Modelle werden mit der Bibliothek *scipy* umgesetzt. Hierbei wurde nicht die aktuelle Implementierung (scipy.interpolate.RBFInterpolator), sondern die vorherige genutzt (scipy.interpolate.Rbf), da diese eine selbstdefinierte Distanzfunktion erlaubt. Die aktuelle Implementierung hat, nach bestem Wissen des Autors, keine Möglichkeit,

[9] https://geostat-framework.readthedocs.io/projects/gstools/en/stable/examples/03_variogram/01_find_best_model.html#sphx-glr-examples-03-variogram-01-find-best-model-py (14.04.2022)

Tabelle 4.4: Mittelwert der Metriken der Interpolationsmodelle je kalibriertem P_{10} (P1_cal) und $P_{2.5}$ (P2_cal) über alle betrachteten Zeitschritte.

PM	Modell	MAE	RMSE	R2
P1_cal	IDW	1,62	2,32	0,72
P1_cal	OK	1,34	2,06	0,75
P1_cal	RBF(Multiquadratic)	0,86	1,47	0,88
P1_cal	RBF(Linear)	**0,83**	**1,42**	**0,89**
P2_cal	IDW	1,14	1,66	0,65
P2_cal	OK	0,99	1,48	0,71
P2_cal	RBF(Multiquadratic)	0,80	1,24	0,79
P2_cal	RBF(Linear)	**0,74**	**1,16**	**0,81**

Geokoordinaten zu verarbeiten. Das Modell mit den besten Metriken wird im Anschluss genutzt, um ein gleichmäßiges Raster je Zeitstempel zu schätzen. Dieses Raster über die Zeit wird in dem nachfolgendem Abschnitt für die Prognose verwendet. Zusätzlich wird evaluiert, welche Zellgröße für das Raster in den folgenden Analysen genutzt werden soll. Hierfür werden die Zellgrößen 0,5 km, 1 km, 5 km, 10 km, 20 km und 50 km in Bezug auf Rechenzeit und Verteilung über dem Gebiet betrachtet.

Modellvergleich

Die Ergebnisse des Vergleichs sind in Abb. 4.6 dargestellt. In der Abbildung ist die Verteilung der Metriken illustriert, die je Zeitschritte berechnet wurden. Die Metriken sind zusätzlich in der Tabelle 4.4 aufgelistet. Alle Modelle haben überwiegend geringe Fehler in MAE und RMSE. Der Median R2-Score ist bei allen Modellen über 0,7 außer bei IDW für P_{10}. OK hat eine größere Streuung in allen Metriken und zusätzlich deutliche Ausreißer, die in der Abbildung nicht mehr gezeigt sind. Insgesamt haben beide RBF Modelle den geringsten Fehler und den höchsten R2-Score, wobei das RBF-Modell mit dem linearen Kernel ein noch besseres Ergebnis erzielt. Dieses Modell wird für die weiteren Analysen verwendet.

Interpolationsbeispiel

Für die Evaluierung der Zellgröße des Rasters werden für das Bundesland Hessen unterschiedlich Zellgrößen verglichen. In Abb. 4.7 auf Seite 49 und Abb. 4.8 auf Seite 50 werden die Ergebnisse des Vergleichs grafisch dargestellt. In den Abbildungen werden nur

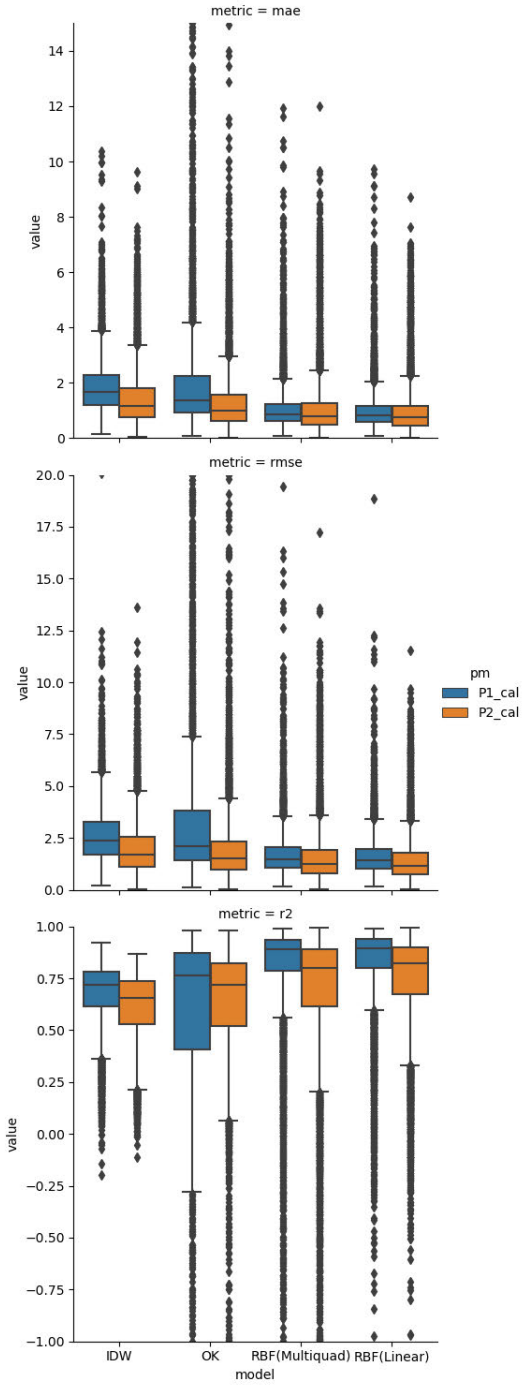

Abbildung 4.6: Vergleich der Interpolationsmodelle (IDW, OK, RBF(Multiquadratic, RBF(Linear)) für die kalbrierten Feinstaubmesswerte P_{10} (P1_cal) und $P_{2.5}$ (P2_cal). In den Zeilen sind die Metriken MAE, RMSE und R2. Die Verteilung beschreibt die Metriken für die betrachten Zeitschritte. Der Wertebereich der Metriken wurde zur bessere Übersicht begrenzt. Insbesondere OK hat extreme Ausreißer.

die Teile des Rasters gezeigt, die innerhalb der Landesgrenzen von Hessen liegen. Im Falle von 50 km großen Zellen ist die Anzahl der interpolierten Punkte innerhalb der Grenzen minimal. Bei 20 km ist die Verteilung der Interpolationspunkte erkennbar und zuvor freie Flächen wirken gleichmäßig gefüllt. Bei 10 km und 5 km wird die Verteilung dichter und teilweise sind mehr Interpolationspunkte vorhanden, als Messungen. Bei einer Zellgröße von 1 km und 0,5 km sind in der Darstellungsweise keine einzelnen Interpolationspunkte erkennbar.
Die Berechnungsdauer für den betrachteten Zeitraum von einem Monat (16.12.2019 bis 16.01.2020) ist in Tabelle 4.5 aufgeführt. Insbesondere die Berechnung des Rasters mit einer Zellgröße von 0,5 km hat bereits über fünf Stunden für die Interpolation benötigt. Die kleineren Zellgrößen konnten schneller berechnet werden, allerdings war das Beispiel auf einen Monat und die Fläche Hessens begrenzt. Die Fläche Hessens beträgt etwa 6 % der Fläche von Deutschland. Da insbesondere die Anzahl der zu interpolierenden Punkte Auswirkung auf die Berechnungszeit hat, wird für die weiteren Analysen dieser Arbeit eine Zellgröße von 20 km genutzt.

Tabelle 4.5: Die benötigte Berechnungsdauer für die Interpolation für verschiedene Rastergrößen am Beispiel Hessen für den Zeitraum von einem Monat

Location	Model	Cell size [km]	Points on Lat.	Points on Lon.	# Points	Computation time
Hessen	RBF(Linear)	0,5	500	354	177.000	05:15:39
Hessen	RBF(Linear)	1	250	177	44.250	00:20:40
Hessen	RBF(Linear)	5	50	35	1.750	00:05:35
Hessen	RBF(Linear)	10	25	17	425	00:02:40
Hessen	RBF(Linear)	20	12	8	96	00:01:57
Hessen	RBF(Linear)	50	5	3	15	00:01:51

4.6 Prognose

In diesem Abschnitt wird ein Modell für die kurzfristige Feinstaubprognose entwickelt. Die Vorhersage wird erstellt auf Basis der letzten 24 Stunden und prognostiziert die folgende Stunde. Für die Prognose wird ein Deep-Neural-Network mit ConvLSTM-Schichten erstellt. Das Modell wird mit dem Datensatz trainiert, der über die vorangegangenen Abschnitte aufgebaut wurde.

4.6.1 Material und Methoden

Datensatz

In diesem Abschnitt wir der resultierende Datensatz aus dem Abschnitt 4.4 genutzt. Der Datensatz enthält ein gleichmäßiges Raster an Feinstaubdaten für P_{10} und $P_{2.5}$. Das Raster hat eine Zellgröße von 20 km. Das ergibt pro Zeitstempel 1.562 Datenpunkte je Messwert. In diesem Abschnitt wird das Prognose Modell aus zeitlichen Gründen ausschließlich für P_{10} entwickelt. Daher wird nur dieser Teil des Datensatzes betrachtet.

ConvLSTM

ConvLSTM wurden von [Shi et al, 2015] für die kurzfristige Vorhersage von Niederschlag entwickelt. In dieser Arbeit wird diese Art eines neuronalen Netzes auf die kurzfristige Entwicklung von Feinstaub angewandt. Ein *ConvLSTM*-Netz ist eine Kombination aus einem LSTM-Netz und einem CNN, wobei in der Berechnung der Gates einer LSTM-Zelle die Matrix-Vektor-Multiplikation der Gewichte durch eine Faltungsoperation ersetzt wird (siehe Gleichung 4.11 auf Seite 51). Dementsprechend werden keine einzelnen Gewichte trainiert, sondern ein Faltungskern, der *Feature Maps* erzeugt. Diese *Feature Maps* repräsentieren eine Eigenschaft in dem betrachteten Bild. Auf diese Weise können räumliche Zusammenhänge erlernt werden und durch die Rückkopplung innerhalb einer LSTM-Zelle wird der zeitliche Kontext berücksichtigt [Shi et al, 2015].

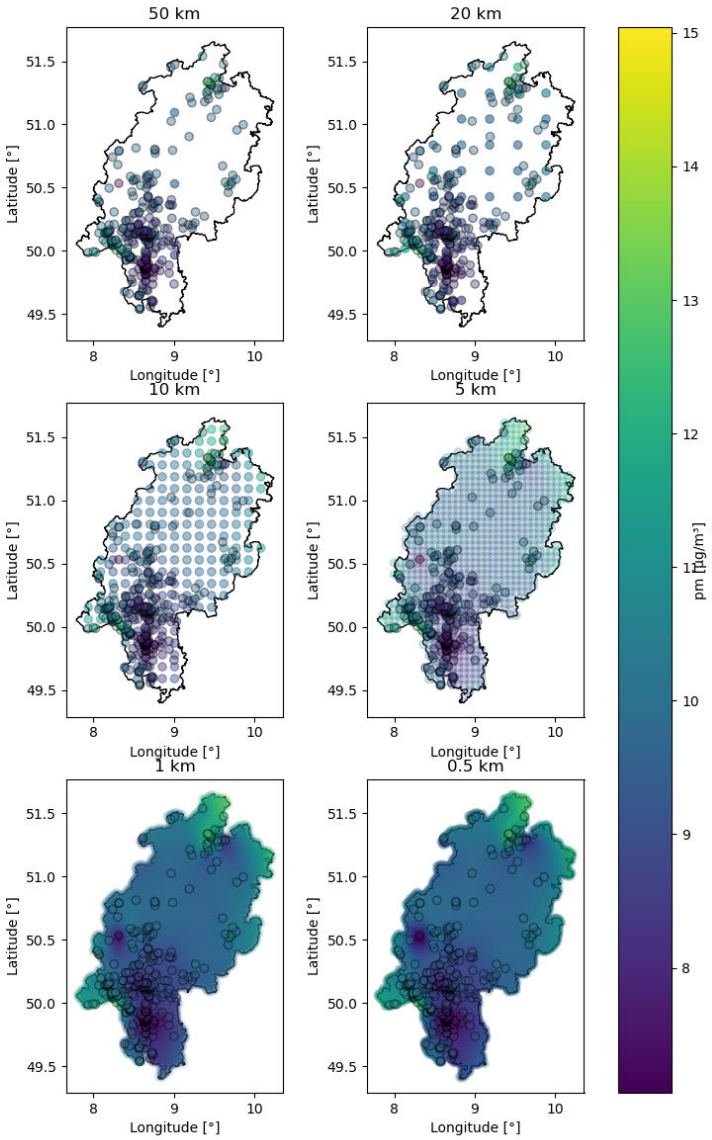

Abbildung 4.7: Interpolationsergebnisse für P_{10} in Hessen. Verglichen werden unterschiedliche Zellgrößen des zu schätzenden Rasters.

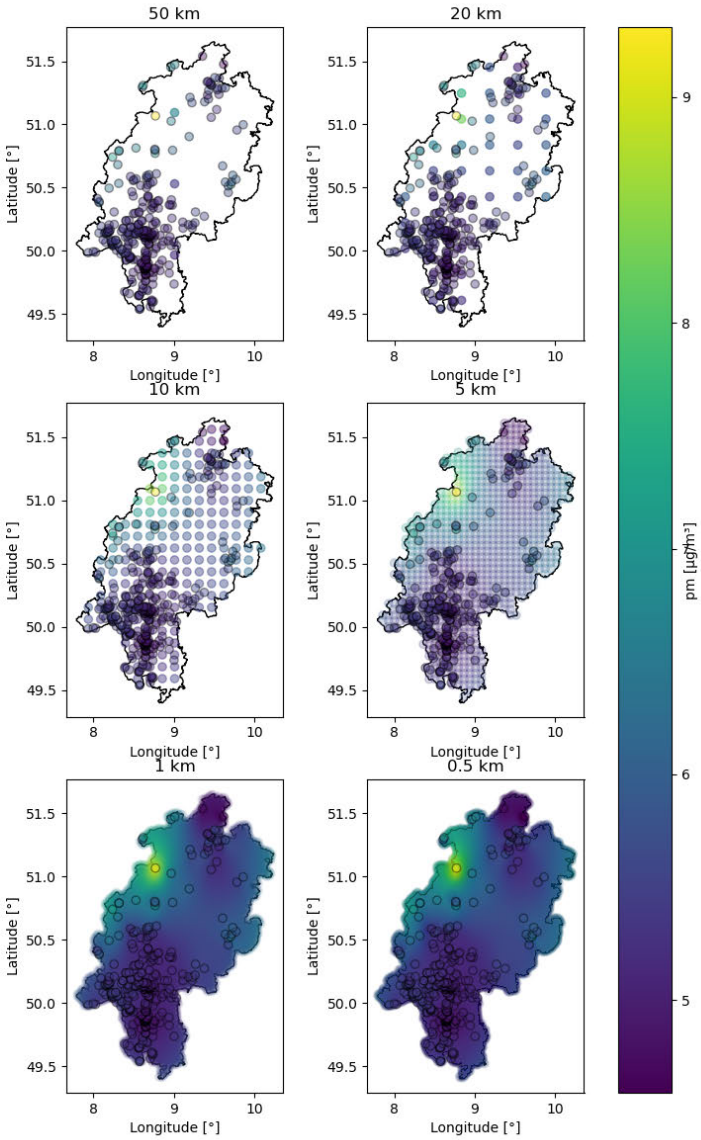

Abbildung 4.8: Interpolationsergebnisse für $P_{2.5}$ in Hessen. Verglichen werden unterschiedliche Zellgrößen des zu schätzenden Rasters.

$$
\begin{aligned}
i_t &= \sigma(W_{xi} * X_t + W_{hi} * H_{t-1} + W_{ci} \circ Ct - 1 + b_i) \\
f_t &= \sigma(W_{xf} * X_t + W_{hf} * H_{t-1} + W_{cf} \circ Ct - 1 + b_f) \\
C_t &= f_t \circ Ct - 1 + i_t \circ tanh(W_{xc} * X_t + W_{hc} * H_{t-1} + b_c) \\
o_t &= \sigma(W_{xo} * X_t + W_{ho} * H_{t-1} + W_{co} \circ Ct + b_o) \\
H_t &= o_t \circ tanh(C_t)
\end{aligned} \tag{4.11}
$$

Skalierung

ConvLSTM verarbeiten Daten in einem Wertebereich von [0-1]. Daher wird der Datensatz auf diesen Wertebereich skaliert. Hierfür wird die Bibliothek *sklearn* [Fabian Pedregosa et al, 2011] genutzt. Es werden drei unterschiedliche Methoden verglichen. „MinMax" skaliert die Daten auf den Wertebereich [0-1] über den maximalen und minimalen Wert:

$$x_{minmax} = (x - min(X)/(max(X) - min(X)) \tag{4.12}$$

wobei x ein Wert in der Menge X ist und x_{minmax} ist der skalierte Wert. „robustMinMax" ist eine Kombination aus MinMax und einem robusten Skalierer. Dieser zentriert die Daten um den Median und skaliert die Daten zwischen dem erstem und drittem Quartil. Dadurch ist der Skalierer robust gegenüber Ausreißern. Das Ergebnis des robusten Skalierers wird daraufhin durch MinMax auf den Wertebereich von [0-1] skaliert. „StandardMinMax" ist ähnlich aufgebaut zu „robustMinMax". Der Standard Skalierer ist wie folgt definiert:

$$x_{std} = (x - E(X))/\sigma(X) \tag{4.13}$$

wobei $E(X)$ der Mittelwert von X ist und σ die Standardabweichung. Das Ergebnis wird wiederum durch MinMax auf den Wertebereich von [0-1] skaliert.

4.6.2 Durchführung und Ergebnisse

Für die Transformation der Daten wird die Python-Bibliothek *numpy* [Harris et al, 2020] genutzt. Das ConvLSTM-Netz wird mit *Tensorflow* [Mart\'ın Abadi et al, 2015] erstellt.

Außerdem wird *Tensorflow* genutzt für die Überwachung des Trainings, durch *Tensorboard*. Bevor die Daten für das ConvLSTM Netz genutzt werden können müssen sie transformiert werden. Zuerst werden fehlende Zeitstempel durch mit NaN gefüllte Raster ersetzt. Daraufhin wird der Datensatz in „Beispiele" aufgeteilt. Ein Beispiel enthält 25 Stunden an Rasterdaten. Die vergangenen 24 Stunden werden als *Feature* und eine folgende Stunde als *Label* für das Training genutzt. Beispiele, die Raster mit NaN enthalten, werden entfernt.
Für die weitere Vorbereitung des Datensatzes wird sich an dem Dokumentationsbespiel[10] von Keras orientiert. Die Beispiele in dem Datensatz werden gemischt und anschließend wird der Datensatz in 80 % Trainingsdaten und 20 % Validierungsdaten geteilt. Daraufhin wird der Skalierer auf Basis der Tainingsdaten erstellt und sowohl Trainingsdaten als auch Validierungsdaten auf einen Wertebereich von [0-1] skaliert. Als letzten Schritt vor dem Training werden beide Datensätze in *Feature* und *Label* geteilt.
Das ConvLSTM-Netz wird in unterschiedlichen Konfigurationen trainiert und im Nachhinein verglichen, da, nach bestem Wissen des Autors, in der Literatur keine eindeutige Konfiguration vorherrscht und auch noch keine Anwendung die Vorhersage von Feinstaub betrachtet hat. Als Aktivierungs-Funktionen der letzten Schicht werden *ReLu*, *Linear* und *Sigmoid* getestet. Außerdem werden als *Loss*-Funktion *MSE*, *MAE* und *MAPE* (Mean Average Percentage Error) betrachtet. Der Aufbau der Netze wird mit unterschiedlichen Architekturen erprobt:

- drei, fünf und sieben Schichten á 64 LSTM-Zellen
- drei Schichten á 128 LSTM-Zellen
- zwei Schichten á 64 und zwei Schichten á 128 LSTM-Zellen

Als *Batchsize* wird 20 genutzt. Die ersten Tests werden mit 20 Epochen durchgeführt und die erfolgreicheren Konfigurationen werden im Anschluss noch mit 40 Epochen getestet. Es wird *Early-Stopping* bei geringer Entwicklung des Validierungs-*Loss* nach 10 Epochen genutzt und ebenso wird die Lernrate mit einem Beobachtungsfenster von 5 Epochen reduziert.
Die erfolgreichsten Modelle basierend auf dem RMSE der Validierungsdaten werden im Anschluss genutzt, um Prognosen für den gesamten Datensatz zu erstellen. Die Vorhersage-Fähigkeit der Modelle wird gegen die naive Annahme, den vorherigen Messwert zu nutzen, verglichen.

[10] https://keras.io/examples/vision/conv_lstm/ (16.04.2022)

Modellvergleich

In der Tabelle 4.6 sind die zehn besten Modelle basierend auf RMSE des Validation-*Loss* dargestellt. Die lineare Aktivierungsfunktion wird bei allen Modellen genutzt. Außerdem ist MAE die erfolgreichste *Loss*-Funktion. Der Skalierer „minMax" wird von den meisten und führenden Modellen genutzt, wobei „standardMinMax" am zweithäufigsten vertreten ist. Bei allen Modellen besteht kein oder kaum *Overfitting*. Die beiden besten Modelle mit fünf und drei Schichten je 64 LSTM-Zellen haben einen Abstand von etwa 19 % des RMSE zum dritten Modell. Diese beiden Modelle werden genutzt, um Prognosen für den gesamten Datensatz zu erstellen.

Tabelle 4.6: Übersicht der besten Prognose Modelle basierend auf dem RMSE der Validierungsdaten

Schichten	Aktivierung	Loss	Epochen	Skalierer	Loss (Train.)	Loss (Val.)	RMSE (Train.)	RMSE (Val.)
64-64-64-64-64	Linear	MAE	39	minMax	8,49e-4	8,48e-4	6,09e-4	**4,58e-4**
64-64-64	Linear	MAE	39	minMax	1,08e-4	1,08e-4	6,22e-4	4,83e-4
64-64-128-128	Linear	MAE	39	minMax	8,73e-5	8,91e-5	6,04e-4	5,94e-4
64-64-128-128	Linear	MAE	39	standardMinMax	8,57e-5	8,65e-5	5,77e-4	6,07e-4
128-128-128	Linear	MAE	39	minMax	1,94e-4	1,96e-4	7,22e-4	6,19e-4
64-64-128-128	Linear	MAE	39	robustMinMax	1,56e-4	1,57e-4	6,48e-4	6,82e-4
64-64-64-64-64-64-64	Linear	MAE	39	minMax	2,20e-4	2,21e-4	7,18e-4	7,67e-4
64-64-64-64-64	Linear	MAE	9	standardMinMax	3,10e-4	3,08e-7	8,54e-4	8,81e-4
64-64-128-128	Linear	MSE	39	standardMinMax	7,36e-7	7,90e-7	8,58e-4	8,89e-4
64-64-64-64-64	Linear	MSE	35	minMax	7,97e-7	8,53e-7	8,93e-4	9,23e-4

Tabelle 4.7: Vergleich der Metriken der beiden besten Prognosemodelle basierend auf dem RMSE der Validierungsdaten und dem naiven Ansatz des letzten Messwerts

Model	MAE	RMSE	R2
P1_cal_5x64	**0,74**	**2,41**	**0,93**
P1_cal_3x64	0,76	2,42	0,93
P1_cal_shifted	0,90	2,79	0,91

Prognose Ergebnisse

In der Tabelle 4.7 sind die Metriken der Prognosemodelle aufgeführt und zusätzlich die Metriken für die naive Annahme, als Prognose den letzten Messwert zu nutzen. Zu erkennen ist, dass beide Modelle einen besseren R2-Score haben, als die naive Methode und die Fehlermaße ebenfalls geringer sind. Außerdem hat das Modell mit den fünf Schichten á 64 LSTM-Zellen bessere Metriken, als das Modell mit den drei Schichten, passend zu den besseren Ergebnissen im Training.
In Abb. 4.9 ist ein Ausschnitt der Prognosen gezeigt, wobei der Mittelwert über das Raster dargestellt ist. Der Verlauf beider Prognosemodelle ist näher an dem der naiven Methode, als an dem Verlauf der Messungen. Insbesondere starke Änderungen der Messungen verfehlen die Prognosen. Bei langsameren und gleichbleibenden Änderungen sind die Prognosen allerdings besser als die naive Methode.

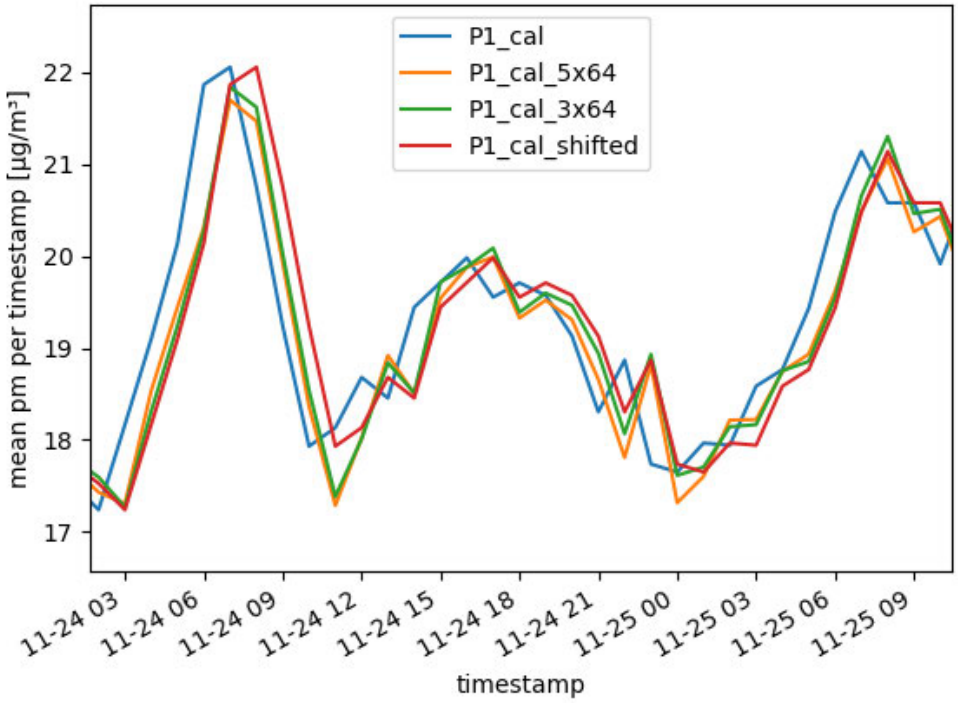

Abbildung 4.9: Ausschnitt aus den Prognosen für P_{10} (P1_cal) mit den beiden besten Modellen unterschieden durch den Aufbau des Schichten des Neuronalen Netzes (5x64 und 3x64). Zusätzlich sind die ursprünglichen Daten versetzt um eine Stunde dargestellt (shifted).

5 Diskussion und Ausblick

5.1 Kalibrierung der Sensordaten

In Abschnitt 4.4 wurde die Kalibrierung der Sensoren untersucht. Dabei wurden *Geographically Weighted Regression* und *Lineare Regression* genutzt. Die beiden Modelle wurden mit jeweils fünf unterschiedlich großen Einzugsbereichen um eine Referenzstation verglichen. Außerdem wurde untersucht, inwiefern zusätzliche Temperatur- und Feinstaubdaten die Kalibrierung beeinflussen. Das erfolgreichste Modell nutzte *Geographically Weighted Regression*, ausschließlich Feinstaubdaten in einem Einzugsbereich von 5 km. Dieses Modell wurde im Anschluss auf den gesamten Datensatz angewandt und erzielte einen *R-Square*-Score von 0,70 für P_{10} und 0,79 für $P_{2.5}$. Die Fehlermaße *Mean Absolut Error* und *Root Mean Squared Error* lagen für P_{10} bei 3,5 und 6,99 und für $P_{2.5}$ bei 2,06 und 3,99. Damit gab es eine deutliche Verbesserung im Vergleich zu den Messungen vor der Kalibrierung.
Ein Teil der Temperatur- und Luftfeuchtigkeitsdaten wurden durch die naheliegendste Wetterstation des Deutschen Wetterdienst ergänzt. Wie weit die nächste Station letztlich von der Sensorstation entfernt ist, wurde nicht betrachtet. Mit Interpolationsmethoden, wie sie in Abschnitt 4.5 gezeigt wurden, könnten in zukünftigen Analysen die fehlenden Daten aus einer größeren Anzahl umliegender Wetterstationen geschätzt werden. Hierbei würde die räumliche Entwicklung der Werte beachtet werden und möglicherweise passendere Werte ermittelt werden.
In dieser Arbeit wurden räumliche und zeitliche Analysen für Feinstaubdaten untersucht. Die Kalibrierung wurde dabei ausschließlich in einem räumlichen Zusammenhang betrachtet und für jeden Zeitstempel wurde ein separates Kalibrierungsmodell bestimmt. [Que et al, 2020] stellt *Spatiotemporal Weighted Regression* vor. Dieses Modell ist eine Weiterentwicklung von *Geographically Weighted Regression* und betrachtet zusätzlich den zeitlichen Verlauf. Über diesen Ansatz könnte ein Modell erstellt werden, welches auf alle Zeitstempel des Datensatzes angewendet werden kann. Dadurch wäre das Modell zusätzlich robuster gegen fehlende Referenzwerte zu einzelnen Zeitstempeln.

Die Analyse wurde für die gesamte Fläche Deutschlands durchgeführt. In zukünftigen Untersuchungen könnte die Nutzung von Temperatur- und Luftfeuchtigkeitsdaten in einem kleineren, lokalen Bereich überprüft werden. Für einige Teilbereiche können unterschiedlich große Einzugsbereiche sinnvoll sein, da ortsbezogene Besonderheiten Auswirkungen auf die Feinstaubemission haben. Daher wurde in [Barkjohn et al, 2021] eine sehr spezifische Teilmenge an Sensoren betrachtet. Unter Berücksichtigung dieser Besonderheit waren die Modelle, welche zusätzlich Temperatur- und Luftfeuchtigkeitsdaten nutzten, besser, als das Modell, welches ausschließlich Feinstaubdaten nutzt.

5.2 Interpolation zu einem gleichmäßigen Raster

In Abschnitt 4.5 wurde auf Basis der zuvor kalibrierten Daten ein gleichmäßiges Raster geschätzt. Dafür wurden die räumlichen Interpolationsmodelle *Inverse Distance Weighting*, *Radial Basis Function* und *Ordinary Kriging* verglichen. Das *Radial Basis Function*-Modell wurde mit einer linearen und multiquadratischen Kernelfunktion untersucht. Hierbei waren die *Radial Basis Function*-Modelle besonders zuverlässig, wobei das Modell mit der linearen Kernelfunktion das beste war mit einem *R-Square*-Score von 0,89 für P_{10} und 0,81 für $P_{2.5}$. Die Fehlermaße *Mean Absolut Error* und *Root Mean Squared Error* lagen für P_{10} bei 0,83 und 1,42 und für $P_{2.5}$ bei 0,74 und 1,16. In einem zweiten Schritt wurden gleichmäßige Raster für das Bundesland Hessen mit unterschiedlichen Zellgrößen geschätzt. Aufgrund der benötigten Berechnungsdauer wurde für die Interpolation über Deutschland ein Raster mit einer Zellgröße von 20 km genutzt, da dieses eine vergleichbare Dichte zu den verfügbaren Sensorstationen hat.
In dem Rastergrößenvergleich über Hessen ist zu erkennen, dass es eine Ballung von Sensoren in urbanen Regionen gibt. [Malings et al, 2020] hat bereits berichtet, dass die Nutzung von Low-Cost-Sensoren besonders passend für den urbanen Raum ist aufgrund der Größe und einfachen Einrichtung. Sollte eine Analyse ausschließlich einen kleinen urbanen Bereich betrachten, kann es sinnvoll sein, ein feineres Raster zu wählen, damit die Unterschiede zwischen einzelnen Sensorstationen besser erfasst werden.
Bei großflächigen Analysen, wie in dieser Arbeit, können gröbere Raster gewählt werden. Dabei bleiben die erstellten Interpolationsmodelle gleich und können je nach Bedarf unterschiedliche Rastergrößen schätzen.
In den vorherigen Arbeiten konnte die Effektivität von *Ordinary Kriging* dargestellt werden. *Ordinary Kriging* benötigt eine gleichbleibende Autokorrelationsstruktur in einem

betrachteten Bereich. In zukünftigen Analysen für kleinere Bereiche kann erneut evaluiert werden, inwiefern *Ordinary Kriging* eventuell besser geeignet ist als *Radial Basis Function*.

5.3 Kurzfristige Prognose der Feinstaubentwicklung

In Abschnitt 4.6 wurde ein neuronales Netz für die kurzfristige Vorhersage von Feinstaubdaten entwickelt. Dabei wurden unterschiedliche Architekturen und Konfigurationen eines *Convolutional Long short-term Memory* (ConvLSTM)-Netzes für die Vorhersage von P_{10} verglichen. Es wurden unter anderem Kombinationen verschiedener Größen der Schichten des neuronalen Netzes, Aktivierungs- und Lossfunktionen und Skalierungsmethoden untersucht. Hierbei wurde gezeigt, dass die lineare Aktivierungsfunktion und *Mean Absolut Error* als Lossfunktion zu den besten Ergebnissen führt. Der „minMax"-Skalierer wurde bei den drei besten Modellen genutzt, wobei andere Skalierer ebenfalls in den besten zehn Modellen genutzt wurden. Die Modelle mit 64 ConvLSTM-Zellen, sowie fünf und drei Schichten haben einen Abstand von 23 % beziehungsweise 19 % zu dem drittbesten Modell auf Basis des *Root Mean Squared Error* zu den Validierungsdaten. Beide Modelle wurden auf den Datensatz angewendet und mit dem naiven Ansatz verglichen, den letzten Messwert als Prognose zu nutzen. Dabei konnten beide Modelle das naive Modell schlagen mit einem *R-Square*-Score von je 0,93. Die naive Methode hat einen Score von 0,91. In der graphischen Darstellung ist allerdings zu erkennen, dass insbesondere bei starken Änderungen der Werte die Modelle näher an der naiven Methode sind, als an der korrekten Prognose.
Die Entwickler von ConvLSTM [Shi et al, 2015] nutzten für die Prognosen von Niederschlag Daten, die in einem sechs-Minuten-Takt aufgenommen wurden. In zukünftigen Arbeiten kann die Frequenz der Daten erhöht werden. Da die Daten des Deutschen Wetterdienst, sowie des Bundesumweltamts nur stündlich zur Verfügung stehen, müsste zur Kalibrierung beispielsweise *Spatiotemporal Weighted Regression* genutzt werden. Damit ist die Kalibrierung unabhängig von der Datenfrequenz der Referenz- und ergänzenden Daten.
Außerdem kann in einer kommenden Analyse untersucht werden, inwiefern die Prognose in einem kleineren Bereich mit einem feineren Raster auch Entwicklungen in einem Mikroklima vorhersagen kann.

5.4 Generalisierung und Übertragbarkeit

Der in dieser Arbeit vorgestellte Analyseprozess nutzt die Daten aus einem crowd-based Sensornetz und verarbeitet die Daten auf eine Weise, dass letztlich Prognosen auf Basis eines gleichmäßigen Rasters erstellt wurden. Dieser Prozess wird in dieser Arbeit am Beispiel von Feinstaubdaten vorgestellt. Allerdings kann dieser Prozess prinzipiell auf alle raumzeitlich gemessenen Daten angewendet werden.
Der Schritt der Kalibrierung kann genutzt werden, wenn die betrachteten Daten mit Sensoren erfasst wurden, die nicht kalibriert sind. Mit Hilfe von umliegenden Referenzstationen können die Sensoren mit *Geographically Weighted Regression* kalibriert werden. Insbesondere Klima- und Wetterdaten werden von staatlichen Institutionen gesammelt und werden durch hochwertige Messinstrumente erfasst.
Die Interpolation ermöglicht es, Daten von ungleichmäßig verteilten Messstation in ein gleichmäßiges Raster zu schätzen. Dadurch wird ermöglicht, Methoden des Machine Learning und Deep Learning zu nutzen, die ein gleichmäßiges Raster als Datenformat benötigen.
Die Möglichkeit beliebig verteilte, räumliche Daten für KI-Verfahren nutzbar zu machen ist besonders interessant im Bereich von *Internet of Things* (IoT). Jedes Gerät, das Messungen vornimmt und eine Verbindung zum Internet hat, kann für die Analyse genutzt werden, sofern bekannt ist, zu welcher Zeit an welchem Ort die Daten erfasst wurden. Das können Fahrzeuge im Straßenverkehr sein, die Daten über den derzeitigen Verkehr kommunizieren, wie die Geschwindigkeit des Fahrzeugs. In einer *Smart City* können stationäre Messstationen nicht immer in einem gleichmäßigen Raster positioniert werden. Diese Daten können mit dem vorgestellten Analyseprozess verarbeitet werden, um Auskunft über den Zustand der Stadt oder der Entwicklung des Verkehrs zu geben.
Die Interpolationsmodelle können im Beriech *Industrie 4.0* genutzt werden, um beispielsweise bei der Überwachung von Reinräumen weniger Sensoren für die Überwachung von Unreinheiten zu benötigen. Die Bereiche zwischen den Sensoren würden dann mit den Modellen interpoliert werden.
In einer Welt, in der immer mehr Daten erfasst und Infrastruktur, Städte und Entscheidungen durch Daten getrieben werden, bietet der vorgestellte Prozess die Möglichkeit, zwischen verteilten Sensoren räumlich zu approximieren und ein ergänzendes Bindeglied zwischen unterschiedlichen Domänen, sowie einer professionellen und crowd-based Datenerfassung.

6 Danksagung

An dieser Stelle möchte ich mich bei Sensor.Community[1] bedanken für das Projekt an sich und für die Bereitstellung der Sensordaten.
Ebenso möchte ich mich bei dem Bundesumweltamt[2] für Bereitstellung der Referenzdaten bedanken.
Diese Arbeit hätte nicht durchgeführt ohne die zur Verfügung gestellten Mittel des FTZ SMSY[3], daher möchte ich auch dem CSTI der HAW Hamburg danken.

[1] https://sensor.community/de/
[2] https://www.umweltbundesamt.de/
[3] https://smsy.haw-hamburg.de/

Literaturverzeichnis

(07.06.2019) Datenmenge explodiert. IWD URL https://www.iwd.de/artikel/datenmenge-explodiert-431851/

(09.04.2022) Numeracy, maths and statistics - academic skills kit. URL https://www.ncl.ac.uk/webtemplate/ask-assets/external/maths-resources/statistics/regression-and-correlation/coefficient-of-determination-r-squared.html

(27.03.2022) Wetter und klima - deutscher wetterdienst - aufgaben. URL https://www.dwd.de/DE/derdwd/aufgaben/aufgaben_node.html

Atluri G, Karpatne A, Kumar V (2018) Spatio-temporal data mining. ACM Computing Surveys 51(4):1–41, DOI 10.1145/3161602

Barkjohn KK, Gantt B, Clements AL (2021) Development and application of a united states wide correction for pm2.5 data collected with the purpleair sensor. Atmospheric Measurement Techniques 4(6):4617–4637, DOI 10.5194/amt-14-4617-2021, URL https://amt.copernicus.org/articles/14/4617/2021/amt-14-4617-2021-discussion.html

Benjamin Gutzmann, Andreas Motl, Daniel Lassahn, Ilya Kamenshchikov, Max Bachmann, Michael Schrammel (2021) earthobservations/wetterdienst: Start migrating from dogpile.cache to filesystem_spec. DOI 10.5281/ZENODO.5501071

Braatz A (2019) Raumzeitliches data-mining in dynamischen sensorsystemen: Raumzeitliches data-mining in dynamischen sensorsystemen. Bachelor's thesis, Hochschule für angewandte Wissenschaften Hamburg, URL https://reposit.haw-hamburg.de/handle/20.500.12738/9152?&locale=de

Braatz A (2021a) Evaluation von ausreißer-behandlung und auswirkungen von umwelteinflüssen auf feinstaub-analysen. URL https://users.informatik.haw-hamburg.de/~ubicomp/projekte/master2021-proj/braatz.pdf

Braatz A (2021b) Räumliche interpolation von feinstaub-sensordaten mit hilfe von kriging. URL https://users.informatik.haw-hamburg.de/~ubicomp/projekte/master2021-proj/braatz_hp.pdf

Capgemini Deutschland (2020) Studie: Datengetriebene unternehmen erzielen mehr umsatz und gewinn. URL https://www.capgemini.com/de-de/news/studie-datengetriebene-unternehmen/

Chai T, Draxler RR (2014) Root mean square error (rmse) or mean absolute error (mae)? – arguments against avoiding rmse in the literature. Geoscientific Model Development 7(3):1247–1250, DOI 10.5194/gmd-7-1247-2014

Chu HJ, Ali MZ, He YC (2020) Spatial calibration and pm2.5 mapping of low-cost air quality sensors. Scientific Reports 10(1):22,079, DOI 10.1038/s41598-020-79064-w, URL https://www.nature.com/articles/s41598-020-79064-w#Equ4

Dargan S, Kumar M, Ayyagari MR, Kumar G (2020) A survey of deep learning and its applications: A new paradigm to machine learning. Archives of Computational Methods in Engineering 27(4):1071–1092, DOI 10.1007/s11831-019-09344-w, URL https://link.springer.com/article/10.1007/s11831-019-09344-w

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, Édouard Duchesnay (2011) Scikit-learn: Machine learning in python. Journal of Machine Learning Research 12(85):2825–2830, URL http://jmlr.org/papers/v12/pedregosa11a.html

Fayyad U, Piatetsky-Shapiro G, Smyth P (1996a) From data mining to knowledge discovery in databases. AI Magazine 17(3):37, DOI 10.1609/aimag.v17i3.1230, URL https://ojs.aaai.org//index.php/aimagazine/article/view/1230

Fayyad U, Piatetsky-Shapiro G, Smyth P (1996b) Knowledge discovery and data mining: Towards a unifying framework. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, AAAI Press, KDD'96, pp 82–88

Gau C (2011) Geostatistik in der Baugrundmodellierung: Die Bedeutung des Anwenders im Modellierungsprozess. SpringerLink Bücher, Springer Fachmedien, Wiesbaden, DOI 10.1007/978-3-8348-9774-9

Giordano MR, Malings C, Pandis SN, Presto AA, McNeill VF, Westervelt DM, Beekmann M, Subramanian R (2021) From low-cost sensors to high-quality data: A summary of challenges and best practices for effectively calibrating low-cost particulate matter mass sensors. Journal of Aerosol Science 158:105,833, DOI 10.1016/j.jaerosci.2021.105833, URL https://www.sciencedirect.com/science/article/pii/S0021850221005644

Hagler GSW, Williams R, Papapostolou V, Polidori A (2018) Air quality sensors and data adjustment algorithms: When is it no longer a measurement? Environmental Science & Technology 52(10):5530–5531, DOI 10.1021/acs.est.8b01826

Harris CR, Millman KJ, van der Walt SJ, Gommers R, Virtanen P, Cournapeau D, Wieser E, Taylor J, Berg S, Smith NJ, Kern R, Picus M, Hoyer S, van Kerkwijk MH, Brett M, Haldane A, Del Río JF, Wiebe M, Peterson P, Gérard-Marchant P, Sheppard K, Reddy T, Weckesser W, Abbasi H, Gohlke C, Oliphant TE (2020) Array programming with numpy. Nature 585(7825):357–362, DOI 10.1038/s41586-020-2649-2, URL https://www.nature.com/articles/s41586-020-2649-2

He T, Blum BM, Cao Q, Stankovic JA, Son SH, Abdelzaher TF (2007) Robust and timely communication over highly dynamic sensor networks. Real-Time Systems 37(3):261–289, DOI 10.1007/s11241-007-9025-2, URL https://link.springer.com/article/10.1007/s11241-007-9025-2

Heinrich U (1994) Flächenschätzung mit geostatistischen verfahren — variogrammanalyse und kriging. In: Schroder W (ed) Neuere statistische verfahren und modellbildung in der geookologie, Friedrick Vieweg & Son, [Place of publication not identified], pp 145–164, DOI 10.1007/978-3-322-83735-6{\textunderscore}10

Hengl T (2007) A practical guide to geostatistical mapping of environmental variables, EUR. Scientific and technical research series, vol 22904. Publications Office, Luxembourg

Henrikki Tenkanen, Vuokko Heikinheimo, Håvard Wallin Aagesen (23.02.2022) Nearest neighbor analysis with large datasets. URL https://autogis-site.readthedocs.io/en/latest/notebooks/L3/06_nearest-neighbor-faster.html

Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural computation 9(8):1735–1780, DOI 10.1162/neco.1997.9.8.1735

Hochreiter S, Bengio Y, Frasconi P, Schmidhuber J (2001) Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. In: Kremer SC, Kolen JF (eds) A Field Guide to Dynamical Recurrent Neural Networks, IEEE Press

Hussein II, Stipanovic DM (2006) Effective coverage control using dynamic sensor networks. In: Misra P (ed) 45th IEEE Conference on Decision and Control, 2006, IEEE Service Center, Piscataway, NJ, pp 2747–2752, DOI 10.1109/CDC.2006.377297

IONOS Digitalguide (03.04.2022) Docker-image. URL https://www.ionos.de/digitalguide/server/knowhow/docker-image/

Jiao W, Hagler G, Williams R, Sharpe R, Brown R, Garver D, Judge R, Caudill M, Rickard J, Davis M, Weinstock L, Zimmer-Dauphinee S, Buckley K (2016) Community air sensor network (cairsense) project: evaluation of low-cost sensor performance in a suburban environment in the southeastern united states. Atmospheric Measurement Techniques 9(11):5281–5292, DOI 10.5194/amt-9-5281-2016, URL https://amt.copernicus.org/articles/9/5281/2016/amt-9-5281-2016.html

Kisilevich S, Mansmann F, Nanni M, Rinzivillo S (1980) Spatio-temporal clustering. In: Kohonen T (ed) Content-addressable memories, Springer series in information sciences, Springer, Berlin and Heidelberg, pp 855–874, DOI 10.1007/978-0-387-09823-4{\textunderscore}44

LeCun Y, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, Jackel LD (1989) Backpropagation applied to handwritten zip code recognition. Neural computation 1(4):541–551, DOI 10.1162/neco.1989.1.4.541

LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444, DOI 10.1038/nature14539, URL https://www.nature.com/articles/nature14539

Li Z (2014) Spatiotemporal Pattern Mining: Algorithms and Applications, Springer International Publishing, Cham, pp 283–306. DOI 10.1007/978-3-319-07821-2_12, URL https://doi.org/10.1007/978-3-319-07821-2_12

Lv J, Li Q, Wang X (2018) T-conv: A convolutional neural network for multi-scale taxi trajectory prediction. BigComp URL https://arxiv.org/pdf/1611.07635

Malings C, Tanzer R, Hauryliuk A, Saha PK, Robinson AL, Presto AA, Subramanian R (2020) Fine particle mass monitoring with low-cost sensors: Corrections and long-term performance evaluation. Aerosol Science and Technology 54(2):160–174, DOI 10.1080/02786826.2019.1623863

Mart\'ın Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, Xiaoqiang Zheng (2015) Tensorflow: Large-scale machine learning on heterogeneous systems. URL https://www.tensorflow.org/

Mazimpaka JD, Timpf S (2016) Trajectory data mining: A review of methods and applications. Journal of Spatial Information Science (13), DOI 10.5311/JOSIS.2016.13.263

McKinney W (2010) Data structures for statistical computing in python. In: Proceedings of the 9th Python in Science Conference, SciPy, Proceedings of the Python in Science Conference, pp 56–61, DOI 10.25080/Majora-92bf1922-00a

Merkel D (2014) Docker: lightweight linux containers for consistent development and deployment. Linux journal 2014(239):2

Mirko Mälicke, Egil Möller, Helge David Schneider, Sebastian Müller (2021) mmaelicke/scikit-gstat: A scipy flavoured geostatistical variogram analysis toolbox. DOI 10.5281/ZENODO.4835779

Murphy KP (2013) Machine learning: A probabilistic perspective, 4th edn. Adaptive computation and machine learning series, MIT Press, Cambridge, Mass., URL http://www.cs.ubc.ca/%7Emurphyk/MLbook/

Oteafy SM, Hassanein HS (2017) Resilient iot architectures over dynamic sensor networks with adaptive components. IEEE Internet of Things Journal 4(2):474–483, DOI 10.1109/JIOT.2016.2621998

Petters MD, Kreidenweis SM (2007) A single parameter representation of hygroscopic growth and cloud condensation nucleus activity. Atmospheric Chemistry and Physics 7(8):1961–1971, DOI 10.5194/acp-7-1961-2007

Que X, Ma X, Ma C, Chen Q (2020) A spatiotemporal weighted regression model (stwr v1.0) for analyzing local nonstationarity in space and time. Geoscientific Model Development 13(12):6149–6164, DOI 10.5194/gmd-13-6149-2020

Rey SJ, Anselin L (2007) Pysal: A python library of spatial analytical methods. The Review of Regional Studies 37(1):5–27

Rose Eilenberg S, Subramanian R, Malings C, Hauryliuk A, Presto AA, Robinson AL (2020) Using a network of lower-cost monitors to identify the influence of modifiable factors driving spatial patterns in fine particulate matter concentrations in an urban environment. Journal of Exposure Science & Environmental Epidemiology 30(6):949–961, DOI 10.1038/s41370-020-0255-x, URL https://www.nature.com/articles/s41370-020-0255-x

Shepard D (1968) A two-dimensional interpolation function for irregularly-spaced data. In: Blue RB, Rosenberg AM (eds) Proceedings of the 1968 23rd ACM national conference on -, ACM Press, New York, New York, USA, pp 517–524, DOI 10.1145/800186.810616

Shi X, Chen Z, Wang H, Yeung DY, Wong Wk, Woo Wc (2015) Convolutional lstm network: A machine learning approach for precipitation nowcasting. URL https://arxiv.org/pdf/1506.04214

Tobler WR (1970) A computer movie simulating urban growth in the detroit region. Economic geography 46(sup1):234–240

Umweltbundesamt (06.03.2022) Wer wir sind. URL https://www.umweltbundesamt.de/das-uba/wer-wir-sind

Umweltbundesamt (13.03.2022) Feinstaub-belastung. URL https://www.umweltbundesamt.de/daten/luft/feinstaub-belastung#gesundheitliche-wirkungen

Umweltbundesamt (18.04.2022a) Feinstaub - pm2,5. URL https://www.umweltbundesamt.at/umweltthemen/luft/luftschadstoffe/staub/pm25

Umweltbundesamt (18.04.2022b) Umweltbewusstsein in deutschland. URL https://www.umweltbundesamt.de/themen/nachhaltigkeit-strategien-internationales/umweltbewusstsein-in-deutschland

Umweltbundesamt (Juni 2004) Qualitätsicherungshandbuch des uba-messnetzes. Texte | 28/2004

Wang L, Geng X, Ke J, Peng C, Ma X, Zhang D, Yang Q (2017) Ridesourcing car detection by transfer learning. URL https://arxiv.org/pdf/1705.08409

Wang S, Cao J, Yu P (2020) Deep learning for spatio-temporal data mining: A survey. IEEE Transactions on Knowledge and Data Engineering p 1, DOI 10.1109/TKDE.2020.3025580

Wheeler DC, Páez A (2010) Geographically weighted regression. In: Fischer MM, Getis A (eds) Handbook of Applied Spatial Analysis: Software Tools, Methods and Applications, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 461–486, DOI 10.1007/978-3-642-03647-7{\textunderscore}22

Zhang J (2011) Energy-efficient adaptive dynamic sensor scheduling for target monitoring in wireless sensor networks. ETRI Journal 33(6):857–863, DOI 10.4218/etrij.11.0111.0027

A Anhang

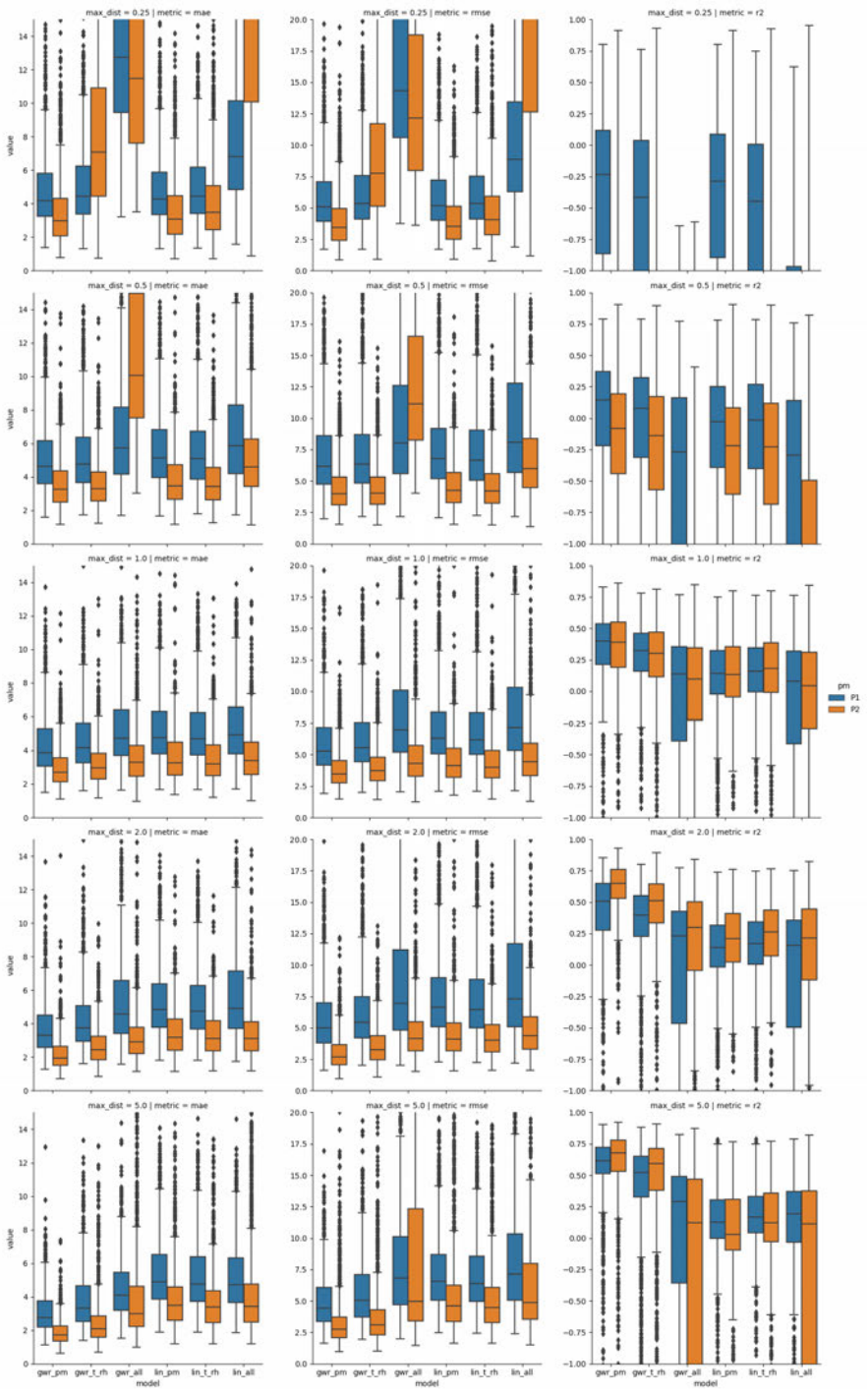

Abbildung A.1: Darstellung der Verteilung der Metriken je Kalibrierungsmodell und Parameterkombination für P_{10} (P1) und $P_{2.5}$ (P2) über alle Zeitstempel. Wobei die fehlenden Luftfeuchtigkeitswerte mit **relativer Luftfeuchtigkeit** ersetzt wurden. Je Grafik sind die sechs Kalibrierungsmodelle aufgezeichnet. In den Zeilen sind die betrachteten Entfernungen dargestellt und in den Spalten die Metriken (MAE, RMSE und R2). Für den Zeitraum 15.12.2017-15.01.2018.

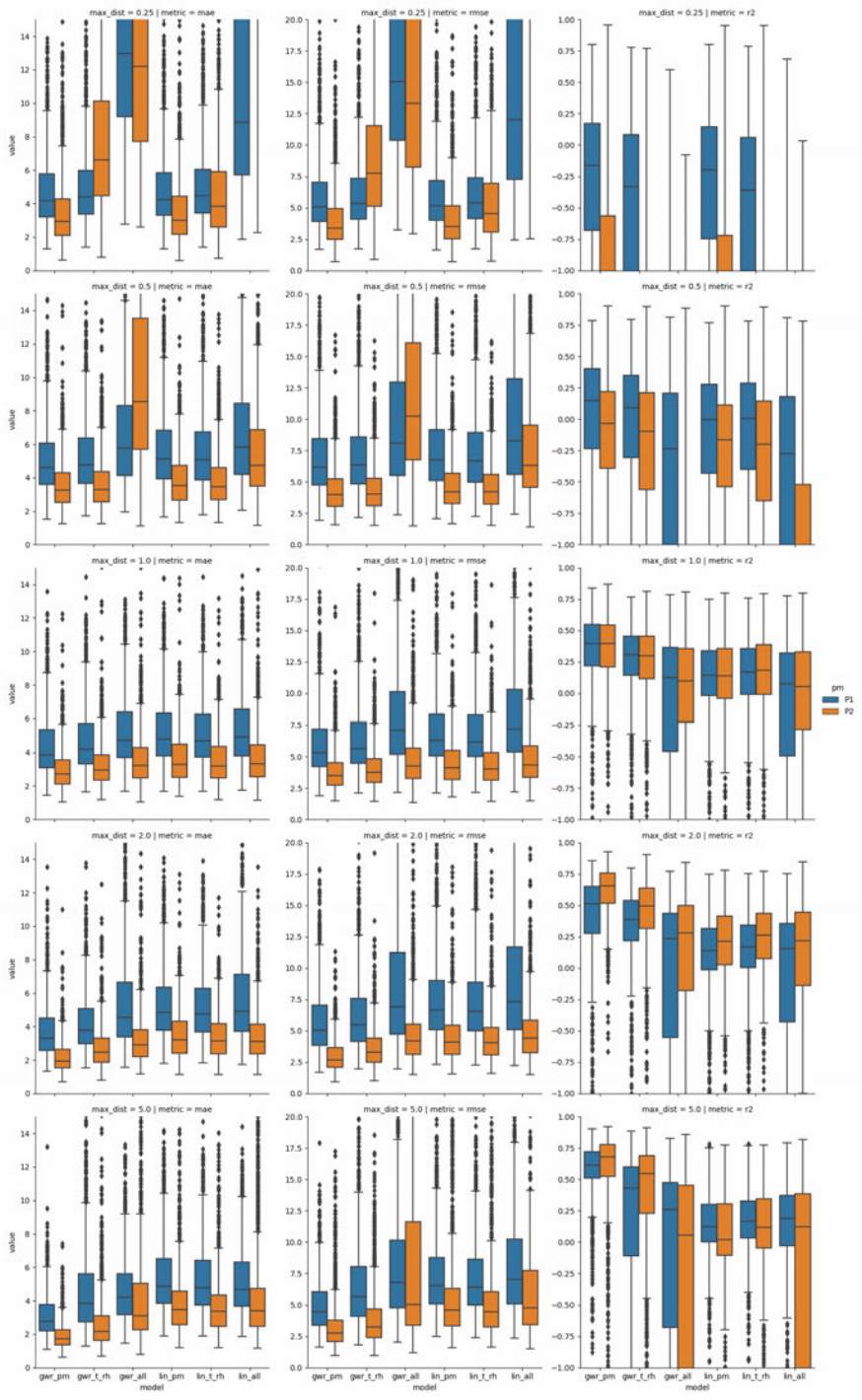

Abbildung A.2: Darstellung der Verteilung der Metriken je Kalibrierungsmodell und Parameterkombination für P_{10} (P1) und $P_{2.5}$ (P2) über alle Zeitstempel. Wobei die fehlenden Luftfeuchtigkeitswerte mit **absoluter Luftfeuchtigkeit** ersetzt wurden. Je Grafik sind die sechs Kalibrierungsmodelle aufgezeichnet. In den Zeilen sind die betrachteten Entfernungen dargestellt und in den Spalten die Metriken (MAE, RMSE und R2). Für den Zeitraum 15.12.2017-15.01.2018.

Erklärung zur selbstständigen Bearbeitung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

Ort	Datum	Unterschrift im Original