

Projekt: Edge AI DLBAIPEAI01_D

Fallstudie

Studiengang: Angewandte Künstliche Intelligenz

Sven Behrens

Matrikelnummer: 42303511

Prof. Dr. Bertram Taetz

13. Februar 2026

Inhaltsverzeichnis

Abbildungsverzeichnis	III
Tabellenverzeichnis	IV
Abkürzungsverzeichnis	V
1 Einleitung	1
2 Hauptteil	2
2.1 Projektumgebung	2
2.2 Datengrundlage	2
2.3 Datenvorverarbeitung	3
2.4 Modelltraining	3
2.4.1 Train/Test-Split	3
2.4.2 PM2.5-Modelle	4
2.4.3 NO2-Modelle	4
2.5 Feature Importance und Feature-Reduktion	5
2.6 Edge-Optimierung	6
2.6.1 ONNX-Export	6
2.6.2 Benchmark	6
2.7 Alert-System	6
2.8 Edge-Inference-Pipeline	6
3 Fazit	6
3.1 Zielerreichung und Projektergebnisse	6
3.2 Kritische Reflexion	6
3.3 Verbesserungspotenziale	6
3.4 Ausblick	6

Projektrepository	6
Literaturverzeichnis	7
Verzeichnis der Anhänge	8
Anhang	8

Abbildungsverzeichnis

Tabellenverzeichnis

1	Feature-Importance-Ranking für PM2.5 (Durchschnittsrang aus vier Methoden)	8
2	Modellvergleich PM2.5: Baseline (28 Features) vs. reduziertes Set (17 Features)	8
3	Modellvergleich NO2: Baseline (28 Features) vs. reduziertes Set (17 Features)	9

Abkürzungsverzeichnis

GRU Gated Recurrent Unit

MAE Mean Absolute Error

ONNX Open Neural Network Exchange

RMSE Root Mean Squared Error

WHO World Health Organization

1 Einleitung

Die Luftqualität in städtischen Gebieten stellt ein wachsendes Gesundheitsrisiko dar. Insbesondere Feinstaub (PM_{2.5}) und Stickstoffdioxid (NO₂) werden mit Atemwegserkrankungen, Herz-Kreislauf-Beschwerden und einer erhöhten Mortalität in Verbindung gebracht (Lelieveld et al., 2021; World Health Organization, 2016). Besonders gefährdet sind vulnerable Gruppen wie Kinder, ältere Menschen und Personen mit Vorerkrankungen (European Environment Agency, 2023). An stark befahrenen Straßen in der Nähe von Schulen und Wohngebieten können die Schadstoffkonzentrationen kurzfristig kritische Werte erreichen, ohne dass rechtzeitig Gegenmaßnahmen eingeleitet werden.

Klassische Umweltüberwachungssysteme erfassen zwar kontinuierlich Messdaten, leiten daraus jedoch keine unmittelbaren Handlungsempfehlungen ab. Die Daten werden zentral gesammelt und erst mit Verzögerung ausgewertet. Für zeitkritische Anwendungen wie die dynamische Verkehrssteuerung bei Schadstoffspitzen ist dieser Ansatz unzureichend. Edge AI bietet hier einen vielversprechenden Lösungsansatz: Durch die lokale Ausführung von Vorhersagemodellen direkt an der Messstation können Prognosen in Echtzeit erstellt und Maßnahmen ohne Umweg über zentrale Server ausgelöst werden.

Vor diesem Hintergrund wurde im Rahmen des Moduls „Projekt: Edge AI“ an der IU Internationalen Hochschule ein System entwickelt, das die Luftqualität an der Messstation Hamburg Habichtstraße vorhersagt. Die Station befindet sich an einer stark befahrenen Straße in unmittelbarer Nähe einer Schule und eignet sich daher besonders als Anwendungsfall. Das System prognostiziert sowohl die PM_{2.5}- als auch die NO₂-Konzentration eine Stunde im Voraus und leitet daraus ein dreistufiges Ampelsystem ab, das bei erhöhter Belastung automatisch Verkehrsmaßnahmen wie Geschwindigkeitsbegrenzungen oder Umleitungen auslösen kann.

Die methodische Vorgehensweise umfasst mehrere aufeinander aufbauende Schritte. Zunächst werden stündliche Messdaten der OpenAQ-Plattform vorverarbeitet und durch Feature Engineering angereichert. Anschließend werden vier Modelltypen, lineare Regression, Random Forest, Gradient Boosting und Gated Recurrent Unit (GRU), trainiert und verglichen. Durch eine systematische Feature-Importance-Analyse wird das Feature-Set von 28 auf 17 Merkmale reduziert. Die besten Modelle werden in das Open Neural Network Exchange (ONNX)-Format exportiert und hinsichtlich Inferenzzeit und Modellgröße für den Edge-Einsatz evaluiert. Abschließend wird eine Edge-Inference-Pipeline implementiert, die den gesamten Ablauf von der Sensorablesung bis zur Ampelentscheidung abbildet.

Das vorliegende Projekt gliedert sich wie folgt: Nach der Beschreibung der Projektumgebung und der Datengrundlage werden die Vorverarbeitung und das Modelltraining erläutert. Anschließend werden die Edge-Optimierung, das Alert-System und die Inference-Pipeline vorgestellt. Abschließend werden die Ergebnisse kritisch reflektiert und Verbesserungspotenziale aufgezeigt.

2 Hauptteil

2.1 Projektumgebung

Zu Beginn des Projekts wurde ein GitHub-Repository¹ angelegt, um eine nachvollziehbare Versionsverwaltung zu gewährleisten. Als Implementierungssprache wurde Python gewählt. Für die Reproduzierbarkeit wurde eine virtuelle Umgebung mit `venv` eingerichtet, in der alle Abhängigkeiten über eine `requirements.txt` installiert werden. Zu den zentralen Bibliotheken zählen `pandas` und `numpy` für die Datenverarbeitung, `scikit-learn` für die klassischen Machine-Learning-Modelle, `torch` (CPU-Version) sowie `onnxruntime` und `skl2onnx` für den ONNX-Export und die Edge-Inferenz.

Die gesamte Pipeline, von den Rohdaten bis zur Edge-Inference-Demo, lässt sich mit einem einzigen Befehl (`python run_all.py`) ausführen. Das Skript ruft alle Verarbeitungsschritte sequenziell auf und stellt sicher, dass jedes Teilskript im korrekten Arbeitsverzeichnis ausgeführt wird. Dadurch kann ein Tutor die Ergebnisse ohne manuelle Konfiguration reproduzieren.

2.2 Datengrundlage

Für das Training und die Evaluation der Modelle werden reale Luftqualitätsdaten benötigt. Im Vorfeld wurden drei Datenquellen evaluiert: das UCI Air Quality Dataset (De Vito, 2016) (globale Städte), das Umweltbundesamt (Umweltbundesamt, 2025) (zuverlässig, aber manueller Download) und die OpenAQ-Plattform (OpenAQ, 2025) (globale API mit deutschen Stationen). Die Wahl fiel auf OpenAQ, da die Plattform einen automatisierten API-Zugang bietet und die Aufgabenstellung diese Quelle explizit als Beispiel nennt.

Als Messstation wurde zunächst Hamburg Max-Brauer-Allee II (Straßenstation) in Betracht gezogen. Eine Datenanalyse ergab jedoch, dass für PM_{2.5} lediglich 486 Datenpunkte aus dem Dezember 2025 vorlagen, während PM₁₀, NO₂ und CO über alle zwölf Monate hinweg knapp 8 000 Messpunkte aufwiesen. Da ein vollständiges Jahr für die Abbildung saisonaler Muster notwendig ist, wurde stattdessen die Station Hamburg Habichtstraße (OpenAQ Location ID 3010) gewählt. Diese Straßenstation befindet sich in unmittelbarer Nähe einer Schule und liefert für alle vier Parameter, PM_{2.5}, PM₁₀, NO₂ und CO, jeweils rund 7 900 stündliche Messwerte über das gesamte Jahr 2025.

Der Datenabruf erfolgt über ein Python-Skript (`download_openAQ.py`), das die OpenAQ v3-API monatsweise abfragt und die Ergebnisse als CSV im Long-Format speichert. Die monatsweise Paginierung ist notwendig, um die API-Limits von 1 000 Datensätzen pro Anfrage einzuhalten. Das resultierende Rohdatenfile enthält rund 31 500 Messungen mit den Spalten `parameter`, `value`, `datetime_utc` und `datetime_local`.

¹<https://github.com/svenb23/EdgeAI>

2.3 Datenvorverarbeitung

Die Vorverarbeitung erfolgt in fünf aufeinander aufbauenden Schritten, die jeweils das Ergebnis des vorherigen Schritts einlesen und um neue Merkmale ergänzen.

Im ersten Schritt wird das Rohdatenfile vom Long-Format (eine Zeile pro Messung) in ein Wide-Format (eine Zeile pro Stunde, eine Spalte pro Schadstoff) überführt. Fehlende Werte werden linear interpoliert, wobei Lücken von maximal sechs Stunden geschlossen werden. Verbleibende Lücken werden entfernt.

Der zweite Schritt erzeugt zwölf zeitbasierte Merkmale. Neben den Rohwerten für Stunde, Wochentag und Monat werden binäre Indikatoren für Wochenende und Berufsverkehr (7–9 und 16–18 Uhr) erstellt. Zusätzlich werden Stunde, Monat und Wochentag zyklisch als Sinus- und Kosinuswerte kodiert, um die Periodizität für die Modelle abzubilden.

Im dritten Schritt werden sechs Lag-Features erzeugt, die die Schadstoffwerte der vorangegangenen Stunden als Prädiktoren bereitstellen. Für PM2.5 werden drei Lags (1h, 2h, 3h) erstellt, für NO2, CO und PM10 jeweils ein Lag von einer Stunde.

Der vierte Schritt berechnet gleitende Statistiken über verschiedene Zeitfenster. Für PM2.5 werden die rollierenden Mittelwerte über 3, 6 und 24 Stunden sowie die rollierende Standardabweichung über 3 Stunden berechnet. Diese vier Features erfassen lokale Trends und die kurzfristige Variabilität der Feinstaubkonzentration.

Im fünften Schritt werden drei schadstoffübergreifende Merkmale ergänzt: das Verhältnis PM2.5/PM10 als Indikator für den Anteil von Verbrennungspartikeln am Gesamteinstaub sowie die stündliche und dreistündliche Änderungsrate von PM2.5 ($\Delta 1h$, $\Delta 3h$).

Nach Abschluss aller fünf Schritte umfasst der Datensatz die vier Rohwerte sowie 25 engineerete Merkmale, insgesamt also 29 Spalten. Durch die Lag- und Rolling-Berechnungen gehen die ersten 23 Datenpunkte verloren, sodass der finale Datensatz rund 7 850 stündliche Beobachtungen enthält.

2.4 Modelltraining

2.4.1 Train/Test-Split

Für die Aufteilung des Datensatzes wird ein chronologischer 80/20-Split verwendet. Im Gegensatz zu einem zufälligen Split bewahrt diese Strategie die zeitliche Reihenfolge der Messdaten und verhindert Data Leakage, da das Modell ausschließlich auf vergangenen Daten trainiert und auf zukünftigen Daten evaluiert wird. Die ersten 80 % der Beobachtungen (ca. 6 280 Stunden) bilden den Trainingssatz, die letzten 20 % (ca. 1 570 Stunden) den Testsatz.

Als Zielvariablen werden die PM2.5- und die NO2-Konzentration jeweils eine Stunde in die Zukunft

verschoben. Der Wert von `target_pm25_1h` zum Zeitpunkt t entspricht somit dem tatsächlichen PM2.5-Wert zum Zeitpunkt $t + 1$. Analog wird `target_no2_1h` gebildet. Durch diese Verschiebung lernen die Modelle, aus den aktuellen Messwerten und den engineernden Merkmalen die Schadstoffbelastung der nächsten Stunde vorherzusagen. Die letzte Zeile des Datensatzes, für die kein Zielwert existiert, wird entfernt. Als Eingabemerkmale dienen alle 28 Features aus der Vorverarbeitung.

2.4.2 PM2.5-Modelle

Für die Vorhersage der PM2.5-Konzentration werden vier Modelltypen trainiert, die sich in Komplexität und Modellierungsansatz unterscheiden.

Die lineare Regression dient als Baseline-Modell. Sie wird ohne weitere Hyperparameter mit den Standardeinstellungen von scikit-learn trainiert und liefert einen ersten Referenzwert für die erreichbare Vorhersagegenauigkeit.

Das Random-Forest-Modell verwendet 100 Entscheidungsbäume mit einer maximalen Tiefe von 15. Durch die Begrenzung der Baumtiefe wird Overfitting reduziert, während die Ensemble-Methode dennoch nichtlineare Zusammenhänge erfassen kann. Die Berechnung erfolgt parallelisiert über alle verfügbaren CPU-Kerne.

Das Gradient-Boosting-Modell baut sequenziell 200 Bäume mit einer maximalen Tiefe von 5 und einer Lernrate von 0,1 auf. Die geringere Baumtiefe im Vergleich zum Random Forest kompensiert das Boosting-Verfahren durch die höhere Anzahl an Iterationen. Die Lernrate steuert den Beitrag jedes einzelnen Baumes zum Gesamtmodell.

Das GRU-Netzwerk ist das einzige sequenzielle Modell im Vergleich. Es verarbeitet Eingabesequenzen von sechs aufeinanderfolgenden Stunden, um die Konzentration der siebten Stunde vorherzusagen. Die Architektur besteht aus einer GRU-Schicht mit 32 Hidden Units, gefolgt von einer linearen Ausgabeschicht. Vor dem Training werden alle Features mit einem StandardScaler normalisiert. Das Modell wird über 30 Epochen mit dem Adam-Optimizer (Lernrate 0,001) und einer Batch-Größe von 64 trainiert. Als Verlustfunktion wird der Mean Squared Error verwendet.

Alle vier Modelle werden auf dem Testsatz anhand von Mean Absolute Error (MAE), Root Mean Squared Error (RMSE) und R^2 evaluiert. Die klassischen Modelle werden als Pickle-Dateien gespeichert, das GRU-Modell als PyTorch-State-Dict zusammen mit dem zugehörigen Scaler.

2.4.3 NO2-Modelle

Für die Vorhersage der NO2-Konzentration werden dieselben vier Modellarchitekturen mit identischen Hyperparametern trainiert. Das Eingabe-Feature-Set bleibt unverändert bei 28 Merkmalen, lediglich die

Zielvariable wechselt von `target_pm25_1h` zu `target_no2_1h`. Diese einheitliche Konfiguration ermöglicht einen direkten Vergleich der Modellleistung zwischen beiden Schadstoffen und zeigt, wie gut die PM2.5-orientierten Lag- und Rolling-Features auch für die NO2-Vorhersage geeignet sind.

2.5 Feature Importance und Feature-Reduktion

Um das Feature-Set für den Edge-Einsatz zu reduzieren, wird eine systematische Feature-Importance-Analyse durchgeführt. Dabei kommen vier Methoden zum Einsatz: die absolute Pearson-Korrelation mit der jeweiligen Zielvariable, die Mean Decrease in Impurity (MDI) des Random-Forest-Modells, die MDI des Gradient-Boosting-Modells sowie die Permutation Importance auf dem Testsatz (zehn Wiederholungen, Gradient-Boosting-Modell). Jede Methode erzeugt ein Ranking der 28 Features von 1 (wichtigstes) bis 28 (unwichtigstes). Aus den vier Einzelrankings wird der Durchschnittsrang berechnet (siehe Table 1 im Anhang).

Für PM2.5 dominieren die physikalisch-chemischen Merkmale: Der aktuelle PM2.5-Wert (Durchschnittsrang 1,0), die stündliche Änderungsrate (6,5), der 24-Stunden-Mittelwert (6,75) und der erste Lag (6,75) belegen die vorderen Plätze. Die elf zeitbasierten Features (Stunde, Wochentag, Monat jeweils als Rohwert sowie Sinus- und Kosinuskodierung, dazu die binären Indikatoren für Wochenende und Berufsverkehr) weisen durchweg Durchschnittsränge von 19 oder höher auf und tragen kaum zur Vorhersage bei.

Für NO2 ergibt sich ein differenzierteres Bild. Der aktuelle NO2-Wert (Rang 1,0) und der zugehörige Lag (3,0) sind erwartungsgemäß dominant. Einige zeitbasierte Features wie der Kosinus der Stunde (3,75) erreichen hier deutlich höhere Ränge als bei PM2.5, was auf ausgeprägtere tageszeitliche Muster der NO2-Konzentration hindeutet. Dennoch wird für beide Schadstoffe dasselbe reduzierte Feature-Set verwendet, um auf dem Edge-Gerät eine einheitliche Feature-Berechnung zu ermöglichen.

Die Reduktion von 28 auf 17 Features (39 % Reduktion) entfernt ausschließlich die elf zeitbasierten Merkmale. Das reduzierte Set umfasst die vier Rohwerte (PM2.5, PM10, NO2, CO), sechs Lag-Features, vier Rolling-Statistiken sowie drei schadstoffübergreifende Merkmale. Anschließend werden alle vier Modelle für beide Schadstoffe mit dem reduzierten Feature-Set neu trainiert und mit den Baseline-Ergebnissen (28 Features) verglichen (siehe Tables 2 and 3 im Anhang).

Für PM2.5 zeigt der Vergleich, dass die Feature-Reduktion die Vorhersagequalität nicht verschlechtert. Die lineare Regression verbessert sich sogar leicht (MAE 0,914 statt 0,929), und auch das GRU-Netzwerk profitiert deutlich von der Reduktion (MAE 1,544 statt 1,743). Dies deutet darauf hin, dass die entfernten zeitbasierten Features für PM2.5 eher Rauschen als Information beigetragen haben.

Für NO2 zeigt sich hingegen eine leichte Verschlechterung, da die tageszeitlichen Features hier relevanter sind. Der MAE der linearen Regression steigt von 4,019 auf 4,199, der des Random Forest von 3,903 auf 4,223. Die Einbußen bleiben jedoch moderat und rechtfertigen den Vorteil eines einheitlichen,

schlankere Feature-Sets für den Edge-Einsatz.

2.6 Edge-Optimierung

2.6.1 ONNX-Export

Für den Edge-Einsatz werden die besten Modelle in das ONNX-Format exportiert. Da die lineare Regression bei PM2.5 das beste Ergebnis liefert und gleichzeitig die geringste Komplexität aufweist, wird sie für beide Zielvariablen als Edge-Modell gewählt. Der Export erfolgt mit der Bibliothek `skl2onnx`, die scikit-learn-Modelle direkt in ONNX-Graphen überführt. Die Eingabe wird als Float-Tensor mit 17 Features definiert. Nach dem Export wird jedes Modell mit `onnxruntime` verifiziert, indem die Vorhersagen auf dem Testsatz mit den Ergebnissen des Originalmodells verglichen werden. Beide Modelle erreichen identische MAE-Werte (PM2.5: 0,914; NO2: 4,199) und bestätigen damit die korrekte Konvertierung.

2.6.2 Benchmark

Die exportierten ONNX-Modelle werden hinsichtlich Modellgröße und Inferenzzeit evaluiert. Beide Modelle belegen jeweils nur 0,3 KB Speicherplatz, was selbst für stark speicherbeschränkte Mikrocontroller geeignet ist. Die Inferenzzeit wird über 100 Durchläufe gemittelt. Eine einzelne Vorhersage benötigt unter 0,02 ms, eine Batch-Inferenz über den gesamten Testsatz (ca. 1 570 Datenpunkte) unter 0,1 ms. Diese Werte liegen weit unter den Anforderungen einer stündlichen Vorhersage und belegen die Echtzeitfähigkeit des Systems.

2.7 Alert-System

2.8 Edge-Inference-Pipeline

3 Fazit

3.1 Zielerreichung und Projektergebnisse

3.2 Kritische Reflexion

3.3 Verbesserungspotenziale

3.4 Ausblick

Projektrepository

Der vollständige Quellcode ist im GitHub-Repository verfügbar: <https://github.com/svenb23/EdgeAI>

Literatur

- De Vito, S. (2016). *Air Quality*. <https://doi.org/10.24432/C59K5F>
- European Environment Agency. (2023, 24. April). *Air pollution and children's health*. <https://www.eea.europa.eu/en/analysis/publications/air-pollution-and-childrens-health>
- Lelieveld, J., Hahad, O., Daiber, A., & Münzel, T. (2021). Luftverschmutzung und Herz-Kreislauf-Erkrankungen. *Aktuelle Kardiologie*, 10(06), 510–515. <https://doi.org/10.1055/a-1546-7355>
- OpenAQ. (2025). *OpenAQ – Open Air Quality Data*. <https://openaq.org/>
- Umweltbundesamt. (2025). *Aktuelle Luftdaten*. <https://www.umweltbundesamt.de/daten/luft/luftdaten>
- World Health Organization. (2016, 12. Mai). *Air pollution levels rising in many of the world's poorest cities*. <https://www.who.int/news/item/12-05-2016-air-pollution-levels-rising-in-many-of-the-world-s-poorest-cities>

Verzeichnis der Anhänge

Anhang

Tabelle 1: Feature-Importance-Ranking für PM2.5 (Durchschnittsrang aus vier Methoden)

Rang	Feature	Korr.	RF MDI	GB MDI	Perm.	Ø Rang
1	pm25	1	1	1	1	1,00
2	pm25_diff_1h	20	2	2	2	6,50
3	pm25_roll_mean_24h	9	3	3	12	6,75
4	pm25_lag_1	3	5	16	3	6,75
5	pm25_roll_mean_3h	2	4	14	7	6,75
6	pm10_lag_1	8	7	10	5	7,50
7	pm25_roll_mean_6h	4	14	4	11	8,25
8	pm10	7	12	12	4	8,75
9	co	11	9	9	8	9,25
10	pm25_roll_std_3h	15	8	6	10	9,75
11	no2	17	6	5	14	10,50
12	pm25_lag_2	5	16	15	6	10,50
13	pm25_pm10_ratio	10	11	7	15	10,75
14	co_lag_1	12	13	11	9	11,25
15	pm25_diff_3h	19	10	8	13	12,50
16	no2_lag_1	18	15	13	18	16,00
17	pm25_lag_3	6	20	22	21	17,25
18	hour_cos	25	17	17	17	19,00
19	hour	27	19	18	16	20,00
20	hour_sin	24	18	20	19	20,25
21	month_sin	13	23	23	23	20,50
22	month	14	22	21	25	20,50
23	dow_sin	22	21	19	20	20,50
24	month_cos	16	24	24	27	22,75
25	day_of_week	21	25	26	22	23,50
26	dow_cos	23	26	25	28	25,50
27	is_rushhour	28	27	27	24	26,50
28	is_weekend	26	28	28	26	27,00

Korr. = Absolute Pearson-Korrelation, RF MDI = Random Forest Mean Decrease in Impurity, GB MDI = Gradient Boosting MDI, Perm. = Permutation Importance. Die oberen 17 Features (über dem Strich) bilden das reduzierte Feature-Set.

Tabelle 2: Modellvergleich PM2.5: Baseline (28 Features) vs. reduziertes Set (17 Features)

Modell	MAE			RMSE			R^2		
	28	17	Diff.	28	17	Diff.	28	17	Diff.
Linear Regression	0,929	0,914	−0,015	1,537	1,529	−0,008	0,930	0,931	+0,001
Random Forest	0,981	0,977	−0,004	1,578	1,581	+0,003	0,926	0,926	±0,000
Gradient Boosting	0,990	0,975	−0,015	1,579	1,634	+0,055	0,926	0,921	−0,005
GRU	1,743	1,544	−0,199	2,493	2,302	−0,191	0,817	0,844	+0,027

Tabelle 3: Modellvergleich NO2: Baseline (28 Features) vs. reduziertes Set (17 Features)

Modell	MAE			RMSE			R^2		
	28	17	Diff.	28	17	Diff.	28	17	Diff.
Linear Regression	4,019	4,199	+0,180	5,394	5,604	+0,210	0,880	0,871	−0,009
Random Forest	3,903	4,223	+0,320	5,200	5,648	+0,448	0,889	0,869	−0,020
Gradient Boosting	4,015	4,285	+0,270	5,368	5,757	+0,389	0,881	0,864	−0,018
GRU	5,989	6,756	+0,767	7,824	8,819	+0,995	0,749	0,681	−0,068